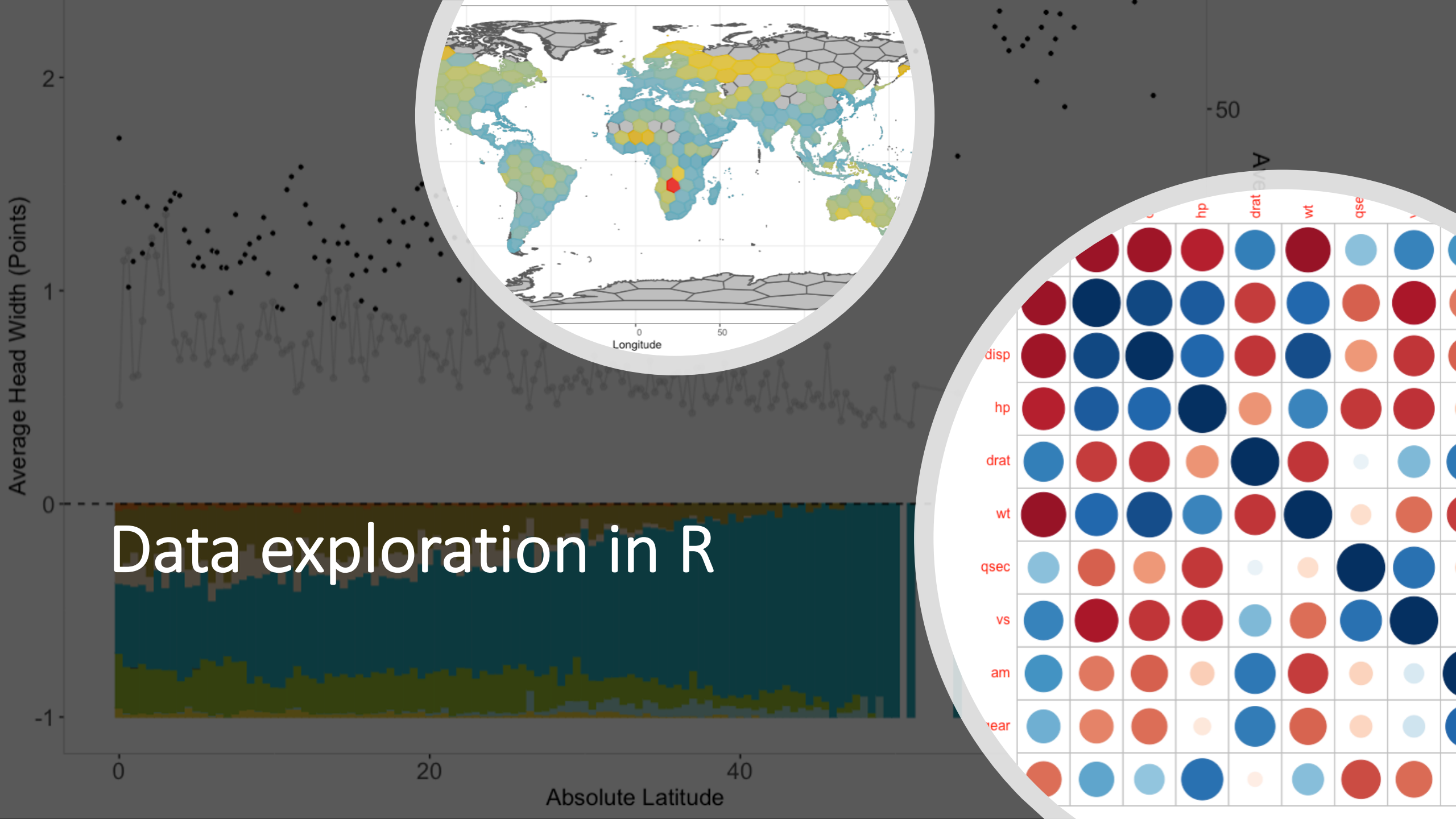


Data exploration in R

Why data exploration?

- Knowing your data is key to analyzing it
  - Limitations of your data
  - Limitations of analyses you can run
- Prevents misuse of statistical approaches and methods
- Makes *you* the expert of *your data*
- *Gives/ introduces you to a marketable skill in an age of data*

Cautions

- Data exploration can get messy *fast*
  - R scripts become disorganized

- How to avoid messy scripts:
  - Annotate your scripts
    - Seriously annotate
  - Use GitHub (resources at the end of the Powerpoint)
  - USE TIDYVERSE, results in less objects in the R environment
    - Others are likely to better understand what you did

```
function addlist(i){
document.getElementById(i).addEventListener("click",
function(){
  document.getElementById(i).style.backgroundColor = "green
  document.getElementById(i).setAttribute("data-checked","t
  if ($("[data-checked='true']").length == 2){
  console.log("2 checked");
 var  x1 = $("[data-checked='true']")[0].style.left.substri
  var  x2 = $("[data-checked='true']")[1].style.left.substr
  var  y1 = $("[data-checked='true']")[0].style.top.substri
  var  y2 =
  console.
var c = do
var ctx =
ctx.beginP
ctx.moveTo
ctx.lineTo
ctx.stroke
}
```

df_sub	1766826 obs. of 16 variables
empty_df	7908 obs. of 18 variables
func_role	353 obs. of 2 variables
func_rolef	Large gls (16 elements, 954.6 Kb)
func_rolefint	Large gls (16 elements, 956.1 Kb)
grid_ants_subsample	1766826 obs. of 15 variables
grid_ants2b	1766826 obs. of 18 variables
grid_ants4b_func_role	353 obs. of 15 variables
grid_antsb	1766826 obs. of 16 variables
grid3bfunc	503 obs. of 16 variables
hell	39418 obs. of 13 variables
hell2	39308 obs. of 18 variables
hell55	9 obs. of 13 variables
hex_background	503 obs. of 2 variables
hex_bins	503 obs. of 15 variables
hex_bins_func	503 obs. of 16 variables
hex_bins_func2	299 obs. of 16 variables
hex_bins2	311 obs. of 15 variables
hex_bins3	311 obs. of 28 variables
hexbins_2.1	311 obs. of 13 variables
model_1	Large lm (12 elements, 1.5 Mb)
model_df	311 obs. of 3 variables

