R Markdown Guide

Isaac Quintanilla Salinas

2020-12-20

Introduction

This document contains an introduction on R Markdown files and creating PDF files. For more information, go to https://bookdown.org/yihui/rmarkdown/ for a complete guide on R Markdown. Other useful materials can be found on RStudio's website: https://rmarkdown.rstudio.com/lesson-1.html.

Before reading this document, download the Rmd, bib, and PDF files. Reading the Rmd file and PDF file simultaneously provides a better context on the syntax and formatting for an Rmd file. The syntax in an Rmd file may be difficult to understand, so looking at the Rmd file simultaneously may clarify the explanations. Lastly, to `knit` the R Markdown file, make sure LaTex, `formatR`, and `xtable` are installed. Additionally, have the `Example.bib` file located in the same folder as the Rmd File. More information on each item is provided below.

What is R Markdown?

R Markdown is a file type used to create technical reports while including both R code and output in a document. An Rmd file is just a really fancy R Script containing extra capabilities. R Markdown also allows for citations, footnotes, mathematical expressions, links, and many more. Once the document is finished, it can be rendered to a word file, pdf, or html file.

Anatomy of R Markdown

There are three main components in an R Markdown file: the YAML header, R code, and basic text.

The YAML header contains information on how to render the document. It is usually located at the beginning of the document. The YAML header is usually surrounded by 3 dashes (`---`) above and below it. For starters, the YAML header will contain a 'title', 'author', 'date', and 'output' line. Visit the R Markdown website for more information.

The R code is located in a block known as chunks. A chunk tells RStudio to read the next lines as code. A chunk begins with three back ticks followed by `{r}` and ends with three back

ticks. Everything in between is R code. A chunk with R code will render like below:

```
mean(mtcars$mpg)
```

```
## [1] 20
```

Notice the chunk includes the code in a block followed by the output from the console.

The last component of an R Markdown document is the text. Just write anywhere in the document, and it will be rendered as is.

Chunk Options

The code chunks in R have options that will alter how the code or the output is rendered. The chunk options can be set either globally to affect the entire R Markdown document or locally to affect only an individual chunk. For more information about chunk options visit https://yihui.name/knitr/options/

Global Chunk Options

To set up chunk options globally, create a chunk at the beginning and use the `knitr::opts_chunk$set()` function to set chunk options.

```
knitr::opts_chunk$set(cache = T)
```

A couple of recommended chunk options set globally are `cache=T` and `tidy=T`. These options make rendering the document easier. The `cache=T` option tells RStudio to run the chunk and save the output in an `RData` file when it is rendered. When re-rendering a document, RStudio will only run chunks that are new or chunks that were altered. All other chunks' output will be obtained from the `RData` file. This speeds up the rendering process by not running code that was not altered.

The other chunk option is `tidy=T`. More specifically: `tidy.opts=list(width.cutoff=60)` and `tidy=TRUE`. This tells RStudio to prevent code from printing off the page. For example, look at the output of this chunk:

```
## This comment is designed to show what happens when all your code is in 1 line. This
```

Notice the comment being printed off the page. Using the options `tidy.opts=list(width.cutoff=60)` and `tidy=TRUE`, the chunk is rendered as

```
## This comment is designed to show what happens when all your
## code is in 1 line. This is fine when you are coding, but
## when you are putting it in a report, it will run off the
## page.
```

Notice the comment being printed on 4 lines of code instead of 1. The `width.cutoff=60` option may need to be adjusted to get all the code one the page. To use this tidy option, install the `formatR` package once:

```
install.packages("formatR")
```

To set these two options as global options, place this in a setup chunk at the beginning of the document.

```
knitr::opts_chunk$set(cache = T)
knitr::opts_chunk$set(tidy.opts = list(width.cutoff = 60), tidy = TRUE)
```

Note: Sometimes, the cutoff may stop working and your code is running off the page. To fix this, change the value for the width.cutoff. Then change it back.

One last option to include is `options(digits=2)` This is used for inline code. This is discussed later in the document.

Local Chunk Options

There are local chunk options that may be beneficial to put in each individual chunk. Each option is placed in `{r}` separated by commas.

