ggplot2 - January 2019 Data Carpentry Lesson

Maria Kamenetsky

January 17, 2019

Outline

- 0) Prep the Data
- 1) `ggplot()` function
  - base plot
  - x and y
- 2) *geoms*
  - `*` vs. %>%
- 3) Other Aesthetics
- 4) Layering (line plot + boxplot)
- 5) Facetting
  - `facet_grid()`
  - `facet_wrap()`
- 6) Themes
- 7) Saving plots (if time allows)

Why `ggplot2`?

- exploratory data analysis
  - quick exploration for trends in data, potential outliers, etc.
- if you can think of a visualization, there's probably a way to do it using ggplot
- customization is easy
- 1 viz package to rule them all
- syntax is consistent
- **gg** = grammar of graphics. There is a syntax to creating plots, which involves adding things in brick by brick

0) Prep the Data

Load packages we will be using:

```
library(dplyr)
library(ggplot2)
```

GOAL: Use `dplyr` tools to reformat our data so that we can make visualizations for *mammals* data.

- We will make three datasets from *portal_clean.csv*:
  - 1) **just_dm** dataset only containing species=="DM"

- 2) **stat_summary** dataset containing mean weight, mean height, and observation count by species
- 3) **year_summary** dataset containing mean weight, mean height, and observation count by species, year, and sex.

Load data:

```
download.file("http://kbroman.org/datacarp/portal_clean.csv",
              "C:/Users/Maria/Desktop/DataCarpentry/Clean/portal_clean.csv")
surveys <- read.csv("C:/Users/Maria/Desktop/DataCarpentry/Clean/portal_clean.csv")
```

Create three datasets:

```
#just_dm
just_dm <- surveys %>% filter(species_id=="DM")
str(just_dm)
```

```
## 'data.frame':    9727 obs. of  13 variables:
##  $ record_id      : int  226 233 245 251 257 259 268 346 350 354 ...
##  $ month          : int  9 9 10 10 10 10 10 11 11 11 ...
##  $ day            : int  13 13 16 16 16 16 16 12 12 12 ...
##  $ year           : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ plot_id        : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ species_id     : Factor w/ 19 levels "BA","DM","DO",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ sex            : Factor w/ 2 levels "F","M": 2 2 2 2 2 2 1 1 2 2 ...
##  $ hindfoot_length: int  37 25 37 36 37 36 36 37 37 38 ...
##  $ weight         : int  51 44 39 49 47 41 55 36 47 44 ...
##  $ genus          : Factor w/ 9 levels "Baiomys","Chaetodipus",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ species        : Factor w/ 18 levels "albigula","baileyi",..: 12 12 12 12 12 12 12 12 12 12 ...
##  $ taxa           : Factor w/ 1 level "Rodent": 1 1 1 1 1 1 1 1 1 1 ...
##  $ plot_type      : Factor w/ 5 levels "Control","Long-term Krat Exclosure",..: 1 1 1 1 1 1 1 1 1 1
```

```
#stat_summary
stat_summary <- surveys %>%
    group_by(species_id) %>%
    summarize(mean_wt=mean(weight),
              mean_hfl=mean(hindfoot_length),
              n=n())
str(stat_summary)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    19 obs. of  4 variables:
##  $ species_id: Factor w/ 19 levels "BA","DM","DO",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ mean_wt   : num  8.6 43.1 48.9 120.2 158.8 ...
##  $ mean_hfl  : num  13 36 35.6 50 32.2 ...
##  $ n         : int  45 9727 2790 2023 1045 905 2081 2803 1198 1469 ...
```

```
#year_summary
year_summary <- surveys %>%
    group_by(species_id, year, sex) %>%
    summarize(mean_wt=mean(weight),
              mean_hfl=mean(hindfoot_length),
              n=n())
str(year_summary)
```

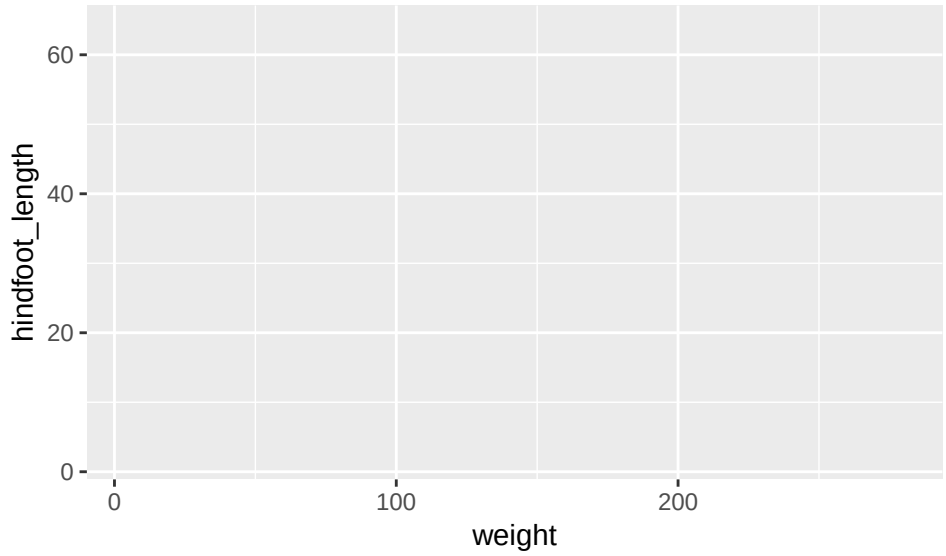
```
## Classes 'grouped_df', 'tbl_df', 'tbl' and 'data.frame':  613 obs. of  6 variables:
##  $ species_id: Factor w/ 19 levels "BA","DM","DO",..: 1 1 1 1 1 1 1 2 2 2 ...
##  $ year      : int  1989 1990 1990 1991 1991 1992 1992 1977 1977 1978 ...
##  $ sex       : Factor w/ 2 levels "F","M": 2 1 2 1 2 1 2 1 2 1 ...
```

```
##  $ mean_wt   : num  7 8.38 7 9.74 7.67 ...
##  $ mean_hfl  : num  13 13.8 14 12.8 13 ...
##  $ n         : int  3 8 3 19 6 4 2 75 106 165 ...
##  - attr(*, "vars")= chr  "species_id" "year"
##  - attr(*, "drop")= logi TRUE
```

1) `ggplot()` function

Goal: scatterplot of weight (x) by hindfoot_length (y) using *surveys* dataset.

```
ggplot(surveys, aes(x = weight, y = hindfoot_length))
```

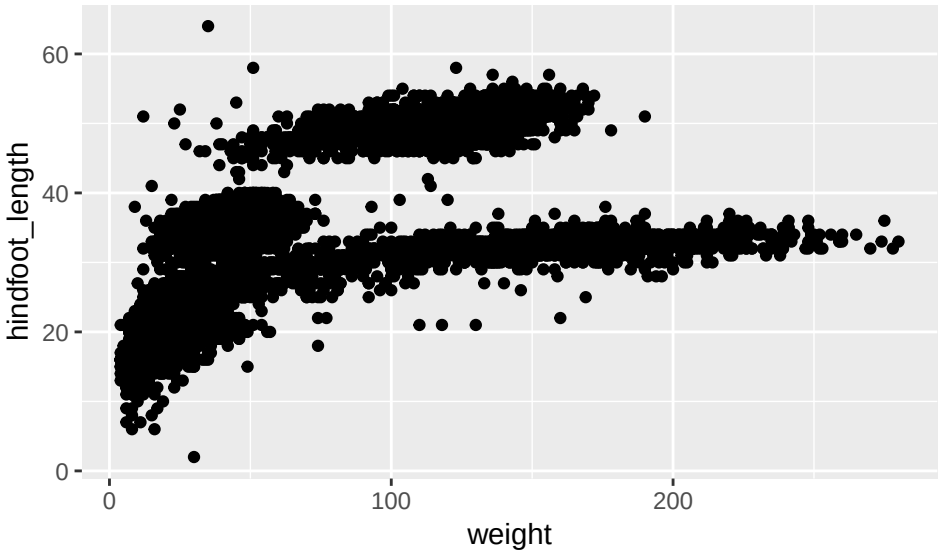


Empty plot! We need to tell `ggplot()` what kind of plot we want. Default is to only plot the axes. To select the plot type, we need to learn about **geom**'s or **geometries**.

