Contents

R: Getting Started

Outline:

By the end of this module, you should be able to understand and execute the following tasks:

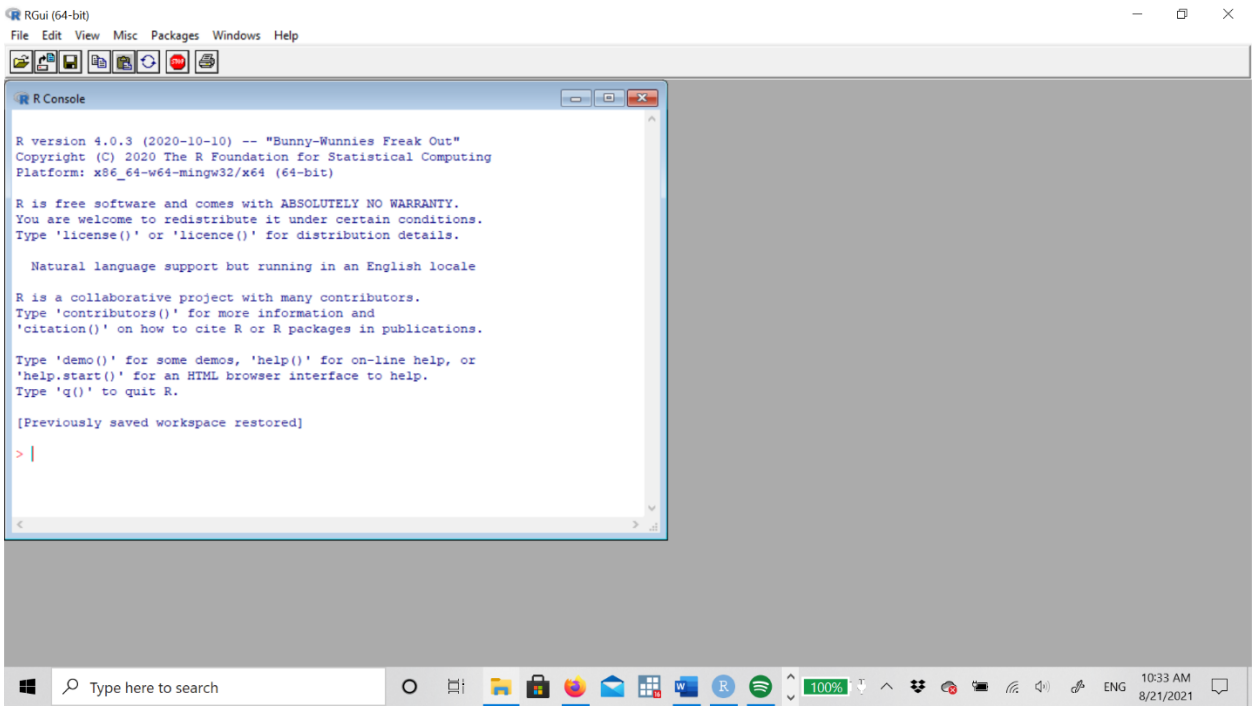
- Understand the basic pros and cons of R
- Set the working directory
- Access and utilize different programs within R
- Import and export datasets
- View data

