

# Scrollytelling and statistical inference

# Overview

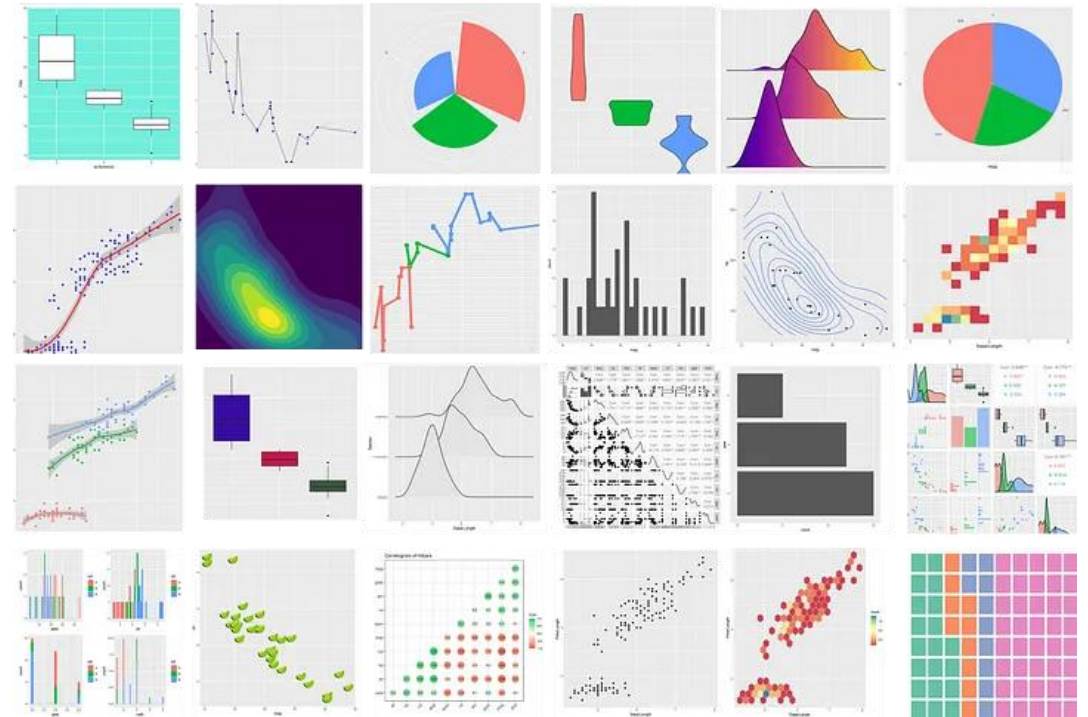
## A few last topics in ggplot

- Quick review of the grammar of graphics
- ggplot bonus features

## Scrollytelling with Closeroad

## If there is time

- A brief overview of statistical inference
- Hypothesis tests



# Homework 4

Create a scrollytelling webpage where you analyze a data set of your choosing

Homework 4 plan:

- Draft of homework due by 11pm on Monday July 28th
- Final version is due by 10pm on Wednesday July 30<sup>th</sup>

You will upload a quarto document and a html document to Canvas

How did homework 3 go?

# Class exam

Tuesday July 29<sup>th</sup> **in person** during regular class time

- Exam is on paper

If you have accommodations for extra time, just stay in the classroom and keep working on the exam



# Exam “cheat sheet”

You are allowed an exam “cheat sheet”

One page, single-side, that contains **only code and equations**

- No code comments allowed
- E.g., `plot(x, y)`

Cheat sheet must be on a regular 8.5 x 11 piece of paper

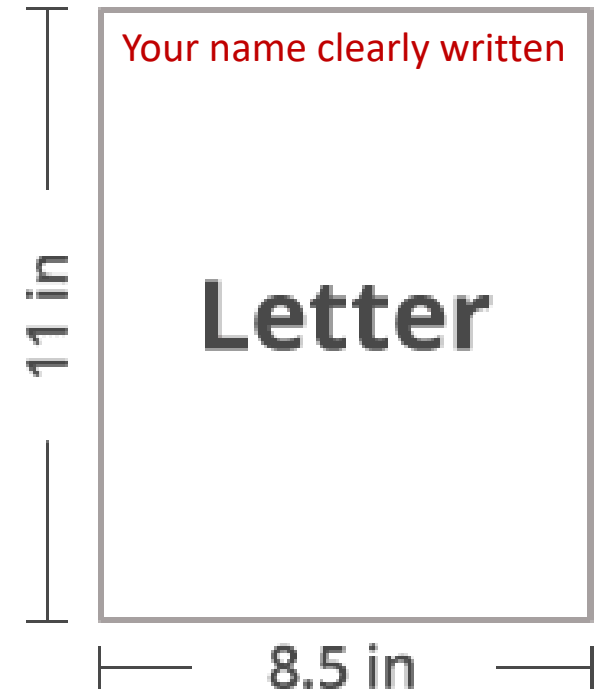
- Your name on the upper left side of the paper

I recommend making a typed list of all functions discussed in class and on the homework

- This will be useful beyond the exam

You must turn in your cheat sheet with the exam

- Failure to do so will result in a 20 point deduction



Questions?

