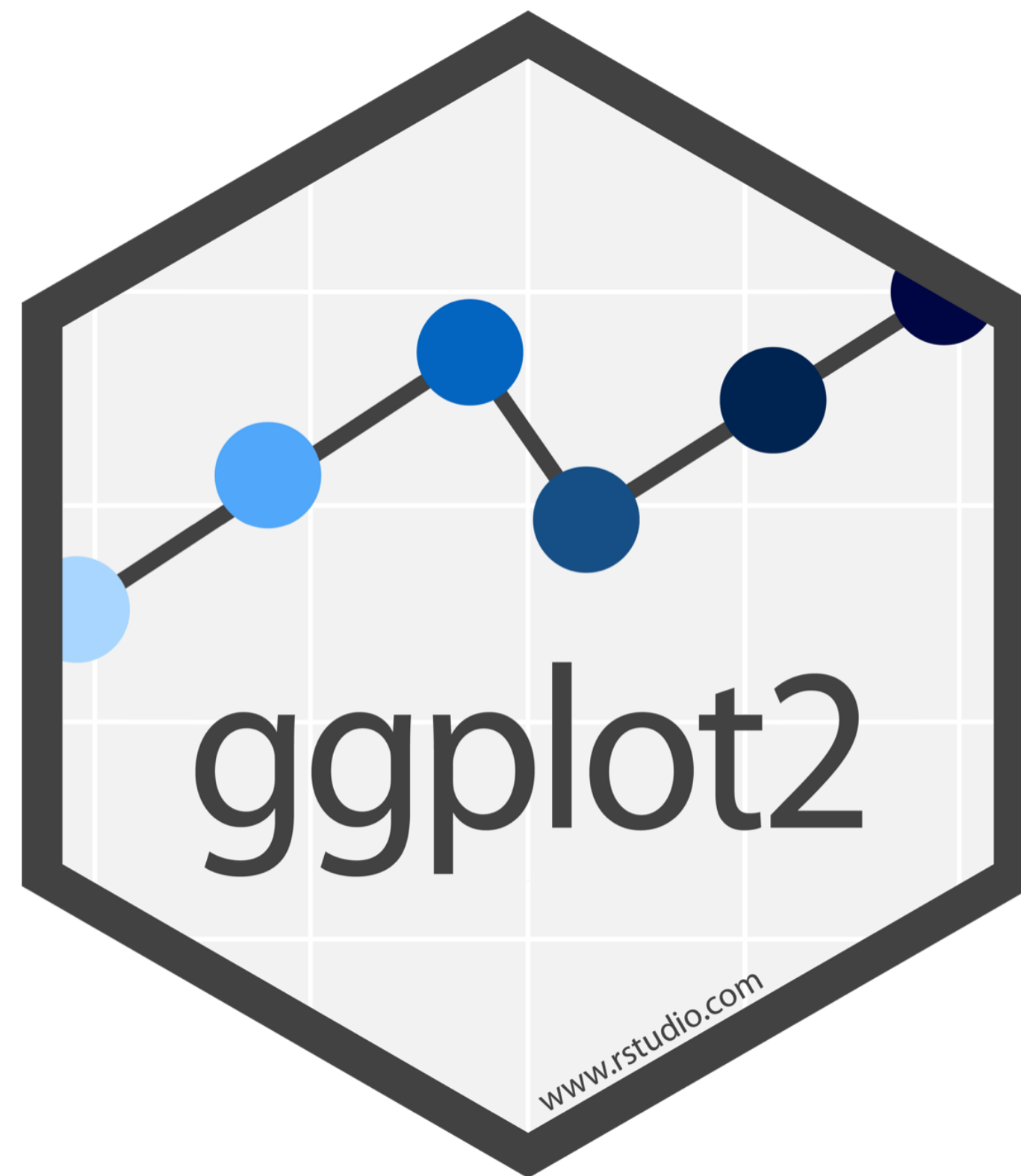
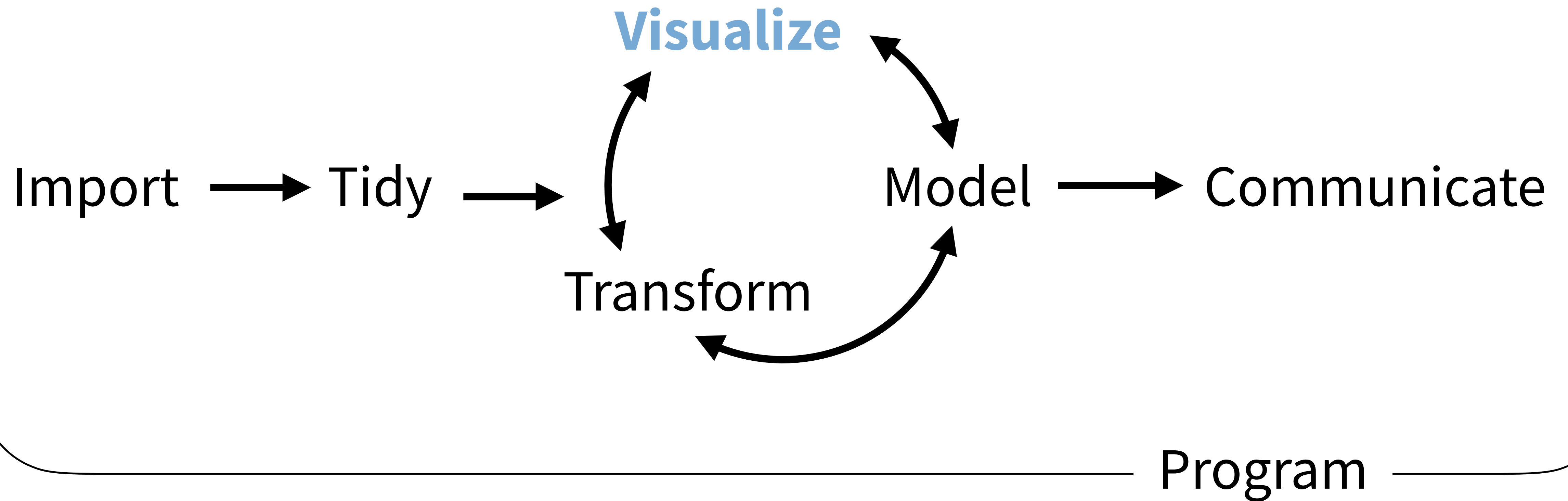


Visualize Data with



(Applied) Data Science



"The simple graph has brought more information to the data analyst's mind than any other device."

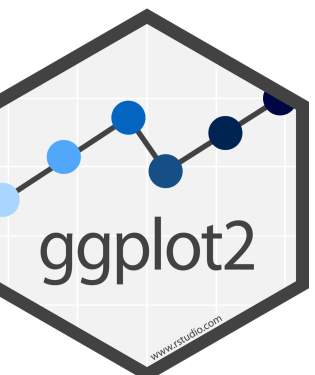
- John Tukey

mpg

Fuel economy data for 38 models of car.

mpg

manufacturer <chr>	displ <dbl>	year <int>	cyl <int>	trans <chr>	drv <chr>	cty <int>	hwy <int>	fl <chr>	class <chr>
audi	1.8	1999	4	auto(l5)	f	18	29	p	compact
audi	1.8	1999	4	manual(m5)	f	21	29	p	compact
audi	2.0	2008	4	manual(m6)	f	20	31	p	compact
audi	2.0	2008	4	auto(av)	f	21	30	p	compact
audi	2.8	1999	6	auto(l5)	f	16	26	p	compact
audi	2.8	1999	6	manual(m5)	f	18	26	p	compact
audi	3.1	2008	6	auto(av)	f	18	27	p	compact



Quiz

What relationship do you expect to see between engine size (displ) and mileage (hwy)?

No peeking ahead!

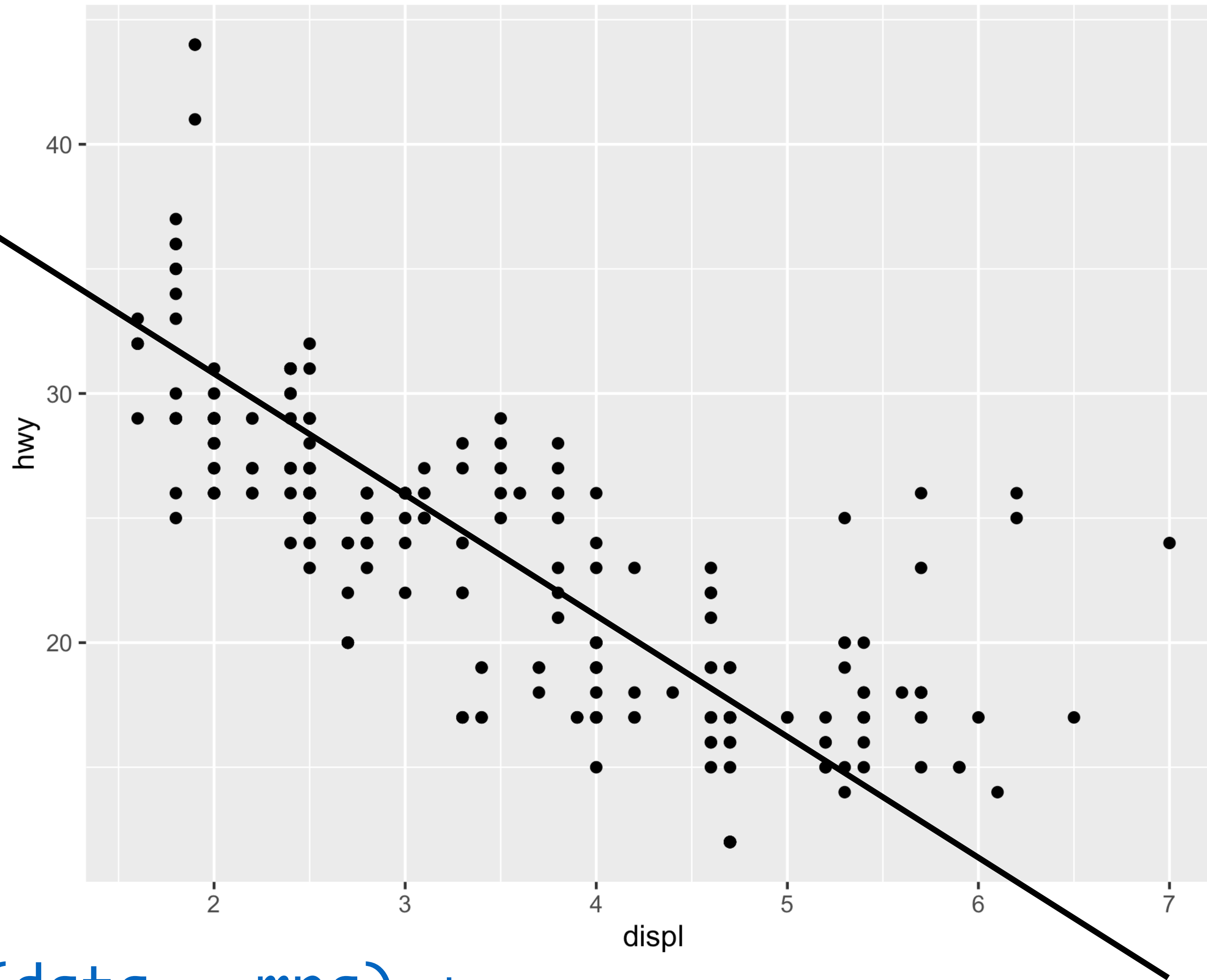
01:00

Your Turn 1

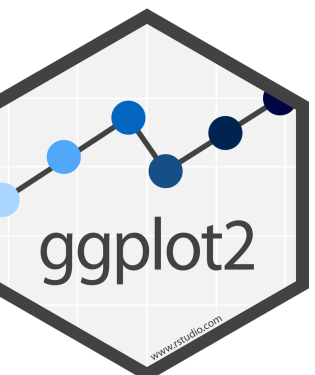
Run this code in **02-Visualize-Exercises.qmd** to make a graph. Pay strict attention to spelling, capitalization, and parentheses!

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





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



1. "Initialize" a plot with `ggplot()`
2. Add layers with `geom_` functions

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

Pro tip: Always put the + at the end of a line, Never at the start

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

data

+ before new line

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

type of layer

aes()

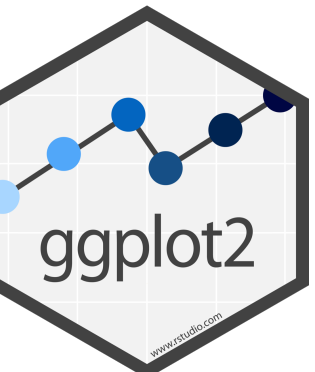
x variable

y variable

A template

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

```
geom_point(mapping = aes(x = displ, y = hwy))
```

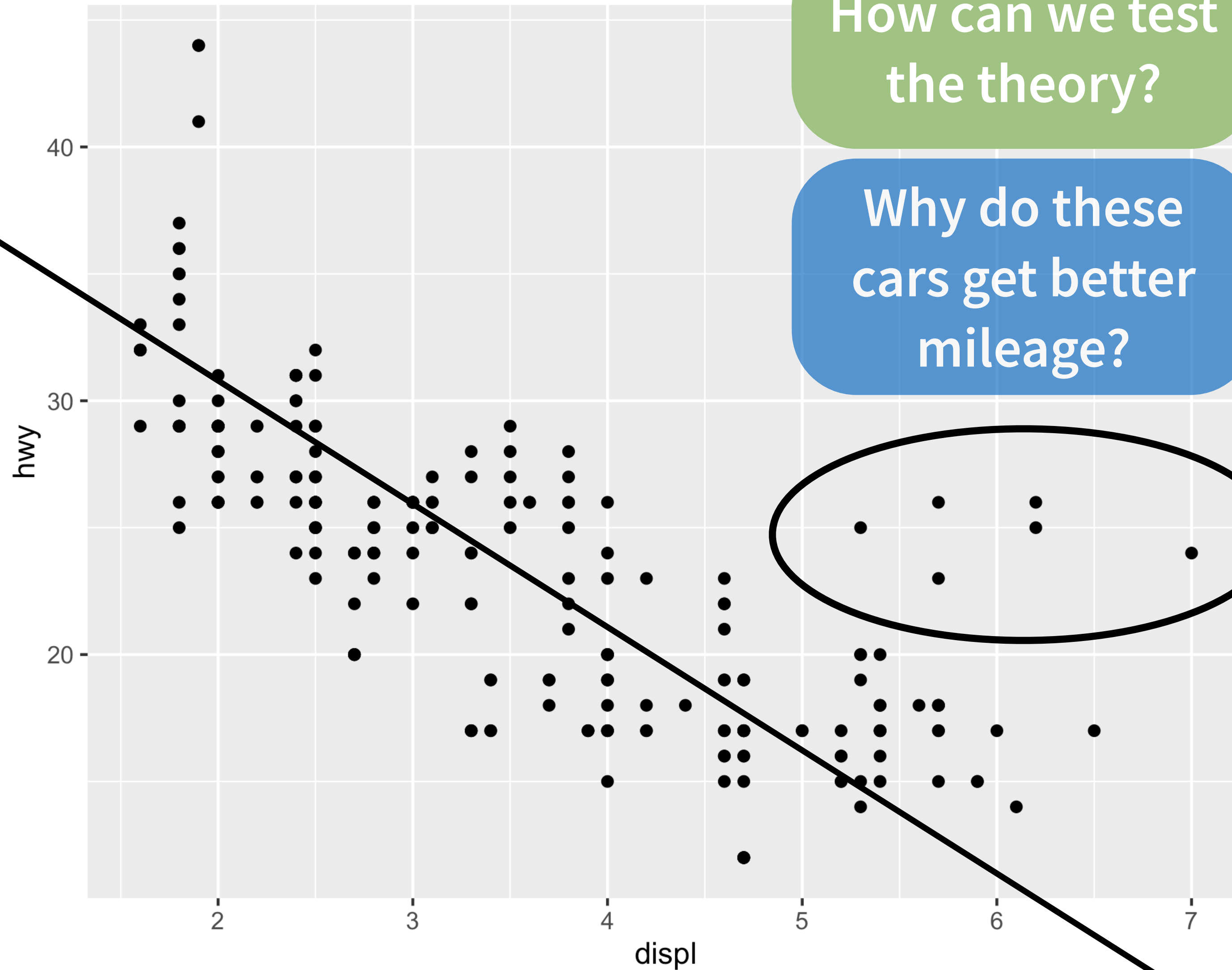


Mappings



"The greatest value of a picture is when it forces us to notice what we never expected to see."

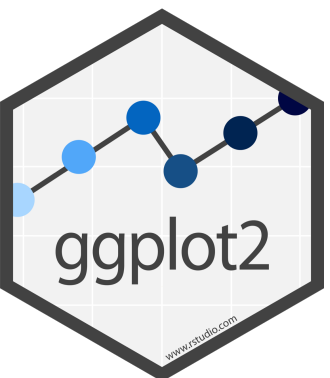
- John Tukey



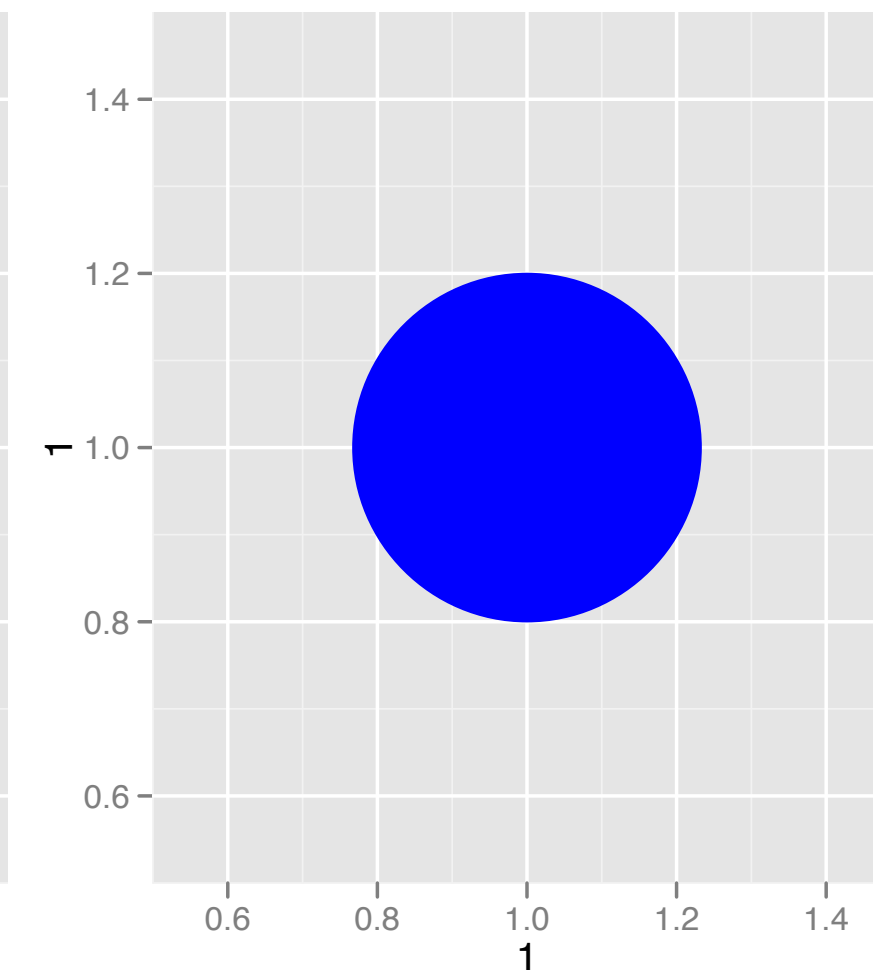
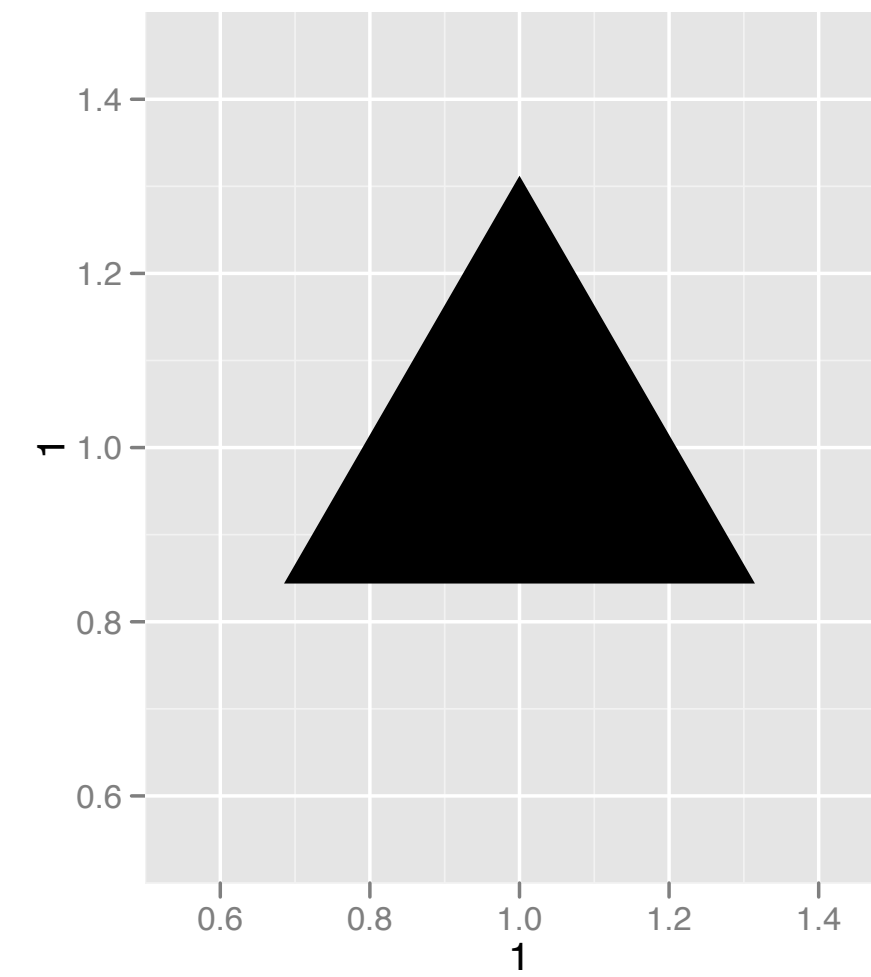
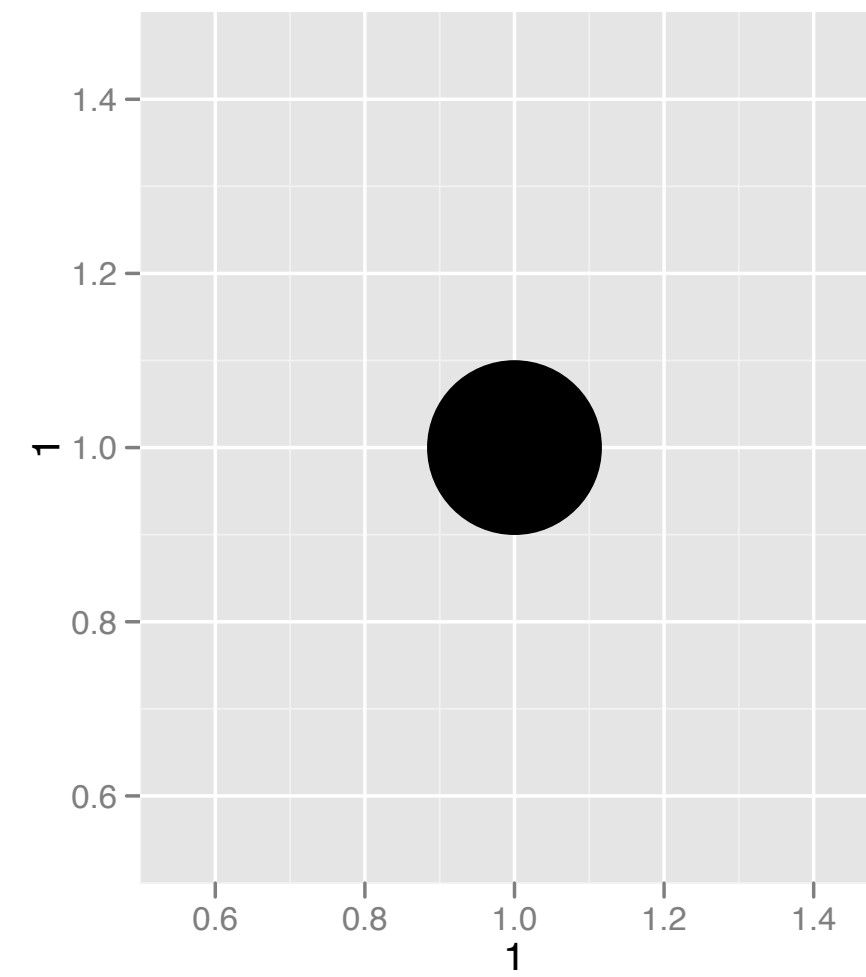
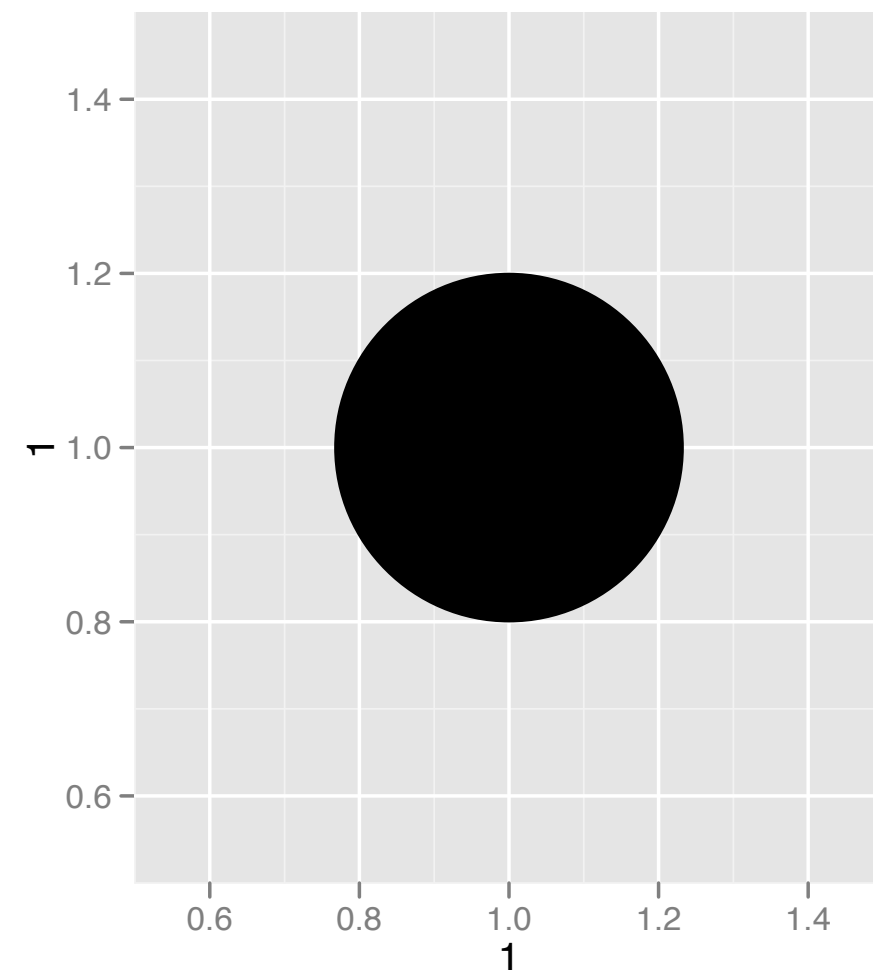
How can we test the theory?

Why do these cars get better mileage?

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



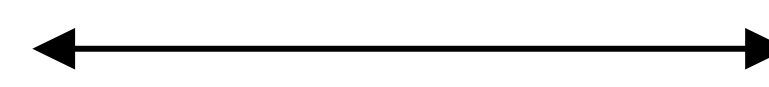
Aesthetics



Visual Space

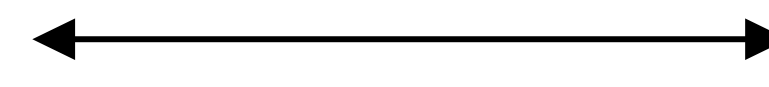
Data Space

color



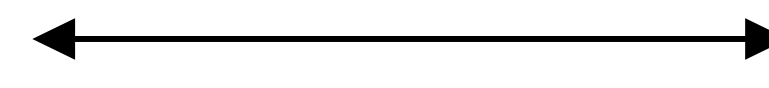
class

Red



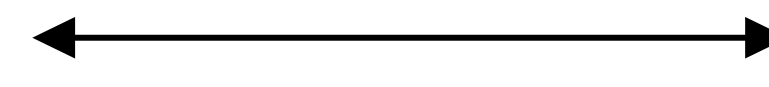
2seater

Brown



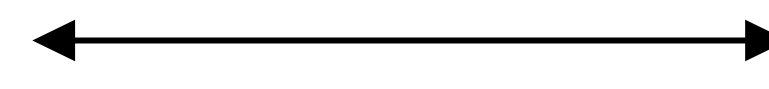
compact

Green



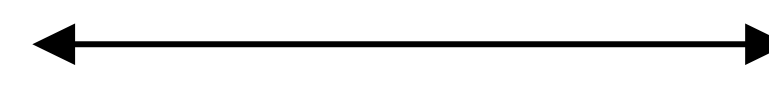
midsize

Aqua



minivan

Blue



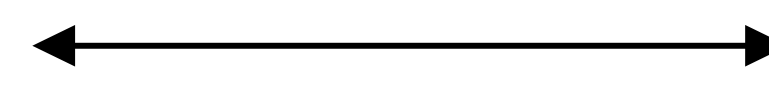
pickup

Violet



subcompact

Pink



SUV

Aesthetics

aesthetic
property

Variable to
map it to

```
ggplot(mpg) + geom_point(aes(x = displ, y = hwy, color = class))  
ggplot(mpg) + geom_point(aes(x = displ, y = hwy, size = class))  
ggplot(mpg) + geom_point(aes(x = displ, y = hwy, shape = class))  
ggplot(mpg) + geom_point(aes(x = displ, y = hwy, alpha = class))
```


Your Turn 2

In the next chunk, add color, size, alpha, and shape aesthetics to your graph. Experiment.

Do different things happen when you map aesthetics to discrete and continuous variables?

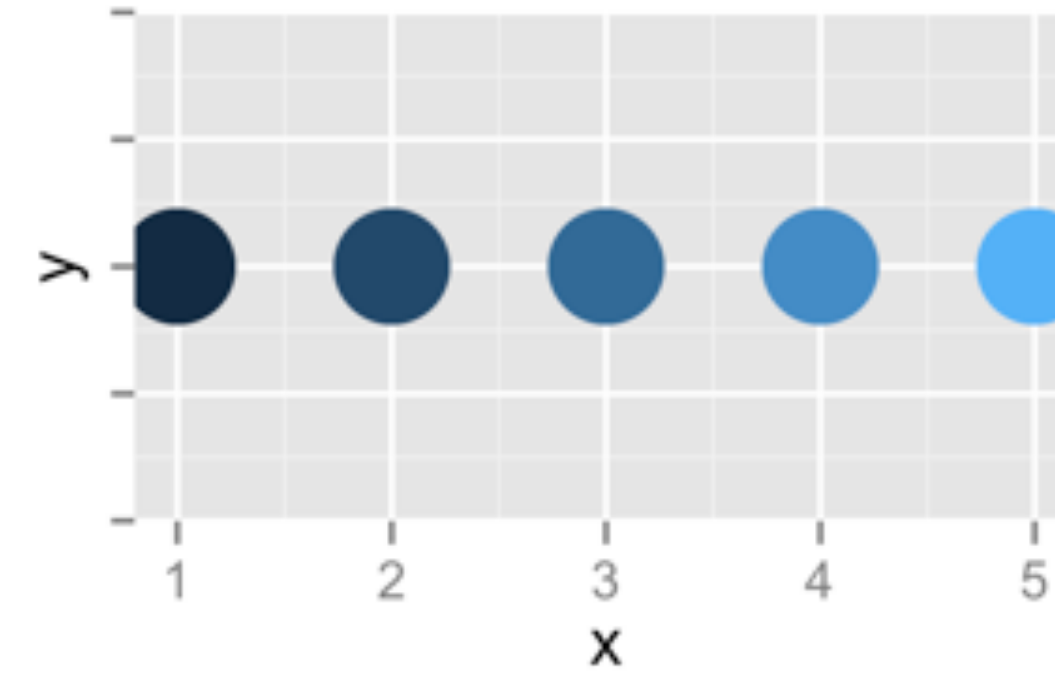
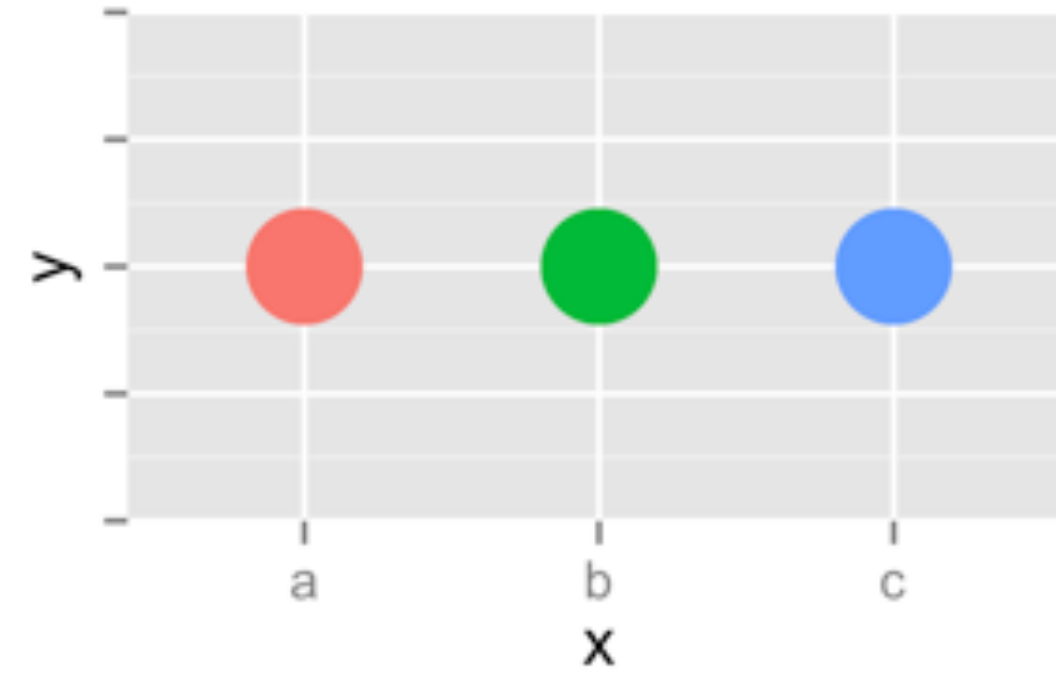
What happens when you use more than one aesthetic?

