2

HERE'S WHY WE LOVE R AND HOW TO GET STARTED

Difficulty Scale ☺ ☺
(takes a bit of practice, but after a while a new way to look at things!)

WHAT YOU WILL LEARN IN THIS CHAPTER

- ✦ Understanding what R is and a short history of where R came from
- ✦ Understanding the pluses and minuses of using R
- ✦ Defining *open source*
- ✦ Downloading and installing R
- ✦ Learning a new language, R
- ✦ Using R Help
- ✦ Learning some important lingo
- ✦ Downloading and installing RStudio
- ✦ Understanding RStudio

A VERY SHORT HISTORY OF R

Way back in the olden days (1996), Ross Ihaka and Robert Gentleman published a paper titled "R: A Language for Data Analysis and Graphics" in the *Journal of Computational and Graphical Statistics*. That paper was the formal beginning of this adventure they called a "statistical computing language." They actually started work around 1992. And, yes, it's called R because of the first letter in each of the first names of the original authors.

Many statistical and programming packages existed before then (such as SPSS, Stata, Minitab, and S that R is based on), and many were very popular and (still are) successful. It's difficult to judge what the motivation was for Ihaka and Gentleman to make this significant contribution, but at least part of it was to deliver a set of very powerful and flexible tools to the community at large and one that would be free for the using and open to change.

Currently, R (often referred to as **R base**) is maintained and kept very alive through the R Development Core Team and the R Foundation.

THE PLUSES OF USING R

There are several reasons why R has become so popular over the past few years.

First, it is absolutely, no questions asked *free,* meaning you never need to spend a penny for new software every time there's a new version or rent it from some company.

It is *open source,* meaning that the underlying code in which the R environment is constructed is available to anyone (and that means anyone) who wants to tinker with it, improve it, try out new ideas, and so forth. No more proprietary warnings or licenses for you to agree to, no more subscriptions or monthly payments. More about this in a note to you later in this chapter.

Instant Startup!

Want to jump right in and find out who has contributed to the creation and maintenance of R? If you've already installed R (perhaps you did before this class started or it's on the school's server you are working from), type

```
> contributors()
```

exactly as you see it here at the prompt in the R Console window, which is a "greater than" symbol that looks like this:

```
>
```

and press the Enter or Return key and take a look. If you have yet to install R, be patient—that's to come and you can then try the `contributors()` function.

Enter or Return or Nothing!

And remember this, R will just sit there and look at you unless you press the Enter or the Return key. R has no idea what you want it to do unless you press one of those keys that tells R to go ahead and get busy.

UPPERCASE Is Uppercase and lowercase is lowercase

There are all kinds of differences between computer programs based on different programming languages. R, which is based on a programming language called S, follows very strict rules about using uppercase (such as TRUE) and lowercase (such as true). For example, `contributors()` will get you a list of contributors, but `Contributors()` will get you an error message. We promise.

THE MINUSES OF USING R

But even with all those advantages above (and there are plenty more), R can be complicated and a bit intimidating. Some people just don't want to mess with anything that resembles a programming experience and that's okay. The help text can also seem cryptic at times until you become comfortable reading it (hint: check out the examples to see how the function works with real-life data). And while there are other minuses of R, most of those minuses are beyond the scope of this book.

R has developed some interfaces that make some R tasks close to a point-and-click experience like so many of the other programs that you use every day. In Chapter 3, we'll show you how to use RStudio, an interface that makes R easier to understand and to use.

OTHER REASONS TO USE R?

- R *can be installed on almost any computer* with any operating system, including Windows, Mac, Linux, and others as well.
- R *produces great graphics!* The quality is superior to those in commercial software such as SPSS, SAS, and Stata.
- R has *very few system requirements,* meaning that it will work on almost any computer that has enough memory to operate a word processor or a spreadsheet.
- Finally, R is *incredibly flexible,* which is a large part of the reason why R has become so popular. If you want to, you can dive deep into the very basis of the programming part of R and tweak your commands and results as you see fit. For nerds (especially) and non-nerds, this is a very big plus. But if you're a beginner, then no worries—R can take you as far as you want to go, and you will still see impressive results.

A SHORT NOTE TO YOU (AND TO YOUR INSTRUCTOR) ABOUT OPEN SOURCE (AGAIN!)

R is a particularly interesting tool for a whole variety of reasons, but there are a few more things that are important to know.

First, it is an open source programming environment, which means that it is open to ideas for how it might be changed for future users, and it is open to change by anyone who wants to do such. That's the really good thing about any open source product. And, most open source products are free.

Second, the nature of open source endeavors means that the programs change on a regular basis, so it is important for you to always check to see if the version of R that you have installed on your computer (which you will learn about in this chapter) is the most current.

Third, because it is open source, this means that R nerds and non-nerds alike are always making contributions that make the product better, easier to use, more convenient, and more powerful. So, at times, what you see on your computer screen may not match exactly what you see in the illustrations in the pages that follow—things change. But, relax. Nothing will be so different that you will not be able to follow and learn.

Fourth, the flexibility and open source nature of R also means that there are many different ways to do the same thing. And there will be more ways to do the same thing in the future. Our goal in this book is to show you what we think is one of several simple ways to reach a specific outcome. You may find other ways to accomplish a task and that would be great. We're here to show you how to use R, in one of many different ways, as an analytic tool.

And finally, R has many different uses and functions. Because it is undergoing an unending revision where there is always something new, you can take what you read here and do fine throughout the rest of the pages of *Statistics for People Who (Think They) Hate Statistics Using R*. Or, you can delve further and further, as far as you want to go, and explore the many facets of R on your own—and we encourage you to do so.

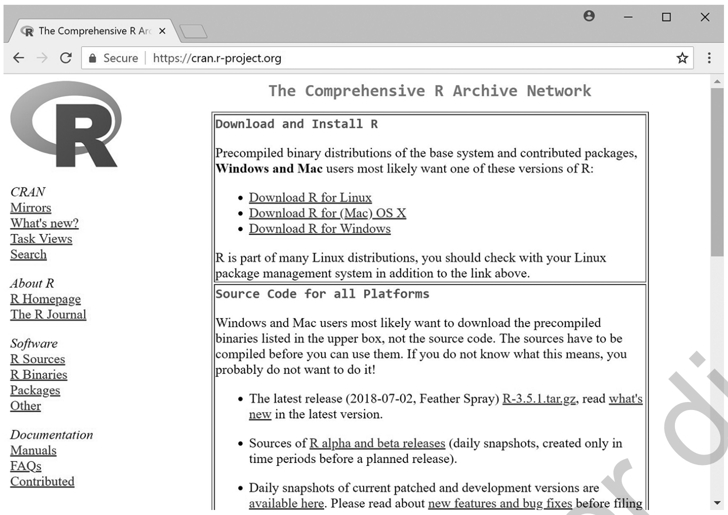
If you master the material in this chapter (getting R installed, picking an editor, and learning some new lingo) and the next, you will have a sufficient set of tools to master the chapters on specific statistical analysis techniques that follow.