Review and continuation of ggplot

# Review: Graphs are composed of...

**A Frame:** Coordinate system on which data is placed

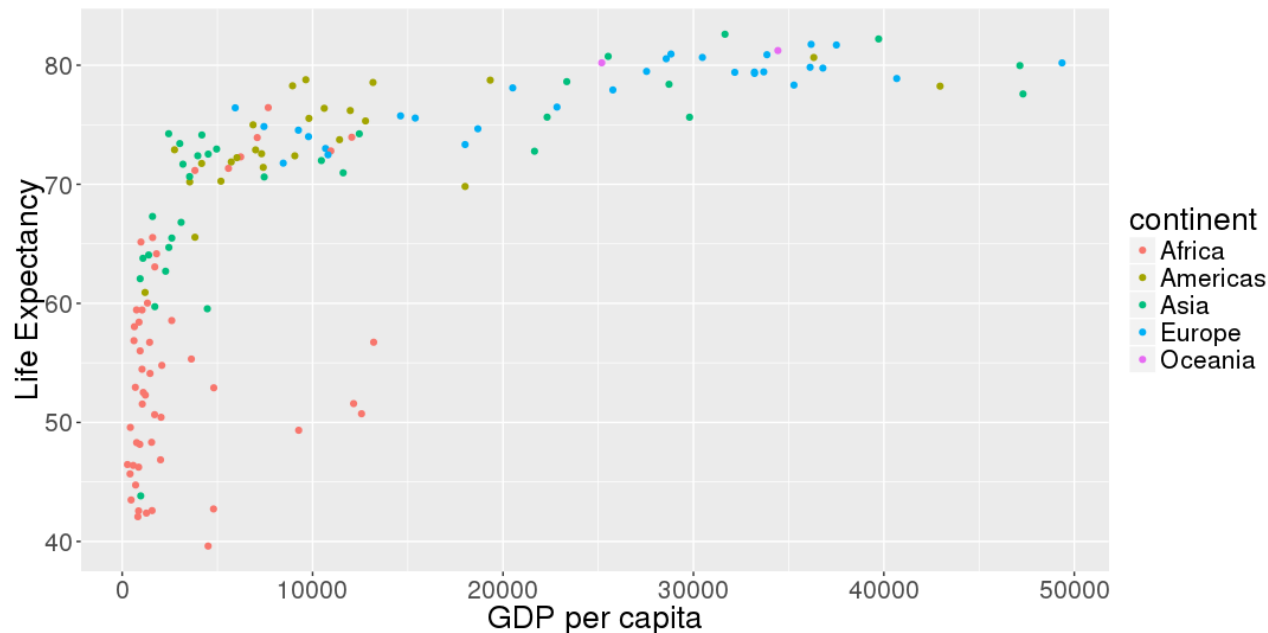
- `ggplot()` +

**Glyphs:** basic graphic unit representing cases or statistics

- Data is **mapped** onto these aesthetics such as: shape, color, size, etc. and/or aesthetics can be set to a fixed value
  - `geom_point(aes(x = gdpPercap, y = lifeExp, color = continent))` `geom_point(aes(x = gdpPercap, y = lifeExp), color = "red")`

**Scales and guides:** shows how to interpret axes and other properties of the glyphs

- `scale_x_continuous(trans = "log10")` `scale_color_brewer(type = "qua", palette = 2)`



# Review: Plots can also contain...

**Facets:** allows for multiple side-by-side graphs based on a categorical variable

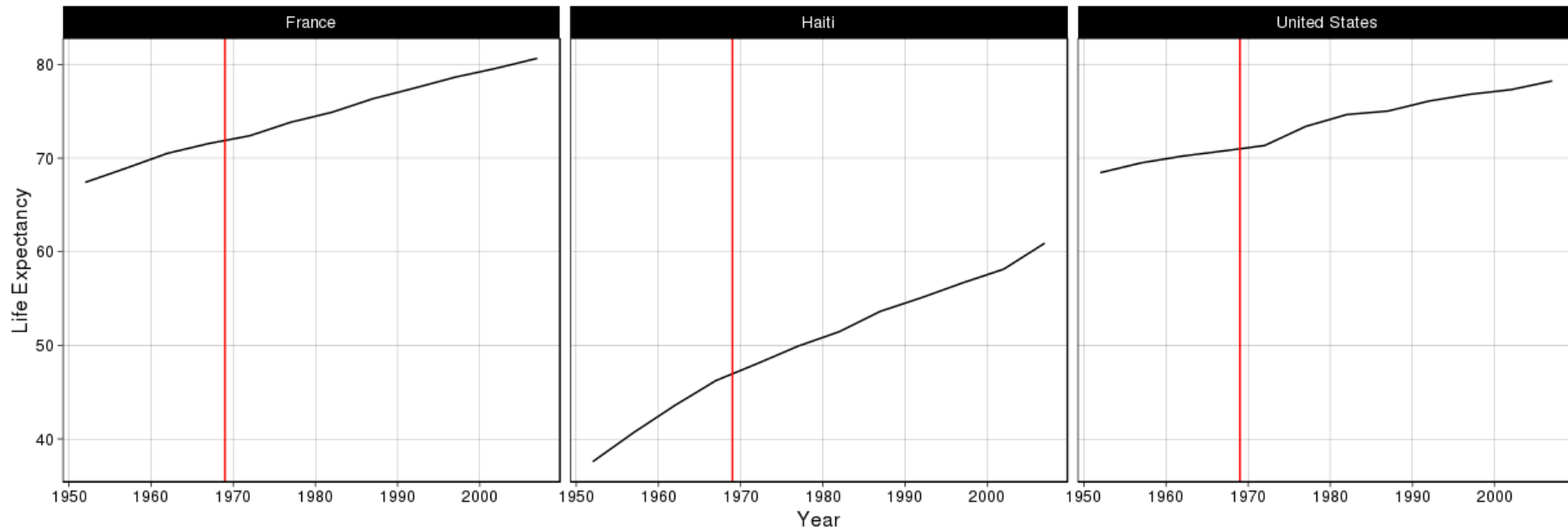
- `facet_wrap(~country)`

**Layers:** allows for more than one types of data to be mapped onto the same figure

- `geom_vline(xintercept = 1969, col = "red")`

**Theme:** contains finer points of display (e.g., font size, background color, etc.)

- `theme_wsj()`





# Questions?

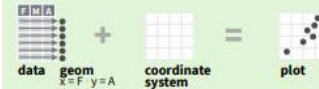
Let's discuss a few last features of ggplot by continuing with the class 6 code...

## Data visualization with ggplot2 : : CHEAT SHEET



### Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and **geoms**—visual marks that represent data points.



To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.



Complete the template below to build a graph.

