CHAPTER 1

INTRODUCTION

What Is R?

R is a powerful, flexible, and free tool that can be used for statistics. Beginning to use this tool can seem overwhelming, though. The flexibility, which is eventually a great asset, can make the initial learning curve appear steep. This book introduces a few aspects of this tool. As you become comfortable with these aspects, you develop a foundation from which you can later explore R and the packages available for R more thoroughly. This introduction will not explain every possible way to analyze data or perform a specific type of analysis. Rather, it focuses on the analyses that are traditionally included in an undergraduate statistics course and provides one or two ways to run these analyses in R. The goal is to reduce the initial learning curve while providing a useful and coherent introduction to this tool.

DOWNLOADING R AND RSTUDIO

The first step to using R (version 3.6.0; R Core Team, 2019) is downloading the program. There are versions of the program for PC, Mac, and Linux users (https://cran.r-project .org/), and the version available as you read this book may be different than the version listed. When you start the download, you will be prompted to select a mirror (server with stored information); select the `0-Cloud` option. Then, provide permission for your computer to download the files and walk through the prompts.

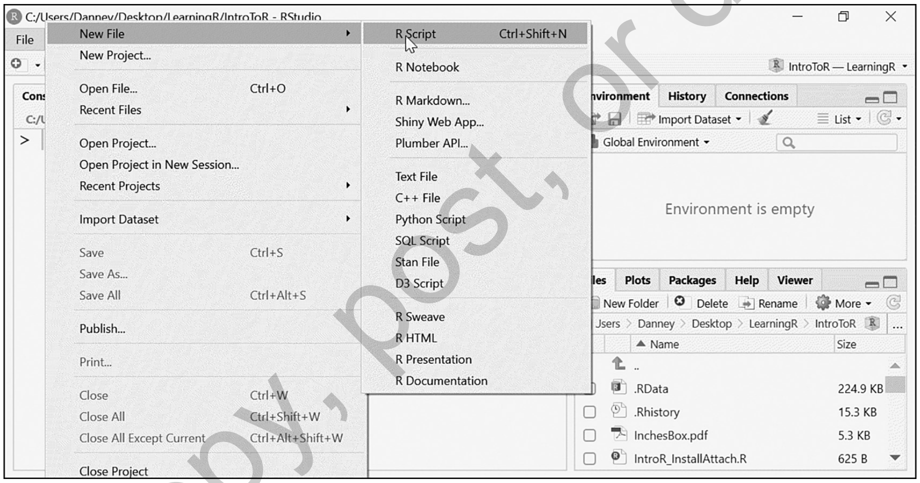
All examples in this book use RStudio (version 1.2.1335; RStudio Team, 2018). A free version of this integrated development environment is available for download: https://www.RStudio.com/products/RStudio/download/. This environment allows you to use the R program more easily and decreases the initial learning curve. RStudio allows for importing data without writing code, and it makes viewing data and output files simpler. In addition, RStudio uses predictive text to suggest function names as you begin to type, which facilitates learning new packages.

CREATING A PROJECT FOLDER

One of the nice utilities in RStudio is that you can create a project folder that contains all of the files that you are using for a particular set of analyses. These files may be script files (which contain the code you use), data files (which contain the information from your participants), output files, and so on. This structure allows you to call files more quickly and easily because RStudio assumes the files are in the working project folder unless you state otherwise; this saves a lot of typing and headaches as you do not have to remember "C:/Users/MrAwesome/Desktop/StatsPurgatory/RforNewbies/Practice-Data.csv." Instead, you simply type "PracticeData.csv" and R assumes the rest. This feature saves time, improves work flow, and makes importing data easier.

First, create a new folder on your desktop named *LearningR*. This folder will hold the project and related files. If this is all new to you, instructions for creating a new folder and setting the project folder are provided with pictures in Appendix 1A.

Second, create a project in RStudio. Go to *File > New Project…*. Then, select (1) *New Directory*, (2) *New Project*, (3) *Browse…*, (4) *Desktop*, and (5) your recently created folder: *LearningR*. Finally, click *Open*, enter *IntroToR* as the *Directory Name:*, and select *Create Project.* RStudio will reopen with the new project active.

