

Data Visualization with ggplot2

Session 1: The Grammar of Graphics

Nathan Barron

 nathanbarron@ou.edu

Graphics are like onions

The `ggplot2` package in R is designed with a **grammar**, meaning that there are *core principles* that can be applied to create nearly *any* type of graphic.

- Data: the information you want to visualize
- Mapping: the description of how the data's variables are “mapped” to aesthetic attributes

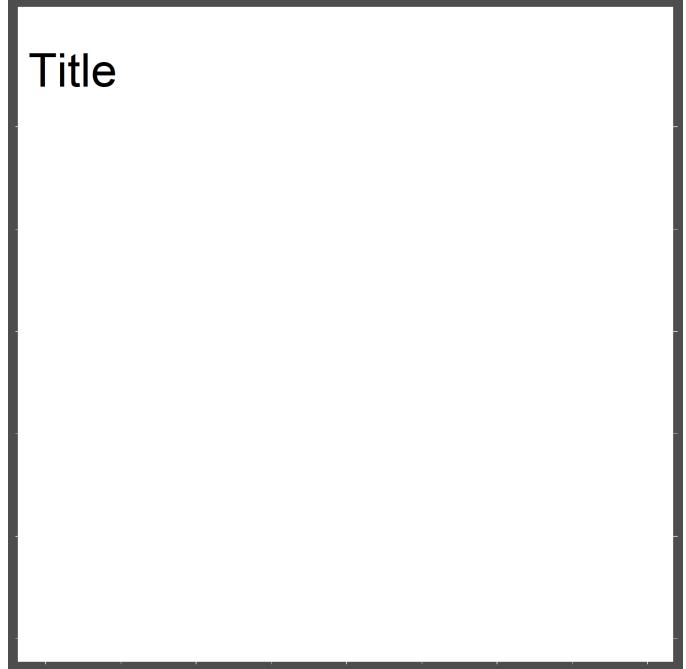
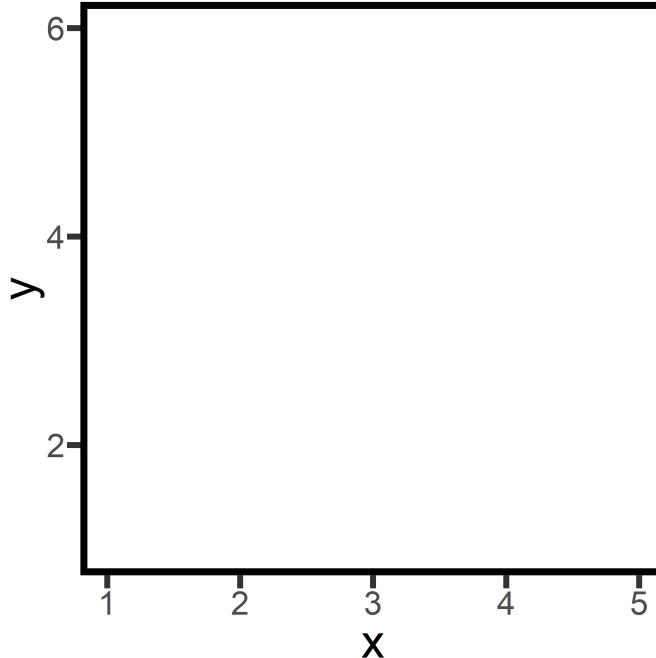
Essential Mapping Components

- Layer: collection of geometric elements (or *geoms*)
- Geom: visual components of a plot (e.g. points, lines, bars, etc.)
- Scales: map values in the data space to values in the aesthetic space (e.g. color, shape, size, etc.)
- Coord: coordinate system (e.g., the axes and gridlines)
- Facet: how to display subsets of data
- Theme: display features (e.g. font size, background color, etc.)



Plot layers

•
•
•
•
•



These are the **geom_point** elements. They are the visual representations of your data.

These are the scales and coordinate system. They are the axes and legends so that we can read values from the graph

These are the plot annotations. They provide crucial information for easy interpretation.

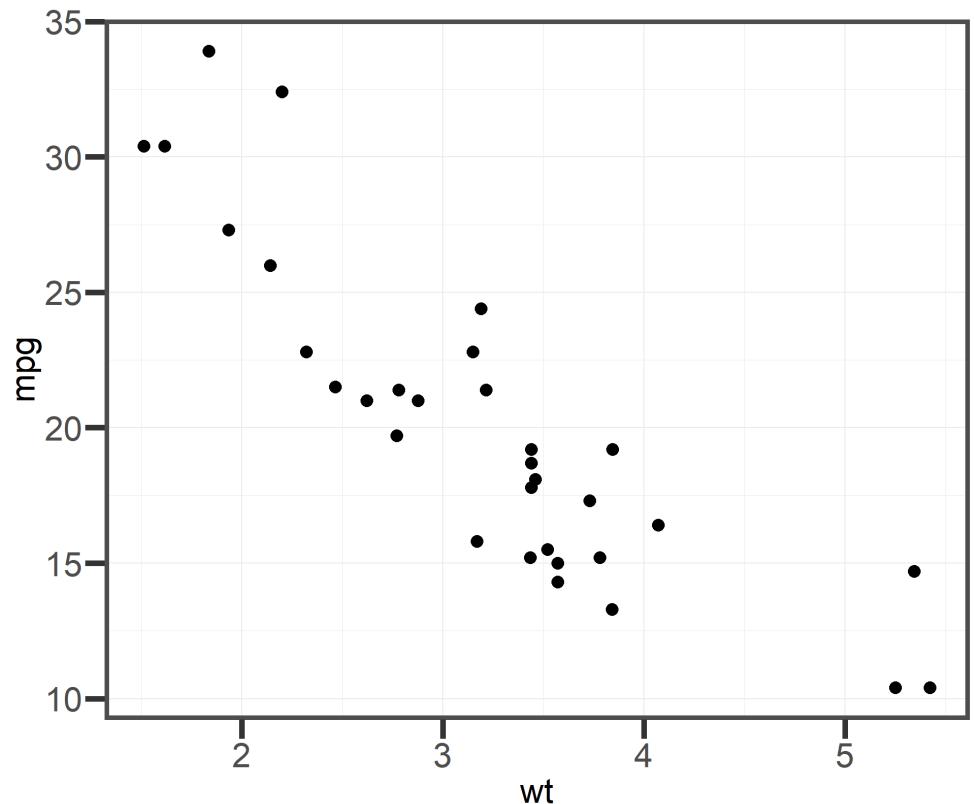
The foundation

```
1 ggplot(mtcars)
```



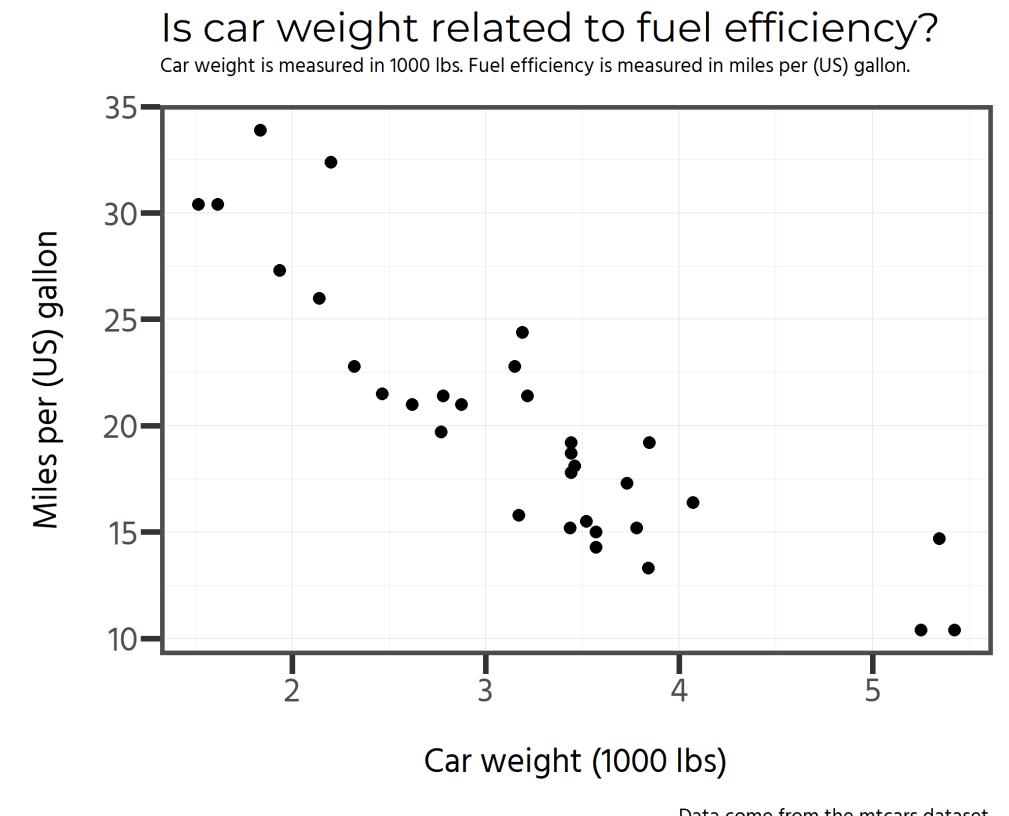
Plotting the data

```
1 ggplot(mtcars) +  
2   geom_point(aes(x=wt, y=mpg))
```



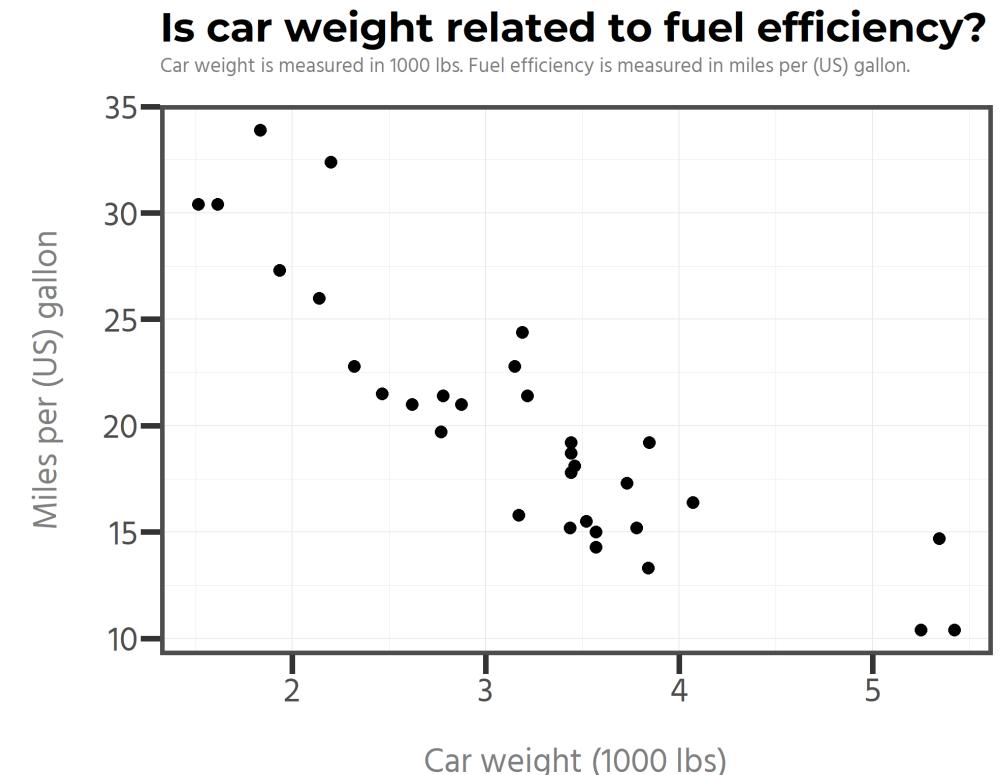
Annotating your plot

```
1 ggplot(mtcars) +  
2   geom_point(aes(x=wt, y=mpg)) +  
3   labs(x='Car weight (1000 lbs)',  
4       y='Miles per (US) gallon',  
5       title = 'Is car weight related to fuel efficiency?',  
6       subtitle = 'Car weight is measured in 1000 lbs. Fu  
7       caption = 'Data come from the mtcars dataset.'
```



Changing the theme

```
1 ggplot(mtcars) +  
2   geom_point(aes(x=wt, y=mpg)) +  
3   labs(x='Car weight (1000 lbs)',  
4       y='Miles per (US) gallon',  
5       title = 'Is car weight related to fuel efficiency?',  
6       subtitle = 'Car weight is measured in 1000 lbs. Fuel efficiency is measured in miles per (US) gallon.',  
7       caption = 'Data come from the mtcars dataset.') +  
8   theme(axis.title = element_text(color = 'grey50'),  
9         plot.title = element_text(face = 'bold'),  
10        plot.subtitle = element_text(color = 'grey50'),  
11        plot.caption = element_text(face = 'italic'))
```



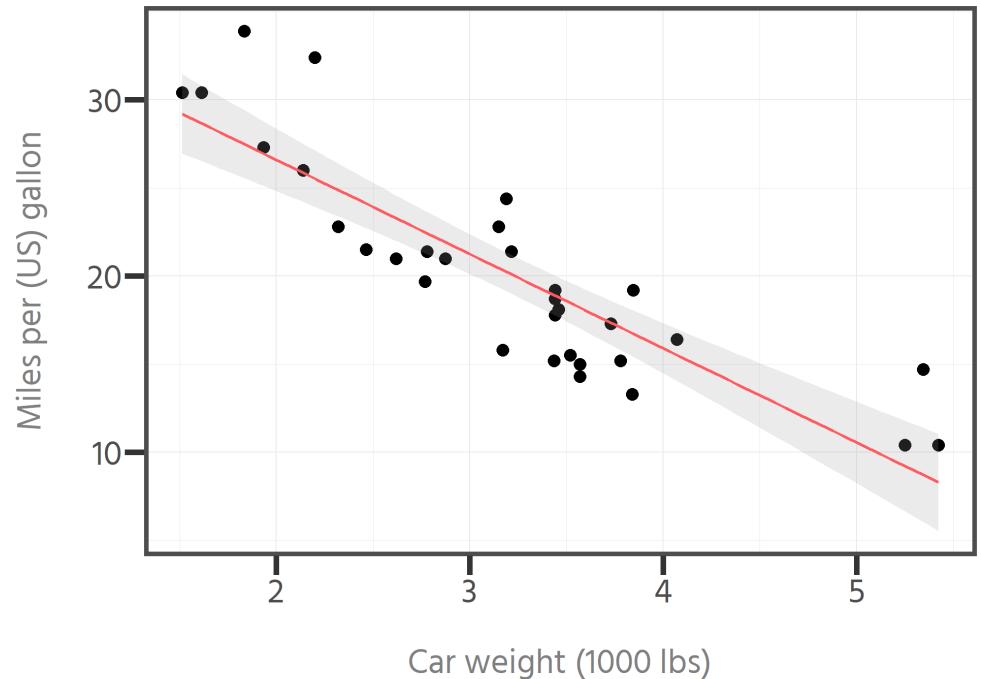
Data come from the mtcars dataset.

Adding new layers (fitted line)

```
1 ggplot(mtcars) +  
2   geom_point(aes(x=wt, y=mpg)) +  
3   geom_smooth(aes(x=wt, y=mpg),  
4                 method = 'lm',  
5                 alpha = .20,  
6                 color = '#FF5A5F') +  
7   labs(x='Car weight (1000 lbs)',  
8         y='Miles per (US) gallon',  
9         title = 'Is car weight related to fuel efficiency?',  
10        subtitle = 'Car weight is measured in 1000 lbs. Fu',  
11        caption = 'Data come from the mtcars dataset.') +  
12        theme(axis.title = element_text(color = 'grey50'),  
13          plot.title = element_text(face = 'bold'),  
14          plot.subtitle = element_text(color = 'grey50'),  
15          plot.caption = element_text(face = 'italic'))
```

Is car weight related to fuel efficiency?

