Data Analytics
CS301
Chapter 2,
Intro to R, Workflows

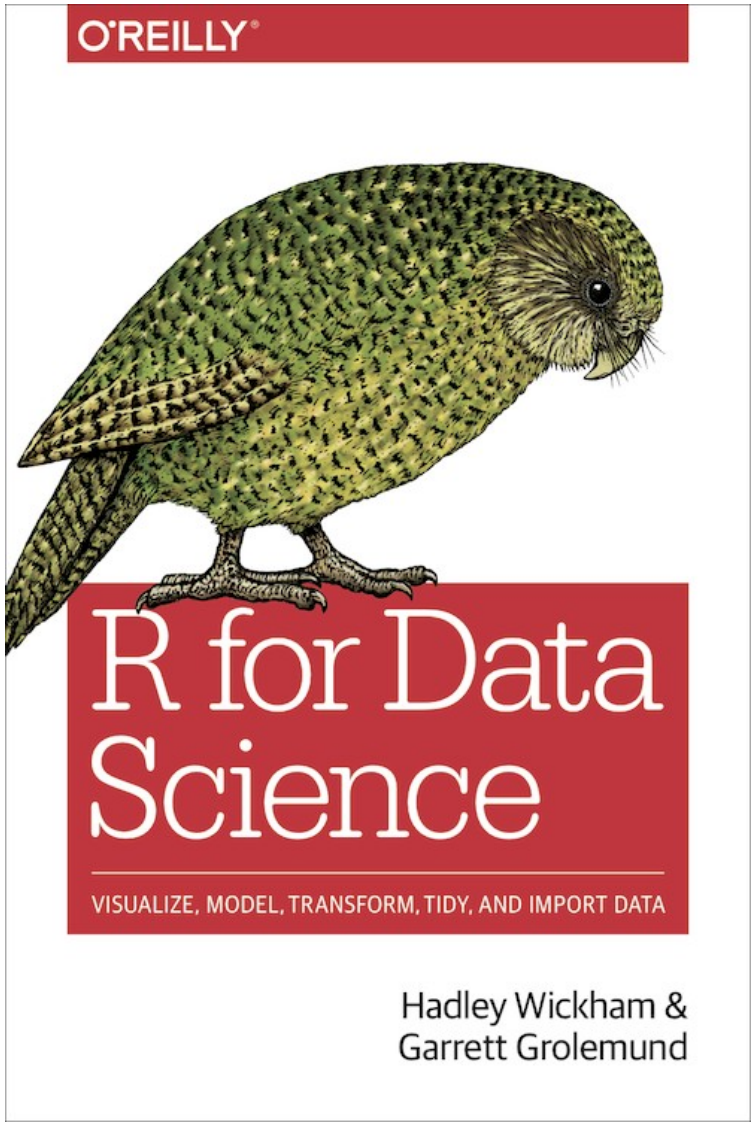
Week 3: 15th Sept
Fall 2020
Oliver BONHAM-CARTER

Where To Now?

- Google Analytics is a tool allowing for convenient analysis of web sites
- The code was written by developers for this purpose.
- What if you need tools and there are no current developers to create them?

Develop Your Own Tools!!

We will be using the Book

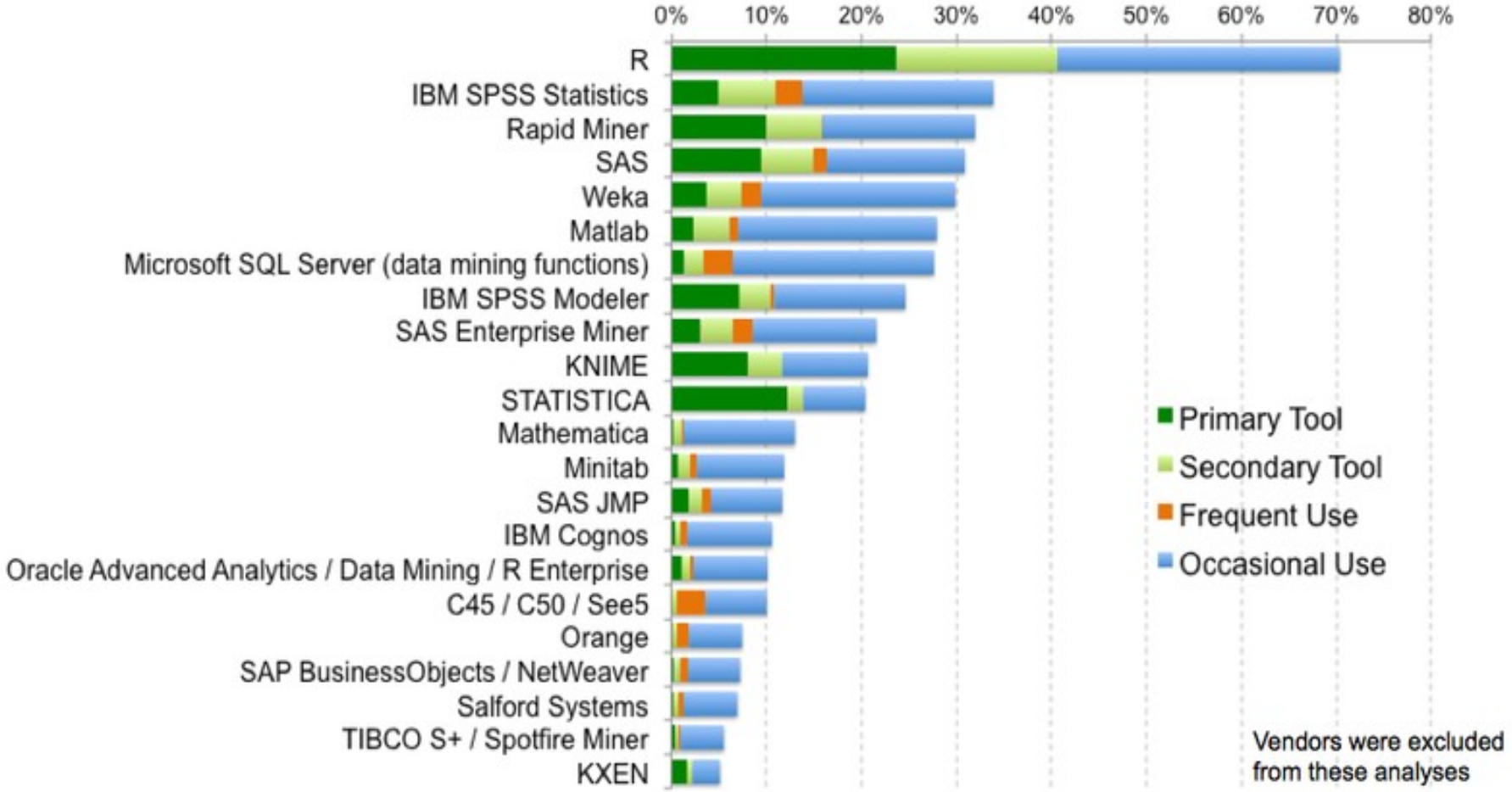


- Note the chapters between the book and the website are not numbered identically!
- Book:
  - Chap 1: Data Visualization with ggplot
  - **Chap 2: Workflow; Basics**
- On the web site:
  - http://r4ds.had.co.nz/
  - Chap 3: Data Visualization
  - **Chap 4: Workflow; Basics**