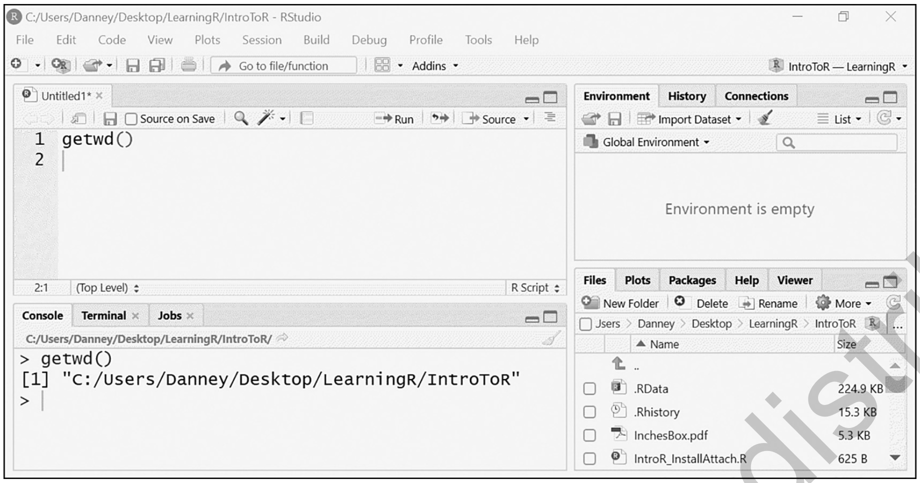


You can check that the correct project is active by opening a script to ***get*** the ***w***orking ***d***irectory. Open a new script file; go to *File > New File > R Script*. Type *getwd()* in the script file, and hit *ctrl + enter* on your keyboard.

Congratulations! You used your first command. *Ctrl + enter* runs the line of code where your cursor is located, and you can see the output in the *Console* pane on the bottom left. This output shows that the active working directory is the new project file we created.

Note: If you need to set the working directory (*setwd*), you can right click the folder you want to use and open "Properties." This window provides the location for the folder, which you can copy and paste into the script file. You will need to change the backslashes *(\)* to forward slashes *(/)*, though.

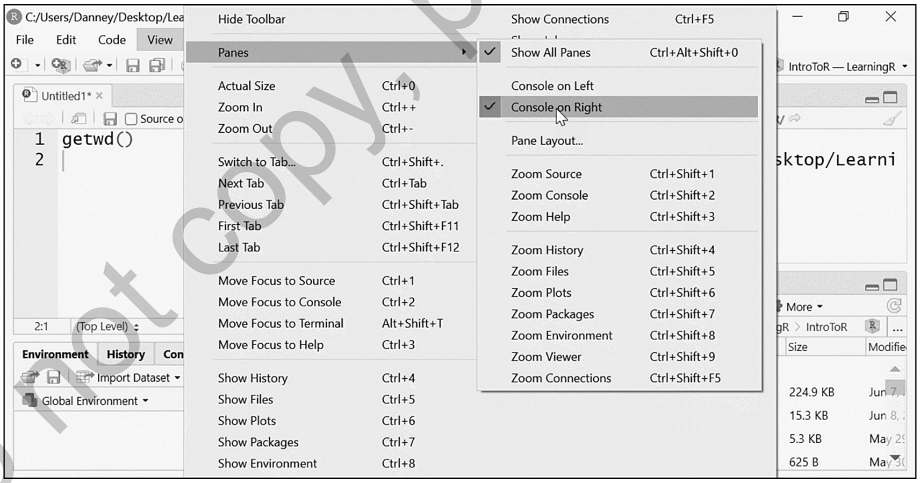
```
setwd("C:/Users/Danney/Desktop/LearningR/IntroToR")
```


```
getwd()
```

```
> getwd()
[1] "C:/Users/Danney/Desktop/LearningR/IntroToR"
>
```
C:/Users/Danney/Desktop/LearningR/IntroToR - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
IntroToR — LearningR
Untitled1*
Source on Save
Run
Source
(Top Level)
R Script
Console Terminal Jobs
C:/Users/Danney/Desktop/LearningR/IntroToR/
Environment History Connections
Import Dataset
List
Global Environment
Environment is empty
Files Plots Packages Help Viewer
New Folder Delete Rename More
Users > Danney > Desktop > LearningR > IntroToR
Name Size
.RData 224.9 KB
.Rhistory 15.3 KB
InchesBox.pdf 5.3 KB
IntroR_InstallAttach.R 625 B

GETTING ACQUAINTED WITH THE RSTUDIO ENVIRONMENT

There are four panes (i.e., windows) in RStudio. Due to the length of some code throughout the book, we are going to adjust the pane layout so the working document (*`Source`*) and output (*`Console`*) are beside each other. Select *`View > Panes > Console on Right`*.



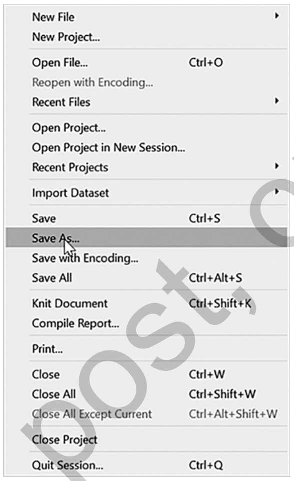
Learning RStudio Panes

The pane in the upper left is called the *`Source`* pane. This pane typically contains your working files (e.g., scripts that hold new code). Using scripts is important because they allow you to save and edit your code. Once you run the code, you will see it in the *Console pane*, which cannot be edited.

