Data visualization

ggplot2

Matilde Trevisani

Data visualization

Data visualization

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

- Data visualization is the creation and study of the visual representation of data
- Many tools for visualizing data -- R is one of them
- Many approaches/systems within R for making data visualizations -- **ggplot2** is one of them, and that's what we're going to use

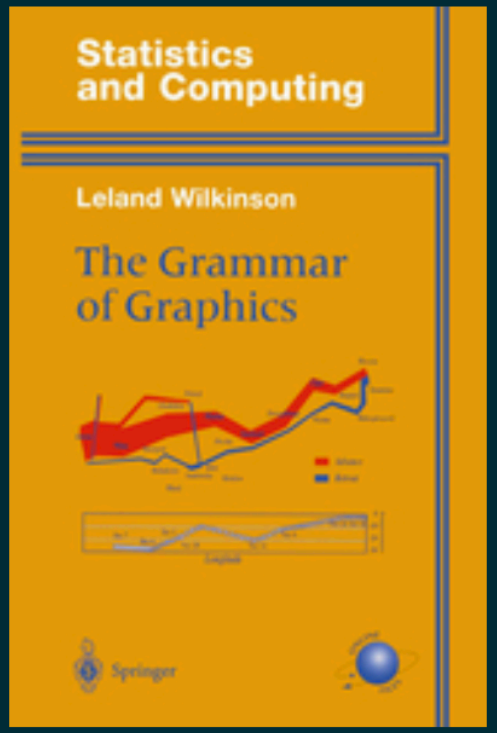
ggplot2 ∈ tidyverse

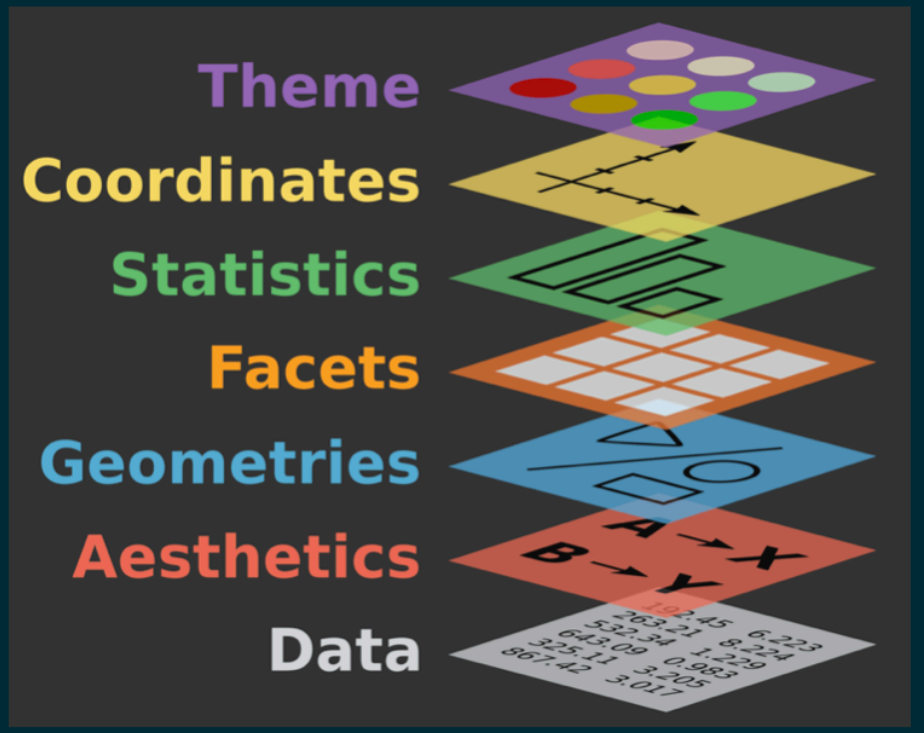
- **ggplot2** is tidyverse's data visualization package
- gg in "ggplot2" stands for Grammar of Graphics
- Inspired by the book **Grammar of Graphics** by Leland Wilkinson
- Structure of the code for plots can be summarized as

```
ggplot(data = [dataset],
       mapping = aes(x = [x-variable],
                     y = [y-variable])) +
   geom_xxx() +
   other options
```

Grammar of Graphics

A grammar of graphics is a tool that enables us to concisely describe the components of a graphic





Source: BloggoType

What's in the Star Wars data?

Take a `glimpse` at the data:

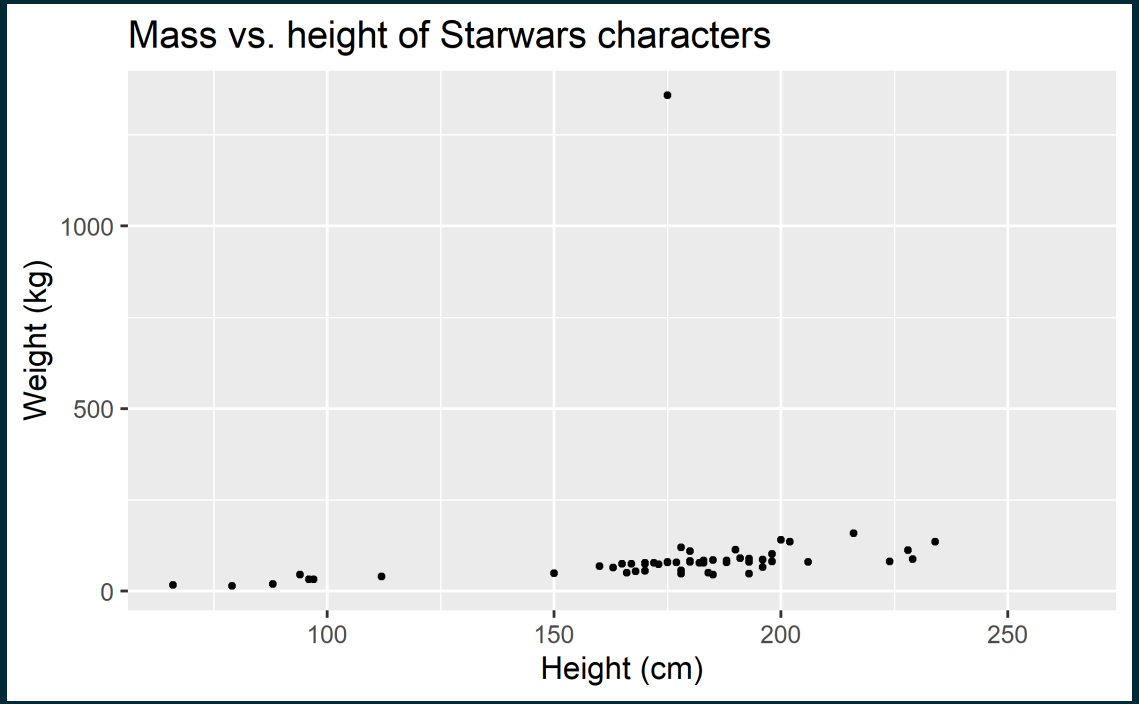
```
glimpse(starwars)
```

```
## Rows: 87
## Columns: 14
## $ name       <chr> "Luke Skywalker", "C-3PO", "R2-D2", "Darth V…
## $ height     <int> 172, 167, 96, 202, 150, 178, 165, 97, 183, 1…
## $ mass       <dbl> 77.0, 75.0, 32.0, 136.0, 49.0, 120.0, 75.0, …
## $ hair_color <chr> "blond", NA, NA, "none", "brown", "brown, gr…
## $ skin_color <chr> "fair", "gold", "white, blue", "white", "lig…
## $ eye_color  <chr> "blue", "yellow", "red", "yellow", "brown", …
## $ birth_year <dbl> 19.0, 112.0, 33.0, 41.9, 19.0, 52.0, 47.0, N…
## $ sex        <chr> "male", "none", "none", "male", "female", "m…
## $ gender     <chr> "masculine", "masculine", "masculine", "masc…
## $ homeworld  <chr> "Tatooine", "Tatooine", "Naboo", "Tatooine",…
## $ species    <chr> "Human", "Droid", "Droid", "Human", "Human",…
## $ films      <list> <"A New Hope", "The Empire Strikes Back", "…
## $ vehicles   <list> <"Snowspeeder", "Imperial Speeder Bike">, <…
## $ starships  <list> <"X-wing", "Imperial shuttle">, <>, <>, "TI…
```

Mass vs. height

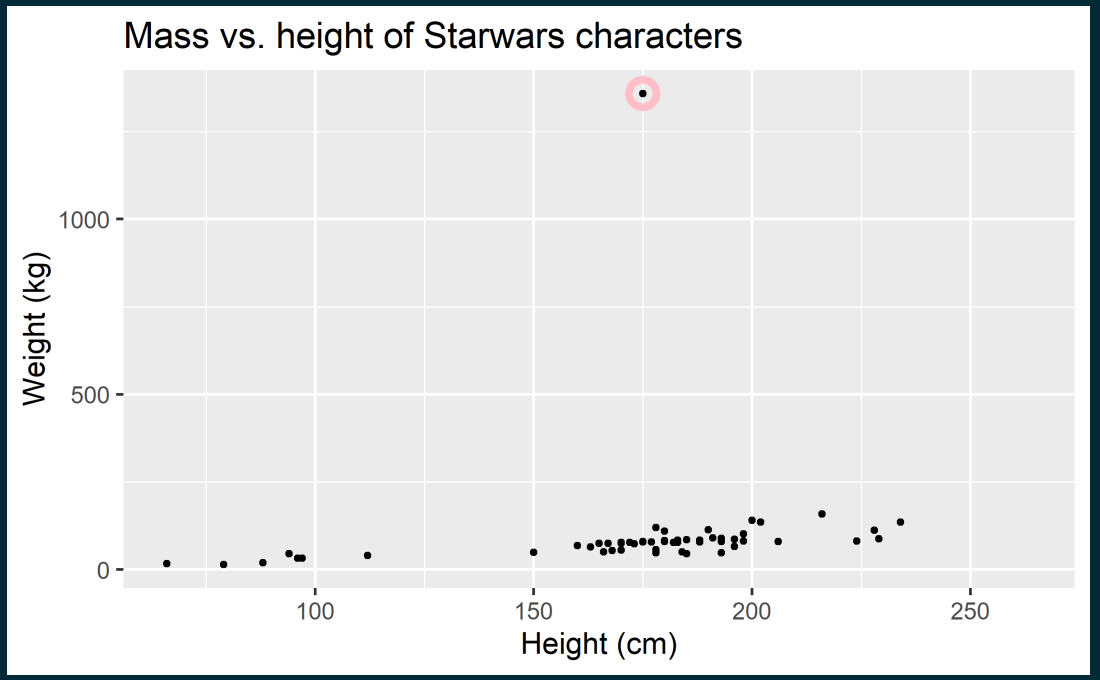
```
ggplot(data = starwars, mapping = aes(x = height, y = mass)) +
  geom_point() +
  labs(title = "Mass vs. height of Starwars characters",
       x = "Height (cm)", y = "Weight (kg)")
```

```
## Warning: Removed 28 rows containing missing values or values outside the
## scale range (`geom_point()`).
```

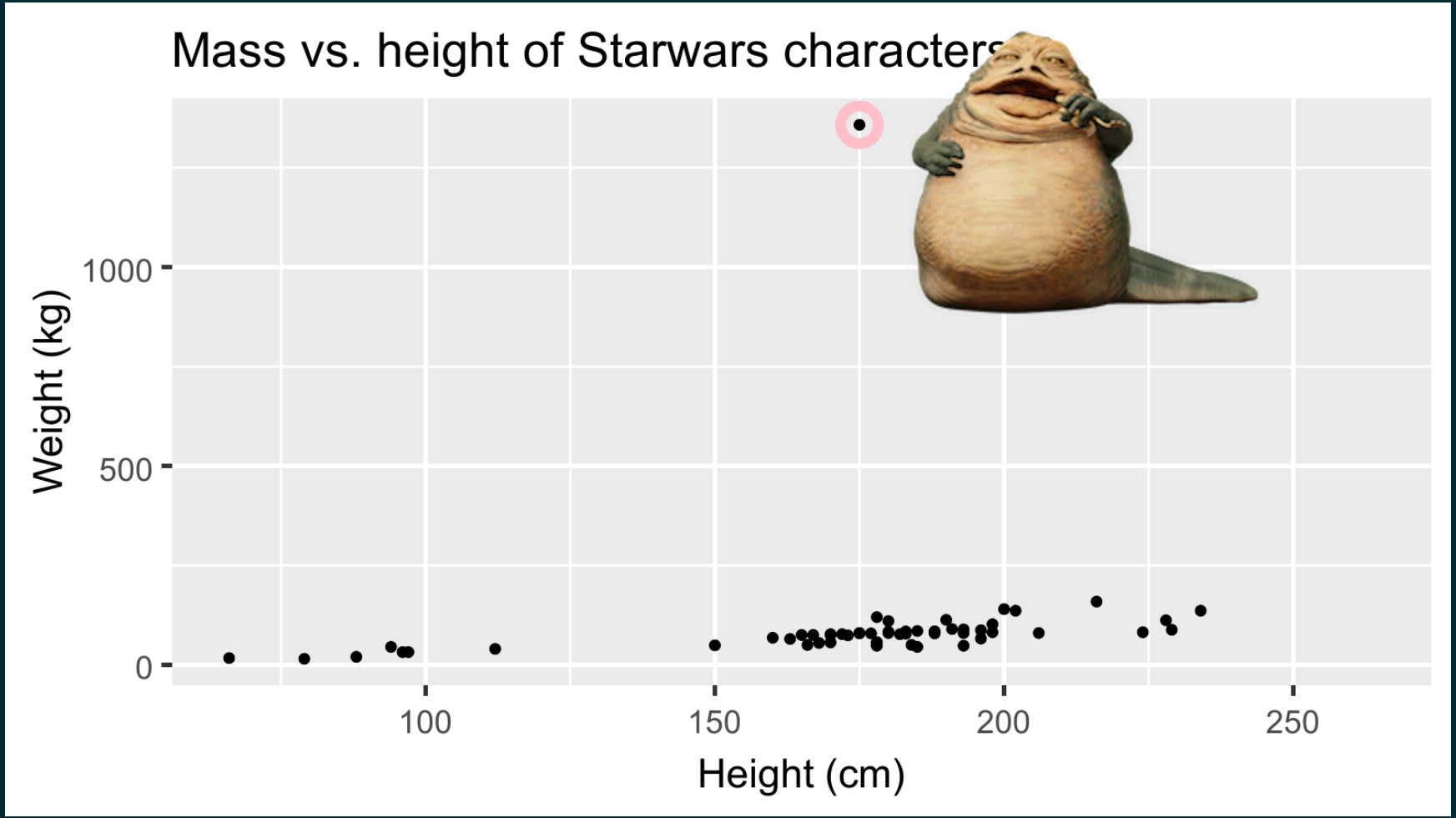


Mass vs. height interpretation

How would you describe the relationship between mass and height of Starwars characters? Who is the not so tall but really chubby character? What other variables would help us understand data points that don't follow the overall trend?



