R AND RSTUDIO®

INTRODUCTION

This chapter begins our exploration of social science data and statistics using R. While this book may be used as a companion with another statistics resource, it can also stand alone as a basic introductory statistics guide integrated with instructions for how to utilize the components of R to calculate statistics, produce graphical and tabular output, and to interpret these elements that are produced.

Here, we introduce readers to the object-oriented framework in R and RStudio and provide instructions on how to download and install the applications. We utilize screenshots to identify and explore the four primary windows within R/RStudio and discuss details about their basic functions. Although this book does not make use of user-created packages available in R/RStudio, we demonstrate how to search for, download, and install or uninstall packages as needed. We also introduce the **tidyverse** package and provide examples of how to read R user-created package documentation. We then provide step-by-step instructions for opening data in .txt, .csv, and URL formats and saving the data files for later use.

STATISTICAL SOFTWARE OVERVIEW

There are many options available for individuals seeking computer software to perform statistical analysis as well as data management. Some of the more popular software packages include IBM's SPSS Statistics, the SAS System, and Stata. R is a type of computer programming language. This computer programming language can be used to organize and analyze data and employ statistical techniques. Using R, one can also utilize basic calculations in the same way that most computer programming language can perform calculations such as addition, subtraction, multiplication, and division.

By almost all accounts, R is much more difficult to use than the major statistical packages available for purchase. Its interface is neither pleasant nor nimble. Its output is not

as elegant. Moreover, there is a much steeper learning curve with R than with, say, SPSS Statistics. To make matters worse, for a student learning statistics while having also to deal with the particulars and peculiarities of R, this might seem like a daunting task, especially by comparison to learning statistics with a popular commercial software package (e.g., SPSS Statistics). Also, SPSS Statistics is far more widely used in most classroom environments. However, once the learning curve is overcome, R is a very powerful statistical tool. Moreover, R is gaining in popularity.

Having said all that, it has one tremendous advantage over SPSS Statistics, the SAS System, Stata, and many other packages. Its price simply cannot be beat—R is free. That alone is enough to merit an investigation into its capabilities and explore opportunities that this type of programming language may offer for perhaps more flexibility than other software packages in particular applications. That is to say that R can be used for basic, traditional statistical analysis, but it can also be used for more advanced and less mainstream statistical analysis.

As you begin and continue further along on your journey of understanding statistics and the R software, you will begin to understand how to make things happen, things that may not seem clear or intuitive at all when you first begin. Similarly, you will also gain awareness of the kinds of things that can be done with R and how to facilitate those with a design that you can lay out in a "logical" way. To gain this understanding of the logic of the programming will take some time and understanding, particularly of the basic functions and operations with the R programming language. To be fair, while SPSS Statistics and the others are more intuitive and have more bells and whistles, there are times when using a programming language makes things work more smoothly. One example of this would be if you need recode multiple variables in a similar fashion. With the complex GUI (graphical user interface) in SPSS Statistics, for instance, you would need to go through the process by point-and-click steps each time, for each separate variable. With programming languages, it's typically a far more elegant process involving copying and pasting with some editing of the pasted code.

Granted, this can be done using the "Syntax" option with SPSS. (This option is not available for the limited use student version.) Many users either do not know how to do that or find it difficult to switch back and forth. With R, you will become accustomed to the programming approach and, despite the steep learning curve, you will no doubt be rewarded with knowledge of faster and easier ways to do some of the more basic statistical and data management tasks.

In addition, many skilled users of R feel that typing the code to produce their output is almost always faster than dealing with the GUI. (Note that there is a GUI with RStudio, to be addressed in the next section, but it is not intended to be at the same level of those of the popular commercial statistical software packages. Its purpose will be revealed shortly.) For the complex GUIs to accomplish what is produced with programming code, a series of windows, check boxes, and other visual tools are used. The process of going through these, while much more intuitive, actually takes more time. The point here is to indicate that not only is R free, but after you become more familiar with the application, R might actually suit your needs better than other programs, particularly if you are or become a regular user of statistical software.

To that end, let us first download and install the software.

DOWNLOADING R AND RSTUDIO

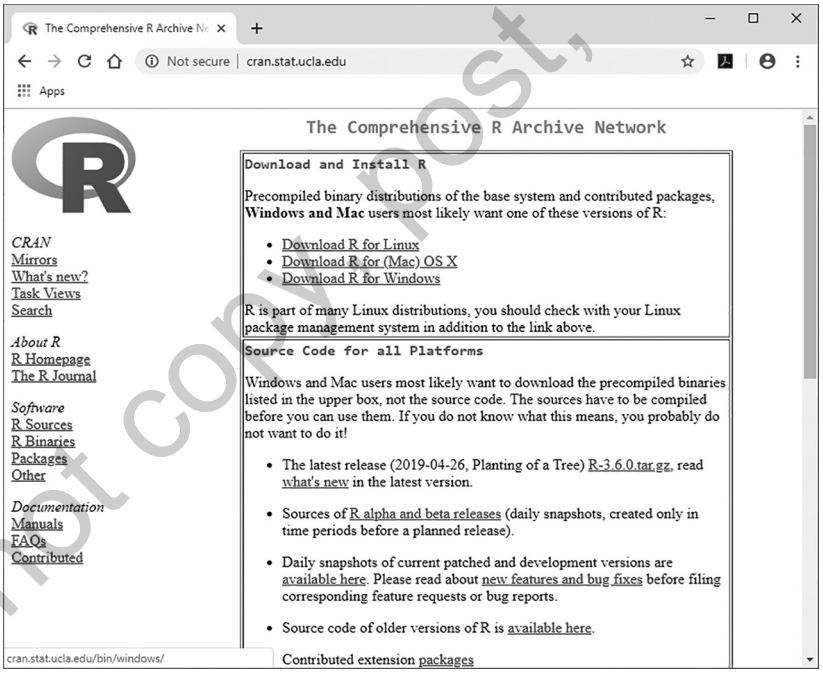
R

To download R software, visit this website: https://www.r-project.org/

Choose the download CRAN selection on the menu to the left. CRAN means Comprehensive R Archive Network. There will be a list of "mirror" sites, representing servers around the world where you can download the free software. Choose the one closest to your location to get started. Then, select the appropriate version for your computer (e.g., Linux, MacOS X, Windows). Once you have downloaded the package, follow the instructions and install the software on your computer.

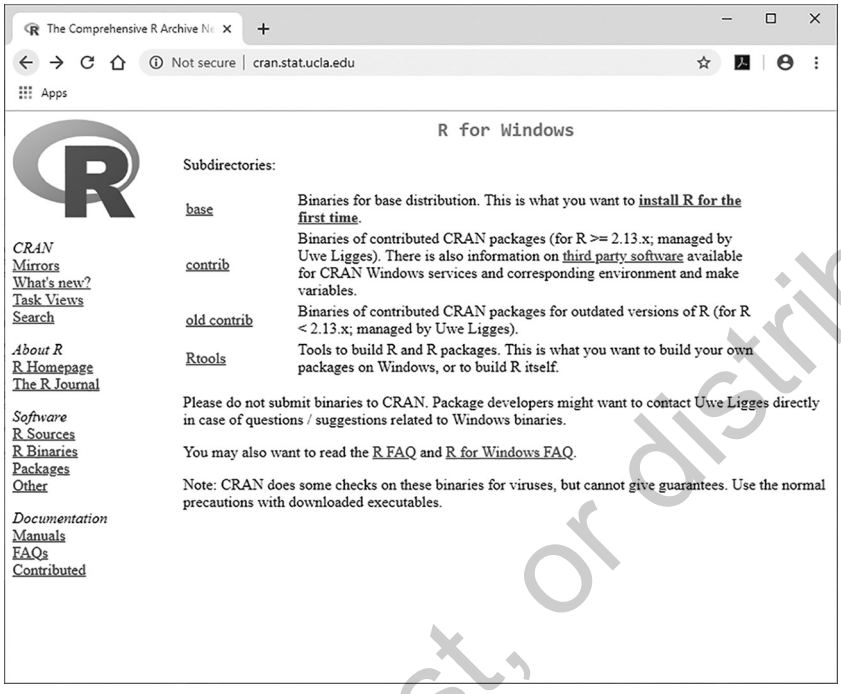
Once you click the download link, you will be presented with a list of server locations from where you can download the software.

