RMarkdown - January 2019 Data Carpentry Lesson

Maria Kamenetsky

January 17, 2019

Outline

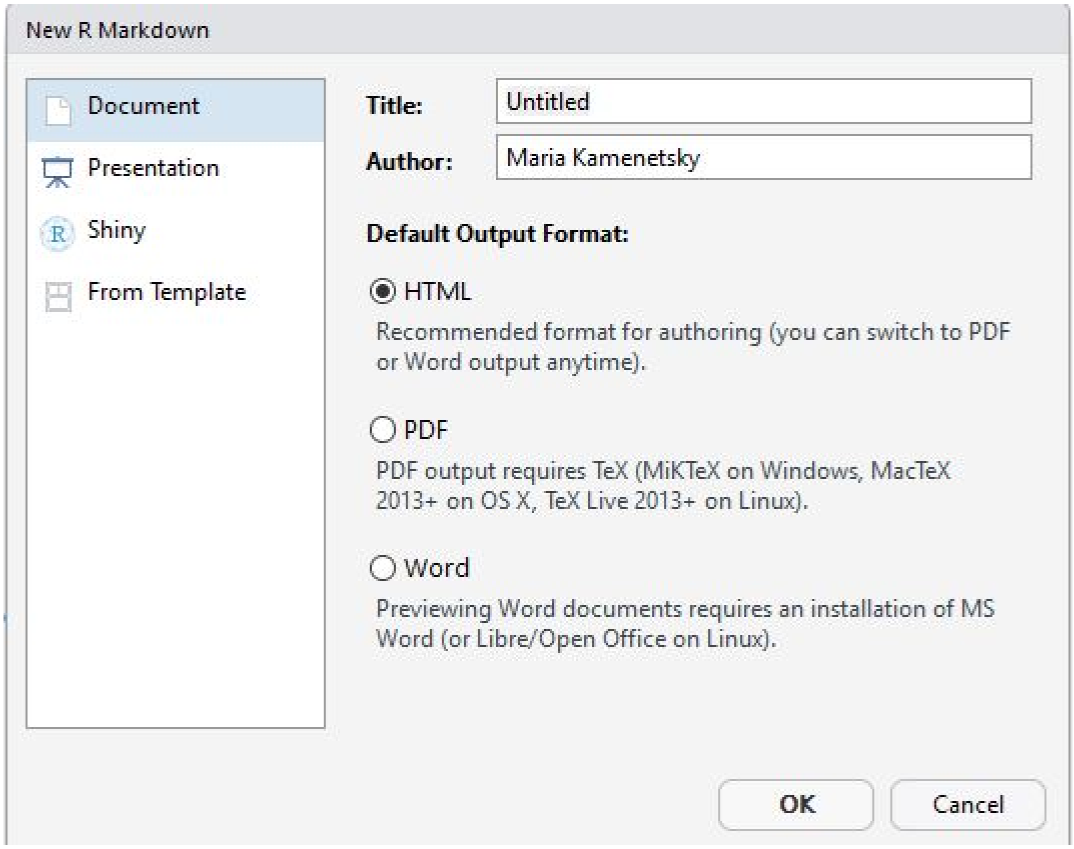
- 0) Why RMarkdown
- 1) Basics of RMarkdown
- 2) Rcode chunks
- 3) Code chunk options
- 4) Inline R code
- 5) Other Output formats

Why `RMarkdown`?

- combines text, code, figures, tables
- write papers without having to remember to swap figures, change cell values in table.
- automatically generate reports
- *reproducible* documents

1) RMarkdown Basics

RStudio -> File -> New File ->R Markdown (keep defaults, add title)



Title: Untitled

Author: Maria Kamenetsky

Default Output Format:

● HTML

Recommended format for authoring (you can switch to PDF or Word output anytime).