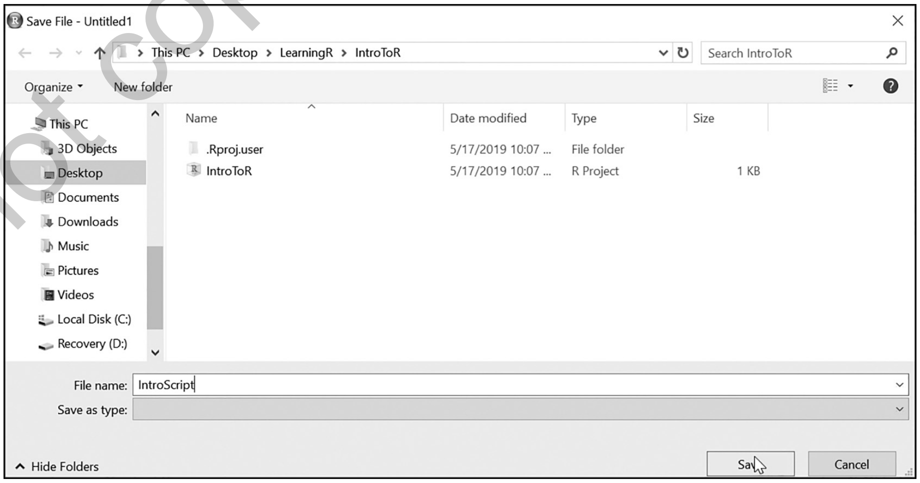
Saving code in the script has a number of advantages over coding directly in the *Console* pane: (1) You can modify or run the code later, (2) others can see exactly what you did with your data and how you completed your analyses (yay for reproducibility!), and (3) you can save time later when you complete similar analyses by using previous scripts as a starting point and reminder about how to call a function: a set of code that performs an action.

Furthermore, you can make comments in the script file. These comments serve as reminders later about how to use a function or what a particular portion of the code does. To create a comment, simply type *#* in front of the text to start the comment.

To save the script, select *File* > *Save As*....



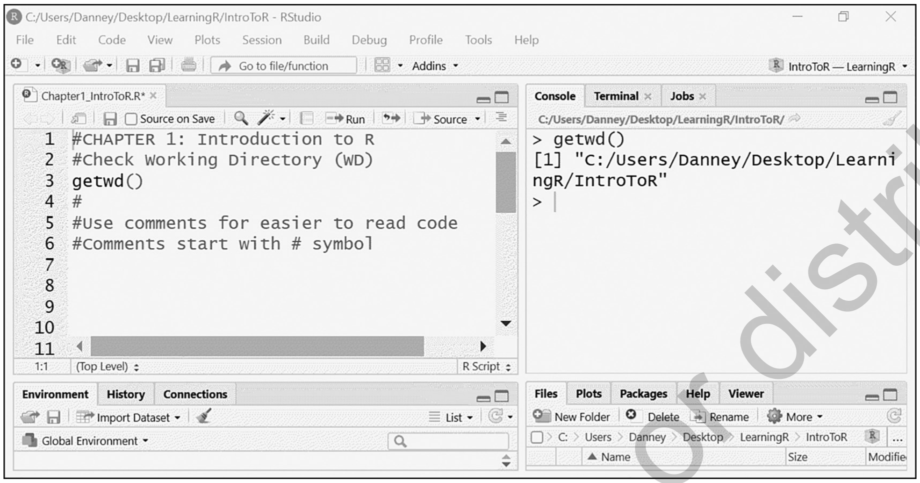
Because you set the working directory using the project folder, RStudio assumes you want to save the new script to the project folder (you are saving time already). You then enter a name for the script such as *IntroScript*.



