An Overview of R Markdown

NASSAU

MATT HEAPHY, FSA, MAAA
VP and Actuary, Data Analytics
November 12, 2020

ACHS
ACTUARIES' CLUB
OF HARTFORD & SPRINGFIELD

ACTUARIES' CLUB OF
BOSTON

SOCIETY OF ACTUARIES
Antitrust Compliance Guidelines

Active participation in the Society of Actuaries is an important aspect of membership. While the positive contributions of professional societies and associations are well-recognized and encouraged, association activities are vulnerable to close antitrust scrutiny. By their very nature, associations bring together industry competitors and other market participants.

The United States antitrust laws aim to protect consumers by preserving the free economy and prohibiting anti-competitive business practices; they promote competition. There are both state and federal antitrust laws, although state antitrust laws closely follow federal law. The Sherman Act, is the primary U.S. antitrust law pertaining to association activities. The Sherman Act prohibits every contract, combination or conspiracy that places an unreasonable restraint on trade. There are, however, some activities that are illegal under all circumstances, such as price fixing, market allocation and collusive bidding.

There is no safe harbor under the antitrust law for professional association activities. Therefore, association meeting participants should refrain from discussing any activity that could potentially be construed as having an anti-competitive effect. Discussions relating to product or service pricing, market allocations, membership restrictions, product standardization or other conditions on trade could arguably be perceived as a restraint on trade and may expose the SOA and its members to antitrust enforcement procedures.

While participating in all SOA in person meetings, webinars, teleconferences or side discussions, you should avoid discussing competitively sensitive information with competitors and follow these guidelines:

- **Do not** discuss prices for services or products or anything else that might affect prices
- **Do not** discuss what you or other entities plan to do in a particular geographic or product markets or with particular customers.
- **Do not** speak on behalf of the SOA or any of its committees unless specifically authorized to do so.
- **Do** leave a meeting where any anticompetitive pricing or market allocation discussion occurs.
- **Do** alert SOA staff and/or legal counsel to any concerning discussions
- **Do** consult with legal counsel before raising any matter or making a statement that may involve competitively sensitive information.

Adherence to these guidelines involves not only avoidance of antitrust violations, but avoidance of behavior which might be so construed. These guidelines only provide an overview of prohibited activities. SOA legal counsel reviews meeting agenda and materials as deemed appropriate and any discussion that departs from the formal agenda should be scrutinized carefully. Antitrust compliance is everyone’s responsibility; however, please seek legal counsel if you have any questions or concerns.

Presentation Disclaimer

rmarkdown

www.rstudio.com

Why is R Markdown important?

Transparency

Reproducibility

ASOP 41 - Actuarial Communications, Section 3.2:

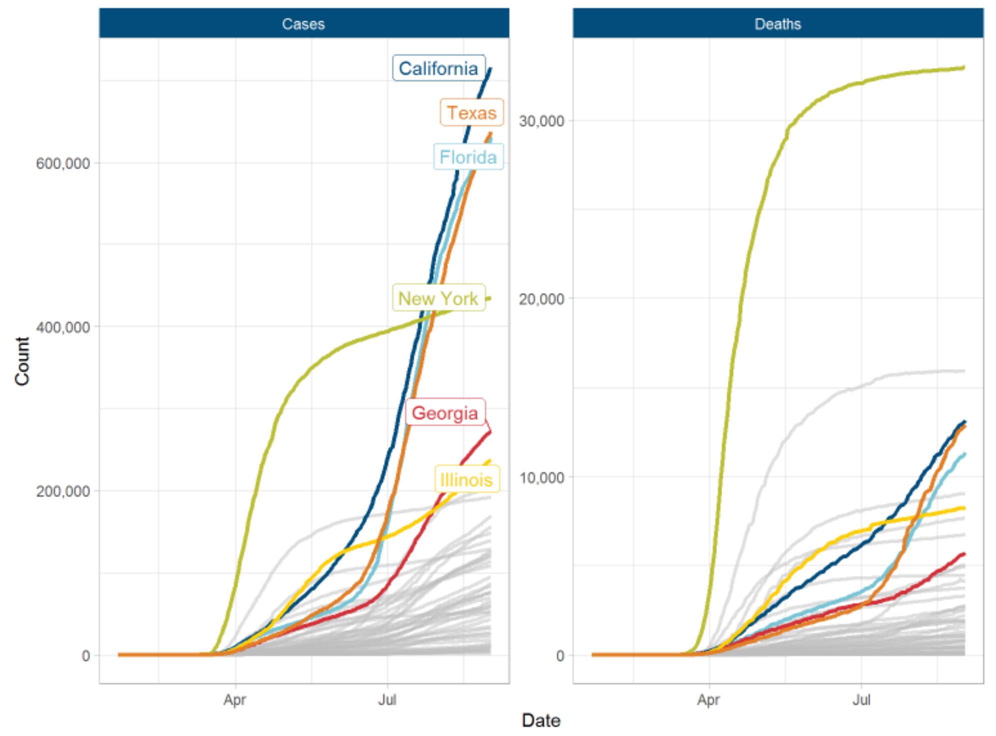
> *"...the actuary should state the actuarial findings, and identify the methods, procedures, assumptions, and data used by the actuary with sufficient clarity that another actuary qualified in the same practice area could make an objective appraisal of the reasonableness of the actuary's work as presented in the actuarial report"*