05:00

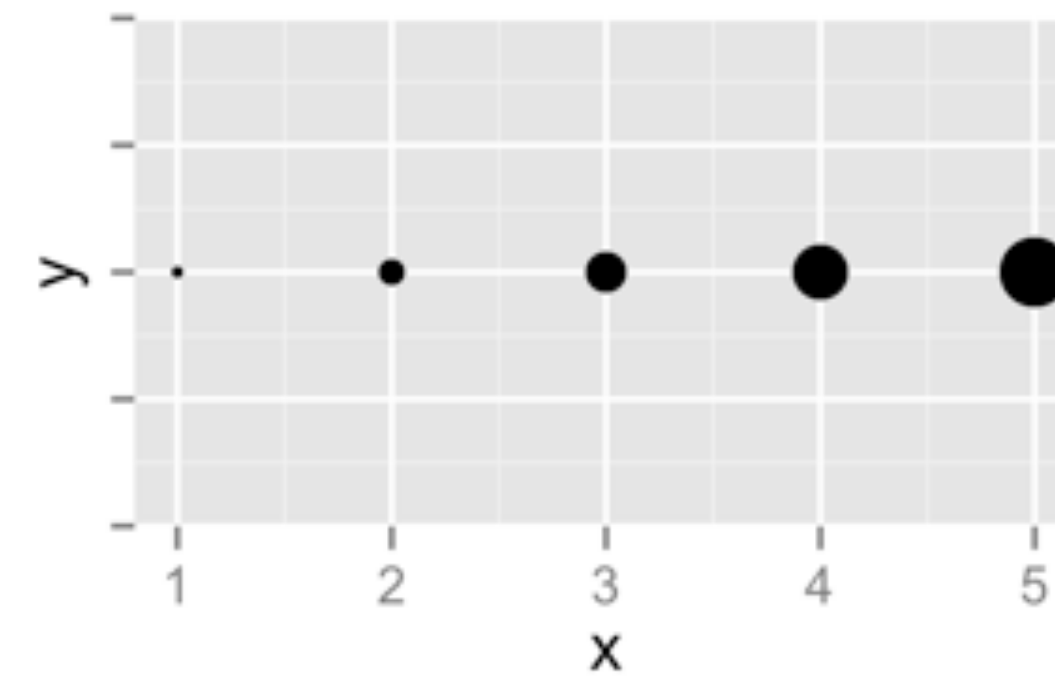
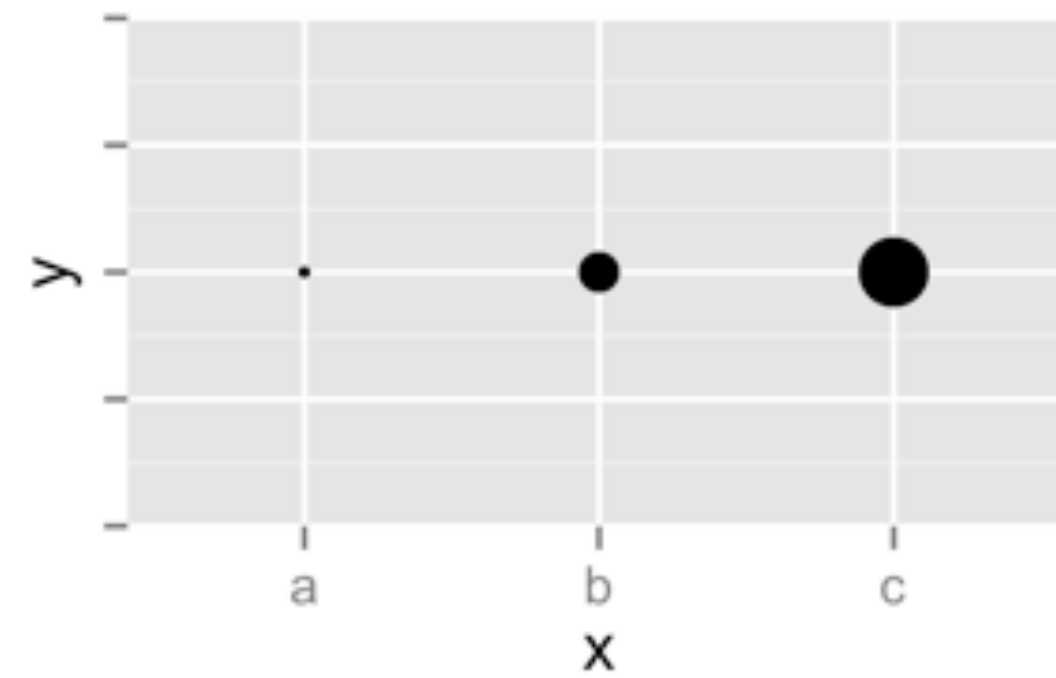
Discrete

Continuous

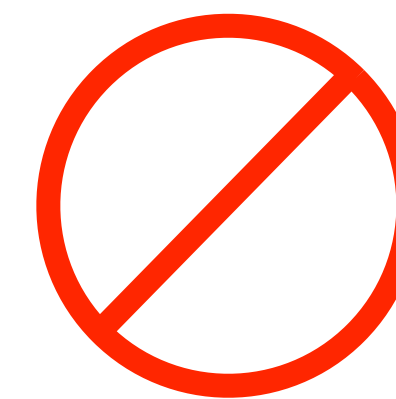
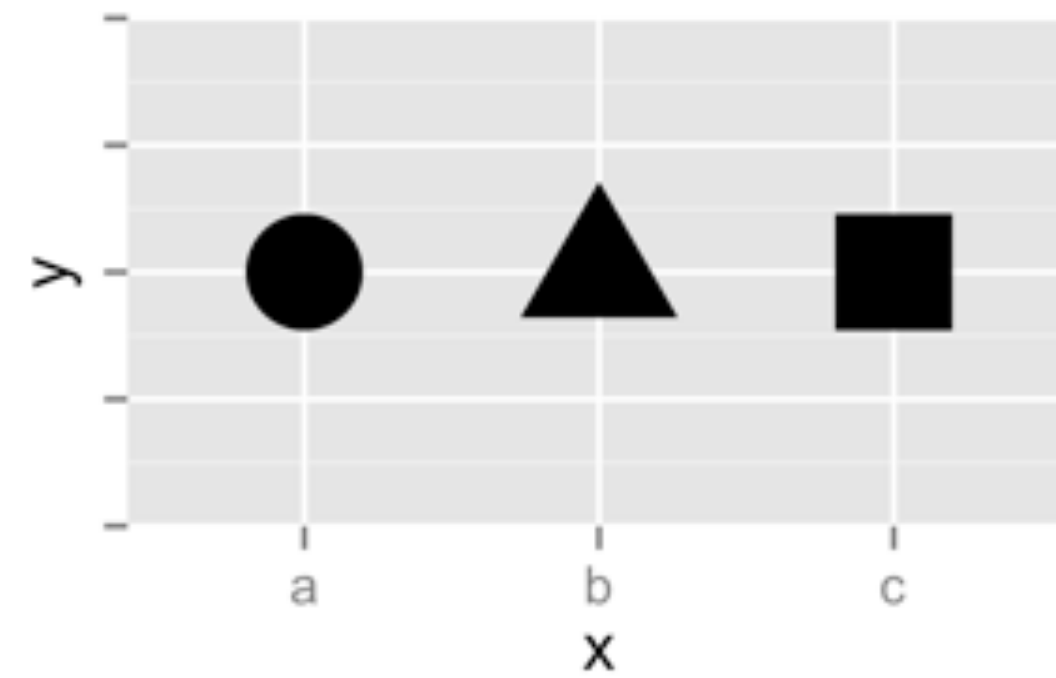
Color

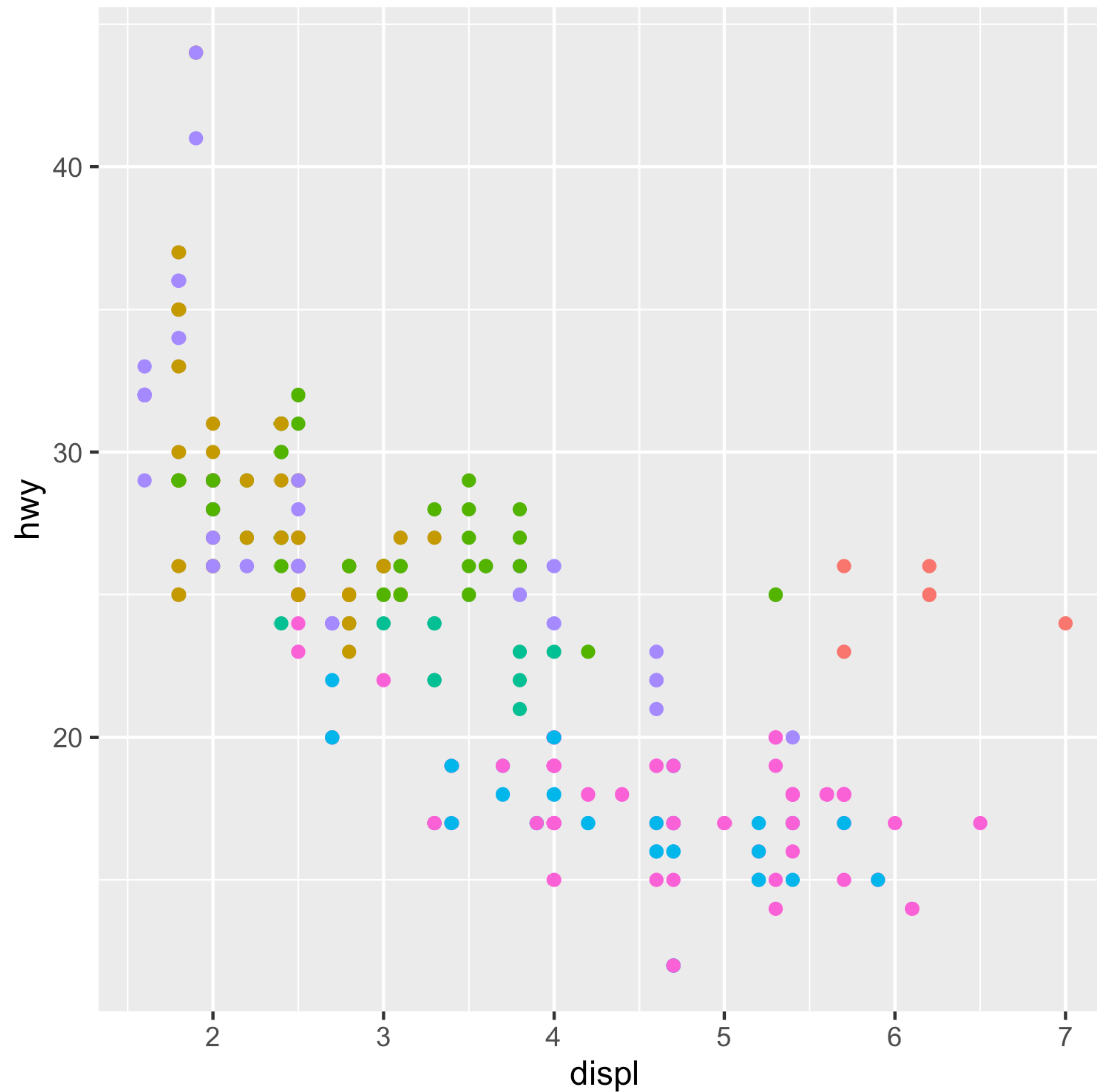


Size



Shape

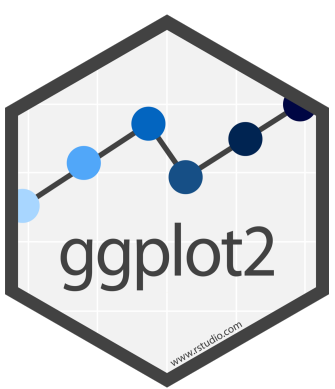




Legend added automatically

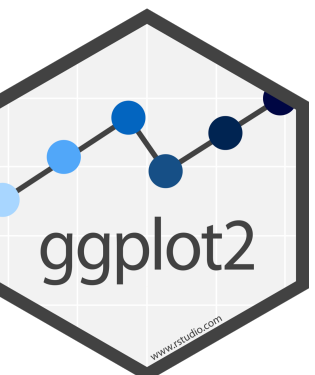
- class
- 2seater
 - compact
 - midsize
 - minivan
 - pickup
 - subcompact
 - suv

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



ERROR!

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

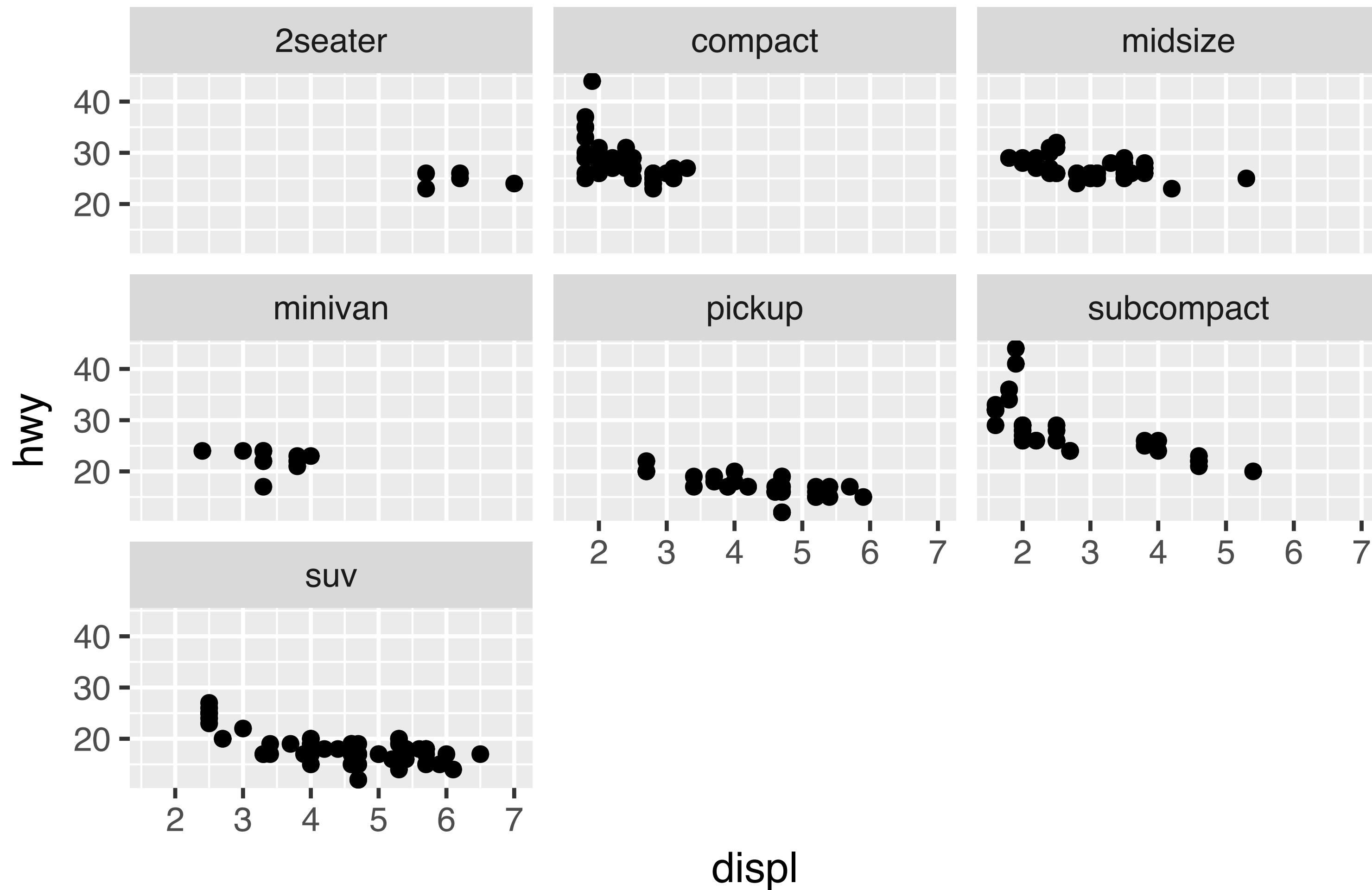


Facets



Facets

Subplots that display subsets of the data.



Help me

What do `facet_grid` and `facet_wrap` do?

```
q <- ggplot(mpg) + geom_point(aes(x = displ, y = hwy))
```

```
q + facet_grid(cols = vars(cyl))
```

```
q + facet_grid(rows = vars(drv))
```

```
q + facet_grid(rows = vars(drv), cols = vars(cyl))
```

```
q + facet_wrap(facets = vars(class))
```


summary

facet_grid() - 2D grid, one variable in rows, one variable in columns
facet_wrap() - 1D ribbon wrapped into 2D

A ggplot2 template

Make any plot by filling in the parameters of this template

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>)) +  
  <FACET_FUNCTION>
```


Your Turn 3

Add the black code to your graph. What does it do?

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(displ, hwy, color = class)) +  
  labs(title = "Fuel Efficiency by Engine Size",  
        subtitle = "Data facетted by class",  
        x = "Engine Size (displacement in liters)",  
        y = "Fuel Efficiency (MPG)",  
        color = "Class of\nAutomobile",  
        caption = "Data from the EPA")
```

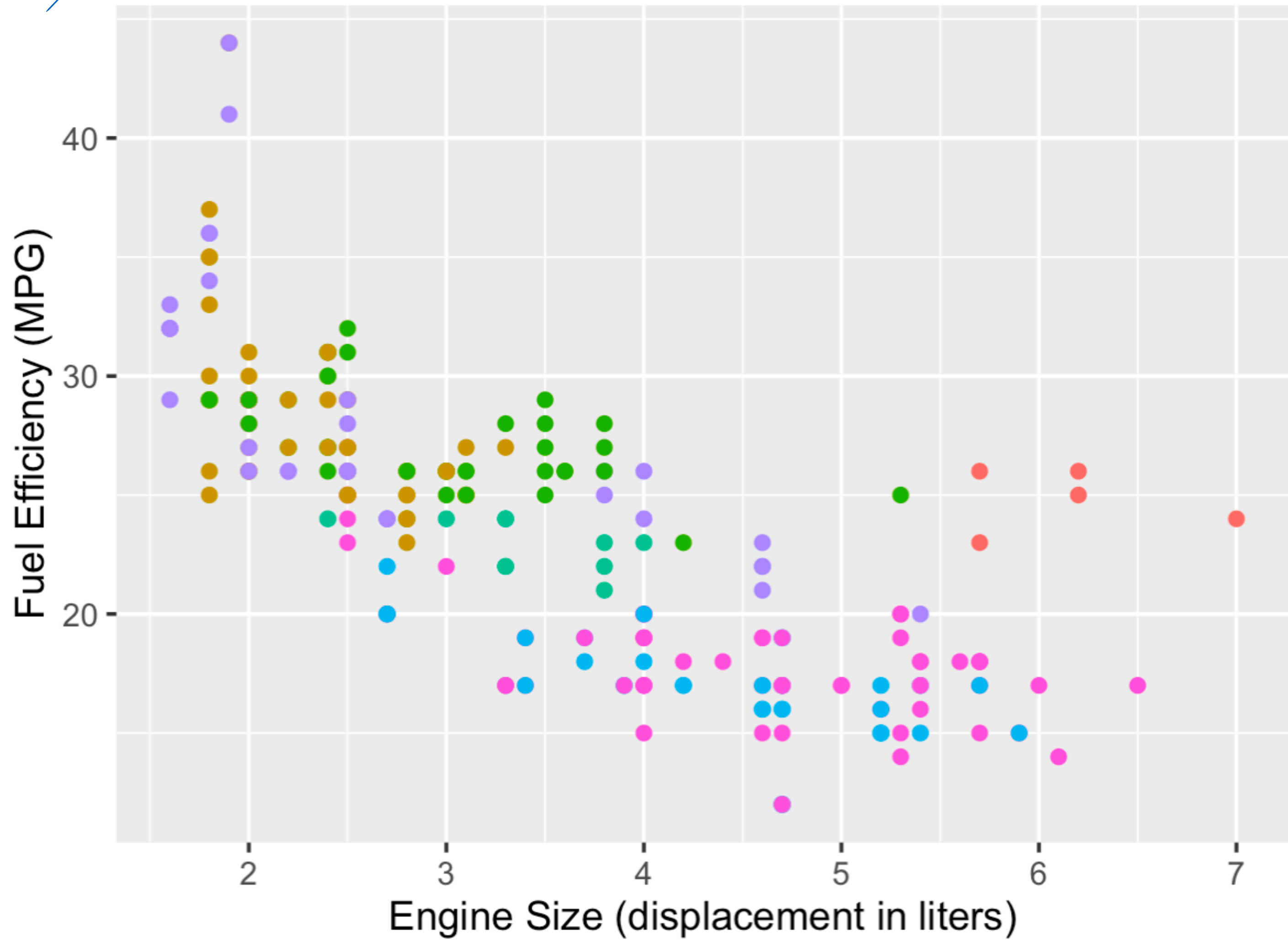
02:00

TITLE
SUBTITLE

Fuel Efficiency by Engine Size
Data faceted by class

COLOR

Y



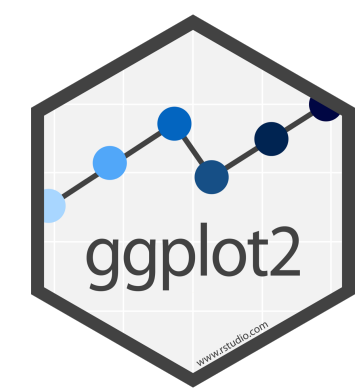
Class of Automobile

- 2seater
- compact
- midsize
- minivan
- pickup
- subcompact
- suv

CAPTION

Data from the EPA

X

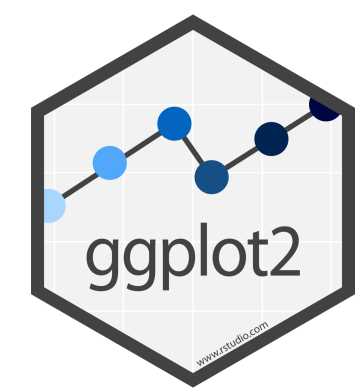
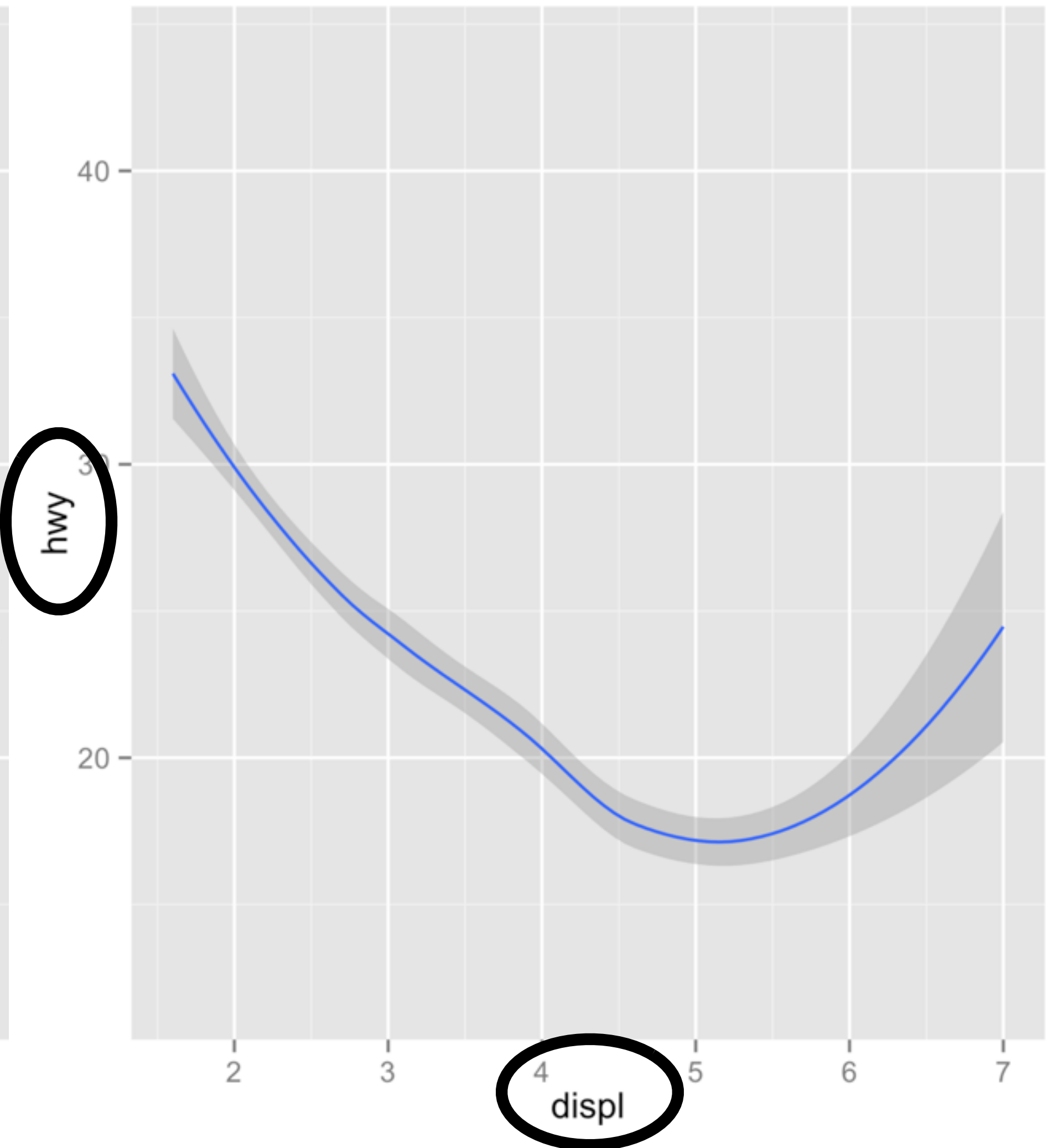
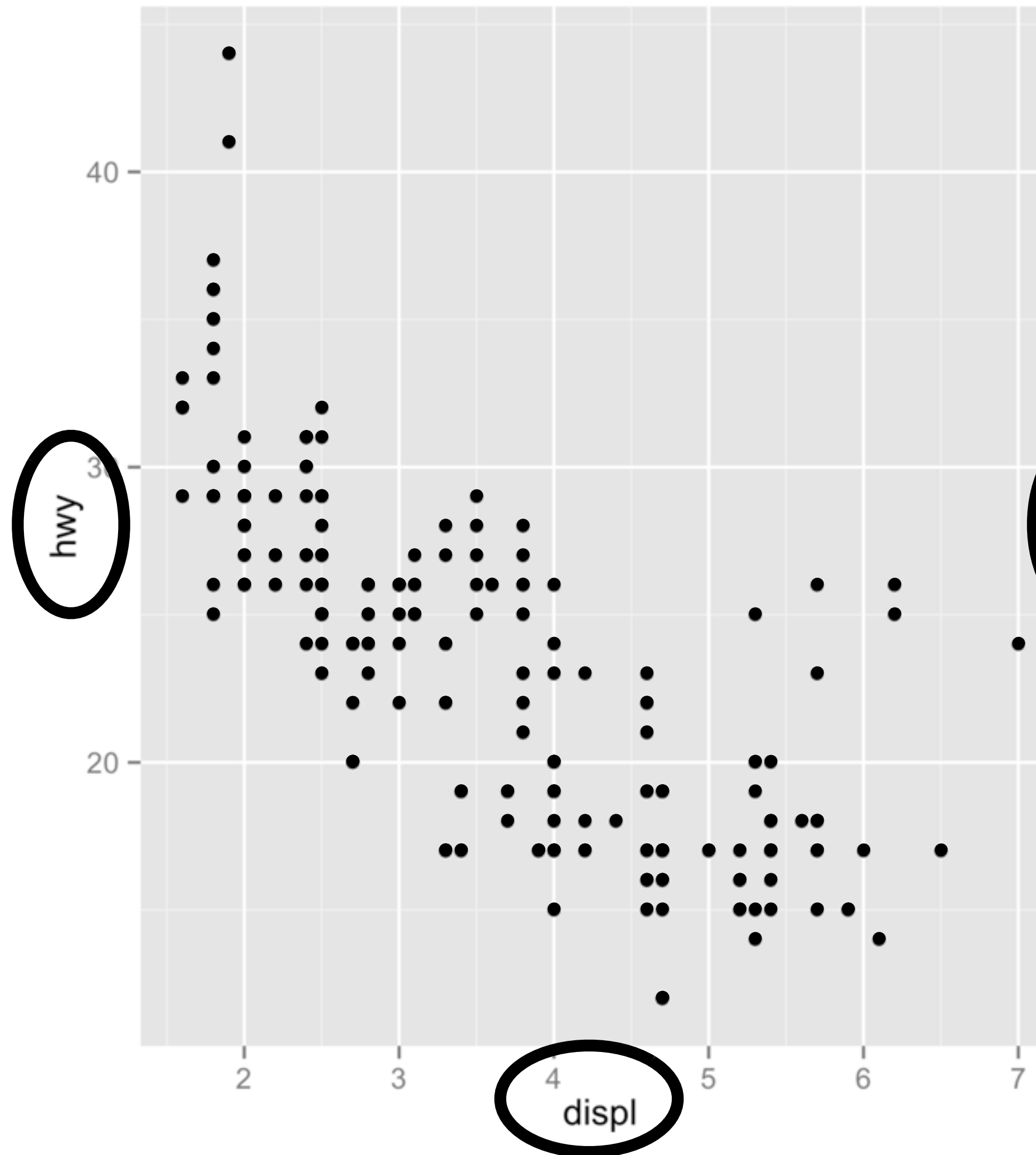


Geoms



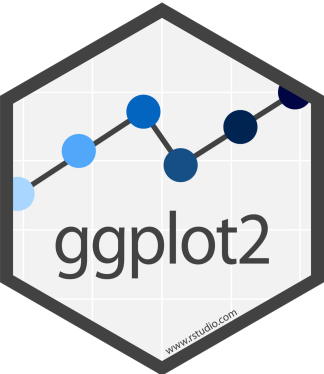
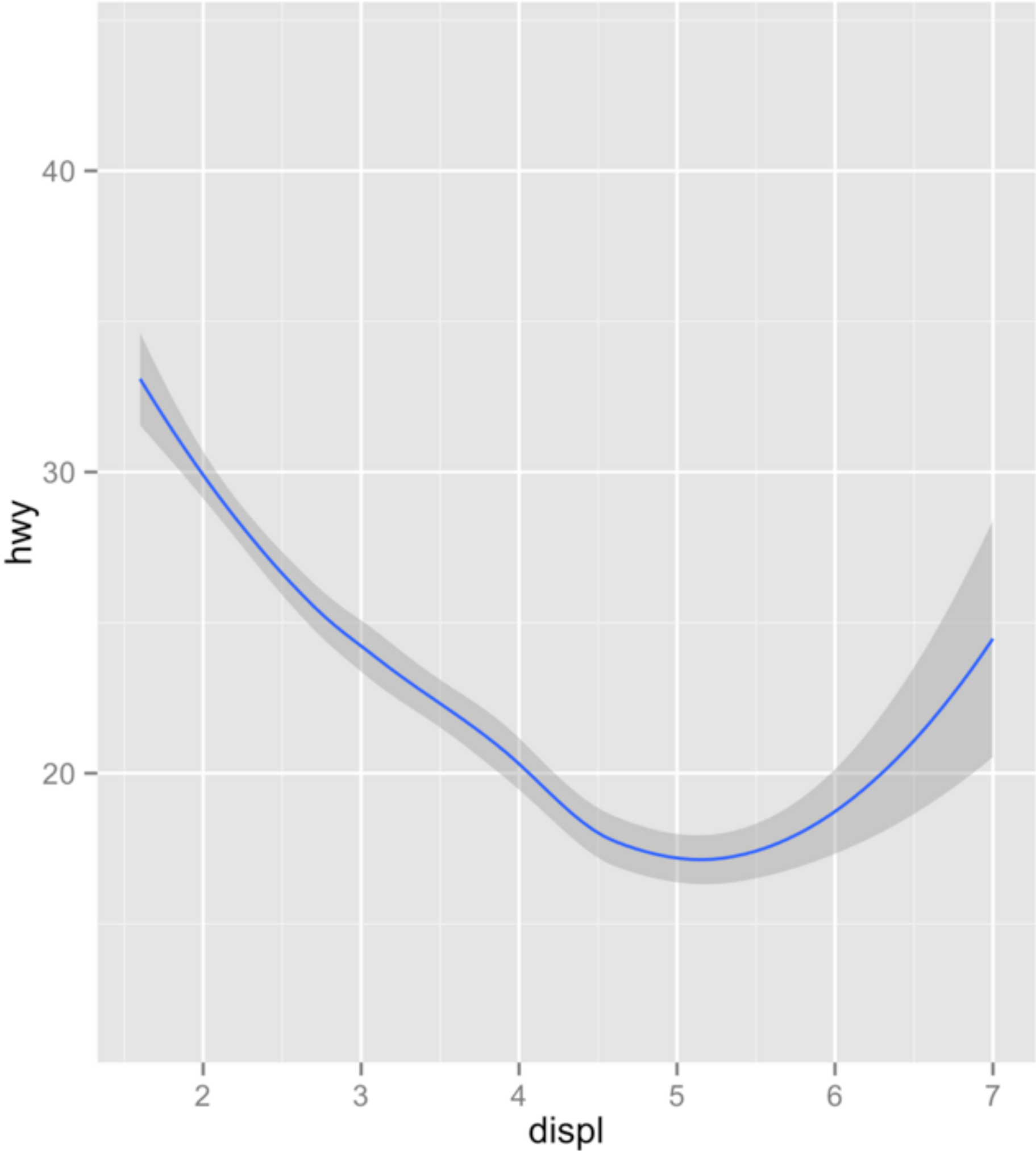
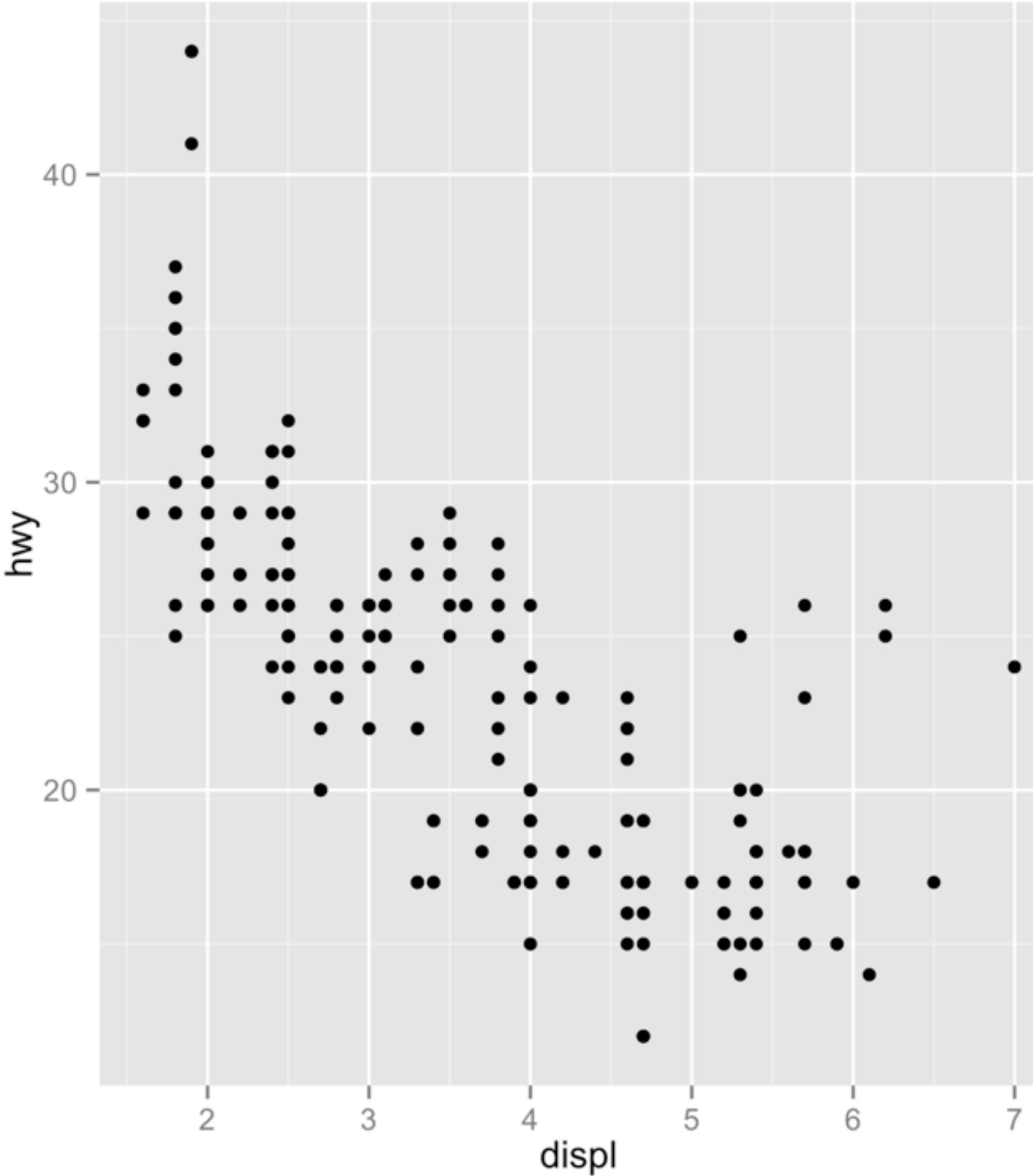
How are these plots similar?

