Intro to RMarkdown

The Magic of "knitting"

Thomas Mock | 2019-08-12

Today's Agenda in `rmarkdown`

Basics of `rmarkdown`

Output Formats

- Presentations
  - `Xaringan`
  - Powerpoint
- Websites
  - `Bookdown`
  - `Distill`
- Reports
  - Parameterization
  - `Flexdashboard`


Rmarkdown
TEXT. CODE. OUTPUT.
(GET IT TOGETHER, PEOPLE.)
CODE
#HEADERS
italics
TEXT
PANDOC
KNITR
FAIRY DUST
RECIPE
1. ADD TEXT
2. ADD CODE
3. KNIT
4. (magic)
5. CELEBRATE PERCEIVED WIZARDRY
YAML
rmarkdown
knitr
pandoc
.Rmd
.md
Horst '19

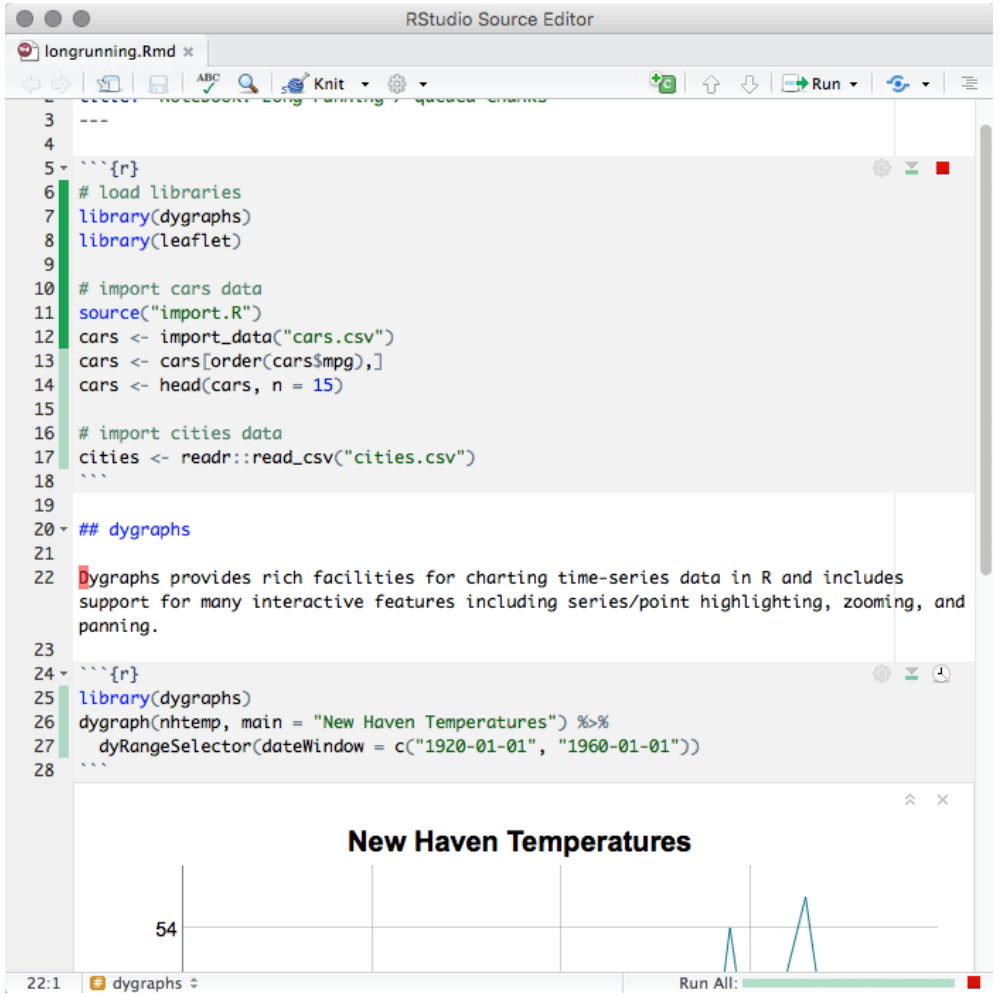
rmarkdown

- Starts as a notebook style interface
    - Mixes `code` with **prose**
- **Knits** to dozens of different formats
    - HTML
    - PDF
    - Handouts
    - Books
    - Reports
    - Dashboards
    - Interactive `shiny` apps
    - Articles
    - Websites

rmarkdown

- Starts as a notebook style interface
  - Mixes code with **prose**
- **Knits** to dozens of different formats
  - HTML
  - PDF
  - Handouts
  - Books
  - Reports
  - dashboards
  - Interactive `shiny` apps
  - Articles
  - Websites

