Teaching students reproducibility

Amelia McNamara @AmeliaMN
Current: Program in Statistical & Data Sciences, Smith College
Fall 2018: Department of Computer & Information Sciences, University of St Thomas

Reproducible research

"[T]he code and data are assembled in a way so that another group can re-create all of the results (e.g., the figures in a paper). Adopting a workflow that will make your results reproducible will ultimately make your life easier; if a problem (or question) arises somewhere down the line, it will be much easier to correct (or explain)."
- Karl Broman

http://kbroman.org/steps2rr/

Excel

- 750 million users
- Reactive programming!
- Combines data input, wrangling, modeling, visualization, output in one document
- Not reproducible
- For more, see Felienne Herman's talk, Functional programming in Excel

Gene name errors are widespread in the scientific literature

Mark Ziemann, Yotam Eren and Assam El-Osta. Genome Biology, 2016.
https://genomebiology.biomedcentral.com/articles/10.1186/s13059-016-1044-7

"We downloaded and screened supplementary files from 18 journals published between 2005 and 2015 using a suite of shell scripts. […]

Of the selected journals, the proportion of published articles with Excel files containing gene lists that are affected by gene name errors is 19.6 %. […]

Journals that had the highest proportion of papers with affected supplementary files were Nucleic Acids Research, Genome Biology, Nature Genetics, Genome Research, Genes and Development and Nature (>20 %)."

Perspectives: Teaching chemists to code

Charles J. Weiss. Volume 95 Issue 35, 2017.
https://cen.acs.org/articles/95/i35/Perspectives-spreadsheets-programming.html

"Spreadsheets are a standard tool in chemistry for simple tasks such as data analysis and graphing. Chemistry students are often introduced to spreadsheets their first year of college, if not earlier, and those who continue on to do research will likely use them as a means of handling and visualizing data. […]

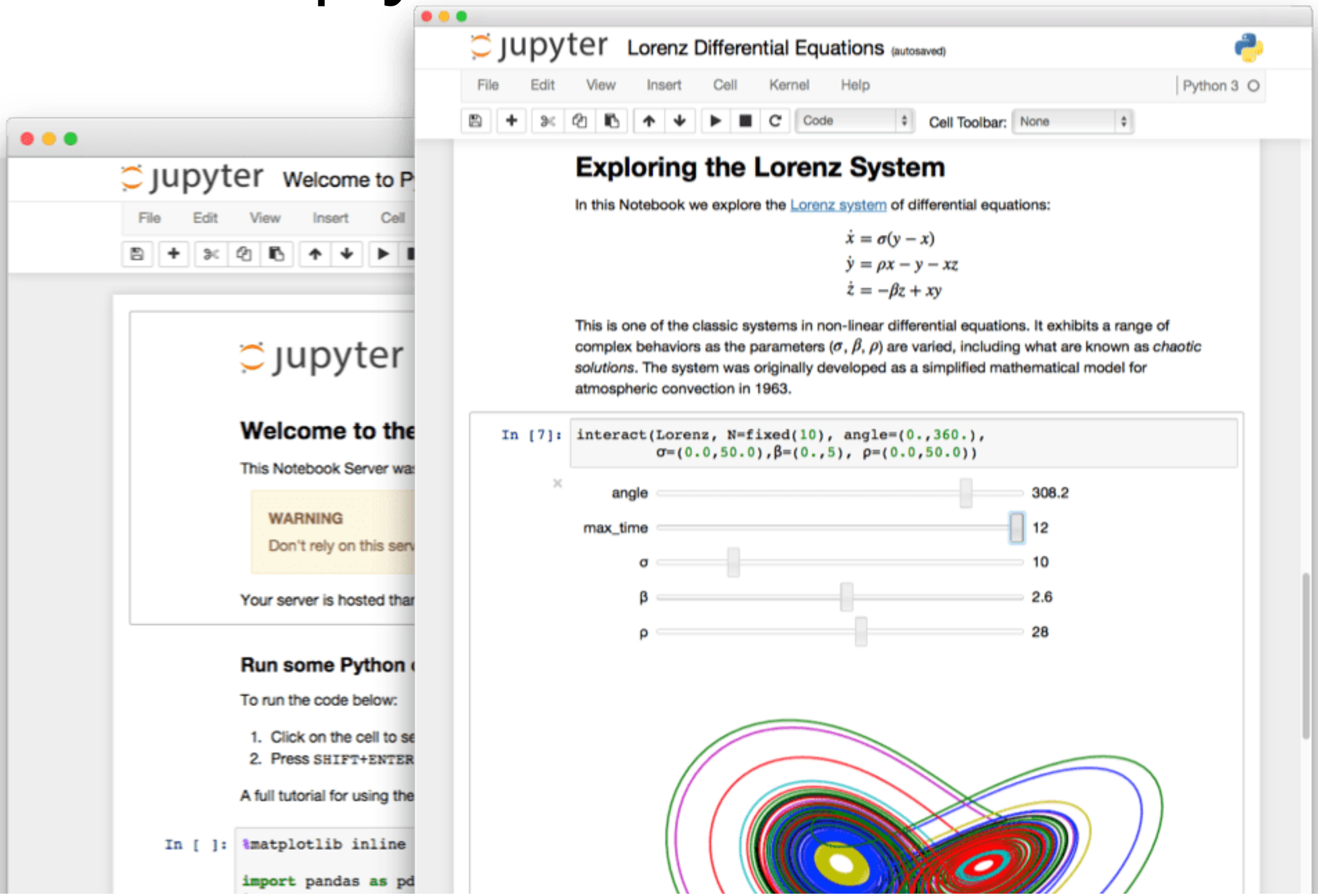
Software better geared to those earning chemistry degrees or conducting research is readily available. Common examples include MATLAB, Python's SciPy stack, and GNU Octave."

Scientific computing: Code alert

Monya Baker. Nature, 541. 2017.
https://www.nature.com/naturejobs/science/articles/10.1038/nj7638-563a

"Andrew Durso can vouch for those upsides. The ecology graduate student at Utah State University in Logan started his research career using programs with graphical interfaces. Whenever he clicked buttons or checked boxes on a computer screen, he would try to write those steps down on paper in case he wanted to redo an analysis — a strategy that was both time-consuming and unreliable. "

Jupyter notebooks



http://jupyter.org/

Jupyter notebooks

- Grew out of iPython notebook
- Works best with Python, but can be used with many languages, including R, Julia, and Scala
- More for working than publishing

http://jupyter.org/

RMarkdown

~/Documents/rmarkdown - gh-pages - RStudio

Go to file/function | Addins | rmarkdown

1-example.Rmd

Knit | Run

```
1  ---
2  title: "Viridis Demo"
3  output: html_document
4  ---
5
6  ```{r include = FALSE}
7  library(viridis)
8  ```
9
10 The code below demonstrates two color palettes in the
   [viridis](https://github.com/sjmgarnier/viridis) package. Each
   plot displays a contour map of the Maunga Whau volcano in
   Auckland, New Zealand.
11
12 ## Viridis colors
13
14 ```{r}
15 image(volcano, col = viridis(200))
16 ```
17
18 ## Magma colors
19
20 ```{r}
21 image(volcano, col = viridis(200, option = "A"))
22 ```
23
```