Same: x var , y var , data



How are these plots different?

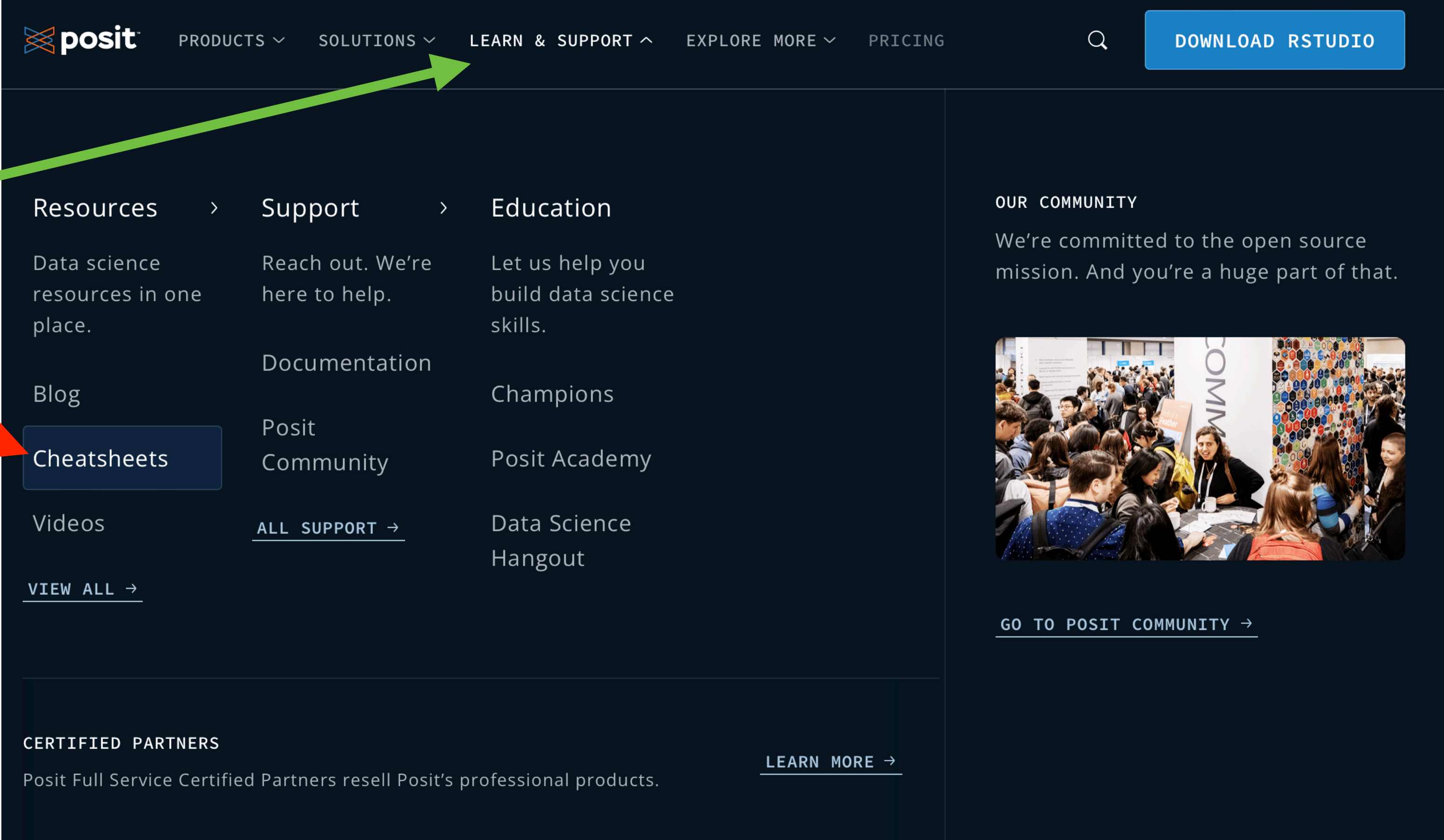
Different: geometric object (geom), e.g. the visual object used to represent the data



geoms

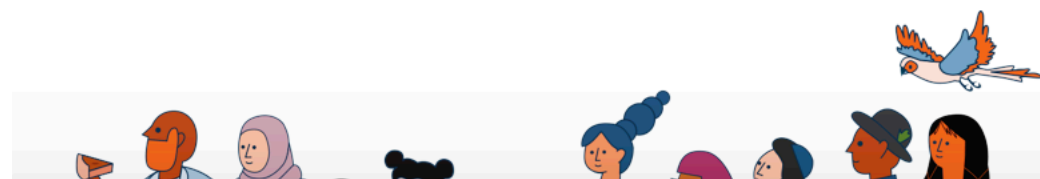
```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```


<https://posit.co/resources/cheatsheets/>

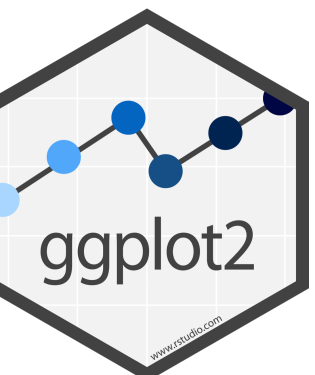


The screenshot shows the Posit website's navigation menu. The 'LEARN & SUPPORT' tab is highlighted with a green arrow pointing to it from the left. Below this tab, a sub-menu is displayed with three columns: 'Resources', 'Support', and 'Education'. The 'Resources' column includes 'Data science resources in one place.', 'Blog', 'Cheatsheets' (highlighted with a red arrow), and 'Videos'. The 'Support' column includes 'Reach out. We're here to help.', 'Documentation', 'Posit Community', and 'ALL SUPPORT →'. The 'Education' column includes 'Let us help you build data science skills.', 'Champions', 'Posit Academy', and 'Data Science Hangout'. A 'VIEW ALL →' link is at the bottom of the Resources column. On the right side of the page, there is a 'OUR COMMUNITY' section with a photo of a group of people and a 'GO TO POSIT COMMUNITY →' link. At the bottom of the page, there is a 'CERTIFIED PARTNERS' section with a 'LEARN MORE →' link.

**CLICK
CHEATSHEETS
IN THE
LEARN & SUPPORT
TAB**

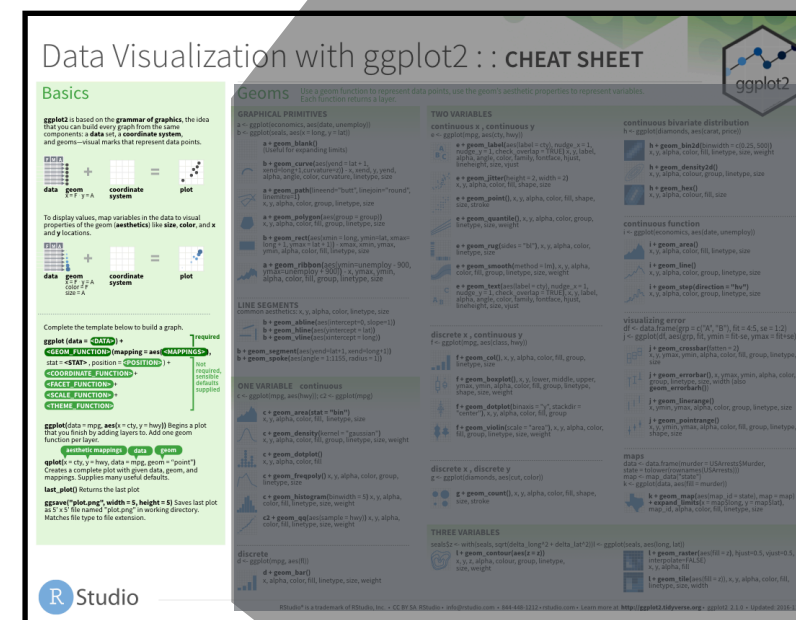


posit::conf(2024)



geom_ functions

Each requires a mapping argument.



ggplot2

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, unemployment))
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, 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 = unemployment - 900, ymax = unemployment + 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

THREE VARIABLES

seals\$z <- with(seals, sqrt(delta_long^2 + delta_lat^2)) | <- 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

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, unemployment))

- 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

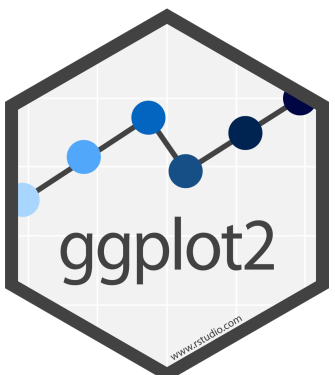
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))

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

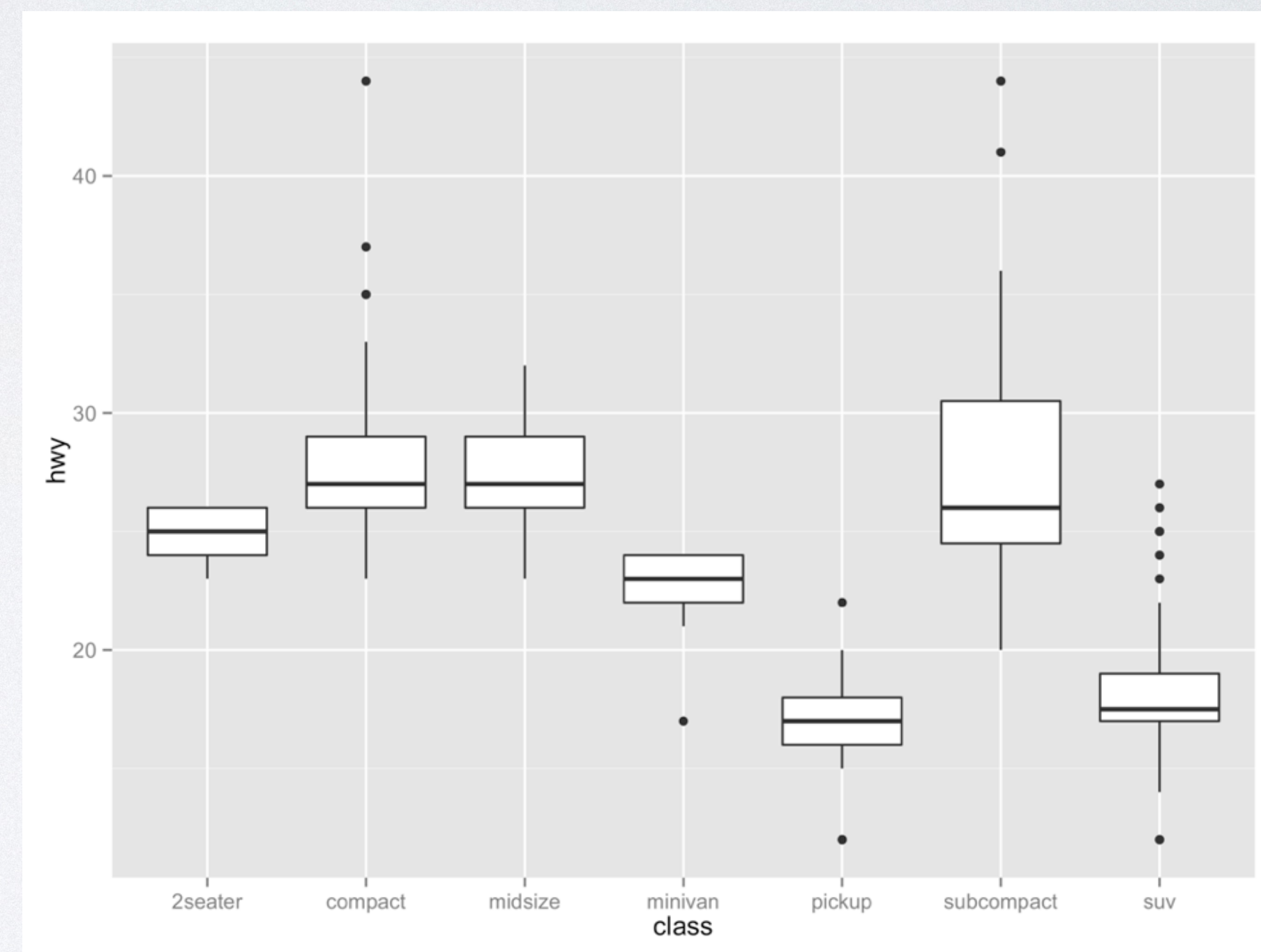
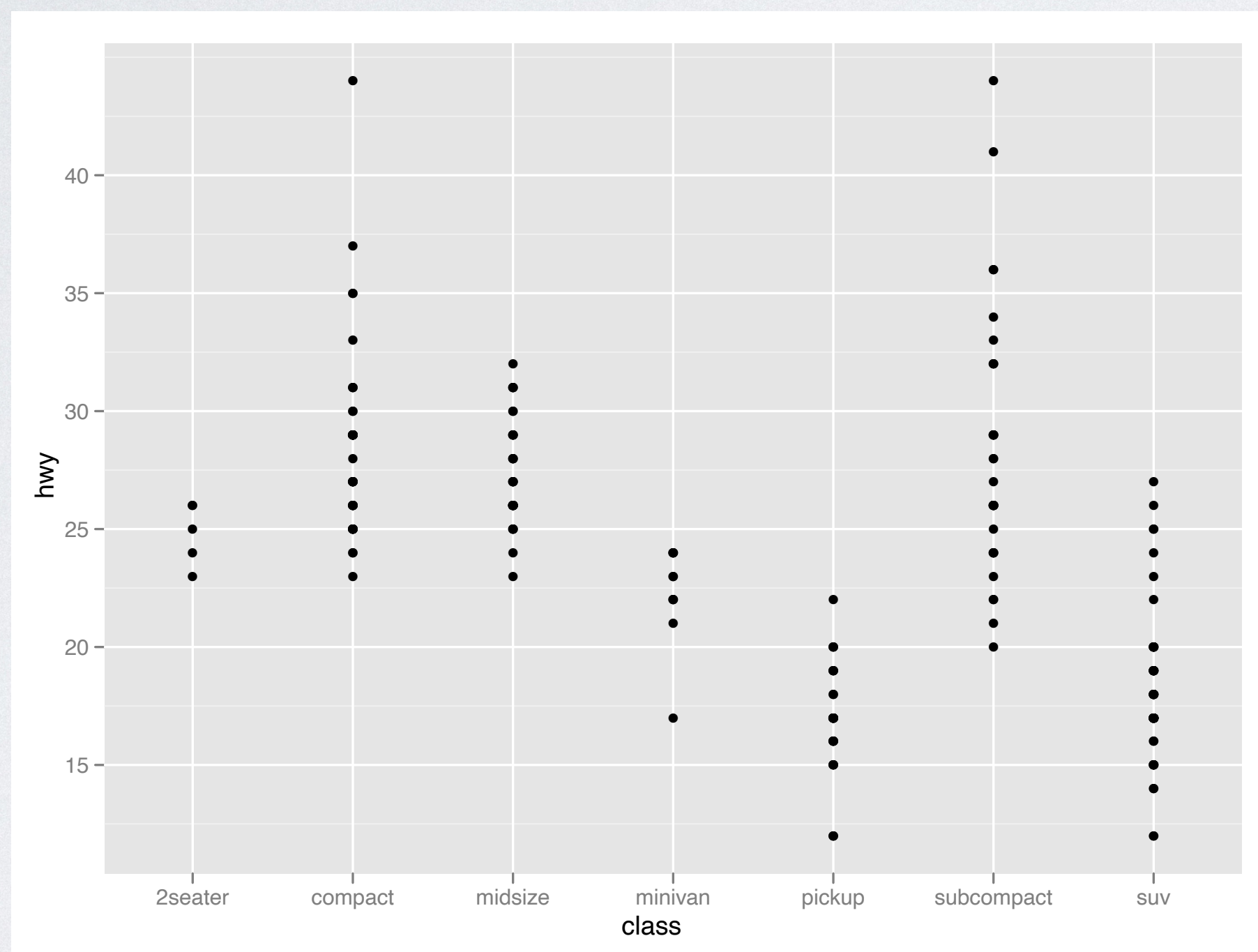
data <- data.frame(murder = USArrests\$Murder, state = tolower(rownames(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))

- k + geom_map(aes(map_id = state), map = map) + expand_limits(x = map\$long, y = map\$lat, map_id, alpha, color, fill, linetype, size)**



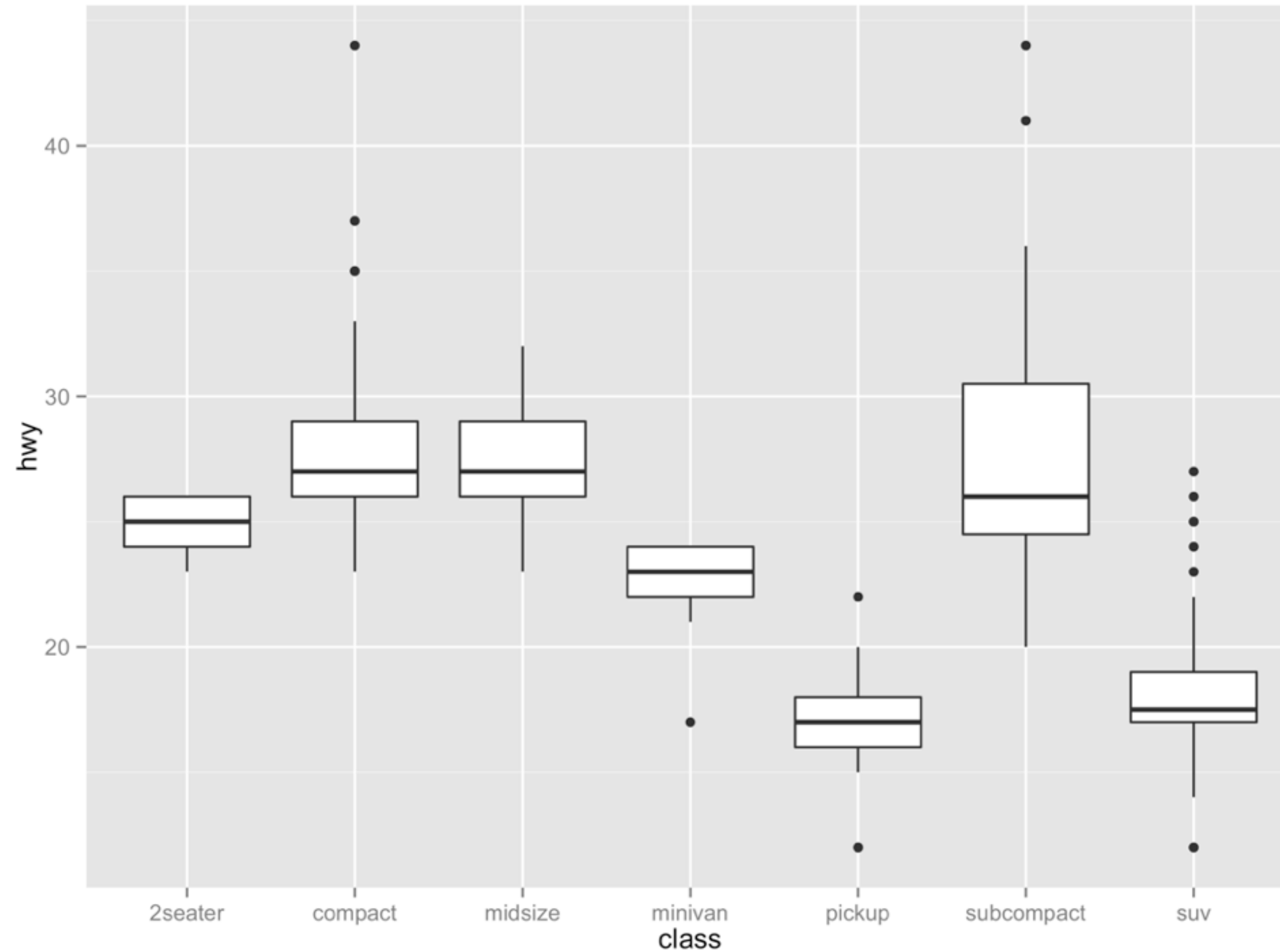
Your Turn 4

Decide how to replace this scatterplot with one that draws boxplots. Use the cheatsheet. Try your best guess.

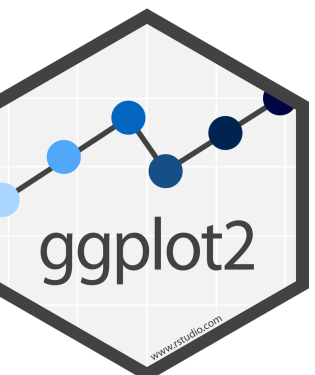


```
ggplot(mpg) + geom_point(aes(class, hwy))
```

02:00

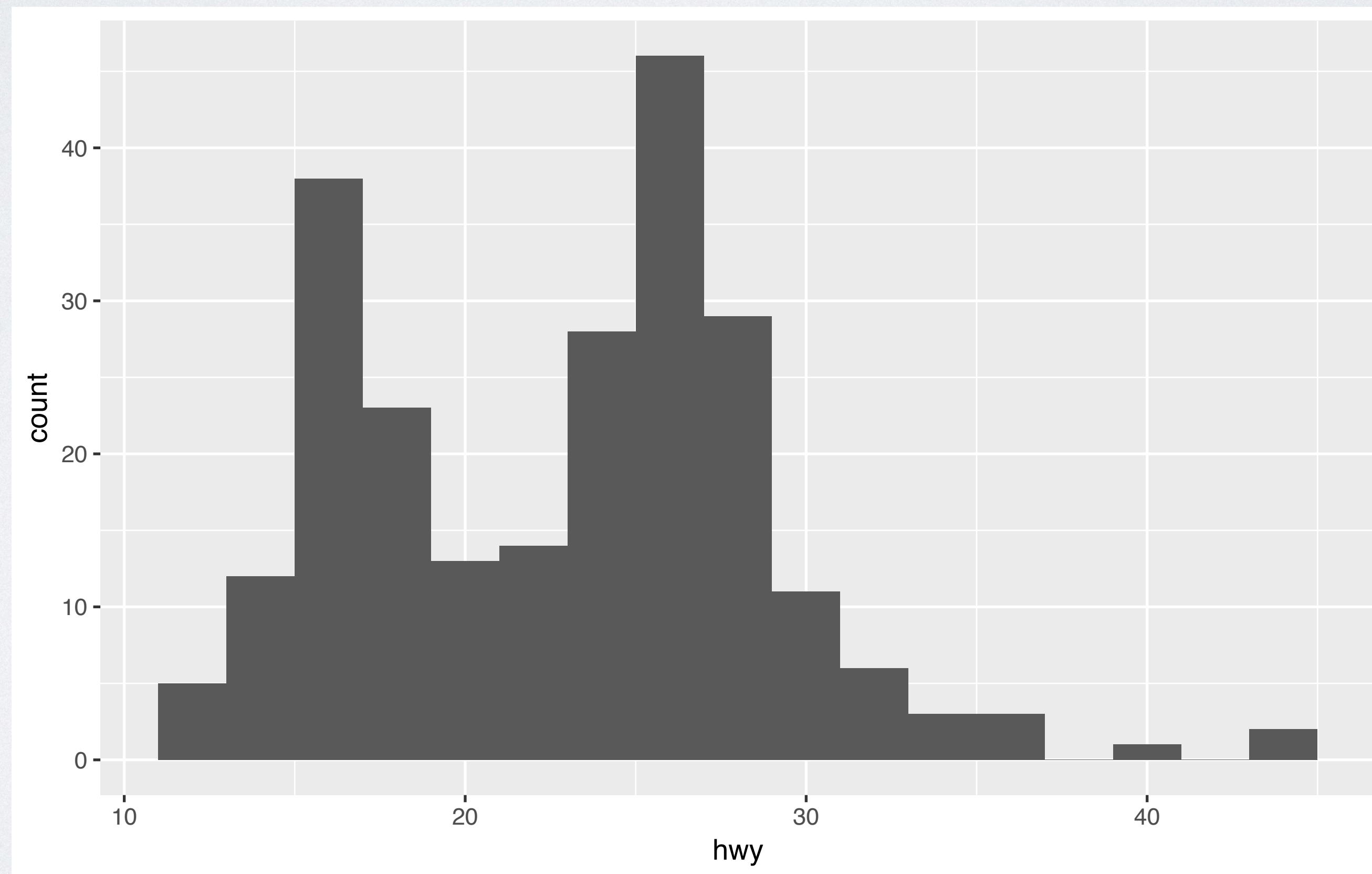


