Structuring analysis with R Markdown

Daniel Kaplan

dtkaplan

dtkaplan@gmail.com

Embedding R code

“Knitting R into Markdown”

Two ways to run R inside Markdown

- Code chunks
  - *This is the most powerful*
    - provides the most control over computation, and
    - handles plots and other non-text output formats
- Inline code
  - *Simple, but handles only outputs that are easily converted to text, e.g.*
    - character strings (which can include HTML, Markdown, TeX, etc)
    - scalar numbers (not vectors)

Code chunks

- A code chunk is the main vehicle for connecting R code to Markdown
- "Fenced" regions of code with established (almost automatic) connections to Markdown
  - *Default: Rendering the .Rmd file runs each code chunk and embeds the results beneath the code chunk in your final report.*
  - *Chunks have an engine, a (unique) title and parameters for input/output/evaluation*
- Insert new chunks with
  - *the Add Chunk button in the editor toolbar, or*
  - *typing the chunk delimiters ```{r} and ```, or*
  - *the keyboard shortcut* ***Ctrl + Alt/Option + I***

Inline Code

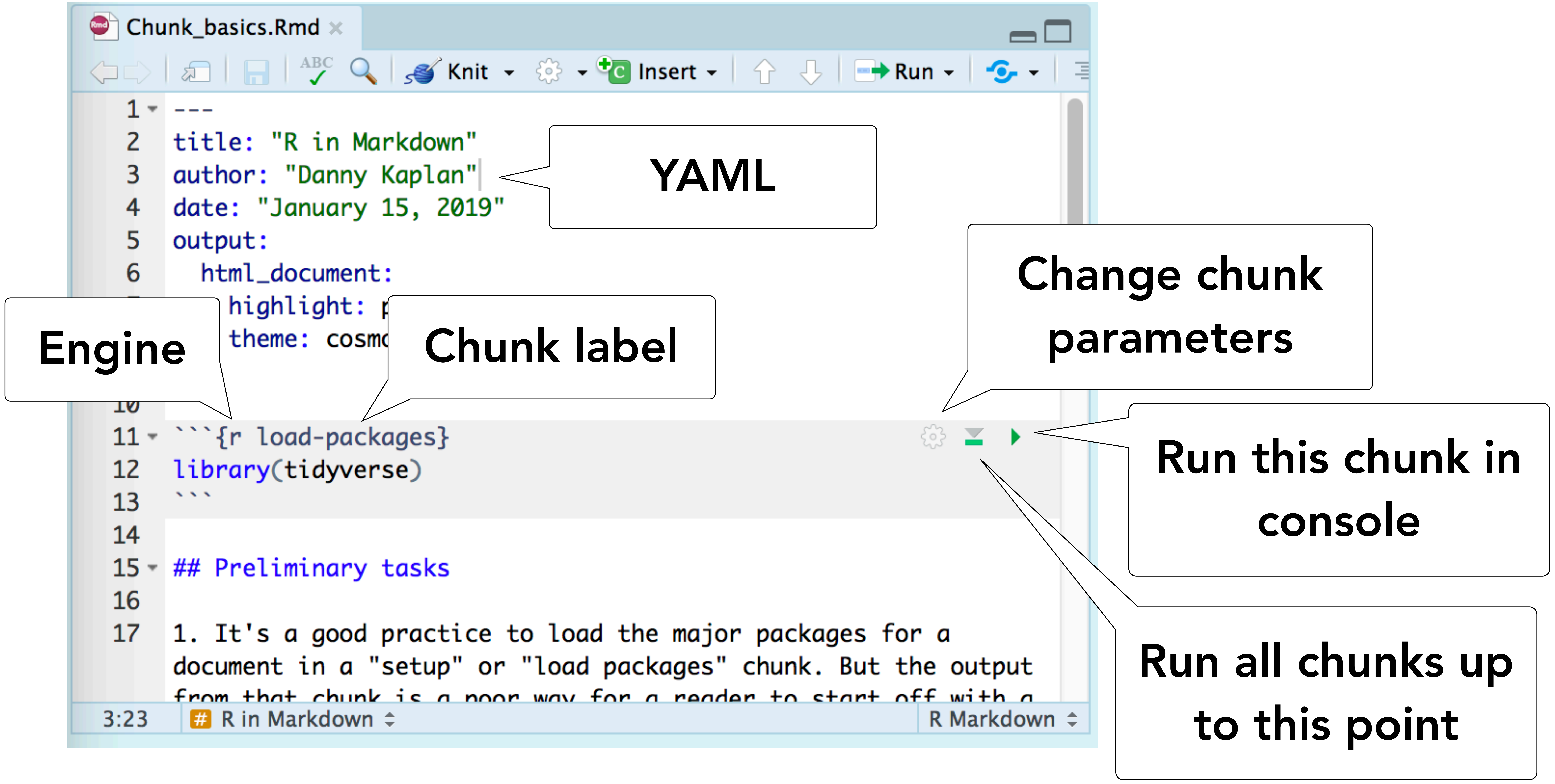
- Can be anywhere in the document … except a chunk.
- Fence is single back-ticks, opening with r-space: `` `r contents` ``
- "Contents" is typically a R expression evaluating to a character string or a number. No vectors.
  - *The character string can be plain text, HTML, Markdown, or TeX*
  - *Formatting reflects the active style at that point in the document.*
- Inline expressions do **not** take knitr options
- Uses current workspace
  - *Values are drawn from the current workspace.*
  - *Assignment is done in the current workspace.*

Your turn

(A) Open the file Chunk_basics.Rmd

(B) Compile it.

