Quantitative Politics with R

Erik Gahner Larsen and Zoltán Fazekas

Februar 06, 2018

Table of Contents

Chapter 1

Introduction

If you want to conduct quantitative analyses of political phenomena, `R` is by far the best software you can use. Importantly, data analysis is no longer restricted to analyzing survey data, but does now include social media data, texts, images, geographic data (*GIS*), and so forth.

In this book, we aim to provide an easily accessible introduction to `R` for the study of different types of data. The book will teach you how to get different types of data into `R` and manipulate, analyze and visualize the data.

Compared to other statistical softwares, such as Excel, SPSS, Stata and SAS, you will experience that `R` is completely different. First in a bad way: things are not as easy as they used to be. Then in a good way: once you learn how to do different tasks in `R`, you will be ashamed when you look back at the old you doing analyses in SPSS.

In this chapter you will find an introduction to `R`. First, we ask the obvious and important question, why R? Second, we help you install what you need. Third, we introduce you to the basic logic of `R` so you are ready for the chapters to come.

1.1 Why `R`?

First, `R` is an *open source* statistical programming language. `R` is free, and while you might not pay for Stata or SPSS because you are a student, you will not have free access to this forever. This is not the case with `R`. On the contrary, you will *never* have to pay for `R`.

Second, `R` provides a series of opportunities you don't have in SPSS and Stata. `R` has an impressive package ecosystem on CRAN (the **c**omprehensive **R** **a**rchive **n**etwork) with more than 12,000 packages created by other users of `R`.

Third, some of the most beautiful figures you will find today are created in `R`. Big media outlets such as The New York Times and FiveThirtyEight use `R` to create figures. In particular the package `ggplot2` is popular to create figures and we will work with this package below.

Fourth, there is a great community of `R` users that are able to help you when you encounter a problem (which you undoubtly will). `R` is a very popular software and in great demand meaning that you will not be the first (nor the last) to experience specific issues in your data analysis. Accordingly, you will find a lot of help on Google and other places to a much greater extent than for other types of software.

Fifth, while you can't do as much point-and-click as in SPSS and Stata, this approach facilitates that you can reproduce your work. When you are doing something i `R` with commands (in a script) is it easy to document. So, while you do not see a pedagogical graphical user interface in `R` with a limited set of buttons to click, this is more of an advantage than a limitation.

1.2 Installing `R`

To install the `R`, you will have to install 1) the `R` language and 2) RStudio, the graphical user interface. To install the `R` language, follow this procedure:

1. Go to https://cloud.r-project.org.
2. Click *Download R for Windows* if you use Windows or *Download R for (Mac) OS X* if you use Mac.

If you use Windows:

3. Click on *base*.
4. Click the top link where you can download `R` for Windows.
5. Follow the installation guide.

If you use Mac:

3. Select the most recent `.pkg` file under *Files:* that fits your OS X.
4. Follow the installation guide.

If you encounter problems with the installation guide, make sure that you did download the correct file *and* that your computer meets the requirements. If you did this and still encounter problems, you should get an error message you can type into Google and find relevant information on what to do.