```
ggplot(data = mpg) +  
  geom_boxplot(mapping = aes(x = class, y = hwy))
```

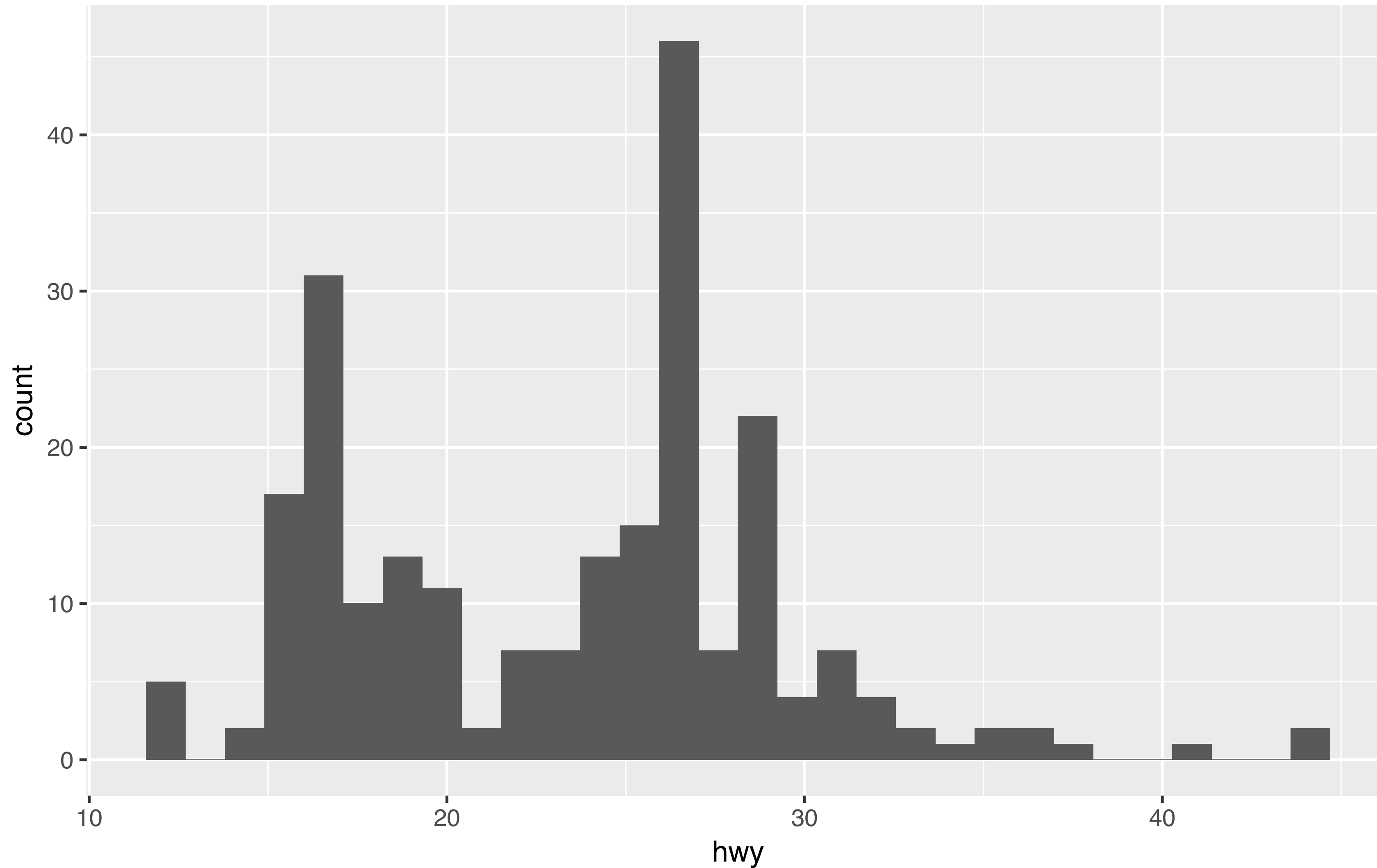


Your Turn 5

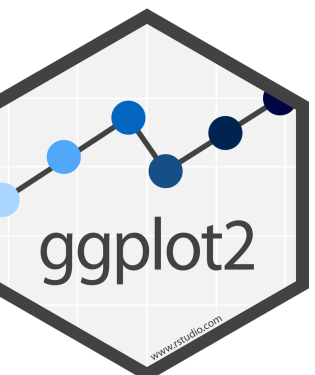
Make the histogram of **hwy** below. Use the cheatsheet. Hint: do not supply a **y** variable.



02:00

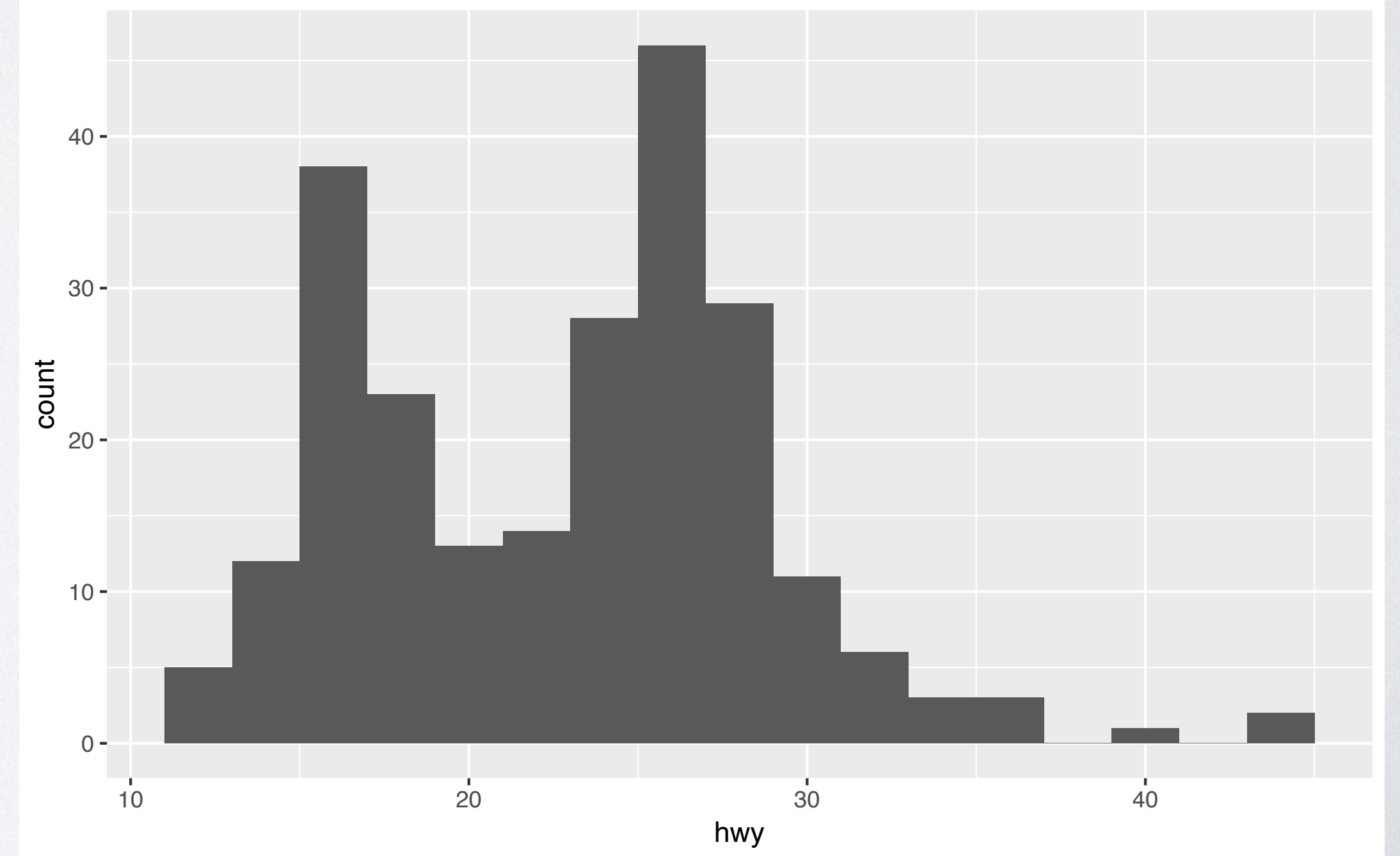
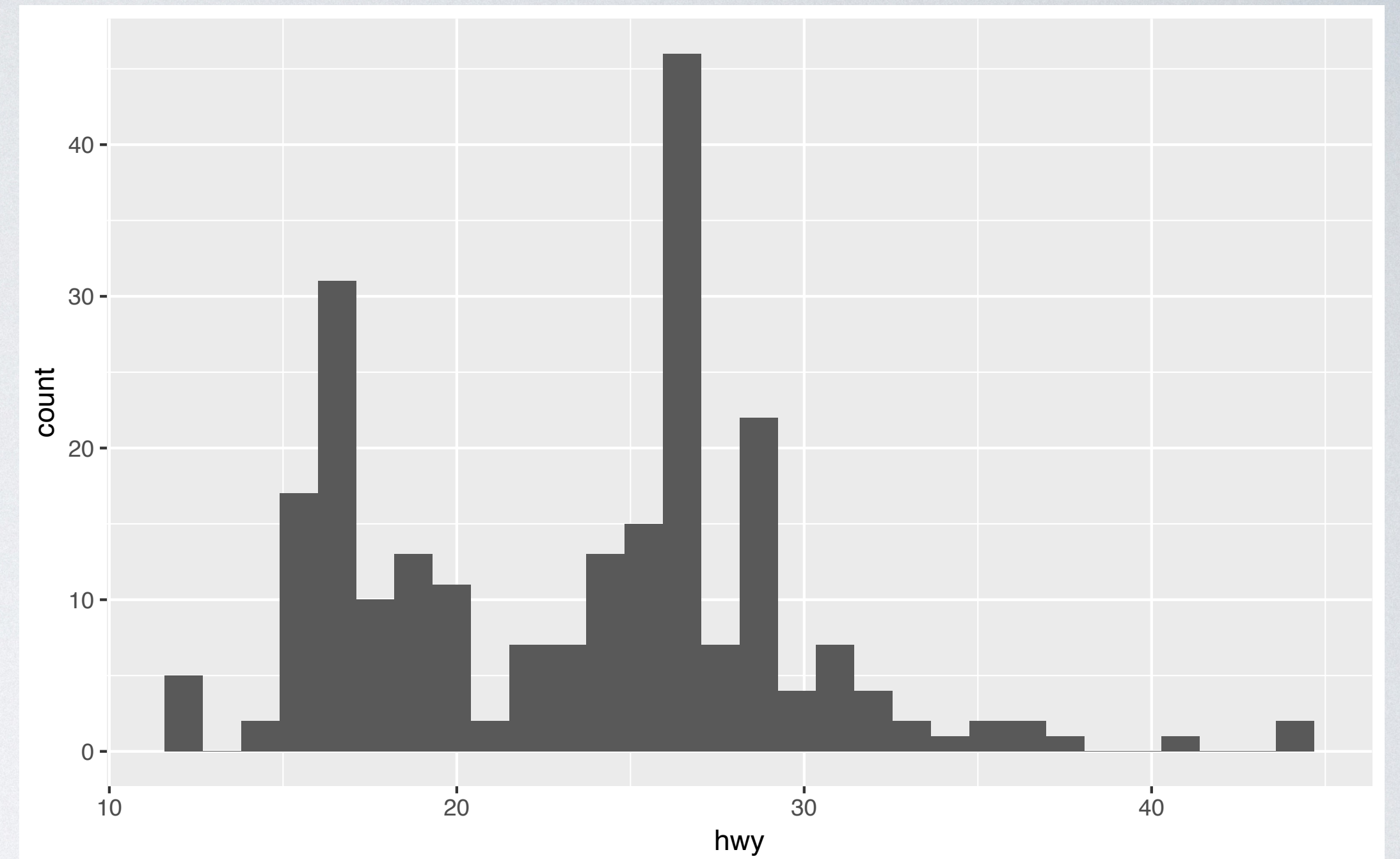


```
ggplot(data = mpg) +  
  geom_histogram(mapping = aes(x = hwy))
```



Quiz

What is the difference?



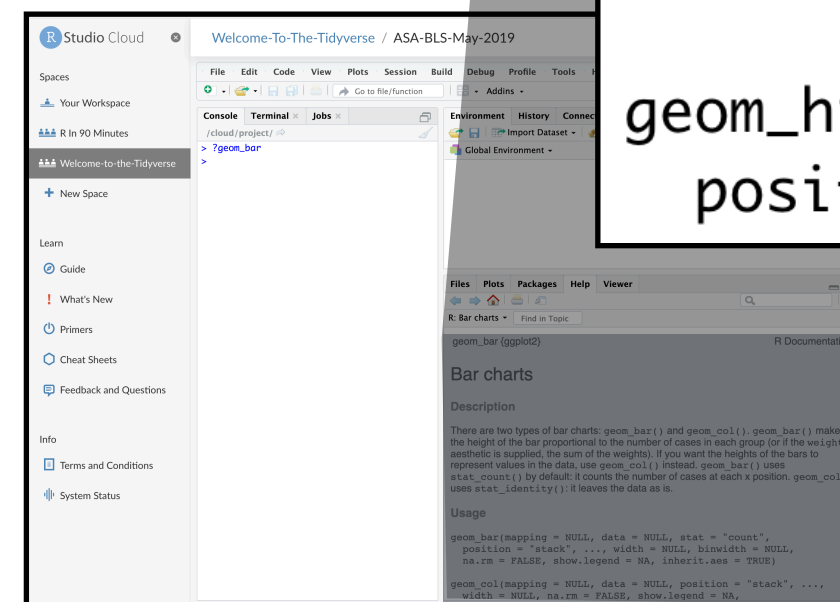
"Help" pages

To open the documentation for a function, type

?geom_histogram

?

function name (no parentheses)



geom_freqpoly {ggplot2}

R Documentation

Histograms and frequency polygons

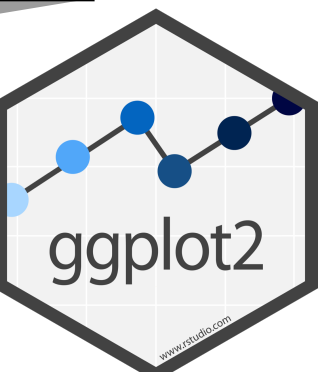
Description

Visualise the distribution of a single continuous variable by dividing the x axis into bins and counting the number of observations in each bin. Histograms (`geom_histogram()`) display the counts with bars; frequency polygons (`geom_freqpoly()`) display the counts with lines. Frequency polygons are more suitable when you want to compare the distribution across the levels of a categorical variable.

Usage

```
geom_freqpoly(mapping = NULL, data = NULL, stat = "bin",  
              position = "identity", ..., na.rm = FALSE, show.legend = NA,  
              inherit.aes = TRUE)
```

```
geom_histogram(mapping = NULL, data = NULL, stat = "bin",  
              position = "stack", ..., binwidth = NULL, bins = NULL,
```



Tips

- **scan** page for relevant info
- **ignore** things that don't make sense
- **try** out the examples

geom_freqpoly {ggplot2}

R Documentation

Histograms and frequency polygons

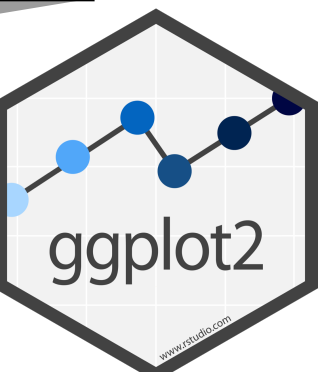
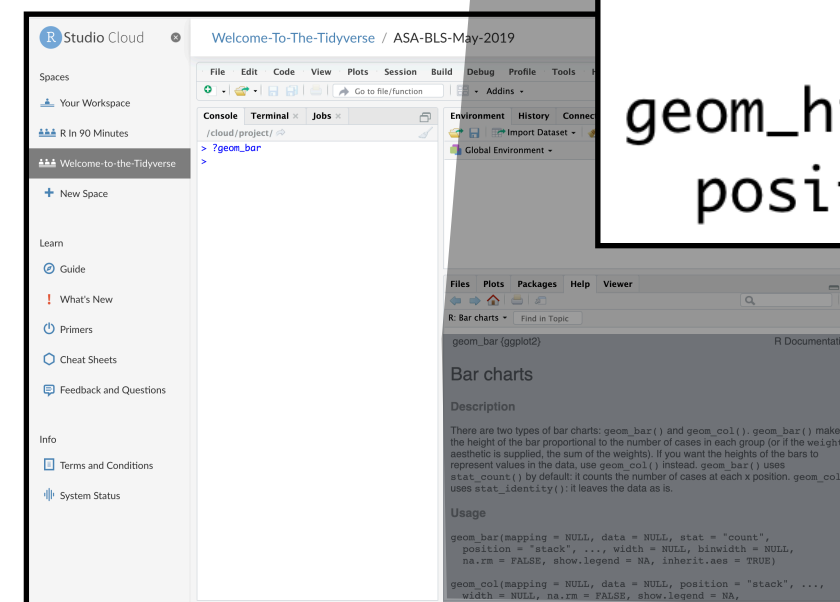
Description

Visualise the distribution of a single continuous variable by dividing the x axis into bins and counting the number of observations in each bin. Histograms (`geom_histogram()`) display the counts with bars; frequency polygons (`geom_freqpoly()`) display the counts with lines. Frequency polygons are more suitable when you want to compare the distribution across the levels of a categorical variable.

Usage

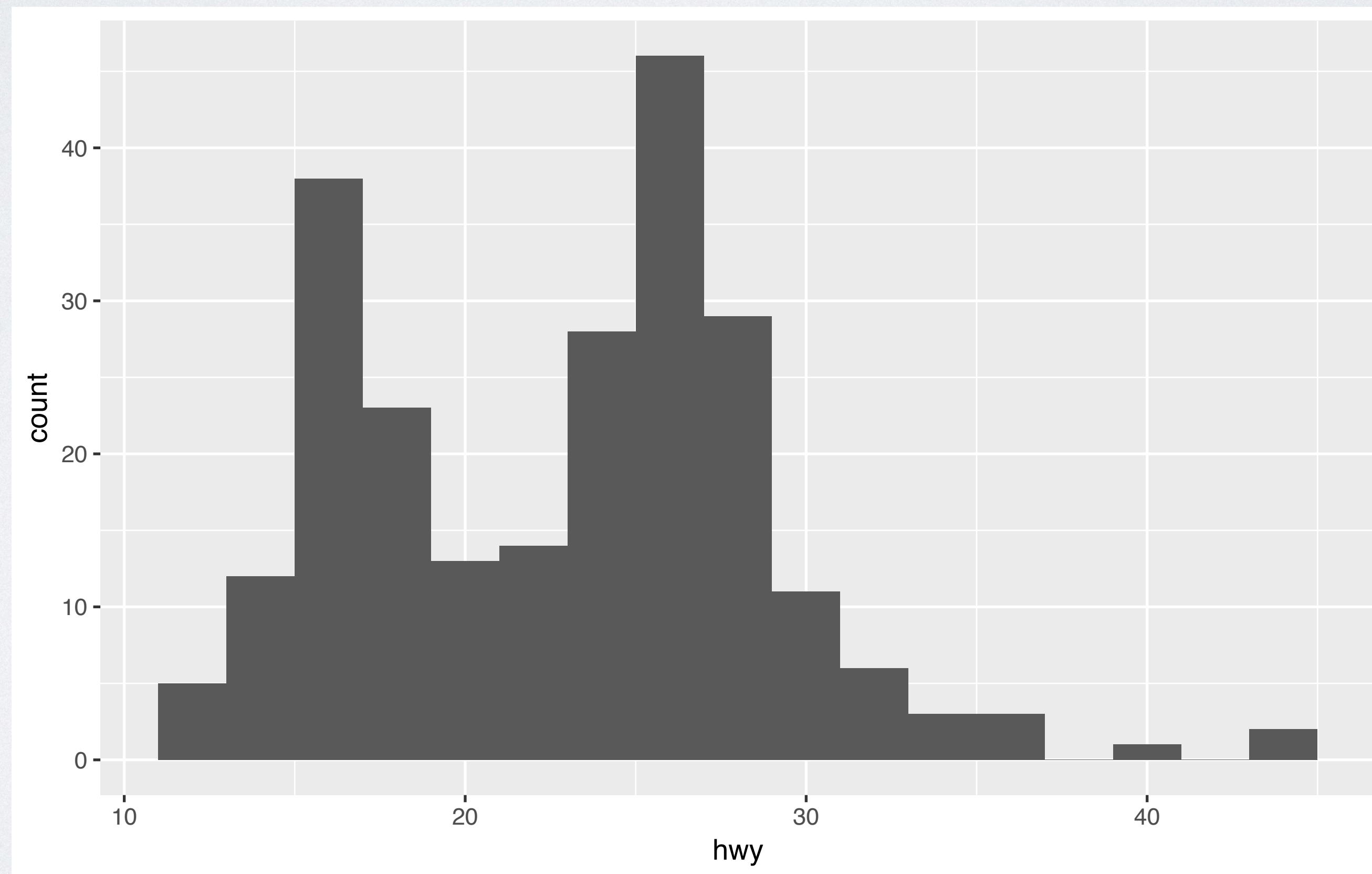
```
geom_freqpoly(mapping = NULL, data = NULL, stat = "bin",  
              position = "identity", ..., na.rm = FALSE, show.legend = NA,  
              inherit.aes = TRUE)
```

```
geom_histogram(mapping = NULL, data = NULL, stat = "bin",  
               position = "stack", ..., binwidth = NULL, bins = NULL,
```

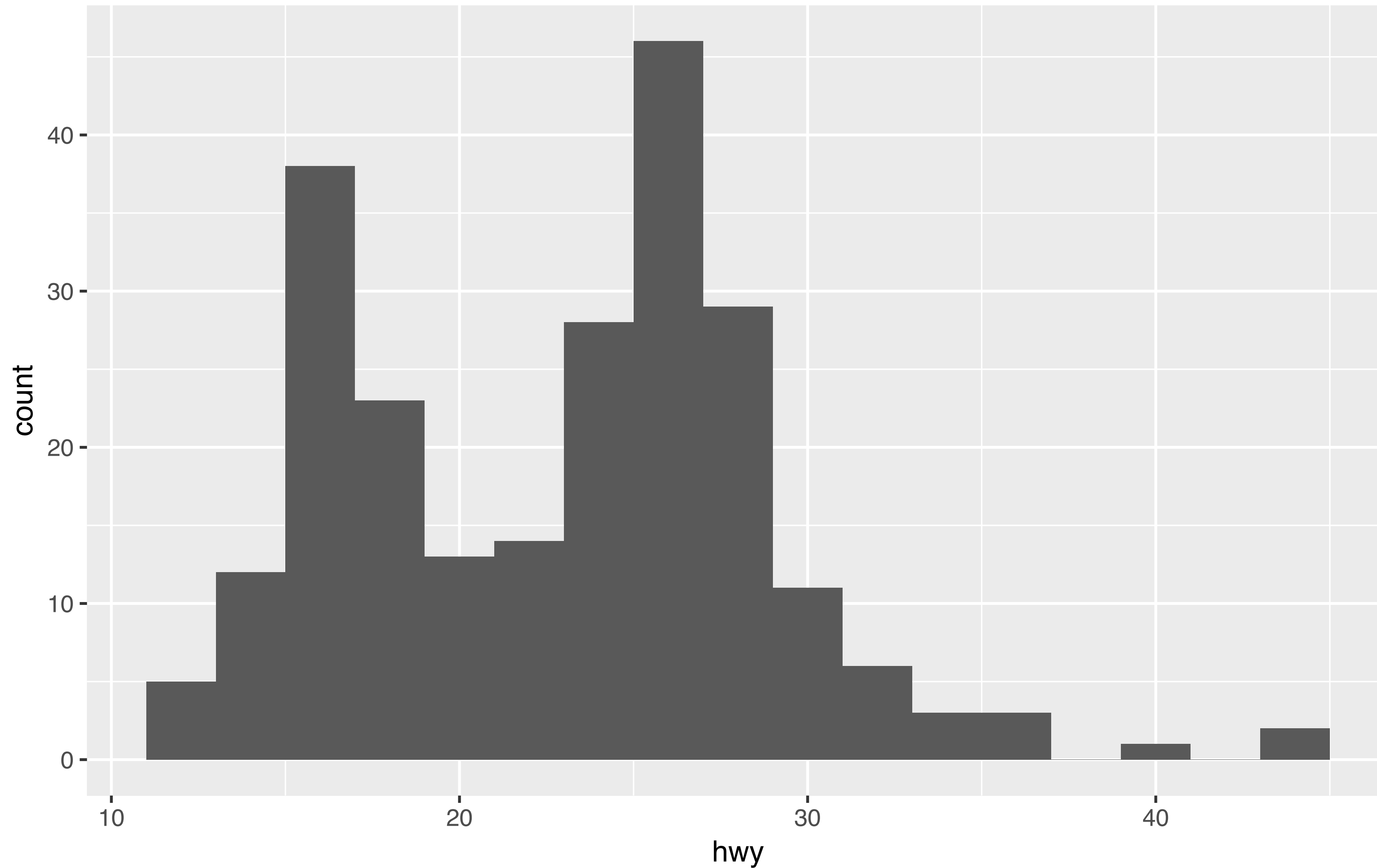


Your Turn 6

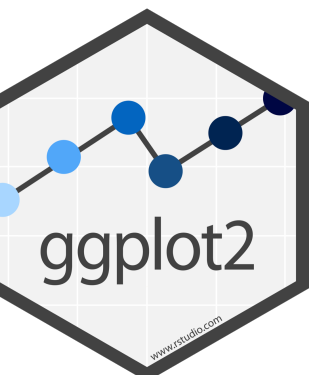
Use the help page for `geom_histogram` to make the bins 2 mpg wide.



02:00



```
ggplot(data = mpg) +  
  geom_histogram(mapping = aes(x = hwy), binwidth = 2)
```



ggplot2.tidyverse.org

ggplot2 3.4.4 Reference News ▾ Articles ▾ Extensions

Search for



ggplot2

Overview

ggplot2 is a system for declaratively creating graphics, based on [The Grammar of Graphics](#). You provide the data, tell ggplot2 how to map variables to aesthetics, what graphical primitives to use, and it takes care of the details.

Installation

```
# The easiest way to get ggplot2 is to install the whole tidyverse:  
install.packages\("tidyverse"\)
```

LINKS

[View on CRAN](#)

[Browse source code](#)

[Report a bug](#)

[Learn more](#)

[Extensions](#)

LICENSE

[Full license](#)

[MIT](#) + file [LICENSE](#)

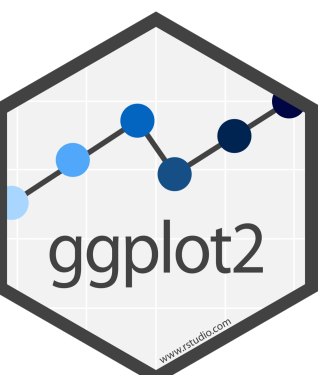
COMMUNITY

[Contributing guide](#)

[Code of conduct](#)

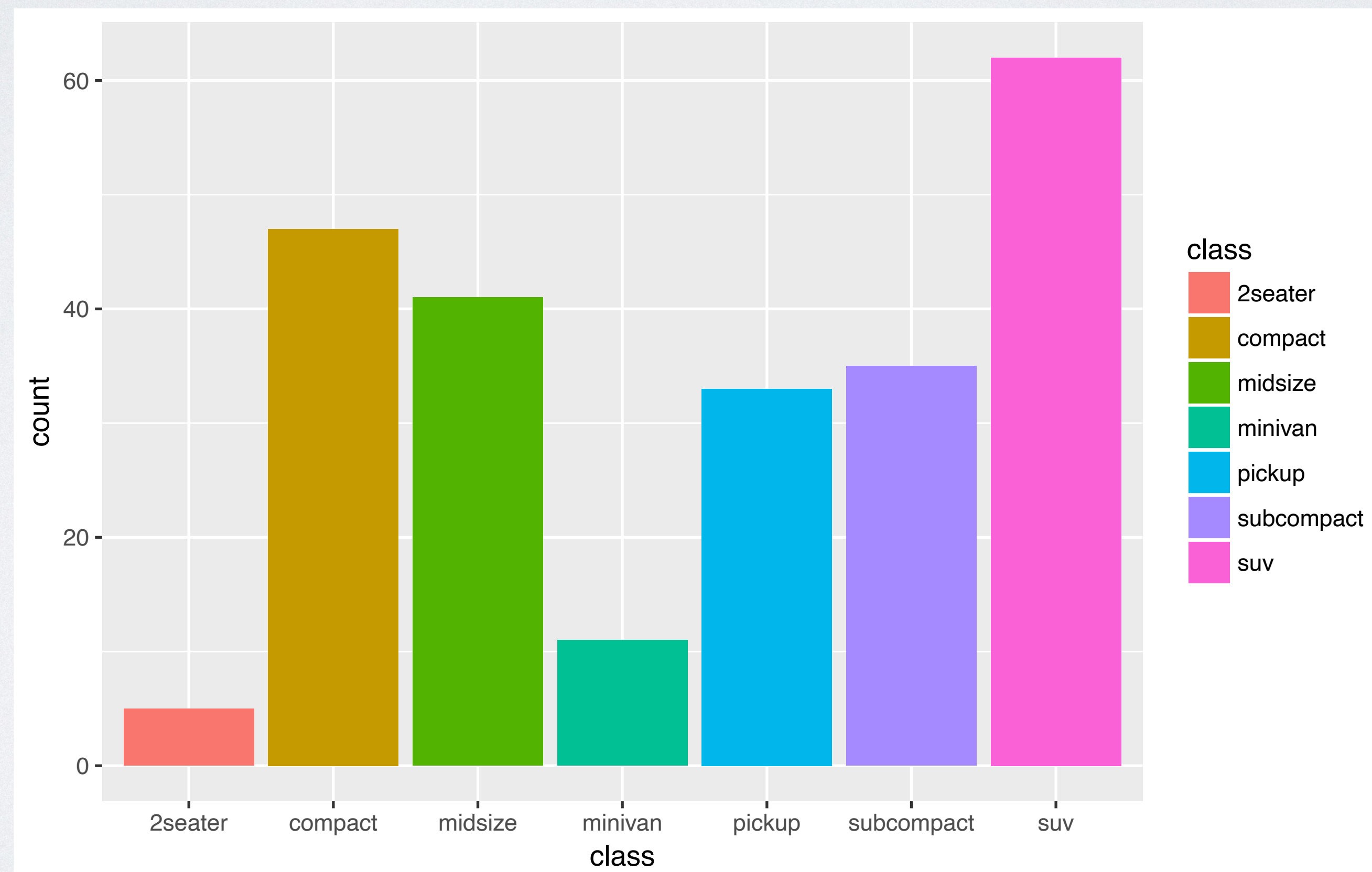
CITATION

[Citing ggplot2](#)

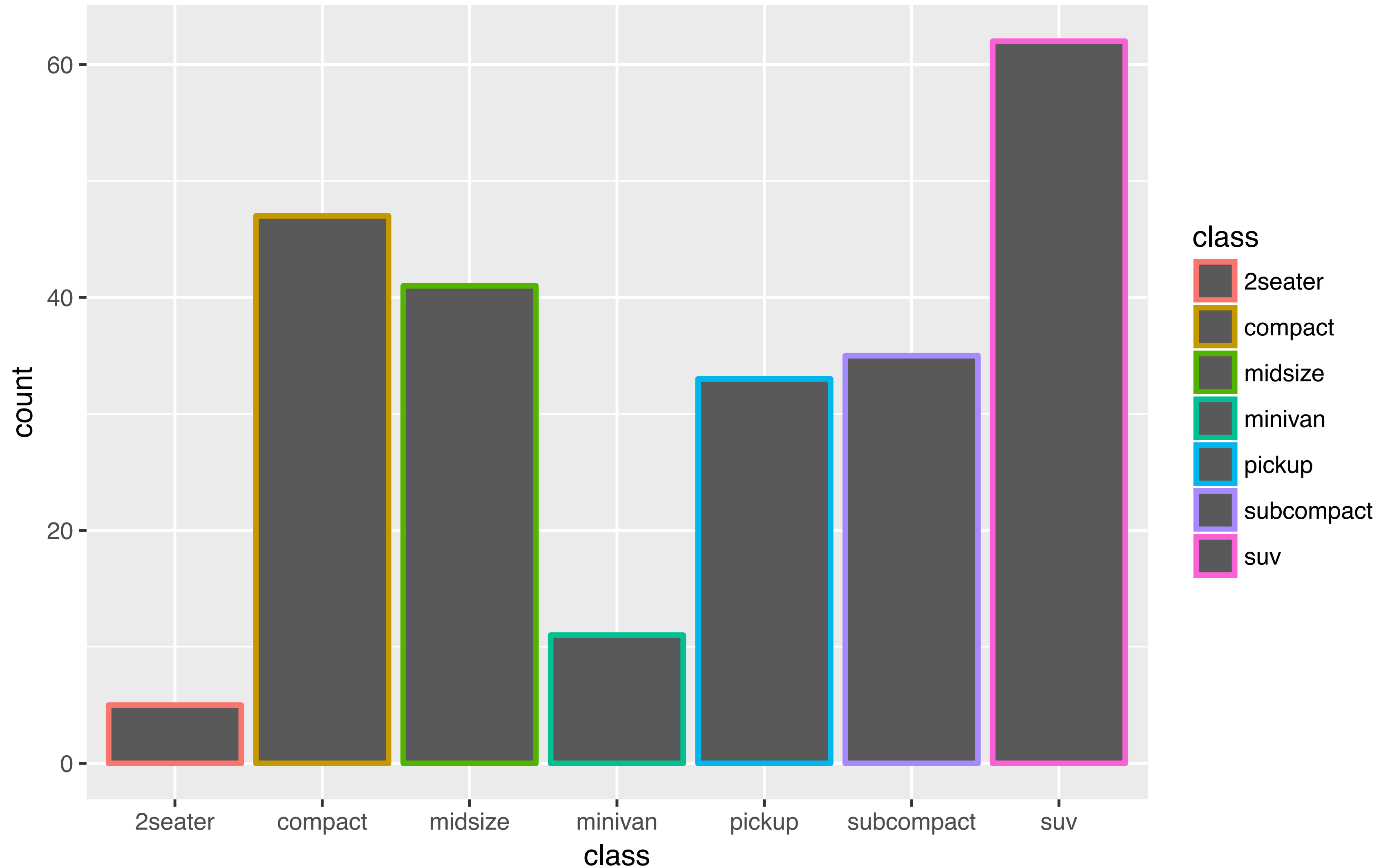


Your Turn 7

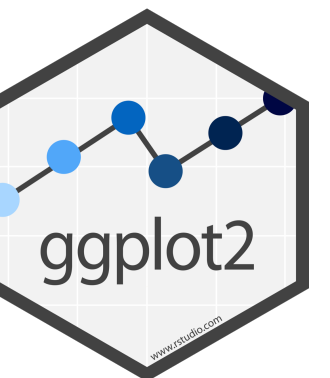
Make the bar chart of **class** below. Use the cheatsheet. Hint: do not supply a **y** variable.

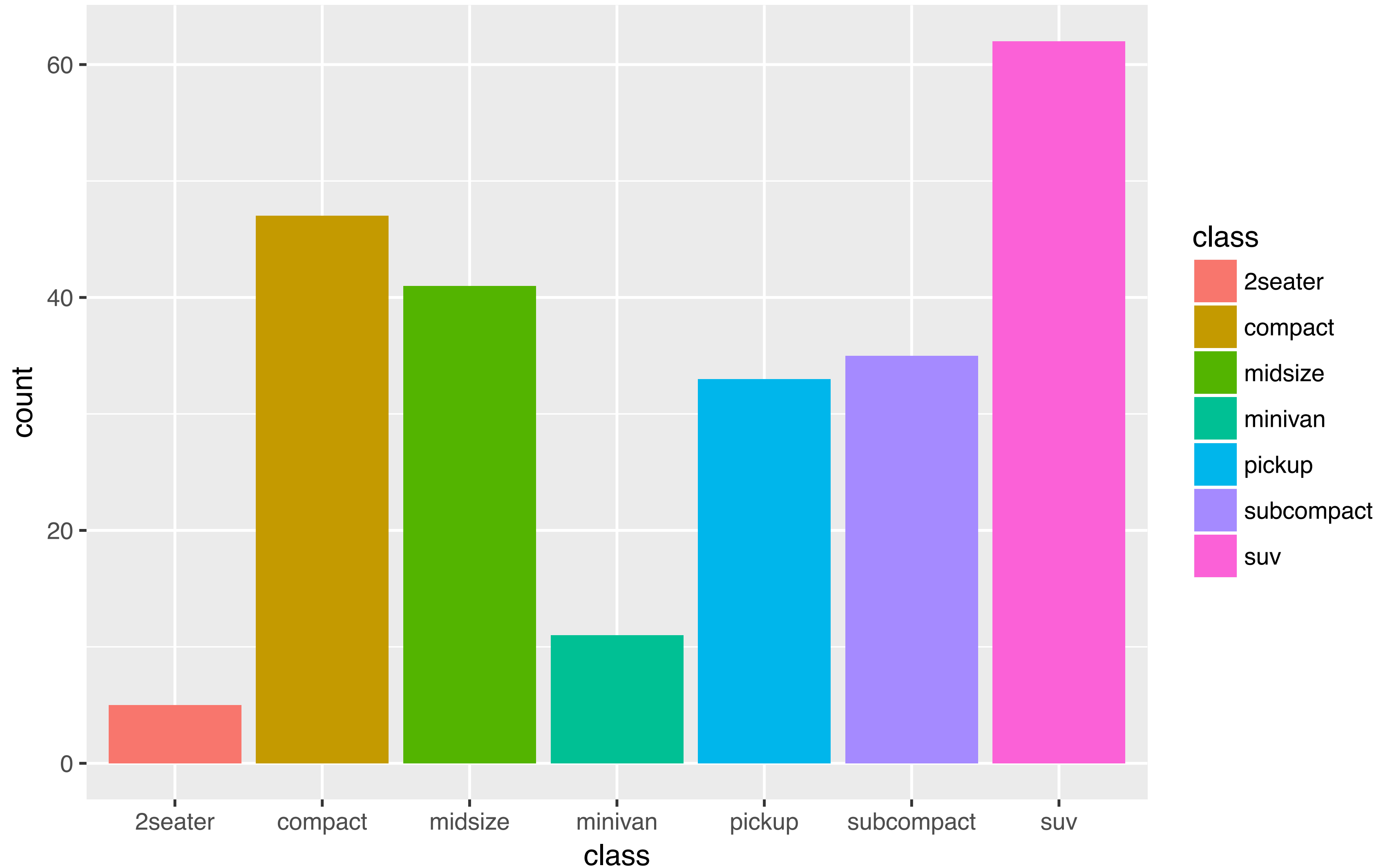


02:00

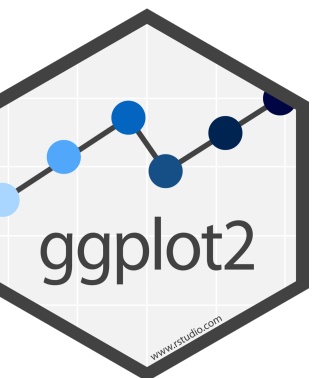


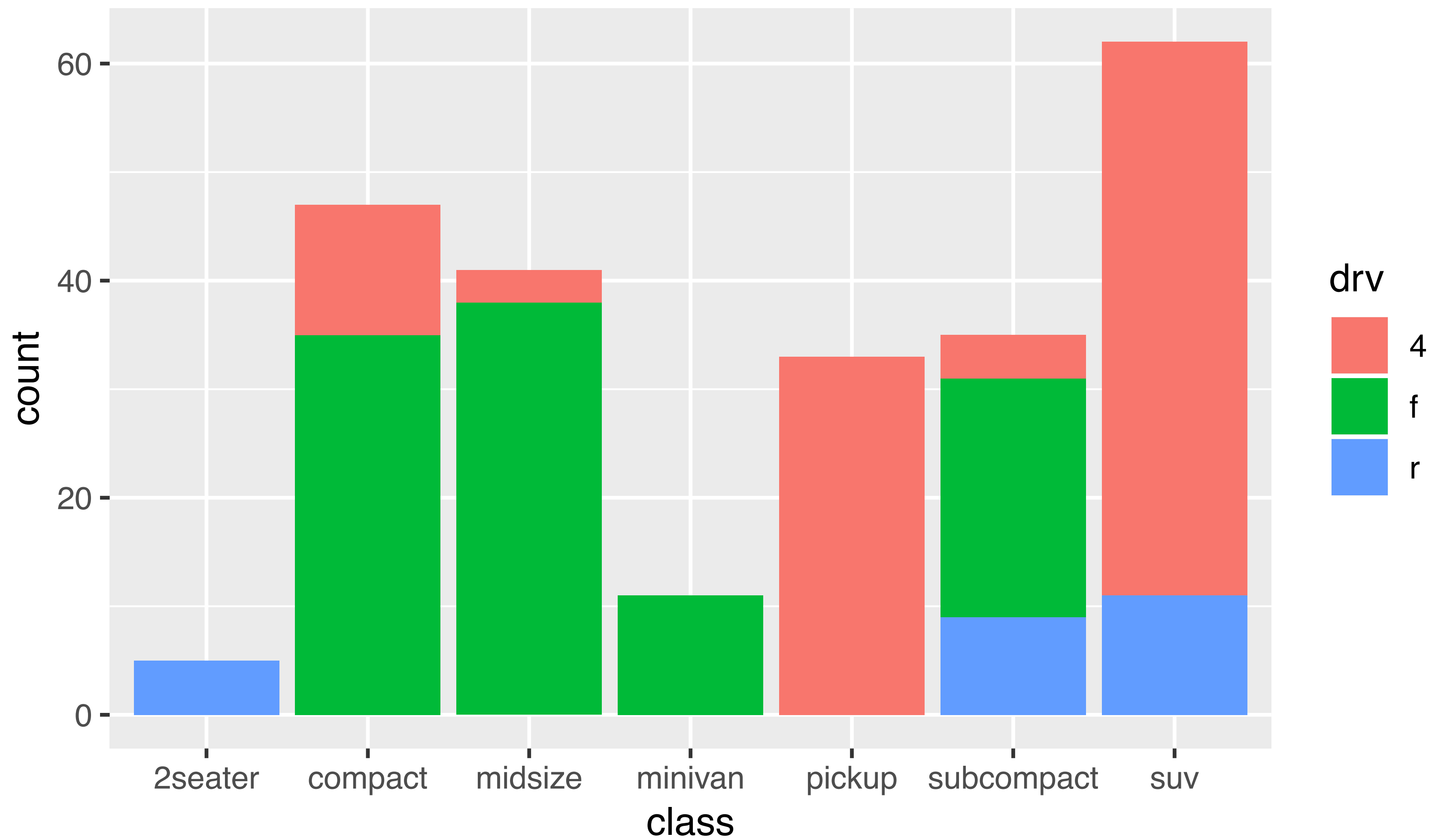
```
ggplot(data = mpg) +  
  geom_bar(mapping = aes(x = class, color = class))
```



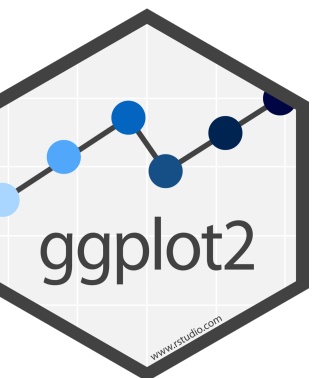