1:1 | Viridis Demo | R Markdown

Console

Environment | History | Build | Git

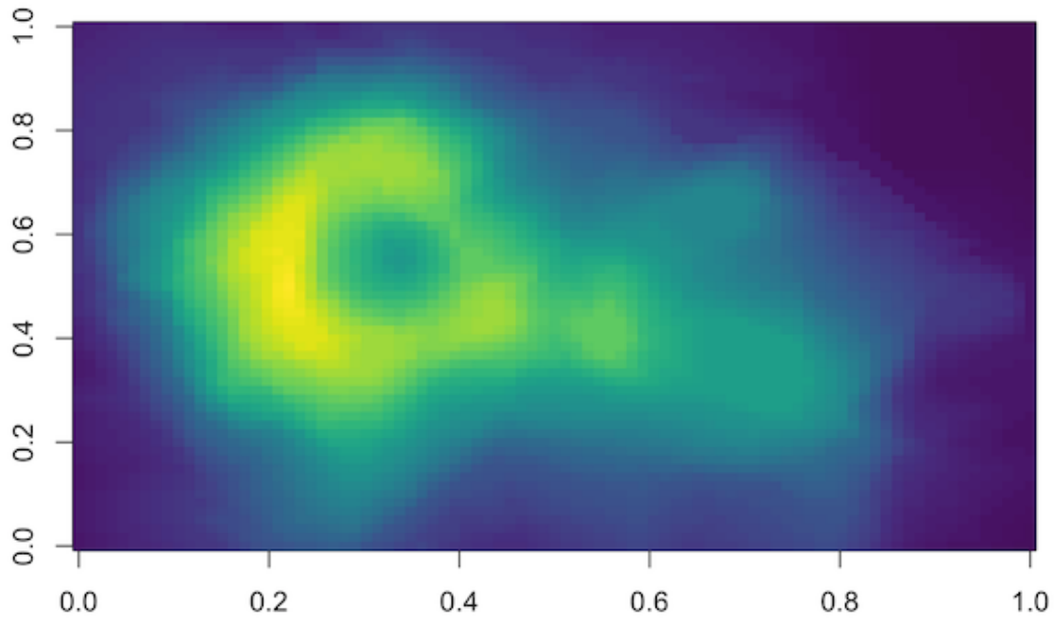
Files | Plots | Packages | Help | Viewer

Viridis Demo

The code below demonstrates two color palettes in the viridis package. Each plot displays a contour map of the Maunga Whau volcano in Auckland, New Zealand.

Viridis colors

```
image(volcano, col = viridis(200))
```

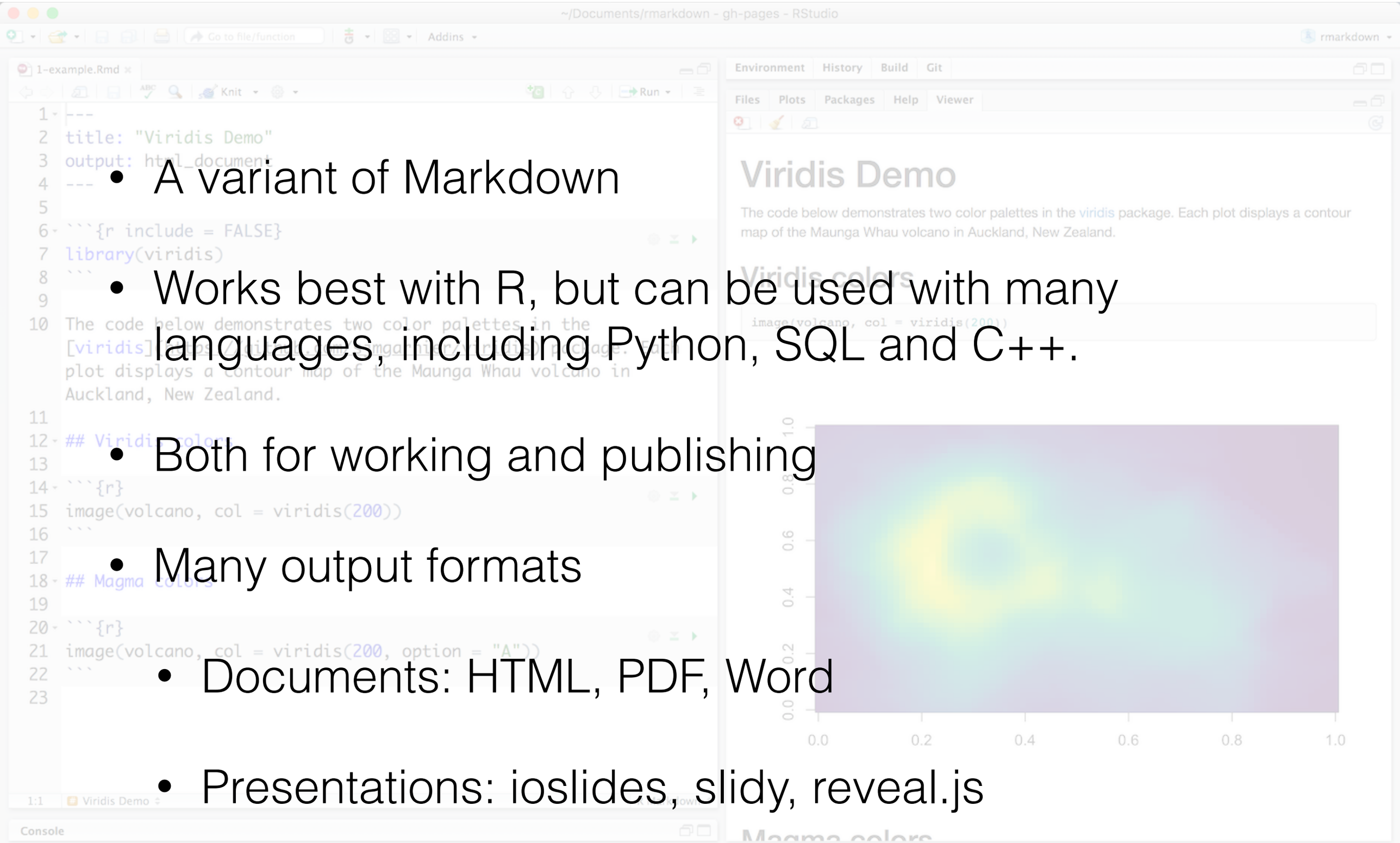


Magma colors

http://rmarkdown.rstudio.com/

RMarkdown

- A variant of Markdown
- Works best with R, but can be used with many languages, including Python, SQL and C++.
- Both for working and publishing
- Many output formats
- Documents: HTML, PDF, Word
- Presentations: ioslides, slidy, reveal.js



Our path to better science in less time using open data science tools

Julia Stewart Lowndes, et al. Nature Ecology & Evolution v1.
https://www.nature.com/articles/s41559-017-0160

We thought we were doing reproducible science. For the first global OHI assessment in 2012 we employed an approach to reproducibility that is standard to our field, which focused on scientific methods, not data science methods. Data from nearly one hundred sources were prepared manually—that is, without coding, typically in Microsoft Excel—which included organizing, transforming, rescaling, gap-filling and formatting data. Processing decisions were documented primarily within the Excel files themselves, e-mails, and Microsoft Word documents. We programmatically coded models and meticulously documented their development, (resulting in the 130-page supplemental materials), and upon publication we also made the model inputs (that is, prepared data and metadata) freely available to download. This level of documentation and transparency is beyond the norm for environmental science.