Understand the basic pros and cons of R

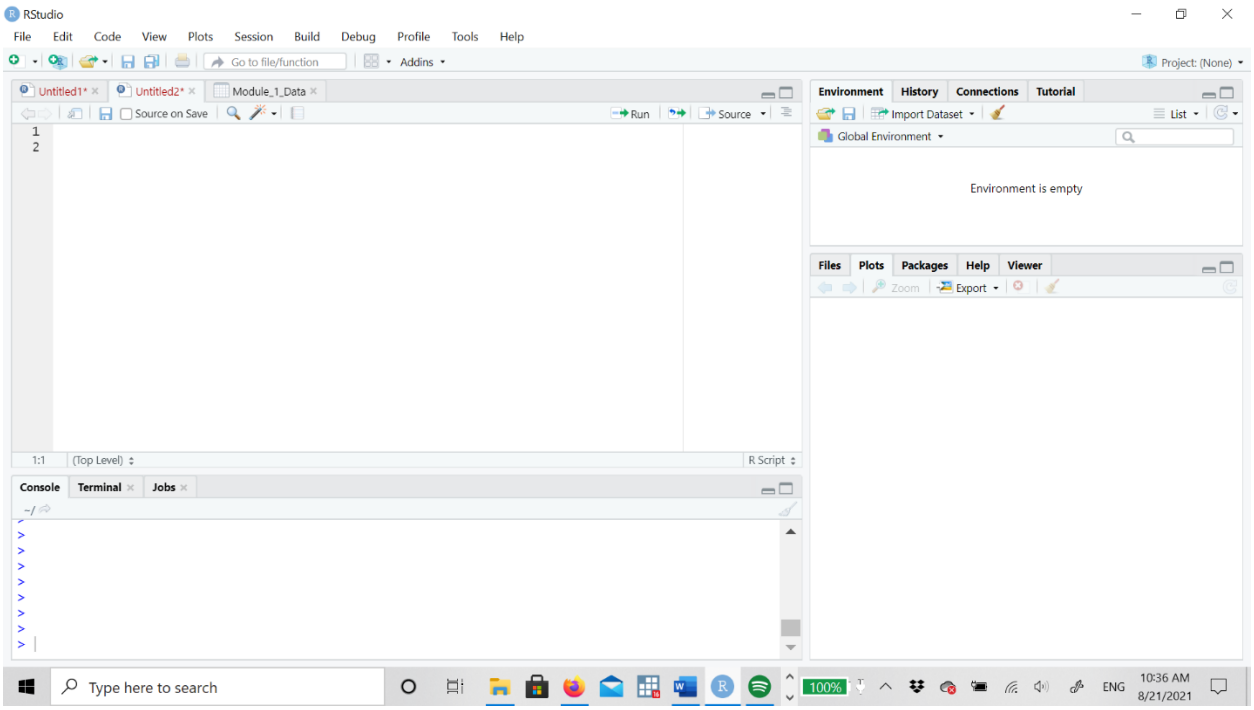
R is a very useful statistical software used for a variety of tasks such as descriptive statistics, modeling, graphics, and more. One of the main pros of R is its accessibility. In comparison to other programs, R is free and has thousands of different packages to download. One of the main pitfalls of R is the learning process. R is not like excel or SPSS where you can perform most tasks by pointing and clicking. Instead, R uses code and a specific syntax. Think about it like any movie with a computer hacking scene, some hero is “hacking” into some database and they are typing into a black box which then gives some sort of output. R is a bit like that. You type commands in a box and you either get an output or you get an error message. This may seem daunting at first, especially if you have never programmed before, but once you learn the basis components of writing code in R, you can use those skills and apply them to more difficult tasks in R.

R and R-Studio:

R has two main forms, you have base R which looks like this:



I would personally not recommend using base R unless you have a firm grasp of how to use R and are comfortable inputting commands. Instead these modules will utilize R-Studio, which looks like this:



The top left box is where you will type all your commands. The bottom left box is the console, this is where your commands will be outputted for the most part (if you receive an error message, it will pop up in this box). The top right box is your R-Studio environment, when you upload datasets or create objects, they will appear in this section. Finally, on the bottom right is the plots section. If you create graphics, they will be displayed in this section. This section also has tabs for packages and help, which you can use to download packages and search for help but this module will also show you how to code ways to download packages and search for help.

Throughout this module there will be screenshots of R-Studio, so if you ever get confused about where something will be displayed or you have some confusion about the code, the screenshots are there to guide you.

To download R and R-Studio go to:

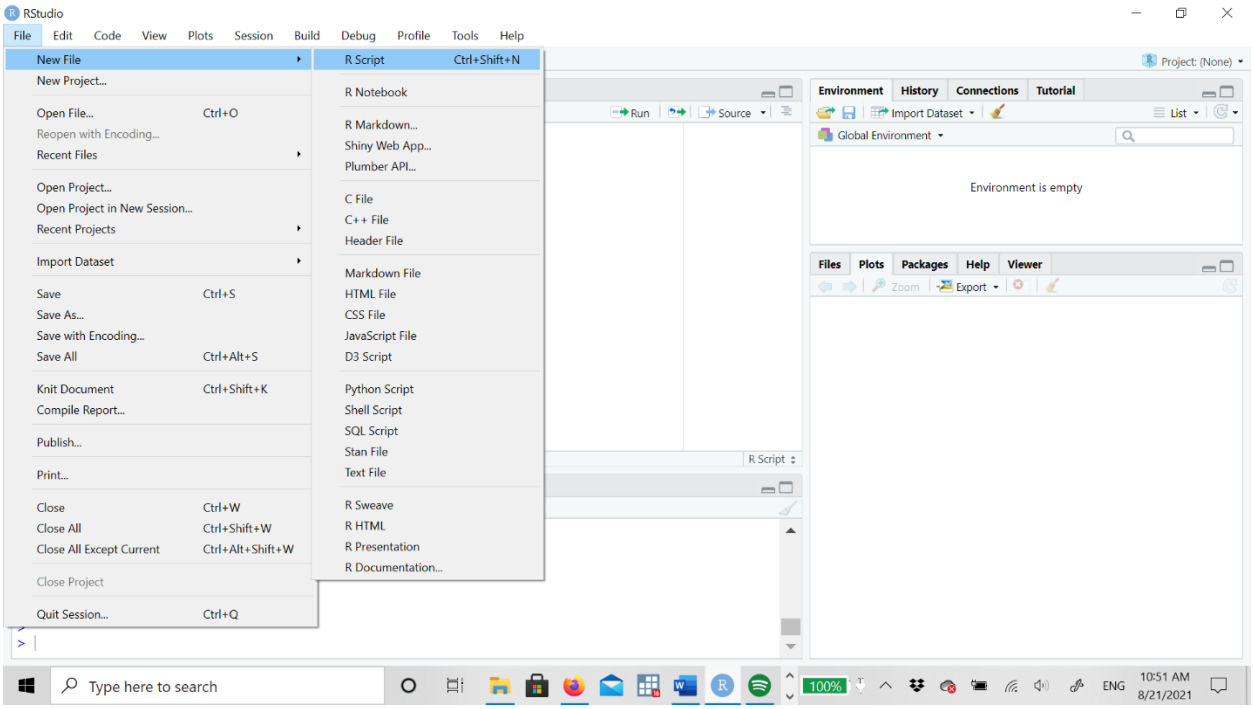
https://www.r-project.org/

https://www.rstudio.com/products/rstudio/download/

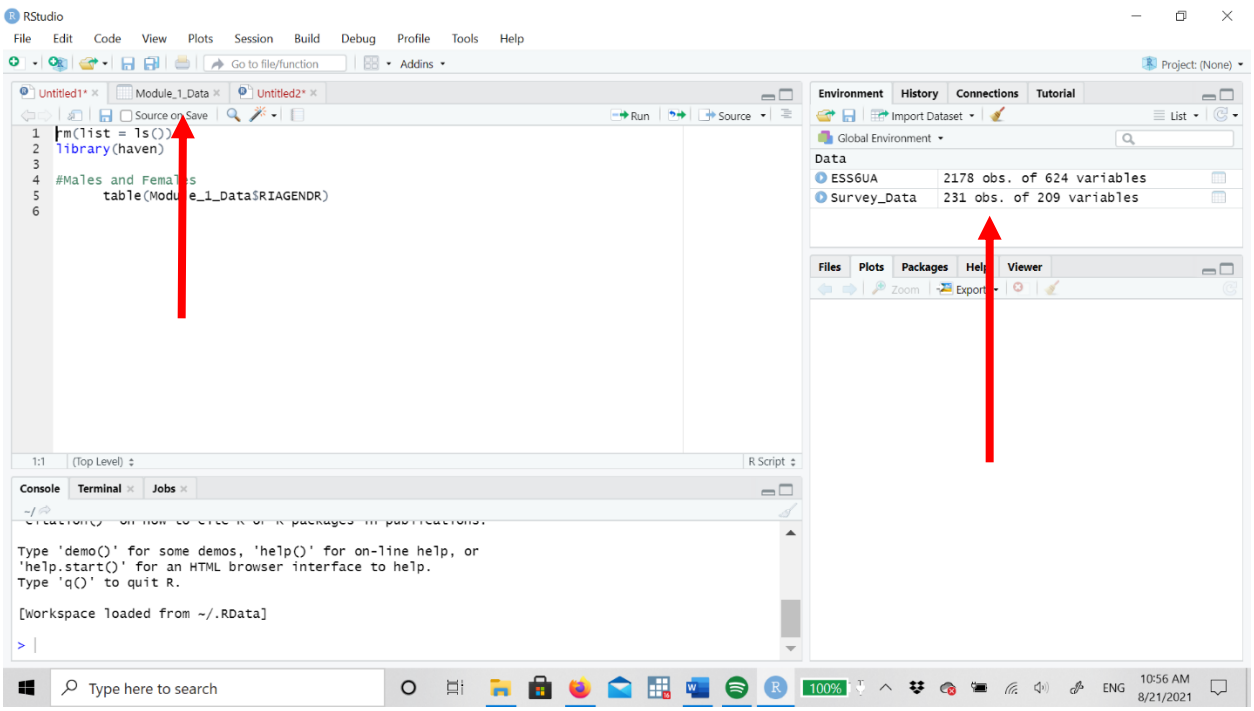
Getting Started:

There are a couple of tasks you should perform every time you open R-Studio: set your working directory, clear the environment, download libraries, and upload your data.

