Graphics for Communication with ggplot2

Dr. Austin Brown

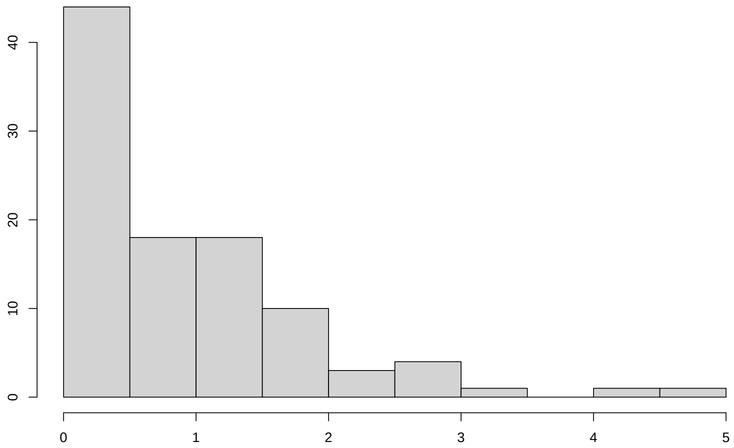
Kennesaw State University

Introduction

- We learned in the last class session about how to use `ggplot2` to create a few different types of graphs.
- As mentioned, being able to describe data visually is an important skill as visuals are often quite effective at describing data to a variety of audiences.
- In this class, we will build upon what we went over last week to show how `ggplot2` can be used to create effective graphs in common data situations.

Labels

- Obviously, it is quite a difficult task for a graph to “stand alone” if there are not descriptive axis titles or a descriptive main title.



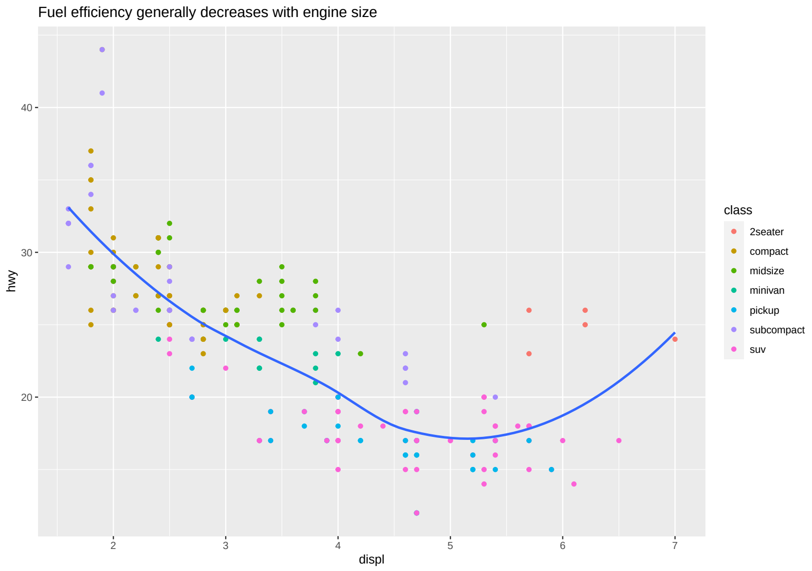
Labels

- We already saw last week how to add axis/main labels using the `labs()` function. Let's review the fuel efficiency data that we were working with last week.

```r
library(tidyverse)
p <- mpg |>
  ggplot(aes(x=displ,y=hwy)) +
  geom_point(aes(color=class)) +
  geom_smooth(se = FALSE) +
  labs(
    title = paste("Fuel efficiency generally decreases",
    "with engine size",sep=" "))
```

Labels

```
p
```


Fuel efficiency generally decreases with engine size
hwy
40
30
20
displ
2
3
4
5
6
7
class
2seater
compact
midsize
minivan
pickup
subcompact
suv

Labels

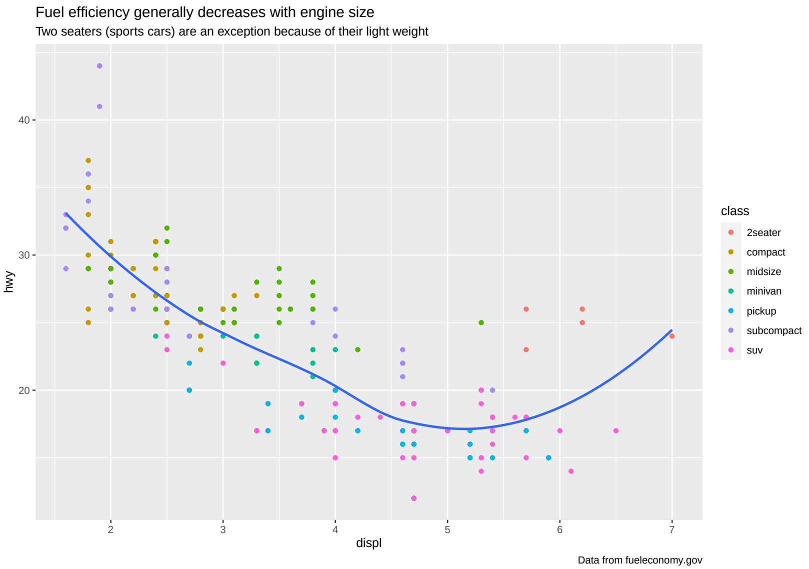
- A good title is pretty useful for communicating the main finding or takeaway from the graph.

