

1.5 – Optimize Workflow

ECON 480 • Econometrics • Fall 2020

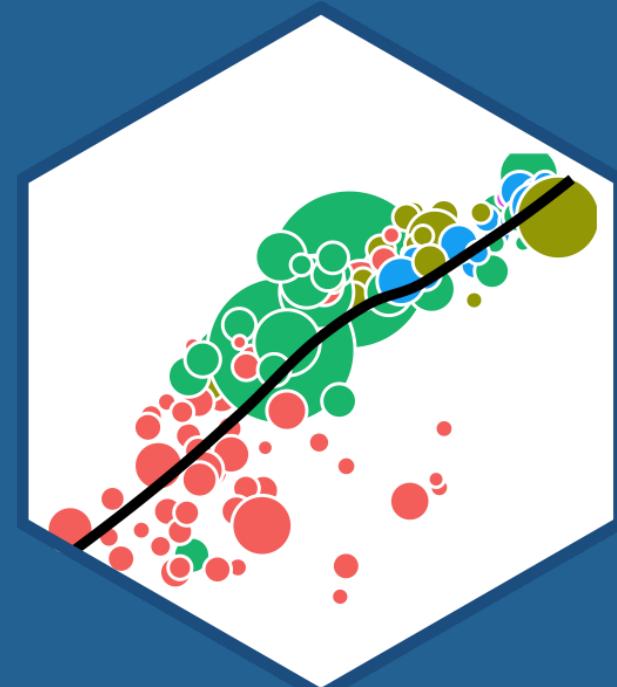
Ryan Safner

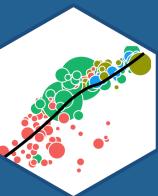
Assistant Professor of Economics

 safner@hood.edu

 [ryansafner/metricsF20](https://github.com/ryansafner/metricsF20)

 metricsF20.classes.ryansafner.com





The Office Model

The Plain Text Model

R Markdown

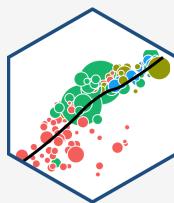
Compiling Your Documents

R Projects

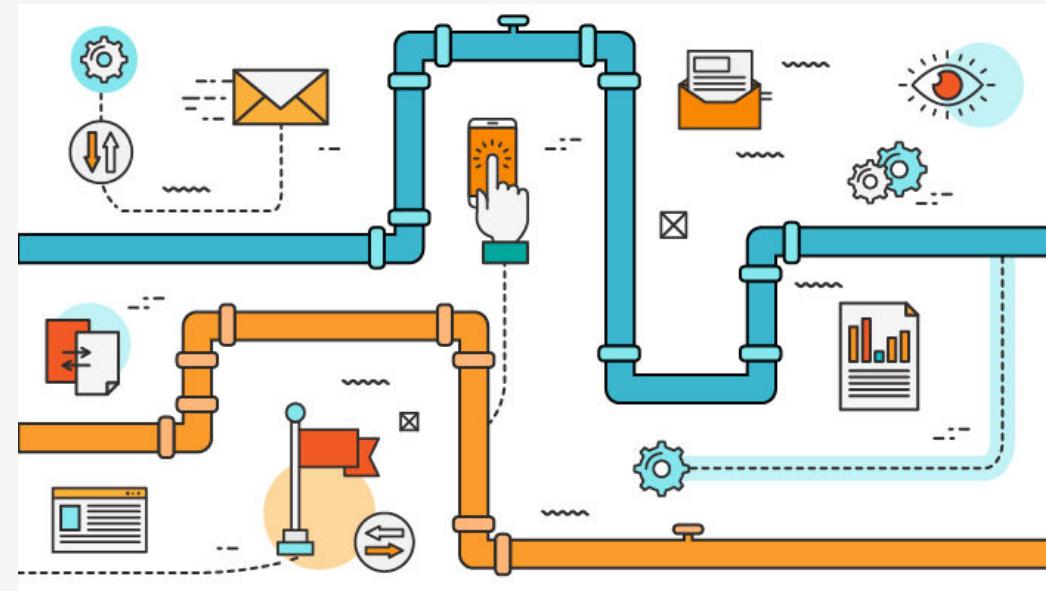
Version Control

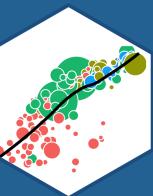
Resources

Your Workflow Has a Lot of Moving Parts



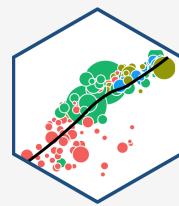
1. Writing text/documents
2. Managing citations and bibliographies
3. Performing data analysis
4. Making figures and tables
5. Saving files for future use
6. Monitoring changes in documents
7. Collaborating and sharing with others
8. Combining into a deliverable (report, paper, presentation, etc.)



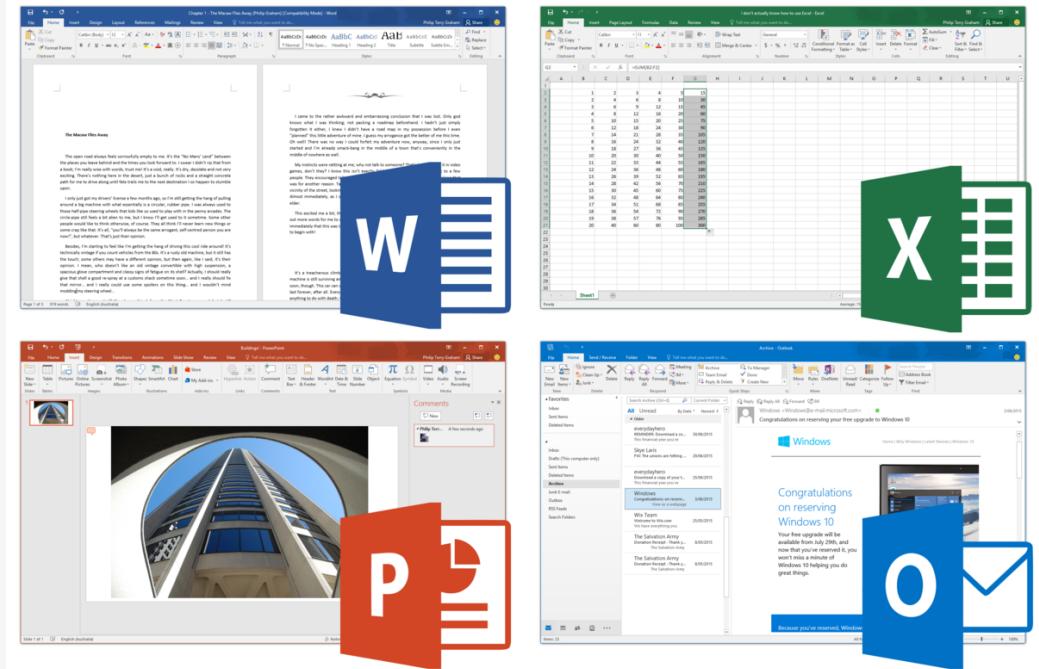


The Office Model

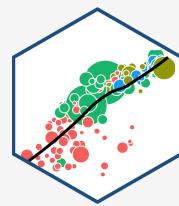
The Office Model I



1. Writing text/documents
2. Managing citations and bibliographies
3. Performing data analysis
4. Making figures and tables
5. Saving files for future use
6. Monitoring changes in documents
7. Collaborating and sharing with others
8. Combining into a deliverable (report, paper, presentation, etc.)

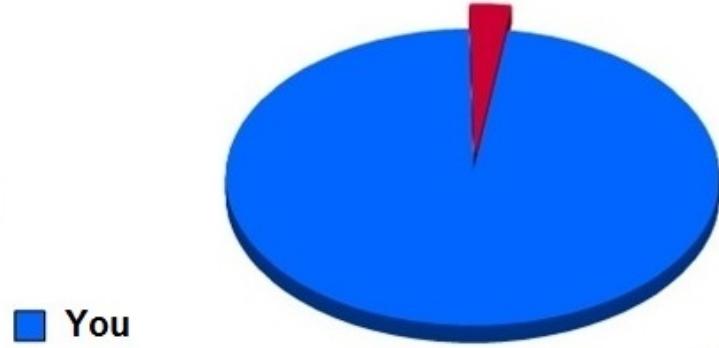


The Office Model II



- A lot of **copy-pasting**
- A lot of...

