Data Visualization with ggplot2

John Tipton

Readings

- R for data science
  - Introduction
  - Chapters 1 (Data visualization with `ggplot2`) and 5 (Exploratory data analysis)

`data.frame` The first class object

- A `data.frame` is a rectangular collection of variables (columns) and observations (rows).
- Built in datasets can be explored with `data()`

```
data()
```

Example: Palmer Penguins

```
library(tidyverse)
library(palmerpenguins)
```

```
penguins
```

```
## # A tibble: 344 x 8
##    species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g sex     year
##    <fct>   <fct>              <dbl>         <dbl>             <int>       <int> <fct>  <int>
##  1 Adelie  Torgersen           39.1          18.7               181        3750 male    2007
##  2 Adelie  Torgersen           39.5          17.4               186        3800 female  2007
##  3 Adelie  Torgersen           40.3          18                 195        3250 female  2007
##  4 Adelie  Torgersen           NA            NA                  NA          NA <NA>    2007
##  5 Adelie  Torgersen           36.7          19.3               193        3450 female  2007
##  6 Adelie  Torgersen           39.3          20.6               190        3650 male    2007
##  7 Adelie  Torgersen           38.9          17.8               181        3625 female  2007
##  8 Adelie  Torgersen           39.2          19.6               195        4675 male    2007
##  9 Adelie  Torgersen           34.1          18.1               193        3475 <NA>    2007
## 10 Adelie  Torgersen           42            20.2               190        4250 <NA>    2007
## # … with 334 more rows
```

First ggplot

- start by exploring the data. Use the `str()` or `glimpse()` functions to explore the data

```
str(penguins)
```

```
## tibble[,8] [344 × 8] (S3: tbl_df/tbl/data.frame)
##  $ species          : Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ island           : Factor w/ 3 levels "Biscoe","Dream",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ bill_length_mm   : num [1:344] 39.1 39.5 40.3 NA 36.7 39.3 38.9 39.2 34.1 42 ...
##  $ bill_depth_mm    : num [1:344] 18.7 17.4 18 NA 19.3 20.6 17.8 19.6 18.1 20.2 ...
##  $ flipper_length_mm: int [1:344] 181 186 195 NA 193 190 181 195 193 190 ...
##  $ body_mass_g      : int [1:344] 3750 3800 3250 NA 3450 3650 3625 4675 3475 4250 ...
##  $ sex              : Factor w/ 2 levels "female","male": 2 1 1 NA 1 2 1 2 NA NA ...
##  $ year             : int [1:344] 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

```
glimpse(penguins)
```

```
## Rows: 344
## Columns: 8
## $ species           <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, A
## $ island            <fct> Torgersen, Torgersen, Torgersen, Torgersen, Torgersen, Torgersen, Torgers
## $ bill_length_mm    <dbl> 39.10000000000000142109, 39.50000000000000000000, 40.29999999999999715783
## $ bill_depth_mm     <dbl> 18.69999999999999928946, 17.39999999999999857891, 18.00000000000000000000
## $ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195, 193, 190, 186, 180, 182, 191, 198,
## $ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 3625, 4675, 3475, 4250, 3300, 3700, 320
## $ sex               <fct> male, female, female, NA, female, male, female, male, NA, NA, NA, NA, fem
## $ year              <int> 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2
```

flipper length and body mass

- Describe what we see

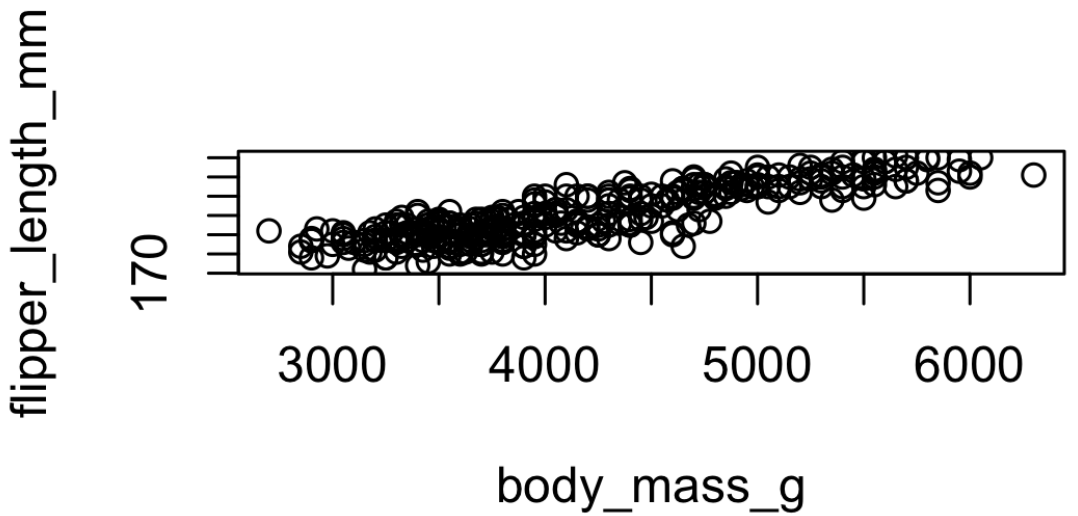
```
ggplot(data = penguins) +
    geom_point(mapping = aes(x = body_mass_g, y = flipper_length_mm))
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```

Plotting aesthetics

- ggplot is a lot of code for a simple plot!

```
plot(flipper_length_mm ~ body_mass_g, data = penguins)
```



- Wasn't that easier?

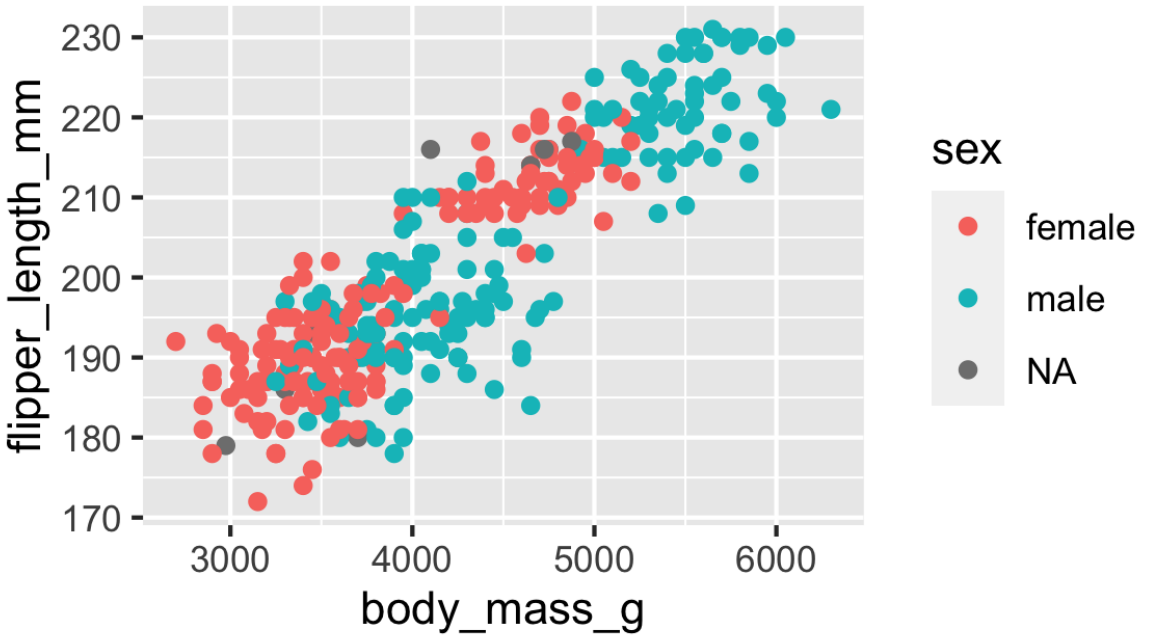
Aesthetics

- What if I want to color my plot by sex?
- What if I want to make my circles to have size relative to bill depth?
- What if...
- Use aesthetics.
    - Aesthetics map a characteristic given a variable.
    - Great for quick visually communication.

Using aesthetics

```
ggplot(data = penguins) +
    geom_point(mapping = aes(x = body_mass_g, y = flipper_length_mm,
                             color = sex))
```

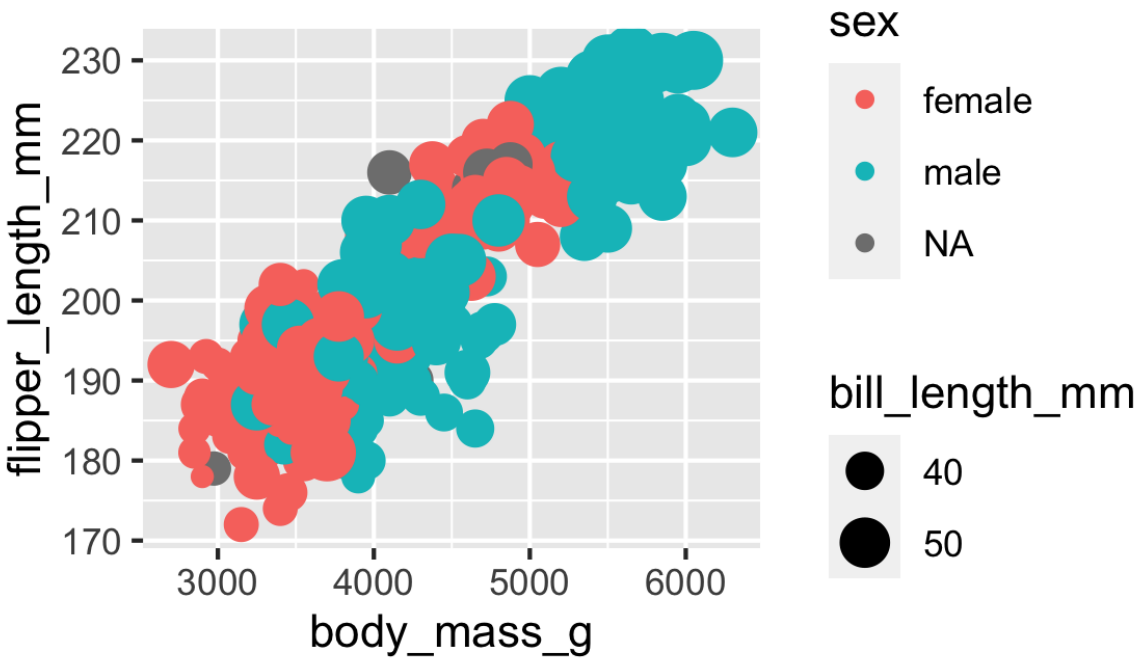
```
## Warning: Removed 2 rows containing missing values (geom_point).
```



