

R in the computer labs and at home

Why R?

R is an implementation of a theoretical programming language called S. It has been developed and is maintained by the world academic community and unlike S-plus, the other currently available implementation of S, it is free. R is most widely used for statistical computing and graphics, but is a fully functional programming language well suited to scientific programming in general. Versions of R are available for a wide variety of UNIX platforms (including Linux), Windows and MacOS.

Using R in the lab

We run R under Windows XP using a freeware text editor Tinn-R, which you use to edit your source files (the text files which contain the R programmes). You need to have both R and Tinn-R running at the same time, but it does not matter in which order you start them. Both can be accessed from the **Start -> Programs** menu and note that the Windows version of R is called **Rgui.exe**.

We use Tinn-R for two reasons. Firstly, when you open a file with a **.r** extension, Tinn-R applies colour highlighting of the R syntax rules, which helps you see many common mistakes. Secondly, at the press of a button, Tinn-R will run R programmes for you using R's **source(file)** command. The button is called **Send file (source)** and appears in the top left corner of the Tinn-R window. It can also be accessed from the menu **R -> Send to R -> File (source)**. This button can only be used if R is running and you have saved your programme.

Do *not* use the button **Send all**, which is next to **Send file (source)**. **Send all** just copies and pastes your programme into R, which will cause problems when your programmes require input from the keyboard.

You can set up Tinn-R so that after it sends a programme to R it either stays with R or returns to Tinn-R. Because we want to see the results of our programmes we will want to stay with R. If this is not happening, then uncheck the button named **Toggle return focus after sending to R**. It can also be accessed from the menu **Options -> Return focus after sending to R**.

When you run R, it has a working directory, which is where it first looks for user written programmes and data files. You can discover the current working directory using the R command **getwd()** and change it using **setwd(dir)**. For example, if you had a USB flash drive mounted as drive F and you wanted to run and save programmes for computer laboratory 1 in the directory **F:\lab1**, you would type **setwd("F:/lab1/")**. Alternatively **Rgui.exe** has a menu command for changing the working directory. Note that R uses the UNIX convention of forward slashes / in directory and file addresses. If you are using Tinn-R then you will probably not have to use the **getwd()** and **setwd(dir)** commands, because if you use the **Send file (source)** button then Tinn-R uses the absolute pathname of your programme file.

An occasionally important button is the interrupt or stop button. Both **Rgui.exe** and Tinn-R have stop buttons, whose job is to stop a programme that has gotten carried away and won't stop on its own.

When you are finished using R you quit with the command **q()**.

Saving your work

Your own files must be saved on Disk D (Student Data) or a removable drive such as a USB flash drive. The Student Data disk is not a suitable place to save your work long term as files there can be deleted overnight. At the end of each lab session you should save your work either on a removable drive or else email it to yourself.

Getting R to use at home

The computer labs in the Richard Berry building are open from 9 am. to 6 pm. However it is strongly recommended that you to get R running on your computer at home, should you have one.

You can download R from one of the many mirror sites of the Comprehensive R Archive Network (CRAN), for example <http://cran.ms.unimelb.edu.au/>. Advice on downloading and installing R is available from the FAQs provided on the CRAN site. If you have a slow internet connection at home then you may find it easier to download the R installation file onto a USB flash drive using a university machine, then take that home.

Tinn-R is available from <http://www.sciviews.org/Tinn-R/>. We recommend version 1.17.2.4, which is easier to set up than the latest version. If you are using a UNIX environment then Emacs and Xemacs work well with R. For those using the MacOS, the AlphaX text editor apparently interacts well with R, though I have no personal experience of this.

In UNIX you start an R session simply by entering the command `R` (assuming your path has been updated to include the R binaries). The MacOS R implementation is called `R.app`.

Help

To find out more about an R command or function `x` you can type `help(x)` or just `?x`. If you cannot remember the exact name of the command or function you are interested in then `help.search("x")` will search the available help files for the keyword `x`.

For a nice HTML help interface type `help.start()`. This will allow you to search for help and also provides links to a number of manuals, in particular the highly recommended “An Introduction to R”. For further documentation, a good place to start is the CRAN network <http://cran.ms.unimelb.edu.au/>, which gives references and links to online resources provided by the R community.

Tinn-R has its own R quick reference guide called R card, which you can access from the panel on the left of the Tinn-R window. R card groups R commands under a variety of headings (Math and Programming are particularly relevant to us) and provides a very brief description of each command.