Markdown template for scientific report using pandoc

Marc Weber

2017.09.26

Background

Introduction

This document is a template for academic writing, science notebook records and reports. It is written in the Markdown format: it's plain text, human-readable, and can easily be translated into other formats. It allows to write easily structured text with figures, mathematical formulas and bibliographic references without bothering about formatting. One of the biggest advantage is that by separating writing from formatting, I can keep a single source file in a lightweight text format and generate several different output documents in an automatic fashion. These output, for the moment, are: i) a PDF document generated by LaTeX, ii) a stand-alone HTML with integrated images and iii) a Wordpress post on my personal website.

Resources:

- Writing academic papers in plain text with Markdown and Jupyter notebook | Sylvain Deville
- Writing Workflow 2016, Part 1: Markdown Writing Environment · v & r

Citations

Citations

Citations are managed by the `pandoc-citeproc` package (see pandoc manual section about citations). The use is as follows, now I want to insert an inline citation here: Gillespie (1977). A citation in square brackets format it with parenthesis: (Miller 1965). Then, I could also use a grouped citations to multiple

references like this: (Danino et al. 2010; Prindle et al. 2012). We can also use citations in footnotes.[1]

Citations go inside square brackets and are separated by semicolons. Each citation must have a key, composed of '@' + the citation identifier from the database, and may optionally have a prefix, a locator, and a suffix. To make your citations hyperlinks to the corresponding bibliography entries, add `link-citations: true` to your YAML metadata.

Outline header 1

Outline header 2

Outline header 3

Outline header 4

Outline header 5

This is the outline text with a short list. This is the outline text with a short list. This is the outline text with a short list. This is the outline text with a short list. This is the outline text with a short list. This is the outline text with a short list. This is the outline text with a short list. This is the outline text with a short list.

- item 1
- item 2
- subitem 1
- subitem 2

Important note: we need an empty line just *before* the list for pandoc markdown. In other markdown flavors this is not necessary.

We also test here the "citation" html tag: This is a short citation but we probably could use the quote tag for this, so it seems redundant..

Code highlighting

This is an inline code example in html `<pre class="lang:python">code example</pre>`.

This is a code block in python language:

[1] Si2017

```
print("this is a test")
def new_function(var1=23):
    return [x for x in range(10) if x > 2]

raise SystemExit
```

This is a pre block:

```
print("this is a test")
def new_function(var1=23):
    return [x for x in range(10) if x > 2]

raise SystemExit
```

This is a code cell in jupyter notebook:

```
# css: "/users/lserrano/mweber/Research_Dropbox/Python_mwTools/CSS_for_scientific_reports/p
# css: "/users/lserrano/mweber/Research_Dropbox/Python_mwTools/CSS_for_scientific_reports/g
print("test")
def function(arg):
    return list([x for x in range(0,10)])
```

Latex math

Latex formulas are rendered by different plugins depending on the output file format. The html for wordpress output will use QuickLatex wordpress plugin. The self-contained html output will use MathJax script. And the pdf output will use the xelatex Latex engine.

How to write latex expressions in the markdown master file

In the *markdown master file*, we should wrap latex expression with tags that are understood by both the jupyter latex engine and the pandoc markdown math syntax, namely:

- inline math `$ ... $`
- display math `$$ ... $$`
- math environments, e.g. `\begin{align} .. \end{align}`

Then, we will have to tweak pandoc to correctly convert these tags to tags compatible with the different latex engines of each output format.

Editing the markdown master file with Jupyter notebook

If using the jupyter notebook as an editing tool for the markdown master file, we would like to use latex tags that are recognized by the the jupyter markdown rendering engine. Note that in the case of editing the markdown master file

with a simple text editor, this would not be important. As mentionned in the documentation, jupyter uses MathJax to render latex, however with some limitations that are not reported in the documentation.

- inline math `$ ... $`
- display math `$$ ... $$`
- math environments (at least `align` and `multline`, probably all environments supported by MathJax)

Pandoc markdown to pdf using xelatex

From the pandoc documentation:

> Anything between two $ characters will be treated as TeX math. The opening $ must have a non-space character immediately to its right, while the closing $ must have a non-space character immediately to its left, and must not be followed immediately by a digit. Thus, $20,000 and $30,000 won't parse as math. If for some reason you need to enclose text in literal $ characters, backslash-escape them and they won't be treated as math delimiters.

