04 - R Markdown

Data Science with R · Summer 2021

Uli Niemann · Knowledge Management & Discovery Lab

https://brain.cs.uni-magdeburg.de/kmd/DataSciR/

Photo courtesy of Ulrich Arendt

What is R Markdown? from RStudio, Inc. on Vimeo.

R Markdown (index.html)

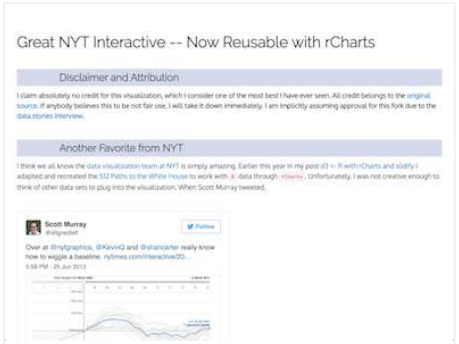
from (https://www.rstudio.com/) Get Started (lesson-1.html) Gallery (gallery.html) Formats (formats.html) Articles (articles.html) Book (https

Gallery

Check out the range of outputs and formats you can create using R Markdown.

Documents

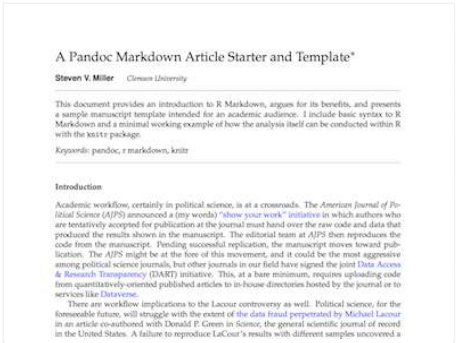
With R Markdown, you write a single .Rmd file and then use it to render finished output in a variety of formats.



(http://timelyportfolio.github.io/rCharts_nyt_home_price/)

HTML (http://timelyportfolio.github.io/rCharts_nyt_home_price/)

HTML documents for web publishing.



(https://github.com/svmiller/svm-r-markdown-templates/blob/master/article-example/svm-rmarkdown-article-example.pdf)

Microsoft Word (https://github.com/svmiller/svm-r-markdown-templates/tree/master/word-example)

(https://rstudio.github.io/tufte/)

Handouts (https://rstudio.github.io/tufte/)

Tufte styled documents for handouts.
Example Code

Why reproducible documents?

https://youtu.be/s3JldKoA0zw

R Markdown

- Introduction
- Output formats
- Components of an `.Rmd` file: Markdown, Code, YAML
- `knitr` and `pandoc`
- Basic chunk options
- Embedding graphics & tables
- Caching
- References and bibliography
- Parametrized documents
- R Markdown extensions

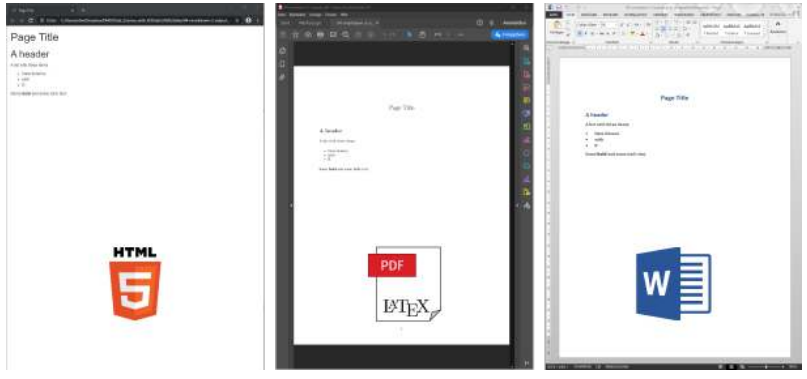
Introduction

R Markdown is a file format to combine **code**, the associated **results** and **narrative text** in a simple text file, to create **reproducible reports** which can be **flexibly distributed in multiple ways**.

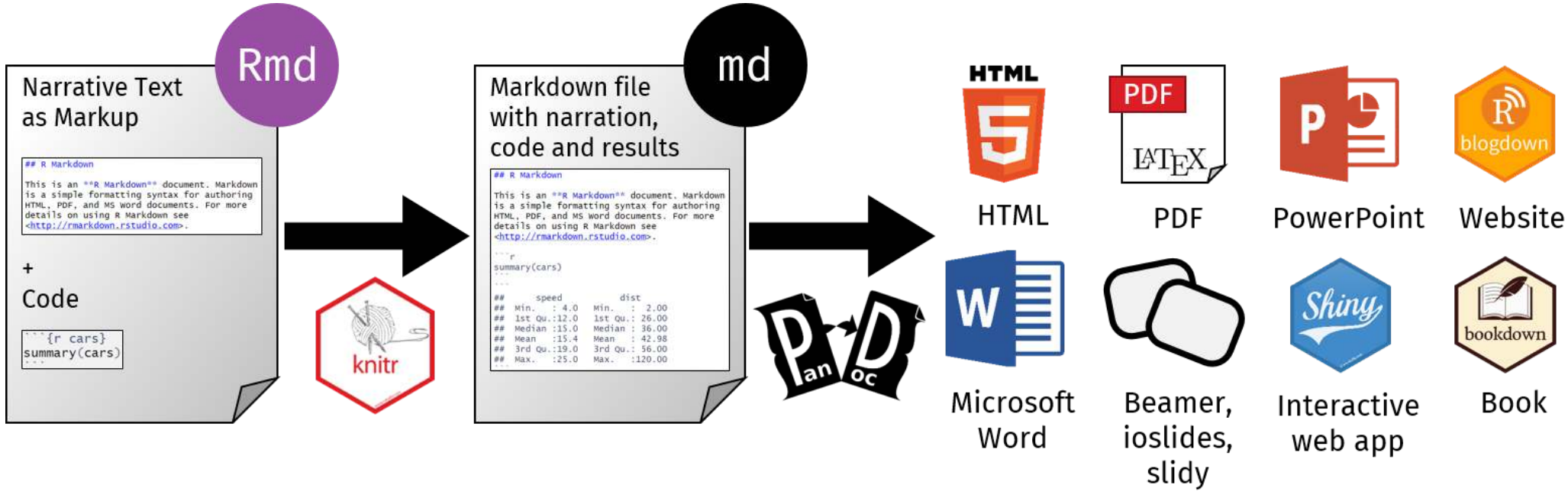
An R Markdown document is saved as `.Rmd` file. It contains both the (`R`) code and a prose description of a data analysis task. The R Markdown document can be rendered as HTML, PDF, Word and various other output formats.

