Markdown + RMarkdown Reference

Stephen D. Turner, Ph.D.

About

Why?

This reference was created in support of a workshop on reproducible research using dynamic documents in R.

Hey, stuff's missing!

This isn't meant to be a comprehensive guide to either Markdown or RMarkdown. You can find plenty of those elsewhere. This is here to provide a bare-bones, zero-entry introduction to writing documents with Markdown and dynamic documents with RMarkdown for someone who's never written anything outside of a word processor. That is, here you'll find only the stuff you need to get started and not all the extra frills to distract you.

Download the PDF

Click here to download a PDF of this guide.

Source

The source is on GitHub.

Markdown Reference

Emphasis

```
Italics with _underscores_.

Bold with **double asterisks**.

Combined emphasis with **asterisks and _underscores_**.
```

Italics with *underscores*.

Bold with **double asterisks**.

Combined emphasis with **asterisks and *underscores***.

Section headings

```
# Header 1

A single # results in a top-level major heading.

## Header 2

Two ##s results in a sub-heading.

### Header 3

Three ###s results in a sub-sub-heading.

#### Header 4

And so on...
```

Header 1

A single # results in a top-level major heading.

Header 2

Two ##s results in a sub-heading.

Header 3

Three ###s results in a sub-sub-heading.

Header 4

And so on...

Lists

Unordered lists (i.e., bullets)

```
- Unordered list item
- Unordered list item
    - Unordered list sub-item
    - Unordered list sub-item
- Unordered list item
```

- Unordered list item
- Unordered list item
    - Unordered list sub-item
    - Unordered list sub-item
- Unordered list item

Ordered lists (i.e., numbers)

```
1. Ordered list item
2. Ordered list item
    1. Ordered list sub-item
    2. Ordered list sub-item
3. Ordered list item
```

1. Ordered list item
2. Ordered list item
    1. Ordered list sub-item
    2. Ordered list sub-item
3. Ordered list item

Mixed lists

```
1. First ordered list item in a mixed list
2. Another item
    - Unordered sub-list.
1. Actual numbers don't matter, just that it's a number
    1. Ordered sub-list
    1. Ordered sub-list
4. And another item.
```

1. First ordered list item in a mixed list
2. Another item
    - Unordered sub-list.
3. Actual numbers don't matter, just that it's a number
    1. Ordered sub-list
    2. Ordered sub-list
4. And another item.

Links

There are two ways to create links.

```
I'm an inline link to [Google's homepage](https://www.google.com).
```

I'm an inline link to Google's homepage.

```
Bare URLs in angle brackets get turned into links <https://www.google.com>.
```

Bare URLs in angle brackets get turned into links https://www.google.com.

Images

```
![Caption text (if supported)](assets/rmarkdown-workflow.png)
```

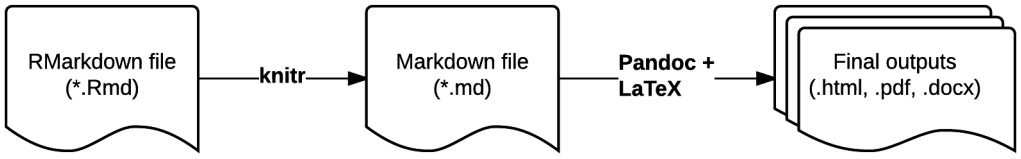

Figure 1: Caption text (if supported)

Code and Syntax Highlighting

Inline code

```
Inline `code` has `back-ticks around` it.
```

Inline `code` has `back-ticks around` it.

Code blocks

````
```
Blocks of code are "fenced" by lines containing three backticks.
If you don't specify, there will be no syntax highlighting applied.
```
````

```
Blocks of code are "fenced" by lines containing three backticks.
If you don't specify, there will be no syntax highlighting applied.
```

Code blocks with syntax highlighting

E.g., R code

````
```r
# Just specify the language after the opening fence.
library(dplyr)
gm <- read.csv("gapminder.csv")
matrix(rnorm(25), nrow=5)[1:2,3:4]
```
````

```r
# Just specify the language after the opening fence.
library(dplyr)
gm <- read.csv("gapminder.csv")
matrix(rnorm(25), nrow=5)[1:2,3:4]
```

E.g., Python code

```
```python
# This is a python comment
s = "Python syntax highlighting"
print s
```
```

```python
# This is a python comment
s = "Python syntax highlighting"
print s
```