Example



Basic `html_document`

Basic RMarkdown

Thomas Mock

updated: 2019-08-10

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

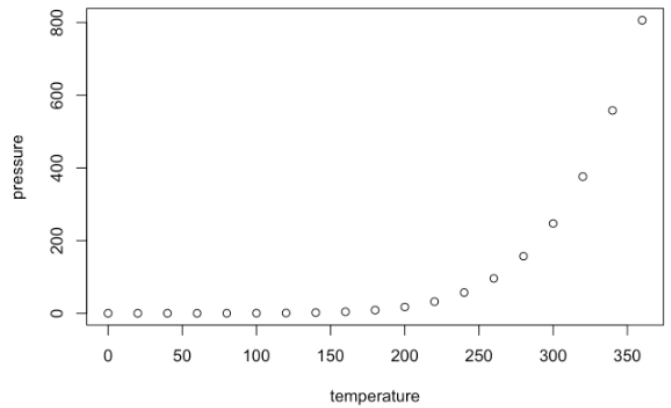
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
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.

Live Demo

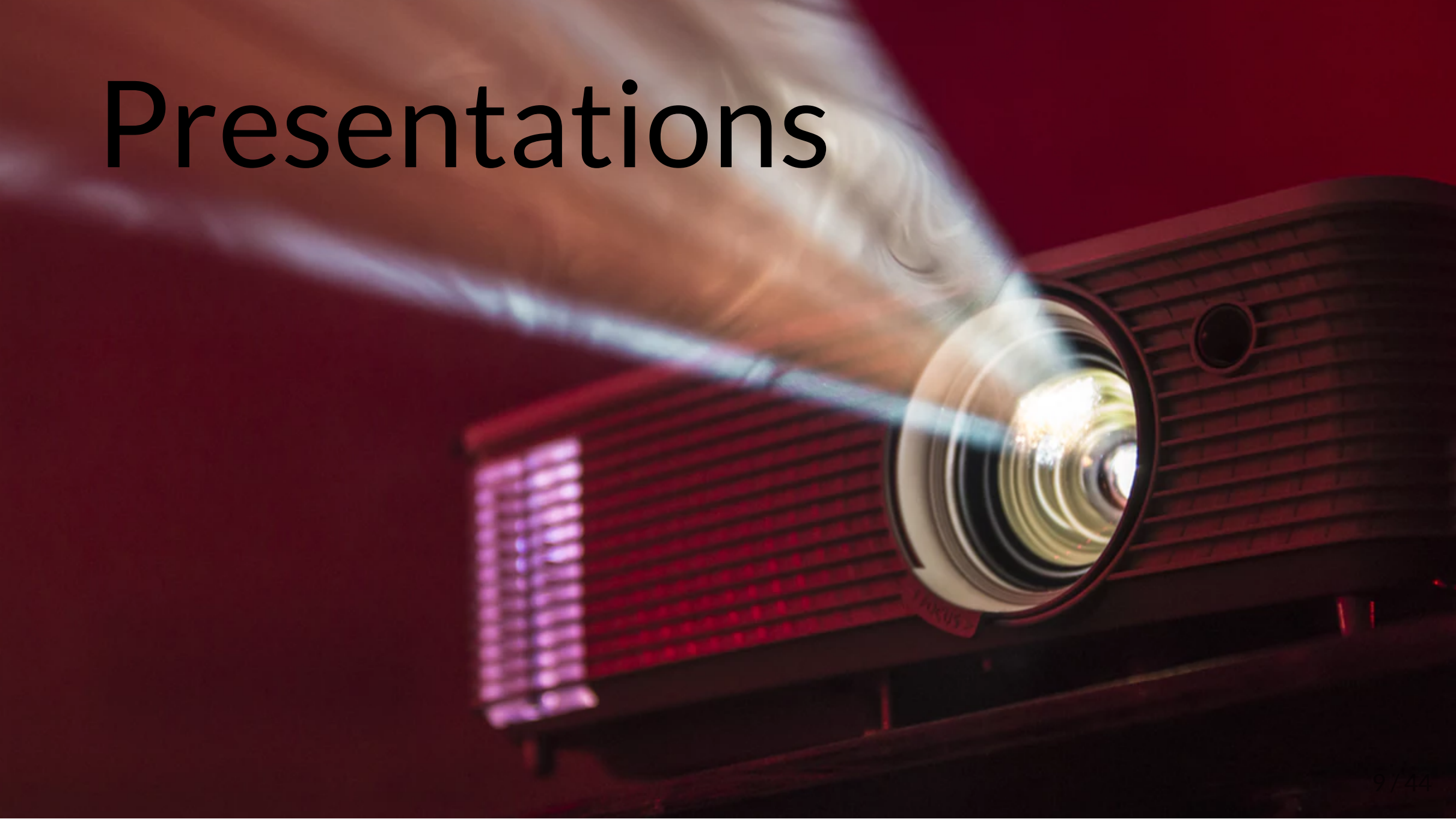
Resources

R Markdown Guide

R Markdown Book

R Markdown Cheatsheet

Presentations



Knit to Powerpoint

The basics start with output.

```
---
title: "My Presentation"
output:
  powerpoint_presentation
---
```