Since we are downloading the application in Los Angeles, California, we chose the link at UCLA (University of California, Los Angeles). After selecting the closest mirror link, you will be presented with the following window, where you can select the type of computer on which you will be using R:



The last step in the download process for R is to choose the correct installation package. First, note that we will download the base package.

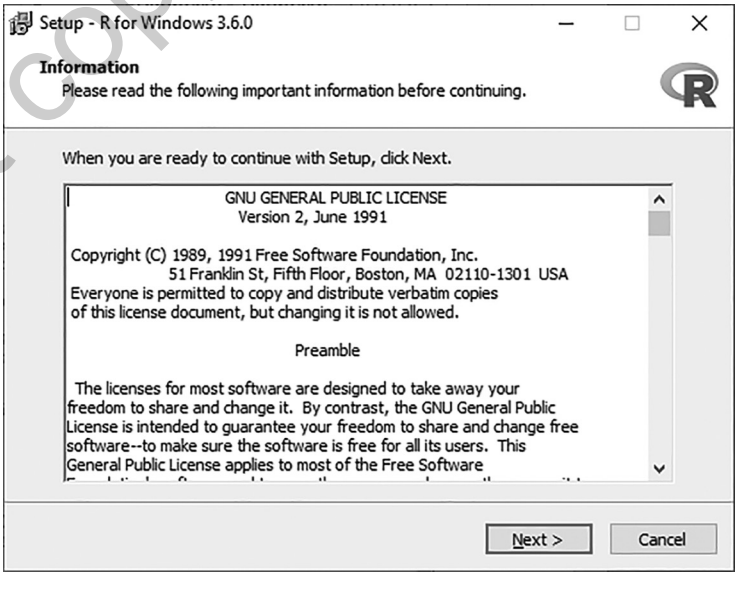
Tip: While you will notice that, in our example, the latest download version is 3.6.0, the version number will change as updated versions are released.



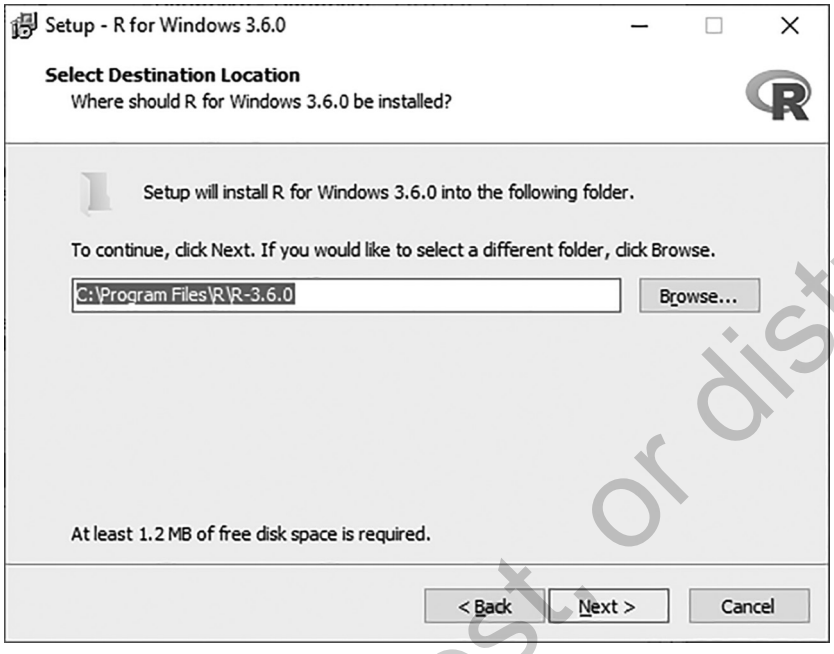
Now that you have downloaded the software, you will need to install it on your computer.

Follow these steps:

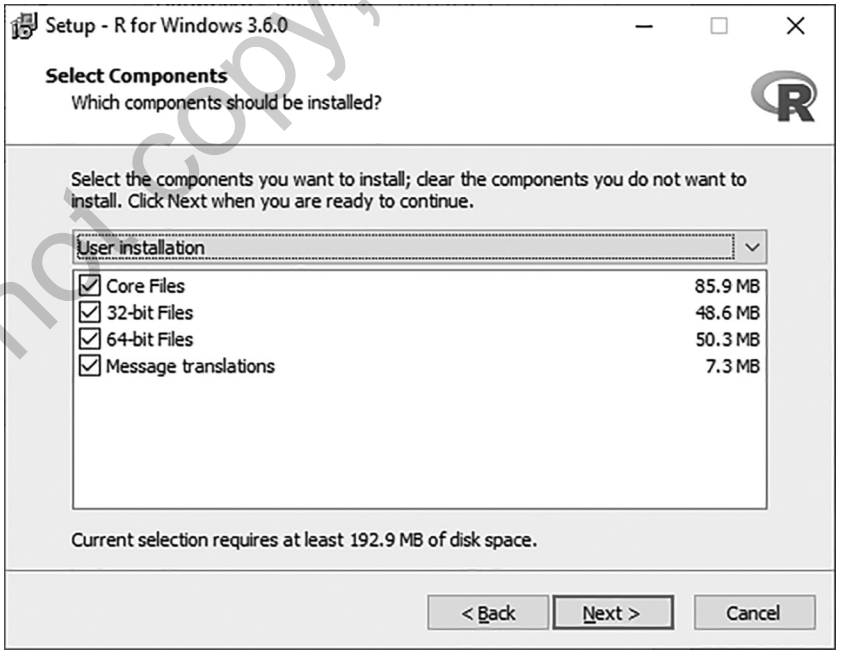
1. Click the icon for "R-3.6.0.pkg".
2. The following window will appear—read the license agreement and select "Next >".



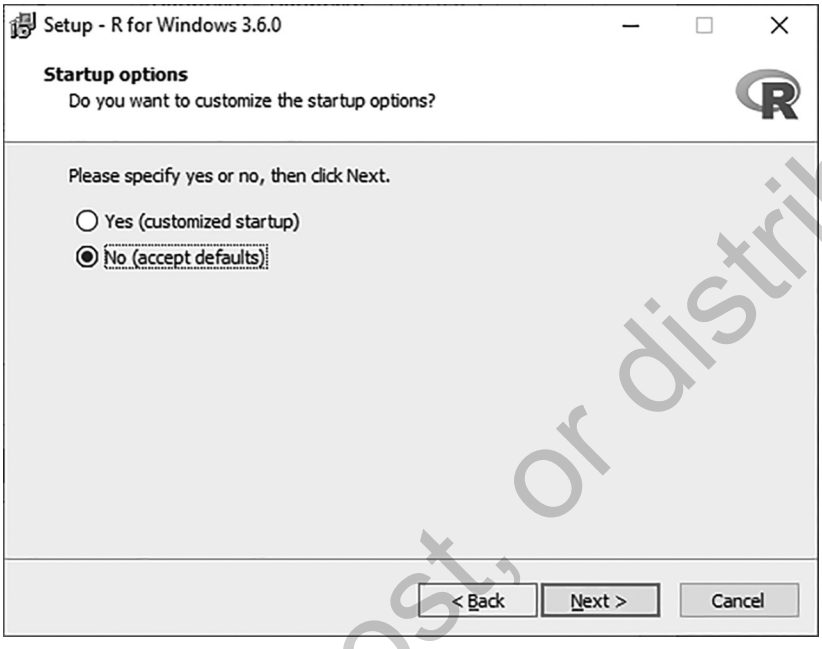
3. Choose the directory where you would like to install R; then click "Next >".



