You can't do data science in a GUI

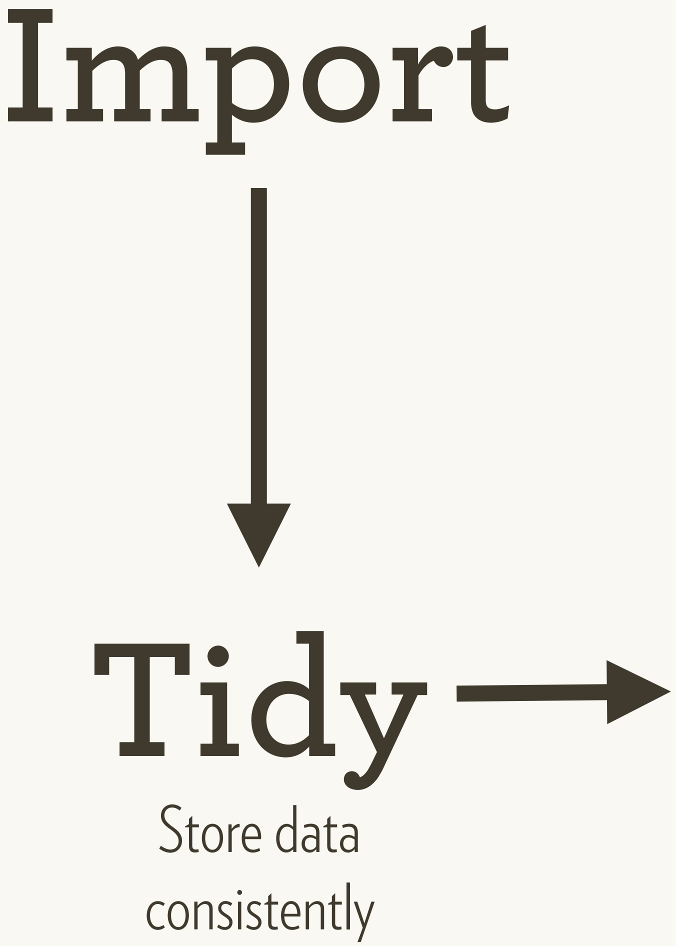
March 2018

Hadley Wickham
@hadleywickham
Chief Scientist, **RStudio**

Data science is the process
by which data becomes
understanding, knowledge
and insight

Data science is the process
by which data becomes
understanding, knowledge
and insight

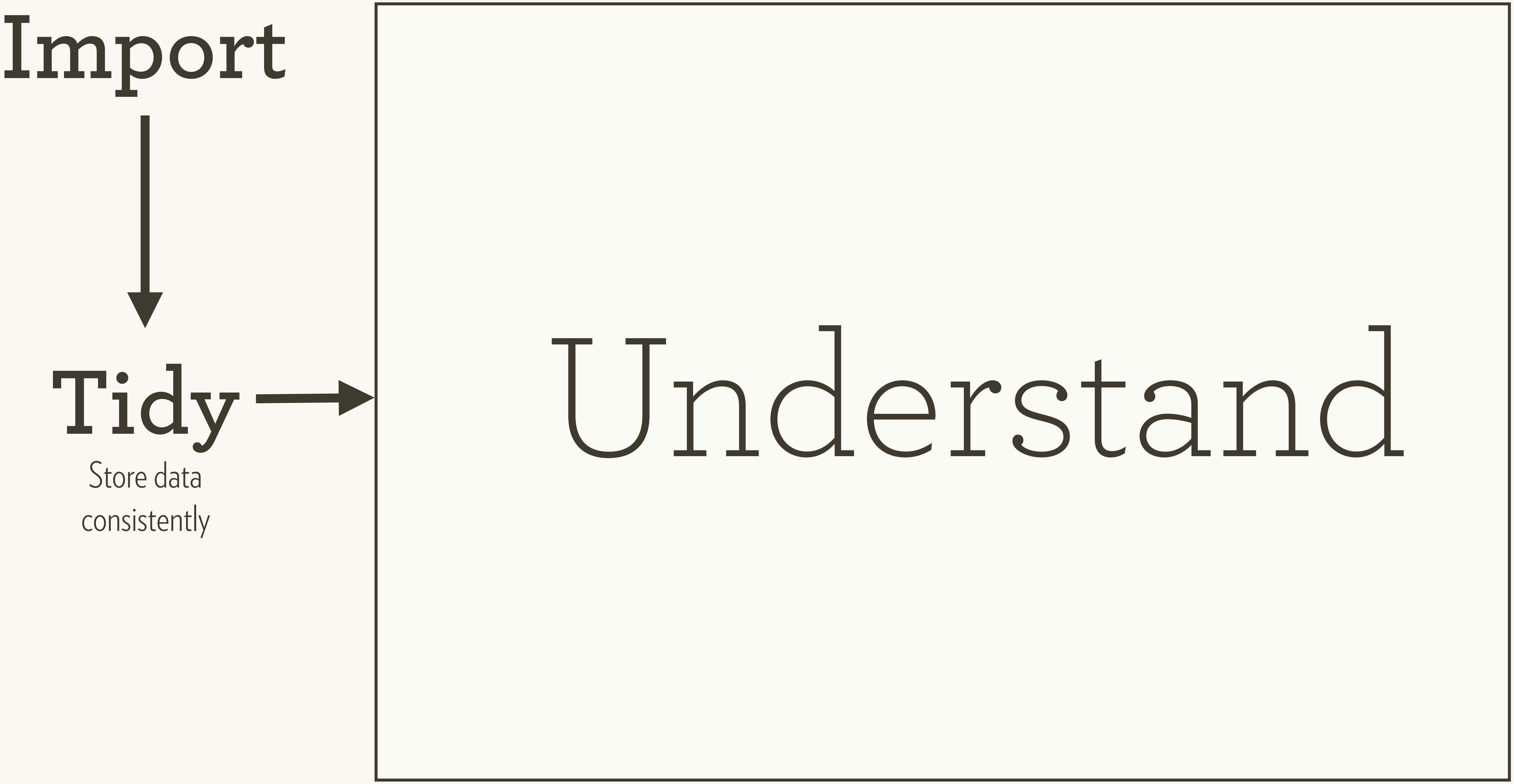
Data science is the process by which data becomes understanding, knowledge and insight



Import

Tidy

Store data consistently

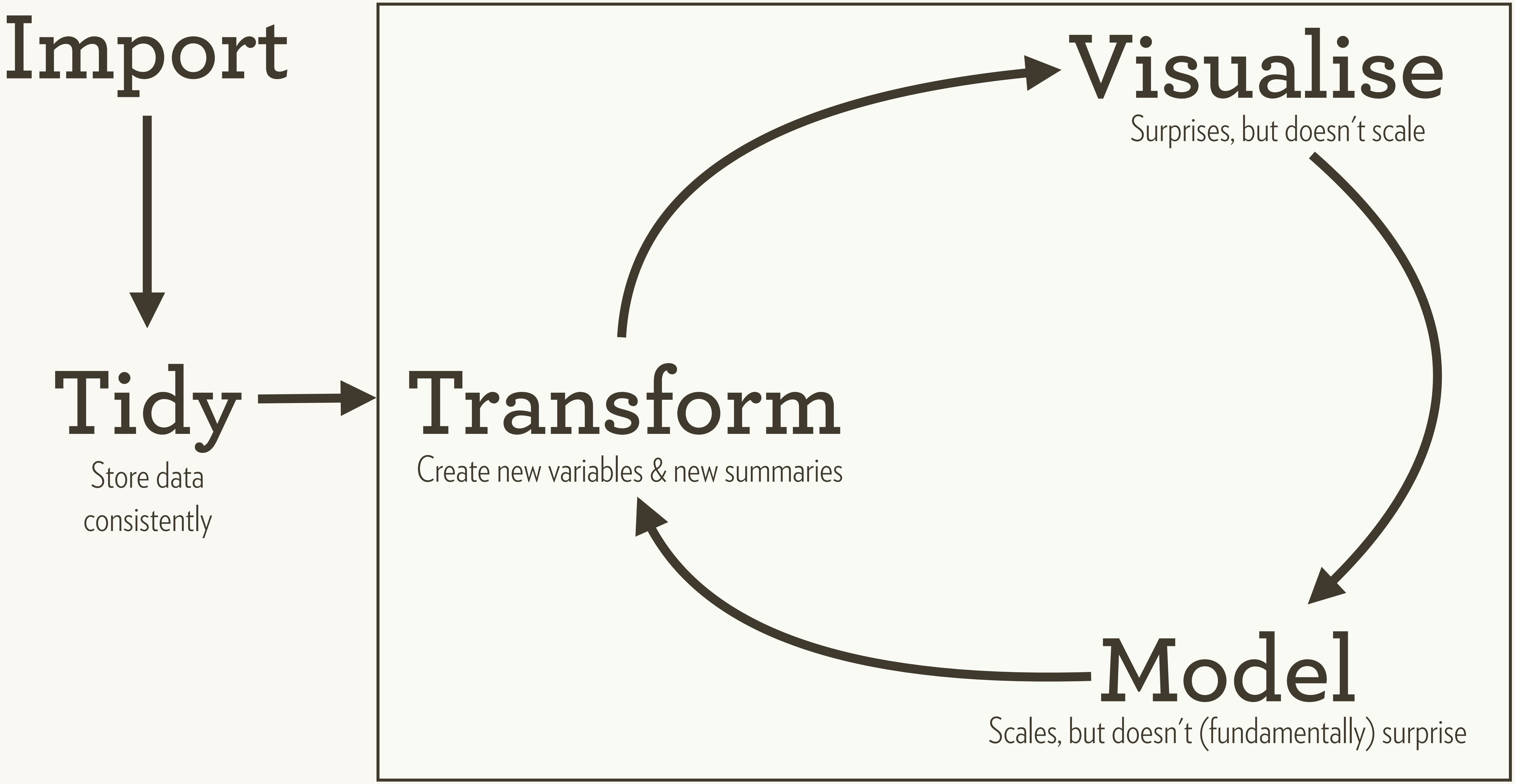


Import

Tidy

Store data consistently

Understand



Import

Tidy

Store data consistently

Transform

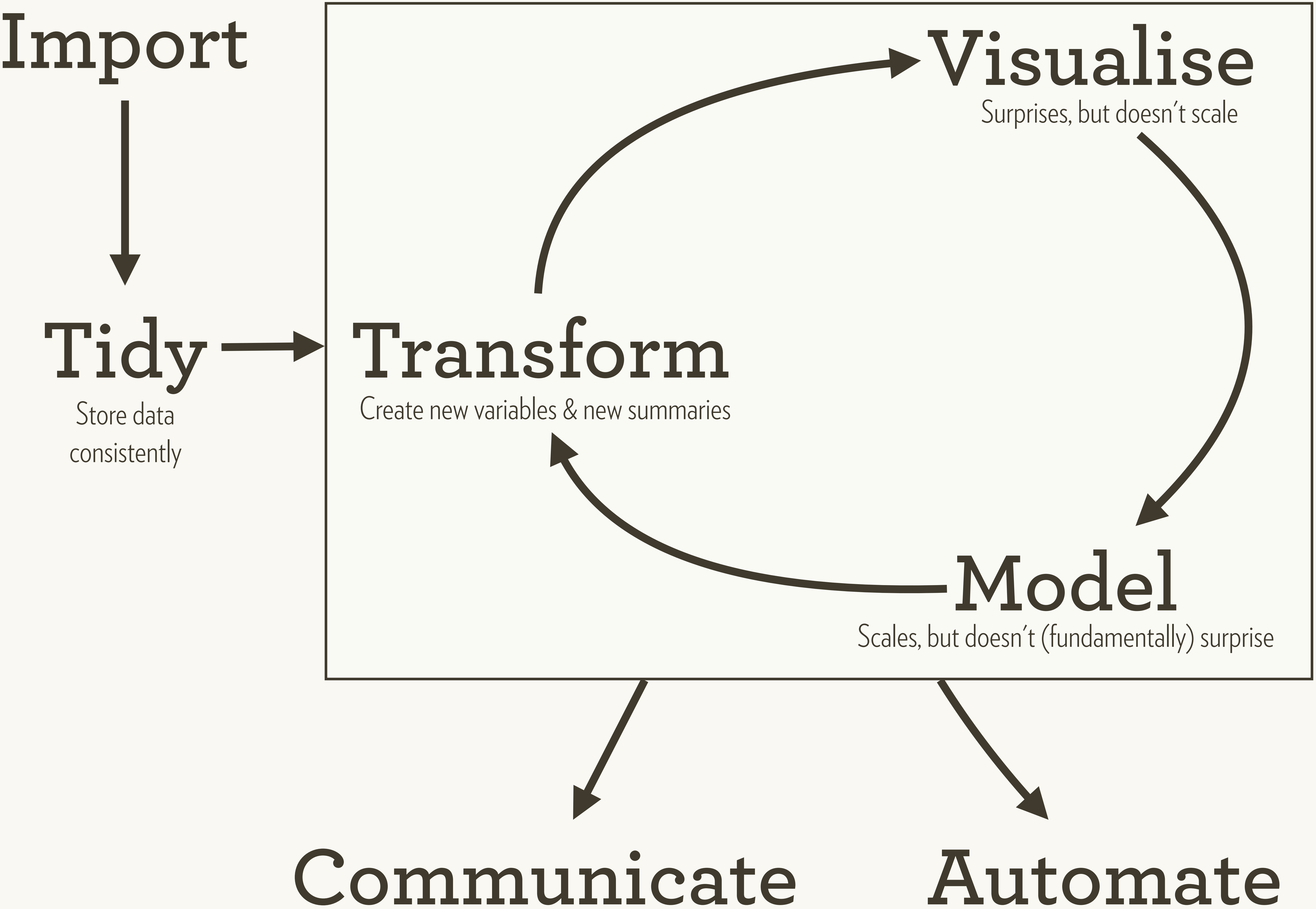
Create new variables & new summaries

Visualise

Surprises, but doesn't scale

Model

Scales, but doesn't (fundamentally) surprise



Import

Tidy

Store data consistently

Transform

Create new variables & new summaries

Visualise

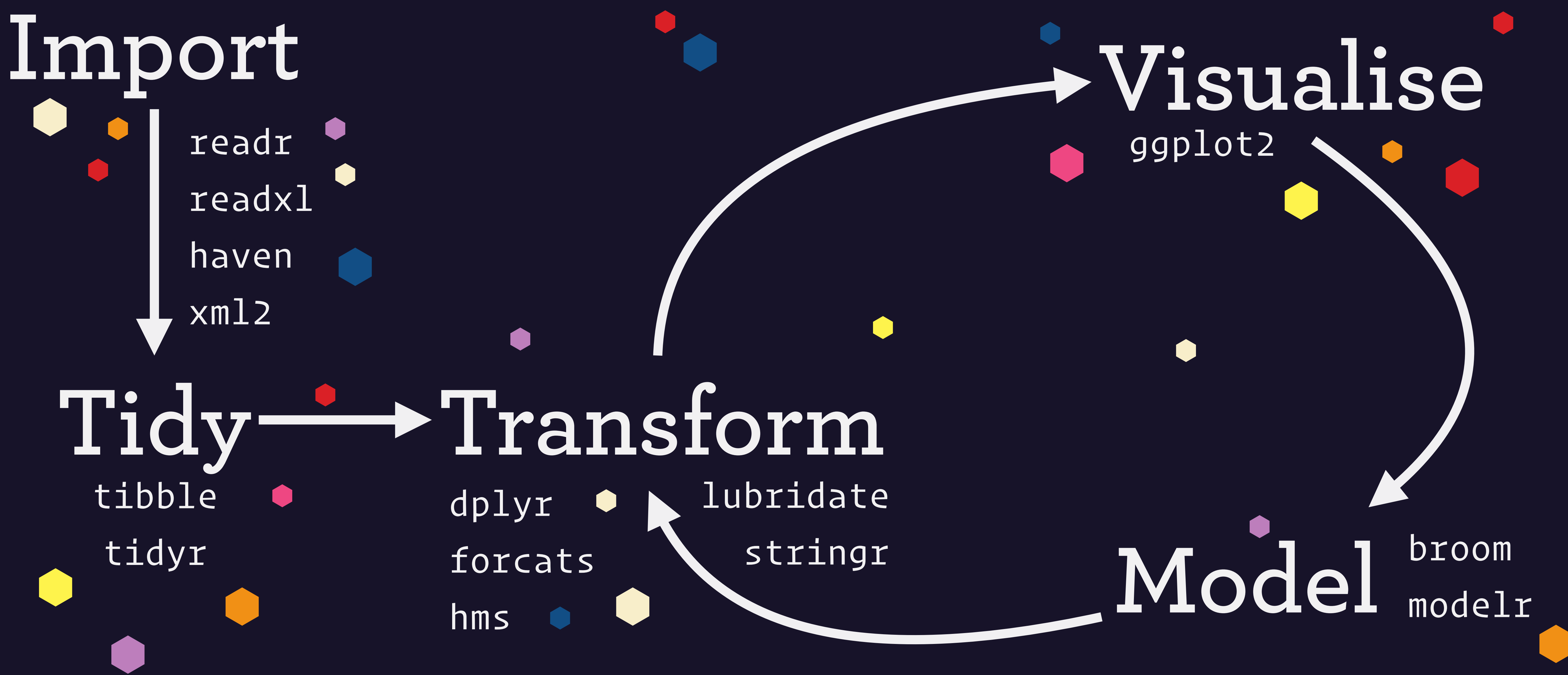
Surprises, but doesn’t scale

Model

Scales, but doesn’t (fundamentally) surprise

Communicate

Automate



Import

readr

readxl

haven

xml2

Tidy

tibble

tidyr

Transform

dplyr

forcats

hms

lubridate

stringr

Visualise

ggplot2

Model

broom

modelr

purrr

magrittr

Program

R packages for data science

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying philosophy and common APIs.