The R Programming Language

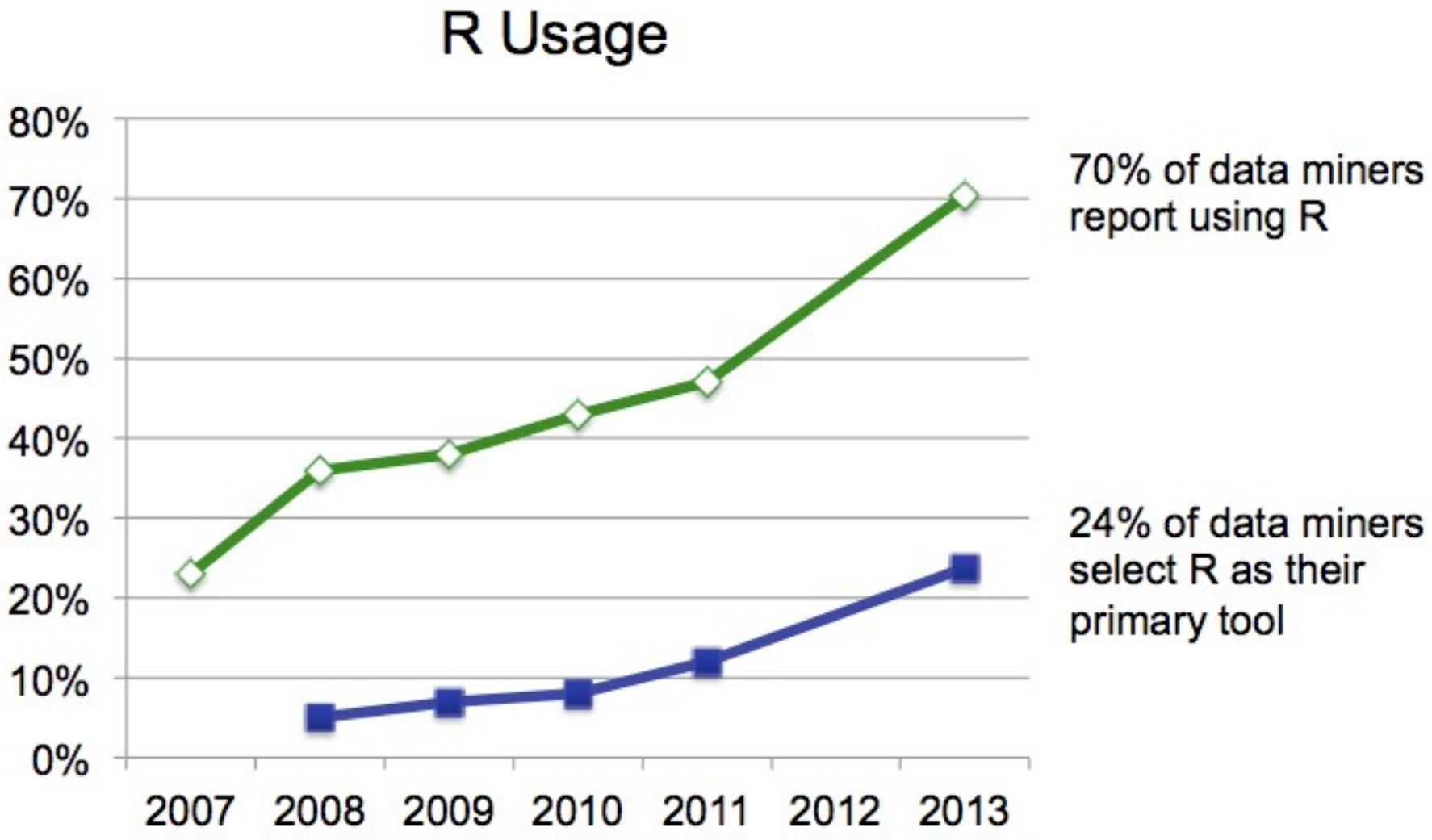
- https://www.r-project.org/
- What is the R language?
  - An open source, well-developed programming platform for work in statistics, mathematics and data analytics
  - Cross platform; runs on major OSs
  - Popular programming skill among Big Data analysts, and data scientists
- Community Blogs:
  - https://www.r-bloggers.com/
  - https://twitter.com/rstudiotips/
  - https://towardsdatascience.com/

R: The Most Popular Data Mining Tool



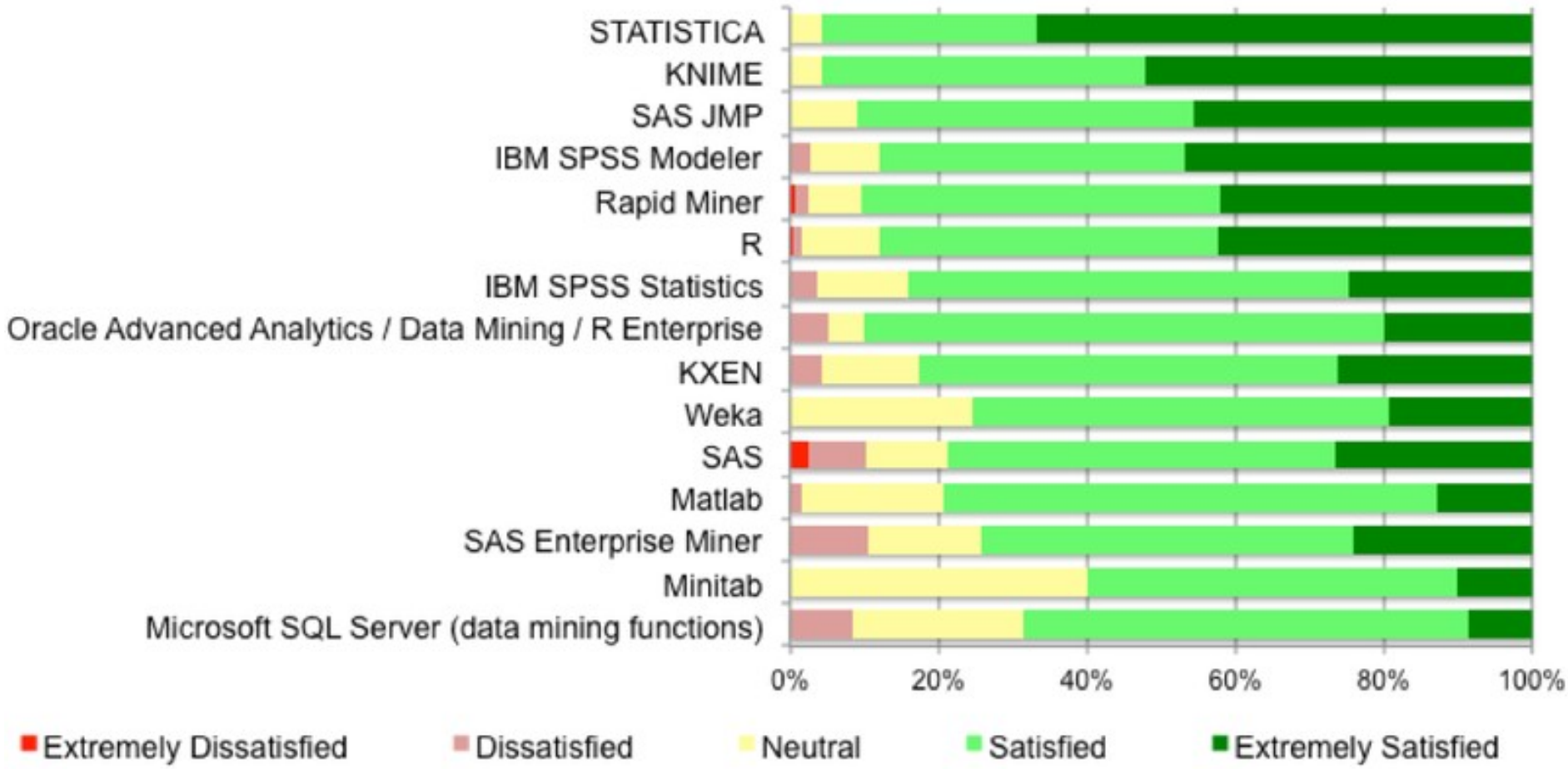
http://blog.revolutionanalytics.com/2013/10/r-usage-skyrocketing-rexer-poll.html

