The RStudio Environment

Susan Holmes (c)

Exploring the different Panes

We organize our environment by using the View tab in the top of RStudio and have included a pane for the **Console** in the bottom left hand, a pane for **help/packages/plots** in the bottom right hand. The top right hand shows us our **Environment and History**.

The top left hand pane is used for editing and saving **Source code**, we haven't used this very much yet, but it is very useful for creating scripts, functions and code for easy reproducibility.

As an example, let's create a file by clicking on the File tab at the top: you see many choices when you click on new, you'll see a whole list of different file types:
- R Script
- R Notebook
- R Markdown

… are the first three on the list, we'll visit these in turn, let's start by opening a script called Record.R, this will serve us to capture a record of some of our work.

You can tailor you panes in your environment once and for all in the **Preferences** (access it by clicking on the RStudio tab at the top on the Mac or use the Options dialog **Tools > Options** menu in Windows),
then choose the Pane icon (forth one down).

More details on customization at RStudio

Taking material from your history to the Code window.

If you click on the history tab on the top right pane of RStudio, you can click on a few rows (click on the first row then command click on the next). Once you see that the rows

```
sqrs <- (2:17)^2
sqrs
```

have been chosen you can execute them by choosing the `<- To Console arrow` or you can store them in your Code window with the `<- To Source`

Try both now, first to execute those two lines, then to stock them in your Source Code window.

The Help Pane

If you type `?letters` The help about the vector of letters available in R appears in the right hand lower pane, if you have clicked on the Help tab.

Question Look at the help for other R objects.

Some functions whose help you can explore: `?log` `?sqrt` `?c` `?q`

Some data classes: `?vector` `?Date`

Note: On the top of the help page of the relevant topic a word appears in curly brackets, in most of the cases shown above you see `{base}`, this is because these objects and functions come from the base package. In fact we will download and use many supplementary R packages, for this we need access to the internet again.

Click on the Packages tab in the lower right hand pane.

Installing and using packages

Under the three icons, Install, Update and Packrat: You see a list with Name and Description, these are the packages that have already been installed on your system, let's add a few.

Click on the Install icon, apopup window will ask you to specify your preferred repository, for the time being we are going to install from CRAN, so leave that choice, then type in the package `ggplot2` leave the path as it is.

Install to libraries set as it is (unless you don't have permission to write to that directory).

ATTN Make sure there is a tick mark in the `Install dependencies` box. (many packages install other packages they need and depend on to run at the same time). Now the package ggplot2 should appear in the list under the g's.

A package can also be installed at the command line. If you look at what was happening in the Console when you installed ggplot2 you will notice install.packages("ggplot2") appeared.

So for instance try typing

```
install.packages("dplyr")
```

Installed packages have been downloaded and recorded on your hard drive, however, for every new session, you will need to tell R to load that particular package, this is done by typing: `library(ggplot2)`

You can also load it by clicking in the box next to the relevant package so that a blue tickmark appears. To find out more about the packages, click on their names the help pane will come up for instance, click on the `datasets` package.

Go down and click on the `women` data set, what does it consist of?

```
women
```

```
##    height weight
## 1      58    115
## 2      59    117
## 3      60    120
## 4      61    123
## 5      62    126
## 6      63    129
## 7      64    132
## 8      65    135
## 9      66    139
## 10     67    142
## 11     68    146
## 12     69    150
## 13     70    154
## 14     71    159
## 15     72    164
```

The most useful shortcuts

We have already seen some of these in action, here is a more complete list:
- Up arrow in the Console window goes through your command history.

- Ctrl + up arrow (command + up arrow on a Mac) is tailored history and auto-complete tool. Start typing and hit that key combination, and it shows you a list of every command in your historry that start with those keys. Select the one you want and hit return. This works only in the Console.

- Tab is a useful auto-complete key. If you start typing in the console or editor and hit the tab key, RStudio suggests names of functions or files, you then select the one you want and hit either tab or enter to accept it.

