THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

SCIENCE

2023 西南大学
Coursebook

DATA ANALYSIS

STATS 201

Table of Contents

Introduction to R for 20x . . . iii

Chapter 1: Getting started with linear regression **1**

Chapter 2: The basics of simple linear regression **13**
How to write **Method and Assumption Checks** and **Executive Summary** sections . . . 28
Case Study 2.1 . . . 29
Case Study 2.2 . . . 31
Case Study 2.3 . . . 34

Chapter 3: Equivalence of the null linear model and the one-sample t-test **37**
Case Study 3.1 . . . 47

Chapter 4: Fitting curves with the linear model **49**
Case Study 4.1 . . . 55

Chapter 5: Linear models with a 2-level categorical (factor) explanatory variable **57**
Case Study 5.1 . . . 66

Chapter 6: Multiplicative linear models **68**
Case Study 6.1 . . . 84
Case Study 6.2 . . . 87
Case Study 6.3 . . . 90
Case Study 6.4 . . . 92

Chapter 7: Power law linear models **95**
Case Study 7.1 . . . 102

Chapter 8: Linear models with both numeric and factor explanatory variables - Part 1: The interaction model **105**
Case Study 8.1 . . . 117
Case Study 8.2 . . . 120

Chapter 9: Linear models with both numeric and factor explanatory variables - Part 2: The model without interaction **123**
Case Study 9.1 . . . 133
Case Study 9.2 . . . 138
Case Study 9.3 . . . 142
Case Study 9.4 . . . 145

Chapter 10: Multiple linear regression models **151**
Case Study 10.1 . . . 164
Case Study 10.2 . . . 169

Chapter 11: Linear models with a single factor explanatory variable having three or more levels (One-way analysis of variance) **173**
Case Study 11.1 . . . 182
Case Study 11.2 . . . 185

Chapter 12: Linear models with two explanatory factor variables (Two-way analysis of variance) **187**
Case Study 12.1 . . . 200
Case Study 12.2 . . . 203
Case Study 12.3 . . . 206

Chapter 13: Modelling count data using the Poisson distribution **210**

Chapter 14: Poisson modelling of count data: Two examples **224**
Case Study 14.1 . . . 230
Case Study 14.2 . . . 233

Chapter 15: Modelling proportion data using the binomial distribution **235**
Case Study 15.1 . . . 253
Case Study 15.2 . . . 255

Chapter 16: Analysis of contingency tables **259**
Case Study 16.1 272

Handout R: Introduction to R for 20x

Russell Millar

R

The scatter plot shows a linear trend.

Development of the R language began in the early 1990's by two lecturers at the University of Auckland, Robert Gentleman and Ross Ihaka. The initial form of R was loosely based on a commercial language called S that originated at AT&T Bell Laboratories in the mid 1970's.

The motivation for R was to provide an open source language that was free for all to use and to extend, and one of its first uses was for the teaching of STATS 20x.

From its humble beginnings, R is now the most widely used statistical language in the world. Moreover, in 2015, R became the 6th most widely used programming language (behind Java, C, C++, Python and C#).

R, RStudio, and R markdown

In the first week of 20x you may encounter some very unfamiliar looking programming code. Don't panic - there is plenty of help available and your lecturer will be careful to explain what is going on.

- R is the underlying language you will be running
- R markdown is an R extension that enables you to create documents (i.e., your assignments)
- RStudio is the IDE (integrated development environment) that you will use to run R and R markdown.

Tutorials will be provided in a computer lab during the first week of classes - be sure to check Canvas for times. Please take advantage of these tutorials as they have been designed to help you breeze through the more technical parts of your assignments.

Note that there are also a variety of Assistance Room times for 20x students throughout the entire semester. The Assistant Room demonstrators are always willing to help students better understand the more technical side of this course and can help out with any problems or questions you have regarding R.

There are also plenty of online resources, e.g., See https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf

Assignments in 20x

You will be using R markdown to write your assignments. The great advantage of R markdown is that it automatically inserts the output of your statistical analyses into your homework document. **Copy and paste is now obsolete.**

This document was created by using R markdown to process the contents of file H0_R.Rmd. Your lecturer will show you how.

R markdown

If you look at file H0_R.Rmd you will see that some of the text has been interpreted as formatting. For example, the double hash ## is used at the start of a line to get a section header that uses the "Header 2" style.

In the next sections we see the use of R Chunks for running R commands, and how to execute those commands.

Writing and running R Chunks

- A code chunk begins with ```` ```{r} ````
- A code chunk terminates with ```` ``` ````

Anything inside the chunk will be interpreted as R code.

You can run selected lines or chunks of code in R using the Run Icon on the top right corner of the RStudio window, or alternatively the corresponding hotkeys. The R code and its output will automatically be inserted into your document.

The document is produced using a process called knitting, because it knits together the text and R code/output into a single document. The knit Icon at the top of the RStudio window lets you choose whether to produce the document in HTML, Word or PDF format.
Try Knitting this file now as a Word document.

Some example R chunks

This first example simply creates two numeric values, `y` and `z`.

```
y = 3
y
```

```
## [1] 3
```

```
z = y * 4
z
```

```
## [1] 12
```

Note that the lines of R output are prefixed by `##` (this is an option that can be changed).

It is important to be aware that R is an object-oriented language and everything that is created within an R session is some form of object. We refer to `y` and `z` as "objects'. In this case they are numeric objects.

Following are a few more commands. Can you figure out what each one does?

```
w = c(1, 2, 3, 4, 5, 6, 7, 8, 9)
w
```

```
## [1] 1 2 3 4 5 6 7 8 9
```

```
X = matrix(w, nrow = 3, ncol = 3, byrow = T)
X
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
```

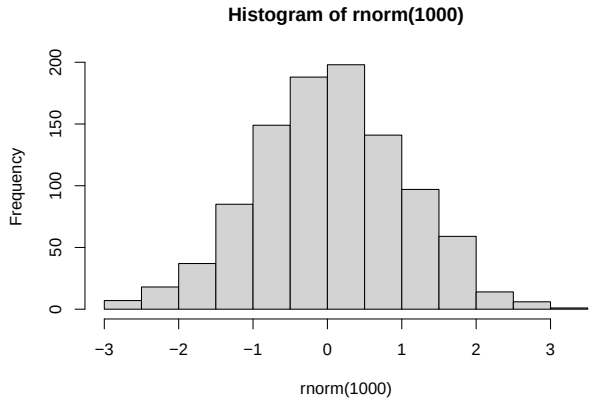
In the first line of this chunk we have used the `c` function, to create vector `w` from its arguments.

In the second line we have used the `matrix` function to create matrix `X` using the values in vector `w`.

R graphics

As you will discover, R is very good at producing plots. For example, here is a histrogram of one thousand randomly generated values from a normal distribution

```
#Using the hist function to draw a histogram
hist(rnorm(1000))
```




```
summary(rnorm(1000))
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -3.486853 -0.682714  0.008241 -0.003278  0.669437  3.420614
```

Note the use of `#` within the chunk to add a comment to the R code.

Reading data from a file

We will use the `read.table` function to input data from a data file. It is easiest if the data file is in the same folder as your R markdown file.

```
df = read.table("example.txt",header=T)
df
```

```
##   Height Weight      BMI
## 1   1.83     90 26.87450
## 2   1.72     89 30.08383
## 3   1.80     72 22.22222
```

Take a look at the `example.txt` file. The `header = T` (short for `TRUE`) argument tells R that the variable names `Height`, `Weight` and `BMI` are on the first line of this file. This is how data files will typically be provided in this course.

The data in `example.txt` have been saved in an object called `df`, which is a dataframe object.

Writing equations

R markdown can also write equations. Here is one you might recognize, $\bar{y} = \sum y_i/n$.

Challenges

The biggest challenge is to deal with the error messages that you will get from R and R markdown. They are often not very informative - though they typically give the line number in your R markdown file where the error occurred, and that is a good place to start looking.

Also, be aware that R is case sensitive, so `df` is different from `Df`.

- The R command `df` will print out the dataframe created above.
- The R command `Df` will cause an error and this document will not be produced.

Your lecturer will do his/her best to demonstrate a range of common errors that can occur!

Expect to see lots of error messages when you work on Assignment 1 - start this assignment early

Chapter 1:
Getting started with linear regression

STATS 201/8

University of Auckland

Learning outcomes

In this chapter you will learn about:

- Getting started using R with an example that analyses data from a previous class of STATS 20x students
- Getting data into R – creating a dataframe
- Working with dataframes in R
- How to fit straight lines to your data – the simple linear model
- How to predict using the fitted model
- How good is the fitted model for prediction? – R^2
- Some technical asides and relevant R-code for this section.

Section 1.1
A motivating example - previous 20x students

Example – Former STATS 20x Students

Over the next few weeks we will repeatedly make use of a dataset we collected from previous students who sat STATS 20x.

In this chapter we will examine a simple straight line model to see if a student's test mark can explain and predict their final exam mark.

Data collected from former students in 20x

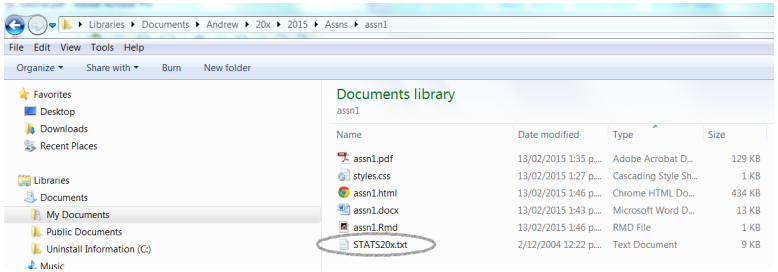
`Exam`	the student's final exam mark (out of 100)
`Degree`	the degree the student is enrolled for: Arts = BA, Commerce = BCom, Science = BSc, Other = Other
`Gender`	the student's gender: Female or Male
`Attend`	whether the student attended lectures regularly: No or Yes
`Assign`	the student's Assignment mark (out of 20)
`Test`	the student's midterm mark (out of 20)
`Stage1`	the student's grade for Stage 1 Statistics: A, B or C
`Years.Since`	the number of years since the student passed Stage 1 Statistics
`Repeat`	whether the student is repeating the course: No or Yes.

Note: In this Chapter we'll be using only `Test` to predict `Exam`. What other variables in the dataset might be useful for predicting `Exam`?

Section 1.2
Getting data into R – creating a dataframe

How do we import this data into R?

The original data are stored in a text file called `STATS20x.txt` in a suitable folder.



Text files often have `.txt` appended to their name. The letters after the full stop in the file name are sometimes called a file-name *extension*, and are often used by the operating system (Windows, Mac OS X, Linux) to decide which application/programme should be used to open the file.

What do the data look like?

As data analysts we love our data to be in 'rectangular' (i.e., matrix or array) form. Our raw data looks like this in the Notepad text editor.

Grade	Pass	Exam	Degree	Gender	Attend	Assign	Test
C	Yes	42	BSc	Male	Yes	17.2	9.1
B	Yes	58	BCom	Female	Yes	17.2	13.6
A	Yes	81	Other	Female	Yes	17.2	14.5
A	Yes	86	Other	Female	Yes	19.6	19.1
D	No	35	Other	Male	No	8	8.2
A	Yes	72	BCom	Female	Yes	18.4	12.7
D	No	42	BSc	Female	Yes	14.4	7.3
D	No	25	BA	Male	No	8.8	10.9
C	Yes	36	BCom	Female	Yes	17.6	10.9
C	Yes	48	BCom	Female	Yes	12	9.1
D	No	29	BSc	Female	No	12.4	9.1
B	Yes	54	BSc	Female	Yes	18	14.5
C	Yes	49	BSc	Female	No	18	11.8
C	Yes	52	Other	Female	No	12	15.5
D	No	28	BCom	Female	No	16.4	10
D	No	34	BCom	Female	Yes	16	7.3
D	No	51	BCom	Male	Yes	5.6	10
A	Yes	81	BCom	Male	No	16.8	17.3
A	Yes	80	BA	Female	Yes	16.4	15.5

Note that the first row of this `.txt` file contains the variable (i.e. column) names. Each row represents a different student. The columns within each row are separated by a `TAB` character – i.e. the file is *`TAB` delimited*.

How do we import this data-file?

```
> ## Importing data into R
> Stats20x.df = read.table("Data/STATS20x.txt", header=TRUE)
```

Here we are importing the `STATS20x.txt` data from a folder called `Data` that resides within our "working directory". We are also telling `R` that the 1st line of data contains the variable names by using the argument `header = TRUE`.[1]

If the data file resides in the working directory then it is enough to use

```
> Stats20x.df = read.table("STATS20x.txt", header=TRUE)
```

You can set the working directory using the `Session` menu of RStudio.

Note: In `R` code, we preface comment lines with the `#` symbol.

[1] `T` can be used as an abbreviation for `TRUE`.

Section 1.3
Working with dataframes in R

Where are the data?

It is there all right![2] The data exist in the `R` session as a dataframe object with the name `Stats20x.df`.

You can use whatever name you want (within reason) for objects you create in your `R` session, so always try to use meaningful names that will help you remember what the object is. That is why we used the `.df` suffix for our dataframe.

To see the dataframe you just need to ask. For example:

```
> ## 1st 5 rows and 7 columns of data set Stats20x.df
> Stats20x.df[1:5,1:7]

  Grade Pass Exam Degree Gender Attend Assign
1     C  Yes   42    BSc   Male    Yes   17.2
2     B  Yes   58   BCom Female    Yes   17.2
3     A  Yes   81  Other Female    Yes   17.2
4     A  Yes   86  Other Female    Yes   19.6
5     D   No   35  Other   Male     No    8.0
```

[2] Provided that you have **Run** the code chunk – knitting does not save the objects created.

How do we obtain the dimensions of a dataframe?

What are the dimensions of the `Stats20x.df` dataframe?

```
> dim(Stats20x.df)

[1] 146  15
```

It consists of 146 rows (students) and 15 columns. Each column represents a recorded measurement, characteristic, or feature of the students – that is, a variable.

Note: `R` is case sensitive. For example, the dataframe `stats20x.df` does not exist, and so asking for its dimensions will give an error:

```
> dim(stats20x.df)

Error in try(dim(stats20x.df)) : object 'stats20x.df' not found
```

Summarising STATS 20x exam marks

You can ask for one variable at a time using the `$` sign after the dataframe name. For example, we type `Stats20x.df$Exam` to obtain the `Exam` variable.

Here we just look at the first 10 observations for brevity.

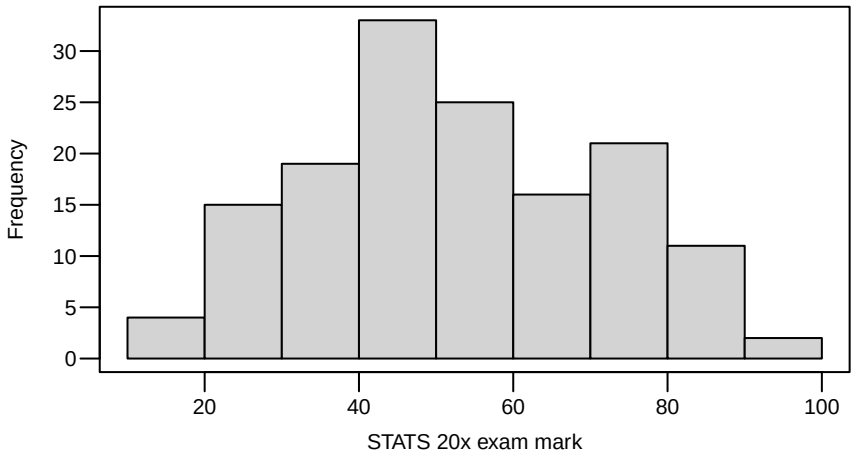
```
> Stats20x.df$Exam[1:10]

 [1] 42 58 81 86 35 72 42 25 36 48
```

If we want to look at the distribution of `Exam`, the visualisation tool of choice will be a histogram.

Summarising STATS 20x exam marks...

A histogram

```
> #Using xlab="STATS 20x exam score" to label the x-axis.
> hist(Stats20x.df$Exam, xlab="STATS 20x exam score")
```



Summarising STATS 20x exam marks...

Numerical summaries

Let us obtain some numerical summaries.

```
> summary(Stats20x.df$Exam)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  11.00   40.00   51.50   52.88   68.50   93.00
```

- So the lowest mark was 11 (`Min.` $\equiv$ minimum),
- the highest 93 (`Max.` $\equiv$ maximum),
- $\frac{1}{4}$ got 40 or less (`1st Qu.` $\equiv$ lower quartile),
- $\frac{1}{2}$ got less/more than 51.5 (`Med.` $\equiv$ median),
- $\frac{1}{4}$ got more than 68.5 (`3rd Qu.` $\equiv$ upper quartile),
- and the average (`Mean`) mark was about 53 (52.88) marks.

Section 1.4
How to fit straight lines to your data - a.k.a. the simple linear model

Example – Former STATS 20x students' exam marks

From the above summary we have some understanding about STATS 20x students' final exam marks. This is mildly interesting, but perhaps we could ask more interesting questions than this?

Here is one: Does my mid-term test mark relate to my final exam mark?

To be honest, I am pretty sure we know the answer to this question but let us answer this question, based on our data (not based on a strong opinion) and see if this suspected relationship is real or not.

The specific research question addressed in this chapter is **can we use mid-term test score to explain final exam score?**

Exam vs. Test marks

The particular variables of interest

`Exam` the student's exam mark (out of 100)
`Test` the student's Test mark (out of 20)

As scientists our first task is to determine whether or not there really is an association between `Test` and `Exam`.

If `Test` **is** found to be associated with `Exam` then the next step is to quantify the magnitude of that relationship.

Note: that our question really is about the 'typical' or average relationship. Some students may do well in the term test but not in the exam and vice-versa. We are really interested in the 'expected' effect of test on exam.

As our variables `Test` and `Exam` are both numerical we draw a scatter-plot to see what relationship is suggested (if any). A scatter-plot (sometimes called a *xy*-plot) has two axes, a horizontal *x*-axis and a vertical *y*-axis.

Exam vs. Test marks...

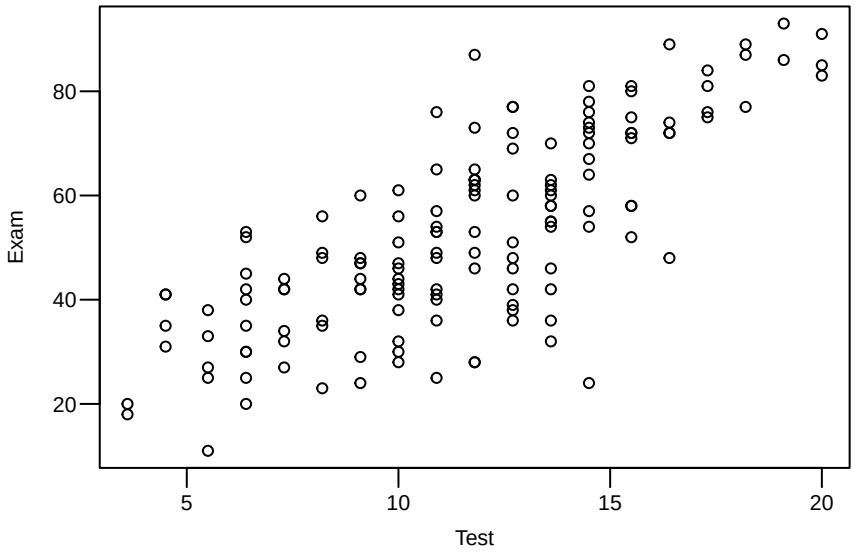
Let us look at these data

`Test` is the explanatory (or independent) variable and goes on the *x*-axis.
`Exam` is the response (or dependent) variable and goes on the *y*-axis.

```
> ## Importing data into R
> Stats20x.df = read.table("Data/STATS20x.txt", header=T)
> ## Plot the data
> plot(Exam ~ Test, data = Stats20x.df)
```


Exam vs. Test marks...

Let us look at these data...



Exam vs. Test marks...

Let us look at these data...

Note that the `plot(Exam ~ Test, data = Stats20x.df)` statement asks R to produce a plot suitable for showing how `Test` (x-axis variable) is related to `Exam` (y-axis variable).

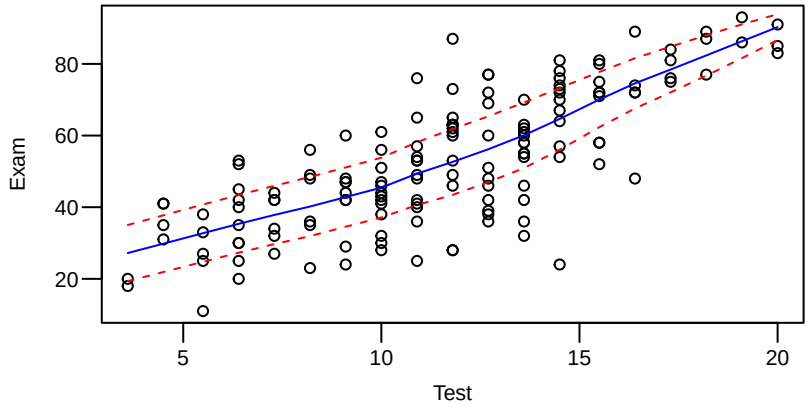
Looking at this plot, it is clear that there is some relationship, but there is also a lot of variability in exam score amongst students with the same test score, especially in the middle of the data.

As we are interested in the 'typical' exam score for a given test score we would like to visually see what the underlying trend is, and how much the exam scores vary (scatter) about that trend. It is helpful to run a smooth trend line through the scatter-plot.

Exam vs. Test marks...

Let us look at these data...

```
> ## Load the s20x library to use the trendscatter function
> library(s20x)
> trendscatter(Exam ~ Test, data = Stats20x.df)
```



Exam vs. Test marks...

Let us look at these data...

This is our first application of a 'bespoke'[3] function, `trendscatter`, made just for you. It is one of the many functions to be found when you load the `s20x` package. The `trendscatter` function computes a lowess[4] smoother to the data (the blue line) along with estimates of the variation about that line (the red lines).

The underlying trend here looks like a straight line. The variability looks roughly constant – except, perhaps, for those students who got high test marks.

Note: When we talk about "variability" it is in the vertical direction. That is, we are talking about the variability in exam score for a given fixed value of test score.

[3] (Of a computer program) written or adapted for a specific user or purpose: *"completely bespoke software systems" – Oxford Dictionary.*

[4] *lowess* stands for locally weighted scatterplot smoothing.

Section 1.5
How to predict using the fitted model?

Exam vs. Test marks...

Fitting a simple linear model

Pro Tip: If it looks like a straight line then fit a straight line.

A straight line is a simple (but not the simplest) form of a linear model.

```
> examtest.fit = lm(Exam ~ Test, data = Stats20x.df)
> summary(examtest.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   9.0845     3.2204   2.821  0.00547 **
Test          3.7859     0.2647  14.301  < 2e-16 ***
---
Residual standard error: 12.05 on 144 degrees of freedom
Multiple R-squared:  0.5868,Adjusted R-squared:  0.5839
F-statistic: 204.5 on 1 and 144 DF,  p-value: < 2.2e-16
```

Note that `examtest.fit` now resides in our R session as an `lm` *object*. An object, in the R language, is a structure than can contain data, variables, functions or other objects.

Exam vs. Test marks...

The simple linear model

Let us see what this means.

The summary output has a number of different components. Let us start with the `Estimate` column.

You will (or should!) recall from high school and/or STATS 10x that the equation for a straight line,

$$y = a + bx,$$

is described using two numbers (parameters), where:

- a is called the *intercept* since it is the value of y when the line intercepts the y-axis. Note that the y-axis corresponds to $x = 0$.
- b is called the *slope*. It tells us the rate of increase (or decrease) in y, for every additional unit increase in the value x.

Exam vs. Test marks...

The simple linear model

In the linear modelling context, it is the expected value of variable y, $\mathrm{E}[Y|x]$, that is assumed to follow a linear relationship with x. We read $\mathrm{E}[Y|x]$ as "the expected value of y given x." This is called a *conditional expectation*. It means that the expected value (or mean) of y will, or may, change depending on the value of x. That is, we assume

$$\mathrm{E}[Y|x] = a + bx$$

The first row of the above `summary(examtest.fit)` table is labelled `Intercept`, and in the `Estimate` column is the estimated value of a (i.e., the estimated value of $y =$ `Exam` when $x =$ `Test = 0`). The second row gives the estimated value of b, which is the increase in expected value of $y =$ `Exam` when $x =$ `Test` increases by one mark.

Exam vs. Test marks...

The simple linear model, general notation

In the previous slides we used a and b to denote the intercept and slope parameters. This is just a naming choice, and we could have used

$$\mathrm{E}[Y|x] = \theta + \gamma x$$

or

$$\mathrm{E}[Y|x] = c + dx,$$

or

$$\mathrm{E}[Y|x] = \beta_0 + \beta_1 x\ .$$

In this class we shall use the β_0, β_1, choice of parameter names since it is easiest to generalize to more complex models having many parameters.

Exam vs. Test marks...

The simple linear model

In the linear modelling context, it is the expected or typical value of variable y that follows a linear model. So, for the simple linear model:

$$\mathsf{E}[Y|x] = \beta_0 + \beta_1 x.$$

We can say that each y can be broken into what we expect to see plus a component that is random and, therefore, cannot be explained.[5]

We can write this as

$$y = \mathsf{E}[Y|x] + \varepsilon = \beta_0 + \beta_1 x + \varepsilon$$

where ε (epsilon) is random with expected value 0. This expresses how an observation varies around its expected or typical value. We will discuss these ideas in greater detail in the next chapter.

[5] This also applies to the more complex linear models we see later.

Exam vs. Test marks...

Let us look at this model

Let us extract the coefficients component from the fitted model object:

```
> coef(examtest.fit)

(Intercept)        Test
   9.084463    3.785924
```

You can see this corresponds to the `Estimate` column of the regression summary table.

We are saying that the 'best' fit of the data tells us that the predicted exam score for a student with test score `Test` is given by the following equation (to two decimal places):

$$\widehat{\texttt{Exam}} = 9.08 + 3.79 \times \texttt{Test}^{6}.$$

[6] We use the notation $\widehat{\texttt{Exam}}$ as this is an estimated value.

Exam vs. Test marks...

Let us look at this model...

So, if you have received 0 in the `Test`, then we estimate that you should be able (on average) to get 9.08 marks for just turning up to the exam and guessing your way through it.

Thereafter, for every extra mark you get in the the test (out of 20) your final exam score will increase (on average) by about 3.79 marks.

Here are some predictions for typical students who got test mark of 0, 10 or 20:

- `Test` = 0 then $\widehat{\texttt{Exam}} = 9.08 + 3.79 \times 0 = 9.08\%$ in the exam,
- `Test` = 10 then $\widehat{\texttt{Exam}} = 9.08 + 3.79 \times 10 = 46.94\%$ in the exam,
- `Test` = 20 then $\widehat{\texttt{Exam}} = 9.08 + 3.79 \times 20 = 84.80\%$ in the exam.

Exam vs. Test marks...

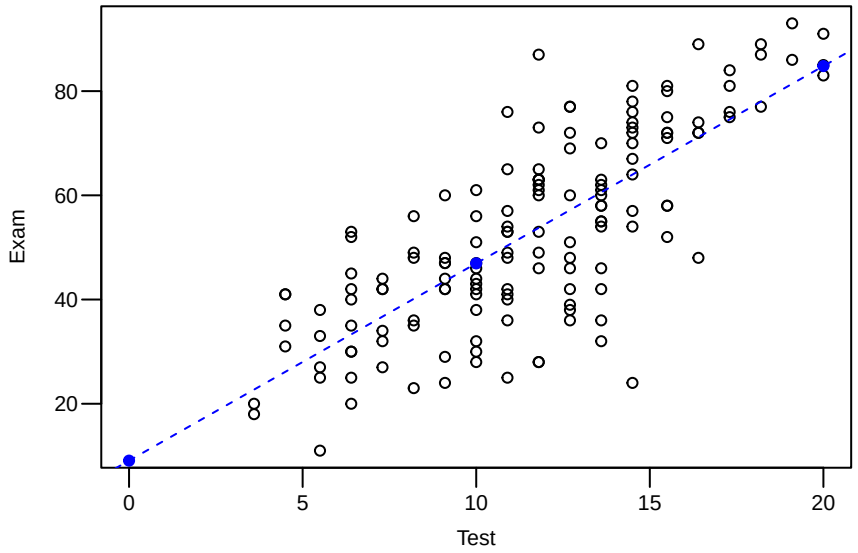
Let us look at this model...

Let us see how this model works out overall. Here we are plotting the 'best' estimated straight line that we obtained from fitting our model.

```
> plot(Exam ~ Test, data = Stats20x.df, xlim = c(0, 20))
> ## Add the lm estimated line to this plot
> abline(examtest.fit, lty = 2, col = "blue")
> ## Calculate predicted values for students who get test=0,10,20
> predns=predict(examtest.fit, newdata = data.frame(Test=c(0,10,20)))
> ## Add the predicted values to the plot
> points(c(0,10,20), predns,col = "blue", pch = 19)
```

Exam vs. Test marks...

Let us look at this model...



Section 1.6
How good is the fitted model for prediction? – R^2

Exam vs. Test marks...

The null model

This model is not too bad. It appears to do a reasonable job of describing the data and appears to make sense – but compared to what? How useful is the variable `Test` in explaining and predicting `Exam`?

We can fit a model that is even simpler than the straight line model. This is called the null[7] model and simply estimates the typical exam mark without an explanatory variable x. That is , $\mathrm{E}[Y|x] = \mathrm{E}[Y]$. Here is our null model for $y = Exam$:

$$\begin{aligned} y &= \mathrm{E}[Y|x] + \varepsilon \\ &= \mathrm{E}[Y] + \varepsilon \\ &= \beta_0 + \varepsilon \\ &\equiv \mu + \varepsilon \end{aligned}$$

That is, $x = Test$, say, is not related to exam mark.

[7] The null model is the model assumed by a one sample t-test, as seen in Chapter 3.

Exam vs. Test marks...

The null model...

Let us fit this *null* model:

```
> ## Null model
> examnull.fit=lm(Exam ~ 1, data = Stats20x.df)
> summary(examnull.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   52.877      1.546   34.21   <2e-16 ***
---
Residual standard error: 18.68 on 145 degrees of freedom
```

In comparison, the simple linear model `examtest.fit` gives us different predictions for different test marks. Naturally we would suspect this to be a better model for predicting *Exam* as we are pretty sure that *Test* explains *Exam* quite well.

Exam vs. Test marks...

The null model...

Our null model was $Exam = \beta_0 + \varepsilon$ where $\varepsilon \overset{iid}{\sim} N(0, \sigma^2_{Null})$[8] with the best estimate of $\mu = \beta_0$ being the sample mean, $\bar{y} = 52.88$.

Note that the estimate of the standard deviation (σ) of the random ε component was $\hat{\sigma}_{Null} = s_{Null} = 18.68$.[9]

This is also called the residual standard error.

[8] If we were were doing a one sample t-test we would use parameter name μ instead of the more general β_0.

[9] Here we use the hat notation to denote that $\hat{\sigma}_{Null}$ is an estimate of the unknown parameter σ_{Null}.

Exam vs. Test marks...

Null vs. simple linear model

Compare the null model to the simple linear model that used `Test` as an explanatory variable.

```
> ## Exam vs. Test model
> summary(examtest.fit)
```

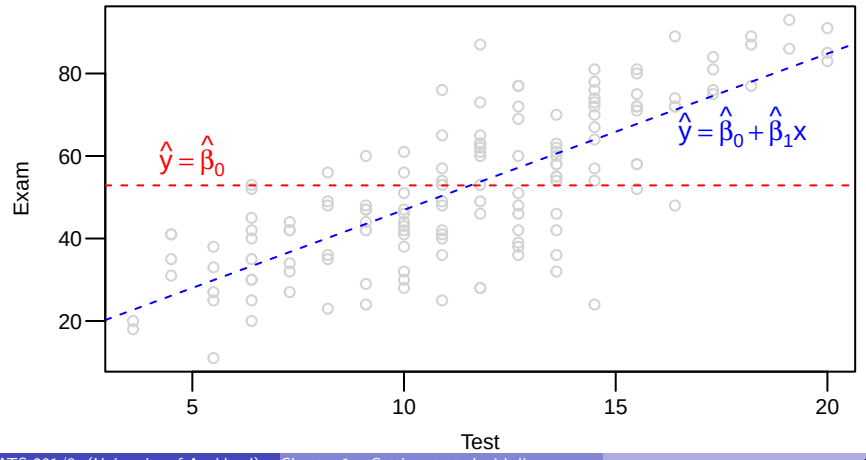
```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   9.0845     3.2204   2.821  0.00547 **
Test          3.7859     0.2647  14.301  < 2e-16 ***
---
Residual standard error: 12.05 on 144 degrees of freedom
Multiple R-squared:  0.5868,Adjusted R-squared:  0.5839
F-statistic: 204.5 on 1 and 144 DF,  p-value: < 2.2e-16
```

Note that the estimate (s_{Test}) of the standard deviation (σ_{Test}) of the random (ε) component has been reduced to $s_{Test} = 12.05$.

Exam vs. Test marks...

How good is this simple linear model?

If $y =$ `Exam` and $x =$ `Test`, then here are our fitted null (the red line) and linear (the blue line) models. We clearly see that the blue line does a far better job of describing this relationship than the red line.



Exam vs. Test marks...

Multiple R-squared

The overall variation in the null model is given by the sum of the squared residuals, denoted by SS_{Null}. In Chapter 2 we will see that $SS_{Null} = 50586$. SS_{Null} is more commonly known as the **total sum of squares**, SS_{Tot}.

The sum of squares of the residuals from the model using the variable `Test`, denoted here by SS_{Test}, is 20901.

The reduction in overall variation from the `Null` model to the `Test` model can be expressed as a proportion of SS_{Null}:

$$\frac{SS_{Null} - SS_{Test}}{SS_{Null}} = 1 - \frac{SS_{Test}}{SS_{Null}} = 1 - \frac{20901}{50586} \approx 0.59.$$

This statistic is usually called multiple **R-squared** (R^2) or the proportion of variation explained.

In the above output it is: `Multiple R-squared:  0.5868`

R-squared interpretation

We can say that $(100 \times R^2) = 59\%$ of the (total) variation in the exam mark is explained by using a straight line relationship (i.e., simple linear model) with test score.

So by using just one explanatory variable, $x = Test$, we can explain 59% of the variation that we observe in $y = Exam$. That is a pretty good return on one 'piece of information', the `Test` score.

You may have seen (from STATS 210 perhaps) that the estimates of σ_{Null} and σ_{Test} are obtained from the residual sums of squares: $s_{Null} = \sqrt{SS_{Null}/(n-1)}$ and $s_{Test} = \sqrt{SS_{Test}/(n-2)}$.

In the above calculation, n is the number of observations ($n = 146$ in this example), and $n - 1$ is the degrees of freedom of the null model and $n - 2$ is the degrees of freedom of the straight line model.

Word of caution

In practice we need to first check the assumptions of our model before we can safely use it for inference such as calculating confidence intervals, making predictions, etc.

This is the focus of the next Chapter.

Section 1.7
Some technical asides

Aside: Getting help on functions in R

In this first chapter we have already used several `R` functions. It is easy to forget the details (e.g., the arguments that the function needs), in which case you can get help about the function by simply preceding its name with a question mark.

For example, try

```
> ?summary
```

or

```
> ?lm
```

In RStudio this opens a sub-window for the help file. It can be useful to look at the examples (at the bottom of the help file) of a function's usage to get an idea of how it is used.

Unfortunately, the `R` help can be rather difficult to understand at first because of all the computer jargon it uses. There is plenty of other online help on most `R` functions that can be found through your favourite search engine.

Aside: Data types in R

There are several data types in R, but the three that we need to know about are *numerical*, *character* and *logical*.

We'll see in Chapter 5 that we can also use character data as an explanatory variable. In that chapter we use the character variable class attendance ("Yes" or "No") to explain exam mark. We regard attendance as a two-level factor variable – stay tuned...

```
> summary(Stats20x.df$Attend)

   Length     Class      Mode
      146 character character
```

```
> summary(as.factor(Stats20x.df$Attend))

 No Yes
 46 100
```


Section 1.8
Relevant R-code

Most of the R-code you need for this chapter

Creating a data frame by importing the text file Data/STATS20x.txt from the subfolder Data:

```
> Stats20x.df = read.table("Data/STATS20x.txt", header=TRUE)
```

Plotting a scatter plot of y (Exam) vs. x (Test):

```
> plot(Exam~Test, data=Stats20x.df)
```

and/or a trend-scatter-plot:

```
> trendscatter(Exam ~Test, data = Stats20x.df)
```

Fitting a linear (straight line) model and getting the estimated values:

```
> examtest.fit=lm(Exam~Test, data=Stats20x.df)
> summary(examtest.fit)
```

Adding your fitted/predicted blue-dashed line to the scatter-plot created above:

```
> abline(examtest.fit, lty = 2, col = "blue")
```

Chapter 2:
The basics of simple linear regression

STATS 201/8

University of Auckland

Learning outcomes

In this chapter you will learn about:

- Fitting the linear model by minimizing the sum of the squared residuals
- The standard assumptions of the linear model
- Model checks of these assumptions – independence, EOV and normality
- Checking for points of undue influence using `cooks20x`
- Making inference from your model (provided the assumptions check out)
- Confidence intervals for population-level inference
- Prediction intervals for individual-level inference
- A recipe for subsequent analyses and relevant `R`-code.

Section 2.1
Fitting the linear model by minimizing the sum of the squared residuals

The fitted model

Fitted values and residuals

Recall, our simple linear model is $y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$, where β_0, β_1 and ε_i are all unknown. After we fit this model to data, we can write

$$\begin{aligned} y_i &= \hat{y}_i + r_i \\ &= \left(\hat{\beta}_0 + \hat{\beta}_1 x_i\right) + r_i \end{aligned}$$

The two components in the above formulation are:

- The fitted (or predicted) value, $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$. This is the estimated value of $E[Y_i|x_i] = \beta_0 + \beta_1 x_i$.
- And what is left over, the **residual**, r_i, the estimated value of ε_i.

A natural question is: How are $\hat{\beta}_0$ and $\hat{\beta}_1$ estimated from the data???

We will answer this question in the next few slides in the context of our exam vs test mark example.

Exam vs. Test marks...

Fitted values and residuals...

First, let us examine one student in particular – student (observation) number $i = 21$.

```
> ## Student # 21
> Stats20x.df = read.table("Data/STATS20x.txt", header=TRUE)
> Stats20x.df[21,]

   Grade Pass Exam Degree Gender Attend Assign Test  B  C MC Colour Stage1
21     A  Yes   77    BSc Female     No   16.4 12.7 15 18 22  Green      A
   Years.Since Repeat
21          0.5     No

> examtest.fit = lm(Exam ~ Test, data = Stats20x.df)
```

She got 12.7 out of 20 for the `Test` and 77 for an exam mark. Again if we say $y =$ `Exam` and $x =$ `Test` then for her: $y_{21} = 77$ and $x_{21} = 12.7$.

Exam vs. Test marks...

Fitted values and residuals...

According to the fitted model her fitted/predicted value is

$$\hat{y}_{21} = \hat{\beta}_0 + \hat{\beta}_1 x_{21} = 9.08 + 3.79 \times 12.7 = 57.1657$$

so she seems to have done better than predicted! The residual value for her is:

$$r_{21} = y_{21} - \hat{y}_{21} = 77 - 57.1657 = 19.8343.$$

These calculations are done for all $i = 1, 2, \ldots, n = 146$ students.

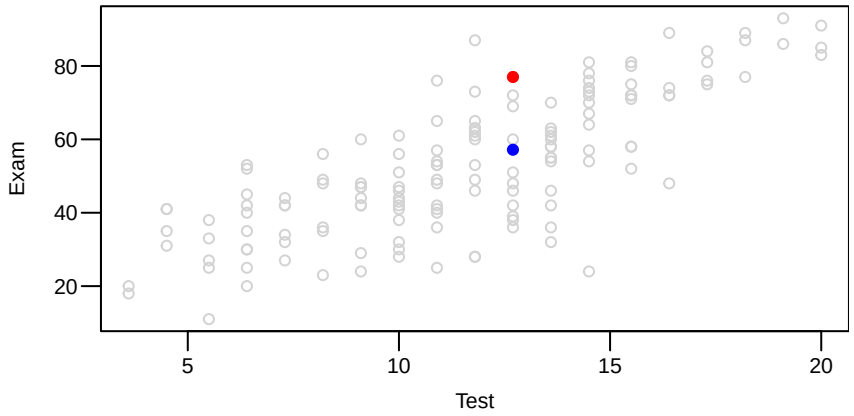
`R` did these calculations when we created the `lm` object `examtest.fit`. Here is how to extract this information for student $i = 21$:

```
> resid(examtest.fit)[21]

     21
19.8343

> fitted(examtest.fit)[21]

     21
57.1657
```


Exam vs. Test marks...

Fitted values and residuals...

Here is her observed value y_{21} - (red point) and her associated fitted[1] value $\hat{y}_{21}$ marked (blue point).

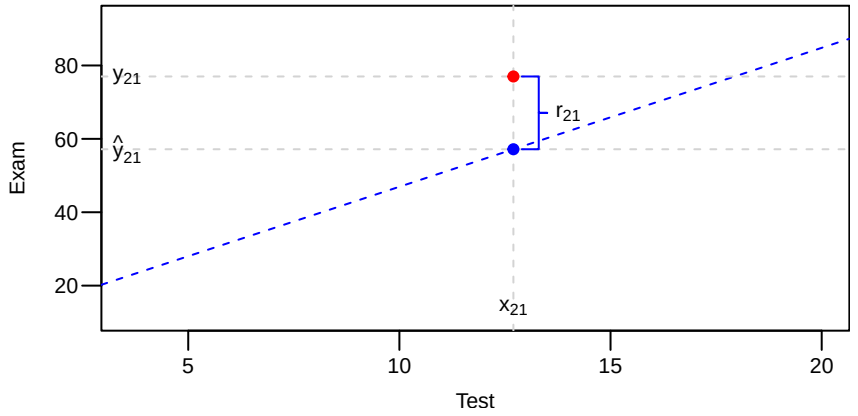


[1] We frequently use these expressions "fitted" values and "predicted" values interchangeably – with some exceptions – more later!

Exam vs. Test marks...

Fitted values and residuals...

In greater detail:



The residual is the vertical distance between the observed exam mark and the predicted exam mark.

Calculating $\hat{\beta}_0$ and $\hat{\beta}_1$

Least squares

The estimates, $\hat{\beta}_0$ and $\hat{\beta}_1$, are obtained as the values of β_0 and β_1 that minimise the least squares equation:

$$\sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2.$$

This minimized value is

$$\sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i)^2 = \sum_{i=1}^{n} r_i^2.$$

In effect this minimises the sum (over all observations) of the squared residuals. Hence the name **"least squares"**.

This can be solved using a bit of calculus and/or linear algebra taught in, say, MATHS 208. A bit more detail is also given in STATS 330.

Exam vs. Test marks...

Residual sum of squares and R^2.

Recall that we've already seen the sum of the squared residuals from the fitted model, $\sum_{i=1}^{n} r_i^2$. It is the residual sum of squares (RSS). It can be calculated using the R code

```
> sum(resid(examtest.fit)^2)

[1] 20901.08
```

Similarly, the RSS of the null model is the total sum of squares,

```
> sum(resid(lm(Exam~1,data=Stats20x.df))^2)

[1] 50585.78
```

Note that, compared to the null model, the RSS is reduced by 59% when test score is in the model. That is, the R^2 of `examtest.fit` is 0.59.

Section 2.2
The standard assumptions of the linear model

General assumptions of all linear models

In this chapter we will discuss the set of assumptions we make when using linear models for inference.

We will show how these assumptions relate specifically to the simple linear model we have seen so far.

However, whatever we learn here is also applicable to the other types of linear models that we will encounter.

In later chapters we will encounter situations where these assumptions cannot hold. However, we will find appropriate ways to still use a linear model.

General assumptions of all linear models...

The underlying assumptions

Let us state and examine our model a little more carefully.

Our simple linear regression model is

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \quad \varepsilon_i \overset{iid}{\sim} N(0, \sigma^2)$$

where each observation is indexed[2] with a subscript i and *iid* stands for **i**ndependently and **i**dentically **d**istributed.

The above model formula is saying that

$$\varepsilon_i = y_i - \beta_0 + \beta_1 x_i$$

is distributed according to

$$\varepsilon_i \overset{iid}{\sim} N(0, \sigma^2)$$

[2]The indexing variable i is simply a labelling device. So, in out test vs exam example where $y =$ `Exam` and $x =$ `Test` we are saying that each student (observation) has a label associated with them from $i = 1, 2, 3 \ldots, n = 146$.

General assumptions of all linear models...

The underlying assumptions...

The model assumptions are with regard to the random variability component

$$\varepsilon_i \overset{iid}{\sim} N(0, \sigma^2)$$

1. **Independent:** We require every observation to be independent of every other observation.[3] This requires that the data are collected in an appropriate manner, such as a simple random sample. In practice, achieving truely independent data can be challenging.
2. **Identically** distributed: The ε_i must all come from the same distribution, and this distribution must have mean of 0. All ε_i must necessarily have the same variance, which we refer to as the equality of variance, (**EOV**), assumption.
3. **Distributed** normally: The ε_i must be from a normal distribution, at least approximately.

[3]If the y_i are independent then so too are the ε_i.

Model assumptions

Exam vs. Test marks

Let's consider the underlying assumptions about $\varepsilon_i \overset{iid}{\sim} N(0, \sigma^2)$ in the context of our ongoing example.

In decreasing order of importance:

- **Independent:** It was an invigilated exam so the variabilities in the exam scores of students should all be independent.
- **Identically** distributed: If the **linear** relationship is reasonable then
  - The residuals[4] will be randomly scattered around 0.
  - The residuals should have roughly constant scatter, i.e., satisfy the equality Of Variance (EOV) requirement.
- **Distributed** normally: The residuals should be (at least roughly) normally distributed – only check this if the first two assumptions are validated.

[4]Recall, the residuals r_i are our estimates of ε_i

Very important aside!

If the linear model assumptions do not hold, it **does not** mean we cannot do anything.

When assumptions are not satisfied there are often techniques that we can use to transform our data and/or model so that the assumptions **will** be satisfied. We will see some of these techniques later in this course.

There are also other techniques that can "free up" some of the assumptions. For example, lack of independence in time series data is covered later in this course, and other techniques are seen in more advanced courses.

Section 2.3
Checks of model assumptions – independence, EOV and normality

EOV check: Plotting residuals vs fitted values

Let us check the **EOV** assumption. We are assuming that what cannot be explained has random constant scatter about zero regardless of the fitted value.

That is, for each fitted (or predicted) value we have a similar distribution for what cannot be explained. We can check this by looking at each fitted (predicted) value versus its associated residual value.

Our fitted model is:

$$y_i = \hat{y}_i + r_i = \left(\hat{\beta}_0 + \hat{\beta}_1 x_i\right) + r_i.$$

It is useful to draw a scatter plot with $\hat{y}_i$ on the horizontal axis and r_i on the vertical axis.

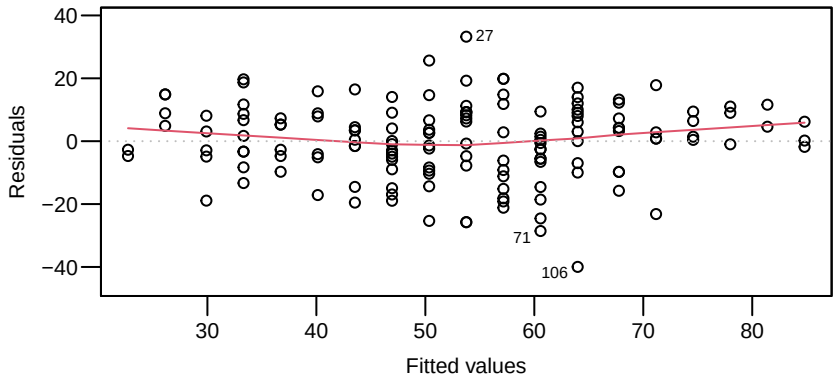
This is more generally known as a *residuals vs fitted values* plot.

Exam vs. Test marks. . .

EOV check: Plotting residuals vs fitted values. . .

Producing a scatter plot of residuals versus fitted values is easy[5].

```
> plot(examtest.fit, which=1)
```



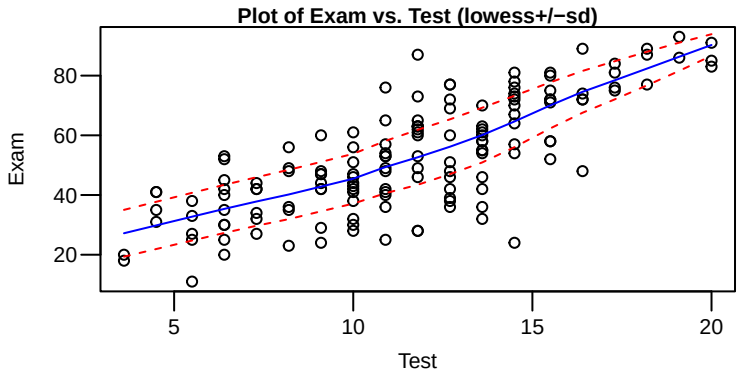
[5]You can also use the s20x function eovcheck, i.e., eovcheck(exam.fit)

Exam vs. Test marks. . .

EOV check: Plotting residuals vs fitted values. . .

The above residual plot exhibits a patternless band of random points - this is to be expected since the original scatter plot of the data had a straight line trend and reasonably constant scatter:

```
> trendscatter(Exam~Test, data =Stats20x.df)
```



Exam vs. Test marks...

EOV check: Plotting residuals vs fitted values...

The plot of residuals versus fitted values confirms reasonably constant scatter (variance).

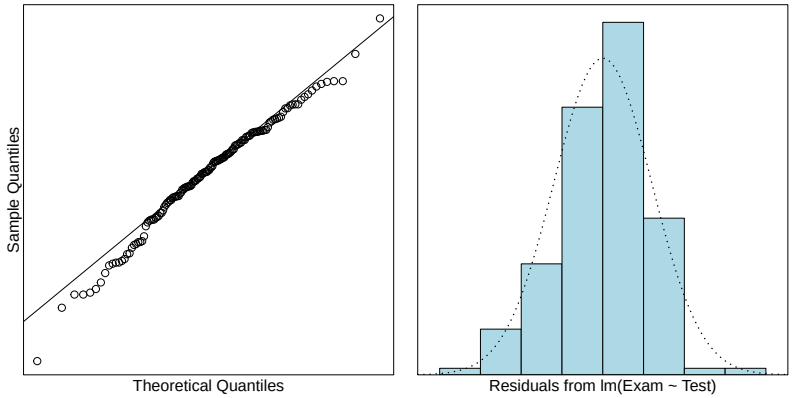
Note that at the high end of exam/test scores there are not that many students (unfortunately!) so the relative lack of spread is not surprising.

Exam vs. Test marks...

Normality of residuals

Having validated the **EOV** assumption, let us perform the normality check by examining the residuals. We have written a function in R called `normcheck` to do just that:

```
> normcheck(examtest.fit)
```



Exam vs. Test marks...

Normality of residuals

The histogram of the residuals (right plot) tells us we are okay since it closely matches a normal bell-shaped curve (shown by the dashed line).

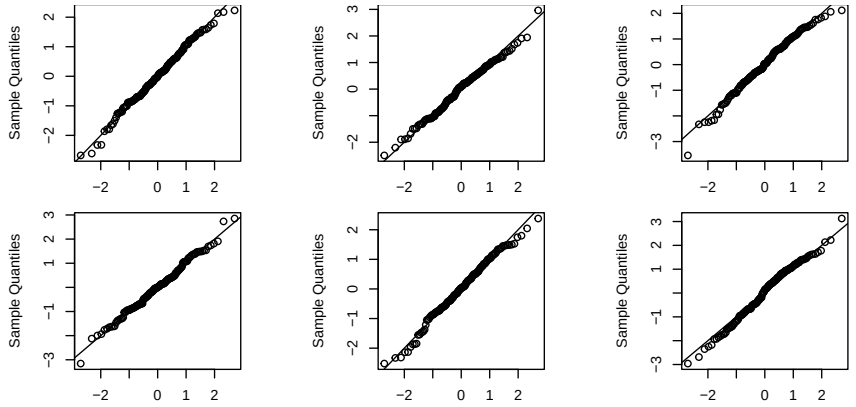
The normal quantile-quantile-plot (`qqplot`) in the left hand plot gives further insight. In this example, the 146 theoretical quantile values on the x-axis can be regarded as the values that would give a near perfect bell-shaped histogram (i.e., would best match the dashed bell-shaped curve in the right hand plot). Ideally, our residuals should match these theoretical quantiles and lie approximately on the straight line shown on the `qqplot`.

It can be difficult to judge if a qq-plot is sufficiently close to a straight line or not. Some intuition can be gained by seeing how the qq-plot looks when the assumptions are satisfied, as shown on the next slide. In comparison, the above qq-plot does look a little atypical, but not enough to cause major concerns.

Exam vs. Test marks...

qq-plot intuition

```
> for(i in 1:6) {
+   y=rnorm(146)
+   qqnorm(resid(lm(y~1)),main=""); abline(0,1) }
```



Exam vs. Test marks...

A further check: Influence

Before we rush in and conclude that all is well, there is one other concern that we should investigate, namely *influence*.[6]

Essentially, we need to check whether the fit is highly influenced by any single (or small group of) observation(s).

[6] This is not a model assumption per se, but is a prudent model check.

Section 2.4
Checking for points of undue influence using `cooks20x`

Points of undue influence

The fitted model is based on minimizing the sum of the squared residuals. Because of this, a single observation with a very large residual can have a big influence on the overall fit. We can be misled if we assume all is fine when one (or several) data points are unduly influential, and perhaps of questionable validity.

Moreover, a data point can also be influential on the model fit by having an extreme x value, since it can apply a lot of "torque" to the fit.

Exam vs. Test marks...

Points of undue influence...

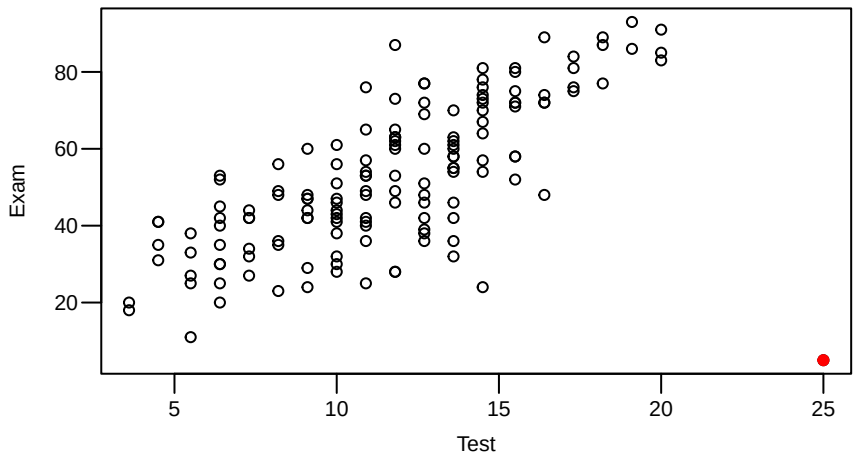
To illustrate: Imagine a student had mistakenly been awarded 25 out of 20 in their `Test`, yet obtained just 5 in the final exam.

Here is some tricky `R` code to add this incorrect data point to our dataframe:

```
> n=nrow(Stats20x.df)
> ## Add extra point by repeating last observation, n(=146)
> Stats20xnew.df=Stats20x.df[c(1:n,n),]
> ## Replace with new test /Exam #'s
> Stats20xnew.df[n+1,c("Test", "Exam")]=c(25,5)
> ## Plot our new dataset
> plot(Exam~ Test, data=Stats20xnew.df)
> ## Mark our new observation
> points(25,5,pch=19,col="red")
```

Exam vs. Test marks...

Points of undue influence...



The bogus student is shown on the plot as the red point!

Exam vs. Test marks...

Points of undue influence...

Here is the new model fitted to the altered data.

```
> examtest.fit2=lm(Exam~Test, data=Stats20xnew.df)
> coef(summary(examtest.fit2))
```

```
             Estimate Std. Error   t value     Pr(>|t|)
(Intercept) 15.237389  3.7171579  4.099204 6.875383e-05
Test         3.200551  0.3022693 10.588407 9.162607e-20
```

Compare this to the original fit:

```
> coef(summary(examtest.fit))
```

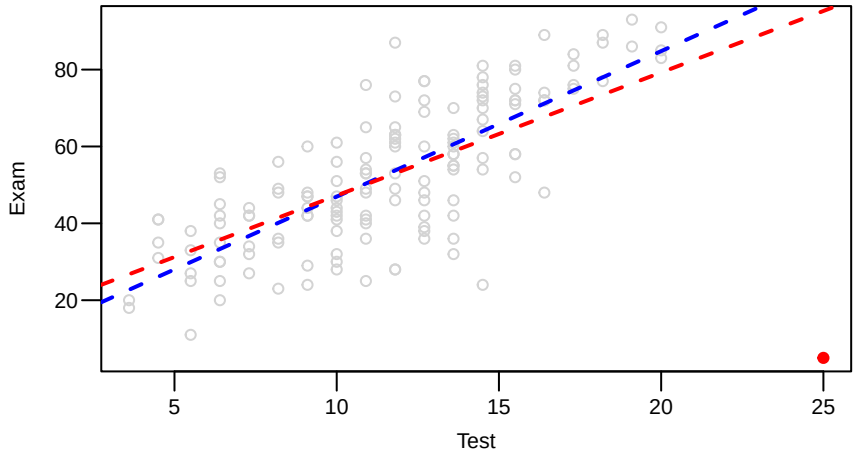
```
            Estimate Std. Error   t value     Pr(>|t|)
(Intercept) 9.084463  3.2204410  2.820876 5.465681e-03
Test        3.785924  0.2647333 14.300897 1.985079e-29
```

Note how $\hat{\beta}_1$ has changed from 3.79 to 3.20. This is a change of more than two standard errors (here, we are referring to the standard error from the original fit, 0.265).

Exam vs. Test marks...

Points of undue influence...

Here is the old model fitted line (blue) compared to this new model (red).



Exam vs. Test marks...

Points of undue influence...

How does this change predictions?

In this model we would make a prediction using the following estimates (to 2 d.p.). $\texttt{Exam} = 15.24 + 3.20 \times \texttt{Test}$ compared to the old model estimates: $\texttt{Exam} = 9.08 + 3.79 \times \texttt{Test}$

Our new vs. previous estimates are now (previous estimate in brackets):

- $\texttt{Test} = 0$ then $\texttt{Exam} = 15.24 + 3.20 \times 0 = 15.24$ (9.08),
- $\texttt{Test} = 10$ then $\texttt{Exam} = 15.24 + 3.20 \times 10 = 47.24$ (46.94),
- $\texttt{Test} = 20$ then $\texttt{Exam} = 15.24 + 3.20 \times 20 = 79.25$ (84.80).

Note that this has had a huge change for the extreme Test scores – for those who got 0 in the Test we predict a higher mark than before and those who got a 20 (out of 20) get a lower predicted mark than before.

Identifing points of undue influence

The addition of student 147 has drastically changed our fitted model and its predictions. This studend is so atypical of the other 146 students that we should exclude them from the analysis. In this case it is clearly a data entry mistake, but in practice it is not always so obvious.

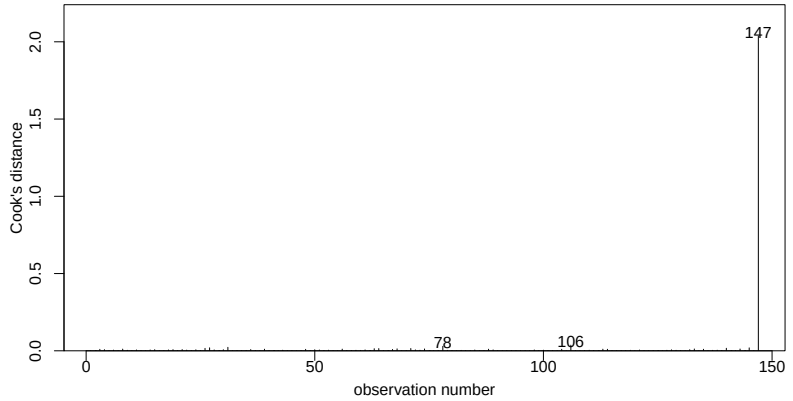
We've seem that to determine whether an observation is influential we need to remove it from the dataset and refit the model. This can be a lot of work when there a lot of observations. The Cook's distance uses linear algebra to avoid this work.

Identifing points of undue influence. . .

Cook's distance

The following Cook's distance plot gives the influence of all 147 observations in the altered dataframe. Student 147 dominates this plot and is clearly identified as being highly influential.

```
> cooks20x(examtest.fit2)
```



Identifing points of undue influence. . .

As a rough "rule of thumb"[7], an observation is deemed to be influential if:

- Removal of the observation changes any estimated parameter value by more than one standard error, or
- Its Cook's distance is greater than 0.4.

In STATS 20x we'll use the above Cook's distance threshold of 0.4. (Be aware that other courses or texts may use other thresholds, and the threshold may depend on the number of observations.)

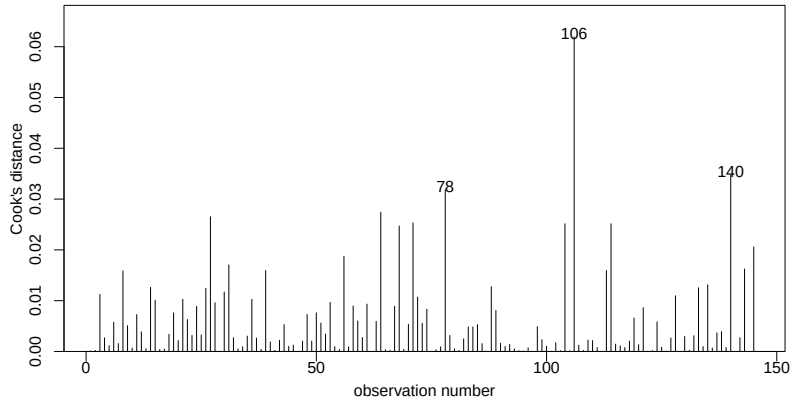
Let us get back to the original fitted model and see what its Cook's distance plot looks like.

[7] **A rule of thumb** is a principle with broad application that is not intended to be strictly accurate or reliable for every situation. It is an easily learned and easily applied procedure for approximately calculating or recalling some value, or for making some determination. Thanks Wikipedia.

Exam vs. Test marks. . .

Identifing points of undue influence. . .

```
> cooks20x(examtest.fit)
```



Exam vs. Test marks...

Validity of assumptions

We may now conclude that we can (mostly) trust the output of our analysis as our underlying assumptions seem reasonable and we do not have any unduly influential data points.

Section 2.5
Making inference from your model (provided the assumptions check out)

Statistical properties of the fitted linear model

Provided that **all** of the assumptions of the linear model hold, it can be shown (see STATS 310) that $\hat{\beta}_0$ and $\hat{\beta}_1$ are normally distributed[8] with expected values β_0 and β_1, respectively.

This means that the standard statistical techniques for making inference from normally distributed data can still be used.

In particular, the t-statistic is used for testing hypotheses, and the t-multiplier for constructing confidence and prediction intervals.

[8] That is, under repetition of the experiment that generated the data.

Interpreting the output

Testing for significant effects

```
> examtest.fit = lm(Exam ~ Test, data = Stats20x.df)
> summary(examtest.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   9.0845     3.2204   2.821  0.00547 **
Test          3.7859     0.2647  14.301  < 2e-16 ***
---
Residual standard error: 12.05 on 144 degrees of freedom
Multiple R-squared:  0.5868,Adjusted R-squared:  0.5839
F-statistic: 204.5 on 1 and 144 DF,  p-value: < 2.2e-16
```

So here we can see that the P-value associated with the `Test` variable is highly statistically significant `< 2e-16 ***`.[9]

Our underlying belief was that we could describe the relationship between exam mark and test mark using an increasing straight line. This belief has been confirmed.

[9] `2e-16` $= 2 \times 10^{-16}$, which is so small that we may just as well call it zero.

Interpreting the output...

Testing for significant effects...

The null hypothesis is always that there is no effect. In this case, that there is no relationship between test score and exam score, i.e.,

H_0: there is no relationship between `Test` and `Exam`.

Or equivalently,

$H_0 : \beta_1 = 0$, which is equivalent to saying that the data follow the `null` model $y = \beta_0 + \varepsilon$.

We will pretend that the null hypothesis is true (i.e., our working model of an increasing linear relationship between test and exam is incorrect), at least until we get evidence to the contrary.

This approach is known as the philosophy of "falsification". It is not the natural way we like to do things as humans, but is a great way to ensure we do not come to erroneous conclusions based on wishful thinking.

Interpreting the output...

Calculating the t value

Under the null hypothesis we assume $\beta_1 = 0$ and compare this to the estimated value here of $\hat{\beta}_1 = 3.7859$. As we are statisticians, we need to measure how precise this estimate is – this is measured by the standard error: $se(\hat{\beta}_1) = 0.2647$.

So under the assumption that $\beta_1 = 0$ we can say that our data tell us that we are

$$\frac{3.7859 - 0}{0.2647} = 14.301$$

standard deviations[10] away from this value. You might recognize this as the t-statistic for testing $H_0 : \beta_1 = 0$. It is the `t value` in our output.

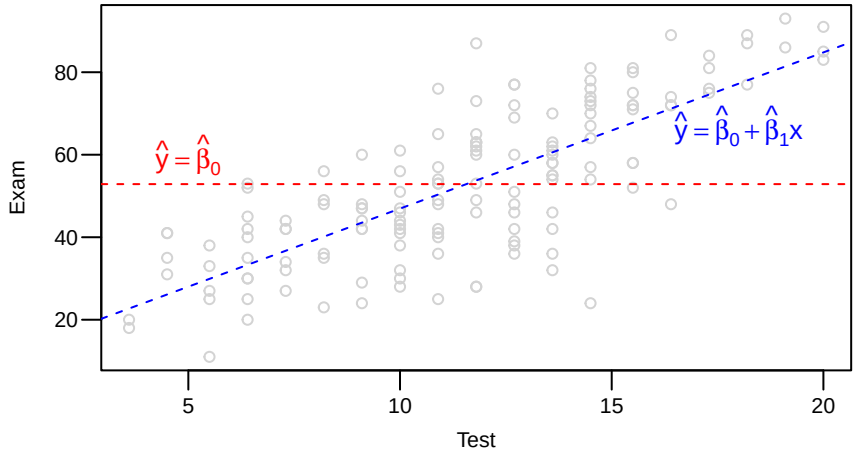
Here, we conclude that our working model was reasonable. That is, it is unreasonable to assume that `Test` had no effect on `Exam` as the chances of fluking such a result is $< 2 \times 10^{-16}$.

[10] The term "standard error" is used to refer to the standard deviation of an estimated value – in this case the standard deviation of the estimated linear effect of `Test`.

Interpreting the output...

Null vs fitted models

If $y =$ `Exam` and $x =$ `Test`, then here are our fitted null (the red line) and working (the blue line) models.



Statistical inference from our model

Confidence intervals for effects

We can also get confidence intervals for β_1 (and β_0 if need be) by:

```
> confint(examtest.fit)

               2.5 %    97.5 %
(Intercept) 2.719020 15.449907
Test        3.262659  4.309189
```

Here we can say that, on average, every increase in a student's test mark of 1 unit (out of 20) results in a 3.26 to 4.31 increase in the student's exam result.

Note: the default setting is 95% in `confint` – but this can be changed with the optional `level` argument. E.g., a 99% CI is given by

```
> confint(examtest.fit,level=0.99)

                0.5 %    99.5 %
(Intercept) 0.6778171 17.491110
Test        3.0948635  4.476984
```

Using confidence intervals to do hypothesis tests

We can be confident that the 95% confidence interval for β_1 contains the true value of β_1 because, under repetition of the experiment, 95% of the calculated confidence intervals will contain the true value of β_1. That is, we'd have to be pretty unlucky for a confidence interval not to contain what we are trying to estimate.

This means that if a hypothesized value of β_1 is not in our confidence interval, then we can be confident that it is not the true value of β_1. For example, if the value 0 is not in our 95% confidence for β_1 then we can reject the null hypothesis $H_0 : \beta_1 = 0$ (at the 5% level of significance). In fact, in the current example we would reject all values that are less than 3.26 or greater than 4.31

Choosing the best model

In our exam vs test mark example the effect of `Test` was highly significant. That is, the null hypothesis $H_0 : \beta_1 = 0$ was rejected, and so we used the fitted simple linear regression model `examtest.fit` for inference.

However, in situations where the null hypothesis $H_0 : \beta_1 = 0$ is **not** rejected, then we would conclude that there is little evidence of an association between x and y. In that case the null model would be our preferred model, and would be used for inference.

Also, note that we did not do a hypothesis test of the intercept, i.e., $H_0 : \beta_0 = 0$. This is because the intercept is not an "explanatory variable" and so this hypothesis is rarely of interest. We generally leave the intercept term in the model regardless of whether it is significant or not.[11]

[11] There may be circumstances where it is meaningful to test the statistical significance of the intercept term, but these are very rare.

Section 2.6
Confidence intervals for coefficients and fitted values, and prediction intervals for individual predictions

Estimation of fitted values

Getting fitted values using `predict`

Earlier, we calculated fitted/predicted values for students who got 0, 10, or 20 in their `Test`. Here is a way to avoid doing this by hand using the `predict` function.

Before using `predict` we have to set up a new dataframe that contains the `Test` values for which we want to predict `Exam`.

```
> ## Create data.frame of values of interest: Test = 0, 10, 20:
> ## Names of vars must be exactly the same as in the data data.frame
> preds.df=data.frame(Test=c(0,10,20))
> predict(examtest.fit, preds.df)

       1        2        3
9.084463 46.943703 84.802942
```

These values are our estimates of the expected `Exam` scores for students with `Test` scores of 0, 10 or 20, respectively.

Estimation of fitted values...

Confidence intervals for expected values using `predict`

The above fitted values are **point** estimates of the expected `Exam` score when `Test` is 0, 10, or 20. We would also like to have confidence intervals for these expected scores. This is easy:

```
> predict(examtest.fit, preds.df, interval="confidence")

        fit      lwr      upr
1  9.084463  2.71902 15.44991
2 46.943703 44.80912 49.07828
3 84.802942 79.97021 89.63568
```

Note how the CI is relatively narrow for `Test = 10` compared to the extreme `Test` values. Why is this?

Prediction of new observations

Prediction intervals for new y values using `predict`

It may also be of interest to ask for an interval that predicts the `Exam` score for a single student (rather than an interval for the expected score, as done above). This is also easy – just ask for a prediction interval instead:

```
> predict(examtest.fit, preds.df, interval="prediction")

        fit       lwr       upr
1  9.084463 -15.56475  33.73368
2 46.943703  23.03510  70.85231
3 84.802942  60.50438 109.10151
```

These intervals are much wider as they have to include the variability in individual students, and we have seen that this is large. In fact, there are some out-of-range results in these predictions – `Exam = -15.565` and `Exam = 109.102`!!

Exam vs. Test marks...

Conclusions about predictive ability

This is telling us that we have reached the limits of this linear model approximation. It does a fairly good job of explaining the trend but falls down as we can only account for 59% of the overall variation from `Test` and /or the straight line relationship may be a little too naive.

At the top end of exam/test there are fewer students, and there may be a different dynamic at this end of the data due to the constraint that exam mark must be between 0 and 100.

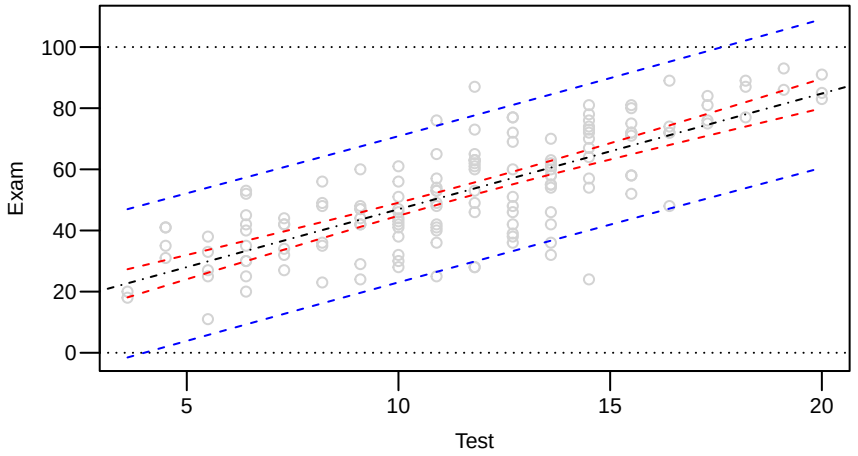
We could not use this to give a student an aegrotat mark based only on their test mark alone.

We would, ideally, like to explain more of the variability in `Exam` by using additional variables of interest – this is known as multiple (as opposed to simple) linear regression – which has the same underlying assumptions – so stay tuned.

Exam vs. Test marks...

Conclusions about predictive ability...

Here is what the confidence (red) and prediction intervals (blue) look like for these data.



Section 2.7
A recipe for subsequent analyses

A recipe for subsequent analyses

Here is a recipe (algorithm[12]) to use for problems like this where we are interested in a numerical **response** variable y (e.g.: `Exam`) and its relationship with a possible **explanatory** variable x (e.g.:`Test`):

[12] In mathematics and computer science, an algorithm is a self-contained step-by-step set of operations to be performed. Named after the the mathematician, Mohammed ibn-Musa al-Khwarizmi, who was part of the royal court in Baghdad and who lived from about 780 to 850 AD.

A recipe for subsequent analyses...

- Plot the data and see what sort of relation (if any) it suggests (there may also be a statement of research intent to guide you). Propose an appropriate working model. In the above example we decided that $y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \;\; \varepsilon_i \overset{iid}{\sim} N(0, \sigma^2)$ (where $\beta_1 > 0$.)
- Fit the working model using `lm`.
- Check the assumptions you are using and validate them. Independence OK? (how were the data collected?), EOV Okay? – `plot(examtest.fit, which = 1)`, Normality Okay? – `normcheck`. If these are okay then,
- Remove any non-significant explanatory variables where appropriate (more about this later). If so, check new working model.
- Make sure that individual points are not having undue influence and, perhaps, eliminate/correct them – `cooks20x`. If these are okay then,
- Make conclusions/predictions, discuss limitations, and answer relevant research questions.

Note in the above do not go to the next step until you are satisfied with the current step.

Section 2.8
Relevant R-code

Most of the R-code you need for this chapter

Fitting a linear (straight line) model to these data.

```
> examtest.fit=lm(Exam~Test, data=Stats20x.df)
```

Note: Independence is evaluated by investigating how the data was collected. Observations are assumed to be acting independently of each other. Much thought must go into making sure this assumption holds.

Checking for EOV

```
> plot(examtest.fit.which=1)
```

Checking approximate normality

```
> normcheck(examtest.fit)
```

Checking for points of undue influence:

```
> cooks20x(examtest.fit)
```


Most of the R code you need for this chapter...

Estimated values from the fitted model:

```
> summary(examtest.fit)
```

Confidence intervals for the model parameters (intercept and slope)

```
> confint(examtest.fit)
```

Creating a data frame of new values (for, say, Test =0, 10 or 20):

```
> preds.df=data.frame(Test=c(0,10,20))
```

Confidence intervals for expected values and prediction intervals for new observations:

```
> # confidence interval for expected value:
> predict(examtest.fit, preds.df, interval="confidence")
> # prediction interval for new observation:
> predict(examtest.fit, preds.df, interval="prediction")
```

Case studies, Method and Assumption Checks, and Executive Summaries

Russell Millar

One focus of STATS 20x is learning how to clearly and concisely write up your analysis and the results, using plain English. Specifically, in your assessments you will be required to write **Method and Assumption Checks** and/or **Executive Summaries**. These can be worth 50% or more of the total marks.

The Case Studies in this coursebook are a valuable resource for learning how to score well on **Methods and Assumption Checks** and **Executive Summaries**. Also, see the Model Answers from previous tests and exams (under Modules in CANVAS) for a wide variety of additional examples.

The requirements for scoring full marks on **Method and Assumption Checks** and **Executive Summaries** are given below.

Method and Assumptions Checks

The exact content will depend on the particular situation and modeling steps taken, but would typically include mention of:

- Justification of the steps taken to arrive at the fitted model that is used for inference. For example,
  - Inspection of plots to assess features of the data
  - Decisions made in choosing the best fitted model (e.g., transformations, variable selection, removal of outliers, use of a quasi model, etc)
  - Clear statement of the form of the chosen model (e.g., simple linear, t-test, power law, two-way ANOVA, interaction, quasi-Poisson, etc)
  - Validity, or not, of assumptions
- Equation of the fitted model (unless told otherwise)
- Statement about R^2 (the proportion of variation explained. This is only relevant to linear models - doesn't apply to generalized linear models)

Executive Summary

Typically:

- Brief introduction.
- Clear statement of the conclusions expressed in a complete sentence (e.g., ``There was a significant linear relationship between test score and exam score'').
- A clear quantification of the effect of the significant terms in the model, expressed as confidence intervals (in STATS 20x the actual estimated values are not required)
- Prediction intervals if requested. If there is reason to doubt the validity of the prediction intervals then this should be mentioned. Reasons include:
  - Residuals from a linear model being clearly non-normal (this could arise in a situtation where there is a large number of observations, so non-normality is OK for estimation of effects and confidence intervals, but not for prediction of new observations).
  - The intervals don't make sense. E.g., they have lower or upper bounds that are impossible, such as predicting a percentage and getting bounds below 0% or above 100%.
- Be very careful with your interpretation of the intervals. E.g.,
  - Is the effective additive or multiplicative?
  - Is the interval for a predicted value, or an expected value, or a median value, or a log-odds?
- ALSO, be sure to address any additional specific questions of interest that may have been asked.

Case Study 2.1: Exam vs Test

Tou Ohone Andate - staff number 1234567

Problem

We wish to quantify the relationship between test mark and exam mark, especially for the purpose of being able to predict a student's exam mark with their test mark (to aid in making decisions about aegrotat passes for students who do not sit the exam). In particular, we want to predict a student's exam mark when their test mark is either 0, 10, or 20.

The variables of interest are:

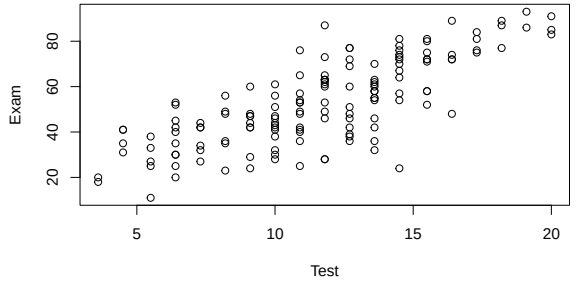
- `Exam`: Exam mark out of 100.
- `Test`: Test mark out of 20.

Question of Interest

We want to build a model to predict exam marks with test marks. In particular, we want to predict a student's exam mark when their test mark is either 0, 10, or 20.

Read in and Inspect the Data

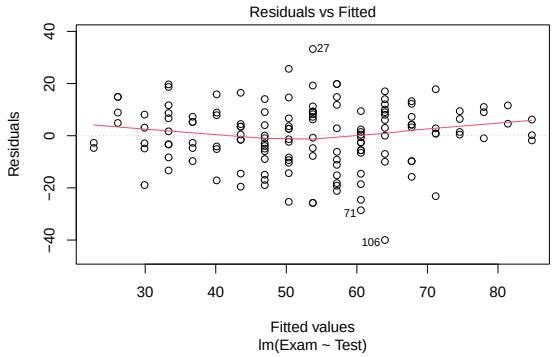
```
Stats20x.df = read.table("STATS20x.txt", header = T)
plot(Exam ~ Test, data = Stats20x.df)
```



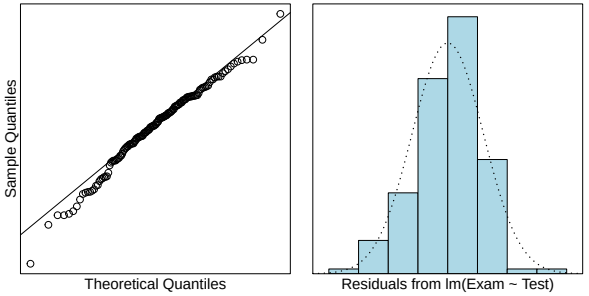
The plot reveals a positive linear relationship between exam marks and test marks.

Model Building and Check Assumptions

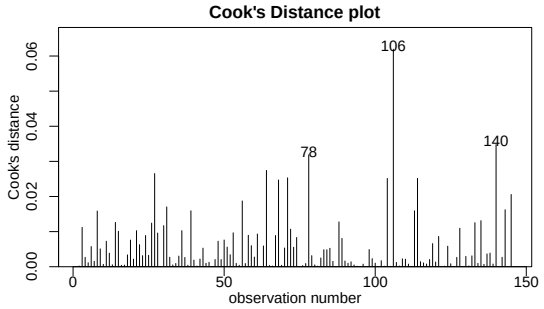
```
examTest.fit = lm(Exam ~ Test, data = Stats20x.df)
plot(examTest.fit, which = 1)
```




```
normcheck(examTest.fit)
```




```
cooks20x(examTest.fit)
```




```
summary(examTest.fit)
```

```
##
## Call:
## lm(formula = Exam ~ Test, data = Stats20x.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.980  -6.471   0.826   8.575  33.242
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   9.0845     3.2204   2.821  0.00547 **
## Test          3.7859     0.2647  14.301  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.05 on 144 degrees of freedom
## Multiple R-squared:  0.5868, Adjusted R-squared:  0.5839
## F-statistic: 204.5 on 1 and 144 DF,  p-value: < 2.2e-16
```

```
confint(examTest.fit)
```

```
##                2.5 %    97.5 %
## (Intercept) 2.719020 15.449907
## Test        3.262659  4.309189
```

Prediction Output


```
predTest.df = data.frame(Test = c(0, 10, 20))
predict(examTest.fit, predTest.df, interval = "prediction")
```

```
##         fit       lwr       upr
## 1  9.084463 -15.56475  33.73368
## 2 46.943703  23.03510  70.85231
## 3 84.802942  60.50438 109.10151
```

Method and Assumption Checks

A scatter plot of exam marks vs test marks showed a linear association with approximately constant scatter and so a linear model was fitted.

All model assumptions appear to be satisfied - a slight trend in the residual plot was observed but does not seem to be of major concern.

Our final model is

$$Exam_i = \beta_0 + \beta_1 \times Test_i + \epsilon_i,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Our model explained a modest 59% of the variability in the students' final exam marks.

Executive Summary

We were interested in building a model to predict exam mark from test mark.

There was a significant linear relationship between test mark and exam mark ($P\text{-}value \approx 0$). We estimate that each additional test mark (out of 20) obtained by the student would increase their exam mark by between 3.3 to 4.3 (out of 100) on average.

For test marks of 0, 10 and 20, we predict exam marks (for individual students) between -15.6 to 33.7, 23.0 to 70.9, and 60.5 to 109.1, respectively. These intervals are very wide[1] and some of these intervals have bounds that are outside of the feasible values of exam mark (0-100). The model is not reliable for prediction.

[1] Due to considerable variabilty remaining even after taking into account the test mark.

CS2.2: Fire Damage

Tou Ohone Andate - staff number 1234567

Problem

An insurance company wants to predict the repair cost of damage (in $000's) to a house in a particular area if a fire occurs. The damage, and distance from the fire station were recorded for a random sample of 15 house fires in the area of interest. They were also particularly interested in predicting the repair cost of fire damage to a randomly chosen house at distances of 1 and 4 miles from the fire station.

The variables measured were:

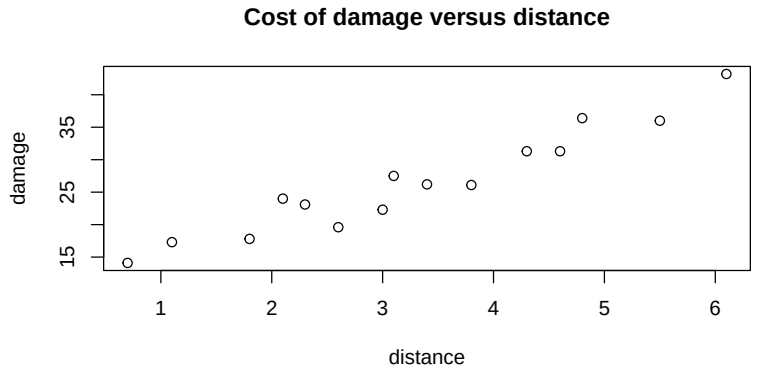
- `damage`: repair cost of damage (in $000's)
- `distance`: distance from the fire station (in miles)

Question of interest/goal of the study

We were interested in predicting the cost of damage caused to houses by fires depending on how far away they are from the nearest fire station. In particular, we want to predict the cost of damage to individual randomly chosen houses that are 1 and 4 miles from the fire station.

Read in and inspect the data:

```
# import the data
fire.df=read.table("fire.txt", header=T)
plot(damage~distance,main="Cost of damage versus distance", data=fire.df)
```

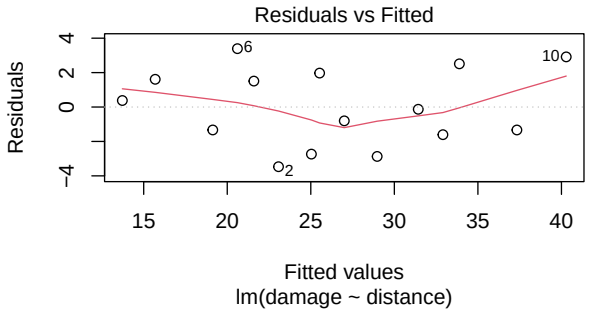


Comment on the plot

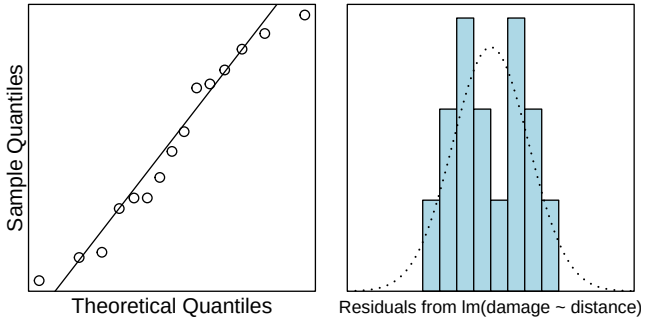
There is, not surprisingly, an increasing trend with the distance from the fire station and the cost of fire damage suffered by the house. The trend looks reasonably linear and the variability around this trend looks reasonably constant.

Fit model and check assumptions

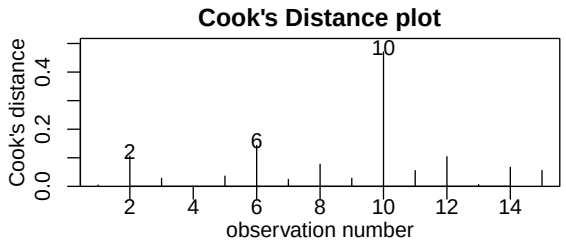
```
# fit the model
fire.fit<-lm(damage~distance,data=fire.df)

#Assumption checks
plot(fire.fit,which=1)
```




```
normcheck(fire.fit)
```



```
cooks20x(fire.fit)
```



Observation 10 exceeds the Cook's distance threshold of 0.4. However, from the fitted vs residuals plot we see that this point is not at all anomalous – it is influential because it is the house that is furtherest from the fire station (i.e., has an "extreme" value for the explanatory variable). Moreover, there are only 15 observations, so it is not surprising that a single point could be influential. There is no reason to remove this point.

```
#Get summary output and confidence intervals
summary(fire.fit)
```

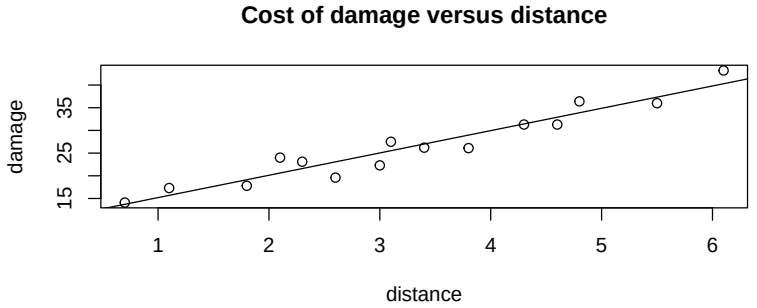
```
##
## Call:
## lm(formula = damage ~ distance, data = fire.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4682 -1.4705 -0.1311  1.7915  3.3915
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.2779     1.4203   7.237 6.59e-06 ***
## distance      4.9193     0.3927  12.525 1.25e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.316 on 13 degrees of freedom
## Multiple R-squared:  0.9235, Adjusted R-squared:  0.9176
## F-statistic: 156.9 on 1 and 13 DF,  p-value: 1.248e-08
```

```
confint(fire.fit)
```

```
##                2.5 %    97.5 %
## (Intercept) 7.209605 13.346252
## distance    4.070851  5.767811
```


Plot with superimposed line

```
plot(damage~distance, main="Cost of damage versus distance", data=fire.df)
abline(fire.fit)
```



Get additional predicted output

```
preddistance.df=data.frame(distance=c(1,4))
predict(fire.fit,preddistance.df,interval="prediction")
```

```
##        fit      lwr      upr
## 1 15.19726  9.67879 20.71573
## 2 29.95525 24.75100 35.15951
```

Method and Assumption Checks

A scatter plot of damage vs distance showed a linear association with approximately constant scatter and so a simple linear regression model was fitted.

We have a random sample of fires so the results should be independent of each other. A slight trend in the residual plot was observed, but does not appear to be of major concern. Observation 10 had a Cook's distance of about 0.5, but is not an anomalous point. We conclude that all assumptions are reasonably well satisfied.

Our final model is $damage_i = \beta_0 + \beta_1 \times distance_i + \epsilon_i$ where $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Our model explains 92% of the variation in house fire damage cost.

Executive Summary

An insurance company wanted to predict the cost of damage that occurs when a house catches fire using distance from the fire station as an explanatory variable.

Not surprisingly, there was strong evidence that the further a house is away from a fire station the more fire damage it suffers. We estimate that for each additional mile from the fire station, the expected fire damage cost increases by between $4,100 and $5,800.

Our model explains 92% of the variation in house fire damage and should therefore be a reasonable model for prediction. We predict that if a new fire occurs in a house that is 1 mile from the fire station, the damage will be between $9,700 and $20,700. For a house that is 4 miles from the fire station, we predict the damage will be between $24,800 and $35,200.

Aside 1

If one was concerned about the slight curvature in the residual plot then this could be checked by adding a quadratic term in distance. This quadratic term is not significant, and so the above simple linear model is preferred.

Aside 2

Although not a question of interest, the intercept corresponds to the cost of damage of houses next door (i.e., zero distance) to the fire station. In this case, we estimate an expected fire damage cost of $7,200 to $13,300.

CS2.3: Diamond Rings

Tou Ohone Andate - staff number 123456789

Problem

This data set contains the prices of ladies' diamond rings and the carat size of their diamond stones from a random sample of rings from Singaporean retailers. The rings are made of 20 carat[1] gold and are each mounted with a single diamond stone. The data was collected by a lecturer quite a few years ago when they were in Singapore and they were interested in building a model to explain the price of diamond rings.

In particular, it was hoped that the prices of two rings could be predicted using the model: a 0.3-carat diamond ring and a 1.2-carat diamond ring.[2]

The variables measured were:

- `price`: price of ring (in Singapore dollars)
- `weight`: weight of diamond (in carats)

Question of interest/goal of the study

We were interested in building a model to explain the price of diamond rings. In particular, we want to predict the price of a 0.3-carat diamond ring and a 1.2-carat diamond ring.

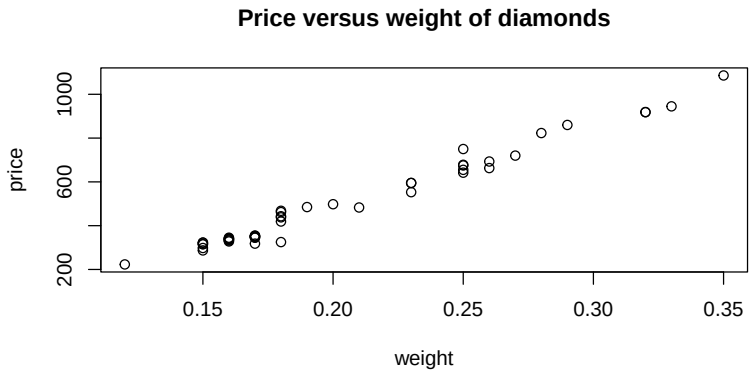
Read in and inspect the data:

```
# import the data
diamonds.df=read.table("diamonds.txt", header=T)
head(diamonds.df)
```

```
##   weight price
## 1   0.17   355
## 2   0.16   328
## 3   0.17   350
## 4   0.18   325
## 5   0.25   642
## 6   0.16   342
```

```
plot(price~weight,main="Price versus weight of diamonds",data=diamonds.df)
```

[1] In the context of gold, "carat" refers to the purity of the gold.
[2] In the context of diamonds, "carat" refers to the weight, specifically, one carat is 200 milligrams.

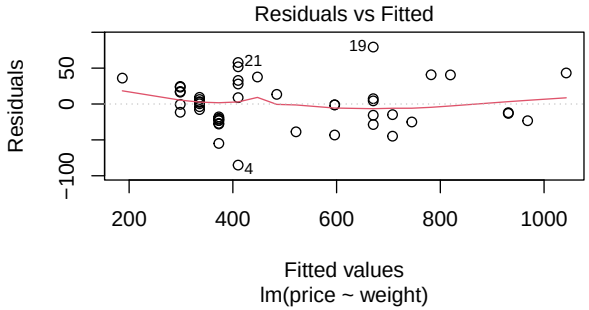


Comment on the plot

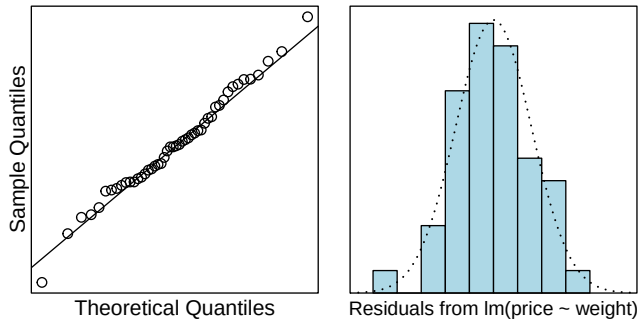
The scatter plot of price versus weight shows a strong, increasing, linear relationship. The greater the weight of the diamond, the greater the mean price of the diamond ring.

Fit model and check assumptions

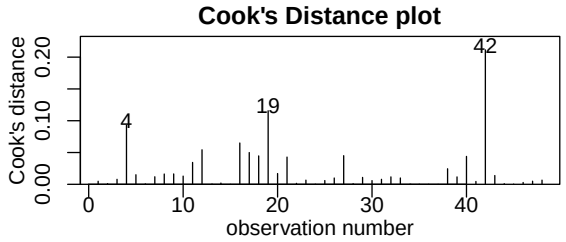
```
# fit the model
diamond.fit<-lm(price~weight,data=diamonds.df)

#Assumption checks
plot(diamond.fit,which=1)
```



```
normcheck(diamond.fit)
```




```
cooks20x(diamond.fit)
```




```
#Get summary output and confidence intervals
summary(diamond.fit)
```

```
##
## Call:
## lm(formula = price ~ weight, data = diamonds.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -85.159 -21.448  -0.869  18.972  79.370
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -259.63      17.32  -14.99   <2e-16 ***
```



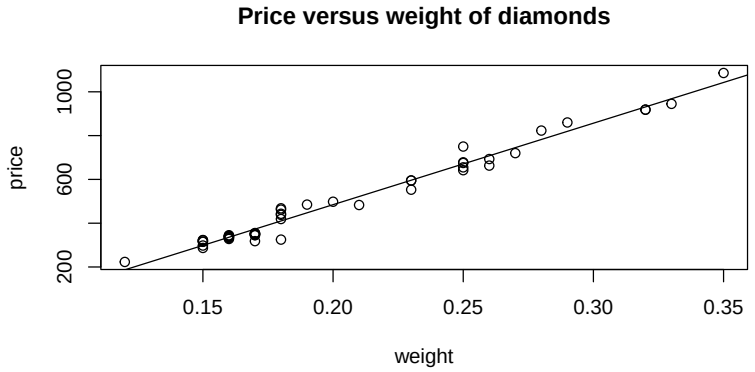
```
## weight       3721.02      81.79   45.50   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.84 on 46 degrees of freedom
## Multiple R-squared:  0.9783, Adjusted R-squared:  0.9778
## F-statistic:  2070 on 1 and 46 DF,  p-value: < 2.2e-16
```

```
confint(diamond.fit)
```

```
##                2.5 %    97.5 %
## (Intercept) -294.487 -224.7649
## weight      3556.398 3885.6513
```

Plot with superimposed line

```
plot(price~weight,main="Price versus weight of diamonds",data=diamonds.df)
abline(diamond.fit$coef[1],diamond.fit$coef[2])
```



Get additional predicted output required

```
predweight.df=data.frame(weight=c(0.3,1.2))
predict(diamond.fit,predweight.df,interval="prediction")
```

```
##         fit       lwr       upr
## 1  856.6815  790.0316  923.3315
## 2 4205.6039 4029.3376 4381.8702
```

Method and Assumption Checks

A scatter plot of price vs diamond weight showed a linear association with approximately constant scatter and so a simple linear regression model was fitted.

