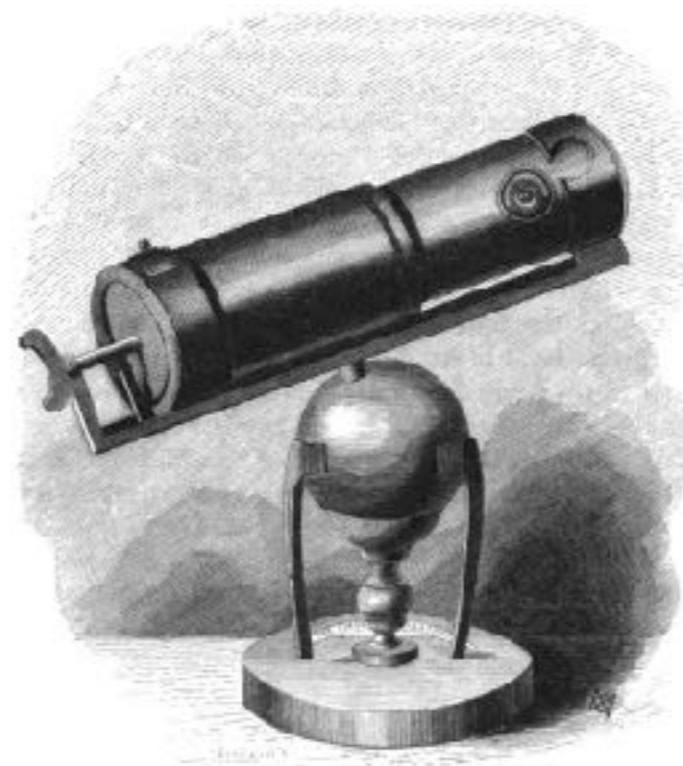


TidyBiology: An Introduction to Biological Data Science in R



Matthew Hirshey, Ph.D.

Duke University Medical Center

Department of Medicine, Pharmacology & Cancer Biology

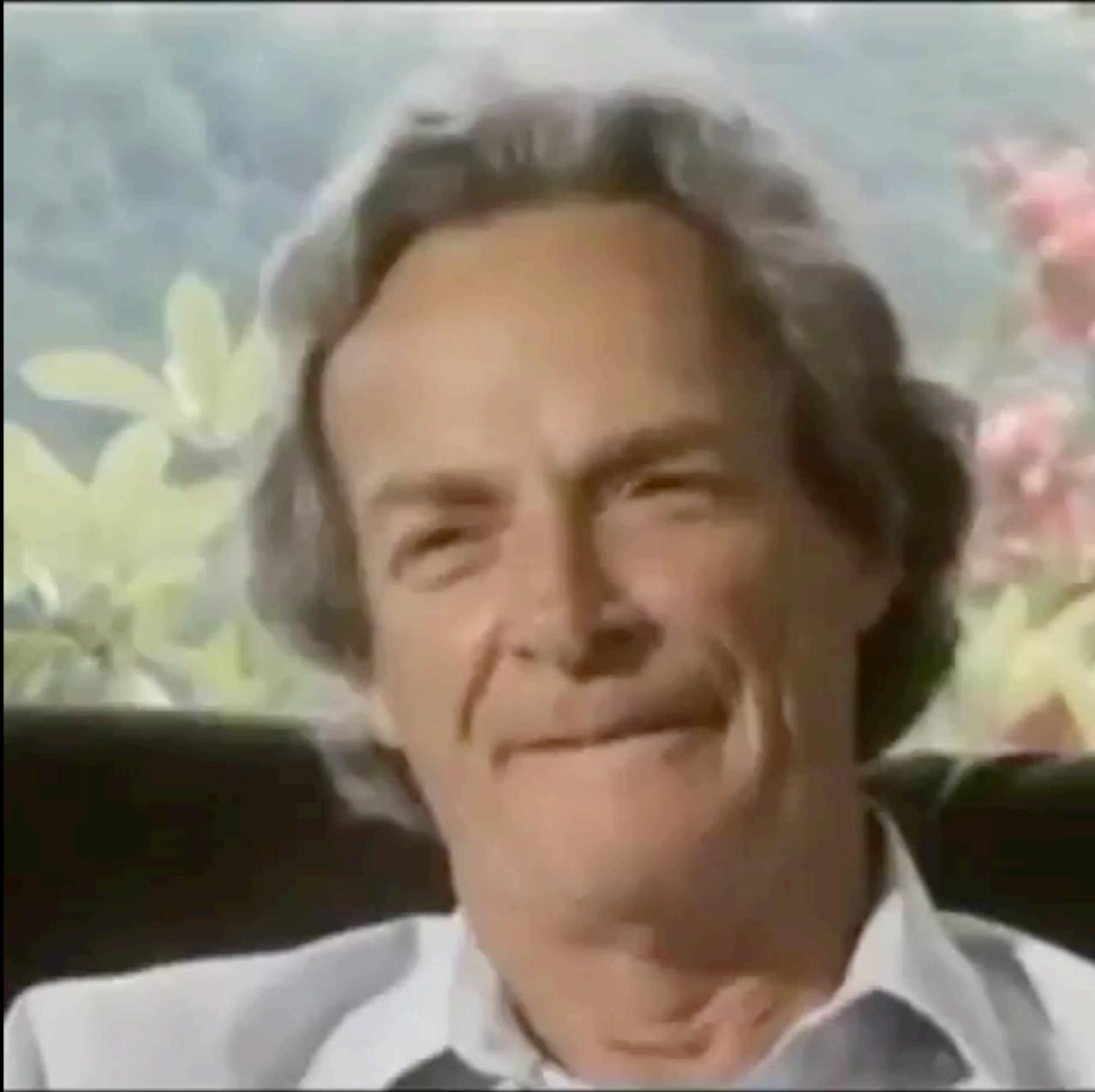
Duke Molecular Physiology Institute

Durham, NC, USA

RESULTS AND DISCUSSION

Identification of YFG as a Potential YFG-ase

Because [this metabolite is unique], we hypothesized that [enzymes] with [special] domains may have YFG-ase activity. To identify potential candidates, we first searched the Swiss-Prot database using the terms "YFG-ase" and "[metabolite]" as keywords and obtained 31 candidate proteins (Figure S1). We next excluded enzymes expressed in yeast, because yeast do not express [enzymes like YFG]. After these screening steps, we obtained nine candidates, including all YFG family members (Figure S1). Among the YFG family members, YFG1 is the most expressed isoform in the liver, where [we think it is important]...So we tested whether YFG1 is a YFG-ase.



“See that bird? It’s a brown-throated thrush, ...and even if you know the name for it, you still know nothing about the bird. You only know something about people; what they call the bird.”

-Richard Feynman

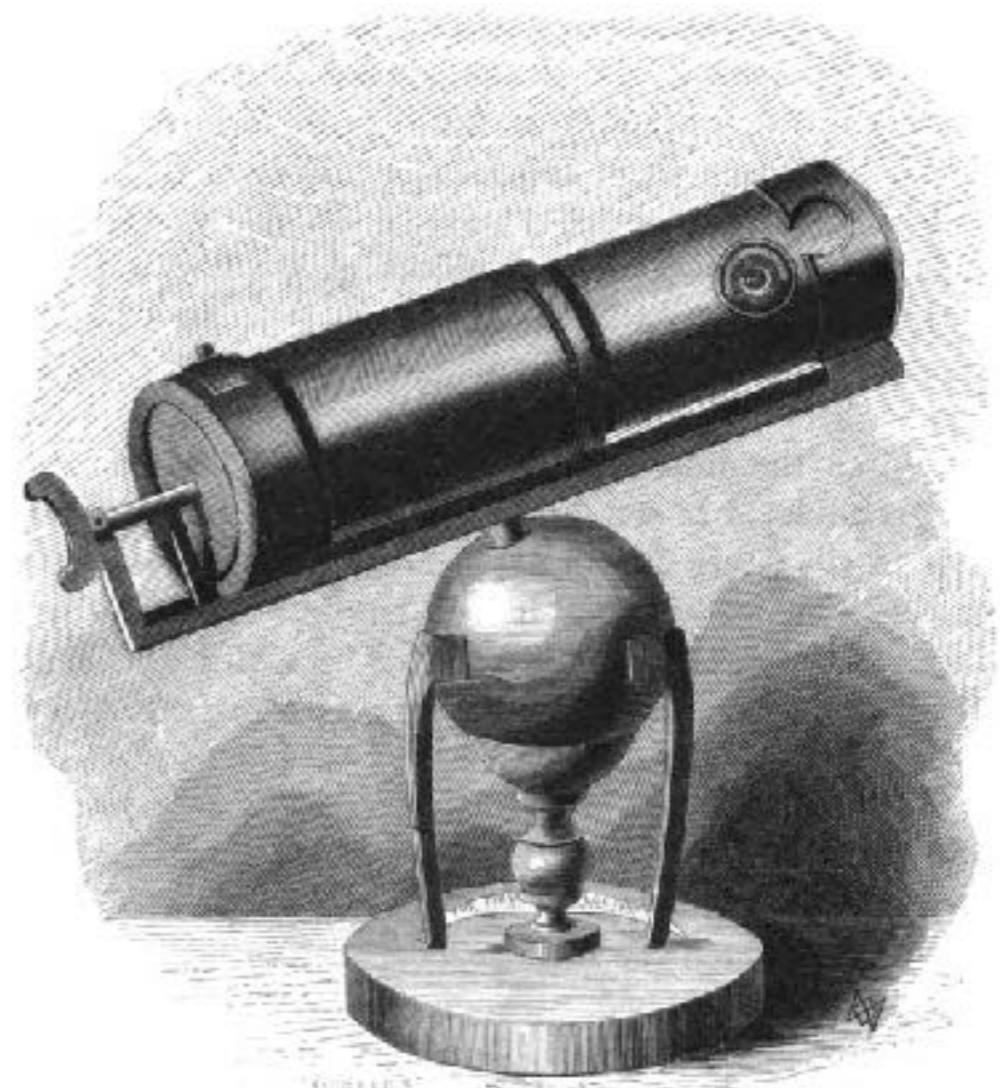
Goal of science is to know something

I don't know what I may seem to the world, but as to myself, I seem to have been only like a boy playing on the sea-shore and diverting myself in now and then finding a smoother pebble or a prettier shell than ordinary, whilst the great ocean of truth lay all undiscovered before me."

— Isaac Newton (1642-1727)

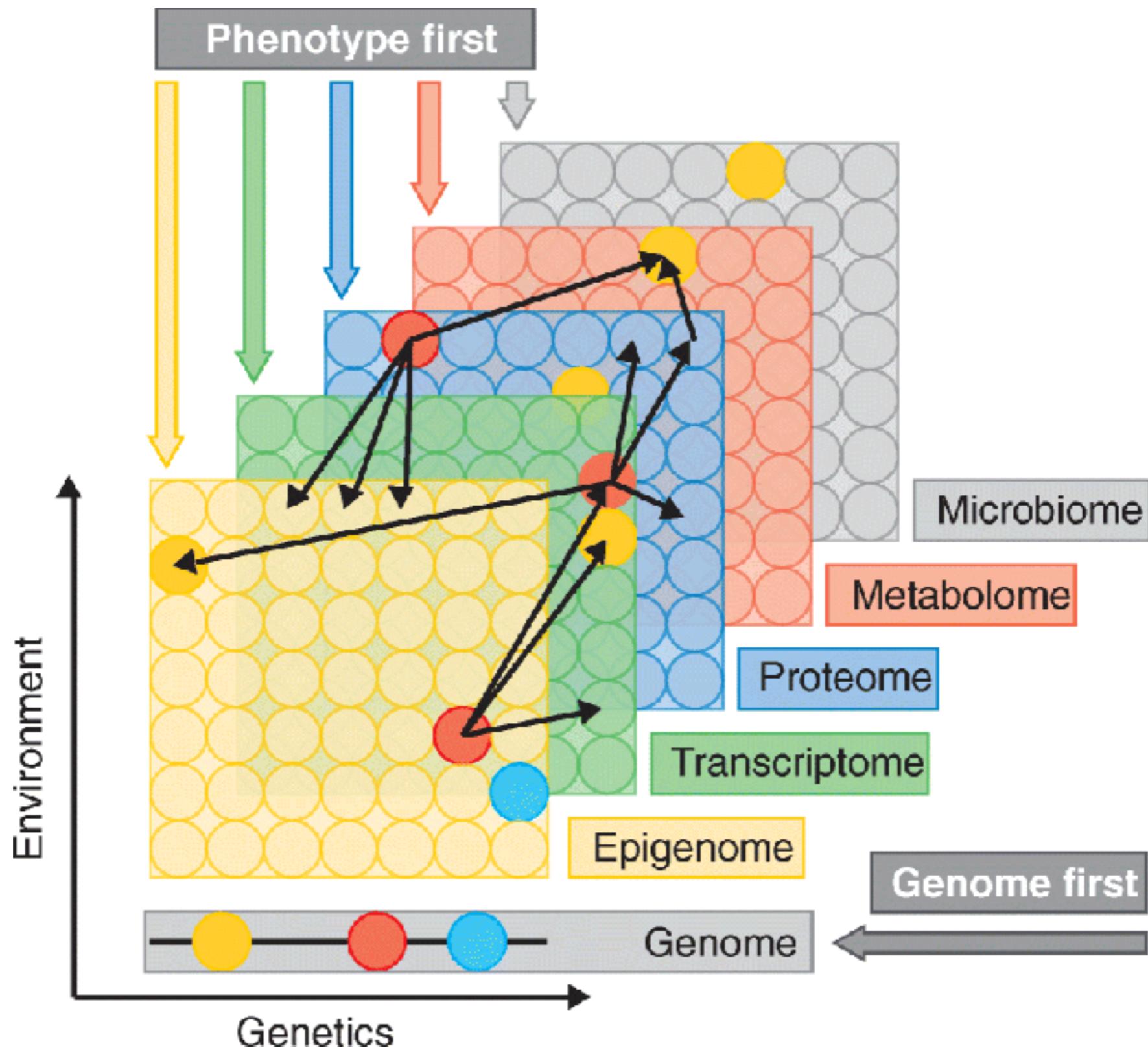
Scientific Method

1. Define a question
2. Observe (gather information and think)
3. Form a hypothesis
4. Test the hypothesis (performing an experiment and collecting data in a reproducible manner)
5. Analyze data (null-hypothesis significance test)
6. Interpret the data and draw conclusions (starting point for new hypotheses)