Who is the outlier? Jabba!



Coding

- What are the functions doing the plotting?
- What is the dataset being plotted?
- Which variables map to which features (aesthetics) of the plot?
- What does the warning mean?[+]

```
ggplot(data = starwars, mapping = aes(x = height, y = mass)) +
  geom_point() +
  labs(title = "Mass vs. height of Starwars characters",
       x = "Height (cm)", y = "Weight (kg)")
```

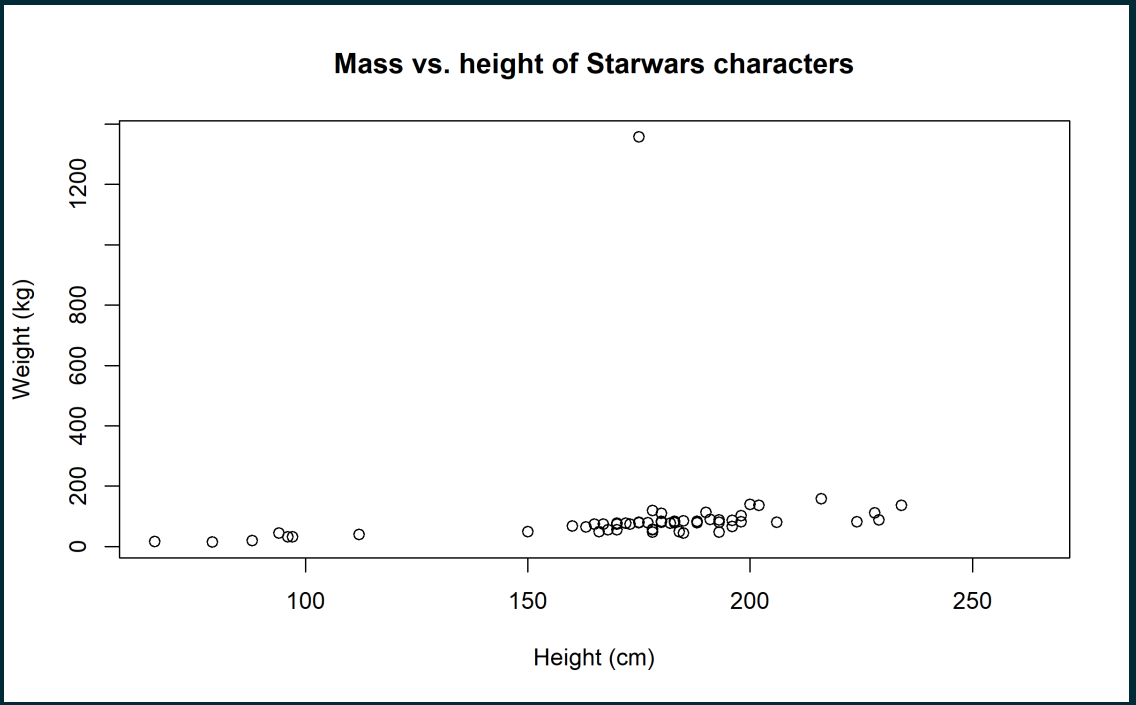
```
## Warning: Removed 28 rows containing missing values or values outside the
## scale range (`geom_point()`).
```

[+] Suppressing warning to subsequent slides to save space

Compare to basic R

```
plot(x, y = NULL, type = "p",  xlim = NULL, ylim = NULL, log = "", main = NULL, sub = NULL,
xlab = NULL, ylab = NULL, ann = par("ann"), axes = TRUE, frame.plot = axes, panel.first = NULL, pa
     ...)
```

```
plot(starwars$height, starwars$mass,
main = "Mass vs. height of Starwars characters", xlab = "Height (cm)", ylab = "Weight (kg)")
```



Hello ggplot2!

- `ggplot()` is the main function in ggplot2
- Plots are constructed in layers
- Structure of the code for plots can be summarized as

```
ggplot(data = [dataset],
       mapping = aes(x = [x-variable], y = [y-variable])) +
   geom_xxx() +
   other options
```

- The ggplot2 package comes with the tidyverse

```
library(tidyverse)
```

- For help with ggplot2, see ggplot2.tidyverse.org

Why do we visualize?

Anscombe's quartet

```
##    set  x     y
## 1    I 10  8.04
## 2    I  8  6.95
## 3    I 13  7.58
## 4    I  9  8.81
## 5    I 11  8.33
## 6    I 14  9.96
## 7    I  6  7.24
## 8    I  4  4.26
## 9    I 12 10.84
## 10   I  7  4.82
## 11   I  5  5.68
## 12  II 10  9.14
## 13  II  8  8.14
## 14  II 13  8.74
## 15  II  9  8.77
## 16  II 11  9.26
## 17  II 14  8.10
## 18  II  6  6.13
## 19  II  4  3.10
## 20  II 12  9.13
## 21  II  7  7.26
## 22  II  5  4.74
```

```
##    set  x     y
## 23 III 10  7.46
## 24 III  8  6.77
## 25 III 13 12.74
## 26 III  9  7.11
## 27 III 11  7.81
## 28 III 14  8.84
## 29 III  6  6.08
## 30 III  4  5.39
## 31 III 12  8.15
## 32 III  7  6.42
## 33 III  5  5.73
## 34  IV  8  6.58
## 35  IV  8  5.76
## 36  IV  8  7.71
## 37  IV  8  8.84
## 38  IV  8  8.47
## 39  IV  8  7.04
## 40  IV  8  5.25
## 41  IV 19 12.50
## 42  IV  8  5.56
## 43  IV  8  7.91
## 44  IV  8  6.89
```

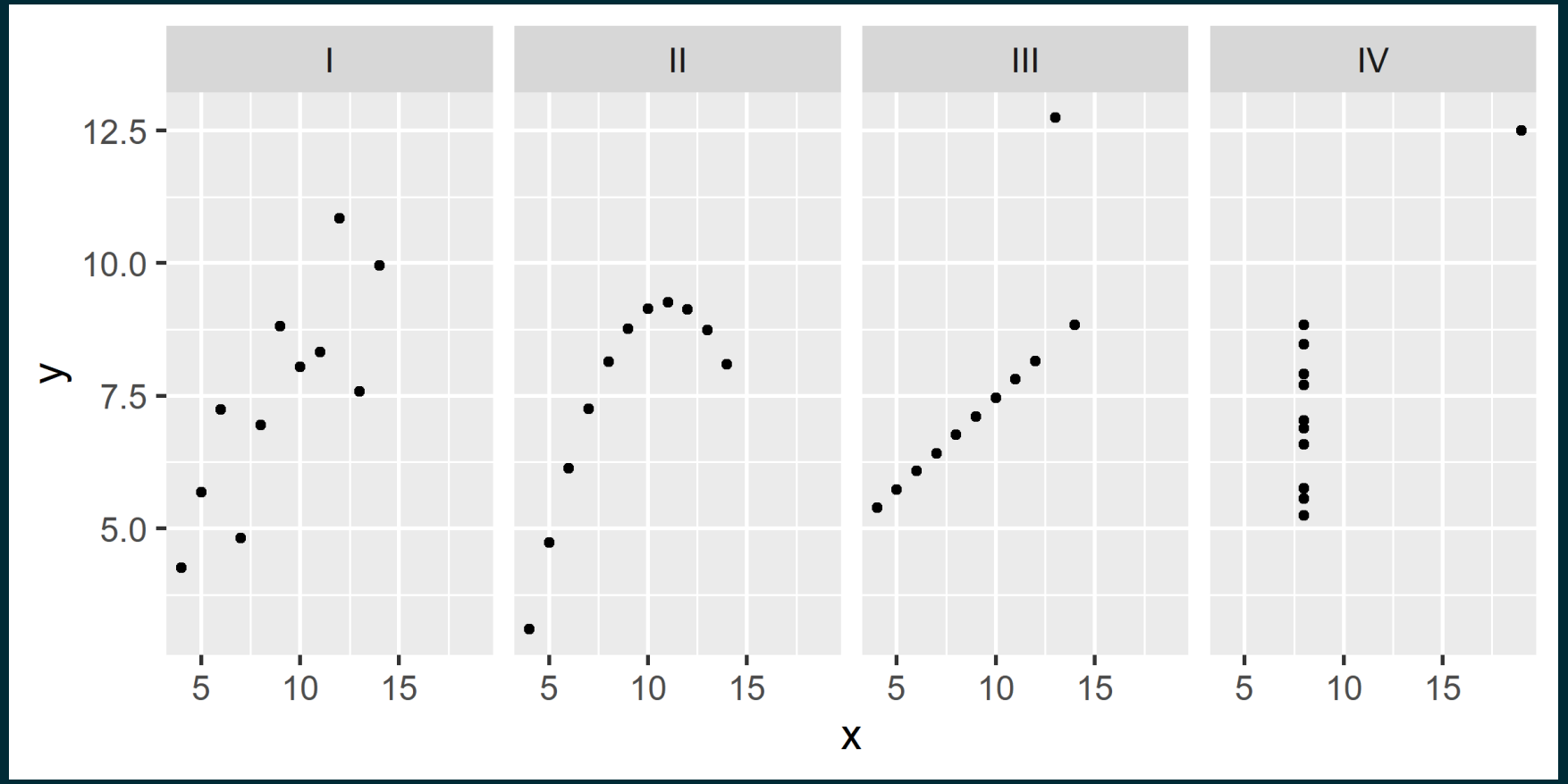
Summarising Anscombe's quartet

```
quartet %>%
  group_by(set) %>%
  summarise(
    mean_x = mean(x),
    mean_y = mean(y),
    sd_x = sd(x),
    sd_y = sd(y),
    r = cor(x, y)
  )
```

```
## # A tibble: 4 × 6
##   set   mean_x mean_y  sd_x  sd_y     r
##   <fct>  <dbl>  <dbl> <dbl> <dbl> <dbl>
## 1 I          9   7.50  3.32  2.03 0.816
## 2 II         9   7.50  3.32  2.03 0.816
## 3 III        9   7.5   3.32  2.03 0.816
## 4 IV         9   7.50  3.32  2.03 0.817
```

Visualizing Anscombe's quartet

```
ggplot(quartet, aes(x = x, y = y)) +
  geom_point() +
  facet_wrap(~ set, ncol = 4)
```



