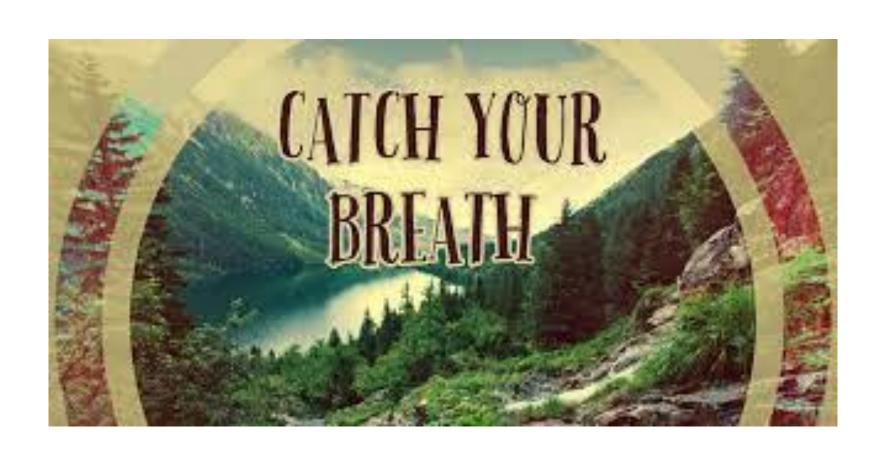
# Review



### Overview

Review of ggplot

Review of other class material

If there is time/interest

• Bonus features of ggplot: special geoms, animation, interactive graphics

### Announcements

### Midterm exam is on Thursday

- Bring a pen and a pencil
- One page (2 sides) of notes on code and equations
  - You will turn in this page of notes with your exam

### Office hours this week

- I will have my office hours on zoom and in person on Wednesday at 11:30am
- No TA office hours

Review of the grammar of graphics and ggplot

# The grammar of graphics

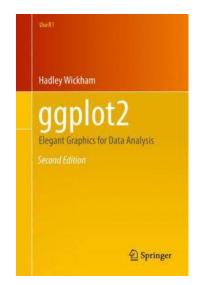
Leland Wilkinson noticed similarities between many graphs and tried to generate a 'grammar' that could be used to express a graph

• i.e., a list elements that can be combined together to create a graph

Statistics and Computing
Leland Wilkinson

The Grammar of Graphics
Second Edition

Hadley Wickham implemented these ideas in R in the ggplot2 package



# 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. aesthetics can be set to a fixed value and/or

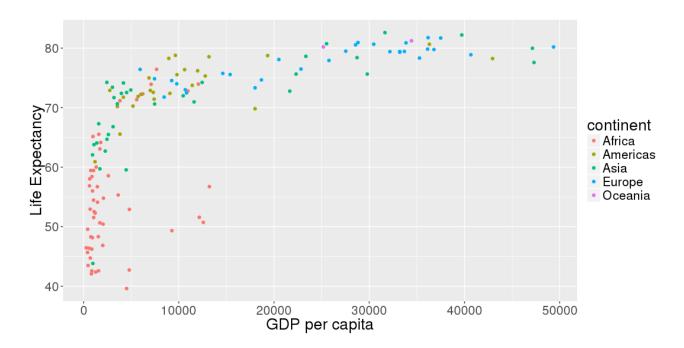
```
geom_point(aes(x = gdpPercap, y = lifeExp, color = continent))
```

geom\_point(aes(x = gdpPercap, y = lifeExp), color =

**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)



### Plots can also contain...

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

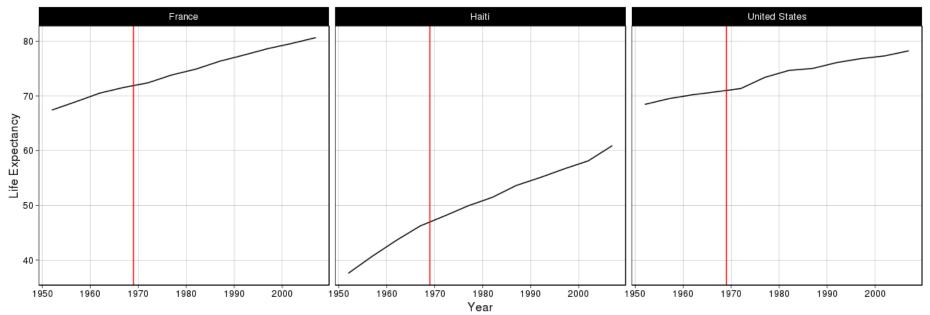
facet\_wrap(~country)

