ODSC: Data Visualization with ggplot2

Part 1: Thinking with graphs

by Martin Frigaard

Written: February 08 2022

Updated: March 13 2022

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Resources

Links:

- Conference Website
- Website
- Part 1
- Part 2

Materials:

- RStudio.Cloud
- Github Repo

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Outline

Part 1

Exploratory data analysis

- *What is it, who does it, and why it's important*

A Bayesian mindset

- *Priors → new information → posteriors*

The grammar of graphics

- *Layers, aesthetics, and geoms*

Part 2

Build labels first

- *Set expectations*

Exercises & solutions

- *RStudio.Cloud*

Creating graphs

- *Building graphs layer-by-layer, global vs. local mapping, visual encodings*

Applying the grammar

- *Mapping vs. setting aesthetics, combining layers, facets*

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

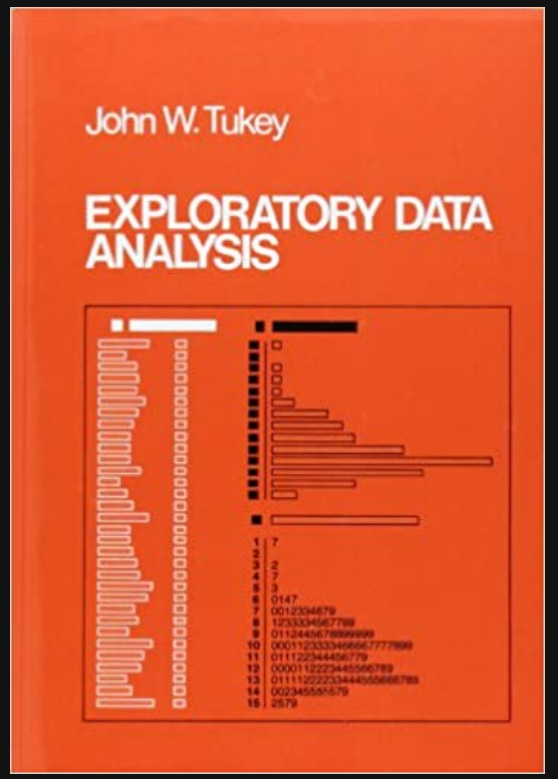
PART 1

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Exploratory Data Analysis (EDA)

"EDA"

"Exploratory Data Visualization" first coined by American mathematician John Tukey in 1977



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

What is EDA?

John T. Behrens, Principles and Procedures of Exploratory Data Analysis:

Emphasis on substantive understanding of data

- i.e. "what is going on here?"

Iterative process with a focus on graphic representations of data

What is EDA?

John T. Behrens, Principles and Procedures of Exploratory Data Analysis:

> *- Includes subset analyses, skepticism, and flexibility*
>
> *- The role of the data analyst is to listen to the data in as many ways as possible until a plausible "story" of the data is apparent*

Who does EDA?

John Tukey, Exploratory Data Analysis:

A detective investigating a crime needs both tools and understanding.

If he has no fingerprint powder, he will fail to find fingerprints on most surfaces.

If he does not understand where the criminal is likely to have put his fingers, he will not look in the right places.

Equally, the analyst of data needs both tool and understanding.

EDA is a 'state of mind'

Hadley Wickham, R for Data Science:

> *More than anything, EDA is a state of mind.*
>
> *During the initial phases of EDA you should feel free to investigate every idea that occurs to you. Some of these ideas will pan out, and some will be dead ends.*
>
> *As your exploration continues, you will home in on a few particularly productive areas that you'll eventually write up and communicate to others.*

Why is EDA important?

"Data are becoming the new raw material of business" - Craig Mundie, CEO at Microsoft

"Data is the oil of the digital era" - The Economist

https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data

Why is EDA important?

The Future of AI Gathers Here

Data are complex:

It's hard to derive insight from data in it's raw form!

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

EDA is a means of visualizing complexity

ODSC The Future of AI Gathers Here

- It's hard to make sense of a dataset or database with millions of rows and thousands of columns

- Fortunately, humans are excellent at seeing patterns:

Superior pattern processing is the essence of the evolved human brain - Frontiers in Neuroscience

What do you need?

Tools = R, RStudio, Adobe, sketch pad, text editor (Atom, Sublime Text, Vim)

Understanding = *...experience and feedback*

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

A Bayesian Mindset

A Bayesian Mindset

What we thought we knew (what we expect)

+

New information (what we see)

=

What we think now (what we've learned)

A Bayesian Mindset

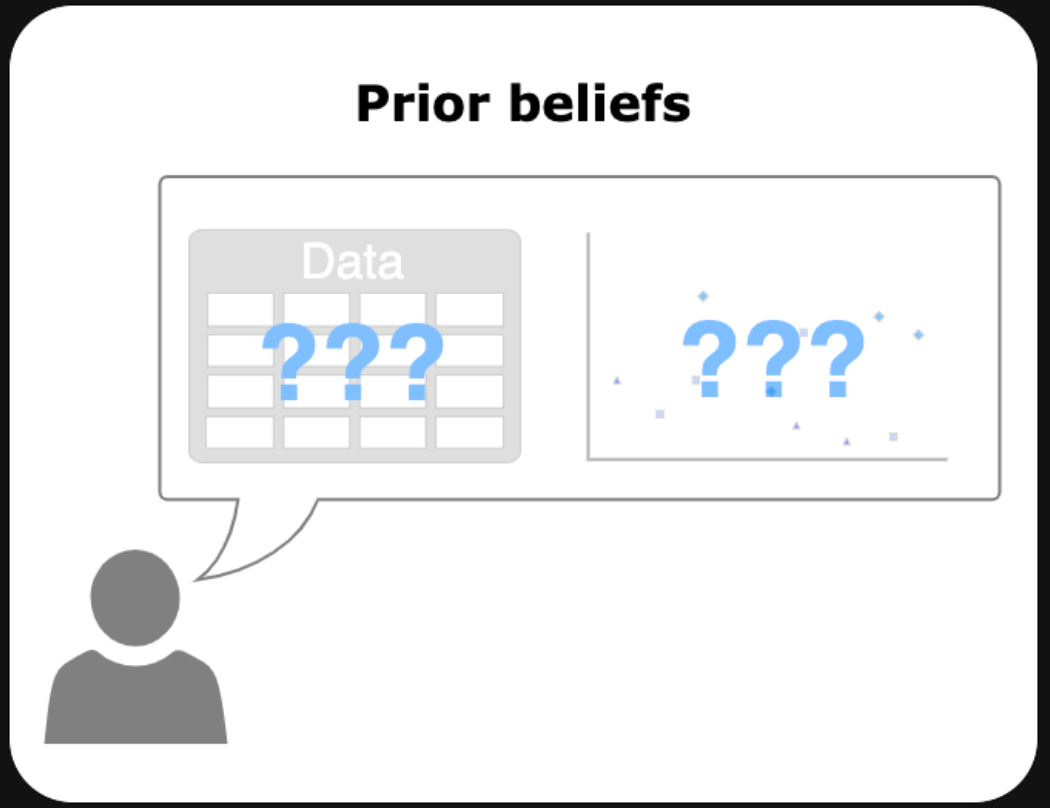
We all have implicit beliefs (*'priors'*) about the world

When we encounter new data or information, our *priors* get updated

These updated beliefs (*'posteriors'*) depend on our implicit beliefs and our **perceptions** of the new information

A Bayesian Mindset

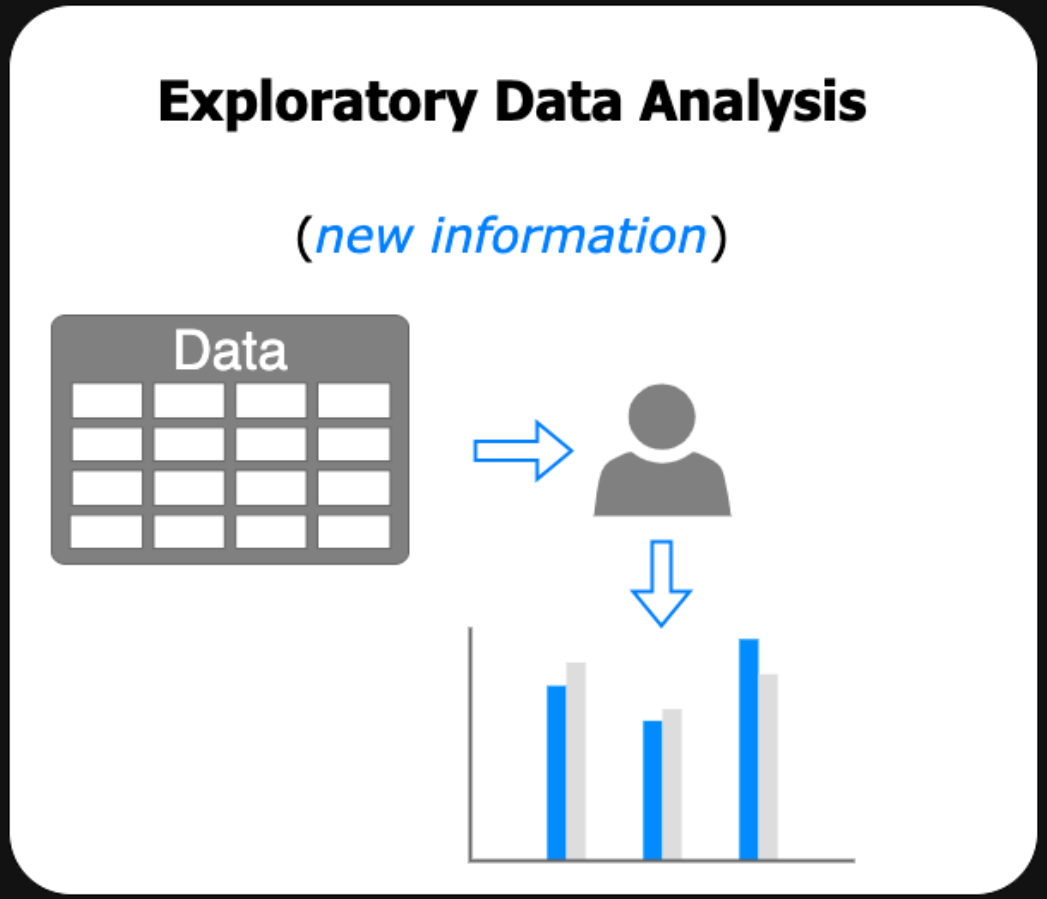
Before EDA, we start with expectations and/or assumptions about the data