But you can also use a reference presentation for formatting

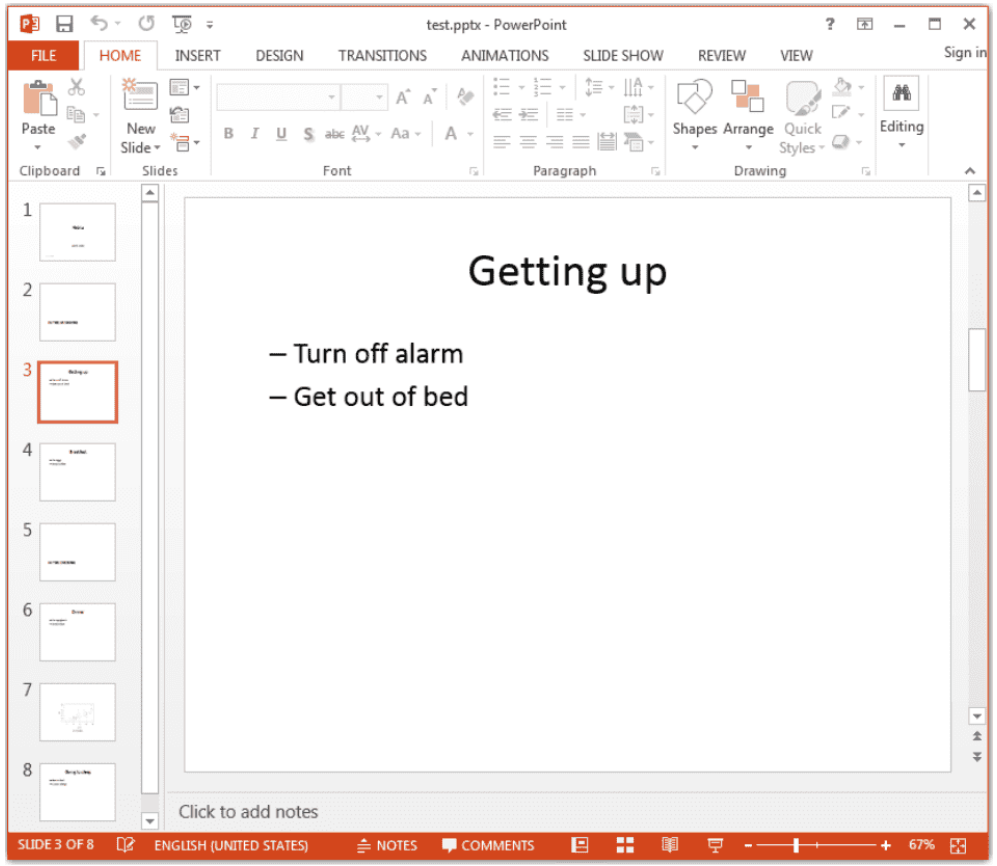
```
---
title: "My Presentation"
output:
  powerpoint_presentation:
    reference_doc: my-styles.pptx
---
```

Example from `rmarkdown` book

Example from RStudio Solutions

Powerpoint Example

```
---
title: "Habits"
author: John Doe
date: March 22, 2005
output: powerpoint_presentation
---

# In the morning

## Getting up

- Turn off alarm
- Get out of bed

## Breakfast

- Eat eggs
- Drink coffee

# In the evening

## Dinner

- Eat spaghetti
- Drink wine

---
```



But there is more to life than Powerpoint!

In fact, this is a presentation.

And THIS presentation was written in `R` code through an `rmarkdown` document.

The code for this presentation is *relatively* simple thanks to the `xaringan` package!

Code to generate previous slide

```
---

### But there is more to life than Powerpoint!

--

In fact, this is a presentation.

--

And THIS presentation was written in `R` code through an `rmarkdown` document.

--

The code for this presentation is *relatively* simple thanks to the `xaringan` package!

<img src="https://user-images.githubusercontent.com/163582/45438104-ea200600-b67b-11e8-80fa-d9f2a99a03b0.png" width=

---
```

xaringan

- Technically, these slides are HTML5
- But we write almost pure RMarkdown
- xaringan does our conversion!
- This concept is the power of RMarkdown
    - Write R code -> knit to ???

xaringan guide

Why write slides with code?

- Quickly reproduce
- Borrow and edit with code
- Generate tables, plots, and words without copy pasting!
- Stay close to the source (code)

Back to `mtcars`

Because `rmarkdown` can execute `R` code, you can generate reports/outputs inside of a presentation

The `mtcars` dataset has 11 columns and 32 rows.