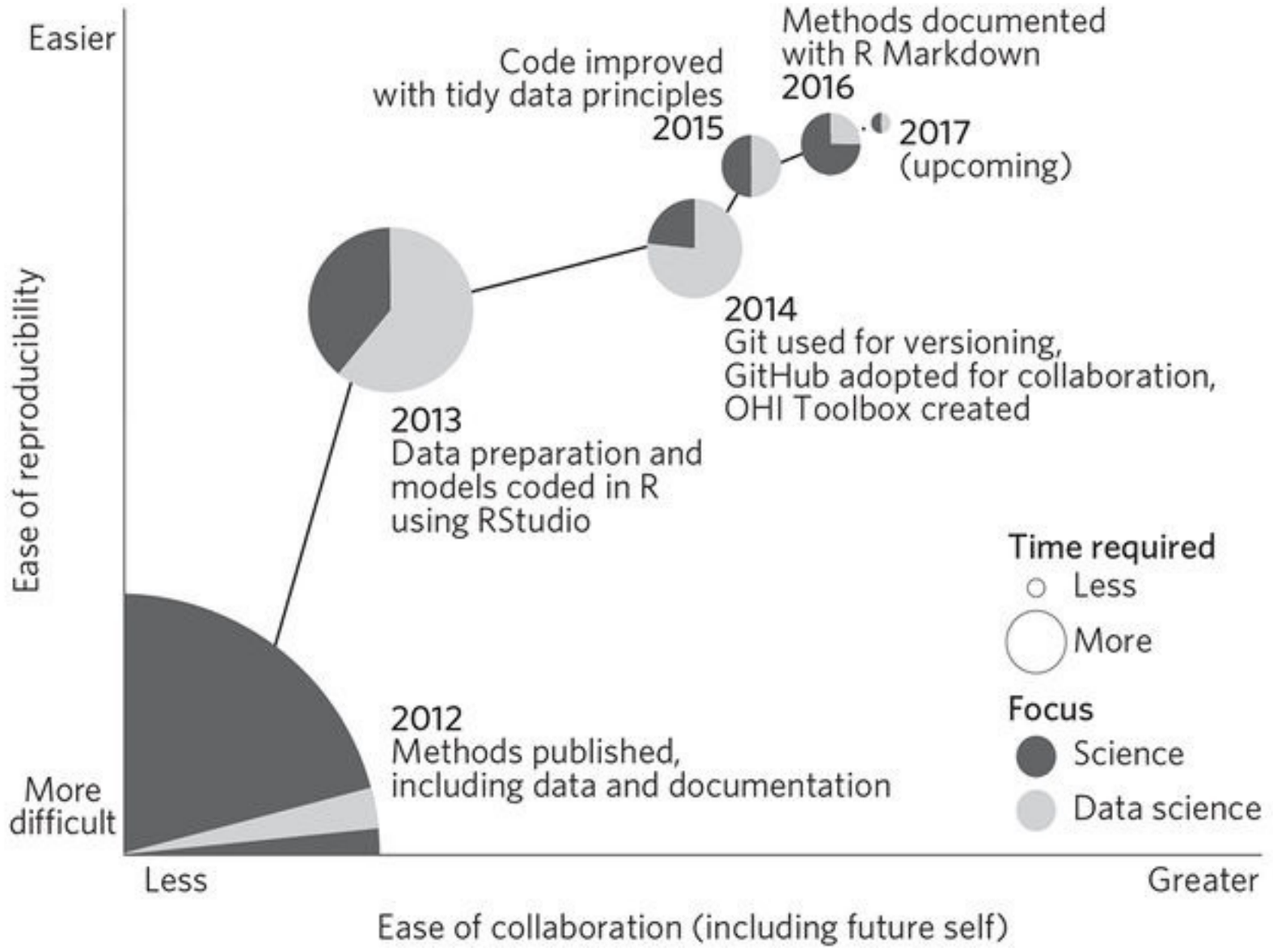
Our path to better science in less time using open data science tools. Julia Stewart Lowndes, et al. Nature Ecology & Evolution v1. https://www.nature.com/articles/s41559-017-0160

We decided to base our work in R and RStudio for coding and visualization, Git for version control, GitHub for collaboration, and a combination of GitHub and RStudio for organization, documentation, project management, online publishing, distribution and communication.

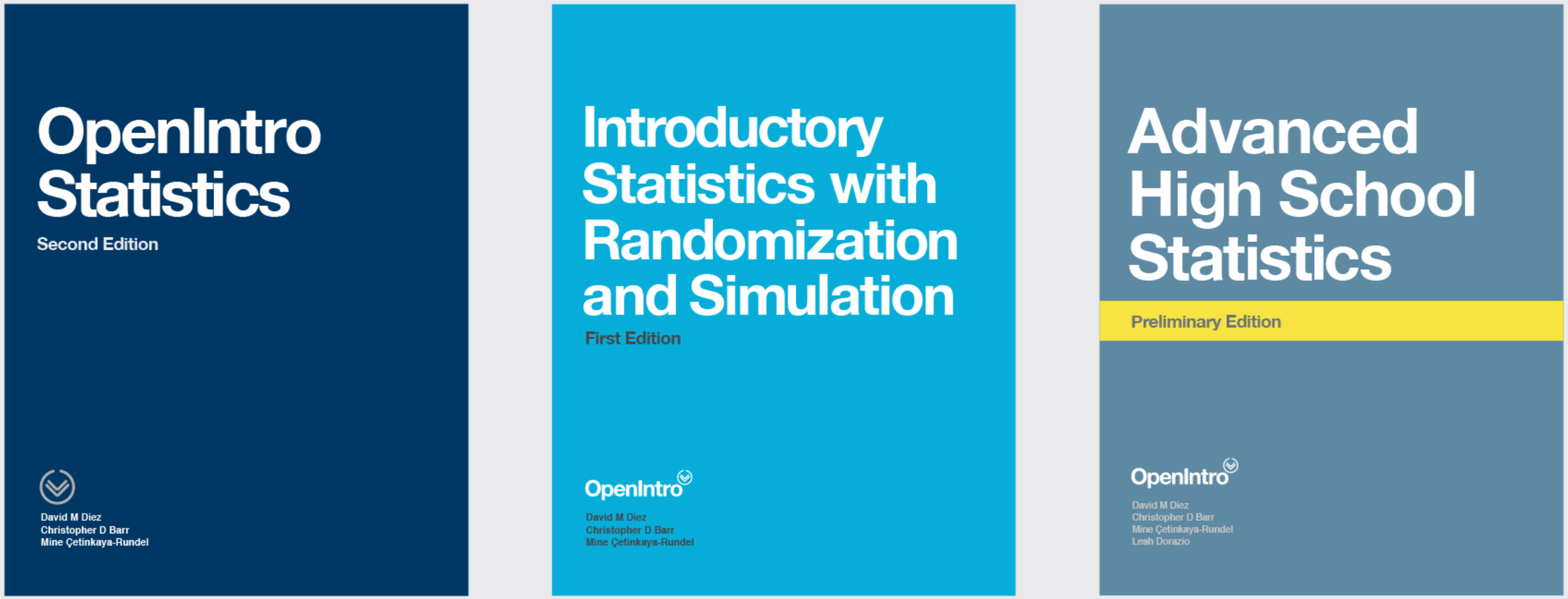
Data preparation: coding and documenting. Our first priority was to code all data preparation, create a standard format for final data layers, and do so using a single programmatic language, R. Code enables us to reproduce the full process of data preparation, from data download to final model inputs, and a single language makes it more practical for our team to learn and contribute collaboratively. We code in R and use RStudio to power our workflow because it has a user-friendly interface and built-in tools useful for coders of all skill levels, and, importantly, it can be configured with Git to directly sync with GitHub online (See 'Collaboration').

