

Today

1. Final project guidelines
2. Article discussion (Klein; Goodman)
3. Reproducible workflow lecture
4. Break
5. R Markdown & package control tutorials
6. Go over function homework
7. Work on R Markdown hw (if time)

Final project dates/components

- Project due 3/17 at 1:30pm on Canvas
 - Link to your Github repo
- Week 10 “hackathon”
 - Bring code that you are working on
 - Share/discuss ideas about changes you want to make but are stuck on
 - Help getting things set up on Github
- No presentation!

Github repo “before” and “after”

- Take your project files and put them up as is
- Commit & push the initial files so that we can see the project at its start
- Continue to commit and push changes so that we can see what you changed
- If you made significant changes since the class started, that’s OK (just let us know)

What your repo should contain

- Well-organized files/folders with data and cleaning/exploration scripts
- An R Markdown report where you describe the changes you made
 - Can use code chunks to show before/after code
 - Can embed plots to show EDA
 - More about this in today's tutorial

Your changes should include at least one of each

- A custom function to reduce code repetition and/or split long scripts into more man- ageable files
- Some form of automation (map, for loop, across, read_csv) to replace repetitive code or manual data entry/copy/paste
- Plotted graphs or created data checks to explore the data
- Improved documentation of the project, file organization, and/or readability of the code

Public vs. private sharing

- Please be careful not to make your repo public if it contains information that can't be shared
 - Invite Jake and I to a private repo instead
- Other options
 - Create fake data to put on Github to demonstrate your workflow, and share real, de-identified summary data

Article discussion

PSYC 259: Principles of Data Science

Week 8: Sharing & Reproducibility

Research transparency (Klein et al. 2018)

- Replicability - find same result when rerunning the study (ideally 3rd party)
- Analytic reproducibility - statistical analysis can be confirmed by a 3rd party
- Analytic robustness - result is stable despite changes analytical procedures

Sharing is key to allow replication and reproducibility

- Study protocol, materials
 - Allows replication
- Metadata, raw/summarized data, scripts
 - Allows analytical reproducibility checks by other researchers
 - 3rd party researchers can tweak workflow/analytical approach to determine robustness

Restrictions on sharing

- Sensitive and/or identifiable
 - Depends on IRB/local/institutional guidelines
 - Audio/video CAN be shared with IRB approval
- De-identified data can usually be shared, but make all efforts to anonymize
 - Especially in a small sample, combinations of birthdate, ZIP code, sex, race/ethnicity could identify participants
 - Consent should indicate what might be shared and with whom; sharing can be optional

Where to share data?

- Things to consider
 - Permanent DOI
 - Linking to preprint/published paper
 - Used commonly by the field
 - Access control
 - Archive size/duration
 - Embargo period (private -> public option)
 - Include an informative readme, clear file/folder organization

Common storage sites

- Open Science Foundation
 - Integrates different types of cloud storage
 - Allows for document registration
- Github (+ Zenodo)
 - Better for code and smaller files
- Databrary
 - Specialized for video
 - Flexible access controls (by participant)

How to make your analysis reproducible

- Documentation helps, but isn't everything
- Use open source software (e.g., R, Python)
- Avoid absolute file paths (*here* package)
- Set random number seed
- Document hardware and software versions

Document your R and package environment for posterity

```
Sys.time()
```

```
## [1] "2021-02-24 08:57:52 PST"
```

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Catalina 10.15.7
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRblas.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats      graphics  grDevices  utils      datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] digest_0.6.27      R6_2.5.0           jsonlite_1.7.2     magrittr_2.0.1
##  [5] evaluate_0.14      rlang_0.4.10       stringi_1.5.3      jquerylib_0.1.3
##  [9] bslib_0.2.4        rmarkdown_2.7      tools_4.0.3        stringr_1.4.0
## [13] xfun_0.21          yaml_2.2.1         compiler_4.0.3     htmltools_0.5.1.1
## [17] knitr_1.31         sass_0.3.1
```

Groundhog package

If we wanted to load a single package on a single date, instead of running:

```
library( "dplyr" )
```

we would instead run:

