

Quantitative Methods in R Graphics and Statistics

Zahid Asghar



Lecture 4 — Data Graphics



Dr. Zahid Asghar
Professor of Economics
 zasghar@qau.edu.pk
 [zahedasghar](https://twitter.com/zahedasghar)
 zahidasghar.com

Graphics and Statistics

- Admittedly, we still need to cover basic descriptive statistics and data fundamentals
 - mean, median, variance, standard deviation
 - random variables, distributions, PDFs, Z-scores
 - bargraphs, boxplots, histograms, scatterplots
- All of this is coming in next few lecture
- But let's start with the fun stuff right away, even if you don't fully know the *reasons*: **data visualization**



Our Data Source

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

```
1 library(ggplot2)
2
3 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(15) f        18     29 p    compa...
2 audi          a4      1.8  1999     4 manual(m5) f        21     29 p    compa...
3 audi          a4      2.0  2008     4 manual(m6) f        20     31 p    compa...
4 audi          a4      2.0  2008     4 auto(av)   f        21     30 p    compa...
5 audi          a4      2.8  1999     6 auto(15)  f        16     26 p    compa...
6 audi          a4      2.8  1999     6 manual(m5) f        18     26 p    compa...
```

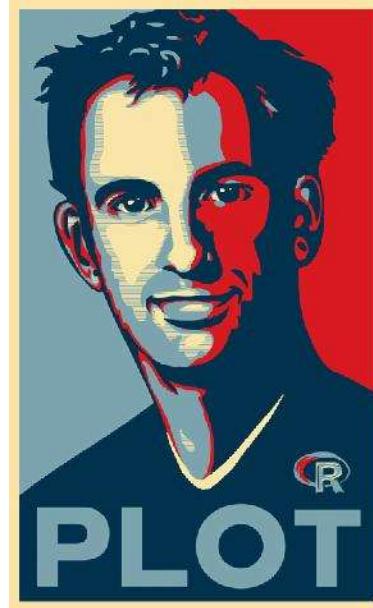
ggplot2 and the tidyverse

tidyverse

Zahid Asghar

ggplot2

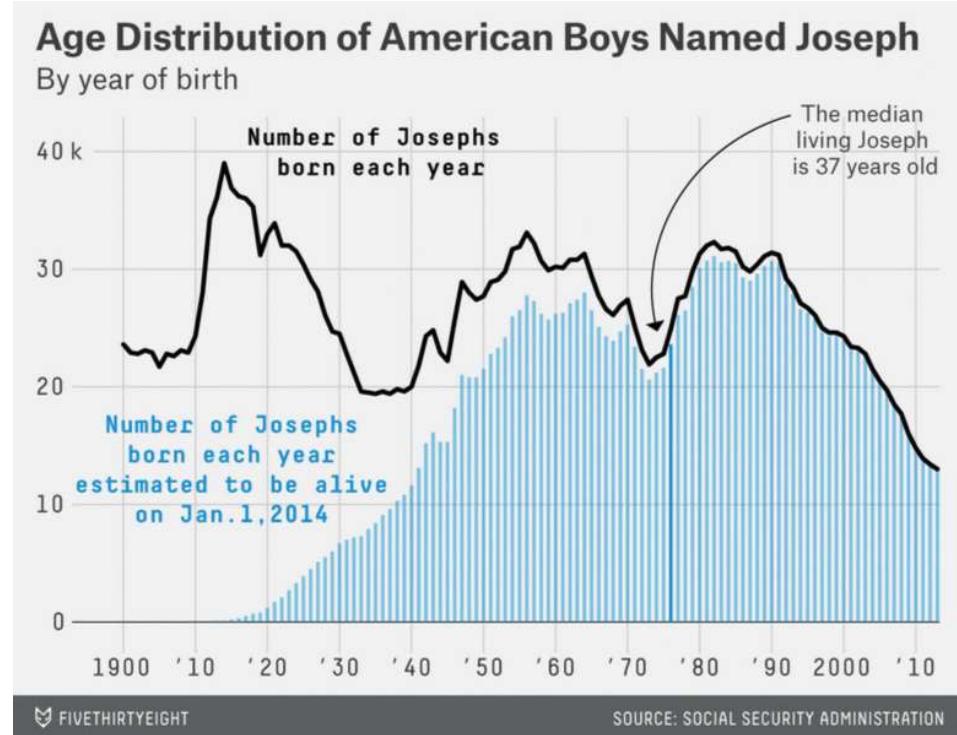
- `ggplot2` is perhaps the most popular package in `R` and a core element of the `tidyverse`
- `gg` stands for a **grammar of graphics**
- Very powerful and beautiful graphics, very customizable and reproducible, but requires a bit of a learning curve
- All those “cool graphics” you’ve seen in the New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



ggplot: All Your Figure are Belong to Us



Source: [fivethirtyeight](#)



Source: [fivethirtyeight](#)

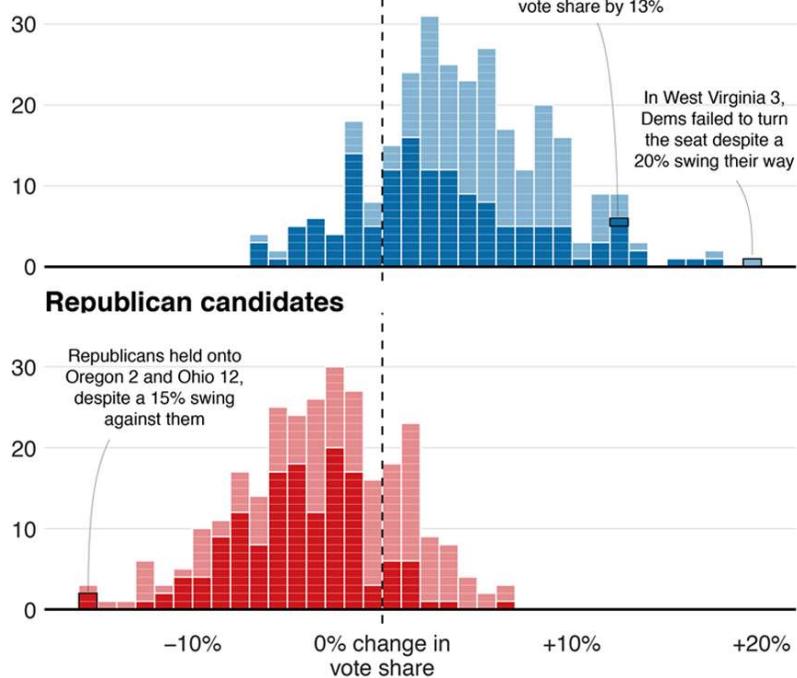
Zahid Asghar

ggplot: All Your Figure are Belong to Us

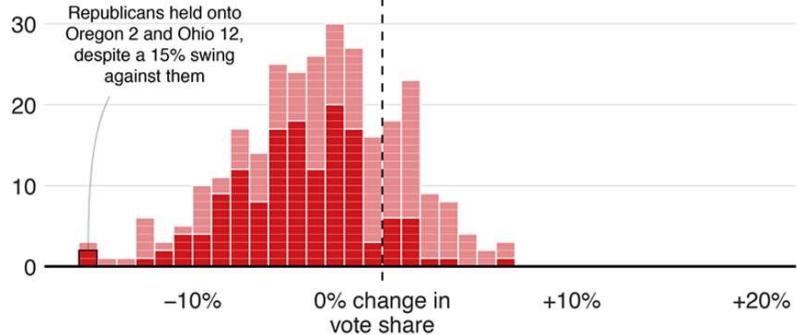
Blue wave

■ Won seat ■ Didn't win

Democrat candidates

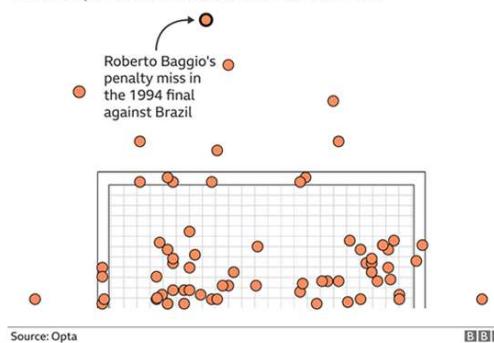


Republican candidates



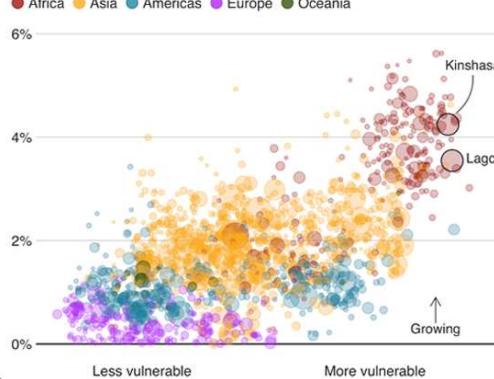
Where penalties are saved

World Cup shootout misses and saves, 1982-2014

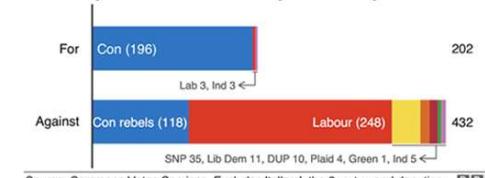


Fast-growing cities face worse climate risks

Population growth 2018-2035 over climate change vulnerability

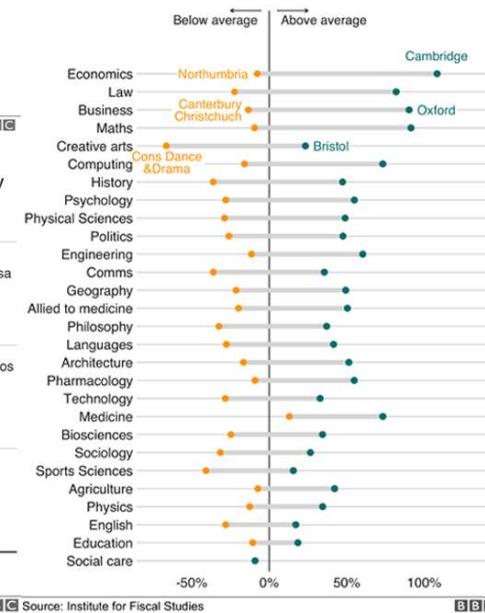


MPs rejected Theresa May's deal by 230 votes



Earnings vary across unis even within subjects

Impact on men's earnings relative to the average degree



Source: BBC's bbplot

Zahid Asghar

Why Go gg?