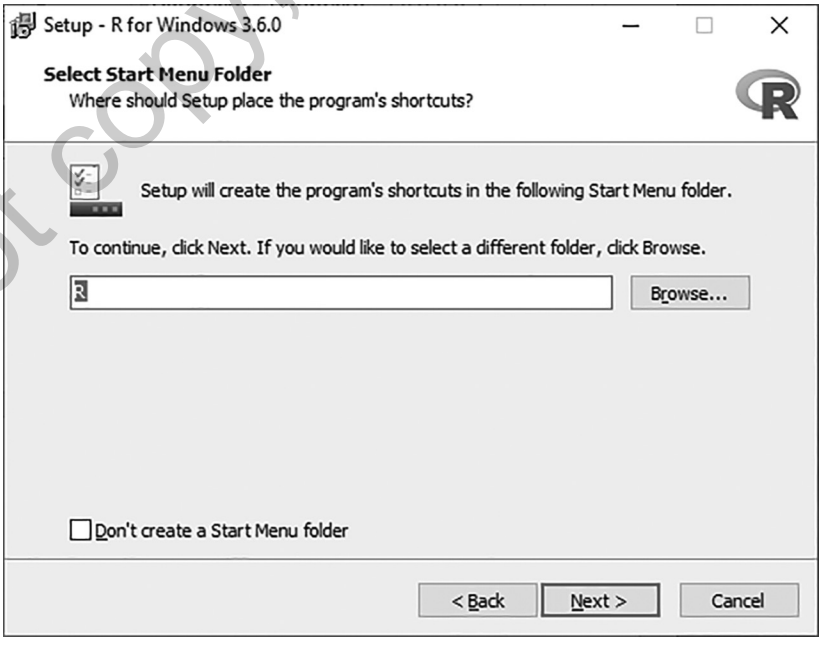
4. Select components to install. You may choose to install only 32- or 64-bit files, depending on the machine you are using. For this example, we will download both. Select "Next >".



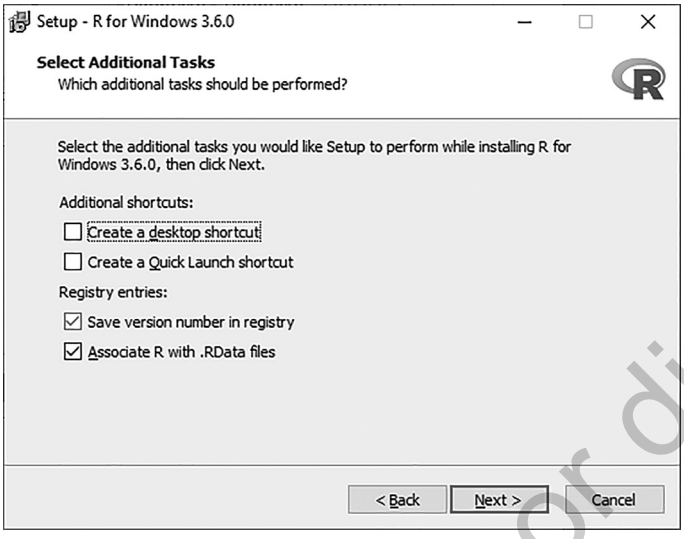
5. You will have the option to customize startup options. As a beginning user, we recommend to click "No," which accepts the default options. Then, click "Next >".



6. Determine in which folder of the startup menu you would like to have R shortcuts located and then click "Next >".

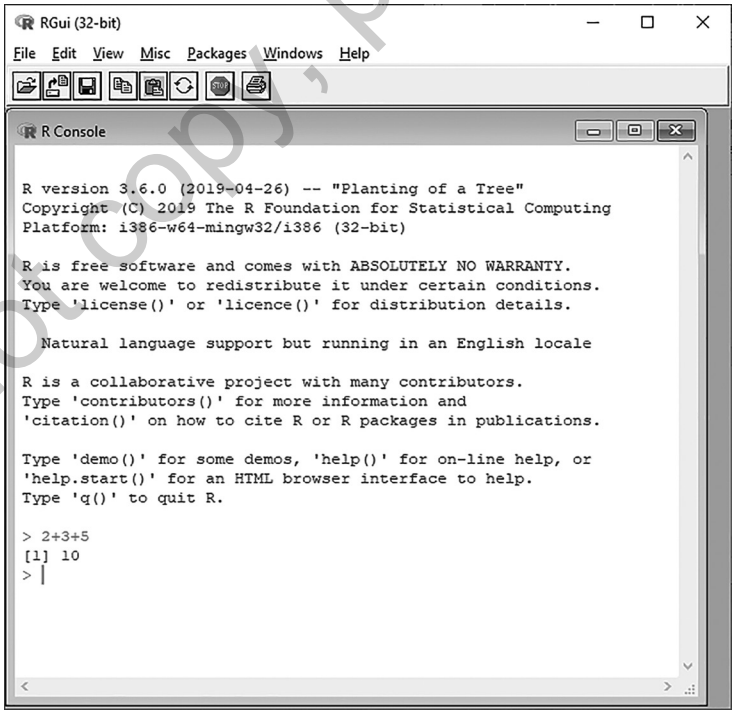


7. Make your final selections and then click "Next >" to install R.



8. You are now ready to launch R.

Once you launch R, you will be presented with the R console. There is a prompt where you are able to enter commands or instructions to tell R what you would like it to calculate or create. In the console image below, you will see an example using simple arithmetic: "2+3+5" was entered and then R returned "10".



In its most basic form, R can function like a calculator, but our purpose of downloading R goes far beyond just simple calculations. This, however, is a good opportunity to become comfortable entering commands/instructions into R's console. For example, if you want to calculate the average of a group of numbers, you can enter them, within parentheses, and then divide by how many numbers there are. For example,

```
(5+4+9+10+6+21+1)/7
```

```
8
```

R added the numbers together and then divided by 7, yielding the mean for the distribution—8.

RSTUDIO

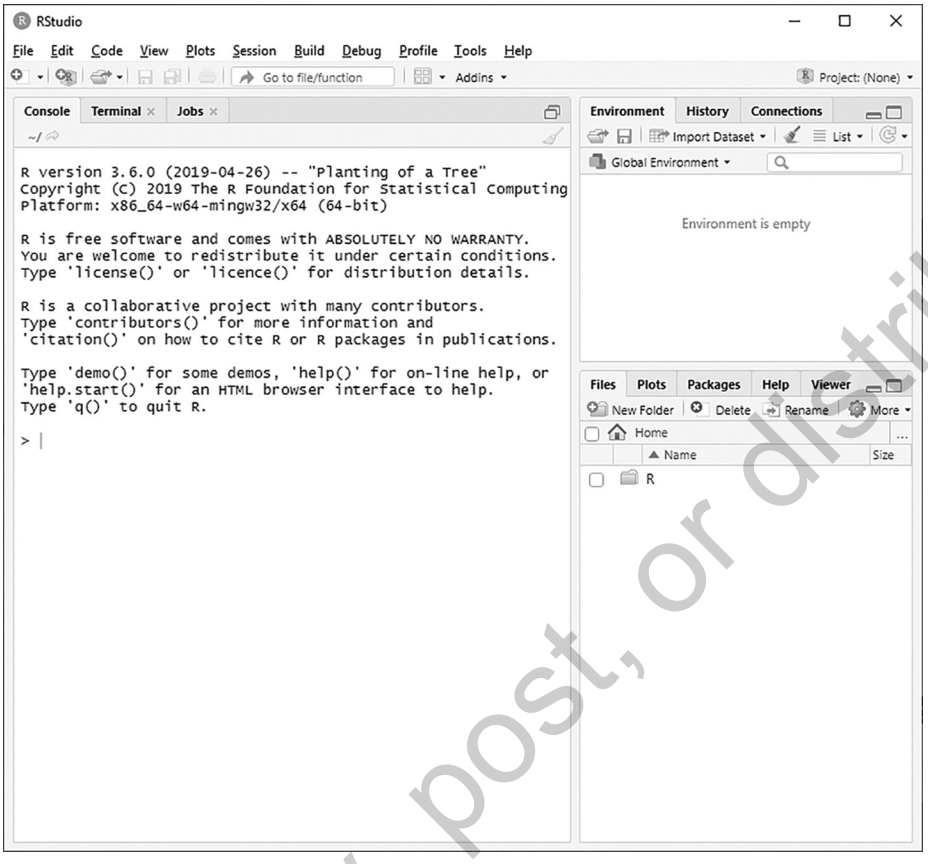
Before installing RStudio, you must install R ("base R"). We have already done this in the previous section, so now we proceed with the installation of RStudio.