Using aesthetics

```
ggplot(data = penguins) +
    geom_point(mapping = aes(x = body_mass_g, y = flipper_length_mm,
                             color = sex, size = bill_length_mm))
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



Available aesthetics

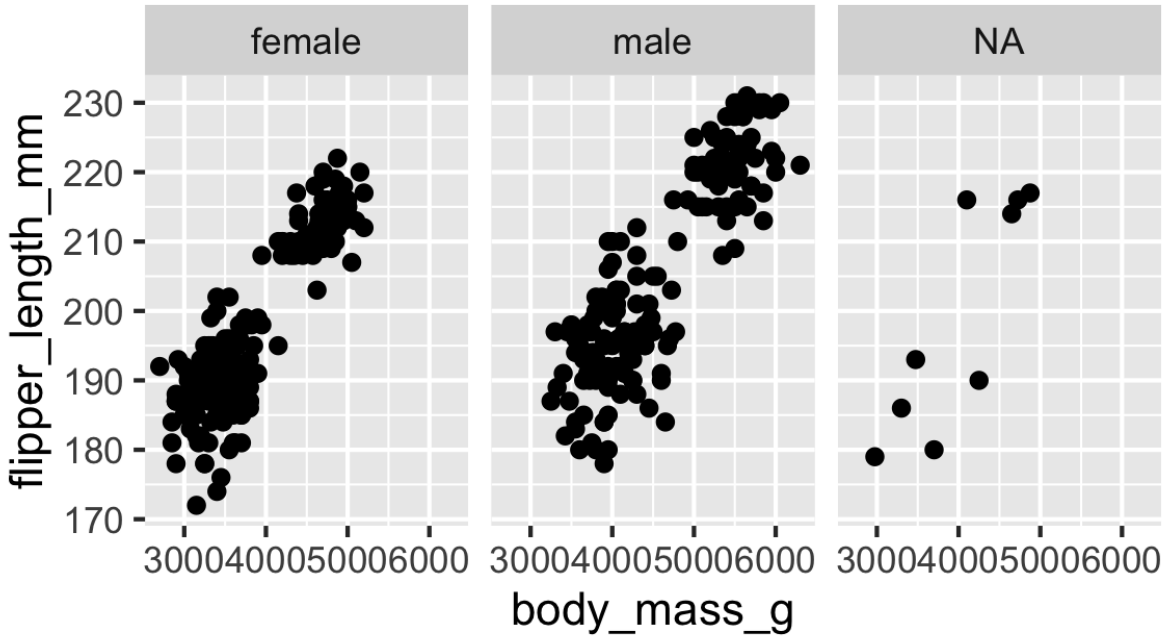
- size
- color
- fill
- shape
- transparency (`alpha`)

Facets

- What if I want to show multiple plots where each plot is determined by a variable?
- Facets.
  - `facet_wrap()` for a single variable.
  - `facet_grid()` for a grid of two variables.

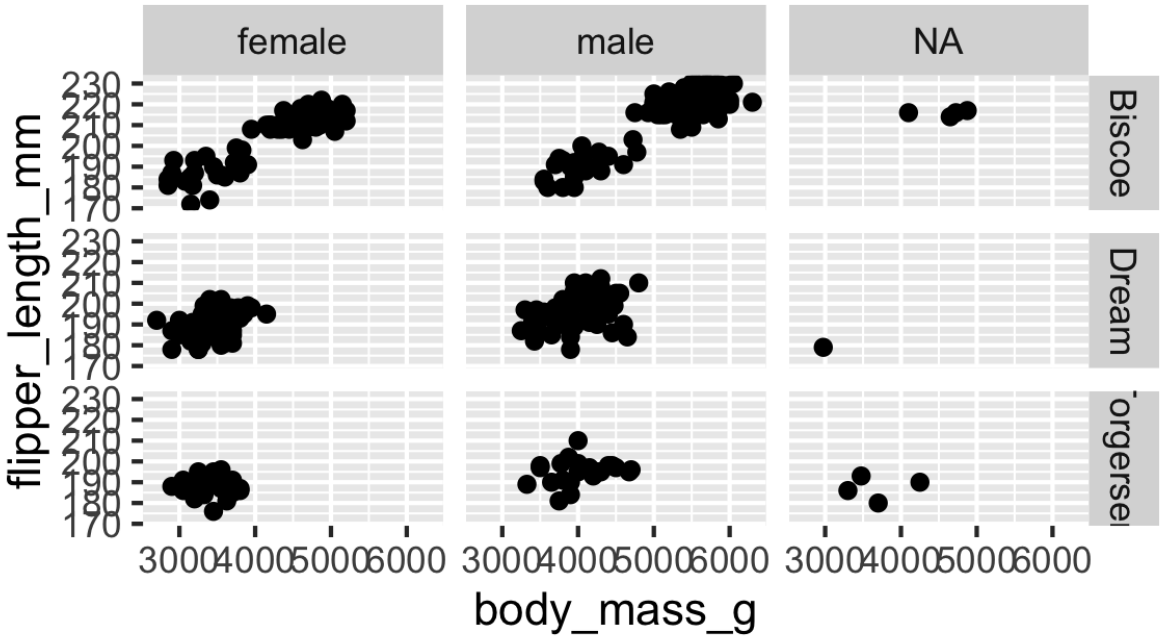
```
ggplot(data = penguins) +
    geom_point(mapping = aes(x = body_mass_g, y = flipper_length_mm)) +
    facet_wrap(~ sex)
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



```
ggplot(data = penguins) +
    geom_point(mapping = aes(x = body_mass_g, y = flipper_length_mm)) +
    facet_grid(island ~ sex)
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



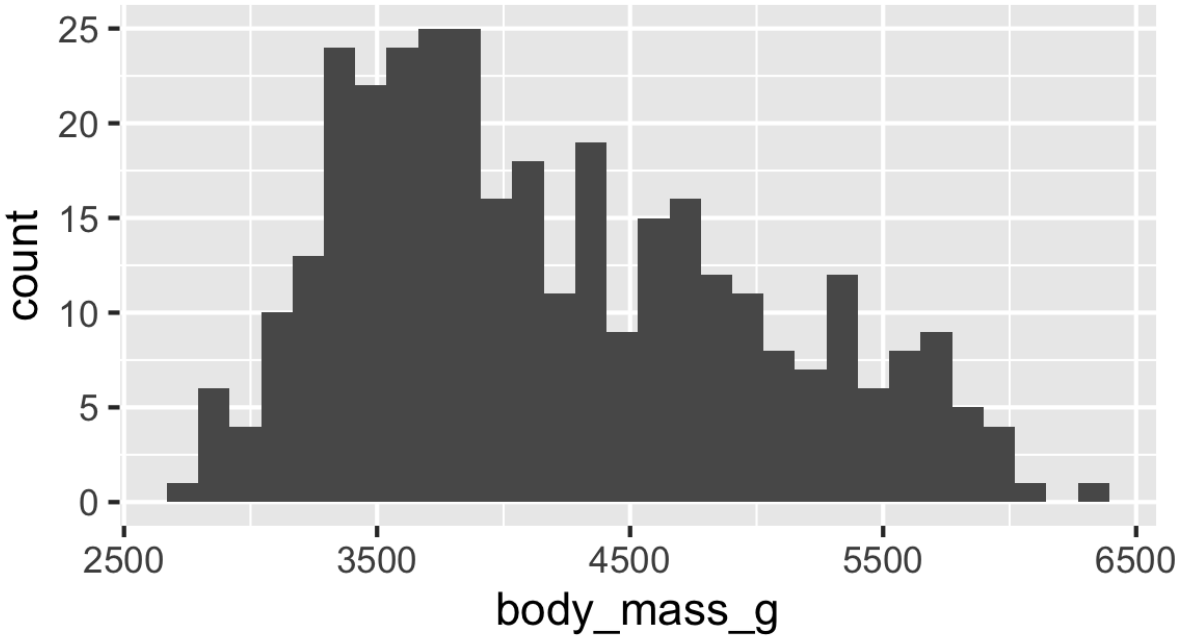
Geometries

- What kind of plot do you want?
  - histograms --`geom_histogram()`
  - scatterplot -- `geom_point()`
  - boxplots -- `geom_box()`
  - dotplot -- `geom_dotplot()`
  - line plots -- `geom_line()`
- Each `geom` has specific aesthetic requirements
- Use the help for specifics
  - `?geom_scatter`
- Many, many others -- virtually any kind of plot you want.
  - ggplot cheatsheet
- Many custom packages for specific plot types

```
ggplot(data = penguins) +
    geom_histogram(mapping = aes(x = body_mass_g))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