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

A Bayesian Mindset

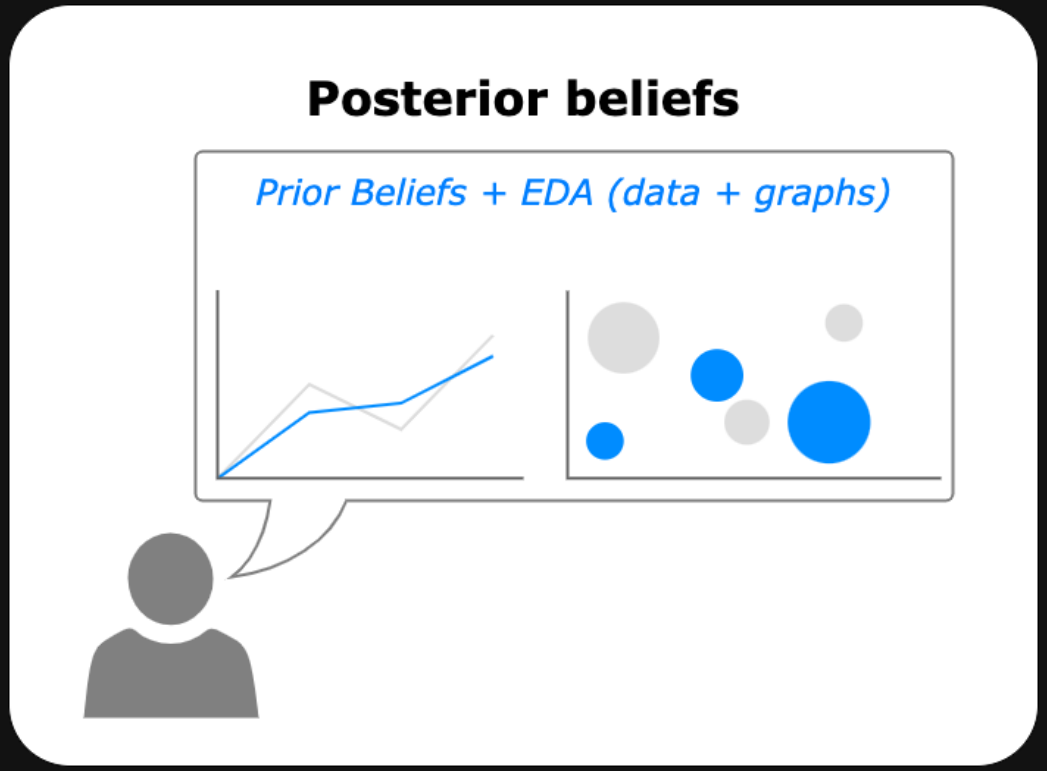
During EDA, we observe new information that either confirms or contradicts our prior beliefs



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

A Bayesian Mindset

After EDA, we have a new set of beliefs which account for the observed data



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

EDA is systematic, technical creativity

The 'exploration' stems from:

1) articulating our prior beliefs,

2) having clear ideas for what we expect to see, and

3) accurately describing our discoveries

A Grammar Of Graphics

ggplot2: grammar & syntax

The Future of AI Gathers Here

Grammar: the system of rules for any given language

Syntax: the form, structure and order for constructing statements

ggplot2: the benefits of grammar & syntax

"objects are like the R language's nouns, and functions (fn) are like verbs"

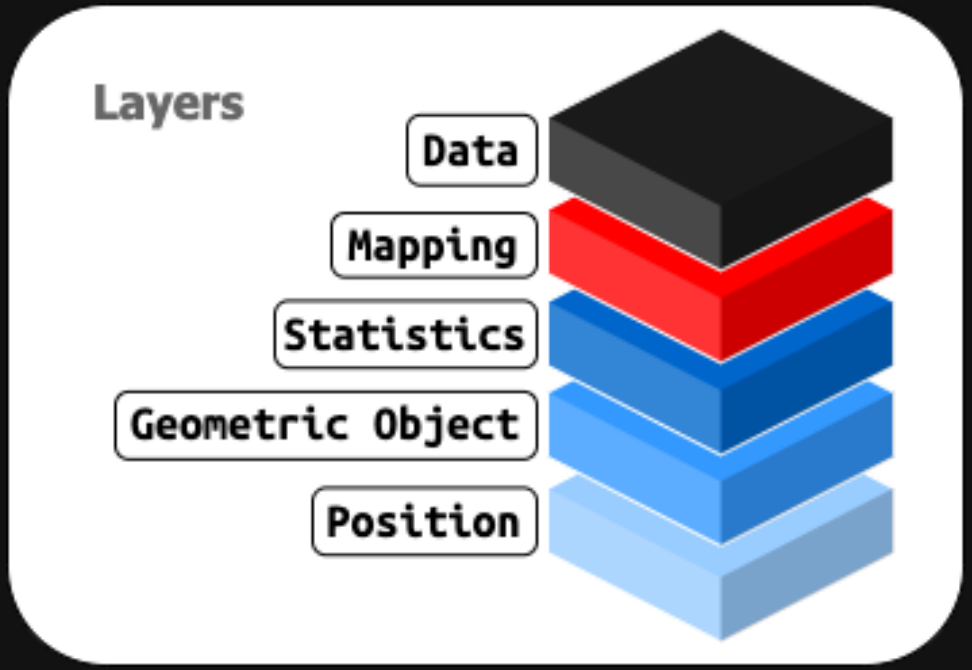
functions do things to objects

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: a layered language for graphs

ggplot2 is comprised of layers

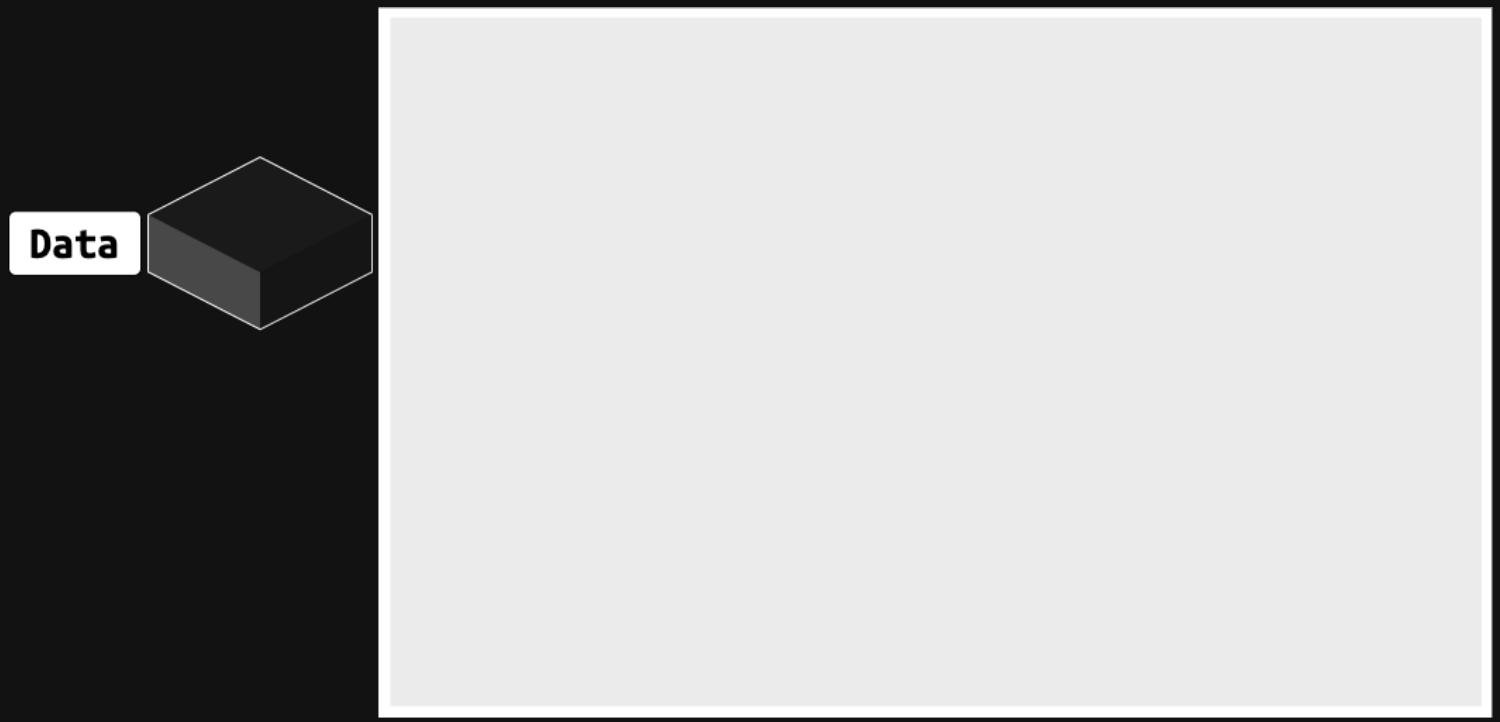
- Data
- Mapping
- Statistics
- Geometric objects
- Position adjustments



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

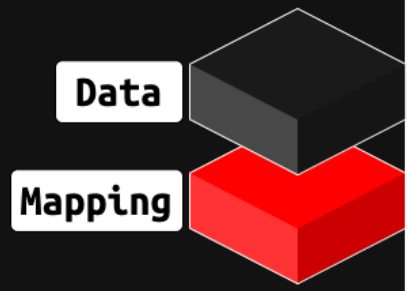
ggplot2: data

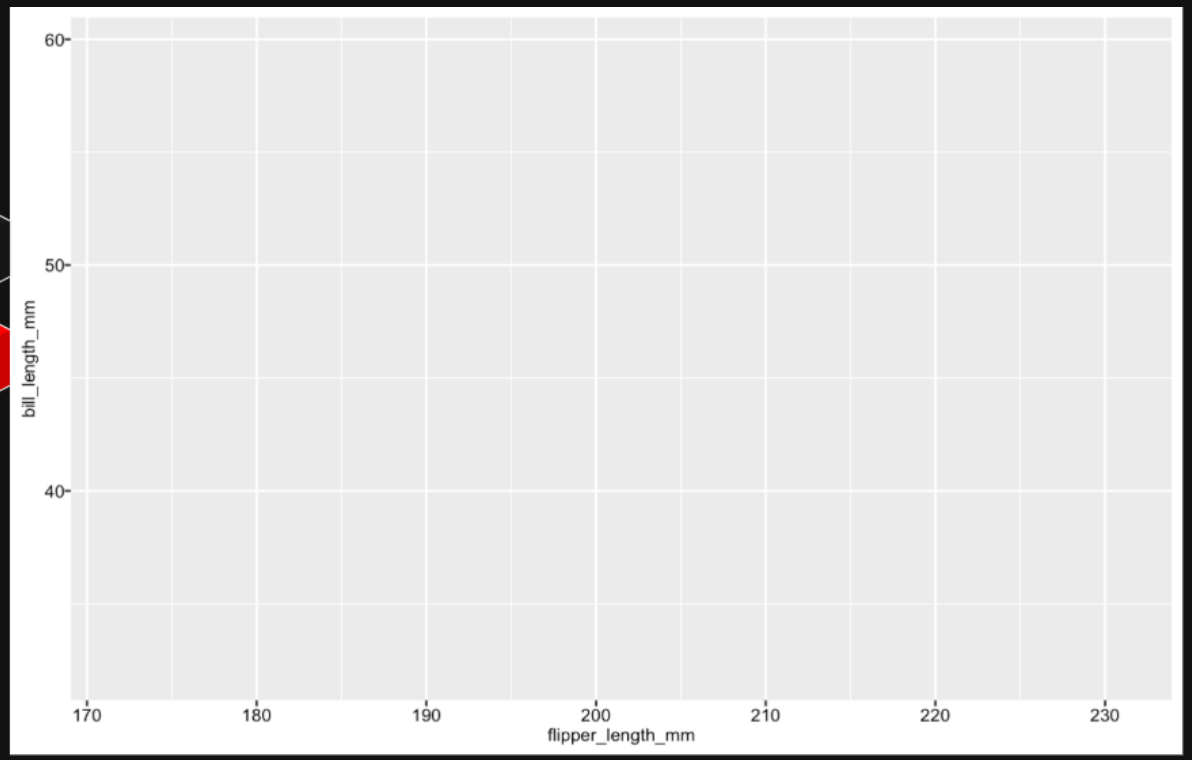
The data layer consists of a rectangular object (like a spreadsheet) with columns and rows



ggplot2: mapping

The mapping layer assigns columns (variables) from the data to a visual property (i.e. graph 'aesthetic')

