Introduction to R

## Learning Objectives

1. Be able to download and open R and RStudio.
2. Understand how to import data to R.
3. Understand how to create a graph in R.

[Link to Lecture](https://docs.google.com/presentation/d/1VJUOzvqMubmcRoJWAklv6ZlwM5WydgO7vyAwgsnPYsk/edit?usp=sharing)

Links to Tutorials

[Importing data into R](https://rpubs.com/jwesner/530743) [Transforming data from wide to long format (Tidying)](https://rpubs.com/jwesner/530744) [Visualization](https://rpubs.com/jwesner/530745) [Model - linear regression](https://rpubs.com/jwesner/530746) [Model with merged data (merging two data sets and running a linear regression)](https://rpubs.com/jwesner/531334)

# R and RStudio

1. Download R: (<https://r-project.org>)
2. Download RStudio: (<https://rstudio.com/products/rstudio/download>)

R is a free statistical and graphing program (R Core Team 2020). All of the examples in this book are generated using R. Actually, that’s not quite correct. While R is the workhorse, the examples in this book are generated through an interface to R called RStudio (RStudio Team 2020).

R looks like this:  <— *Download only*

RStudio looks like this:  <— *Download, open, and use*

Once you’ve downloaded both programs, you’ll only need to open RStudio. It automatically uses R in the background. It is possible to do everything only in base R, but we prefer RStudio as a more user-friendly interface.[[1]](#footnote-32).

# Why to use Coding Instead of Clicking

R is a programming language, which means that it can only do what you tell it to do by typing. RStudio has a few clickable shortcuts, but it still requires nearly everything to be typed into a script.

There are other programs that conduct statistical and graphical analyses without using code. We choose to use R instead for several reasons.

1. **Clicking isn’t Actually Easier** Undergraduates are incredibly savvy with some aspects of computers, particularly in nagivating social media platforms. But in our experience, students often struggle with even rudimentary tasks in programs that professors think are “easy”, such Microsoft Excel or SPSS. These programs have their own bewildering array of shortcuts and buttons. For example, while this Excel function might make perfect sense to a seasoned user - “=STDEV.P(A$1:A$7)" - it can be just as confusing to a new student as the similar function in R "\*sd(\*data$column*)*”.

Similarly, while it may seem easier to run an ANOVA in SPSS by simply clicking the ANOVA button, this too is often misleading. Having helped students that are partway through a project in SPSS or other clickable programs, we almost always have to start their entire analysis over when a problem arises. The reason is that, by the time the ANOVA button is clicked, there have already been a series of steps in data preparation and uploading that might have generated a problem. In R, we can find these problems easily, because the script leaves a breadcrumb trail of each step. In SPSS, we can’t. There are no breadcrumbs, so solving the problem becomes much more complicated.

1. **Data Ethics** A basic requirement in modern science is that the results of scientific findings could be reproduced by someone else. There are two levels to this. The first level is the description of the experimental approach, which is contained in a Methods section in a scientific publication. This ensure that someone else could read a Methods section and reproduce the steps of the experiment exactly without having to ask the author (who may no longer be alive or just doesn’t respond to email).

