Rmarkdown Basics

CENTER FOR RESEARCH METHODS & DATA ANALYSIS
College of Liberal Arts & Sciences

Paul Johnson, Director, CRMDA <pauljohn@ku.edu>
February 24, 2018

Abstract

This discusses the basics of Rmarkdown that are important for authors who want to formulate their strategies for document preparation. In our center, we expect all authors who prepare documents with Rmarkdown should understand the issues.

1 Introduction: Terminology

`Rmarkdown` is a style of markdown document used for preparing guides and reports about R. It is not the same as "markdown," for which there are many different dialects, and will Rmarkdown does not always follow the same syntax. The general framework to which it is closest is `pandoc` markdown because the program `pandoc` is used in several parts of the document compilation process.

2 Document-wide options

In the beginning of a markdown document, there is a header written in the YAML format (YAML, which originally meant *Yet Another Markup Language* but its developers now prefer *YAML Ain't Markup Language*).

The header for the document you are reading now in PDF is:

```
---
title: "Rmarkdown Basics"
author:
- affiliation: CRMDA
  description: Director
  email: pauljohn@ku.edu
  name: Paul Johnson
output:
  pdf_document:
    fig_caption: true
    keep_tex: true
    latex_engine: pdflatex
    highlight: haddock
    citation_package: natbib
    pandoc_args: [
        --listings
    ]
    template: theme/report-boilerplate.tex
fontsize: 12pt
vignette: >
    %\VignetteIndexEntry{stationery}
    %\VignetteEngine{knitr::rmarkdown}
    %\VignetteEncoding{UTF-8}
header-includes:
-  \usepackage{xcolor}
-  \usepackage{amsmath}
-  \usepackage{amssymb}
-  \usepackage{fancybox}
-  \usepackage{calc}
-  \usepackage{subfig}
addr:
    l1: 1425 Jayhawk Blvd
    l2: Watson Library, Suite 470
    l3: Lawrence, KS 66045-7594
    r1: "Phone: 785-864-3353"
    r2: "Web: http://crmda.ku.edu"
    r3: "Email: crmda@ku.edu"
logo: theme/logo.pdf
keywords: guides, report, pandoc, Rmarkdown, sweave, knitr
---
```

What do we have there? Main sections that begin with keywords `title`, `author`, `output`, `header-includes`, and `keywords`. The output spection specifies `pdf_document`, which means that the `Rmarkdown` file will be converted from `Rmarkdown` to `markdown`, then into LaTeX, then PDF. The `header-includes` section is relevant only to the LaTeXstage of document preparation.

Observations about YAML

1. The header begins with, in column 1

   `---`

   and ends with

   `---`

2. The format of the YAML header is **VERY** sensitive. Errors in formatting are fatal.

   A. Subsections are represented by indentation

   B. Use spaces to intend, NEVER tabs. Use an editor that displays in fixed width fonts, so that the number of spaces is apparent.

3. Parameter and value are separated by `:`, a colon, rather than equal sign.
4. The backend, either HTML or PDF, is specified here. This choice determines the other settings that are allowed.

   Beyond the very simple headers, be aware that it is generally NOT possible to change from a PDF to HTML backend without making a lot of changes in the header. Please be warned that the Rstudio pulldown menu that says "Knit to PDF" or "Knit to HTML" is deceptive, implying one can easily choose between backends and that the document will work well in either output format. This is decidedly untrue. Be warned also that Rstudio will revise the content of the document header, possibly corrupting its formatting.
5. The package `rmarkdown` provides a function named `render`. This function is the orchestrator of the document production process. R code chunks are processed (knitted) into the markdown document, then it is "compiled" into HTML or PDF. The `render` function allows arguments that can override the YAML header arguments. They generally have the same name.

   For example, the header setting `keep_tex: true`, will be overridden by argument `keep_tex=FALSE` given to the render function.

3 Preparing an Rmarkdown Document

3.1 Use a text editor

An R markdown file is suffixed ".Rmd", *not* ".rmd". It is a "raw text" file. Don't edit it with MS Word. Instead, use a programmers file editor, such as Emacs, Notepad++, or an integrated development environment, such as Rstudio, to edit the file.