All the assumptions were met so we have no problems with the analysis.

Our final model is $price_i = \beta_0 + \beta_1 \times weight_i + \epsilon_i$ where $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Our model explains 98% of the variation in diamond ring prices.

Executive Summary

We have data on diamond ring prices and the weights of the diamonds in those rings from Singapore retailers. Our aim is to predict the price of a diamond ring using the weight of the diamond.

There is a strong positive association between the weight of the diamond and the price of the ring - the bigger the diamond, the higher the price of the diamond ring.

We estimate that for every 0.1-carat increase in the weight of the diamond, the mean diamond ring price increases by somewhere between $360 and $390.

Our model explains 98% of the variation in diamond ring prices and therefore should be excellent for predicting the price of diamond rings.

Using our model, we predict that the 0.3-carat diamond ring will be priced between $790 and $920.

Our data only has diamond rings weighing up to 0.35 carats, so we cannot rely on the predictions for the 1.2-carat ring.

[Note: the range of the orignal data was around 900 dollars, this has been reduced to around 130 dollars which is roughly 15% of this and much more useful for prediction.]

Chapter 3: Equivalence of the null linear model and the one-sample t-test

STATS 201/8

University of Auckland

Learning outcomes

In this chapter you will learn about:

- The equivalence of fitting the null model and doing a one-sample *t*-test.
- The paired *t*-test
- Relevant R-code.

Section 3.1
Revisiting the null model

Null vs simple linear regression fits

We have already encountered an example of the null model in Chapter 1.

We saw that we could explain approximately 59% of the variation of *Exam* by fitting a straight line model using *Test*. We calculated this by comparing the sums-of-squares of the residuals for the simple linear model to the sums-of-squares of the residuals of the null model and noticed it had decreased by 59% (ie., $R^2 = 0.59$).

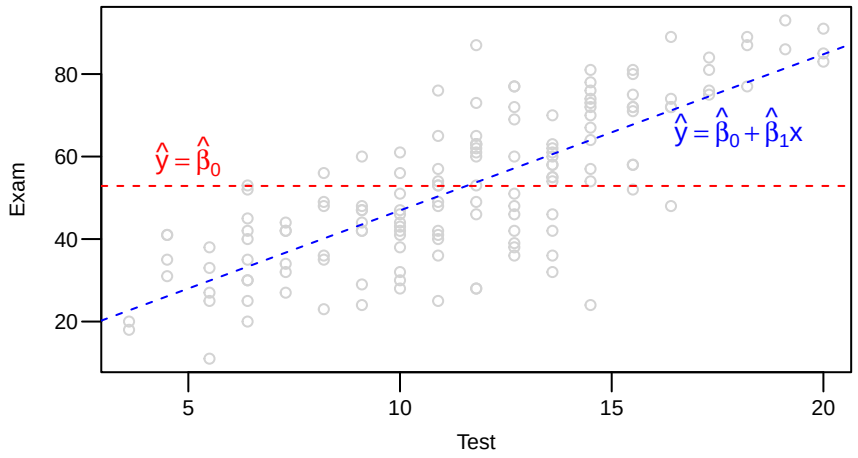
In this chapter we will examine the null model in greater detail and see that it is equivalent to applying the t-distribution for obtaining confidence intervals, and to the one-sample *t*-test of the null hypothesis that the population mean is zero.

It is also the model we need to use when we have paired comparisons – two repeated measures on the same subject.

Null vs simple linear regression fits...

Exam vs. Test marks

If $y =$ `Exam` and $x =$ `Test`, then here are our fitted null (the red line) and linear (the blue line) models.



A note on the model formula

In linear models, the intercept parameter (β_0) is fitted by default. That is why `lm(y~x)` fits not just the effect of x, but also an intercept. In fact, `lm(y~x)` is a shortened version of `lm(y~1+x)`. The latter form makes it explicit that the model being fitted is

$$y = 1 \times \beta_0 + x \times \beta_1 + \varepsilon,$$

where the ε are *iid* $N(0, \sigma^2)$. This is why the null or intercept-only model can be can be fitted using `lm(y~1)` since this specifies

$$y = 1 \times \beta_0 + \varepsilon.$$

Since the null model is simply just specifying the mean (i.e., expected) value of y, it is common practive to relabel β_0 as μ, in which case we have $y = \mu + \varepsilon$.

This can be abbreviated with the model formula $y \overset{iid}{\sim} N(\mu, \sigma^2)$.

Inference about the expected exam mark

What do we mean by "expected exam mark"??? (Hint: It is also called the population mean.)

Remember, the data are assumed to be a random sample from a bigger population.

Every STATS 20x class differs a bit in the difficulty of the test and exam, for the simple reason that there are different questions every semester. The teaching staff also differ each semester.....so, it would be naive to regard these data as a random sample from all STATS 20x students that we have or will ever teach.

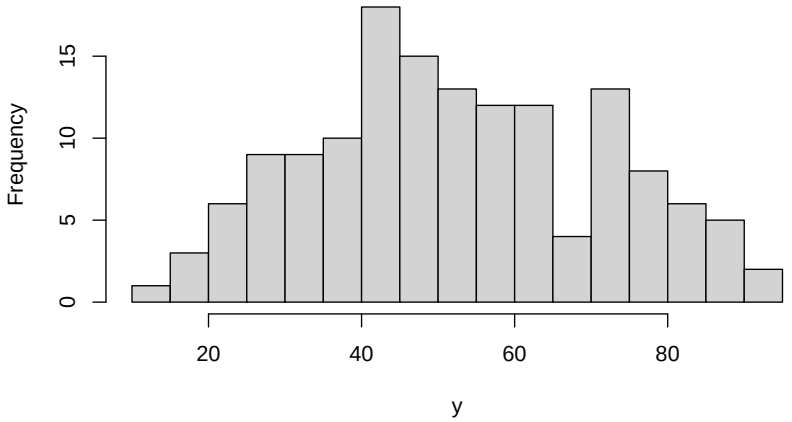
However, it would be reasonable to assume they are from the hypothetical population of all students who could have taken STATS 20x in that particular semester.

Here we wish to see what we can say about the average, or typical, value a student in this hypothetical population will get in the exam in the absence of any other information about them.

Inference about the expected exam mark...

To save some typing we'll let `y` be the vector `Stats20x.df$Exam` of exam scores.

```
> y=Stats20x.df$Exam
> hist(y,breaks=20,main="") #Use main to suppress plot title
```



Inference about the expected exam mark...

Using the null model

The histogram could be better (i.e., more normal in shape), but we'll go ahead with using the null model. We're going to be lazy[1] and not include the assumption checks!

```
> null.fit=lm(y~1)
> coef(summary(null.fit)) #Only give coefficients from summary

            Estimate Std. Error  t value     Pr(>|t|)
(Intercept) 52.87671   1.545802 34.20666 2.632011e-71

> confint(null.fit)

               2.5 %   97.5 %
(Intercept) 49.8215 55.93193
```

[1] Actually, it's more like we are taking a shortcut – the assumption checks won't tell us anything more than what we already see in the histogram of the exam marks.

Inference about the expected exam mark...

Using the null model...

Conclusions

- The near zero `Pr(>|t|)` p-value totally rejects the null hypothesis that $H_0 : \mu \equiv \beta_0 = 0$.
- The 95% confidence interval for μ is 49.82 to 55.93.

The confidence interval is useful... but the p-value for $H_0 : \mu = 0$ is absolutely useless since we would never be interested in asking whether $\mu = 0$.

Inference about the expected exam mark...

A more meaningful null hypothesis

It might be more interesting to test a hypothesis like $H_0 : \mu = 60$, say, where we suppose that 60 corresponds to the target expected score when lecturers prepare STATS 20x exams.[2]

Note that the above null hypothesis is $H_0 : \mathrm{E}[Y] = 60$ and is equivalent to $H_0 : \mathrm{E}[Y - 60] = 0$.

So, if we use the response variable $y - 60$ instead of y then we can get a P-value for this H_0 using the `lm` function, as shown on the next slide.

[2] FYI, in recent semesters the average test and exam scores have been around 70.

Inference about the expected exam mark...

A more meaningful null hypothesis...

```
> null.fit60=lm(I(y-60)~1)
> coef(summary(null.fit60))

             Estimate Std. Error   t value     Pr(>|t|)
(Intercept) -7.123288   1.545802 -4.608151 8.828149e-06
```

Here we have used the inhibit function `I()` to prevent `lm` from mis-interpreting the model formula.

We see that the sample average exam score is about 4.61 standard errors below the target population exam score. The small p-value shows that this is implausible under the null hypothesis.

Question: In plain English, how would you state our conclusion?

Section 3.2
Revisiting the t-test

Inference about the population mean...

Using the t-test

Recall from STATS 10x that we can use the t-distribution to make inference about μ when $y \overset{iid}{\sim} N(\mu, \sigma^2)$.

First, we will do it the hard way, by hand.

Then, we'll let R do it for us.

Inference about the population mean...

Using the t-test...

It can be shown (in STATS 310) that

$$T = \frac{\overline{y} - \mu}{\frac{s}{\sqrt{n}}} \sim t_{n-1},$$

where $\overline{y}$ and s are the sample mean and sample standard deviation we calculate from our sample.[3]

We can use this result to do hypothesis tests and get confidence intervals about the quantity of interest, μ (the population mean exam mark).[4]

[3] Recall that this is interpretted as being the distribution of T when the experiment is repeated. That is, if other random samples are taken from the population.

[4] Recall that T is t_{n-1}-distributed rather than normal as we have additional variability from using s^2 to estimate σ^2.

Inference about the population mean...

Calculating the t-value

In 10x you were taught how to calculate a 95% CI for μ from your sample of n observations having sample mean $\bar{y}$ and sample standard deviation s:

$$\bar{y} \pm t_{n-1}^{(0.975)} \frac{s}{\sqrt{n}}$$

where $t_{n-1}^{(0.975)}$ is the t-multiplier. For a 95% CI this multiplier is pretty close to 2 provided that $n > 30$. [5]

[5] In this example, $t_{n-1}^{(0.975)} = t_{145}^{(0.975)} = 1.97646 \approx 2$.

Inference about the population mean...

Calculating the t-value...

Let us calculate the t-statistic for the null hypothesis that $\mu = 60$. This is

$$T = \frac{\bar{y} - 60}{\frac{s}{\sqrt{n}}}.$$

```
> n=length(y) #146 students
> tstat=(mean(y)-60)/(sd(y)/sqrt(n))
> tstat

[1] -4.608151
```

How does this t-value compare to the one from `coef(summary(null.fit60))` that we saw a few pages earlier?

Inference about the population mean...

Calculating the confidence interval

Let us now manually compute a 95% CI based on the t-distribution:

```
> ## t-multiplier
> tmult = qt(1-.05/2, df=n-1)
> ## We want the upper 97.5% (or 1-.05/2) bound of the CI
> ## NOTE: mean = sample mean; sd = standard deviation; sqrt = square root
> mean(y) - tmult*sd(y)/sqrt(n)

[1] 49.8215

> ## Upper bound of CI
> mean(y) + tmult*sd(y)/sqrt(n)

[1] 55.93193

> ## Or if we want both the lower and upper bounds of the CI in one statement
> mean(y) + c(-1,1)*tmult*sd(y)/sqrt(n)

[1] 49.82150 55.93193
```

How does this CI compare to the one from `confint(null.fit)`?

Inference about the population mean...

The `t.test` function in R

Of course, R has a convenient function to do the t-test calculations for us.

To test $H_0 : \mu = 60$, we include `mu=60` in the call of `t.test`

```
> t.test(y,mu=60)


One Sample t-test

data:  y
t = -4.6082, df = 145, p-value = 8.828e-06
alternative hypothesis: true mean is not equal to 60
95 percent confidence interval:
 49.82150 55.93193
sample estimates:
mean of x
 52.87671
```


Inference about the population mean...

The null model versus the t-test

We've seen that we get the same results from using the null model as we do from using the t-test. This is because they are both based on the same statistical theory.

The null model is a special case (in fact, it is the simplest case) of a linear model.[6]

For completeness we will also look at using the bootstrap, which you saw in STATS 10x – recall that the bootstrap samples **with replacement** from the data.

[6] If you have done some maths courses, you might recall that a linear model is one that has constant derivative with respect to its coefficients

Section 3.3
Inference about μ using the bootstrap (Non examinable)

Inference about the population mean...

Bootstrapping

```
> ## Resampling the exam marks, N times with replacement:
> N=10000 # The number of bootstrap resamples we want
> # The new sample means are stored in ybar
> ybar=rep(NA,N) ## A vector of length N to store our resampled means
>
> ## A loop - allows us to do something N (10,000) times
> for (i in 1:N){
+   ## Take the average of this sample (below) from a sample of size n = 146 from y
+   ybar[i]=mean( sample(y,n, replace=T) )
+ }
> mean(ybar)

[1] 52.84468
```

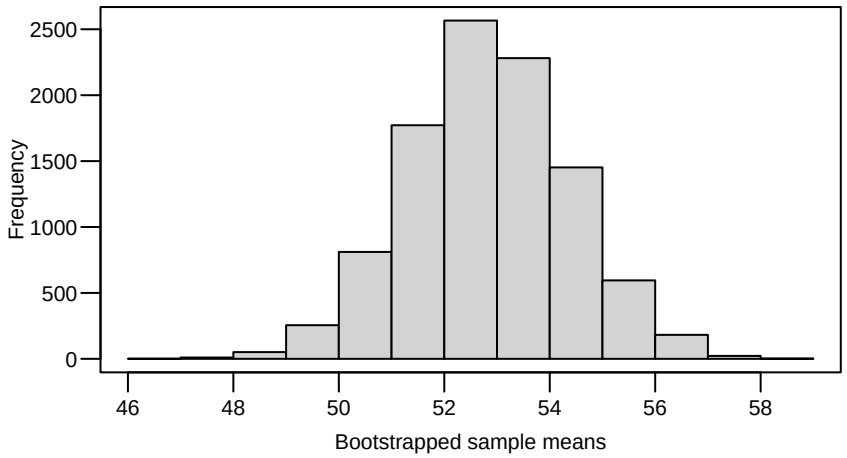
Here is a simpler way of doing the bootstrap, but it requires the `bootstrap` package to be installed:

```
> library(bootstrap)
> ybar = bootstrap(y, 10000, mean)$thetastar
```


Inference about the population mean...

Bootstrapping ...

```
> ## Histogram of these 10,000 bootstrap means
> hist(ybar,xlab="Bootstrapped sample means")
```



Inference about the population mean...

Bootstrapping ...

This looks very 'Normal'.

Note: the mean value of `ybar`, 52.84, is about the same as that of the original sample (52.88), but the values have less scatter. They have lower quartile of 51.8 (in the original sample it was 40), and upper quartile 53.9 (versus 68.5).

A 95% bootstrap confidence interval for the expected exam mark is given by the following code:

```
> ## 2.5% in the lower tail and 2.5% in the upper tail
> quantile(ybar, c(.025, .975))

    2.5%    97.5%
49.86301 55.85616
```

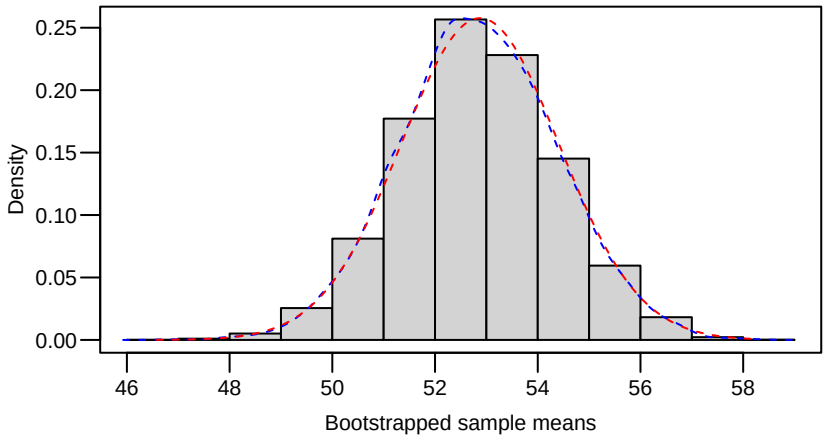
We say: We are 95% confident that the expected exam mark is somewhere between 49.9 to 55.9 marks.[7]

[7] The magic of the bootstrap is that these quantiles of the bootstrapped means give a CI for the population mean – see STATS 730 for proof of this.

Inference about the population mean...

Bootstrapping...

Here is the distribution of sample means we bootstrapped, with a density plot (in blue – which can be though of as a 'fine-grain' histogram), along with the underlying theoretical Normal based-distribution (in red):



Inference about the population mean...

Bootstrapping and the t-distribution

So what is going on here? Provided we have a large enough sample, we know that the distribution of all possible sample means we could have obtained from repeating the experiment is distributed approximately normally[8] and hence, these sample means can be described well with a normal distribution.

This is known as the **Central limit effect** or **theorem**, referred henceforth as the **CLT**. Both the bootstrap and t-distribution follow the CLT.

[8] There are some other necessary conditions but this holds for most populations.

Inference about the population mean...

Bootstrapping and the t-distribution...

Let us compare the bootstrap CI to the t-distribution CI:

```
> ## Bootstrap:
> quantile(ybar, c(.025, .975))

    2.5%    97.5%
49.86301 55.85616

> ## t-distribution:
> mean(y) + c(-1,1)*tmult*sd(y)/sqrt(n)

[1] 49.82150 55.93193
```

Very similar.

Note: From now on we will not do any more bootstrapping as it does not generalize easily to more complex models.

Section 3.4
The paired t-test

Paired comparisons ≡ one-sample *t*-test

Recall (from STATS 10x) that the paired t-test is just an application of the one-sample t-test applied to differences.

Here, we demonstrate this by comparing *Test* and *Exam* marks.

Comparing *Test* and *Exam* marks

Suppose we want to know if the midterm test marks and exam marks have the same expected value. Note that the test and exam scores are not independent (why?).

The data are paired since we have a test score and exam score from each student[9].

For a meaningful comparison, We will need to make them have the same scale, so we multiply the test mark by 5 so that it is also out of 100. This can be done very easily with the following R code:

```
> Stats20x.df$Test2 = 5 * Stats20x.df$Test
> ## Check that it worked
> Stats20x.df[1:3, c("Exam","Test","Test2")]

  Exam Test Test2
1   42  9.1  45.5
2   58 13.6  68.0
3   81 14.5  72.5
```

[9] Two measurements on the same student constitutes a paired measure and this is an example of a repeated (twice) measures study.

Comparing *Test* and *Exam* marks...

We wish to see how the exam score and scaled test score (out of 100) differ. We might suspect that they have the same expected value μ.

A student who scores high on the exam would be expected to score high on the test and vice-versa. So these two measurements are not independent of each other. However, when we look at their differences (`Diff = Test2 - Exam`) these constitute a single measurement from each student, and moreover these differences could reasonably be assumed to be independent of each other.

In effect, we have eliminated the student effect on test and exam scores by working with the difference between test and exam for each student.

Comparing *Test* and *Exam* marks...

Calculating the difference

Let us name the new variable `Diff` (and check that we have done it correctly):

```
> Stats20x.df$Diff = Stats20x.df$Test2 - Stats20x.df$Exam
> ## Check the first 5 measurements
> Stats20x.df[1:5, c("Test2","Exam","Diff")]

  Test2 Exam Diff
1  45.5   42  3.5
2  68.0   58 10.0
3  72.5   81 -8.5
4  95.5   86  9.5
5  41.0   35  6.0

> ## Looks good!
```

Comparing *Test* and *Exam* marks...

Null hypothesis for the expected difference

If test and exam scores have the same expected value, then their difference must have an expected value of 0. We will denote this $\mu_{diff} = 0$. This is our null hypothesis.

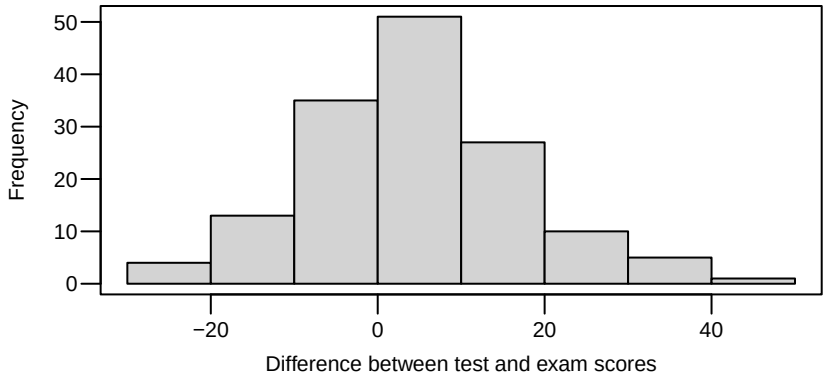
Before we do the test of $H_0 : \mu_{diff} = 0$ we should do some assumption checks.

We have independence sorted, and we are necessarily assuming identical distribution. Let us produce a histogram to check normality.

Comparing *Test* and *Exam* marks...

Inspecting the differences

```
> hist(Stats20x.df$Diff,xlab="Difference between test and exam scores")
```



Looks very normalish. Let's fit a null linear model (i.e., one-sample t-test) to the differences.

Testing for a significant difference

```
> diff.fit = lm(Diff~1, data=Stats20x.df)
> coef(summary(diff.fit))

            Estimate Std. Error  t value     Pr(>|t|)
(Intercept) 4.958904   1.063718 4.661861 7.042125e-06
```

```
> confint(diff.fit)

                2.5 %   97.5 %
(Intercept) 2.856509 7.061299

> t.test(Stats20x.df$Diff)


One Sample t-test

data:  Stats20x.df$Diff
t = 4.6619, df = 145, p-value = 7.042e-06
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 2.856509 7.061299
sample estimates:
mean of x
 4.958904
```


Comparing *Test* and *Exam* marks...

Conclusions

So it appears that, on average, students do worse in the exam than the term test (P-value $\approx 7 \times 10^{-6}$).

We estimate this difference to be about 2.9 to 7.1 marks (out of 100).

The exam was considerably harder than the test – this is something that lecturers normally try to avoid. It's not a good idea for the test to be easier than the exam, since it may lull students into a false sense of security.

Comparing *Test* and *Exam* marks...

Closing remarks

Some history; The `Stats20x.df` data were collected some years ago. Back then, lecturers were able to scale marks as they saw fit, and were also to choose the final grade ranges for awarding the letter grades (A+, A,...etc).

So, lecturers would often deliberately set very challenging tests and exams since they would be able to adjust grades upwards and lower the grade requirement for a letter grade. This would have been the case for the test and exam marks in `Stats20x.df`.

Lecturers no longer have as much flexibility, and so these days test and exam (and assignment) marks must now be more representative of the final grade.

Section 3.5
Relevant R-code

Most of the R-code you need for this chapter

Fitting the model with no explanatory variables (i.e. no x):

```
> exam.fit=lm(Exam~1, data=Stats20x.df)
> confint(exam.fit)
> exam.fit60=lm(I(Exam-60)~1, data=Stats20x.df)
> coef(summary(exam.fit60))
```

Equivalent output can be obtained from the t-test:

```
> t.test(Stats20x.df$Exam,mu=60)
```

For a paired comparison we need to create the difference variable:

```
> Stats20x.df$Diff = Stats20x.df$Test2 - Stats20x.df$Exam # create differences
```

Then either of these will suffice:

```
> # confidence interval for fitted value:
> diff.fit=lm(Diff~1, data = Stats20x.df)
> coef(summary(diff.fit))
> confint(diff.fit)
```

or equivalently

```
> t.test(Stats20x.df$Diff)
```

Case Study 3.1: Exam by itself

Tou Ohone Andate - staff number 1234567

Problem

We wish to investigate the distribution of exam marks. In particular, we want to test the hypothesis that the underlying mean value of exam score is the "historical average'' of 55.

The variable of interest is:

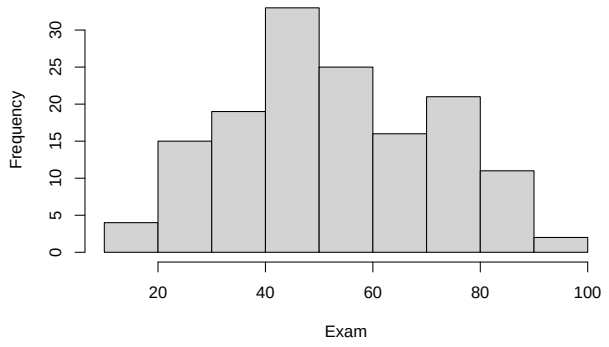
- `Exam`: Exam mark out of 100.

Question of Interest

We were interested in building a model to describe exam marks. In particular, we want to test the hypothesis that the underlying mean value of exam score is the "historical average'' of 55.

Read in and Inspect Data

```
Stats20x.df = read.table("STATS20x.txt", header = T)
hist(Stats20x.df$Exam,xlab="Exam",main="")
```




```
summaryStats(Stats20x.df$Exam)
```

```
## Minimum value:           11
## Maximum value:           93
## Mean value:              52.88
## Median:                  51.5
## Upper quartile:          68.5
## Lower quartile:          40
## Variance:                348.87
## Standard deviation:      18.68
## Midspread (IQR):         28.5
## Skewness:                0.16
## Number of data values:   146
```

The exams marks are centred just above 50. The data look reasonably unimodal and symmetrical – roughly normal. Some slight right-skewness, but does not look like a problem.

Fitting the null model

By hand...

```
( mn_exam = mean(Stats20x.df$Exam) ) # Sample mean
```

```
## [1] 52.87671
```

```
( sd_exam = sd(Stats20x.df$Exam) ) # Sample standard deviation
```

```
## [1] 18.67799
```

```
( n_exam = length(Stats20x.df$Exam) ) # Sample size
```

```
## [1] 146
```

```
( tmult_exam = qt(1 - 0.05/2, df = n_exam - 1) ) # t-multiplier
```

```
## [1] 1.97646
```

```
( CI_exam = mn_exam + tmult_exam * c(-1, 1) * sd_exam/sqrt(n_exam) ) # Confidence Interval
```

```
## [1] 49.82150 55.93193
```

```
( se_exam = sd_exam/sqrt(n_exam) ) # Standard error
```

```
## [1] 1.545802
```

```r
(t_stat_exam = (mn_exam - 55)/(se_exam) ) # t-stat
```

```
## [1] -1.373583
```

```r
(pval_exam = 2 * (1 - pt(abs(t_stat_exam), df = n_exam - 1)) ) # p-value
```

```
## [1] 0.171691
```

Using lm...

```r
examNull.fit55 = lm(I(Exam-55) ~ 1, data = Stats20x.df)
( pval_exam = coef(summary(examNull.fit55))[4] )
```

```
## [1] 0.171691
```

```r
55+confint(examNull.fit55)
```

```
##               2.5 %   97.5 %
## (Intercept) 49.8215 55.93193
```

Finaly, with the t.test function

```r
t.test(Stats20x.df$Exam, mu = 55)
```

```
##
##  One Sample t-test
##
## data:  Stats20x.df$Exam
## t = -1.3736, df = 145, p-value = 0.1717
## alternative hypothesis: true mean is not equal to 55
## 95 percent confidence interval:
##  49.82150 55.93193
## sample estimates:
## mean of x
##  52.87671
```

Method and Assumption Checks

There are no explanatory variables, and so a null model was fitted.

From examining the histogram it appears that the data are roughly normally distributed, so model assumptions are satisfied.

Our final model is

$$Exam_i = \beta_0 + \epsilon_i \text{ (or } Exam_i = \mu + \epsilon_i) \ ,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$

Executive Summary

We were interested in building a model to describe exam marks.

We estimate the expected exam mark to be between 49.8 and 55.9 (out of 100).

We have no reason to believe that the expected exam mark differs from the historical average value of 55 (out of 100) ($P\text{-}value = 0.17$).

Chapter 4:
Fitting curves with the linear model

STATS 201/8

University of Auckland

Learning outcomes

In this chapter you will learn about:

- Identifying a curved relationship between x and y
- Fitting a quadratic curve using a linear model
- Relevant R-code.

Section 4.1
Identifying a curved relationship

New Example – Exam vs. assignment marks

We'll continue working with the STATS 20x data, but now we are interested to see if assignment mark is associated with exam mark.

Again, we are pretty sure we know what the answer is, but we need to formally confirm our suspicions. Also, we want to use assignment mark to help explain (i.e., make inference about) exam mark.

The variables of interest are:

`Exam`	the student's exam mark (out of 100)
`Assign`	the student's assignment mark (out of 20).

Once again, `Exam` is the (numeric) response variable, and now `Assign` is the (numeric) explanatory variable.

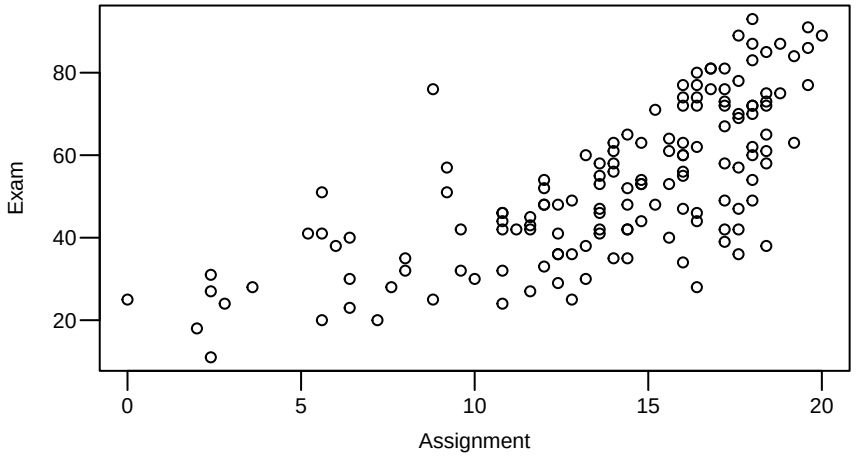
Exam vs. assignment marks...

Setting things up

```
> ## Load the s20x library into our R session
> library(s20x)
> ## Importing data into R
> Stats20x.df = read.table("Data/STATS20x.txt", header=T)
> ## Examine the data
> plot(Exam ~ Assign, data = Stats20x.df,xlab="Assignment")
```


Exam vs. assignment marks...

Scatterplot of the data

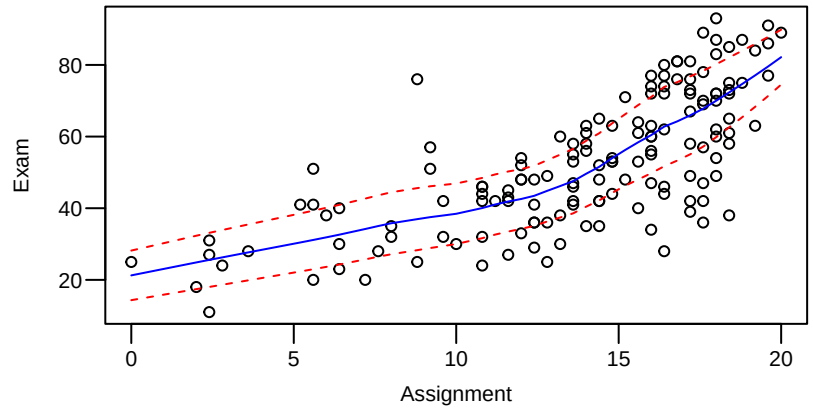


Hmmm, not quite a straight line – could be some curvature.
Maybe `trendscatter` will paint a clearer picture.

Exam vs. assignment marks...

Scatterplot with trend line

```
> trendscatter(Exam ~ Assign, data = Stats20x.df,xlab="Assignment")
```



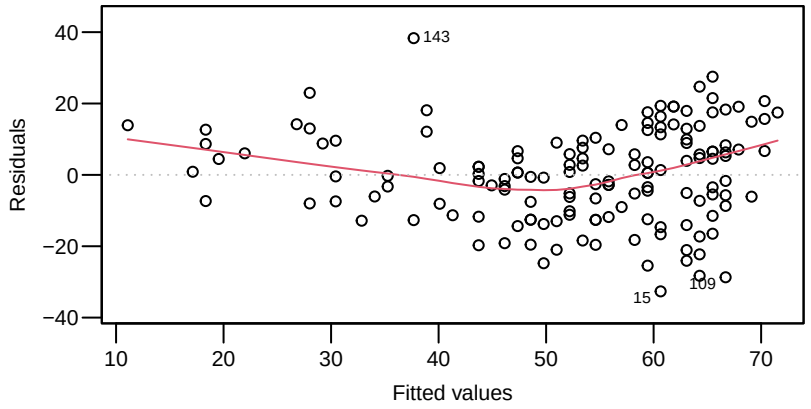
Sure looks like some curvature, but, at least the scatter looks fairly constant around this curve.

Exam vs. assignment marks...

Simple linear model

Let's fit a simple linear model to these data and see if it works out or not.

```
> examassign.fit=lm(Exam~ Assign,data = Stats20x.df)
> plot(examassign.fit,which=1)
```



Not surprisingly, we still have a curved relationship.

Exam vs. assignment marks...

Dealing with curvature

The assumption of identical distribution with expected value of 0 looks to be questionable here. There tend to be more negative residuals in the middle, but more positive residuals at the extremes of the fitted values.

Potential solution – add a quadratic (squared term) for `Assign`.

Section 4.2
Fitting a quadratic model

The quadratic curve

The standard notation for a quadratic curve is[1]

$$y = ax^2 + bx + c$$

Here we will use different notation: $\beta_0 \equiv c$, $\beta_1 = b$ and $\beta_2 = a$ and use the quadratic curve to describe the expected value of our dependent variable y. That is,

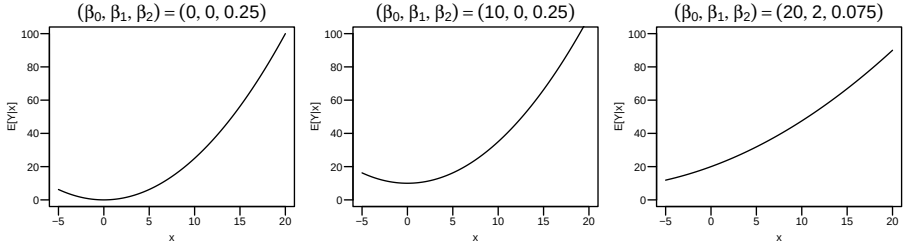
$$\mathrm{E}[Y|x] = \beta_0 + \beta_1 x + \beta_2 x^2$$

This is a linear model with explanatory terms x and x^2 – remember, the intercept β_0 is implicitly included.

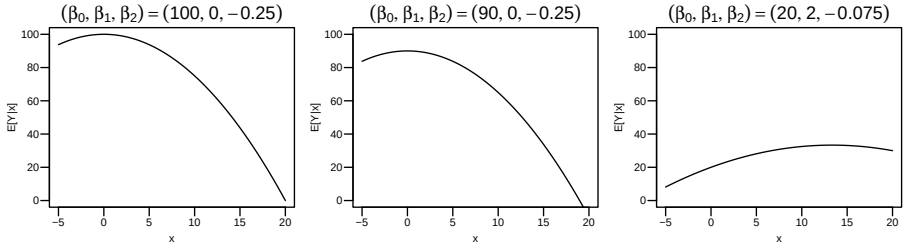
[1] If you have done a bit of calculus, then you might recall that the roots (the values of x that give $y = 0$) of a quadratic are $\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$.

The quadratic curve...

If $\beta_2 > 0$, then the quadratic has slope that increases with increasing x:



If $\beta_2 < 0$, then the quadratic has slope that decreases with increasing x:



How can a quadratic be a linear model?

(Non-examinable)

Throughout this course, when we use the term "linear model" we mean a model that is linear with respect to the β coefficients.

This mean that the derivative of the linear model with respect to any β coefficient is a constant.

The quadratic curve model

$$E[Y|x] = \beta_0 + \beta_1 x + \beta_2 x^2$$

is a quadratic model for x.

The derivatives of this quadratic with respect to β_0, β_1 and β_2 are 1, x and x^2, respectively. These derivatives are all considered "constants" because they do not depend on any β coefficient.

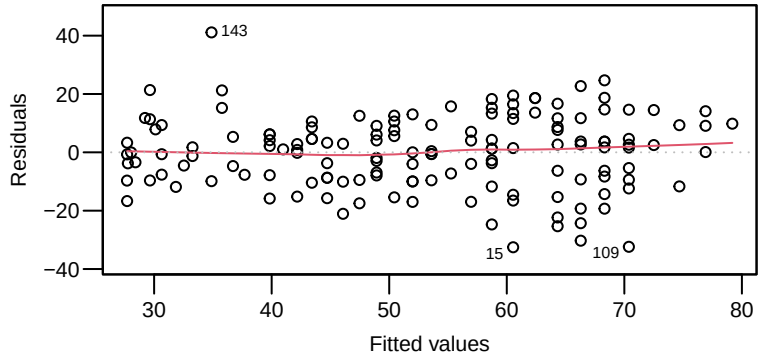
That is, the quadratic (in x) model is linear in β_0, β_1 and β_2.

Exam vs. assignment marks...

Adding a squared term

Add a squared term for `Assign` via `I(Assign^2)`, like this:[2]

```
> examassign.fit2=lm(Exam~ Assign + I(Assign^2), data = Stats20x.df)
> plot(examassign.fit2,which=1)
```



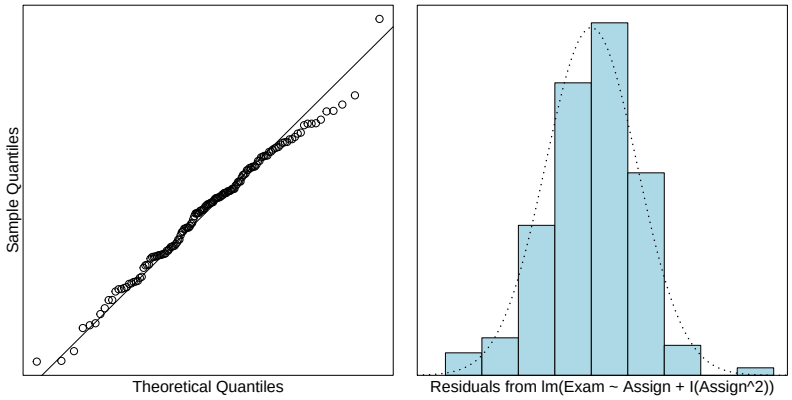
That is looking much better.

[2] **NOTE:** In the `lm` formula it is necessary to enclose the `Assign^2` term inside `I()` so that `lm` can make sense of it.

Exam vs. assignment marks...

Normality check of the quadratic model

```
> normcheck(examassign.fit2)
```

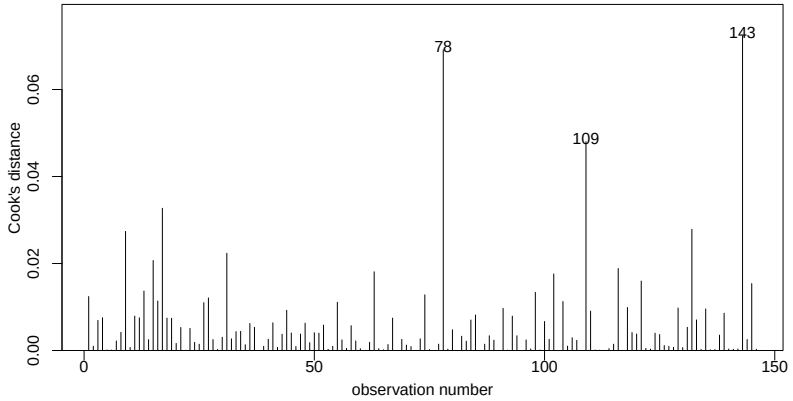


Looking good. There is one potential outlier. Let us check if it is influential.

Exam vs. assignment marks...

Influence check of the quadratic model

```
> cooks20x(examassign.fit2)
```



No high influence points.

Exam vs. assignment marks...

The fitted models

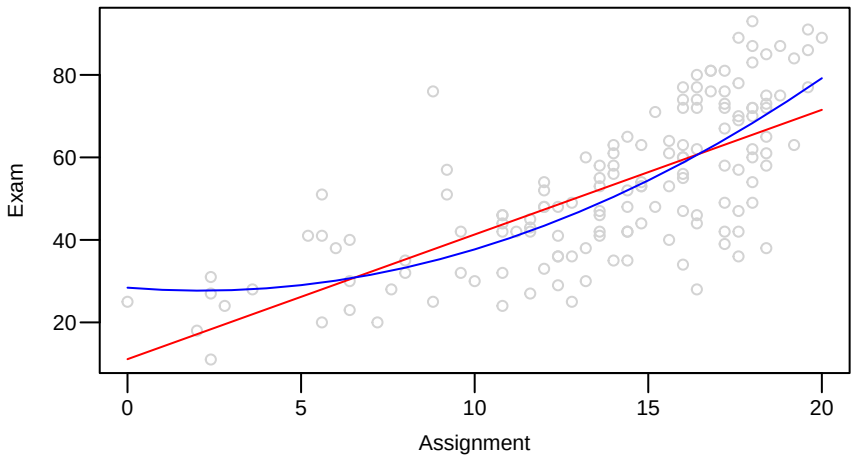
We have fitted a quadratic to see if we can add the 'curviness' in the relationship between test score and exam mark into our model.

Let us compare the two models visually – model 1 (linear) in red and model 2 (quadratic) in blue.

```
> plot(Exam~ Assign, data = Stats20x.df, xlab="Assignment")
> x=0:20 #Assignment values at which to predict exam mark
> ## Plot model 1
> lines(x, predict(examassign.fit,data.frame(Assign=x)), col="red")
> ## Plot model 2
> lines(x, predict(examassign.fit2,data.frame(Assign=x)), col="blue")
```


Exam vs. assignment marks...

The fitted models...



Exam vs. assignment marks...

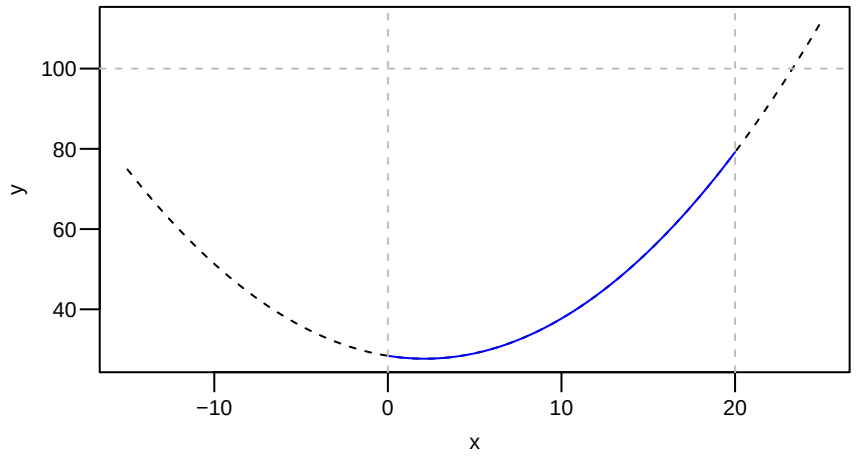
The fitted quadratic model

To plot the quadratic over a wider range of x (=Assign) values we can use the following code:

```
> x=seq(-15, 25,by =.10) #Sequence of from -15 to 25, in steps of 0.1
>
> y=predict(examassign.fit2,newdata = data.frame(Assign=x))
> plot(y~x, type="l",lty=2)
>
> ## The bits we want,  0<=x<=20 - N.B. Here & (ampersand) = AND
> lines(x[x>=0&x<=20],y[x>=0&x<=20],col="blue")
>
> ## The range of assign & exam respectively
> abline(v=range(Stats20x.df$Assign),lty=2, col="grey")
> abline(h=c(0,100),lty=2, col="grey")
```


Exam vs. assignment marks...

The fitted quadratic model...



Exam vs. assignment marks...

Comparison of straight line and quadratic models

```
> summary(examassign.fit)   ## Straight line model

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  11.0874     3.5954   3.084  0.00245 **
Assign        3.0222     0.2478  12.195  < 2e-16 ***
---
Residual standard error: 13.15 on 144 degrees of freedom
Multiple R-squared:  0.508,Adjusted R-squared:  0.5046
F-statistic: 148.7 on 1 and 144 DF,  p-value: < 2.2e-16
```

```
> summary(examassign.fit2)  ## Model with quadratic term

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 28.41396    5.99081   4.743 5.05e-06 ***
Assign      -0.68172    1.07242  -0.636 0.525999
I(Assign^2)  0.16102    0.04545   3.542 0.000536 ***
---
Residual standard error: 12.65 on 143 degrees of freedom
Multiple R-squared:  0.5477,Adjusted R-squared:  0.5414
F-statistic: 86.59 on 2 and 143 DF,  p-value: < 2.2e-16
```


Exam vs. assignment marks...

Comparison of straight line and quadratic models...

The small P-value ($= 0.000536$) for testing $H_0 : \beta_2 = 0$ tells us that that the quadratic term is statistically significant. Our model went from:

$Exam_i = \beta_0 + \beta_1 \times Assign_i + \varepsilon_i$ where $\varepsilon_i \overset{iid}{\sim} N(0, \sigma^2)$ to

$Exam_i = \beta_0 + \beta_1 \times Assign_i + \beta_2 \times Assign_i^2 + \varepsilon_i$ where $\varepsilon_i \overset{iid}{\sim} N(0, \sigma^2)$.

Note that the coefficient $\beta_2 > 0$ associated with the `I(Assign)^2` term results in an improvement in expected exam score that 'accelerates' as `Assign` increases.

We could consider removing the non-significant 'straight line' `Assign` term. **What would you do?**

We have done a better job of modelling this data by adding this extra term, and the R^2 explained another 4% of the total variation.

MORAL: If it looks like a curve then fit a curve – provided the scatter about the curve is constant (**EOV**).

Section 4.3
Relevant R-code

Most of the R-code you need for this chapter

If you suspect the relationship between your x and y variables follows a curve rather than a straight line (as revealed in the plot of residuals vs fitted values), and the scatter remains constant around this curve, then fit a quadratic:

```
> examassign.fit2=lm(Exam~ Assign + I(Assign^2), data = Stats20x.df)
> #Check the residual plot again - hopefully the curvature is gone.
> plot(examassign.fit2,which=1)
```

NOTE: If the null hypothesis $H_0 : \beta_2 = 0$ is **not** rejected (i.e, P-value> 0.05), then our preferred model would be the simple linear regression model.

Case Study 4.1: Exam vs Assignment mark

Tou Ohone Andate - staff number 1234567

Problem

We wish to quantify the relationship between exam mark and assignment score. In particular, we want to estimate the typical exam mark for all students when the assignment marks are 0, 10, and 20.

The variables of interest are:

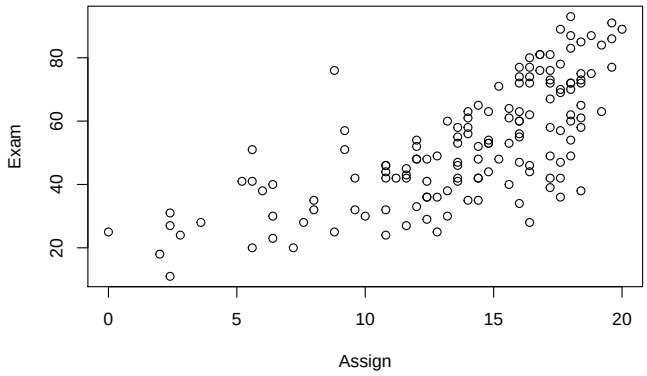
- `Exam`: Exam mark out of 100.
- `Assign`: Assignment mark out of 20.

Question of Interest

We want to build a model to estimate exam marks with assignment marks. In particular, we want to estimate the typical exam mark for all students when the assignment marks are 0, 10, and 20.

Read in and Inspect the Data

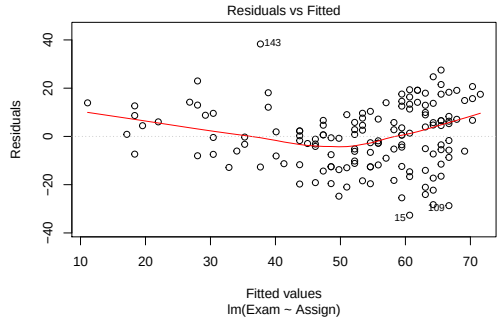
```
Stats20x.df = read.table("STATS20x.txt", header = T)
plot(Exam ~ Assign, data = Stats20x.df)
```



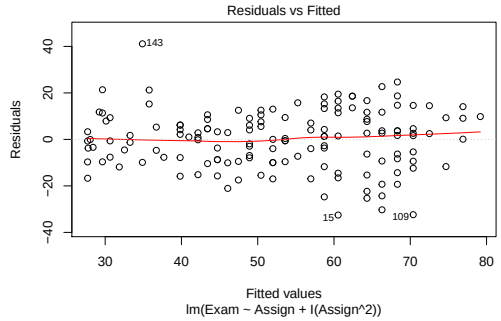
We see an increasing relationship between assignment marks and exam marks. However, this relationship does not seem very strong, and there is some suggestion of curvature in the relationship. We'll need to fit a linear model and scrutinise the residual plot to verify the extent of the curvature.

Model Building and Check Assumptions

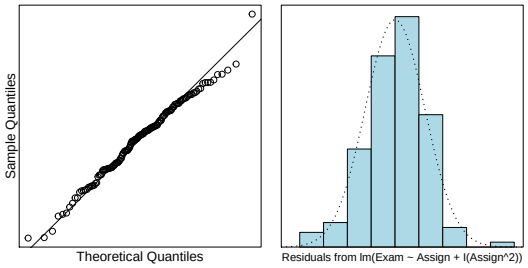
```
examAssign.fit = lm(Exam ~ Assign, data = Stats20x.df)
plot(examAssign.fit, which = 1) # We can use this code as an alternative to eovcheck
```



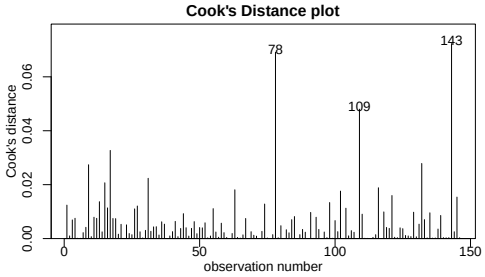

```
examAssign.fit2 = lm(Exam ~ Assign + I(Assign^2), data = Stats20x.df)
plot(examAssign.fit2, which = 1)
```




```
normcheck(examAssign.fit2)
```




```
cooks20x(examAssign.fit2)
```




```
summary(examAssign.fit2)
```

```
##
## Call:
## lm(formula = Exam ~ Assign + I(Assign^2), data = Stats20x.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -32.541  -9.149   1.273   9.087  41.116
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 28.41396    5.99081   4.743 5.05e-06 ***
## Assign      -0.68172    1.07242  -0.636 0.525999
## I(Assign^2)  0.16102    0.04545   3.542 0.000536 ***
## ---
```



```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.65 on 143 degrees of freedom
## Multiple R-squared:  0.5477, Adjusted R-squared:  0.5414
## F-statistic: 86.59 on 2 and 143 DF,  p-value: < 2.2e-16
```

```
confint(examAssign.fit2)
```

```
##                   2.5 %     97.5 %
## (Intercept) 16.57197462 40.2559372
## Assign      -2.80156333  1.4381240
## I(Assign^2)  0.07117039  0.2508643
```

Get additional Prediction Intervals

Note, since we are after typical exam marks we have use confidence intervals rather than prediction intervals.

```
predAssign.df = data.frame(Assign = c(0, 10, 20))
predict(examAssign.fit2, predAssign.df, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 28.41396 16.57197 40.25594
## 2 37.69849 34.29319 41.10379
## 3 79.18650 73.61953 84.75346
```

Method and Assumption Checks

The scatter plot of exam mark vs assignment mark suggested curvature in the relationship.

We began with a linear model to describe exam marks with assignment marks. The residual plot from the fit of a simple linear model showed fairly constant scatter but had strong curvature. So, a quadratic term was added to the linear model.

All model assumptions look satisfied once we added the quadratic term to the linear model.

Our final model is

$$Exam_i = \beta_0 + \beta_1 \times Assign_i + \beta_2 \times Assign_i^2 + \epsilon_i,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Our model explained 55% of the variability in the students' final exam marks.

Executive Summary

We were interested in building a model to estimate exam marks with assignment marks.

The relationship between expected exam mark and assignment score modelled was quadratic.

Here, for a one mark increase in assignment score, the increase in expected exam score was greater as assignment score increased. For example, there was little difference in expected exam score for those getting 7 or 8 in assignments, but a much bigger difference for those getting 17 or 18.[1]

For assignment marks of 0, 10 and 20, the estimate expected exam marks were between 16.6 to 40.3, 34.3 to 41.1, and 73.6 to 84.8, respectively.

[1] What could be causing this? Cheating on assignments? Over-zealous lab demonstrators giving too many hints?

Chapter 5:
Linear models with a 2-level categorical (factor) explanatory variable

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

- Using a 2-level categorical variable as an explanatory variable in a linear model by using indicator variables
- Putting the two-sample *t*-test into the linear model framework
- Relevant R-code.

Section 5.1
Using categorical variables as explanatory variables by using indicator variables

New Example – Exam marks vs Attendance

We have gained some understanding about STATS 20x students' final exam marks and how they are related to test and assignment scores, both of which are numeric explanatory variables.

Here, we are going to see if class attendance helps to explain exam score, where class attendance (did or did not) is a categorical explanatory variable.

I am pretty sure we know there is going to be a relationship, but let us answer the question anyway. This also lets us estimate the magnitude of the "attendance effect".

Exam marks vs Attendance

The particular variables of interest

`Exam` the student's Exam mark (out of 100)

`Attend` whether the student regularly attended lectures or not – *Yes* or *No*.[1]

Note: As always, our question really is about the 'typical' or average relationship. Some students may attend regularly but not do well in the exam, and vice-versa. That is part of the statistical variability, which we can deal with.

[1] This was measured by taking 4 rolls throughout the semester of lecture attendance. If a student was present for at least 3 of those 4 rolls, then they were recorded as a regular 'attender'.

Exam marks vs Attendance. . .

Preliminary exploration of the data

```
> ## Invoke the s20x library
> library(s20x)
> ## Importing data into R
> Stats20x.df = read.table("Data/STATS20x.txt", header=T)
> ## Change Attend from a character variable to a factor variable
> Stats20x.df$Attend = as.factor(Stats20x.df$Attend)
> ## Examine the data
> Stats20x.df$Attend[1:20]
 [1] Yes Yes Yes Yes No  Yes Yes No  Yes Yes No  Yes No  No  No  Yes Yes No  Yes
[20] Yes
Levels: No Yes
```

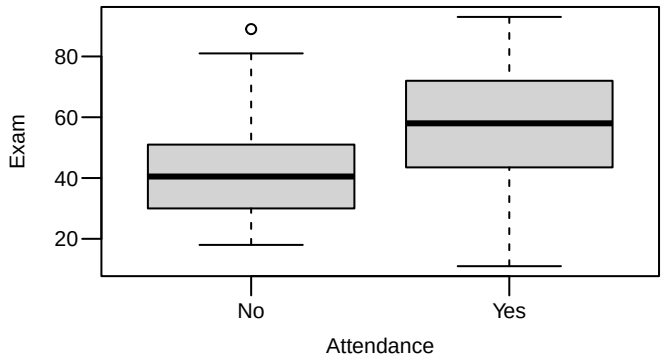
The `Attend` variable has been formatted as a factor variable with two levels - `Levels: No Yes`.

By default, `No` is the first level and `Yes` is the second level, because of alphabetical order – but this can be changed if need be. In this case, we wish to contrast the attenders (`Yes`) against the non-attenders (`No`) and (as you will see later) this ordering of the levels suits us.

Exam marks vs Attendance. . .

Preliminary exploration of the data. . .

```
> summaryStats(Stats20x.df$Exam,Stats20x.df$Attend)
    Sample Size     Mean Median  Std Dev Midspread
No           46 42.21739   40.5 16.34206     20.50
Yes         100 57.78000   58.0 17.67757     28.25
> plot(Exam~ Attend, data = Stats20x.df, xlab="Attendance")
```



NOTE: `Attend` is a factor, so `R` used a boxplot to display the data.

Exam marks vs Attendance. . .

Preliminary exploration of the data. . .

Here, we are asking how a student's attendance mark (x) is related to a their exam (y) mark.

As we are interested in the 'typical' student, we want to see what the underlying trend is and how students vary (scatter) about that trend.

Looking at the boxplot, it seems that regular attendance is associated with higher exam scores. Also, the equality of variance assumption seems okay.

Exam marks vs Attendance...

Preliminary exploration of the data...

If you wish to use `trendscatter`, we need to create a new x variable that is numerical. We can do this by recoding non-attenders as 0s and attenders as 1s.

Here is the code for doing this recoding, with some checks to see that it has been successful.

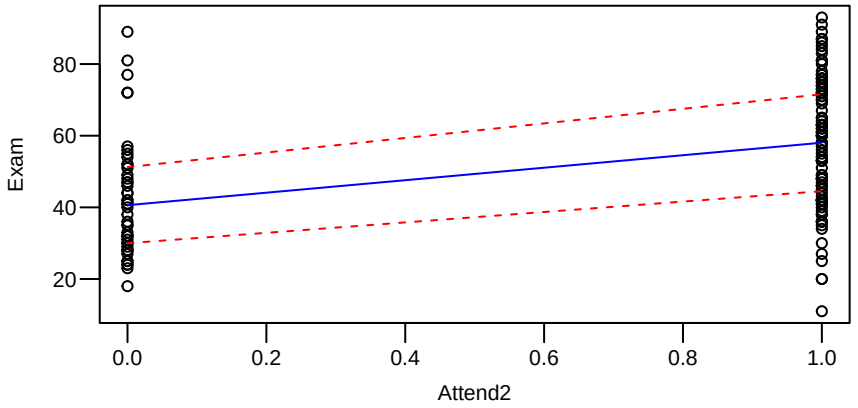
```
> #Make a new variable Attend2 which is 1 if Attend = "Yes" and 0 otherwise
>
> #Note how we use two equal signs, ==, to test equality
> Stats20x.df$Attend2 = as.numeric(Stats20x.df$Attend=="Yes")
> with(Stats20x.df, table(Attend, Attend2))
      Attend2
Attend   0   1
   No   46   0
   Yes   0 100
```

The `with` function lets us use the variables in the dataframe without having to type the dataframe name every time.

Exam marks vs Attendance...

Preliminary exploration of the data...

```
> trendscatter(Exam~ Attend2, data = Stats20x.df)
```



EOV assumption seems valid.

Exam marks vs Attendance...

Fitting a linear model using `Attend2`

Note that `Attend2` is a numeric explanatory variable (albeit with only two different values). So, we can fit a simple linear regression model to see how well `Attend2` explains `Exam`. What would this model tell us?

The linear model for the expected value of `Exam` is

$$\texttt{E[Exam|Attend2]} = \beta_0 + \beta_1 \texttt{Attend2} \ .$$

and so the full model equation is

$$\texttt{Exam}_i = \beta_0 + \beta_1 \texttt{Attend2}_i + \varepsilon_i \text{ where } \varepsilon_i \overset{iid}{\sim} N(0, \sigma^2) \ .$$

If we use the notation `E[Exam|Attend2=0]` to denote the expected value of `Exam` when `Attend2`=0 then we have

$$\texttt{E[Exam|Attend2=0]} = \beta_0 \ .$$

Similarly, when `Attend2`=1

$$\texttt{E[Exam|Attend2=1]} = \beta_0 + \beta_1 \ .$$

Exam marks vs Attendance...

Fitting a linear model using `Attend2`...

We see that β_0 is the mean exam mark for non-attenders and $\beta_0 + \beta_1$ is the mean mark for attenders, and so β_1 is the difference in mean mark for attenders compared to non-attenders.

Our question of interest was to make inference about the "attendance effect" – this "attendance effect" is simply β_1, so we can use the methods of inference about the slope of a linear model that we have already seen.

In particular, we know how to test the null hypothesis of no attendance effect, $H_0 : \beta_1 = 0$, and how to obtain confidence intervals. Let's give it a go...

Exam marks vs Attendance...

Fitting a linear model using Attend2...

```
> examattend2.fit = lm(Exam~ Attend2, data = Stats20x.df)
> summary(examattend2.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   42.217      2.547  16.578  < 2e-16 ***
Attend2       15.563      3.077   5.058 1.27e-06 ***
---
Residual standard error: 17.27 on 144 degrees of freedom
Multiple R-squared:  0.1508,Adjusted R-squared:  0.145
F-statistic: 25.58 on 1 and 144 DF,  p-value: 1.271e-06
```

Having to create the indicator variable[2] `Attend2` is a nuisance. The good news is that the linear model function `lm` implicitly does this for us.

Here we made the choice to recode non-attenders as 0s and attenders as 1s, rather than the other way around. This was deliberate – it is the same choice that `lm` would make....see below.

[2] So-called because it indicates whether attendance is No or Yes. Some people call these dummy variables.

Exam marks vs Attendance...

Fitting a linear model using Attend

We now fit a linear model to these data using the factor variable `Attend`. The `lm` function automatically attributes `Attend=="No"` the zero value [3] because **No** comes before **Yes** in the alphabet, and `Attend=="Yes"` is attributed the value 1.

```
> examattend.fit = lm(Exam~ Attend, data = Stats20x.df)
> summary(examattend.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   42.217      2.547  16.578  < 2e-16 ***
AttendYes     15.563      3.077   5.058 1.27e-06 ***
---
Residual standard error: 17.27 on 144 degrees of freedom
Multiple R-squared:  0.1508,Adjusted R-squared:  0.145
F-statistic: 25.58 on 1 and 144 DF,  p-value: 1.271e-06
```

What is the difference? Nothing really – slightly different formats of the coefficient names as `Attend2` is numerical and `Attend` is a categorical variable (factor).

[3] This is called the **baseline** or reference level of the factor variable.

Section 5.2
Model checking and inference

Exam marks vs Attendance...

The fitted model

Note that the p-value for `Attend` is very small, so we conclude that there is indeed a significant effect of attendance.

The estimates of β_0 and β_1 are

```
(Intercept)   AttendYes
   42.21739    15.56261
```

That is, (formatted to 2 decimal places) $\hat{\beta}_0 = 42.22$ and $\hat{\beta}_1 = 15.56$.

Using these estimated coefficients[4] our estimated values for the exam score of a 'typical' student are:

$$\widehat{\texttt{Exam}} = 42.22 + 15.56 \times \texttt{Attend2}$$

or

$$\widehat{\texttt{Exam}} = \begin{cases} 42.22 & \text{, for non-attenders and} \\ 42.22 + 15.56 & \text{, for attenders.} \end{cases}$$

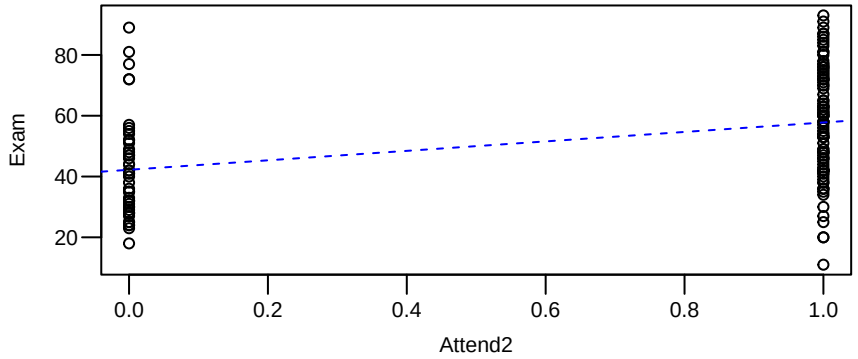
[4] Subject to verification of the model assumptions - stay tuned

Exam marks vs Attendance...

The fitted model...

Let's visualize the fit. Here we are plotting the 'best' estimated straight line that we obtained from fitting our model using the indicator variable `Attend2`.

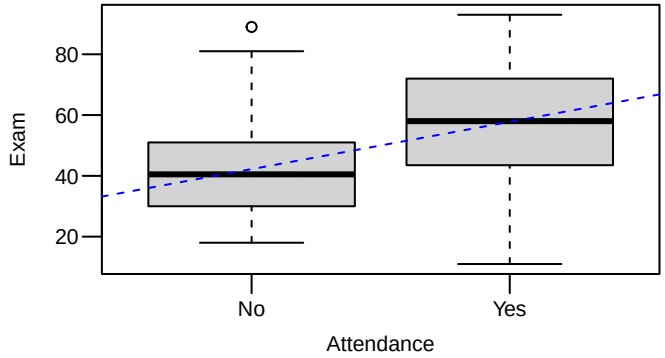
```
> plot(Exam ~ Attend2, data = Stats20x.df)
> ## Add the lm estimated line to this  plot  where a=intercept, b=slope
> abline(coef(examattend.fit),lty=2, col="blue")
```



Exam marks vs Attendance...

The fitted model...

Here is the same thing using the factor variable `Attend`, with the fit overlaid on the boxplots.



The two plots above essentially present the same information, repackaged in a slightly different way.

Exam marks vs Attendance...

Checking assumptions

We should check that our assumptions hold before we report (or use) the results from this analysis. Remember, we require independence, identical distribution and normality of the random components

$$\varepsilon_i \overset{iid}{\sim} N(0, \sigma^2).$$

The assumptions (in order of importance):

iid – independence. We check this by investigating how we obtained the data.

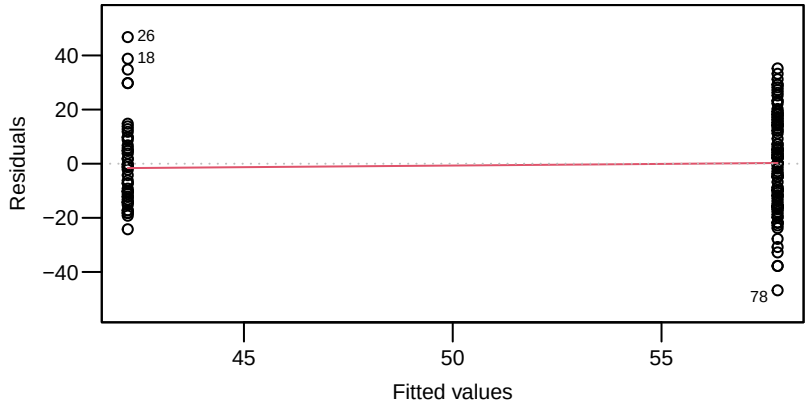
i**i**d – identically distributed. This should result in the variation of the residuals being roughly constant (regardless of the fitted value) and the residuals more-or-less averaging around zero. We can use `plot()` with the `which=1` argument.

ii**d** – Normality. We only check this having validated the first two assumptions, using `normcheck`.

Exam marks vs Attendance...

Checking our assumptions

```
> plot(examattend.fit,which=1)
```

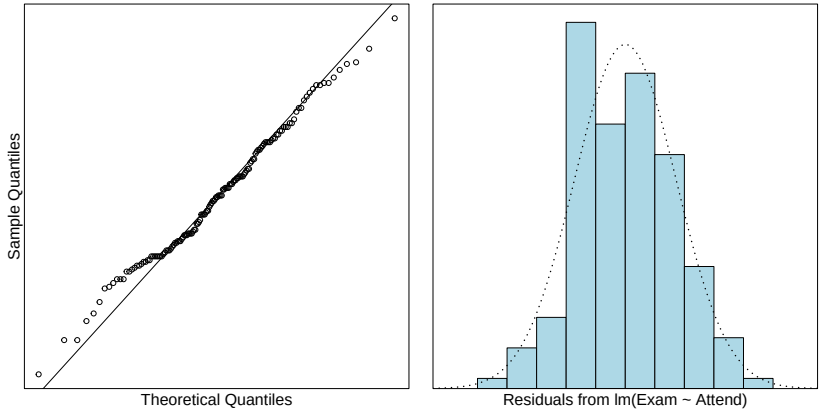


No cause for concern.

Exam marks vs Attendance...

Checking our assumptions...

```
> normcheck(examattend.fit)
```

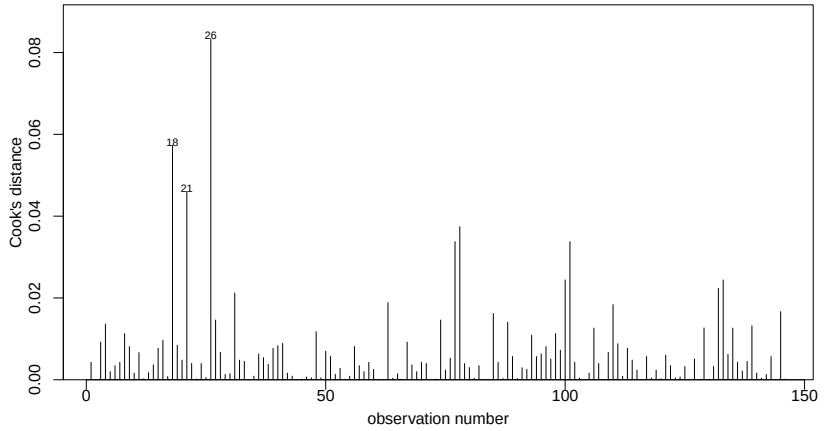


Looks good - the residuals appear to have a near normal distribution.

Exam marks vs Attendance...

Checking our assumptions...

```
> cooks20x(examattend.fit)
```



No unduly influential points.

Inference about Exam marks vs Attendance

Testing the null hypothesis

The model assumptions look to be reasonably well satisfied, so we can use the fitted model to make statistical inference (i.e., to answer questions of interest).

We begin by testing the null hypothesis that there is no effect of the explanatory variable `Attend`.[5] Recall that this is $H_0 : \beta_1 = 0$.

```
> coef(summary(examattend.fit))
            Estimate Std. Error   t value     Pr(>|t|)
(Intercept) 42.21739   2.546517 16.578482 3.358318e-35
AttendYes   15.56261   3.076969  5.057773 1.271370e-06
```

The P-value for `AttendYes` is testing the null hypothesis $H_0 : \beta_1 = 0$. It is highly statistically significant $p = 1.27 \times 10^{-6}$, which is just over 1 in a million. We have extremely strong evidence that attendance is related to exam score.

[5] In this case, the null hypothesis corresponds to the `Exam` scores being iid.

Inference about Exam marks vs Attendance...

Calculating confidence intervals for effect size

We can get confidence intervals for β_1 (and β_0 if need be) by:

```
> confint(examattend.fit)
                2.5 %   97.5 %
(Intercept) 37.184009 47.25077
AttendYes    9.480749 21.64447
```

Here we can say that, on average, regular attenders will obtain an increased exam mark of between 9.5 to 21.6 compared to non-attenders.

Alternative wording would be: the expected exam mark of a student who regularly attends class is 9.5 to 21.6 marks higher than that of a non-attendee.

Inference about Exam marks vs Attendance...

Calculating confidence and prediction intervals

We might also want to estimate and/or predict exam marks based on attendance status.

Recall that we need to make a new dataframe containing the values of the explanatory variable to be used for the prediction. In this case, that is just "No" and "Yes".

```
> ## Create data frame of values of interest: Attend=="Yes" and "No"
> ## Make sure that the names of vars are exactly the same as in the data frame
> preds.df = data.frame(Attend = c("No", "Yes"))
> predict(examattend.fit, preds.df, interval = "confidence")
       fit      lwr      upr
1 42.21739 37.18401 47.25077
2 57.78000 54.36619 61.19381
> predict(examattend.fit, preds.df, interval = "prediction")
       fit       lwr      upr
1 42.21739  7.710259 76.72452
2 57.78000 23.471673 92.08833
```


Inference about Exam marks vs Attendance...

Confidence and prediction intervals

Here, we estimate the exam mark for non-regular attenders is between 37.2 and 47.3 on average, whereas for those who regularly attend it is between 54.4 and 61.2.

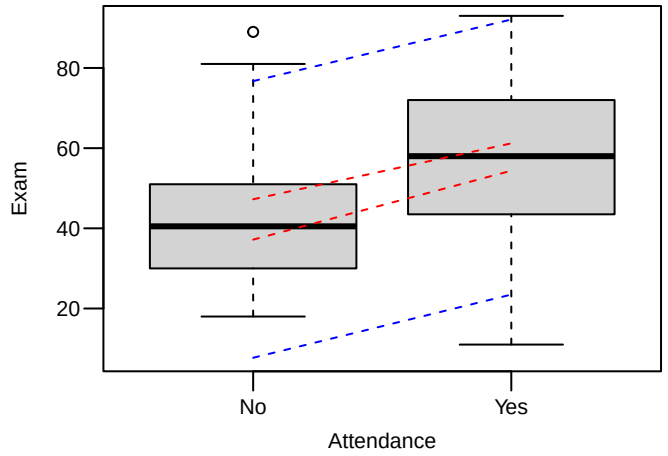
For any individual student, if they are a non-regular attender then we predict their exam mark to be between 7.7 and 76.7, or between 23.5 and 92.1 if they do attend regularly.

The prediction intervals are very wide – which is not surprising as regular attendance only explains 15% of the variation in exam score, and there is plenty of variability between individual students.

Inference about Exam marks vs Attendance...

Confidence and prediction intervals

Here is what the confidence (red)/prediction intervals (blue) look like:



Section 5.3
Putting the two-sample t-test into the linear model framework

Two-sample *t*-test in disguise

We have just done an analysis to see if two groups (attendees and non-attendees) differ in expected value. You have encountered this scenario previously – the two-sample *t*-test.

By default, the `t.test` function in `R` relaxes the equality of variance assumption, so we have to explicitly tell it not to (using the `var.equal=TRUE` argument) if we want to reproduce our `lm` results exactly.

```
> t.test(Exam~ Attend, var.equal=TRUE, data = Stats20x.df)

Two Sample t-test

data:  Exam by Attend
t = -5.0578, df = 144, p-value = 1.271e-06
alternative hypothesis: true difference in means between group No and group Yes is
95 percent confidence interval:
 -21.644468  -9.480749
sample estimates:
 mean in group No mean in group Yes
         42.21739          57.78000
```


Exam marks vs Attendance...

Two-sample *t*-test

The two-sample *t*-test has a variant which relaxes the EOV assumption. This is known as the **Welch** form of the *t*-test, and is the default in the `t.test` function.

```
> t.test(Exam~ Attend, var.equal=FALSE, data = Stats20x.df)
```

or just

```
> t.test(Exam~ Attend, data = Stats20x.df)

Welch Two Sample t-test

data:  Exam by Attend
t = -5.2076, df = 94.09, p-value = 1.122e-06
alternative hypothesis: true difference in means between group No and group Yes is
95 percent confidence interval:
 -21.496121  -9.629096
sample estimates:
 mean in group No mean in group Yes
         42.21739          57.78000
```


Exam marks vs Attendance...

Two-sample *t*-test...

The Welch form loses some degrees of freedom because it adds more uncertainty as we have to now estimate the variability of the sample in each group (attenders and non-attenders) rather than 'pooling' them based on the EOV assumption.

In this case, there is negligible difference between the standard and **Welch** forms of the two-sample *t*-test.

Summary

We have now seen that `lm` can be used when the explanatory variable is numeric (e.g., `Test` or `Assign`), and also when the explanatory variable is categorical (e.g., `Attend`).

When the explanatory variable is categorical, `lm` automatically creates numeric indicator variables to use in the model formula. The indicator variables indicate the category level (relative to the baseline level) and take the value 0 or 1 – for this reason they are sometimes referred to as indicator variables.

The natural question to ask here is: can we use `Test` and `Attend` *together* to explain exam score?

Stay tuned...

Section 5.4
R tips and relevant code

R tips and tricks

Use of -1 in model formulae

In some situations it is useful to fit the model without a baseline level for the intercept.

This is easy, just add -1 to the model formula.

```
> NoBaseline.fit=lm(Exam~ Attend-1, data = Stats20x.df)
> summary(NoBaseline.fit)
```

```
Coefficients:
          Estimate Std. Error t value Pr(>|t|)
AttendNo    42.217      2.547   16.58   <2e-16 ***
AttendYes   57.780      1.727   33.45   <2e-16 ***
---
Residual standard error: 17.27 on 144 degrees of freedom
Multiple R-squared:  0.9064,Adjusted R-squared:  0.9051
F-statistic:   697 on 2 and 144 DF,  p-value: < 2.2e-16
> confint(NoBaseline.fit)
             2.5 %   97.5 %
AttendNo  37.18401 47.25077
AttendYes 54.36619 61.19381
```

Note: R^2 has no meaning when there is no intercept term in the model.

Most of the R-code you need for this chapter

You do not need to create indicator variables as R does this for you. It will choose the baseline for you, so be careful. You can change this if needed – you will see an example of this soon.

```
> examattend.fit = lm(Exam~ Attend, data = Stats20x.df)
```

This is equivalent to

```
> t.test(Exam~ Attend, var.equal=TRUE, data = Stats20x.df)
```

If it is clear that the two groups have massively different variances then one approach would be to abandon the use of a linear model and use the modified t-test without the equality of variance assumption[6]

```
> t.test(Exam~ Attend, var.equal=FALSE, data = Stats20x.df)
```

However, in most cases the technique shown in the next Chapter is a better way to cope with inequality of variance.

[6] The modified t-test approach will **never** be used in this class.

Case Study 5.1: Exam vs Attendance

Tou Ohone Andate - staff number 1234567

Problem

We wish to determine if regular attendance in class is associated with exam mark. In particular, we want to estimate the expected exam marks of attendees and non-attendees, and predict the actual exam scores of individual attendees and non-attendees.

The variables of interest are:

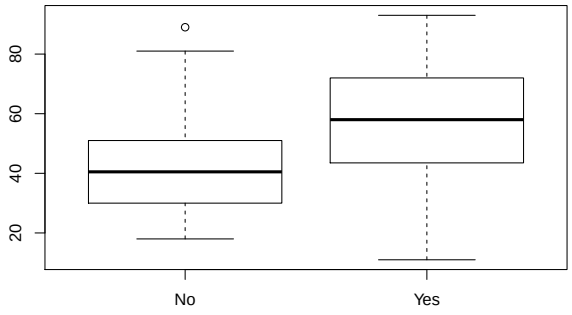
- `Exam`: Exam mark out of 100.
- `Attend`: A two-level categorical variable which has the levels `Yes` and `No`.

Question of Interest

We want to quantify the relationship between exam marks and attendance. In particular, we want to estimate the expected exam marks of attendees and non-attendees, and predict the actual exam scores of individual attendees and non-attendees.

Read in and Inspect the Data

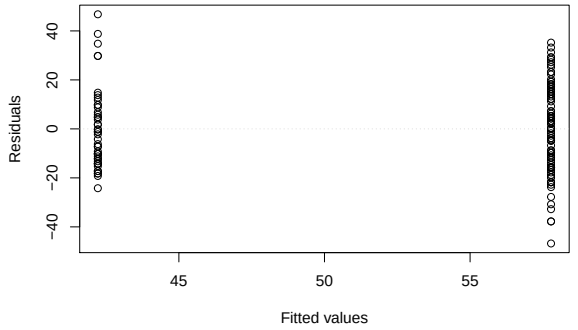
```
Stats20x.df = read.table("STATS20x.txt", header = T)
# Could of also used plot(Exam ~ Attend, data = Stats20x.df)
boxplot(Exam ~ Attend, data = Stats20x.df)
```



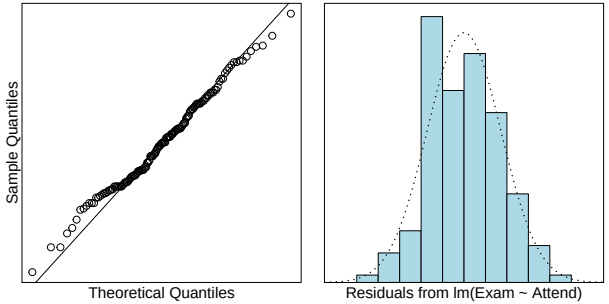
The exam marks for the students attended seem to be centred than the students who did not attend lectures regularly. The boxplots look reasonably symmetric in both groups.

Model Building and Check Assumptions

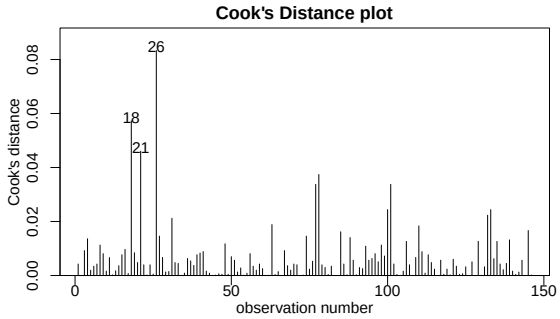
```
examAttend.fit = lm(Exam ~ Attend, data = Stats20x.df)
eovcheck(examAttend.fit)
```




```
normcheck(examAttend.fit)
```




```
cooks20x(examAttend.fit)
```




```
summary(examAttend.fit)
```

```
##
## Call:
## lm(formula = Exam ~ Attend, data = Stats20x.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -46.780 -13.108  -0.217  12.642  46.783
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   42.217      2.547  16.578  < 2e-16 ***
## AttendYes     15.563      3.077   5.058 1.27e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.27 on 144 degrees of freedom
## Multiple R-squared:  0.1508, Adjusted R-squared:  0.145
## F-statistic: 25.58 on 1 and 144 DF,  p-value: 1.271e-06
```

```
confint(examAttend.fit)
```

```
##                 2.5 %   97.5 %
## (Intercept) 37.184009 47.25077
## AttendYes    9.480749 21.64447
```

Get additional Prediction Intervals

```
predAttend.df = data.frame(Attend = c("No", "Yes"))

# Using `interval = "confidence"' to get CI for expected exam score
# for each level of Attend
predict(examAttend.fit, predAttend.df, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 42.21739 37.18401 47.25077
## 2 57.78000 54.36619 61.19381
```

```
# Using `interval = "predition"' to get PI for individual student's exam score
# for each level of Attend
predict(examAttend.fit, predAttend.df, interval = "prediction")
```

```
##        fit       lwr      upr
## 1 42.21739  7.710259 76.72452
## 2 57.78000 23.471673 92.08833
```

Method and Assumption Checks

We wish to explain exam marks with attendance, a two-level factor. So, we have fitted a linear model with a single explanatory dummy variable. (Note, this is equivalent to conducting a two-sample t-test).

Four non-attending students did unusually well (i.e., large positive residuals), but since the sample size was large, this will be of little consequence. Hence, all model assumptions were satisfied.

Our final model is

$$Exam_i = \beta_0 + \beta_1 \times Attend.Yes_i + \epsilon_i,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$. Here $Attend.Yes_i = 1$ if the student regularly attended (i.e., answered "Yes"), otherwise it is zero (i.e. "No").

Our model explained a small 15% of the variability in the students' final exam marks.

Executive Summary

We wanted to quantify the relationship between exam marks and attendance.

There was strong evidence that exam marks were higher for students who attend class versus students who didn't attend class ($P\text{-}value \approx 10^{-6}$). We estimate that regular attendance could increase their expected exam mark between 9.5 to 21.6 exam marks (out of 100).

The expected exam marks of non-attendees and attendees are between 37.2 to 47.3, and 54.4 to 61.2, respectively.

The predicted exam marks for individual non-attendees and attendees are between 7.7 to 76.7, and 23.5 to 92.1, respectively.

Our model only explains 15% of the variability in the students' final exam marks, this would not be very good for prediction. We can see this in how wide our prediction intervals are.

Chapter 6: Multiplicative linear models

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

- Mean versus median – which to use?
- Transforming the response variable using the log function
- Multiplicative models
- Why inference is about the median (not the mean)
- A multiplicative simple linear regression example
- A multiplicative two-sample t-test example
- Relevant `R`-code.

Section 6.1
Mean versus median – which to use?

Auckland house prices

In early 2021 one of the STATS 20x lecturers was looking to buy a house in a working-class suburb of Auckland, about 15 minutes by train to downtown.[1]

The lecturer downloaded 94 recent sales prices for the suburb, and put the prices ($1000's) in a text file called `AkldHousePrices.txt`. The lecturer just wants to know the typical house price in the suburb – there are no explanatory variables.

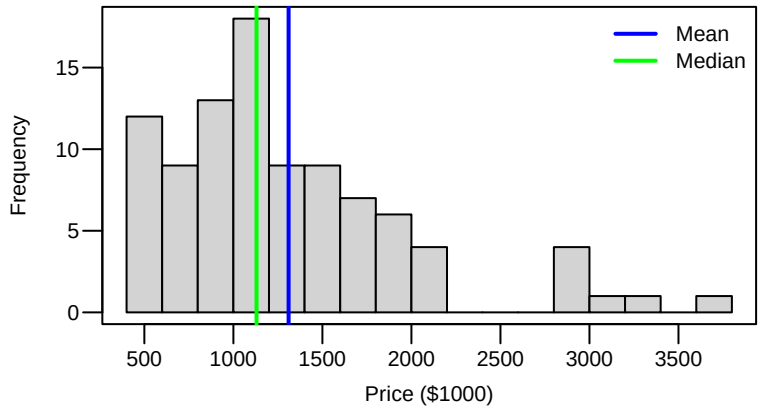
It sounds like an easy analysis. We just need to fit a null model. Let's take a look...

[1] Auckland house prices are very high by international standards.

Auckland house prices...

Inspect the data

```
> Houses.df=read.table("Data/AkldHousePrices.txt",header=T)
> hist(Houses.df$price, breaks=20,main="",xlab="Price ($1000)")
> abline(v = c(mean(Houses.df$price), median(Houses.df$price)),
+        col = c("blue", "green"), lwd = 2)
```



Auckland house prices...

Inspect the data...

```
> summary(Houses.df$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  450.0   832.5  1130.0  1310.1  1597.5  3710.0
```

Clearly, we are not dealing with data that come from a normal distribution. See how the (sample) median is markedly lower than the (sample) mean.

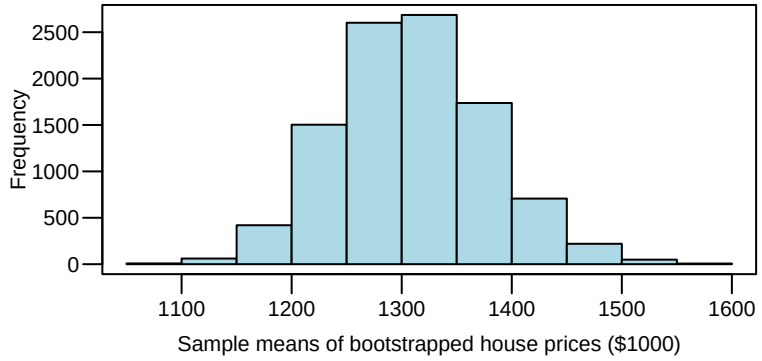
This type of right-skew distribution is very common when it comes to things involving money ($$$), resources, growth, salary, age, advantage and energy, to name but a few.

We can still make inference using a linear model for the expected house price because we can apply the **CLT** since we have a largish number of observations ($n = 94$). Alternatively, we can use the bootstrap since it does not require the assumption of normality.

Auckland house prices...

Inference about the mean

Here is histogram of 10,000 bootstrap sample means from resampling the house price data.



The approximate normality ot the bootstrapped sample means shows that the sample mean has an approximate normal distribution.

Auckland house prices...

Inference about the mean...

Here is the bootstrap 95% CI for the expected price, along with output from the null model.

```
> quantile(bootstrappedMeanPrices, c(.025, .975))
    2.5%    97.5%
1181.168 1452.875
```

```
> HousesNull.fit=lm(price~1, data=Houses.df)
> summary(HousesNull.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   1310.1       70.1   18.69   <2e-16 ***
---
Residual standard error: 679.7 on 93 degrees of freedom
```

```
> confint(HousesNull.fit)
                2.5 %   97.5 %
(Intercept) 1170.899 1449.313
```

Here we see that, to within less than NZ$10 000, the bootstrap CI and the model CI are the same. This is confirmation that the **CLT** works.

Auckland house prices...

Inference about the mean...

So we can safely say that the mean (i.e., expected) sale price for the entire suburb is somewhere between, say, NZ$1.17 million to NZ$1.45 million. However, the gross non-normality of the data prevents us from making prediction intervals for house prices.

Moreover, the estimated mean price of NZ$1.31 million is somewhat misleading.[2] Our house-hunting lecturer just wants a typical house, about midway in affordability at most. It would make more sense to estimate the **median** house price, since our lecturer will then know that half of the houses going on the property market will sell for less than that price.

In general, for highly skewed data the median is usually a better measure of 'typical' value than the mean.

[2] It is also not statistically robust since it is very sensitive to the number of expensive ($\geq$ \$3 million, say) houses sold

Auckland house prices...

Inference about the median

To estimate the median sale price of the entire suburb the natural estimate is the median of our sample:

```
> median(Houses.df$price)
[1] 1130
```

and we can use a bootstrap to get a 95% CI for the suburb median

```
> quantile(bootstrappedMedianPrices, c(.025, .975))
 2.5% 97.5%
 1040  1320
```

so we can say that the median sale price for the entire suburb is somewhere between, say, NZ$1.04 million to NZ$1.32 million.

The 95% CI for the median comes as something of a relief to our house-hunting lecturer. It is much more reasonable than the \$1.17 to \$1.45 million CI for the mean which was markedly higher due to the data being so right-skewed.

Inference about the median

In the above house price example we are working with iid data, so it is natural to use the sample median to estimate the population median.

We'll see in the next section that the linear model framework can also be used to make inference about the median provided that the logged response variable is approximately normally distributed. This approach has the advantage that it can also be applied to more general situations where we have explanatory variables that may be associated with the response variable.

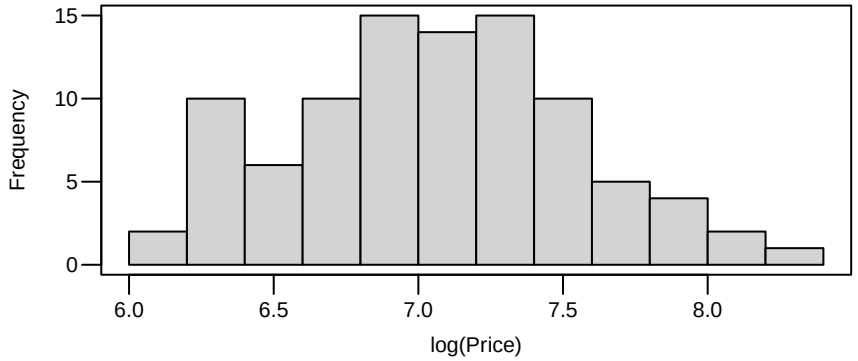
We'll also see that fitting linear models to logged response data results in the effects of explanatory variables acting multiplicatively on medians.

Section 6.2
Transforming the response variable using the log function

Auckland house prices...

Transforming the data

Let's consider making a transformation of the prices. In particular, the log transformation. Here is the histogram of `log(price)`.



This looks reasonably close to normal, so if we fit a linear model to these data then all inferences will be valid.

Auckland house prices...

Null model fitted to logged house price data

```
> LoggedPriceNull.fit=lm(log(price)~1, data=Houses.df)
> coef(summary(LoggedPriceNull.fit))
             Estimate Std. Error  t value      Pr(>|t|)
(Intercept) 7.060405 0.04974049 141.9448 1.628721e-110
```

```
> confint(LoggedPriceNull.fit)
               2.5 %  97.5 %
(Intercept) 6.96163 7.15918
```

Well that is interesting, but logged house prices don't mean much to anyone who is hoping to buy a house. The inference needs to be back-transformed to NZ$.

Since we've used the log transformation, the back-transformation is the exponential function `exp()`.

```
> exp(confint(LoggedPriceNull.fit))
                2.5 %   97.5 %
(Intercept) 1055.353 1285.856
```


Auckland house prices...

The effect of transforming

The above confidence interval is quite different from the one we calculated for the mean house price of the suburb. The reason for this is because the above CI is for the **median** house price.

To see why this is, let's take a look at what happens when we transform summary statistics using the `log()` and `exp()` functions:

Summaries of price:

```
> summary(Houses.df$price);
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  450.0   832.5  1130.0  1310.1  1597.5  3710.0
```

Summaries of log(price):

```
> summary(log(Houses.df$price))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  6.109   6.724   7.030   7.060   7.376   8.219
```

Back-transformed summaries of log(price):

```
> exp(summary(log(Houses.df$price)))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  450.0   832.5  1130.0  1164.9  1597.0  3710.0
```


Auckland house prices...

The effect of transforming...

Note that the sample mean changed after transforming then back-transforming, but the other summary values did not. This is because transforming and back-transforming does not change the order of the data – the smallest value will still be the smallest value, the middle value will still be the middle value.

Auckland house prices...

Why inference is about the population median

If the logged response variable is normally distributed, then on the log scale its population mean and median are exactly the same number.[3] In that case, a 95% confidence interval for the population mean of the logged response variable is also a 95% CI for the population median of the logged response variable.

Since the median is unaffected by transforming and back-transforming, it follows that back-transforming the above 95% confidence interval will give a 95% confidence interval for the population median on the original scale. However, it will not be a 95% confidence interval for the population mean.

We now have a recipe for calculating a confidence interval for the **median** – fit a linear model to the log-transformed data, calculate a 95% confidence interval for the **mean** (and hence **median**), and back transform.

[3] The sample mean and median may differ a little, as we saw with the logged house prices

Auckland house prices...

Inference about the median

Our back-transformed estimate ($\exp(\hat{\beta}_0)$) and 95% CI for the median suburb sale price are

```
> exp(coef(LoggedPriceNull.fit))
(Intercept)
   1164.917
> exp(confint(LoggedPriceNull.fit))
                2.5 %   97.5 %
(Intercept) 1055.353 1285.856
```

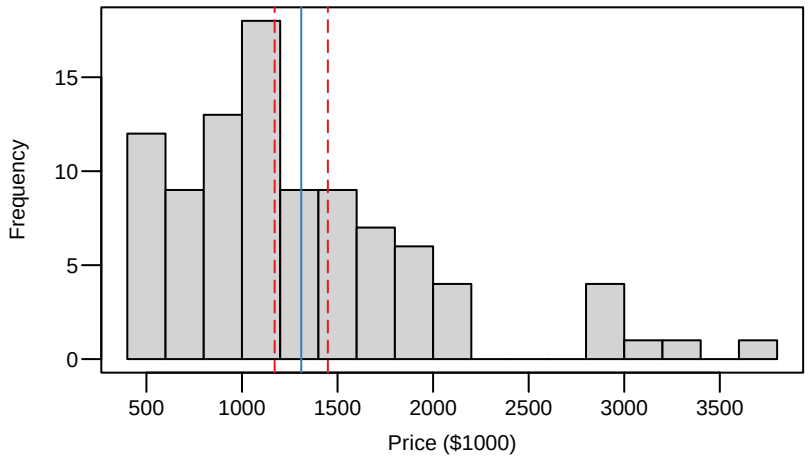
So we can say we are reasonably sure (95% confident) that the median house price is somewhere between NZ$1.06 and NZ$1.29 million.

These values differ a little from the sample median (NZ$1130) and the 95% bootstrap CI of NZ$1040 to NZ$1320 we saw in the previous section. This is because the machinery being used is different.

Auckland house prices...