We can also create a table for example via `knitr::kable()`.

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

Previous Slide

````
---

# Back to `mtcars`

Because `rmarkdown` can execute `R` code, you can generate reports/outputs inside of a presentation

--

The `mtcars` dataset has `r ncol(mtcars)` columns and `r nrow(mtcars)` rows.

--

We can also create a table for example via `knitr::kable()`.

--

`r knitr::kable(head(mtcars), 'html')`

---
````

DT package

```
library(DT)
mtcars %>%
  datatable()
```

DT Guide

Show 10 entries Search:

	mpg	cyl	disp	hp	drat	wt	qsec
Mazda RX4	21	6	160	110	3.9	2.62	16.46
Mazda RX4 Wag	21	6	160	110	3.9	2.875	17.02
Datsun 710	22.8	4	108	93	3.85	2.32	18.61
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44
Hornet Sportabout	18.7	8	360	175	3.15	3.44	17.02
Valiant	18.1	6	225	105	2.76	3.46	20.22
Duster 360	14.3	8	360	245	3.21	3.57	15.84
Merc 240D	24.4	4	146.7	62	3.69	3.19	20
Merc 230	22.8	4	140.8	95	3.92	3.15	22.9
Merc 280	19.2	6	167.6	123	3.92	3.44	18.3

Showing 1 to 10 of 32 entries Previous 1 2 3 4 Next

gt package

```
library(gt)
head(mtcars) %>%
  gt()
```

gt Guide

mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

gt does fancier tables too

```
## # A tibble: 6 x 6
##   date        open  high   low close     volume
##   <date>     <dbl> <dbl> <dbl> <dbl>      <dbl>
## 1 2010-06-14 1095  1106. 1089. 1090. 4425830000
## 2 2010-06-11 1083. 1092. 1077. 1092. 4059280000
## 3 2010-06-10 1059. 1088. 1059. 1087. 5144780000
## 4 2010-06-09 1063. 1078. 1052. 1056. 5983200000
## 5 2010-06-08 1051. 1063. 1042. 1062  6192750000
## 6 2010-06-07 1066. 1071. 1050. 1050. 5467560000
```

gt does fancier tables too

```r
# Define the start and end dates for the data range
start_date <- "2010-06-07"
end_date <- "2010-06-14"

# Create a gt table based on preprocessed
# `sp500` table data
sp500 %>%
  dplyr::filter(date >= start_date & date <= end_date) %>%
  dplyr::select(-adj_close) %>%
  gt() %>%
  tab_header(
    title = "S&P 500",
    subtitle = glue::glue("{start_date} to {end_date}")
  ) %>%
  fmt_date(
    columns = vars(date),
    date_style = 3
  ) %>%
  fmt_currency(
    columns = vars(open, high, low, close),
    currency = "USD"
  ) %>%
  fmt_number(
    columns = vars(volume),
    suffixing = TRUE
  )
```

gt does fancier tables too

S&P 500					
2010-06-07 to 2010-06-14					
date	open	high	low	close	volume
Mon, Jun 14, 2010	$1,095.00	$1,105.91	$1,089.03	$1,089.63	4.43B
Fri, Jun 11, 2010	$1,082.65	$1,092.25	$1,077.12	$1,091.60	4.06B
Thu, Jun 10, 2010	$1,058.77	$1,087.85	$1,058.77	$1,086.84	5.14B
Wed, Jun 9, 2010	$1,062.75	$1,077.74	$1,052.25	$1,055.69	5.98B
Tue, Jun 8, 2010	$1,050.81	$1,063.15	$1,042.17	$1,062.00	6.19B
Mon, Jun 7, 2010	$1,065.84	$1,071.36	$1,049.86	$1,050.47	5.47B

Fuel report

Fuel efficiency is declining with increased displacement

- The larger engine displacement accounts for 72% of the decline in fuel efficiency

