An introduction to #rstats through analysis

import > viz > prep > test

Section 1

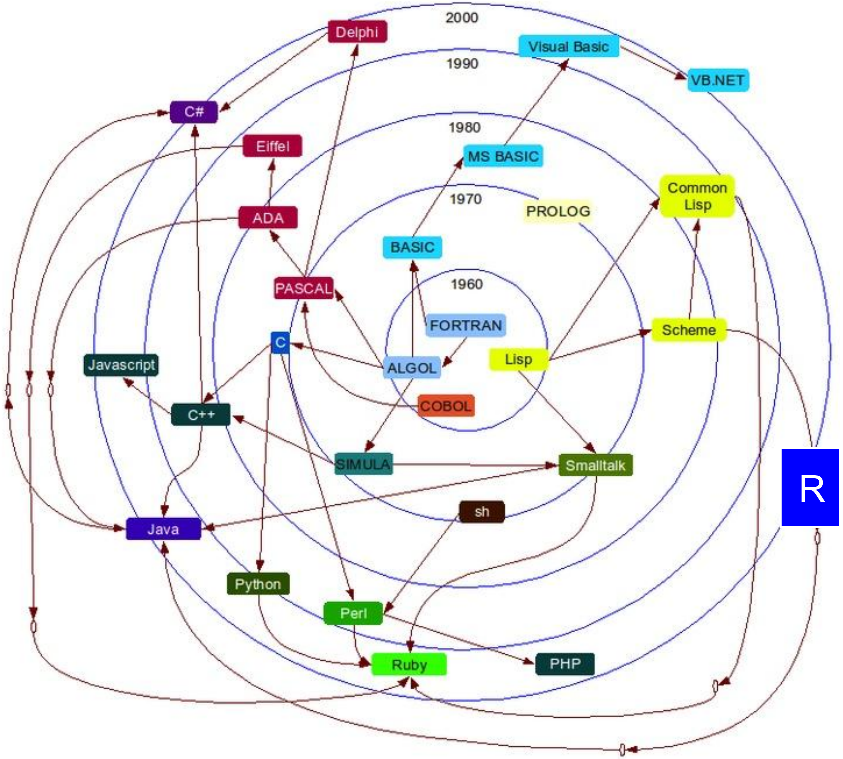
- Computer language overview
- Building blocks of R
  - Classes/Types
  - Operators/Functions
  - Vectors/Lists/Data-frames

Why program?

- Powerful way to work
  - Recordable/reproducible
  - Automation
- You already know some programming languages
  - Math
    - +, -, *, ^, log
  - Excel formulas
    - =SUM(A1:A12)

Language Genealogy

- All attempt to describe logic
  - Providing exact instructions is hard
    - Think about the complexity of cooking a meal.
- Each is focused on solving a different problem
  - Popularity varies
  - Application varies
  - Style varies



https://github.com/stereobooster/programming-languages-genealogical-tree/blob/gh-pages/img/radial.jpg

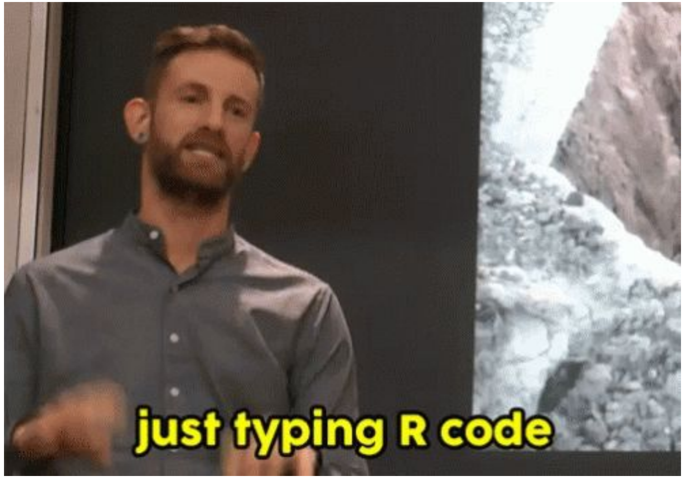
Why R?

https://en.wikipedia.org/wiki/R_(programming_language)