Install the complete tidyverse with:

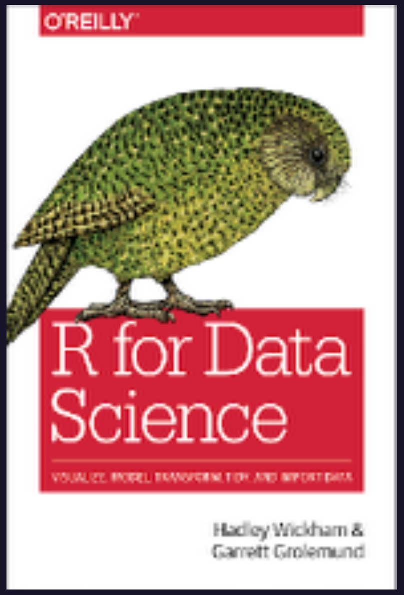
```
install.packages("tidyverse")
```

Learn the tidyverse

See how the tidyverse makes data science faster, easier and more fun with "R for Data

Building tidy tools

tidyverse.org



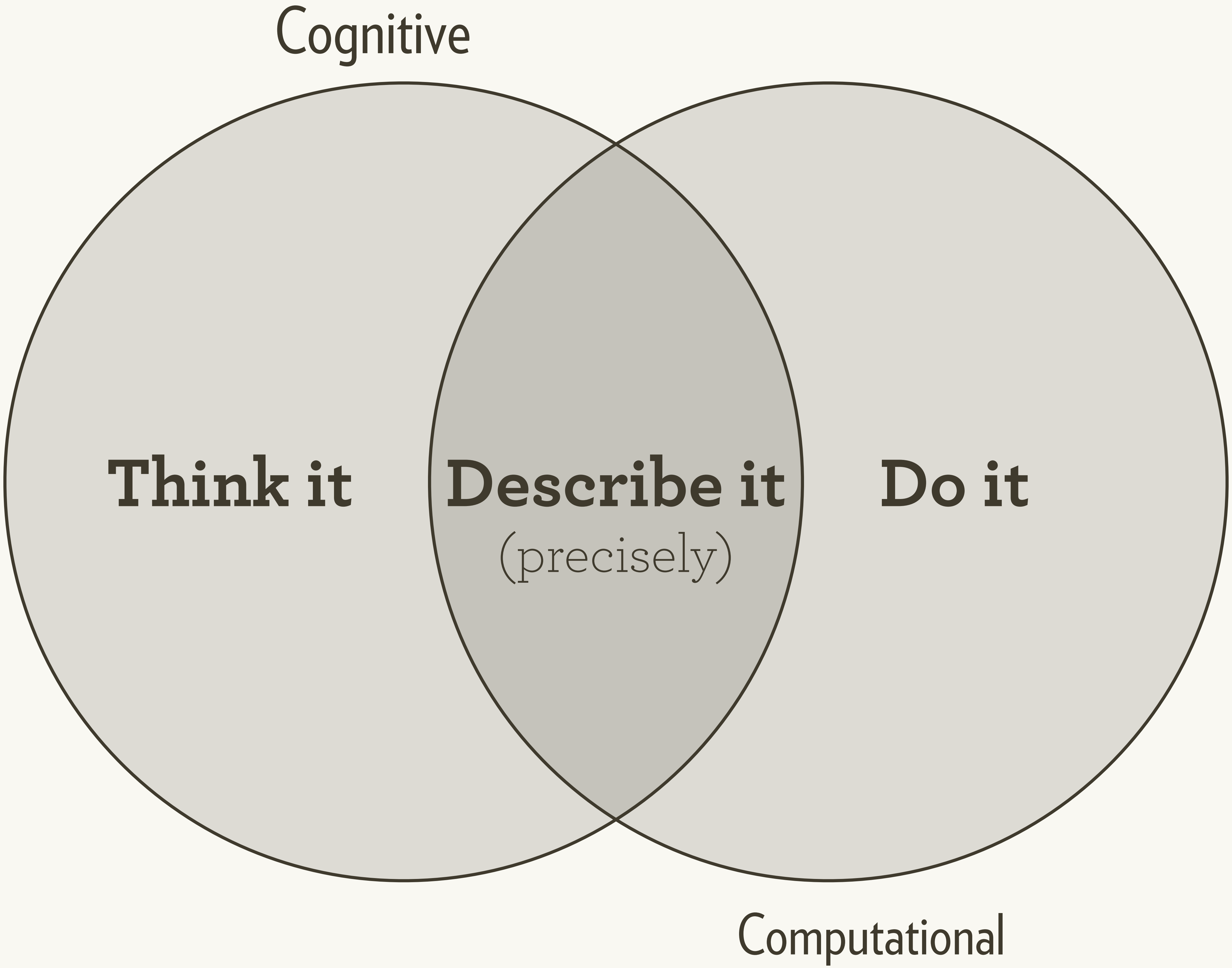
R for Data Science

VISUALIZE, MODEL, TRANSFORM, TIDY, AND IMPORT DATA

Hadley Wickham & Garrett Grolemund

r4ds.had.co.nz

Why program?



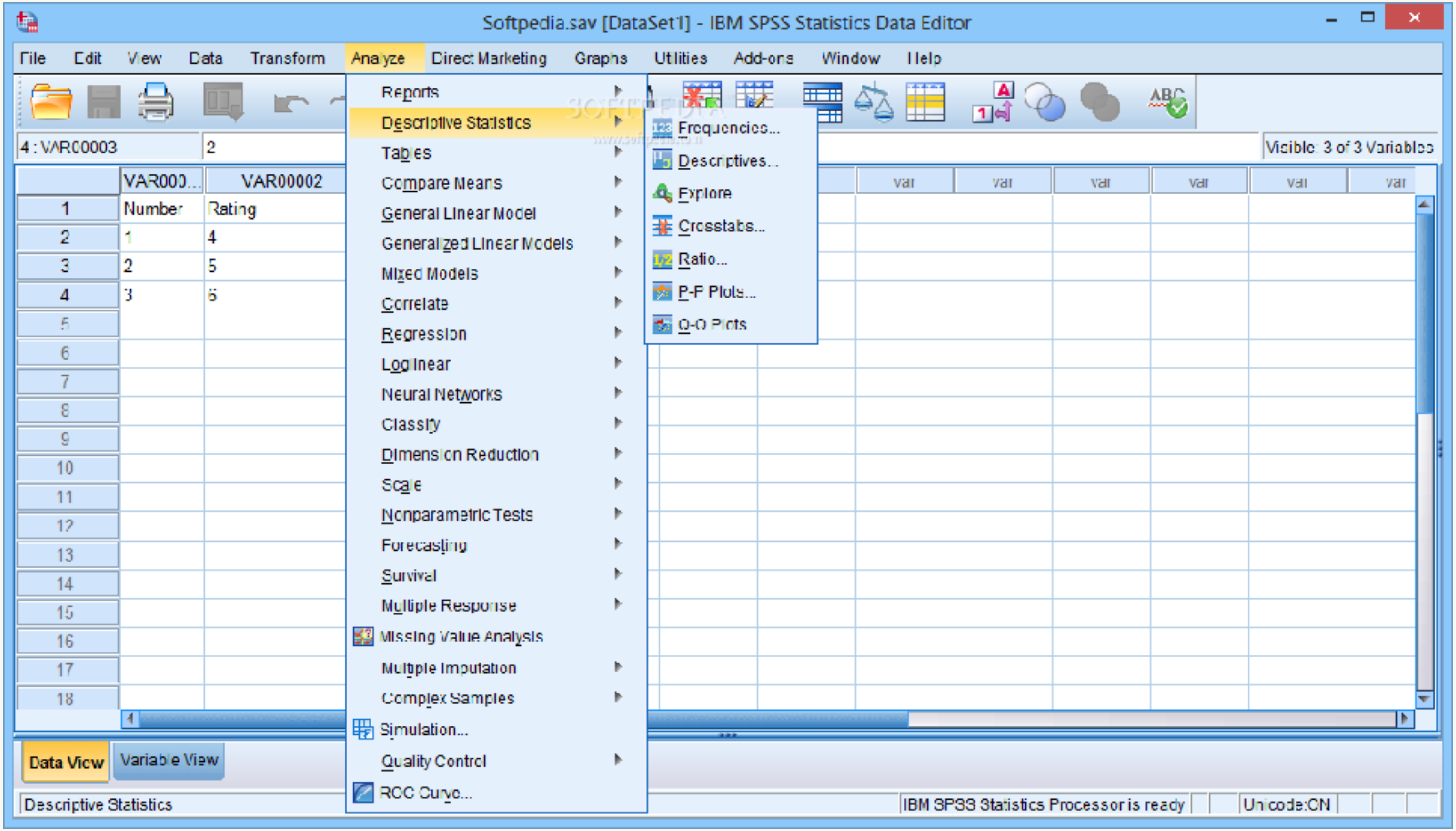
Cognitive

Think it

Describe it
(precisely)

Do it

Computational



4 : VAR00003 2

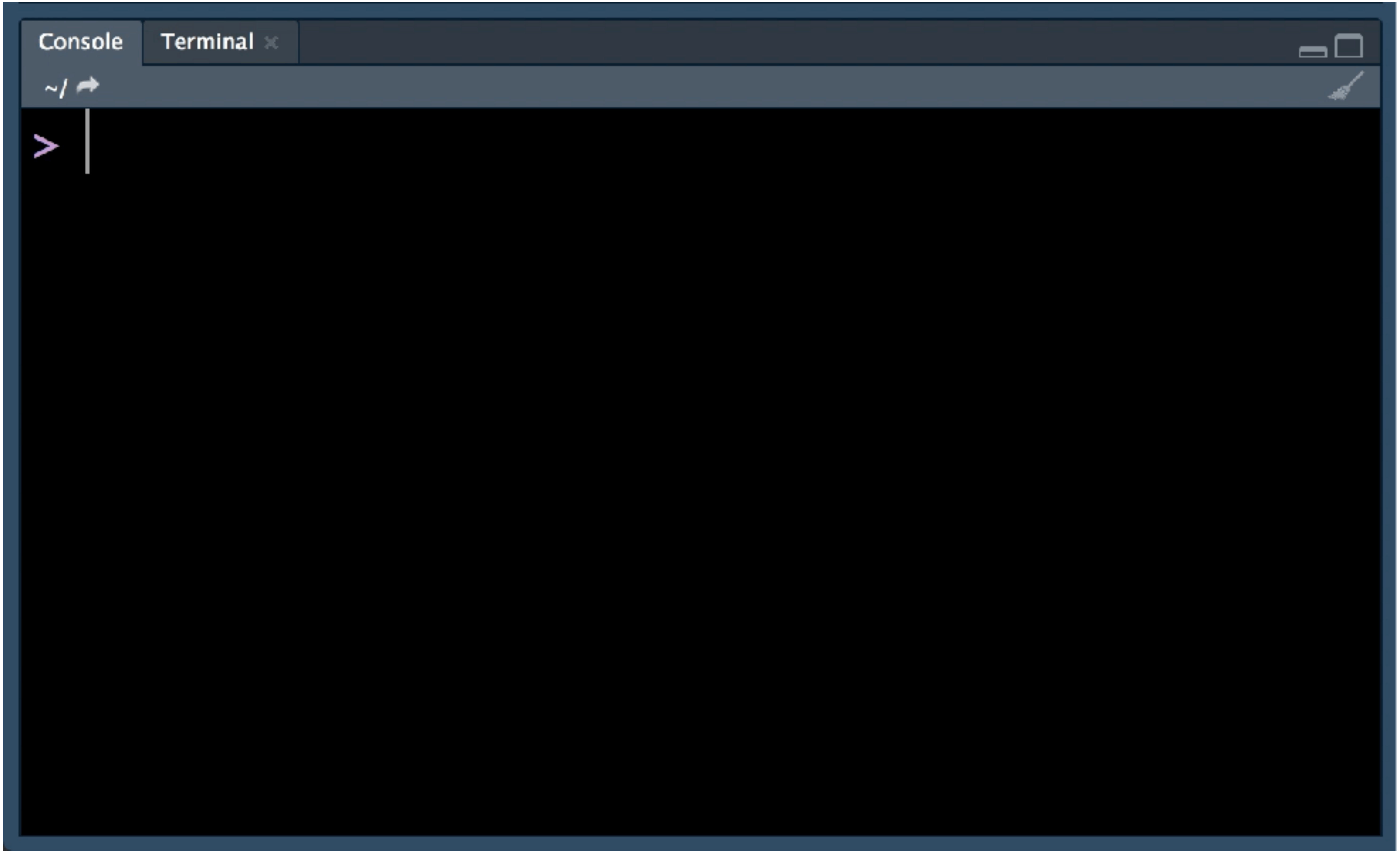
Visible: 3 of 3 Variables

	VAR000...	VAR00002
1	Number	Rating
2	1	4
3	2	5
4	3	6
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		

Descriptive Statistics

IBM SPSS Statistics Processor is ready

Unicode:ON



Console
Terminal
~/

```
>
```

Programming languages are **languages**

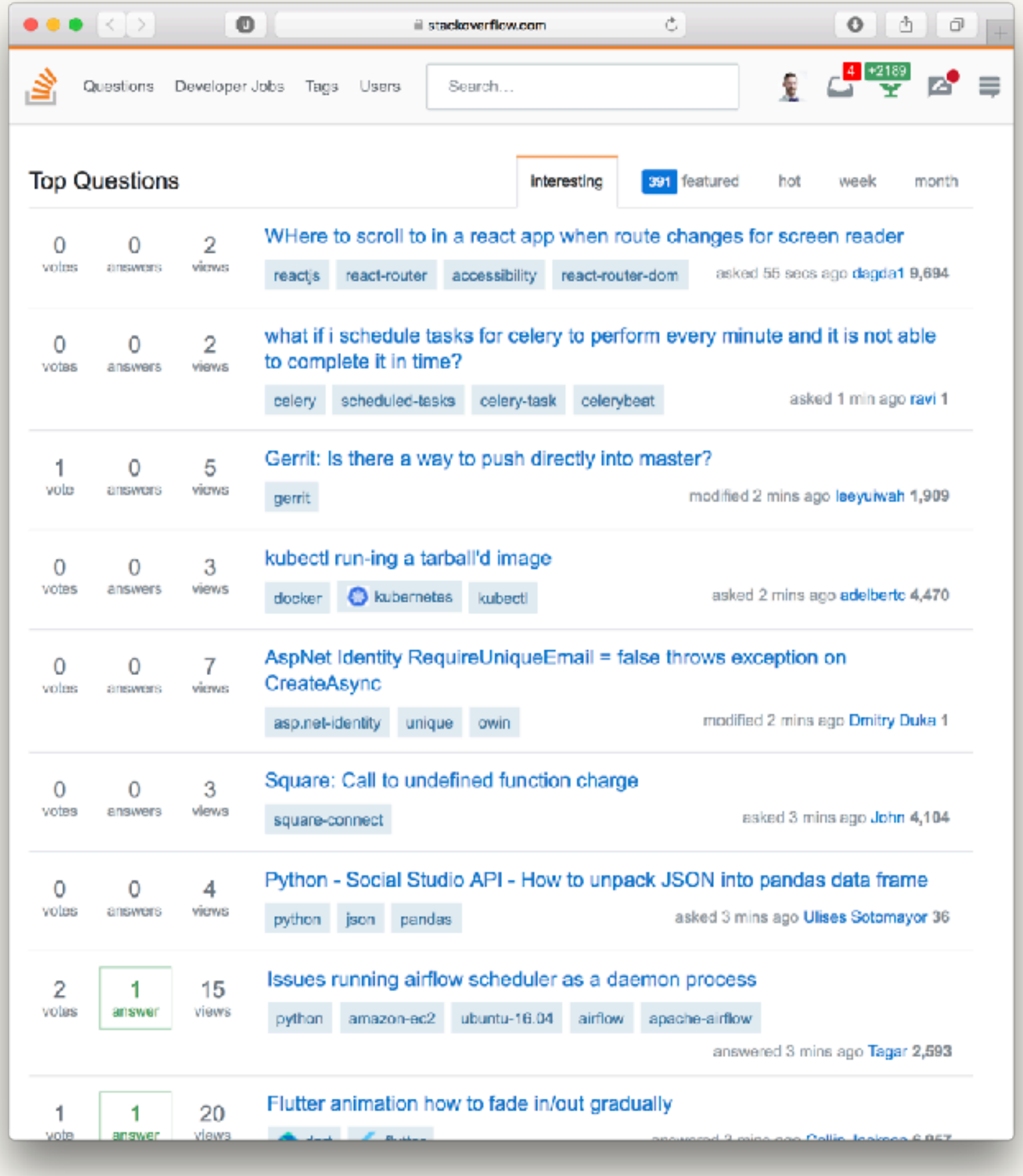
```
table %>%
  rename(player = X1, team = X2, position = X3) %>%
  filter(player != 'PLAYER') %>%
  mutate(
    college = ifelse(player == position, player, NA)
  ) %>%
  fill(college) %>%
  filter(player != college)
```

It’s just text!

