NRI 7350

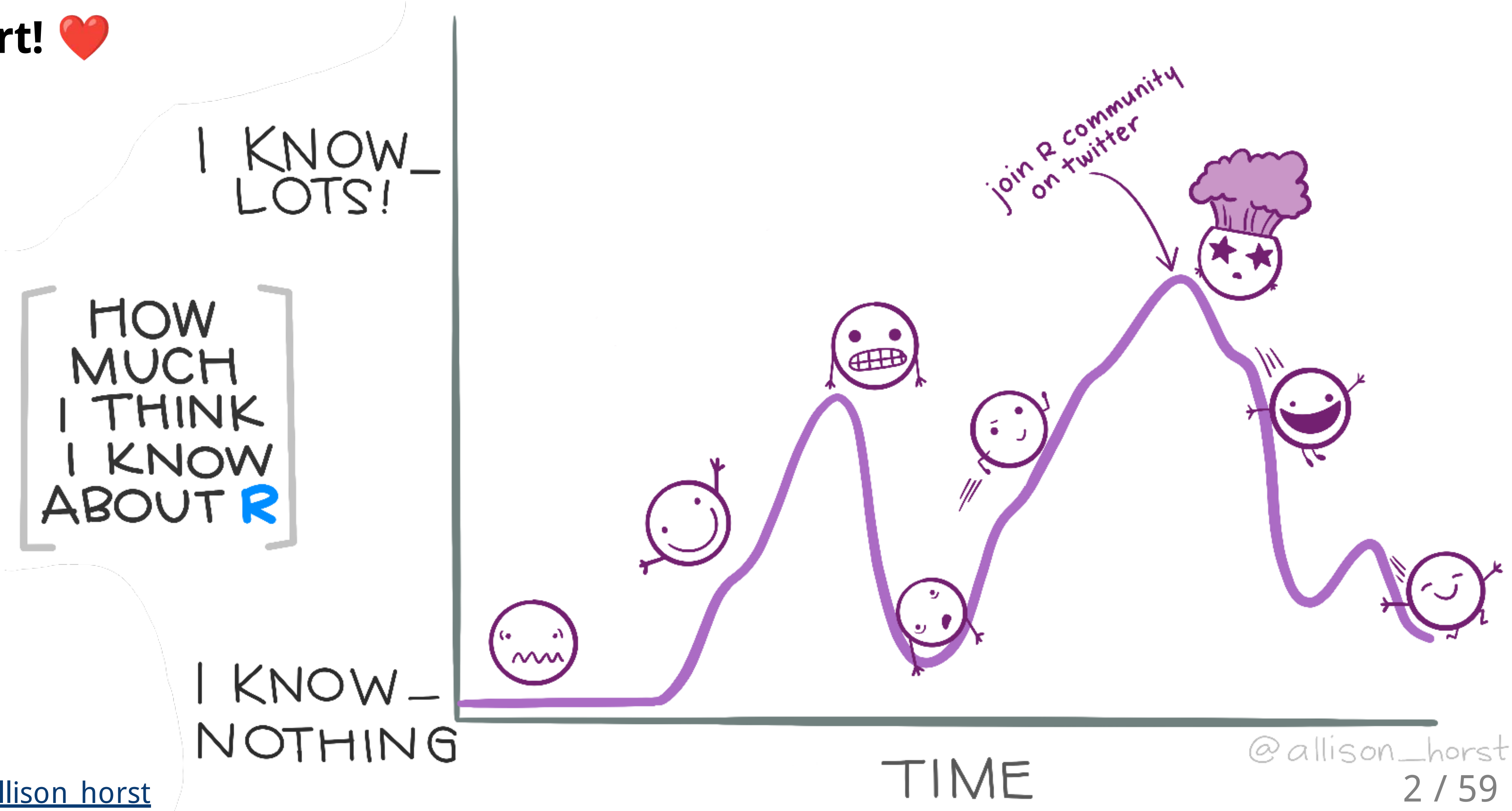
Data Exploration

Also `GGally`, `skimr`, `dplyr`, and `moments`

Artwork by [@allison_horst]

How Are we Doing?

Take Heart! ❤️



Artwork by [@allison_horst]

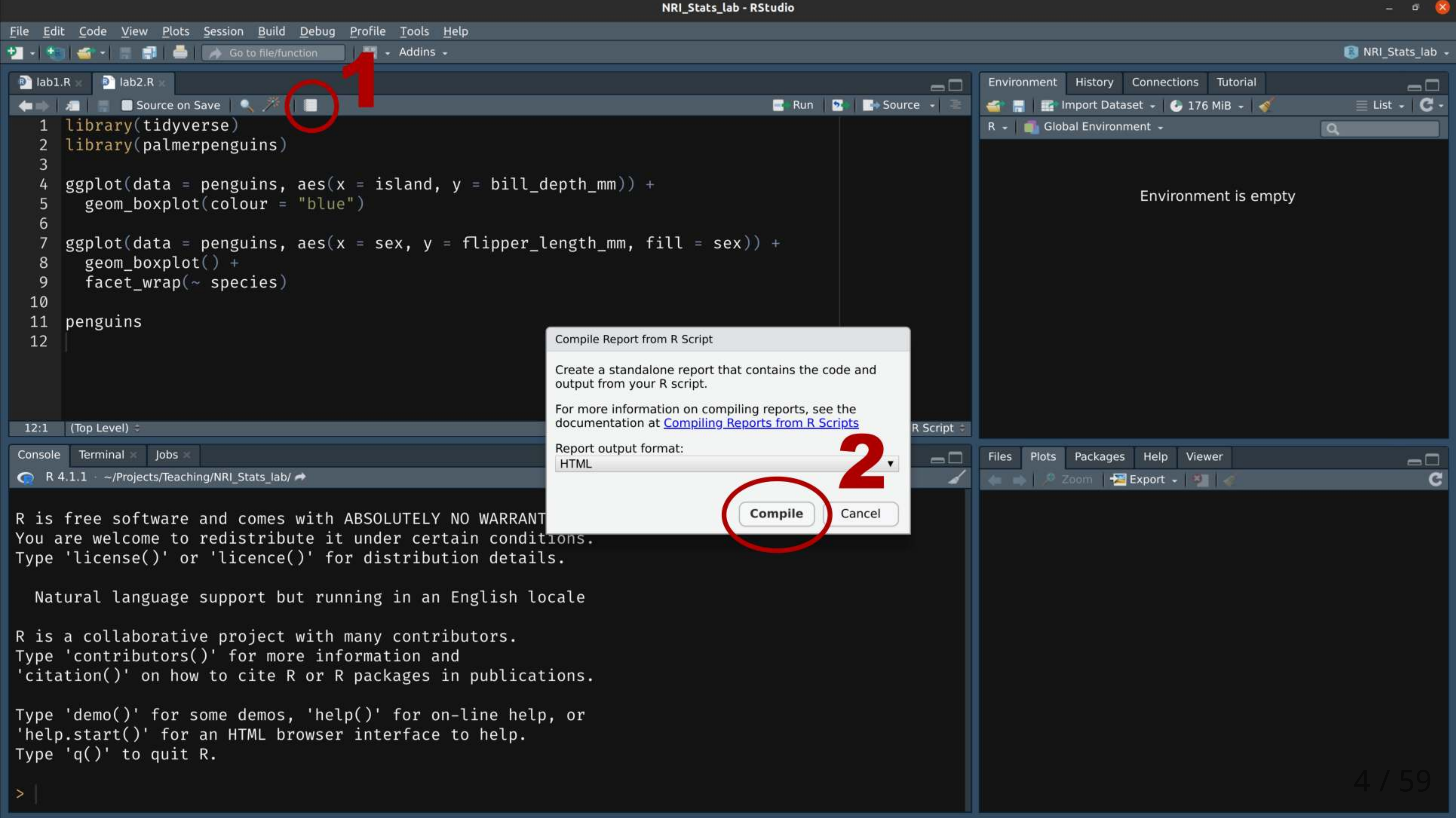
Learning to Compile Reports!

Open RStudio

Open your NRI project

Open a **old** script (if it's not already open)

'Files' pane (lower right) > Click on the script name



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

ggplot(data = penguins, aes(x = island, y = bill_depth_mm)) +
  geom_boxplot(colour = "blue")

ggplot(data = penguins, aes(x = sex, y = flipper_length_mm, fill = sex)) +
  geom_boxplot() +
  facet_wrap(~ species)

penguins
```

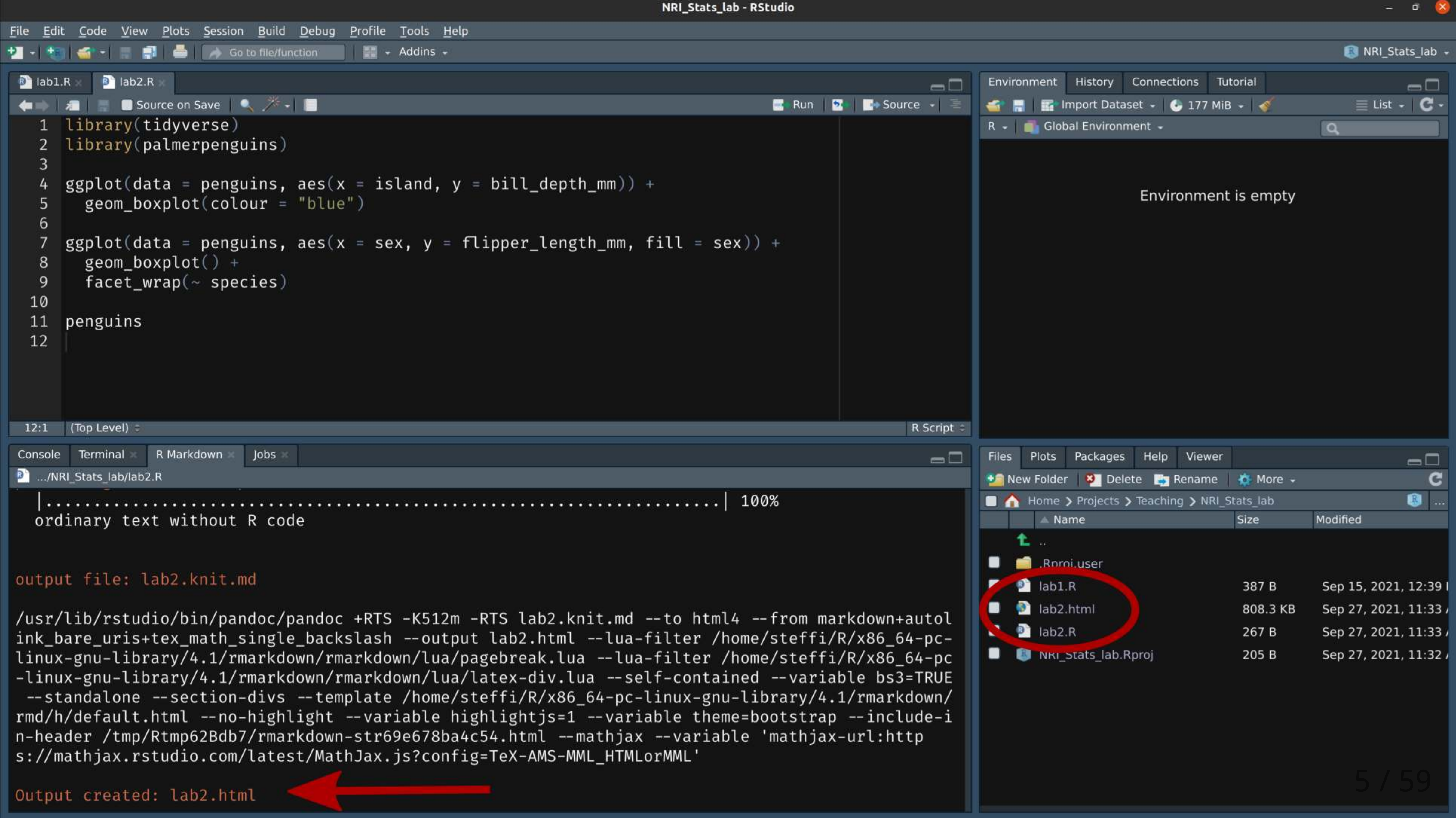
Compile Report from R Script

Create a standalone report that contains the code and output from your R script.

For more information on compiling reports, see the documentation at Compiling Reports from R Scripts

Report output format: HTML

```
R is free software and comes with ABSOLUTELY NO WARRANT
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```