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),  
    stat = <STAT>, position = <POSITION>) +  
  <COORDINATE_FUNCTION> +  
  <FACET_FUNCTION> +  
  <SCALE_FUNCTION> +  
  <THEME_FUNCTION>
```

required  
Not required, sensible defaults supplied

**ggplot**(data = mpg, aes(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

**last\_plot()** Returns the last plot.

**ggsave**("plot.png", width = 5, height = 5) Saves last plot as 5" x 5" file named "plot.png" in working directory. Matches file type to file extension.

### Aes

Common aesthetic values.

**color** and **fill** - string ("red", "#RRGGBB")

**linetype** - integer or string (0 = "blank", 1 = "solid", 2 = "dashed", 3 = "dotted", 4 = "dotdash", 5 = "longdash", 6 = "twodash")

**lineend** - string ("round", "butt", or "square")

**linejoin** - string ("round", "mitre", or "bevel")

**size** - integer (line width in mm)

**shape** - integer/shape name or a single character ("a")



### 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** <- ggplot(economics, aes(date, unemploy))  
**b** <- ggplot(seals, aes(x = long, y = lat))

**a + geom\_blank()** and **a + expand\_limits()**  
Ensure limits include values across all plots.

**b + geom\_curve**(aes(yend = lat + 1, xend = long + 1, curvature = 1) - x, yend, yend, alpha, angle, color, curvature, linetype, size)

**a + geom\_path**(lineend = "butt", linejoin = "round", linemitre = 1) - x, y, alpha, color, group, linetype, size

**a + geom\_polygon**(aes(alpha = 50)) - x, y, alpha, color, fill, group, subgroup, linetype, size

**b + geom\_rect**(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1) - xmin, xmax, ymax, ymin, alpha, color, fill, linetype, size)

**a + geom\_ribbon**(aes(ymin = unemploy - 900, ymax = unemploy + 900)) - x, ymax, ymin, alpha, color, fill, group, linetype, size

#### LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size

**b + geom\_abline**(aes(intercept = 0, slope = 1))  
**b + geom\_hline**(aes(yintercept = lat))  
**b + geom\_vline**(aes(xintercept = long))

**b + geom\_segment**(aes(yend = lat + 1, xend = long + 1))  
**b + geom\_spoke**(aes(angle = 1:1155, radius = 1))

#### ONE VARIABLE continuous

**c** <- ggplot(mpg, aes(hwy)); **c2** <- ggplot(mpg)

**c + geom\_area**(stat = "bin")  
x, y, alpha, color, fill, linetype, size

**c + geom\_density**(kernel = "gaussian")  
x, y, alpha, color, fill, group, linetype, size, weight

**c + geom\_dotplot**()  
x, y, alpha, color, fill

**c + geom\_freqpoly**()  
x, y, alpha, color, group, linetype, size

**c + geom\_histogram**(binwidth = 5)  
x, y, alpha, color, fill, linetype, size, weight

**c2 + geom\_qq**(aes(sample = hwy))  
x, y, alpha, color, fill, linetype, size, weight

#### discrete

**d** <- ggplot(mpg, aes(fill))

**d + geom\_bar**()  
x, alpha, color, fill, linetype, size, weight

#### TWO VARIABLES

**both continuous**

**e** <- ggplot(mpg, aes(cty, hwy))

**e + geom\_label**(aes(label = cty), nudge\_x = 1, nudge\_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

**e + geom\_point**()  
x, y, alpha, color, fill, shape, size, stroke

**e + geom\_quantile**()  
x, y, alpha, color, group, linetype, size, weight

**e + geom\_rug**(sides = "bl")  
x, y, alpha, color, linetype, size

**e + geom\_smooth**(method = lm)  
x, y, alpha, color, fill, group, linetype, size, weight

**e + geom\_text**(aes(label = cty), nudge\_x = 1, nudge\_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

#### one discrete, one continuous

**f** <- ggplot(mpg, aes(class, hwy))

**f + geom\_col**()  
x, y, alpha, color, fill, group, linetype, size

**f + geom\_boxplot**()  
x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

**f + geom\_dotplot**(binaxis = "y", stackdir = "center")  
x, y, alpha, color, fill, group

**f + geom\_violin**(scale = "area")  
x, y, alpha, color, fill, group, linetype, size, weight

#### both discrete

**g** <- ggplot(diamonds, aes(cut, color))

**g + geom\_count**()  
x, y, alpha, color, fill, shape, size, stroke

**e + geom\_jitter**(height = 2, width = 2)  
x, y, alpha, color, fill, shape, size

#### THREE VARIABLES

**sealsSz** <- with(seals, sqrt(delta\_long^2 + delta\_lat^2)); **l** <- ggplot(seals, aes(long, lat))

**l + geom\_contour**(aes(z = z))  
x, y, z, alpha, color, group, linetype, size, weight

**l + geom\_contour\_filled**(aes(fill = z))  
x, y, alpha, color, fill, group, linetype, size, subgroup

#### continuous bivariate distribution

**h** <- ggplot(diamonds, aes(carat, price))

**h + geom\_bin2d**(binwidth = c(0.25, 500))  
x, y, alpha, color, fill, linetype, size, weight

**h + geom\_density\_2d**()  
x, y, alpha, color, group, linetype, size

**h + geom\_hex**()  
x, y, alpha, color, fill, size

#### continuous function

**h** <- ggplot(economics, aes(date, unemploy))

**i + geom\_area**()  
x, y, alpha, color, fill, linetype, size

**i + geom\_line**()  
x, y, alpha, color, group, linetype, size

**i + geom\_step**(direction = "hv")  
x, y, alpha, color, group, linetype, size

#### visualizing error

**df** <- data.frame(grp = c("A", "B"), fit = 4.5, se = 1.2)  
**j** <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))

**j + geom\_crossbar**(fatten = 2) - x, y, ymax, ymin, alpha, color, fill, group, linetype, size

**j + geom\_errorbar**() - x, ymax, ymin, alpha, color, group, linetype, size, width  
Also **geom\_errorbarh**()

**j + geom\_linerange**()  
x, ymin, ymax, alpha, color, group, linetype, size

**j + geom\_pointrange**() - x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

#### maps

**data** <- data.frame(murder = USArrests\$Murder, state = tolower(rownames(USArrests)))

**map** <- map\_data("state")

**k** <- ggplot(data, aes(fill = murder))

**k + geom\_map**(aes(map\_id = state), map = map) + **expand\_limits**(x = map\$long, y = map\$lat)  
map\_id, alpha, color, fill, linetype, size



Scrollytelling with Closeroad

# Scrollytelling

“Scrollytelling” is a web design technique that uses visual and textual elements to tell a story as the reader scrolls through a page

- [Closeread examples](#), and [other examples](#)

We can create scrollytelling documents by using the Closeread custom format in Quarto

To use Closeread, you need to install the Quarto extension by running the following at the terminal tab:

```
quarto add qmd-lab/closeread
```

This create a directory called `_extensions` which has the code necessary to make a Closeread webpage



# Creating a Closeread Quarto document

To create a Closeread Quarto document, we start by specifying the appropriate quarto header

---

**title:** My Closeread story

**format:** closeread-html

---

```
1 ---
2 title: "class 25 notes and code"
3 format: pdf
4 ---
5
6
7 <!-- Please run the code in the R chunk below once. This will
8 install some packages and download data and images needed for
9 these exercises. -->
10
11
12 ```{r message=FALSE, warning = FALSE, echo = FALSE, eval = FALSE}
13 SDS230::download_data("IPED_salaries_2016.rda")
14 ```
15
16
17
18 ```{r setup, include=FALSE}
19 # install.packages("latex2exp")
20
21 library(latex2exp)
22 library(dplyr)
23 library(ggplot2)
24
25 #options(scipen=999)
26
27
28 knitr::opts_chunk$set(echo = TRUE)
29
30 set.seed(123)
31
32 ```
33
34
```

# Closeread components

Every Closeread document consists of **sections** of the document flagged as Closeread sections

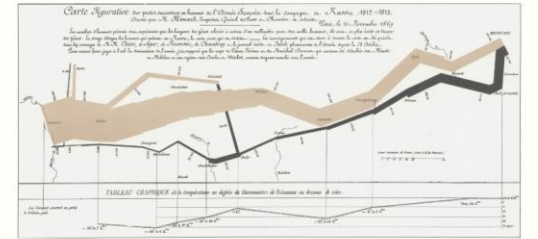
Within each Closeread section:

1. “**sticky**” **elements** are tagged to appear in the main column of the Closeread section
2. “**narration**” text appears in the “narrative column” and triggers the appearance of the sticky element

Narrative  
column

Sticky element

It may well  
be the best  
statistical  
graphic ever  
drawn.



# Closeread sections

Closeread sections specify what is part of document is a scrollingtelling story

- Elements outside of a section will appear as a normal text
  - (i.e., as a normal webpage/HTML document)

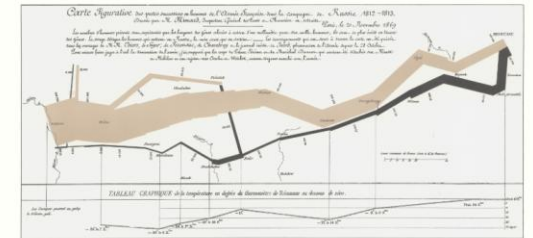
::::{.cr-section}

Text and images/code go in here...

::::

Closeread sections must have at least 3 colons.  
They can have more colons to make sections stand out.

It may well  
be the best  
statistical  
graphic ever  
drawn.



# Adding stickies

Elements that are pinned as one scrolls are called "stickies"

We can add stickies using:

```
...{#cr-stickyname}  
    sticky content
```

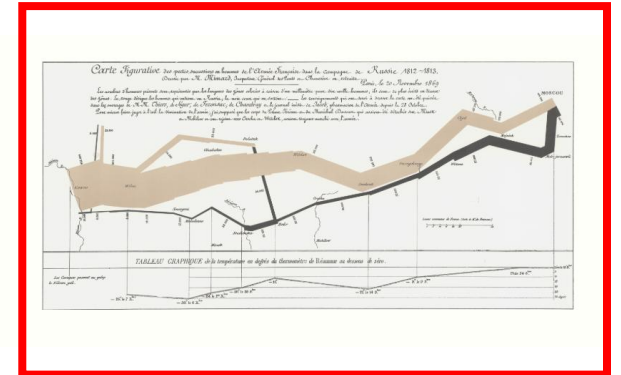
...

Must start with #cr-

```
...{#cr-myimage}  
    
```

...

It may well  
be the best  
statistical  
graphic ever  
drawn.



```
...{#cr-myplot}  
    ``{r}  
    hist(rnorm(15))
```

...

...



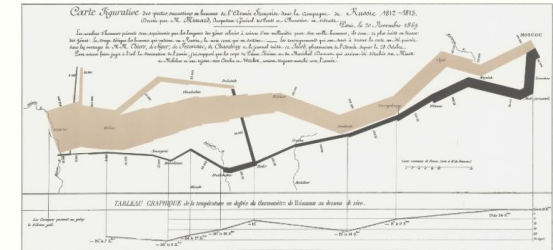
# Triggers

Any text (or other elements) in a Closeroad section that are not stickies will be placed in the narrative column

You can set a “narrative text” to trigger focus on a particular sticky by using: [@cr-mysticky](#)

It may well be the best statistical graphic ever drawn  
[@cr-myimage](#)

It may well be the best statistical graphic ever drawn.



Let's now show a histogram  
[@cr-myplot](#)



# Focus effects

We can add “focus effects” which can do the following:

- Scale, pan, or zoom in on images (or other elements)
- Highlight lines of text

Example:

This is where we load the library. `[@cr-dplyr]{highlight="1"}`  
This calculates summary statistics. `[@cr-dplyr]{highlight="2-3"}`

```
:::{#cr-dplyr}
```

```
``{r}
```

```
library(dplyr)
```

```
group_by(mtcars, am) |>
```

```
  summarize(avg_wt = mean(wt))
```

```
...
```

```
:::
```

This is where we load the library.