And this gives you access to two extremely powerful techniques

⌘C

⌘V



Top Questions

0 votes 0 answers 2 views
WHere to scroll to in a react app when route changes for screen reader
reactjs react-router accessibility react-router-dom
asked 56 secs ago dagda1 9,694

0 votes 0 answers 2 views
what if i schedule tasks for celery to perform every minute and it is not able to complete it in time?
celery scheduled-tasks celery-task celerybeat
asked 1 min ago ravi 1

1 vote 0 answers 5 views
Gerrit: Is there a way to push directly into master?
gerrit
modified 2 mins ago leeyuiwah 1,909

0 votes 0 answers 3 views
kubectl run-ing a tarball'd image
docker kubernetes kubectl
asked 2 mins ago adelbertc 4,470

0 votes 0 answers 7 views
AspNet Identity RequireUniqueEmail = false throws exception on CreateAsync
asp.net-identity unique owin
modified 2 mins ago Dmitry Duka 1

0 votes 0 answers 3 views
Square: Call to undefined function charge
square-connect
asked 3 mins ago John 4,104

0 votes 0 answers 4 views
Python - Social Studio API - How to unpack JSON into pandas data frame
python json pandas
asked 3 mins ago Ulises Sotomayor 36

2 votes 1 answer 15 views
Issues running airflow scheduler as a daemon process
python amazon-ec2 ubuntu-16.04 airflow apache-airflow
answered 3 mins ago Tagar 2,593

1 vote 1 answer 20 views
Flutter animation how to fade in/out gradually
dart flutter
answered 3 mins ago Collin Jackson 6,857

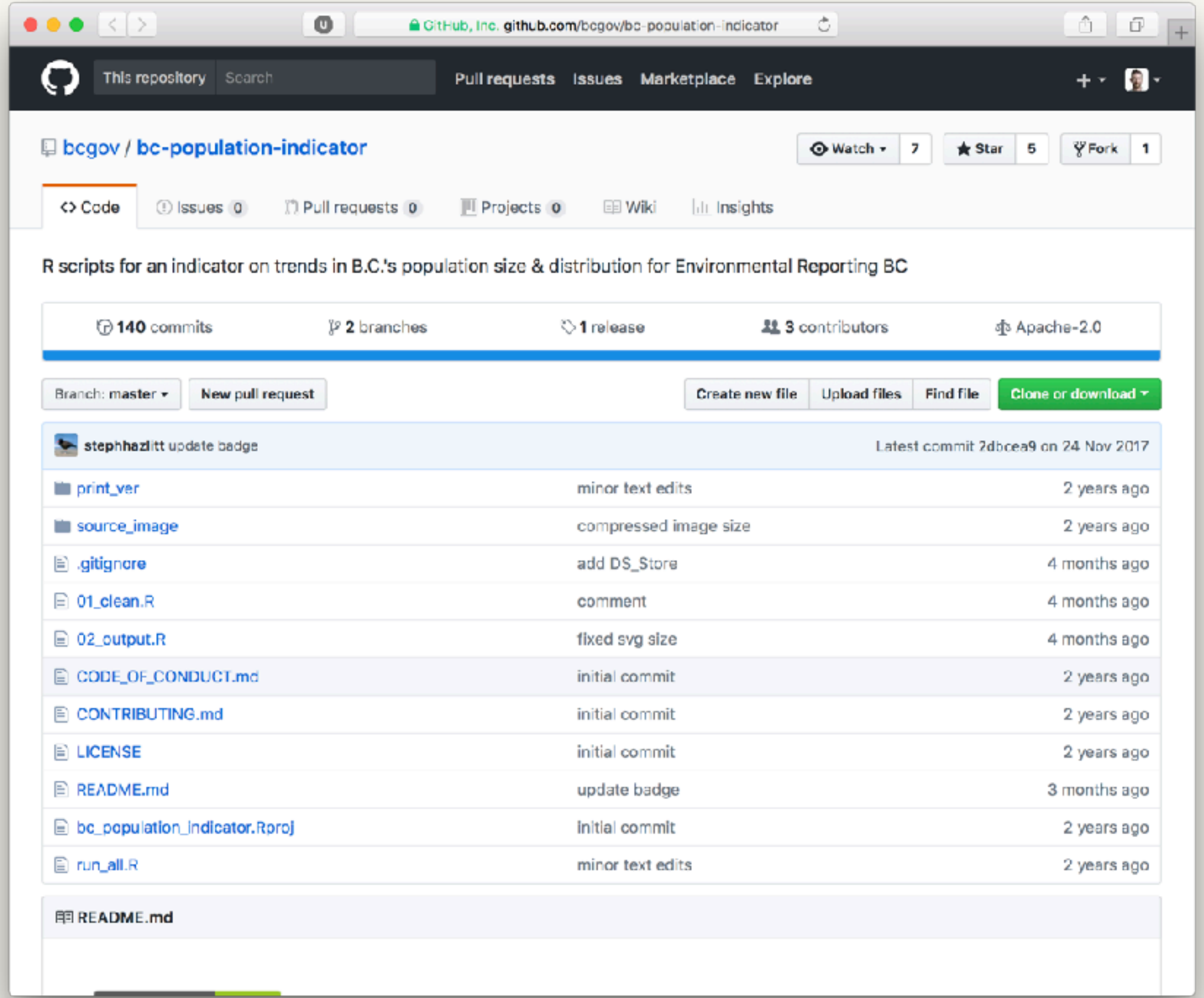
And provides provenance

Reproducible

Diffable

Readable

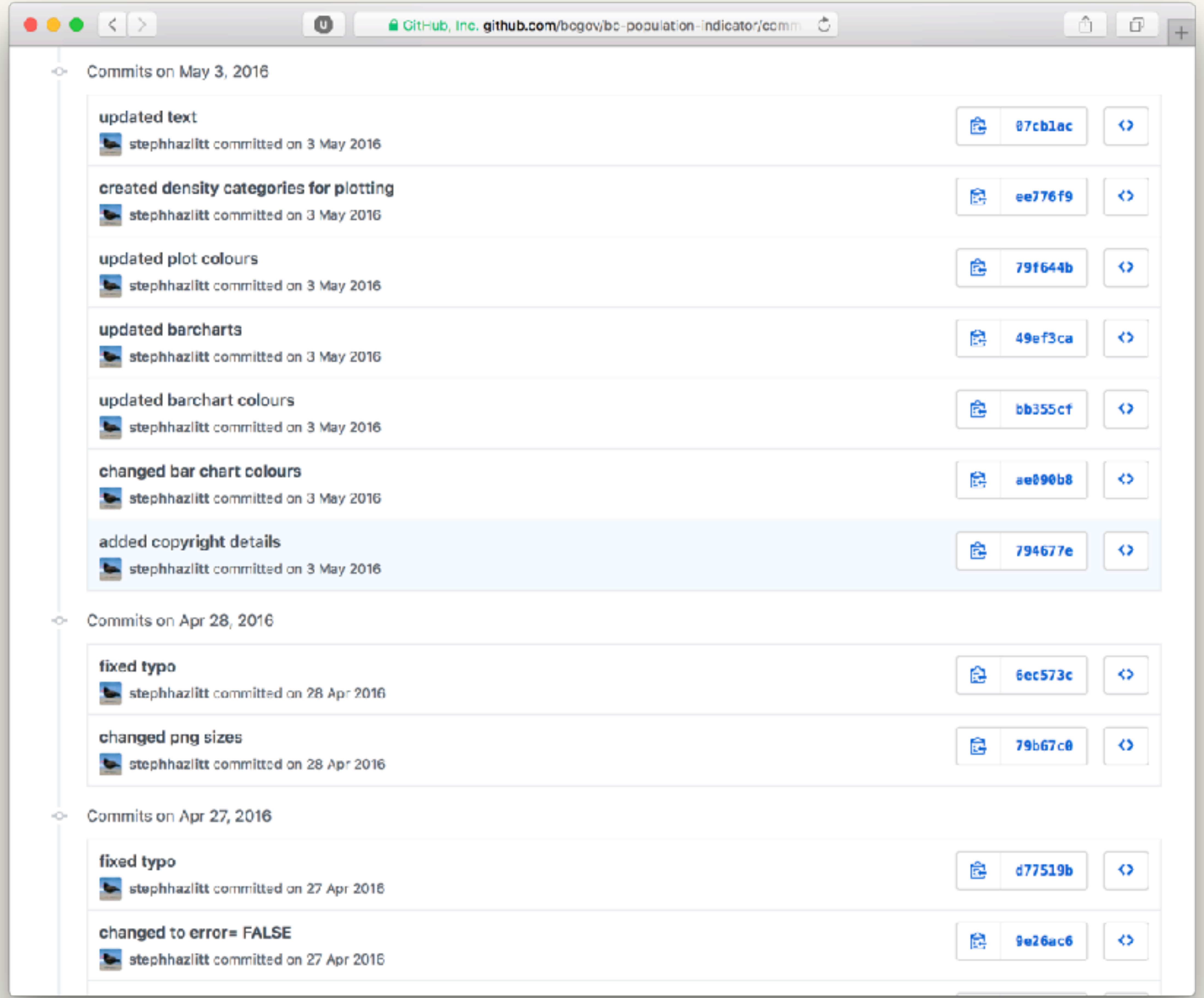
Open



R scripts for an indicator on trends in B.C.'s population size & distribution for Environmental Reporting BC

stephhazlitt update badge — Latest commit 2dbcea9 on 24 Nov 2017

print_ver	minor text edits	2 years ago
source_image	compressed image size	2 years ago
.gitignore	add DS_Store	4 months ago
01_clean.R	comment	4 months ago
02_output.R	fixed svg size	4 months ago
CODE_OF_CONDUCT.md	initial commit	2 years ago
CONTRIBUTING.md	initial commit	2 years ago
LICENSE	initial commit	2 years ago
README.md	update badge	3 months ago
bc_population_indicator.Rproj	initial commit	2 years ago
run_all.R	minor text edits	2 years ago



Commits on May 3, 2016

updated text
stephhazlitt committed on 3 May 2016 — 07cb1ac

created density categories for plotting
stephhazlitt committed on 3 May 2016 — ee776f9

updated plot colours
stephhazlitt committed on 3 May 2016 — 79f644b

updated barcharts
stephhazlitt committed on 3 May 2016 — 49ef3ca

updated barchart colours
stephhazlitt committed on 3 May 2016 — bb355cf

changed bar chart colours
stephhazlitt committed on 3 May 2016 — ae090b8

added copyright details
stephhazlitt committed on 3 May 2016 — 794677e

Commits on Apr 28, 2016

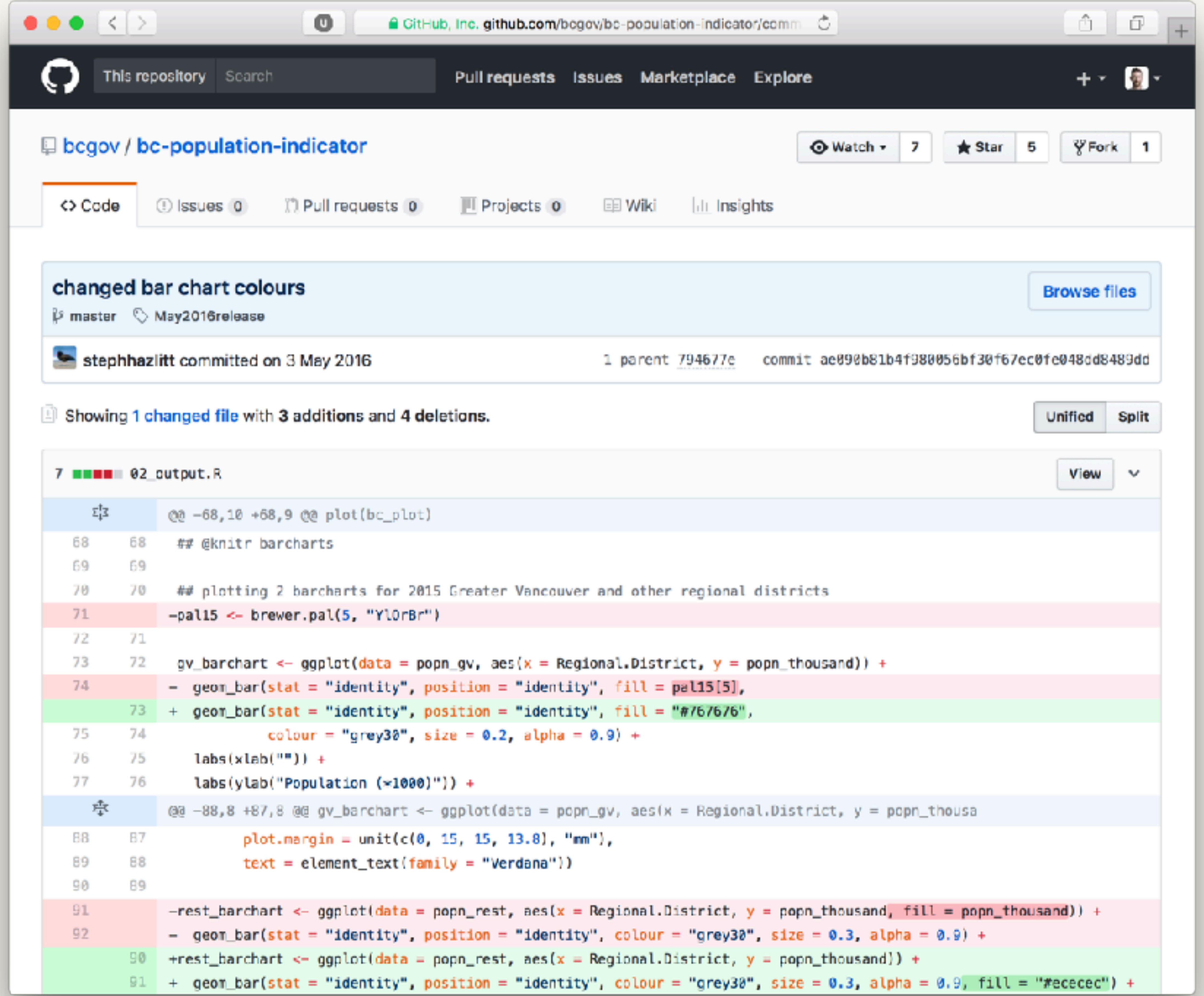
fixed typo
stephhazlitt committed on 28 Apr 2016 — 6ec573c