WHERE TO FIND AND DOWNLOAD R

The first step in using R is, of course, to download it to your computer. Here you go:

1. Go to https://cran.r-project.org/, and as you can see in Figure 2.1, there is a link to install R on one of the many platforms that is supported.

FIGURE 2.1 ⬢ The opening screen for installing R.



The Comprehensive R Archive Network

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

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:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

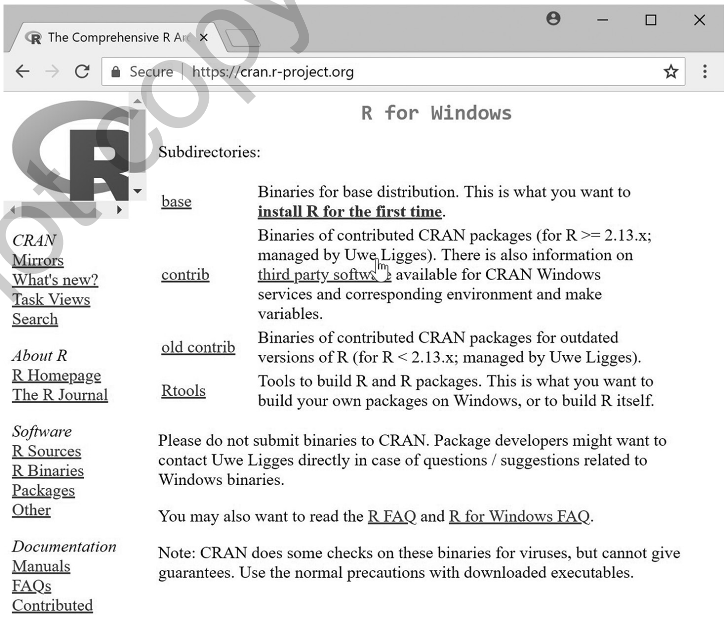
Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2018-07-02, Feather Spray) R-3.5.1.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing

Visit **edge.sagepub.com/salkindshaw** to watch an R tutorial video on this topic.

We'll be installing the Windows version, but as you can see, the Linux and Macintosh versions are readily available. Click on the version you want to install. As you can see in Figure 2.2, you can click on the "install R for the first time" link and R will start the download process of the R base (the most basic version and all you need).

FIGURE 2.2 ⬢ Installing R for Windows.



R for Windows

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Subdirectories:

base	Binaries for base distribution. This is what you want to **install R for the first time**.
contrib	Binaries of contributed CRAN packages (for R >= 2.13.x; managed by Uwe Ligges). There is also information on third party software available for CRAN Windows services and corresponding environment and make variables.
old contrib	Binaries of contributed CRAN packages for outdated versions of R (for R < 2.13.x; managed by Uwe Ligges).
Rtools	Tools to build R and R packages. This is what you want to build your own packages on Windows, or to build R itself.

Please do not submit binaries to CRAN. Package developers might want to contact Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the R FAQ and R for Windows FAQ.

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

From this point on, the installation is just like the installation of any Windows program (or whatever other platform or operating system you are using such as Linux or Macintosh). We installed R Version 3.5.1 because that was the most recent version when we were writing this book. You can install that version or a newer version. The examples, practice, and problems use functions that should work regardless of which version of R you install on your computer.

Finding R

You can get most of what you need at https://cran.r-project.org/ with the *cran* part of this web address representing Comprehensive R Archive Network (or CRAN), but there are many other places as well. Take a look at http://www.r-project.org/ to reach the R Project for Statistical Computing site that will also provide the latest news about R, new versions that are available, information about the R Foundation, and more. You can also find resources and help here.

Staying Current

You can go to https://www.r-project.org/ and find out what the most current version of R is by looking at the contents headed "Under News." You can always check how current your version is by looking at the RGui opening screen, the R Console, as shown in Figure 2.3 (which in this case was R Version 3.5.1). If your version is out of date, then a simple reinstall of R will fix that.

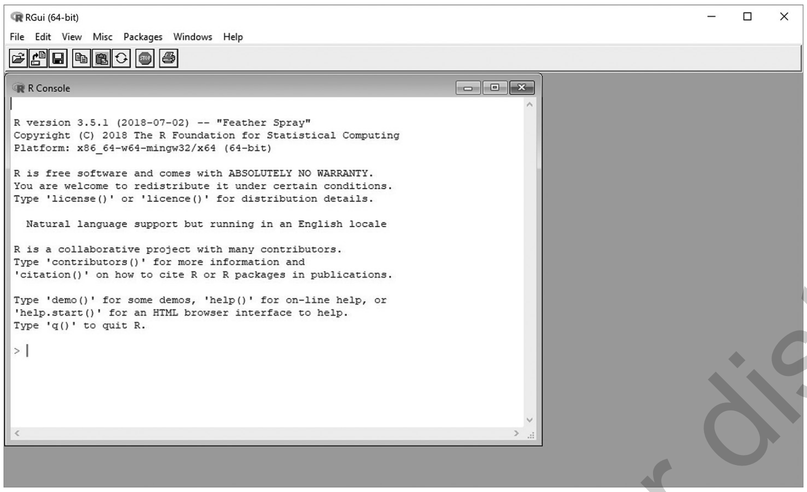
2. Once the file is downloaded, open it up and follow your computer's operating system instructions to install R. Once this is complete, an R icon (which looks like this R) should appear on your desktop. When the time comes to use R, you can double-click that icon and the RGui opening screen as you see in Figure 2.3 will appear. Go ahead and try it!

The Opening R Screen

This opening screen is called the **R Console** (look in the upper left-hand corner of the window). It's where you enter all the R command lines you want to execute as well as read in data and much more. In fact, it's R central, and as you will see in the next chapter, it's where you'll find results from your commands. At the top are menus, which are described in Appendix A. We are now going to spend a little time in the R Console to make sure R installed properly and introduce you to some new lingo. Later in this chapter, we will install RStudio and use RStudio throughout the rest of the book.

Let's first check to see if R is working properly on your computer. If you have launched R on your computer and can see the R Console window, then you have successfully installed R, the first step required for the chapters that follow. If R was

FIGURE 2.3 ⬢ The opening R screen.



```
R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 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.

  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.

> |
```

installed correctly, you should be able to easily add some numbers together in the R Console window. At the >, type in the following and hit the Enter or Return key:

```
> 2 + 3
```

R should have added those two numbers together for you, printing the result in the next line like this:

```
[1] 5
```

The "[1]" at the start of the line refers to one line of output. The 5 is the total of the two numbers you asked R to add. If you did that successfully, then R was successfully installed. For now, you can follow along with the next few examples as we learn some new terms and how to get help, or you can close the RGui and wait until we talk about RStudio later in this chapter. All commands we will show work in both RGui and RStudio as they both use R installed on your computer.