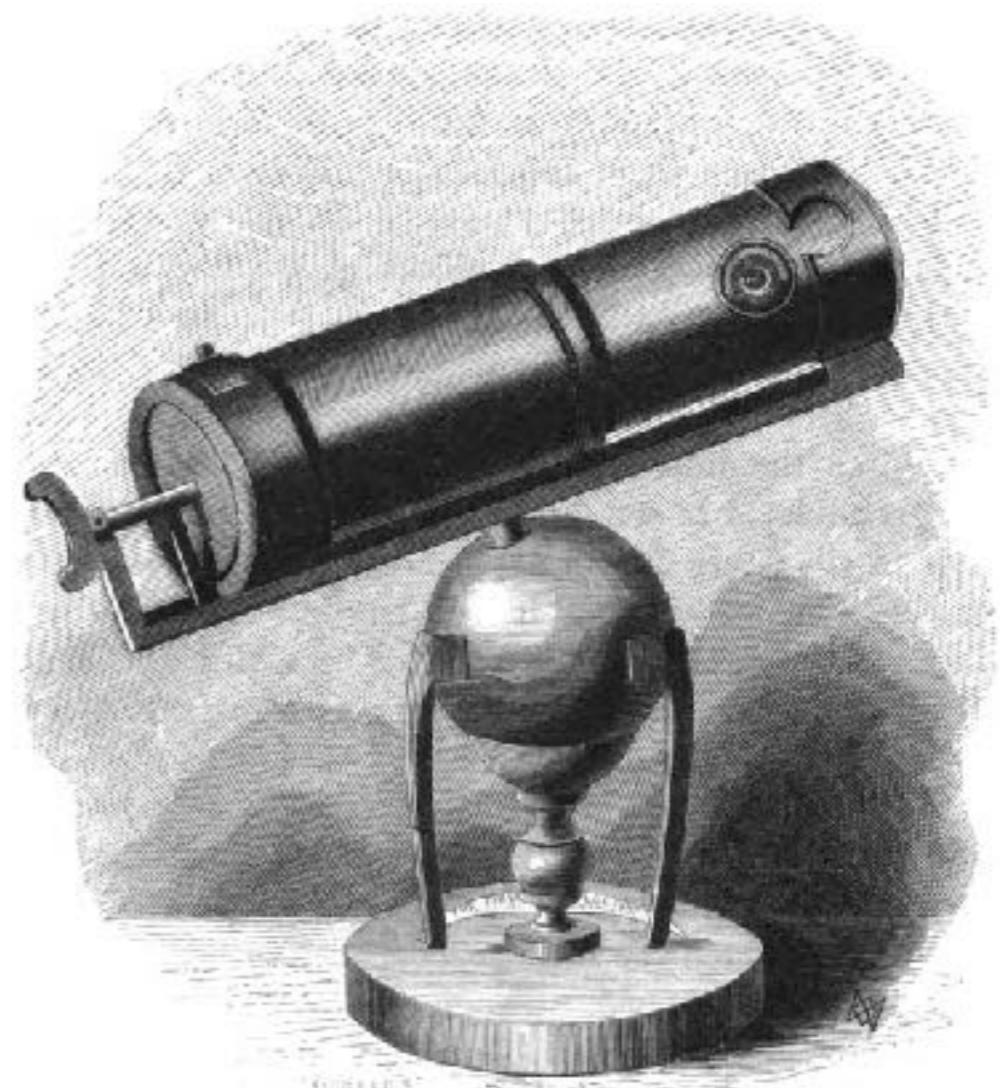
Is there enough evidence to reject the null hypothesis?

Screen

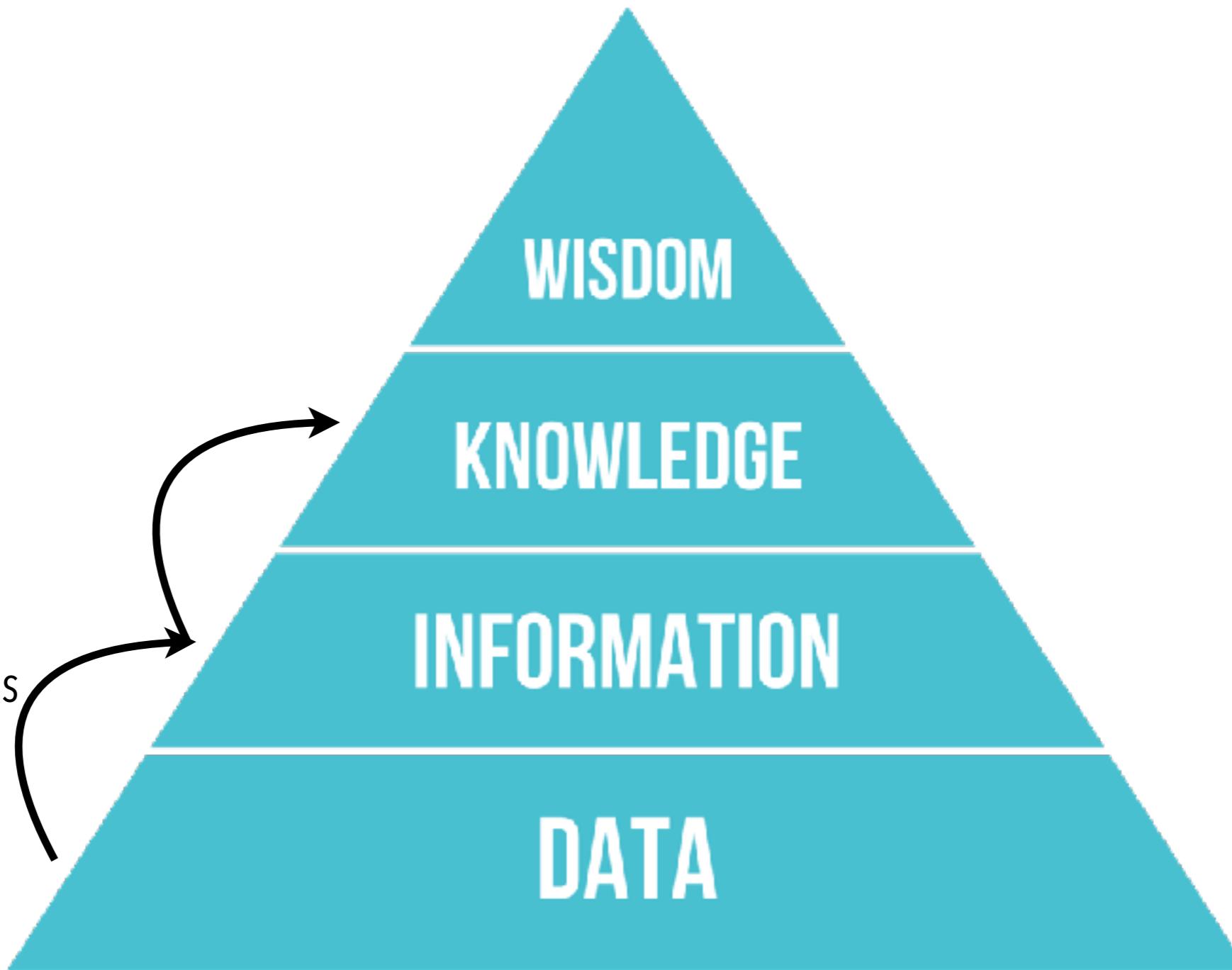


Scientific Method

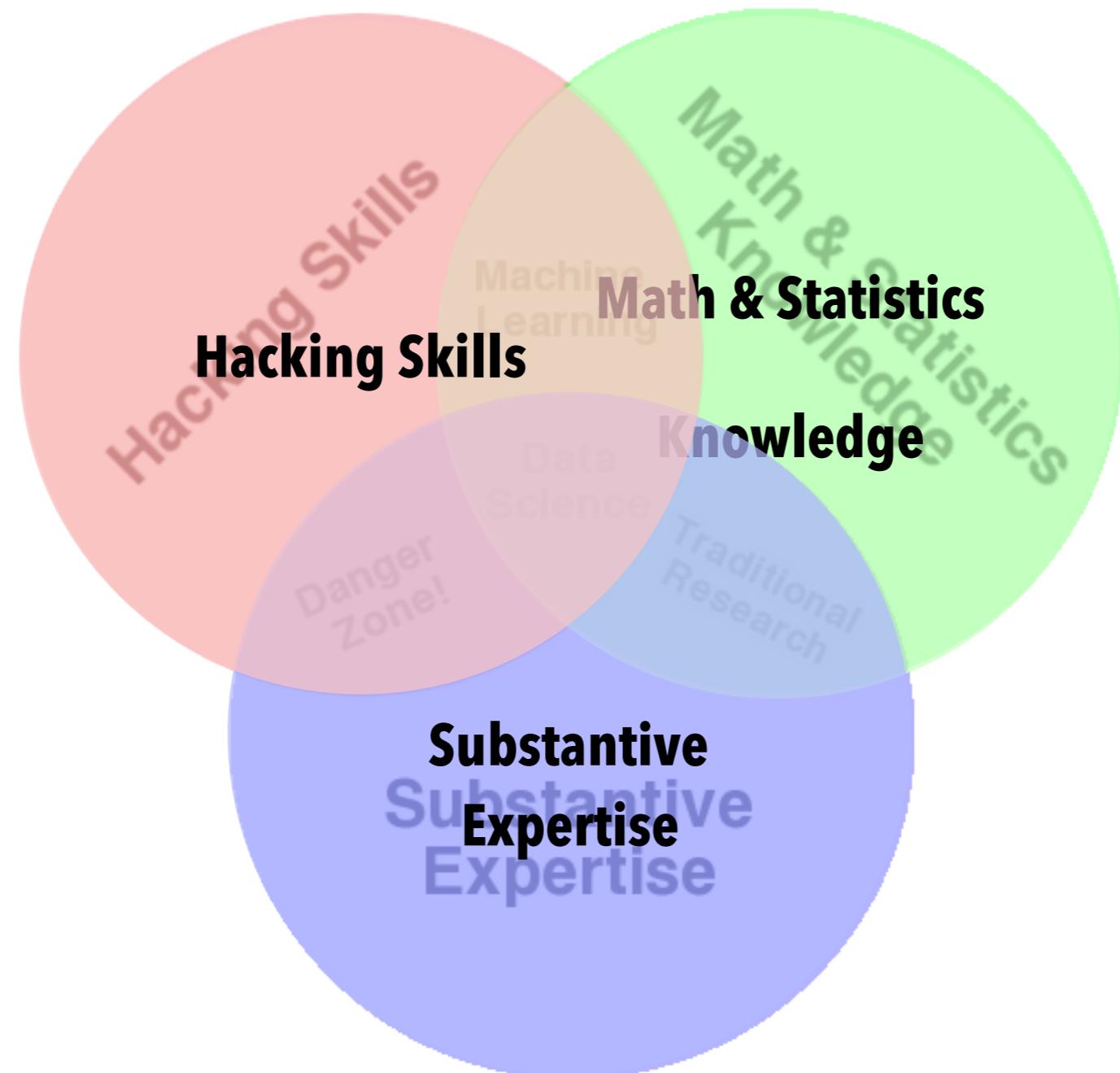
1. Define a question
2. Observe (gather information and think, **screen**)
3. Form a hypothesis
4. Test the hypothesis (performing an experiment and collecting data in a reproducible manner)
5. Analyze data (null-hypothesis significance test)
6. Interpret the data and draw conclusions (starting point for new hypotheses)



