Creating Vectors and working with them

20160608:15.33

This module is designed to enable you to:

- Create, write, name and save R scripts
- Understand when and how to use the colon operator, and 'combine' **c()** and 'sequence' **seq()** functions
- Understand what is a **vector** and how **vectors** can be used
- Know how to find 'help' using the various inbuilt R and RStudio mechanisms
- Understand 'command completion' and how to use it
- Have a basic understanding of how the data in a vector can be viewed

Preamble

- It is suggested that you use <ALT> + <TAB> to alternate between this Workbook module and RStudio.
- In this module where a series of commands open drop down menus or dialogue boxes we will provide the sequence to follow separated by the pipe "|" character (e.g. **Files |New File | R Script** to open up a new R script)
- This module assumes that you have a basic familiarity with the four panel layout of RStudio. If that is not the case then we recommend that you first study the module **"Introducing RStudio"**

Initial setup

- Click on **Files** tab and navigate to the **RStud-master** folder
- If you do not have a folder of that name then we would recommend that you either create a folder of that name (by clicking on 'New Folder') or select another folder to be your Working Directory
- Open this and set it as your Working Directory (hint – click on **More**)

Starting and writing a Script

First we will create a script which we will then use to carry out a number of exercises. At the top left of the screen click on **Files |New File | R Script** (note that the short cut **<CTRL> + <SHIFT> + N** can also be used).
You should now have a blank script with a name similar to 'Untitled1'

In the margin of the script you will see the line number 1. Try entering "# My workscript" and then <ENTER>. Note:

1. The name of the script has turned red. The filename will always turn red if you make any change to the script
2. The cursor has moved on to line 2
3. Your first line started with a "#" and so will be treated by R as a comment

In the second line of the script try assigning the value 70 to a variable called "Weight" (click here for help)

Run this line and note that a new object "Weight" has appeared in the Environment panel. This is a simple scalar variable of type number

To save the script either click on the blue floppy disk icon, or press **<CTRL> + S**, or do **File | Save**. The first time that you try to save a new script these will all result in a dialogue box expecting you to enter a filename of your choice. Enter "MyWorkScript" or any preferred name and click **Save**. Note that:

- The Script filename has now changed to the name that you entered and is no longer red
- A file called MyWorkScript.R, or with the name that you selected, has appeared in your Files panel

Vectors and the use of the colon operator and functions c and seq

In R a vector is a simple kind of data object which consists of a sequence of data elements of the same type. The individual data elements are known as the 'components' of a vector.

The variable "Weight" so far has only one component. It is in fact a **Vector** even though this is not obvious. This will become obvious when we assign a sequence of components to it. We can do this in a number of ways. Below we

will demonstrate three ways of doing this – each with an example provided. Try adding each of these examples to your R script one by one on new lines and run each line in turn to observe the results

Use of the colon operator

The colon ":" operator provides a short hand way of creating a sequence of integer components in increments of 1 starting from the value to the left of the colon and ending with the value to the right of the colon. For example the following line:

```
Weight <- 70:80
```

Use of the c() "combine" function

The Combine c() function can be used to assign a series of individual values in any order to a vector. For example the following line:

```
Weight <- c(70, 69.3, 71, 68.7, 79, 77.4)
```

Use of Sequence generator seq()

The Sequence Generation seq() function cab be used to assign a series of components with start and end values that increment or decrement by a fixed amount. For example the following line:

```
Weight <- seq(72,73,0.2)
#Or
Weight <- seq(73,71.8,-0.3)
```

Note that:

- In the first case components were generated starting from 72 and ending at 73 in steps of 0.2
- In the latter case the values started at 73 and were decremented at each step by 0.3 down to the value 71.8
- The function seq has three arguments "from= ", "to= ", "by= " and the first example could have been written as:
  Weight <- seq(from=72,to=73,by=0.2)

c() in more detail

Note that the series of values handled by c() can include a combination of individual values, sequences declared with ":" and sequences declared with seq(). For example the following line:

```
Weight <- c(65, 67, 70:74, 69, seq(75,77,0.4))
```

Also, as we saw in the "Introducing RStudio" module, c() can be used to combine nominal components into a vector. For example the following line:

```
Colnames <- c("Id","Age","GenderMF","Hypert","SBP","DBP")
```

With each of the above examples you will have created a **Vector** variable and then assigned a series of components to it. Note that in the Environment panel it is possible to see:

- The type of variable – you will probably have examples "chr" and "num"
- The number of components in square brackets [1:n] where n will vary according to the example being run. As we will see shortly, the figures in square brackets in fact reflect indexing
- The first few values of the series of components held in the vector

Just as we can work with single scalar variables so we can also work with vectors. We can illustrate this by calculating bmi from weight and height. Into a fresh line of your script type or paste the following three lines:

```
Weight <- 70
Height <- 1.8
BMI <- Weight/Height^2
```

This should result in a variable called BMI with a single value. The last line assigned the result of Weight/Height^2 to a new variable called BMI.