changed png sizes
stephhazlitt committed on 28 Apr 2016 — 79b67c0

Commits on Apr 27, 2016

fixed typo
stephhazlitt committed on 27 Apr 2016 — d77519b

changed to error= FALSE
stephhazlitt committed on 27 Apr 2016 — 9e26ac6



changed bar chart colours

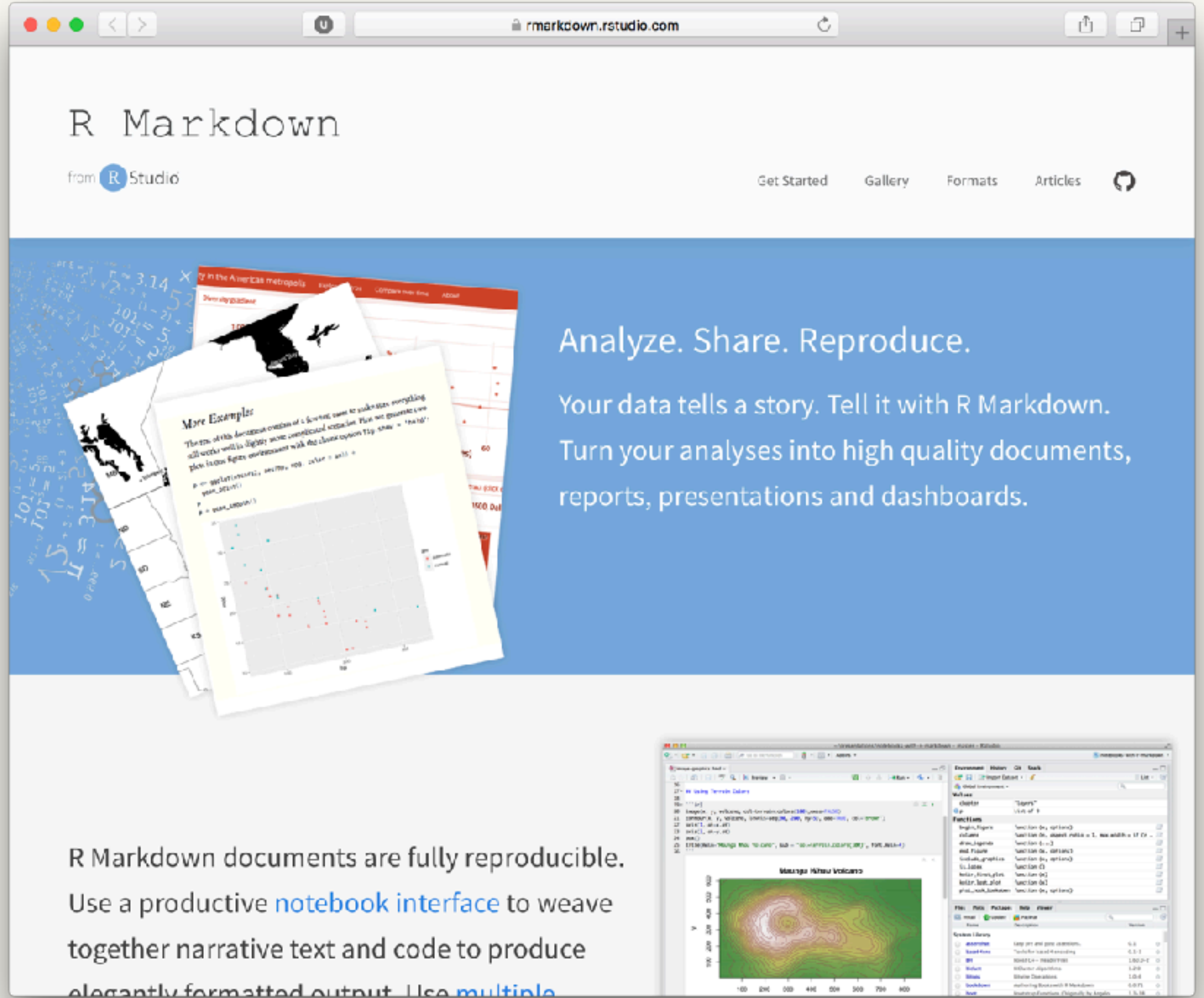
master May2016release

stephhazlitt committed on 3 May 2016 1 parent 794677e commit ae090b81b4f980056bf30f67ec0fe048dd8489dd

Showing 1 changed file with 3 additions and 4 deletions.

7 02_output.R

```
@@ -68,10 +68,9 @@ plot(bc_plot)
 ## @knitr barcharts

 ## plotting 2 barcharts for 2015 Greater Vancouver and other regional districts
-pal15 <- brewer.pal(5, "YlOrBr")

 gv_barchart <- ggplot(data = popn_gv, aes(x = Regional.District, y = popn_thousand)) +
-  geom_bar(stat = "identity", position = "identity", fill = pal15[5],
+  geom_bar(stat = "identity", position = "identity", fill = "#767676",
             colour = "grey30", size = 0.2, alpha = 0.9) +
   labs(xlab("")) +
   labs(ylab("Population (*1000)")) +
@@ -88,8 +87,8 @@ gv_barchart <- ggplot(data = popn_gv, aes(x = Regional.District, y = popn_thousa
          plot.margin = unit(c(0, 15, 15, 13.8), "mm"),
          text = element_text(family = "Verdana"))

-rest_barchart <- ggplot(data = popn_rest, aes(x = Regional.District, y = popn_thousand, fill = popn_thousand)) +
-  geom_bar(stat = "identity", position = "identity", colour = "grey30", size = 0.3, alpha = 0.9) +
+rest_barchart <- ggplot(data = popn_rest, aes(x = Regional.District, y = popn_thousand)) +
+  geom_bar(stat = "identity", position = "identity", colour = "grey30", size = 0.3, alpha = 0.9, fill = "#ececec") +
```

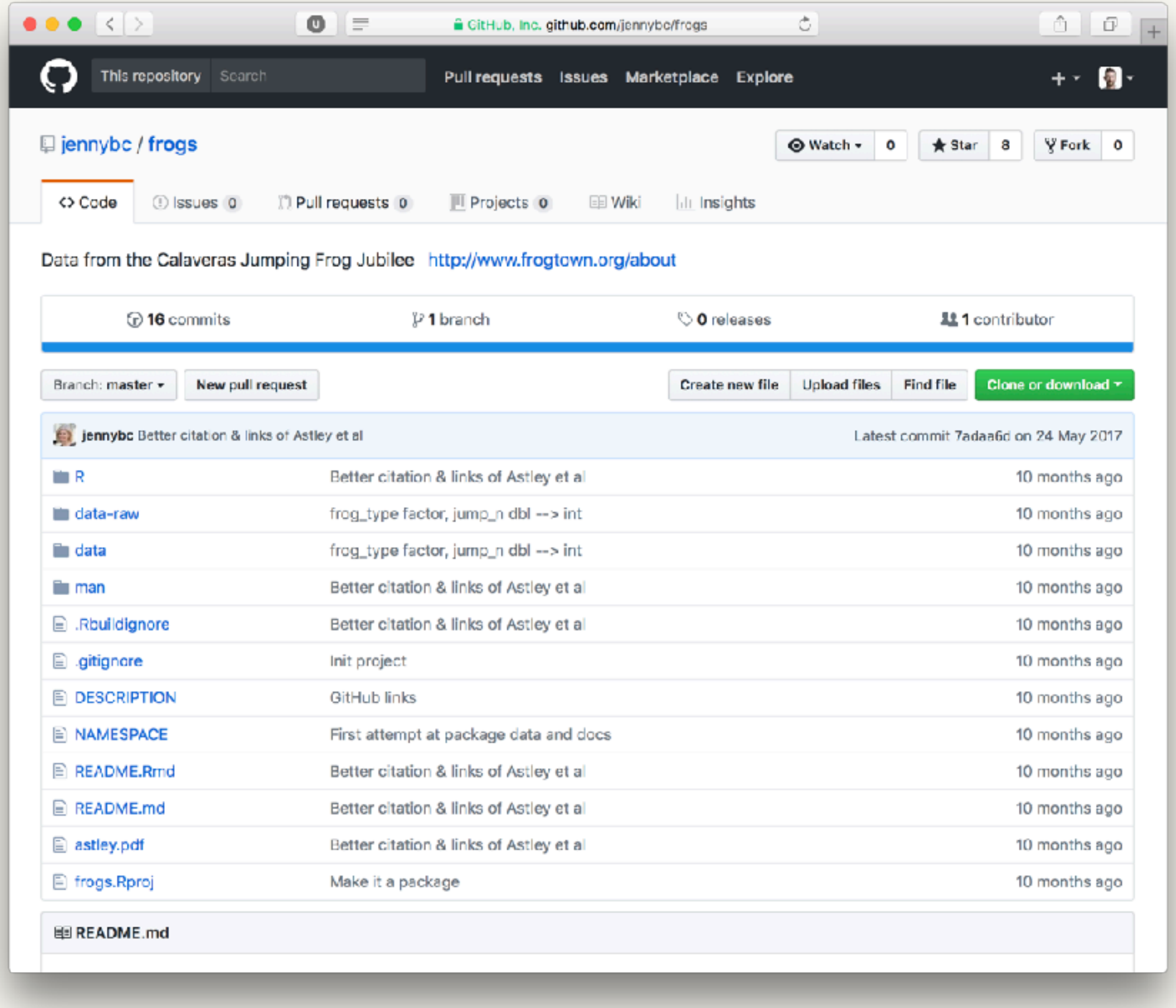


R Markdown

Analyze. Share. Reproduce.

Your data tells a story. Tell it with R Markdown. Turn your analyses into high quality documents, reports, presentations and dashboards.

R Markdown documents are fully reproducible. Use a productive notebook interface to weave together narrative text and code to produce elegantly formatted output. Use multiple

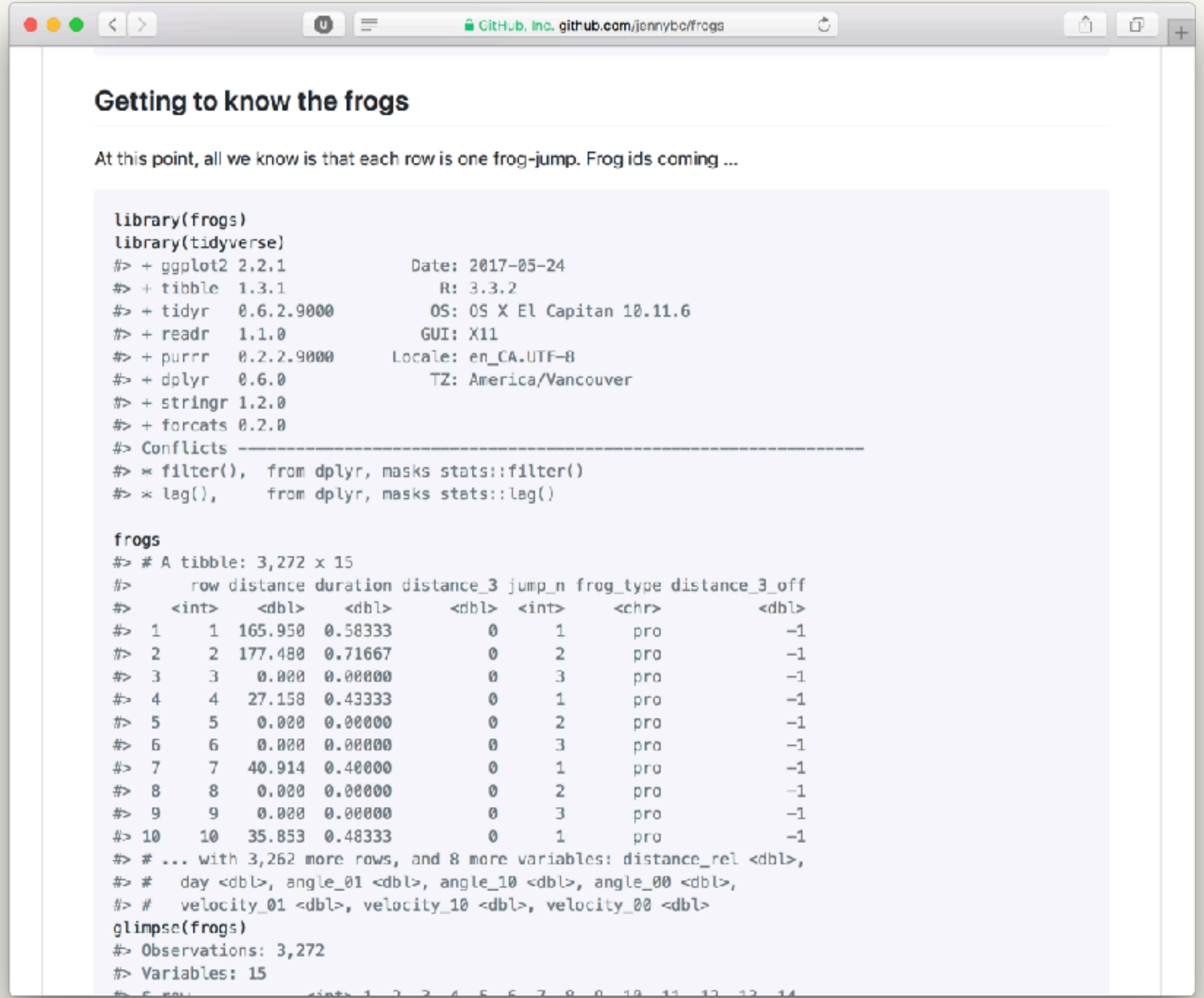


jennybc / frogs

Data from the Calaveras Jumping Frog Jubilee http://www.frogtown.org/about

jennybc Better citation & links of Astley et al — Latest commit 7adaa6d on 24 May 2017

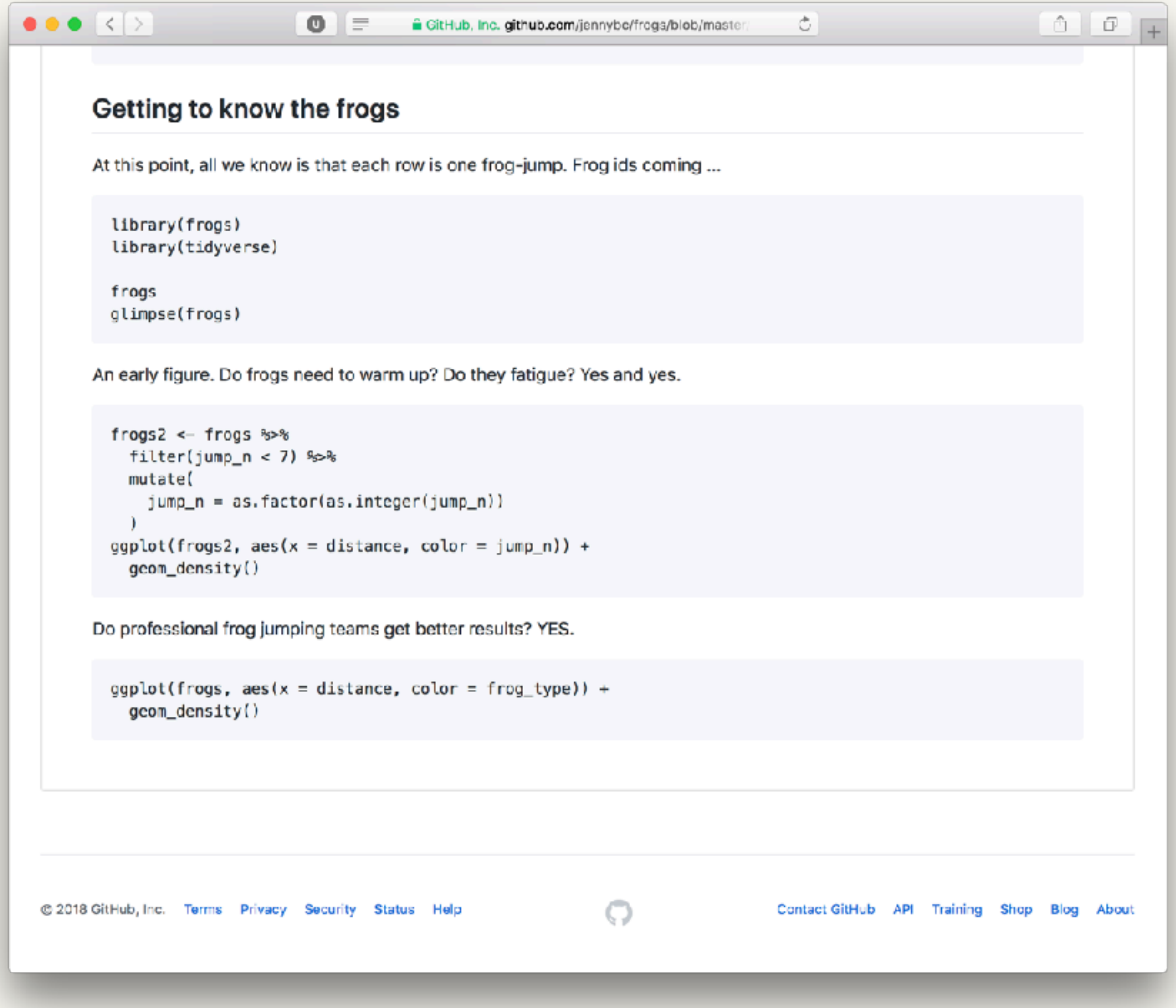
R	Better citation & links of Astley et al	10 months ago
data-raw	frog_type factor, jump_n dbl --> int	10 months ago
data	frog_type factor, jump_n dbl --> int	10 months ago
man	Better citation & links of Astley et al	10 months ago
.Rbuildignore	Better citation & links of Astley et al	10 months ago
.gitignore	Init project	10 months ago
DESCRIPTION	GitHub links	10 months ago
NAMESPACE	First attempt at package data and docs	10 months ago
README.Rmd	Better citation & links of Astley et al	10 months ago
README.md	Better citation & links of Astley et al	10 months ago
astley.pdf	Better citation & links of Astley et al	10 months ago
frogs.Rproj	Make it a package	10 months ago



Getting to know the frogs

At this point, all we know is that each row is one frog-jump. Frog ids coming ...

```
library(frogs)
library(tidyverse)
#> + ggplot2 2.2.1             Date: 2017-05-24
#> + tibble  1.3.1                R: 3.3.2
#> + tidyr   0.6.2.9000          OS: OS X El Capitan 10.11.6
#> + readr   1.1.0              GUI: X11
#> + purrr   0.2.2.9000      Locale: en_CA.UTF-8
#> + dplyr   0.6.0               TZ: America/Vancouver
#> + stringr 1.2.0
#> + forcats 0.2.0
#> Conflicts ------------------------------------------------------------------
#> ✖ filter(),  from dplyr, masks stats::filter()
#> ✖ lag(),     from dplyr, masks stats::lag()

frogs
#> # A tibble: 3,272 x 15
#>      row distance duration distance_3 jump_n frog_type distance_3_off
#>    <int>    <dbl>    <dbl>      <dbl>  <int>     <chr>          <dbl>
#>  1     1  165.950  0.58333          0      1       pro             -1
#>  2     2  177.480  0.71667          0      2       pro             -1
#>  3     3    0.000  0.00000          0      3       pro             -1
#>  4     4   27.158  0.43333          0      1       pro             -1
#>  5     5    0.000  0.00000          0      2       pro             -1
#>  6     6    0.000  0.00000          0      3       pro             -1
#>  7     7   40.914  0.40000          0      1       pro             -1
#>  8     8    0.000  0.00000          0      2       pro             -1
#>  9     9    0.000  0.00000          0      3       pro             -1
#> 10    10   35.853  0.48333          0      1       pro             -1
#> # ... with 3,262 more rows, and 8 more variables: distance_rel <dbl>,
#> #   day <dbl>, angle_01 <dbl>, angle_10 <dbl>, angle_00 <dbl>,
#> #   velocity_01 <dbl>, velocity_10 <dbl>, velocity_00 <dbl>
glimpse(frogs)
#> Observations: 3,272
#> Variables: 15
#> $ row            <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14
```



Getting to know the frogs

At this point, all we know is that each row is one frog-jump. Frog ids coming ...