2) *geom*'s

- *geom* is short for geometry. This determines the type of plot that ggplot will make
- *many* different geom's available
- scatterplot: `geom_point()`

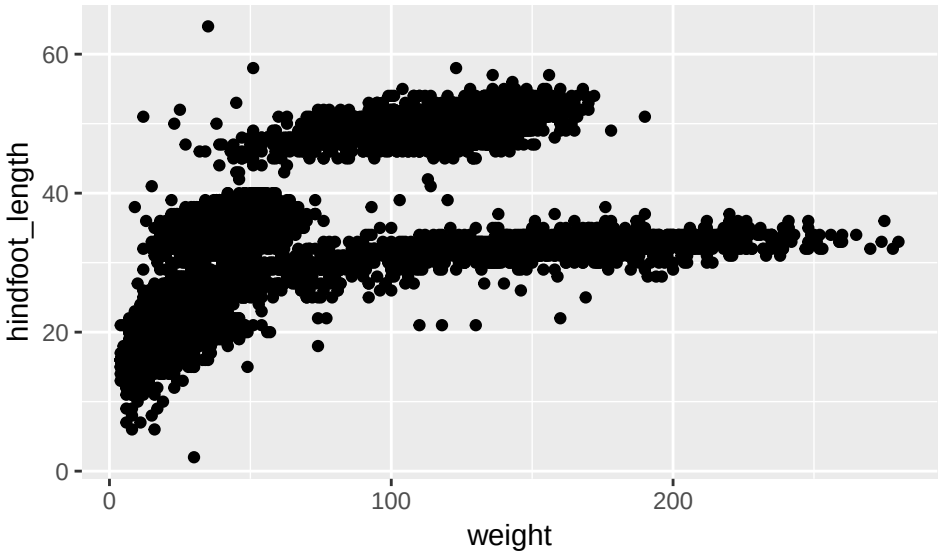
```
ggplot(surveys, aes(x = weight, y = hindfoot_length)) + geom_point()
```



Can assign this plot to an object:

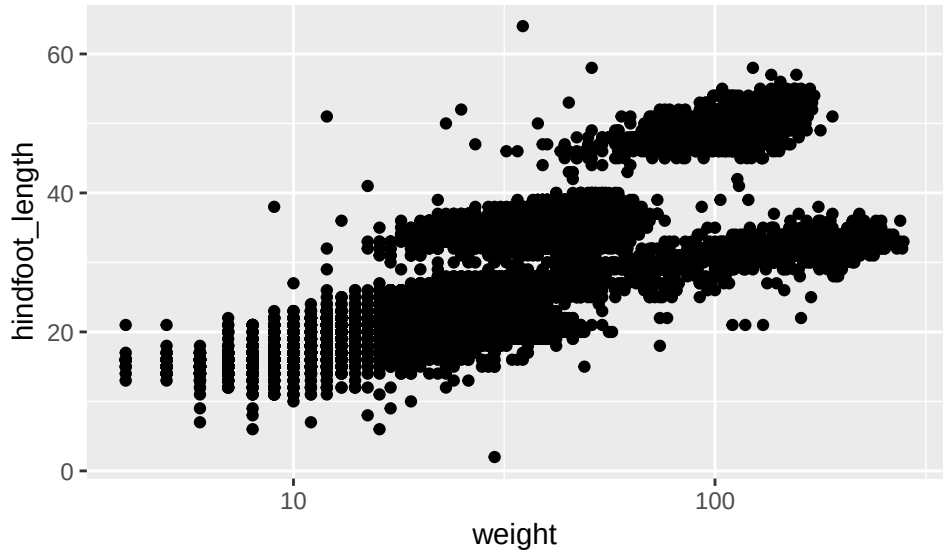
```
p1 <- ggplot(surveys, aes(x = weight, y = hindfoot_length)) + geom_point()
#nothing happens
```

```
p1
```

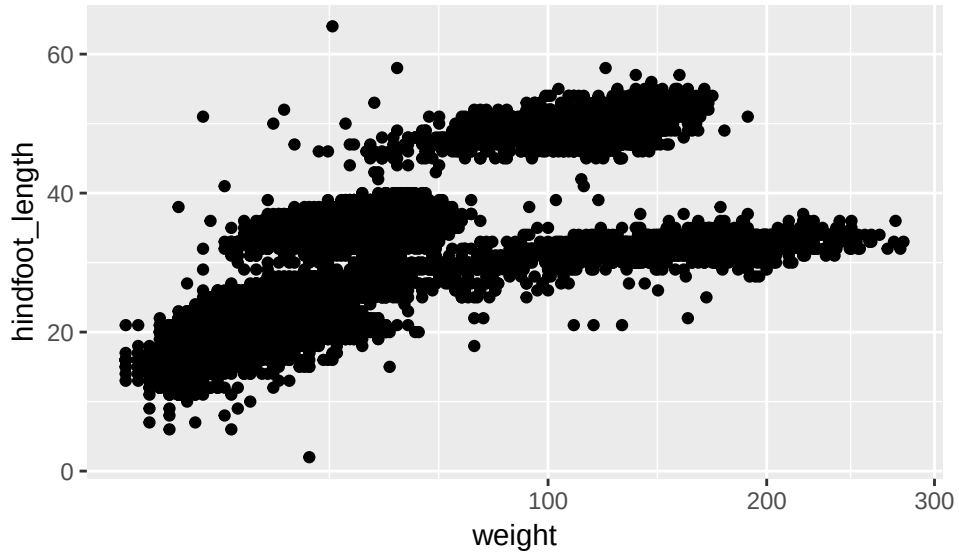


This makes it easy to try different things using + operator.

```
#log scale for x-axis
p1 + scale_x_log10()
```



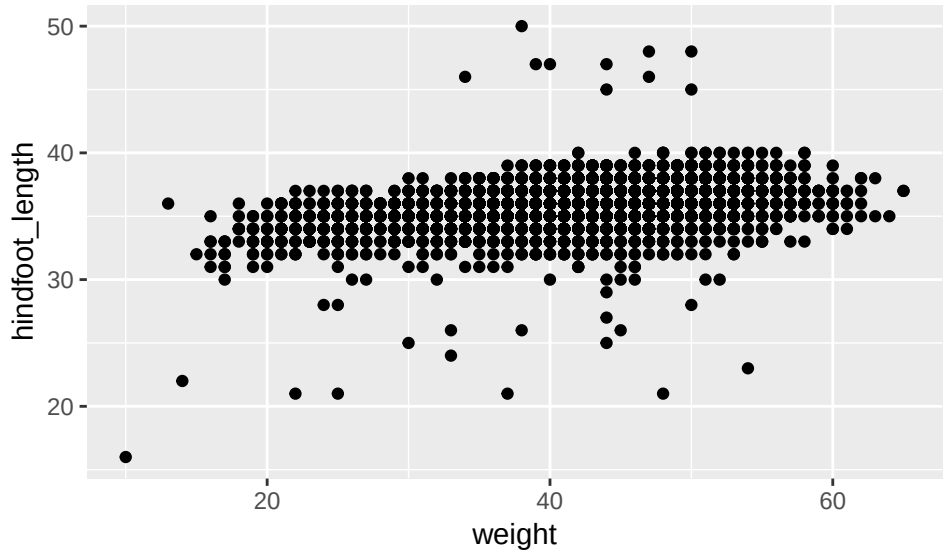

```
#square root scale for x-axis
p1 + scale_x_sqrt()
```



CHALLENGE 1: Make a scatterplot of *hindfoot_length* vs. *weight* but only for *species_id* "DM"

- Use the dataset we created,**just_dm**
- Use our `ggplot2()` code above but with this new dataset in place of *surveys*.