- Occassionally, we may need to add more labels to provide additional information. For example, maybe a subtitle with another insight or a footnote with the data source.

```r
p <- mpg |>
  ggplot(aes(x=displ,y=hwy)) +
  geom_point(aes(color=class)) +
  geom_smooth(se = FALSE) +
  labs(
    title = paste("Fuel efficiency generally decreases",
    "with engine size",sep=" "),
    subtitle = paste("Two seaters (sports cars) are an",
    "exception because of their light weight",sep=" "),
    caption = "Data from fueleconomy.gov")
```

Labels

```
p
```


Fuel efficiency generally decreases with engine size
Two seaters (sports cars) are an exception because of their light weight
hwy
40
30
20
displ
2
3
4
5
6
7
class
2seater
compact
midsize
minivan
pickup
subcompact
suv
Data from fueleconomy.gov

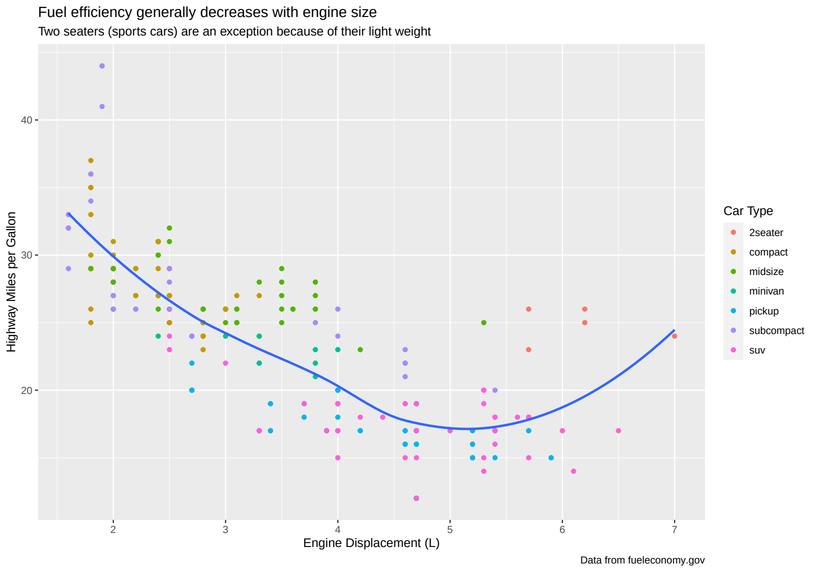
Labels

- As we learned last week, we can also modify the names of the axis titles and legend titles using the `labs()` function as well:

```
p <-
  p + labs(x = "Engine Displacement (L)",
           y="Highway Miles per Gallon",
           color = "Car Type")
```

Labels

```
p
```



Fuel efficiency generally decreases with engine size
Two seaters (sports cars) are an exception because of their light weight
Highway Miles per Gallon
40
30
20
2
3
4
5
6
7
Engine Displacement (L)
Car Type
2seater
compact
midsize
minivan
pickup
subcompact
suv
Data from fueleconomy.gov

Labels

- Sometimes, we need to add a mathematical symbol or equation to our graph in order to describe a graph.
- For example, it is known that if we have sample data that come from a normal distribution, that the mean/expected value of the sample standard deviation, s, is biased.