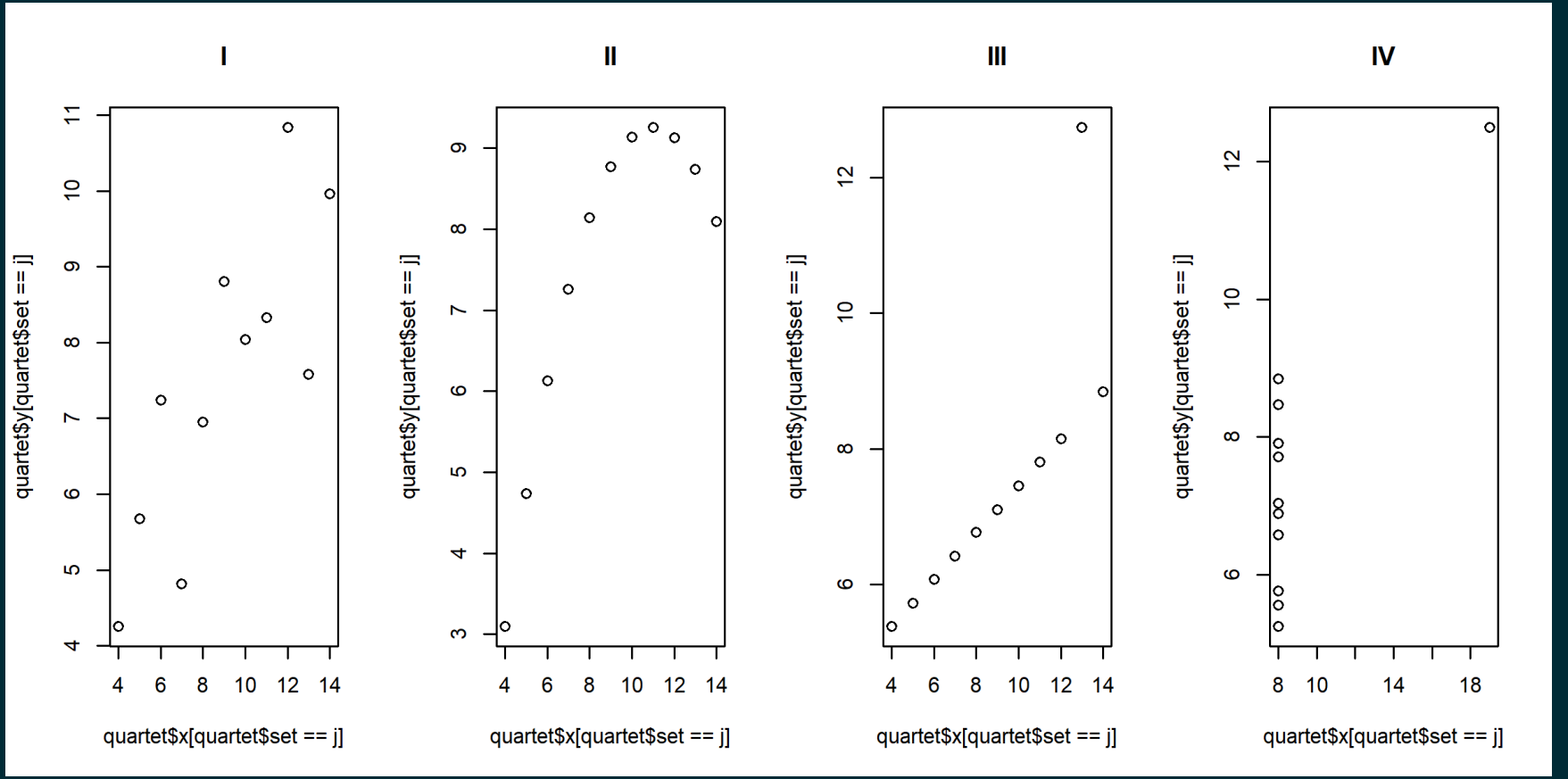
Compare to basic R

```
str(quartet)
```

```
## 'data.frame':    44 obs. of  3 variables:
##  $ set: Factor w/ 4 levels "I","II","III",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ x  : int  10 8 13 9 11 14 6 4 12 7 ...
##  $ y  : num  8.04 6.95 7.58 8.81 8.33 ...
```

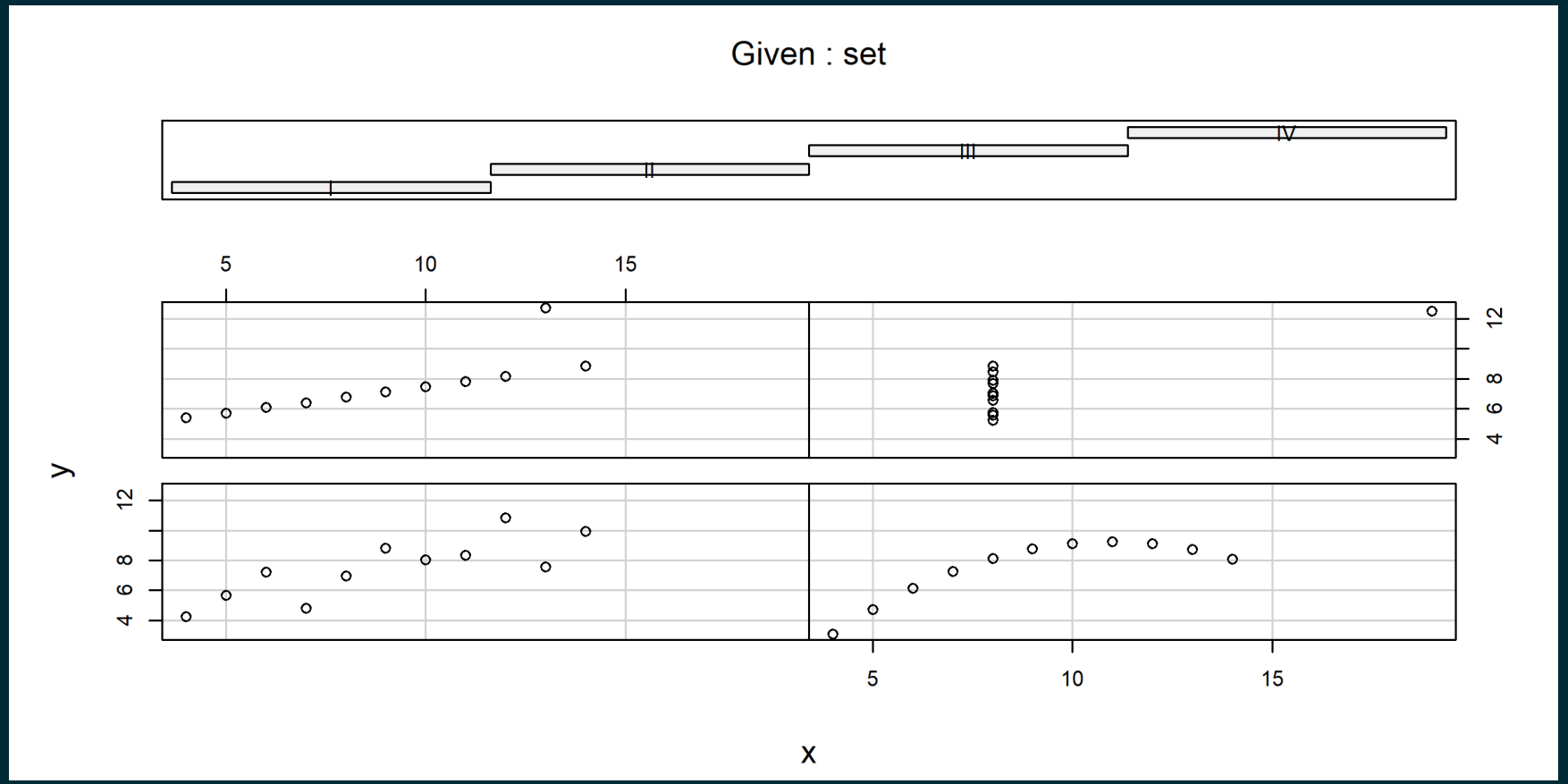
```
par(mfrow=c(1,4))

for(j in levels(set))
  plot(quartet$x[quartet$set==j], quartet$y[quartet$set==j], main=j)
```

Conditional plots



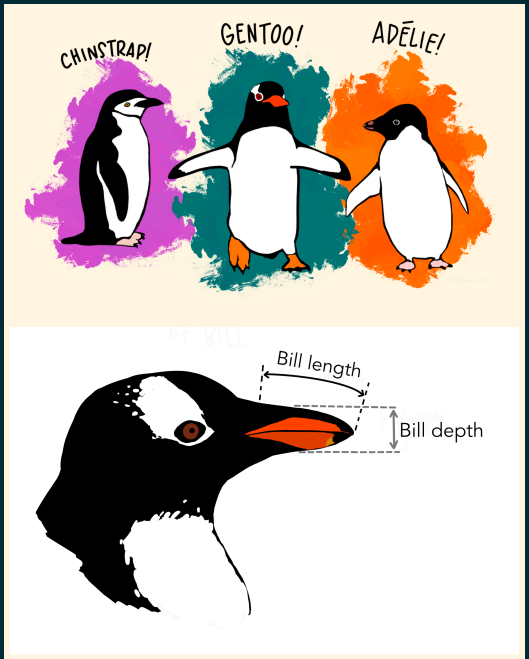
Conditional plots - 2

```
coplot(y ~ x | set, data = quartet)
```



Data: Palmer Penguins

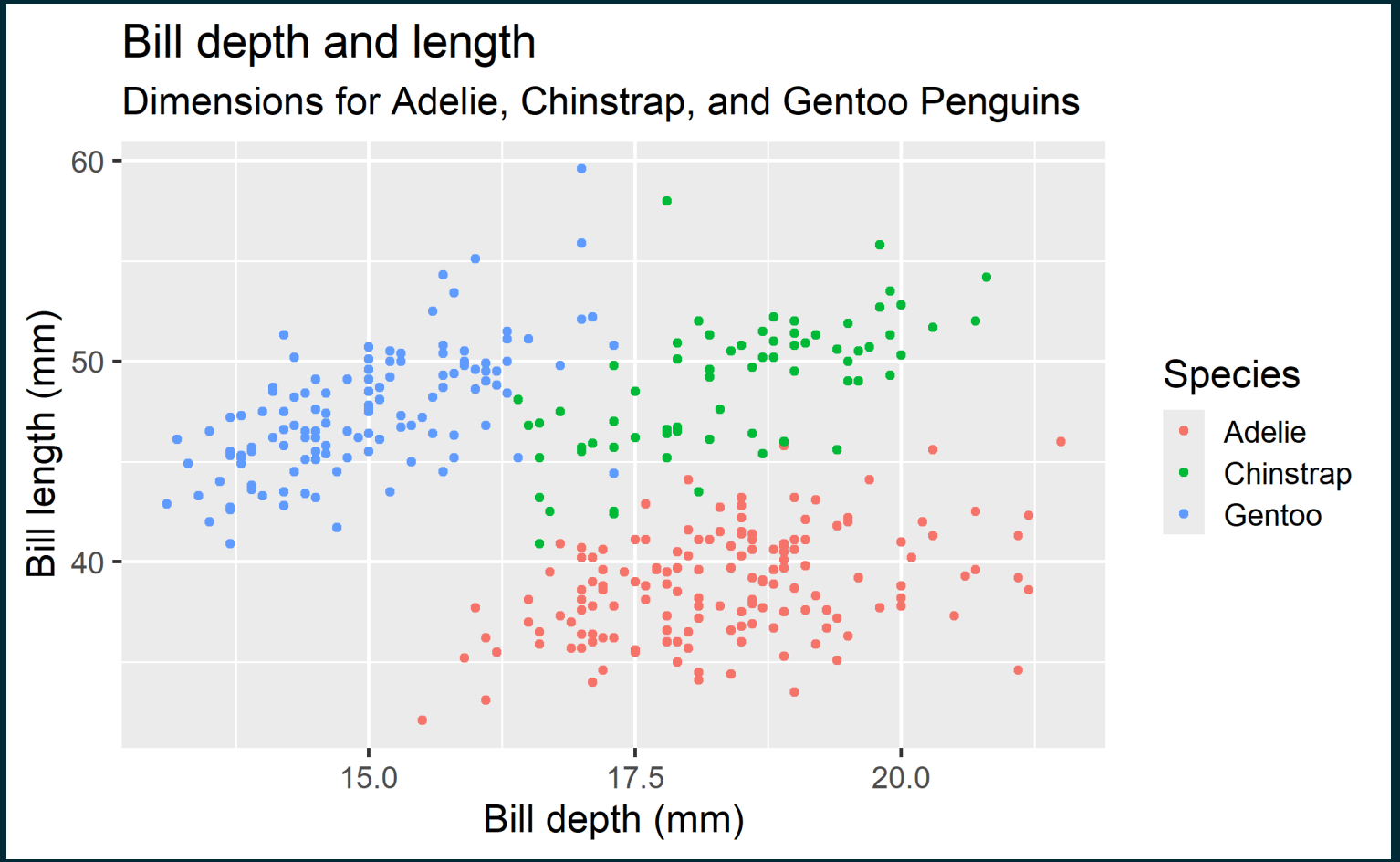
Measurements for penguin species, island in Palmer Archipelago, size (flipper length, body mass, bill dimensions), and sex.




```
library(palmerpenguins)
glimpse(penguins)
```

```
## Rows: 344
## Columns: 8
## $ species           <fct> Adelie, Adelie, Adelie, Adelie, Adeli…
## $ island            <fct> Torgersen, Torgersen, Torgersen, Torg…
## $ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.…
## $ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.…
## $ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195…
## $ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 362…
## $ sex               <fct> male, female, female, NA, female, mal…
## $ year              <int> 2007, 2007, 2007, 2007, 2007, 2007, 2…
```

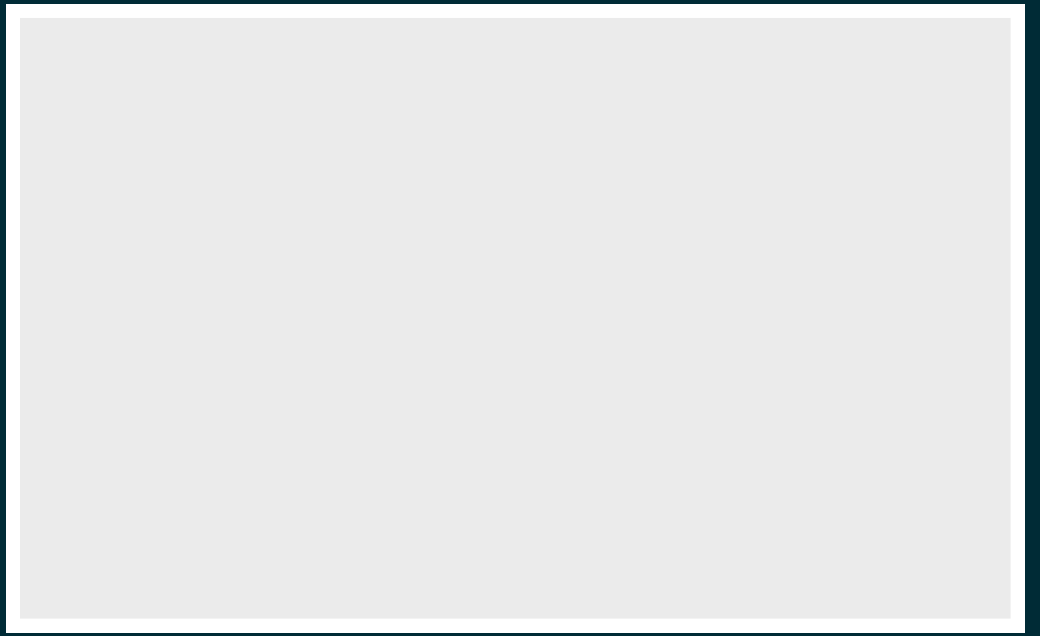
Plot Code


Bill depth and length
Dimensions for Adelie, Chinstrap, and Gentoo Penguins
60
50
40
Bill length (mm)
15.0
17.5
20.0
Bill depth (mm)
Species
Adelie
Chinstrap
Gentoo