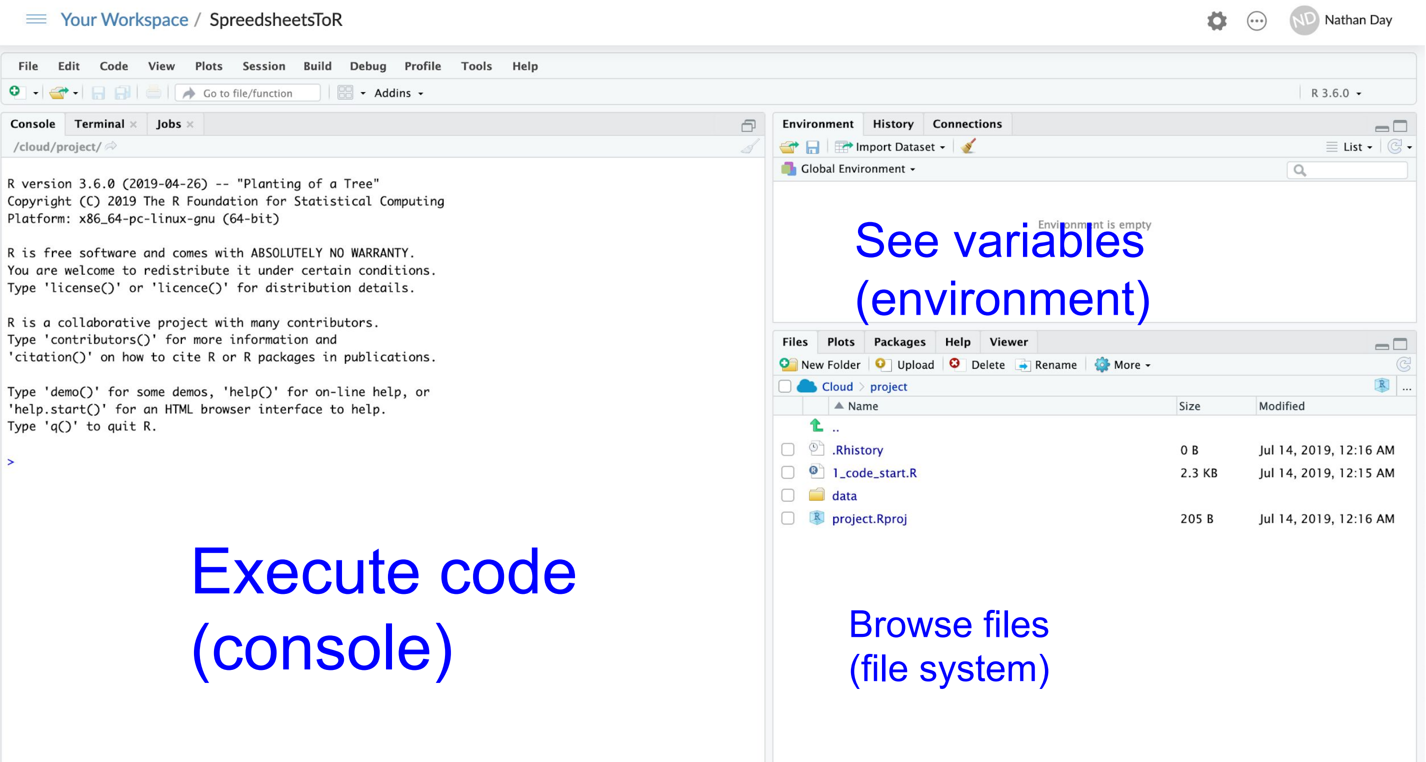
- Built for data
- Developed in New Zealand
- Wonderful community

What are you doing tonight?

https://rstudio.cloud/project/411105



R Studio layout


Your Workspace / SpreedsheetsToR
Nathan Day
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
R 3.6.0
Console Terminal Jobs
/cloud/project/
Environment History Connections
Import Dataset
List
Global Environment
Environment is empty
Files Plots Packages Help Viewer
New Folder Upload Delete Rename More
Cloud > project

```
R version 3.6.0 (2019-04-26) -- "Planting of a Tree"
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.

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.

>
```