Sharing methods and instruction. We use R Markdown not only for data preparation but also for broader communication. R Markdown files can be generated into a wide variety of formatted outputs, including PDFs, slides, Microsoft Word documents, HTML files, books or full websites.

Our path to better science in less time using open data science tools. Julia Stewart Lowndes, et al. Nature Ecology & Evolution v1. https://www.nature.com/articles/s41559-017-0160



Our path to better science in less time using open data science tools. Julia Stewart Lowndes, et al. Nature Ecology & Evolution v1. https://www.nature.com/articles/s41559-017-0160



https://www.openintro.org/stat/textbook.php

README.md

OpenIntro Labs - dplyr and ggplot2

OpenIntro Labs promote the understanding and application of statistics through applied data analysis. Labs are titled based on topic area, which correpond to particular chapters in all three versions of OpenIntro Statistics, a free and open-source textbook. The textbook as well as the html version of the labs can be found at http://www.openintro.org/stat/labs.php.

This repository is a fork of the original base-R labs. It incoperates the 'tidyverse' syntax from the `dplyr` package, and `ggplot` graphics.

We currently support our source files in the RMarkdown (.Rmd) format, which can be output into html format (though output to pdf is also possible). The source files are processed using the knitr package in R, and are easiest to use in RStudio.

It is our hope that these materials are useful for instructors and students of statistics. If you end up developing some interesting variants of these labs or creating new ones, please let us know!

Feedback / collaboration

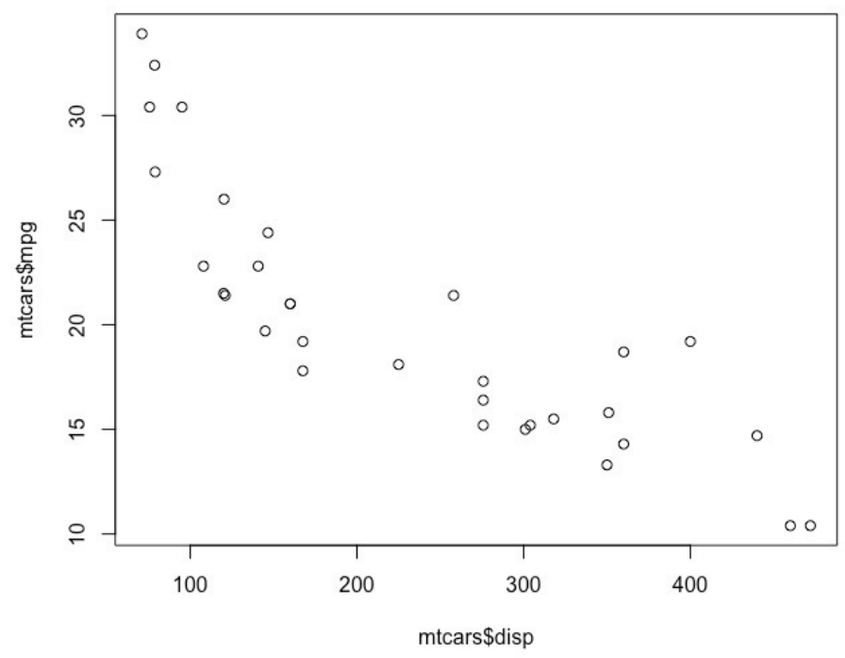
Your feedback is most welcomed! If you have suggestions for minor updates (fixing typos, etc.) please do not hesitate to issue a pull request. If you have ideas for major revamp of a lab (replacing outdated code with modern version, overhaul in pedagogy, etc.) please create an issue so to start the conversation.

https://github.com/OpenIntroOrg/oiLabs-dplyr-ggplot

Three graphics libraries for

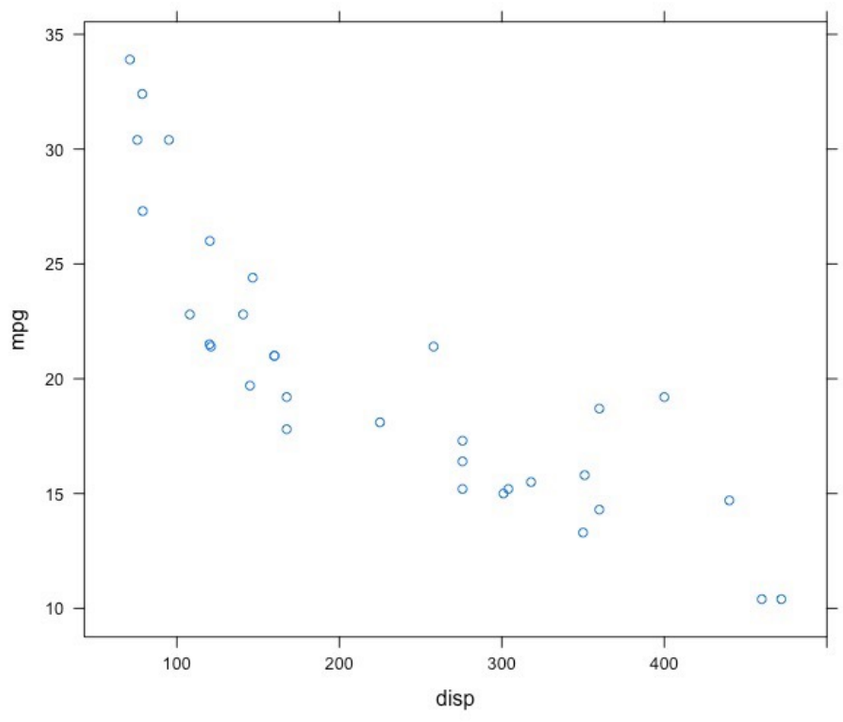
base

```
plot(mtcars$disp, mtcars$mpg)
```