Coding out loud

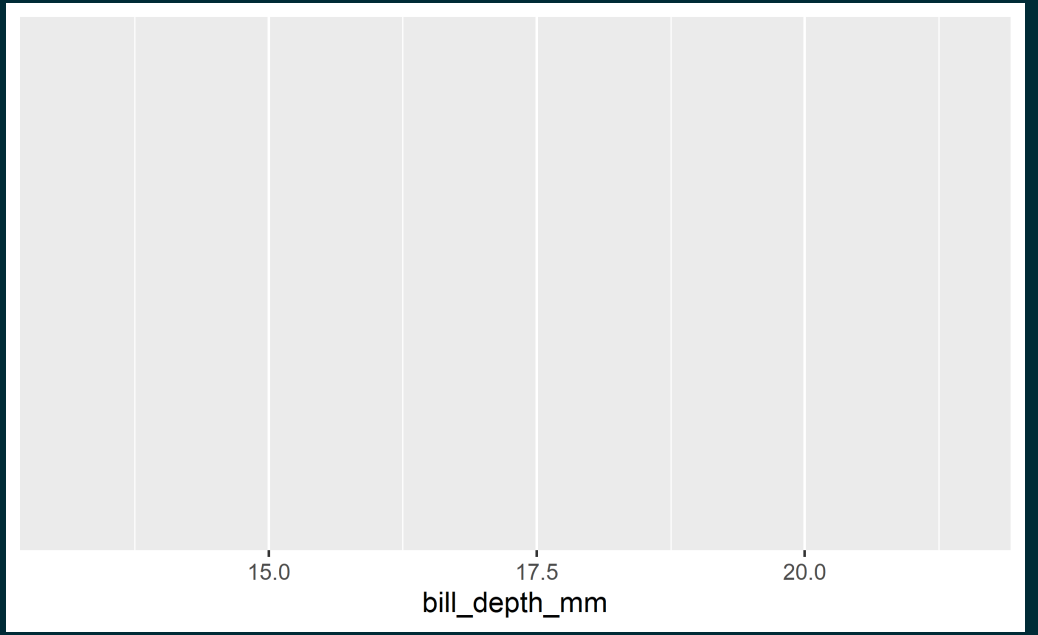
Start with the `penguins` data frame

```
ggplot(data = penguins)
```



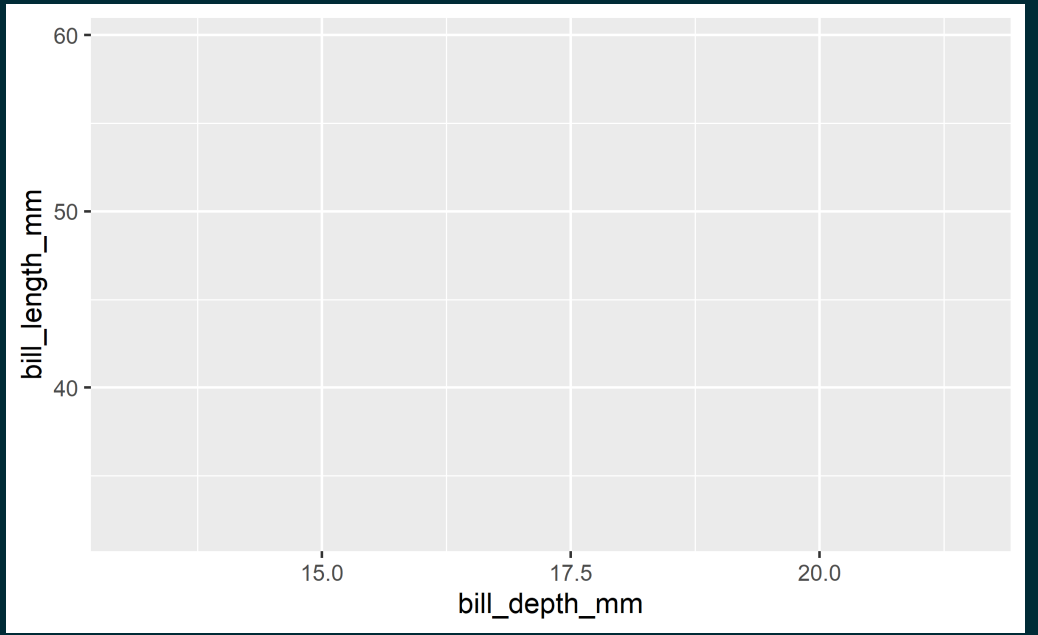
Start with the `penguins` data frame, **map bill depth to the x-axis**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm))
```



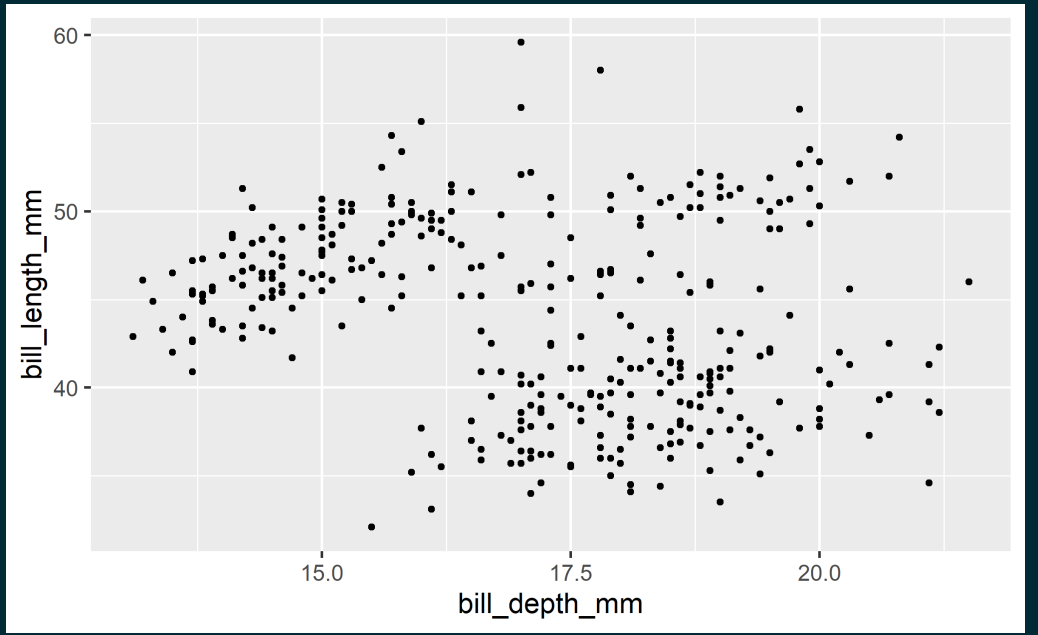
> Start with the `penguins` data frame, map bill depth to the x-axis **and map bill length to the y-axis.**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm))
```



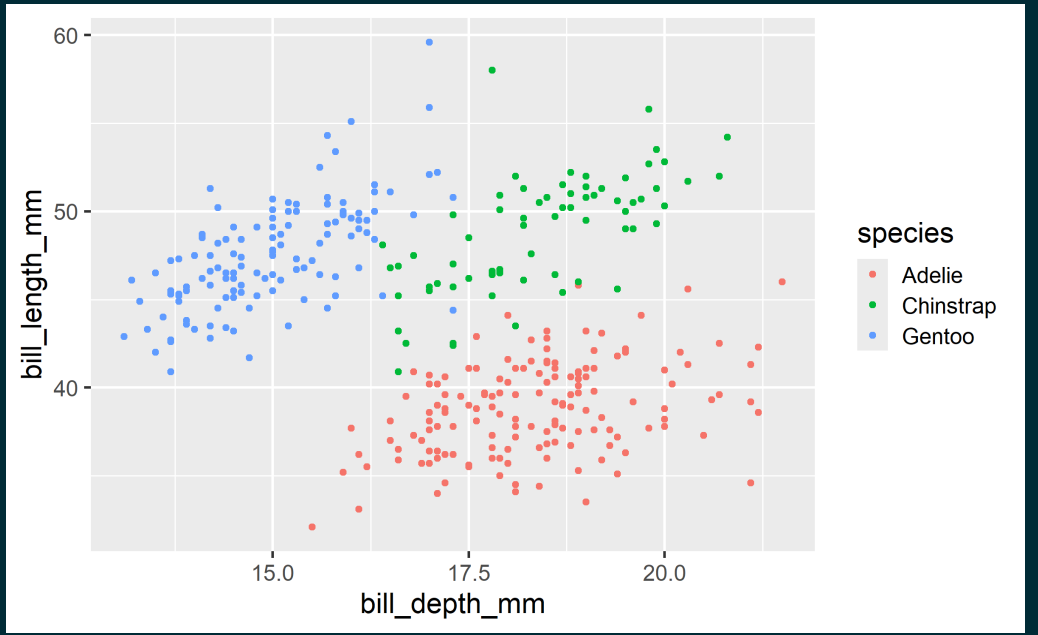
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. **Represent each observation with a point**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point()
```



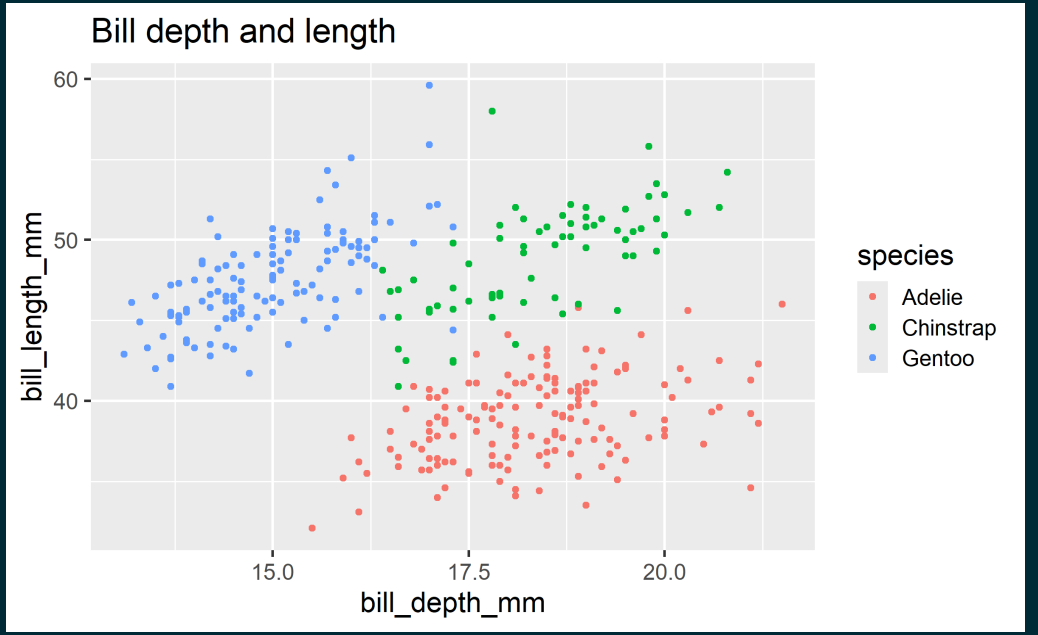
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point **and map species to the colour of each point.**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point()
```