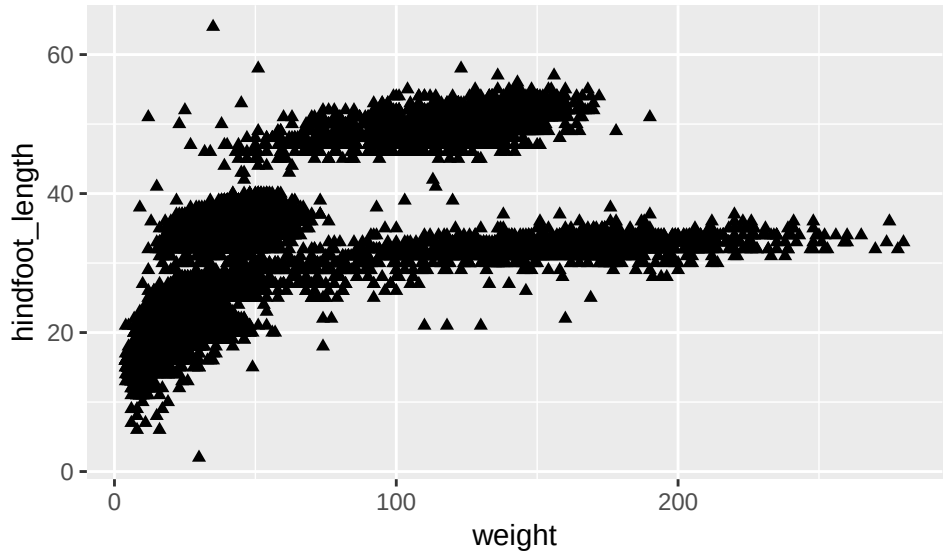
```
#Challenge solution

ggplot(just_dm, aes(x=weight, y= hindfoot_length)) + geom_point()
```



Other Aesthetics

- shape

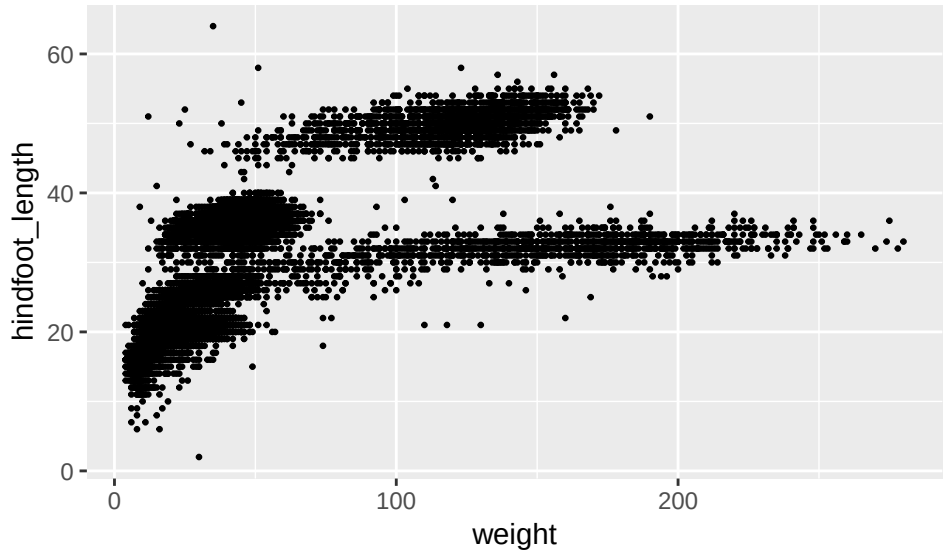
```
ggplot(surveys, aes(x = weight, y = hindfoot_length)) +
    geom_point(shape="triangle")
```




```
#assign base plot to p2 to avoid extra typing
p2 <- ggplot(surveys, aes(x = weight, y = hindfoot_length))
```

- point size

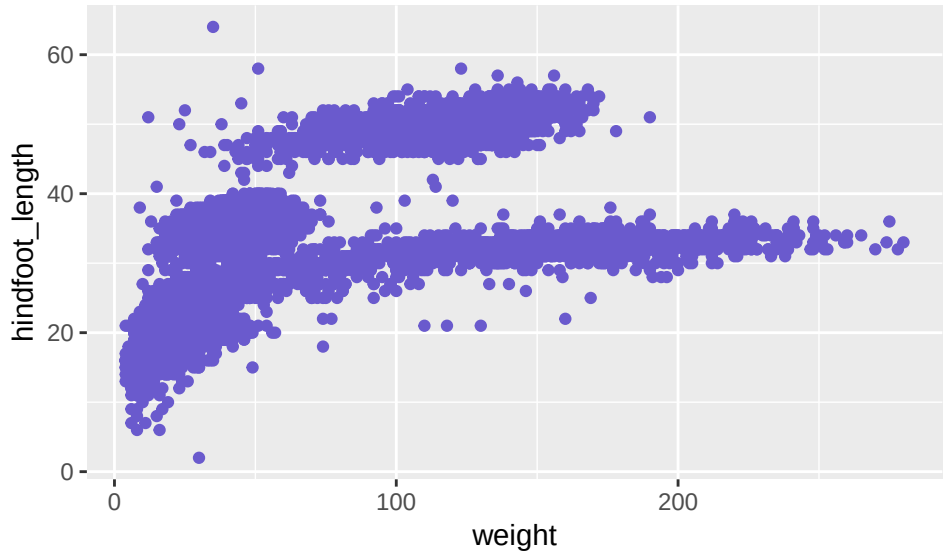
```
#These two plots are equivalent because I assigned p2 <- ggplot(surveys, aes(x = weight, y = hindfoot_l
p2 + geom_point(size=0.5)
```




```
# ggplot(surveys, aes(x = weight, y = hindfoot_length)) +
#     geom_point(size=0.5)
```

- color

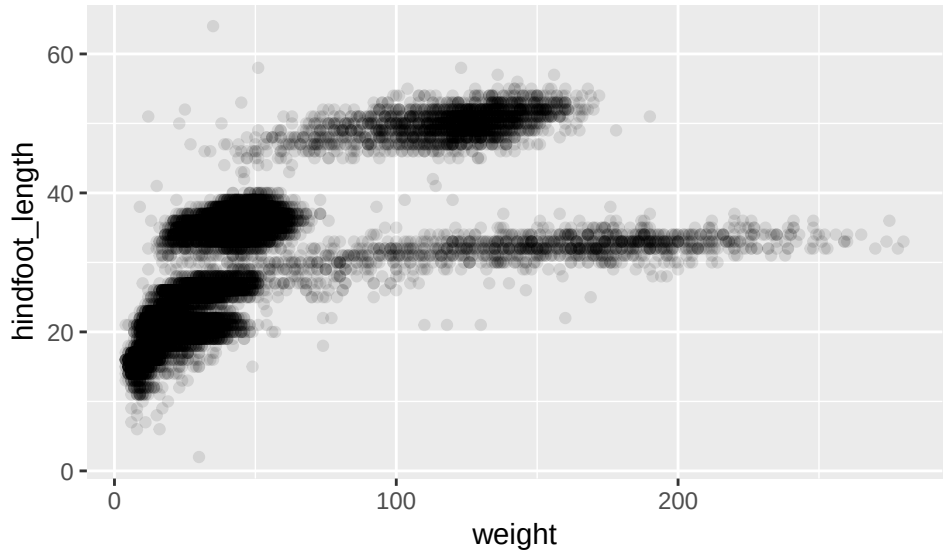
```
p2 + geom_point(color = "slateblue")
```




```
# ggplot(surveys, aes(x = weight, y = hindfoot_length)) +
#     geom_point(color = "slateblue")
```