Hadley Wickham
Chief Scientist, R Studio

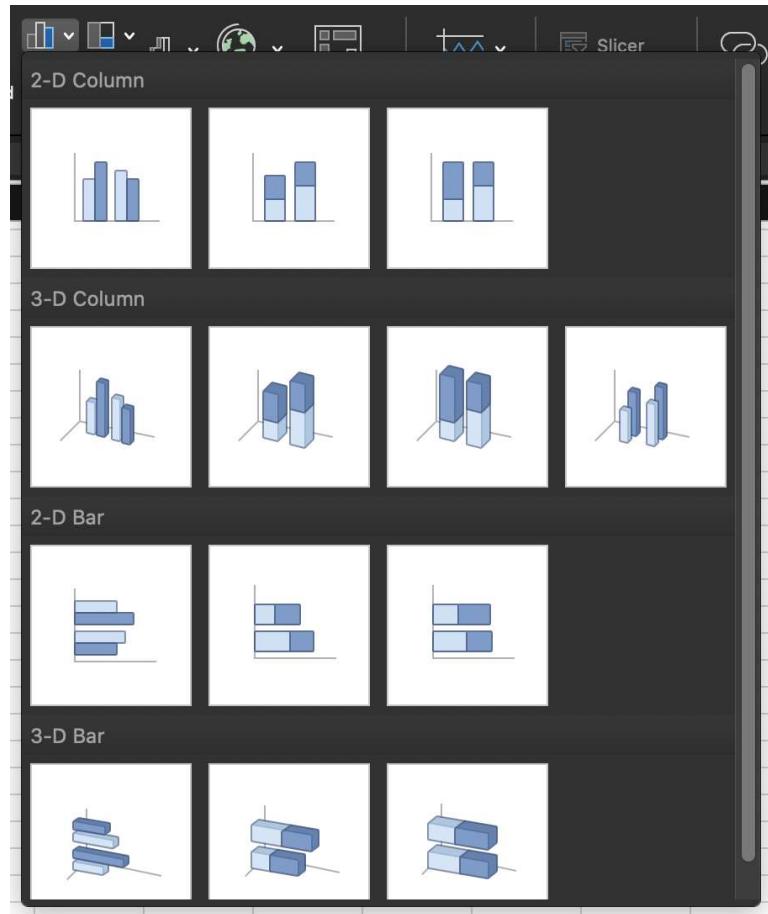
“The transferrable skills from ggplot2 are not the idiosyncrasies of plotting syntax, but a powerful way of thinking about visualisation, as a way of **mapping between variables and the visual properties of geometric objects** that you can perceive.”

Source

Zahid Asghar

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. **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. **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!

```
1 ggplot(data = mpg) +  
2   aes(x = displ,  
3         y = hwy) +  
4   geom_point() +  
5   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

```
1 # make and save plot as p
2 p <- ggplot(data = mpg) +
3   aes(x = displ,
4       y = hwy) +
5   geom_point()
6
7 p # view plot
8
9 # to add a layer...
10 p + geom_smooth() # shows the new plot
11
12 p <- p + geom_smooth() # overwrites p
13 p2 <- p + geom_smooth() # saves new object
```

Plot Layers

Zahid Asghar

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 “**tidy1:**

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

Zahid Asghar

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

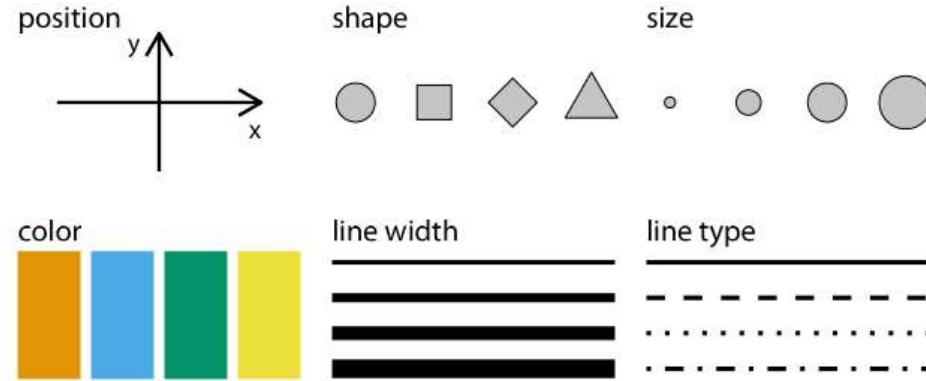
gg: Mapping Aesthetics I

Data

Aesthetics map data to visual elements or parameters

Aesthetics

+aes(...)



gg: Mapping Aesthetics II

Data

Aesthetics map data to visual elements or parameters

Aesthetics

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

`+aes(...)`

gg: Mapping Aesthetics III

Data

Aesthetics map data to visual elements or parameters

Aesthetics

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

+aes(...)

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

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

gg: Geoms I

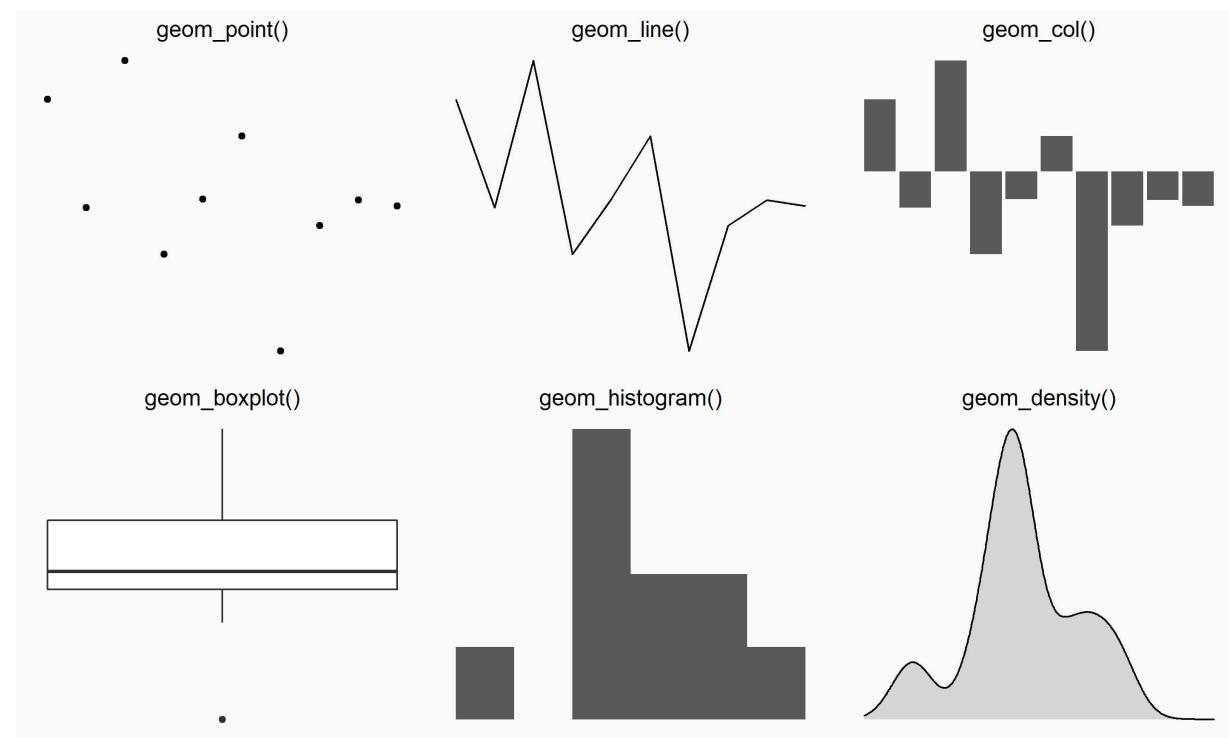
Data

Aesthetics

Geoms

+geom_*(...)

Geometric objects displayed on the plot



gg: Geoms II

Data

Aesthetics

Geoms

+geom_*(...)

Geometric objects displayed on the plot

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

Type	geom
Point	<code>geom_point()</code>
Line	<code>geom_line()</code> , <code>geom_path()</code>
Bar	<code>geom_bar()</code> , <code>geom_col()</code>
Histogram	<code>geom_histogram()</code>
Regression	<code>geom_smooth()</code>
Boxplot	<code>geom_boxplot()</code>
Text	<code>geom_text()</code>
Density	<code>geom_density()</code>

gg: Geoms III

Data

Aesthetics

Geoms

+geom_*(...)

Geometric objects displayed on the plot

1	## [1]	"geom_abline"	"geom_area"	"geom_bar"	"geom_bin2"
2	## [5]	"geom_blank"	"geom_boxplot"	"geom_col"	"geom_cont"
3	## [9]	"geom_count"	"geom_crossbar"	"geom_curve"	"geom_dens"
4	## [13]	"geom_density_2d"	"geom_density2d"	"geom_dotplot"	"geom_errd"
5	## [17]	"geom_errorbarh"	"geom_freqpoly"	"geom_hex"	"geom_hist"
6	## [21]	"geom_hline"	"geom_jitter"	"geom_label"	"geom_line"
7	## [25]	"geom_linerange"	"geom_map"	"geom_path"	"geom_poin"
8	## [29]	"geom_pointrange"	"geom_polygon"	"geom_qq"	"geom_qq_1"
9	## [33]	"geom_quantile"	"geom_raster"	"geom_rect"	"geom_ribb"
10	## [37]	"geom_rug"	"geom_segment"	"geom_sf"	"geom_sf_1"
11	## [41]	"geom_sf_text"	"geom_smooth"	"geom_spoke"	"geom_step"
12	## [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

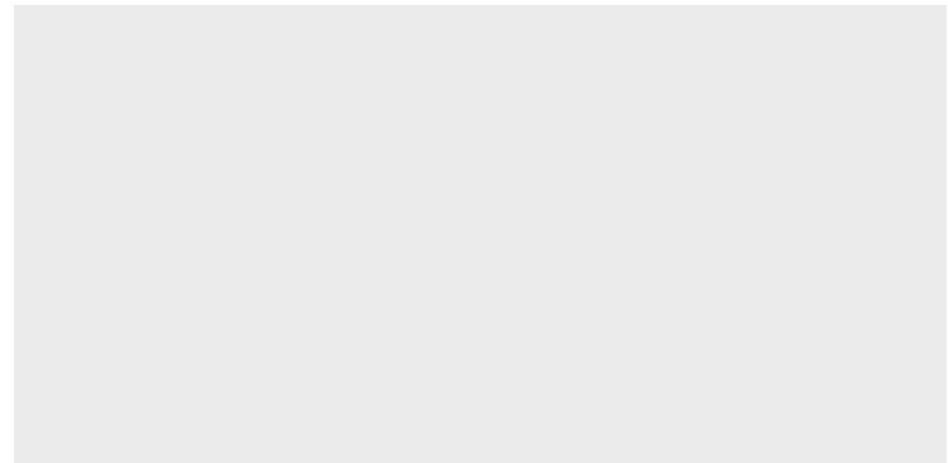
Or just start typing `geom_` in R Studio!

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



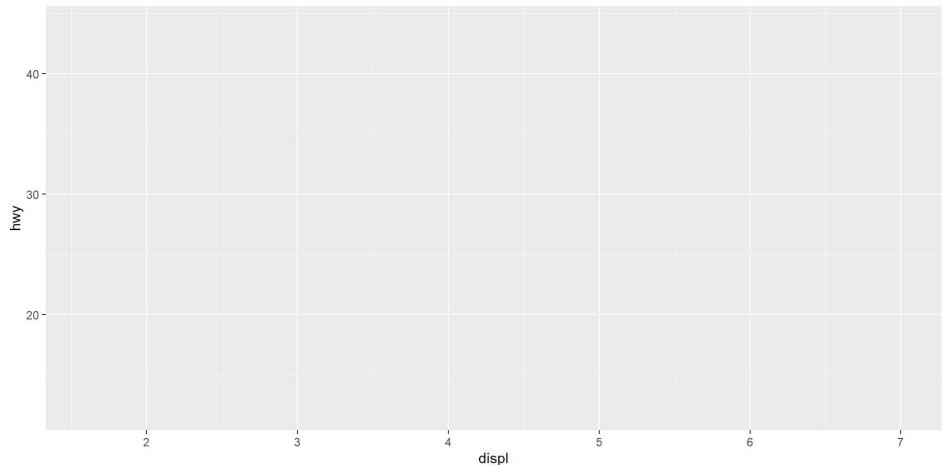
Let's Make a Plot!

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



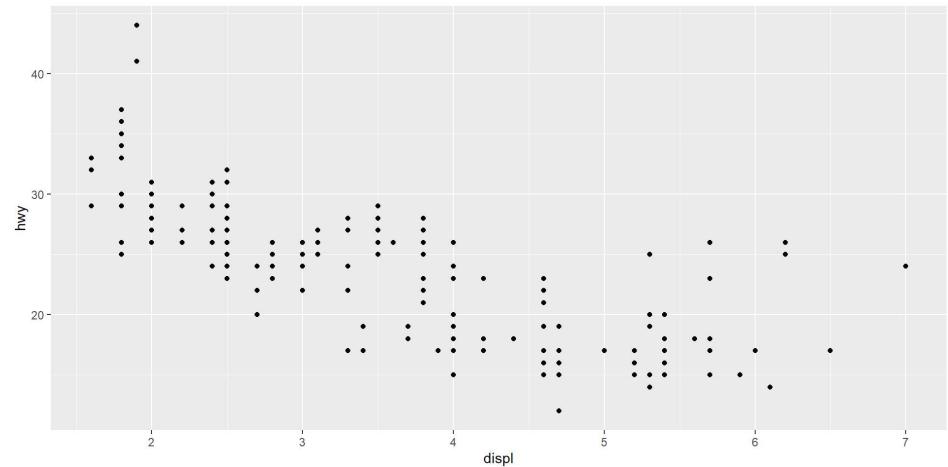
Let's Make a Plot!