$$E[s] = \sigma \sqrt{\frac{2}{n-1}} \frac{\Gamma(n/2)}{\Gamma((n-1)/2)}$$

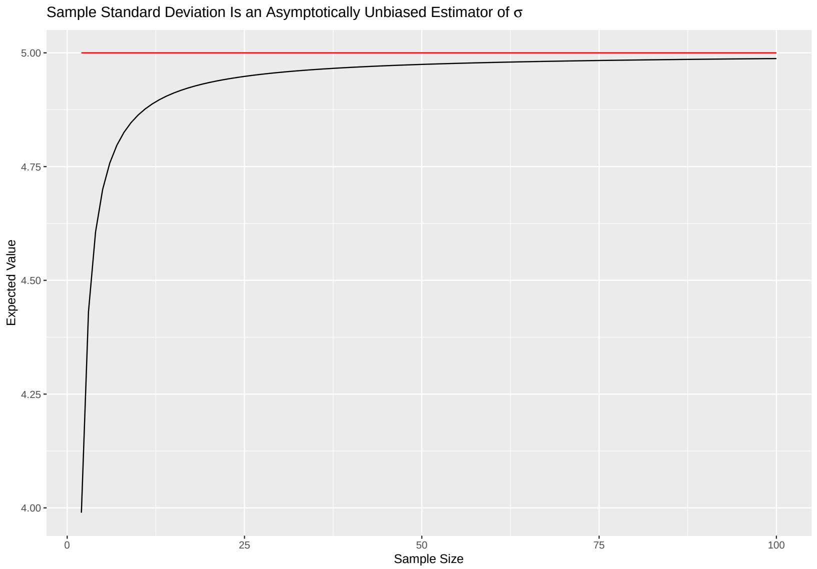
Labels

- For a large enough sample size, the bias goes away. Suppose we wanted to plot a function showing this phenomenon:

```r
sigma <- 5
n <- seq(2,100,by=1)
ev <- 5*sqrt(2/(n-1))*gamma(n/2)/gamma((n-1)/2)
p <- ggplot() +
  geom_line(aes(x=n,y=ev),color="black") +
  geom_line(aes(x=n,y=sigma),color="red") +
  labs(x = "Sample Size",
       y = "Expected Value",
       title = expression(
         paste("Sample Standard Deviation ",
               "Is an Asymptotically Unbiased ",
               "Estimator of ",sigma)))
```

Labels

```
p
```



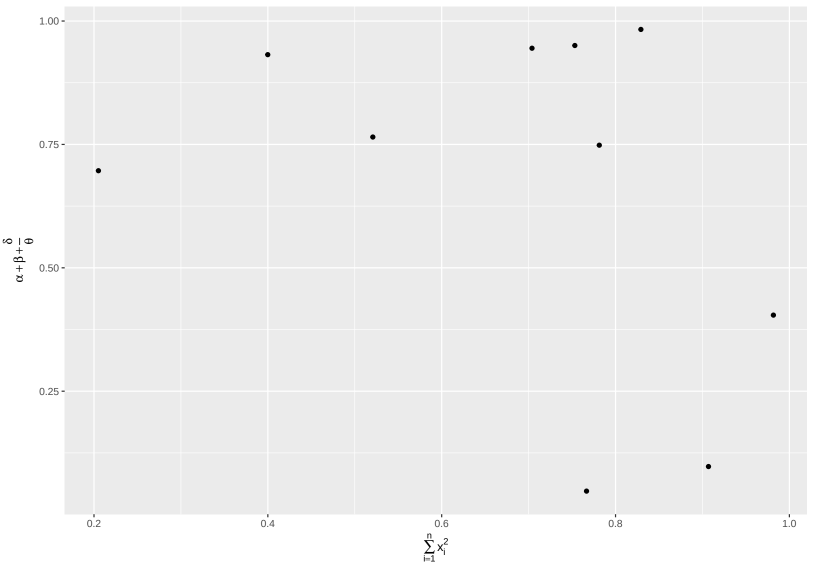
Sample Standard Deviation Is an Asymptotically Unbiased Estimator of σ
Expected Value
5.00
4.75
4.50
4.25
4.00
0
25
50
75
100
Sample Size

Labels

```r
df <- tibble(x = runif(10),
             y = runif(10))
p <- df |> ggplot(aes(x,y)) +
  geom_point() +
  labs(
    x = quote(sum(x[i]^2,i==1,n)),
    y = quote(alpha + beta + frac(delta,theta)))
```

Labels

```
p
```



1.00
0.75
0.50
0.25
0.2
0.4
0.6
0.8
1.0
$\alpha+\beta+\frac{\delta}{\theta}$
$\sum_{i=1}^{n} x_i^2$

Annotations

- Having clear, effective labels for our axes, legends, and plot titles are important.
- There may be instances where we need to label individual observations or groups of observations. In `ggplot`-speak, these are referred to as *annotations*.
- For example, with the `mpg` data, it might be effective to include the name of the most fuel efficient car in each of our car classes.

Annotations

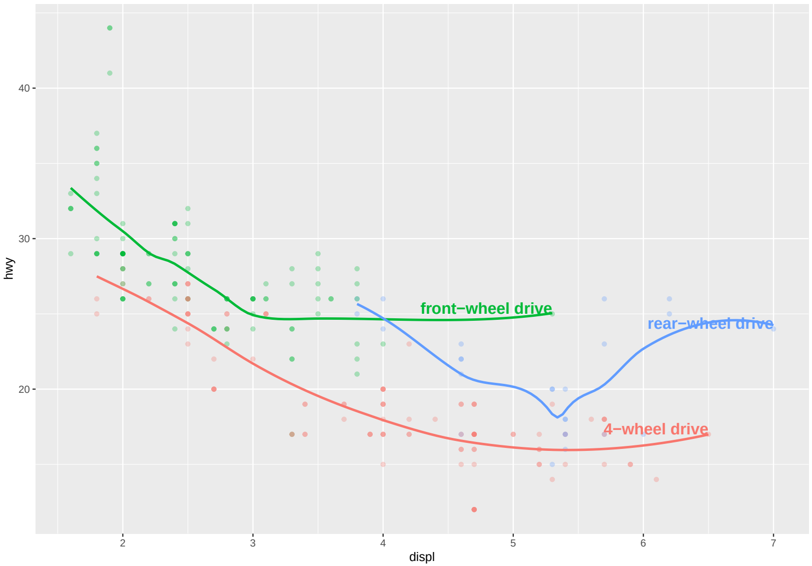
```r
label_info <- mpg |>
  group_by(drv) |>
  arrange(desc(displ)) |>
  slice_head(n = 1) |>
  mutate(
    drive_type = case_when(
      drv == "f" ~ "front-wheel drive",
      drv == "r" ~ "rear-wheel drive",
      drv == "4" ~ "4-wheel drive"
    )
  ) |>
  select(displ, hwy, drv, drive_type)
```