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

ggplot(data = penguins, aes(x = island, y = bill_depth_mm)) +
  geom_boxplot(colour = "blue")

ggplot(data = penguins, aes(x = sex, y = flipper_length_mm, fill = sex)) +
  geom_boxplot() +
  facet_wrap(~ species)

penguins

```

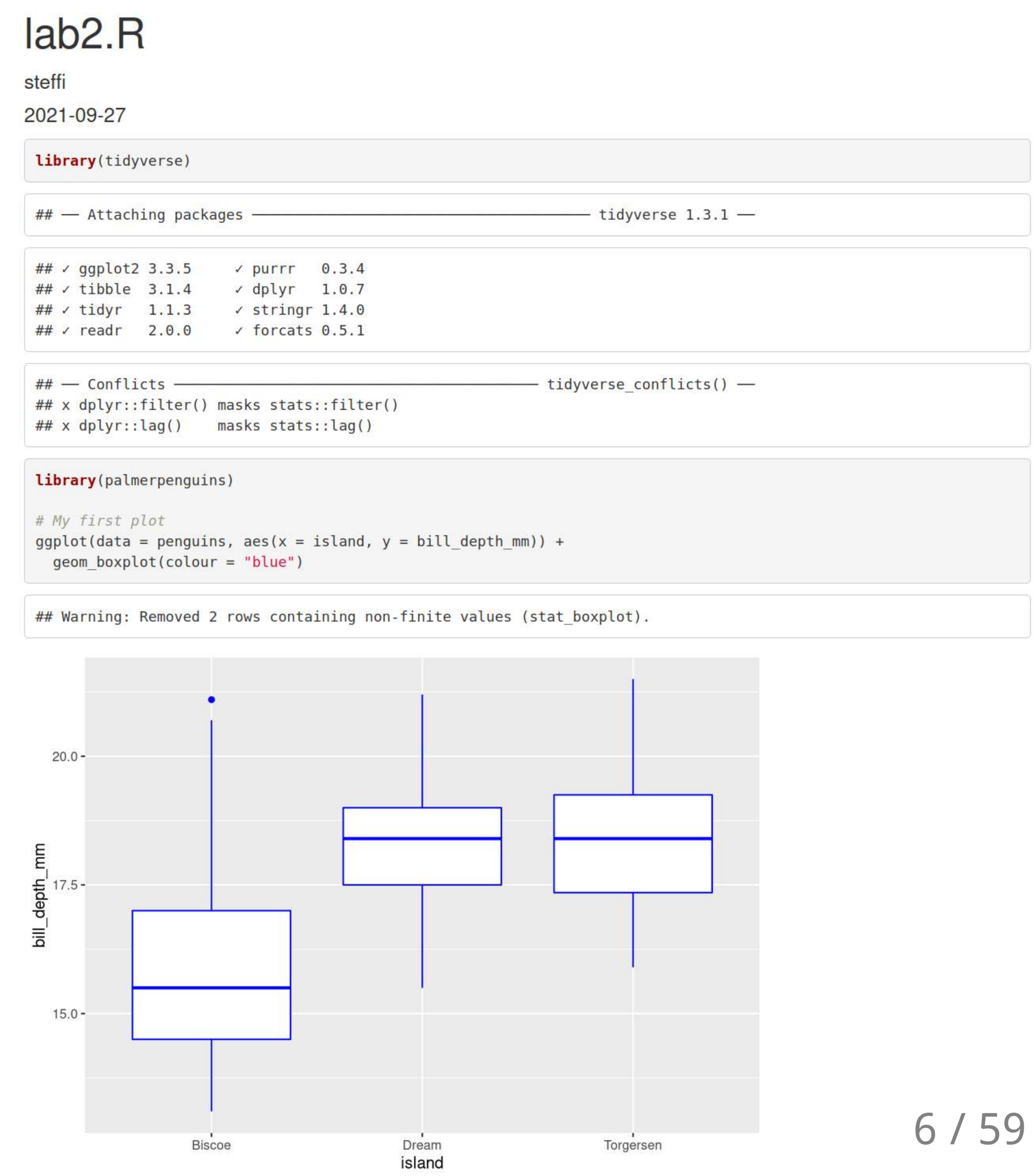
```
  |.......................................................................| 100%
  ordinary text without R code


output file: lab2.knit.md

/usr/lib/rstudio/bin/pandoc/pandoc +RTS -K512m -RTS lab2.knit.md --to html4 --from markdown+autol
ink_bare_uris+tex_math_single_backslash --output lab2.html --lua-filter /home/steffi/R/x86_64-pc-
linux-gnu-library/4.1/rmarkdown/rmarkdown/lua/pagebreak.lua --lua-filter /home/steffi/R/x86_64-pc
-linux-gnu-library/4.1/rmarkdown/rmarkdown/lua/latex-div.lua --self-contained --variable bs3=TRUE
 --standalone --section-divs --template /home/steffi/R/x86_64-pc-linux-gnu-library/4.1/rmarkdown/
rmd/h/default.html --no-highlight --variable highlightjs=1 --variable theme=bootstrap --include-i
n-header /tmp/Rtmp62Bdb7/rmarkdown-str69e678ba4c54.html --mathjax --variable 'mathjax-url:http
s://mathjax.rstudio.com/latest/MathJax.js?config=TeX-AMS-MML_HTMLorMML'

Output created: lab2.html
```

Name	Size	Modified
..		
.Rproj.user		
lab1.R	387 B	Sep 15, 2021, 12:39
lab2.html	808.3 KB	Sep 27, 2021, 11:33
lab2.R	267 B	Sep 27, 2021, 11:33
NRI_Stats_lab.Rproj	205 B	Sep 27, 2021, 11:32

Compile script into report

This is **all I need**:

- Your assignment answers
- This html report



lab2.R

steffi

2021-09-27

```r
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──
```

```
## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.4     ✓ dplyr   1.0.7
## ✓ tidyr   1.1.3     ✓ stringr 1.4.0
## ✓ readr   2.0.0     ✓ forcats 0.5.1
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(palmerpenguins)

# My first plot
ggplot(data = penguins, aes(x = island, y = bill_depth_mm)) +
  geom_boxplot(colour = "blue")
```

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

However!

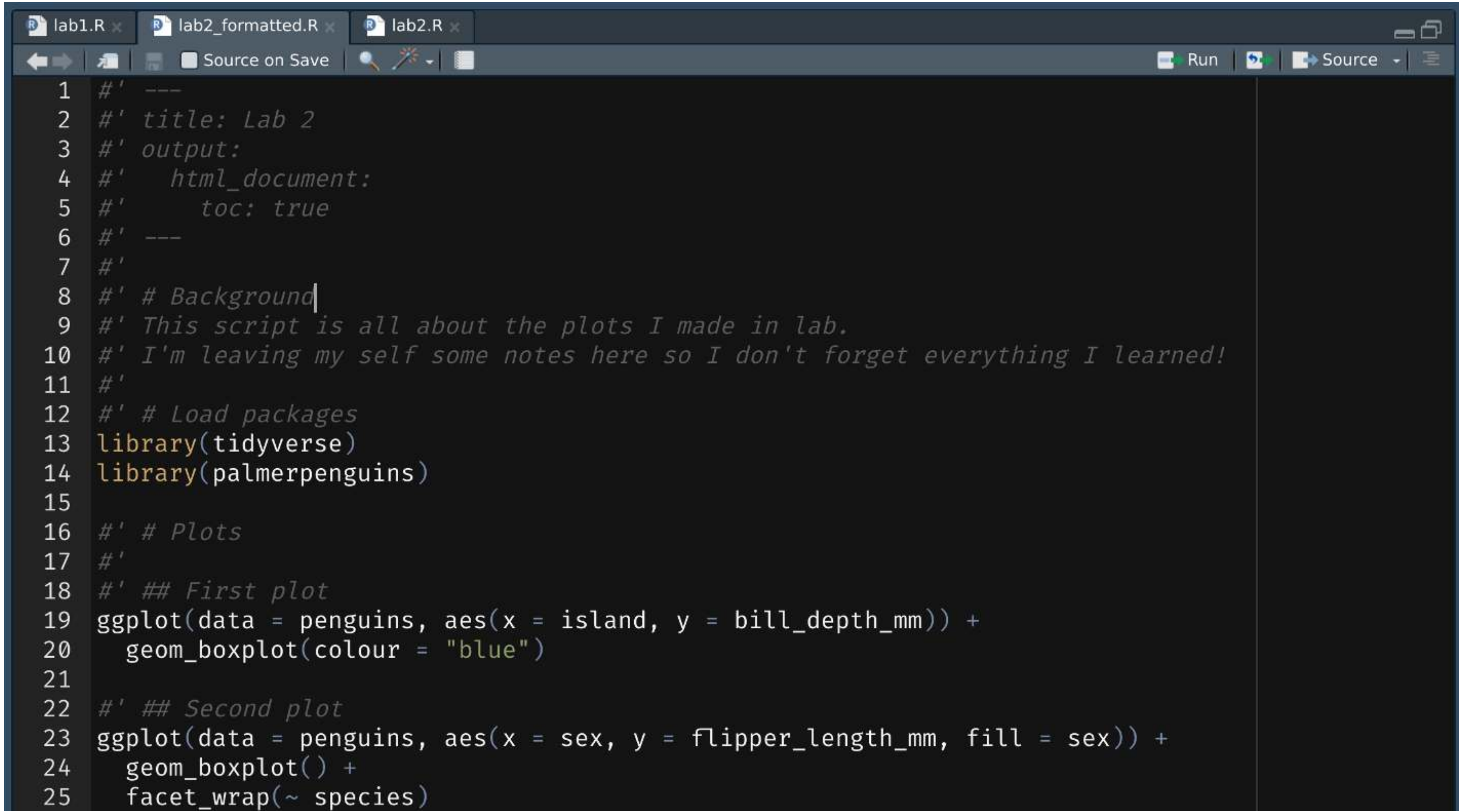
If you want to take it further:

- You can use Markdown formating to make things look nice
- You just need to put `#'` in front of any Markdown notation or text (DEMO)

Markdown

- `**bold**`
- `*italic*`
- `#`, `##`, `###` for headings (first-, second-, third-level)

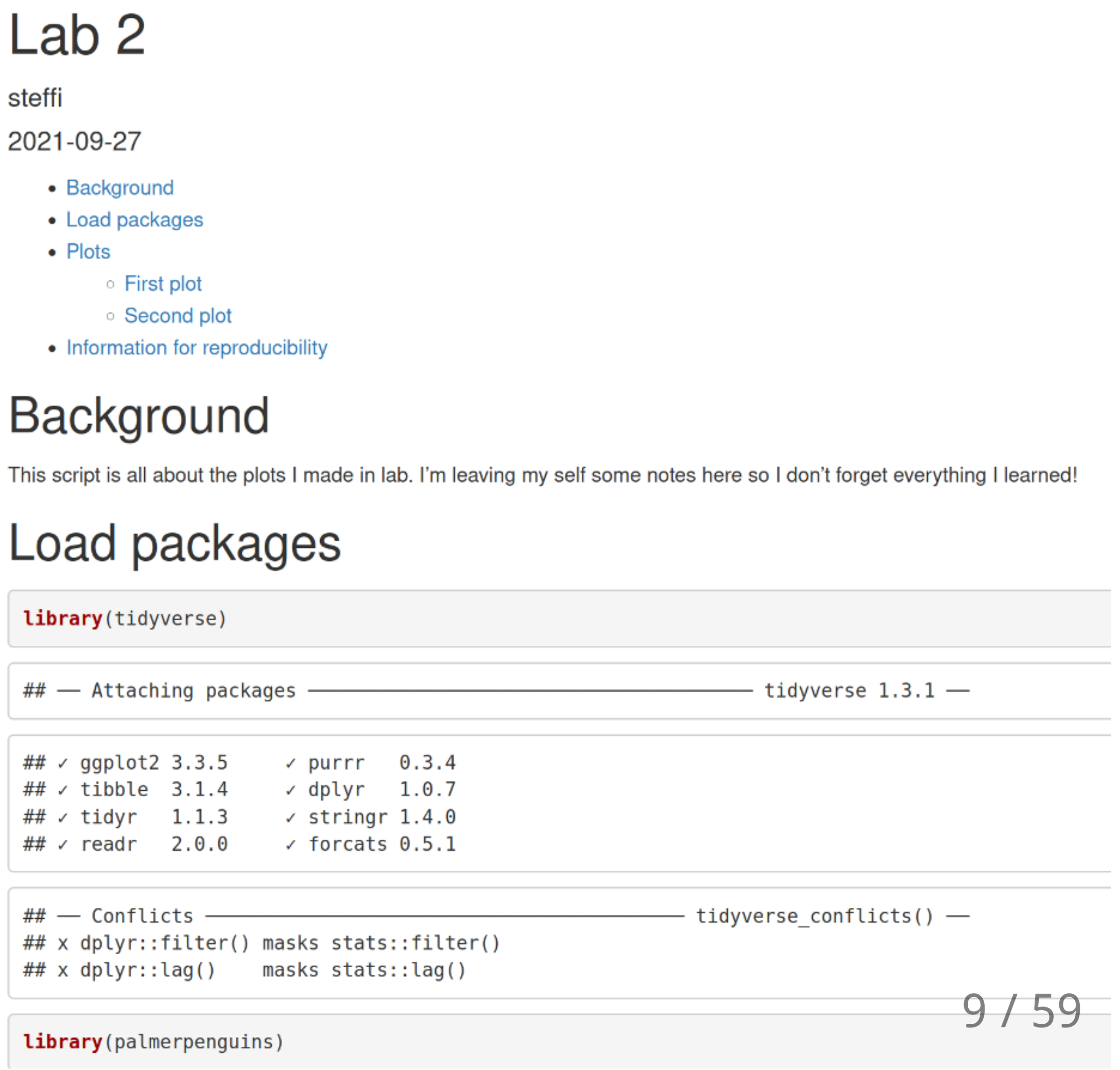
> **Familiar with Rmd?**
>
> Think of this as the mirror image:
>
> Instead of marking **what is** R code, you mark **what is NOT** R code



```r
#' ---
#' title: Lab 2
#' output:
#'   html_document:
#'     toc: true
#' ---
#'
#' # Background
#' This script is all about the plots I made in lab.
#' I'm leaving my self some notes here so I don't forget everything I learned!
#'
#' # Load packages
library(tidyverse)
library(palmerpenguins)

#' # Plots
#'
#' ## First plot
ggplot(data = penguins, aes(x = island, y = bill_depth_mm)) +
  geom_boxplot(colour = "blue")

#' ## Second plot
ggplot(data = penguins, aes(x = sex, y = flipper_length_mm, fill = sex)) +
  geom_boxplot() +
  facet_wrap(~ species)
```

Further reading

- R Markdown Cookbook
    - Chapter 3.3 Render an R script to a report
- R for Data Science
    - Chapter 27 Rmarkdown



Lab 2

steffi

2021-09-27

- Background
- Load packages
- Plots
    - First plot
    - Second plot
- Information for reproducibility

Background

This script is all about the plots I made in lab. I'm leaving my self some notes here so I don't forget everything I learned!

Load packages

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──
```

```
## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.4     ✓ dplyr   1.0.7
## ✓ tidyr   1.1.3     ✓ stringr 1.4.0
## ✓ readr   2.0.0     ✓ forcats 0.5.1
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(palmerpenguins)
```

Getting started (again)

Hoping you can work with some of your own data today!