Name	Date modified	Type	Size
.Rproj.user	5/17/2019 10:07 ...	File folder	
IntroToR	5/17/2019 10:07 ...	R Project	1 KB

File name: IntroScript
Save as type:

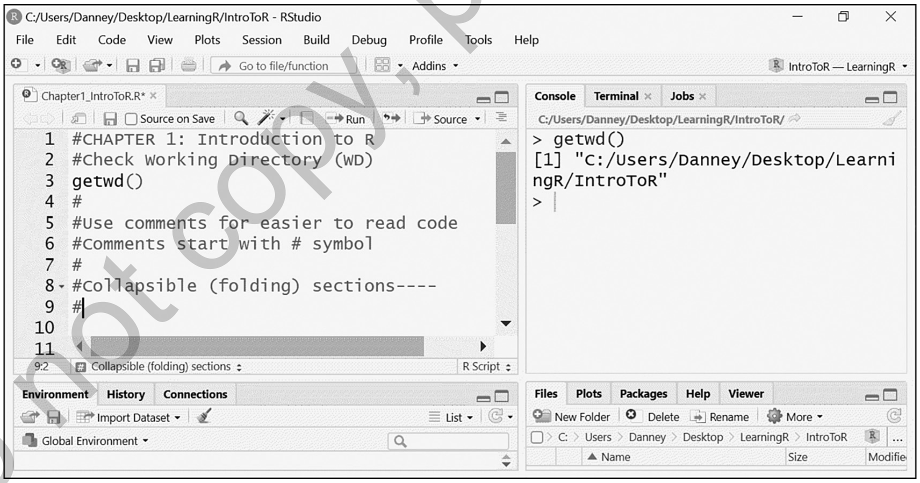
We can add some notes at the beginning of the script using the comments (remember: start the comment with #). These comments will jog your memory later when you look at this file.


C:/Users/Danney/Desktop/LearningR/IntroToR - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
IntroToR — LearningR
Chapter1_IntroToR.R*
Source on Save
Run
Source
Console Terminal Jobs
C:/Users/Danney/Desktop/LearningR/IntroToR/
1:1
(Top Level)
R Script
Environment History Connections
Import Dataset
List
Global Environment
Files Plots Packages Help Viewer
New Folder Delete Rename More
C: Users Danney Desktop LearningR IntroToR
Name Size Modified

```
#CHAPTER 1: Introduction to R
#Check Working Directory (WD)
getwd()
#
#Use comments for easier to read code
#Comments start with # symbol
```

```
> getwd()
[1] "C:/Users/Danney/Desktop/Learni
ngR/IntroToR"
>
```

You can improve the utility of your code by adding collapsible (foldable) sections so you can work on one section at a time. This feature is especially useful for large projects that have multiple tests or hypotheses. You can create separate scripts for each hypothesis, and you can use foldable sections to divide the code into manageable chunks.


C:/Users/Danney/Desktop/LearningR/IntroToR - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
IntroToR — LearningR
Chapter1_IntroToR.R*
Source on Save
Run
Source
Console Terminal Jobs
C:/Users/Danney/Desktop/LearningR/IntroToR/
9:2
Collapsible (folding) sections
R Script
Environment History Connections
Import Dataset
List
Global Environment
Files Plots Packages Help Viewer
New Folder Delete Rename More
C: Users Danney Desktop LearningR IntroToR
Name Size Modified

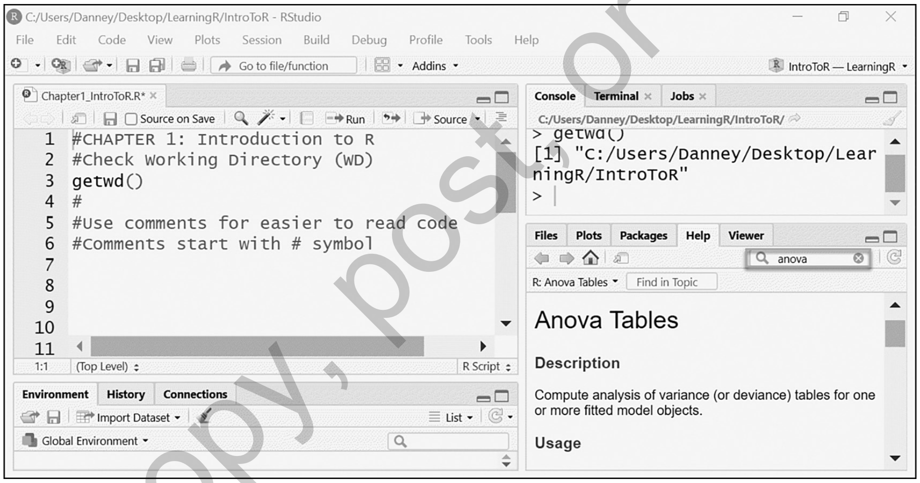
```
#CHAPTER 1: Introduction to R
#Check Working Directory (WD)
getwd()
#
#Use comments for easier to read code
#Comments start with # symbol
#
#Collapsible (folding) sections----
#
```

```
> getwd()
[1] "C:/Users/Danney/Desktop/Learni
ngR/IntroToR"
>
```

In addition to the `Source` pane, there is a `Console` pane, which can be seen on the right. This pane is where code that is run and the output from that code is located. You may occasionally run code and not see any output besides the line of code you ran. This is normal and probably means you are assigning values or output to objects (i.e., a data structure that holds information for later use). If you want to call the output, you can simply type the name of the object (see Coding in R: Object-Oriented Programming in Chapter 2).

