Data Handling and Linear Model fitting in **R**

1 Introduction

You should, by now, be reasonably familiar with the use of R for entering data and performing basic statistical analyses. So far you have carried out some basic data exploration using summary statistics and plots, and carried out t-tests, constructed confidence intervals for means (or differences between means) and performed an F-test for the equality of variances between two populations.

You should be used to undertaking your analysis in R by the running of *commands* in a **script file**, and this is the way you should continue to use the software in connection with the STATISTICAL ANALYSIS course.

2 Data Handling in R

2.1 Working with Data Frames

In R, data matrices are usually stored in the form of *data frames*. A data frame is a generalization of a matrix in that the different column vectors can have different types of data - character values, numerical data or factors - whereas a matrix is restricted to one type of data only.

In many of the examples that will be presented to you, the data will be grouped into data frames. This is not always necessary. However, in general, data frames can provide a useful means of displaying variables, or simply of storing variable objects efficiently for inclusion in an analysis.

- The individual variables within the frame can be made available from an (imported) data frame, using `attach(data.frame)`, which adds the data frame to the search path, and similarly 'hidden' using `detach()`. For example, returning to the grapefruit example of Exercises 1:

```
> grapefruit.df<-data.frame(shaded,exposed)
> rm(shaded,exposed)
> attach(grapefruit.df)
> shaded
```

- Alternatively, variables within a data frame can be accessed by specifying, for example

```
> grapefruit.df$shaded
```

- New columns (*variables*) can also be added to existing data frames. e.g.

```
> diff<-shaded-exposed
> grapefruit.df<-data.frame(grapefruit.df,diff)
> rm(diff)
```

Warning: when adding variables to an existing data frame which is attached to the search path, it may be necessary to detach and reattach in order for your new variables to be accessible.

```
> detach( )
> attach(grapefruit.df)
```

Note: `detach()` detaches the last data frame to be attached. An alternative is to use `detach("grapefruit.df")`.

- Data frames can be saved as (`.Rda`) files which can be opened and attached if needed in future using `load`:

```
> save(grapefruit.df,file="data.Rda")
> load("data.Rda")
```

2.2 Reading data from files

To input data from external files use the function `read.table()`, `read.fwf()` or `scan()`. Note that the external files must be in ASCII format so that if created in Word they need to be saved as an ASCII file. Word refers to ASCII files as 'text files', and forces you to save them with a `.txt` extension. The use of Word to prepare files for other applications is *not* recommended. As a general rule, you should use a good editor to produce ASCII files (e.g. `WordPad` and `Notepad`).

Example
Use WordPad to create a text file in your directory, called `somedata.txt` and containing the following (spacing is unimportant):

```
2  100.0
3  44.5
```

Now, in R you can read the data as follows:

```
> x <- read.table("somedata.txt", header=FALSE)
> x
  V1    V2
1  2 100.0
2  3  44.5
```

3 **Linear Modelling in R:** An introductory Example

In Lecture 2 you saw how parameter estimates are calculated for the multiple regression model. The following example will give you some first-hand experience of fitting a *linear model object*, `lm`, in R. [This example will be referred to in Lectures over the coming weeks].

Example: oil

An estimate is required of the percentage yield of petroleum spirit from crude oil, based upon certain rough laboratory determinations of properties of the crude oil. The table on the following page shows actual percentage yields of petroleum spirit, y, and four properties, x_1, x_2, x_3, x_4, of the crude oil, for samples from 32 different crudes.

The variables recorded are as follows:

y:	percentage yield of petroleum spirit
x_1:	specific gravity of the crude
x_2:	crude oil vapour pressure, measured in pounds per square inch
x_3:	the ASTM 10% distillation point, in °F
x_4:	the petroleum fraction end point, in °F

Over the coming weeks we will investigate a model for the response yield of petroleum spirit `spirit`, using subsets of the four explanatory variables `gravity`, `pressure`, `distil` and `endpoint`. We will, however, begin our exploration of linear modelling in R with reference to this example, so that you can immediately begin to explore some of the modelling capabilities of this package.

The data is available as a text file `oil.txt` on Moodle

`https://moodle.bbk.ac.uk/`

which you can download to your working directory and import using the `read.table()` function described earlier. For example,

```
> oil <- read.table("oil.txt", header=FALSE)
> names(oil) <- c("spirit","gravity","pressure","distil","endpoint")
```

• The `names()` function adds variable names to the data frame which were not included in the text file.