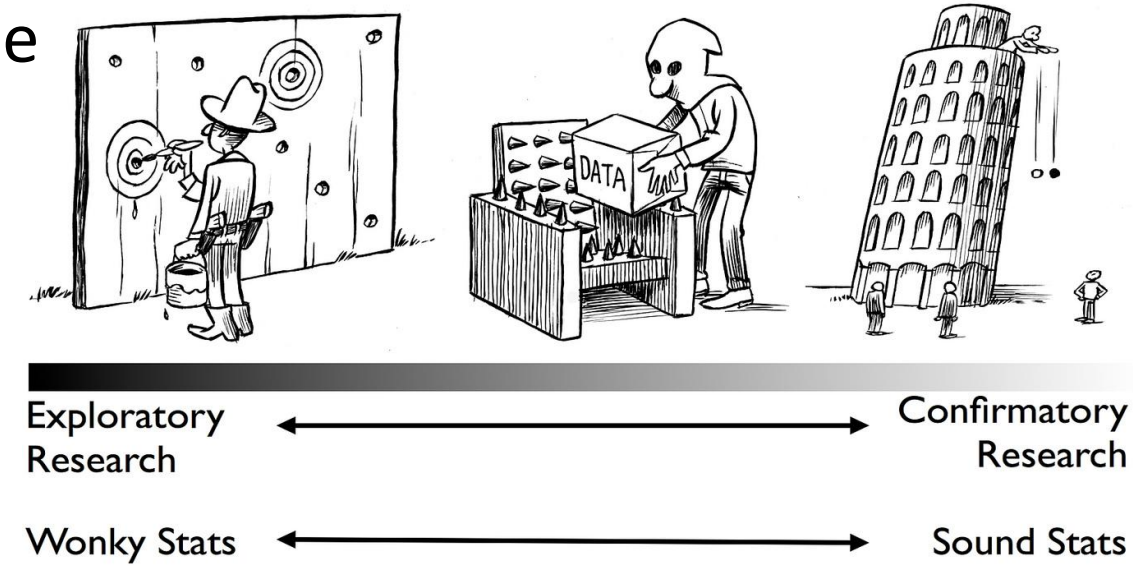
Cautions

- A priori hypotheses are extremely important
- You **shouldn't** create hypotheses on the fly while exploring data

DATA DREDGING

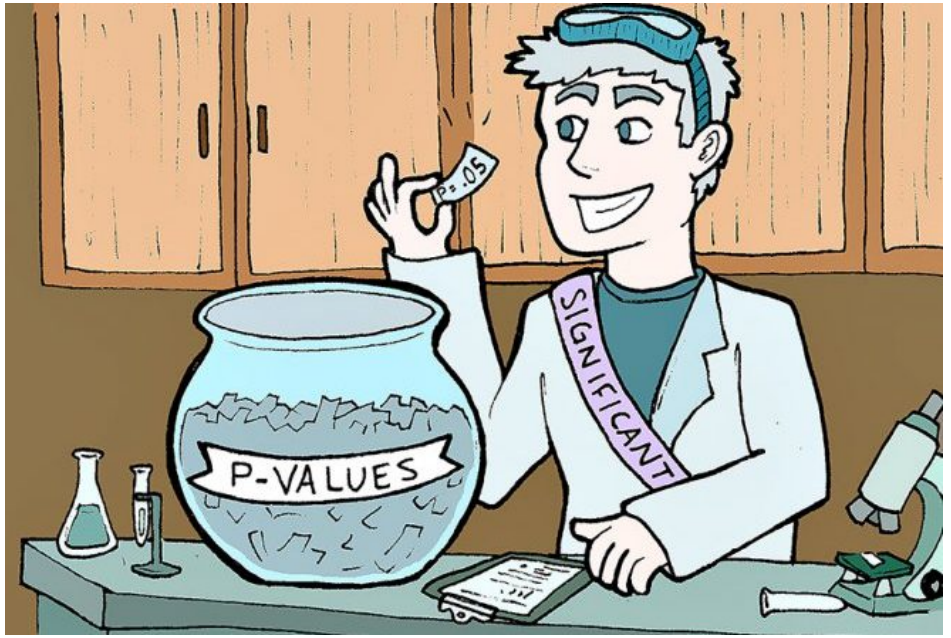
Repeatedly testing new hypotheses against the same set of data, failing to acknowledge that most correlations will be the result of chance.

- Exploratory science is fine but should be grounded in some research context



Cautions

- Exploratory research is still important
  - Many disciplines often focus on identifying patterns and fitting statistical models to them
    - e.g. Large-scale studies, macroecology etc.
  - Empirical studies often use data exploration and are valuable



Age of big data

- We live in the age of data
- It’s not just big-data, data exploration is applicable to even smaller sets of data
- Knowing how to access, work with, and deliver data will likely be part of many science jobs

February 11, 2012

The Age of Big Data

By STEVE LOHR

This article discusses the power of big-data (e.g. stock markets, Sabermetrics etc.) and the dangers (e.g. P-hacking, self-fulfilling hypotheses).

Major points to data exploration

1. Examine and validate data structure: factors, strings, numeric, integers
2. Assess outliers in the dataset
3. Assess distributions of your data and relevant summary statistics
4. Measures of central tendency
5. Assess relationships/collinearity between variables
6. Basic descriptive exploration approaches

Examine and validate data structure: factors, strings, numeric, integers

- Make sure your data is formatted correctly
  - e.g. Strings may not be converted to factors e.g. use ‘stringsAsFactors’ argument in read.csv()
- To quickly evaluate the structure of your data in R use the str() command

```
> ## Data exploration
> data("iris")
> str(iris)
'data.frame':   150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Examine and validate data structure: factors, strings, numeric, integers cont.

- If you need to convert, then you can do it in the R

```
> iris$string_species<-as.character(iris$Species)
> str(iris)
'data.frame':   150 obs. of  6 variables:
 $ Sepal.Length  : num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width   : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length  : num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width   : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species       : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ..
 $ string_species: chr  "setosa" "setosa" "setosa" "setosa" ...
```

Assessing distributions of your data and relevant summary statistics

- Basic averages, minimums, maximums etc. of each of your columns can help you understand what sort of data you are working with
  - The summary() helps to quickly show this information

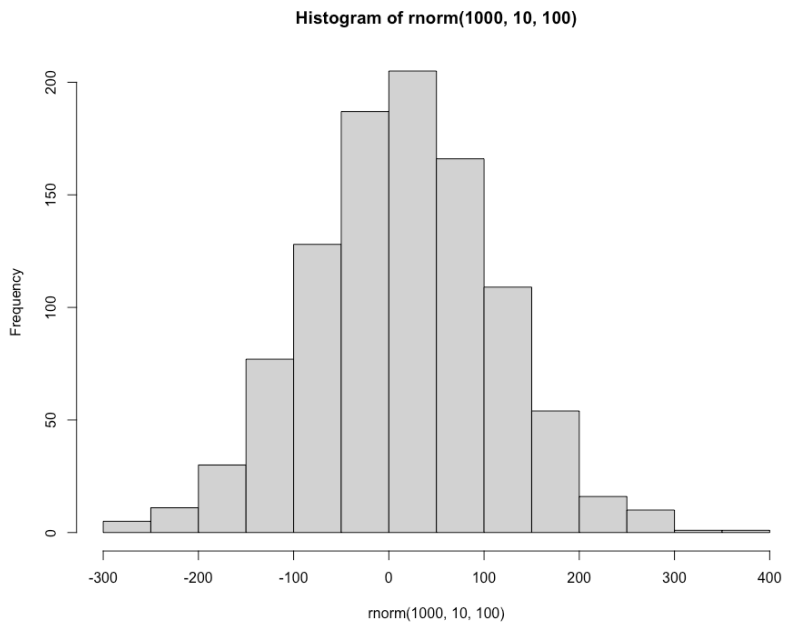
```
> data("iris")
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa     :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

```
> summary(iris)
  Sepal.Length    Sepal.Width         Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :    2.000   Min.   :1.000   Min.   :0.100   setosa     :50
 1st Qu.:5.100   1st Qu.:    2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :    3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :   69.703   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:    3.375   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :10000.000   Max.   :6.900   Max.   :2.500
```