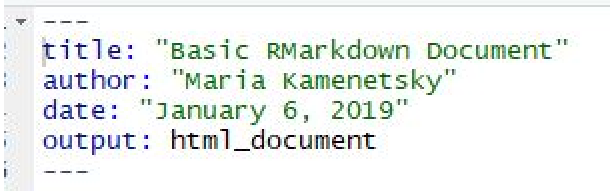
○ PDF

PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

○ Word

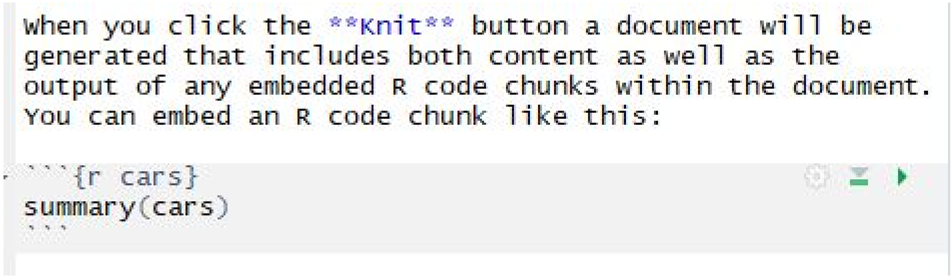
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

Figure 1: Prompt to start new RMarkdown document.



```
---
title: "Basic RMarkdown Document"
author: "Maria Kamenetsky"
date: "January 6, 2019"
output: html_document
---
```

Figure 2: Header for RMarkdown document.



~~~
When you click the **Knit** button a document will be
generated that includes both content as well as the
output of any embedded R code chunks within the document.
You can embed an R code chunk like this:

```{r cars}
summary(cars)
```
~~~

Figure 3: Text and code chunk.

- **Markdown**:
  - system for writing web pages, marking up text (like in an email).

  - Marked up text gets converted to html, with *marks* replaced by proper html code.
- Markdown Basics:
  - **a bolded statement**
  - *an italicized statement*
  - code-type font: `code-type font`
  - this is a bulleted list

1. A numbered
2. List
3. Can be made
4. Like this

```
1. A numbered
1. List
1. Can be made
1. Like this
```

HTML Headers:

```
#Header 1
##Header 2
###Header 3
```

Header 1

Header 2

Header 3

Generate basic RMarkdown document.

CHALLENGE 1: Make a new RMarkdown document. Delete all of the R code chunks and write a bit of Markdown (some sections, italicized/bold text, itemized list). Knit to HTML

More Markdown (if time allows)

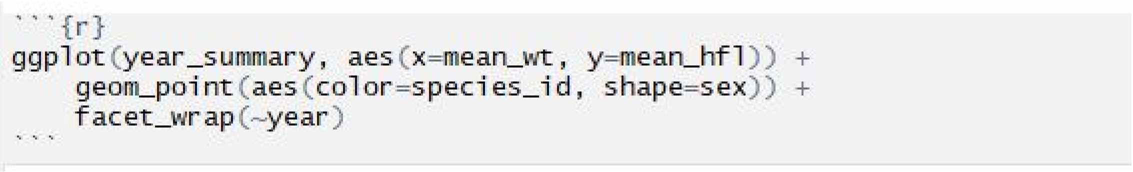
- add a hyperlink: `[text to show](http://the-web-page.com)`
- include an image: `![image caption](http://url/for/file)`
- Sub-script (F$_2$): `F~2~` and super-script (F^2): `F^2^`.
- LaTeX code: ($E = mc^2$) `$E=mc^2$` or formulas:

$$y = \mu + \sum_{i=1}^{p} \beta_i x_i + \epsilon$$

  (`$$y = \mu + \sum_{i=1}^p \beta_i x_i + \epsilon$$`).

2)R code chunks

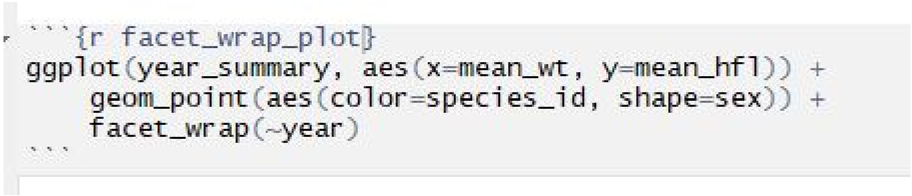
A main code chunk:



~~~
```{r}
ggplot(year_summary, aes(x=mean_wt, y=mean_hfl)) +
    geom_point(aes(color=species_id, shape=sex)) +
    facet_wrap(~year)
```
~~~

Figure 4: Code chunk example

Place R code between the sets of ticks. You may also give each code chunk a name, which can help you find errors:



~~~
```{r facet_wrap_plot}
ggplot(year_summary, aes(x=mean_wt, y=mean_hfl)) +
    geom_point(aes(color=species_id, shape=sex)) +
    facet_wrap(~year)
```
~~~

