Data Visualisation

Dr. Zahid Asghar
Professor of Economics
zasghar@qau.edu.pk
zahedasghar
zahidasghar.com

Graphics and Statistics

Our Data Source

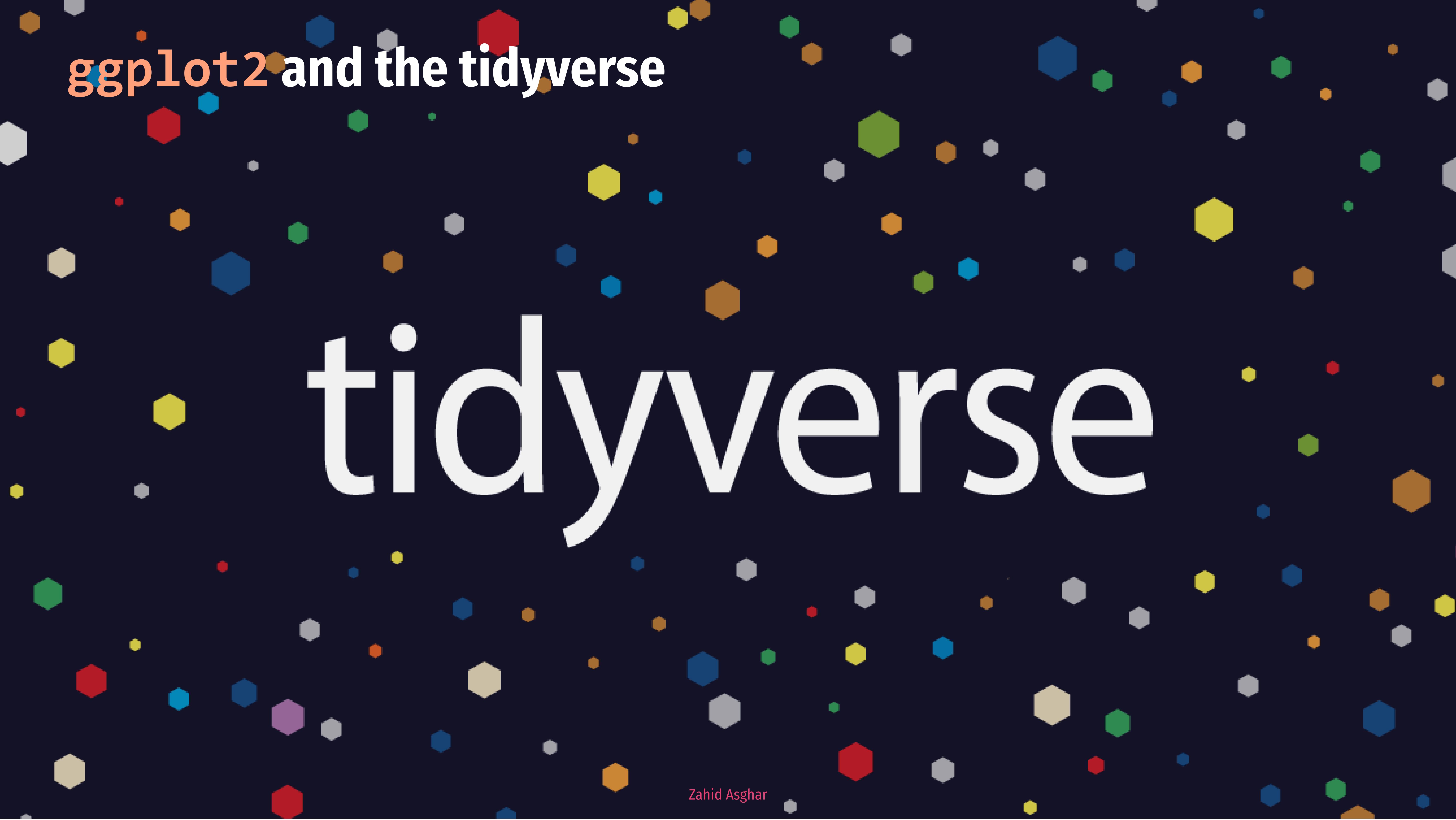
- For our examples, we’ll use a dataset `mpg` from the `ggplot2` library

```r
library(ggplot2)

head(mpg)
```

```
# A tibble: 6 × 11
  manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
  <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
1 audi         a4      1.8  1999     4 auto(l5)   f        18    29 p     compa…
2 audi         a4      1.8  1999     4 manual(m5) f        21    29 p     compa…
3 audi         a4      2    2008     4 manual(m6) f        20    31 p     compa…
4 audi         a4      2    2008     4 auto(av)   f        21    30 p     compa…
5 audi         a4      2.8  1999     6 auto(l5)   f        16    26 p     compa…
6 audi         a4      2.8  1999     6 manual(m5) f        18    26 p     compa…
```

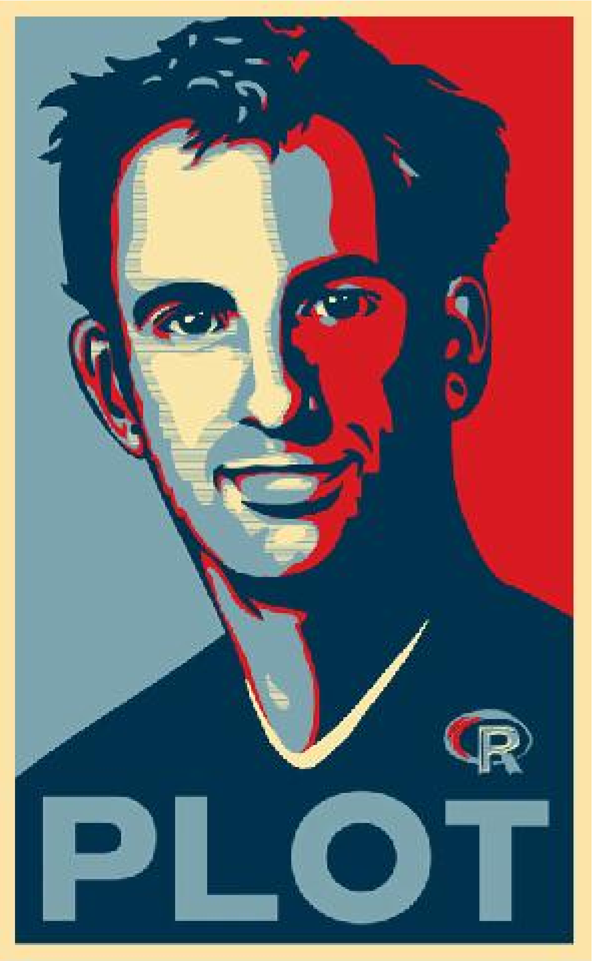
ggplot2 and the tidyverse



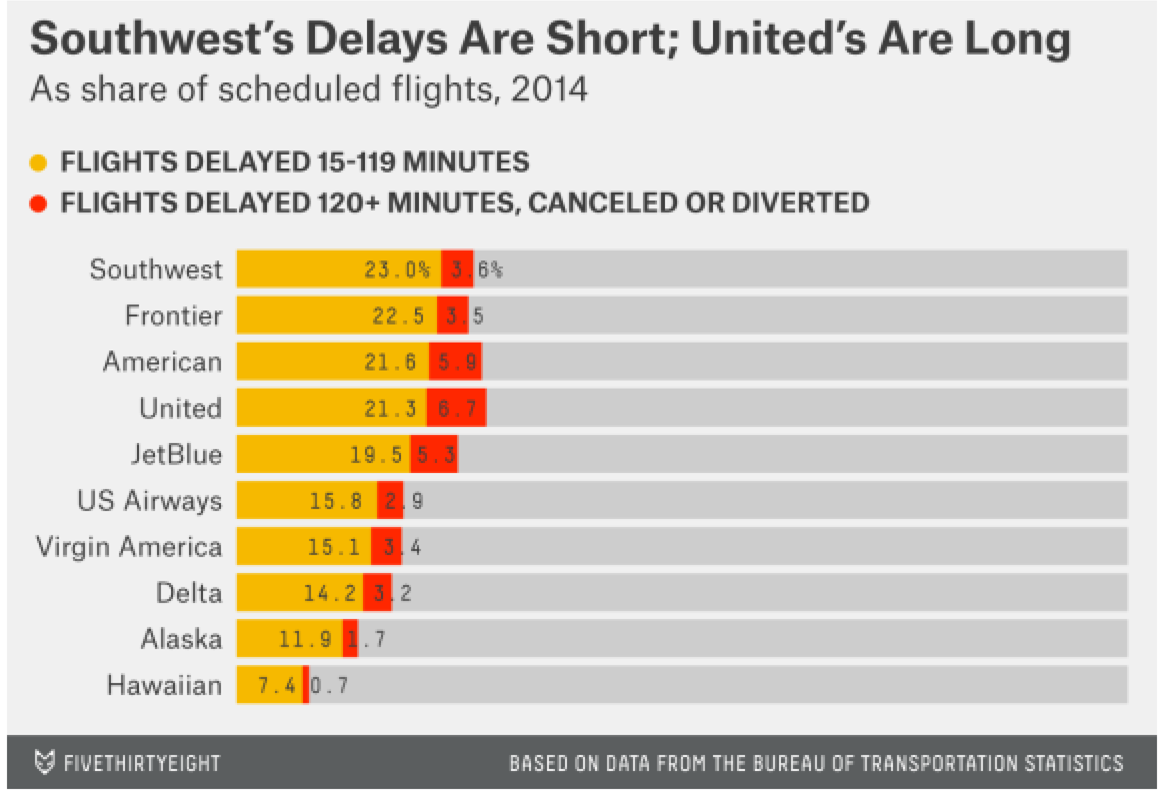
Zahid Asghar

ggplot2

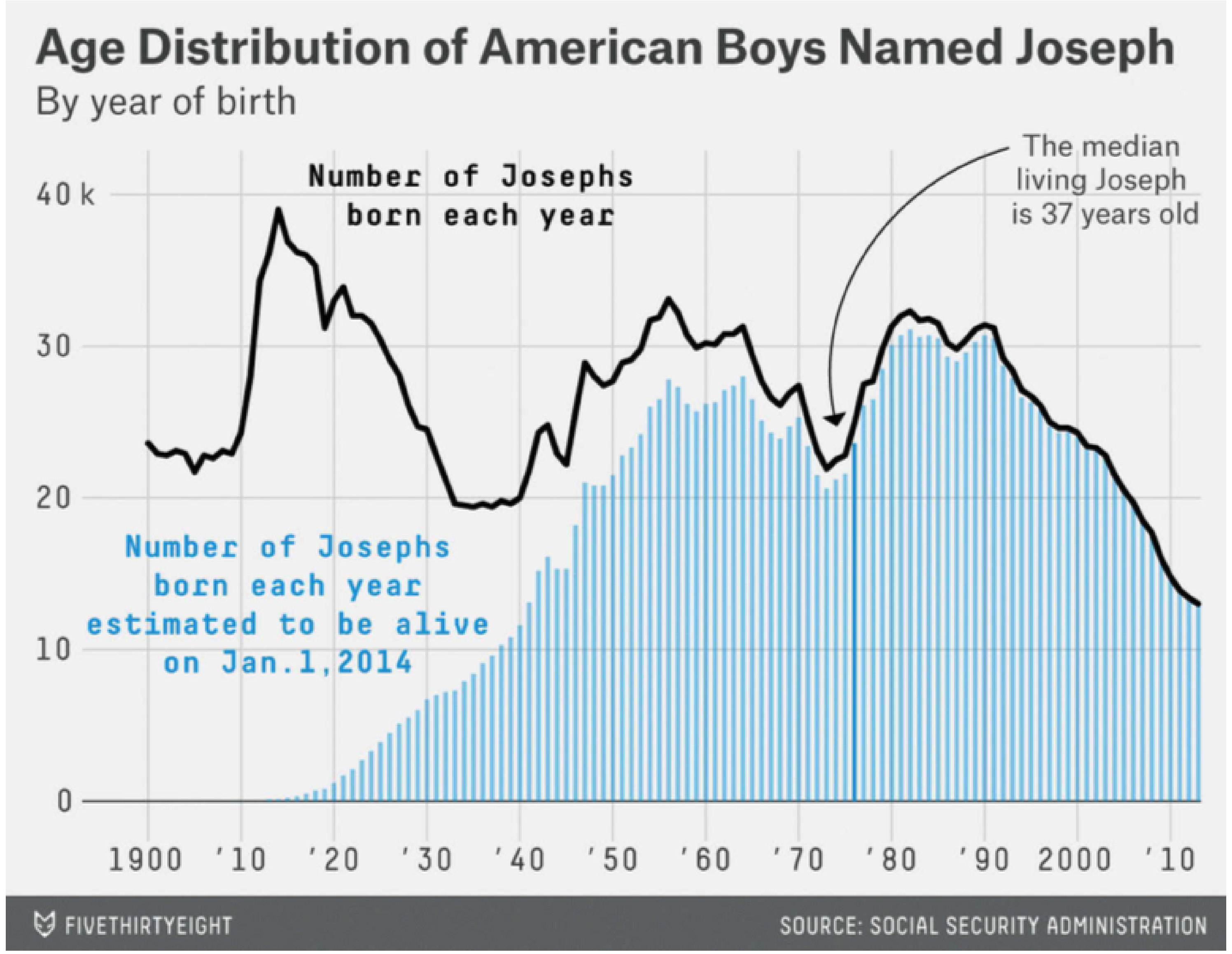
- `ggplot2` is perhaps the most popular package in `R` and a core element of the `tidyverse`
- `gg` stands for a **grammar of graphics**
- Very powerful and beautiful graphics, very customizable and reproducible, but requires a bit of a learning curve
- All those "cool graphics" you've seen in the New York Times, fivethirtyeight, the Economist, Vox, etc use the grammar of graphics



ggplot: All Your Figure are Belong to Us

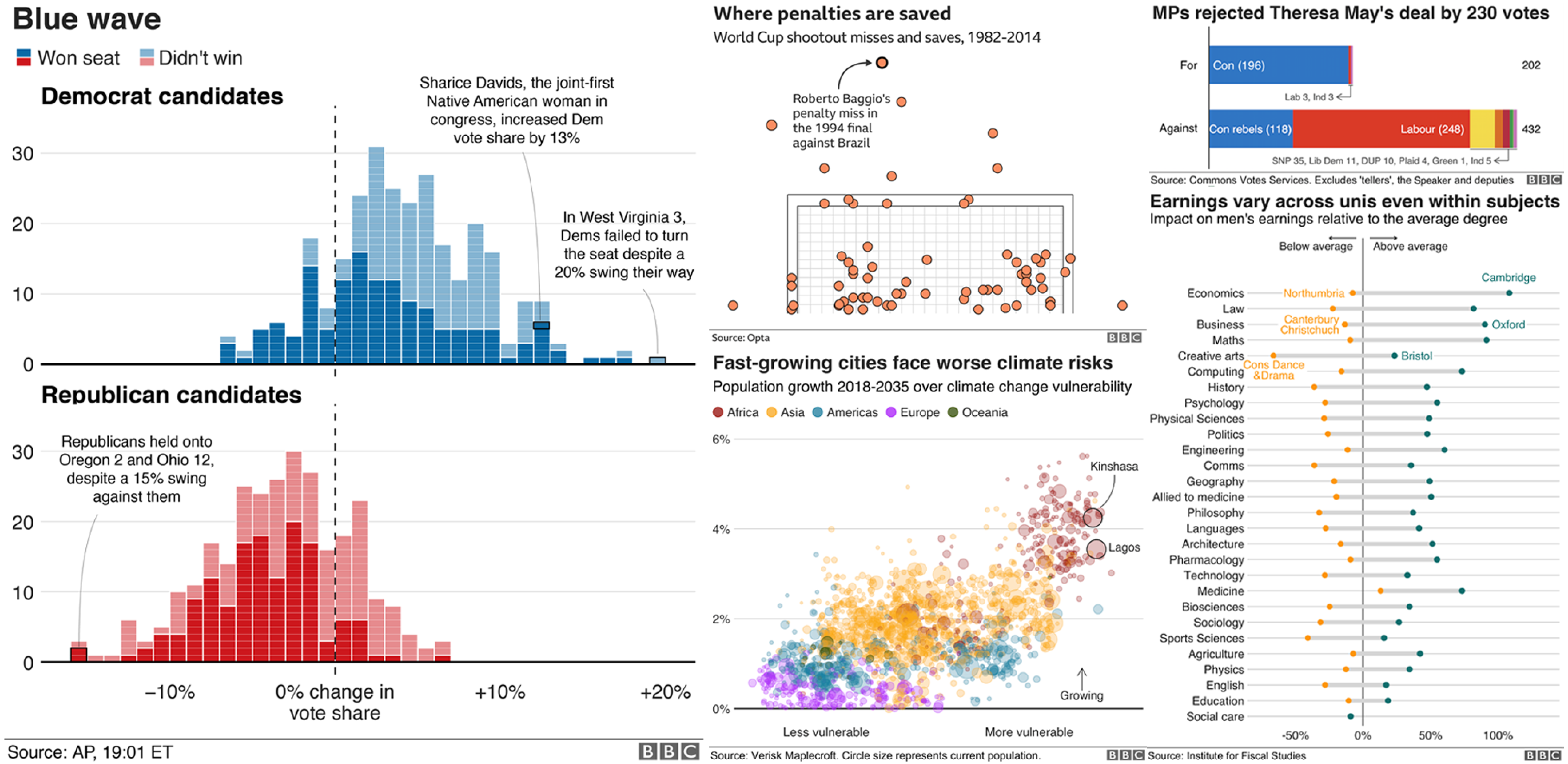


Source: fivethirtyeight



Source: fivethirtyeight

ggplot: All Your Figure are Belong to Us



Blue wave

Source: AP, 19:01 ET

Where penalties are saved

World Cup shootout misses and saves, 1982-2014

Source: Opta

Fast-growing cities face worse climate risks

Population growth 2018-2035 over climate change vulnerability

Source: Verisk Maplecroft. Circle size represents current population.

MPs rejected Theresa May's deal by 230 votes

Source: Commons Votes Services. Excludes 'tellers', the Speaker and deputies

Earnings vary across unis even within subjects

Impact on men's earnings relative to the average degree

Source: Institute for Fiscal Studies

Source: BBC's bbplot

Why Go gg?