What Is a GUI?

Ever since modern programming came on the scene (actually around the mid-19th century with Ada Lovelace and her work on Charles Babbage's difference engine—whew), how information was "input" into computers has resulted in many different programming languages.

(Continued)

(Continued)

At one point, it was all *assembly language,* represented by the binary state of 1s or 0s. Then came a multitude of programs such as Basic, Unix, Fortran, Python, C++, and more and more. And, then came input devices such as the mouse and the stylus.

But still, we humans (and especially those of us who are not programmers) need as simple a way as possible to tell a computer what to do with the information we want manipulated, analyzed, transformed, and so on.

Enter the *graphical user interface* or *GUI* (yes, pronounced "gooey") that allows users to more easily understand what needs to be done and how. Windows is GUI based, as is the Macintosh user interface. And, now we bring you the RGui once you install R—two for the price of one! You could use R at the command prompt, but there are few reasons to use R that way. The RGui enables you to run commands and get answers. More user-friendly is RStudio, software that we will install later in this chapter. RGui is what you launch when you double-click on the R icon on your desktop.

When R loads up (R base is what's loaded) and appears on your computer, R is ready to use. The folks behind R believe that analyses you can do with R base are such convenient tools that to make them immediately available is the best bet to get you started using R as quickly as possible.

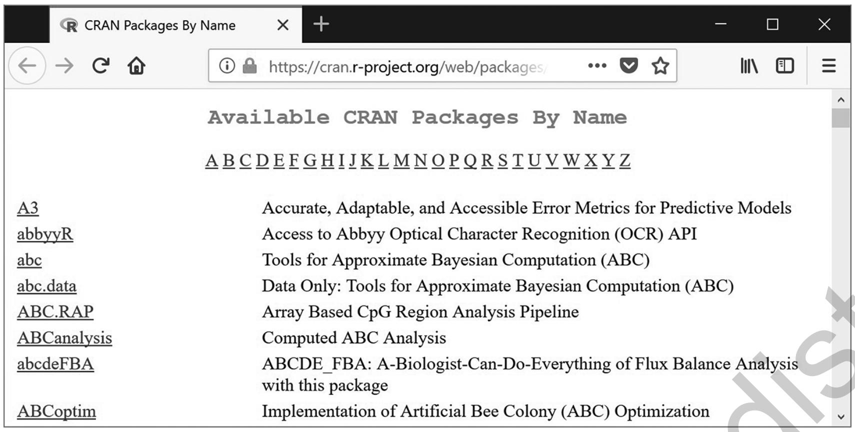
PACKAGES AND FUNCTIONS IN R

Earlier in this chapter, we talked about contributions that can make R better. You can use these programs by installing what are called **packages**. Packages (available in the thousands and thousands—no kidding) were written to perform certain types of tasks and allow you to easily locate, use, and update them when necessary. For example, the *stats* package, which we will use in every chapter, is automatically part of the R base and automatically available when you start R. Each package has a collection of functions and often some data you can use to practice, and the stats package alone has 37 functions that start with the letter *a*. But, what are functions?

Functions are, simply, a collection of commands that do some work for you. For example, if you use the sort function on a set of numbers, R will reorganize the numbers from smallest to largest. What if you wanted the numbers to start with the largest value and work down to the smallest? R can do that for you! We just need to learn how to tell R what we want.

How do you know what packages are available and what they do? The CRAN website stores all of the packages and maintains a list of all packages currently available for your use. If you go to any web browser and search for "R packages list," one of the top links should take you to this website: https://cran.r-project.org/web/packages/available_packages_by_name.html. The left side of the page is an alphabetized list of package names. The right side shows a short description for each package. The top of that webpage should look similar to Figure 2.4. And remember, new packages are always being added (even by the hour). In the next chapter, we will walk through finding out what packages you have on your computer.

FIGURE 2.4 ● The start of the list of currently available packages in R.



Available CRAN Packages By Name

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

A3	Accurate, Adaptable, and Accessible Error Metrics for Predictive Models
abbyyR	Access to Abbyy Optical Character Recognition (OCR) API
abc	Tools for Approximate Bayesian Computation (ABC)
abc.data	Data Only: Tools for Approximate Bayesian Computation (ABC)
ABC.RAP	Array Based CpG Region Analysis Pipeline
ABCanalysis	Computed ABC Analysis
abcdeFBA	ABCDE_FBA: A-Biologist-Can-Do-Everything of Flux Balance Analysis with this package
ABCoptim	Implementation of Artificial Bee Colony (ABC) Optimization

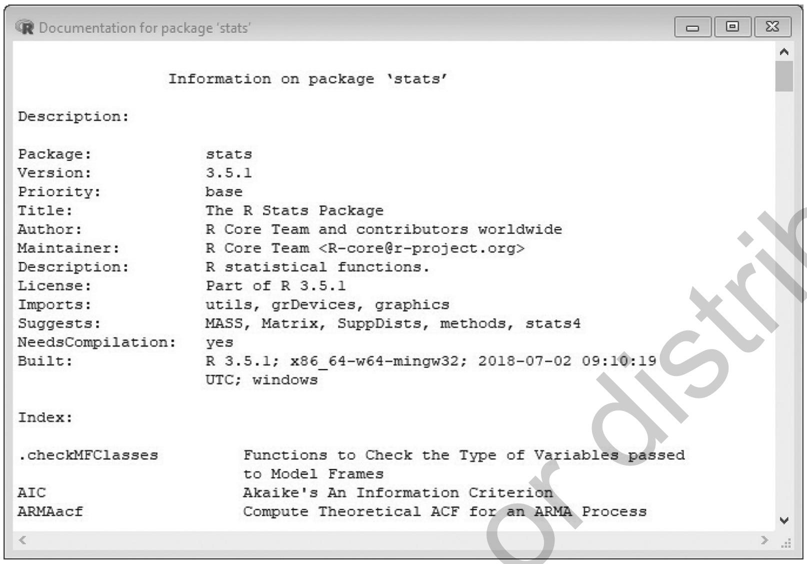
Want to find out what's contained in a package and what that package might allow you to do? Easy. If you are looking at the webpage above, find the package in the list you want to know more about and click on the name. From R we can use the help function. For help on the stats package, we would type the following in the Console at the > prompt:

```
> library(help = "stats")
starting httpd help server . . . done
>
```

and hit Enter. RGui should launch a web browser with some basic information about the stats package. Click on the Index link in the center bottom of the browser. You will see a window as shown in Figure 2.5 that explains what the package contains and a brief description of what each function does (and the version, which in this case is 3.5.1).

