Introduction to Markdown and Modern Document Workflow

Jiawei Li

21 April 2021

Abstract

Outputting documents has been a critical step when doing Data Science. After finishing your project, your professors or supervisors typically ask you to upload a PDF for reporting. Yes, Microsoft Word (and other WYSIWYG software) is great but there are certainly options for the rest of us who want to focus on writing rather than formatting, include citations and math equations painlessly, and execute code to generate data visualizations. During this talk, I will first give a brief introduction on why I prefer Markdown to Microsoft Word and LaTeX. Then, I will present the basics of Markdown syntax with a minimal Markdown editor Typora. Finally, it will be a tour on modern document workflow using a selection of tools in the Markdown ecosystem, i.e. R Markdown and Pandoc.

Introduction

There are so much pain to use Microsoft Word (and other WYSIWYG software). Some of them can be mitigated and some of them can not. Because there is no separation of content and formatting, when you paste something from web pages, you have to try to clear or brush off the awkward font size. This also means that you are constantly distracted by the fonts or buttons. There is also little code syntax highlighting, math equation and citation support. Finally, you can hardly generate graphs and visualizations.

On the other hand, academics and power users tend to use LaTeX for complex document rendering. However, LaTeX source code is way too bloated and doesn't really separate content from formats. Here is an example, we can see the code that describes the font size and paper size, makes title, underline some words and begins the document. Even though these code are power, they are still a distraction.

```
\documentclass[12pt, a4paper]{article}
\usepackage[utf8]{inputenc}

\title{A Simple Document}
\author{Jiawei Li}
\date{April 2021}
\begin{document}
\maketitle
Some of the \textbf{greatest} discoveries in \textit{science} were made by \textbf{\textit{accident}}
\end{document}
```

Can we write something simple and deal with fancy stuff later? Yes, use Markdown. Markdown is intended to be easy-to-read and easy-to-write even in a text message. Here is the Markdown code that generate the exact same document:

```
---
title: A Simple Document
```

```
author: Jiawei Li
date: April 2021
fontsize: 12pt
papersize: a4
---
```

```
Some of the **greatest** discoveries in *science* were made by ***accident***.
```

Learn Markdown Syntax with Typora

Markdown Basics

A document typically is divided into multiple sections. To make a new section, you give it a header. The headers start with a hashtag (`#`). Using one hashtag means a level-one header. Two hashtags means a level-two header.

```
# Introduction

## Motivation

## History

### 1910s

### 1920S

# Main Texts

# Conclusion
```

We emphasize part of a text by putting it in italic or boldface. In Markdown, we wrap asterisks (`*`) around the word to achieve this. One asterisk puts the work in italic and two asterisks put it in boldface.

```
Some of the **greatest** discoveries in *science* were made by ***accident***.
```

To create a numbered list, you put a number, followed by a period, at the start of a line and write the list item after it.

```
1. This is a numbered list.
2. Where this is list item two.
3. And this is list item three.
    1. A numbered list inside a list.
    2. This is item 3.2.
```

To create a unnumbered list, you simply put a dash (`-`) or an asterisk (`*`) in the beginning of the line.

```
- This is a unnumbered list.
- Where this is list item two.
- And this is list item three.
    - You can also indent a list here.
```

To add a quote to your text, put a `>` before the quoted text.

```
> The idea is that a Markdown-formatted document should be publishable
> as-is, as plain text, without looking like it's been marked up with
```

```
> tags or formatting instructions.
> If you want multiple lines where you include new lines,
> you should add the `>` to each line.
```

Exercise 1: Countries' GDP

1. Make a top level header "EU Countries Ordered by GDP".
2. Quote the famous "in the long run we all die" sentence by Keynes.
3. Create a numbered list of countries below ordered by their GDP. Note that some nations of The UK should be indented.
4. Emphasize **Germany** by making it bold.

Bonus: Strike UK and its nations since it has left EU.

Links, Images, Code and Math

Links can be created by using both `[]` and `()`.

```
[This is a text for the link](the link itself)
```

Image is simply a link with `!` in the beginning.

```
![This is a caption for the image](the link to the image)
```

Inline code can be identified by backticks.

```
Variable can be assigned using `a = 12`.
```

Code block should be surrounded by three backticks.

````
```python
print("Hello World!")
```
````

Math blocks follows the same logic and uses the dollar sign (`$`). Inline math is wrapped by the dollar sign.

```
Here we define $m = ab$.
```

A math block is surrounded by two dollar signs.

```
We know that

$$
e = mc^2
$$
```

Exercise 2: Badges and Stuff

Shields.io is a website to generate badges, for example a YouTube channel subscribers' count, that can be embedded in your blog post. During this exercise, you are going to create two YouTube channel badges which are clickable. Then you should use a thumbnail image which is also clickable. You then finish off with a line of Python code and a math equation.

Bonus: Add another formula which is Euler's identity.

Use R Markdown with RStudio

Metadata and Citation

We can include some metadata that describes the formats and detailed information in the header of the markdown file, e.g. title and author.

```
---
title: A Report Genrated by R Markdown
author: Jiawei Li
abstract: Here are some words.
---

# Introduction
This is your texts.
```

The margins, document class (article, report or book) and fonts can be described as well. For options available in the header, refer to Pandoc User's Guide.

```
---
geometry:
  - top=30mm
  - left=30mm
  - right=30mm
  - bottom=30mm
  - heightrounded
fontsize: 12
toc: true
toc_depth: 2
documentclass: article
---
```

When I use table of contents and documentclass is article, I usually add a `\pagebreak` after the header to make the table of contents separated from my main contents.

```
---
toc: true
documentclass: article
---
\pagebreak
```

Citations can be inserted in the header as well, though you need generate a BibTex file from ZoteroBib or other citation generator. If you wants to use a specific citation style, you also need download a `csl` file from Zotero Style Repository. For more information, refer to R Markdown documentation and Pandoc User's Guide.

```
---
bibliography: citation.bib
csl: harvard-cite-them-right.csl
---
```

Exercise 3: Theses

Generate a thesis with author name, date, abstract and a table of contents. Cite the book "Forecasting: Principles and Practice".

Extra exercise: Each section should be numbered. For example, “Introduction” is “1. Introduction”. You may not write these numbers explicitly.

Table and Graphs

Simply add a `{r}` in your code fence, the code should be executable in RStudio and generate table and graphs for you the report.

````
```{r, include = FALSE, echo = FALSE}
library(tidyverse)
```

```{r}
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy, color = class))
```
````

Templates

After install `stevetemplates` and `binb`, you can either start using the templates when create a new R Markdown document or put the template in the `output` field. However, different templates require different dependencies from LaTeX and sometimes they may not run on your computer, this requires some basic knowledge of programming and LaTeX to tweak and debug. Here are some of templatates that I find clean and beautiful:

- svmiller/stevetemplates
- eddelbuettel/binb
- kjhealy/pandoc-templates
- ihrke/markdown-paper

How R Markdown Works

Internally, R Markdown run the code in the file and generate the output as well as the text you wrote in a Markdown file. This md file will then be translated into PDF or html using Pandoc. Therefore, if you are comfortable with command line and do not need code execution, simply run like this:

```
pandoc --citeproc --pdf-engine=xelatex -o your_work.pdf your_work.md
```

Conclusion

> We shape our tools and our tools shape us. – Marshall McLuhan