Hadley Wickham

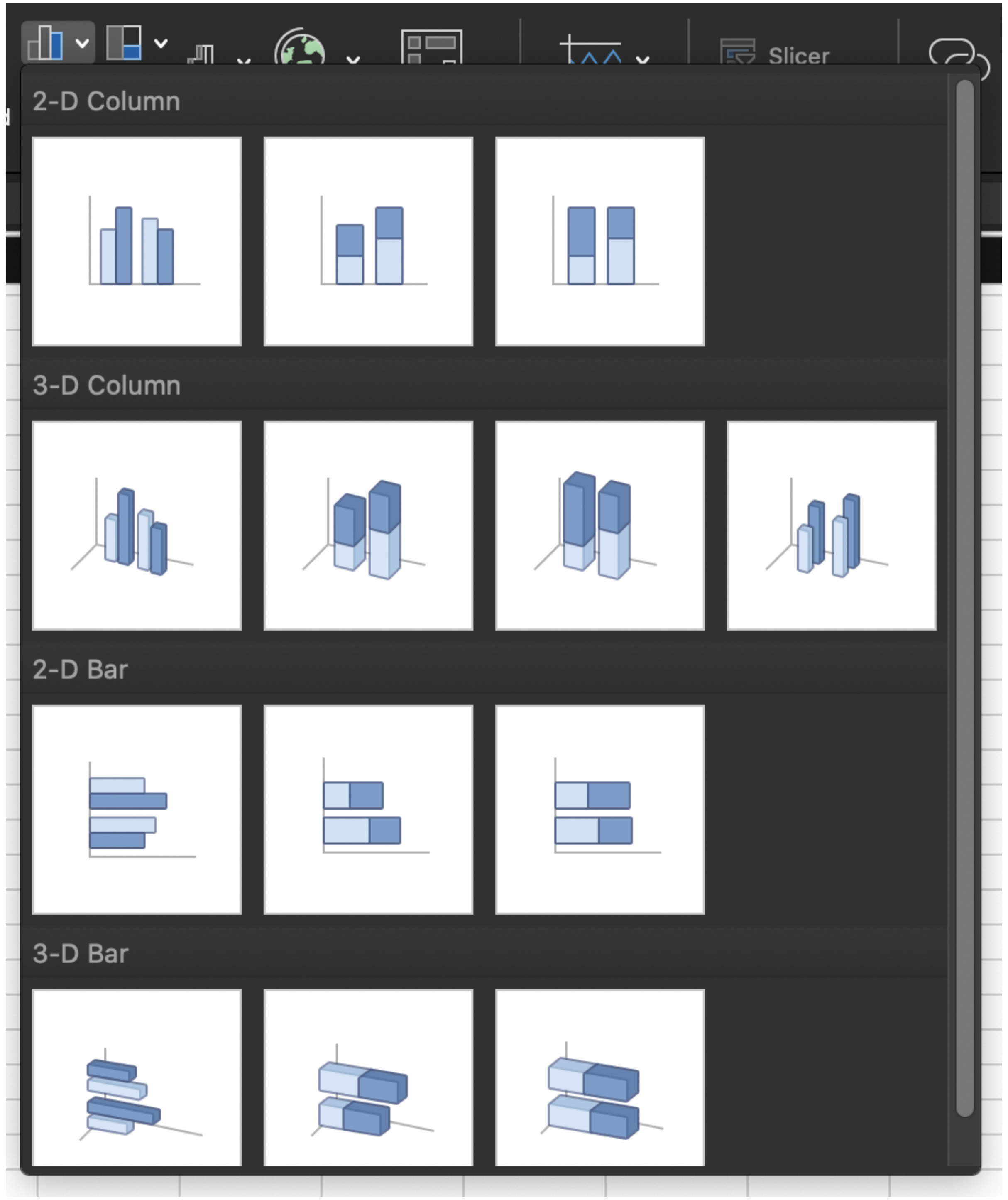
Chief Scientist, R Studio

> “The transferrable skills from ggplot2 are not the idiosyncracies of plotting syntax, but a powerful way of thinking about visualisation, as a way of **mapping between variables and the visual properties of geometric objects** that you can perceive.”

Source

The Grammar of Graphics (gg)

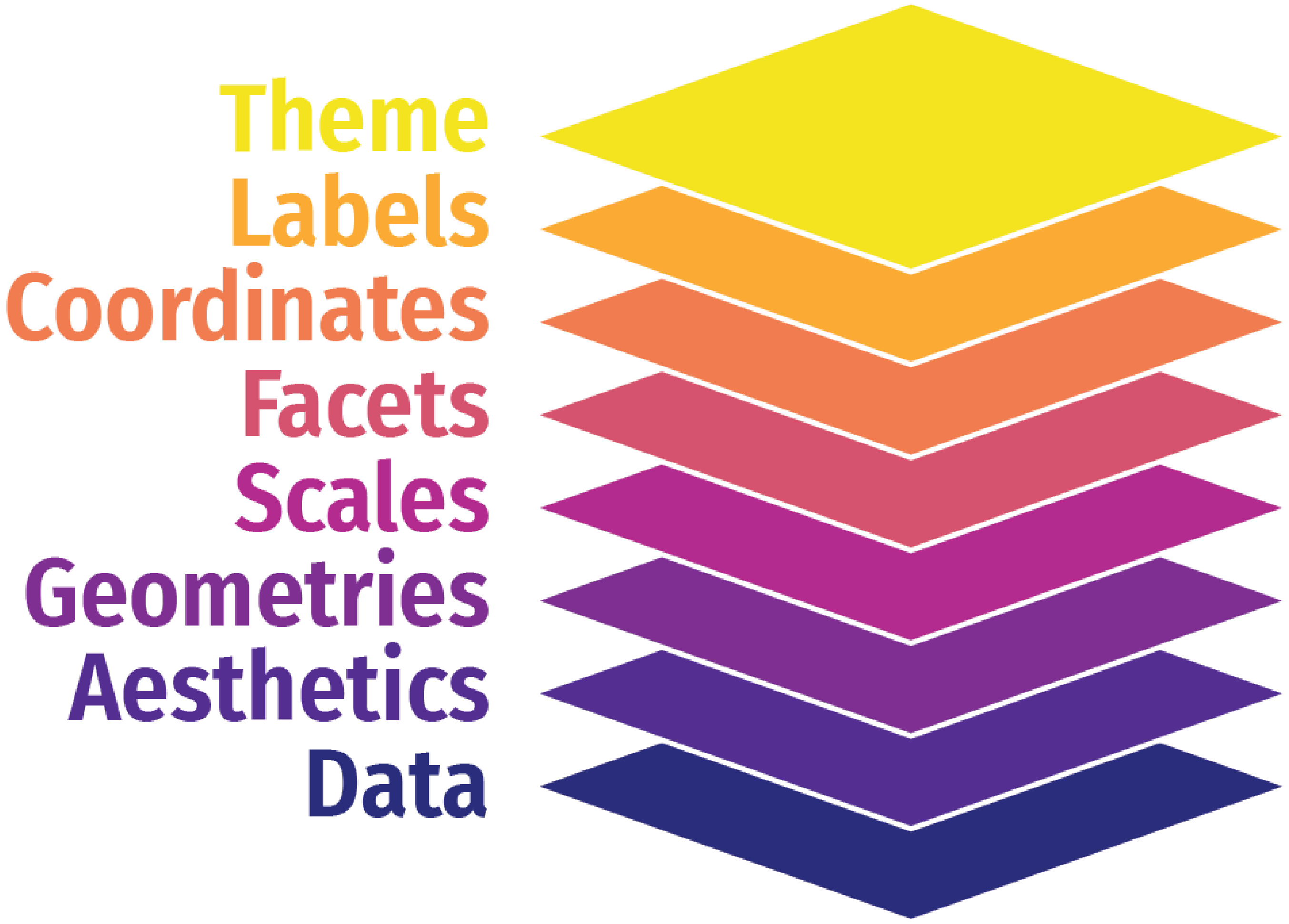
- This is a true *grammar*
- We *don't* talk about specific chart **types**
  - That you have to hunt through in Excel and reshape your data to fit it
- Instead we talk about specific chart **components**



The Grammar of Graphics (gg) I

- Any graphic can be built from the same components:

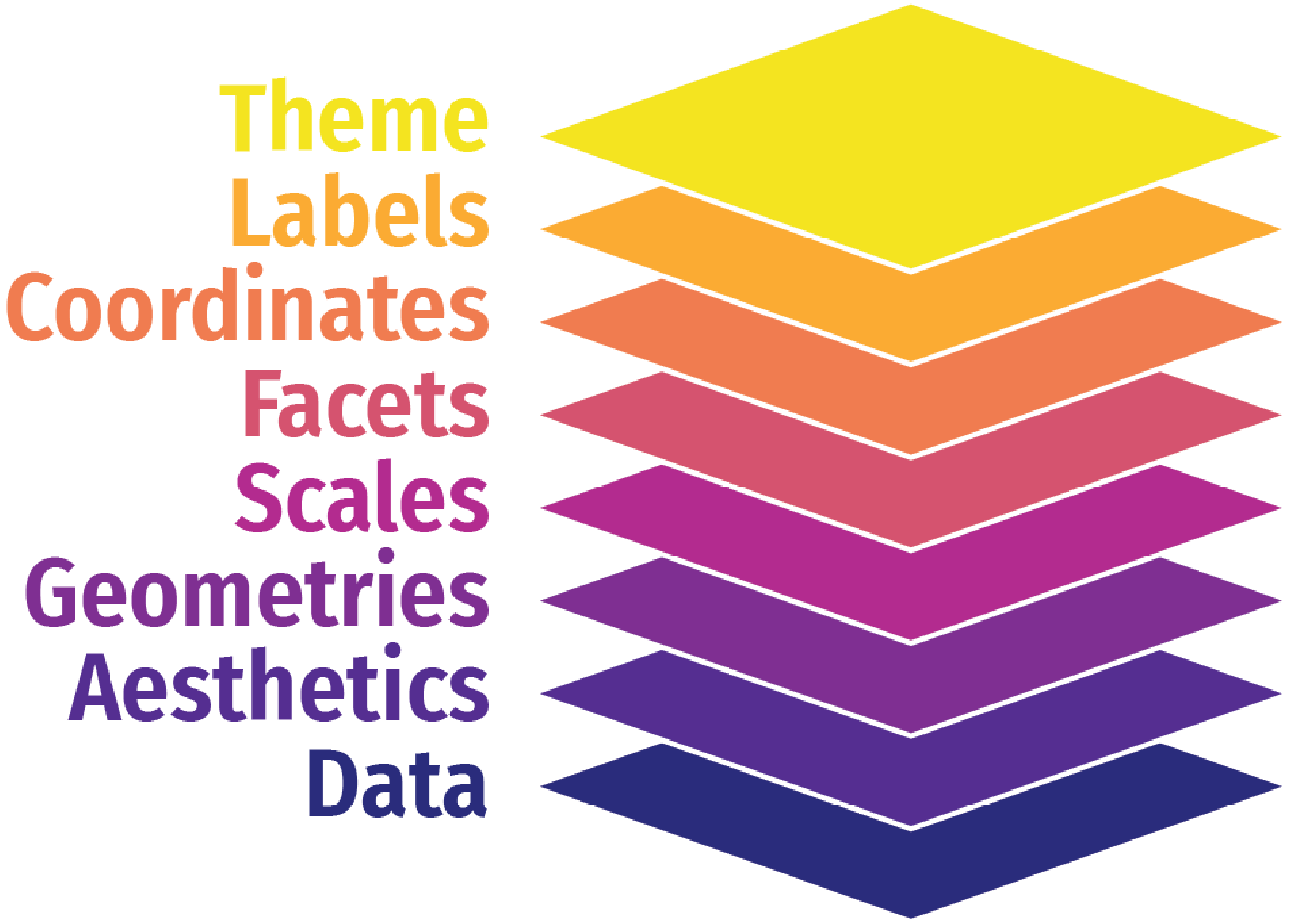
  1. **Data to be drawn from**
  2. **Aesthetic mappings from data to some visual marking**
  3. **Geometric objects on the plot**
  4. **Scales** define the range of values
  5. **Coordinates** to organize location
  6. **Labels** describe the scale and markings
  7. **Facets** group into subplots
  8. **Themes** style the plot elements



The Grammar of Graphics (gg) I

- Any graphic can be built from the same components:

1. `data` **to be drawn from**
2. `aes`**thetic mappings from data to some visual marking**
3. `geom`**etric objects on the plot**
4. `scale` define the range of values
5. `coord`**inates** to organize location
6. `labels` describe the scale and markings
7. `facet` group into subplots
8. `theme` style the plot elements



The Grammar of Graphics (gg): All at Once

All in One Command

Produces plot output in viewer

- Does not save plot (if done in console)
  - Save with `Export` menu in viewer
- Adding layers requires whole code for new plot
- Perfectly fine if it's a code chunk in a Quarto document!

```r
ggplot(data = mpg)+
  aes(x = displ,
          y = hwy)+
  geom_point()+
  geom_smooth()
```

The Grammar of Graphics (gg): As R Objects

Saving as an object

- Saves your plot as an R object
- Does *not* show in viewer
  - Execute the name of your object to see it
- Can add layers by calling the original plot name

```r
# make and save plot as p
p <- ggplot(data = mpg)+
  aes(x = displ,
      y = hwy)+
  geom_point()

p # view plot

# to add a layer...
p + geom_smooth() # shows the new plot

p <- p + geom_smooth() # overwrites p
p2 <- p + geom_smooth() # saves new object
```

Plot Layers

The Grammar of Graphics (gg): Tidy Data

Data

```
ggplot(data = mpg)
```

Data is the source of our data. As part of the `tidyverse`, `ggplot2` requires data to be **“tidy”**[1]:

1. Each variable forms a **column**
2. Each observation forms a **row**
3. Each observational unit forms a table

[1] Data “tidyness” is the core element of all `tidyverse` packages. Much more on all of this next class.

gg: Data Layer

Data

```
ggplot(data = mpg)
```

- Add a layer with + at the end of a line (never at the beginning!)
- Style recommendation: start a new line after each + to improve legibility!
- We will build a plot layer-by-layer

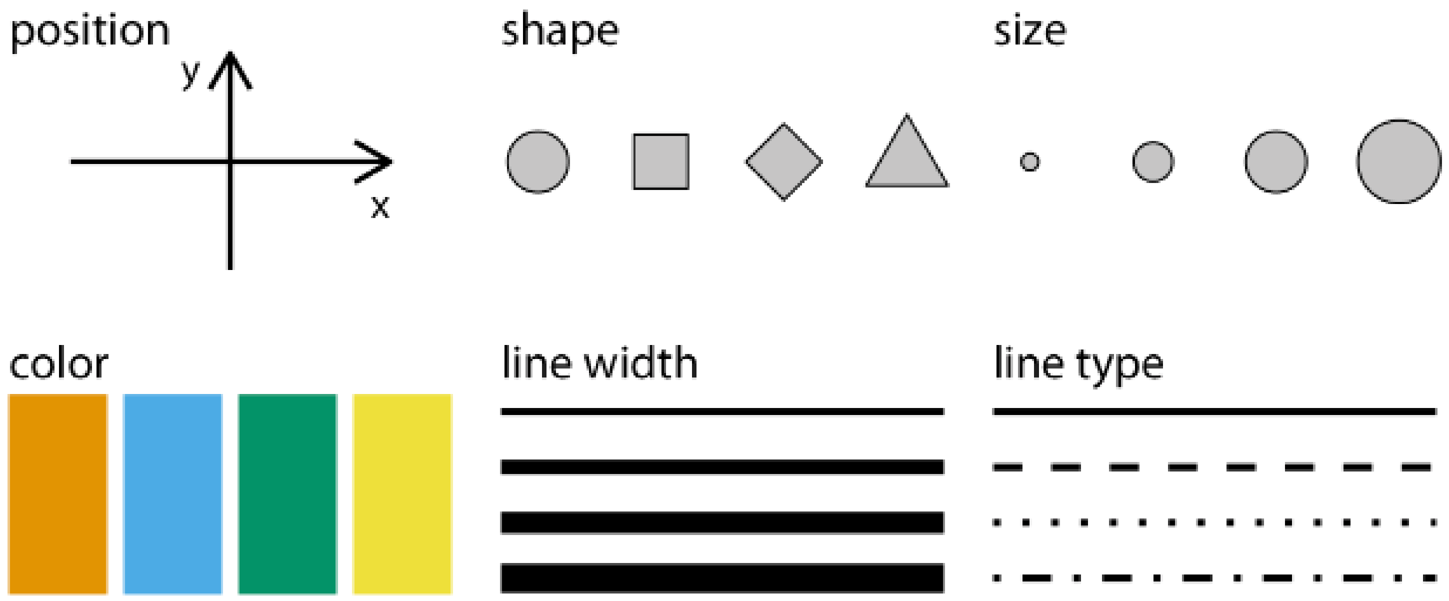
gg: Mapping Aesthetics I

Data

Aesthetics

```
+aes(...)
```

Aesthetics map data to visual elements or parameters



gg: Mapping Aesthetics II

Data

Aesthetics

`+aes(...)`

Aesthetics map data to visual elements or parameters

- `displ`
- `hwy`
- `class`

gg: Mapping Aesthetics III

Data

Aesthetics

`+aes(...)`

Aesthetics map data to visual elements or parameters

- `displ` → **x**
- `hwy` → **y**
- `class` → **color**, (or **shape**, **size**, etc.)