There is also a pane for *`Files`*. These are files that are active, and you can use this pane to activate new data files, script files, and so on. The *`Plots`* tab within this same pane contains figures and tables when you create them as part of your output. The *`Packages`* tab allows you to see and manage the packages that are attached (i.e., activated). You also can install new packages using this tab, although we will install new packages using code in the script.

You might wonder how you can find new packages that perform the analyses you need to complete. This wondering leads us to the next tab. The *`Help`* tab is a great place to start if you need to find information about a function or search for a new function to complete a specific type of analysis. For example, if you wanted to do an analysis of variance (ANOVA) and you did not know how in R, you could simply type *`anova`* in the search bar in the *`Help`* tab, and you are off and running.



```
#CHAPTER 1: Introduction to R
#Check Working Directory (WD)
getwd()
#
#Use comments for easier to read code
#Comments start with # symbol
```

```
> getwd()
[1] "C:/Users/Danney/Desktop/Lear
ningR/IntroToR"
> |
```

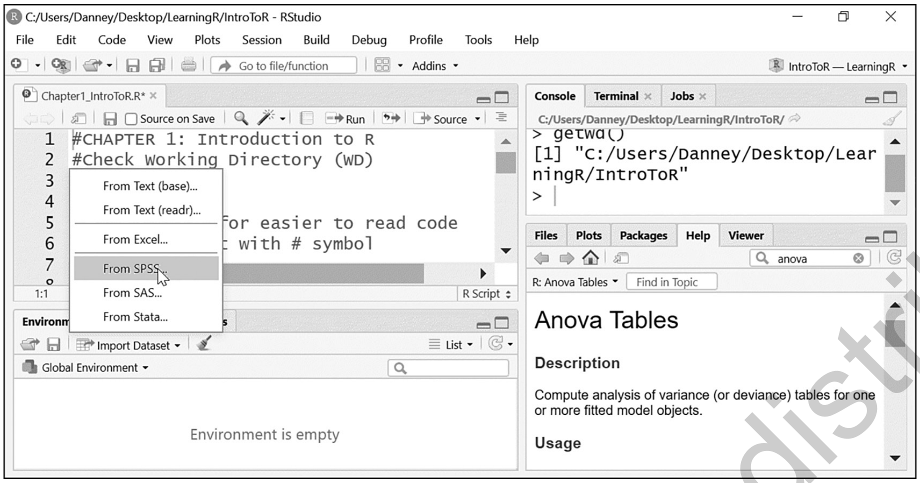
Anova Tables

Description

Compute analysis of variance (or deviance) tables for one or more fitted model objects.

Usage

The final pane we discuss is the *`Environment`* pane. This pane is useful in seeing the objects you have created, and you can attach a data frame (basically what R users call a dataset) using *`Import Dataset`*.



```
#CHAPTER 1: Introduction to R
#Check Working Directory (WD)


for easier to read code
t with # symbol
```

```
C:/Users/Danney/Desktop/LearningR/IntroToR/
> getwd()
[1] "C:/Users/Danney/Desktop/Lear
ningR/IntroToR"
>
```

Anova Tables

Description

Compute analysis of variance (or deviance) tables for one or more fitted model objects.

Usage

Finding Help

There are several ways to find help in R. We discussed the *`Help`* tab earlier, and you can access that source of information through the script too using *`help(anova)`* or *`?anova`*. These initial searches work well if you know the function or package name. A couple of possible issues commonly arise, though. Occasionally, the text you search causes problems because of the syntax. You often can solve this issue by adding quotes around the term: *`help("anova")`*. Second, your initial search may not return a useful result. You can expand your search by using *`??anova`* or *`help.search(anova)`*.

Once you know the function you need to use, it can be helpful to see an *`example`*, *`demo`*, or *`vignette`* for the function. The demonstrations and vignettes provide more thorough illustrations of how to use the function indicated along with explanations of the code. You can see the vignettes available for your installed packages using *`browseVignettes()`*.

```
example(anova)
demo(anova)
vignette(anova)
```

