First steps with R

1 Getting started

1.1 Installing the program

All of the data sets we shall use, including those which appear in the lectures and problem sheets, are supplied as text files and will carry the extension `.txt`. To set things up, I suggest that you create a new directory on your hard disk into which you can download your data files: for the sake of the rest of this introduction to the package, I shall assume that you will be addressing `c:\data`.

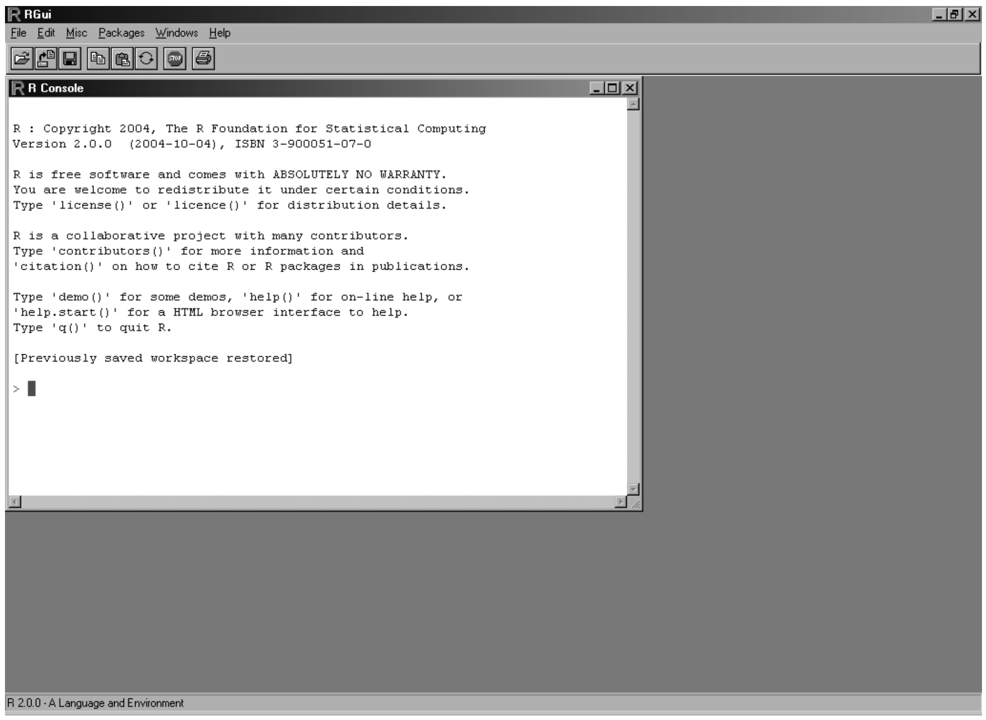
First and foremost you need to get the free package. Go to

`www.stats.bris.ac.uk/R/`

and download by following the sequence

Windows (95 and later) → base → R-2.4.1-win32.exe

Once you have done this you can run `R-2.4.1-win32.exe` which will install the program and place an icon on your desktop. Now start up the program and you should see the following screen.


RGui
File Edit Misc Packages Windows Help
R Console
R 2.0.0 - A Language and Environment

```
R : Copyright 2004, The R Foundation for Statistical Computing
Version 2.0.0  (2004-10-04), ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for a HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

>
```

You can quit the program at any time by clicking on File Exit at the top left of the screen.

1.2 R commands

It is important to realise that R is case sensitive so that, for example, `A` and `a` would be regraded as different symbols. Care is therefore needed when typing in commands. Individual commands may be separated either by a semi-colon (;) or by a new line (i.e.by hitting <return>). Comments can be put in anywhere by starting with a hashmark (#); everything to the end of the line will then be a comment.

If a command is not complete by the end of the line (i.e. when you hit <return>), the prompt at the next line will be

```
+
```

and will continue on futher, subsequent lines ntil the command is syntactically complete. This is very handy because it means that you can enter a very long command without having to run over the screen width by taking several lines. It also means that, if you fail to enter, say, a closing bracket, it will simply keep prompting you until you do enter it.

Command lines can be recalled and edited by using the up and down arrow keys to scroll through them. Once a command is located in this way, the horizontal arrow keys can be used to edit it (<del> is used to delete and the other keys are used to add in text) and the <return> key to execute it.
An expression is evaluated, printed and then discarded.

1.3 Vectors and assignment

R works on data structures which are identified by having names, the simplest such structure being a data vector (i.e. an ordered collection of numbers). Suppose you want to set up a vector `x` comprising the numbers 3.2, 5.1, 1.4, 2.3, 6.8, 19.7. Simply type in

```
> x <- c(3.2, 5.1, 1.4, 2.3, 6.8, 19.7)
```

This *assigns* the vector to `x` using the *function* `c()`. Typing in the assignment operator, which is the two characters `<-` (i.e. "less than" followed by "hyphen") causes the vector to be received by `x`, and the operator can be viewed as an arrow which sends it there. The arrow can point either way, so it does not matter which way round you make this assignment. Entering

```
> c(3.2, 5.1, 1.4, 2.3, 6.8, 19.7) -> x
```

has exactly the same result. Try doing this and then enter

```
> x
```

The result will be that x will be printed on screen. If you were to enter

```
> 1/x
```

the reciprocals of the six values will be printed to screen. Note that these are shown on-screen and then lost. If you want to assign the reciprocals to, say, `y`, you enter

```
> y <- 1/x
```

The `c()` function is used for concatenating, so entering

```
> z <- c(x, 0, -x)
```

produces a vector with 13 entries consisting of the six values, a zero in the middle, and the six values made negative.

