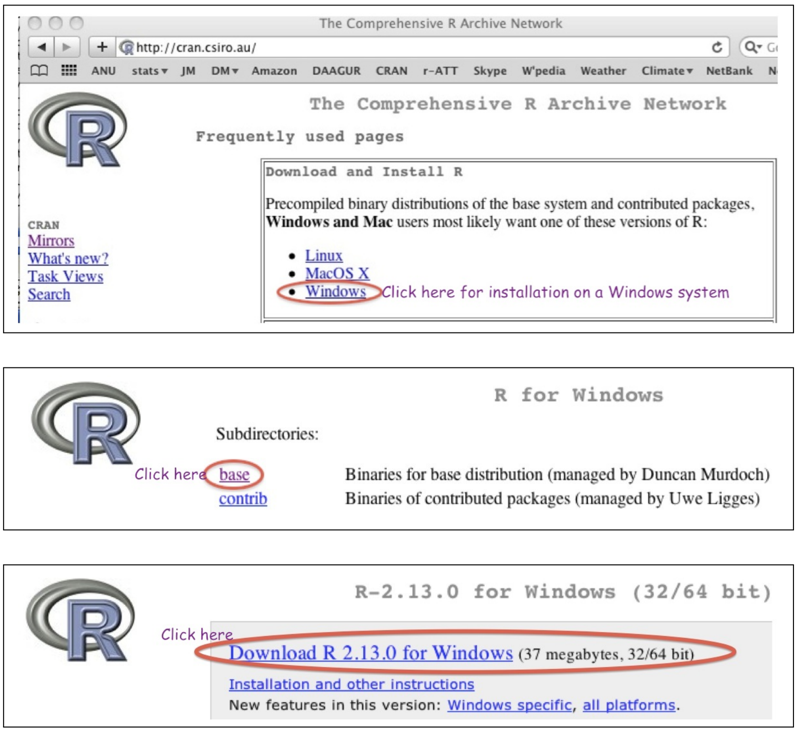
1

Preliminaries

1.1 Installation of R

Click as indicated in the successive panels to download R for Windows from the web page `http://cran.csiro.au`:



The Comprehensive R Archive Network

Frequently used pages

CRAN
Mirrors
What's new?
Task Views
Search

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Linux
- MacOS X
- Windows Click here for installation on a Windows system

R for Windows

Subdirectories:

Click here base Binaries for base distribution (managed by Duncan Murdoch)
contrib Binaries of contributed packages (managed by Uwe Ligges)

R-2.13.0 for Windows (32/64 bit)

Click here
Download R 2.13.0 for Windows (37 megabytes, 32/64 bit)
Installation and other instructions
New features in this version: Windows specific, all platforms.

Figure 1.1: This shows a sequence of clicks that will download the R installation file from `cran.csiro.edu`. At the time of writing, the website will offer R-3.0.2 rather than R-2.13.0. The site `cran.csiro.edu` is one of two Australian CRAN (Comprehensive R Archive Network) sites. The other is: `http://cran.ms.unimelb.edu.au/`

Figure 1.2: On 64-bit Windows systems the default installation process creates two icons, one for 32-bit R and one for 64-bit R. Additional icons can be created as desired.

Click on the downloaded file to start installation. Most users will want to accept the defaults. The effect is to install the R base system, plus recommended packages. Windows users will find that one or more desktop R icons have been created as part of the installation process.

Depending on the intended tasks, it may be necessary to install further packages. Section 1.3 describes alternative ways to install packages.

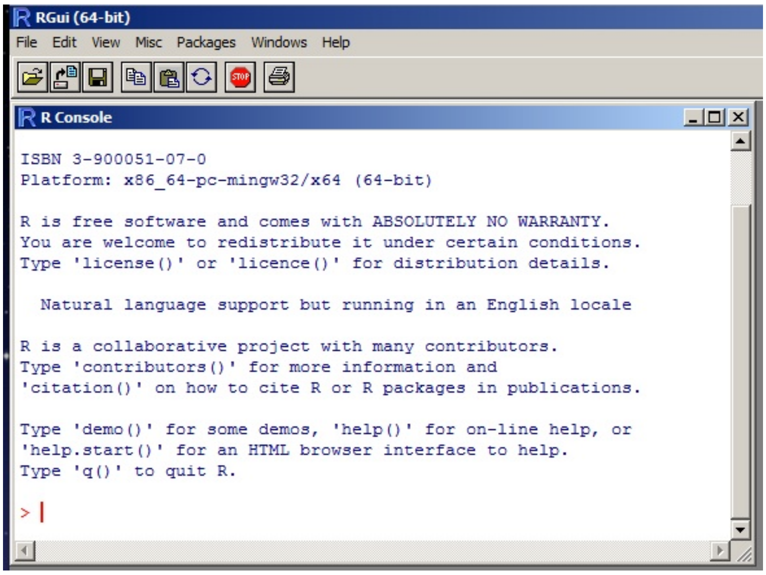
An optional additional step is to install RStudio. RStudio has abilities that help in managing workflow, in navigating between projects, and in accessing R system information. See Chapter 3.