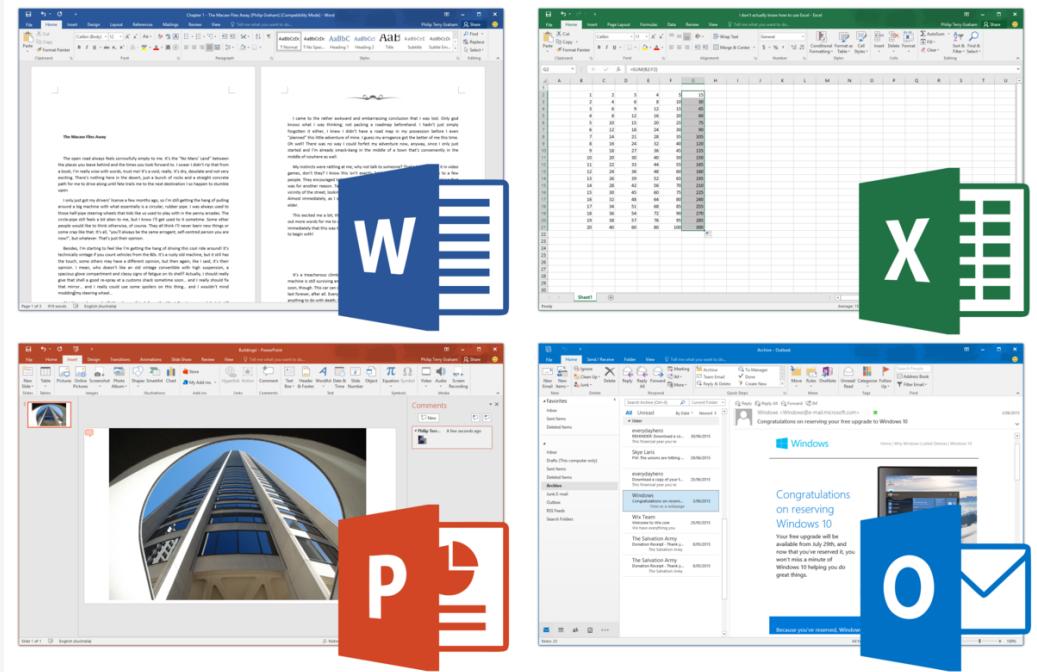
Moving a picture in Microsoft Word



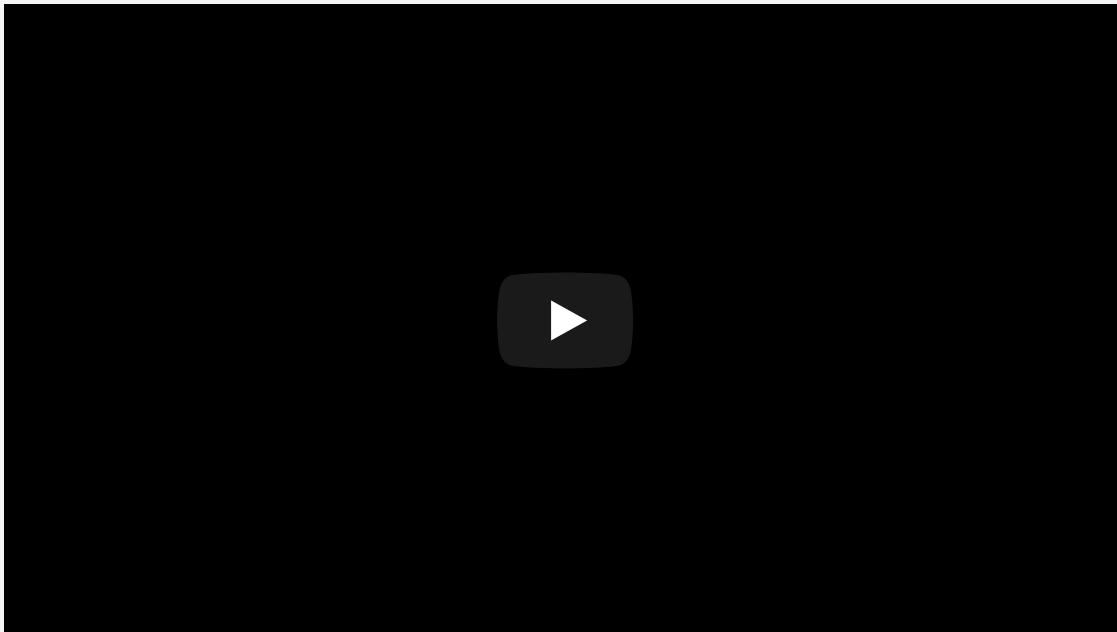
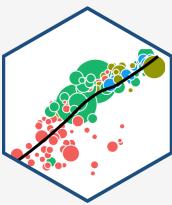
■ You
the

It
actually
does
what
you
want

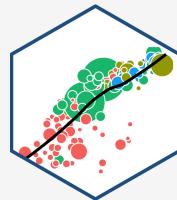
mess up
whole document



The Office Model: A Short Horror Movie



The Office Model: Mistakes



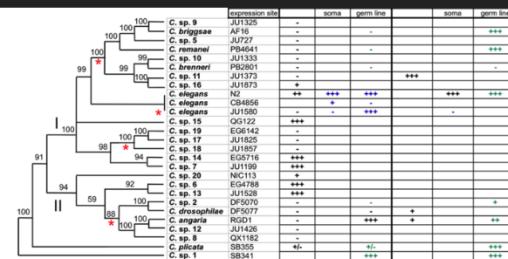
AAAS Become a Member

Science

Contents News Careers Journals

Log In ScienceMag.org Search

164 49



expression site

C. sp. 9 JU1590 - soma germ line soma germ line

C. briggsae AF16 - - ***

C. sp. 10 JU727 - - ***

C. sp. 11 JU1333 - - ***

C. sp. 12 JU1373 - - ***

C. sp. 13 JU1873 - - ***

C. elegans CB4816 - - ***

C. elegans JU1590 - - ***

C. elegans CB4812 - - ***

C. sp. 19 EG6142 - - ***

C. sp. 17 JU1825 - - ***

C. sp. 16 JU1877 - - ***

C. sp. 14 EG5716 - - ***

C. sp. 15 JU1199 ***

C. sp. 20 N2 - - ***

C. sp. 6 EG4788 - - ***

C. sp. 13 JU1529 - - ***

C. sp. 12 DF5076 - - ***

C. drosophila DF5077 - - ***

C. elegans JU1426 - - ***

C. elegans QX1182 - - ***

C. elegans SR350 - - ***

C. elegans SR341 - - ***

PLOS ONE PHYLOGENY/Flickr (CC BY 2.0)

One in five genetics papers contains errors thanks to Microsoft Excel

By Jessica Boddy | Aug. 29, 2016, 1:45 PM

Autoformatting in Microsoft Excel has caused many a headache—but now, a new study shows that one in five genetics papers in top scientific journals contains errors from the program, *The Washington Post* reports. The errors often arose when gene names in a spreadsheet were automatically changed to calendar dates or numerical values. For example, one gene called *Septin-2* is commonly shortened to *SEPT2*, but is changed to 2-SEP and stored as the date 2 September 2016 by Excel. The researchers, who published their analysis in *Genome Biology*, say the issue can be fixed by formatting Excel columns as text and remaining vigilant—or switching to Google Sheets, where gene names are stored exactly as they're entered.

Got a tip? How to contact the news team

Latest News

Trending Most Read

- Crystalline nets harvest water from desert air, turn carbon dioxide into liquid fuel
- This bird really can fly over Mount Everest, wind tunnel experiments reveal
- Stalled in Hawaii, giant telescope faces roadblocks at its backup site in the Canary Islands
- Study challenges idea that autism is caused by an overly masculine brain
- Genetics may explain up to 25% of same-sex behavior, giant analysis reveals

Related Jobs

Engineer I / II, Sterile Product Technology
Moderna

Source: [Science Magazine](#)

Bloomberg the Company & Its Products | Bloomberg Anywhere Remote Login | Bloomberg Terminal Demo Request

Menu Search

Bloomberg Businessweek Sign In Subscribe

■ April 18, 2013, 6:31 AM EDT

FAQ: Reinhart, Rogoff, and the Excel Error That Changed History

By Peter Coy

LIVE ON BLOOMBERG Watch Live TV > Listen to Live Radio >

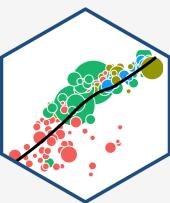
Bloomberg Television



PHOTOGRAPH BY GREGOR SCHUSTER

Source: [Bloomberg](#)

The Office Model: Not Reproducible



Kaitlin Thaney 
@kaythaney



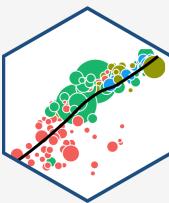
"Reproducible research' is a redundant term. 'Irreproducible research' just used to be known as 'bullshit'." - [@fperez_org](#)
::slow clap::

7:11 PM · May 8, 2014



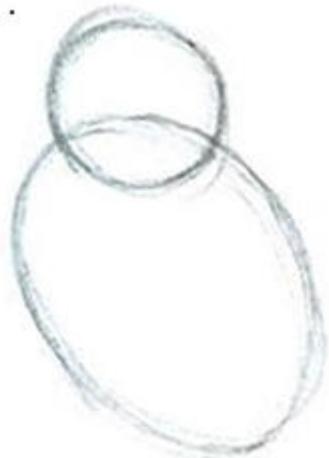
 57  110 people are Tweeting about this

...The Rest of the Owl



How to draw an owl

1.



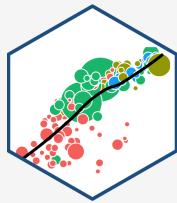
2.



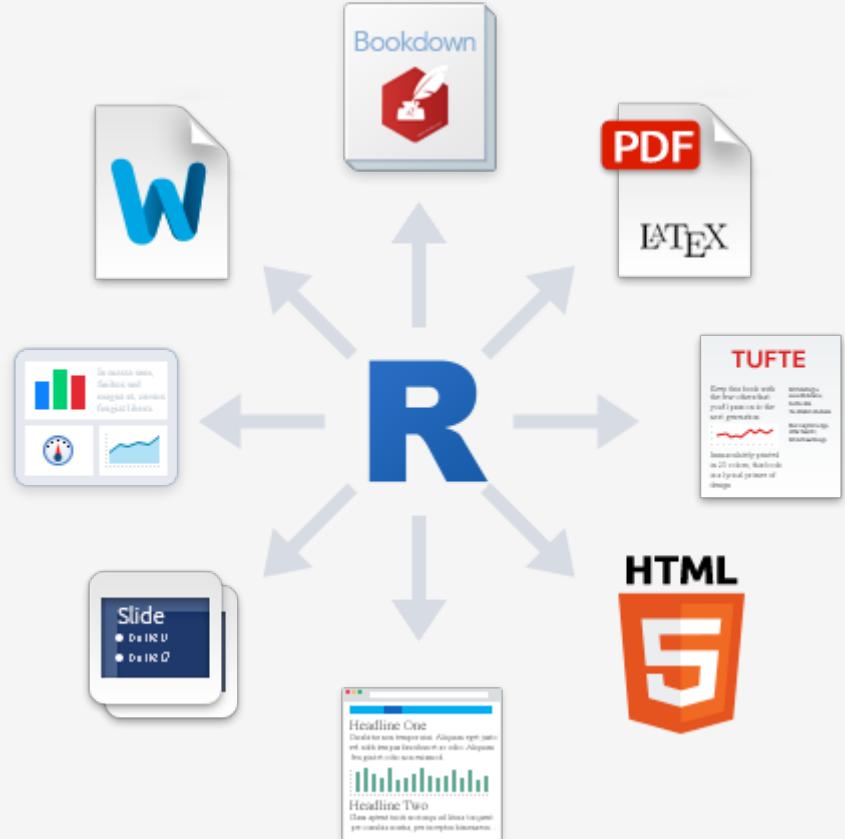
1. Draw some circles

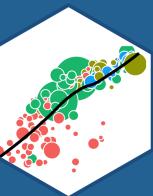
2. Draw the rest of the fucking owl

What I'm About to Show You



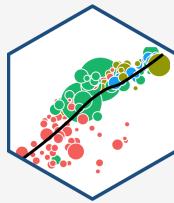
- This is how I make my...
 - Research papers
 - Course documents
 - Websites
 - Slides and presentations
- I have not used any MS Office products since 2011 (good riddance!)
- **This stuff is *optional***
 - If you like your office model, you can keep it
 - But this is what most people who take this course continue to use (R is only really if you have data work)