Normality

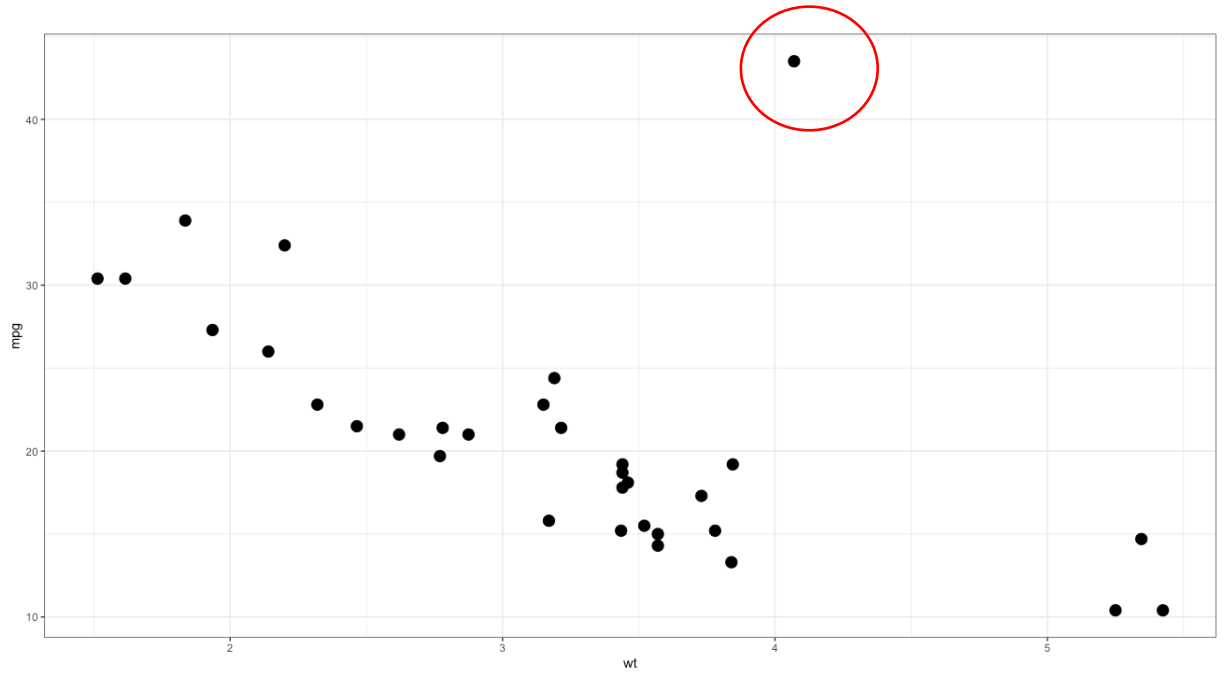
- Oftentimes assumptions of normality are based on model residuals and not necessarily the data
- BUT normally distributed data can help
- NO REQUIREMENT for your predictor variables to be normally distributed!

- Usually check for normality with histograms



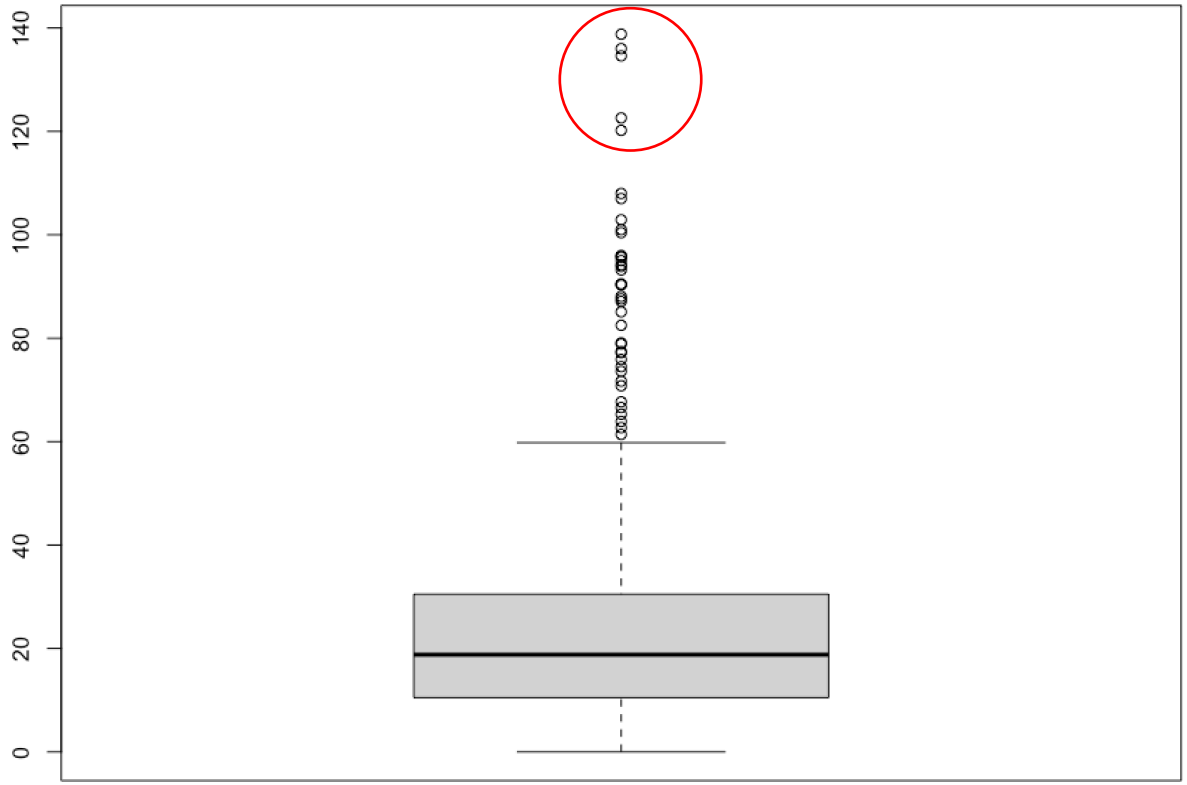
Outliers

- What are outliers?
  - Data that are significantly different from the rest of your data
- They can be a result of:
  - User error
  - Extreme variation
    - E.g. Allometric data has a lot of variation



How to identify outliers?