Example COVID-19 Report

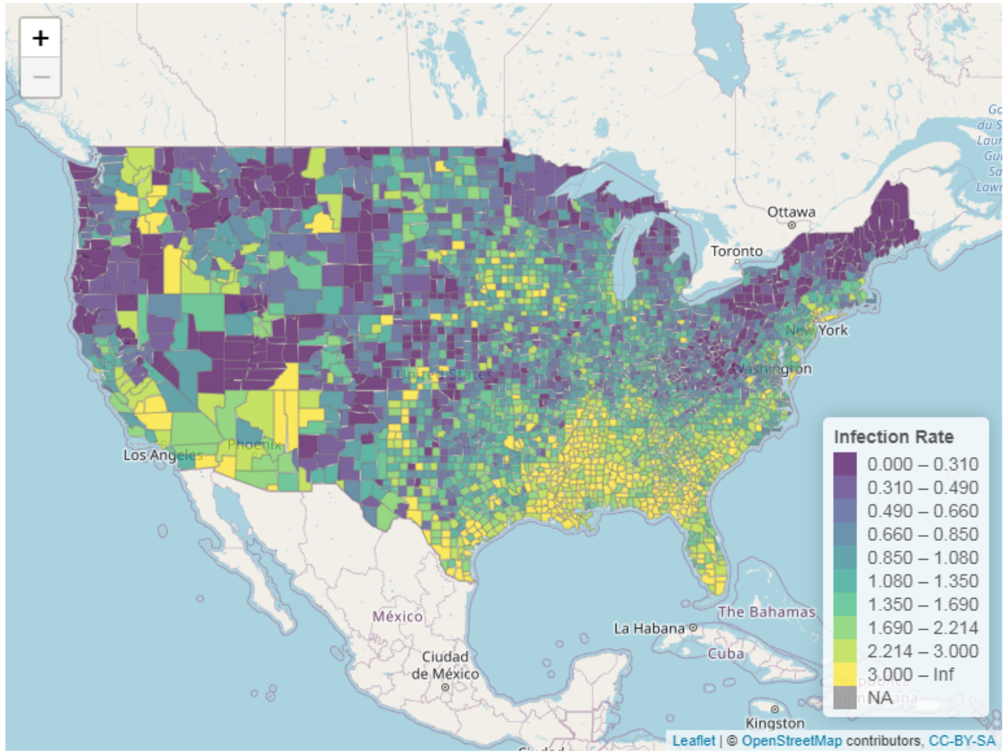
Data from JHU CSSE COVID-19: https://github.com/CSSEGISandData/COVID-19

State Cases and Deaths

Totals | Crude Infection and Mortality



County Level Map



If you want to follow along, you'll need:

- **R** v4.0 or higher, **R Studio** v1.3 or higher
- R packages: tidyverse, rmarkdown, knitr, lubridate, praise, gghighlight, leaflet, maps, glue, viridis, flexdashboard
- Create a new project in R Studio
- Save a copy of **COVID_US.rmd** and **download_cleanup.R** from https://github.com/mattheaphy/pa4_rmarkdown to your project folder
- Run **download_cleaup.R**. If successful, you should see the file **covid19-us.rds** in your project folder