R is Exploding in Growth



http://blog.revolutionanalytics.com/2013/10/r-usage-skyrocketing-rexer-poll.html

Most users are satisfied with R



http://blog.revolutionanalytics.com/2013/10/r-usage-skyrocketing-rexer-poll.html

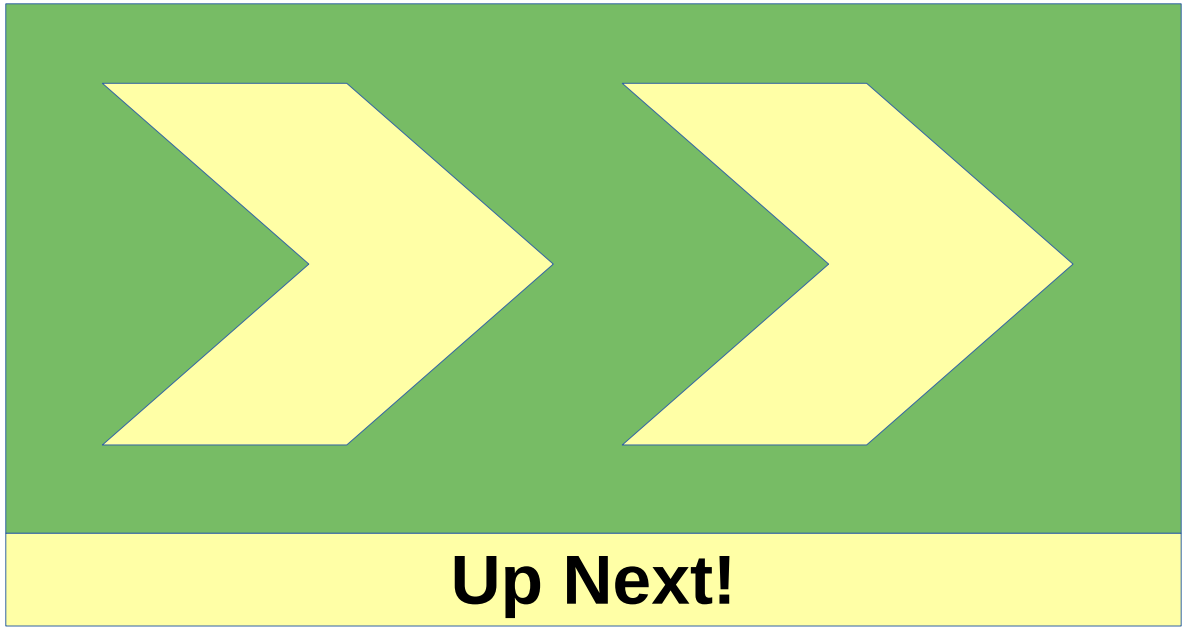
Ranking To Others: IEEE 2017

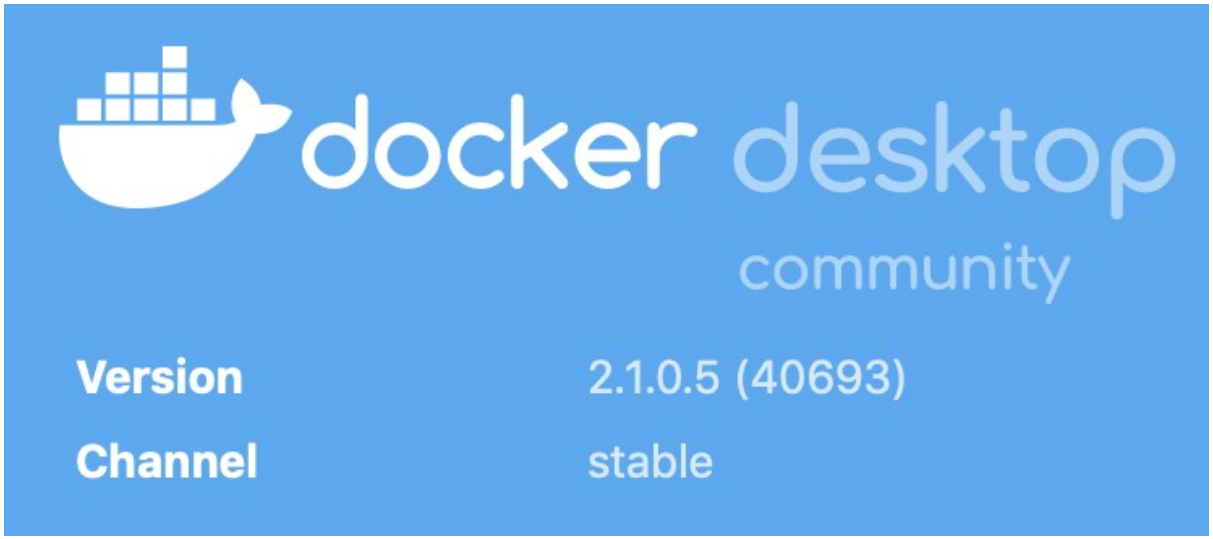
Language Rank	Types	Spectrum Ranking
1. Python		100.0
2. C		99.7
3. Java		99.4
4. C++		97.2
5. C#		88.6
6. R		88.1
7. JavaScript		85.5
8. PHP		81.4
9. Go		76.1
10. Swift		75.3

Find more amazing studies about R:

http://blog.revolutionanalytics.com/2018/06/pypl-programming-language-trends.html

Let's Try Some Code!