- We say “knit,” but better to think of it as a software source file and adopt software-like approaches to handling complexity, e.g.
  - Make small changes and test. Repeat until done.
- After a short tour … follow the instructions in the file



~~~
---
title: "R in Markdown"
author: "Danny Kaplan"
date: "January 15, 2019"
output:
  html_document:
    highlight: p
    theme: cosm

```{r load-packages}
library(tidyverse)
```

## Preliminary tasks

1. It's a good practice to load the major packages for a
document in a "setup" or "load packages" chunk. But the output
from that chunk is a poor way for a reader to start off with a
~~~

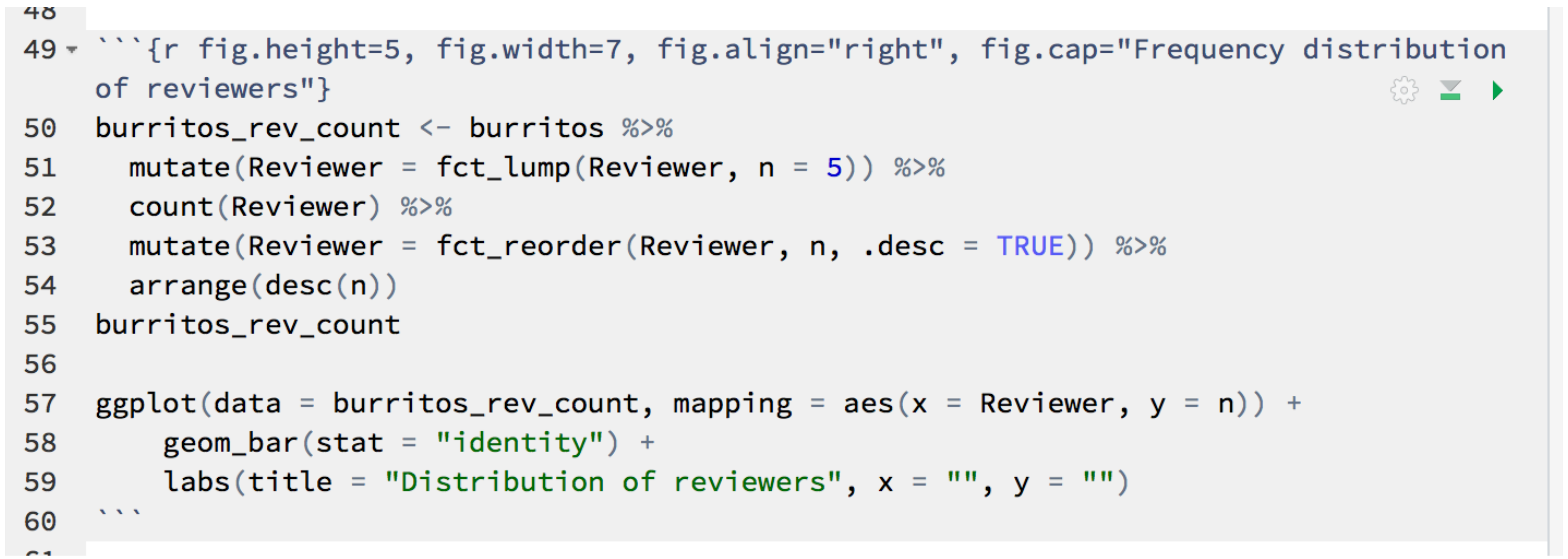
Your turn

(A) Follow the numbered instructions in Chunk_basics.Rmd

5m 00s

Chunk options for figures

- fig.height: Plot canvas height in inches
- fig.width: Plot canvas width in inches
- out.width: Canvas scaling w.r.t. document, e.g. "50%"
- fig.align: "left", "right", or "center"
- fig.cap: Figure caption as character string



~~~
```{r fig.height=5, fig.width=7, fig.align="right", fig.cap="Frequency distribution
of reviewers"}
burritos_rev_count <- burritos %>%
  mutate(Reviewer = fct_lump(Reviewer, n = 5)) %>%
  count(Reviewer) %>%
  mutate(Reviewer = fct_reorder(Reviewer, n, .desc = TRUE)) %>%
  arrange(desc(n))
burritos_rev_count

ggplot(data = burritos_rev_count, mapping = aes(x = Reviewer, y = n)) +
    geom_bar(stat = "identity") +
    labs(title = "Distribution of reviewers", x = "", y = "")
```
~~~

Tips

- Give chunks labels of your own. This helps you troubleshoot and navigate your source document.
- Avoid spaces in chunk labels, even though technically they are "allowed." Especially if you work with GitHub — *more on this later!*
- If you're having a hard time coming up with a short label that describes what the chunk is doing, consider breaking it down into shorter chunks ➡ especially useful for troubleshooting!

What happens when a chunk parameter is malformed? Misnamed?

What happens when there's an error in the R code in a chunk? What feedback/error does R give? (Activate chunk "neg_logs")

What about when there are multiple chunks with errors? (Activate chunk "another_error")

5m 00s

Chunk options

Safe to play with:

- echo = FALSE: code runs, code doesn’t appear in rendered file, results do ➡ for when your audience doesn’t need to see the code
- include = FALSE: code runs but neither code nor results appear in rendered file ➡ results can be used by other chunks
- eval = FALSE: code doesn’t run but they appear in the rendered file ➡ when you want to show/teach code

Requires care:

- message = FALSE: hides messages ➡ especially useful/harmful for loading packages and data
- warning = FALSE: hides messages ➡ especially useful/harmful for functions that throw many warnings

DEMO

bit.ly/shiny-rmd-cloud

bit.ly/shinymark

Your turn

(A) Open sd-burritos.Rmd, knit the document, and view the rendered file to get a sense of the data and the analysis.

(B) Add an inline R chunk to your document. This should be something that results in a numerical value or a character string.

(C) Add another inline R chunk and put some code in here that would result in a plot. What does a plot defined in an inline code chunk look like? Compare your answer with your neighbors'.

(D) **Stretch goal:** Update the date field in the YAML so that the date at the time of knitting the document is printed.

5m 00s

Your turn

(A) Add labels to each chunk.