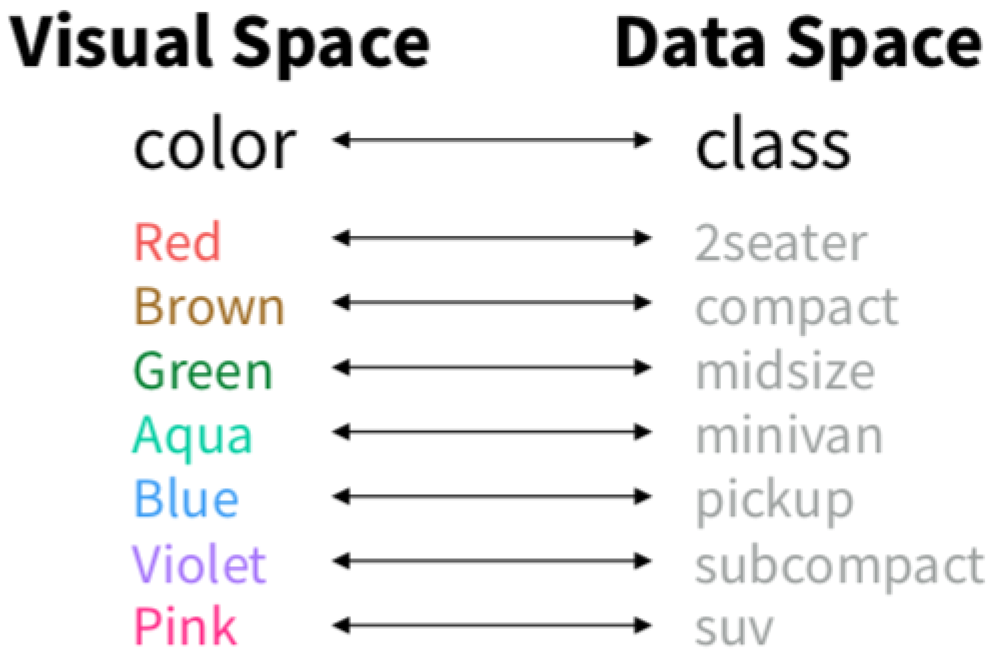
gg: Mapping Aesthetics IV

Data

Aesthetics

`+aes(...)`

Aesthetics map data to visual elements or parameters



Visual Space	Data Space
color	class
Red	2seater
Brown	compact
Green	midsize
Aqua	minivan
Blue	pickup
Violet	subcompact
Pink	suv

gg: Mapping Aesthetics V

Data

Aesthetics

`+aes(...)`

Aesthetics map data to visual elements or parameters

```
aes(x = displ,
    y = hwy,
    color = class)
```

gg: Geoms I

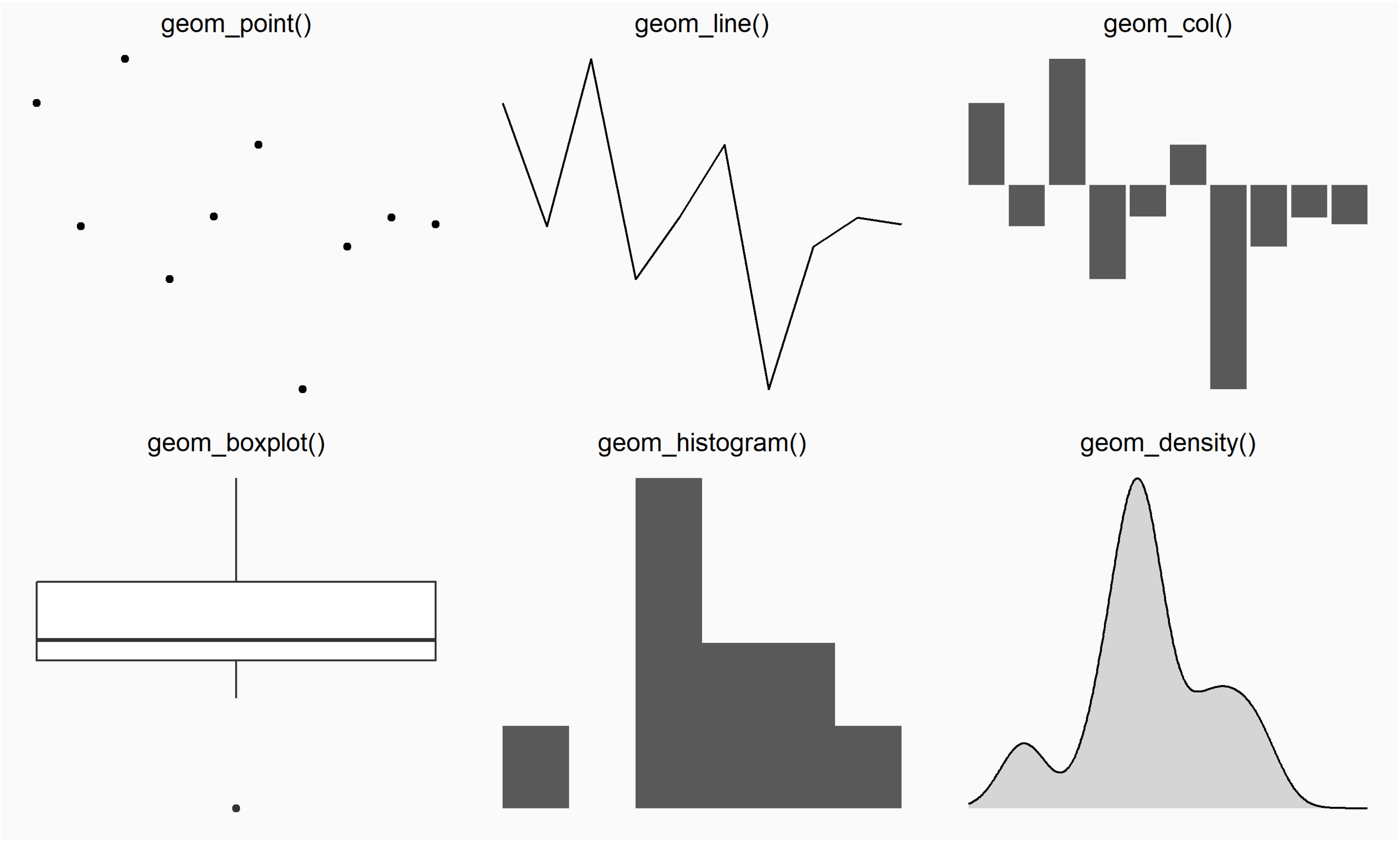
Data

Aesthetics

Geoms

`+geom_*(...)`

Geometric objects displayed on the plot



gg: Geoms II

Data

Aesthetics

Geoms

`+geom_*(...)`

Geometric objects displayed on the plot

- What `geom`s you should use depends on what you want to show:

Type	**`geom`**
Point	`geom_point()`
Line	`geom_line()`, `geom_path()`
Bar	`geom_bar()`, `geom_col()`
Histogram	`geom_histogram()`
Regression	`geom_smooth()`
Boxplot	`geom_boxplot()`
Text	`geom_text()`
Density	`geom_density()`

gg: Geoms III

Data

Aesthetics

Geoms

`+geom_*(...)`

Geometric objects displayed on the plot

```
##  [1] "geom_abline"     "geom_area"       "geom_bar"        "geom_bin2
##  [5] "geom_blank"      "geom_boxplot"    "geom_col"        "geom_cont
##  [9] "geom_count"      "geom_crossbar"   "geom_curve"      "geom_dens
## [13] "geom_density_2d" "geom_density2d"  "geom_dotplot"    "geom_erro
## [17] "geom_errorbarh"  "geom_freqpoly"   "geom_hex"        "geom_hist
## [21] "geom_hline"      "geom_jitter"     "geom_label"      "geom_line
## [25] "geom_linerange"  "geom_map"        "geom_path"       "geom_poin
## [29] "geom_pointrange" "geom_polygon"    "geom_qq"         "geom_qq_l
## [33] "geom_quantile"   "geom_raster"     "geom_rect"       "geom_ribb
## [37] "geom_rug"        "geom_segment"    "geom_sf"         "geom_sf_l
## [41] "geom_sf_text"    "geom_smooth"     "geom_spoke"      "geom_step
## [45] "geom_text"       "geom_tile"       "geom_violin"     "geom_vlin
```

See http://ggplot2.tidyverse.org/reference for many more options

gg: Geoms IV

Data

Aesthetics

Geoms

`+geom_*(...)`

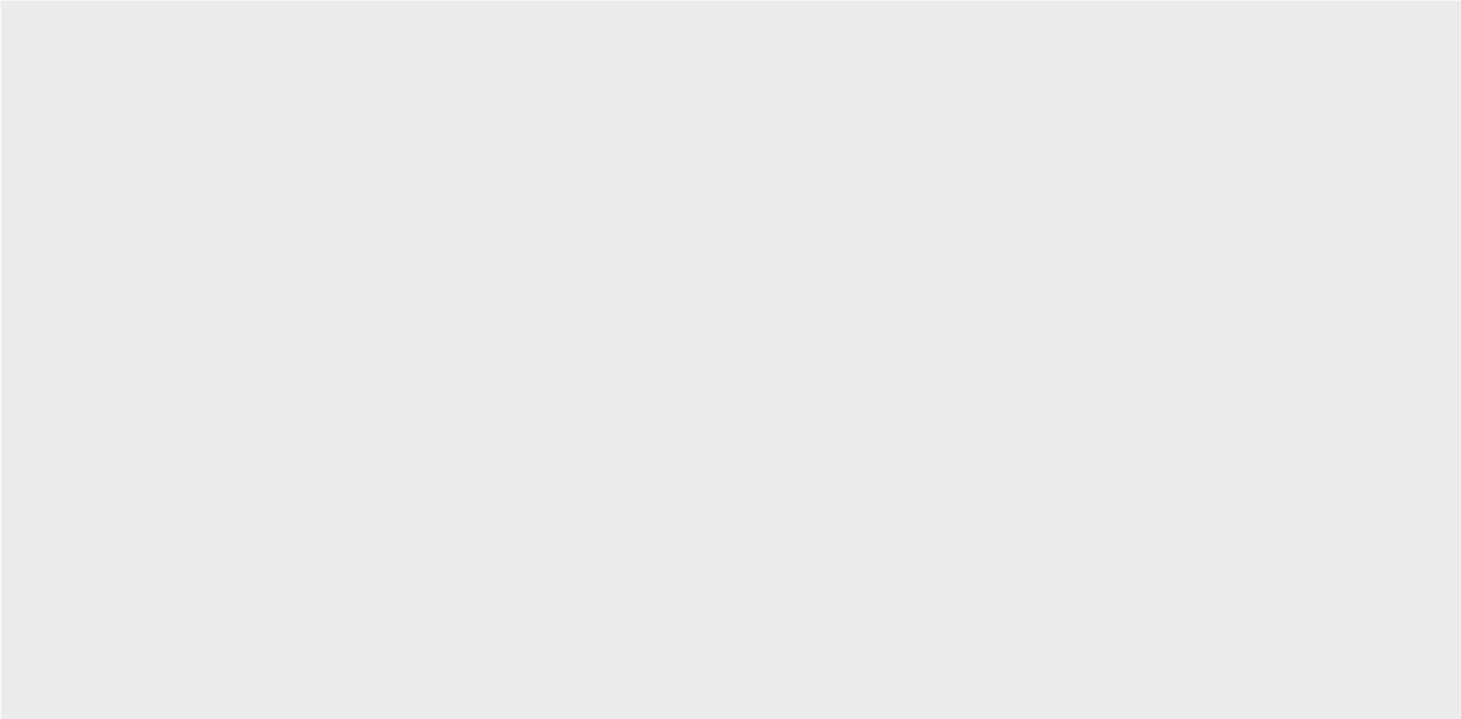
Geometric objects displayed on the plot

Or just start typing `geom_` in R Studio!

```
ggplot(df_geom) +
  aes(x, y) +
  
```

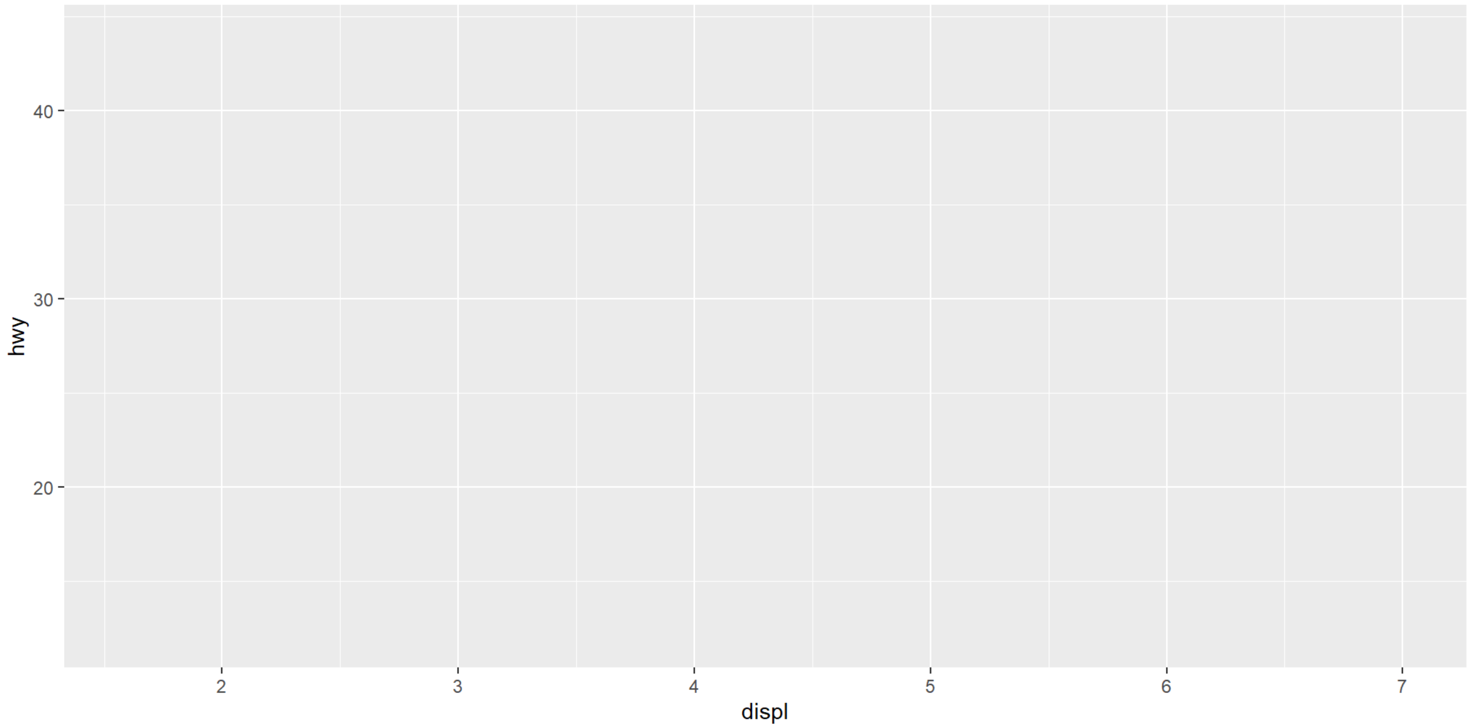
Let’s Make a Plot!

```r
ggplot(data = mpg)
```



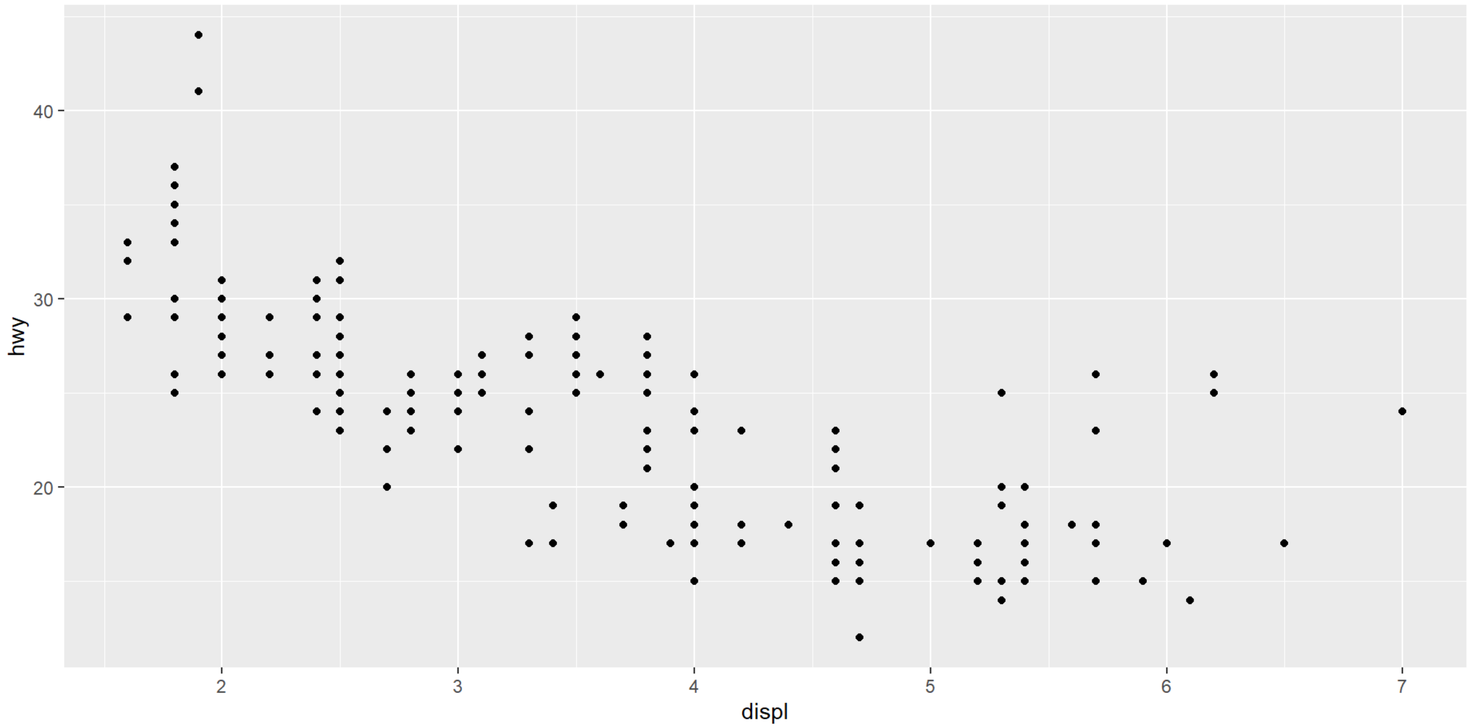
Let’s Make a Plot!

```r
ggplot(data = mpg)+
  aes(x = displ,
      y = hwy)
```



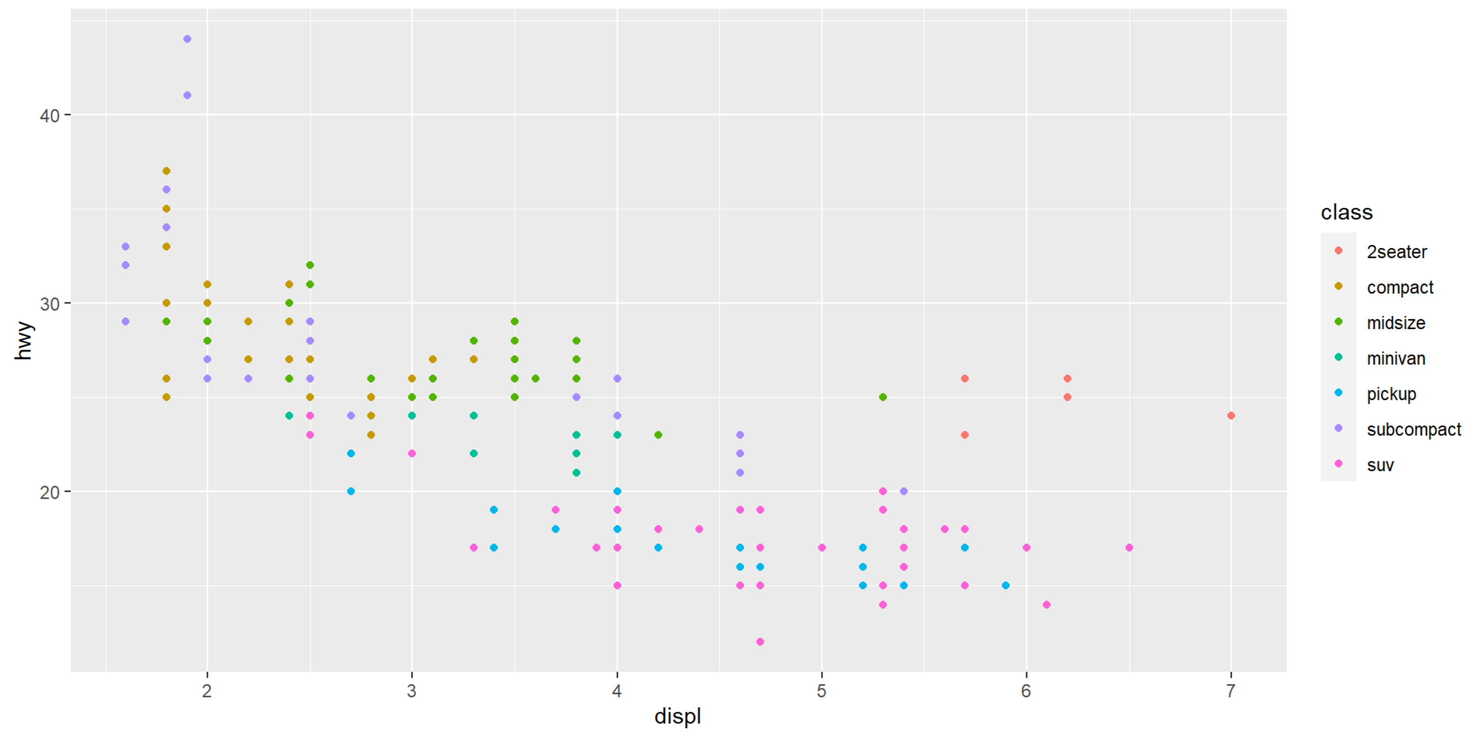
Let’s Make a Plot!

```r
ggplot(data = mpg) +
  aes(x = displ,
      y = hwy) +
      geom_point()
```