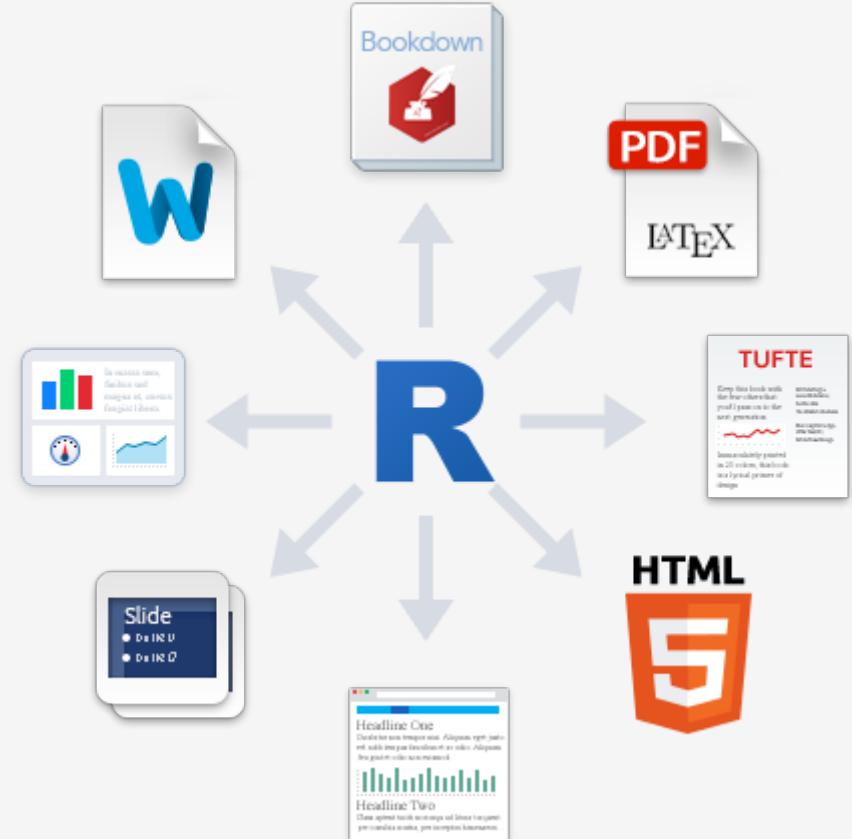


The Plain Text Model

The Plain Text Model I

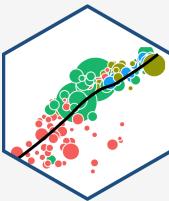


- Meet [R Markdown](#), which can do *all of this* in one pipeline
 - 1. Writing text/documents
 - 2. Managing citations and bibliographies
 - 3. Performing data analysis
 - 4. Making figures and tables
 - 5. Saving files for future use
 - 6. Monitoring changes in documents
 - 7. Collaborating and sharing with others
 - 8. Combining into a deliverable (report, paper, presentation, etc.)

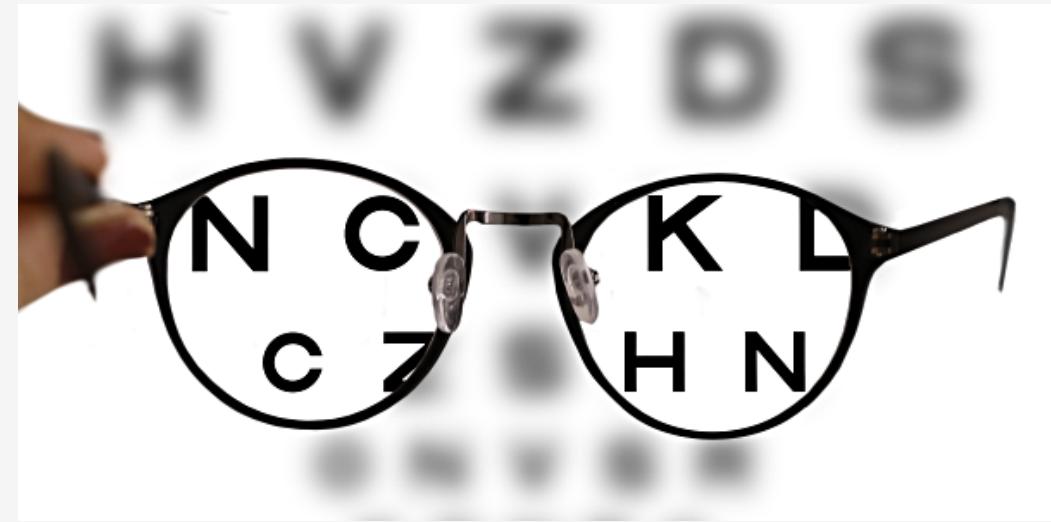


From [R Studio's R Markdown Cheatsheet](#)

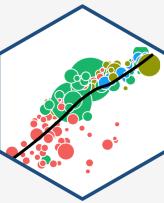
The Plain Text Model II



- Plain text files: readable by *both* machines and humans
 - Understand how a document is structured and formatted via code and markup to text
- Focus entirely on the *actual writing of the content* instead of the formatting and aesthetics
 - You can still customize, but with precise commands instead of point, click, drag, guess, pray



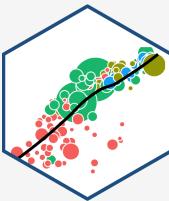
The Plain Text Model III



- **Open Source:** free, useable forever, often very small file size
 - Proprietary software is a gamble - can you still open a `.doc` file from Microsoft Word 1997?
- **Automate and Minimize Errors,** especially in repetitive processes
- Can be used with **version control** (see below)



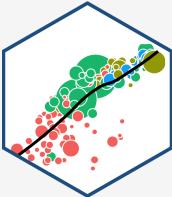
Making Your Work Reproducible



One day you will need to quit R, go do something else and return to your analysis the next day. One day you will be working on multiple analyses simultaneously that all use R and you want to keep them separate. One day you will need to bring data from the outside world into R and send numerical results and figures from R back out into the world. To handle these real life situations, you need to make two decisions: What about your analysis is "real", i.e. what will you save as your lasting record of what happened? Where does your analysis "live"?

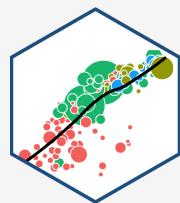
- Hadley Wickham, [R For Data Science](#)
- We've talked about `.R` script files that let you "keep" commands
- What about output? Must you save and copy/paste to MS Word? **No!**

Making Your Work Reproducible



- R Markdown file (`.Rmd`) is the "real" part of your analysis, *everything* can live in this plain-text file!
- Document text in `markdown`
- R code executed in "chunks"
- Plots and tables generated from R code
- Citations and bibliography automated with `.bib` file

The Future of Science is Open Source Plain Text

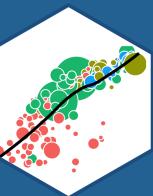


The screenshot shows a science article from The Atlantic. The main image is a photograph of several elongated human skulls on fire. The title of the article is "Genetic analysis of elongated skulls suggests extensive female-biased immigration to early Medieval Bavaria". Below the title is a dense block of author names and their affiliations. The text of the article begins with "The Scientific Paper Is Obsolete". A sidebar on the right contains a "TABLE OF CONTENTS" with links to various sections of the paper. At the bottom, there are social sharing buttons for Facebook, Twitter, and LinkedIn, along with a "Subscribe Now" button.

Source: [The Atlantic](#)

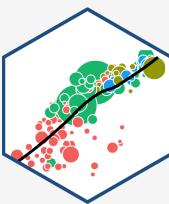
The screenshot shows a blog post by Paul Romer titled "Jupyter, Mathematica, and the Future of the Research Paper" dated April 13, 2018. The post discusses the transition from traditional scientific papers to open-source tools like Jupyter and Mathematica. To the right of the main content is a "TABLE OF CONTENTS" sidebar with several links. The sidebar includes sections like "The open question", "The difference that matters", "The answer to the question and the lesson we should learn", "My experience with Mathematica", and "I'm happy with Jupyter" and "I'm frightened by the Vandals".

Source: [Paul Romer \(2018 Economics Nobel\)](#)



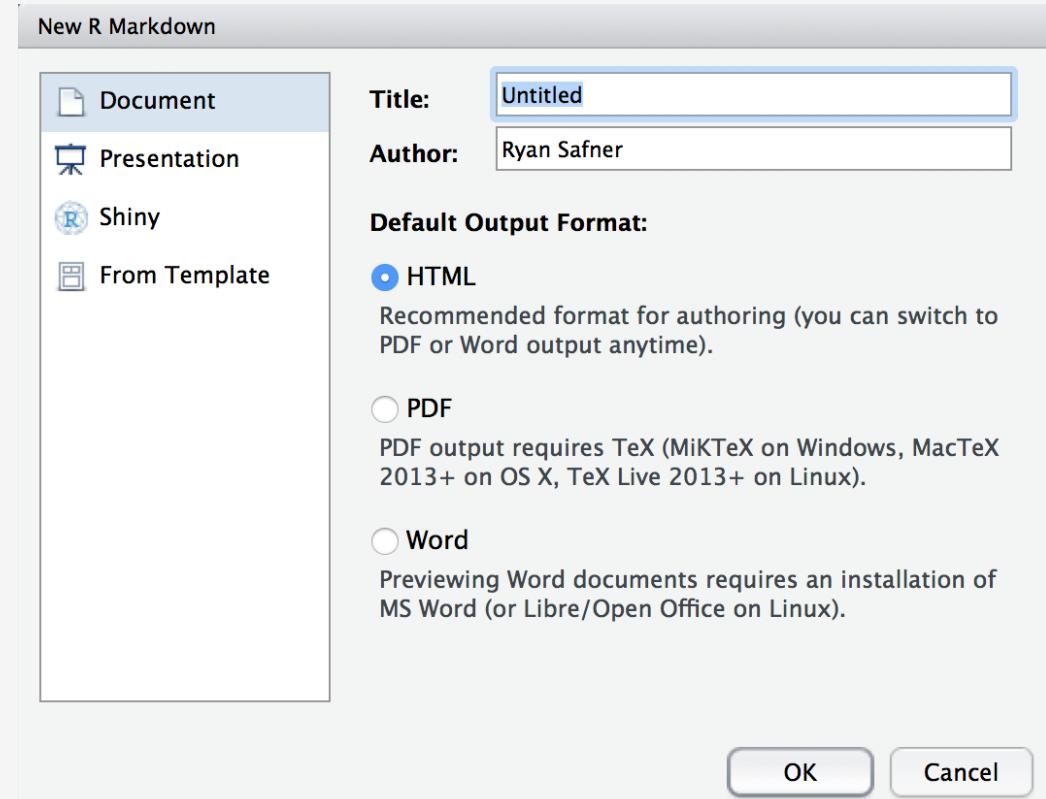
R Markdown

Creating an R Markdown Document I

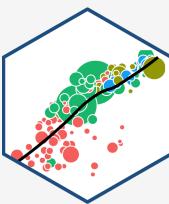


File -> New File -> R
Markdown...