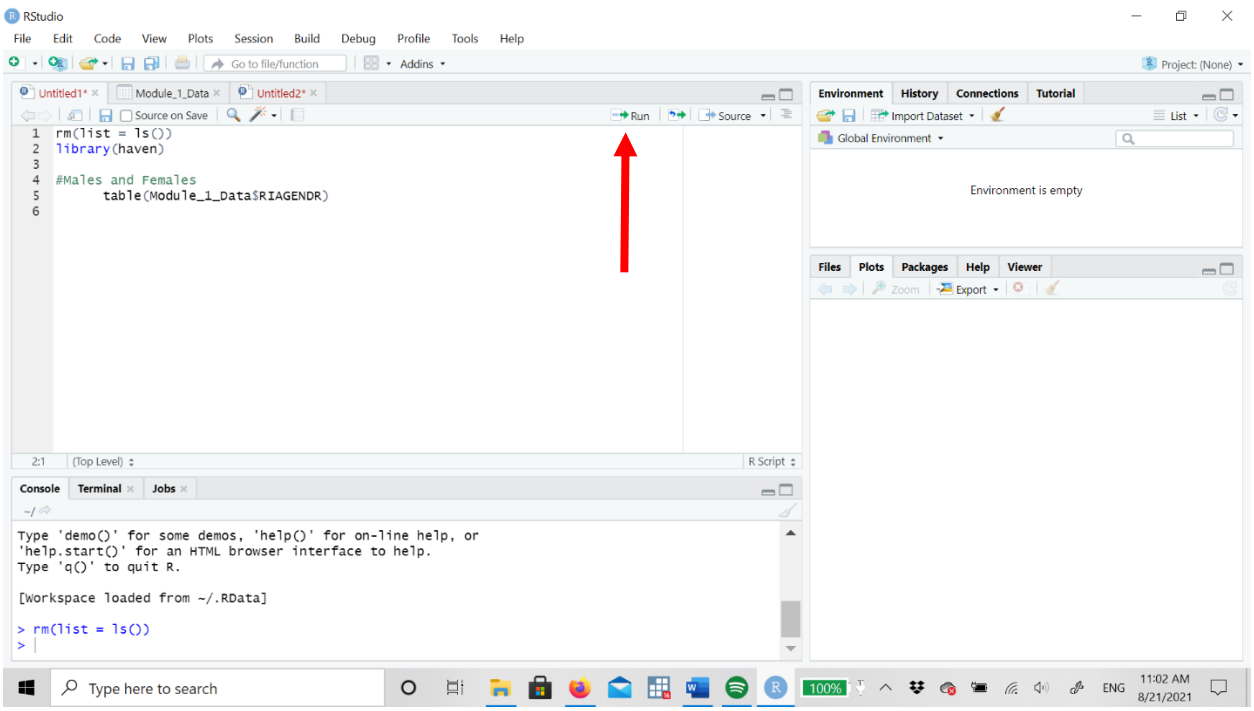
When you open up R-Studio it will automatically open up an R-script file where you can type commands, but if you want to open up another file you would go to the top left corner select file- new file- R script.



If you worked on and saved projects in R-Studio, R-Studio will automatically open you most recent R-Script file and dataset. For example, when I open R-Studio on my PC it pops up several R-Script files and several datasets in the Environment.



