# R5

# Using scripts

**Covered in R5**

- Using Scripts to enter and run commands
- Using comments in a script
- Saving scripts
- Opening scripts
- Editing scripts outside of R
- Sourcing scripts

## 1 Using scripts to enter commands

In previous chapters we executed commands in R by typing the commands directly into the console window and then hitting enter (*ie* return) to execute them.

However, this is not ideal for a number of reasons. It's harder to spot errors (as commands are interspersed with the outputs), it has to be run line by line so using the up arrow to re-run multiple line commands causes issues, and it's also harder to share our work with our colleagues.

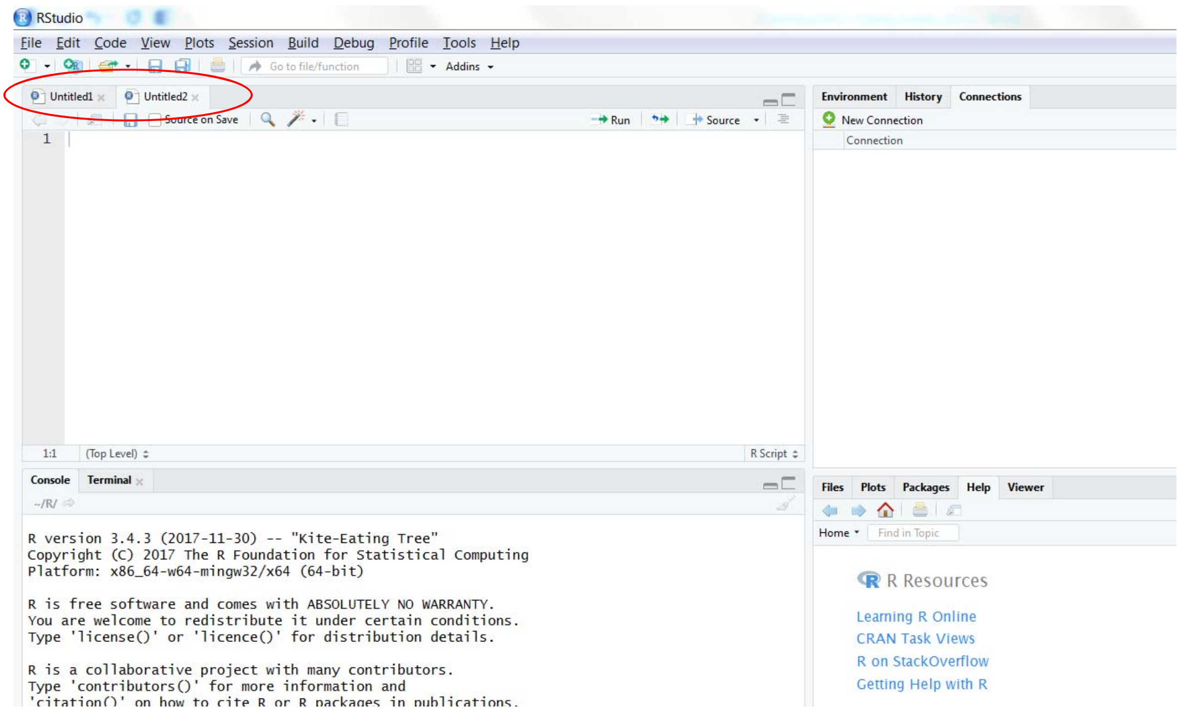
A better method is to use something called a *Script*. A Script is simply a text file with R commands in it.

### Starting a new script

We can start a new script in a number of ways:

- using RStudio's menu: File, New File, R Script
- using the drop-down menu next to the plus sign underneath the File menu
- pressing Ctrl+Shift+N.

RStudio opens up a new script window in the top-left corner of the screen. If a script is already open then it will open up a new tab in the same window so you can have multiple scripts open at the same time.



```
R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

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.

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.
```