- alpha (= transparency)

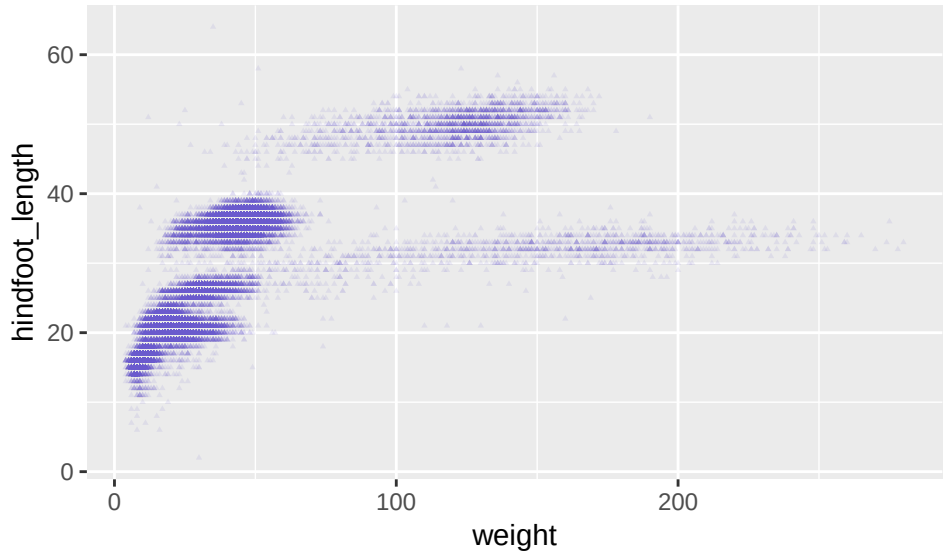
```
p2 + geom_point(alpha = 0.1)
```




```
# ggplot(surveys, aes(x = weight, y = hindfoot_length)) +
#     geom_point(alpha = 0.1)
```

- Putting it all together

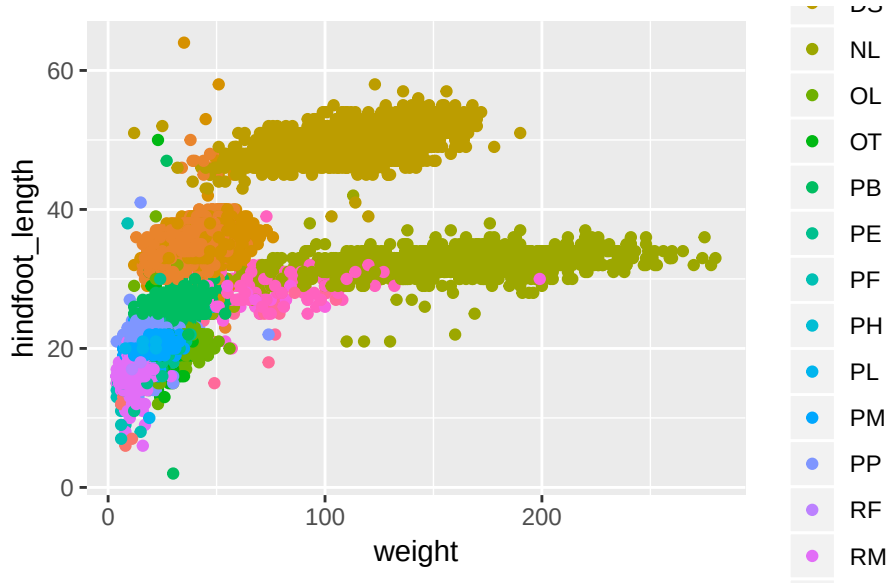
```
p2 + geom_point(shape="triangle", size=0.5, color="slateblue", alpha=0.1)
```



Assign Aesthetics to Data

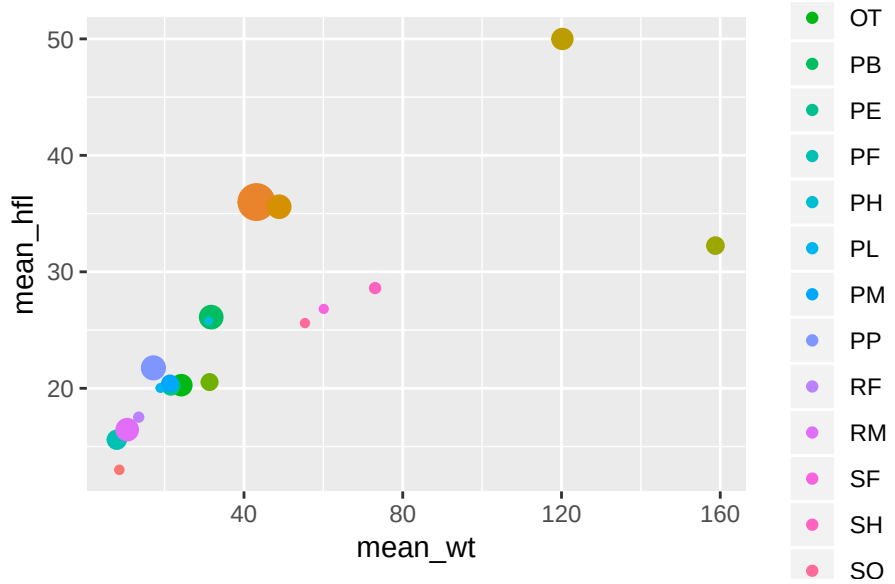
- we can also assign aesthetics to data instead of fixed values or colors.

```
p2 + geom_point(aes(color= species_id))
```



- **CHALLENGE 2: Make a scatterplot of mean *hindfoot_length* vs. mean *weight*, where each point is a species, and where the sizes of the points indicate the sample size**
  - Use the dataset `stat_summary`
  - Use our ggplot code with the aesthetics `x=mean_wt` and `y=mean_hfl`, plus `size=n`.

```
ggplot(stat_summary, aes(x=mean_wt, y=mean_hfl)) + geom_point(aes(color = species_id, size=n))
```



Layering

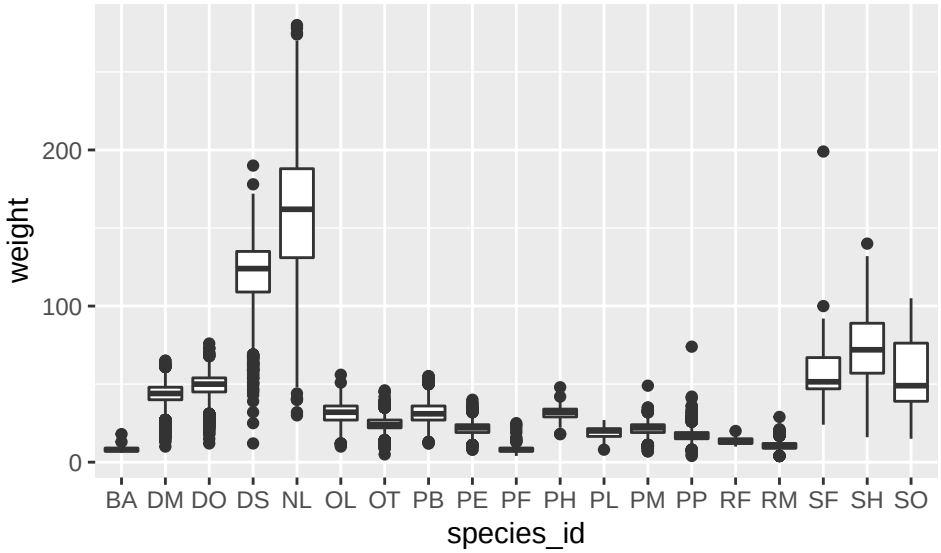
- recall, grammar of graphics, add things on layer by layer, brick by brick.