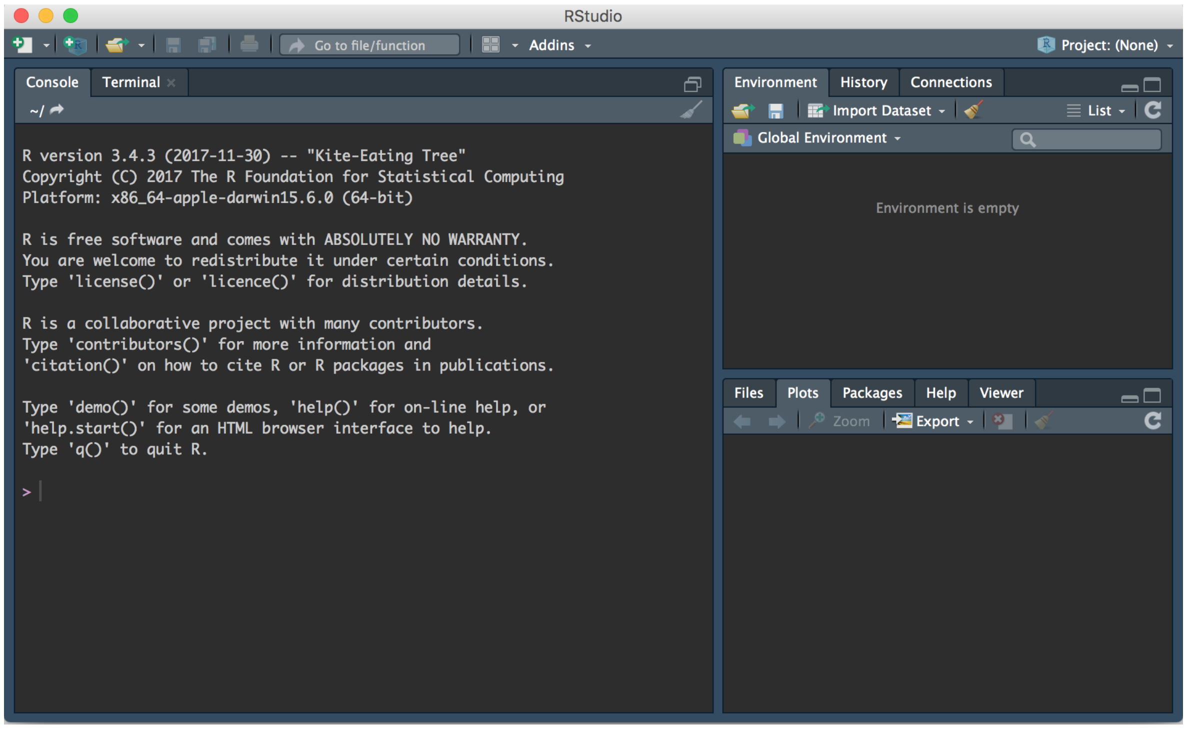
You should now have the R language installed on your computer.

1.3 Installing RStudio

RStudio is an integrated development environment (IDE) and makes it much easier to work in R compared to the standard ("base") R. This is also available for free. To install RStudio, follow these steps:

1. Go to: https://www.rstudio.com/products/rstudio/download/#download.
2. Click on the installer file for your platform, e.g. Windows or Mac OS X.
3. Follow the installation guide.

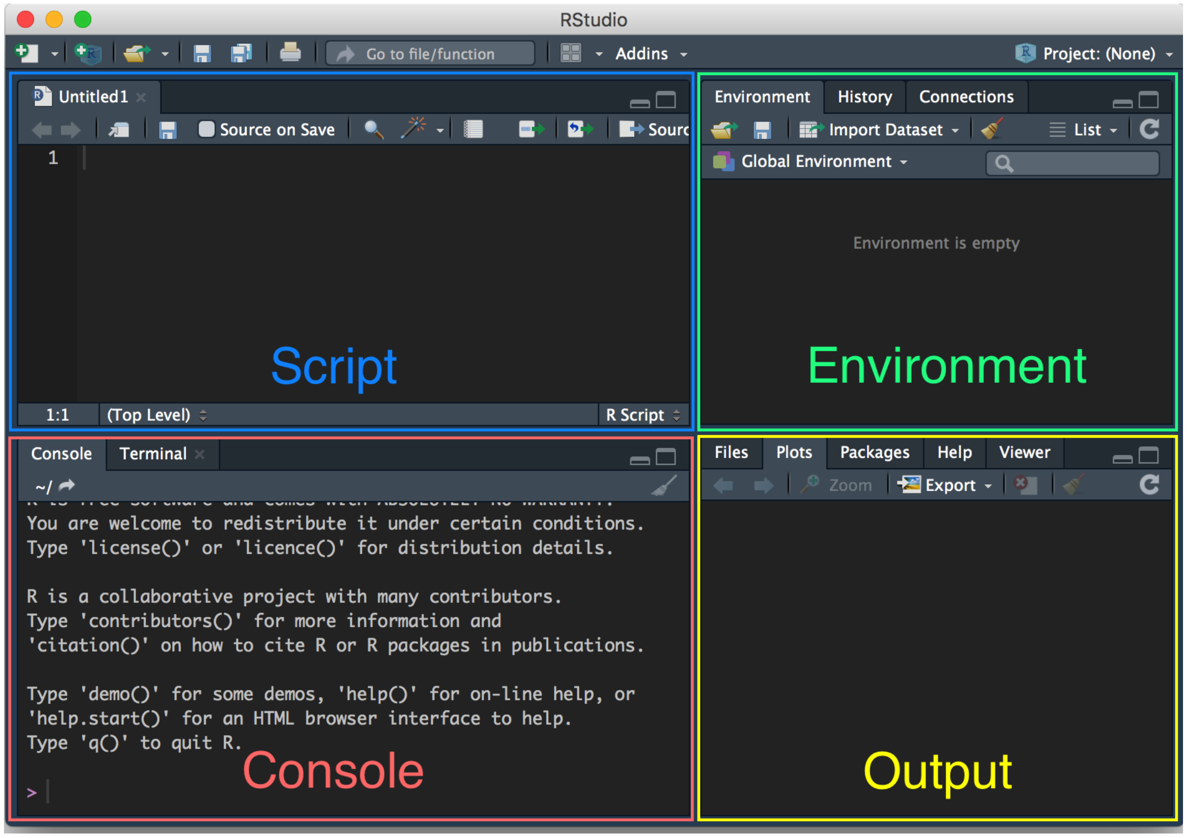
You should now have RStudio installed on your computer. When you open R you will see a graphical interface as in Figure 1.1.



```
R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

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.

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.

>
```