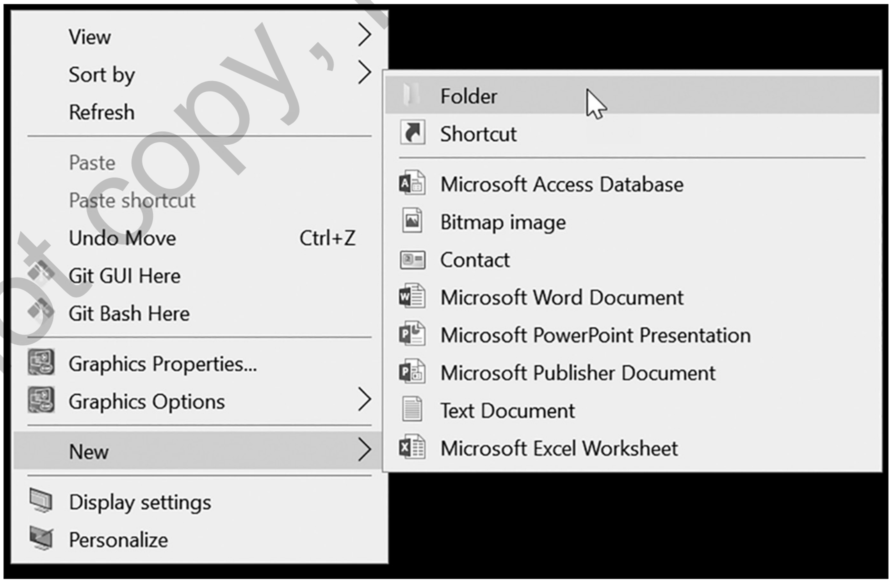
If you tried these examples, you already learned there are not examples, demonstrations, and vignettes for every function. Fortunately, you can also use online resources if you have an active Internet connection. The *`RSiteSearch()`* function allows you to search for a word or phrase in the help pages for all functions in the Comprehensive R Archive Network (CRAN). Finally, you can use https://rseek.org if you want an R-specific search engine, and you can use websites like https://www.r-bloggers.com and https://stackoverflow.com to find answers to specific questions.

Chapter 1 Summary of Key Functions (AKA: Function Cheat Sheet)

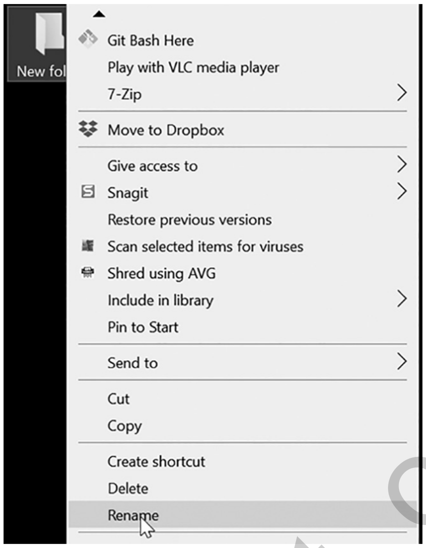
Function Call	Package	Description
getwd	Included in base	Gets working directory
setwd	Included in base	Sets working directory
help	Included in base	Helps with installed packages and functions
?	Included in base	Helps with installed packages and functions
help.search	Included in base	Searches help system for words matching input
??	Included in base	Searches help system for words matching input
example	Included in utilities (utils)	Provides example for function
demo	Included in utils	Finds demonstrations for packages and functions
vignette	Included in utils	Searches vignettes that explain package uses
browseVignettes	Included in utils	Lists vignettes for currently installed packages
RSiteSearch	Included in utils	Searches all CRAN package documentation

APPENDIX 1A: PREPARING RSTUDIO PROJECT FOLDER

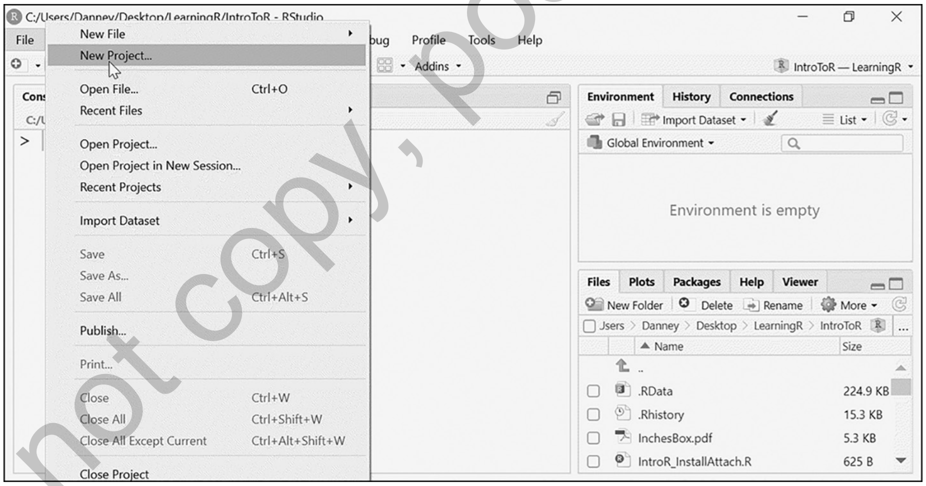
If you need more direction on creating a folder and setting the RStudio project folder, follow these instructions. Right click on your desktop, go to `New,` and select `Folder`.