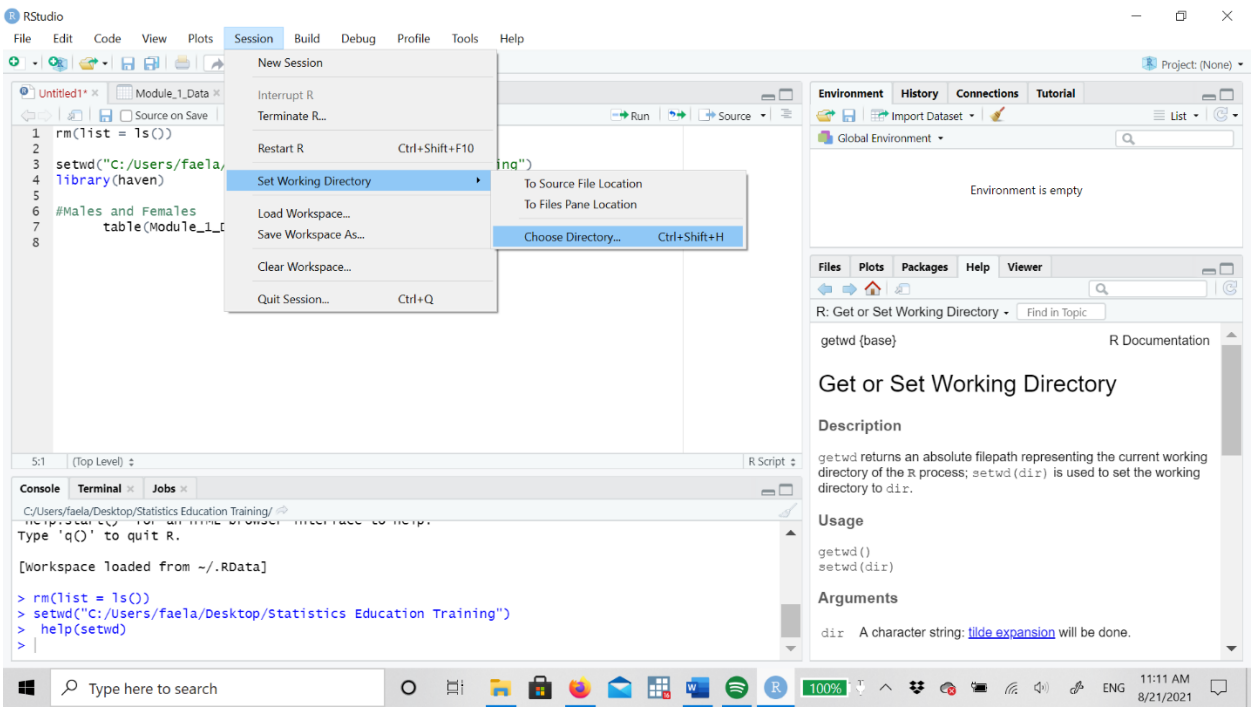
I'm no longer working with the datasets in the Environment so I use the `rm(list = ls())` command to delete all the files from the Environment. It is usually best practice to clear the Environment every time you start a new project, but be warned, if you have data or objects in the Environment and you run the rm command, they will disappear. So don't run it if you have anything you want to keep in the Environment. You run lines of code by putting your cursor at the beginning of the line you want to run and either selecting the "Run" button in the right corner of the script file or pressing ctrl+Enter.



Now that we have a clean working space, we can set our working directory. The working directory is where R looks to upload and save files. If you don't set your working directory, R will save your files wherever you saved last and you may not know where that is! So its always best practice to set your working directory before you do any coding. You can set the working directory by using the code:

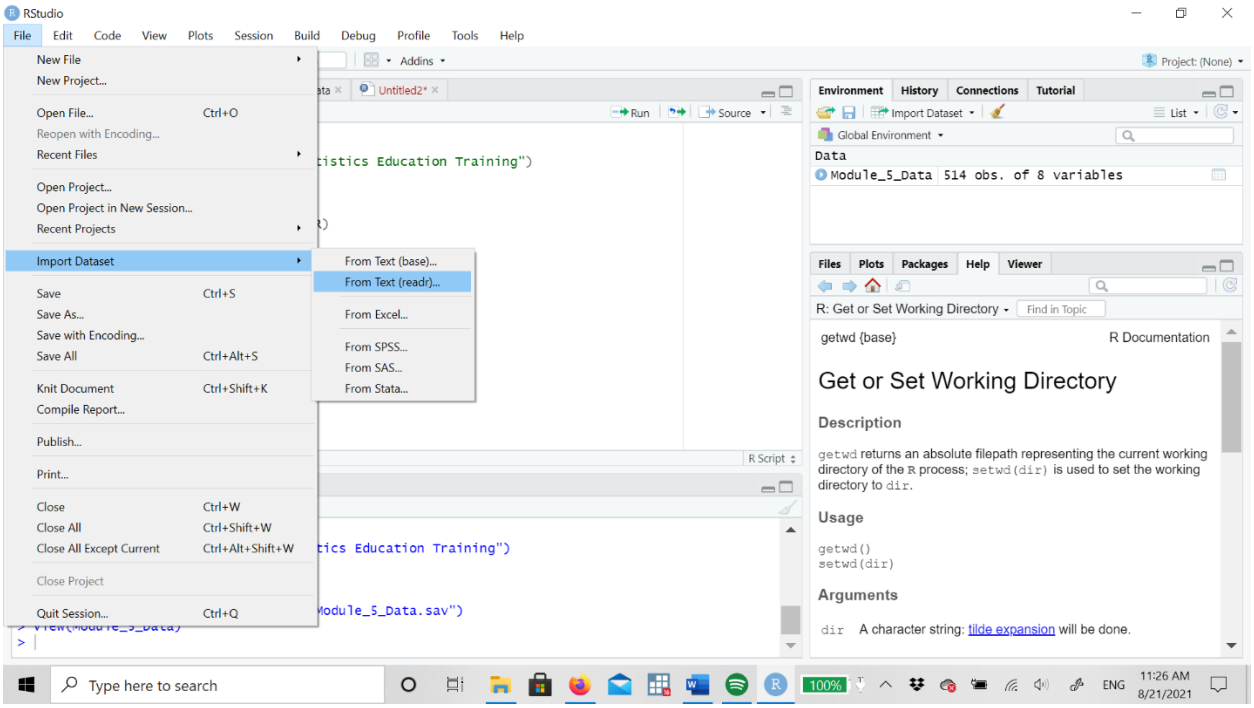
```
setwd("C:/Users/faela/Desktop/Statistics Education Training")
```