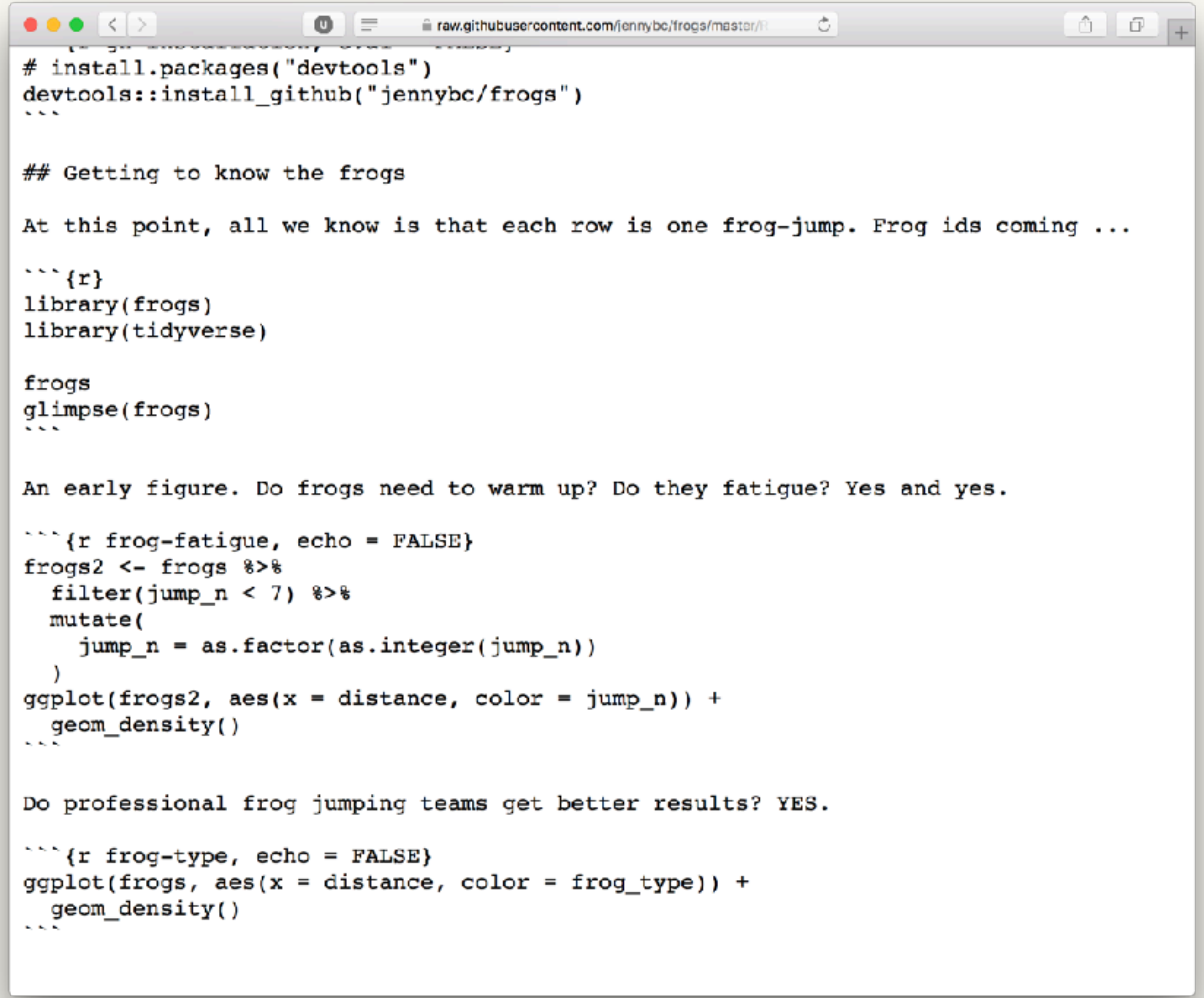
```
library(frogs)
library(tidyverse)

frogs
glimpse(frogs)
```

An early figure. Do frogs need to warm up? Do they fatigue? Yes and yes.

```
frogs2 <- frogs %>%
  filter(jump_n < 7) %>%
  mutate(
    jump_n = as.factor(as.integer(jump_n))
  )
ggplot(frogs2, aes(x = distance, color = jump_n)) +
  geom_density()
```

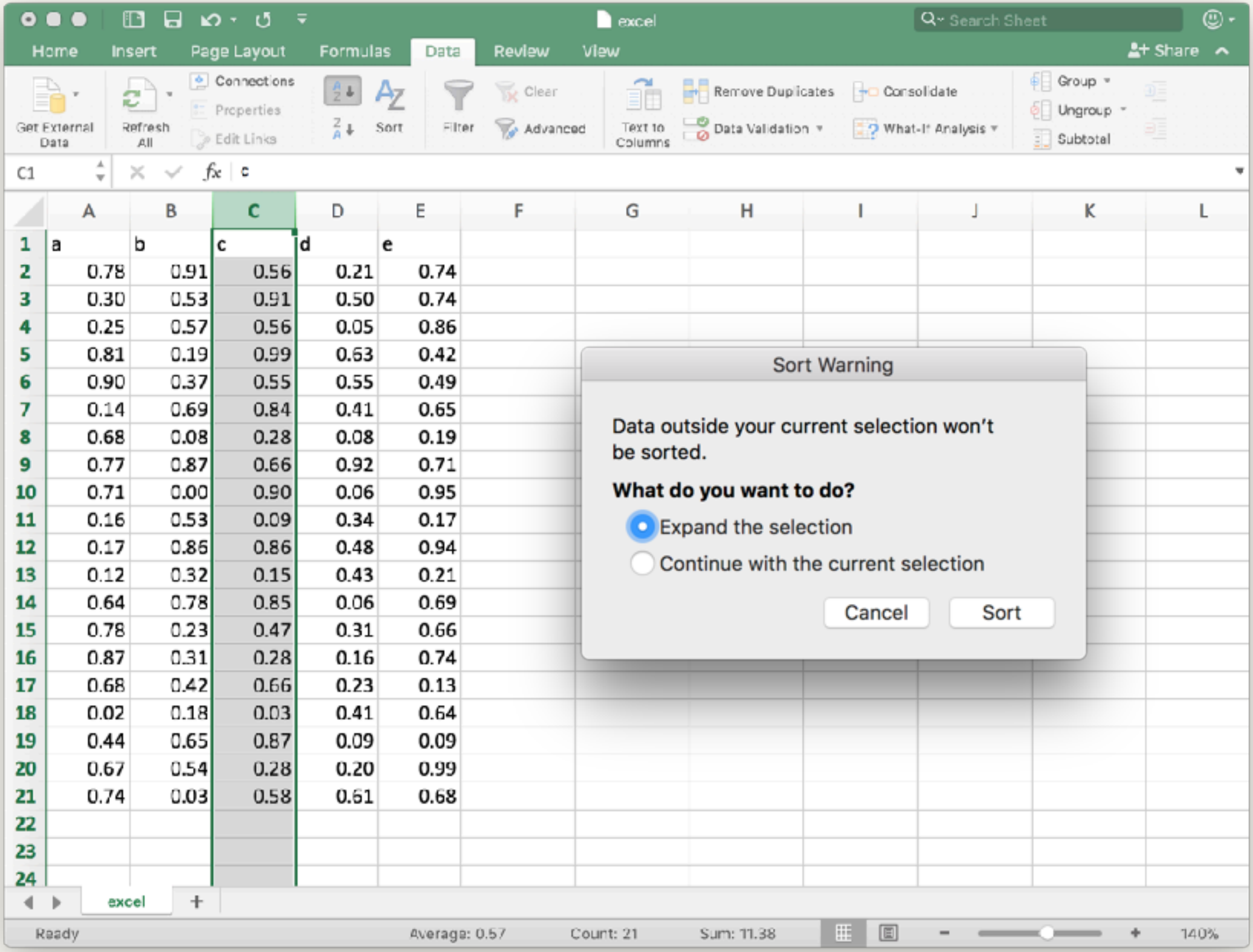
Do professional frog jumping teams get better results? YES.

```
ggplot(frogs, aes(x = distance, color = frog_type)) +
  geom_density()
```



~~~
# install.packages("devtools")
devtools::install_github("jennybc/frogs")
```


## Getting to know the frogs

At this point, all we know is that each row is one frog-jump. Frog ids coming ...

```{r}
library(frogs)
library(tidyverse)

frogs
glimpse(frogs)
```


An early figure. Do frogs need to warm up? Do they fatigue? Yes and yes.

```{r frog-fatigue, echo = FALSE}
frogs2 <- frogs %>%
  filter(jump_n < 7) %>%
  mutate(
    jump_n = as.factor(as.integer(jump_n))
  )
ggplot(frogs2, aes(x = distance, color = jump_n)) +
  geom_density()
```


Do professional frog jumping teams get better results? YES.

```{r frog-type, echo = FALSE}
ggplot(frogs, aes(x = distance, color = frog_type)) +
  geom_density()
```
~~~

I live in fear of clicking the wrong thing



C1 | *fx* c

	A	B	C	D	E
1	a	b	c	d	e
2	0.78	0.91	0.56	0.21	0.74
3	0.30	0.53	0.91	0.50	0.74
4	0.25	0.57	0.56	0.05	0.86
5	0.81	0.19	0.99	0.63	0.42
6	0.90	0.37	0.55	0.55	0.49
7	0.14	0.69	0.84	0.41	0.65
8	0.68	0.08	0.28	0.08	0.19
9	0.77	0.87	0.66	0.92	0.71
10	0.71	0.00	0.90	0.06	0.95
11	0.16	0.53	0.09	0.34	0.17
12	0.17	0.86	0.86	0.48	0.94
13	0.12	0.32	0.15	0.43	0.21
14	0.64	0.78	0.85	0.06	0.69
15	0.78	0.23	0.47	0.31	0.66
16	0.87	0.31	0.28	0.16	0.74
17	0.68	0.42	0.66	0.23	0.13
18	0.02	0.18	0.03	0.41	0.64
19	0.44	0.65	0.87	0.09	0.09
20	0.67	0.54	0.28	0.20	0.99
21	0.74	0.03	0.58	0.61	0.68

Sort Warning

Data outside your current selection won't be sorted.

What do you want to do?

● Expand the selection

○ Continue with the current selection

Cancel | Sort

Ready | Average: 0.57 | Count: 21 | Sum: 11.38 | 140%

Why program in R?

R is a vector language

```
x <- sample(100, 10)