- Outputs:
 - Document (what you'll use for most things)
 - Presentation (for making slides in various formats)
 - Shiny (an html and R based web app, advanced)
 - Templates (some built-in, other packages like `rticles` or `xaringan` add neat templates)

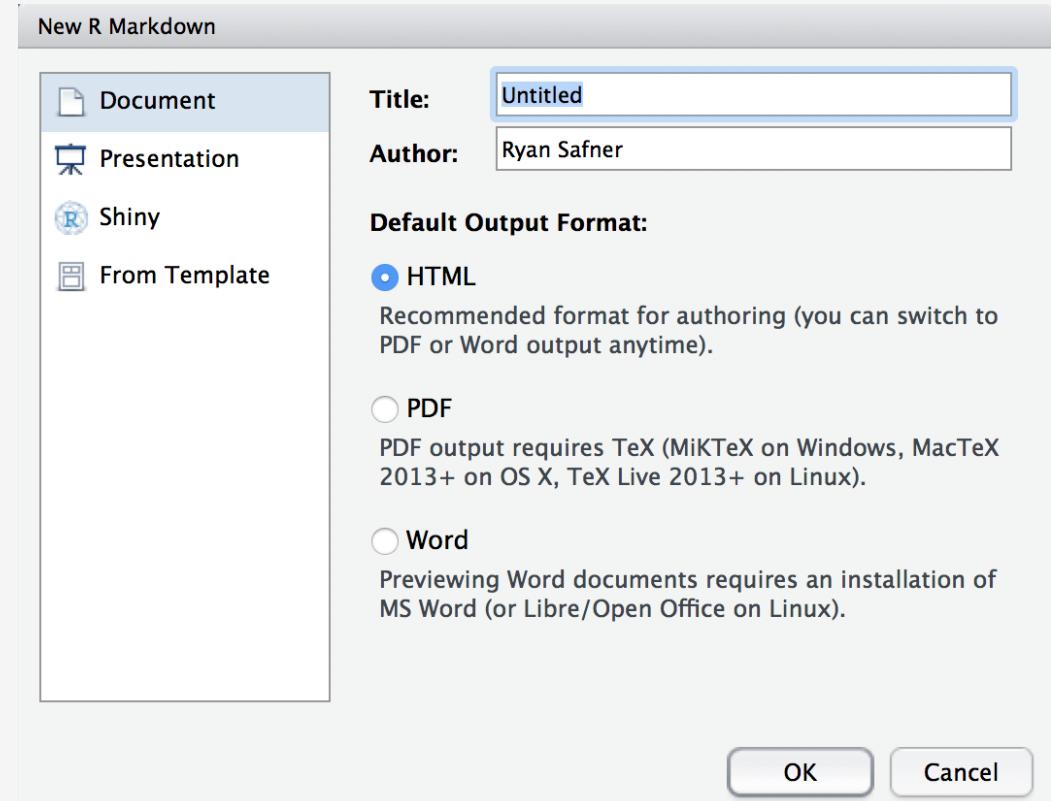


Creating an R Markdown Document II

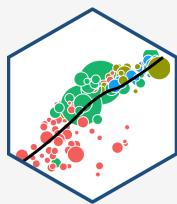


File -> New File -> R Markdown...

- `html`: renders a webpage, viewable in any browser
 - default, easiest to produce and share
 - can have interactive elements (gifs, animations, web apps)
 - requires internet connection to host and share (*you can view offline*)
- `pdf`: renders a PDF document
 - most common document format around
 - requires `LaTeX` distribution to render (more on that soon)
- `word`: create a Microsoft Word document
 - ...if you must



Structure of an R Markdown Document



Entire document is written in a single file:¹

with three types of content:

1. [YAML](#) header for metadata
2. Text of the document written with [markdown](#)
3. [R](#) chunks for data analysis, plots, figures, tables, statistics, as necessary

The screenshot shows the RStudio interface. On the left, the 'chunks.Rmd' file is open in the editor. It contains the following content:

```
1 R Code Chunks
2 -----
3
4 With R Markdown, you can insert R code
| chunks including plots:
5
6 ````{r qplot, fig.width=4, fig.height=3,
| message=FALSE}
7 # quick summary and plot
8 library(ggplot2)
9 summary(cars)
10 qplot(speed, dist, data=cars) +
11   geom_smooth()
12 ...
13 |
```

The preview window on the right shows the rendered HTML output. The title is 'R Code Chunks'. The content includes:

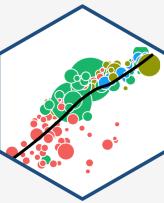
- A section titled 'With R Markdown, you can insert R code chunks including plots:' followed by the R code from the file.
- A summary table of the 'cars' dataset:

	speed	dist
Min.	4.0	2
1st Qu.	12.0	1st Qu.: 26
Median	15.0	Median : 36
Mean	15.4	Mean : 43
3rd Qu.	19.0	3rd Qu.: 56
Max.	25.0	Max. :120

- The rendered R plot: `qplot(speed, dist, data = cars) + geom_smooth()`. The plot shows a scatter of data points with a fitted regression curve.

¹ The one exception is for managing bibliographies, this requires one additional [.bib](#) file!

YAML Header I



- Top of a document contains the YAML¹ separated by three dashes --- above and below
- Contains the **metadata** of the document, such as:

```
title: "My Title"
author: "Ryan Safner"
date: "`r Sys.Date()`" # here I'm using R code to generate today's date!
output: pdf_document
```

- `output` *must* be specified, everything else can be left blank, and other options can be added as necessary
- In most cases, you can safely ignore other things in the `yaml` until you are ready

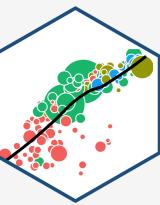
¹ YAML stands for "YAML Ain't Markup Language." Nerds love recursive acronyms.

YAML Header: Example from one of my research papers



```
title: Distributing Patronage[I would like to thank the Board of Associates of Hood College...]
subtitle: Intellectual Property in the Transition from Natural State to Open Access Order
date: \today
author:
- Ryan Safner[Hood College, Department of Economics and Business Administration; safner@hood.edu]
abstract: |
  | "This paper explores the emergence of the modern forms of copyright and patent in ...
  | *JEL Classification:* 030, 043, N43
  | *Keywords:* Copyright, intellectual property, economic history, freedom of the press, economic development
bibliography: patronage.bib
geometry: margin = 1in
fontsize: 12pt
mainfont: Fira Sans Condensed
output:
  pdf_document:
    latex_engine: xelatex
    number_sections: true
    fig_caption: yes
header-includes:
- \usepackage{booktabs}
```

R Chunks I



- You can create a "**chunk**" of R code with **three backticks**¹ above and below your code
- After the first pair of backticks, signify the **language** of the code² inside braces, e.g:

Input

```
```{r}  
2+2 # code goes here!
```
```

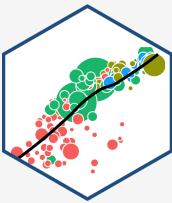
Output

```
2+2 # code goes here!  
## [1] 4
```

¹ The key to the left of the #1 key on your keyboard.

² Yes that does mean you can use other coding languages!

R Chunks II



Input

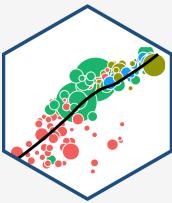
```
```{r}
head(mpg, n=2)
```
```

Output

```
head(mpg, n=2)

## # A tibble: 2 x 11
##   manufacturer model  displ  year   cyl trans
##   <chr>        <chr> <dbl> <int> <int> <chr>
## 1 audi         a4     1.8   1999     4 auto(l5)
## 2 audi         a4     1.8   1999     4 manual(m5)
```

R Chunks III



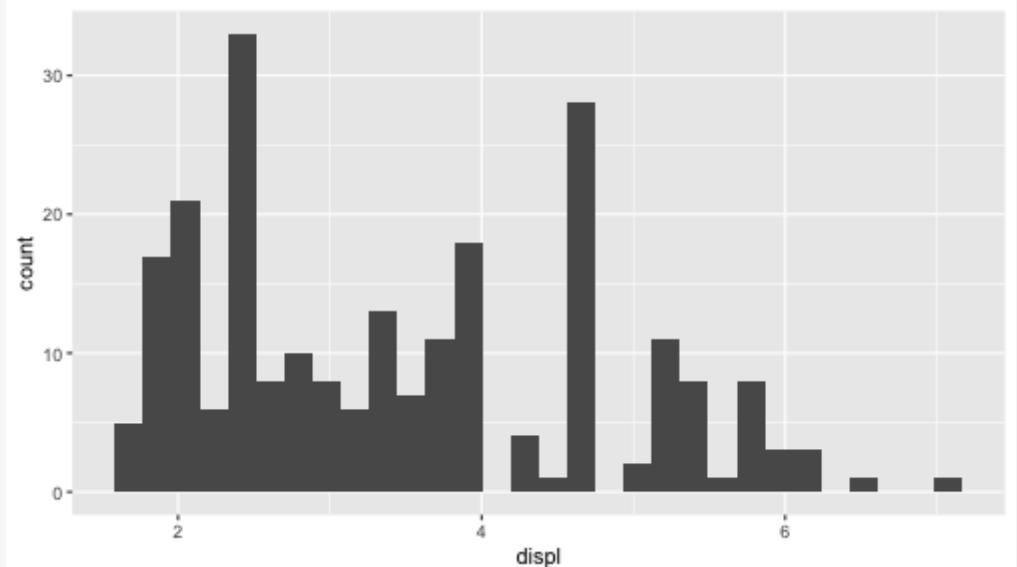
Input

```
```{r}
library("ggplot2") # load ggplot2
ggplot(data = mpg) +
 aes(x = displ) +
 geom_histogram()
```

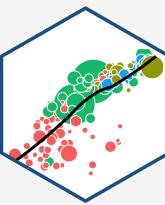
```

Output

```
library("ggplot2") # load ggplot2
ggplot(data = mpg) +
  aes(x = displ) +
  geom_histogram()
```



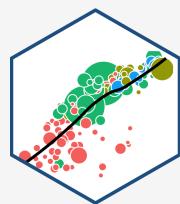
R Chunks Options



- You can add additional options inside the `{braces}` after `r`, some common options:
 - This is the only option that goes after `r` but *before* a comma
- `echo`
 - set `=TRUE` to display the R code input
 - set `=FALSE` shows will not show your code
- `eval`
 - set `=TRUE` to run your code
 - `=FALSE` only displays your code without running it
- `fig` has a lot of options for displaying plot outputs (`fig.height`, `fig.width`, `fig.asp`, etc)

```
```{r my_cool_chunk, echo=F, warning = F}  
...
```
```

R Chunks Options Example



Input

```
```{r check-data, echo = T}
get top 3 avg displacement by manuf
mpg %>%
 group_by(manufacturer) %>%
 summarize(avg = mean(displ)) %>%
 arrange(desc(avg)) %>%
 slice(1:3)
````
```

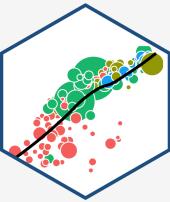
```
```{r make-plot, echo = F, fig.height=2}
ggplot(data = mpg) +
 aes(x = displ) +
 geom_histogram()
````
```

Output

```
# get top 3 avg engine displacement by manuf
mpg %>%
  group_by(manufacturer) %>%
  summarize(avg = mean(displ)) %>%
  arrange(desc(avg)) %>%
  slice(1:3)

## # A tibble: 3 x 2
##   manufacturer     avg
##   <chr>           <dbl>
## 1 lincoln          5.4
## 2 chevrolet        5.06
## 3 jeep             4.58
```

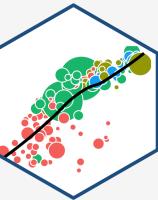
R Chunks Options: Set Defaults



- If you want to be fancy, you can set global options that affect *all chunks*
- Use a special named `setup` chunk at top (comes in default `.Rmd` template)
 - set global options inside the `knitr::opts_chunk$set()` command
- Example on right is what I commonly use in my slides:
 - hide all code by default
 - hide all messages and warnings (from slides)
 - make figure resolution

```
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = FALSE,
 message = FALSE,
 warning = FALSE,
 fig.retina = 3)
...
...
```

# R Inline Code I



- If you just want to display some code (or at least format it like code) in the middle of a sentence, **place between a single backtick on either side**. If I mention `tidyverse` or `gapminder`, it formats the text as `in-line` code.
- To actually *execute R* code to output something in the middle of a sentence, put `r` as the first character inside the backticks, and then run the actual code such as `pi` is equal to 3.1415927.