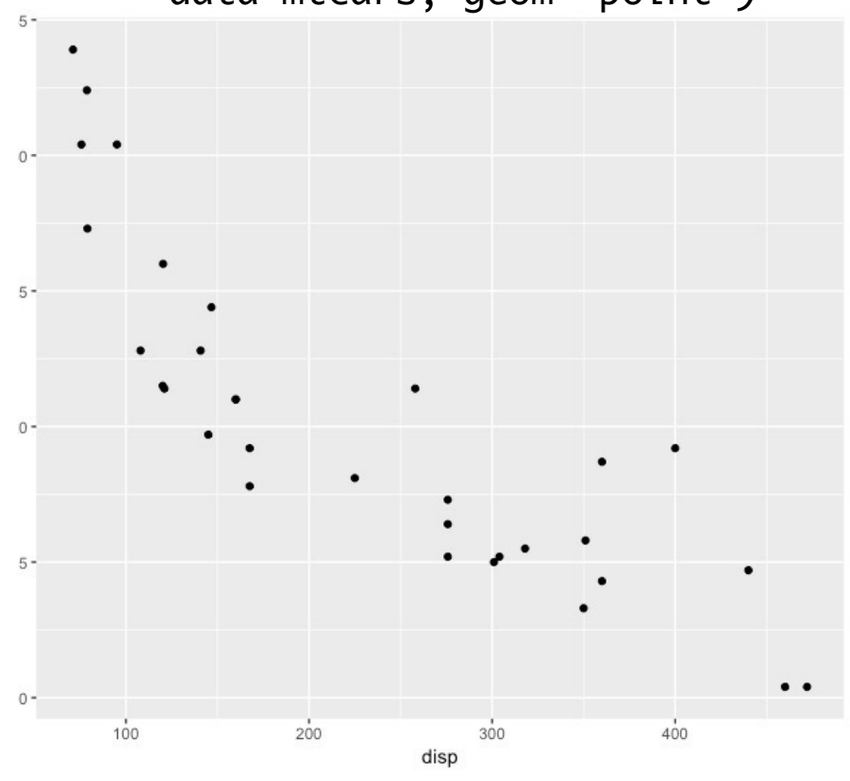
lattice

```
xyplot(mpg~disp, data=mtcars)
```



ggplot2

```
qplot(x=disp, y=mpg,
    data=mtcars, geom="point")
```



R Syntax Comparison : : **CHEAT SHEET**

http://bit.ly/R-syntax-sheet

Dollar sign syntax

```
goal(data$x, data$y)
```

SUMMARY STATISTICS:

one continuous variable:
```
mean(mtcars$mpg)
```

one categorical variable:
```
table(mtcars$cyl)
```

two categorical variables:
```
table(mtcars$cyl, mtcars$am)
```

one continuous, one categorical:
```
mean(mtcars$mpg[mtcars$cyl==4])
mean(mtcars$mpg[mtcars$cyl==6])
mean(mtcars$mpg[mtcars$cyl==8])
```

PLOTTING:

one continuous variable:
```
hist(mtcars$disp)

boxplot(mtcars$disp)
```

one categorical variable:
```
barplot(table(mtcars$cyl))
```

two continuous variables:
```
plot(mtcars$disp, mtcars$mpg)
```

two categorical variables:
```
mosaicplot(table(mtcars$am, mtcars$cyl))
```

one continuous, one categorical:
```
histogram(mtcars$disp[mtcars$cyl==4])
histogram(mtcars$disp[mtcars$cyl==6])
histogram(mtcars$disp[mtcars$cyl==8])

boxplot(mtcars$disp[mtcars$cyl==4])
boxplot(mtcars$disp[mtcars$cyl==6])
boxplot(mtcars$disp[mtcars$cyl==8])
```

WRANGLING:

subsetting:
```
mtcars[mtcars$mpg>30, ]
```

making a new variable:
```
mtcars$efficient[mtcars$mpg>30] <- TRUE
mtcars$efficient[mtcars$mpg<30] <- FALSE
```

Formula syntax

```
goal(y~x|z, data=data, group=w)
```

SUMMARY STATISTICS:

one continuous variable:
```
mosaic::mean(~mpg, data=mtcars)
```

one categorical variable:
```
mosaic::tally(~cyl, data=mtcars)
```

two categorical variables:
```
mosaic::tally(cyl~am, data=mtcars)
```

one continuous, one categorical:
```
mosaic::mean(mpg~cyl, data=mtcars)
```

tilde

PLOTTING:

one continuous variable:
```
lattice::histogram(~disp, data=mtcars)

lattice::bwplot(~disp, data=mtcars)
```

one categorical variable:
```
mosaic::bargraph(~cyl, data=mtcars)
```

two continuous variables:
```
lattice::xyplot(mpg~disp, data=mtcars)
```

two categorical variables:
```
mosaic::bargraph(~am, data=mtcars, group=cyl)
```

one continuous, one categorical:
```
lattice::histogram(~disp|cyl, data=mtcars)

lattice::bwplot(cyl~disp, data=mtcars)
```

The variety of R syntaxes give you many ways to "say" the same thing

read **across** the cheatsheet to see how different syntaxes approach the same problem

Tidyverse syntax

```
data %>% goal(x)
```

SUMMARY STATISTICS:

one continuous variable:
```
mtcars %>% dplyr::summarize(mean(mpg))
```

one categorical variable:
```
mtcars %>% dplyr::group_by(cyl) %>%
dplyr::summarize(n())
```

the pipe

two categorical variables:
```
mtcars %>% dplyr::group_by(cyl, am) %>%
dplyr::summarize(n())
```

one continuous, one categorical:
```
mtcars %>%  dplyr::group_by(cyl) %>%
  dplyr::summarize(mean(mpg))
```

PLOTTING:

one continuous variable:
```
ggplot2::qplot(x=mpg, data=mtcars, geom = "histogram")

ggplot2::qplot(y=disp, x=1, data=mtcars, geom="boxplot")
```

one categorical variable:
```
ggplot2::qplot(x=cyl, data=mtcars, geom="bar")
```

two continuous variables:
```
ggplot2::qplot(x=disp, y=mpg, data=mtcars, geom="point")
```

two categorical variables:
```
ggplot2::qplot(x=factor(cyl), data=mtcars, geom="bar") +
        facet_grid(.~am)
```

one continuous, one categorical:
```
ggplot2::qplot(x=disp, data=mtcars, geom = "histogram") +
        facet_grid(.~cyl)

ggplot2::qplot(y=disp, x=factor(cyl), data=mtcars,
               geom="boxplot")
```