```
groundhog.library( "dplyr", "2020-01-01" )
```


Host your project on a cloud computing environment

- Lets users open and run your project without having to transfer files to their local computer; preserves software versions
 - RStudio Cloud
 - CodeOcean
 - Binder
 - ReproZip

CodeOcean example

+

+

↑

Core Files ?

▶ metadata659 B✔

▶ environment919 B✔

▼ code24.55 KB✔

calibration_error.R1.54 KB✔

LICENSE1.04 KB✔

lss1_summary_analyses.R8.56 KB✔

lss2_peak_analyses.R2.78 KB✔

lss2_summary_analyses.R10.33 KB✔

run287 B✔

▼ data Manage Datasets2.43 MB⊗

calibration_LSS1.csv11.4 KB⊗

calibration_LSS2.csv5.49 KB⊗

LICENSE18.88 KB⊗

summary_stats_LSS1.csv12.12 KB⊗

summary_stats_LSS2_peaks....2.37 MB⊗

summary_stats_LSS2.csv15.46 KB⊗

.gitignore7 B✔

Results ?

▼ results

Your files will appear in the timeline.

[View latest results](#)

Other Files ?

readme.txt738 B✔

Published

Adapting the coordination of eyes and head for task-specific visual exploration in the... (John Franchak, Brianna McGee & Gabriell...)

Edit Original

Files

App Panel

Reproducibility

readme.txt

1 data files and analyses for Franchak, McGee, & Blanch QJEP "Adapting the coordination of eyes and head for task-specific visual exploration in the context of locomotion"

2

3 Files tagged "LSS1" correspond to Study 1 (Eyes only), "LSS2" corresponds to Study 2 (eyes, head, and gaze)

4

5 There are 4 R scripts.

6 1) Calibration error reports summary statistics for the calibration error for both Study 1 and Study 2

7 2) lss1_summary_analyses reports t-tests for spread, speed, and walking GPS data for Study 1

8 3) lss2_summary_analyses reports LMMs (eye vs head) for spread and speed, and t-tests for walking GPS data for Study 1

9 4) lss2_peak_analyses reports the head contribution analysis

10

11 Each R script produces the figures used in the publication.

Reproducible Run

or launch a cloud workstation

lab

R Studio

jupyter

>

Shiny

Timeline

Nov 5, 2020

Published Version 1.0

Currently viewing

Author ran Nov 5, 2020 00:00:22

▼ Published Result

head_contribution.pdf4.92 KB

lss1_position_sd.pdf11.65 KB

lss1_speed.pdf11.55 KB

lss1_walking_stats_c...24.85 KB

lss2_position_sd.pdf14.61 KB

lss2_speed.pdf11.31 KB

lss2_walking_stats_c...14.92 KB

output24.55 KB

John Franchak committed Nov 5, 2020

Version 1.0

Nov 5, 2020

Created capsule

Share dynamic documents

- Instead of comments, use R Markdown to mix prose, code, tables, and figures
- From within RStudio, you can render the document in different output formats: html, word, pdf, slides, and more
- The *papaja* package can even render your document as an APA manuscript

R Markdown components

- YAML header – metadata (title, author), output format and options
 - Always comes first
 - Does not use R syntax, pay attention to spacing

