Lab 1

What is R?

R is a programming language for statistical data manipulation and analysis. R is very popular both for being a free software and because many are people contribute to it through a package system that distributes code and data. The statistical applications of R are constantly increasing because of the contributed packages. It is also well documented and the documentation is available either from the program help menu or from the website. It compiles and runs on a wide variety of operating systems: Windows, MacOS and Unix, and it is easy to download.

How to download R

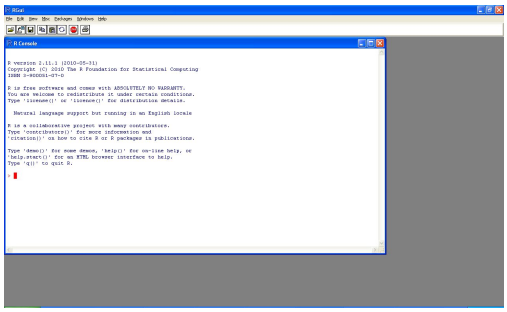
The software and the additional packages can be downloaded from the R project website. Go online to **http://www.r-project.org/index.html**, select **download R** and choose your preferred CRAN mirror, e.g. the Irish one at $HEAnet, Dublin$ by selecting the corresponding link **http://ftp.heanet.ie/mirrors/cran.r-project.org**. In the *Download and Install R* box select the operation system, e.g. Windows and from the *Subdirectories* list select *base* then click on the *Download R* 2.11.1 $for\ Windows$ link to start downloading R. Press *Run* to install the $R-2.11.1-win32.exe$ application and then follow the instruction to finalize the installation.

How to open R

Once R is downloaded, open it by double-clicking on the icon created at installation on the Desktop or double-clicking any file with an $.RData$ extension which is associated with R.

R has no Graphical User Interface (GUI) for statistics and it runs in terminal mode, making it a little bit less user friendly than the statistical software based on spreadsheets (Excel,Minitab, SPSS). When opening it you should be able to see the following screen in which code can be written in the

command line:



The details of the *R* session are presented below.

```
R version 2.11.1 (2010-05-31)

Copyright (C) 2010 The R Foundation for Statistical Computing

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.


Natural language support but running in an English locale


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 an HTML browser interface to help.

Type 'q()' to quit R.

>
```

To input commands in R just type them in the prompter line after the > sign. The R window acts as an output (results) window as well. Typing a command or instruction at the prompt will cause it to be executed immediately. For example if you type in the command line

$> 3 + 7$

press $Enter$ and R will automatically display the result in the main console window

[1] 10

Along the top of the window is a set of drop-down menus(File, Edit, View, Misc, Packages, Windows, Help), which can be used for various tasks including opening, loading and saving script windows, loading and saving your workspace and installing packages.

To exit, type $q()$. You are then prompted whether you want to save your working session. Answering either Yes or No will lead you out. For the beginning answer No.

Functions in R

All commands in R are regarded as functions and they operate on arguments such as the arithmetic functions $mean(x)$, $median(x)$, $sum(x)$, $prod(x)$, $length(x)$, $sqrt(y)$, $log(y)$, where x is a numeric vector and y is a numeric scalar.

To evaluate a function, just type its name followed by the list of arguments between brackets. For example, $sqrt$ is the name of the function that calculates square root of a number. The command

> sqrt(2) returns

[1] 1.414214

Other functions can return a whole set of numerical values, for example $rnorm$ generates a random sample from the normal distribution. The command $rnorm(n = 10, mean = 0, sd = 1)$ will generate a sample of size $n = 10$ from a normal distribution with $mean$=0 and standard deviation of 1. If you type

$> rnorm(n = 10, mean = 0, sd = 1)$

or omit the argument names and input their values only

$> rnorm(10, 0, 1)$

R will generate 10 random values from the $N(0, 1)$ distribution

[1] -0.53105620 0.32813459 0.51974048 -0.02019856 1.04873796 0.81025615

[7] 1.91691165 -1.18911755 -1.08723748 1.31396768

>

The results above were displayed immediately. The results of a function can be assigned, invisibly, to a variable using the " $<-$" symbol

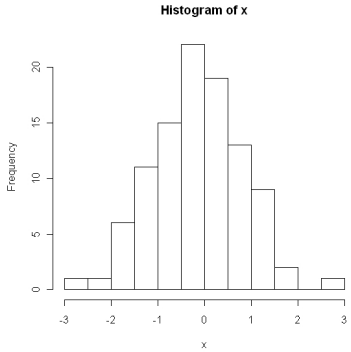
$> x <- rnorm(100, 0, 1)$

which creates a vector x of 100 normal standard random variables. Assignments do not automatically display anything, therefore to see what is stored in the new variable type its name in the prompter line and press *Enter*.

$> x$

This will display the values of all the 100 sampled values. To obtain a graphical summary of the random sample use the *hist* function

$> hist(x)$



Variables can also be assigned numerical values directly using the format

$> \ a <- 3$ to set a to the value 3.

Attention must be paid to typing as R is case sensitive so A and a are different objects. In the example bellow the value 5 is assigned to the small a variable and the value of 10 is assigned to the capital A variable.

$> a <- 5$

$> A <- 10$

$> a + A$

[1] 15

A few variables names should be avoided as they are reserved for R functions, for example t, c, q, F, T, df.

R objects

R works by creating different objects and using various function calls that create and use those objects. The most used objects in this course are the vectors, matrices and data frames.

Vectors are the simplest type of object in R. The vectors can be numeric, character or logical depending on the nature of the values it consists of. To create a numeric vector x consisting of 5 numbers, 10, 11, 7, 8, 9, use the **c** command

$> \ x \ <- c(10, 11, 7, 8, 9)$

To print the contents of x:

$> x$

[1] 10, 11, 7, 8, 9

The [1] in front of the result is the index of the first element in the vector x. To access an individual element of a vector such as the 3-rd element of x use its index between square brakets

$> \ x[3]$

[1] 7

Various commands can be performed on vectors such as $sum(x)$ or $mean(x)$.

Matrices are two-dimensional arrays of numbers and in R are defined using the $matrix$ command.

For example $M <- matrix(c(1,2,3,4,5,6), nrow = 2, ncol = 3)$ creates the matrix

$$M = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

By default, R will create the matrix by columns, i.e. one column at a time. If you add the argument $byrow = TRUE$ or $byrow = T$ the matrix is filled in by rows, i.e. one row at a time.

$N <- matrix(c(1,2,3,4,5,6), nrow = 2, ncol = 3, byrow = T)$ creates the matrix

$$N = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

Elements of matrices are individually extracted using square brackets.

$> \ N[2,1]$

[1] 4

selects the element in the 2-nd row, 1-st column.

To see the names of all the variables you have created so far in your session, type ls().

Saving the commands in R

R maintains a record of command history so that we can scroll through previous commands using ↑ and ↓ keys. It may sometimes be useful to save commands for future use. We can save the command history at any stage using $> savehistory("filename.Rhistory")$

The commands entered so far will be saved in a plain text format. In order to restore saved commands use $> \ loadhistory("filename.Rhistory")$.