WRANGLING:

subsetting:
```
mtcars %>% dplyr::filter(mpg>30)
```

making a new variable:
```
mtcars <- mtcars %>%
dplyr::mutate(efficient = if_else(mpg>30,  TRUE, FALSE))
```

Search or jump to... / Pull requests Issues Marketplace Explore

OpenIntroOrg / oiLabs-dplyr-ggplot

Unwatch 8 Star 9 Fork 19

Code Issues 6 Pull requests 2 Projects 0 Wiki Insights

OpenIntro Labs in R using the dplyr syntax for data manipulation

317 commits 4 branches 0 releases 5 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

mine-cetinkaya-rundel render HTML Latest commit 6c53b64 on Jun 19, 2017

01_intro_to_r	Package name styling: bolded first mention, normal there onwards (a l...	11 months ago
02_intro_to_data	eval = FALSE for posting	11 months ago
03_normal_distribution	Styling of package name	11 months ago
04_probability	eval = FALSE for posting	11 months ago
05_sampling_distributions	styling of package names	11 months ago
06_confidence_intervals	render HTML	11 months ago
07_inf_for_numerical_data	fix typo	2 years ago
08_inf_for_categorical_data	add global chunk options	2 years ago
09_simple_regression	update formatting	2 years ago
10_multiple_regression	removed pdf output from YAML in MLR	2 years ago

https://github.com/OpenIntroOrg/oiLabs-dplyr-ggplot

Search or jump to... / Pull requests Issues Marketplace Explore

OpenIntroOrg / oiLabs-dplyr-ggplot

Unwatch 8 Star 9 Fork 19

Code Issues 6 Pull requests 2 Projects 0 Wiki Insights

OpenIntro Labs in R using the dplyr syntax for data manipulation

317 commits 4 branches 0 releases 5 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

mine-cetinkaya-rundel render HTML

01_intro_to_r	Package name styling: bolded first mention, normal th	
02_intro_to_data	eval = FALSE for posting	
03_normal_distribution	Styling of package name	
04_probability	eval = FALSE for posting	
05_sampling_distributions	styling of package names	11 months ago
06_confidence_intervals	render HTML	11 months ago
07_inf_for_numerical_data	fix typo	2 years ago
08_inf_for_categorical_data	add global chunk options	2 years ago
09_simple_regression	update formatting	2 years ago
10_multiple_regression	removed pdf output from YAML in MLR	2 years ago

Clone with HTTPS Use SSH

Use Git or checkout with SVN using the web URL.

https://github.com/OpenIntroOrg/oiLabs-d

Open in Desktop Download ZIP

https://github.com/OpenIntroOrg/oiLabs-dplyr-ggplot

R Markdown: Integrating A Reproducible Analysis Tool into Introductory Statistics

2014 | **Author(s):** Baumer, Ben; Cetinkaya-Rundel, Mine; Bray, Andrew; Loi, Linda; Horton, Nicholas J.

Main Content | Metrics | Author & Article Info

Abstract

Nolan and Temple Lang argue that "the ability to express statistical computations is an es- sential skill." A key related capacity is the ability to conduct and present data analysis in a way that another person can understand and replicate. The copy-and-paste workflow that is an artifact of antiquated user-interface design makes reproducibility of statistical analysis more difficult, especially as data become increasingly complex and statistical methods become increasingly sophisticated. R Markdown is a new technology that makes creating fully-reproducible statistical analysis simple and painless. It provides a solution suitable not only for cutting edge research, but also for use in an introductory statistics course. We present experiential and statistical evidence that R Markdown can be used effectively in introductory statistics courses, and discuss its role in the rapidly-changing world of statistical computation.

Main Content

View Larger

1. Introduction

Statistical analysis of data is both increasingly common and increasingly sophisticated. While the imperative to convey findings with clarity remains, the modern statistical analyst faces a variety of challenges that may make analyses more difficult to understand. First, as the field of statistics deepens, applications of statistics are increasingly complex. Second, collaboration among researchers is now the norm, rather than the exception. Third, much of that collaboration is conducted remotely, with written analyses, data files, and computing scripts shared via electronic means. Fourth, the

Jump To

Article
- Abstract
- Main Content

Metrics

Author & Article Info

Related Items

The fivethirtyeight R Package: "Tame Data" Principles for Introductory Statistics and Data Science Courses
Kim, Albert Y.; Ismay, Chester; Chunn, Jennifer

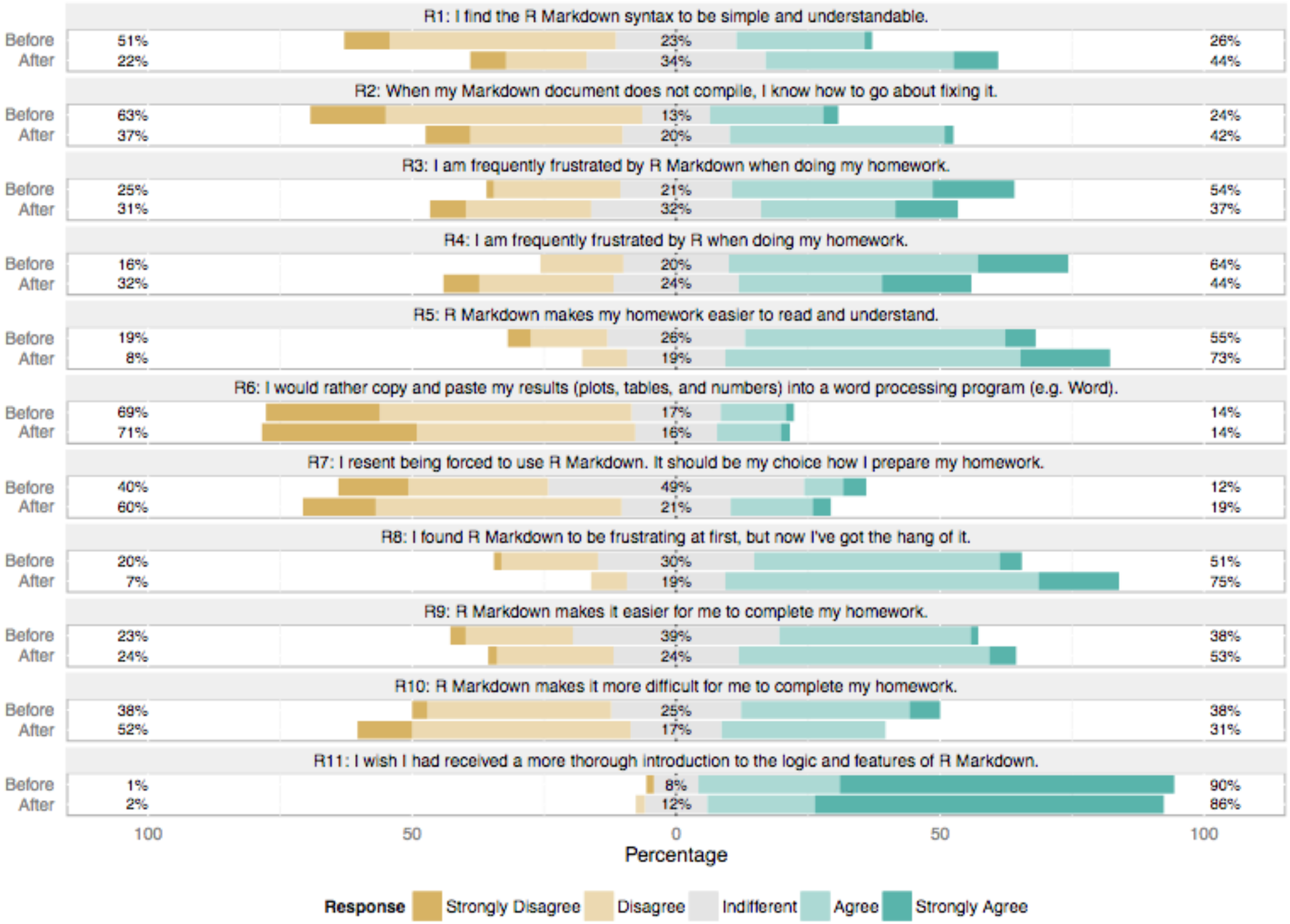
Dynamic Data in the Statistics Classroom
Hardin, Johanna