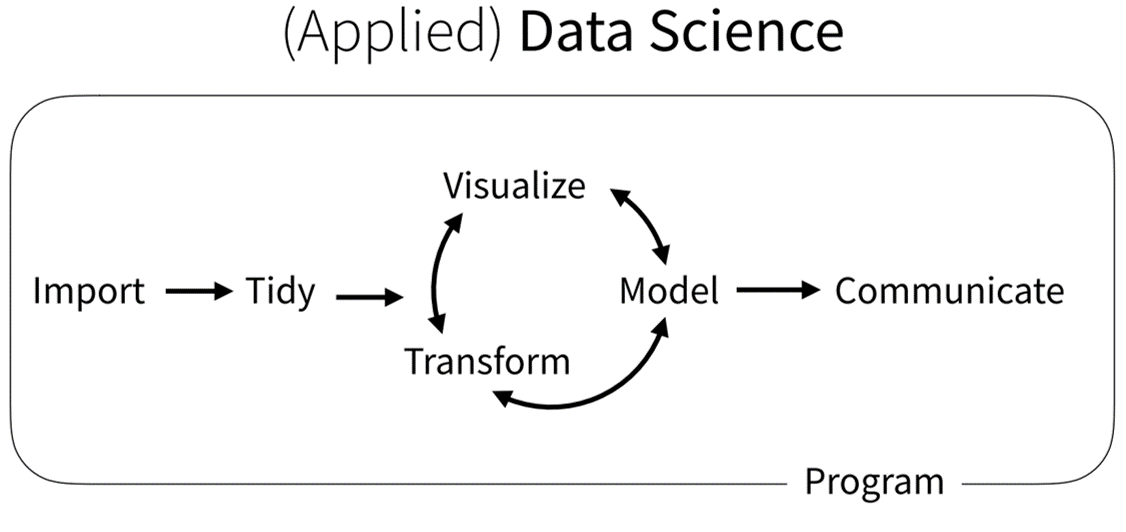
The second level of reproducibility is in the analysis of a dataset presented in a scientific publicatoin. All analyses involve myriad human decisions. For example, what do we do with outliers (extreme data values that may be real or may be a result of data entry error or errors in the instruments)? What if half of our fish died in the middle of an experiment. Should we replace them with new ones? There are no easy answers to these questions. Each experiment has its own quirks and they will all involve a subjective decision by the scientist.

What do we do about these subjective decisions? The golden rule is to be transparent about them in two ways. First, describe them in the Methods and provide a justification for them. Second, always include the raw data and make any modifications to the raw data transparent in the script. This is where using computer code over clicking makes a huge difference. If the raw data and script are available, then it is simple for someone else to run the analysis later and see the decisions you made about the data ‘quirks’. Different scientists will make different decisions about each of those ‘quirks’. The most important thing is not which decision is made per se, but that the trail of breadcrumbs exists to *allow* a decision to be known.

That may seem a little daunting. It is scary to have someone else see all of your decisions. But here’s the actual truth: The person who will benefit most from your transparent data and code is not another scientist. It is you. In two days, two months, or two years, you will eventually have to return to an analysis you did. You’ll need it to wrap up that semester’s term paper or to reanalyze something from your thesis. You will NOT NOT NOT NOT NOT remember what you have done, no matter how obvious it seemed when you were doing it. For that reason, having script that is reproducible will save you hours, maybe weeks, of otherwise wasted time. Trust us…just trust us.