```
1 ggplot(data = mpg) +  
2   aes(x = displ,  
3       y = hwy)
```



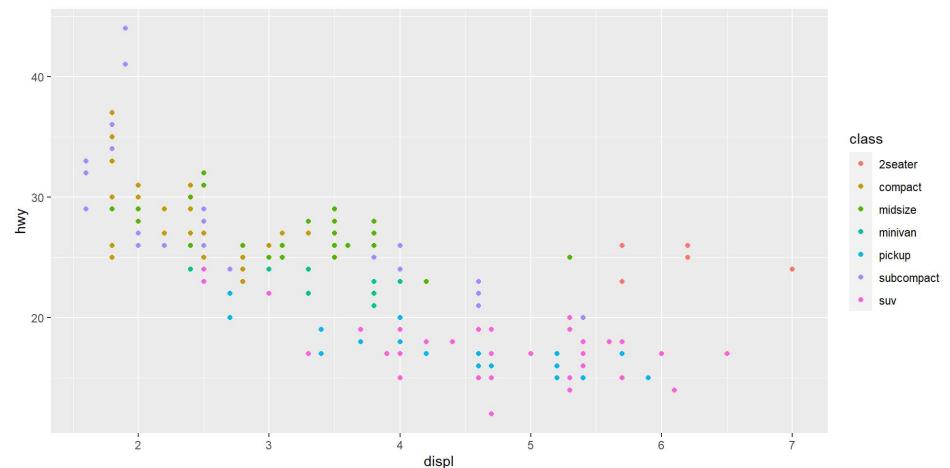
Let's Make a Plot!

```
1 ggplot(data = mpg) +  
2   aes(x = displ,  
3       y = hwy) +  
4   geom_point()
```



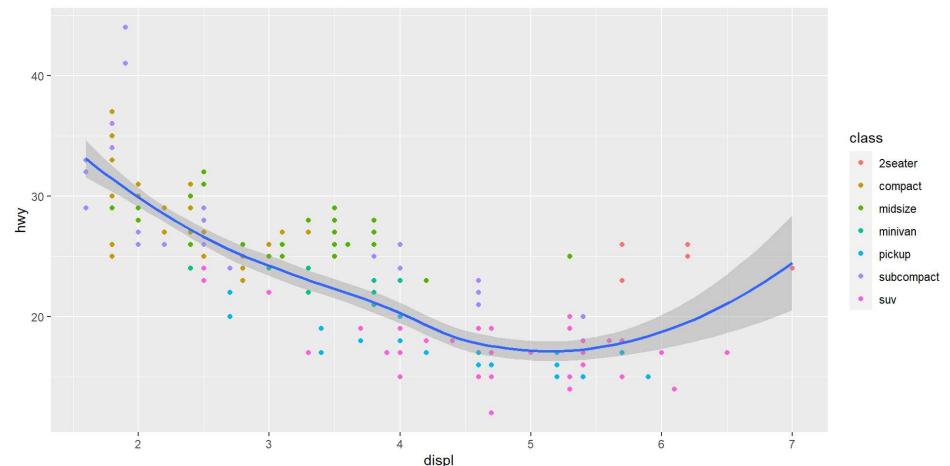
Let's Make a Plot!

```
1 ggplot(data = mpg) +  
2   aes(x = displ,  
3       y = hwy) +  
4   geom_point(aes(color = class))
```



Let's Make a Plot!