Car weight is measured in 1000 lbs. Fuel efficiency is measured in miles per (US) gallon.



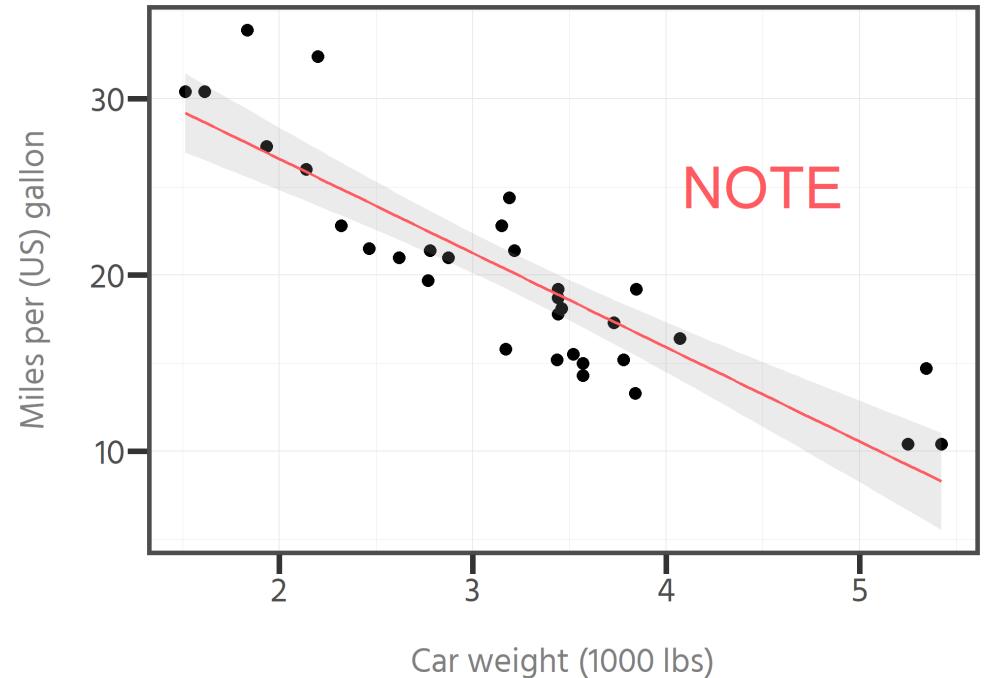
Data come from the mtcars dataset.

Adding new layers (annotations)

```
1 ggplot(mtcars) +  
2   geom_point(aes(x=wt, y=mpg)) +  
3   geom_smooth(aes(x=wt, y=mpg),  
4                 method = 'lm',  
5                 alpha = .20,  
6                 color = '#FF5A5F') +  
7   annotate("text", x=4.5, y=25, label = 'NOTE', color = 'red') +  
8   labs(x='Car weight (1000 lbs)',  
9         y='Miles per (US) gallon',  
10        title = 'Is car weight related to fuel efficiency?',  
11        subtitle = 'Car weight is measured in 1000 lbs. Fuel efficiency is measured in miles per (US) gallon.',  
12        caption = 'Data come from the mtcars dataset.') +  
13   theme(axis.title = element_text(color = 'grey50'),  
14         plot.title = element_text(face = 'bold'),  
15         plot.subtitle = element_text(color = 'grey50'),  
16         plot.caption = element_text(face = 'italic'))
```

Is car weight related to fuel efficiency?

Car weight is measured in 1000 lbs. Fuel efficiency is measured in miles per (US) gallon.



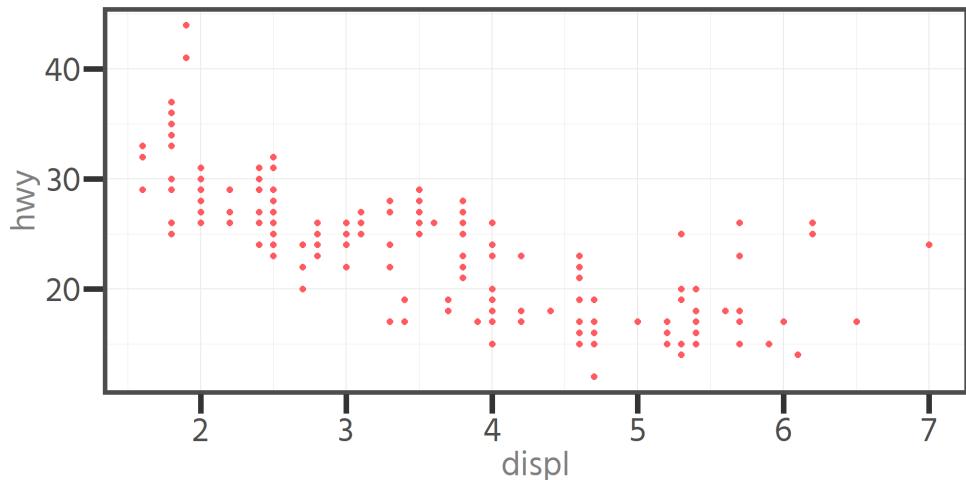
Data come from the mtcars dataset.

Aesthetic attributes

Aesthetics are parameters that can change the appearance of a plot.

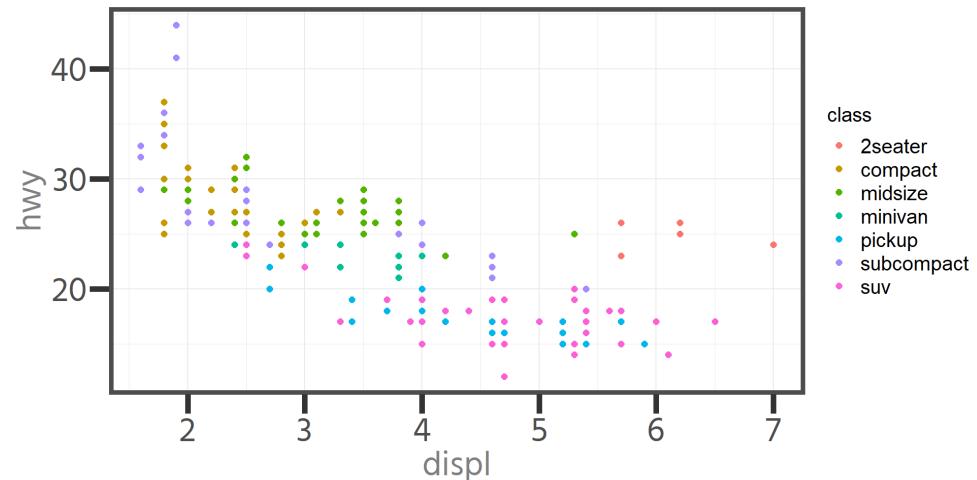
Global aesthetics affect all data at the same time.

```
1 ggplot(mpg) +  
2   geom_point(aes(displ, hwy),  
3               color = 'red')
```



Mapped aesthetics affect data based on the values of some specified variable.

```
1 ggplot(mpg) +  
2   geom_point(aes(displ, hwy, color = class))  
3 # Notice that 'color = _____' is now inside aes()
```



Linetype and shape aesthetics

6.'twodash'



5.'longdash'



4.'dotdash'



3.'dotted'



2.'dashed'



1.'solid'



0.'blank'



0 □ 1 ○ 2 ▲ 3 + 4 ×

5 ◇ 6 ▽ 7 ▨ 8 * 9 ◆

10 ⊕ 11 ✖ 12田 13⊗ 14□

15 ■ 16 ● 17▲ 18◆ 19●

20● 21● 22■ 23◆ 24▲ 25▼

Scales

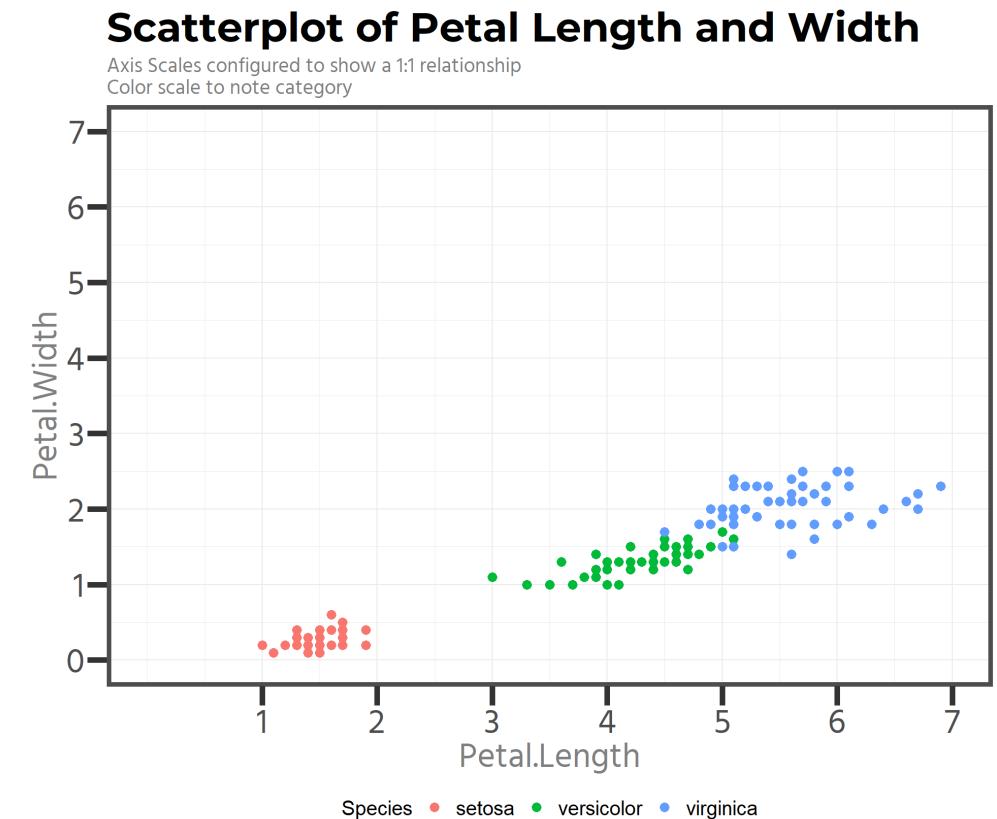
Scales refer to how data are translated into visual representations. This includes the x-axis, y-axis, colors, shapes, etc. Scales are essential components of the mapping process.

Key Elements

Name: a basic, though often neglected, description of scale

Limits: the range of a scale

Breaks: the subdivisions of range used to mark gradations



Facets

Facets show subsets of the data in a sequential, side-by-side format.

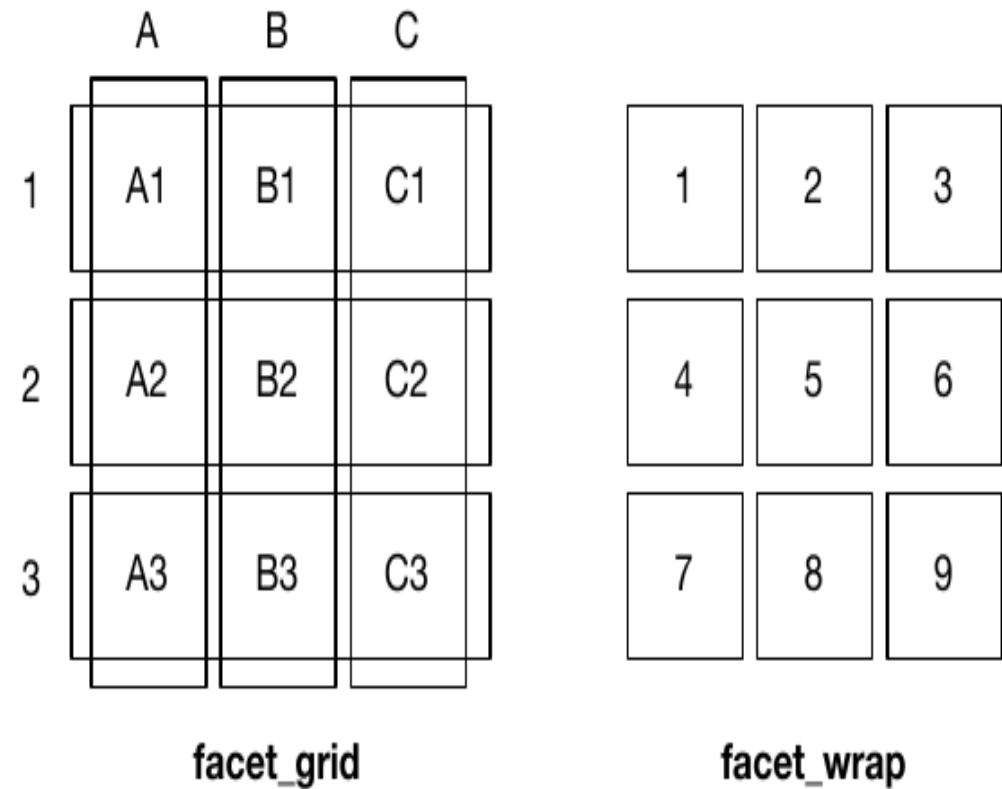
Two types of facets:

`facet_wrap`:

Subsetting data on one variable

`facet_grid`:

Subsetting data on more than one variable



Exercise #1

Data Visualization with ggplot2

Session 2: Building a Custom Theme

Nathan Barron

 nathanbarron@ou.edu

The theme layer

The `theme()` function allows you to override default graphics settings. There are two main components involved:

Elements: non-data elements of the graphic that you can control

Examples:

- `plot.title`
- `axis.ticks.x`
- `legend.key.height`
- etc.

Element functions: the description of element features

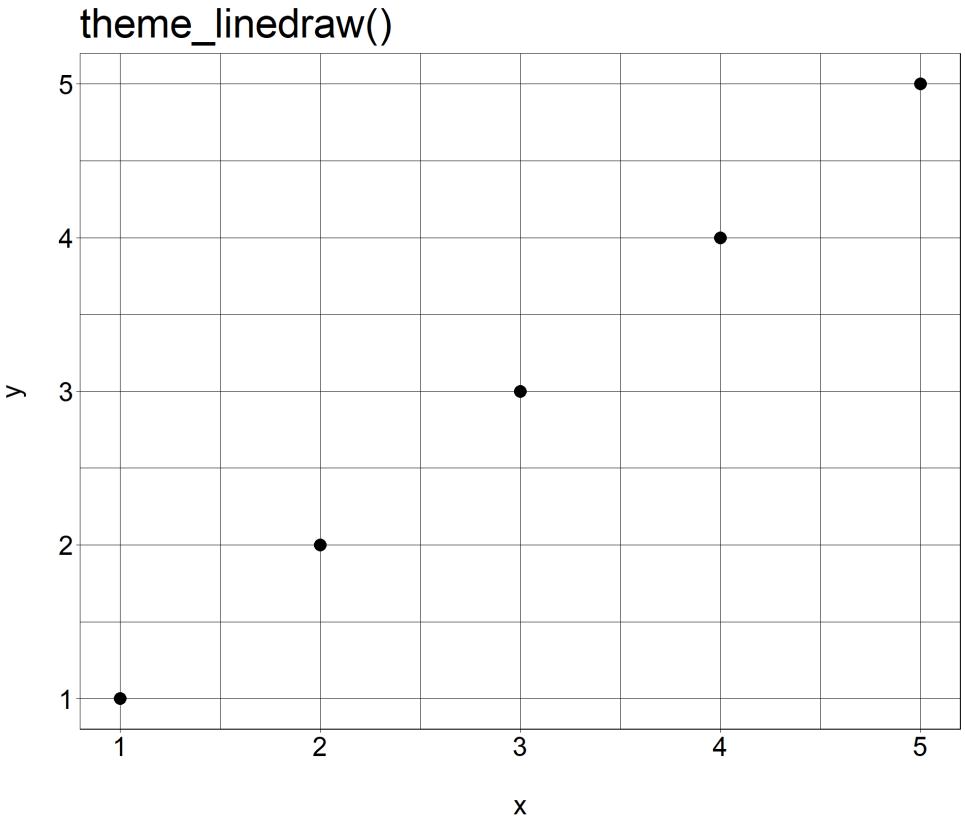
Examples:

- `element_text()`
- `element_rect()`
- `element_line()`
- etc.