- Boxplots can help for initial assessments
  - Here we see that this data looks like it has 10-20 suspicious points



Testing for outliers

- Rosner test
  - Ideal for sample size > 20
  - Can test multiple values at once

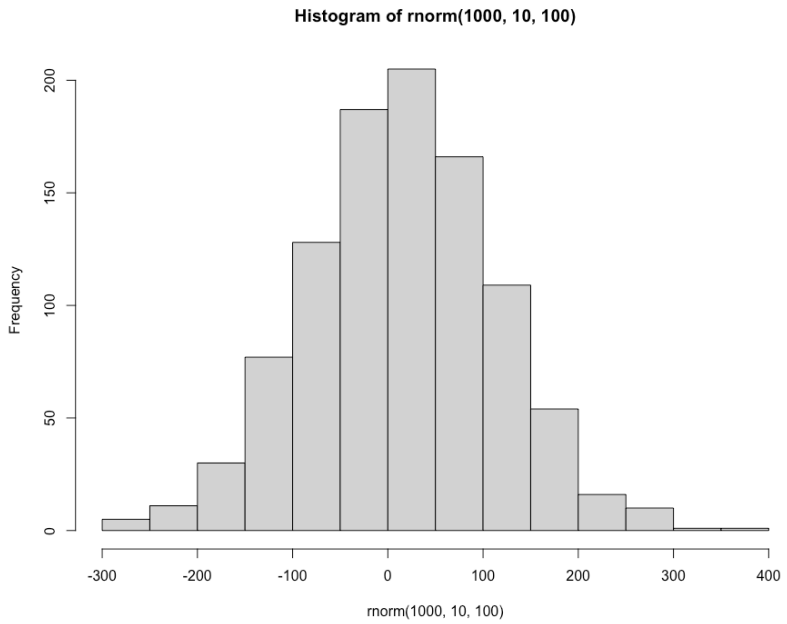
- TRUE = outlier
- FALSE = non-outlier
- Value = value of the specific data point

```
27  library(EnvStats)
28  test <- rosnerTest(tomato$yield, k =15)
29  test$all.stats
```

	i	Mean.i	SD.i	Value	Obs.Num	R.i+1	lambda.i+1	Outlier
1	0	28.13459	29.03062	138.8	246	3.812024	3.689282	TRUE
2	1	27.71698	28.27374	136.0	192	3.829809	3.688199	TRUE
3	2	27.30682	27.52622	134.6	264	3.897854	3.687111	TRUE
4	3	26.89886	26.76713	122.6	138	3.575323	3.686019	FALSE
5	4	26.53359	26.15339	120.2	120	3.581426	3.684922	FALSE
6	5	26.17471	25.54911	108.0	140	3.202666	3.683820	FALSE
7	6	25.86000	25.08639	107.0	191	3.234423	3.682714	FALSE
8	7	25.54672	24.62011	102.9	102	3.141874	3.681602	FALSE
9	8	25.24690	24.18958	101.0	194	3.131642	3.680486	FALSE
10	9	24.95214	23.76801	100.4	156	3.174345	3.679364	FALSE
11	10	24.65742	23.33931	96.0	32	3.056757	3.678238	FALSE
12	11	24.37765	22.95105	95.8	14	3.111942	3.677106	FALSE
13	12	24.09646	22.55196	95.7	119	3.175048	3.675969	FALSE
14	13	23.81344	22.14008	95.0	158	3.215280	3.674828	FALSE
15	14	23.53095	21.72249	94.2	86	3.253267	3.673681	FALSE

Caveats

- Many of these ”outlier tests” assume your variable is normally distributed without the outliers
  - This is rare in ecological data
- Outliers will usually be determined by first-hand knowledge of collection methods



How to deal with outliers?

- Remove them
  - Should have a good justification to do this (e.g. compromised sampling unit etc.)

- Dampen them:
  - If you're analyzing averages then:
    - Use geometric averages
    - Dampens effects of outliers
    - Used frequently with allometric data

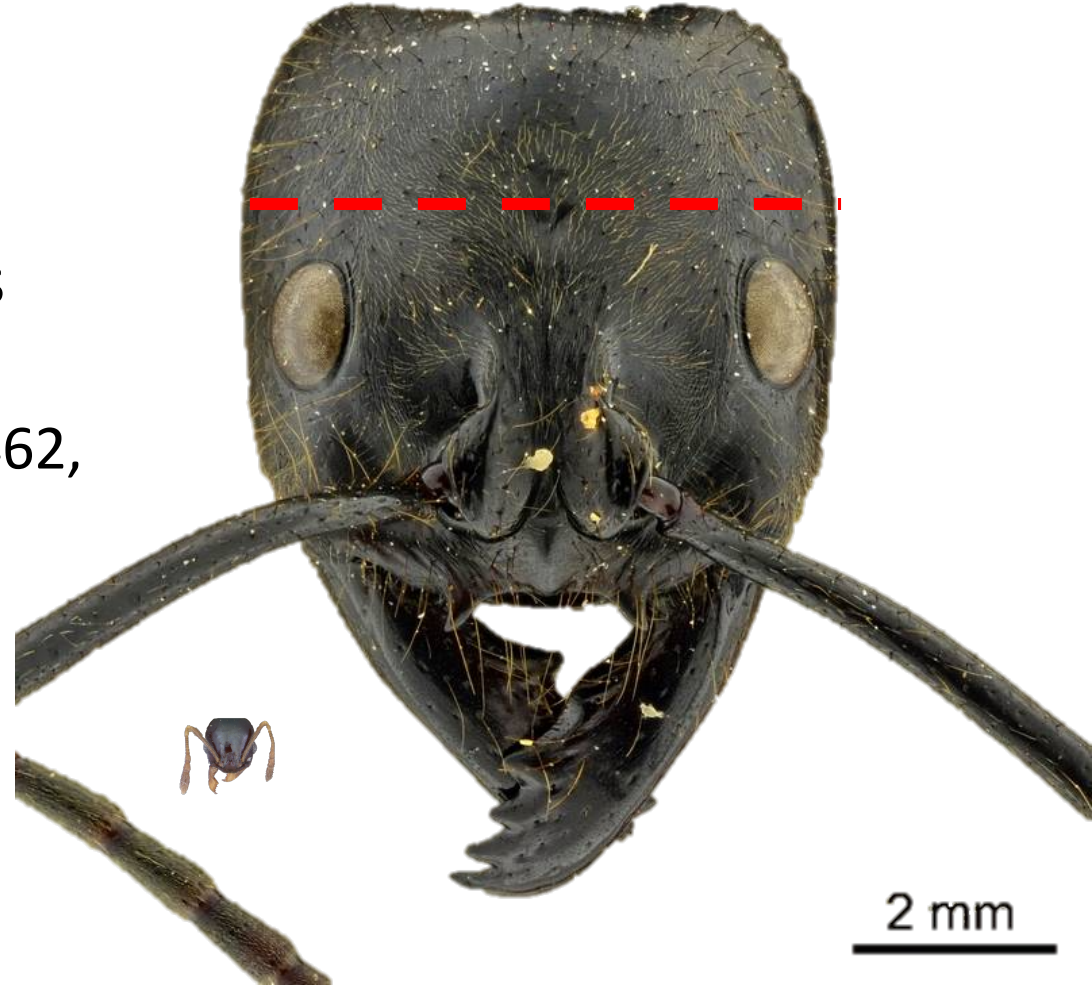
Measures of central tendency

- What are measures of central tendency?
  - One value that attempts to describe an entire distribution of data
  - Value often represents the center of this data or distribution