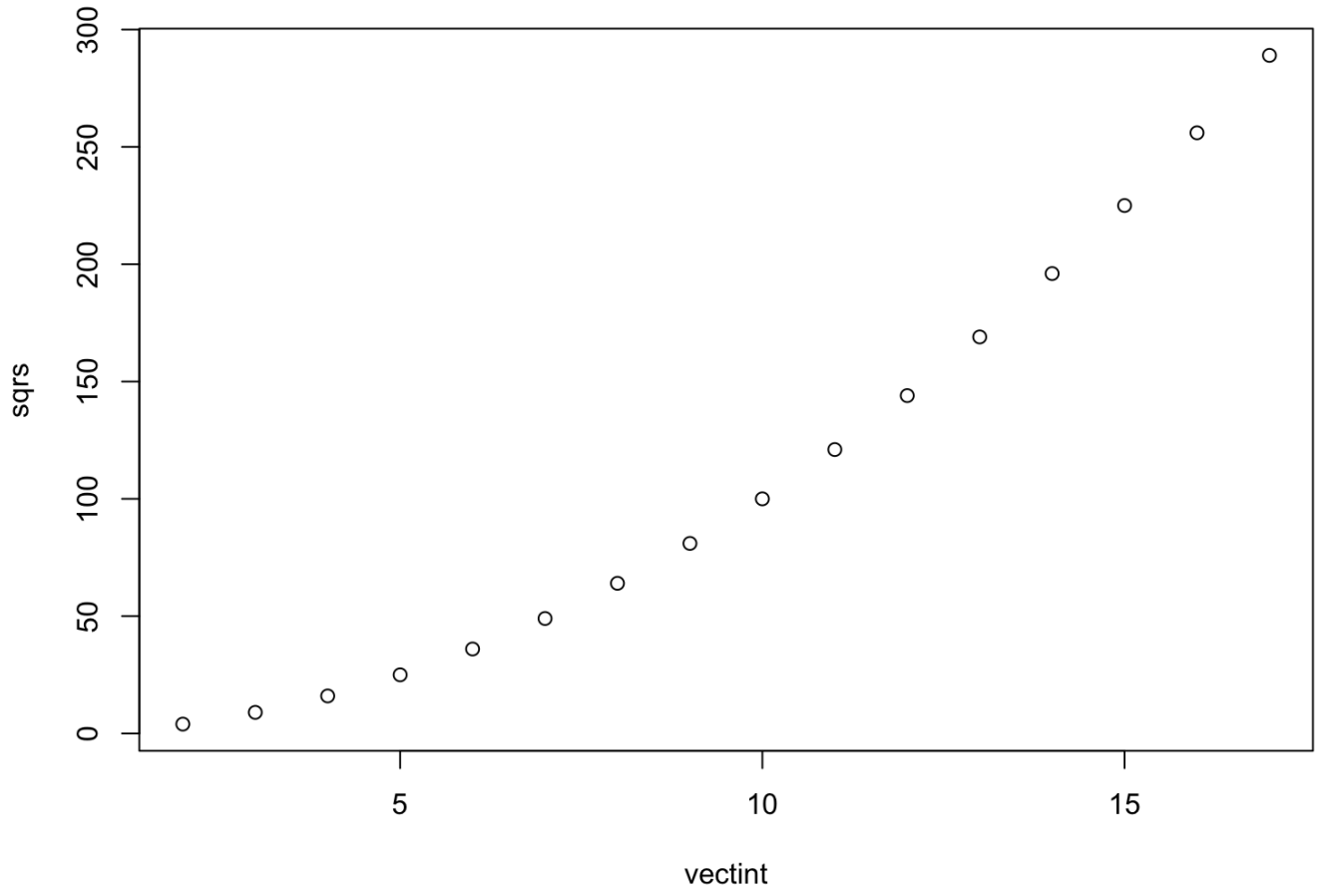
- When editing in the Source Code pane use Ctrl + enter (command + enter on a Mac) to execute the current line of code in the console. If you select multiple lines of code in the editor and then hit ctrl/cmd + enter, all of them will run.

- Interrupt the current command with *Esc*, this can be useful when you send the program into doing something you regret.

Simple plotting

If you click on the `Plots` tab in the lower right hand pane, you will see results from plots you ask for.

```
vectint <- 2:17
sqrs <- vectint^2
plot(vectint,sqrs)
```



