How to Use Script Files

At this point, students should be fairly comfortable installing the R program, downloading data sets from the web or WebCT and have a better understanding of how to control and manage their R environment. In this guide, we turn to the question of how to best manage and organize the various analyses that we do in this course and in R more generally.

As we have learned, commands are entered into and executed from the R console. There are several features that make this easier, including:

1. The use of the up and down arrow keys to cycle through previous commands.
2. Saving the R workspace so we do not have to re-create variables or download data sets repeatedly.
3. You also may have noticed it is possible to save and load your R history (a history of commands run in a previous analysis) from the "File" menu.

However, far and away the easiest way to manage your analyses for homework assignments, in-class lab exercises, and exams is to become comfortable using *script files*. Script files are nothing more than text files of the commands that one enters for their analyses. However, they offer several clear advantages over simply using the R console:

1. You can execute or run your R commands directly from the script file.
2. You can run the entire file at once, executing multiple commands in quick succession, or a subset of commands by highlighting only those that you wish to execute.
3. The above means that you rarely have to "save" your R Workspace, helping to ensure that you do not permanently alter your data sets and have to reload them. If you alter a data set by mistake, you can simply exit out of R, restart it, and execute the script file up to the point of the mistake.
4. You can more easily handle long R commands, especially those with titles, labels and other arguments that make the command go longer than the visible window.
5. You can add comments, allowing you to keep a record of what you did and why.
6. You can save and load these files quickly.

To get you started, from the R console, click on "File" at the top left and choose "New Script". This will bring up a new window entitled "Untitled – R Editor".

Note: You may also use any basic text editor or word processing software, such as Notepad or Wordpad (avoid using Word for this) and then simply cut-and-paste your R commands into the New Script File and execute. ***Mac users*** *apparently have problems using the built-in script file and must use an alternate text editor.*

Next, type in the following 5 commands into the script file or into your alternate text editor, hitting "Enter" after each one:

```
attach(POL201)
mean(height, na.rm=T)
median(height, na.rm=T)
sd(height, na.rm=T)
hist(height)
```

You'll see that it does not execute the commands at this point. To execute the first command, simply right click on it and choose "Run Line or Selection". You should see that command displayed in the R Console window and automatically executed.

Note: If the POL201 data set has already been attached then skip the first command and run the second—otherwise, you will get a "Masking" warning! For this reason and repeated runnings of script files, I often keep "attach()" commands out of script files and simply run them "by hand" like usual in the R console.

To run these commands all at once or a subset of them, simply highlight them, right click, and choose "Run Line or Selection".

Adding Comments to Your Script Files:

The ability to add comments to your script files is extremely useful for keeping track of what you have done and why. To do this, you need to precede all your comments with the # symbol. Going back to the list of commands above, for example, add in the following comments.

```
# Attach the POL201 data set for analysis
attach(POL201)
# Calculate the mean of student heights
mean(height, na.rm=T)

median(height, na.rm=T) # Calculate the median of student heights

sd(height, na.rm=T) # Calculate the standard deviation of student heights
hist(height)
# Create a histogram of student heights
```

Notice, you can put comments anywhere: before the command, after the command on the same line, or after the command on a separate line. This is all for your own benefit. R simply ignores anything after a # sign and does not execute it as a command. Notice also that you can put spaces between commands and/or comments as you desire to organize these files. R does not care.

Executing Long Commands with Script Files:

Another big advantage of using script files is that it makes it much easier to execute long commands. To illustrate, enter the following command ***all on one line*** into the R console (not the script file):

> plot(gpa ~ height, pch=9, col="red", main="Scatterplot of Student Height and GPA", xlab="Student Height", ylab="Student GPA")

You should quickly run out of space and you can no longer see the left side of the command, making it difficult to double-check it for accuracy. Even within the R console, there is a better solution: as long as you fail to complete the command, R will allow you to go to the next line to await the rest of the command. It does this by changing the R prompt from ">" to "+".

For example, using the same command above, type in just the first part of the command like so and hit Enter:

>plot(gpa ~ height,
+

You'll see that the prompt changes to a "+" sign meaning that is waiting for you something else, namely the rest of the command. So, let's continue with the command, like so:

+ pch=9, col="red",
+ main = "Scatterplot of Student Height and GPA",
+ xlab = "Student Height",
+ ylab = "Student GPA")

And, you should see the scatterplot appear in the graph window.

Breaking up the command this way is useful, but it has one distinct disadvantage: if you made an error early on in the command, when you cycle through them with the up and down arrow keys, you have to repeat the entire command again and in order. A bit annoying to say the least!

Script files eliminate this problem. To illustrate, let's enter the same command now into the script file. You'll likely find that it is also too long, given the size of the window for your script editor. No problem, you can just hit Enter whenever you want and continue with the command on the next line. You do not need to include the "+" sign for this.

Thus, your full command in the R script file might look something like this:

plot(gpa ~ height, pch=9, col="red", main="Scatterplot of Student
Height and GPA", xlab="Student Height", ylab="Student GPA")

Be sure to highlight the whole command in order to execute it properly.

Saving and Loading Script Files:

To save your script files, simply choose "File" from the pull down menu (when you have the script file open and active) and choose "Save As".

You can save it anywhere just like a "normal" file, but unlike "normal" files, R does not actually have or even require a meaningful end part (e.g., ".doc", ".txt", etc.) for script files. It calls them .R files, but this is not necessary. This means that it will not produce any special icon connected to a specific program. As a result, you will have to remember where you saved it and what you called it when trying to re-load in the future.

Remember, script files are just text files—any basic word processor can open them and as long as you save them again in basic text format, R will open them with little problem.

To load script files, go to "File" from the pull down menu at the R console and choose "Open Script". Find the file and open it.