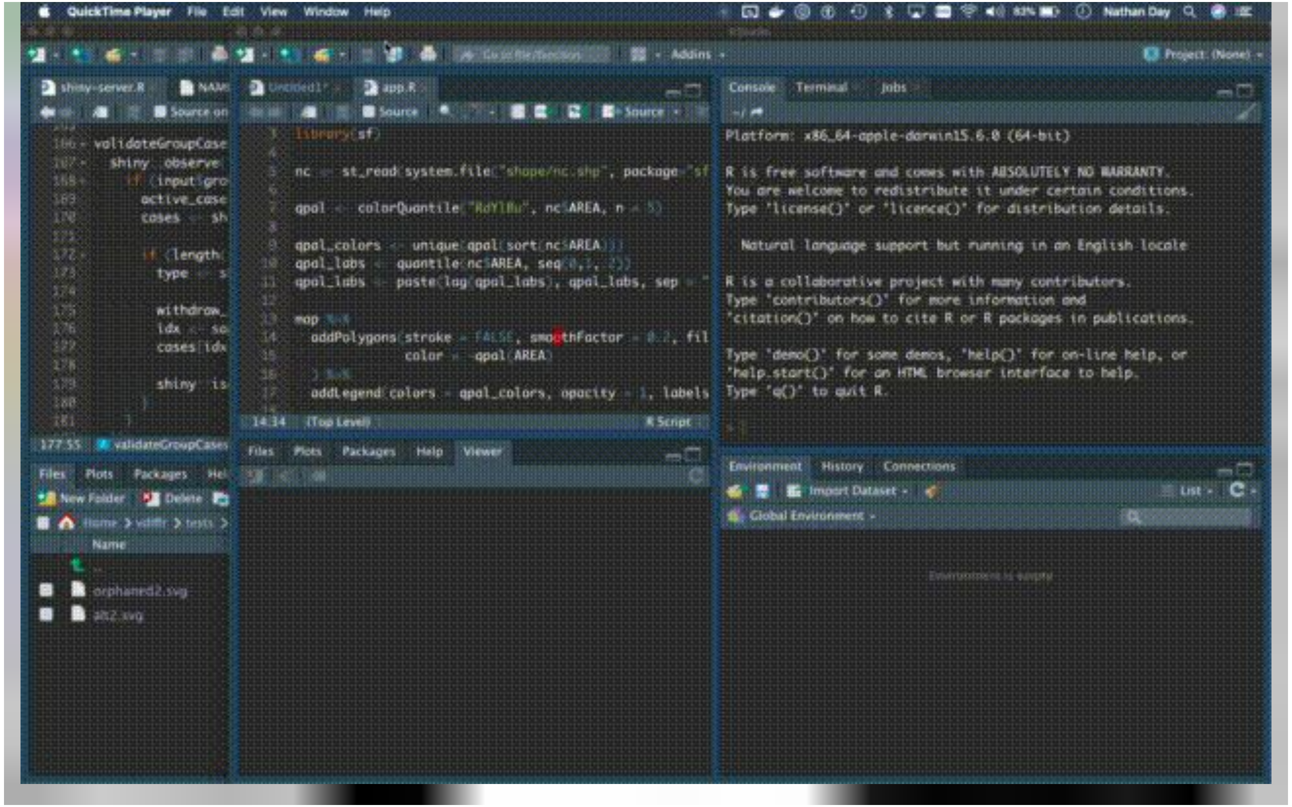
Name	Size	Modified
..		
.Rhistory	0 B	Jul 14, 2019, 12:16 AM
1_code_start.R	2.3 KB	Jul 14, 2019, 12:15 AM
data		
project.Rproj	205 B	Jul 14, 2019, 12:16 AM

See variables (environment)

Execute code (console)

Browse files (file system)

First things first...



```
library(sf)

nc <- st_read(system.file("shape/nc.shp", package="sf

qpal <- colorQuantile("RdYlBu", nc$AREA, n = 5)

qpal_colors <- unique(qpal(sort(nc$AREA)))
qpal_labs <- quantile(nc$AREA, seq(0,1,.2))
qpal_labs <- paste(lag(qpal_labs), qpal_labs, sep = "

map %>%
  addPolygons(stroke = FALSE, smoothFactor = 0.2, fil
              color = ~qpal(AREA)
  ) %>%
  addLegend(colors = qpal_colors, opacity = 1, labels
```

```
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.

>
```

Operators

- Define **<-**
- Arithematic **+**, **-**, *****, **/**, **^**
- Range 1:3
- Index [1], [[2]]
- Tabular-index [x, y], [row, column]

functions()

- More complex
  - Multiple steps wrapped into one name
  - Take arguments
  - Return something new
  - Predefined*
- Have help pages
  - See one… `?class()`

```
?help_pages()
```

Types & Classes

Both control behavior, but on different levels

Types (low-level / memory storage)	Classes (high-level / object properties)
● Numeric ○ Integer ○ Real ○ Complex ● Character ● Logical	● A numeric can interact with the function \`mean()\` ● A character can not

Most of the time `class()` is enough....

Objects in R

Vectors	List	Data Frame
● Must be all one type ● Indexed by length ● *Will be coerced to most flexible type*	● Can hold anything ● Indexed by length or element	● Special list ○ Infinite number of elements all the same length ● Indexed by columns, rows or x,y coordinates

Section 2

- Recap
  - Languages are unique, but related
  - Operators/Functions do things
  - Classes matter for doing things
- Up next…
  - Getting more functions
  - Working with Excel data
  - Plotting with ggplot2

Packages

- Collection of functions with a common purpose
- Access other peoples solutions
- Tested by a town
- Lego bricks of usefulness

Picking the right package

- Google: “R package to {do something}”
    - “... read XLSX file”
    - “... make plots”
- If multiple options exist:
    - Check when last updated
    - Star gazers

