Exercises

Exercise 1 : Getting to know R and RStudio

1. Start RStudio on your computer. If you're using one of the University computers follow the instructions in Section 1.4 of the course manual.

2. Create a new R script (on the menu bar select File –> New File –> R Script). Save this Script as 'Aberdeen_R_course' (or something similar) in a convenient location on your computer (or on your USB drive if you're using a University computer). This will be your main R script file for the course to which you will add all your commands as you work through the manual and exercises.

3. Explore RStudio making sure you understand the functionality of each of the four windows. Take your time and check out each of the tabs in the windows. The function of some of these tabs will be obvious whereas others won't be useful right now. In general, you will write your R code in the script editor window (usually top left window) and then source your code into the R console (usually bottom left) by clicking anywhere in the relevant line of code with your mouse and then clicking on the 'Run' button at the top of the script editor window. If you don't like clicking buttons (I don't!) then you can use the keyboard shortcut 'ctrl + enter' (on Windows) or 'command + enter' (on Mac OSX).

4. Practice writing code in the script editor and sourcing this code into the R console by using the `help()` function to display the help file for the function `mean`. Notice that the help file is displayed in the bottom right window (if not then click on the 'Help' tab). Examine the different components of the help file (especially the examples section at the end of the help file).

5. The content displayed in the bottom right window is context dependent. For example if we write the code `plot(1:10)` and source it into the R console the bottom right window will display this plot (don't worry about understanding the R code right now, hopefully this will become clear later on in the course!).

6. Next, let's practice creating a variable and assigning a value to this variable. Create a variable called `first.num` and assign it the value `186282`. Click on the 'Environment' tab in the top right window to display the variable and value. Now create another variable called `first.char` and assign it a value `"my first character"`. Notice this variable is now also displayed in the 'Environment' along with it's value and class (`chr` - short for character class).

7. Remove the variable `first.num` from your environment. Check out the 'Environment' to ensure the variable has been removed.

8. Let's see what happens if we assign another value to an existing variable. Assign the value `"my second character"` to the variable `first.char` you created in Q6. Notice the value has changed in the 'Environment'. To display the value of `first.char` enter the name of the variable in the console.

9. Ok, let's leave RStudio for a minute. Using your favourite web browser, navigate to the R-project website and explore links that catch your eye. Make sure you find the R manuals page and the user contributed documents section. Download any manuals that you think you might find useful (some are listed in the course manual) and save them on your computer (or USB drive).

10. Click on the 'Search' link on the R-Project website. Use 'Rseek' to search for the term 'mixed model p values' (this is a controversial subject!) and explore anything that looks interesting. Also experiment with the 'R site search' and 'Nabble R Forum' links. Learning how to search for help when you run into a problem when using R is an acquired skill and something you get better at over time. One note of caution, often you will find many different solutions to solving a problem in R, some written by experienced R users and others by people with less experience. Whichever solution you choose make sure you understand what the code is doing and thoroughly test it to make sure it's doing what you want.

11. Ok, back to RStudio. Sometimes you may forget the exact name of a function you want to use so it would be useful to be able to search through all the function names. For example, you want to create a design plot but can only remember that the name of the function has the word 'plot' in it. Use the `apropos()` function to list all the functions with the word plot in their name. Look through the list and once you have figured what the correct function is then bring up the help file for this function.

12. Another strategy would be to use the `help.search()` function to search through R's help files. Search the R help system for instances of the character string 'plot'. Narrow your search by only searching for 'plot' in the `nlme` package.

13. R's working directory is the default location of any files you read into R, or save out of R. Although you won't be reading in or exporting files just yet (that's tomorrows job) it is useful to be able to determine what your current working directory is. So, figure out how to display your current working directory.

14. Now change your working directory to a more convenient location on your computer (or USB drive). Don't forget to check whether your working directory has been successfully changed.

End of Exercise 1