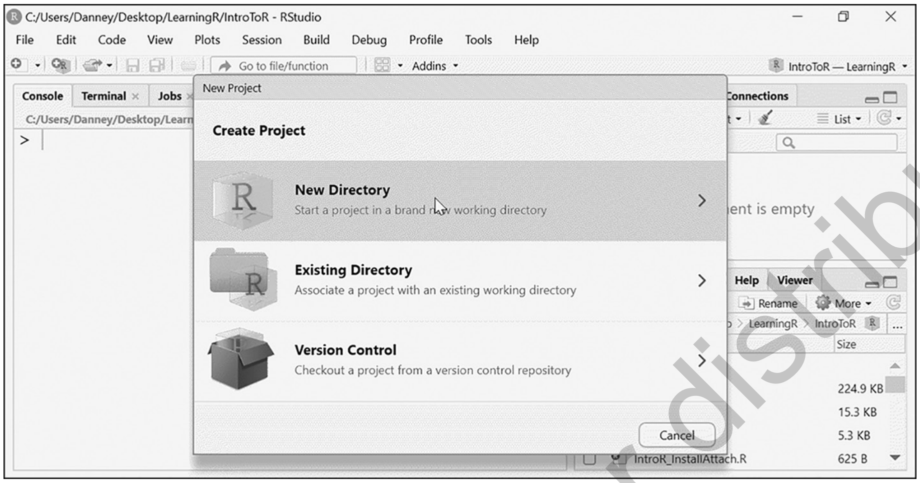
Next, right click the *New folder*, select *Rename*, and enter *LearningR* as the folder name.



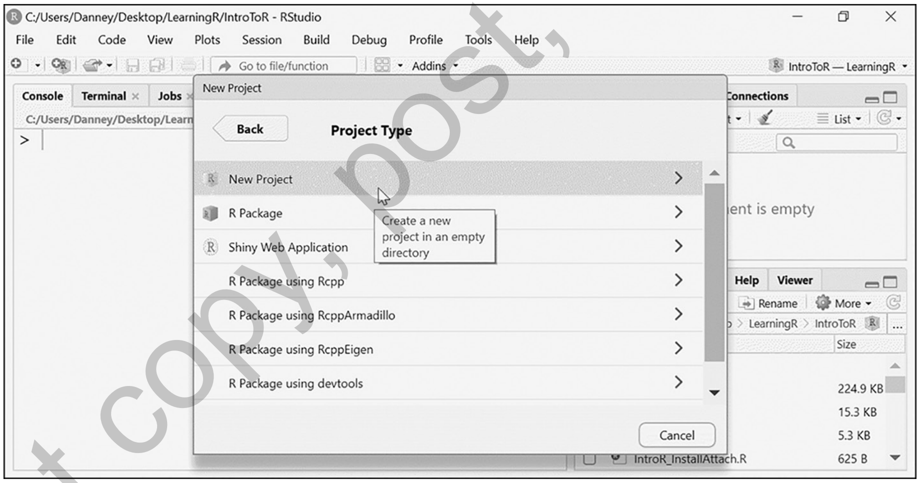
Next, create a project in RStudio. In RStudio, go to *File* > *New Project…*.



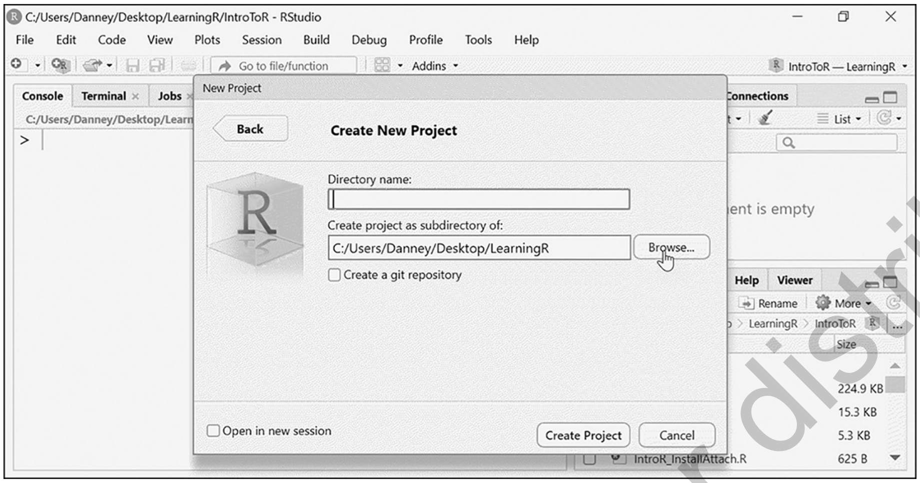
Select *New Directory*.