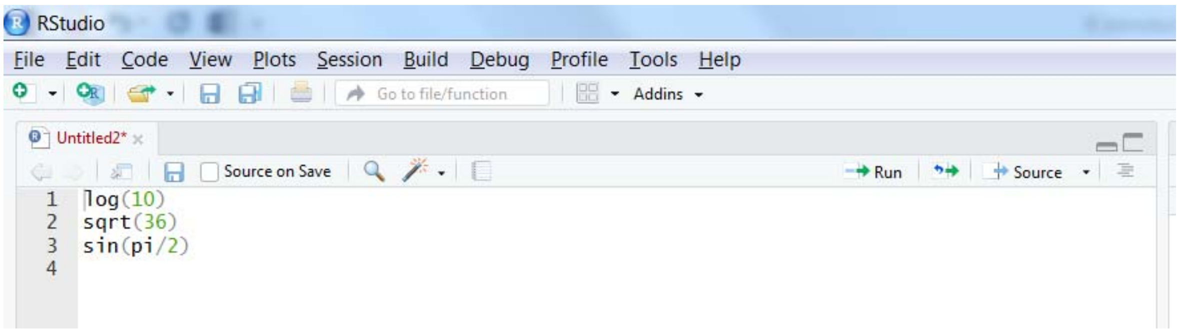
## Running commands from a script

We can type our commands in the editor but pressing enter does not execute them. For example we could type:

log(10)

sqrt(36)

sin(pi/2)



```
log(10)
sqrt(36)
sin(pi/2)
```

Just like a script for a play or movie which contains the lines that you read out – it has the lines of commands which can be "read out" into the console.

We can do this by copying commands line by line or in blocks (using Ctrl+C or Edit/Copy) from the editor and then pasting them into the console window (using Ctrl+V or Edit/Paste) and then pressing enter to execute them. If you want to copy everything then first select all using the standard shortcut command Ctrl+A.

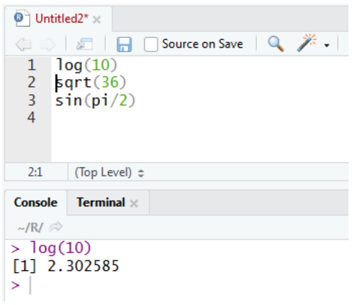
This makes scripts a very efficient way of running the same code again.

However, a quicker way than copying and pasting is to "run" the commands from the script editor.

To run a single command line in the Script, place the cursor in that line and then do one of:

- press Ctrl+Enter
- click on Run at the top of the Script widow
- use the Menu: Code, Run Selected Line(s).

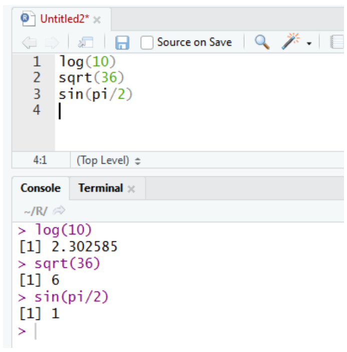
This "reads" the script into the console window *and* executes it. Note that each line of code, and its output, will appear in the Console as we run it:



```
log(10)
sqrt(36)
sin(pi/2)
```

```
> log(10)
[1] 2.302585
> 
```

Even more useful is that our cursor moves onto the next line in the script window ready for the next command. So if we press Ctrl+Enter three times, we will run the three lines of code displayed in this script:

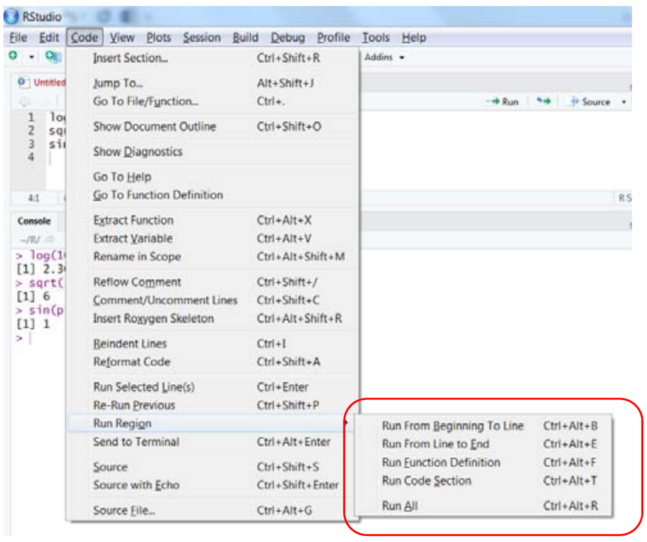


```
log(10)
sqrt(36)
sin(pi/2)
```

```
> log(10)
[1] 2.302585
> sqrt(36)
[1] 6
> sin(pi/2)
[1] 1
>
```

Similarly to run several lines in the script we just select at least part of those lines and press Ctrl+Enter.

