BIOSTATS

Akira Terui

Last updated on 2022-09-14

Contents

Chapter 1

Introduction

This textbook aims to introduce fundamental statistical techniques and their applications to biological data. A unique aspect of this book is the "flipped-order" introduction. Many statistics courses start with theory; yet, I found it difficult for those unfamiliar with statistics. I will start with a real example of the method, followed by an explanation for an underlying theory/concept. The author is an ecologist, so some methods in this book might not be popular in other fields.

Chapter 2

Project Management

2.1 R Project

For this entire book, I will use `R Project` as a fundamental unit of work-space, in which all the relevant materials (e.g., R scripts `.R` and data files) are assembled together. There are many ways to organize your project, but I usually make a single `R Project` for a collection of scripts and data that will lead to a single publication (see example here). To setup an `R Project` you will need *R Studio* in addition to base *R*. While *R* is a stand-alone software, I strongly recommend to use it with *R Studio*. *R Studio* has many functions that help your data analysis. *R* and *R Studio* can be installed from the following websites:

- R (you can choose any CRAN mirror to download)
- R Studio

Once you open *R Studio*, you will see the following interface (Figure 2.1). There are three major panels in its first appearance – `Console`, `Environment`, and `Files`. `Console` is the place where you write your codes and execute calculation/data manipulation/analysis. `Environment` lists items you saved as an object. `Files` list any files in a designated location in your computer.

Although you can work on your data as it appears, **it is actually a BAD idea**. As you work on your project, numerous materials will be generated. How do you manage files? If you randomly locate those materials within your computer, you will lose necessary items sooner or later. For this reason, I assemble all the relevant materials in a single `R Project`. You can create a new `R Project` with the following procedure.

a. Go to `File > New Project` on the top menu
b. Select `New Directory`
c. Select `New Project`