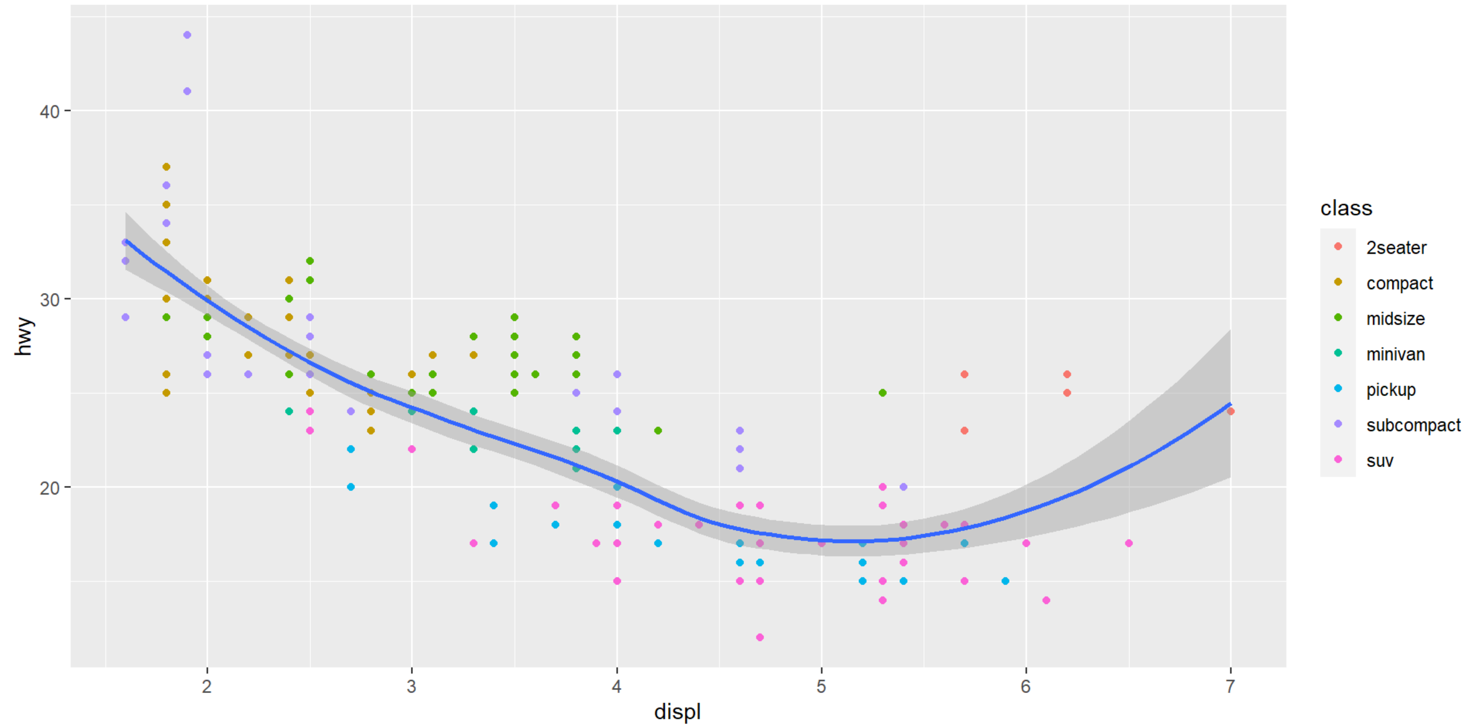
Let’s Make a Plot!

```r
ggplot(data = mpg) +
  aes(x = displ,
      y = hwy) +
      geom_point(aes(color = class))
```



Let’s Make a Plot!

```r
ggplot(data = mpg)+
  aes(x = displ,
      y = hwy)+
      geom_point(aes(color = class))+
      geom_smooth()
```



More Geoms

Data

Aesthetics

Geoms

`+geom_*(...)`

```
geom_*(aes, data, stat, position)
```

- `data`: geoms can have their own data
  - has to map onto global coordinates
- `aes`: geoms can have their own aesthetics
  - inherits global aesthetics by default
  - different geoms have different available aesthetics

More Geoms II

Data

Aesthetics

Geoms

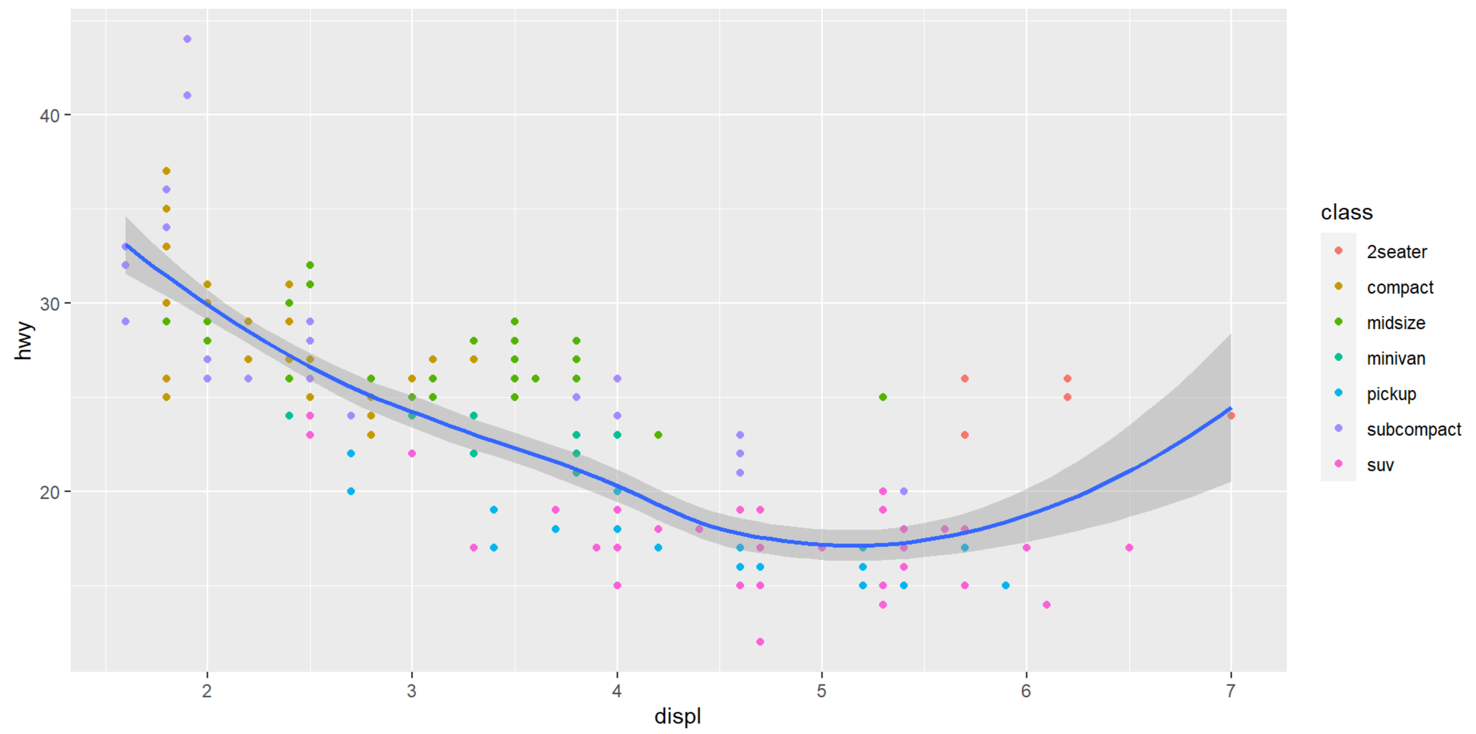
`+geom_*(...)`

```
geom_*(aes, data, stat, position)
```

- `stat`: some geoms statistically transform data
  - `geom_histogram()` uses `stat_bin()` to group observations into bins
- `position`: some adjust location of objects
  - `dodge`, `stack`, `jitter`

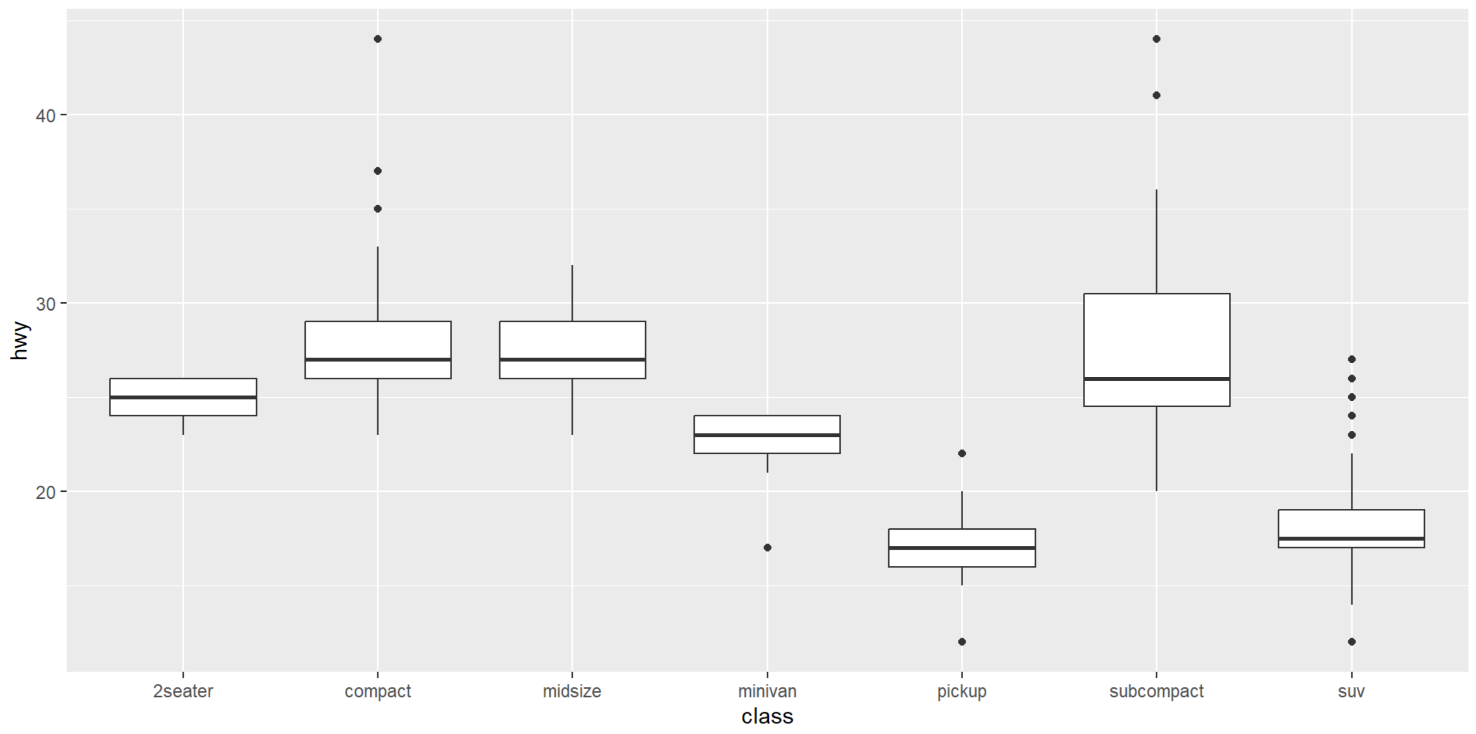
Our Plot

```r
ggplot(data = mpg)+
  aes(x = displ,
      y = hwy)+
      geom_point(aes(color = class))+
      geom_smooth()
```



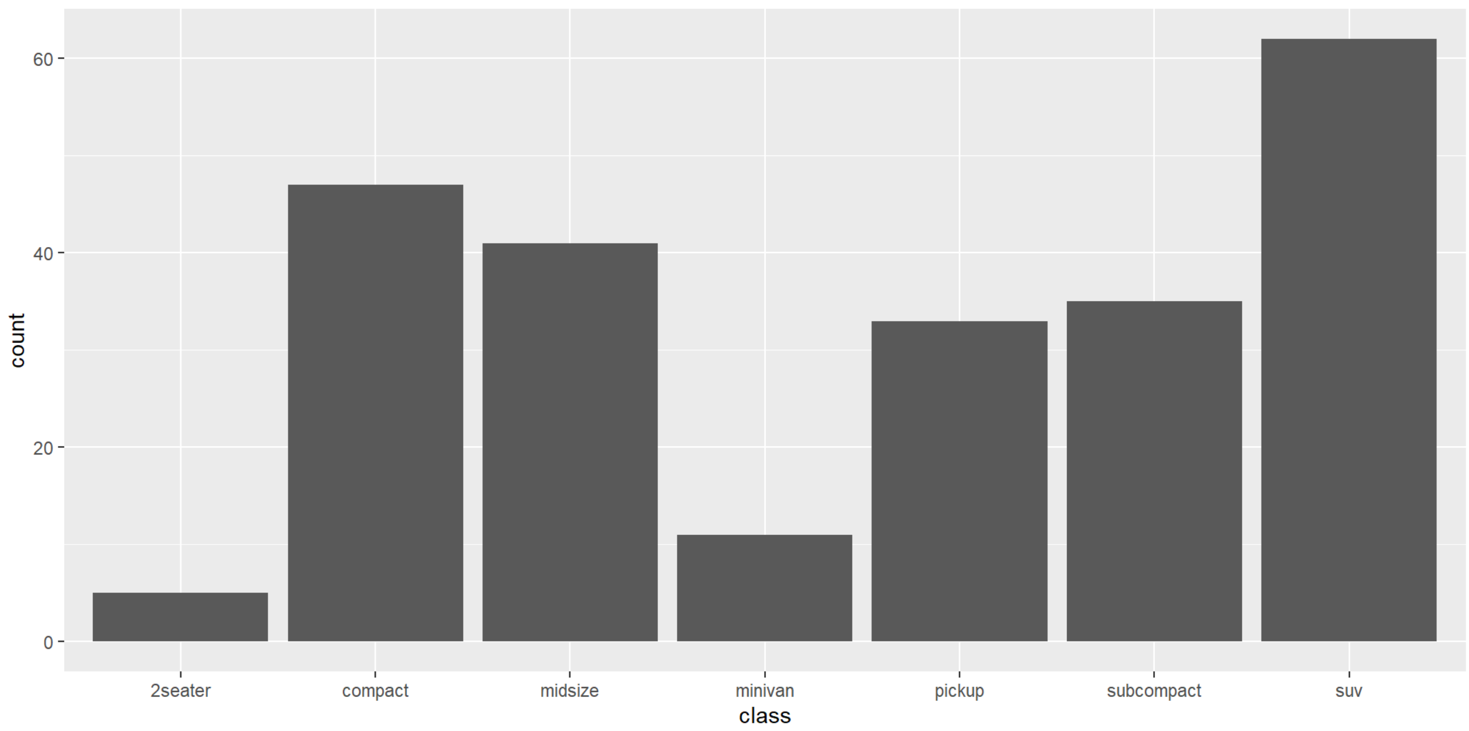
Change Our Plot

```r
ggplot(data = mpg) +
  aes(x = class,
      y = hwy) +
      geom_boxplot()
```



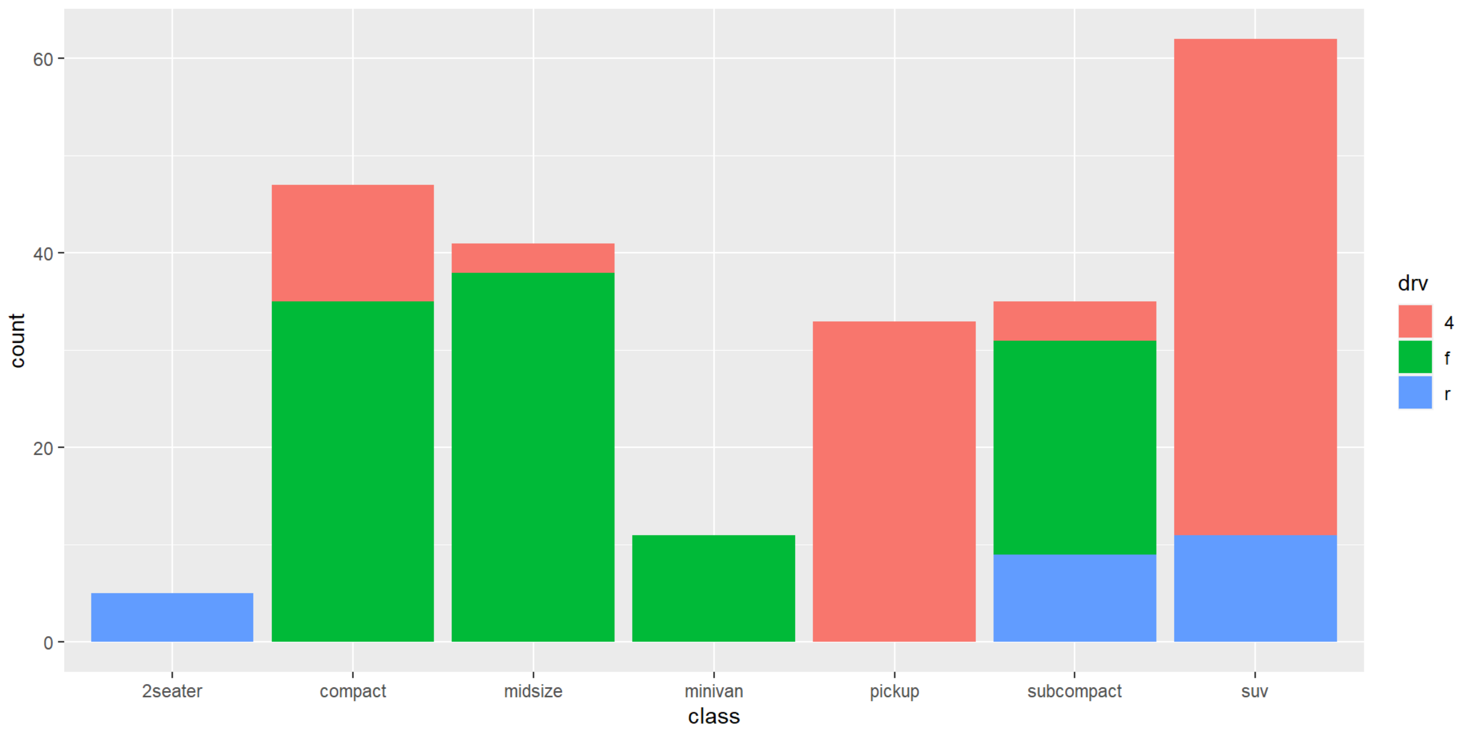
Change Our Plot

```
ggplot(data = mpg)+
  aes(x = class)+
      geom_bar()
```