Pros of R Markdown documents:

- **reproducibility** (reduce **copy&paste**)
- **simple Markdown syntax** for text
- a **single source document** that can be rendered for different target audiences and purposes
- **simple, future-proof file format** that can be managed by a version control system like Git or SVN



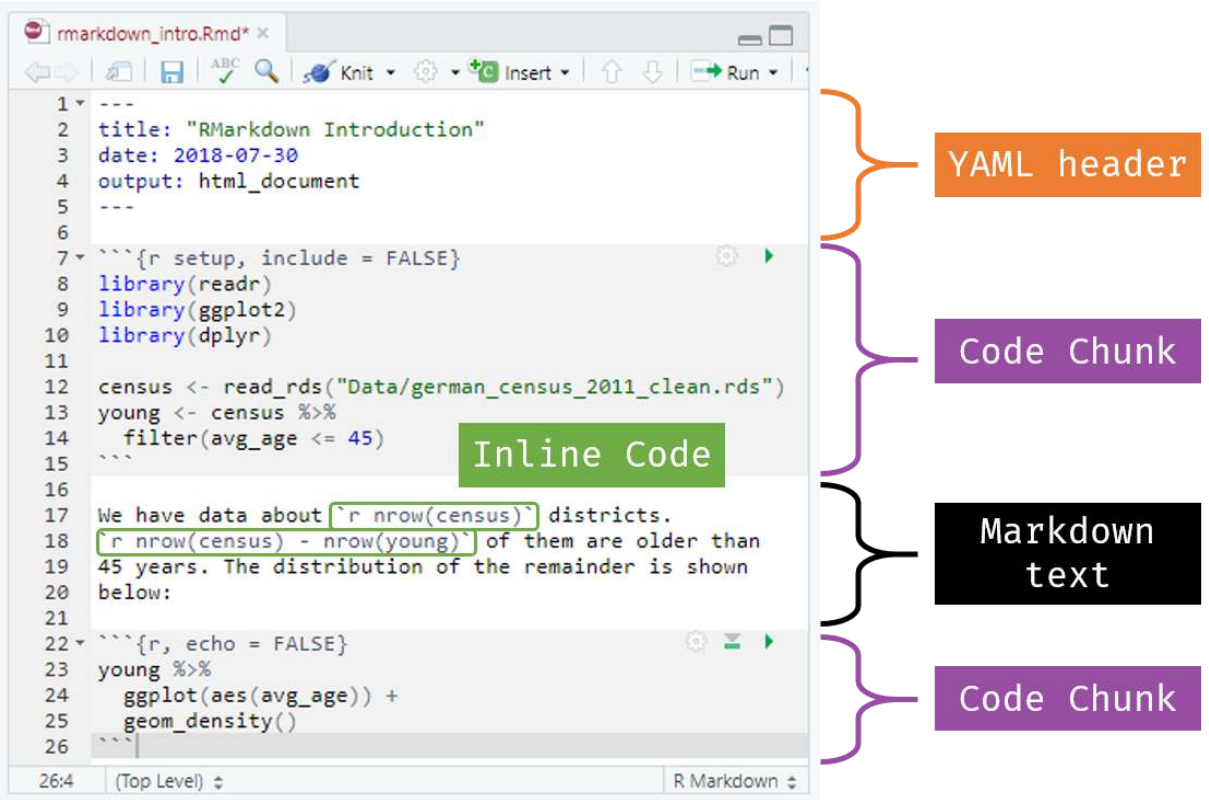
The process of "knitting" an R Markdown file



A .Rmd file contains narrative text in Markdown syntax and (`R`) code. The `knitr` package provides functions to execute the code, collect the results (statistics, tables, plots, etc.) and combine them with the Markdown text into a Markdown file (.md). Pandoc, a universal document converter, then converts this Markdown file into the requested output format, e.g., .html, .pdf or .docx.

Components of an R Markdown file

Structure | Output



- **YAML header** (metadata)
  - enclosed by lines with three dashes
  - collection of key-value pairs separated by colons
- narrative text as **Markdown markup**
- `R` **code**
  - as **code chunks** surrounded by ```` ```{r} ```` and ```` ``` ````
  - as **inline code** surrounded by `` `r `` and `` ` ``

Components of an R Markdown file

Structure | Output

R Markdown Intro

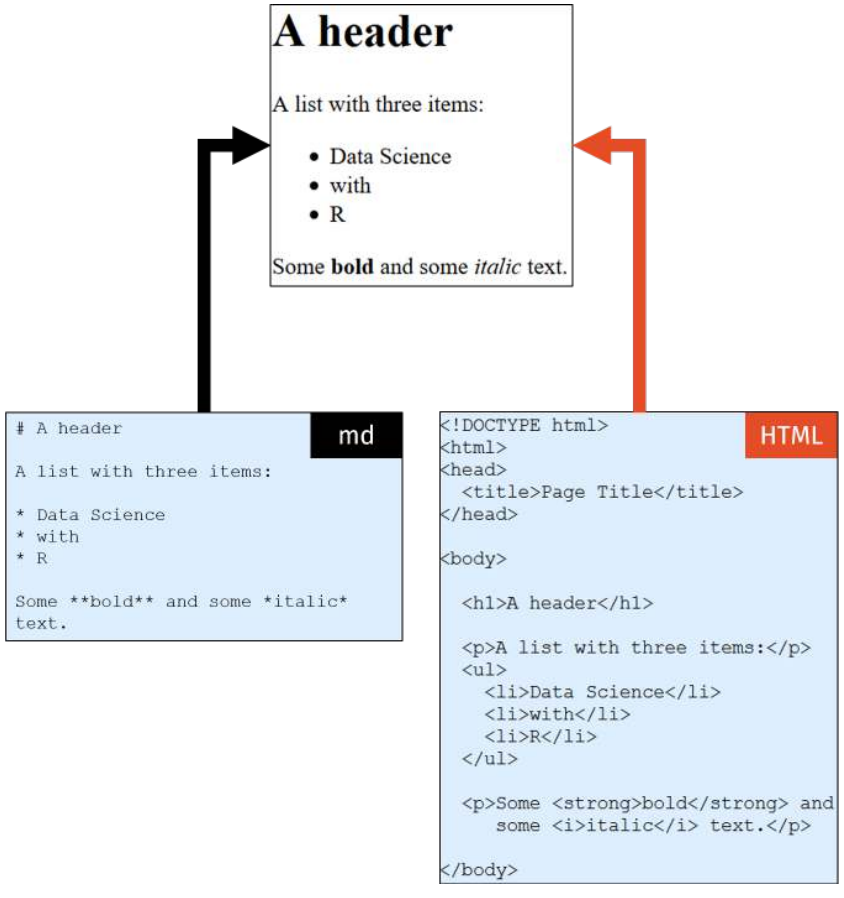
2018-07-30

```
library(tidyverse)
age_threshold <- 45
census <- read_rds(here::here("datasets", "german_census_2011_clean.rds"))
(young <- census %>%
  filter(avg_age <= age_threshold))
```