Docker Container Setup: R Programming at bash

Note: the directory where you run this becomes your local directory in the container.

```
R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (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.
```

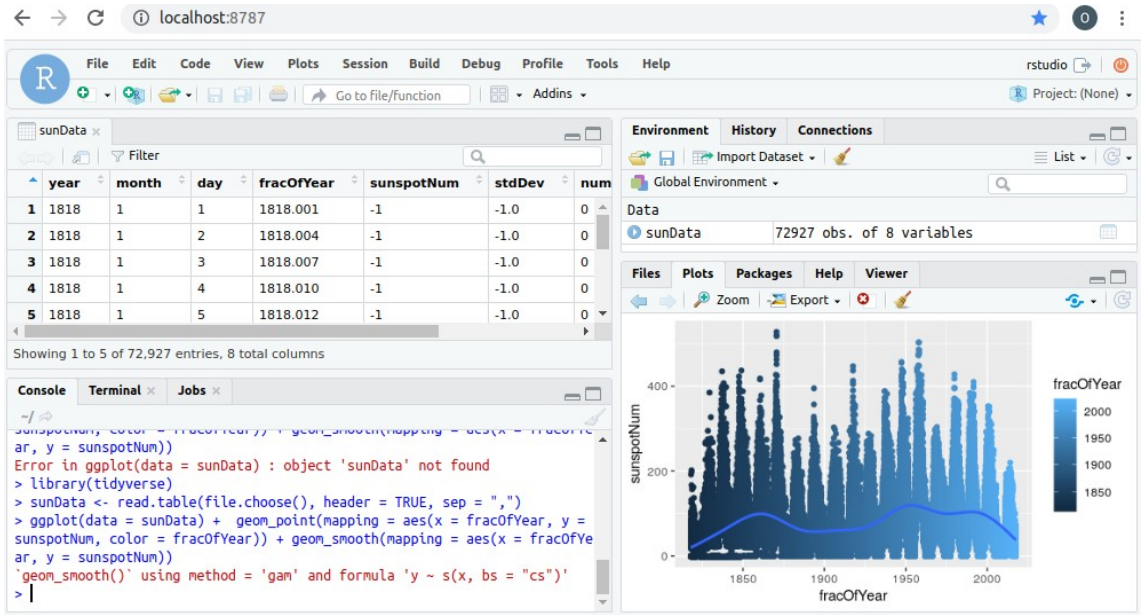
- Build /run container:
  - docker run -ti --rm r-base

- Build, mount local drive and run container :
  - docker run -ti --rm -v "$PWD":/home/docker -w /home/docker -u docker r-base

Docker Container Setup: rStudio

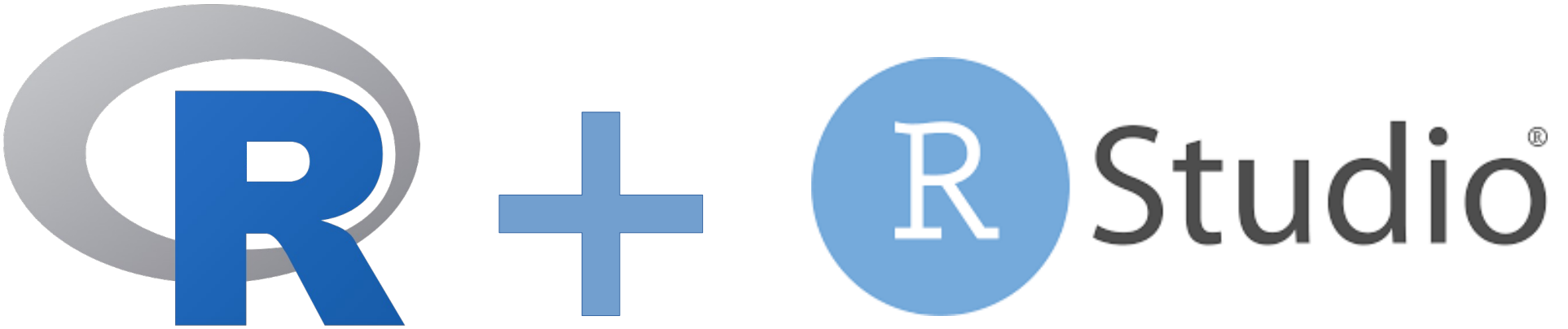
Note: the directory where you run this becomes your local directory in the container.

Username: *rstudio*
Password: *letmein*



- Build, mount local drive and run container:
  - sudo docker run --rm -e **PASSWORD=letmein** -p 8787:8787 -v $PWD:/home/rstudio/ rocker/verse
- Browser:
  - URL: Use Browser address: http://localhost:8787/

A Local Install of rStudio



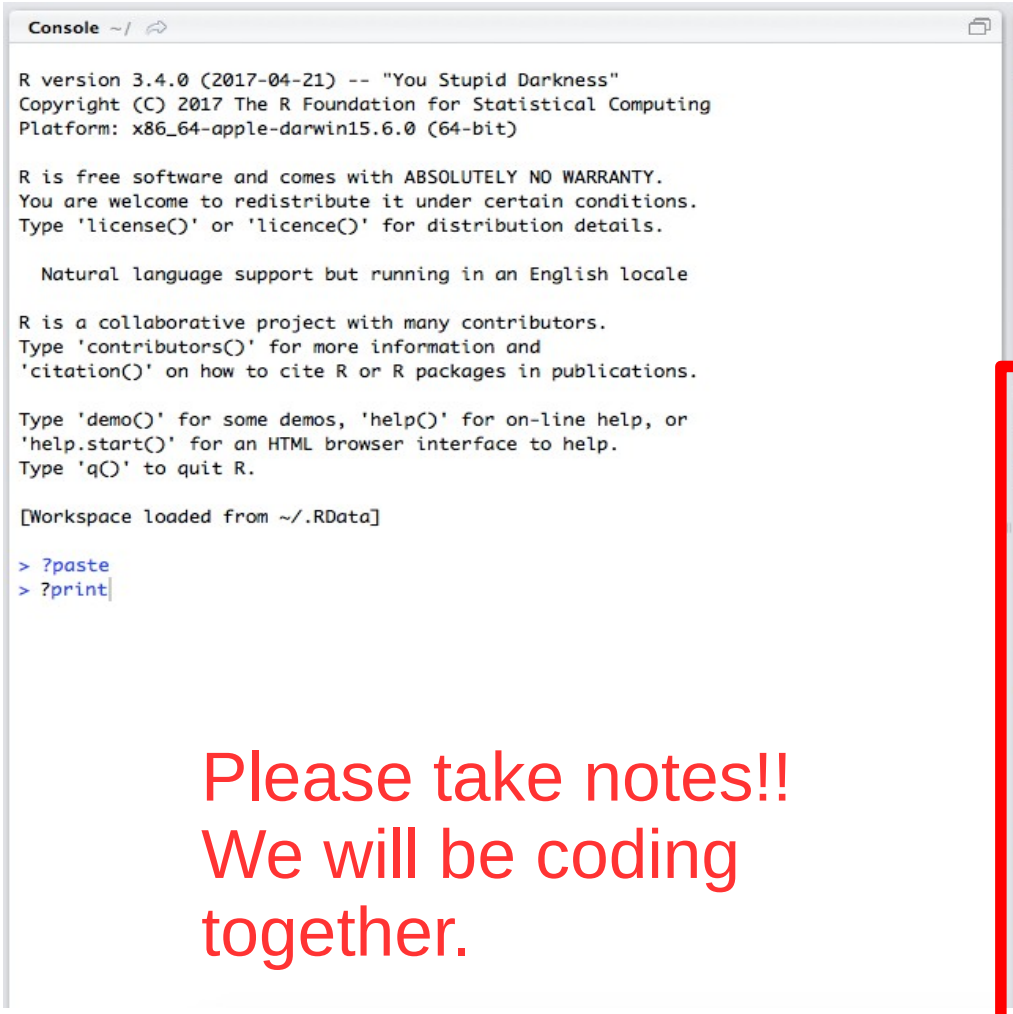
- **You must first install R and then rStudio**
  - The R programming language
    - https://cran.rstudio.com/
  - Rstudio
    - https://rstudio.com/products/rstudio/download/