Open RStudio
Open your NRI project
Open your data-loading script:
'Files' Pane > Click on script name
RUN YOUR SCRIPT

Make sure to load packages at the top:

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

(if working with penguins today)

Exploring everything at once

Visualize with `ggpairs()`

- From **`GGally`** package
  - **Caution!** If you have a lot of columns, **`select()`** only a few to work with

```
library(GGally)

penguins_sub <- select(penguins, -sex, -island, -year)
ggpairs(penguins_sub)
```

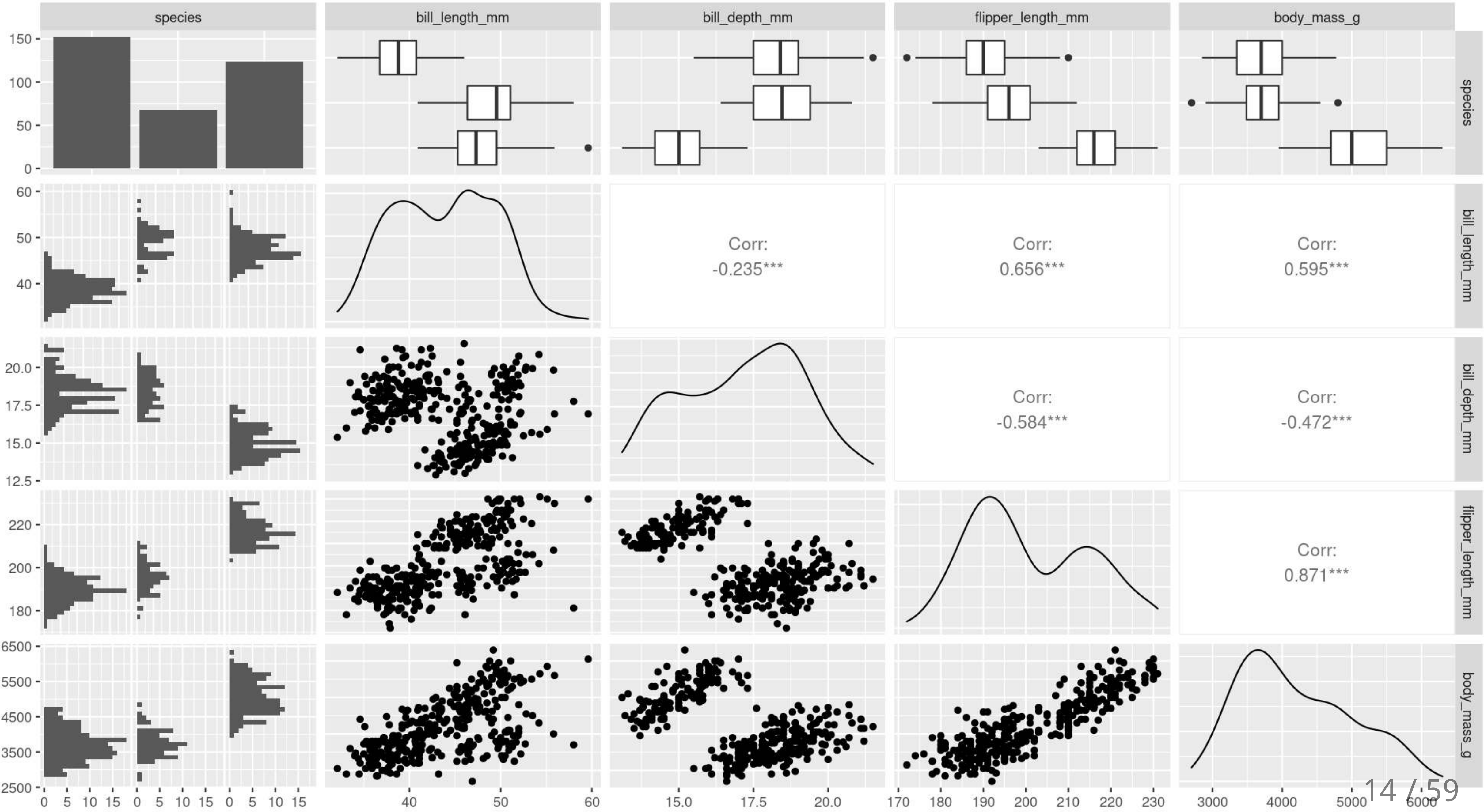
Side Note: `tidyverse` functions

- From **`GGally`**
  - **Caution!** If you have a lot of columns, **`select()`** only a few to work with

```
library(GGally)

penguins_sub <- select(penguins, -sex, -island, -year)
ggpairs(penguins_sub)
```

`select()`

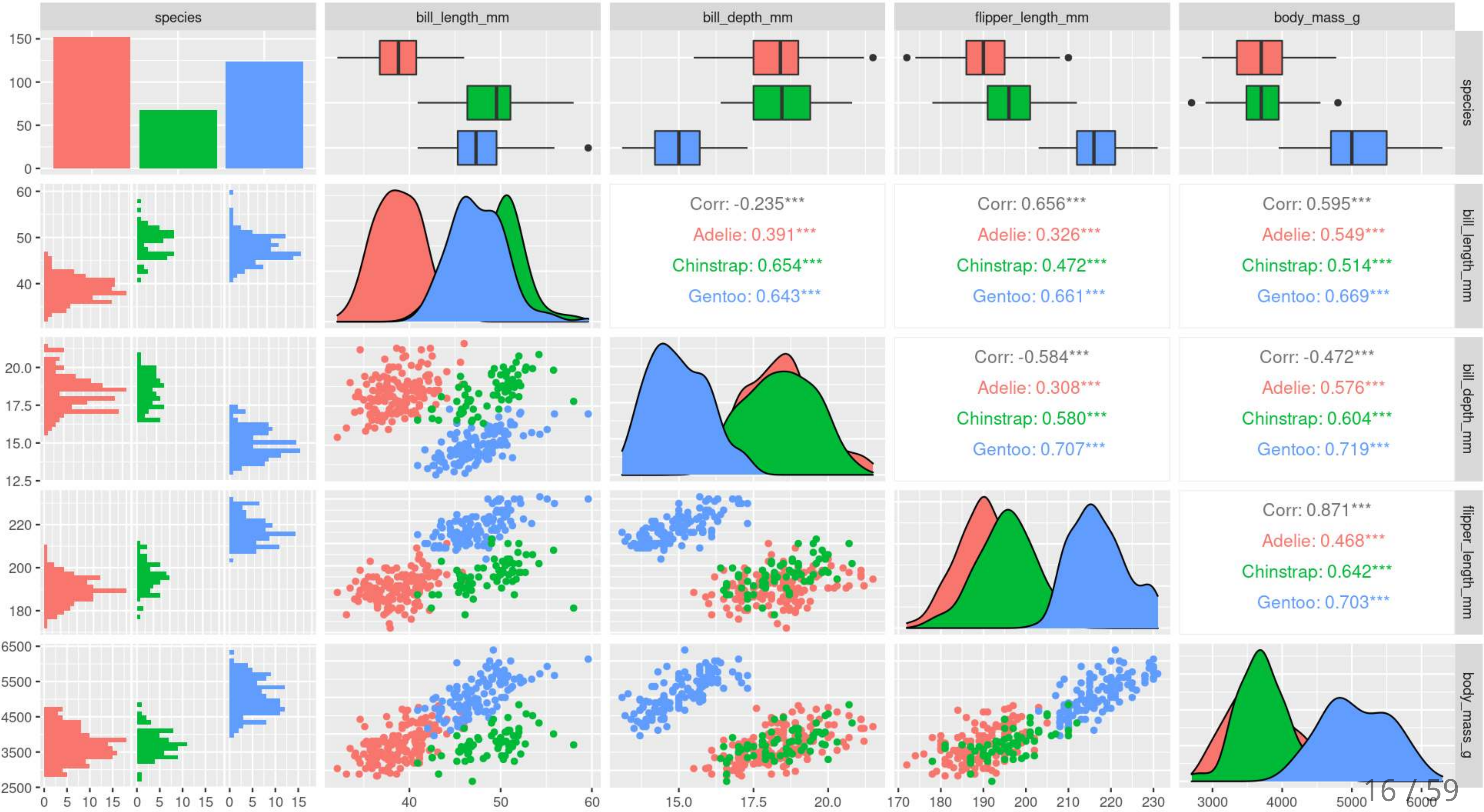
- **`tidyverse`** functions always start with the **data**, followed by other arguments
- you can reference any **column** from '**data**'
- **`select()`** chooses columns to keep or to remove (with **-**)


species
bill_length_mm
bill_depth_mm
flipper_length_mm
body_mass_g
150
100
50
0
60
50
40
20.0
17.5
15.0
12.5
220
200
180
6500
5500
4500
3500
2500
Corr:
-0.235***
Corr:
0.656***
Corr:
0.595***
Corr:
-0.584***
Corr:
-0.472***
Corr:
0.871***
0 5 10 15
0 5 10 15
0 5 10 15
40
50
60
15.0
17.5
20.0
170
180
190
200
210
220
230
3000
4000
5000
6000
species
bill_length_mm
bill_depth_mm
flipper_length_mm
body_mass_g

Visualize with **`ggpairs()`**

```
library(GGally)

ggpairs(select(penguins, -sex, -island, -year), aes(colour = species))
```

> **`ggpairs()`** builds on **`ggplot()`** so we can use an **`aes()`** specification


species
bill_length_mm
bill_depth_mm
flipper_length_mm
body_mass_g
Corr: -0.235***
Adelie: 0.391***
Chinstrap: 0.654***
Gentoo: 0.643***
Corr: 0.656***
Adelie: 0.326***
Chinstrap: 0.472***
Gentoo: 0.661***
Corr: 0.595***
Adelie: 0.549***
Chinstrap: 0.514***
Gentoo: 0.669***
Corr: -0.584***
Adelie: 0.308***
Chinstrap: 0.580***
Gentoo: 0.707***
Corr: -0.472***
Adelie: 0.576***
Chinstrap: 0.604***
Gentoo: 0.719***
Corr: 0.871***
Adelie: 0.468***
Chinstrap: 0.642***
Gentoo: 0.703***
150
100
50
0
60
50
40
20.0
17.5
15.0
12.5
220
200
180
6500
5500
4500
3500
2500
0 5 10 15 0 5 10 15 0 5 10 15
40 50 60
15.0 17.5 20.0
170 180 190 200 210 220 230
3000 4000 5000 6000

Visualize with `ggpairs()`

```
library(GGally)
penguins_sub <- select(penguins, -sex, -island, -year)

ggpairs(penguins_sub)

ggpairs(penguins_sub, aes(colour = species))
```

Your Turn!