# Some Tips for First Time R Users

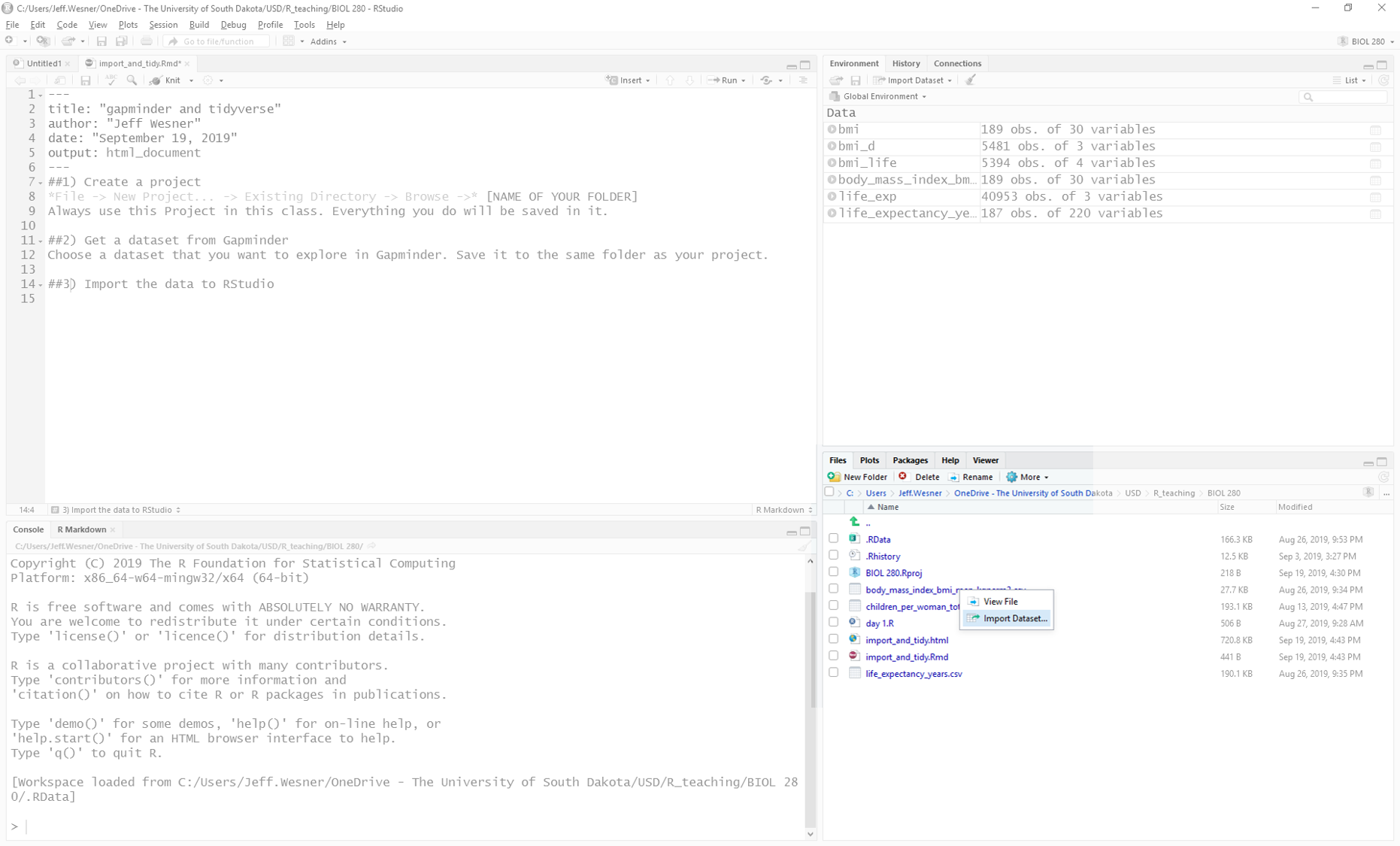
We assume that users of this book are undergraduate science majors who have little or no eperience in coding. Having taught ~500 students how to produce realistic analyses and publication quality graphics, we are confident that you can learn it, regardless of your background. We are also confident that it will be frustrating at times, particularly early on.