Almost any R function begins with a directive (such as data or help) followed by open and closed parentheses like this `()`, but within those parentheses is what you want the R function to operate on. So, for example, `help()` will give you a general window on help, and as you have seen earlier, `data()` will give you a general window on data sets available in R. The stuff that goes between the parentheses is called an **argument**.

FIGURE 2.5 ● Finding out what a package can do.



```
                Information on package 'stats'

Description:

Package:            stats
Version:            3.5.1
Priority:           base
Title:              The R Stats Package
Author:             R Core Team and contributors worldwide
Maintainer:         R Core Team <R-core@r-project.org>
Description:        R statistical functions.
License:            Part of R 3.5.1
Imports:            utils, grDevices, graphics
Suggests:           MASS, Matrix, SuppDists, methods, stats4
NeedsCompilation:   yes
Built:              R 3.5.1; x86_64-w64-mingw32; 2018-07-02 09:10:19
                    UTC; windows

Index:

.checkMFClasses         Functions to Check the Type of Variables passed
                        to Model Frames
AIC                     Akaike's An Information Criterion
ARMAacf                 Compute Theoretical ACF for an ARMA Process
```

A NOTE ABOUT FORMATTING

Throughout the rest of the book, R syntax and output will be presented like in the section above where we asked R for some help. Specifically,

- The ">" at the start of the R Console line where syntax is entered
- `Blue, monospaced font` mimicking what you type in R—the syntax
- `Black, monospaced font` mimicking what R returns—the output

When we instruct you to enter some syntax at the prompt, we are telling you to place your cursor after the ">" in the R Console and start entering the syntax from the example. Once each command is complete, just hit Enter or Return (depends on your keyboard—most of the time we will tell you to hit Enter), and in most cases, R will tell you what it thinks. Easy? Great!

BUNCHES OF DATA—FREE!

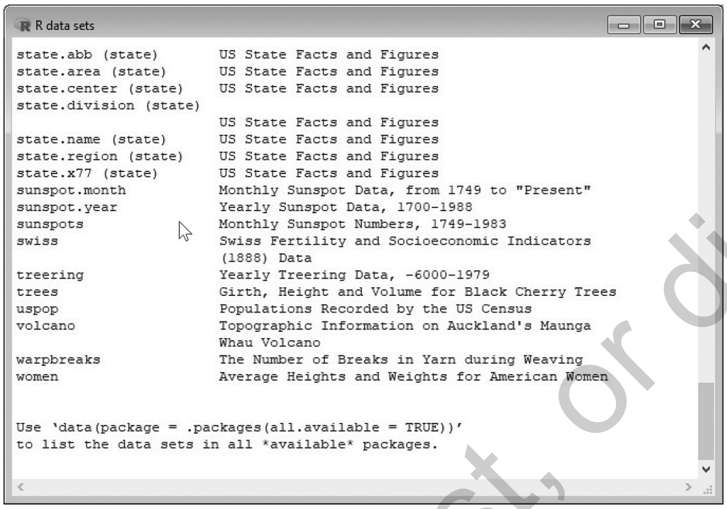
Just one of the very cool things about R base is that it comes with a ton of free data sets that you are welcome to explore. So, when you first install R, you will find that there are hundreds of data sets ready to read into R.

If you want to see them, enter

```
> data()
```

in the R Console. Look at Figure 2.6, which shows a subset of all the data sets that come with the downloaded version of R base. For example, you can see the data sets named *state.abb* and *women* (see women at the bottom of Figure 2.13).

FIGURE 2.6 ● A list of R data sets—free for the using.



```
state.abb (state)       US State Facts and Figures
state.area (state)      US State Facts and Figures
state.center (state)    US State Facts and Figures
state.division (state)
                        US State Facts and Figures
state.name (state)      US State Facts and Figures
state.region (state)    US State Facts and Figures
state.x77 (state)       US State Facts and Figures
sunspot.month           Monthly Sunspot Data, from 1749 to "Present"
sunspot.year            Yearly Sunspot Data, 1700-1988
sunspots                Monthly Sunspot Numbers, 1749-1983
swiss                   Swiss Fertility and Socioeconomic Indicators
                        (1888) Data
treering                Yearly Treering Data, -6000-1979
trees                   Girth, Height and Volume for Black Cherry Trees
uspop                   Populations Recorded by the US Census
volcano                 Topographic Information on Auckland's Maunga
                        Whau Volcano
warpbreaks              The Number of Breaks in Yarn during Weaving
women                   Average Heights and Weights for American Women


Use 'data(package = .packages(all.available = TRUE))'
to list the data sets in all *available* packages.
```

GETTING R HELP

Everyone, especially those starting out on such an ambitious endeavor as learning R, can use help once in a while. R offers a myriad of help, including menu options on the Help menu.

If you are a brand-new user, the best place to start is the FAQ on R (https://cran.r-project.org/doc/FAQ/R-FAQ.html). The FAQ lists question after question, starting with an Introduction through "What Is R," as well as add-on packages (same as the packages we mentioned earlier) and lots more.

This is the place to start and an even better place to become familiar with R just by spending some time browsing, paying attention to examples of how other people have used R to do what you want to do.

GETTING HELP ON HELP

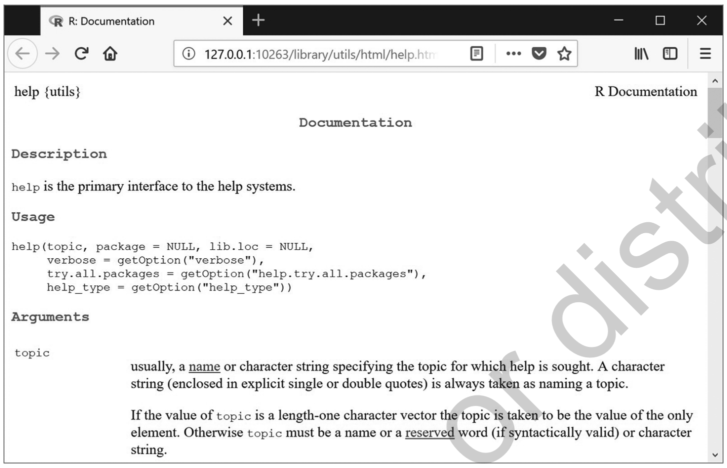
To get help on help, you just type

```
> help()
```

in the R Console.

And, when you type `help(help)`, you will see a window as shown in Figure 2.7, which provides you with information on what type of help R offers.

FIGURE 2.7 ● Help on help!



help {utils} R Documentation

Documentation

Description

`help` is the primary interface to the help systems.

Usage

```
help(topic, package = NULL, lib.loc = NULL,
     verbose = getOption("verbose"),
     try.all.packages = getOption("help.try.all.packages"),
     help_type = getOption("help_type"))
```

Arguments

`topic` usually, a name or character string specifying the topic for which help is sought. A character string (enclosed in explicit single or double quotes) is always taken as naming a topic.