With this command I am telling R to look in drive C: to find the file Users then faela, Desktop, and finally Statistics Education Training. I would recommend creating a file organization scheme that will make sense to you. Don't just save all your R files, outputs, and datasets to a single file. It's easy to lose work that way. Instead create multiple files with useful file names to keep your work organized. You can also set the working directory by selecting Session, Set Working Directory, Choose Directory.

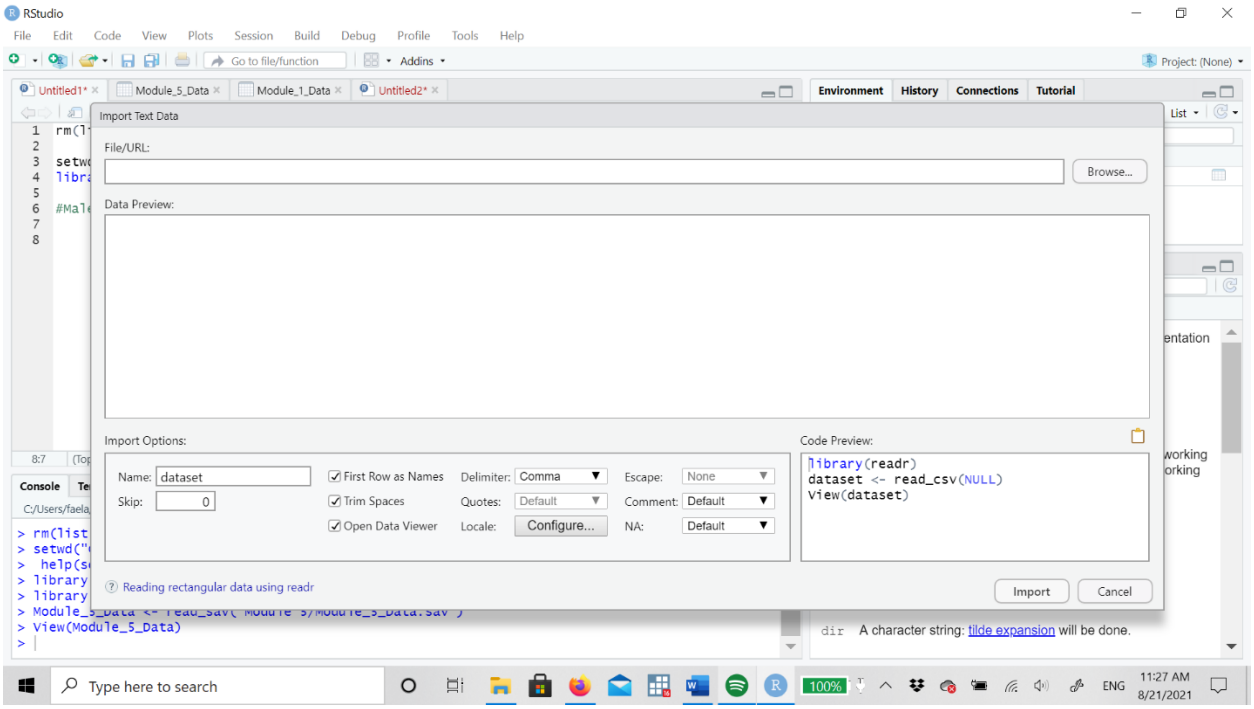


Now that we have a working directory we can start importing packages and libraries. Packages include multiple different libraries, and libraries contain specific commands you can use to perform tasks. R-Studio has a number of commands you can use without installing any packages (we refer to this as base R), but you will eventually need to perform tasks that R does not know how to do. Installing packages is like downloading new instructions for R. Base R may not know how to perform a task, so you're downloading a set of instructions with `install.packages` and then your telling R which instructions you want it to use with the `library()` command. You only need to install packages once and R will remember, but you will need to tell R which set of instructions you need (library) every time you open an R-script file. At the beginning of every module for the rest of the course, you will have a section that tells you if you need to install packages and which libraries will be necessary for the Module.

Now that we have our R set up we need to upload data! You can upload data two ways: the point and click method or the command method. With point and click you will go to file, Import Dataset, and then you will select the type of dataset you are working with (Excel, .csv, SPSS, etc).