Complete themes

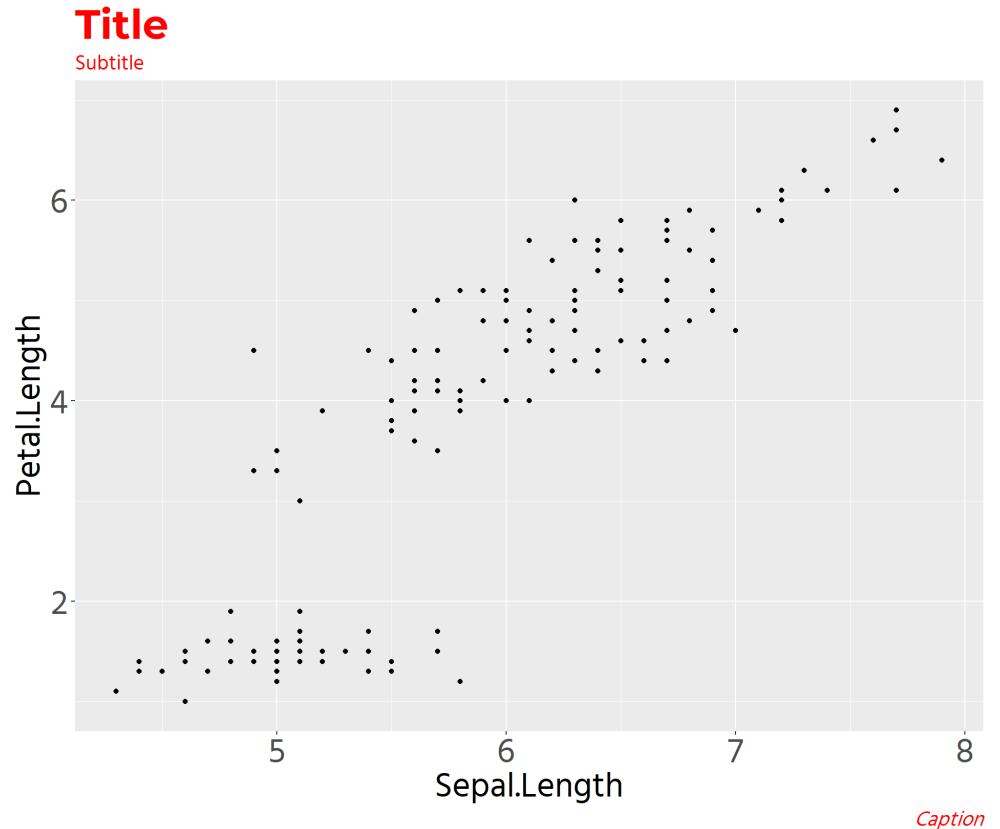
`ggplot2` has eight pre-made themes built in:

- `theme_grey()`
 - `theme_bw()`
- `theme_linedraw()`
 - `theme_light()`
 - `theme_dark()`
- `theme_minimal()`
- `theme_classic()`
- `theme_void()`



Plot elements

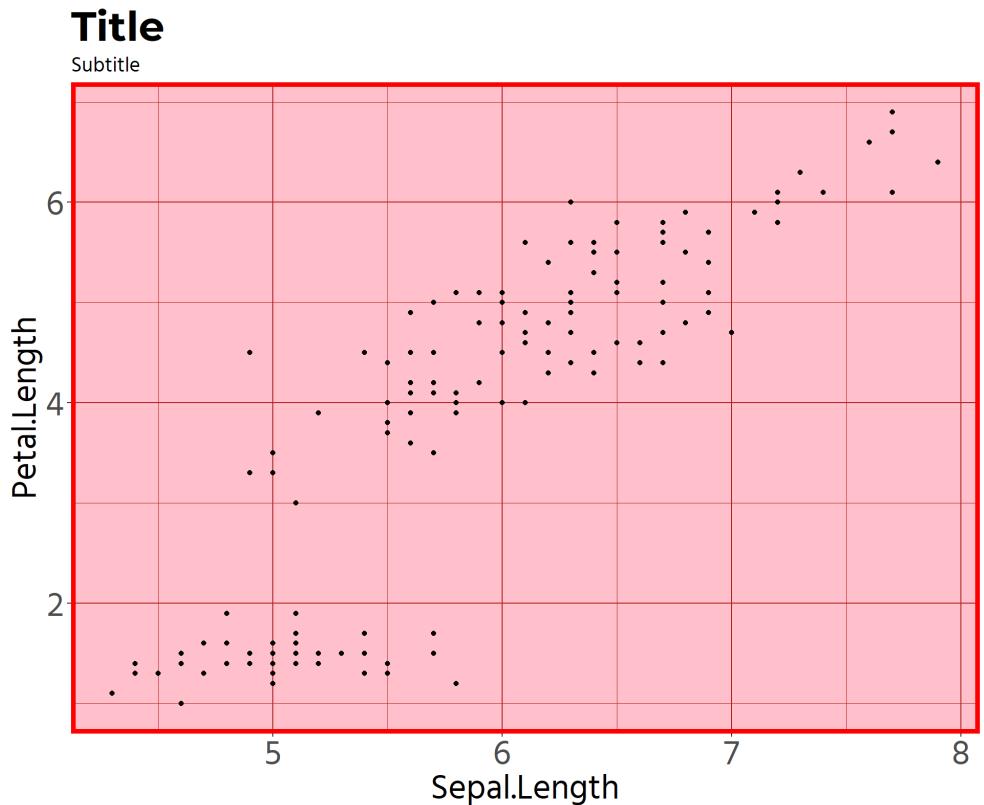
- `plot.background`
- `plot.title`
- `plot.title.position`
- `plot.subtitle`
- `plot.caption`
- `plot.caption.position`
- `plot.tag`
- `plot.tag.position`



Caption

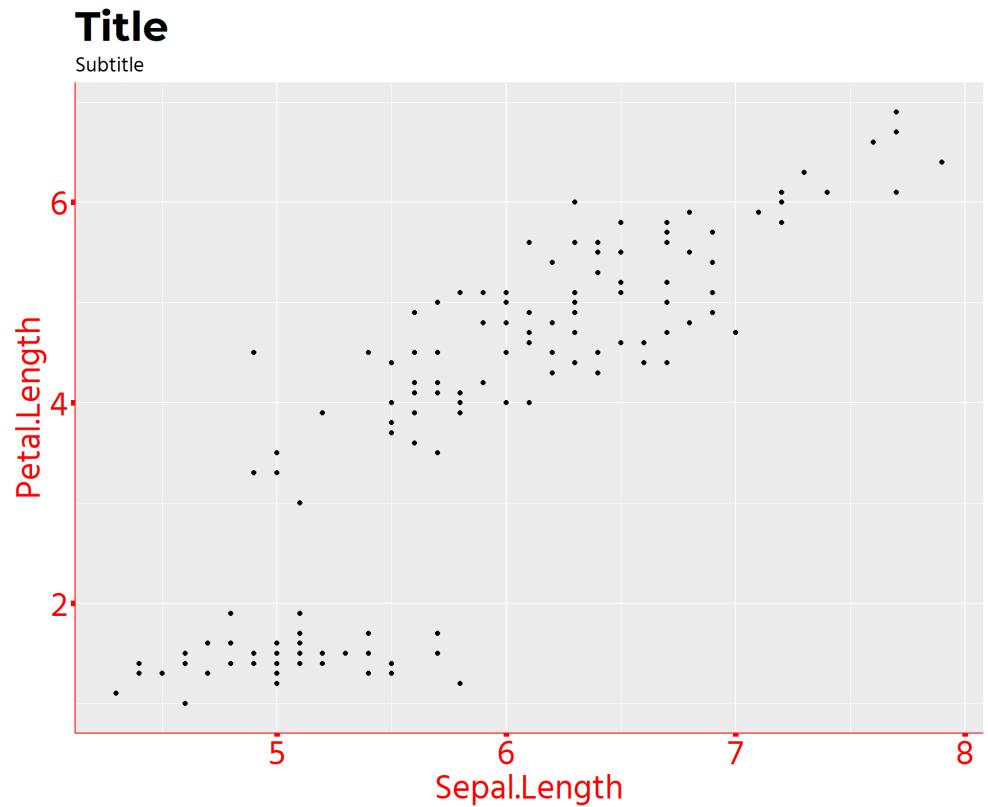
Panel elements

- panel.background
- panel.border
- panel.grid
 - panel.grid.major
 - panel.grid.major.x
 - panel.grid.major.y
 - panel.grid.minor
 - panel.grid.minor.x
 - panel.grid.minor.y



Axis elements

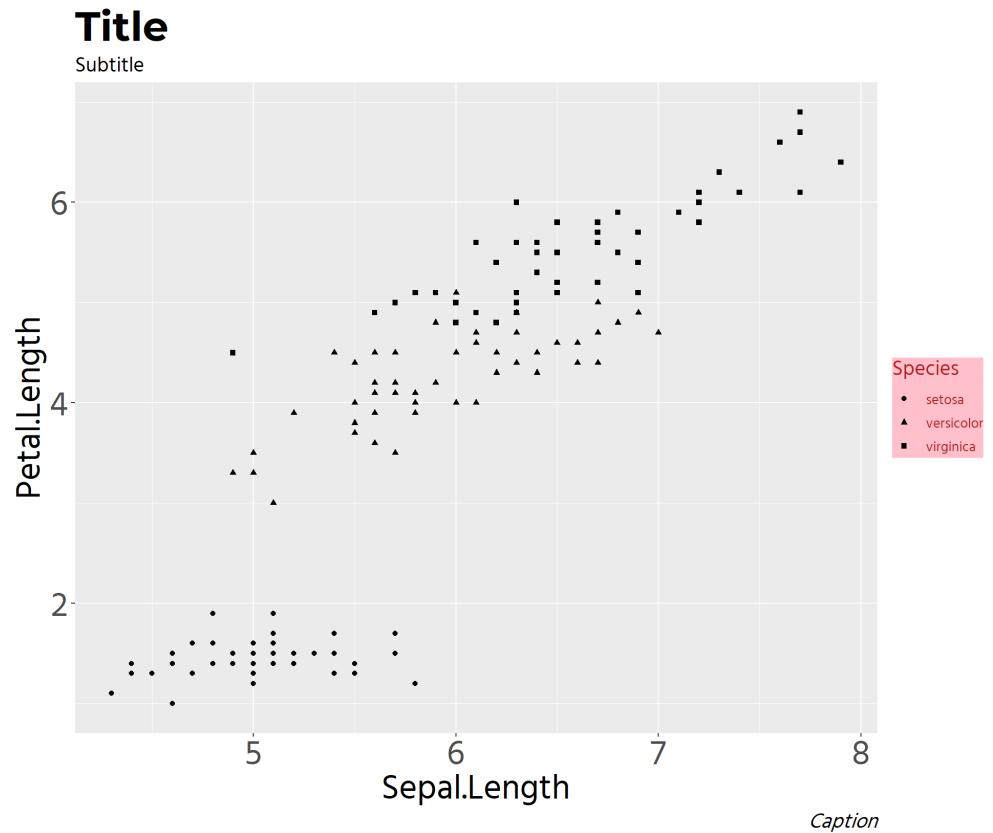
- `axis.title`
 - `axis.title.x` (.top/.bottom)
 - `axis.title.y` (.left/.right)
- `axis.text`
 - `axis.text.x` (.top/.bottom)
 - `axis.text.y` (.left/.right)
- `axis.ticks`
 - `axis.ticks.x` (.top/.bottom)
 - `axis.ticks.y` (.left/.right)
- `axis.ticks.length`
 - `axis.ticks.length.x` (.top/.bottom)
 - `axis.ticks.length.y` (.left/.right)
- `axis.line`
 - `axis.line.x` (.top/.bottom)
 - `axis.line.y` (.left/.right)



Caption

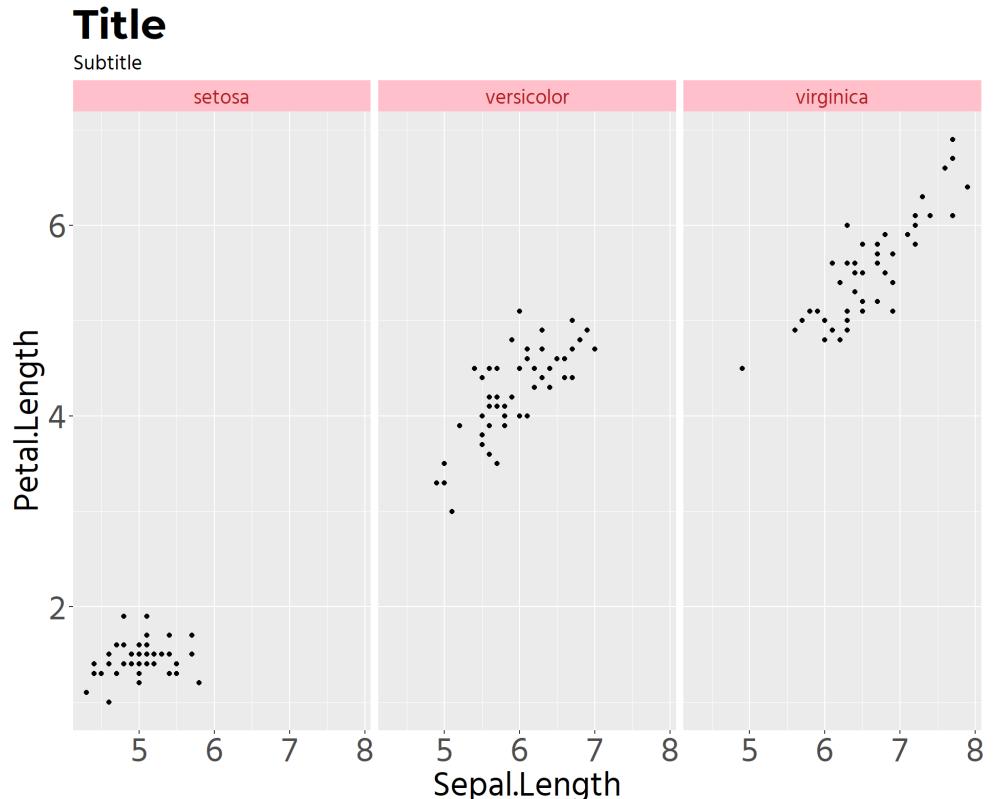
Legend elements

- `legend.background`
- `legend.margin`
- `legend.spacing`
 - `legend.spacing.x`
 - `legend.spacing.y`
- `legend.key`
 - `legend.key.size`
 - `legend.key.height`
 - `legend.key.width`
- `legend.text`
 - `legend.text.align`
- `legend.title`
 - `legend.title.align`
- `legend.position`
- `legend.direction`
- `legend.justification`
- `legend.box`
 - `legend.box.just`
 - `legend.box.margin`
 - `legend.box.background`
 - `legend.box.spacing`



Faceting elements

- `strip.background`
 - `strip.background.x`
 - `strip.background.y`
- `strip.placement`
- `strip.text`
 - `strip.text.x`
 - `strip.text.x.bottom`
 - `strip.text.x.top`
 - `strip.text.y`
 - `strip.text.y.left`
 - `strip.text.y.right`
- `panel.spacing`
 - `panel.spacing.x`
 - `panel.spacing.y`



Colors in R

Colors can be easily referenced in with a hex code, rbg code, or by name (if applicable).