```
1 ---  
2 title: "Exploratory Data Analysis"  
3 author: "John Franchak"  
4 date: "2/19/2021"  
5 output:  
6   html_document:  
7     toc: true  
8     toc_float: true  
9     code_folding: show  
10 ---
```

R Markdown components

- Markdown text
 - Special characters identify headings, lists, bold, italic, etc as opposed to using toolbars (i.e. Word)

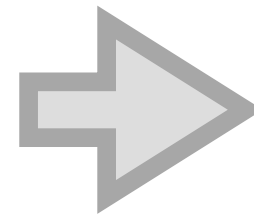
Level 3 heading

Lists are easy to create.

- * Top level of an unordered list
 - * 2nd level of list using tab
 - * 2nd level again
- * Back to top level

1. Numbered lists work the same way
1. But just keep using "1".
1. Rmd will figure it out for you
1. See!

Hyperlinks are easy to embed in text, like so: [[click here](https://padlab.ucr.edu)](https://padlab.ucr.edu). Or if you want the URL to display like this [<https://padlab.ucr.edu>](https://padlab.ucr.edu).



Level 3 heading

Lists are easy to create.

- Top level of an unordered list
 - 2nd level of list using tab
 - 2nd level again
- Back to top level

1. Numbered lists work the same way
2. But just keep using "1".
3. Rmd will figure it out for you
4. See!

Hyperlinks are easy to embed in text, like so: [click here](https://padlab.ucr.edu). Or if you want the URL to display you can do so like this <https://padlab.ucr.edu>.

Pandoc's Markdown

Write with syntax on the left to create effect on right (after render)

Plain text
End a line with two spaces
to start a new paragraph.
italics and **bold**
``verbatim code``
sub/superscript²~
~~strikethrough~~
escaped: * _ \\
endash: --, emdash: ---
equation: $A = \pi * r^2$
equation block:
$$E = mc^2$$

> block quote
Header1 {#anchor}
Header 2 {#css_id}
Header 3 {.css_class}
Header 4
Header 5
Header 6
<!--Text comment-->