Inference for the house price data

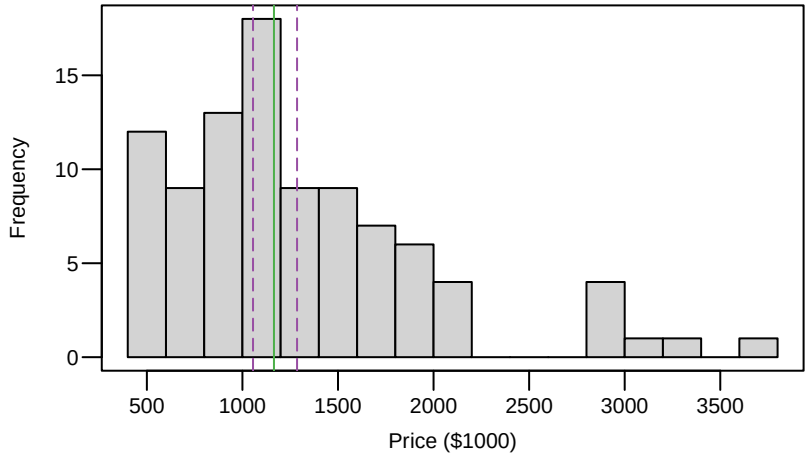
Here is the confidence interval for the mean suburb price with the sample mean shown in blue.



Auckland house prices...

Inference for the house price data...

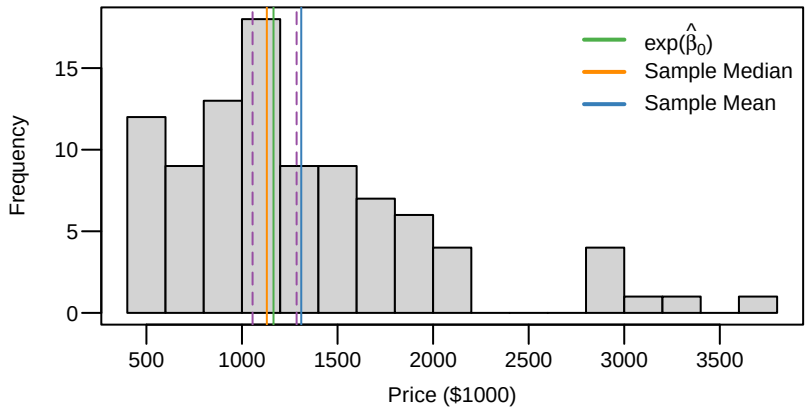
Here is the confidence interval for the median suburb price with the estimate obtained from the linear model approach ($\exp(\hat{\beta}_0)$) shown in green.



Auckland house prices...

Inference for the house price data...

Here is the above plot with the sample mean and sample median shown as well.



Section 6.3
The log function turns multiplicative effects in to additive effects

Why the log transformation?

In the above house price example it would have been possible to use many other choices for our transformation of the prices. How did we know that the log-transformation would transform the data to be approximately normally distributed? We didn't! But, the log transformation has a very special property that makes it a very sensible choice. **The log transformation turns multiplicative effects in to additive effects.**

In many situations, the response variable is subject to effects that act multiplicatively. E.g., would proximity to a train station act additively or multiplicatively on the price of a house?[4] What about having a view or being beside the estuary?

In such cases, taking the log of the response variable results in these effects acting additively on the log scale...and by virtue of the CLT, the consequence of a lot of effects adding together is approximate normality on the log scale.

[4] In other words, does being close to the train station make a house worth, say, $150,000 more, or 10% more?

Logarithm refresher

Turning multiplication in to addition

You have used logarithms before. When you multiply 100 by 100,000 you add the zeroes to create 10,000,000. That is, two 0s + five 0s = seven 0s = 10,000,000. More formally:

$$100 \times 100,000 = 10^2 \times 10^5 = 10^{2+5} = 10^7 = 10,000,000$$

Using base 10 (the number of fingers/toes we humans have) we can write $log_{10}(100) = log_{10}(10^2) = 2$. So ,

$$\begin{aligned} log_{10}(100 \times 100,000) &= log_{10}(10^2 \times 10^5) \\ &= log_{10}(10^{2+5}) = 2 + 5 = 7 \\ &= log_{10}(100) + log_{10}(100,000) \ . \end{aligned}$$

So we see that the log of a product (i.e., multiplication) is the sum (i.e, addition) of the logs.

In Springfield, they *should* work in base 8. This means that "100" in Springfield would correspond to our 64.

Why use base e?

In the scientific world we use base e logarithms[5] ($\log_e$) which are referred to as natural logs[6].

In `R` the `log` function is the natural logarithm. If you need to calculate the logarithm base 10, then you can use the `log10` function, or type `log(x, base = 10)`, where `x` is the value for which you are trying to calculate the logarithm.

In many situations, it does not matter which base you use as the laws of logarithms hold regardless of the base you use. That is,

$$\log_b(x \times y) = \log_b(x) + \log_b(y)$$

for any base $b > 0$, and values $x, y > 0$.

[5] $e = \lim_{n\to\infty}(1 + \frac{1}{n})^n = 2.718282\ldots$

[6] Some programming languages and books use ln for the $\log_e$ function.

The inverse log function, e^x

Turning addition in to multipication...

Because we use natural logs ($\log_e$), it follows that the inverse transformation (i.e., back transformation) is the exponential function. This is the `exp` function in `R`.

Exponentiating a sum is equivalent to multiplying the separate exponentials. That is,

$$e^{x+y} = e^x \times e^y$$

For example:

```
> exp(2)
[1] 7.389056
> exp(3)
[1] 20.08554
> exp(2+3)
[1] 148.4132
> exp(2)*exp(3)
[1] 148.4132
```


Section 6.4
Example 1: Multiplicative simple linear regression model

Multiplicative model with numerical explanatory variable

Mazda price data

When we also have explanatory variables (x), using the log transformation (applied to the response variable y) changes the shape of the relationship between x and y from additive to multiplicative. This is because we use the exponential to back-transform our additive linear model fitted to $\log(y)$.

We demonstrate with a new example:

The year of manufacturer and asking price of 123 Mazda cars were collected from the Melbourne Age newspaper in 1991. The variables measured were:

`price`	price of vehicle in Australian $
`year`	year of manufacture (1990 = 90)

We will assume that these data are a random sample from the population.[7]

[7] What would the population be here?

Multiplicative model with numerical explanatory variable

Depreciation of Mazda cars

Here we wish to understand how the cars lose value as they age. We have the year the car was manufactured so we can ascertain its age since these data were collected in 1991 (or '91). That is, `age=91-year`.

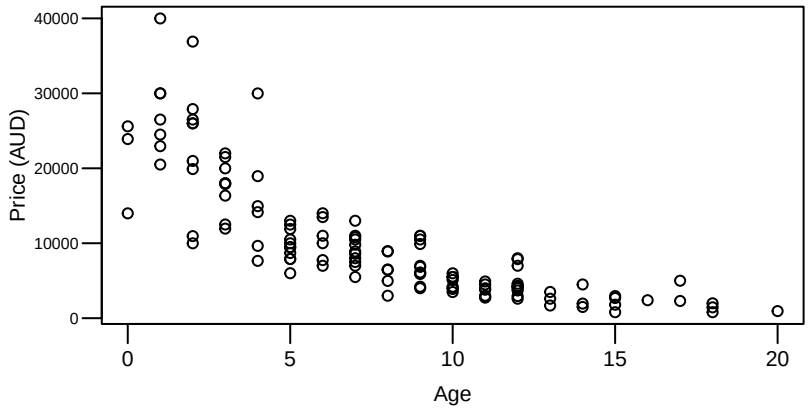
Intuition: What does common sense tell us about the shape of the relationship between car price and age? That is, what do we *expect* to see?

- An additive decrease in price with age?
- A multiplicative decrease in price with age?

Multiplicative model with numerical explanatory variable

Depreciation of Mazda cars...

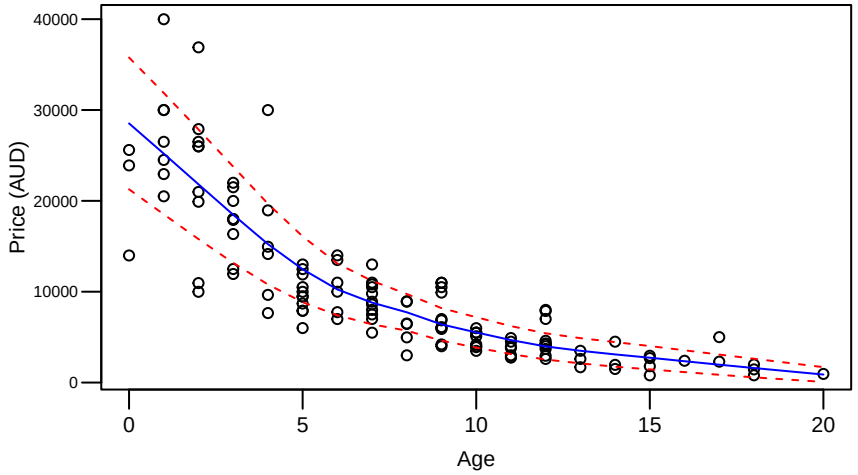
```
> Mazda.df = read.table("Data/mazda.txt",header=T)
> Mazda.df$age = 91 - Mazda.df$year ## Create the age variable
> plot(price~age, data = Mazda.df, xlab = "Age", ylab = "Price (AUD)")
```



Multiplicative model with numerical explanatory variable

Depreciation of Mazda cars...

Let us look at the two components we are interested in, trend and scatter.



Multiplicative model with numerical explanatory variable

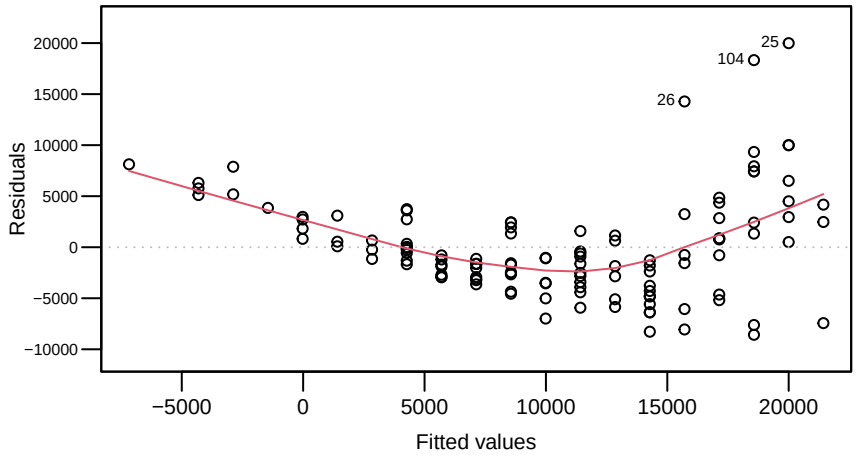
Depreciation of Mazda cars: A naïve price vs age models

The trend is decreasing (exponentially), along with decreasing scatter – these are classic symptoms of an underlying multiplicative model. Assuming **EOV** would be naïve in this case. Let us be naïve and see where it takes us. Let us fit a a linear model and see what the residual plot tells us.

Multiplicative model with numerical explanatory variable

Naïve price vs age models...

```
> PriceAge.fit=lm(price~age, data=Mazda.df)
> plot(PriceAge.fit,which=1)
```



Multiplicative model with numerical explanatory variable

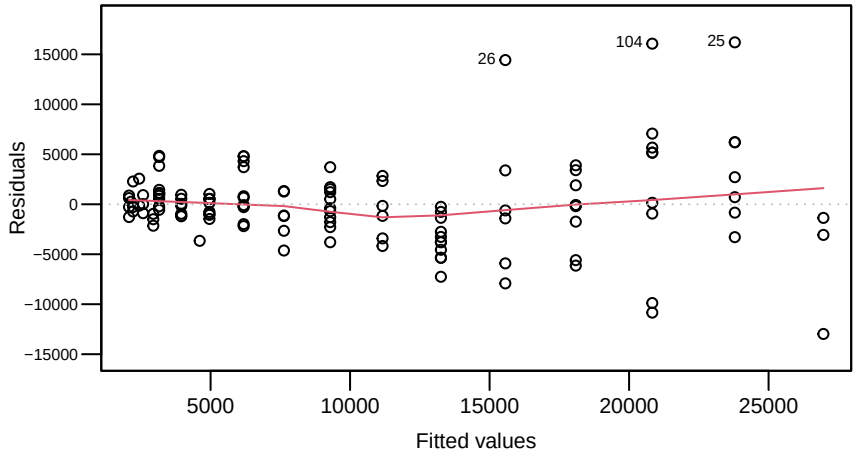
Naïve price vs age models...

The decreasing non-linear trend and non-constant scatter has become even more apparent. Note: the higher fitted values on the horizontal axis are associated with newer cars. Let us be slightly less naïve and deal with the trend by fitting a quadratic term.

Multiplicative model with numerical explanatory variable

Naïve price vs age models...

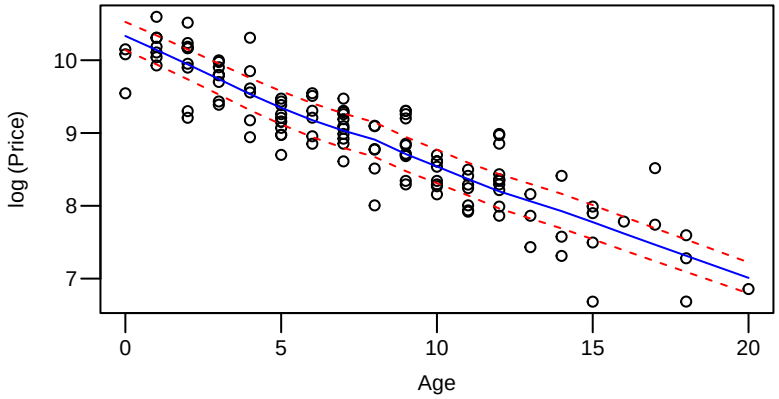
```
> PriceAge.fit2=lm(price~age+I(age^2), data=Mazda.df)
> plot(PriceAge.fit2,which=1)
```



Multiplicative model with numerical explanatory variable

Multiplicative price vs age model

We have eliminated trend from these residuals but the **EOV** assumption is still violated. Let us 'tear up' this approach and take logs of price.



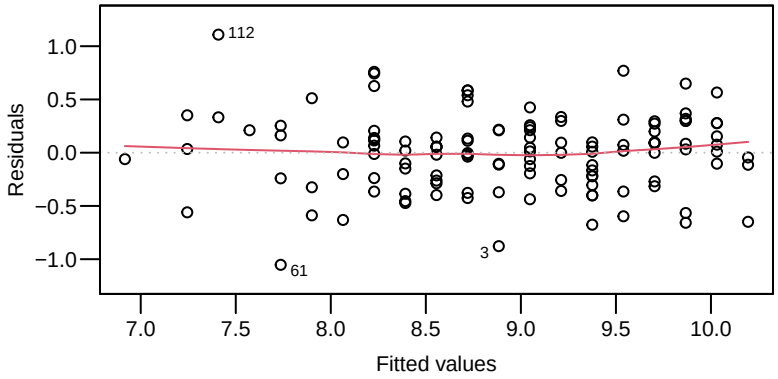
This is something we have seen before, is it not?

Multiplicative model with numerical explanatory variable

Multiplicative price vs age model...

Let us fit the more appropriate multiplicative model.

```
> LogPriceAge.fit=lm(log(price)~age, data=Mazda.df)
> plot(LogPriceAge.fit,which=1)
```



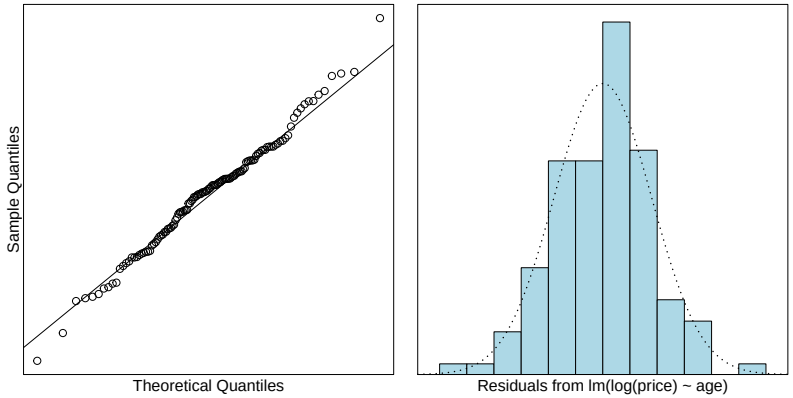
EOV assumption is now satisfied.

Multiplicative model with numerical explanatory variable

Multiplicative price vsage model...

Check for normality of the residuals.

```
> normcheck(LogPriceAge.fit)
```



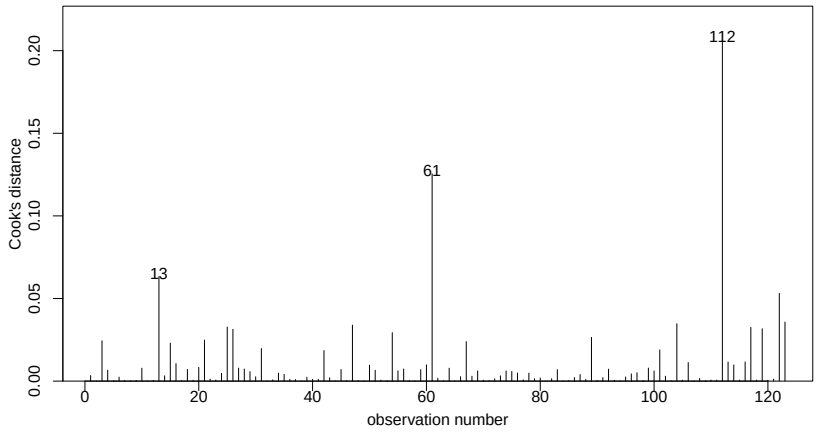
Normality assumption seems justified.

Multiplicative model with numerical explanatory variable

Multiplicative price vs age model...

Check for unduly influential data points.

```
> cooks20x(LogPriceAge.fit)
```



It looks fine.

Multiplicative model with numerical explanatory variable

Multiplicative price vs age model...

Our assumptions are sound, so we can trust the resulting output.

```
> summary(LogPriceAge.fit)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.195210   0.063602   160.3   <2e-16 ***
age         -0.163915   0.007034   -23.3   <2e-16 ***
---
Residual standard error: 0.3615 on 121 degrees of freedom
Multiple R-squared:  0.8178,Adjusted R-squared:  0.8163
F-statistic: 543.1 on 1 and 121 DF,  p-value: < 2.2e-16
```

```
> confint(LogPriceAge.fit)
                 2.5 %     97.5 %
(Intercept) 10.0692935 10.3211263
age         -0.1778406 -0.1499902
```

There is a massively significant effect of age.

But how do we turn these estimated values into something meaningful?

Multiplicative model with numerical explanatory variable

Inference from multiplicative price vs age model

The above fitted model gives median log-price for a car of a given age. So, exponentiating this model will give median price on the raw scale of $A. Using the equation for the fitted model, the estimated median prices is:

$$\begin{aligned}\widehat{\texttt{price}} &= e^{\hat{\beta}_0+\hat{\beta}_1\times\texttt{age}} \\ &= e^{\hat{\beta}_0} e^{\hat{\beta}_1\times\texttt{age}} \\ &= e^{10.195210} \times e^{-0.163915\times\texttt{age}} \\ &\approx \$26,775 \times (0.85)^{\texttt{age}}\end{aligned}$$

This means that for every year the car ages it is worth .85 (85% of) the previous year. That is, there is a 15% decline in value. In accounting this is known as depreciation of an asset.
E.g., we estimate that the median price of a new Mazda is about $26,775, a 1 year old about $\$26,775 \times 0.85^1 \approx \$22,727$, and a 2 year old about $\$26,775 \times 0.85^2 \approx \$19,291\ldots$

Multiplicative model with numerical explanatory variable

Inference from multiplicative price vs age model

Note: We exponentiate the model fitted to the logged Mazda prices to obtain a model for Mazda price.

Assumption checks, CIs, predictions (etc) use the model you fitted using the `lm` 'machinery'. That is, on the log scale.

The CIs and predictions are for use in the 'real world', and so must be back-transformed onto the scale of the raw data.

Multiplicative model with numerical explanatory variable

CIs from multiplicative price vs age model

We can obtain the confidence interval for median price of a new car by back-transforming the CI for the intercept value, just like we did with the null model discussed earlier.

```
> exp(confint(LogPriceAge.fit))
                  2.5 %       97.5 %
(Intercept) 2.360688e+04 3.036744e+04
age         8.370758e-01 8.607164e-01
```

The output: `(Intercept) 2.3607e+04 3.0367e+04` [8] says that the we are confident that the median price of a new Mazda car in 1991 is somewhere between AUD23,600 and AUD30,400 (to the nearest $100).

[8] `2.3607e+04`$= 2.3607 \times 10^4 = \$23607$

Multiplicative model with numerical explanatory variable

CIs from multiplicative price vs age model...

Now let us look at the coefficient for `age` i.e. $\hat{\beta}_1$.

If we exponentiate we get $e^{\hat{\beta}_1} = e^{-0.1639154} \approx 0.85$ or using `R`:

```
> exp(coef(LogPriceAge.fit)[2])
      age
0.8488138
```

From the previous page we see that the 95% CI for this coefficient is 8.37076e-01 to 8.60716e-01. That is, between 0.837 and 0.861, say.

Alternatively, we can calculate the percentage rate of depreciation:

```
> 100*(exp(confint(LogPriceAge.fit)[2,])-1)
    2.5 %    97.5 %
-16.29242 -13.92836
```

This says that our 95% CI for the annual depreciation in median price of Mazda cars is between $100\% \times (1 - 0.861) = 13.9\%$ and $100\% \times (1 - 0.837) = 16.3\%$

Multiplicative model with numerical explanatory variable

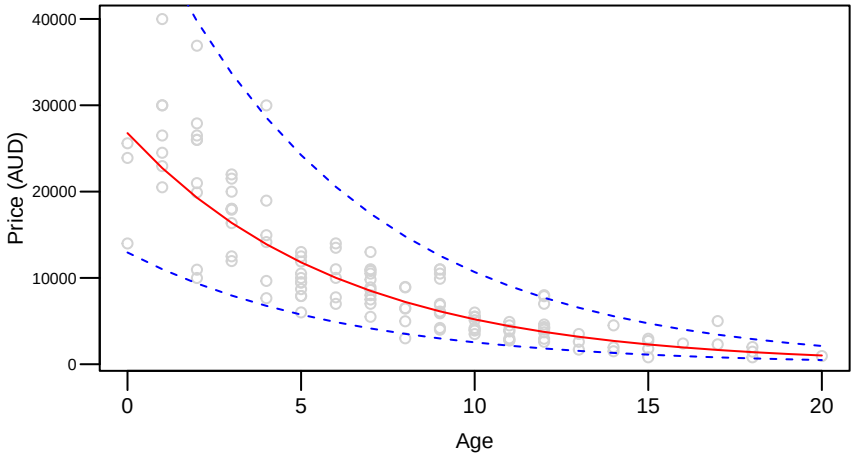
Predictions from multiplicative price vs age model

Let us now construct prediction intervals for cars that are 0 – 20 years old:

```
> plot(price~age, col="light grey",data=Mazda.df)
>
> ## Create a data.frame which has car ages from 0 to 20
> pred.df=data.frame(age=0:20)
>
> ## Exponentiate the fitted values - in the same order as above
> preds = exp(predict(LogPriceAge.fit,pred.df, interval="prediction"))
>
> lines(0:20, preds[,"fit"],col="red") ## Predicted value
> lines(0:20, preds[,"lwr"],col="blue", lty=2) ## Lower bound
> lines(0:20, preds[,"upr"],col="blue", lty=2) ## Upper bound
```


Multiplicative model with numerical explanatory variable

Predictions from multiplicative price vs age model...



Notice how the fitted line is at the half way value (median) and is robust against extreme values.

Multiplicative model with numerical explanatory variable

Executive summary: Mazda price vs age

We were interested in how the price of Mazda cars changed as they aged.

We estimate that the median value of a new car is about $26,800 (we are confident the median is somewhere between $23,600 and $30,400) and that for every year the car ages the median price depreciates at about 15.1% per year, and we are confident it is somewhere between 16.3 and 13.9%

Take home message:
Do not buy new cars!!!...unless you can truly afford the depreciation hit.

Mazda price vs. age

Multi-year depreciation

Most of us keep our cars for more than a year, so we might be interested to know how much Mazdas depreciate over a 5 year period, say.

The CI for 5-year depreciation is obtained by raising the CI (for 1-year depreciation) to the power of 5. This is an intuitive thing to do, as depreciation acts multiplicatively.

```
> exp(confint(LogPriceAge.fit)[2,])^5
    2.5 %    97.5 %
0.4109832 0.4723896
```

As a percentage change this is

```
> 100*(exp(confint(LogPriceAge.fit)[2,])^5-1)
    2.5 %    97.5 %
-58.90168 -52.76104
```

Ouch, the median price of Mazdas drops between 52.8% and 58.9% over 5 years.

Mazda price vs. age...

Multi-year depreciation...

Equivalently, the above CI can be obtained by back-transforming using five times the estimated effect.

```
> exp(5*confint(LogPriceAge.fit)[2,])
    2.5 %    97.5 %
0.4109832 0.4723896
```

As a percentage change this is

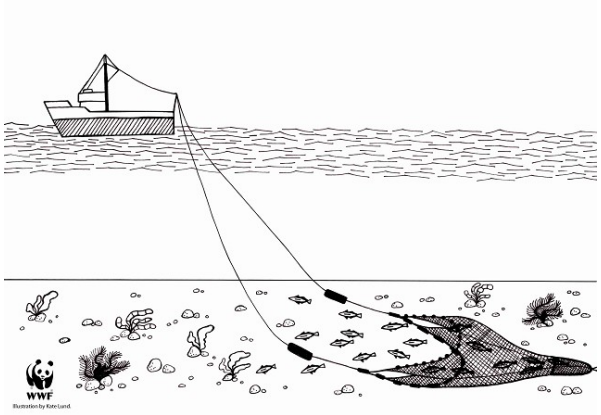
```
> 100*(exp(5*confint(LogPriceAge.fit)[2,])-1)
    2.5 %    97.5 %
-58.90168 -52.76104
```


Section 6.5
Example 2: Multiplicative model with categorical explanatory variable

Multiplicative model with categorical explanatory variable

Trawl bycatch

It was of interest to compare the amount of bycatch caught by two types of fishing trawl.

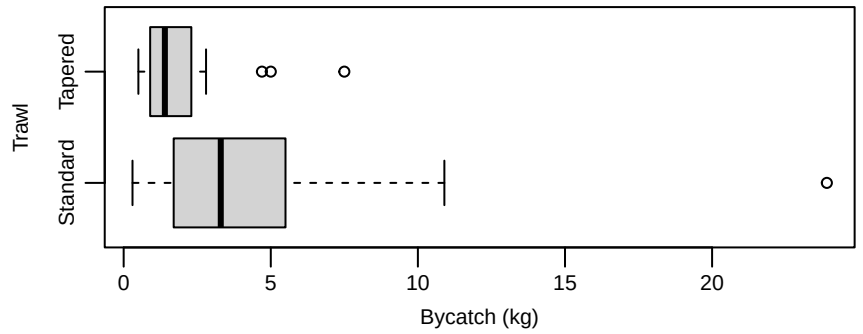


Intuition: Would you expect the effect of trawl type to be additive or multiplicative?

Multiplicative model with categorical explanatory variable

Trawl by-catch...

```
> Bycatch.df=read.table("Data/Bycatch.txt",header=T)
> boxplot(Bycatch~Trawl,data=Bycatch.df, horizontal=T,xlab="Bycatch (kg)")
```



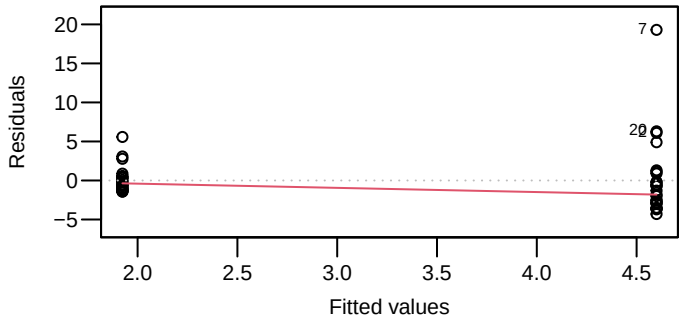

```
> summaryStats(Bycatch~Trawl,data=Bycatch.df)
         Sample Size  Mean Median  Std Dev Midspread
Standard          25 4.600    3.3 4.983138       3.8
Tapered           25 1.924    1.4 1.643999       1.4
```

Multiplicative model with categorical explanatory variable

Trawl by-catch...

This seems to confirm our intuition that these data should be modelled on the log scale. We will play stupid for now, and fit a linear model to the raw data....

```
> Trawl.lm=lm(Bycatch~Trawl,data=Bycatch.df)
> plot(Trawl.lm,which=1)
```

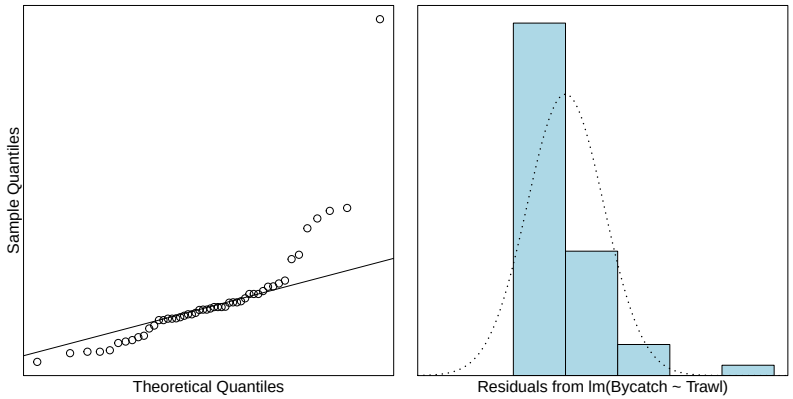


The **EOV** assumption is very doubtful. Normality?

Multiplicative model with categorical explanatory variable

Trawl by-catch...

```
> normcheck(Trawl.lm)
```

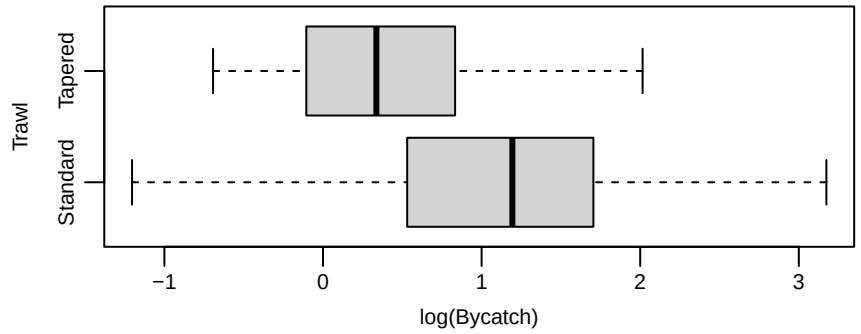


These plots of the residuals confirm that **EOV** and normality do not hold.

Multiplicative model with categorical explanatory variable

Using logged trawl by-catch

```
> boxplot(log(Bycatch)~Trawl,data=Bycatch.df,horizontal=T,xlab="log(Bycatch)")
```

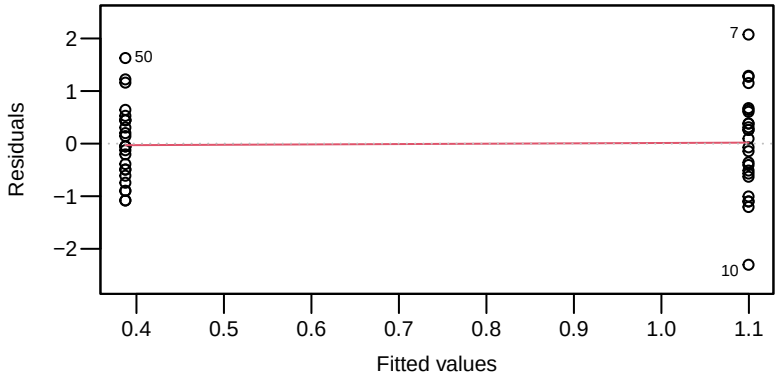


Looking much better.

Multiplicative model with categorical explanatory variable

Using logged trawl by-catch...

```
> Trawl.lmlog=lm(log(Bycatch)~Trawl,data=Bycatch.df)
> plot(Trawl.lmlog,which=1)
```

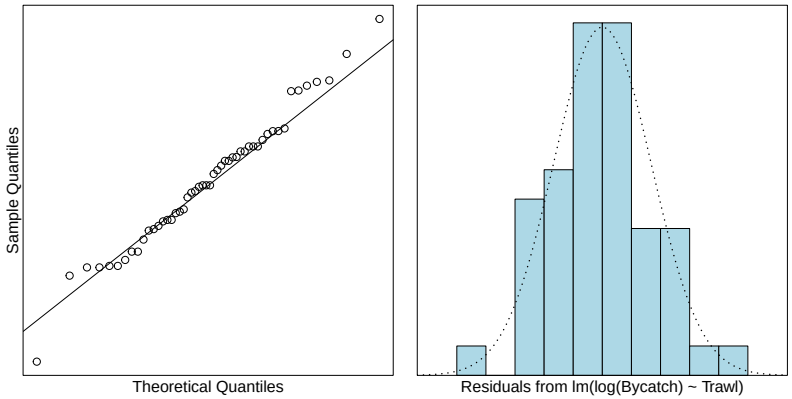


Sweet as.

Multiplicative model with categorical explanatory variable

Using logged trawl by-catch...

```
> normcheck(Trawl.lmlog)
```

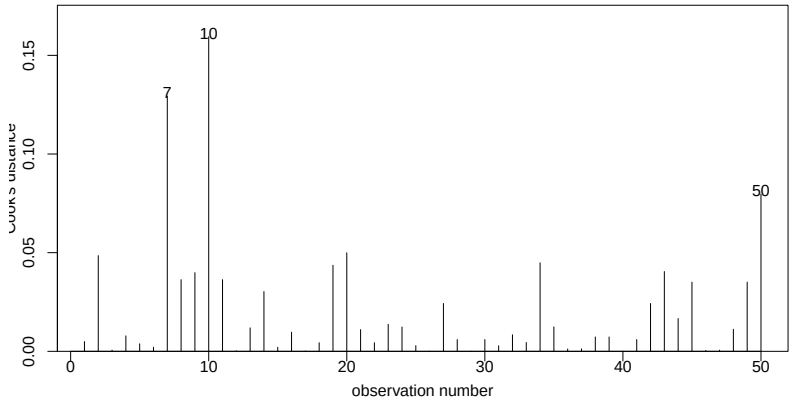


Looking good.

Multiplicative model with categorical explanatory variable

Using logged trawl by-catch...

```
> cooks20x(Trawl.lmlog)
```



No problems.

Multiplicative model with categorical explanatory variable

Results: Using logged trawl by-catch

Assumptions are satisfied. We can trust the fitted model.

```
> summary(Trawl.lmlog)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.0996     0.1700   6.469 4.79e-08 ***
TrawlTapered  -0.7122     0.2404  -2.963  0.00473 **
---
Residual standard error: 0.8498 on 48 degrees of freedom
Multiple R-squared:  0.1546,Adjusted R-squared:  0.137
F-statistic:  8.78 on 1 and 48 DF,  p-value: 0.004728
```

There is a statistically significance effect of trawl type (P-value ≈ 0.005). However, our model only explained 15% of the variability in the logged data and will not be very good for prediction.

Multiplicative model with categorical explanatory variable

Interpretation: Using logged trawl by-catch

We need to back-transform to the raw data scale to make sense of the above results:

```
> exp(confint(Trawl.lmlog))
                2.5 %    97.5 %
(Intercept)  2.1336329 4.2261691
TrawlTapered 0.3025531 0.7953873
```

The median by-catch using the tapered trawl was estimated to be between 30% and 80% that of the standard trawl.

Alternatively,

```
> 100*(exp(confint(Trawl.lmlog)[2,])-1)
    2.5 %    97.5 %
-69.74469 -20.46127
```

the median by-catch using the tapered trawl was estimated to be between 20% and 70% smaller than when using the standard trawl.

Section 6.6
Closing remarks and relevant R-code

Closing remarks

NOTE: Any of the types of linear model that you encounter *could* be applied to logged data if that is more appropriate, either due to rationale and/or due to right-skewness in the data.

In all cases you will follow the same procedure:

1. Fit the linear model to the logged data.
2. Back-transform (i.e., exponentiate) the relevant estimated coefficients, and confidence or prediction intervals.
3. **Remember** that the effect is multiplicative, and that back-transformed coefficients and CIs are for the median.

Most of the new R-code you need for this chapter

If any of the following hold:

- A multiplicative effect of the explanatory variables seems appropriate
- Right skewness of the variability (inspect the residuals to check)
- A funnel effect in the plot of residuals vs fitted values

then a log transformation of the response variable y may be appropriate. The usual steps (fitting a linear model and assumption checking, CIs, etc) are then applied to the logged response.

```
> Trawl.lmlog=lm(log(Bycatch)~Trawl,data=Bycatch.df)
> # then check if it's okay
> plot(Trawl.lmlog,which=1)
```

Confidence intervals can be back-transformed and interpreted as a multiplicative increases/decreases - in this case relative to baseline:

```
> exp(confint(Trawl.lmlog))
                 2.5 %    97.5 %
(Intercept)  2.1336329 4.2261691
TrawlTapered 0.3025531 0.7953873
```

and we interpret this with respect to change in the median value of y.

Case Study 6.1: Mazda price vs age

Tou Ohone Andate - staff number 1234567

Problem

The ages and prices of 123 Mazda cars were collected from the Melbourne Age newspaper in 1991. We want to learn about Mazda prices, and how they decrease with age.

The variables measured are:

- `price`: Price in Australian $.
- `year`: Year of manufacture (note that 1990 = 90).

Question of Interest

We want to see how Mazda car prices decrease with age.

Read in and Inspect the Data

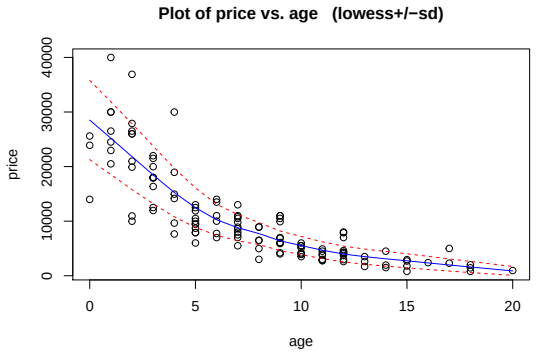
```
Mazda.df = read.table("mazda.txt", header = T)
head(Mazda.df)
```

```
##   year price
## 1   79  2950
## 2   82  5900
## 3   83  2999
## 4   88 11950
## 5   82  6100
## 6   90 26500
```

```
# We need to creates a new variable called age ourselves
Mazda.df$age = 91 - Mazda.df$year
head(Mazda.df)
```

```
##   year price age
## 1   79  2950  12
## 2   82  5900   9
## 3   83  2999   8
## 4   88 11950   3
## 5   82  6100   9
## 6   90 26500   1
```

```
# Plot these data
trendscatter(price ~ age, data = Mazda.df)
```

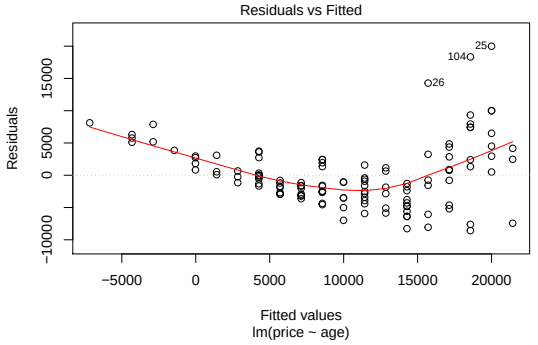
The scatter plot shows a decreasing non-linear relationship. As the age increases, the price decreases - but the rate of decrease is rapid at first, then declines, so also decrease. This suggests an exponentially decreasing relationship.

We also see that the scatter around the trend is not constant: it is higher when the price is higher and lower when the price is lower, so higher centre is associated with higher spread.

Let's fit a naive simple linear model using age for now.[1]

Model Building and Check Assumptions

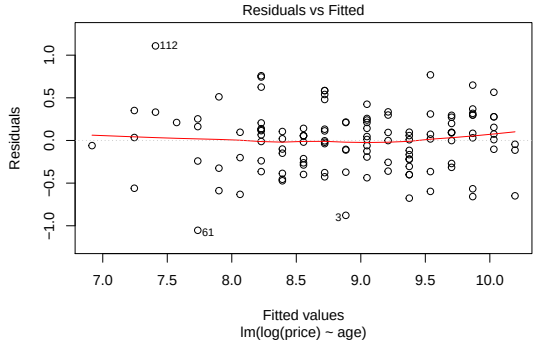
```
PriceAge.fit = lm(price ~ age, data = Mazda.df)
plot(PriceAge.fit, which = 1)
```



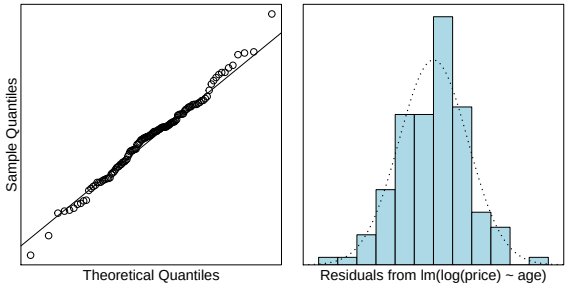
[1] In practice, one could omit this step since our assumptions are obviously not valid.

```
trendscatter(log(price) ~ age, data = Mazda.df)
```

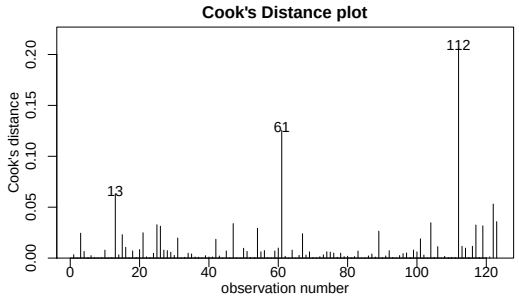

```
PriceAge.fit2 = lm(log(price) ~ age, data = Mazda.df)
plot(PriceAge.fit2, which = 1)
```




```
normcheck(PriceAge.fit2)
```


```
cooks20x(PriceAge.fit2)
```




```
summary(PriceAge.fit2)
```

```
##
## Call:
## lm(formula = log(price) ~ age, data = Mazda.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0531 -0.2398  0.0311  0.2110  1.1085
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.195210   0.063602   160.3   <2e-16 ***
```

```
## age         -0.163915   0.007034   -23.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3615 on 121 degrees of freedom
## Multiple R-squared:  0.8178, Adjusted R-squared:  0.8163
## F-statistic: 543.1 on 1 and 121 DF,  p-value: < 2.2e-16
```

```
# Backtransform
exp(confint(PriceAge.fit2))
```

```
##                    2.5 %       97.5 %
## (Intercept) 2.360688e+04 3.036744e+04
## age         8.370758e-01 8.607164e-01
```

```
# Backtransform to % difference
100 * (exp(confint(PriceAge.fit2)) - 1)
```

```
##                     2.5 %        97.5 %
## (Intercept) 2360588.14537 3036644.20481
## age             -16.29242     -13.92836
```

Method and Assumption Checks

The scatter plot of age vs price showed clear nonlinearity and an increase in variability with price.

Residuals from a simple linear model showed failed the equality of variance and no-trend assumptions, and so the prices were log transformed. A simple linear model fitted to logged price satisfied all assumptions.

Our final model is

$$log(Price_i) = \beta_0 + \beta_1 \times Age_i + \epsilon_i,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Our model explained 82% of the variability in the logged Mazda prices.

Executive Summary

We wanted to see how Mazda car prices decrease with age.

There was clear evidence the price of the cars was exponentially decreasing as the cars got older ($P\text{-}value \approx 0$).

We estimate that the median price for new Mazda cars (in 1991) was between A$23,600 to A$30,400 (to the nearest A$100).

We estimate that each additional year in age results in depreciation of between 13.9% to 16.3%.

Case Study 6.2: Students' expenditure on haircuts by gender

James Curran & Russell Millar

Problem

The question of interest is "Do females spend more money on their hair than males?" Also, "how much do students typically spend on haircuts?" To answer these questions, a lecturer carried out a survey on 200 students. Also, A variety of variables were measured including `hair` and `sex`.

The variables of interest are:

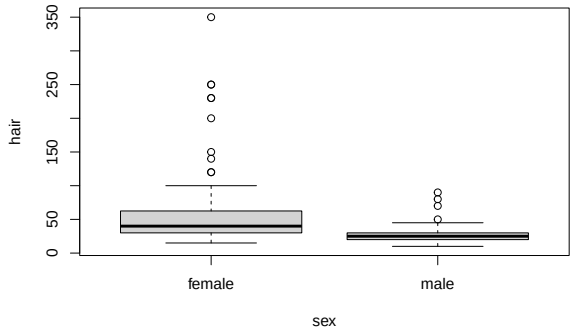
- `hair`: The student's estimated monthly expenditure on haircuts.
- `sex`: The student's gender, Male or Female.

Question of Interest

Do females spend more money on their hair than males? Also, how much do students typically spend on haircuts?

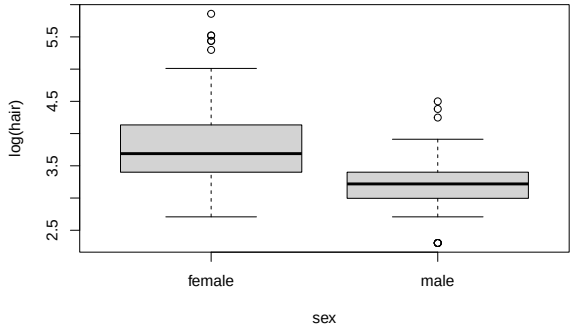
Read in and Inspect the Data

```
survey.df = read.table("survey.txt", header = TRUE, stringsAsFactors = T)
# To make things a little less cluttered we put the data in its own dataframe
hair.df = with(survey.df, data.frame(hair = hair, sex = sex))
plot(hair ~ sex, data = hair.df)
```

Females seem to spend more money on haircuts than males. We can see the data is quite right-skewed. There also appears to be a problem with equality of variance. Maybe taking logs will help.

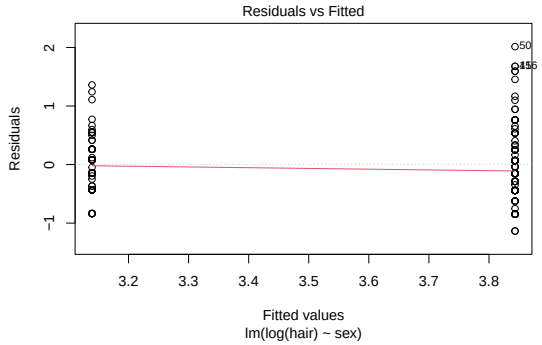
```
plot(log(hair) ~ sex, data = hair.df)
```



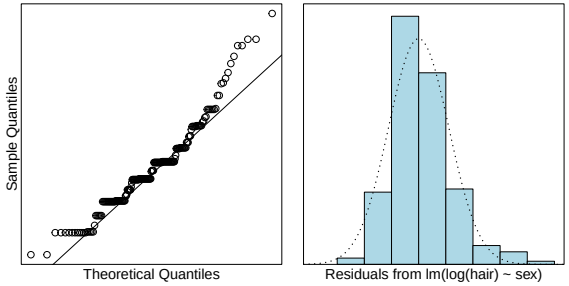
Definitely makes things nicer, we should stick with the log-scale. So perhaps we should perform our inference on the log-scale. Even on the log-scale females appear to be spending more money on haircuts than males.

Model Building and Check Assumptions

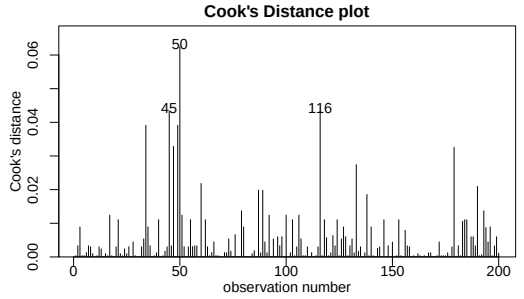
```
hair.fit = lm(log(hair) ~ sex, data = hair.df)
plot(hair.fit, which = 1)
```




```
normcheck(hair.fit)
```




```
cooks20x(hair.fit)
```


```
summary(hair.fit)
```

```
##
## Call:
## lm(formula = log(hair) ~ sex, data = hair.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.13528 -0.43125 -0.04246  0.26189  2.01460
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.84333    0.05380  71.443  < 2e-16 ***
## sexmale     -0.70403    0.07889  -8.924 3.05e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5565 on 198 degrees of freedom
## Multiple R-squared:  0.2868, Adjusted R-squared:  0.2832
## F-statistic: 79.64 on 1 and 198 DF,  p-value: 3.048e-16
```

```
# Column bind the backtransformed output together
cbind(exp(coef(hair.fit)), exp(confint(hair.fit)))
```

```
##                              2.5 %     97.5 %
## (Intercept) 46.6807537 41.9821788 51.9051852
## sexmale      0.4945885  0.4233304  0.5778413
```

Confidence Interval Output

```r
pred.df = data.frame(sex = c("female", "male"))
exp(predict(hair.fit, pred.df, interval = "confidence"))
```

```
##        fit      lwr      upr
## 1 46.68075 41.98218 51.90519
## 2 23.08776 20.60453 25.87028
```

Methods and Assumption Checks

The boxplot of `hair` vs `sex` revealed that the data were right-skewed. Hence, we logged `hair`. The boxplot of `log(hair)` vs `sex` show visible improvement. So, we have fitted a linear model with `log(hair)` being explained by `sex`.

We probably wouldn't say the data is normal from the Q-Q plot. What we have is evidence of a lot of rounding—people rounding to the nearest \$5, \$10, etc. However, the CLT should make things okay. The histogram of the residuals was unimodal and reasonably symmetric. The rest of our model assumptions have been satisfied.

Our final model is

$$log(Hair_i) = \beta_0 + \beta_1 \times SexMale_i + \epsilon_i,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$. Here $SexMale_i = 1$ if the student was male, otherwise it was zero.

Our model explained 29% of the variability in the logged students' hair expenditure.

Executive Summary

We have strong evidence that females typically spend more money on their hair than males.

We estimate that males spend approximately half as much as females do on their hair. We are confident this factor is between 42% and 58%.

We estimate the median amount females spend on their hair each month is \$47 and are confident it is between \$42 and \$52. For males we estimate a median spend of \$23 and are confident it is between \$21 and \$26.

Case Study 6.3: Students' expenditure on clothing

James Curran

Problem

The question of interest is "How much money do students spend on clothing on average?" This information was also recorded in the survey discussed in Case Study 6.2. A variety of variables were measured including `clothes`.

The variables of interest are:

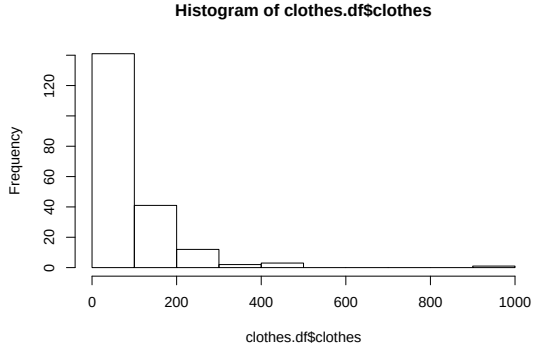
- `clothes`: The student's estimated estimated monthly expenditure on clothing.

Question of Interest

How much money do students spend on clothing on average?

Read in and Inspect the Data

```
survey.df = read.table("survey.txt", header = TRUE)
# To make things a little less cluttered we put the data in its own dataframe.
clothes.df = with(survey.df, data.frame(clothes = clothes))
hist(clothes.df$clothes)
```



We can see the data is quite right-skewed. With data this skewed the median looks to be a better measure of centre. Logging the data should help. It also looks like there is one really "silly"" value. Maybe we should omit it?

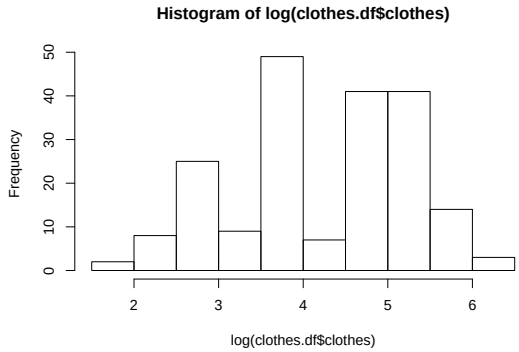
```
subset(clothes.df, clothes > 800)
```



```
##    clothes
## 15  999.99
```

This is clearly a "joke." Let's remove it from further analysis. It isn't likely to have much effect, but we don't believe it is a genuine answer, so we might as well remove it.

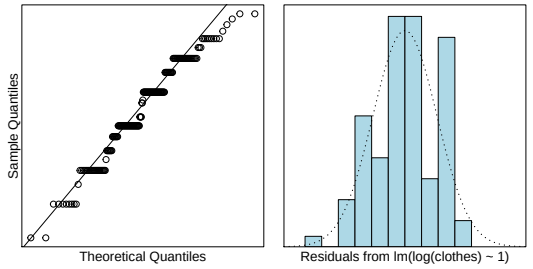
```
clothes.df = subset(clothes.df, clothes < 999)
hist(log(clothes.df$clothes))
```



Perhaps it makes the data more symmetric, but it is normal? There are multiple modes here. Let's see what happens if we ignore this.

Model Building and Check Assumptions

```
clothes.fit = lm(log(clothes) ~ 1, data = clothes.df)
normcheck(clothes.fit)
```



How about normality? Lots of repeated values—this shows up as the "step" pattern in the Q-Q plot, and multiple modes in the residual plot. But the CLT will save us right?

```
est = exp(coef(clothes.fit))
ci = exp(confint(clothes.fit))
median(clothes.df$clothes)
```

```
## [1] 85
```

```
est
```

```
## (Intercept)
##    70.38002
```

```
ci
```

```
##                2.5 % 97.5 %
## (Intercept) 61.53073 80.502
```

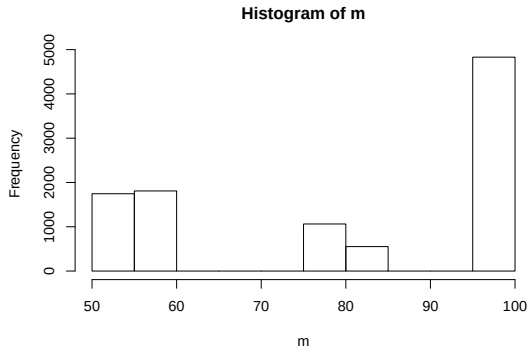
Oh dear! Our sample median is outside of our confidence interval for the median. Maybe we should forget the normal theory and try bootstrapping? **Note:** to perform this next step you need to install the `bootstrap` package. You can do this by uncommenting (removing `#`) the first line in the R chunk below. You only have to do this one time, and then it is installed forever more. So put back the `#` after knitting the document for the first time.

```
# install.packages("bootstrap")
library(bootstrap)
m = bootstrap(clothes.df$clothes, 10000, median)$thetastar
quantile(m, c(0.025, 0.975))
```

```
##  2.5% 97.5%
##    50   100
```

Well that's better, but is it? This is our distribution of bootstrapped medians.

```
hist(m)
```



Would you be happy giving a confidence interval on this? Probably not. This is why we said the inference is *approximate*. When we make inference on the back-transformed logged data, we're really making inference

about the geometric mean. This is defined as

$$geomean(x) = \left(\prod_{i=1}^{n} x_i\right)^{1/n}$$

In practice it is calculated by

$$geomean(x) = \exp\left(\frac{1}{n}\sum_{i=1}^{n} \log_e x_i\right)$$

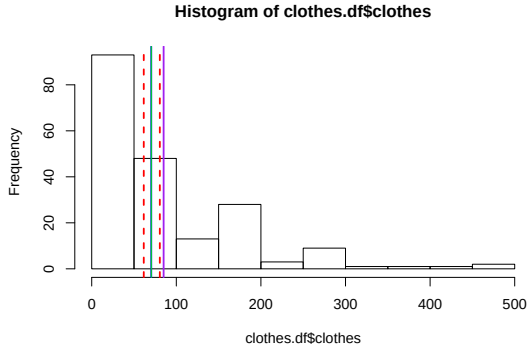
It is not hard to show that these two are algebraically equivalent, but the second formula is less susceptible to computational overflow, which happens when numbers get too big. This will always happen for a big enough sample size, assuming that most of the x_is are greater than 1.

So we need to briefly check out whether inference about the geometric mean makes more sense. R doesn't have this function built-in, so we need to define it.

```
# Define the geometric mean for later on
geomean = function(x) {exp(mean(log(x)))}
```

All we need to do is make sure that our geometric mean is a) within the confidence limits, and b) in the same position on the original scale.

```
hist(clothes.df$clothes)
abline(v = est, col = "blue", lwd = 2)
abline(v = ci, col = "red", lty = 2, lwd = 2)
# If this is correct, then the blue line should be replaced by a green line
# I've made it width 1, so you might be able to see it is right in the middle
# of the estimate, exactly where it should be
abline(v = geomean(clothes.df$clothes), col = "green")
# This is the median
abline(v = median(clothes.df$clothes), col = "purple", lwd = 2)
```



The geometric mean seems more appropriate here. So this example emphasizes that we really need the data to be *normal* on the log-scale for this approximate inference to work. Note that it works for the geometric mean regardless, but this is not examinable in this course.

Case Study 6.4: Clouding seeding

James Curran & Russell Millar

Problem

The data in this question come from an experiment to see whether cloud-seeding works. Cloud-seeding is the process of firing silver nitrate ($AgNO_3$) into the clouds. Water coalesces around the silver nitrate particles, hopefully getting large enough to precipitate (rain). The measurements in this study are in acre-feet. One acre-foot is the amount of water it takes to fill one acre uniformly to the depth of one foot. One acre-foot is about 1.2 million litres of water (1233481.85532 litres to be exact). This data set is included in the `s20x` library.

The variables of interest are:

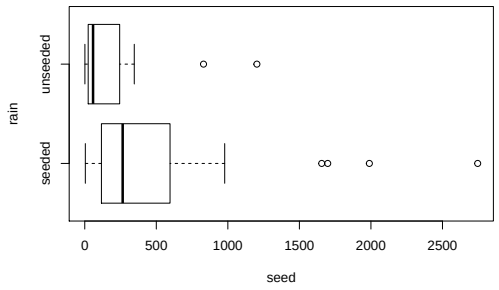
- `rain`: total rainfall.
- `seed`: is a factor with two levels, `seeded` and `unseeded`.

Question of Interest

We wish to see whether cloud seeding produces more rain. Also, what is the typical rainfall from seeded and unseeded clouds?

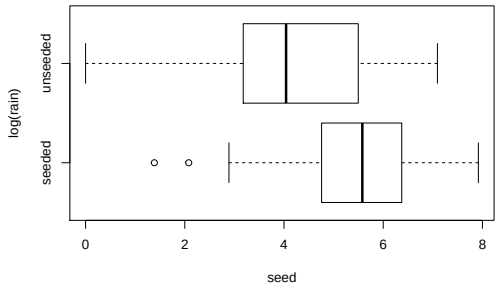
Read in and Inspect the Data

```
data(rain.df)
boxplot(rain ~ seed, data = rain.df, horizontal = TRUE)
```



Looks like our old right-skewed friend again. Maybe, just maybe, a log-transformation would work. Let's try it.

```
boxplot(log(rain) ~ seed, data = rain.df, horizontal = TRUE)
```

Yup. Looks much better, and we have preliminary evidence that the seeding seems to work. Should we be worried about equality of variance?

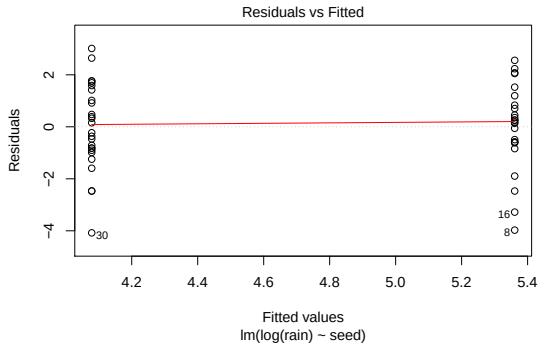
```
summaryStats(log(rain) ~ seed, data = rain.df)
```

```
##           Sample Size     Mean   Median  Std Dev Midspread
## seeded             24 5.360901 5.579017 1.665145  1.518865
## unseeded           26 4.078695 4.042127 1.661585  2.198249
```

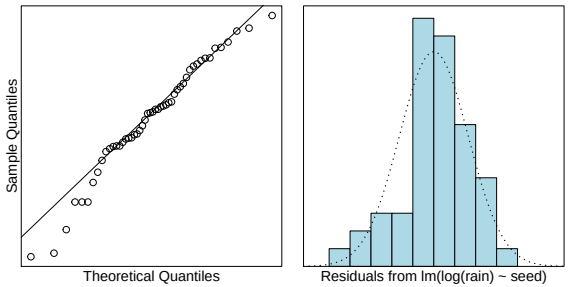
No. On the log-scale, the standard deviations are nearly identical, and the midspreads are well below a factor of two different. Let's go ahead and fit the model.

Model Building and Check Assumptions

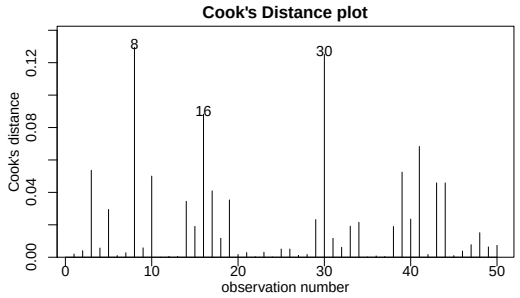
```
rain.fit  = lm(log(rain) ~ seed, data = rain.df)
plot(rain.fit, which = 1)
```



```
normcheck(rain.fit)
```




```
cooks20x(rain.fit)
```




```
summary(rain.fit)
```

```
##
## Call:
## lm(formula = log(rain) ~ seed, data = rain.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0787 -0.8206  0.1679  1.1496  3.0139
##
## Coefficients:
```



```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    5.3609     0.3395  15.790  < 2e-16 ***
## seedunseeded  -1.2822     0.4708  -2.723  0.00899 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.663 on 48 degrees of freedom
## Multiple R-squared:  0.1338, Adjusted R-squared:  0.1158
## F-statistic: 7.416 on 1 and 48 DF,  p-value: 0.008985
```

```
# Let's refit the model so that `unseeded` is the base level.
rain.df$seed = relevel(rain.df$seed, ref = "unseeded")
rain.fit.2 = lm(log(rain) ~ seed, data = rain.df)
summary(rain.fit.2)
```

```
##
## Call:
## lm(formula = log(rain) ~ seed, data = rain.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.0787 -0.8206  0.1679  1.1496  3.0139
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.0787     0.3262  12.504  < 2e-16 ***
## seedseeded    1.2822     0.4708   2.723  0.00899 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.663 on 48 degrees of freedom
## Multiple R-squared:  0.1338, Adjusted R-squared:  0.1158
## F-statistic: 7.416 on 1 and 48 DF,  p-value: 0.008985
```

```
exp(confint(rain.fit.2))
```

```
##                 2.5 %     97.5 %
## (Intercept) 30.656034 113.813484
## seedseeded   1.398702   9.289329
```

```
100 * (exp(confint(rain.fit.2)) - 1)
```

```
##                  2.5 %     97.5 %
## (Intercept) 2965.60341 11281.3484
## seedseeded    39.87024   828.9329
```

Confidence Interval Output

```
pred.df = data.frame(seed = c("unseeded", "seeded"))
exp(predict(rain.fit.2, pred.df, interval = "confidence"))
```

```
##         fit       lwr      upr
## 1  59.06835  30.65603 113.8135
## 2 212.91668 107.58209 421.3853
```

Methods and Assumption Checks

The boxplots of `log(rain)` showed that the groups were comparable. So, we fitted a linear model to explain `log(rain)` with the explanatory factor `seed`.

All model assumptions were satisified.

Our final model is

$$log(Rain_i) = \beta_0 + \beta_1 \times Seed.Seeded_i + \epsilon_i,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$. Here $Seed.Seeded_i = 1$ if the observation was done in a seeded environment, otherwise it is zero.

Our model describes 13% of the variability in the logged measurements of total rainfall.

Executive Summary

We wish to see whether cloud seeding produces more rain.

There is strong evidence cloud-seeding works ($P\text{-}value < 0.01$).

We estimate that the median rain from the seeded clouds is about 40% to 830% higher than that from unseeded clouds.

The unseeded clouds produce median rainfall of 30.7 to 113.8 acre-feet, whereas for the seeded clouds it is 107.6 to 421.4 acre-feet of rain.

Chapter 7: Power law linear models

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

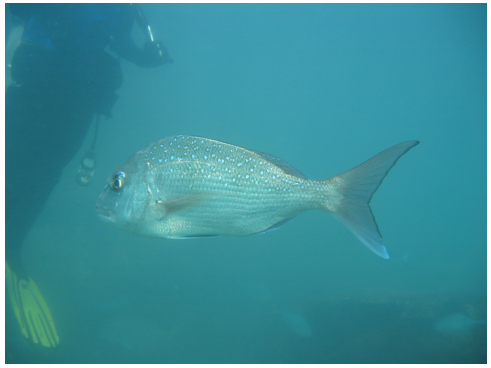
- Power law models
- How to interpret the effect of the explanatory variable
- Relevant R-code.

Section 7.1 Power law model example

Example – Weight of snapper as a function of length

Those of you who fish in the Hauraki Gulf will know that the minimum legal size for retaining a snapper is 30 cm. Here, we want to use snapper length to explain snapper weight, and in particular we want to estimate the weight of 30 cm snapper.

Note that this research question is highly relevant, since the relationship between length and weight is crucial for the stock assessments of snapper.



Weight of snapper as a function of length

What does our intuition tell us about the shape of the relationship between length and weight?

- Straight line?
- Quadratic?
- Exponential?
- Other?

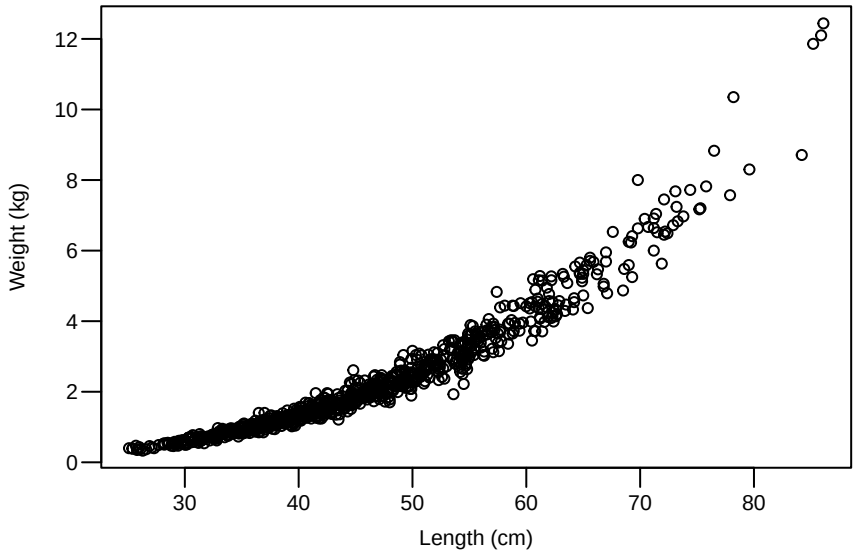
Weight of snapper as a function of length...

The data file `SnapWgt.txt` contains measurements on 844 snapper. The variables are:

`len`	fork length (cm)
`wgt`	weight (kg)

```
> Snap.df=read.table("SnapWgt.txt",header=TRUE)
> plot(wgt~len,data=Snap.df,xlab="Length (cm)",ylab="Weight (kg)")
```


Weight of snapper as a function of length...



Weight of snapper as a function of length...

Clearly there is a non-linear relationship between weight and length.

Geometry tells us that if an object changes in overall size while keeping the same shape (i.e., same ratio between height, depth and length), then its volume will increase with the 3rd power of length.

- For a cube with sides of length l, $volume = len^3$.
- For a sphere with radius r, $volume = \frac{4}{3}\pi r^3$.

That is, $volume \propto len^3$. In other words

$$volume = k_1 \times len^3$$

for some constant k_1.

Assuming weight of a solid object is proportional to its volume, we conclude

$$weight = \alpha \times len^3$$

for some constant α.

Weight of snapper as a function of length...

Those of you who have caught snapper will know that they do exhibit a small change in shape as they grow larger, so it would be better to use the model

$$weight = \alpha \times len^{\beta_1}$$

where β_1 is some constant that may be close to, but not necessarily equal to 3.

Taking logs gives

$$\log(weight) = \log(\alpha) + \beta_1 \times \log(len)$$

which we can rewrite as

$$\log(weight) = \beta_0 + \beta_1 \times \log(len) \ .$$

Weight of snapper as a function of length...

The formula on the previous slide specifies an assumed (i.e, expected) relationship between $\log(weight)$ and $\log(len)$ of a snapper.

Of course, snapper of a given length will have some variability in their weight, just as humans of a given height vary in their weight. So, what we are really saying is that the weight (kg) of an individual snapper of length len (cm) is

$$\log(weight) = \beta_0 + \beta_1 \times \log(len) + \varepsilon$$

where ε is some random variability (i.e., error around the expected value).

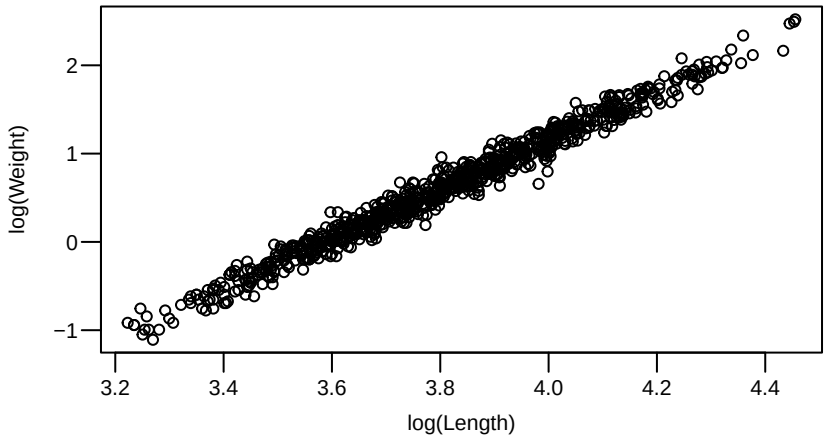
The above formula should be of very familiar form to you by now. Provided that we make the assumption that $\varepsilon \sim N(0, \sigma^2)$ then this is precisely the simple linear regression model with response variable $\log(weight)$ and explanatory variable $\log(len)$.

Weight of snapper as a function of length...

Fitting a simple linear model using log(weight) and log(length)

Let us look at the relationship between log(wgt) and log(len)

```
> plot(log(wgt)~log(len),data=Snap.df,xlab="log(Length)",ylab="log(Weight)")
```

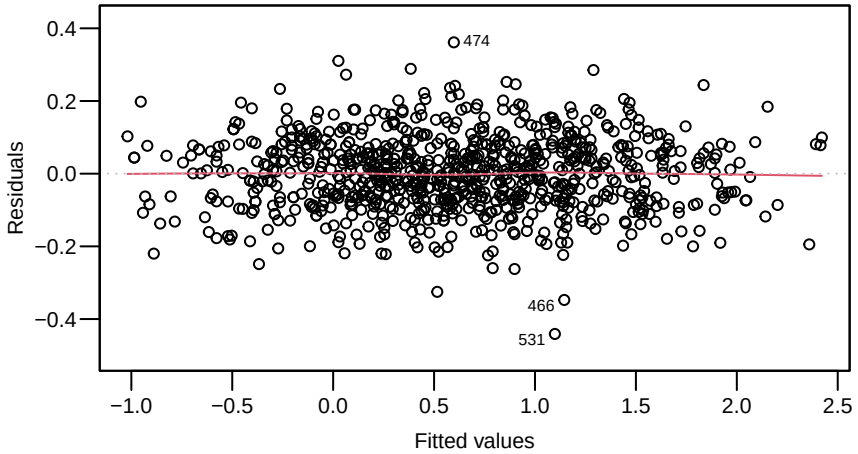


Looking good.

Weight of snapper as a function of length...

Fitting a simple linear model using log(weight) and log(length)...

```
> Snap.lm=lm(log(wgt)~log(len),data=Snap.df)
> plot(Snap.lm,which=1)
```

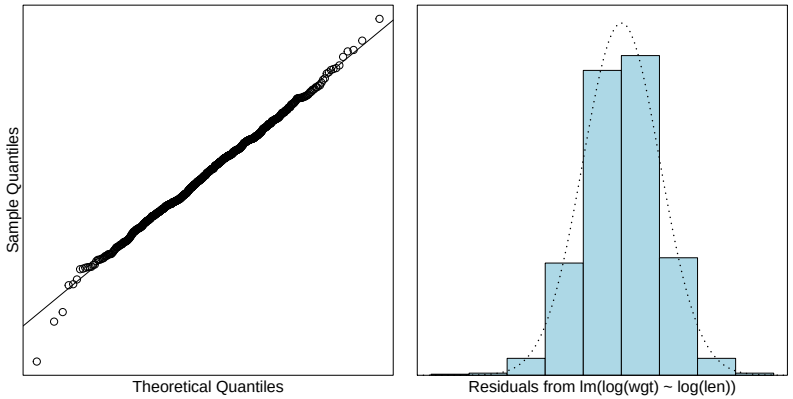


Weight of snapper as a function of length...

Fitting a simple linear model using log(weight) and log(length)...

Check the Normality assumption.

```
> normcheck(Snap.lm)
```

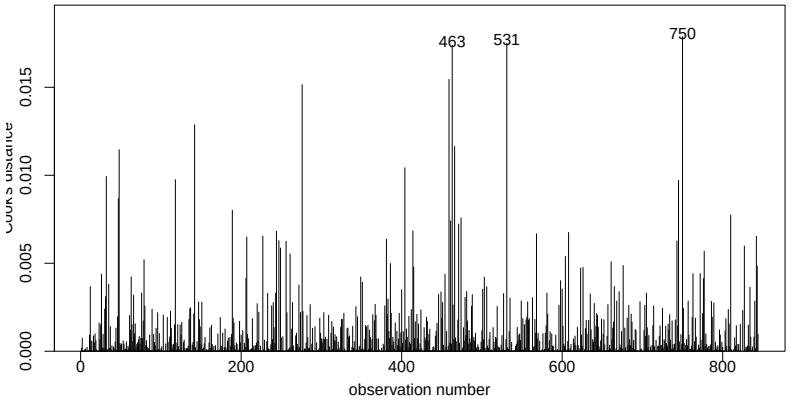


Weight of snapper as a function of length...

Fitting a simple linear model using log(weight) and log(length)...

Check for influential observations.

```
> cooks20x(Snap.lm)
```



Weight of snapper as a function of length...

Making inference

We can trust the fitted model.

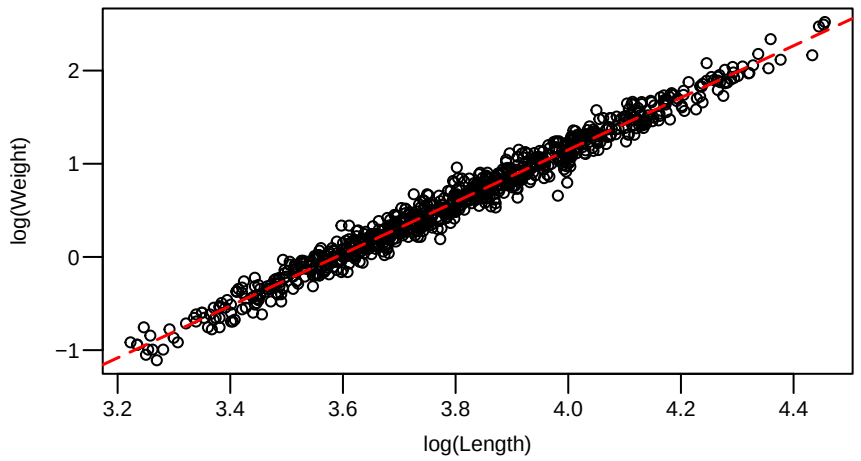
```
> summary(Snap.lm)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -10.01416    0.05602  -178.7   <2e-16 ***
log(len)      2.79104    0.01469   190.0   <2e-16 ***
---
Residual standard error: 0.1012 on 842 degrees of freedom
Multiple R-squared:  0.9772,Adjusted R-squared:  0.9772
F-statistic: 3.609e+04 on 1 and 842 DF,  p-value: < 2.2e-16
> confint(Snap.lm)
                2.5 %    97.5 %
(Intercept) -10.124126 -9.904199
log(len)      2.762204  2.819879
```


Weight of snapper as a function of length...

The fitted line on the log scale

```
> plot(log(wgt)~log(len),data=Snap.df,xlab="log(Length)",ylab="log(Weight)")
> abline(coef(Snap.lm),lty=5, col="red")
```



Weight of snapper as a function of length...

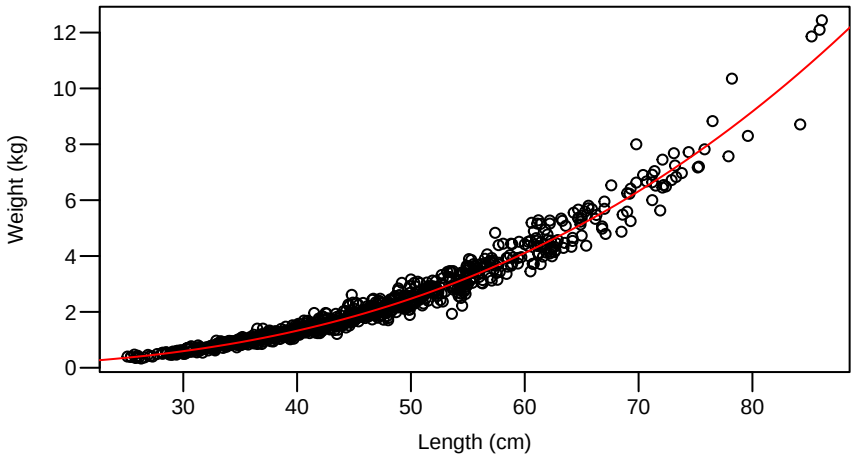
The fitted line on the log scale...

Let us redo the plot on the raw scale (rather than log scale):

```
> plot(wgt~len, data = Snap.df)
> pred.df = data.frame(len = 20:90)
> Snap.pred = exp(predict(Snap.lm, pred.df))
> lines(pred.df$len, Snap.pred, col="red")
```


Weight of snapper as a function of length...

The fitted line on the raw scale



Weight of snapper as a function of length...

Estimated weight of a 30cm snapper

Recall that we wanted to estimate the weight of 30 cm snapper. Since the linear model is fitted to log(wgt), we must back-transform, and are making inference about median weight.

```
> Pred.df=data.frame(len=30)
> exp(predict(Snap.lm,Pred.df,interval="confidence"))
       fit       lwr       upr
1 0.5937602 0.5857844 0.6018445
```

That is, we estimate 30 cm snapper to have median weight between 586 and 602 grams.

Note: If the research question had asked up to *predict* the weight of a 30 cm snapper then we would use

```
> exp(predict(Snap.lm,Pred.df,interval="prediction"))
       fit       lwr       upr
1 0.5937602 0.4865954 0.7245262
```

We predict a 30 cm snapper to weigh between about 487 and 725 grams.

Weight of snapper as a function of length...

Testing $H_0 : \beta_1 = 3$

A few slides earlier we deduced that the power coefficient β_1 should be close to, though not necessarily equal to 3.

Let us examine this formally by testing the null hypothesis $H_0 : \beta_1 = 3$.

Question 1 Is this hypothesis rejected at the 5% level? (Hint: the answer can be worked out from output already seen)

Question 2 What is the *P*-value for $H_0 : \beta_1 = 3$? (This takes a bit more work).

What is the *P*-value?

```
> beta1 = coef(Snap.lm)[2]
> seBeta1 = summary(Snap.lm)$coefficients[2,2]
> hyp = 3
> tstat = (beta1 - hyp)/seBeta1
> tstat
 log(len)
-14.22257
> pval = 2 * (1 - pt(abs(tstat), df = nrow(Snap.df) - 2))
> pval
log(len)
       0
```

The *P*-value is so small that it been rounded down to zero.
If we really want to see the *P*-value then we can get the `pt` function to do the probability calculation on the log scale, and then exponentiate:[1]

```
> logP = log(2)+pt(-abs(tstat), df = nrow(Snap.df) - 2, log.p = T)
> exp(logP)
    log(len)
2.677086e-41
```

We see that the *P*-value() is in the order of 10^{-41}, i.e., virtually zero.[2]

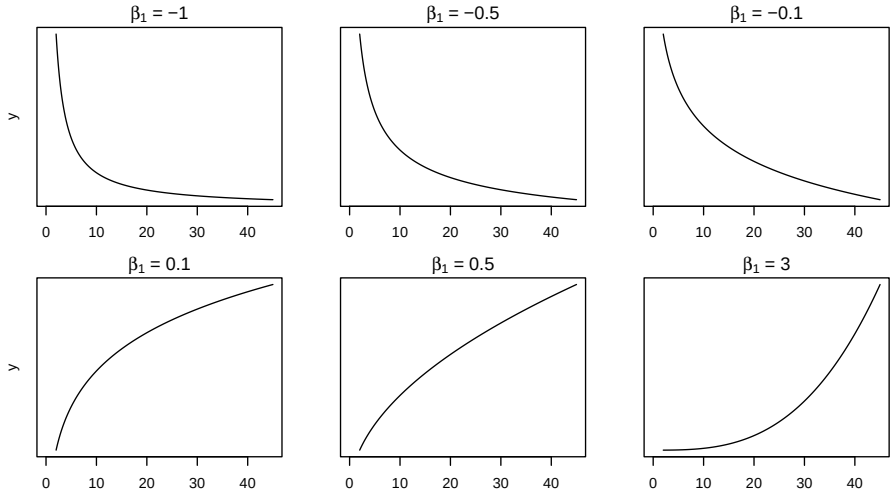
[1] Not examinable

[2] By way of comparison, the earth has about 10^{50} atoms.

Section 7.2
Power law curves and their interpretation

Power law relationships

Other examples of power curves



In general, a power law model could be used whenever one has good reason to believe that the relationship between x and y takes any of these forms.

Power relationships. . .

Interpretation of power curves

Recall, we are fitting the model $\log(y) = \beta_0 + \beta_1 \log(x)$, which is equivalent to $y = e^{\beta_0} x^{\beta_1} = \alpha x^{\beta_1}$.

For the snapper example the confidence interval for β_1 is

```
> confint(Snap.lm)[2,]
   2.5 %   97.5 %
2.762204 2.819879
```

One way to interpret this is to say that a 1% increase in the $x = len$ value results in a 2.76% to a 2.82% increase in the median value of $y = wgt$.

Explanation: Increasing x by 1% is the same as multiplying x by 1.01. So the relative change in y is:

$$\Delta y = \frac{\alpha(1.01x)^{\beta_1}}{\alpha x^{\beta_1}} = \frac{\alpha x^{\beta_1}(1.01)^{\beta_1}}{\alpha x^{\beta_1}} = 1.01^{\beta_1}$$

$1.01^{\beta_1} \approx 1 + \beta_1 \times .01$ for reasonable values of β_1 (Taylor's series). So, a 1% increase in x results in an approximate relative increase in y of $\Delta y \approx \beta_1 \times .01$ or an increase of β_1%

Power relationships...

Interpretation of power curves...

Alternatively, it might be more meaningful to quantify the change in the median of y arising from a 50% increase in x, or perhaps a doubling of x, say 50% increase in x: Then y changes from αx^{β_1} to

$$\alpha(1.5x)^{\beta_1} = \alpha x^{\beta_1} 1.5^{\beta_1}$$

i.e., the median of y gets multiplied by 1.5^{β_1}.

Doubling in x: Then y changes from αx^{β_1} to

$$\alpha(2x)^{\beta_1} = \alpha x^{\beta_1} 2^{\beta_1}$$

i.e., the median of y gets multiplied by 2^{β_1}.

Power relationships...

Interpretation of power curves...

```
> 1.5^confint(Snap.lm)[2,]
   2.5 %   97.5 %
3.064785 3.137300
```

That is, increasing length by 50% corresponds to an increase in median snapper weight between 206% and 214%[3]. [4]

```
> 2^confint(Snap.lm)[2,]
   2.5 %   97.5 %
6.784319 7.061030
```

That is, doubling length corresponds to an increase in median weight between 578% and 606%.

[3] These percentages are given by `100*(1.5^confint(Snap.lm)[2,]-1)`.

[4] One could also say that median snapper weight is between 2.06 and 2.14 times higher. Be careful **not** to say that it is between 3.06 and 3.14 times higher (since it actually multiplies by between 3.06 and 3.14). It is probably safest to talk about % change.

Section 7.3
Relevant R-code

Most of the R-code you need for this chapter

When your response variable is right skew and you have a good reason to believe the underlying relationship follows a power relationship then try taking logs of both y and x.

```
> Snap.lm=lm(log(wgt)~log(len),data=Snap.df)
```

We state the effect as the % change in the median of y for a given % change in x.

The confidence interval for β_1 is

```
> confint(Snap.lm)[2,]
```

and is the (approximate) % change in the median y for a 1% increase in x.

In general, for a z% increase in x, the multiplier for the median of y is

```
> (1+z/100)^confint(Snap.lm)[2,]
```

or alternatively, the percentage change in the median of y is

```
> 100*((1+z/100)^confint(Snap.lm)[2,]-1)
```

Case Study for Chapter 7: Power law model for cherry tree volumes

Tou Ohone Andate - staff number 1234567

Problem

We want to build a model to estimate the volume of a tree from the measurement of its height and diameter. This data set provides measurements of the diameter, height and volume of timber in 31 felled black cherry trees[1].

The data consist of 31 observations on the following 3 variables:

- `diameter`: Tree diameter in inches (measured at 4 ft 6 in above the ground).
- `height`: Height in feet.
- `volume`: Volume of timber in cubic feet.

Question of Interest

The objective is to be able to estimate the volume of the tree from the measurement of its height and diameter.

Model Justification

Using basic geometry, it can argued that volume will have a power law relationship with diameter and height. Specifically, it may be that the trunk of a tree can be reasonably approximated by a cylinder/or cone.

For a cylinder the volume is $V = \pi h r^2$ where r is radius and h is height.

For a cone it is $V = \frac{\pi}{3} h r^2$.

In general, this correponds to the power law $V \propto h d^2$, where d is diameter.

That is, $\log(V) = \beta_0 + \log(h) + 2 * \log(d)$.

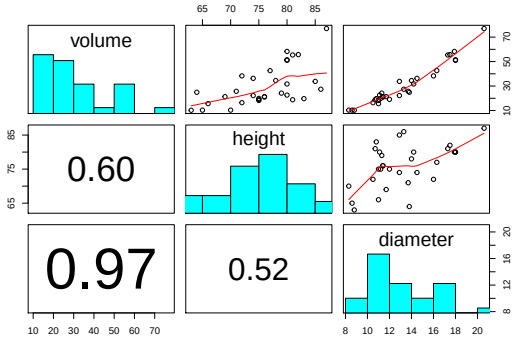
This can be written $\log(V) = \beta_0 + \beta_1 * \log(h) + \beta_2 \log(d)$ where $\beta_1 = 1$ and $\beta_2 = 2$.

Hence, to explore whether the above geometric argument is valid, we also need to test the hypotheses $H_0 : \beta_1 = 1$, $H_0 : \beta_2 = 2$.

Read in and Inspect the Data

```
Cherry.df = read.table("Cherry.txt", header = T)
pairs20x(Cherry.df[, c("volume", "height", "diameter")])
```

[1] From Ryan, T. A., Joiner, B. L. and Ryan, B. F. (1976) The Minitab Student Handbook. Duxbury Press.

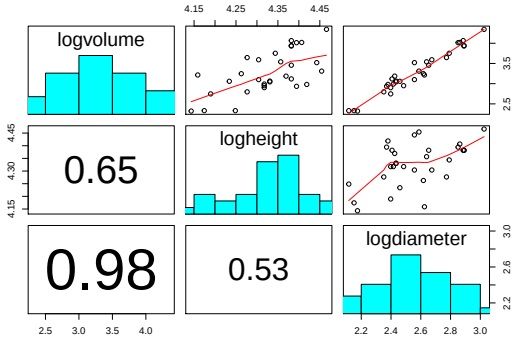


Scatter in volume increases with size. Relationship between volume and height looks close to linear, and with diameter looks to be increasing in slope with size.

```
# Let's create some new variables in Cherry.df
Cherry.df$logvolume = log(Cherry.df$volume)
Cherry.df$logheight = log(Cherry.df$height)
Cherry.df$logdiameter = log(Cherry.df$diameter)

# Check out the new names
dimnames(Cherry.df)[[2]]
```

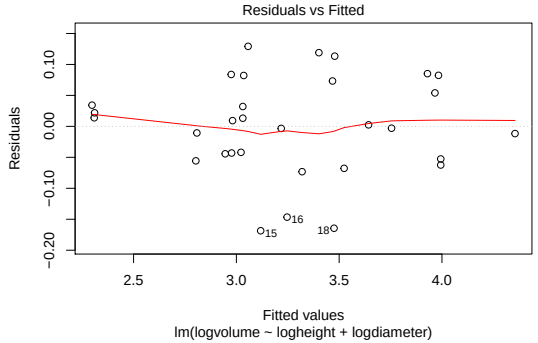
```
## [1] "diameter"    "height"      "volume"      "logvolume"   "logheight"
## [6] "logdiameter"
```

```
pairs20x(Cherry.df[, c("logvolume", "logheight", "logdiameter")])
```

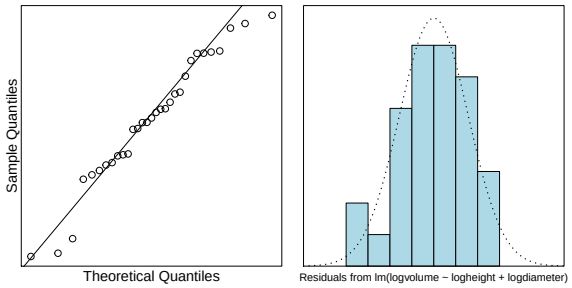


Scatter in logvolume looks constant, and relationship with logheight and logdiameter looks linear.

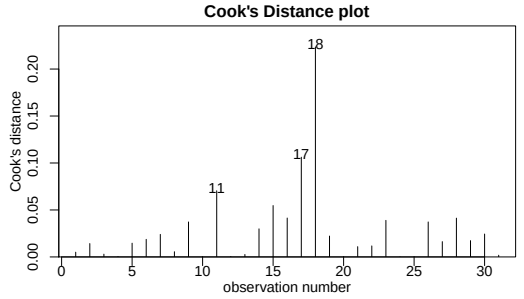
```
Cherry.fit = lm(logvolume ~ logheight + logdiameter, data = Cherry.df)
plot(Cherry.fit, which = 1)
```




```
normcheck(Cherry.fit)
```




```
cooks20x(Cherry.fit)
```


```
summary(Cherry.fit)
```

```
##
## Call:
## lm(formula = logvolume ~ logheight + logdiameter, data = Cherry.df)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.168561 -0.048488  0.002431  0.063637  0.129223
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.63162    0.79979  -8.292 5.06e-09 ***
## logheight    1.11712    0.20444   5.464 7.81e-06 ***
## logdiameter  1.98265    0.07501  26.432  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08139 on 28 degrees of freedom
## Multiple R-squared:  0.9777, Adjusted R-squared:  0.9761
## F-statistic: 613.2 on 2 and 28 DF,  p-value: < 2.2e-16
```

```
confint(Cherry.fit)
```

```
##                 2.5 %    97.5 %
## (Intercept) -8.269912 -4.993322
## logheight    0.698353  1.535894
## logdiameter  1.828998  2.136302
```

Test the Two Requested Hypotheses

```
# Obtain the estimates of interest
(estimates = summary(Cherry.fit)$coeff[2:3, 1])
```

```
##   logheight logdiameter
```

```
##    1.117123    1.982650
```

```
# and their associated std errors
(std.errors = summary(Cherry.fit)$coeff[2:3, 2])
```

```
##   logheight logdiameter
##  0.20443706  0.07501061
```

```
# t-values for the two hypotheses beta1=1 and beta2=2
(tstats = (estimates - c(1, 2))/std.errors)
```

```
##   logheight logdiameter
##   0.5729066  -0.2313018
```

```
# p-values
(pvals = 2 * (1 - pt(abs(tstats), 28)))
```

```
##   logheight logdiameter
##   0.5712805   0.8187623
```

Method and Assumption Checks

After logging the variables, the scatter plots showed the desired linear relationship between volume and the height and diameter variables, and constant scatter. So, we fitted a power law model explaining log volume with both log height and log diameter.

The underlying model assumptions appear valid with no unduly influential points.

Our final model is

$$log(volume_i) = \beta_0 + \beta_1 \times log(height_i) + \beta_2 * log(diameter_i) + \epsilon_i,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Our power law model explained 97.8% of the total variability in (log) volume.

Executive Summary

The objective is to be able to estimate the volume of the tree from the measurement of its height and diameter. The postulated model was that cherry tree volume would follow a power law relationship with height and diameter. In particular, that volume would increase linearly with height, and with the square of diameter.

Cherry tree volume was seen to follow a power law model with respect to both height and weight, and there was no evidence that these powers differ from the hypothesized values of 1 (*P-value = 0.57*) and 2 (*P-value = 0.82*), respectively. It would appear that our geometrical arguments were valid.

We estimate that a 1% increase in height corresponds to an increase in median volume of between 0.70% and 1.54%, and a 1% increase in diameter corresponds to an increase in median volume of between 1.83% and 2.14%.

Addendum (not examinable)

If you believe (and we have no reason not to) that the underlying relationship is indeed that $V \propto hr^2$ then the only term to be estimated is β_0. This is how you'd estimate it:

```
# Intercept only model.
fit.beta0 = lm(I(logvolume-logheight-2*logdiameter)~1, data = Cherry.df)
# beta0 changes from -6.63 to -6.17
summary(fit.beta0)
```



```
##
## Call:
## lm(formula = I(logvolume - logheight - 2 * logdiameter) ~ 1,
##     data = Cherry.df)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.168446 -0.047355 -0.003518  0.066308  0.136467
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.16917    0.01421  -434.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0791 on 30 degrees of freedom
```

Chapter 8:
Linear models with both numeric and factor explanatory variables

Part 1: The interaction model

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

- Models which contain categorical and numeric explanatory variables[1]
- Useful plots to display the data
- The meaning of **interaction**
- Fitting a model with an interaction term
- Interpretting the fitted model
- Relevant R-code.

[1] Models of this type are commonly known as Analysis of Covariance models, which is abbreviated to ANCOVA.

Section 8.1
Example: Using both test score and attendance to explain exam score

Part A: Exploratory analysis

Example – Exam vs. test **and** attendance

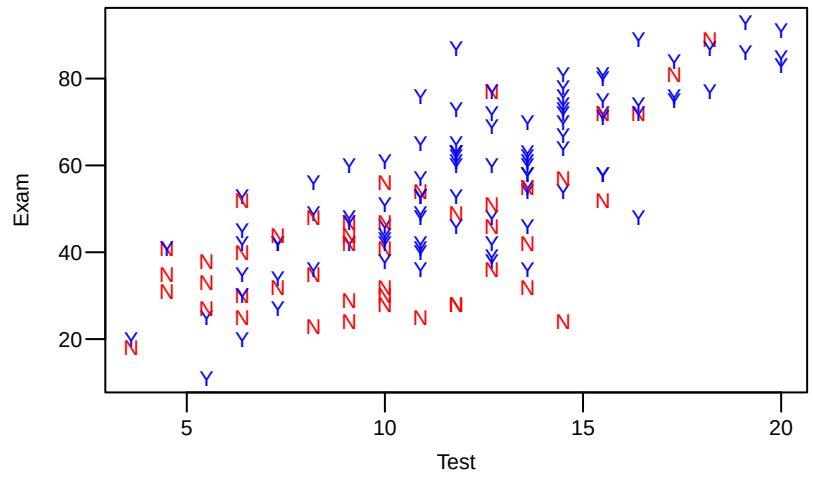
We have have learnt how to deal with the effect of test mark on exam score, and of attendance on exam score, individually.