Following colors are available by name:

| | | | | | | | | | | |
|----------------|-----------------|--------------|--------------|--------|---------|----------------------|-------------------|----------------|------------|-------------|
| coral3 | deeppink4 | gray27 | gray87 | grey39 | grey99 | lightpink1 | mistyrose1 | pink4 | slategray1 | yellowgreen |
| coral2 | deeppink3 | gray26 | gray86 | grey38 | grey98 | lightpink | mistyrose | pink3 | slategray | yellow4 |
| coral1 | deeppink2 | gray25 | gray85 | grey37 | grey97 | lightgrey | mintcream | pink2 | slateblue4 | yellow3 |
| coral | deeppink1 | gray24 | gray84 | grey36 | grey96 | lightgreen | midnightblue | pink1 | slateblue3 | yellow2 |
| chocolate4 | deeppink | gray23 | gray83 | grey35 | grey95 | lightgray | mediumvioletred | pink | slateblue2 | yellow1 |
| chocolate3 | darkviolet | gray22 | gray82 | grey34 | grey94 | lightgoldenrodyellow | mediumturquoise | slateblue1 | slategray1 | yellow |
| chocolate2 | darkturquoise | gray21 | gray81 | grey33 | grey93 | lightgoldenrod1 | mediumspringgreen | peachpuff4 | slateblue1 | yellow3 |
| chocolate1 | darkslategray | gray20 | gray80 | grey32 | grey92 | lightgoldenrod2 | mediumstateblue | peachpuff3 | skyblue4 | yellow2 |
| chocolate | darkslategray4 | gray19 | gray79 | grey31 | grey91 | lightgoldenrod3 | mediumseagreen | peachpuff2 | skyblue3 | yellow1 |
| chartreuse4 | darkslategray3 | gray18 | gray78 | grey30 | grey90 | lightgoldenrod4 | mediumpurple4 | peachpuff1 | skyblue2 | yellow |
| chartreuse3 | darkslategray2 | gray17 | gray77 | grey29 | grey89 | lightgoldenrod1 | mediumpurple3 | peachpuff | skyblue1 | yellow1 |
| chartreuse2 | darkslategray1 | gray16 | gray76 | grey28 | grey88 | lightgoldenrod | mediumpurple2 | papayawhip | skyblue | yellow |
| chartreuse1 | darkslategray | gray15 | gray75 | grey27 | grey87 | lightcyan4 | mediumpurple1 | sienna4 | wheat4 | yellow |
| chartreuse | darkslateblue | gray14 | gray74 | grey26 | grey86 | lightcyan3 | mediumpurple | sienna3 | wheat3 | yellow3 |
| cadetblue4 | darkseagreen4 | gray13 | gray73 | grey25 | grey85 | lightcyan2 | mediumpurple | sienna2 | wheat2 | yellow2 |
| cadetblue3 | darkseagreen3 | gray12 | gray72 | grey24 | grey84 | lightcyan1 | mediumorchid4 | sienna1 | wheat1 | yellow1 |
| cadetblue2 | darkseagreen2 | gray11 | gray71 | grey23 | grey83 | lightcyan | mediumorchid3 | palevioletred1 | wheat | yellow |
| cadetblue1 | darkseagreen1 | gray10 | gray70 | grey22 | grey82 | lightcyan | mediumorchid2 | palevioletred2 | wheat4 | yellow4 |
| cadetblue | darkseagreen | gray9 | gray69 | grey21 | grey81 | lightblue4 | mediumorchid1 | palevioletred3 | wheat3 | yellow3 |
| burlywood4 | darksalmon | gray8 | gray68 | grey20 | grey80 | lightblue3 | mediumorchid | palevioletred2 | wheat2 | yellow2 |
| burlywood3 | darkred | gray7 | gray67 | grey19 | grey79 | lightblue2 | mediumblue | palevioletred1 | wheat1 | yellow1 |
| burlywood2 | darkorchid4 | gray6 | gray66 | grey18 | grey78 | lightblue1 | mediumblue | palevioletred | wheat | yellow |
| burlywood1 | darkorchid3 | gray5 | gray65 | grey17 | grey77 | lightblue | mediumblue | palevioletred4 | wheat4 | yellow4 |
| burlywood | darkorchid2 | gray4 | gray64 | grey16 | grey76 | lemonchiffon4 | mediumblue | palevioletred3 | wheat3 | yellow3 |
| brown4 | darkorchid1 | gray3 | gray63 | grey15 | grey75 | lemonchiffon3 | mediumblue | palevioletred2 | wheat2 | yellow2 |
| brown3 | darkorchid | gray2 | gray62 | grey14 | grey74 | lemonchiffon2 | mediumblue | palevioletred1 | wheat1 | yellow1 |
| brown2 | darkorange4 | gray1 | gray61 | grey13 | grey73 | lemonchiffon1 | mediumblue | palevioletred | wheat | yellow |
| brown1 | darkorange3 | gray0 | gray60 | grey12 | grey72 | lemonchiffon | mediumblue | palevioletred4 | wheat4 | yellow4 |
| brown | darkorange2 | gray | gray59 | grey11 | grey71 | lawngreen | mediumblue | palevioletred3 | wheat3 | yellow3 |
| blueviolet | darkorange1 | goldenrod4 | goldenrod3 | grey58 | grey70 | lavenderblush4 | mediumblue | palevioletred2 | wheat2 | yellow2 |
| blue4 | darkorange | goldenrod2 | goldenrod1 | grey57 | grey71 | lavenderblush3 | mediumblue | palevioletred1 | wheat1 | yellow1 |
| blue3 | darkolivegreen4 | goldenrod | goldenrod | grey56 | grey72 | lavenderblush2 | mediumblue | palevioletred | wheat | yellow |
| blue2 | darkolivegreen3 | goldenrod1 | goldenrod | grey55 | grey73 | lavenderblush1 | mediumblue | palevioletred4 | wheat4 | yellow4 |
| blue1 | darkolivegreen2 | goldenrod | goldenrod | grey54 | grey74 | linen | mediumblue | palevioletred3 | wheat3 | yellow3 |
| blue | darkolivegreen1 | gold4 | gold3 | grey53 | grey75 | lavender | mediumblue | palevioletred2 | wheat2 | yellow2 |
| blanchedalmond | darkolivegreen | gold2 | gold2 | grey52 | grey76 | lightyellow4 | mediumblue | palevioletred1 | wheat1 | yellow1 |
| black | darkmagenta | gold1 | gold1 | grey51 | grey77 | khaki4 | mediumblue | palevioletred | wheat | yellow |
| bisque4 | darkkhaki | gold1 | gold1 | grey50 | grey78 | khaki3 | mediumblue | palevioletred4 | wheat4 | yellow4 |
| bisque3 | darkgrey | gold | gold | grey49 | grey79 | khaki2 | mediumblue | palevioletred3 | wheat3 | yellow3 |
| bisque2 | darkgreen | ghostwhite | ghostwhite | grey48 | grey80 | khaki1 | mediumblue | palevioletred2 | wheat2 | yellow2 |
| bisque1 | darkgray | gainsboro | gainsboro | grey47 | grey81 | khaki | mediumblue | palevioletred1 | wheat1 | yellow1 |
| bisque | darkgoldenrod4 | forestgreen | forestgreen | grey46 | grey82 | lightyellow4 | mediumblue | palevioletred | wheat | yellow |
| beige | darkgoldenrod3 | floralwhite | floralwhite | grey45 | grey83 | lightyellow3 | mediumblue | palevioletred4 | wheat4 | yellow4 |
| azure4 | darkgoldenrod2 | firebrick4 | firebrick4 | grey44 | grey84 | lightyellow2 | mediumblue | palevioletred3 | wheat3 | yellow3 |
| azure3 | darkgoldenrod1 | firebrick3 | firebrick3 | grey43 | grey85 | lightyellow1 | mediumblue | palevioletred2 | wheat2 | yellow2 |
| azure2 | darkgoldenrod | firebrick2 | firebrick2 | grey42 | grey86 | orange4 | mediumblue | palevioletred1 | wheat1 | yellow1 |
| azure1 | darkcyan | firebrick1 | firebrick1 | grey41 | grey87 | orange3 | mediumblue | palevioletred | wheat | yellow |
| azure | darkblue | firebrick | firebrick | grey40 | grey88 | orange2 | mediumblue | palevioletred4 | wheat4 | yellow4 |
| aquamarine4 | cyan4 | dodgerblue4 | dodgerblue4 | grey39 | grey89 | orange1 | mediumblue | palevioletred3 | wheat3 | yellow3 |
| aquamarine3 | cyan3 | dodgerblue3 | dodgerblue3 | grey38 | grey90 | orange | mediumblue | palevioletred2 | wheat2 | yellow2 |
| aquamarine2 | cyan2 | dodgerblue2 | dodgerblue2 | grey37 | grey91 | orange3 | mediumblue | palevioletred1 | wheat1 | yellow1 |
| aquamarine1 | cyan1 | dodgerblue1 | dodgerblue1 | grey36 | grey92 | orange2 | mediumblue | palevioletred | wheat | yellow |
| aquamarine | cyan | dodgerblue | dodgerblue | grey35 | grey93 | orange1 | mediumblue | palevioletred4 | wheat4 | yellow4 |
| antiquewhite4 | cornsilk4 | dimgray | dimgray | grey34 | grey94 | orange | mediumblue | palevioletred3 | wheat3 | yellow3 |
| antiquewhite3 | cornsilk3 | dimgray | dimgray | grey33 | grey95 | orange2 | mediumblue | palevioletred2 | wheat2 | yellow2 |
| antiquewhite2 | cornsilk2 | deepskyblue4 | deepskyblue4 | grey32 | grey96 | orange1 | mediumblue | palevioletred1 | wheat1 | yellow1 |
| antiquewhite1 | cornsilk1 | deepskyblue3 | deepskyblue3 | grey31 | grey97 | navajowhite4 | mediumblue | palevioletred | wheat | yellow |
| antiquewhite | cornsilk | deepskyblue2 | deepskyblue2 | grey30 | grey98 | navajowhite3 | mediumblue | palevioletred4 | wheat4 | yellow4 |
| aliceblue | cornflowerblue | deepskyblue1 | deepskyblue1 | grey29 | grey99 | navajowhite2 | mediumblue | palevioletred3 | wheat3 | yellow3 |
| white | coral4 | deepskyblue | deepskyblue | grey28 | grey100 | navajowhite1 | mediumblue | palevioletred2 | wheat2 | yellow2 |

Fonts in R

You can use any font available on your computer in a `ggplot2` graphic.

For Windows users

```
1 install.packages('extrafont')
2 library('extrafont')
3
4 font_import()
5 # You will be prompted to continue [y/n]
6 # Type 'y' and press enter
7
8 loadfonts(device="win")
```

For Mac users

```
1 install.packages('extrafont')
2 library('extrafont')
3
4 font_import()
5 # You will be prompted to continue [y/n]
6 # Type 'y' and press enter
7
8 loadfonts()
```

Afterwards, you should be able to utilize fonts:

```
1 ggplot() +
2   ...
3   theme(
4     ...
5       = element_text(family = "font-name",
6                       face = "type-face-name")
7   )
```

Be sure to know which font families and font faces you have available.

Building a custom theme

Instead of re-using all of your `theme()` code (which can be lengthy), try saving the code as an object. Then, you can add the object to a ggplot.

```
1 # You have a pre-theme plot saved as 'existing_plot'  
2 existing_plot <- ggplot() + ...  
3  
4 # Save your theme settings as an object ('my_theme')  
5 my_theme <- theme(plot.title = element_text(...),  
6                     ...  
7                     ...  
8                     )  
9  
10 # Add your custom theme to the existing plot  
11 existing_plot + my_theme
```

Exercise #2

Data Visualization with ggplot2

Session 3: Visualizing Diverse Data

Nathan Barron

 nathanbarron@ou.edu

Different data need different visualizations

| data | mpg | cyl | am | cyl x am | mpg | cyl | disp | hp | drat | wt | qsec | vs | am | gear | carb |
|-------------------|------|-----|-------|----------|------|-------|-------|----|------|----|------|----|----|------|------|
| Mazda RX4 | 21.0 | 6 | 160.0 | 110 | 3.90 | 2.620 | 16.46 | 0 | 1 | 4 | 4 | 0 | 1 | 4 | 4 |
| Mazda RX4 Wag | 21.0 | 6 | 160.0 | 110 | 3.90 | 2.875 | 17.02 | 0 | 1 | 4 | 4 | 0 | 1 | 4 | 4 |
| Datsun 710 | 22.8 | 4 | 108.0 | 93 | 3.85 | 2.320 | 18.61 | 1 | 1 | 4 | 1 | 1 | 1 | 4 | 1 |
| Hornet 4 Drive | 21.4 | 6 | 258.0 | 110 | 3.08 | 3.215 | 19.44 | 1 | 0 | 3 | 1 | 1 | 0 | 3 | 1 |
| Hornet Sportabout | 18.7 | 8 | 360.0 | 175 | 3.15 | 3.440 | 17.02 | 0 | 0 | 3 | 2 | 0 | 0 | 3 | 2 |
| Valiant | 18.1 | 6 | 225.0 | 105 | 2.76 | 3.460 | 20.22 | 1 | 0 | 3 | 1 | 1 | 0 | 3 | 1 |
| Duster 360 | 14.3 | 8 | 360.0 | 245 | 3.21 | 3.570 | 15.84 | 0 | 0 | 3 | 4 | 0 | 0 | 3 | 4 |
| Merc 240D | 24.4 | 4 | 146.7 | 62 | 3.69 | 3.190 | 20.00 | 1 | 0 | 4 | 2 | 1 | 0 | 4 | 2 |
| Merc 230 | 22.8 | 4 | 140.8 | 95 | 3.92 | 3.150 | 22.90 | 1 | 0 | 4 | 2 | 1 | 0 | 4 | 2 |
| Merc 280 | 19.2 | 6 | 167.6 | 123 | 3.92 | 3.440 | 18.30 | 1 | 0 | 4 | 4 | 1 | 0 | 4 | 4 |