Alternatively, there are useful options in the Code menu, as well as some more short-cut keys which you may find useful:


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Addins
Run
Source
Insert Section... Ctrl+Shift+R
Jump To... Alt+Shift+J
Go To File/Function... Ctrl+.
Show Document Outline Ctrl+Shift+O
Show Diagnostics
Go To Help
Go To Function Definition
Extract Function Ctrl+Alt+X
Extract Variable Ctrl+Alt+V
Rename in Scope Ctrl+Alt+Shift+M
Reflow Comment Ctrl+Shift+/
Comment/Uncomment Lines Ctrl+Shift+C
Insert Roxygen Skeleton Ctrl+Alt+Shift+R
Reindent Lines Ctrl+I
Reformat Code Ctrl+Shift+A
Run Selected Line(s) Ctrl+Enter
Re-Run Previous Ctrl+Shift+P
Run Region
Send to Terminal Ctrl+Alt+Enter
Source Ctrl+Shift+S
Source with Echo Ctrl+Shift+Enter
Source File... Ctrl+Alt+G
Run From Beginning To Line Ctrl+Alt+B
Run From Line to End Ctrl+Alt+E
Run Function Definition Ctrl+Alt+F
Run Code Section Ctrl+Alt+T
Run All Ctrl+Alt+R

Later in this chapter we'll give another way of running the whole script without even opening it.

## 2 Using comments

Since we may be coming back to our work some time later or sharing our scripts with colleagues, it might be helpful to add comments as an audit trail so that we, or our colleagues, can follow what we are doing.

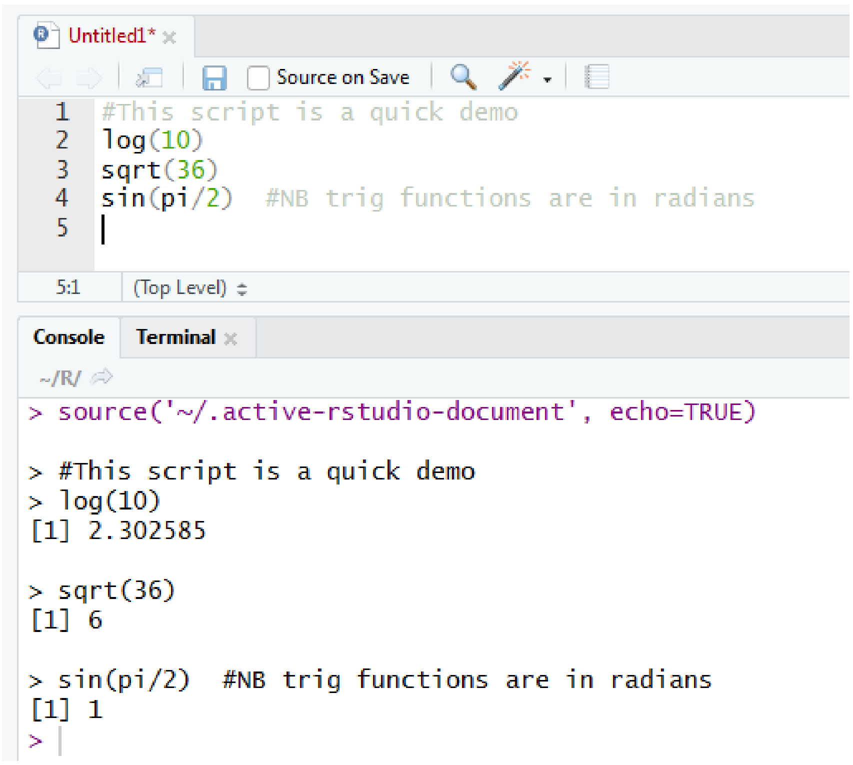
We enter comments by preceding them with a #. R then ignores anything to the right of the #

You can put comments on a separate line or after a command at the end of a line.

So let's put a comment before our three commands in our script and also one on the same line of a command but after it:

```
#This script is a quick demo
log(10)
sqrt(36)
sin(pi/2)  #NB trig functions are in radians
```

When we run all of this script we obtain the following:



```
#This script is a quick demo
log(10)
sqrt(36)
sin(pi/2)   #NB trig functions are in radians
```

```
> source('~/.active-rstudio-document', echo=TRUE)

> #This script is a quick demo
> log(10)
[1] 2.302585

> sqrt(36)
[1] 6

> sin(pi/2)   #NB trig functions are in radians
[1] 1
>
```

The comments are "executed" but there is no output, *ie* R ignores them.

