Tips on Using RStudio and R Markdown

Thomas E. Love, Ph.D.

January 6, 2018

Contents

This document may be found at
`https://github.com/THOMASELOVE/500-2018/tree/master/data-and-code`

1 Use R Studio Projects Whenever You Can

- As the documentation suggests, RStudio Projects "make it straightforward to divide your work into multiple contexts, each with their own working directory, workspace, history, and source documents."
- Open R Studio, and either start a new Project (using the File ... New Project menu) or open an existing Project by clicking on the Open Project button at the top right of R Studio.

2 What is R Markdown?

R Markdown is the most useful tool I have learned in the past few years. It is ... "an authoring format that enables easy creation of dynamic documents, presentations, and reports from R. It combines the core syntax of markdown (an easy-to-write plain text format) with embedded R code chunks that are run so their output can be included in the final document. R Markdown documents are fully reproducible (they can be automatically regenerated whenever underlying R code or data changes)."

Moving from simply writing scripts in R to Markdown files is a short step, and well worth the effort. Within R Studio, you can write R Markdown syntax, run it to see your results, and then produce a final document which looks great, and is completely reproducible. This is a tool you will get a lot of use from in this course, and, I expect, in your future work.

2.1 Learning More about Writing Markdown Files

1. I encourage you to visit the help page linked to directly within R Studio, by clicking on the question mark box in R Studio that appears when you open a Markdown file.

2. To get started, try simply writing your report in plain text. Markdown syntax is used to describe how to format text in the final report, and to embed R code.

3. To learn more about writing Markdown files, you can look at the examples I'll provide in class (all of my presentations are built using Markdown).

4. R Studio provides cheat sheets that others have found to be very helpful at `https://www.rstudio.com/resources/cheatsheets/`. They have stuff for R Markdown (including a detailed reference guide), R Studio, Data Wrangling with dplyr and tidyr, Data Visualization, etc.

2.2 Creating an HTML, Word or PDF file

1. An R Markdown file is essentially a text document, with interspersed R code that lets you produce reports that combine narration with results, and that can be easily exported as an HTML, PDF or Word file.

2. Open a new R Markdown file (File ... New File ... R Markdown), or an existing one (File ... Open File), and indicate the specific type of output you want to make.

3. R Markdown files use the `.Rmd` extension.

4. Assuming you have Microsoft Office installed on your computer, you should be immediately able to write a Markdown file and render it in either HTML or as a Word document, by creating a Markdown document and then clicking on the Knit HTML or Knit Word button at the top of R Studio.

5. In order to get Markdown to generate a PDF file directly (rather than Word or HTML) you'll want to download an installation of the TeX (TeX is pronounced "tech") software, which builds those relatively pretty documents. I use MacTeX on my Mac, and MikTeX on my PC. This also lets me include LaTeX and TeX commands directly within a Markdown file, which is something you might eventually want to do, and that I did to build this document. To get the (free) software, visit:

    - `http://miktex.org/download` to get MiKTeX for the PC. There is a Tutorial on this page if you need step-by-step guidance.
    - `https://tug.org/mactex/mactex-download.html` to get MacTeX for a Mac.
    - On either the Mac or the PC, I would also consider downloading and installing TeXworks, which provides a nice front end if you're actually going to write things in LaTeX or TeX. Visit `https://www.tug.org/texworks/` and click Get It.

2.3 Some Specific Tips

1. When using R Markdown, you need to be sure that Markdown is looking for your files in the proper directory. The easiest way for me to do this is to build a separate directory for each new analysis, and open up a new Project in that directory before developing your Markdown code.

- The directory being used is almost inevitably the directory in which the Markdown file is stored by RStudio.
- A smart option is to run the command getwd() within your Markdown file, which should specify for you which working directory Markdown is looking in. To change this directory, you can then use the setwd() command within the Markdown file to specify the directory you want Markdown to use.

2. When writing code in Markdown, you need to name each chunk, and give each chunk a different name. So, for instance, you'd use code like this:

```
‘‘‘{r chunkname, commandstomarkdownifany}
R commands go here
‘‘‘
```

   - If you name something `chunkname`, then your next chunk of code needs to be named something different than `chunkname`, like `chunk2`, or whatever.
   - Good coding practice suggests the use of a name that describes what the chunk of code does. Any name is acceptable including spaces, etc., so long as it precedes the }.

3. Putting a comma after a chunk name lets you, in addition to naming the chunk, specify commands (after the comma) to R Markdown about what you want it to do with the chunk. Here are a few useful commands, some of which also appear at `http://rmarkdown.rstudio.com/authoring_rcodechunks.html`:

   - `‘‘‘{r chunk01, echo=FALSE}` tells Markdown to execute your code, but simply write the result, rather than the code, into the results file.
   - `‘‘‘{r chunk01, eval=FALSE}` tells Markdown to write the code into the results, but not execute the code.
   - `‘‘‘{r chunk01, message=FALSE}` tells Markdown to not print any of the messages your code generates.
   - `‘‘‘{r chunk01, warning=FALSE}` tells Markdown to not print any of the warnings your code generates.
   - `‘‘‘{r chunk01, fig.height=4, fig.width=6}` tells Markdown to keep any figures produced by this chunk to a maximum of 4 inches high, and 6 inches wide. The default values depend on the type of output you are generating. You can use fig.height or fig.width alone if you want to keep the other value at its default.
   - `‘‘‘{r chunk01, fig.align=‘center‘}` tells Markdown to align your figure in the center (other options are left or right.)
   - `‘‘‘{r chunk01, tidy=TRUE}` tells Markdown to tidy up your code for presentation, so it will still print, but it will (perhaps) be a bit more organized.

4. Here's the default header information I use when writing materials for a PDF document with a table of contents...

```
---
title: "Basic R Materials for 500"
author: "Thomas E. Love"
date: "January 7, 2016"
output:
  pdf_document:
    number_sections: yes
    toc: yes
    toc_depth: 3
fontsize: 12pt
geometry: margin=1in
---
```

5. To set up R Markdown so it doesn't add any comment symbols before the results, I use the following code chunk:

```
‘‘‘{r set-options, echo=FALSE, cache=FALSE, message=FALSE}
knitr::opts_chunk$set(comment=NA)
‘‘‘
```