Types of data

Levels of Data Measurement

- Categorical
 - Binary
 - Non-binary
- Continuous

[.](#) Cat.: Binary Cat.: Non-binary

Continuous

Click through each header to read more about each type

Storing Data in R

- Boolean
- Character
- Numerical
- Factor

[.](#) Boolean Character Numerical

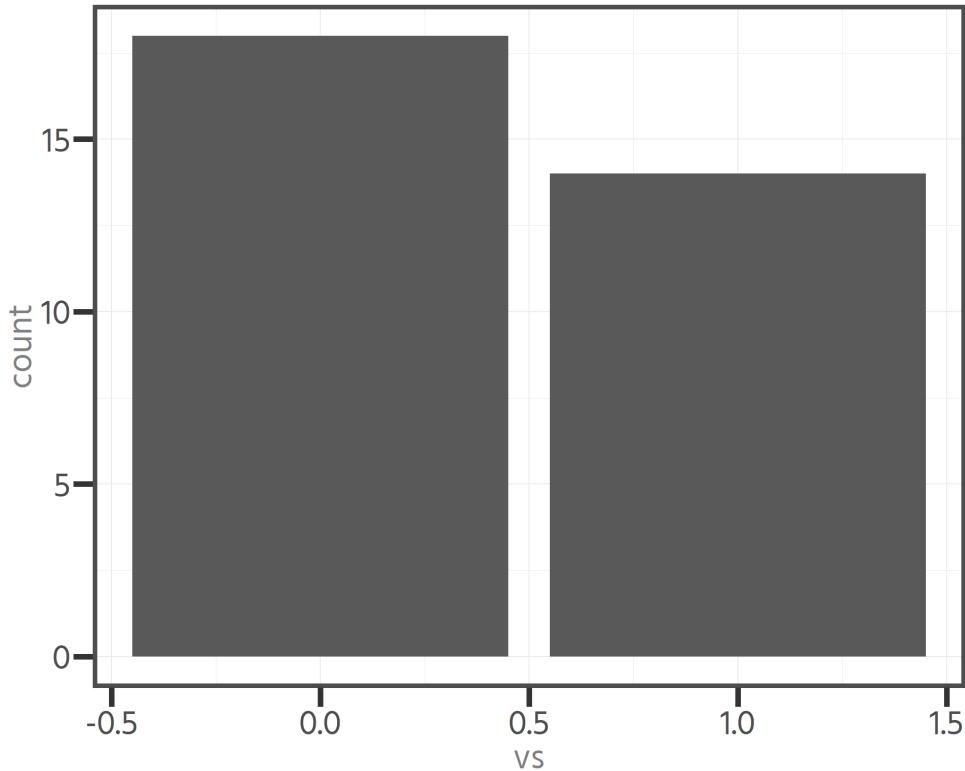
Factor Dates

Click through each header to read more about each type

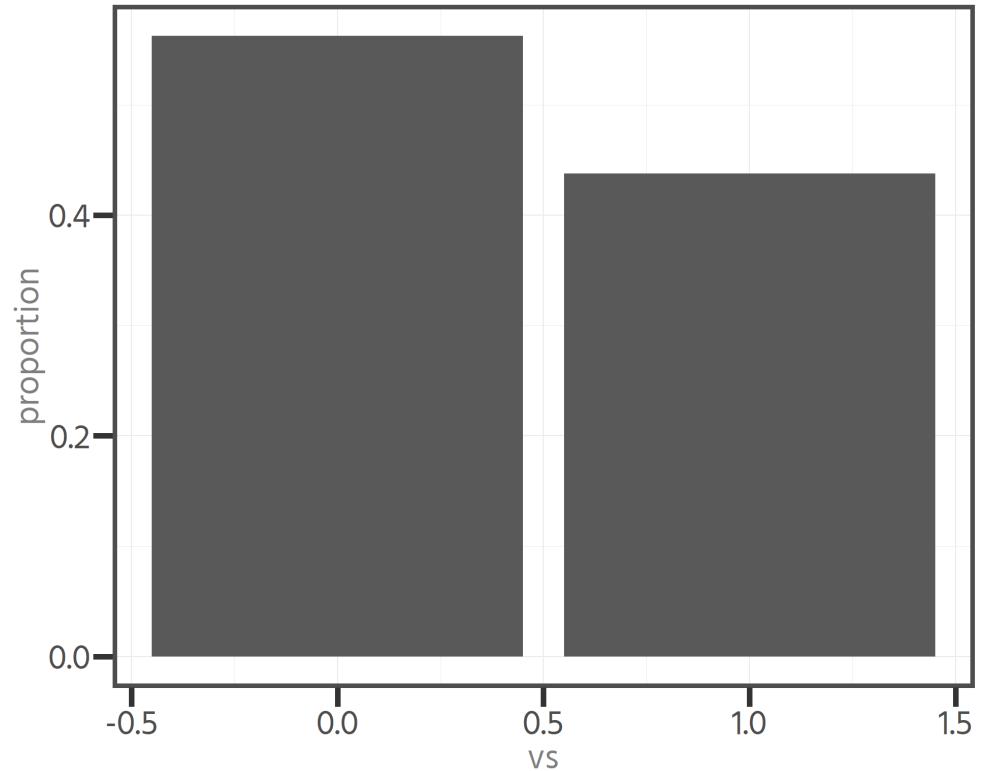
Visualizing a single variable

Binary

Barplot (count)



Barplot (proportion)

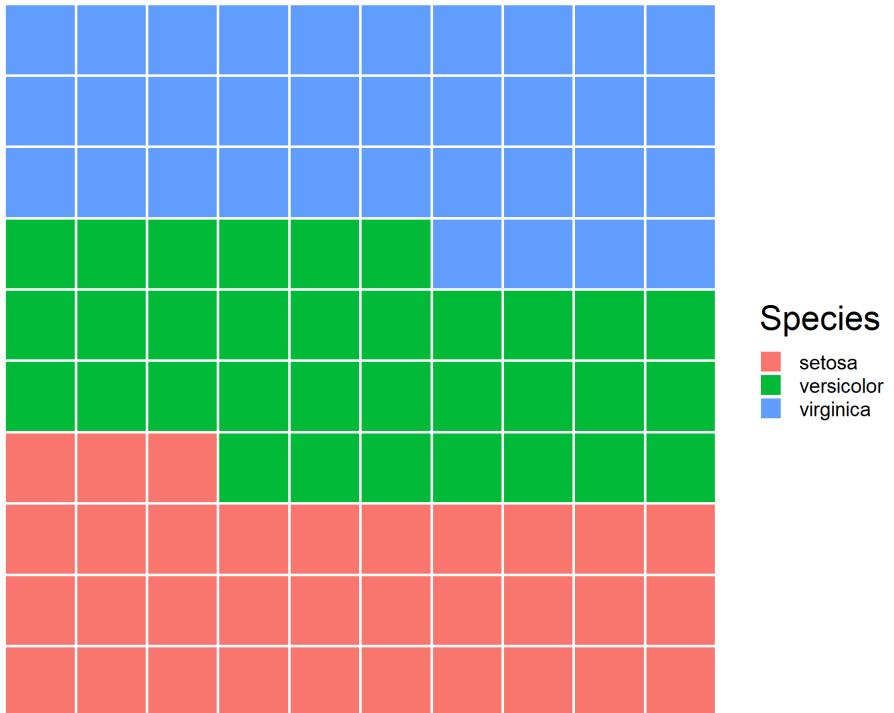


```
1 # Barplot with frequency on y-axis
2 ggplot(mtcars) +
3   geom_bar(aes(x=vs))
```

```
1 # Barplot with proportion on y-axis
2 ggplot(mtcars) +
3   geom_bar(aes(x=vs, y=..count../sum(..count..)))
```

Non-binary Categorical

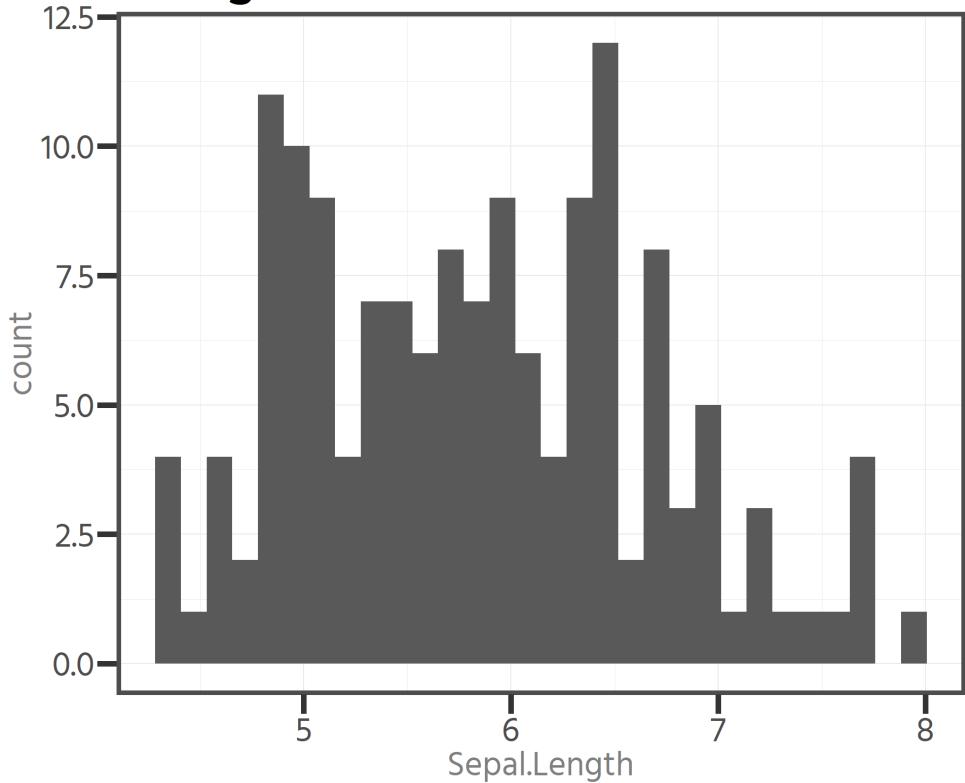
Waffle plot



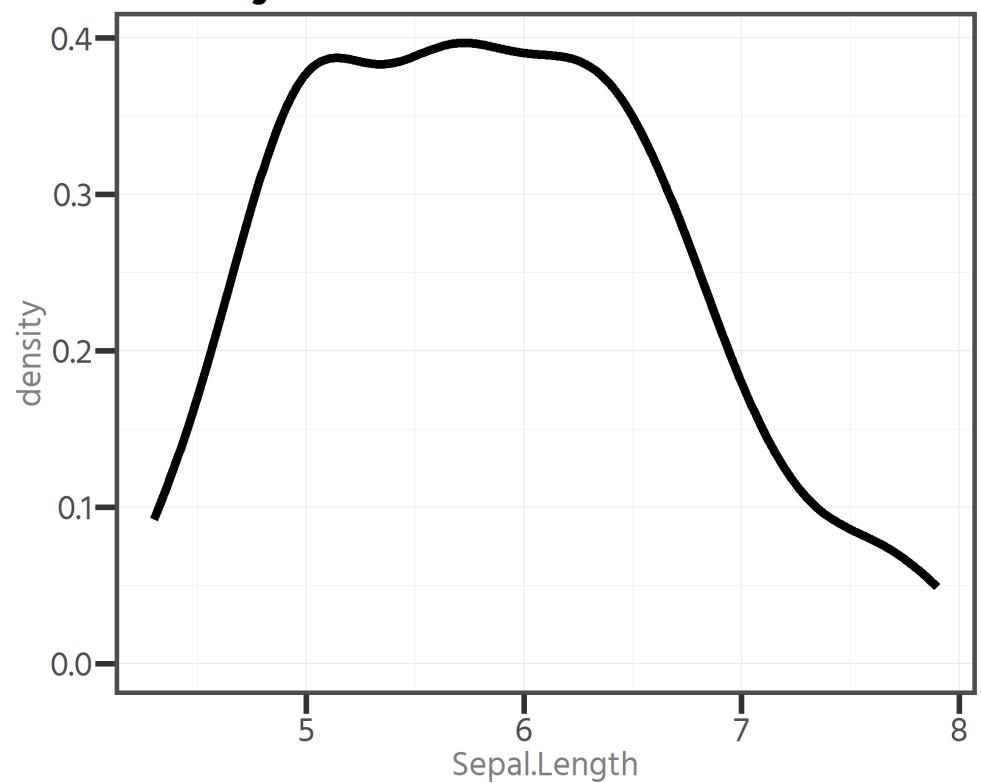
```
1 # install.packages("waffle",
2 #                      repos = "https://cinc.rud.is")
3
4 library(waffle)
5 library(dplyr)
6
7 iris %>%
8   count(Species) %>%
9   ggplot(aes(fill = Species, values = n)) +
10  geom_waffle(size = 1,
11               colour = "white",
12               na.rm=TRUE,
13               flip = TRUE,
14               make_proportional = TRUE) +
15  theme_void() +
16  coord_equal()
```

Continuous

Histogram



Density



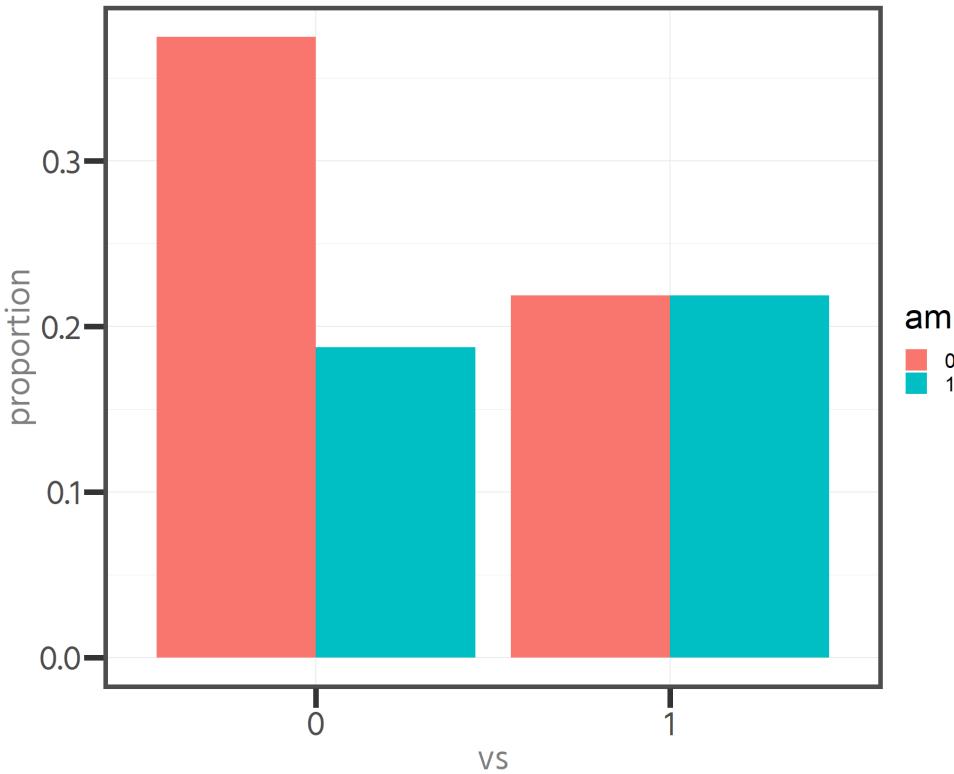
```
1 ggplot(iris) +  
2   geom_histogram(aes(x=Sepal.Length))
```

```
1 ggplot(iris) +  
2   geom_density(aes(x=Sepal.Length))
```

Visualizing multiple variables

Multiple categorical variables

Barplot (proportion)



```
1 mtcars %>%
2   mutate(vs = factor(vs), am = factor(am)) %>%
3   ggplot(.) +
4   geom_bar( aes(x=vs,
5                 y=..count../sum(..count..),
6                 group=am,
7                 fill=am),
8             position = 'dodge')
```

Multiple categorical variables II

Mosaic plot

Do you recline?

always

usually

about half
the time

once
in a while

never

no

somewhat

yes

Is it rude to recline?

```
1 # library(ggmosaic)
2 # library(tidyr)
3
4 ggplot(ggmosaic::fly) +
5   geom_mosaic(aes(x = product(do_you_recline, rude_to_recline),
6     .drop = FALSE)) +
7   theme(panel.grid = element_blank(),
8     legend.position = 'none')
```