```
ggplot(data = mpg) +  
  geom_bar(mapping = aes(x = class, fill = class))
```





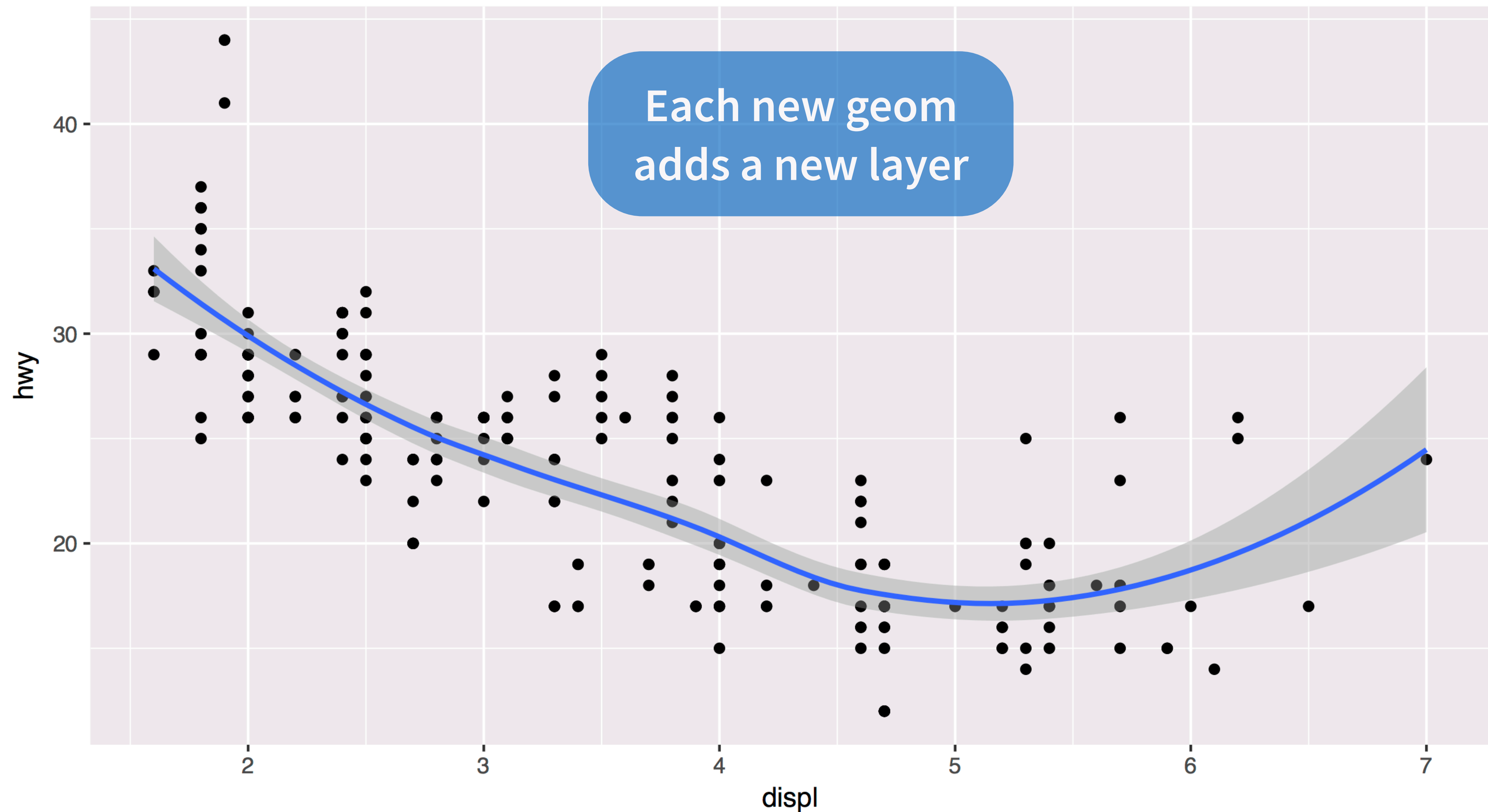
```
ggplot(data = mpg) +  
  geom_bar(mapping = aes(x = class, fill = drv))
```



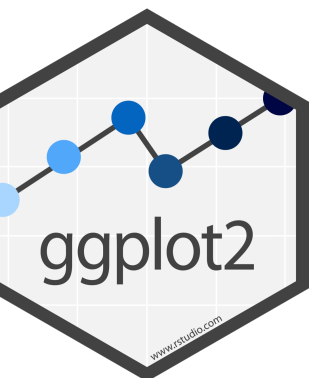
Quiz

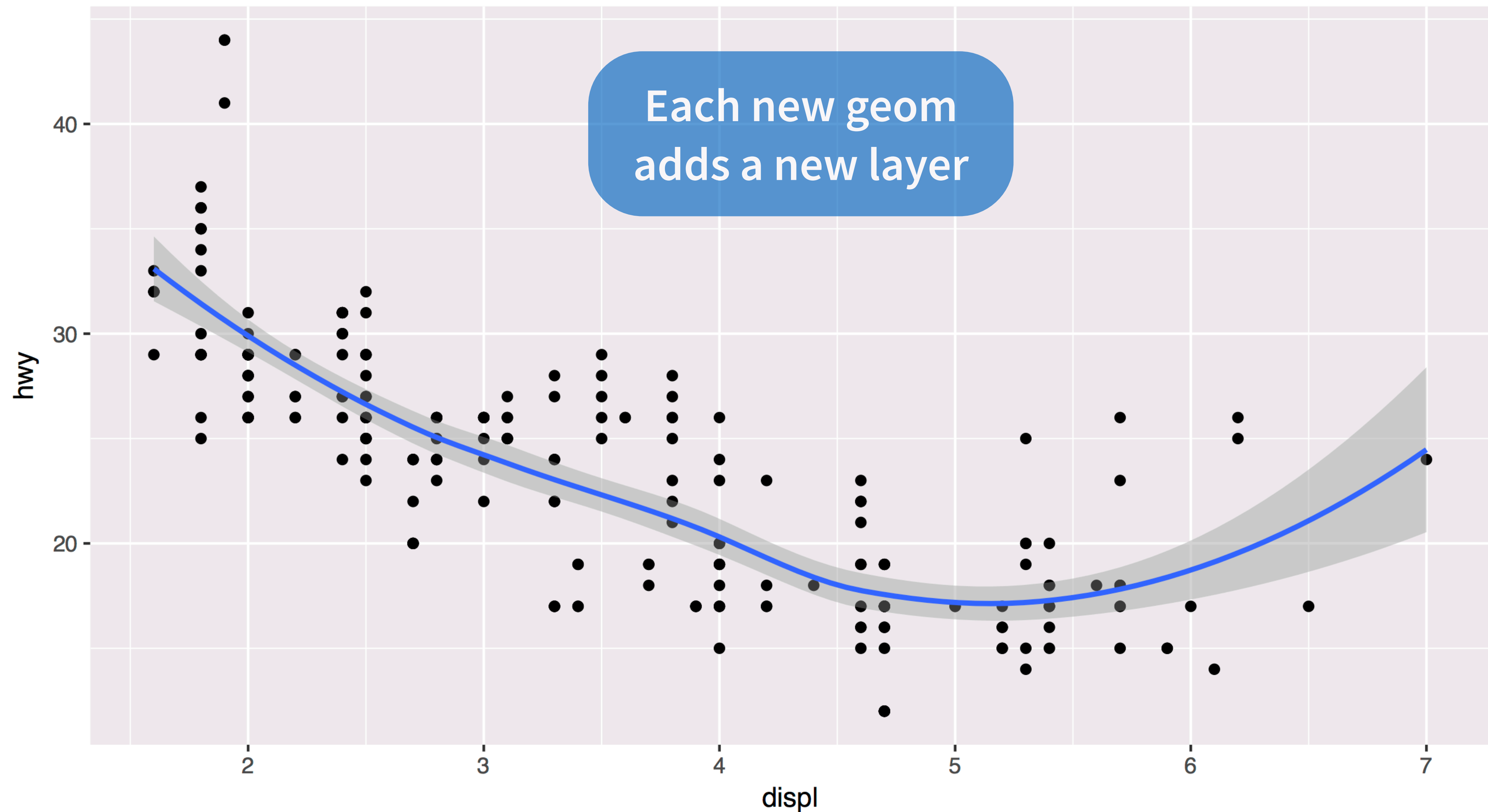
What will this code do?

```
ggplot(mpg) +  
  geom_point(aes(displ, hwy)) +  
  geom_smooth(aes(displ, hwy))
```

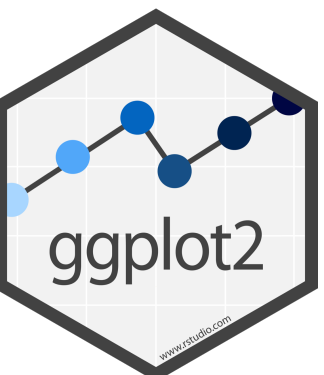



```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  geom_smooth(mapping = aes(x = displ, y = hwy))
```



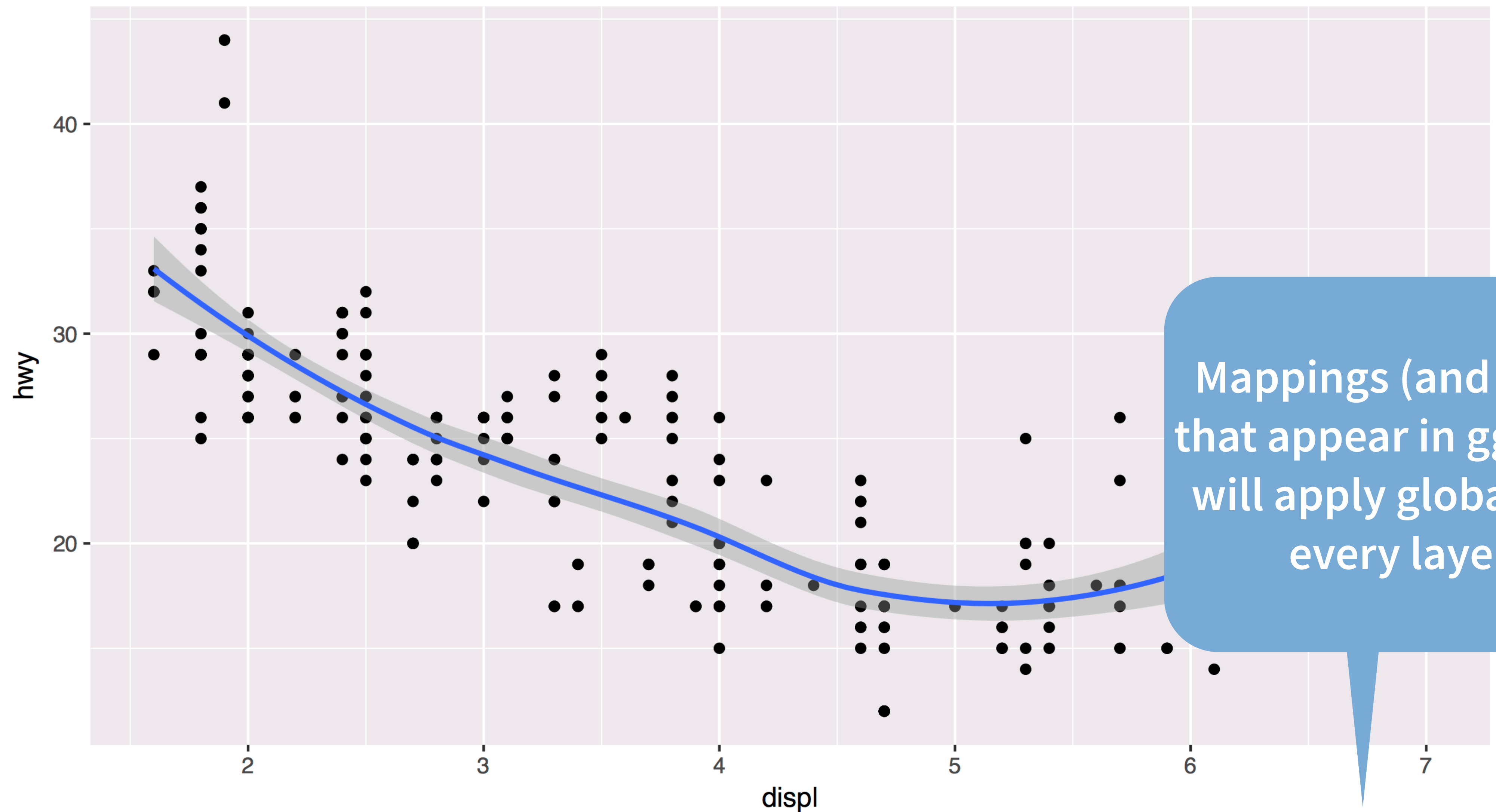


```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  geom_smooth(mapping = aes(x = displ, y = hwy))
```

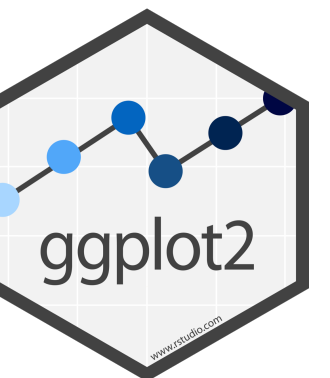


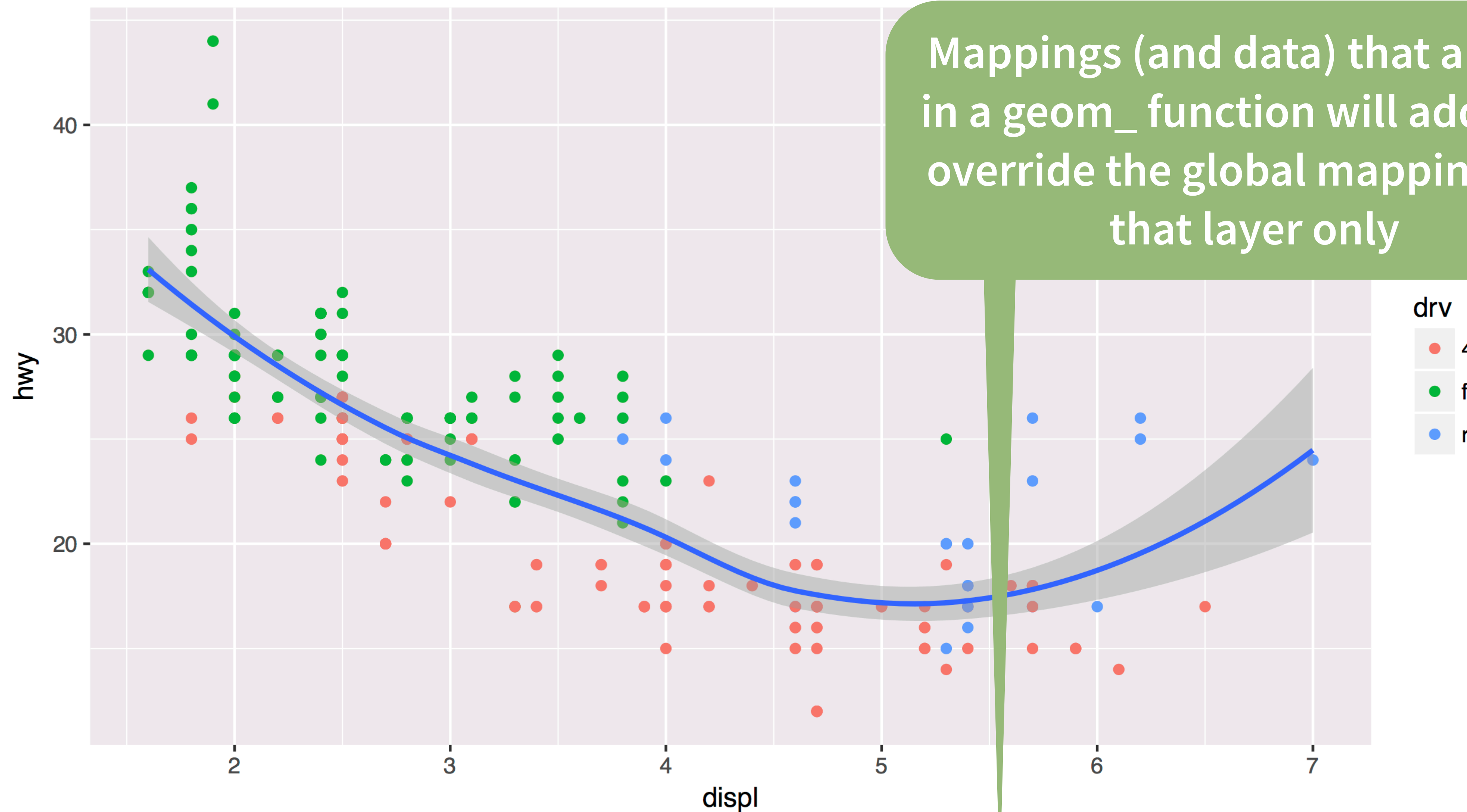
global vs. local



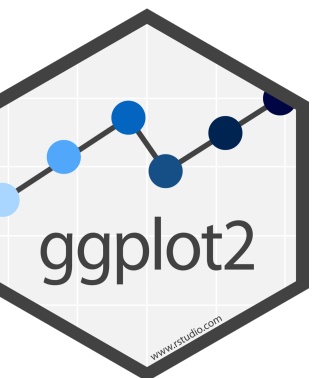


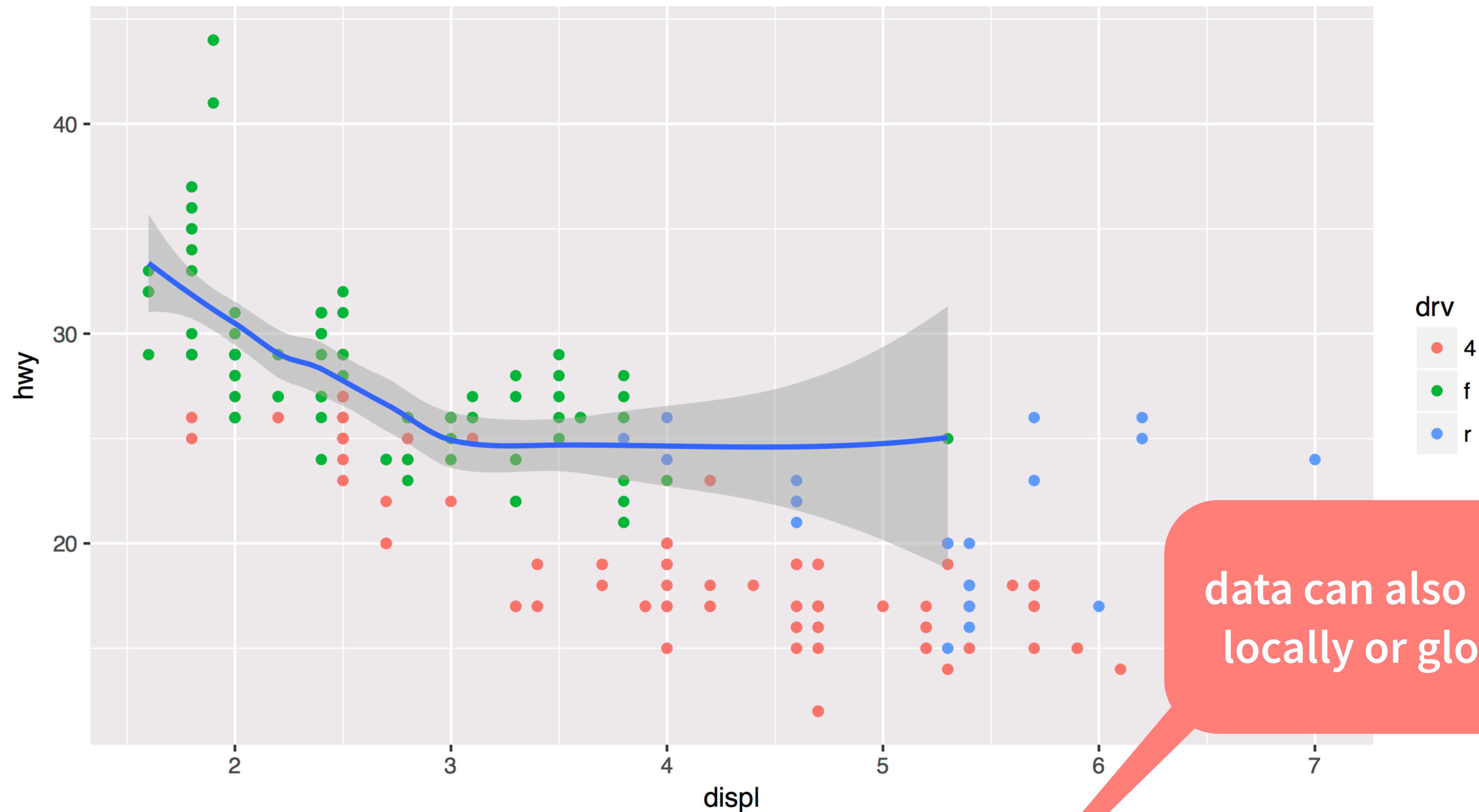
```
ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +  
  geom_point() +  
  geom_smooth()
```





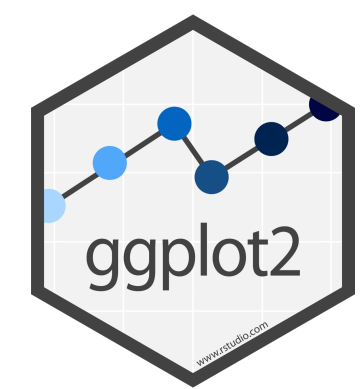
```
ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +
  geom_point(mapping = aes(color = drv)) +
  geom_smooth()
```





data can also be set locally or globally

```
ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +  
  geom_point(mapping = aes(color = drv)) +  
  geom_smooth(data = filter(mpg, drv == "f"))
```



Quiz

What is different about this plot? Run the code!

```
p <- ggplot(mpg) +  
  geom_point(aes(displ, hwy)) +  
  geom_smooth(aes(displ, hwy))
```

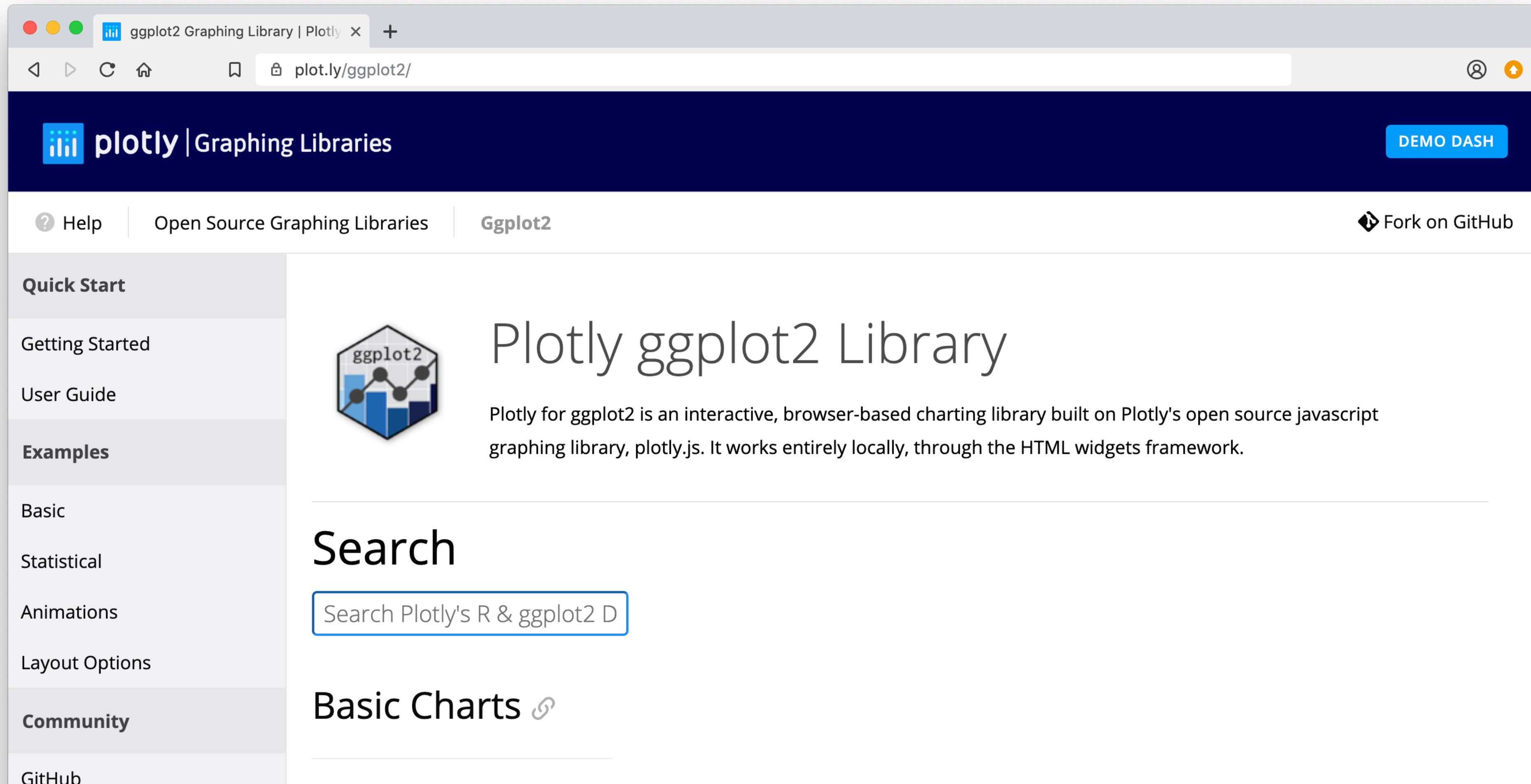
```
library(plotly)  
ggplotly(p)
```


interactivity



Plotly

Tools for making interactive plots. plot.ly/ggplot2/



The screenshot shows a web browser window with the URL `plot.ly/ggplot2/`. The page features a dark blue header with the Plotly logo and the text "plotly | Graphing Libraries". A "DEMO DASH" button is visible in the top right corner. Below the header, there is a navigation bar with links for "Help", "Open Source Graphing Libraries", and "Ggplot2", along with a "Fork on GitHub" button. The main content area is titled "Plotly ggplot2 Library" and includes a description: "Plotly for ggplot2 is an interactive, browser-based charting library built on Plotly's open source javascript graphing library, plotly.js. It works entirely locally, through the HTML widgets framework." A search bar is present with the placeholder text "Search Plotly's R & ggplot2 D". Below the search bar, there is a link for "Basic Charts". A sidebar on the left contains a "Quick Start" section with links for "Getting Started" and "User Guide", an "Examples" section with links for "Basic", "Statistical", "Animations", and "Layout Options", and a "Community" section with a link for "GitHub".

ggplot2 Graphing Library | Plotly x +

plot.ly/ggplot2/

plotly | Graphing Libraries

DEMO DASH

Help | Open Source Graphing Libraries | Ggplot2 | Fork on GitHub

Quick Start

- Getting Started
- User Guide

Examples

- Basic
- Statistical
- Animations
- Layout Options

Community

- GitHub

ggplot2

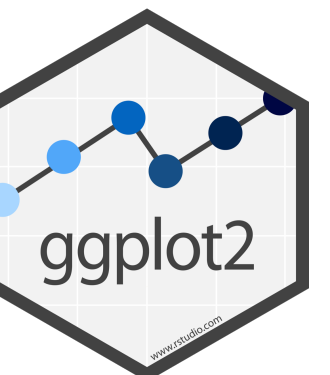
Plotly ggplot2 Library

Plotly for ggplot2 is an interactive, browser-based charting library built on Plotly's open source javascript graphing library, plotly.js. It works entirely locally, through the HTML widgets framework.

Search

Search Plotly's R & ggplot2 D

Basic Charts [↗](#)



Saving graphs



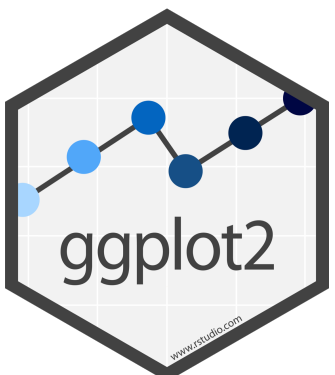
GUI method

Right click on the plot

The screenshot shows the RStudio interface with a ggplot2 scatter plot. The plot displays highway mileage (hwy) on the y-axis (ranging from 20 to 40) and engine displacement (displ) on the x-axis (ranging from 2 to 7). The data points are colored by car class: minivan (green), pickup (blue), subcompact (purple), and suv (pink). A right-click context menu is open over the plot, with the 'Save image as...' option highlighted. The menu also includes 'Copy Image', 'Copy Image Address', 'Reload', and 'Inspect Element'. The 'Inspect Element' option shows a legend for the car classes.