```
library(dplyr)
```

```
group_by(mtcars, am) |>  
  summarize(avg_wt = mean(wt))
```

```
# A tibble: 2 × 2
```

```
  am avg_wt
```

```
  <dbl> <dbl>
```

```
1     0   3.77
```

```
2     1   2.41
```

# Layouts

We can also change the layout of where the narration text and stickies appear by modifying the Quarto header

---

format:

closead-html:

cr-section:

layout: "overlay-center"

---

Options are:

- sidebar-left
- sidebar-right
- overlay-right
- overlay-right
- overlay-center

This is where we load the library.

```
library(dplyr)

group_by(mtcars, am) |>
  summarize(avg_wt = mean(wt))
```

```
# A tibble: 2 × 2
  am avg_wt
<dbl> <dbl>
1     0  3.77
2     1  2.41
```

# Let's try it in R!

`SDS111:download_class_code(7)`

Also see [this app](#) which can help create Closeroad documents

# A brief introduction to Statistical inference

# Statistical Inference

In **statistical inference** we use a sample of data to make claims about a larger population (or process)

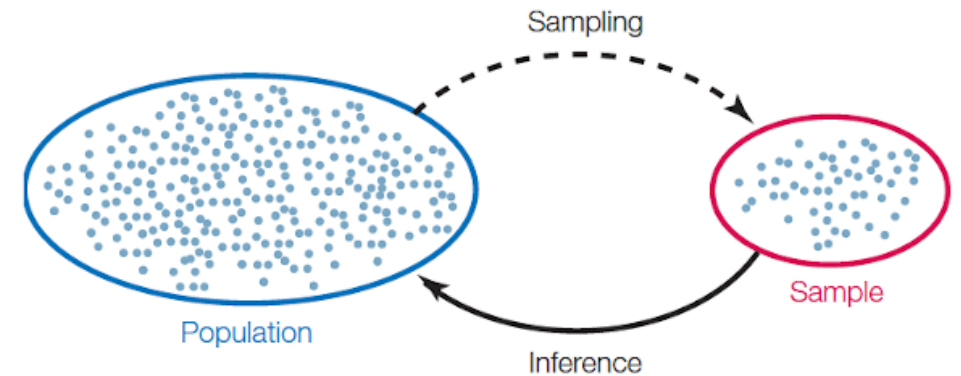
Examples:

## OkCupid data

- **Sample:** 59,000 OkCupid users from San Francisco
- **Population:** What is the population of interest?

## Billionaire data:

- **Sample:** 2,781 billionaires in 2024
- **Population:** What is the population of interest?



**Population:** all individuals/objects of interest

**Sample:** A subset of the population

# Terminology

**A parameter** is number associated with the population/process

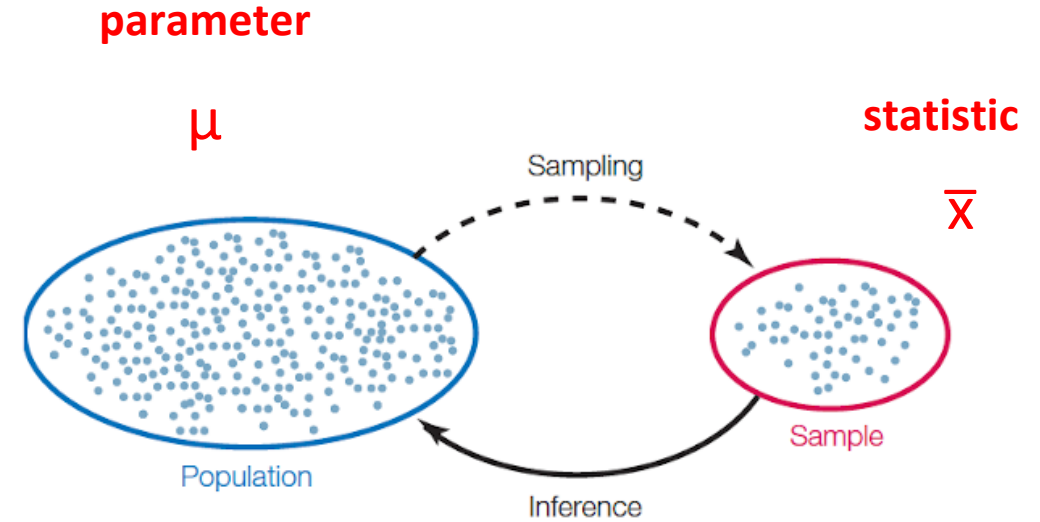
- e.g., population mean height of all men  $\mu$

**A statistic** is number calculated from the sample

- e.g., the height of a sample of  $n = 59,000$  men  $\bar{x}$

A statistic can be used as an estimate of a parameter

- A parameter is a single fixed value
- Statistics tend to vary from sample to sample



Example:

- Using the mean height of 59,000 men ( $\bar{x}$ ) to estimate the mean height of all men ( $\mu$ )