Remember that the markdown philosophy is that a document should look reasonably nice, even before it is compiled. That means authors should take care that their document has reasonably "shaped" paragraphs. Lines should generally be 80 characters or less.

3.2 Document body has markdown, plus

The body of the Rmarkdown document can include both "markdown" syntax, which we hope is compatible with either PDF or HTML backends, as well as a selection of backend-specific code input. In a document with an HTML backend, authors can write "raw" HTML when they can't find Rmarkdown syntax that works.

In our experience, it is also *mostly* workable to include LaTeX
code in `Rmarkdown` documents. Recall that documents in `Rmarkdown` are converted from `Rmd` to `md` by the knitting process, and then into LaTeX by `pandoc`, and then into PDF by a LaTeX compiler (`pdflatex` or `xetex`). As a result, one can insert quite a bit of LaTeX code and it will often be successfully post-processed in the last stage of document compilation.

We have found that some LaTeX code does not work well, however, so it is not possible at the current time to say "everything works". However, many things do work.

As in LaTeX itself, when an author wants to add features, it is generally necessary to add `\usepackage{}` statements in the LaTeX
document preamble. Note that, as we insert LaTeX code in this document, we need to insert LaTeX `\usepackage code` to make the features available to the compiler. These appear in the YAML header like so.

```
header-includes:
-  \usepackage{amsmath}
```

That one was needed when some math expressions were inserted in the document.

We use a LaTeX template for this document. That template creates a document style and it also can include `\usepackage{}` statements. Packages that are not included in the template (where we include packages that we *always* expect to use) are tucked into the document header.

4 Basic Rmarkdown

These are the most frequently used markdown idioms that work well in both HTML and PDF backends. These things work well in either backend because they are understood to `pandoc`, which has the job to convert the `markdown` document into either LaTeX or HTML.

1. Character styling.

   - *Italics* result from asterisks (`*italics*`)
   - **bold face** results from two asterisks (`**boldface**`).
   - `computer code` should be a typewriter fixed width font produced by using two "back-ticks" (`computer code`)

2. Section headers.

   - `#` is level 1 header
   - `##` is level 2 subheading
   - levels below are used differently in HTML and PDF. The ability of html documents to include specialized subsections is one of the advantages of that backend. In markdown, they provide for 6 levels of headings.

3. Paragraph styling

   - Paragraphs are created by blank lines
   - Single-spaced line: Two blank spaces at end of line signify new line without creating new paragraph (blank line) below.
   - Indented material like this:

     > quotations or other highlighted material (block quotes)

   begins with a > character in column 1, followed by a space:

```
  > quotations or other highlighted material (block quotes)
```

- code listings that are not chunks, such as

```
int x = 3
```

are requested by three ticks before and after

~~~
  ```
  int x = 3
  ```
~~~

4. Lists.
    - Numbered lists can be created by putting a number in column 1 of a new line, followed by a space.
    - Bullet lists can be created by putting an asterix in column 1 of a new line, followed by a space.
    - Lists can be nested by typing four blank spaces for each level of nesting.
    - Question About lists: how do we insert separate paragraphs under list headings and then re-continue the list in Rmarkdown ?? ??.
5. Hyperlinks, such as CRMDA, for which the markdown is `[CRMDA](https://crmda.ku.edu)`.
6. Citations. Need to know how to use natbib citations in a markdown document ?? ???

This *basic markdown* is compatible with either HTML or PDF output. While I am writing in a document intended for PDF, I can use LaTeX code `\emph{italicize}` to *italicize*, as you can see, but I can also use markdown syntax `*italicize*` to *italicize*. If I were intending this document for HTML, I could insert `<i>italicize</i>` to reach the same result. Where possible, we should use the generic syntax rather than backend-specific markup, because this keeps open the door that we might want to change the backend from PDF to HTML at some point. Of course, *the key assumption is that the backend-neutral methods actually work* to get the job done, and that is simply untrue in my experience. Every document worth presenting will have at least one backend-specific feature.