```
12  
13 {r}  
14 ggplot(data = mpg) +  
15   geom_point(aes(x = displ, y = hwy, color = class))  
16 }
```

Class	displ	hwy
minivan	2.0	30
minivan	2.0	31
minivan	2.0	32
minivan	2.0	33
minivan	2.0	34
minivan	2.0	35
minivan	2.0	36
minivan	2.0	37
minivan	2.0	38
minivan	2.0	39
minivan	2.0	40
minivan	2.0	41
minivan	2.0	42
minivan	2.0	43
minivan	2.0	44
minivan	2.0	45
minivan	2.0	46
minivan	2.0	47
minivan	2.0	48
minivan	2.0	49
minivan	2.0	50
minivan	2.0	51
minivan	2.0	52
minivan	2.0	53
minivan	2.0	54
minivan	2.0	55
minivan	2.0	56
minivan	2.0	57
minivan	2.0	58
minivan	2.0	59
minivan	2.0	60
minivan	2.0	61
minivan	2.0	62
minivan	2.0	63
minivan	2.0	64
minivan	2.0	65
minivan	2.0	66
minivan	2.0	67
minivan	2.0	68
minivan	2.0	69
minivan	2.0	70
minivan	2.0	71
minivan	2.0	72
minivan	2.0	73
minivan	2.0	74
minivan	2.0	75
minivan	2.0	76
minivan	2.0	77
minivan	2.0	78
minivan	2.0	79
minivan	2.0	80
minivan	2.0	81
minivan	2.0	82
minivan	2.0	83
minivan	2.0	84
minivan	2.0	85
minivan	2.0	86
minivan	2.0	87
minivan	2.0	88
minivan	2.0	89
minivan	2.0	90
minivan	2.0	91
minivan	2.0	92
minivan	2.0	93
minivan	2.0	94
minivan	2.0	95
minivan	2.0	96
minivan	2.0	97
minivan	2.0	98
minivan	2.0	99
minivan	2.0	100



Code method

ggsave() saves the last plot.

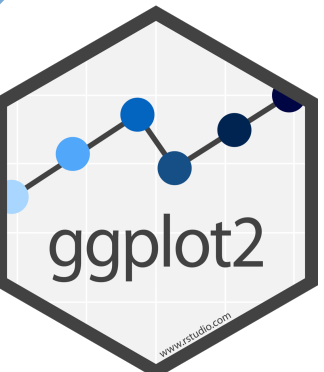
Uses size on screen:

```
ggsave("my-plot.pdf")  
ggsave("my-plot.png")
```

Specify size in inches

```
ggsave("my-plot.pdf", width = 10, height = 10)
```

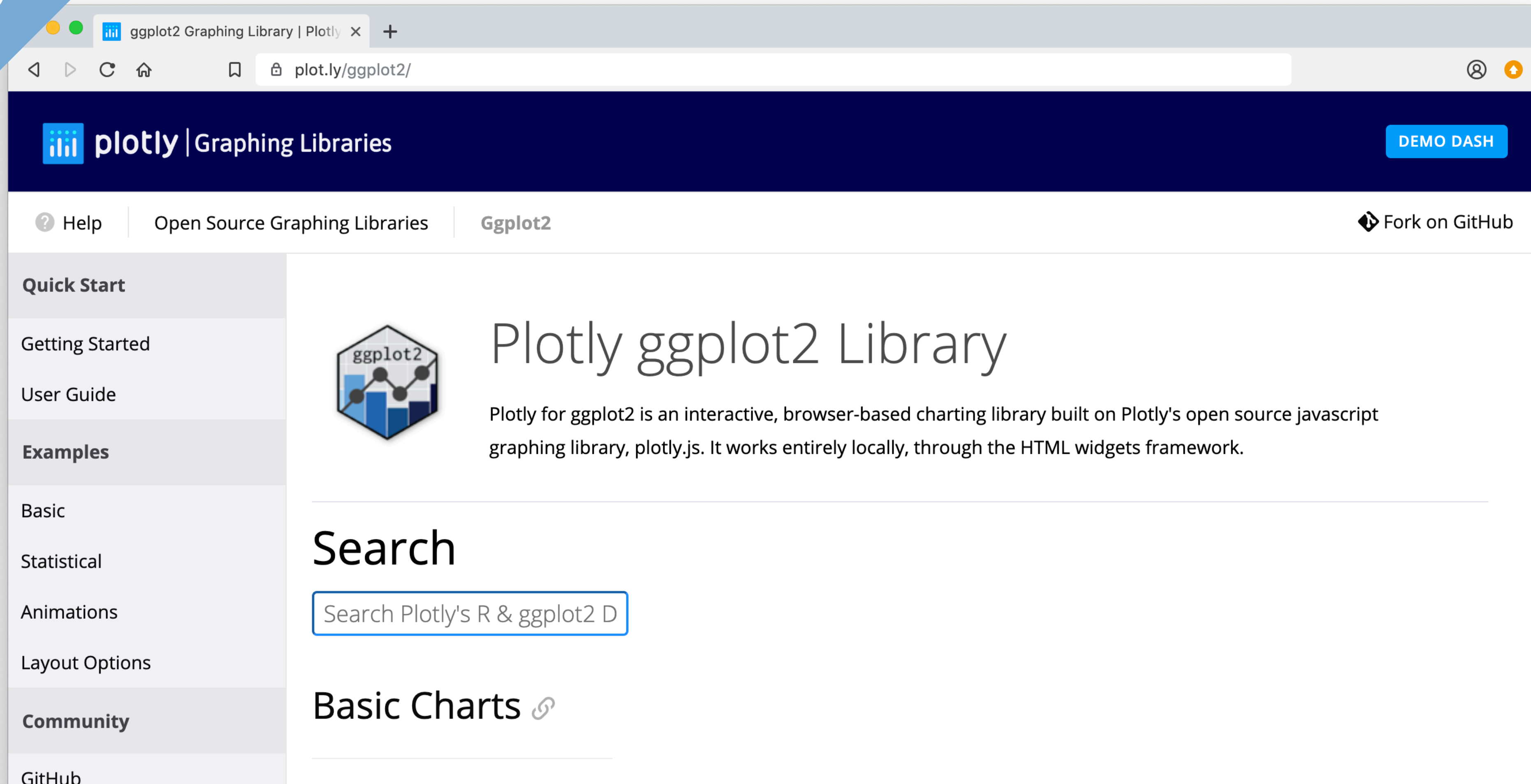
Q: But where will
it save it?
A: Alongside
your .qmd



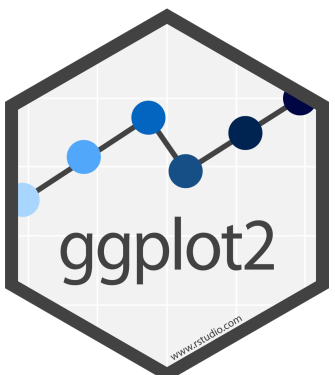
Save with
htmlwidgets::saveWidget()

Plotly

for making interactive plots. plot.ly/ggplot2/



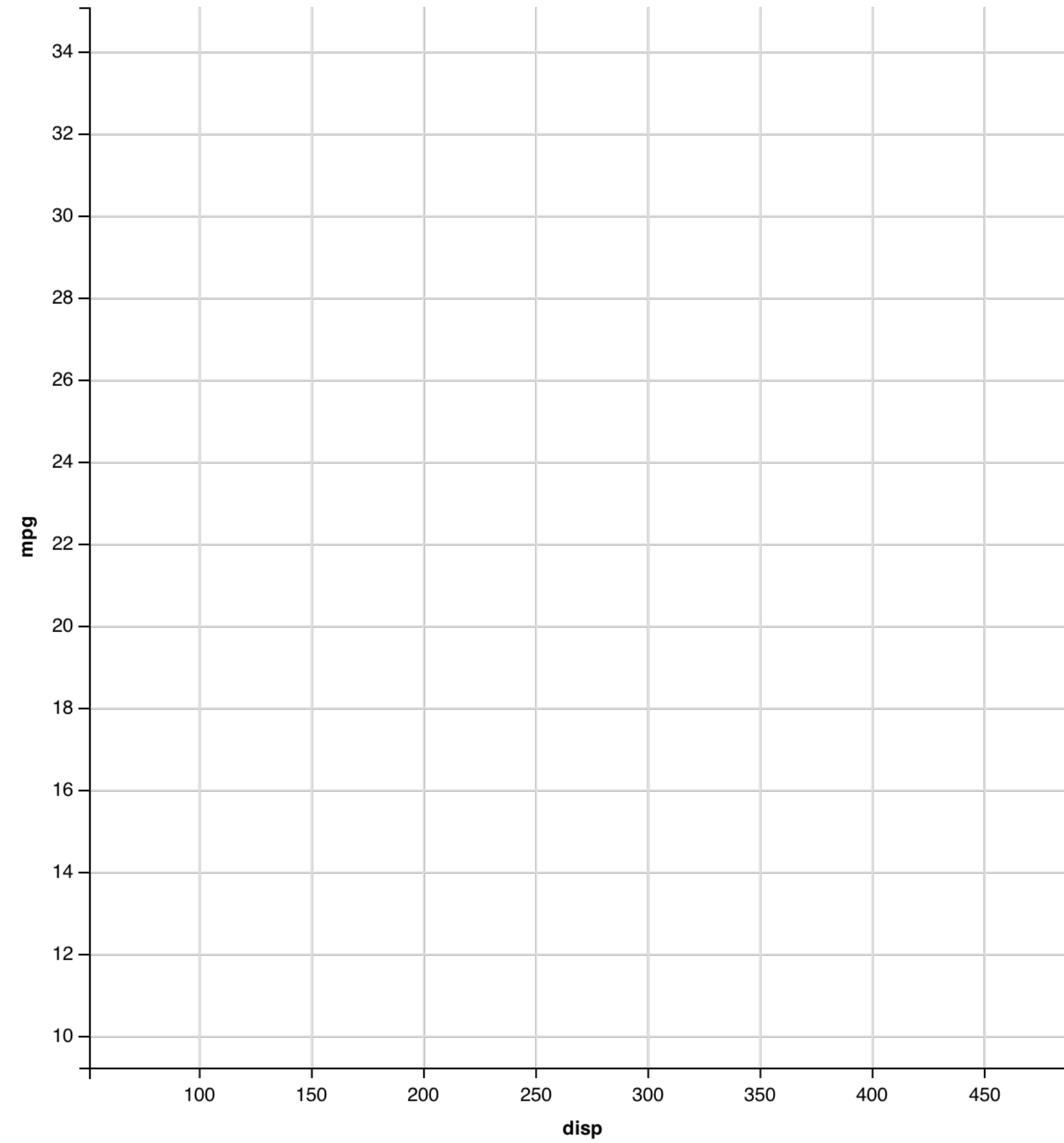
The screenshot shows the Plotly ggplot2 Library website. The browser address bar displays 'plot.ly/ggplot2/'. The page header includes the Plotly logo and 'Graphing Libraries' with a 'DEMO DASH' button. The navigation menu contains 'Help', 'Open Source Graphing Libraries', 'Ggplot2', and 'Fork on GitHub'. A left sidebar lists navigation options: 'Quick Start', 'Getting Started', 'User Guide', 'Examples', 'Basic', 'Statistical', 'Animations', 'Layout Options', 'Community', and 'GitHub'. The main content area features the 'ggplot2' logo, the title 'Plotly ggplot2 Library', and a description: 'Plotly for ggplot2 is an interactive, browser-based charting library built on Plotly's open source javascript graphing library, plotly.js. It works entirely locally, through the HTML widgets framework.' Below this is a 'Search' section with a search box containing the text 'Search Plotly's R & ggplot2 D'. At the bottom, there is a link for 'Basic Charts'.



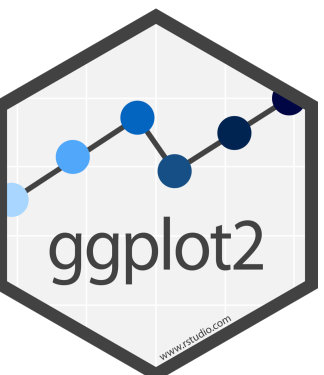
Grammar of Graphics



mpg	cyl	disp	hp
21,0	6	160,0	2
21,0	6	160,0	2
22,8	4	108,0	1
21,4	6	258,0	2
18,7	8	360,0	3
18,1	6	225,0	2
14,3	8	360,0	5
24,4	4	146,7	1
22,8	4	140,8	1
19,2	6	167,6	2
17,8	6	167,6	2
16,4	8	275,8	3
17,3	8	275,8	3
15,2	8	275,8	3
10,4	8	472,0	4
10,4	8	460,0	4
14,7	8	440,0	4
32,4	4	78,7	1
30,4	4	75,7	1
33,9	4	71,1	1



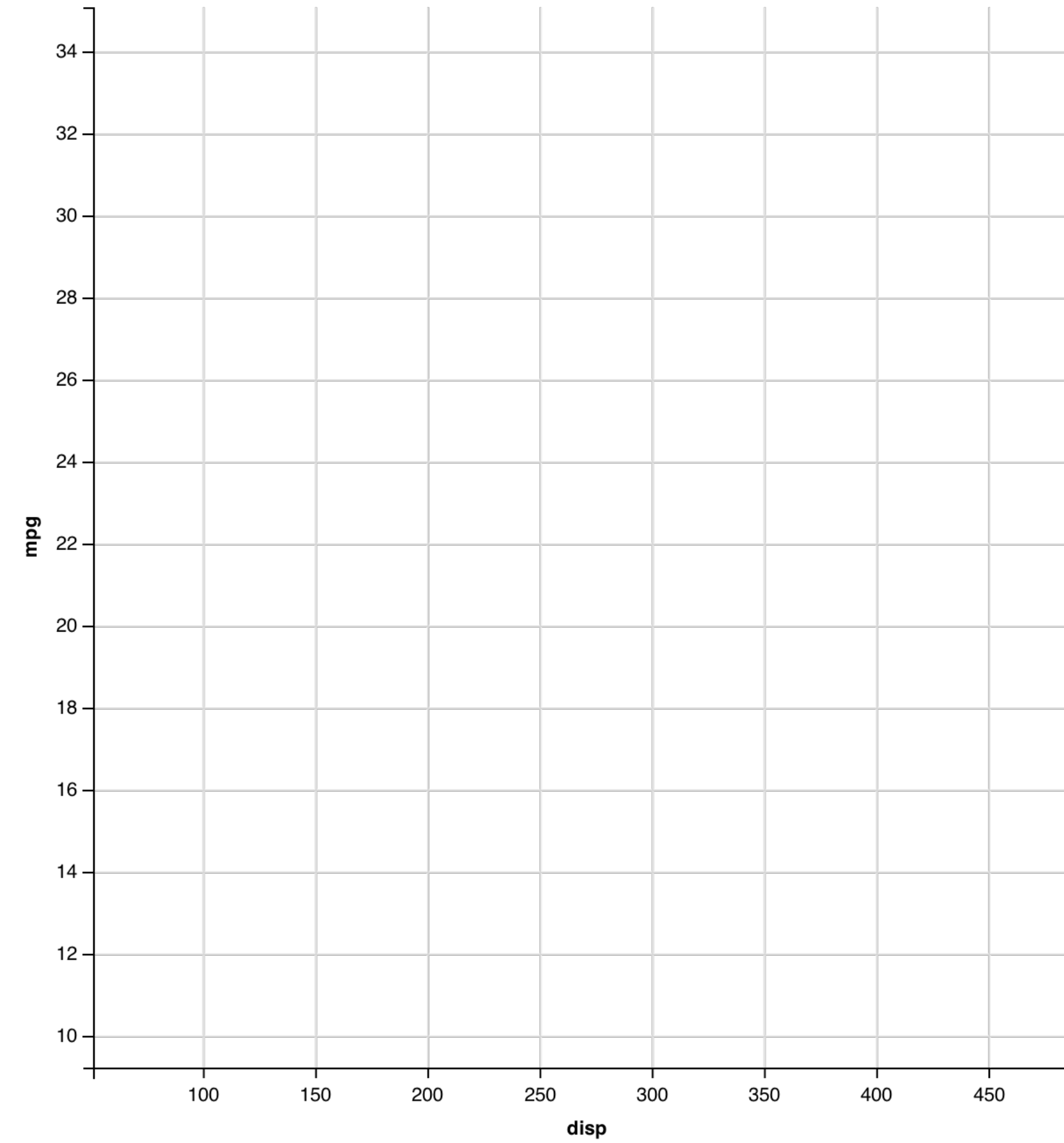
data geom



mappings

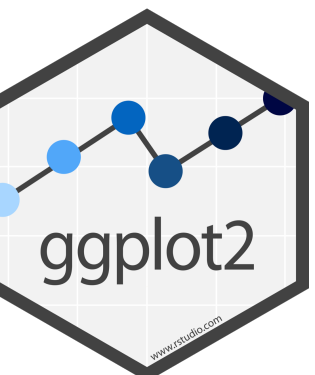
fill
↑↓

mpg	cyl	disp	hp	fill
21,0	6	160,0	2	●
21,0	6	160,0	2	●
22,8	4	108,0	1	●
21,4	6	258,0	2	●
18,7	8	360,0	3	●
18,1	6	225,0	2	●
14,3	8	360,0	5	●
24,4	4	146,7	1	●
22,8	4	140,8	1	●
19,2	6	167,6	2	●
17,8	6	167,6	2	●
16,4	8	275,8	3	●
17,3	8	275,8	3	●
15,2	8	275,8	3	●
10,4	8	472,0	4	●
10,4	8	460,0	4	●
14,7	8	440,0	4	●
32,4	4	78,7	1	●
30,4	4	75,7	1	●
33,9	4	71,1	1	●



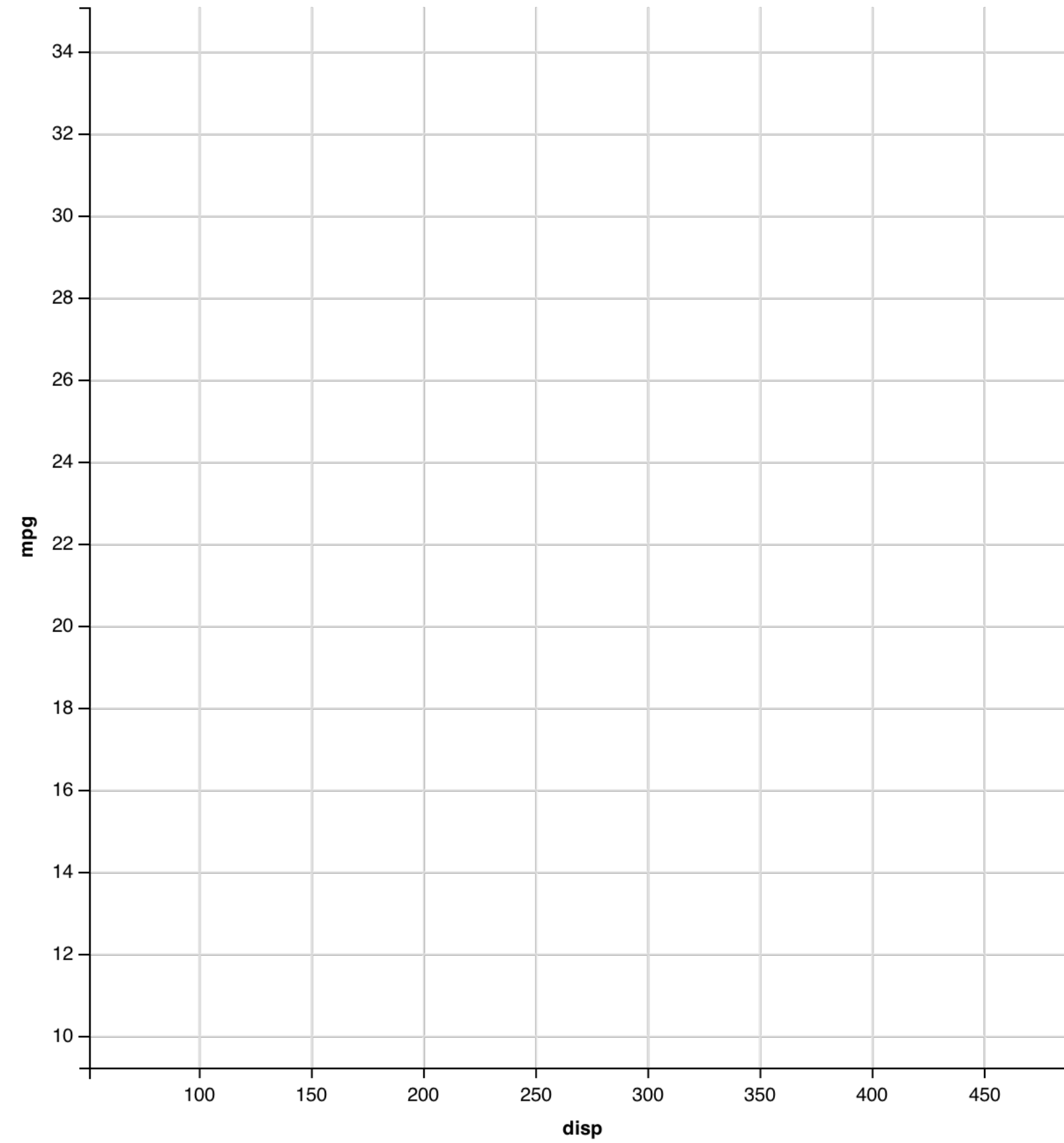
data

geom



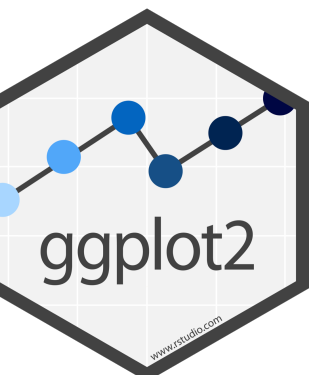
mappings

mpg	shape		hp	fill
	cyl	disp		
21,0	6 +	160,0	2	+
21,0	6 +	160,0	2	+
22,8	4 ●	108,0	1	●
21,4	6 +	258,0	2	+
18,7	8 ◆	360,0	3	◆
18,1	6 +	225,0	2	+
14,3	8 ◆	360,0	5	◆
24,4	4 ●	146,7	1	●
22,8	4 ●	140,8	1	●
19,2	6 +	167,6	2	+
17,8	6 +	167,6	2	+
16,4	8 ◆	275,8	3	◆
17,3	8 ◆	275,8	3	◆
15,2	8 ◆	275,8	3	◆
10,4	8 ◆	472,0	4	◆
10,4	8 ◆	460,0	4	◆
14,7	8 ◆	440,0	4	◆
32,4	4 ●	78,7	1	●
30,4	4 ●	75,7	1	●
33,9	4 ●	71,1	1	●



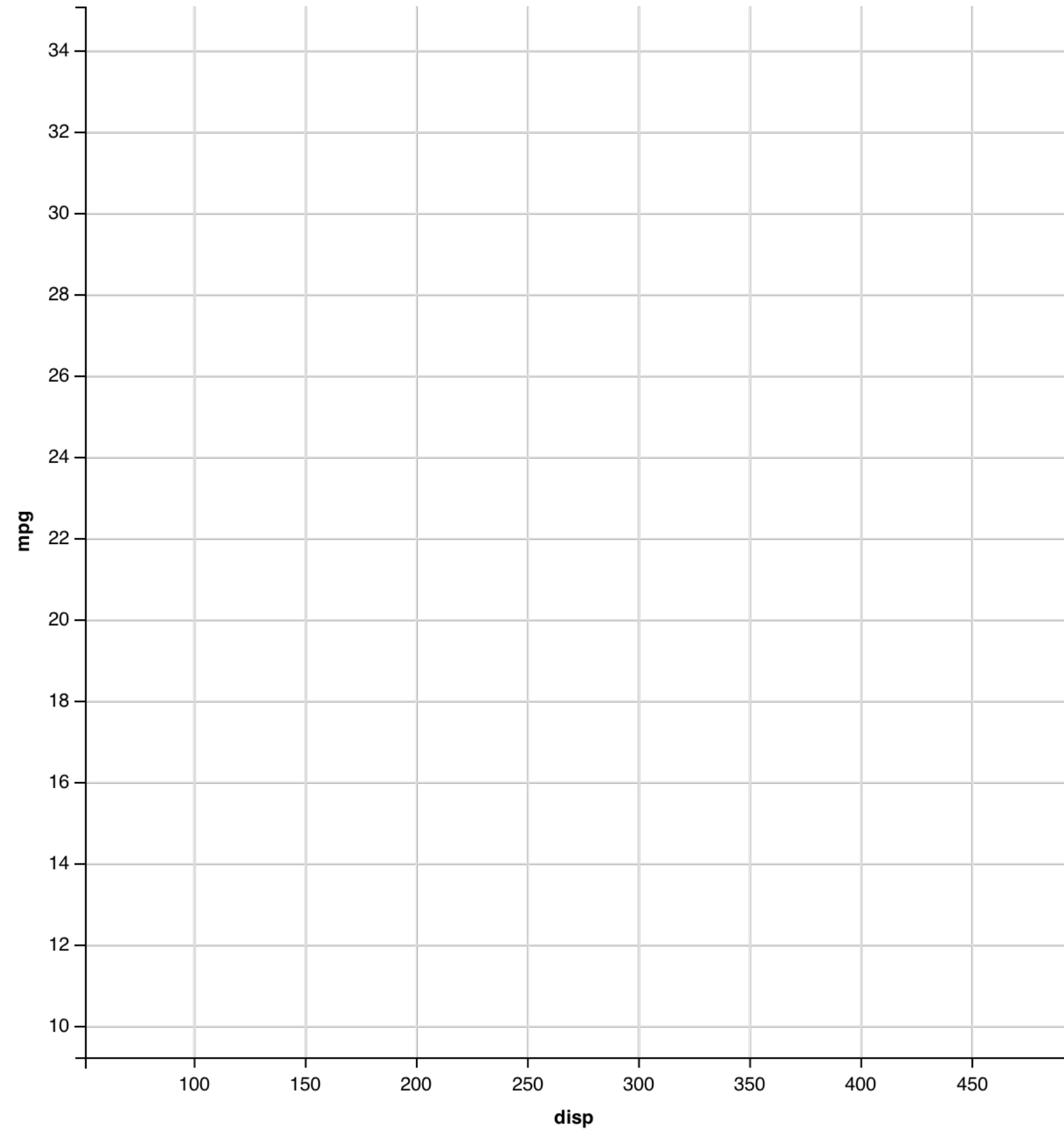
data

geom



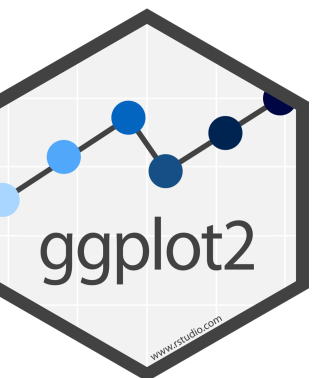
mappings

mpg	shape ↑↑ cyl	x ↑↓ disp	fill ↑↑ hp	
21,0	6	160,0	2	+
21,0	6	160,0	2	+
22,8	4	108,0	1	●
21,4	6	258,0	2	+
18,7	8	360,0	3	◆
18,1	6	225,0	2	+
14,3	8	360,0	5	◆
24,4	4	146,7	1	●
22,8	4	140,8	1	●
19,2	6	167,6	2	+
17,8	6	167,6	2	+
16,4	8	275,8	3	◆
17,3	8	275,8	3	◆
15,2	8	275,8	3	◆
10,4	8	472,0	4	◆
10,4	8	460,0	4	◆
14,7	8	440,0	4	◆
32,4	4	78,7	1	●
30,4	4	75,7	1	●
33,9	4	71,1	1	●



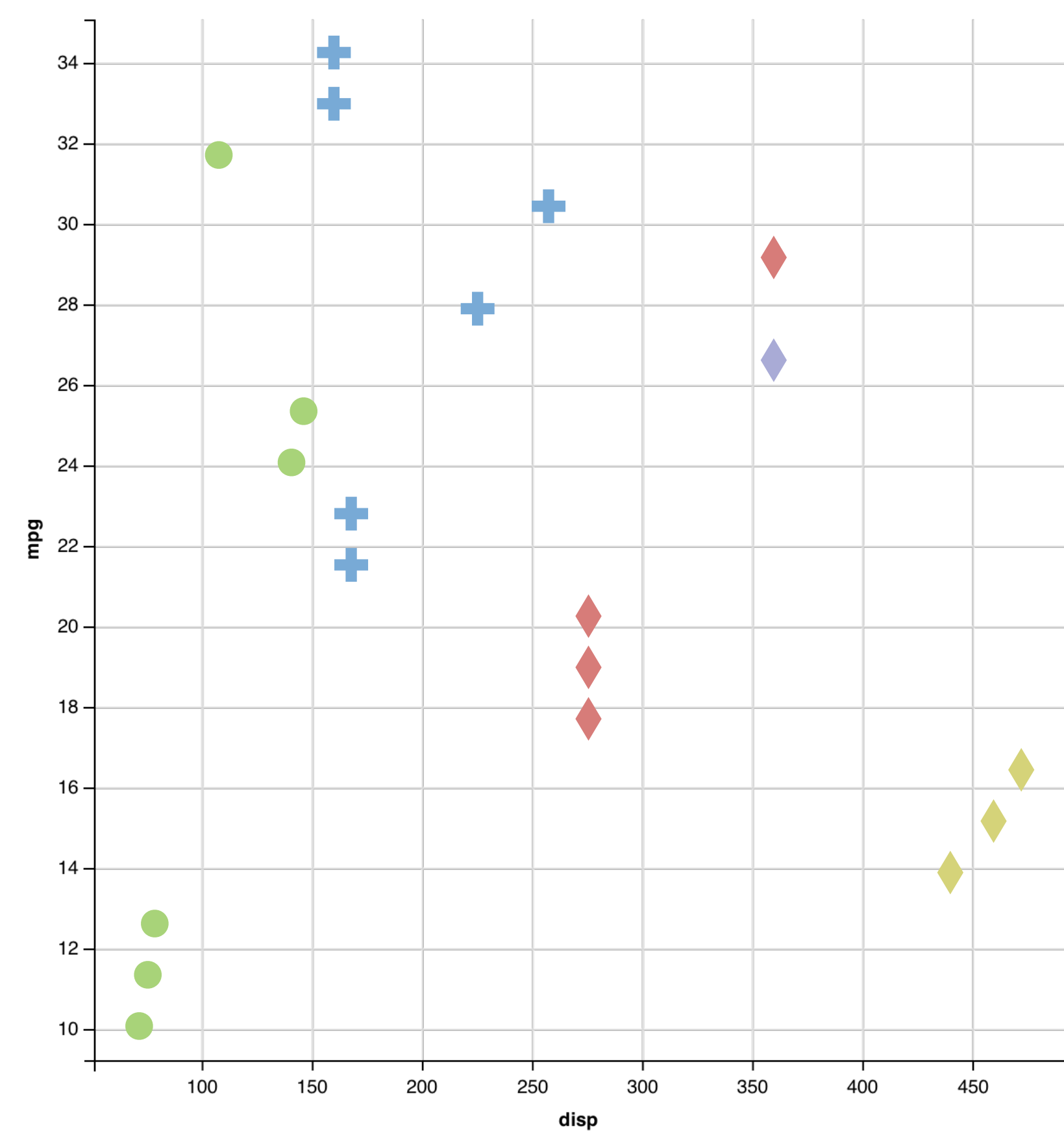
data

geom

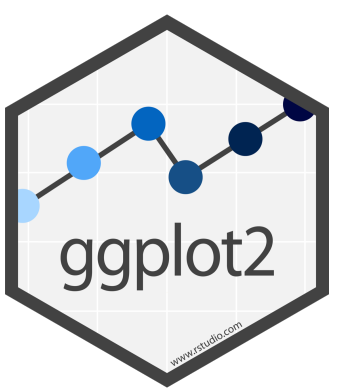


mappings

y	shape	x	fill
mpg	cyl	disp	hp
21,0	6	160,0	2
21,0	6	160,0	2
22,8	4	108,0	1
21,4	6	258,0	2
18,7	8	360,0	3
18,1	6	225,0	2
14,3	8	360,0	5
24,4	4	146,7	1
22,8	4	140,8	1
19,2	6	167,6	2
17,8	6	167,6	2
16,4	8	275,8	3
17,3	8	275,8	3
15,2	8	275,8	3
10,4	8	472,0	4
10,4	8	460,0	4
14,7	8	440,0	4
32,4	4	78,7	1
30,4	4	75,7	1
33,9	4	71,1	1

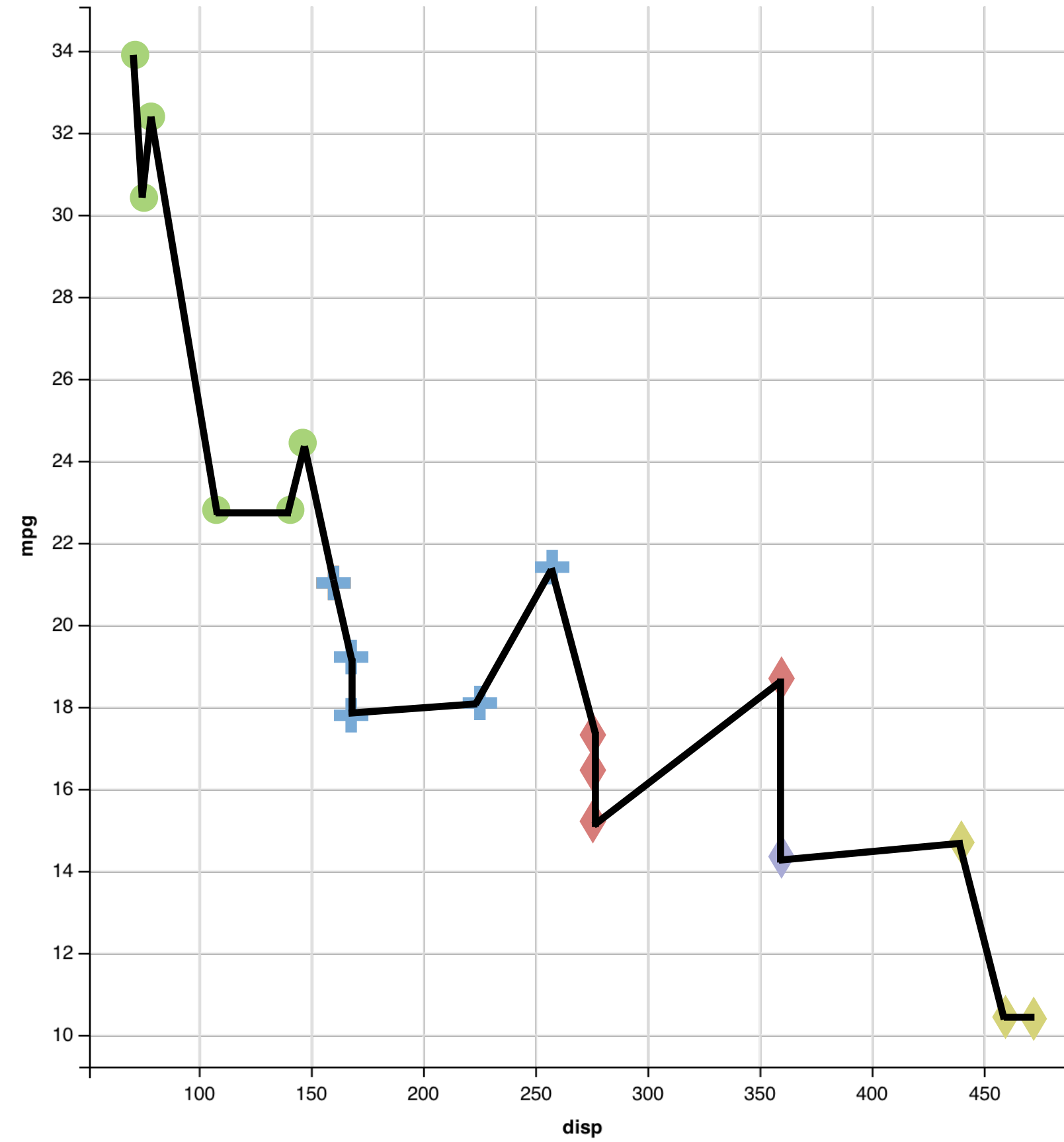


data geom



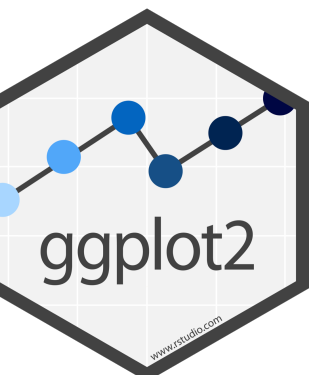
mappings

↑↑ y	↑↑ shape	↑↑ x	↑↑ fill	
mpg	cyl	disp	hp	
21,0	6	160,0	2	—
21,0	6	160,0	2	—
22,8	4	108,0	1	—
21,4	6	258,0	2	—
18,7	8	360,0	3	—
18,1	6	225,0	2	—
14,3	8	360,0	5	—
24,4	4	146,7	1	—
22,8	4	140,8	1	—
19,2	6	167,6	2	—
17,8	6	167,6	2	—
16,4	8	275,8	3	—
17,3	8	275,8	3	—
15,2	8	275,8	3	—
10,4	8	472,0	4	—
10,4	8	460,0	4	—
14,7	8	440,0	4	—
32,4	4	78,7	1	—
30,4	4	75,7	1	—
33,9	4	71,1	1	—


























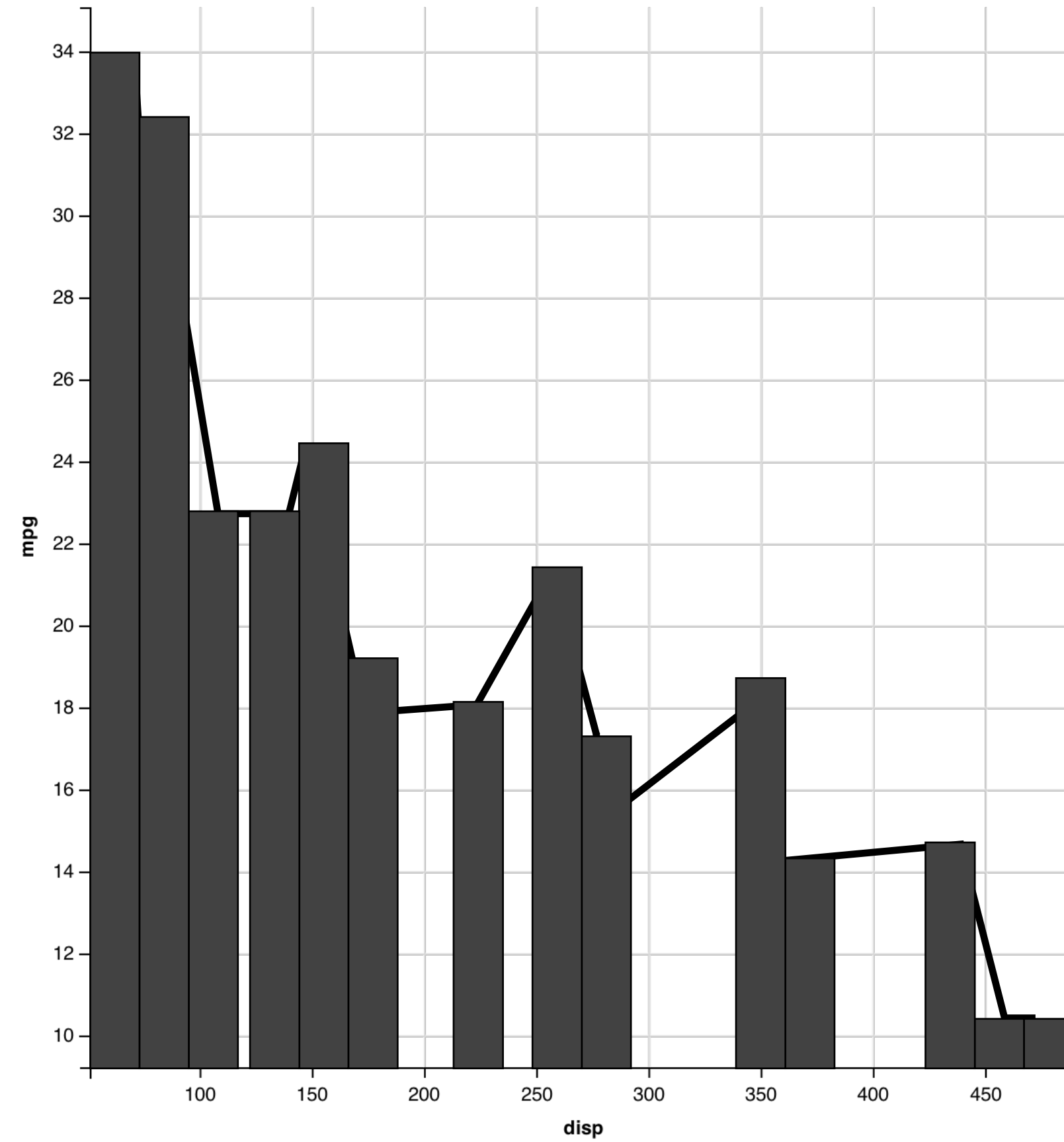
data

geom
points
lines



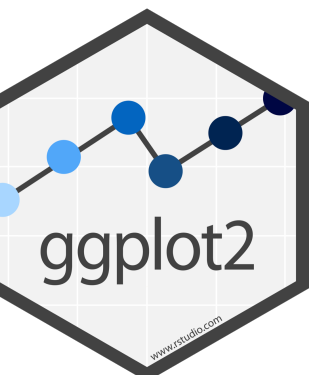
mappings

 mpg	cyl	 disp	hp	
21,0	6	160,0	2	
21,0	6	160,0	2	
22,8	4	108,0	1	
21,4	6	258,0	2	
18,7	8	360,0	3	
18,1	6	225,0	2	
14,3	8	360,0	5	
24,4	4	146,7	1	
22,8	4	140,8	1	
19,2	6	167,6	2	
17,8	6	167,6	2	
16,4	8	275,8	3	
17,3	8	275,8	3	
15,2	8	275,8	3	
10,4	8	472,0	4	
10,4	8	460,0	4	
14,7	8	440,0	4	
32,4	4	78,7	1	
30,4	4	75,7	1	
33,9	4	71,1	1	



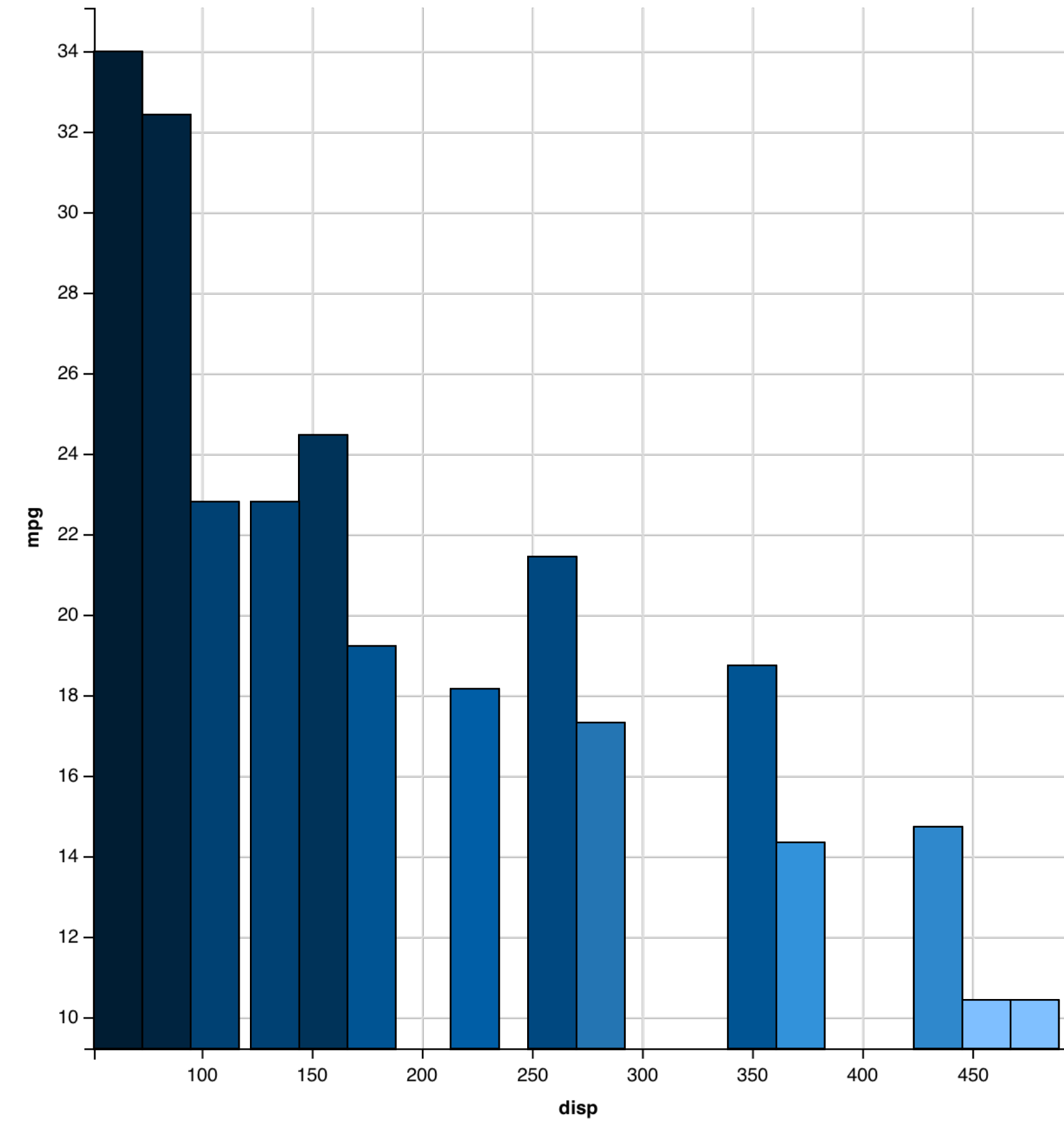
data

geom
points
lines
bars



mappings

mpg	cyl	disp	hp
21,0	6	160,0	2
21,0	6	160,0	2
22,8	4	108,0	1
21,4	6	258,0	2
18,7	8	360,0	3
18,1	6	225,0	2
14,3	8	360,0	5
24,4	4	146,7	1
22,8	4	140,8	1
19,2	6	167,6	2
17,8	6	167,6	2
16,4	8	275,8	3
17,3	8	275,8	3
15,2	8	275,8	3
10,4	8	472,0	4
10,4	8	460,0	4
14,7	8	440,0	4
32,4	4	78,7	1
30,4	4	75,7	1
33,9	4	71,1	1



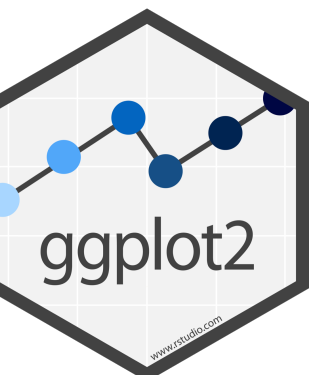
data

geom

points

lines

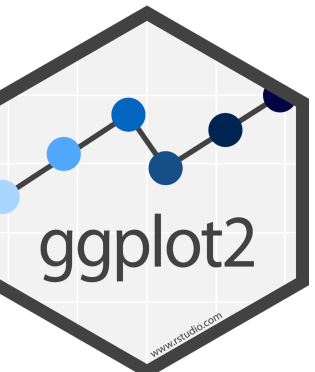
bars



To make a graph

[template]

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```



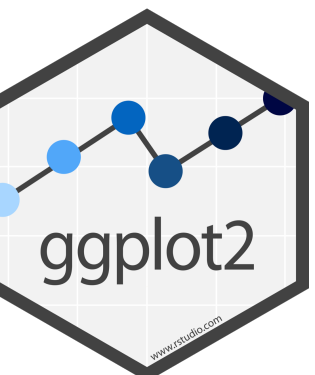
To make a graph

1. Pick a **data** set

mpg	cyl	disp	hp
21,0	6	160,0	2
21,0	6	160,0	2
22,8	4	108,0	1
21,4	6	258,0	2
18,7	8	360,0	3
18,1	6	225,0	2
14,3	8	360,0	5
24,4	4	146,7	1
22,8	4	140,8	1
19,2	6	167,6	2
17,8	6	167,6	2
16,4	8	275,8	3
17,3	8	275,8	3
15,2	8	275,8	3
10,4	8	472,0	4
10,4	8	460,0	4
14,7	8	440,0	4
32,4	4	78,7	1
30,4	4	75,7	1
33,9	4	71,1	1

data

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```



To make a graph

mpg	cyl	disp	hp
21,0	6	160,0	2
21,0	6	160,0	2
22,8	4	108,0	1
21,4	6	258,0	2
18,7	8	360,0	3
18,1	6	225,0	2
14,3	8	360,0	5
24,4	4	146,7	1
22,8	4	140,8	1
19,2	6	167,6	2
17,8	6	167,6	2
16,4	8	275,8	3
17,3	8	275,8	3
15,2	8	275,8	3
10,4	8	472,0	4
10,4	8	460,0	4
14,7	8	440,0	4
32,4	4	78,7	1
30,4	4	75,7	1
33,9	4	71,1	1

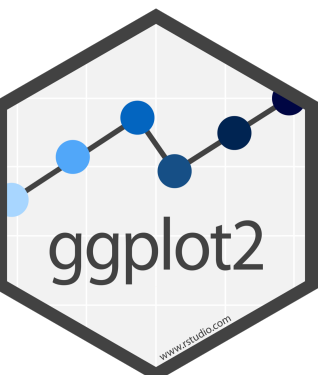
data

geom

1. Pick a **data** set

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

2. Choose a **geom** to display cases



To make a graph

mappings

mpg	cyl	disp	hp	fill
21,0	6	160,0	2	●
21,0	6	160,0	2	●
22,8	4	108,0	1	●
21,4	6	258,0	2	●
18,7	8	360,0	3	●
18,1	6	225,0	2	●
14,3	8	360,0	5	●
24,4	4	146,7	1	●
22,8	4	140,8	1	●
19,2	6	167,6	2	●
17,8	6	167,6	2	●
16,4	8	275,8	3	●
17,3	8	275,8	3	●
15,2	8	275,8	3	●
10,4	8	472,0	4	●
10,4	8	460,0	4	●
14,7	8	440,0	4	●
32,4	4	78,7	1	●
30,4	4	75,7	1	●
33,9	4	71,1	1	●

data

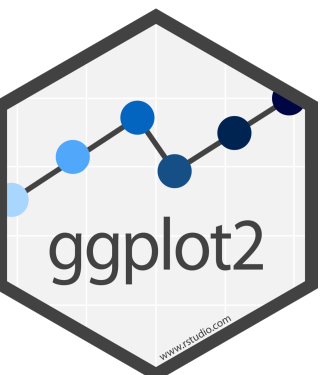
geom

1. Pick a **data** set

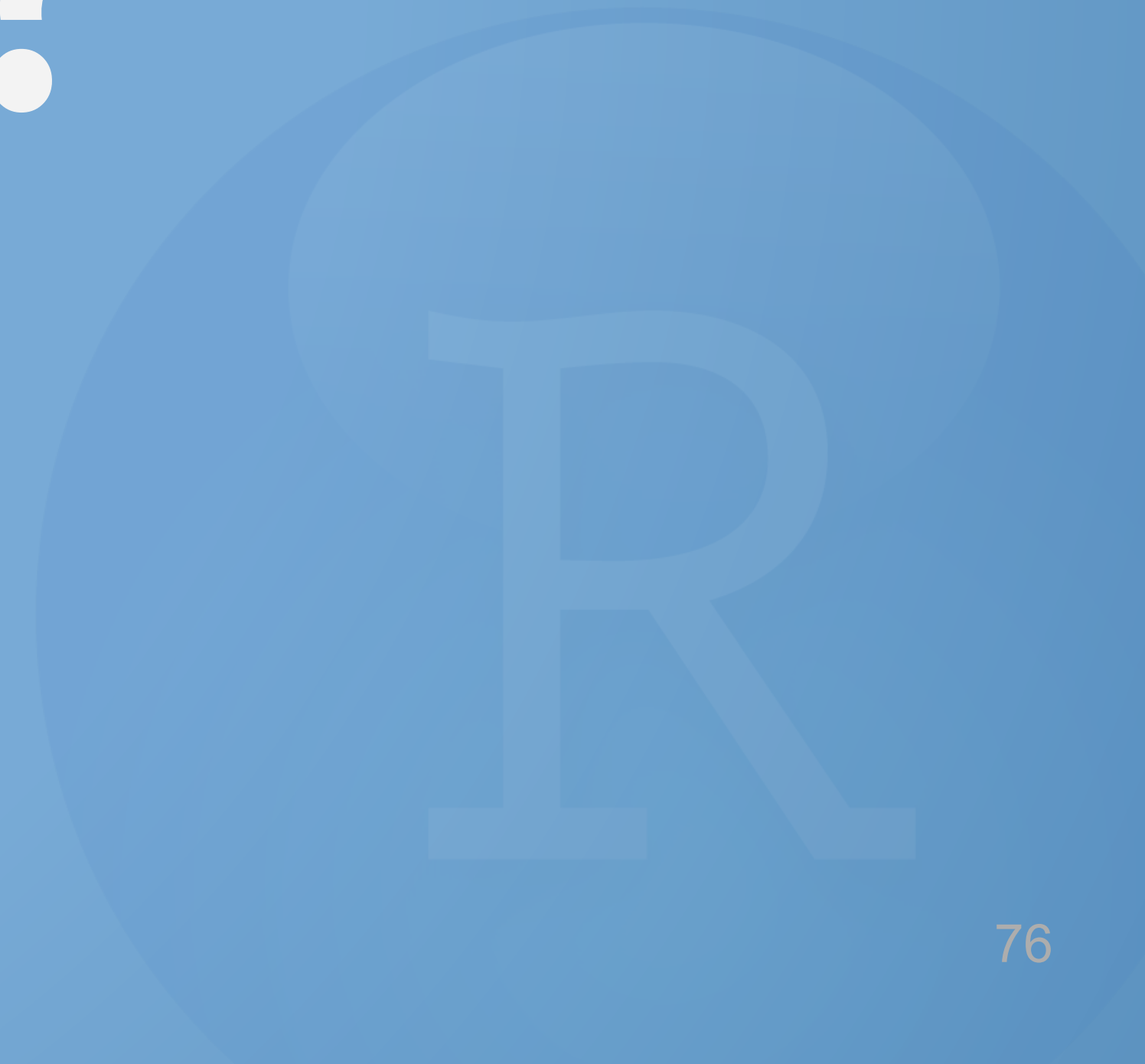
```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