- Means, Medians, Modes
  - Median: middle of the distribution or set of numbers, often less biased than the mean
    - 1,2,3,4,5,5,6
  - Mode: most frequent number
    - 1,2,3,4,5,5,6

Geometric means

- Example: I measure the head widths of ants (proxy for body size):
  - Measurements in millimeters: 0.321, 0.543, 0.462, 6.023

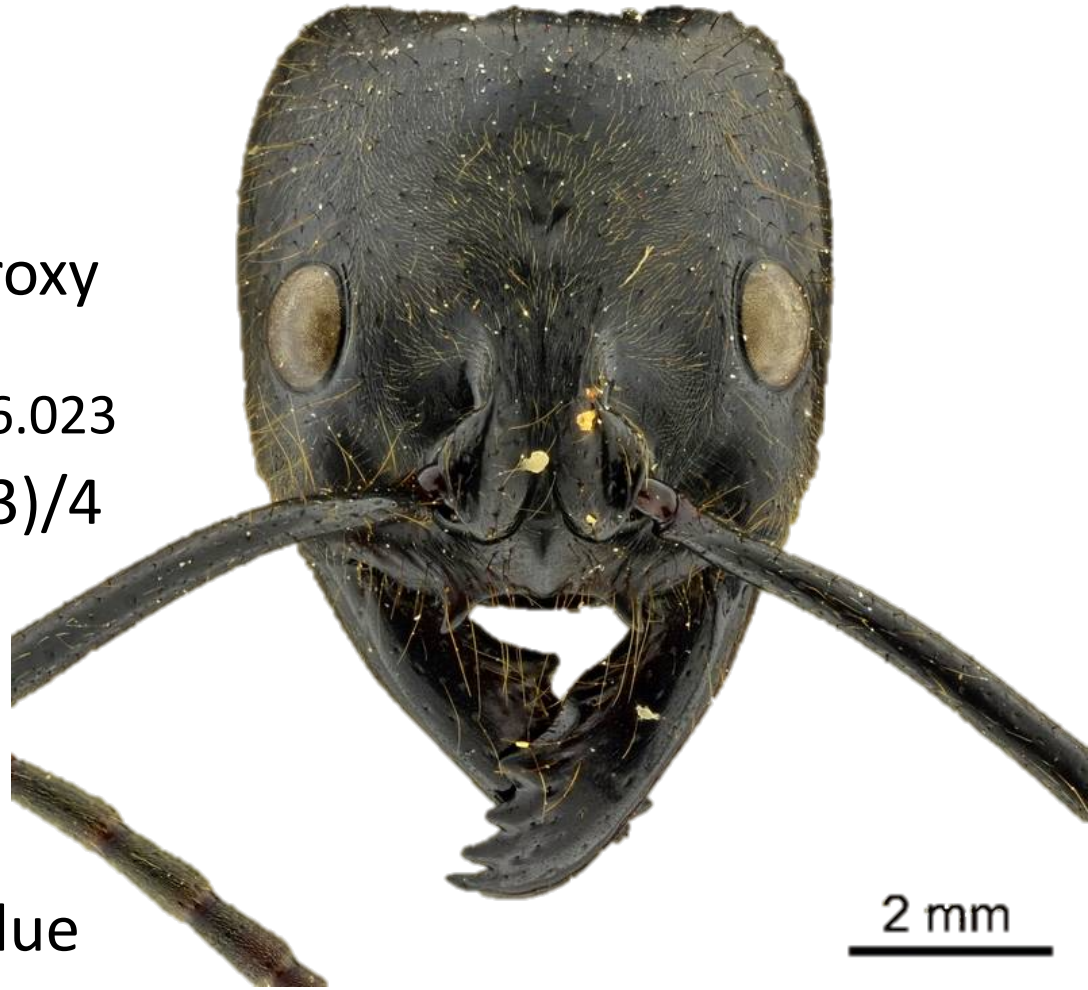


Geometric means

- Example: I measure the head widths of ants (proxy for body size):
  - Measurements in millimeters: 0.321, 0.543, 0.462, 6.023
- Arithmetic mean: (0.321+ 0.543 + 0.462 + 6.023)/4 = 1.84

Geometric means

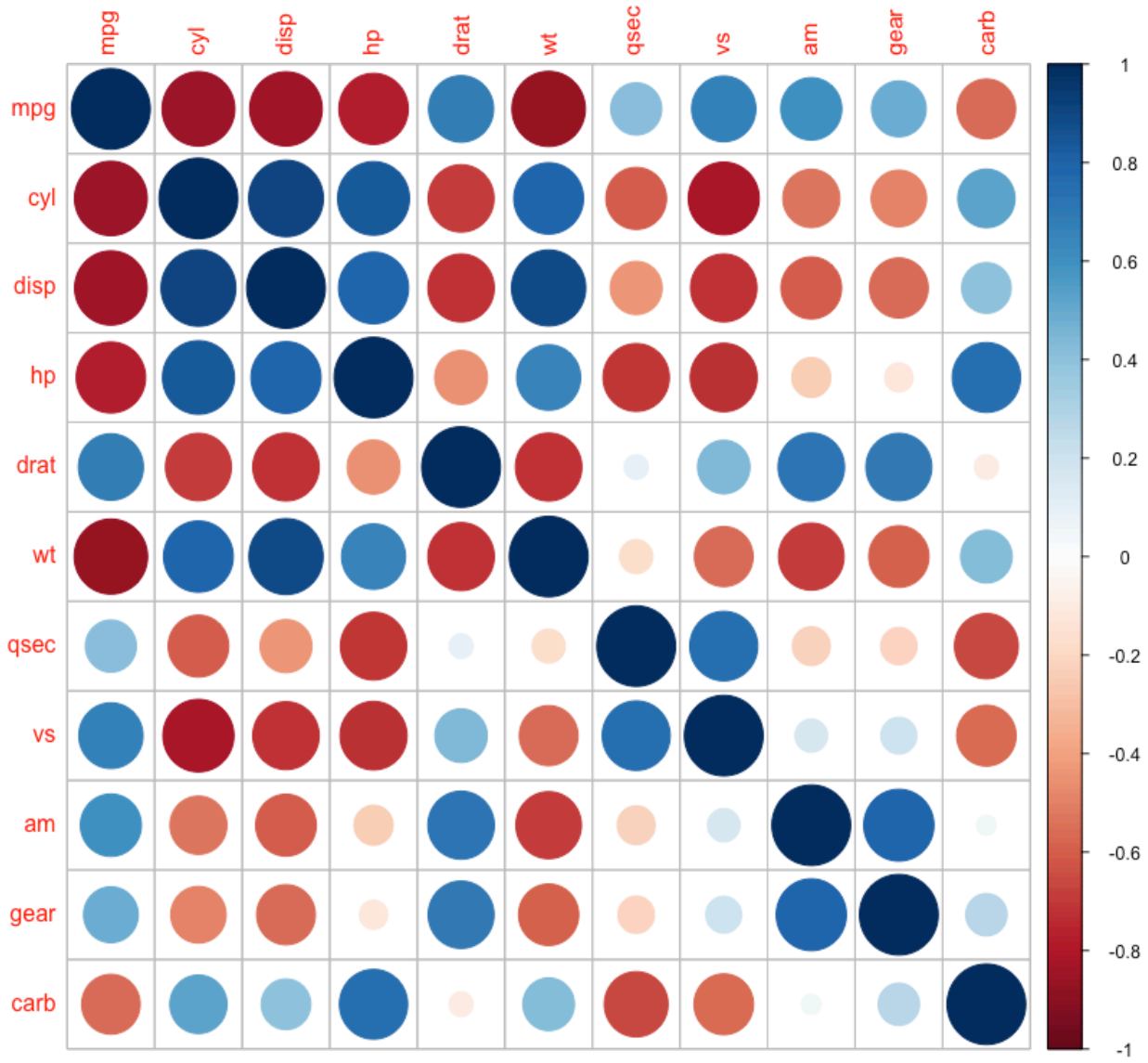
- Example: I measure the head widths of ants (proxy for body size):
  - Measurements in millimeters: 0.321, 0.543, 0.462, 6.023
- Arithmetic mean: (0.321+ 0.543 + 0.462 + 6.023)/4 = 1.84
- Geometric mean: $\sqrt[4]{(0.321 * 0.543 * 0.462 * 6.023)}$ = 0.83