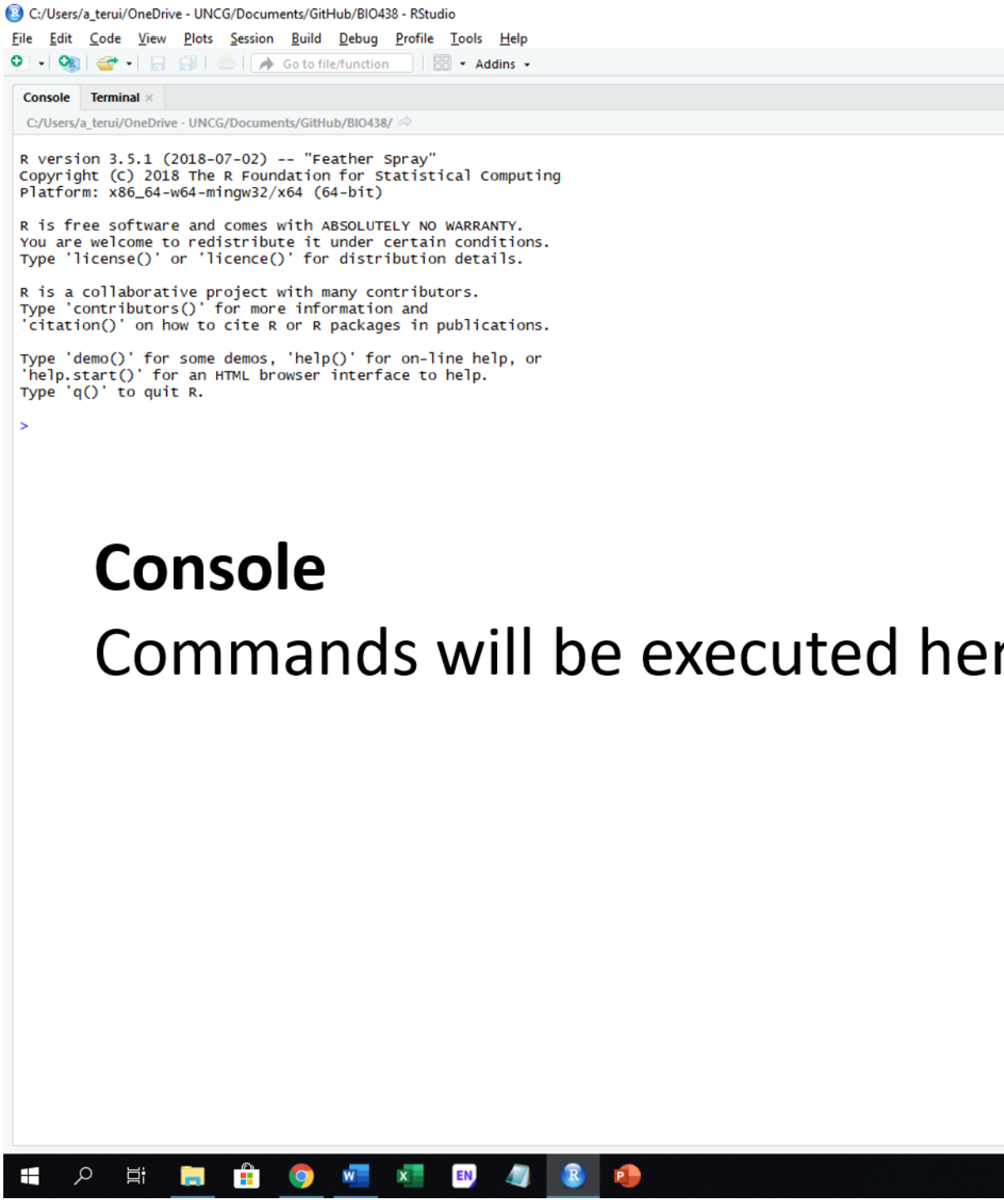

Figure 2.1: R Studio interface.

A new window pops up and prompts you to name a directory with a location in your computer. Click `Browse` to select a location for the directory.

IMPORTANT: When you locate your project directories in your computer, I would strongly recommend to create a designated space. For example, in my computer, I have a folder named `/r_project` in which all the `R Project` directories are located.

2.2 Internal Structure

The internal structure of an `R Project` is extremely important to navigate yourself (others once it's published). `R Project` will be composed of multiple types of files, typically `.R`, `.csv`, `.rds`, `.Rmd` among others. Unless those files are arranged in an organized manner, it is VERY LIKELY to make severe errors in coding. So I take this seriously. Table 2.1 is my suggested subdirectory structure.

Table 2.1: Suggested internal structure of `R Project`

Name	Content
`README.md`	Markdown file explaining contents in the `R Project`. Can be derived from `README.Rmd`.
`/code`	Sub-directory for R scripts (`.R`).
`/data_raw`	Sub-directory for raw data before data manipulation (`.csv` or other formats). Files in this sub-directory MUST NOT be modified unless there are changes to raw data entries.
`/data_formatted`	Sub-directory for formatted data (`.csv`, `.rds`, or other formats).
`/output`	Sub-directory for result outputs (`.csv`, `.rds`, or other formats). This may include statistical estimates from linear regression models etc.
`/rmd`	(Optional) Sub-directory for Rmarkdown files (`.Rmd`). Rmarkdown allows seamless integration of R scripts and text.

It is also critical to have **consistent naming rules** for your files. As you make progress on your project, the number of files in each sub-directory will increase, perhaps exponentially. You will find it difficult navigating yourself unless you have clear naming rules for files (and even worse for others). Here are some recommendations:

- **NO SPACE.** Instead, use underscore.
  - Do: `script_week1.R`
  - Don't: `script week1.R`
- **NO UPPERCASE.** Use lowercase for file names.
  - Do: `script_week1.R`
  - Don't: `Script_week1.R`
- **BE CONSISTENT.** Apply consistent naming rules within a project.

- Do: R scripts for figures always start with a common prefix, e.g., `figure_XXX.R` `figure_YYY.R`(`XXX` and `YYY` specifies further details).
- Don't: R scripts for figures start with random text, e.g., `XXX_fig.R` , `Figure_Y2.R` , `plotB.R`.