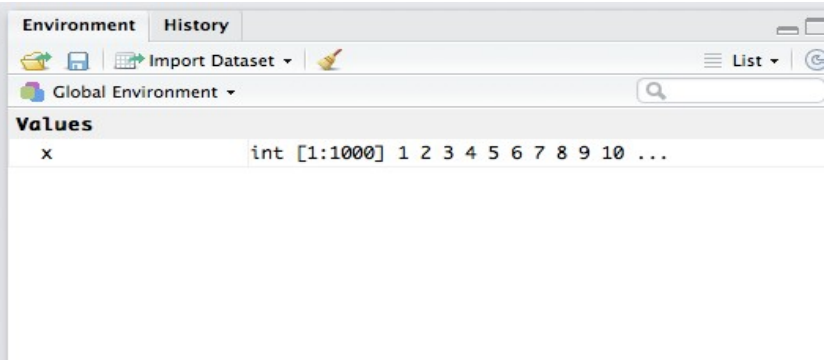
Failing that: R by Jdoodle

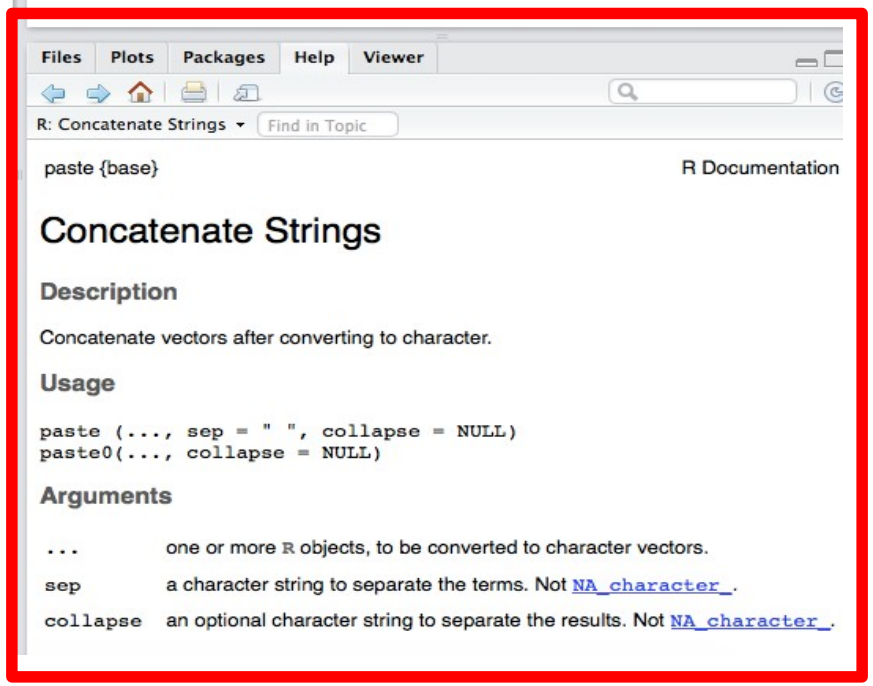
- https://www.jdoodle.com/execute-r-online

Getting Help in R

- Online help: place a “?” in front of a keyword
  - Ex: ?print



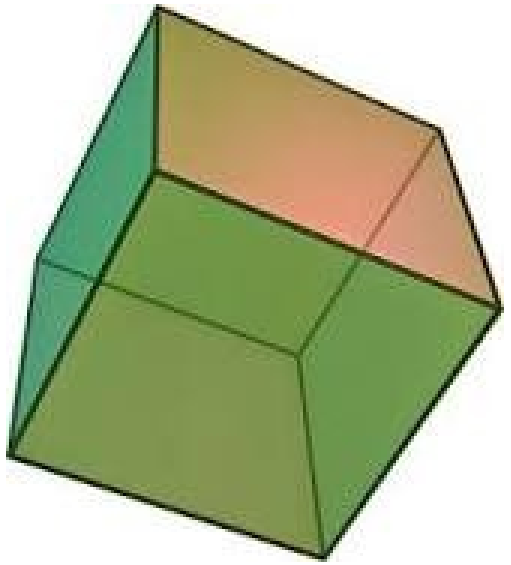




Please take notes!!
We will be coding together.

Variable Names

- Variable Names:
  - Begin with a letter, and can only include letters, numbers, periods and hyphens.
  - Hyphens: “-”
  - Periods: “.”
- SnakeCase (recommended by book)
  - val_of_height,
  - val_of_length,
  - val_of_width



Basic Math

- Mathematics
  - Addition: 1 + 1
  - Subtraction: 1 - 1
  - Multiplication: 3 * 7
  - Division: 1 / 4
- More complicated math, var assignments:
  - 4*(7+3)/10+1 Note: watch the order of operations!
  - Parameter of circle ($C = 2 * pi * r$)
    - *R* <- 4, Note the "<-" means *equal* in R.
    - *C* <- 2 * pi * R = 2 * 3.1415 * 4
    - *C* is 25.13274

Variable Names

- CamalCase:
  - valOfHeight,
  - valOfLength,
  - valOfWidth
- Period.Case
  - Val.of.height,
  - Val.of.length,
  - Val.of.width
- What-EVER.Case
  - Val.ofHEIGHT,
  - Val.Of_Length,
  - Val.oF.Width

Assigning Variables

- Assign a variable
  - *x* = 1, or
  - *x* <- 1
  - *y* = 3
  - *y* <- 3
  - Run:

    x + y

```
> x <- 1
> y <- 3
> x + y
[1] 4
```

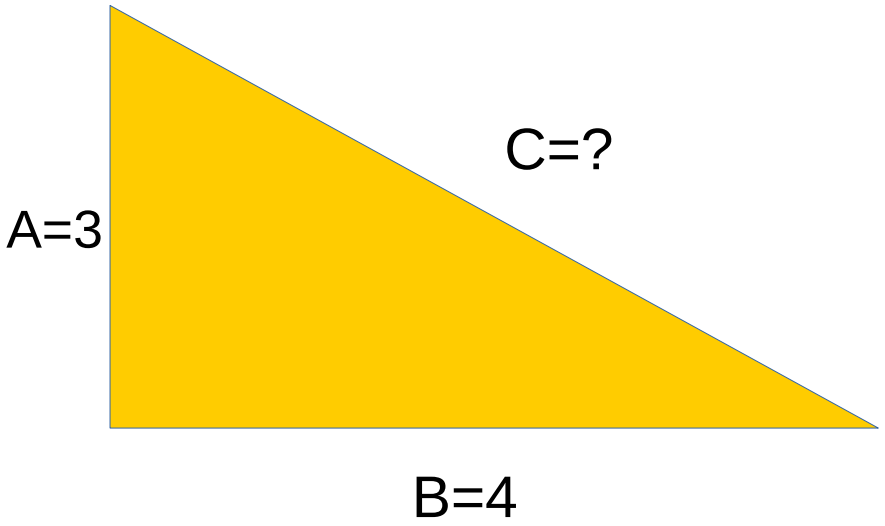
  - *myNum <- -2*
  - *myOtherNum <- -4*
  - Run:

    myNum + myOtherNum

```
> myNum <- -2
> myOtherNum <- -4
> myNum + myOtherNum
[1] -6
```

Variables and Assignments

- X <- 10.
- You could also use “X=10” but this is not traditional programming in R...
- Hypotenuse = c = sqrt(a^2 + b^2)
- A <- 3
- B <- 4
- C <- sqrt(A^2 + B^2)
- C is ??