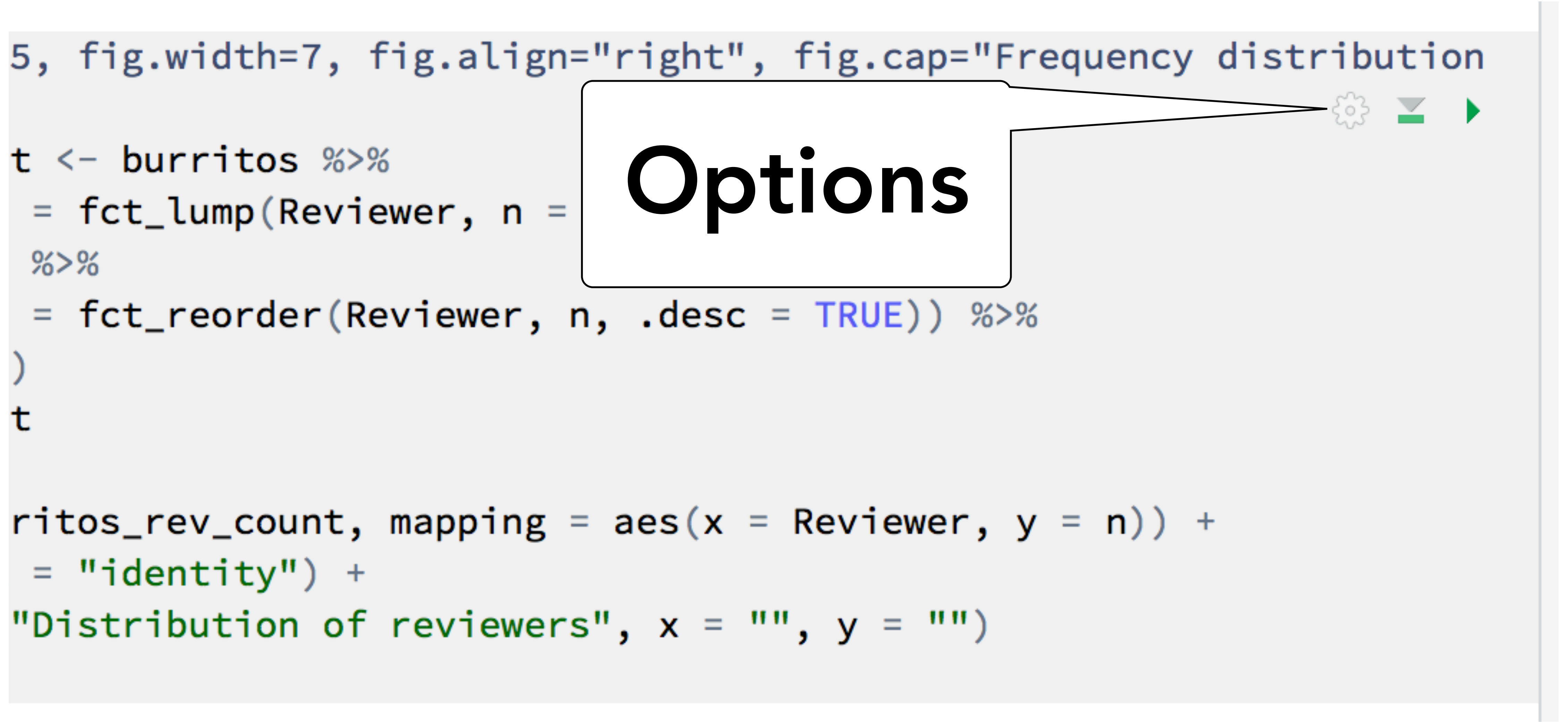
(B) Change the chunk options (echo, eval, include, message, warning) to explore what changes in the output. Then, decide on an appropriate option for each of the chunks. Compare your choices to your neighbors'.

(C) **Stretch goal:** What does the option collapse (set to TRUE or FALSE) do? What is the default setting for this option? Which code chunk(s) does changing this option affect?

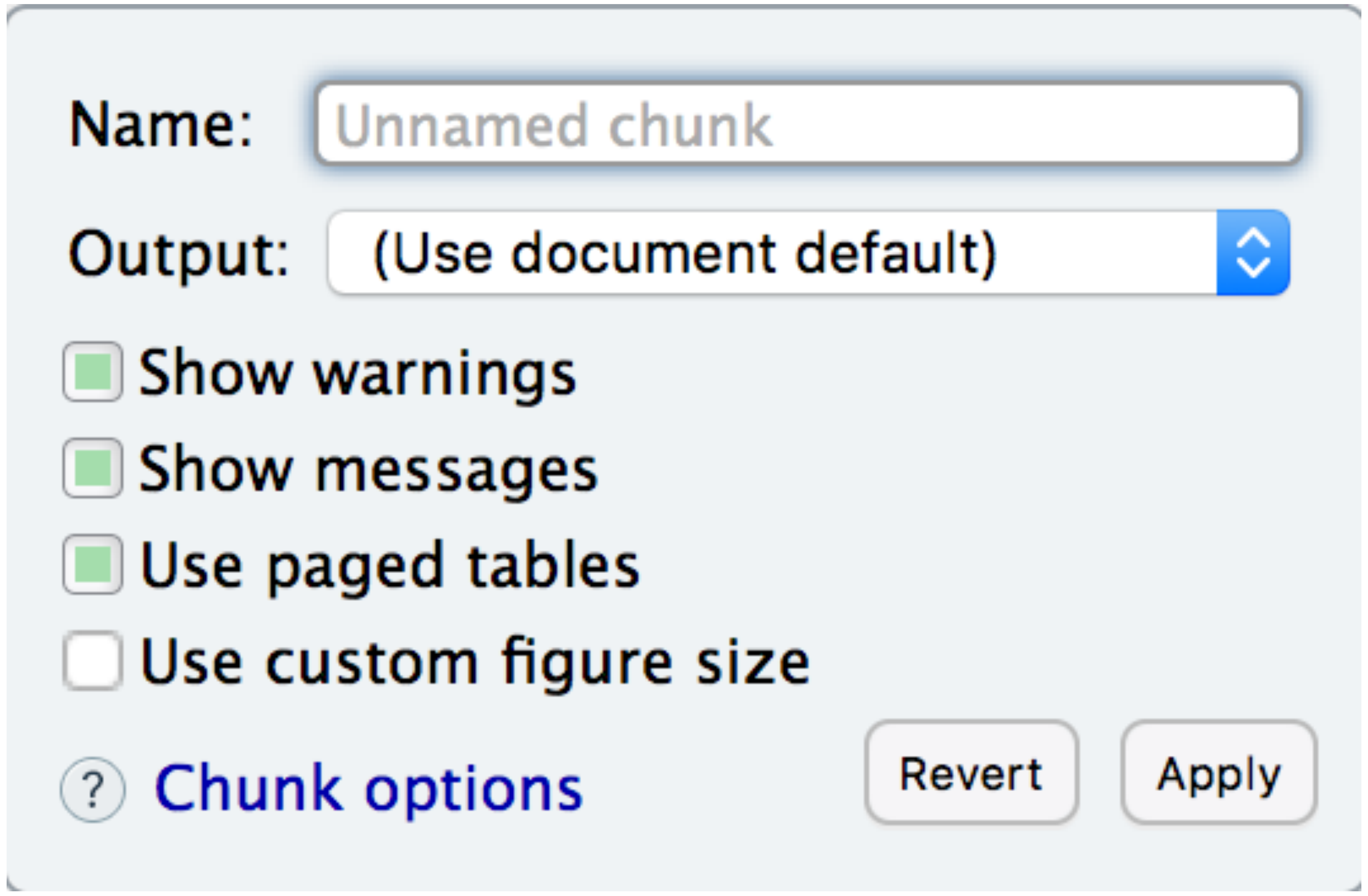
10m 00s

Setting chunk options via GUI

Some of the chunk options can be set via a handy GUI that you can access by clicking on the gear icon on a given chunk.



```
5, fig.width=7, fig.align="right", fig.cap="Frequency distribution

t <- burritos %>%
 = fct_lump(Reviewer, n =
 %>%
 = fct_reorder(Reviewer, n, .desc = TRUE)) %>%
)
t

ritos_rev_count, mapping = aes(x = Reviewer, y = n)) +
 = "identity") +
"Distribution of reviewers", x = "", y = "")
```