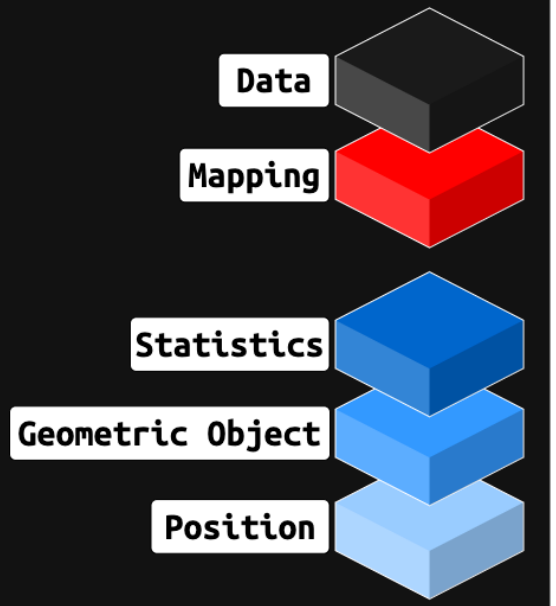


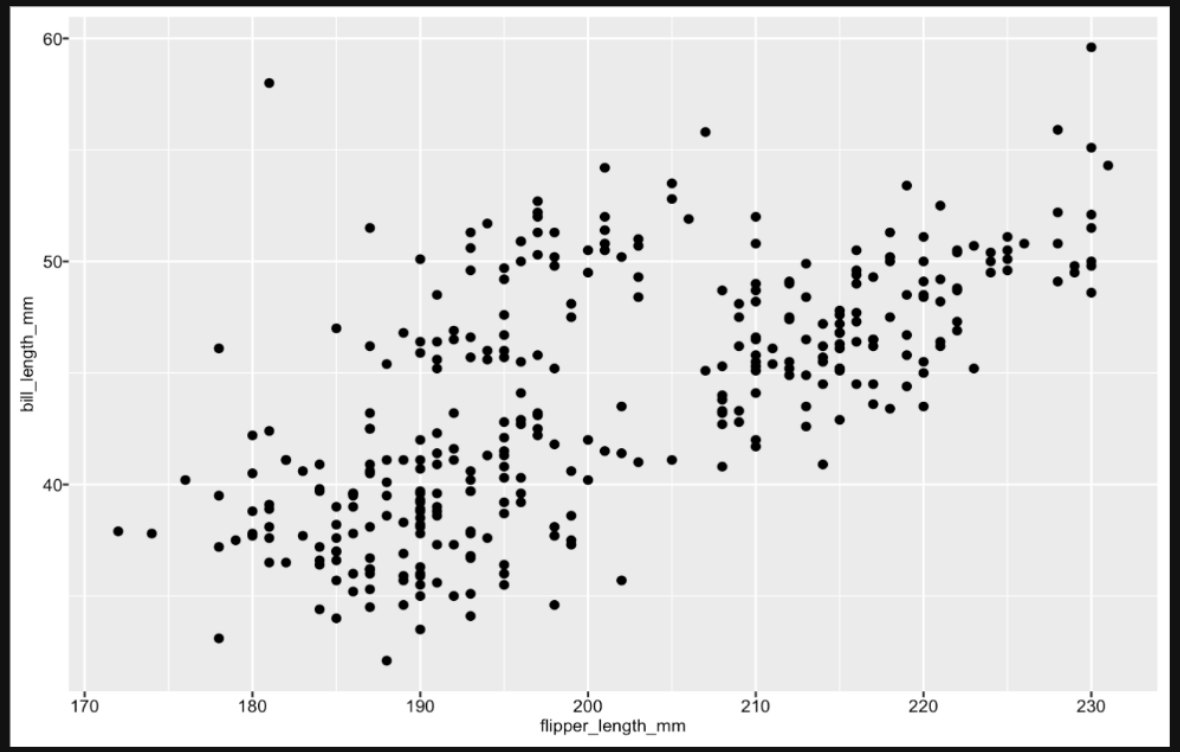


https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: geoms

geom_*() functions include statistical transformations, shapes, and position adjustments for how to 'draw' the data on the graph

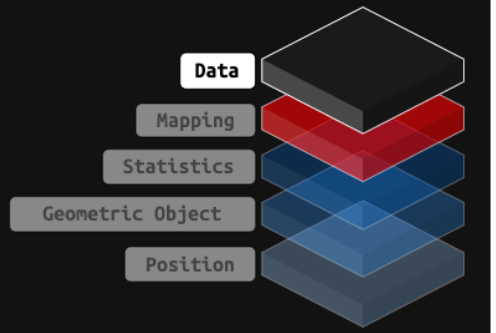


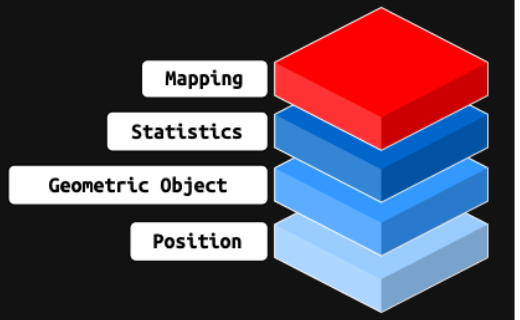


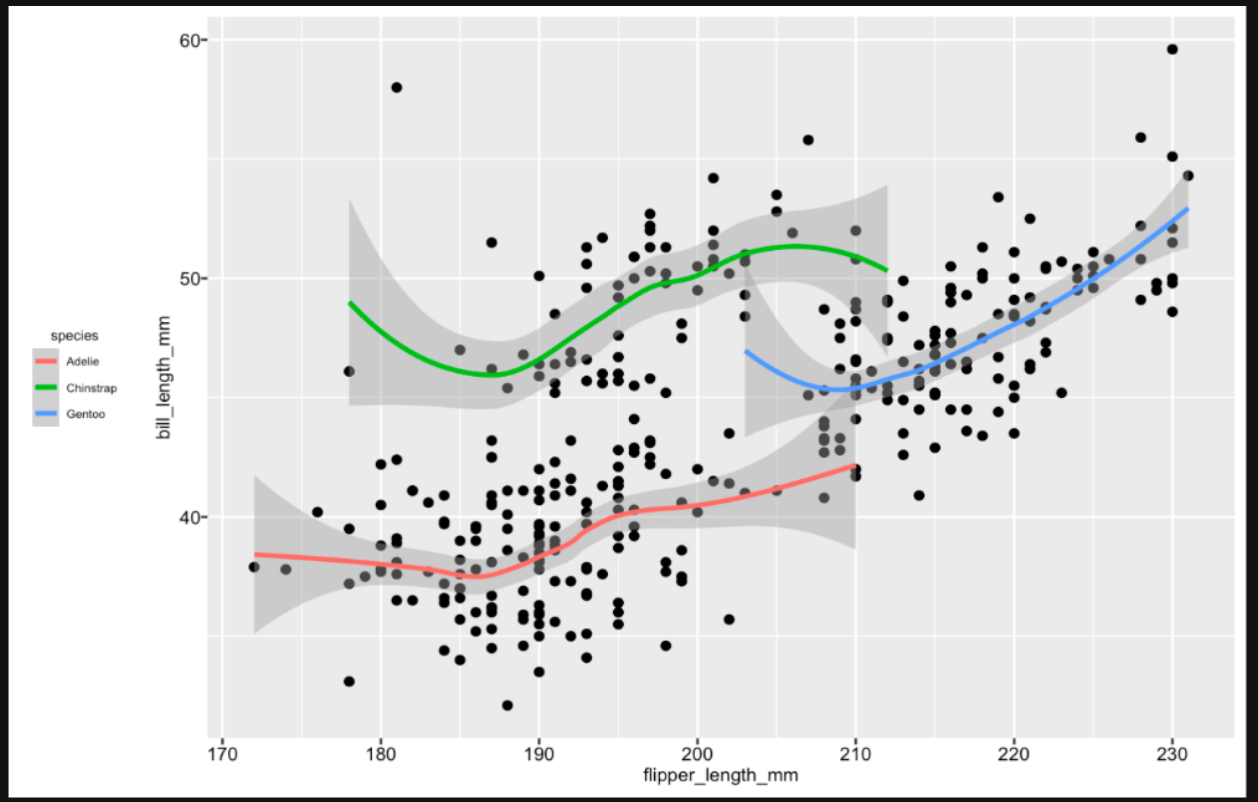
https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: layers

We can have multiple layers (data, mappings, geoms) in a single graph







https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: layers = infinitely extensible

Language is a system for

> “making infinite use of finite means.” - Wilhelm von Humboldt

With a finite number of objects & functions, we can combine ggplot2s grammar and syntax to create an infinite number of graphs!