Figure 1.1: Graphical interface in RStudio

There are three different windows. However, one is missing, and that is the window where you will write most of your scripts. You can get this window by going to the top menu and select `File` → `New File` → `R Script`. This should give you four windows as shown in Figure 1.2.



```
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

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.

>
```

Figure 1.2: Graphical interface in RStudio, explained

In the figure, we have emphasized the four windows: script, environment, output, and console. The *script* is where you will have your `R` code and make changes. The *environment* is where you can see what datasets, variables and other parts you have loaded into R. The *output* is where you can see figures you create. The *console* is where you can see some output and run commands.

Everything you do in `R` can be written as commands. This ensures that you will always be able to document your work (in your script). In the console, you can see a prompt (`>`). Here, you can write what what you want `R` to do. Try to write `2+2` and hit `Enter`. This should look like this:

```r
2+2
```

```
[1] 4
```

The code you have entered in the console cannot be traced later. Accordingly, you will have to save the commands you want to keep in the script. Even better, you should write your commands in the script and "run" them from there. If you write `2+2` in the script, you can mark it and press `CTRL+R` (Windows) or `CMD+ENTER` (Mac). Then it will run the part of the script that is marked in the console. Insert the code below in your script and run it in the console:

```
50*149
3**2          # 3^2
2**3          # 2^3
sqrt(81)      # 81^0.5
```

As you can see, we have used `#` as well. The `#` sign tells `R` that everything after that sign on that line shouldn't be read as code but as a comment. In other words, you can write comments in your script that will help you remember what you are doing - and help others understand the meaning of your script. For now, remember to document everything you do in your script.

Notice also that we use a function in the bottom, namely `sqrt()`. A lot of what we will be doing in `R` is with functions. For example, to calculate a mean later we will use the `mean()` function. In the next section we will use functions to install and load packages.

1.4 Installing R packages

We highlighted above that one of the key advantages of using `R` is the package system. In `R`, a package is a collection of data and functions that makes it easier for you do to what you want. The sky is the limit and the only thing you need to learn know is how to install and load packages.

To install packages, you will have to use a function called `install.packages()`. We will install a package that installs a lot of the functions we will be using to manipulate and visualise data. More specifically, we will work within the tidyverse (you can read more at tidyverse.org). To intall this package type:

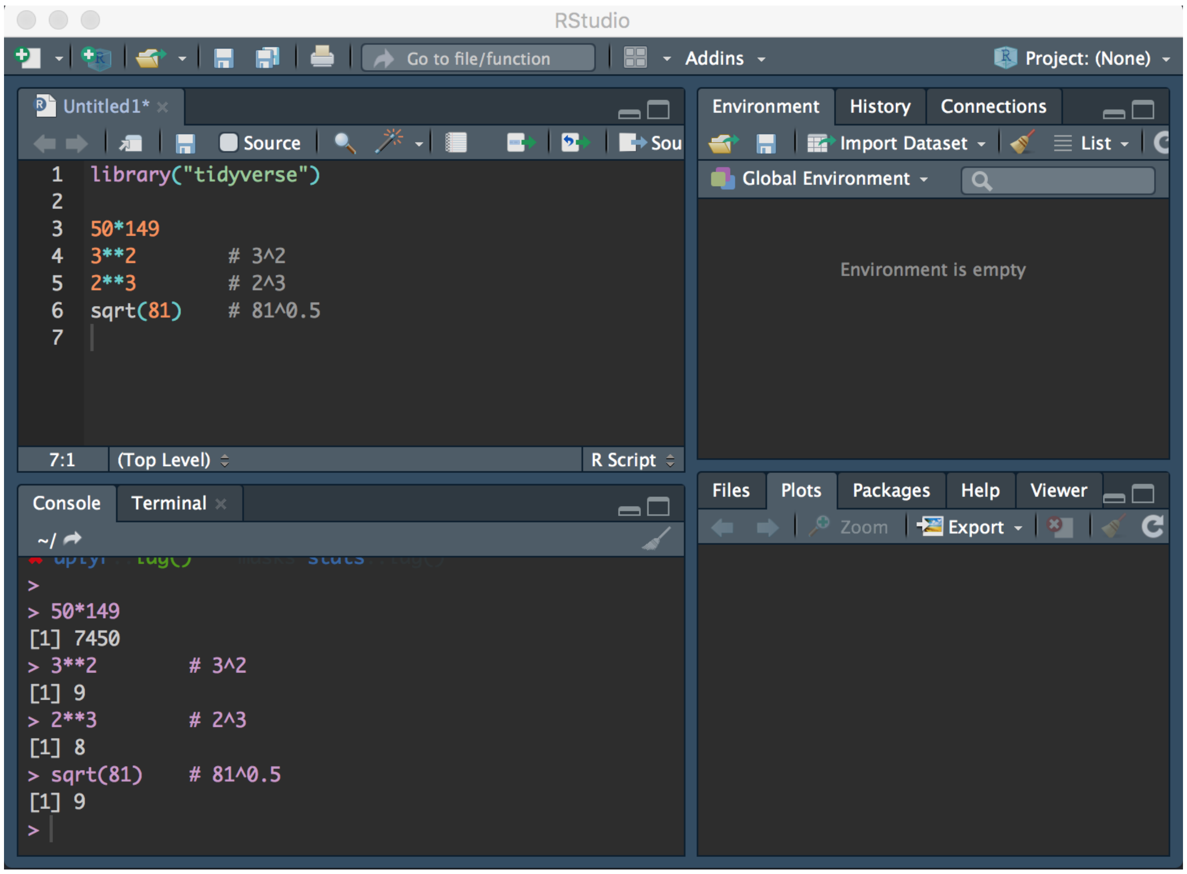
```
install.packages("tidyverse")
```

You only need to install the package once. In other words, when you have used `install.packages()` to install a packagae, you will not need to install that specific package again. Note that we put `tidyverse` in quotation marks. This is important when you install a package. If you forget this, you will get an error.

While you only need to install a package once, you need to load the package every time you open `R`. This is a good thing as you don't want to have all your installed `R` packages working at the same time. For this reason, most scripts begin with loading the packages that is needed. To load a package, we use the function `library()`:

```
library("tidyverse")
```

To recap, it is always a good idea to begin your script with the package(s) you will be working with. If we want to have a script where we load the `tidyverse` package and have some of the commands we ran above, the script could look like the script presented in Figure 1.3.



```
library("tidyverse")

50*149
3**2         # 3^2
2**3         # 2^3
sqrt(81)     # 81^0.5
```

```
>
> 50*149
[1] 7450
> 3**2         # 3^2
[1] 9
> 2**3         # 2^3
[1] 8
> sqrt(81)     # 81^0.5
[1] 9
>
```

Figure 1.3: A script in RStudio

If you want to save your script you can select `File` → `Save`, where you can pick a destination for your script.

1.5 Errors and help

As noted above, you will encounter problems and issues when you do stuff in `R`. Sadly, there are many potential reasons to why your script might not be working. Your version of `R` or/and RStudio might be too old or too new, you might be using a function that has a mistake, you might not have the data in the right format etc.

Consequently, we cannot provide a comprehensive list of errors you might get. The best thing to do is to learn how to find help online. Here, the best advice is to use Google and, when you search for help, always remember to mention R in your search string, and, if you are having problems with a specific package, also the name of the package.