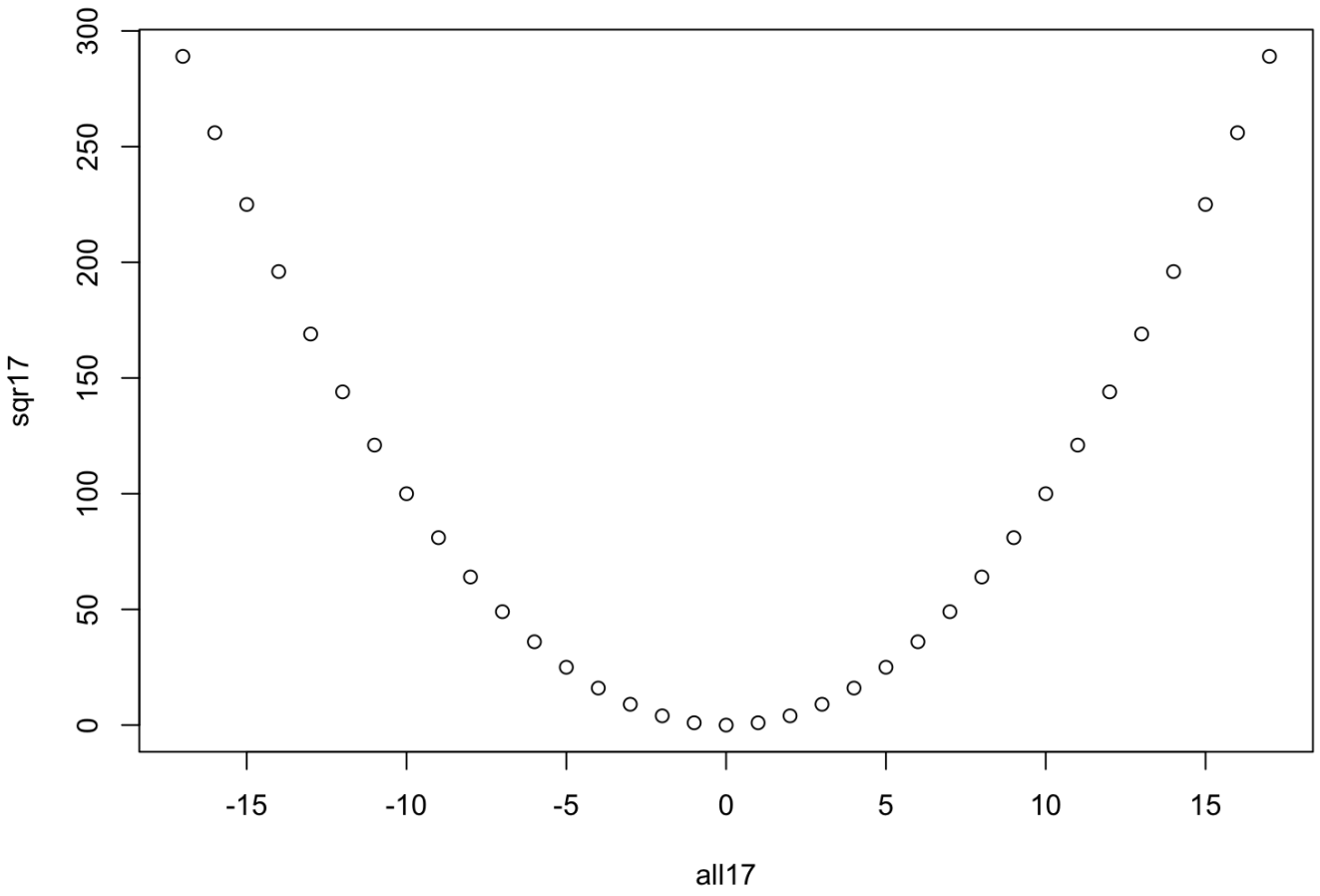
You'll notice that if you just use the Ctl-Enter way of asking for the plot, the plot may appear in your Console screen,

this is true for recent versions of RStudio that have a Notebook type interface like Python notebooks.

To have the plot appear in the Plots pane, use cut and paste to the Console screen. Once you have made the plot you can choose export to save the plot to a file.

Suppose that now we would like to plot the squares of all the numbers between -17 and 17.

```
all17 <- -17:17
sqr17 <- all17^2
plot(all17,sqr17)
```



Q: Try different options in `plot`:

- `plot(all17,sqr17,type='l')`
- `plot(all17,sqr17,type='b')`
- `plot(all17,sqr17,type='c')`

Summary of this Session:

In this session we have learned more about using the RStudio environment:

- There are four possible panes that can be customized using the Options and Preferences.
- Cutting and pasting across different panes reduces the amount of typing necessary.
- There are many different keyboard shortcuts available, up arrow and ctrl uparrow, and tab are some really important ones.
- The Pane on the lower right can help you install packages, visualize plots or peruse help files.

Your Turn:

Go to https://support.rstudio.com/hc/en-us/sections/200107586-Using-RStudio and look in the Keyboard Shortcuts, how do you move the cursor to the Console with a keyboard shortcut (without using the mouse)?

Customize the appearance of your code and console, change the background.