Agenda

Basics

YAML

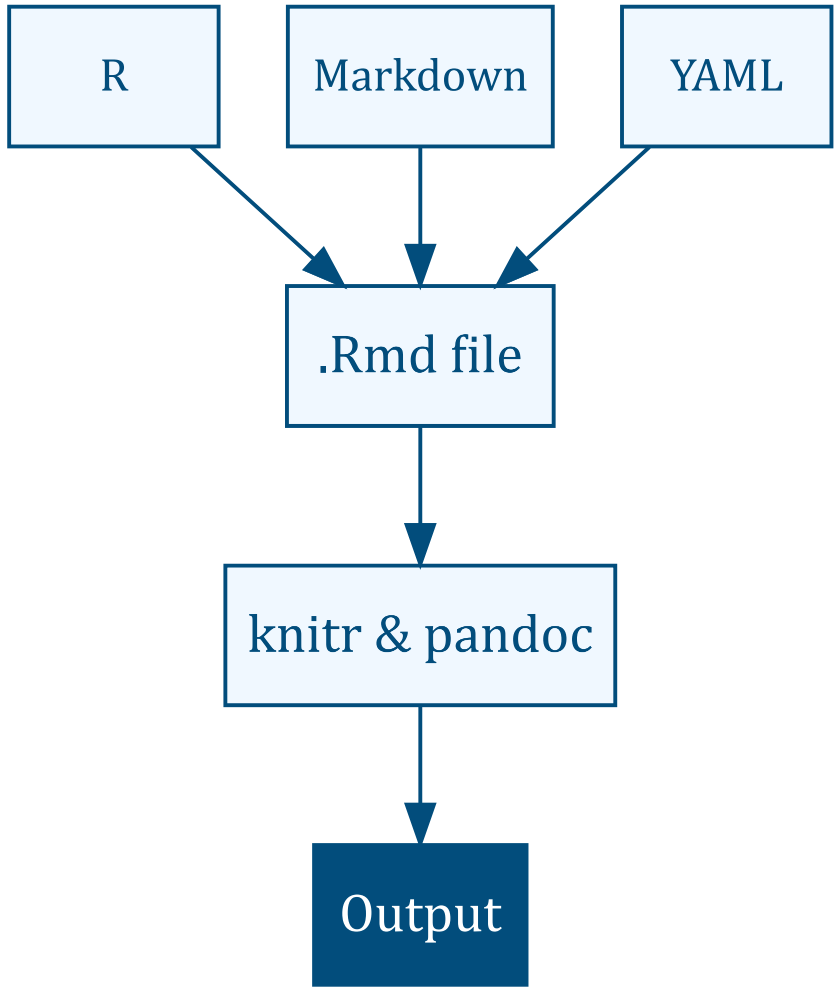
Markdown

Code chunks

Extras

R Markdown Process Flow

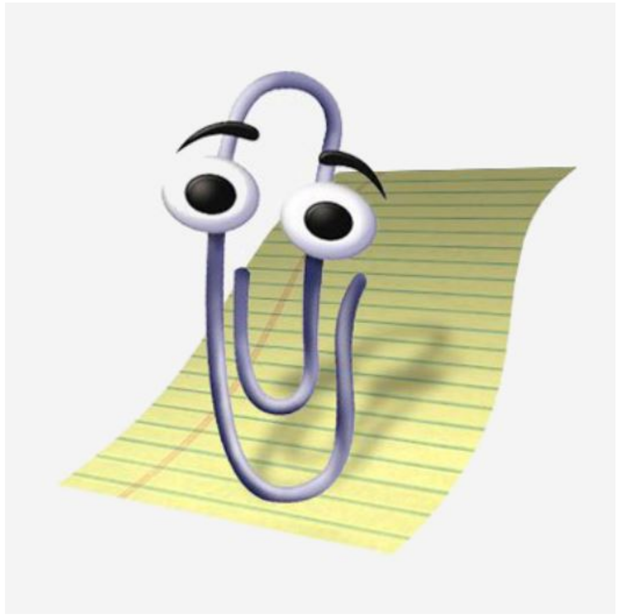
1. Create a new RMD file
2. Write a YAML header
3. Write the report using a combination of `Markdown` and R
4. Render, or *"knit"* the RMD file into the output format of your choice