Clicking on the RStudio icon to start a session will at the same time start R. RStudio has its own command line interface, where users can type R commands.

1.2 First steps

Click on an R icon to start an R session. This opens an R command window, prints information about the installed version of R, and gives a command prompt.

Readers who have RStudio running can type their commands in the RStudio command line panel.



```
ISBN 3-900051-07-0
Platform: x86_64-pc-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.

  Natural language support but running in an English locale

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.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

Figure 1.3: Windows command window at startup. This shows the default MDI (multiple display) interface. For running R from the R Commander, the alternative SDI (single display) interface may be required, or may be preferable. The Mac GUI has a SDI type interface; there is no other option.

The > prompt that appears on the final line is an invitation to start typing R commands:

For example, type 2+5 and press the Enter key. The display shows:

```
> 2+5
```

```
[1] 7
```

The [1] says, a little strangely, "first requested element will follow". Here, there is just one element.

The result is 7. The output is immediately followed by the > prompt, indicating that R is ready for another command.

Try also:

```
> result ← 2+5
> result
```

```
[1] 7
```

The value 7 is now stored (in the user *workspace*) with the name `result`. Typing `result` on the command line prints the value 7.

Type `ls()` to list the objects in the workspace, thus:

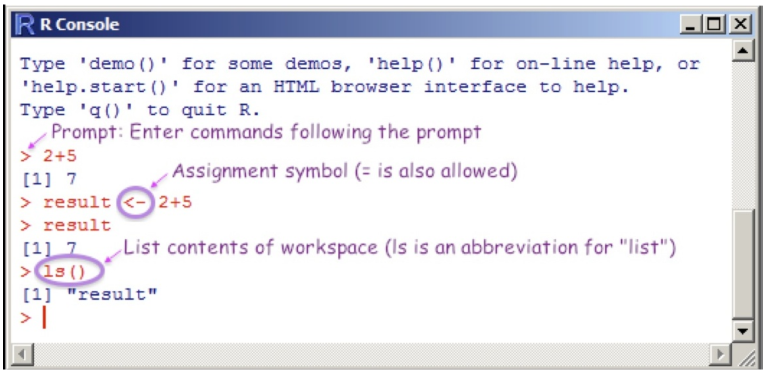
```
> ls()
```

```
[1] "oldopt" "result"
```

The session *workspace* holds objects that the user has created or input, or that were there at the start of the session and not later removed.

Starting from an empty workspace, `result` was the only object stored.

Figure 1.4 shows, with annotations, the screen as it appears following the above sequence of commands.



```
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
> 2+5
[1] 7
> result <- 2+5
> result
[1] 7
> ls()
[1] "result"
>
```

Prompt: Enter commands following the prompt

Assignment symbol (= is also allowed)

List contents of workspace (ls is an abbreviation for "list")

Figure 1.4: This shows the sequence of commands that are demonstrated in the text, as they appear on the screen, with added annotation.

1.2.1 Points to note

Printing	Typing the name of an object (and pressing Enter) displays (prints) it contents.
Quitting	To quit, type `q()`), (not `q`)
Case matters	`volume` is different from `Volume`

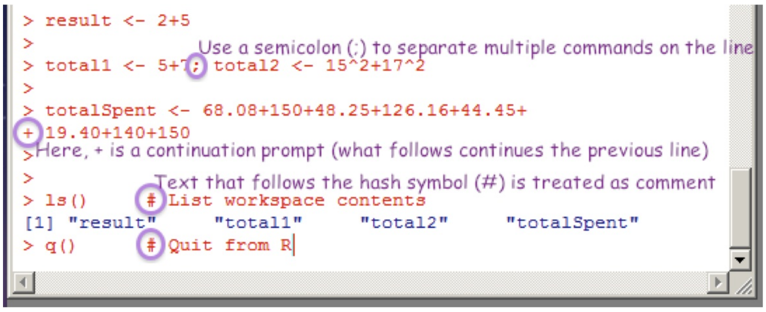
Typing the name of an object (and pressing the Enter key) causes the printing of its contents, as above when `result` was typed. This applies to functions also. Thus type `q()` in order to quit, not `q`.[1]

Upon typing `q()` and pressing the Enter key, a message will ask whether to save the workspace image. Clicking Yes (usually the safest option) will save the objects that remain in the workspace – any that were there at the start of the session (unless removed or overwritten) and any that have been added since. The workspace that has been thus saved is automatically reloaded when an R session is restarted in the working directory to which it was saved.

Note that for names of R objects or commands, case is significant. Thus `Myr` (millions of years, perhaps) differs from `myr`. For file names,[2] the operating system conventions apply.

[1] Typing `q` lists the code for the function.

[2] Under Windows, case does not distinguish file names. Under Unix (the Mac OS X version is a partial exception), case does so distinguish.