Categorical heatmap

Do you recline?

always

usually

about half
the time

once
in a while

never

no

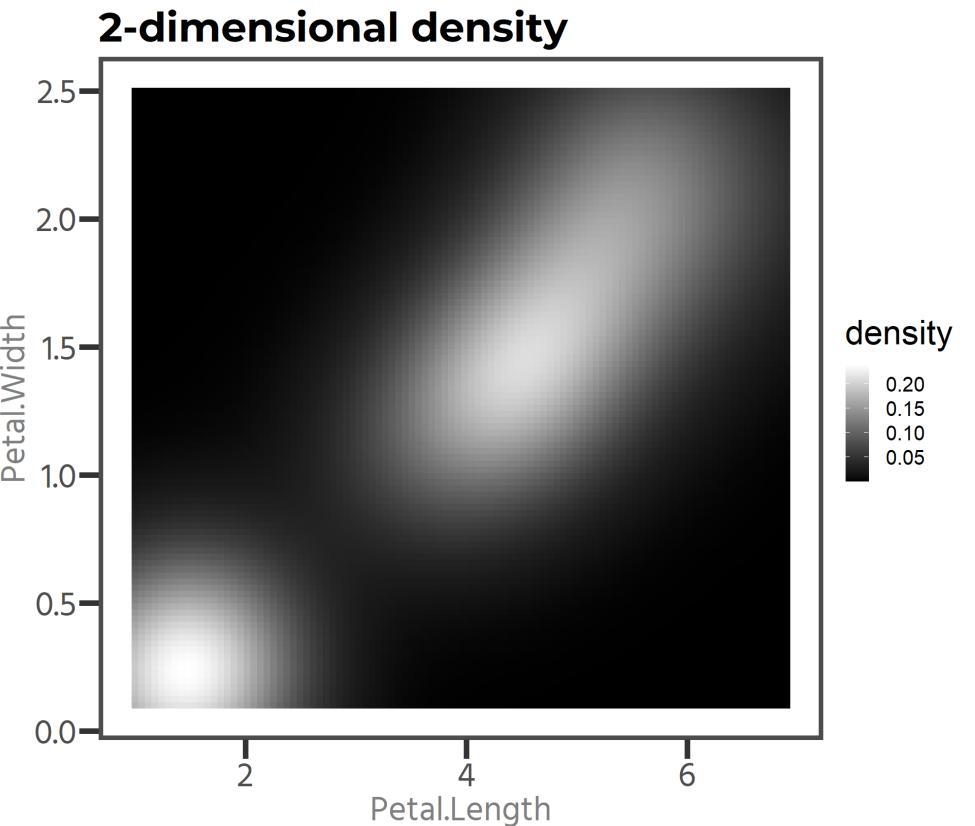
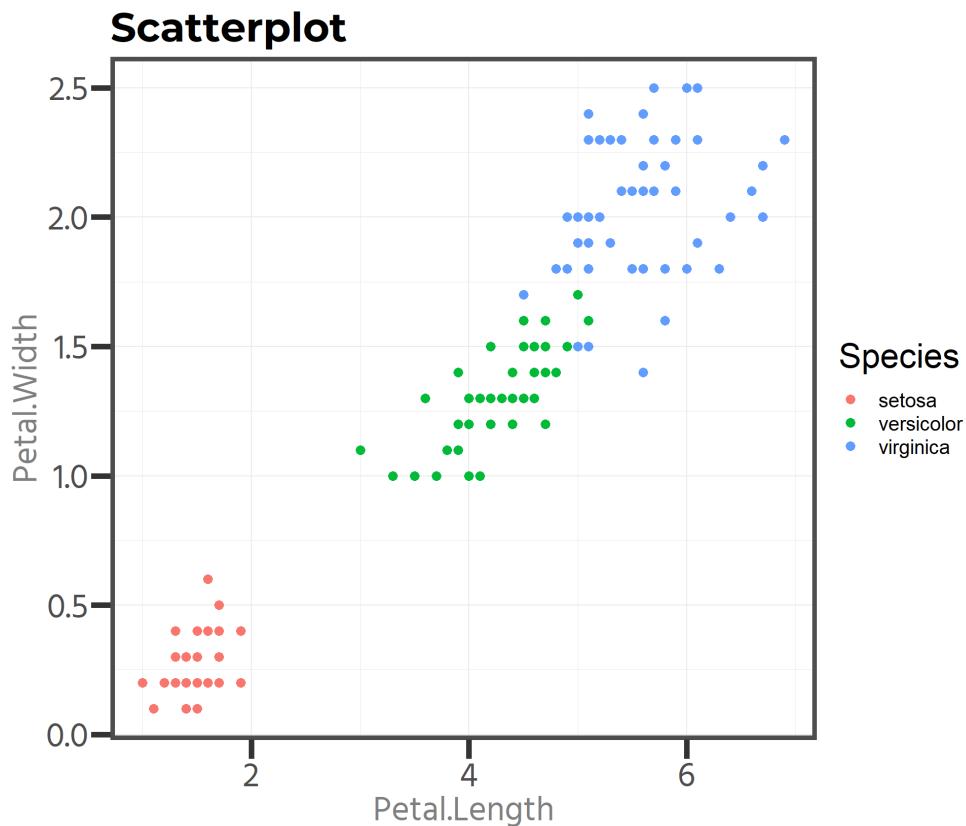
somewhat

yes

Is it rude to recline?

```
1 ggmosaic::fly %>%
2   count(rude_to_recline, do_you_recline, .drop = FALSE) %
3   ggplot(.) +
4   geom_tile(aes(x = rude_to_recline,
5                 y = do_you_recline,
6                 fill = n)) +
7   scale_fill_gradient(low = 'navy', high = "yellow")
```

Multiple continuous variables



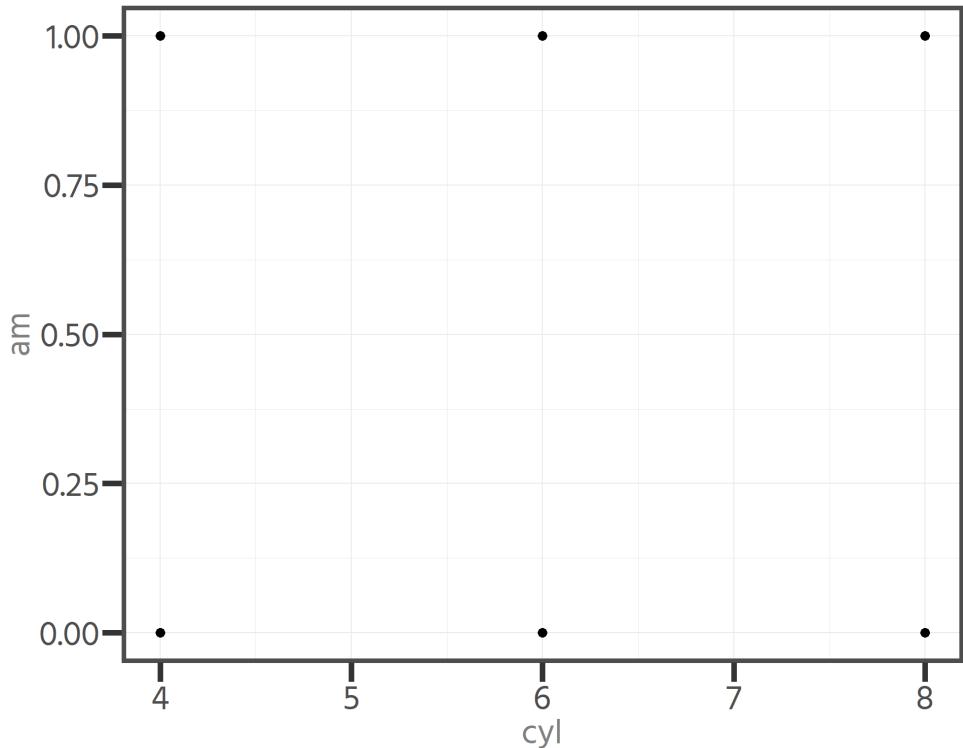
```
1 ggplot(iris) +  
2   geom_point(aes(x=Petal.Length,  
3                 y=Petal.Width,  
4                 color=Species))
```

```
1 ggplot(iris, aes(x=Petal.Length,y=Petal.Width)) +  
2   stat_density2d(aes(fill = ..density..),  
3                   geom = "raster", contour = FALSE) +  
4   scale_fill_gradient(low = "black", high = "white")
```

Hybrid (some categorical, some continuous)

Overplotting

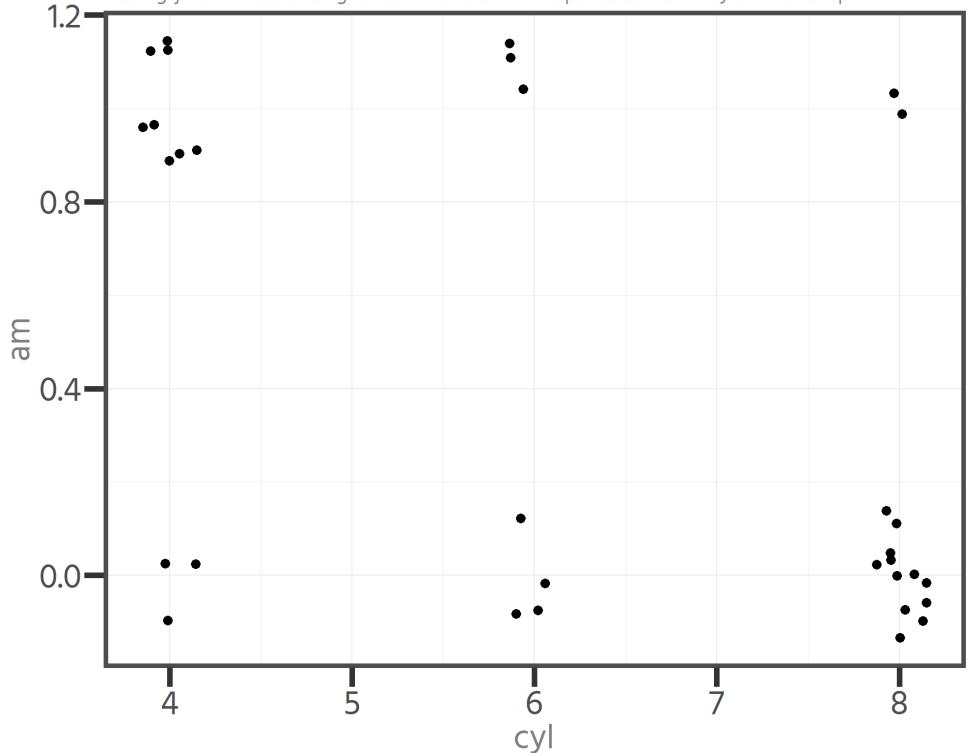
Overplotting is when data or labels overlap, meaning that the viewer loses important information



```
1 ggplot(mtcars) +  
2   geom_point(aes(x=cyl, y=am))
```

Jitterplot

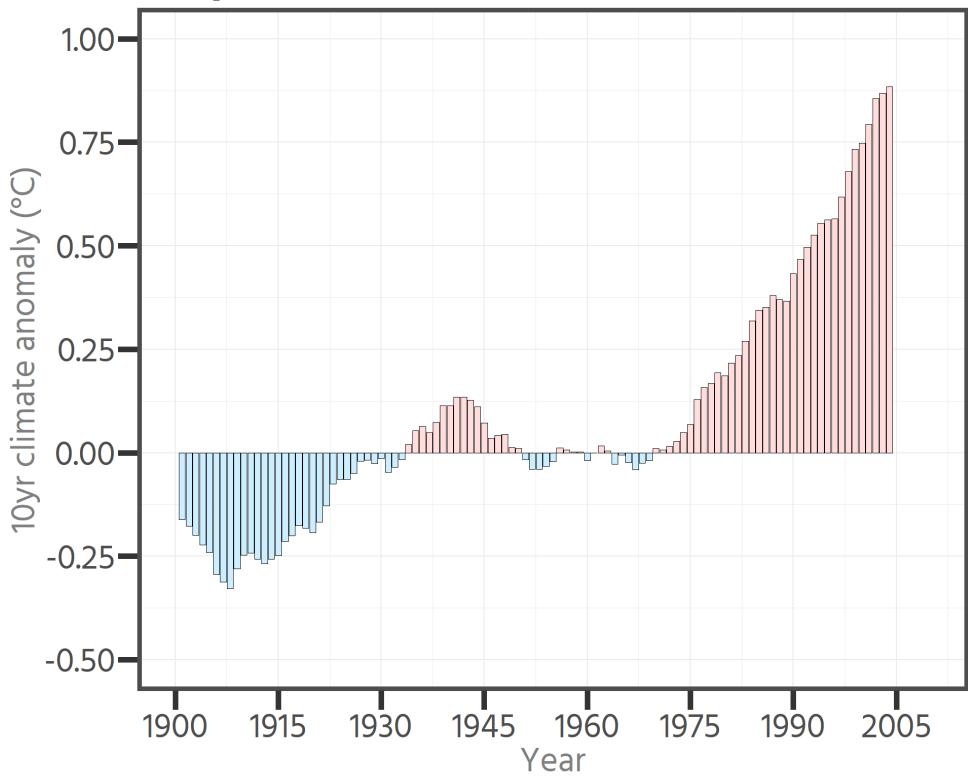
Adding 'jitter' means adding random variation to the points so that they don't overlap



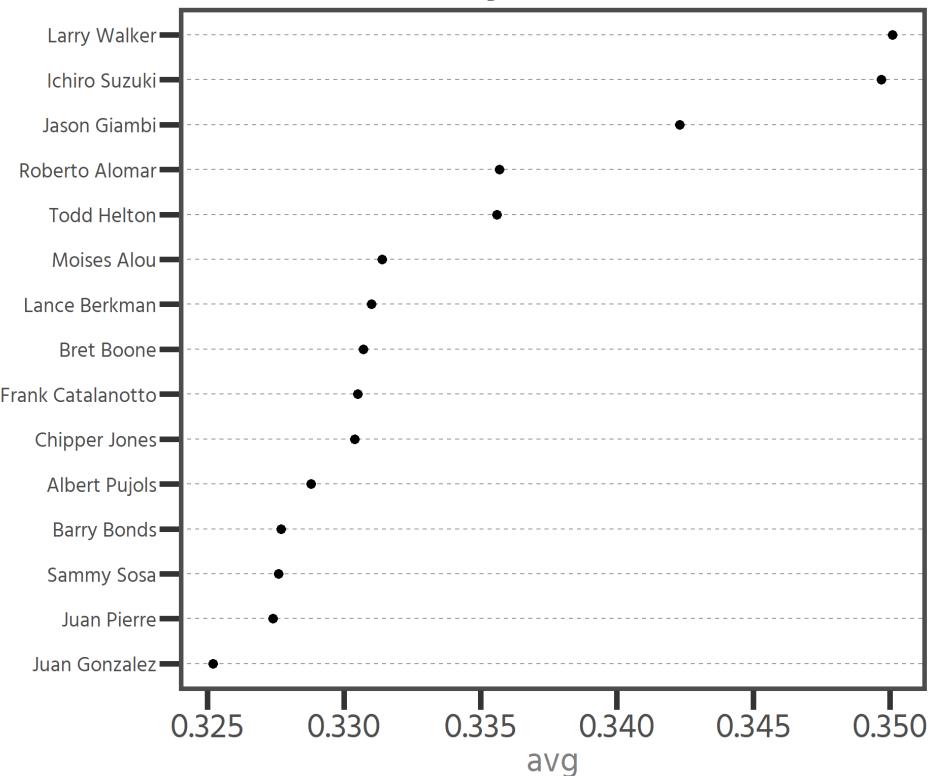
```
1 ggplot(mtcars) +  
2   geom_jitter(aes(x=cyl, y=am))  
3  
4 # You can also specify 'width' and 'height'  
5 # as additional arguments to refine the  
6 # jitter. Remember, jitter is random:  
7 # jitter plots will change whenever you  
8 # re-run the code.
```

Hybrid II

Barplot with +/- indicator



Cleveland dot plot



```
1 library(gcookbook)
2 climate_sub <- climate %>%
3   filter(Source == "Berkeley" & Year >= 1900) %>%
4   mutate(pos = Anomaly10y >= 0)
5
6 ggplot(climate_sub, aes(x = Year, y = Anomaly10y, fill =
7   geom_col(position = "identity", colour = "black", size
8   scale_fill_manual(values = c("#CCEEFF", "#FFDDDD")), gui
```

```
1 library(gcookbook)
2 tophit <- tophitters2001[1:15, ]
3
4 ggplot(tophit) +
5   geom_point(aes(x = avg, y = reorder(name, avg)))
6
7 # The `reorder` function is helpful
8 # when making plots of a non-binary categorical
9 # variable and a corresponding continuous variable.
```

There is no “one size fits all”

Data viz is creative ...

and creativity requires inspiration.