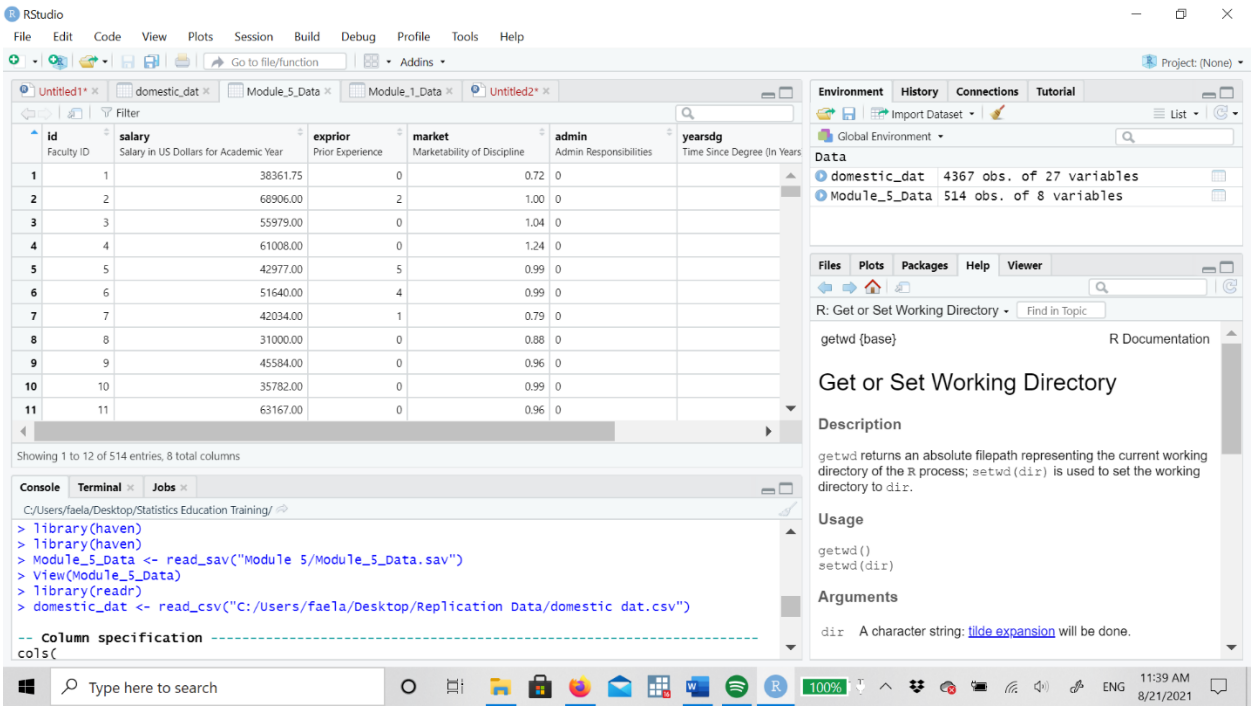


You will then get a box that will direct you to your folders. Remember, we set our working directory, so R will automatically go to that folder to look for the data you want to use. You can import data from other files, you will just have to navigate through your folders to find the file you want.



You can also tell R manually what file you want to upload and where to look for it. If you tell R `file_name <- read_csv("file name.csv")`, R will look in your working directory for a .csv file titled "file name" and save it as "file name" in the Environment. If you want R to look somewhere else for a file to import, you have to specify a path. Say I wanted to upload data from a completely different file, you have to tell R the path to look through just like when we told R what file path to go through when we set our working directory. For example, say I wanted to use data from a previous class that has a file on my desktop the command would look like `domestic_dat <- read_csv("C:/Users/faela/Desktop/Replication Data/domestic dat.csv")`. If you are importing data from anywhere other than your working directory, I would recommend the point and click method because file paths are complicated and if you spell a file wrong or capitalize something that isn't supposed to be capitalized, the command will not run.

Once you upload the data, R will automatically open the data as a tab in the main R screen.