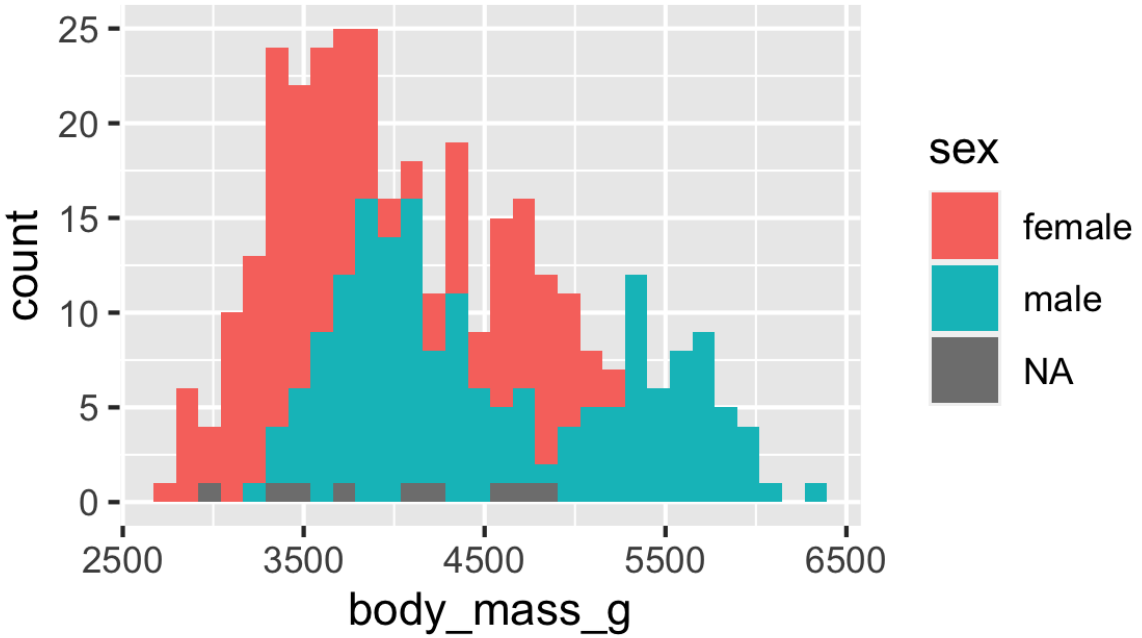
```
## Warning: Removed 2 rows containing non-finite values (stat_bin).
```



```
## what happens if you use color = sex?
ggplot(data = penguins) +
    geom_histogram(mapping = aes(x = body_mass_g, fill = sex))
```

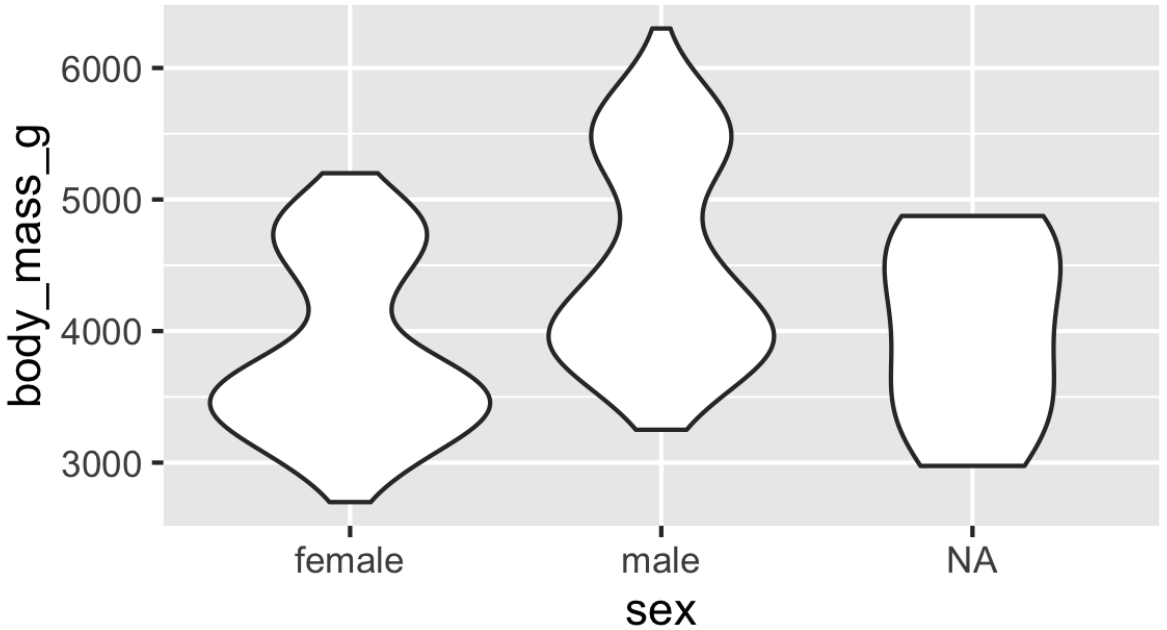
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 2 rows containing non-finite values (stat_bin).
```



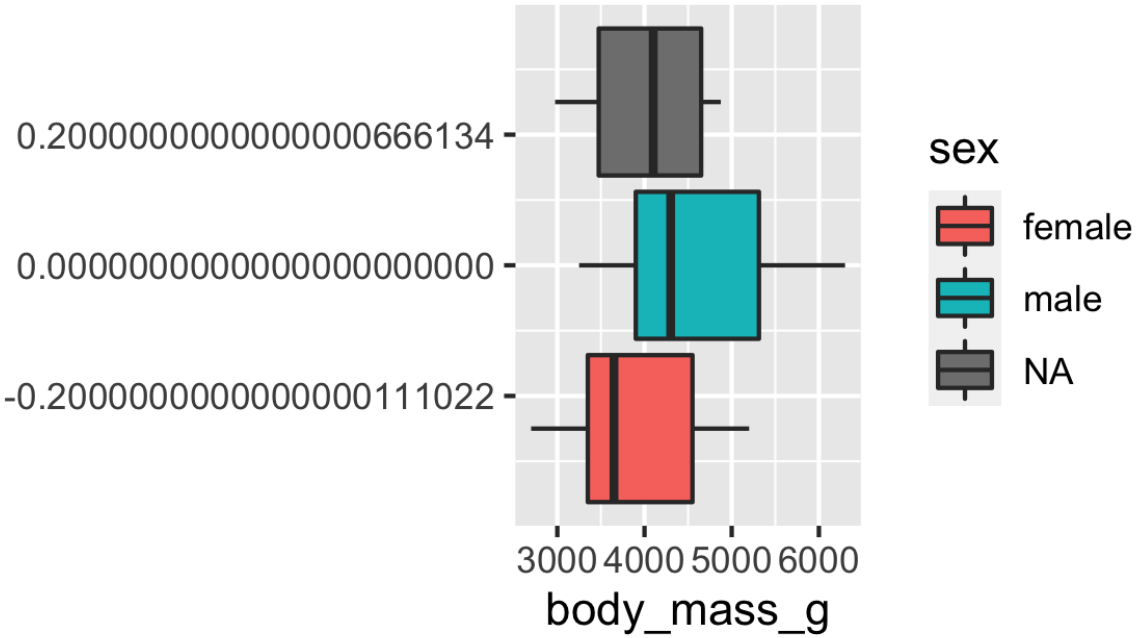
```
## smooth over the density
ggplot(data = penguins) +
    geom_violin(mapping = aes(y = body_mass_g, x = sex))
```

```
## Warning: Removed 2 rows containing non-finite values (stat_ydensity).
```



```
## what happens if you use color = sex?
ggplot(data = penguins) +
    geom_boxplot(mapping = aes(x = body_mass_g, fill = sex))
```

```
## Warning: Removed 2 rows containing non-finite values (stat_boxplot).
```



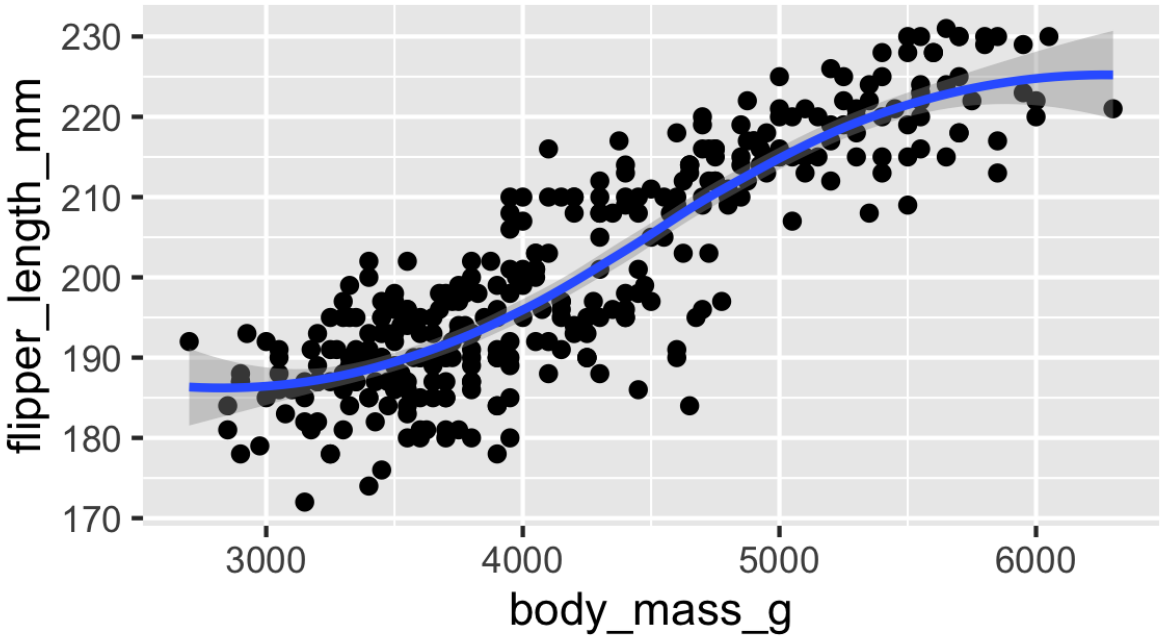
- You can even use multiple aesthetics

```
ggplot(data = penguins) +
    geom_point(mapping = aes(x = body_mass_g, y = flipper_length_mm)) +
    geom_smooth(mapping = aes(x = body_mass_g, y = flipper_length_mm))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## Warning: Removed 2 rows containing non-finite values (stat_smooth).