Annotations

```r
p <- mpg |>
  ggplot(aes(x = displ, y = hwy, color = drv)) +
  geom_point(alpha = 0.3) +
  geom_smooth(se = FALSE) +
  geom_text(
    data = label_info,
    aes(x = displ, y = hwy, label = drive_type),
    fontface = "bold", size = 5, hjust = "right", vjust = '
  ) +
  theme(legend.position = "none")
```

Annotations

```
p
```



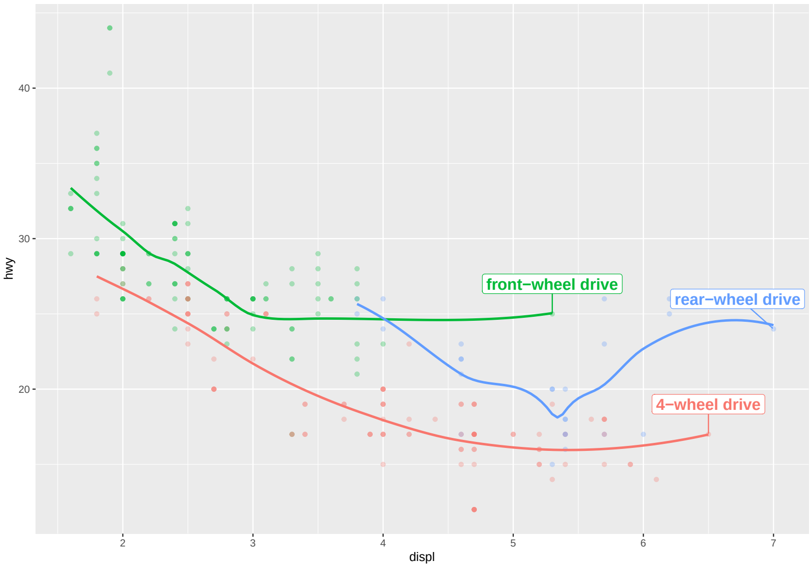
Annotations

- Okay, so this isn't very useful because some of the labels overlap with each other. To get around this, we can make use of functions within the ggrepel package.

```r
library(ggrepel)
p <- mpg |>
  ggplot(aes(x = displ, y = hwy, color = drv)) +
  geom_point(alpha = 0.3) +
  geom_smooth(se = FALSE) +
  geom_label_repel(
    data = label_info,
    aes(x = displ, y = hwy, label = drive_type),
    fontface = "bold", size = 5, nudge_y = 2
  ) +
  theme(legend.position = "none")
```

Annotations

```
p
```



Annotations

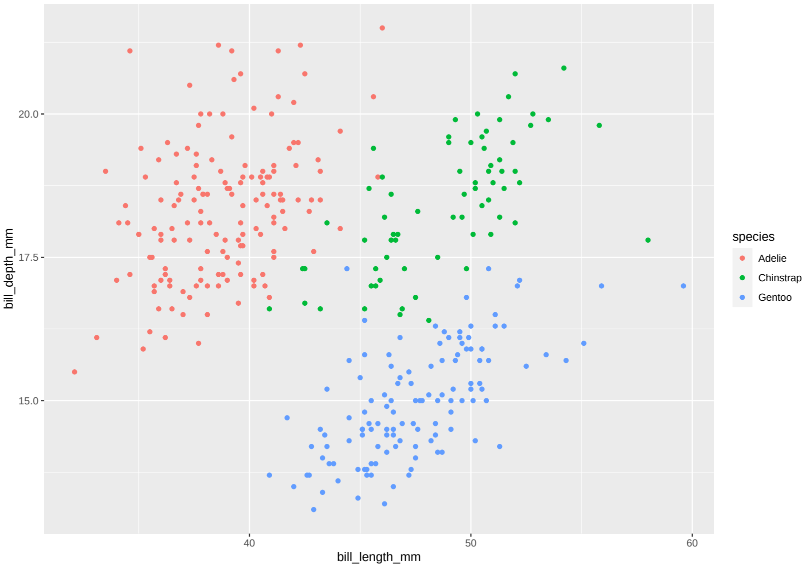
- Sometimes, instead of having a legend, it can be useful and more aesthetically pleasing to have data labels serve the same purpose as a legend.