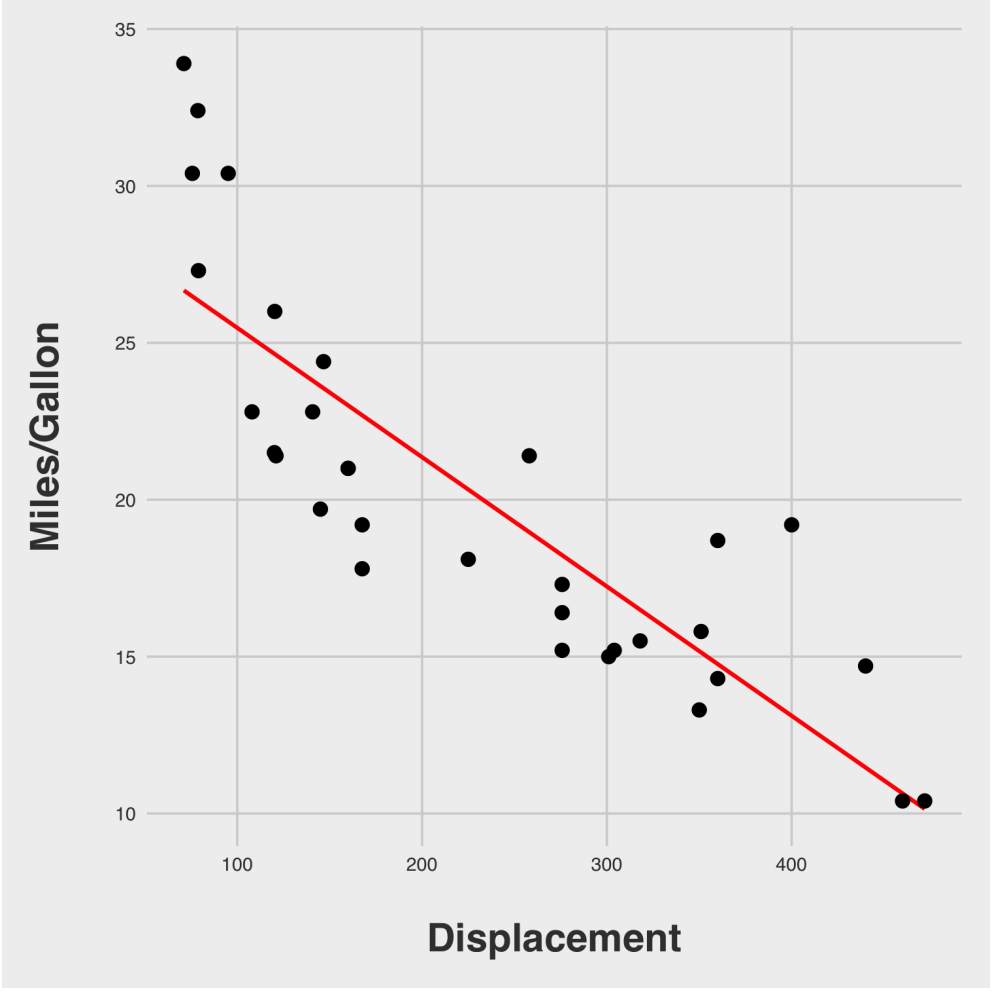
Show 6 entries Search:

	car	disp	mpg
1	Mazda RX4	160	21
2	Mazda RX4 Wag	160	21
3	Hornet 4 Drive	258	21.4
4	Valiant	225	18.1
5	Duster 360	360	14.3
6	Merc 240D	146.7	24.4

Showing 1 to 6 of 20 entries

Previous 1 2 3 4 Next

DISP vs MPG (R^2: 0.72)

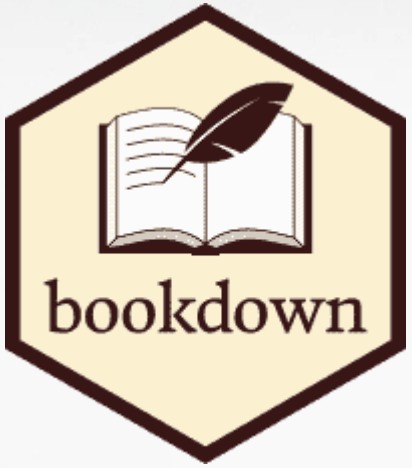


Previous Slide

```
r_2 <- lm(mpg~disp, mtcars) %>%
  summary() %>%
  .$r.squared %>%
  round(2)

Fuel efficiency is declining with increased displaceme
- The larger engine displacement accounts for `r r_2 *

library(DT)
mtcars %>%
  rownames_to_column(var = "car") %>%
  group_by(cyl) %>%
  top_n(6) %>%
  ungroup() %>%
  select(car, disp, mpg) %>%
  datatable(
    options = list(
  pageLength = 6
    )
  )
```

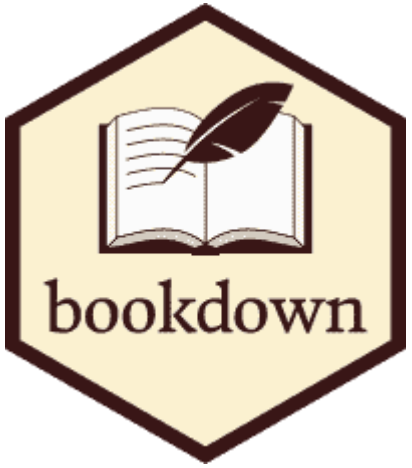
```
### DISP vs MPG (R<sup>2</sup>: `r r_2`)

mtcars %>%
  ggplot(aes(x = disp, y = mpg)) +
  geom_smooth(method = "lm", se = F, color = "red") +
  geom_point(size = 3) +
  ggthemes::theme_fivethirtyeight() +
  labs(x = "\nDisplacement",
       y = "Miles/Gallon\n") +
  theme(axis.title = element_text(face = "bold", size
```