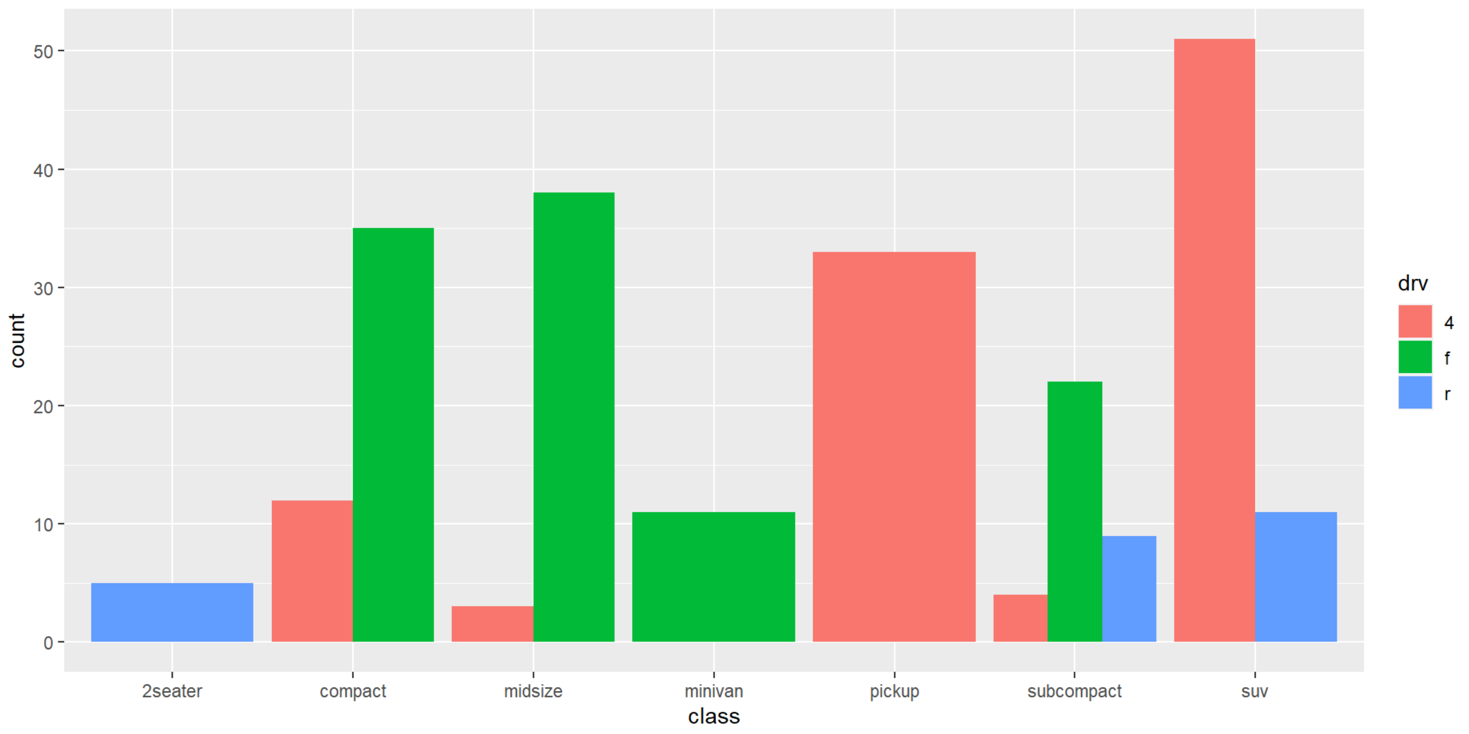
Change Our Plot

```r
ggplot(data = mpg)+
  aes(x = class,
      fill = drv)+
      geom_bar()
```



Change Our Plot

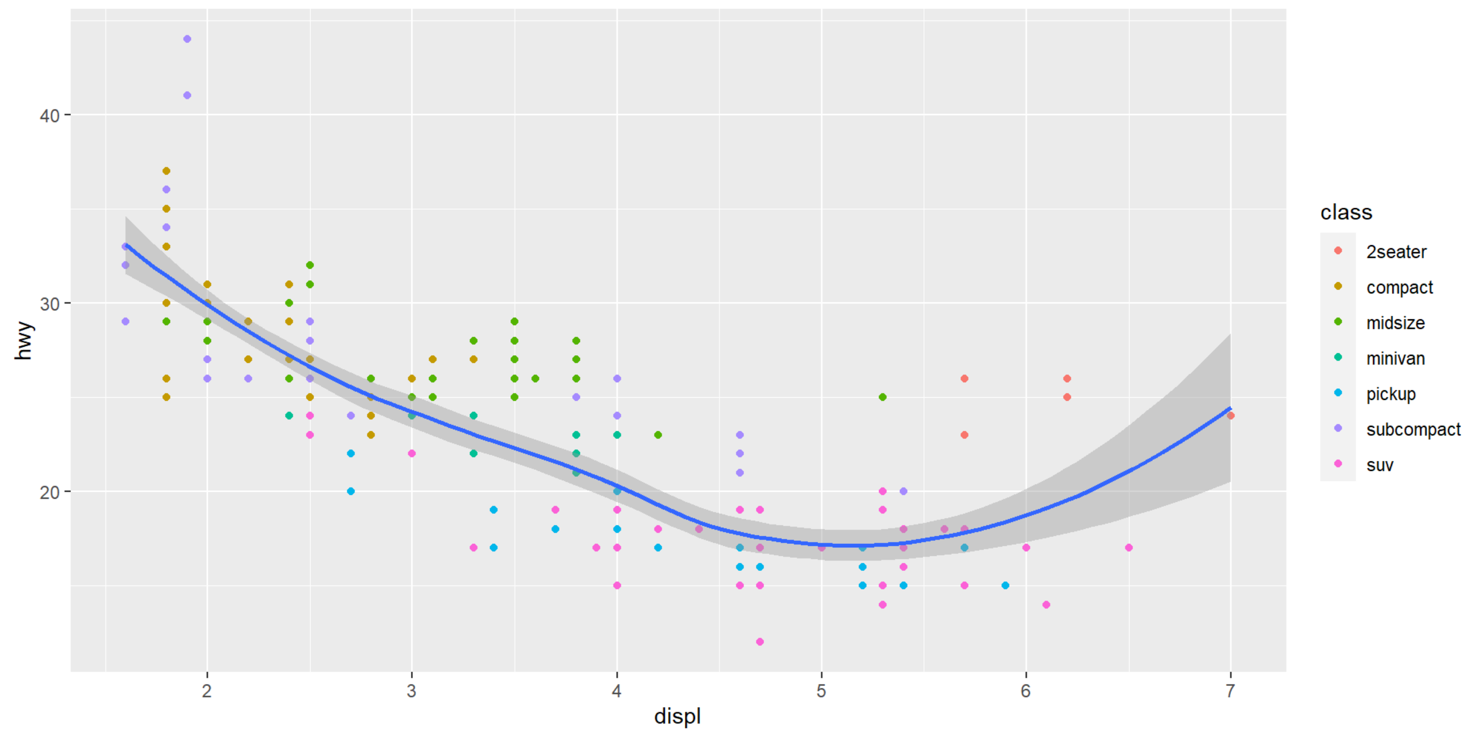
```
ggplot(data = mpg) +
  aes(x = class,
      fill = drv) +
      geom_bar(position = "dodge")
```



Back to the Original (and Saving It)

```r
# save plot as p
p <- ggplot(data = mpg)+
  aes(x = displ,
        y = hwy)+
  geom_point(aes(color = class))+
  geom_smooth()

p # show plot
```



gg: Facets I

Data

Aesthetics

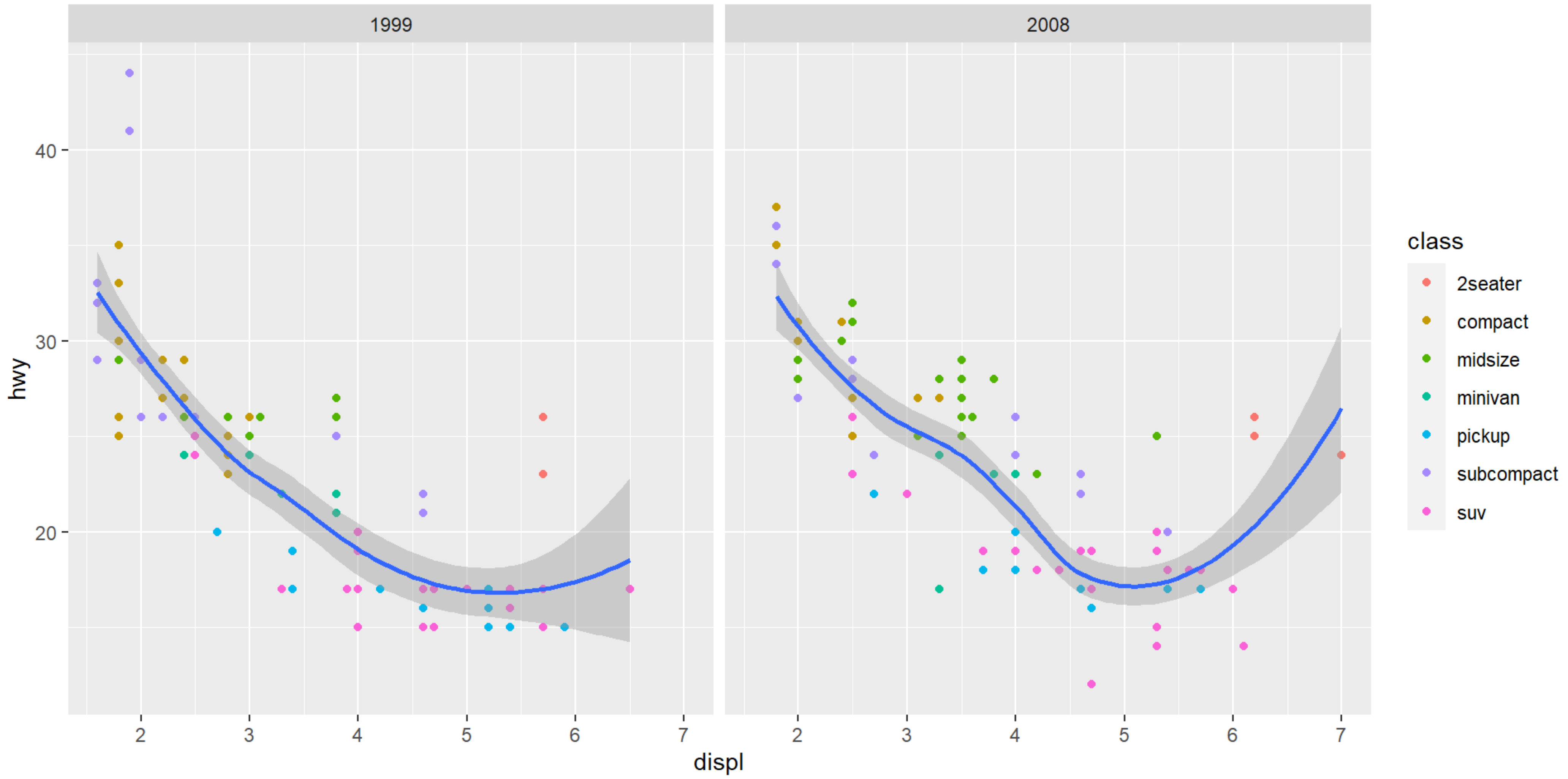
Geoms

Facets

`+ facet_wrap()`

`+ facet_grid()`

```r
p + facet_wrap(~year)
```



gg: Facets II

Data

Aesthetics

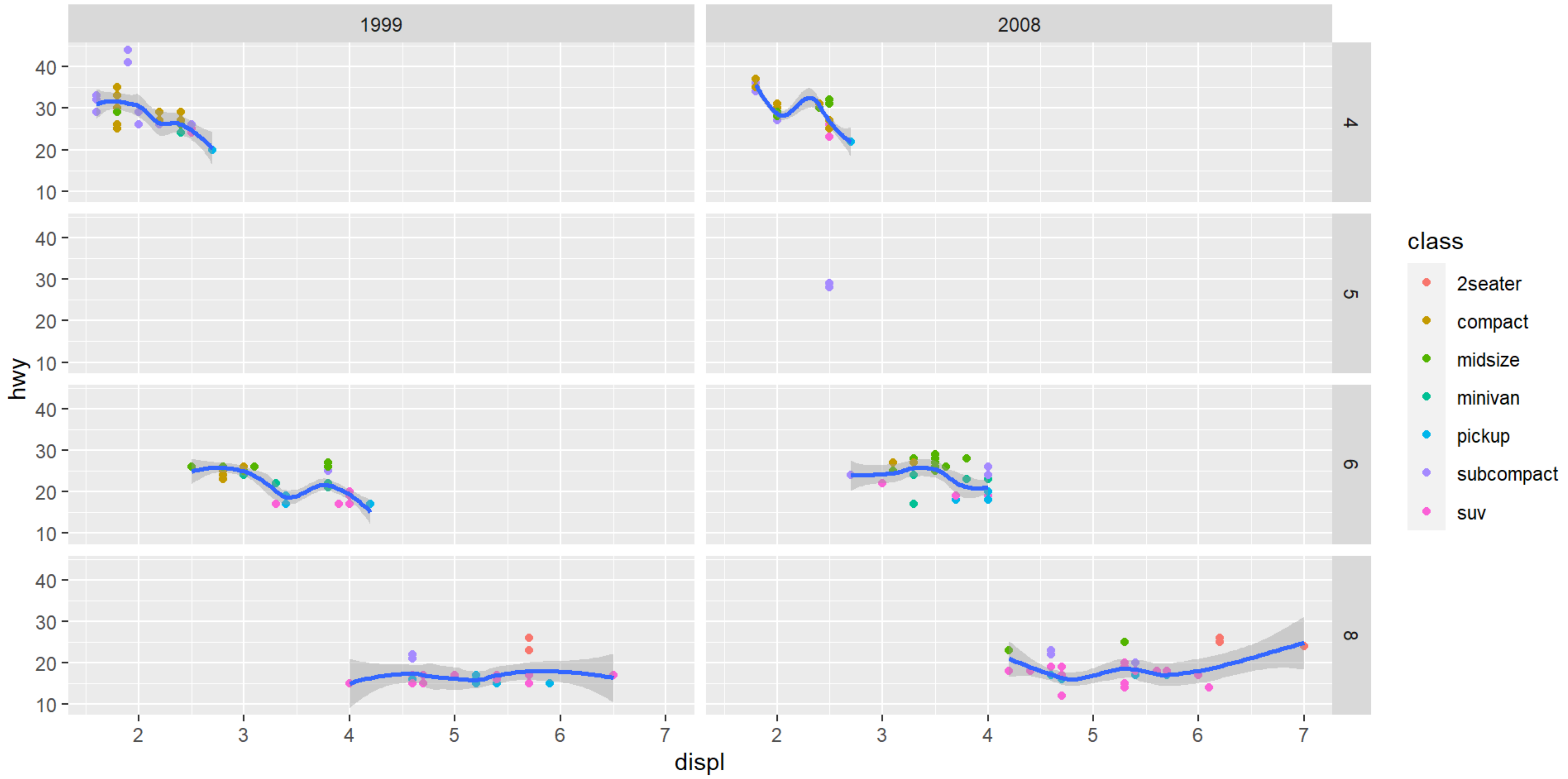
Geoms

Facets

`+ facet_wrap()`

`+ facet_grid()`

```r
p + facet_grid(cyl ~ year)
```



gg: Labels

Data

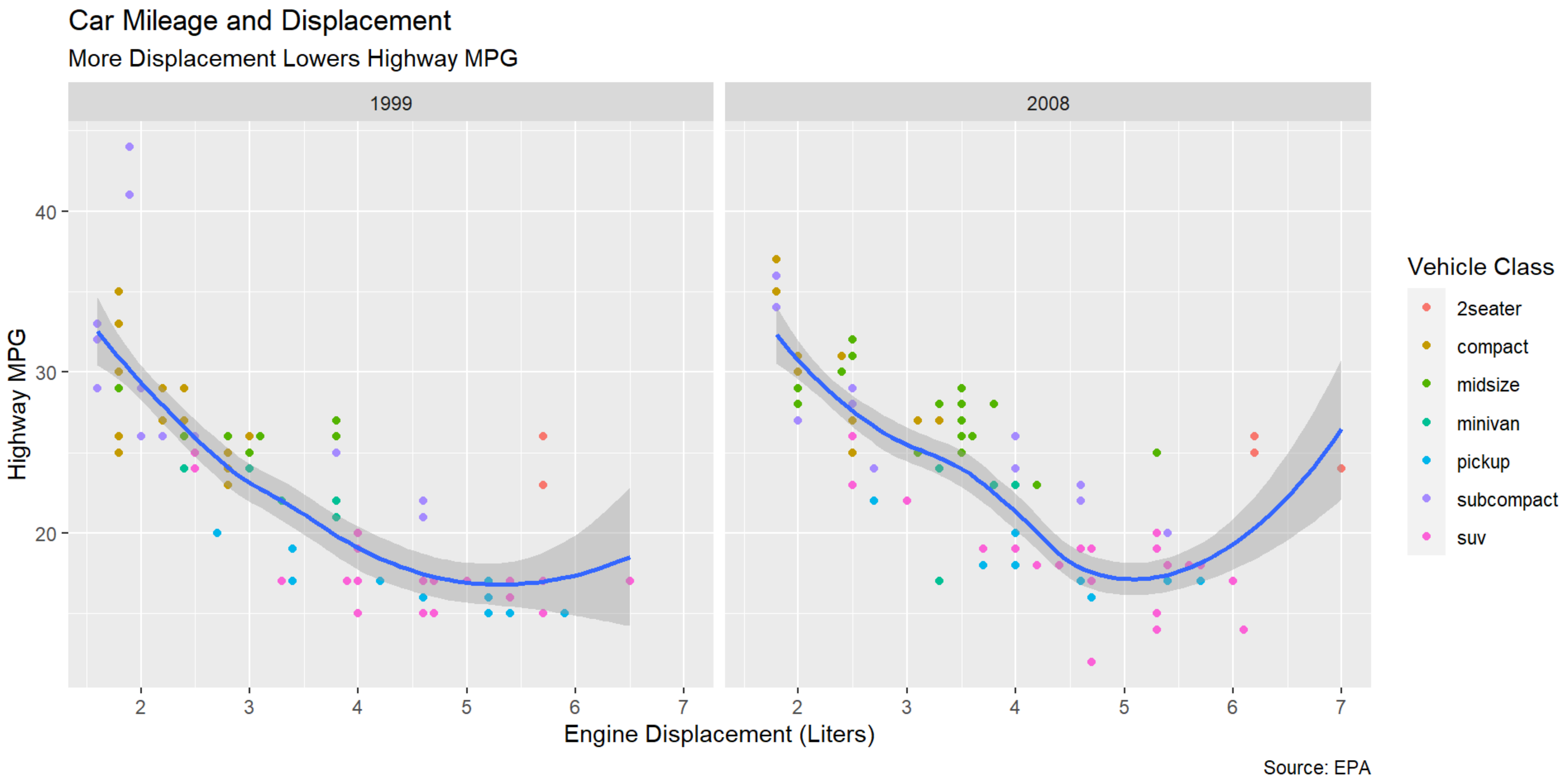
Aesthetics

Geoms

Facets

`+ labs()`

```r
(p <- p + facet_wrap(~year)+
  labs(x = "Engine Displacement (Liters)",
       y = "Highway MPG",
       title = "Car Mileage and Displacement",
       subtitle = "More Displacement Lowers Highway MPG",
       caption = "Source: EPA",
       color = "Vehicle Class"))
```