Name: Unnamed chunk

Output: (Use document default)

[✓] Show warnings

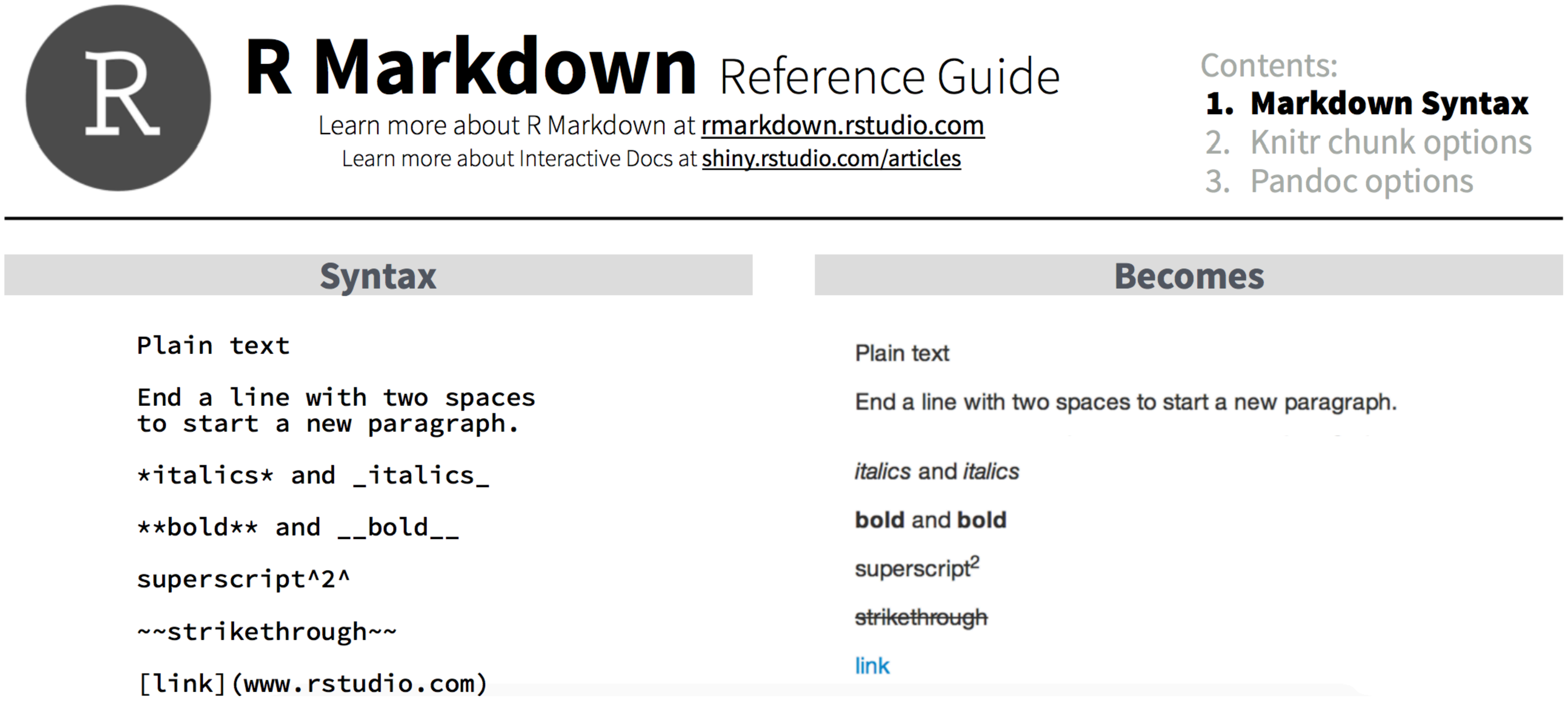
[✓] Show messages

[✓] Use paged tables

[ ] Use custom figure size

So many more chunk options!

https://www.rstudio.com/resources/cheatsheets/



R Markdown Reference Guide

Learn more about R Markdown at rmarkdown.rstudio.com

Learn more about Interactive Docs at shiny.rstudio.com/articles

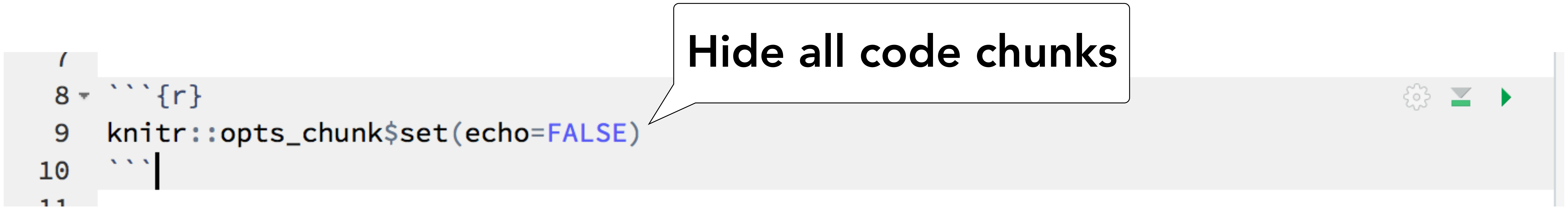
Contents:

1. **Markdown Syntax**
2. Knitr chunk options
3. Pandoc options

Syntax	Becomes
`Plain text`	Plain text
`End a line with two spaces to start a new paragraph.`	End a line with two spaces to start a new paragraph.
`*italics* and _italics_`	*italics* and *italics*
`**bold** and __bold__`	**bold** and **bold**
`superscript^2^`	superscript^2
`~~strikethrough~~`	~~strikethrough~~
`[link](www.rstudio.com)`	link

Global options

- To set global options that apply to every subsequent chunk in your file, call knitr::opts_chunk$set in a code chunk
- Knitr will treat each option that you pass to knitr::opts_chunk$set as a global default that can be overwritten in individual chunk headers



~~~
```{r}
knitr::opts_chunk$set(echo=FALSE)
```
~~~

Your turn

(A) Add a new code chunk to sd-burritos.Rmd and set relevant options for that particular chunk. You create a plot, calculate summary statistics, or if you prefer, just do some basic calculation (without using the data).