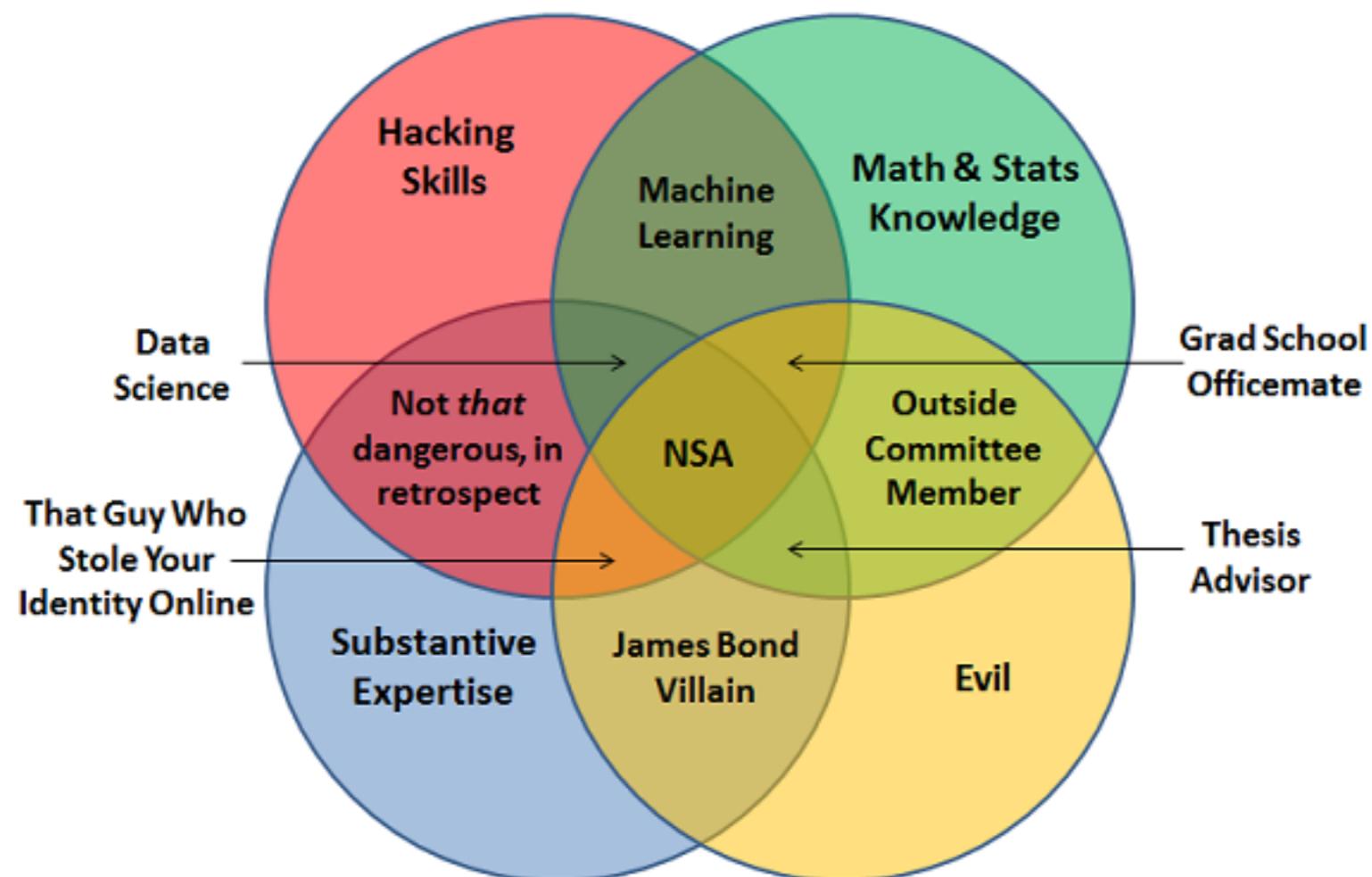
Hierarchy of Knowledge Gathering



Emerging field of Data Science



Venn Diagram of Data Science v2.0



Overall goal is to generate knowledge



Several approaches in data science

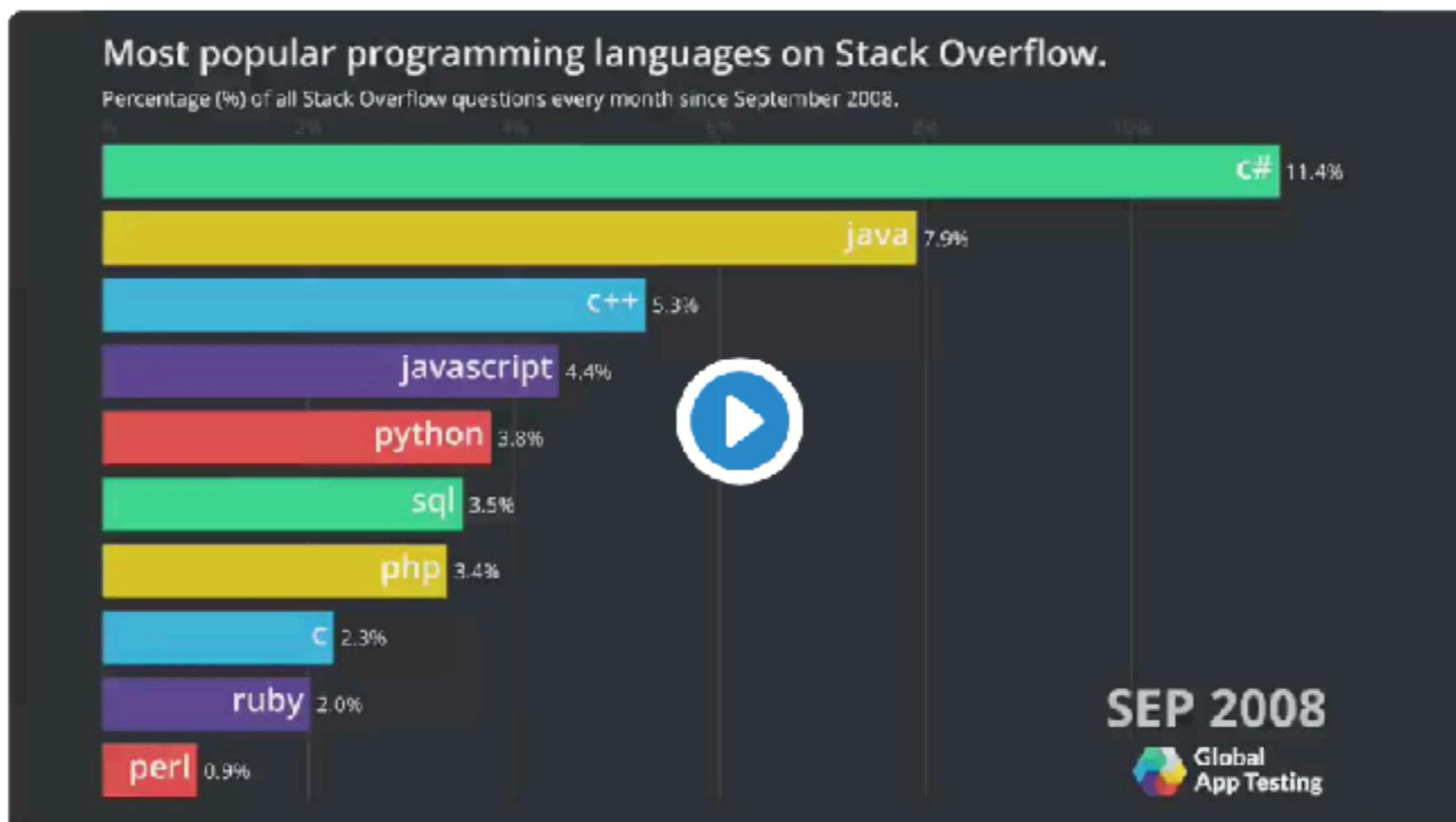


Veronika Romashkina

@vronnie911



StackOverflow language popularity through the years



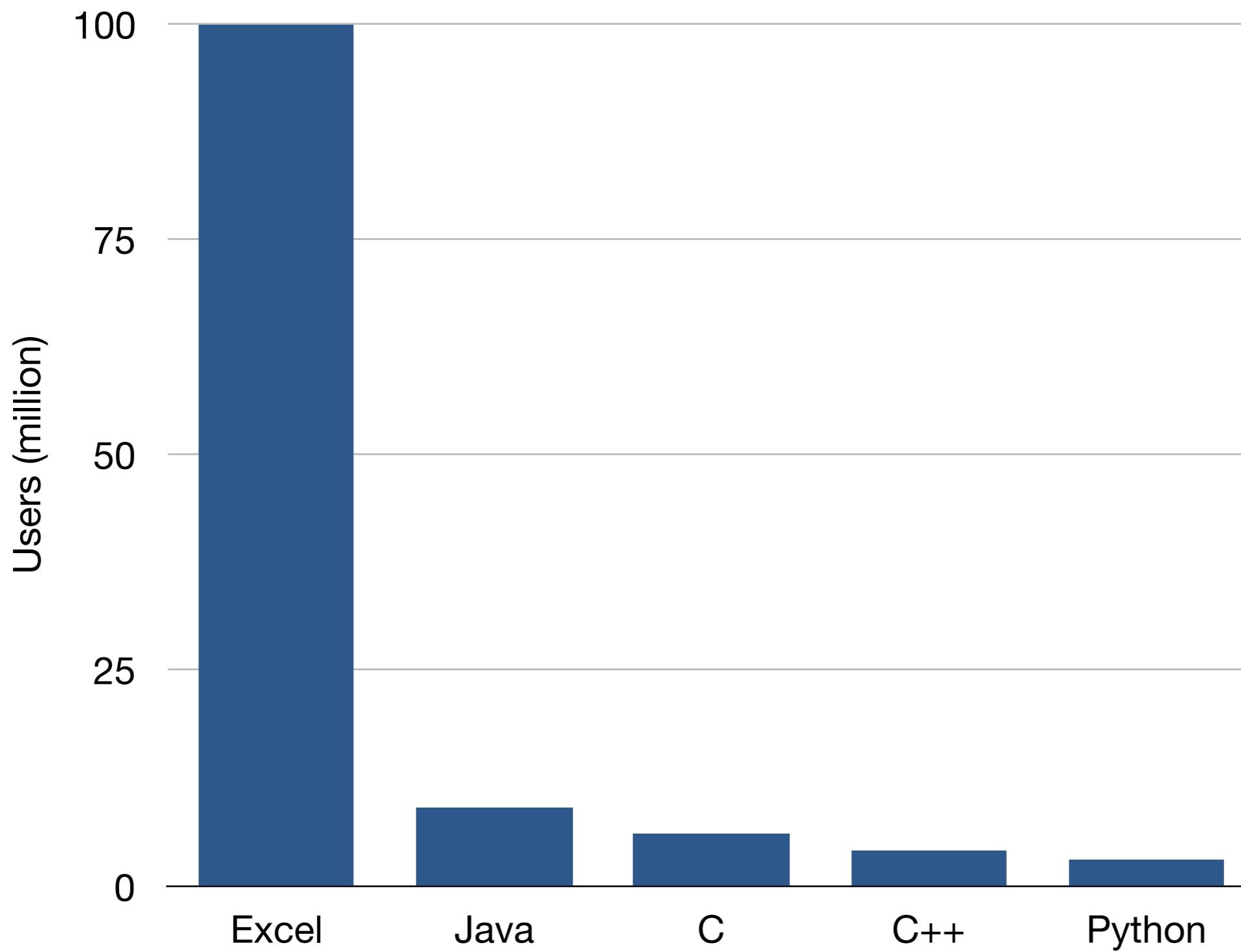
8,301 1:18 PM - May 6, 2019



4,838 people are talking about this



World's most popular programming languages



R
Language

R the language

Values: 1, "Florida", "2010-01-25"

R the language

Values: 1, "Florida", "2010-01-25"

Objects: x <- 22/7

A name without
quotes

< followed by -
(it looks like an
arrow)

A value, object,
or function
result

R the language

Values - 1, "Florida", "2010-01-25"

Objects - x <- c(22/7, 0.99, 3)

To put multiple values in an object,
combine the values with c()

R the language

Values - 1, "Florida", "2010-01-25"

Objects - x <- c(22/7, 0.99, 3)

Functions - round(x, digits = 3)

A name
without
quotes

followed by
() to run the
function

Arguments:
values, objects, or
function results

Which of these are numbers?

1

"1"

"one"

one

Which of these are numbers?

1

number

"1"

"one"

one

Which of these are numbers?

1
number

"1"
"one"
words (strings)

one

Which of these are numbers?

1
number

"1"
words (strings)

one
object

Which of these will work?

Suppose `one <- 1.`

`log(1)`

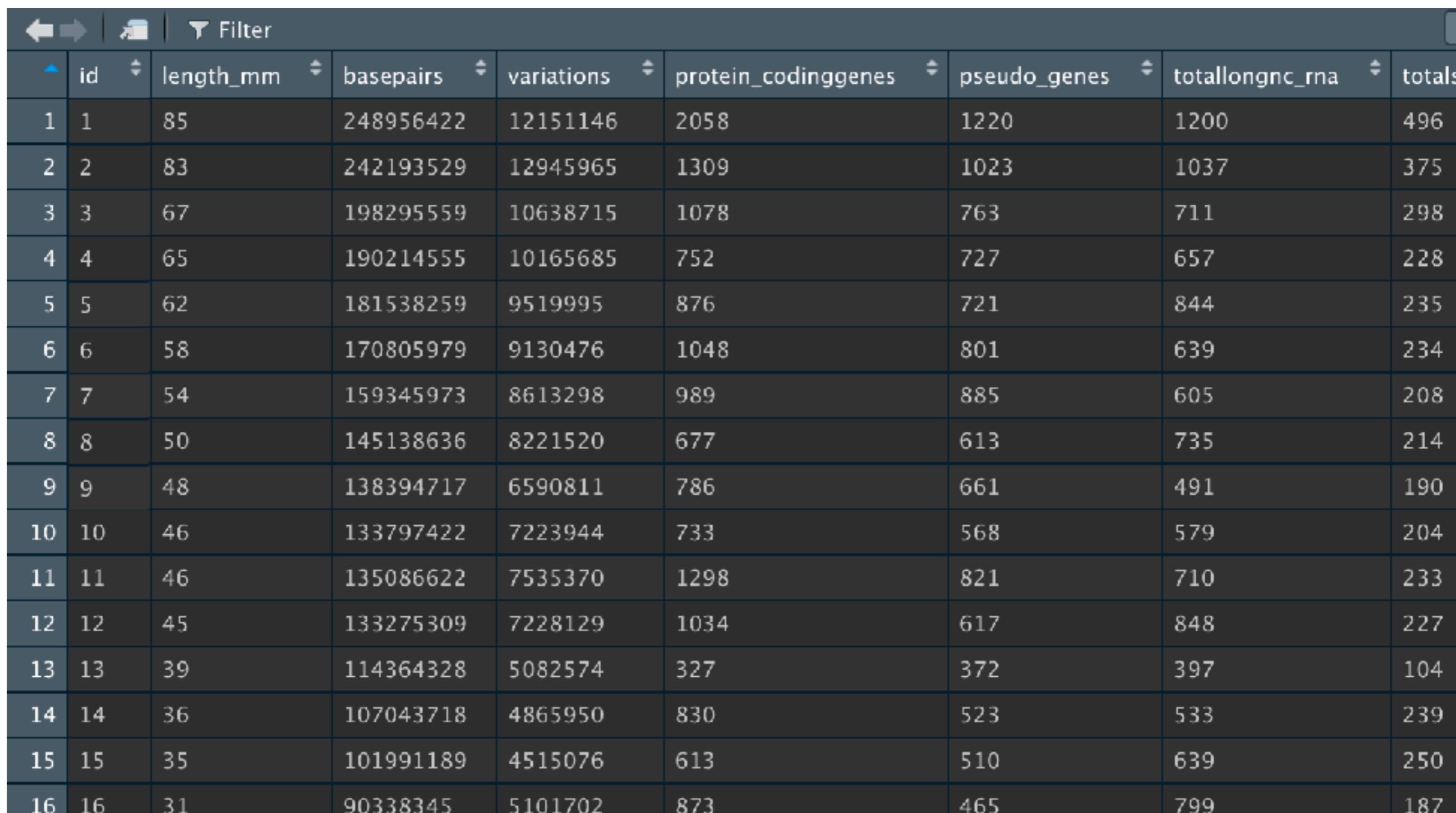
`log("1")`

`log("one")`

`log(one)`

Data are stored in a dataframe

Data stored in a dataframe are conceptually equivalent to a spreadsheet with rows and columns
(chromosome\$id)