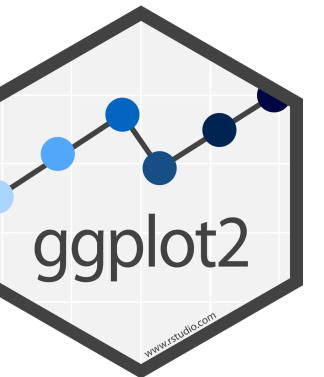
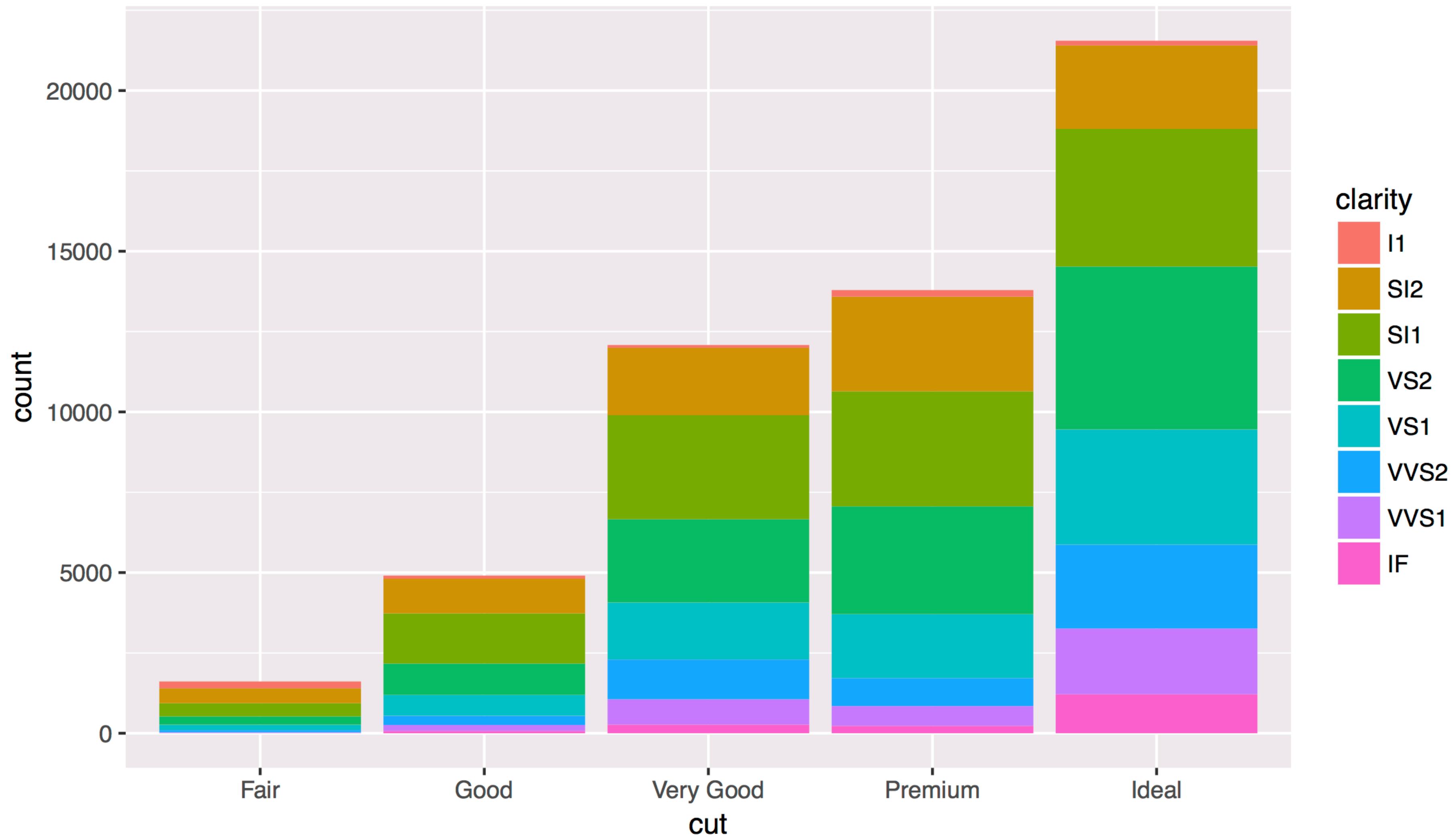
2. Choose a **geom**
to display cases

3. **Map** aesthetic
properties to
variables



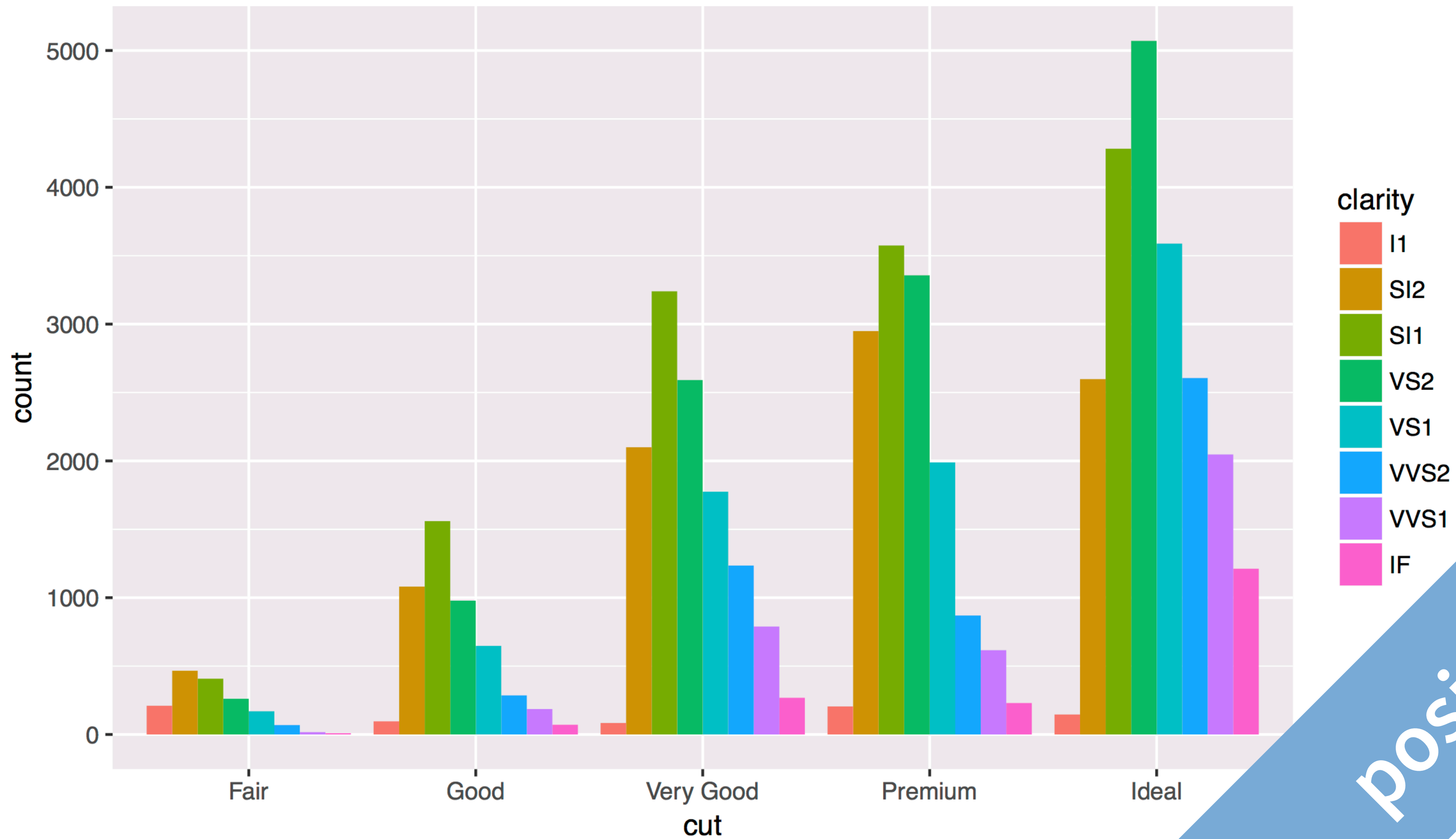
What else?



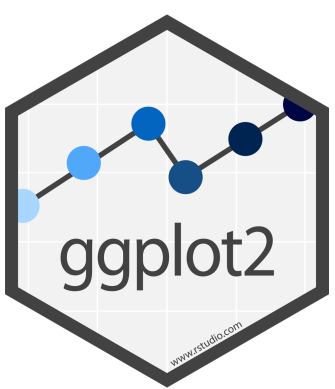


Position Adjustments

How overlapping objects are arranged

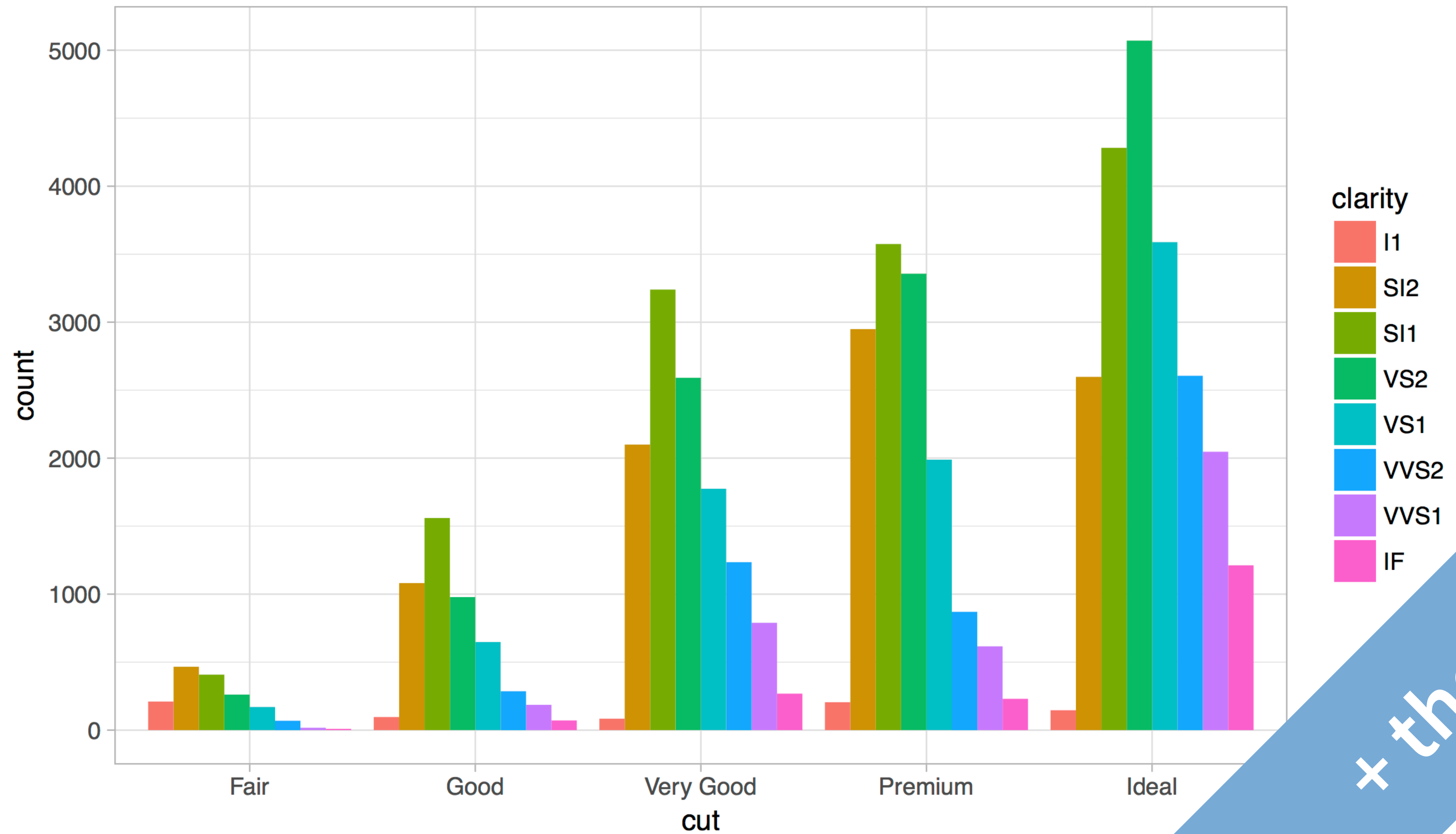


position_*()

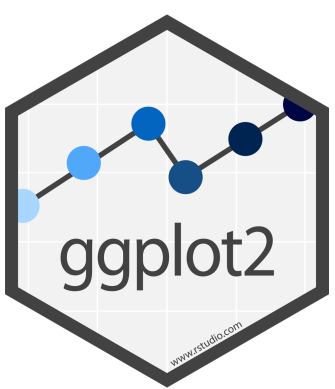


Themes

Visual appearance of non-data elements

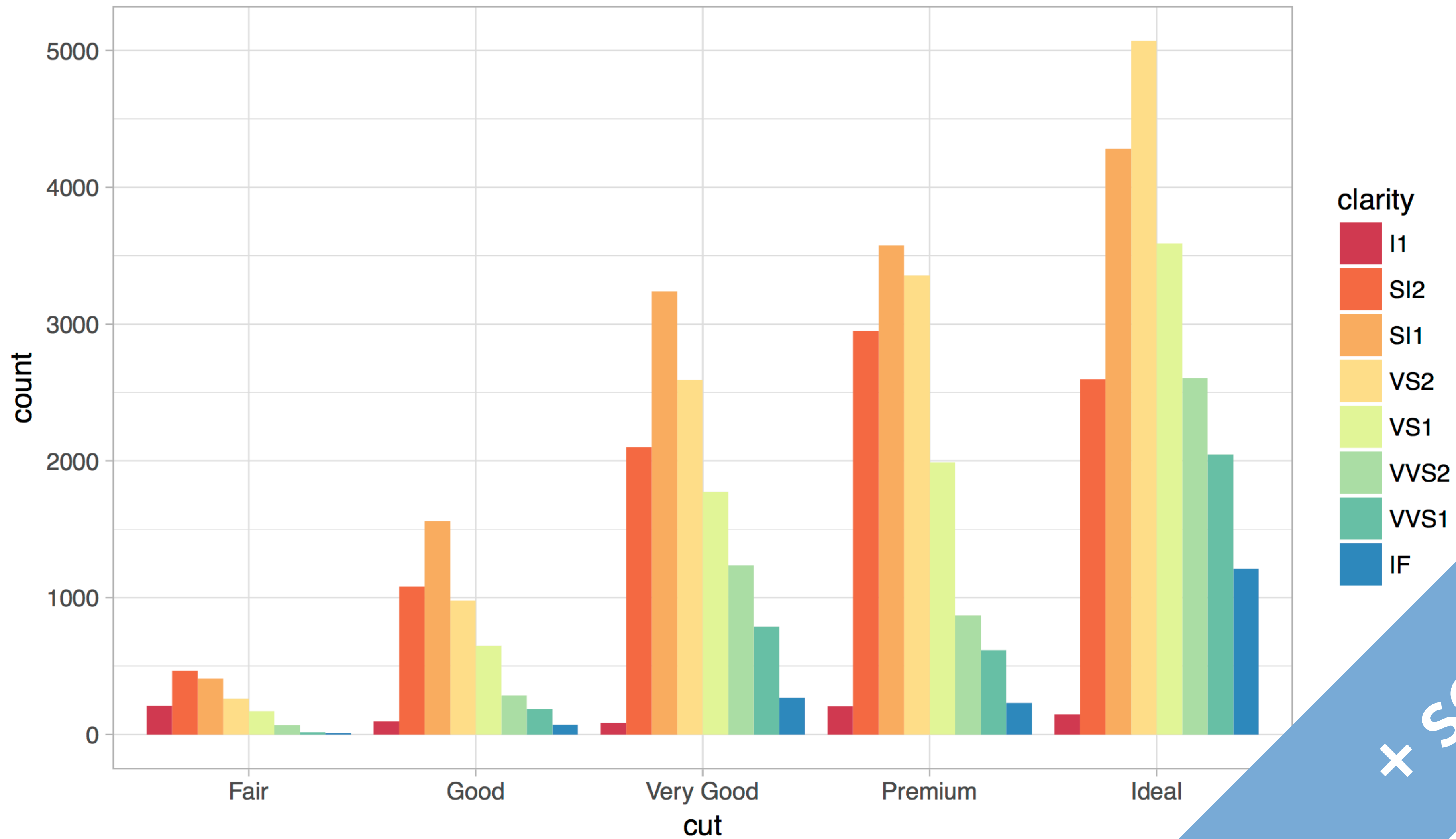


+ theme_*()



