Why R?

David John Baker

12/23/2018

Why R?

Before starting out, it's worth mentioning that R has a steep learning curve compared to other statistical softwares. While there are tons of blog posts as to why you should learn R, I will keep my list quick so if you get discouraged at any point, you can come back to this list and get re-inspired before R starts paying you back.

1. The R community is fantastic, check out #rstats on Twitter as well as everyone affiliated with the tidyVerse
2. R will always be free because the people behind it believe in open source principles.
3. Time spent learning R is time spent learning how computers work. If you learn about R, you are also learning computer programming. Time spent in something like SPSS or SAS does not easily transfer to other programs.
4. On r-jobs.com the way they decide to split jobs is jobs that make above and below $100,000.
5. R is your ticket out of academia, if you need it. It's also insane to think people would learn so much about statistics, the hardest part about becoming a data scientist, without learning the software to get you in the door.
6. When you make analyses and graphs in R they are very easy to reproduce. You just press 'Run' again.
7. If you do your data cleaning in R, then each step is documented. There is less chance for human error.
8. It makes gorgeous graphs.
9. There are a lot of ways that R integrates into other software. This book is written in bookdown, my website is written in blogdown, you can also make interactive data applications.
10. It's kind of fun.

R, RStudio, and Tidyverse

R

You can download R for your computer by going to CRAN and selecting the appropriate `Download and Install R` links. Make sure to install R first before installing RStudio.

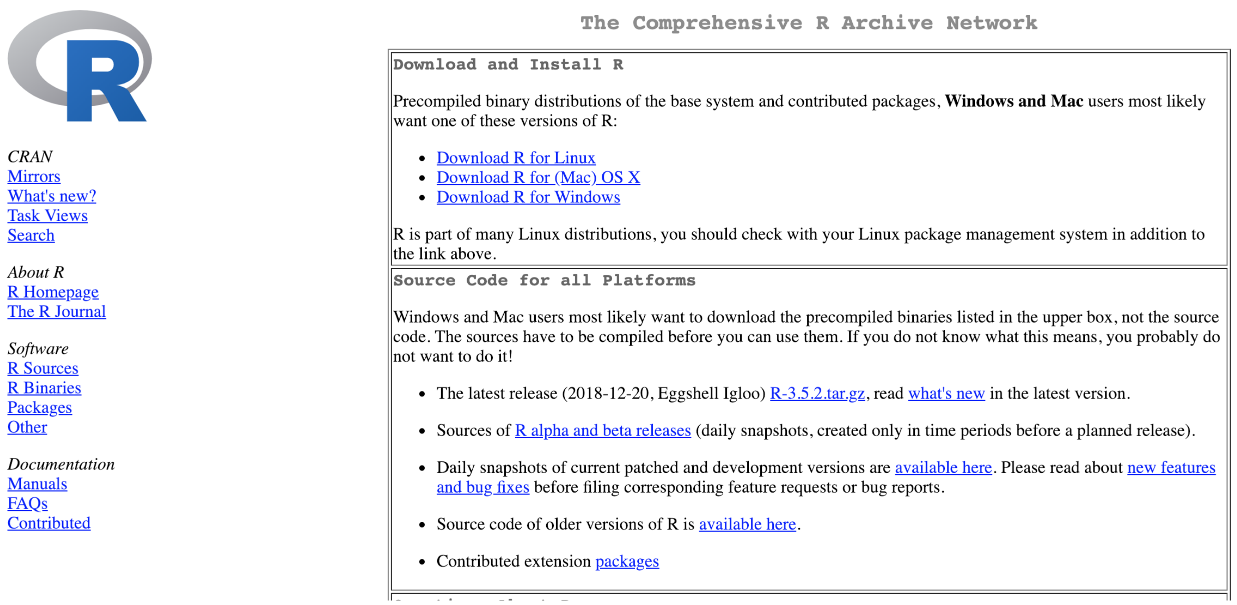
RStudio

RStudio is an intergrated development environment (IDE) for R[1]. RStudio is basically your workbench where you can access everything you need for managing your scripts, data, and project structure. By using RStudio, you also can use a host of other features ranging from Markdown documents (like this one!), interactive data dashboards like Shiny, and the tidyverse.

Packages and the tidyverse

What makes R different than programs like Excel or SPSS is that R as you download it does not come with everything installed. Downloading R from CRAN gives you what is referred to as **base R**. The idea is that since you do not need all the software, all the time, you just load in what you need. R can be used just by