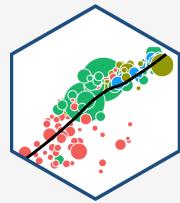
## Input

pi is equal to `r pi` .

## Output

pi is equal to 3.1415927.

# R Inline Code II



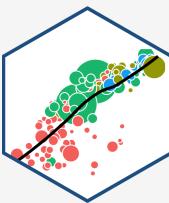
## Input

```
The average GDP per capita is `r gapminder %>%
mean(gdpPercap) %>% round(2)` with a standard
deviation of `r
round(sd(gapminder$gdpPercap),2)` .
```

## Output

The average GDP per capita is \$7215.33 with a standard deviation of \$9857.45.

# Writing Text with Markdown



- **Markdown** is a lightweight markup language geared towards HTML (i.e. the internet)
  - **Markup languages** used to add commands about how to display plain text
- Very simple and intuitive
- Write normal text as usual in any word processor
- Change font styling with tags (asterisks):
  - **\*italics text\*** creates *italics text*
  - **\*\*bold text\*\*** creates **bold text**

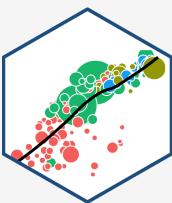
The screenshot shows the R Markdown editor on the left and the generated HTML output on the right. The R Markdown file (example.Rmd) contains the following content:

```
1 # Header 1
2
3 This is an R Markdown document. Markdown is a
4 simple formatting syntax for authoring webpages.
5 Use an asterisk mark to provide emphasis, such
6 as *italics* or **bold**.
7 Create lists with a dash:
8
9 - Item 1
10 - Item 2
11 - Item 3
12
13 ``
14 Use back ticks to
15 create a block of code
16 ``
17
18 Embed LaTeX or MathML equations,
19 $\frac{1}{n} \sum_{i=1}^n x_i$
20
21 Or even footnotes, citations, and a
22 bibliography. [^1]
23 [^1]: Markdown is great.
24
```

The generated HTML output (example.html) includes:

- A header section with the title "Header 1".
- The explanatory text from the R Markdown file.
- A bulleted list: "Item 1", "Item 2", "Item 3".
- A code block example: "Use back ticks to create a block of code".
- A mathematical equation:  $\frac{1}{n} \sum_{i=1}^n x_i$ .
- A note about footnotes, citations, and a bibliography, followed by a footnote reference [^1] and its content: "Markdown is great."

# Writing Text with Markdown: Lists



- Create an unordered list with lines of (- or + or \* ), e.g.:
- Markdown is great for taking notes quickly!

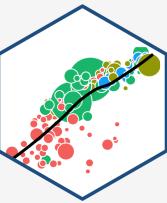
## Input

```
- item 1
- item 2
 - item 2a
- item 3
```

## Output

- item 1
- item 2
  - item 2a
- item 3

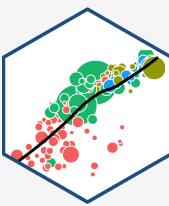
# Writing Text with Markdown: Headings & Comments



Markdown	Output
# Heading 1	<b>Heading 1</b>
## Heading 2	<b>Heading 2</b>
### Heading 3	<b>Heading 3</b>

Comment your code (will not print in output) with `<!-- Unprinted comments here -->` (this comes from html)

# Writing Text with Markdown: Tables



## Input

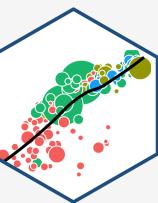
```
| Header 1 | Header 2 |
|-----|-----|
| Cell 1 | Cell 2 |
| Cell 3 | Cell 4 |
```

## Output

Header 1	Header 2
Cell 1	Cell 2
Cell 3	Cell 4

- For more complicated tables, there are other packages and techniques
  - LaTeX (pdf only)
  - `kableExtra` package
  - `huxtable` package (for regression tables)
  - `gt` package

# Writing Math I



- Add beautifully-formatted math with the `$` tag before and after the math, two  `$$` before/after for a centered equation
- In-line math example: `$1^2=\frac{\sqrt{16}}{4}` produces  $1^2 = \frac{\sqrt{16}}{4}$
- Centered-equation example:

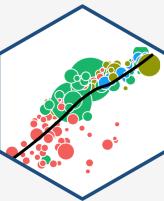
**Input**

```
$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^n (X_i - \bar{X})^2} $$
```

**Output**

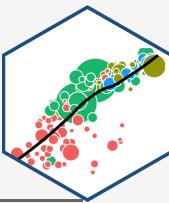
$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^n (X_i - \bar{X})^2}$$

# Writing Math II



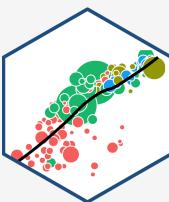
- Math uses a (much older) language called [LaTeX](#), used by mathematicians, economists, and others to write papers and slides with perfect math and formatting
  - I used to use for everything before I found [R](#) and [markdown](#)
  - Producing [pdf](#) or [html](#) output actually converts [markdown](#) files into *T<sub>E</sub>X* first! (See [the process described below](#))
  - Much steeper learning curve, [a good cheatsheet](#)
  - An extensive library of mathematical symbols, notation, formats, and ligatures, e.g.

# Writing Math III



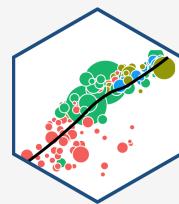
Input	Output
$\alpha$	$\alpha$
$\pi$	$\pi$
$\frac{1}{2}$	$\frac{1}{2}$
$\hat{x}$	$\hat{x}$
$\bar{y}$	$\bar{y}$
$x_{1,2}$	$x_{1,2}$
$x^{a-1}$	$x^{a-1}$
$\lim_{x \rightarrow \infty}$	$\lim_{x \rightarrow \infty}$
$A = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$ <pre>\$A=\begin{bmatrix} a_{1,1} &amp; a_{1,2} \\ a_{2,1} &amp; a_{2,2} \end{bmatrix}\$</pre>	$A = \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$

# Citations, References, and Bibliography



- Manage your citations and bibliography automatically with `.bib` files
- First create a `.bib` file to list all of your references in
  - You can do this in R via: `File -> New File -> Text File` (and save with `.bib` at the end)
  - See `examplebib.bib` in this repository used in this document
  - At the top of your YAML header in the main document, add `bibliography: examplebib.bib` so R knows to pull references from this file
  - For each reference, add information to a `.bib` file, like so:

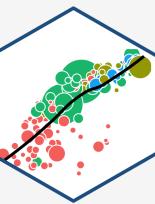
# An Example .bib File



```
@article{safner2016,
author = {Ryan Safner},
year = {2016},
journal = {Journal of Institutional Economics},
title = {Institutional Entrepreneurship, Wikipedia,
and the Opportunity of the Commons},
volume = {12},
number = {4},
pages = {743-771}
}
```

- A `.bib` file is a plain text file with entries like this
- Classes for `@article`, `@book`, `@collectedwork`, `@unpublished`, etc.
  - Each will have different keys needed (e.g. `editor`, `publisher`, `address`)
- First input after the `@article` is your **citation key** (e.g. `safner2016`)
  - Whenever you want to cite this article, you'll invoke this key

# An Example .bib File

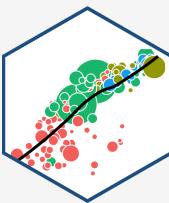


- Whenever you want to cite a work in your text, call up the **citation key** with `@`, like so: `@safner2016[]`, which produces (Safner, 2016)
- You can customize citations, e.g.:

Write	Produces
<code>[@Safner2016]</code>	(Safner, 2016)
<code>@Safner2016</code>	Safner 2016
<code>-@Safner2016</code>	(2016)
<code>@Safner2016[p. 743-744]</code>	(Safner, 2016, p.743-744)

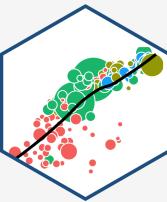
- BibTeX will automatically collect all works cited at the end and produce a **bibliography** according to a style you can choose

# Reference Management Software



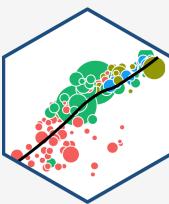
- For more information and examples, see [R Studio's R Markdown Guide on Bibliographies](#)
- Lot of programs can help you manage references and export complete `.bib` files to use with [R Markdown](#)
  - [Mendeley](#) and [Zotero](#) are free and cross-platform
  - I use [Papers](#) (Paid and Mac only)
  - Simplest program (what I use) that makes `.bib` files is [Bibdesk](#)

# Plain-Text Editors



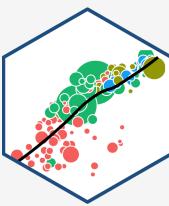
- Markdown files are **plain text** files and can be edited in *any* text editor
  - something as basic (and boring!) as "**Notepad**," for example
  - many good text editors out there, I like Typora or Ulysses (Mac only) for writing (and previewing) Markdown in a simple interface, with no distractions
- Any good editor will have **syntax highlighting** and **coloring** when you use tags (like **bold**, *italic*, `code`, and `code #comments`).

# R Studio is My Text Editor of Choice

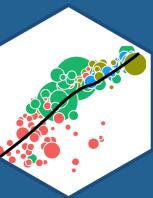


- Honestly, I write **everything** in R Studio's text editor
  - Syntax highlighting
  - Actually can *run* R code, autocomplete, etc
  - Can render the markdown to an output format: html, pdf, etc.
- You can *write* R code in other text editors, but you can't *execute* them outside of *R Studio* (or the command line, but that's too advanced.) Same with actually rendering your markdown to an output (pdf, html, etc)