If the value of `topic` is a length-one character vector the topic is taken to be the value of the only element. Otherwise `topic` must be a name or a reserved word (if syntactically valid) or character string.

Have a Question? Use ?

Another very easy way to get help is just to type a question mark and the function you need help on, such as `?mean` or `?plot`. If there's no help on whatever function you enter, R will tell you so. Use two question marks for searching keywords, like `??mean`. This will return a list of help pages that contain the word *mean*, including the mean function.

Once you start to use R, you can search for keywords, packages, or even functions. Now, sometimes these help windows can provide lots of information that is very concise without much explanation as to how to accomplish a particular task. But, it's a good beginning. As you use R more, this level of help will become more and more informative.

Finally, it may be overwhelming, but hundreds of sites on the Internet can help you, and here are just a few. There are also many other users who were also new to R, and they might have asked the exact same question you want to ask. Try typing your question into any search engine to see how it was answered. Sometimes seeing an example is all you need to get going again on your statistics adventure with R.

If no one has asked your question, you might want to ask your question to get help. Most R websites are community based, so whatever question you ask will often be answered by several experts who can guide you along the way. Remember, there are no stupid questions—just ones from folks who are not as experienced as others. Take advantage!

Getting Help with R (at https://www.r-project.org/help.html) is the R Foundation's help desk, maybe your first and last stop because it provides so much clear information about getting help with R. It's a treasure.

Stack Overflow (at https://stackoverflow.com/) is a site where developers (or coders or programmers or whatever you want to call them) can be asked questions. If the question is a repeat, they will tell you and usually provide a link to the original post, which will also contain an answer to your question.

Help for R (at http://search.r-project.org/) offers a ton of topics devoted to R. You may also want to start with R Wiki, which offers you a very good basic understanding of what R is and how it works.

RStudio, the next application you will install, has its own community website (at https://community.rstudio.com/). In fact, the RStudio community is one of the more welcoming sites to new R users. Check out the Welcome to the RStudio Community topic to get the most use out of this website and start giving back to others new to R.

How About an Example?

Okay, you've tried help and you have some good information to start with but you really crave an example of how a function might be used. In the R Console, type `example(mean)` and hit Enter. You'll see examples that will probably prove to be very useful.

SOME IMPORTANT LINGO

Every programming language has its own way of framing concepts and ideas, and some of the most important are listed and defined here.

An **object** is anything that is created in R. For example, if you create a data set that consists of 20 scores for one variable named *time,* that set of scores is called an object. Or if you run an analysis and save the results, the results are an object. If you compute the product of 4 times 3 and store the result, then the product, 12, is an object.

A **vector** is a collection of data that are the same type (in this case all numeric), such as 1, 2, 3 or 4564, 6545, 34356. A single number could be stored, and it would still be called a vector. You could also store a collection of words in a vector, such as months of the year like May, December, August, and February. If you try to put both numbers and text in a vector, R will treat the whole vector like text. Vectors are the most basic objects in R. Throughout *Statistics for People Who (Think They) Hate Statistics Using R,* we will create vectors from the various data sets that will be used for practice.

ATOMIC Vectors

There are six types of information that you can store in a vector in R. But what does *atomic* mean? Vectors can store information, and how it is stored depends on its type, or atom, the smallest unit that you as an R user will enter and manipulate.

(Continued)

(Continued)

- Logical: TRUE or FALSE
- Integer: a whole number
- Double: a number that can contain a decimal point
- Complex: a number that contains an imaginary part (yes, this is a real thing!)
- Character: text that is also referred to as a string
- Raw: bytes of information expressed as hexadecimals

A **function** is a tool that R uses to complete some kind of operation. For example, the `mean()` function provides an average of a set of values by adding up the values and dividing by the number of observations, and `plot()` uses data to create a graph.

An **argument** is what a function acts on. So, if you see the function `mean(x)`, and the argument is `x`, R will produce an average for whatever you defined as the object named x.

And a package is a collection of functions such as the *stats* package (which we will use often), but some others are named such things as *survival* (which does survival analysis), *class* (which helps classify information), and *graf* (a graphics package). Want to see (in a way different than we did before) all the packages that have been installed in the version of R that you downloaded? In the Console after >, type

```
> library()
```

and then press Enter. You will see a listing of R packages that are currently available on your computer with this download of R.

A **data frame** is collection of information (vectors) that is arranged in rows and columns (and can just be a single row or a single column) and can contain different types of variables such as you see (both text and numeric).

Weight	Items Completed	Age
154	65	24
210	72	35
156	77	41
151	81	34
98	60	12

Data Frame or Matrix?

Another rows-and-columns organizational tool in R is called a matrix, which is a combination of rows and columns that contains values all of the same type. You can have all numbers, all text fields, or even a collection of true/false values.

A **list** is an object that contains many different types of objects within it. For example, you could create a list that would hold a data frame as one element, a vector of means for your data frame variables in the second element, and a vector of standard deviations for your third element. To make things even more confusing, if you look up the definition of a list on r-project.org, the definition for list will tell you that it is a vector. What? The official R language distinguishes between atomic vectors and list vectors. We won't spend time on lists in this book, but if you stick with R, you may find you like using them to store your objects.

Here are some of the major statistical functions we'll use throughout *Statistics for People Who (Think They) Hate Statistics Using R* and the chapters in which they will appear. Some of them have already been downloaded along with R.

Chapter	Function	What It Does
2	`help()`	Find information on a package or function so you know how to use it
3	`read.csv()`	Import a data set into R for use
4	`mean()`	Calculate the arithmetic mean of a vector of numbers
5	`sd()`	Compute the standard deviation of a vector of numbers
6	`plot()`	Draw a picture of your data
7	`cor()`	Calculate the correlation between two variables
8	`alpha()`	Calculate a statistic of reliability for a measure
10	`scale()`	Change the mean and possibly standard deviation of your vector
12	`pnorm()`	Determine the probability for a *z* statistic
13	`t.test()`	Compute the *t* test statistic for independent samples
14	`t.test()`	Compute the *t* test statistic for paired samples
15	`aov()`	Calculate analysis of variance
16	`Anova()`	Obtain Type III sum of squares results
17	`cor.test()`	Calculate the correlation and get the probability of the statistic
18	`lm()`	Estimate a linear model for regression
19	`chisq.test()`	Calculate a chi-square statistic

OK. Let's review what you have learned so far.

1. R is a programming environment often used to assist in the analysis of data.
2. It is an open source package that is incredibly flexible but also requires lots of attention to detail.
3. When you download R, you get a ton of additional resources including data sets, packages, and extensive help.
4. Case (upper and lower) counts, so be sure and enter a function correctly and exactly.
5. R is a treasure chest of tools. The more you use it, the more you will be able to maximize its power.