Boxplots

Goal: Boxplots of weight by species.

We will use the `geom_boxplot()` geom.

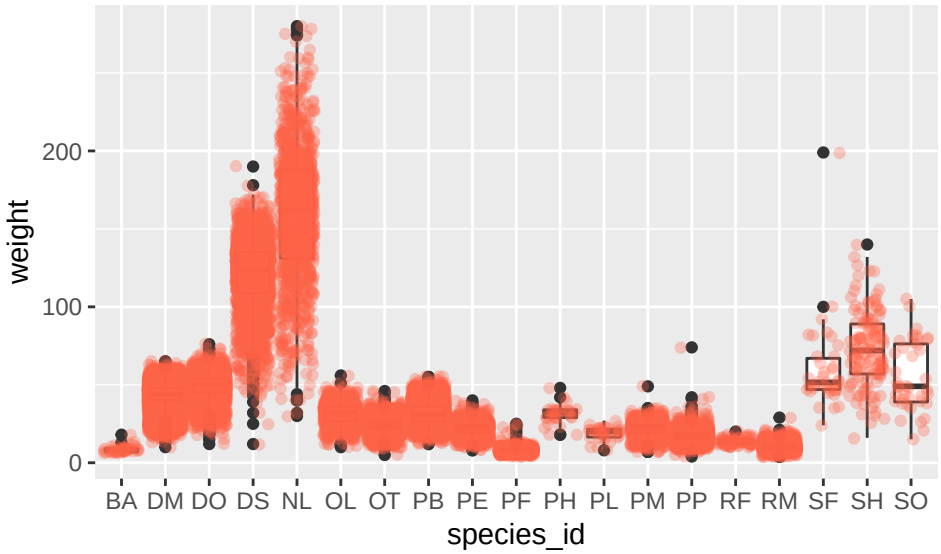
```
ggplot(surveys, aes(x=species_id, y = weight)) +
    geom_boxplot()
```



Can add points to boxplots to get better idea of number of measurements and their distribution. We will add `geom_jitter()`.

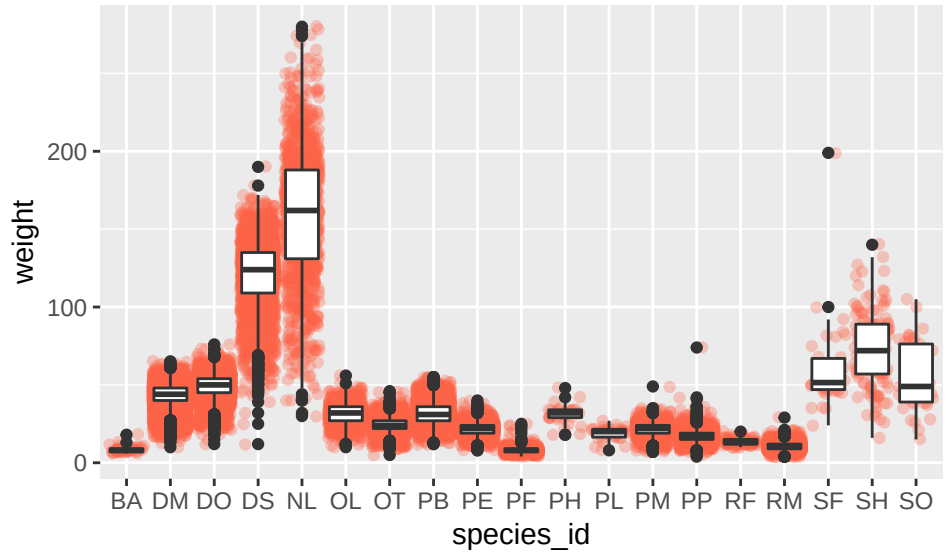
- `geom_boxplot() + geom_jitter()`

```
ggplot(surveys, aes(x=species_id, y = weight)) +
    geom_boxplot() +
    geom_jitter(alpha =0.3, color="tomato")
```



- `geom_jitter() + geom_boxplot()`

```
ggplot(surveys, aes(x=species_id, y = weight)) +
    geom_jitter(alpha =0.3, color="tomato") +
    geom_boxplot()
```



Line Plots (Time-Series)

Goal: Make a lineplot counts of animals by year. We will use `geom_line()` geom.

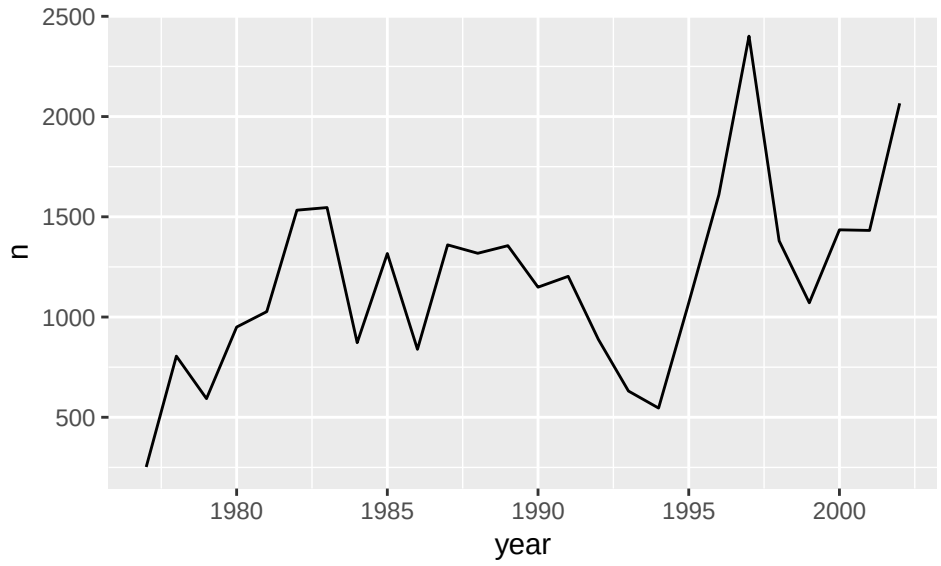
First, create new dataset using `dplyr`

```
count_by_year <- surveys %>%
    group_by(year) %>%
    tally
str(count_by_year)
```

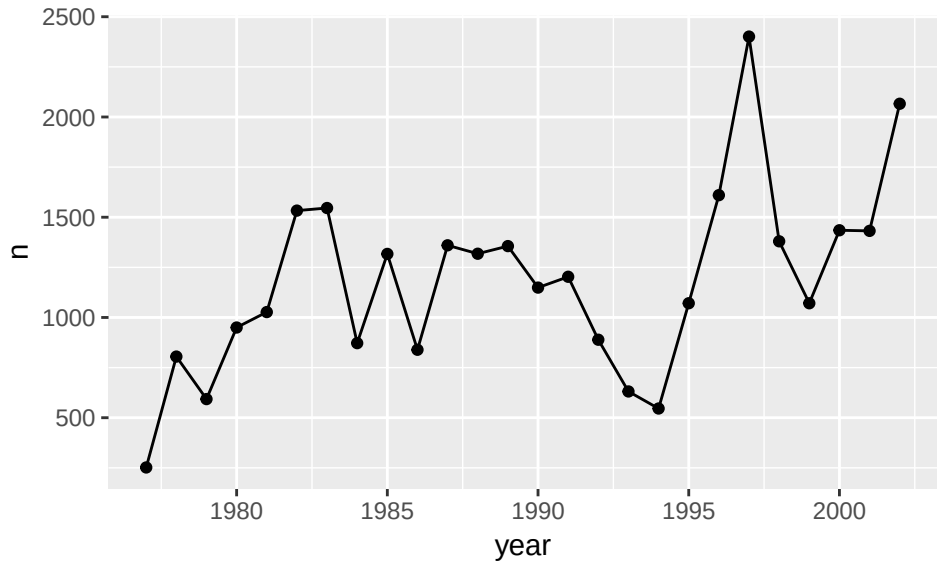
```
## Classes 'tbl_df', 'tbl' and 'data.frame':    26 obs. of  2 variables:
##  $ year: int  1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 ...
##  $ n   : int  252 805 593 950 1027 1533 1546 872 1317 839 ...
```