```
1 ggplot(data = mpg) +  
2   aes(x = displ,  
3       y = hwy) +  
4   geom_point(aes(color = class)) +  
5   geom_smooth()
```



More Geoms

Data

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

Aesthetics

- `data`: geoms can have their own data
 - has to map onto global coordinates

Geoms

- `aes`: geoms can have their own aesthetics
 - inherits global aesthetics by default
 - different geoms have different available aesthetics

`+geom_*(...)`

More Geoms II

Data

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

Aesthetics

- `stat`: some geoms statistically transform data

- `geom_histogram()` uses `stat_bin()` to group observations into bins

Geoms

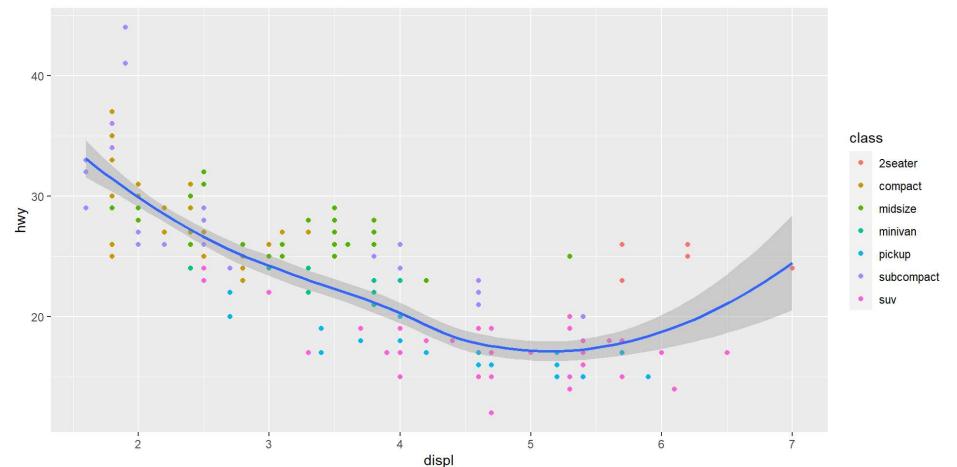
`+geom_*(...)`

- `position`: some adjust location of objects

- `dodge, stack, jitter`

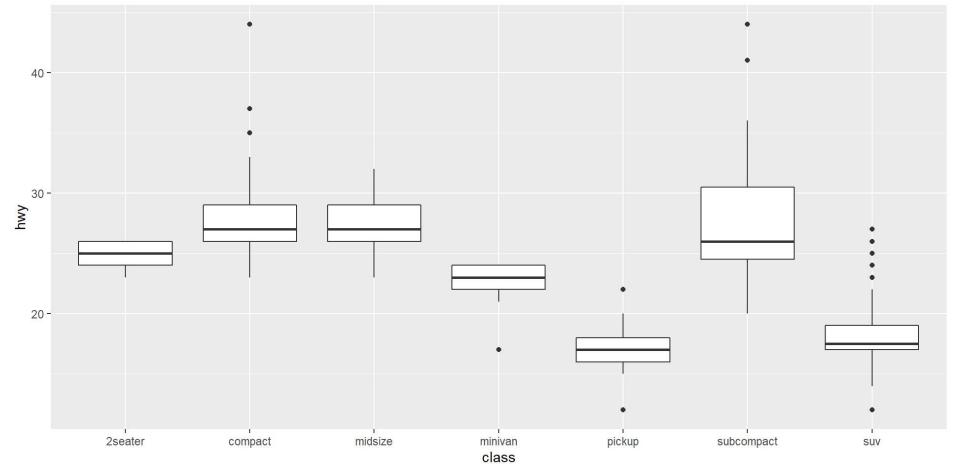
Our Plot

```
1 ggplot(data = mpg) +  
2   aes(x = displ,  
3       y = hwy) +  
4   geom_point(aes(color = class)) +  
5   geom_smooth()
```



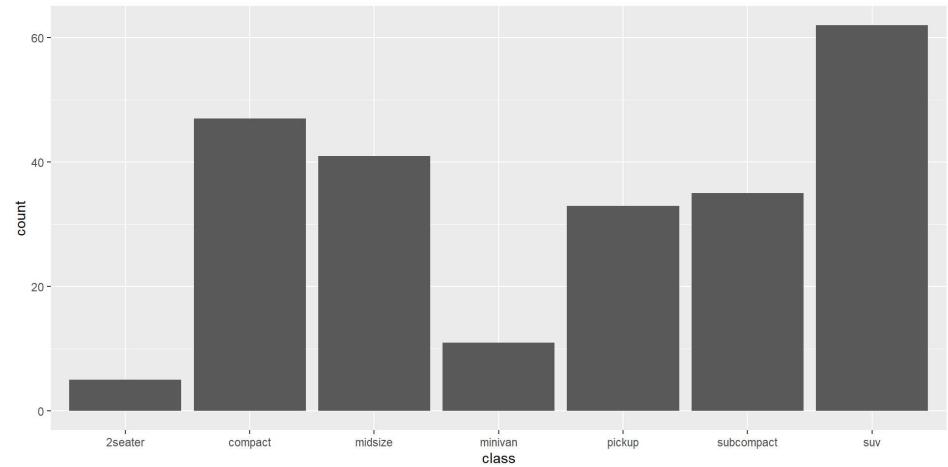
Change Our Plot

```
1 ggplot(data = mpg) +  
2   aes(x = class,  
3       y = hwy) +  
4   geom_boxplot()
```



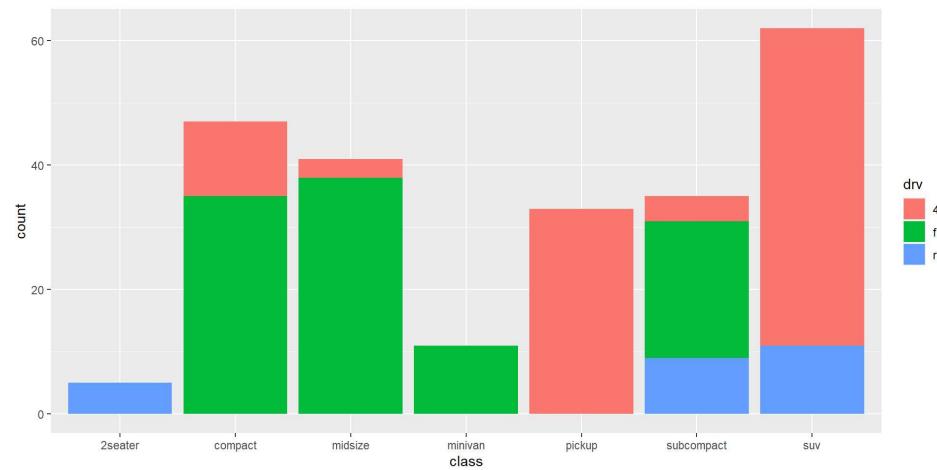
Change Our Plot

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



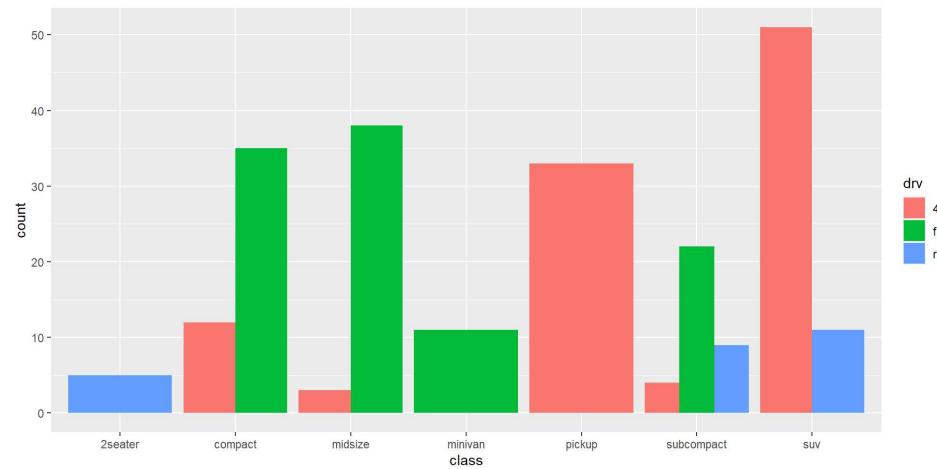
Change Our Plot

```
1 ggplot(data = mpg) +  
2   aes(x = class,  
3         fill = drv) +  
4   geom_bar()
```



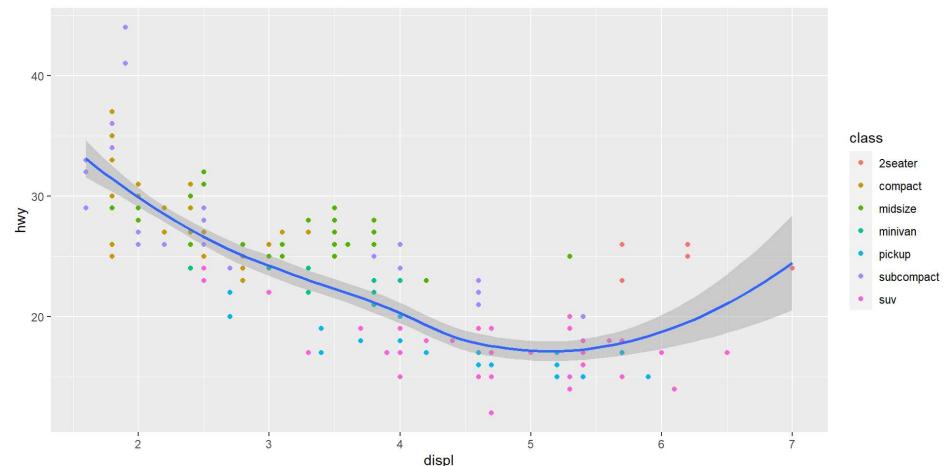
Change Our Plot

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



Back to the Original (and Saving It)