Logical Operations

- Booleans: Returning True or False:

  3 > 4, 3 < 4,

  2 + 4 == 6,

  2 + 3 == 4 + 1

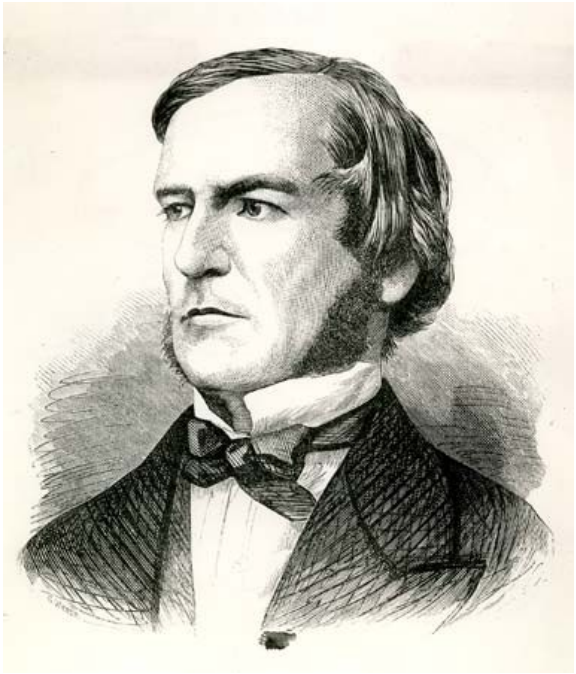
  T == TRUE

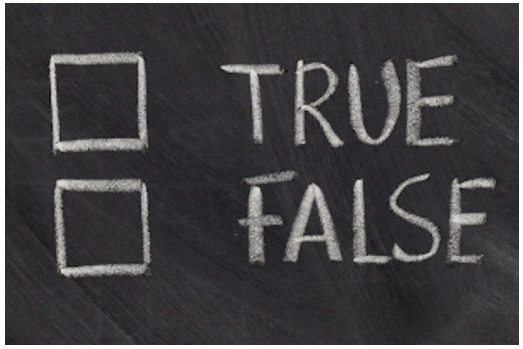
  F == FALSE

  3 + 4 != 5

  3 + 4 == 7

  5 * 2 != 11





Try some of These in R!

- Logical **AND**
- (&&)

  F && F: F

  F && T: F

  T && F: F

  T && T: T

- Logical **OR**
- (||)

  F || F: F
  F || T: T
  T || F: T
  T || T: T

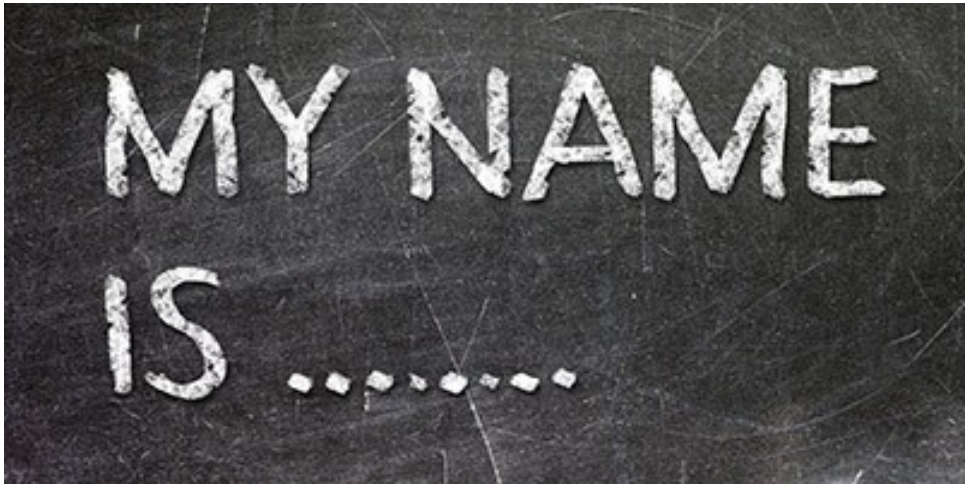
- Logical **NOT**
- (!)

  !F: T
  !T: F

Truth Tables:
https://en.wikipedia.org/wiki/Truth_table
De Morgan's Laws:
https://en.wikipedia.org/wiki/De_Morgan%27s_laws

Simple Strings

- Strings
  - “Hello World”
- Concatenation of strings
  - H <- “Hello”
  - W <- “world”
  - paste(H,W, sep = “ ”)
    - What is the result here??



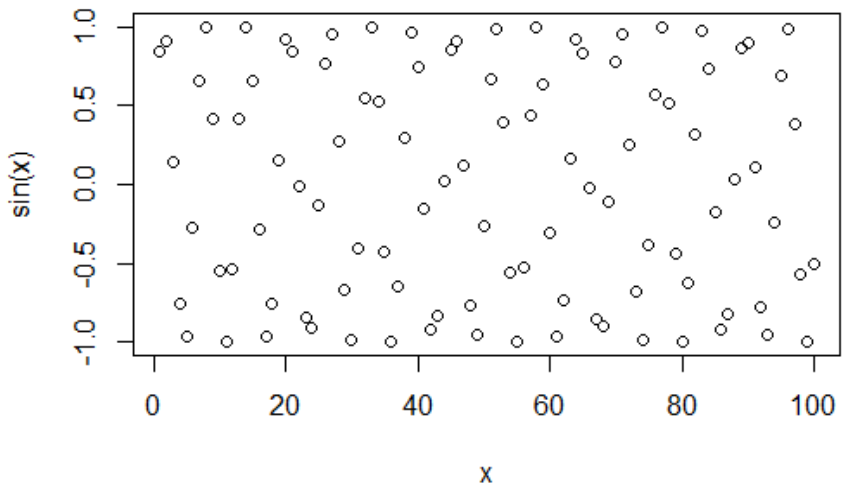
- You try: print your full name!
  - *first* <- Sherlock
  - *last* <- Holmes
  - paste(*first*,*last*, sep =" ")

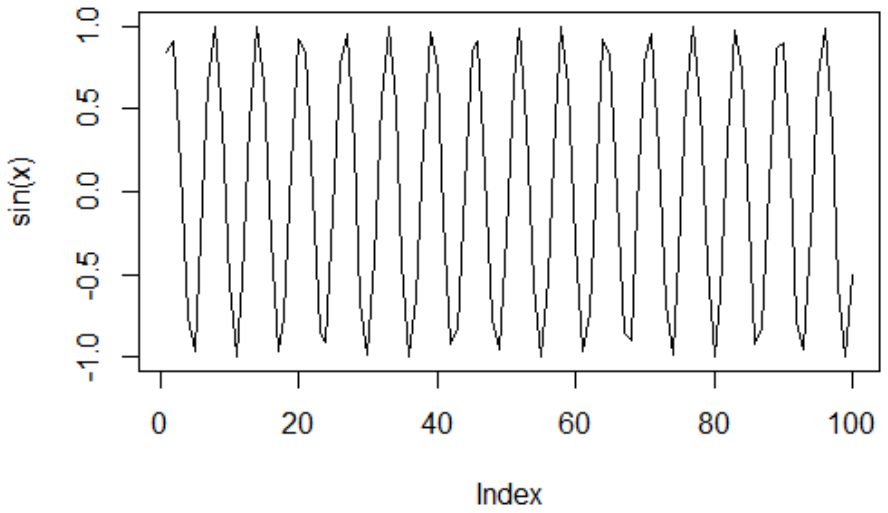
Built-in Functions

- R has a large collection of built-in functions:
  - function_name(arg1 = val1, arg2 = val2, …)
- Try calling this function:
  - seq(0,10)
  - Gives a sequence, *S* ={0,…,10}
  - What happens when you press TAB after typing, “seq”?
- Use the sum() function to add two numbers.
- sum() to add three numbers?
- sum() to add a whole lot of numbers?