**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?

# ggplot2 cheat sheet

### 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>

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")

Studio

size - integer (line width in mm) 0 1 2 3 4 5 6 7 8 9 00 11 12 □○△+×◇▽宮\*◆●苅田 shape - integer/shape name or 13 14 15 16 17 18 19 20 21 22 23 24 25 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, xend, y, yend, alpha, angle, color, curvature, linetype, size

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

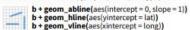
a + geom\_polygon(aes(alpha = 50)) - x, y, alpha, color, fill, group, subgroup, linetype, size

b + geom\_rect(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1)) - xmax, xmin, 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



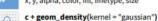
b + geom\_segment(aes(vend = lat + 1, xend = long + 1)) b + geom\_spoke(aes(angle = 1:1155, radius = 1))

### ONE VARIABLE continuous

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



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



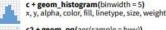
x, y, alpha, color, fill, group, linetype, size, weight



x, y, alpha, color, fill



x, y, alpha, color, group, linetype, size



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

d <- ggplot(mpg, aes(fl))



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

### TWO VARIABLES

both continuous e <- ggplot(mpg, aes(cty, hwy))



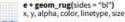
e + geom\_label(aes(label = cty), nudge\_x = 1, nudge\_y = 1) - x, y, label, alpha, angle, color. family, fontface, hjust, lineheight, size, vjust



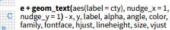
x, y, alpha, color, fill, shape, size, stroke



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



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



### one discrete, one continuous f <- ggplot(mpg, aes(class, hwy))





x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight



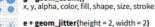
f + geom\_violin(scale = "area") x, y, alpha, color, fill, group, linetype, size, weight

### both discrete

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



g + geom\_count()





### 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()

ggplot.



x, y, alpha, color, group, linetype, size



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

### continuous function

i <- ggplot(economics, aes(date, unemploy))



i + geom area()

x, y, alpha, color, fill, linetype, size



x, y, alpha, color, group, linetype, size

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

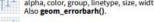
### visualizing error

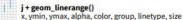
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2) j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))



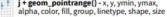
j + geom\_crossbar(fatten = 2) - x, y, ymax, ymin, alpha, color, fill, group, linetype, size







j + geom\_pointrange() - x, y, ymin, ymax,



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

### THREE VARIABLES

seals\$z <- with(seals, sqrt(delta\_long^2 + delta\_lat^2)); I <- ggplot(seals, aes(long, lat))



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

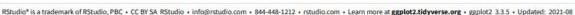
I + geom\_contour\_filled(aes(fill = z))



l + geom\_raster(aes(fill = z), hjust = 0.5, viust = 0.5, interpolate = FALSE) x, y, alpha, fill



l + geom\_tile(aes(fill = z)) x, y, alpha, color, fill, linetype, size, width



### Review

Let's create groups of ~4 people and spend ~6 minutes:

1. Go around a mention any questions you have about the material covered in class

2. See if you can answer questions that others have

3. Write down questions that your group could not answer and we can go over them as a class

# ggplot bonus features

# Additional geometries: emoGG

There are also additional packages that add more geoms

```
> library(emoGG)
```

```
> ggplot(mtcars, aes(wt, mpg)) + geom_emoji(emoji="1f697")
```

# Plotly – interactive plots

### Animation

We can create animated images (gifs) using the gganimate package

- > library(gganimate)
- > library(gapminder)

In the gapminder video, Hans had the following mapping:

- x = gpd per capita
- y = life expectancy
- size = population
- color = continent
- frame = year

# Recreating gapminder plot

```
ggplot(gapminder, aes(gdpPercap, lifeExp, size = pop)) +
 geom point(alpha = 0.7, show.legend = FALSE) +
 scale x \log 10() +
 facet_wrap(~continent) +
  # Here comes the gganimate specific bits
  labs(title = 'Year: {frame time}',
       x = 'GDP per capita', y = 'life expectancy') +
  transition time(year) +
  ease aes('linear')
```

## Data visualization

What interesting data visualizations did you find?