- inline math `$ ... $` with the whitespace restrictions.
- display math `$$ ... $$`
- math environments

Wordpress post with QuickLatex plugin

QuickLatex supports many Latex features, as AMS math, and also custom preamble, which makes it very powerful. Also, all equations are rendered as SVG images, which display crisp at every zooming scale. Because all equations are rendered once and included as SVG images in the post, the load on the client browser is null resulting in fast download and display of the webpage, unlike MathJax script. The supported latex tags are the same as for MathJax:

- inline math `\( ... \)` (`$ ... $` is also supported but is deactivated on my wordpress installation!).
- display math `\[ ... \]` and `$$ ... $$`
- math environments: `equation`, `align`, `displaymath`, `eqnarray`, `multline`, `flalign`, `gather`, and `alignat`.

Note: unfortunately quicklatex is parsing inside `<code></code>` block. This is why double dollar syntax was getting displayed as latex display equation with dots `...` inside. This could happend quite often when printing inline code with regex with escaped parenthesis, for example `re.sub(r'\(test\)', r'\1', "string")`. There is no problem in seperate code block because these get in a `pre` block, which QuickLatex does not parse. Solution is to enclose inline code in a `span` tag with `crayon-inline` class, which will enable

basic syntax highlighting while preventing any latex parser to mess with it: re.sub(r'\(test\)', r'\1', "string").

```
import re
re.sub(r'\(test\)', r'\1', "string")
0 < 1
re.sub(r'$$abc$$', r'a$$abcd$$', string) = "this should not be parsed as latex expression"
```

Now we have to choose one particular syntax for inline and block code, because there is no standard in html5 about how to do that. People either use specific tags: `<codeblock>`, `<code>`, or the more general `<pre>` tag with class attribute that defines if the code is inline or in block. The Crayon plugin for syntax highlighting that I use on the wordpress site prefers 1) for code block the `<pre class="lang:python">` tag with the class defining the language, and 2) for inline code the `<span class="lang:python decode:true crayon-inline ">` tag with the class giving the language attribute and the `crayon-inline` attribute which defines to insert the code inline. We will use these tags in the output for wordpress post.

Self-contained html with MathJax script

MathJax preprocessor can look for whatever markers to define math delimiters. In the default configuration, the latex tags are

- inline math `\( ... \)`. Important: `$ ... $` is **not** supported by default.
- display math `$$ ... $$`
- math environments
- Need spaces around `<` and `>` symbols in order not to interfere with html syntax, example: `$x < y$`

I wish MathJax could render also all equations as SVG and include them as URI images in the self-contained html file. However, is seems not possible at this time. Latex tags get correctly exported with the pandoc tool by passing the `--mathjax` option.

Examples of LaTeX formulas in markdown master file

Inline math

inline latex expression examples enclosed by `$ .. $`: $\exp(e^{\tau}) < 1$

- jupyter nbconvert to Markdown: `$\exp(e^{\tau)} < 1$` (unchanged)
- pandoc markdown to pdf using xelatex: ok
- self-contained html with MathJax script: ok. Converted to html code `<span class="math inline">\(\exp(e^{\tau)} &lt; 1\)</span>`
- wordpress post: ok, enclosed in span class and recognized by QuickLatex `<span class="math inline">\(\exp(e^{\tau)} &lt; 1\)</span>`

Important: inline latex expression examples enclosed by `\( .. \)` is **not compatible** with Pandoc markdown math syntax and will result in an error in pandoc conversion to pdf using latex engine, for example.

Display math

display latex expression example enclosed by `$$ ... $$`:

$$\exp\left(e^{\tau}\right) < 1$$

- jupyter nbconvert to Markdown: `$$\exp(e^{\tau)} < 1$$` (unchanged)
- pandoc markdown to pdf using xelatex: ok
- self-contained html with MathJax script: ok. Converted to html code `<span class="math display">\[\exp(e^{\tau)} < 1\]</span>`
- wordpress post: ok, enclosed in span class and recognized by QuickLatex `<span class="math display">\[\exp(e^{\tau)} < 1\]</span>`

Math environments

`align` environment:

$$x + y < 0 \tag{1}$$

$$\int_{i=0}^{N} e^{\alpha t} dt = \pi \tag{2}$$

- jupyter nbconvert to Markdown: (unchanged)
- pandoc markdown to pdf using xelatex: ok
- self-contained html with MathJax script: ok. Left unchanged in html code.
- wordpress post: ok, Left unchanged in html code and recognized by QuickLatex.

`multline` environment:

$$\begin{aligned} x + y + c + a + b + d + e + f + g + x + y + c + a + b + d + \\ e + f + g + x + y + c + a + b + d + e + f + g + \\ + 4 + a + b + d < \pi \end{aligned} \tag{3}$$

- jupyter nbconvert to Markdown: (unchanged)
- pandoc markdown to pdf using xelatex: ok
- self-contained html with MathJax script: ok. Left unchanged in html code.
- wordpress post: ok, Left unchanged in html code and recognized by QuickLatex.

