Regression Modelling

Lab 1: Exploring relationships and fitting linear models

Learning Outcomes

After attending this practical, you should

- be able to use `R` to produce scatterplots of report quality;
- be able to use `R` to calculate and perform inference on the correlation coefficient;
- be able to use `R` to do calculations;
- be able to use `R` to fit a linear model.

Introduction

In the lectures, we have examined relationships between variables using scatterplots. We have considered how to fit a linear model, through estimating the parameters such as β in the linear model $Y_i = \alpha + \beta x_i + \varepsilon_i$, by the method of least squares or maximum likelihood. In today's lab we shall use `R`Studio to explore and fit relationships using a variety of linear model examples.

We assume that the data take the form (y_i,x_i) for $i=1,\ldots,n$ and we will model the relationship $Y_i = f(x_i) + \varepsilon_i$.

Getting Started

This course assumes basic knowledge of `R` and `R`Studio. The starting point is opening `R`Studio by going to:

```
Maths & Stats Apps > RStudio
```

Download the data and R file from moodle

Open the R file

```
File > Open File >
```

Then, start by setting the working directory in `R`Studio by typing:

```
Session > Set Working Directory > To Source File Location
```

To run this line command, either: place the cursor on the line and press *Ctrl* and *Enter* or hit the *Run* button in the top right of the script file. This will point RStudio towards this directory to enable us to use files within that directory.

It is useful to annotate your script file using comments so that you know what has been done when you look at it in the future. Comments can be added to a script file by starting the line with a # symbol.

Example 1: Power and Weight

In an experiment conducted by the GU physiology department, a sample of volunteers had their power output measured (in watts) while they ran up stairs as fast as possible under different test conditions. Their gender was noted and their weight and leg length measurements recorded.

The data are available from the csv file **Phys1.csv.** To open this data file in RStudio type:

```
phys <- read.csv("phys1.csv")
```

This will assign the data to the object `phys`.

This worksheet contains six columns, described as follows:

C1	**Gender**
C2	**Weight (kgs.)**
C3	**Leg Length (metres)**
C4	**Power1: Power output in the stair test**
C5	**Power2: Power output in a test with a ramp on the stairs**
C6	**Power3: Power output with ramp on the stairs and a fixed stride length**

This can be seen by typing:

```
names(phys)
```

in the script file. The column names will then appear in the *Console* window below.

A spreadsheet of the data can be viewed by typing:

```
View(phys)
```

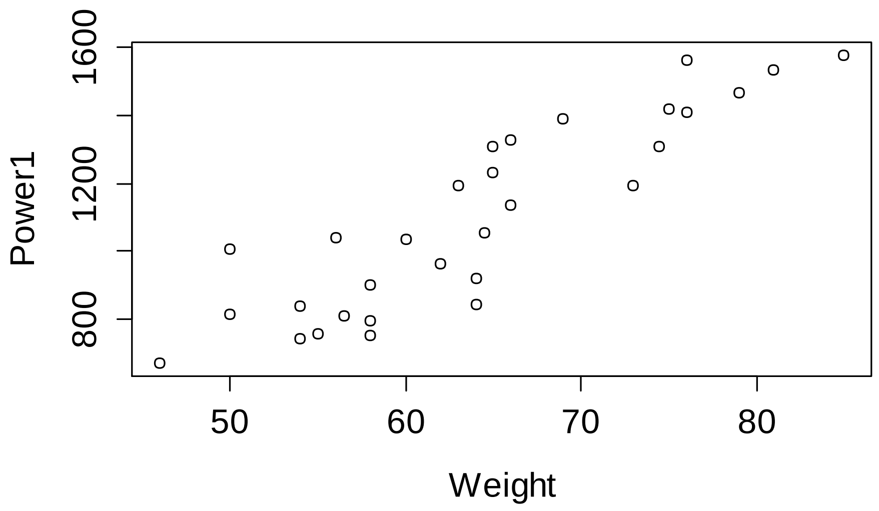
but note that you will need to close this window before trying to excute any further commands. The spreadsheet can also be viewed by clicking on the `phys` object in the *Workspace* (top right of the screen).

Creating Scatterplots

The **relationship between Power1 and Weight** can be examined using a scatterplot.