[1]https://www.rstudio.com/products/RStudio/


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

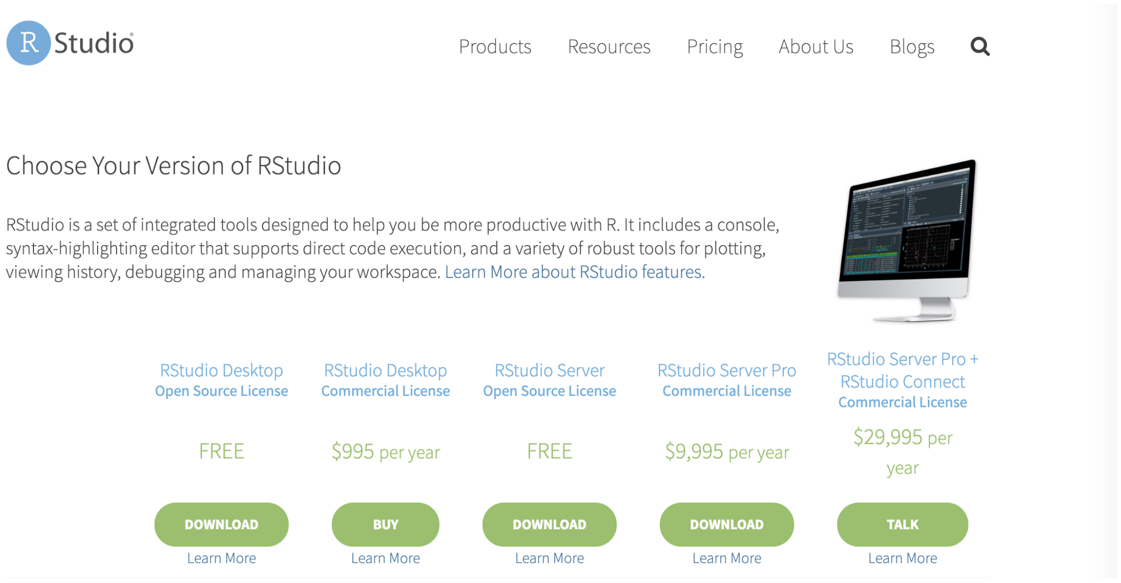
R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

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 (2018-12-20, Eggshell Igloo) R-3.5.2.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 corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Figure 1: CRAN Homepage



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	RStudio Server Pro + RStudio Connect Commercial License
FREE	$995 per year	FREE	$9,995 per year	$29,995 per year
DOWNLOAD	BUY	DOWNLOAD	DOWNLOAD	TALK
Learn More	Learn More	Learn More	Learn More	Learn More

Figure 2: RStudio

itself, but the real advantage of it is that so many people use it and write software for it, if there is something that you need to do that is somewhat common, chances are that someone has written software for it already. The external software that you load in are referred to as **packages** which are kept in your **library**. Some packages are very small and simple, others have extensive teams developing them. One of the most important packages in R is the tidyverse.

The tidyverse is a collection of packages that were developed to make manipulating data more intuitive. As noted on its homepage,

> All packages share an underlying design philophy, grammar, and data structures.

While this might seem trivial now, having your data be `tidy` opens up an entire world of data manipulation and modeling. Once you you get over the initial learning curve of R, the tidyverse makes it so that you can pretty much take off and learn very q quickly.

Quiz

- See Slies

RStudio Environment

Once you now have R and RStudio installed, it's time to open up RStudio. By opening RStudio, you are also starting R. R will be running under the hood of RStudio. After installing R, you can run it on it's own by typing `R` into your terminal on a Unix machine (Mac, Linux). Though after seeing how RStudio works, you would realize why doing this is basically mascochistic. (If you do this, you can quit out of the terminal R with `quit()` followed by `n`).

When you first open RStudio will see a few different panels. In it's default settings, the bottom left is the Console. The top right has your Environment, History, and version control commands. The bottom right has your Viewer, Library for your packages, and a system to navigate your files. The top left will be where you write your code.