One option is the `eval` option. When set to `FALSE`, RStudio will only display the code but not run the line. For example, this chunk contains `{r,eval=FALSE}`:

```
mean(mtcars$mpg)
```

Nothing is printed out. When the chunk contains `{r,eval=TRUE}`:

```
mean(mtcars$mpg)
```

```
## [1] 20
```

Another option is the `echo` option. When set to `FALSE`, the code will disappear from the document. This next chunk contains `{r,echo=TRUE}`:

```
mean(mtcars$mpg)
```

```
## [1] 20
```

Everything looks the same. Now the chunk contains `{r,echo=FALSE}`

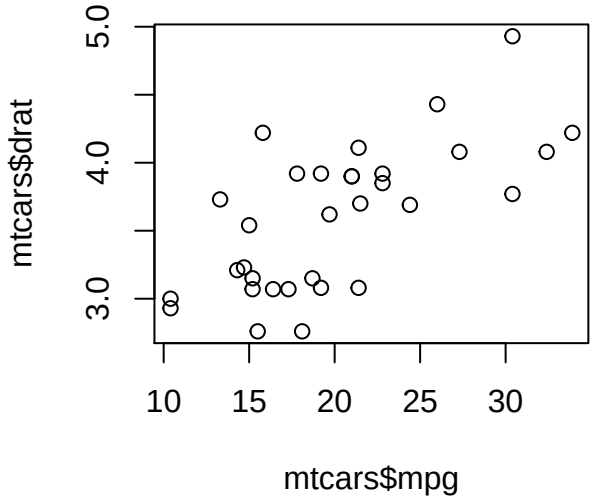
```
## [1] 20
```

The R Code disappears.

There are chunk options for figures as well. A few options are `fig.height`, `fig.width`, `fig.align`, and `fig.cap`.

This chunk contains `{r,fig.height=3.5,fig.width=3.5,fig.align='left'}`.

```
plot(mtcars$mpg, mtcars$drat)
```



The chunk options tells RStudio to create an image that is 3.5 inches in height and width, and align the image to the left.

The following chunk contains `{r, fig.height=3.5, fig.width=3.5, fig.align='center', fig.cap="\\label{fig1}This is a scatter plot of MTCARS' MPG and DRAT"}`.

```r
plot(mtcars$mpg, mtcars$drat)
```

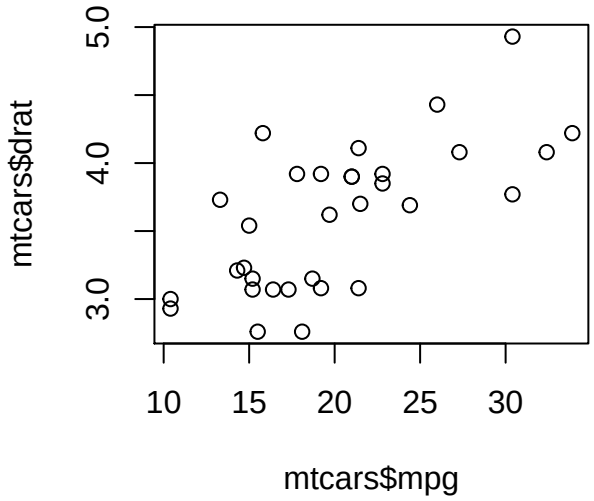


Figure 1: This is a scatter plot of MTCARS' MPG and DRAT

This figure may be placed on a different page. The chunk contains an additional option: `fig.cap="\\label{fig1}This is a scatter plot of MTCARS' MPG and DRAT"`. It adds a caption to the figure. At the beginning of the caption, the option contains `\\label{fig1}`. This labels the plot to be referenced later in the document. Figure 1 can be referenced with `\ref{fig1}`.

Formatting

R Markdown contain basic formatting capabilities. The use of the `#` followed by text creates a heading. Using two or more `#` symbols will create subheadings based on the number of `#`. A text is *italicized* by surrounding the text with one asterisk (`*italicized*`). A text is **boldfaced** by surrounding it with 2 asterisk (`**boldfaced**`).

To create an unordered list, use the `+` symbol at the beginning of each line. To create a sub-item, press the tab button twice (4 spaces), then the `+` symbol. Repeat this method for further sub-items.

- First Item
- Second Item
  - First Sub-Item
    - First Sub-Sub-Item
      - First Sub-Sub-Sub-Item

To created an ordered list, type the number followed by a period for each line. To create sub-lists, press the tab button twice and order them appropriately.

1. First
2. Second
    a. First
    b. Second
        1) First
        2) Second

    A block quote is created with the `>` symbol at the beginning of a line.