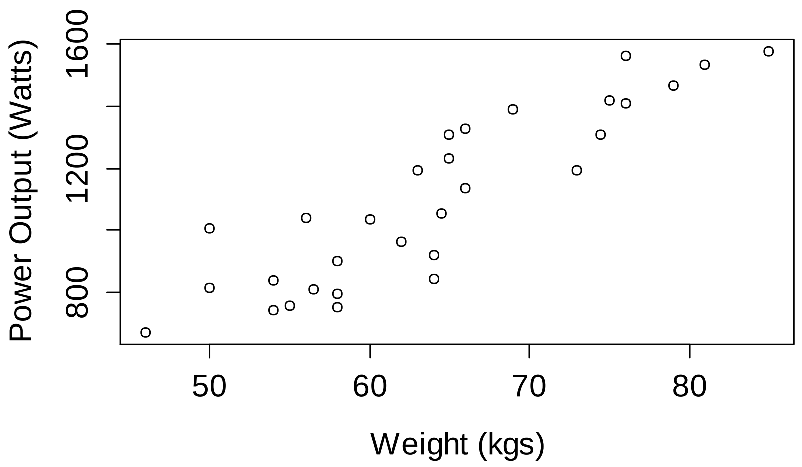
```
plot(Power1~Weight, data=phys)
```

This will produce a basic scatterplot of **Power1** (y-axis) against **Weight** (x-axis).



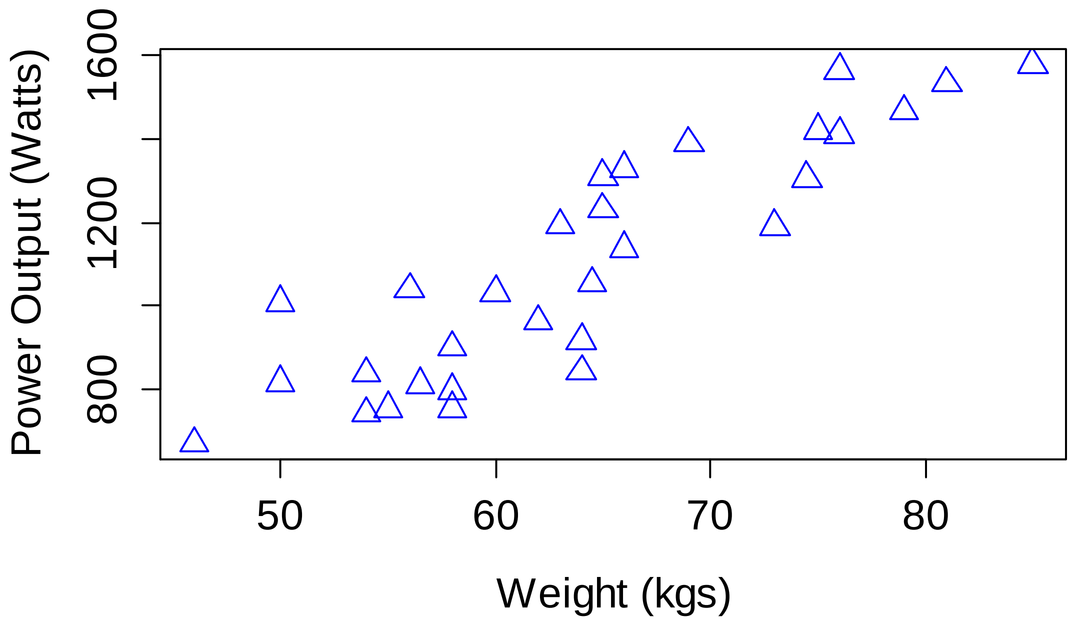
There are several options for annotating plots.
First of all, you should add sensible axis labels to the plot. For example, by typing:

```
plot(Power1~Weight, data=phys, xlab="Weight (kgs)",
          ylab="Power Output (Watts)")
```



Now edit the points by changing the symbol-type, size and colour.

```
plot(Power1~Weight, data=phys, xlab="Weight (kgs)",
ylab="Power Output (Watts)", pch=2, cex=1.5, col=4)
```



Comment on the relationship between power and weight:

Producing labelled scatterplots

We have examined the scatterplot of Power Output in a stair test against Weight, but have ignored the possibility that any relationship between these variables might be different for males and females. So, we now label the plot according to the gender of the subjects. The following provides one approach to doing this in RStudio.