The screenshot shows a data viewer interface with a dark header bar containing navigation icons (back, forward, search, filter) and a "Filter" button. The main area is a table with 16 rows and 11 columns. The columns are labeled: id, length_mm, basepairs, variations, protein_codinggenes, pseudo_genes, totallongnc_rna, and totals. The data represents genomic features across chromosomes 1 through 16. The "length_mm" column shows values decreasing from 85 to 31. The "basepairs" column shows values increasing from 248956422 to 90338345. The "variations" column shows values increasing from 12151146 to 5101702. The "protein_codinggenes" column shows values ranging from 2058 to 873. The "pseudo_genes" column shows values ranging from 1220 to 465. The "totallongnc_rna" column shows values ranging from 1200 to 799. The "totals" column shows values ranging from 496 to 187.

	id	length_mm	basepairs	variations	protein_codinggenes	pseudo_genes	totallongnc_rna	totals
1	1	85	248956422	12151146	2058	1220	1200	496
2	2	83	242193529	12945965	1309	1023	1037	375
3	3	67	198295559	10638715	1078	763	711	298
4	4	65	190214555	10165685	752	727	657	228
5	5	62	181538259	9519995	876	721	844	235
6	6	58	170805979	9130476	1048	801	639	234
7	7	54	159345973	8613298	989	885	605	208
8	8	50	145138636	8221520	677	613	735	214
9	9	48	138394717	6590811	786	661	491	190
10	10	46	133797422	7223944	733	568	579	204
11	11	46	135086622	7535370	1298	821	710	233
12	12	45	133275309	7228129	1034	617	848	227
13	13	39	114364328	5082574	327	372	397	104
14	14	36	107043718	4865950	830	523	533	239
15	15	35	101991189	4515076	613	510	639	250
16	16	31	90338345	5101702	873	465	799	187

Extract or create new objects

You can call a single part of the data frame

```
chromosome$id
```

id
1
2
3
4
5
6
7
8
9

And save it as an object for later use

```
ids <- chromosome$id
```

R
IDE



R

```
R version 3.5.2 (2018-12-20) -- "Igganell Igloo"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

R is Free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type "license()" or "licence()" for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type "contributors()" for more information and
"citation()" on how to cite R or R packages in publications.

Type "demo()" for some demos, "help()" for on-line help, or
"help.start()" for an HTML browser interface to help.
Type "q()" to quit R.

[Rapp GUI 1.16 (7982) x86_64-apple-darwin15.6.0]

[Workspace restored from /Users/matthewhinsley/RData]
[History restored from /Users/matthewhinsley/.Rapp.history]

>
```

RStudio

Integrated Developer Environment

The screenshot shows the RStudio interface. The left pane displays a new notebook titled 'Untitled.Rmd' with the following content:

```
1 ---  
2 title: "R Notebook"  
3 output: html_notebook  
4 ---  
5  
6 This is an [R Notebook](https://simardumont.rstudio.com) Notebook. When you execute code within the notebook,  
the results appear beneath the code.  
7  
8 Try executing this chunk by clicking the "Run" button within the chunk or by placing your cursor inside it  
and pressing Cmd+Shift+Enter.  
9  
10 ## {r}  
11 plot(cars)  
12  
13 Add a new chunk by clicking the "Insert Chunk" button on the toolbar or by pressing Cmd+Option+I.  
14  
15 When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the "Preview" button or press Cmd+Shift+P to preview the HTML file).  
16  
17 The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike "Knit",  
"Preview" does not run any R code chunks. Instead, the output of the chunk when it was last run in the  
editor is displayed.  
18  
19 ## [1] "R Notebook"
```

The right pane shows the file browser with the following structure:

File	Plot	Packages	Help	Viewer
<input checked="" type="checkbox"/> NewFolder	<input type="checkbox"/> Delete	<input type="checkbox"/> Rename	<input type="checkbox"/> More	
<input checked="" type="checkbox"/> Home : Downloads > DUGI > PROJECTS > proteinomics_workflow				
A Name	Size	Modified		
<input type="checkbox"/>				
<input type="checkbox"/> code.Rmd				
<input type="checkbox"/> data				
<input type="checkbox"/> ideas				



Demo

What am I looking at?

R v. Rstudio

Go to 01_intro

Open 01_demo.Rmd

Follow along if you'd like.

R Markdown

R Markdown Introduction

Go to 01_intro

Open 01_exercise.Rmd

Read through the file and do everything it tells you to do.



R Markdown

An authoring format for Data Science.

The screenshot shows the RStudio interface with an R Markdown file open. The code editor pane contains the following content:

```
1: ---
2: title: "R Notebook"
3: output: html_notebook
4:
5:
6: Text written in **markdown**
7:
8: ```{r}
9: # code written in R
10: (x <- rnorm(7))
11:
12: Text written in _markdown_
13:
14: ```{r}
15: # code written in R
16: hist(x)
17:
18:
19: Text written in __markdown__
20:
21: ```{r}
22: # code written in R
23:
24: 
```

The R console pane shows the output of the R code:

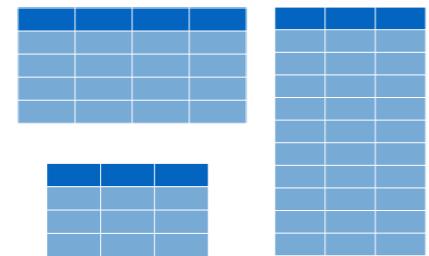
```
[1] -1.2  1.0 -0.5  0.9 -0.6 -1.1 -1.5
```

Three callout boxes point to specific parts of the interface:

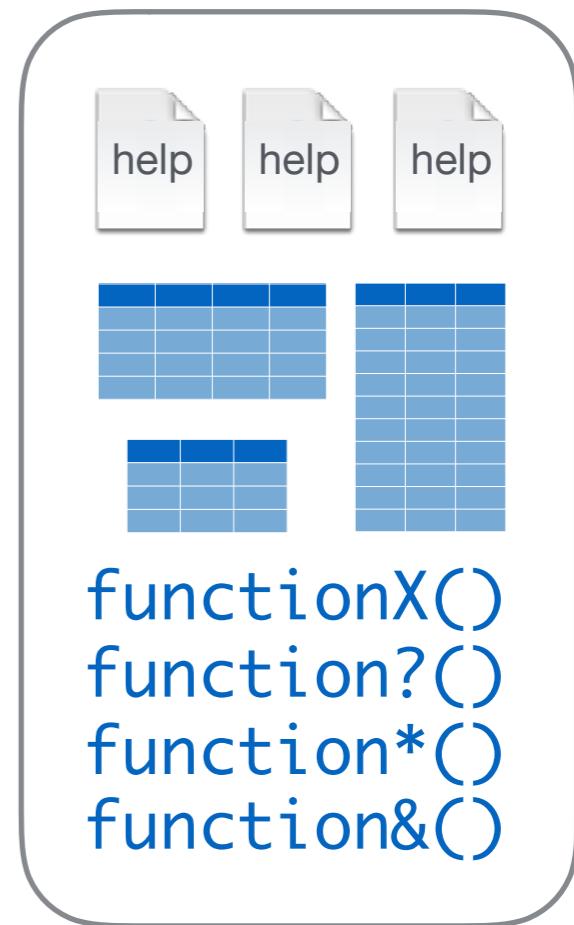
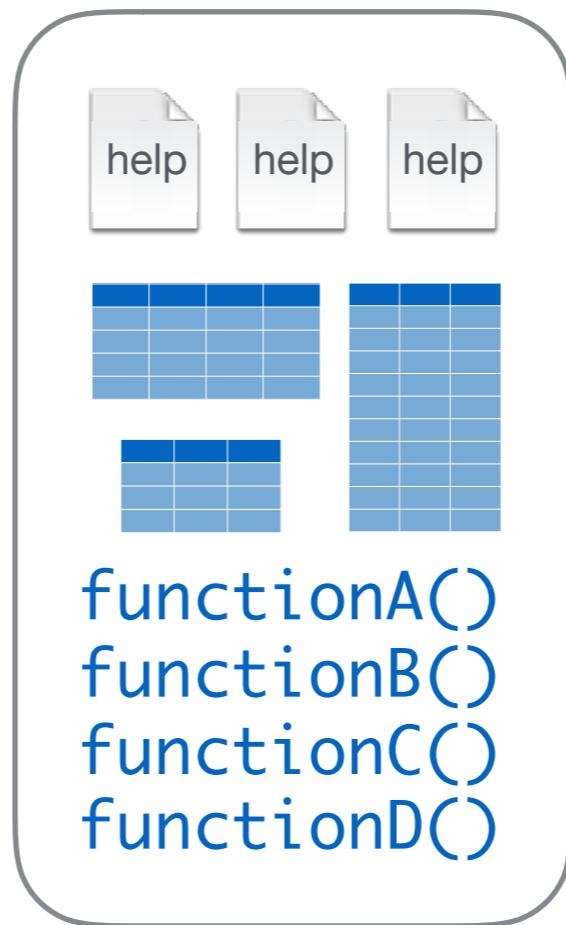
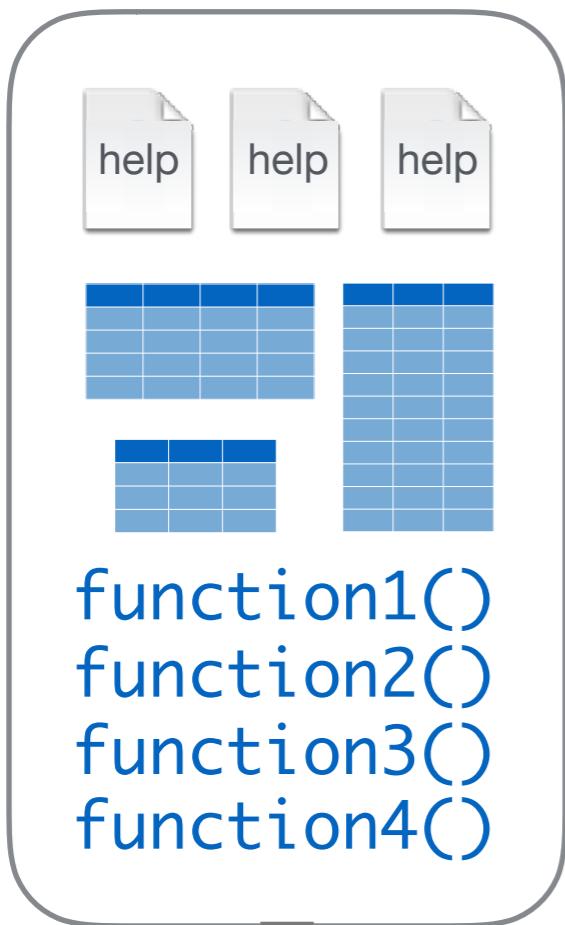
- A grey box points to the code chunk starting at line 8, containing the text "Click to run code in chunk".
- A grey box points to the "Code result" output at the bottom of the code editor, containing the text "Code result".
- A grey box points to the "Preview" tab in the top bar, containing the text "Code goes in a chunk".