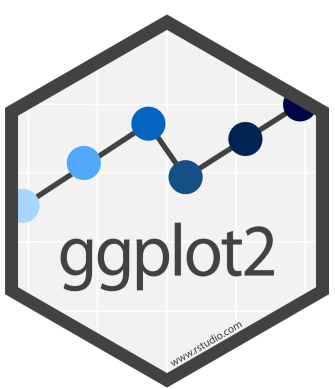
Scales

Customize color scales, other mappings



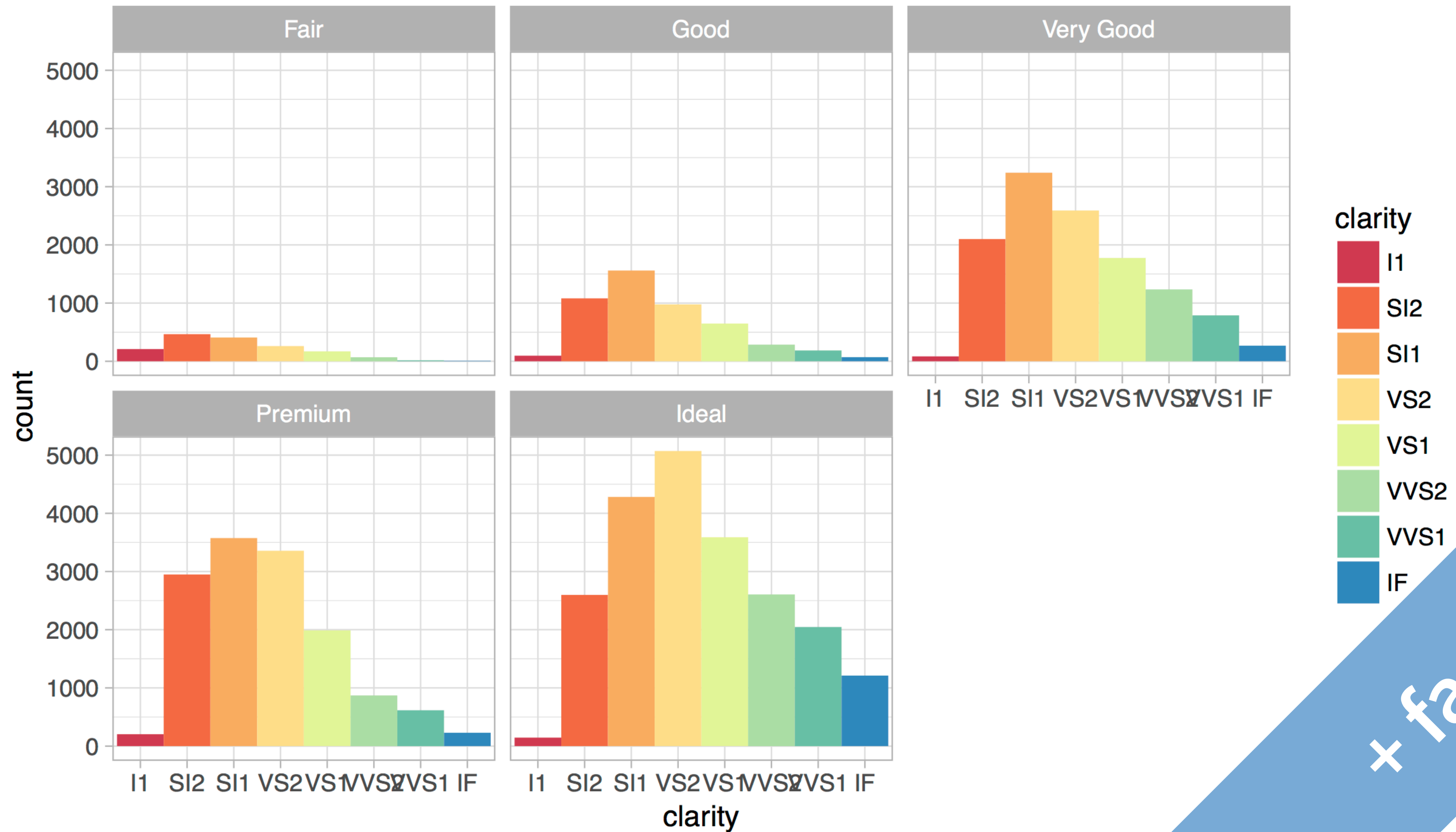
clarity
I1
SI2
SI1
VS2
VS1
VVS2
VVS1
IF

`+ scale_*()`

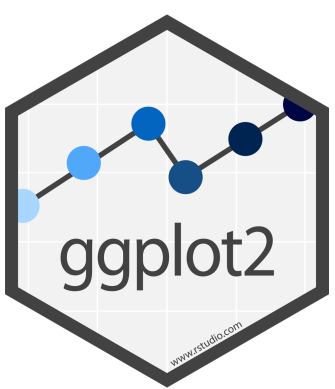


Facets

Subplots that display subsets of the data.



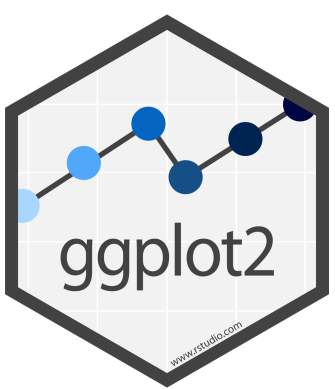
+ facet_*()



Coordinate systems



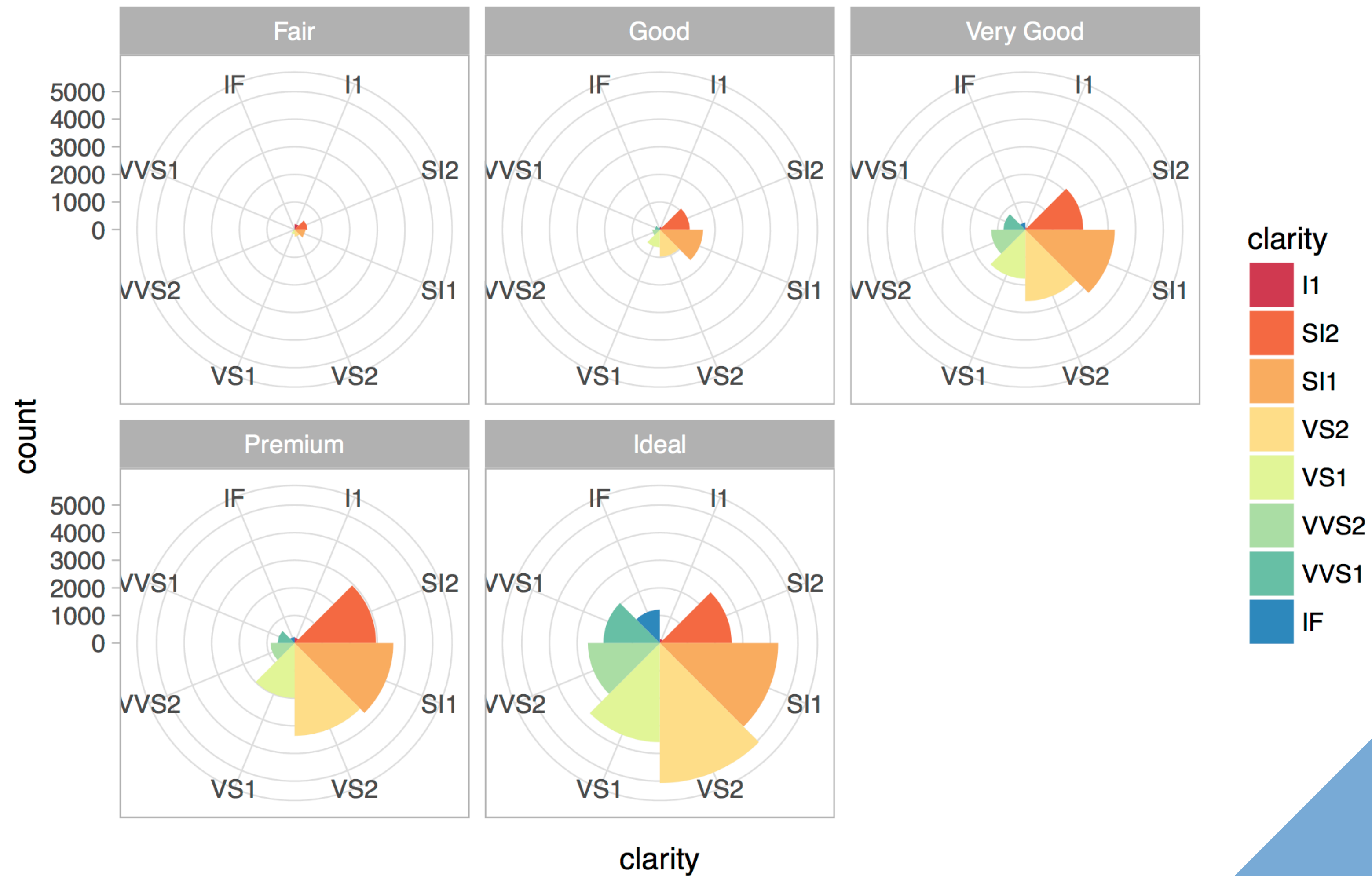
+ coord_*()



Titles and captions

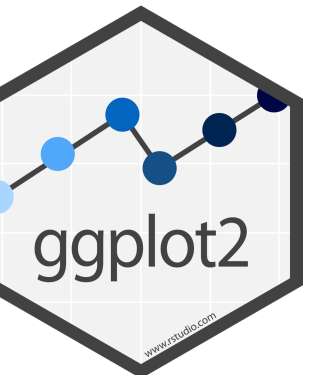
Diamonds data

The data set is skewed towards ideal cut diamonds



Data by Hadley Wickham

+ labs()



A ggplot2 template

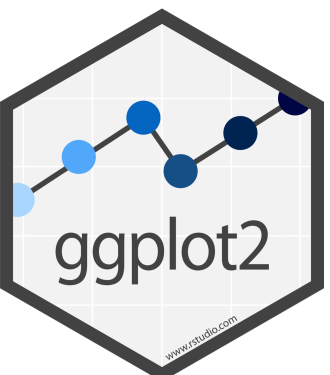
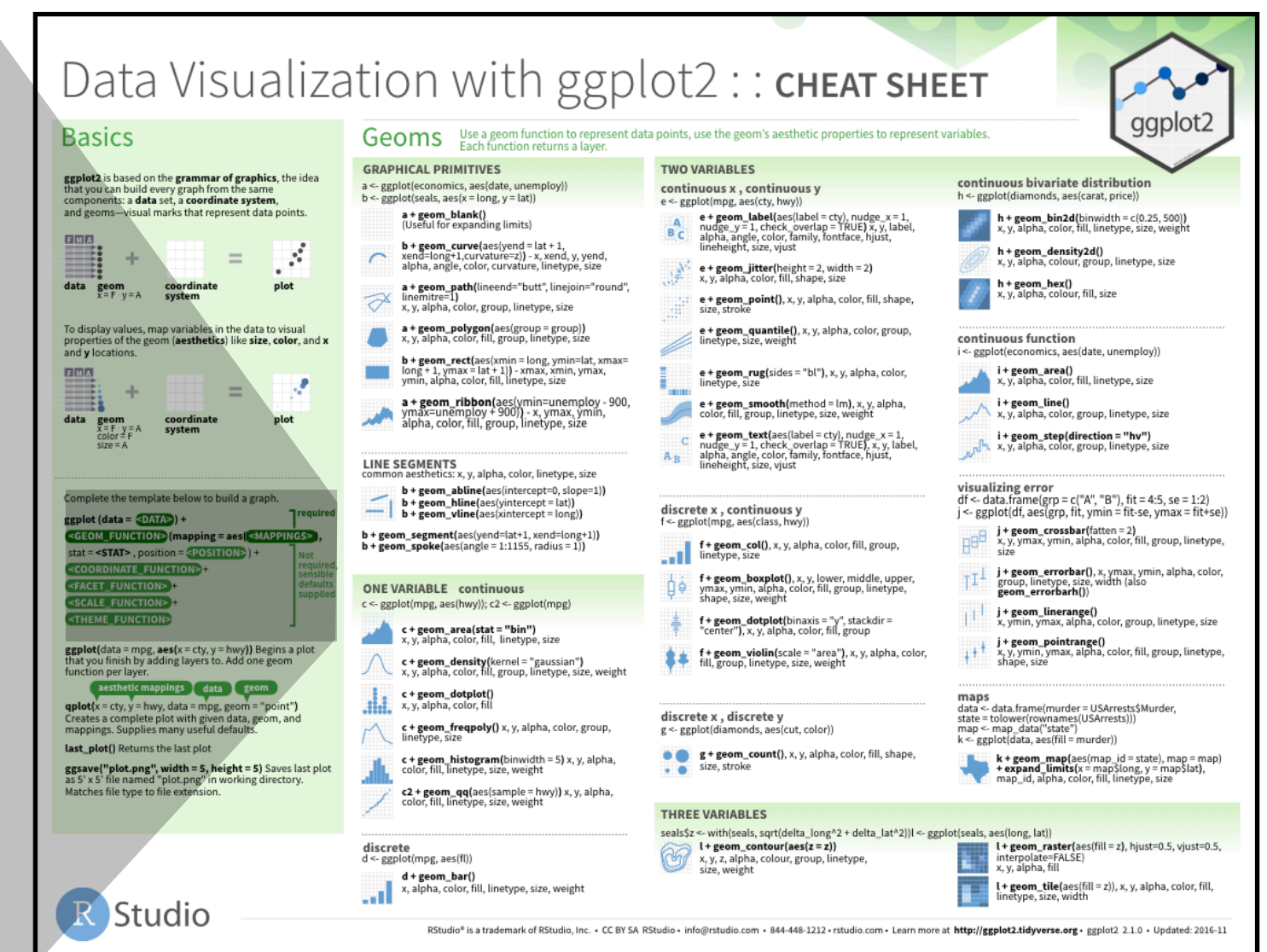
Make any plot by filling in the parameters of this template

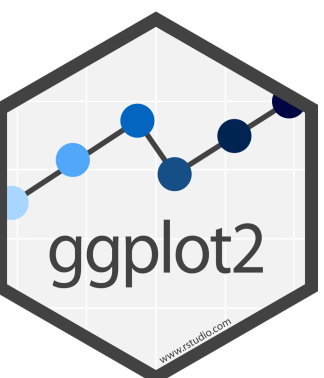
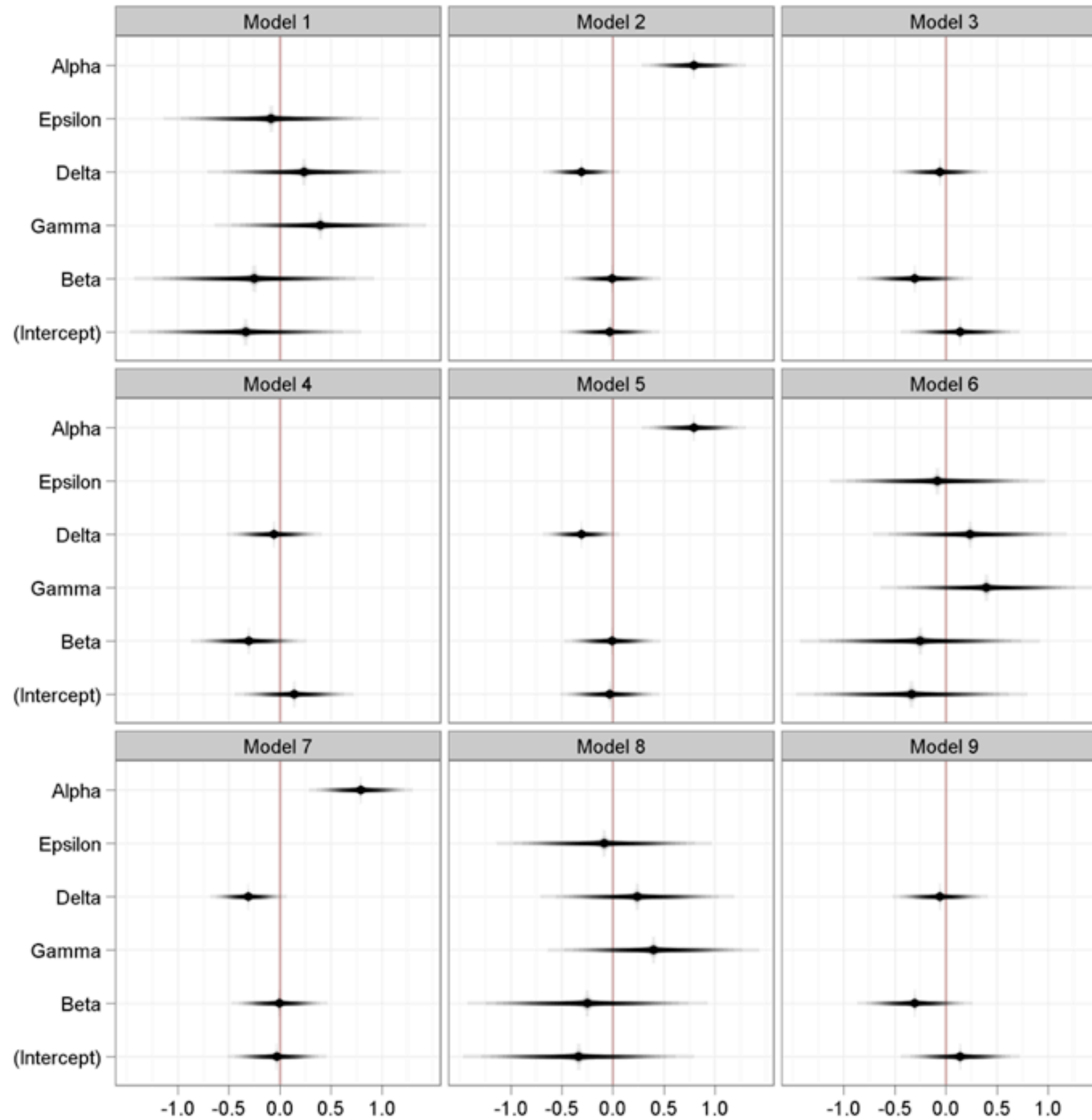
Complete the template below to build a graph.

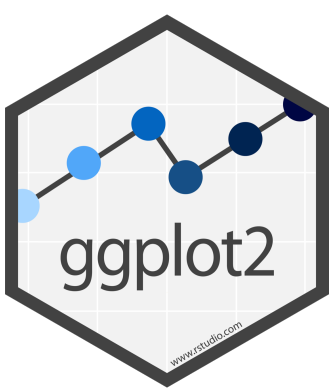
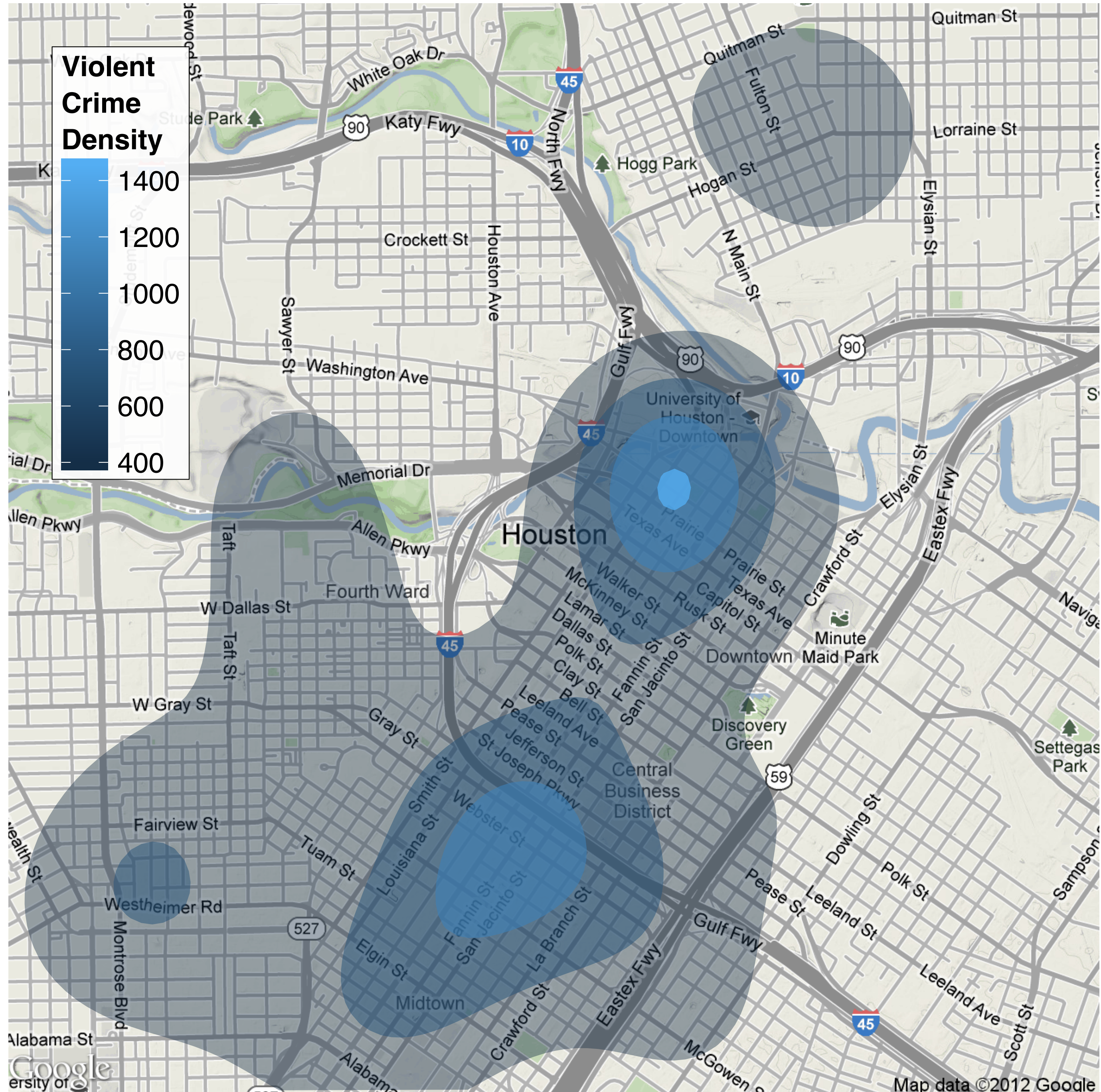
ggplot (**data** = **<DATA>**) +
<GEOM_FUNCTION> (**mapping** = **aes(<MAPPINGS>**),
stat = **<STAT>**, **position** = **<POSITION>**) +
<COORDINATE_FUNCTION> +
<FACET_FUNCTION> +
<SCALE_FUNCTION> +
<THEME_FUNCTION>

required

Not required, sensible defaults supplied

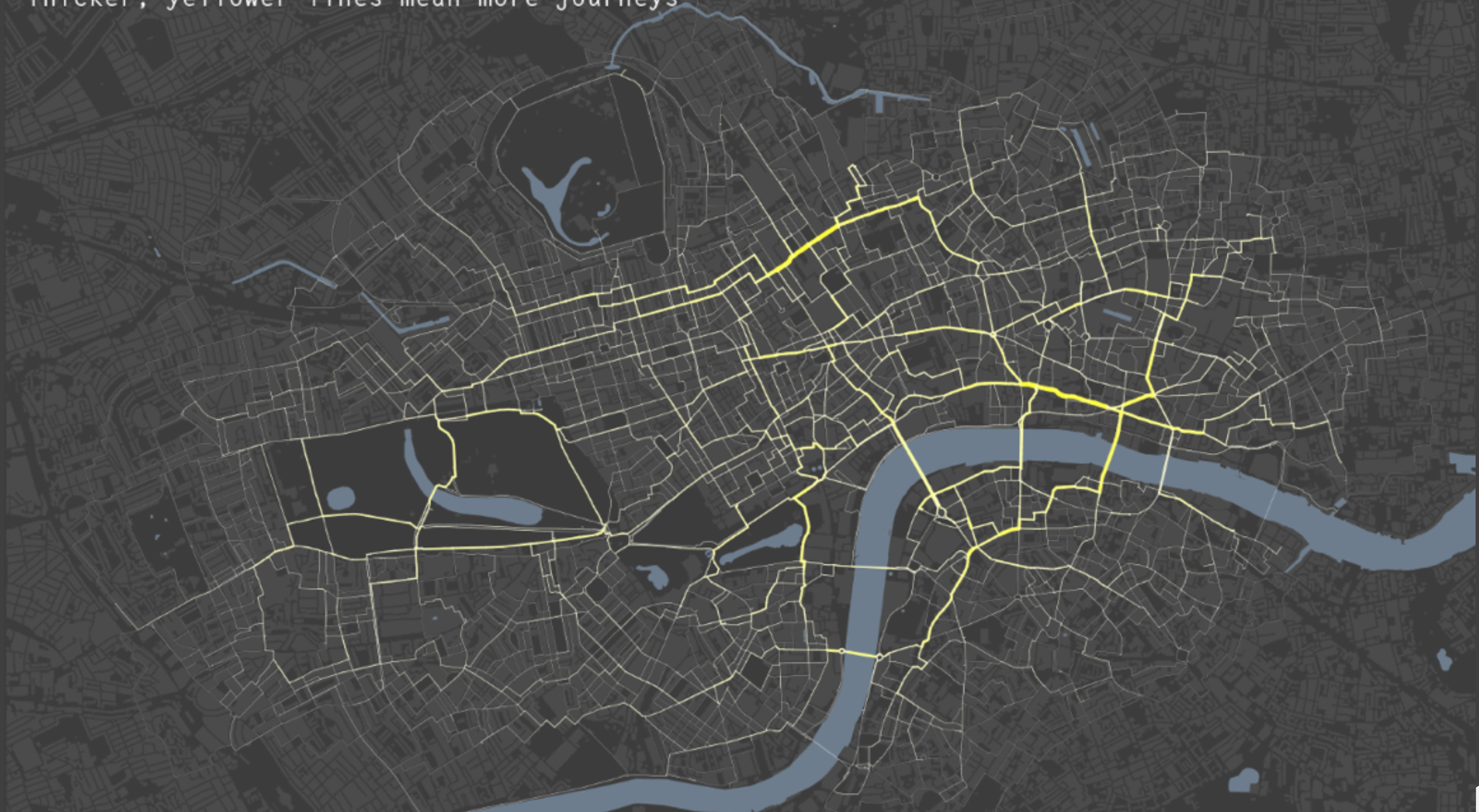






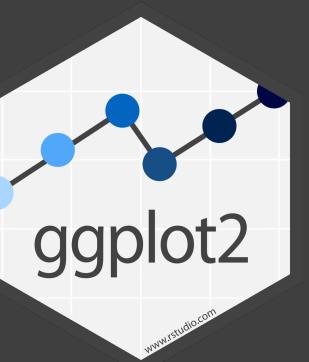
London Cycle Hire Journeys

Thicker, yellower lines mean more journeys



Data: 3.2 Million Journeys (from TfL)
Routing: Ollie O'Brien (@oobr) + OpenStreetMap cc-by-sa
Buildings: OS Opendata Crown Copyright 2011
Map: James Cheshire (@spatialanalysis)

James Cheshire, <http://bit.ly/xqHhAs>



Useful resources

<https://exts.ggplot2.tidyverse.org/gallery/>

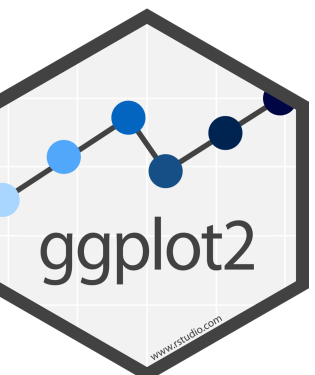
<https://ggforce.data-imaginist.com>

<https://github.com/dkahle/ggmap>

<https://eliocamp.github.io/ggnewscale/>

<https://www.rayshader.com/>

<https://ggplot2-book.org>



<https://r4ds.hadley.nz>

R for Data Science
(2e)  

Welcome

Preface to the second edition

1 Introduction

Whole game ▾

2 Data visualization

3 Workflow: basics

4 Data transformation

5 Workflow: code style

6 Data tidying

7 Workflow: scripts and projects

8 Data import

9 Workflow: getting help

Visualize ▾

10 Layers

11 Exploratory data analysis

12 Communication

Transform ▾

R for Data Science (2e)

Welcome

This is the website for the 2nd edition of “**R for Data Science**”. This book will teach you how to do data science with R: You’ll learn how to get your data into R, get it into the most useful structure, transform it and visualize.

In this book, you will find a practicum of skills for data science. Just as a chemist learns how to clean test tubes and stock a lab, you’ll learn how to clean data and draw plots—and many other things besides.

These are the skills that allow data science to happen, and here you will find the best practices for doing

each of these things with R. You’ll learn how to use the grammar of graphics, literate programming, and reproducible research to save time. You’ll also learn how to manage cognitive resources to facilitate discoveries when wrangling, visualizing, and exploring data.

This website is and will always be free, licensed under the [CC BY-NC-ND 3.0](https://creativecommons.org/licenses/by-nc-nd/3.0/) License. If you’d like a physical copy of the book, you can order it on [Amazon](https://www.amazon.com/). If you appreciate

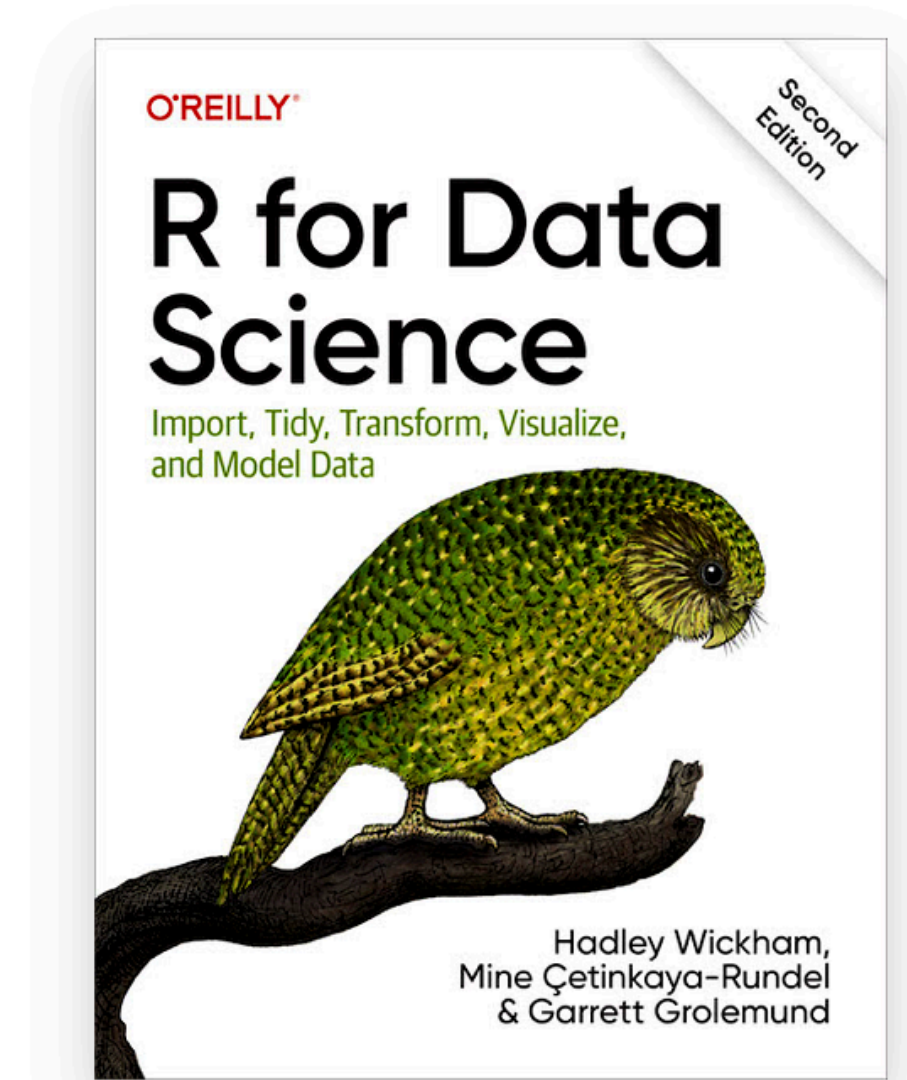



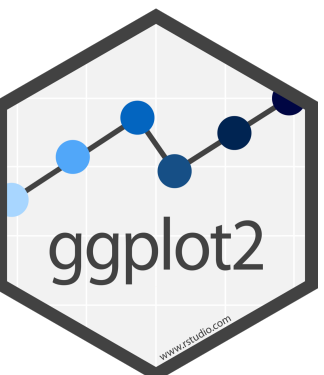
Table of contents

Welcome

Acknowledgements

 Edit this page

Report an issue



Your Turn

Navigate to the main page of the class: **<https://astamm.github.io/data-science-with-r/>**.

Download **03-Transform-Exercises.qmd** from the outline table and open it.

Visualize Data with