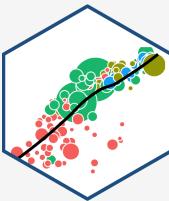
# Tips with Markdown



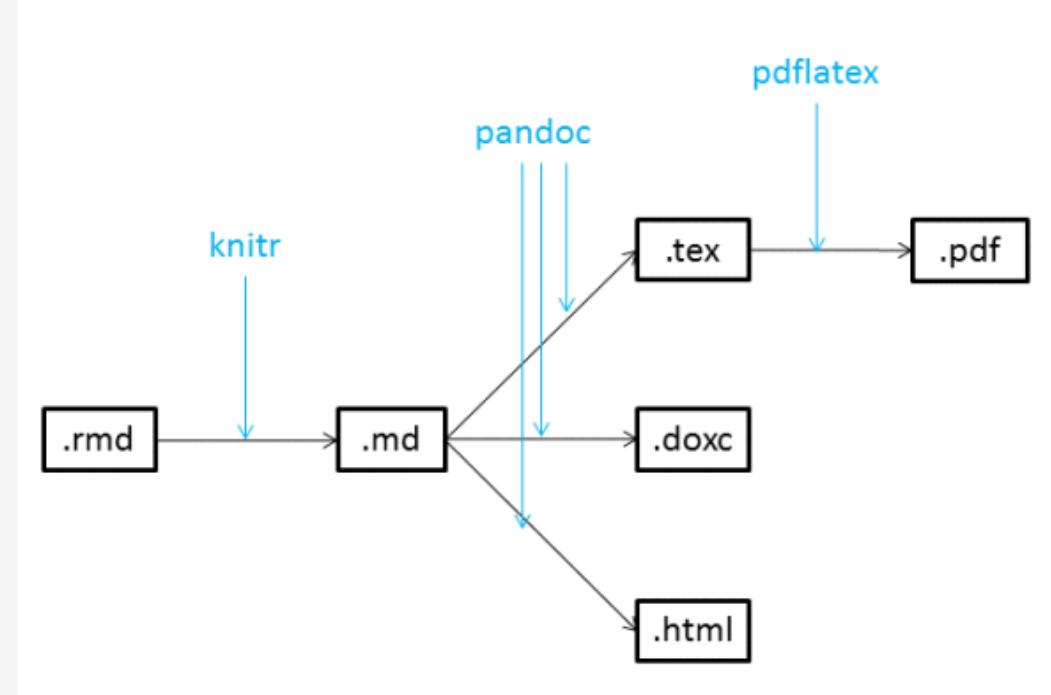
- Empty space is *very important* in markdown
- Lines that begin with a space may not render properly
- Math that contains spaces *between* the dollar-signs may not render properly
- Moving from one type of content to another (e.g. a heading to a list to text to an equation to text) requires *blank lines between them* to work
- Here is a [great general tutorial on markdown syntax](#)



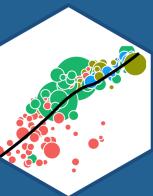
# Compiling Your Documents



- When you are ready, you "compile" your markdown and code into an output format using:
- [knitr](#)<sup>1</sup>, an R package that "knits" your R code and markdown `.Rmd` into a `.md` file for:
- [pandoc](#) is a "swiss-army knife" utility that can convert between *dozens* of document types
- All you need to do is click the `Knit` button at the top of the text editor!

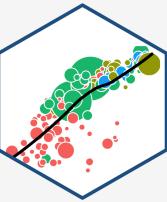


<sup>1</sup> `knitr` also relies on the `rmarkdown` package, which will probably be installed when you first `knit`.



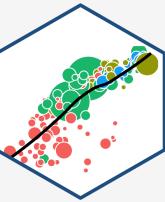
# R Projects

# R Projects I



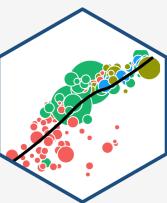
- A **R Project** is a way of systematically organizing your **R** history, working directory, and related files in a single package
- Can easily be sent to others who can reproduce your work easily
- Connects well with version control software like GitHub
- Can open multiple projects in multiple windows

# R Projects II



- Projects solve all of the following problems:
  1. Organizing your files (data, plots, text, citations, etc)
  2. Having an accessible working directory (for loading and saving data, plots, etc)
  3. Saving and reloading your commands history and preferences
  4. Sending files to collaborators, so they have the same working directory as you

# Creating a Project I



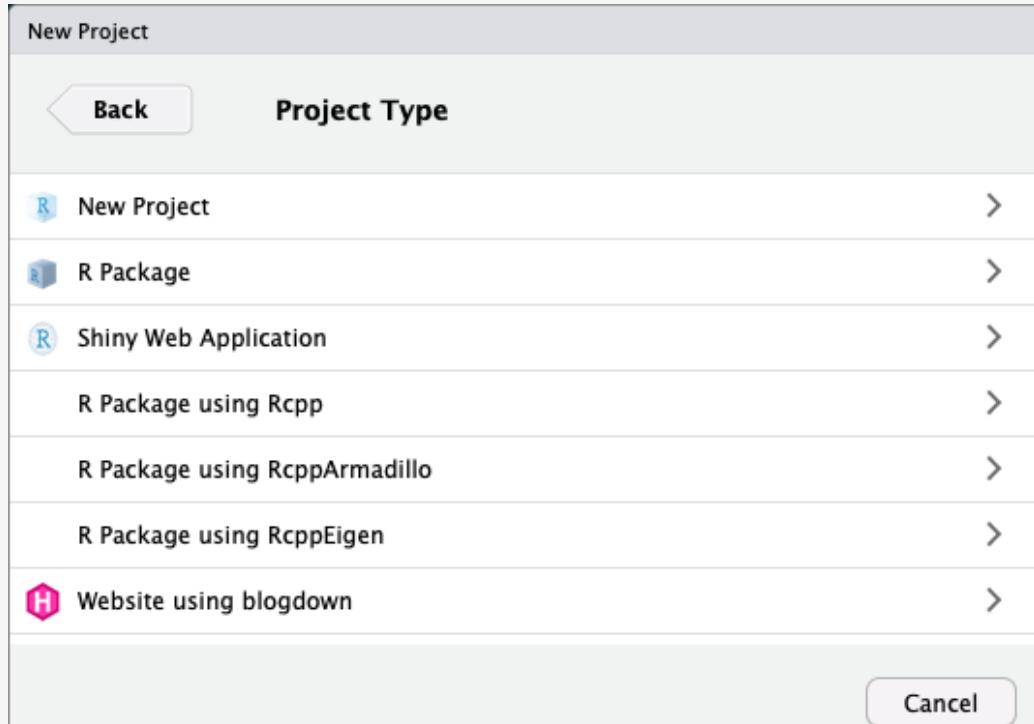
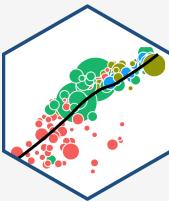
New Project

## Create Project

- **New Directory**  
Start a project in a brand new working directory >
- **Existing Directory**  
Associate a project with an existing working directory >
- **Version Control**  
Checkout a project from a version control repository >

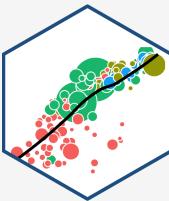
Cancel

# Creating a Project II



- In almost all cases, you simply want a **New Project**
- For more advanced uses, your project can be an **R Package** or a **Shiny Web Application**
- If you have other packages that create templates installed (as I do, in the previous image), they will also show up as options

# Creating a Project III



New Project

Back Create New Project

Directory name:

Create project as subdirectory of:

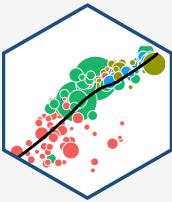
Create a git repository

Use packrat with this project

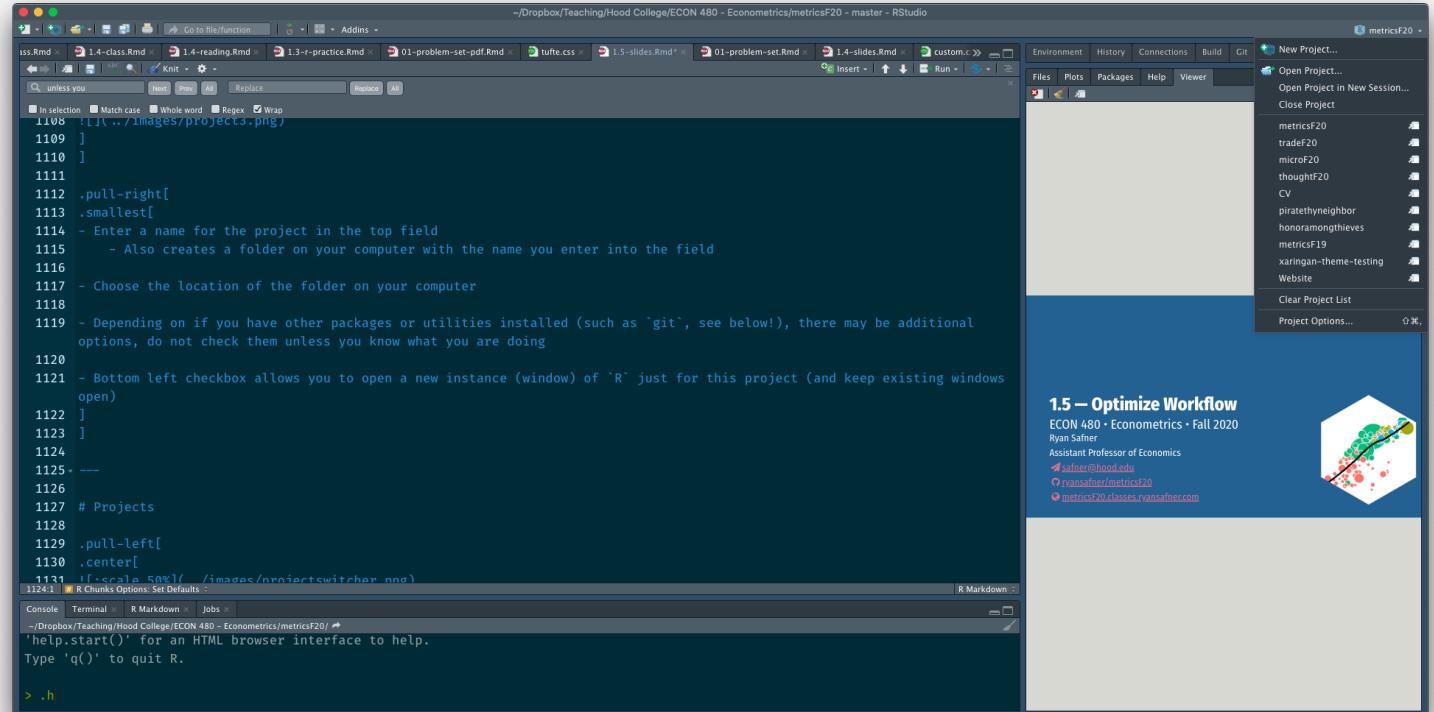
Open in new session

- Enter a name for the project in the top field
  - Also creates a folder on your computer with the name you enter into the field
- Choose the location of the folder on your computer
- Depending on if you have other packages or utilities installed (such as `git`, see below!), there may be additional options, do not check them unless you know what you are doing
- Bottom left checkbox allows you to open a new instance (window) of `R` just for this project (and keep existing windows open)

# Projects

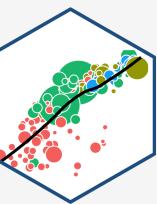


Switch between each project (Window) on your computer (this is on a Mac).



- At top right corner of RStudio
    - Click the button to the right of the name to open in a new window!

# Loading Others' Projects



ryansafner / workflow

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags

ryansafner Updated presentation deb947a on Dec 3, 2018 6 commits

Bibliography	Updated for viewing	2 years ago
Data	Updated for viewing	2 years ago
Figures	More complete paper	2 years ago
Presentation	Updated presentation	2 years ago
Scripts	More complete paper	2 years ago
.gitignore	Initial files	2 years ago
Example_paper.Rmd	More complete paper	2 years ago
Example_paper.pdf	More complete paper	2 years ago
README.md	Fix Readme bullets	2 years ago
workflow.Rproj	Initial files	2 years ago

README.md

## Managing Your Workflow with R Projects

This repository is an R Project used to show the benefits of using R Studio's Project feature (in combination with R Markdown and R) for data analysis, writing papers, and presentations.