Tables

Tables aren't part of the Markdown core, but they are supported by some renderers.

```
Colons can be used to align columns.

| Tables        | Are           | Cool  |
| ------------- |:-------------:| -----:|
| col 3 is      | right-aligned | $1600 |
| col 2 is      | centered      |   $12 |
| zebra stripes | are neat      |    $1 |

There must be at least 3 dashes separating each header cell.
The outer pipes (|) are optional, and you don't need to make the
raw Markdown line up prettily. You can also use inline Markdown.

Markdown | Less | Pretty
--- | --- | ---
*Still* | `renders` | **nicely**
1 | 2 | 3
```

Colons can be used to align columns.

Tables	Are	Cool
col 3 is	right-aligned	$1600
col 2 is	centered	$12
zebra stripes	are neat	$1

There must be at least 3 dashes separating each header cell. The outer pipes (|) are optional, and you don't need to make the raw Markdown line up prettily. You can also use inline Markdown.

Markdown	Less	Pretty
Still	`renders`	**nicely**
1	2	3

Blockquotes

```
> It was the best of times, it was the worst of times, it was the age of
wisdom, it was the age of foolishness, it was the epoch of belief, it was the
```

```
epoch of incredulity, it was the season of Light, it was the season of Darkness,
it was the spring of hope, it was the winter of despair, we had everything
before us, we had nothing before us, we were all going direct to Heaven, we were
all going direct the other way - in short, the period was so far like the
present period, that some of its noisiest authorities insisted on its being
received, for good or for evil, in the superlative degree of comparison only.

Quote break.

> Oh yeah, you can _put_ **Markdown** into a blockquote.
```

> It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way – in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.

Quote break.

> Oh yeah, you can *put* **Markdown** into a blockquote.

Horizontal Rule

```
Three or more hyphens, asterisks, or underscores...

---

Gives you a horizontal rule
```

Three or more hyphens, asterisks, or underscores...

Gives you a horizontal rule.

Further resources

- Markdown reference: http://commonmark.org/help/
- A quick tutorial: http://commonmark.org/help/tutorial/
- A more extensive cheat sheet: https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet
- Desktop-based Markdown editors with live previews
  - Mac & Windows: Typora (https://typora.io/)
  - Windows: Markdownpad (http://markdownpad.com/)
  - Mac: MacDown (http://macdown.uranusjr.com/)
- Web-based Markdown authoring tools:
  - A minimal in-browser markdown editor: (http://bioconnector.org/markdown-editor).
  - Dillinger (http://dillinger.io/): connects to Dropbox, Github, Google Docs, and OneDrive.
  - StackEdit (https://stackedit.io/): connects to Dropbox, Google Drive, and can push documents directly to a blog hosted on Blogger, Tumblr, Wordpress, etc.

RMarkdown

What is RMarkdown?

RMarkdown is an enhanced version of Markdown that lets you embed R code into the document. When the document is compiled/rendered, the R code is executed by R, the output is then automatically rendered as Markdown with the rest of the document.

RMarkdown code chunks

RMarkdown code as written

When writing an RMarkdown document, put R code to be executed in a fenced block with `{r}` after the first set of fences. Here's an example of what RMarkdown looks like:

````
Here's some plain old Markdown text. We can use _italics_ or **bold** or whatever we want here. But che

```{r}
# This is a comment.
# This is the start of an R code chunk.
# Let's generate 10 random numbers between 1 and 100
set.seed(42)
x <- sample(1:100, 5)
x
```

Now we're out of our R code and back into plain Markdown. Why don't we take the mean of `x`?

```{r}
mean(x)
```

Neat eh?
````

Result after rendering

Here's some plain old Markdown text. We can use *italics* or **bold** or whatever we want here. But check this out below. We're going to embed an R code chunk.

```r
# This is a comment.
# This is the start of an R code chunk.
# Let's generate 10 random numbers between 1 and 100
set.seed(42)
x <- sample(1:100, 5)
x
```

```
## [1] 92 93 29 81 62
```

Now we're out of our R code and back into plain Markdown. Why don't we take the mean of `x`?

```r
mean(x)
```

```
## [1] 71.4
```

Neat eh?

How does it work?

The workflow looks something like this.