Data Visualization on Day One: Bringing Big Ideas into Intro Stats Early and Often
Wang, Xiaofei; Rush, Cynthia; Horton, Nicholas Jon

Web Application Teaching Tools for Statistics Using R and Shiny
DOI, JIMMY; POTTER, GAIL; WONG, JIMMY; ALCARAZ, IRVIN; CHI, PETER

Student Approaches to Constructing Statistical Models using TinkerPlots ™
Noll, Jennifer; Kirin, Dana

https://escholarship.org/uc/item/90b2f5xh



https://escholarship.org/uc/item/90b2f5xh

Advanced features of R Markdown

Code

R Markdown is a document authoring format used by many data scientists. In this lab, you will explore some of the advanced formatting features of R Markdown to achieve a professional look.

Goal: by the end of this lab, you will be able to format an article in R Markdown using many advanced features.

R Markdown HTML

An R Markdown document can be rendered into many different formats. Since the piece we are writing is for the Web, we will render our document into HTML. In addition to the `knitr` chunk options that control how your R code gets rendered, R Markdown provides a number of features that can make your HTML document more expressive.

These features can be unlocked by setting parameters in the YAML header. YAML is an abbreviation for "Yet Another Markup Language", and it is just a syntax for specifying options (like you might in a configuration file).

The following features are described in the **R Markdown HTML documentation**. Please consult that for instructions on how to use these features.

Figure options

For the web, it's a good idea to make your figures as wide as the text around which they are inserted. Please also use captions to contextualize the graphic! [By "figures", here we mean data graphics—not images.]

Exercise 1 Start a new R Markdown document (from the File menu) and render it. Experiment with the `fig_width` YAML setting and note how it changes the figure widths.

Data frame printing

If you have to display a large data table, it would be nice to allow your readers the chance to page and scroll through it. Use the Paged Printing option by setting `df_print: paged` in YAML.

```
---
title: "My document"
output:
  html_document:
    df_print: paged
---
```

Exercise 2 Turn on paged data frame printing and then print a data frame.

Code folding

Please use code folding, with the default set to `hide`.

Exercise 3 Turn on `code_folding` for your Markdown document and set the default to `hide`.

Themes

If you want to experiment with different themes, choose one from this Bootswatch theme gallery.

https://beanumber.github.io/sds192/lab-rmarkdown.html

Challenges

- Technical issues

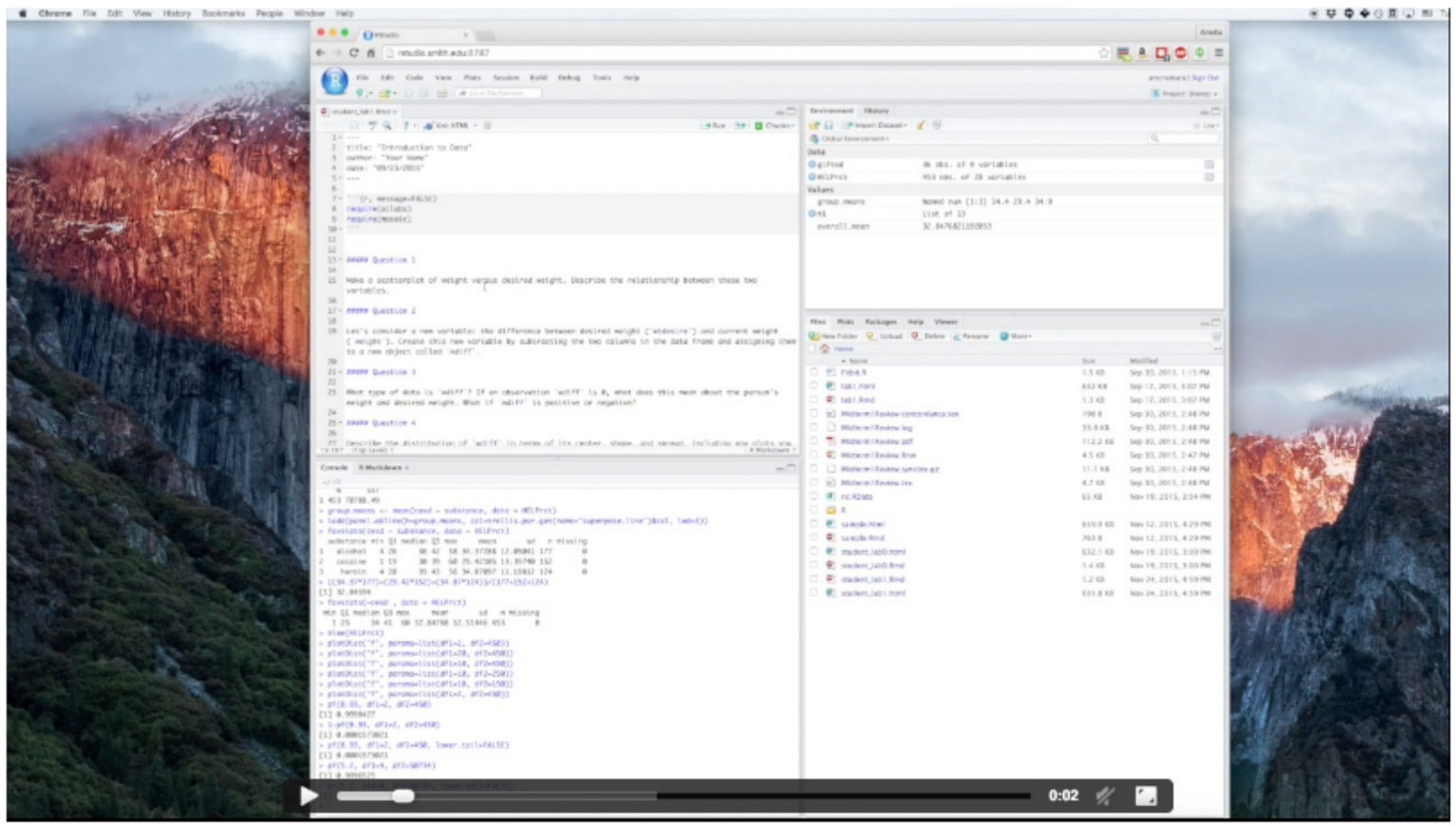
- Motivating students to use

Troubleshooting in R Markdown

There are a few common problems that people have had with their labs. This page will be updated with new problems when they arise, and it's a good place to look if you're having trouble.