R Packages

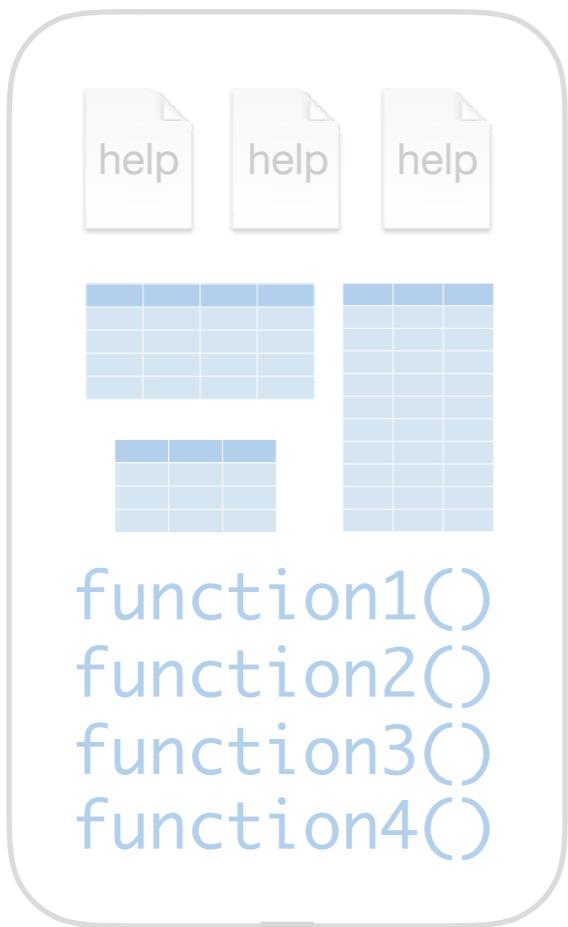




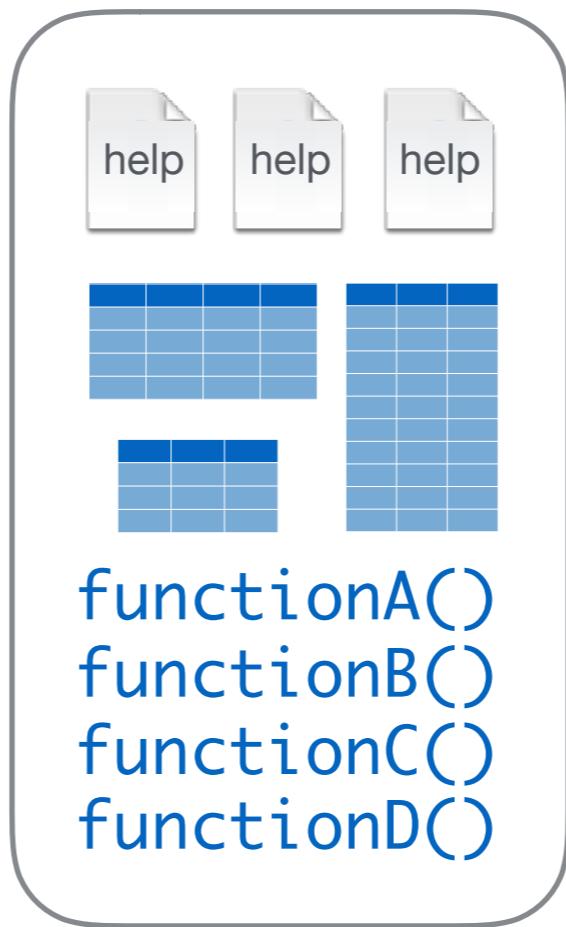
function1()
function2()
function3()
function4()



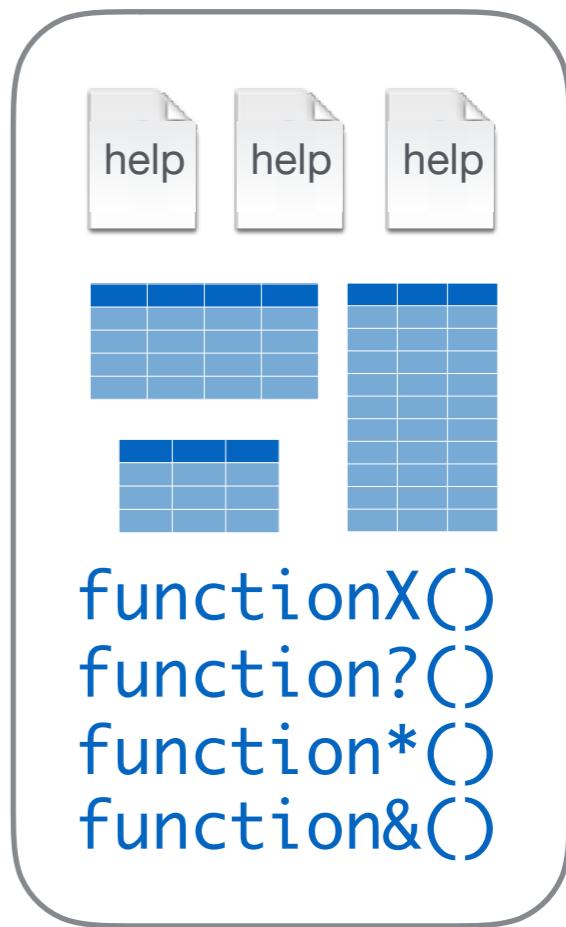
Base R



Base R



R Packages



“Verbs” (i.e. functions) act on data

These “verbs” act on data

```
do_this(to_that)  
do_this(to_that, using_these)
```

Packages contain functions, documentation, data

Package





dplyr part of the [tidyverse](#)
0.8.3

Overview

dplyr is a grammar of data manipulation, providing a consistent set of verbs that help you solve the most common data manipulation challenges:

- `mutate()` adds new variables that are functions of existing variables
- `select()` picks variables based on their names.
- `filter()` picks cases based on their values.
- `summarise()` reduces multiple values down to a single summary.
- `arrange()` changes the ordering of the rows.

The Comprehensive R Archive Garrett

Secure | https://cran.r-project.org



[CRAN
Mirrors](#)
[What's new?](#)
[Task Views](#)
[Search](#)

[About R](#)
[R Homepage](#)
[The R Journal](#)

[Software](#)
[R Sources](#)
[R Binaries](#)
[Packages](#)
[Other](#)

[Documentation](#)
[Manuals](#)
[FAQs](#)
[Contributed](#)

[A3](#)
[abyyR](#)
[abc](#)
[ABCAnalysis](#)
[abc.data](#)
[abcdeFBA](#)

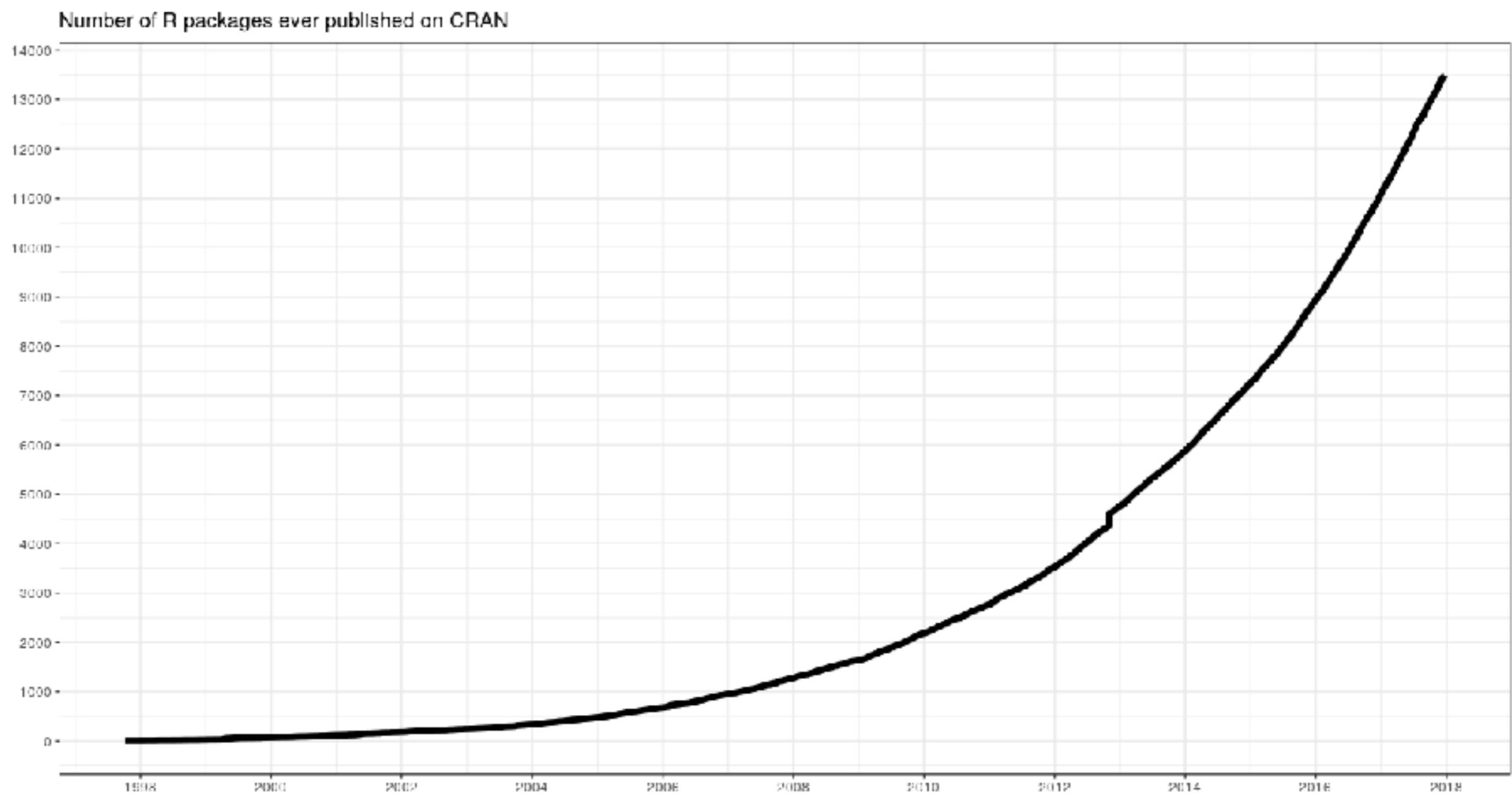
[ABCOptim](#)
[ABCp2](#)
[ABC.RAP](#)
[abcrf](#)
[abctools](#)
[ahd](#)
[abf2](#)
[ABHgenotypeR](#)
[abind](#)
[abjutils](#)
[ahn](#)
[abodOutlier](#)
[AbsFilterGSEA](#)
[AbSim](#)
[abundant](#)
[ACA](#)
[acc](#)
[accelrometry](#)
[accelmissing](#)
[AcceptanceSampling](#)

Available CRAN Packages By Name

[A](#) [B](#) [C](#) [D](#) [E](#) [F](#) [G](#) [H](#) [I](#) [J](#) [K](#) [L](#) [M](#) [N](#) [O](#) [P](#) [Q](#) [R](#) [S](#) [T](#) [U](#) [V](#) [W](#) [X](#) [Y](#) [Z](#)

Accurate, Adaptable, and Accessible Error Metrics for Predictive Models
Access to Abbyy Optical Character Recognition (OCR) API
Tools for Approximate Bayesian Computation (ABC)
Computed ABC Analysis
Data Only: Tools for Approximate Bayesian Computation (ABC)
ABCDE_FBA: A-Biologist-Can-Do-Everything of Flux Balance Analysis with this package
Implementation of Artificial Bee Colony (ABC) Optimization
Approximate Bayesian Computational Model for Estimating P2
Array Based CpG Region Analysis Pipeline
Approximate Bayesian Computation via Random Forests
Tools for ABC Analyses
The Analysis of Biological Data
Load Gap-Free Axon ABF2 Files
Easy Visualization of ABH Genotypes
Combine Multidimensional Arrays
Useful Tools for Jurimetrical Analysis Used by the Brazilian Jurimetrics Association
Modelling Multivariate Data with Additive Bayesian Networks
Angle-Based Outlier Detection
Improved False Positive Control of Gene-Permuting GSEA with Absolute Filtering
Time Resolved Simulations of Antibody Repertoires
High-Dimensional Principal Fitted Components and Abundant Regression
Abrupt Change-Point or Aberration Detection in Point Series
Exploring Accelerometer Data
Functions for Processing Minute-to-Minute Accelerometer Data
Missing Value Imputation for Accelerometer Data
Creation and Evaluation of Acceptance Sampling Plans

14108
packages
as of 5/2019



Using packages

Step 1

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

Downloads files to computer

1 x per
computer



R packages for data science

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

Pop Quiz!

The tidyverse contains the following packages. How would you install them?

ggplot2	tibble	DBI	
dplyr	hms	haven	rvest
tidyr	stringr	httr	xml2
readr	lubridate	jsonlite	modelr
purrr	forcats	readxl	tidyverse

```
install.packages("ggplot2")
install.packages("dplyr")
install.packages("tidyr")
install.packages("readr")
install.packages("purrr")
install.packages("tibble")
install.packages("hms")
install.packages("stringr")
install.packages("lubridate")
install.packages("forcats")
install.packages("DBI")
install.packages("haven")
install.packages("httr")
install.packages("jsonlite")
install.packages("readxl")
install.packages("rvest")
install.packages("xml2")
install.packages("modelr")
install.packages("broom")
```

tidyverse



An R package that serves as a short cut for installing and loading the components of the tidyverse.