Multiple Output formats

- HTML documents
- HTML slides shows (`ioslides`, `beamer`, `xaringan`)
- HTML Dashboards (`flexdashboard`)
- PDF documents (requires a separate installation of `LaTeX`)

- Microsoft Word
- Microsoft PowerPoint



Creating an `rmd` file

Beginners:

File > New File > R Markdown > then choose a template

Experts:

- Create a blank file with **Ctrl + Shift + N**
- Change the document type in the lower-right of the Source pane to **R Markdown**
- Save the file with an .rmd extension

Knitting

Click the Knit button in R Studio

Or, press **Ctrl + Shift + K**

Or, knit using the function `rmarkdown::render`

```
rmarkdown::render("COVID_US.Rmd", output_file = "New England Report.html",
                  params = list(states = c("Connecticut", "Massachusetts",
                                          "Rhode Island", "Maine",
                                          "New Hampshire", "Vermont")))
```

Basics

YAML

Markdown

Code chunks

Extras

YAML Headers

YAML Headers

The YAML[1] header specifies important information about your document

- Series of key-value pairs
- Starts and ends with `---`
- Hierarchical - indentation matters

```yaml
---
title: "COVID-19 Analysis"
author: "`r praise::praise('${Adjective} Actuary')`"
date: "2020-09-25"
output:
    html_document:
        toc: true
        toc_float: true
        theme: flatly
---
```

[1] YAML = YAML Ain't Markup Language
For a list of themes, see https://bootswatch.com/3/

More on output types

Use these R functions in YAML to specify output formats

Output type	Function
HTML	`html_document`
HTML slide shows	`ioslides_presentation`, `beamer_presentation`, `xaringan::moon_reader`
HTML Dashboard	`flexdashboard::flex_dashboard`
PDF	`pdf_document`
Word	`word_document`
PowerPoint	`powerpoint_presentation`

Other Common YAML Tags

High level tags

- `title:` document's title
- `subtitle:` document's subtitle
- `author:` author's name
- `date:` date
- `params:` document variables
- `runtime: shiny` enables **shiny** interactivity

html_document tags

- `number_sections:` whether headers (#, ##, etc.) should be numbered
- `toc:` table of contents
- `toc_depth:` number of header levels in table of contents
- `toc_float:` "floating" table of contents
- `code_folding:` allows readers to hide / unhide R code
- `theme:` load a pre-built theme
- `css:` add your custom theme

Basics

YAML

Markdown

Code chunks

Extras

Markdown Syntax

Markdown

```
Most Markdown looks like plain text. Symbols
can be added for **bold**,*italics*,
~~strikethrough~~, and `code` formatting.
```

Inline R code

```
You rolled a `r sample(1:6, 1)`!
```

Mathematical expressions

```
$a_x=\sum_{t=1}^{\infty}{v^t}
{_t }{p_x}$
```

Hyperlinks

```
[SOA hyperlink](www.soa.org)
```

Images

```
![](https://imgs.xkcd.com/comics/here_to_help.png)
```

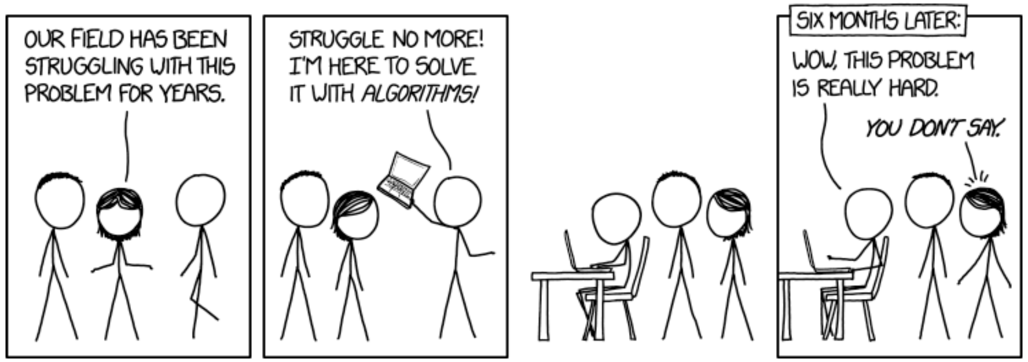
Rendered

Most Markdown looks like plain text. Symbols can be added for **bold**, *italics*, ~~strikethrough~~, and `code` formatting.