R Markdown allows a table to be constructed in 2 ways, manually or with the `xtable` package. A table is manually created by using `|`, `:`, and `-`. The first line contains `|` and the column names in between. The second line contains `|:-|:-|` which indicates how the table is aligned. The location of `:` symbol just tells RStudio about the alignment. Go to the math section for more elaborate examples.

The `xtable::xtable` function creates a table from a dataframe or R object. Here is an example a table using the `mtcars` dataset.

```
print(xtable::xtable(head(mtcars), caption = "The MTCARS Dataset",
    label = "mtcarsdata", digits = 1, align = c("l", rep("c",
        11))), comment = F)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6.0	160.0	110.0	3.9	2.6	16.5	0.0	1.0	4.0	4.0
Mazda RX4 Wag	21.0	6.0	160.0	110.0	3.9	2.9	17.0	0.0	1.0	4.0	4.0
Datsun 710	22.8	4.0	108.0	93.0	3.9	2.3	18.6	1.0	1.0	4.0	1.0
Hornet 4 Drive	21.4	6.0	258.0	110.0	3.1	3.2	19.4	1.0	0.0	3.0	1.0
Hornet Sportabout	18.7	8.0	360.0	175.0	3.1	3.4	17.0	0.0	0.0	3.0	2.0
Valiant	18.1	6.0	225.0	105.0	2.8	3.5	20.2	1.0	0.0	3.0	1.0

Table 1: The MTCARS Dataset

Notice that Table 1 is produced easily. Table 1 is referenced by using the label created in the function and the `\ref{mtcarsdata}`. The `xtable::xtable()` function can be used to create tables from commonly used R functions such as `lm()`. The `xtable` requires the `print()` function and `comment=F` argument to prevent a note from printing. Additionally, the chunk options `results='asis'` is needed. To install `xtable` run the following line:

```
install.packages("xtable")
```

References

R Markdown contains capabilities to add citations and a bibliography. For example, to cite your textbook (Mendenhall and Sincich 2012), use the `@` symbol followed by a citation identifier from the `.bib` file surrounded by square brackets, `[@mendenhallSecondCourseStatistics2012]`. To cite your textbook again (2012) without the authors names, use a `-` sign in front of the `@` symbol, `[-@mendenhallSecondCourseStatistics2012]`. To cite multiple books (Casella and Berger 1990; Rohatgi and Saleh 2015; Resnick 2014; Lehmann and Casella 1998; Lehmann and Romano 2005), add each citation inside the square brackets with the `@` symbol and separate them with semicolons, `[@casellaStatisticalInference1990; @rohatgiIntroductionProbabilityStatistics2015; @resnickProbabilityPath2014; @lehmannTheoryPointEstimation1998; @lehmannTestingStatisticalHypotheses2005]`.

A reference page can be added at the end of your document by adding the following line: `# References`. R Markdown will automatically add references at the end.

To use citations and references, R Markdown needs to know the references. This is done by providing a `.bib` file containing all the information needed to construct the citations and references. First, save the `.bib` file in the same folder (directory) as your `Rmd` file. Then add the line `bibliography: Example.bib` to the YAML header. Make any changes appropriately to the line, such as the name of the `.bib` file.

Bib File

The `.bib` file is just a normal text file that contains the extension `.bib`. All you will do is add your references to the file. Open the `Example.bib` file and notice that it contains a bunch of lines indicating certain things about a reference. Creating a `.bib` file is tedious. However, there are reference managers that can help. I recommend Zotero. It is an open-source reference manager designed to help with many things. With Zotero, you can import citations easily with their browser extension. Once a citation is in Zotero, you can export your library to a `.bib` file. That's it! A couple tips is to check your citations in Zotero and fix them as needed. Importing citations from UCR's library seem to provide accurate citations. Other online resources may provide weird results.

Math

R Markdown is capable of writing mathematical formulas using LaTeX code. A mathematical symbol can be written inline using single `$` signs. For example, `$\alpha$` is viewed as α in a document. To write mathematical formulas on its own line use either `$$` or `\[\]`. For example, `$$Y=mX+b$$` is viewed as

$$Y = mX + b$$

or `\[Y=mX+b\]` is viewed as

$$Y = mX + b.$$

The next page contain tables of LaTeX syntax for mathematical symbols.

Mathematical Notation

Notation	code
$x = y$	`$x=y$`
$x > y$	`$x>y$`
$x < y$	`$x<y$`
$x \geq y$	`$x\geq y$`
$x \leq y$	`$x\leq y$`
x^y	`$x^{y}$`
x_y	`$x_{y}$`
$\bar{x}$	`$\bar x$`
$\hat{x}$	`$\hat x$`
$\tilde{x}$	`$\tilde x$`
$\frac{x}{y}$	`$\frac{x}{y}$`
$\frac{\partial x}{\partial y}$	`$\frac{\partial x}{\partial y}$`
$x \in A$	`$x\in A$`
$x \subset A$	`$x\subset A$`
$x \subseteq A$	`$x\subseteq A$`
$x \cup A$	`$x\cup A$`
$x \cap A$	`$x\cap A$`
$\{1, 2, 3\}$	`$\{1,2,3\}$`
$\int_a^b f(x)dx$	`$\int_a^bf(x)dx$`
$\left\{\int_a^b f(x)dx\right\}$	`$\left\{\int_a^bf(x)dx\right\}$`
$\sum_{i=1}^n x_i$	`$\sum^n_{i=1}x_i$`
$\prod_{i=1}^n x_i$	`$\prod^n_{i=1}x_i$`
$\lim_{x\to 0} f(x)$	`$\lim_{x\to0}f(x)$`
$X \sim \Gamma(\alpha, \beta)$	`$X\sim \Gamma(\alpha,\beta)$`

Greek Letters

Letter	Lowercase	Code	Uppercase	Code
alpha	α	`\alpha`	–	–
beta	β	`\beta`	–	–
gamma	γ	`\gamma`	Γ	`\Gamma`
delta	δ	`\delta`	Δ	`\Delta`
epsilon	ϵ	`\epsilon`	–	–
zeta	ζ	`\zeta`	–	–
eta	η	`\eta`	–	–
theta	θ	`\theta`	Θ	`\Theta`
iota	ι	`\iota`	–	–
kappa	κ	`\kappa`	–	–
lambda	λ	`\lambda`	Λ	`\Lambda`
mu	μ	`\mu`	–	–
nu	ν	`\nu`	–	–
xi	ξ	`\xi`	Ξ	`\Xi`
pi	π	`\pi`	Π	`\pi`
rho	ρ	`\rho`	–	–
sigma	σ	`\sigma`	Σ	`\Sigma`
tau	τ	`\tau`	–	–
upsilon	υ	`\upsilon`	Υ	`\Upsilon`
phi	ϕ	`\phi`	Φ	`\Phi`
chi	χ	`\chi`	–	–
psi	ψ	`\psi`	Ψ	`\Psi`
omega	ω	`\omega`	Ω	`\Omega`
varepsilon	ε	`\varepsilon`	–	–

Rendering a Document

An R Markdown can be rendered into either an html file, pdf document or word document. Rendering the R Markdown to an html file or word document can be easily done using the knit button above. However, rendering the R Markdown file to a pdf document requires LaTeX to be installed. There are two methods to install LaTeX: from the LaTeX website or from R. I recommend installing the full LaTeX distribution from the https://www.latex-project.org/get/. This provides you with everything you may need. You can also install it from R:

```
install.packages("tinytex")
tinytex::install_tinytex()
```

You will only need to run these lines of code once and then you can render pdf documents easily.

Tips

- Render your document often so it easier to identify problems with rendering

References

Casella, George, and Roger L. Berger. 1990. *Statistical Inference*. The Wadsworth & Brooks/Cole Statistics/Probability Series. Pacific Grove, Calif: Brooks/Cole PubCo.

Lehmann, E. L., and Joseph P. Romano. 2005. *Testing Statistical Hypotheses*. 3rd ed. Springer Texts in Statistics. New York: Springer.

Lehmann, Erich L., and George Casella. 1998. *Theory of Point Estimation*. Second Edition. Springer Texts in Statistics. New York, NY: Springer New York.

Mendenhall, William, and Terry Sincich. 2012. *A Second Course in Statistics: Regression Analysis*. Seventh edition. Boston: Prentice Hall.

Resnick, Sidney I. 2014. *A Probability Path*. Modern Birkhäuser Classics. New York: Birkhäuser.

Rohatgi, V. K., and A. K. Md Ehsanes Saleh. 2015. *An Introduction to Probability and Statistics*. Third edition. Wiley Series in Probability and Statistics. Hoboken, New Jersey: Wiley.