Simple Plots

- x<- seq(1,100) # assign *x* to the sequence 1 to 100
- plot(x) # plot this sequence
- plot(sin(x)) or plot(x,sin(x)) # see left plot below
- plot(sin(x)) or plot(x,sin(x), type = “l”) # see right plot below





Now, You Try

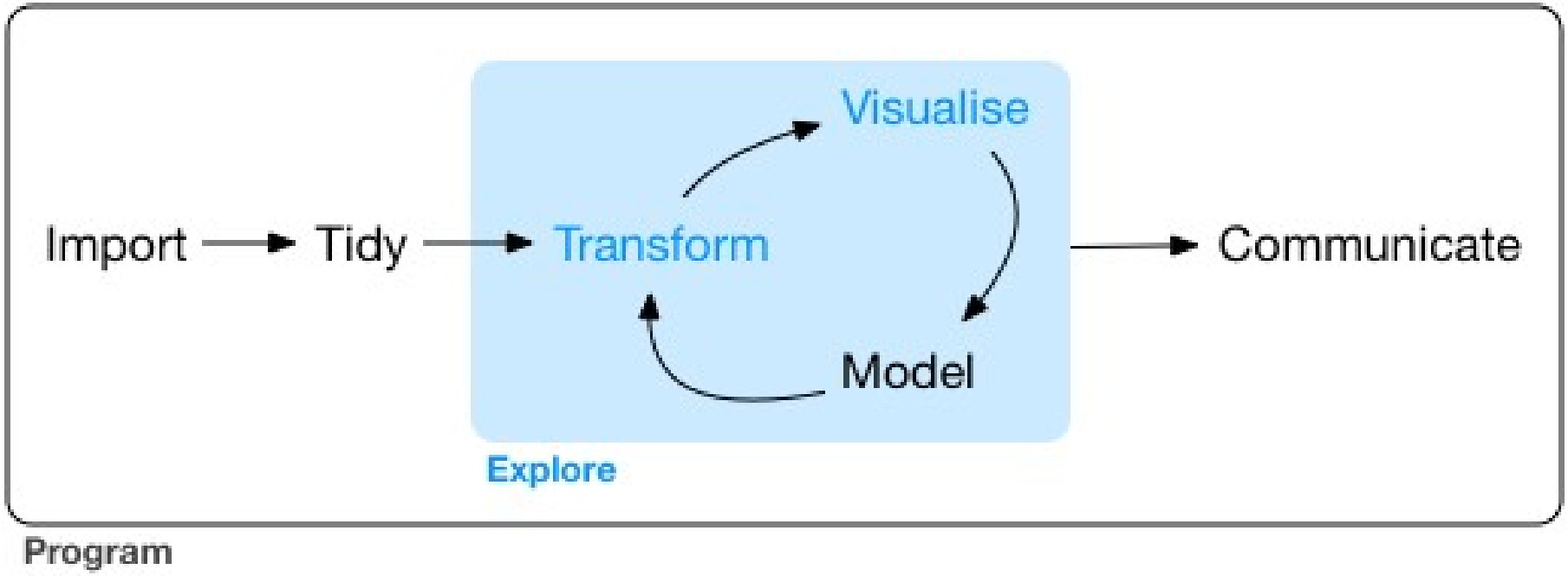
- Use R to write a command that...
  - Finds the **sum** of all numbers, 0 through 100
  - Finds the **sum** of all numbers, 0 through 10000

    (now, set a variable equal to the sequence first)
  - Use the plot function, **plot(x,y,type = "l")** to plot a line of the function, f(*x*) = sin(*x*) for *x* in {0, …, 30}
  - Plots the function, f(*x*) = cos(x) for *x* in {0, …, 30}
  - Plots the function, f(*x*) = tan(x) for *x* in {0, …, 30}

Exiting R:
q()

Explore the Data Of Your World

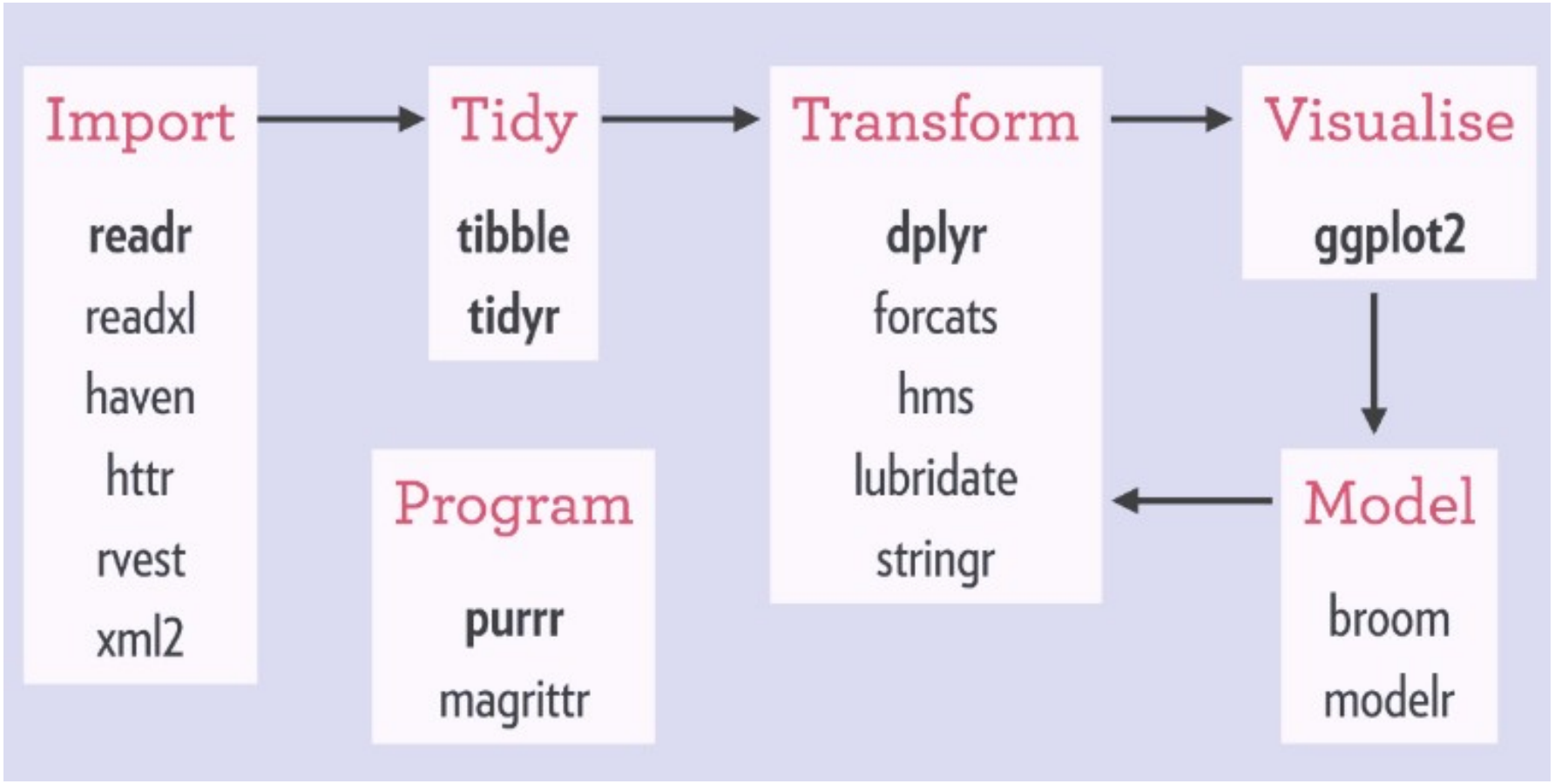
"Data exploration is the art of looking at your data, rapidly generating hypotheses, testing them, then repeating again and again..."