x > 50
#>  [1]  TRUE FALSE FALSE  TRUE  TRUE
#>  [6]  TRUE  TRUE FALSE FALSE  TRUE

sum(x > 50)
#> [1] 6

# (There are no scalars! 😱)
```

Missing values are baked in

```
y <- sample(c(1:5, NA))
y
#> [1]  1 NA  2  3  5  4

y > 2
#> [1] FALSE    NA FALSE  TRUE  TRUE  TRUE

y == NA
#> [1] NA NA NA NA NA NA
```

An example makes this clearer

```
john_age <- NA
mary_age <- NA

john_age == mary_age
#> [1] NA
```

Missing values are baked in

```
y <- sample(c(1:5, NA))
y
#> [1]  1 NA  2  3  5  4

y > 2
#> [1] FALSE    NA FALSE  TRUE  TRUE  TRUE

is.na(y)
#> [1] FALSE  TRUE FALSE FALSE FALSE FALSE
```

So are relational tables (aka data frames/tibbles)

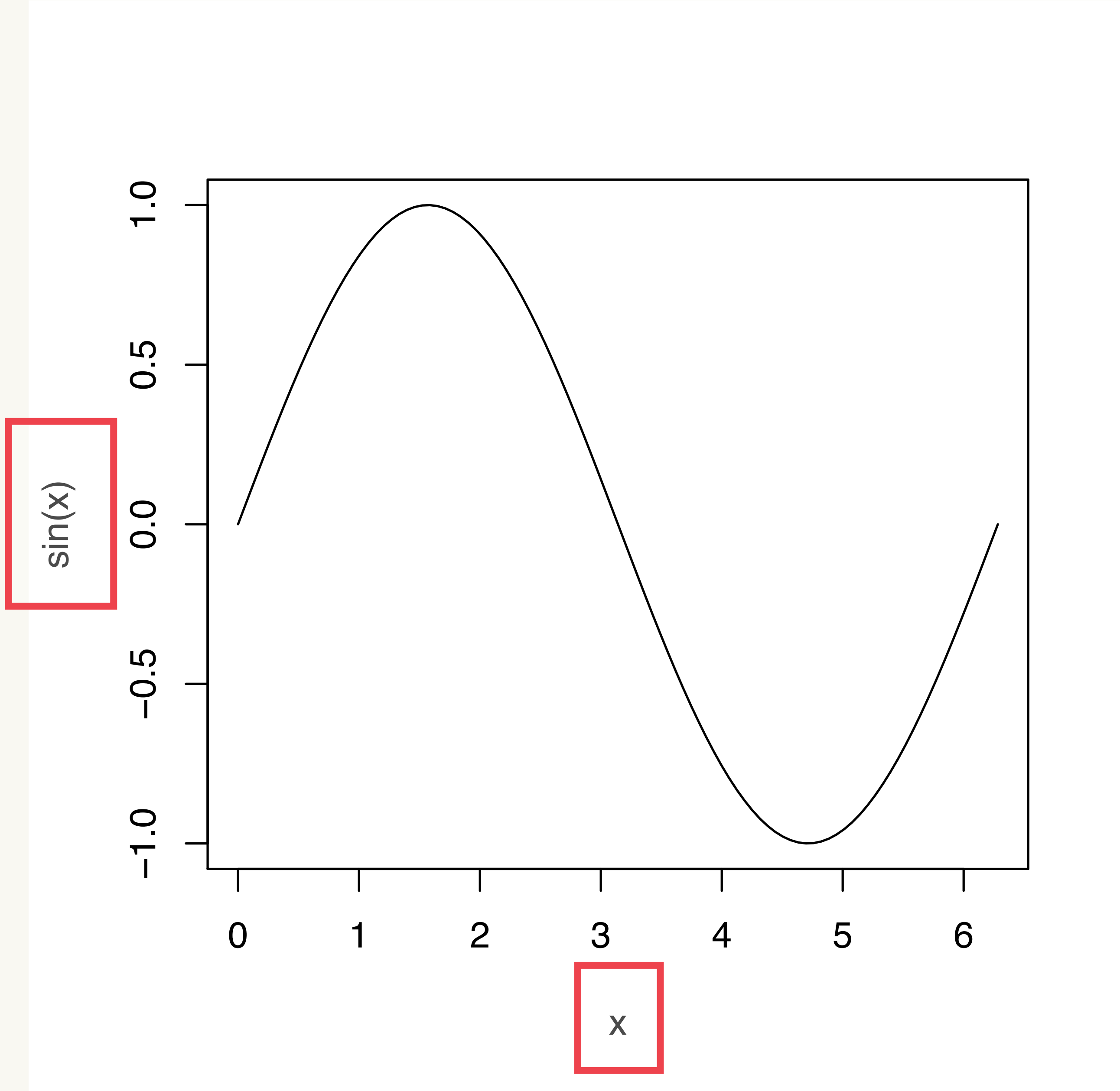
```
data.frame(
  x = 1:4,
  y = sample(letters[1:4]),
  z = runif(4)
)
#>   x y         z
#> 1 1 c 0.1189635
#> 2 2 a 0.0518956
#> 3 3 b 0.4471441
#> 4 4 d 0.0818547
```

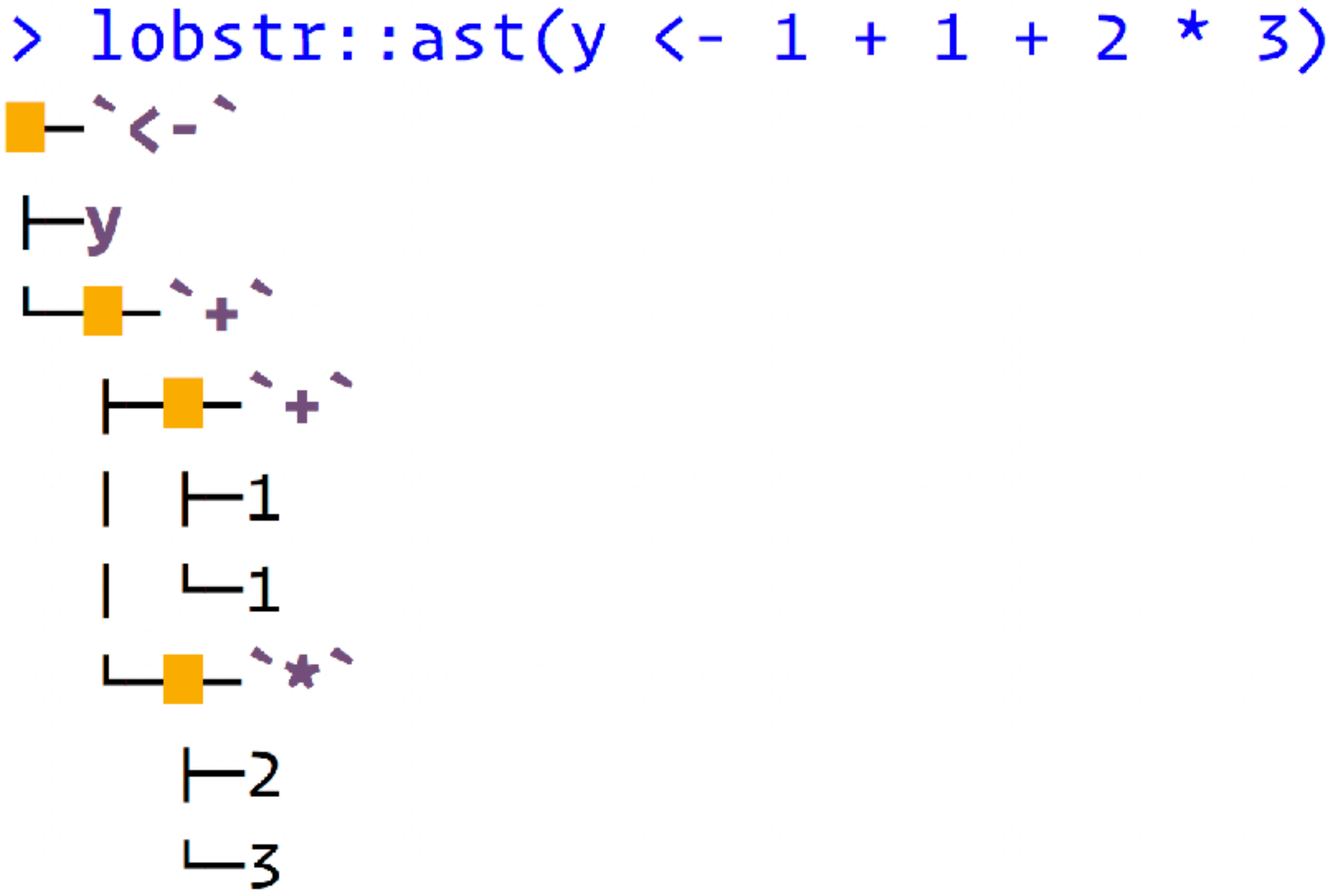
Functional programming

```
# It's well suited to data science but I
# can't (yet) articulate why

# Something about having a standard
# container for 80% of problems, and
# needing to do something to each element
# of that container

# Whole object thinking?
```

Metaprogramming

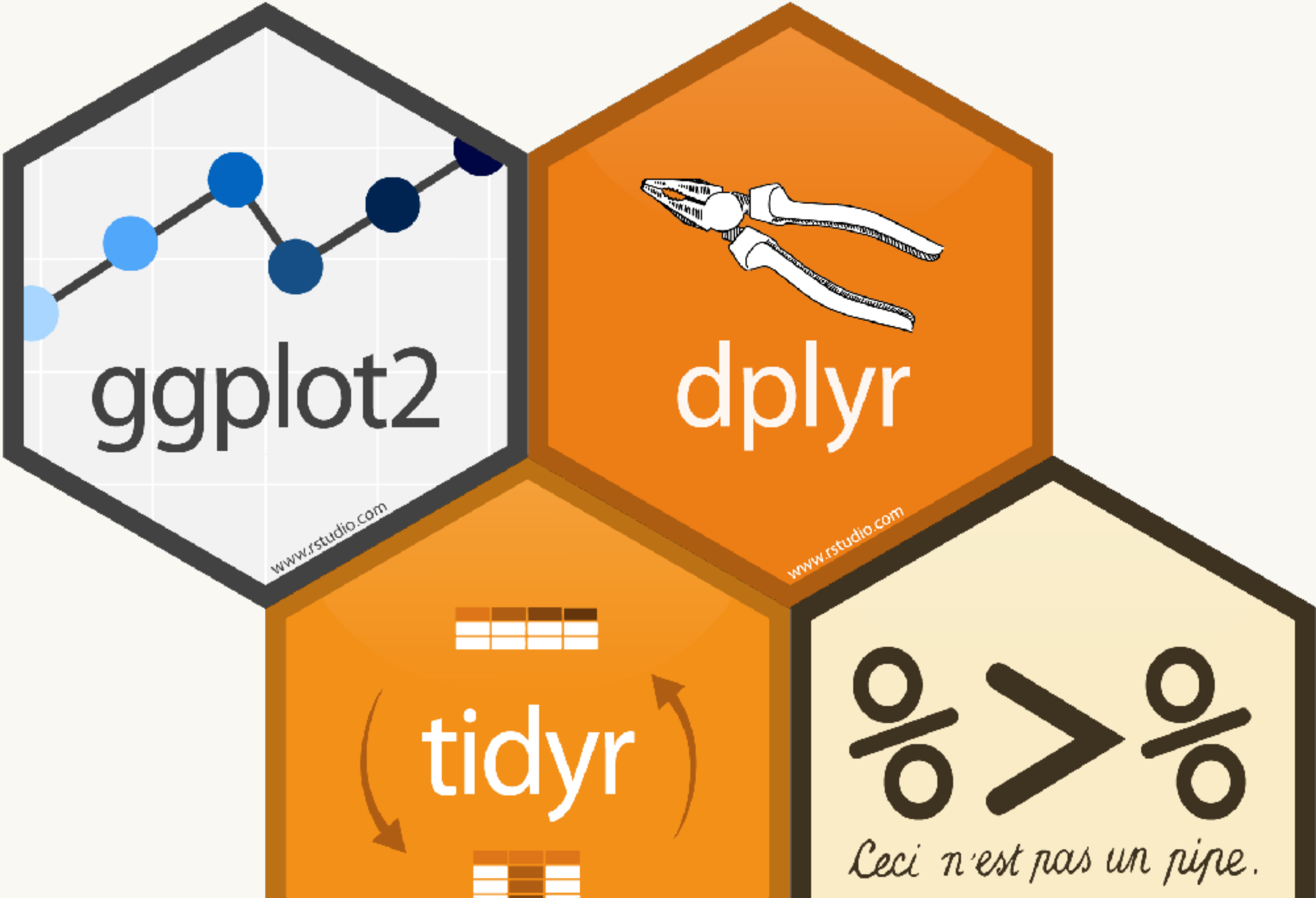
```
x <- seq(0, 2 * pi, length = 100)
plot(x, sin(x), type = "l")
```





```
> lobstr::ast(y <- 1 + 1 + 2 * 3)
■─`<-`
├─y
└─■─`+`
  ├─■─`+`
  │ ├─1
  │ └─1
  └─■─`*`
    ├─2
    └─3
```

Which makes it a great place to write DSLs



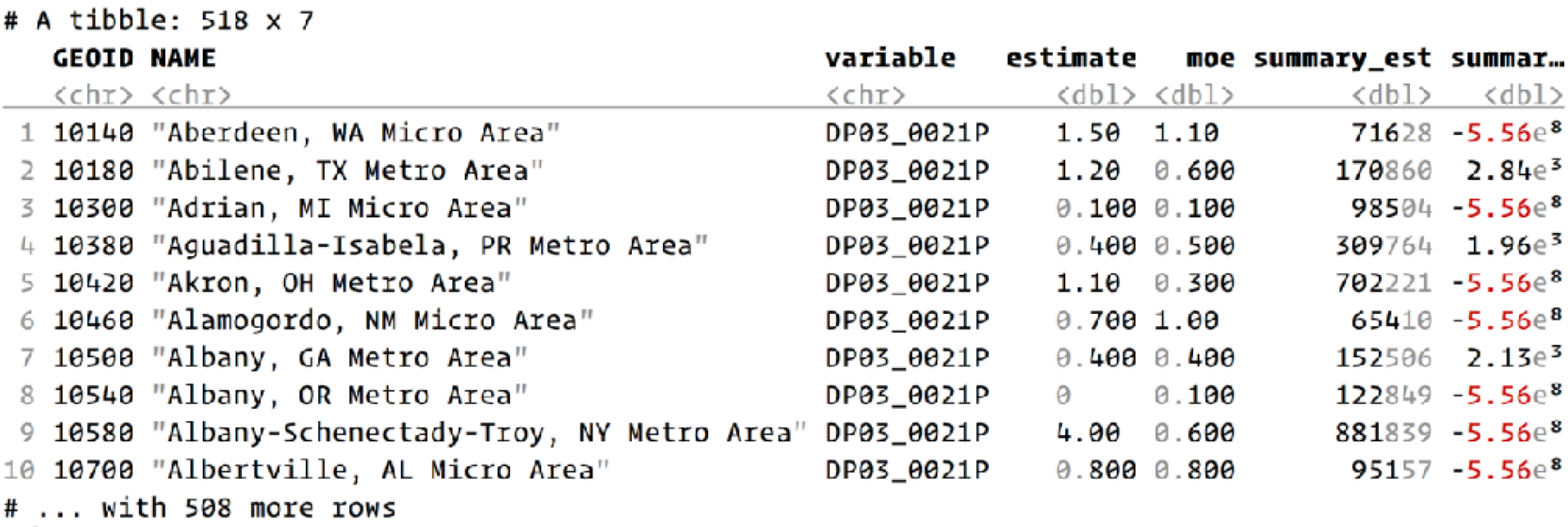
Why program in R with the tidyverse?

Solve complex problems by combining simple pieces

https://unsplash.com/photos/tjX_sniNzgQ

A small example

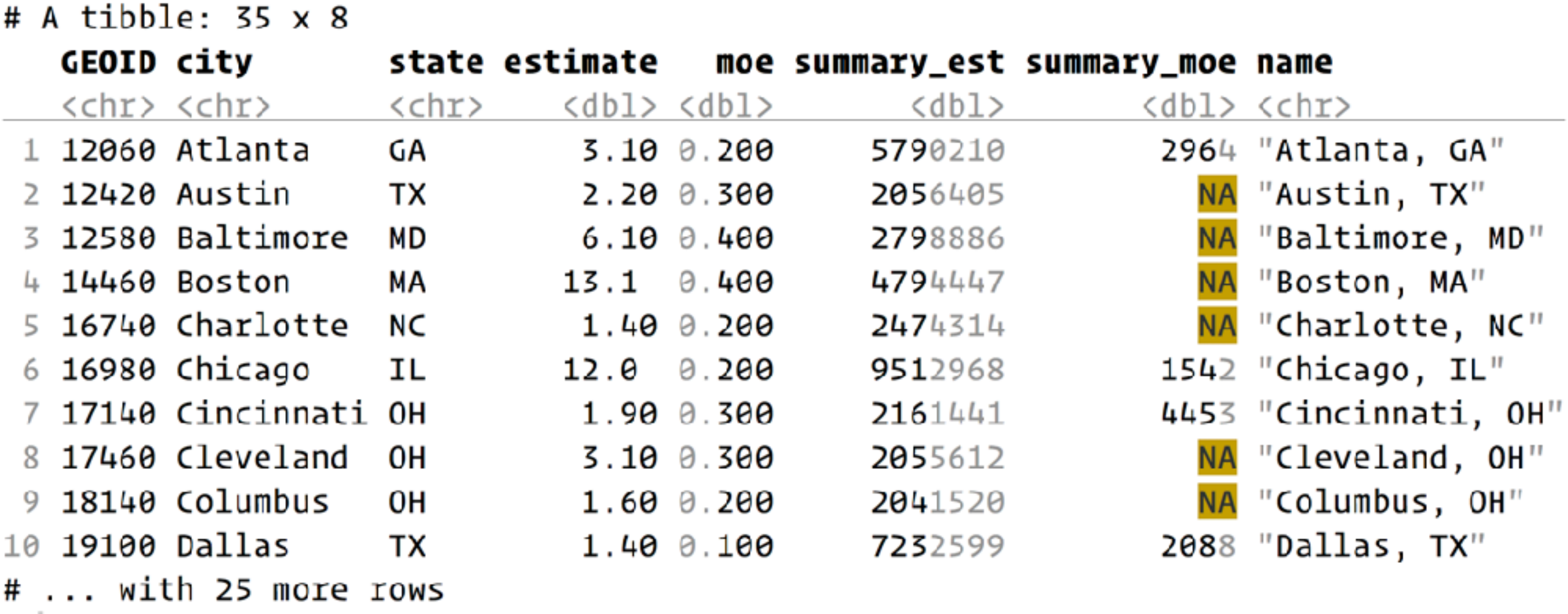
```
library(tidycensus)
geo <- get_acs(
  geography = "metropolitan statistical area...",
  variables = "DP03_0021PE",
  summary_var = "B01003_001",
  survey = "acs1",
  endyear = 2016
)