So what is stopping us from using both? Absolutely nothing.

Let's begin by visualizing how test score relates to the exam score for the attenders and the non-attenders.

```
> ## Invoke the s20x library
> #library(s20x)
> ## Importing data into R
> Stats20x.df = read.table("Data/STATS20x.txt", header=T)
> Stats20x.df$Attend=as.factor(Stats20x.df$Attend)
> ## Plot blue "Y" for "Yes" (regular attenders), and red "N" for "No"
> plot(Exam ~ Test,data = Stats20x.df,pch=substr(Attend,1,1),
+          col=ifelse(Attend=="Yes","blue","red"))
```

Exam vs. test **and** attendance...

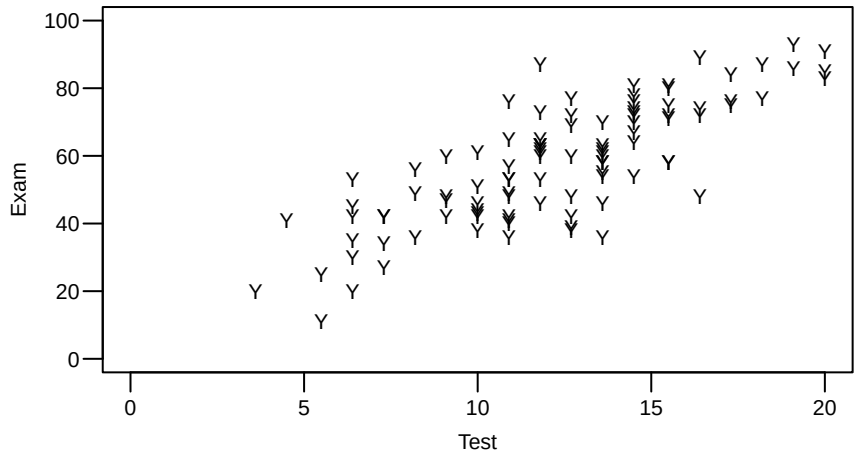


The above plot is a bit cluttered, so we could also draw a plot for each attendance type. For the sake of comparison it is important to ensure that the horizontal and vertical limits of the two scatter plots are the same.

Exam vs. test **and** attendance...

Here is the plot for the regular attenders.

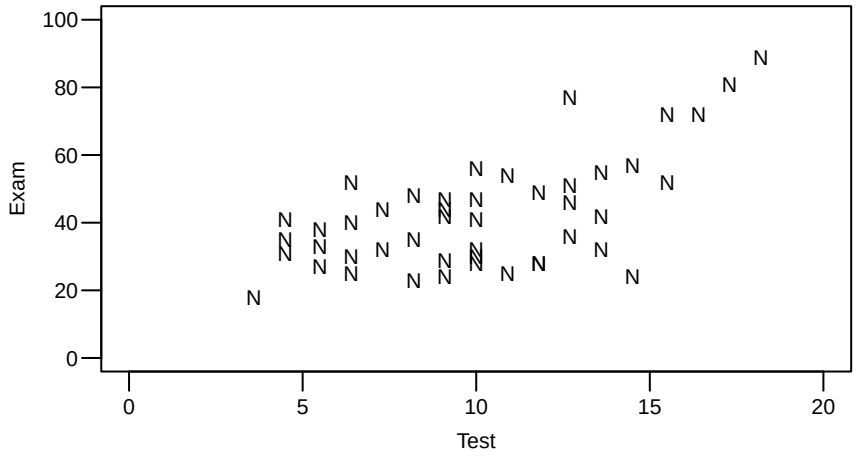
```
> Attendees.df = subset(Stats20x.df, Attend == "Yes")
> plot(Exam ~ Test, data = Attendees.df, xlim = c(0, 20), ylim = c(0, 100),
+      pch = "Y", cex = 0.7)
```



Exam vs. test **and** attendance...

Here is the plot for the non-attenders.

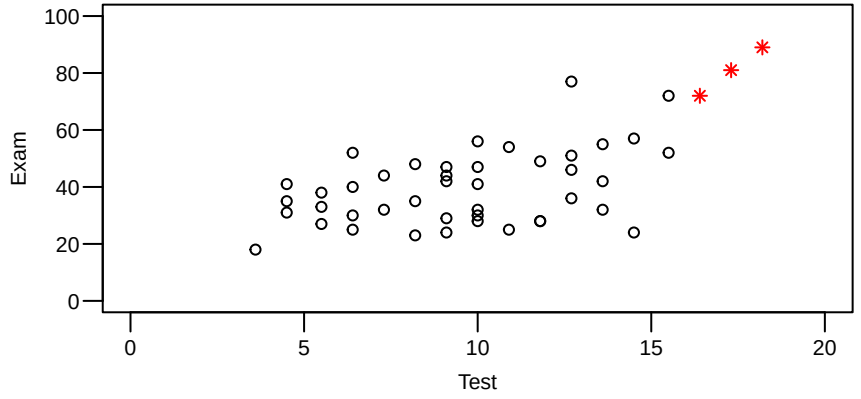
```
> Absentees.df = subset(Stats20x.df, Attend == "No")
> plot(Exam ~ Test, data = Absentees.df, xlim = c(0, 20), ylim = c(0, 100),
+      pch = "N", cex = 0.7)
```



Exam vs. test **and** attendance...

Also, there seems to be some non-attenders who do well in the test and exam so we could (and will) see whether we should include these people. They are identified in red (stars) with the R code below.

```
> plot(Exam ~ Test, data = Absentees.df, xlim = c(0, 20), ylim = c(0, 100),
+      cex = 0.7, col = ifelse(Absentees.df$Test <=16 , "black", "red"),
+      pch = ifelse(Absentees.df$Test <=16 , 1, 8))
```



Exam vs. test **and** attendance...

Hmmm—it seems that the non-attenders may get less 'return on investment' on test efforts compared to the regular attenders.

What we mean is that is that the slope looks less steep for non-attenders than regular attenders.

Can we explore this idea with a linear model? **Yes!**

Section 8.2
Example: Using both test score and attendance to explain exam score

Part B: Fitting the linear model

Exam vs. test **and** attendance...

So, it looks like we need to fit two different lines depending on whether the student is a regular attender or not.

One approach would be to fit separate linear models to the data in the `Attendees.df` and `Absentees.df` dataframes. However, this approach limits the questions that we can answer.

A more powerful approach is to use a single `lm` model to fit the two lines.

We can do this by using indicator variables (recall Chap 5).

Let us recode attendance using an indicator variable to indicate whether the student was an attender or not.

Exam vs. test **and** attendance...

We will call our indicator variable `D` for greater convenience of notation.[2]

```
> ## Boolean statement if Attend ="Yes" (TRUE) D=1, othwerwise 0 (FALSE);
> Stats20x.df$D = as.numeric(Stats20x.df$Attend=="Yes")
> table(Stats20x.df$Attend, Stats20x.df$D) ## Check it is okay

       0   1
  No   46   0
  Yes   0 100
```

[2] Last time (see Chapter 5) we called this indicator variable `Attend2` – but `D` is easier to use in an equation.

Exam vs. test **and** attendance...

Formulating the model

Our straight line model for the non-attenders (i.e., `D = 0`) students will be:

$$Exam = \beta_0 + \beta_1 \times \texttt{Test} + \varepsilon \text{ where } \varepsilon \overset{iid}{\sim} N(0, \sigma^2).$$

We would expect some benefit from doing better in the test, so we would suspect that $\beta_1 > 0$. The null hypothesis is, as usual,

$$H_0 : \beta_1 = 0.$$

Exam vs. test **and** attendance...

Formulating the model...

From the scatter plots we suspect there may be a different slope value for the regular attenders (in fact steeper, i.e., greater positive value.)

So what we are saying is that the slope for attenders is the slope for non-attenders plus a positive number so that it increases the slope.

We will call this additional (positive) number β_3.[3]

So we can say that the slope for any student is:

$$\beta_1 + D \times \beta_3$$

where `D = 0` when the student is a non-attender and `D = 1` when they attend regularly.

[3] The choice of symbol β_3 will be come obvious soon.

Exam vs. test **and** attendance...

Formulating the model...

For the non-attenders (`D = 0`) the slope is:

$$\begin{aligned} \beta_1 + D \times \beta_3 &= \beta_1 + 0 \times \beta_3 \\ &= \beta_1. \end{aligned}$$

For the regular attenders (`D = 1`) the slope is:

$$\begin{aligned} \beta_1 + 1 \times \beta_3 &= \beta_1 + 1 \times \beta_3 \\ &= \beta_1 + \beta_3. \end{aligned}$$

We suspect that $\beta_3 > 0$. The null hypothesis, as usual, is $H_0 : \beta_3 = 0$. The scatter plots suggest that the slope associated with `Test` changes depending on whether the student attends or not. This idea is known as interaction. That is, the effect of `Test` interacts with the students' attendance behaviour.

In our example, students who attend regularly, we suspect, get 'more' from the `Test` than non-attenders.

Exam vs. test **and** attendance...

Formulating the model...

The intercept for both groups can be formulated in a similar way. That is:

$$\beta_0 + D \times \beta_2.$$

So, for the non-attenders (`D = 0`) the intercept is β_0.

For the regular attenders (`D = 1`): the intercept is $\beta_0 + \beta_2$.

We do not have much interest in the intercept terms if the equal slope hypothesis ($H_0 : \beta_3 = 0$) is rejected[4], since they give the expected exam scores for students who get zero test score. So, we would then not be interested in testing $H_0 : \beta_2 = 0$.

[4] If the equal slope hypothesis is **not** rejected then we **do** have interest in β_2 since it is then the difference between attenders and non-attenders for any given test score

Exam vs. test **and** attendance...

Formulating the model...

So, our model is:

$$\begin{aligned}\texttt{Exam} &= \beta_0 + \beta_2 \times D + (\beta_1 + \beta_3 \times D)\texttt{Test} + \varepsilon \\ &= (\beta_0 + \beta_2 \times D) + (\beta_1 + \beta_3 \times D)\texttt{Test} + \varepsilon \\ &= \beta_0 + \beta_1 \times \texttt{Test} + \beta_2 \times D + \beta_3 \times D \times \texttt{Test} + \varepsilon\end{aligned}$$

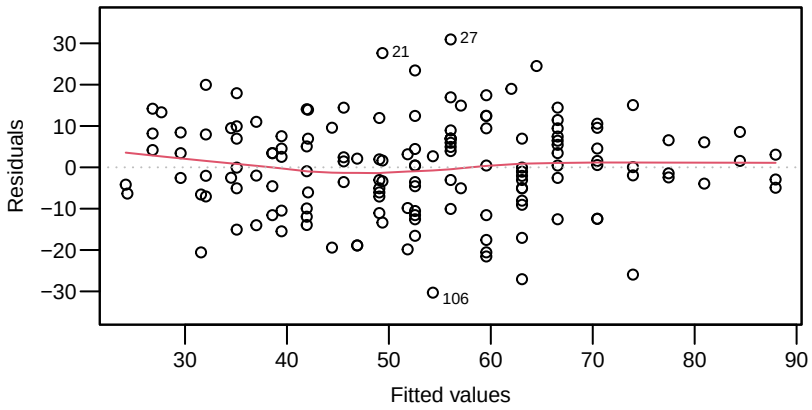
In order to fit this model we create an additional explanatory variable $D \times \texttt{Test}$.

```
> Stats20x.df$TestD  = with(Stats20x.df, {TestD = D * Test})
> TestAttend.fit = lm(Exam ~ Test + D + TestD, data = Stats20x.df)
```


Exam vs. test **and** attendance...

Assumption checks

```
> plot(TestAttend.fit,which=1)
```

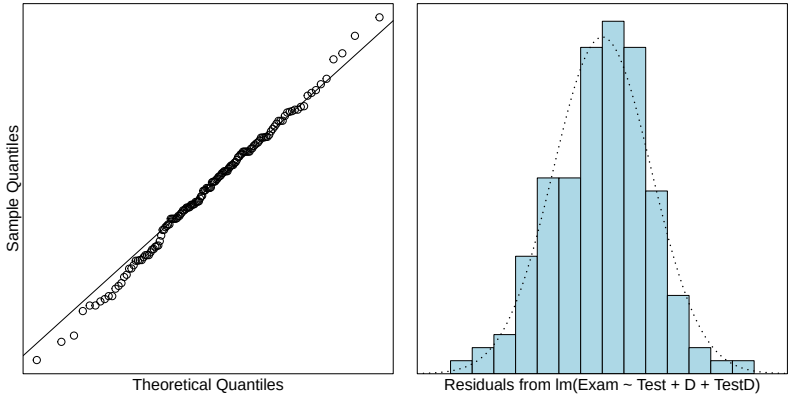


This looks okay. There is a narrowing at the higher values of fit, but there are fewer observations there.

Exam vs. test **and** attendance...

Assumption checks...

```
> normcheck(TestAttend.fit)
```

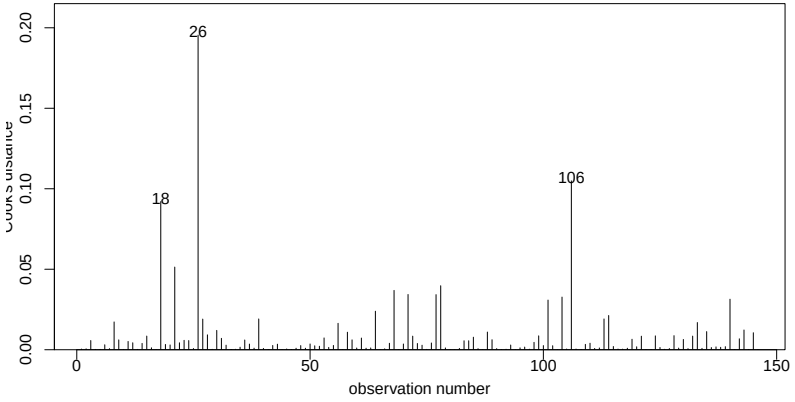


The normality assumption seems fine.

Exam vs. test **and** attendance...

Assumption checks...

```
> cooks20x(TestAttend.fit)
```



No unduly influential points here.

Exam vs. test **and** attendance...

Let us look at the fit

We can now trust the fitted `lm`. The summary output is:

```
> summary(TestAttend.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  14.4467     4.9443   2.922  0.00405 **
Test          2.7496     0.4603   5.973 1.78e-08 ***
D            -4.2582     6.3723  -0.668  0.50506
TestD         1.1380     0.5577   2.040  0.04316 *
---
Residual standard error: 11.41 on 142 degrees of freedom
Multiple R-squared:  0.6347,Adjusted R-squared:  0.627
F-statistic: 82.25 on 3 and 142 DF,  p-value: < 2.2e-16
```

Note that the coefficient for `TestD` is significant which means our intuition seem correct. That is, the regular attenders do get a 'greater return' on their `Test` 'investment'.

Exam vs. test **and** attendance...

Making sense of it all

Let's take a closer look at the model we have just fitted. We will produce a separate plot for each attender group.

Recall: `D = 0` if the student was a non-attender (the baseline level).

Therefore, the estimated coefficients that are associated with non-attenders are $\hat{\beta}_0$ and $\hat{\beta}_1$:

```
> coef(TestAttend.fit)[1:2]
(Intercept)        Test
  14.446750    2.749568
```


Exam vs. test **and** attendance...

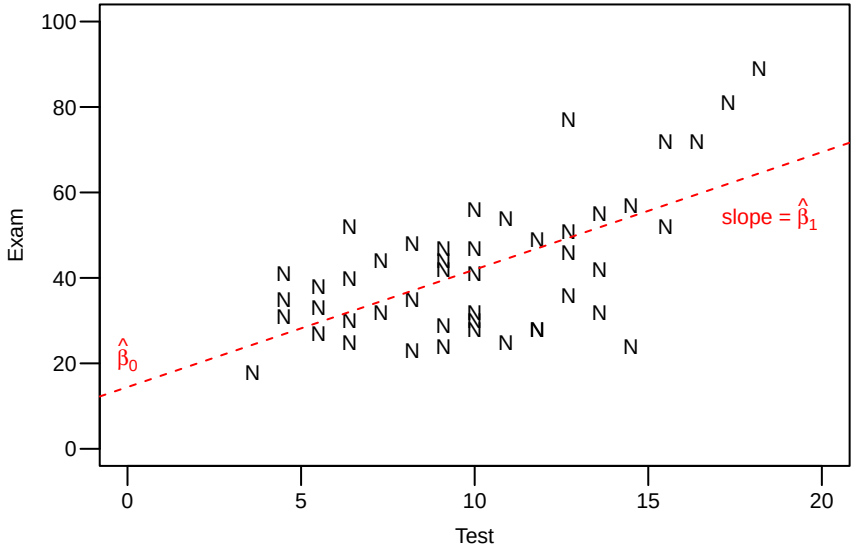
Making sense of it all...

Here is the `R` code for a plot of the non-attender group with their fitted line (`D = 0`).

```
> plot(Exam~Test,data=Absentees.df,pch="N",cex=0.7,xlim=c(0,20),ylim=c(0,100))
> abline(TestAttend.fit$coef[1:2],lty=2, col="red")
> text(0, 22,expression(hat(beta)[0]),col="red", cex = 0.7)
> text(18.5, 55, expression("slope = "*hat(beta)[1]),col="red", cex = 0.7)
```


Exam vs. test **and** attendance...

Making sense of it all...



Exam vs. test **and** attendance...

Making sense of it all...

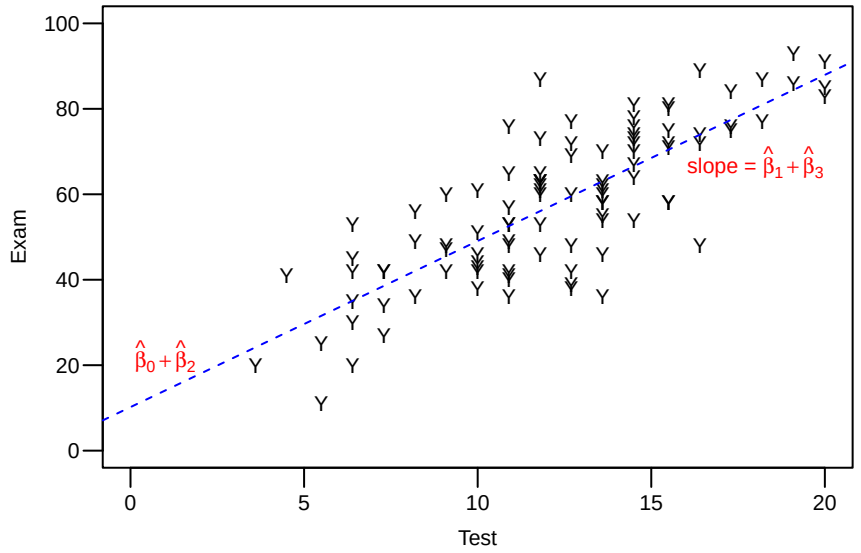
Here is the R code for a plot of the regular attender group with their fitted line (D = 1).

```
> plot(Exam ~ Test, data = Attendees.df, pch = "Y", cex = 0.7,
+      xlim = c(0, 20), ylim = c(0, 100))
> coeffs = TestAttend.fit$coef ## Easier to work with these terms
> abline(coeffs[1:2] + coeffs[3:4], lty = 2, col = "blue")
> text(1, 22, expression(hat(beta)[0] + hat(beta)[2]), col = "red", cex = 0.7)
> text(18, 67.5,expression(paste("slope = ", hat(beta)[1] + hat(beta)[3])),
+      col = "red", cex = 0.7)
```

Note that as D = 1 we add $\hat{\beta}_2$ and $\hat{\beta}_3$, to the baseline intercept and slope terms, respectively.

Exam vs. test **and** attendance...

Making sense of it all...



Exam vs. test **and** attendance...

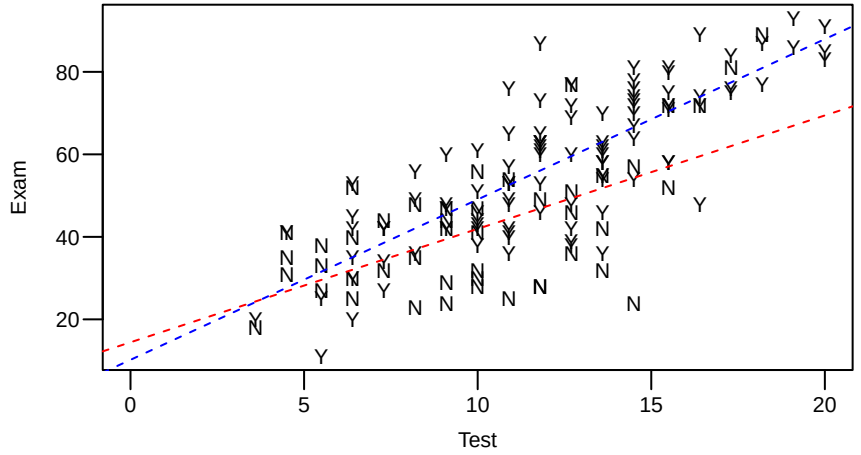
Making sense of it all...

All together now:

```
> ## Plot these data all together
> b=coef(TestAttend.fit) # easier to work with these terms
> plot(Exam ~ Test,data = Stats20x.df,pch=substr(Attend,1,1),cex=0.7,xlim=c(0,20))
> ## Red for "No" and blue for "Yes".
> abline(b[1:2], lty =2, col="red")
> abline(b[1]+b[3],b[2]+b[4],lty=2, col="blue" )
```


Exam vs. test **and** attendance...

Making sense of it all...



This a a visual confirmation of our intuition that the regular attenders get a 'greater return' on their Test 'investment'.

Fitting the interaction model directly with `lm`

All that hard work we did with constructing `D` and `TestD` can be avoided since `lm` will automatically do this for us.

We were interested to see whether the effect of `Test` interacts with the students `Attend` variable. Using `lm` we simply specify `Test * Attend` to fit the model with interaction. That is,

```
> TestAttend.fit2=lm(Exam~Test*Attend, data=Stats20x.df)
> summary(TestAttend.fit2)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    14.4467     4.9443   2.922  0.00405 **
Test            2.7496     0.4603   5.973 1.78e-08 ***
AttendYes      -4.2582     6.3723  -0.668  0.50506
Test:AttendYes  1.1380     0.5577   2.040  0.04316 *
---
Residual standard error: 11.41 on 142 degrees of freedom
Multiple R-squared:  0.6347,Adjusted R-squared:  0.627
F-statistic: 82.25 on 3 and 142 DF,  p-value: < 2.2e-16
```


Fitting the interaction model directly with `lm`...

Compare this with

```
> Summary(TestAttend.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  14.4467     4.9443   2.922  0.00405 **
Test          2.7496     0.4603   5.973 1.78e-08 ***
D            -4.2582     6.3723  -0.668  0.50506
TestD         1.1380     0.5577   2.040  0.04316 *
---
Residual standard error: 11.41 on 142 degrees of freedom
Multiple R-squared:  0.6347,Adjusted R-squared:  0.627
F-statistic: 82.25 on 3 and 142 DF,  p-value: < 2.2e-16
```

We have the same outputs, but with slightly different names.

Note: `Test * Attend` is shorthand notation. You can be more explicit about the individual terms in the model by writing

```
> TestAttend.fit2=lm(Exam ~Test + Attend + Test:Attend, data=Stats20x.df)
```

We read this as, *the effect of `Test`, plus the effect of `Attend`, plus the interaction between `Test` and `Attend`, which is denoted by `Test:Attend`.*

Section 8.3
Interpretting the fitted model

Exam vs. test **and** attendance...

Let us take stock of this model

```
> summary(TestAttend.fit2)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    14.4467     4.9443   2.922  0.00405 **
Test            2.7496     0.4603   5.973 1.78e-08 ***
AttendYes      -4.2582     6.3723  -0.668  0.50506
Test:AttendYes  1.1380     0.5577   2.040  0.04316 *
---
Residual standard error: 11.41 on 142 degrees of freedom
Multiple R-squared:  0.6347,Adjusted R-squared:  0.627
F-statistic: 82.25 on 3 and 142 DF,  p-value: < 2.2e-16
```

When we looked at the models that used explanatory variables `Test` and `Attend` by themselves (in Chapter 2 & 5 resp.), they explained 59% and 15% of the variability of `Exam` respectively. When we use them together, we explain about **63**% of the variability. **Why is it not** $59 + 15 = 74\%$**?**.

It is because the addition of `Attend` can only explain variability that has not already been explained by `Test`. If `Attend` and `Test` are closely related then this may not be much – this is called **multi-collinearity**.

Exam vs. test **and** attendance...

Let us take stock of this model...

We see that our intuition was correct. That is, the slope for `Test` of attenders is greater that for non-attenders. This is because the estimate of the difference in these slopes `TestD`.

```
> coef(TestAttend.fit2)[4]
Test:AttendYes
       1.13799
```

is positive and statistically significant (P-value ≈ 0.04).

We were not sure about the differences in intercept and we see that this estimate is not significantly different from the hypothesised value of 0 (P-value ≈ 0.51).

Recall that we are not particularly interested in the null hypothesis $H_0 : \beta_2 = 0$ when the slopes are different, so it is standard practice to leave this term in the model when β_3 is statistically significant.

Exam vs. test **and** attendance...

Let us take stock of this model...

The baseline slope that measures the effect of `Test` for non-attenders is statistically significant.

The slope for attenders is significantly higher, so we say that the effect of `Test` interacts with the `Attend` variable, as the effect depends on the level of `Attend`.

Exam vs. test **and** attendance...

Some Inference

Confidence intervals may be needed for the coefficients:

```
> confint(TestAttend.fit2)
                    2.5 %    97.5 %
(Intercept)      4.67287511 24.220625
Test             1.83956971  3.659567
AttendYes      -16.85506294  8.338572
Test:AttendYes   0.03547053  2.240510
```

Recall: Statistical significance (at the 5% level) of a coefficient is equivalent to the (95%) confidence interval **NOT** containing zero.

Exam vs. test **and** attendance...

Some predictions

```
> predTestAttend.df = data.frame(Test = c(0, 10, 10, 20),
+                                Attend = factor(c("No", "No", "Yes", "Yes"))
+                                )
> predTestAttend.df
  Test Attend
1    0     No
2   10     No
3   10    Yes
4   20    Yes
```

Let us estimate the expected exam scores for these values of test score and attendance:

```
> predict(TestAttend.fit2,predTestAttend.df, interval="confidence")
       fit       lwr      upr
1 14.44675  4.672875 24.22062
2 41.94243 38.616376 45.26849
3 49.06409 46.412194 51.71599
4 87.93968 82.610100 93.26926
```

Exam vs. test **and** attendance...

Some predictions...

Now, let us predict the exam score for individual students with those test scores and attendance:

```
> predict(TestAttend.fit2,predTestAttend.df, interval="prediction")
       fit       lwr       upr
1 14.44675 -10.13028  39.02378
2 41.94243  19.14848  64.73639
3 49.06409  26.35871  71.76947
4 87.93968  64.76845 111.11092
```

This is not the best model for predicting individual student exam scores as the intervals are too wide and in some case are meaningless (at the extremes of `Test`).

This is related to the fact that we can only explain 63% of the variability in `Exam` or, equivalently, we can not account for 37% of this variability. Also, our linear model does not "know" about the constraint that exam scores must be between 0 and 100.

Exam vs. test **and** attendance...

Methods and assumption checks

A plot of the data showed that exam scores appear to increase linearly with test score, but with different lines for attenders and non-attenders.

The model with interaction was fitted and the attendance/test score interaction was found to be significant.[5]

The students should be acting independently of each other as this was an exam.

The EOV assumption appears to be reasonable notwithstanding some reduced variability at the high end of test and exam scores. The data also look approximately normal. There do not appear to be any unduly influential data points.

Our model explains 63% of the total variability in exam scores.

[5] See the Case Study for the model equation.

Exam vs. test **and** attendance...

Executive summary

There is a clear linear relationship between Test and Exam, but it differs in strength between non-attenders and attenders.

We estimate that each additional test mark (out of 20) will increase the expected exam mark of a non-attender by 1.8 to 3.7.

There is a further increase of between 0 to 2.2 marks for regular attenders.

Note: In an assignment or exam situation you might also be required to comment on additional confidence and/or prediction intervals—this will be made clear in the instructions.

Exam vs. test **and** attendance...

Executive summary...

Note that the above Executive Summary is missing the confidence interval for the effect of test mark on attenders. To obtain this CI we need to change attenders to the baseline level of `Attend`.

```
> Stats20x.df$Attend2=relevel(Stats20x.df$Attend,ref="Yes")
> TestAttend.fit2b=lm(Exam ~Test*Attend2, data=Stats20x.df)
> coef(summary(TestAttend.fit2b))
                Estimate Std. Error    t value     Pr(>|t|)
(Intercept)    10.188504  4.0199956  2.5344566 1.234648e-02
Test            3.887559  0.3148792 12.3461898 3.020895e-24
Attend2No       4.258246  6.3722922  0.6682439 5.050626e-01
Test:Attend2No -1.137990  0.5577265 -2.0404094 4.316270e-02
> confint(TestAttend.fit2b)
                   2.5 %      97.5 %
(Intercept)     2.241733 18.13527583
Test            3.265102  4.51001561
Attend2No      -8.338572 16.85506294
Test:Attend2No -2.240510 -0.03547053
```

We estimate that each additional test mark will increase the expected exam mark of an attender by 3.3 to 4.5.

Section 8.4
Assessing influence of the atypical students

Exam vs. test **and** attendance...

Sensitivity check

In the exploratory phase of the analysis we identified three students as being potentially anomolous. Recall, these were the 3 non-attending students who scored greater than 16 on the test.

If we have reason to think these students are 'atypical' (do we?) then it might make sense to do the analysis without them.

What do you think happens to our respective straight lines?

Let us see what happens. We will pull these three students out of the dataframe.

```
> ## Remove atypical points - Note that ! means 'not'
> Subset.df=subset(Stats20x.df, !(Test>16&Attend=="No"))
> TestAttend.fit3=lm(Exam ~Test*Attend, data=Subset.df)
> summary(TestAttend.fit3)
```


Exam vs. test **and** attendance...

Sensitivity check...

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    22.6522     5.2776   4.292  3.3e-05 ***
Test            1.7590     0.5213   3.374 0.000960 ***
AttendYes     -12.4637     6.5505  -1.903 0.059146 .
Test:AttendYes  2.1285     0.6034   3.527 0.000569 ***
---
Residual standard error: 11.01 on 139 degrees of freedom
Multiple R-squared:  0.6495,Adjusted R-squared:  0.6419
F-statistic: 85.86 on 3 and 139 DF,  p-value: < 2.2e-16
```

Note that we have smaller degrees of freedom as we have 3 fewer data points. Look at how the parameters have changed.

Our R^2 value has increased slightly as we have less variability overall.

Exam vs. test **and** attendance...

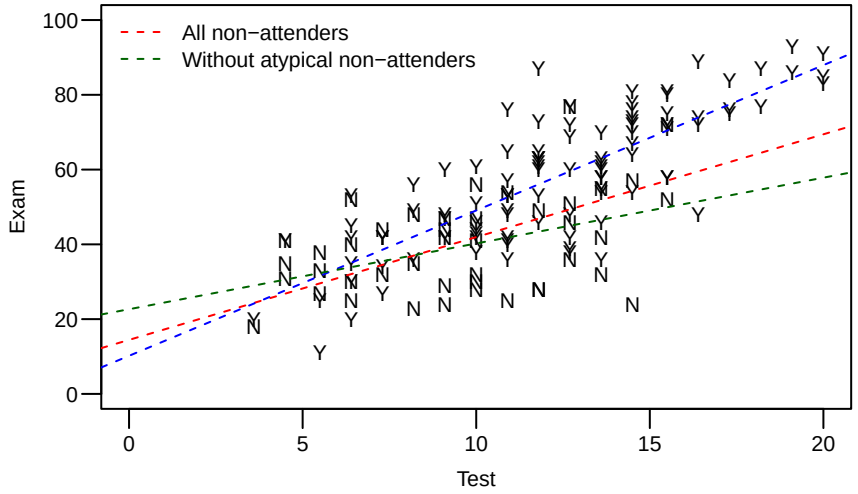
Sensitivity check...

R code to generate the plot of the full data with all three of the fitted lines:

```
> ## Plot these data all together
> plot(Exam ~ Test, data = Subset.df,pch = substr(Attend, 1, 1),cex = 0.7)
>
> ## Remember that we've defined b in Slide 26
> ## Each abline() will have a different colour
> abline(b[1:2], lty = 2, col = "red")
> abline(b[1] + b[3], b[2] + b[4], lty = 2, col = "blue")
>
> ## The fitted line without the 3 atypical points
> b2 = coef(TestAttend.fit3) ## Easier to work with these terms
> abline(b2[1:2], lty = 2, col = "green")
>
> ## Add a legend to help us differentiate between the lines for non-attenders
> legend("topleft", legend = c("All non-attenders", "Without atypical non-attenders
+        lty = 2, col = c("red", "green"), bty = "n")
```

Exam vs. test **and** attendance...

Sensitivity check...



Comments?

Section 8.5
Some insight and relevant R-code.

R tips and tricks

`model.matrix`

Recall that `lm` automatically created the necessary indicator variables to fit the above models.

To explicitly see the model formula that `lm` is using, we only have to ask:

```
> ModMat=model.matrix(~Test*Attend,data=Stats20x.df)
> cbind(Stats20x.df[,c("Test","Attend")],ModMat)[1:10,]
   Test Attend (Intercept) Test AttendYes Test:AttendYes
1   9.1    Yes           1  9.1         1           9.1
2  13.6    Yes           1 13.6         1          13.6
3  14.5    Yes           1 14.5         1          14.5
4  19.1    Yes           1 19.1         1          19.1
5   8.2     No           1  8.2         0           0.0
6  12.7    Yes           1 12.7         1          12.7
7   7.3    Yes           1  7.3         1           7.3
8  10.9     No           1 10.9         0           0.0
9  10.9    Yes           1 10.9         1          10.9
10  9.1    Yes           1  9.1         1           9.1
```

The `AttendYes` variable is `D`, and `Test:AttendYes` is `D` × `Test`.

Most of the R-code you need for this chapter

As always, your code requires the usual code (data exploration, etc) and model checks discussed in chapters 1 and 2.[6]

When y can be explained by a categorical (i.e., factor) variable and also a numeric (i.e., continuous) variable then you can use both.

You do not need to create indicator variables as `R` does this for you. It will choose the baseline for you, so be careful. You can change this if needed, using the `relevel` function.

Fit as follows:

```
> TestAttend.fit2=lm(Exam ~Test*Attend, data=Stats20x.df)
> #check to see it's okay
> plot(TestAttend.fit2, which=1) #followed by normcheck and cooks20x
> # then see if you need a separate slope for each level of your factor var.
> summary(TestAttend.fit2)
```

Interpret accordingly. In particular, if the interaction term is not significant then proceed to the next Chapter.

[6] That is, until we reach Chapter 13.

Case Study 8.1: Exam vs attendance and test mark

Tou Ohone Andate - staff number 1234567

Problem

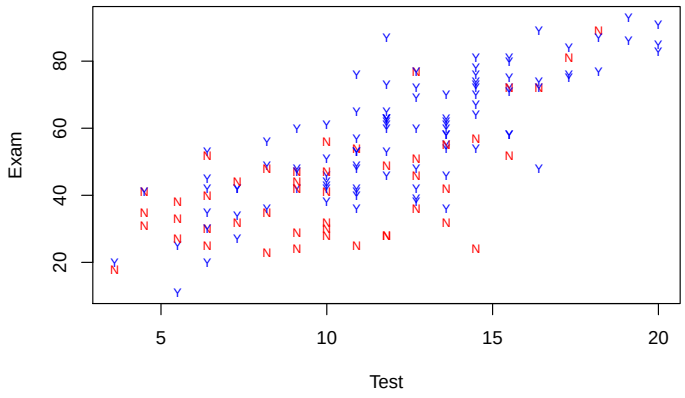
We've previously used test marks and attendance seperately in order to explain variability in exam marks. The objective here is to use them together.

Question of Interest

To quantify students' exam marks relationship with attendance and test marks. Also, does any relationship between exam marks and test marks depend on whether students attended lectures.

Read in and Inspect the Data

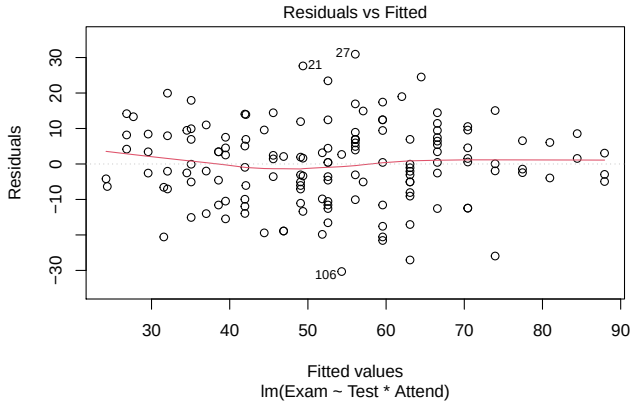
```
Stats20x.df = read.table("STATS20x.txt", header = T)
plot(Exam ~ Test, data = Stats20x.df, pch = substr(Attend, 1, 1), cex = 0.7,
    col = ifelse(Attend == "Yes", "blue", "red"))
```



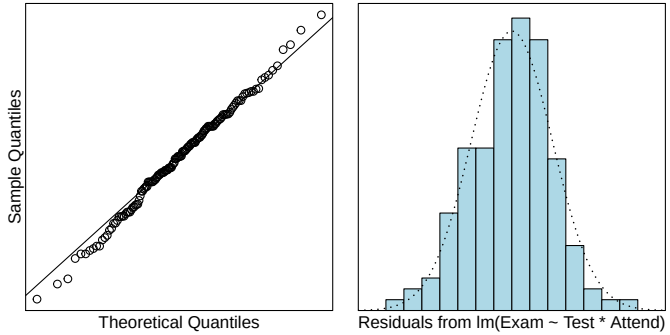
A scatter plot of test score versus exam suggested that the positive relationship between test and exam was reasonably linear within each attendance group ("Yes" or "No"), but that the slope could be different in the two groups.

Model Building and Check Assumptions

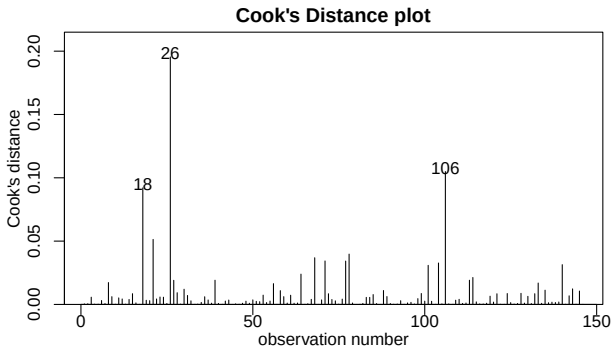
```
examTestAttend.fit = lm(Exam ~ Test * Attend, data = Stats20x.df)
plot(examTestAttend.fit, which = 1)
```




```
normcheck(examTestAttend.fit)
```



```
cooks20x(examTestAttend.fit)
```




```
summary(examTestAttend.fit)
```

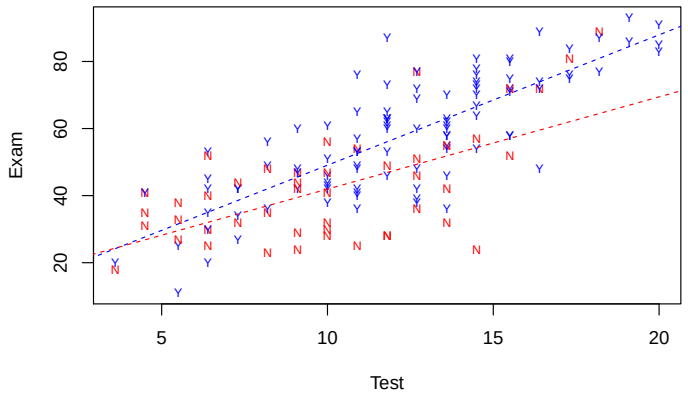
```
##
## Call:
## lm(formula = Exam ~ Test * Attend, data = Stats20x.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -30.3155  -6.5139   0.4383   7.3166  30.9383
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     14.4467     4.9443   2.922  0.00405 **
## Test             2.7496     0.4603   5.973 1.78e-08 ***
## AttendYes       -4.2582     6.3723  -0.668  0.50506
## Test:AttendYes   1.1380     0.5577   2.040  0.04316 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.41 on 142 degrees of freedom
## Multiple R-squared:  0.6347, Adjusted R-squared:  0.627
## F-statistic: 82.25 on 3 and 142 DF,  p-value: < 2.2e-16
```

```
confint(examTestAttend.fit)
```

```
##                      2.5 %    97.5 %
## (Intercept)      4.67287511 24.220625
## Test             1.83956971  3.659567
## AttendYes      -16.85506294  8.338572
## Test:AttendYes   0.03547053  2.240510
```


Visualise the Final Model

```
predAttend.df = data.frame(Test = 1:21, Attend = "Yes")
predSlackers.df = data.frame(Test = 1:21, Attend = "No")
plot(Exam ~ Test, data = Stats20x.df,pch = substr(Attend, 1, 1), cex = 0.7,
    col = ifelse(Attend == "Yes", "blue", "red"))
lines(1:21, predict(examTestAttend.fit, predAttend.df), col = "blue", lty = 2)
lines(1:21, predict(examTestAttend.fit, predSlackers.df), col = "red", lty = 2)
```



Method and Assumption Checks

As we have two explanatory variables, one numeric and one factor, we have fitted a linear model that used different intercept and slopes for each attendance group (i.e., interaction model). We could not drop the interaction term ($P\text{-}value = 0.043$).

All model assumptions were satisfied.

Our final model is

$$Exam_i = \beta_0 + \beta_1 \times Test_i + \beta_2 \times Attend_i + \beta_3 \times Attend_i \times Test_i + \epsilon_i,$$

where $Attend_i = 1$ if student i is a regular attender, otherwise 0, and $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Our model explained a modest 63% of the variability in students' exam marks.

Executive Summary

We wanted to quantify students' exam marks relationship with attendance and test marks.[1]

[1] Since there are different slopes in the two groups, we need to discuss each slope individually.

There was a clear linear relationship between test and exam scores, but this relationship differed between students who attended and who did not attend lectures.

We estimate that each additional test mark (out of 20) obtained by a non-attending student would increase their expected exam mark by between 1.8 to 3.7.

For regular attenders, the increase is an additional 0.04 to 2.2 expected exam marks per test mark.

Case Study 8.2: Refractive Index Calibration

James Curran

Note this handout requires that you install the package `dafs` before you can try the examples.

Glass evidence can be important to forensic scientists. If a window, or some other source of glass, is broken during the commission of a crime, then tiny fragments maybe transferred to the breaker. Studies have shown that glass from the same source has a similar refractive index (RI), whereas glass from different sources usually have very different RIs. Therefore, the presence of glass fragments on the clothing of a person of iterest (POI) that have similar RI values to a broken source at a crime scene can associate that individual with the crime.

Refractive index describes the way light "bends" when it passes from one optical medium (like air) to medium of a different density (like glass or water). If you want to think about this, then think how a pencil appears to be "cut" when you see it viewed through a glass of water. This is because both the glass and the water are more dense than the air. RI of glass is determined by placing it in silicone oil. The optical density of silicone oil can changed by heating it or cooling it. A glass fragment is placed in a droplet of silicone oil which is then heated until the glass becomes optically indistinguishable from the oil. The temperature at which this happens is recorded. The the oil is cooled and the temperature at which the glass reappears is recorded. The average of these two temperatures is called the match temperature. This match temperature is then translated into RI by means of a calibration line. Calibration is the process where by a set of "standards"" with known Y values are measured with an instrument to observe the corresponding X. These pairs of values can be used to produce a calibration curve which can be used to interpolate (or predict) the unknown value of Y for an observed X. In some sense this is "inverse regression"—usually we know X and we measure Y.

In this example we have a set of 12 standards (labelled B1–B12) of known RI value. Each of these standards has been measured multiple times by two different forensic scientists, Rachel Bennett (RB), and my Polish colleague Grzegorz "Greg" Zadora (GZ). The RI and match temperature has been recorded for each measurement of each standard by each scientist. We want to know whether the two different labs/scientists have two different calibration lines.

```
data("ri.calibration.df")

## Let's make the name shorter so it is easier to work with
ri.df = ri.calibration.df
names(ri.df)
```

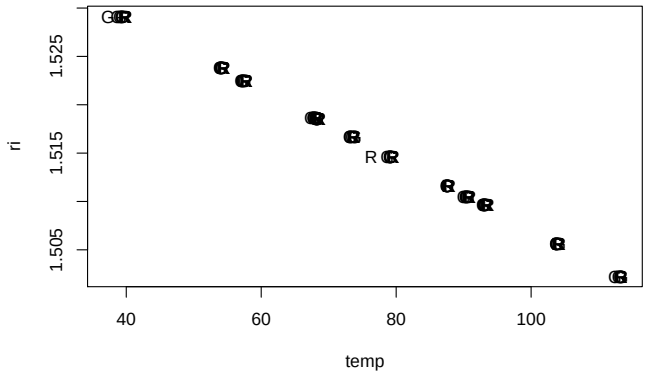
```
## [1] "standard" "ri"       "temp"     "owner"
```

```
head(ri.df)
```

```
##   standard      ri  temp owner
## 1       B1 1.52912 39.97    RB
## 2       B1 1.52912 39.90    RB
## 3       B1 1.52912 39.97    RB
## 4       B1 1.52912 39.70    RB
## 5       B1 1.52912 39.76    RB
## 6       B2 1.52381 54.50    RB
```

First we plot the data.

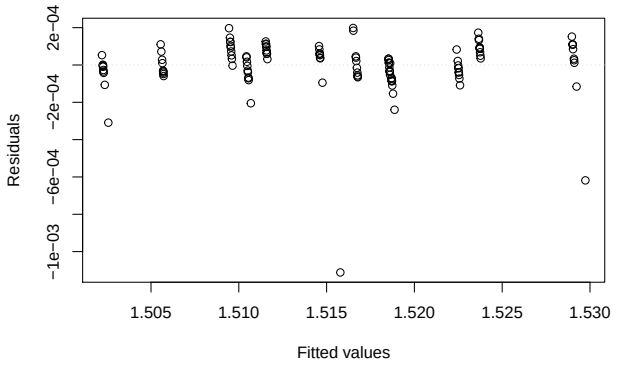
```
plot(ri ~ temp, pch = substr(ri.df$owner, 1, 1), data = ri.df)
```

It is really hard to tell if there is a difference from this plot. One of the reasons is that this data is super precise. We can see a couple of potential outliers.

We will fit a model. We will ignore the fact that we have multiple measurements on the same standard in this analysis.

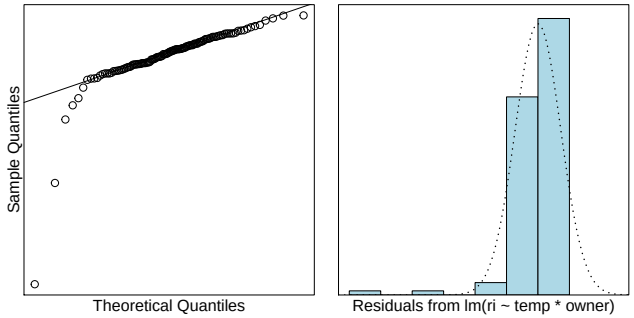
```
ri.fit = lm(ri ~ temp * owner, data = ri.df)
eovcheck(ri.fit)
```



The data is very patterned. We expect this because each standard has been measured multiple times. We can see there are 2–3 big outliers, but basically equality of variance looks fine.

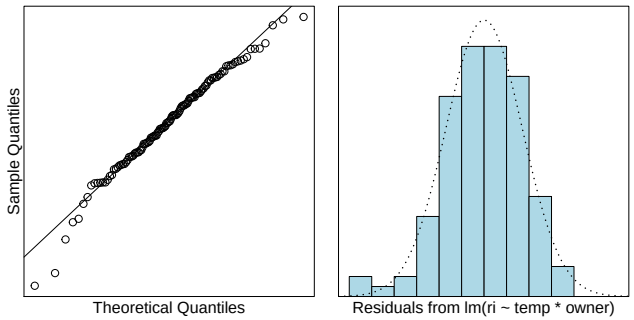
How about normality?

```
normcheck(ri.fit)
```



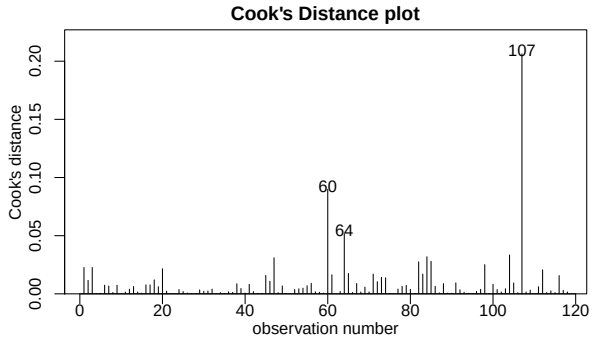
Looks pretty good again except for some outliers. Probably if we got rid of points 31 and 109, things would be much better. Let's try that:

```
ri.fit2 = lm(ri ~ temp * owner, data = ri.df, subset = -c(31, 109))
normcheck(ri.fit2)
```



That looks way better. The picture isn't being distorted by huge outliers. How about influential points. Do we really expect any?

```
cooks20x(ri.fit2)
```



Nothing much to worry about there. So we can trust the inference.

```
summary(ri.fit2)
```

```
##
## Call:
## lm(formula = ri ~ temp * owner, data = ri.df, subset = -c(31,
##     109))
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -2.834e-04 -5.323e-05  1.866e-06  5.897e-05  1.836e-04
##
## Coefficients:
##                Estimate Std. Error   t value Pr(>|t|)
## (Intercept)   1.543e+00  4.481e-05 34444.651  < 2e-16 ***
## temp         -3.627e-04  5.569e-07  -651.328  < 2e-16 ***
## ownerRB       2.701e-04  6.253e-05     4.320 3.34e-05 ***
## temp:ownerRB -1.643e-06  7.778e-07    -2.113   0.0368 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.629e-05 on 114 degrees of freedom
## Multiple R-squared:  0.9999, Adjusted R-squared:  0.9999
## F-statistic: 2.916e+05 on 3 and 114 DF,  p-value: < 2.2e-16
```

```
p = summary(ri.fit2)$coefficients[4, 4]
b = coef(ri.fit2)
```

We first look at the overall P-value, which is very small (< 2.2e-16). This says our variables are important in predicting the RI value.

Next, we can see that the interaction term `temp:ownerRB` is significant with a P-value of 0.0368. Therefore we have evidence to suggest that the calibration lines are different for Rachel and for Greg. Greg will use the fitted model

$$\hat{RI} = 1.543325 + -0.000363 \times temp$$

whereas Rachel will use the fitted model

$$\hat{RI} = 1.543595 + -0.000364 \times temp$$

These may look mind-bogglingly close to each other, but in the RI world where diferences in the fourth to sixth decimal place are critical, these are quite different lines. You can also see that this is a very good model for prediction because R^2 is almost 1.

Chapter 9:
Linear models with both numeric and factor explanatory variables

Part 2: The model without interaction

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

- More about linear models with both categorical and numeric explanatory variables
- Model selection using Occam's razor
- The `anova` function for joint hypothesis tests
- Fitting a model without an interaction term
- Interpretting fitted models with factors having more than two levels
- Changing the reference level of a categorical variable
- Relevant `R`-code.

Section 9.1
Example: Using both IQ and teaching method to explain increase in language proficiency

Student language score by teaching method and IQ

In the following example, educational experts were interested in which of three different teaching methods was most effective in increasing a language score for students of a range of abilities – as measured by IQ.

In order to do this, 30 students were randomly allocated into three groups and each group was taught using a different teaching method. Each student's IQ was measured before the teaching programme began.

This randomisation was done to ensure that a range of student abilities was represented in each group. As students were in a test environment we can assume that their test scores are independent of each other.

The variables measured were:

`lang`	the language test score achieved by the student after instruction
`IQ`	the student's IQ
`method`	the teaching method used (1, 2 or 3)

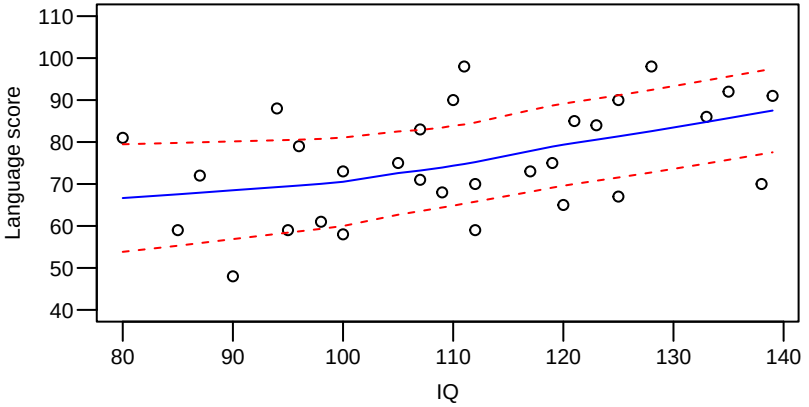
Student language score by teaching method and IQ...

It was of interest to see if there was any difference in the students' expected language score between teaching methods, and if this difference depended on IQ.

As usual, we begin by inspecting the data:

```
> ## Invoke the s20x library
> library(s20x)
> ## Importing data found in the s20x library into R
> data(teach.df)
> ## Plot the data with trendscatter()
> trendscatter(lang ~ IQ, f = 0.8, ylim = c(40, 110), data = teach.df,
+              ylab="Language score")
> ## Note that f is the proportion of points in the plot which influence the
> ## smooth at each value. Larger values of f give more smoothness!
```


Student language score by teaching method and IQ...



Hmmm – positive relationship with IQ suggested, but statistical significance not obvious. A plot that shows the teaching method would be more useful.

Student language score by teaching method and IQ...

In dataframe `teach.df` the `method` is recorded as a number, 1, 2 or 3.

```
> teach.df$method
 [1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3
> class(teach.df$method)
[1] "integer"
```

However, these are just labels and could as easily have been "A", "B" or "C". So, `method` needs to be 'coerced' into a factor so that it does not get treated as a numeric variable[1].

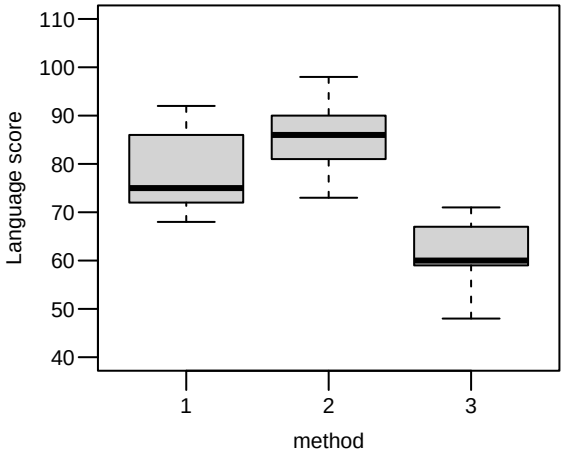
```
> teach.df$method = factor(teach.df$method)
> teach.df$method
 [1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3
Levels: 1 2 3
> class(teach.df$method)
[1] "factor"
```

Not much has changed, but when we use `method` as the explanatory variable, plot functions and `lm` will now do the right thing since it is now a factor variable.

[1] What would happen if `lm` was treated `method` as numeric???

Student language score by teaching method and IQ...

```
> plot(lang ~ method, ylim=c(40,110), data=teach.df, ylab="Language score")
```

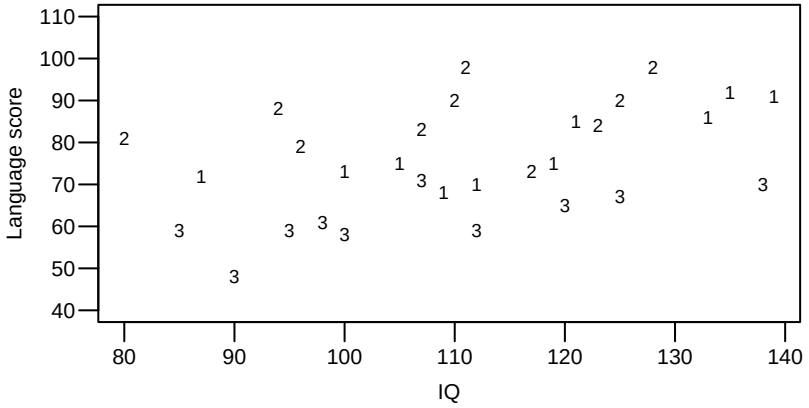


Teaching method 2 seems best. The worst appears to be method 3.

Student language score by teaching method and IQ...

A more useful plot:

```
> plot(lang ~ IQ, ylim=c(40,110), pch=as.character(method), data = teach.df,
+      ylab="Language score")
```



What do you see in this plot?

Student language score by teaching method and IQ...

What we see here is that the data for methods 1, 2, 3 seem to be scattered around approximately parallel straight lines. This means the `IQ` effect appears the same regardless of the teaching method. Conversely, the `Method` effect appears to be the same regardless of IQ.

In other words, it looks like `IQ` and `method` **do not** interact, i.e., there is no need to complicate the model by using a different slope for each method.

When we look at this data we see that the method 2 students are (on average) doing considerably better than method 1 students, who are (on average) better than the method 3 students.

So, we suspect that we there may be a difference in the expected teaching scores between methods, and that the effect of IQ may be the same regardless of method. We'll need to fit an appropriate linear model to find out for sure.

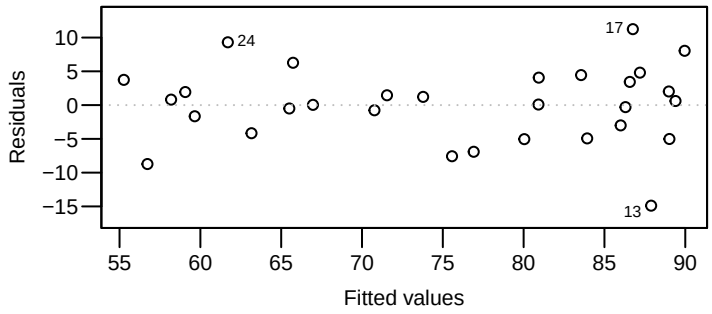
Student language score by teaching method and IQ...

Fitting and checking the interaction model

We will fit the model with interaction first, anticipating that the interaction will not be significant.

Here goes..., along with the usual assumption checks.

```
> TeachIQmethod.fit=lm(lang~IQ*method, data=teach.df)
> plot(TeachIQmethod.fit, which = 1)
```

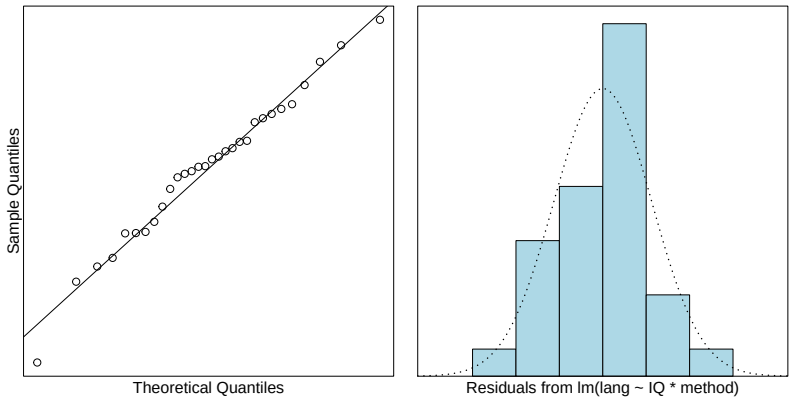


The **EOV** and no-trend assumptions seem to be okay.

Student language score by teaching method and IQ...

Fitting and checking the interaction model...

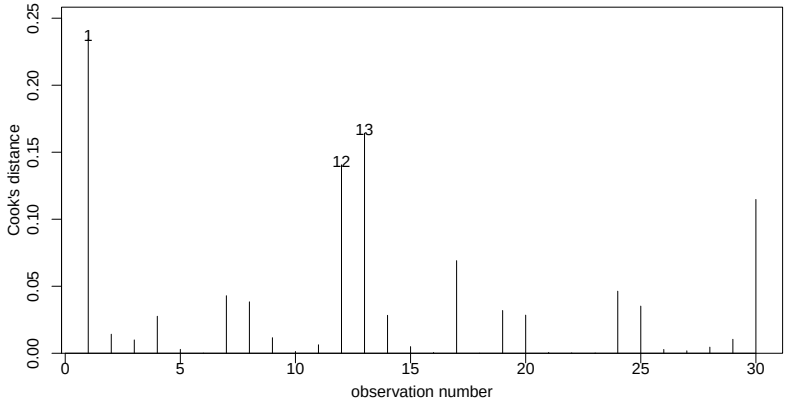
```
> normcheck(TeachIQmethod.fit)
```



Student language score by teaching method and IQ...

Fitting and checking the interaction model...

```
> cooks20x(TeachIQmethod.fit)
```



It looks like we can trust the output of the fitted model.

Section 9.2
Model selection using Occam's razor

Student language score by teaching method and IQ...

The fitted interaction model

Our fitted interaction model is:

```
> summary(TeachIQmethod.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  26.8346    14.5250   1.847  0.07704 .
IQ            0.4471     0.1241   3.604  0.00142 **
method2      39.0098    20.7473   1.880  0.07227 .
method3       3.5617    19.7222   0.181  0.85820
IQ:method2   -0.2587     0.1831  -1.413  0.17042
IQ:method3   -0.1546     0.1749  -0.883  0.38574
---
Residual standard error: 6.199 on 24 degrees of freedom
Multiple R-squared:  0.8121,Adjusted R-squared:  0.7729
F-statistic: 20.74 on 5 and 24 DF,  p-value: 5.284e-08
```


Model selection using Occam's razor

In previous Chapters we've seen that we remove terms that are not significant if doing so simplifies the fitted model. This is a very important principle of model selection and is an application of the "principle of parsimony", also known as **Occam's Razor**.[2] :

"The principle states that among competing models that predict equally well, the one with the fewest parameters should be selected."

In STATS20x we sometimes call it the **KISS** principle – "keep it simple, statistician".

In this class the general model selection approach we use is to do a hypothesis test to determine if we can remove the most complicated term from our current model.[3]

[2] Named after William of Ockham (c. 1287-1347), who was an English Franciscan friar and scholastic philosopher and theologian.

[3] In STATS 330 you will several holistic approaches to model selection that are based on the estimated predictive ability of the model.

Student language score by teaching method and IQ...

Model selection using Occam's razor

In our above interaction model for language score, the interaction term is the most complicated term, so we need to test whether the interaction is significant.

Hmmm, the null hypothesis of no interaction is saying that the last two coefficients in the summary table are both zero. That is,
$H_0 : \beta_4 = \beta_5 = 0$.[4]

To do a single test about whether two or more coefficients are zero we need to produce an ANOVA table, using the `anova` function.

[4] A hypothesis that specifies the value of two or more coefficients is called a **joint** hypothesis.

Student language score by teaching method and IQ...

Model selection using Occam's razor...

```
> anova(TeachIQmethod.fit)
Analysis of Variance Table

Response: lang
          Df  Sum Sq Mean Sq F value    Pr(>F)
IQ         1 1004.42 1004.42 26.1416 3.124e-05 ***
method     2 2901.83 1450.91 37.7625 3.867e-08 ***
IQ:method  2   78.82   39.41  1.0257    0.3737
Residuals 24  922.13   38.42
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The *P*-value associated with the interaction term `IQ:method` is large. So we conclude we do not have evidence of an interaction.

This means we do not need to compute a different slope for each teaching method – just different intercepts. Our instincts were right. This saves us having to have an extra two parameters in our final model, and simplifies the interpretation of our model since the lines are parallel.

Understanding the ANOVA table

An `anova` table is based on partitioning the total sums of squares, TSS. Recall, this is the residual sums of squares of the null model. The TSS is given by summing the values in the `Sum sq` column of the table.

In the above table, TSS $= 1004.42 + 2901.83 + 78.82 + 922.13 = 4907.2$. The amount of unexplained variability in our fitted model is given by the `Residuals` value of 922.13. This means that $922.13/4907.2 = 0.188$ is the proportion of the total variability that is not explained. The R^2 is therefore $1 - 0.188 = 0.812$.

Now, back to our example, where we now need to consider the formulation of the parallel line model.

Student language score by teaching method and IQ...

Fitting the model without interaction

Occam's razor dictates that we sharpen our model by removing the interaction term. To do this, we simply replace `*` by `+` in the model formula.

Non-interaction models are sometimes referred to as *additive models* (as the effects 'add up'), or 'main effects' models.

```
> TeachIQmethod.fit2=lm(lang~IQ+method, data=teach.df)
> summary(TeachIQmethod.fit2)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  42.08552    8.73921   4.816 5.47e-05 ***
IQ            0.31564    0.07341   4.299 0.000213 ***
method2       9.87793    2.82068   3.502 0.001688 **
method3     -14.15922    2.85240  -4.964 3.70e-05 ***
---
Residual standard error: 6.205 on 26 degrees of freedom
Multiple R-squared:  0.796,Adjusted R-squared:  0.7725
F-statistic: 33.82 on 3 and 26 DF,  p-value: 3.986e-09
```

All the assumptions are fine (not shown) – not surprising, as all we have done is remove a term that was not significant.

Student language score by teaching method and IQ...

Interpretting the no-interaction model

The equation for the parallel lines (i.e, no-interaction) model is

$$\texttt{lang} = \beta_0 + \beta_1 \times \texttt{IQ} + \beta_2 \times \texttt{D2} + \beta_3 \times \texttt{D3} + \varepsilon$$

where, as usual $\varepsilon \overset{iid}{\sim} N(0, \sigma^2)$.
There are two indicator variables since teaching method has three levels:

- `D2` is an indicator variable whereby: `D2 = 1` if teaching method 2 is taught – otherwise it is 0.
- `D3` is an indicator variable whereby: `D3 = 1` if teaching method 3 is taught – otherwise it is 0.
- Teaching method 1 is the **reference/baseline** level group.

β_2 and β_3 represent the change in expected score (for any fixed student IQ) when we compare teaching method 2 or 3 to teaching method 1 (baseline).

Student language score by teaching method and IQ...

Model selection using Occam's razor...

The KISS principle requires us to determine if we can further simplify our model.

The no-interaction model can be simplified by removing the `IQ` term and/or the `method` terms. We see immediately from the summary table that `IQ` is highly significant, and so should not be removed.

To see if the `method` terms can be removed we want to test the joint null hypothesis that the intercepts are all identical, $H_0 : \beta_2 = \beta_3 = 0$. This would simplify our model to a simple linear regression model.

Recall: To do a test about whether two or more coefficients are zero we need to produce an ANOVA table, using the `anova` function.

Student language score by teaching method and IQ...

Model selection using Occam's razor...

Let us see if we really do have identical intercepts.

```
> anova(TeachIQmethod.fit2)
Analysis of Variance Table

Response: lang
          Df Sum Sq Mean Sq F value    Pr(>F)
IQ         1 1004.4  1004.4  26.090 2.529e-05 ***
method     2 2901.8  1450.9  37.688 2.077e-08 ***
Residuals 26 1001.0    38.5
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The *P*-value associated with the `method` term is very small, so we conclude that the intercepts are different.

We do have to fit different intercepts for each teaching method. Our instincts were right.

Student language score by teaching method and IQ...

Our preferred model

Our preferred model is the no-interaction model:

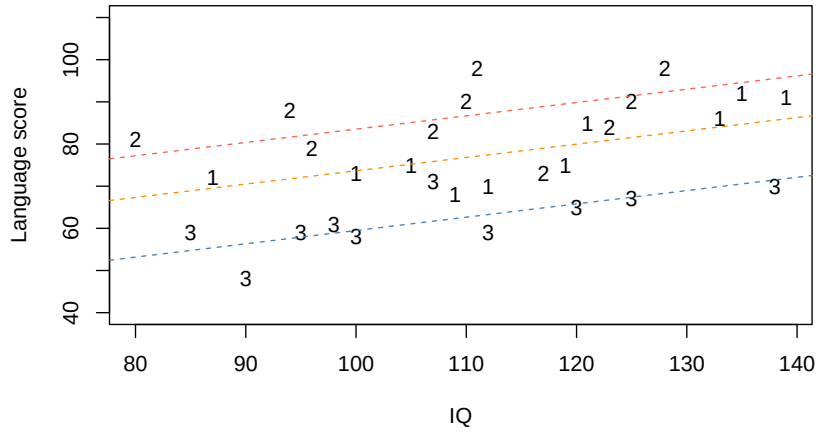
```
> summary(TeachIQmethod.fit2)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  42.08552    8.73921   4.816 5.47e-05 ***
IQ            0.31564    0.07341   4.299 0.000213 ***
method2       9.87793    2.82068   3.502 0.001688 **
method3     -14.15922    2.85240  -4.964 3.70e-05 ***
---
Residual standard error: 6.205 on 26 degrees of freedom
Multiple R-squared:  0.796,Adjusted R-squared:  0.7725
F-statistic: 33.82 on 3 and 26 DF,  p-value: 3.986e-09
```

Student language score by teaching method and IQ...

The fitted model

```
> plot(lang ~ IQ, ylim=c(40,110), pch=as.character(method), data = teach.df,
+      ylab="Language score")
> b = coef(TeachIQmethod.fit2)
> abline(b[1], b[2], lty = 2, col = "darkorange")
> abline(b[1] + b[3], b[2], lty = 2, col = "tomato")
> abline(b[1] + b[4], b[2], lty = 2, col = "steelblue")
```



Student language score by teaching method and IQ...

Interpreting the output...

We are now able to deduce:

- $\beta_1 > 0 \implies$ [5] IQ has a common positive effect on the expected language score of all students
- $\beta_2 > 0 \implies$ teaching method 2 is better than teaching method 1 regardless of a student's IQ.
- $\beta_3 < 0 \implies$ teaching method 3 is worse than teaching method 1 regardless of a student's IQ.

Our initial hunch about the best model has been justified.

[5] $\implies$ means *implies that*.

Section 9.3
Changing the reference level of teaching method

Student language score by teaching method and IQ...

Changing the reference level

The above statements are useful, but are incomplete – we still don't know how method 2 differs from method 3?[6]

The lm function has chosen the baseline (i.e., reference) level to be method 1 since "1" comes before "2" and "3" when these values are sorted by a computer. We need to change this to make method 2 (or alternatively method 3) the baseline. The fitted model will be exactly the same, but the intercept coefficients will change due to the change in reference level.

Here is the R code to do this:

```
> teach.df$method = relevel(teach.df$method, ref = "2")
> TeachIQmethod.fit3=lm(lang~IQ+method, data=teach.df)
```

Let us compare the summarys of TeachIQmethod.fit3 and TeachIQmethod.fit2.

[6] The issue of estimating all pairwise differences between factor levels is an issue we deal with more fully in a later chapter.

Student language score by teaching method and IQ...

Interpreting the output

```
> summary(TeachIQmethod.fit3)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  51.96345    8.24637   6.301 1.14e-06 ***
IQ            0.31564    0.07341   4.299 0.000213 ***
method1      -9.87793    2.82068  -3.502 0.001688 **
method3     -24.03715    2.77910  -8.649 3.97e-09 ***
---
```

```
> summary(TeachIQmethod.fit2)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  42.08552    8.73921   4.816 5.47e-05 ***
IQ            0.31564    0.07341   4.299 0.000213 ***
method2       9.87793    2.82068   3.502 0.001688 **
method3     -14.15922    2.85240  -4.964 3.70e-05 ***
---
```


Student language score by teaching method and IQ...

Interpreting the output...

Note that the `method1` coefficient in `fit3` has the same magnitude, but opposite sign to `method2` in `fit2`. This is because `fit3` measures how method 1 differs form method 2 whereas `fit2` measures how method 2 differs from method 1.

Note also that the original missing 'contrast', method 2 vs method 3, is now available in `fit3`. We see that method 3 is markedly worse than method 2.

Student language score by teaching method and IQ...

Interpreting the output...

Let us put confidence bounds on our effects.

```
> ## Baseline method here is method1.
> confint(TeachIQmethod.fit2)
                 2.5 %     97.5 %
(Intercept)  24.1218063 60.0492251
IQ            0.1647361  0.4665482
method2       4.0799363 15.6759248
method3     -20.0224212 -8.2960209
```

```
> ## Baseline method here is method2.
> confint(TeachIQmethod.fit3)
                 2.5 %      97.5 %
(Intercept)  35.0127936  68.9140989
IQ            0.1647361   0.4665482
method1     -15.6759248  -4.0799363
method3     -29.7496781 -18.3246250
```


Student language score by teaching method and IQ...

An Executive Summary

The best teaching method is method 2 regardless of the student's IQ.

With 95% confidence, we can make the following statements:

1. For students experiencing the same teaching method, we estimate that the mean language test score increases by between 1.6 and 4.7 marks for each additional ten[7] IQ points.
2. For students with the same IQ, we estimate that the mean language test score...
   - ▷ for students taught using method 2 is between 4.1 and 15.7 marks higher than those taught using method 1,
   - ▷ for students taught using method 3 is between 8.3 and 20 marks lower than those taught using method 1,
   - ▷ for students taught using method 3 is between 18.3 and 29.7 marks lower than those taught using method 2.

[7] This is the CI for $\beta_1 \times 10$ to demonstrate stating it on a different scale.

Section 9.4
Closing remarks and relevant R-code.

Most of the R-code you need for this chapter

When your response variable can be explained by a categorical variable and a numeric variable then you can use both explanatory variables in your analysis to explain y.

You do not need to create indicator variables for levels of the categorical variable since R does this for you. It will choose the reference level for you, so be careful. You can change this if needed. For example:

```
> teach.df$method = relevel(teach.df$method, ref = "2")
```


Most of the R-code you need for this chapter...

Follow the following steps:

Gain some intuition from looking at a suitable scatter plot of the data – this could suggest whether the response should be logged, etc.

Then, fit the model with interaction term,

```
> TeachIQmethod.fit=lm(lang~IQ*method, data=teach.df)
```

Then, assuming no reason to question the independence assumption, do our usual EOV, normality and influence checks.

If checks are OK, then see whether you really need a separate slope for each level of your factor variable. We can use `summary` if the factor variable has two levels, otherwise we must use `anova`

```
> data.frame(anova(TeachIQmethod.fit))
```


Most of the R-code you need for this chapter...

If the interaction term is not significant (large P-value for the ':' term) then apply Occam's razor by fitter the simpler main effects model.

To fit the simpler 'main effects' (parallel lines) model:

```
> TeachIQmethod.fit2=lm(lang~IQ+method, data=teach.df)
> summary(TeachIQmethod.fit2)
```

Note that sometimes we may see that we don't need one or more of the explanatory variables at all – do this by checking for a large P-value on each term. Apply Occam's razor accordingly.

Cautionary remark about anova

When using anova to test a joint hypothesis for a multi-level explanatory factor, that factor **must be the bottom term** in the anova output.[8] That is, immediatately above the Residuals line.

By way of example, suppose we change the order of the explanatory variables when we fit the model using lm,

```
> TeachIQmethod.fit2b=lm(lang~method+IQ, data=teach.df)
> anova(TeachIQmethod.fit2b)
```

```
          Df Sum Sq Mean Sq F value    Pr(>F)
method     2 3194.6 1597.30  41.490 8.112e-09 ***
IQ         1  711.6  711.65  18.485 0.0002134 ***
Residuals 26 1001.0   38.50
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In this case, the change in *P*-values (see slide 23) makes no difference to conclusions , but this may not always be the case.

[8] Unlike the summary table, the *P*-values for the terms in the anova output depend on the terms above it.

Case Study 9.1: Language score vs teaching method and student IQ

Tou Ohone Andate - staff number 1234567

Problem

Educational experts were interested in which of three different teaching methods was most effective in increasing a student-tested language score for children of a range of abilities—as measured by IQ. Moreover, they wanted to know if the relative effectiveness of the methods differed according to IQ.

An experiment was conducted whereby 30 students were randomly allocated into three groups and each group was taught using a different teaching method. This randomisation was done to ensure that a range of student abilities was represented in each group. As students were in a test environment we can assume that their test scores are independent of each other.

The variables of interest were:

- `lang`: The student's language score.
- `IQ`: The student's IQ before the teaching programme began.
- `method`: The type of teaching method used on the student.

Question of Interest

We wish to see if the language score achieved depended on the teaching method. We want to check for any confounding effect of IQ.

Read in and Inspect the Data

```
data(teach.df)
head(teach.df)
```

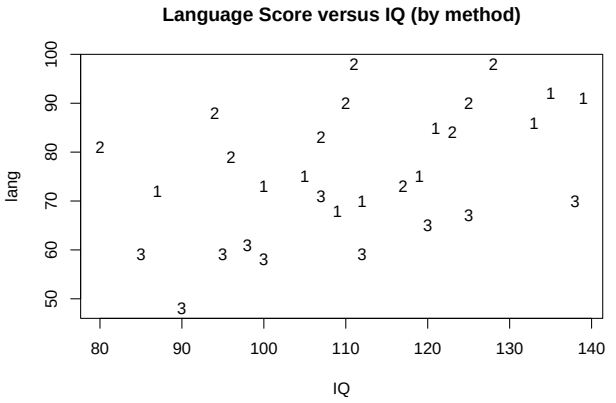
```
##   lang  IQ method
## 1   72  87      1
## 2   75 119      1
## 3   85 121      1
## 4   70 112      1
## 5   73 100      1
## 6   86 133      1
```

```
str(teach.df)
```

```
## 'data.frame':    30 obs. of  3 variables:
##  $ lang  : int  72 75 85 70 73 86 92 68 91 75 ...
##  $ IQ    : int  87 119 121 112 100 133 135 109 139 105 ...
##  $ method: int  1 1 1 1 1 1 1 1 1 1 ...
```



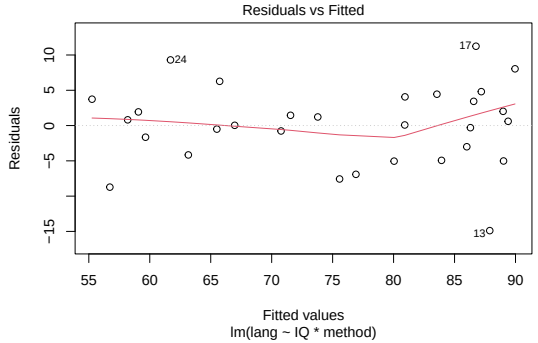
```
# We need to convert method into a factor variable
teach.df$method = factor(teach.df$method)
plot(lang ~ IQ, main = "Language Score versus IQ (by method)",
    pch = as.character(teach.df$method), data = teach.df)
```



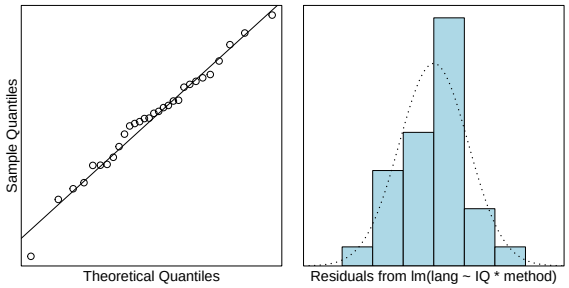
Looking at the coded scatter plot, we can see three parallel lines. It appears that the score is increasing with IQ and that method 2 is scoring highest and method 3 is scoring lowest. The variability around these individual lines is much lower than the variability seen in the separate plots.

Model Building and Check Assumptions

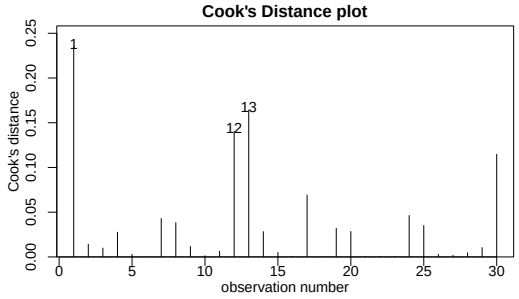
```
teach.fit = lm(lang ~ IQ * method, data = teach.df)
plot(teach.fit, which = 1)
```




```
normcheck(teach.fit)
```



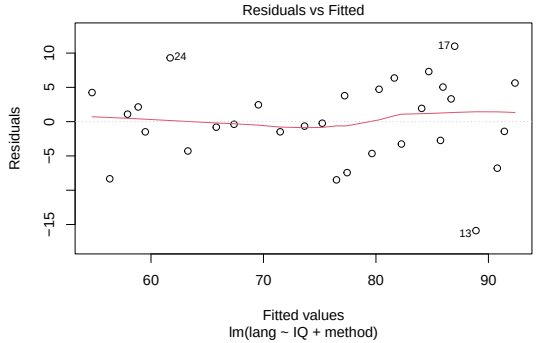

```
cooks20x(teach.fit)
```

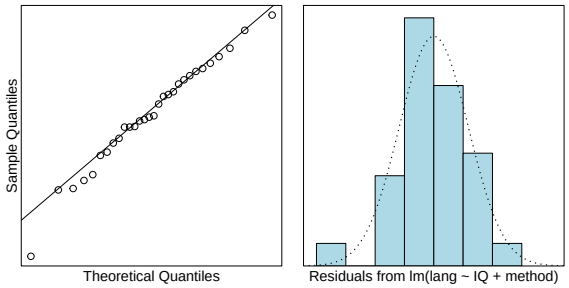

```
anova(teach.fit)
```

```
## Analysis of Variance Table
##
## Response: lang
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## IQ         1 1004.42 1004.42 26.1416 3.124e-05 ***
## method     2 2901.83 1450.91 37.7625 3.867e-08 ***
## IQ:method  2   78.82   39.41  1.0257    0.3737
## Residuals 24  922.13   38.42
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

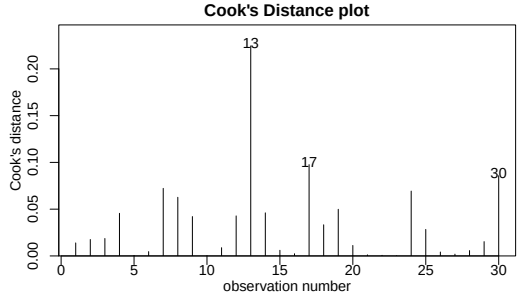
```
teach.fit2 = lm(lang ~ IQ + method, data = teach.df)
plot(teach.fit2, which = 1)
```




```
normcheck(teach.fit2)
```




```
cooks20x(teach.fit2)
```


```
anova(teach.fit2)
```

```
## Analysis of Variance Table
##
## Response: lang
##           Df Sum Sq Mean Sq F value    Pr(>F)
## IQ         1 1004.4  1004.4  26.090 2.529e-05 ***
## method     2 2901.8  1450.9  37.688 2.077e-08 ***
## Residuals 26 1001.0    38.5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(teach.fit2)
```

```
##
## Call:
## lm(formula = lang ~ IQ + method, data = teach.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.8936  -3.1331  -0.3047   4.1294  11.0003
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  42.08552    8.73921   4.816 5.47e-05 ***
## IQ            0.31564    0.07341   4.299 0.000213 ***
## method2       9.87793    2.82068   3.502 0.001688 **
## method3     -14.15922    2.85240  -4.964 3.70e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.205 on 26 degrees of freedom
## Multiple R-squared:  0.796,  Adjusted R-squared:  0.7725
```

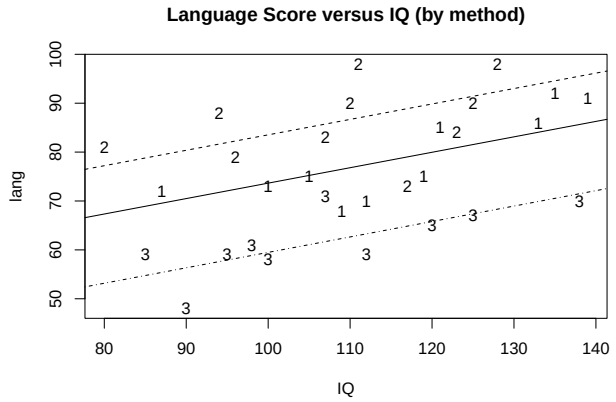
```
## F-statistic: 33.82 on 3 and 26 DF,  p-value: 3.986e-09
```

```
confint(teach.fit2)
```

```
##                   2.5 %     97.5 %
## (Intercept)  24.1218063 60.0492251
## IQ            0.1647361  0.4665482
## method2       4.0799363 15.6759248
## method3     -20.0224212 -8.2960209
```

Visualise the Final Model

```
plot(lang ~ IQ, main = "Language Score versus IQ (by method)",
    pch = as.character(teach.df$method), data = teach.df)
abline(teach.fit2$coef[1], teach.fit2$coef[2], lty = 1)
abline(teach.fit2$coef[1] + teach.fit2$coef[3], teach.fit2$coef[2], lty = 2)
abline(teach.fit2$coef[1] + teach.fit2$coef[4], teach.fit2$coef[2], lty = 4)
```



Generate Model Output for when `method`'s baseline is "2"

```
teach.df$method = relevel(teach.df$method, ref = "2")
teach.fit3 = lm(lang ~ IQ + method, data = teach.df)
confint(teach.fit3)
```

```
##                   2.5 %      97.5 %
## (Intercept)  35.0127936  68.9140989
```



```
## IQ            0.1647361   0.4665482
## method1     -15.6759248  -4.0799363
## method3     -29.7496781 -18.3246250
```

Method and Assumption Checks

To explain language score, we first fitted the model with explanatory variables teaching method, IQ, and their interaction. But, the interaction term was not significant ($P\text{-}value = 0.37$). The model was refitted with the interaction term removed.

All model assumptions were satified. [*Optional:* The students should be acting independent of each other as they were randomly allocated to the method taught and they students are measured under test conditions.]

Our final model is

$$lang_i = \beta_0 + \beta_1 \times IQ_i + \beta_2 \times method.method2_i + \beta_3 \times method.method3_i + \epsilon_i,$$

where:

- $method.method2_i$ is set to one if student i received method 2, otherwise it is zero,
- $method.method3_i$ is set to one if student i received method 3, otherwise it is zero,
- and $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Here method 1 is our baseline.

The final model was also refitted with method 2 as the baseline. **Note:** When we change the baseline (to level 2), the values of the dummy variables switch, so that $method.method2_i$ becomes $method.method1_i$. Hence, $method.method1_i$ is set to one if student i received method 1, otherwise it is zero.

Our model explains almost 80% of the variation in language score.

Executive Summary

We were interested in comparing the effectiveness of three teaching methods on language scores acheived by students. We also wanted to see how this was effected by students IQ's.

We found that the effects of the teaching methods are the same regardless of IQ and the effect of IQ is the same regardless of teaching method.

In particular teaching method 2 is significantly better than the other two methods. Also, both methods 1 and 2 are significantly better than method 3.

Not surprisingly, students with higher IQ tended to score higher.

With 95% confidence:

- For students experiencing the same teaching method, we estimate that the expected language test score increases by between 1.6 and 4.7 marks for each additional 10 IQ points,
- For students with the same IQ, we estimate that the expected language test score for students taught using method 2 is between 4.1 and 15.7 marks higher than those taught using method 1,
- For students with the same IQ, we estimate that the expected language test score for students taught using method 1 is between 8.3 and 20.0 marks higher than those taught using method 3,
- For students with the same IQ, we estimate that the expected language test score for students taught using method 3 is between 18.3 and 29.7 marks lower than those taught using method 2.

What happens if we don't adjust for IQ?

We expect the confidence intervals for the methods to get wider because we have to "absorb" the extra variation not explained by IQ.

```
teach.df$method = relevel(teach.df$method, ref = "1")
teach.fit5 = lm(lang ~ method, data = teach.df)
teach.df$method = relevel(teach.df$method, ref = "2")
teach.fit6 = lm(lang ~ method, data = teach.df)
ci = confint(teach.fit5)
ci2 = confint(teach.fit6)
r2 = round(100*summary(teach.fit5)$r.squared)
```

With 95% confidence :

- For students taught using method 2 is between 0.4 and 15 marks higher than those taught using method 1,
- For students taught using method 3 is between 17.4 and 32 marks lower than those taught using method 1,
- For students taught using method 3 is between 9.7 and 24.3 marks lower than those taught using method 2.

Our model explains almost 65% of the variation in language score. You should be able to see that every one of these intervals is wider than before.

Multiple comparisions adjustment (Chap 11)

This will only make sense after Chapter 11:

```
require(emmeans)
emmeans(teach.fit2, specs = pairwise ~ method, infer=T)$contrasts
```

```
##  contrast estimate   SE df lower.CL upper.CL t.ratio p.value
##  1 - 2       -9.88 2.82 26   -16.89    -2.87  -3.502  0.0046
##  1 - 3       14.16 2.85 26     7.07    21.25   4.964  0.0001
##  2 - 3       24.04 2.78 26    17.13    30.94   8.649  <.0001
##
## 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
```

```
#Compare to
cbind(coef(summary(teach.fit2)),confint(teach.fit2))
```

```
##             Estimate Std. Error t value  Pr(>|t|)    2.5 %  97.5 %
## (Intercept)  42.0855    8.73921   4.816 5.466e-05  24.1218 60.0492
## IQ            0.3156    0.07341   4.299 2.134e-04   0.1647  0.4665
## method2       9.8779    2.82068   3.502 1.688e-03   4.0799 15.6759
## method3     -14.1592    2.85240  -4.964 3.696e-05 -20.0224 -8.2960
```



```
cbind(coef(summary(teach.fit3)),confint(teach.fit3))
```

```
##             Estimate Std. Error t value  Pr(>|t|)    2.5 %   97.5 %
## (Intercept)  51.9634    8.24637   6.301 1.137e-06  35.0128  68.9141
## IQ            0.3156    0.07341   4.299 2.134e-04   0.1647   0.4665
## method1      -9.8779    2.82068  -3.502 1.688e-03 -15.6759  -4.0799
## method3     -24.0372    2.77910  -8.649 3.971e-09 -29.7497 -18.3246
```

Case Study 9.2: Thyroid drug experiment

Tou Ohone Andate - staff number 1234567

Problem

Hypothyroidism; (from hypo- meaning under or reduced, plus thyroid), often called under-active thyroid or low thyroid and sometimes hypothyreosis, is a common endocrine disorder in which the thyroid gland does not produce enough thyroid hormone. It can cause a number of symptoms, such as tiredness, poor ability to tolerate cold, and weight gain. In children, hypothyroidism leads to delays in growth and intellectual development, which is called cretinism in severe cases[1].

Drug companies attempt to develop drugs to help to grow the thyroid gland. Before these drugs are tested on humans they are tested on mice as they share a lot of physical characteristics of humans.

With the above in mind, an experiment was conducted to assess the effect of a newly developed drug that was intended to increase the weight of the thyroid gland (in mg).

Sixteen laboratory animals (mice) were randomly allocated into 2 groups:

1. The control group (1) received water orally for 7 days.
2. The drug group (2) received the new drug orally for 7 days.

The body weight of each mouse (g) was also recorded, since it was felt that it might explain variability in thyroid weight.

The variables of interest were:

- `thyroid`: The weight of the mouse's thyroid gland.
- `body`: The body weight of the mouse (g).
- `group`: A two-level factor with levels "1" and "2"

Question of Interest

To assess the effect of a newly developed drug that was intended to increase the weight of the thyroid gland (in mg).

Read in and Inspect the Data

```
thyroid.df = read.table("Thyroid.txt", header = TRUE)
head(thyroid.df)
```

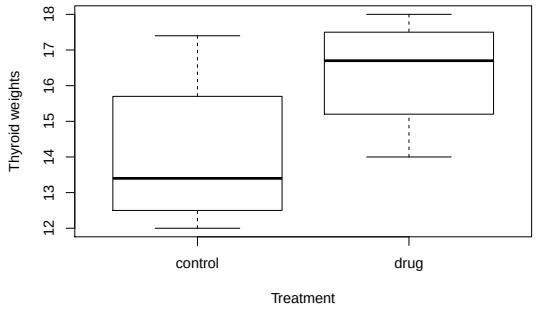
```
##   thyroid body group
## 1    16.0  203     1
## 2    13.0  180     1
## 3    12.0  183     1
## 4    15.4  191     1
## 5    17.4  204     1
## 6    13.0  178     1
```

```
str(thyroid.df)
```

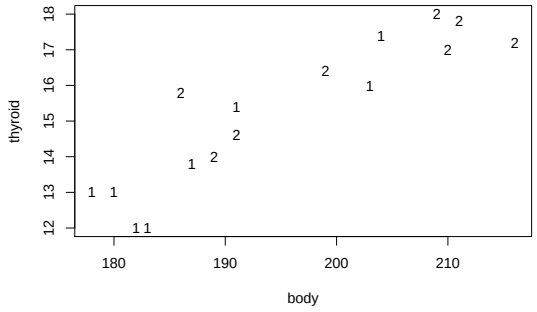
[1]Shamelessly stolen form Wiki: http://en.wikipedia.org/wiki/Hypothyroidism


```
## 'data.frame':    16 obs. of  3 variables:
##  $ thyroid: num  16 13 12 15.4 17.4 13 12 13.8 17 16.4 ...
##  $ body   : int  203 180 183 191 204 178 182 187 210 199 ...
##  $ group  : int  1 1 1 1 1 1 1 1 2 2 ...
```

```
# Create a factor variable for the drug vs control
thyroid.df$trt = with(thyroid.df, factor(ifelse(group == 1, "control", "drug")))
plot(thyroid ~ trt, data = thyroid.df, xlab = "Treatment", ylab = "Thyroid weights")
```




```
plot(thyroid ~ body, type = "n", data = thyroid.df)
text(thyroid.df$body, thyroid.df$thyroid, thyroid.df$group)
```



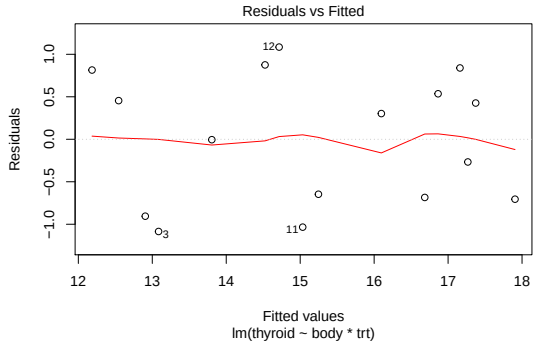
In the first plot it looks like there is an effect due to the treatment. Should we worry about the innate differences between animals? That is, do we expect larger mice to have larger thyroids?

In the second plot it looks like we should indeed adjust for the differences in body weight. Let's fit an

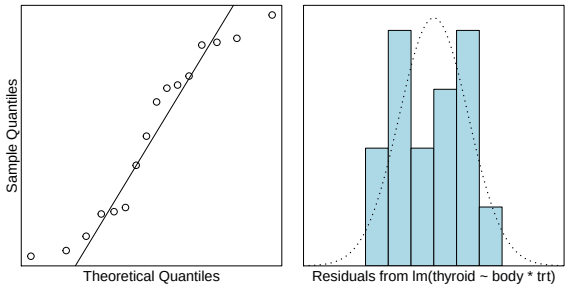
interaction model.

Model Building and Check Assumptions

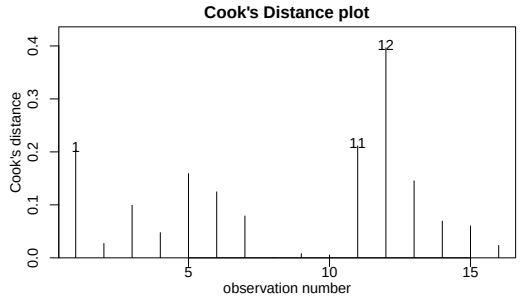
```
thyroid.fit = lm(thyroid ~ body * trt, data = thyroid.df)
plot(thyroid.fit, which = 1)
```




```
normcheck(thyroid.fit)
```




```
cooks20x(thyroid.fit)
```

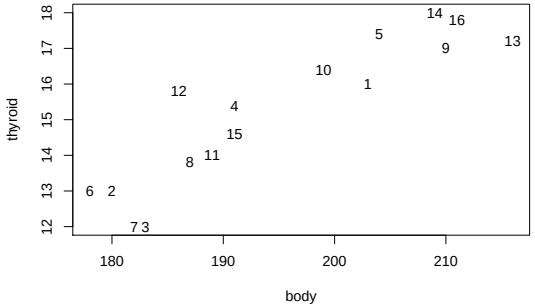



```
cooks.distance(thyroid.fit)[12]
```

```
##        12
## 0.3963979
```

Is Observation 12 Influential?

```
plot(thyroid ~ body, type = "n", data = thyroid.df)
text(thyroid.df$body, thyroid.df$thyroid, 1:16)
```



Model Building and Check Assumptions Continued...