Import : Bringing in the raw data to work on it
Tidy: Cleaning it up so that numbers are numbers and etc.
Transform: Converting the data into something more *convenient* to use
Visualize: Finding general trends in data
Model: Testing phases, learning how to predict from the data.
Communicate: Publish and change the world!

Tidyverse's Packages

The steps of the Tidyverse canonical data science workflow, as well as, the individual packages that the steps involve.



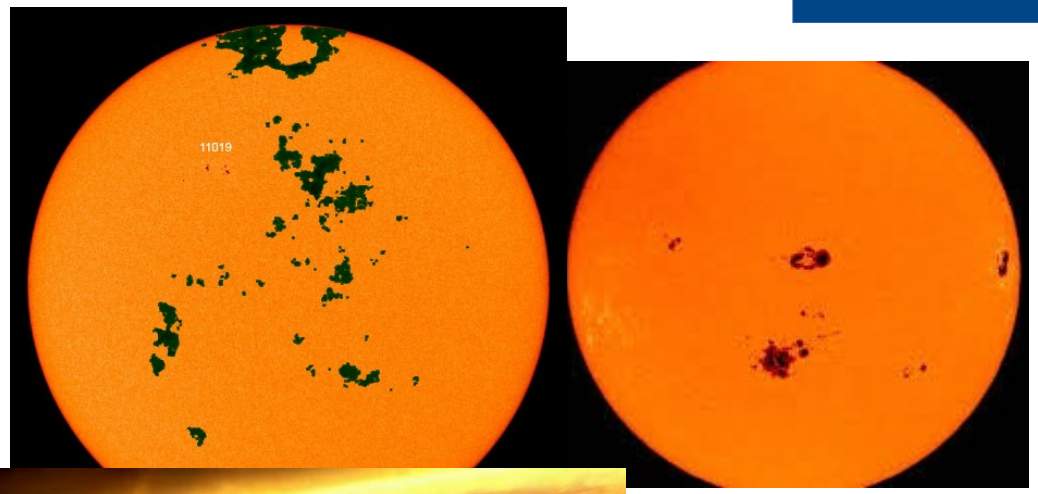
Data and Plotting

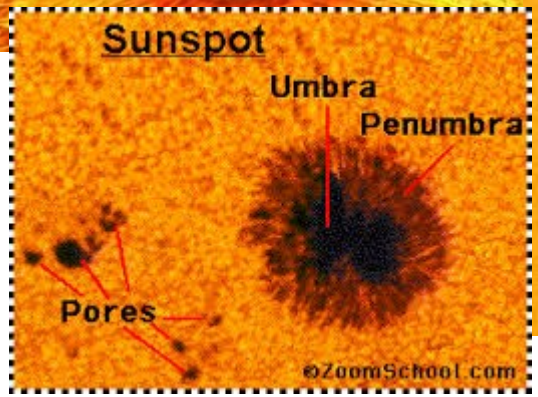
The Tidyverse library in R: a coherent system
of packages for data manipulation, exploration and visualization

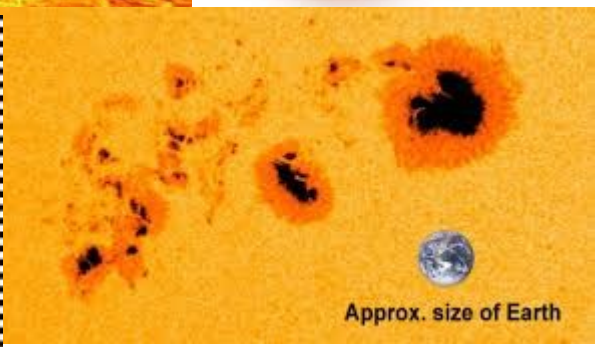
- *library(tidyverse)*
  - If you need to install it: *Install.packages(tidyverse)*

Exploring Sun-Spot Data

- Sunspots – magnetic disturbances on the sun that can be observed from Earth
- Spots cycles are noted to repeatedly increase and then decrease over time







Articulating the Research Question

- Is there something predictable about the sunspot data?
- Can we collect come evidence of a pattern in the data?
- Could we use this pattern to predict?
- What does a pattern look like in the data?

Load and Plot Sunspot Data

```
#Load library
library(tidyverse)

# find your sandbox file
sunData <- read.table(file.choose(), header =
TRUE, sep = ",")

# See what the data looks like
View(sunData)
```

```
# Plot the data:
ggplot(data = sunData) +  geom_point(mapping = aes(x =
fracOfYear, y = sunspotNum))

# Save a file to the Desktop/ (or wherever) if you
want...
ggsave("~/Desktop/myplot.png")
```

Code for a Simple GGPlot

- **library(tidyverse) or if not present,**
- **install.packages("tidyverse")**
- **ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy))**

- Establish the *canvas* (where the plot is shown)
- **Ggplot()**

- Link to the data (set is called, 'mpg')
  - **ggplot(data = mpg)**

- Compute the geometery of point placement on canvas
  - **geom_point(mapping = …)**

- Compute the aesthetics of the plot (titles, color, point type, etc)
  - **aes(x = displ, y = hwy)**

Consider this ...

```
names(sunData)

ggplot(data = sunData) +  geom_point(mapping = aes(x = fracOfYear, y =
sunspotNum))

# Add a smooth line to see general trends
ggplot(data = sunData) +  geom_point(mapping = aes(x = fracOfYear, y =
sunspotNum)) + geom_smooth(mapping = aes(x = fracOfYear, y =
sunspotNum))

# Color by year
ggplot(data = sunData) +  geom_point(mapping = aes(x = fracOfYear, y =
sunspotNum, color = fracOfYear)) + geom_smooth(mapping = aes(x =
fracOfYear, y = sunspotNum))

# Color by month
ggplot(data = sunData) +  geom_point(mapping = aes(x = fracOfYear, y =
sunspotNum, color = month)) + geom_smooth(mapping = aes(x = fracOfYear,
y = sunspotNum, color = fracOfYear))
```

Run this code
to make other plots.
What do you see?

file: sandbox/sunspots.r