STAT 1361: Introduction to R Markdown

Nick Kissel adapted from Tim Coleman

January 2019

R Markdown

Introduction And Formatting

Rmarkdown is a way of presenting R code nicely while performing analyses. Rmarkdown takes a base file (.RMD) and "knits" it into one of the three following file formats:

- an `html` file
- a `pdf` file
- a `doc` file (the filetype associated with Microsoft Word)

Rmarkdown is not a what-you-see-is-what-you-get editor (unlike say, Microsoft word). Instead, there are conventions for formatting output that you should become familiar with. A quick guide:

- Use the "*" symbol to create a bulleted list (like we're doing right now!)
  - can also use "-" or "+"
    - note: whether you use *, -, or +, the bullet shape won't change. Indentation determines the shape.
- To bold font, use either **two underscores** before and after or **two asterisks** before and after
- For italics, use either a single *underscore* or a single *asterisk*!

You can also run rudimentary LaTeX in R markdown, like:

$$\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

$$\mathbb{E}(\bar{X}_n) = \mu$$

This can be helpful for including mathematical expressions in your work.

What is knitting? Knitting takes the input .Rmd file and converts it (via a chain of programs) to your desired output.

You can think of a .RMD file as being composed of 3 different elements:

- Plain text
- TeX
- R Code

Coding using R Markdown

To run code inside an R Markdown document, you need to insert a chunk. There are three ways to do so:

1. The keyboard shortcut Cmd/Ctrl + Alt + I
2. The "Insert" button icon in the editor toolbar.
3. By manually typing the chunk delimiters ` ```{r} ` and ` ``` `.

Okay, now let's run some basic chunks. Here's some flavors:

VANILLA: Your run of the mill code chunk. Usually useful

```r
print("What you see is what you get")
```

```
## [1] "What you see is what you get"
```

SHOW BUT DON'T TELL: This option evaluates your code but does not include it.

```
## [1] "Where did this come from?"
```

ALL TALK AND NO WALK: Useful for expository code chunks, i.e. code you want included but not run

```r
print("Doesn't matter what goes here.")
```

WHY IS THIS HERE?: Useful for getting rid of chunks you don't want included.

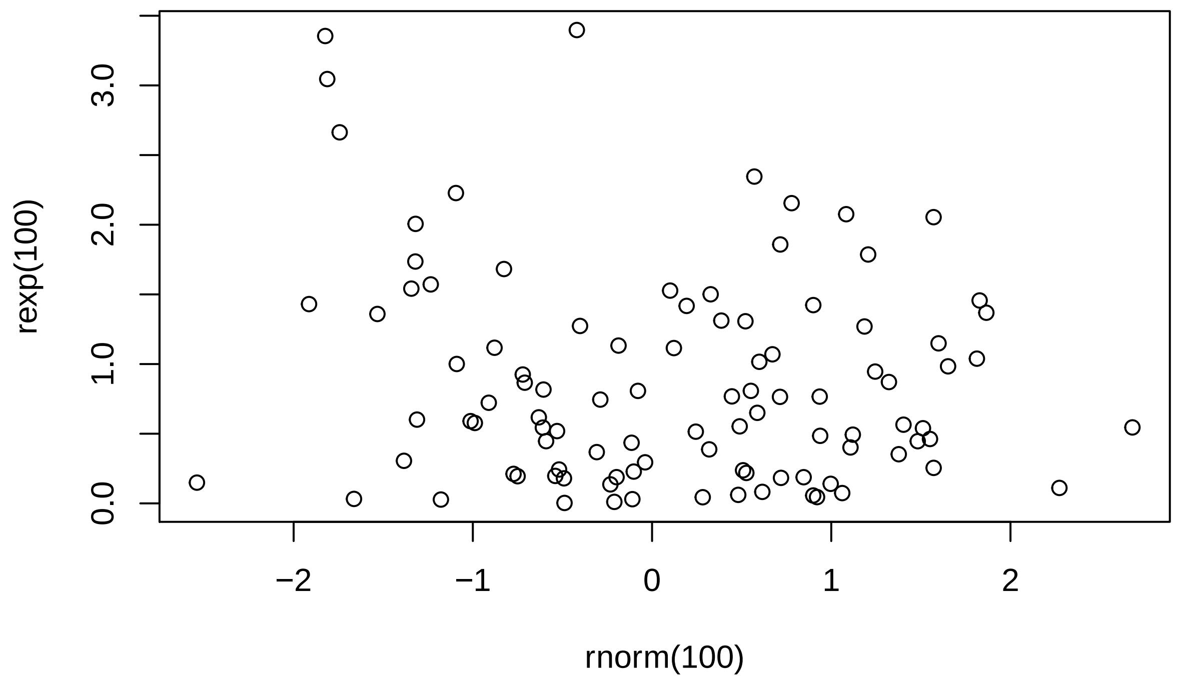
GETTING RID OF LOADING NONSENSE: This gets rid of messages associated with many actions in R, such as loading packages. Useful for truncating output.

```r
library(stats)
```

Pictures

Rmarkdown makes it easy to include plots/figures generated from R code inline. For example:

```r
plot(rnorm(100), rexp(100))
```



You can modify the size of the figures using options in the code chunk:

```
knitr::opts_chunk$set(fig.width=12, fig.height=8)
```

See https://sebastiansauer.github.io/figure_sizing_knitr/ for more information on figure sizing, which can be surprisingly tricky. If you want to include an image from the web, the standard approach is something like:

```
![optional caption text](path/to/img.png)
```

Why won't my document knit?

In order to knit your document, you need to make sure each code chunk produces no errors. This can be frustrating, because it means that one coding mistake can de-rail the entire endeavor. Nevertheless, there is a work around:

```
sqrt("Can't take a square root of a string")
```

```
## Error in sqrt("Can't take a square root of a string"): non-numeric argument to mathematical function
```

Another common source of errors is failing to load the necessary packages in a code chunk. The common symptom of this is that your code runs fine inline, but when knitting, an error of the nature "Error: cannot find function ______" appears. This means you need to include a code chunk loading the packages at the beginning of the document. For example:

```
cars %>% head(10)
```

```
## Error in cars %>% head(10): could not find function "%>%"
```

** An Aside **: Don't install packages in code chunks. Why? Well, you don't want to have to connect to the internet every time you knit, and overlaying installations can create problems when referencing R packages.

Lastly, here is a quick guide to Rmarkdown formatting conventions:

```
Text formatting
------------------------------------------------------------

*italic*  or _italic_
**bold**   __bold__
`code`
superscript^2^ and subscript~2~

Headings
------------------------------------------------------------

# 1st Level Header

## 2nd Level Header

### 3rd Level Header

Lists
------------------------------------------------------------

*   Bulleted list item 1

*   Item 2

    * Item 2a

    * Item 2b
```

```
1.  Numbered list item 1

1.  Item 2. The numbers are incremented automatically in the output.

Links and images
------------------------------------------------------------

<http://example.com>

[linked phrase](http://example.com)

![optional caption text](path/to/img.png)

Tables
------------------------------------------------------------

First Header  | Second Header
------------- | -------------
Content Cell  | Content Cell
Content Cell  | Content Cell
```