```
summary(thyroid.fit)
```

```
##
## Call:
## lm(formula = thyroid ~ body * trt, data = thyroid.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0852 -0.6897  0.1487  0.6052  1.0848
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -19.84972    5.97937  -3.320 0.006114 **
## body           0.17997    0.03168   5.681 0.000102 ***
## trtdrug       14.78742    8.16681   1.811 0.095283 .
## body:trtdrug  -0.07364    0.04201  -1.753 0.105068
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8465 on 12 degrees of freedom
## Multiple R-squared:  0.8637, Adjusted R-squared:  0.8296
## F-statistic: 25.34 on 3 and 12 DF,  p-value: 1.77e-05
```

```
thyroid.fit2 = lm(thyroid ~ body + trt, data = thyroid.df)
summary(thyroid.fit2)
```

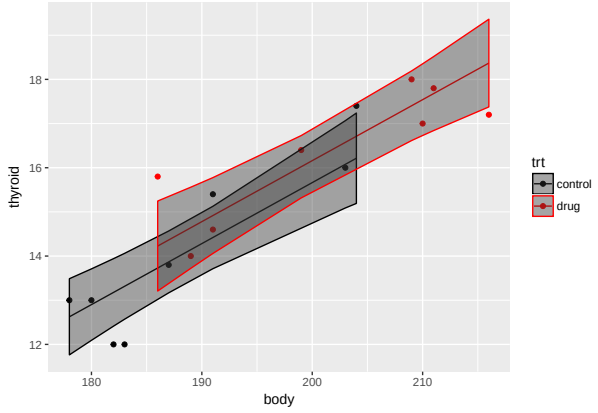
```
##
## Call:
## lm(formula = thyroid ~ body + trt, data = thyroid.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.31554 -0.56603  0.01542  0.43274  1.57304
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -11.9538     4.2349  -2.823   0.0144 *
## body          0.1381     0.0224   6.164 3.41e-05 ***
## trtdrug       0.4972     0.5394   0.922   0.3734
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9116 on 13 degrees of freedom
## Multiple R-squared:  0.8287, Adjusted R-squared:  0.8024
## F-statistic: 31.46 on 2 and 13 DF,  p-value: 1.044e-05
```

Is `trt` really insignifcant?

```
b = coef(thyroid.fit2)
# Note this won't work unless you have ggplot2 installed
library(ggplot2)
```



```
thyroid.df = cbind(thyroid.df, predict(thyroid.fit2, interval = "confidence"))
ggplot(thyroid.df, aes(x = body, y = thyroid, color = trt)) + geom_point() +
      geom_line(aes(y = fit)) + geom_ribbon(aes(ymin = lwr, ymax = upr), alpha = 0.4) +
      scale_color_manual(values = c("#000000", "#FF0000"))
```



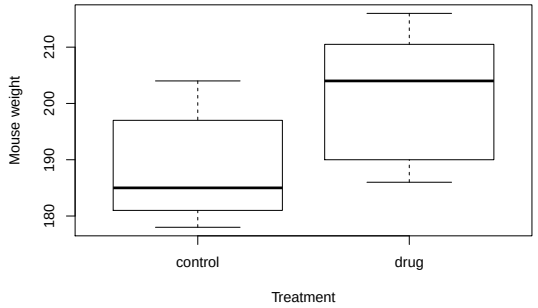
Model Building and Check Assumptions Continued...

```
thyroid.fit3 = lm(thyroid ~ body, data = thyroid.df)
summary(thyroid.fit3)
```

```
##
## Call:
## lm(formula = thyroid ~ body, data = thyroid.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.43232 -0.42579 -0.00513  0.60886  1.92031
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.85763    3.67718  -3.769  0.00208 **
## body          0.14913    0.01883   7.921 1.54e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9066 on 14 degrees of freedom
## Multiple R-squared:  0.8176, Adjusted R-squared:  0.8045
## F-statistic: 62.74 on 1 and 14 DF,  p-value: 1.538e-06
```

Why was `trt` insignficant?

```
plot(body ~ trt, data = thyroid.df, main = "", xlab = "Treatment", ylab = "Mouse weight")
```



We can see that heavier mice have heavier thyroids, and the mice in the drug group are heavier.

Methods and Assumption Checks

The exploratory plots suggested to explain thyroid weight in mice, we first fitted the linear model with explanatory variables drug treatment and body weight, and their interaction. But, the interaction term was not significant ($P\text{-}value = 0.11$). The model was refitted with the interaction term removed. However, treatment was not signficant ($P\text{-}value = 0.37$), and was removed.

Mice were randomly allocated to groups and the measurements were done independently of each other. All other model assumptions were satisfied. (One potentially influential observation was investigated, however given its location and the stupidly small size of this data set (16 obs.) we will overlook it.)

Our final model is

$$thyroid_i = \beta_0 + \beta_1 \times weight_i + \epsilon_i,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Our model explains about 82% of the variability in the weight of thyroid glands.

Executive Summary

We wanted to assess the effect of a newly developed drug that was intended to increase the weight of the thyroid gland (in mg).

The only thing we can report is that larger mice have larger thyroids. We could report on the size of this effect, but won't since it wasn't part of the research question.

We have no evidence to believe that this drug increases the expected size of the thyroid gland in mice.

Addendum

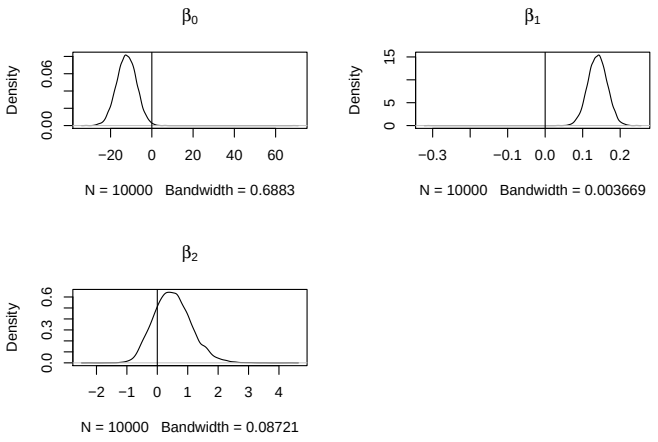
From the initial boxplot it appeared that mice in the drug group had noticeably higher thyroid weight than those in the control group. However, it turned out that the drug group had bigger mice! The randomisation has, unfortunately, failed us here. Usually randomisation works out—but don't assume it always does for smaller sample sizes like this.

In the future it may be best to put 2 mice into each of 8 weight classes and randomly allocate (via a coin toss) one of these two mice into the treatment group or not.

Bootstrapping

We can bootstrap to avoid the normality assumption. The vertical line is where the coeffcient is zero. If it's towards the middle of the plot then that is evidence for the null—i.e. that there is no treatment effect.

```
set.seed(123)
# Note this won't work unless you have bootstrap installed
library(bootstrap)
theta = function(rows, data = thyroid.df) {
  fit = lm(thyroid ~ body + trt, data = data[rows, ])
  coef(fit)
}
b = bootstrap(1:nrow(thyroid.df), 10000, theta)$thetastar
par(mfrow = c(2,2))
plot(density(b[1, ]), expression(beta[0]))
abline(v = 0)
plot(density(b[2, ]), expression(beta[1]))
abline(v = 0)
plot(density(b[3, ]), expression(beta[2]))
abline(v = 0)
```



Case Study 9.3: Water Hardness and Mortality

James Curran

The data in this case study were collected in an investigation of environmental causes of disease. They show the annual mortality rate per 100,000 for males, averaged over the years 1958–1964, and the calcium concentration (in parts per million) in the drinking water supply for 61 large towns in England and Wales. (The higher the calcium concentration, the harder the water.) Towns at least as far north as Derby are identified in the data set with the code `N`. In this study we will use R to investigate how are mortality and water hardness related, and if there a geographical factor in the relationship. The data is in the file `water.csv` on Canvas

```
water.df = read.csv("WATER.csv")
head(water.df)
```

```
##   Mortality  Ca Location
## 1      1247 105
## 2      1668  17        N
## 3      1466   5
## 4      1800  14        N
## 5      1609  18        N
## 6      1558  10        N
```

It's always useful to check for missing values.

```
sum(is.na(water.df))
```

```
## [1] 0
```

No missing values. All good. How about a plot? The ideal plot would use `Location` as a plotting symbol, however we can see that the towns to the South are coded as blanks. We should change that. How? How about making all the values that are not `N` be `S`?

```
water.df$Location[water.df$Location != 'N'] = 'S'
```

```
## Warning in `[<-.factor`(`*tmp*`, water.df$Location != "N", value =
## structure(c(NA, : invalid factor level, NA generated
```

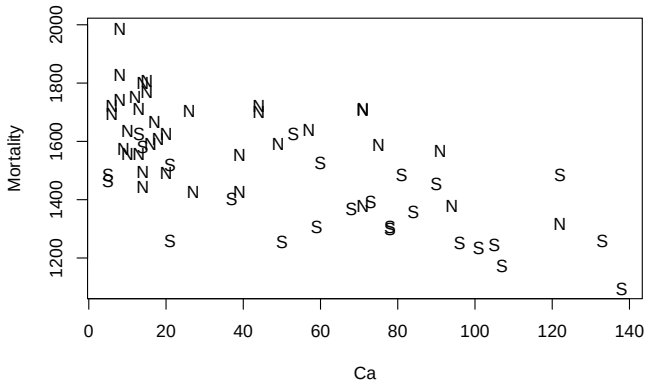
Hmm. That did not work, as planned. That is because `Location` is a factor. We have a number of choices. One is to make a new variable. The other is to make `Location` a character vector, make the change and, then make it a factor again. Although this second option sounds like a lot of work it is really only one line of code. It is preferable because we will not clutter our workspace with redundant information.

```
water.df$Location = with(water.df, {
  Location = as.character(Location);
  Location[is.na(Location)] = 'S';
  Location = factor(Location)
})
water.df$Location
```

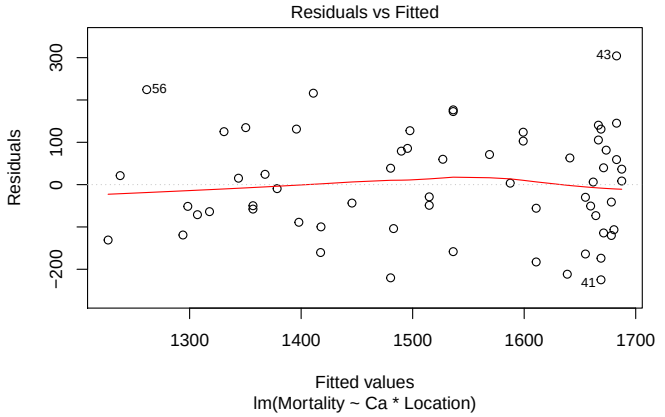
```
##  [1] S N S N N N N S N S S S N S S N N N N S N N N N S N S N S S S N N S
## [36] S S N S N N N N N S S N N N N N S N S S S N N N S N
## Levels: N S
```

Much better. Now how about that plot?

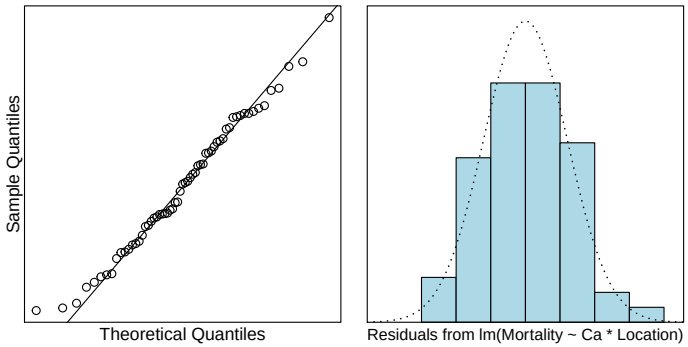
```
plot(Mortality~Ca, pch = as.character(water.df$Location), data = water.df)
```

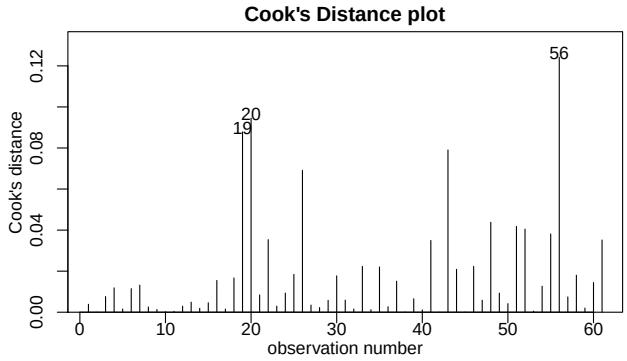

```
water.fit = lm(Mortality ~ Ca * Location, data = water.df)
plot(water.fit, which = 1)
```




```
normcheck(water.fit)
```




```
cooks20x(water.fit)
```



All looks pretty good.

```
summary(water.fit)
```

```
##
## Call:
```



```
## lm(formula = Mortality ~ Ca * Location, data = water.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -224.878  -88.953    3.495   85.617  304.172
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1701.4272    30.9493  54.975  < 2e-16 ***
## Ca             -2.3249     0.6996  -3.323  0.00156 **
## LocationS    -175.7321    58.5586  -3.001  0.00399 **
## Ca:LocationS    0.1598     0.9460   0.169  0.86643
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 123.9 on 57 degrees of freedom
## Multiple R-squared:  0.5857, Adjusted R-squared:  0.5639
## F-statistic: 26.86 on 3 and 57 DF,  p-value: 5.855e-11
```

It does not look like there is any evidence of different slopes. Do we get any additional info from the ANOVA table?

```
anova(water.fit)
```

```
## Analysis of Variance Table
##
## Response: Mortality
##             Df Sum Sq Mean Sq F value    Pr(>F)
## Ca           1 906185  906185 59.0005 2.330e-10 ***
## Location     1 331091  331091 21.5569 2.065e-05 ***
## Ca:Location  1    438     438  0.0285    0.8664
## Residuals   57 875459   15359
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Nope. We only have two levels so we do not need the ANOVA table for this analysis.

So we do not have interaction, but it looks like there is a relationship with hardness and it looks like there is a North/South effect (`Location`). We should refit the model.

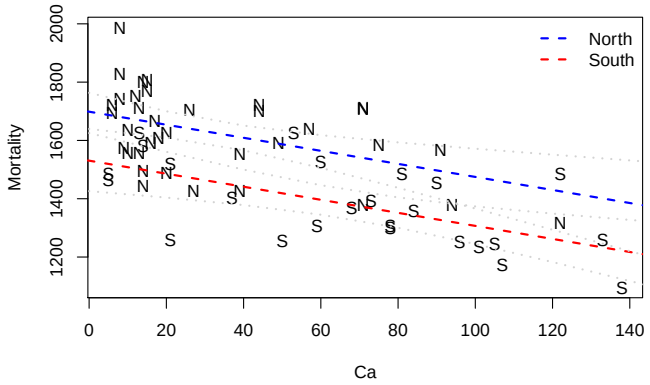
```
water.fit2 = lm(Mortality ~ Ca + Location, data = water.df)
summary(water.fit2)
```

```
##
## Call:
## lm(formula = Mortality ~ Ca + Location, data = water.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -223.607  -89.582    2.091   83.303  306.353
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1698.5472    25.6148  66.311  < 2e-16 ***
## Ca            -2.2375     0.4669  -4.792 1.19e-05 ***
## LocationS   -167.9518    35.8694  -4.682 1.75e-05 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 122.9 on 58 degrees of freedom
## Multiple R-squared:  0.5855, Adjusted R-squared:  0.5712
## F-statistic: 40.96 on 2 and 58 DF,  p-value: 8.091e-12
```

Minimal change in R^2. How do the fitted lines look on the plot?

```
b = coef(water.fit2)
plot(Mortality~Ca, pch = as.character(water.df$Location), data = water.df)
abline(b[1:2], col = "blue", lty = 2, lwd = 2)
abline(b[1] + b[3], b[2], col = "red", lty = 2, lwd = 2)
legend("topright", lty = 2, lwd = 2, col = c("blue", "red"),
    legend = c("North", "South"), bty = "n")
# This code puts some confidence bounds around the lines.
# It's pretty complicated if you don't know R and it isn't examinable
pred.df = data.frame(Ca = rep(seq(0, 160, by = 20), 2),
    Location = rep(c('N', 'S'), c(9, 9)))
water.pred = predict(water.fit, newdata = pred.df, interval = "confidence")
for(i in 1:2){
  idx = 1:9 + (i - 1) * 9
  for(j in c("lwr", "upr")){
      lines(pred.df$Ca[idx], water.pred[idx, j], lty = 3, lwd = 2, col = "lightgrey");
  }
}
```



So it looks like mortality decreases as the calcium concentration in the water increases, and that mortality is lower in the South than the North.

Case Study 9.4: Forced Expiratory Volume

James Curran

Problem

The data in this case study comes from a study which was interested in the relationship between smoking and Forced Expiratory Volume (FEV). FEV measures how much air a person can exhale during a forced breath. You can think of it as a measure of lung function—the higher the FEV value, the better your lungs work. The data in this study is a random sample of 654 youths, aged 3 to 19, in the area of East Boston during middle to late 1970's. The question of interest concerns the relationship between smoking and FEV.

The variables of interest are:

- `age`: age in years
- `fev`: FEV measurement
- `ht`: height in inches
- `sex`: 0 or 1 (we don't know which is male or female but we might be able to guess)
- `smoke`: 1 for smoker and 0 for non-smoker

Question of Interest

To quantify the relationship between smoking and FEV.

Read in and Inspect the Data

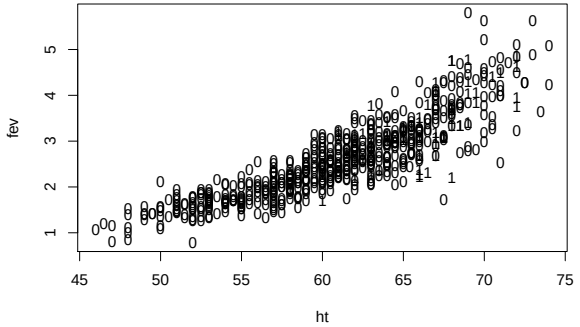
```
fev.df = read.csv("fev.csv")
head(fev.df)
```

```
##   age   fev   ht sex smoke
## 1   9 1.708 57.0   0     0
## 2   8 1.724 67.5   0     0
## 3   7 1.720 54.5   0     0
## 4   9 1.558 53.0   1     0
## 5   9 1.895 57.0   1     0
## 6   8 2.336 61.0   0     0
```

```
# Any missing values?
sum(is.na(fev.df))
```

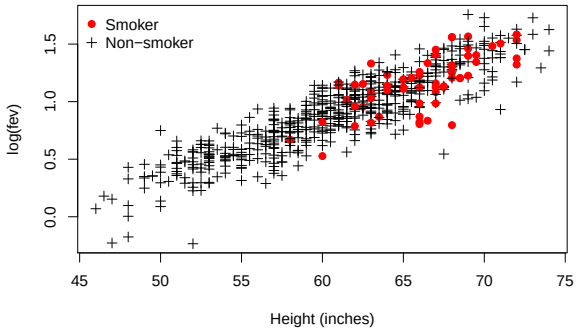
```
## [1] 0
```

```
plot(fev ~ ht, pch = as.character(smoke), data = fev.df)
```

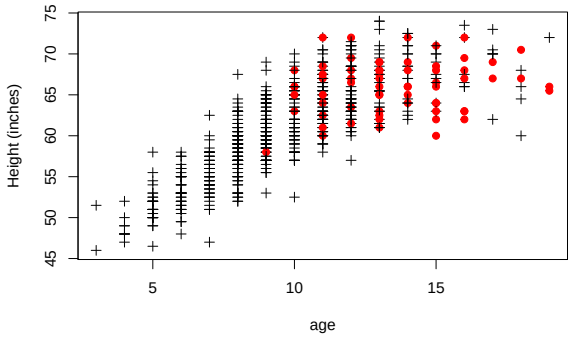
It's a little hard to see what is happening here. We can see that FEV gets more variable as the height increases. We should probably take logs. How about if we change visualising smokers from text to colours?

```
plot(log(fev) ~ ht, col = ifelse(smoke == 1, "red", "black"), pch = ifelse(smoke ==
    1, 19, 3), data = fev.df, xlab = "Height (inches)")
legend("topleft", col = c("red", "black"), pch = c(19, 3), legend = c("Smoker",
    "Non-smoker"), bty = "n")
```



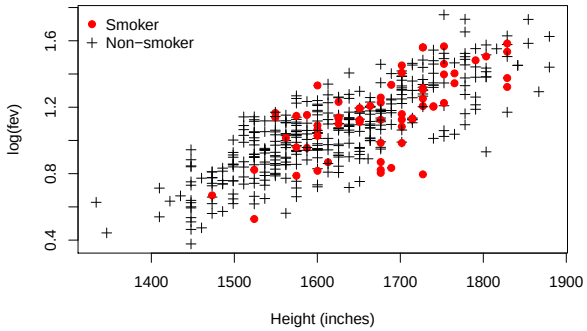
Taking logs has definitely helped, and it looks like there is a positive relationship between FEV and height. However, what also looks a little odd is that most of the smokers seem to be taller. What is going on?

```
plot(ht ~ age, col = ifelse(smoke == 1, "red", "black"), pch = ifelse(smoke ==
    1, 19, 3), data = fev.df, ylab = "Height (inches)")
```



Hmmm–sneaky sneaky. Looks like we've got quite a few very young people in this study. Let's remove everyone below 9 years old (under the youngest smoking case) and take another look at the data.

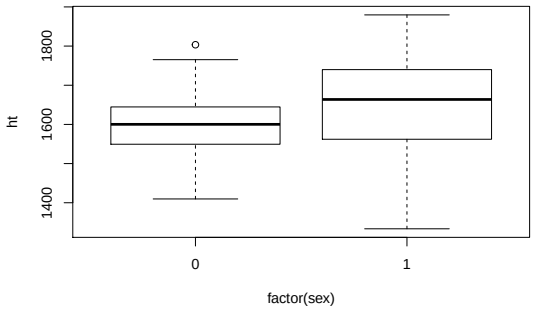
```
teens.df = subset(fev.df, age >= 9)
teens.df$ht = 25.4 * teens.df$ht  # Who wants height in inches. We want millimetres (mm)
plot(log(fev) ~ ht, col = ifelse(smoke == 1, "red", "black"), pch = ifelse(smoke ==
    1, 19, 3), data = teens.df, xlab = "Height (inches)")
legend("topleft", col = c("red", "black"), pch = c(19, 3), legend = c("Smoker",
    "Non-smoker"), bty = "n")
```



So is there any difference between the smokers and the non-smokers? It's a little hard to tell, but we should fit an interaction model to find out. We're also going to include the effect of sex, so there is an extra intraction to be included. We said we might be able to guess which are the males and which are the females. How?

Possibly shorter?

```
plot(ht ~ factor(sex), data = teens.df)
```



Well it is marginal. We could assume that the females are smaller than the males, but remember girls often grow faster at this age than boys, so that might be a silly assumption. Does it matter? Not really at this point. Let's just leave it undetermined at this point in time. Because each of our factors only has two levels, we don't need to really worry about coding them as factors, however we will do it anyway.

```
teens.df$sex = factor(teens.df$sex)
teens.df$smoke = factor(ifelse(as.numeric(teens.df$smoke) == 1, "Yes", "No"),
    levels = c("Yes", "No"))
```

Let's have a quick look at the gender balance of smokers and non-smokers. `table` gives us a table of counts, `prop.table` turns it into row proportions (if we set `margin` to 1).

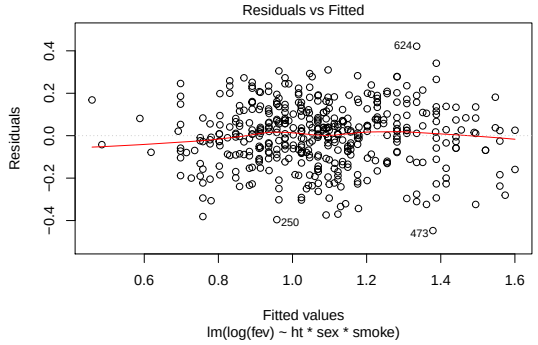
```
prop.table(table(teens.df$sex, teens.df$smoke), margin = 1)
```

```
##
##           Yes        No
##   0 0.1884058 0.8115942
##   1 0.1120690 0.8879310
```

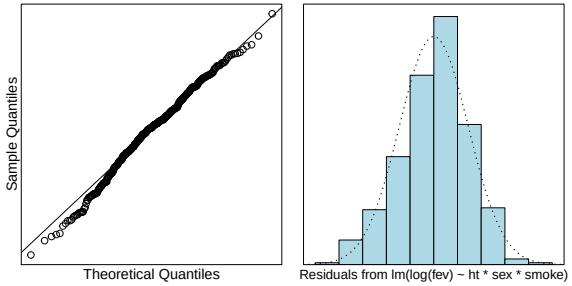
Looks okay.

Model Building and Check Assumptions

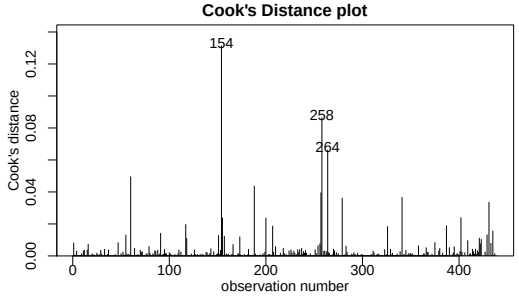
```
teens.fit = lm(log(fev) ~ ht * sex * smoke, data = teens.df)
plot(teens.fit, which = 1)
```




```
normcheck(teens.fit)
```




```
cooks20x(teens.fit)
```


```
anova(teens.fit)
```

```
## Analysis of Variance Table
##
## Response: log(fev)
##               Df  Sum Sq Mean Sq  F value    Pr(>F)
## ht             1 17.6734 17.6734 822.6036 < 2.2e-16 ***
## sex            1  0.0000  0.0000   0.0014  0.969741
## smoke          1  0.0103  0.0103   0.4776  0.489878
## ht:sex         1  0.2269  0.2269  10.5597  0.001246 **
## ht:smoke       1  0.0011  0.0011   0.0498  0.823463
## sex:smoke      1  0.0017  0.0017   0.0806  0.776635
## ht:sex:smoke   1  0.2082  0.2082   9.6893  0.001977 **
## Residuals    431  9.2599  0.0215
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(teens.fit)
```

```
##
## Call:
## lm(formula = log(fev) ~ ht * sex * smoke, data = teens.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44749 -0.08704  0.01096  0.09385  0.42172
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.0862107  0.6704649   0.129 0.897747
## ht             0.0006042  0.0004087   1.478 0.140035
## sex1          -3.5217144  0.9118061  -3.862 0.000130 ***
## smokeNo       -1.8549551  0.7161829  -2.590 0.009921 **
## ht:sex1        0.0021271  0.0005427   3.920 0.000103 ***
## ht:smokeNo     0.0011412  0.0004386   2.602 0.009586 **
```

```
## sex1:smokeNo     2.9629222  0.9576028   3.094 0.002103 **
## ht:sex1:smokeNo -0.0017828  0.0005727  -3.113 0.001977 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1466 on 431 degrees of freedom
## Multiple R-squared:  0.6618, Adjusted R-squared:  0.6563
## F-statistic: 120.5 on 7 and 431 DF,  p-value: < 2.2e-16
```

```
confint(teens.fit)
```

```
##                         2.5 %        97.5 %
## (Intercept)     -1.2315769198  1.4039983827
## ht              -0.0001990619  0.0014074005
## sex1            -5.3138540484 -1.7295747656
## smokeNo         -3.2626005864 -0.4473095401
## ht:sex1          0.0010604532  0.0031937824
## ht:smokeNo       0.0002791809  0.0020032276
## sex1:smokeNo     1.0807699274  4.8450744468
## ht:sex1:smokeNo -0.0029084423 -0.0006570752
```

```
100 * (exp(confint(teens.fit)) - 1)
```

```
##                         2.5 %        97.5 %
## (Intercept)     -70.81679810  3.071447e+02
## ht               -0.01990421  1.408391e-01
## sex1            -99.50770831 -8.226402e+01
## smokeNo         -96.17113046 -3.606540e+01
## ht:sex1           0.10610157  3.198888e-01
## ht:smokeNo        0.02792198  2.005235e-01
## sex1:smokeNo    194.69476136  1.261127e+04
## ht:sex1:smokeNo  -0.29042169 -6.568594e-02
```

Visualising the Final Model

Looking at the ANOVA table we can see that there is evidence of an interaction between height, sex, and smoking. Let's proceed with this model first (we might simplify it later). You might think this is really complicated, but it isn't. Each factor in our model has two levels, therefore there are four (2×2) combinations of the levels. These are:

1. sex = 0, smoke = "Yes"
2. sex = 0, smoke = "No"
3. sex = 1, smoke = "Yes"
4. sex = 1, smoke = "No"

So we can think of this as a model with four lines, each with different intercepts and slopes. To draw straight lines we only need two points—the start and the finish. The range of the heights is

```
range(teens.df$ht)
```

```
## [1] 1333.5 1879.6
```

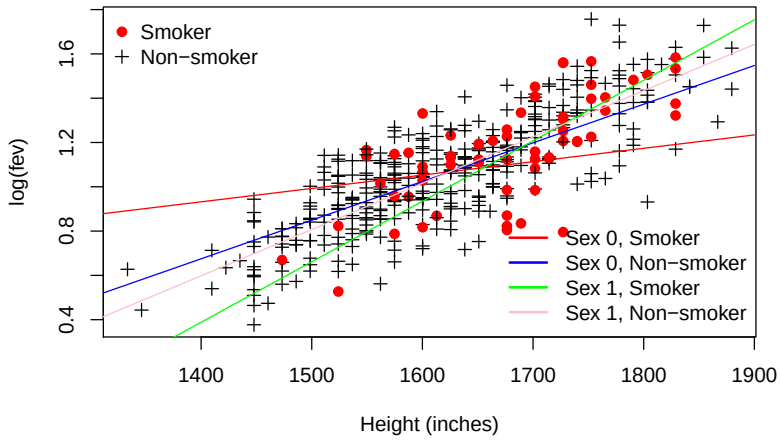
so let's build a data.frame that predicts at 1300 mm and 1900 mm, for each combination of the factors

```
pred.df = data.frame(ht = rep(c(1300, 1900), 4), sex = factor(rep(c(0, 1), c(4,
    4))), smoke = rep(c("Yes", "No"), c(2, 2)))
pred.df
```



```
##     ht sex smoke
## 1 1300   0   Yes
## 2 1900   0   Yes
## 3 1300   0    No
## 4 1900   0    No
## 5 1300   1   Yes
## 6 1900   1   Yes
## 7 1300   1    No
## 8 1900   1    No
```

```
teens.pred = predict(teens.fit, newdata = pred.df)
pred.df = cbind(pred.df, teens.pred)
plot(log(fev) ~ ht, col = ifelse(smoke == "Yes", "red", "black"), pch = ifelse(smoke ==
    "Yes", 19, 3), data = teens.df, xlab = "Height (inches)")
legend("topleft", col = c("red", "black"), pch = c(19, 3), legend = c("Smoker",
    "Non-smoker"), bty = "n")
lines(teens.pred ~ ht, data = pred.df[1:2, ], col = "red")
lines(teens.pred ~ ht, data = pred.df[3:4, ], col = "blue")
lines(teens.pred ~ ht, data = pred.df[5:6, ], col = "green")
lines(teens.pred ~ ht, data = pred.df[7:8, ], col = "pink")
legend("bottomright", col = c("red", "blue", "green", "pink"), lty = 1, lwd = 2,
    legend = c("Sex 0, Smoker", "Sex 0, Non-smoker", "Sex 1, Smoker", "Sex 1, Non-smoker"),
    bty = "n")
```



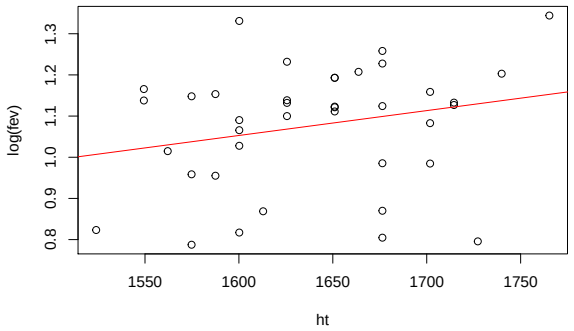
Looking at this plot, do we think we could get away with different lines? Probably not. To interpret this model we need to back-transform. However, this case study is probably complicated enough already :) What we can see is that the most striking difference is between the two genders in the smoking group. For simplicity,

let's assume that `Sex 0` is female, and `Sex 1` is male. If we have this coding, then we can see that the rate of increase in FEV (with height) is much more severely affected for female smokers, in that tall female smokers seem to have a lower rate of increase in FEV than the other groups. Let's just take a quick look at the group `Sex 0, Smoker`.

```
femaleSmokers.df = subset(teens.df, sex == 0 & smoke == "Yes")
nrow(femaleSmokers.df)
```

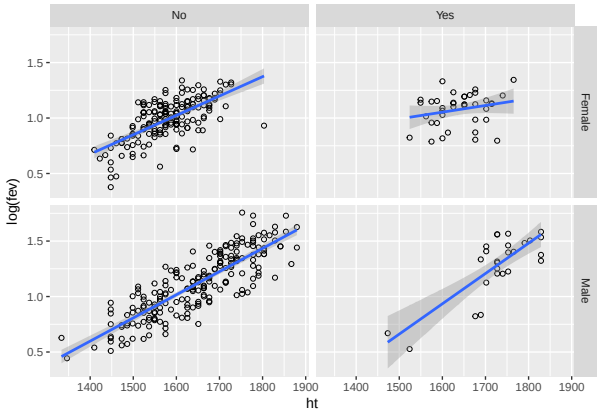
```
## [1] 39
```

```
plot(log(fev) ~ ht, data = femaleSmokers.df)
lines(teens.pred ~ ht, data = pred.df[1:2, ], col = "red")
```



What we're seeing here is that the smoking is making this line close to flat, regardless of height. It is probably this group alone that is driving the interaction between smoking, gender and height. We might also ask whether we really think males who smoke have a higher rate of increase in FEV?

```
teens.df$smoke = relevel(teens.df$smoke, ref = "No")
teens.df$sex = factor(ifelse(teens.df$sex == 0, "Female", "Male"))
library(ggplot2)
ggplot(data = teens.df, aes(x = ht, y = log(fev))) + geom_point(shape = 1) +
    facet_grid(sex ~ smoke) + stat_smooth(method = "lm")
```

I'm guessing probably not, and if you wanted to spend the time, you might find that the slope is the same as the other groups. It looks like there are two males who might be influencing our view of things.

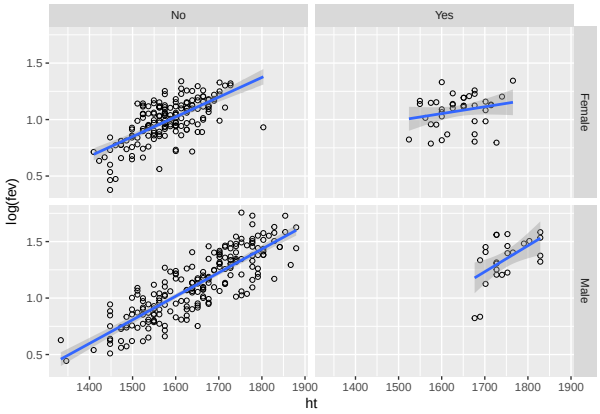
"The Hanging Chads"

Who are these guys? Well they're shorter than the rest, they're male and they smoke. Let's remove them and see what happens to the plot

```
outM = with(teens.df, which(sex == "Male" & smoke == "Yes" & log(fev) < 0.75))
outM
```

```
## [1]  60 154
```

```
ggplot(data = teens.df[-outM, ], aes(x = ht, y = log(fev))) + geom_point(shape = 1) +
    facet_grid(sex ~ smoke) + stat_smooth(method = "lm")
```


No
Yes
Female
Male
log(fev)
ht
1.5
1.0
0.5
1400
1500
1600
1700
1800
1900

Chapter 10: Multiple linear regression models

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

- Models with several explanatory variables
- Exploring pairwise relationships between all variables
- Multiple linear regression and the problem of multi-collinearity
- Fixing multi-collinearity
- Relevant R-code.

Section 10.1
Example: Modelling birth weights using several explanatory variables

Multiple explanatory variables

We have learned how to model the effects of numeric and/or factor explanatory variables using linear models.

More generally, we can (in principle) fit as many explanatory variables as we like. However, we shall see that this is not always a good idea.

Caution needs to be applied.

By way of example, let us examine which variables might explain the birth weight of babies.

Example: Birth weight of babies

`bwt`	birth weight in ounces (=28.35gm)
`gestation`	length of pregnancy in days
`not.first.born`	0=first born, 1=not first-born
`age`	mother's age in years
`height`	mother's height in inches
`weight`	mother's pre-pregnancy weight in pounds
`smoke`	smoking status of mother 0=not, 1=smoker.

The response variable is the baby's birth weight (`bwt`).

This dataset[1] was obtained from
http://www.stat.berkeley.edu/users/statlabs/labs.html.

The dataset has 1174 observations.

[1] It accompanies the excellent text Stat Labs: Mathematical Statistics through Applications Springer-Verlag (2001) by Deborah Nolan and Terry Speed.

Section 10.2
Exploring relationships between the variables

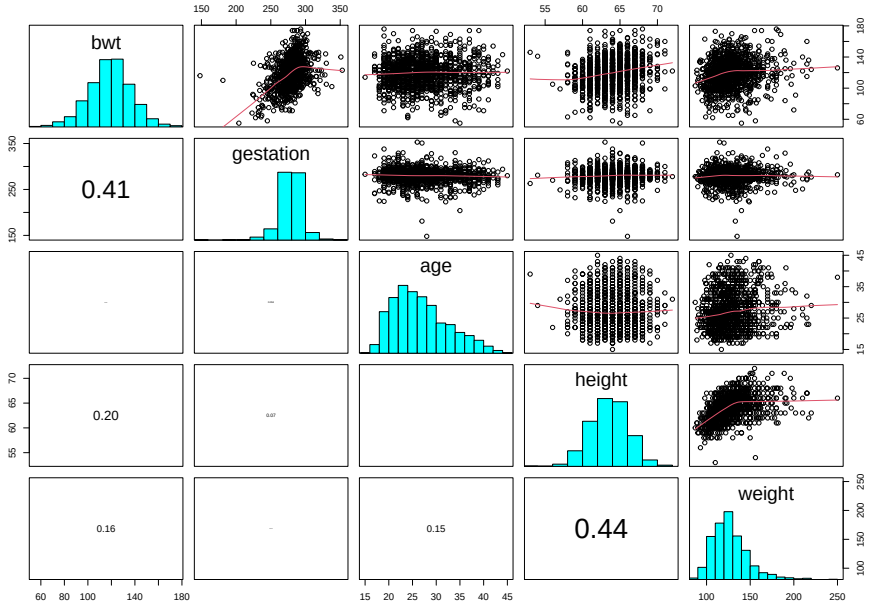
Birth weight of babies

Let us first inspect the relationships between the numerical explanatory variables and the response variable. The numeric explanatories are `gestation`, `age`, `height` and `weight`.

The five variables are in columns 1,2,4,5 and 6 in the data frame `Babies.df`.

```
> ## Invoke the s20x library
> library(s20x)
> ## Importing data into R
> Babies.df = read.table("Data/babies_data.txt", header=T)
> ## Create the pairs plot of the five numeric variables
> pairs20x(Babies.df[,c(1,2,4,5,6)])
```

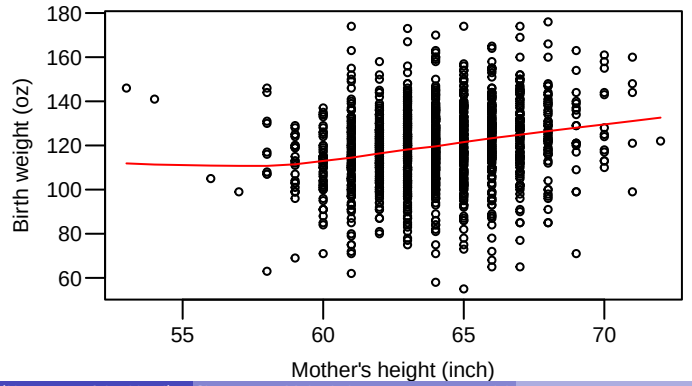

Birth weight of babies...



Birth weight of babies...

`pairs20x` gives a histogram of each variable in the diagonal cells. Above the diagonal, in the (i,j) cell ($i < j$) it gives scatter plots of variable i (y-axis) against variable j (x-axis). To illustrate, variable 1 is `bwt` and variable 4 is the mother's height (`height`). The scatter plot in cell (1,4) is

```
> plot(bwt ~ height,data = Babies.df,
+      xlab="Mother's height (inch)",ylab="Birth weight (oz)")
> lines(lowess(Babies.df$height,Babies.df$bwt))
```



Birth weight of babies...

The correlation coefficient between `height` and `bwt` is in cell (4,1). It is 0.20, indicating that a straight-line relationship is, at best, weak.

This correlation coefficient can only measure the strength of a straight line relationship between x (`height`) and a y (`bwt`). It can be useful but can, on occasion, be misleading. In other words, look at the scatter plot and use it only if the relationship looks like a straight line.

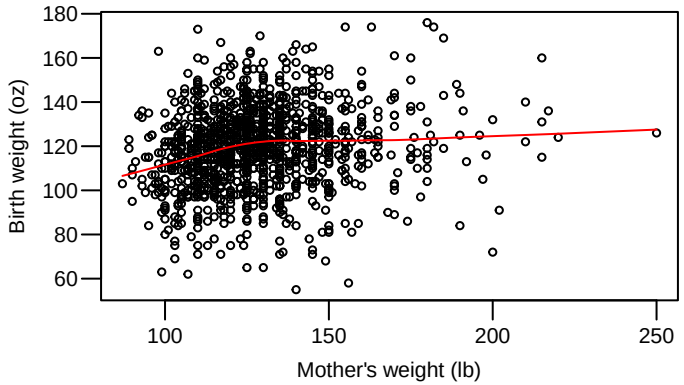
Note: In a simple linear regression of y on x, the resulting R^2 value is the square of the sample correlation coefficient. To illustrate:

```
> summary(lm(bwt ~ height,data = Babies.df))$r.squared
[1] 0.04149539
> cor(Babies.df$bwt,Babies.df$height)^2
[1] 0.04149539
```


Birth weight of babies...

Looking at the pairs plot again, we also see a somewhat weak relationship between `bwt` and mother's `weight`.

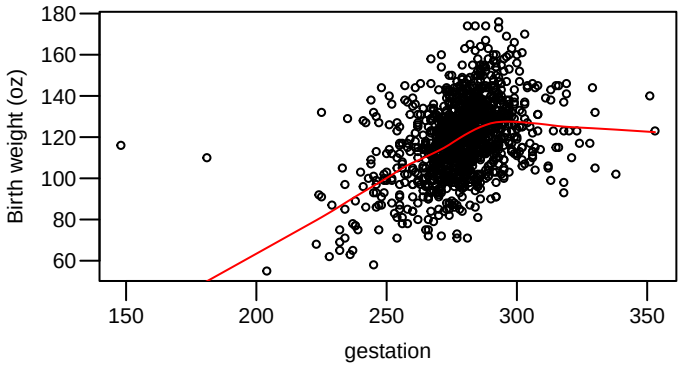
```
> plot(bwt ~ weight,data = Babies.df,
+           xlab="Mother's weight (lb)",ylab="Birth weight (oz)")
> lines(lowess(Babies.df$weight,Babies.df$bwt))
```



Birth weight of babies...

There is a stronger relationship between the `gestation` time for the babies and its `bwt` which is not surprising, as the longer the child is in the mother's womb the longer the child has had time to have nutrition and grow. But, this relationship distinctly flattens out beyond a certain gestational age – some people call this a "hockey stick" curve.

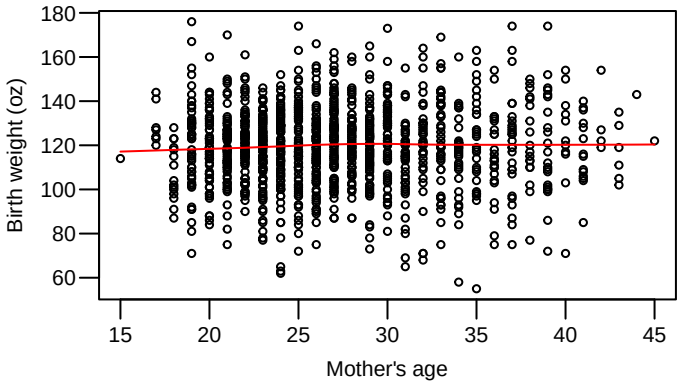
```
> plot(bwt ~ gestation,data = Babies.df,ylab="Birth weight (oz)")
> lines(lowess(Babies.df$gestation,Babies.df$bwt))
```



Birth weight of babies...

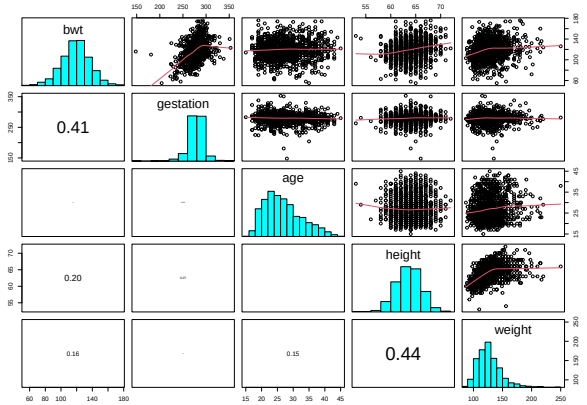
There does not seem to be any relationship between a mother's age and her child's `bwt`.

```
> plot(bwt ~ age,data = Babies.df,xlab="Mother's age",ylab="Birth weight (oz)")
> lines(lowess(Babies.df$age,Babies.df$bwt))
```



Birth weight of babies...

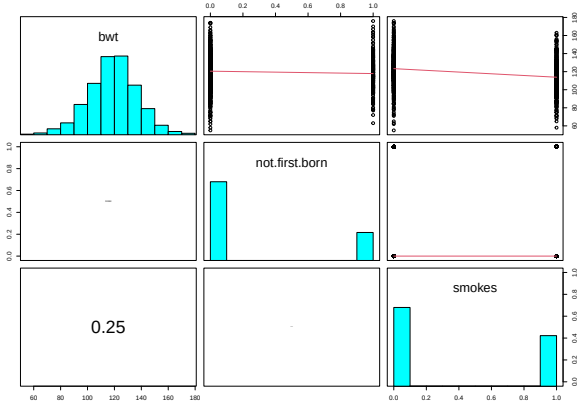
Note: There seem to be some outlying data points in these plots. There does not appear to be much of a relationship between the x variables, except between `height` and `weight`.



Birth weight of babies...

Let us look at the categorical (factor) explanatory variables against the baby's birth weight `bwt`.
The categorical variables are `not.first.born` and `smoke`, in columns 3 and 7 of the data frame `Babies.df`.

```
> pairs20x(Babies.df[,c(1,3,7)])
```



Birth weight of babies...

We see a slight decrease in babies `bwt` if the mother smokes. This increases the chance of a mother having a low birth weight baby if she smokes – perhaps another reason to avoid tobacco!

The variable `not.first.born` does not appear to have too much of an effect. This is perhaps not a surprise given that this variable may not be as important as it once was as family size has deceased markedly in the developed world (this is US data) and prenatal care has improved.

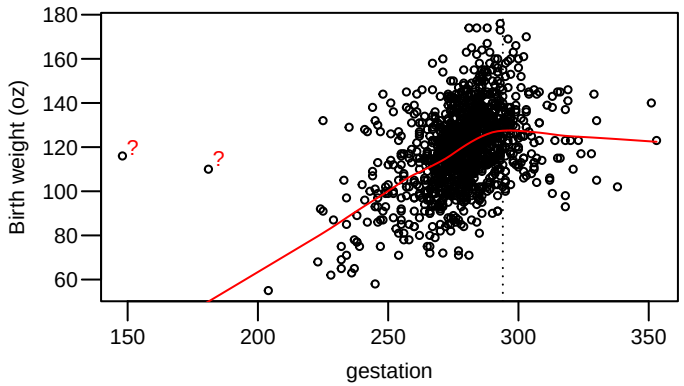
We will know begin our linear modelling of these data...

Birth weight of babies...

Relationship between birth weight and gestational age

Let us start with an understanding of `gestation` to explain `bwt` since it is the strongest relationship. The atypical data points have been marked with question marks. We will add other explanatory variables later.

```
> plot(bwt ~ gestation, data = Babies.df, ylab="Birth weight (oz)" )
> lines(lowess(Babies.df$gestation, Babies.df$bwt), col = "red")
> text(c(152, 185),c(120, 115), "?", col = "red")
> abline(v = 294,lty = 3)
```



Birth weight of babies...

Relationship between birth weight and gestational age...

Let us identify the two points denoted by the '?' symbol.

We can easily identify them in the plot as they have `gestation` < 200.

They look extremely implausible as they have typical birth-weight but have a gestational age that is extremely low for these data.

```
> id=(Babies.df$gestation<200)
> Babies.df[id,]
    bwt gestation not.first.born age height weight smokes
239 116       148             0  28     66    135      0
820 110       181             0  27     64    133      0
```

These points (observations 239 and 820) may be be unduly influential.

Birth weight of babies...

Relationship between birth weight and gestational age...

The above plot has a vertical line at 294 days. The relevance of 294 days is explained in the article "How Your Baby Grows During Pregnancy".

Most babies are born before 42 weeks $= 42 \times 7 = 294$ days. It seems that beyond this point babies cease to grow and hence the 'flattening out' and/or decrease. In other words, it looks like the effect of gestational age depends on whether the baby is overdue or not. That is, the effect of gestational age appears to change with overdue status.

We want to fit a model that fits a straight line for `gestation` ≤ 294 and then changes the slope of that line when `gestation` > 294.

We'll need to put our statistical thinking caps on, and devise a way to fit such a model.

Birth weight of babies...

Relationship between birth weight and gestational age...

For `gestation` ≤ 294 days we'll use the familiar simple linear regression model

$$\mathrm{E}[\texttt{bwt}] = \beta_0 + \texttt{gestation} \times \beta_1$$

We'd like to extend this model by adding an extra term so that the slope changes when `gestation` > 294. That is,

$$\mathrm{E}[\texttt{bwt}] = \beta_0 + \texttt{gestation} \times \beta_1 + v \times \beta_2$$

where v is some suitable explanatory variable. What should v be?

- For `gestation` ≤ 294 the extended model is just the simple linear regression model, so that means $v = 0$ when `gestation` ≤ 294.
- For `gestation` > 294 we need another slope effect for gestational age. In fact, we need $v =$ `gestation` $- 294$.[2]

[2] We substract the 294 so that the simple linear regression model and extended model have the same value when `gestation` $= 294$, because then $v = 0$.

Birth weight of babies...

Relationship between birth weight and gestational age...

Let's create the new explanatory $v = \texttt{gestation} - 294$ that is described above. We'll give it the name `ODdays` because it is the number of days that the baby is overdue.

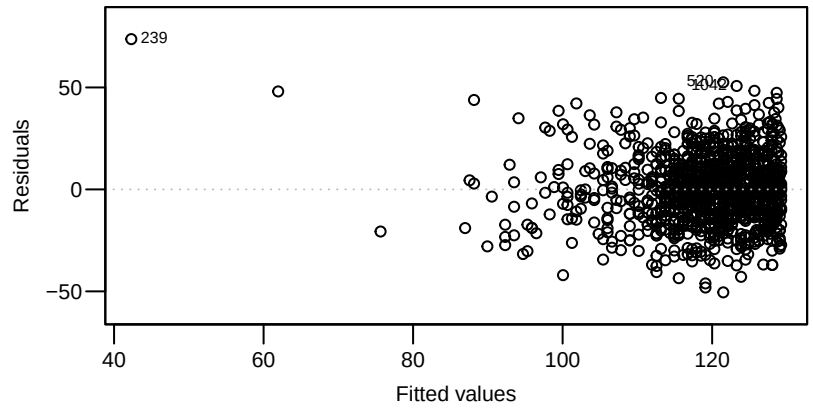
```
> head(Babies.df,12) #Print first 12 lines of dataframe
   bwt gestation not.first.born age height weight smokes ODdays
1  120       284              0  27     62    100      0      0
2  113       282              0  33     64    135      0      0
3  128       279              0  28     64    115      1      0
4  108       282              0  23     67    125      1      0
5  136       286              0  25     62     93      0      0
6  138       244              0  33     62    178      0      0
7  132       245              0  23     65    140      0      0
8  120       289              0  25     62    125      0      0
9  143       299              0  30     66    136      1      5
10 140       351              0  27     68    120      0     57
11 144       282              0  32     64    124      1      0
12 141       279              0  23     63    128      1      0
```


Section 10.3
Fitting the initial model

Birth weight of babies...

Our initial fitted model is the hockey stick model for the effect of gestational age.

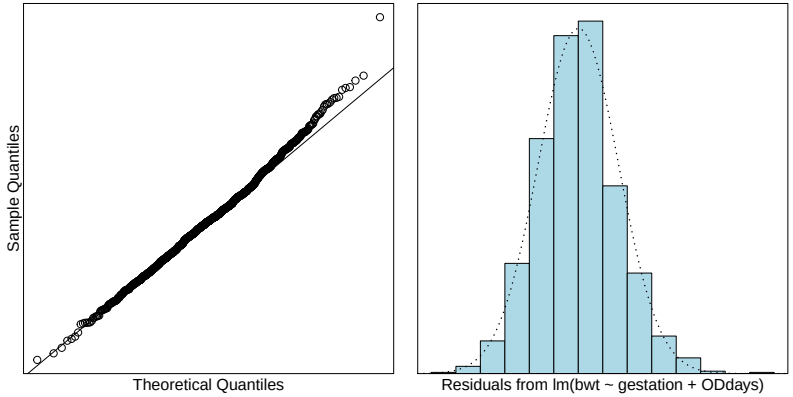
```
> bwt.fit=lm(bwt~ gestation+ODdays, data = Babies.df)
> plot(bwt.fit, which = 1, add.smooth = FALSE)
```



Observation 239 is a problem.

Birth weight of babies...

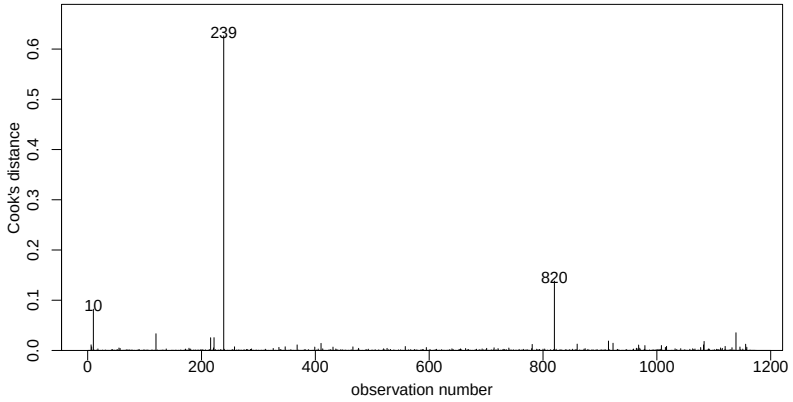
```
> normcheck(bwt.fit)
```



Other than observation 239, things look pretty good.

Birth weight of babies. . .

```
> cooks20x(bwt.fit)
```

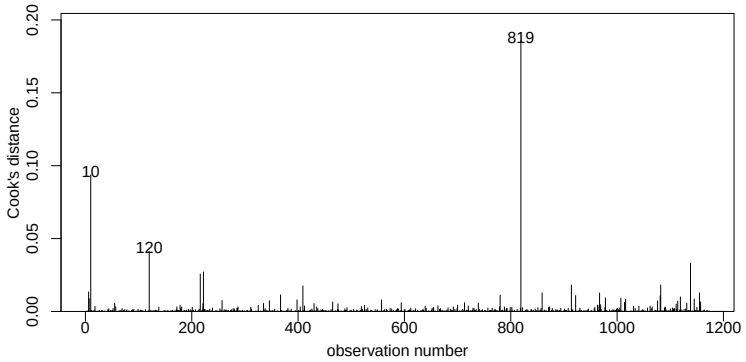


Point 239 is unduly influential. This baby has a gestational age of just 148 days, and yet has a weight typical of a full term baby. It is clearly a data-entry mistake and we will remove this data point.

Birth weight of babies. . .

Let us refit with observation 239 removed.

```
> bwt.fit2=lm(bwt~ gestation+ODdays,data = Babies.df[-239,])
> cooks20x(bwt.fit2)
```



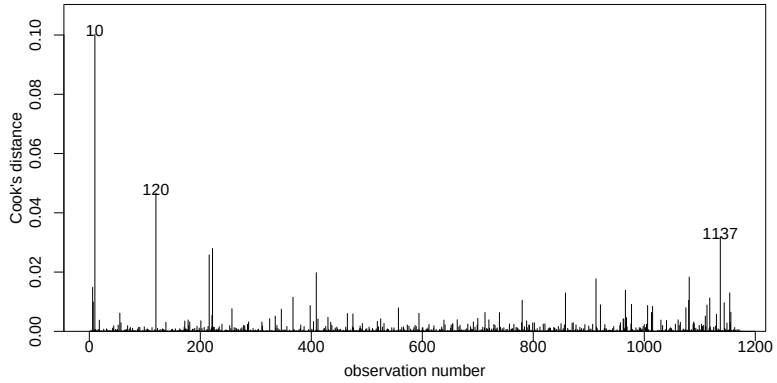
Although observation 820 [3] is not unduly influential, we shall make a judgement call, and remove it.

[3] Note that it is now identified as point 819 in this plot, but it was point 820 before we dropped point 239.

Birth weight of babies. . .

We refit the model using the reduced data.

```
> #This time we demonstrate using the subset argument to remove points
> bwt.fit3=lm(bwt~ gestation+ODdays,data = Babies.df, subset = -c(239, 820))
> cooks20x(bwt.fit3)
```

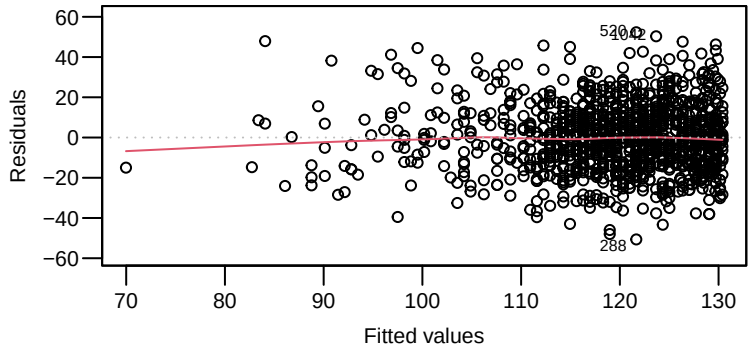


Now we have no unduly influential data points.

Birth weight of babies. . .

Let us recheck the residuals now that we have removed these two points.

```
> plot(bwt.fit3,which=1)
```

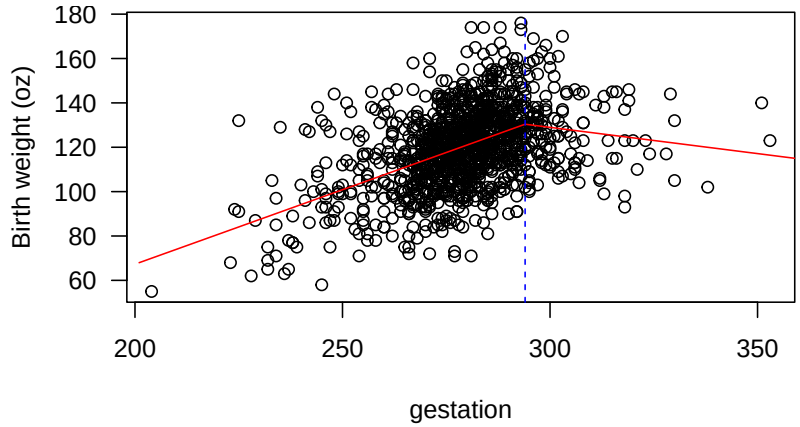


EOV seems fine now, and the residuals seem to be centred around zero.

Birth weight of babies. . .

Let's take a look at our fitted hockey stick model.

```
> gestation.seq=201:360 #Explanatory values at which to get predictions
> ODdays.seq=ifelse(gestation.seq<=294,0,gestation.seq-294)
> fit.seq=predict(bwt.fit3,new=data.frame(gestation=gestation.seq,
+                                         ODdays=ODdays.seq))
> plot(bwt~gestation,data=Babies.df[-c(239, 820),],ylab="Birth weight (oz)")
> lines(gestation.seq,fit.seq,col="red"); abline(v=294,lty=2,col="blue")
```



Model checks are good and no influential points remain, so we can trust this fit. Let's interpret the output.

```
> summary(bwt.fit3)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -66.95336   10.42810   -6.42 1.97e-10 ***
gestation     0.67124    0.03757   17.87  < 2e-16 ***
ODdays       -0.90783    0.11745   -7.73 2.31e-14 ***
---
Residual standard error: 16.23 on 1169 degrees of freedom
Multiple R-squared:  0.2188,Adjusted R-squared:  0.2174
F-statistic: 163.7 on 2 and 1169 DF,  p-value: < 2.2e-16
```

The fitted model is:

$$\mathrm{E}[\texttt{bwt}] = -66.95 + 0.67 \times \texttt{gestation} - 0.91 \times \texttt{ODdays}$$

Birth weight of babies. . .

So, for $\texttt{gestation} \leq 294$ days (i.e., $\texttt{ODdays = 0}$)

$$\mathrm{E}[\texttt{bwt}] = -66.95 + 0.67 \times \texttt{gestation}$$

That is, on average, babies initially grow at 0.67 oz per day until about 130 oz[4] at week 42 (i.e., day 294).

For $\texttt{gestation} > 294$ days (i.e., $\texttt{ODdays = gestation-294}$)

$$\begin{aligned}\mathrm{E}[\texttt{bwt}] &= -66.95 + 0.67 \times \texttt{gestation} - 0.91 \times (\texttt{gestation-294}) \\ &= 199.95 - 0.24 \times \texttt{gestation}\end{aligned}$$

So, on average, it is estimated that overdue babies lose about 0.24 oz per day after week 42.[5]

[4] $130 \approx -66.95 + 0.67 \times 294$

[5] **Question:** How could we test whether this is significantly different from zero?

Birth weight of babies. . .

Note that this model only explains about 22% of the variation in babies' birth weight, so it would be worth seeing if adding the other explanatory variables will help explain more.

In the `pairs20x` plot above we saw that `height` and `weight` had correlations of 0.20 and 0.16 with `bwt`.

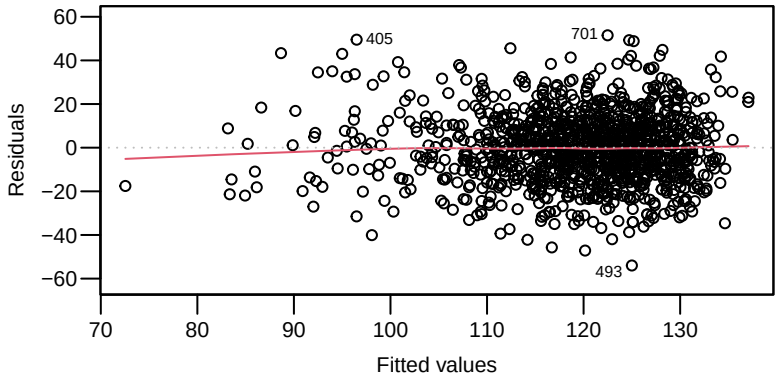
So let us see what we find when we introduce the `height` variable into the model. We will proceed with selecting variables one at a time (with reflection) – this is one of many multiple regression strategies!

Section 10.4
Multiple linear regression model: Adding more terms to the model and the peril of multi-collinearity

Birth weight of babies. . .

Let us add the height variable and see how it works out.

```
> bwt.fit4 = lm(bwt ~ gestation + ODdays + height, data = Babies.df,
+     subset = -c(239,820))
> plot(bwt.fit4,which=1)
```



All seems okay. Let us make sure that this makes sense in terms of output.

Birth weight of babies. . .

```
> summary(bwt.fit4)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -139.20571   15.05961  -9.244  < 2e-16 ***
gestation      0.65219    0.03703  17.613  < 2e-16 ***
ODdays        -0.89039    0.11543  -7.714 2.61e-14 ***
height         1.21083    0.18495   6.547 8.79e-11 ***
---
Residual standard error: 15.94 on 1168 degrees of freedom
Multiple R-squared:  0.2464,Adjusted R-squared:  0.2445
F-statistic: 127.3 on 3 and 1168 DF,  p-value: < 2.2e-16
```

This seems to make sense, whereby mother's height is positively related to a baby's birth weight (on average).

Note: We will drop the checking of fitted vs residuals plots as it has been okay to date and it is starting to get a little tedious. We will recheck this once we get to the final model.

Birth weight of babies. . .

Let us add weight to the model. We're going to save some typing and use the update function to update our model.[6]

```
> bwt.fit5 = update(bwt.fit4,~. + weight)
> summary(bwt.fit5)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -131.68169   15.14974  -8.692  < 2e-16 ***
gestation      0.65624    0.03688  17.795  < 2e-16 ***
ODdays        -0.90868    0.11502  -7.900 6.41e-15 ***
height         0.90486    0.20453   4.424 1.06e-05 ***
weight         0.08535    0.02485   3.434 0.000615 ***
---
Residual standard error: 15.87 on 1167 degrees of freedom
Multiple R-squared:  0.254,Adjusted R-squared:  0.2514
F-statistic: 99.32 on 4 and 1167 DF,  p-value: < 2.2e-16
```

This makes sense. Heavier mothers can be expected to have heavier babies.

[6] In the above use of update the ~. term is used to denote the model containing the explanatory variables in bwt.fit4.

Birth weight of babies. . .

The mother being very underweight or excessively overweight can have negative effects on their babies health, but neither `height` or `weight` directly measures this.

We will construct a new variable, body mass index `bmi`.

$$BMI = \frac{\texttt{mass in kg}}{\texttt{height in metres}^2} = \frac{\texttt{mass in lb}}{\texttt{height in inches}^2} \times 703$$

The World Health Organisation classifies BMIs in the range 18.5–25 as healthy, 25–30 as overweight, and 30+ as obese.

Birth weight of babies. . .

Let us add `bmi` to the current model.

```
> # Create the variable BMI and add it to the model
> Babies.df$bmi = (Babies.df$weight / (Babies.df$height^2) ) * 703
> bwt.fit6 = update(bwt.fit5,~. + bmi)
> summary(bwt.fit6)
```

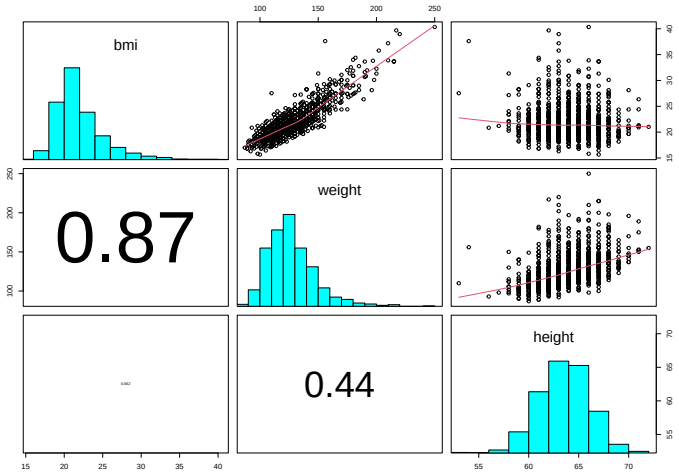
```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -216.33575   79.63707  -2.717  0.00669 **
gestation      0.65629    0.03688  17.798  < 2e-16 ***
ODdays        -0.90980    0.11502  -7.910 5.94e-15 ***
height         2.22845    1.23940   1.798  0.07243 .
weight        -0.24252    0.30382  -0.798  0.42490
bmi            1.90870    1.76280   1.083  0.27914
---
Residual standard error: 15.87 on 1166 degrees of freedom
Multiple R-squared:  0.2547,Adjusted R-squared:  0.2515
F-statistic:  79.7 on 5 and 1166 DF,  p-value: < 2.2e-16
```

Hang on. Everything has gone weird!!! None of `weight`, `height` or `bmi` is statistically significant (at the 5% level). So what is going on?

Birth weight of babies. . .

Let's look at these three variables to see what is happening.

```
> pairs20x(Babies.df[-c(239,820), c(9,6,5)])
```



Not surprisingly, we see that `bmi` and `weight` seem to explain each other.

Birth weight of babies. . .

The problem is that we have a redundancy in our explanatory variables. Here, `bmi` is explained by `weight` and vice-versa. Note that adding `bmi` to the model barely changed R^2 and so is telling us that it did not increase our ability to explain variability in birth weight.

In essence the statistically significance (i.e., *P*-value) of an explanatory variable is measuring its contribution toward explaining variability in the response variable (in our case `bwt`) *having adjusted for any other explanatory variables in the model*.

So `bmi` explains little variability in `bwt` since `weight` has already explained most of that variability, and vice-versa.

This problem is given the name **multi-collinearity**[7].

In linear algebra, we say we have linear dependence (as opposed to linear independence) in these variables.

[7] The double 'l' is not a mistake.

Birth weight of babies...

Back to the drawing board. Let us refit this model with `bmi` and `height`, but without `weight`.

```
> bwt.fit7 = update(bwt.fit6,~. - weight)
> summary(bwt.fit7)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -153.99353   15.56725  -9.892  < 2e-16 ***
gestation      0.65633    0.03687  17.801  < 2e-16 ***
ODdays        -0.90933    0.11500  -7.907 6.06e-15 ***
height         1.25013    0.18440   6.779 1.91e-11 ***
bmi            0.50629    0.14415   3.512 0.000461 ***
---
Residual standard error: 15.87 on 1167 degrees of freedom
Multiple R-squared:  0.2543,Adjusted R-squared:  0.2518
F-statistic:  99.5 on 4 and 1167 DF,  p-value: < 2.2e-16
```

Let us next investigate whether the categorical variable (`smokes`) helps to explain further variability in `bwt`.

Birth weight of babies...

Let us add `smokes` to this analysis.

```
> bwt.fit8=update(bwt.fit7,~. + smokes)
> summary(bwt.fit8)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -144.07719   15.15079  -9.510  < 2e-16 ***
gestation      0.63198    0.03589  17.608  < 2e-16 ***
ODdays        -0.88104    0.11164  -7.892 6.84e-15 ***
height         1.28081    0.17898   7.156 1.46e-12 ***
bmi            0.41516    0.14029   2.959  0.00315 **
smokes        -7.93655    0.92711  -8.561  < 2e-16 ***
---
Residual standard error: 15.4 on 1166 degrees of freedom
Multiple R-squared:  0.2984,Adjusted R-squared:  0.2954
F-statistic: 99.18 on 5 and 1166 DF,  p-value: < 2.2e-16
```

As we might have suspected, a mother smoking is associated with decreased birth weight.

Birth weight of babies...

Let us see if `not.first.born` is useful:

```
> bwt.fit9=update(bwt.fit8,~. + not.first.born)
> summary(bwt.fit9)
```

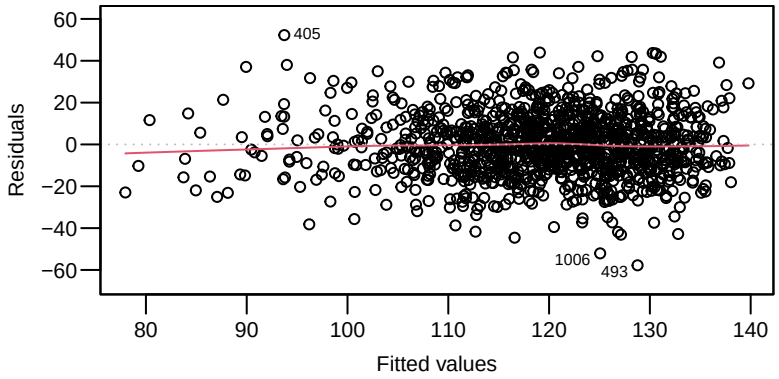
```
Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)  -145.56797   15.08855  -9.648  < 2e-16 ***
gestation       0.64129    0.03583  17.897  < 2e-16 ***
ODdays         -0.89215    0.11119  -8.024 2.48e-15 ***
height          1.29912    0.17825   7.288 5.78e-13 ***
bmi             0.35469    0.14078   2.520 0.011882 *
smokes         -7.98201    0.92301  -8.648  < 2e-16 ***
not.first.born -3.51137    1.02978  -3.410 0.000672 ***
---
Residual standard error: 15.33 on 1165 degrees of freedom
Multiple R-squared:  0.3053,Adjusted R-squared:  0.3018
F-statistic: 85.34 on 6 and 1165 DF,  p-value: < 2.2e-16
```

Hmmm, does the negative effect of `not.first.born` seem reasonable???

Birth weight of babies...

Let us check the assumptions on this final model:
Independence should be okay, as this is (hopefully) a random sample of data from a carefully designed study.

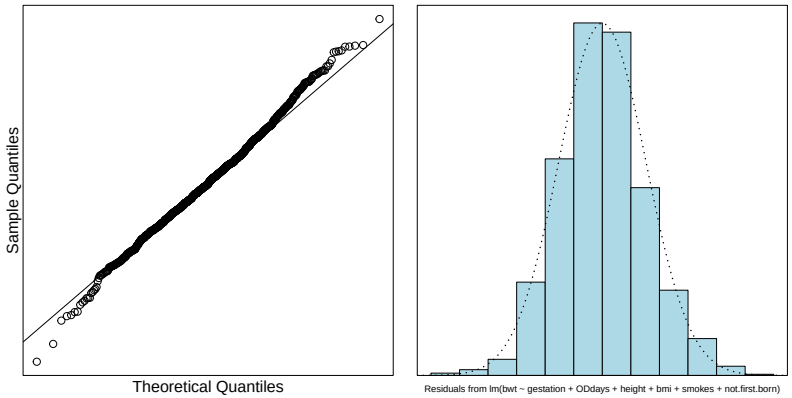
```
> plot(bwt.fit9,which=1)
```



No trend, and EOV assumption is fine.

Birth weight of babies...

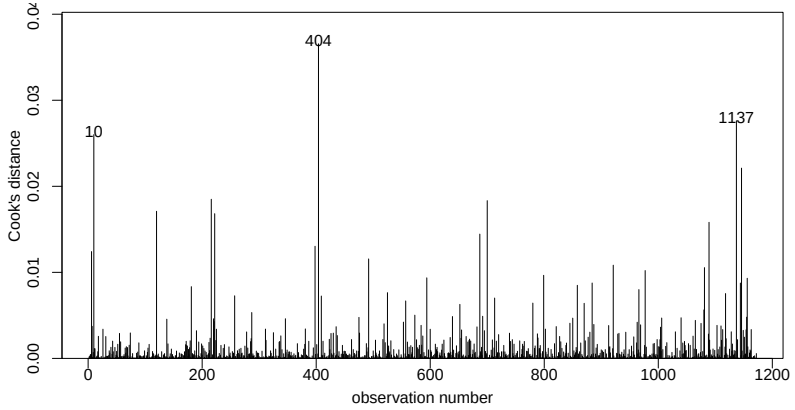
```
> normcheck(bwt.fit9)
```



Normality assumption looks fine.

Birth weight of babies...

```
> cooks20x(bwt.fit9)
```



No unduly influential points.

Birth weight of babies...

Let us get the CIs on this trusted output.

```
> confint(bwt.fit9)
                      2.5 %       97.5 %
(Intercept)    -175.17174865 -115.9641969
gestation         0.57098950    0.7115993
ODdays           -1.11029525   -0.6740001
height            0.94938608    1.6488470
bmi               0.07849275    0.6308947
smokes           -9.79295880   -6.1710576
not.first.born   -5.53179563   -1.4909493
```

See Case Study 10.1 for a detailed executive summary.

Birth weight of babies...

Closing remarks

Recall that we can fit as many explanatory variables as we like. So, did fitting all of these explanatory variables help us describe the variability of the birth weight of babies?

	What we did	Multiple R^2
`bwt.fit3`	Added `gestation+ODdays`	21.9%
`bwt.fit4`	Added `height`	24.6%
`bwt.fit5`	Added `weight`	25.4%
`bwt.fit6`	Added `bmi`	25.5%
`bwt.fit7`	Dropped `weight`	25.4%
`bwt.fit8`	Added `smokes`	29.8%
`bwt.fit9`	Added `not.first.born`	30.5%

Our final model, `bwt.fit9`, includes explanatory variables we deemed suitable and it has a Multiple R^2 of 30.5%.

Section 10.5
Closing remarks and relevant R-code

Closing remarks

In situations where there are many explanatory variables, some of which may be strongly correlated, selecting the best subset for the final model can be challenging.

Model selection is a crucial component of statistical modelling and machine learning, especially in the context of "big data" where there may be millions of observations and thousands of potential explanatory variables.

STATS 330 (Advanced Statistical Modelling) covers this topic in more detail, using techniques such as stepwise variable selection, AIC (Akaike's information criterion), and assessment of prediction error using cross validation.

Most of the R-code you need for this chapter

Note that this code comes with the usual code/checks discussed in chapters 1 and 2.

Useful tools for inspecting many relationships are:

```
> ## Create the pairs plot of the five numeric variables
> pairs20x(Babies.df[,c(1,2,4,5,6)])
```

and for the factor variables:

```
> pairs20x(Babies.df[,c(1,3,7)])
```

Then it is a process of repeatedly updating the model and using Occam's razor to determine a preferred model. E.g.,

```
> model2=update(model1, ~. + xvariable2)
```

This requires constant vigilance to avoid multi-collinearity

Also note that some times several different models may be selected that all make sense and are acceptable.

Case Study 10.1: Birthweight of babies

Tou Ohone Andate - staff number 1234567

Background

Let's examine what affects the birth weight of babies.

- `bwt`: birth weight in ounces
- `gestation`: length of pregnancy in days
- `not.first.born`: 0=first born, 1=not first-born
- `age`: mother's age in years
- `height`: mother's height in inches
- `weight`: mother's pre-pregnancy weight in pounds
- `smoke`: smoking status of mother 0=not now, 1=yes.

This dataset was obtained from http://www.stat.berkeley.edu/users/statlabs/labs.html.

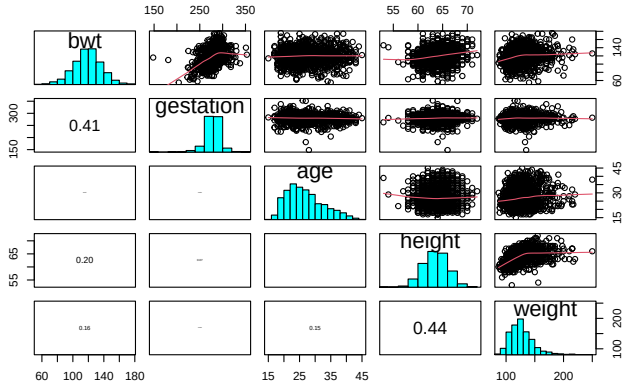
It accompanies the excellent text Stat Labs: Mathematical Statistics through Applications Springer-Verlag (2001) by Deborah Nolan and Terry Speed.

Question of Interest

We want to build a model to explain the birth weight of babies.

Read in and Inspect the Data

```
Babies.df = read.table("babies_data.txt", header = T)
pairs20x(Babies.df[, c(1, 2, 4, 5, 6)])
```



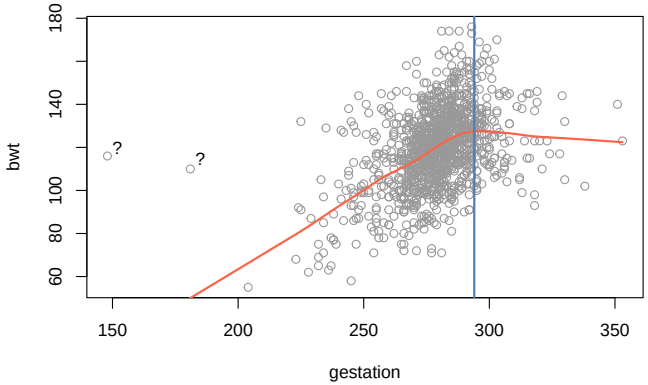
Looking at the pairs plot, we see a somewhat weak relationship between `bwt` and mother's `height` and `weight`.

There is a stronger relationship between the gestation time (`gestation`) for the babies and it's `bwt` which is not surprising, as the longer the child is in the mother's womb the longer the child has had time to have nutrition and grow — up to a point — then it 'flattens out' somewhat.

There doesn't seem to be any relationship between a mother's `age` and her child's `bwt`.

Let us look deeper into the relationship between `bwt` and `gestation`.

```
plot(bwt ~ gestation, data = Babies.df, col = "gray60")
lines(lowess(Babies.df$gestation,Babies.df$bwt), col = "tomato", lwd = 2)
text(152, 120, "?")
text(185, 115, "?")
abline(v = 294, col = "steelblue", lwd = 2)
```



Note also that there seems to be some 'weird' data points in these plots. There does not appear to be much of a relationship between the Xs. That is, the explanatory variables do not seem to have any strong relationships between them.

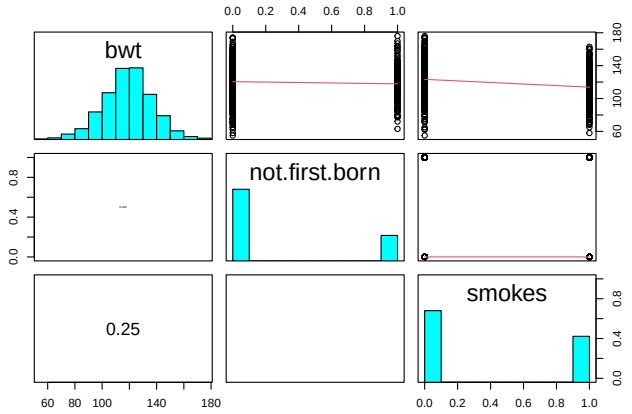
Most babes are born before 42 weeks $= 42 * 7 = 294$ days[1]. It seems that beyond this point babies cease to grow and hence the 'flattening out' and/or decrease. We'll create a dummy variable `OD` (for overdue) for this time point.

Let's look at the categorical (factor) data variables against the baby's birth weight (`bwt`).

They are `not.first.born` and `smoke`.

[1] "American College of Obstetricians and Gynaecologists - How Your Baby Grows During Pregnancy". See https://www.acog.org/-/media/For-Patients/faq156.pdf?dmc=1&ts=20150329T2112264959.

```
pairs20x(Babies.df[, c(1, 3, 7)])
```



Here, we only see a slight relationship between whether the mother smokes (`smoke`) and `bwt`. There is a slight decrease in babies `bwt` if the mother smokes. This increases the chance of a mother having a low birth weight baby if she smokes – perhaps another reason to avoid tobacco!

The variable `not.first.born` does not appear to have too much of an effect — which is perhaps not a surprise given that this variable may not be as important as it once was as family size has deceased markedly in the developed world (this is US data). We'll check this out later.

Model Building and Check Assumptions

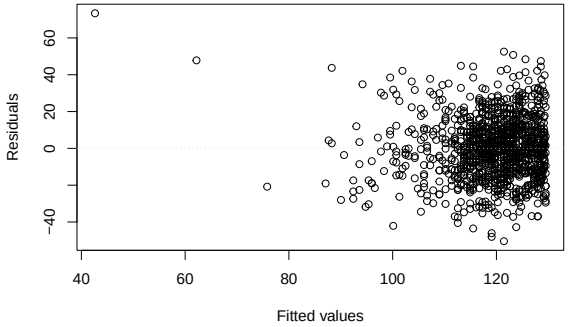
```
# Let's create OD as mentioned earlier.
Babies.df$OD = 1 * (Babies.df$gestation > 294)
range(Babies.df$gestation[Babies.df$OD == 0]) # Check
```

```
## [1] 148 294
```

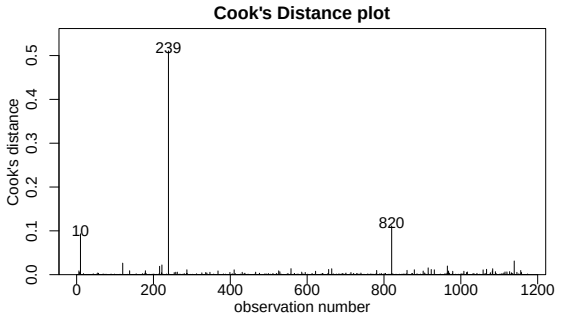
```
range(Babies.df$gestation[Babies.df$OD == 1]) # Check
```

```
## [1] 295 353
```

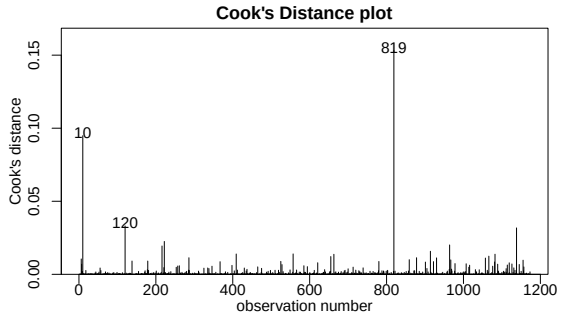
```
bwt.fit = lm(bwt ~ gestation * OD, data = Babies.df)
eovcheck(bwt.fit)
```

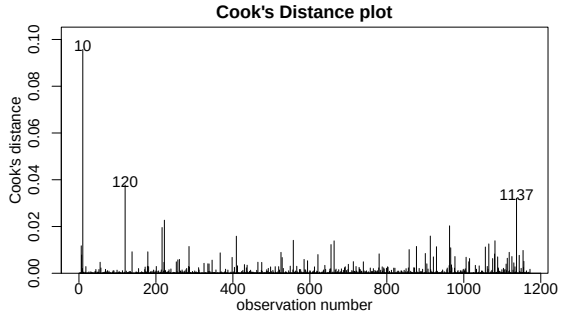

```
cooks20x(bwt.fit)
```




```
bwt.fit2 = lm(bwt ~ gestation * OD, data = Babies.df[-239, ])
cooks20x(bwt.fit2)
```




```
bwt.fit3 = lm(bwt ~ gestation * OD, data = Babies.df[-c(239, 820), ])
cooks20x(bwt.fit3)
```




```
bwt.fit4 = lm(bwt ~ gestation * OD + weight, data = Babies.df[-c(239, 820), ])
summary(bwt.fit4)
```

```
##
## Call:
## lm(formula = bwt ~ gestation * OD + weight, data = Babies.df[-c(239,
##     820), ])
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -49.23 -11.16  -0.26  10.01  48.71
##
```



```
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -85.06704   11.17499  -7.612 5.54e-14 ***
## gestation      0.67509    0.03907  17.279  < 2e-16 ***
## OD           263.71433   41.35432   6.377 2.60e-10 ***
## weight         0.13329    0.02255   5.910 4.50e-09 ***
## gestation:OD  -0.90002    0.13658  -6.589 6.67e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16 on 1167 degrees of freedom
## Multiple R-squared:  0.2416, Adjusted R-squared:  0.239
## F-statistic: 92.92 on 4 and 1167 DF,  p-value: < 2.2e-16
```

```
bwt.fit5 = lm(bwt ~ gestation * OD + weight + height, data = Babies.df[-c(239, 820), ])
summary(bwt.fit5)
```

```
##
## Call:
## lm(formula = bwt ~ gestation * OD + weight + height, data = Babies.df[-c(239,
##     820), ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -53.099 -10.586   0.089  10.005  47.764
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -132.96277   15.50163  -8.577  < 2e-16 ***
## gestation       0.66108    0.03889  16.998  < 2e-16 ***
## OD            258.01065   41.04989   6.285 4.61e-10 ***
## weight          0.08541    0.02486   3.436 0.000612 ***
## height          0.90454    0.20460   4.421 1.07e-05 ***
## gestation:OD   -0.88049    0.13558  -6.494 1.23e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.88 on 1166 degrees of freedom
## Multiple R-squared:  0.2541, Adjusted R-squared:  0.2509
## F-statistic: 79.43 on 5 and 1166 DF,  p-value: < 2.2e-16
```

```
# Let's create BMI from both of these measurements
Babies.df$bmi = with(Babies.df, weight/(height^2) * 703)
bwt.fit6 = lm(bwt ~ gestation * OD + weight + height + bmi,
    data = Babies.df[-c(239, 820), ])
summary(bwt.fit6)
```

```
##
## Call:
## lm(formula = bwt ~ gestation * OD + weight + height + bmi, data = Babies.df[-c(239,
##     820), ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -53.577 -10.367   0.066  10.042  47.803
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  -217.97090   79.76837  -2.733  0.00638 **
## gestation       0.66127    0.03889  17.004  < 2e-16 ***
## OD            258.08808   41.04678   6.288 4.55e-10 ***
## weight         -0.24369    0.30395  -0.802  0.42287
## height          2.23309    1.23990   1.801  0.07196 .
## bmi             1.91588    1.76352   1.086  0.27753
## gestation:OD   -0.88083    0.13557  -6.497 1.21e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.87 on 1165 degrees of freedom
## Multiple R-squared:  0.2548, Adjusted R-squared:  0.251
## F-statistic:  66.4 on 6 and 1165 DF,  p-value: < 2.2e-16
```

```
bwt.fit7 = lm(bwt ~ gestation * OD + height + bmi + not.first.born,
    data = Babies.df[-c(239, 820), ])
summary(bwt.fit7)
```

```
##
## Call:
## lm(formula = bwt ~ gestation * OD + height + bmi + not.first.born,
##     data = Babies.df[-c(239, 820), ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.278 -10.402   0.002   9.650  46.957
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -156.43942   15.86220  -9.862  < 2e-16 ***
## gestation         0.66898    0.03881  17.237  < 2e-16 ***
## OD              263.77416   40.92055   6.446 1.68e-10 ***
## height            1.26751    0.18384   6.895 8.83e-12 ***
## bmi               0.44904    0.14480   3.101  0.00197 **
## not.first.born   -3.37234    1.06276  -3.173  0.00155 **
## gestation:OD     -0.89933    0.13515  -6.654 4.38e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.81 on 1165 degrees of freedom
## Multiple R-squared:  0.2608, Adjusted R-squared:  0.257
## F-statistic:  68.5 on 6 and 1165 DF,  p-value: < 2.2e-16
```

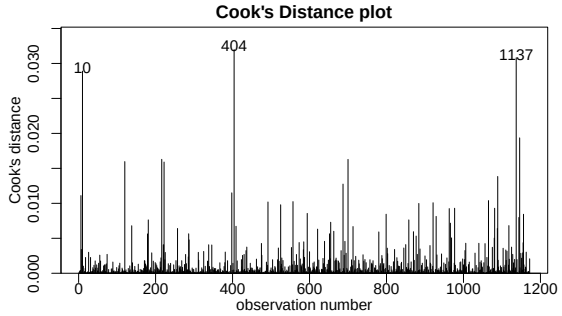
```
bwt.fit8 = lm(bwt ~ gestation * OD + height + bmi + not.first.born + smokes,
    data = Babies.df[-c(239, 820), ])
summary(bwt.fit8)
```

```
##
## Call:
## lm(formula = bwt ~ gestation * OD + height + bmi + not.first.born +
##     smokes, data = Babies.df[-c(239, 820), ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -57.805  -9.985  -0.623   9.184  52.282
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
```

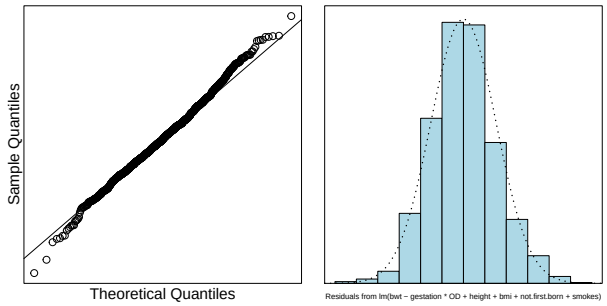


```
## (Intercept)    -146.36091   15.42725  -9.487  < 2e-16 ***
## gestation         0.64424    0.03775  17.067  < 2e-16 ***
## OD              256.69266   39.69307   6.467 1.47e-10 ***
## height            1.29903    0.17832   7.285 5.93e-13 ***
## bmi               0.35512    0.14084   2.521 0.011821 *
## not.first.born   -3.50274    1.03078  -3.398 0.000701 ***
## smokes           -7.98064    0.92340  -8.643  < 2e-16 ***
## gestation:OD     -0.87488    0.13110  -6.673 3.86e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.33 on 1164 degrees of freedom
## Multiple R-squared:  0.3054, Adjusted R-squared:  0.3012
## F-statistic:  73.1 on 7 and 1164 DF,  p-value: < 2.2e-16
```

```
cooks20x(bwt.fit8)
```




```
normcheck(bwt.fit8)
```




```
confint(bwt.fit8)
```

```
##                       2.5 %       97.5 %
## (Intercept)    -176.62923297 -116.0925891
## gestation         0.57017733    0.7182955
## OD              178.81470454  334.5706219
## height            0.94916219    1.6489067
## bmi               0.07878781    0.6314538
## not.first.born   -5.52512194   -1.4803515
## smokes           -9.79235265   -6.1689248
## gestation:OD     -1.13210128   -0.6176549
```

Method and Assumption Checks

Looking at the pairs plot, we saw that birthweight was related to a number of our explanatory variables. We will construct a multiple linear regression model with a suitable selection of the explanatory variables.

Observations 239 and 820 were found to be highly influential. They were deemed to be anomolous and were removed from the dataset.

The hockey stick relationship between gestational age and birthweight required allowing the age effect to differ depending on whether the baby was overdue, and was fitted by including an interaction term between age and overdue status. Moreover, we also decided to include body mass weight as an explanatory variable, but had to remore weight as an explanatory due to multicollinearity. All model assumptions were satisfied by our final model.

Using forward model selection (i.e., adding the most promising explanatory variables in turn), our final model is

$$bwt_i = \beta_0 + \beta_1 \times gestation_i + \beta_2 \times OD_i + \beta_3 \times height_i \ + \beta_4 \times bmi_i + \beta_5 \times not.first.born_i + \\ \beta_6 \times smokes_i + \beta_7 \times gestation_i \times OD_i + \epsilon_i,$$

where $\epsilon_i \ iid \sim N(0, \sigma)$. Here our three indicator variables take the value 1 if the baby was overdue, not the first born, and the mother smokes, respectively.

Our model only explains about 31% of the variability in a baby's birthweight.

Executive Summary.

We wanted to build a model to explain the birth weight of babies.

Keeping all other varaiables constant:

- A child has a higher expected birth-weight the longer its gestation time — up to a 42 weeks — then it starts decreasing in size the longer it stays unborn. We estimated an expected increase of 0.57 to 0.71 ounces per gestation day. After 42 weeks this will decrease by about -0.61 to -1.13 ounces per gestational day [NOTE: it might have been better had we changed the OD baseline].
- We estimated that for each additional inch of mother's height the baby's birthweight increases by 0.94 to 1.64 ounces, on average.
- We estimated that for each unit change in a mothers BMI the baby'sb irthweight increases by 0.08 to 0.63 ounces, on average.
- If the mother smokes this reduces the baby's birthweight by 6.17 to 9.79 ounces, on average.
- Not being first born seems to reduce the baby's birthweight by 1.48 to 5.52 ounces, on average.

Exercise

Is there significant evidence that expected birthweight decreases with increasing gestational age for babies that are overdue? Provide a confidence interval for this effect.

Case Study 10.2: Gender differences in salary

Tou Ohone Andate - staff number 1234567

Problem

These are the salary data used in Weisberg's book[1] consisting of observations on six variables for 52 professors in a small college. We want to build a model to predict salary. Of particular interest was the effect (if any) of gender on salary.

The variables of interest were:

- `sx`: Sex, (female or male)
- `rk`: Rank, (assistant, associate or full)
- `yr`: Number of years in current rank
- `dg`: Highest degree (masters or doctorate)
- `yd`: Number of years since highest degree was earned,
- `sl`: Academic year salary, in dollars.

Question of Interest

We want to build a model to explain salary. In particular, the effect (if any) of gender on salary.