(B) Remove the figure height and width options from individual chunks and set them as global options.

5m 00s

Output options

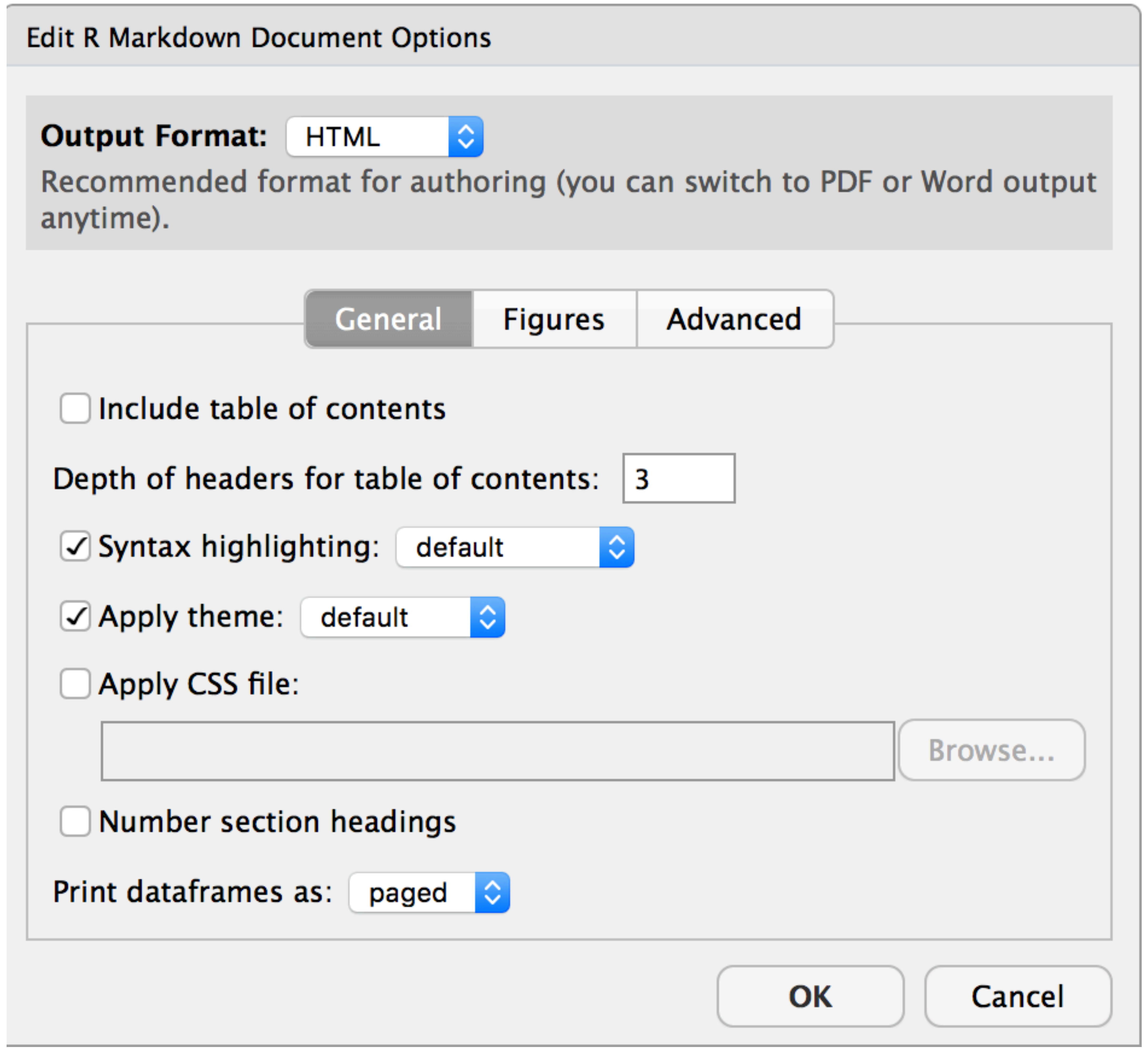
Output options

- Output options are defined in the YAML
- They are similar to setting global knitr options
- To learn which arguments a format takes, read the format's help page in R, e.g. `?html_document`

```
html_notebook(toc = FALSE, toc_depth = 3, toc_float = FALSE,
  number_sections = FALSE, fig_width = 7, fig_height = 5,
  fig_retina = 2, fig_caption = TRUE, code_folding = "show",
  smart = TRUE, theme = "default", highlight = "textmate",
  mathjax = "default", extra_dependencies = NULL, css = NULL,
  includes = NULL, md_extensions = NULL, pandoc_args = NULL,
  output_source = NULL, self_contained = TRUE, ...)
```

Setting output options via GUI

Some of the output options for some of the output types can be set via a handy GUI that you can access by clicking on the gear icon on the toolbar


Edit R Markdown Document Options

Output Format: HTML

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

- [ ] Include table of contents

Depth of headers for table of contents: 3

- [✓] Syntax highlighting: default
- [✓] Apply theme: default
- [ ] Apply CSS file:

- [ ] Number section headings

Print dataframes as: paged

rmarkdown

Your turn

(A) Add a floating table of contents to the html document output.