Summarize with **`skim()`**

`skim()` from `skimr` package

```r
library(skimr)
skim(penguins)
```

```
## ── Data Summary ────────────────────────
##                            Values
## Name                       penguins
## Number of rows             344
## Number of columns          8
## _______________________
## Column type frequency:
##   factor                   3
##   numeric                  5
## ________________________
## Group variables            None
##
## ── Variable type: factor ────────────────────────────────────────────────────────────────────────
##   skim_variable n_missing complete_rate ordered n_unique top_counts
## 1 species               0         1     FALSE          3 Ade: 152, Gen: 124, Chi: 68
## 2 island                0         1     FALSE          3 Bis: 168, Dre: 124, Tor: 52
## 3 sex                  11         0.968 FALSE          2 mal: 168, fem: 165
##
```

Summarize with **`skim()`**

`skim()` from `skimr` package

```r
library(skimr)
skim(penguins)
```

```
##
## ── Variable type: factor ────────────────────────────────────────────────────────────────────────
##   skim_variable n_missing complete_rate ordered n_unique top_counts
## 1 species               0         1     FALSE          3 Ade: 152, Gen: 124, Chi: 68
## 2 island                0         1     FALSE          3 Bis: 168, Dre: 124, Tor: 52
## 3 sex                  11         0.968 FALSE          2 mal: 168, fem: 165
##
## ── Variable type: numeric ───────────────────────────────────────────────────────────────────────
##   skim_variable     n_missing complete_rate    mean      sd     p0    p25    p50    p75   p100 hist
## 1 bill_length_mm            2         0.994   43.9    5.46    32.1   39.2   44.4   48.5   59.6 ▃▇▇▆▁
## 2 bill_depth_mm             2         0.994   17.2    1.97    13.1   15.6   17.3   18.7   21.5 ▅▅▇▇▂
## 3 flipper_length_mm         2         0.994  201.    14.1    172    190    197    213    231   ▂▇▃▅▂
## 4 body_mass_g               2         0.994 4202.   802.    2700   3550   4050   4750   6300   ▃▇▆▃▂
## 5 year                      0         1     2008.     0.818 2007   2007   2008   2009   2009   ▇▁▇▁▇
```

Summarize with **skim()**

skim() from **skimr** package

Your Turn!

```
library(skimr)
skim(penguins)
```

```
##
## ── Variable type: factor ────────────────────────────────────────────────────────────────────────
##   skim_variable n_missing complete_rate ordered n_unique top_counts
## 1 species               0         1     FALSE          3 Ade: 152, Gen: 124, Chi: 68
## 2 island                0         1     FALSE          3 Bis: 168, Dre: 124, Tor: 52
## 3 sex                  11         0.968 FALSE          2 mal: 168, fem: 165
##
## ── Variable type: numeric ───────────────────────────────────────────────────────────────────────
##   skim_variable     n_missing complete_rate   mean      sd     p0   p25   p50   p75  p100 hist
## 1 bill_length_mm            2         0.994   43.9   5.46    32.1  39.2  44.4  48.5  59.6 ▃▇▇▆▁
## 2 bill_depth_mm             2         0.994   17.2   1.97    13.1  15.6  17.3  18.7  21.5 ▅▅▇▇▂
## 3 flipper_length_mm         2         0.994  201.   14.1    172   190   197   213   231   ▂▇▃▅▂
## 4 body_mass_g               2         0.994 4202.  802.    2700  3550  4050  4750  6300   ▃▇▆▃▂
## 5 year                      0         1     2008.    0.818 2007  2007  2008  2009  2009   ▇▁▇▁▇
```

Summarize with `skim()`

`group_by` from `dplyr` package

```
penguins_sp <- group_by(penguins, species)
skim(penguins_sp)
```

Side Note: `tidyverse` functions

`group_by` from `dplyr` package

```
penguins_sp <- group_by(penguins, species)
skim(penguins_sp)
```

`group_by()`

- **`tidyverse`** functions always start with the **data**, followed by other arguments
- you can reference any **column** from '**data**'
- **`group_by()`** assigns grouping to a data frame. Here, we group **`penguins`** by species

Extra:

In the console look at **`penguins`** (type in **`penguins`** and hit enter), and then look at **`penguins_sp`** (type in **`penguins_sp`** and it enter).

How does the output differ? (Hint very little! But there is one difference...)

Summarize with `skim()`

`group_by` from `dplyr` package

```
penguins_sp <- group_by(penguins, species)
skim(penguins_sp)
```

```
## ── Data Summary ───────────────────────
##                            Values
## Name                       skimp
## Number of rows             344
## Number of columns          8
## _______________________
## Column type frequency:
##   factor                   2
##   numeric                  5
## ________________________
## Group variables            species
##
## ── Variable type: factor ────────────────────────────────────────────────────────────────────────
##   skim_variable species   n_missing complete_rate ordered n_unique top_counts
## 1 island        Adelie            0         1     FALSE          3 Dre: 56, Tor: 52, Bis: 44
## 2 island        Chinstrap         0         1     FALSE          1 Dre: 68, Bis: 0, Tor: 0
## 3 island        Gentoo            0         1     FALSE          1 Bis: 124, Dre: 0, Tor: 0
## 4 sex           Adelie            6         0.961 FALSE          2 fem: 73, mal: 73
```

Summarize with `skim()`

`group_by` from `dplyr` package

```
penguins_sp <- group_by(penguins, species)
skim(penguins_sp)
```

```
##
## ── Variable type: factor ─────────────────────────────────────────────────────────────────────────
##   skim_variable species   n_missing complete_rate ordered n_unique top_counts
## 1 island        Adelie            0         1     FALSE          3 Dre: 56, Tor: 52, Bis: 44
## 2 island        Chinstrap         0         1     FALSE          1 Dre: 68, Bis: 0, Tor: 0
## 3 island        Gentoo            0         1     FALSE          1 Bis: 124, Dre: 0, Tor: 0
## 4 sex           Adelie            6         0.961 FALSE          2 fem: 73, mal: 73
## 5 sex           Chinstrap         0         1     FALSE          2 fem: 34, mal: 34
## 6 sex           Gentoo            5         0.960 FALSE          2 mal: 61, fem: 58
##
## ── Variable type: numeric ────────────────────────────────────────────────────────────────────────
##     skim_variable     species   n_missing complete_rate   mean      sd    p0   p25    p50    p75
##  1 bill_length_mm    Adelie            1         0.993  38.8  2.66   32.1  36.8  38.8  40.8
##  2 bill_length_mm    Chinstrap         0         1      48.8  3.34   40.9  46.3  49.6  51.1
##  3 bill_length_mm    Gentoo            1         0.992  47.5  3.08   40.9  45.3  47.3  49.6
##  4 bill_depth_mm     Adelie            1         0.993  18.3  1.22   15.5  17.5  18.4  19
##  5 bill_depth_mm     Chinstrap         0         1      18.4  1.14   16.4  17.5  18.4  19.4
##  6 bill_depth_mm     Gentoo            1         0.992  15.0  0.981  13.1  14.2  15    15.7
```

Summarize with **`skim()`**

`group_by` from `dplyr` package

Your Turn!

```
penguins_sp <- group_by(penguins, species)
skim(penguins_sp)
```

```
##
## ── Variable type: factor ───────────────────────────────────────────────────────
##   skim_variable species   n_missing complete_rate ordered n_unique top_counts
## 1 island        Adelie            0         1     FALSE          3 Dre: 56, Tor: 52, Bis: 44
## 2 island        Chinstrap         0         1     FALSE          1 Dre: 68, Bis: 0, Tor: 0
## 3 island        Gentoo            0         1     FALSE          1 Bis: 124, Dre: 0, Tor: 0
## 4 sex           Adelie            6         0.961 FALSE          2 fem: 73, mal: 73
## 5 sex           Chinstrap         0         1     FALSE          2 fem: 34, mal: 34
## 6 sex           Gentoo            5         0.960 FALSE          2 mal: 61, fem: 58
##
## ── Variable type: numeric ──────────────────────────────────────────────────────
##    skim_variable     species   n_missing complete_rate   mean      sd     p0    p25    p50    p75
##  1 bill_length_mm    Adelie            1         0.993   38.8   2.66    32.1   36.8   38.8   40.8
##  2 bill_length_mm    Chinstrap         0         1       48.8   3.34    40.9   46.3   49.6   51.1
##  3 bill_length_mm    Gentoo            1         0.992   47.5   3.08    40.9   45.3   47.3   49.6
##  4 bill_depth_mm     Adelie            1         0.993   18.3   1.22    15.5   17.5   18.4   19
##  5 bill_depth_mm     Chinstrap         0         1       18.4   1.14    16.4   17.5   18.4   19.4
##  6 bill_depth_mm     Gentoo            1         0.992   15.0   0.981   13.1   14.2   15     15.7
```

Exploring variable by variable

Here, use the penguins data set

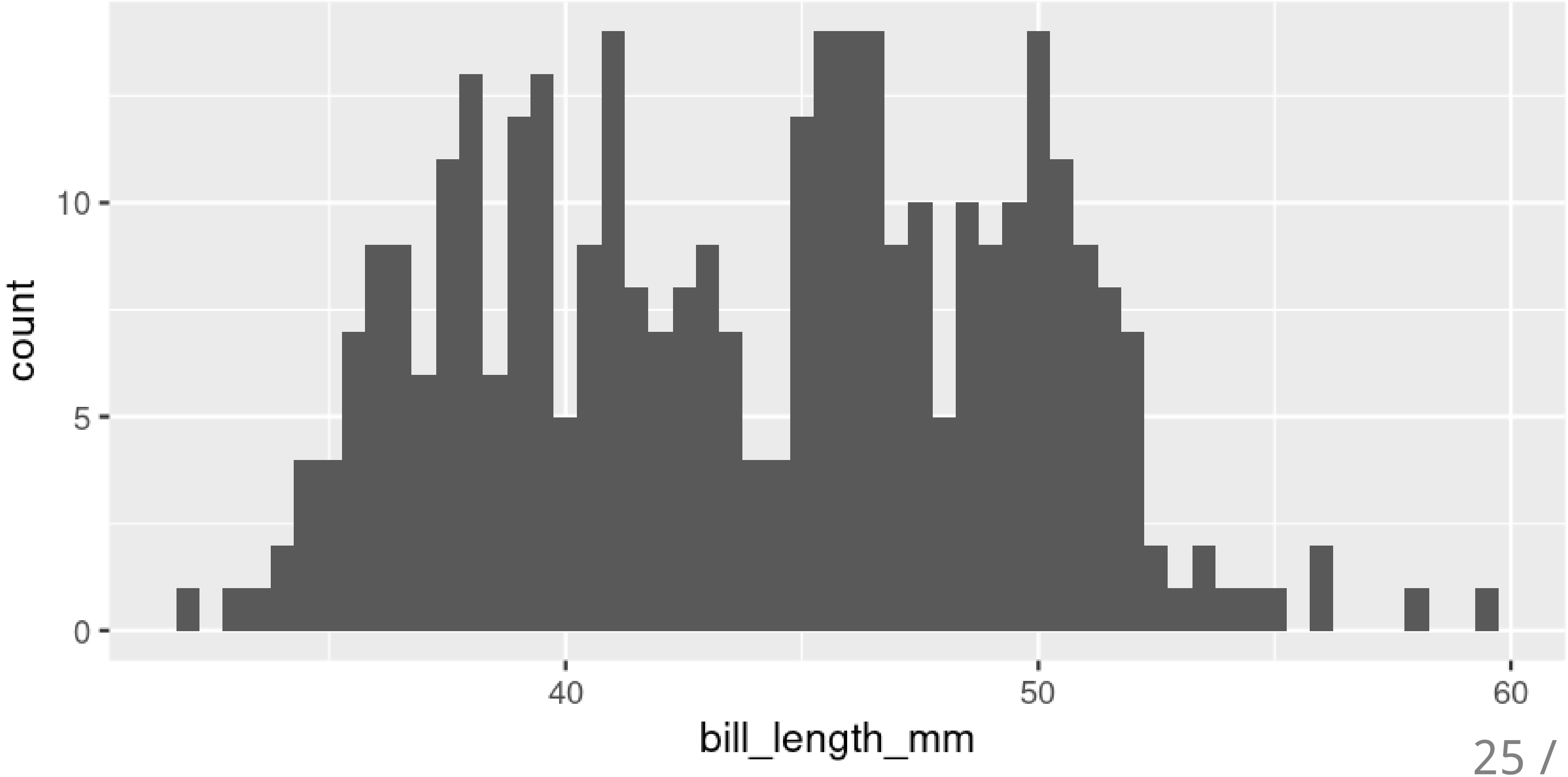
(explore your own for the assignment!)

Visualize with **`ggplot()`**

From last week...