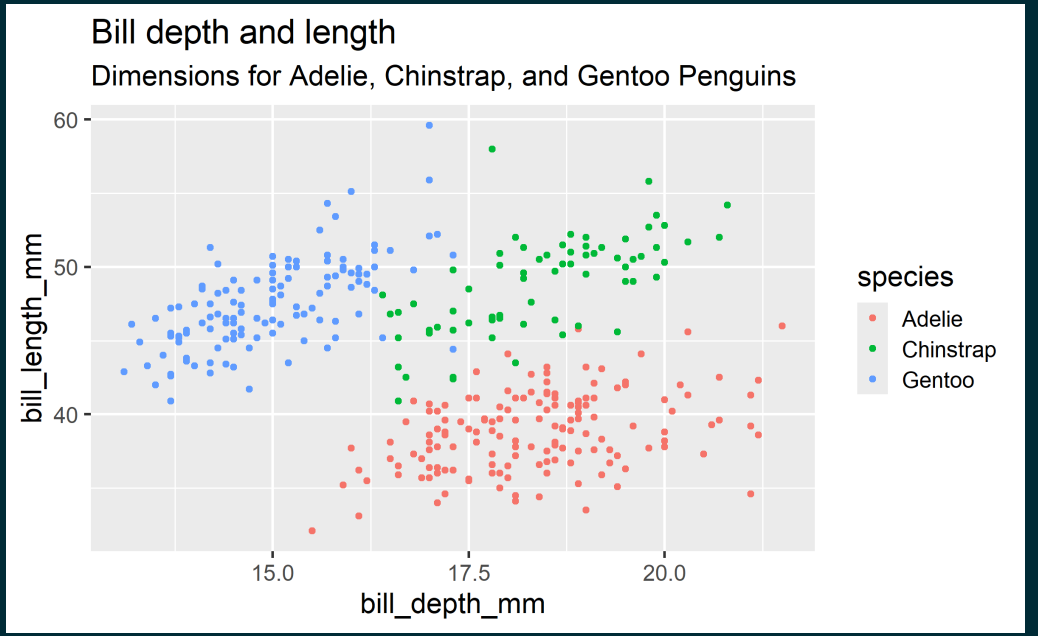
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. **Title the plot "Bill depth and length"**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length")
```



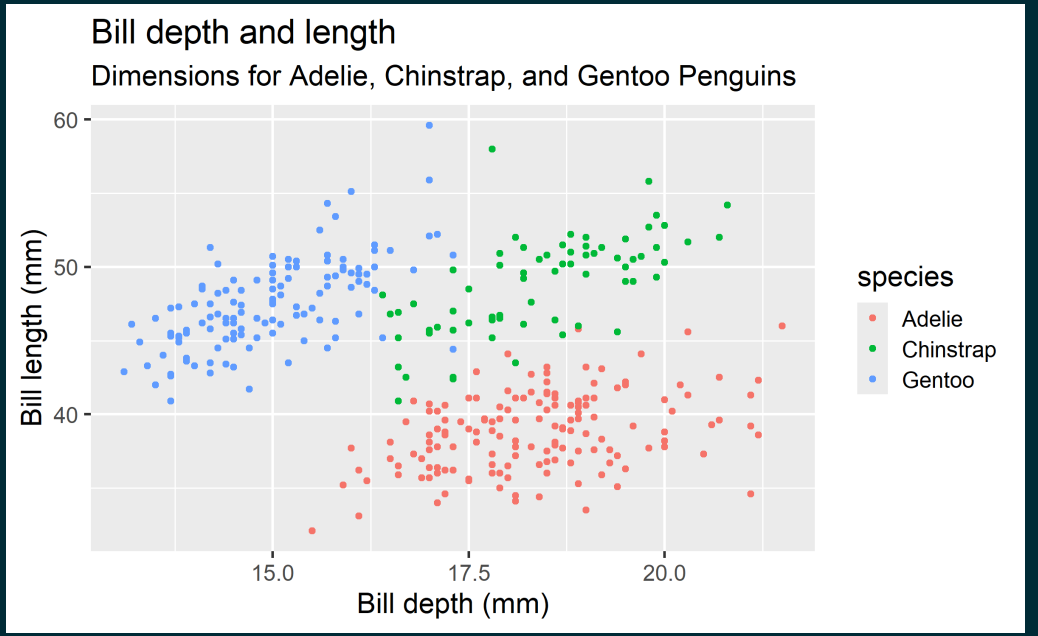
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", **add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins"**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chi
```



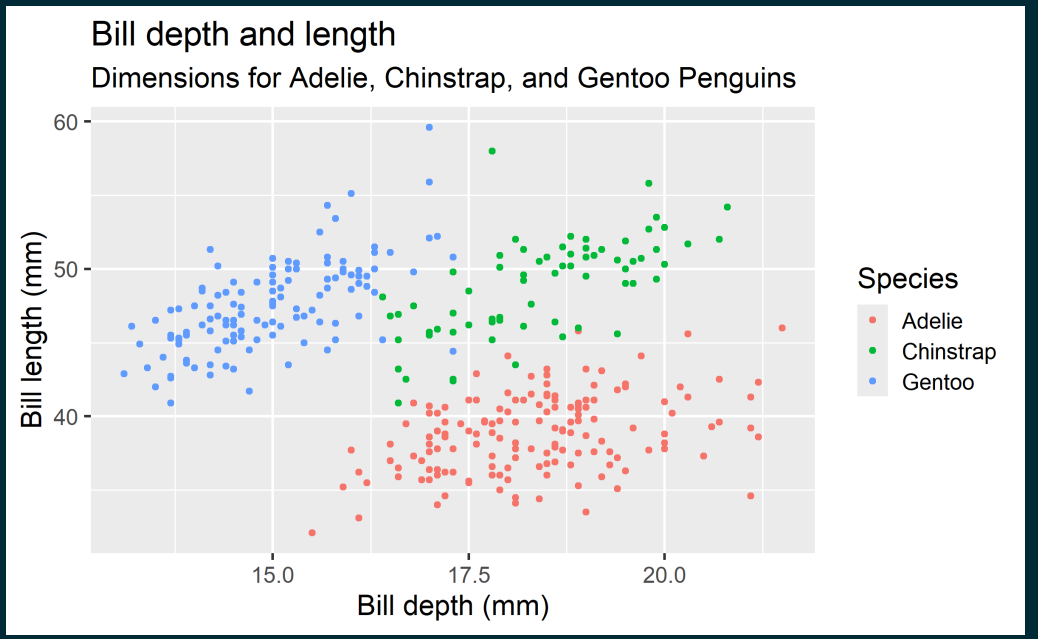
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins", **label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chi
       x = "Bill depth (mm)", y = "Bill lengt
```



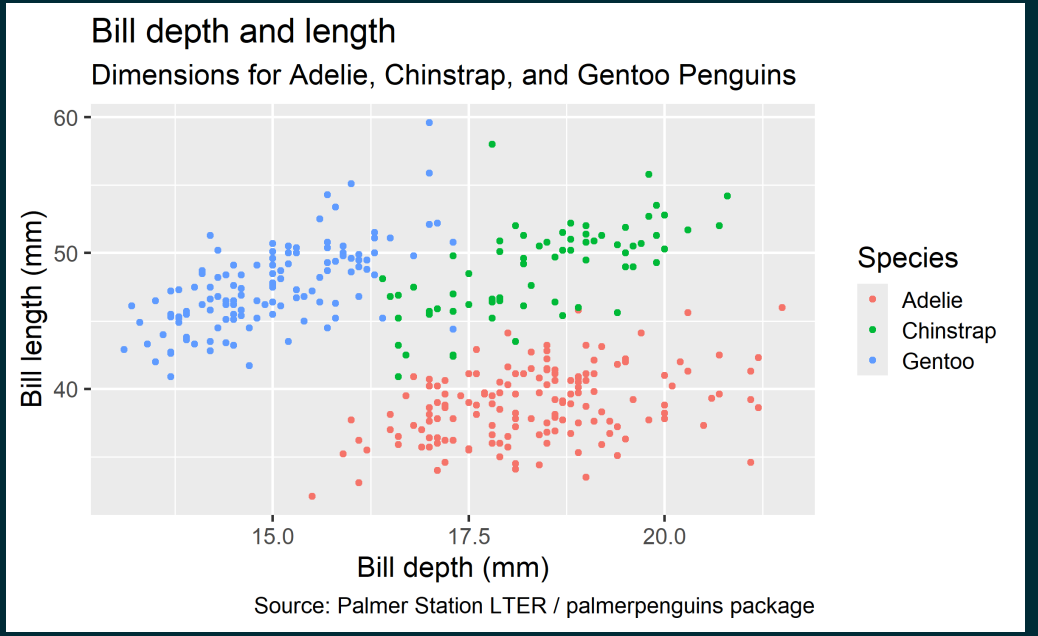
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins", label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively, **label the legend "Species"**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chir
       x = "Bill depth (mm)", y = "Bill length
       colour = "Species")
```



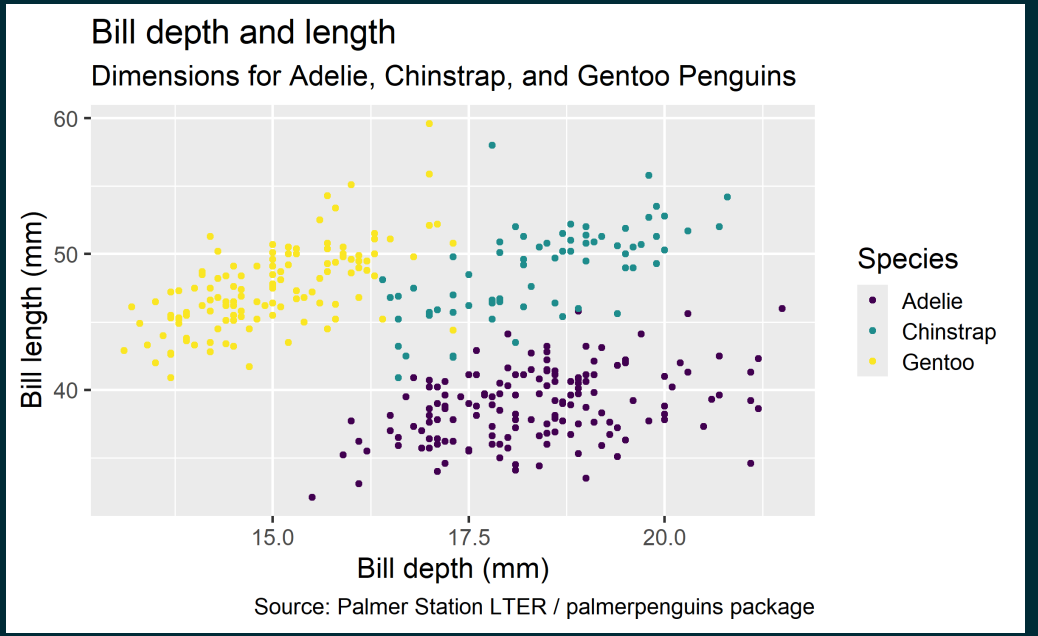
> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins", label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively, label the legend "Species", **and add a caption for the data source.**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chir
       x = "Bill depth (mm)", y = "Bill lengt
       colour = "Species",
       caption = "Source: Palmer Station LTER
```

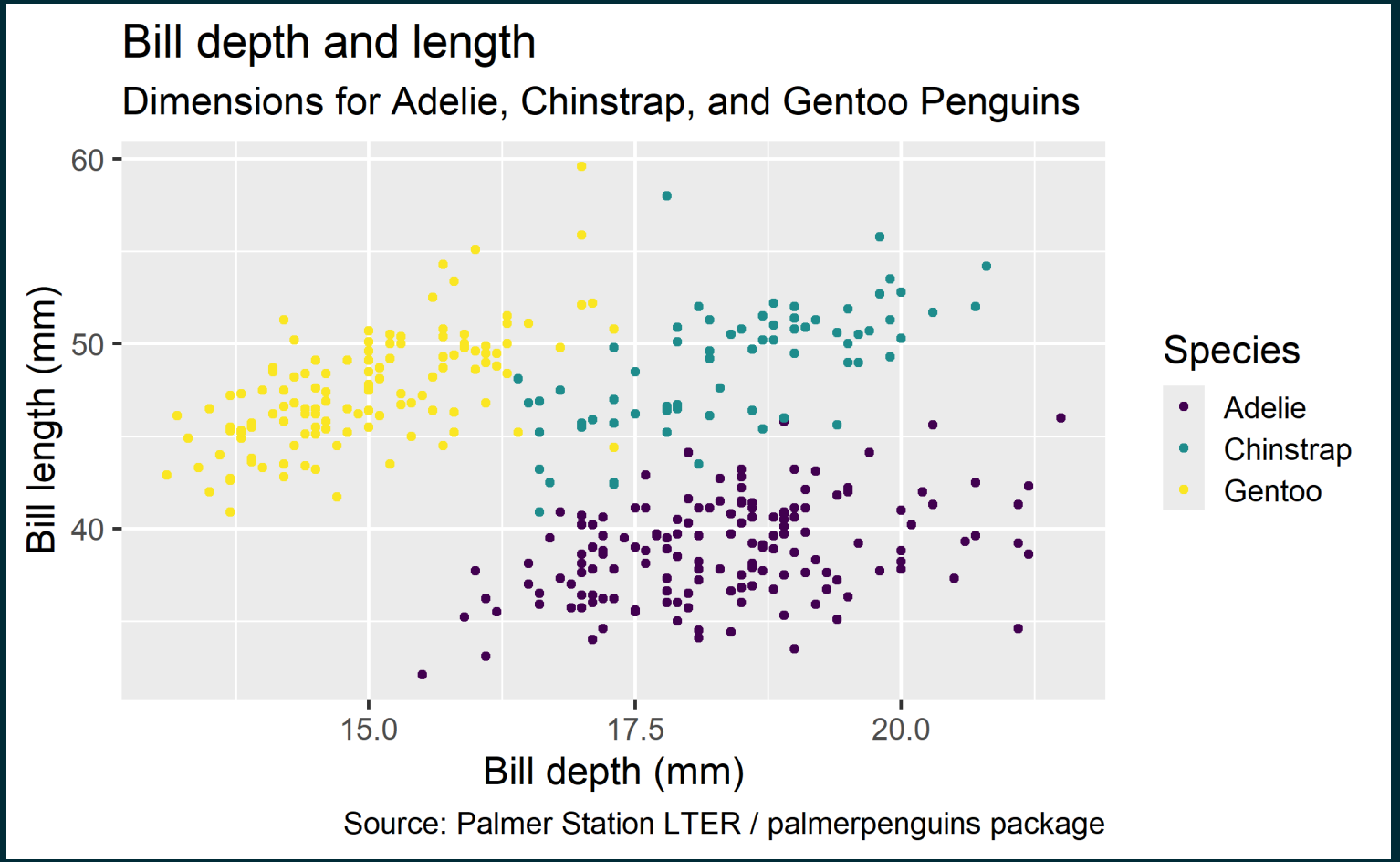


> Start with the `penguins` data frame, map bill depth to the x-axis and map bill length to the y-axis. Represent each observation with a point and map species to the colour of each point. Title the plot "Bill depth and length", add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins", label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively, label the legend "Species", and add a caption for the data source. **Finally, use a discrete colour scale that is designed to be perceived by viewers with common forms of colour blindness.**

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  labs(title = "Bill depth and length",
       subtitle = "Dimensions for Adelie, Chir
       x = "Bill depth (mm)", y = "Bill lengt
       colour = "Species",
       caption = "Source: Palmer Station LTER
  scale_colour_viridis_d()
```