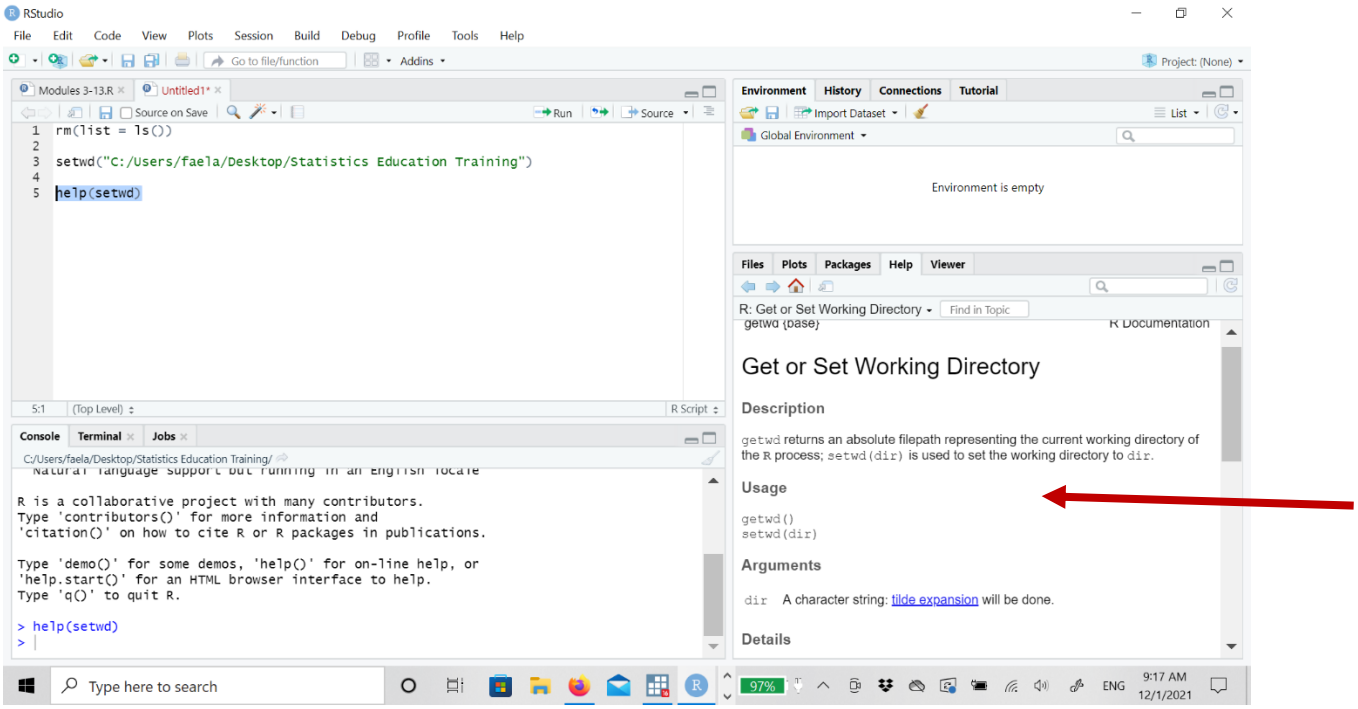


Feel free to close the data tabs if you get overwhelmed with tabs. If you ever want to view the data you can either click on the data file in the Environment or use the command `View(file name)`. You can also use the view command to look at specific variables. For example, you wanted to look at salary from the above dataset you would type `View(file name$variable name)`. The $ sign tells R which dataset to look in for the variable "variable name." If you were to type just `View(variable name)` R would pop up the message "object not found." The $ helps R know where to look for the variable you want, in some instances you wont need to specify which dataset to look in for the variable you want, but for now assume you always need to put `file name$variable name` for every variable specific command.

Also, note neither the file name or variable name are in quotations. There are instances when pieces of commands need to be in quotations and when they don't. As we go along in the

Modules, you will get feeling of when quotations are necessary, but when in doubt you can always use the `help()` command.

`Help()` is the command we use to find help about specific commands. Say we want help with the working directory command we used earlier, we would type out `help(setwd)` and run the command. The help will pop up in the plots section (bottom right window).



The help window that pops up will tell you the description of the command, the arguments, details, and will give some examples on how to run the command. The help function is very useful when you know the command but you don't understand the specifics of the syntax, but in many scenarios we may not even know the name of the command! For those scenarios, the internet is the most helpful tool you have. Because R is an open source software, there are many different options for getting help in R. Some websites I recommend for finding help:

https://www.r-project.org/

https://www.rdocumentation.org/

https://www.r-bloggers.com/

You can also just search for the task you need help with say "t-test in R" and you will get thousands of results to help you perform t-tests in R.

Here is a book that gives more information about introductory R:

https://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf

Final Helpful Hints for R:

- There are almost always multiple ways to perform functions in R, throughout these modules I will try to provide and easier and more difficult way to perform a task in R.
- Punctuation really matters in R. If you get an error message, sometimes the problem is you put a quotation mark where you should not have or you had a rogue parenthesis somewhere.
- R has a steep learning curve, don't be discouraged if you don't understand the functions right away. It takes time to learn how to code and once you understand how R works, it will be easier to code in R.
- Google and R Markdown will always be your friend when using R. If your having problems, the likelihood is someone else had those same problems and posted about them on the internet.
- Finally, and most importantly, practice! You can't learn R just by reading or watching videos, you have to execute commands. The more you practice using R, the more comfortable you will feel with R.