MAST7866 — Foundations of Data Science

Computing Session 1 — Getting started with R and RStudio

In this module we will introduce a computing language called R. We use an interface to R called RStudio which provides a more user-friendly setting in which to code in. This worksheet will guide you through the installation or R and RStudio on your own computers, will explain how to access RStudio on the University computers and will introduce you to some basic aspects of using R. Work through each of the tasks below and answer the questions set.

Task 1a – Installing R and RStudio and your own computer

The first thing you will need to do is install R and RStudio on your own computer. Instructions of how to do this for both Windows and Mac computers are given as pdf files on moodle.

or Task 1b – Accessing R and RStudio from the University computers

Once you have logged onto a University PC, type RStudio in the Search box at the bottom left-hand corner of the desktop and then open it.

Task 2 – Navigating RStudio

Watch the video "Introduction to RStudio". This video explains the basic structure of the 4 windows within RStudio.

Task 3 – Setting working directories and running/writing scripts in R

Watch the video "RScripts" on moodle.

Create a folder for your R session and set your working directory to this folder.

Task 4 – Reading data into R

When you have a small amount of data you can type data directly into R. The c( ) function in R is used to create a vector with values you provide explicitly. A vector is simply a sequence of data elements of the same basic type.

The > is called the prompt in R. In what follows in the instructions below *you do not type it into R*, but we are using it to indicate where we are providing R code.

For example, suppose we wish to define a variable X to be a vector of numbers 1, 5, 6 and 10. To do this in R we would type:

```
> X <- c(1, 5, 6, 10)
```

These 4 values are then stored in an object named X and you can call them by typing:

```
> X
[1]  1  5  6 10
```

The command `<-` means "is assigned", `x<-10` is equivalent of saying variable x takes the value `10`.

To call one of the elements of X we can do this using square brackets, [ ]. For example,

```
> X[2]
[1] 5
```

calls the 2nd element of X, which is the value 5.

Download the R script TestScript.R from moodle and save it in your working directory.

Open the script file in RStudio and practice running the script. What does this R code do?

Data sets can get very large and so it is useful to be able to load data in directly from other file types such as text, excel or csv files.

Watch the video "Loading data into R" on moodle.

There are some sample data sets on moodle: birdexample.xls, Temperature.xls and parkrun.txt. Download these files into your working directory.

Load parkrun.txt into R using the technique demonstrated in the video. The data set is now stored in an object called parkrun. parkrun contains 3 variables – Run_Number, Male and Female. Suppose we wish to define a vector, which we shall call PRmale, of the variable Male. We can do this by using the command:

```
> PRmale<-parkrun$Male
```

Alternatively suppose we wish to be able to just call the variables Run_Number, Male and Female, we can do this by attaching parkrun to our environment using the command

```
> attach(parkrun)
```

Now you can simply call:

```
> Male
```

Try loading the other two data sets, birdexample.xls and Temperature.xlsx into R and check that the variables have got appropriate names and that they have appeared in your R environment. Test calling the variables using the `$` option and using the `attach` command.

Task 5 – Installing R Packages

R packages are collections of functions and data sets developed by the R community. They provide additional functionality that base R does not provide.

Watch the video "Installing and using R packages" on moodle.

You will have plenty of practice at installing and using R packages in future worksheets!

Challenge 1

Use the function `c()` to define the vector `x = (12, 3, 2, 6).`

Enter the following function, what does it do?

```
> rep(x,2)
```

Enter the following function, what does it do?

```
> rep(x,c(2,2,2,2))
```

What command would we use to get the following output?

```
   [1] 12 3 3 2 2 6 6 6
```

Challenge 2

Recall that a vector is an ordered set of values and the simplest way to define a vector is by the `c()` command. Define the vectors x and y:

```
> x<-c(1,2,3)

> y<-c(0,1,2,3)
```

To add 1 to each element of x we type:

```
> x+1
```

How would we obtain the following output?

```
[1] 2 3 4 5
```

What do the following commands do?

```
> v <-c(3,6,9,6)

> 1/v

> max(v)
> min(v)

> sum(v)

> prod(v)

> sort(v)

> order(v)
```