C:/Users/Danney/Desktop/LearningR/IntroToR - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
New Project
Create Project
New Directory
Start a project in a brand new working directory
Existing Directory
Associate a project with an existing working directory
Version Control
Checkout a project from a version control repository
Cancel

Select *New Project*.


C:/Users/Danney/Desktop/LearningR/IntroToR - RStudio
New Project
Back
Project Type
New Project
R Package
Shiny Web Application
R Package using Rcpp
R Package using RcppArmadillo
R Package using RcppEigen
R Package using devtools
Create a new project in an empty directory
Cancel

Select *Browse...*.



Directory name:

Create project as subdirectory of: C:/Users/Danney/Desktop/LearningR [Browse...]

☐ Create a git repository

☐ Open in new session

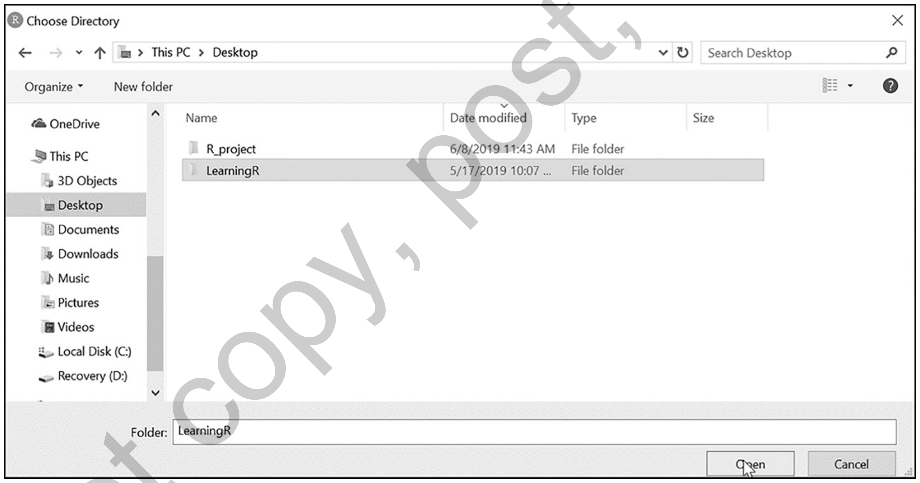
C:/Users/Danney/Desktop/Learn

ent is empty

LearningR > IntroToR

224.9 KB
15.3 KB
5.3 KB
IntroR_InstallAttach.R 625 B

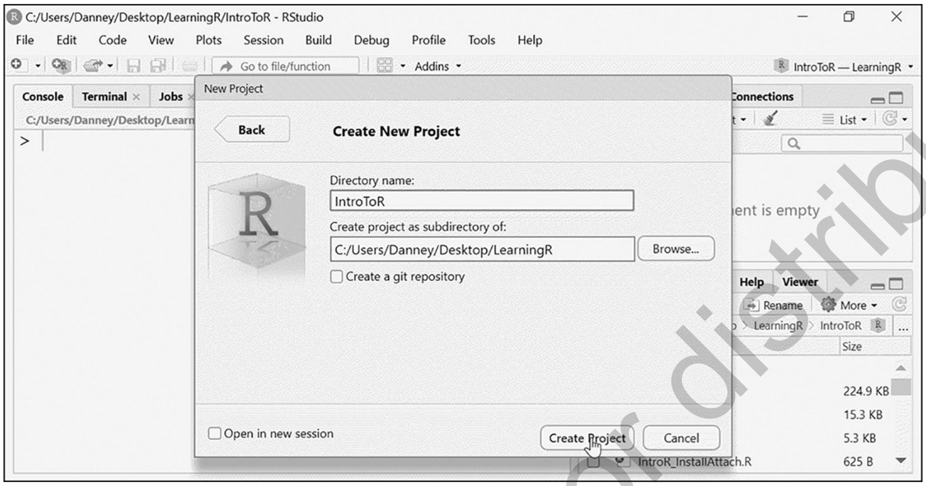
Select *Desktop*. Then, select your folder (*LearningR*) and click *Open*.



Name	Date modified	Type	Size
R_project	6/8/2019 11:43 AM	File folder	
LearningR	5/17/2019 10:07 ...	File folder	

Folder: LearningR

Finally, enter *IntroToR* as the *Directory Name:* and select *Create Project.* RStudio will reopen with the new project active.



Directory name:
IntroToR
Create project as subdirectory of:
C:/Users/Danney/Desktop/LearningR
☐ Create a git repository
☐ Open in new session