You rolled a 3!

$a_x = \sum_{t=1}^{\infty} v^t {}_t p_x$

SOA hyperlink


OUR FIELD HAS BEEN STRUGGLING WITH THIS PROBLEM FOR YEARS.
STRUGGLE NO MORE! I'M HERE TO SOLVE IT WITH ALGORITHMS!
SIX MONTHS LATER:
WOW, THIS PROBLEM IS REALLY HARD.
YOU DON'T SAY.

Markdown

```
# Level 1 header

## Level 2 header

...

- bulleted
  - list

1. numbered
1. list

> "*Insightful comment*", expert
```

Rendered

Level 1 header

Level 2 header

...

- bulleted
  - list

1. numbered
2. list

> "*Insightful comment*", expert

Basics

YAML

Markdown

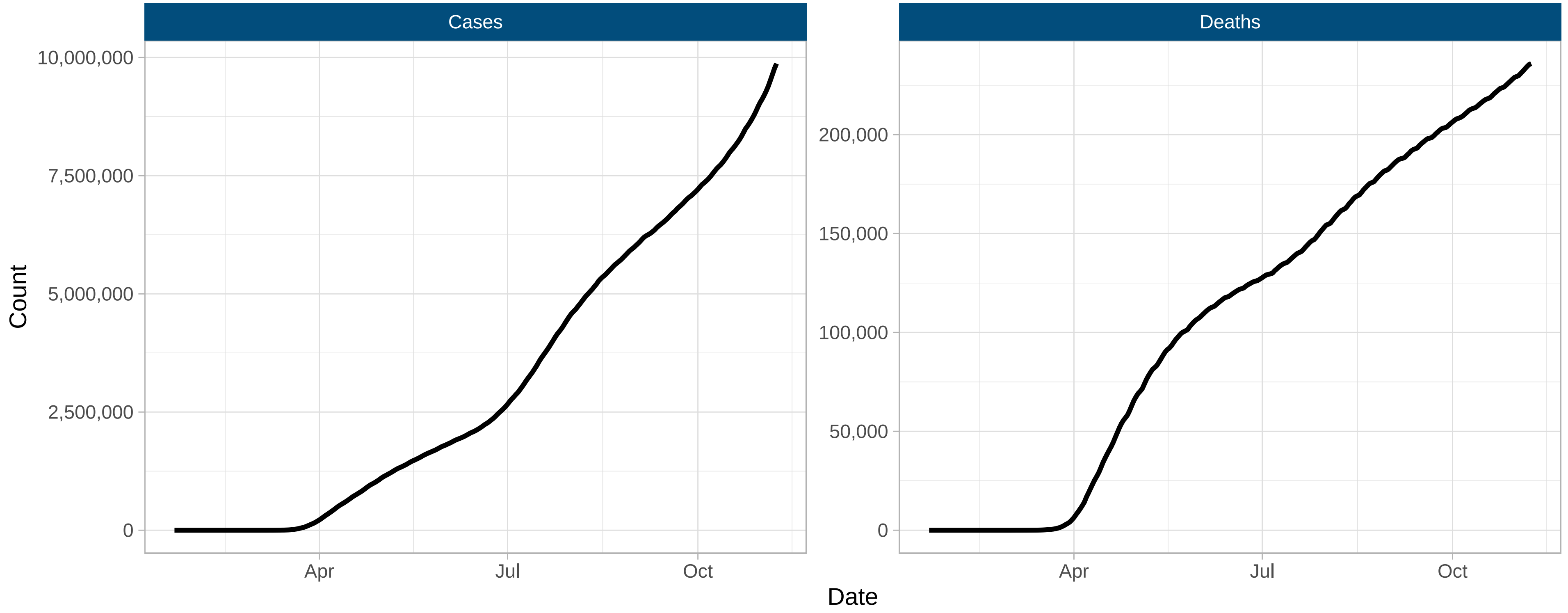
Code chunks

Extras

R Code Chunks

R + Markdown = R Markdown

```r
prep(dat) %>% filter(metric %in% c("Cases", "Deaths")) %>%
    ggplot(aes(cal_date, value)) + geom_line(lwd = 1.05) +
    scale_y_continuous(labels = scales::comma) +
    facet_wrap(~ metric, scales = "free_y") + labs(x = "Date", y = "Count")
```



