Introduction to RMarkdown

Abhinav Anand, IIMB

2019/06/18

Setup

The packages `rmarkdown` and `knitr` need to be installed prior to running the commands below. To install, type in the console `install.packages(c("knitr", "rmarkdown"))`.

Background

A text processor such as MS Word uses a "What-You-See-Is-What-You-Get" (WYSI-WYG) editor. We see the output of the "rendered" text directly. However, to exercise finer control over the rendering, a "markup" language is better. Latex and HTML are examples of markup languages: they contain ordinary text, as well as special commands that govern the final rendering of the text. Markdown is an especially convenient markup language that preserves the finer aspects of text formatting without being too hard to read.

There are many programs for rendering documents written in Markdown into documents in the .html, .pdf and .docx formats (among many others). R Markdown extends Markdown to incorporate text formatting, mathematics, figures, tables; as well as R code and its output directly into the rendered document.

Behind the scenes, when we "knit" the file, R Markdown sends the `.rmd` file to `knitr()` which executes all code chunks and creates a new markdown file (extension `.md`) which includes both the code and its output. This file is then used by `pandoc` (essentially a free and open source document converter) to convert to the desired format. This two-step workflow can help to create a very wide range of formatting options for eventual publishing.

Literate Programming and Reproducibility

The objective for which RMarkdown proves most useful is its potential for literate programming—a programming concept due to the famous computer scientist Donald Knuth (also the father of TeX)—in which the code and its explanation in a natural language occur in the same document.

RMarkdown provides us with a way of including both code snippets and marked up text to *build* a document that contains codes, its output as well as documentation. This also helps make research both *replicable* and *reproducible* since the `.rmd` file with the code can be sourced by readers themselves. The study can be replicated with new data using the codes in the `.rmd` file while the study can be reproduced by simply re-sourcing the document. Reproducibility is also essential for our future selves who may need to revisit old research projects.

The Main Components of RMarkdown

Each RMarkdown document has three main components—header, text; and code chunks.

The (YAML) Header

The current file's header includes the following lines:

```
---
title: "Introduction to RMarkdown"
author: Abhinav Anand

output:
  pdf_document: # other options: .html, .docx etc.
    keep_tex: true # keeps the intermediate .tex document

fontsize: 11pt # latex parameter
documentclass: article # latex class
geometry: margin = 1.5in # latex package
```

```
linkcolor: blue
urlcolor: red
citecolor: magenta

citation_package: natbib # latex style referencing
bibliography: Working_Paper.bib

header-includes:
  - \linespread{1.25}


---
```

Text

To insert a break between paragraphs, include a single completely blank line.

For other text-formatting, see below:

1. `*italic*`: *italic*
2. `_italic_`: *italic*
3. `**bold**`: **bold**
4. `__bold__`: **bold**
5. `superscript^2^`: superscript^2
6. `subscript~2~`: subscript$_2$
7. "Section" heading: `#`
8. "Subsection" heading: `##`
9. "Subsubsection" heading: `###`
10. Math: Just use the LaTeX style
11. Link: `<https://cran.r-project.org/.com>`: https://cran.r-project.org/
12. Word link: `[CRAN project](https://cran.r-project.org/)`: CRAN project
13. Simple Tables:

```
First Column  | Second Column
------------- | -------------
$c_1$  | $c_2$
$c_3$  | $c_4$
```

First Column	Second Column
c_1	c_2
c_3	c_4

13. Lists:

```
*   Bulleted list item 1

*   Item 2

    * Item 2a

    * Item 2b

1.  Numbered list item 1
2. Item 2
3. Item 3
```

- Bulleted list item 1
- Item 2
  - Item 2a
  - Item 2b

1. Numbered list item 1
2. Item 2
3. Item 3

Table

If additional table formatting is needed, use the `knitr::kable` function.

```r
knitr::kable(gapminder::gapminder[1:10, ],
             caption = "The first 10 rows of gapminder")
```

Table 2: The first 10 rows of gapminder

country	continent	year	lifeExp	pop	gdpPercap
Afghanistan	Asia	1952	28.801	8425333	779.4453
Afghanistan	Asia	1957	30.332	9240934	820.8530
Afghanistan	Asia	1962	31.997	10267083	853.1007
Afghanistan	Asia	1967	34.020	11537966	836.1971
Afghanistan	Asia	1972	36.088	13079460	739.9811
Afghanistan	Asia	1977	38.438	14880372	786.1134
Afghanistan	Asia	1982	39.854	12881816	978.0114
Afghanistan	Asia	1987	40.822	13867957	852.3959
Afghanistan	Asia	1992	41.674	16317921	649.3414
Afghanistan	Asia	1997	41.763	22227415	635.3414

Bibliography and Citations

Declare in YAML header as shown above. To cite, pre-fix `@` with the citation key in the bibliography file.

For example:

1. This is the simple style of citation: `@FH:2009` yielding Foster and Hart (2009)
2. For multiple citations, a semi-colon is used: `@Rachev:2008; @Fama:1963`—Rachev, Stoyanov, and Fabozzi (2008); Fama (1963)
3. For citing within parentheses, use: square brackets `[@Bollerslev:1986]`, (Bollerslev 1986)

It is common practice to end the rmarkdown file with a section header for the bibliography, such as `# References` or `# Bibliography` since unlike Latex, there is no automatic section heading for the bibliography.

Code Chunks

All inline codes must be within backticks along with the symbol `r`.

For more involved computation, we need to use code chunks, enclosed within three backticks, followed by `r` and then followed by three closing backticks: ` ```{r } ` followed by ` ``` `. It is considered good practice to include a short name following the symbol `r`.

Chunk options

The following are the main options:

1. `eval = FALSE`: This will ensure no evaluation of the code chunk.
2. `include = FALSE`: Run the code but show neither the code nor its results.
3. `echo = FALSE`: Run the code, show the results but don't show the code.
4. `message = FALSE` or `warning = FALSE`: Don't show messages or warnings.

If these choices are global—the same for all code chunks in the document—then we can include the options at the beginning of the document via:

```
knitr::opts_chunk$set(echo = T,
                      warning = T,
                      message = F,
                      eval = T,
                      include = T
                      )
```

If we need to override global options in a code chunk, we may simply set that parameter manually after declaring the code chunk name: say, `eval = F` for one particular chunk.

References

Bollerslev, T. 1986. “Generalized Autoregressive Conditional Heteroskedasticity.” *Journal of Econometrics* 31: 307–27.

Fama, Eugene F. 1963. “Mandelbrot and the Stable Paretian Hypothesis.” *Journal of Business* 36: 420–29.

Foster, Dean P., and Sergiu Hart. 2009. “An Operational Measure of Riskiness.” *Journal of Political Economy* 117 (5): 785–814.

Rachev, Svetlozar T., Stoyan V. Stoyanov, and Frank J. Fabozzi. 2008. *Advanced Stochastic Models, Risk Assessment, and Portfolio Optimization.* Hoboken, New Jersey: John Wiley; Sons.