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

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

does the equivalent of

```
install.packages("ggplot2")
install.packages("dplyr")
install.packages("tidyr")
install.packages("readr")
install.packages("purrr")
install.packages("tibble")
install.packages("hms")
install.packages("stringr")
install.packages("lubridate")
install.packages("forcats")
install.packages("DBI")
install.packages("haven")
install.packages("httr")
install.packages("jsonlite")
install.packages("readxl")
install.packages("rvest")
install.packages("xml2")
install.packages("modelr")
install.packages("broom")
```

I've already installed all of
the packages you need

Using packages

Step1

```
install.packages("tidybiology")
```

Downloads files to computer

1 x per computer

Step2

```
library("tidybiology")
```

Loads package

1 x per R Session

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

does the equivalent of

```
install.packages("ggplot2")
install.packages("dplyr")
install.packages("tidyr")
install.packages("readr")
install.packages("purrr")
install.packages("tibble")
install.packages("stringr")
install.packages("forcats")
install.packages("lubridate")
install.packages("hms")
install.packages("DBI")
install.packages("haven")
install.packages("httr")
install.packages("jsonlite")
install.packages("readxl")
install.packages("rvest")
install.packages("xml2")
install.packages("modelr")
install.packages("broom")
```

```
library("tidyverse")
```

does the equivalent of

```
library("ggplot2")
library("dplyr")
library("tidyr")
library("readr")
library("purrr")
library("tibble")
library("stringr")
library("forcats")
```

Setup

The setup chunk is always run once before anything else

```
1 ---  
2 title: "Data Visualization"  
3 output: html_notebook  
4 ---  
5  
6 ```{r setup}  
7 library(tidyverse)  
8 ...  
9  
10 ```{r}  
11 mpg  
12 ...  
13  
14  
15 ## Your Turn 1  
16  
17 Run the code on the slide to make a graph. Pay strict attention to  
spelling, capitalization, and parentheses!  
18  
19 ```{r}  
20  
21 ...  
22  
23 ## Your Turn 2  
24  
25 Add `color`, `size`, `alpha`, and `shape` aesthetics to your graph.  
Experiment.  
26  
27 ```{r}  
28 ggplot(data = mpg) +  
29   geom_point(mapping = aes(x = displ, y = hwy))  
30 ...  
31  
32 ## Your Turn 3  
33  
34 Replace this scatterplot with one that draws boxplots. Use the cheatsheet.  
Try your best guess.  
35  
36 ```{r}  
37 ggplot(mpg) + geom_point(aes(class, hwy))  
38 ...  
39
```

chunk labels are optional, the setup label is special

Exercise

Add a setup chunk (as shown below) to the top of 01_exercise.Rmd.
Use it to load the tidyverse package (*remember to run this chunk)
Then uncomment and run the final code chunk at the bottom of your file.



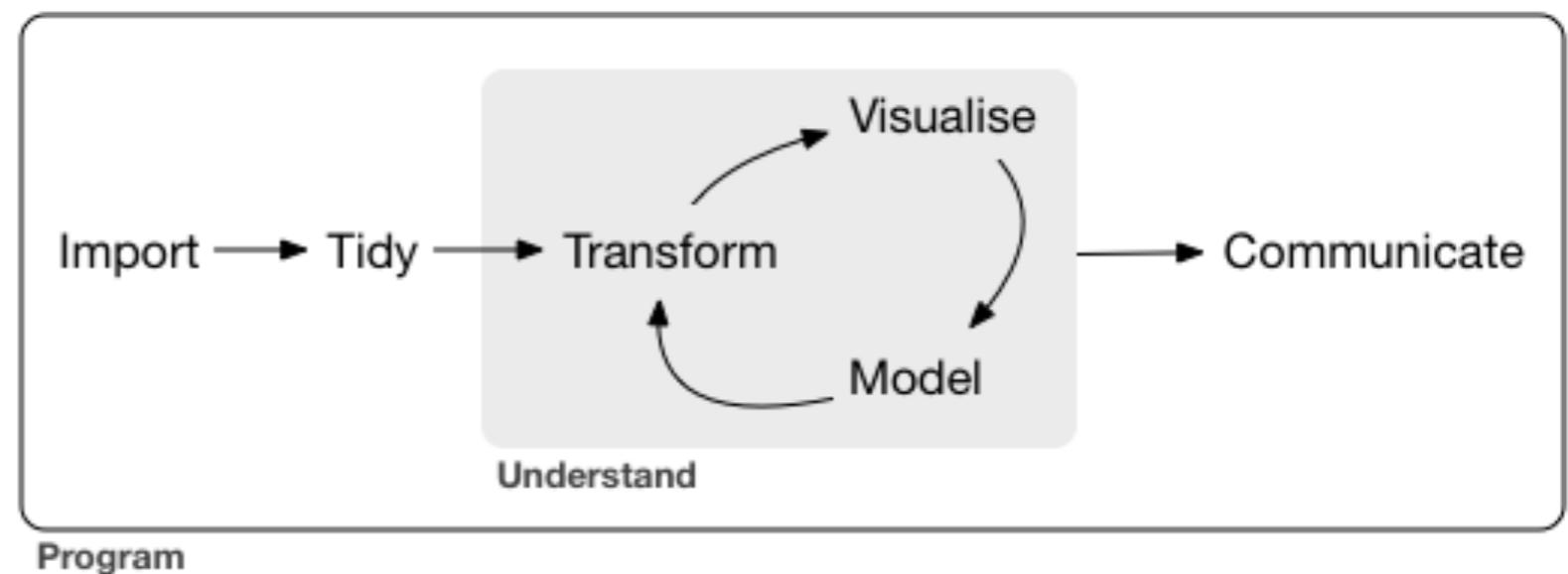
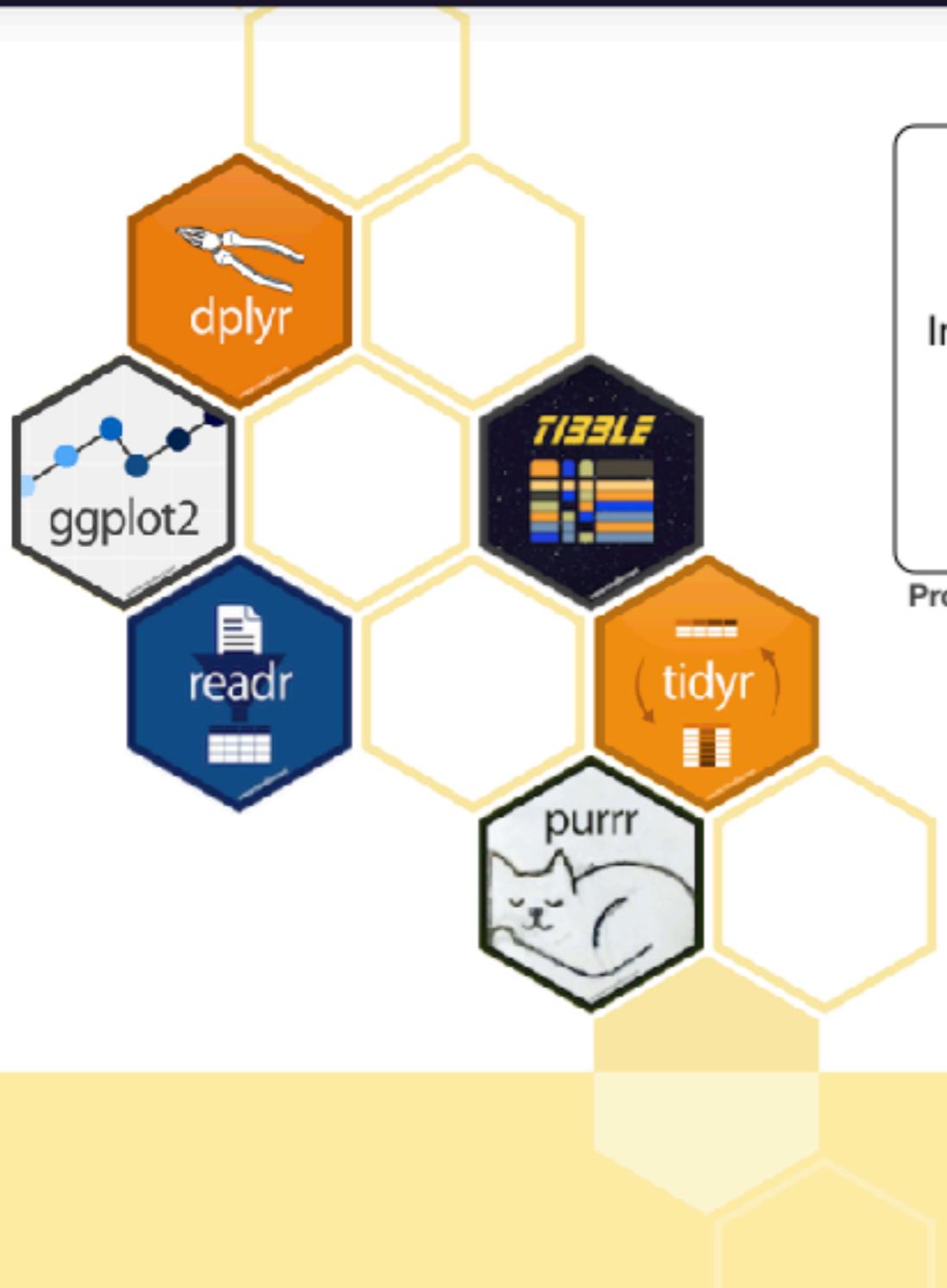
```
1 ---  
2 title: "Data Visualization"  
3 output: html_notebook  
4 ---  
5  
6 ```{r setup}  
7 library(tidyverse)  
8 ```  
9  
10 ```{r}  
11 mpg  
12  
13  
14  
15 ## Your Turn 1  
16  
17 Run the code on the slide to make a graph. Pay strict attention to  
spelling, capitalization, and parentheses!  
18  
19 ```{r}  
20  
21 ...  
22  
23 ## Your Turn 2  
24  
25 Add `color`, `size`, `alpha`, and `shape` aesthetics to your graph.  
Experiment.  
26  
27 ```{r}  
28 ggplot(data = mpg) +  
29   geom_point(mapping = aes(x = displ, y = hwy))  
30 ...  
31  
32 ## Your Turn 3
```

chunk labels are optional, the setup label is special

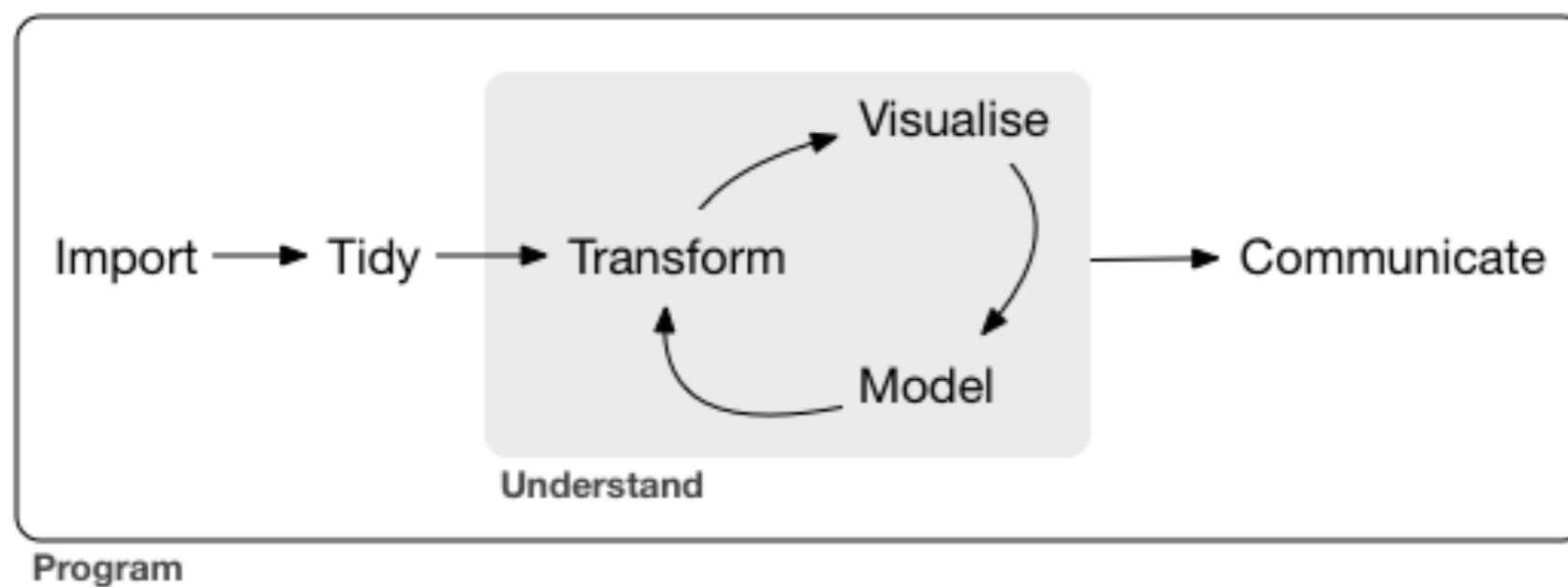


Tidyverse

Packages Articles Learn Help Contribute



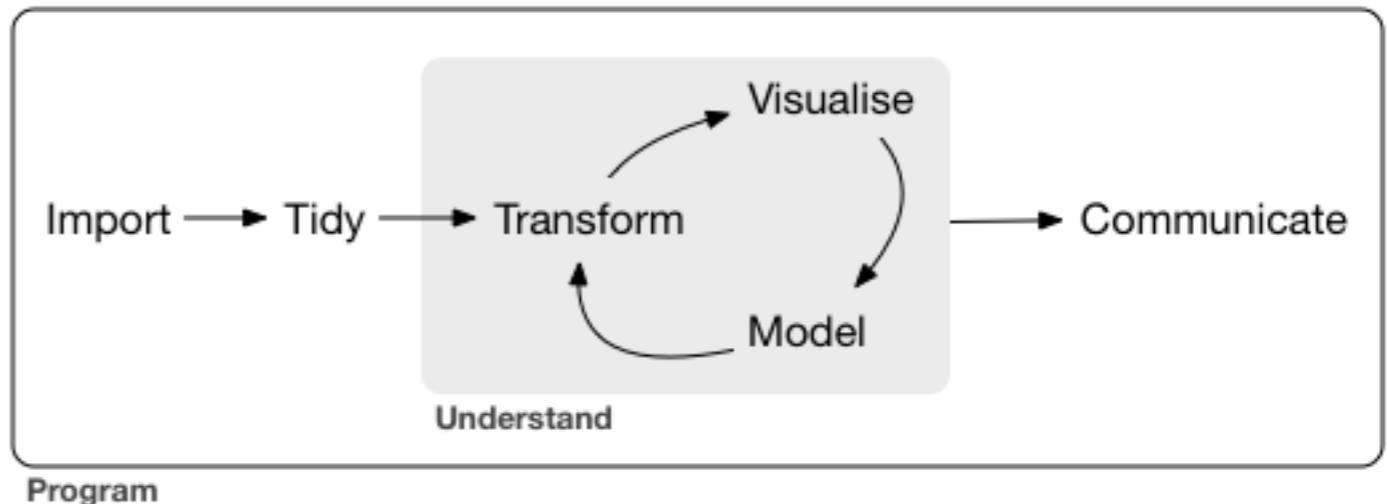
Exploratory Data Analysis (EDA)



Basic Tidyverse Principles

IMPORT(readr):

- » `read_csv()`
- » `read_tsv()`
- » `read_delim()`



TIDY & TRANSFORM(dplyr):

- » `mutate()` adds new variables that are functions of existing variables
- » `select()` picks variables based on their names.
- » `filter()` picks cases based on their values.
- » `summarise()` reduces multiple values down to a single summary.
- » `arrange()` changes the ordering of the rows.