Code chunk skeleton

- To create a chunk, press `Ctrl + Alt + I`
- Or select `Insert > R` within R Studio
- Or type it by hand

An empty chunk

~~~
```{r}

```
~~~

A chunk with a name

~~~
```{r the-answer}
2 * sum(1:6)
```
~~~

```
## [1] 42
```

Code chunk options

TRUE / FALSE options

- `echo`: Should the `R` code be printed?
- `eval`: Should the `R` code be executed?
- `message` and `warning`: Should messages and warnings be printed?
- `include`: Set to `FALSE` to execute code but prevent any printing

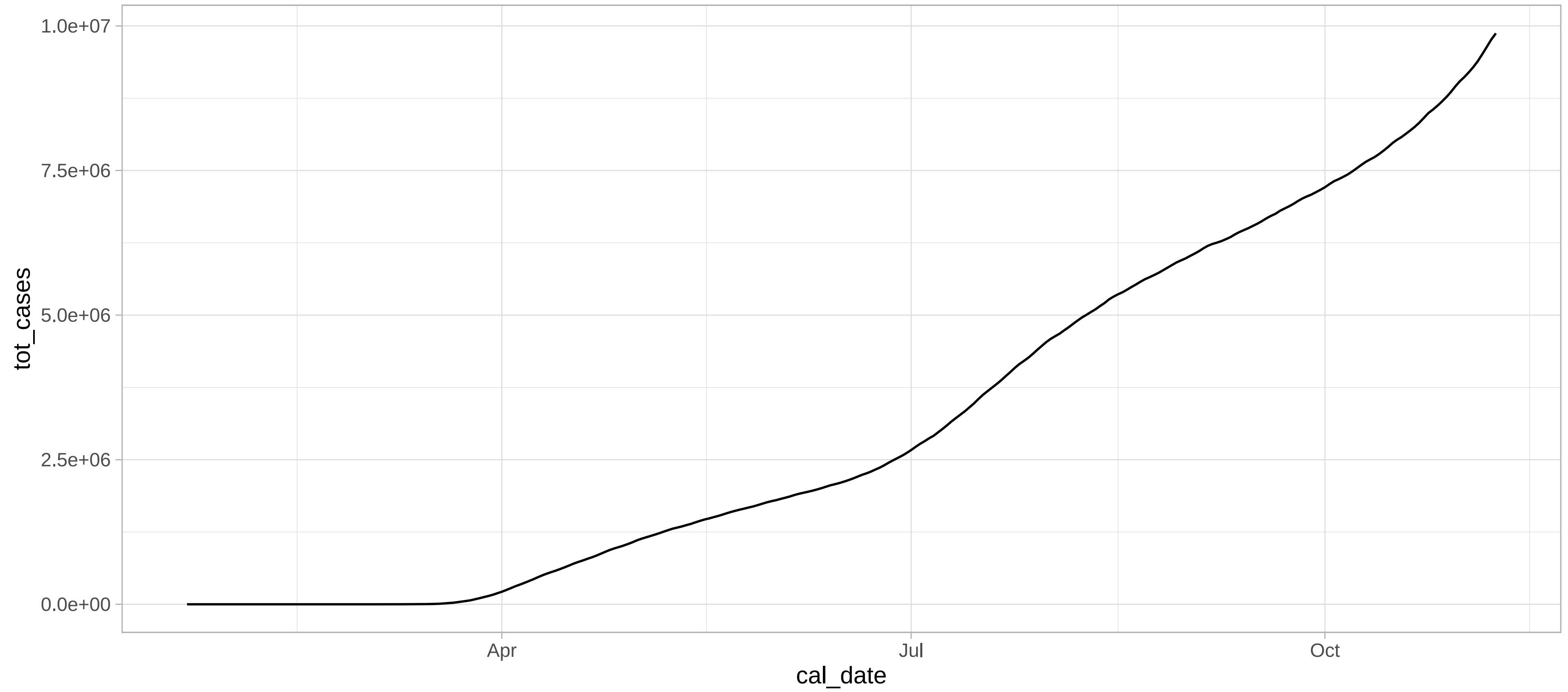
Options for plots and tables

- `fig.height`, `fig.width`: Dimensions of output figures in inches
- `fig.cap`: Caption