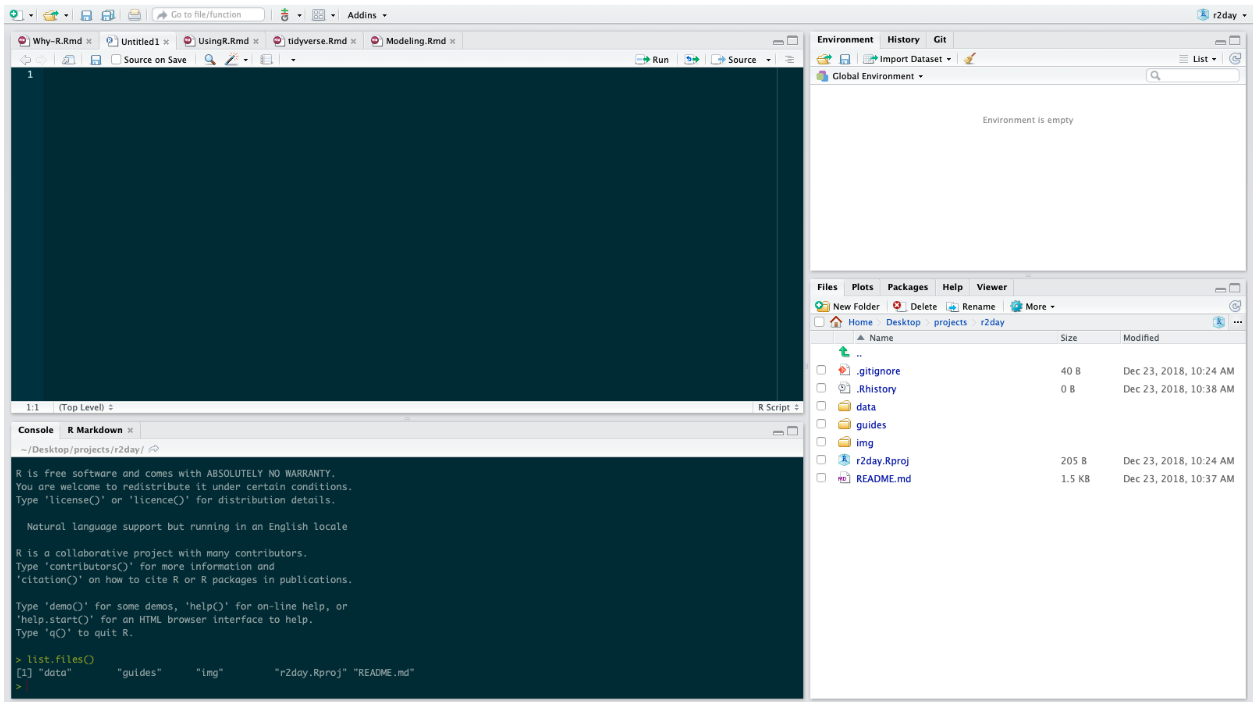
Environment

The top left has information about your current Environment. As you make new things in an R session you can track them here. There is also a History tab here that keeps track of code you wrote. Additionally there is a Git tab that will eventually allow you to do version control. You don't have to know what that is, but one day you might read about it.

Viewer

The bottom right is your File Explorer/Finder window. Try to click around on the **Files** tab. When you click **Plots** there should be nothing there as you have not made any plots yet. Your **Packages** tab will have a listing of software that you can load into R. Notice that if you click one of the package names, it will navigate you to the **Help** tab. Lastly, the Viewer tab will let you display any documents that you make while writing in R. This could be markdown documents or maybe a website that you are writing eventually.

> It is important to note that you will probably "break" R and RStudio many times when learning. Know that this is OK and the some of the best advice for learning how to program is by just seeing what happens when you change something and Googling your problems.



```
~/Desktop/projects/r2day/
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.

> list.files()
[1] "data"        "guides"      "img"         "r2day.Rproj" "README.md"
>
```

Name	Size	Modified
..		
.gitignore	40 B	Dec 23, 2018, 10:24 AM
.Rhistory	0 B	Dec 23, 2018, 10:38 AM
data		
guides		
img		
r2day.Rproj	205 B	Dec 23, 2018, 10:24 AM
README.md	1.5 KB	Dec 23, 2018, 10:37 AM

Figure 3: RStudio Environment

Console

The Console in R is where you can run one-off R commands. Try to type a few of to following commands into the Console.

```
list.files()
```

```
## [1] "Modeling.Rmd"  "tidyverse.Rmd" "UsingR.Rmd"    "Why-R.html"
## [5] "Why-R.Rmd"
```

```
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
2 + 2
```

```
## [1] 4
```

Each of these will create a different kind of output. Now try to put something in your R console that will create an error message. Maybe some math that ends with an operation sign? Maybe some text? In the next session, we will go over what is legal and illegal input in the R Console.

Editor

The top left panel is where you edit your documents. RStudio allows you to handle many different types of documents. In this course we will mostly use RMarkdown files. These files end in `.Rmd` and allow for both text and R code. R scripts on the other hand only handle R code.

Using the editor, you should also familiarze yourself with the keyboard shortcuts in RStudio. For example, to run a line of code in the Editor, you can press `CMD + ENTER` on Mac or `CTRL + ENTER` on Windows. When the cursor is on a line that has runable R code, this will run that line in the console. You can also use your mouse highlight many lines of R code an run the same commands. We will get a lot of practice with this in the next session.

Theory of Workflow

One of the advantages of R is that you can basically place your entire workflow in suspended animation. If this does not make sense, right away, let's imagine an example analysis.

1. You need to make a plot of your company's yearly earnings.
2. You open up Excel and load in your data.
3. You perform some serious clicking on the spreadsheet to make the nessecary analysis and resulting plots.
4. You finishing your analysis, save your graph, and send it to your boss.
5. He asks you to re-do it with the updated data file he forgot to send you.
6. You repeat all the clicks again to get the new plot.

This might be familiar to some people. The problem with this situation is that to do the analysis, you have to perform each click again yourself. With R, the idea is that you basically write out each click you do as a line of code. While this might seem tedious, the more complex the program or analysis, the more you want to have documented and mainstreamed.

Scripts

The idea behind writing code is that you are saving yourself having to re-do by hand tons of clicks within a software. Not only that, but you can re-run code many times over and don't even have to be there to do it. You just press GO and the computer does the work itself.

** Quiz II