- Let's look at an example using the `penguins` dataframe. We know we could use:

```r
p <- palmerpenguins::penguins |>
  ggplot(aes(bill_length_mm,bill_depth_mm)) +
  geom_point(aes(color=species))
```

Annotations

```
p
```



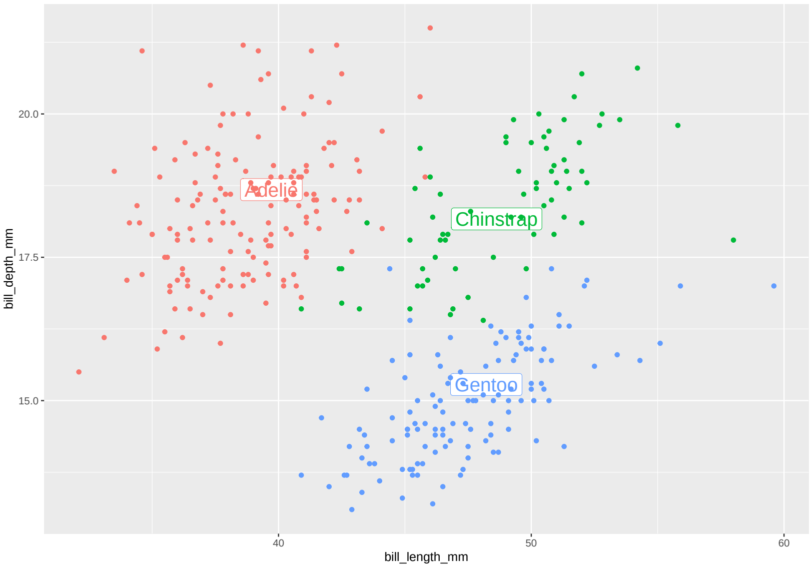
Annotations

```
species_avg <- palmerpenguins::penguins |>
  group_by(species) |>
  summarize(
    bill_length_mm = median(bill_length_mm,na.rm=T),
    bill_depth_mm = median(bill_depth_mm,na.rm=T))

p <- palmerpenguins::penguins |>
  ggplot(aes(bill_length_mm,bill_depth_mm,color=species)) -
  geom_label_repel(aes(label=species),
                          data=species_avg,
                          size = 6, label.size = 0,
                          segment.color = NA
                          ) +
  geom_point() +
  theme(legend.position = "none")
```

Annotations

```
p
```



Annotations

- There are other instances when we want to put a single data label on a graph.
- For example, maybe instead of our plot title explaining the graph, we include a brief explanation on the plot itself.

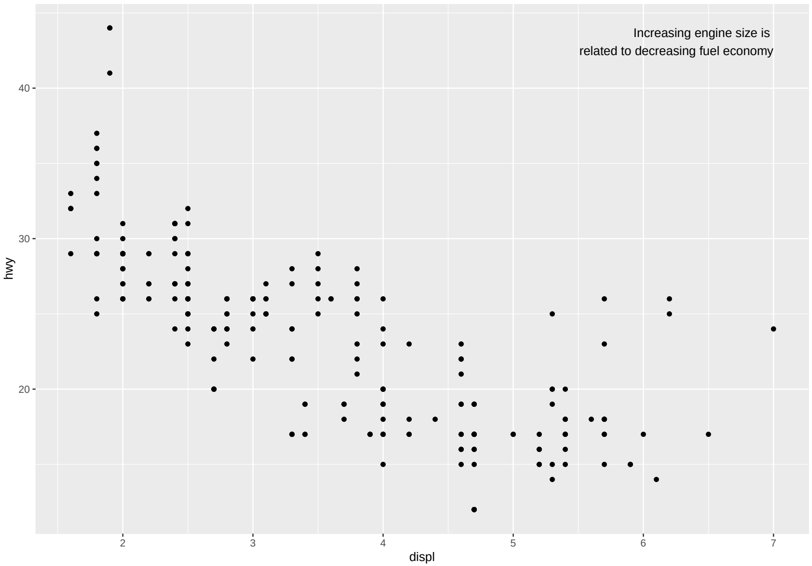
```r
label <- mpg |>
  summarize(
    displ = max(displ),
    hwy = max(hwy),
    label = paste(
      "Increasing engine size is \n related to",
      "decreasing fuel economy"
    )
  )
```

Annotations

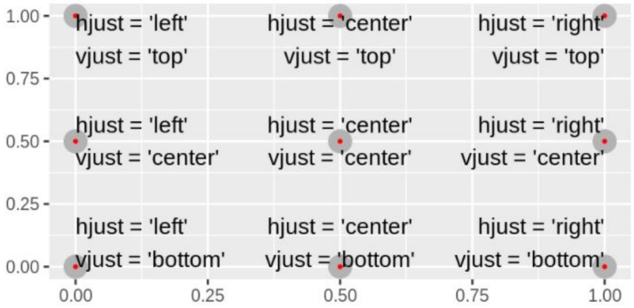
```r
p <- ggplot(mpg,aes(displ,hwy)) +
  geom_point() +
  geom_text(
    aes(label = label),
    data = label,
    vjust = "top",
    hjust = "right"
  )
```

Annotations

```
p
```



Annotations



Scales

- Another tool that can be handy for improving your graph is to modify the scales.

- Scales control things like the tick marks on the `x` and `y` axes. For all geoms, there are default scales that `ggplot` uses which correspond to the type of variable being used on the `x` and `y` axis.