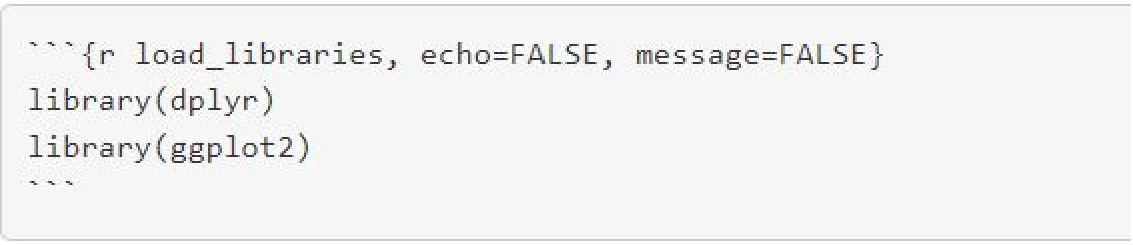
Figure 5: Named code chunk example

Can create a new code chunk manually (with backticks) or short-cut: **CTRL+ALT+i**.

CHALLENGE 2: Add code chunks to load ggplot2 package, read in portal data, create a plot.

3) Code chunk options

- customize R chunk output

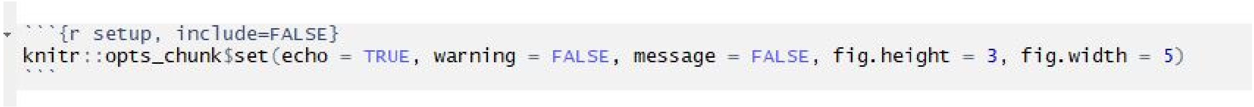


~~~
```{r load_libraries, echo=FALSE, message=FALSE}
library(dplyr)
library(ggplot2)
```
~~~

Figure 6: Example of R code chunk options specifyin that the code should not be output in the final report (echo=FALSE) and any messaged should also be suppressed (message=FALSE)

- Useful code chunk options:
  - `echo=FALSE`: supress code from being printed in final report
  - `results="hide"`: avoid having any results printed.
  - `eval=FALSE`: do not evaulate the code in the chunk.
  - `warning=FALSE` and `message=FALSE` hides any warnings or messages produced.
  - `fig.height`, `fig.width` controls size of figures (in inches).
  - `fig.cap`: adds a caption to the figures.
  - `fig.path`: defines path where figures will be saved. Example: `..., fig.path="Figs/",...`

- code chunk options can be set locally (for each code chunk) or globally (for the entire RMarkdown document)



~~~
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE, warning = FALSE, message = FALSE, fig.height = 3, fig.width = 5)
```
~~~

Figure 7: Global code chunk

CHALLENGE 3: Use chunk options to control the size of a figure and hide the code.

4) Inline R Code

- You can make every number in your report reproducible in the text. Use ‘r and ‘ for an in-line code chunk.
  - For example: ‘r round(some_value, 2)‘. The code will be executed and replaced with the value of the result.
- Don’t let these R chunks split across lines - they will not work.
- If you have some calculations to do, you can have a preceeding R chunk to calculate the results, hide the code and results using `echo=FALSE` and `results="hide"` (which is equivalent to `include=FALSE`).

CHALLENGE 4: Try out in-line code in R

```
surveys <- read.csv("C:/Users/Maria/Desktop/DataCarpentry/Clean/portal_clean.csv")
```

```
There are `r nrow(surveys)` observations in the *surveys* dataset.
```

There are 30652 observations in the *surveys* dataset.

5) Other Output formats

(May take some additional installation on your computer)

In addition to HTML documents, RMarkdown can also *knit* to PDF or Word documents.

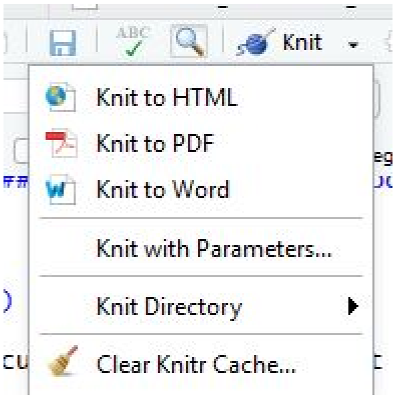


Figure 8: Other output format options

Resources

- Knitr in a knutshell tutorial
- Dynamic Documents with R and knitr (book)
- R Markdown documentation
- R Markdown cheat sheet