\textbf{Tex ignored in HTML}
HTML ignored in pdfs
<http://www.rstudio.com>
[link](www.rstudio.com)
Jump to [Header 1](#anchor)
image:
![Caption](smallorb.png)

Plain text
End a line with two spaces
to start a new paragraph.
italics and **bold**
`verbatim code`
sub/superscript²
~~strikethrough~~
escaped: * _ \
endash: –, emdash: —
equation: $A = \pi * r^2$
equation block:
$$E = mc^2$$

block quote

Header1


Header 2

Header 3

Header 4

Header 5

Header 6

HTML ignored in pdfs
<http://www.rstudio.com>
link
Jump to [Header 1](#)
image:

Caption

- * unordered list
 - + sub-item 1
 - + sub-item 2
 - sub-sub-item 1
- * item 2
 - Continued (indent 4 spaces)

1. ordered list
2. item 2
 - i) sub-item 1
 - A. sub-sub-item 1

(@) A list whose numbering
continues after

(@) an interruption

Term 1

: Definition 1

Right	Left	Default	Center
12	12	12	12
123	123	123	123
1	1	1	1

- slide bullet 1
- slide bullet 2

(>- to have bullets appear on click)

horizontal rule/slide break:

A footnote ^[^1]

^[^1]: Here is the footnote.

- unordered list
 - sub-item 1
 - sub-item 2
 - sub-sub-item 1
- item 2
 - Continued (indent 4 spaces)

1. ordered list
2. item 2
 - i. sub-item 1
 - A. sub-sub-item 1

1. A list whose numbering
continues after

2. an interruption

Term 1

Definition 1

Right	Left	Default	Center
12	12	12	12
123	123	123	123
1	1	1	1

- slide bullet 1
- slide bullet 2

(>- to have bullets appear on click)

horizontal rule/slide break:

A footnote ¹

1. Here is the footnote. ↩

R Markdown components

- Code chunks

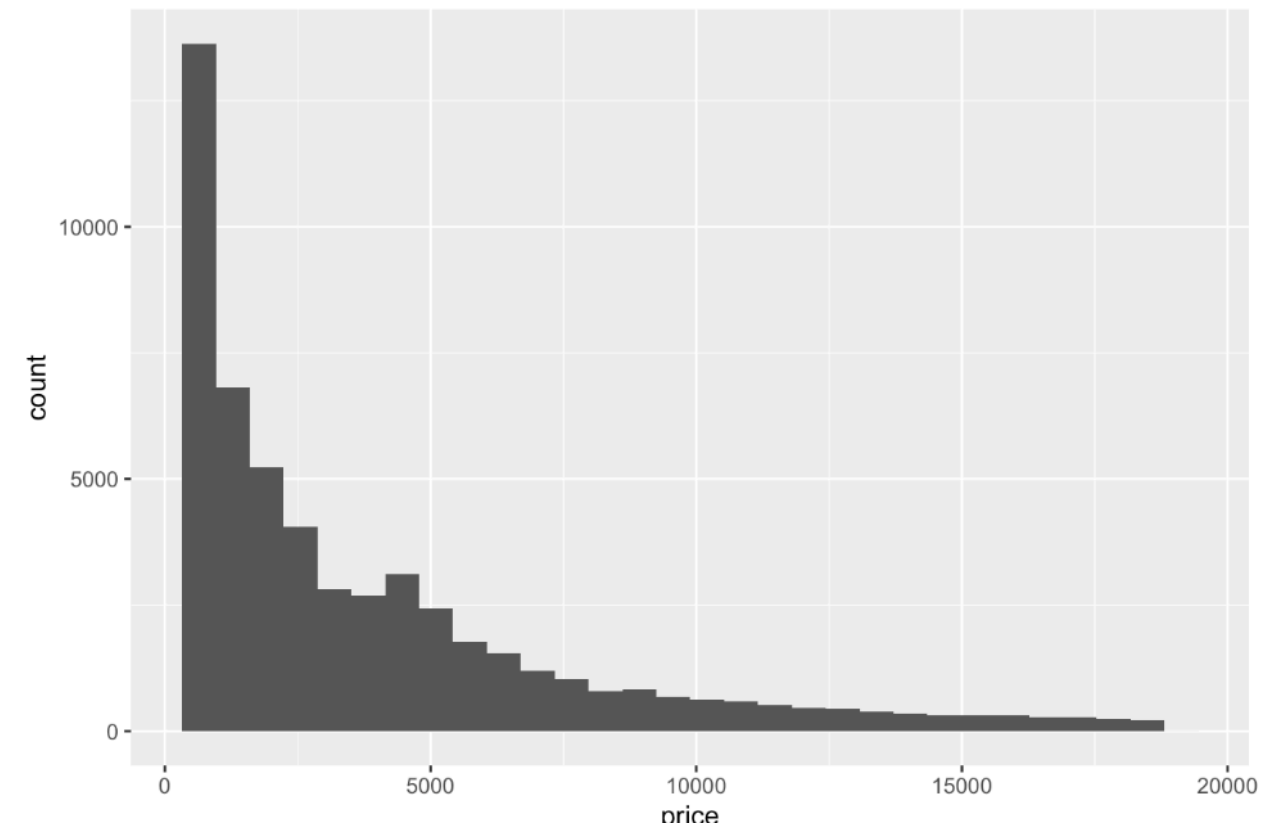
- Lines in between ```{r}` and ````
- Settings to tell R to show code/warnings/results/plots

Figures

A code chunk that produces a figure will insert that figure into the output document.