ggplot2: layers = infinitely extensible

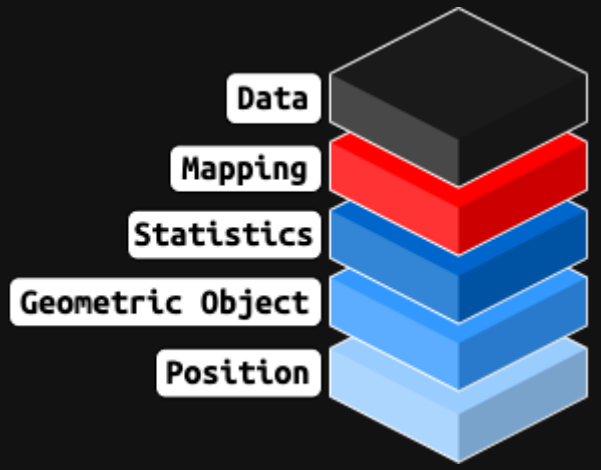
The Future of AI Gathers Here

We can build graphs layer-by-layer

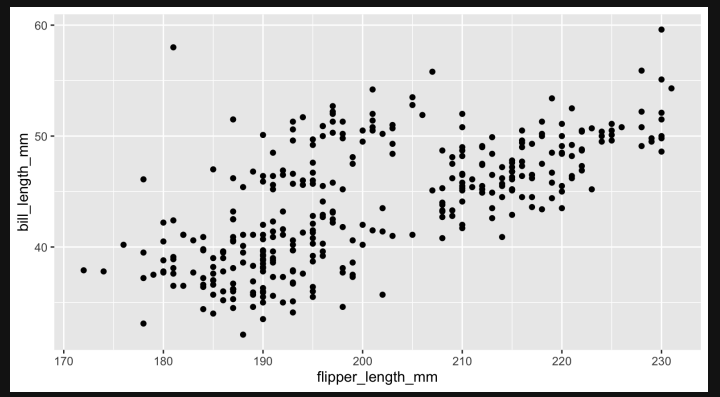
code

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

layer



graph



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

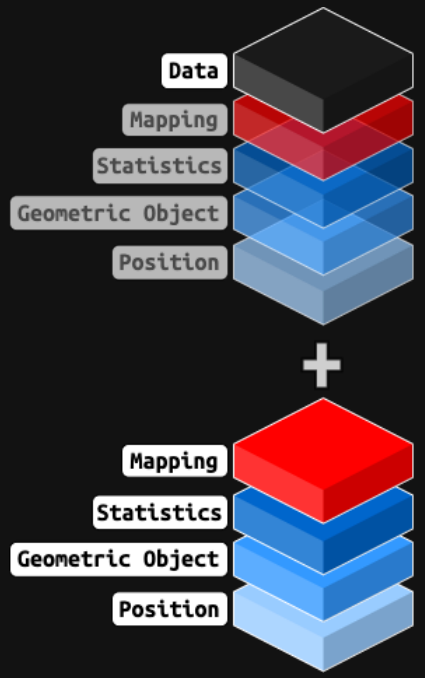
ggplot2: layers = infinitely extensible

New layers can 'inherit' data from previous layers (or include their own data)

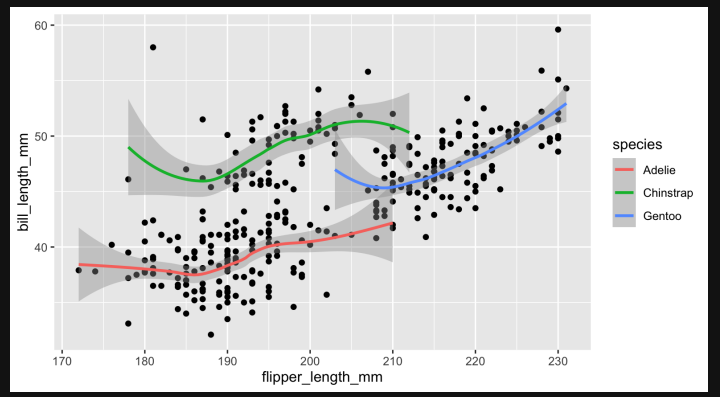
code

```
ggplot(data = penguins,

   mapping = aes(x = flipper_length_mm,
                 y = bill_length_mm)) +

  geom_point() +

  geom_smooth(
    mapping = aes(x = flipper_length_mm,
                  y = bill_length_mm,
                  color = species))
```

layer



graph



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: layers = infinitely extensible

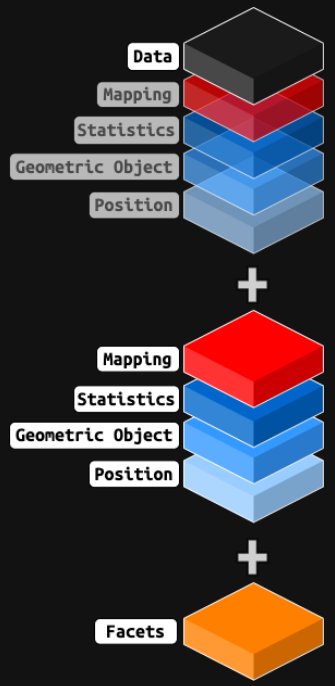
The Future of AI Gathers Here

Additional functions for facets, themes, etc.

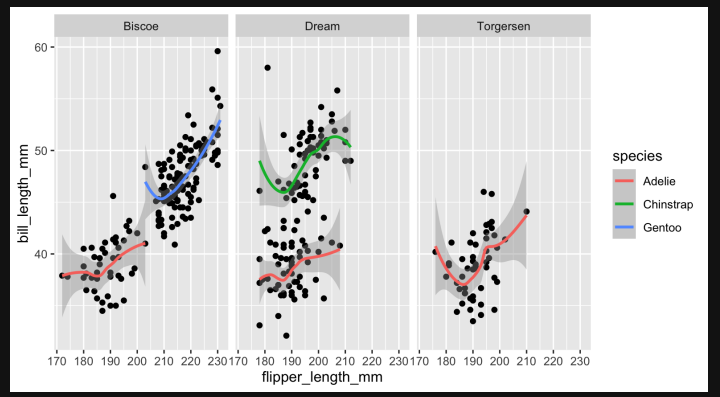
code

```
ggplot(data = penguins,

    mapping = aes(x = flipper_length_mm,
                  y = bill_length_mm)) +
    geom_point() +

    geom_smooth(
    mapping = aes(x = flipper_length_mm,
                  y = bill_length_mm,
                  color = species)) +

    facet_wrap(facets = . ~ island)
```

layer



graph



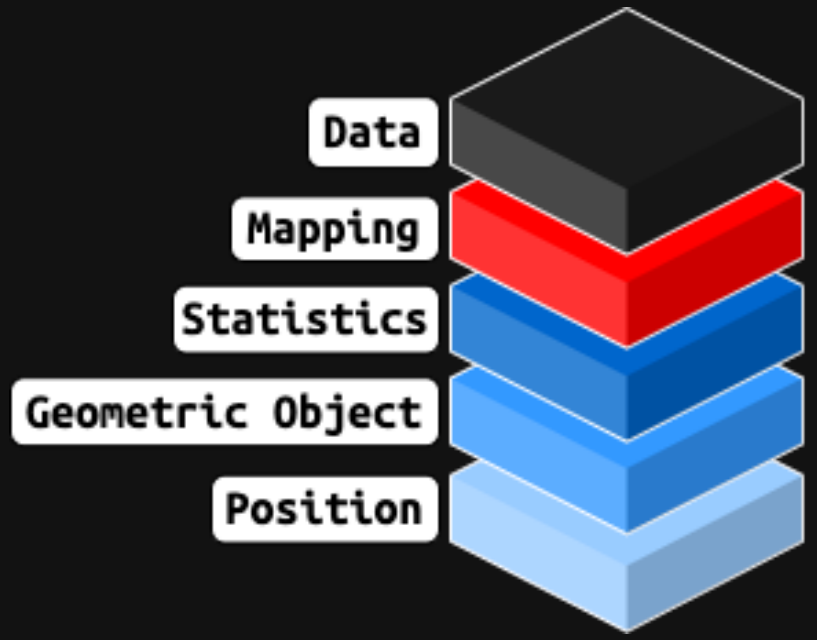
https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: templates

The Future of AI Gathers Here

Basic Template: Data, aesthetic mappings, geom

```
ggplot(data = <DATA>) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>))
```



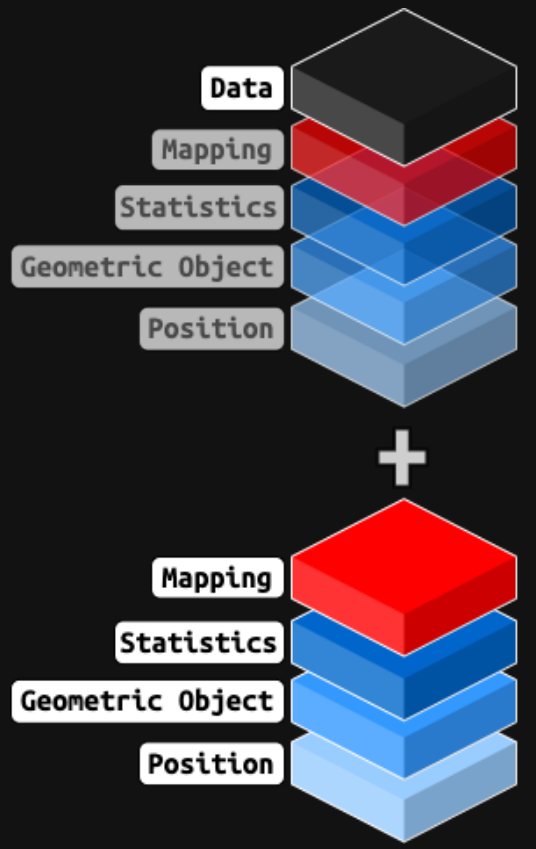
https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: templates

The Future of AI Gathers Here

Template + 1 Layer: More geoms and aesthetic mappings

```
ggplot(data = <DATA>) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>)) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>))
```

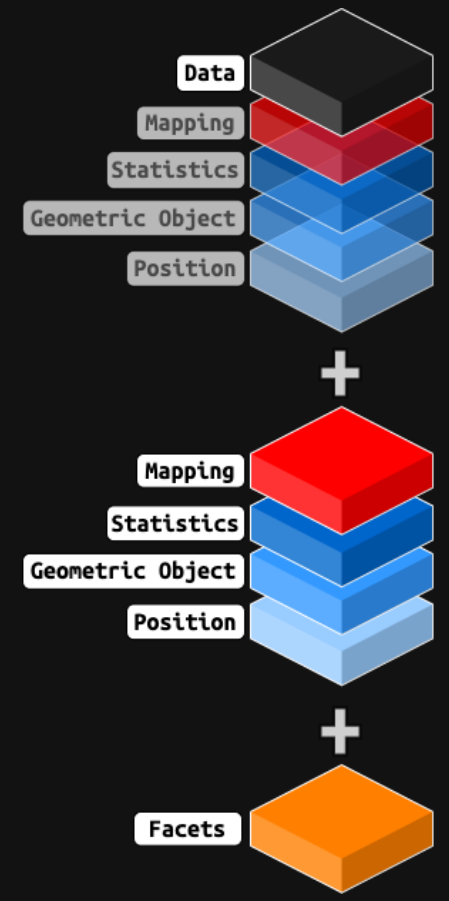


https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: templates

Template + 2 Layers: Faceting

```
ggplot(data = <DATA>) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>)) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>)) +
     facet_*
```



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

templates = infinitely extensible!

The Future of AI Gathers Here

Themes

```
ggplot(data = <DATA>) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>)) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>)) +
     facet_* +
     theme_*
```

Don't forget labels!

```
ggplot(data = <DATA>) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>)) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>)) +
     facet_* +
     theme_* +
     <LABELS>
```


https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Next up: Part 2!

@mjfrigaard
@mjfrigaard
mjfrigaard@pm.e
What does "λέξις" mean?