Scales

- So for example, when we plotted out:

```r
p <- mpg |>
  ggplot(aes(displ,hwy)) +
  geom_point(aes(color=class))
```

Scales

- What ggplot is doing behind the scenes is:

```r
p <- mpg |>
  ggplot(aes(displ,hwy)) +
  geom_point(aes(color=class)) +
  scale_x_continuous() +
  scale_y_continuous() +
  scale_color_discrete()
```

Scales

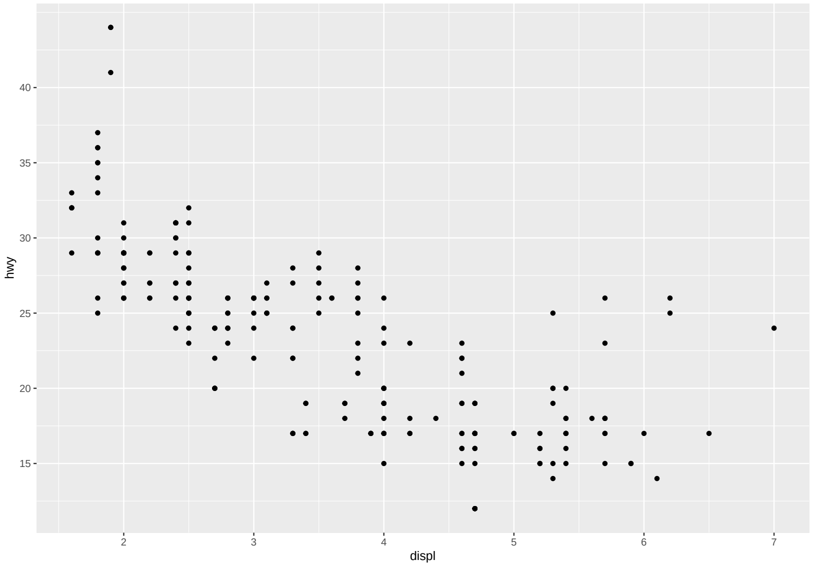
- Obviously, it's a nice feature that we don't have to type all that out every time, but there are times where we need to make changes to improve the quality of our graph.
- For example, there are lots of instances where we want to change our tick marks. Lete's say instead of our `hwy` tick marks being in increments of 10 starting at 20, we want them to be in increments of 5 starting at 15.
- Here, we can make an adjustment to the `scale_y_continuous` function to do just that.

Scales

```r
p <- mpg |>
  ggplot(aes(displ,hwy)) +
  geom_point() +
  scale_y_continuous(breaks = seq(15,40,by=5))
```

Scales

```
p
```



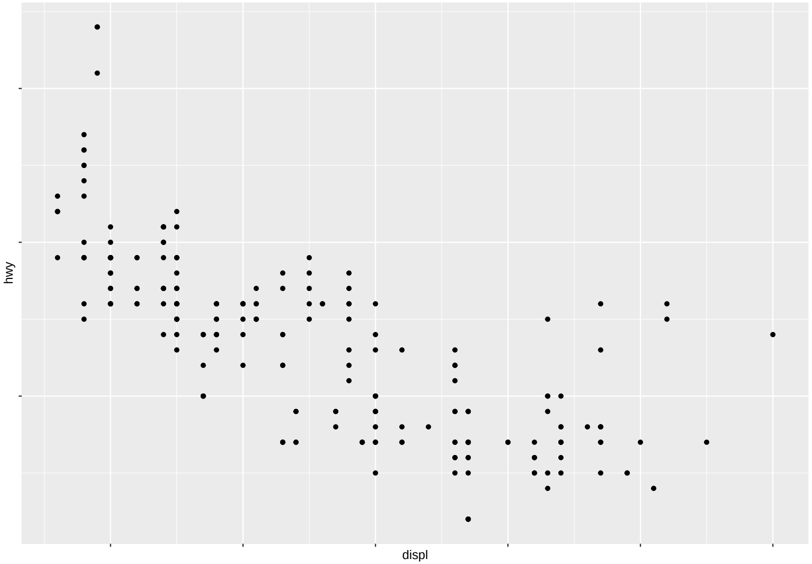
Scales

- There are other instances where we might need to repress the tick mark labels all together. I have found this be useful for plotting maps, but it could also be useful in cases where data is private, but you can still share the plot sans the data.

```r
p <- mpg |>
  ggplot(aes(displ,hwy)) +
  geom_point() +
  scale_x_continuous(labels = NULL) +
  scale_y_continuous(labels = NULL)
```