```
1 # save plot as p
2 p <- ggplot(data = mpg) +
3   aes(x = displ,
4       y = hwy) +
5   geom_point(aes(color = class)) +
6   geom_smooth()
7
8 p # show plot
```



gg: Facets I

Data

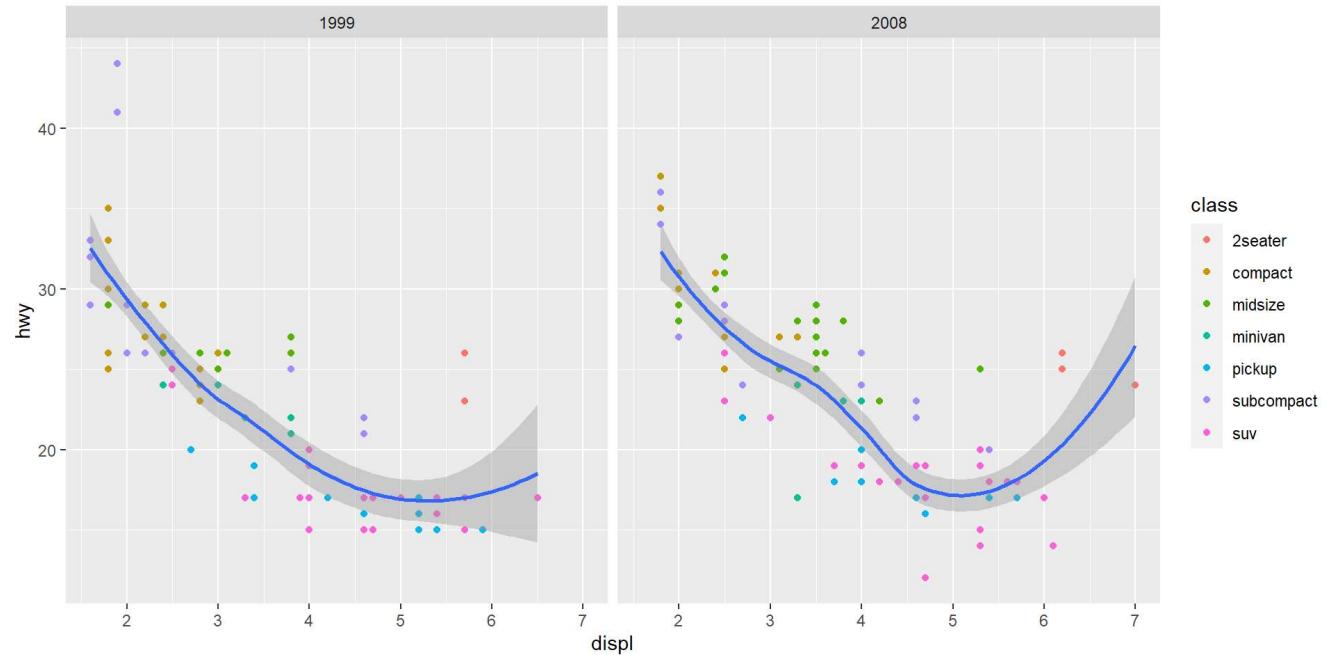
Aesthetics

Geoms

Facets

- + `facet_wrap()`
- + `facet_grid()`

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



gg: Facets II

Data

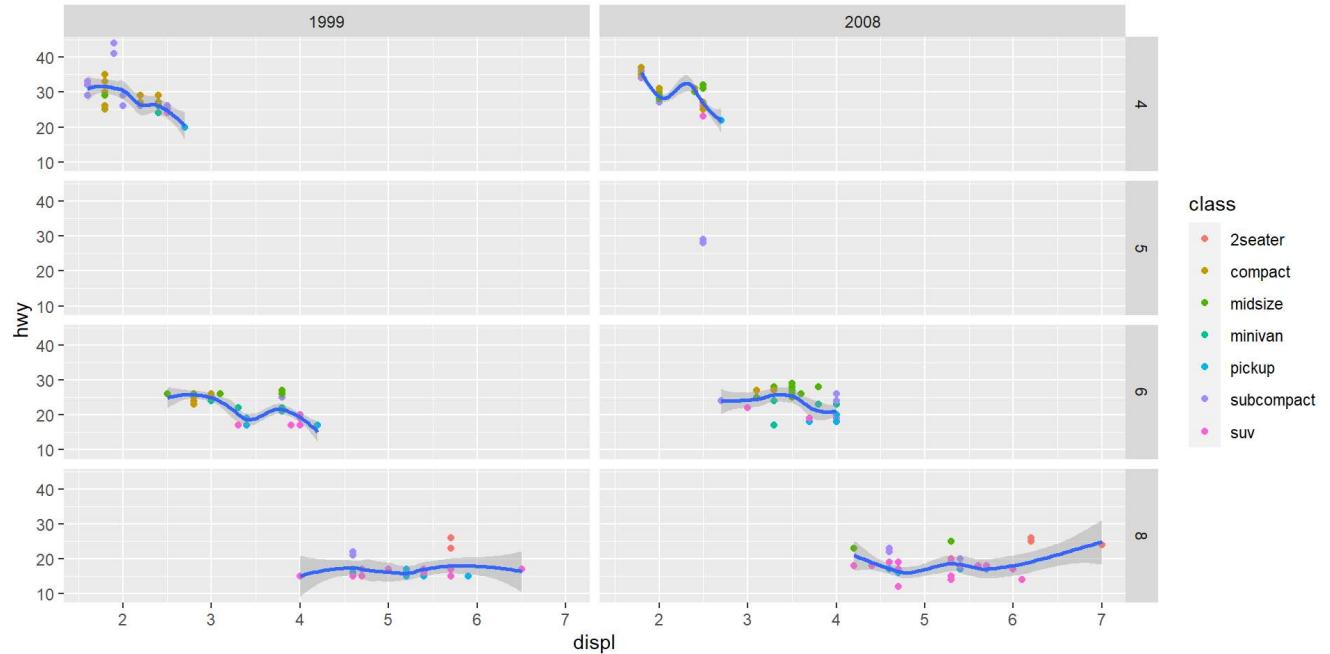
Aesthetics

Geoms

Facets

- + `facet_wrap()`
- + `facet_grid()`

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



gg: Labels

Data

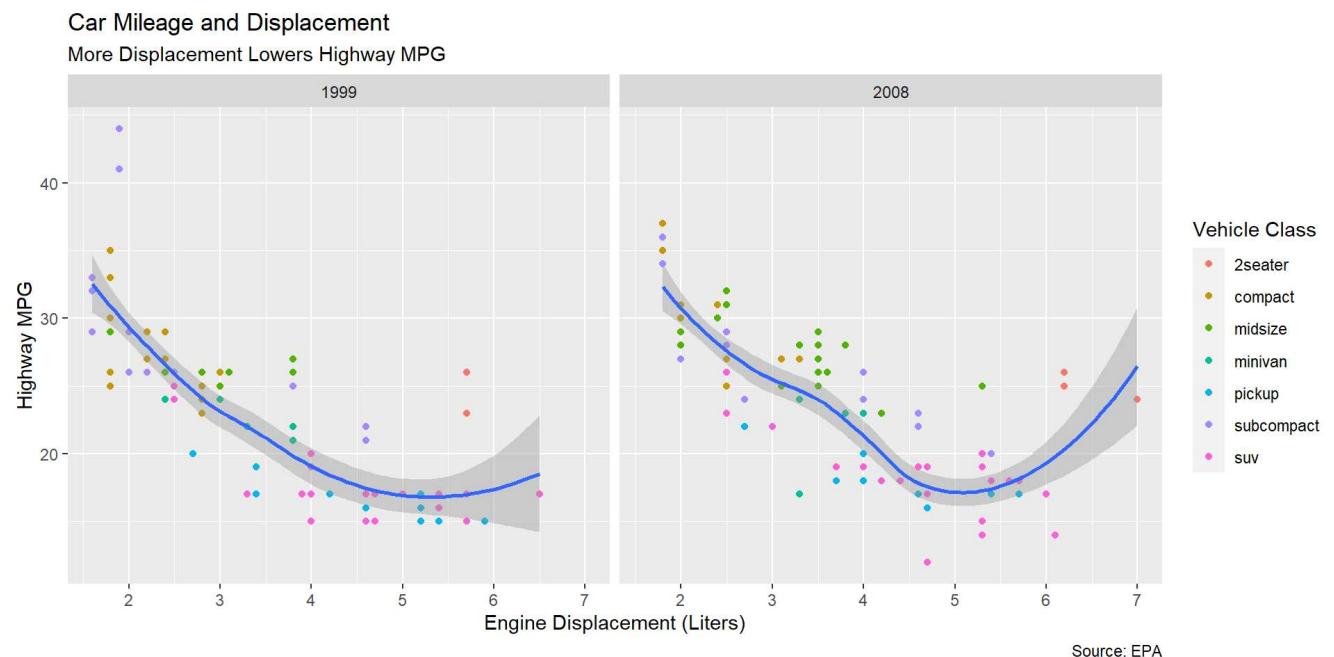
Aesthetics

Geoms

Facets

+ labs()

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



Zahid Asghar

gg: Scales I

Data

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

Aesthetics

- `<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()`

Geoms

Facets

Scales

+ `scale_*_*()`

gg: Scales II

Data

Aesthetics

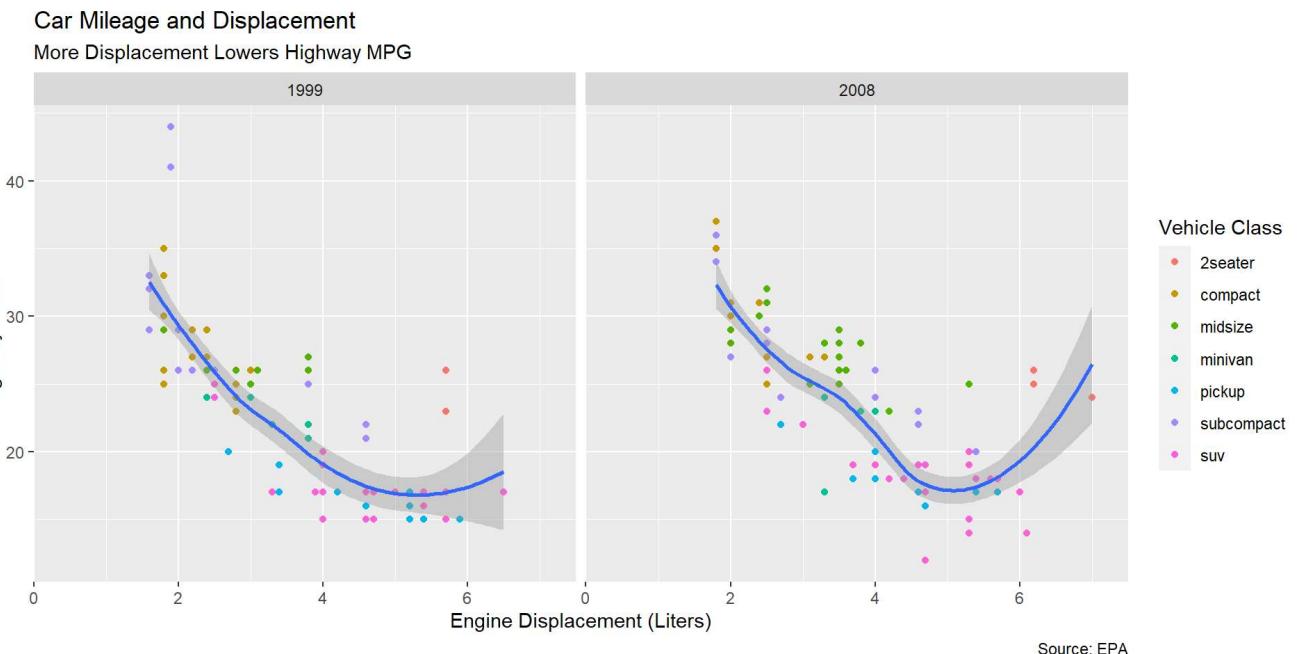
Geoms

Facets

Scales

+ `scale_*_*()`