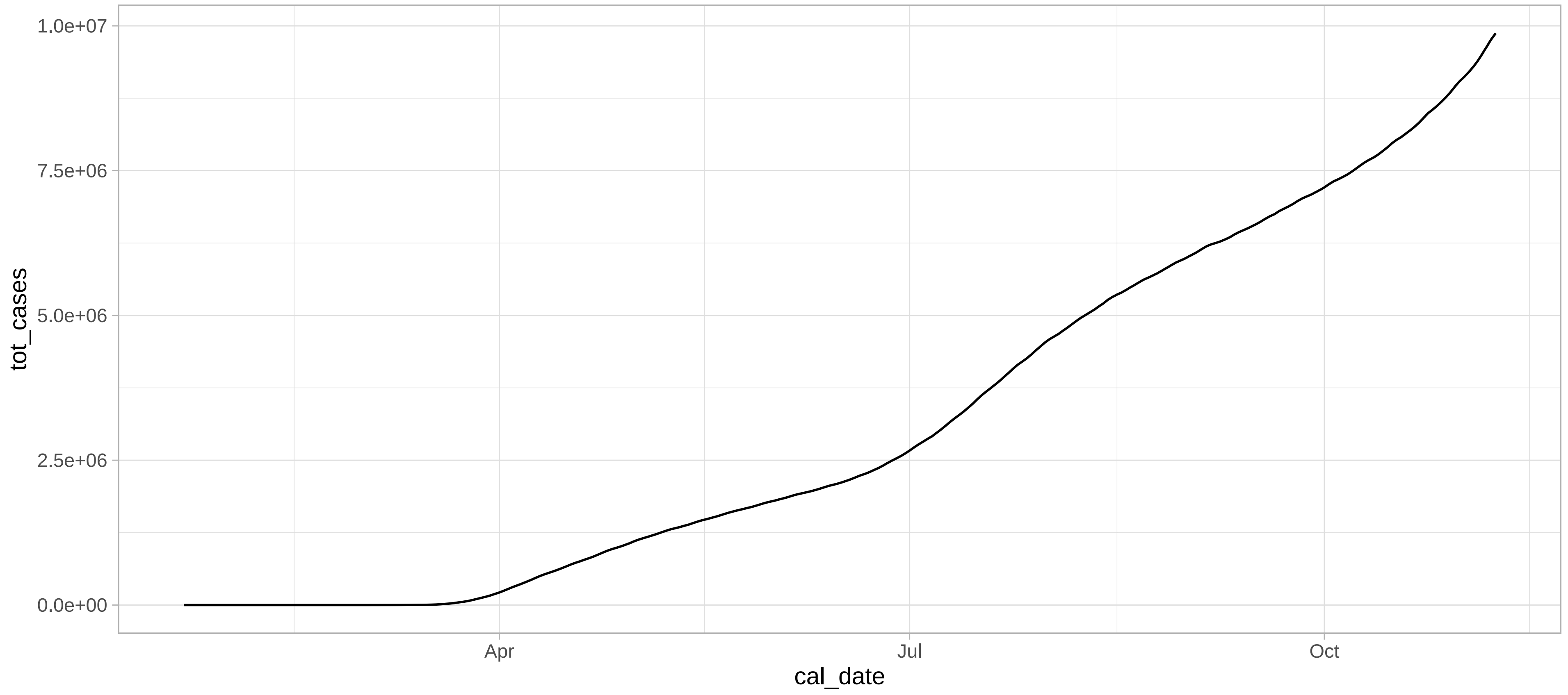
... and many more

{r, echo=TRUE}

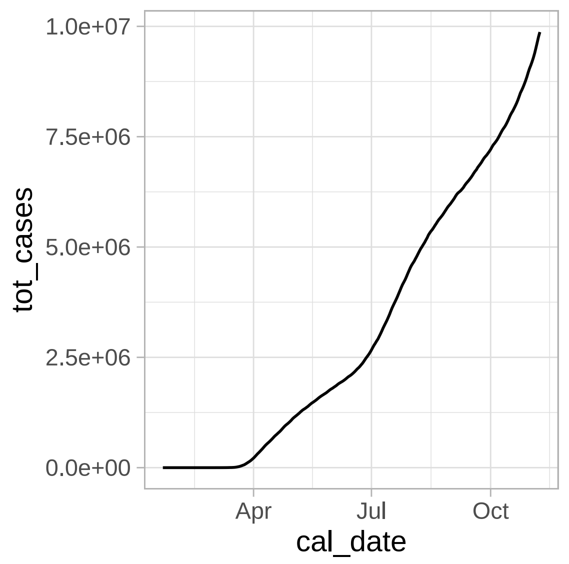
```
dat %>% group_by(cal_date) %>% summarize(tot_cases = sum(tot_cases)) %>%
    ggplot(aes(cal_date, tot_cases)) + geom_line()
```



{r, echo=FALSE}



{r, echo=FALSE, fig.height = 3, fig.width = 3}



Default chunk options

- Set defaults at the beginning of each document with `knitr::opts_chunk$set`
- Common practice is to include a chunk named **setup**
- Always set `include` to FALSE

Example:

~~~
```{r setup, include = FALSE}
knitr::opts_chunk$set(echo = FALSE, message = FALSE, warning = FALSE,
                      fig.height = 6, fig.width = 8)
```
~~~

Parameterized documents

Setting up default parameters

```
params:
    states: "All"
```

Using parameters in a report

```
if (params$states[[1]] != "All") dat <- filter(dat, State %in% params$states)
```

Multiple report automation with parameters

```
walk(state.name, ~ rmarkdown::render("COVID_US.Rmd",
    output_file = str_glue("{.x}.html"),
    params = list(states = .x)))
```

Basics

YAML

Markdown

Code chunks

Extras

Extras

Working with tables

Popular options for tables:

- Paged data frames (`paged.print = TRUE`)
- `kable()` from the `knitr` package
- `datatable()` from the `DT` package

```r
x <- mutate(economics, Year = year(date)) %>%
  group_by(Year) %>%
  summarize(`Savings Rate` =
              mean(psavert) %>% round(1)) %>%
  tail()