Writing comments as an audit trail is a really good habit to foster – especially for the CP2 exam which tests audit trails.

## 3 Saving a script

There are a number of ways of saving a script:

- click on one of the save icons in either the toolbar or at the top of the Script window
- use the Menu: File, Save or Save As
- Press Ctrl+S.

(Note that if you are working in R and not RStudio, and your cursor is in the console window then using CTRL+S or the save icon won't save the script but something called the workspace which we'll cover in a later chapter.)

If you try to close down a script editor window before trying to save it then you will be prompted to save it.

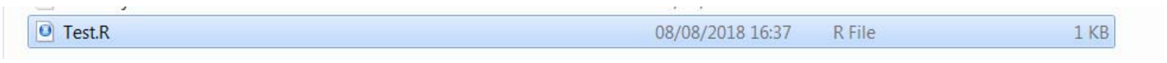
However, if you close down RStudio without saving a script then you won't be given a prompt. But don't worry, the script should still be there the next time you open RStudio.

If you don't change the file location, by default R will save the script in what is called the "working directory" (*ie* folder for those of you too young to remember DOS).

We will see how to change the working directory later.

Let's save the current script as "test" (note where it is being saved).

If you look in the folder where you saved the file – you can see that it has been saved with the extension ".R":



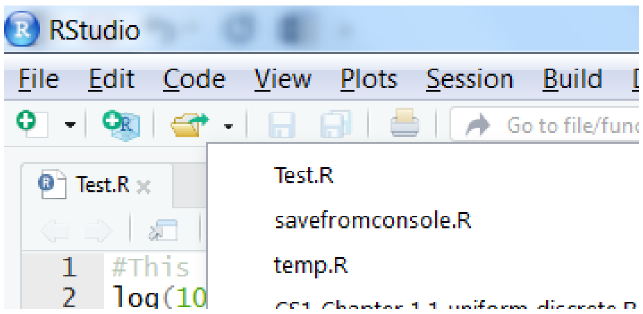
```
Test.R          08/08/2018 16:37     R File     1 KB
```

This let's R know it's a script file.

## 4 Opening a script

There are several ways of opening a script:

- use the menu: File, Open file
- use the shortcut key Ctrl+O
- use the open file icon in the toolbar. If you click on the drop-down arrow you will see a list of recently used files:

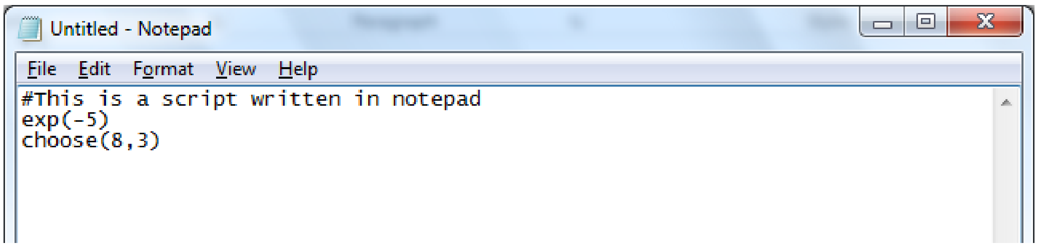


```
1  #This
2  log(10
```

## 5 Editing a script outside of R

Remember a script is just a text file containing R commands. So we could create such a file outside of windows using a text editor such as notepad.

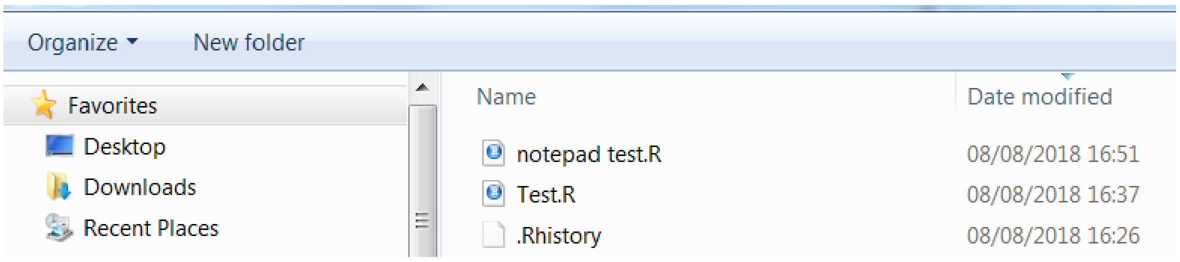
Let's type some commands into notepad:



```
#This is a script written in notepad
exp(-5)
choose(8,3)
```

And then when we save this file let's put a ".R" extension on the end of the file name, for example, "notepad test.R".

