Very Brief Intro to R with R Markdown

Cheng Peng

1/10/2021

Contents

0.0.1 Introduction

This brief note will introduce the basics of Rstudio, R Markdown, and R.

- **RStudio** is an integrated development environment (IDE) for R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and work space management.
- **R Markdown** is a file format for making dynamic documents with R. An R Markdown document is written in markdown (an easy-to-write plain text format) and contains chunks of embedded R code and the output generated from the R code. This note is written in R Markdown. This is also a tutorial showing how to use R Markdown to write an R Markdown report. – RStudio documentation.
- **R** is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can be considered as a different implementation of S. There are some important differences, but much code written for S runs unaltered under R.

0.0.2 RStudio GUI

The RStudio interface consists of several windows. I insert an image of a regular RStudio GUI.

```
![RStudio GUI](https://datacarpentry.org/r-intro-geospatial/fig/01-rstudio.png)
```

0.0.2.1 Console We can type commands directly into the console, or write in a text file, and then send the command to the console. It is convenient to use the console if your task involves on one line of code. Otherwise, we should always use an editor to write code and then run the code in the Console.

0.0.2.2 Source Editor Generally we will want to write programs longer than a few lines. The Source Editor can help you open, edit and execute these programs.

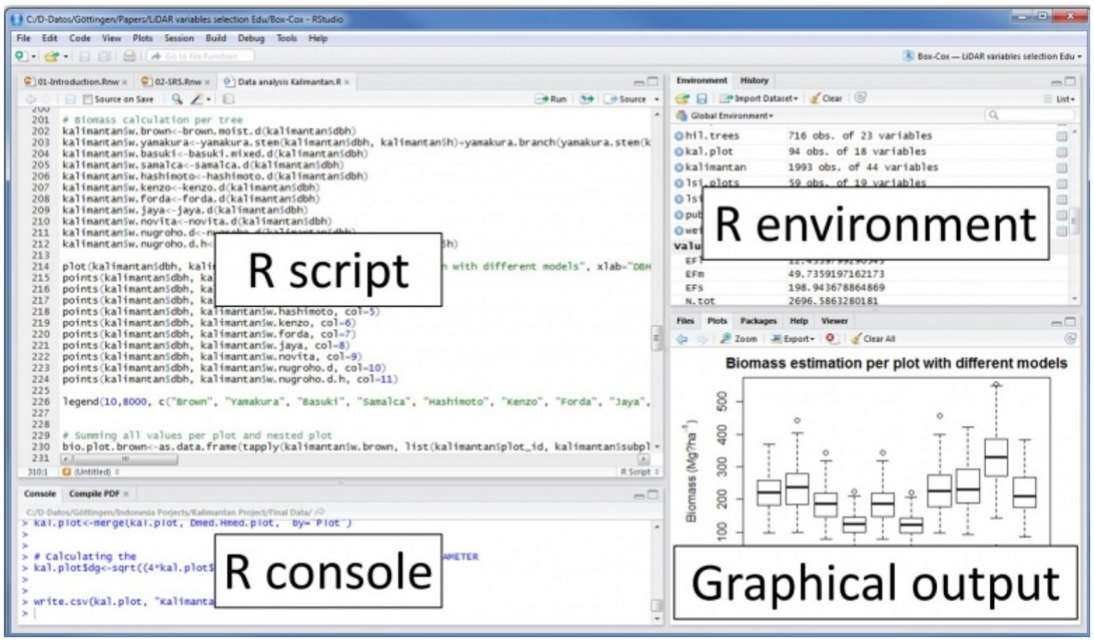


Figure 1: List of all variables and the description of each variable

0.0.2.3 Environment Window The Environment window shows the objects (i.e., dataframes, arrays, values and functions) in the environment (workspace). We can see the descriptive information such as types as dimension of the objects in your environment. We also choose data source from the environment to view in the source window like a spreadsheet.

0.0.2.4 System and Graphic files The Files tab has a navigable file manager, just like the file system on your operating system. The Plot tab is where graphics you create will appear. The Packages tab shows you the packages that are installed and those that can be installed (more on this just now). The Help tab allows you to search the R documentation for help and is where the help appears when you ask for it from the Console.

0.0.3 R Markdown

An R Markdown document is a text based file format that allows you to include both descriptive text, code blocks and code output. It can be converted to other types of files such as PDF, HTML, and WORD that can include code, plots, outputs generated from the code chunks.

0.0.3.1 Code Chunk In R Markdown, we can embed R code in the code chunk defined by symbol ```` ```{} ```` and closed by ```` ``` ````. The symbol ‘, also called **backquote** or **backtick**,can be found on the top left corner of the standard keyboard as shown in the following.