Plot Code Narrative



Argument names

You can omit the names of first two arguments when building plots with `ggplot()`.

```
ggplot(data = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     colour = species)) +
  geom_point() +
  scale_colour_viridis_d()
```

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species)) +
  geom_point() +
  scale_colour_viridis_d()
```

Aesthetics

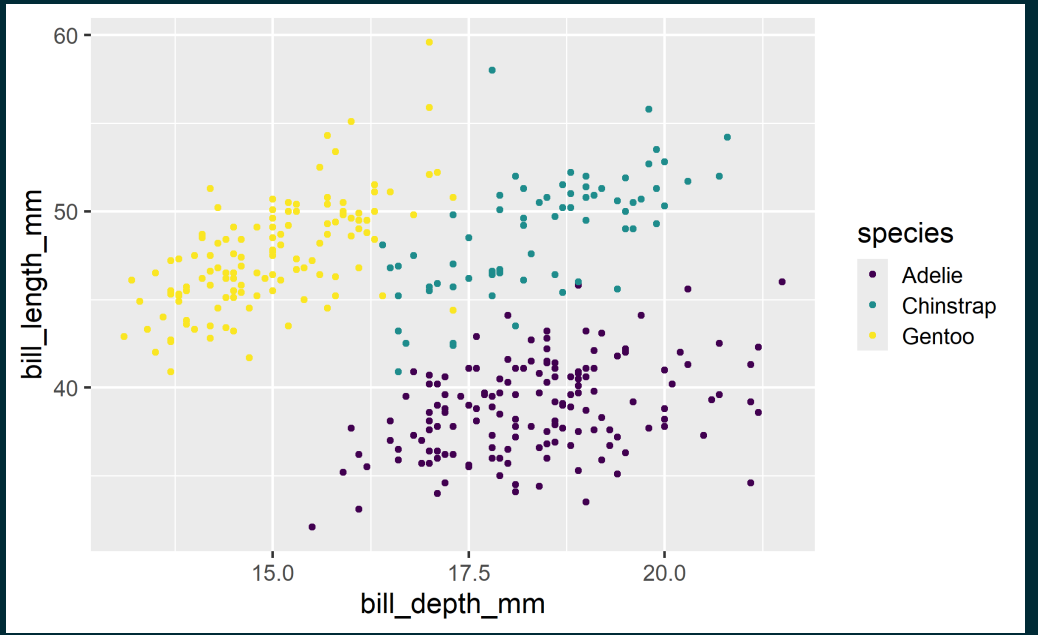
Aesthetics options

Commonly used **visual properties** (aesthetics) of geoms which variables in the data are mapped to are

- `colour`
- `shape`
- `size`
- `alpha` (transparency)

Colour

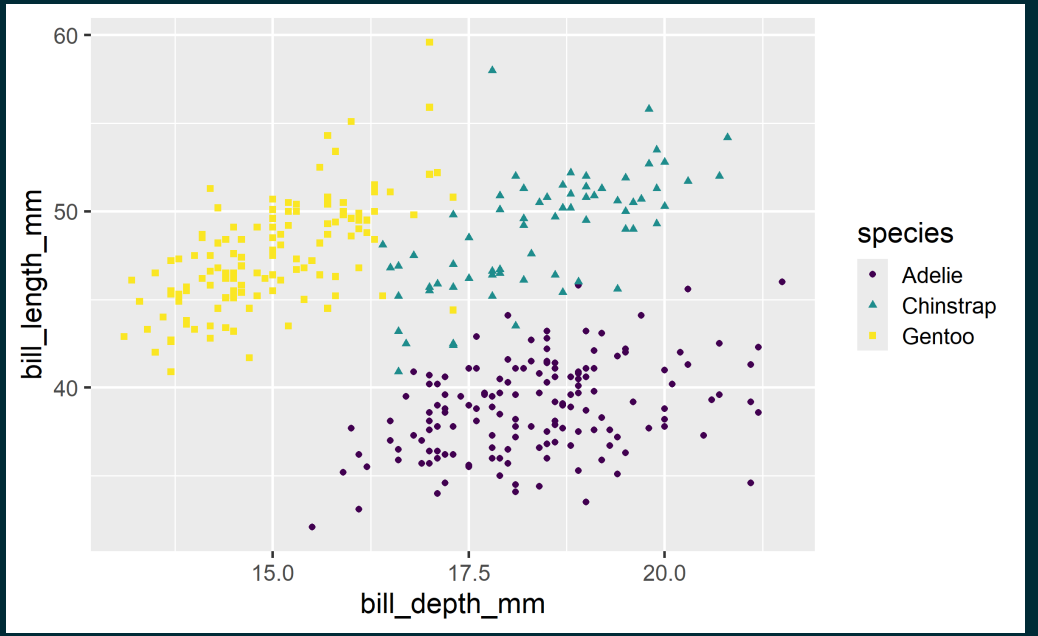
```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species)) +
  geom_point() +
  scale_colour_viridis_d()
```



Shape

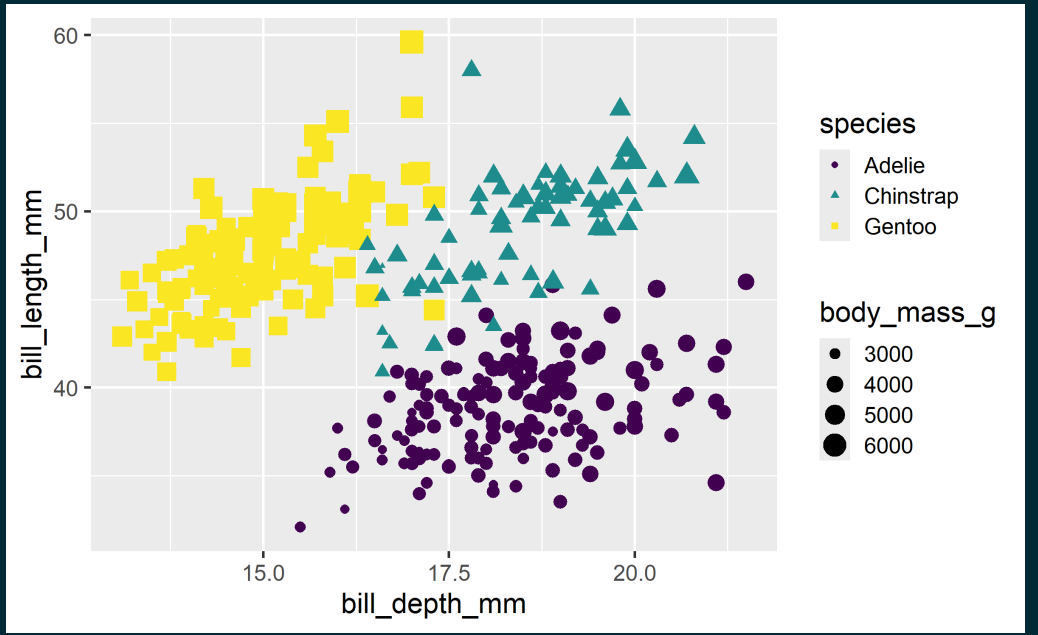
Mapped to same variable as `colour`

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species,
           shape = species)) +
  geom_point() +
  scale_colour_viridis_d()
```



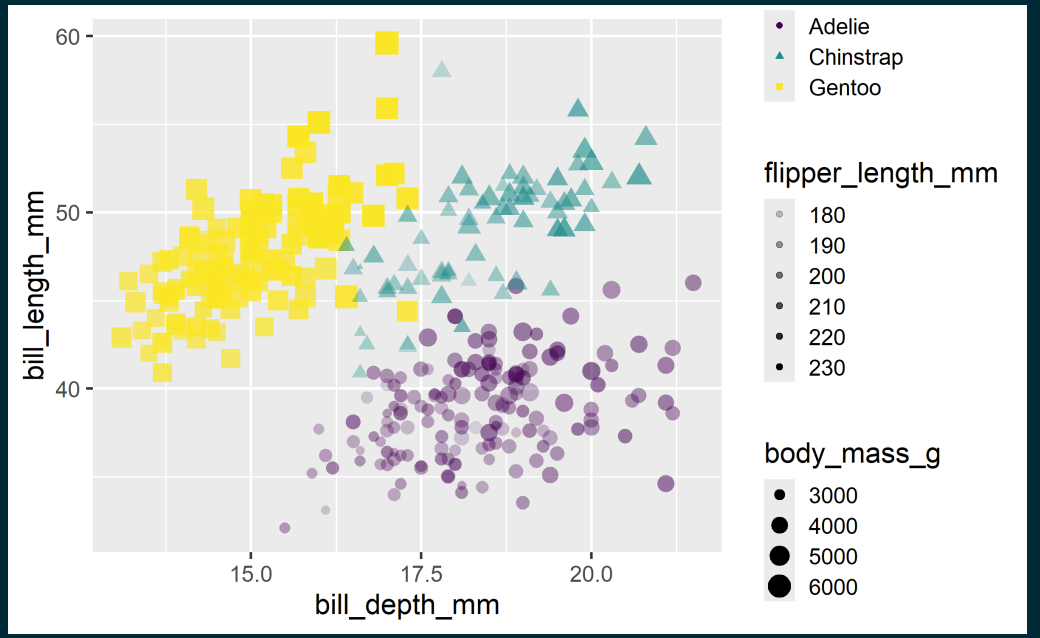
Size

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species,
           shape = species,
           size = body_mass_g)) +
  geom_point() +
  scale_colour_viridis_d()
```



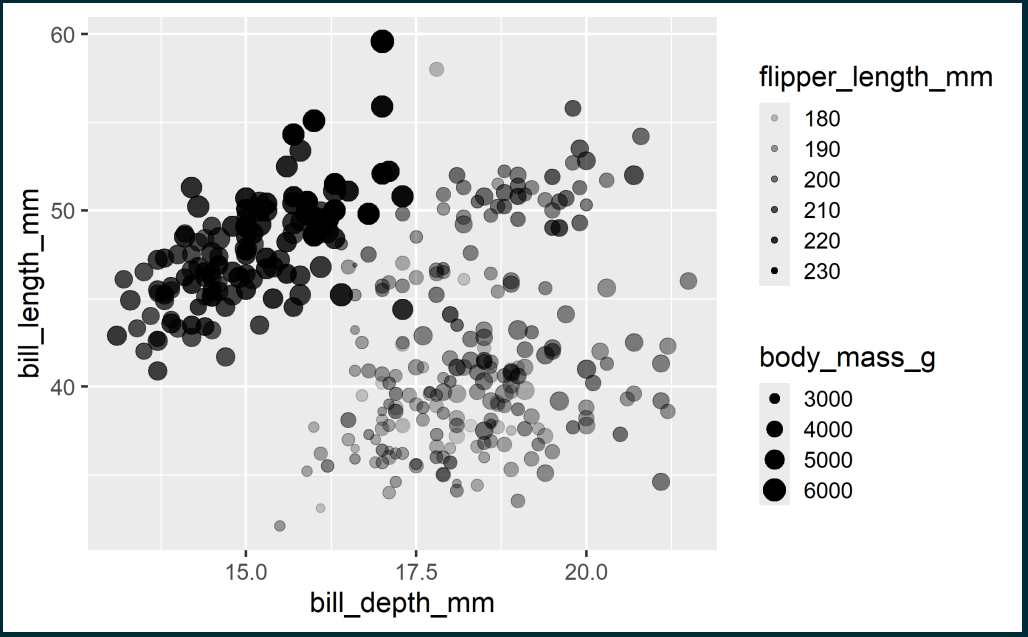
Alpha

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           colour = species,
           shape = species,
           size = body_mass_g,
           alpha = flipper_length_mm)) +
  geom_point() +
  scale_colour_viridis_d()
```