# Thanks to Kyle Walker (@kyle_e_walker)
# For package and example
```



```
# A tibble: 518 x 7
   GEOID NAME                                        variable    estimate   moe summary_est summar…
   <chr> <chr>                                       <chr>          <dbl> <dbl>       <dbl>   <dbl>
 1 10140 "Aberdeen, WA Micro Area"                   DP03_0021P     1.50  1.10        71628 -5.56e8
 2 10180 "Abilene, TX Metro Area"                    DP03_0021P     1.20  0.600      170860  2.84e3
 3 10300 "Adrian, MI Micro Area"                     DP03_0021P     0.100 0.100       98504 -5.56e8
 4 10380 "Aguadilla-Isabela, PR Metro Area"          DP03_0021P     0.400 0.500      309764  1.96e3
 5 10420 "Akron, OH Metro Area"                      DP03_0021P     1.10  0.300      702221 -5.56e8
 6 10460 "Alamogordo, NM Micro Area"                 DP03_0021P     0.700 1.00        65410 -5.56e8
 7 10500 "Albany, GA Metro Area"                     DP03_0021P     0.400 0.400      152506  2.13e3
 8 10540 "Albany, OR Metro Area"                     DP03_0021P     0     0.100      122849 -5.56e8
 9 10580 "Albany-Schenectady-Troy, NY Metro Area"    DP03_0021P     4.00  0.600      881839 -5.56e8
10 10700 "Albertville, AL Micro Area"                DP03_0021P     0.800 0.800       95157 -5.56e8
# ... with 508 more rows
```

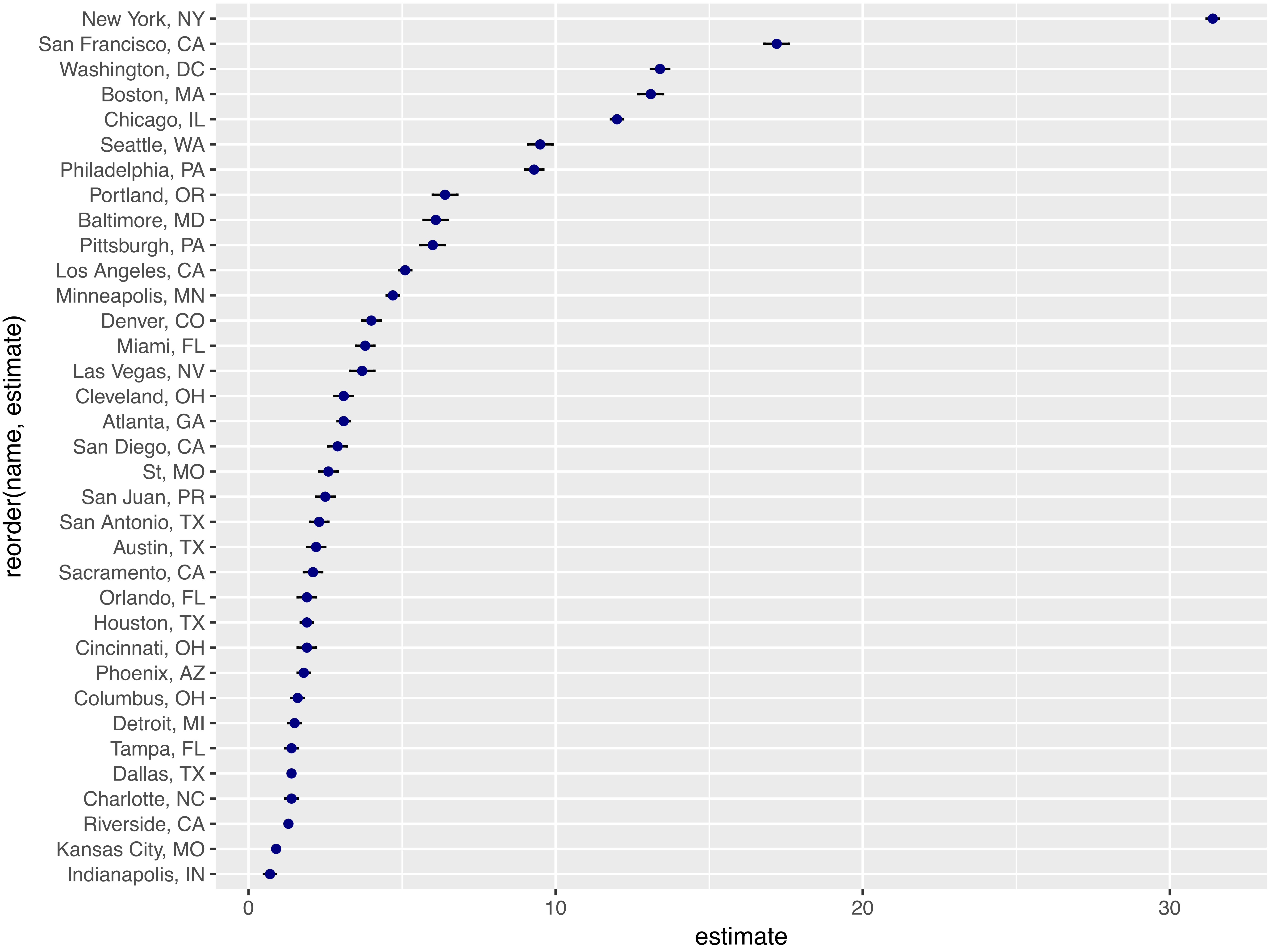
Followed by data munging

```
big_metro <- geo %>%
  filter(summary_est > 2e6) %>%
  select(-variable) %>%
  mutate(
    NAME = gsub(" Metro Area", "", NAME)
  ) %>%
  separate(NAME, c("city", "state"), ", ") %>%
  mutate(
    city = str_extract(city, "^[A-Za-z ]+"),
    state = str_extract(state, "^[A-Za-z ]+"),
    name = paste0(city, ", ", state),
    summary_moe = na_if(summary_moe, -555555555)
  )
```



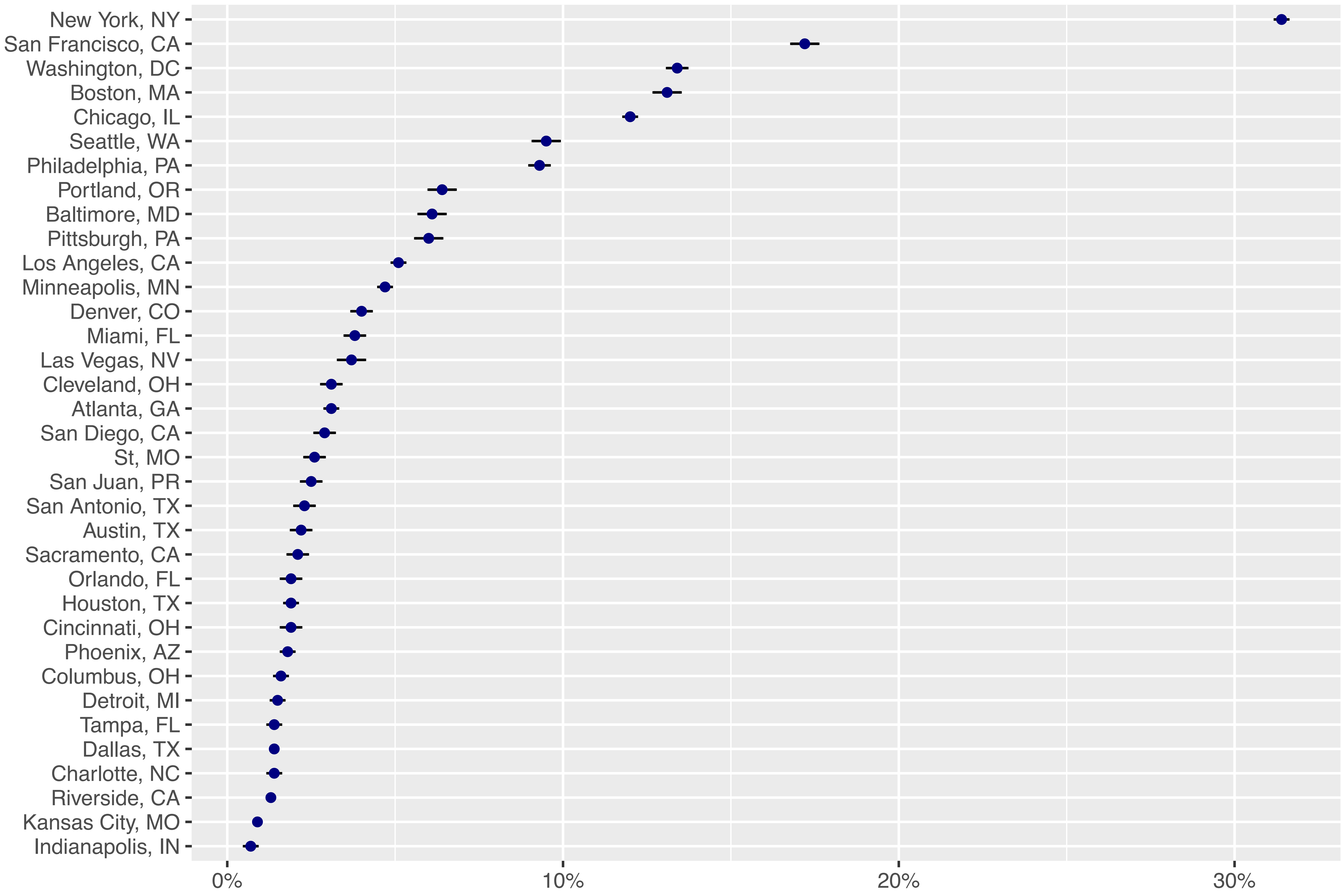
```
# A tibble: 35 x 8
   GEOID city       state estimate   moe summary_est summary_moe name
   <chr> <chr>      <chr>    <dbl> <dbl>       <dbl>       <dbl> <chr>
 1 12060 Atlanta    GA        3.10 0.200     5790210        2964 "Atlanta, GA"
 2 12420 Austin     TX        2.20 0.300     2056405          NA "Austin, TX"
 3 12580 Baltimore  MD        6.10 0.400     2798886          NA "Baltimore, MD"
 4 14460 Boston     MA       13.1  0.400     4794447          NA "Boston, MA"
 5 16740 Charlotte  NC        1.40 0.200     2474314          NA "Charlotte, NC"
 6 16980 Chicago    IL       12.0  0.200     9512968        1542 "Chicago, IL"
 7 17140 Cincinnati OH        1.90 0.300     2161441        4453 "Cincinnati, OH"
 8 17460 Cleveland  OH        3.10 0.300     2055612          NA "Cleveland, OH"
 9 18140 Columbus   OH        1.60 0.200     2041520          NA "Columbus, OH"
10 19100 Dallas     TX        1.40 0.100     7232599        2088 "Dallas, TX"
# ... with 25 more rows
```

```r
big_metro %>%
  ggplot(aes(
    x = estimate,
    y = reorder(name, estimate))
  ) +
  geom_errorbarh(
    aes(
      xmin = estimate - moe,
      xmax = estimate + moe
    ),
    width = 0.1
  ) +
  geom_point(color = "navy")
```


New York, NY
San Francisco, CA
Washington, DC
Boston, MA
Chicago, IL
Seattle, WA
Philadelphia, PA
Portland, OR
Baltimore, MD
Pittsburgh, PA
Los Angeles, CA
Minneapolis, MN
Denver, CO
Miami, FL
Las Vegas, NV
Cleveland, OH
Atlanta, GA
San Diego, CA
St, MO
San Juan, PR
San Antonio, TX
Austin, TX
Sacramento, CA
Orlando, FL
Houston, TX
Cincinnati, OH
Phoenix, AZ
Columbus, OH
Detroit, MI
Tampa, FL
Dallas, TX
Charlotte, NC
Riverside, CA
Kansas City, MO
Indianapolis, IN
reorder(name, estimate)
0
10
20
30
estimate

Residents who take public transportation to work

2016 1−year ACS estimates



Source: ACS Data Profile variable DP03_0021P / tidycensus

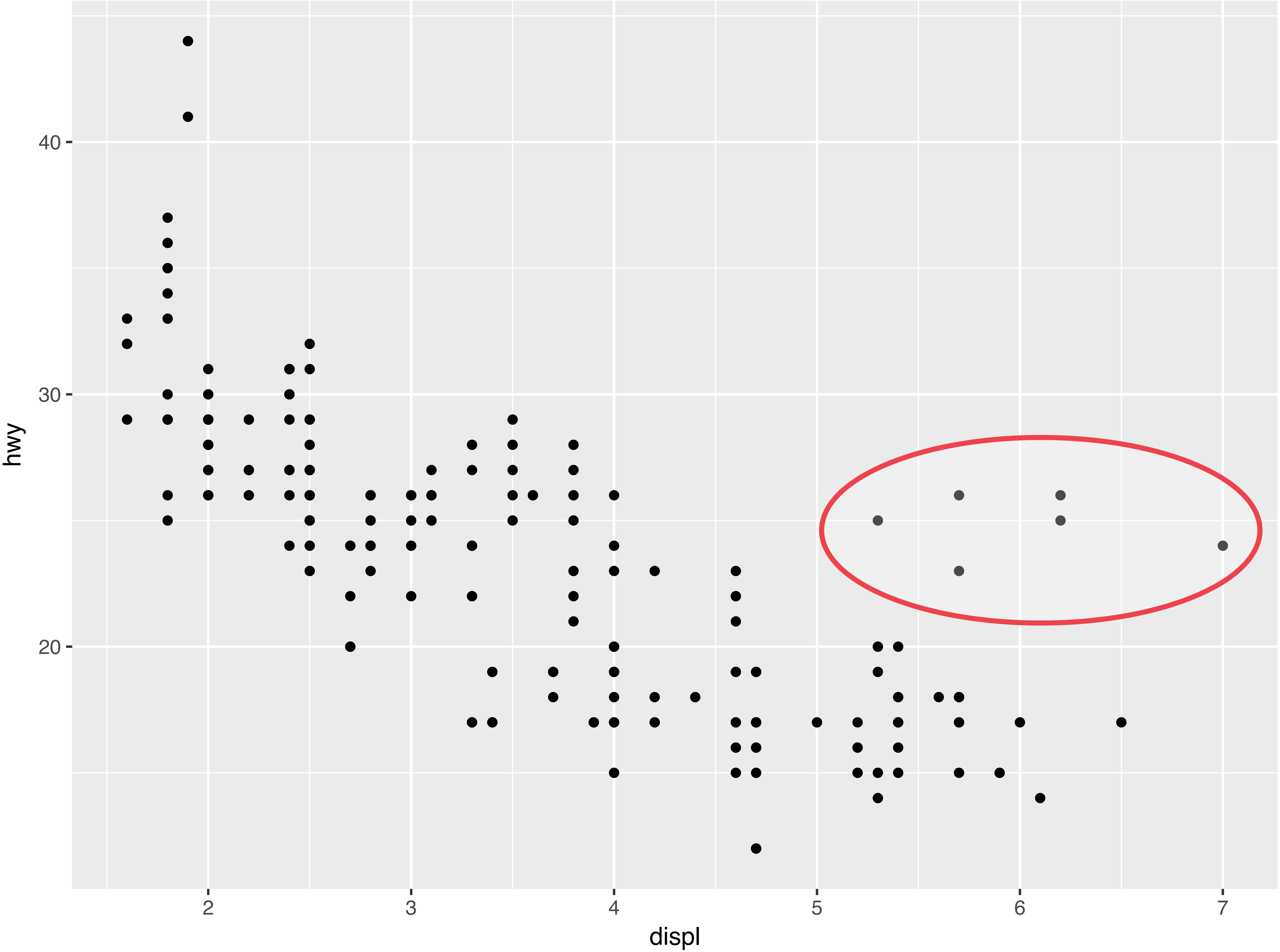
No matter how complex and polished the individual operations are, it is often the quality of the glue that most directly determines the power of the system.

— *Hal Abelson*

My goal is to make a **pit of success**

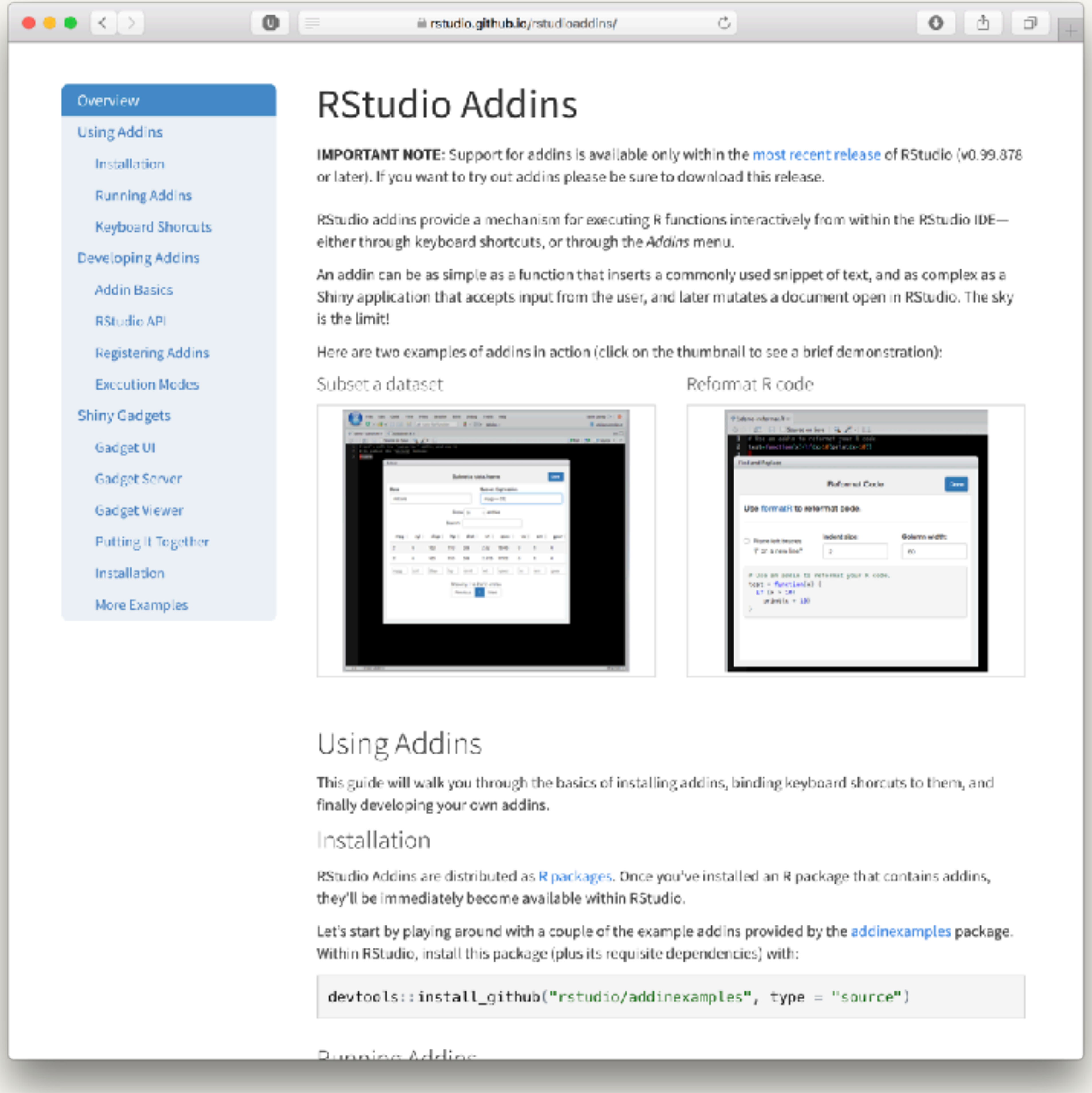
http://blog.codinghorror.com/falling-into-the-pit-of-success/

But


40
30
20
hwy
2
3
4
5
6
7
displ

But this is painful!