Read in and Inspect the Data

```
salary.df = read.table("salary.txt", header = T)
salary.df$sx=factor(salary.df$sx)
salary.df$dg=factor(salary.df$dg)
salary.df$rk=factor(salary.df$rk)
head(salary.df)
```

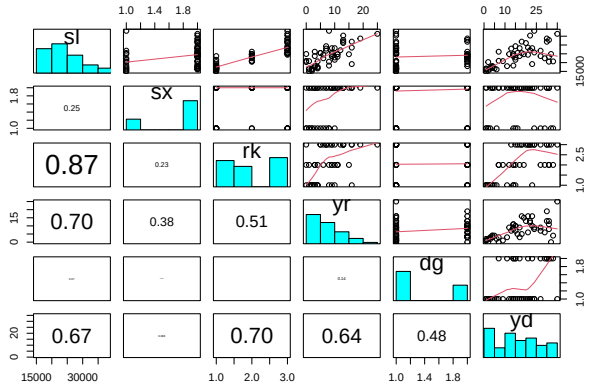
```
##       sx   rk yr        dg yd    sl
## 1   male full 25 doctorate 35 36350
## 2   male full 13 doctorate 22 35350
## 3   male full 10 doctorate 23 28200
## 4 female full  7 doctorate 27 26775
## 5   male full 19   masters 30 33696
## 6   male full 16 doctorate 21 28516
```

```
tail(salary.df)
```

```
##        sx        rk yr        dg yd    sl
## 47 female assistant  2 doctorate  6 16150
## 48 female assistant  2 doctorate  2 15350
## 49   male assistant  1 doctorate  1 16244
## 50 female assistant  1 doctorate  1 16686
## 51 female assistant  1 doctorate  1 15000
## 52 female assistant  0 doctorate  2 20300
```

```
pairs20x(salary.df[, c("sl", "sx", "rk", "yr", "dg", "yd")])
```

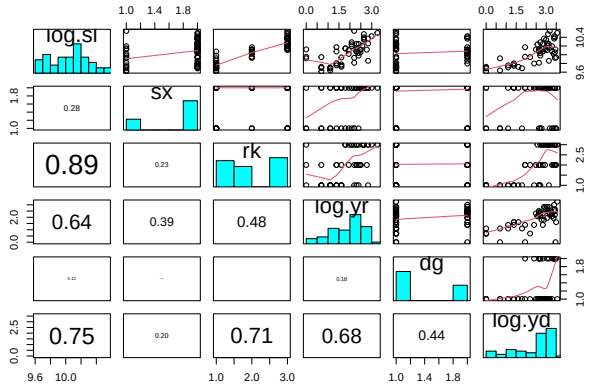
[1] S. Weisberg (1985). Applied Linear Regression, Second Edition. New York: John Wiley and Sons. Page 194.



It seems that males have a larger mean `sl` than females. The higher the rank, the higher the `sl` (Note that the levels of `rk` are in alphabetical order). As `yr` increases the expected `sl` increases. Not too much going on in the relationship between `sl` and `dg`. There is a weak positive relationship between `sl` and `yd`. Also, as `yd` increases the variability in `sl` increases.

Since the response variable is salary, it would make sense to use log-salary so that effects will be multiplicative. Some previous analyses have also used `log(yr)` and `log(yd)` as explanatory variables. It is not obvious that this is the best choice, but for consistency we will follow these previous analyses.

```
salary.df$log.yr = log(salary.df$yr+1) # log(yr + 1) since log(0) = -infinity
salary.df$log.yd = log(salary.df$yd)
salary.df$log.sl= log(salary.df$sl)
pairs20x(salary.df[, c("log.sl", "sx", "rk", "log.yr", "dg", "log.yd")])
```

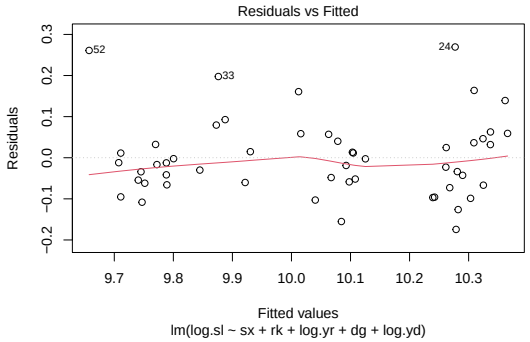


It seems that males have a larger mean `log(sl)` than females. The higher the rank, the higher the `log(sl)`. As `log(yr)` increases the expected `log(sl)` increases. However, the low `log(sl)`s do not follow this observed trend. Still not too much going on in the relationship between `log(sl)` and `dg`. There is a weak positive relationship between `log(sl)` and `log(yd)`. Also, as `log(yd)` increases the variability in `log(sl)` increases. In comparison to `sl` and `yd`, there is some improvement.

We'll find our preferred model by fitting all terms, and then successively removing the one that is least significant, until all remaining terms are significant.

Model Building and Check Assumptions

```
salary.fit = lm(log.sl ~ sx + rk + log.yr + dg + log.yd, data = salary.df)
plot(salary.fit, which = 1)
```




```
summary(salary.fit)
```

```
##
## Call:
## lm(formula = log.sl ~ sx + rk + log.yr + dg + log.yd, data = salary.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.17438 -0.06067 -0.01456  0.04168  0.26938
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.654427   0.043548 221.698  < 2e-16 ***
## sxmale      -0.003494   0.036384  -0.096 0.923914
## rkassociate  0.213574   0.050898   4.196 0.000126 ***
## rkfull       0.429959   0.056213   7.649 1.12e-09 ***
## log.yr       0.081603   0.027493   2.968 0.004787 **
## dgmasters    0.026184   0.039074   0.670 0.506203
## log.yd       0.004235   0.031797   0.133 0.894646
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1012 on 45 degrees of freedom
## Multiple R-squared:  0.8528, Adjusted R-squared:  0.8332
```



```
## F-statistic: 43.46 on 6 and 45 DF,  p-value: < 2.2e-16
```

```
salary.fit2 = lm(log.sl ~ rk + log.yr + dg + log.yd, data = salary.df)
summary(salary.fit2)
```

```
##
## Call:
## lm(formula = log.sl ~ rk + log.yr + dg + log.yd, data = salary.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.17515 -0.06095 -0.01423  0.04172  0.27191
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 9.652977   0.040407 238.892  < 2e-16 ***
## rkassociate 0.212025   0.047750   4.440 5.59e-05 ***
## rkfull      0.428683   0.054027   7.935 3.69e-10 ***
## log.yr      0.080503   0.024722   3.256  0.00212 **
## dgmasters   0.025820   0.038468   0.671  0.50545
## log.yd      0.005058   0.030286   0.167  0.86809
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1001 on 46 degrees of freedom
## Multiple R-squared:  0.8528, Adjusted R-squared:  0.8368
## F-statistic:  53.3 on 5 and 46 DF,  p-value: < 2.2e-16
```

```
salary.fit3 = lm(log.sl ~ rk + log.yr + dg, data = salary.df)
summary(salary.fit3)
```

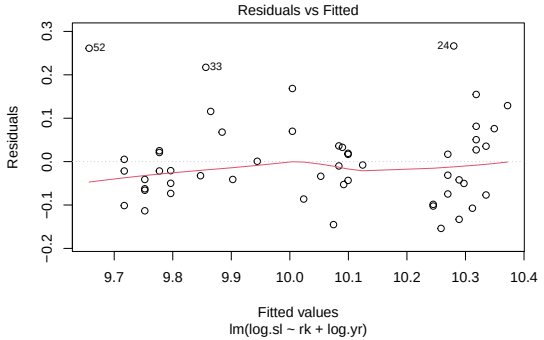
```
##
## Call:
## lm(formula = log.sl ~ rk + log.yr + dg, data = salary.df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.17723 -0.06084 -0.01646  0.04136  0.27343
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.65585    0.03618 266.856  < 2e-16 ***
## rkassociate  0.21669    0.03832   5.655 8.92e-07 ***
## rkfull       0.43522    0.03686  11.807 1.16e-15 ***
## log.yr       0.08284    0.02015   4.111 0.000157 ***
## dgmasters    0.02934    0.03186   0.921 0.361820
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09902 on 47 degrees of freedom
## Multiple R-squared:  0.8527, Adjusted R-squared:  0.8402
## F-statistic: 68.02 on 4 and 47 DF,  p-value: < 2.2e-16
```

```
salary.fit4 = lm(log.sl ~ rk + log.yr, data = salary.df)
summary(salary.fit4)
```

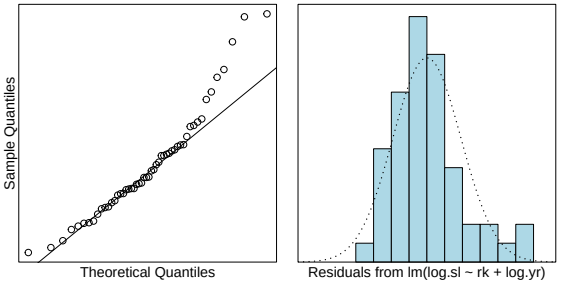
```
##
## Call:
## lm(formula = log.sl ~ rk + log.yr, data = salary.df)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.15389 -0.06344 -0.02122  0.03568  0.26648
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.65710    0.03610 267.503  < 2e-16 ***
## rkassociate  0.22719    0.03653   6.220 1.16e-07 ***
## rkfull       0.43264    0.03670  11.789 8.85e-16 ***
## log.yr       0.08661    0.01970   4.396 6.08e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09886 on 48 degrees of freedom
## Multiple R-squared:   0.85,  Adjusted R-squared:  0.8407
## F-statistic:  90.7 on 3 and 48 DF,  p-value: < 2.2e-16
```

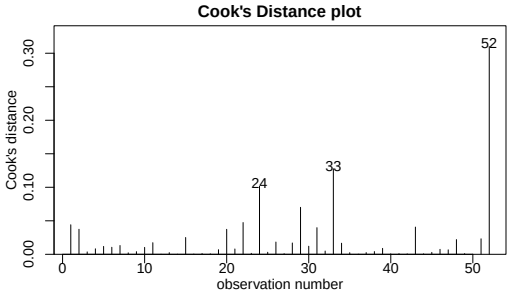
```
plot(salary.fit4, which = 1)
```




```
normcheck(salary.fit4)
```


```
cooks20x(salary.fit4)
```




```
exp(confint(salary.fit4))
```

```
##                    2.5 %       97.5 %
## (Intercept) 14537.893426 16809.266030
## rkassociate     1.166196     1.350709
## rkfull          1.431690     1.659352
## log.yr          1.048118     1.134534
```

Get `log.yr` Effect for Doubling of Time at Current Rrank

```
2^confint(salary.fit4)[4, ]
```

```
##    2.5 %   97.5 %
## 1.033112 1.091432
```

Method and Assumption Checks

We have a numeric response and multiple explanatory vairables, so we fitted a multiple linear regression model. After looking at pairwise plots, salary and the two year explanatory variables were logged.

After fitting all explanatory terms, Occam's razor was applied to remove those that were not significant[2]. We dropped the variables in the following order:

- `sx` ($P\text{-}value = 0.92$).
- `log(yd)` ($P\text{-}value = 0.87$).
- `dg` ($P\text{-}value = 0.36$).

All model assumptions looked reasonably well satisfied, notwithstanding that the residuals were a bit right skewed.

Our final model is

$$log(salary_i) = \beta_0 + \beta_1 \times rank.associate_i + \beta_2 \times rank.full_i + \beta_3 \times log(yr_i) + \epsilon_i,$$

where $\epsilon_i \sim iid\ N(0, \sigma^2)$. Here $rank.associate$ and $rank.full$ are equal to 1 if rank is associate or full, respectively, otherwise they are zero.

Our model explains about 85% of the variability in the log of salary.

Executive Summary

We wanted to build a model to explain salary. In particular, the effect (if any) of gender on salary.

Our final model used the rank of the professor and the number of years at the current rank to explain their salary. After adjusting for these, gender, highest degree, and number of years since highest degree was earned were not required.

We estimate that being promoted from assistant to associate professor increases median salary by between 17% to 35%.

We estimate that being promoted from assistant to full professor increases median salary by between 43% to 66%.

We estimate that a doubling in years[3] (at current rank) increases median salary by between 3.3% and 9.1%.

It appears that after adjusting for these in our model variables the effect of gender is not significant, however other question may need to be asked about why so few females are in senior academic positions in the first place – it could be argued that more work needs to be done to determine the cause of the "gender gap".

[2] This is an example of backward selection - see STATS 330.

[3] Actually, it's a doubling in (years + 1) since 'log.yr = log(yr + 1)'

Addendum - what happened to the gender gap?

In a regression by itself (i.e., two sample t-test), we can show that gender is marginally significant — what could be driving this apparent effect?

```
table(salary.df$rk, salary.df$sx)
```

```
##
##             female male
##   assistant      8   10
##   associate      2   12
##   full           4   16
```

```
table(salary.df$dg, salary.df$sx)
```

```
##
##             female male
##   doctorate     10   24
##   masters        4   14
```

To really understand what is going on, we need to ask why there are so few females in the higher professorial ranks (relative to males).

Exercises

- Perform a two-sample t-test using only sex as the explanatory factor variable (this confirms that sex is marginally statistically significant if no other explanatories are included in the model.)
- Redo the analysis using forward selection (as in the Chapter notes). That is, build the model up by adding the most relevant terms in succession (provided they are significant). Do you end up with the same final model?
- Redo the analysis but using `yr` and `yd` rather than `log(yr)` and `log(yd)`. Does this make any meaningful difference?

Chapter 11: Linear models with a single factor explanatory variable having three or more levels (One-way analysis of variance)

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

- Fitting a model with a single explanatory factor variable with multiple levels, a.k.a., one-way analysis of variance (ANOVA)[1]
- Interpretting the fitted model
- Multiple pairwise comparisons of means
- Using `emmeans` to solve the multiple comparisons problem
- Relevant `R`-code
- Alternative parameterizations of the one-way ANOVA model[2]

[1] **NOTE:** When people use the term ANOVA (Analysis of Variance), they are referring to a linear regression model in which all the explanatory variable are factors.
[2] Optional section.

Section 11.1 Example with a 5-level explanatory factor variable

Example – Fruit flies

In this case study we look at how the male fruit-fly's longevity is related to his reproductive activity.

Fruit flies are a very commonly used animal for laboratory experiments because they are easy to maintain and breed. Their short lifespan allows several generations to be observed within a few months. They also have a genome that is very close to that of humans with many genes discovered in humans also found in fruit flies.[3]

Previous studies have shown that the longevity (life span) of female fruit flies decreases with an increase in reproduction, and this leads to a similar question related to males.

[3] See https://www.yourgenome.org/facts/why-use-the-fly-in-research for more background on research with fruit flies.

Fruit fly

The experiment compared the lifespan of males that were divided into 5 treatment groups that varied according to the presence or absence and number of uninterested or interested females.[4]

How does one define "interest" in female fruit flies? Here is this study's definition:

> Newly inseminated females will not usually mate again for at least two days. So, the males in the uninterested females treatment groups were always living with newly inseminated females.

The **primary focus of this Example** is the following: Due to the explanatory factor variable (treatment group) having several levels, we will need to apply an adjustment to relevant *P*-values and confidence intervals when we are making inference about differences in the expected lifespans between pairs of treatment groups.

[4] Had there been only two treatment groups then we could have used the two sample two-sample *t*-test discussed in Chapter 5

Fruit fly...

The response variable measured was `days`, the number of days the male fly lived.

The explanatory factor variable was `group`, with five levels:

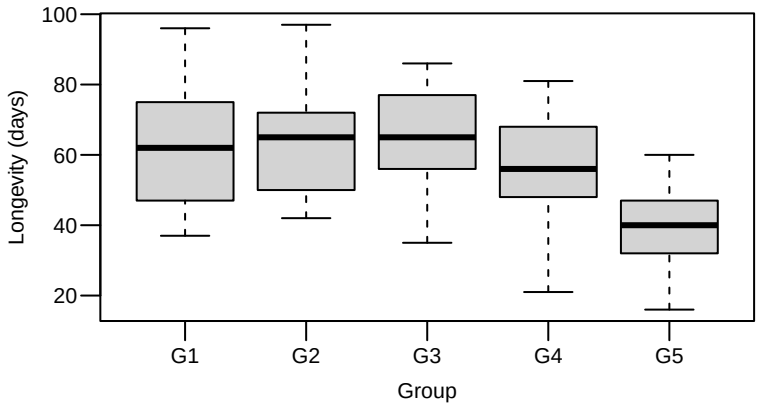
- `G1` males living alone,
- `G2` males living with one interested female,
- `G3` males living with eight interested females,
- `G4` males living with one uninterested female, and
- `G5` males living with eight uninterested females.

There were 25 male flies in each group, for a total sample size of 125.

Fruit fly...

Let us take a look at the data:

```
> Fruitfly.df = read.csv("Data/Fruitfly.csv", header=T)
> Fruitfly.df$group=factor(Fruitfly.df$group)
> boxplot(days ~ group, data = Fruitfly.df, ylab = "Longevity (days)")
```



It looks like male fruit flies do not live as long when in the presence of 'uninterested' females (`G5`), especially when there are several of them.

Fruit fly...

Linear model with multi-level (> 2) explanatory factor

As seen in previous chapters that involved categorical explanatory variables, our model specification uses indicator variables. In this case:

$$\texttt{days} = \beta_0 + \beta_1 \times \texttt{D2} + \beta_2 \times \texttt{D3} + \beta_3 \times \texttt{D4} + \beta_4 \times \texttt{D5} + \epsilon$$

where, as usual $\epsilon \overset{iid}{\sim} N(0, \sigma^2)$, and

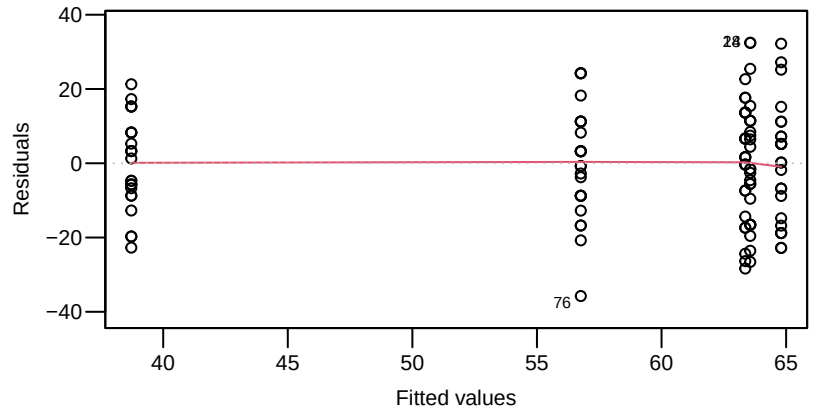
- `D2` is an indicator variable whereby `D2`=1 if the fruit fly is in group 2, otherwise it is 0.
- `D3` is an indicator variable whereby `D3`=1 if the fruit fly is in group 3, otherwise it is 0.
- ... and so on.

For example, β_1 and β_2 represent the differences in expected longevity (`days`) when we compare groups 2 and 3 to group 1 (the baseline).

Fruit fly...

Assumption checks

```
> Fruitfly.fit = lm(days ~ group, data = Fruitfly.df)
> plot(Fruitfly.fit, which=1)
```

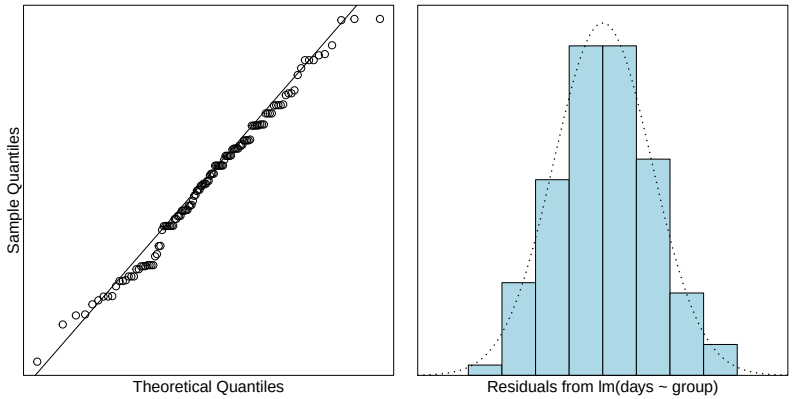


The **EOV** assumption seem to be okay.

Fruit fly...

Assumption checks...

```
> normcheck(Fruitfly.fit)
```

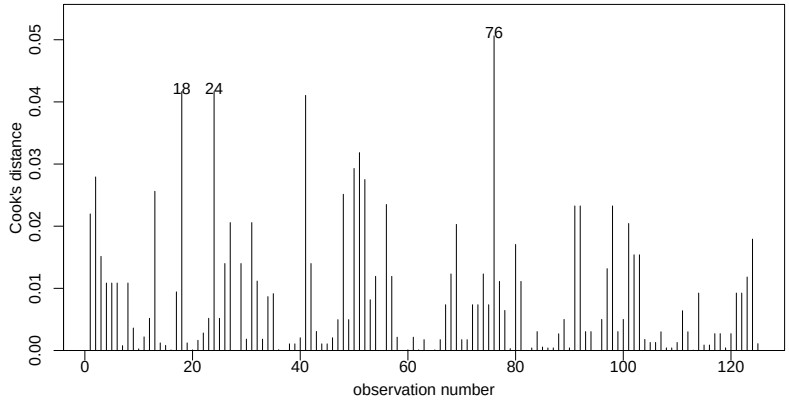


The normality assumption seem to be okay.

Fruit fly...

Assumption checks...

```
> cooks20x(Fruitfly.fit)
```



No unduly influential data points.

Fruit fly...

R^2 and ANOVA table

We can trust the fitted model. What can we conclude?[5]

```
> anova(Fruitfly.fit)
Analysis of Variance Table

Response: days
           Df Sum Sq Mean Sq F value    Pr(>F)
group       4  11939 2984.82  13.612 3.516e-09 ***
Residuals 120  26314  219.28
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This allows us to say that there is very strong evidence of a difference in expected longevity between the five groups, which was fairly obvious from the boxplot.

A significant result means we should now investigate how the groups differ from one another – there is more work to be done.

[5] Recall from Chapter 9 that we have to use the `anova` function to check the significance of a factor variable with more than two levels.

Section 11.2
Interpreting the output

Fruit fly...

Interpretation

Now we know that the variable `group` helps explain longevity, what can we say about these groups? Let us investigate.

```
> summary(Fruitfly.fit)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   63.560      2.962  21.461  < 2e-16 ***
groupG2        1.240      4.188   0.296    0.768
groupG3       -0.200      4.188  -0.048    0.962
groupG4       -6.800      4.188  -1.624    0.107
groupG5      -24.840      4.188  -5.931 2.98e-08 ***
---
Residual standard error: 14.81 on 120 degrees of freedom
Multiple R-squared:  0.3121,Adjusted R-squared:  0.2892
F-statistic: 13.61 on 4 and 120 DF,  p-value: 3.516e-09
```

Our fitted model only gives the pairwise difference between the baseline group `G1` and the other four groups, and so is only providing partial information. What else can we do?

Fruit fly...

Interpretation of grand and group means

Some researchers like to examine the group means and their deviations from the overall (or so-called "grand") mean.[6] These deviations are commonly called group "effects".

The estimated grand mean is simply the sample mean over all 125 male flies:

```
> grand.mean=mean(Fruitfly.df$days)
> grand.mean
[1] 57.44
```

The estimated group means are just the sample means within each group. We can quickly obtain these using the incredibly useful[7] `dplyr` package:

```
> library(dplyr())
> Df=Fruitfly.df %>% group_by(group) %>% summarize(group.mean=mean(days)) %>%
+   data.frame()
```

[6] See the optional final Section of this Chapter for more on this topic.

[7] `dplyr` and its associated packages are widely used for "data wrangling" (the process of cleaning and re-arranging data sets for easy access and analysis).

Fruit fly...

Interpretation of grand and group means

The above code groups the data by `group` and then applies the `mean` function to the `days` values within each group.

The estimated group means are

```
> Df$group.mean
[1] 63.56 64.80 63.36 56.76 38.72
```

and the estimated group effects are therefore

```
> Df$group.mean-grand.mean
[1]   6.12   7.36   5.92  -0.68 -18.72
```

We have seen that the overall average longevity of the 125 male flies in the study is about 57.4 days.

We also see that group `G5` has markedly lower longevity (18.72 fewer days) compared to the overall mean.

We could test null hypotheses and calculate confidence intervals for the above conclusions, but our focus on this course will be making inference about the differences in group means.

Fruit fly...

Pairwise comparisons

We'd really like to get the pairwise comparisons between every possible pair of groups. However, we've seen that the fitted model is restricted to examining how the groups G2–G5 differ from the baseline group G1.

If we wish to see how the other groups differed from group G2, say, then we could achieve this by changing the baseline group to group G2. Recall that this can be done using the relevel function:

```
> Fruitfly.df$newgroup = relevel(Fruitfly.df$group, ref="G2")
```

But to get all pairwise comparisons (i.e., G3 vs G4, G4 vs G5, ...) we have to do this re-leveling for G2, G3 and G4, and refit the model each time. This is too tedious.

We can get R to do the 'heavy lifting' for us by using the emmeans function from the R package of the same name. Moreover, emmeans solves the multiple comparisons problem that is discussed below.

Section 11.3
The multiple comparisons problem

Fruit fly...

Multiple comparisons

Note that when we are looking at all pair-wise comparisons of 5 groups, we have a total of 10 different possibilities:

G1 vs G2, G1 vs G3, G1 vs G4, G1 vs G5, (4 comparisons)
G2 vs G3, G2 vs G4, G2 vs G5, (3 comparisons)
G3 vs G4, G3 vs G5, (2 comparisons)
G4 vs G5, (1 comparisons).

In general, if there are m groups then there are ${}^{m}C_{2}$ possible pairwise comparisons.[8]

Each comparison requires a hypothesis test for a significant difference and an accompanying confidence interval. The multiple comparisons problem arises because, of all null hypotheses that are true, 5% are falsely rejected (Type 1 error). Equivalently, of all 95% confidence intervals, 5% of them do not contain the true parameter value.

[8] In R this is given by choose(m,2) and is the number of ways of choosing 2 objects from m objects. E.g., choose(5,2)=10.

Erroneous evidence of an effect from multiple testing

Multiple comparisons...

The following R code fits a simple linear regression model to iid (independent and identically distributed) normal data.

NOTE: The null hypothesis H_0 : slope $= 0$ is **true**.

```
> x = 1:30 ## Our explanatory variable
> x
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
[26] 26 27 28 29 30
> y = rnorm(30) ## y has NO relationship with x
> summary(lm(y~x))$coef ## Print only the coefficient table
             Estimate Std. Error   t value   Pr(>|t|)
(Intercept)  0.4945317  0.3884378  1.273130 0.21344114
x           -0.0413633  0.0218802 -1.890444 0.06908972
```

If this code is run many times over, then approximately 5% of the time the slope will have P-value < 0.05.[9]

That is, there will be erroneous evidence of an effect of x (i.e., evidence for a non-zero slope) about 1 time in 20!

[9] In fact, it can be shown that the P-value is uniformly distributed between 0 and 1 when H_0 is true.

Erroneous evidence of an effect from multiple testing...

Multiple comparisons...

When we do multiple tests (i.e., the 10 paired comparisons in this example) then we greatly increase the probability of obtaining at least one erroneous conclusion[10].

This is known as the multiple comparison problem. It essentially says that if you look at enough things you will find something 'happening', even when there's nothing going on.

Remember, data always have variability, and if we are not careful we can 'discover' false structure that is not really there.

So, when we look at these 10 comparisons we need to adjust so that the overall error rate (the probability of any spurious significance) over all 10 comparison is no more the 5%. This can be done using a Tukey adjustment.

[10] Assuming independent comparisons, if we do 10 95% CIs we have an overall error rate of $1 - (1 - .05)^{10} = 40\%$, which is much higher than our original 5% error rate per comparison.

Example—Fruit fly

Tukey simultaneous confidence intervals

Let's get *simultaneous* 95% confidence intervals for all 10 comparisons via the `pairs` and `emmeans` functions of the `emmeans` package.[11]

```
> library(emmeans)
> Fruitfly.pairs = pairs(emmeans(Fruitfly.fit, ~group, infer=T))
> Fruitfly.pairs
 contrast estimate   SE  df t.ratio p.value
 G1 - G2     -1.24 4.19 120  -0.296  0.9983
 G1 - G3      0.20 4.19 120   0.048  1.0000
 G1 - G4      6.80 4.19 120   1.624  0.4854
 G1 - G5     24.84 4.19 120   5.931  <.0001
 G2 - G3      1.44 4.19 120   0.344  0.9970
 G2 - G4      8.04 4.19 120   1.920  0.3127
 G2 - G5     26.08 4.19 120   6.227  <.0001
 G3 - G4      6.60 4.19 120   1.576  0.5158
 G3 - G5     24.64 4.19 120   5.883  <.0001
 G4 - G5     18.04 4.19 120   4.307  0.0003

P value adjustment: tukey method for comparing a family of 5 estimates
```

[11] These confidence intervals are called "simultaneous" since we can be 95% confident that **they all** contain the true group difference simultaneously.

Fruit fly

Tukey simultaneous confidence intervals...

We see that the majority of these pairwise comparisons are not significantly different. Let's extract only the CIs where the Tukey adjusted *P*-values are less than 0.05.

```
>   Fruitfly.pairs=data.frame(Fruitfly.pairs)
>   ## Which pairwise comparisons have a P-value less than 0.05?
>   mc.signif = Fruitfly.pairs[,"p.value"] < 0.05
>   mc.signif
 [1] FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE
>   ## Print comparisons which have a P-value less than 0.05
>   print(Fruitfly.pairs[mc.signif, ], digits = 4)
   contrast estimate    SE  df t.ratio   p.value
4   G1 - G5    24.84 4.188 120   5.931 2.958e-07
7   G2 - G5    26.08 4.188 120   6.227 7.232e-08
9   G3 - G5    24.64 4.188 120   5.883 3.701e-07
10  G4 - G5    18.04 4.188 120   4.307 3.240e-04
```

Note the use of the `data.frame` function in the above code. We needed to convert `Fruitfly.pairs` to a dataframe before we could take the subset of rows, otherwise `emmeans` gets confused and thinks we are doing a smaller number of pairwise comparisons.

Fruit fly...

Some conclusions:

- Our model explains 31% of variability in fruit fly longevity.
- We see that group 5 (males with 8 uninterested females) is different from all the others.

On average, group 5 males live fewer days than:

- Group 1 (males living alone) by 13 to 36 fewer days.
- Group 2 (males living with one interested female) by 14 to 38 fewer days.
- Group 3 (males living with eight interested females) by 13 to 36 fewer days.
- Group 4 (males living with one uninterested female) by 6 to 30 fewer days.

Fruit fly...

On a lighter note there is little evidence of a difference in longevity if no females or only one uninterested female is about, or if females are there and 'interested' in them — but in the presence of multiple uninterested females they die earlier (they 'drop like flies').

Recall also that in the other studies it was seen that females did not live as long if they reproduced, which can be attributed to the physical demands of producing and laying eggs. With males, perhaps it is sexual frustration that is killing them!

For more on this topic see the research article written by Branco et al. (2017, Reproductive activity triggers accelerated male mortality and decreases lifespan: genetic and gene expression determinants in Drosophila. Heredity 118, 221-228 https://doi.org/10.1038/hdy.2016.89) at https://www.nature.com/articles/hdy201689.

Section 11.4
Closing remarks and relevant R-code

Understanding the anova function output

```
> anova(Fruitfly.fit)
Analysis of Variance Table

Response: days
           Df Sum Sq Mean Sq F value    Pr(>F)
group       4  11939 2984.82  13.612 3.516e-09 ***
Residuals 120  26314  219.28
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In the above output we see that the variability we observe in our longevity data can be broken down into two components `group` and `residual`.

The amount of variability that the variable `group` (as shown in the `Sum Sq` column) explains is 11939. The residual variability (left over) is 26314. The total variability is $11939 + 26314 = 38253$. The % of variability explained by `group` is therefore

$$100 \times \left(\frac{11939}{11939 + 26314}\right) = 100 \times \left(1 - \frac{26314}{11939 + 26314}\right) = 31\%.$$

Note that we have just calculated the R^2 – the proportion of the variability in the response variable that is explained by the explanatory variables, 0.31.

Most of the R-code you need for this chapter

Use box plots to inspect the data for each level of the factor.

```
> boxplot(days ~ group, data = Fruitfly.df)
```

You do not need to create indicator variables - R does that for you. The baseline can be changed if you wish by using the `relevel` function.

```
> Fruitfly.df$newgroup = relevel(Fruitfly.df$group, ref="G2")
```

Fit the model and use the ANOVA table to see if any of the means differ from one another (regardless of the baseline chosen).

```
> anova(Fruitfly.fit)
```

Adjust confidence intervals for multiple pairwise comparisons by using the Tukey adjustment to obtain simultaneous intervals CIs:

```
> Fruitfly.pairs = pairs(emmeans(Fruitfly.fit, ~group, infer=T))
```

Section 11.5
Alternative parameterizations of the 1-way ANOVA model

(This is an optional Section:
- your lecturer will advise whether it is examinable)

Alternative parameterizations of the 1-way ANOVA model

The reference cell model

Recall the linear model[12] we used to represent the longevity, in days, of a male fruitfly, i.e.

$$\texttt{days} = \beta_0 + \beta_1 \times \texttt{D2} + \beta_2 \times \texttt{D3} + \beta_3 \times \texttt{D4} + \beta_4 \times \texttt{D5} + \epsilon$$

The *parameters* $\beta_0, \beta_1, \ldots, \beta_4$ denote the true values of some attribute (e.g. longevity) of the population of male fruitflies. Here, β_0 represents the mean longevity of male fruitflies in group `G1`. The parameters $\beta_1, \ldots, \beta_4$ represent the deviations in mean longevity of males in groups `G2,...,G5`, respectively, from the mean longevity of males in group `G1`.

The values in the `Estimate` column of the regression summary table[13] result in the following equation for predicted longevity:

$$\widehat{\texttt{days}} = 63.56 + 1.24 \times \texttt{D2} + (-0.20) \times \texttt{D3} + (-6.80) \times \texttt{D4} + (-24.84) \times \texttt{D5}$$

[12] See slide 8.
[13] See slide 14; Coefficients rounded to 2 decimal places.

Alternative parameterizations of the linear model

The reference cell model

Each cell within a column in the table below corresponds to a level of the `Group` factor. One way to 'parametrise' these cells is to use means, i.e. $\mu_1, \mu_2, \ldots, \mu_5$. Another is to select one of the cells as a reference cell (here `Group G1`) and the remaining cells are then parametrised the deviations of the current row's group mean from the reference cell's group mean.

		parameterization			
Group	Data	Means	Estimate[14]	Reference cell	Estimate[15]
G1	$40, 37, \ldots, 44$	μ_1	63.56	$\beta_0 = \mu_1$	63.56
G2	$46, 42, \ldots, 92$	μ_2	64.80	$\beta_1 = \mu_2 - \mu_1$	1.24
G3	$35, 37, \ldots, 77$	μ_3	63.36	$\beta_2 = \mu_3 - \mu_1$	−0.20
G4	$21, 40, \ldots, 68$	μ_4	56.76	$\beta_3 = \mu_4 - \mu_1$	−6.80
G5	$16, 19, \ldots, 44$	μ_5	38.72	$\beta_4 = \mu_5 - \mu_1$	−24.84

The parameterization of the model shown on the previous slide is therefore known as the *reference cell* model.

[14] See estimates of `Group` means on slide 16
[15] See regression coefficients table on slide14

Alternative parameterizations of the linear model

The means model

From the above table we can see that there is an alternative, but equivalent, *means* model parameterization, i.e. linear model for the longevity of the jth ($j = 1, 2, \ldots, 25$) male fruitfly in `Group` i ($i = 1, 2, \ldots, 5$) may be written as

$$days_{ij} = \mu_i + \epsilon_{ij}$$

where μ_i denotes the mean longevity, in days, of a male in `Group` i and, as usual, $\epsilon_{ij} \overset{iid}{\sim} N(0, \sigma^2)$.

Alternative parameterizations of the linear model

The effects model

Another parameterization is to set the overall mean longevity, μ, as the reference and then define the *effect*, τ_i, on longevity due to being in `Group` i as the difference between the `Group` i mean and the overall mean, i.e. $\tau_i = \mu_i - \mu$.

		parameterization			
Group	Data	Means	Estimate	Effects	Estimate[16]
G1	$40, 37, \ldots, 44$	μ_1	63.56	$\tau_1 = \mu_1 - \mu$	6.12
G2	$46, 42, \ldots, 92$	μ_2	64.80	$\tau_2 = \mu_2 - \mu$	7.36
G3	$35, 37, \ldots, 77$	μ_3	63.36	$\tau_3 = \mu_3 - \mu$	5.92
G4	$21, 40, \ldots, 68$	μ_4	56.76	$\tau_4 = \mu_4 - \mu$	-0.68
G5	$16, 19, \ldots, 44$	μ_5	38.72	$\tau_5 = \mu_5 - \mu$	-18.72

The linear *effects* model for the longevity of the jth ($j = 1, 2, \ldots, 25$) male fruitfly in `Group` i ($i = 1, 2, \ldots, 5$) may therefore be written as

$$days_{ij} = \mu + \tau_i + \epsilon_{ij}$$

where, again, $\epsilon_{ij} \overset{iid}{\sim} N(0, \sigma^2)$.

[16] See overall mean (57.44 days) and deviations of group means from overall means on slide 16.

Case Study 11.1: Fruit-flies, sex and frustration

Tou Ohone Andate - staff number 1234567

Problem:

In this study, we look at how the male fruit-flies longevity is related to reproductive activity. (Data is from http://www.cvgs.k12.va.us:81/digstats/Imain.html)

How does one define "interest" in fruit-flies? Here is this study's definition:

> Newly inseminated females will not usually mate again for at least two days.

So the males in the uninterested groups were always living with newly inseminated females!

The hypothesis was that the males living alone and with the uninterested females would live longer than the males living with the interested females. Since there are more than two group means, a one-way ANOVA is used to determine if there is a significant difference between the group means.

The design of the study placed male fruit-flies in the following groups:

1) Males living alone,
2) Males living with one interested female,
3) Males living with eight interested females,
4) Males living with one uninterested female, and
5) Males living with eight uninterested females.

The variables of interest were:

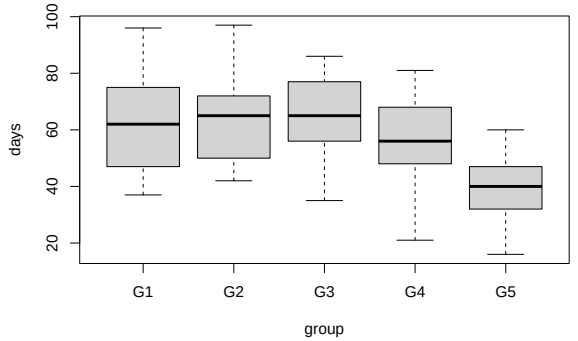
- `days`: A male fruitfly's longevity in days.
- `group`: A five-level factor with the levels which corresponds to the group a male fruitfly is in:
  - "G1": males living alone,
  - "G2": males with one interested female,
  - "G3": males with eight interested females,
  - "G4": males with one uninterested female,
  - "G5": males with eight uninterested females,

Question of Interest

How does sexual activity affect male fruitfly longevity?

Read in and Inspect the Data


```
Fruitfly.df = read.csv("Fruitfly.csv", stringsAsFactors=TRUE)
plot(days ~ group, data = Fruitfly.df)
```



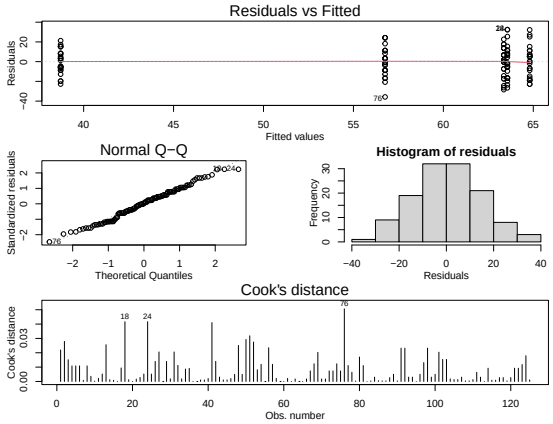

```
summaryStats(days ~ group, Fruitfly.df)
```

```
##    Sample Size  Mean Median  Std Dev Midspread
## G1          25 63.56     62 16.45215        28
## G2          25 64.80     65 15.65248        22
## G3          25 63.36     65 14.53983        21
## G4          25 56.76     56 14.92838        20
## G5          25 38.72     40 12.10207        15
```

Our hypothesis that the males living alone and with the uninterested females would live longer than the males living with the interested females does not seem plausible with our data.

Model Building and Check Assumptions

```
ff.fit = lm(days ~ group, data = Fruitfly.df)
modelcheck(ff.fit)
```




```
anova(ff.fit)
```

```
## Analysis of Variance Table
##
## Response: days
##            Df Sum Sq Mean Sq F value    Pr(>F)
## group       4  11939 2984.82  13.612 3.516e-09 ***
## Residuals 120  26314  219.28
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(ff.fit)
```

```
##
## Call:
## lm(formula = days ~ group, data = Fruitfly.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -35.76  -8.76   0.20  11.20  32.44
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   63.560      2.962  21.461  < 2e-16 ***
## groupG2        1.240      4.188   0.296    0.768
## groupG3       -0.200      4.188  -0.048    0.962
## groupG4       -6.800      4.188  -1.624    0.107
## groupG5      -24.840      4.188  -5.931 2.98e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```



```
## Residual standard error: 14.81 on 120 degrees of freedom
## Multiple R-squared:  0.3121, Adjusted R-squared:  0.2892
## F-statistic: 13.61 on 4 and 120 DF,  p-value: 3.516e-09
```

Multiple Comparisons Output

```
library(emmeans)
Fruitfly.emm = emmeans(ff.fit, ~group)
# View all pairwise comparisons:
pairs(Fruitfly.emm, infer=TRUE)
```

```
##  contrast estimate   SE  df lower.CL upper.CL t.ratio p.value
##  G1 - G2     -1.24 4.19 120   -12.84     10.4  -0.296  0.9983
##  G1 - G3      0.20 4.19 120   -11.40     11.8   0.048  1.0000
##  G1 - G4      6.80 4.19 120    -4.80     18.4   1.624  0.4854
##  G1 - G5     24.84 4.19 120    13.24     36.4   5.931  <.0001
##  G2 - G3      1.44 4.19 120   -10.16     13.0   0.344  0.9970
##  G2 - G4      8.04 4.19 120    -3.56     19.6   1.920  0.3127
##  G2 - G5     26.08 4.19 120    14.48     37.7   6.227  <.0001
##  G3 - G4      6.60 4.19 120    -5.00     18.2   1.576  0.5158
##  G3 - G5     24.64 4.19 120    13.04     36.2   5.883  <.0001
##  G4 - G5     18.04 4.19 120     6.44     29.6   4.307  0.0003
##
## Confidence level used: 0.95
## Conf-level adjustment: tukey method for comparing a family of 5 estimates
## P value adjustment: tukey method for comparing a family of 5 estimates
```

```
# View only the comparisons that are significant at the 5% level:
Fruitfly.pairs = data.frame(pairs(Fruitfly.emm, infer=T))
subset(Fruitfly.pairs, p.value<0.05)
```

```
##    contrast estimate       SE  df  lower.CL upper.CL  t.ratio      p.value
## 4   G1 - G5    24.84 4.188358 120 13.239532 36.44047 5.930724 2.958395e-07
## 7   G2 - G5    26.08 4.188358 120 14.479532 37.68047 6.226783 7.232306e-08
## 9   G3 - G5    24.64 4.188358 120 13.039532 36.24047 5.882973 3.701044e-07
## 10  G4 - G5    18.04 4.188358 120  6.439532 29.64047 4.307177 3.240300e-04
```

Methods and Assumption Checks

The boxplot of `days` by `group` indicated that males living with 8 uninterested females have shorter lives compared to their counterparts in other groups. So, we fitted a One-way ANOVA model to these data.

The model assumptions seem satisfied.

Our final model is

$$\text{days}_i = \beta_0 + \beta_1 \times Group2_i + \beta_2 \times Group3_i + \beta_3 \times Group4_i + \beta_4 \times Group5_i + \epsilon_i,$$

where $GroupX_i$ is 1 if the ith male fruitfly is in group X and 0 otherwise, and $\epsilon_i \sim iidN(0, \sigma^2)$.

Alternatively, our final model could be written as

$$\text{days}_{ij} = \mu + \alpha_i + \epsilon_{ij},$$

where μ is the overall mean survival time and α_i is the effect of being in the ith group and $\epsilon_{ij} \sim iid\ N(0, \sigma^2)$.

Our model explained 31% of variability in male fruitfly longevity.

Executive Summary

Researchers were interested in how sexual activity affects male fruitfly longevity.

We see that the effect of Group 5, males with 8 uninterested females, seems markedly different from all the others.

In particular group 5 males, on average, lived fewer days than:

- Group 1 males (living alone) by between 13 to 36 fewer days.
- Group 2 males (living with one interested female) by between 14 to 38 fewer days.
- Group 3 males (living with eight interested females) by between 13 to 36 fewer days.
- Group 4 males (living with one uninterested female) by between 6 to 30 fewer days.

On a lighter note these male fruit flies are fine if no females are about or if they are there they need to be 'interested' in them — otherwise they die earlier (they 'drop like flies'). It's tempting to make similar inference about the human species but that may be going too far!

Case Study 11.2: Exam vs Degree

Tou Ohone Andate - staff number 1234567

Problem

We want to quantify the expected final exam mark (out of 100) in Stats 20x for each type of degree. In particular, we want to investigate whether there is a "degree" effect on the final exam mark.

The variables of interest were:

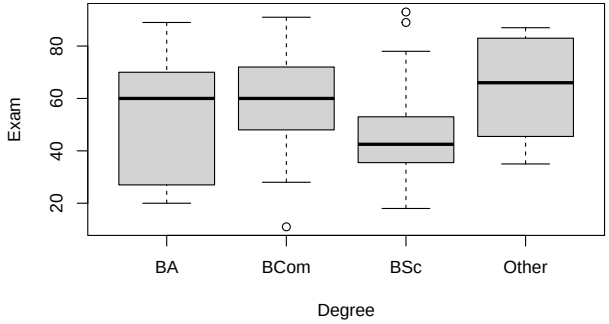
- `Exam`: A student's exam mark out of 100.
- `Degree`: A four-level factor with levels corresponding to a student's degree.
  - "BA", "BCom", "BSc", and "Other".

Question of Interest

Is the degree a student is enrolled for related to their final 20x exam score?

Read in and Inspect the Data

```
Stats20x.df = read.table("STATS20x.txt", header = T)
Stats20x.df$Degree=factor(Stats20x.df$Degree)
#Draw boxplot
plot(Exam ~ Degree, data = Stats20x.df)
```



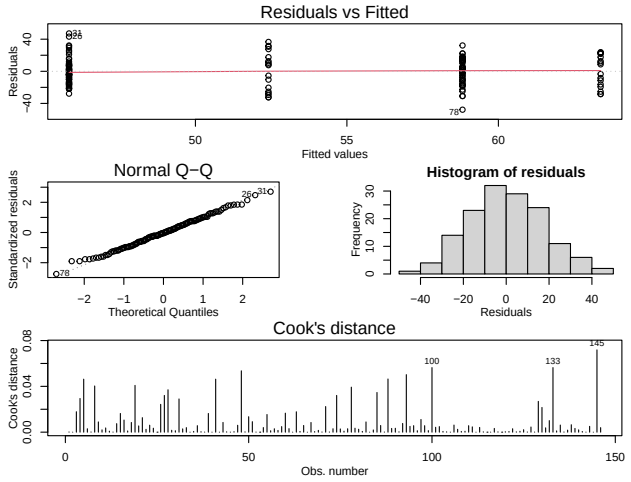

```
#Summary stats:
summaryStats(Exam ~ Degree, Stats20x.df)
```

```
##        Sample Size     Mean Median  Std Dev Midspread
## BA              17 52.41176   60.0 24.57402     43.00
## BCom            49 58.81633   60.0 16.23868     24.00
## BSc             64 45.82812   42.5 15.80090     17.25
## Other           16 63.37500   66.0 19.76824     35.75
```

The "BSc" group is centred noticeably lower than the others. The standard deviations are within a factor of two from smallest to largest, so we can accept the equality of variance assumption. (The midspreads do exceed the factor-of-two rule-of-thumb, so we might need to be cautious in our interpretations.)

Model Building and Check Assumptions

```
degree.fit = lm(Exam ~ Degree, data = Stats20x.df)
modelcheck(degree.fit)
```




```
anova(degree.fit)
```

```
## Analysis of Variance Table
##
## Response: Exam
##            Df Sum Sq Mean Sq F value    Pr(>F)
## Degree      3   6675 2225.15  7.1958 0.0001568 ***
```

```
## Residuals 142  43910  309.23
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(degree.fit)
```

```
##
## Call:
## lm(formula = Exam ~ Degree, data = Stats20x.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.816 -12.456  -0.816  12.487  47.172
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   52.412      4.265  12.289   <2e-16 ***
## DegreeBCom     6.405      4.950   1.294   0.1978
## DegreeBSc     -6.584      4.798  -1.372   0.1722
## DegreeOther   10.963      6.125   1.790   0.0756 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.58 on 142 degrees of freedom
## Multiple R-squared:  0.132,  Adjusted R-squared:  0.1136
## F-statistic: 7.196 on 3 and 142 DF,  p-value: 0.0001568
```

Multiple Comparisons Output

```
pairs(emmeans(degree.fit, ~Degree), infer=T)
```

```
##  contrast     estimate   SE  df lower.CL upper.CL t.ratio p.value
##  BA - BCom       -6.41 4.95 142   -19.27     6.46  -1.294  0.5683
##  BA - BSc         6.58 4.80 142    -5.89    19.06   1.372  0.5189
##  BA - Other     -10.96 6.12 142   -26.89     4.96  -1.790  0.2825
##  BCom - BSc      12.99 3.34 142     4.31    21.67   3.891  0.0009
##  BCom - Other    -4.56 5.06 142   -17.72     8.61  -0.900  0.8047
##  BSc - Other    -17.55 4.92 142   -30.32    -4.77  -3.570  0.0027
##
## Confidence level used: 0.95
## Conf-level adjustment: tukey method for comparing a family of 4 estimates
## P value adjustment: tukey method for comparing a family of 4 estimates
```

Methods and Assumption Checks

We wish to explain exam marks using degree, a factor with four levels, so we fitted a One-way ANOVA model to these data.

The model assumptions seem satisfied.

Our final model is

$$\text{Exam}_i = \beta_0 + \beta_1 \times Degree.BCom_i + \beta_2 \times Degree.BSc_i + \beta_3 \times Degree.Other_i + \epsilon_i,$$

where $Degree.x_i$ is 1 if a student is enrolled in degree x and 0 otherwise (with $x \in \{\text{BCom}, \text{BSc}, \text{Other}\}$), and $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Alternatively, our final model could be written as

$$\text{Exam}_{ij} = \mu + \alpha_i + \epsilon_{ij},$$

where μ is the overall mean exam mark and α_i is the effect of being in the ith degree (with $i \in \{\text{BA}, \text{BCom}, \text{BSc}, \text{Other}\}$), and $\epsilon_{ij} \sim iid\ N(0, \sigma^2)$.

Our model explained 13.2% of the variability in students' exam marks.

Executive Summary

Is the degree a student is enrolled in related to their final 20x exam mark?

We do have evidence that expected exam marks were not identical between the four degree groups (Ba, BCom, BSc, and Other). However, the only significant differences we found were that BSc students had lower marks than BCom and Other degree students.

With 95% confidence we can say that:

- on average, "BSc" students do worse than "BCom" students by between 4 and 22 marks.
- on average, "BSc" students do worse than "Other" students by between 5 and 30 marks.

Chapter 12: Linear models with two explanatory factor variables (Two-way analysis of variance)

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

- Fitting a model with two explanatory factors, a.k.a., two-way ANOVA
- Interaction plots
- Interpretting the fitted model
- Pairwise comparisons using `emmeans`
- Simplifying the model where possible
- Relevant R-code
- Alternative parameterizations of the two-way ANOVA model[1]

[1] Optional section.

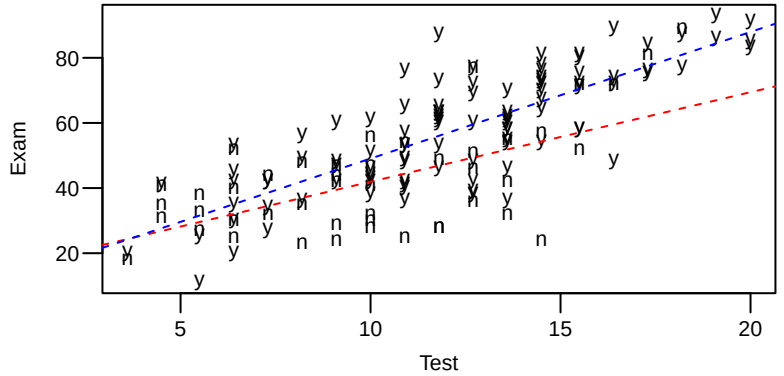
Section 12.1
Example: Using test success and attendance to explain exam score

Two-way ANOVA with interaction

Exam score vs test success and attendance

In Chapter 8 we investigated whether the effect of a student's test mark on exam score changed depending on whether they regularly attend or not.

We saw that those who attended regularly (blue line and "y" for "yes") got more 'return' for each additional test mark than non-attenders.



Exam score vs test success and attendance...

Here we will use the same two explanatory variables as in Chapter 8 but are going to change the explanatory test score variable so that it only has two states – passed or did not pass.

That is, we are going to use the dichotomous factor variable "test success", rather than the raw numeric test score value.

We shall also be using attendance as a second explanatory factor.

For this example we shall use a two-way ANOVA, since there are two explanatory factors.

First, read in the data and change the class of `Attend` to factor:

```
> ## Importing data into R
> Stats20x.df = read.table("Data/STATS20x.txt", header=T)
> Stats20x.df$Attend=factor(Stats20x.df$Attend)
```


Exam score vs test success and attendance...

We next transform the numeric `Test` variable into a factor with two levels, `pass` and `nopass`.

Let us create the new factor variable `Pass.test`:

```
> Stats20x.df$Pass.test=with(Stats20x.df,
+                            factor(ifelse(Test>=10,"pass","nopass")))
> ## Check to see if the call above does what we expect
> min(Stats20x.df$Test[Stats20x.df$Pass.test=="pass"])
[1] 10
> max(Stats20x.df$Test[Stats20x.df$Pass.test=="nopass"])
[1] 9.1
```

We can now examine whether passing the test results in better exam marks and vice-versa, on average. We can also ask the same question of regular attendance.

Section 12.2
Interaction plots

Exam score vs test success and attendance...

`interactionPlots()`

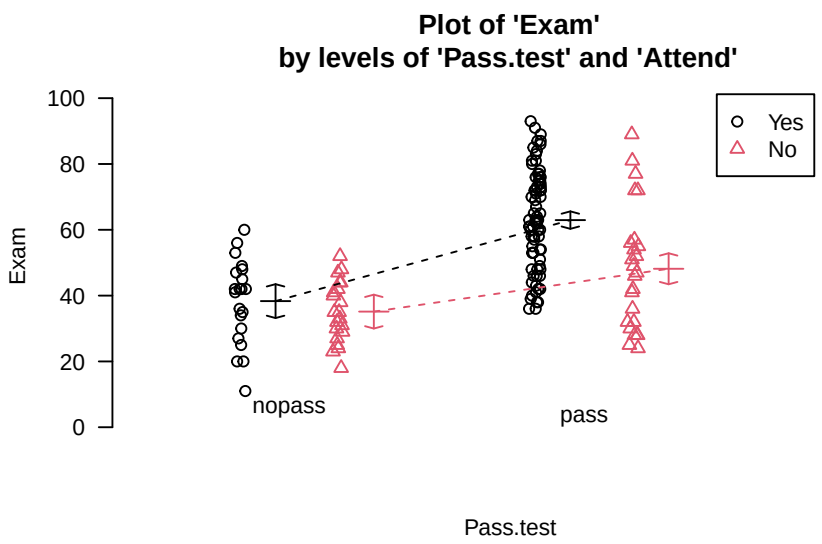
Let us see how these data explain `Exam` by using an `s20x` function `interactionPlots()`.

This is designed specifically for plotting a continuous Y (in our case `Exam`) against two factor variables (here they are `Attend` and the newly created `Pass.test`).

Exam vs test success and attendance...

`interactionPlots()`...

```
> interactionPlots(Exam ~ Pass.test + Attend, data = Stats20x.df)
```



Exam vs test success and attendance...

Here we see that 'attenders' who pass the test seem to be doing markedly better than most other students. Note that we do not have parallel lines, thereby indicating that there could be an interaction between the two factors.

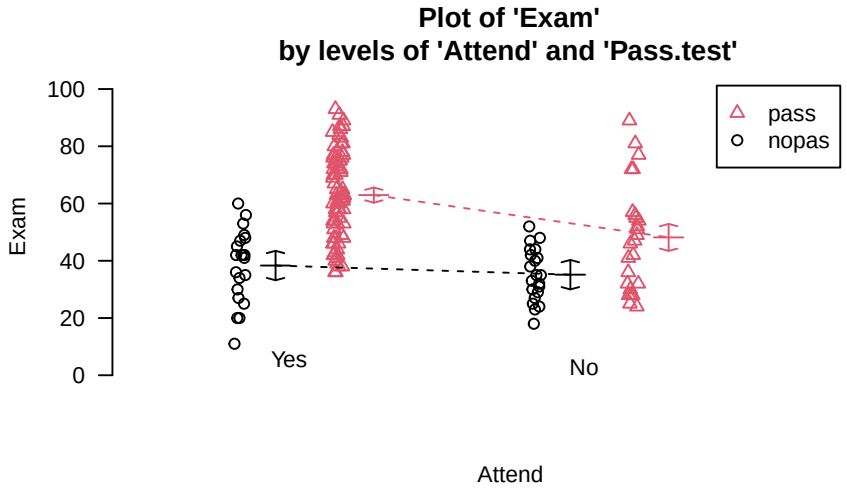
In other words, the effect on exam score of passing the test may depend on whether a student regularly attended or not.

As shown below, we can rearrange the layout of the interaction plot by reversing the order in which the explanatory variables are given in the right-hand side of the model formula argument.

Exam vs test success and attendance...

We still conclude the same insights as above.

```
> interactionPlots(Exam ~ Attend + Pass.test, data = Stats20x.df)
```



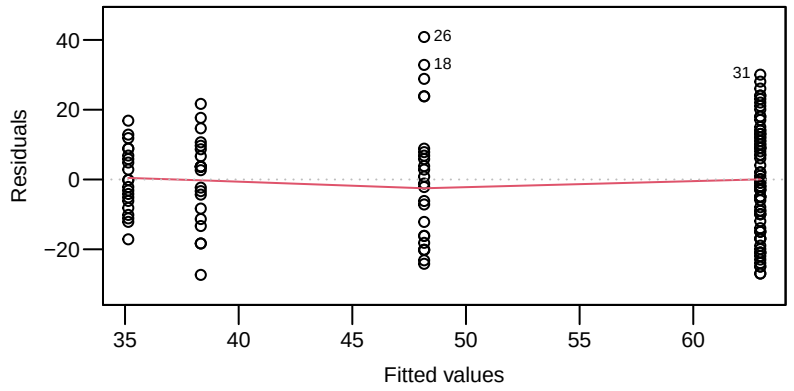
Section 12.3
Fitting the interaction model

Exam vs test success and attendance. . .

Assumption checks

Let us fit the model with interaction, and check the assumptions.

```
> Exam.fit = lm(Exam ~ Attend*Pass.test, data = Stats20x.df)
> plot(Exam.fit, which=1)
```

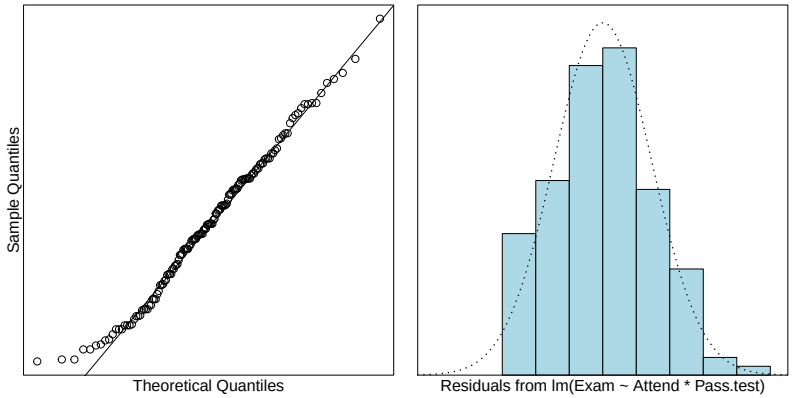


The **EOV** assumption seems to be okay.

Exam vs test success and attendance. . .

Assumption checks. . .

```
> normcheck(Exam.fit)
```

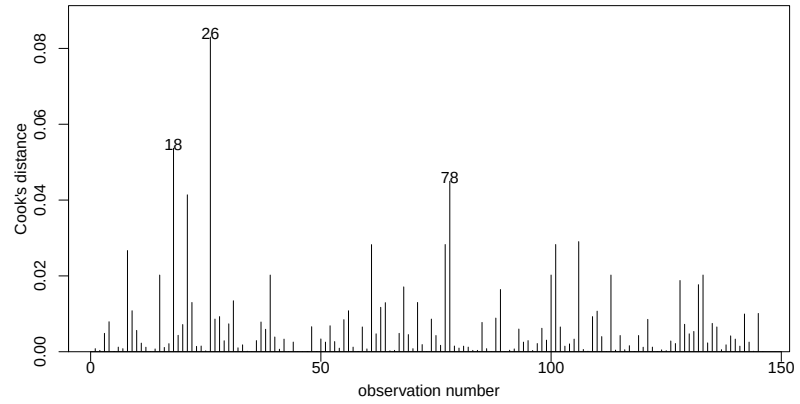


The normality assumption seems to be reasonably good, other than a lack of large negative residuals.

Exam vs test success and attendance. . .

Assumption checks. . .

```
> cooks20x(Exam.fit)
```



No unduly influential data points.

Exam vs test success and attendance. . .

We conclude that we can trust the output. Let us see what it is telling us.

```
> anova(Exam.fit)
Analysis of Variance Table

Response: Exam
                Df  Sum Sq Mean Sq F value    Pr(>F)
Attend           1  7630.8  7630.8  34.990 2.364e-08 ***
Pass.test        1 11076.9 11076.9  50.791 4.763e-11 ***
Attend:Pass.test 1   909.7   909.7   4.171   0.04297 *
Residuals      142 30968.4   218.1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The P-value of 0.043 is just under 0.05 and so establishes that there is a significant interaction: The effect of passing the test depends on whether the student has regularly attended lectures or not.

So, we cannot simply state the effect of passing the test, because the size of this effect depends on whether the student attended or not.

One way to think of this is that we have to consider all 4 (2×2) different test success/attendance possibilities separately.

Exam vs test success and attendance...

Let us investigate what our model tells us in terms of the estimated parameters:

```
> summary(Exam.fit)
```

```
Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)            35.143      3.223  10.905  < 2e-16 ***
AttendYes               3.190      4.557   0.700  0.48504
Pass.testpass          13.017      4.371   2.978  0.00341 **
AttendYes:Pass.testpass   11.599      5.679   2.042  0.04297 *
---
Residual standard error: 14.77 on 142 degrees of freedom
Multiple R-squared:  0.3878,Adjusted R-squared:  0.3749
F-statistic: 29.98 on 3 and 142 DF,  p-value: 4.452e-15
```

The P-value for interaction is the same as before.

Note also that the $R^2 = 39\%$ can be obtained from the ANOVA table above as follows: $R^2 = 100 \times \left(1 - \frac{30968}{30968+910+11077+7631}\right)$ is the proportion of variability that is explained by our model terms.

Exam vs test success and attendance...

The formula for the above two-way ANOVA can be written as:

$$\texttt{Exam} = \beta_0 + \beta_1 \times \texttt{Attend}_{\texttt{Yes}} + \beta_2 \times \texttt{Pass.test}_{\texttt{pass}} + \beta_3 \times \texttt{Attend}_{\texttt{Yes}} \times \texttt{Pass.test}_{\texttt{pass}} + \varepsilon$$

where $\texttt{Attend}_{\texttt{Yes}}$ and $\texttt{Pass.test}_{\texttt{pass}}$ are indicator variables, and $\varepsilon \overset{iid}{\sim} N(0, \sigma^2)$.

This model is relative to the baseline levels of `Attend` and `Pass.test`. These baselines are `No` and `nopass`, respectively, since they are the levels with the lowest alphanumeric value.

Exam vs test success and attendance...

Alternative parameterizations: the group means model

As seen in the previous chapter, another option is to remove the baseline with the addition of -1 in the model formula. One other complication is that we have to use `:` rather than `*` when specifying the interaction term.

```
> Exam.fitNoBaseline=lm(Exam~Attend:Pass.test-1,data=Stats20x.df)
> coef(summary(Exam.fitNoBaseline))
                         Estimate Std. Error  t value     Pr(>|t|)
AttendNo:Pass.testnopass  35.14286   3.222594 10.90515 1.707299e-20
AttendYes:Pass.testnopass 38.33333   3.222594 11.89518 4.515124e-23
AttendNo:Pass.testpass    48.16000   2.953556 16.30577 2.405316e-34
AttendYes:Pass.testpass   62.94937   1.661505 37.88696 4.012344e-76
```

This is the group means model. It is simply giving the estimated group mean for each of the four combinations of attendance and test success.

Exam vs test success and attendance...

Alternative parameterizations: the means and effects model

A further alternative is to use the means and effects model formula for the two-way ANOVA:

$$\texttt{Exam}_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}, \text{ where } \epsilon_{ijk} \overset{iid}{\sim} N(0, \sigma^2)$$

where:

- μ is the overall mean,
- α_is are the `Attend` effects relative to the overall mean,
- β_js are the `Pass.test` effects relative to the overall mean,
- γ_{ij}s are the interaction effects between levels of `Attend` and `Pass.test` relative to the overall mean.

This is simply a different parameterization of the two-way ANOVA model. That is, there is no change to the model, but just in the way we choose to write it.

In this course for Executive Summaries we will typically be interested in estimating relevant pairwise group differences, for which we will once again use `emmeans`.

Section 12.4
Interpretting the output using pairwise differences

Exam vs test success and attendance

Interpreting the output

In studies in which *all* of the explanatory variables are factors, our interest typically lies in making statistical inferences about the sizes of pairwise differences between the means of different treatment combinations.

We could calculate these pairwise differences from the above output, but what we really need is Tukey-adjusted confidence intervals for the differences. These are needed because of the multi-comparisons problem. Once again, we will use the `emmeans()` function to perform these calculations for us.

In the field of statistics, pairwise comparisons are often called *contrasts*[2] because they are comparing (i.e., contrasting) two different means.

[2] The term *contrasts* includes other more complicated forms of comparisons.

Exam vs test success and attendance

Pairwise comparisons

```
> library(emmeans)
> exam.pairs = pairs(emmeans(Exam.fit, ~ Attend*Pass.test), infer=T)
> exam.pairs
 contrast               estimate   SE  df lower.CL upper.CL t.ratio p.value
 No nopass - Yes nopass    -3.19 4.56 142    -15.0     8.66  -0.700  0.8969
 No nopass - No pass      -13.02 4.37 142    -24.4    -1.65  -2.978  0.0178
 No nopass - Yes pass     -27.81 3.63 142    -37.2   -18.38  -7.669  <.0001
 Yes nopass - No pass      -9.83 4.37 142    -21.2     1.54  -2.248  0.1155
 Yes nopass - Yes pass    -24.62 3.63 142    -34.0   -15.19  -6.789  <.0001
 No pass - Yes pass       -14.79 3.39 142    -23.6    -5.98  -4.364  0.0001

Confidence level used: 0.95
Conf-level adjustment: tukey method for comparing a family of 4 estimates
P value adjustment: tukey method for comparing a family of 4 estimates
```

Typically (and in this course) we are only interested in within-level comparisons. That is, pairwise comparisons in which there is a level in common across the two treatment[3] combinations being compared.

[3] The word *treatment* is often used to refer to a level of a factor variable.

Exam vs test success and attendance

Pairwise comparisons...

To see only the within-level comparisons we can use the `displayPairs` function from the `s20x` package.

```
> displayPairs(exam.pairs, c("No","Yes"), c("nopass","pass"))
```

```
 within               contrast        est       lwr        upr         pval
     No    No nopass - No pass -13.017143 -24.38137  -1.652912 1.776097e-02
    Yes  Yes nopass - Yes pass -24.616034 -34.04182 -15.190247 1.701486e-09
 nopass No nopass - Yes nopass  -3.190476 -15.03851   8.657554 8.969010e-01
   pass     No pass - Yes pass -14.789367 -23.59933  -5.979408 1.423139e-04
```

For example, the first row of the above output says that the estimated difference between the means of the two levels of `Pass.test` conditional on the level of `Attend` = `No` is −13.02. So, for students who do not regularly attend lectures, those who pass the test can expect to score 13 points higher in the exam than those who fail.

Exam vs test success and attendance. . .

Statements for the Executive Summary

We interpret this output as follows (noting that the effect is always conditional on the level of the other factor):

- We estimate that for students who attend regularly, those who pass the test can expect to get 15 to 34 more marks in the exam than those who do not pass the test.
- For students who do not attend regularly, those who pass the test can expect to get 2 to 24 more marks in the exam than those who do not pass the test.
- For students who pass the test, those who regularly attend can expect to get between 6 and 24 more marks in the exam than those who do not attend regularly.
- And, for those who do not pass the test, those who regularly attend can expect to get between 9 marks less and 15 more marks than those who do attend regularly.[4]

[4] Since this difference is not statistically significant, it should **not** be reported in an Executive Summary.

Exam vs test success and attendance. . .

Closing remarks

Recall that the data used in this example are from a single STATS 20x summer school class. The above statements are only relevant to the population of students who could have taken the course that summer.

Moreover, the data were collected pre-covid, so it is certainly the case that attendance at lectures would now not have anywhere near as much effect.

Section 12.5
Example 2: Using gender and attendance to explain exam score

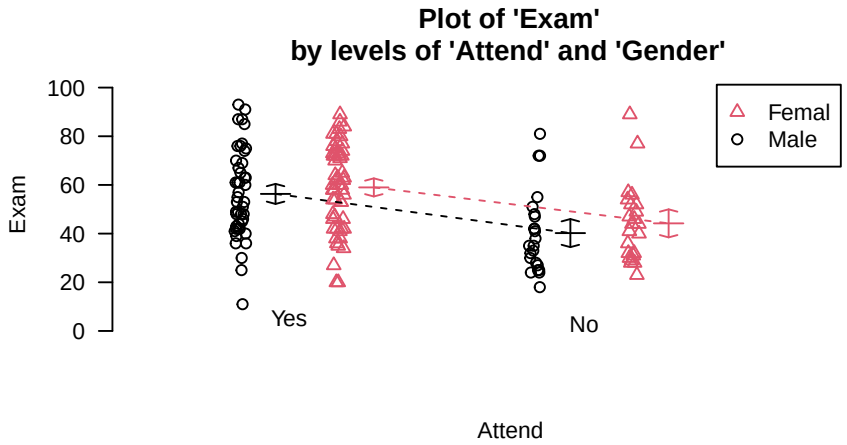
Two-way ANOVA without interaction

Exam score vs gender and attendance

Let us do another analysis where we will ask whether the effect of gender (on exam score) changes depending on whether the student attends regularly or not.

Exam vs gender and attendance

```
> interactionPlots(Exam ~ Attend + Gender, data = Stats20x.df)
```



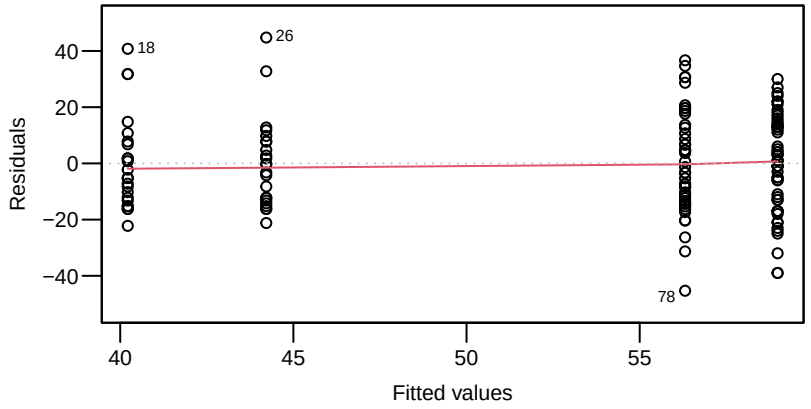
Not so much going on here as our plots are parallel. Looks like there is an effect of attendance (no surprise), and the parallel lines suggests this effect is the same for both genders. There is little difference between genders.

Exam vs gender and attendance. . .

Assumption checks

Let us fit an interaction model and check the assumptions.

```
> Exam.fit2 = lm(Exam ~ Attend*Gender, data = Stats20x.df)
> plot(Exam.fit2, which=1)
```

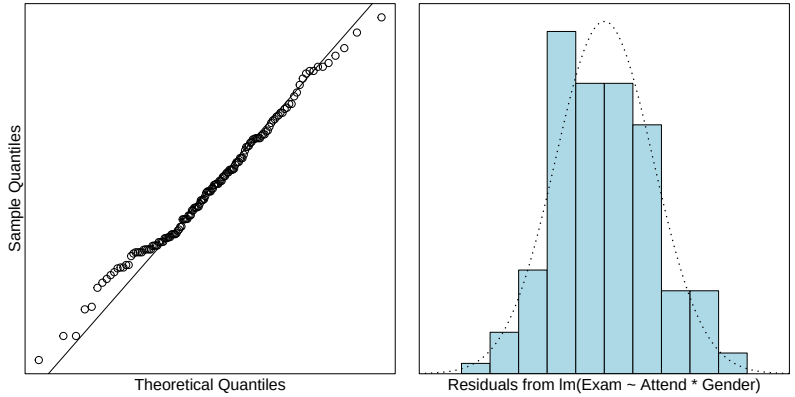


The EOV assumption seems to be okay.

Exam vs gender and attendance. . .

Assumption checks. . .

```
> normcheck(Exam.fit2)
```

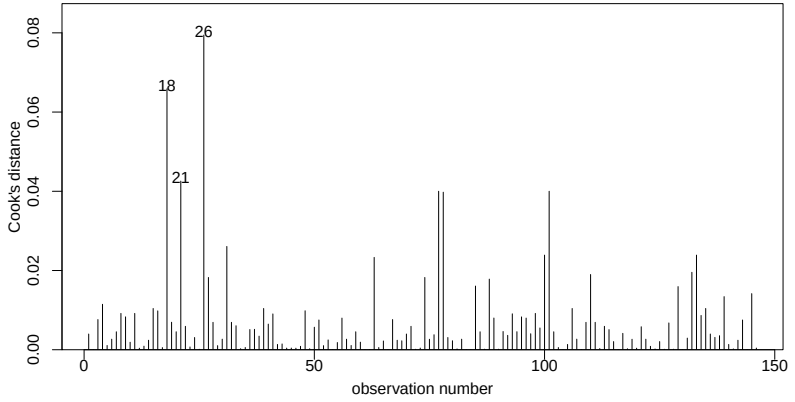


The normality assumption seems to be okay.

Exam vs gender and attendance. . .

Assumption checks. . .

```
> cooks20x(Exam.fit2)
```



No unduly influential data points.

Exam vs gender and attendance...

We can trust the model. Lets see what it is telling us.

```
> anova(Exam.fit2)
Analysis of Variance Table

Response: Exam
               Df Sum Sq Mean Sq F value    Pr(>F)
Attend          1   7631  7630.8 25.4393 1.372e-06 ***
Gender          1    347   346.7  1.1557    0.2842
Attend:Gender   1     14    13.9  0.0463    0.8300
Residuals     142  42594   300.0
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There is definitely no evidence of an interaction, so we'll apply Occam's razor[5] and fit a simpler main-effects model (i.e., no interaction term).

[5] "With all things being equal, the simplest explanation tends to be the right one", William of Ockham, 1287-1347.

Exam vs gender and attendance...

The main-effects model

Recall that we use + rather that * in the model formula to fit the main-effects model.

```
> Exam.fit3 = lm(Exam ~ Attend + Gender, data = Stats20x.df)
> anova(Exam.fit3)
Analysis of Variance Table

Response: Exam
           Df Sum Sq Mean Sq F value    Pr(>F)
Attend      1   7631  7630.8 25.6101 1.264e-06 ***
Gender      1    347   346.7  1.1634    0.2826
Residuals 143  42608   298.0
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that the gender is also not significant here[6], so we again apply Occam's razor and remove this term.

[6] If it were significant then we could look at the output from `pairs(emmeans( Exam.fit3,~ Attend ), infer=T)` and `pairs(emmeans( Exam.fit3,~ Gender ), infer=T)` – but this is pointless here since the factors only have two levels and there is only one pairwise comparison for each factor and so no multi-comparison issue.

Exam vs gender and attendance...

Removal of the gender term reduces the model to one with just a factor with two levels. This is the two sample t-test scenario of Chapter 5.

```
> Exam.fit4 = lm(Exam ~ Attend, data = Stats20x.df)
> summary(Exam.fit4)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   42.217      2.547  16.578  < 2e-16 ***
AttendYes     15.563      3.077   5.058 1.27e-06 ***
---
Residual standard error: 17.27 on 144 degrees of freedom
Multiple R-squared:  0.1508,Adjusted R-squared:  0.145
F-statistic: 25.58 on 1 and 144 DF,  p-value: 1.271e-06
```

```
> confint(Exam.fit4)
                2.5 %   97.5 %
(Intercept) 37.184009 47.25077
AttendYes    9.480749 21.64447
```


Section 12.6
Relevant R-code

Most of the R-code you need for this chapter

You do not need to create indicator variables - R does that for you. The baseline can be changed if you wish rather than having R choose it for you – see relevant R-code from Chapter 9.

Use interaction-plots to inspect the data. Non-parallel lines indicate that interaction may exist.

```
> interactionPlots(Exam ~ Pass.test + Attend, data = Stats20x.df)
```

Fit the interaction model (use the * in the model formula)

```
> Exam.fit = lm(Exam ~ Attend*Pass.test, data = Stats20x.df)
```

and use the ANOVA table to see if there is evidence of interaction.

```
> anova(ExamTestAttend.fit)
```

In the first example we had evidence of interaction (small P-value associated with ":" part of the ANOVA output) and we inspect the pairwise interactions using emmeans to correct CIs for multi-comparisons:

```
> exam.pairs = pairs(emmeans(Exam.fit, ~ Attend*Pass.test), infer=T)
> displayPairs(exam.pairs, c("No","Yes"), c("nopass","pass"))
```


Most of the R-code you need for this chapter

If you don't have any evidence of interaction then simplify your model to a main-effects model and then see if the individual terms are significant.

The main-effects model replaces * with a + in the model formula. E.g.,

```
> Exam.fit3=lm(Exam ~ Attend+Gender, data = Stats20x.df)
```

If both variables are significant in the main-effects model then use that model for inference (this model is referred to here as additive.fit)

```
> pairs(emmeans(additive.fit, ~ variable1), infer=T)
```

```
> pairs(emmeans(additive.fit, ~ variable2), infer=T)
```

Otherwise, delete non-significant variables until you have the simplest model possible.

Section 12.7
Alternative parameterizations of the two-way ANOVA model

(This is an optional Section
- your lecturer will advise if it is examinable)

Alternative parameterizations of two-way ANOVA

The reference cell model

Recall the reference cell model we used to represent Exam score:

$$\texttt{Exam} = \beta_0 + \beta_1 \times \texttt{Attend}_{\texttt{Yes}} + \beta_2 \times \texttt{Pass.test}_{\texttt{pass}} + \beta_3 \times \texttt{Attend}_{\texttt{Yes}} \times \texttt{Pass.test}_{\texttt{pass}} + \varepsilon,$$

where $\varepsilon \overset{iid}{\sim} N(0, \sigma^2)$.

The parameter β_0 denotes the overall true baseline mean exam score. Notice that neither the no level of Attend nor the nopass level of Pass.test appear as subscripts in the above model. This tells us that these are the baseline levels, i.e. β_0 denotes the mean over Exam scores from students who neither regularly attended lectures nor passed the test.

So, what do the parameters $\beta_1, \beta_2,$ and β_3 represent? To help us answer this question we consider the means model[7] formulation for Exam score.

[7] We first encountered the means model for the single factor male fruitflies study in Chapter 11.

Alternative parameterizations of two-way ANOVA

The means model

The means model parameterization for exam score is

$$\texttt{Exam}_{ijk} = \mu_{ij} + \varepsilon_{ijk},$$

where μ_{ij} denotes the true mean exam score of 20x students who are in the ith level of `Attend` and jth level of `Pass.test` ($i =$ `no` or `yes`; $j =$ `nopass` or `pass`). The error term $\varepsilon_{ijk} \overset{iid}{\sim} N(0, \sigma^2)$ denotes the deviation of the kth student's exam score from the mean exam score, μ_{ij}.

But, how can we use μ_{ij} to assess whether one or both of `Attend` and `Pass.test` have an effect on `Exam` score?

Alternative parameterizations of two-way ANOVA

Relating the means and reference cell models

We decompose each mean response, μ_{ij}, into four terms:

1. μ_{11}, the baseline or reference-level mean response;
2. $\mu_{i1} - \mu_{11}$, the *main effect*[8] of the ith level of the first factor, where i does not equal the baseline level;
3. $\mu_{1j} - \mu_{11}$, the *main effect* of the jth level of the second factor, where j does not equal the baseline level;
4. Interaction, the part of μ_{ij} that is left over after eliminating the contributing components defined by terms 1–3 above, i.e.

$$\begin{aligned}\text{Interaction} &= \mu_{ij} - \mu_{11} - (\mu_{i1} - \mu_{11}) - (\mu_{1j} - \mu_{11}) \\ &= \mu_{ij} - \mu_{i1} - \mu_{1j} + \mu_{11}\end{aligned}$$

[8] A main effect is defined as the difference between the mean response when all factors, except the one of interest, are at the baseline level and the reference-level mean.

Alternative parameterizations of two-way ANOVA

Relating the means and reference cell models

We now have the tools to re-express the mean `Exam` score in terms of the main effects of the ith level of `Attend` and the jth level of `Pass.test`, and their interaction, i.e.

$$\mu_{ij} = \mu_{11} + (\mu_{1j} - \mu_{11}) + (\mu_{i1} - \mu_{11}) + (\mu_{ij} - \mu_{i1} - \mu_{1j} + \mu_{11}).$$

The following two-way table[9] illustrates how each term in the above decomposition relates to each combination of the levels of `Attend` and `Pass.test`:

	Pass.test	
Attend	nopass	pass
no	μ_{11}	$\mu_{i1} - \mu_{11}$
yes	$\mu_{1j} - \mu_{11}$	$\mu_{ij} - \mu_{11} - (\mu_{i1} - \mu_{11}) - (\mu_{1j} - \mu_{11})$

[9] More generally, differences in the first row of a two-way reference model decomposition table correspond to the main effects of the column factor. The differences in the first column correspond to the row factor main effects. The terms in each of the of the remaining cells, except the reference cell, correspond to interaction effects.

Alternative parameterizations of two-way ANOVA

Relating the means and reference cell models

Factors		Parameterization			
Attend	Pass.test	Means	Estimate[10]	Reference cell	Estimate[11]
no	nopass	μ_{11}	35.1	$\beta_0 = \mu_{11}$	35.1
yes	nopass	μ_{21}	38.3	$\beta_1 = \mu_{21} - \mu_{11}$	3.2
no	pass	μ_{12}	48.2	$\beta_2 = \mu_{12} - \mu_{11}$	13.1
yes	pass	μ_{22}	62.9	$\beta_3 = \mu_{22} - \mu_{21} - \mu_{12} + \mu_{11}$	11.5

From the above table we see that:

- β_1 represents the effect of `Attend = yes` at the reference level of `Pass.test = nopass`
- β_2 represents the effect of `Pass.test = pass` at the reference level of `Attend = no`
- β_3 represents the `Attend` $\times$ `Pass.test` interaction effect when `Attend = yes` and `Pass.test = pass`

[10] See estimates of `Attend`×`Pass.test` treatment means on slide **??**.
[11] See regression coefficients table on slide 17.

Alternative parameterizations of two-way ANOVA

The reference cell model

The values in the `Estimate` column of the regression summary table[12] result in the following equation for predicted longevity:

$$\widehat{\texttt{Exam}} = 35.14 + 3.19 \times \texttt{Attend}_{\texttt{Yes}} + 13.02 \times \texttt{Pass.test}_{\texttt{pass}} + 11.60 \times \texttt{Attend}_{\texttt{Yes}} \times \texttt{Pass.test}_{\texttt{pass}}$$

[12] See slide 17; Coefficients rounded to 2 decimal places.

Alternative parameterizations of two-way ANOVA

Relating the means and effects models

We saw in Chapter 11 that the effects model offers an alternative to the reference model parameterization.[13] To relate the means and effects models we use an alternative decomposition of each mean response, μ_{ij}, into:

1. $\mu = \mu_{\cdot\cdot}$, the reference overall mean response;
2. $\alpha_i = \mu_{i\cdot} - \mu$, the *main effect* of the ith level of the first factor[14];
3. $\pi_j = \mu_{\cdot j} - \mu$, the *main effect* of the jth level of the second factor[14];
4. $(\alpha\tau)_{ij}$, the interaction effect reflecting the component of μ_{ij} left over after eliminating the components corresponding to the terms defined in 1–3 above, i.e. $(\alpha\tau)_{ij} = \mu_{ij} - \mu - \alpha_i - \pi_j$.

[13] The effects model is the default parameterization used in the analysis of data from designed experiments and, therefore, in STATS 240 (Design and Structured Data).

[14] The distance between the mean response of the first (second) factor at the ith (jth) level and the overall mean.

Alternative parameterizations of two-way ANOVA

The effects model

It directly follows from point 4 above that we can re-express the mean `Exam` score, μ_{ij}, in terms of α_i, the effect of the ith level of `Attend`, π_j, the effect of the jth level of `Pass.test`, and their interaction, $(\alpha\pi)_{ij}$, i.e.

$$\mu_{ij} = \mu + \alpha_i + \pi_j + (\alpha\pi)_{ij}.$$

This means that the effects model formula for the two-way ANOVA is

$$\texttt{Exam}_{ijk} = \mu + \alpha_i + \pi_j + (\alpha\pi)_{ij} + \epsilon_{ijk},$$

where $\epsilon_{ijk} \overset{iid}{\sim} N(0, \sigma^2)$.

Alternative parameterizations of two-way ANOVA

Relating the means and effects models

The following two-way layout illustrates how the above decomposition of μ_{ij} relates to each combination of the levels of `Attend` and `Pass.test`:

Attend	Pass.test		Row mean	α_i, Row effect
	nopass	pass		
no	$\mu_{11} = 35.1$	$\mu_{12} = 48.2$	$\mu_{1\cdot} = 41.7$	
yes	$\mu_{21} = 38.3$	$\mu_{22} = 62.9$	$\mu_{2\cdot} = 50.6$	
Column mean	$\mu_{\cdot 1} = 36.7$	$\mu_{\cdot 2} = 55.6$	$\mu = 52.9$	
π_i, Column effect				

```
> emmeans(Exam.fit,~Attend)
 Attend emmean   SE  df lower.CL upper.CL
 No       41.7 2.19 142     37.3     46.0
 Yes      50.6 1.81 142     47.1     54.2

Results are averaged over the levels of: Pass.test
Confidence level used: 0.95
> emmeans(Exam.fit,~Pass.test)
 Pass.test emmean   SE  df lower.CL upper.CL
 nopass      36.7 2.28 142     32.2     41.2
 pass        55.6 1.69 142     52.2     58.9
```

Alternative parameterizations of two-way ANOVA

Relating the means and effects models

Factors		Parameterization			
Attend	Pass.test	Means	Estimate[15]	Reference cell	Estimate[16]
no	nopass	μ_{11}	35.1	$\beta_0 = \mu_{11}$	35.1
yes	nopass	μ_{21}	38.3	$\beta_1 = \mu_{21} - \mu_{11}$	3.2
no	pass	μ_{12}	48.2	$\beta_2 = \mu_{12} - \mu_{11}$	13.1
yes	pass	μ_{22}	62.9	$\beta_3 = \mu_{22} - \mu_{21} - \mu_{12} + \mu_{11}$	11.5

From the above table we see that:

- β_1 represents the effect of `Attend = yes` at the reference level of `Pass.test = nopass`
- β_2 represents the effect of `Pass.test = pass` at the reference level of `Attend = no`
- β_3 represents the `Attend` x `Pass.test` interaction effect when `Attend = yes` and `Pass.test = pass`

[15] See estimates of `Attend`×`Pass.test` treatment means on slide **??**.

[16] See regression coefficients table on slide 17.

Case Study 12.1: Exam vs Test & Attendance

Tou Ohone Andate - staff number 1234567

Problem

We wish to determine if regular attendance in class and passing the test is associated with exam mark. We have seen individually that passing the test and regular attendance in class increases the chances for a good exam mark — on average.

The variables of interest are:

- `Exam`: Exam mark out of 100.
- `Attend`: A two-level factor with the levels `Yes` and `No`.
- `Pass.test`: A two-level factor with the levels `Yes` and `No`.

Question of Interest

We are interested in the relationship between final exam mark and passing the test, and whether this relationship differs for students who did and did not attend lectures regularly.

Read in and Inspect the Data

```
Stats20x.df = read.table("STATS20x.txt", header = T)
head(Stats20x.df)
```

```
##   Grade Pass Exam Degree Gender Attend Assign Test  B  C MC Colour Stage1
## 1     C  Yes   42    BSc   Male    Yes   17.2  9.1  5 13 12   Blue      C
## 2     B  Yes   58   BCom Female    Yes   17.2 13.6 12 12 17 Yellow      A
## 3     A  Yes   81  Other Female    Yes   17.2 14.5 14 17 25   Blue      A
## 4     A  Yes   86  Other Female    Yes   19.6 19.1 15 17 27 Yellow      A
## 5     D   No   35  Other   Male     No    8.0  8.2  4  1 15   Blue      C
## 6     A  Yes   72   BCom Female    Yes   18.4 12.7 15 17 20   Blue      A
##   Years.Since Repeat
## 1         2.5    Yes
## 2         2.0     No
## 3         3.0     No
## 4         0.0     No
## 5         3.0     No
## 6         1.5     No
```

```
# So, we have to create the variable `Pass.test' using the `Test` variable
Stats20x.df$Pass.test = factor(ifelse(Stats20x.df$Test >= 10, "pass", "nopass"))
# Check if our new variable `Pass.test` was generated correctly
min(Stats20x.df$Test[Stats20x.df$Pass.test == "pass"])
```

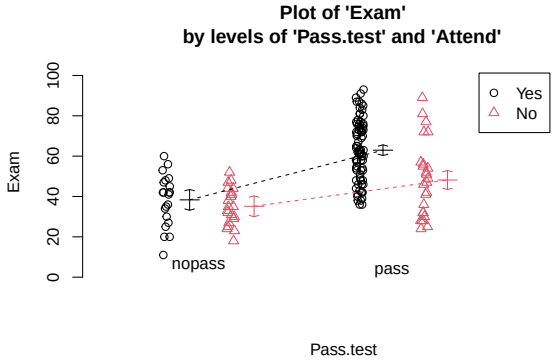


```
## [1] 10
```

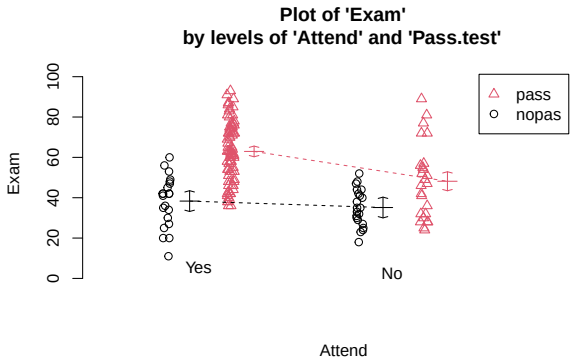
```
max(Stats20x.df$Test[Stats20x.df$Pass.test == "nopass"])
```

```
## [1] 9.1
```

```
interactionPlots(Exam ~ Pass.test + Attend, data = Stats20x.df)
```



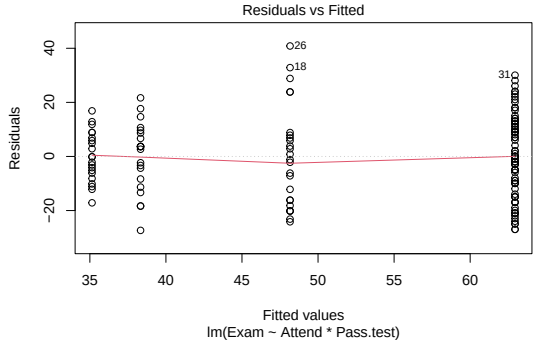

```
# Also look at the interaction plot the other way around:
interactionPlots(Exam ~ Attend + Pass.test, data = Stats20x.df)
```



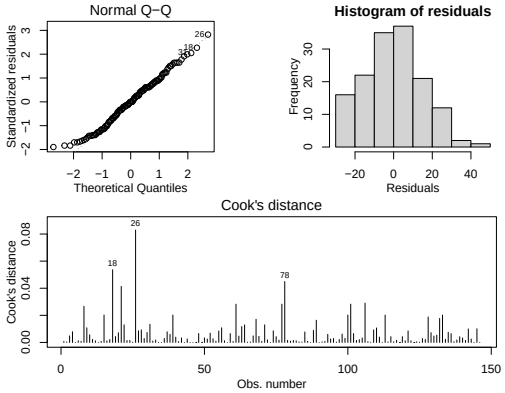
Exam marks for students who passed the test are centred higher than for students who did not. However, this difference appears to be greater for students who attended lectures than for non-attenders, which can be seen from the non-parallel coloured lines on the interaction plots. In other words, the impact of passing the test appears to differ according to attendance group, suggesting there may be an interaction between the two predictor variables.

Model Building and Check Assumptions

```
Exam.fit = lm(Exam ~ Attend * Pass.test, data = Stats20x.df)
# The model formula could of also been `Exam ~ Attend + Pass.test + Attend:Past.test'
plot(Exam.fit,1)
```




```
modelcheck(Exam.fit,2:3)
```


```
anova(Exam.fit)
```

```
## Analysis of Variance Table
##
## Response: Exam
##                   Df  Sum Sq Mean Sq F value    Pr(>F)
## Attend             1  7630.8  7630.8  34.990 2.364e-08 ***
## Pass.test          1 11076.9 11076.9  50.791 4.763e-11 ***
## Attend:Pass.test   1   909.7   909.7   4.171   0.04297 *
## Residuals        142 30968.4   218.1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(Exam.fit)
```

```
##
## Call:
## lm(formula = Exam ~ Attend * Pass.test, data = Stats20x.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -27.333 -10.893  -0.046   9.513  40.840
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)               35.143      3.223  10.905  < 2e-16 ***
## AttendYes                  3.190      4.557   0.700  0.48504
## Pass.testpass             13.017      4.371   2.978  0.00341 **
## AttendYes:Pass.testpass   11.599      5.679   2.042  0.04297 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.77 on 142 degrees of freedom
```

```
## Multiple R-squared:  0.3878, Adjusted R-squared:  0.3749
## F-statistic: 29.98 on 3 and 142 DF,  p-value: 4.452e-15
```

Paiwise comparisons

```
library(emmeans)
Exam.pairs = pairs(emmeans(Exam.fit, ~Attend*Pass.test), infer=T)
Exam.pairs
```

```
##  contrast               estimate   SE  df lower.CL upper.CL t.ratio p.value
##  No nopass - Yes nopass    -3.19 4.56 142    -15.0     8.66  -0.700  0.8969
##  No nopass - No pass      -13.02 4.37 142    -24.4    -1.65  -2.978  0.0178
##  No nopass - Yes pass     -27.81 3.63 142    -37.2   -18.38  -7.669  <.0001
##  Yes nopass - No pass      -9.83 4.37 142    -21.2     1.54  -2.248  0.1155
##  Yes nopass - Yes pass    -24.62 3.63 142    -34.0   -15.19  -6.789  <.0001
##  No pass - Yes pass       -14.79 3.39 142    -23.6    -5.98  -4.364  0.0001
##
## Confidence level used: 0.95
## Conf-level adjustment: tukey method for comparing a family of 4 estimates
## P value adjustment: tukey method for comparing a family of 4 estimates
```

```
# Get a simpler display using displayPairs:
displayPairs(Exam.pairs, c("Yes", "No"), c("pass", "nopass"))
```

```
## Note: displayPairs is a s20x function that displays only the within-level
## comparisons from allpairs. To see all comparisons, inspect the allpairs
## output directly.
##
## $Yes
##               contrast       est       lwr       upr         pval
##  Yes nopass - Yes pass -24.61603 -34.04182 -15.19025 1.701486e-09
##
## $No
##             contrast       est       lwr       upr       pval
##  No nopass - No pass -13.01714 -24.38137 -1.652912 0.01776097
##
## $pass
##            contrast       est       lwr       upr         pval
##  No pass - Yes pass -14.78937 -23.59933 -5.979408 0.0001423139
##
## $nopass
##                contrast       est       lwr      upr     pval
##  No nopass - Yes nopass -3.190476 -15.03851 8.657554 0.896901
```

Methods and Assumption Checks

We have two explanatory factors, `Pass.test` and `Attend`, and one numeric response `Exam`, so we fitted a two-way ANOVA model with interaction term. The interaction was significant ($P\text{-}value = 0.04$) so it was retained in the model.