# Examples of parameters and statistics

	Sample statistic	Population parameter	Bechdel example
Mean	$\bar{x}$	$\mu$	<code>mean(profiles\$height)</code> $\bar{x} = 68.3$
Proportion	$\hat{p}$	$\pi$	<code>prop.table(table(profiles\$drinks))[6]</code> $\hat{p} = 0.697$
Correlation	$r$	$\rho$	<code>cor(min_temp, max_temp)</code> $r = 0.967$

# Sampling

**Simple random sample:** each member in the population is equally likely to be in the sample

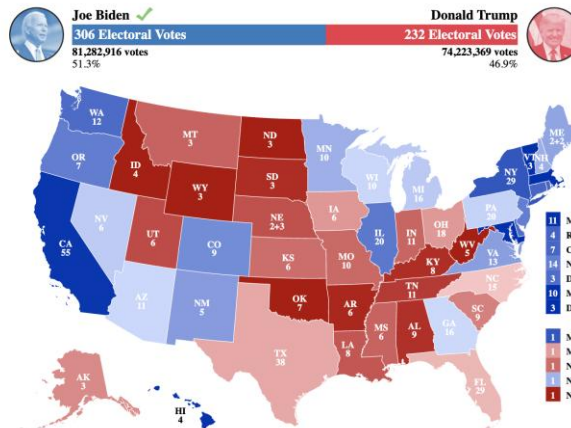
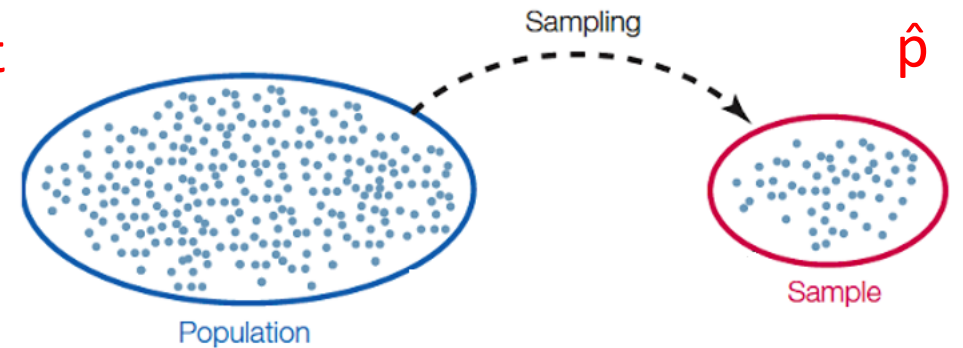
- Allows for generalizations to the population

parameter

$\pi$

statistic

$\hat{p}$



Polls of 1,000 voters:  $\hat{p}_{\text{Trump}}$

Vote on election day:  $\pi_{\text{Trump}}$



# Hypothesis tests

# Statistical tests (hypothesis test)

A **statistical test** uses data from a sample to assess a claim about a population (parameter)

Example 1: The average body temperature of humans is  $98.6^{\circ}$

How can we write this using symbols?

# Statistical tests (hypothesis test)


A **statistical test** uses data from a sample to assess a claim about a population (parameter)

Example 2: More than half of voters disapprove of Trump's job performance

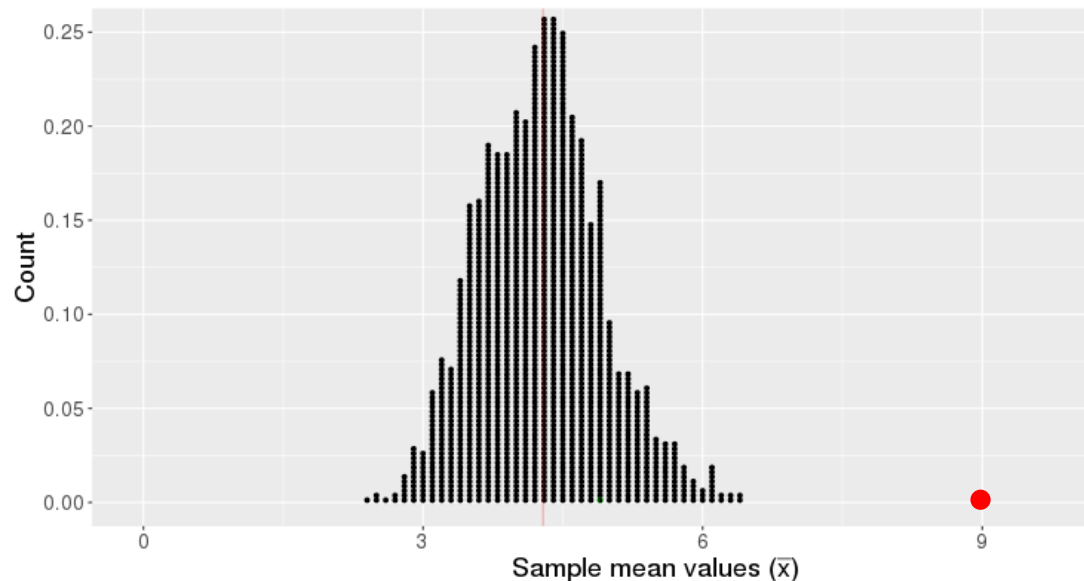
How can we write this using symbols?

# Basic hypothesis test logic

We start with a claim about a population parameter

- E.g.,  $\mu = 4$  

This claim implies we should get a certain distribution of statistics



If our observed statistic is highly unlikely, we reject the claim

# Example claims (hypotheses)

Let's see if we can write the following claims (hypotheses) using symbols

**Claim:** 88% of Yale students graduate within four years

**Claim:** The average age of a Yale undergraduate is 20

**Claim:** 70.7% of Yale classrooms have fewer than 20 students in attendance

# Testing claims (hypotheses)

Claim: 88% of Yale students graduate within four years