## Warning: Removed 2 rows containing missing values (geom_point).
```



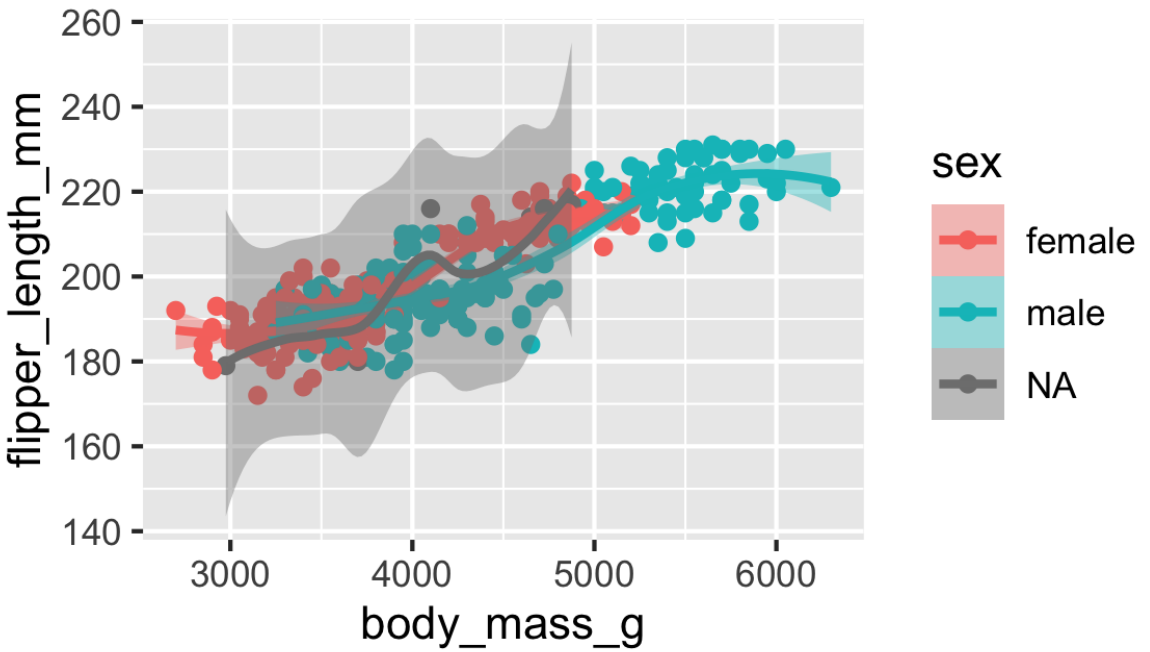
- geoms can have different aesthetics `aes()`.

```
ggplot(data = penguins) +
    geom_point(mapping = aes(x = body_mass_g, y = flipper_length_mm,
                             color = sex)) +
    geom_smooth(mapping = aes(x = body_mass_g, y = flipper_length_mm,
                              fill = sex, color = sex))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## Warning: Removed 2 rows containing non-finite values (stat_smooth).

## Warning: Removed 2 rows containing missing values (geom_point).
```



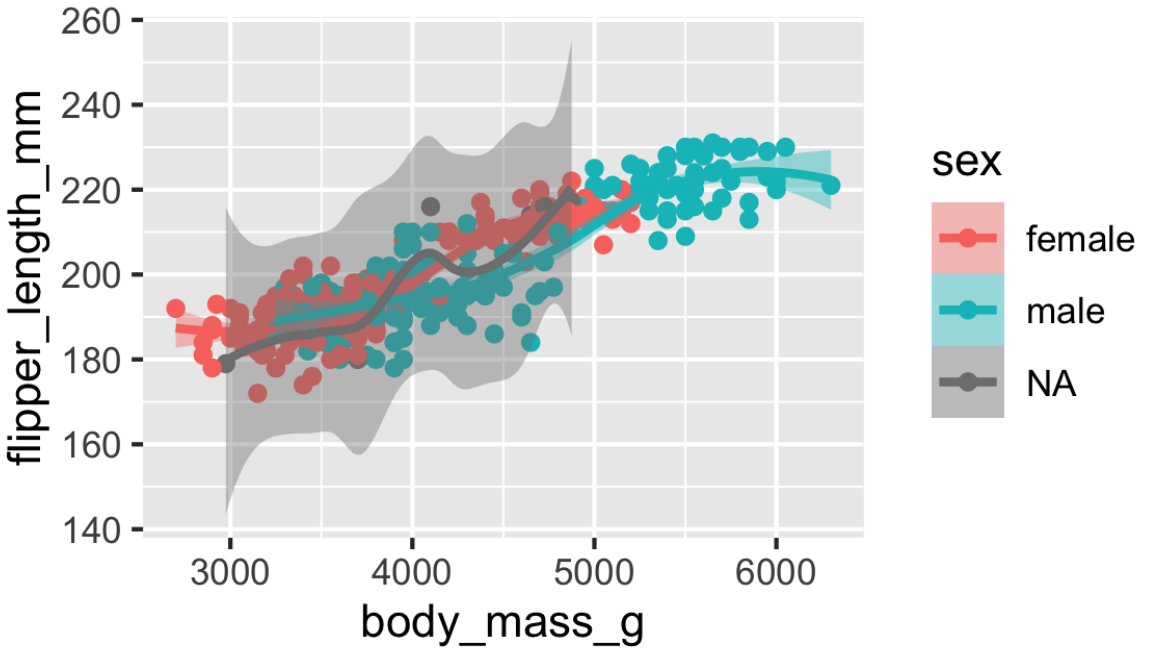
- Common aesthetics `aes()` can be added to the ggplot function.

```
ggplot(data = penguins,
       mapping = aes(x = body_mass_g, y = flipper_length_mm, color = sex)) +
    geom_point() +
    geom_smooth(mapping = aes(fill = sex))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## Warning: Removed 2 rows containing non-finite values (stat_smooth).

## Warning: Removed 2 rows containing missing values (geom_point).
```



Statistical transformations

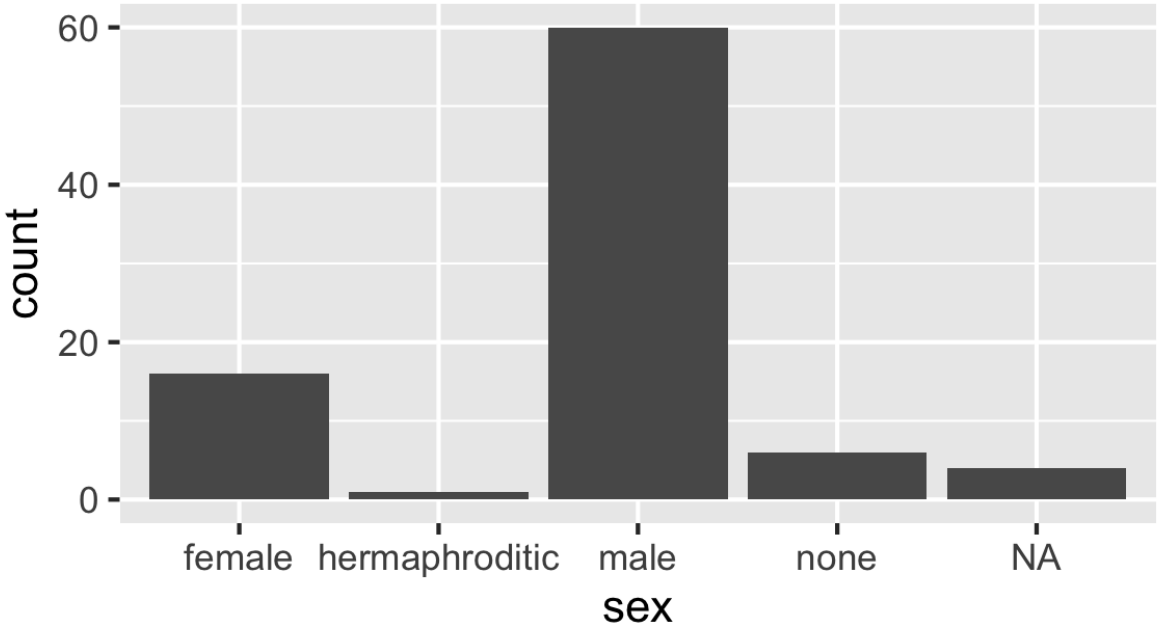
- `ggplot` can perform statistical transformations and fit basic models
- start with `starwars` data