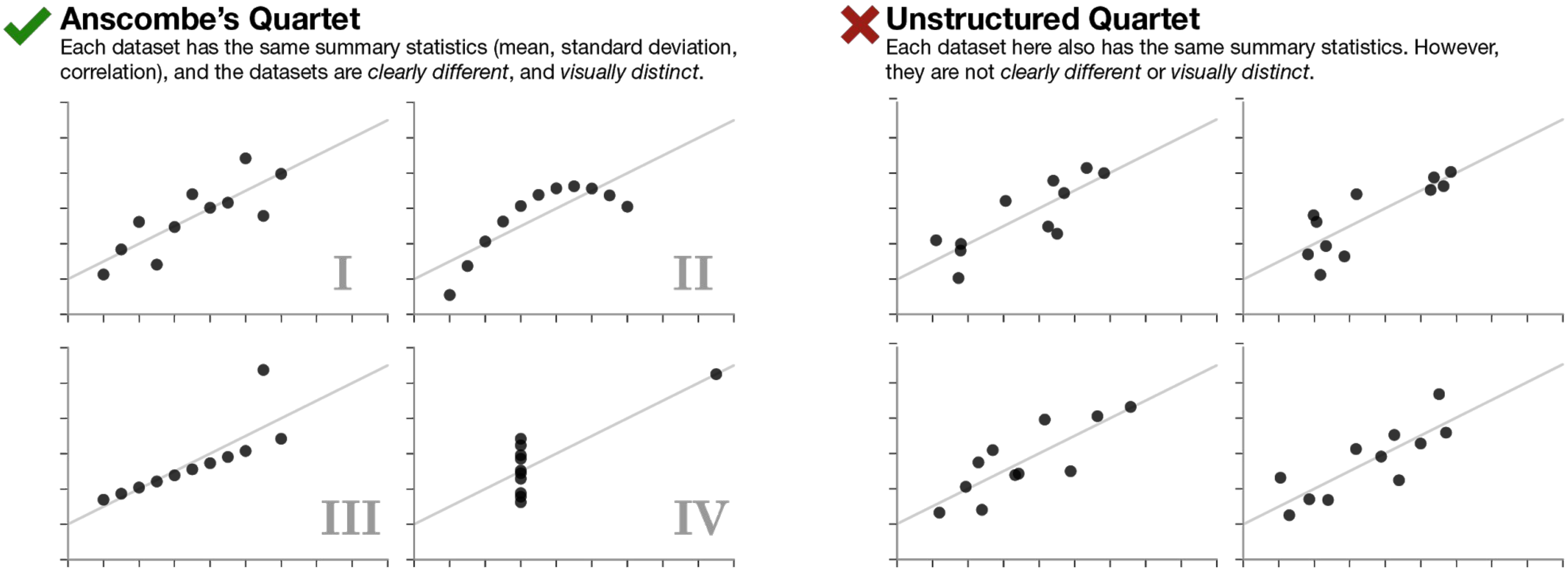
Data in spreadsheets

- Excel is ubiquitous
  - More complicated than a CSV
  - Access existing workflows/data-streams
- Code records the interaction with a document
  - Ability to automate
- Protect the original
  - Reproducible research

readxl

- Reads both .xls and .xlsx
- Control read in by:
    - Sheet - name or index
    - Rows
    - Columns

Anscombe’s quartet



✓ Anscombe’s Quartet

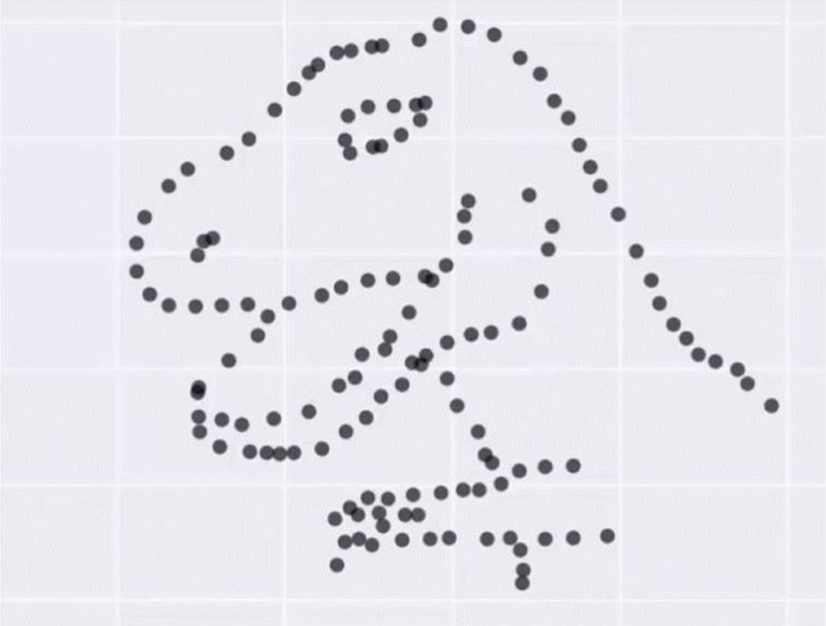
Each dataset has the same summary statistics (mean, standard deviation, correlation), and the datasets are *clearly different*, and *visually distinct*.

✗ Unstructured Quartet

Each dataset here also has the same summary statistics. However, they are not *clearly different* or *visually distinct*.