Vectors can also be used in doing arithmetic. If you enter

```
> 2 + 3
   [1]  5
```

you can see that the numbers are added. If you enter vectors

```
> x + y
   [1] 3.512500 5.296078 2.114286 2.734783 6.947059 19.750761
```

you get the vectors added element by element.

```
> z <- 2*x + y + 1
```

will over-write the `z` you created earlier with a new vector with the i^{th} element being $2x_i + y_i + 1$.

2 Data files

All of the data sets you will be given in this course will be text files in tabular form which can be inspected by using any text editor (e.g. Notepad, Textpad or MSWord). They can be loaded into R with the command `read.table`. For example, suppose you have a file called *moses.txt* in a directory called c:\data. You can assign this to, say, Moses (or any other name you fancy) by entering

```
> Moses <- read.table("c:/data/moses.txt", header = TRUE)
```

The qualifier "`header = TRUE`" is there to tell R that the first line of the file is to be read as a header for each column. In this particular file there is only one column. If you now enter

```
> Moses
```

you will see the file printed on-screen. It comprises a single column containing the winning times of the athlete Ed Moses throughout his career. *E.g. adding two numbers*

```
  > 2 + 3
   [1]  5
```

E.g. calculating the mean
The object Ed.Moses is

```
  > mean(Ed.Moses)
   [1]  48.52844
```

Assignments are performed by using the assignment operator <-.

An assignment evaluates the expression and gives the value to a variable.

The result is not printed automatically.

E.g.

```
> m <- mean(Ed.Moses);
    v <- var(Ed.Moses)
> m/sqrt(v)
[1]  66.31589
```

New functions can be written and used in the same way as existing ones.

E.g. p-value for a t-test

```
> st.dev <- function(x)sqrt(var(x))
> t.test.p <- function(x, mu=0){
    n <- length(x)
    t <- sqrt(n)*(mean(x) - mu)/st.dev(x)
    2*(1 - pt(abs(t), n - 1)
    }
```

For Ed Moses' times, test the hypothesis $\mu = 48.25$.

```
> t.test.p(Ed.Moses, 48.25)
[1]  0.00005074456
```

Suppose you need some guidance on the syntax for a particular R function. There are two ways in which you can do this. Suppose, for example, you want help with the variance function. In the command window, enter

```
> help(var)
```

or

```
> ?var
```

which will cause a window to open with the help you need.

If you need help with some particular statistical procedure for which you do not know the name of the function, you can also use `help.search`. For example, suppose you need to find out the R function that will do a normal QQ plot for you. Enter

```
> help.search("QQ plot")
```

That would tell you that `qqnorm` might be the function you are after, so you follow that with

```
> ?qqnorm
```

and you will find what you need.

The best way to learn the package is to use it. Therefore let us switch on the computer and try working through the exercise sheet which follows.

Introductory R session

This session is designed to make you familiar with the R syntax and graphics. Some of the expressions which appear below will not be familiar: this is deliberate and you should access the Help facility to find out what they do and how they are used. In the four exercises which follow, the left column contains the R code and the right column explains what you are trying to achieve.

The extra data sets you will need are on the website. They are the text files `Anorexia.txt`, `Hills.txt`, `Moses.txt` and you will have to import them into R.

Remember, if you want to stop any R routine while it is running, just press the <Esc> key.

1.

`attach(Moses)`	Make the columns in the data frame visible as variables.
`hist(Ed.Moses)`	Histogram of Ed Moses' times.
`boxplot(Ed.Moses)`	Boxplot of Ed Moses' times.
`x <- seq(1,length(Ed.Moses))`	Creates a vector of integers. Since the times are in chronological order, the scatterplot shows how his times varied throughout his career. How did his performance vary as he became older?
`plot(x,Ed.Moses)`	

2.

`x <- rnorm(10000)`
`y <- rnorm(10000)`
	Generate two pseudo random normal vectors with 10,000 entries and assign them to `x` and `y`.

`hist(c(x,y+2),nbins=25)`	Histogram of a mixture of normal distributions. Experiment with the number of bins (25) and the shift (2) of the second component.
`rm(x,y)`	Remove objects no longer needed. (The clean up phase.)

3. Now we look at another real data set on record times of Scottish hill races against distance and total height climbed.

`Hills`	List the data (don't forget to read it in first and assign it to Hills)
`attach(Hills)`	Make columns available by name.
`plot(Dist,Time)`	Scatterplot of data.
`identify(Dist,Time,row.names(Hills))`	Use the left-hand mouse button to identify outlying points. Their row numbers are returned. Use the right-hand mouse button to quit.
`abline(lm(Time ~Dist))`	Show the least squares regression line.
`detach()`	Clean up again.

4. Now, in your own time, look at the data file `Anorexia.txt`. It contains data on weights, in kg, of young girls receiving three different kinds of anorexia treatment over a fixed period of time. The control group received the standard treatment (STD), one group received cognitive behavioural treatment (CBT) and the third received family therapy (FT). Weights were measured before and after treatment, so that STDb and STDa refer to the respective before and after measurements for the standard treatment, and so on. The objective is to compare the three treatments.

   (i) Look at the three scatterplots of *after/before*. What preliminary conclusions can you suggest?

   (ii) Construct three vectors of *before* - *after* differences. Produce comparative boxplots.

   (iii) Use t-tests to assess whether or not each treatment produces a significant difference. Is this a sensible way to proceed?

   (iv) Write a function which will use a pooled variance estimate to carry out t-tests for comparison of the control group with the other treatments. Produce a normal scores plot to investigate the residuals for normality. What conclusions can you draw?