```
data("starwars")
glimpse(starwars)
```

```
## Rows: 87
## Columns: 14
## $ name       <chr> "Luke Skywalker", "C-3PO", "R2-D2", "Darth Vader", "Leia Organa", "Owen Lars'
## $ height     <int> 172, 167, 96, 202, 150, 178, 165, 97, 183, 182, 188, 180, 228, 180, 173, 175,
## $ mass       <dbl> 77.000000000000000000, 75.000000000000000000, 32.000000000000000000, 13
## $ hair_color <chr> "blond", NA, NA, "none", "brown", "brown, grey", "brown", NA, "black", "aubur
## $ skin_color <chr> "fair", "gold", "white, blue", "white", "light", "light", "light", "white, re
## $ eye_color  <chr> "blue", "yellow", "red", "yellow", "brown", "blue", "blue", "red", "brown", '
## $ birth_year <dbl> 19.000000000000000000, 112.000000000000000000, 33.000000000000000000, 4
## $ sex        <chr> "male", "none", "none", "male", "female", "male", "female", "none", "male", '
## $ gender     <chr> "masculine", "masculine", "masculine", "masculine", "feminine", "masculine",
## $ homeworld  <chr> "Tatooine", "Tatooine", "Naboo", "Tatooine", "Alderaan", "Tatooine", "Tatooir
## $ species    <chr> "Human", "Droid", "Droid", "Human", "Human", "Human", "Human", "Droid", "Huma
## $ films      <list> <"The Empire Strikes Back", "Revenge of the Sith", "Return of the Jedi", "A
## $ vehicles   <list> <"Snowspeeder", "Imperial Speeder Bike">, <>, <>, <>, "Imperial Speeder Bike
## $ starships  <list> <"X-wing", "Imperial shuttle">, <>, <>, "TIE Advanced x1", <>, <>, <>, <>, '
```

Bar plot

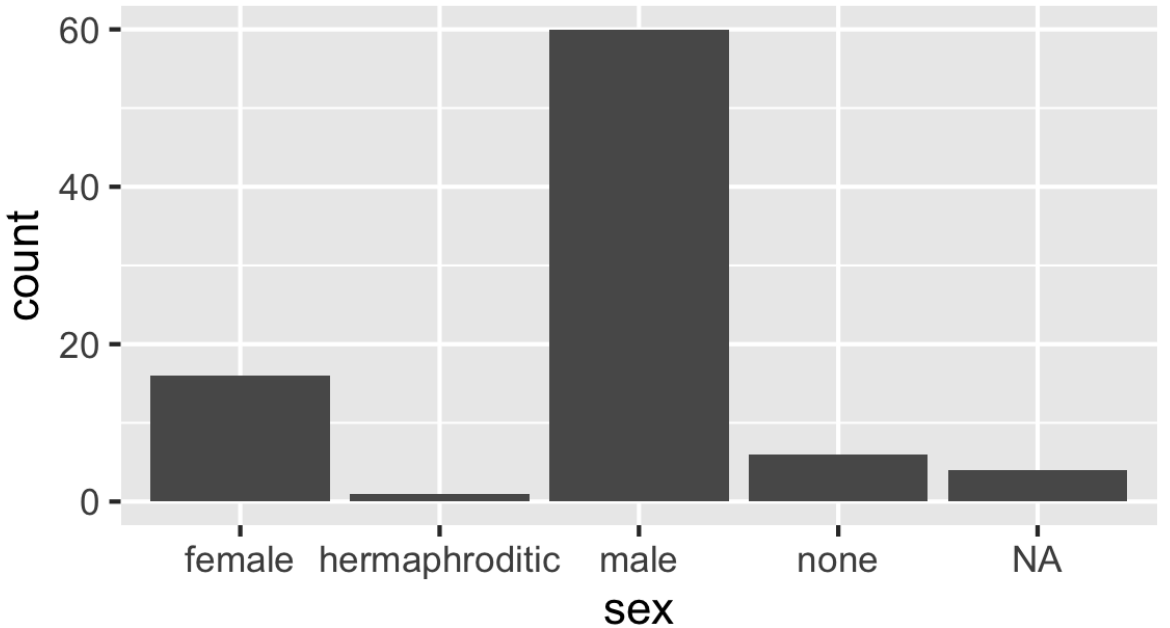
```
ggplot(data = starwars, mapping = aes(x = sex)) +
  geom_bar()
```



Bar plot using statistics

- count the number in each category of sex

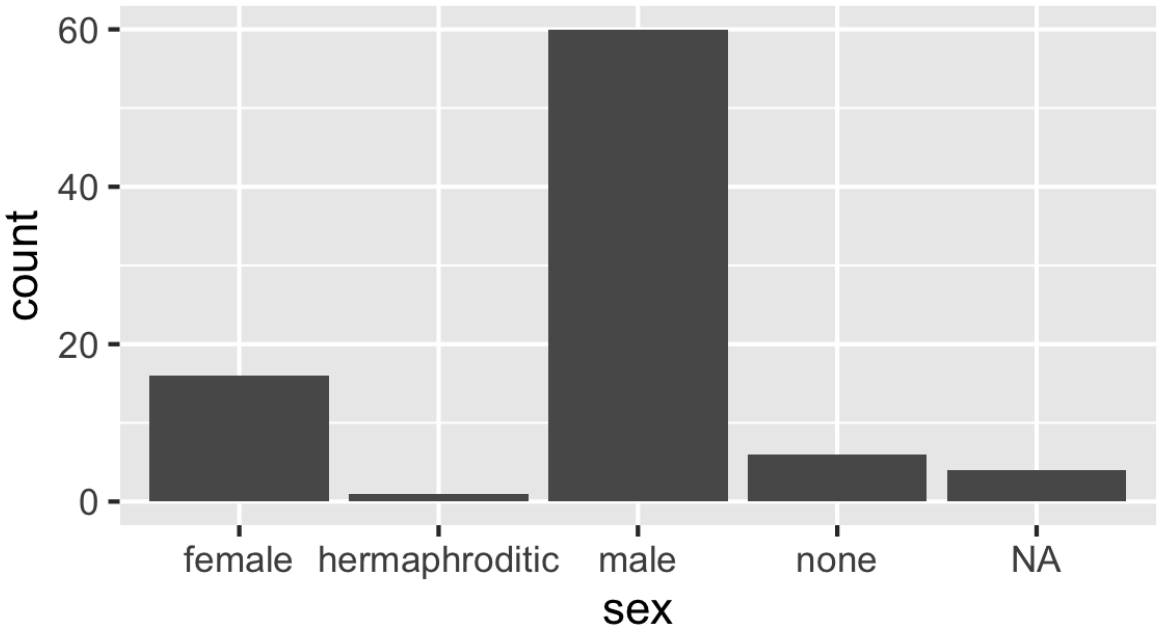
```
ggplot(data = starwars, mapping = aes(x = sex)) +
  stat_count()
```



Bar plot using statistics

- count the number in each category of sex

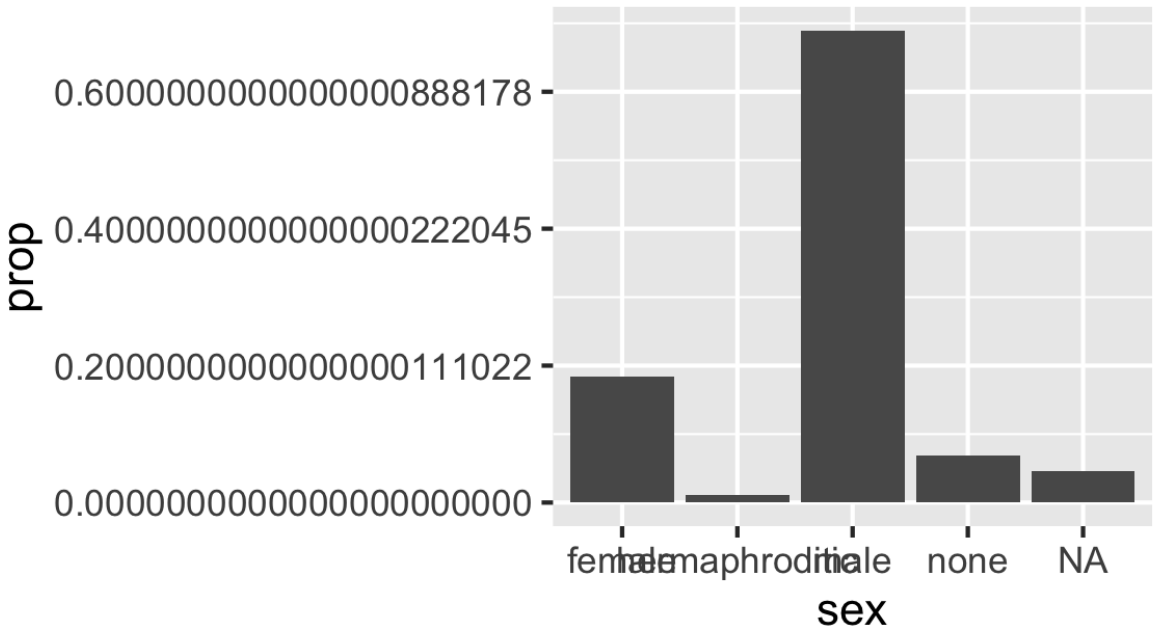
```
ggplot(data = starwars, mapping = aes(x = sex)) +
  geom_bar(stat = "count")
```



Bar plot using statistics

- The relative number in each category of sex

```
ggplot(data = starwars, mapping = aes(x = sex, y = stat(prop), group = 1)) +
  geom_bar()
```



stat_summary

- plot the extent of the

```
ggplot(data = starwars, aes(x = sex, y = height)) +
  stat_summary(
    fun.min = min,
    fun.max = max,
    fun     = mean
  )
```

```
## Warning: Removed 6 rows containing non-finite values (stat_summary).
```

stat_summary

- Plot the quantiles