- $H: \pi = 0.88$
- To test this claim, we could randomly selected  $n = 100$  Yale graduates
- If we found the proportion that graduated in 4 years is  $\hat{p} = .80$ , would we believe the claim?

# Testing claims (hypotheses)

Claim: The average age of a Yale undergraduate is 20

- $H: \mu = 20$
- To test this claim, we could randomly selected  $n = 50$  Yale graduates
- If we found the average age of in our sample of students was  $\bar{x} = 20.2$ , would we believe the claim?

Visual hypothesis test



# Visual hypothesis test

In visual hypothesis tests, we create data visualizations to try to assess whether particular relationships exist in our data.

- One way this is done through a visual lineup



# Visual hypothesis test

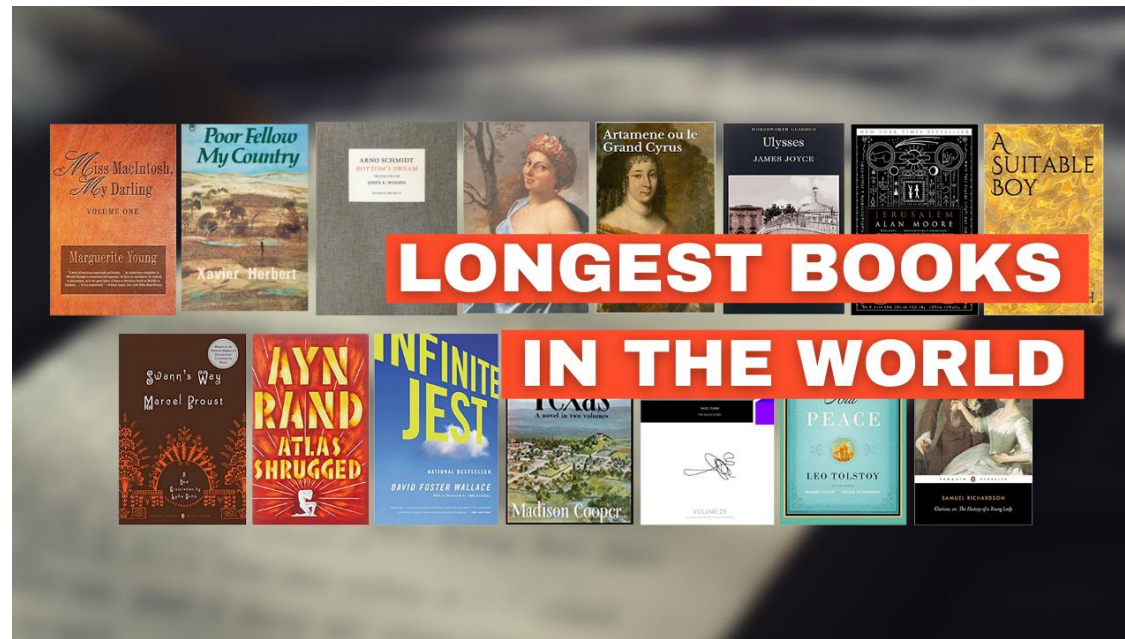
Which plot shows the true relationship between a car's weight and the number of miles per gallon a car gets?

Let's try it in R!

# Hypothesis tests for correlation

# Hypothesis tests for correlation

Is there a positive correlation between the number of pages in a book and the price of the book?



What is the population parameter and the statistic of interest?

# Hypothesis testing for correlation

1. Write down the null and alternative in symbols and words

## Null hypothesis:

- There is no correlation between book price and the number of pages

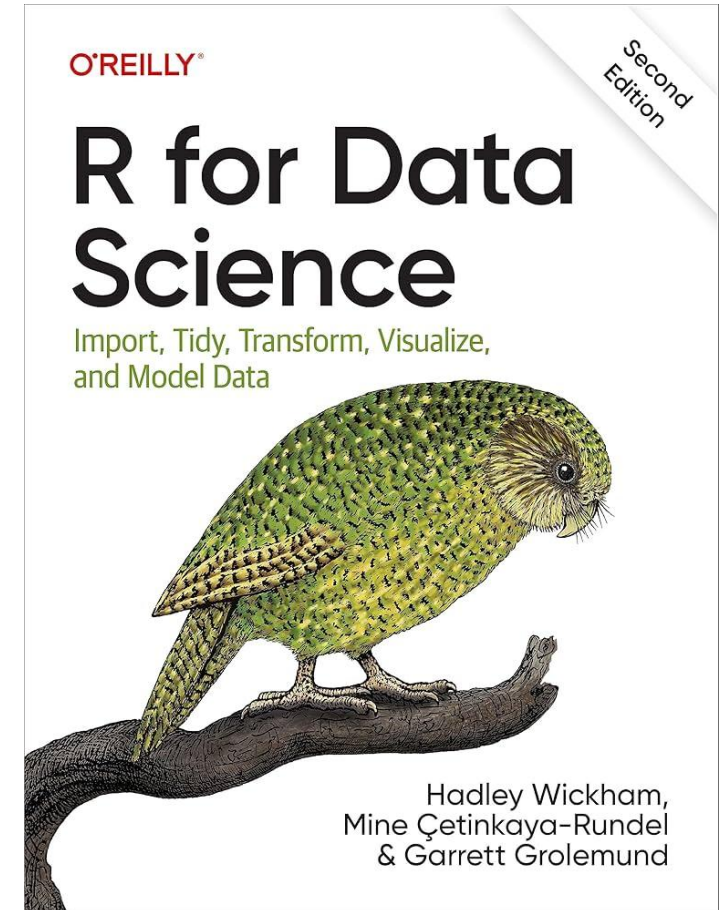
## Alternative hypothesis:

- There is a positive correlation between book price and the number of pages

## In symbols:

$$H_0: \rho = 0$$

$$H_A: \rho > 0$$



Has 548 pages

# Significance tests for correlation

Let's look at the books from Amazon.com

Title	List.Price	NumPages
1,001 Facts that Will Scare the S#*t Out of You	12.95	304
21: Bringing Down the House	15.00	273
100 Best-Loved Poems	1.50	96
1421: The Year China Discovered America	15.99	672