RStudio

With the install of R, you also received RGui, the default graphical user interface that comes with R. On Windows, RGui can be challenging to use, even for experienced R users. Instead, we are going to use RStudio from the company RStudio (https://www.rstudio.com/). The RStudio interface is actually called an *IDE* or an *integrated development environment*. It accesses R on your computer like RGui does, but RStudio contains some features that will make life easier as you learn.

RGui or RStudio?

RStudio uses the same commands we used in the sections above when we introduced you to `help()`, `data()`, and `library()`. RStudio is just another (and easier) way to use R. So, all the R commands you will ever learn will work in RStudio. RStudio uses R that you installed on your computer, only dressed up.

Have You Already Installed R?

RStudio uses R that is already on your computer. If you decided to wait until later to install R, you have come to later! Go back to "Where to Find and Download R" and follow the steps to install R on your computer. Then, come back here to follow the steps to install RStudio.

WHERE TO FIND RSTUDIO AND HOW TO INSTALL IT

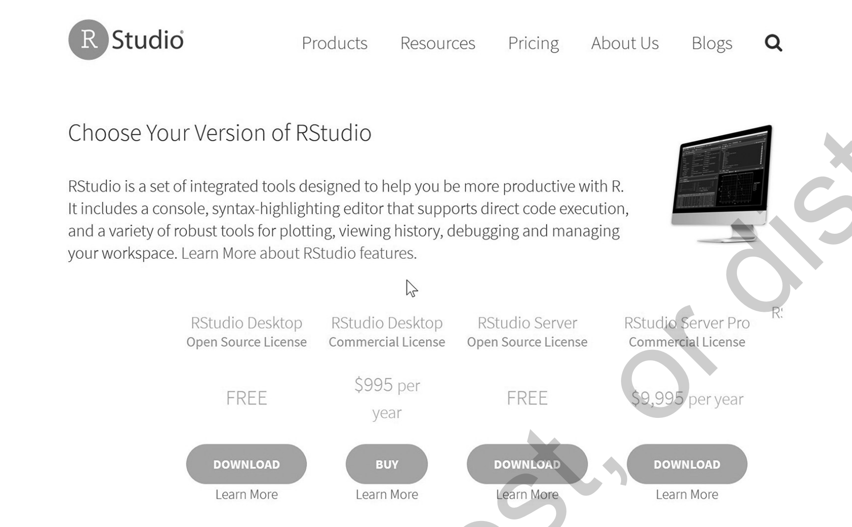
Here are the steps to follow to install RStudio:

1. Go to https://www.rstudio.com/products/rstudio/download.

2. As you can see in Figure 2.8, you have several choices as to the version of RStudio you want to install, including a free version for the individual computer, referred to as RStudio Desktop. Click the DOWNLOAD button for the RStudio Desktop for either the Open Source or Commercial License.

Visit **edge.sagepub.com/salkindshaw** to watch an R tutorial video on this topic.

FIGURE 2.8 ● The RStudio installation site.



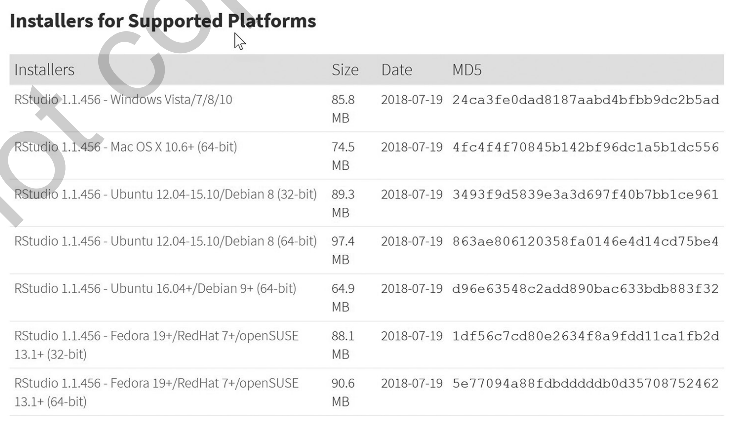
Choose Your Version of RStudio

RStudio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, and a variety of robust tools for plotting, viewing history, debugging and managing your workspace. Learn More about RStudio features.

RStudio Desktop Open Source License	RStudio Desktop Commercial License	RStudio Server Open Source License	RStudio Server Pro Commercial License
FREE	$995 per year	FREE	$9,995 per year
DOWNLOAD	BUY	DOWNLOAD	DOWNLOAD
Learn More	Learn More	Learn More	Learn More

3. As you see in Figure 2.9, you will be given several options regarding which operating system or platform you are using. Double-click the one that fits your platform (such as RStudio Windows or RStudio Mac OSX).

FIGURE 2.9 ● Selecting a version of RStudio to install.



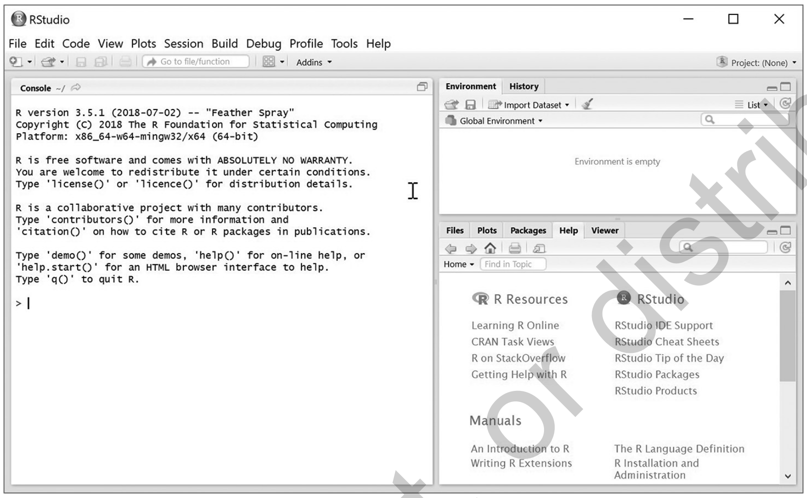
Installers for Supported Platforms

Installers	Size	Date	MD5
RStudio 1.1.456 - Windows Vista/7/8/10	85.8 MB	2018-07-19	24ca3fe0dad8187aabd4bfbb9dc2b5ad
RStudio 1.1.456 - Mac OS X 10.6+ (64-bit)	74.5 MB	2018-07-19	4fc4f4f70845b142bf96dc1a5b1dc556
RStudio 1.1.456 - Ubuntu 12.04-15.10/Debian 8 (32-bit)	89.3 MB	2018-07-19	3493f9d5839e3a3d697f40b7bb1ce961
RStudio 1.1.456 - Ubuntu 12.04-15.10/Debian 8 (64-bit)	97.4 MB	2018-07-19	863ae806120358fa0146e4d14cd75be4
RStudio 1.1.456 - Ubuntu 16.04+/Debian 9+ (64-bit)	64.9 MB	2018-07-19	d96e63548c2add890bac633bdb883f32
RStudio 1.1.456 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (32-bit)	88.1 MB	2018-07-19	1df56c7cd80e2634f8a9fdd11ca1fb2d
RStudio 1.1.456 - Fedora 19+/RedHat 7+/openSUSE 13.1+ (64-bit)	90.6 MB	2018-07-19	5e77094a88fdbddddb0d35708752462

4. Proceed as you install any new application for your operating system and you will shortly see the RStudio icon on your desktop, which looks like this R. In this case, a shortcut was placed on the Windows desktop.