Data on yields of petroleum spirit

y	x_1	x_2	x_3	x_4
6.9	38.4	6.1	220	235
14.4	40.3	4.8	231	307
7.4	40.0	6.1	217	212
8.5	31.8	0.2	316	365
8.0	40.8	3.5	210	218
2.8	41.3	1.8	267	235
5.0	38.1	1.2	274	285
12.2	50.8	8.6	190	205
10.0	32.2	5.2	236	267
15.2	38.4	6.1	220	300
26.8	40.3	4.8	231	367
14.0	32.2	2.4	284	351
14.7	31.8	0.2	316	379
6.4	41.3	1.8	267	275
17.6	38.1	1.2	274	365
22.3	50.8	8.6	190	275
24.8	32.2	5.2	236	360
26.0	38.4	6.1	220	365
34.9	40.3	4.8	231	395
18.2	40.0	6.1	217	272
23.2	32.2	2.4	284	424
18.0	31.8	0.2	316	428
13.1	40.8	3.5	210	273
16.1	41.3	1.8	267	358
32.1	38.1	1.2	274	444
34.7	50.8	8.6	190	345
31.7	32.2	5.2	236	402
33.6	38.4	6.1	220	410
30.4	40.0	6.1	217	340
26.6	40.8	3.5	210	347
27.8	41.3	1.8	267	416
45.7	50.8	8.6	190	407

We begin, by examining the *full* regression on all explanatory variables. A linear model is fitted in R using the `lm( )` function:

```
oil.lm <- lm(spirit ~ gravity + pressure + distil + endpoint, data = oil)
```

The *model formula* `spirit ~ gravity + pressure + distil + endpoint` specifies the form of model to be fitted, i.e. `spirit` is to be modelled as a linear combination of the four explanatory variables. The model includes an intercept by default, which can be removed (if required) by adding `-1` to the formula. The inclusion of `data=oil` in the model call, tells R where the variables can be found, alternatively the data frame can be attached to the search path.

A *summary* of the fitted model may be displayed using

```
> summary(oil.lm)
Call:
lm(formula = spirit ~ gravity + pressure + distil + endpoint,
    data = oil)

Residuals:
    Min      1Q  Median      3Q     Max
-3.5804 -1.5223 -0.1098  1.4237  4.6214

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -6.820774  10.123152  -0.674   0.5062
gravity      0.227246   0.099937   2.274   0.0311 *
pressure     0.553726   0.369752   1.498   0.1458
distil      -0.149536   0.029229  -5.116 2.23e-05 ***
endpoint     0.154650   0.006446  23.992  < 2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 2.234 on 27 degrees of freedom
Multiple R-squared:  0.9622,Adjusted R-squared:  0.9566
F-statistic: 171.7 on 4 and 27 DF,  p-value: < 2.2e-16
```

This results in the default output above - which will be discussed in greater detail in Lecture 3. Take some time to examine what information this tells you about the model based on the information covered in lectures so far. In particular, do you recall how the figures shown in the `Estimate` and `Std. Error` columns for the coefficients have been obtained? What is the fitted regression model?

3.1 Simple Linear Regression

It is instructive to look more closely at the case of simple linear regression (on one explanatory variable) to learn more about the `lm` object.

Consider a simple linear regression of `spirit` in the oil example, on say `endpoint`.

The relationship between these variables can be explored using a simple scatterplot (shown later):

```
> plot(oil$endpoint, oil$spirit, main ="spirit v's endpoint")
```

Our aim is to calculate the fitted regression line between these variables and add it to the plot of the data.

We begin by fitting the model.

```
> oil1.lm <- lm(spirit ~ endpoint, data = oil)
> summary(oil1.lm)
```

```
Call:
lm(formula = spirit ~ endpoint, data = oil)

Residuals:
     Min       1Q   Median       3Q      Max
-14.7584  -6.2783   0.0525   5.1624  17.8481

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -16.66206    6.68721  -2.492   0.0185 *
endpoint      0.10937    0.01972   5.546 4.98e-06 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 7.659 on 30 degrees of freedom
Multiple R-squared:  0.5063,Adjusted R-squared:  0.4898
F-statistic: 30.76 on 1 and 30 DF,  p-value: 4.983e-06
```