Next, use `geom_line()` geom.

```
ggplot(count_by_year, aes(x=year, y=n)) +
    geom_line()
```

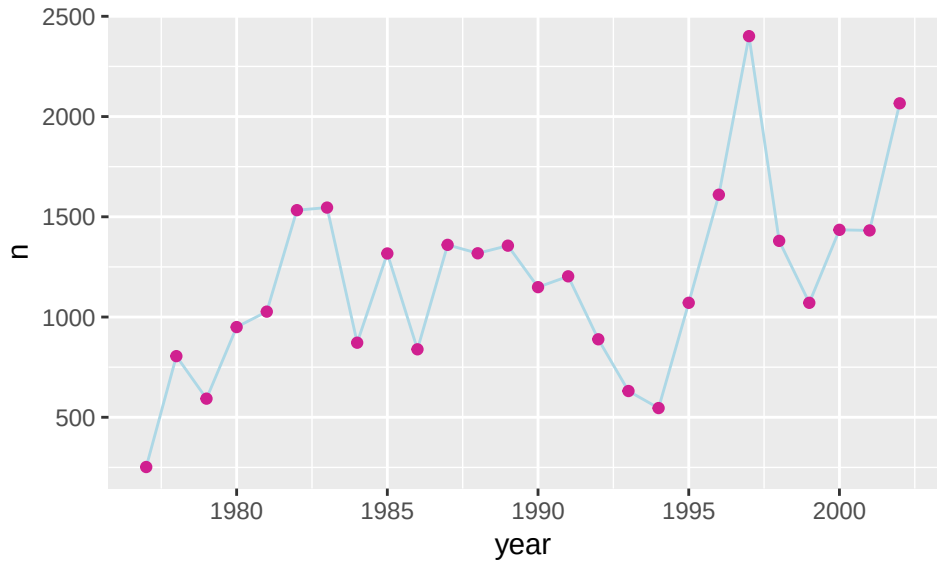



```
ggplot(count_by_year, aes(x=year, y=n)) +
    geom_line() +
    geom_point()
```



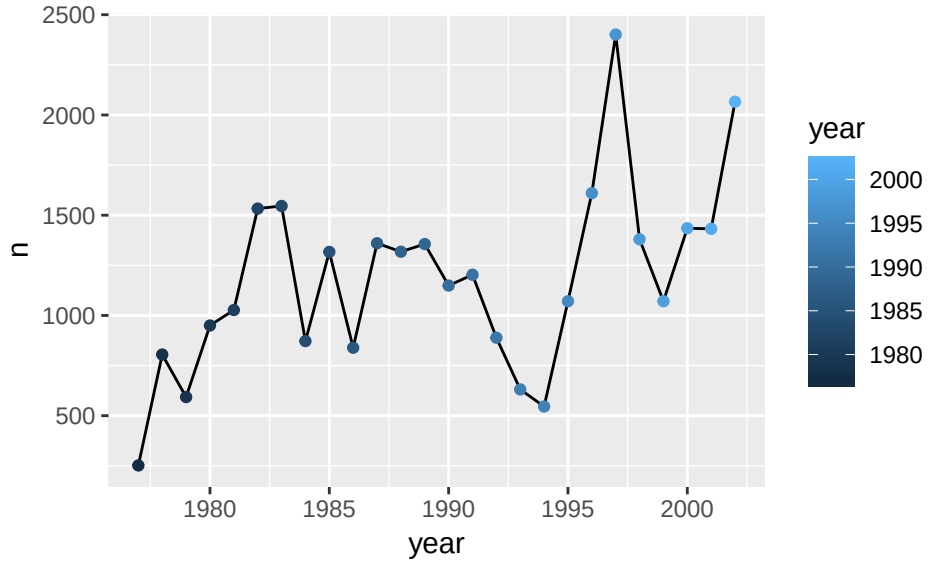
We know that since `geom_point()` was called after `geom_line()`, the points are placed on top of the lines. We can confirm this if we plot the lines and points in contrasting colors:

```
ggplot(count_by_year, aes(x=year, y=n)) +
    geom_line(color="lightblue") +
    geom_point(color="violetred")
```



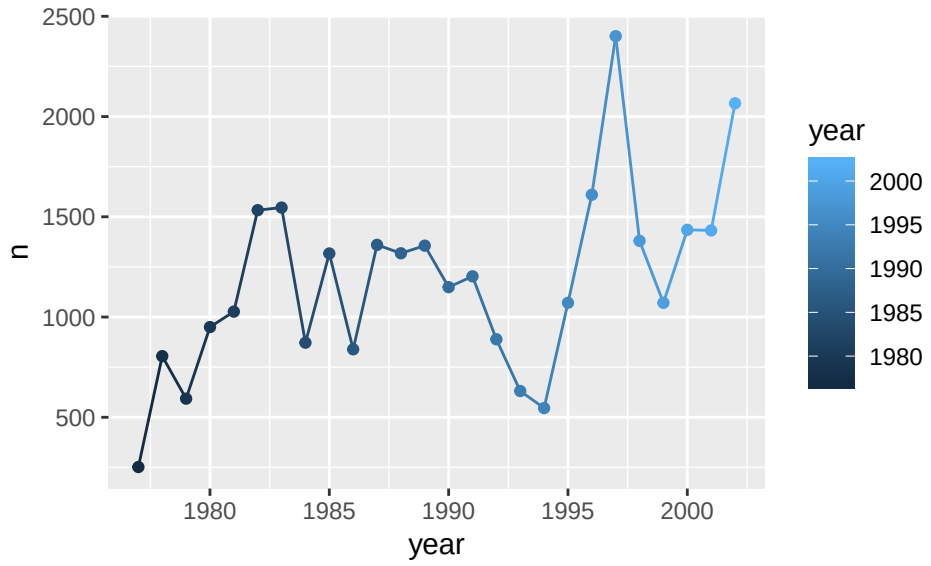
We could also control the aesthetics separately for each layer if we want to:

```
ggplot(count_by_year, aes(x=year, y=n)) +
    geom_line() +
    geom_point(aes(color=year))
```

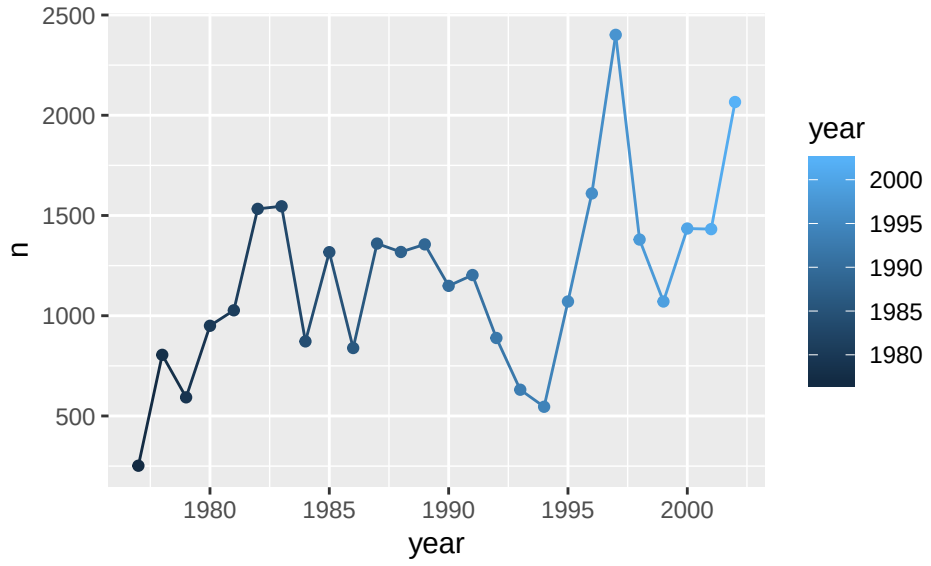


Or if we want to color both points and lines:

```
ggplot(count_by_year, aes(x=year, y=n, color=year)) +
    geom_line() +
    geom_point()
```