4.1 Math in Markdown documents

There's a fairly broad range of mathematical markup that will work in Rmarkdown documents intended for PDF and HTML. So far as we can tell, documents intended for PDF (which can add LaTeX packages) can exploit all of the mathematical features, while HTML documents do not have that same bragging point. There are other weaknesses in the math tool chaing for HTML documents, to which we shall return.

Here are the highlights.

1. In-line math markup is dollar-sign bracketed, just as in LaTeXitself. We can write `$f_{ij} x_{i} \times y_{j}^2$` for $f_{ij}x_i \times y_j^2$. Using the usual LaTeX format, we can write the Greek letters, binary relations, and other important symbols. Behold:

$$\alpha, \beta, \gamma, \Gamma, \chi, \approx, \neq, \geq, \leq, \pm, \sim, \rightarrow, \pi, \text{and} \infty.$$

2. Displayed equations–equations that are centered or otherwise set apart from the text–can be bracketed by `\[` and `\]`. In LaTeX, it is not longer recommended to use the double-dollar sign notation for displayed equations.

Compare the inline mode, $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$ with the display equation:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i.$$

that results from

```
\[
\bar{x} = \frac{1}{n} \sum_{i=1}^{n}x_{i}.
\]
```

After some testing, it appears that in order to have equations with numbers (via labels, allowing cross references), the best avenue is to change the display markup to use `amsmath` equations. In the following, we use the LaTeX `\begin{equation}`, which comes from the LaTeX package *amsmath*.

```
\begin{equation}
\bar{x} = \frac{1}{n} \sum_{i=1}^{n}x_{i}.
\label{eq:mean}
\end{equation}
```

This equation is numbered, because the label macro is included within the equation:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i. \tag{1}$$

Because we labeled the equation, we can now use cross references. As the reader can see in equation (1), the arithmetic mean is a very lovely formula. If we insert equations before or after that one, they will be renumbered automatically and cross references will be adjusted.

Another benefit of introducing the amsmath support is that arrays can be demonstrated.

$$A_{m,n} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \tag{2}$$

Given the fact that the `equation` environment provides all of the needed features, it seems obvious that authors should prefer to use it, rather than the simpler `\[` markup.

This is not the right place for a comprehensive guide to LaTeX. Users will need to install a LaTeX distribution and then a user-friendly editor that will help create equation markup. Many users in the CRMDA lab use LyX, which offers a "View source" feature to inspect LaTeX coding for equations.

4.2 Syntax and code chunks

Code chunks are discussed in a separate vignette that is available with the `stationery` package, "Code Chunks". Code chunks are allowed in `Rmarkdown` documents intended for either HTML or PDF.

In `Rmarkdown` documents intended for HTML output, `knitr` is the only chunk-processing technology that is available. In `Rmarkdown` documents, the outer boundaries of a chunk are demarcated by three back ticks, along with squiggly braces and the letter "r", which designates the language being processed (`knitr` can handle several languages).

~~~
```{r}
lm(y ~ x, data = dat)
```
~~~

In contrast, a chunk in a `noweb`/LaTeX document would look like this:

```
<<>>=
lm(y ~ x, data = dat)
@
```

Options which control chunk management are quite elaborate in `knitr`, but most authors will find that there are 5 (or so) options that they use frequently and the others can be discovered in the documentation when emergencies arise:

1. eval
2. echo
3. include
4. results

These control if the chunk is calculated, displayed, and included in the output.

`knitr` has fine grained settings for sizing, saving, and placing figures. These are discussed in our vignette, "Code Chunks". We expect that all authors will not have too much difficulty creating chunks that

1. are not evaluated, but are displayed "beautifully" to the reader.
2. are not displayed, but are evaluated, and the results may (or may not) be displayed for the reader.
3. create graphics, which are automatically included in the document.
4. create graphics (or other files) that are not automatically included in the document. Graphics are saved in image files that can be inserted at a different part in the document
5. import previous chunks for re-analysis.

4.3 Presentable tables

In a report document, it would be unusual to display a large chunk of R code or output. Instead, reports typically include stylized tables, graphs, and equations.