To download RStudio, visit the following website: http://www.rstudio.com and click the button marked "Download R Studio". Here, you are presented with some choices—some free, others for a charge. We will download the free version for Desktop; you will have the choice of Windows, Mac OS, Ubuntu 32- or 64-bit, Fedora 32- or 64-bit. Depending on your machine, choose an operating system, download the file, and click the installer.

Tip: In the Mac OS environment, you will need to click on the RStudio icon and drag it into the "Applications" folder right next to it. This will move the program to the Applications folder on your computer. In a Windows environment, there is a simple wizard to guide your installation; follow the three screens of the installation wizard.

Now you have another free software package on your computer: RStudio.

Note that the Console window, on the left, functions exactly as the console in the base R software that we began to explore earlier in this chapter. You can use basic arithmetic functions in exactly the same way. RStudio offers an environment that is a little bit friendlier than just a basic console.



Introduction to RStudio

RStudio is an integrated development environment for R; as such, it is a requirement that you first have installed the base R software. After following the instructions in the previous section to install RStudio, you will notice at first glance, RStudio looks a bit more like typical modern computer programs/applications than does the base R package. There are more menus to choose from at the top of the screen; this means that there are some more options for completing tasks that can be carried out by pointing and clicking, or at least initially starting off with a point-and-click approach. Also, as you can see in the image below, the RStudio window is actually comprised of three separate windows. More discussion of the GUI (graphical user interface) from within the RStudio environment will be discussed in the next section, RStudio GUI.

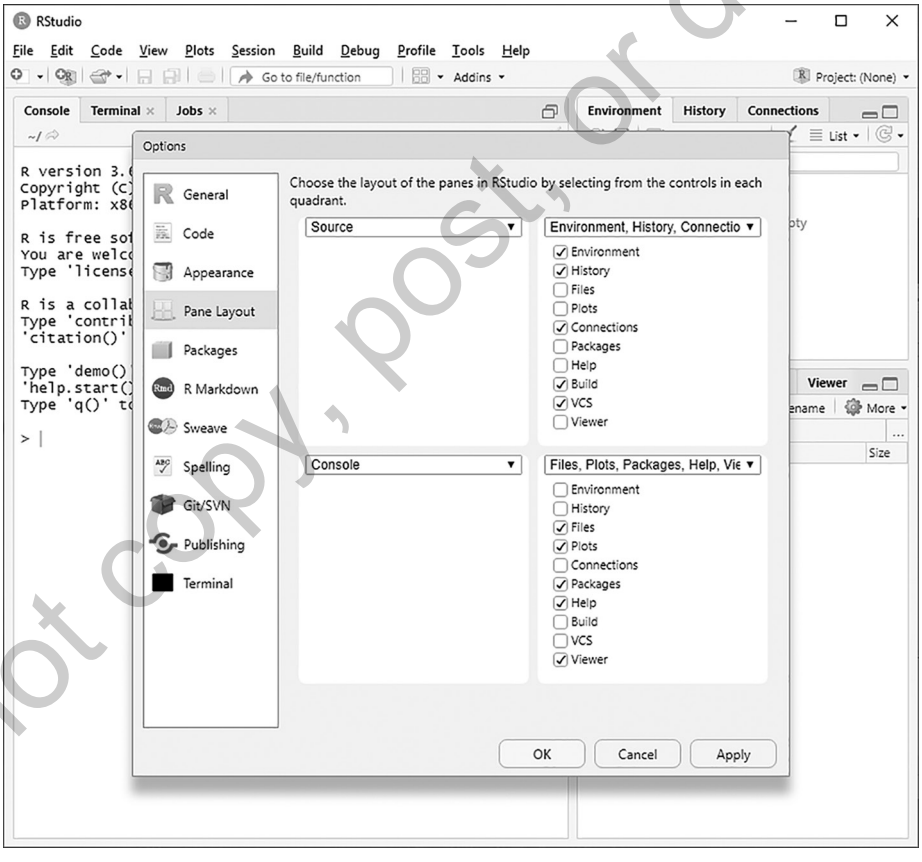
RStudio GUI

The acronym, GUI (graphical user interface), may make it seem like R has now been transformed into a different sort of program and will function exactly like, for instance, SPSS Statistics. This is not the case. However, the RStudio GUI does make many things a bit easier to understand and, among other advantages, the GUI often makes it simpler to manage data and other files.

You will notice that there are several windows within the RStudio main window console. RStudio has the capability to change the location or order of these smaller windows, or panes. To do that, you can follow these menus on a Windows computer:

PC: Tools → Global Options

Macintosh: RStudio → Preferences



In the Options window above, you are able to change not only the organization of the panes within the larger window, but you also have the option to pick and choose menu options within some of those panes. We are not going to make any adjustments here for now, but this is something you might find helpful in the future, depending on which features of RStudio you most frequently use.

The Help Function

There is more than one way to get to the help you need. Here, we will discuss three ways to seek help from within R and RStudio.

First, you can use the programming code in the terminal window to summon help. That command is

```
help.start()
```

After clicking return, you will notice that a help window has been opened in the lower right [default] window pane. Here, you can begin to explore the options within the Help function. These will be described in more detail below.

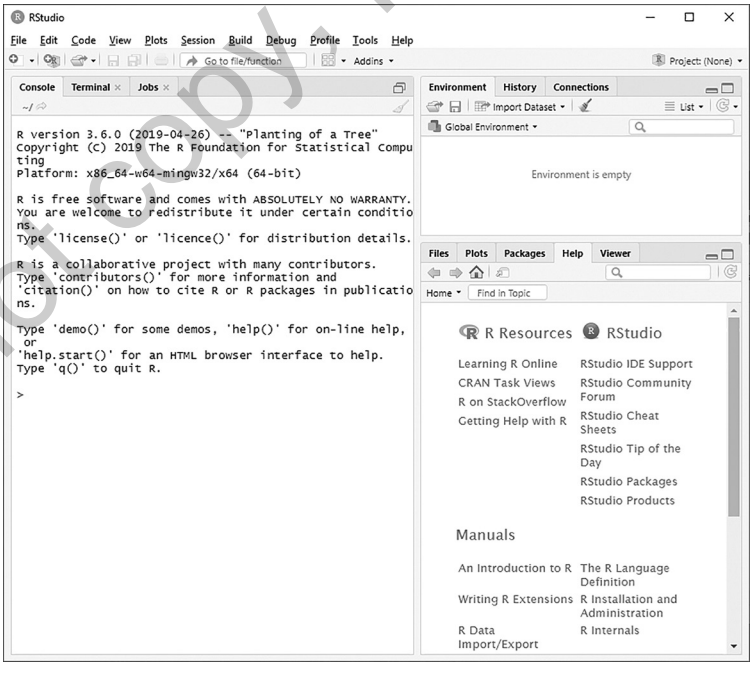
If you are interested in guidance for working with a particular command, simply type a question mark and then the command. For example, if you want guidance concerning the **`getwd()`** command, you would type the following:

```
? getwd
```

Second, if you cannot recall or do not know the name of a specific command, but instead want to find relevant commands for a particular task or function, you obviously cannot use "?" method. Instead, you can use the Help search engine, described in detail below. This can be accessed using the **`help.start()`** command.

Help → R Help

After selecting the help option, you will see that the [default] lower right pane now shows a variety of links, which provide different types of help support.