- Histograms

```
ggplot(data = penguins, aes(x = bill_length_mm)) +
  geom_histogram(binwidth = 0.5)
```

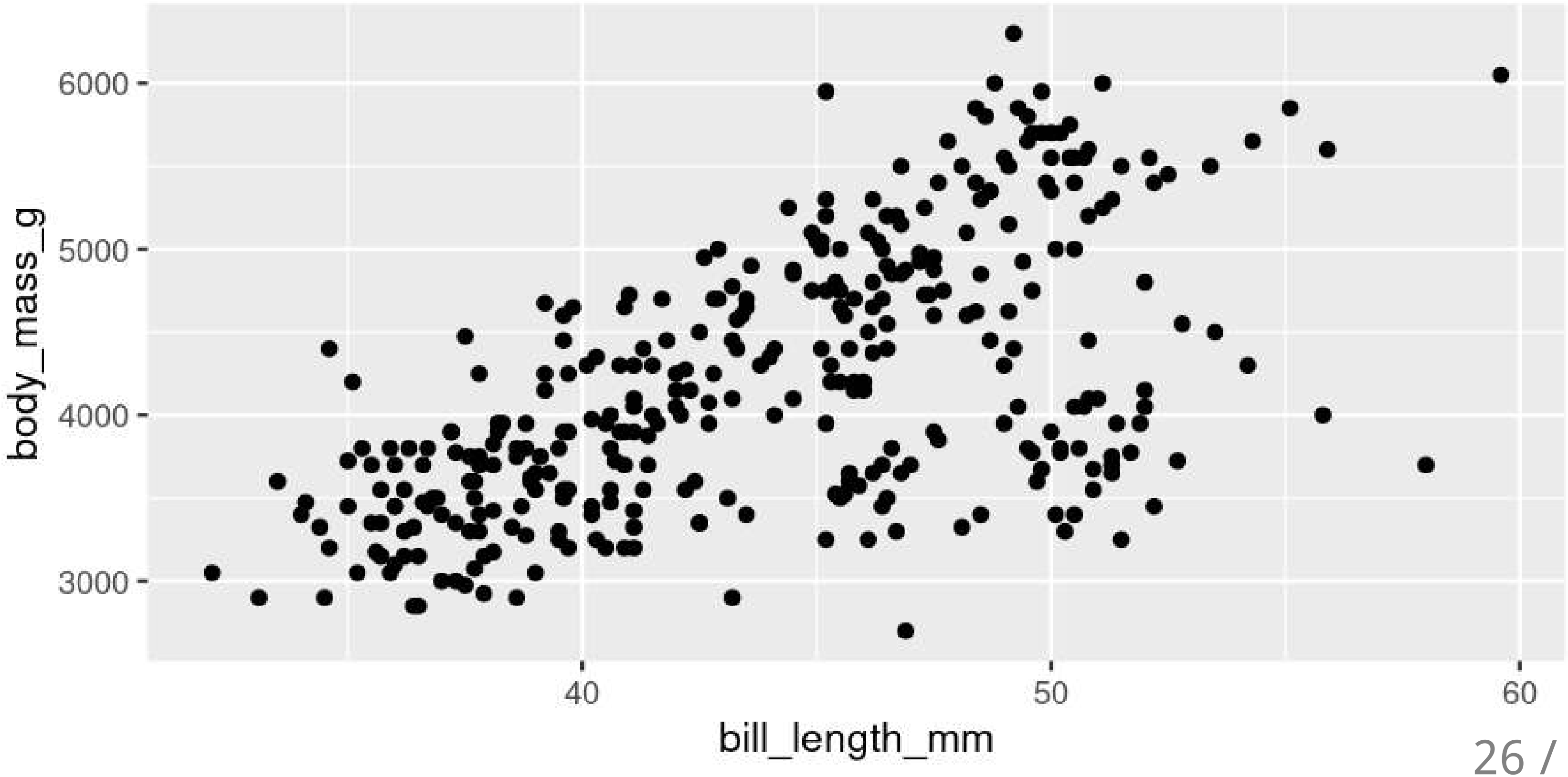


Visualize with **`ggplot()`**

From last week...

- Histograms
- Scatterplots

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

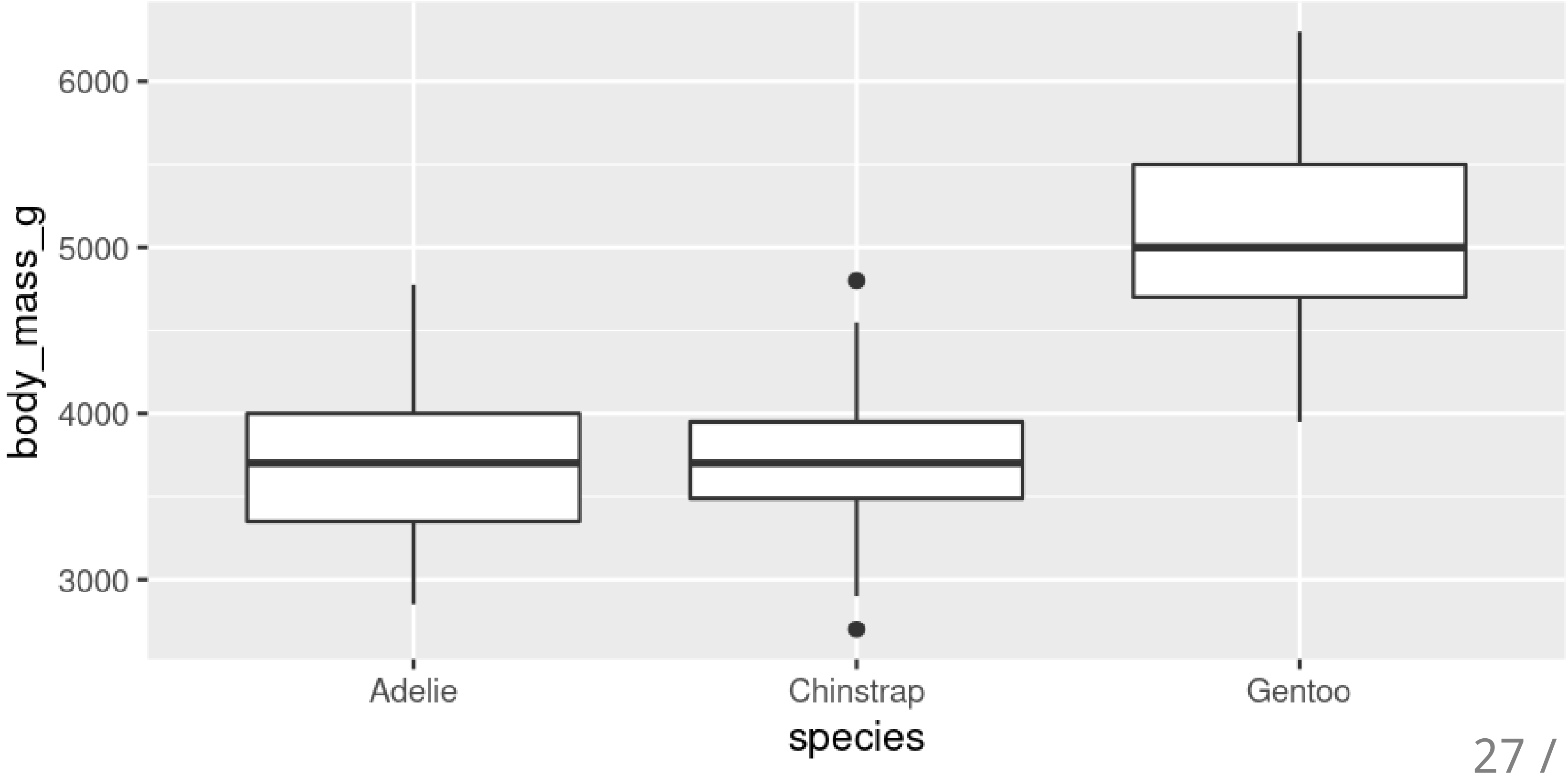


Visualize with **`ggplot()`**

From last week...

- Histograms
- Scatterplots
- Boxplots

```
ggplot(data = penguins, aes(x = species, y = body_mass_g)) +
  geom_boxplot()
```

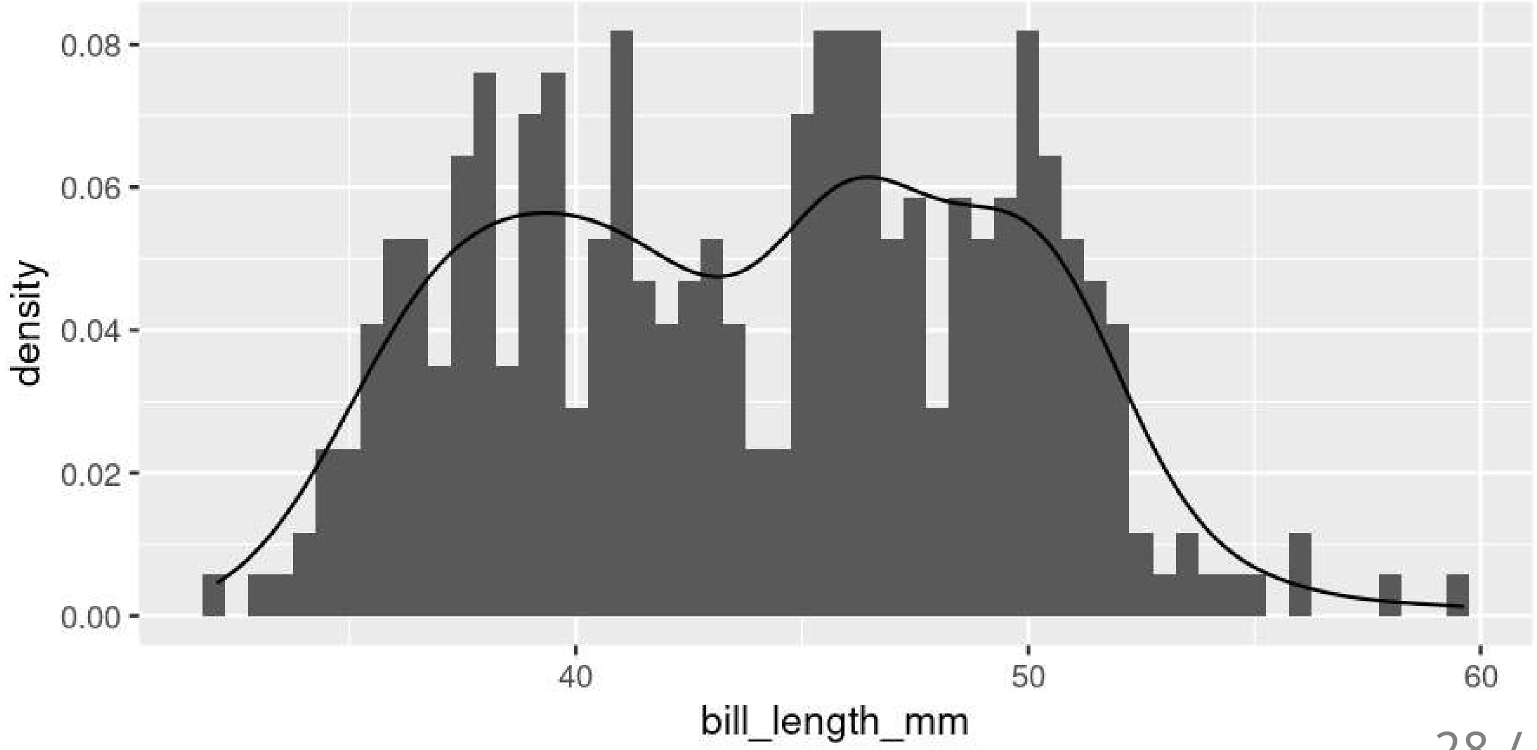


Visualize with **`ggplot()`**

Histogram with Density

- Default uses counts
- Here use density
  `y = ..density..`
- Same as density curve
  `geom_density()`
- Use to assess shape and distribution of data

```
ggplot(data = penguins, aes(x = bill_length_mm, y = ..density..)) +
    geom_histogram(binwidth = 0.5) +
    geom_density()
```

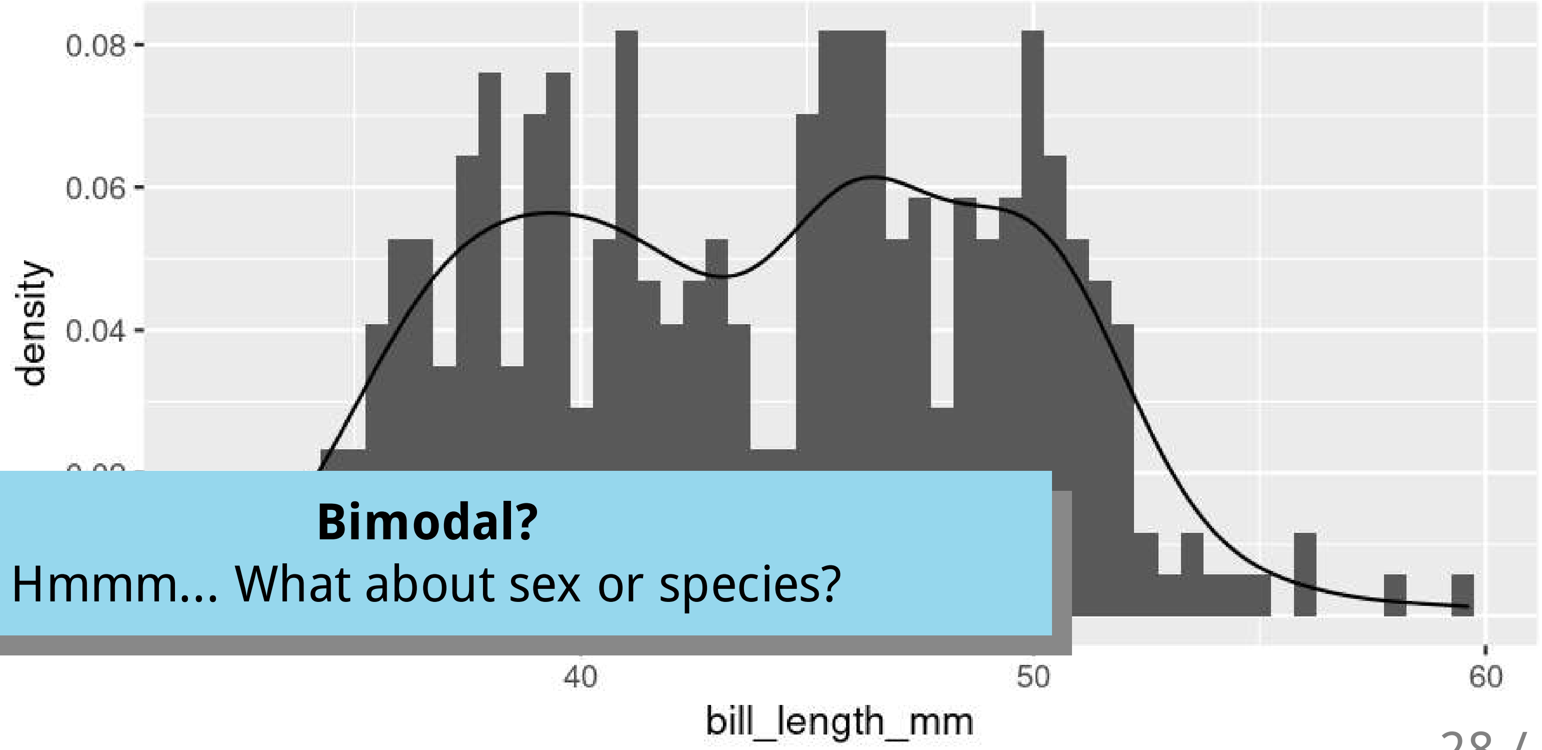


Visualize with **`ggplot()`**

Histogram with Density

- Default uses counts
- Here use density **`y = ..density..`**
- Same as density curve **`geom_density()`**
- Use to assess shape and distribution of data

```
ggplot(data = penguins, aes(x = bill_length_mm, y = ..density..)) +
    geom_histogram(binwidth = 0.5) +
    geom_density()
```