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ODSC: Data Visualization with ggplot2

Part 2: Creating graphs with ggplot2

by Martin Frigaard

Written: February 08 2022

Updated: March 13 2022

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Resources

Links:

- Conference Website
- Website
- Part 1
- Part 2

Materials:

- RStudio.Cloud
- Github Repo

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Outline

The Future of AI Gathers Here

Part 1

Exploratory data analysis

- *What is it, who does it, and why it's important*

A Bayesian mindset

- *Priors → new information → posteriors*

The grammar of graphics

- *Layers, aesthetics, and geoms*

Part 2

Build labels first

- *Set expectations*

Exercises & solutions

- *RStudio.Cloud*

Creating graphs

- *Building graphs layer-by-layer, global vs. local mapping, visual encodings*

Applying the grammar

- *Mapping vs. setting aesthetics, combining layers, facets*

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

PART 2

Tip: writing code can be frustrating, especially in the beginning...

...it doesn't always produce a tangible result...

...but creating visualizations is rewarding!!!

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: before we start

Build the labels first!

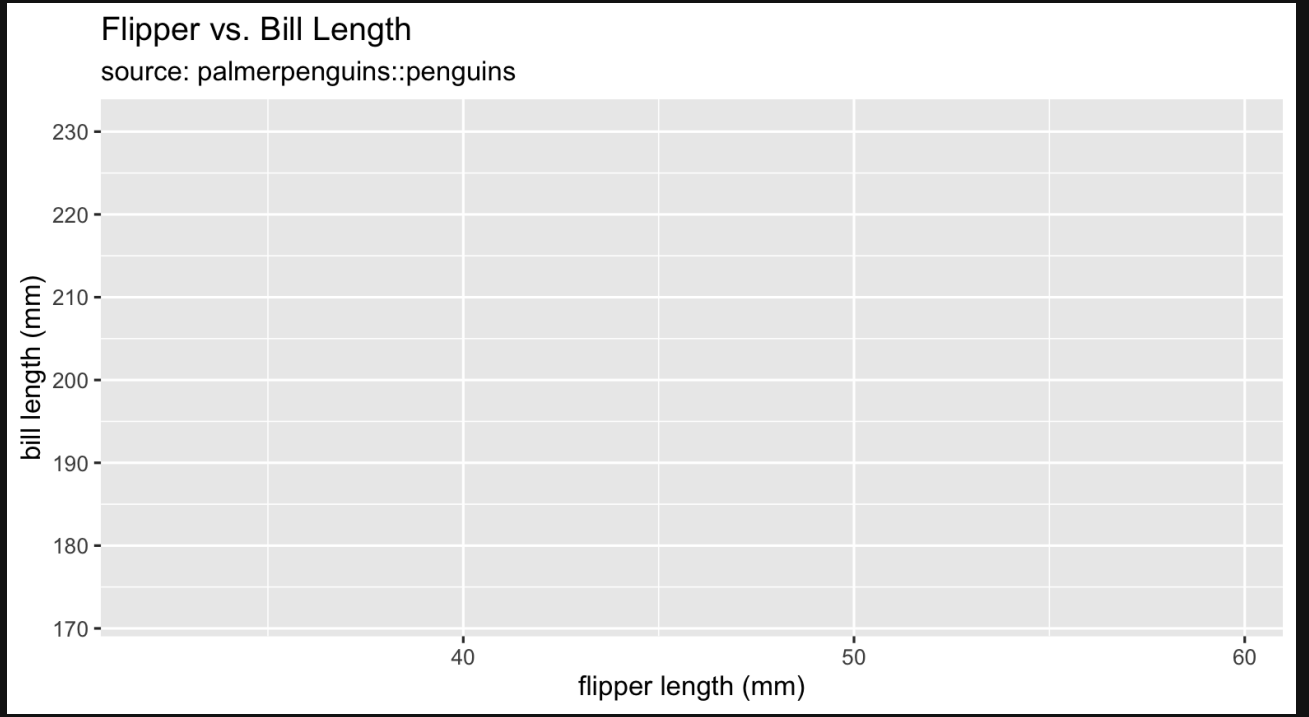
- Create a title, subtitle (with data source), and x/y axis labels

```
labs_pengiuns <- ggplot2::labs(
              title = "Flipper vs. Bill Length",
              subtitle = "source: palmerpenguins::penguins",
              x = "flipper length (mm)",
              y = "bill length (mm)")
```

← our expectations

ggplot2: build graph, check labels

```
labs_pengiuns <- ggplot2::labs(
    title = "Flipper vs. Bill Length",
    subtitle = "source: palmerpenguins::penguins",
    x = "flipper length (mm)",
    y = "bill length (mm)")

ggplot(data = penguins,
       mapping = aes(x = bill_length_mm,
                     y = flipper_length_mm)
       ) +
  labs_pengiuns
```

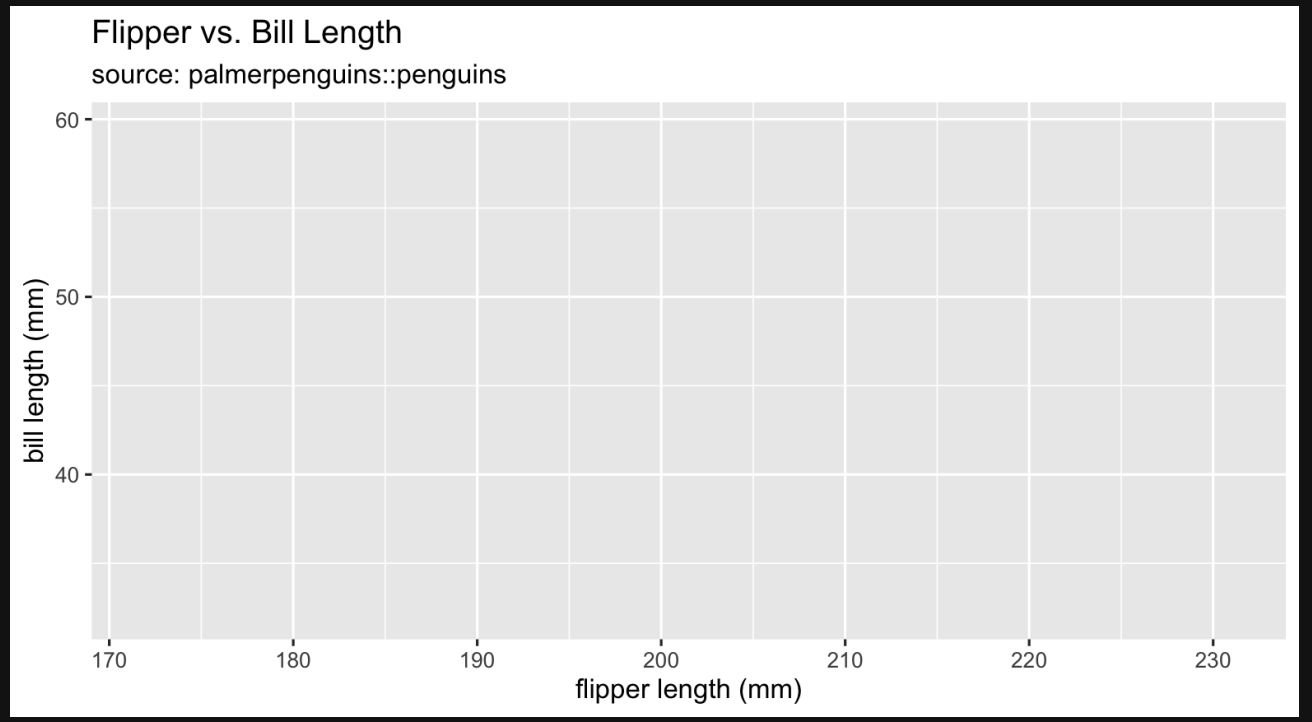


What's wrong here?

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: build graph, check labels, revise

```
labs_pengiuns <- ggplot2::labs(
    title = "Flipper vs. Bill Length",
    subtitle = "source: palmerpenguins::penguins",
    x = "flipper length (mm)",
    y = "bill length (mm)")

ggplot(data = penguins,
       mapping = aes(x = flipper_length_mm,
                     y = bill_length_mm)
       ) +
  labs_pengiuns
```



FIXED!!!

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ggplot2: build graph, check labels, REVISE

The Future of AI Gathers Here

Revision Sharpens Thinking: More particularly, rewriting is the key to improved thinking.

It demands a real open-mindedness and objectivity. It demands a willingness to cull verbiage so that ideas stand out clearly. And it demands a willingness to meet logical contradictions head on and trace them to the premises that have created them.

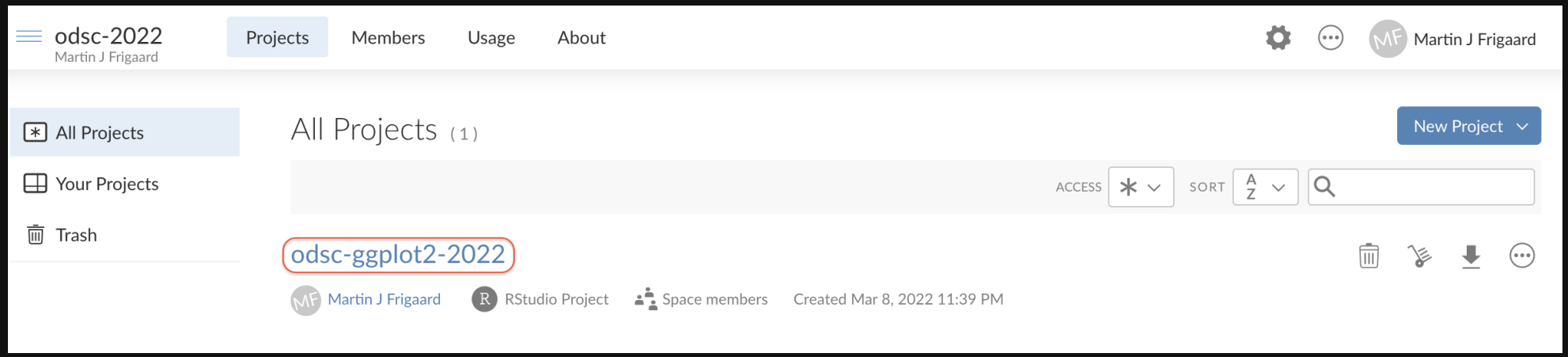
In short, it forces a writer to get up his courage and expose his thinking process to his own intelligence. — Marvin H. Swift, HBR Clear Writing Means Clear Thinking Means...