VISUALISE(ggplot): creating graphics, based on 'The Grammar of Graphics'

- » `aes()`
- » `geom_x() + layers`

MODEL(broom):

- » `tidy()`, `glance()`, `augment()`
- » `lm(~)`



Ceci n'est pas une pipe.

%>%
magrittr

Ceci n'est pas un pipe.

magrittr package by Stefan Milton Bache developed the concept of the pipe, which is used heavily in the tidyverse

“and then”

The “pipe” is a sequence of functions, that are sequentially applied to an object

```
wakeup(self) %>%  
  put_on("clothes") %>%  
  eat("breakfast") %>%  
  go(to = "work")
```

Alternative nested code

```
go(eat(put_on(wakeup(self), "clothes"), "brekfast"), to = "work")
```

What does this code do?

```
wakeup(self) %>%  
  put_on("clothes") %>%  
  eat("breakfast") %>%  
  fmk() %>%  
  go(to = "work")
```

```
morning_routine <- wakeup(self) %>%  
  put_on("clothes") %>%  
  eat("breakfast") %>%  
  fmk() %>%  
  go(to = "work")
```

The “pipe” is a sequence of functions, that are sequentially applied to an object

What does this code do?

```
chromosome %>%
  select("id", "variations", "basepairs") %>%
  mutate(percent_var = variations/basepairs) %>%
  arrange(desc(percent_var))
```

Project

Go to 01_Intro

Open 01_homework.Rmd

Read through the file and the code, to do everything it tells you to do.

Writing code is NOT like drawing an owl

How to draw an owl

1.



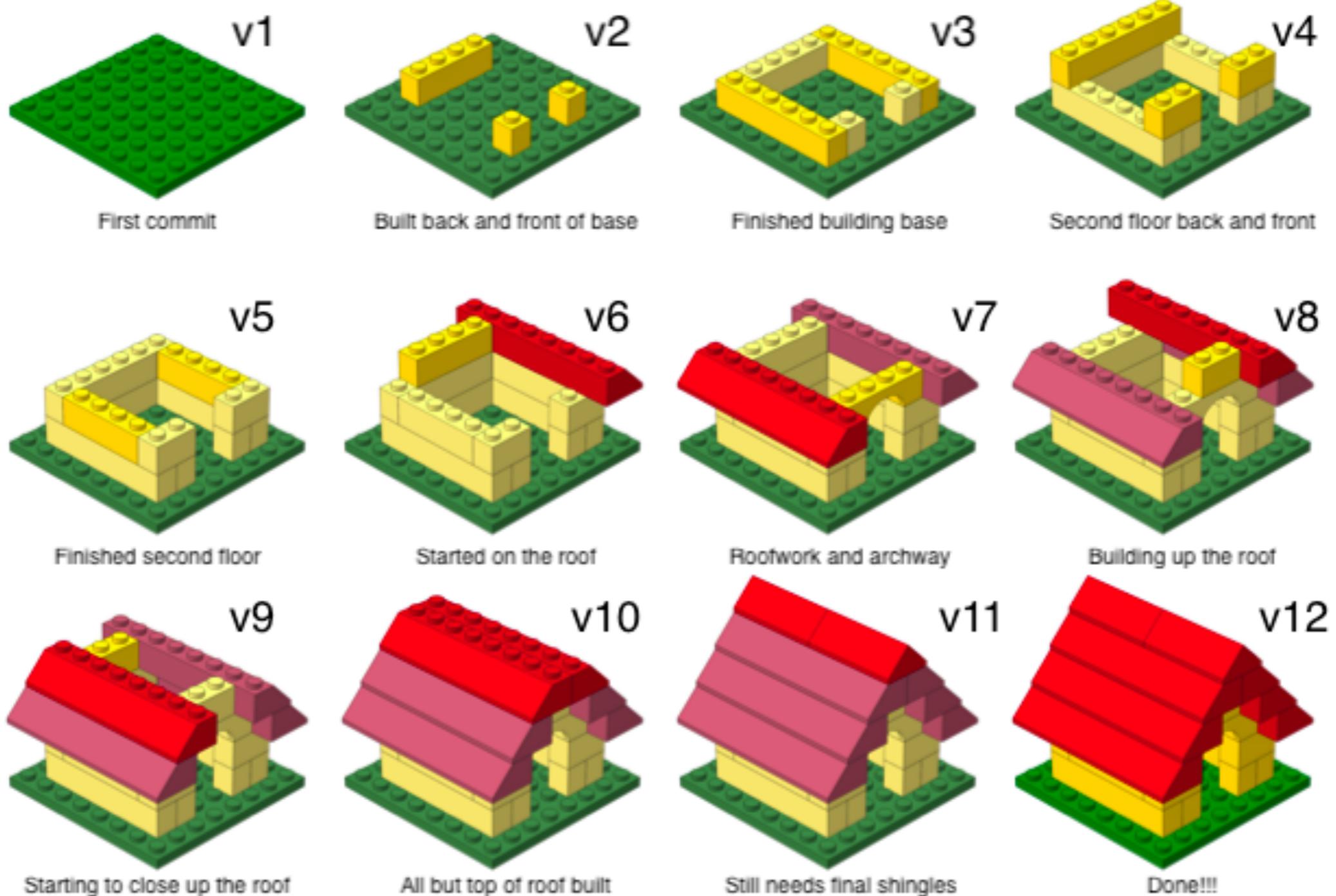
2.



1. Draw some circles

2. Draw the rest of the fucking owl

Writing code is a step-wise process



Resources: Cheat Sheets

Cheat sheets make it easy to learn about and quickly refer to function in some of the common packages.

Data Import :: CHEAT SHEET

R's tidyverse is built around `tidy data` stored in `tibbles`, which are enhanced data frames. The front side of this sheet shows how to read files into R with `readr`. The reverse side shows how to create tibbles with `tibble` and layout tidy data with `tidyR`.

OTHER TYPES OF DATA
Try one of the following packages to import other types of files

- `haven` - SPSS, Stata, and SAS files
- `readxl` - excel files (.xls and .xlsx)
- `DBI` - databases
- `jsonlite` - JSON
- `xml2` - XML
- `httr` - Web APIs
- `rvest` - HTML (Web Scraping)

Save Data
Save x, an R object, to path, a file path, as:

```
Code: write_csv(x, path) #> "NA", append = FALSE, col_names = (append)
      
```

Comma delimited file
`write_csv(x, path, na = "NA", append = FALSE, col_names = (append))`

File with arbitrary delimiter
`write_delim(x, path, delim = " " na = "NA", append = FALSE, col_names = (append))`

CSV for excel
`write_excel(x, path, na = "NA", append = FALSE, col_names = (append))`

String to file
`write_file(x, path, append = FALSE)`

String vector to file, one element per line
`write_lines(x, path, na = "NA", append = FALSE)`

Object to nos
`write_rds(x, path, compress = c("none", "gz", "bz2", "xz", ...))`

Tab delimited files
`write_tsv(x, path, na = "NA", append = FALSE, col_names = (append))`

Read Tabular Data These functions share the common arguments:

```
Code: read(*file, col_names = TRUE, col_types = NULL, locale = default_locale(), na = "", NA, quoted_na = TRUE, comment = "", trim_ws = TRUE, skip = 0, n_max = Inf, guess_max = min(1000, n_max), progress = interactive())
      
```

Comma Delimited Files
`read_csv(*file.csv)`

Semi-colon Delimited Files
`read_csv2(*file.csv)`

Fixed Width Files
`read_fwf(*file.fwf, col_positions = c(1, 3, 5))`

Tab Delimited Files
`read_tsv(*file) #> also read_table(*file)`

USEFUL ARGUMENTS

Example file
`write_csv(a, b, c, n1, 1, 2, 3, n4, 5, "file.csv")`

No header
`read_csv(a, b, c, n1, 1, 2, 3, n4, 5, NA, skip = 1)`

Provide header
`read_csv(a, b, c, n1, 1, 2, 3, n4, 5, NA, header = "n1,n2,n3")`

Missing Values
`read_csv(a, na = c("1", ""))`

Read Non-Tabular Data

Read a file into a single string
`read_file(*file, locale = default_locale())`

Read each line into its own string
`read_lines(*file, skip = 0, n_max = -1L, na = "character", locale = default_locale(), progress = interactive())`

Read Apache log file
`read_log(*file, col_names = FALSE, n_max = 0, n_max = 1, progress = interactive())`

R Studio

Data Transformation with dplyr :: CHEAT SHEET

dplyr functions work with pipes and expect `tidy data`. In tidy data:

Manipulate Cases

EXTRACT CASES
Row functions return a subset of rows as a new table.

pipes
Each variable is in its own column
Each observation, or case, is in its own row
x %>% fly becomes `(x, y)`

Summarise Cases

These apply `summary` functions to columns to create a new table of summary statistics. Summary functions take vectors as input and return one value (see back).

EXTRACT VARIABLES
Column functions return a set of columns as a new vector or table.

Manipulate Variables

EXTRACT VARIABLES
Column functions return a set of columns as a new vector or table.

manipulate data, var = -1 Extract column values as a vector. Choose by name or index, `pull(riris, Sepal.Length)`

select data... Extract columns as a table. Also `select_if`, `select(riris, Sepal.Length, Species)`

Use these helpers with select(...), e.g. `select(riris, starts, with("Sepo"))`

contains(match) num_range(prefix, range) `iris %>% contains("Sepo", 1:3)`

ends_with(match) one...of... `iris %>% ends_with("Sepo")`

matches(match) starts_with(match) `iris %>% matches("Sepo", starts_with("Sepo"))`

MAKE NEW VARIABLES

These apply `vectorised` functions to columns. Vectorised funs take vector as input and return vectors of the same length as output (see back).

vectorized function

Manipulate Cases

filter(data, ...) Extract rows that meet logical criteria. `filter(riris, Sepal.Length > 7)`

distinct(data, ..., keep = FALSE) Remove rows with duplicates. `distinct(iris, Species)`

sample(data, n, size = 1, replace = FALSE) Randomly sample n rows from data. `sample(riris, 5, replace = TRUE)`

count(data, wt = NULL, sort = FALSE) Count number of rows in each group defined by the variables in data. Also `tally`, `count(riris, Species)`

slice(data, ...) Select rows by position. `slice(riris, 10:15)`

top(n, x, wt) Select and order top n entries by group if grouped data. `top(riris, 5, Sepal.Width)`

VARIATIONS

summarise_all - Apply funs to every column. `summarise_all(~ mean(mpg))`

summarise_at - Apply funs to specific columns. `summarise_at(mtcars, v6 ~ mean(mpg))`

summarise_if - Apply funs to all cols of one type. `summarise_if(mtcars, is.numeric, mean)`

Group Cases

Use `group_by()` to create a "grouped" copy of a table. `dplyr` functions will manipulate each "group" separately and then combine the results.

ARANGE CASES

arrange(data, ...) Orders rows by values of a column or columns (low to high), with `desc` to order from high to low. `arrange(mtcars, mpg, desc(mpg))`

group_by(data, ..., add = TRUE) Groups copy of table grouped by. `group_by(iris, Species)`

ungroup(x, ...) Returns ungrouped copy of table. `ungroup(iris)`

ADD CASES

add_row(data, ..., before = NULL, after = NULL) Add one or more rows to a table. `add_row(faithful, eruptions = 1, waiting = 1)`

add_tidy(data, ..., add_col = TRUE) Add new columns. Also `add_count`, `add_row(faithful, eruptions = 1, waiting = 1)`

rename(data, ...) Rename columns. `rename(iris, Length = Sepal.Length)`

R Studio

Data Visualization with ggplot2 :: CHEAT SHEET

Basics

`ggplot` is based on the grammar of graphics, the idea that you can build a graph from the same components: a data set, a coordinate system, and geoms—visual marks that represent data points.