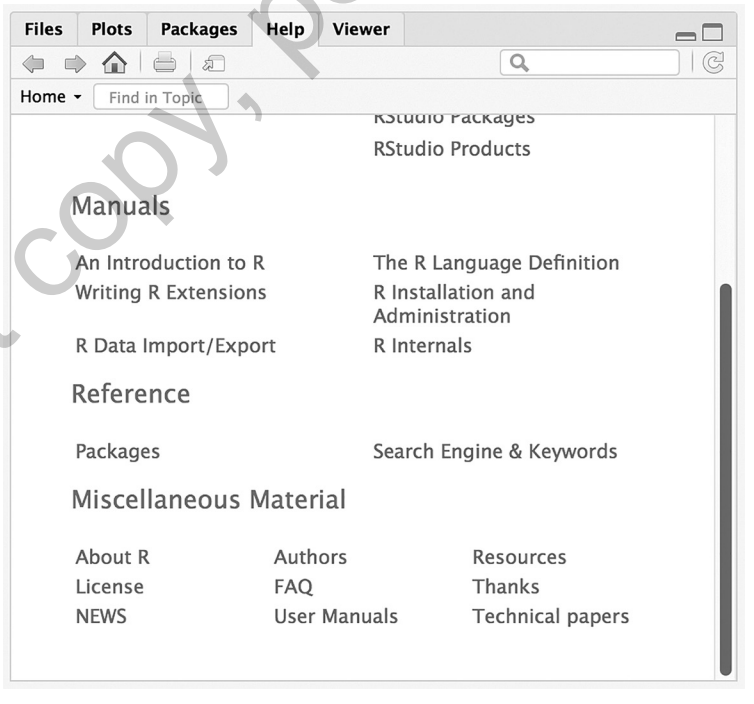


Third, notice that within the pane on the lower right side of the main window, there are some menu tabs at the top. The default menus are "Files," "Plots," "Packages," "Help," and "Viewer." If you select the Help tab, you will see how to access the R Help portal in RStudio. If you've accessed the Help function at an earlier time or if you are using a shared computer and/or software package, then it is possible that someone else has used the Help function at some earlier point in time and what appears may be different from what is shown in the window pane below.

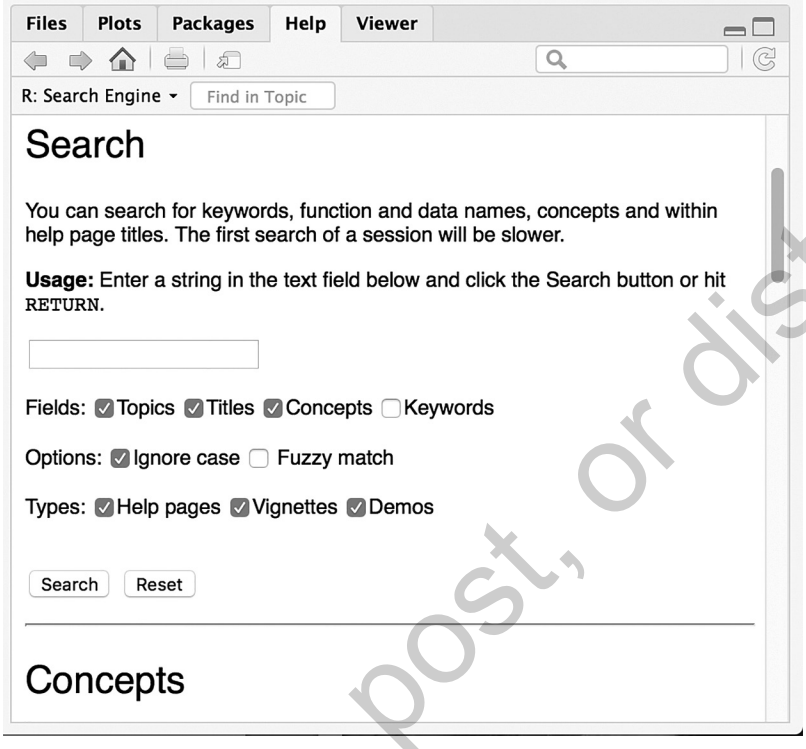
To get back to the Help homepage, you can click on the "Home" icon; this icon looks like a house and can be found in the row underneath the menus for this pane (Files, Plots, Packages, Help, Viewer). There are also forward and reverse arrows that function like a Web browser window, allowing you to go back and review pages already seen and then go forward again.

The Help portal offers a number of different kinds of assist, depending upon your predicament. For example, there are resources specific to R and RStudio. Resources to help you learn to use R to complete specific tasks are also available, including user manuals provided under the heading "Manuals."

What might be comforting and familiar within this group of helpful resources is that there is a search engine that operates in a way you might be familiar with while using other software packages: It provides a way to search for specifically what you are looking for. To reach this search engine, you may need to scroll down in that lower right pane a bit. You will see a heading called "Reference." Under the Reference heading, you will see an option for "Search Engine & Keywords."



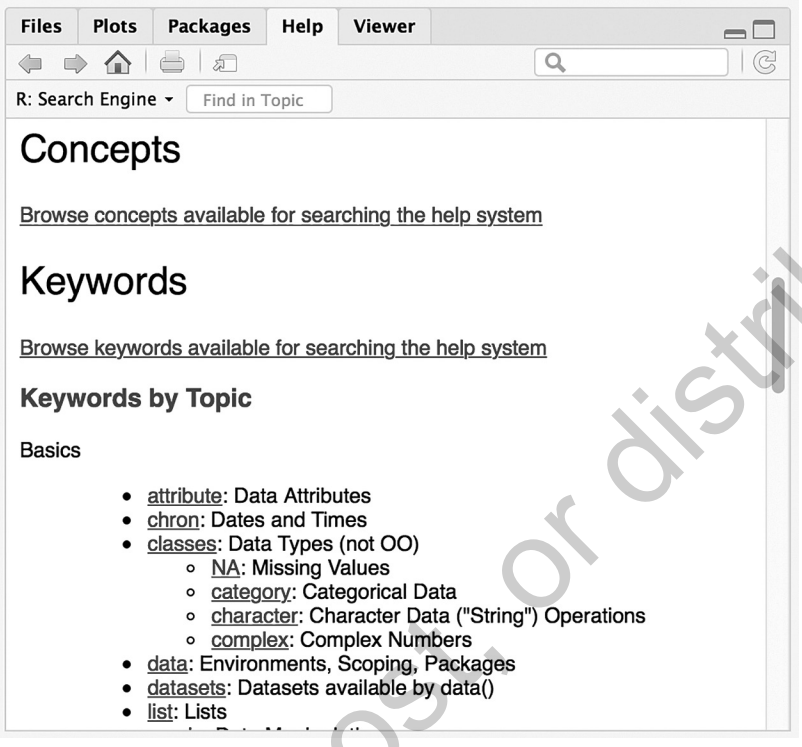
Click on the "Search Engine & Keywords" link and you will find yourself at the R Search Engine prompt, as illustrated here:



Using the available box, you have the option to type in a word, words, or phrases just as you would with any search engine. Beneath the search box, you will see that there are choices to make the search more specific. In the first line, "Fields," you are able to mark the boxes of the areas where the engine will search for your term. You can choose to ask the search engine to return results that appear in topics, titles, concepts, or keywords. Depending on the frequency of the term(s) you are using to search, this may allow you to limit an unwieldy amount of search results or to increase an insufficient search result for terms that might be less common.

Among the options, you can ignore the letter case for your search terms, which means the search will be restricted based on whether or not the words are capitalized. The options under "Types" indicate where you will be linked in the search results, including standard help pages, vignettes, or demonstrations.

If you scroll further down through this window, you will see other sections, such as "Concepts" and "Keywords." Keywords provides a list of keywords that have been entered into the system, which return useful information in Help searches. You can click directly on any of the keywords and you will be taken to a list of associated pages. Depending on the nature of your need for help, this might be an effective strategy to begin browsing for assistance.



R is essentially a group of base commands packaged as "base R," with a multitude of packages available to be added on to the baseline version. Part of what makes this interesting is that each user may customize the packages that are included and so each instance of R across multiple computers might be very different. This can also make getting help a bit more complicated in some cases. One way to streamline that might be to begin with the "Packages" menu in the lower right window pane. Here, you will be shown a list of packages. By clicking on any of those links, you will be taken to a page that will provide a documentation file and then a list of Help pages related to that particular package.