About

Managing your workflow with R Projects

Readme

Releases

No releases published [Create a new release](#)

Packages

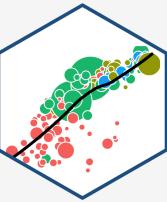
No packages published [Publish your first package](#)

Languages

R 59.1% TeX 40.9%

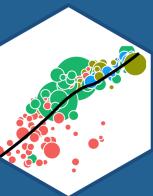
This project is on [GitHub](#), click the green button, download to your computer, open `.Rproj` file in R Studio

# A Good File Structure



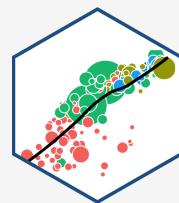
Files Plots Packages Help Viewer		
New Folder	Delete	Rename More
Home > Dropbox > Teaching > R > workflow		
Name	Size	Modified
..		
Bibliography		
Data		
Figures		
Presentation		
Scripts		
Example_paper.pdf	408.1 KB	Nov 29, 2018, 4:36 PM
README.md	50 B	Nov 28, 2018, 10:25 PM
.gitignore	40 B	Nov 28, 2018, 10:25 PM
workflow.Rproj	204 B	Nov 28, 2018, 10:25 PM
Example_paper.Rmd	4.2 KB	Nov 29, 2018, 4:36 PM

- Look through this on your own
- Read the [README](#) of this repository on GitHub for instructions (automatically shows on the main page)
- Look at the [Example\\_paper.Rmd](#)
  - Uses data from **Data** folder
  - Uses `.R` scripts from **Scripts** folder
  - Uses figures from **Figures** folder
  - Uses [bibexample.bib](#) from **Bibliography** folder



# Version Control

# Have You Done This?

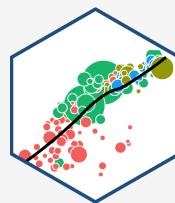


FINAL.doc!



FINAL\_rev.2.doc

# Have You Done This?

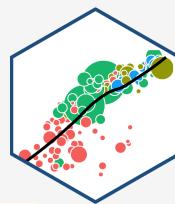


FINAL\_rev.6.COMMENTS.doc

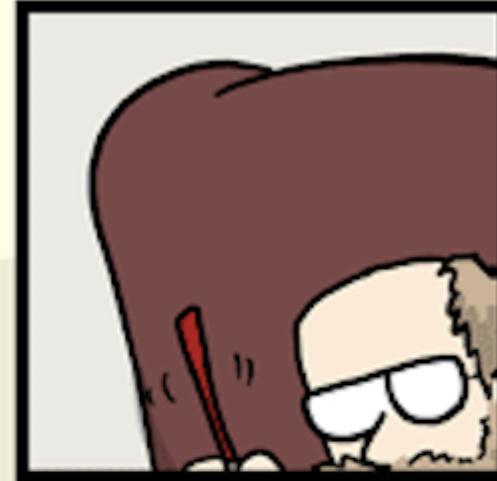
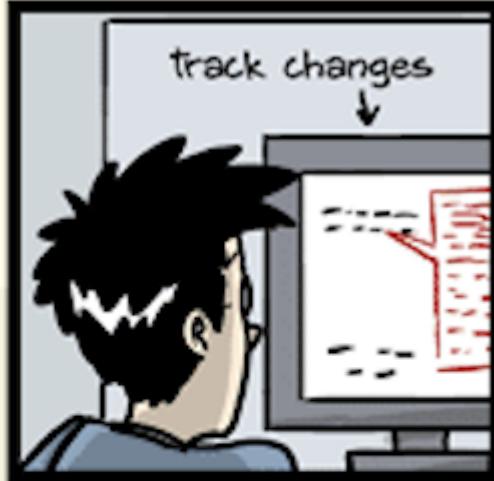


FINAL\_rev.8.comments5.  
CORRECTIONS.doc

# Have You Done This?

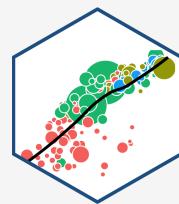


JORGE CHAM © 2012



FINAL\_rev.22.comments49.  
corrections.10.#@\$%WHYDID  
ICOMETOGRAD SCHOOL????.doc

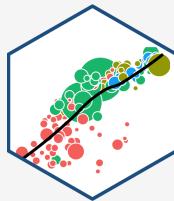
# Do You Want to Be Able To



- Keep your files backed up
- Track changes
- Collaborate on the same files with others
- Edit files on one computer and then open and continue working on another?



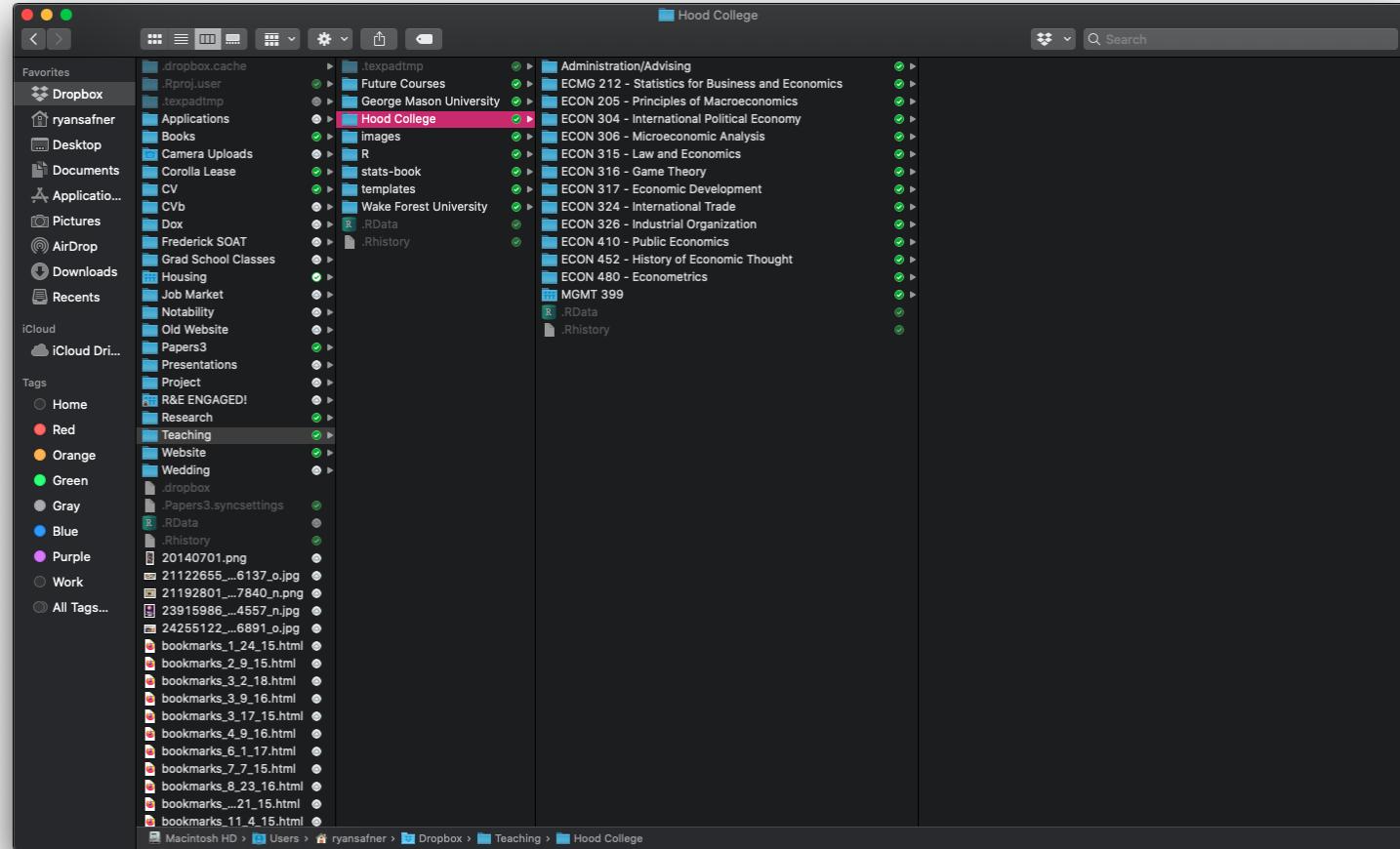
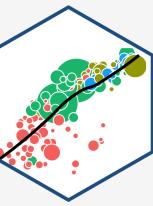
# The Training-Wheels Version



[Dropbox.com](https://www.dropbox.com)

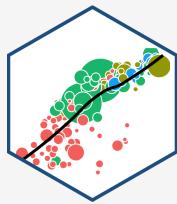
- Register an account for free
- Set up a location on your computer for the [Dropbox/](#) folder
- Anything you put in this folder will sync to the cloud
  - As soon as you change files, they *automatically* update and sync!
  - Can download any of these files from the *website* on any device
  - Set this up on multiple computers so when you change a file on one, it updates on all the others!

# The Training-Wheels Version

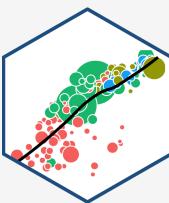


My Dropbox - my life goes here

# The Training-Wheels Version

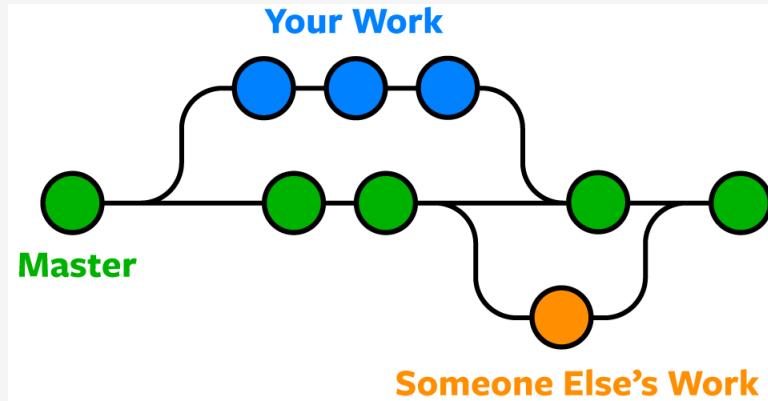
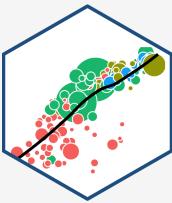