Geoms Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

GRAPHICAL PRIMITIVES

a = `geom_point(aes(x=longitude, y=latitude))`
b = `geom_rect(aes(xmin=longitude, xmax=longitude, ymin=latitude, ymax=latitude))`

TWO VARIABLES

continuous x, continuous y
e = `geom_line(aes(x=longitude, y=latitude))`
b = `geom_bar(aes(x=longitude, y=latitude))`
c = `geom_text(aes(x=longitude, y=latitude))`

CONTINUOUS BIVARIATE DISTRIBUTION

b = `geom_hex(aes(x=longitude, y=latitude))`
h = `geom_density2d(aes(x=longitude, y=latitude))`
g = `geom_hex(aes(x=longitude, y=latitude))`

CONTINUOUS FUNCTION

i = `geom_ribbon(aes(xmin=longitude, xmax=longitude, ymin=latitude, ymax=latitude))`
j = `geom_rect(aes(xmin=longitude, xmax=longitude, ymin=latitude, ymax=latitude))`
k = `geom_smooth(aes(x=longitude, y=latitude))`

DISCRETE X, CONTINUOUS Y

f = `geom_bar(aes(x=longitude, y=latitude))`
g = `geom_text(aes(x=longitude, y=latitude))`
h = `geom_bar(aes(x=longitude, y=latitude))`

DISCRETE SEGMENTS

b = `geom_bar(aes(x=longitude, y=latitude))`
c = `geom_bar(aes(x=longitude, y=latitude))`
d = `geom_bar(aes(x=longitude, y=latitude))`

ONE VARIABLE, CONTINUOUS

c = `geom_area(aes(x=longitude, y=latitude))`
b = `geom_bar(aes(x=longitude, y=latitude))`
d = `geom_bar(aes(x=longitude, y=latitude))`

CONTINUOUS FUNCTION

e = `geom_bar(aes(x=longitude, y=latitude))`
f = `geom_bar(aes(x=longitude, y=latitude))`
g = `geom_bar(aes(x=longitude, y=latitude))`

DISCRETE X, DISCRETE Y

g = `geom_bar(aes(x=longitude, y=latitude))`
h = `geom_bar(aes(x=longitude, y=latitude))`
i = `geom_bar(aes(x=longitude, y=latitude))`

THREE VARIABLES

s = `geom_bar(aes(x=longitude, y=latitude, z=altitude))`
t = `geom_bar(aes(x=longitude, y=latitude, z=altitude))`
u = `geom_bar(aes(x=longitude, y=latitude, z=altitude))`

discrete

d = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
e = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
f = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete x, discrete y

g = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
h = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
i = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete x, continuous y

j = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
k = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
l = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

maps

m = `geom_map(aes(map_id = map_id, fill = color))`
n = `geom_map(aes(map_id = map_id, fill = color))`
o = `geom_map(aes(map_id = map_id, fill = color))`

discrete

p = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
q = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
r = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

s = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
t = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
u = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete x, discrete y

v = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
w = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
x = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete x, continuous y

y = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
z = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
a = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

b = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
c = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
d = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

e = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
f = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
g = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

h = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
i = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
j = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

k = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
l = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
m = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

n = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
o = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
p = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

q = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
r = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
s = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

t = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
u = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
v = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

w = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
x = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
y = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

z = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
a = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
b = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

c = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
d = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
e = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

f = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
g = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
h = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

i = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
j = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
k = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

l = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
m = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
n = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

o = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
p = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
q = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

r = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
s = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
t = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

u = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
v = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
w = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

x = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
y = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
z = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

a = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
b = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
c = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

d = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
e = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
f = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

g = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
h = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
i = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

j = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
k = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
l = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

m = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
n = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
o = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

p = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
q = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
r = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

s = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
t = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
u = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

v = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
w = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
x = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

y = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
z = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
a = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

b = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
c = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
d = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

e = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
f = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
g = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

h = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
i = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
j = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

k = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
l = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
m = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

n = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
o = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
p = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

q = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
r = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
s = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

t = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
u = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
v = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

w = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
x = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
y = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

z = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
a = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
b = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

c = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
d = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
e = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

f = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
g = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
h = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

i = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
j = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
k = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

l = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
m = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
n = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

o = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
p = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
q = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

r = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
s = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
t = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

u = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
v = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
w = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

x = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
y = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
z = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

a = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
b = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
c = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

d = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
e = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
f = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

g = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
h = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
i = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

discrete

j = `geom_bar(aes(x=longitude, y=latitude, fill=category))`
k = `geom_bar(aes(x=longitude, y=latitude, fill=category))`

Resources: Stackoverflow

stack overflow Products Customers Use cases Log in Sign up

Home PUBLIC Stack Overflow Tags Users Jobs TEAMS What's this? Join Private Q&A

Search Results

Results for removing legend title ggplot

removing legend title ggplot

20 results

Q: remove legend title in ggplot
99 votes 3 answers
I'm trying to **remove the title of a legend** in ggplot2: `df <- data frame(g = rep(letters[1:2], 5), x = rnorm(10), y = rnorm(10)) library(ggplot2) ggplot(df, aes(x, y, colour=g ...`
asked Feb 8 '13 by [smiling](#)

Q: How to change the position of plotly figures using ggplotly
0 votes 1 answer
I am trying to **remove the legend title** using ggplotly without success. I'm sure there is an easy fix, but I cannot find the documentation for it - and **removing the legend title** (or changing the ...) + `geom_boxplot() + theme(legend.title=element_blank())` a # No Legend Title # plotly puts back the **legend title** ggplotly(a) Any ideas how to change / **remove the title** of the graph? Should it be done using ggplotly or ggplot? ...
asked Dec 26 '15 by [Nick](#)

Q: Remove legend in ggplot in Python
1 vote 0 answers
The **title** says it all: Is there any way to **remove the legend** in ggplot in Python? I tried to google but could only find the solution for R ...
asked Jul 25 '15 by [Giang Do](#)

Q: ggplot legend / label change with various guides
0 votes
already used to **remove** the titles and also **remove legend** for a 3rd aes, but I do not manage to do it. Do you have a solution? Here is my code : `p <- ggplot(data, aes(x = Nb)) p + geom_ribbon(aes(ymin ...` I am trying to

Siemens Siemens AG 9 Singapore Electronics Public 10k+ people Senior Data Scientist machine-learning sql **HTML5 Code King** html5 css3 **Business Architect** enterprise

Hot Network Questions

Resources

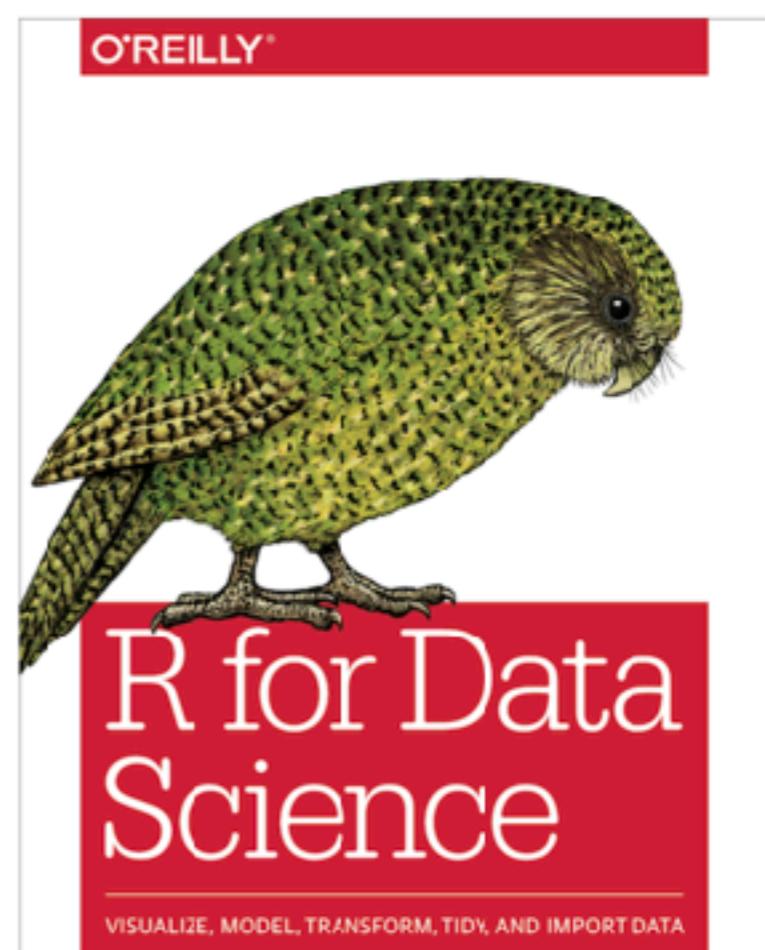
R for Data Science

Garrett Grolemund

Hadley Wickham

Welcome

This is the website for “**R for Data Science**”. This book will teach you how to do data science with R: You’ll learn how to get your data into R, get it into the most useful structure, transform it, visualise it and model it. In this book, you will find a practicum of skills for data science. Just as a chemist learns how to clean test tubes and stock a lab, you’ll learn how to clean data and draw plots – and many other things besides. These are the skills that allow data science to happen, and here you will find the best practices for doing each of these things with R. You’ll learn how to use the grammar of graphics, literate programming, and reproducible research to save time. You’ll also learn how to manage cognitive resources to facilitate discoveries when wrangling, visualising, and exploring data.



Resources: Stackoverflow

stack overflow Products Customers Use cases Log in Sign up

Home PUBLIC Stack Overflow Tags Users Jobs TEAMS What's this? Join Private Q&A

Search Results

Results for removing legend title ggplot

removing legend title ggplot

20 results

Q: remove legend title in ggplot
99 votes 3 answers
I'm trying to **remove the title of a legend** in ggplot2: `df <- data frame(g = rep(letters[1:2], 5), x = rnorm(10), y = rnorm(10)) library(ggplot2) ggplot(df, aes(x, y, colour=g ...`
asked Feb 8 '13 by [smiling](#)

Q: How to change the position of plotly figures using ggplotly
0 votes 1 answer
I am trying to **remove the legend title** using ggplotly without success. I'm sure there is an easy fix, but I cannot find the documentation for it - and **removing the legend title** (or changing the ...) + `geom_boxplot() + theme(legend.title=element_blank())` a # No Legend Title # plotly puts back the **legend title** ggplotly(a) Any ideas how to change / **remove the title** of the graph? Should it be done using ggplotly or ggplot? ...
asked Dec 26 '15 by [Nick](#)

Q: Remove legend in ggplot in Python
1 vote 0 answers
The **title** says it all: Is there any way to **remove the legend** in ggplot in Python? I tried to google but could only find the solution for R ...
asked Jul 25 '15 by [Giang Do](#)

Q: ggplot legend / label change with various guides
0 votes
already used to **remove** the titles and also **remove legend** for a 3rd aes, but I do not manage to do it. Do you have a solution? Here is my code : `p <- ggplot(data, aes(x = Nb)) p + geom_ribbon(aes(ymin ...` I am trying to

Siemens Siemens AG 9 Singapore Electronics Public 10k+ people
Senior Data Scientist machine-learning sql
HTML5 Code King html5 css3
Business Architect enterprise

[View all 14 job openings!](#)

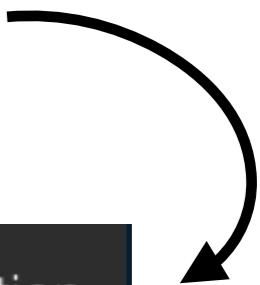
Hot Network Questions

Resources: Help file documentation

Type in the console >

```
>  
>  
> ?mean  
> |
```

Or navigate to the Help tab and search there, to see this



mean {base} R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)
```

Resources: Help file documentation

The name of the function, and the library it is in.

mean [base]

R Documentation

Arithmetic Mean

What it does.

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)  
## Default S3 method:  
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

- > An R object. Currently there are methods for numeric/logical vectors and [date](#), [date-time](#) and [time interval](#) objects. Complex vectors are allowed for `trim = 0`, only.
- `trim` the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of `trim` outside that range are taken as the nearest endpoint.
- `na.rm` a logical value indicating whether `NA` values should be stripped before the computation proceeds.
- ... further arguments passed to or from other methods.

The function's name, and in the parentheses the named arguments it expects, in the order it expects them. If an argument has a default value, it is shown. Arguments without default values (e.g. `x`) must be provided by you.

More details on each named argument. This will tell you what class of thing each argument has to be—an object, a number, a data frame, a logical value, etc.

What the function returns—i.e., the result of whatever operation or calculation it performs. This can be a single number, as here, or a multi-part object such as a list, a data frame, a plot, or a model.

Value

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including `integer`) or complex, `NA_real_` is returned, with a warning.
If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

[weighted.mean](#), [mean](#), [POSIXct](#), [colMeans](#) for row and column means

Examples

```
x <- c(0:10, 50)  
xm <- mean(x)  
c(xm, mean(x, trim = 0.10))
```

The ellipsis allows other arguments to be passed to and from the function.

Other related functions

Self-contained examples that you can run at the console. These may use built-in datasets or other R functions.

[Package base version 3.4.3 [Index](#)]

Visit the package's Index page to look for Demos and Vignettes detailing how it works.

Help files

Go to helpfile_vignette in the tidybiology-2019 workspace

Open students.Rmd, run all code, and View (students) to see your assigned vignette and associated function.

Open pivot_longer.Rmd, and knit the code; be inspired.

Open helpfile_template.Rmd, and see the pre-populated code template for you to work on your own function

Each student will generate a 'help vignette' as part of the final project.