FINDING R AND RSTUDIO PACKAGES

As discussed above, R is a software package that consists of base R. We have added the RStudio package which ensconces the base R within an interface that is a bit more pleasing, in that it offers a bit more point and click functionality, rather than relying solely on programming commands within the base R program. There are many other packages that can be added to customize the abilities of your version of R. In fact, your version of R can be as unique as you like it to be, in terms of the setting, format, and screen layout. It can also be as unique as you like in terms of the packages you choose to add to your base R.

Base R Packages

You can find R packages to run specific commands (based on your needs) at the following website: https://cran.r-project.org/web/packages/

At the time of this writing, there are over 13,000 packages identified at this site. For a complete list, you can select either the link, "Table of available packages, sorted by date of publication" or "Table of available packages, sorted by name." By clicking on the package name, you will be directed to a webpage where you will find a short description of the package in addition to links to download the package, along with other documentation.

RStudio Packages

R packages can also be found through the RStudio website, or directly at the following link: https://www.rstudio.com/products/rpackages/

Here, you will find a variety of project site links. When you click on any of these links, you are directed to a separate website established for that particular package. That page contains detailed information about the package, how to install, and how to seek help. Since these packages are all distinct projects, there is not a unifying single template for these websites.

Information will be presented in different formats across the packages with varying degrees of detail. In general, there are sufficient details to get started—often these details include information about how to utilize the package in more complex ways.

The Tidyverse

One suite of packages that appears on the RStudio site is called ***tidyverse***. **Tidyverse** is a collection of packages that comprise the tidyverse core. As of the release of tidyverse 1.2.0, these include the following: **ggplot2**, **dplyr**, **tidyr**, **readr**, **purrr**, **tibble**, **stringr**, and **forcats**. Information on each of the packages that are included in the tidyverse can be found at the following link: https://www.tidyverse.org/packages/

> **Tip:** There are a few other packages included in the **tidyverse**, but less frequently used. For instance, if you are familiar with or have used other popular statistical packages, you will appreciate a particular package among this group called ***haven***. This package assists users importing data files from the most popular commercial statistical packages: SPSS Statistics, SAS, and Stata.

To install the collection of packages in the **tidyverse**, you can enter the following on the command line in the R console window:

```
install.packages("tidyverse")
```

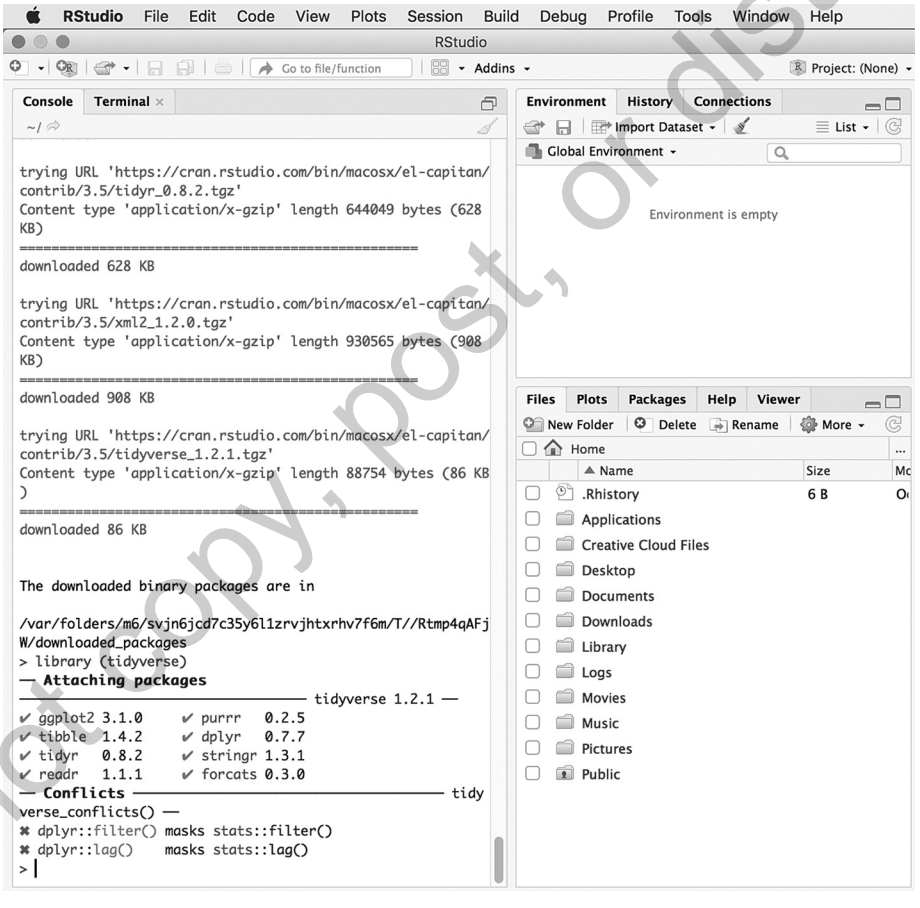
After you enter the command, you will be given information in the console about what files have been downloaded and where they are located on your machine. The next step is to

attach the packages to your version of R. To do this, enter the following on the command line in the console window pane:

```
library (tidyverse)
```

Tip: Note that the package is case sensitive.

Now, you will get output in the console window such as that shown in the following image:



Notice that the eight core **tidyverse** packages have been attached to the R package on your machine. Online documentation generally offers information about downloading, installing, and using the package.

OPENING DATA

R is like any other major statistical package—in order to make calculations or produce graphical or tabular output, data are required. In this section, we will focus on opening existing data files.

It is important to consider the format of data files. R can store and retrieve data files that are comma separated values files, with the suffix .csv. With the help of additional packages, R can also handle other file formats from commercial statistical packages. From whatever format the data are drawn, it is important to consider where the data are located, as well. You will need to set the working directory to match the location of the data. Alternatively, you can download data directly from the Web.

Opening Existing Data

Using R command codes, it is first necessary to locate the data file that we plan to open. To determine the current working directory (i.e., the file folder where R will look for data or save data), you can enter the following on the command line in the R console window pane:

```
getwd()
```

In this case, R returned the following location:

```
"C:/Users/Desktop"
```

Since this is not where the data file that we need to access is located, it is necessary to change the working directory. This can be done by entering the following on the command line in the console window pane:

```
setwd("C:/Users/Desktop/R")
```

If the function was a success, nothing will happen. Another command line prompt ">" will appear beneath the line where you have typed the **setwd** command.

If, however, there was an error that prevented the change from being made, R will inform you of the error:

```
Error in setwd("C:/Users/Desktop/Rbook/"): cannot change
working directory
```

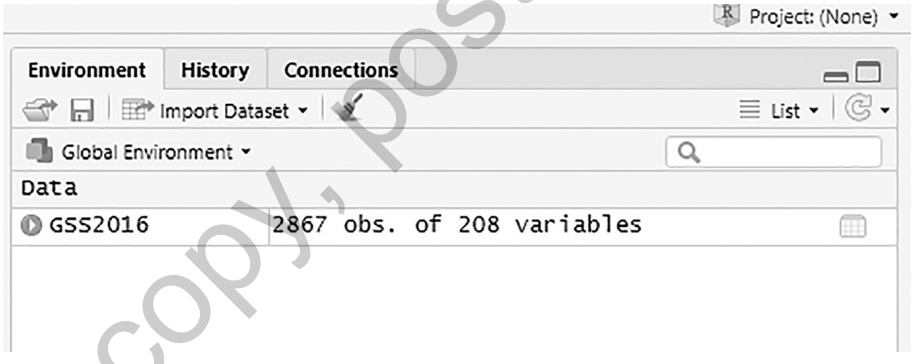
If you did not receive an error message, but would like to verify that the working directory has indeed been changed, you can do that very easily by, again, typing `getwd()` on the command line. It should return the directory that you have just set.