Check out these cool resources:

- R graphics gallery
- Nightingale: The Magazine of the Data Visualization Society
- Daydreaming Numbers Blog

Data viz is also programming ...

and programming requires code.

Check out these cool resources:

- R Graphics Cookbook
- R Graph Gallery

Data Visualization with ggplot2

Session 4: Making Good Visualizations

Nathan Barron

 nathanbarron@ou.edu

4 Principles of Good Data Visualization

Know your data

- You can't visualize what you don't understand
- A deep knowledge of the data, its structure, and the insights to be drawn are essential to data visualization
- When preparing your visualization,
form follows function

Data for price of groceries since 1970

| year | item | unit | price | type |
|------|--------|--------|-------|-----------|
| 1970 | milk | gallon | 0.49 | namebrand |
| 1970 | eggs | dozen | 0.60 | namebrand |
| 1970 | butter | lb | 0.87 | namebrand |
| 1970 | flour | 5lbs | 0.59 | namebrand |
| 1970 | sugar | 5lbs | 0.67 | namebrand |
| 1971 | milk | gallon | 0.50 | namebrand |
| 1971 | eggs | dozen | 0.53 | namebrand |
| 1971 | butter | lb | 0.88 | namebrand |
| 1971 | flour | 5lbs | 0.60 | namebrand |
| 1971 | sugar | 5lbs | 0.79 | namebrand |

Sketch

The **creative** and **programming** aspects of data visualization require different parts of your brain.

1. Start with *What do I want?*
2. Draw it
3. Follow up with *How do I get that?*
4. Code it!

Visualize the price of groceries since 1970

Simple Visuals

Don't try to include too much or irrelevant information.

Do provide detailed captions that offer additional context and information to the graphic

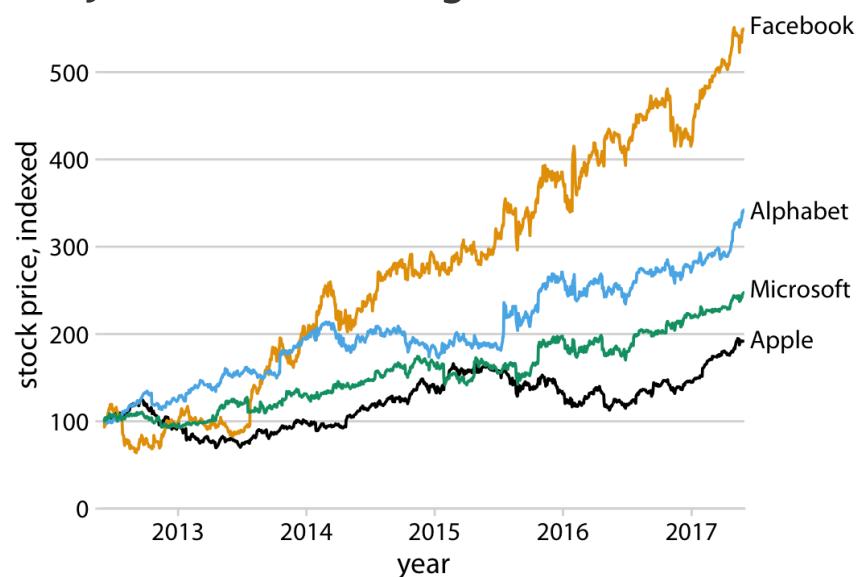
Use Color Effectively

- Only use colors when:
 - There are fewer than 6 categories; or
 - Visualizing continuous data with color gradient
- Use monotonic scales
- Use accessible colors: [Color Blindness Simulator](#)
- Use colors and shading to highlight important information

[Color Palette Generator](#)

Legends

- If there is a clear visual ordering in your data, make sure to match it in the legend.
- Whenever possible, design your figures so they don't need a legend.



Tell a story well

Don't try to tell too much in a single visualization

Do know *what* you want to convey

Do design for the 'aha' moment

Code-along #1

```
library(ggplot2)

# 1. Plotting layers ----
## 1.1 Base layer ----
ggplot(mtcars)

## 1.2 Adding geom and scale layers ----
ggplot(mtcars) +
  geom_point(aes(x=wt, y=mpg))

## 1.3 Adding title annotations ----
ggplot(mtcars) +
  geom_point(aes(x=wt, y=mpg)) +
  labs(x='Car weight (1000 lbs)',
       y='Miles per (US) gallon',
       title = 'Is car weight related to fuel efficiency?',
       subtitle = 'Car weight is measured in 1000 lbs. Fuel efficiency is measured in miles per (US) gallon.',
       caption = 'Data come from the mtcars dataset.')

## 1.4 Adjusting the theme layer ----
ggplot(mtcars) +
  geom_point(aes(x=wt, y=mpg)) +
  labs(x='Car weight (1000 lbs)',
       y='Miles per (US) gallon',
       title = 'Is car weight related to fuel efficiency?',
       subtitle = 'Car weight is measured in 1000 lbs. Fuel efficiency is measured in miles per (US) gallon.',
       caption = 'Data come from the mtcars dataset.') +
  theme(axis.title = element_text(color = 'grey50'),
        plot.title = element_text(face = 'bold'),
        plot.subtitle = element_text(color = 'grey50'),
        plot.caption = element_text(face = 'italic'))

## 1.5 Adding another geom layer (line of best fit) ----
ggplot(mtcars) +
  geom_point(aes(x=wt, y=mpg)) +
  labs(x='Car weight (1000 lbs)',
       y='Miles per (US) gallon',
       title = 'Is car weight related to fuel efficiency?',
       subtitle = 'Car weight is measured in 1000 lbs. Fuel efficiency is measured in miles per (US) gallon.',
       caption = 'Data come from the mtcars dataset.') +
  theme(axis.title = element_text(color = 'grey50'),
        plot.title = element_text(face = 'bold'),
        plot.subtitle = element_text(color = 'grey50'),
        plot.caption = element_text(face = 'italic')) +
  geom_smooth(aes(x=wt, y=mpg),
              method = 'lm')

## 1.6 Adding another geom layer (text annotation) ----
ggplot(mtcars) +
  geom_point(aes(x=wt, y=mpg)) +
  labs(x='Car weight (1000 lbs)',
       y='Miles per (US) gallon',
       title = 'Is car weight related to fuel efficiency?',
       subtitle = 'Car weight is measured in 1000 lbs. Fuel efficiency is measured in miles per (US) gallon.',
       caption = 'Data come from the mtcars dataset.') +
```

```

theme(axis.title = element_text(color = 'grey50'),
      plot.title = element_text(face = 'bold'),
      plot.subtitle = element_text(color = 'grey50'),
      plot.caption = element_text(face = 'italic')) +
geom_smooth(aes(x=wt, y=mpg),
            method = 'lm') +
annotate("text", x=4.5, y=25, label = 'NOTE')

# 2. Aesthetics ----
## 2.1 Example of Global Aesthetics ----
ggplot(mpg) +
  geom_point(aes(displ, hwy),
             color = 'red')

## 2.2 Example of Mapped Aesthetics ----
ggplot(mpg) +
  geom_point(aes(displ, hwy, color = class))

# 3. Linetypes and shapes ----
## 3.1 Linetype as global aesthetic ----
ggplot(mtcars) +geom_smooth(aes(x=wt, y=mpg), method = 'lm', linetype = 2)

## 3.2 Linetype as mapped aesthetic ----
ggplot(mtcars) +geom_smooth(aes(x=wt, y=mpg, linetype=as.factor(cyl)), method = 'lm')

## 3.3 Shape as global aesthetic ----
ggplot(mtcars) +geom_point(aes(x=wt, y=mpg), shape = 2)

## 3.4 Shape as mapped aesthetic ----
ggplot(mtcars) +geom_point(aes(x=wt, y=mpg, shape = as.factor(cyl)))

# 4. Scales ----
# 4.1 Adjusting axis scales (continuous) ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Species)) +
  scale_x_continuous(limits=c(0,7), breaks = seq(1,7,1)) +
  scale_y_continuous(limits=c(0,7), breaks = seq(0,7,1))

# 4.2 Adjusting axis scales (discrete) ----
ggplot(iris) +
  geom_boxplot(aes(Species,Petal.Length)) +
  scale_x_discrete(limits = c('virginica','versicolor','setosa'),
                   labels = c('VIRGINICA','VERSICOLOR','SETOSA'))

# Caution: In many cases, the ordering of discrete values on the X axis
#           is better manipulated by factoring the variable while cleaning
#           prior to plotting.

# 4.3 Adjusting color scales (continuous) ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Species))

### 4.3a Low/High Gradient ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Petal.Width)) +

```

```

scale_color_gradient(low = 'grey40', high = 'orange')

### 4.3b Low/High Steps ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Petal.Width)) +
  scale_color_steps(low = 'grey40', high = 'orange', n.breaks = 3)

### 4.3c Color Brewer Palettes ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Petal.Width)) +
  scale_color_fermenter(palette = 'Set2')

# Use scale_color_brewer() for gradient.
# Use scale_color_fermenter() for steps.
# See more palettes: https://r-graph-gallery.com/38-rcolorbrewers-palettes.html

### 4.3d Viridis Palettes ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Petal.Width)) +
  scale_color_viridis_c()

# Use scale_color_viridis_c() for gradient.
# Use scale_color_viridis_b() for steps.
# See more palettes: https://cran.r-project.org/web/packages/viridis/vignettes/intro-to-viridis.html

# 4.4 Adjusting color scales (discrete) ----
### 4.4a Manually ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Species)) +
  scale_color_manual(values = c('grey40','firebrick','orange'))

### 4.4b Greyscale ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Species)) +
  scale_color_grey()

### 4.4c Color Brewer Palettes ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Species)) +
  scale_color_brewer(palette = 'Set2')

# See more palettes: https://r-graph-gallery.com/38-rcolorbrewers-palettes.html

### 4.4d Viridis Palettes ----
ggplot(iris) +
  geom_point(aes(Petal.Length,Petal.Width, color=Species)) +
  scale_color_viridis_d(option = 'turbo')

# See more palettes: https://cran.r-project.org/web/packages/viridis/vignettes/intro-to-viridis.html

# 5. Faceting ----
## 5.1 Example of facet_wrap ----
ggplot(mtcars) +
  geom_point(aes(hp, mpg)) +
  facet_wrap(~as.factor(cyl))

```

```
## 5.2 Example of facet_grid ----  
ggplot(mtcars) +  
  geom_point(aes(hp, mpg)) +  
  facet_grid(as.factor(cyl) ~ as.factor(gear))
```

Code-along #2

```
library(ggplot2)

# 1. Complete themes ----
## 1.1 theme_grey() ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) + theme_grey()

## 1.2 theme_bw() ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) + theme_bw()

## 1.3 theme_linedraw() ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) + theme_linedraw()

## 1.4 theme_light() ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) + theme_light()

## 1.5 theme_dark() ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) + theme_dark()

## 1.6 theme_minimal() ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) + theme_minimal()

## 1.7 theme_classic() ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) + theme_classic()

## 1.7 theme_void() ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) + theme_void()

# 2. Plot elements ----
## 2.1 plot.background ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) +
  labs(title = 'Title', subtitle = 'Subtitle') +
  theme(plot.background = element_rect(fill='pink'))

## 2.2 plot.title and plot.subtitle ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) +
  labs(title = 'Title', subtitle = 'Subtitle',
       caption = 'Caption') +
  theme(plot.title = element_text(color='red', size = 25),
        plot.subtitle = element_text(color='grey40', face = 'italic'),
        plot.caption = element_text(color='grey40'))

# 3. Panel elements ----
## 3.1 panel.background ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) +
  labs(title = 'Title', subtitle = 'Subtitle',
       caption = 'Caption') +
  theme(panel.background = element_rect(fill = 'pink'))

## 3.2 panel.border ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) +
  labs(title = 'Title', subtitle = 'Subtitle',
       caption = 'Caption') +
  theme(panel.border = element_rect(color = 'red', fill=NA))
```

```

## 3.3 panel.grid ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) +
  labs(title = 'Title', subtitle = 'Subtitle',
       caption = 'Caption') +
  theme(panel.grid = element_line(color = 'red'))

# 4. Axis elements ----
## 4.1 axis.title ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) +
  labs(title = 'Title', subtitle = 'Subtitle',
       caption = 'Caption') +
  theme(axis.title = element_text(color = 'red'))

ggplot(mtcars) + geom_point(aes(wt,mpg)) +
  labs(title = 'Title', subtitle = 'Subtitle',
       caption = 'Caption') +
  theme(axis.title.x = element_text(color = 'forestgreen'),
        axis.title.y = element_text(color = 'blue'))

## 4.2 axis.text ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) +
  labs(title = 'Title', subtitle = 'Subtitle',
       caption = 'Caption') +
  theme(axis.text = element_text(size=25, angle = 40),
        axis.text.y = element_text(color = 'forestgreen'))

## 4.3 axis.ticks and axis.ticks.length ----
ggplot(mtcars) + geom_point(aes(wt,mpg)) +
  labs(title = 'Title', subtitle = 'Subtitle',
       caption = 'Caption') +
  theme(axis.ticks = element_line(linewidth=8),
        axis.ticks.length = unit(18, 'points'))

# 5. Fonts ----
## 5.1 For Windows users ----
install.packages('extrafont')
library('extrafont')

font_import()
# You will be prompted to continue [y/n]
# Type 'y' and press enter

loadfonts(device="win")

## 5.2 For Mac users ----
install.packages('extrafont')
library('extrafont')

font_import()
# You will be prompted to continue [y/n]
# Type 'y' and press enter

loadfonts()

# 5.3 Check which fonts are available ----
fonts()

# 6. Custom theme ----

```

```
# You have a pre-theme plot saved as 'existing_plot'
existing_plot <- ggplot(mtcars) + geom_point(aes(wt,mpg)) + labs(title =
'TITLE',subtitle='Subtitle')

# Save your theme settings as an object ('my_theme')
my_theme <- theme(plot.title = element_text(size=25, family = 'Urbanist', face = 'bold',
color = 'blue'),
                    plot.subtitle = element_text(size=15, family = 'Urbanist', face =
'italic', color = 'grey40'),
                    axis.title = element_text(size = 10, family = 'Urbanist'),
                    axis.text = element_text(size = 8, family = 'Urbanist'),
)

# Add your custom theme to the existing plot
existing_plot + my_theme
```