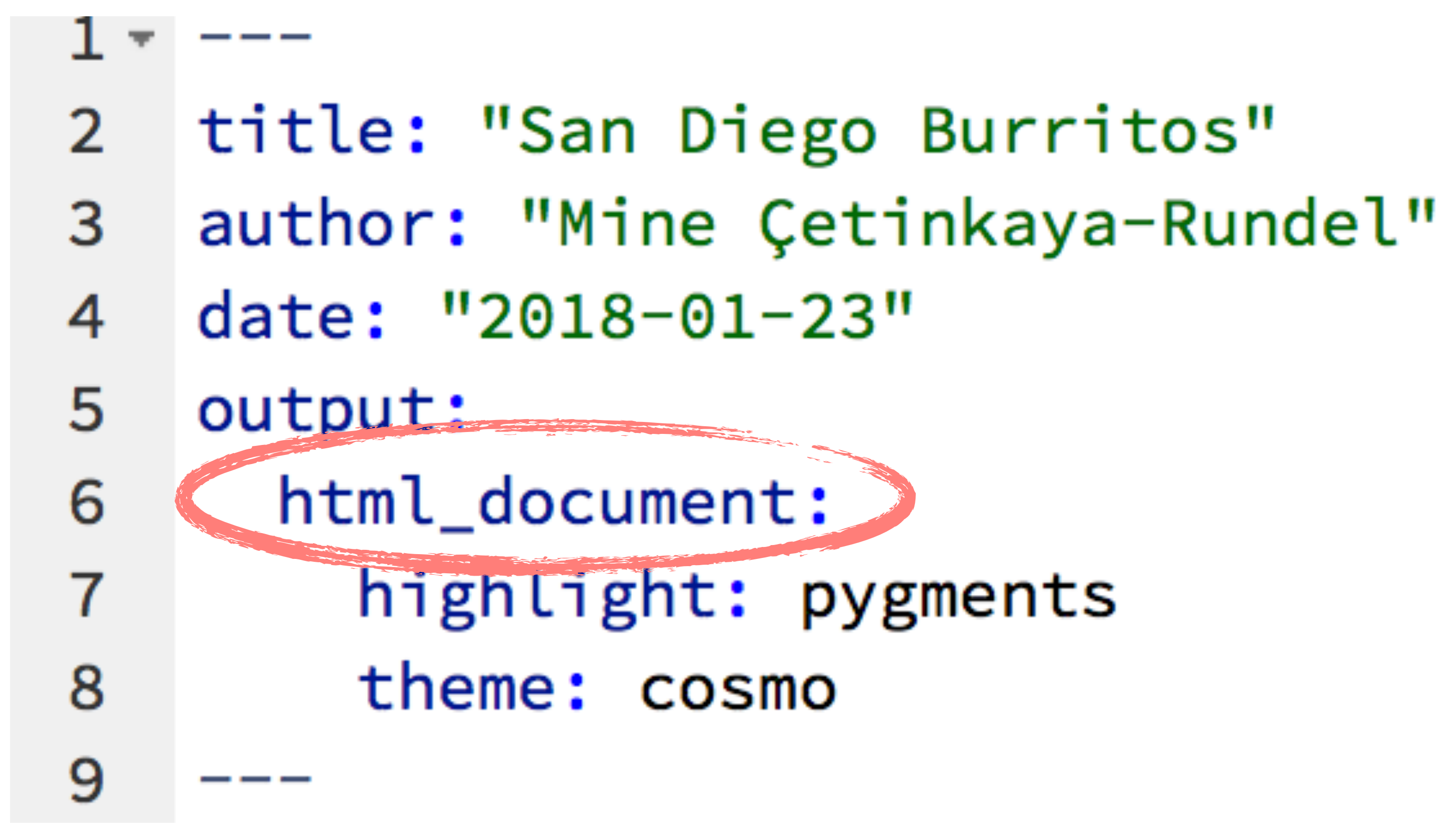
(B) Also set all figures to be 4 x 7.

3m 00s

Output formats

Documents

- Most commonly used output is html_document — HTML document w/ Bootstrap CSS
- Other document options are as follows:
  - *html_notebook* - *Interactive R Notebooks*
  - *pdf_document* - *PDF document (via LaTeX template)*
  - *word_document* - *Microsoft Word document (docx)*
  - *odt_document* - *OpenDocument Text document*
  - *rtf_document* - *Rich Text Format document*
  - *md_document* - *Markdown document (various flavors)*



```yaml
---
title: "San Diego Burritos"
author: "Mine Çetinkaya-Rundel"
date: "2018-01-23"
output:
  html_document:
    highlight: pygments
    theme: cosmo
---
```

Your turn

Convert html_document output to html_notebook. What changed?

Presentations

- ioslides_presentation - HTML presentation with ioslides
- revealjs::revealjs_presentation - HTML presentation with reveal.js
- slidy_presentation - HTML presentation with W3C Slidy
- beamer_presentation - PDF presentation with LaTeX Beamer
- xaringan::moon_reader - remark.js slides

Other output formats

- flexdashboard::flex_dashboard - Interactive dashboards
- tufte::tufte_html - HTML handouts in the style of Edward Tufte
- html_vignette - R package vignette (HTML)
- github_document - GitHub Flavored Markdown document

Your turn

(A) **Before you start:** Delete any files and folders created during the caching exercise. Turn off caching, and remove the Sys.sleep(60) command.

(B) Change output to github_document. Knit the document.

(C) What changed, other than cosmetic changes in the output? Discuss with your neighbors.

(D) **Stretch goal (if you are a git/GitHub user):** Push this document, and any other necessary files, so that it can be previewed on Github (with figures and output).

3m 00s

Caching

> There are only two hard things in Computer Science: cache invalidation and naming things.

–Phil Karlton

Martin Fowler, Two Hard Things. https://martinfowler.com/bliki/TwoHardThings.html.

Caching

➡ If document rendering becomes time consuming due to long computations you can use caching to improve performance

➡ If cache = TRUE is set:

- *Cached chunks are skipped, but objects created in these chunks are (lazy-) loaded from previously saved databases (.rdb and .rdx) files*
- *These files are saved when a chunk is evaluated for the first time, or when cached files are not found*
- *Results of the code will still be included in the output even when cache is used, because knitr also caches the printed output of a code chunk as a character string*

Chunk options for caching

- cache.path: Directory to save cached results in (default = “cache/”)
- dependson: Chunk dependencies for caching (default = NULL)
- …

Your turn

(A) **Before you start:** Make sure all of your code chunks are labeled!

(B) Add the following to the chunk that creates your plot: `Sys.sleep(60)`

(C) Turn on caching for the code chunk that creates the bar plot by setting the relevant chunk option. Knit the document (this is going to take a while, a bit more than 60 seconds). Take a look at the folder where Chunk_basics.Rmd lives. What else is new there?

(D) Knit the document again without making any changes to this particular code chunk. How long did knitting the document take the second time around?

(E) Add a code chunk before this one and slice and overwrite the data so that `mtcars` is now just the first 5 observations in the original dataset. Knit the document again. What is the problem with your plot?

(F) **Stretch goal:** How do you fix this?