https://www.autodeskresearch.com/publications/samestats

The Datasaurus Dozen



https://www.autodeskresearch.com/publications/samestats

Data visualization

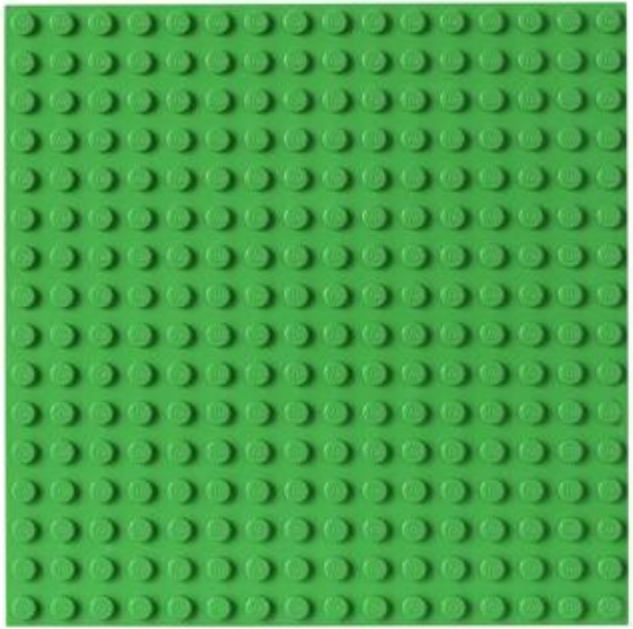
- Critical to *EVERY* analysis
- Exploration
  - Look for patterns
- Evaluation
  - Model a change
- Validation
  - Check model behaves

ggplot2

- Implements the Grammar of Graphics
  - Plots are built in layers
- Layers are stacked with the **+** operator
- Works on columns of a data frame

ggplot(data = NULL, mapping = aes()) + ...

- Creates a new plot
  - Returns the foundation layer
- Every added layer listens to this one
  - Cascades down until you stop it
- `aes()` is special



https://www.shopbecker.com/blocks-and-manipulatives/lego-and-duplo/_/lego-small-building-plates/?

aes(x, y, ...)

- Construct aesthetic mapping
    - Returns a legend and required scales
- Allows name reference to columns in `data`
- Scales supported:
    - Color (outline)
    - Size
    - Alpha (transparancy)
    - Shape (categorical only)
    - Fill (interior)
    - Stroke (outline size of points)
    - Linetype

`geom_someshape(mapping = NULL) + ...`

- Adds a new layer on top
  - Creates the useful part / carries information
- Mapping cascades down from `ggplot() +`
- Redefine a mapping variable to override the cascade
  - `NULL` with remove an existing mapping without re-assigning
- Infinitely stackable

stat_summary()

- Specialized layer
  - Assumes a grouping
  - Performs summary statistics
  - Adds new layer with those results
- Summary functions are adjustable
  - Specified on new variables on y-axis
  - Can calculate 1 (y) or 3 (ymin/y/ymax) values
- So are resulting geoms
  - Must match values created by summary function
  - Point ~ 1 value
  - Crossbar ~ 3 values

labs(...)

- Modify axis, legend, and plot labels
- Critical for readability/usefulness
  - Column names are usually not suitable by themselves

Section 3

- Recap
  - Packages exist for everything*
  - Data viz is always important
  - ggplot is a language of layers
- Up next...
  - Data wrangling
    - Sort
    - Subset
    - Summarize
    - Augment

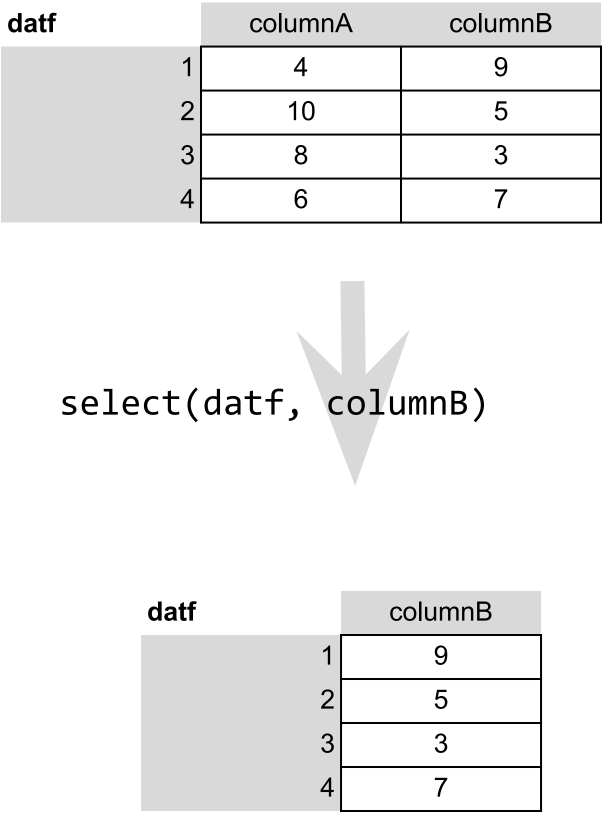
There’s an R package for dat

- dplyr - dee-plier
- Language of data frame manipulation
  - select() picks/drops columns
  - arrange() sorts rows
  - filter() picks/drops rows
  - mutate() adds new columns
  - summarise() adds new collapsed columns
- Share the same pattern

https://dplyr.tidyverse.org/index.html

`select(.data, ...)`

- Keep columns of `.data`
- `...` column names
- `-column_name` removes column
- Allows rearrangement of columns
- Helper functions to select multiple at once
  - `starts_with(‘a pattern’)`
  - `ends_with(‘a pattern’)`
  - `matches(‘another pattern’)`