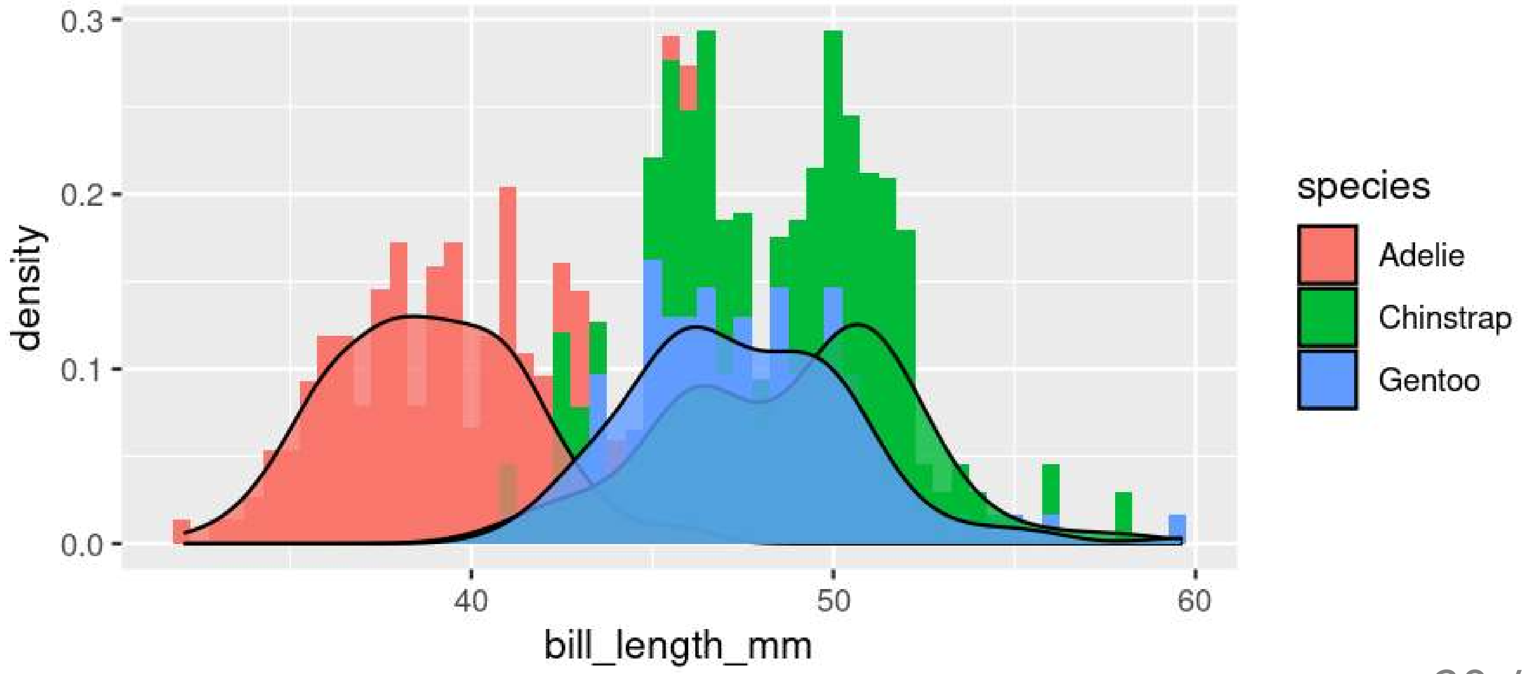
Bimodal?
Hmmm... What about sex or species?

Visualize with **`ggplot()`**

Histogram with Density

- Default uses counts
- Here use density **`y = ..density..`**
- Same as density curve **`geom_density()`**
- Use to assess shape and distribution of data

```
ggplot(data = penguins, aes(x = bill_length_mm, y = ..density..,
                            fill = species)) +
    geom_histogram(binwidth = 0.5) +
    geom_density(alpha = 0.8)
```

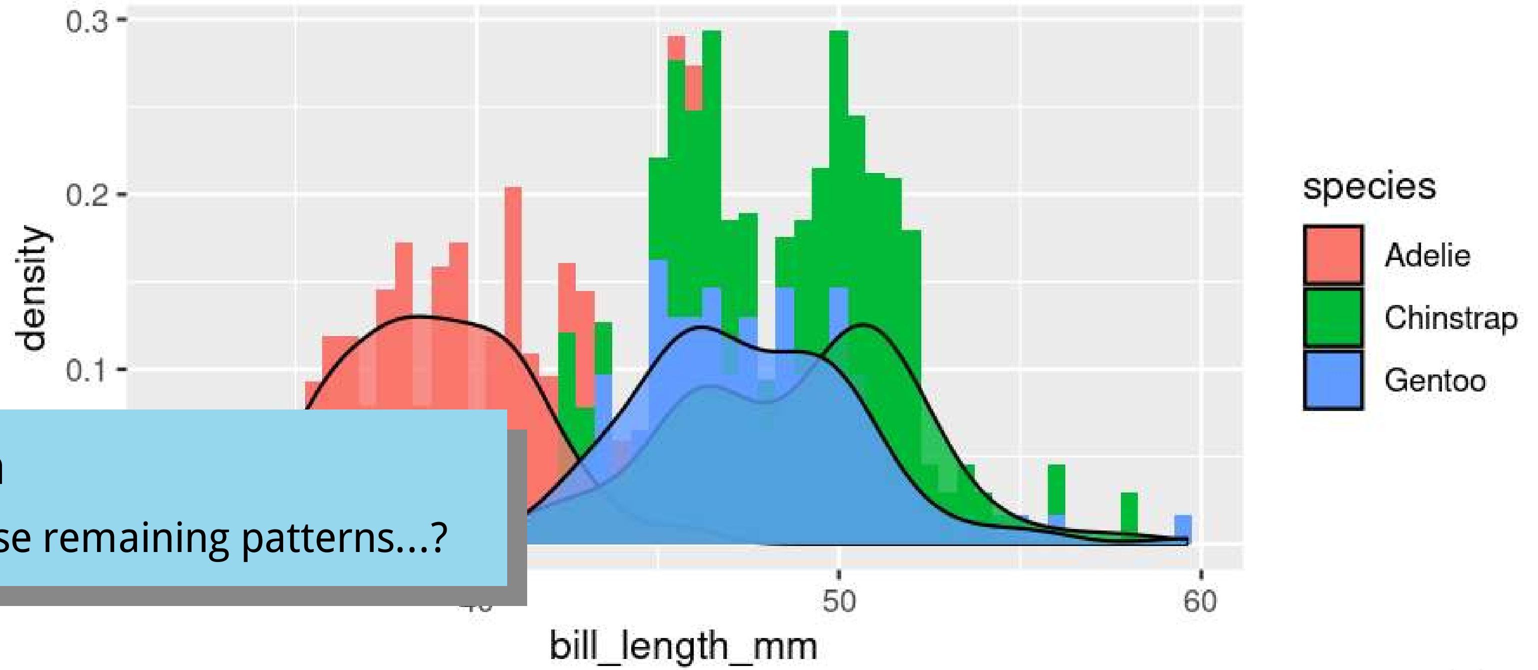


Visualize with **`ggplot()`**

Histogram with Density

- Default uses counts
- Here use density
  `y = ..density..`
- Same as density curve
  `geom_density()`
- Use to assess shape and distribution of data

```
ggplot(data = penguins, aes(x = bill_length_mm, y = ..density..,
                            fill = species)) +
    geom_histogram(binwidth = 0.5) +
    geom_density(alpha = 0.8)
```



Extra

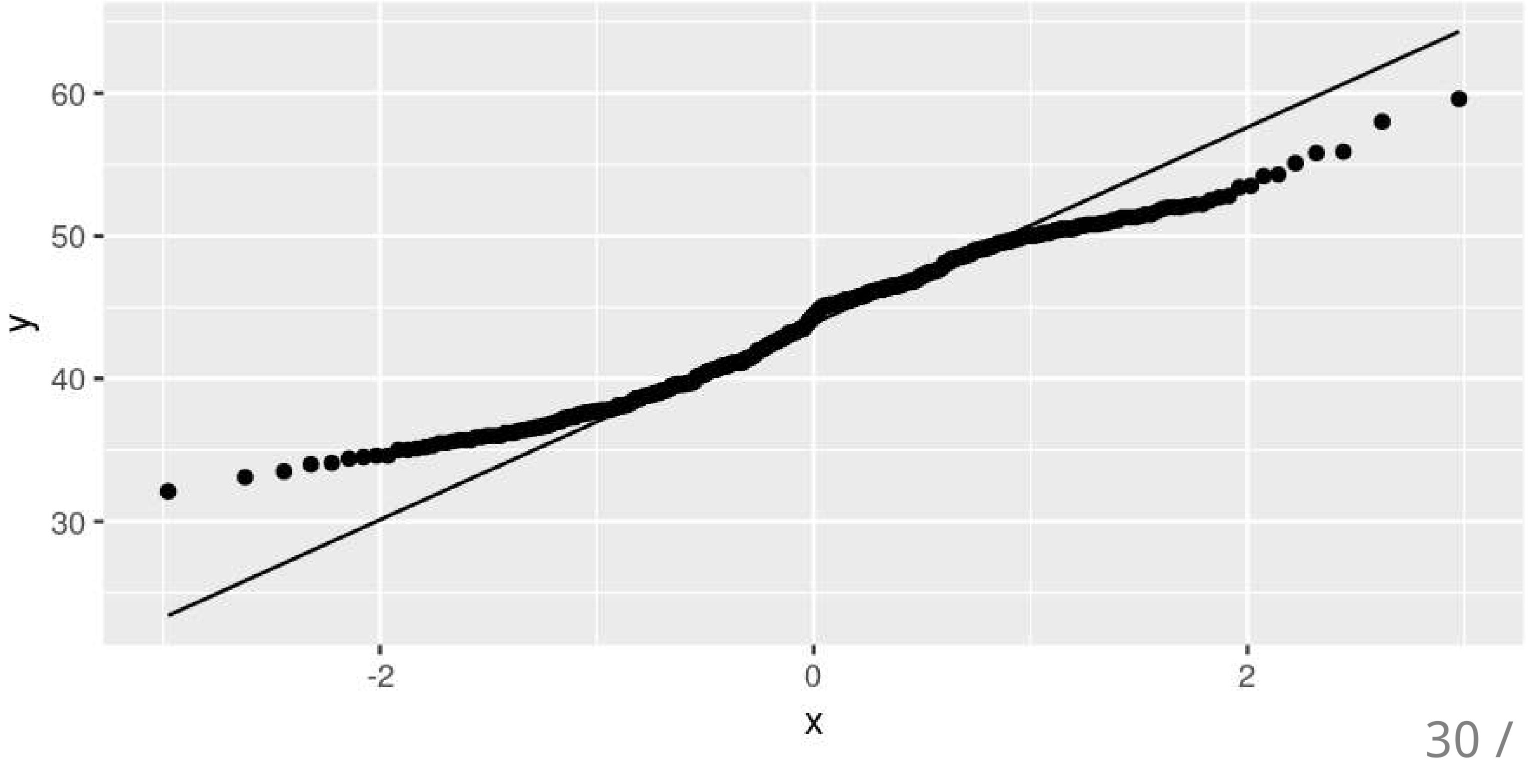
Perhaps sex plays a role in those remaining patterns...?