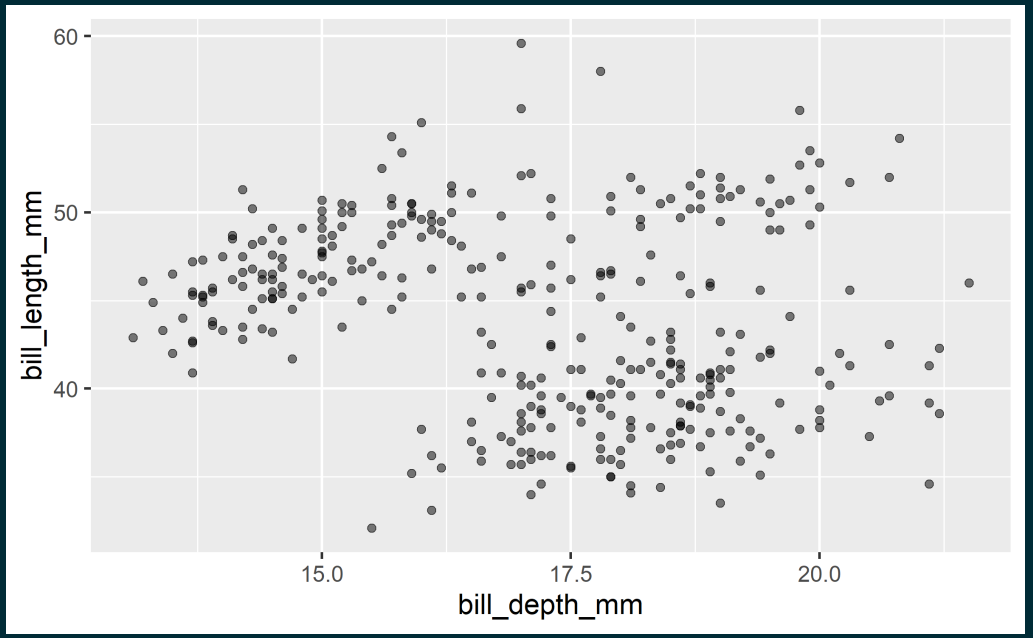
Mapping

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm,
           size = body_mass_g,
           alpha = flipper_length_mm)) +
  geom_point()
```



Setting

```
ggplot(penguins,
       aes(x = bill_depth_mm,
           y = bill_length_mm)) +
  geom_point(size = 2, alpha = 0.5)
```



Mapping vs. setting

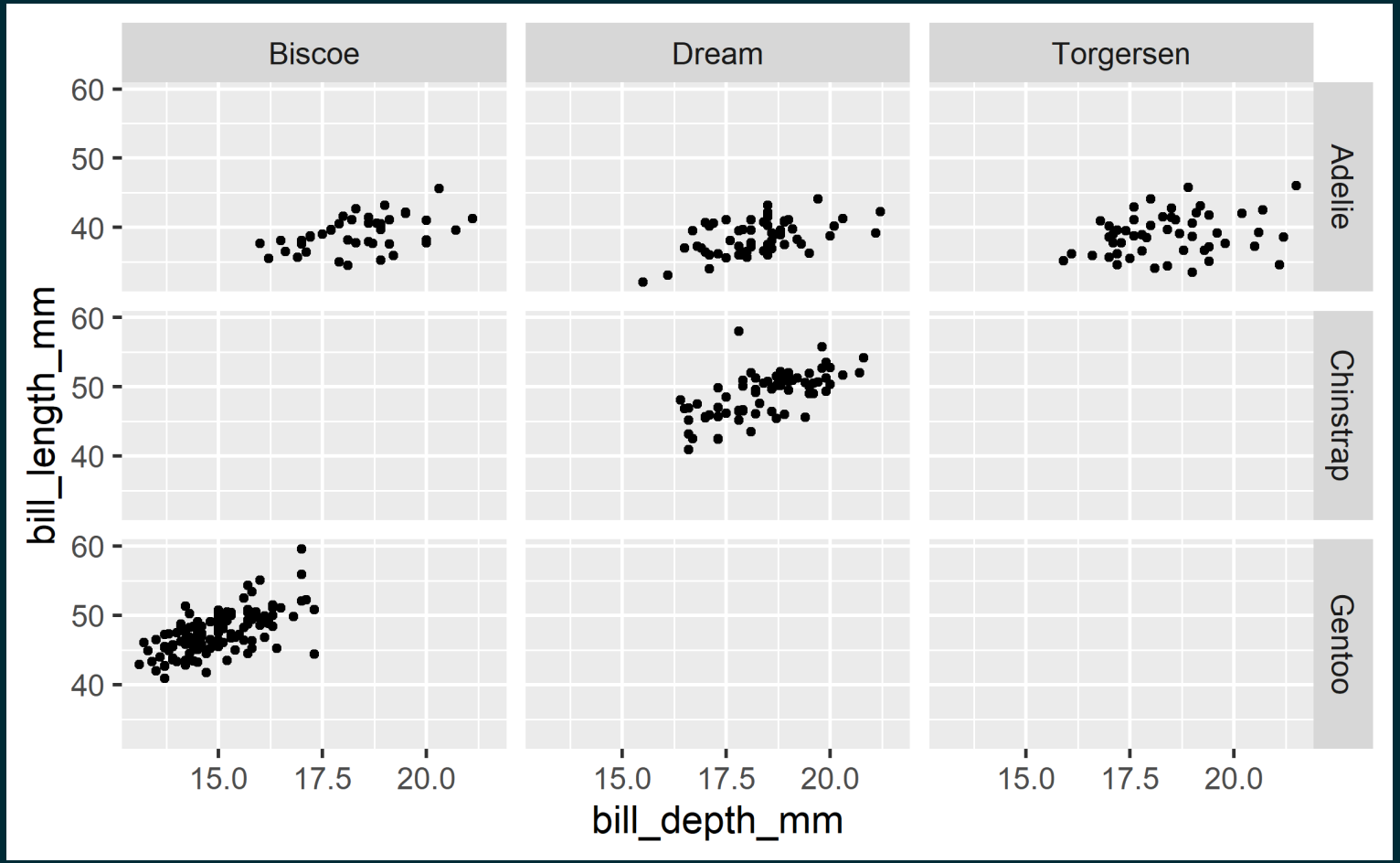
- **Mapping:** Determine the size, alpha, etc. of points based on the values of a variable in the data
    - goes into `aes()`
- **Setting:** Determine the size, alpha, etc. of points **not** based on the values of a variable in the data
    - goes into `geom_*()` (this was `geom_point()` in the previous example, but we'll learn about other geoms soon!)

Faceting

Faceting

- Smaller plots that display different subsets of the data
- Useful for exploring conditional relationships and large data

Plot Code

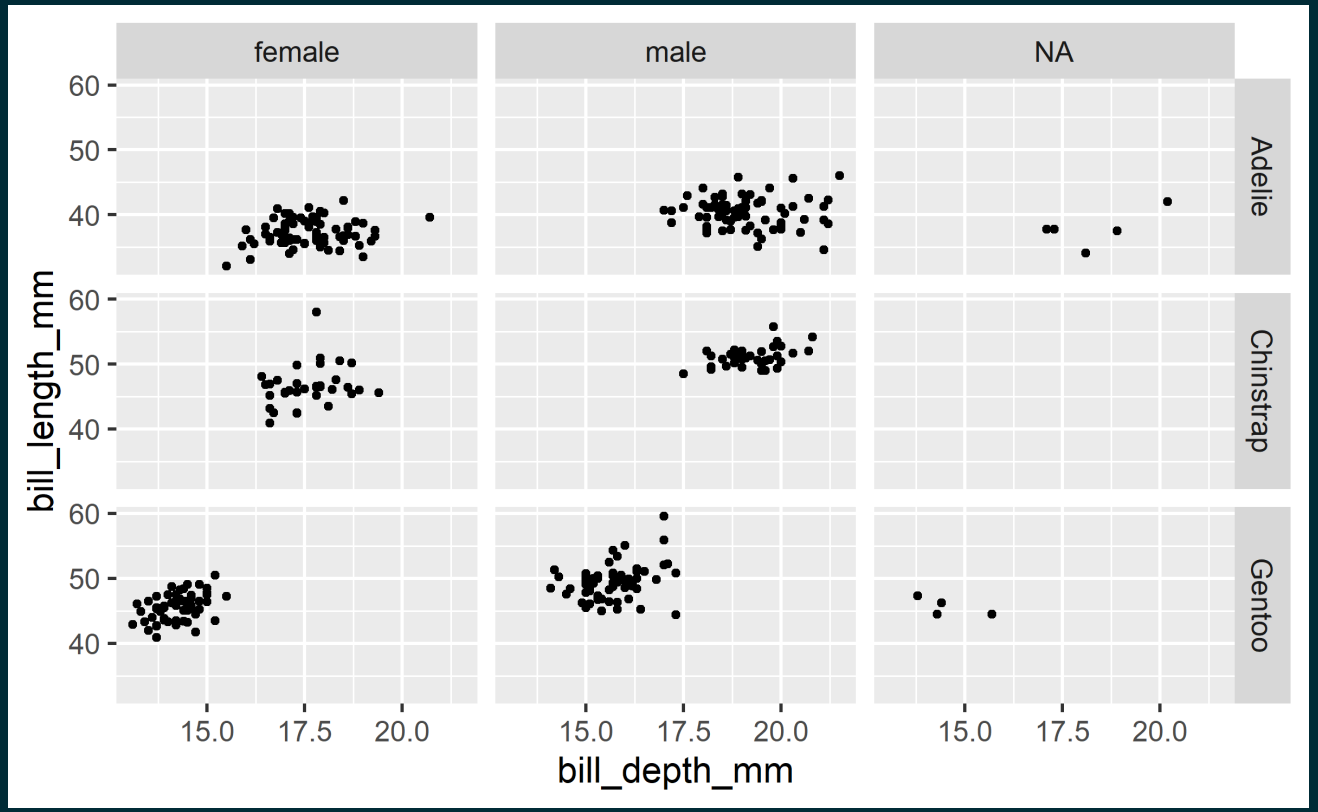

Biscoe
Dream
Torgersen
Adelie
Chinstrap
Gentoo
bill_length_mm
bill_depth_mm
60
50
40
15.0
17.5
20.0

Various ways to facet

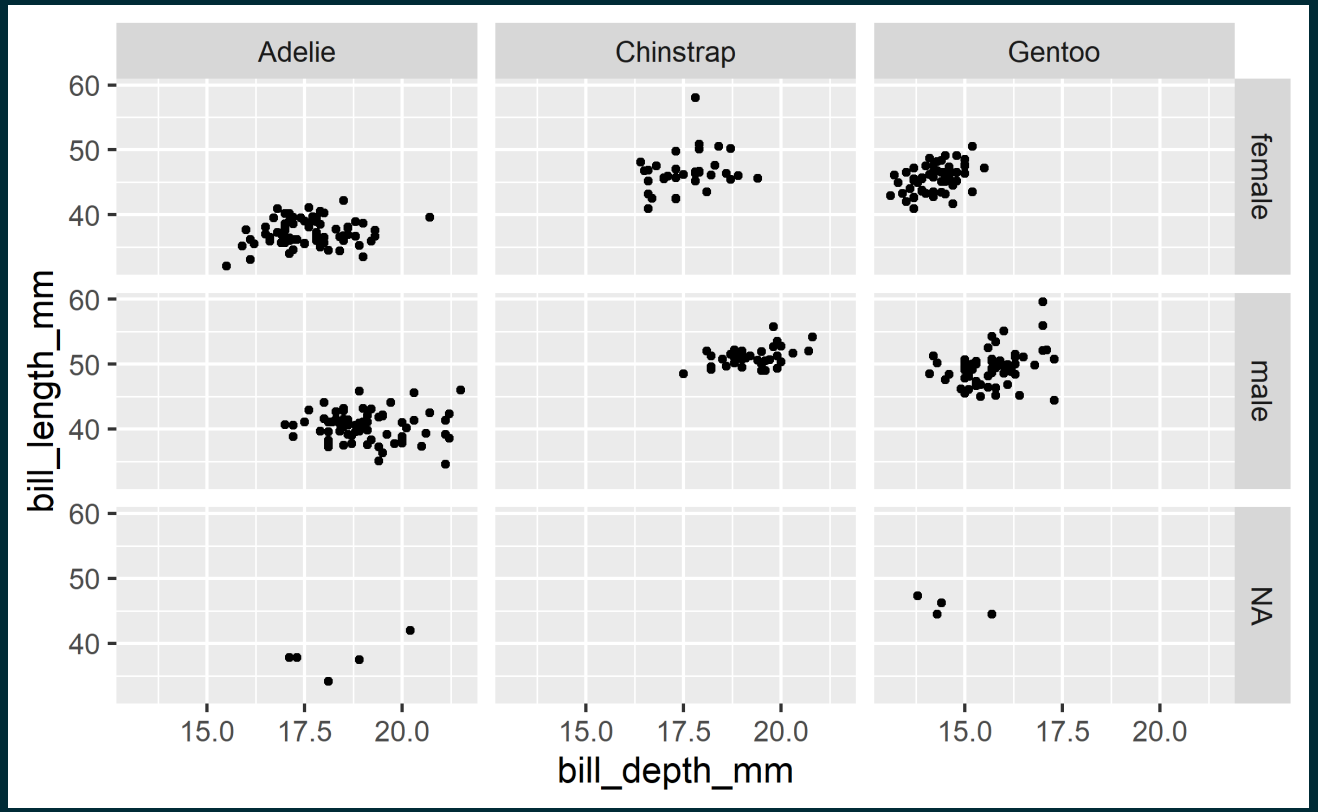
In the next few slides describe what each plot displays. Think about how the code relates to the output.

Note: The plots in the next few slides do not have proper titles, axis labels, etc. because we want you to figure out what's happening in the plots. But you should always label your plots!

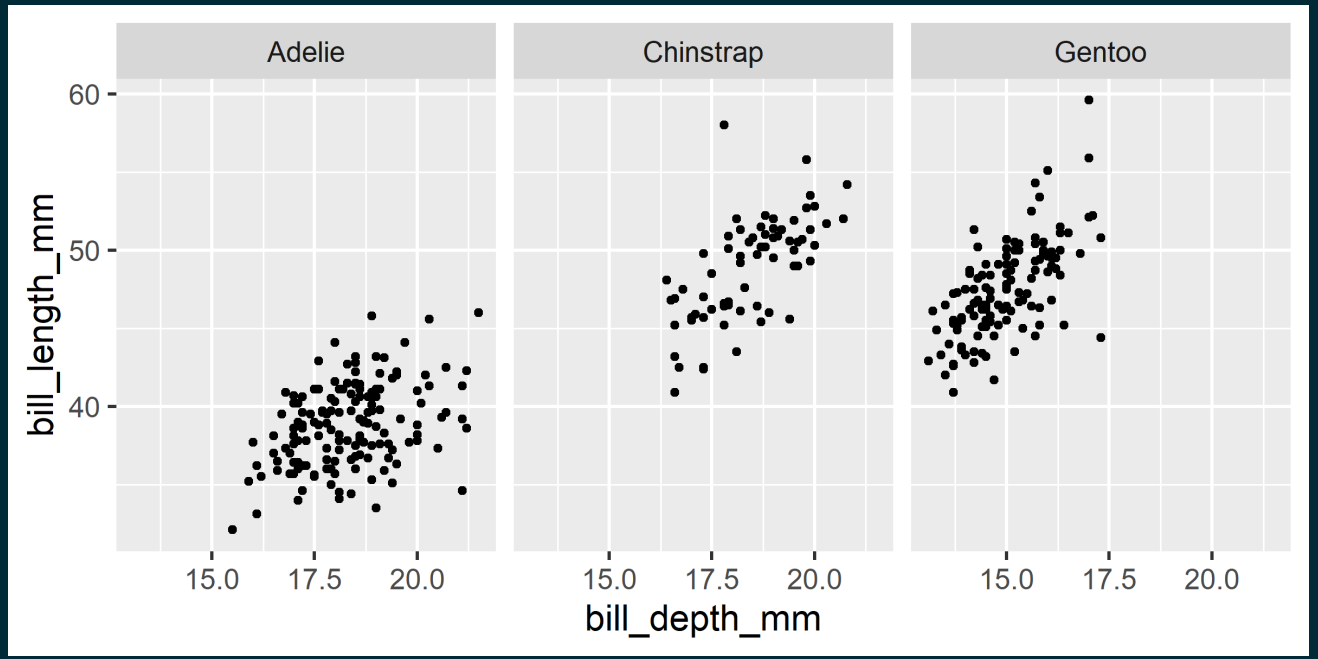
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_grid(species ~ sex)
```