Code-along #3

```
library(ggplot2)

# 1. Visualizing a single variable ----
## 1.1 Binary varialbes ----
### 1.1a Barplot (count) ----
ggplot(mtcars) + geom_bar(aes(x=vs))

### 1.1b Barplot (proportion) ----
ggplot(mtcars) + geom_bar(aes(x=vs, y=..count../sum(..count..)))

## 1.2 Non-binary categorical ----
### 1.2a Waffle plot ----
# install.packages("waffle", repos = "https://cinc.rud.is")
library(waffle)
library(dplyr)

iris %>%
  count(Species) %>%
  ggplot(aes(fill = Species, values = n)) +
  geom_waffle(size = 1,
              colour = "white",
              na.rm=TRUE,
              flip = TRUE,
              make_proportional = TRUE) +
  theme_void() +
  coord_equal()

## 1.3 Continuous ----
### 1.3a Histogram ----
ggplot(iris) + geom_histogram(aes(x=Sepal.Length))

### 1.3b Density ----
ggplot(iris) + geom_density(aes(x=Sepal.Length))

# 2. Visualizing multiple variables ----
## 2.1 Multiple categorical variables ----
### 2.1a Barplot (proportion) ----
mtcars %>%
  mutate(vs = factor(vs), am = factor(am)) %>%
  ggplot(.) +
  geom_bar( aes(x=vs,
                 y=..count../sum(..count..),
                 group=am,
                 fill=am),
            position = 'dodge')

### 2.1b Mosaic plot ----
# library(ggmosaic)
# library(tidyr)

ggplot(ggmosaic::fly) +
  geom_mosaic(aes(x = product(do_you_recline, rude_to_recline), fill=do_you_recline)) +
  theme(panel.grid = element_blank(),
        legend.position = 'none')

### 2.1c Categorical heatmap ----
ggmosaic::fly %>%
```

```

count(rude_to_recline, do_you_recline, .drop = FALSE) %>%
ggplot(.) +
geom_tile(aes(x = rude_to_recline,
              y = do_you_recline,
              fill = n)) +
scale_fill_gradient(low = 'navy', high = "yellow")

## 2.2 Multiple continuous variables ----
### 2.2a Scatterplot ----
ggplot(iris) +
  geom_point(aes(x=Petal.Length,
                 y=Petal.Width,
                 color=Species))

### 2.2b Two-dimensional density ----
ggplot(iris, aes(x=Petal.Length,y=Petal.Width)) +
  stat_density2d(aes(fill = ..density..),
                 geom = "raster", contour = FALSE) +
  scale_fill_gradient(low = "black", high = "white")

## 2.3 Hybrid ----
### 2.3a Jitterplot ----
ggplot(mtcars) +
  geom_jitter(aes(x=cyl,y=am))

# You can also specify 'width' and 'height'
# as additional arguments to refine the
# jitter. Remember, jitter is random:
# jitter plots will change whenever you
# re-run the code.

### 2.3b Barplot with +/- indicator ----
library(gcookbook)
climate_sub <- climate %>%
  filter(Source == "Berkeley" & Year >= 1900) %>%
  mutate(pos = Anomaly10y >= 0)

ggplot(climate_sub, aes(x = Year, y = Anomaly10y, fill = pos)) +
  geom_col(position = "identity", colour = "black", size = 0.25) +
  scale_fill_manual(values = c("#CCEEFF", "#FFDDDD"), guide = 'none')

### 2.3c Cleveland dot plot ----
library(gcookbook)
tophit <- tophitters2001[1:15, ]

ggplot(tophit) +
  geom_point(aes(x = avg, y = reorder(name, avg)))

# The `reorder` function is helpful
# when making plots of a non-binary categorical
# variable and a corresponding continuous variable.

```

Data Visualization with ggplot2 in R

Activity #1: Creating Base Plots

0.0 Getting Started

Install and load the **palmerpenguins** package. You should notice that you now have access to the `penguins` dataset, even though it does not appear in the global environment.

```
install.packages("palmerpenguins")
```

```
library(ggplot2)
library(palmerpenguins)
head(penguins)
```

```
# A tibble: 6 × 8
  species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
  <fct>   <fct>        <dbl>          <dbl>            <int>        <int>
1 Adelie  Torgersen     39.1          18.7            181        3750
2 Adelie  Torgersen     39.5          17.4            186        3800
3 Adelie  Torgersen     40.3           18             195        3250
4 Adelie  Torgersen      NA            NA              NA         NA
5 Adelie  Torgersen     36.7          19.3            193        3450
6 Adelie  Torgersen     39.3          20.6            190        3650
# ℹ 2 more variables: sex <fct>, year <int>
```

1.0 Creating a Base Plot

Using the `penguins` dataset, create a scatterplot (using `geom_point()`) of `flipper_length_mm` (Y axis) by `body_mass_g` (X axis). Your plot should include *each* of the following:

- Title
- Subtitle
- X-axis title
- Y-axis title

NOTE: Reference the code available to you in the slides or the code-along .R file.

2.0 Adding a Layer

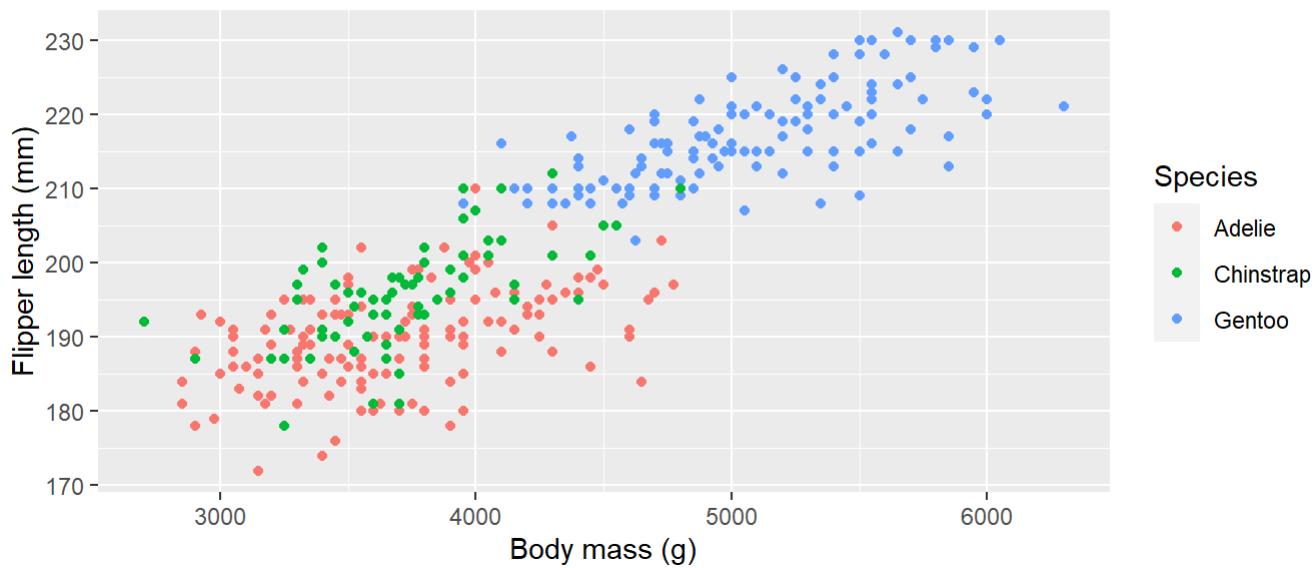
Add a layer to your base plot with any *one* of the following:

- Mapped aesthetic
- Scale change (axis, color, size, etc.)
- Facet

3.0 Examples

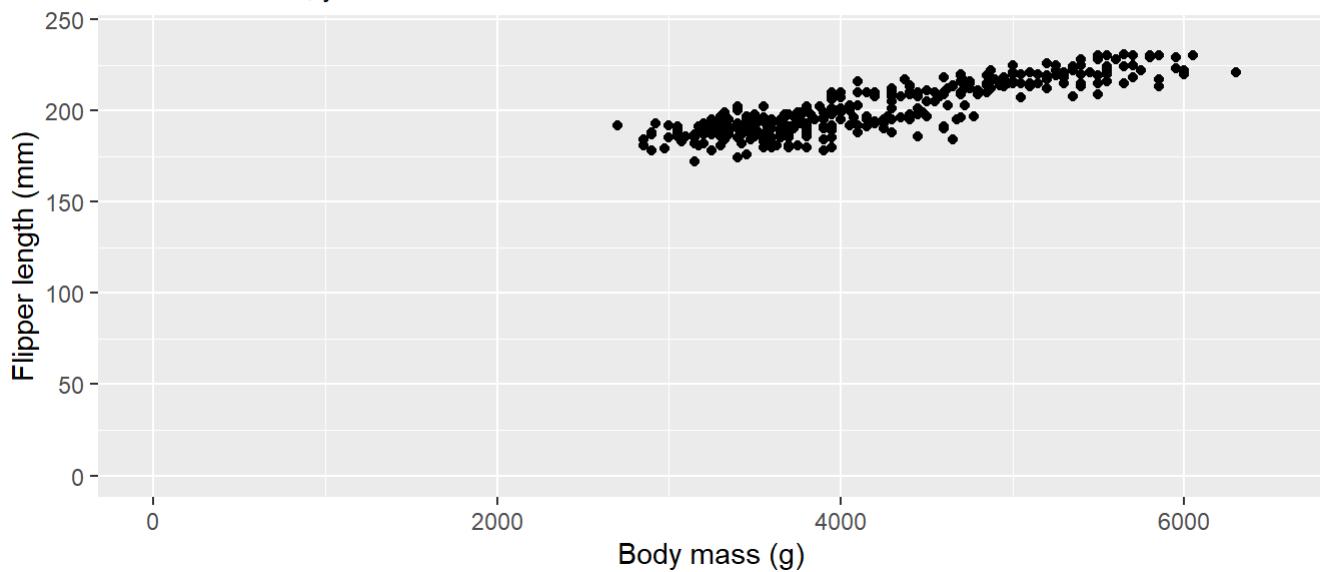
Flipper Length by Body Mass in Penguins

Species type in color



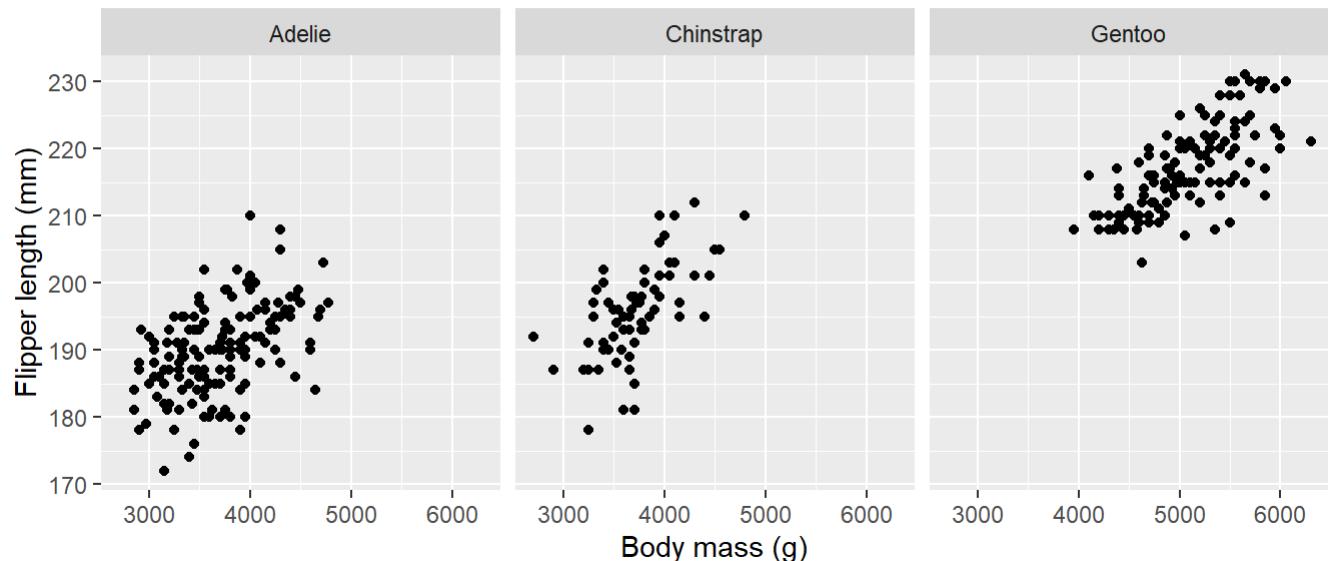
Flipper Length by Body Mass in Penguins

Rescaled to $x=0$, $y=0$



Flipper Length by Body Mass in Penguins

faceted by species



Data Visualization with ggplot2 in R

Activity #2: Customizing Themes

0.0 Getting Started

Load the appropriate packages. Recreate the base plot you generated in Activity #1. Store the plot as `plot`.

```
library(ggplot2)
library(palmerpenguins)
```

1.0 Using pre-packaged themes

Add a pre-packaged theme to your base plot:

- `theme_grey()`
- `theme_bw()`
- `theme_linedraw()`
- `theme_light()`
- `theme_dark()`
- `theme_minimal()`
- `theme_classic()`

2.0 Customizing your own theme

2.1 Add fonts

Add fonts that R can use for customizing plots.

```
##### For Windows Users #####
install.packages('extrafont')
library('extrafont')

font_import()
# You will be prompted to continue [y/n]
# Type 'y' and press enter

loadfonts(device="win")
```

```
##### For Mac Users #####
install.packages('extrafont')
library('extrafont')
```

```
font_import()  
# You will be prompted to continue [y/n]  
# Type 'y' and press enter  
  
loadfonts()
```

Check what fonts are available with `fonts()`.

2.2 Customizing theme elements

Using the `theme()` function, customize the following theme elements:

- `plot.title`
- `plot.subtitle`
- `panel.background`
- `panel.border`
- `panel.background`
- `panel.grid`
- `axis.title`
- `axis.text`

You should use some combination of the following adjustments in your theme:

- `color`
- `fill`
- `size`
- `linewidth`
- `family`
- `face` (bold, italic, etc.)

Be creative. Effective data communication requires important information to be interpretable *and* visually inviting.

2.3 Saving your theme

Store your theme as `my_theme`.

2.4 Reusing your theme

Run the following code:

```
plot + my_theme
```

3.0 Examples

Flipper Length by Body Mass in Penguins

Species type in color



Data Visualization with ggplot2 in R

Activity #3: Creating and Customizing Complex Graphics

0.0 Getting Started

Load the appropriate packages.

```
library(ggplot2)  
library(palmerpenguins)
```

1.0 Visualizing Data, from start to finish

Choose one of the following challenges. You must perform the following:

1. Decide which type of graphic is most appropriate for the graphic.
2. Hand sketch the desired plot.
3. Determine what additional features/aesthetics are required.
4. Build the base plot.
5. Customize the theme.

There is no single correct approach, though you should be mindful of the guidelines discussed during the workshop. You will have an opportunity to present your graphic and hear feedback from the instructors and fellow workshop participants.

Alternatively, you can create a visualization using your own data. You will still have an opportunity to present and receive feedback.

Challenge 1 (Difficulty : Low)

Plot how many penguins were studied by island and species.

Challenge 2 (Difficulty : Med.)

Show the relationship between body mass (x-axis) and bill length and bill depth (both y-axis) for each species. Provide both the line of best fit and the underlying data points.

Challenge 3 (Difficulty : High)

Plot the change in the following for each species from 2007 to 2009:

- bill_length_mm
- bill_depth_mm
- flipper_length_mm
- body_mass_g