```
ds <- diamonds
ggplot(ds, aes(x = price)) + geom_histogram() + ggtitle(
  "A histogram")
```

A histogram



Figures

A code chunk that produces a figure will insert that figure into the output document.

```
```${r warning = FALSE, message=FALSE}
```

```
ds <- diamonds
```

```
ggplot(ds, aes(x = price)) + geom_histogram() +
```

```
ggtitle("A histogram")
```

```
```
```

R Markdown :: CHEAT SHEET

What is R Markdown?

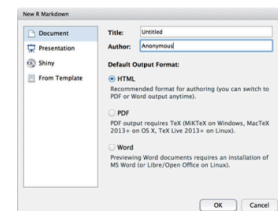


.Rmd files • An R Markdown (.Rmd) file is a record of your research. It contains the code that a scientist needs to reproduce your work along with the narration that a reader needs to understand your work.

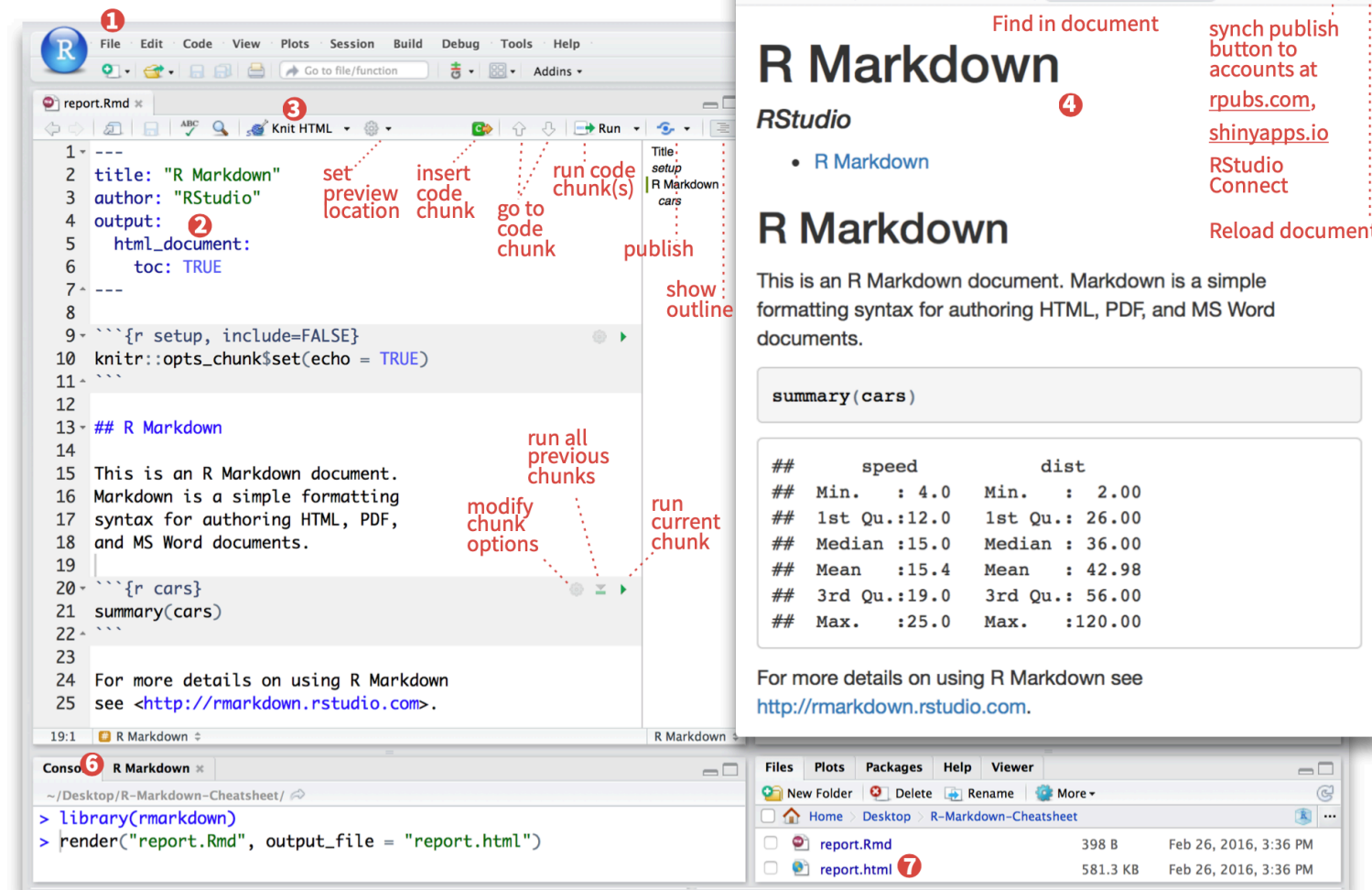
Reproducible Research • At the click of a button, or the type of a command, you can rerun the code in an R Markdown file to reproduce your work and export the results as a finished report.

Dynamic Documents • You can choose to export the finished report in a variety of formats, including html, pdf, MS Word, or RTF documents; html or pdf based slides, Notebooks, and more.

Workflow



- 1 **Open a new .Rmd file** at File ► New File ► R Markdown. Use the wizard that opens to pre-populate the file with a template
- 2 **Write document** by editing template
- 3 **Knit document to create report**; use knit button or `render()` to knit
- 4 **Preview Output** in IDE window
- 5 **Publish** (optional) to web server
- 6 **Examine build log** in R Markdown console
- 7 **Use output file** that is saved along side .Rmd



render

Use `rmarkdown::render()` to render/knit at cmd line. Important args:

input - file to render
output_format

output_options - List of render options (as in YAML)

output_file
output_dir

params - list of params to use

envir - environment to evaluate code chunks in

encoding - of input file

Embed code with knitr syntax

INLINE CODE

Insert with ``r <code>``. Results appear as text without code.

Built with ``r getRversion()`` ➔ Built with 3.2.3

CODE CHUNKS

One or more lines surrounded with ````\{r\}` and `````. Place chunk options within curly braces, after `r`. Insert with

````\{r echo=TRUE\}`  
`getRversion()`  
`````

`getRversion()`
`## [1] '3.2.3'`

GLOBAL OPTIONS

Set with `knitr::opts_chunk$set()`, e.g.