Now that the current working directory has been set, you are ready to read in the data file. Using the command line, one way to do this is by setting the active data file equal to the dataset that you have identified. In this case, we have identified the dataset this will be made available to you with this book and will be used with examples throughout the chapters. This is the *General Social Survey* (GSS) (Smith, Davern, Freese, & Morgan, 2019) 2016 file. This can be done entering the following command:

```
GSS2016 <- read.csv("GSS2016.csv")
```

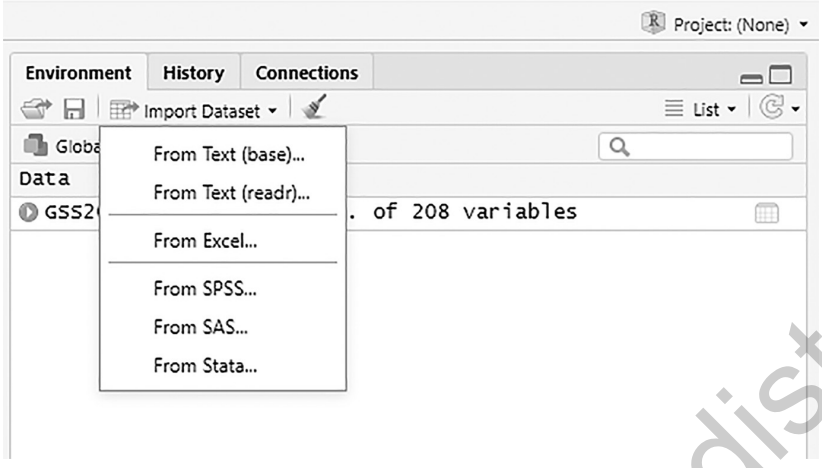
Tip: In later chapters, we list the entire filepath to the data—however, setting the working directory is a useful way to bypass listing all of the embedded folders leading to the location of the data.

If there is a problem, R will provide an error code to let you know that there was a problem. If no error code has been provided, then your operation was likely a success. We can see if the active data file has been set to the GSS2016 file in the upper right window pane. As seen below, it will show GSS2016 and indicate the number of observations (cases) and variables that are contained in the file.

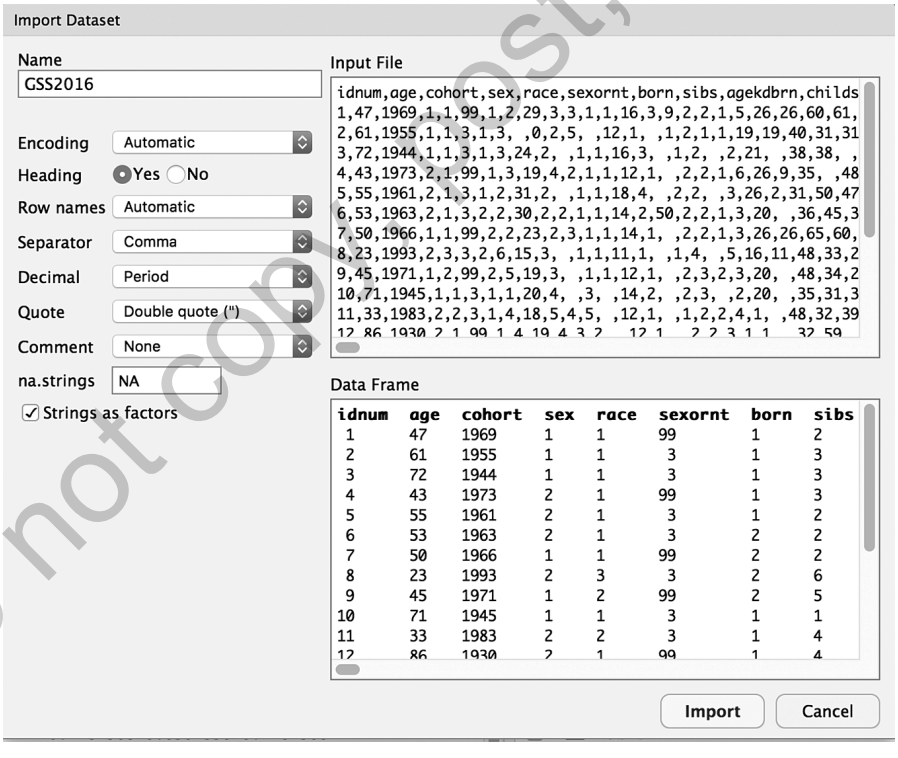


If you look at the Menu tabs in the window from above, you will notice that there is a pull down menu labeled, "Import Dataset." This point-and-click method can also be employed to open an existing file. You will see that there are two options available to open text files: "From Text (base) . . ." and "From Text (readr) . . . ." Either one of these options will work.

The base option is a little less sophisticated than the readr option, which has a slightly more user-friendly window interface for the import process. First, we will try the base option. Click the Import Dataset drop-down menu and then select From Text (base), as shown in the following screenshot.



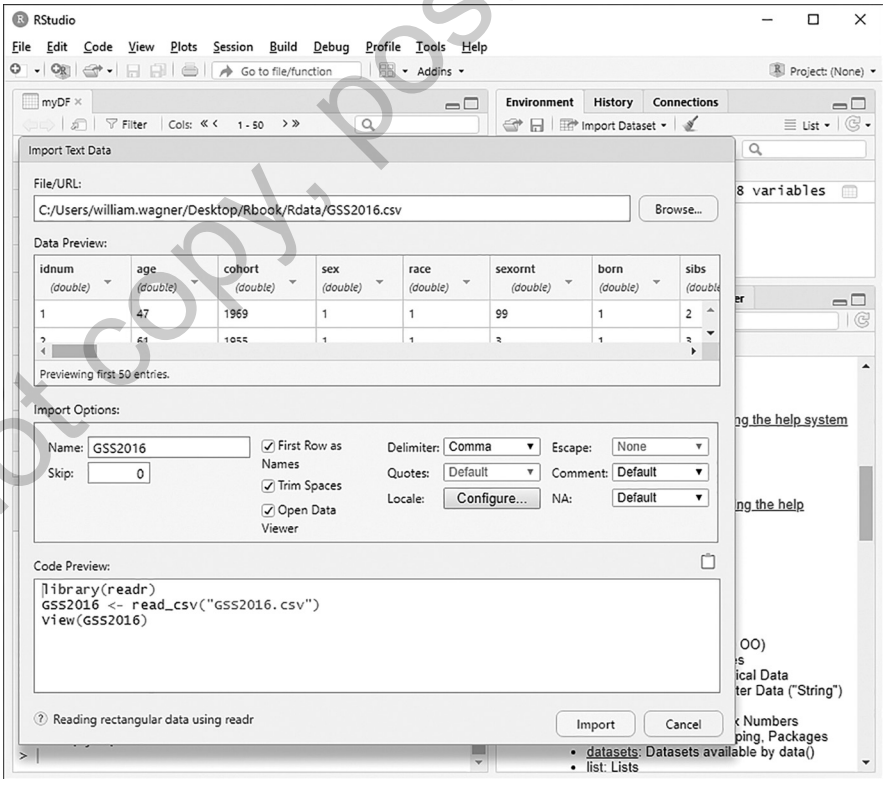
After making that selection, you will be given the option to locate a file on your machine. This might on a hard disk, a flash drive, a network drive, or however you typically access your data files. After locating the data file and selecting it, you will be taken to the next window:



Here, you need to determine whether to use the first line of data for variable names or whether there are not variable names in this file and the data begins on line one. If there are variables names in the first line, then the "Yes" radio button next to "Heading" should be selected. If, however, there are not variable names in the file, then be sure to choose the "No" radio button next to "Heading."

Tip: The data frame shown here reveals that the bolded first row contains variable names (headings) as "Yes" had been selected for "Heading" in this window. If "No" had been selected, those variable names would not be bolded and variable name placeholders (e.g., var1, var2, . . .) would have been used instead.