```
> result <- 2+5
>
> total1 <- 5+7; total2 <- 15^2+17^2
>
> totalSpent <- 68.08+150+48.25+126.16+44.45+
+ 19.40+140+150
>
>
> ls()       # List workspace contents
[1] "result"      "total1"      "total2"      "totalSpent"
> q()        # Quit from R
```

Use a semicolon (;) to separate multiple commands on the line

Here, + is a continuation prompt (what follows continues the previous line)

Text that follows the hash symbol (#) is treated as comment

Figure 1.5: Note the use of the special characters: **;** to separate multiple commands on the one line, + (generated by the system) to denote continuation from previous line, and # to introduce comment that extends to end of line.

Commands may, as demonstrated in Figure 1.5, continue over more than one line. By default, the continuation prompt is +. As with the > prompt, this is generated by R, and appears on the left margin. Including it when code is entered will give an error!

Here is a command that extends over two lines:

```
> result <-
+ 2+5
```

1.2.2 The working directory

Associated with each session is a working directory where R will by default look for files. In particular:

Under Windows, if R is started by clicking on an R icon, the working directory is that specified in the Start in directory specified in the icon Preferences. Subsection A.1 has details on how to specify the Start in directory for an icon.

- If a command inputs data from a file into the workspace and the path is not specified, this is where R will look for the file.
- If a command outputs results to a file, and the path is not specified, this is where R will place the file.
- Upon quitting a session, the “out of the box” setup will ask whether you wish to save an “image” of the session. Answering “Yes” has the result that the contents of the workspace are saved into a file, in the working directory, that has the name **.RData**. Next time a session is started in that working directory, the last **.RData** file that was saved in that directory (if any) will be used to restore the workspace.

For regular day to day use of R, it is advisable to have a separate working directory for each different project.

RStudio users will be asked to specify a working directory when setting up a new “project”.

1.3 Installation of R Packages

Installation of R Packages (Windows & MacOS X)

Start R (e.g., click on the R icon). Then use the relevant menu item to install packages via an internet connection. This is (usually) easier than downloading, then installing.

For use of command line instructions to install packages, see below.

A fresh install of R packages is typically required when moving to a new major release (e.g., from a 3.0 series release to a 3.1 series release).

The R GUIs, under both MacOS X and Windows, have helpful abilities for package installation.

The MacOS X GUI is especially good.

Installation of packages from the command line

To install the R Commander from the command line, enter:

```
install.packages("Rcmdr", dependencies=TRUE)
```

By default, a CRAN mirror is searched for the required package. Subsection 4.3.1 notes available repositories.

Among the dependencies are the graphics packages *rgl* (3D dynamic graphics), *scatterplot3d*, *vcd* (visualization of categorical data) and *colorspace* (generation of color palettes, etc).

Installation of Bioconductor packages

To set your system up for use of Bioconductor packages, type:

```
source("http://bioconductor.org/biocLite.R")
biocLite()
```

For installation of Bioconductor packages from the GUI, see Subsection A.4.

Additional packages can be installed thus:

```
biocLite(c("GenomicFeatures", "AnnotationDbi"))
```

See further `http://www.bioconductor.org/install/`.

1.4 Summary

One use of R is as a calculator, to evaluate arithmetic expressions. Calculations can be carried out in parallel, across all elements of a vector at once.

The R Commander GUI can be helpful in getting quickly into use of R for many standard purposes. It may, depending on requirements, be limiting for serious use of R.

Use `q()` to quit from an R session. To retain objects in the workspace, accept the offer to save the workspace.