# The Expert Version



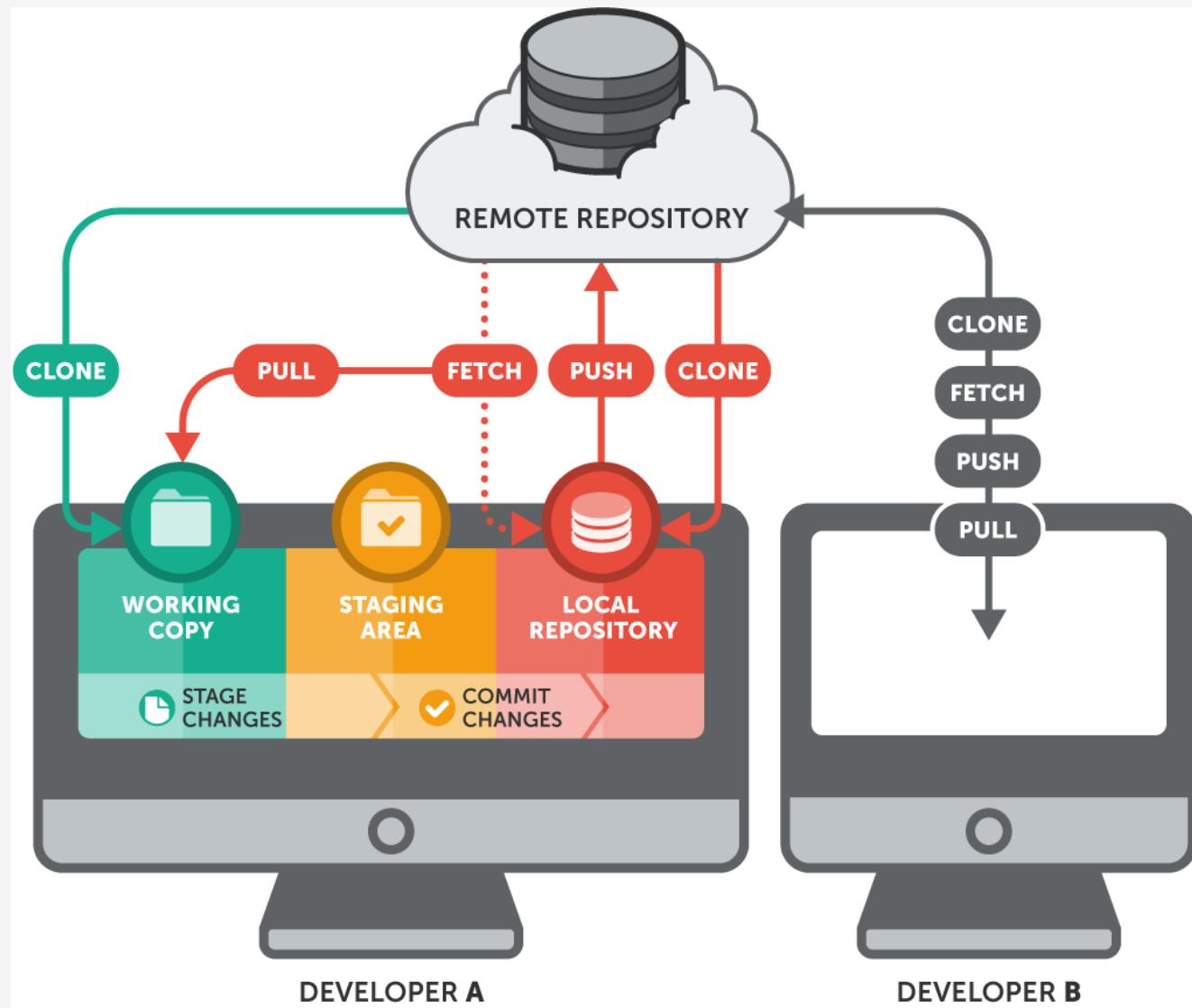
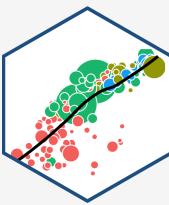
- Git is an "open source distributed version control system" *widely* used in the software development industry
- **Track changes on steroids** (if MS Word's Track Changes and Dropbox had a baby)
  - Organize folders/files to track (a "repository")
  - Take a snapshot of all of your files (a "commit") with "comments"
  - push these to the cloud

# The Expert Version

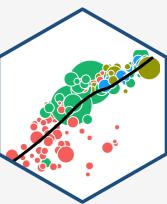


- Shows history (`versions`) of files with comments
  - Can `fork` or `branch` repository into multiple versions at once
  - Good for "testing" things out without destroying old versions!
  - `revert` back to original versions as needed

# The Expert Version

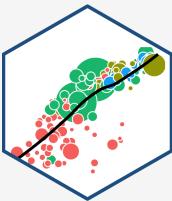


# The Expert Version



- Requires *some* advanced set up, see [this excellent guide](#)
- R Studio integrates git and github commands nicely

# This Class on GitHub



Screenshot of the GitHub repository page for `ryansafner/metricsF20`. The repository has 1 branch and 0 tags. It contains several files and folders:

- `content`: Format HW 1 page (3 days ago)
- `data`: Add HW 1 (3 days ago)
- `pandoc`: Initial major commit, shell from metricsF19, customize (3 months ago)
- `static`: Format HW 1 page (3 days ago)
- `themes/ath-tufte-hugo_18-19`: Add 1.2 content (12 days ago)
- `.gitignore`: Add 1.3 pdf (finally worked) and fix 1.4 practice formatting (5 days ago)
- `README.md`: Initial commit (3 months ago)
- `config.yaml`: Fix class meeting times! (16 days ago)
- `index.Rmd`: Initial major commit, shell from metricsF19, customize (3 months ago)
- `metricsF20.Rproj`: Initial major commit, shell from metricsF19, customize (3 months ago)

The `README.md` file contains the following content:

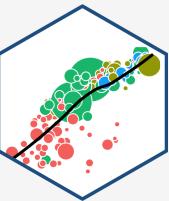
```
metricsF20
Course website for ECON 480 - Econometrics, Fall 2020 semester @
```

Screenshot of the GitHub repository page for `ryansafner/metricsF20`. The repository has 1 branch and 0 tags. It shows a history of commits:

- Commits on Aug 28, 2020:
  - Format HW 1 page (ryansafner committed 3 days ago)
  - Add HW 1 (ryansafner committed 3 days ago)
  - Moar ggplot extensions! (ryansafner committed 4 days ago)
  - Update some 1.3 class appendices (ryansafner committed 4 days ago)
- Commits on Aug 27, 2020:
  - Add 1.3 R practice answers (ryansafner committed 5 days ago)
  - Add 1.3 pdf (finally worked) and fix 1.4 practice formatting (ryansafner committed 5 days ago)
  - Add 1.4 pdf (ryansafner committed 5 days ago)
  - Minor tweaks to practice 1.4 (ryansafner committed 5 days ago)
- Commits on Aug 26, 2020:
  - Add 1.4 content (ryansafner committed 5 days ago)

[github.com/ryansafner/metricsF20](https://github.com/ryansafner/metricsF20)

# Most Packages Start on GitHub



Screenshot of the GitHub repository [tidyverse / tidyverse](https://github.com/tidyverse/tidyverse). The repository has 72 pull requests, 751 stars, and 156 forks. It contains 234 commits, 2 branches, 5 releases, and 24 contributors. The latest commit was 6 days ago. The repository page includes a list of recent commits:

Commit	Message	Time Ago
hadley Update paper.md	Latest commit 9a4cbe9	6 days ago
.github	Add move bot config.	last year
R	Only look at packages when considering conflicts	4 months ago
man	Add repos option to tidyverse_(deps,update) (#82)	4 months ago
pkgdown/favicon	Use retina logo	8 months ago
revdep	Re-run revdep checks	4 months ago
tests	Add first test	last year
vignettes	Update paper.md	6 days ago
.Rbuildignore	Merge branch 'master' into joss-paper	3 months ago
.covignore	use_tidy_ci()	8 months ago
.gitignore	Remove built website	8 months ago
.travis.yml	use_tidy_ci()	3 months ago
DESCRIPTION	Update for new R version	4 months ago
LICENSE	Include GPL license text	3 years ago

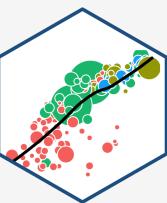
[github.com/tidyverse/tidyverse](https://github.com/tidyverse/tidyverse)

Screenshot of the GitHub repository [jennybc / gapminder](https://github.com/jennybc/gapminder). The repository has 14 pull requests, 147 stars, and 183 forks. It contains 126 commits, 1 branch, 4 releases, and 4 contributors. The latest commit was 2 years ago. The repository page includes a list of recent commits:

Commit	Message	Time Ago
jennybc	Add a test so country_codes stay fixed	Latest commit ddfe773 on Mar 6
.aspell	Use .aspell	2 years ago
R	Document country_codes	2 years ago
data-raw	Be careful about preserving the int type of iso_num	6 months ago
data	Be careful about preserving the int type of iso_num	6 months ago
inst/extdata	Fix country codes for North Korea	last year
man	Be careful about preserving the int type of iso_num	6 months ago
tests	Add a test so country_codes stay fixed	6 months ago
.Rbuildignore	correct area in bubble plot; fixes #7	4 years ago
.gitignore	"compile notebook" for cleaning scripts; fixes #1	4 years ago
DESCRIPTION	attempt to get hyperlinks recognized (#28)	6 months ago
NAMESPACE	import tibble for printing purposes	3 years ago
NEWS.md	Fix country codes for North Korea	last year
README.Rmd	attempt to get hyperlinks recognized (#28)	6 months ago
README.md	Be careful about preserving the int type of iso_num	6 months ago
cran-comments.md	Update cran-comments	2 years ago

[github.com/jennybc/gapminder](https://github.com/jennybc/gapminder)

# My Workflow (that I suggest to you)



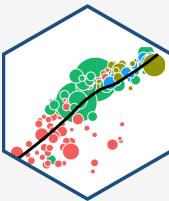
1. Create a new repository on Github.\*
2. Start a New R Project in R Studio (link it to the github repository\* - [see guide](#))
3. Create a logical file system ([see example](#)), such as:

```
project # folder on my computer (the new working directory)
|
|- Data/ # folder for data files
|- Scripts/ # folder .R code
|- Bibliography/ # folder for .bib files
|- Figures/ # folder to plots and figures to
|- paper.Rmd # write document here
```

4. Write document in `paper.Rmd`, loading/saving files from/to various folders in project
  - e.g. load data like `df<-read_csv("Data/my_data")`; save plots like `ggsave("Figures/p.png")`
5. Knit document to `pdf` or `html`.
6. Occasionally, `stage` and `commit` changes with a description, `push` to GitHub.\*

\* Optional and a bit advanced, remember this is *my workflow*.

# Resources



1. R Studio's [R Markdown Cheatsheet](#) for a quick overview of R markdown
2. R Studio's [Overview of R Markdown](#) for some tutorials
3. R Studio's [R Markdown Reference Guide](#) for more specific options and issues
4. Kieran Healey's [The Plain Person's Guide to Plain Text Social Science](#) on managing workflow with plain text files, R, and Git
5. Yihui Xie's (and coauthors) [R Markdown: the Definitive Guide](#) on R Markdown syntax and customization options
6. Hadley Wickham's (and Garrett Grolemund) [R for Data Science](#) on how to use R and R Markdown for data science work
7. Jenny Bryan's [Happy Git with R](#) on how to use git and GitHub with R as a version control system