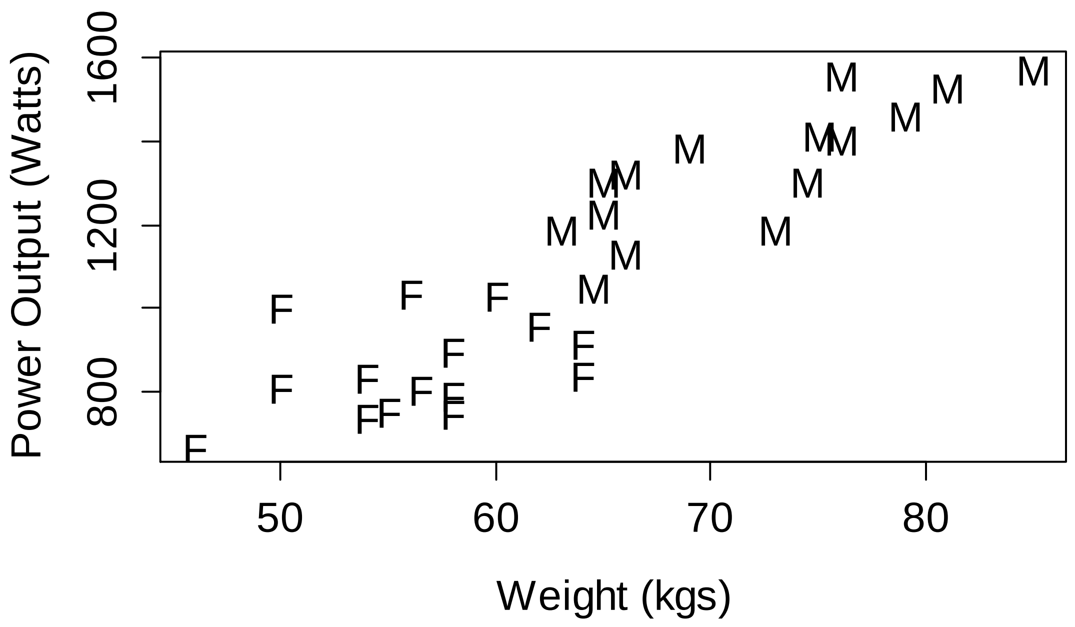
The variable Gender within the data is a factor with two levels: Male and Female. Firstly, we will create a character variable for Gender, by using the command:

```
Gender2 <- as.character(phys$Gender)
```

Note: the $ sign extracts the variable Gender from the dataframe `phys`.

We can now use this new object to label the points by gender. This can be done by typing the following:

```
plot(Power1~Weight, data=phys, xlab="Weight (kgs)",
ylab="Power Output (Watts)", pch=Gender2)
```



Comment on the relationship between power and weight for males and females separately:

Inference for a Population Correlation Coefficient

Having considered plots of Power1 versus Weight for males and females in this sample of data, we now assess the statistical significance of the observed correlations between these variables in the wider populations of (a) males and (b) females.

Firstly, we subset the data for males and females,

One way to do this is:

```
physM <- subset(phys, phys$Gender=="Male")
physF <- subset(phys, phys$Gender=="Female")
```

Test of Zero Population Correlation

We now perform a test of the Null Hypothesis that in the population of males the correlation between Power 1 and Weight is zero against the Alternative Hypothesis that in the population of males the correlation between Power1 and Weight is not equal to zero.

The following command can be used to compute the sample correlation, perform a hypothesis test of the null hypothesis that the population correlation coefficient is zero and provide a confidence interval for the correlation (i.e. a range of plausible values for the true population correlation). The hypothesis test produces a p-value and we reject the null hypothesis for small values of the p-value (typically for p-value < 0.05).

In the following command the $ notation is required to access the variables from the subsetted data. Initially for the males data:

```
    cor.test(physM$Power1, physM$Weight)
```

Now record the sample correlation coefficient and the p-value.

Formal Conclusion:

Now repeat this test using the female data.

Now record the sample correlation coefficient and the p-value.

Formal Conclusion:

Fitting a Linear Model

Example 2: Hubble’s Constant

Source: Hubble, E. (1929) "A Relationship Between Distance and Radial Velocity among Extra-Galactic Nebulae," *Proceedings of the National Academy of Science,* 168.

Context: In 1929 Edwin Hubble investigated the relationship between distance and radial velocity of extra-galactic nebulae (celestial objects). It was hoped that some knowledge of this relationship might give clues as to the way the universe was formed and what may happen later. His findings revolutionized astronomy and are the source of much research today on the ‘Big Bang’. Given here is the data that Hubble used for 24 nebulae. It is of interest to determine the effect of distance on velocity.

Data file: **hubble.csv**

Columns:	**C1:**	**Distance**	**(in Megaparsecs) from Earth**
	C2:	**Velocity**	**the recession velocity (in km/sec)**

Read in the data using:

```
hubble <- read.csv(“hubble.csv”)
```

Exploratory Analysis:

(a) Produce a scatterplot of the data with Velocity on the vertical axis.

(b) Describe the shape of the relationship. Is it linear? Does it seem plausible that it passes through the origin? Do the data fan out?