Visualize with `ggplot()`

QQ Norm plots

- Assess whether data follows normal distribution

```
ggplot(data = penguins, aes(sample = bill_length_mm)) +
  stat_qq() +     # Add the points
  stat_qq_line() # Add the line
```



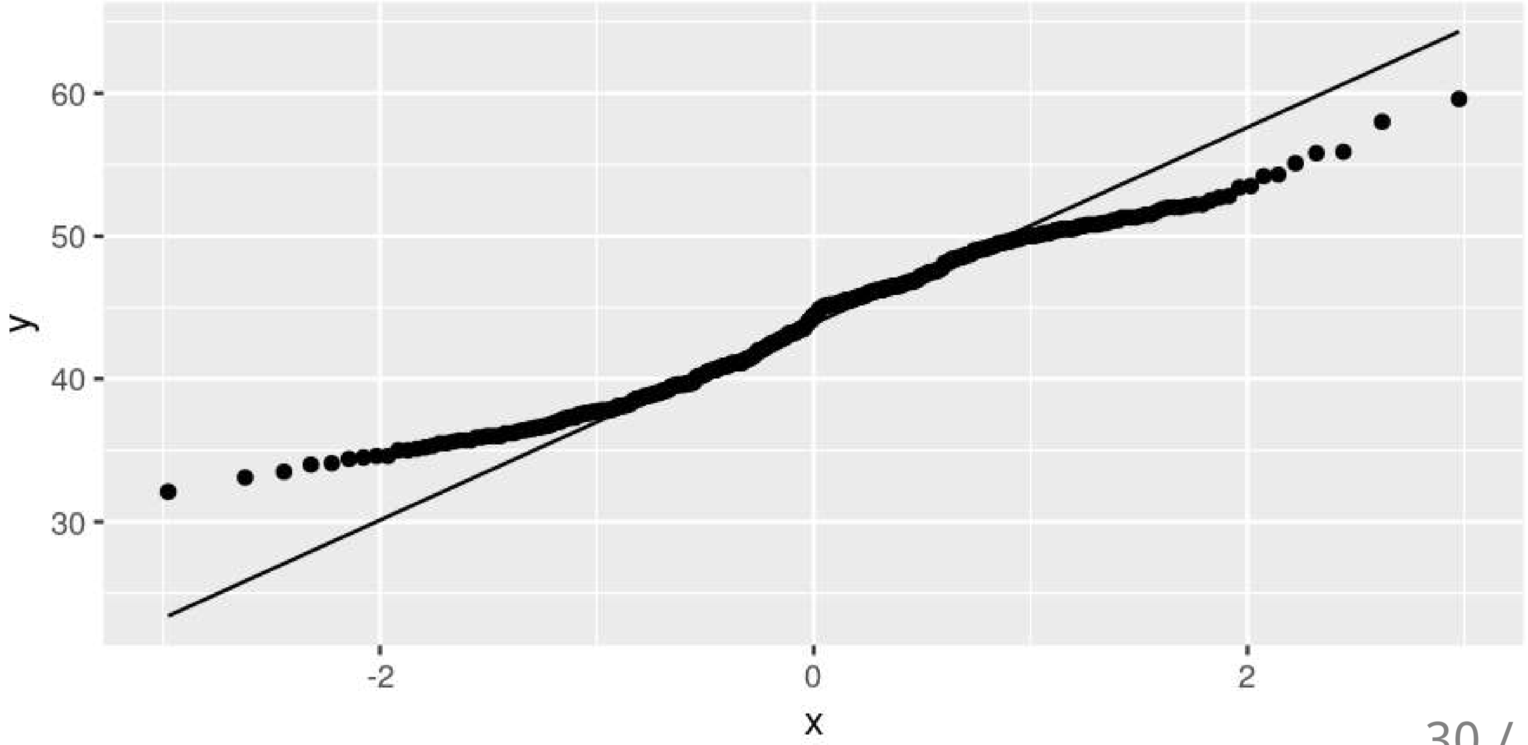
Visualize with `ggplot()`

QQ Norm plots

- Asses follow distrib

Here we are **NOT** assessing assumptions of normality for a model

That involves model residuals: Stay tuned for next week!



Summarize with `summarize()` Ha!

- From **`dplyr`** package (part of **`tidyverse`**)

```
summarize(penguins,
          mean_mass = mean(body_mass_g),
          sd_mass = sd(body_mass_g),
          median_mass = median(body_mass_g))
```

Side Note: `tidyverse` functions

- From **`dplyr`** package (part of **`tidyverse`**)

```
summarize(penguins,
          mean_mass = mean(body_mass_g),
          sd_mass = sd(body_mass_g),
          median_mass = median(body_mass_g))
```

`summarize()`

- **`tidyverse`** functions always start with the **data**, followed by **other arguments**
- you can reference any **column** from '**data**'
- **`summarize()`** creates a data frame with **new columns** (summarizes your data)

Summarize with **`summarize()`**

- From **`dplyr`** package (part of **`tidyverse`**)

```
summarize(penguins,
          mean_mass = mean(body_mass_g),
          sd_mass = sd(body_mass_g),
          median_mass = median(body_mass_g))
```

```
## # A tibble: 1 × 3
##   mean_mass sd_mass median_mass
##       <dbl>   <dbl>       <int>
## 1        NA      NA          NA
```

Summarize with **`summarize()`**

- From **`dplyr`** package (part of **`tidyverse`**)

```
summarize(penguins,
          mean_mass = mean(body_mass_g),
          sd_mass = sd(body_mass_g),
          median_mass = median(body_mass_g))
```

```
## # A tibble: 1 × 3
##   mean_mass sd_mass median_mass
##       <dbl>   <dbl>       <int>
## 1        NA      NA          NA
```

> Why all **`NA`**s?

Summarize with `summarize()`

- **`mean()`**, **`sd()`**, **`median()`**

Need to tell summary statistic functions to remove missing values

`na.rm = TRUE`

```
summarize(penguins,
          mean_mass = mean(body_mass_g, na.rm = TRUE),
          sd_mass = sd(body_mass_g, na.rm = TRUE),
          median_mass = median(body_mass_g, na.rm = TRUE))
```

```
## # A tibble: 1 × 3
##   mean_mass sd_mass median_mass
##       <dbl>   <dbl>       <dbl>
## 1     4202.    802.        4050
```

Summarize with `summarize()`

- **`mean()`**, **`sd()`**, **`median()`**, **`quantile()`**, **`n()`***

```
summarize(penguins,
          mean_mass = mean(body_mass_g, na.rm = TRUE),
          sd_mass = sd(body_mass_g, na.rm = TRUE),
          median_mass = median(body_mass_g, na.rm = TRUE),
          q25_mass = quantile(body_mass_g, probs = 0.25, na.rm = TRUE),
          n = n(),                                                      # Sample size
          n_no_missing = sum(!is.na(body_mass_g))                       # Non-missing sample size
```

```
## # A tibble: 1 × 6
##   mean_mass sd_mass median_mass q25_mass     n n_no_missing
##       <dbl>   <dbl>       <dbl>    <dbl> <int>        <int>
## 1     4202.    802.        4050     3550   344          342
```

* **`n()`** only works *inside* **`summarize()`**/**`mutate()`**

Your Turn: **`summarize()`**

Calculate summary statistics for **Bill Length**

```
summarize(penguins,
                                bill_length_mm               ,
                            bill_length_mm               ,
                                    bill_length_mm               ,
                                   bill_length_mm                             ,
          ,
                                        bill_length_mm  )
```

Your Turn: **`summarize()`**

Calculate summary statistics for **Bill Length**

```
summarize(penguins,
          mean_bill_length = mean(bill_length_mm, na.rm = TRUE),
          sd_bill_length = sd(bill_length_mm, na.rm = TRUE),
          median_bill_length = median(bill_length_mm, na.rm = TRUE),
          q25_bill_length = quantile(bill_length_mm, probs = 0.25, na.rm = TRUE),
          n_mass = n(),
          n_no_missing_mass = sum(!is.na(bill_length_mm)))
```

```
## # A tibble: 1 × 6
##   mean_bill_length sd_bill_length median_bill_length q25_bill_length n_mass n_no_missing_mass
##              <dbl>          <dbl>              <dbl>           <dbl>  <int>             <int>
## 1             43.9           5.46               44.4            39.2    344               342
```

Side Note: Removing **NA**s

- With arguments
  - `na.rm = TRUE` (summary stats i.e. `mean()`, `sd()`)
  - `na.action = na.exclude` (models i.e., `lm()`, `lmer()`)
- You can remove all `NA`s from your data (`drop_na()`)
- You can selectively remove `NA`s from your data (`filter()`)

Side Note: Removing **`NA`**s

Remove all `NA`s

- This removes **every** row that has an **`NA`** in **any** column
- **`drop_na()`** function from **`tidyr`** package (part of **`tidyverse`**)

```
penguins_no_na <- drop_na(penguins)
```

- Consider removing columns with lots of **`NA`**s first (assuming you don't need them)

```
penguins_no_na <- select(penguins, -sex)
penguins_no_na <- drop_na(penguins_no_na)
```

Side Side Note: `tidyverse` functions

- From **`tidyr`** package (part of **`tidyverse`**)

```
penguins_no_na <- drop_na(penguins)
```

`drop_na()`

- **`tidyverse`** functions always start with the **data**, followed by other arguments
- here, there are no other arguments

Side Note: Removing **NAs**

Selectively remove `NA`s with `filter()`

- From **`dplyr`** package (part of tidyverse)

```
filter(penguins, !is.na(body_mass_g))
```

- `is.na()` checks if there is an `NA` and returns `TRUE` if so
- `!` turns a `TRUE` into a `FALSE`
- `filter()` only keeps rows that are `TRUE`
- **Thus** any row with an `NA` in `body_mass_g` is removed

Side Side Note: `tidyverse` functions

- From **`dplyr`** package (part of **`tidyverse`**)

```
filter(penguins, !is.na(body_mass_g))
```

`filter()`

- **`tidyverse`** functions always start with the **data**, followed by other arguments
- you can reference any **column** from '**data**'
- **`filter()`** keeps only rows that return **TRUE** to the logical statements

Summarize with **`summarize()`** (and **`group_by()`**)

- Can also use **`group_by()`** to calculate summaries by groups

```
penguins_sp <- group_by(penguins, species)
summarize(penguins_sp,
          mean_mass = mean(body_mass_g, na.rm = TRUE),
          sd_mass = sd(body_mass_g, na.rm = TRUE),
          median_mass = median(body_mass_g, na.rm = TRUE))
```

```
## # A tibble: 3 × 4
##   species   mean_mass sd_mass median_mass
##   <fct>         <dbl>   <dbl>       <dbl>
## 1 Adelie        3701.    459.        3700
## 2 Chinstrap     3733.    384.        3700
## 3 Gentoo        5076.    504.        5000
```

Summarize with **`summarize()`** (and **`group_by()`**)

- Can also use **`group_by()`** to calculate summaries by groups

```
penguins_sp_sex <- group_by(penguins, species, sex)
summarize(penguins_sp_sex,
          mean_mass = mean(body_mass_g, na.rm = TRUE),
          sd_mass = sd(body_mass_g, na.rm = TRUE),
          median_mass = median(body_mass_g, na.rm = TRUE))
```

```
## # A tibble: 8 × 5
## # Groups:   species [3]
##   species   sex    mean_mass sd_mass median_mass
##   <fct>     <fct>      <dbl>   <dbl>       <dbl>
## 1 Adelie    female     3369.    269.       3400
## 2 Adelie    male       4043.    347.       4000
## 3 Adelie    <NA>       3540     477.       3475
## 4 Chinstrap female     3527.    285.       3550
## 5 Chinstrap male       3939.    362.       3950
## 6 Gentoo    female     4680.    282.       4700
## 7 Gentoo    male       5485.    313.       5500
## 8 Gentoo    <NA>       4588.    338.       4688.
```

Summarize with **`summarize()`** (and **`group_by()`**)

- Can also use **`group_by()`** to calculate summaries by groups

```
penguins_sp_sex <- group_by(penguins, species, sex)
summarize(penguins_sp_sex,
          mean_mass = mean(body_mass_g, na.rm = TRUE),
          sd_mass = sd(body_mass_g, na.rm = TRUE),
          median_mass = median(body_mass_g, na.rm = TRUE))
```

```
## # A tibble: 8 × 5
## # Groups:   species
##   species   sex     mean_mass sd_mass median_mass
##   <fct>     <fct>         <dbl>   <dbl>       <dbl>
## 1 Adelie    female      3369.    269.        3400
## 2 Adelie    male        4043.    347.        4000
## 3 Adelie    <NA>        3540     477.        3475
## 4 Chinstrap female      3527.    285.        3550
## 5 Chinstrap male        3939.    362.        3950
## 6 Gentoo    female      4680.    282.        4700
## 7 Gentoo    male        5485.    313.        5500
## 8 Gentoo    <NA>        4588.    338.        4688.
```

Where are the decimal points?

Side Note: Where are the decimal points?