Now, try the same operation, but use the alternative option—readr. Click the drop-down menu for "Import Dataset", then from the list, choose "From Text (readr) . . . ." At this point, you will be shown a window to locate your data file. Since we are looking for a data file on our machine, you can click the "Browse . . ." button and you will then be able to navigate to the location of your file. Here, you will need to make a determination about whether variable names are included in the first row of the data file. If the first row of your data file contains variable names, be sure to check the box labeled, "First Row as Names." If the first row of your data file does not contain variable names, make sure that box is unchecked.



If your data file does contain variable names as the first line of delimited data, but you do not want to use those variables names, you can import the data and omit the variable names by making sure that the "First Row as Names" is not checked AND the fill-in-box next to "Skip:" is changed from a 0 to a 1. This will drop the top row down from variable name to data case, but the first row will then be skipped.

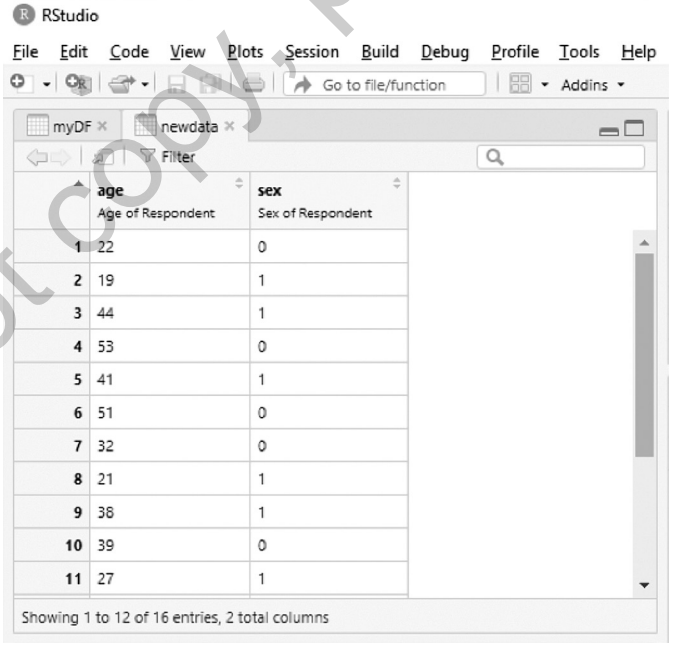
Tip: Another helpful feature of this window is that the "Code Preview:" pane at the bottom of the window exhibits the command code that will be executed once this dialog box is completed. In other words, this is the command code that would be entered into the Console to complete the task without any point-and-click guidance.

Importing Other Formats

Data other than *.csv files can also be imported into R. This can be done with the help of the (**haven**) package. In this example, we show code to read in an SPSS Statistics data file to R. The following command code would be entered in the R console:

```
library(haven)
   newdata <- read_sav("~/Desktop/SPSS/data files/newdata.sav")
   View(newdata)
```

Tip: This is just to provide sample code for this process since we do not provide additional SPSS data for these examples.



The image above shows what is delivered to the upper left pane of the R Studio window. Of course, this can also be achieved using the "Import Dataset" drop-down menu. When selecting "Import Dataset", scroll down and choose "From SPSS . . ." You will be presented with a window like the one you received when opening a CSV file. Follow along the same instructions to open your file in R.

Reading Data From a URL

A data file located at a URL is opened in a similar way to a data file on your computer. The only real difference is specifying that there is a URL. The following command code can be used to open data. In the event that you wanted to open an SPSS Statistics data file on the Web at https://study.sagepub.com/system/files/demo.sav, the following command code works just the same as if it were located on your computer or an attached drive:

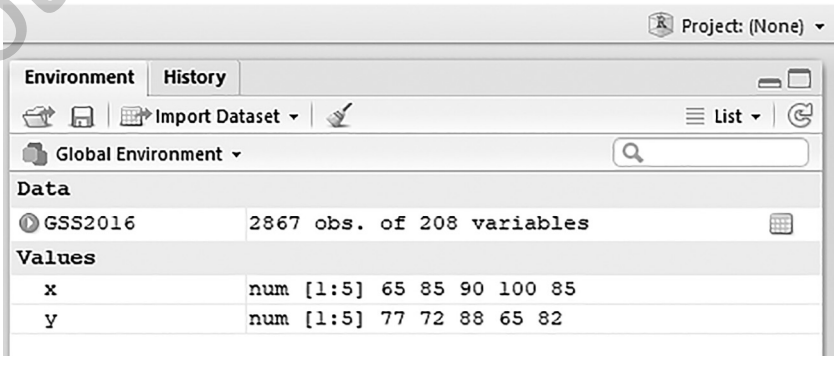
```
library(haven)
  demo <-
  read _ sav("https://study.sagepub.com/system/files/demo.sav")
  View(demo)
```

Entering Data

The process of entering a large amount of data into R can be extremely cumbersome. Frankly, it is often more efficient to create a data file with another program and read it in to R. However, it is certainly possible to enter data into R directly. One way is through use of the command line. Suppose we need to create a data file with two variables, *x* and *y*. Further suppose that each of those variables has five cases, corresponding to scores on two exams taken by five individuals. Including the scores themselves, below you will find the command code to produce the variables, populated with the data, within R:

```
x <- c(65,85,90,100,85)
y <- c(77,72,88,65,82)
```

Once that is done, you can see the values have been added to the upper right pane of the R window:



Once the data have been entered, you can perform operations, calculate statistics, and so on. For example, if you wanted to calculate the correlation between *x* and *y* (the procedure is discussed in a later chapter), you would enter cor (*x*, *y*), and the following result would be returned:

```
cor (x, y)
```

```
-0.276335
```

SAVING DATA FILES

One way to save data is by quitting R. This is done either by typing "quit()" in the console at the command prompt or by closing the R window or using the computer menus (command + Q in Mas OS; alt + F4 in MS Windows). When prompted to "Save workspace image?," be sure to click "Yes" and all of your data will be saved for next time.

One way to save a data file for use on another computer—or to share with others perhaps working on a different machine—is to use the **`save()`** command. This command code can be used to save one or more R objects to .rda file format. This data file can be read back into an instance of R using the **`load()`** command.

Suppose you have created new variables as independent objects in the workspace. In order to save them along with the data file, they first need to be aggregated. This can be done with the following command code, which aggregates the variables into a data frame, testscoresdata:

```
Testscoresdata <- data.frame(x, y)
view(testscoresdata)
```

Now to save the newly created data file, you can use the following command code:

```
save(testscoresdata, file = "testscores.RData")
```

The new file will be saved to the workspace directly as an R data file. R data files can always be loaded using the **`load()`** command.

You can take this opportunity to save the *General Social Survey* 2016 data (Smith et al., 2019) that was loaded in as a .csv file; it can now be saved as a *.Rdata file. Use the following command to save the file as a native R data file:

```
save(GSS2016, file = "GSS.RData")
```

Tip: We will be importing the .csv file each time we load the data since these are the data that are available on this book's website: https://study.sagepub.com/researchmethods/statistics/gillespie-r-for-statistics

CONCLUSION

R is a programming language designed to analyze data. While R can be a bit unwieldy, RStudio is software that makes R a bit more intuitive—and perhaps a bit more accessible for most users. Going further, there are "packages" such as ***tidyverse*** that can be added to RStudio to help standardize tasks so that the wheel need not be reinvented for each job. Still, R will not be as polished or intuitive as commercial statistics software, such as SPSS Statistics; however, R does have the advantage of being free and it can be a powerful and flexible way to analyze data for patient users. In this chapter, we have given you the tools to embark on your journey with R. Throughout the rest of this book, we explain how to use R to accomplish a wide range of data analysis activities.

References

Smith, T. W., Davern, M., Freese, J., & Morgan, S. L. (2019). *General Social Surveys, 1972–2018* [Machine-Readable Data File]. Chicago, IL: NORC at the University of Chicago. Retrieved from http://www.gss.norc.org/getthedata/Pages/Home.aspx

Supplementary Digital Content

Download datasets and R code at the companion website at https://study.sagepub.com/researchmethods/statistics/gillespie-r-for-statistics