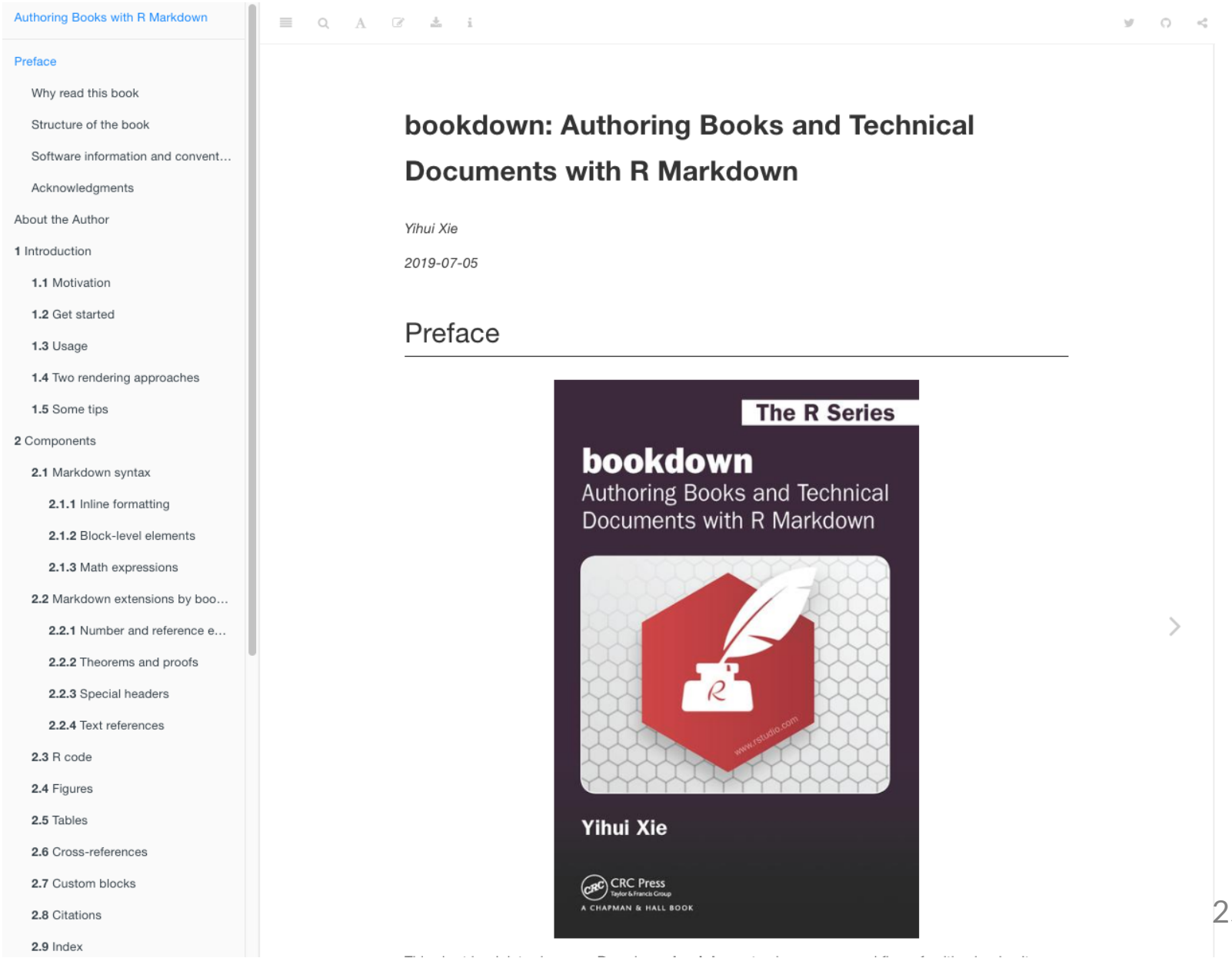
bookdown



bookdown

Best resource is at the published bookdown *bookdown*. 😉





Use cases for writing books in `rmarkdown`

- Writing books for external consumption
    - Examples of many published books at the `bookdown` website
- Writing books for internal consumption
    - Example of the `tidyverse` style guide
- Alternative example of the BBC R Graphics cookbook
    - Created with `cosmo` theme of `rmarkdown` (not `bookdown`)
    - Just has a table of contents (floating on left)
    - See their GitHub for all code to create

distill

Website

Use Cases

- Website
- Blogs
- Technical docs

Examples

- RStudio Environments
- My personal blog

Reports

Reports

Pain Point

- Writing reports is:
  - Manual
  - Tedious
  - Slow 🐢

Solution

Paramaters in `rmarkdown`!