- **`tibble`** hides them for easy viewing

```
penguins_sum <- summarize(penguins_sp_sex,
                          mean_mass = mean(body_mass_g, na.rm = TRUE),
                          sd_mass = sd(body_mass_g, na.rm = TRUE),
                          median_mass = median(body_mass_g, na.rm = TRUE))
penguins_sum
```

```
## # A tibble: 8 × 5
## # Groups:   species [3]
##   species   sex    mean_mass sd_mass median_mass
##   <fct>     <fct>      <dbl>   <dbl>       <dbl>
## 1 Adelie    female     3369.    269.        3400
## 2 Adelie    male       4043.    347.        4000
## 3 Adelie    <NA>       3540     477.        3475
## 4 Chinstrap female     3527.    285.        3550
## 5 Chinstrap male       3939.    362.        3950
## 6 Gentoo    female     4680.    282.        4700
## 7 Gentoo    male       5485.    313.        5500
## 8 Gentoo    <NA>       4588.    338.        4688.
```

Side Note: Where are the decimal points?

- **`tibble`** hides them for easy viewing

```
penguins_sum <- summarize(penguins_sp_sex,
                          mean_mass = mean(body_mass_g, na.rm = TRUE),
                          sd_mass = sd(body_mass_g, na.rm = TRUE),
                          median_mass = median(body_mass_g, na.rm = TRUE))
penguins_sum
```

```
## # A tibble: 8 × 5
## # Groups:   species [3]
##   species   sex    mean_mass sd_mass median_mass
##   <fct>     <fct>      <dbl>   <dbl>       <dbl>
## 1 Adelie    female     3369.    269.        3400
## 2 Adelie    male       4043.    347.        4000
## 3 Adelie    <NA>       3540     477.        3475
## 4 Chinstrap female     3527.    285.        3550
## 5 Chinstrap male       3939.    362.        3950
## 6 Gentoo    female     4680.    282.        4700
## 7 Gentoo    male       5485.    313.        5500
## 8 Gentoo    <NA>       4588.    338.        4688.
```

Note

If you want to keep the output, you need to assign (**`<-`**) it to an object. Here, **`penguins_sum`**

Side Note: Where are the decimal points?

- **`as.data.frame()`** to see the raw data

```
as.data.frame(penguins_sum)
```

```
##     species    sex mean_mass  sd_mass median_mass
## 1    Adelie female  3368.836 269.3801      3400.0
## 2    Adelie   male  4043.493 346.8116      4000.0
## 3    Adelie   <NA>  3540.000 477.1661      3475.0
## 4 Chinstrap female  3527.206 285.3339      3550.0
## 5 Chinstrap   male  3938.971 362.1376      3950.0
## 6    Gentoo female  4679.741 281.5783      4700.0
## 7    Gentoo   male  5484.836 313.1586      5500.0
## 8    Gentoo   <NA>  4587.500 338.1937      4687.5
```

- Or click on the name in the Environment pane

Side Note: Where are all my data?

```
penguins
```

```
## # A tibble: 344 × 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
##  7 Adelie  Torgersen           38.9          17.8               181        3625
##  8 Adelie  Torgersen           39.2          19.6               195        4675
##  9 Adelie  Torgersen           34.1          18.1               193        3475
## 10 Adelie  Torgersen           42            20.2               190        4250
## # … with 334 more rows, and 2 more variables: sex <fct>, year <int>
```

`... with 334 more rows, and 2 more variables: sex <fct>, year <int>`

Side Note: Where are all my data?

```r
print(penguins, n = Inf)
```

```
## # A tibble: 344 × 8
##     species   island    bill_length_mm bill_depth_mm flipper_length_… 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
##   7 Adelie    Torgersen           38.9          17.8              181        3625
##   8 Adelie    Torgersen           39.2          19.6              195        4675
##   9 Adelie    Torgersen           34.1          18.1              193        3475
##  10 Adelie    Torgersen           42            20.2              190        4250
##  11 Adelie    Torgersen           37.8          17.1              186        3300
##  12 Adelie    Torgersen           37.8          17.3              180        3700
##  13 Adelie    Torgersen           41.1          17.6              182        3200
##  14 Adelie    Torgersen           38.6          21.2              191        3800
##  15 Adelie    Torgersen           34.6          21.1              198        4400
```

Side Note: Where are all my data?

```r
as.data.frame(penguins)
```

```
##         species    island bill_length_mm bill_depth_mm flipper_length_mm body_mass_g    sex year
## 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          18.0               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           42.0          20.2               190        4250   <NA> 2007
## 11     Adelie Torgersen           37.8          17.1               186        3300   <NA> 2007
## 12     Adelie Torgersen           37.8          17.3               180        3700   <NA> 2007
## 13     Adelie Torgersen           41.1          17.6               182        3200 female 2007
## 14     Adelie Torgersen           38.6          21.2               191        3800   male 2007
## 15     Adelie Torgersen           34.6          21.1               198        4400   male 2007
## 16     Adelie Torgersen           36.6          17.8               185        3700 female 2007
## 17     Adelie Torgersen           38.7          19.0               195        3450 female 2007
```

Summarize with `summarize()`

`skewness()`, `kurtosis()`

- From **`moments`** package

```
library(moments)

summarize(penguins,
          skew_mass = skewness(body_mass_g, na.rm = TRUE),
          kurt_mass = kurtosis(body_mass_g, na.rm = TRUE))
```

```
## # A tibble: 1 × 2
##   skew_mass kurt_mass
##       <dbl>     <dbl>
## 1     0.468      2.27
```

1. Normal distribution, skew = 0, kurtosis = 3*
2. Remember that it's best to evaluate the distribution **both** visually and statistically

* ***Excess*** **kurtosis** would be 0 for a normal distribution, but this functions measures **kurtosis**

Summarize with **`summarize()`**

Confidence Intervals

- By hand!
- 95% Confidence interval ranges from [mean - (1.96 *SE)] to [mean + (1.96* SE)]
- You can also express this interval as: mean +/- (1.96 * SE)
- Standard Errors (SE) can be calculated by SD / sqrt(n)

```r
summarize(penguins,
          mean_mass = mean(body_mass_g, na.rm = TRUE),
          sd_mass = sd(body_mass_g, na.rm = TRUE),
          n = n(),
          se_mass = sd_mass / sqrt(n),        # Calculate Standard Error
          ci_mass = 1.96 * se_mass,           # CI margin of error
          ci_low_mass = mean_mass - ci_mass,  # The lower range
          ci_high_mass = mean_mass + ci_mass) # The upper range
```

Summarize with `summarize()`

Confidence Intervals

- By hand!
- 95% Confidence interval ranges from [mean - (1.96 *SE)] to [mean + (1.96* SE)]
- You can also express this interval as: mean +/- (1.96 * SE)
- Standard Errors (SE) can be calculated by SD / sqrt(n)

```
## # A tibble: 1 × 7
##   mean_mass sd_mass     n se_mass ci_mass ci_low_mass ci_high_mass
##       <dbl>   <dbl> <int>   <dbl>   <dbl>       <dbl>        <dbl>
## 1     4202.    802.   344    43.2    84.7       4117.        4287.
```

Put it All Together

```r
penguins_sp <- group_by(penguins, species)
summarize(penguins_sp,
          mean_mass = mean(body_mass_g, na.rm = TRUE),
          sd_mass = sd(body_mass_g, na.rm = TRUE),
          q25_mass = quantile(body_mass_g, probs = 0.25, na.rm = TRUE),
          median_mass = median(body_mass_g, na.rm = TRUE),
          q75_mass = quantile(body_mass_g, probs = 0.25, na.rm = TRUE),
          n = n(),
          n_no_missing = sum(!is.na(body_mass_g)),
          skew_mass = skewness(body_mass_g, na.rm = TRUE),
          kurt_mass = kurtosis(body_mass_g, na.rm = TRUE),
          se_mass = sd_mass / sqrt(n),
          ci_mass = 1.96 * se_mass,
          ci_low_mass = mean_mass - ci_mass,
          ci_high_mass = mean_mass + ci_mass)
```

Put it All Together

```
##     species mean_mass  sd_mass q25_mass median_mass q75_mass   n n_no_missing  skew_mass kurt_mass
## 1    Adelie  3700.662 458.5661   3350.0        3700   3350.0 152          151 0.28249381  2.405611
## 2 Chinstrap  3733.088 384.3351   3487.5        3700   3487.5  68           68 0.24194125  3.463681
## 3    Gentoo  5076.016 504.1162   4700.0        5000   4700.0 124          123 0.06878276  2.257871
##    se_mass  ci_mass ci_low_mass ci_high_mass
## 1 37.19462 72.90146    3627.761     3773.564
## 2 46.60747 91.35065    3641.738     3824.439
## 3 45.27097 88.73111    4987.285     5164.747
```

Put it All Together **(Advanced!)**

`pivot_longer()` transposes data

- from **`tidyr`** package (part of **`tidyverse`**)

```
penguins_long <- pivot_longer(penguins,
                              cols = c(bill_length_mm, bill_depth_mm, flipper_length_mm,
body_mass_g),
                              names_to = "measurement", values_to = "values")

penguins_long
```

```
## # A tibble: 1,376 × 6
##    species island    sex     year measurement       values
##    <fct>   <fct>     <fct>  <int> <chr>              <dbl>
##  1 Adelie  Torgersen male    2007 bill_length_mm      39.1
##  2 Adelie  Torgersen male    2007 bill_depth_mm       18.7
##  3 Adelie  Torgersen male    2007 flipper_length_mm  181
##  4 Adelie  Torgersen male    2007 body_mass_g       3750
##  5 Adelie  Torgersen female  2007 bill_length_mm      39.5
##  6 Adelie  Torgersen female  2007 bill_depth_mm       17.4
##  7 Adelie  Torgersen female  2007 flipper_length_mm  186
```

Put it All Together **(Advanced!)**

`pivot_longer()` transposes data

- from **`tidyr`** package (part of **`tidyverse`**)

```
penguins_long <- pivot_longer(penguins,
                              cols = c(bill_length_mm, bill_depth_mm, flipper_length_mm,
body_mass_g),
                              names_to = "measurement", values_to = "values")

penguins_long
```

```
## # A tibble: 1,376 × 6
##    species island    sex     year measurement       values
##    <fct>   <fct>     <fct>  <int> <chr>              <dbl>
##  1 Adelie  Torgersen male    2007 bill_length_mm      39.1
##  2 Adelie  Torgersen male    2007 bill_depth_mm       18.7
##  3 Adelie  Torgersen male    2007 flipper_length_mm  181
##  4 Adelie  Torgersen male    2007 body_mass_g       3750
##  5 Adelie  Torgersen female  2007 bill_length_mm      39.5
##  6 Adelie  Torgersen female  2007 bill_depth_mm       17.4
##  7 Adelie  Torgersen female  2007 flipper_length_mm  186
```

Extra

Compare **`penguins`** to **`penguins_long`**. Can you see what the **`pivot_longer()`** function is doing?

Put it All Together **(Advanced!)**

```
penguins_long_sp <- group_by(penguins_long, species, measurement)

summarize(penguins_long_sp,
          mean = mean(values, na.rm = TRUE),
          sd = sd(values, na.rm = TRUE),
          q25 = quantile(values, probs = 0.25, na.rm = TRUE),
          median = median(values, na.rm = TRUE),
          q75 = quantile(values, probs = 0.25, na.rm = TRUE),
          n = n(),
          n_no_missing = sum(!is.na(values)),
          skew = skewness(values, na.rm = TRUE),
          kurt = kurtosis(values, na.rm = TRUE))
```

Put it All Together **(Advanced!)**

```
## `summarise()` has grouped output by 'species'. You can override using the `.groups` argument.
```

```
## # A tibble: 12 × 11
## # Groups:   species [3]
##    species   measurement        mean     sd    q25 median    q75     n n_no_missing     skew
kurt
##    <fct>     <chr>             <dbl>  <dbl>  <dbl>  <dbl>  <dbl> <int>        <int>    <dbl>
<dbl>
##  1 Adelie    bill_depth_mm      18.3   1.22   17.5   18.4   17.5   152          151  0.318
2.90
##  2 Adelie    bill_length_mm     38.8   2.66   36.8   38.8   36.8   152          151  0.160
2.81
##  3 Adelie    body_mass_g      3701.  459.   3350   3700   3350     152          151  0.282
2.41
##  4 Adelie    flipper_length_mm  190.   6.54  186    190    186     152          151  0.0865
3.28
##  5 Chinstrap bill_depth_mm      18.4   1.14   17.5   18.4   17.5    68           68  0.00673
2.10
##  6 Chinstrap bill_length_mm     48.8   3.34   46.3   49.6   46.3    68           68 -0.0886
2.95
```

All Data vs. Variable by Variable

Depends on what you need

- `ggpairs()` and `skim()`
    - Lots of data quickly summarized and examined
    - Less easily customized (but still possible!)

- `ggplot()` and `summarize()`
    - Take a bit longer to write out
    - Very customizable
    - Can easily include stats not available in `ggpairs()` and `skim()`

Wrapping up: Further reading (all **Free**!)

- RStudio > Help > Cheatsheets > Data Transformation with dplyr
- R for Data Science
  - Data transformation
  - Exploratory Data Analysis