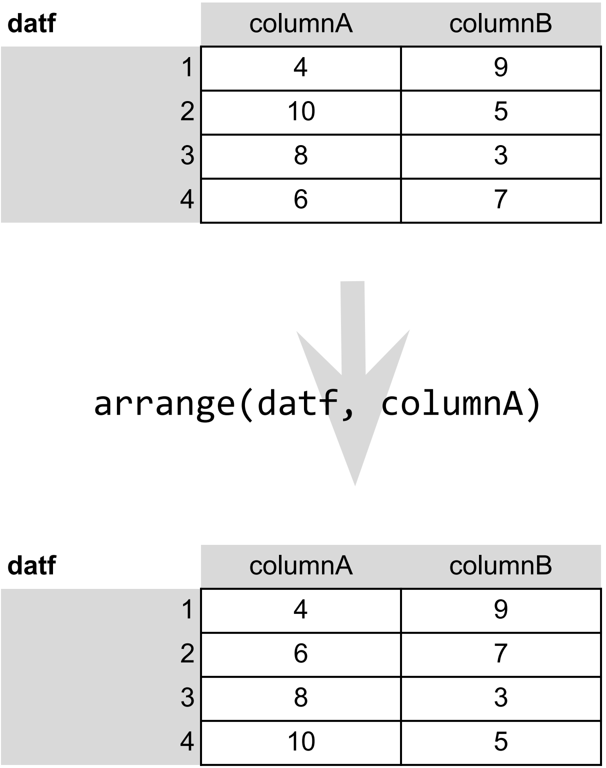
datf	columnA	columnB
1	4	9
2	10	5
3	8	3
4	6	7

```
select(datf, columnB)
```

datf	columnB
1	9
2	5
3	3
4	7

arrange(.data, ...)

- Sort .data by column(s)
- ... column names
- Default is ascending order
    - Use `desc(column_name)` to get descending order



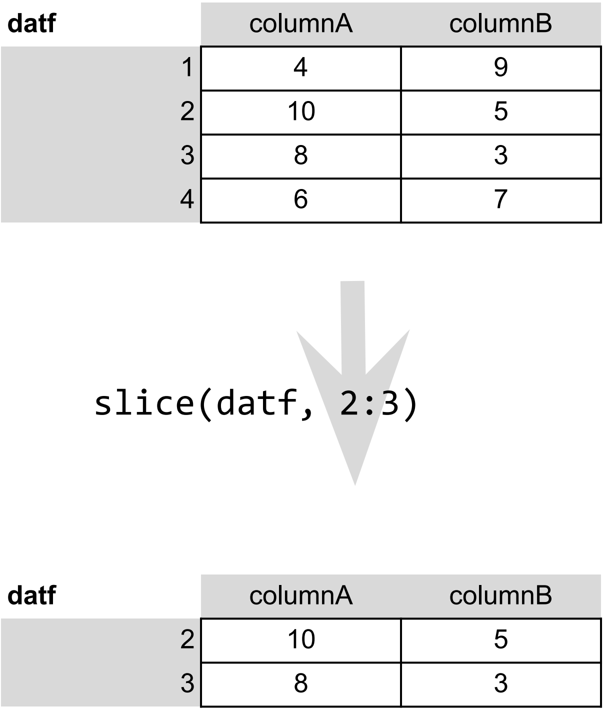
datf	columnA	columnB
1	4	9
2	10	5
3	8	3
4	6	7

```
arrange(datf, columnA)
```

datf	columnA	columnB
1	4	9
2	6	7
3	8	3
4	10	5

slice(.data, ...)

- Keep rows of `.data` by index(s)
- `...` integers
- Negative values remove rows



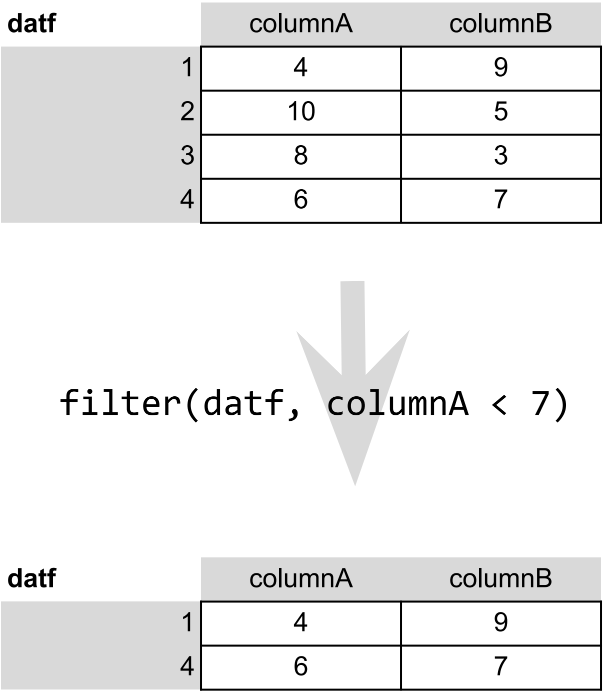
datf	columnA	columnB
1	4	9
2	10	5
3	8	3
4	6	7

```
slice(datf, 2:3)
```

datf	columnA	columnB
2	10	5
3	8	3

filter(.data, ...)