```
ggplot(data = starwars, aes(x = sex, y = height)) +
  stat_summary(
    fun.min = function(z) quantile(z, prob = 0.1),
    fun.max = function(z) quantile(z, prob = 0.9),
    fun     = median
  )
```

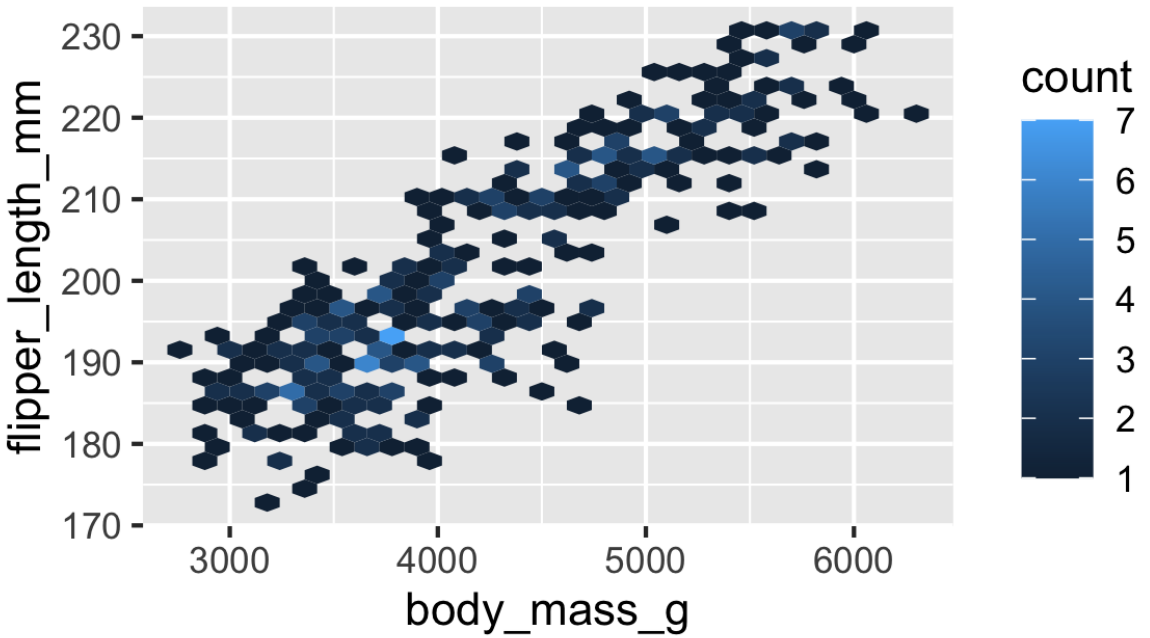
```
## Warning: Removed 6 rows containing non-finite values (stat_summary).
```

Plotting big data

- Use binning

```
ggplot(data = penguins, aes(x = body_mass_g, y = flipper_length_mm)) +
    geom_hex()
```

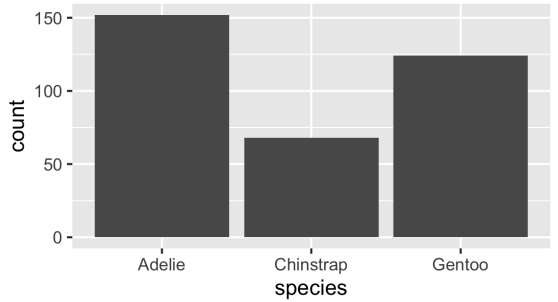
```
## Warning: Removed 2 rows containing non-finite values (stat_binhex).
```



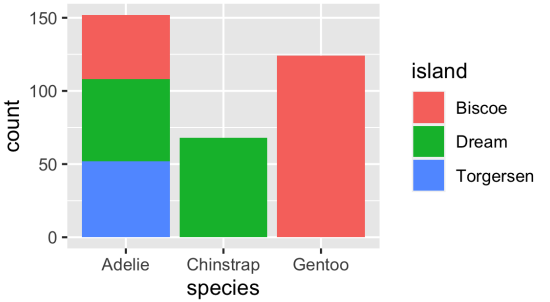
Positions

- can use fill and color to highlight subsets of data
- positions of "dodge" and "jitter" can improve visualization.

```
ggplot(penguins, aes(x = species)) +
  geom_bar()
```

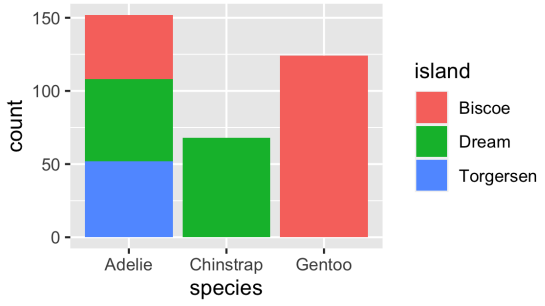



```
ggplot(penguins, aes(x = species, fill =
  geom_bar()
```

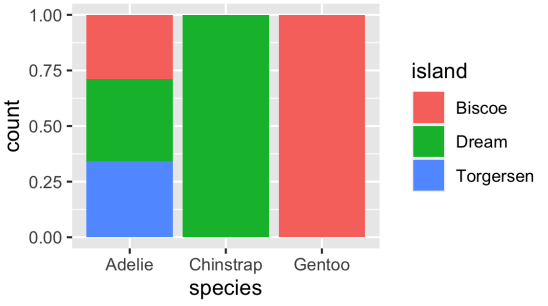


- position = "fill"

```
ggplot(penguins, aes(x = species, fill =
  geom_bar()
```

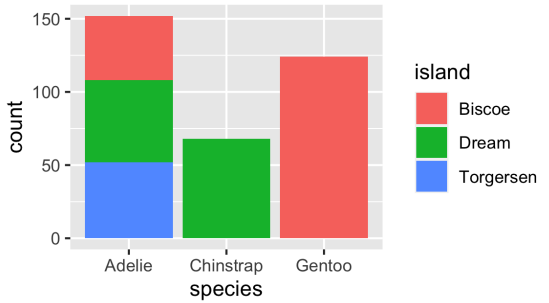



```
ggplot(penguins, aes(x = species, fill =
  geom_bar(position = "fill")
```

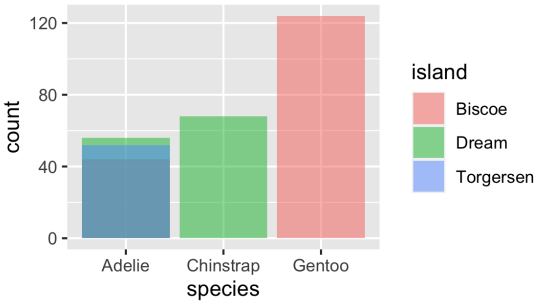


- position = "identity"
  - notice these are overlapping

```
ggplot(penguins, aes(x = species, fill =
  geom_bar()
```

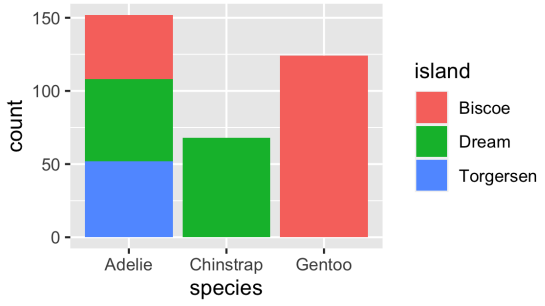



```
ggplot(penguins, aes(x = species, fill =
  geom_bar(position = "identity", alpha =
```

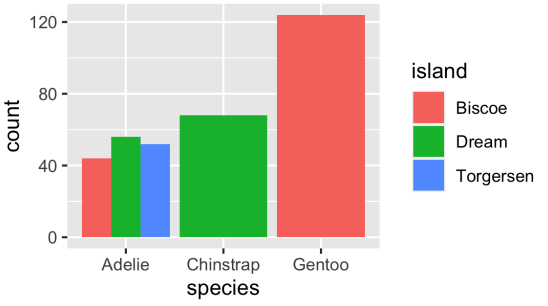


- position = "dodge"

```
ggplot(penguins, aes(x = species, fill =
  geom_bar()
```




```
ggplot(penguins, aes(x = species, fill =
  geom_bar(position = "dodge")
```



- position = "jitter"
    - use the built-in `mpg` dataset

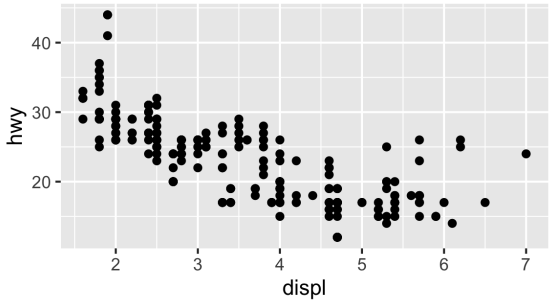
```
glimpse(mpg)
```

```
## Rows: 234
## Columns: 11
## $ manufacturer <chr> "audi", "audi", "audi", "audi", "audi", "audi", "audi", "audi", "audi", "au
## $ model        <chr> "a4", "a4", "a4", "a4", "a4", "a4", "a4", "a4 quattro", "a4 quattro", "a4 q
## $ displ        <dbl> 1.8000000000000000044409, 1.8000000000000000044409, 2.0000000000000000000000,
## $ year         <int> 1999, 1999, 2008, 2008, 1999, 1999, 2008, 1999, 1999, 2008, 2008, 1999, 199
## $ cyl          <int> 4, 4, 4, 4, 6, 6, 6, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 8, 8, 8, 8, 8, 8, 8, 8,
## $ trans        <chr> "auto(l5)", "manual(m5)", "manual(m6)", "auto(av)", "auto(l5)", "manual(m5)
## $ drv          <chr> "f", "f", "f", "f", "f", "f", "f", "4", "4", "4", "4", "4", "4", "4", "4",
## $ cty          <int> 18, 21, 20, 21, 16, 18, 18, 18, 16, 20, 19, 15, 17, 17, 15, 15, 17, 16, 14,
## $ hwy          <int> 29, 29, 31, 30, 26, 26, 27, 26, 25, 28, 27, 25, 25, 25, 25, 24, 25, 23, 20,
## $ fl           <chr> "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p",
## $ class        <chr> "compact", "compact", "compact", "compact", "compact", "compact", "compact"
```

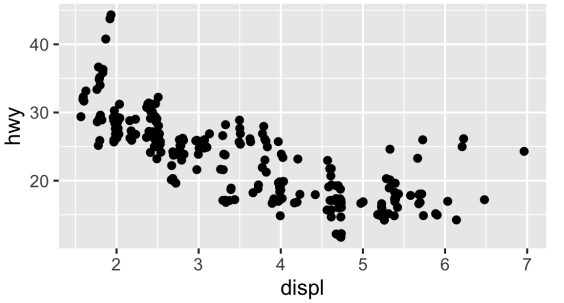
Positions

- position = "jitter"
  - Useful when observations overlap

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



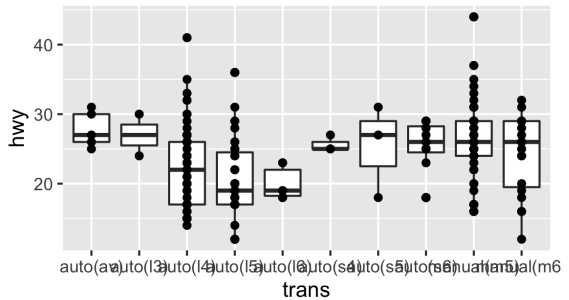

