2016

Introducing R

Lawrence Leemis

Chapter 1

Introducing R

R is a free language that can easily be downloaded from the internet. Regardless of whether you work in industry, academics, or government, or if you just use the language for personal use, you will always have access to the R language. So why use R? There are at least six answers to this question. First, R is freeware. It can be used on a desktop or a laptop computer. Second, R is capable of performing numerical calculations on scalars, vectors and matrices, and can be used as a high-powered calculator. Third, R has built-in functions for performing probability and statistical calculations, and R can run simulation experiments using these functions. Fourth, R has extensive graphics capabilities so that it can produce production-level graphics. Fifth, R can be used as a programming language. R allows the user to execute several types of loops for iteration and also supports conditional execution of code. Sixth, hundreds of contributors have written R code contained in "packages," which continue to extend the language in the same way that apps have transformed handheld devices.

The R language, originally known as S, was developed in 1976 at Bell Labs by John Chambers and his colleagues. S-Plus is the current commercial version of S. In 1993, the free, open source R language was first developed by Ross Ihaka and Robert Gentleman (notice the common first initial) from the University of Auckland. R and S have a very large overlap of capabilities. If you know one, you essentially know the other. The differences are minor. One advantage to R over S, however, is the number of packages that have been written in R that are capable of extending the base language. These packages can be quite useful in specific niche applications.

The orientation for R is as follows. R is a vector-based language that uses vectors as its primary data structure. R has a command-line orientation rather than a Graphical User Interface (GUI) menu orientation. Although this might seem a bit antiquated at first, the size and capability of the language force this orientation. The focus here will be on the syntax, that is, the rules for issuing R commands. There are plenty of other R books on the market that are much more encyclopedic in nature or cover specific applications. This book is designed to be a quick introduction to the language for R novices. More advanced books on R will be much easier to read once you master the basics presented here.

You will notice that R tends to favor short, abbreviated variable and function names, which is helpful on your fingers in terms of saving keystrokes. It is also a heritage from the C language and the Unix environment that was present at Bell Labs in the 1970s.

The R language can be located by doing a web search on the letter R in a browser. Alternatively, you can use the website `http://www.r-project.org`. Once you get to the website you should pick a mirror site that is near your location and install the binary version. You will need to choose an appropriate platform: Windows, Mac, or Linux. For your first installation, it is probably best

to select the default location for installing R. Installing R is similar to installing any other type of software. A license agreement must be accepted and you will want to install the most recent base version of the software. R is a fairly large language, so installing it will typically take a few minutes. On most platforms, an icon with the letter R will appear on your desktop after downloading and installation. To launch R, double-click the R icon on your desktop. This will bring up a window which contains a greater-than prompt, which looks like this:

```
>
```

The prompt indicates that R is waiting for an R command from you. The remainder of this book describes what you can type after the prompt and the associated results. If you type any valid R command at this point and press the return key, R will display the output. If a command is too long for a single line, it is fine to press the return key and continue the command over several lines. There is a + prompt that reminds you that you are completing a command from the previous line. If a command requires significant processing, there will be a time delay for processing before a new prompt appears.

When your R session is finished, you can quit R by typing `q()`, for quit, at the prompt. A dialog box will open that asks: `Save workspace image? [y/n/c]`. This determines whether R saves the objects that you have created during the R session for a future R session. After responding to that question, your R session is completed.

Experienced programmers who are used to working with compiled languages may find R's orientation foreign. The practice of submitting a command, viewing the results, then perhaps submitting a subsequent command based on the results is not the way compiled languages are generally used. This orientation is largely a result of the roots of R in exploratory data analysis. One preliminary look at the data often guides subsequent analysis techniques. It is easy to execute a series of R commands that are stored in an external file if the exploratory data analysis approach is not appropriate in a particular setting. Also, R stores its matrices in a column-major orientation, as opposed to the more standard row-major orientation. This is also an artifact of R's roots in exploratory data analysis and the associated data structures required to efficiently store data sets.

R is an interpreted rather than a compiled language, which means that each R command must be interpreted and then executed. Therefore, run times for R code will be a bit slower than comparable code in a compiled language. Some speed pick-up is possible, however, with vector-based programming, which will be described later in the text.

"Comments" are useful for documenting the purpose of a particular R command for yourself or for someone else who might use the code. A comment consists of text that is ignored by R as it processes your R input command. When R encounters the # symbol, it ignores all subsequent input on that line.

One weird trick that can save you some typing is to use the up arrow to repeat a previous command. Every time the up arrow is pressed, the previous R command is displayed. If this command needs to be altered, you can use the destructive backspace, or the nondestructive backspace (the left arrow) to delete or edit R commands. Pressing the enter key executes the original or modified previous R command.

The R language does not have a natural way to sequence the introduction of the various topics. I have decided to sequence the chapters in the following fashion.

- Chapters 2 and 3 introduce elementary arithmetic operations and named storage in R.
- Chapters 4, 5, and 6 introduce three elementary data structures: vectors, matrices, and arrays. In these early chapters, these data structures are filled with numeric elements.

- Chapters 7 and 8 introduce built-in and user-written functions.
- Chapter 9 introduces helpful utilities.
- Chapters 10, 11, and 12 introduce three other types of elements that can be stored in a data structure: complex numbers, character strings, and logical elements.
- Chapters 13 and 14 introduce methods for comparing elements and coercing elements to a particular type.
- Chapters 15 and 16 introduce two advanced data structures: lists and data frames.
- Chapter 17 surveys some data sets that are built into R.
- Chapter 18 introduces functions that interface R with the outside world.
- Chapter 19 introduces functions that are useful in probability calculations.
- Chapters 20 and 21 introduce R's graphical capabilities.
- Chapters 22, 23, and 24 introduce R's programming capabilities.
- Chapter 25 shows how to conduct simulations in R.
- Chapter 26 surveys R's capability for performing statistical inference.
- Chapter 27 introduces linear algebra functions that are built into R.
- Chapter 28 shows how to extend R's base capability by using packages.

You will find R to be a very efficient language for performing both simple and complex calculations. R can be used for analyzing data, performing simulations, performing calculations, etc. The value of a language like R was recognized long ago.

> Civilization advances by extending the number of important operations which we can perform without thinking about them. —Alfred North Whitehead (1861–1947)

The next chapter introduces R in its simplest form—as a calculator.

Exercises

1.1 Download R from the website `http://www.r-project.org` onto your desktop or laptop computer.

1.2 Visit the comprehensive R archive network (cran) website `http://cran.r-project.org`. Browse some of the online documentation for R.