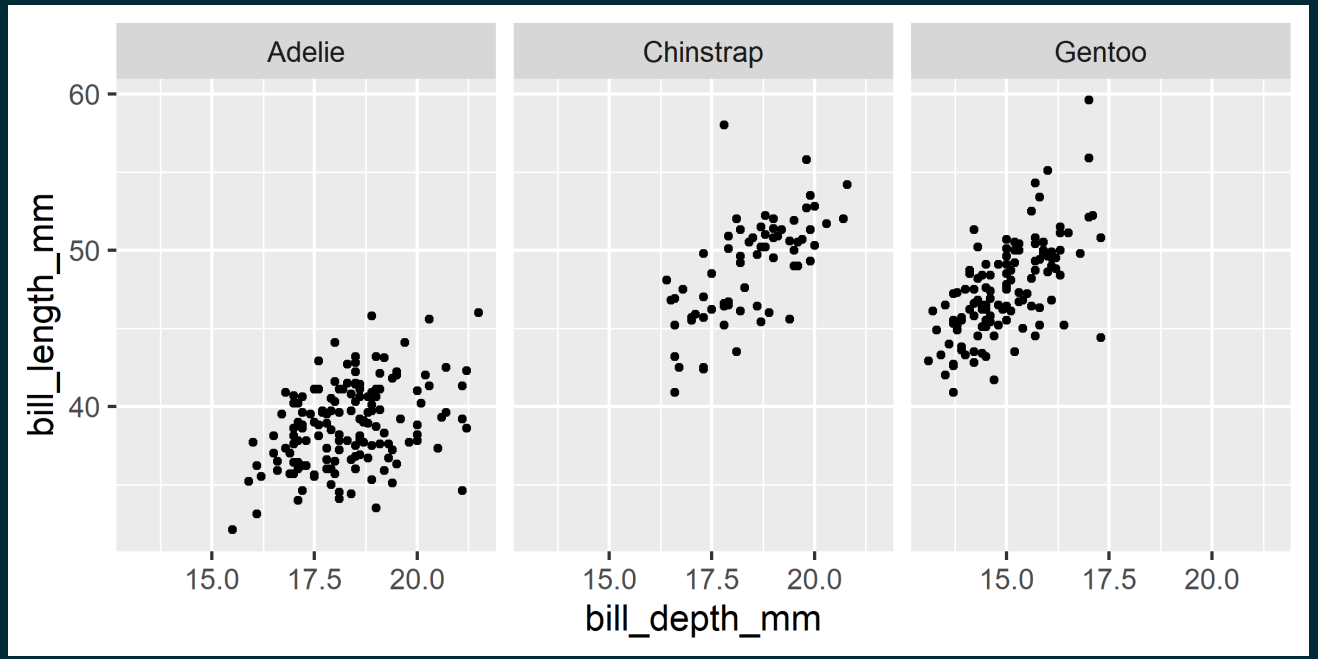
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_grid(sex ~ species)
```



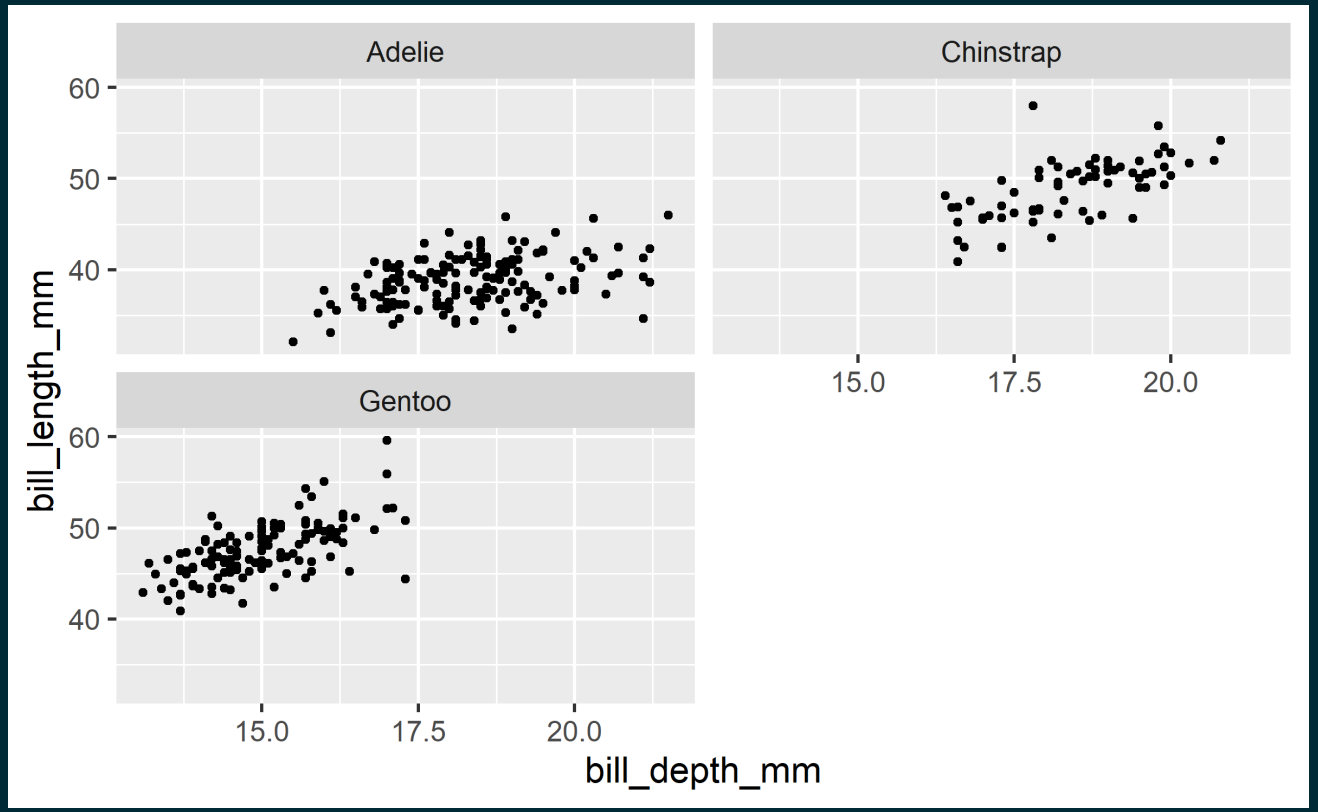
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_wrap(~ species)
```



```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_grid(. ~ species)
```



```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_wrap(~ species, ncol = 2)
```

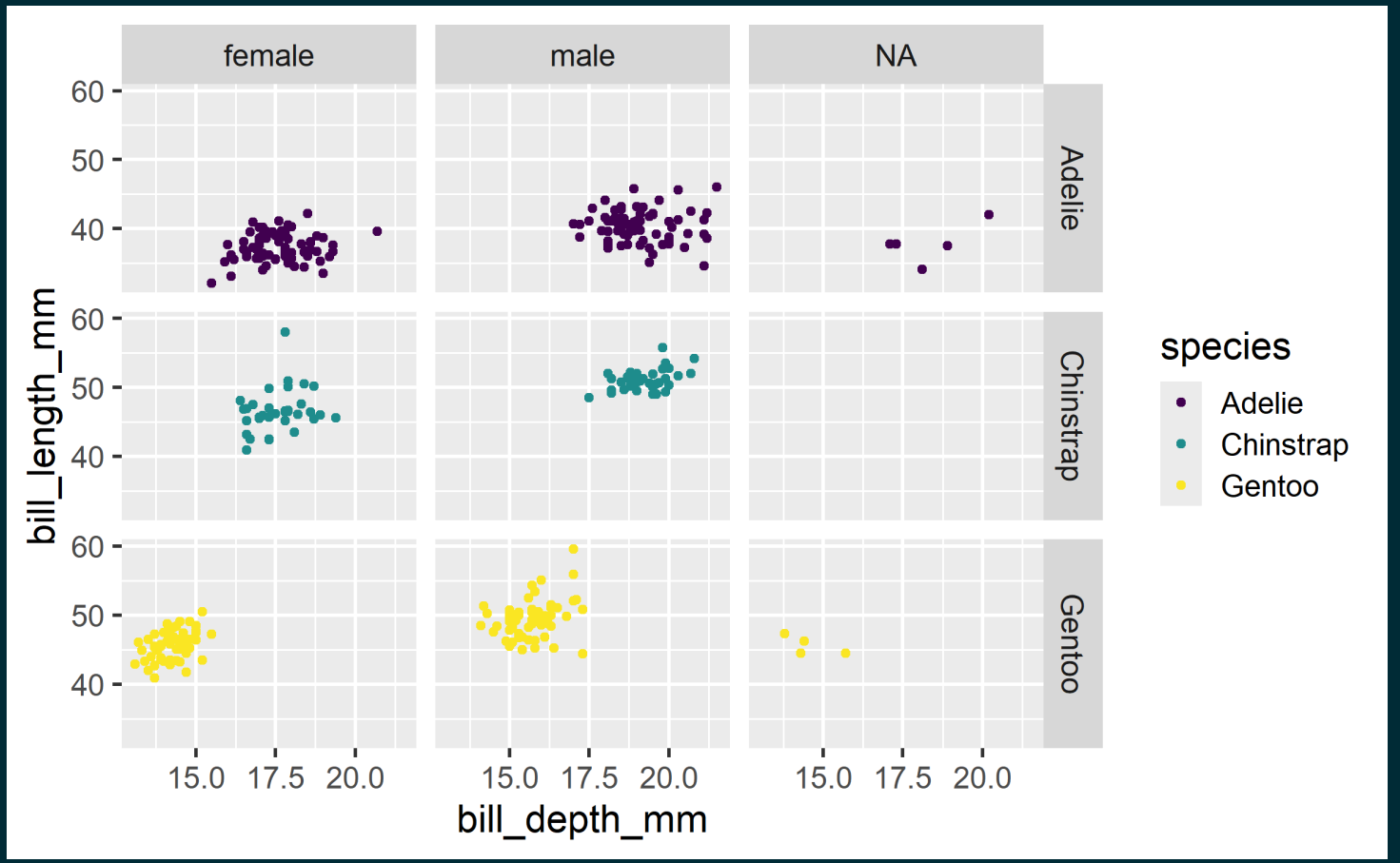


Faceting summary

- `facet_grid()`:
  - 2d grid
  - `rows ~ cols`
  - use `.` for no split
- `facet_wrap()`: 1d ribbon wrapped according to number of rows and columns specified or available plotting area

Facet and color

```
ggplot(
  penguins,
  aes(x = bill_depth_mm,
      y = bill_length_mm,
      color = species)) +
  geom_point() +
  facet_grid(species ~ sex) +
  scale_color_viridis_d()
```



Face and color, no legend

```
ggplot(
  penguins,
  aes(x = bill_depth_mm,
      y = bill_length_mm,
      color = species)) +
  geom_point() +
  facet_grid(species ~ sex) +
  scale_color_viridis_d() +
  guides(color = "none")
```