We would like to be able to extract the estimated intercept $\hat{\alpha} = b_0$ and slope $\hat{\beta} = b_1$ parameters from the model object. We can see what *terms* are available to us in the stored object via

```
> attributes(oil1.lm)
$names
 [1] "coefficients"  "residuals"     "effects"       "rank"          "fitted.values"
 [6] "assign"        "qr"            "df.residual"   "xlevels"       "call"
[11] "terms"         "model"

$class
[1] "lm"
```

so that

```
> oil1.lm$coefficients     # alternatively >coef(oil1.lm)
(Intercept)    endpoint
 -16.662057    0.109371
```

and we can therefore extract these values (by assigning them to new *objects* `a` and `b` say)

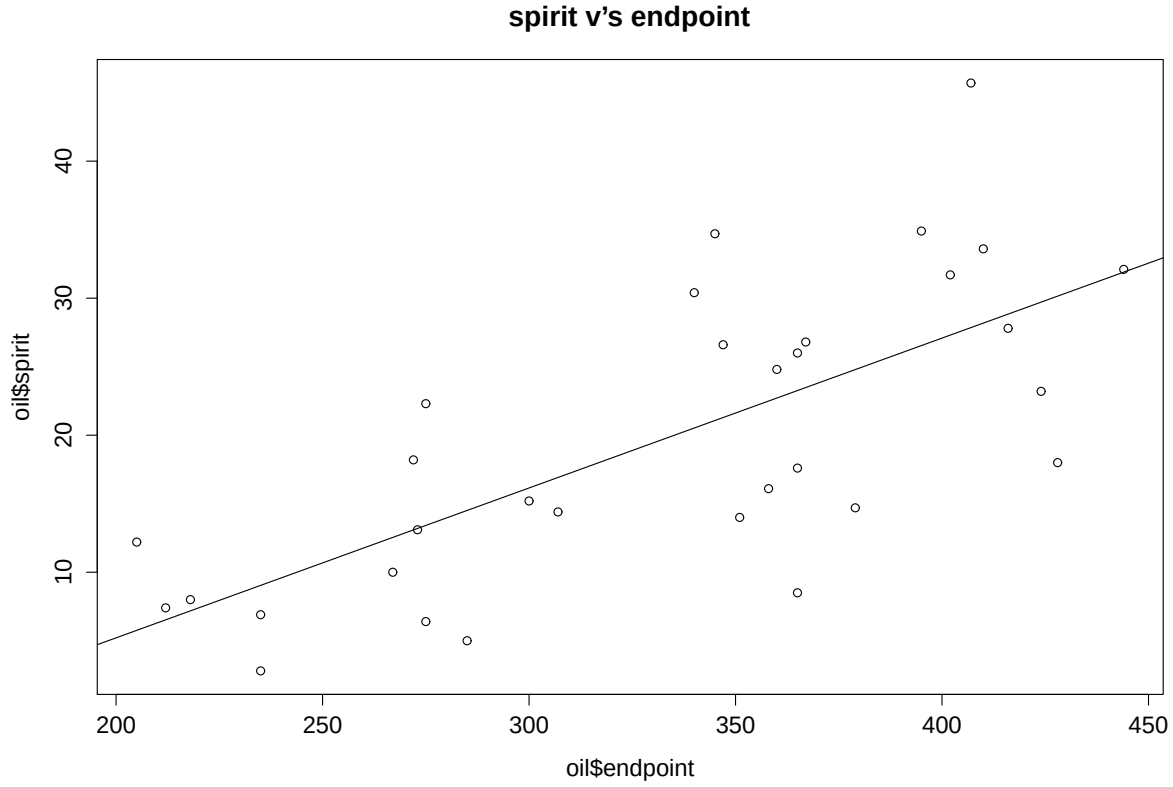
```
> a <- coef(oil1.lm)[1]
> a
(Intercept)
  -16.66206
> b <- coef(oil1.lm)[2]
> b
endpoint
0.109371
```

[You will see later that vectors of residuals, fitted values, and so on can be accessed similarly].

To add the straight line to the plot we can use the function `abline(a,b)` where `a` and `b` are the intercept and slope parameters respectively, we use the following commands

```
> plot(oil$endpoint, oil$spirit, main = "spirit v's endpoint")
> abline(a,b)
```

This results in the plot shown below.



Alternatively, the following code can be used to add the line

```
> x <- seq(200, 450, 0.1)
> y <- a + b * x
> lines(x, y)
```

where the `lines()` function adds a line through the given `(x,y)` values to the *current* plot.

Exercise:

• Try fitting separate linear regressions of `spirit` on each of the remaining explanatory variables `gravity`, `pressure` and `distil`.

• Which of the four explanatory variables appears to have the strongest linear relationship individually with `spirit`?