```
ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(position = "jitter")
```



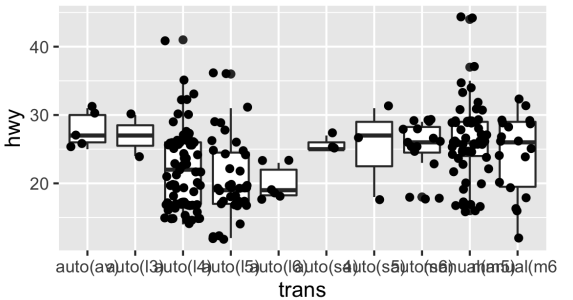
Positions

- position = "jitter"

```
ggplot(mpg, aes(x = trans, y = hwy)) +
  geom_boxplot() +
  geom_point()
```




```
ggplot(mpg, aes(x = trans, y = hwy)) +
  geom_boxplot() +
  geom_point(position = "jitter")
```



Exploratory Data Analysis

- EDA is an artform
- Explore and learn from the data
- Guides model development
- Identifies transformations of the data that might be helpful
- Helps in formulating questions about the data
- Express creativity!

Tidy data

- Each row is an observation
  - An observation is a set of measurements about an element
  - The element is the object on which measurement is made
- Each column is a variable
  - A variable is a characteristic of the element that can take on an value
- Tidy data is tabular where each row is an observation and each column is a variable

```
str(diamonds)
```

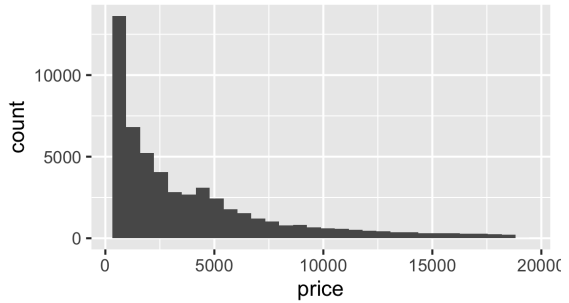
```
## tibble[,10] [53,940 × 10] (S3: tbl_df/tbl/data.frame)
##  $ carat  : num [1:53940] 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
##  $ cut    : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...
##  $ color  : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...
##  $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...
##  $ depth  : num [1:53940] 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
##  $ table  : num [1:53940] 55 61 65 58 58 57 57 55 61 61 ...
##  $ price  : int [1:53940] 326 326 327 334 335 336 336 337 337 338 ...
##  $ x      : num [1:53940] 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
##  $ y      : num [1:53940] 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
##  $ z      : num [1:53940] 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

Distributions

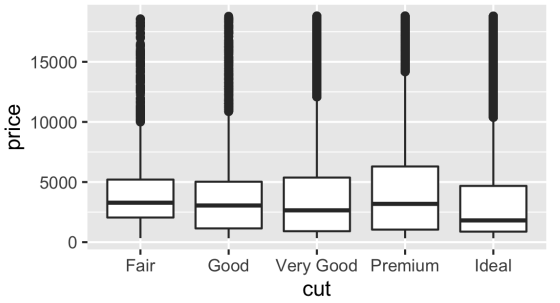
- Visualize marginal (one variable) distributions
  - histograms `geom_histogram()`
  - density plots `geom_dens()`
  - boxplots `geom_boxplot()`
  - violin plots `geom_violin()`

Distributions

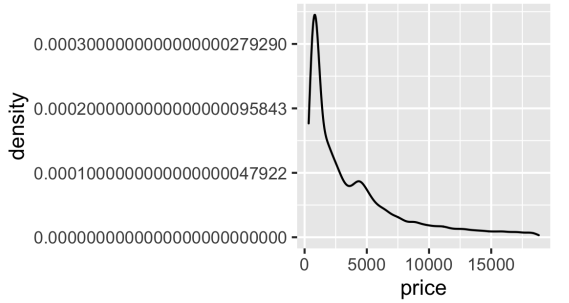
```
ggplot(diamonds, aes(x = price)) +
  geom_histogram()
```



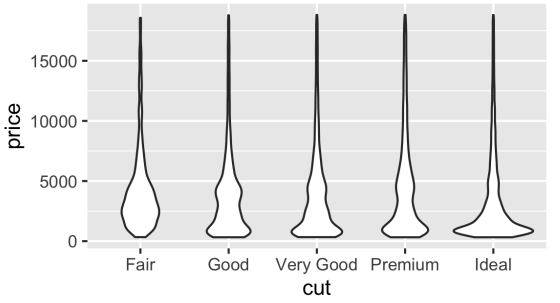

```
ggplot(diamonds, aes(x = cut, y = price)) +
  geom_boxplot()
```




```
ggplot(diamonds, aes(x = price)) +
  geom_density()
```




```
ggplot(diamonds, aes(x = cut, y = price)) +
  geom_violin()
```



Missing values

- As a rule, don't just ignore missing values blindly
    - I conduct a survey about income and leisure time. Why shouldn't I ignore those people that don't respond?
- Many functions have a `na.rm` option

```
mean(penguins$bill_length_mm)
```

```
## [1] NA
```

```
mean(penguins$bill_length_mm, na.rm = TRUE)
```

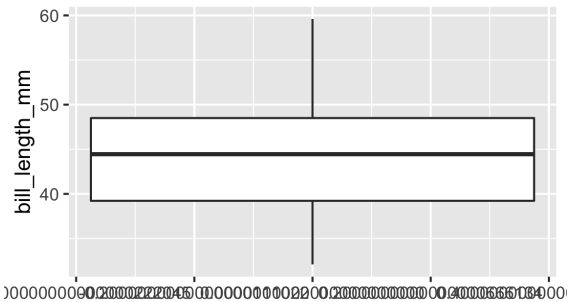
```
## [1] 43.92192982456140271097
```

Missing values

- Notice the warning

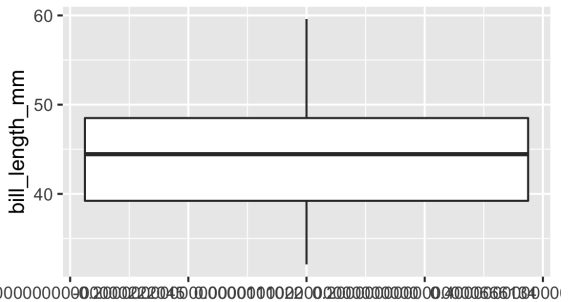
```
ggplot(penguins, aes(y = bill_length_mm)) +
  geom_boxplot()
```

```
## Warning: Removed 2 rows containing non-finite val
```



- No warning with `na.rm = TRUE`

```
ggplot(penguins, aes(y = bill_length_mm)) +
  geom_boxplot(na.rm = TRUE)
```

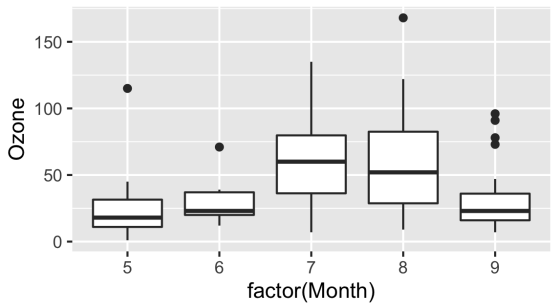


Missing values

- What if missing values are important?
- `airquality` dataset is missing the Ozone variable in June.

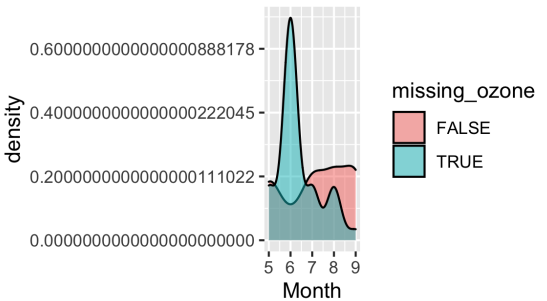
- Ozone measurements vs. Month

```
airquality %>%
  ggplot(aes(x = factor(Month), y = Ozone
  geom_boxplot()
```



- Missing values vs. Month

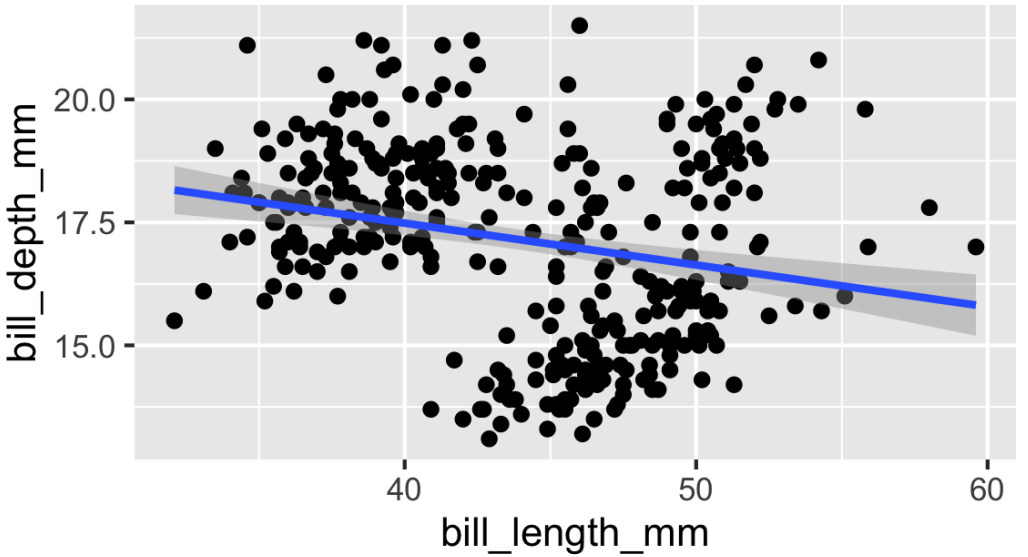
```
airquality %>%
  mutate(missing_ozone = is.na(Ozone)) %>
  ggplot(aes(fill = missing_ozone, x = Mo
  geom_density(alpha = 0.5)
```



Covariation

- Statistical modeling is about finding patterns and covariation in data
- Is there a relationship between bill depth and bill length?