Car Mileage and Displacement

More Displacement Lowers Highway MPG

Source: EPA

gg: Scales I

Data

Aesthetics

Geoms

Facets

Scales

`+ scale_*_*()`

`scale+_+<aes>+_+<type>+()`

- `<aes>`: parameter to adjust
- `<type`: type of parameter
- Discrete x-axis: `scale_x_discrete()`
- Continuous y-axis: `scale_y_continuous()`
- Rescale x-axis to log: `scale_x_log10()`
- Use different color palette: `scale_fill_discrete()`, `scale_color_manual()`

gg: Scales II

Data

Aesthetics

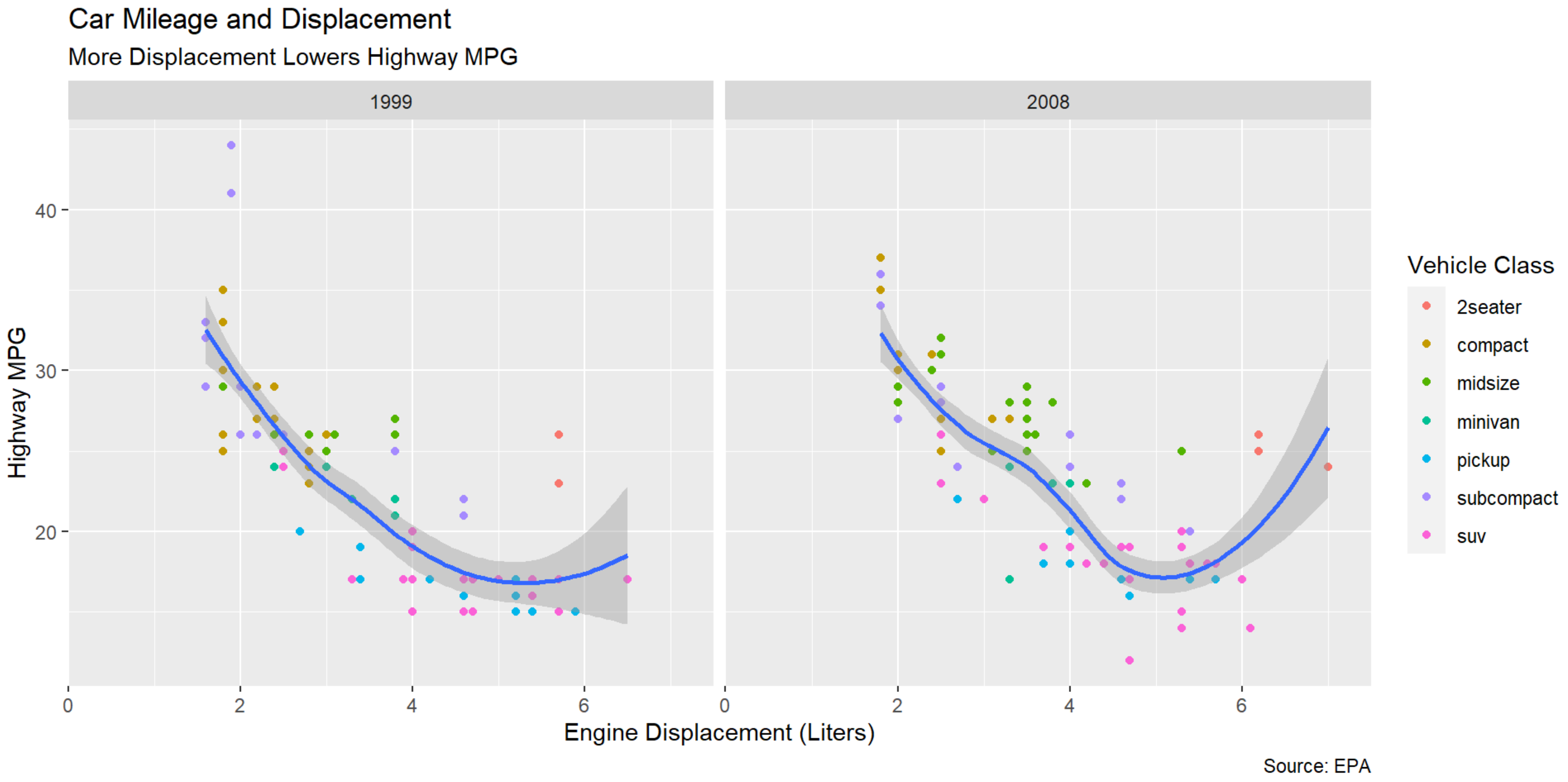
Geoms

Facets

Scales

`+ scale_*_*()`

```r
p + scale_x_continuous(breaks = seq(0, 10, 2),
                       limits = c(0,7.5),
                       expand = c(0,0)
)
```



gg: Scales II

Data

Aesthetics

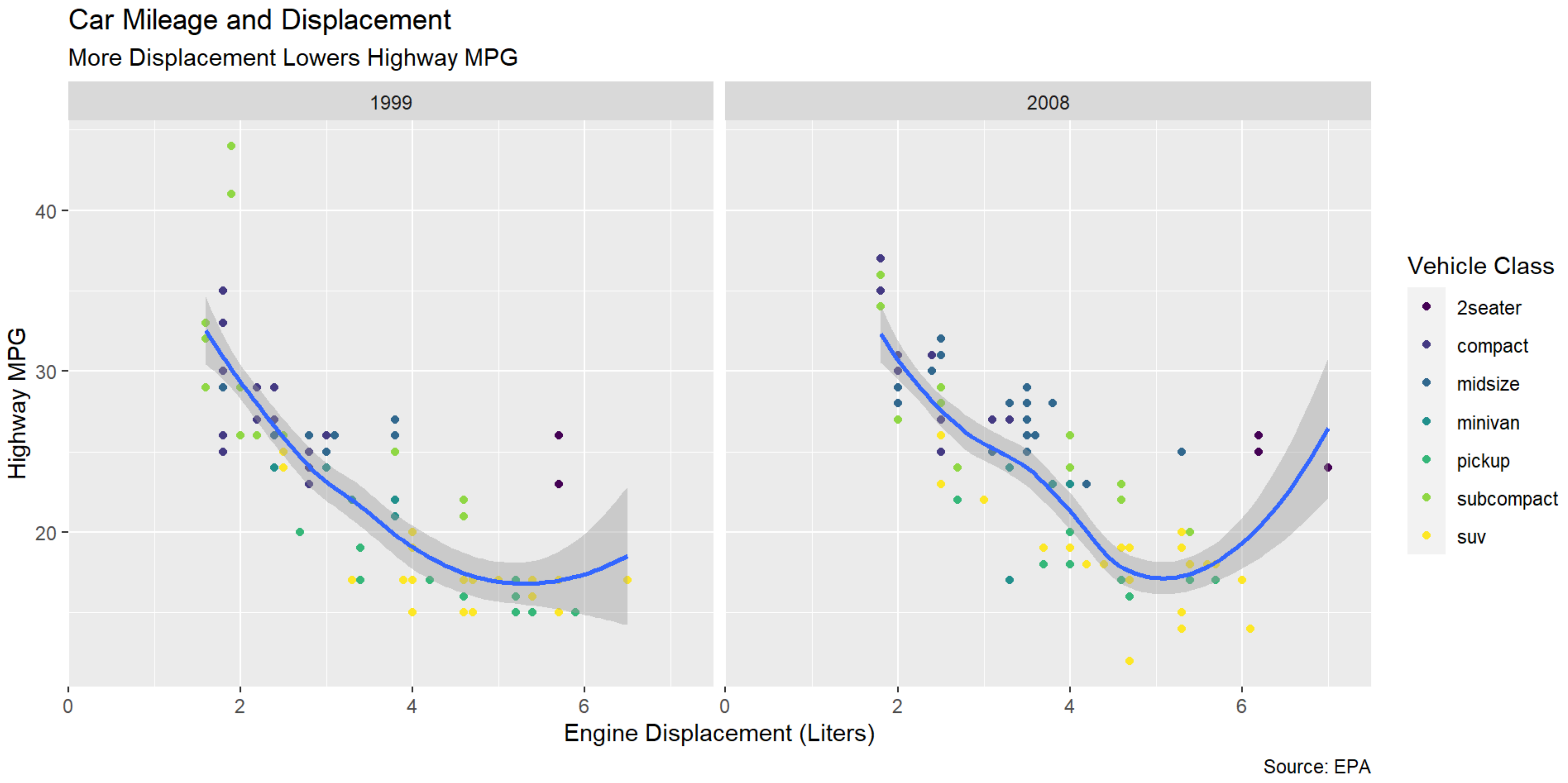
Geoms

Facets

Scales

`+ scale_*_*()`

```r
p + scale_x_continuous(breaks = seq(0, 10, 2),
                       limits = c(0,7.5),
                       expand = c(0,0)
                       ) +
  scale_color_viridis_d()
```



Car Mileage and Displacement

More Displacement Lowers Highway MPG

1999

2008

Source: EPA

gg: Themes I

Data

Aesthetics

Geoms

Facets

Scales

Themes

`+ theme_*()`

Theme changes appearance of plot decorations (things not mapped to data)

- Some themes that come with `ggplot2`:
  - `+ theme_bw()`
  - `+ theme_dark()`
  - `+ theme_gray()`
  - `+ theme_minimal()`
  - `+ theme_light()`
  - `+ theme_classic()`

gg: Themes II

Data

Aesthetics

Geoms

Facets

Scales

Themes

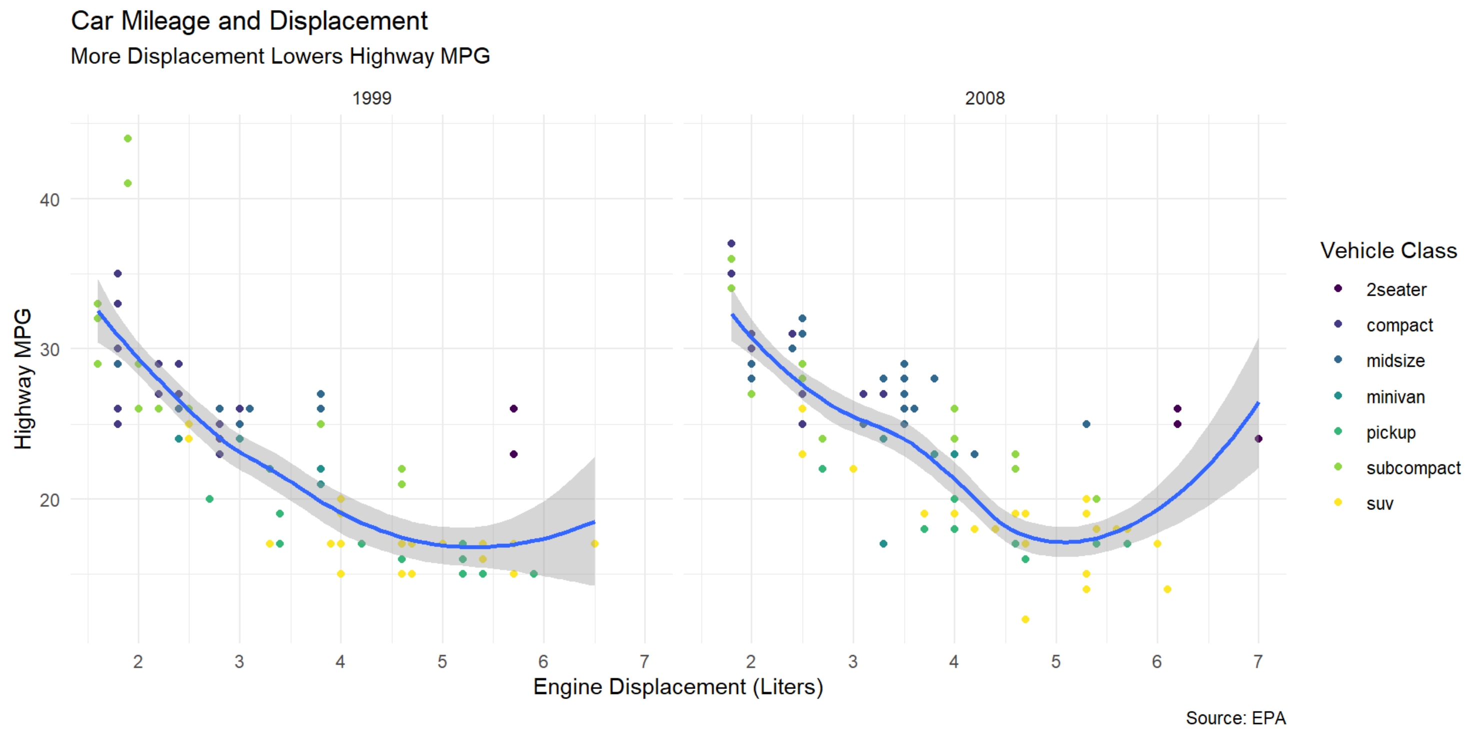
`+ theme_*()`

Theme changes appearance of plot decorations (things not mapped to data)

- Many parameters we could customize
- Global options: `line`, `rect`, `text`, `title`
- `axis`: x-, y-, or other axis title, ticks, lines
- `legend`: plot legends for fill or color
- `panel`: actual plot area
- `plot`: whole image
- `strip`: facet labels

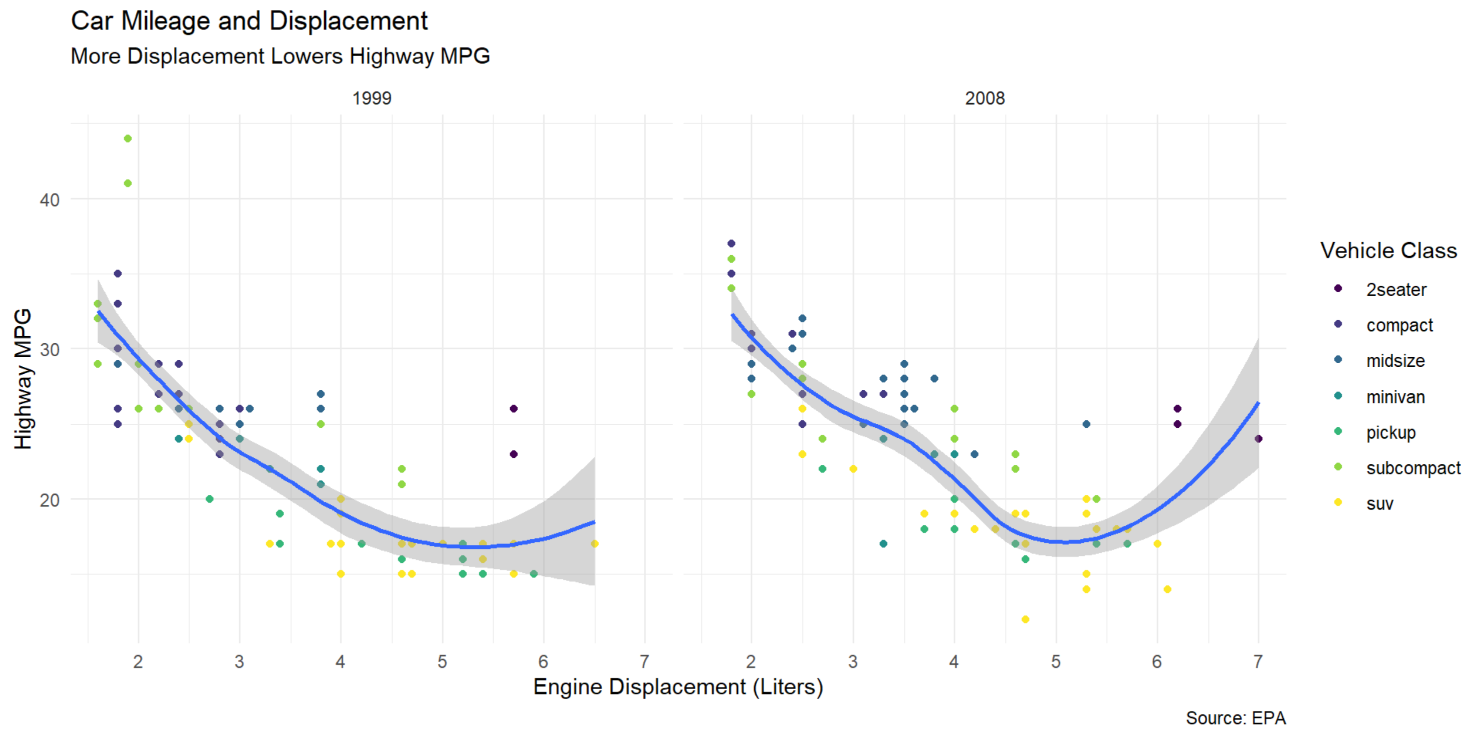
gg: Themes III

```
ggplot(data = mpg)+
  aes(x = displ,
      y = hwy)+
  geom_point(aes(color = class))+
  geom_smooth()+
  facet_wrap(~year)+
  labs(x = "Engine Displacement (Liters)",
       y = "Highway MPG",
       title = "Car Mileage and Displacement",
       subtitle = "More Displacement Lowers Highwa
       caption = "Source: EPA",
       color = "Vehicle Class")+
  scale_color_viridis_d()+
  theme_minimal()
```