- Keep rows of `.data` by logical(s)
- `...` conditionals
  - ==
  - >, <, >=, <=
  - Any function that returns TRUE/FALSE
- Linked by AND `&` or OR `|`



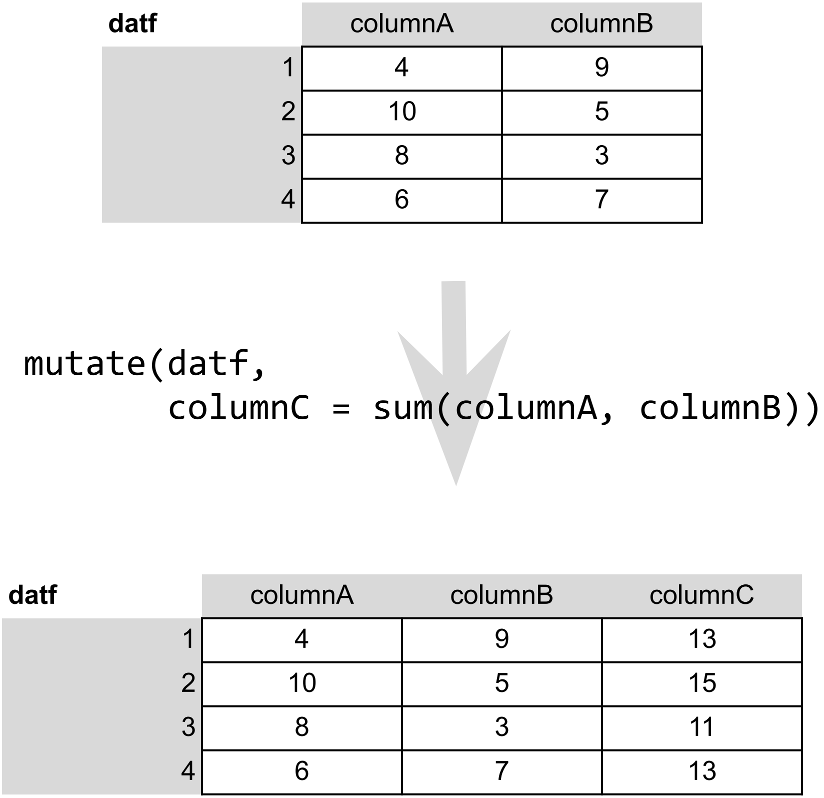
datf	columnA	columnB
1	4	9
2	10	5
3	8	3
4	6	7

```
filter(datf, columnA < 7)
```

datf	columnA	columnB
1	4	9
4	6	7

mutate(.data, ...)

- Add new column(s) to .data
- ... name = values pairings
- All `values` must be of length 1 or nrow(.data)
  - Single values will be repeated
  - Usually the result of a function



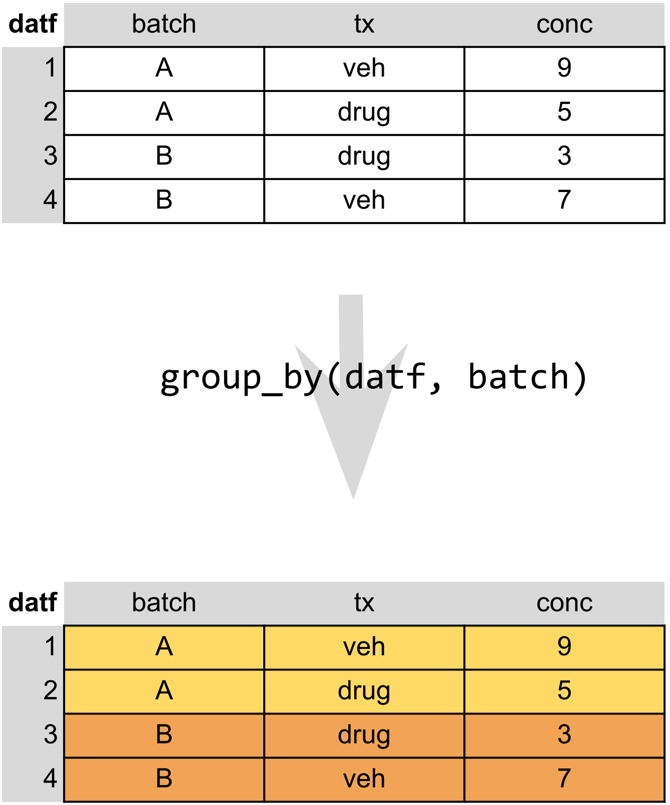
datf	columnA	columnB
1	4	9
2	10	5
3	8	3
4	6	7

```
mutate(datf,
       columnC = sum(columnA, columnB))
```