```
amazon = read_csv("amazon.csv")
```

# Try this in R!

## Step 2: What is the observed statistic?

- Also say whether you think you will be able to reject the null hypothesis based on a plot of your data

## Step 3: Create a distribution of null statistics

- To start with: how we can create one statistic consistent with the null hypothesis?
  - Hint: think about shuffling the data

## Step 4: What is the p-value that you get?

## Step 5: What decision would you make?



Has 1012 pages

# Confidence intervals



# Interval estimate based on a margin of error

Null hypothesis tests tell us if a particular parameter value is **implausible**

- E.g., in the average height of OkCupid users is  $\mu = 70$ "

An **interval estimate** give a range of **plausible** values for a population parameter

Example: 42% of American approve of Trump's job performance, plus or minus 3%

How do we interpret this?

Says that the population parameter  $\pi$  lies somewhere between 39% to 45%

- i.e., if they sampled all Americans the true population proportion would be likely be in this range

# Confidence Intervals

A **confidence interval** is an interval computed by a method that will contain the *parameter* a specified percent of times

- i.e., if the estimation were repeated many times, the interval will have the parameter  $x\%$  of the time

The **confidence level** is the percent of all intervals that contain the parameter

# Think ring toss...

Parameter exists in the ideal world

We toss intervals at it

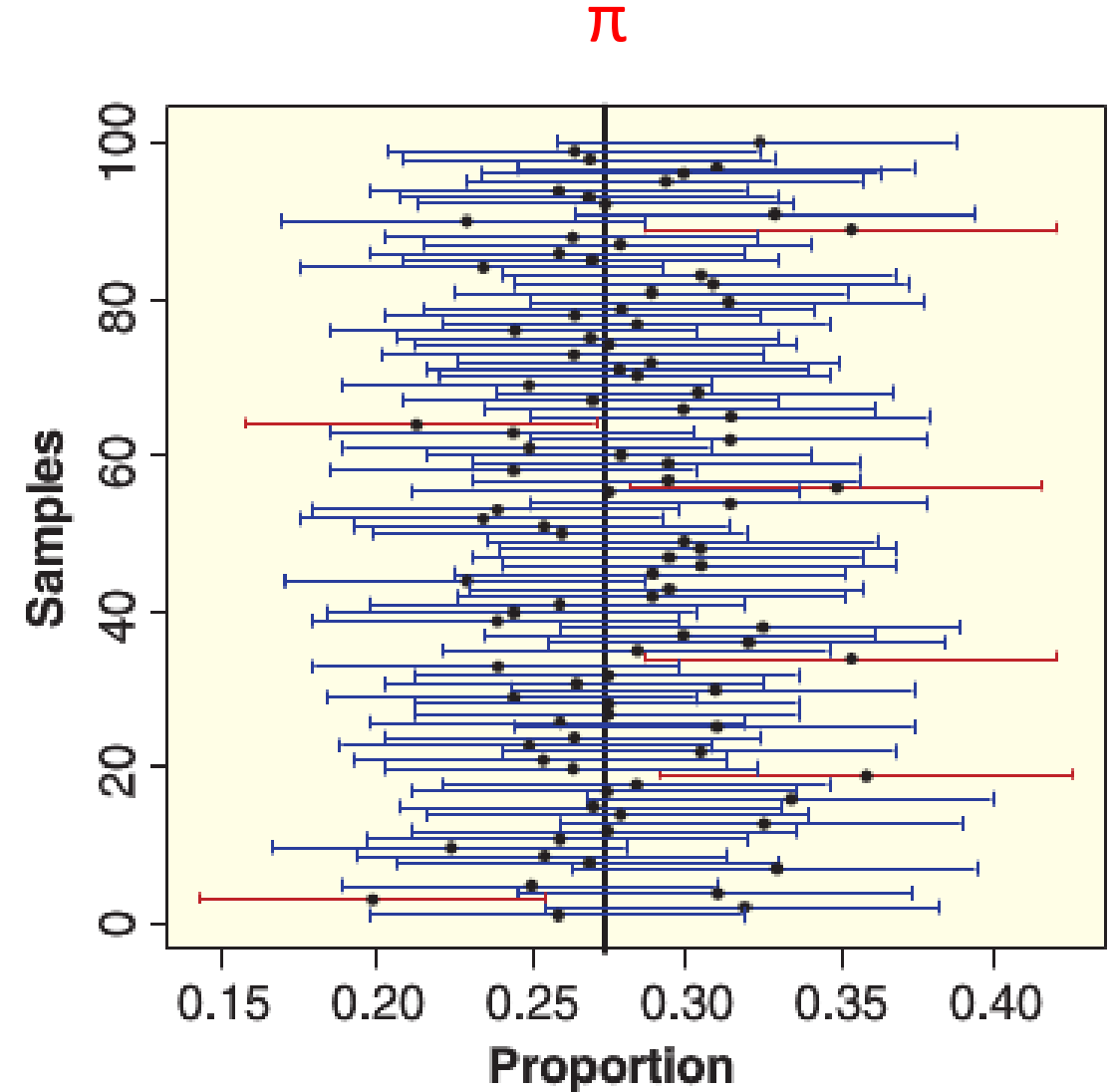
95% of those intervals capture the parameter



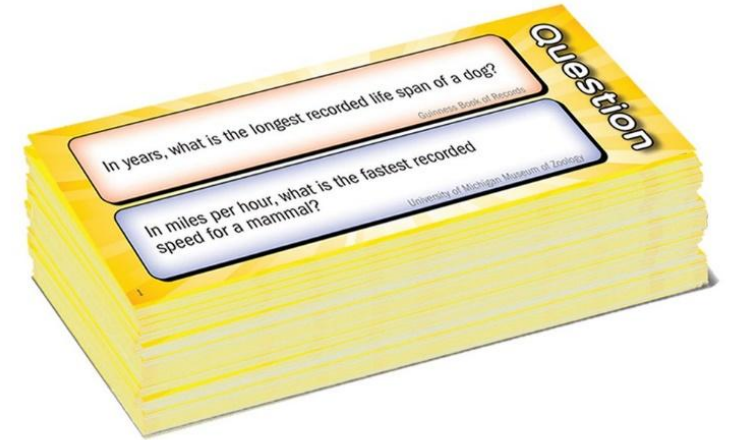
# Confidence Intervals

For a **confidence level** of 95%...

95% of the **confidence intervals** will have the parameter in them



# Wits and Wagers: 90% confidence interval estimator



I will ask 10 questions that have numeric answers

Please come up with a range of values that contains the true value in it for 9 out of the 10 questions

- i.e., be a 90% confidence interval estimator

# Note

For any given confidence interval we compute, we don't know whether it has really captured the parameter

But we do know that if we do this 100 times, 90 of these intervals will have the parameter in it

(for a 90% confidence interval)

# 100% confidence intervals

There is a tradeoff between the **confidence level** (percent of times we capture the parameter) and the **confidence interval size**



# Note

For any given confidence interval we compute, we don't know whether it has really captured the parameter

But we do know that if we do this 100 times, 90 of these intervals will have the parameter in it

(for a 90% confidence interval)



# Constructing confidence intervals

There are several methods that can be used to construct confidence intervals including

- “Parametric methods” that use probability functions
  - E.g., confidence intervals based on the normal distribution
- A “bootstrap method” where data is resampled from our original sample to approximate a sampling distribution

To learn more about these methods, take Introductory Statistics!