```
1 p + scale_x_continuous(breaks = seq(0, 10, 2),
2                             limits = c(0,7.5),
3                             expand = c(0,0)
4 )
```



Zahid Asghar

gg: Scales II

Data

Aesthetics

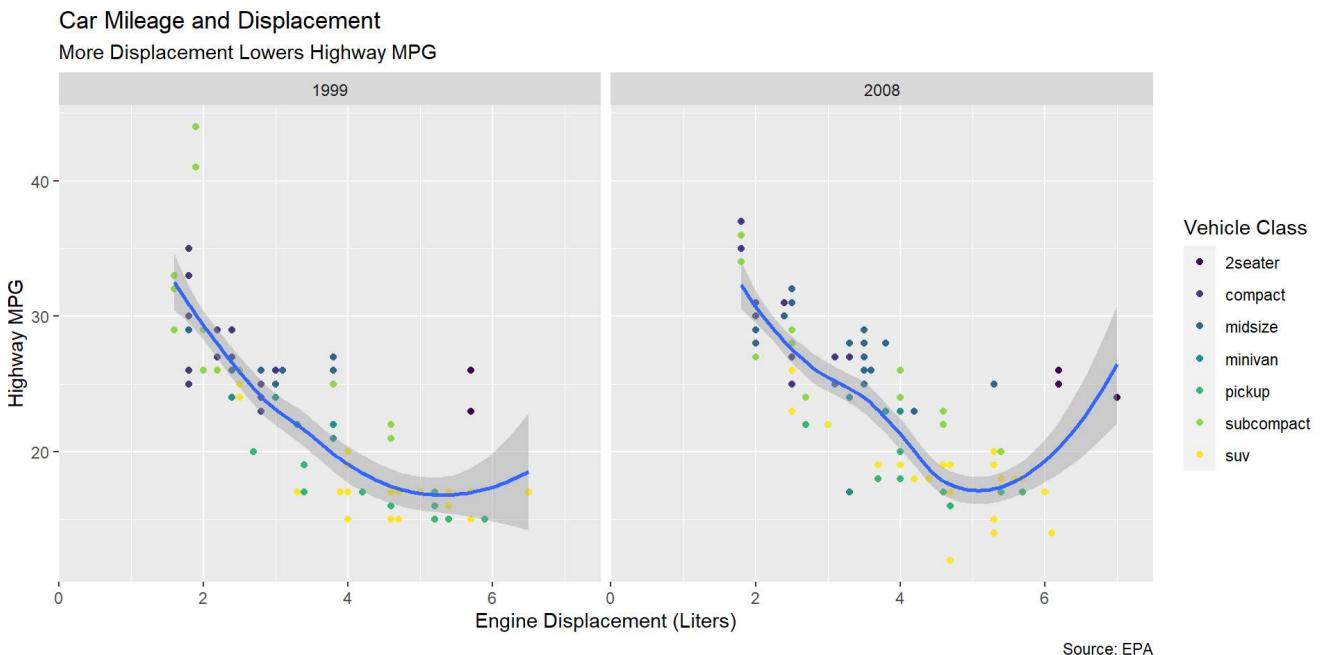
Geoms

Facets

Scales

+ `scale_*_*()`

```
1 p + scale_x_continuous(breaks = seq(0, 10, 2),
2                             limits = c(0,7.5),
3                             expand = c(0,0)
4 )
5     scale_color_viridis_d()
```



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

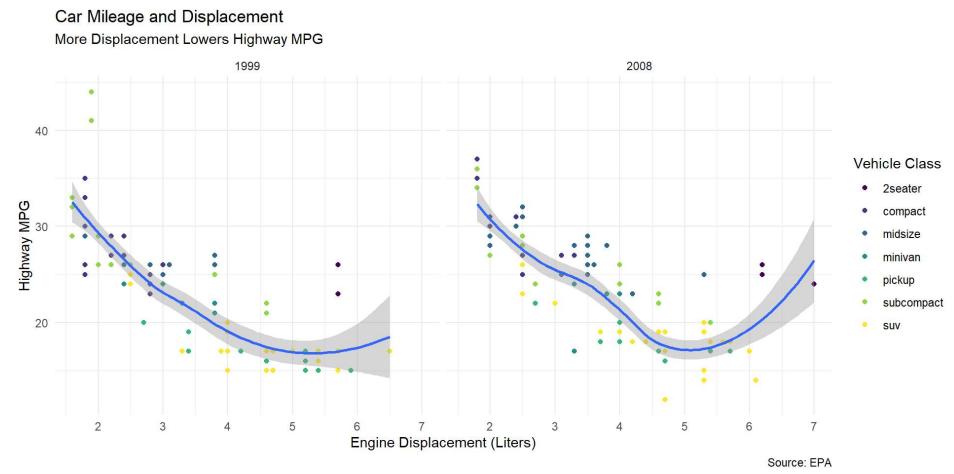
+ `theme_*`()

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

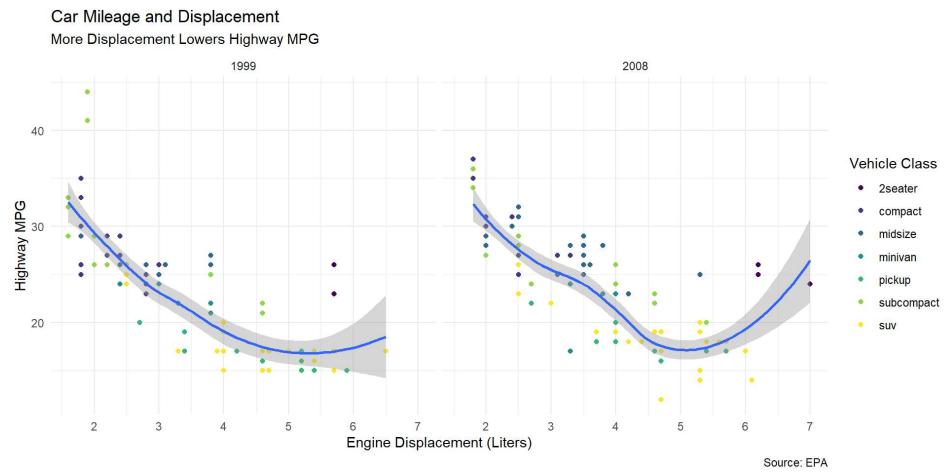
gg: Themes III

```
1 ggplot(data = mpg) +  
2   aes(x = displ,  
3       y = hwy) +  
4   geom_point(aes(color = class)) +  
5   geom_smooth() +  
6   facet_wrap(~year) +  
7   labs(x = "Engine Displacement (Liters)",  
8       y = "Highway MPG",  
9       title = "Car Mileage and Displacement",  
10      subtitle = "More Displacement Lowers Highway MPG",  
11      caption = "Source: EPA",  
12      color = "Vehicle Class") +  
13      scale_color_viridis_d() +  
14      theme_minimal()
```



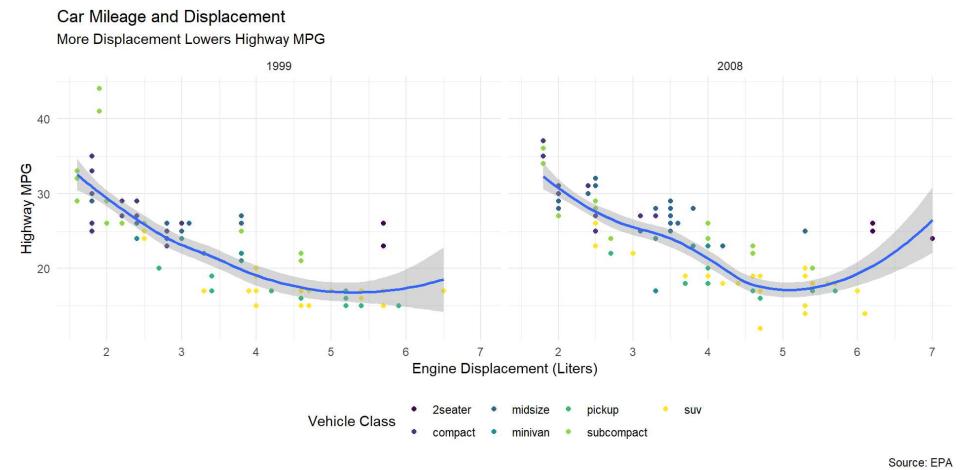
gg: Themes IV