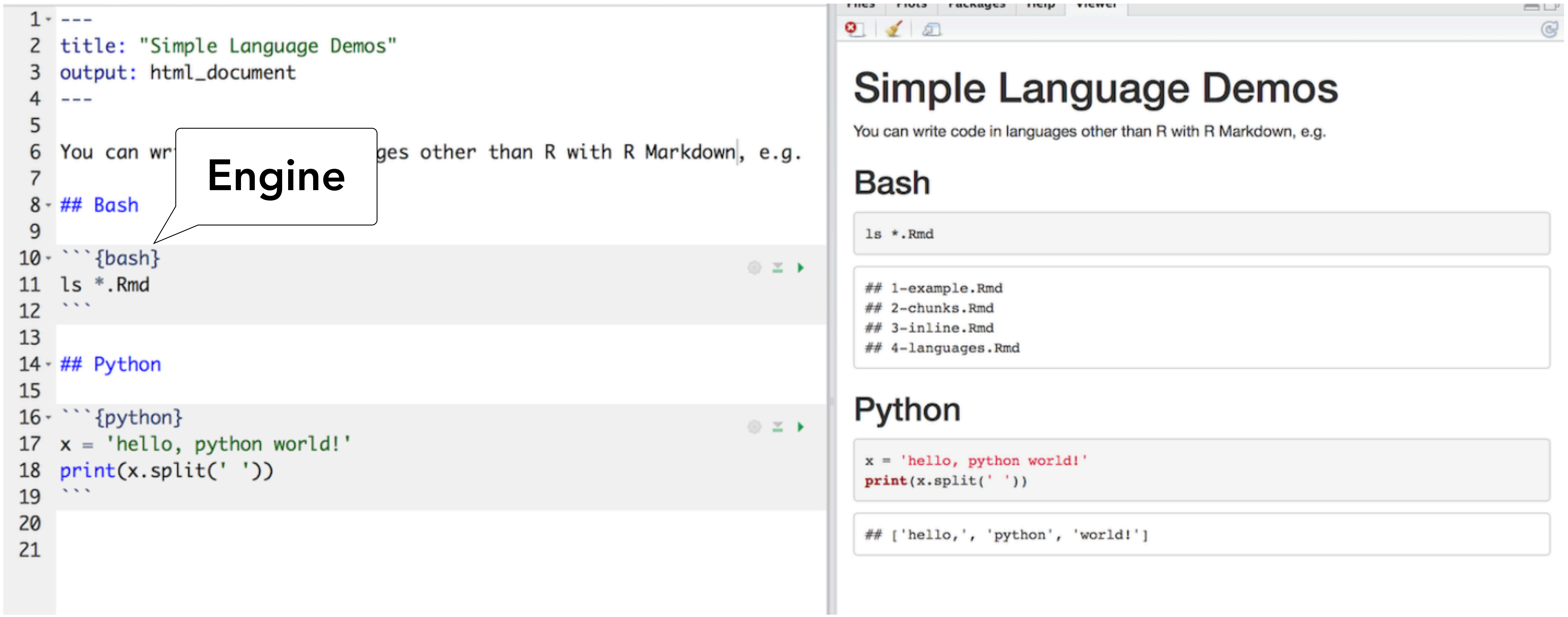
10m 00s

Other languages

Other languages

knitr can execute code in many languages besides R. Some of the available language engines include:

- Python
- SQL
- Bash
- Rcpp
- Stan
- JavaScript
- CSS



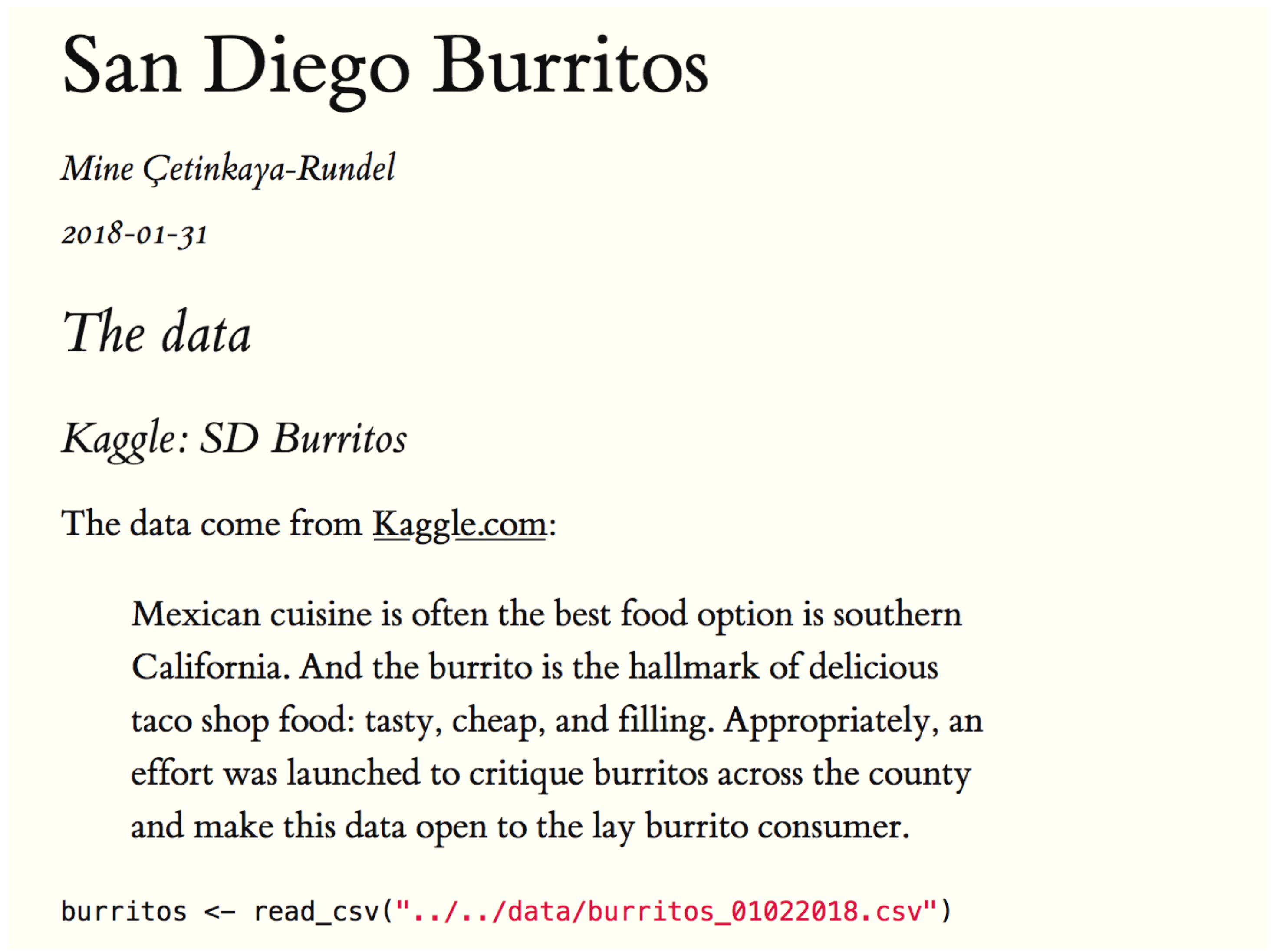
~~~
---
title: "Simple Language Demos"
output: html_document
---

You can wr                    ges other than R with R Markdown, e.g.

## Bash