5. Double-click the RStudio icon and you're ready to go with the RStudio opening screen as shown in Figure 2.10.

FIGURE 2.10 ● The RStudio opening screen.



```
R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 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.

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.

> |
```

TAKE RSTUDIO FOR A TEST RIDE

Recall, after installing R, we tested adding 2 and 3 in the R Console window of the RGui. Let's do the same thing with RStudio. In the R Console window, type

```
> 2 + 3
```

and hit the Enter or Return key. Like before, you should see

```
[1] 5
```

to indicate one line of output and the result of your addition is "5."

If you have gotten this far, you have installed both R and RStudio and tried some simple commands in the R Console. You have also started to learn some R lingo. You probably saw some words that you know but now know their definitions in R.

We are now going to spend a little more time with RStudio, introducing you to the menus. Pay attention to the big ideas. You will see more new terms and phrases on the menus, like "Workspace" and "Set Working Directory." By the end of Chapter 3, you will know what these mean along with a dozen other words and phrases, so don't worry about it. Right now, you are trying to learn how things are organized, so if you need to find something in the coming chapters, you will have an idea of where to look.

Windows or Mac

What if you have an older Mac OS and could not install RStudio? Don't worry. Students using the Mac OS have told us that RGui on Mac is nice to use. And in Chapter 3 and the rest of the book, where possible, tasks that can be done by clicking on a menu or entering a command in the R Console will be demonstrated both ways, so you will be able to follow along using the commands.

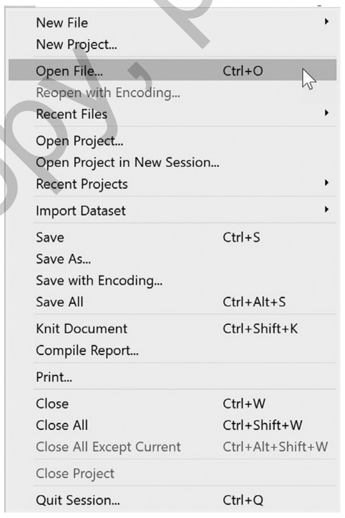
ORDERING FROM RSTUDIO

As with any GUI, RStudio comes with a series of menus that contain a variety of options, kind of like any menu in a restaurant. Here's a summary of RStudio's 11 menus and what you can find on each one.

File

By using options on the File menu, you can do such things as start a new project, save the data you are working with (very important), access recent projects, close a project you are working on, and quit RStudio altogether. The File menu shown in Figure 2.11 is where you will spend a considerable amount of time as you develop sets of RStudio commands (called *scripts*) and want to continue working on them.

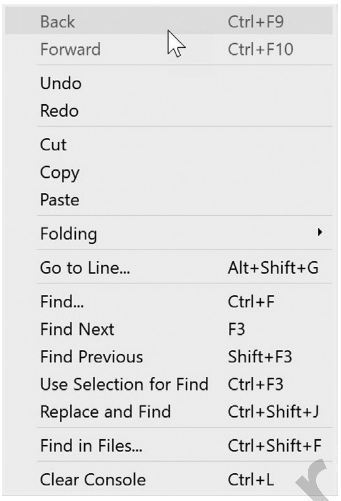
FIGURE 2.11 ● The File menu.



Edit

Our first try at creating a set of commands or a script often will need some touchups, and that's where the options on the Edit menu (shown in Figure 2.12) come in. Here you can do the usual Cut, Copy, and Paste but also clear the entire Console with one menu item click, Redo and Undo what you have just entered, and search for specific commands.

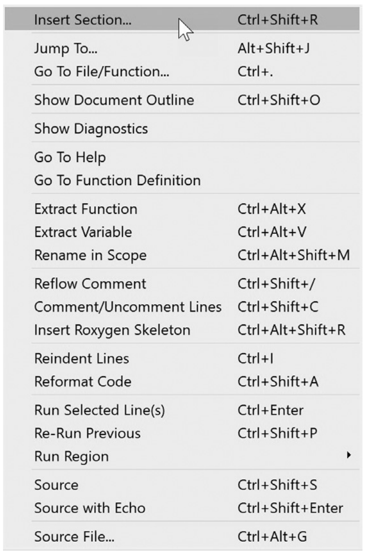
FIGURE 2.12 ● The Edit menu.



Code

Coding (the focus of the Code menu), in any programming environment, is the name of the game and such is surely the case with RStudio. In Figure 2.13, you can easily insert an entire section of script, find the definition of a function, have RStudio run only those lines of script you want to, and examine the Source File or the set of commands with which you are working.

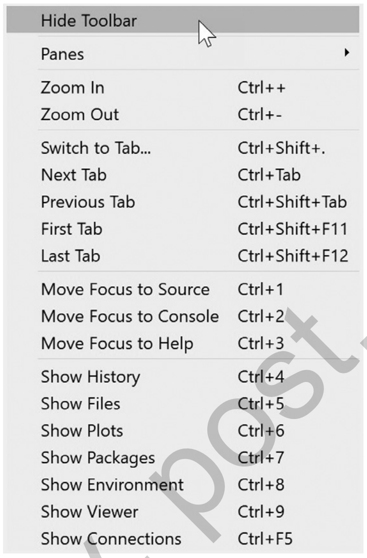
FIGURE 2.13 ● The Code menu.


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
Source Ctrl+Shift+S
Source with Echo Ctrl+Shift+Enter
Source File... Ctrl+Alt+G

View

If you want to rearrange the way that RStudio panes are displayed on your monitor, commands on the View menu shown in Figure 2.14 are where you want to be. Additionally, you can increase or decrease the size of elements on the screen; show files, plots, and packages (and more); and use the cool Tab key to help you insert commands (more about that soon).

FIGURE 2.14 ● The View menu.



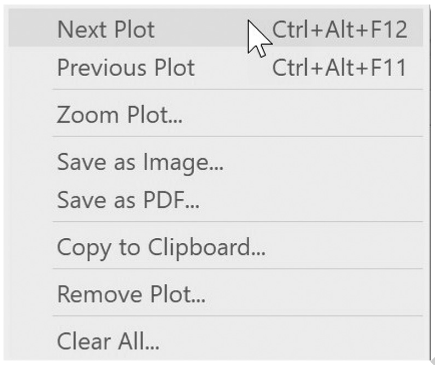
Rearranging Panes

If you find that you want to rearrange the way that RStudio panes appear on your screen, click View→Panes→Pane Layout and you will see an Options dialog box. There you can select which of the four RStudio panes you want in each of the four possible positions. You can even check the spelling or add a custom dictionary.