datf	columnA	columnB	columnC
1	4	9	13
2	10	5	15
3	8	3	11
4	6	7	13

`group_by(...)`

- `...` column(s) to set sub-grouping
- Only useful when paired with other functions
  - Remember `stat_summary()`?
- Can be used with:
  - `arrange()`
  - `slice()`
  - `filter()`
  - `mutate()`
  - **`summarise()`**



datf	batch	tx	conc
1	A	veh	9
2	A	drug	5
3	B	drug	3
4	B	veh	7

```
group_by(datf, batch)
```

datf	batch	tx	conc
1	A	veh	9
2	A	drug	5
3	B	drug	3
4	B	veh	7

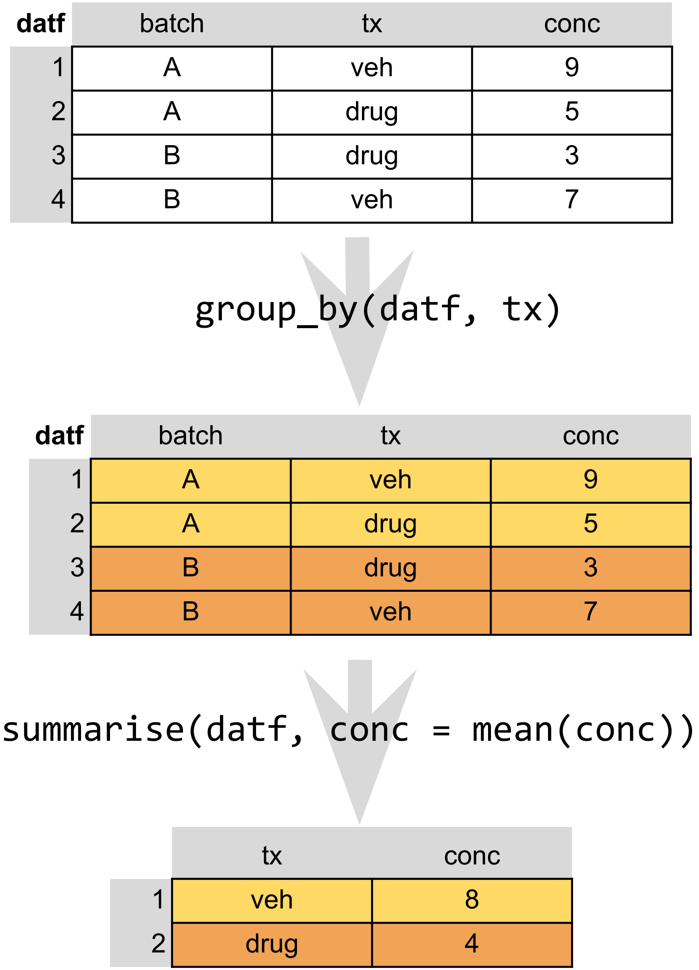
%>%

- The “pipe” operator
- Useful for linking functions together
  - Helpful for breaking out nested function calls
- Carries the result of function1 into a argument for function2

Rene Magritte, *The Treachery of Images*, 1929

summarise(.data, ...)

- Add new columns to `.data`
  - Remember `mutate()`?
- Collapses all rows in a group to a single row
- `...` new_column_name = new_values pairings
- `new_values` must be single value
  - Usually the result of a function



datf	batch	tx	conc
1	A	veh	9
2	A	drug	5
3	B	drug	3
4	B	veh	7

`group_by(datf, tx)`

datf	batch	tx	conc
1	A	veh	9
2	A	drug	5
3	B	drug	3
4	B	veh	7

`summarise(datf, conc = mean(conc))`

	tx	conc
1	veh	8
2	drug	4

Section 4

- Recap
  - Subset data
  - Calculate new values
  - Link multiple functions
- Up next
  - Statistical tests
    - Formulas
  - Bringing it all together

Capstone

1. Get data out of ‘curveball.xlsx’
   a. Combine all sheets into one data frame
2. See the treatment effect
   a. Show all of the data points
   b. Show some group statistics
   c. Summarise as a results table
      i. By group calculate: n, mean, standard deviation
3. Test your hypothesis
   a. T-test
   b. Linear model
   c. Build a final plot with results

Formulas

- Special R syntax for models
  - Reference columns in `data` by name
- *Left Hand Side ~ Right Hand Side*
  - *Response ~ Predictor(s)*

t.test(formula, data)

- Are groups different?
  - Is one group different than zero?
  - Are two groups different from each other?
- Focused on working with small samples
- Developed to monitor quality of stout at Guinness brewery
  - William Sealy Gosset published under a pen-name *Student*, due to company policy against publishing
  - Friends with both Karl Pearson and R.A. Fischer

https://en.wikipedia.org/wiki/Student%27s_t-test

lm(formula, data)

- lm = linear model
  - Estimate relationship response and predictor(s)
- Two main uses:
  - Prediction - forecasting
  - **Explanation - quantify strength of relationships**
- This is machine learning

Thanks for learning!

- Github repository
  - https://github.com/nathancday/Spreadsheet-to-Rstats
- Rstudio Cloud
  - https://rstudio.cloud/project/411105