learnRmd

Michael Dillon

December 7, 2018

Contents

R Markdown

R markdown has three components: `to force the new line, I added 2 spaces at the end`
1. metadata (between 2 `---` at top of document)
2. text (of any kind, anywhere)
3. and code, surrounded by 3 backticks for a code chunk, or a single backtick for inline calculations (2x2=4).

R Markdown syntax is Pandoc Markdown. For help on all the options, check the pandoc user guide.

Metadata

`note the 3 hashes to make a subheading`

Basics of YAML

The YAML ("YAML ain't markup language") header is picky about formatting, including indentation. Use examples or go to a reference to solve more complicated problems. (see bookdown.org for lots of info). The `keep_md: true` option is nice for hosting on github as the .md renders nicely there.

Table of contents

Can be floating or not, several other options.

Themes

Specify the theme or set it to null and point to a .css document.

Figures

Can specify `fig_width`, `fig_height`, `fig_caption`, `dev`.

Figure 1: **Figure 1**. Nice beetle.

data frame output

Set the `df_print` option to: kable, tibble, paged

Code folding

Can include code, but have it hidden by default by setting html_document, `code_folding: hide`

Templates

See documentation, but this allows for customization of html, latex, pdf, word, etc. such that the markdown text is rendered to preset formatting guidelines.

Text

Again, this is the stuff you are writing to either *document* your **analyses** or discuss them (for example, in a paper).

Some help with text: To add an image, use `![**Figure 1**. Nice beetle.](figures/bug.jpeg)`:

> "Not everyone likes bugs, but I do."
>
> – famous entomologist

Inline latex based equations! $f(k) = \circ p^k 1 - p_n + 2$ (And it updates a box showing you the equation in real time!)

Code

Can be r, python, and many others an you can control whether the code shows, the output shows, or both. Specify options for code chunks in the curly brackets opening the chunk. Some useful ones: `collapse=TRUE` will make the code and the output format into the same block (see cars example below), `eval=TRUE` will actually evaluate the code (vs not for header type info), `echo=TRUE` will echo the source code in the output document (so you can sometimes only show results or both code and output), `include=FALSE` to not include anything from the code chunk in the output (but if eval is true, the code will still be executed; good for setup stuff?)

Compile

Using the knit button or Ctrl-Shift-K.

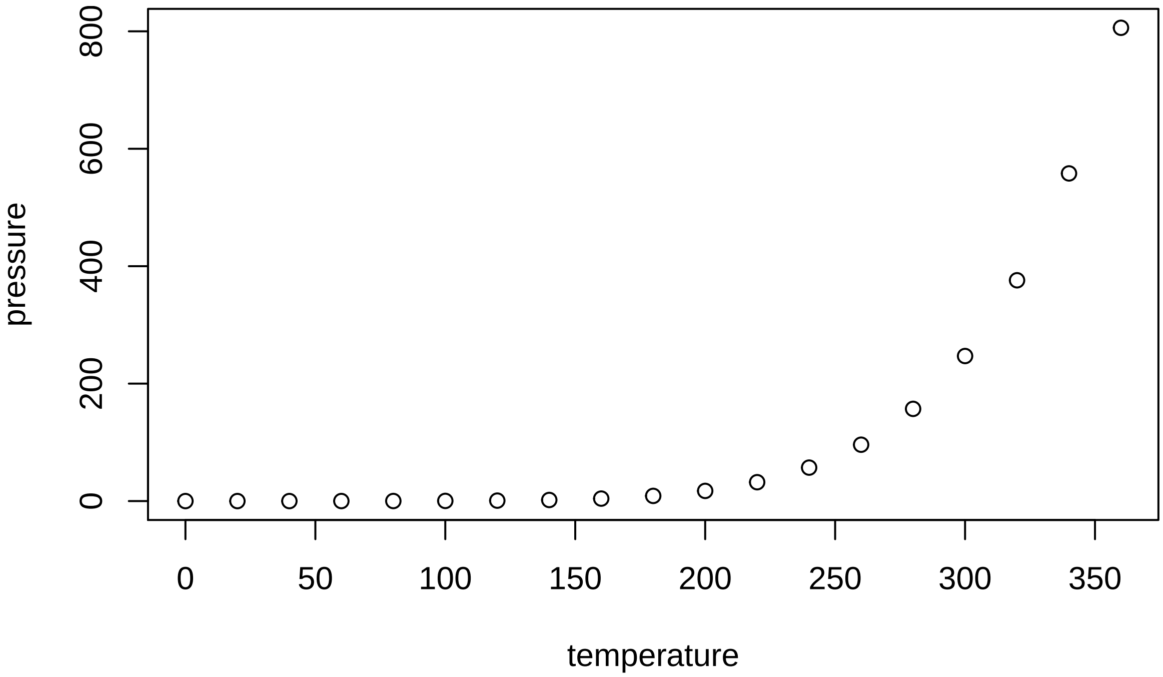
This renders in a new R session … a very important point because it means that anything in the console from the previous session is now gone. This is a good thing, related to why you shouldn't save your .Rhistory as it can lead to issues with reproducibility.

You can also render via the command-line or a script, giving you the ability to have automated report generation but you have to worry about reproducibility in that case.

```
summary(cars)
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00  
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Shared options

To share options across documents within a directory, include a file named `_output.yml` in the directory (the file will have standard YAML syntax but without the opening and closing `---`). All documents within that directory will then inherit yaml settings from that file, with any specifications in an individual file overriding or adding to those “baseline” specifications.