```
df %>%
  select(
    date = `Date Created`,
    name = Name,
    plays = `Total Plays`,
    loads = `Total Loads`,
    apv = `Average Percent Viewed`
  )
```



RStudio Addins

IMPORTANT NOTE: Support for addins is available only within the most recent release of RStudio (v0.99.878 or later). If you want to try out addins please be sure to download this release.

RStudio addins provide a mechanism for executing R functions interactively from within the RStudio IDE—either through keyboard shortcuts, or through the *Addins* menu.

An addin can be as simple as a function that inserts a commonly used snippet of text, and as complex as a Shiny application that accepts input from the user, and later mutates a document open in RStudio. The sky is the limit!

Here are two examples of addins in action (click on the thumbnail to see a brief demonstration):

Subset a dataset

Reformat R code

Using Addins

This guide will walk you through the basics of installing addins, binding keyboard shortcuts to them, and finally developing your own addins.

Installation

RStudio Addins are distributed as R packages. Once you've installed an R package that contains addins, they'll be immediately become available within RStudio.

Let's start by playing around with a couple of the example addins provided by the addinexamples package. Within RStudio, install this package (plus its requisite dependencies) with:

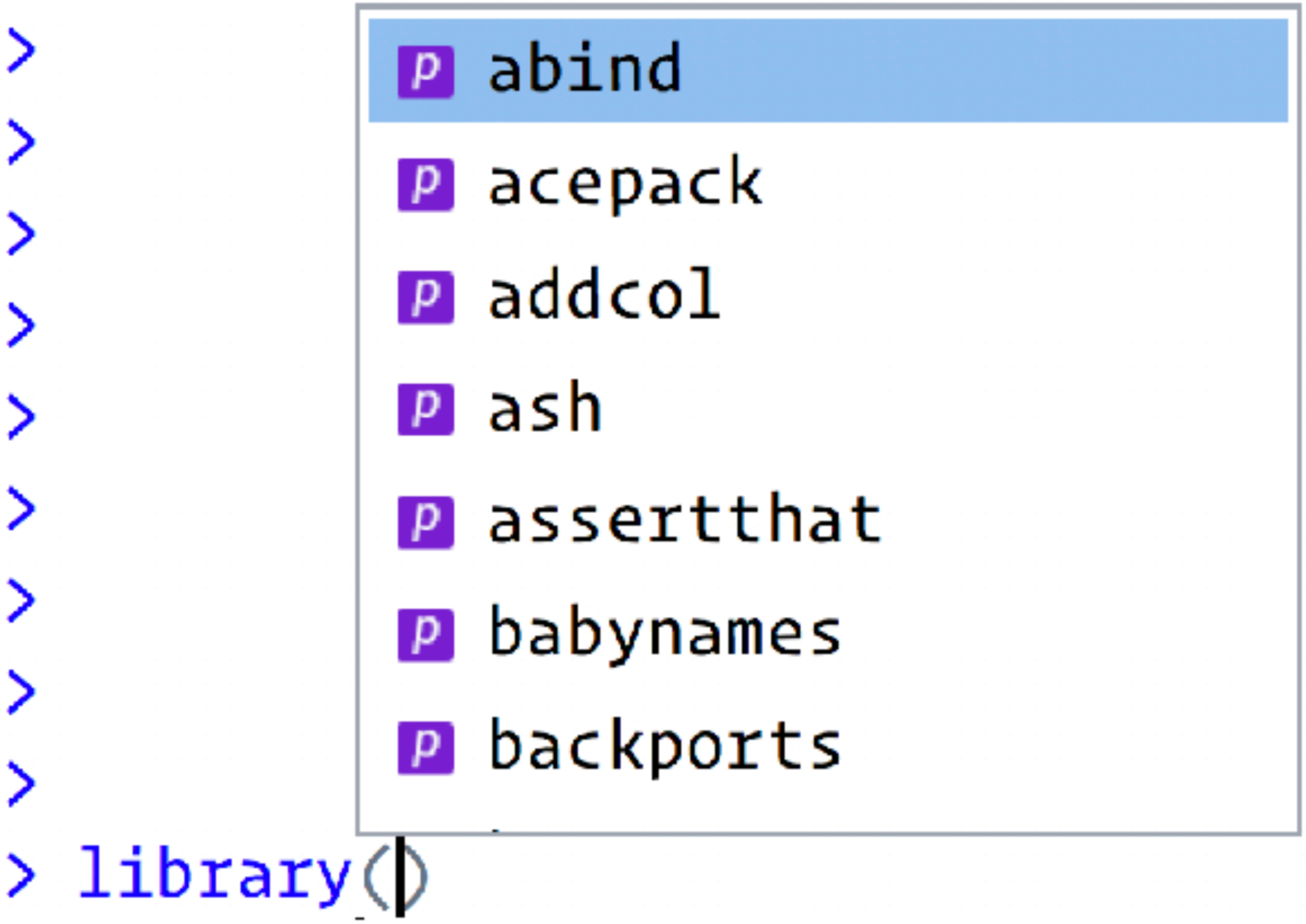
```
devtools::install_github("rstudio/addinexamples", type = "source")
```

Running Addins

What next?

```
df %>%
  filter(n > 1e6) %>%
  mutate(x = f(y))) %>%
  ???

# How predictable is next step from
# previous steps?
```

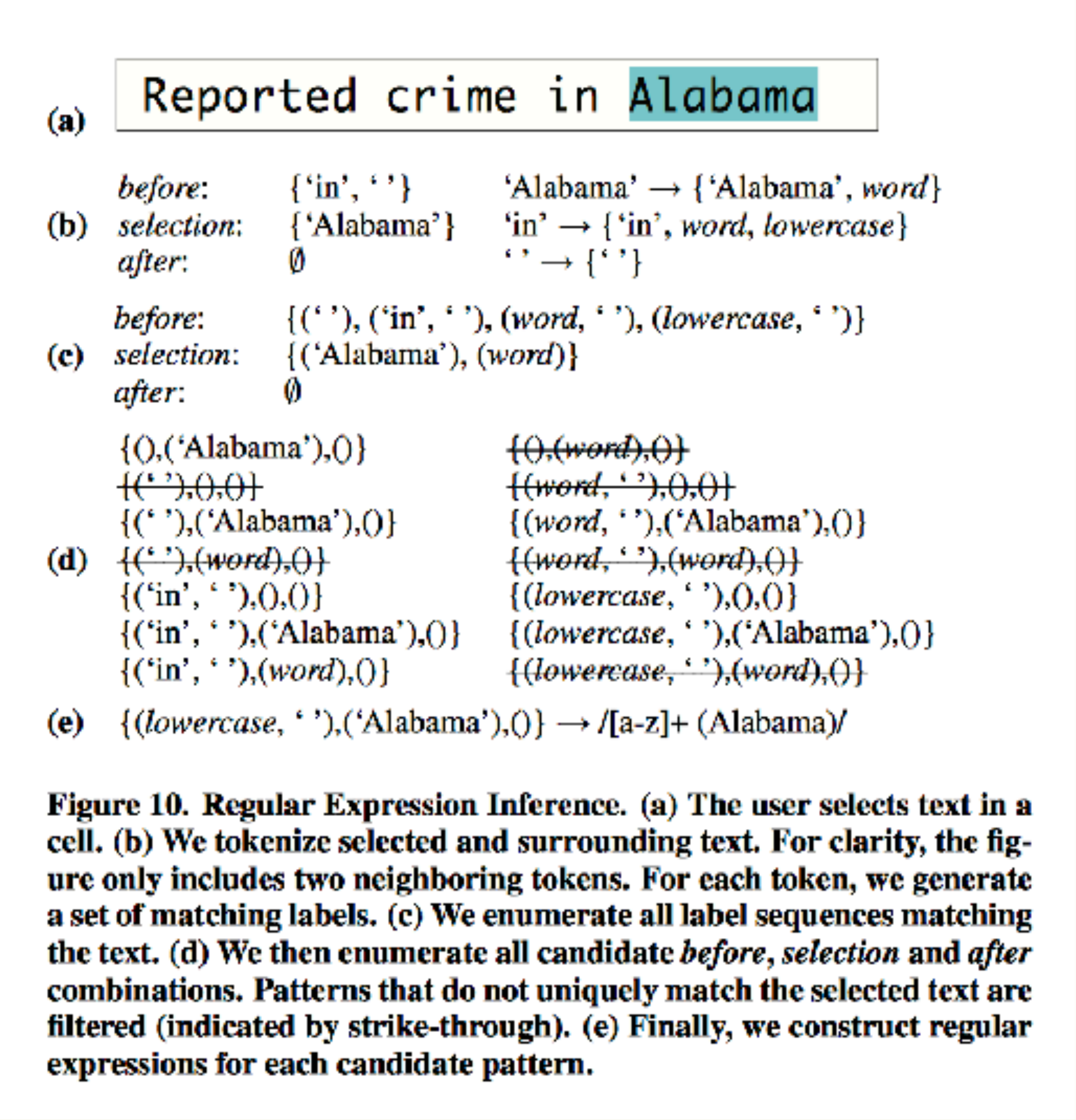
Can we do more with autocomplete?



```
>
>
>
>
>
>
>
>
>
> library()
```

Where do dialogs and autocomplete intersect?

Learning from examples



(a) Reported crime in Alabama

(b)

before:	{'in', ' '}	'Alabama' → {'Alabama', *word*}
selection:	{'Alabama'}	'in' → {'in', *word*, *lowercase*}
after:	∅	' ' → {' '}

(c)

before:	{(' '), ('in', ' '), (*word*, ' '), (*lowercase*, ' ')}
selection:	{('Alabama'), (*word*)}
after:	∅

(d)

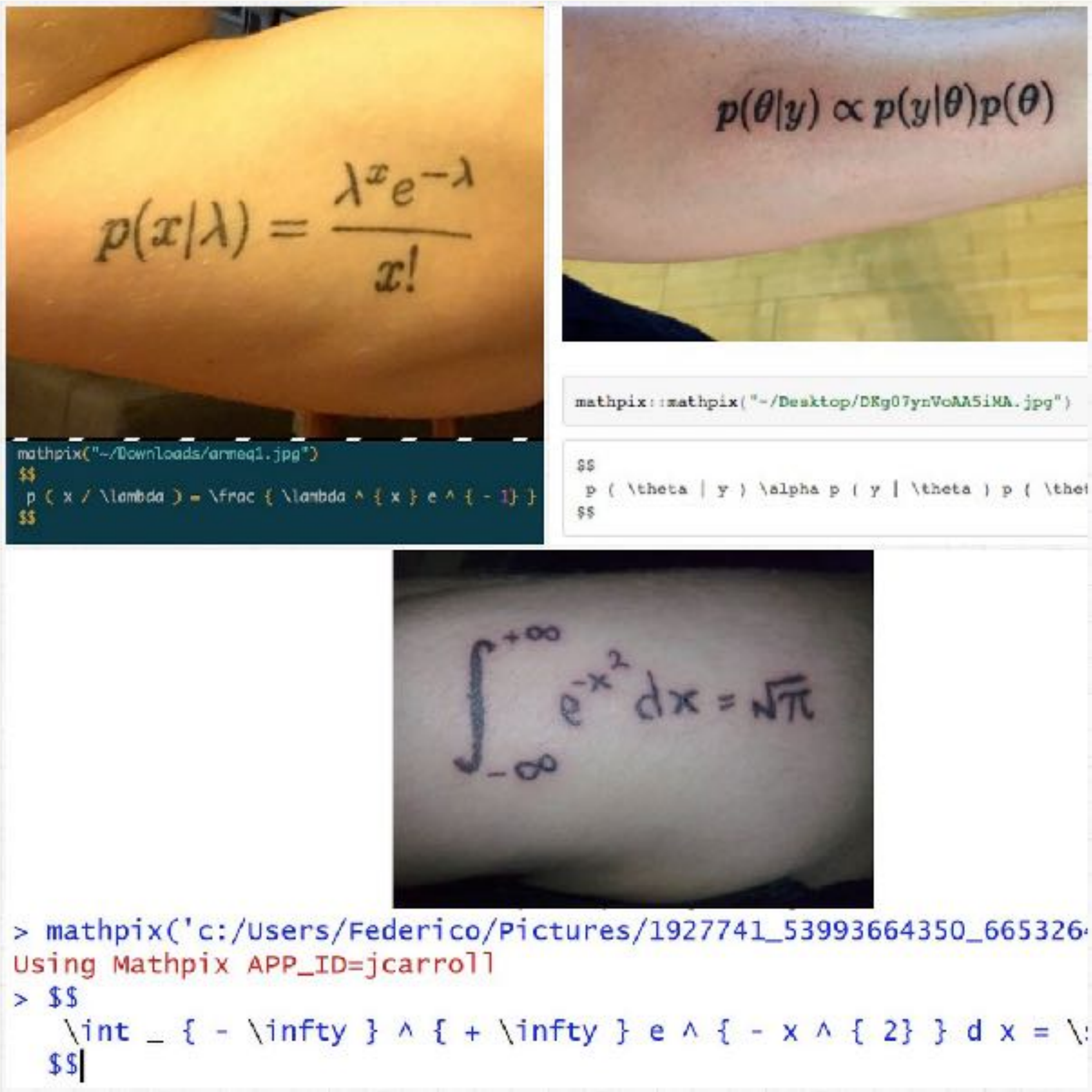
{(),('Alabama'),()}	~~{(),(*word*),()}~~
~~{(' '),(),()}~~	~~{(*word*, ' '),(),()}~~
{(' '),('Alabama'),()}	{(*word*, ' '),('Alabama'),()}
~~{(' '),(*word*),()}~~	~~{(*word*, ' '),(*word*),()}~~
{('in', ' '),(),()}	{(*lowercase*, ' '),(),()}
{('in', ' '),('Alabama'),()}	{(*lowercase*, ' '),('Alabama'),()}
{('in', ' '),(*word*),()}	~~{(*lowercase*, ' '),(*word*),()}~~

(e) {(*lowercase*, ' '),('Alabama'),()} → /[a-z]+ (Alabama)/

Figure 10. Regular Expression Inference. (a) The user selects text in a cell. (b) We tokenize selected and surrounding text. For clarity, the figure only includes two neighboring tokens. For each token, we generate a set of matching labels. (c) We enumerate all label sequences matching the text. (d) We then enumerate all candidate *before*, *selection* and *after* combinations. Patterns that do not uniquely match the selected text are filtered (indicated by strike-through). (e) Finally, we construct regular expressions for each candidate pattern.

http://vis.stanford.edu/papers/wrangler

What about deep learning?


$p(x|\lambda) = \frac{\lambda^x e^{-\lambda}}{x!}$
$p(\theta|y) \propto p(y|\theta)p(\theta)$
$\int_{-\infty}^{+\infty} e^{-x^2} dx = \sqrt{\pi}$

```
mathpix("~/Downloads/armeq1.jpg")
$$
 p ( x / \lambda ) = \frac { \lambda ^ { x } e ^ { - 1} }
$$
```

```
mathpix::mathpix("~/Desktop/DKg07ynVoAA5iMA.jpg")
```

```
$$
 p ( \theta | y ) \alpha p ( y | \theta ) p ( \thet
$$
```

```
> mathpix('c:/Users/Federico/Pictures/1927741_53993664350_665326
Using Mathpix APP_ID=jcarroll
> $$
   \int _ { - \infty } ^ { + \infty } e ^ { - x ^ { 2} } d x = \:
  $$
```

https://twitter.com/carroll_jono/status/914254139873361920

Conclusion

I believe that:

1. Huge advantages to code
2. R provides great environment
3. DSLs help express your thoughts
4. Code should be primary artefact (but might be generated other than typing)