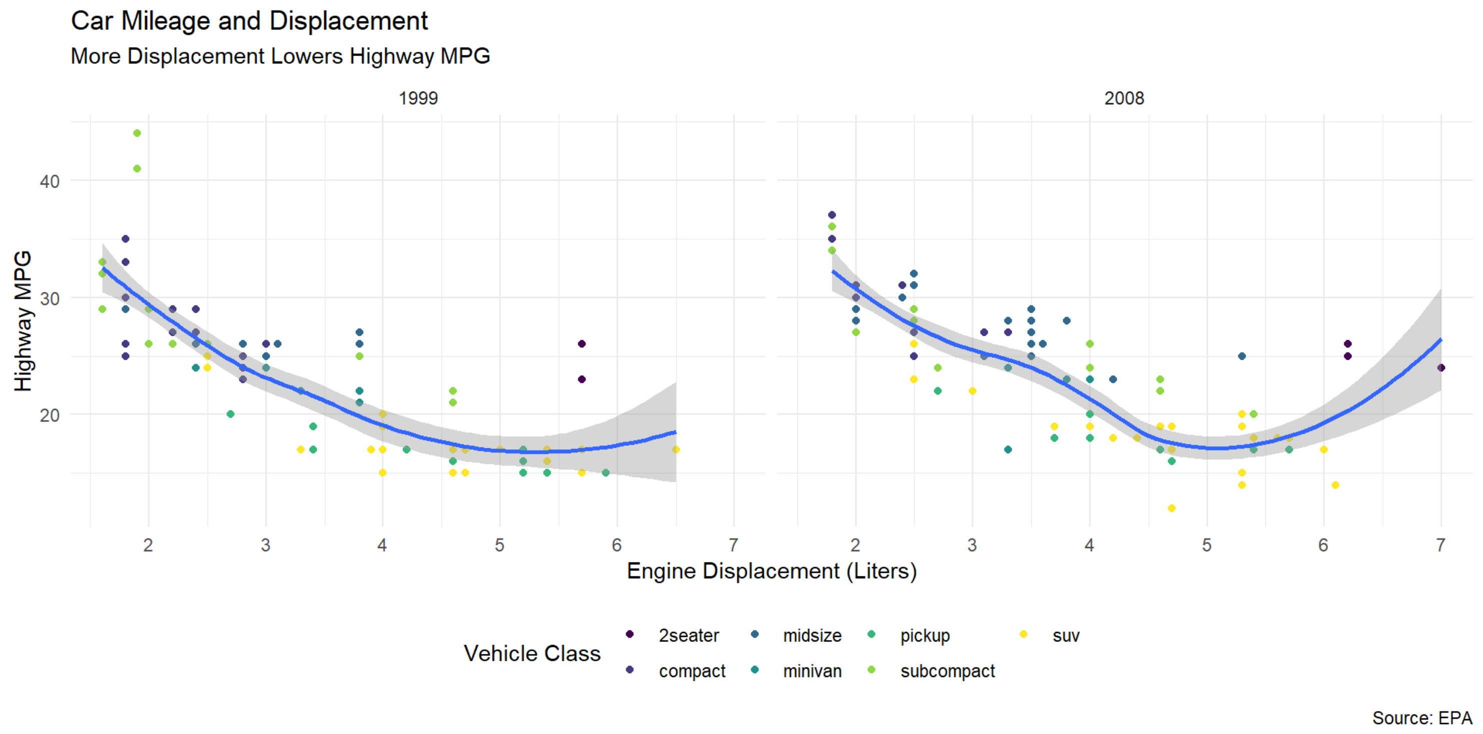
gg: Themes IV

```r
ggplot(data = mpg)+
  aes(x = displ,
      y = hwy)+
  geom_point(aes(color = class))+
  geom_smooth()+
  facet_wrap(~year)+
  labs(x = "Engine Displacement (Liters)",
       y = "Highway MPG",
       title = "Car Mileage and Displacement",
       subtitle = "More Displacement Lowers Highwa
       caption = "Source: EPA",
       color = "Vehicle Class")+
  scale_color_viridis_d()+
  theme_minimal()+
  theme(text = element_text(family = "Fira Sans"))
```



gg: Themes V

```r
ggplot(data = mpg)+
  aes(x = displ,
      y = hwy)+
  geom_point(aes(color = class))+
  geom_smooth()+
  facet_wrap(~year)+
  labs(x = "Engine Displacement (Liters)",
       y = "Highway MPG",
       title = "Car Mileage and Displacement",
       subtitle = "More Displacement Lowers Highwa
       caption = "Source: EPA",
       color = "Vehicle Class")+
  scale_color_viridis_d()+
  theme_minimal()+
  theme(text = element_text(family = "Fira Sans"),
        legend.position = "bottom")
```



gg: Themes VI

Data

Aesthetics

Geoms

Facets

Scales

Themes

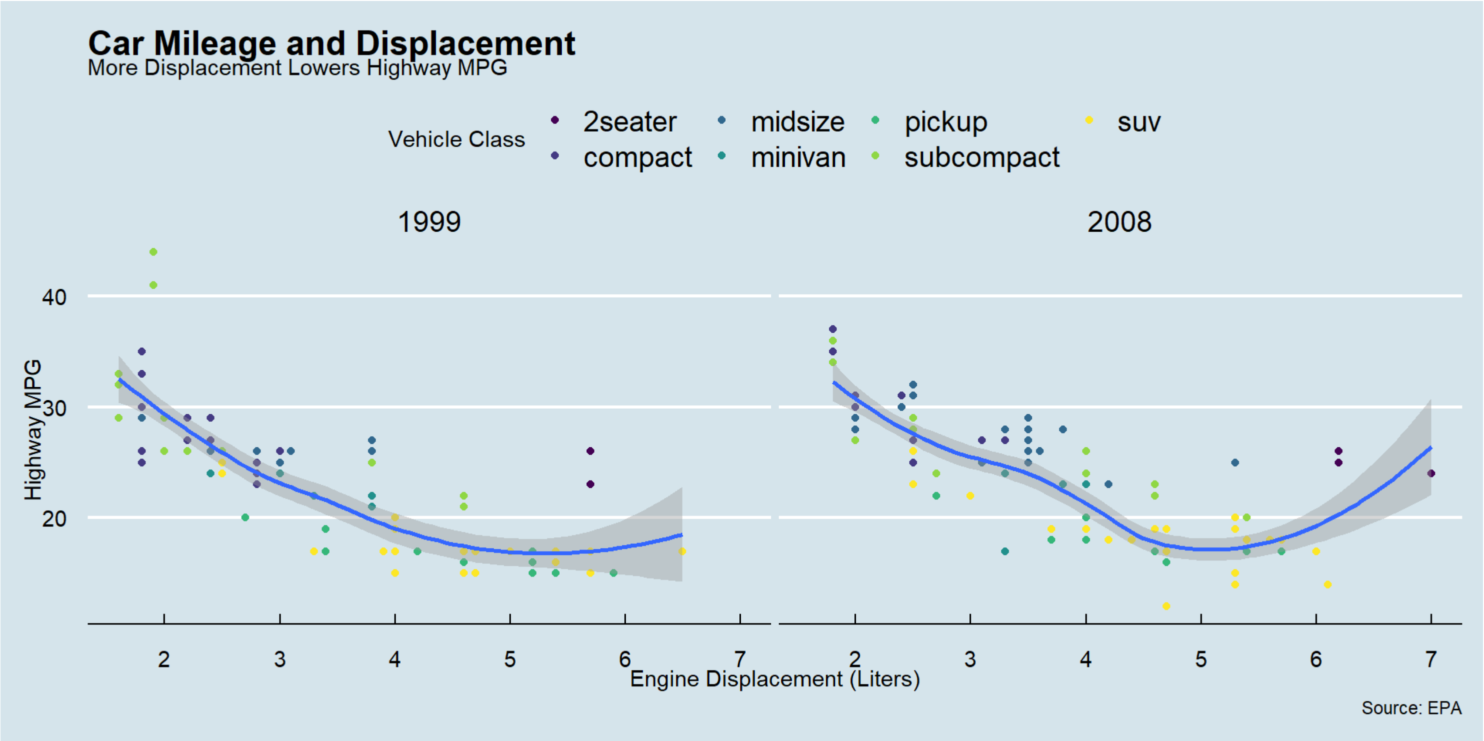
`+ theme_*()`

- `ggthemes` package adds some other nice themes

```r
# install if you don't have it
# install.packages("ggthemes")
library("ggthemes") # load package
```

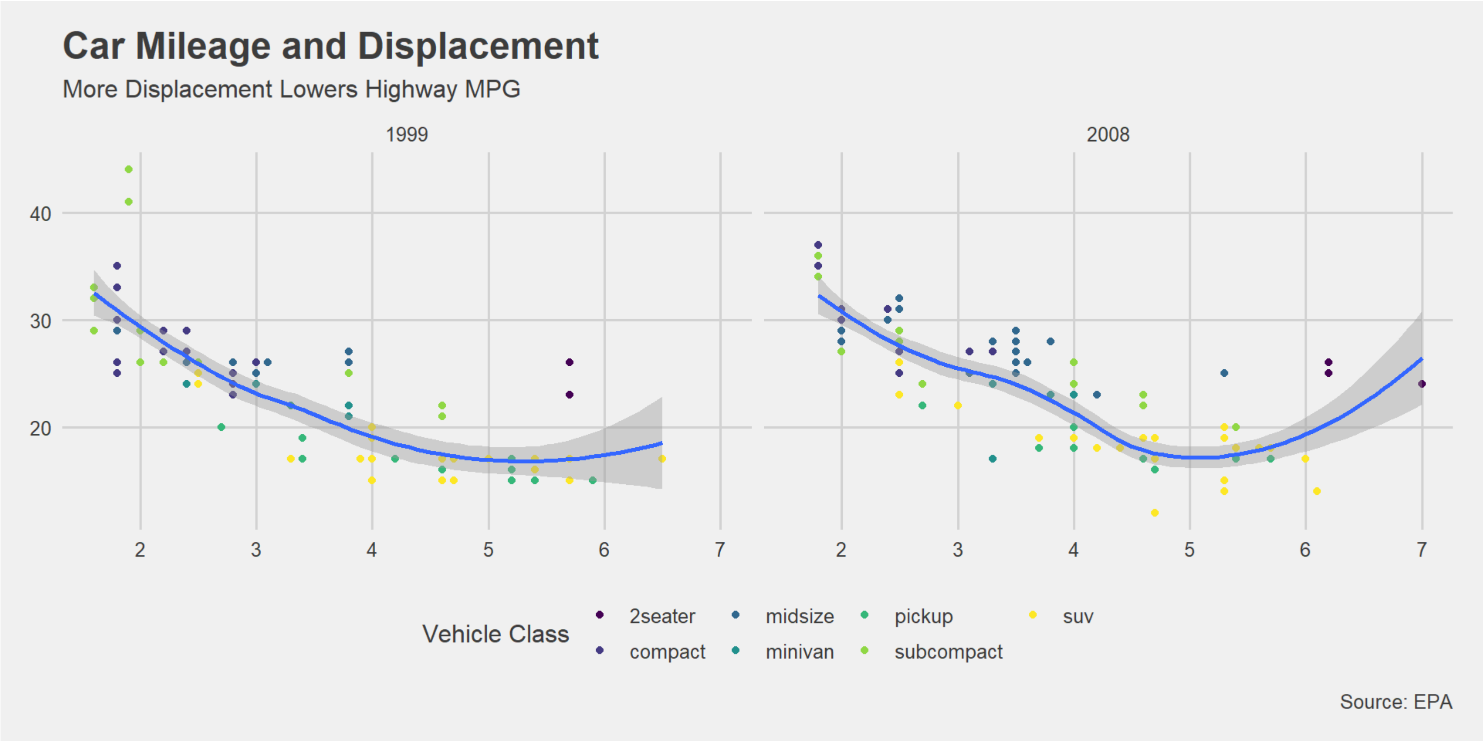
gg: Themes VII

```r
library(ggthemes)
ggplot(data = mpg)+
  aes(x = displ,
      y = hwy)+
  geom_point(aes(color = class))+
  geom_smooth()+
  facet_wrap(~year)+
  labs(x = "Engine Displacement (Liters)",
       y = "Highway MPG",
       title = "Car Mileage and Displacement",
       subtitle = "More Displacement Lowers Highwa
       caption = "Source: EPA",
       color = "Vehicle Class")+
  scale_color_viridis_d()+
  theme_economist()+
  theme(text = element_text(family = "Fira Sans"))
```



gg: Themes VIII

```r
library(ggthemes)
ggplot(data = mpg) +
  aes(x = displ,
      y = hwy) +
  geom_point(aes(color = class)) +
  geom_smooth() +
  facet_wrap(~year) +
  labs(x = "Engine Displacement (Liters)",
       y = "Highway MPG",
       title = "Car Mileage and Displacement",
       subtitle = "More Displacement Lowers Highwa
       caption = "Source: EPA",
       color = "Vehicle Class") +
  scale_color_viridis_d() +
  theme_fivethirtyeight() +
  theme(text = element_text(family = "Fira Sans"))
```



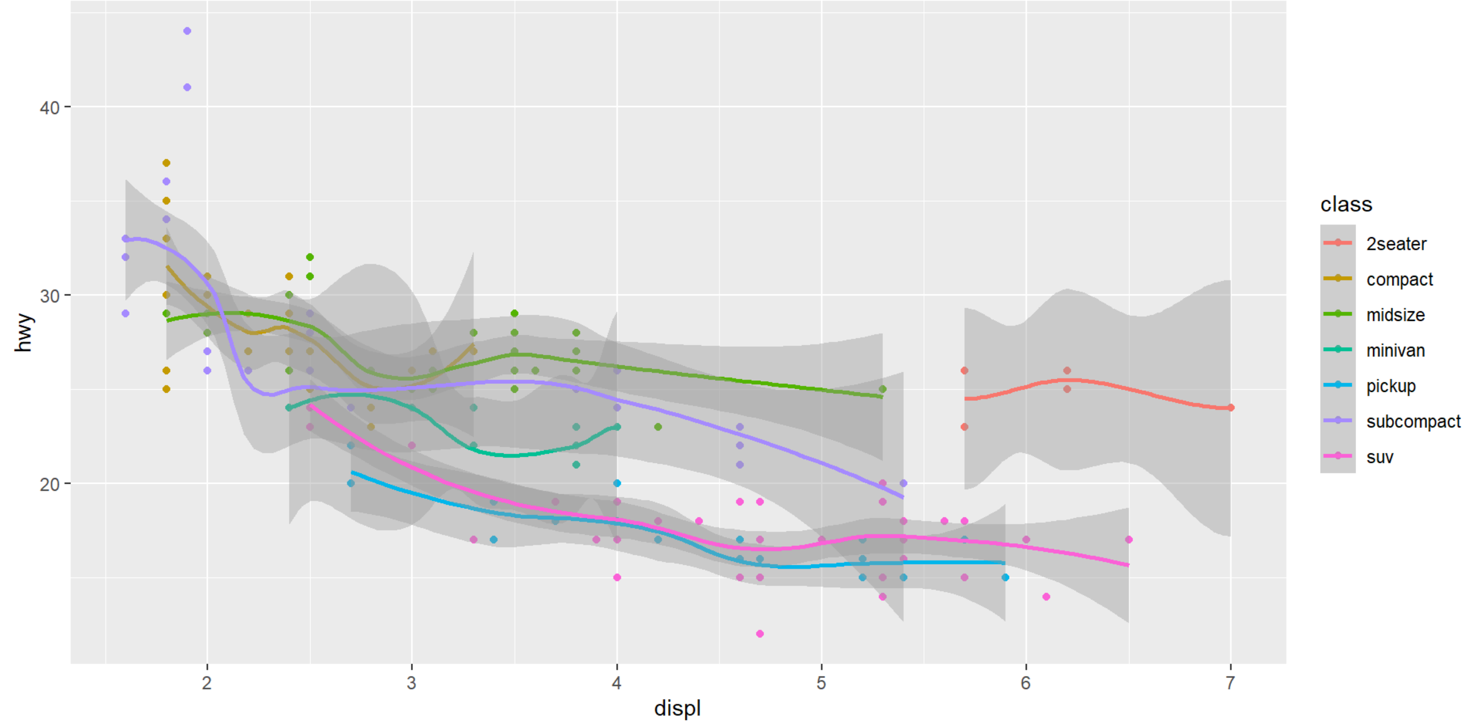
Some Troubleshooting

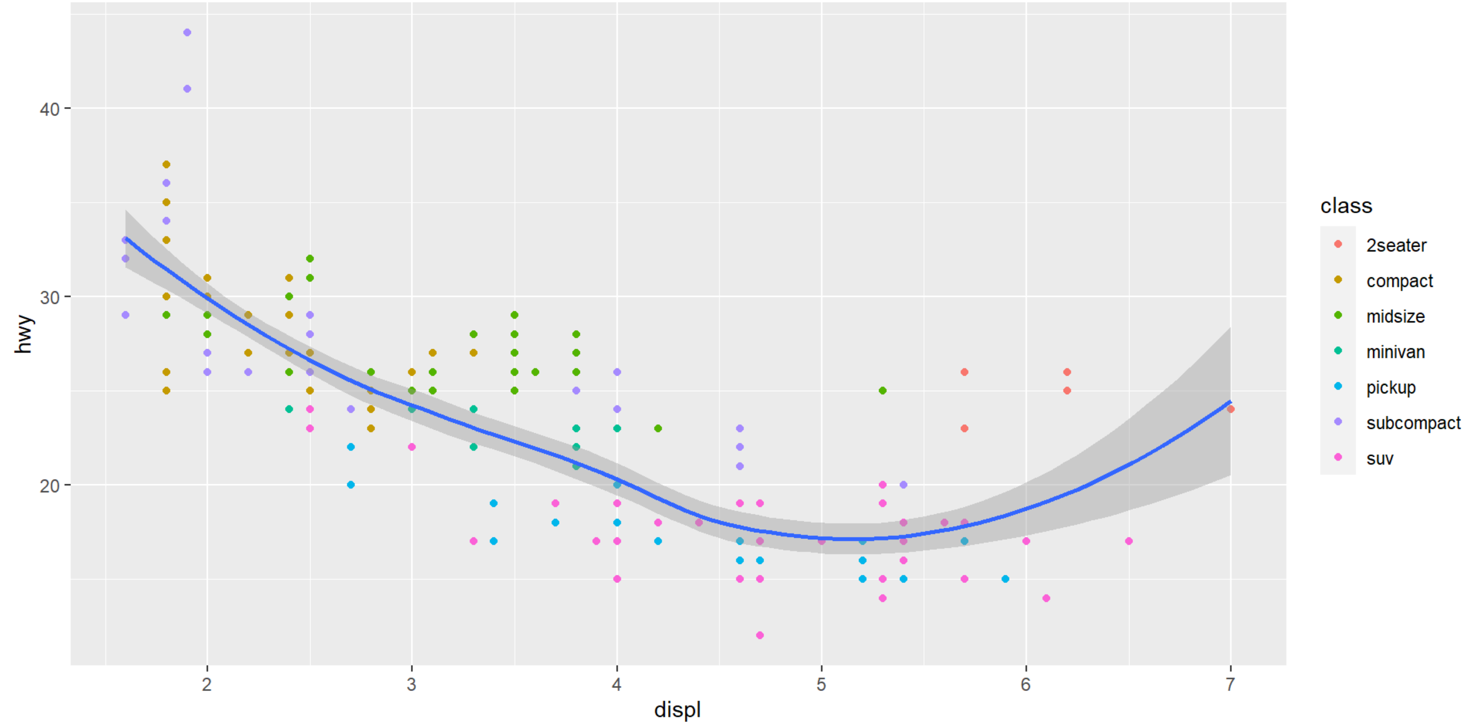
Global vs. Local Aesthetic Mappings

- `aes()` can go in base (`data`) layer and/or in individual `geom()` layers
- All `geoms` will inherit global `aes` from `data` layer unless overridden

```r
# ALL GEOMS will map data to colors
ggplot(data = mpg, aes(x = displ,
                       y = hwy,
                       color = class))+
  geom_point()+
  geom_smooth()
```

```r
# ONLY points will map data to colors
ggplot(data = mpg, aes(x = displ,
                       y = hwy))+
  geom_point(aes(color = class))+
  geom_smooth()
```