````\{r include=FALSE\}`  
`knitr::opts_chunk$set(echo = TRUE)`  
`````

IMPORTANT CHUNK OPTIONS

cache - cache results for future knits (default = FALSE)

cache.path - directory to save cached results in (default = "cache/")

child - file(s) to knit and then include (default = 'R')

collapse - collapse all output into single block (default = FALSE)

comment - prefix for each line of results (default = "##")

dependson - chunk dependencies for caching (default = NULL)

echo - Display code in output document (default = TRUE)

engine - code language used in chunk (default = 'R')

error - Display error messages in doc (TRUE) or stop render when errors occur (FALSE) (default = FALSE)

eval - Run code in chunk (default = TRUE)

fig.align - 'left', 'right', or 'center' (default = 'default')

fig.cap - figure caption as character string (default = NULL)

fig.height, **fig.width** - Dimensions of plots in inches

highlight - highlight source code (default = TRUE)

include - Include chunk in doc after running (default = TRUE)

message - display code messages in document (default = TRUE)

results (default = 'markup')

'asis' - passthrough results

'hide' - do not display results

'hold' - put all results below all code

tidy - tidy code for display (default = FALSE)

warning - display code warnings in document (default = TRUE)

.rmd Structure

rmarkdown

YAML Header
Optional section of render (e.g. pandoc) options written as key:value pairs (YAML).

At start of file

Between lines of ---

Text

Narration formatted with markdown, mixed with:

Code Chunks

Chunks of embedded code. Each chunk:

Begins with ````\{r\}`

ends with `````

R Markdown will run the code and append the results to the doc.

It will use the location of the .Rmd file as the **working directory**

Parameters

Parameterize your documents to reuse with new inputs (e.g., data, values, etc.)

params:
n: 100
d: !r Sys.Date()

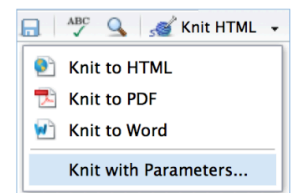
1. **Add parameters** • Create and set parameters in the header as sub-values of params

2. **Call parameters** • Call parameter values in code as `params$<name>`

Today's date
is `r params\$d`

3. **Set parameters** • Set values with Knit with parameters or the params argument of render():

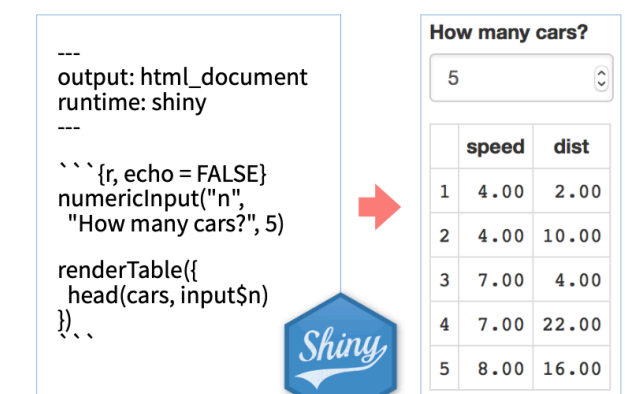
`render("doc.Rmd", params = list(n = 1, d = as.Date("2015-01-01")))`



Interactive Documents

Turn your report into an interactive Shiny document in 4 steps

1. Add runtime: shiny to the YAML header.
2. Call Shiny input functions to embed input objects.
3. Call Shiny render functions to embed reactive output.
4. Render w `rmarkdown::run` or click Run Document in RStudio IDE



Embed a complete app into your document with `shiny::shinyAppDir()`

Publish on RStudio Connect, to share R Markdown documents securely, schedule automatic

Break

R Markdown tutorial

Last homework (due week 10)

- Take a heavily-commented R script and turn it into an R Markdown report
- The files are from our in-class EDA group exercise
- Create a new Rmd file and transfer over the comments/code to format them correctly
- An example output is provided