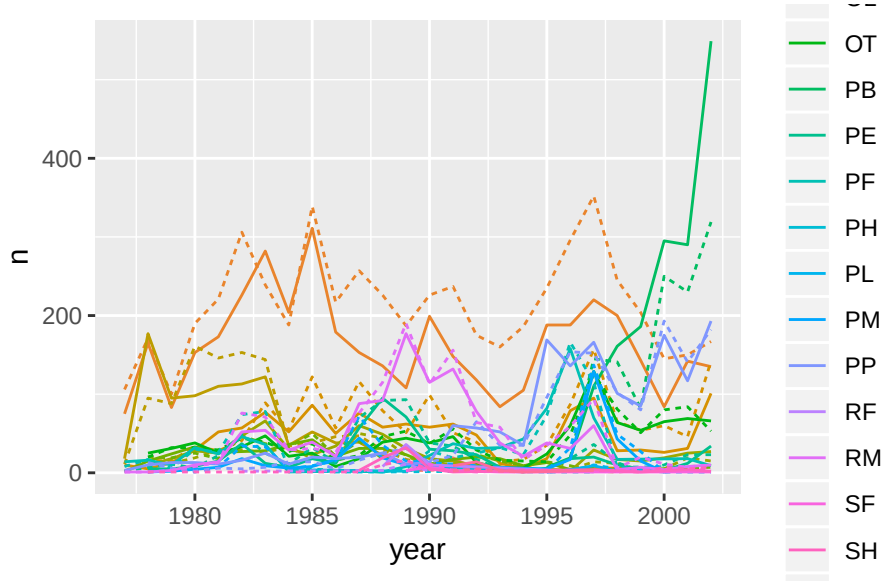



```
#this is equivalent to:

ggplot(count_by_year, aes(x=year, y=n)) +
    geom_line() +
    geom_point() +
    aes(color=year)
```



- **Challenge 3:**
  - **Use the `year_summary` dataset to make a line plot of counts of each species by year, with a different colored line for each species**
  - **Use `aes(linetype=sex)` to have different line types for the two sexes**

```
ggplot(year_summary, aes(x=year, y=n)) +
    geom_line(aes(color=species_id, linetype=sex))
```



Facetting

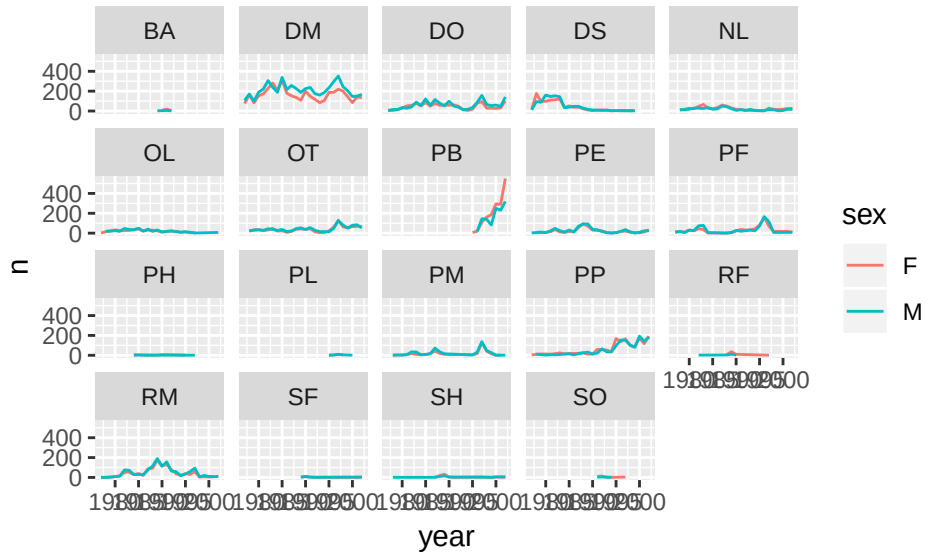
General Goal: Plot data in `year_summary` into multiple panels.

`facet_wrap()`

- `facet_wrap()` geometry extracts plots into an arbitrary number of dimensions to allow the to fit on one page

Specific Goal: Plot count by year, with separate lines for sex and separate panels for species.

```
ggplot(year_summary, aes(x = year, y = n)) +
    geom_line(aes(color=sex)) +
    facet_wrap(~ species_id)
```

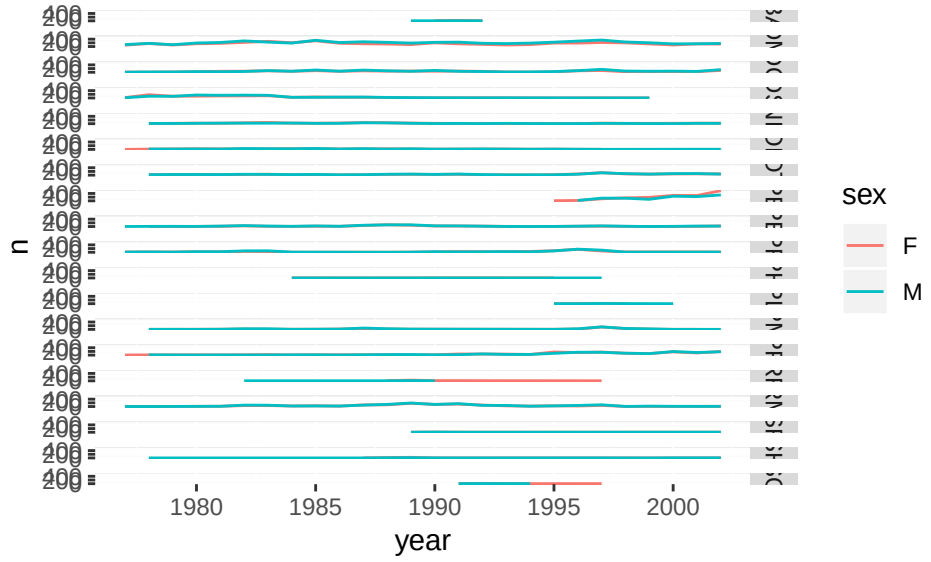


facet_grid()

Specific Goal: Plot count by year, with separate lines for sex and separate panels for species.