Downloading the HTML so you can upload it to Moodle

To download the knitted HTML, go to the Files tab (lower right corner, same pane as Plots and Help) and select the checkbox next to your document's name. Make sure it is the HTML file with the same filename as the Rmd file you were editing. Then click the More button and select Export. This will download the file onto your computer and you can then upload it to Moodle. For a short video showing this process, see here.



Document won't knit

There could be many reasons for this. Usually, the error message will pinpoint the location of the problem. **Read the error messages!!**

Some most common problems are:

- including output in your code chunks, like

```
mean(~speed, data = cars) 169.683
```

```
## Error: <text>:1:27: unexpected numeric constant
## 1: mean(~speed, data = cars) 169.683
```

http://bit.ly/TroubleshootingRmd

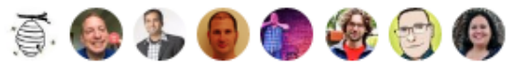
Karl Broman
@kwbroman

Following

.@xieyihui's homework to reinforce repro research:
- week 1: analyze these data
- week 2: [made some changes to those data] go back & redo

9:19 AM - 3 Aug 2016

3 Retweets **5** Likes



2 3 5

Tweet your reply

Karl Broman @kwbroman · 3 Aug 2016

Replying to @kwbroman

Audience member uses this approach but in wks 2 and 3 w/ week 1 being about how to do things reproducibly. #JSM2016

> **Karl Broman** @kwbroman
> .@xieyihui's homework to reinforce repro research:
> - week 1: analyze these data
> - week 2: [made some changes to those data] go back & redo

1

Noam Ross @noamross · 3 Aug 2016

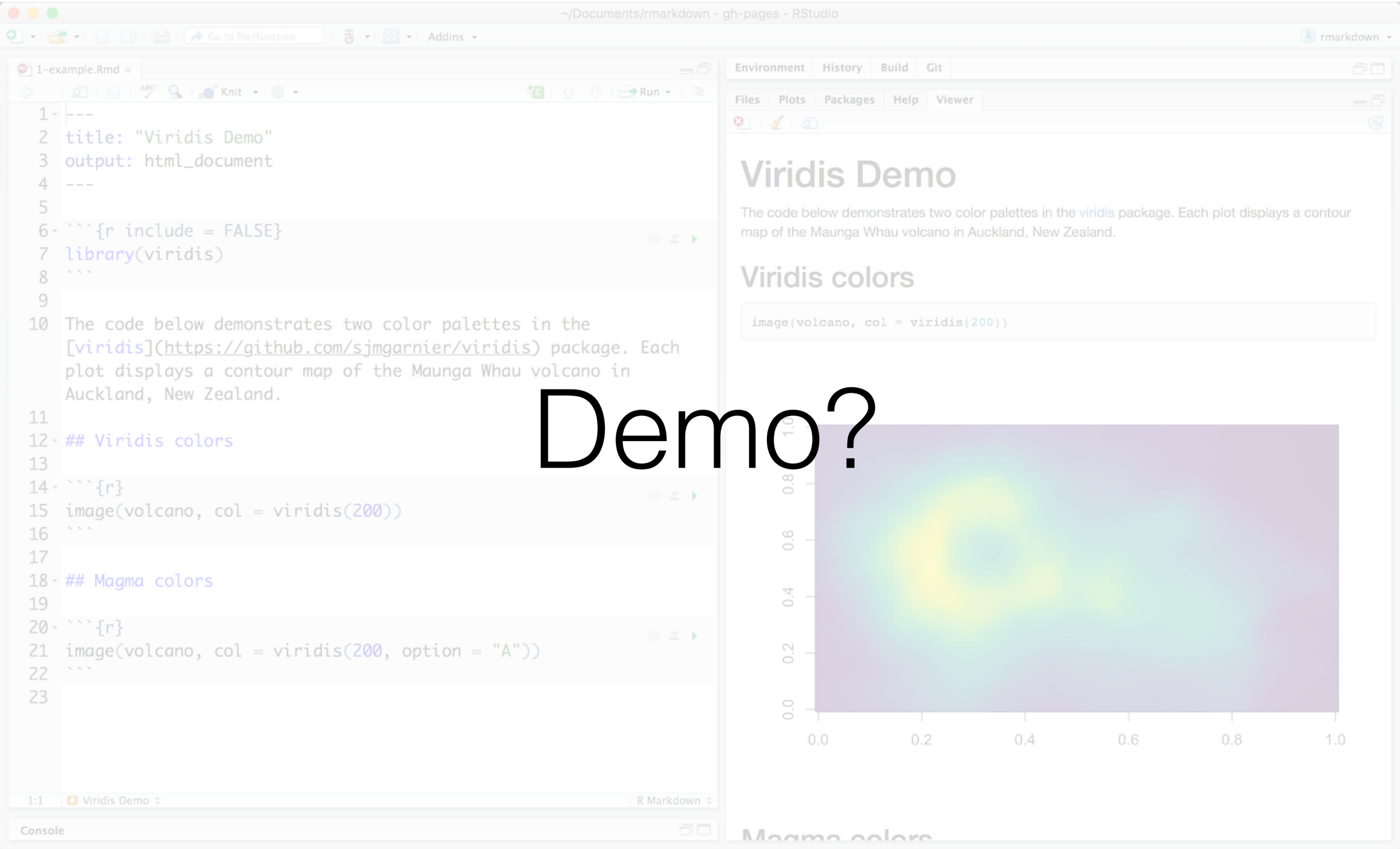
Replying to @kwbroman @xieyihui

Doesn't @JennyBryan do this, but the students' *classmate* has to redo their analysis?

https://twitter.com/kwbroman/status/760857559859429376

RMarkdown

Demo?



http://rmarkdown.rstudio.com/

Thank you

Amelia McNamara @AmeliaMN
Current: Program in Statistical & Data Sciences, Smith College
Fall 2018: Department of Computer & Information Sciences, University of St Thomas