The model assumptions seem satisfied.

Our final model is

$$\text{Exam}_i = \beta_0 + \beta_1 \times Test.pass_i + \beta_2 \times Attend.Yes_i + \beta_3 \times Test.pass_i \times Attend.Yes_i + \epsilon_i,$$

where $Test.pass_i$ and $Attend.Yes_i$ are dummy variables that take the value 1 if student i passed the test and if student i regularly attended lectures respectively, otherwise they are 0; and $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Alternatively, our final model could be written as

$$\text{Exam}_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk},$$

where Exam_{ijk} is the mark of student k in test-pass group i and attendance group j; and where μ is the overall mean exam mark, α_i is the effect of whether the student passed the test, β_j is the effect of whether the student regularly attended lectures, γ_{ij} is the interaction effect for the combination of a student passing the test and attendance, and $\epsilon_{ijk} \sim iid\ N(0, \sigma^2)$. Here, $i \in \{\text{pass},\ \text{nopass}\}$ and $j \in \{\text{Yes},\ \text{No}\}$.

Our model explained 39% of variability in students' exam marks.

Executive Summary

We are interested in the relationship between final exam mark and passing the test, and whether this relationship differs for those students that attended lectures regularly and those that didn't.

We have evidence that the relationship between exam mark and test-pass status does depend upon attendance practice.

We estimate that:

- Among students who attended regularly, the average exam mark for those who passed the test was between 15 and 34 marks higher than for those who didn't pass the test.
- Among students who did not attend regularly, the average exam mark for those who passed the test was between 2 and 24 marks higher than for those who didn't pass the test.
- Among students who passed the test, the average exam mark for those who attended regularly was between 6 and 24 marks higher than for those who didn't attend regularly.

Case Study 12.2: Impact of a cyclone on coral reefs

Marti Anderson

Problem

The Australian Institute of Marine Science monitors coral and fish assemblages along the Great Barrier Reef. These data consist of percentage cover of hard corals recorded from underwater video transects from various reef sites. In 1997, parts of the reef were affected by tropical cyclone "Justin". We wish to investigate the impact of the cyclone on coral cover.

Source of data: Anderson, M.J. and Thompson, A.A. (2004). Multivariate control charts for ecological and environmental monitoring. Ecological Applications 14: 1921-1935.

This type of design is called a **BACI** design, for *before-after*, *control-impact*. One factor is `Time` with two levels: `Before` and `After`. The other factor is `Reef` (or `Site`) with two levels: `Control` and `Impact`. An

interaction would imply that the trend in coral cover over time differed between control sites and impacted sites. Here, the sites classified as `impacted` were outer reefs which take the brunt of any storms.

The variables of interest were:

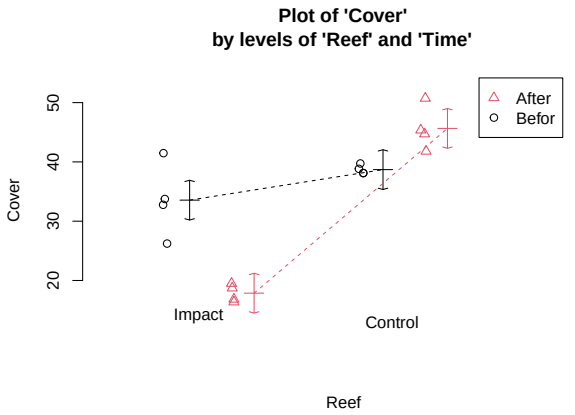
- `Cover`: percentage cover of hard corals in the reef.
- `Reef`: a two-level factor with levels `Control` and `Impact`.
- `Time`: a two-level factor with levels `Before` and `After`.

Research question

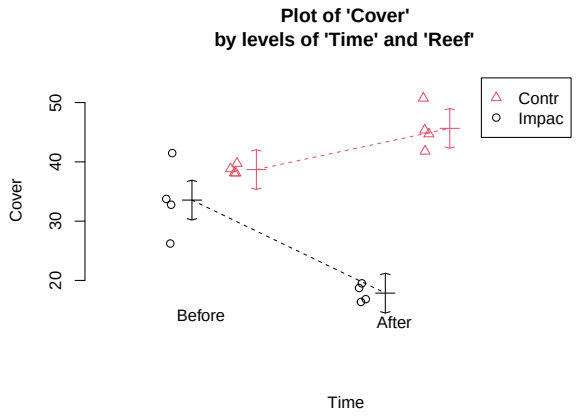
What was the impact of cyclone Justin on coral cover in the affected sites?

Read in and Inspect the Data

```
Reef.df = read.csv("Reef.csv")
Reef.df = transform(Reef.df, Reef=factor(Reef), Time=factor(Time))
interactionPlots(Cover ~ Reef + Time, data = Reef.df)
```



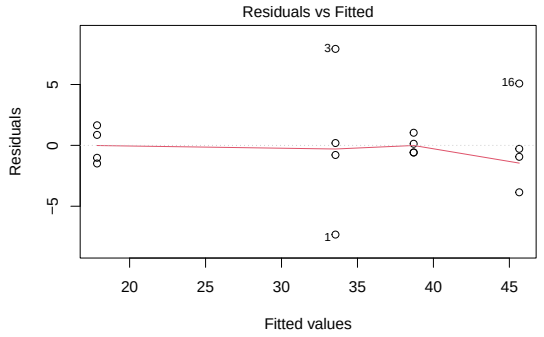

```
# Also look at the interaction plot the other way around:
interactionPlots(Cover ~ Time + Reef, data = Reef.df)
```



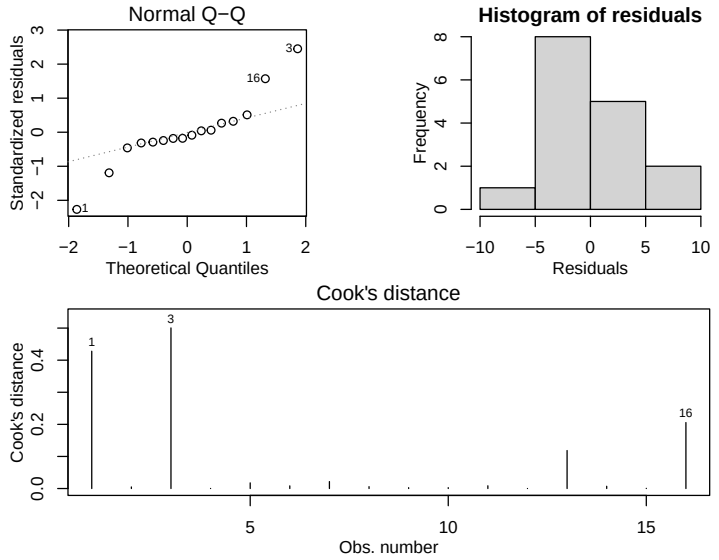
The trend in coral cover over time appears to differ for impacted and control reefs. The cover appears fairly similar in both reef categories before Cyclone Justin. However, for control reefs, the cover increased after the cyclone, whereas for impacted reefs, it decreased. From the highly non-parallel appearance of the coloured lines we suspect there is an interaction between the `Time` and `Reef` predictors.

Model Building and Check Assumptions

```
Cover.fit = lm(Cover ~ Time * Reef, data = Reef.df)
plot(Cover.fit, which=1)
```




```
modelcheck(Cover.fit,which=2:3)
```




```
anova(Cover.fit)
```

```
## Analysis of Variance Table
##
## Response: Cover
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## Time       1   76.46   76.46  5.4834   0.03727 *
## Reef       1 1085.24 1085.24 77.8318 1.362e-06 ***
## Time:Reef  1  512.84  512.84 36.7802 5.631e-05 ***
## Residuals 12  167.32   13.94
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(Cover.fit)
```

```
##
## Call:
## lm(formula = Cover ~ Time * Reef, data = Reef.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.3282 -0.9609 -0.4308  0.9049  7.9257
```

```
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             45.651      1.867  24.451 1.32e-11 ***
## TimeBefore              -6.951      2.640  -2.633   0.0219 *
## ReefImpact             -27.795      2.640 -10.527 2.05e-07 ***
## TimeBefore:ReefImpact   22.646      3.734   6.065 5.63e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.734 on 12 degrees of freedom
## Multiple R-squared:  0.9092, Adjusted R-squared:  0.8864
## F-statistic: 40.03 on 3 and 12 DF,  p-value: 1.583e-06
```

Pairwise comparisons using emmeans

```
Cover.pairs <- pairs(emmeans(Cover.fit, ~ Time*Reef), infer=T)
# Simplify the display:
displayPairs(Cover.pairs, c("Before", "After"), c("Control", "Impact"))
```

```
## Note: displayPairs is a s20x function that displays only the within-level
## comparisons from allpairs. To see all comparisons, inspect the allpairs
## output directly.
##
## $Before
##                         contrast    est       lwr      upr      pval
##  Before Control - Before Impact 5.1485 -2.690584 12.98758 0.2593307
##
## $After
##                       contrast     est      lwr      upr         pval
##  After Control - After Impact 27.7945 19.95542 35.63358 1.079047e-06
##
## $Control
##                         contrast   est        lwr      upr      pval
##  After Control - Before Control 6.951 -0.8880843 14.79008 0.0886507
##
## $Impact
##                       contrast     est       lwr       upr         pval
##  After Impact - Before Impact -15.695 -23.53408 -7.855916 0.0003413045
```

Methods and Assumption Checks

We have a numeric response, `Cover`, and two explanatory factors, `Reef` and `Time`, so we fitted a two-way ANOVA model with interaction between `Reef` and `Time`. The interaction term was significant (P-value $<$ 0.001) so it was retained.

The assumption checks were reasonable, except for the Cook's distance which exceeded 0.4 for a couple of points. However, this was not of great concern, since there were no obvious outliers. It is not surprising that some data points will have high influence when there are only 16 observations in the dataset.

Our final model was

$$\text{Cover}_i = \beta_0 + \beta_1 \ \times \text{Time.Before}_i + \beta_2 \times \text{Reef.Impact}_i + \beta_3 \times \text{Time.Before}_i \times \text{Reef.Impact}_i + \epsilon_i,$$

where Time.Before_i and Reef.Impact_i are the dummy variables that take the value 1 if observation i was respectively recorded before Cyclone Justin and in an impacted reef site, otherwise 0; and $\epsilon_i \sim iid\ N(0, \sigma^2)$.

Alternatively, our final model could be written as

$$\text{Cover}_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk},$$

where Cover_{ijk} denotes the coral cover for observation k in time category i and reef category j, and where μ is the overall mean percentage cover of the reefs, α_i is the effect of time category i, β_j is the effect of reef category j, γ_{ij} is the interaction effect for the combination of time category i and reef category j, and $\epsilon_{ijk} \sim iid\ N(0, \sigma^2)$.

Our model explained 91% of the variation in the coral cover measurements.

Executive Summary

We aimed to investigate the effect of Cyclone Justin on coral cover in impacted reef sites.

There was no evidence of a difference in the average amount of coral cover between impacted sites and control sites before Cyclone Justin. However, there was strong evidence of a difference after the cyclone.

We estimate that the average coral cover in impacted reefs was between 8 and 24 percentage points lower after the cyclone than before. By comparison, there was no loss in the average coral cover in control reefs (in fact, there was weak evidence of an increase).

After the cyclone, we estimate that the average coral cover was between 20 and 36 percentage points greater in control reefs than in impacted reefs.

Case Study 12.3: Extracting dietary supplements from seaweed

Russell Millar

Problem

Fucoidan is a dietary supplement found in several species of brown seaweed, including the invasive species *Undaria* which is now widespread in New Zealand.

One silver lining to the invasion of *Undaria* is that it has potential to be a valuable source of *fucoidan*. These data come from a study investigating the yield of *fucoidan* from *Undaria* under different laboratory conditions.

The response is `HikDa` (a measurement based on molecular weight), and the explanatory variables are the two factor variables `fTemp` (temperature level) and `fTime` (time level), each of which has three levels.

The variables of interest were:

- `HiKda`: A measurement based on molecular weight.
- `fTemp`: A three-level factor with the levels "60", "70", and "80".
- `fTime`: A three-level factor with the levels "2", "3", and "4".

Question of Interest

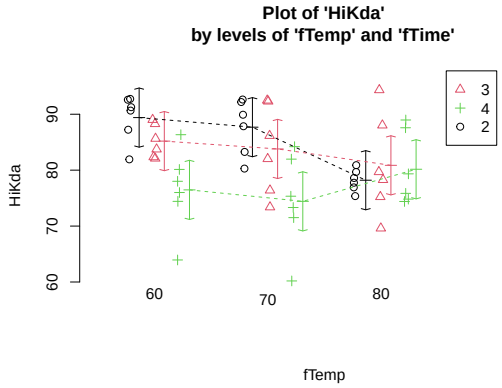
We wish to determine how the yield of fucoidan from *Undaria* depends on the temperature and time level, and whether these factors influence each other.

Read in and Inspect the Data

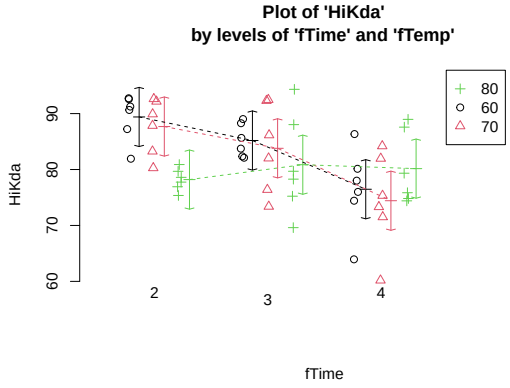
```
# The call for tab delimited data
Weed.df = read.table("Weed.txt", header = TRUE, sep = "\t")
# Several R commands to subset the data for answering the question of interest:
WeedDf = transform(Weed.df, logAlg = log(Alg), Ratio = as.character(Ratio))
WeedDf = transform(WeedDf, Ratio = as.numeric(substr(Ratio, 3, nchar(Ratio))))
WeedDf = transform(WeedDf, fTime = factor(Time), fTemp = factor(Temp),
                           fRatio = factor(Ratio))
```



```
KdaDf = subset(WeedDf, subset = (!is.na(HiKda)))
interactionPlots(HiKda ~ fTemp + fTime, data = KdaDf)
```



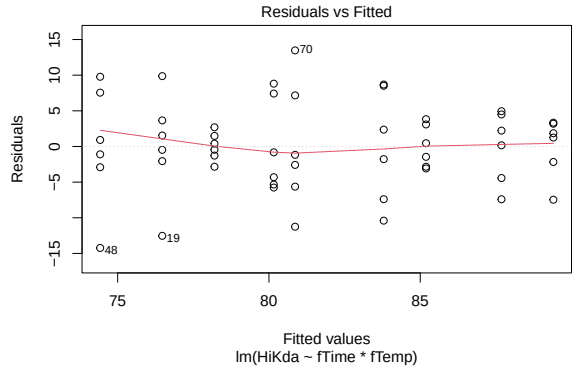

```
# Also look at the interaction plot the other way around:
interactionPlots(HiKda ~ fTime + fTemp, data = KdaDf)
```



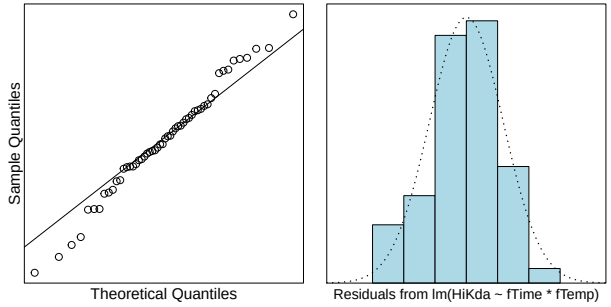
An interaction is suggested because of the non-parallel lines. When the temperature is 80 degrees, the yield is similar for all time levels. When temperature is 60 and 70 degrees, the yield decreases as time increases.

Model Building and Check Assumptions

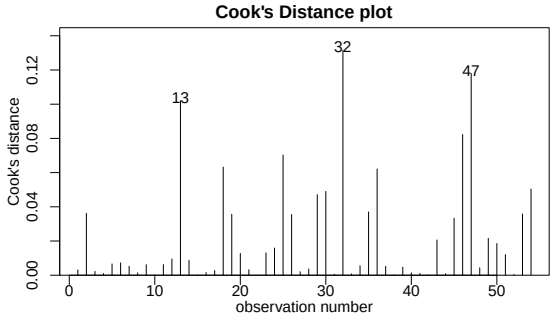
```
Kda.lm = lm(HiKda ~ fTime * fTemp, data = KdaDf)
plot(Kda.lm, which=1)
```




```
normcheck(Kda.lm)
```




```
cooks20x(Kda.lm)
```




```
anova(Kda.lm)
```

```
## Analysis of Variance Table
##
## Response: HiKda
##             Df  Sum Sq Mean Sq F value   Pr(>F)
## fTime        2  646.56  323.28  7.8746 0.001168 **
## fTemp        2  140.99   70.49  1.7171 0.191142
## fTime:fTemp  4  455.71  113.93  2.7751 0.038203 *
## Residuals   45 1847.41   41.05
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(Kda.lm)
```

```
##
## Call:
## lm(formula = HiKda ~ fTime * fTemp, data = KdaDf)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.2317  -2.8942  -0.1425   3.3025  13.4767
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     89.4000     2.6158  34.177  < 2e-16 ***
## fTime3          -4.2083     3.6993  -1.138  0.26130
## fTime4         -12.9267     3.6993  -3.494  0.00108 **
```

```
## fTemp70         -1.7150     3.6993  -0.464  0.64516
## fTemp80        -11.2000     3.6993  -3.028  0.00407 **
## fTime3:fTemp70   0.3167     5.2315   0.061  0.95200
## fTime4:fTemp70  -0.3367     5.2315  -0.064  0.94897
## fTime3:fTemp80   6.8717     5.2315   1.314  0.19567
## fTime4:fTemp80  14.8867     5.2315   2.846  0.00665 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.407 on 45 degrees of freedom
## Multiple R-squared:  0.4023, Adjusted R-squared:  0.296
## F-statistic: 3.785 on 8 and 45 DF,  p-value: 0.001811
```

Pairwise comparisons

```
Kda.pairs = pairs(emmeans(Kda.lm, ~ fTime*fTemp), infer=T)
# Simplify the display of pairwise comparisons.
# Because factor levels are numbers, need to enter as "fTime2", "fTime3", etc:
displayPairs(Kda.pairs, c("fTime2", "fTime3", "fTime4"), c("fTemp60", "fTemp70", "fTemp80"))
```

```
## Note: displayPairs is a s20x function that displays only the within-level
## comparisons from allpairs. To see all comparisons, inspect the allpairs
## output directly.
##
## $fTime2
##                         contrast    est         lwr    upr       pval
##  fTime2 fTemp60 - fTime2 fTemp70  1.715 -10.3339988 13.764 0.99993090
##  fTime2 fTemp60 - fTime2 fTemp80 11.200  -0.8489988 23.249 0.08688628
##  fTime2 fTemp70 - fTime2 fTemp80  9.485  -2.5639988 21.534 0.23050599
##
## $fTime3
##                         contrast      est        lwr      upr      pval
##  fTime3 fTemp60 - fTime3 fTemp70 1.398333 -10.650666 13.44733 0.9999856
##  fTime3 fTemp60 - fTime3 fTemp80 4.328333  -7.720666 16.37733 0.9587448
##  fTime3 fTemp70 - fTime3 fTemp80 2.930000  -9.118999 14.97900 0.9965610
##
## $fTime4
##                         contrast       est        lwr       upr      pval
##  fTime4 fTemp60 - fTime4 fTemp70  2.051667  -9.997332 14.100666 0.9997341
##  fTime4 fTemp60 - fTime4 fTemp80 -3.686667 -15.735666  8.362332 0.9843199
##  fTime4 fTemp70 - fTime4 fTemp80 -5.738333 -17.787332  6.310666 0.8249778
##
## $fTemp60
##                         contrast       est        lwr      upr       pval
##  fTime2 fTemp60 - fTime3 fTemp60  4.208333 -7.8406655 16.25733 0.96494777
##  fTime2 fTemp60 - fTime4 fTemp60 12.926667  0.8776678 24.97567 0.02712413
##  fTime3 fTemp60 - fTime4 fTemp60  8.718333 -3.3306655 20.76733 0.33196381
##
## $fTemp70
##                         contrast       est       lwr      upr       pval
##  fTime2 fTemp70 - fTime3 fTemp70  3.891667 -8.157332 15.94067 0.97805768
##  fTime2 fTemp70 - fTime4 fTemp70 13.263333  1.214334 25.31233 0.02124471
```



```
##  fTime3 fTemp70 - fTime4 fTemp70  9.371667 -2.677332 21.42067 0.24400944
##
## $fTemp80
##                         contrast       est       lwr       upr      pval
##  fTime2 fTemp80 - fTime3 fTemp80 -2.6633333 -14.71233  9.385666 0.9982321
##  fTime2 fTemp80 - fTime4 fTemp80 -1.9600000 -14.00900 10.088999 0.9998109
##  fTime3 fTemp80 - fTime4 fTemp80  0.7033333 -11.34567 12.752332 0.9999999
```

Methods and Assumption Checks

We have a numeric response `HiKda`, and two explanatory factors, `fTime` and `fTemp`, so we fitted a two-way ANOVA model with interaction. The interaction term was significant ($P\text{-}value$ =0.04) so it was retained for the final model.

The model assumptions were satisfied.

The final model is

$$\text{HiKda}_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk},$$

where μ is the overall mean yield, α_i is the effect of the ith time-point, β_j is the effect of the jth temperature, γ_{ij} is the interaction effect for the combination of the ith time-point and the jth temperature, and $\epsilon_{ijk} \sim iid\ N(0, \sigma^2)$.

Our model explained 40% of the variability in the yield (`HiKda`).

Executive Summary

We wish to determine how the yield of fucoidan from *Undaria* depends on the temperature and time level, and whether these factors influence each other.

We found that the effect that temperature had on the yield depended on the time level, so we could not look at the effects of temperature and time individually.

At the temperature of 60 degrees, the expected yield was estimated to be between 1 and 25 units higher at time-point 2 than time-point 4. The same statement also applies to the temperature of 70 degrees.[1]

There was some (though weaker) evidence that yield at time-point 2 was higher at a temperature of 60 degrees than at a temperature of 80 degrees.

Additional Comments

Looking at the interaction plot, there appears to be little if any difference between the distribution of `HiKda` values at the two lower temperatures. It might be worth combining these two groups into a single group, because this would reduce `fTemp` to a two-level factor which increases the degrees of freedom and reduces the magnitude of the multi-comparison adjustment, thereby resulting in more statistical power. This is something that would need to be discussed with the researchers.

[1] Note that only simple contrasts are reported.

Example code if there is no interaction

Had there been no interaction then the multi-comparison adjustment would have proceeded thus:

```
Kda.Wrong.lm = lm(HiKda ~ fTime + fTemp, data = KdaDf)
pairs(emmeans(Kda.Wrong.lm, ~fTime), infer=T)
```

```
##  contrast        estimate   SE df lower.CL upper.CL t.ratio p.value
##  fTime2 - fTime3     1.81 2.29 49   -3.711     7.34   0.793  0.7091
##  fTime2 - fTime4     8.08 2.29 49    2.553    13.60   3.534  0.0026
##  fTime3 - fTime4     6.26 2.29 49    0.741    11.79   2.741  0.0228
##
## Results are averaged over the levels of: fTemp
## 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
```

```
pairs(emmeans(Kda.Wrong.lm, ~fTemp), infer=T)
```

```
##  contrast          estimate   SE df lower.CL upper.CL t.ratio p.value
##  fTemp60 - fTemp70     1.72 2.29 49    -3.80     7.25   0.753  0.7330
##  fTemp60 - fTemp80     3.95 2.29 49    -1.58     9.47   1.727  0.2054
##  fTemp70 - fTemp80     2.23 2.29 49    -3.30     7.75   0.974  0.5967
##
## Results are averaged over the levels of: fTime
## 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
```

Chapter 13: Modelling count data using the Poisson distribution

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

- The nature of count data
- Example - modelling the number of R packages
- The Poisson Distribution
- The generalized linear model (GLM)
- GLM re-analysis of the example
- The quasi-Poisson correction
- Interpretting the fitted GLM
- Relevant R-code.

Section 13.1 The nature of count data

The nature of count data

In many studies the **response** variable will be a count.

A count variable is one where the measurement is the count of the number of times some event occurs. Obviously, it can only take values that belong to the non-negative integers $\{0,1,2,3...\}$.

In statistical parlance, a count variable is an example of a *discrete* variable, since the values it can take are discretely separated.

(In contrast, a variable with a Normal Distribution is an example of a *continuous variable*, since if can take any value on a continuum.)

The nature of count data...

In this course we shall encounter three types of count data:

1. Counts of the number of "events" occurring, where ideally, the events occur independently of one another and with no specific upper limit on the maximum number.
2. Counts of the number of "successes"[1] from a fixed number of trials. E.g., the number of Heads from tossing a coin 10 times. In this case, the response variable y is the proportion of successes (see Chapter 15).
3. Counts of the number of items in a category. E.g., The count of A, B, and C grades in the course.

In this Chapter we focus on the first of the above types. Some examples are given below.

[1] Here, "successes" is used generically. E.g., It could be the number of females in a class of 146.

The nature of count data...

Some examples of counting the number of events occurring

- The success of a marketing campaign can be assessed by analysing the change in number of purchases.
- In epidemiology, the spread of an infectious disease can be modelled by observing the number of those infected each day.
- In operations research, a business can run more efficiently by being able to predict the number of customer interactions in a day, and hence the number of staff required.
- In ecology, the success of habitat restoration can be assessed by gathering counts of the number of animals before and after the restoration.
- The efficacy of fluoridation can be assessed by counting the number of tooth cavities in the mouths of subjects.

Question: For which of the above scenarios would it be reasonable to assume that the events being counted are occurring independently?

Analysis of count data

Why do we care?

In some situations linear regression can be successfully used to analyse count data. In fact, you may have already seen this done in this course. However, these are special cases.[2]

Linear regression is not generally applicable to count data. This is demonstrated in the example below.

[2] We will soon see an example where an inappropriate application of a linear regression model to count data was responsible for the most expensive human mistake in history (up to that point in time).

Section 13.2
Example: Number of R packages submitted to the Comprehensive R Archive Network (CRAN)

Submissions to CRAN

In the following example we will model a count variable (number of R packages submitted to the CRAN over the years) using what we know to date: the *linear model* via the function `lm`.

We will contrast this to a more general and more appropriate way of modelling counts: the *generalized linear model* (GLM) via the function `glm`.

Along the way you will be introduced to a new distribution, the Poisson distribution.

Submissions to CRAN...

As R is an open-source programming environment, people are encouraged to submit software packages used in their research for others to access via the CRAN[3], which was created in 2005. We would like to understand the nature of its growth. With this in mind, the number of new packages submitted in each year, from 2005 to 2016, has been recorded:

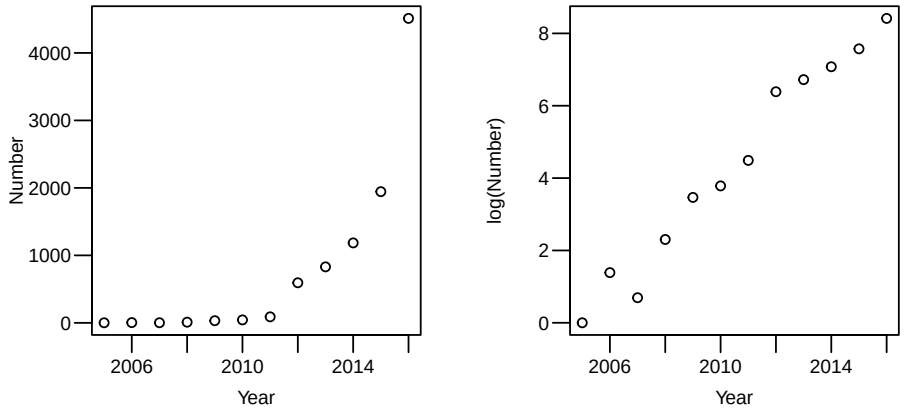
```
> CRAN.df = read.table("Data/CRAN.txt", header=T)
> CRAN.df
   Year Number
1  2005      1
2  2006      4
3  2007      2
4  2008     10
5  2009     32
6  2010     44
7  2011     89
8  2012    594
9  2013    830
10 2014   1185
11 2015   1944
12 2016   4512
```

[3] You probably downloaded your R installation from the CRAN.

Submissions to CRAN...

Plotting the data

```
> ## One-by-two figure layout
> par(mfrow=c(1,2))
> ## Scatter plot using raw y
> plot(Number ~ Year, data = CRAN.df)
> ## Scatter plot using log y
> plot(log(Number) ~ Year, data = CRAN.df)
```

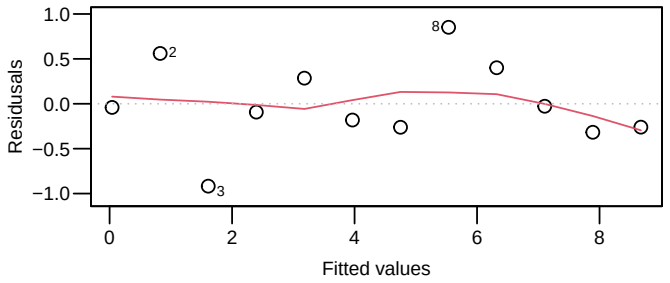


Submissions to CRAN...

Analysis via lm

The relationship between year and number of submissions looks reasonably linear on the log scale, so we'll fit a linear model to $\log(Y)$.

```
> CRAN.fit = lm(log(Number) ~ Year, data = CRAN.df)
> plot(CRAN.fit,which=1)
```

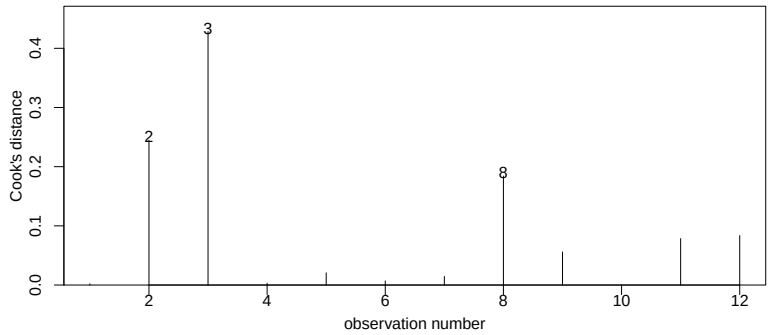


No concerns.

Submissions to CRAN...

Analysis via `lm`

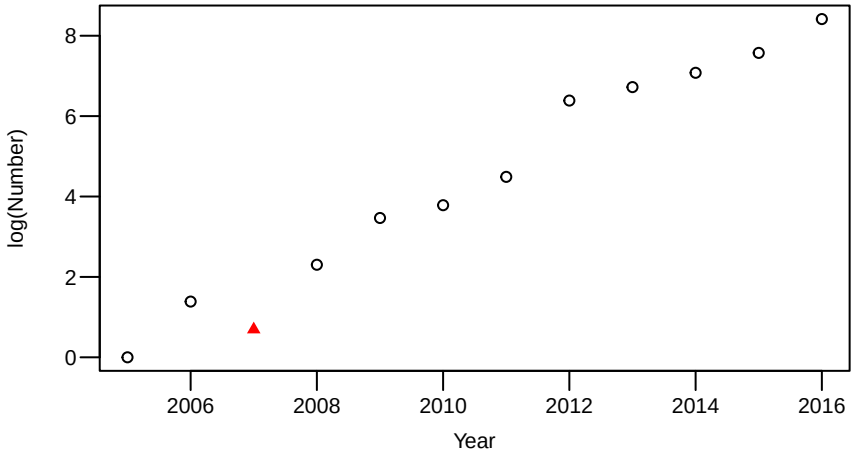
```
> cooks20x(CRAN.fit)
```



The Cook's distance for observation 3 exceeds our threshold of 0.4. However, with only 12 observations this is perhaps not of great concern.

Submissions to CRAN...

Analysis via `lm`...



Indeed, the third observation in red does not seem to be influential in the exploratory plots. So, we will keep it in the model.

Submissions to CRAN...

Analysis via `lm`...

```
> summary(CRAN.fit)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.574e+03  8.245e+01  -19.09 3.39e-09 ***
Year         7.849e-01  4.101e-02   19.14 3.30e-09 ***
---
Residual standard error: 0.4904 on 10 degrees of freedom
Multiple R-squared:  0.9734,Adjusted R-squared:  0.9708
F-statistic: 366.4 on 1 and 10 DF,  p-value: 3.295e-09
```


Submissions to CRAN...

Analysis via `lm`...

Back-transform to get the multiplicative effect of year.

```
> ## Estimated annual multiplier
> exp(CRAN.fit$coef["Year"])
    Year
2.192237
> ## Confidence interval
> exp(confint(CRAN.fit))
                2.5 %  97.5 %
(Intercept) 0.00000 0.00000
Year        2.00081 2.40198
```

So, the Executive Summary would have said that the median annual number of submissions to CRAN multiplies by between 2.00 to 2.40 times each year. In other words, it increases by between 100% and 140% per annum.

Submissions to CRAN...

Analysis via `lm`...

We will use this model to estimate the median of the distribution of the number of submissions in 2017.

```
> predCRAN.df = data.frame(Year = 2017)
> pred2017 = predict(CRAN.fit, predCRAN.df, interval = "confidence")
>
> ## Prediction on the log scale
> pred2017
      fit     lwr      upr
1 9.45978 8.78731 10.13225
> ## Back-transform for the median of the number of submissions in 2017
> exp(pred2017)
       fit      lwr      upr
1 12833.05 6550.586 25140.85
```

So, the Executive Summary would have said that the median of the number of submissions to CRAN in 2017 is between 6550 and 25100.

Submissions to CRAN...

Comments on the analysis via `lm`...

There is a well known saying:

"If you only have a hammer, you tend to see every problem as a nail." – Abraham Maslow [Replace "hammer" with "linear model", and "a nail" with "normally distributed"]

We were able to do an analysis with `lm` (our hammer) by working with the logged data (our normally distributed nails). But, was this really a sensible way to analyse the CRAN data?

Before we answer this question, we'll look at a different methodology that is tailored for the analysis of count data. The first thing we need is a more appropriate type of distribution to replace the normal distribution. One that is tailored for describing the distribution of count data.

Section 13.3
The Poisson Distribution

The Poisson distribution

Statisticians often use the Poisson distribution to describe the behaviour of counts of events that occur randomly in space or time, such as:

- The number of alcohol related arrivals at a hospital.
- The number of dolphins in a pod.
- The number of pods of dolphins sighted during an aerial survey.
- The annual number of road fatalities.
- The annual number of fatal road accidents.

The Poisson is a distribution that takes values on the non-negative integers {0,1,2,3,...} and it has no upper limit.

The Poisson distribution is specified by a single parameter, its mean (i.e., expected value). We will write, Pois(μ) to denote a Poisson distribution with mean of μ.

The Poisson distribution...

The probability that the non-negative integer value y will be observed if generated by a Pois(μ) distribution is given by the following formula:

$$\Pr(y) = \frac{\exp(-\mu)\mu^y}{y!} .$$

where $y! = \texttt{factorial(y)} = 1 \times 2 \times ... \times (y-1) \times y$ (and $0! = 1$).

For $y = 12$ and $\mu = 9.61$, this could be calculated in R using the code

```
> y=12; mu=9.61
> (exp(-mu)*mu^y)/factorial(y)
[1] 0.08685078
```


The Poisson distribution...

In R, the in-built function dpois(y, μ) calculates these Poisson probabilities. E.g., the probability that $y = 12$ will be observed from a Pois(9.61) distribution is

```
> dpois(12,9.61)
[1] 0.08685078
```

You can also generate random Poisson values. E.g., here are 20 random values from a Pois(10) distribution,

```
> rpois(20,10)
 [1]  9  9  8 11 11  9 15  8 11 12 10  9 16  9  6  8 16 12  7 15
```


The Poisson distribution...

An example

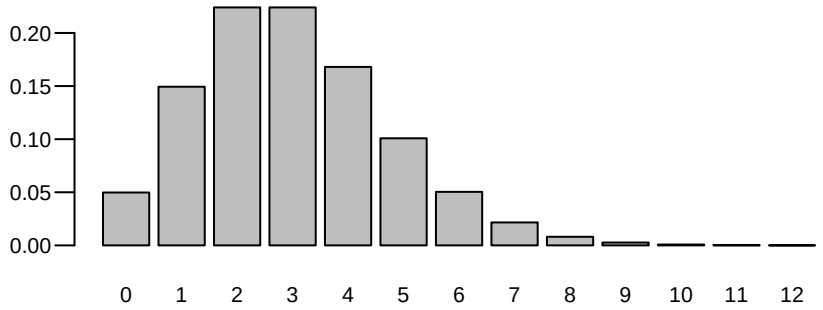
Suppose that a hospital *expects* to handle 3 victims of alcohol related mis-adventure on a Friday night.

Assuming that the *actual* number on any given Friday night can be described by a Poisson distribution with a mean of 3 (Pois(3)), what does the distribution of the number of alchohol victims look like?

The Poisson distribution...

An example...

```
> barplot(dpois(0:12,3),names=0:12)
```



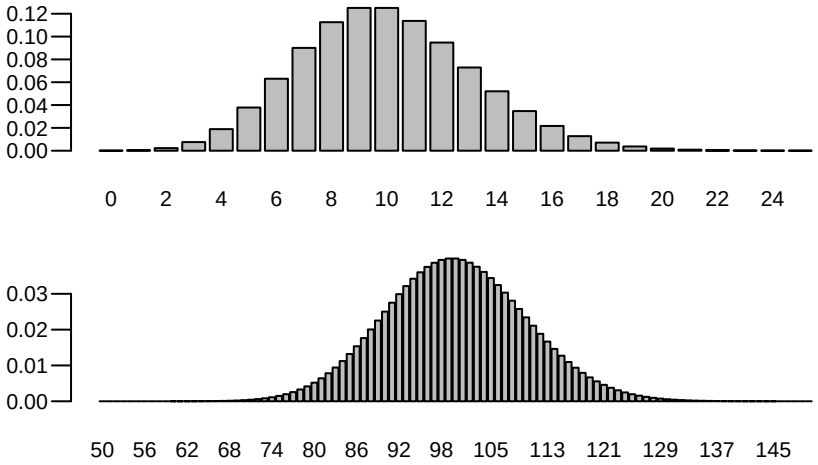
The probabilities for the number of victims from 0 to 20 are:

```
> round(dpois(0:20,3),6)
 [1] 0.049787 0.149361 0.224042 0.224042 0.168031 0.100819 0.050409 0.021604
 [9] 0.008102 0.002701 0.000810 0.000221 0.000055 0.000013 0.000003 0.000001
[17] 0.000000 0.000000 0.000000 0.000000 0.000000
```

The Poisson distribution...

More Poisson distributions

```
> par(mfrow=c(2,1)) ## Two-by-One figure layout
> barplot(dpois(0:25,10),names=0:25) ## Pois(10)
> barplot(dpois(50:150,100),names=50:150,las=1) ## Pois(100)
```



The Poisson distribution...

Properties of the Poisson distribution

- Variability increases with the mean μ. In fact, the variance of a Pois(μ) is also μ. That is, if Y is Pois(μ) distributed then

$$\text{Var}(Y) = \text{E}(Y) = \mu$$

- The Poisson distribution is right-skewed for small values of μ, but looks very much like a (discretised) normal distribution for large μ.

The Poisson distribution...

Properties of the Poisson distribution...

The Poisson distribution is a good distribution to describe count data provided that the events being counted are independent.

- The number of alcohol related arrivals at a hospital could be described by a Poisson distribution provided that the arrivals occur independently...does this seem reasonable to you???
- What about the number of dolphins in a pod?
- What about the number of pods of dolphins sighted during an aerial survey?
- What about annual number of road fatalities?
- What about annual number of fatal road accidents?

- Would it be sensible for an insurance company to assume that the number of earthquake damage claims was Poisson distributed?

Section 13.4
The generalized linear model (GLM)

The generalized linear model

Count data are not normally distributed, so we need to use a model that replaces the normality assumption with something more reasonable, such as assuming the data are Poisson distributed.

We will use a class of models called generalized linear models. They generalize the standard linear model (for normal data) by making it applicable to other types of data.

The linear model of Chaps 1-12 is a special case of a GLM.

The Poisson regression GLM for the CRAN data

We will redo this analysis using a GLM. That is, the model will now assume that the number of packages in a given year is a Poisson random variable:

$$Y \sim \text{Poisson}(\mu)$$

where the Poisson parameter ($\mu = E[Y]$) changes with respect to an x variable as follows:

$$\mu = \exp(\beta_0 + \beta_1 x).$$

Here, x is year and Y is the number of CRAN submissions in that year.

Since β_0 and β_1 can be negative or positive then so can $\beta_0 + \beta_1 x$, but $\mu = \exp(\beta_0 + \beta_1 x) > 0$, as required.

The relationship $\mu = \exp(\beta_0 + \beta_1 x)$ can equivalently be expressed as

$$\log(\mu) = \log E[Y|x] = \beta_0 + \beta_1 x$$

and so sometimes people call this log-linear modelling.

Fitting a generalized linear model in R

glm function

Fitting a generalized linear model is the easy part. We simply use the `glm` function instead of `lm`, and instruct `glm` that the response variable is Poisson distributed by giving it the `family = poisson` argument.

NOTE: Although only a simple change to the R code is required, the methodology "under the hood" is very different. It is no longer based on sums of squares, and instead uses a technique called maximum likelihood.

Maximum likelihood is the most widely used tool in statistical estimation. Linear regression (of normally distributed data) is a special case of maximum likelihood.

For more on maximum likelihood, see STATS 310, 330 and 730.

Submissions to CRAN using Poisson regression

Analysis via glm

```
> CRAN.gfit = glm(Number ~ Year, family = poisson, data = CRAN.df)
> summary(CRAN.gfit)
```

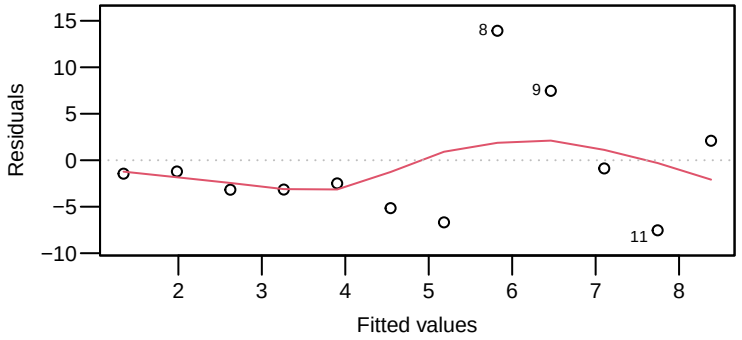
```
Call:
glm(formula = Number ~ Year, family = poisson, data = CRAN.df)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.282e+03  1.384e+01  -92.64   <2e-16 ***
Year         6.401e-01  6.868e-03   93.20   <2e-16 ***
---
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 19374.51  on 11  degrees of freedom
Residual deviance:   402.61  on 10  degrees of freedom
```

Submissions to CRAN using Poisson regression...

Assumption checks

First, check the residuals to see if they look random.

```
> plot(CRAN.gfit, which = 1)
```



Hmmm, a strange pattern, though no clear systematic trend in the residuals.

Submissions to CRAN using Poisson regression...

Assumption checks

It is not as easy to check the assumptions of a GLM compared to a linear model.

In the plot of residuals vs fitted values:

- The fitted values on the x-axis are values of the so-called "linear predictor". That is, they are the fitted values $\hat{\beta}_0 + \hat{\beta}_1 x$.
- The residuals on the y-axis are not standard residuals. They are "standardized residuals" and should be approximately normally distributed with mean of 0 and variance of 1 if the Poisson assumption is satisfied **and** $\mu > 5$ – see STATS 330 for more information. Clearly, the Poisson assumption is not valid here. We shall see a cure for this below.

STATS 330 covers checking of GLMs in greater detail. In particular, it demonstrates the use of simulations to show us when a model is violating assumptions.

Submissions to CRAN using Poisson regression...

Assumption checks...

There is another **very important check** that is essential:

- Checking the Poisson assumption that the variances of the counts are equal to their means.

If the model is correct, then the residual deviance (printed near the bottom of the `summary` output) is approximately distributed[4] as chi-squared χ^2_q where q is the degrees of freedom.[5]

If the residual deviance is unreasonably large then we have evidence that the model is not valid.

In this example, the residual deviance is 402.61, with 10 df. The P-value is

```
> 1 - pchisq(402.61,10)
[1] 0
```

If this P-value is small then we conclude that our model is not adequate. That is clearly the case with the CRAN data.

[4] Subject to μ not being too small for most observations.

[5] q is the number of observations minus the number of parameters estimated.

Submissions to CRAN using Poisson regression...

quasi-Poisson correction

The residual deviance check rejected the Poisson GLM fitted to the CRAN data.

Fortunately there is a very simple fix.

If the model is found to be inadequate, there are two possible reasons:

- We do not have the right explanatory terms in the model.
- The Poisson assumption is wrong.

If the residual plot looks OK, then we can rule out the first possibility. In this example, it looks like we have a problem with the Poisson assumption.

The fix is very simple – we just replace `family = poisson` with `family = quasipoisson`.[6]

[6] Those of you who go on to do more advanced STATS courses will encounter advanced methods to handle count data that do not satisfy the Poisson assumption.

Submissions to CRAN using Poisson regression...

quasi-Poisson correction...

The `family = quasipoisson` specification is saying that the data have different variance than if they were Poisson distributed.

Recall, if Y is Poisson then $\text{Var}(Y) = \text{E}(Y)$. If Y is quasi-Poisson then we assume

$$\text{Var}(Y) \propto \text{E}(Y), \text{or}$$
$$\text{Var}(Y) = k\,\text{E}(Y)$$

When the data have more variance than we would expect under a Poisson assumption, the consequence of using `family = quasipoisson` is that the standard errors of the estimated coefficients are increased, compared to using `family = poisson`.

Submissions to CRAN using Poisson regression...

Without quasi-Poisson correction

Recall:

```
> CRAN.gfit= glm(Number~Year,family=poisson,data=CRAN.df)
> summary(CRAN.gfit)
```

```
Call:
glm(formula = Number ~ Year, family = poisson, data = CRAN.df)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.282e+03  1.384e+01  -92.64   <2e-16 ***
Year         6.401e-01  6.868e-03   93.20   <2e-16 ***
---
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 19374.51  on 11  degrees of freedom
Residual deviance:   402.61  on 10  degrees of freedom
```


Submissions to CRAN using Poisson regression...

With quasi-Poisson correction

Compare with:

```
> CRAN.quasigfit = glm(Number ~ Year, family = quasipoisson, data = CRAN.df)
> summary(CRAN.quasigfit)
```

```
Call:
glm(formula = Number ~ Year, family = quasipoisson, data = CRAN.df)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.282e+03  8.889e+01  -14.42 5.09e-08 ***
Year         6.401e-01  4.411e-02   14.51 4.81e-08 ***
---
(Dispersion parameter for quasipoisson family taken to be 41.25925)

    Null deviance: 19374.51  on 11  degrees of freedom
Residual deviance:   402.61  on 10  degrees of freedom
```

The estimated coefficients are not changed. But, the z-values[7] in the `summary` output will decrease in magnitude, and the P-value will increase.

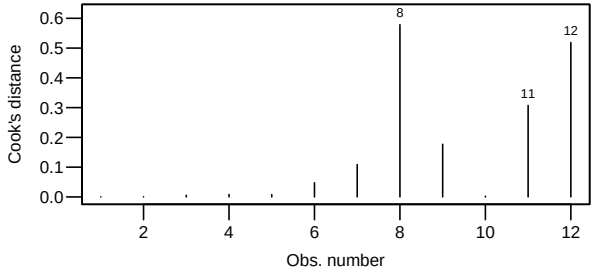
[7] In the quasi-Poisson output the z-values are renamed t-values due to it using an estimated variance.

Submissions to CRAN using Poisson regression...

Influence

Let's check the influence of the observations.

```
> plot(CRAN.quasigfit, which = 4)
```



We see that Observations 8 and 12 exceed our 0.4 threshold. However, this has to be interpretted with caution. Unlike the normal linear regression, in a Poisson regression the observations with higher values of μ are expected to have higher influence than those with lower values of μ.

Section 13.5
Interpretting the GLM output

Submissions to CRAN using Poisson regression...

Inference

Let's look at the estimated rate of annual increase in submissions of R packages to the CRAN.

Recall that our model for the expected number of submissions to the CRAN is

$$\mu = \exp(\beta_0 + \beta_1 \times Year) = \exp(\beta_0) \times \exp(\beta_1)^{Year}$$

so we need to exponentiate our estimate and its confidence interval.

The annual multiplicative effect on μ is $\exp(\beta_1)$, and our estimated value of this effect is $\exp(\hat{\beta}_1)$:

```
> ## The estimated annual multiplier
> exp(CRAN.quasigfit$coef["Year"])
    Year
1.896614
```


Submissions to CRAN using Poisson regression...

Confidence intervals for effects

We can still use the function `confint()` to get confidence intervals for the parameters.

```
> exp(confint(CRAN.quasigfit))
Waiting for profiling to be done...
                2.5 %   97.5 %
(Intercept) 0.000000 0.000000
Year        1.745819 2.075781
```

Sometimes you might see `confint.default()` used instead.

```
> exp(confint.default(CRAN.quasigfit))
                2.5 %   97.5 %
(Intercept) 0.000000 0.000000
Year        1.739517 2.067898
```

They are doing slightly different things, based on two different approximations. In most cases they give very similar results.

It is generally best to use `confint()` rather than `confint.default()`. See STATS 730 for the reason why.

Submissions to CRAN using Poisson regression...

Confidence intervals for effects...

So, our Executive Summary would say that the expected annual number of submissions to CRAN multiplies by between 1.75 and 2.08 times each year. That is, it increases by between 75% and 108% per year.

[This compares to a median multiplier of between 2.00 and 2.40 times each year from the naive `lm` fit to $\log(y)$.]

Note that the GLM model is making statements about the **mean** rather than the **median**. This is because the GLM does not transform y.

Submissions to CRAN using Poisson regression...

Confidence interval for expected number of submissions

And finally, we will use the quasi-Poisson model to estimate the expected number of submissions in 2017. We first calculate the confidence interval on the linear predictor scale, and then exponentiate.

```
> pred2017.quasi=predict(CRAN.quasigfit, predCRAN.df, se.fit=TRUE)
> # CI for log mean
> lower = pred2017.quasi$fit-1.96*pred2017.quasi$se.fit
> upper = pred2017.quasi$fit+1.96*pred2017.quasi$se.fit
> #CI for mean value
> pred2017.ci.mean=exp(c(lower, upper))
> pred2017.ci.mean
       1        1
 6626.596 10383.014
```

So our Executive Summary would say that the expected number of submissions to CRAN in 2017 is between 6600 and 10400.

This compares to the estimated median number being between 6600 and 25100 from the naive `lm` fit to $\log(y)$.

Submissions to CRAN using Poisson regression...

Confidence interval for expected number of submissions...

The above calculations are done for us using the `predictGLM` function.

The confidence interval on the linear predictor scale is

```
> predictGLM(CRAN.quasigfit, predCRAN.df)
***Estimates and CIs are on the link scale***
       fit      lwr      upr
1 9.023387 8.798851 9.247922
```

and on the scale of the response variable is

```
> predictGLM(CRAN.quasigfit, predCRAN.df,type="response")
***Estimates and CIs are on the response scale***
      fit      lwr      upr
1 8294.82 6626.623 10382.97
```

NOTE: For GLMs, it is not straightforward to obtain prediction intervals for new values of the response. This is left to STATS 330.

Section 13.6
Closing Remarks

Linear vs generalized linear model

The linear model that we fitted to the logged CRAN data on slide 12 has some undesirable properties.

- Logged count data do not have equal variance.[8]
- Logged count data are highly variable for small expected counts.
- Logged count data can not handle a zero count, since $\log(0)$ is negative infinity.

The influence plot from the LM fitted to the logged CRAN data showed that the 2nd and 3rd observations were the most influential. This is very dangerous, as these are the most unreliable data points.

[8] In fact, if Y is Poisson, then $\text{Var}(\log(Y)) = \infty$!!!

Linear vs generalized linear model

A GLM is a more natural way to model count data.

- It does not transform y.
- Inference is about **means** (rather than medians).
- It accounts for the fact that the variance of Y increases with the mean.
- It implicitly allows the higher counts to have more influence.[9]

The GLM better models the CRAN data:

- The plot of fitted values versus residuals clearly shows the sudden increase at year 8.
- The influence plot better shows the true influence of each observation.
- It results in narrower confidence intervals for the expected annual increase and expected future counts.

[9] This is because the coefficient of variation, CV, is greater for the smaller counts (the CV is the noise-to-signal ratio, $\text{sd}(Y)/\text{E}[Y] = \frac{1}{\sqrt{\mu}}$).

Linear vs generalized linear model

The two key points of difference are:

- $Y|x$ is assumed to be a Poisson distributed count variable (rather than a normally distributed continuous variable).
- Instead of our model being of additive form

$$\text{E}[Y|x] = \beta_0 + \beta_1 x + \dots$$

it is of multiplicative form

$$\text{E}[Y|x] = \exp(\beta_0 + \beta_1 x + \dots)$$

or equivalently

$$\log(\text{E}[Y|x]) = \beta_0 + \beta_1 x + \dots$$

NOTE: The GLM transforms the linear model rather than transforming Y.

Linear on log scale vs generalized linear model

In Chapter 6 we logged our response variable before fitting the linear model. This assumes that the logged reponses are normally distributed and that

$$\text{E}[\log(Y|x)] = \beta_0 + \beta_1 x + \dots$$

We saw that, under the above model

$$\text{E}[Y|x] \neq \exp(\beta_0 + \beta_1 x + \dots)$$

and that was why we had to make inference about population medians.

In comparison, in the GLM the order of log and E is reversed

$$\log(\text{E}[Y|x]) = \beta_0 + \beta_1 x + \dots$$

Section 13.7
Relevant R-code

Most of the R-code you need for this chapter

We started by identifying what our response is – in this case a count. Therefore our response data is definitely not from a normal distribution. As Y is a count variable we propose to model it using the Poisson distribution.

Instead of using `lm` we use `glm`, and add `family=poisson`.

```
> CRAN.gfit= glm(Number~Year,family=poisson,data=CRAN.df)
> summary(CRAN.gfit)
```

Look at fitted vs residual plot and make sure there's no trend you've missed and/or atypical observations.

```
> plot(CRAN.gfit, which = 1)
```

Do **NOT** check for normality!
Influence can still be assessed using `cooks20x`, but be aware that larger counts will naturally tend to be more influential than smaller counts.

Most of the R-code you need for this chapter...

To check if the data are consistent with the Poisson model you use

```
> 1 - pchisq(CRAN.gfit$deviance,CRAN.gfit$df.residual)
```

and if the P-value is very small (i.e., less that 0.05) then you can to deal with this using the quasi-Poisson model:

```
> CRAN.quasigfit = glm(Number ~ Year, family = quasipoisson, data = CRAN.df)
```

Once you've chosen your final model you can interpret the effect of the variables as a multiplicative change in the **mean** or expected value:

```
> exp(confint(CRAN.quasigfit))
```

If you wish to estimate a 95% CI for an expected value use `predictGLM`, or do it the hard way as follows:

```
> pred2017.quasi=predict(CRAN.quasigfit, predCRAN.df,se.fit=TRUE)
> # CI for log mean
> lower = pred2017.quasi$fit-1.96*pred2017.quasi$se.fit
> upper = pred2017.quasi$fit+1.96*pred2017.quasi$se.fit
> #CI for mean value
> pred2017.ci.mean=exp(c(lower, upper))
> pred2017.ci.mean
```

Chapter 14: Poisson modelling of count data: Two examples.

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about using a Poisson regression GLM to model:

- Earthquake frequencies using earthquake magnitude (numeric) and location (factor) as explanatory variables.
- Snapper counts using location (factor) and reservation status (factor) as explanatory variables.

Section 14.1
Example 1: Earthquake frequency

Earthquake magnitudes

The Gutenberg-Richter law

The Gutenberg-Richter law says that the expected number of earthquakes in a given period of time decreases multiplicatively with their magnitude. The formula is

$$\log_{10} N = a - bM$$

where N is the expected number of earthquakes of magnitude M or more on the Richter scale. Here, a and b are unknown parameters.

The Richter scale is logarithmic (base 10). So, for example, an earthquake that registers 5.0 on the Richter scale has a shaking amplitude 10 times that of an earthquake that registers 4.0. It can be shown that this corresponds to 30 times the release of energy.

FYI, the most powerful earthquake ever recorded was in Chile in 1960, measuring 9.5 on the Richter scale. The largest known seismic events on earth have been caused by asteroid impact, exceeding 13 on the Richter scale.

Earthquake magnitudes...

The Gutenberg-Richter law...

After applying a healthy dash of calculus, this formula can be re-expressed in a form that is more familiar to us

$$E[Y|x] = \exp(\beta_0 + \beta_1 x)$$

where Y is the number of earthquakes having magnitude between $x - \delta$ and $x + \delta$.[1]

E.g., if $x = 6$ and $\delta = 0.125$ then Y is the number of earthquakes between 5.875 and 6.125 in magnitude.

The above formula should look familiar. It is the one we use for a Poisson regression GLM when there is a single numeric explanatory variable x.

[1] In the above formula, β_0 and β_1 depend on a, b and δ in a complicated way that we are not going to concern ourselves with.

Sthn California and Washington earthquakes, 1987–2019

The research question is to quantify the rate of decrease in earthquake frequency (with increasing magnitude) in both Southern California (SC) and the State of Washington (WA), and to assess whether these rates are the same.

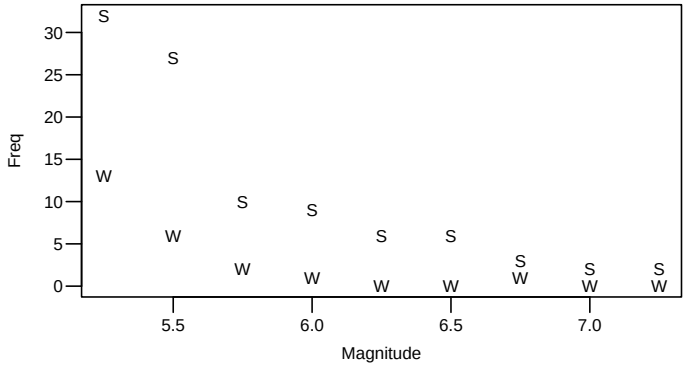
```
> Quakes.df=read.table("Data/EarthquakeMagnitudes.txt",header=TRUE)
> Quakes.df$Locn=as.factor(Quakes.df$Locn)
> subset(Quakes.df,subset=c(Locn=="SC"))[1:4,] #Print first 4 SC observations
  Locn Magnitude Freq
1   SC      5.25   32
2   SC      5.50   27
3   SC      5.75   10
4   SC      6.00    9
> subset(Quakes.df,subset=c(Locn=="WA"))[1:4,] #Print first 4 WA observations
   Locn Magnitude Freq
10   WA      5.25   13
11   WA      5.50    6
12   WA      5.75    2
13   WA      6.00    1
```

Here we have one explanatory variable that is a factor variable, and another that is numeric. We have seen this before in Chapter 8, and we handle it in much the same way as before, but using `glm` instead of `lm`.

Sthn CA and WA earthquakes, 1987–2019...

Plotting the data

```
> plot(Freq~Magnitude,data=Quakes.df,pch=substr(Locn,1,1))
```



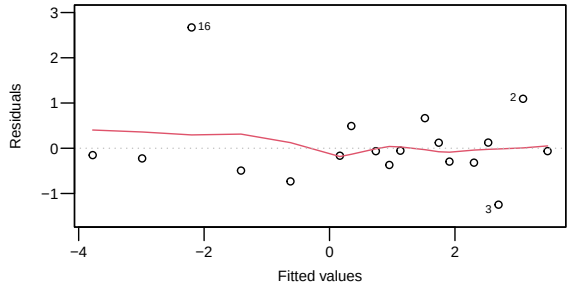
The data look consistent with an exponential decay. It is not clear if the rates of exponential decay are the same.

Sthn CA and WA earthquakes, 1987–2019...

Model fit and assumption checking

Recall from Chapter 8 that we fit the interaction model first, and then simplify if possible.

```
> Quake.gfit = glm(Freq ~ Locn * Magnitude, family = poisson,
+                  data = Quakes.df)
> plot(Quake.gfit, which = 1)
```

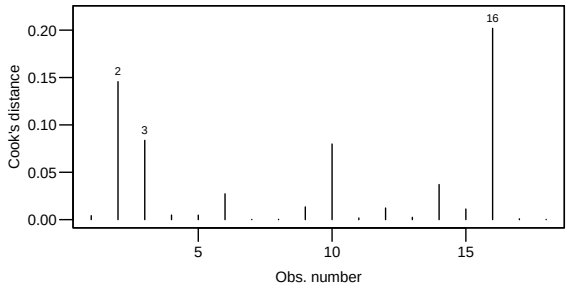


Looks OK, notwithstanding that observation 16 has a high residual. This observation has low expected value (approx $\exp(-2)$), so this residual is not reliable and no cause for concern.

Sthn CA and WA earthquakes, 1987–2019...

Checking influence

```
> plot(Quake.gfit, which = 4)
```



No influential points.

Sthn CA and WA earthquakes, 1987–2019...

Summary output

```
> summary(Quake.gfit)
```

```
Call:
glm(formula = Freq ~ Locn * Magnitude, family = poisson, data = Quakes.df)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      11.6923     1.1762   9.941  < 2e-16 ***
LocnWA            7.3923     3.9500   1.871   0.0613 .
Magnitude        -1.5648     0.2055  -7.616 2.61e-14 ***
LocnWA:Magnitude -1.5884     0.7199  -2.206   0.0274 *
---
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 176.1767  on 17  degrees of freedom
Residual deviance:   8.2295  on 14  degrees of freedom
```

The residual deviance of 8.23 is less than the 14 residual degrees of freedom, so there won't be any problems with the variance check.

```
> 1 - pchisq(8.23, 14)
[1] 0.8770025
```


Sthn CA and WA earthquakes, 1987–2019...

The interaction term *P*-value is 0.027, so we conclude that the effect of magnitude is different at the two locations.

Next, lets quantify these rates. First, for Southern California:

```
> Quake.cis = confint(Quake.gfit)
Waiting for profiling to be done...
> exp(Quake.cis[3,])
    2.5 %    97.5 %
0.1374743 0.3082437
> ## To interpret as percentage decreases
> 100*(1-exp(Quake.cis[3,]))
   2.5 %   97.5 %
86.25257 69.17563
```


Sthn CA and WA earthquakes, 1987–2019...

We change the baseline to get the rate for Washington:

```
> Quakes.df$Locn2=factor(Quakes.df$Locn,levels=c("WA","SC"))
>
> Quake2.gfit=glm(Freq~Locn2*Magnitude,family=poisson,data=Quakes.df)
> (Quake.WA.ci = exp(confint(Quake2.gfit)[3,]))
Waiting for profiling to be done...
      2.5 %      97.5 %
0.009077661 0.140175445
> ## To interpret as percentage decreases
> 100*(1 - Quake.WA.ci)
   2.5 %   97.5 %
99.09223 85.98246
```

Sthn CA and WA earthquakes, 1987–2019...

Executive Summary

Our Executive Summary would say that the rate of decline in the frequency of earthquakes (with increasing magnitude) is more rapid in WA than CA.

In WA, there is a 86 to 99% drop in the expected number of earthquakes for a one unit increase in their magnitude on the Richter scale. In CA, the decrease is between 69 to 86%.

Section 14.2
Example 2: Snapper counts in and around marine reserves

Snapper counts in and around marine reserves

Baited underwater video (BUV) is an established tool for counting fish such as snapper.

BUV was used at two coastal locations in New Zealand, Leigh and Hahei. Each location has a marine reserve. The BUV was deployed at sites inside the marine reserve, and at sites just outside the reserve. There was a total of 18 sites.

The three variables measured were

- Count of snapper
- Location (Leigh or Hahei)
- Reservation status (Yes or No)

It was of interest to explore the count of snapper with regard to location and reservation status.

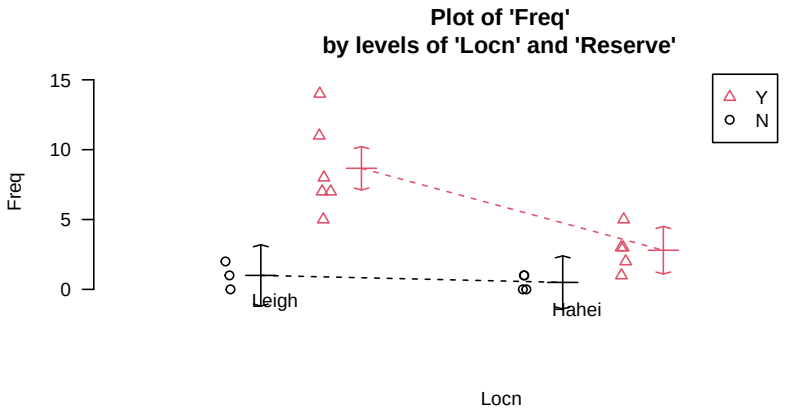
Snapper counts in and around marine reserves...

```
> Snap.df=read.table("Data/SnapperCROPvsHAHEI.txt",header=TRUE)
> with(Snap.df,{Locn=as.factor(Locn); Reserve=as.factor(Reserve)})
> Snap.df
    Locn Reserve Freq
1  Leigh       N    2
2  Leigh       N    1
3  Leigh       N    0
4  Leigh       Y    5
5  Leigh       Y   11
6  Leigh       Y    7
7  Leigh       Y    8
8  Leigh       Y    7
9  Leigh       Y   14
10 Hahei       N    1
11 Hahei       N    0
12 Hahei       N    1
13 Hahei       N    0
14 Hahei       Y    3
15 Hahei       Y    2
16 Hahei       Y    1
17 Hahei       Y    5
18 Hahei       Y    3
```

Snapper counts in and around marine reserves...

Plotting the data

```
> interactionPlots(Freq ~ Locn + Reserve, data = Snap.df)
```



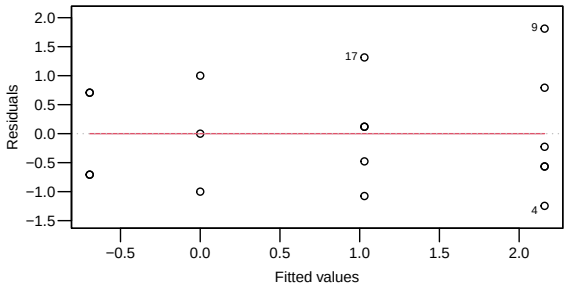
NOTE: Parallel lines no longer corresponds to lack of interaction. **Why?**

Snapper counts in and around marine reserves...

Model fit and assumption checking

There are two categorical explanatory variables, so we follow the steps from Chapter 12. First, we fit an interaction model:

```
> Snap.glm = glm(Freq ~ Locn*Reserve, family = poisson, data = Snap.df)
> plot(Snap.glm, which = 1)
```

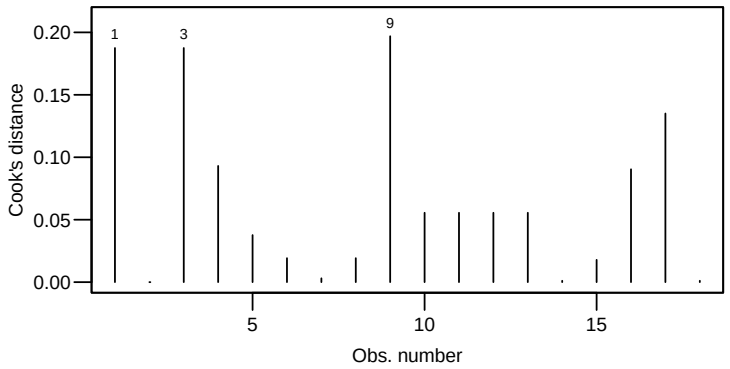


Looks fine.

Snapper counts in and around marine reserves...

Influence checking

```
> plot(Snap.glm, which = 4)
```



No overly influential points.

Snapper counts in and around marine reserves...

Assumption checking...

```
> summary(Snap.glm)
```

```
Call:
glm(formula = Freq ~ Locn * Reserve, family = poisson, data = Snap.df)

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)        -0.6931     0.7071  -0.980   0.3270
LocnLeigh           0.6931     0.9129   0.759   0.4477
ReserveY            1.7228     0.7559   2.279   0.0227 *
LocnLeigh:ReserveY  0.4367     0.9612   0.454   0.6496
---
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 70.453  on 17  degrees of freedom
Residual deviance: 14.678  on 14  degrees of freedom
```

The residual deviance is 14.678 on 14 dof. No problems there.

```
> 1 - pchisq(14.678, 14)
[1] 0.4005141
```

Snapper counts in and around marine reserves...

Apply Occam's razor

We see that the interaction between Location and Reserve is not required, so lets redo the `glm` without the interaction.

```
> Snap2.glm = glm(Freq ~ Locn + Reserve, family = poisson, data = Snap.df)
> summary(Snap2.glm)
```

```
Call:
glm(formula = Freq ~ Locn + Reserve, family = poisson, data = Snap.df)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.9491     0.4884  -1.943 0.051990 .
LocnLeigh     1.0894     0.2845   3.829 0.000128 ***
ReserveY      2.0105     0.4646   4.328 1.51e-05 ***
---
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 70.453  on 17  degrees of freedom
Residual deviance: 14.879  on 15  degrees of freedom
```


Snapper counts in and around marine reserves...

The residual deviance still indicates no evidence of a problem:

```
> 1 - pchisq(14.879, 15)
[1] 0.4601677
```

Lets calculate some confidence intervals, remembering to exponentiate them to get the multiplicative effects of location and reservation status.

```
> (Snap.cis <- exp(confint(Snap2.glm)))
Waiting for profiling to be done...
                2.5 %     97.5 %
(Intercept) 0.1298697  0.9105143
LocnLeigh   1.7443515  5.3626745
ReserveY    3.3224830 21.3481546
```


Snapper counts in and around marine reserves...

Executive Summary

We conclude that the expected count of snapper is between 3.3 and 21.3 times as high in marine reserves than in the area just outside of the reserve.

Moreover, the Leigh location has higher expected snapper counts than Hahei, - they are between 1.7 and 5.4 times as high at Leigh.

Closing remark – use of `anova` with a GLM

In situations where we need to test a joint hypothesis (see Chapter 9) we can continue to use the `anova` function.

However, be aware that `anova` for GLMs can use several different methods for calculating the appproximate *P*-value for the joint hypothesis. We recommend using `test="Chisq"`.

By way of example:"

```
> Snap.anova=anova(Snap2.glm,test="Chisq")
> data.frame(Snap.anova) #Using data.frame removes extraneous output
        Df Deviance Resid..Df Resid..Dev     Pr..Chi.
NULL    NA       NA        17   70.45338           NA
Locn     1 22.65585        16   47.79753 1.937697e-06
Reserve  1 32.91888        15   14.87866 9.608571e-09
```

Note that the *P*-value for the reserve effect is different from that obtained from `summary(Snap2.glm)`, but both *P*-values tell the same story – a very highly significant effect of reserve.

Case Study 14.1: Decline in expected earthquake numbers with magnitude

Tou Ohone Andate - staff number 1234567

Problem

The Gutenberg-Richter law says that the expected frequency of earthquakes decreases multiplicatively with their magnitude. The formula is

$$\log_{10} N = a - bM,$$

where N is the expected number of earthquakes of magnitude M or more on the Richter scale. Here, a and b are unknown parameters.

After applying a healthy dash of calculus, this formula can be re-expressed in a form that is more familiar to us

$$E[Y|x] = \exp(\beta_0 + \beta_1 x),$$

where Y is the number of number of earthquakes of magnitude between $x - \delta$ and $x + \delta$.[1]

The variables of interest were:

- `Magnitude`: The strength of a recorded earthquake.
- `Locn`: A two-level factor which describe the location of the earthquakes.
  - It has two levels Southern California ("SC") and Washington ("WA").
- `Freq`: The number of earthquakes recorded at `Locn` with magnitude `Magnitude`.

Question of Interest

The research question is to quantify the rate of decrease in earthquake frequency (with increasing magnitude) in both California and Washington states, and to assess whether these rates are the same.

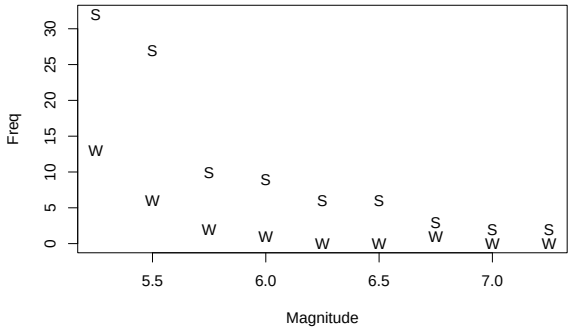
Read in and Inspect the Data

```
Quakes.df = read.table("EarthquakeMagnitudes.txt", header = TRUE)
Quakes.df$Locn=as.factor(Quakes.df$Locn)
Quakes.df
```

```
##    Locn Magnitude Freq
## 1    SC      5.25   32
## 2    SC      5.50   27
## 3    SC      5.75   10
## 4    SC      6.00    9
## 5    SC      6.25    6
## 6    SC      6.50    6
## 7    SC      6.75    3
## 8    SC      7.00    2
## 9    SC      7.25    2
## 10   WA      5.25   13
## 11   WA      5.50    6
## 12   WA      5.75    2
## 13   WA      6.00    1
## 14   WA      6.25    0
## 15   WA      6.50    0
## 16   WA      6.75    1
## 17   WA      7.00    0
## 18   WA      7.25    0
```

[1] In the above formula, β_0 and β_1 depend on a, b and δ in a complicated way that we are not going to concern ourselves with.

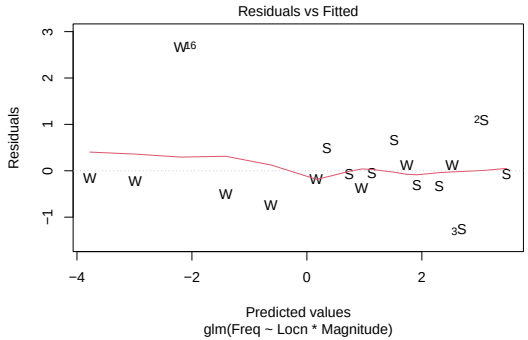

```
plot(Freq ~ Magnitude, data = Quakes.df, pch = substr(Locn, 1, 1))
```



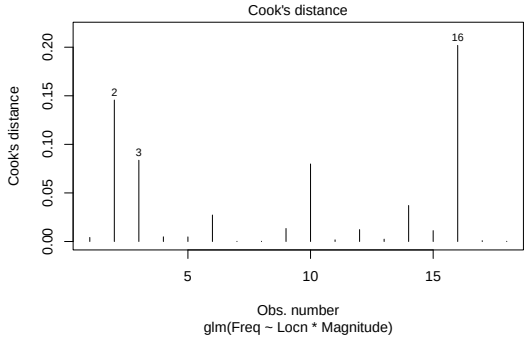
The plot is consistent with an exponential decreasing relationship between frequency and magnitude. It is unclear whether the multiplicative rate of decay is the same in the two locations.

Model Building and Check Assumptions

```
Quake.gfit = glm(Freq ~ Locn * Magnitude, family = poisson, data = Quakes.df)
plot(Quake.gfit, which = 1, pch = substr(Quakes.df$Locn, 1, 1))
```




```
plot(Quake.gfit, which = 4)
```



Observation 16 has a largish residual, but as this observation has a very small fitted value of μ this can be ignored.

```
summary(Quake.gfit)
```

```
##
## Call:
## glm(formula = Freq ~ Locn * Magnitude, family = poisson, data = Quakes.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```



```
## -1.3261  -0.3225  -0.1172   0.1241   1.6190
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      11.6923     1.1762   9.941  < 2e-16 ***
## LocnWA            7.3923     3.9500   1.871   0.0613 .
## Magnitude        -1.5648     0.2055  -7.616 2.61e-14 ***
## LocnWA:Magnitude -1.5884     0.7199  -2.206   0.0274 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 176.1767  on 17  degrees of freedom
## Residual deviance:   8.2295  on 14  degrees of freedom
## AIC: 65.11
##
## Number of Fisher Scoring iterations: 5
```

```
1 - pchisq(8.23, 14)
```

```
## [1] 0.8770025
```

```
exp(confint(Quake.gfit)[3, ]) # Multiplicative annual change in CA
```

```
## Waiting for profiling to be done...
```

```
##     2.5 %    97.5 %
## 0.1374743 0.3082437
```

```
100 * (1 - exp(confint(Quake.gfit)[3, ])) # Percentage annual decrease in CA
```

```
## Waiting for profiling to be done...
```

```
##    2.5 %   97.5 %
## 86.25257 69.17563
```

Additional Output with WA as the baseline Level

```
Quakes.df$Locn = relevel(Quakes.df$Locn, ref = "WA")
Quake2.gfit = glm(Freq ~ Locn * Magnitude, family = poisson, data = Quakes.df)
exp(confint(Quake2.gfit)[3, ]) # Multiplicative annual change in WA
```

```
## Waiting for profiling to be done...
```

```
##       2.5 %      97.5 %
## 0.009077661 0.140175445
```

```r
100 * (1 - exp(confint(Quake2.gfit)[3, ])) # Percentage annual decrease in WA
```

```
## Waiting for profiling to be done...
```

```
##    2.5 %   97.5 %
## 99.09223 85.98246
```

Methods and Assumption Checks

Since the response variable, earthquake frequency, is a count, we have fitted a generalised linear model with a Poisson response distribution. We have two explanatory variables: Magnitude (Numeric) and Location (Categorical). The scatterplot of magnitude vs frequency shows an exponentially decreasing trend for both locations. A Poisson model with interaction between magnitude and location was fitted. It shows that the interaction between magnitude and location is significant ($P\text{-}value \approx 0.03$).

All model assumptions were satisfied. We can trust the results from this model.

Our final model is

$$\log(\mu_i) = \beta_0 + \beta_1 \times Locn.WA_i + \beta_2 \times Magnitude_i + \beta_3 \times Locn.WA_i \times Magnitude_i,$$

where μ_i is the mean number of earthquakes with magnitude i, and $Locn.Wa_i$ is 1 if the earthquake was in Washington and 0 otherwise. The number of earthquakes (with magnitude $Magnitude_i$ at location $Locn.Wa_i$) has a Poisson distribution with mean μ_i.

Executive Summary

The research question is to quantify the rate of decrease in earthquake frequency (with increasing magnitude) in both California and Washington states, and to assess whether these rates are the same.

The rate of decline in the frequency of earthquakes (with increasing magnitude) is more rapid in Washington than California.

In Washington, there is a 86.0 to 99.0% drop in the expected number of earthquakes for a one unit increase in their magnitude on the Richter scale.

In California, the decrease is between 69.2 to 86.3%.

Case Study 14.2: Snapper counts in and around marine reserves

Tou Ohone Andate - staff number 1234567

Problem

Baited underwater video (BUV) is an established tool for counting fish such as snapper.

BUV was used at two locations, Leigh and Hahei. Each location has a marine reserve. The BUV was deployed at sites inside the marine reserve, and at sites just outside the reserve. BUV was used at a total of 18 sites.

The variables of interest were:

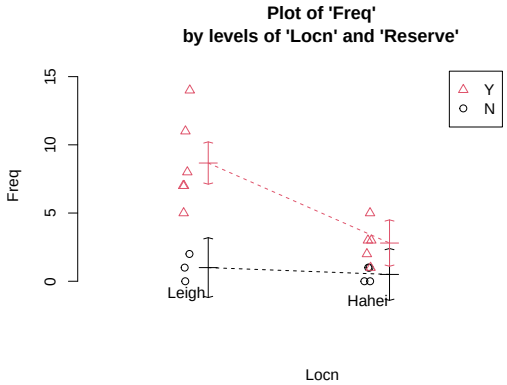
- `Locn`: A two-level factor which describes the BUV's location.
  - It has the levels "Leigh" and "Hahei".
- `Reserve`: A two-level factor which describes whether the BUV's is in a marine reserve.
  - It has the levels "N" and "Y".
- `Freq`: The number of snapped counted by the BUV.

Question of Interest

It was of interest to explore the relative count of snapper with regard to location and reservation status.

Read in and Inspect the Data

```
Snap.df = read.table("SnapperCROPvsHAHEI.txt", header = TRUE)
interactionPlots(Freq ~ Locn * Reserve, data = Snap.df, col.width = 0)
```

In Leigh, it seems that there is a higher frequency of snapper counted in a marine reserve compared to a non-reserve area. In Hahei, it seems that there is little difference in the frequency of snapper counted in a marine reserve compared to a non-reserve area. However, the interaction plot does indicate that the multiplicative effect of reserve could be similar at the two locations (i.e., no interaction).

Model Building and Check Assumptions

```
Snap.glm = glm(Freq ~ Locn * Reserve, family = poisson, data = Snap.df)
summary(Snap.glm)
```

```
##
## Call:
## glm(formula = Freq ~ Locn * Reserve, family = poisson, data = Snap.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4142  -0.8965  -0.1147   0.6215   1.6617
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -0.6931     0.7071  -0.980   0.3270
## LocnLeigh            0.6931     0.9129   0.759   0.4477
## ReserveY             1.7228     0.7559   2.279   0.0227 *
## LocnLeigh:ReserveY   0.4367     0.9612   0.454   0.6496
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 70.453  on 17  degrees of freedom
```

```
## Residual deviance: 14.678  on 14  degrees of freedom
## AIC: 69.143
##
## Number of Fisher Scoring iterations: 5
```

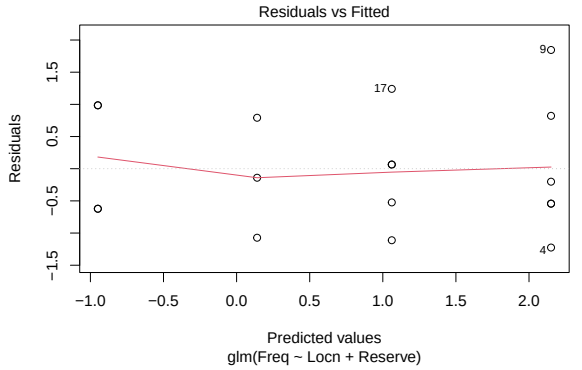
```
1 - pchisq(14.678, 14)
```

```
## [1] 0.4005141
```

```
Snap2.glm = glm(Freq ~ Locn + Reserve, family = poisson, data = Snap.df)
summary(Snap2.glm)
```

```
##
## Call:
## glm(formula = Freq ~ Locn + Reserve, family = poisson, data = Snap.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5170  -0.8002  -0.1739   0.7694   1.6897
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.9491     0.4884  -1.943 0.051990 .
## LocnLeigh     1.0894     0.2845   3.829 0.000128 ***
## ReserveY      2.0105     0.4646   4.328 1.51e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 70.453  on 17  degrees of freedom
## Residual deviance: 14.879  on 15  degrees of freedom
## AIC: 67.344
##
## Number of Fisher Scoring iterations: 5
```

```
plot(Snap2.glm, which = 1)
```


```
exp(confint(Snap2.glm))
```

```
## Waiting for profiling to be done...
```

```
##                 2.5 %     97.5 %
## (Intercept) 0.1298697  0.9105143
## LocnLeigh   1.7443515  5.3626745
## ReserveY    3.3224830 21.3481546
```

Methods and Assumption Checks

As the response variable, `Freq`, is a count, we have fitted a generalised linear model with a Poisson response distribution. We have two expanatory factors: `Locn` and `Reserve`. We initially fitted an interaction between reserve and location. The interaction term was not significant (P-value = 0.65) so the model was refitted with main effects only. Both factors were significant so were retained for the final model.

All assumptions were satisfied. There was no evidence of overdispersion, so we can trust the results from the Poisson model (P-value = 0.40).

For our final model, we assume that the snapper count for observation i is Poisson with mean μ_i, where

$$\log(\mu_i) = \beta_0 + \beta_1 \times \text{Locn.Leigh}_i + \beta_2 \times \text{Reserve.Yes}_i \ ,$$

and where Locn.Leigh_i and Reserve.Yes_i are dummy variables which take the value 1 if observation i is respectively from Leigh and from a marine reserve, otherwise they are 0.

Executive Summary

It was of interest to explore the relative count of snapper with regard to location and reserve status.

We conclude that the expected count of snapper inside a marine reserve is between 3 and 21 times that outside of the reserve.

Additionally, the expected count of snapper in Leigh is between 1.7 and 5.4 times the expected count in Hahei.

Chapter 15: Modelling proportion data using the binomial distribution

STATS 201/8

University of Auckland

Learning outcomes

In this chapter you will learn about:

- Binary (Bernoulli) data, odds and log-odds
- Modelling log-odds
- Modelling the response when it is binary (ungrouped data) via `glm`
- Modelling the response when it is binomial (grouped binary data) via `glm`
- Example 1: Space shuttle *Challenger* accident
- Example 2: Probability of retaining fish in a trawl
- Relevant `R`-code.

Section 15.1 Binary (Bernoulli) data, odds and log-odds

Binary (Bernoulli) data

Here we are considering the situation where the response can only take **two** possible values. These might be coded in the form of:

- Zeros or ones.
- `TRUE` or `FALSE`.
- `Yes` or `No`.
- `Success` or `Failure`..
- Or any other pair of categorical values.

This is called a binary response and can be modelled using the **Bernoulli** distribution.

A Bernoulli distribution is just the special case of the binomial distribution[1]

[1] Recall, the binomial models the number of "successes" out of a fixed number of Bernoulli trials.

Binary (Bernoulli) data...

Examples

A Bernoulli random variable is the outcome for a single trial expressed as a 0 or 1, E.g.,

- Whether or not I get a heads when I flip a coin.
- Whether or not I roll a six with a six-sided dice.
- Whether or not a soccer player scores a penalty kick.
- Whether or not I score a shot in basketball.
- Whether or not a green light is observed at a single set of traffic lights on my regular commute to work.
- Whether or not a patient survives an experimental procedure.

In each of these examples we get to choose which outcomes are assigned the values 0 or 1.

Typically, 0 = "No" (or "Failure"), and 1 = "Yes" (or "Success").

Bernoulli random variables

If Y is a Bernoulli random variable with parameter p, then Y will take the value 1 with probability p, and the value 0 with probability $1 - p$.

Since it is a probability, p must be a value that is between 0 and 1, i.e. $p \in [0, 1]$.

For Bernoulli trials, the usual terminology is to refer to Y as the number of successes, either **zero** or **one**, from a single trial.

It is easy to show[2] that the mean of a Bernoulli random variable is

$$\mathrm{E}[Y] = p$$

and that the variance is

$$\mathrm{Var}(Y) = p(1 - p)$$

[2] See STATS 210.

Odds

We will soon be modelling Bernoulli data using the `glm` function. To be able to make sense of the fitted model we first need to know about **odds**. The odds of an event occurring is given by

$$\text{Odds} = \frac{\text{probability event occurs}}{\text{probability event does not occur}} = \frac{\Pr(Y = 1)}{\Pr(Y = 0)} = \frac{p}{1 - p}$$

Note that Odds must be a value between 0 to infinity, i.e. $\text{Odds} \in [0, \infty)$.

A little of bit of calculus gives us

$$p = \frac{\text{Odds}}{\text{Odds} + 1}$$

So, if we know the odds of an event then we also know the probability of that event (and vice-versa).

What's the odds?

The definition of odds is given in the previous slide.

Luckily for us, the above definition of odds has exactly the same meaning that we give to the word "odds" when we use it in everyday speech when we use "odds" to describe how likely an event is to occur.

By way of example, let event A={I get an A- or better in STATS 201}.

- If the odds of A are 1 (i.e., 1-1, or even odds) then we are saying that A is as likely to occur as not. That is Pr(A)=0.5, and Pr(not A)=0.5.
- If the odds of A are 2 (i.e., 2-to-1) then we are saying that A is twice as likely to occur as not. That is Pr(A)=2/3 and Pr(not A)=1/3.
- If the odds of A are 0.25 (i.e., 0.25-to-1) then we are saying that A is only 1/4 as likely to occur as not. That is Pr(A)=1/5 and Pr(not A)=4/5.

Log-odds

The logarithm of the odds (the log-odds for short) is

$$\text{Log-Odds} = \log(\text{Odds}) = \log\left(\frac{p}{1-p}\right) \equiv \text{logit}(p)$$

Note that the log-odds (as a function of p) is called the *logit* function. Also, since $\text{Odds} \in [0, \infty)$ it follows that Log-Odds can take any value on the real line, i.e. $\text{Log-Odds} \in (-\infty, +\infty)$.

If we know the log-odds, then we can calculate p using the following:

$$p = \frac{\exp(\text{Log-Odds})}{1 + \exp(\text{Log-Odds})}$$

This is called the *logistic* function, and is well-known in mathematics as a function which maps the interval $(-\infty, +\infty)$ to $[0, 1]$. In `R` it is the `plogis`[3] function.

[3] `plogis` is so named because it calculates the probability distribution function of the logistic distribution. This is precisely the logistic function.

Section 15.2
Modelling log-odds

Why log-odds?

Our aim in this Chapter is to be able to fit models for binary data that allow the probability of success, p, to depend on explanatory variables. This is equivalent to allowing the odds or log-odds to depend on the explanatory variables.

For example, if x is a numeric explanatory variable then we have the following modelling options:

- $p = \beta_0 + \beta_1 x$
- $\text{Odds} = \beta_0 + \beta_1 x$
- $\text{Log-Odds} = \beta_0 + \beta_1 x$

Which would your suggest, and why?

Why log-odds?...

Since our model places no restrictions[4] on the values of β_0 or β_1, it is the case that $\beta_0 + \beta_1 x$ can take any value on the real line.

That is, $\beta_0 + \beta_1 x \in (-\infty, \infty)$.

- $p = \beta_0 + \beta_1 x$ ✗
  p must be between 0 and 1.
- $\text{Odds} = \beta_0 + \beta_1 x$ ✗
  Odds must be between 0 and infinity.
- $\text{Log-Odds} = \beta_0 + \beta_1 x$ ✓
  Log-Odds can be any real number.

[4] i.e., $\beta_0 \in (-\infty, \infty), \beta_1 \in (-\infty, \infty)$

Modelling log-odds

We shall apply our linear model to the log-odds, and so our general form of the model is:

$$\begin{aligned}\log(\text{Odds}) &= \text{logit}(p)\\ &= \text{logit}(\mu)\\ &= \text{logit}(E[Y|x_1 \ldots])\\ &= \beta_0 + \beta_1 x_1 + \ldots\end{aligned}$$

Remark: This is analogous to what we did when the response variable was Poisson count data. There we were using the model

$$\log(\mu) = \log(E[Y|x_1 \ldots]) = \beta_0 + \beta_1 x_1 + \ldots$$

and note that $\log(\mu) \in (-\infty, \infty)$.

Modelling log-odds. . .

With a single numeric explanatory variable

When we have just a single explanatory variable x that is numeric (i.e., not a factor) then the linear model for log-odds is:

$$\log(\text{Odds}) = \beta_0 + \beta_1 x$$

In other words,

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x$$

where p is the probability of "success" for a subject with explanatory variable x.
This can be re-arranged in the logistic form

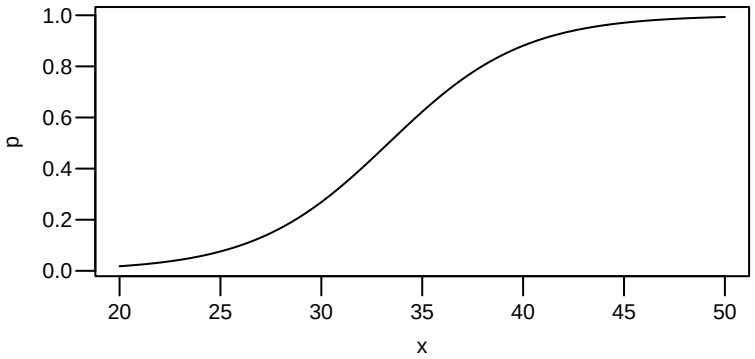
$$p = \frac{\exp(\beta_0 + \beta_1 x)}{1 + exp(\beta_0 + \beta_1 x)}$$

This equation forms an "S" shaped curve as a function of x.

Modelling log-odds. . .

With a single continuous explanatory variable. . .

For example, if $\beta_0 = -10$ and $\beta_1 = 0.3$ then the curve looks like:



- The greater the magnitude of β_1 the steeper the curve.
- If $\beta_1 < 0$ the curve is a reverse "S" shape.
- Changing β_0 changes the horizontal position of the curve.

Logistic regression

- The logistic function gives its name to this approach for modelling binary data – it is commonly called *logistic* regression.
- At the moment we are going to use logistic regression to model binary (*Bernoulli*) data, but it can also be used to model proportion data in the form of the number of "successes" divided by the number of trials. The number of "successes" is assumed to be binomially distributed, so logistic regression is often called a binomial GLM.
- In GLMs, the function that links μ to the linear predictor is called the *link* function. Here, $\mu = p$, and so the link function is the logit.

Section 15.3
Modelling the response when it is binary (ungrouped data) via glm

Example – Basketball

A few years ago the lecturer of an experimental design course conducted a basketball shooting experiment in class. In this experiment

- there were ten females and ten males;
- each shooter shot at the basket from 1m, 2m and 3m;
- each person took one shot from each distance;
- the order of the shooting distance was randomly chosen, and
- a 1 was recorded if the shot was successful, and a 0 was recorded if the shot was missed.

What do **you** think is the most important factor affecting the probability of making the shot in this experiment?

Inspect the data

Ungrouped format

The data are read in to dataframe bb.df. It contains 60 observations, since each of the 20 students takes 3 shots.