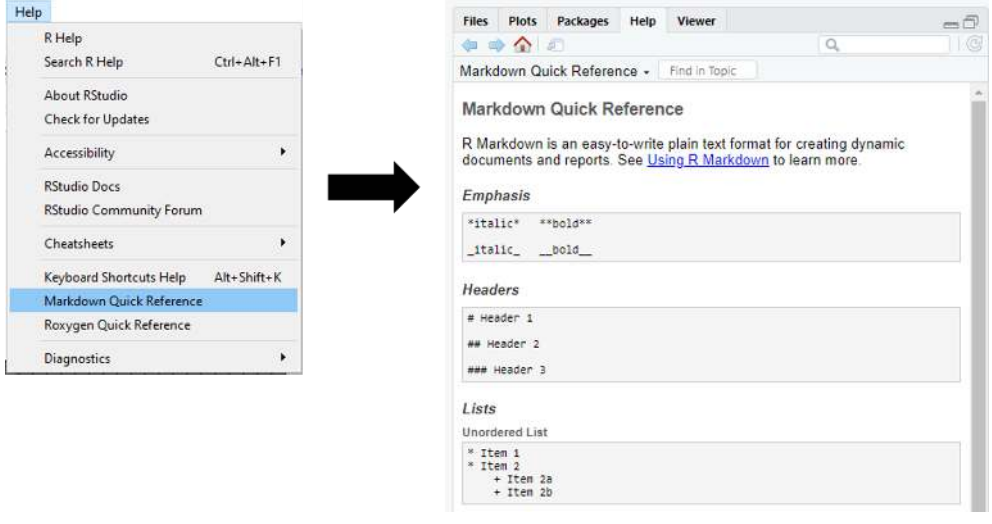
```
## # A tibble: 327 x 7
##    district_name avg_age perc_mig_bg perc_unemp avg_household_s~
##    <chr>           <dbl>       <dbl>      <dbl>             <dbl>
##  1 Flensburg, K~    42.7        16          6.7               1.8
##  2 Kiel, Landes~    41.3        18.9        7.7               1.9
##  3 Lübeck, Hans~    44.2        16.8        7.4               1.9
##  4 Neumünster, ~    43.5        16.9        7.1               2
##  5 Dithmarschen~    44.1         7          5.7               2.2
##  6 Herzogtum La~    43.6        12.6        4                 2.3
##  7 Nordfrieslan~    43.8         8.5        3.6               2.2
##  8 Pinneberg, L~    43.6        15          3.8               2.2
##  9 Plön, Landkr~    45           6.9        4.2               2.3
## 10 Rendsburg-Ec~    43.5         8.5        3.9               2.3
## # ... with 317 more rows, and 2 more variables: perc_children_household <dbl>,
## #   vacancy_rate <dbl>
```

Markdown

Markdown is a **simplified, plain text formatting system** used to structure text using **markup tags**. In contrast to other markup languages like **HTML** and **XML**, Markdown markup is easier to read and write due to its simpler tags, e.g. `#` instead of `<h1></h1>` for headers of level 1.



Markdown Quick Reference in RStudio:
Help → **Markdown Quick Reference**



Learn Markdown basics interactively on https://www.markdowntutorial.com in ~10 minutes.

Markdown is a way to write content for the web. It's written in what people like to call "plaintext", which is exactly the sort of text you're used to writing and seeing. Plaintext is just the regular alphabet, with a few familiar symbols, like asterisks (*) and backticks (`).

Unlike cumbersome word processing applications, text written in Markdown can be easily shared between computers, mobile phones, and people. It's quickly becoming the writing standard for academics (http://chronicle.com/blogs/profhacker/markdown-the-syntax-you-probably-already-know/35295), scientists (http://blogs.plos.org/mfenner/2012/12/13/a-call-for-scholarly-markdown/), writers (https://lifehacker.com/5943320/what-is-markdown-and-why-is-it-better-for-my-to+do-lists-and-notes), and many more. Websites like GitHub (https://www.github.com) and reddit (https://www.reddi
Markdown to style their comments.

Formatting text in Markdown has a very gentle learning curve. It doesn't do anything fancy like change the font size, color, or type. All you h
over is the display of the text—stuff like making things bold, creating headers, and organizing lists.

If you have ten minutes, you can learn Markdown!

In each lesson, you'll be given an introduction to a single Markdown concept. Then, you'll be asked to complete several exercises with that new knowledge.

Ready? Let's get started! (/lesson/1)

Syntax

italic and **bold**

`inline code`

subscript~2~/superscript^2^

~~strikethrough~~

escaped: * _ \\

en-dash: -- em-dash: ---

Header level 1

Header level 2

Manual line break:
(2+ space characters at the line end):

Backe, backe, Kuchen,
der Bäcker hat gerufen!

Backe, backe, Kuchen,**SPACE SPACE**
der Bäcker hat gerufen!

Result

italic and **bold**

`inline code`

subscript$_2$/superscript^2

~~strikethrough~~

escaped: * _ \

en-dash: – em-dash: —

Header level 1

Header level 2

Backe, backe, Kuchen, der Bäcker hat gerufen!

Backe, backe, Kuchen,
der Bäcker hat gerufen!

Syntax

```
- unordered list
    - subitem 1 (4 **SPACE**)
    - subitem 2
        - subsubitem (8 **SPACE**)

1. ordered list
1. item 2
  1. subitem 2.1
  1. subitem 2.2

inline equation: $A = \pi\cdot r^{2}$

math block: $$A = \pi\cdot r^{2}$$


[linked text](https://www.ovgu.de/)

Footnote[^1]

[^1]: This text belongs to the footnote.

<!-- This is a comment that will not be displayed.-->
```

Result

- unordered list
  - subitem 1
  - subitem 2
    - subsubitem

1. ordered list
2. item 2
    1. subitem 2.1
    2. subitem 2.2

inline equation: $A = \pi \cdot r^2$

math block:

$$A = \pi \cdot r^2$$

linked text

Footnote[1]

[1] This text belongs to the footnote.