- Which one is a more representative average value of head width?
- Note geo means don’t work with negatives of zeros

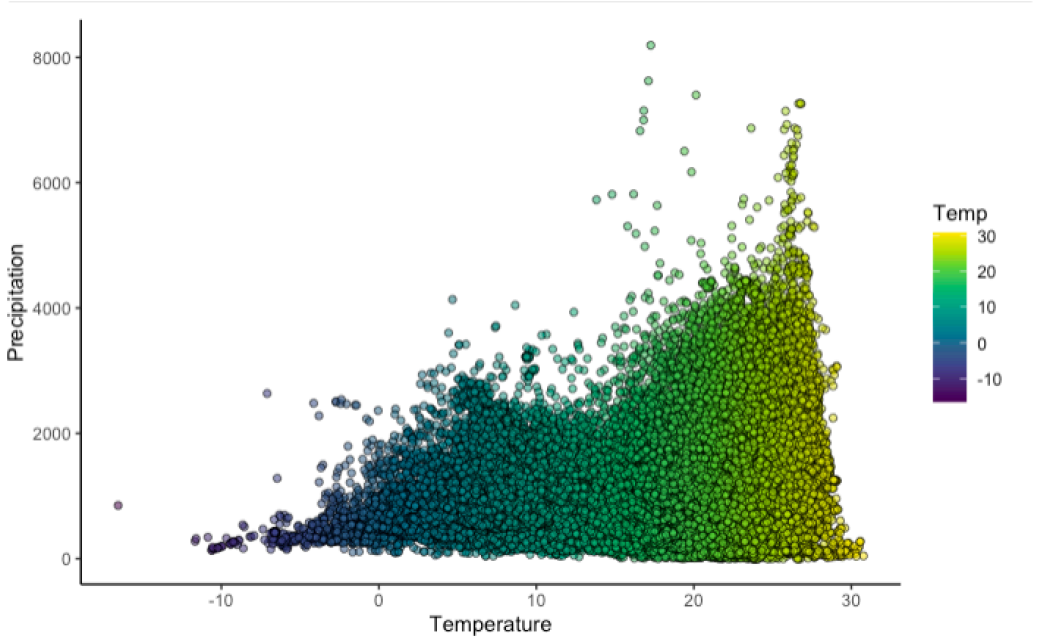
Relationships

- Data exploration also involves the relationship between variables
- Can become overwhelming depending on how many variables you have
- Correlation plots have become useful to deal with this



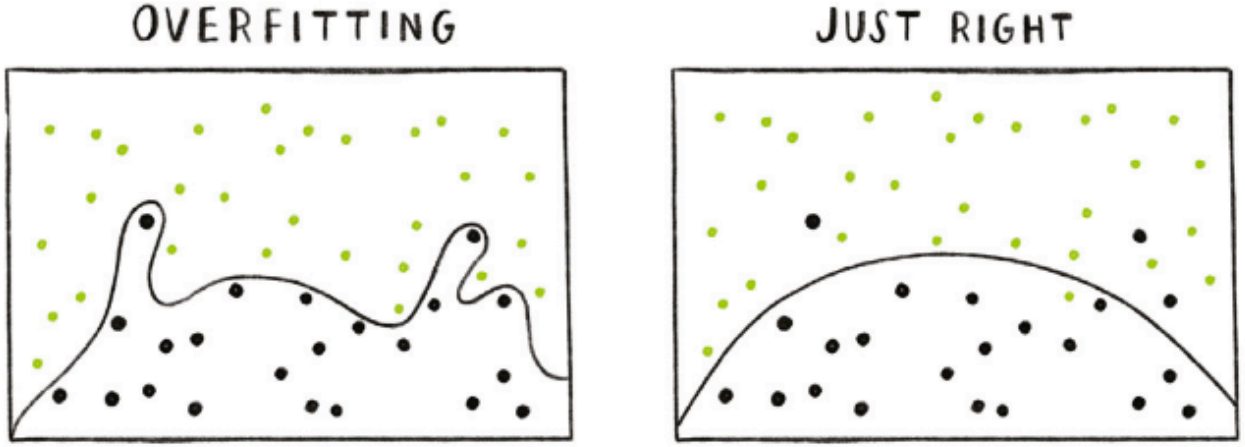
Collinearity

- When two explanatory variables are highly correlated with one another (often in a linear fashion)
  - Classic example: Temperature vs. Precipitation



Collinearity issues in models

- Collinearity can cause model-overfitting
- Overfitting in a model when you're counting a relationship twice
  - Becomes difficult to tell what is driving the actual relationship
- Can influence the estimates of other variables in the model



OVERFITTING

Creating a model that's overly tailored to the data you have and not representative of the general trend.