Unfortunately, this is a point at which the backend-neutral document processing strategies break down. Markdown allows some elementary tables, but these are unsuitable for statistical tables. Instead, one needs to create backend-specific tables in order to have something presentable.

The very nice table-making routines for R were originally developed for PDF/LaTeX, and alternative backends like HTML were added later. At the current time, regression tables can be generated in either LaTeX or HTML by the R package `texreg`, `xtable`, `rockchalk`, and many others.

4.3.1 Warnings aside, here are 2 table-making recipes

I have used two approaches to include tables in documents.

1. The "automatic" method. Write a code chunk that creates an output table that is good enough to go into the report without revision and allow the document to use it. The chunk option `results="asis"` is used to signal that the result of the chunk is valid markup that is to go directly into the backend document.

2. The "slow lane" method. Write a chunk that creates a table in a file on disk. Then one can review and edit the file, and include it in the document manually.

In "guides" and less formal documents, I often use the automatic method (beauty of presentation is not the primary emphasis). In reports to clients, I often need to use the second method because tables almost never come out in a perfectly presentable format.

Because I wrote the `outreg` function in the rockchalk a long time ago, it still holds a special place in my heart. I will use it for a demonstration. `outreg` works with standard R regressions (`lm`, `glm`). Suppose we have data about cancer prevalence (y) and predictors including exposure to plutonium (x1), vegetables (x2), and tobacco (x3).

Regressions are estimated. The second regression includes interactions.

```
m1 <- lm(y ~ x1 + x2 + x3, data = dat)
m2 <- lm(y ~ x1 + x2*x3 + x1*x3, data = dat)
```

The variable labels are created as `varLab` and they are supplied in the call to `outreg`.

~~~
```{r reg20, results="asis"}
varLab <- c(y = "Cancer", x1 = "Plutonium", x2 = "Veggies",
            x3 = "Tobacco", `x2:x3` = "Veg $\\times$ Tob",
            `x1:x3` = "Plut $\\times$ Tob")
or10 <- outreg(list("Cancer OLS" = m1, "Cancer Interaction" = m2),
               tight=FALSE, varLab = varLab)
```
~~~

Table 1: Regression Analysis

	Cancer OLS		Cancer Interaction	
	Estimate	(S.E.)	Estimate	(S.E.)
(Intercept)	-748.645***	(5.643)	-222.425***	(17.264)
Plutonium	16.189***	(0.082)	10.869***	(0.284)
Veggies	14.144***	(0.117)	8.731***	(0.259)
Tobacco	0.110	(0.121)	-10.275***	(0.342)
Veg × Tob	.		0.105***	(0.005)
Plut × Tob	.		0.103***	(0.005)
N	1000		1000	
RMSE	26.573		18.772	
R^2	0.987		0.994	
adj R^2	0.987		0.994	

$*p \le 0.05{**}\ p \le 0.01{***}p \le 0.001$

The table is inserted flush left in the document at the point of the code chunk because I did not create a floating table object in which to situate it.

The other way to do this is to save the table in a file, and then include it in a document. I do this in some reports because the LaTeX (or HTML) tables created by software are never *exactly* right and I want to fine tune them.

```
cat(or10, file = "tmpout/or10file.tex")
```

The file "or10file.tex" can be edited by hand, and then inserted into the LaTeX document (for the PDF backend). If I am using this method, I can insert a floating table with a proper caption and label. See Table 1, which results from the following.

```
\begin{table} \caption{Regression Analysis\label{tab:reg30}}
\input{tmpout/or10file.tex} \end{table}
```

Admittedly, this two step process to create the floating LaTeX label is not truly necessary. The `outreg` function could do that for us. The same result using the results="asis" chunk:

~~~
```{r reg40, results="asis"}
or10 <- outreg(list("Cancer OLS" = m1, "Cancer Interaction" = m2),
   tight=FALSE,
               varLab = varLab, title = "Regression Analysis
                  asdfasdf",
               label="tab:reg30")
```
~~~

Because this document is intended for the PDF backend, any LaTeX
generating table function should work. For linear regressions, multilevel models, and glm fits, I find the `rockchalk::outreg` output is fine, but other kinds of models often arrive with custom table-making routines. I have not found a single approach that works beautifully for all kinds of regressions, and I expect other authors will say the same thing because new packages to create

LaTeX tables seem to crop up in the R repositories on a monthly basis. It is worth mentioning that some of the LaTeX table generators, like texreg, allow the use of bookman markup to beautify the placement of horizontal lines, so the document preamble must be modified to include that package.