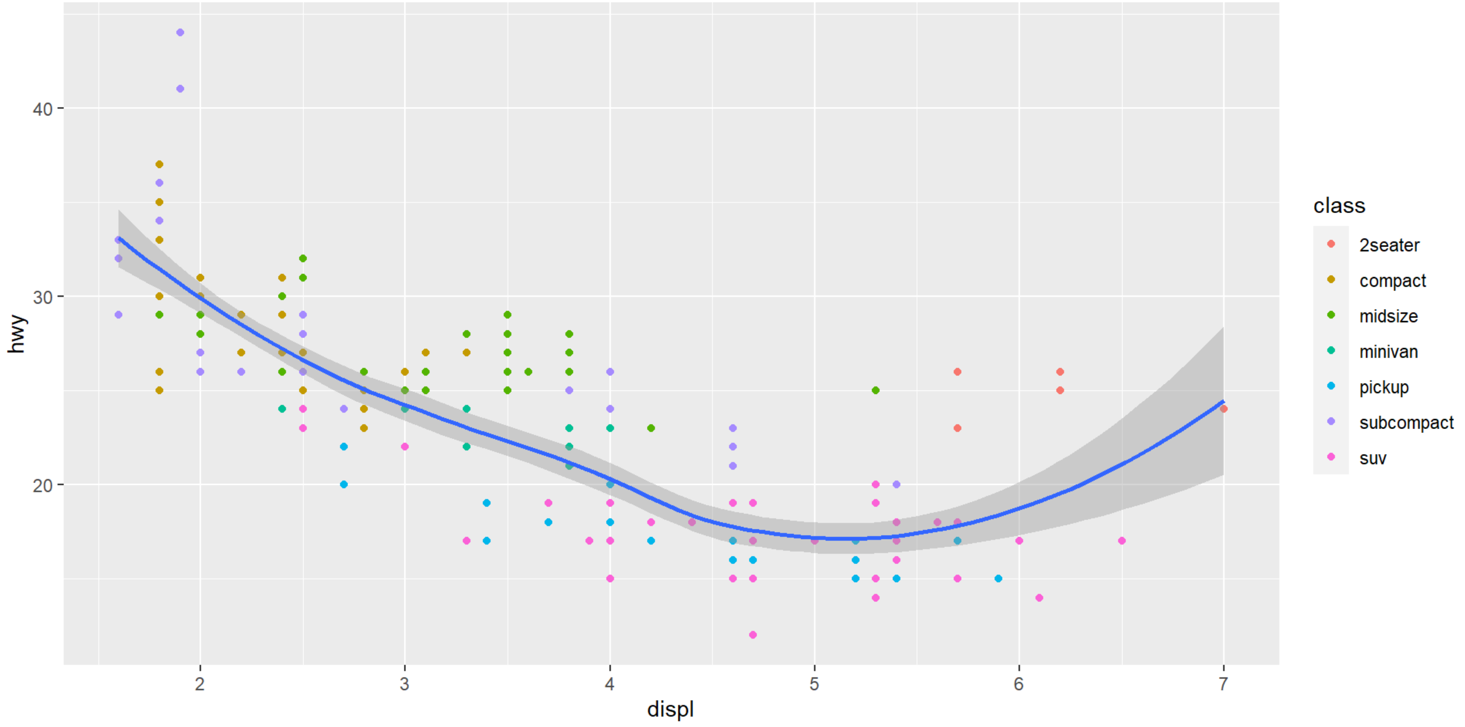
Zahid Asghar

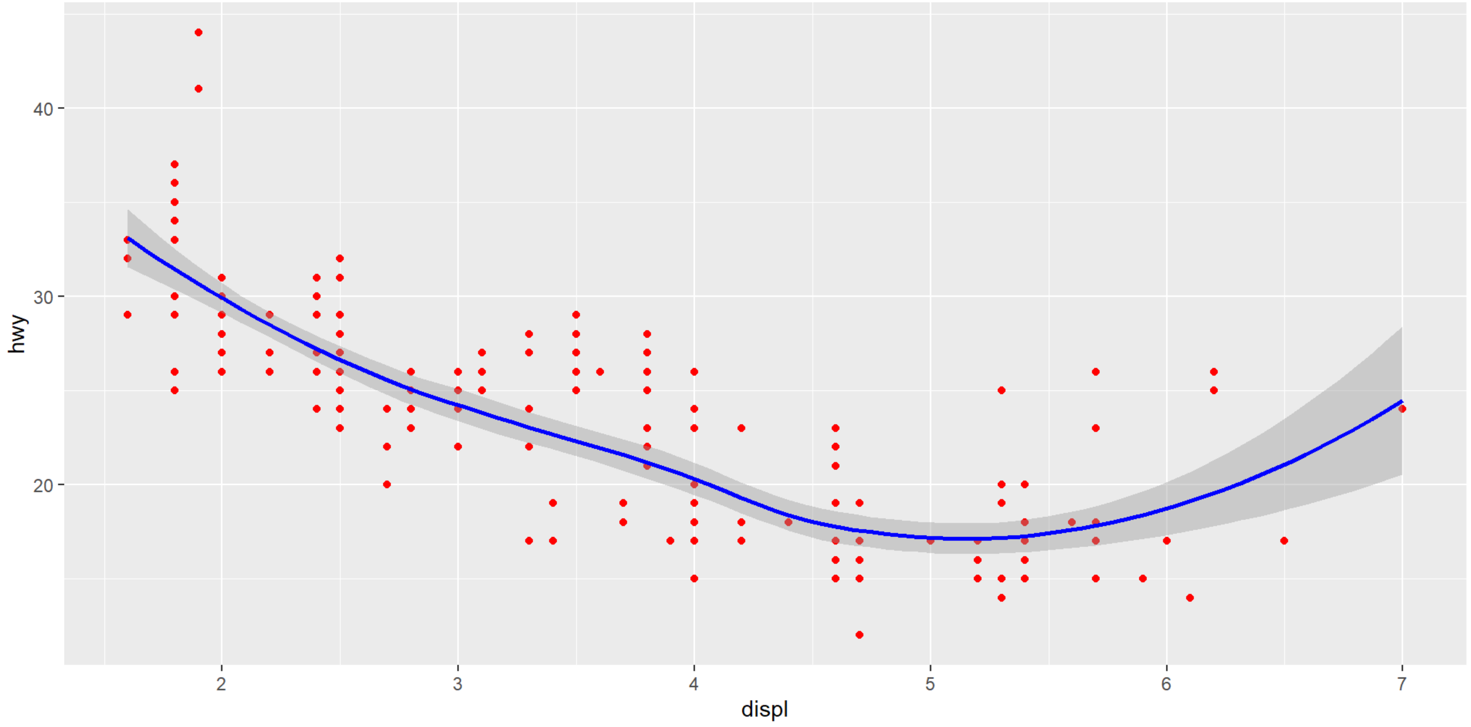
Mapped vs. Set Aesthetics

- `aes`thetics such as `size` and `color` can be mapped from data or set to a single value
- Map *inside* of `aes()`, set *outside* of `aes()`

```r
# Point colors are mapped from class data
ggplot(data = mpg, aes(x = displ,
                       y = hwy))+
  geom_point(aes(color = class))+
  geom_smooth()
```

```r
# Point colors are all set to blue
ggplot(data = mpg, aes(x = displ,
                       y = hwy))+
  geom_point(aes(), color = "red")+
  geom_smooth(aes(), color = "blue")
```



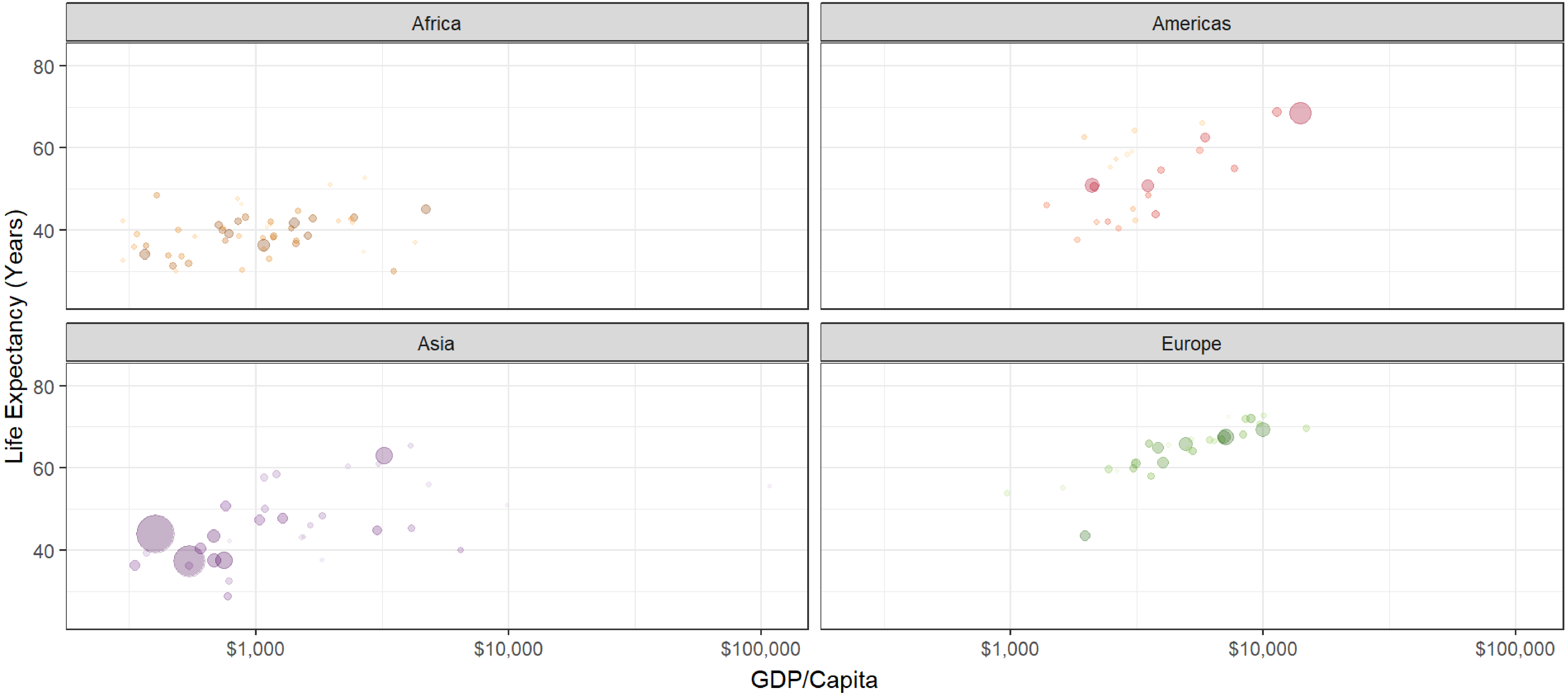


Go Crazy II

Output

Code

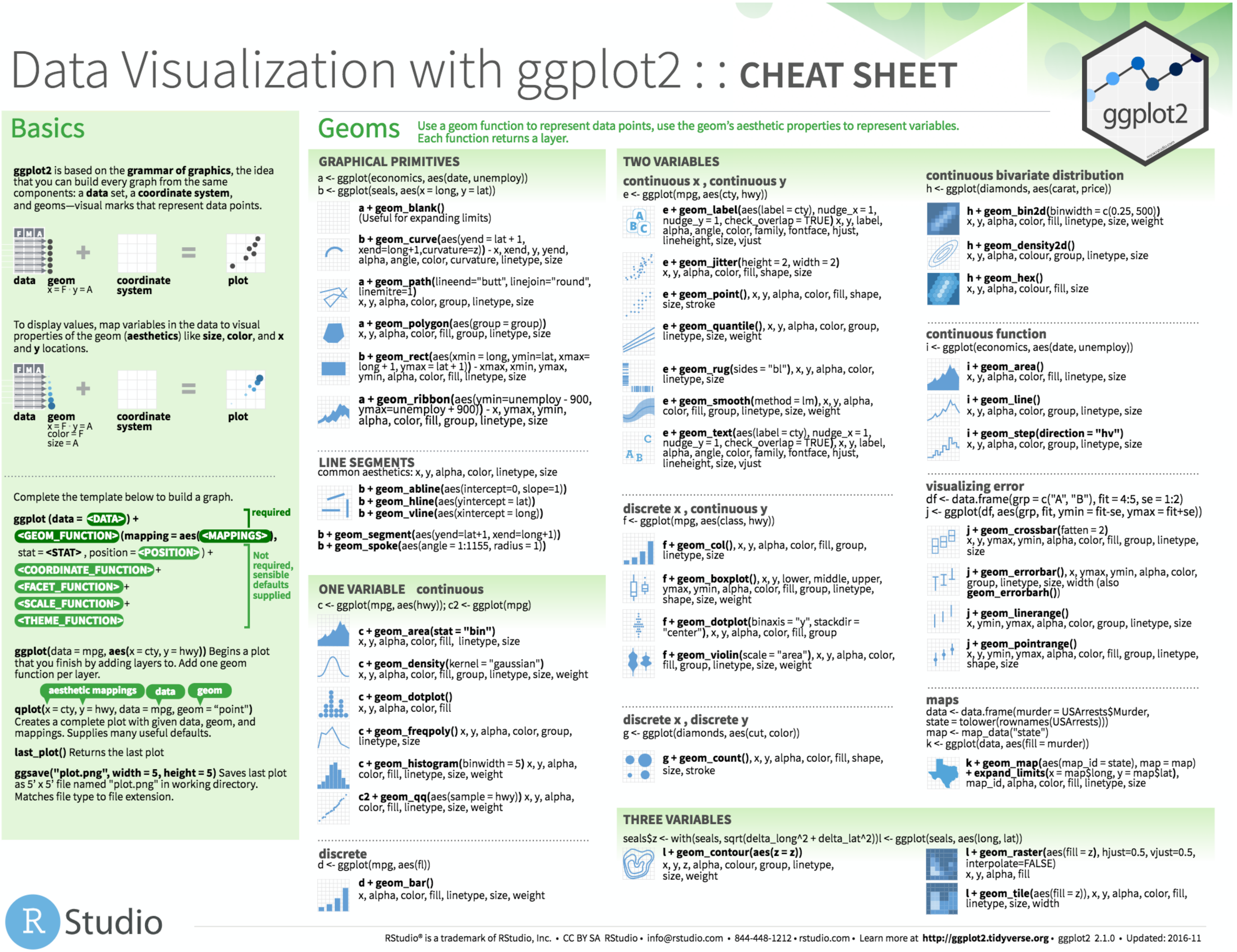
Income & Life Expectancy - 1952



Source: Hans Rosling's gapminder.org

Zahid Asghar

Reference: R Studio Makes Great “Cheat Sheet”s!



Data Visualization with ggplot2 : : CHEAT SHEET

ggplot2

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>) +                                        required
<GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
 stat = <STAT> , position = <POSITION>) +                        Not
<COORDINATE_FUNCTION>+                                           required,
<FACET_FUNCTION>+                                                sensible
<SCALE_FUNCTION>+                                                defaults
<THEME_FUNCTION>                                                 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.

aesthetic mappings | data | geom

qplot(x = cty, y = hwy, data = mpg, geom = “point”) Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

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.

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()
(Useful for expanding limits)

b + geom_curve(aes(yend = lat + 1, xend=long+1,curvature=z)**)** - 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(group = group)**)**
x, y, alpha, color, fill, group, 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_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(fl))

d + geom_bar()
x, alpha, color, fill, linetype, size, weight

TWO VARIABLES

continuous x , continuous y

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

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE**)** x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

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

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, check_overlap = TRUE**)**, x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

discrete x , continuous y

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

discrete x , discrete y

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

g + geom_count(), x, y, alpha, color, fill, shape, size, stroke

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_density2d()
x, y, alpha, colour, group, linetype, size

h + geom_hex()
x, y, alpha, colour, fill, size

continuous function

i <- 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

THREE VARIABLES

seals$z <- 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, colour, group, linetype, size, weight

l + geom_raster(aes(fill = z)**)**, hjust=0.5, vjust=0.5, interpolate=FALSE)
x, y, alpha, fill

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

R Studio

RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more at **http://ggplot2.tidyverse.org** • ggplot2 2.1.0 • Updated: 2016-11

RStudio: ggplot2 Cheat Sheet

Reference

On `ggplot2`

- **R Studio's ggplot2 Cheat Sheet**
- **`ggplot2`'s website reference section**
- Hadley Wickham's R for Data Science book chapter on ggplot2
- STHDA's be awesome in ggplot2
- r-statistic's top 50 ggplot2 visualizations

On data visualization