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Exercises & Solutions

RStudio.Cloud: Set up

In your RStudio.Cloud, you will see the following:

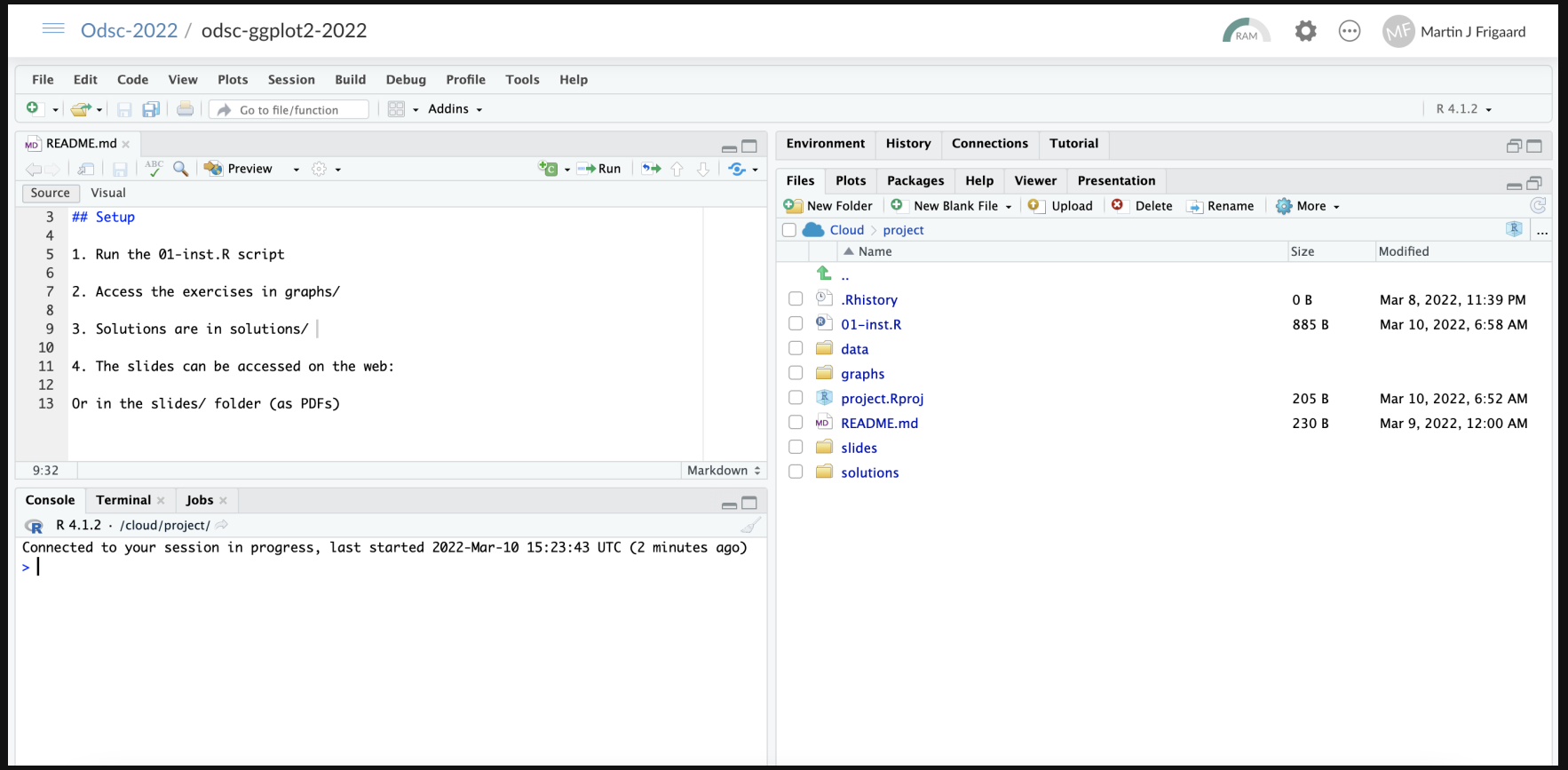


Click on odsc-ggplot2-2022 project

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

RStudio.Cloud: Set up

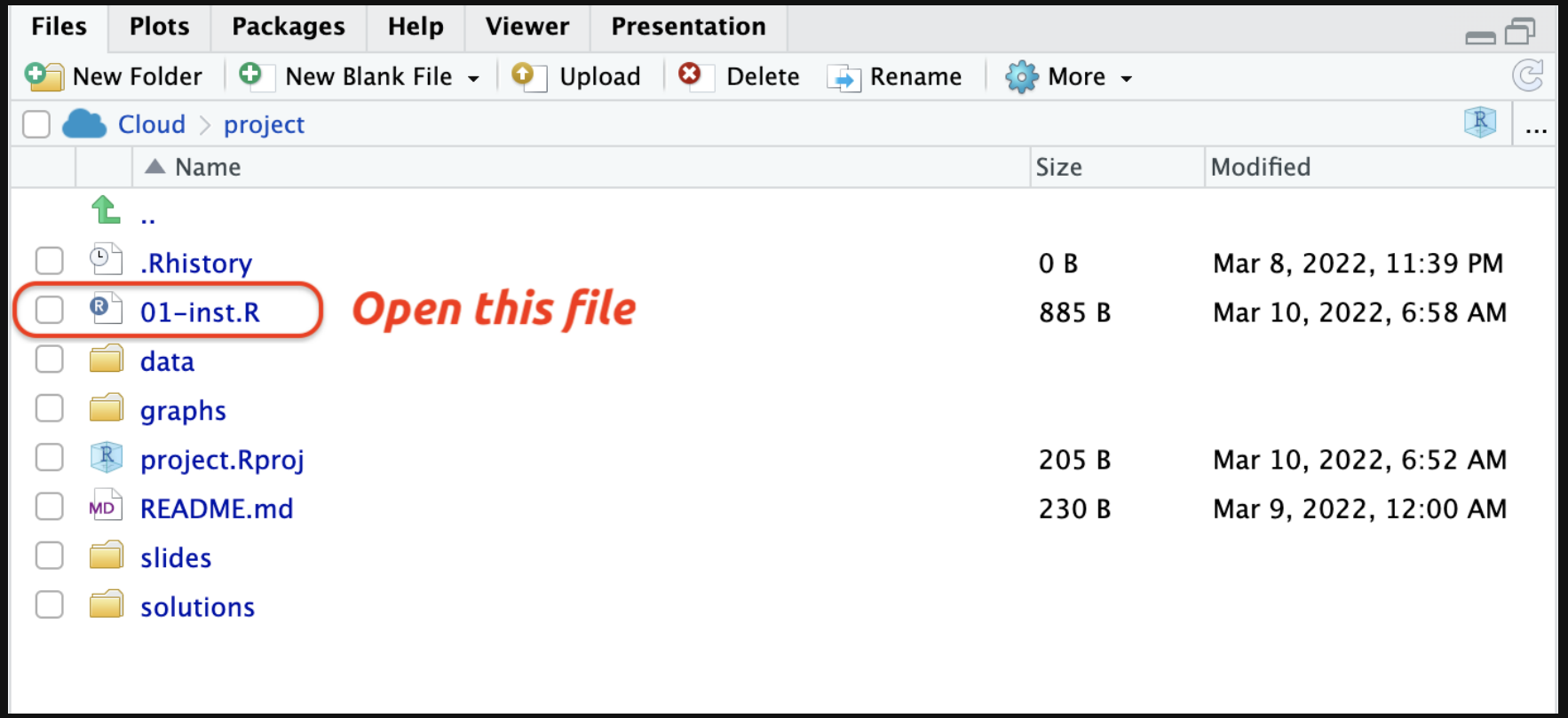
In your RStudio IDE, you will see the following:



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

RStudio.Cloud: Set up

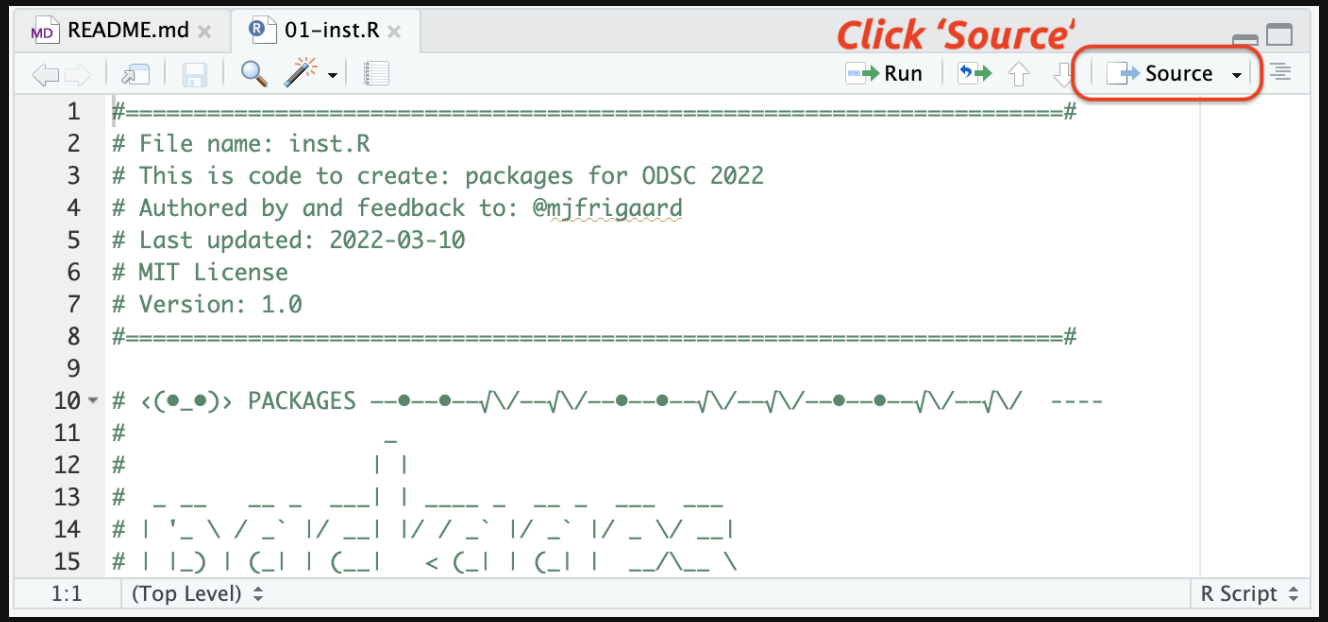
In the Files pane, click on the `inst.R` file to open it