Plots

For those of us who are visual learners and for illustration purposes, RStudio provides an easy-to-use and powerful set of tools to create and manipulate plots through the Plot menu. You can save the plots you create as an image, in PDF or other formats, and copy the plot to the clipboard, where it can then be easily pasted into another application such as Word. See the lower right pane of RStudio for a visual of some of what the Plots menu can do after you select Next Plot as shown in Figure 2.15.

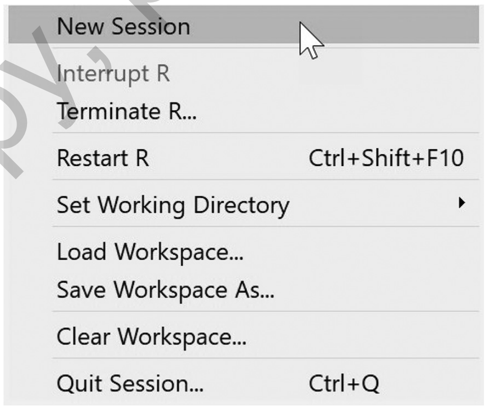
FIGURE 2.15 ● The Plots menu.



Session

Time to start a new session? Set the working directory, or clear the workspace? What do these things even mean? Stay tuned! We will talk about all of these things in the next chapter. Check out the Session menu shown in Figure 2.16.

FIGURE 2.16 ● The Session menu.



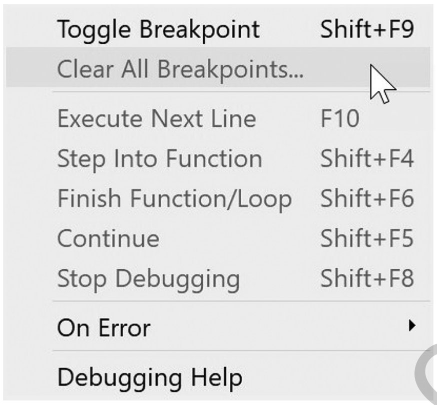
Build

You may not spend lots of time in the Build menu, but if you are looking to build a package or develop some kind of add-on to RStudio, this is where you want to be. The Build menu has one item on it—Configure Build Tools—so we will skip showing a picture of this menu.

Debug

Once that package is developed or a script developed, you want to use the debug tools found on the Debug menu shown in Figure 2.17.

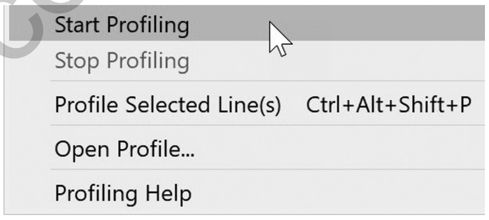
FIGURE 2.17 ● The Debug menu.



Profile

The Profile menu shown in Figure 2.18 is for the more advanced of us who are creating scripts, developing packages, and more. This menu helps the RStudio user to understand how R spends its time and resources in an effort to make things operate more efficiently.

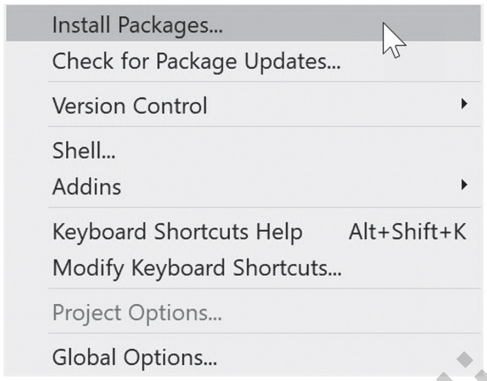
FIGURE 2.18 ● The Profile menu.



Tools

What's a good toolbox without tools? RStudio provides several on the Tools menu shown in Figure 2.19 that allow you to do such things as install packages, check that the packages that are installed are up-to-date, use RStudio add-ins, and define global options such as setting the default working directory.

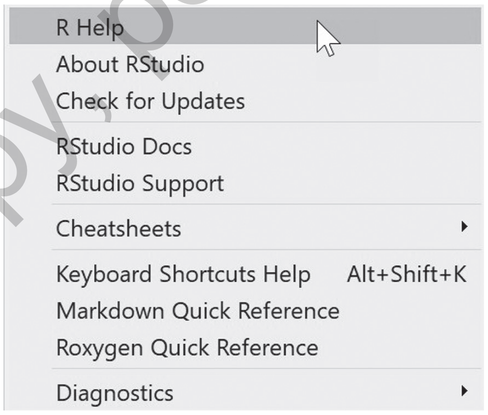
FIGURE 2.19 ⬢ The Tools menu.


Install Packages...
Check for Package Updates...
Version Control
Shell...
Addins
Keyboard Shortcuts Help Alt+Shift+K
Modify Keyboard Shortcuts...
Project Options...
Global Options...

Help(!)

And, finally, RStudio offers a Help menu filled with options that can get you back on track. As shown in Figure 2.20, you can check for updates (and you should do this every so often), access RStudio Support, find out about keyboard shortcuts, and even diagnose what might be happening with the scripts you want to run.

FIGURE 2.20 ⬢ The Help menu.



Summary

Well, some congratulations are in order.

Once you have finished this chapter and completed the Time to Practice exercises that follow, you will have installed R and RStudio, learned some new terms, and performed some simple arithmetic in the R Console window. Now, it takes practice using R and, most of all, just having some fun exploring different functions and seeing the results. Keep in mind that working with a study buddy (as we talked about in Chapter 1) is invaluable, especially when such details as remembering case and including quotes are concerned. In Chapter 3, we'll move on to learning more about R, RStudio, and its panes; revisit packages; start using R as a calculator (either the most or least expensive calculator you have ever owned); and enter data.

Time to Practice

1. What is R, and what are some of the advantages and disadvantages of using it for statistical analysis?
2. Describe two ways to get help in R.
3. Objects in R:
    a. What type of data can you store in a vector?
    b. Can you store both numbers and text in a vector?
    c. How are functions and packages related to each other?
4. In RStudio, run some of the functions we showed in the chapter.
    a. `library(help = "stats")`
    b. `data()`
    c. `help(help)`
    d. `library()`
5. Where did the help text show up once you tried the command for `library(help = "stats")`?
6. What is RStudio, and how does it differ from R?

Student Study Site

Get the tools you need to sharpen your study skills! Visit **edge.sagepub.com/salkindshaw** to access practice quizzes and eFlashcards, watch R tutorial videos, and download data sets!