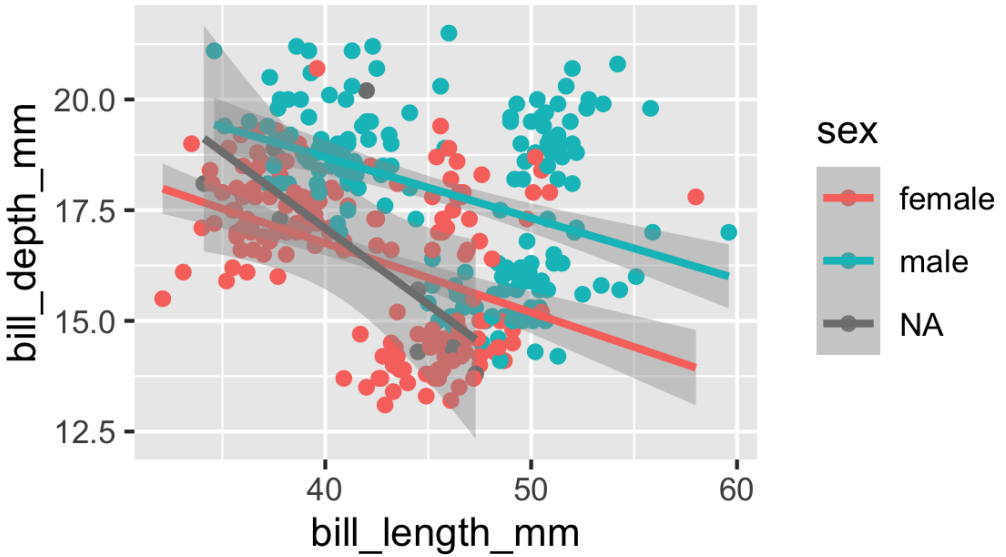
```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm)) +
  geom_point() +
  stat_smooth(method = "lm")
```



Covariation

- Statistical modeling is about finding patterns and covariation in data
- Is there a relationship between bill depth and bill length?

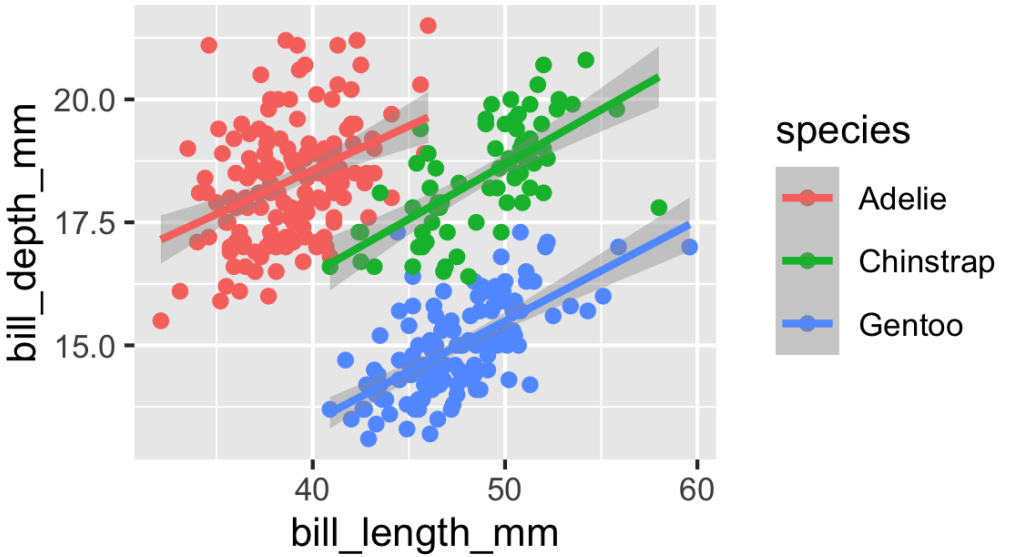
```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm, color = sex)) +
  geom_point() +
  stat_smooth(method = "lm")
```



Covariation

- Statistical modeling is about finding patterns and covariation in data
- Is there a relationship between bill depth and bill length?

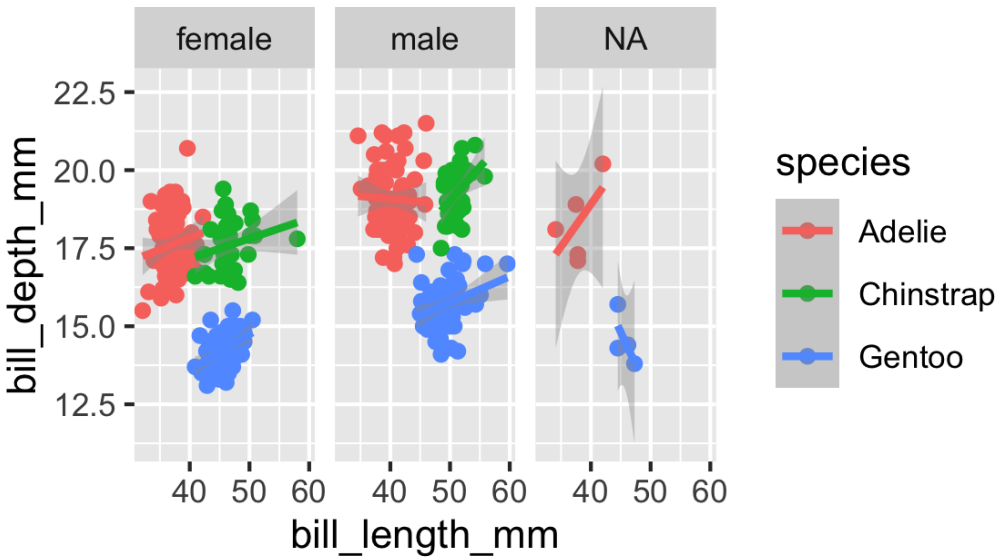
```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm, color = species)) +
  geom_point() +
  stat_smooth(method = "lm")
```



Covariation

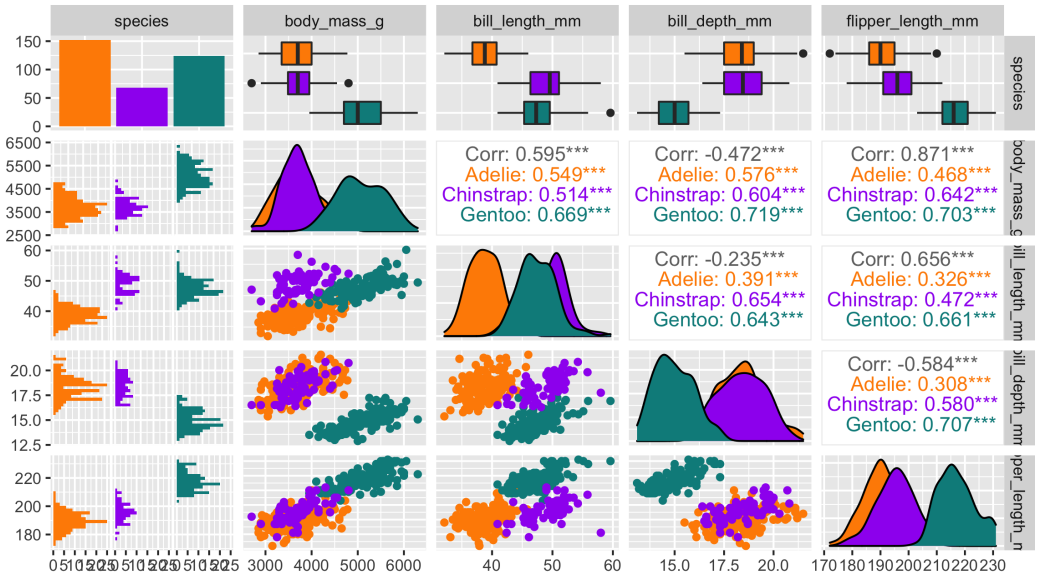
- Statistical modeling is about finding patterns and covariation in data
- Is there a relationship between bill depth and bill length?

```
ggplot(penguins, aes(x = bill_length_mm, y = bill_depth_mm, color = species)) +
  geom_point() +
  facet_wrap(~ sex) +
  stat_smooth(method = "lm")
```



Pairs plots

```
penguins %>%
  select(species, body_mass_g, ends_with("_mm")) %>%
  GGally::ggpairs(aes(color = species)) +
  scale_colour_manual(values = c("darkorange", "purple", "cyan4")) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4"))
```



Namespaces

- What's the deal with the `GGally::ggpairs()` function on the last slide?
    - Often in programming there are different packages that have functions with the same name
    - The **namespace** resolves this issue
    - Use the `ggpairs` function from the `GGally` package