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

RStudio.Cloud: Set up

In the Source pane, click on the *Source* icon to run `inst.R`

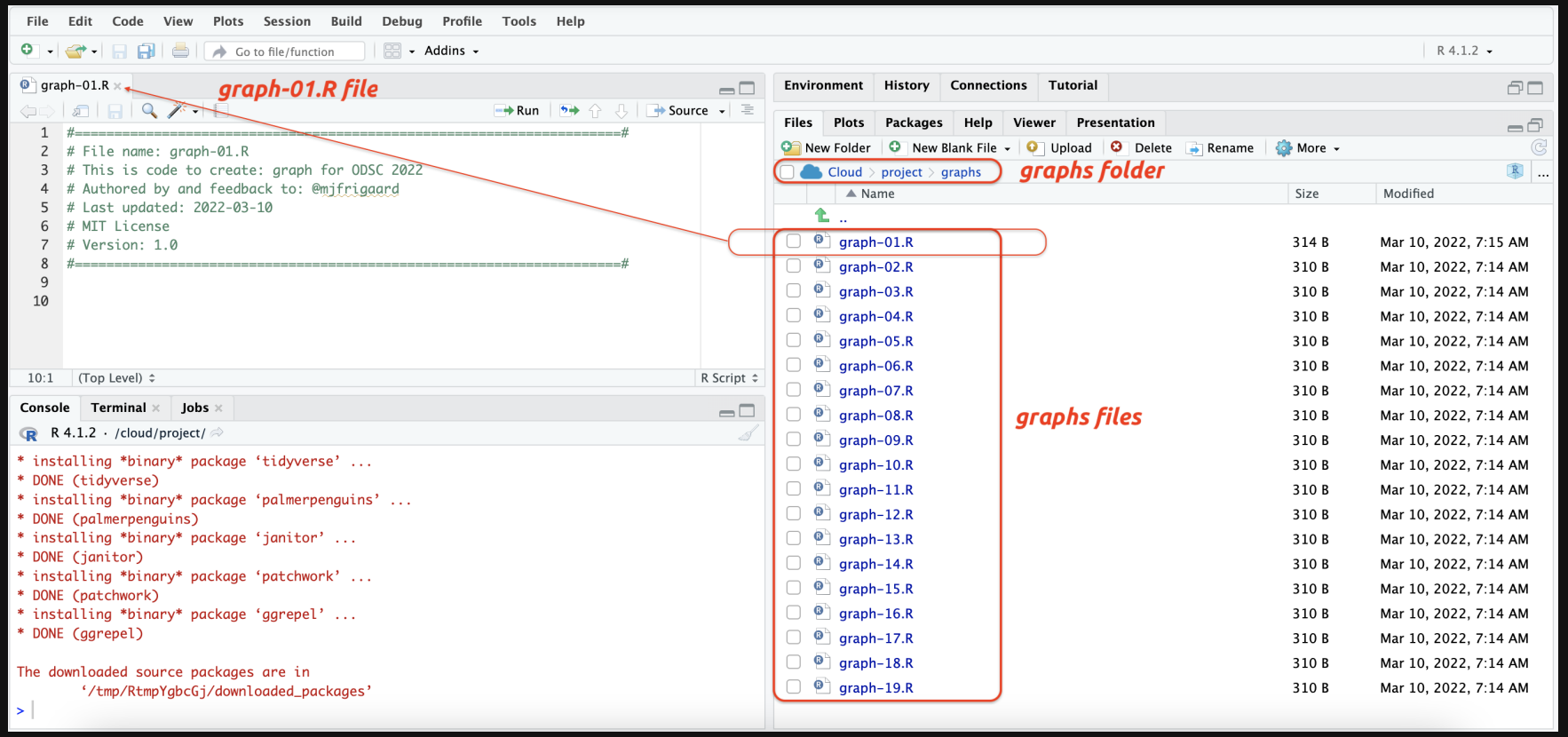


This sends the code in the Source pane to the Console

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

RStudio.Cloud: Exercises

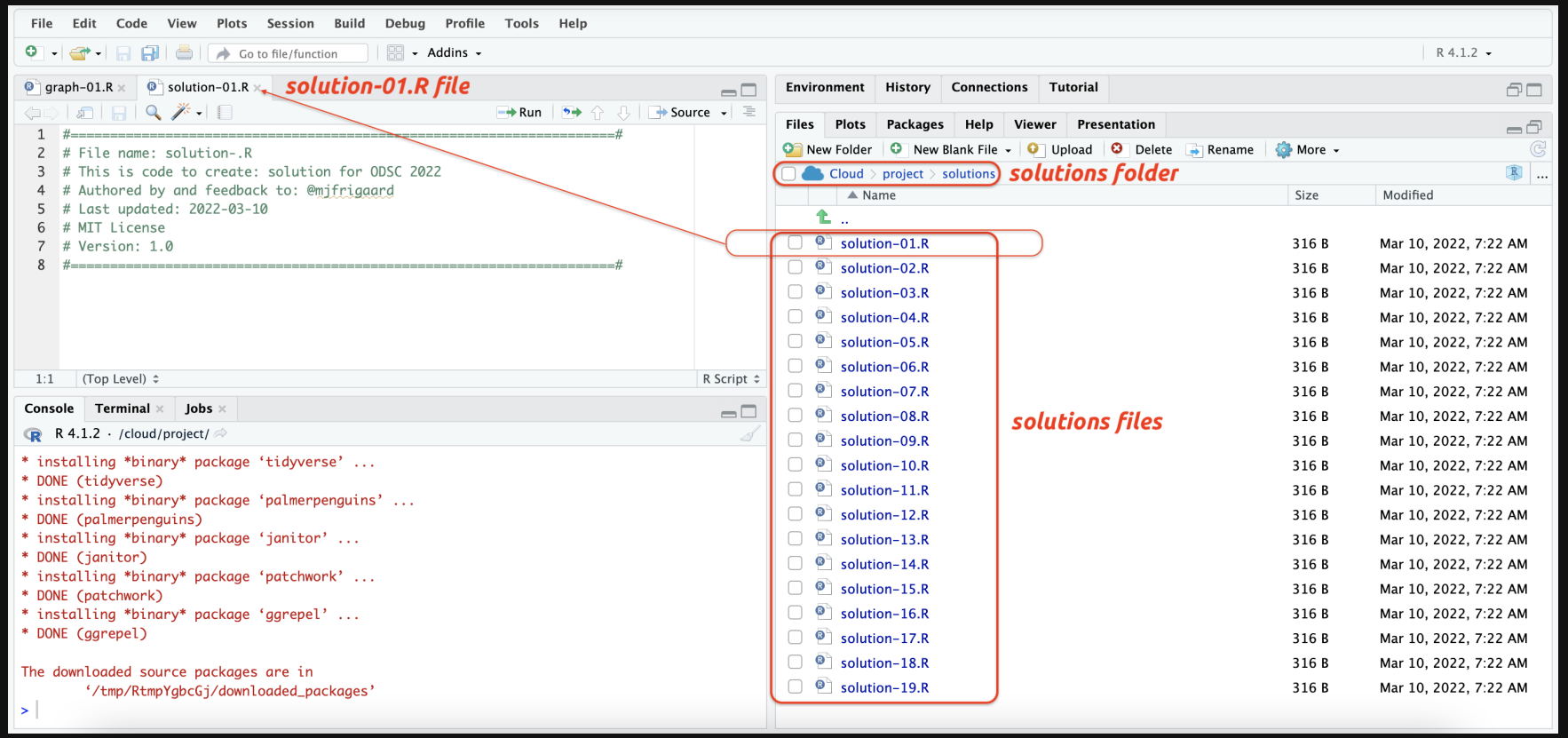
The exercises are in the `graphs/` folder



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

RStudio.Cloud: Solutions

Each exercise has a solution file in `solutions/` folder



The data

We're going to use the `palmerpenguins::penguins`

- Below are three options for viewing a dataset in RStudio:

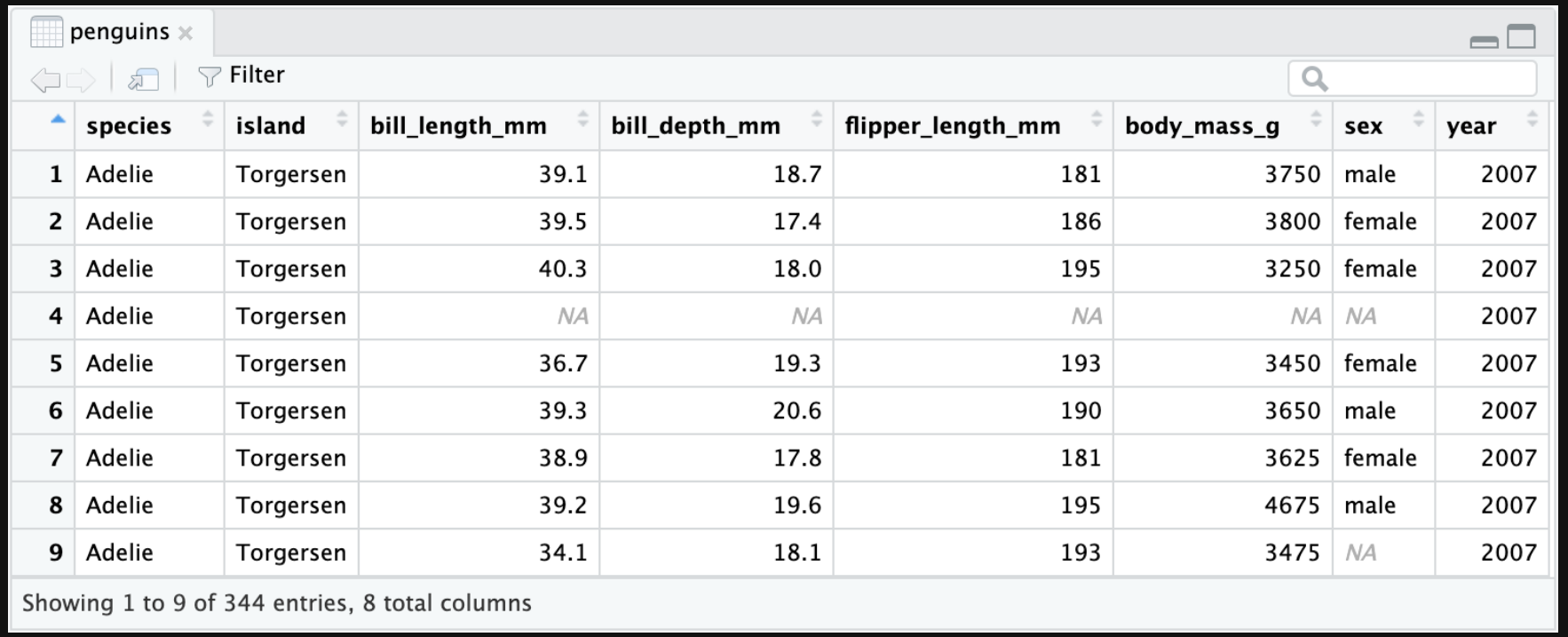
```
View(penguins)
```

```
glimpse(penguins)
```

```
str(penguins)
```