Resolving collinearity issues in models

- Prior knowledge
  - E.g. You know temperature and precipitation are highly correlated, choose the more biologically meaningful one

- Use variance inflation factors (VIFs) which give a measure of multi-collinearity for a model
  - Anything over 5 is often considered an issue

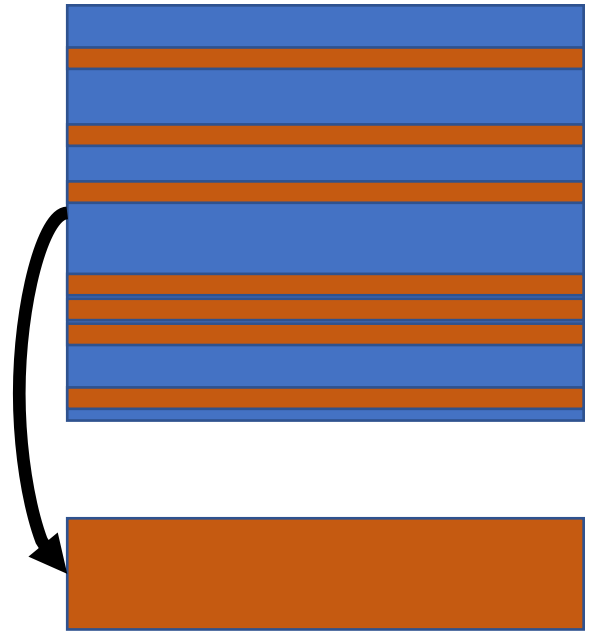
Tools and approaches to data exploration

- Let's use a large dataset as an example
- Data from 'nycflights13' package
- ~336,000 rows X 19 columns
- Shows every flight coming out of three New York airports in 2013
  - Newark
  - LaGuardia
  - JFK

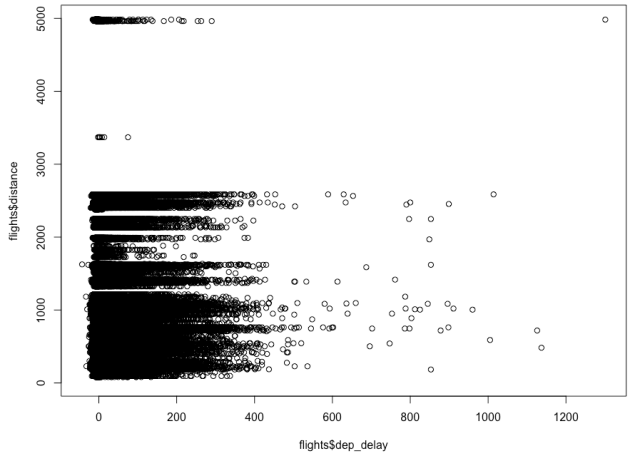
```
> data(flights)
> str(flights)
tibble [336,776 × 19] (S3: tbl_df/tbl/data.frame)
 $ year          : int [1:336776] 2013 2013 2013 201
 $ month         : int [1:336776] 1 1 1 1 1 1 1 1 1
 $ day           : int [1:336776] 1 1 1 1 1 1 1 1 1
 $ dep_time      : int [1:336776] 517 533 542 544 55
 $ sched_dep_time: int [1:336776] 515 529 540 545 60
 $ dep_delay     : num [1:336776] 2 4 2 -1 -6 -4 -5
 $ arr_time      : int [1:336776] 830 850 923 1004 8
 $ sched_arr_time: int [1:336776] 819 830 850 1022 8
 $ arr_delay     : num [1:336776] 11 20 33 -18 -25 1
 $ carrier       : chr [1:336776] "UA" "UA" "AA" "B6
 $ flight        : int [1:336776] 1545 1714 1141 725
 $ tailnum       : chr [1:336776] "N14228" "N24211"
 $ origin        : chr [1:336776] "EWR" "LGA" "JFK"
 $ dest          : chr [1:336776] "IAH" "IAH" "MIA"
 $ air_time      : num [1:336776] 227 227 160 183 11
 $ distance      : num [1:336776] 1400 1416 1089 157
 $ hour          : num [1:336776] 5 5 5 5 6 5 6 6 6
 $ minute        : num [1:336776] 15 29 40 45 0 58 0
 $ time_hour     : POSIXct[1:336776], format: "2013-
-01-01 05:00:00" ...
```

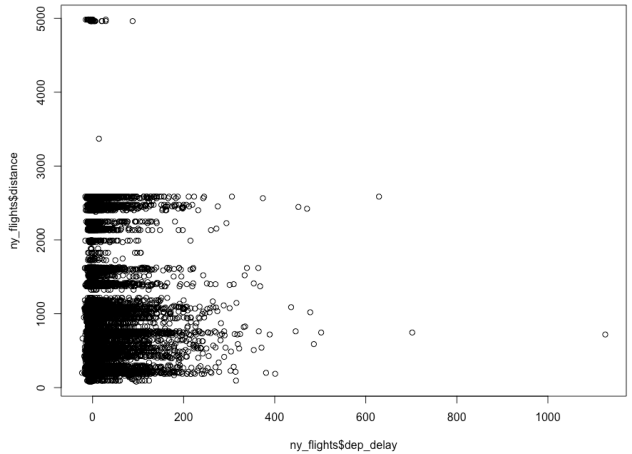
NYCflights13

- Large data = can be slow to work with
- Take a subset to work with it faster



```
flights_random = sample_n(flights, 60000, replace = F)
```