If we try to open a script in RStudio, we can see the file is there:



| Name | Date modified |
|---|---|
| notepad test.R | 08/08/2018 16:51 |
| Test.R | 08/08/2018 16:37 |
| .Rhistory | 08/08/2018 16:26 |

Clicking on it will open it successfully. Try it out just to be sure.

In fact, R and Rstudio can read normal text files (which have the .txt extension) as scripts. So we didn't have to put the ".R" extension on the end. If the file had been saved with a .txt extension then we could just have easily opened this in the same way using RStudio.

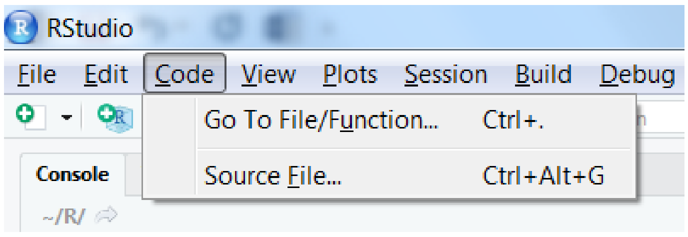
If you are using a word processor such as Word to create scripts then you will need to save the script as a ".txt" file to strip out all the "invisible" coding.

If you don't, RStudio will probably just open up the file in the programme it was written in.

## 6 Sourcing a script

We can actually run a whole script without opening it.

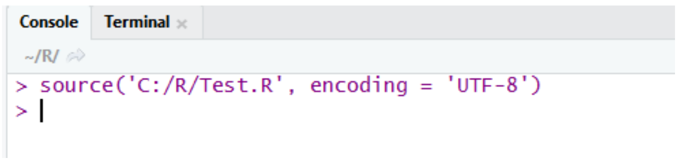
Choose Code, Source File from the menu:



Find the test.R file we created earlier (or a different file you might have created) and open.

(Note that we could again source a text file, *ie* one that ends in ".txt".

Even though it runs, nothing seems to happen in the console – all we have is the "source" command:



```
> source('C:/R/Test.R', encoding = 'UTF-8')
> 
```

Note the forward slashes in the filename path– more on that later.

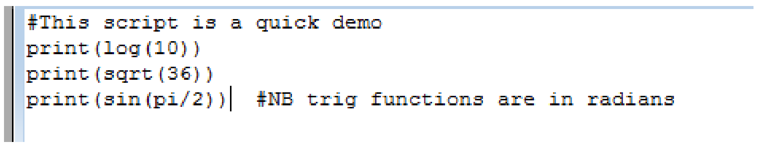
By default, R doesn't display anything unless we explicitly tell it to.

There are two ways of doing this. The first is to use the print command.

**print(<object>)**

Object is the term used in R to describe a "thing" that we perform commands on. We'll talk about objects more in the next chapter.

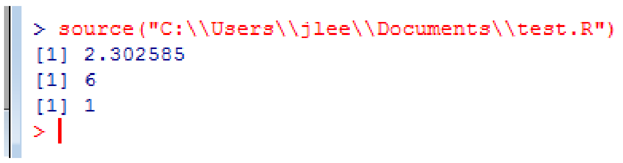
If we load up the "test.R" script and put print( ) around all of the commands as follows:



```
#This script is a quick demo
print(log(10))
print(sqrt(36))
print(sin(pi/2))  #NB trig functions are in radians
```

Now we'll save this script as before (Ctrl+S). Putting the cursor in the console we'll source the file as before using the menu item Code, Source File.

This time something does happen in the console window:



```
> source("C:\\Users\\jlee\\Documents\\test.R")
[1] 2.302585
[1] 6
[1] 1
> 
```

We can see that it has printed the output of the three commands.