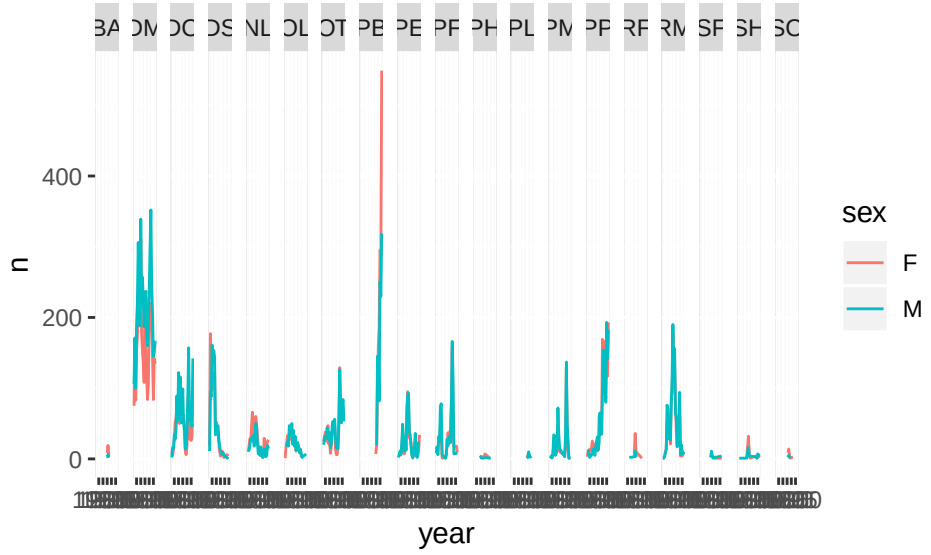
- `facet_grid()` geometry allows you to explicitly specify how you want your plots to be arranged via formula notation (`rows ~ columns`)
  - a `.` can be used as a placeholder that indicates only 1 row or 1 column.
- vertical split

```
ggplot(year_summary, aes(x = year, y = n)) +
    geom_line(aes(color=sex)) +
    facet_grid(species_id~.)
```



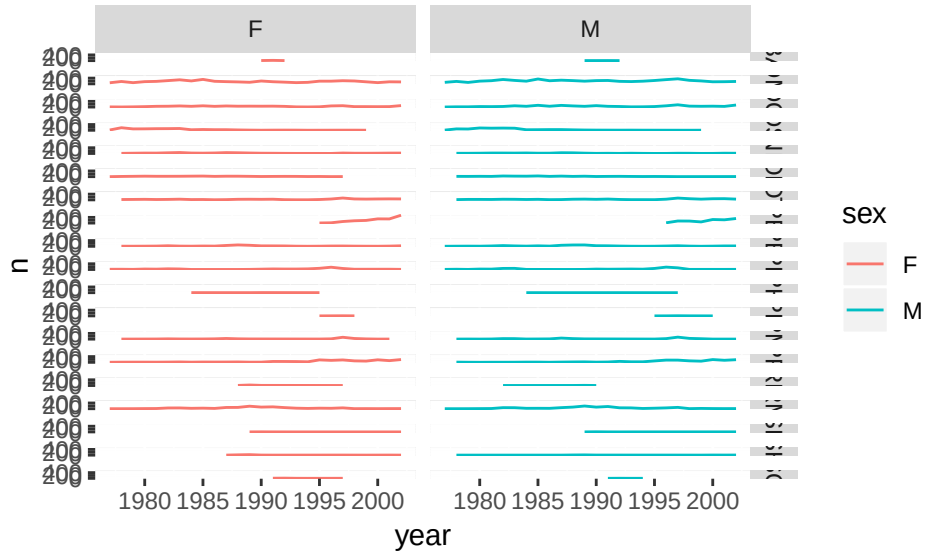
- horizontal split

```
ggplot(year_summary, aes(x = year, y = n)) +
    geom_line(aes(color=sex)) +
    facet_grid(~species_id)
```



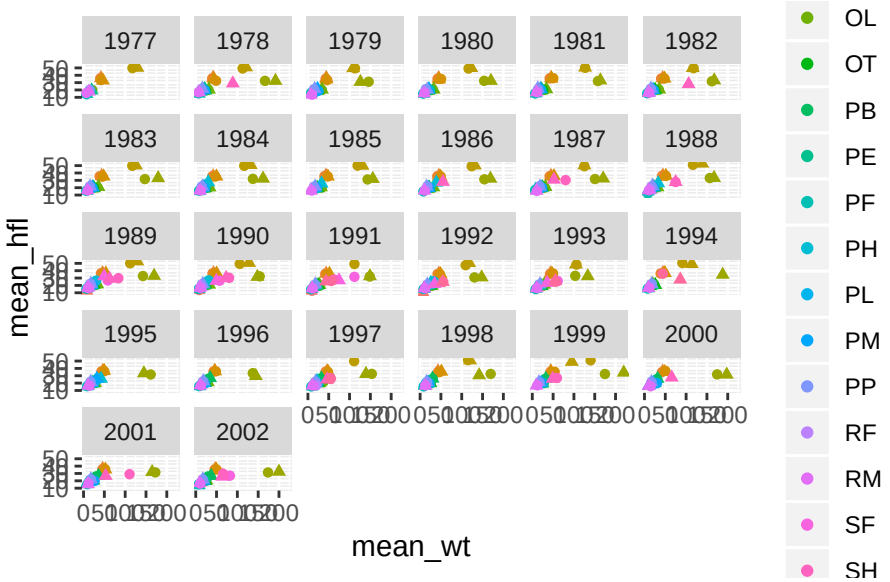
- separate panel for each sex and species

```
ggplot(year_summary, aes(x = year, y = n)) +
    geom_line(aes(color=sex)) +
    facet_grid(species_id ~ sex)
```



- **CHALLENGE 4:**
  - **Use the `year_summary` dataset and make scatterplots of mean hindfoot length vs. mean weight (with each point being a species), facetting by year.**
    - **Use aesthetics `x=mean_wt` and `y=mean_hfl`**
    - **Use geom_point(aes(color=species_id, shape=sex))**
    - **Use `facet_wrap(~year)`**

```
ggplot(year_summary, aes(x=mean_wt, y=mean_hfl)) +
    geom_point(aes(color=species_id, shape=sex)) +
    facet_wrap(~year)
```

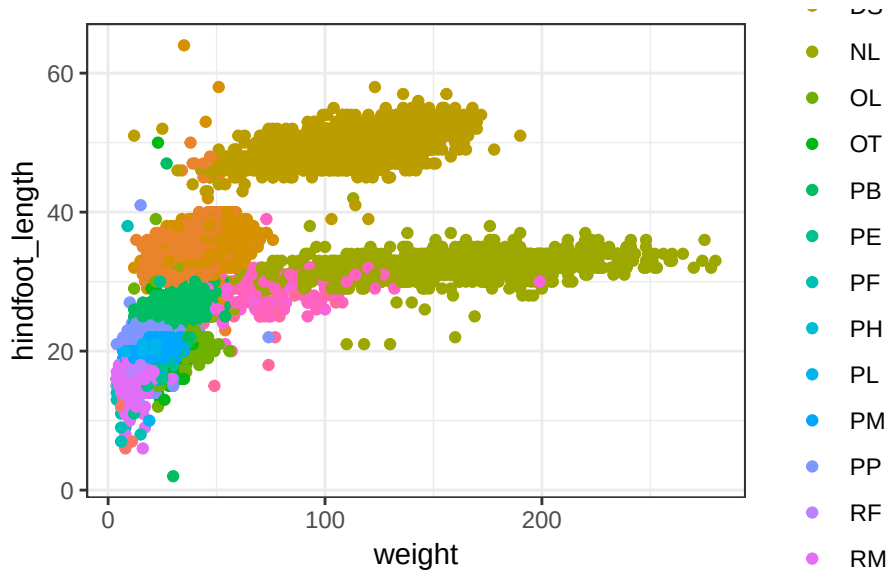


Themes

- the grey background is contested. You can apply a variety of themes to the overall appearance of the plot.

For example:

```
#surveys %>% filter(species_id %in% c("DM", "DS", "DO")) %>%
ggplot(surveys,aes(x=weight, y=hindfoot_length)) +
    geom_point(aes(color=species_id)) +
    theme_bw()
```



Saving Plots

- use the `ggsave()` function
- default is to save the last plot you created.
  - good practice to save plot as an object and pass that to `ggsave()`

```
p <- ggplot(surveys, aes(x=weight, y=hindfoot_length)) + geom_point()
ggsave("scatter.png", p, height=6, width=8)
```

To create a different filetype, just change the extension. FOr example, to make a pdf:

```
ggsave("scatter.pdf", p, height=6, width=8)
```