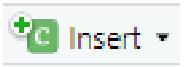
knitr

The `knitr` package is responsible for evaluating the `R` code within an `.Rmd` file and combining the results, figures, code and the Markdown text into a `.md` file.

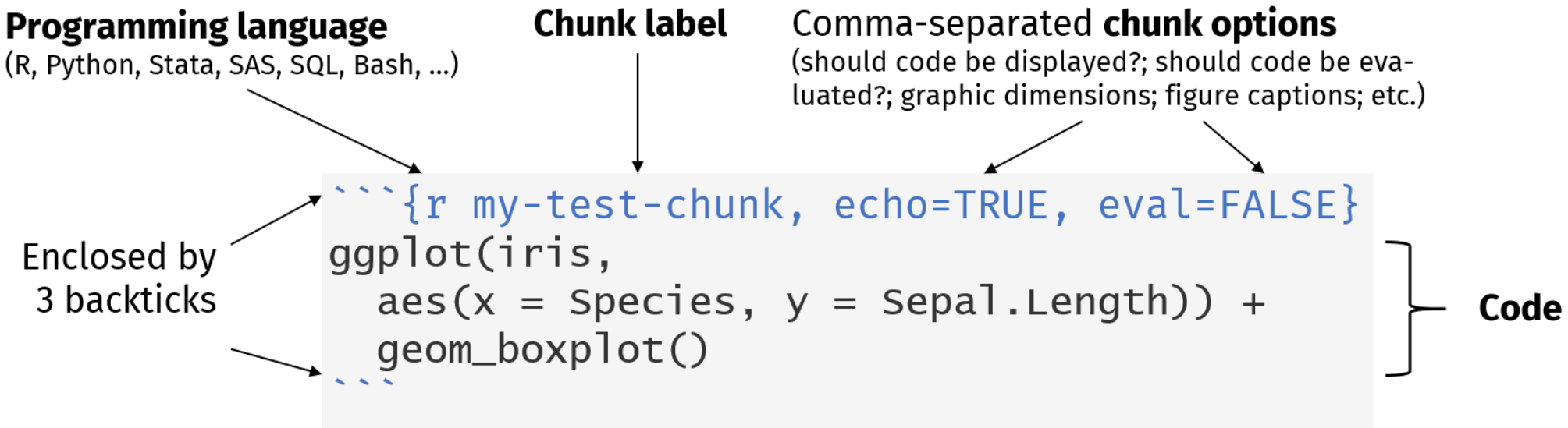
`R` code can be embedded within an `.Rmd` file in two ways:

A) as **inline code** within markdown text delimited by single backticks tags to run a single command: `` `r` ``
B) as separate **code chunk** delimited by triple backticks tags to run multiple commands:

````
```{r}
some code
```
````

Code chunks can be inserted with the 

button in the toolbar or using the keyboard shortcut **Ctrl + Alt + I**.


Programming language
(R, Python, Stata, SAS, SQL, Bash, ...)
Chunk label
Comma-separated **chunk options**
(should code be displayed?; should code be evaluated?; graphic dimensions; figure captions; etc.)
Enclosed by
3 backticks
```{r my-test-chunk, echo=TRUE, eval=FALSE}
ggplot(iris,
aes(x = Species, y = Sepal.Length)) +
geom_boxplot()
```
Code

Chunk options

`knitr` allows to customize each code chunk in the report by providing **optional arguments**:

The most important options include:

- whether to show or hide the source code in the rendered document
- whether to evaluate the code
- whether warnings and messages should be shown or suppressed in the rendered document
- figure dimension

Chunk options are specified comma-separated inside the curly brackets after `r`.

Chunk options

By default, R Markdown includes **messages**, **warnings** and **error messages**.

````
```{r}
c(1,2,3) + c(4,5)
```
````

```
## Warning in c(1, 2, 3) + c(4, 5): longer object length is not a multiple of shorter object
## length
```

```
## [1] 5 7 7
```

We can override this default behavior by setting `warning = FALSE` inside the code chunk header.

````
```{r, warning = FALSE}
c(1,2,3) + c(4,5)
```
````

```
## [1] 5 7 7
```

Popular chunk options

Option	Description	Default
`echo`	Should the code be displayed?	`TRUE`
`eval`	Should the code be run?	`TRUE`
`include`	Should the code and the code results (figures, console output) be displayed?	`TRUE`
`fig.width`	Figure width in inch	`7L`
`fig.height`	Figure height in inch	`7L`
`fig.align`	Figure alignment	`"default"`
`warning`	Should warnings be displayed?	`TRUE`
`message`	Should messages be displayed?	`TRUE`

- Run `knitr::opts_chunk$get()` to see the current default values for all chunk options.
- Use `knitr::opts_chunk$set()` to modify these defaults, e.g. `knitr::opts_chunk$set(warning = FALSE)`. Example: `knitr::opts_chunk$set(fig.width = 8)` changes the default width of a figure to 8 inch.

A list of all chunk options is available at https://yihui.org/knitr/options/#chunk-options.

Options

Chunk options and package options

2020-06-30

Setup chunk

```
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = FALSE)
library(tidyverse)
```
```

The setup chunk at the beginning of the document is useful to:

- set global options for the whole document
- load required packages
- load all required datasets

Before running a code chunk in Nodebook mode, R Markdown will always execute the setup code chunk first.

External graphics

```
```{r r-markdown-logo, fig.cap="R Markdown Logo"}
knitr::include_graphics("figures/rmarkdown-logo.png")
```
```

R Markdown Logo

Tables: kable

```
```{r kable-iris}
knitr::kable(head(iris), caption = "A knitr kable table")
```
```

A knitr kable table

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

R Markdown Cookbook: https://bookdown.org/yihui/rmarkdown-cookbook/kable.html

10.1 The function `knitr::kable()`

The `kable()` function in knitr is a very simple table generator, and is simple by design. It only generates tables for strictly rectangular data such as matrices and data frames. You cannot heavily format the table cells or merge cells. However, this function does have a large number of arguments for you to customize the appearance of tables:

```
kable(x, format, digits = getOption("digits"), row.names = NA,
  col.names = NA, align, caption = NULL, label = NULL,
  format.args = list(), escape = TRUE, ...)
```

10.1.1 Supported table formats

In most cases, `knitr::kable(x)` may be enough if you only need a simple table for the data object `x`.

`kableExtra` package: https://cran.r-project.org/web/packages/kableExtra/vignettes/awesome_table_in_html.html

Create Awesome HTML Table with knitr::kable and kableExtra

Hao Zhu

2021-02-19

> Please see the package documentation site (https://haozhu233.github.io/kableExtra/) for how to use this package in LaTeX.

Overview

The goal of `kableExtra` is to help you build common complex tables and manipulate table styles. It imports the pipe `%>%` symbol from `magrittr` and verbalize all the functions, so basically you can add "layers" to a kable output in a way that is similar with `ggplot2` and `plotly`.

For users who are not very familiar with the pipe operator `%>%` in R, it is the R version of the fluent interface (https://en.wikipedia.org/wiki/Fluent_interface). The ides is to pass the result along the chain for a more literal coding experience. Basically when we say `A %>% B`, technically it means sending the results of A to B as B's first argument.

To learn how to generate complex tables in LaTeX, please visit http://haozhu233.github.io/kableExtra/awesome_table_in_pdf.pdf (http://haozhu233.github.io/kableExtra/awesome_table_in_pdf.pdf)

Tables: DT datatable

`DT` is an R interface to the DataTables JavaScript library.

```
```{r}
DT::datatable(iris)
```
```

Show 10 entries Search:

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa

DT package: https://rstudio.github.io/DT/

DT: An R interface to the DataTables library

The R package **DT** provides an R interface to the JavaScript library **DataTables** (http://datatables.net). R data objects (matrices or data frames) can be displayed as tables on HTML pages, and **DataTables** provides filtering, pagination, sorting, and many other features in the tables.

You may install the stable version from CRAN, or the development version using `remotes::install_github('rstudio/DT')` if necessary (this website reflects the development version of **DT**):

```
# if (!require("DT")) install.packages('DT')
xfun::session_info('DT')
```

```
## R version 3.6.2 (2019-12-12)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Catalina 10.15.2
##
## Locale: en_US.UTF-8 / en_US.UTF-8 / en_US.UTF-8 / C / en_US.UTF-8 / en_US.UTF-8
##
## Package version:
```

Tables: gt

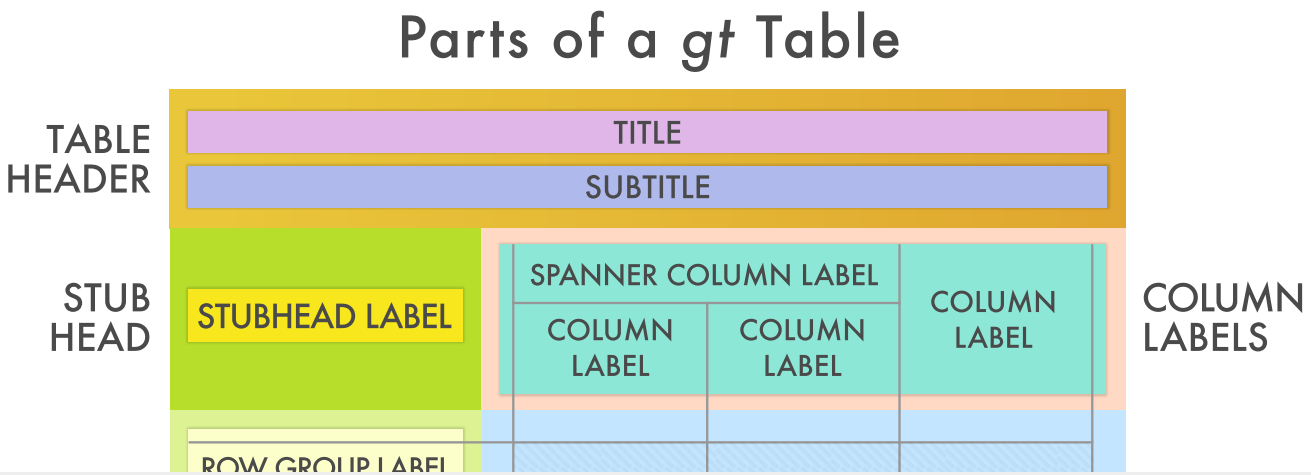
`gt`: "Easily generate information-rich, publication-quality tables from R"

````
```{r}
gt::gt(head(iris))
```
````

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

`gt`: "Easily generate information-rich, publication-quality tables from R" https://gt.rstudio.com/

With the **gt** package, anyone can make wonderful-looking tables using the **R** programming language. The **gt** philosophy: we can construct a wide variety of useful tables with a cohesive set of table parts. These include the *table header*, the *stub*, the *column labels* and *spanner column labels*, the *table body*, and the *table footer*.



Links

Download from CRAN at https://cloud.r-project.org/package=gt (https://cloud.r-project.org/package=gt)

Browse source code at https://github.com/rstudio/gt/ (https://github.com/rstudio/gt/)

Report a bug at https://github.com/rstudio/gt/issues (https://github.com/rstudio/gt/issues)

License

Caching

By default, all code chunks are **recomputed every time** you knit the file. For code chunks that contain **time-demanding operations**, you can use the `cache = TRUE` option. During the **first knit run**, the code is executed and the results are stored in a **local cache**. When the document is knitted again, the results are loaded from cache if the code chunk has not been edited since the first run. If the **code or data changes**, the code will be rerun and the **old cache will be overwritten**.

````
```{r, cache=TRUE}
...time-consuming computations...
```
````

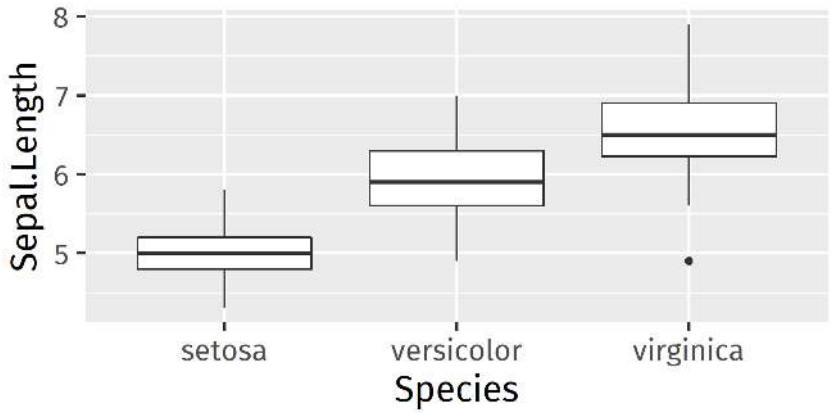
Labeling and reusing code chunks

Code chunks can be given a **label**. The label is specified directly after ```` ```{r ```` . Code chunk labels can be useful to **avoid code duplication**. By using the argument `ref.label` in a later code chunk, the code from the referenced code chunk is copied over to and run in the current code chunk.

Example:

````
```{r iris_plot, echo=FALSE, eval=FALSE}
library(ggplot2)
ggplot(iris, aes(Species, Sepal.Length)) +
  geom_boxplot()
```
````

````
```{r, ref.label='iris_plot', echo = FALSE}
```
````



Inline `R` code

You can embed `R` code into the text of your document with the `` `r ` `` syntax. R Markdown will run the code and replace it with its result, which should be a piece of text, such as a character string or a number.

For example, the line below uses embedded R code to create a complete sentence:

```
A random sample of 5 numbers from the set of numbers between
1 and 10 is `r sample(1:10, 5)`.
```

When you render the document the result will appear as:

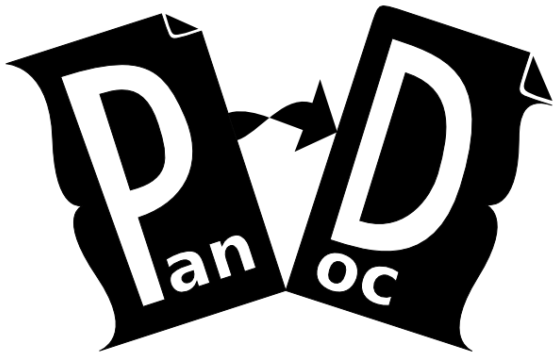
```
A random sample of 5 numbers from the set of numbers between
1 and 10 is 7, 8, 4, 2, 1.
```

Inline code provides a useful way to make your reports completely automatable.

Pandoc

R Markdown uses the document conversion application Pandoc to convert a `.md` file to a variety of output formats.

- using RStudio via the  button
- in the console via `rmarkdown::render()`



YAML

Properties of the whole document can be set via the **YAML** (*Yet Another Markup Language*) header.

The YAML header consists of a collection of **key-value pairs** separated by colons.

The YAML header itself is demarked by lines with three dashes.

Mandatory YAML specification:

```
output: html_document
```

Other output formats:

- pdf_document
- word_document
- beamer_presentation
- slidy_presentation
- ioslides_presentation
- md_document
- and many more...

Customize output layout

Each R Markdown output template is a collection of knitr and pandoc options. You can customize your output by overwriting the default options that come with the template.

For example, the YAML header below causes `knitr` to include a table of contents in the pdf output document:

```
---
title: "My R Markdown Document"
output:
  pdf_document:
    toc: true
---
```

sub-option	description	html	pdf	word	odt	rtf	md	github	ioslides	slidy	beamer
citation_package	The LaTeX package to process citations, natbib, biblatex or none		X				X				X
code_folding	Let readers to toggle the display of R code, "none", "hide", or "show"	X									
colortheme	Beamer color theme to use										X
css	CSS file to use to style document	X							X	X	
dev	Graphics device to use for figure output (e.g. "png")	X	X				X	X	X	X	X
duration	Add a countdown timer (in minutes) to footer of slides									X	
fig_caption	Should figures be rendered with captions?	X	X	X	X				X	X	X
fig_height, fig_width	Default figure height and width (in inches) for document	X	X	X	X	X	X	X	X	X	X
highlight	Syntax highlighting: "tango", "pygments", "kate","zenburn", "textmate"	X	X	X						X	X
includes	File of content to place in document (in_header, before_body, after_body)	X	X		X		X	X	X	X	X
incremental	Should bullets appear one at a time (on presenter mouse clicks)?								X	X	X
keep_md	Save a copy of .md file that contains knitr output	X		X	X	X			X	X	
keep_tex	Save a copy of .tex file that contains knitr output		X								X
latex_engine	Engine to render latex, "pdflatex", "xelatex", or "lualatex"		X								X
lib_dir	Directory of dependency files to use (Bootstrap, MathJax, etc.)	X							X	X	
mathjax	Set to local or a URL to use a local/URL version of MathJax to render equations	X							X	X	
md_extensions	Markdown extensions to add to default definition or R Markdown	X	X	X	X	X	X	X	X	X	X
number_sections	Add section numbering to headers	X	X								
pandoc_args	Additional arguments to pass to Pandoc	X	X	X	X	X	X	X	X	X	X
preserve_yaml	Preserve YAML front matter in final document?						X				
reference_docx	docx file whose styles should be copied when producing docx output			X							
self_contained	Embed dependencies into the doc	X							X	X	
slide_level	The lowest heading level that defines individual slides										X
smaller	Use the smaller font size in the presentation?								X		
smart	Convert straight quotes to curly, dashes to em-dashes, … to ellipses, etc.	X							X	X	
template	Pandoc template to use when rendering file quarterly_report.html).	X	X		X					X	X
theme	Bootswatch or Beamer theme to use for page	X									X
toc	Add a table of contents at start of document	X	X	X		X	X	X			X
toc_depth	The lowest level of headings to add to table of contents	X	X	X		X	X	X			
toc_float	Float the table of contents to the left of the main content	X									

From the RStudio R Markdown Cheat Sheet

Bibliography

```
---
output: html_document
bibliography: refs.bib
csl: ieee.csl
---

Insert a reference [@bib-key1; @bib-key2] in your Markdown text.
```

→ The bibliography will be inserted at the end of the document.

More information about bibliographies in R Markdown documents on the R Markdown Website from RStudio.

Parametrized reports

Create document templates to produce different reports with a single `.Rmd` using **parameters**.

Application scenarios:

- serial letters
- separate reports for different subsets of a dataset
- documents with different `knitr` behavior (e.g. one document showing the code and one document not showing the code)

````
---
output: html_document
params:
  my_species: "versicolor"
---

```{r iris-setup, include=FALSE, message = FALSE, warning = FALSE}
library(dplyr)
library(ggplot2)
my_species <- params$my_species
irs <- iris %>% filter(Species == my_species)
```

```{r hist}
ggplot(irs,aes(Sepal.Length)) +
  geom_histogram(bins=15,
                 color="black",
                 fill= "gray90")
```
````

Parametrized reports

Parametrized reports can be created using the **Knit** button, or by calling `rmarkdown::render()`. The later option is particularly useful if you have to create many parametrized reports.

Example: create a custom report for each species of Iris data.