Latex in code

The expressions in inline code and code blocks should **not** get parsed as latex expression. Examples inline: `code with latex-like tags inside $s = 1$ and $$x + y < 0$$`. And codeblock,

```
print("code with latex-like tags inside")
$s = 1$
$$x + y < 0$$
```

- jupyter editor: unfortunately, the latex expressions inside code blocks gets rendered as LateX expression.
- jupyter nbconvert to Markdown: ok (unchanged)
- pandoc markdown to pdf using xelatex: ok
- self-contained html with MathJax script: ok.
- wordpress post: ok

Latex macros

New Latex commands can be defined by the `\newcommand` command inside the markdown file. They will be parsed and macros will be replaced every in the file when producing html output. When producing latex pdf output, the macros will be included in the latex header.

Images

Images can be inserted into Markdown documents in a variety of ways, each of one is not compatible with the external software that will convert markdown into PDF, html, or other document type. With Pandoc, the syntax is:

Markdown syntax

HTML syntax

Note that in the case of the markdown syntax, the figure is correctly rendered by the Jupyter notebook interface, **however the figure size directive is not taken into account**. The only way to change the image size in the jupyter interface, which uses standard markdown, is to use an HTML tags (see python - How to Include image or picture in jupyter notebook - Stack Overflow). However, images inserted by html tag will probably not get parsed correctly by pandoc (to verify).

We used a custom syntax for the figure captions, both for markdown and html, in order to be able to display the captions below the plot in a markdown rendered cell in the jupyter notebook. These captions will be converted to the pandoc markdown syntax in the parser script, such that:

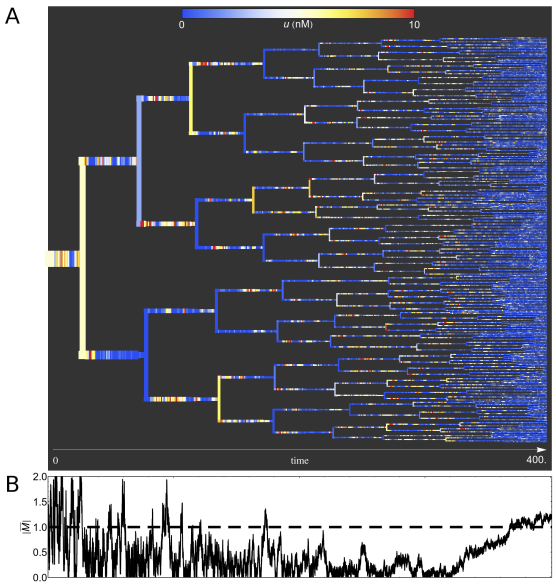


Figure 1: *Phase transition in a growing population of toggle switches interfaced by QS leading to the high u state.* (A) Time evolution of concentration u in color code for an exponentially growing population of toggle switches interfaced by QS.

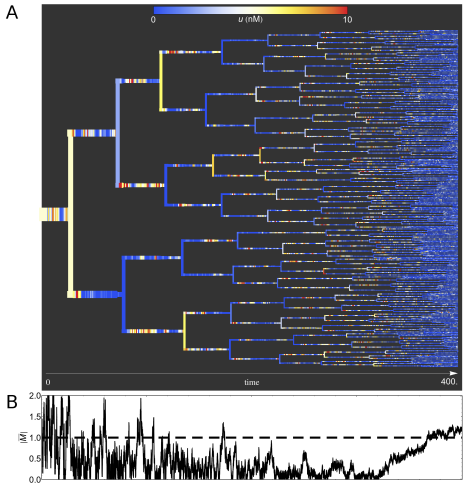


Figure 2: *Phase transition in a growing population of toggle switches interfaced by QS leading to the high u state.* (A) Time evolution of concentration u in color code for an exponentially growing population of toggle switches interfaced by QS.

```
![](Images/image.png){#fig:fig_id width=60%}[[figure caption text.]]
![ figure caption text. ](Images/image.png){#fig:fig_id width="50%"}
```

displays the captions below the figure in the jupyter notebook. The markdown master file will be transformed into the correct pandoc markdown syntax:

```
![figure captions text.](Images/image.png){#fig:fig_id width=60%}
```

The figure number can be automatically referenced using the package `pandoc-fignos` (see github webpage) this is a reference to the figure 1.