```{bash}
ls *.Rmd
~~~

Python

```{python}
x = 'hello, python world!'
print(x.split(' '))
```
```

Simple Language Demos

You can write code in languages other than R with R Markdown, e.g.

Bash

```
ls *.Rmd
```

```
## 1-example.Rmd
## 2-chunks.Rmd
## 3-inline.Rmd
## 4-languages.Rmd
```

Python

```
x = 'hello, python world!'
print(x.split(' '))
```

```
## ['hello,', 'python', 'world!']
```
```

# Your turn

- ➡ Decide which you want to work on: **xaringan** slides or **tufte_html**
- ➡ Make sure the package for the one you choose (xaringan or tufte) is installed and loaded
- ➡ Go to New File ➡ R Markdown… ➡ From Template, and then choose either Ninja Presentation (xaringan) or Tufte Handout (tufte)
- ➡ Convert **sd-burritos.Rmd** into one of these format



San Diego Burritos

*Mine Çetinkaya-Rundel*

*2018-01-31*

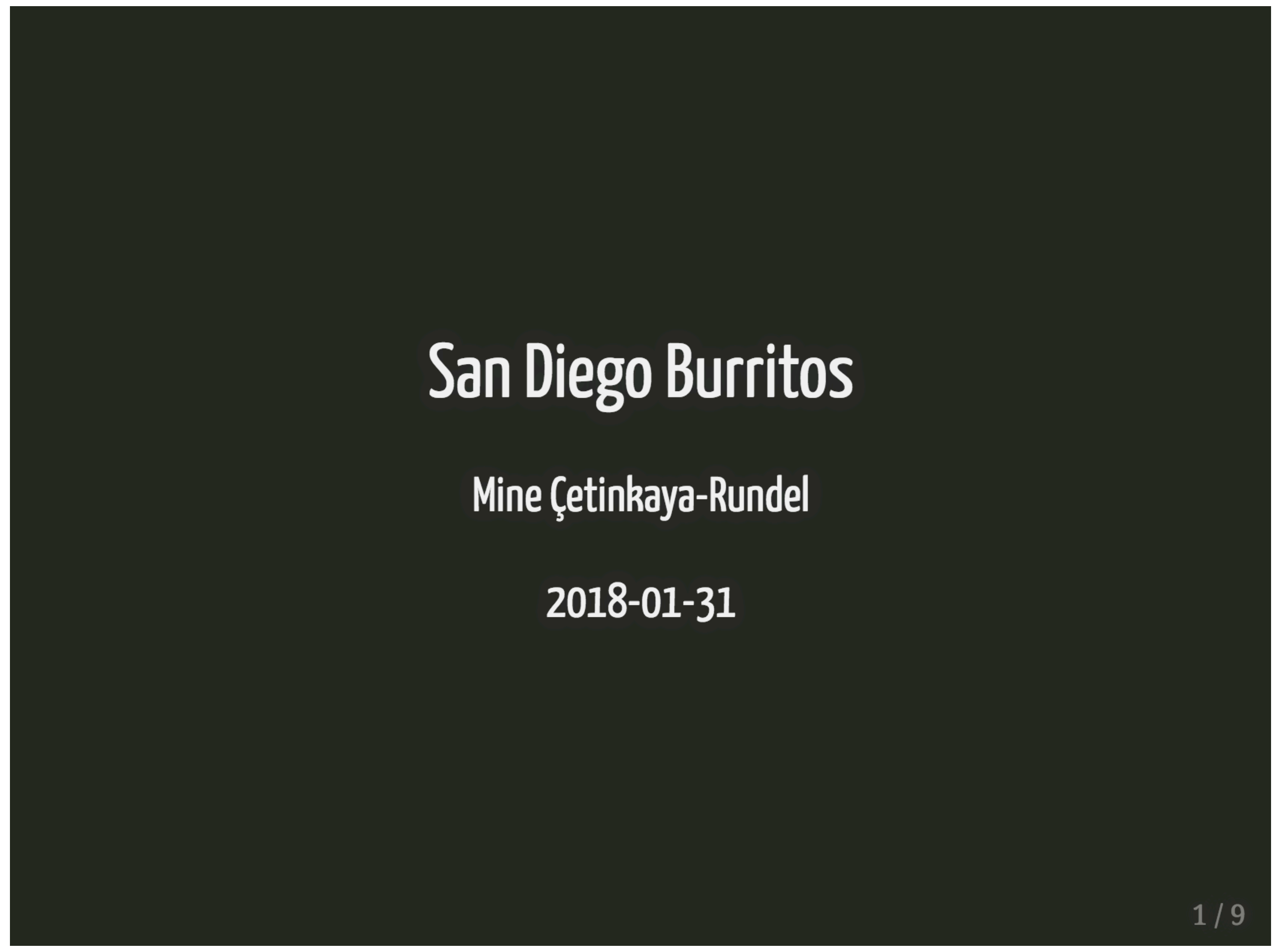
*The data*

*Kaggle: SD Burritos*

The data come from Kaggle.com:

> Mexican cuisine is often the best food option is southern California. And the burrito is the hallmark of delicious taco shop food: tasty, cheap, and filling. Appropriately, an effort was launched to critique burritos across the county and make this data open to the lay burrito consumer.

```
burritos <- read_csv("../../data/burritos_01022018.csv")
```



San Diego Burritos

Mine Çetinkaya-Rundel

2018-01-31

1 / 9

10m 00s