```
library(dplyr)
reports <- tibble(
  my_species = unique(iris$Species),
  output_file = paste0("slides/04-rmarkdown-materials/iris-", my_species, ".html"),
  params = purrr::map(my_species, ~ list(my_species = .))
)
reports
```

```
## # A tibble: 3 x 3
##   my_species output_file                                    params
##   <fct>      <chr>                                          <list>
## 1 setosa     slides/04-rmarkdown-materials/iris-setosa.html     <named list [1]>
## 2 versicolor slides/04-rmarkdown-materials/iris-versicolor.html <named list [1]>
## 3 virginica  slides/04-rmarkdown-materials/iris-virginica.html  <named list [1]>
```

Parametrized reports

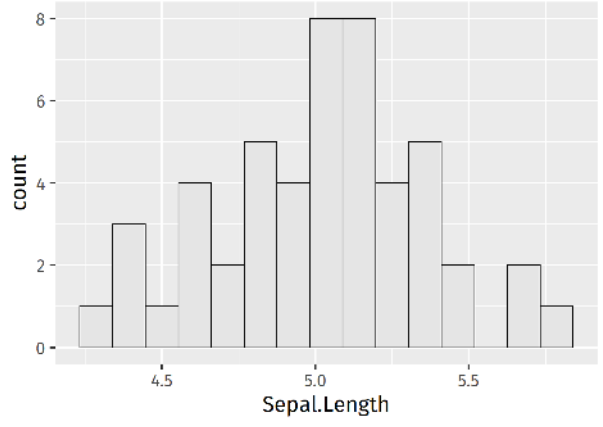
`purrr::pwalk` applies a function to a list of arguments. It is similar to `purrr::pmap`, but instead of returning an output value, it is called for its **side-effects**. Here, it is used to iterate over each row of the `reports` tibble, calling `rmarkdown::render()` to generate a `.html` document for each Iris species.

```
reports %>%
  select(output_file, params) %>%
  mutate(output_file = here::here(output_file)) %>%
  purrr::pwalk(rmarkdown::render, input = "04-rmarkdown-materials/04-rmarkdown-parameter.Rmd")
```

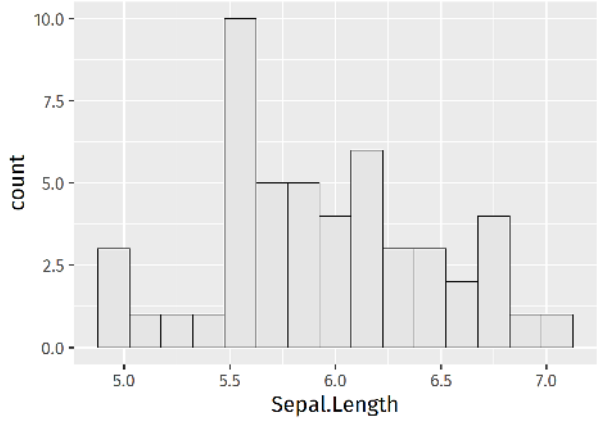
```
##
  |
  |                                                                      |   0%
  |
  |.....................                                                 |  25%
##   ordinary text without R code
##
##
  |
  |.....................................                                 |  50%
## label: iris-setup (with options)
## List of 3
##  $ include: logi FALSE
##  $ message: logi FALSE
##  $ warning: logi FALSE
##
##
  |
  |.................................................                     |  75%
```

Parametrized reports

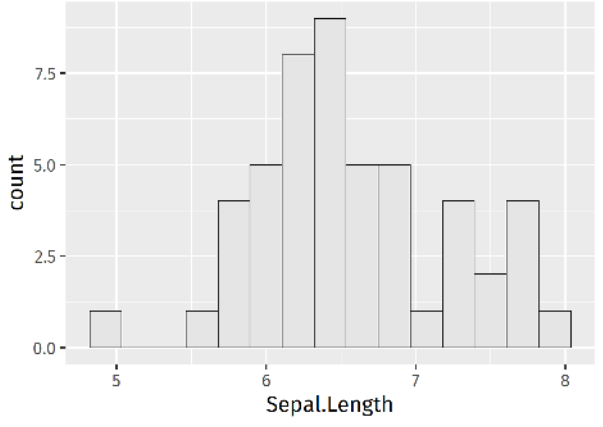
```
ggplot(irs,aes(Sepal.Length)) +
  geom_histogram(bins=15,
                 color="black",
                 fill= "gray90")
```




```
ggplot(irs,aes(Sepal.Length)) +
  geom_histogram(bins=15,
                 color="black",
                 fill= "gray90")
```