I often need to include simpler tabular material in documents and I always try to use the package xtable. xtable is not the simplest package, but it has been around longer than the others and it has more detailed features than any other package. In particular, the longtable class, which allows tables that gracefully split over "page breaks", is available. If I cannot get the result I want with xtable, the second approach I try is the latex function in the Hmisc package. There are plenty of other table-making R packages, but my opinion is that if one can master xtable, then one can generate just about any kind of table.

4.4 Warnings, cautions and other hysterical statements

Rmarkdown is similar to, but the exactly the same, as the general markdown specification. It is different from other specialized dialects, such as the GitHub Flavored Markdown and other variants. Be cautious about using style guides that are not intended explicitly for Rmarkdown documents. There is a good chance their advice is irrelevant. Google searches will sometimes lead to the wrong answer for a particular form of Markdown.

Another major concern is that illegal LaTeX code in documents intended for HTML will be ignored in the document production process. The document compiles without error, but HTML result is incomplete. Where we expect to see math, there is *empty nothing.* By itself, this problem seems serious enough to make me think the HTML backend should not be used. The markdown → HTML conversion is the weak spot. The markdown → PDF transition is not similarly flawed. It halts with error messages. The conclusion from all of this is the following:

> *Authors must exert themselves to proofread the HTML output to make sure it appears as intended.*

Another problem we have discovered is that even valid HTML markup can sometimes be corrupted by pandoc. Valid HTML (and PDF, for that matter) can include empty spaces that are inserted by authors to make their markup easier to read. The spaces are confusing to pandoc, which may interpret them as markdown formatting statements (markdown uses spaces to indicate nesting of lists or code sections. At the current time, it appears to be necessary to process all HTML markup that goes into the markdown document to remove spaces in the beginning of lines.

In our experimentation with Rmarkdown, we have noticed another quirk. I document intended for PDF, one writes \LaTeX to produce LaTeX. On the other, in a document intended for HTML, writing \LaTeX produces one of those empty spaces previously mentioned. Instead, in the document intended for HTML one must insert $\LaTeX$, as if this were a valid mathematical expression, which it is not. Inserting $\LaTeX$ in Rmarkdown intended for PDF results in an error message:

```
! You can't use `\spacefactor' in math mode.
\@->\spacefactor
                \@m {}
l.934 ...to compile and the error message \(\LaTeX

pandoc: Error producing PDF
```

Finally, there is a fundamental weakness of HTML documents. Math is not "in" the HTML file. Instead, there is javascript code that is shown in the Web browser using a framework called MathJax. MathJax allows inclusion of math markup in the page which–when the conditions are right–can be converted by the Web browser to look like equations. The MathJax display system works better and better over time; it *almost always works.* The Web browser must be able to retrieve files from the MathJax server in order to display the document. If the reader is offline, or the MathJax server fails. Users will see markup, rather than equations, when the network fails.

5 # Bibliographical citation test is appropriate.

The leaders in the field (Diggle et al. 2013) are sure that I'm correct about most everything. Diggle and colleagues (2013, 37) also seem confident that they did not agree to the preceeding. Note that to get the full parenthesized statement with names and dates, we insert hard brackets `[`, an `@` sign, the bibtex tag, and acloser `]`. If we don't want their names, we insert a `-` sign. It is also possible to refer to a group of projects (Hsiao 2014; Fitzmaurice, Laird, and Ware 2011; McCullagh 1983).

It is a little bit frustrating that the Rmarkdown idea requires the references to be in the very last header of the document. I would rather place them myself, where I want them. They want us to insert

```
# References
```

to get automatic placement. The very bad thing I notice is that even if you put that in the second-to-last header, the citations will appear at the very last part of the document, but the word `References` will appear where you put it.