```
> bb.df = read.csv("Data/basketball.csv")
> head(bb.df, 10)
   distance gender basket
1         3      M      1
2         1      F      1
3         2      M      1
4         3      M      0
5         1      M      1
6         2      F      1
7         2      F      1
8         1      F      1
9         3      F      0
10        1      F      1
```

These data are in an *ungrouped* format as the response variable for each row is either a success or a failure (a 1 or a 0).

Basketball

Grouped format

Because the ungrouped responses are zeros and ones, it is hard to detect patterns in this kind of data. However we can group the data over each combination of gender and distance.

The xtabs function produces a cross-tabulation table[5]

```
> success.tbl = xtabs(basket~distance+gender, data = bb.df)
> success.tbl
        gender
distance  F  M
       1 10 10
       2  6  5
       3  2  1
```

Our conclusions should be kind of obvious from the table in this example, even without the model fitting.

[5] Also called a frequency table.

Basketball...

Logistic regression model formula

We want to fit a model that estimates how the probability of scoring is related to distance (numeric explanatory) and gender (factor explanatory). The interaction model (Chapter 8) is

$$\begin{aligned}\text{logit}(p_i) &= \log\left(\frac{p_i}{1-p_i}\right)\\ &= \beta_0 + \beta_1 distance_i + \beta_2 gender_i + \beta_3(gender_i \times distance_i)\end{aligned}$$

or in terms of p_i[6]

$$p_i = \frac{exp(\beta_0 + \beta_1 distance_i + \beta_2 gender_i + \beta_3(gender_i \times distance_i))}{1 + exp(\beta_0 + \beta_1 distance_i + \beta_2 gender_i + \beta_3(gender_i \times distance_i))}$$

with $Y_i \sim \text{Bernoulli}(p_i)$ and $gender_i$ is an indicator variable which is 0 if the shooter is female, and 1 if the shooter is male.

[6] We do not recommend that you try to write it this way in a test or assignment!

Basketball...

Fit the model...

Fitting the logistic regression model is easy.

We simply tell the `glm` function that the responses are Bernoulli random variables by setting `family = binomial`.[7]

```
> bb.fit = glm(basket ~ distance * gender, family = binomial,
+              data = bb.df)
```

Note that, as with Poisson GLMs, we **do not** transform the responses.

Aside: By default the `glm()` function uses the logit link function (i.e., does logistic regression) when we set `family = binomial`. Other choices are possible – see STATS 730.

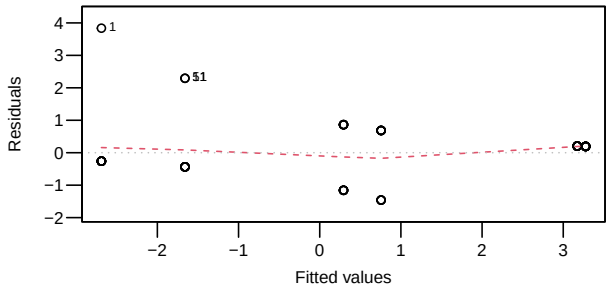
[7] Recall, the Bernoulli distribution is a special case of the binomial distribution.

Basketball...

Check the model...

Model checking is difficult when the data are *ungrouped*.[8]

```
> plot(bb.fit, which = 1, lty=2)
```



The plot of the residuals versus the fitted values is not particularly informative. Even if the model is appropriate, it can looked quite patterned. Influence checks are also of little use.

[8] This is because the data are *sparse*, in the sense that they are either 0's or 1's.

Basketball...

Inspect the fitted model

```
> summary(bb.fit)
```

```
Call:
glm(formula = basket ~ distance * gender, family = binomial,

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)       5.5878     1.9050   2.933  0.00335 **
distance         -2.4159     0.8181  -2.953  0.00314 **
genderM           0.6710     2.9235   0.230  0.81847
distance:genderM -0.5668     1.3213  -0.429  0.66794
---
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 82.108  on 59  degrees of freedom
Residual deviance: 46.202  on 56  degrees of freedom
```

When the data are *ungrouped* we also **cannot** use the residual deviance in the same way we did for Poisson GLMs.

We just have to presume that the fitted model satisfies assumptions.

Basketball...

Simplify the model

There is no evidence of an interaction between gender and distance. We'll apply Occam's razor and drop it from the model.

```
> bb.fit1 = glm(basket ~ distance + gender, family = binomial, data = bb.df)
> summary(bb.fit1)
```

```
Call:
glm(formula = basket ~ distance + gender, family = binomial,

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   6.1469     1.5242   4.033 5.51e-05 ***
distance     -2.6648     0.6364  -4.188 2.82e-05 ***
genderM      -0.5478     0.7486  -0.732    0.464
---
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 82.108  on 59  degrees of freedom
Residual deviance: 46.392  on 57  degrees of freedom
```

Looks like we can drop gender, too.

Basketball...

Simplify the model...

```
> bb.fit2 = glm(basket ~ distance, family = binomial, data = bb.df)
> summary(bb.fit2)
```

```
Call:
glm(formula = basket ~ distance, family = binomial, data = bb.df)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   5.7980     1.4038   4.130 3.63e-05 ***
distance     -2.6310     0.6274  -4.193 2.75e-05 ***
---
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 82.108  on 59  degrees of freedom
Residual deviance: 46.937  on 58  degrees of freedom
```


Basketball...

Interpretting the fitted model

So, our final model is of the form

$$\text{Log-Odds} = \beta_0 + \beta_1 \times \textit{distance}$$

This means that a 1 metre increase in distance increases the log-odds by β_1.

In terms of odds, we have

$$\begin{aligned}\text{Odds} &= \exp(\beta_0 + \beta_1 \times \textit{distance}) \\ &= \exp(\beta_0) \times \exp(\beta_1)^{\textit{distance}}\end{aligned}$$

This means that a 1 metre increase in distance multiplies the odds by $\exp(\beta_1)$.

Basketball...

Interpretting the fitted model...

```
> coef(bb.fit2)
(Intercept)    distance
   5.797968   -2.631033
```

The estimated coefficient on distance, $\hat{\beta}_1$, is -2.63. This says that the *log-odds of success* decreases by 2.63 for every 1 metre increase of the shooter from the goal.

```
> exp(coef(bb.fit2))
 (Intercept)     distance
329.62899018   0.07200401
> 100*(1-exp(coef(bb.fit2)))
(Intercept)    distance
-32862.8990     92.7996
```

By exponentiating $\hat{\beta}_1$ we can now say that a 1 metre increase in the distance results multiplies the odds of shooting a basket by 0.072. However, it would be more natural to say that it results in a $100 \times (1 - 0.072)\% = 92.8\%$ reduction in the odds.

Basketball...

Interpretting the fitted model...

For our Executive Summaries we need confidence intervals rather than point estimates:

```
> (bb.ci2 = confint(bb.fit2))
Waiting for profiling to be done...
                2.5 %    97.5 %
(Intercept)  3.422396  9.037020
distance    -4.103523 -1.568945
> 100*(1-exp(bb.ci2))
                 2.5 %        97.5 %
(Intercept) -2964.27509 -840767.86136
distance       98.34856      79.17351
```

A 1 metre increase in the distance results in a reduction in the odds of scoring of between 79.2% and 98.3%

Estimating p

Basketball example

How do we estimate the probability of a successful shot from distances of 1, 2 and 3 m?

By default, `predict` will give estimates on the log-odds scale, that is, it will calculate the linear predictor $\hat{\beta}_0 + \hat{\beta}_1 \times distance$.

```
> predn.df=data.frame(distance = 1:3)
> bb.logit.pred = predict(bb.fit2, newdata = predn.df)
> bb.logit.pred
        1         2         3
3.1669343 0.5359009 -2.0951325
```

We can easily convert this to the response (i.e., probability) scale

```
> plogis(bb.logit.pred)
        1         2         3
0.9595708 0.6308584 0.1095708
```

Even easier is to use the `type="response"` argument of `predict`:

```
> predict(bb.fit2, newdata = predn.df, type="response")
        1         2         3
0.9595708 0.6308584 0.1095708
```


Confidence Intervals for p

Basketball example...

Confidence intervals need to first be calculated on the log-odds scale, and then transformed back to the probability scale.

```
> bb.logit.predses = predict(bb.fit2, newdata = predn.df, se.fit = TRUE)$se.fit
> bb.logit.predses
        1         2         3
0.8151018 0.3812977 0.6432312
> # Lower and upper bounds of CIs for the log-odds
> lower = bb.logit.pred - 1.96*bb.logit.predses
> upper = bb.logit.pred + 1.96*bb.logit.predses
> ci = cbind(lower, upper)
```

and convert these to probabilities:

```
> plogis(ci)
       lower     upper
1 0.82768876 0.9915452
2 0.44733541 0.7830016
3 0.03370361 0.3027157
```

We see that the probability of a goal from 1 metre distance is between 0.828 and 0.991, but drops to between 0.034 and 0.303 at 3 metres.

Confidence Intervals for p the easy way

Basketball example...

Or we can use the `predictGLM` function from the `s20x` package.

To get confidence intervals for log-odds

```
> predictGLM(bb.fit2,newdata = data.frame(distance = 1:3),type="link")
***Estimates and CIs are on the link scale***
         fit        lwr        upr
1  3.1669343  1.5693642  4.7645045
2  0.5359009 -0.2114289  1.2832308
3 -2.0951325 -3.3558424 -0.8344225
```

or to get confidence intervals for the probabilities

```
> predictGLM(bb.fit2,newdata = data.frame(distance = 1:3),type="response")
***Estimates and CIs are on the response scale***
        fit        lwr       upr
1 0.9595708 0.82769294 0.9915450
2 0.6308584 0.44733881 0.7829992
3 0.1095708 0.03370437 0.3027108
```

In the next section, we reproduce the above analysis after first grouping the data.

Section 15.4
Modelling the response when it is binomial (grouped binary data) via glm

The binomial distribution

A binomial random variable is the number of successes that occur over a fixed number (n) of Bernoulli trials, all with the same probability of success (p):

- The number of heads if I flip a coin fifty times.
- The number of sixes I get if I roll four dice.
- The number of penalties a football team scores in a penalty shoot-out.
- The number of successful basketball shots out of ten attempts.
- The number of green lights out of the total number of traffic lights on my regular commute to work.
- The number of patients who survive an experimental procedure, out of the number that underwent that procedure.
- The number of O-rings that fail when a space shuttle is launched, out of the total of six O-rings on the solid fuel rockets.

One difference between a binomial and Poisson random variable is that the binomial has an upper limit set by the number of trials.

Example revisited – Basketball: Grouped data

In many situations it is possible to format the binary data so that they are *grouped*.

With grouped data, all trials with the same values of the explanatory variables are aggregated in the same row, along with the number of "successes" and "failures".

We demonstrate with our basketball data:

```
> #Load dplyr package to manipulate data frames
> library(dplyr)
> bb.grouped.df = bb.df %>% group_by(gender,distance) %>%
+                       summarize(n=n(),propn=sum(basket)/n)
> #Change tibble back to a data frame
> bb.grouped.df = data.frame(bb.grouped.df)
> bb.grouped.df
  gender distance  n propn
1      F        1 10   1.0
2      F        2 10   0.6
3      F        3 10   0.2
4      M        1 10   1.0
5      M        2 10   0.5
6      M        3 10   0.1
```


Basketball: Grouped data

We can still use glm to fit a logistic regression model to grouped data.

Previously, we considered each of the 60 attempts to score as a separate Bernoulli trial.

Instead, we now consider the number of successes out of 10 attempts at each level of gender and distance. This is a binomial random variable.

Both approaches will give you the same estimates, confidence intervals, and conclusions, but there are some advantages to having grouped data. Especially, being better able to check model assumptions.

Basketball: Grouped data...

Fitting the model

We set the proportion of successes as the response, and use the argument `weights` to specify the number of trials associated with each observation.

```
> bb.fit3 = glm(propn ~ distance * gender, weights = n,
+               family = binomial, data = bb.grouped.df)
> summary(bb.fit3)
```

```
Call:
glm(formula = propn ~ distance * gender, family = binomial, data = bb.grouped.df,

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)       5.5878     1.9050   2.933  0.00335 **
distance         -2.4159     0.8181  -2.953  0.00314 **
genderM           0.6710     2.9236   0.230  0.81848
distance:genderM -0.5668     1.3214  -0.429  0.66795
---
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 38.2749  on 5  degrees of freedom
Residual deviance:  2.3688  on 2  degrees of freedom
```


Basketball: Grouped data...

Checking the model...

The grouped data output is equivalent to that from working with the *ungrouped* data, except that the deviance values have changed.

We may now interpret residual plots and check the residual deviance statistic, notwithstanding that we ideally require the expected count of both successes and failures to be reasonably large, say > 5.[9]

```
> 1 - pchisq(2.3688, 2)
[1] 0.3059297
```

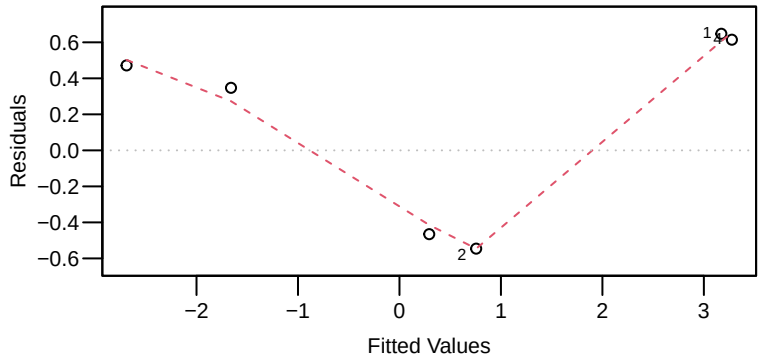
No problems with the residual deviance.

[9] This is not satisfied for the basketball data, but we will proceed anyway.

Basketball: Grouped data...

Checking the model...

```
> plot(bb.fit3, which = 1, lty=2)
```



We only have six observations (one for each combination of distance and gender), so it is difficult to see if there is a pattern in the residuals, but in this case none are large enough to worry us.
The fitted model is not perfect, but should still be useful.

An alternative way to specify the model

Basketball: Grouped data...

Sometimes the grouped data are provided as the number of successes and failures:

```
> bb.grouped.df = transform(bb.grouped.df, success=n*propn, fail=n*(1-propn))
> bb.grouped.df
  gender distance  n propn success fail
1      F        1 10   1.0      10    0
2      F        2 10   0.6       6    4
3      F        3 10   0.2       2    8
4      M        1 10   1.0      10    0
5      M        2 10   0.5       5    5
6      M        3 10   0.1       1    9
```

An alternative way to specify the model...

Then, on the left-hand side of the model formula we provide `glm` with the names of the two columns containing the number of successes and failures, and omit the `weights` argument.

```
> bb.fit4 = glm(cbind(success, fail) ~ distance * gender, family = binomial,
+     data = bb.grouped.df)
> summary(bb.fit4)
```

```
Call:
glm(formula = cbind(success, fail) ~ distance * gender, family = binomial,

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)       5.5878     1.9050   2.933  0.00335 **
distance         -2.4159     0.8181  -2.953  0.00314 **
genderM           0.6710     2.9236   0.230  0.81848
distance:genderM -0.5668     1.3214  -0.429  0.66795
---
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 38.2749  on 5  degrees of freedom
Residual deviance:  2.3688  on 2  degrees of freedom
```


Basketball: Grouped data...

In practice, we would simplify these models by removing the interaction and gender effects in turn, as we did for analysis of the ungrouped data.

We have not done so because the results are unchanged.

Section 15.5
Example 1: Space shuttle Challenger accident

Space shuttle *Challenger* accident

This example tells a sad story showing what can happen if standard linear models are mis-applied to proportion data.

The NASA space shuttle *Challenger* broke up during launch on the cold morning of 28 January 1986. Most of the crew survived the initial break-up, but are believed to have been killed when the crew capsule hit the ocean at high speed.

At the time, it was the most expensive human accident that had ever occurred.[10]

It was particularly traumatic for the American people, because the shuttle was carrying the first civilian astronaut, Christa McAuliffe, who was a high-school social studies teacher. Approximately 17% of Americans were watching the launch live.

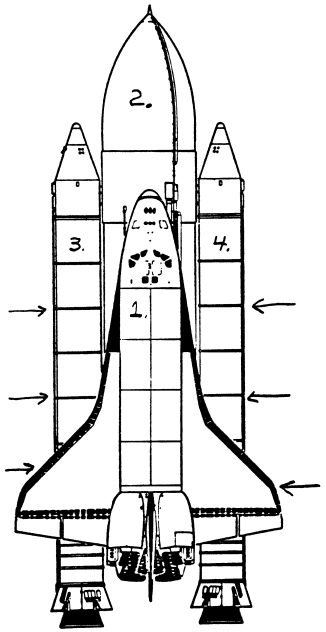
[10] Unfortunately, an even more deadly and expensive accident occurred just months later - Chernobyl.

Space shuttle *Challenger* accident...

Subsequent investigation found that O-ring failures were the cause of the disaster.

The temperature[a] at the launch site was a chilly $31°F$. The risk of O-ring failure in cold weather had been grossly underestimated due to using a linear model on proportion data.

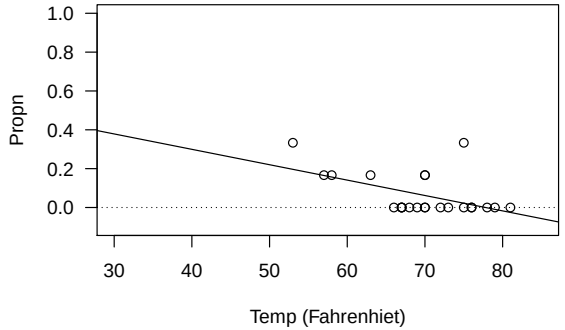
[a] This is a fraction of a a degree Celsius below freezing point.



Space shuttle *Challenger* accident...

The space shuttle solid-fuel rockets have a total of 6 O-rings. It was suspected that O-ring reliability was influenced by temperature.

The rockets retrieved from previous launches were examined for O-ring distress, and the proportion of distressed O-rings was plotted against temperature and a simple linear regression was fitted:



The simple linear regression estimates a negative probability of O-ring distress for temperatures above about $79°F$!?!

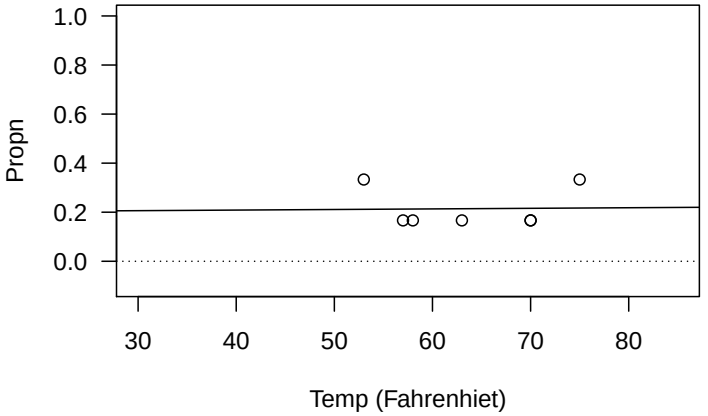
Space shuttle *Challenger* accident...

To fix the problem with the negative estimated probabilities, "statisticians" at NASA decided to remove all the zero data values since they felt that the zero values contained no information about the probability of O-ring distress.

This kind of stupidity occurs even today.

Space shuttle *Challenger* accident...

With the zero values removed there is no evidence of a relationship between temperature and O-ring distress,



and so it was decided to approve the launch on that $31°F$ morning.

Space shuttle *Challenger* accident...

73 seconds after lift-off the shuttle blew apart:

These data should have been analysed using a binomial GLM that is appropriate for proportion data.

Space shuttle *Challenger* accident...

Logistic regression model

```
> Space.df = read.table("Data/ChallengerShuttle.txt", head = TRUE)
> Space.df$Temp
 [1] 66 70 69 68 67 72 73 70 57 63 70 78 67 53 67 75 70 81 76 79 75 76 58
> Space.df$Failure
 [1] 0 1 0 0 0 0 0 0 1 1 1 0 0 2 0 0 0 0 0 0 0 2 0 1
> Space.gfit=glm(cbind(Failure, 6-Failure)~Temp, family = binomial,
+                data = Space.df)
```

```
> summary(Space.gfit)
```

```
Call:
glm(formula = cbind(Failure, 6 - Failure) ~ Temp, family = binomial,

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  5.08498    3.05247   1.666   0.0957 .
Temp        -0.11560    0.04702  -2.458   0.0140 *
---
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 24.230  on 22  degrees of freedom
Residual deviance: 18.086  on 21  degrees of freedom
```


Space shuttle *Challenger* accident...

Conclusions[11]

Our CI for the probability of an O-ring failing at $31°F$ is

```
> predictGLM(Space.gfit, newdata = data.frame(Temp=31), type = "response")
***Estimates and CIs are on the response scale***
        fit       lwr       upr
1 0.8177744 0.1596025 0.9906582
```

The rockets have 6 O-rings, so the expected number of O-rings failures is

```
> 6*predictGLM(Space.gfit, newdata = data.frame(Temp=31), type = "response")
***Estimates and CIs are on the response scale***
       fit      lwr      upr
1 4.906646 0.957615 5.943949
```

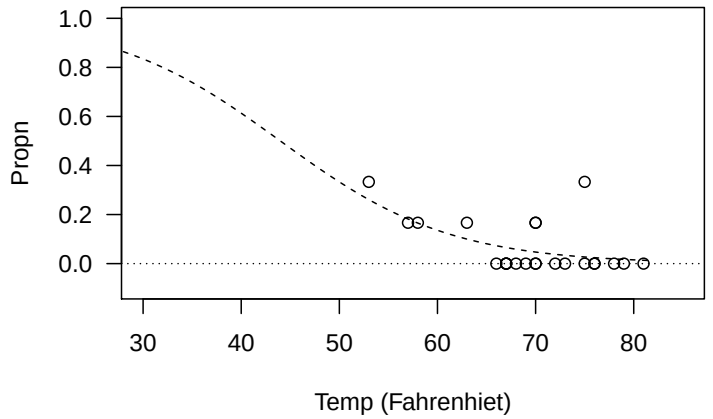
Note that this is quite a wide confidence interval because we are estimating the probability at a temperature that is well beyond those observed in the dataset. This is called "extrapolation", and is always risky.

The real message here is that there **could** be a very high probability of disaster.

[11] Model checks are left as an exercise.

Space shuttle *Challenger* accident...

The fit of this generalized linear model to the O-ring data looks like:



This model predicts that the probability of an O-ring experiencing distress at $31°F$ is 0.818. This corresponds to expecting $6 \times 0.818 = 4.91$ distressed O-rings.

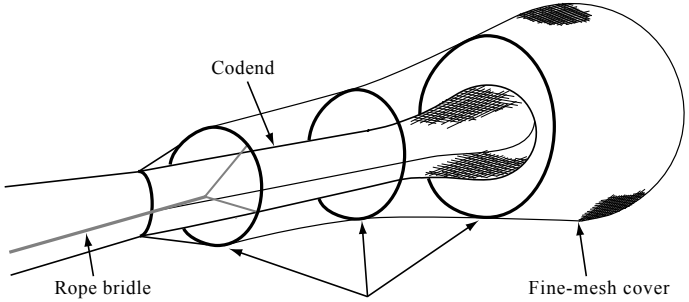
Section 15.6
Example 2: Catching the right size fish

Example – Fishing

In commercial fishing it is important to let small fish escape so they can grow and breed. Experiments are frequently done with trawl gear to determine how the probability of retaining a fish depends on its size.

The experiment consisted of observing the number of fish (at given fork lengths) entering a trawl codend, and the number of those retained by it.

In the dataframe `Haddock.df`, `codend` is the number in the codend and `cover` is the number that escape the codend and are retained in the cover. The total number of haddock is therefore `codend + cover`.



Fishing

```
> Haddock.df = read.table("Data/Haddock.dat", head = TRUE)
> Haddock.df = transform(Haddock.df,propn=codend/(codend+cover))
> head(Haddock.df, 17)
   forklen codend cover      propn
1     19.5      0     2 0.00000000
2     20.5      0     5 0.00000000
3     21.5      0    11 0.00000000
4     22.5      0    28 0.00000000
5     23.5      1    53 0.01851852
6     24.5      5    46 0.09803922
7     25.5      1    35 0.02777778
8     26.5      3    27 0.10000000
9     27.5      1     5 0.16666667
10    28.5      0     3 0.00000000
11    29.5      1     2 0.33333333
12    30.5      0     0        NaN
13    31.5      0     2 0.00000000
14    32.5      3     2 0.60000000
15    33.5      4     4 0.50000000
16    34.5      5    12 0.29411765
17    35.5      8     9 0.47058824
```

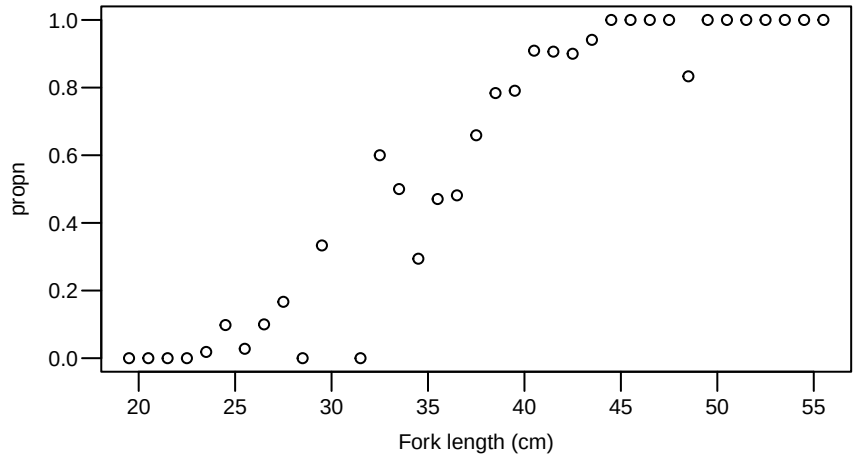

Fishing...

```
> tail(Haddock.df, 20)
   forklen codend cover      propn
18    36.5     13    14 0.4814815
19    37.5     29    15 0.6590909
20    38.5     29     8 0.7837838
21    39.5     34     9 0.7906977
22    40.5     30     3 0.9090909
23    41.5     29     3 0.9062500
24    42.5     18     2 0.9000000
25    43.5     16     1 0.9411765
26    44.5     11     0 1.0000000
27    45.5     12     0 1.0000000
28    46.5      9     0 1.0000000
29    47.5      3     0 1.0000000
30    48.5      5     1 0.8333333
31    49.5      3     0 1.0000000
32    50.5      5     0 1.0000000
33    51.5      2     0 1.0000000
34    52.5      2     0 1.0000000
35    53.5      1     0 1.0000000
36    54.5      1     0 1.0000000
37    55.5      4     0 1.0000000
```

Fishing...

Plot the data

```
> plot(propn ~ forklen, data = Haddock.df, xlab = "Fork length (cm)")
```



Note that the proportions retained seem to follow an "S" shape.

Fishing...

Fitting the model

Let's fit a logistic regression model.

Note that the data are grouped, and the number of "successes" and "failures" are given by the variables `codend` and `cover`, respectively[12].

```
> Haddock.glm = glm(cbind(codend,cover) ~ forklen, family = binomial,
+                   data = Haddock.df)
```

[12] Retaining the fish may by a success for the fisherman, but perhaps not for the fish!

Fishing...

Modelling proportion data using log-odds

```
> summary(Haddock.glm)
```

```
Call:
glm(formula = cbind(codend, cover) ~ forklen, family = binomial,

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -10.63219    0.86468  -12.30   <2e-16 ***
forklen       0.30396    0.02363   12.86   <2e-16 ***
---
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 432.464  on 35  degrees of freedom
Residual deviance:  23.436  on 34  degrees of freedom
```


Fishing...

Checking the fitted model

As with the Poisson case, we need to check the residual deviance by comparing it to a χ^2 distribution. We can do this here as the data are grouped: we have many observations for each level of fork-length, so the data are not 'sparse'.

The residual deviance is 23.44 on 34 degrees of freedom. The P-value is

```
> 1-pchisq(23.44,34)
[1] 0.9132025
```

which indicates no significant problems with the fitted model.

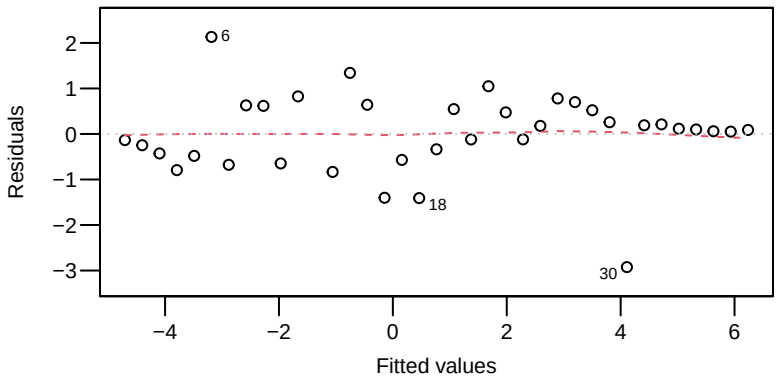
Recall: If the residual deviance indicated lack of fit then we would have to refit using `family = quasibinomial`.

Fishing...

Checking the fitted model...

Let's check the residual plot

```
> plot(Haddock.glm,which=1, lty=2)
```



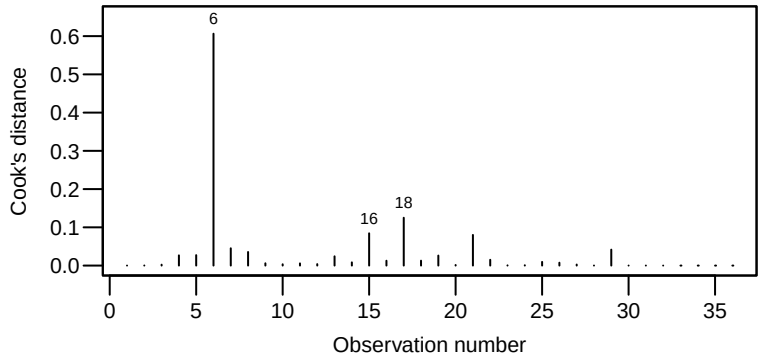
Looks good, other than one largish residual corresponding to the 48.5 cm fish that was in the cover.

Fishing...

Checking the fitted model...

Lets check the influence plot

```
> plot(Haddock.glm,which=4)
```



Hmmm, the fit is somewhat influenced by the retention of small fish of length 24.5 cm. However, this is known to happen with trawl gear, so this observation will not be removed.

Fishing...

Interpretation

```
>   summary(Haddock.glm)$coef
              Estimate Std. Error   z value     Pr(>|z|)
(Intercept) -10.6321889 0.86467598 -12.29615 9.499049e-35
forklen       0.3039569 0.02363082  12.86273 7.295469e-38
```

Remember, the linear model is on the log-odds scale. So, the value 0.304 corresponds to an estimated increase in the log-odds of codend retention for every one cm increase in `forklen`.

The 95% confidence intervals for β_0 and β_1 are:

```
>   confint(Haddock.glm)
Waiting for profiling to be done...
                 2.5 %     97.5 %
(Intercept) -12.4563256 -9.0501022
forklen       0.2604867  0.3535532
```


Fishing...

Interpretation...

Exponentiating the above confidence intervals gives:

```
> exp(confint(Haddock.glm))
Waiting for profiling to be done...
                    2.5 %      97.5 %
(Intercept) 3.893019e-06 0.000117379
forklen     1.297561e+00 1.424118770
```

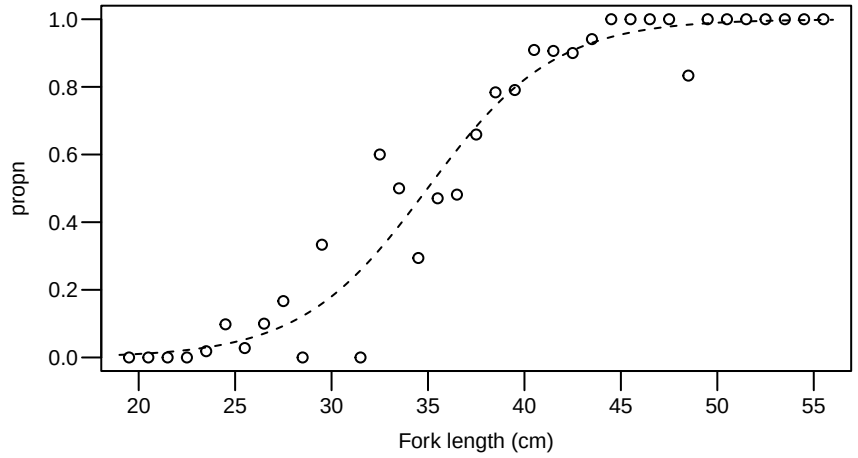
In our **Executive Summary** we could say something like

"Every 1 cm increase in the fork length of a haddock multiplies the odds of it being retained in the codend by between 1.3 and 1.42".

Or, "Every 1 cm increase in the fork length of a haddock corresponds to an increase in odds of it being retained in the codend of between 30% and 42%".

Fishing...

Here is a plot of the data with the fitted "S" curve overlaid:



Fishing...

Confidence intervals for p

To estimate the probability of retention for haddock of lengths 25, 35 and 45 cm:

```
> predn=predictGLM(Haddock.glm, data.frame(forklen=c(25,35,45)), type="response")
***Estimates and CIs are on the response scale***
> predn
         fit        lwr        upr
1 0.04594542 0.02639853 0.07879451
2 0.50157550 0.43868954 0.56441166
3 0.95460393 0.92887749 0.97131200
```

The probability that a 25 cm haddock is retained in the codend is between 0.026 and 0.079. This increases to between 0.439 and 0.564 for 35 cm haddock, and between 0.929 and 0.971 for 45 cm haddock.

Section 15.7
Relevant R-code.

Most of the R-code you need for this chapter

Ungrouped data

Ungrouped data are binary i.e., a 0 or 1 response value for each observation (for example `bb.df`). Fit the model with `glm` using `family=binomial`:

```
> bb.fit2 = glm(basket ~ distance, family = binomial, data = bb.df)
```

These type of data are known as sparse (because either the count of successes is 0 or the count of failures is 0) and we cannot test our model assumptions effectively as we would like to. If possible we should group our data.

Most of the R-code you need for this chapter...

Grouped data

We were also able to analyse the basketball data as grouped observations which allows us to check assumptions including if we have to make a `family= quasibinomial` adjustment to our model.

Syntax 1 - specify proportion of successes and number of trials

```
> bb.fit3 = glm(propn ~ distance * gender,weight=n,
+               family = binomial,data = bb.grouped.df)
```

Syntax 2 - specify frequency of successes and failures

```
> bb.grouped.df = transform(bb.grouped.df, success=n*propn, fail=n*(1-propn))
> bb.fit4 = glm(cbind(success, fail) ~ distance * gender,
+               family = binomial,data = bb.grouped.df)
```


Most of the R-code you need for this chapter...

Both model syntaxs give exactly the same results and we can perform the residual deviance check as follows:

```
> 1-pchisq(bb.fit4$deviance,bb.fit4$df.residual)
> # or directly
> 1 - pchisq(2.3688, 2)
```

If the p-value is below 0.05 then we have evidence against the assumed binomial distribution and need to refit our model using `family= quasibinomial`.

Most of the R-code you need for this chapter...

Confidence intervals can be calculated for the parameters and back-transformed and interpreted as a multiplicative effect on the odds (or as a % change if you prefer).

```
> (bb.ci2 = confint(bb.fit2))
> exp(bb.ci2)
```

Confidence intervals for probabilities take a bit more effort since they require calcuating a confidence interval on the logit scale (i.e., for log-odds) and then backtransforming using the logistic function.

Fortunately, we can use `predictGLM`.

```
> bb.pred.intervals = predictGLM(bb.fit2,
+                            newdata = data.frame(distance = c(1, 2, 3)),
+                            type="response")
***Estimates and CIs are on the response scale***
> bb.pred.intervals
        fit        lwr       upr
1 0.9595708 0.82769294 0.9915450
2 0.6308584 0.44733881 0.7829992
3 0.1095708 0.03370437 0.3027108
```

Case Study 15.1: Haddock retention in a trawl

Tou Ohone Andate - staff number 1234567

Problem

The experiment consisted of observing the number of fish at given fork lengths that entered a trawl codend, and the number of those fish that were retained by it.

In dataframe `Haddock.df`, `codend` is the number retained in the codend and `cover` is the number that escaped the codend into the cover region. The total number of fish is therefore `codend + cover`.

The variables of interest were:

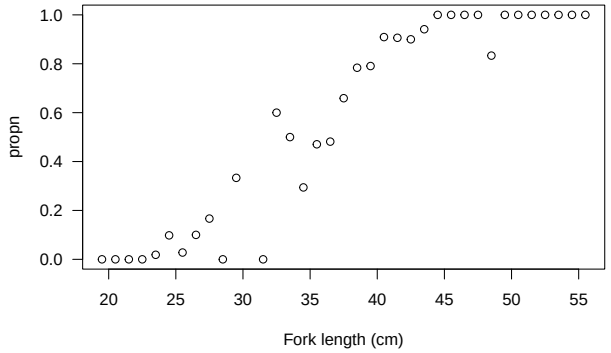
- `propn`: The proportion of fish caught in the trawl codend.
- `forklen`: The fork length of the fish (in cm). (This is the length measured from snout to the end of tail fin rays.)

Question of Interest

How does the length of the fish affect the odds of it being caught in the trawl codend?

Read in and Inspect the Data

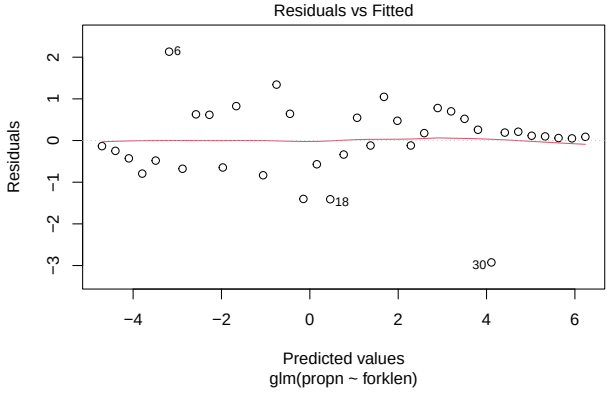
```
Haddock.df = read.table("Haddock.txt", head = T)
Haddock.df$n = with(Haddock.df, codend + cover)
Haddock.df$propn = with(Haddock.df, codend/n)
plot(propn ~ forklen, data = Haddock.df, xlab = "Fork length (cm)", las = 1)
```

We can see that as the length of the fish increases, the proportion of fish caught in the trawl codend increases. Notably, it follows a S-shaped curve.

Model Building and Check Assumptions

```
Haddock.glm = glm(propn ~ forklen, family = binomial, weight = n, data = Haddock.df)
plot(Haddock.glm, which = 1)
```



```
summary(Haddock.glm)
```

```
##
## Call:
## glm(formula = propn ~ forklen, family = binomial, data = Haddock.df,
##     weights = n)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7344  -0.5293   0.1322   0.5701   1.8084
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -10.63219    0.86468  -12.30   <2e-16 ***
## forklen       0.30396    0.02363   12.86   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 432.464  on 35  degrees of freedom
## Residual deviance:  23.436  on 34  degrees of freedom
##   (1 observation deleted due to missingness)
## AIC: 79.75
##
## Number of Fisher Scoring iterations: 5
```

```
1 - pchisq(23.436, 34)
```

```
## [1] 0.9133009
```

```
exp(confint(Haddock.glm))
```

```
## Waiting for profiling to be done...
```

```
##                    2.5 %      97.5 %
## (Intercept) 3.893019e-06 0.000117379
## forklen     1.297561e+00 1.424118770
```

```
100*(exp(confint(Haddock.glm))-1)
```

```
## Waiting for profiling to be done...
```

```
##                 2.5 %    97.5 %
## (Intercept) -99.99961 -99.98826
## forklen      29.75615  42.41188
```


Methods and assumption Checks

The data recorded the number of fish that entered the codend and the number of those fish that were retained, for fish of different fork lengths. We therefore fitted a Binomial GLM with a single predictor of fork length (numeric). The response was treated as grouped data, with each group corresponding to a specific value of fork length.

Taking into account the fact that we expect small positive residuals from length classes with high retention probability, the residual plot from fitting the binomial model showed no strong trends. There was no evidence of overdispersion ($P\text{-}value = 0.91$) so we can trust the results from this binomial model.

Our final model was

$$\log(\text{Odds}_i) = \beta_0 + \beta_1 \times \text{length}_i,$$

where Odds_i is the odds of retention for fish at the ith value of fork length.

Executive Summary

We aimed to investigate the relationship between fish fork length and the odds that the fish is retained in the trawl codend after entering it.

We estimate that for every 1 cm increase in the fork length of a haddock, the odds that it is retained in the codend increase by between 30% and 42%.

Case Study 15.2: Whio chick survival

Tou Ohone Andate - staff number 1234567

Problem

The whio, or blue duck, is an endangered bird that lives in fast-flowing rivers throughout NZ. Although it is a type of duck, it doesn't quack but instead emits a soft whistle that sounds like 'whio' (pronounced 'fee-oh'). It's an iconic NZ species and is pictured on the $10 bank note.

Rangers from the Department of Conservation monitored whio nests in various regions of NZ over several years. For each nest found, they recorded the number of chicks in the nest, and how many of them survived to the point of fledging (leaving the nest). They also recorded whether the region had suffered floods in the corresponding year, because flooded rivers are a potential cause of chick mortality.

The dataframe `Whio.df` has records for several regions and years, with the following columns:

- `Chicks`: total number of chicks in monitored nests in that region and year.
- `Survived`: number of the chicks that survived.
- `Died`: number of the chicks that died.
- `Region`: a categorical variable for location, with levels Fiordland, Oparara, and TeUrewera.
- `Flood`: a categorical variable specifying whether the region was affected by floods in the corresponding year, with levels Yes and No.

Questions of interest

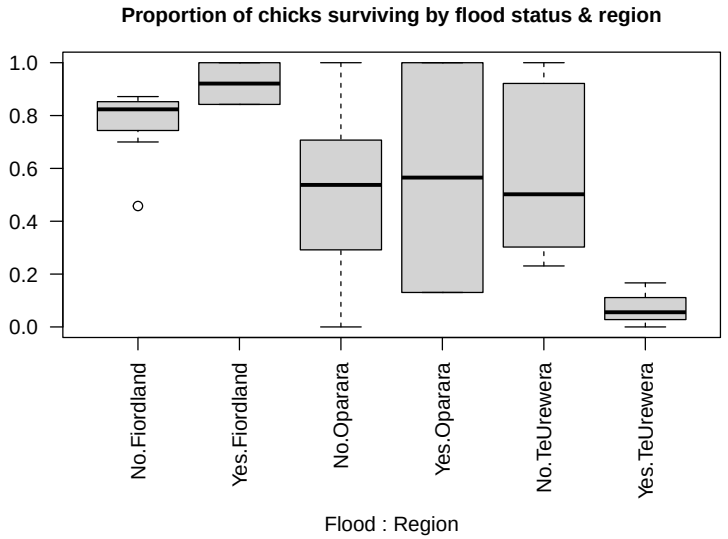
Does whio chick survival differ by region and/or flood status? Quantify all findings, including absolute survival probabilities and any differences found in survival among different predictor categories.

Read in and inspect the data

```
whio.df <- read.csv("WhioChicks.csv", stringsAsFactors=T)
head(whio.df)
```

```
##   Chicks Survived Died    Region Flood
## 1     46       39    7 Fiordland    No
## 2     59       27   32 Fiordland    No
## 3     56       48    8 Fiordland    No
## 4     51       42    9 Fiordland    No
## 5     78       68   10 Fiordland    No
## 6     80       56   24 Fiordland    No
```

```
par(mar=c(8, 4, 3, 2), mgp=c(7, 1, 0))
with(whio.df, boxplot(Survived/Chicks ~ Flood*Region, data = whio.df, las=2))
title("Proportion of chicks surviving by flood status & region", cex.main=1)
```


```
# Numbers of observations by predictor levels:
table(whio.df$Flood, whio.df$Region)
```

```
##
##       Fiordland Oparara TeUrewera
##   No          7       8        10
##   Yes         2       2         3
```

Comment on the plot and summary information

Chick survival appears to be higher in Fiordland than in the other two regions. There is no consistent pattern of average survival with flood status. In Fiordland, average survival appears a little higher in flood years, but in Oparara there appears to be little difference, and in TeUrewera average survival is noticeably lower in flood years. However, we can see from the summary table that there are only a few observations for flood years so it might be difficult to use this data to draw conclusions about the impact of floods.

Model building and check assumptions

```
whio.fit1 <- glm(cbind(Survived, Died) ~ Flood * Region, family=binomial, data=whio.df)
summary(whio.fit1)
```

```
##
```

```
## Call:
## glm(formula = cbind(Survived, Died) ~ Flood * Region, family = binomial,
##     data = whio.df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -5.003  -1.442   0.138   1.745   6.259
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)                1.1583     0.1129  10.255  < 2e-16 ***
## FloodYes                   1.1771     0.6150   1.914  0.05564 .
## RegionOparara             -1.0405     0.2156  -4.826 1.39e-06 ***
## RegionTeUrewera           -0.8706     0.1484  -5.867 4.43e-09 ***
## FloodYes:RegionOparara    -2.4988     0.7929  -3.152  0.00162 **
## FloodYes:RegionTeUrewera  -3.9905     0.9631  -4.144 3.42e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 371.19  on 31  degrees of freedom
## Residual deviance: 257.10  on 26  degrees of freedom
## AIC: 359.79
##
## Number of Fisher Scoring iterations: 4
```

```
1-pchisq(257.10, 26)
```

```
## [1] 0
```

```
whio.fit2 <- glm(cbind(Survived, Died) ~ Flood * Region, family=quasibinomial, data=whio.df)
anova(whio.fit2, test="F")
```

```
## Analysis of Deviance Table
##
## Model: quasibinomial, link: logit
##
## Response: cbind(Survived, Died)
##
## Terms added sequentially (first to last)
##
##
##              Df Deviance Resid. Df Resid. Dev      F  Pr(>F)
## NULL                            31     371.19
## Flood         1   13.300        30     357.89 1.5112 0.22996
## Region        2   70.597        28     287.29 4.0110 0.03032 *
## Flood:Region  2   30.195        26     257.10 1.7156 0.19960
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



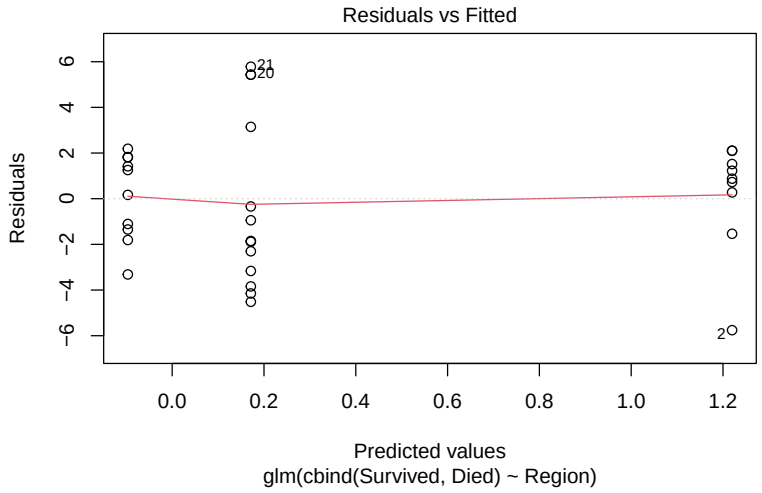
```
whio.fit3 <- glm(cbind(Survived, Died) ~ Flood + Region, family=quasibinomial, data=whio.df)
anova(whio.fit3, test="F")
```

```
## Analysis of Deviance Table
##
## Model: quasibinomial, link: logit
##
## Response: cbind(Survived, Died)
##
## Terms added sequentially (first to last)
##
##
##        Df Deviance Resid. Df Resid. Dev      F  Pr(>F)
## NULL                      31     371.19
## Flood   1   13.300        30     357.89 1.4847 0.23321
## Region  2   70.597        28     287.29 3.9406 0.03105 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
whio.fit4 <- glm(cbind(Survived, Died) ~ Region, family=quasibinomial, data=whio.df)
anova(whio.fit4, test="F")
```

```
## Analysis of Deviance Table
##
## Model: quasibinomial, link: logit
##
## Response: cbind(Survived, Died)
##
## Terms added sequentially (first to last)
##
##
##        Df Deviance Resid. Df Resid. Dev      F  Pr(>F)
## NULL                      31     371.19
## Region  2   72.374        29     298.82 4.0394 0.02834 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(whio.fit4, which=1)
```



Inference output from the final model

```
predictGLM(whio.fit4, type="response", newdata=data.frame(Region=c("Fiordland", "Oparara", "TeUrewera")
```

```
## ***Estimates and CIs are on the response scale***
```

```
##         fit       lwr       upr
## 1 0.7720430 0.6390892 0.8662676
## 2 0.4758621 0.2550035 0.7065836
## 3 0.5427350 0.4078230 0.6716570
```

```
# Use emmeans for pairwise comparisons:
whio.pairs <- pairs( emmeans(whio.fit4, "Region"), infer=T)
# Convert to data-frame and remove unwanted columns for display:
whio.pairs <- data.frame(whio.pairs)[,-c(3,4,7)]
whio.pairs
```

```
##                contrast   estimate   asymp.LCL asymp.UCL    p.value
## 1   Fiordland - Oparara  1.3165101 -0.08417811  2.717198 0.07067840
## 2 Fiordland - TeUrewera  1.0485250  0.03611644  2.060934 0.04031836
## 3   Oparara - TeUrewera -0.2679851 -1.60376512  1.067795 0.88529644
```



```
exp(data.frame(whio.pairs)[,c(2, 3, 4)])
```

```
##    estimate asymp.LCL asymp.UCL
## 1 3.7303801 0.9192675 15.137852
## 2 2.8534393 1.0367766  7.853299
## 3 0.7649192 0.2011378  2.908958
```

```
100*( exp(data.frame(whio.pairs)[,c(2, 3, 4)]) - 1 )
```

```
##    estimate  asymp.LCL asymp.UCL
## 1 273.03801  -8.073249 1413.7852
## 2 185.34393   3.677656  685.3299
## 3 -23.50808 -79.886222  190.8958
```

Methods and Assumption Checks

We fitted a logistic regression using a Binomial GLM to investigate the relationship between whio chick survival and two categorical predicators: Region, specifying which of three regions each record corresponded to; and Flood, specifying whether or not the region was affected by floods for each record.

We initially fitted the model with an interaction between the two categorical predictors. The test for overdispersion was significant so we refitted the model using the quasibinomial family. Having corrected for overdispersion, the interaction term was non-significant so it was dropped from the model. Refitting the model with main effects only, the Flood variable was non-significant so it was also dropped. The final model involved just the single predictor, Region.

The residual plot revealed no concerns with the final model. However, there might be concerns with non-independence between outcomes for different chicks, because many nests will have contained multiple chicks, which might all die together if the whole nest was destroyed. This might explain the overdispersion in the data.

The final model was:

$$\log(\text{odds}_{ij}) = \mu + \alpha_i$$

where odds_{ij} denotes the odds that chick j in region i survives to leave the nest, and where μ is the overall mean and α_i is the region effect for region i.

Executive Summary

We were interested in whether whio chick survival differs according to region and/or flood status.

There was evidence that survival differed by region; in particular, that the Fiordland region had a higher average chick survival rate than the Te Urewera region. There was no evidence that survival differed by flood status, although we note that there were only a few observations from flood years, so more data might be needed to investigate this question more thoroughly.

We estimate the probability of a chick surviving to leave the nest was:

- between 64% and 87% in the Fiordland region;
- between 26% and 71% in the Oparara region;

- between 41% and 67% in the Te Urewera region.

We estimate that the odds of chick survival in the Fiordland region were between 1.04 and 7.85 times those in the Te Urewera region.

[Alternatively: We estimate that the odds of chick survival in the Fiordland region were between 4% and 685% higher than those in the Te Urewera region.]

Chapter 16: Analysis of contingency tables

STATS 201/8

University of Auckland

Learning Outcomes

In this chapter you will learn about:

- Contingency tables from grouping categorical data
- Modelling contingency tables using `family=binomial`
- Modelling contingency tables using `family=poisson`
- The equivalence of the binomial and Poisson models
- A new interpretation of odds
- Odds ratios (optional section)
- Chi-square test of association (optional section)

Section 16.1 Introduction

Categorical data

Count data often arise from the observation of categorical data.

Data are said to be "categorical" if the measurements made on each subject are ALL factor variables.

The levels of the factor variables are the "categories", that is, they are the distinct values that the factor variable can take.

The counts are then the number of times (i.e., frequencies) each combination of factor levels occurs.

The counts can be arranged in the form of a contingency table.

Categorical data example...

Vaccine study

Suppose that two vaccine treatments (Trmt A and Trmt B) are to be compared for local tenderness around the injection site. Each subject receives one of the two vaccines, and the degree of local tenderness is classified into one of four levels.

- no tenderness
- mild tenderness
- moderate tenderness
- severe tenderness

The number of each treatment-tenderness combination can then be counted and presented in the form of a contingency table:

	none	mild	moderate	severe
Trmt A	21	16	11	2
Trmt B	1	22	19	9

Categorical data example

Hair and eye colour study

A genetic study wanted to determine if eye colour is associated with hair colour.

A large collection of randomly chosen online portrait photographs were examined and hair and eye colour was classified as brown or not brown (other). The resulting contingency table was

	Eyes brown	Eyes other
Hair brown	284	613
Hair other	577	2002

Categorical data...

Attendance/Pass

It is of interest to examine whether attendance of STATS 20x lectures had an association with success (pass or fail) in the course. Recall, we have data for the class of 146 students.

The raw format for recording categorical data is the usual rectangular format with rows being the observations on each subject, and columns being the measured factor variables. The first 8 lines of the raw data look like this:

```
> AP.df = read.table("Data/AttendPass.txt",header=T)
> head(AP.df, 8)
  Subject Pass     Attend
1       1 pass     attend
2       2 pass     attend
3       3 pass     attend
4       4 pass     attend
5       5 pass     attend
6       6 fail not.attend
7       7 pass not.attend
8       8 fail     attend
```


Categorical data...

Attendance/Pass...

The contingency table of counts can be obtained using the `table` function.

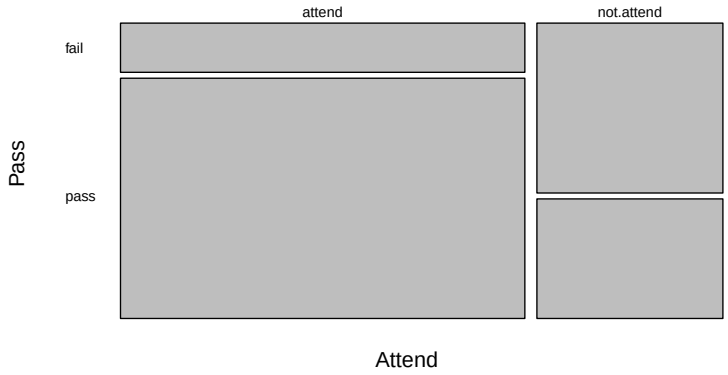
```
> AP.tbl=with(AP.df,table(Attend,Pass))
> AP.tbl
            Pass
Attend       fail pass
  attend       17   83
  not.attend   27   19
```

Categorical data...

Attendance/Pass – plotting the counts

It is easy to get some useful plots of the contingency table. Since `AP.tbl` is a table, the `plot` function produces a mosaic plot:

```
> plot(AP.tbl,main="",las=1)
```



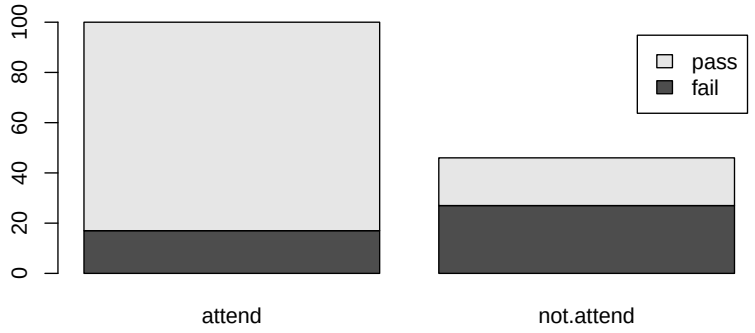
Note that the size of the rectangles is proportional to the count value. What does this plot tell us?

Categorical data...

Attendance/Pass – plotting the counts ...

The `barplot` function provides a useful bar plot:

```
> barplot(t(AP.tbl),legend=T)
```

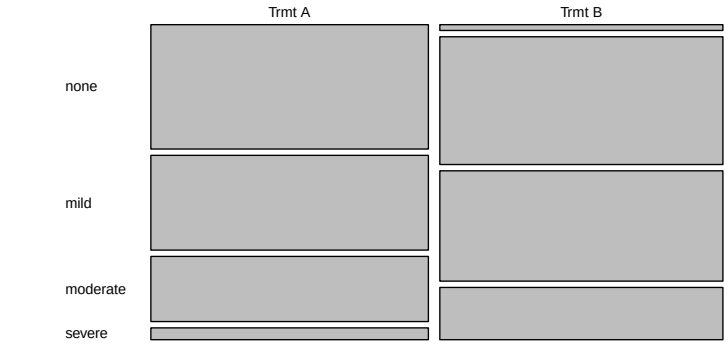


In the above code we used the transpose function `t` to flip the table rows and columns so that each bar would be for a level of `Attend`. If we hadn't done this the bars would have been for the levels of `Pass`, which makes interpretation harder.

Categorical data...

Vaccine tenderness – plotting the counts

```
> vaccines=matrix(c(21,16,11,2,1,22,19,9),nrow=2,byrow=T)
> rownames(vaccines)=c("Trmt A","Trmt B")
> colnames(vaccines)=c("none","mild","moderate","severe")
> vax.tbl=as.table(vaccines)
> vax.tbl
       none mild moderate severe
Trmt A   21   16       11      2
Trmt B    1   22       19      9
> plot(vax.tbl,main="",las=1)
```

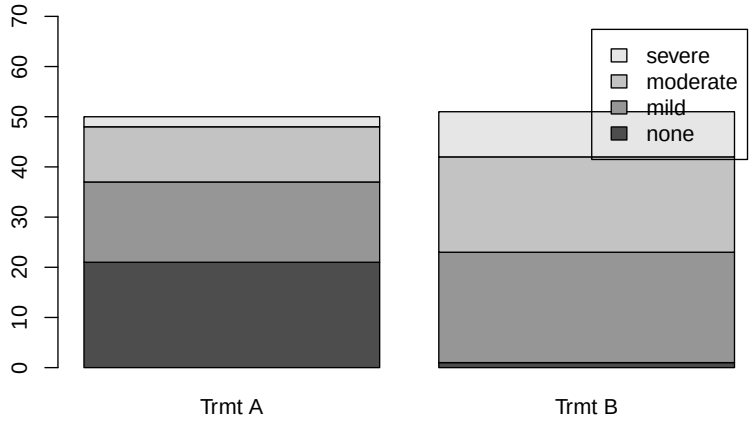


Categorical data...

Vaccine tenderness – plotting the counts ...

The `barplot` function is not very good at placing the legend!

```
> barplot(t(vax.tbl),ylim=c(0,70),legend=T)
```



Remark 1

In each of the above examples the frequencies were presented in the form a two-way (i.e., by row and by column) contingency table.

The contingency table was 2-by-4 for the vaccine study, and 2-by-2 in the hair/eye colour and attendance/pass example. In general, the number of rows and columns in a two-way contingency table is given by the number of levels of the corresponding factors.

More generally, if s factor variables are recorded on each subject then the resulting table will be a s-way contingency table, with dimensions given by the number of levels of each of the s factors.

Remark 2

With categorical data, it may or may not be possible to identify some of the factor variables as explanatory variables and some as response variables.

It totally depends on the particular situation:

- The vaccine study was conducted to see if tenderness depends on which vaccine treatment was given, so it is clear that treatment is the explanatory variable, and the measured tenderness is the response.
- In the hair/eye colour example neither is clearly an explanatory or response.
- In the attendance/pass example it is natural to consider attendance as an explanatory variable for pass, and indeed we have already done that in this course.

Remark 3

With categorical data, it may or may not be possible to say whether the counts are from a fixed number of trials (e.g., binomial data), or are more like Poisson data.

- In the vaccine example it is most likely that there was a fixed number of treatments applied, so if we created a dichotomous response (such as, "no tenderness" or "some tenderness") then we could model it as binomial.
- In the hair/eye colour example there is no clear notion of a fixed number of trials for any hair or eye colour.
- In the attendance/pass example:
  - One could say that the number of attenders and non-attenders was fixed (at 100 and 46, respectively) prior to the exam.
  - Alternatively, one could argue that the the number of attenders and non-attenders was not fixed since it depended on the number of enrolments.

Remarks

The last two slides above note some interesting properties of categorical data. As we shall see, these considerations don't matter to our analysis – but they may determine our interpretation.

The underlying research question is to establish whether or not there is an association between the factors.

If there is an association then we would also like to be able to quantify it.

NOTE: This Chapter concludes with a recap of the methodology seen in STATS 10x for testing for association in contingency tables – the chi-squared test for association. Your lecturer will advise whether it is examinable.

Section 16.2
The binomial approach to contingency table analysis

Attendance/Pass

Two STATS 20x students, Kim and Des, have been assigned to the task of determining whether there is an association between attendance and exam success in STATS 20X.

Kim has decided to use the binomial approach to analyse the data – this makes sense, since we can regard attendance as an explanatory variable, and pass/fail as a Bernoulli outcome.

We'll help Kim out by creating a dataframe in the format needed for a binomial GLM.

```
> Freqs.df = data.frame(Attend=c("not.attend","attend"),Fail=c(27,17),Pass=c(19,83))
> Freqs.df = transform(Freqs.df,Attend=factor(Attend))
> Freqs.df
      Attend Fail Pass
1 not.attend   27   19
2     attend   17   83
```

If an association is detected then Kim also needs to quantity the strength of the assocation with a suitable confidence interval.

Attendance/Pass. . .

Binomial analysis

Kim decides to change the reference level of `Attend` to `not.attend` so that any effect can be expressed as the effect of attending.

Fitting the binomial GLM is easy:

```
> AP.binom=glm(cbind(Pass,Fail)~Attend,data=Freqs.df,family=binomial)
> summary(AP.binom)
```

```
Call:
glm(formula = cbind(Pass, Fail) ~ Attend, family = binomial,

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   -0.3514     0.2994  -1.173    0.241
Attendattend   1.9370     0.4007   4.834 1.34e-06 ***
---
(Dispersion parameter for binomial family taken to be 1)

    Null deviance:  2.5162e+01  on 1  degrees of freedom
Residual deviance: -4.2188e-15  on 0  degrees of freedom
```


Attendance/Pass. . .

Binomial analysis. . .

Note that the residual deviance is zero, on zero degrees of freedom – this is nothing to worry about.[1]

The effect of `Attend` is highly significant, so let's calculate a CI:

```
> exp(confint(AP.binom))[2,]
Waiting for profiling to be done...
    2.5 %    97.5 %
 3.214049 15.552487
```

So, Kim concludes that the odds of an attender passing STATS 20x are between 3.2 and 15.6 times the odds of a non-attender passing. Yikes!

[1] The data consist of two observations, and the model fits two parameters, so there are no degrees of freedom left.

Section 16.3
The Poisson approach to contingency table analysis

Attendance/Pass. . .

Poisson analysis

The other student, Des, feels that the frequencies are best modeled as Poisson counts because the number of students was not fixed at the start of the semester, and depended on the whims of enrolment.

Des needs the data in the following format:

```
> library(dplyr)
> AP.df = read.table("Data/AttendPass.txt",header=T)
> AP.df = transform(AP.df, Pass=factor(Pass), Attend=factor(Attend))
> Freqs2.df = AP.df %>% group_by(Attend,Pass) %>% summarize(freq=n()) %>%
+                   data.frame()
> Freqs2.df
      Attend Pass freq
1     attend fail   17
2     attend pass   83
3 not.attend fail   27
4 not.attend pass   19
```


Attendance/Pass. . .

Poisson analysis. . .

Des fits the interaction model to check for an association between attendance and passing.[2] He also relevels `Attend`.

```
> Freqs2.df$Attend=relevel(Freqs2.df$Attend, ref="not.attend")
> AP.pois=glm(freq~Attend*Pass,family=poisson,data=Freqs2.df)
> summary(AP.pois)
```

```
Call:
glm(formula = freq ~ Attend * Pass, family = poisson, data = Freqs2.df)

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)            3.2958     0.1925  17.126  < 2e-16 ***
Attendattend          -0.4626     0.3096  -1.494    0.135
Passpass              -0.3514     0.2994  -1.173    0.241
Attendattend:Passpass  1.9370     0.4007   4.834 1.34e-06 ***
---
(Dispersion parameter for poisson family taken to be 1)

    Null deviance:  6.9305e+01  on 3  degrees of freedom
Residual deviance: -3.9968e-15  on 0  degrees of freedom
```

[2] That is, to see if the expected numbers in the `pass` group relative to the `fail` group depend on attendance.

Attendance/Pass. . .

Poisson analysis. . .

As with the binomial model,the residual deviance is zero, on zero degrees of freedom. Again, this is nothing to worry about.[3]

Note that the interaction effect is highly significant. In fact, **the `Attend:Pass` interaction effect from Des's Poisson analysis is exactly the same as the `Attend` effect from Kim's binomial analysis**.

The CI for $\exp(\beta_3)$ from the Poisson model is identical to Kim's CI

```
> exp(confint(AP.pois))[4,]
Waiting for profiling to be done...
    2.5 %    97.5 %
 3.214049 15.552487
```

The next section shows why these two different models produce the same estimate.

[3] The data consist of four observed counts, and the model fits four parameters, so there are no degrees of freedom left.

Section 16.4
Equivalence of the binomial and Poisson approaches

Equivalence of binomial and Poisson approaches

Consider the 2-by-2 contingency table:

$$\begin{array}{cc} n_{11} & n_{12} \\ n_{21} & n_{22} \end{array}$$

We'll assume that row 1 and column 1 correspond to the reference levels for the row and column factor variables, respectively.

Then, from Chapters 13 and 14, the Poisson interaction model[4] for these data is

$$\begin{aligned} \log \mathrm{E}[n_{11}] &= \beta_0 \\ \log \mathrm{E}[n_{21}] &= \beta_0 + \beta_1 \\ \log \mathrm{E}[n_{12}] &= \beta_0 + \beta_2 \\ \log \mathrm{E}[n_{22}] &= \beta_0 + \beta_1 + \beta_2 + \beta_3 \end{aligned}$$

[4] Here we assume the model `glm(n ~ RowVar * ColVar, family=poisson)` where `RowVar` and `ColVar` are the row and column factor variables, respectively.

Equivalence of binomial and Poisson approaches...

An odds interpretation...

Equivalently,

$$\begin{aligned} \mathrm{E}[n_{11}] &= \exp(\beta_0) \\ \mathrm{E}[n_{21}] &= \exp(\beta_0 + \beta_1) \\ \mathrm{E}[n_{12}] &= \exp(\beta_0 + \beta_2) = \mathrm{E}[n_{11}] \times \exp(\beta_2) \\ \mathrm{E}[n_{22}] &= \exp(\beta_0 + \beta_1 + \beta_2 + \beta_3) = \mathrm{E}[n_{21}] \times \exp(\beta_2 + \beta_3) \end{aligned}$$

So, we can say that for every one occurrence expected in the $[1, 1]$ cell we expect $\exp(\beta_2)$ occurrences in the $[1, 2]$ cell.

Another way of interpreting this is that within row 1, an occurrence is $\exp(\beta_2)$ times as likely to be in column 2 than column 1.

What we've just said is that, within row 1, the odds of being in column 2 (rather than column 1) is $\exp(\beta_2)$.[5]

[5] Recall this example from Chapter 15: If the odds of A are 2 (i.e., 2-to-1) then we are saying that A is twice as likely to occur as not. That is Pr(A)=2/3 and Pr(not A)=1/3.

Equivalence of binomial and Poisson approaches...

An odds interpretation...

Let's compare the expected occurrences in the $[2, 1]$ and $[2, 2]$ cells.

Within row 2, an occurrence is $\exp(\beta_2 + \beta_3)$ times as likely to be in column 2 than column 1.

That is, within row 2, the odds of being in column 2 (rather than column 1) is $\exp(\beta_2 + \beta_3) = \exp(\beta_2) \times \exp(\beta_3)$.

Note that the multiplicative change in column 2 odds between row 2 and row 1 is $\exp(\beta_3)$.

Attendance/Pass...

Equivalence of binomial and Poisson approaches...

Recall, our contingency table for this example is

```
> Freqs.df
      Attend Fail Pass
1 not.attend   27   19
2     attend   17   83
```

and the CI for $\exp(\beta_3)$ from the Poisson model is

```
> exp(confint(AP.pois))[4,]
Waiting for profiling to be done...
    2.5 %    97.5 %
 3.214049 15.552487
```

We now have the same interpretation from the Poisson model as from the binomial model – the odds of an attender (row 2) passing STATS 20x (column 2) are between 3.2 and 15.6 times the odds of a non-attender passing.

Section 16.5
Closing remarks

Equivalence of binomial and Poisson approaches...

Closing remarks

The binomial and Poisson models fitted above are so-called *saturated* models because there are as many parameters as there are observations, and hence zero degrees of freedom.

A saturated model has perfect fit. To see that, let's look at the fitted counts from the Poisson model fitted to `Freqs2.df`.

```
> predict(AP.pois,type="response")
 1  2  3  4
17 83 27 19
> Freqs2.df
      Attend Pass freq
1     attend fail   17
2     attend pass   83
3 not.attend fail   27
4 not.attend pass   19
```

The fitted counts are exactly the observed counts – a perfect fit!

Equivalence of binomial and Poisson approaches...

Closing remarks

- The odds interpretation that we are using for Poisson analysis of contingency tables is generally not appropriate for the other types of Poisson count data that were seen in Chapters 13 and 14.[6]
- The odds multiplier was $\exp(\beta_1)$ from the binomial model, and equivalently $\exp(\beta_3)$ from the Poisson model. This is commonly called the **odds-ratio**. E.g., in row 2 the odds-ratio (relative to row 1) for the column 2 outcome is between 3.2 and 15.6.
- In practice, the **Poisson approach is most widely used** because
  - The binomial approach is limited to a binary response category.
  - The Poisson approach can handle tables of arbitrary number of row and/or columns.
  - The Poisson approach can handle multi-way tables.
  - The Poisson approach can test other types of hypotheses, e.g., that all row outcomes are equally likely.

[6] You will lose marks if used inappropriately.

Section 16.6
Odds ratios

(This is an optional Section
- your lecturer will advise if it is examinable)

Odds ratios

Recall, the contingency table for the attendance/pass example is

```
> Freqs.df
      Attend Fail Pass
1 not.attend   27   19
2     attend   17   83
```

and the fitted Poisson model is

```
> coef(summary(AP.pois))
                      Estimate Std. Error   z value     Pr(>|z|)
(Intercept)          3.2958369  0.1924501 17.125671 9.550242e-66
Attendattend        -0.4626235  0.3096136 -1.494197 1.351243e-01
Passpass            -0.3513979  0.2994472 -1.173489 2.405999e-01
Attendattend:Passpass  1.9370252  0.4006749  4.834407 1.335434e-06
```

Our estimated value of the odds-ratio is therefore

```
> exp(coef(AP.pois))[4]
Attendattend:Passpass
             6.93808
```


Odds ratios...

It can be shown that the estimated odds-ratio is simply the product of the diagonal values in the table, divided by the product of the off-diagonal values.

In this case

$$\widehat{\text{OR}} = \frac{n_{11} \times n_{22}}{n_{12} \times n_{21}} = \frac{27 \times 83}{17 \times 19}$$

```
> options(digits=4) # Set the number of significant digits to 4
> OR=27*83/(17*19)
> OR
[1] 6.938
```


Inference for odds ratios

The distribution of the estimated odds ratio is difficult to obtain, and is skewed even in large samples.

However, the distribution of $\log(\widehat{\text{OR}})$ is approximately normal for large samples.

It can be shown (see STATS 730) that the standard error of $\log(\widehat{\text{OR}})$ is approximately

$$\text{se}\left(\log(\widehat{\text{OR}})\right) = \sqrt{\frac{1}{n_{11}} + \frac{1}{n_{12}} + \frac{1}{n_{21}} + \frac{1}{n_{22}}}$$

This can be used to obtain approximate confidence intervals for $\log(\widehat{\text{OR}})$, which can be exponentiated to obtain CI's for $\widehat{\text{OR}}$.

Numerous packages in R (e.g., epitools) will do these calculations for you.

Inference for odds ratios

Attendance/Pass example

Evaluating the above formula for the approximate standard error of $\log(\widehat{OR})$ gives

```
> logOR.se=sqrt(1/17+1/83+1/27+1/19)
> logOR.se
[1] 0.4007
```

and the approximate 95% CI is therefore

```
> logOR.CI=log(OR) + c(-1,1)*1.96*logOR.se
> logOR.CI
[1] 1.152 2.722
```

The approximate 95% CI for the odds ratio is therefore

```
> exp(logOR.CI)
[1]  3.164 15.216
```

Compare with

```
> exp(confint.default(AP.pois)[4,] )
 2.5 % 97.5 %
 3.164 15.216
```


Section 16.7
Analysing Attendance/Pass using STATS 10x methods
1) difference in proportions
2) chi-squared test for association

(This is an optional Section
- your lecturer will advise if it is examinable)

Method 1: Difference in proportions

We'll let p_1 and p_2 denote the probabilities of passing for attenders and non-attenders, respectively.

Recall that the standard error of a difference in proportions is estimated by

$$se(\hat{p}_1 - \hat{p}_2) = \sqrt{\frac{\hat{p}_1(1-\hat{p}_1)}{n_1} + \frac{\hat{p}_2(1-\hat{p}_2)}{n_2}}$$

Our approximate 95% confidence interval for $p_1 - p_2$ is calculated as

$$\hat{p}_1 - \hat{p}_2 \pm 1.96 \times se(\hat{p}_1 - \hat{p}_2)$$

where $\hat{p}_1$ and $\hat{p}_2$ are the proportions of passes. That is, $\hat{p}_1 = 83/100 = 0.83$ and $\hat{p}_2 = 19/46 = 0.413$.

In R:

```
> p1 = 83/100; p2 = 19/46
> se = sqrt(p1 * (1 - p1)/100 + p2 * (1 - p2)/46)
> ## Calculate the confidence interval
> p1 - p2 + 1.96 * c(-1, 1) * se
[1] 0.2567 0.5772
```


Attendance/Pass

Using the difference in proportions passing

The above 95% CI does not include 0, so this is a statistically significant difference (p-value < 0.05), so we can conclude that passing is highly associated with attendance.

This tell us that attenders have a probability of passing that is between 0.26 and 0.58 higher than that of non-attenders.

Method 2: Chi-square test for association in a contingency table

Recall:

```
> AP.df = read.table("Data/AttendPass.txt",header=T)
> AP.tbl=with(AP.df,table(Attend,Pass))
> AP.tbl
             Pass
Attend        fail pass
  attend        17   83
  not.attend    27   19
```

These counts are re-arranged in the form of a 2×2 contingency table.

Chi-squared goodness of fit test

Attendance/Pass...

The standard Chi-squared test is given by

```
> chisq.test(AP.tbl,correct=FALSE)

Pearson's Chi-squared test

data:  AP.tbl
X-squared = 26, df = 1, p-value = 3e-07
```

The null hypothesis of no association between attendance and course success. In this case, the test tell us that there is massive evidence against this hypothesis—which we already knew.

We will now examine how this test works.

Tests of association in a 2-way table

Notation

Here is a way of considering a general table of counts with an arbitrary number of levels for each factor variable.

		Outcome				
		1	2	⋯	c	
Group	1	n_{11}	n_{12}	⋯	n_{1c}	n_{1+}
	2	n_{21}	n_{22}	⋯	n_{2c}	n_{2+}
	⋮	⋮	⋮	⋯	⋮	⋮
	r	n_{r1}	n_{r2}	⋯	n_{rc}	n_{r+}
		n_{+1}	n_{+2}	⋯	n_{+c}	

Row totals

Column totals

n_{ij} = number of subjects in group (row) i and having outcome (column) j.

$n = \sum_{i=1}^{r} \sum_{j=1}^{c} n_{ij}$ = total number of subjects.

Tests of association in a 2-way table: χ^2 test

Deriving the χ^2 test using the Poisson assumption

The Chi[7]-squared (χ^2) test of independence (between row and column factors) compares the observed counts to those one would expect if there was no association between the row and column factors.

If the row and columns factors are independent then the probability of a subject being placed in cell (i, j) of the table equals the product of the probability that they are in row i (i.e., level i of the row factor occurs) times the probability that they are in column j (i.e., level j of the row factor occurs).

You have seen the χ^2 test before (I hope). Here, we are going to derive the χ^2 test using what we know about the properties of the Poisson distribution.

[7] Pronounced /kai/ - a long i sound like ride

Tests of association in a 2-way table: χ^2 test

Deriving the χ^2 test using the Poisson assumption...

- Our estimate of the row i probability is $\frac{n_{i+}}{n}$.
- Our estimate of the column j probability is $\frac{n_{+j}}{n}$.
- Under the null hypothesis of row and column independence, our estimate of the cell (i,j) probability is

$$\frac{n_{i+}}{n} \times \frac{n_{+j}}{n}$$

- So, under the null hypothesis, the expected value of the the number of subjects in cell (i,j) is estimated to be

$$\hat{n}_{ij} = n \times \frac{n_{i+}}{n} \times \frac{n_{+j}}{n}$$

Tests of association in a 2-way table: χ^2 test...

Deriving the χ^2 test using the Poisson assumption...

Next we have to determine whether the residuals, $n_{ij} - \hat{n}_{ij}$, are of sufficient magnitude to provide evidence against the null hypothesis.
We do this by calculating a Z-statistic for each cell:

$$Z_{ij} = \frac{n_{ij} - \hat{n}_{ij}}{sd(n_{ij})}$$

where $sd(n_{ij})$ is the standard deviation of n_{ij}.

Now, we will utilize the assumption that the counts are Poisson distributed. Recall that Poisson count data have variance equal to the mean. We've estimated the mean to be $\hat{n}_{ij}$, so we estimate $sd(n_{ij})$ to be

$$sd(n_{ij}) = \sqrt{\text{Var}(n_{ij})} = \sqrt{\text{E}(n_{ij})} \approx \sqrt{\hat{n}_{ij}}$$

and so our Z-statistic for cell $[i,j]$ of the table is

$$Z_{ij} = \frac{n_{ij} - \hat{n}_{ij}}{\sqrt{\hat{n}_{ij}}}$$

Tests of association in a 2-way table: χ^2 test...

Deriving the χ^2 test using the Poisson assumption...

The chi-squared test statistic is given by summing the square of the Z_{ij} statistics over all cells in the table. That is,

$$X = \sum_i \sum_j \frac{(n_{ij} - \hat{n}_{ij})^2}{\hat{n}_{ij}}$$

Under H_0 (no association between rows and columns), X has an approximate $\chi^2_{(r-1)(c-1)}$ distribution, where $(r-1)(c-1)$ is the degrees of freedom[8]. So, for a 2-by-2 table, it is approximately χ^2_1 distributed.

[8] Degrees of freedom are calculated as the number of independent data points minus the number of estimated parameters—can you do the maths here?

Tests of association in a 2-way table: χ^2 test...

The χ^2 test p-value

Large values of X provide evidence against H_0. The P-value is the "probability to the right of X", where the probability is from a $\chi^2_{(r-1)(c-1)}$ distribution.

In `R`, the probability that a χ^2_q is *less than* X is given by `pchisq(X,q)`. The P-value is given by 1 minus this amount. That is,

P-value = `1-pchisq(X,q)`

For a 2-by-2 table, we are working with χ^2_1, so the P-value is the probability that a value from a χ^2_1 distribution exceeds X. So, for a 2-by-2 table with $X = 3.1$, the P-value would be

```
> 1-pchisq(3.1,1)
[1] 0.07829
```

Tests of association in a 2-way table: χ^2 test...

Validity of the χ^2 test

NOTE: The χ^2 test uses the square of Z-statistics. These require that n_{ij} is at least approximately normally distributed.

Recall - this requires the expected counts to be sufficiently large.

That is why implementations of the χ^2 test often give warnings if some of the estimated expected counts ($\hat{n}_{ij}$) are "too small". 'Too small" is commonly taken to be < 5.

If this assumption is violated then there is a solution that uses a permutation technique called Fisher's exact test.

Example—Attendance/Pass

```
> chisq.test(AP.tbl)

Pearson's Chi-squared test with Yates' continuity correction

data:  AP.tbl
X-squared = 24, df = 1, p-value = 9e-07
```

This is a modified form of the chi-squared test that does a continuity correction to the Z_{ij} values, so as to correct for the fact that the cell counts are integer valued.

Attendance/Pass

Chi-squared goodness of fit test

The standard chi-squared test (with X defined as above) is given by

```
> chisq.test(AP.tbl,correct=FALSE)

Pearson's Chi-squared test

data:  AP.tbl
X-squared = 26, df = 1, p-value = 3e-07
```

In this case, both forms of the test tell us that there is massive evidence against the null hypothesis of no association between attendance and course success.

If the difference between the corrected and standard chi-squared tests makes any meaningful difference then you are probably up to no good!

Attendance/Pass...

The expected values can be obtained by saving the result of `chisq.test` as an `R` object, and printing the `$expected` component of it.

```
> AP.chisq = chisq.test(AP.tbl, correct=FALSE)
> AP.chisq$expected
            Pass
Attend        fail  pass
  attend     30.14 69.86
  not.attend 13.86 32.14
```

The actual counts for comparison:

```
> AP.tbl
            Pass
Attend       fail pass
  attend       17   83
  not.attend   27   19
```

Case Study 16.1: Gender bias in admissions to Berkeley

Tou Ohone Andate - staff number 1234567

Problem

The University of California, Berkeley, was sued for gender bias against women based on the enrolment decisions given to 4526 students who applied for entry into the graduate program in 1973.

The variables of interest were:

- `Dept`: Six-level categorical variable with levels 'A', 'B', 'C', 'D', and 'E'.
- `Gender`: Two-level categorical variable with levels 'F' and 'M'.
- `Outcome`: Two-level categorical variable with levels 'In' and Out'.
- `Freq`: A count variable which corresponds to the observed combination of `Dept`, `Gender`, and `Outcome`.

Question of Interest

We want examine the association between gender and admission outcome.

Read in and naive analysis

```
Berk.df = read.table("BerkeleyEnrolments.txt", header = TRUE)
head(Berk.df)
```

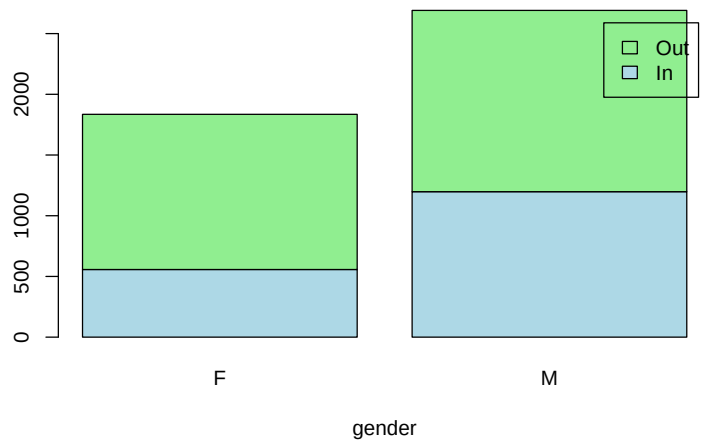
```
##   Dept Gender Outcome Freq
## 1    A      M      In  512
## 2    A      M     Out  313
## 3    A      F      In   89
## 4    A      F     Out   19
## 5    B      M      In  353
## 6    B      M     Out  207
```

```
Berk.df=transform(Berk.df,Dept=factor(Dept),Gender=factor(Gender),Outcome=factor(Outcome))
Berk.tbl = xtabs(Freq ~ Gender + Outcome, data = Berk.df)
Berk.tbl
```

```
##        Outcome
## Gender   In  Out
##      F  557 1278
##      M 1198 1493
```



```
barplot(t(Berk.tbl), main="Berkeley admissions  by gender", col=c("lightblue", "lightgreen"),
xlab="gender",legend.text=c("In", "Out"))
```




```
#Berk.gfit0 = glm(Freq ~ Gender * Outcome,family = poisson, data = Berk.df)
#The above fit is over-dispersed, so quasi fit used
Berk.quasifit0 = glm(Freq ~ Gender * Outcome,family = quasipoisson, data = Berk.df)
coef(summary(Berk.quasifit0))
```

```
##                      Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)         4.5308058  0.4150710 10.9157374 7.108831e-10
## GenderM             0.7658435  0.5023801  1.5244306 1.430588e-01
## OutcomeOut          0.8304864  0.4973647  1.6697736 1.105422e-01
## GenderM:OutcomeOut -0.6103524  0.6258979 -0.9751629 3.411277e-01
```

```
exp(confint(Berk.quasifit0))[4,] #Odds ratio for females being kept out of Berkeley!!!
```

```
## Waiting for profiling to be done...
```

```
##     2.5 %    97.5 %
## 0.1512944 1.8107735
```

This appear to show an open and shut case of strong gender discrimination - the lawyers were rubbing their hands in glee.

However, we need to be careful. In STATS 330 (which follows on from this class) we learn that an association may not be causal. Here, we have a strong association between gender and outcome, but this does **not** in itself establish a causal relationship.

In STATS 330 we would consider whether there is an indirect causal pathway. Really, we want to ask "If all things else are the same, other than gender, are the odds the same for females and males?". This means that we need to take into consideration department, and compare males and females that are applying to the same department.

Taking department in to account makes the model quite a bit more complicated. Fitting a model with 3-way interactions does not allow for meaningful interpretations, so it was decided fit the model containing all of the two-way interactions:

Model Building and Check Assumptions

```
Berk.gfit = glm(Freq ~ Dept * Gender + Dept * Outcome + Gender * Outcome,
    family = poisson, data = Berk.df)
plot(Berk.gfit, which = 1)
```


```
# The next two calls are to reduce the amount of output produced by `anova'
Berk.gfit.anova = anova(Berk.gfit, test = "Chisq")
Berk.gfit.anova[, names(Berk.gfit.anova)]
```

```
##               Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                             23    2650.10
## Dept           5   159.52        18    2490.57   <2e-16 ***
## Gender         1   162.87        17    2327.70   <2e-16 ***
## Outcome        1   230.03        16    2097.67   <2e-16 ***
## Dept:Gender    5  1220.61        11     877.06   <2e-16 ***
## Dept:Outcome   5   855.32         6      21.74   <2e-16 ***
```



```
## Gender:Outcome  1     1.53         5      20.20   0.2159
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(Berk.gfit)
```

```
##
## Call:
## glm(formula = Freq ~ Dept * Gender + Dept * Outcome + Gender *
##     Outcome, family = poisson, data = Berk.df)
##
## Deviance Residuals:
##        1         2         3         4         5         6         7         8
## -0.75481   0.99471   1.96454  -3.15768  -0.03402   0.04449   0.15709  -0.22034
##        9        10        11        12        13        14        15        16
##  1.01273  -0.73839  -0.74367   0.54896   0.06760  -0.04741  -0.06911   0.05080
##       17        18        19        20        21        22        23        24
##  1.05578  -0.61236  -0.73617   0.42678  -0.20117   0.05113   0.19803  -0.05370
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)         4.27291    0.10182  41.966  < 2e-16 ***
## DeptB              -1.47804    0.22510  -6.566 5.17e-11 ***
## DeptC               1.08723    0.11610   9.365  < 2e-16 ***
## DeptD               0.60832    0.12179   4.995 5.89e-07 ***
## DeptE               0.34537    0.13179   2.621  0.00878 **
## DeptF              -1.13555    0.18196  -6.241 4.36e-10 ***
## GenderM             1.99859    0.10593  18.866  < 2e-16 ***
## OutcomeOut         -0.68192    0.09911  -6.880 5.97e-12 ***
## DeptB:GenderM       1.07482    0.22861   4.701 2.58e-06 ***
## DeptC:GenderM      -2.66513    0.12609 -21.137  < 2e-16 ***
## DeptD:GenderM      -1.95832    0.12734 -15.379  < 2e-16 ***
## DeptE:GenderM      -2.79519    0.13925 -20.073  < 2e-16 ***
## DeptF:GenderM      -2.00232    0.13571 -14.754  < 2e-16 ***
## DeptB:OutcomeOut    0.04340    0.10984   0.395  0.69277
## DeptC:OutcomeOut    1.26260    0.10663  11.841  < 2e-16 ***
## DeptD:OutcomeOut    1.29461    0.10582  12.234  < 2e-16 ***
## DeptE:OutcomeOut    1.73931    0.12611  13.792  < 2e-16 ***
## DeptF:OutcomeOut    3.30648    0.16998  19.452  < 2e-16 ***
## GenderM:OutcomeOut  0.09987    0.08085   1.235  0.21672
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 2650.095  on 23  degrees of freedom
## Residual deviance:   20.204  on  5  degrees of freedom
## AIC: 217.26
##
## Number of Fisher Scoring iterations: 4
```

```
1 - pchisq(20.204, 5)
```

```
## [1] 0.001144215
```

```
Berk.quasigfit = glm(Freq ~ Dept * Gender + Dept * Outcome + Gender * Outcome,
    family = quasipoisson, data = Berk.df)
# The next two calls are to reduce the amount of output produced by `anova'
Berk.quasigfit.anova = anova(Berk.quasigfit, test = "F")
Berk.quasigfit.anova[, names(Berk.quasigfit.anova)]
```

```
##               Df Deviance Resid. Df Resid. Dev       F    Pr(>F)
## NULL                              23    2650.10
## Dept           5   159.52        18    2490.57  8.4743 0.0174851 *
## Gender         1   162.87        17    2327.70 43.2616 0.0012192 **
## Outcome        1   230.03        16    2097.67 61.0985 0.0005495 ***
## Dept:Gender    5  1220.61        11     877.06 64.8424 0.0001519 ***
## Dept:Outcome   5   855.32         6      21.74 45.4370 0.0003612 ***
## Gender:Outcome 1     1.53         5      20.20  0.4067 0.5516975
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(Berk.quasigfit)
```

```
##
## Call:
## glm(formula = Freq ~ Dept * Gender + Dept * Outcome + Gender *
##     Outcome, family = quasipoisson, data = Berk.df)
##
## Deviance Residuals:
##        1         2         3         4         5         6         7         8
## -0.75481   0.99471   1.96454  -3.15768  -0.03402   0.04449   0.15709  -0.22034
##        9        10        11        12        13        14        15        16
##  1.01273  -0.73839  -0.74367   0.54896   0.06760  -0.04741  -0.06911   0.05080
##       17        18        19        20        21        22        23        24
##  1.05578  -0.61236  -0.73617   0.42678  -0.20117   0.05113   0.19803  -0.05370
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        4.27291    0.19756  21.628 3.92e-06 ***
## DeptB             -1.47804    0.43678  -3.384 0.019589 *
## DeptC              1.08723    0.22526   4.827 0.004771 **
## DeptD              0.60832    0.23631   2.574 0.049777 *
## DeptE              0.34537    0.25571   1.351 0.234719
## DeptF             -1.13555    0.35307  -3.216 0.023564 *
## GenderM            1.99859    0.20555   9.723 0.000196 ***
## OutcomeOut        -0.68192    0.19231  -3.546 0.016458 *
## DeptB:GenderM      1.07482    0.44358   2.423 0.059892 .
## DeptC:GenderM     -2.66513    0.24466 -10.893 0.000113 ***
## DeptD:GenderM     -1.95832    0.24707  -7.926 0.000515 ***
## DeptE:GenderM     -2.79519    0.27019 -10.345 0.000145 ***
## DeptF:GenderM     -2.00232    0.26333  -7.604 0.000625 ***
## DeptB:OutcomeOut   0.04340    0.21312   0.204 0.846672
## DeptC:OutcomeOut   1.26260    0.20690   6.102 0.001711 **
## DeptD:OutcomeOut   1.29461    0.20533   6.305 0.001477 **
## DeptE:OutcomeOut   1.73931    0.24470   7.108 0.000854 ***
## DeptF:OutcomeOut   3.30648    0.32982  10.025 0.000169 ***
```



```
## GenderM:OutcomeOut  0.09987    0.15687   0.637 0.552355
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 3.764865)
##
##     Null deviance: 2650.095  on 23  degrees of freedom
## Residual deviance:   20.204  on  5  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

Produce the Additional Output Required

```
# Before adjusting for individual departments
chisq.test(Berk.tbl, correct = FALSE)
```

```
##
##  Pearson's Chi-squared test
##
## data:  Berk.tbl
## X-squared = 92.205, df = 1, p-value < 2.2e-16
```

```
# After adjusting for individual departments
Berk.summ = coef(summary(Berk.quasigfit))
Berk.summ[nrow(Berk.summ), ]
```

```
##   Estimate Std. Error    t value   Pr(>|t|)
## 0.09987009 0.15686847 0.63664858 0.55235491
```

```
c(exp(Berk.summ[nrow(Berk.summ), 1]), exp(confint(Berk.quasigfit)[nrow(Berk.summ), ]))
```

```
## Waiting for profiling to be done...
```

```
##               2.5 %    97.5 %
## 1.1050274 0.8137126 1.5058043
```

Method and Assumption Checks

Our response variable is a count broken down by three explanatory factors, gender, outcome, and department. Since we want to access the effect of gender, all else being equal, we fitted a generalized linear 3-way ANOVA model that included all possible two-way interaction terms. From looking at the data by department we stongly anticipate that some of the second-order interaction terms with department will be significant.

The residual plot shows no strong trends. The check of residual deviance had p-value 0.001 and the model was refitted as quasipoisson.

Executive Summary

We want to determine if there was gender bias at the University of Berkeley. A naive analysis of the data strongly suggests there is. However, this analysis does not compare like to like. That is, it does not compare students applying to the same department. We fitted a model that included all department in addition to gender and outcome, and found that there was no significant effect.

For completeness (despite the non-significant effect), we quantify our conclusions below:

The estimated effect is that for every female declined, 1.11 males are declined, once we adjust for the confounding effect of department. [NOTE: This is equivalent to saying that the odds of a male being declined are estimated to be 1.11 compared to females.] That is, on average about 11% more males are declined than females after adjustment for department. The confidence interval for the ratio of odds is 0.81 to 1.51.

We had evidence that the association between gender and admission outcome was significant before adjusting for individual departments ($P\text{-}value \approx 0$). However, once we adjusted for individual departments, there is insufficient evidence to say that the true ratio is different from 1 ($P\text{-}value = 0.55$).

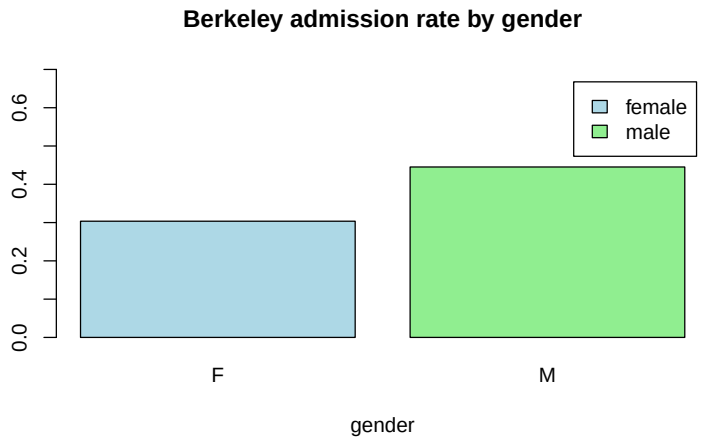
Lecturer Comment

What is happening here is that females are tending to apply more for the departments that have lower admission rates (perhaps law and medicine). Males are tending to apply more for the departments with higher admission rates (perhaps statistics!!). The association between gender and admission outcome is genuine, but is not caused by gender, but rather the difference in departmental preferences between genders.

Plots for revealing the deparmental preferences:

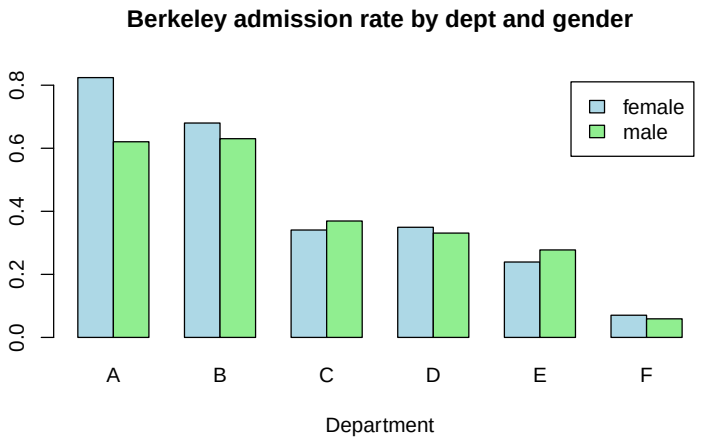
Gender vs Acceptance

```
propn.gender=Berk.tbl[,1]/apply(Berk.tbl,1,sum)
barplot(propn.gender,ylim=c(0,.7), main="Berkeley admission rate by gender", col=c("lightblue", "lightg
        xlab="gender",legend.text=c("female", "male"))
```

Looking at outcome by department for each gender

```
BerkByDept = xtabs(Freq ~ Gender + Outcome + Dept, data = Berk.df)

propn.gender.dept=BerkByDept[,1,]/apply(BerkByDept,c(1,3),sum)
barplot(propn.gender.dept, ylim=c(0,.85), main="Berkeley admission rate by dept and gender",
        col=c("lightblue", "lightgreen"),
        xlab="Department",legend.text=c("female", "male"),beside=T)
```



Females have higher admission rates in 4 of the 6 departments.