```
1 ggplot(data = mpg) +  
2   aes(x = displ,  
3       y = hwy) +  
4   geom_point(aes(color = class)) +  
5   geom_smooth() +  
6   facet_wrap(~year) +  
7   labs(x = "Engine Displacement (Liters)",  
8       y = "Highway MPG",  
9       title = "Car Mileage and Displacement",  
10      subtitle = "More Displacement Lowers Highway MPG",  
11      caption = "Source: EPA",  
12      color = "Vehicle Class") +  
13      scale_color_viridis_d() +  
14      theme_minimal() +  
15      theme(text = element_text(family = "Fira Sans")) )
```



gg: Themes V

```
1 ggplot(data = mpg) +  
2   aes(x = displ,  
3       y = hwy) +  
4   geom_point(aes(color = class)) +  
5   geom_smooth() +  
6   facet_wrap(~year) +  
7   labs(x = "Engine Displacement (Liters)",  
8       y = "Highway MPG",  
9       title = "Car Mileage and Displacement",  
10      subtitle = "More Displacement Lowers Highway MPG",  
11      caption = "Source: EPA",  
12      color = "Vehicle Class") +  
13      scale_color_viridis_d() +  
14      theme_minimal() +  
15      theme(text = element_text(family = "Fira Sans"),  
16              legend.position = "bottom")
```



gg: Themes VI

Data

Aesthetics

Geoms

Facets

Scales

Themes

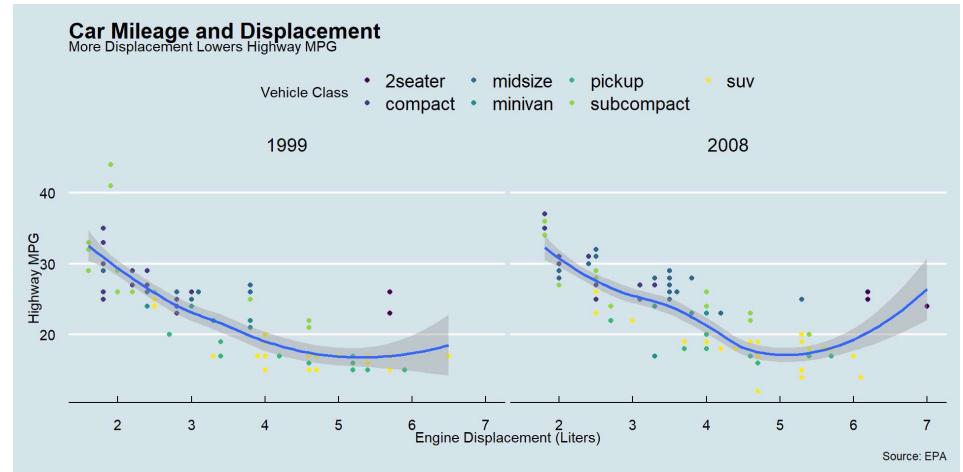
+ `theme_*`()

- `ggthemes` package adds some other nice themes

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

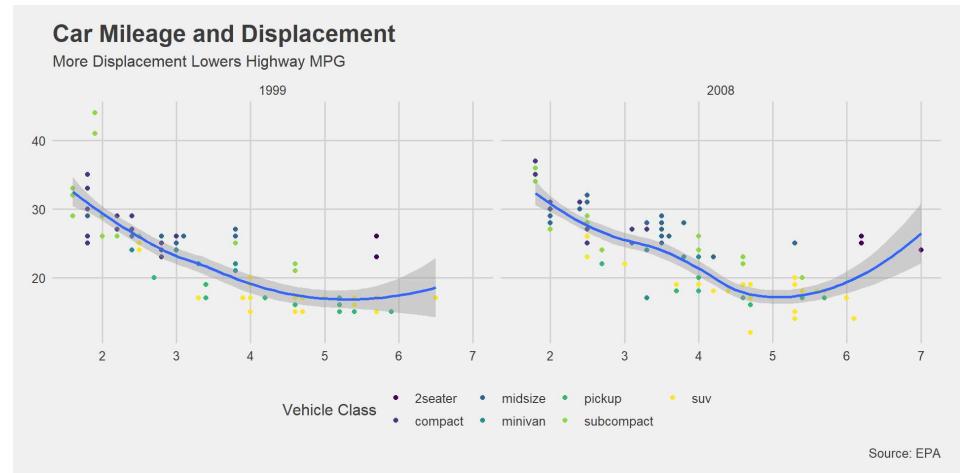
gg: Themes VII

```
1 library(ggthemes)
2 ggplot(data = mpg) +
3   aes(x = displ,
4       y = hwy) +
5   geom_point(aes(color = class)) +
6   geom_smooth() +
7   facet_wrap(~year) +
8   labs(x = "Engine Displacement (Liters)",
9        y = "Highway MPG",
10       title = "Car Mileage and Displacement",
11       subtitle = "More Displacement Lowers Highway MPG",
12       caption = "Source: EPA",
13       color = "Vehicle Class") +
14   scale_color_viridis_d() +
15   theme_economist() +
16   theme(text = element_text(family = "Fira Sans"))
```



gg: Themes VIII

```
1 library(ggthemes)
2 ggplot(data = mpg) +
3   aes(x = displ,
4       y = hwy) +
5   geom_point(aes(color = class)) +
6   geom_smooth() +
7   facet_wrap(~year) +
8   labs(x = "Engine Displacement (Liters)",
9        y = "Highway MPG",
10       title = "Car Mileage and Displacement",
11       subtitle = "More Displacement Lowers Highway MPG",
12       caption = "Source: EPA",
13       color = "Vehicle Class") +
14   scale_color_viridis_d() +
15   theme_fivethirtyeight() +
16   theme(text = element_text(family = "Fira Sans"))
```



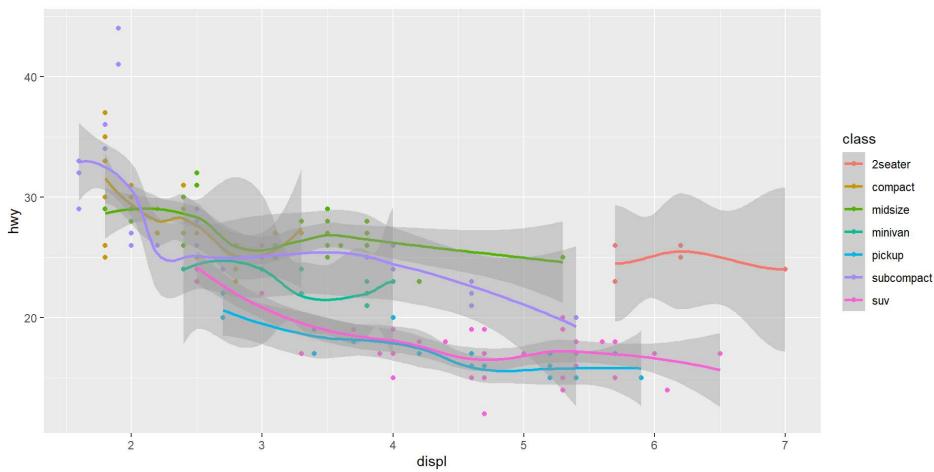
Some Troubleshooting

Zahid Asghar

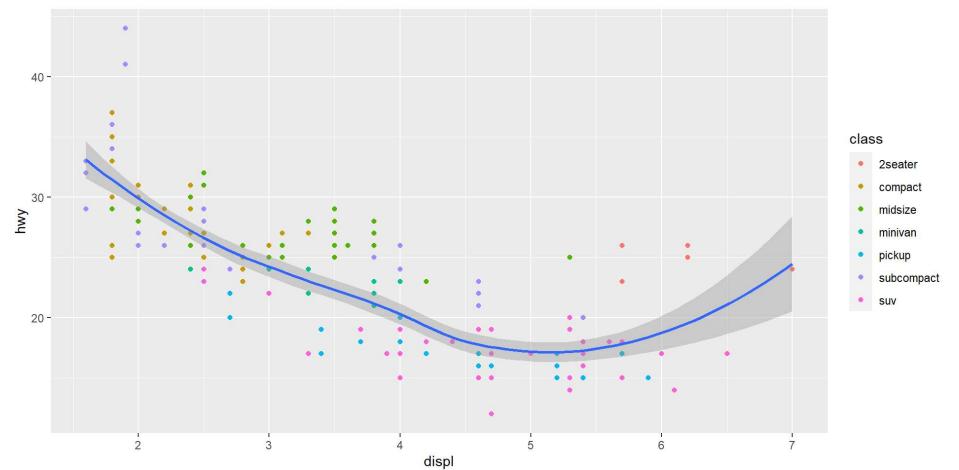
Global vs. Local Aesthetic Mappings

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