Other markdown compatible syntaxes for inserting image

HTML tag:

```
<img     src="Images/Marks2016_chloramphenicol_stalling_motif.png"
width="500px" />
```

General markdown:

```
![la     lune](Images/Marks2016_chloramphenicol_stalling_motif.png
"Voyage to the moon")
```

pandoc style:

```
![legend](Images/Marks2016_chloramphenicol_stalling_motif.png){
width=50% }
```

pandoc style with fignos package:

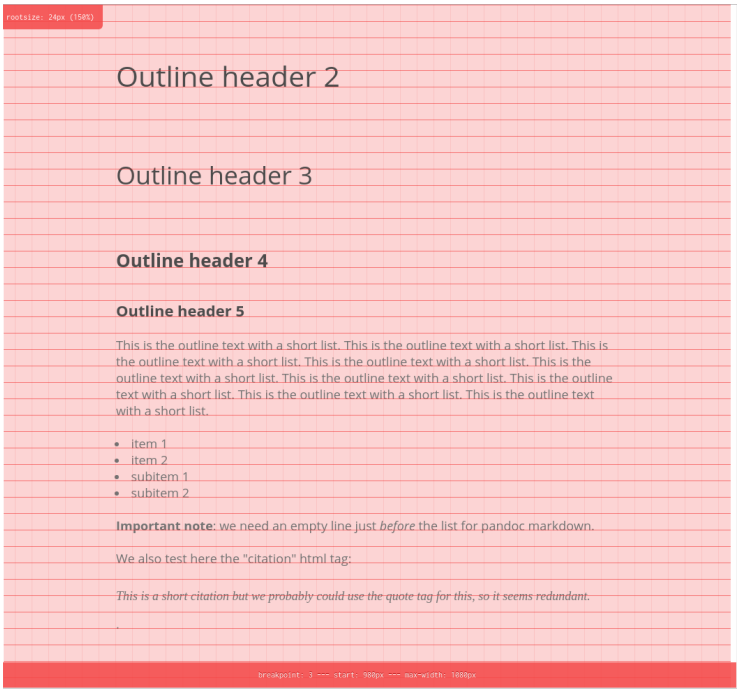
```
![Caption.](image.png){#fig:id}
```

Styling

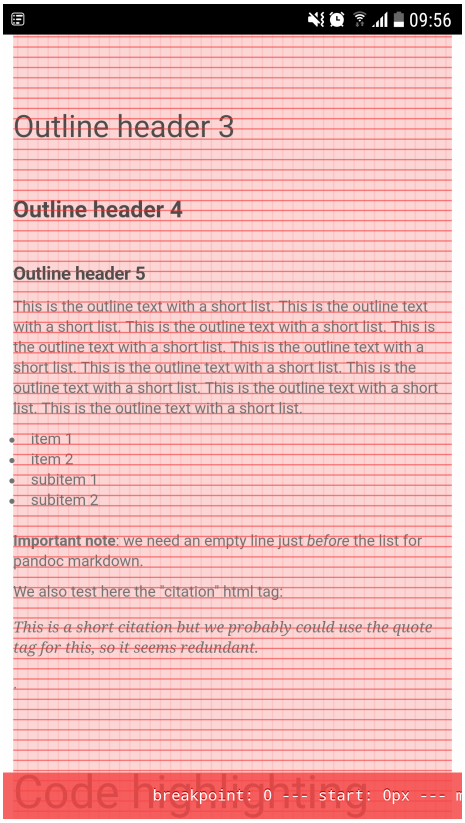
The HTML outputs are styled with custom CSS sheet. I used the package MegaType (SCSS template) to create a responsive design and define all font sizes and line heights of the different elements in order to create a (almost perfect) vertical rhythm, aligning the text baseline on a baseline grid.

More information:

- Web typography is broken. Here's how we can fix it – Thomas Bredin-Grey – Medium
- StudioThick/megatype: Execute typographic structure with ease
- cap-height calculator (useful to calculate the cap height of some common fonts).



Baseline grid on desktop device, google chrome.



Baseline grid on mobile device, google chrome.

Bibliography

Note: the Bibliography header is not automatically included by pandoc.

Danino, Tal, Octavio Mondragón-Palomino, Lev Tsimring, and Jeff Hasty. 2010. "A synchronized quorum of genetic clocks." *Nature* 463 (7279): 326–30. doi:10.1038/nature08753.

Gillespie, D.T. 1977. "Exact stochastic simulation of coupled chemical reactions." *The Journal of Physical Chemistry* 81 (25). American Chemical Society: 2340–61.

Miller, D. G. 1965. "Application of Irreversible Thermodynamics." *J. Phys. Chem* 111 (8): 2639–59. doi:10.1021/j100880a033.

Prindle, Arthur, Phillip Samayoa, Ivan Razinkov, Tal Danino, Lev S Tsimring, and Jeff Hasty. 2012. "A sensing array of radically coupled genetic 'biopixels'." *Nature* 481 (7379). Nature Publishing Group: 39–44. doi:10.1038/nature10722.