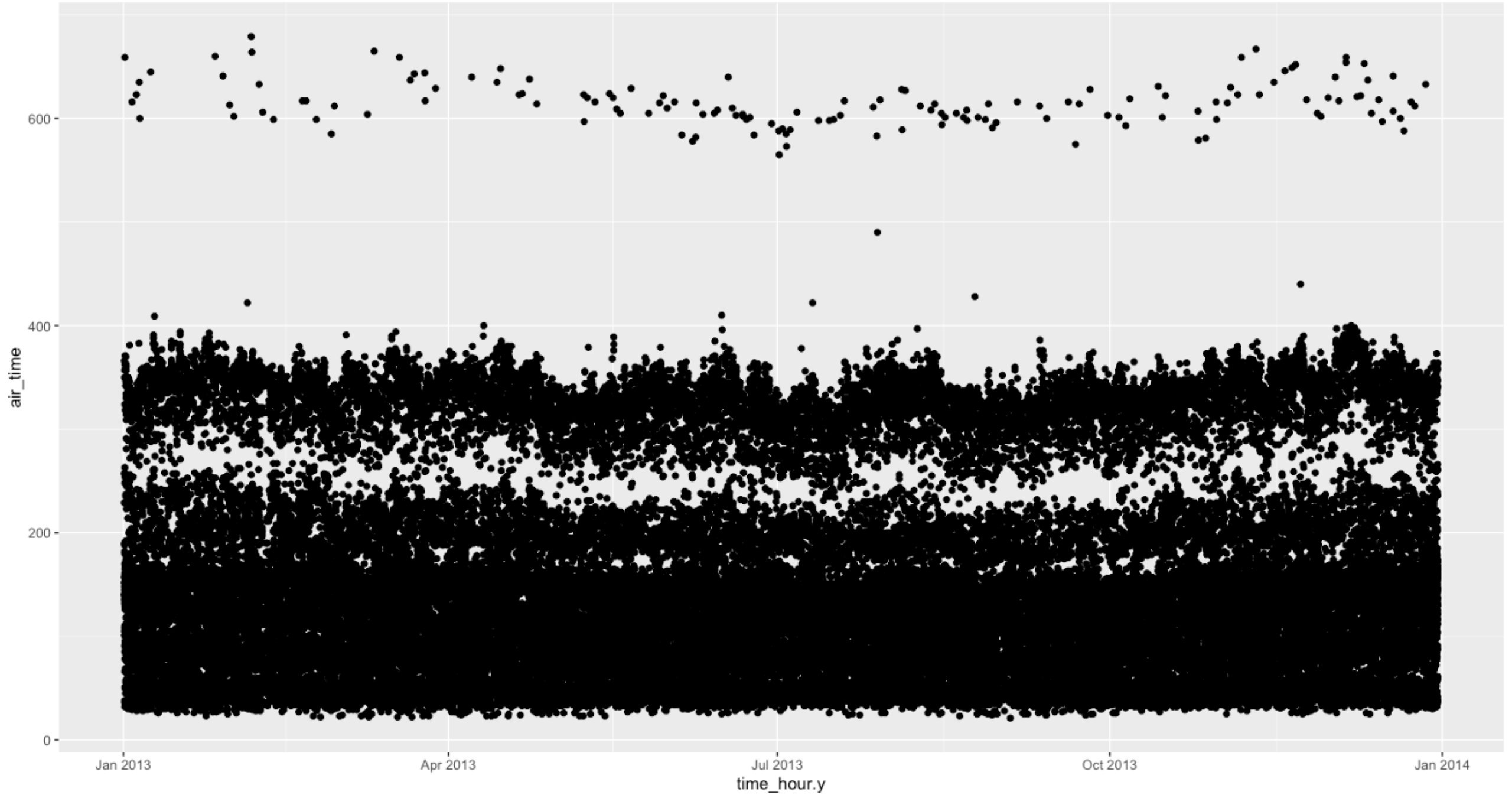


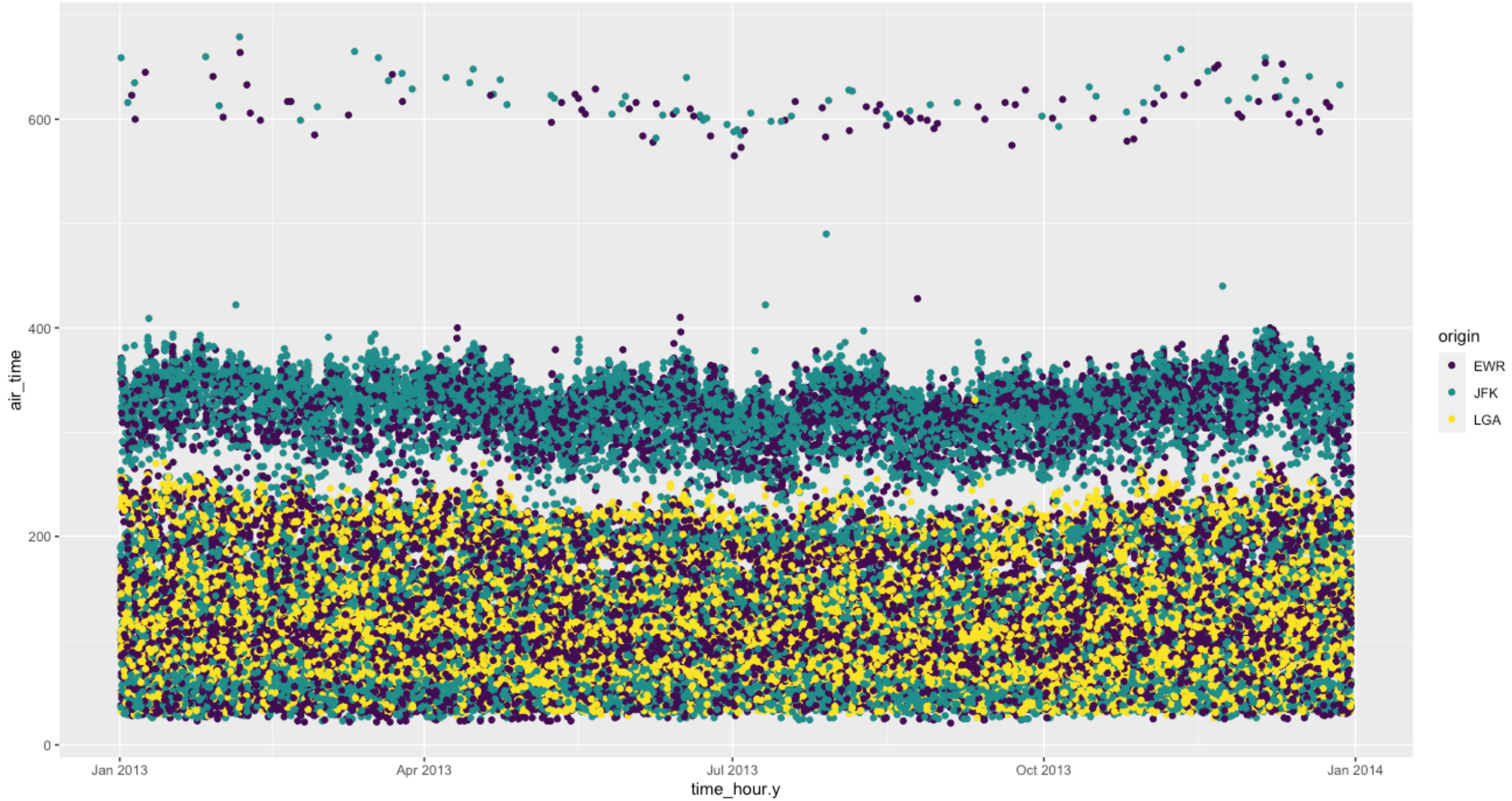


7 times slower to plot with full dataset

Unit: milliseconds

expr	min	lq	mean	median	uq	max	neval	cld
Full_data	3253.5995	3501.7875	3791.8927	3719.1188	3908.0333	5319.7226	100	b
Sub_sample	366.9913	396.7067	528.1458	541.8923	636.1587	917.0958	100	a


600
400
200
0
air_time
Jan 2013
Apr 2013
Jul 2013
Oct 2013
Jan 2014
time_hour.y


600
400
200
0
air_time
Jan 2013
Apr 2013
Jul 2013
Oct 2013
Jan 2014
time_hour.y
origin
EWR
JFK
LGA


9E
AA
AS
B6
DL
EV
F9
FL
HA
MQ
OO
UA
US
VX
WN
YV
air_time
0
200
400
600
Jan 2013
Apr 2013
Jul 2013
Oct 2013
Jan 2014
time_hour.y
origin
EWR
JFK
LGA

Useful links

- GitHub: https://datacarpentry.org/semester-biology/materials/git-in-30-minutes/

- Datasets for practice:
- https://www.kaggle.com/datasets