Scales

```
p
```



Scales

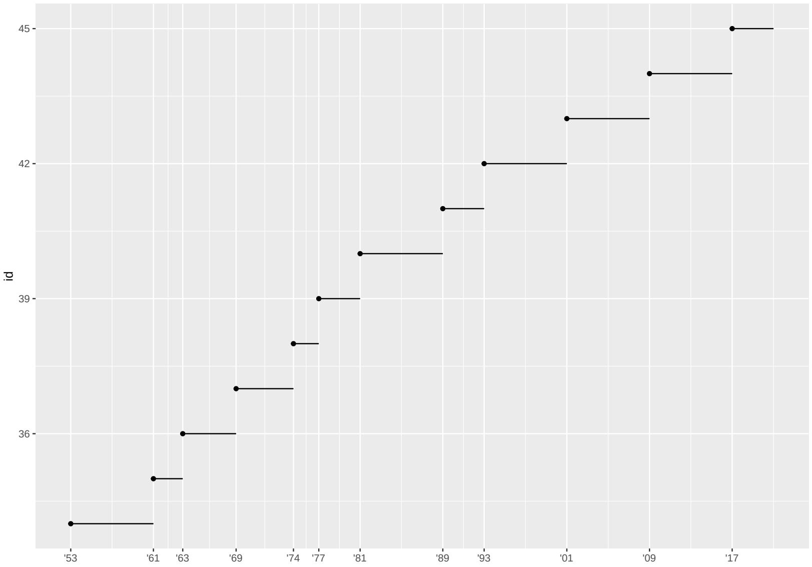
- In most of our examples, and probably in most cases, our tick mark labels are equally spaced. Sometimes, the data is better presented when the tick mark labels are unequally spaced.
- For example, suppose we wanted to show the years spanned by a particular president's time in office. The duration, in years, would be better explained if we had individual tick mark labels for the beginning and ending years of a presidency.

Scales

```r
p <- presidential |>
  mutate(id = 33 + row_number()) |>
  ggplot(aes(start,id)) +
  geom_point() +
  geom_segment(aes(xend = end, yend = id)) +
  scale_x_date(
    NULL,
    breaks = presidential$start,
    date_labels = "'%y"
  )
```

Scales

```
p
```



Scales

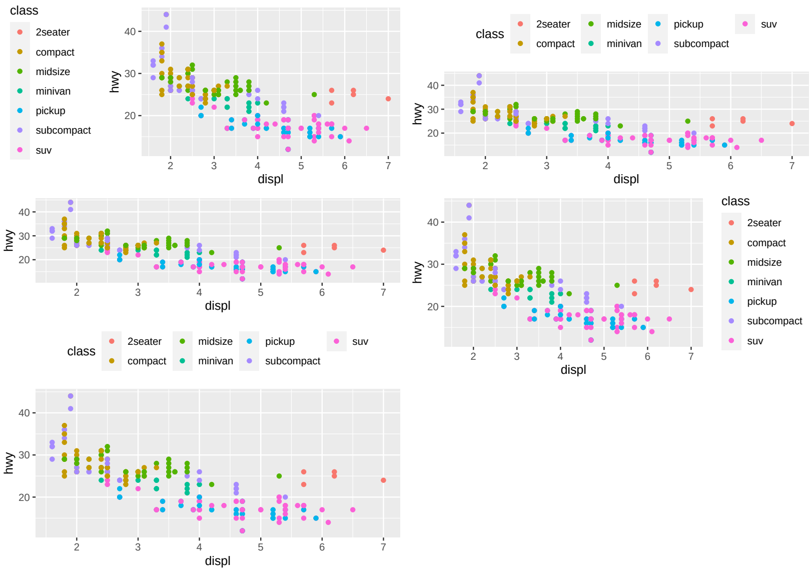
- Another aspect of a plot that we'll likely be in need of modifying at some point is the legend.
- We already learned how to modify its title, but many times, we also need to modify its location on the plot, for whatever reason.
- We can move the legend to the top, bottom, left, or right of the plot in addition to supressing it using the `legend.position` argument inside of the the `theme()` function.

Scales

```r
base <- mpg |>
  ggplot(aes(displ,hwy)) +
  geom_point(aes(color = class))
p1 <- base + theme(legend.position = "left")
p2 <- base + theme(legend.position = "top")
p3 <- base + theme(legend.position = "bottom")
p4 <- base + theme(legend.position = "right")
p5 <- base + theme(legend.position = "none")
```

Scales

```r
gridExtra::grid.arrange(p1,p2,p3,p4,p5,nrow=3)
```



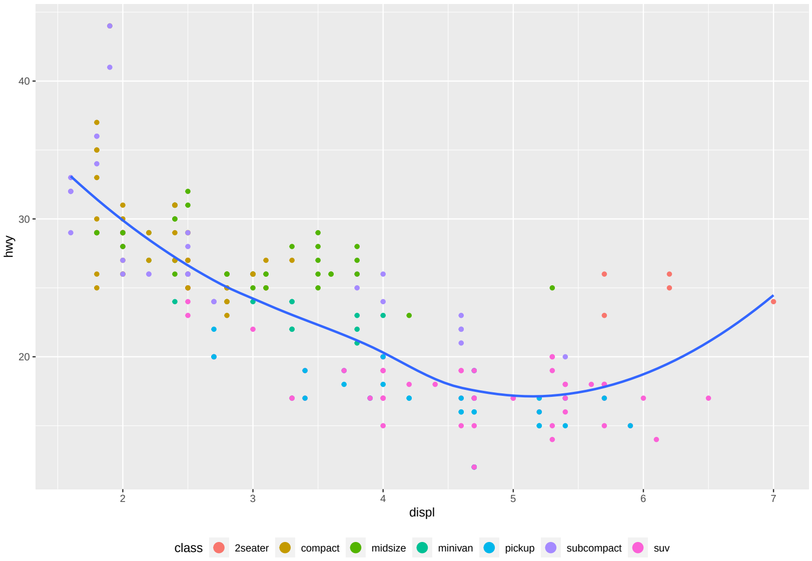
Scales

- We also have the ability to change characteristics of the legend itself using `guides`.
- By default, if we want our legend to be at the bottom of a graph, `ggplot` is breaking up our car types into two rows. But let's say we wanted just one row and we wanted those points to be a bit larger.

```r
p <- base + geom_smooth(se = FALSE) +
  theme(legend.position = "bottom") +
  guides(color = guide_legend(
    nrow = 1, override.aes = list(size = 4)
  ))
```