```
1 # ALL GEOMS will map data to colors
2 ggplot(data = mpg, aes(x = displ,
3                         y = hwy,
4                         color = class)) +
5   geom_point() +
6   geom_smooth()
```



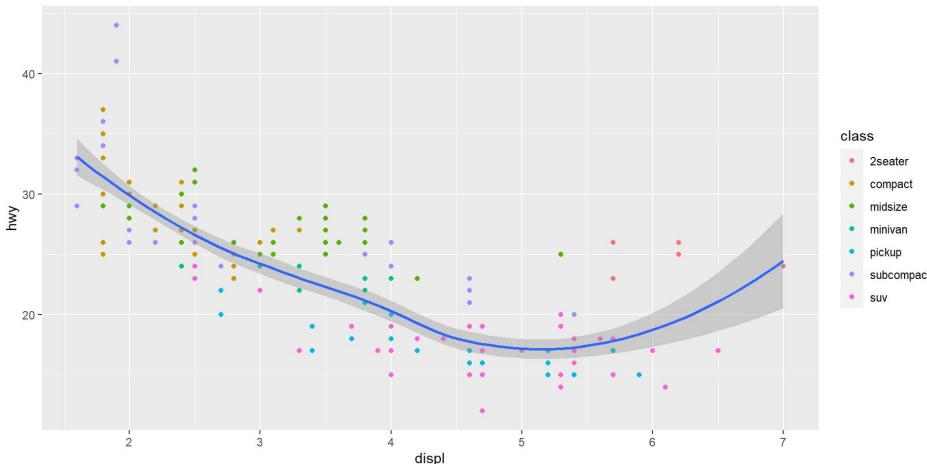
```
1 # ONLY points will map data to colors
2 ggplot(data = mpg, aes(x = displ,
3                         y = hwy)) +
4   geom_point(aes(color = class)) +
5   geom_smooth()
```



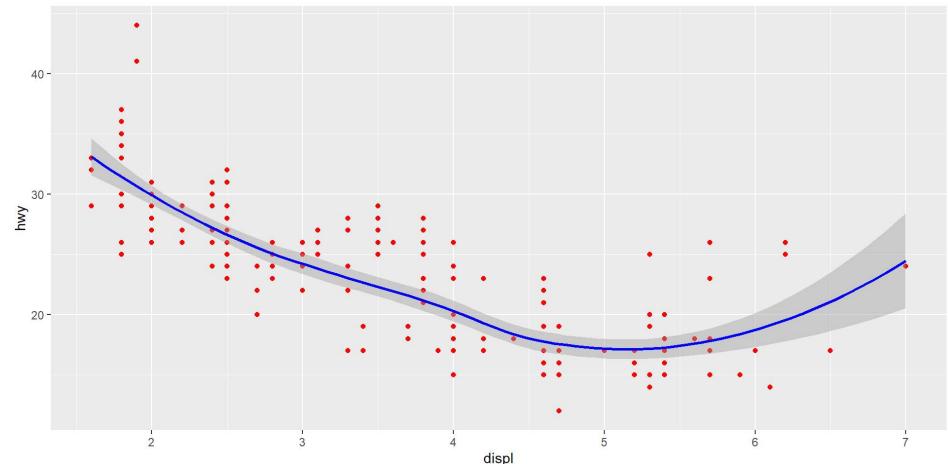
Mapped vs. Set Aesthetics

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

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

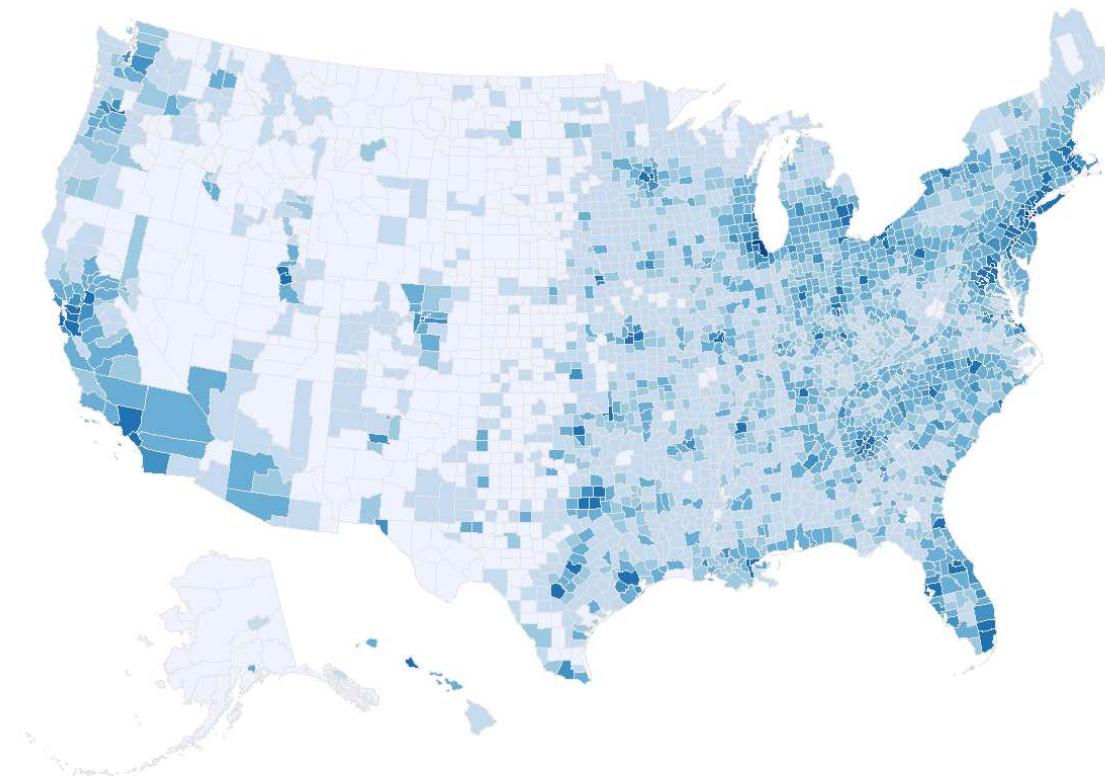


```
1 # Point colors are all set to blue
2 ggplot(data = mpg, aes(x = displ,
3                         y = hwy)) +
4   geom_point(aes(), color = "red") +
5   geom_smooth(aes(), color = "blue")
```



Go Crazy I

Output Code



Population per square mile

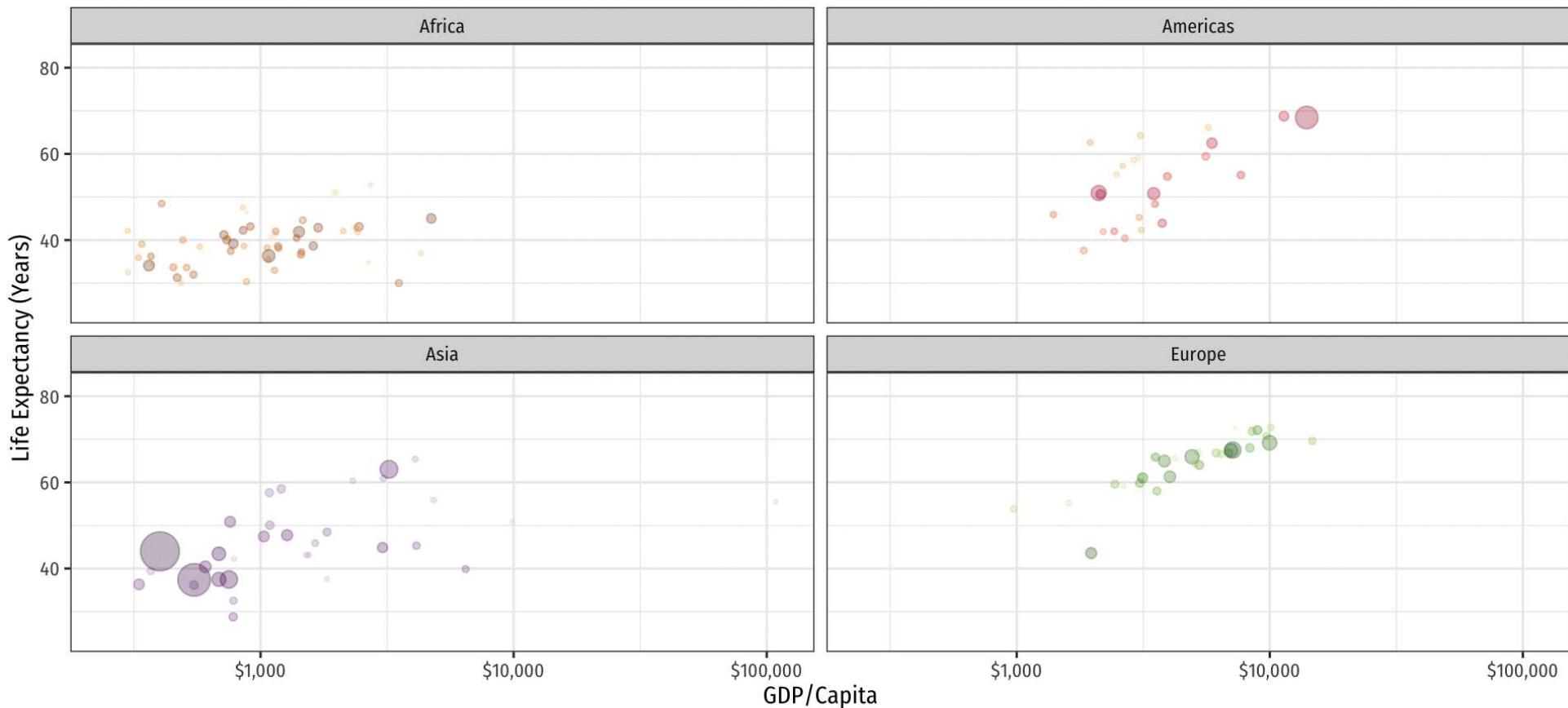
0-10	10-50	50-100	100-500	500-1,000	1,000-5,000	>5,000
------	-------	--------	---------	-----------	-------------	--------

Zahid Asghar

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



The ggplot2 cheat sheet is a comprehensive reference guide for creating data visualizations in R. It is organized into several sections:

- Basics:** Explains the grammar of graphics and provides a template for building a plot.
- Geoms:** A large section detailing various geometric primitives and their properties. It includes:
 - GRAPHICAL PRIMITIVES:** Data, geom, coordinate system, plot.
 - TWO VARIABLES:** continuous x, continuous y.
 - continuous bivariate distribution:** binwidth, density2d, point, quantile, rug, smooth, text.
 - continuous function:** area, line, step.
 - discrete x, continuous y:** abline, hline, line, segment, spoke.
 - discrete x, discrete y:** bar, count, dotplot, histogram, violin.
 - ONE VARIABLE:** continuous, discrete.
 - discrete:** binwidth, contour, raster, tile.
- Three-variable section:** details three-variable plots.

Each section includes code snippets and small images illustrating the different types of plots and their components.



RStudio: ggplot2 Cheat Sheet

Zahid Asghar

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