```

- See the knitr and kableExtra styling guide for tips on working with **kable** tables
- See the DT package documentation for more on datatables

```r
knitr::kable(x, format = "html")
```

Year	Savings Rate
2010	6.6
2011	7.2
2012	8.8
2013	6.4
2014	7.3
2015	7.7

Support for other languages

- **Python, SQL, C++, Julia, Javascript, CSS**, and more...
- The game-changing `reticulate`[1] package enables switching back and forth between **R** and **Python**

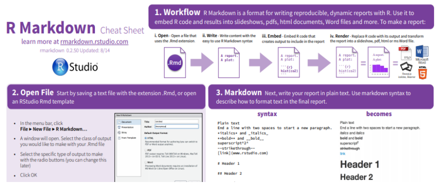
~~~
```{python}
x = 3.14
```

```{r}
py$x <- py$x * 2
```
~~~

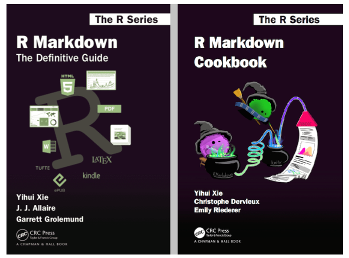
[1] See https://rstudio.github.io/reticulate/index.html

Notebooks

- R Markdown has notebook-like functionality similar to **jupyter** notebooks
- In **R Studio**, click on the gear icon at the top of the source pane and select "Chunk Output Inline"
- All output from code chunks will be printed in the `rmd` file
- Well-suited for exploratory analysis and document drafting

Additional Resources





- Tutorials and articles at https://rmarkdown.rstudio.com/
- R Studio's R Markdown Cheat Sheet: https://rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf
- R Markdown: The Definitive Guide, by Yihui Xie, J.J. Allaire, and Garrett Grolemund: https://bookdown.org/yihui/rmarkdown/
- R Markdown Cookbook, by Yihui Xie, Christophe Dervieux, and Emily Riederer: https://bookdown.org/yihui/rmarkdown-cookbook/verbatim-code-chunks.html

Thank you!