1. You write the RMarkdown document. It's just regular Markdown with R code chunks scattered throughout.
2. When you hit the "Knit" button in RStudio, a package called knitr is envoked in the background to go through your RMarkdown document. Every time it encounters an R code chunk, it actually runs that R code in R. Both the code in the chunk and the result of the evaluation are then woven back into a regular markdown document as valid markdown syntax. You never see this intermediate plain markdown file.
3. A universal text format conversion tool called Pandoc is then called to convert that intermediate Markdown text into some other downstream format (HTML, PDF, DOCX, etc.).

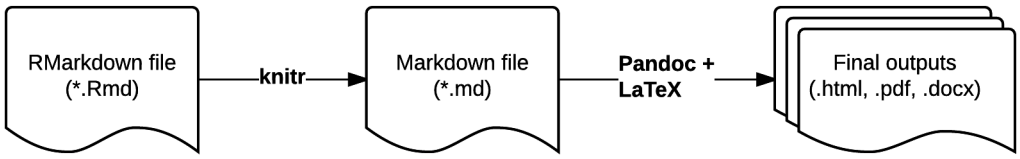

Figure 2:

Inline RMarkdown

In addition to writing chunks of R code, you can also write R code inline with the rest of your Markdown using the syntax `r <code here>`. For example, if you had this line in your RMarkdown document, outside of a chunk:

```
Two to the eighth power is `r 2^8`. Yay!
```

Then the resulting output would include the number 256 in place of the `r 2^8`.

Options

Modify the behavior of an R chunk with options. Options are passed in after a comma on the fence, as shown below.

````
```{r, echo=TRUE, results='hide'}
# R code here
```
````

Some commonly used options include:

- `echo`: (`TRUE` by default) whether to include R source code in the output file.
- `results` takes several possible values:
  - `markup` (the default) takes the result of the R evaluation and turns it into markdown that is rendered as usual.
  - `hide` will hide results.
  - `hold` will hold all the output pieces and push them to the end of a chunk. Useful if you're running commands that result in lots of little pieces of output in the same chunk.
  - `asis` writes the raw results from R directly into the document. Only really useful for tables.

- `include`: (`TRUE` by default) if this is set to `FALSE` the R code is still evaluated, but neither the code nor the results are returned in the output document.
- `fig.width`, `fig.height`: used to control the size of graphics in the output.

See the full list of options here: http://yihui.name/knitr/options/. There are lots!

Printing tables nicely

The knitr package that runs the RMarkdown document in the background also has a function called `kable` that helps with printing tables nicely. It's only useful when you set `echo=FALSE` and `results='asis'`. Check out the difference below.

Without using `kable`

Code in the RMarkdown document:

````
```{r, echo=FALSE}
head(mtcars)
```
````

Resulting intermediate Markdown:

````
```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```
````

Rendered output:

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

Results using `kable`

Code in the RMarkdown document:

````
```{r, echo=FALSE, results='asis'}
library(knitr)
kable(head(mtcars))
```
````

Resulting intermediate Markdown:

```
|                  |  mpg| cyl| disp|  hp| drat|    wt|  qsec| vs| am| gear| carb|
|:-----------------|----:|---:|----:|---:|----:|-----:|-----:|--:|--:|----:|----:|
|Mazda RX4         | 21.0|   6|  160| 110| 3.90| 2.620| 16.46|  0|  1|    4|    4|
```

```
|Mazda RX4 Wag     | 21.0|   6|  160| 110| 3.90| 2.875| 17.02|  0|  1|    4|    4|
|Datsun 710        | 22.8|   4|  108|  93| 3.85| 2.320| 18.61|  1|  1|    4|    1|
|Hornet 4 Drive    | 21.4|   6|  258| 110| 3.08| 3.215| 19.44|  1|  0|    3|    1|
|Hornet Sportabout | 18.7|   8|  360| 175| 3.15| 3.440| 17.02|  0|  0|    3|    2|
|Valiant           | 18.1|   6|  225| 105| 2.76| 3.460| 20.22|  1|  0|    3|    1|
```

Rendered output:

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

Learn more

- The knitr website (**http://yihui.name/knitr/**) has lots of useful reference material about how knitr works, options, and more.
- The RMarkdown documentation (**http://rmarkdown.rstudio.com/**) has an excellent getting started guide, a gallery of demos, and several articles illustrating advanced usage.
- RStudio's RMarkdown Cheat Sheet and RMarkdown Reference Sheet.