There are two code chunks: executable and non-executable chunks. The following code chunk is non-executable since is no argument specified in the `{}`.

```
This is a code chunk
```

To write a code chunk that will be executed, we can simply put letter `r` inside the curly bracket. If the code the code chunk is executable, you will the green arrow on the top-right corner of the chunk.

We can define R objects with and without any outputs. In the above R code chunk, we define an R object under name `x` and assign value 5 to `x` (the fist line of the code). We also request an output that prints the value of `x`. The above executable code chunk give output `[1] 5` in the Markdown document. The same output in the knit output files is in a box with a transparent background in the form `## [1] 5`.

Figure 2: The location of backquote on the standard keyboard

```
```{}
This is a code chunk
```
```

Figure 3: Non-executable code chunk.

```
x = 5
x
```

```
## [1] 5
```

We can also use argument in the code chunk to control the output. For example, the following code chunk will be evaluate when kitting to other format of files. But we can still click the green arrow inside the code chunk to evaluate the code.

```
x = 5
x
```

0.0.3.2 Graphics Generated from R Code Chunks Any function (other than interacting with plots, etc.) that we can realize in R, we can put in our markdown file & knit. For instance, we can generate following image from R and include in the Markdown document and in the knitter output files.

```
library(phytools)
data(anoletree)
plotTree(as.phylo(anoletree), type="fan", lwd=1, fsize=0.7)
```

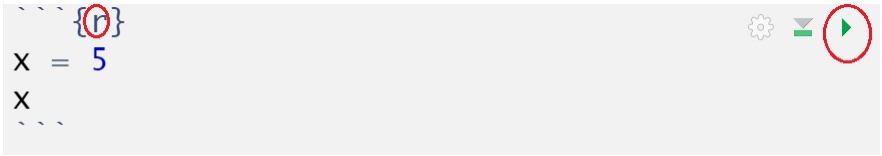
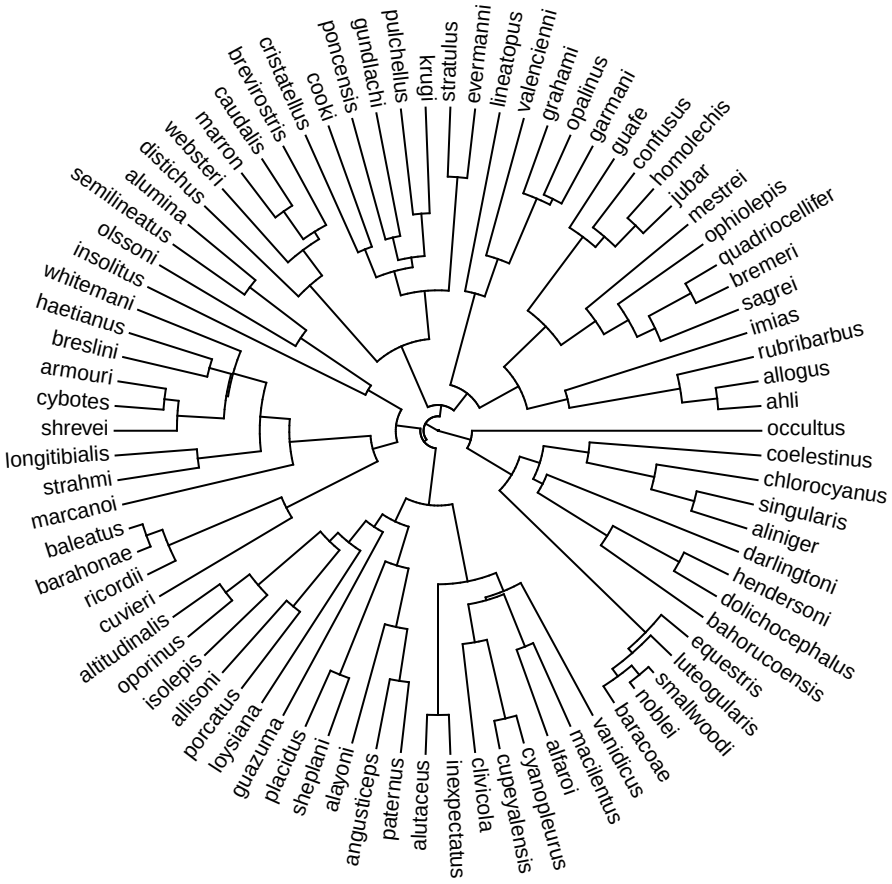


Figure 4: Executable code chunk.

```r
x = 5
x
```

Figure 5: Executable code chunk with control options.



This includes reading and writing to file, although we must be sure that we are currently in the same directory as the file, or that we specify the full path to the file.

Code chunks that we do not want to evaluate we can write as follows:

```r
library(devtools)
install_github("liamrevell/phytools") ## install phytools from GitHub
```