We can repeat this with vectors containing multiple components. Into a fresh line of your script type or paste the following three lines:

```
Weight <- seq(60,72,0.3)
Height <- seq(1.2,2,0.02)
BMI <- Weight/Height^2
```

The variable BMI is now clearly a vector that holds components that are each the result of computing the formula on successive components of weight and height.

It is possible to visualise the data in these vectors in a number of ways. However, before doing that we will explain how help with using R functions can be obtained and also introduce the powerful command completion feature of RStudio.

Getting Help

There are at least four ways to obtain help with R and RStudio. From either the console screen or from a script line in the source panel we can use:

```
help(xxxx)          # help about function xxxx
?xxxx               # help about function xxxx
Press Fn1 (Function key 1) with cursor placed over function name
```

All of the above will open the bottom right RStudio Help panel and provide information about the function.

In addition it is also usually very easy to find help on line by typing "R xxxx" into your search engine (e.g. Google).

Exercise 1: Try all of the above help methods for the function **seq**
Exercise 2: You will find that it can take arguments in addition to 'from=', 'to=', and 'by='. Find out what these are, what they do, and experiment with their use. Hint – you may find that Google is quickest in getting an answer in plain English

Command completion

This is a powerful and very useful RStudio feature. In a nutshell, RStudio will attempt to complete names of objects and of functions when given just one or two of the starting letters. This may save key presses and hence time but it also significantly reduces the risk of mistyping names and getting upper / lower case wrong. The true value of this feature will become apparent when you have wasted 30 minutes trying to find an elusive error in a block of script lines that turns out to be due to the mistyping of an object name.

In summary, RStudio will help you by completing:

- Functions and their arguments
- Names of objects
- Filenames

Command completion can be triggered by pressing <TAB> after entering a few letters. A list of possible completions is offered to you. Clicking on the appropriate choice triggers completion. In some cases where there is only one option pressing <TAB> will automatically autocomplete with that. Clearly, if you have already mistyped you will not find what you are looking for.

Examples

Using summary() function:

On a fresh script line try typing 'sum' followed by <TAB>. Then from the picking list click on 'summary'. Notice that not only is the function name completed but also a double bracket is added along with a prompt for an object name to be added as an argument. Type 'B' followed by <TAB> and select 'BMI'. Then run the script line. Notice that R responds in the console panel with six parameters about the data held in the vector 'BMI'. The summary function can be very useful. From left to right it reports:
Minimum value, 1st quartile, Median, Mean, 3rd quartile, Maximum value.

Using plot() function

On a fresh script line try typing 'pl' followed by <TAB>, and then click on plot. RStudio completes the function name, adds closed brackets and prompts for two object names x, y,...
Initially, just enter 'B' and use object name completion as before.
Run the script line to see a plot of 'Index' on the x axis and 'BMI' on the y axis. You are now looking at all 41 components of the BMI vector indexed with the BMI of each on the y axis.

But what if we want to look at BMI plotted against either Weight or Height? On a new script line repeat the above steps but this time add a comma after adding / auto-completing BMI followed by 'we' and <TAB> to select correctly spelt 'Weight'. This now displays 'BMI' on the y axis and Weight on the x axis.

Exercise 1: Try doing the same for BMI and Height

Exercise 2: Get 'help' on plot and note that extra arguments can be added for main title, types of plot (e.g. point / line /bar). Try adding these. (click for help)

Viewing the data in a vector

Now we can quickly view the data held in the vector BMI. Try adding and running the following lines to your script

```
#Example using View:
        View(BMI)          #care with upper case 'V'
```

```
#Example using summary
        summary(BMI)

#Example using plot
        plot(BMI, weight)


#Example using hist
        hist(BMI)

#Example using boxplot
        boxplot(BMI)
```

Exercise 1:
Try each of the above and use <TAB> to complete and 'help' to find extra arguments. Try adding the simplest ones of these. NB for hist and for boxplot some of these arguments are very advanced and are thus likely to be difficult to comprehend at this stage

Exercise 2:
Try one or two of the above to explore one of the other vectors – Weight or Height

That completes the module 'Creating vectors and working with them'. It is recommended that you save your script before closing down RStudio

Lookup

Assign values to a variable
Use **<-** For example: Weight <- 70
Click **<ALT> + Left Arrow** to return

Set Working directory
File | More | Set as Working Directory
Click **<ALT> + Left Arrow** to return

Setting up plot arguments
plot(BMI,Height,"b",main="Bmi and height",col="blue")
For some reason R help does not mention the col argument but this works for most graphics and all of the obvious colours will work
Click **<ALT> + Left Arrow** to return