```
ggplot(irs,aes(Sepal.Length)) +
  geom_histogram(bins=15,
                 color="black",
                 fill= "gray90")
```



More information in R Markdown: The Definitive Guide. Chapter Parameterized reports.

R Markdown (index.html)

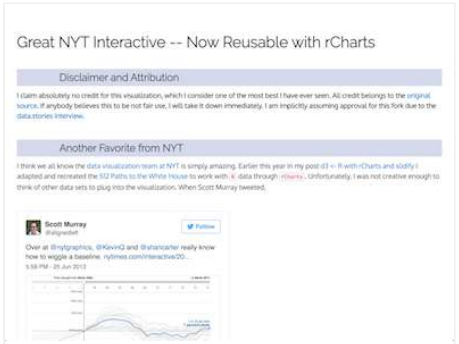
from (https://www.rstudio.com/) Get Started (lesson-1.html) Gallery (gallery.html) Formats (formats.html) Articles (articles.html) Book (https

Gallery

Check out the range of outputs and formats you can create using R Markdown.

Documents

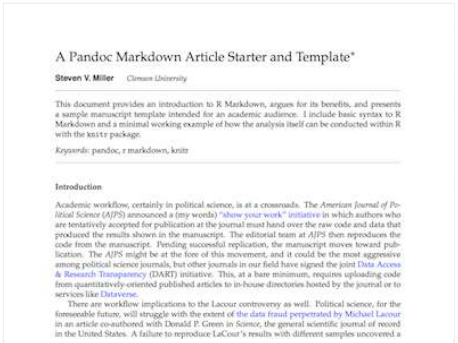
With R Markdown, you write a single .Rmd file and then use it to render finished output in a variety of formats.



(http://timelyportfolio.github.io/rCharts_nyt_home_price/)

HTML (http://timelyportfolio.github.io/rCharts_nyt_home_price/)

HTML documents for web publishing.



(https://github.com/svmiller/svm-r-markdown-templates/blob/master/article-example/svm-rmarkdown-article-example.pdf)

Microsoft Word (https://github.com/svmiller/svm-r-markdown-templates/tree/master/word-example)

(https://rstudio.github.io/tufte/)

Handouts (https://rstudio.github.io/tufte/)

Tufte styled documents for handouts.
Example Code

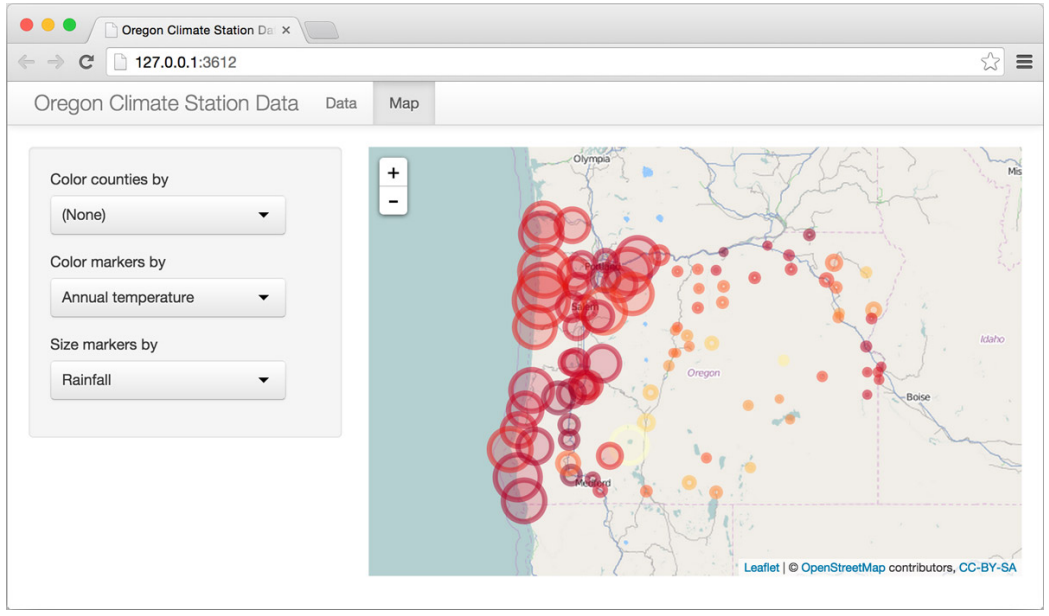
htmlwidgets: http://www.htmlwidgets.org/index.html

Bring the best of JavaScript data visualization to R

Use JavaScript visualization libraries **at the R console**, just like plots

Embed widgets in **R Markdown** documents and **Shiny** web applications

Develop new widgets using a framework that seamlessly bridges R and JavaScript



At the R console In R Markdown docs In Shiny apps

Widgets in action



flexdashboards: https://rmarkdown.rstudio.com/flexdashboard/

t-SNE algorithm

xaringan presentations: https://slides.yihui.org/xaringan/

Presentation Ninja

with xaringan

Yihui Xie

RStudio, PBC

2016/12/12 (updated: 2020-12-21)

1 / 39

blogdown websites: https://bookdown.org/yihui/blogdown/

Example blogdown website: https://alison.rbind.io/

I am a PhD data scientist and professional educator at RStudio. I am an international keynote speaker, award-winning educator, and co-author of the book *blogdown: Creating Websites with R Markdown*. I love creating unique platforms for sharing knowledge and data-driven insights, from websites to presentations and everything in between. I am known for being a compassionate leader and enthusiastic collaborator, and for making user-facing experiences that engage and delight.

Alison Hill

Interests

- Knowledge sharing
- Mentoring
- Data analysis
- Data visualization

Education

PhD in Developmental Psychology & Quantitative Methods, 2008
Vanderbilt University

MSc in Developmental Psychology, 2005

bookdown: https://bookdown.org/

Home About Documentation Books Tags Authors Contest Log in

BOOKDOWN

Write HTML, PDF, ePub, and Kindle books with R Markdown

The **bookdown** package is an open-source R package that facilitates writing books and long-form articles/reports with R Markdown. Features include:

- Generate printer-ready books and ebooks from R Markdown documents.
- A markup language easier to learn than LaTeX, and to write elements such as section headers, lists, quotes, figures, tables, and citations.
- Multiple choices of output formats: PDF, LaTeX, HTML, EPUB, and Word.
- Possibility of including dynamic graphics and interactive applications (HTML widgets and Shiny apps).
- Support a wide range of languages: R, C/C++, Python, Fortran, Julia, Shell scripts, and SQL, etc.
- LaTeX equations, theorems, and proofs work for all output formats.
- Can be published to GitHub, bookdown.org, and any web servers.
- Integrated with the RStudio IDE.
- One-click publishing to https://bookdown.org.

Below is a list of featured books. For a full list, please see the archive page. For the full documentation of the **bookdown** package, please see the free online book *bookdown: Authoring Books and Technical Documents with R Markdown*.

rticles: https://github.com/rstudio/rticles

The **rticles** package provides a suite of custom R Markdown LaTeX formats and templates for various formats, including:

- ACM articles
- ACS articles
- AMS articles
- Elsevier journal submissions
- IEEE Transaction journal submissions
- Sage journal submissions
- Springer journal submissions
- Frontiers articles
- Taylor & Francis articles

..and more.

distill: https://rstudio.github.io/distill/

Distill for R Markdown

Scientific and technical writing, native to the web

AUTHORS

JJ Allaire
Rich Iannone
Alison Presmanes Hill
Yihui Xie

AFFILIATIONS

RStudio
RStudio
RStudio
RStudio

PUBLISHED

Sept. 10, 2018

CITATION

Allaire, et al., 2018

Contents

Distill for R Markdown is a web publishing format optimized for scientific and technical communication. Distill articles include:

Session info

```
##  setting  value
##  version  R version 4.0.4 (2021-02-15)
##  os       Windows 10 x64
##  system   x86_64, mingw32
##  ui       RTerm
##  language EN
##  collate  English_United States.1252
##  ctype    English_United States.1252
##  tz       Europe/Berlin
##  date     2021-04-19
```

package	version	date	source
countdown	0.3.5	2021-03-16	Github (gadenbuie/countdown@a544fa4)
dplyr	1.0.5	2021-03-05	CRAN (R 4.0.4)
DT	0.17	2021-01-06	CRAN (R 4.0.3)
ggplot2	3.3.3	2020-12-30	CRAN (R 4.0.3)


Thank you! Questions?
Photo courtesy of Stefan Berger