- Generate the *same* report BUT with *new data*
- All the code is identical
- The end result is according to the input data

Situation: FAA data analyst

- My boss asks for a report on how many animals were hit by planes in TX last year
  - Goal: Create a report for TX in 2018
  - Result: I get the data, create a graph, paste into doc and send along 😎
- My boss emails me later that day
  - I need the report for 2015, 2016, and 2017 ASAP 😠
  - Goal: Create 3 new reports
  - Result: I go back and get the data again, create 3x graphs, paste into 3 new docs, and send along 😫

Alternative

- Write a paramaterized `rmarkdown` report
  - Write the code 1x
  - Execute `n` times! 🤓

Knit a Report

Render reports programatically

```
rmarkdown::render("param-report-programmatic.rmd",
                  params = list(
                    states = "TX",
                    years = "2018"
                  ))
```

Total Animal Impacts Report

Thomas Mock

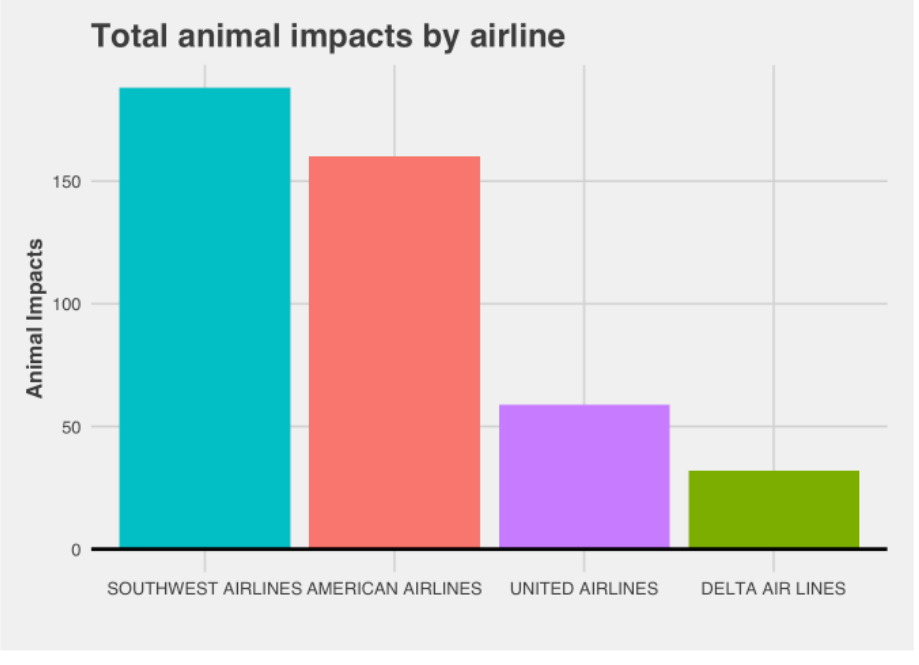
2019-08-11

What state?

The state is TX

What year?

The year is 2018



custom render function

```
render_impact_report <- function(states, years){
  rmarkdown::render(
    "param-report-programmatic.rmd",
    params = list(
      states = states,
      years = years
    ),
    output_file =
      glue::glue("animal-impact-{states}-{years}.html"
  )
}
```

one-off report

```
render_impact_report(states = "TX", years = 2015)
```

Need to tell `render` function

- which file to use (`param-report-programmatic.rmd`)
- what parameters (`params`)
- output_file
  - Using `glue()` function to generate meaningful file names
  - `glue()` pastes in variables surrounded by `{}`

```
states <- "TX"
years <- 2018

glue::glue("animal-impact-{states}-{years}.html")
```

```
## animal-impact-TX-2018.html
```

What about many reports? Say Hi to purrr

New `glue()` function

```
output_state_yr <- function(states, years){
  glue::glue("The state is {states} and year is {years}")
}

output_state_yr("TX", 2019)
```

```
## The state is TX and year is 2019
```

Apply `output_state_yr` across each of the states/years combos

```
states <- rep("TX", 3)
years <- 2016:2018

purrr::pmap(.l = list(states, years), .f = output_state_yr)
```

```
## [[1]]
## The state is TX and year is 2016
##
## [[2]]
## The state is TX and year is 2017
##
## [[3]]
## The state is TX and year is 2018
```

Back to our custom render function

```
render_impact_report <- function(states, years){
  rmarkdown::render(
    "param-report-programmatic.rmd",
    params = list(
      states = states,
      years = years
    ),
    output_file =
      glue::glue("animal-impact-{states}-{years}.html")
  )
}
```