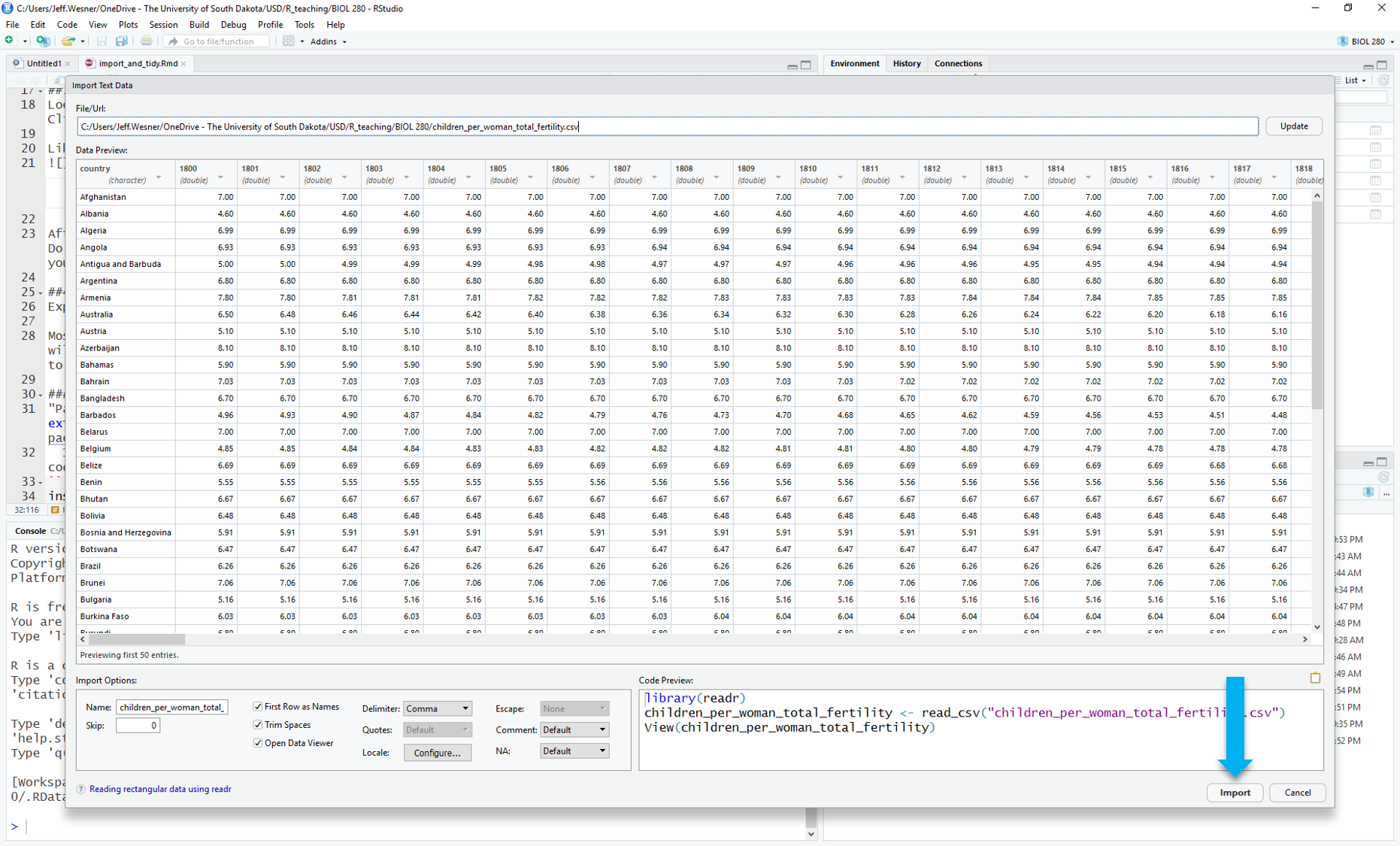
 Nearly every study that includes data has a workflow similar to that above. We gather data, get it into a program (Import), get it in the right format (Tidy), and then analyze it with plots (Visualiation), Models, Transformations, etc. When we’ve finished, we communicate the results to our peers. You’ll learn how to complete these steps in R because its designed specifically for each of these steps. But the workflow applies regardless of the software you use.

##1) Create a project *File -> New Project… -> Existing Directory -> Browse ->* [NAME OF YOUR FOLDER] Always use this Project in this class. Everything you do will be saved in it.

##2) Get data from Gapminder Choose a dataset that you want to explore in Gapminder. Save it to the same folder as your project.

##3) Import the data to RStudio Look in the lower right panel of RStudio. Click on “Files”. Do you see the Gapminder data you saved? Click on your dataset and choose *Import Dataset*.

Like this: 

After you click *Import Dataset*, you’ll see a preview of your dataset like this: 

Click *Import* in the lower right. (hint: you might want to give your data a shorter name before you import. Do that in the lower left of the pop-up window.) Do you see the name of your dataset in the upper right panel? If so, success! If not, re-try the steps above or ask your instructor.

R Core Team. 2020. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. <https://www.R-project.org/>.

1. Except for JR, an English professor who doesn’t understand any of this. We assume he is currently pontificating about the literary importance of using *salve* versus *halve* in the writings of Chaucer (who uses neither word). JR has a large collection of feathered pens and prefers to write on low gloss paper sourced from the Pacific Northeast. [↑](#footnote-ref-32)