Scales

```
p
```



40
30
20
hwy
2
3
4
5
6
7
displ
class
2seater
compact
midsize
minivan
pickup
subcompact
suv

Scales

- Another scale that we will commonly want to modify is the color scale. As we've seen, `ggplot` has a default color scale for each geom.

- However, there are a lot of instances where the default isn't appropriate. Fortunately, changing the color scale is not terribly complicated. We can check out different color scales available in the `RColorBrewer` package at *colorbrewer2.org*.

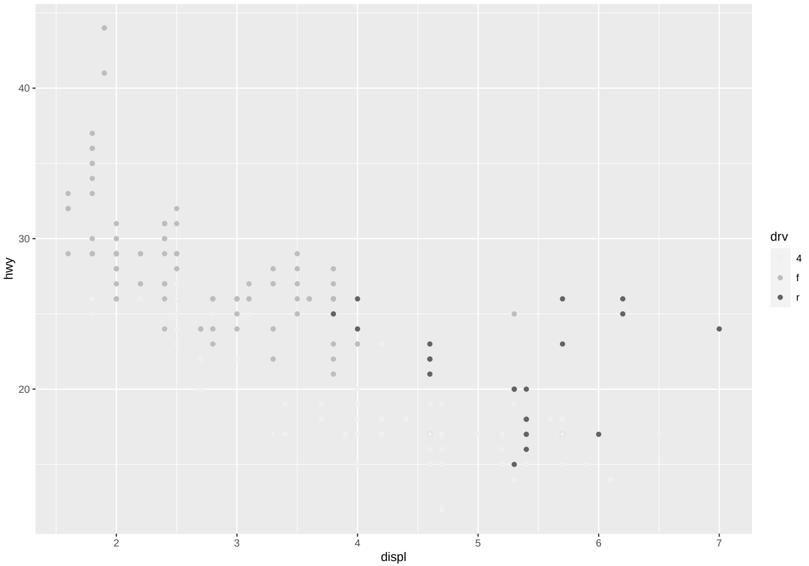
- For example, say we want our scale to be grayscale. Using the `scale_color_brewer()` function, we can change the default palette option to "greys."

Scales

```r
p <- mpg |>
  ggplot(aes(displ,hwy)) +
  geom_point(aes(color = drv)) +
  scale_color_brewer(palette = "Greys")
```

Scales

```
p
```



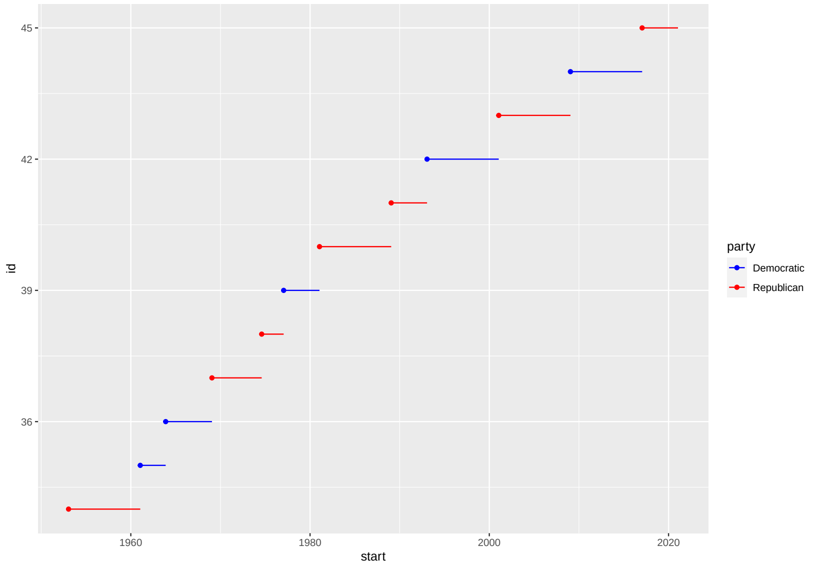
Scales

- We can also manually specify the colors for a categorical variable. For example, let's say we wanted to add the traditional red and blue to our presidents graph to denote political affiliation.

```r
p <- presidential |>
  mutate(id = 33 + row_number()) |>
  ggplot(aes(start,id,color=party)) +
  geom_point() +
  geom_segment(aes(xend = end, yend = id)) +
  scale_color_manual(values = c(Republican = "red",
                                Democratic = "blue"))
```

Scales

```
p
```



45
42
39
36
id
1960
1980
2000
2020
start
party
Democratic
Republican