The style of the bibliography will not match what we generally want. Is close, the Chicago author-year format. We need a "CSL" file to alter the style of the output. See authoring_bibliographies_and_citations.

Unfortunately, it appears this is difficult to get the references in the way we want them in PDF output. This is another reason that I think we should use LaTeX documents rather than Markdown when we intend to output PDF.

6 Conclusion

For the sake of clarity, then, I state the following:

> If the backend in PDF, one should should use functions that provide tables using LaTeX code.

On the other hand,

> if the backend is HTML, then functions that create HTML markup code should be used.

7 Reading Material about Rmarkdown

1. *R Markdown Reference Guide*
2. *R Markdown Cheatsheet*
3. Yihui Xie. 2015. *Dynamic Documents with R and Knitr. 2ed* Boca Raton, Florida: Chapman Hall/CRC. http://yihui.name/knitr. On July 7, 2017, Xie made available a new book manuscript, [*bookdown: Authoring Books and Technical Documents with R Markdown*] (https://bookdown.org/yihui/bookdown).

User-friendly overviews and tutorials on Rmarkdown usage are available.

1. *Markdown Basics*. See also the Rmarkdown page at Rstudio corporation, http://rmarkdown.rstudio.com, which offers a
   "Get Started" Tutorial.
2. Cosma Shalizi, "Using R Markdown for Class Reports", http://www.stat.cmu.edu/~cshalizi/rmarkdown.
3. Wisconsin University Social Science Computing Collaborative. "R for Researchers: R Markdown".
   http://www.ssc.wisc.edu/sscc/pubs/RFR/RFR_RMarkdown.html
4. Karl Broman, "Knitr with R Markdown".
   http://kbroman.org/knitr_knutshell/pages/Rmarkdown.html

8 Session Info

```
warnings()
```

```
NULL
```

```
sessionInfo()
```

```
R version 3.4.3 (2017-11-30)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 17.10

Matrix products: default
BLAS: /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.7.1
LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.7.1

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  base

other attached packages:
[1] rockchalk_1.8.110 stationery_0.73

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.15       compiler_3.4.3     nloptr_1.0.4
 [4] plyr_1.8.4         methods_3.4.3      tools_3.4.3
 [7] digest_0.6.15      lme4_1.1-15        evaluate_0.10.1
[10] nlme_3.1-131       lattice_0.20-35    mgcv_1.8-23
[13] openxlsx_4.0.17    Matrix_1.2-12      yaml_2.1.16
[16] parallel_3.4.3     SparseM_1.77       pbivnorm_0.6.0
[19] stringr_1.2.0      knitr_1.19         MatrixModels_0.4-1
[22] stats4_3.4.3       rprojroot_1.3-2    nnet_7.3-12
[25] grid_3.4.3         foreign_0.8-69     rmarkdown_1.8
[28] lavaan_0.5-23.1097 minqa_1.2.4        car_2.1-6
[31] magrittr_1.5       backports_1.1.2    htmltools_0.3.6
[34] MASS_7.3-48        kutils_1.34        splines_3.4.3
[37] pbkrtest_0.4-7     mnormt_1.5-5       xtable_1.8-2
[40] quantreg_5.35      quadprog_1.5-5     stringi_1.1.6
```

References

Diggle, Peter, Patrick Heagerty, Kung-Yee Liang, and Scott Zeger. 2013. *Analysis of Longitudinal Data*. 2 edition. Oxford: Oxford University Press.

Fitzmaurice, Garrett M., Nan M. Laird, and James H. Ware. 2011. *Applied Longitudinal Analysis*. 2nd ed. Wiley Series in Probability and Statistics. Hoboken, N.J: Wiley.

Hsiao, Cheng. 2014. *Analysis of Panel Data*. Third edition. Econometric Society Monographs 54. New York, NY: Cambridge University Press.

McCullagh, P. 1983. *Generalized Linear Models*. Monographs on Statistics and Applied Probability 37. London: Chapman; Hall.