The second way to display the outputs from the commands in the source script is to use the source command:

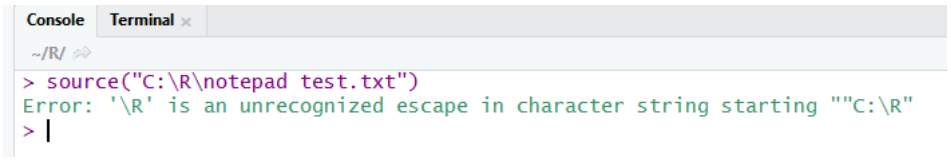
**source("<filename including filepath>" , …echo = …, …)**

The input value is the filename which includes the file path, for example C:\Documents\R. (The easiest way of getting this is to copy it from the address bar in Windows Explorer.)

So let's try running:

source("C:\R\notepad test.txt")

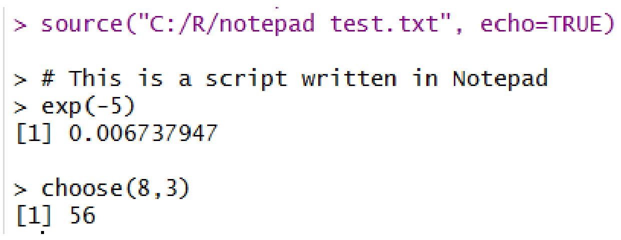
Unfortunately, we get the following error:



```
> source("C:\R\notepad test.txt")
Error: '\R' is an unrecognized escape in character string starting ""C:\R"
> |
```

This is because R doesn't like single backslashes (as they are used for other commands). So either we need to change each of them to double backslashes (*ie* C:\\R\\test.R) or to forward (normal) slashes (*ie* C:/R/test.R).

Let's try again with forward slashes, but also set one of the arguments echo = TRUE. We need to do this so that the output is printed in the console. Try leaving it out and you will see the same problem we had earlier with no output.



```
> source("C:/R/notepad test.txt", echo=TRUE)

> # This is a script written in Notepad
> exp(-5)
[1] 0.006737947

> choose(8,3)
[1] 56
```

In fact, the input as well as the output has been displayed, which is even more useful.

There are a number of other arguments that you coud use with the Source command but given that we don't think you'll use it very much in your CS1 or CS2 studies, we won't discuss them here. You can always use the help menus if you find you need them.

## 7 Summary

### Key terms

| | |
|---|---|
| Script/editor window | A window where commands can be written but not executed. We can transfer them to the console window and execute them using Ctrl+Enter |
| Comment | Text following a #, which R ignores (*ie* does not execute) but we can use to leave an audit trail |
| Object | Something R performs commands on |
| Run | Transfer a command from a script to the console and execute |

### Menus

| | |
|---|---|
| File, New File, R Script | Opens a new script editor window |
| File, Open File | Opens an *existing* script file |
| File, Save | Saves the script currently in view |
| Code, Run Selected Line(s) | Runs (*ie* transfers to the console and executes) the line of script the cursor is on or the lines if several are selected. |
| Code, Run Region, Run All | Runs (*ie* transfers to the console and executes) all the lines of the script in the console |
| Code, Source File | Runs a whole script file without opening it. Only displays outputs in the console if explicitly told to do so (by using the print command) |

### Key commands

| | |
|---|---|
| print(<object>) | Displays the <object> (or its output) in the R console. |
| source("<file>", echo=TRUE) | Runs the script <file> without opening it |
| | <file> includes the file path (eg C:\\test.R) |
| # | Comment: R then ignores anything to the right of the # |
| Ctrl+Enter | Runs the line or selected lines of script in the console |
| Ctrl+O | Opens an existing script file |
| Ctrl+Shift+N | Starts a new script file |
| Ctrl+S | Saves the script file |

## 8 Have a go

You will only get proficient at R by practising.

1. Use R to calculate the following:

   – log1000

   – $^{10}C_6$

2. Without referring to R4 (unless you get stuck), use R to:

   – find the arguments of the source function
   – find the html help page for the source function
   – run the examples for the print function
   – find a list of all 42 functions which contain the word "print".

3. Without referring to this chapter (unless you get stuck), use R to:

   – start a new script
   – write the comment "This is my R5 script" and then put the commands from Q1 into it
   – using any method, run the commands in the script (both individually and all together)
   – clear the console screen
   – save the script as "My R5 script" and close it
   – load the script back into R
   – close the script again and then run the script without opening it so it displays all the commands
   – open up notepad and write a script with the comment "This is my text R5 script". Then put in the commands above but include a command that will ensure the output is displayed. Save as a text file with the title "My text R5 script"
   – load up this script into R and run it
   – close this script and then run the script without opening it.