Fitting a Linear Model:

(c) To find the equation that best describes the relationship between **Velocity** and **Distance** use:

```
lm(Velocity~Distance, data=hubble)
```

This fits a simple linear regression model with the response variable **Velocity** and the explanatory variable **Distance**.

(d) From the output in (c), note down the equation of the fitted line that is given:

Line of best fit: **Velocity =** **+** **distance**

This is the line of best fit, describing the effect of distance on velocity**.**

Plot of the Data with the Fitted Line:

(e) To obtain a plot of the fitted values, we need to store the regression in an object:

```
model1 <- lm(Velocity~Distance, data=hubble)
```

A plot of the data can be re-produced as before with the fitted line added using the `abline` command. This command uses the intercept and slope information from the fitted line in `model1`.

```
plot(Velocity~Distance, data=hubble)
abline(model1)
```

Fitting a Linear Model with no intercept:

(f) Here, it is plausible from the data that the regression line should be forced to go through the origin. A model with no intercept could be fitted using:

```
lm(Velocity~-1+Distance, data=hubble)
```

Note down the equation of the fitted line that is given:

Line of best fit: **Velocity =** **distance**

(g) If we save the intercept only model fit using

```
model.NoInt <- lm(Velocity~-1+Distance, data=hubble)
```

a plot of the data can be re-produced as before with the fitted lines from both linear models added using the `abline` command:

```
plot(Velocity~Distance, data=hubble)
abline(model1, col='red')
abline(model.NoInt, col='blue', lty=2)
```

Comparing the two fitted lines, is the model that passes through the origin plausible?

Example 3: Publishing history: fitting a quadratic

Source: Tweedie, F. J., Bank, D. A. and McIntyre, B. (1998) *Modelling Publishing history, 1475--1640: Change points in the STC*.

Context: The Short Title Catalogue (STC) is a list of all the books that were published in Scotland, England and Ireland between 1475 and 1640. We are interested in finding out if there are any changes in the number of books published between 1500 and 1640.

Data file: **books.csv**

Columns: **C1: Year (Year – 1500)**
C2: Total number of books published

Read in the data using:

```
books <- read.csv("books.csv")
```

Exploratory analysis:

(a) Produce a scatterplot of the data with Number of books on the vertical axis and Year on the horizontal axis.

(b) Describe the shape of the relationship. Does it seem linear?

Fitting a Linear Model:

(c) Use the commands below to fit a quadratic regression:

```
lm(Number.of.Books~Year+I(Year^2), data=books)
```

Note: `Year^2` needs to be 'protected' by `I(…)`, the identity function.

(d) Note down the equation of the quadratic line and plot the fitted line.

Quadratic equation of best fit:
total number of books = + year + year^2

Plotting the fitted line:

```
model2 <- lm(Number.of.Books~Year+I(Year^2), data=books)

plot(Number.of.Books~Year, data=books)
lines(fitted(model2))
```

The `lines` command is required here to plot the quadratic regression.

Example 4: The Taste of Cheese

Data: $(y_i, x_i) \quad i = 1,\ldots,30$

Model: $E(Y_i) = \alpha + \beta x_i, \quad \mathrm{Var}(Y_i) = \sigma^2 \cdot$

Context: This model might be considered for an experiment involving the chemical constituents of cheese and its taste. The dataset (cheese.MTW) contains concentrations of acetic acid, hydrogen sulphide (H_2S) and lactic acid and a subjective taste score. It is of interest to investigate the effects of the different acids on the taste score.

Data file: **cheese.csv**

Columns:	**C1: Case**	**Number of the sample**
	C2: Taste	**A taste score**
	C3: Acetic Acid	**Concentration**
	C4: H2S	**Concentration**
	C5: Lactic Acid	**Concentration**

Read in the data using:

```
cheese <- read.csv("cheese.csv")
```

(a) Produce scatterplots of Taste score (*y*) against Lactic Acid (*x*) and Taste (*y*) score against H_2S (*x*).

(b) Now plot taste score against $\log_e(H_2S)$ and against $\log_e$(Lactic Acid). The command to take a natural log transform is e.g. `log(H2S)`.

(c) Which of the 4 variables (H2S, $\log_e$(H2S), Lactic Acid, $\log_e$(Lactic Acid)) seems best for describing a linear relationship with Taste?

(d) Fit a linear regression (using the `lm` command) with Taste as the Response variable and the explanatory variable you choose at the end of (c). Make a note of the fitted model.

(e) Produce a plot with a line from your fitted model in (d) using the command `abline()`.

(f) How well do you think the model and the data agree?

(g Is the population correlation coefficient between the variable Taste and your chosen variable significantly different from zero?

(i.e. use `cor.test(cheese$Taste, cheese$?)` where ? is your chosen variable)