- We'll iterate across each combination of state + year
- Each combo will generate a report with a custom file name based on the input!
- This process scales from **1 report** to **100s of reports**

Many Reports

Need to specify what states and what years are of interest

```
states <- rep("TX", 4)
years <- 2015:2018

states
```

```
## [1] "TX" "TX" "TX" "TX"
```

```
years
```

```
## [1] 2015 2016 2017 2018
```

Apply the function to each element of the vector

We can use `purrr` to apply function across the years (for loop is also fine, I just find `purrr` more readable)

```
purrr::pmap(.l = list(states, years), .f = render_impact_report)
```

Altogether now

Write the function

```
render_impact_report <- function(states, years){
  rmarkdown::render(
    "param-report-programmatic.rmd",
    params = list(
      states = states,
      years = years
    ),
    output_file =
      glue::glue("animal-impact-{states}-{years}.html")
  )
}
```

Then specify the paramater inputs

```
states <- rep("TX", 4)
years <- 2015:2018
```

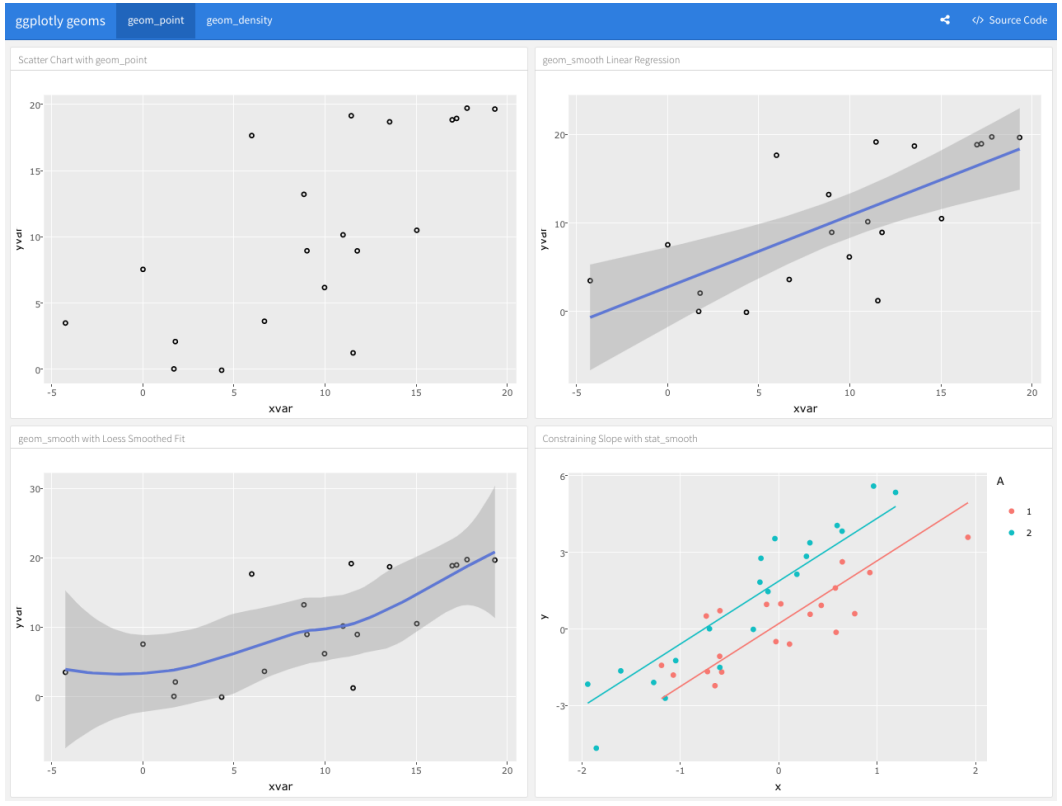
Then `knit`/`render` the 4x reports!

```
purrr::pmap(.l = list(states, years), .f = render_impact_report)
```

Dashboards

Enter `flexdashboard`

If you can create a `ggplot` you can create a `flexdashboard`!



`flexdashboard` Site

Column-based

```
---
title: "Column Orientation"
output: flexdashboard::flex_dashboard
---

Column
-------------------------------------

### Chart 1


Column
-------------------------------------

### Chart 2


### Chart 3
```

Column-output

Row-based

```
---
title: "Row Orientation"
output:
  flexdashboard::flex_dashboard:
    orientation: rows
---

Row
-------------------------------------

### Chart 1



Row
-------------------------------------

### Chart 2



### Chart 3
```

Row-output

Live Demo

Resources

The best single source is `R Markdown`: The Definitive Guide, available online or in text

- `rmarkdown` website
- Paramaterized Reports
- `xaringan` presentations
- Powerpoint Presentations
- `flexdashboard`
- `bookdown`
- `distill`
- HTML Documents

Open RStudio Cloud


END