Viewing data

`View()` opens the RStudio data viewer



penguins

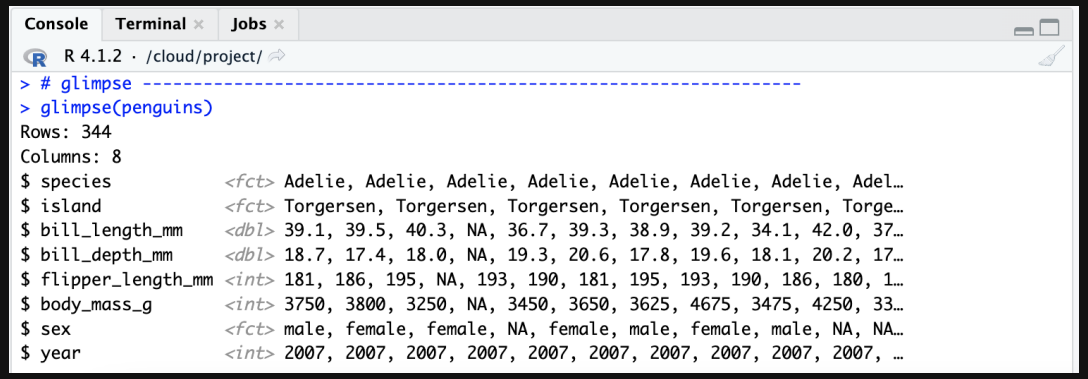
Filter

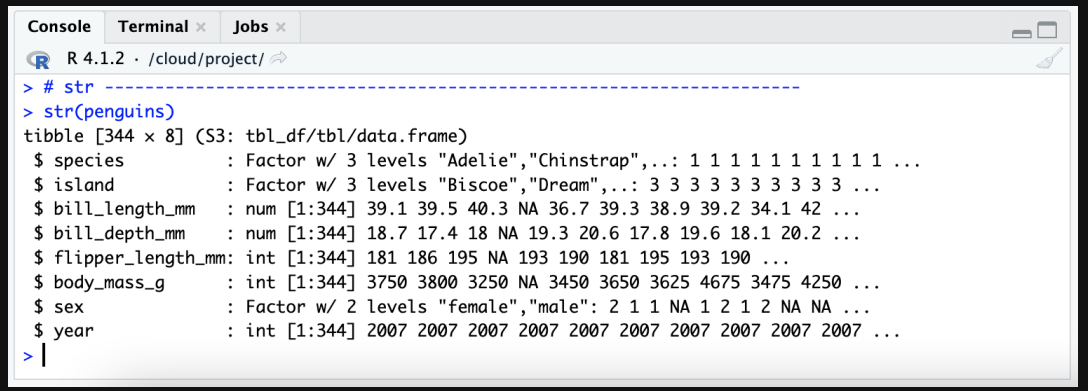
	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	year
1	Adelie	Torgersen	39.1	18.7	181	3750	male	2007
2	Adelie	Torgersen	39.5	17.4	186	3800	female	2007
3	Adelie	Torgersen	40.3	18.0	195	3250	female	2007
4	Adelie	Torgersen	*NA*	*NA*	*NA*	*NA*	*NA*	2007
5	Adelie	Torgersen	36.7	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

Showing 1 to 9 of 344 entries, 8 total columns

Viewing data

`glimpse()` and `str()` are displayed in the console





https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Build from scratch, layer-by-layer

graph 01 Step 0 = LABELS!

We want to build the labels **first**:

- title = "Bill and flipper length of Palmer penguins"
- subtitle = "Size measurements for adult foraging penguins"
- x = "Bill length (mm)"
- y = "Flipper length (mm)"

```
# build labels
labs_bill_vs_flippper <- ggplot2::labs(
  title = "Bill and flipper length of Palmer penguins",
  subtitle = "Size measurements for adult foraging penguins",
  x = "Bill length (mm)",
  y = "Flipper length (mm)")
```

graph 01 Step 1: Initialize plot with data

The `ggplot2::ggplot()` function initiates the plot

```
ggplot(data = )
```

Place `penguins` in the data argument

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

This gives us a blank canvas!

graph 02 Step 2: Map variables to positions

We have our data and labels--we just need to add our variables

Map `bill_length_mm` to `x`

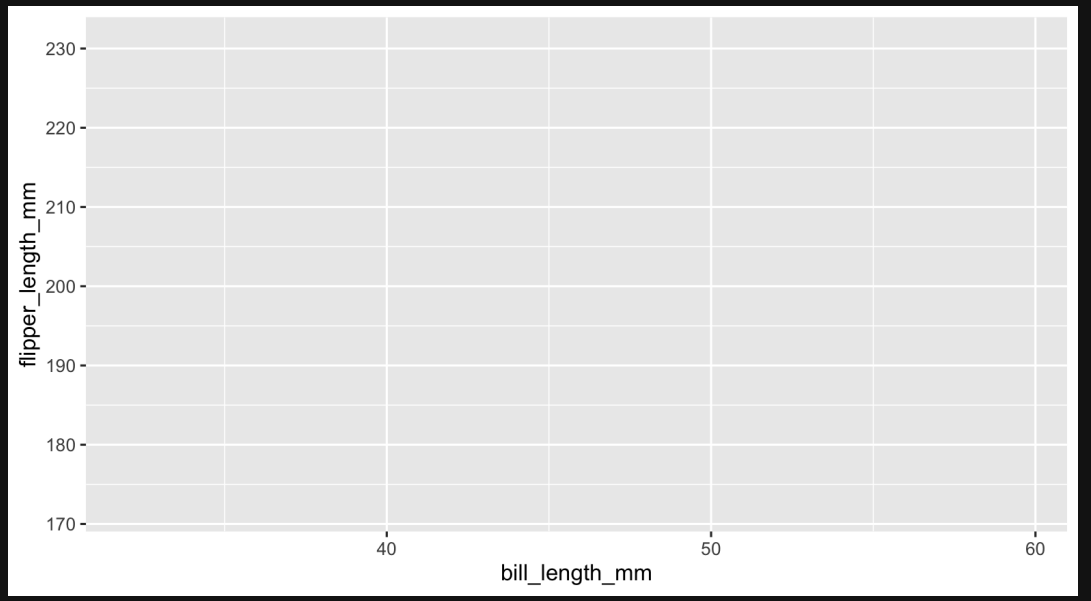
```
ggplot(data = penguins,
       mapping = aes(x = bill_length_mm,
                     ))
```

Map `flipper_length_mm` to `y`

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

Now we have our variables on our graph!

Now our canvas has `x` and `y` axes



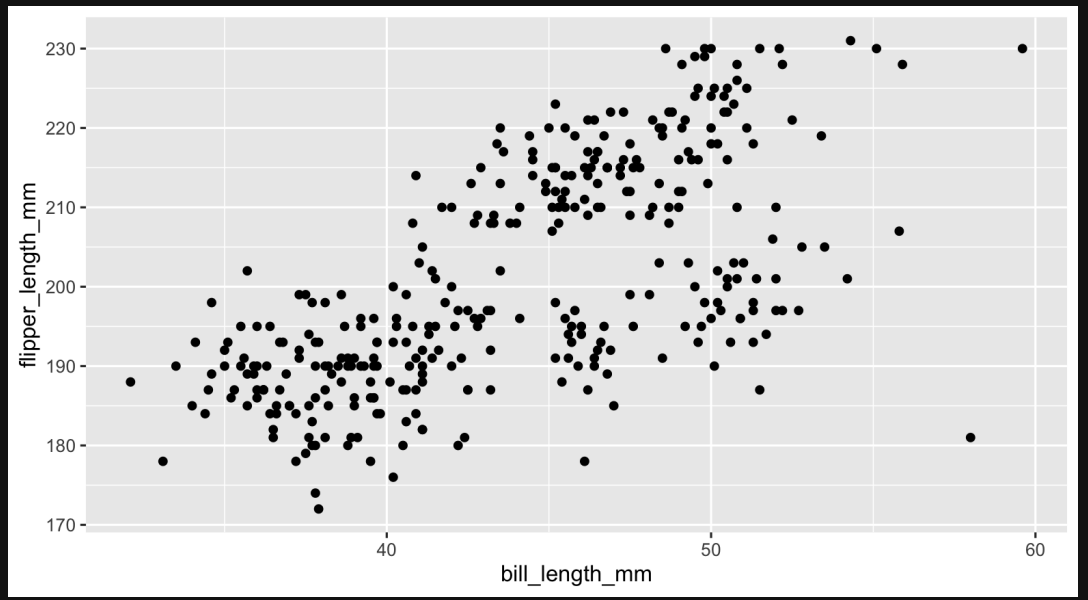
https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

graph 03 Step 3: Adding geoms

Add the `geom_point()` function with the `+` symbol

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

The `geom_point()` function tells R we want to see the points (or dots) on our canvas



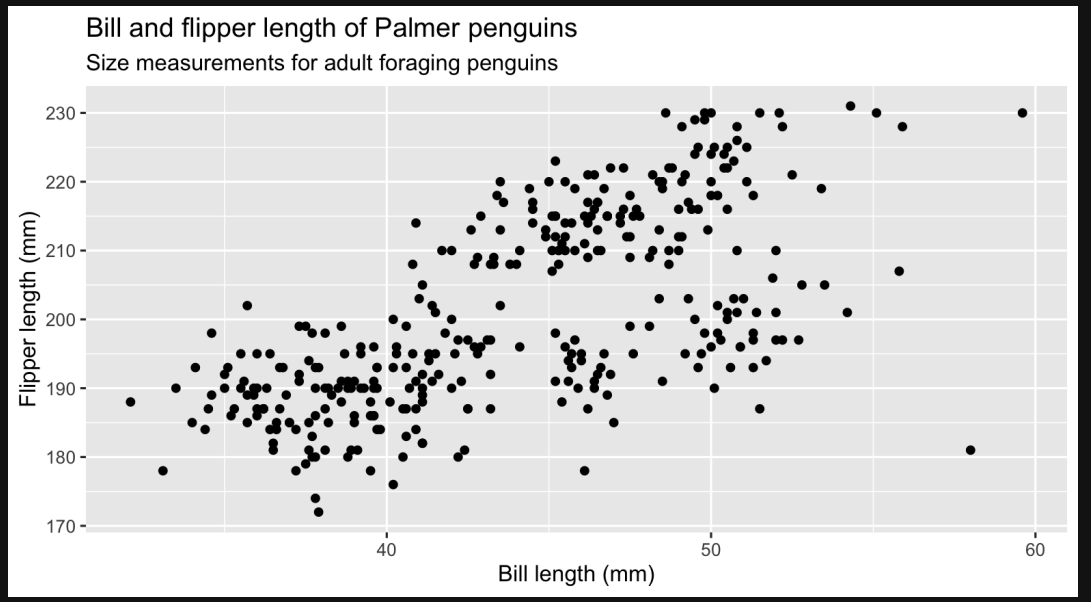
https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

graph 04 Step 4: Don't forget the labels!

Finally, we want to add the labels we created (`labs_bill_vs_flippper`)

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

And we have our first graph!



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Global vs. local mapping

Global vs. local mapping

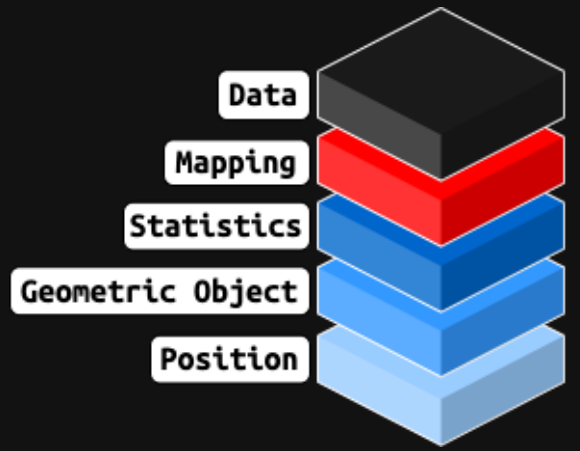
We've just created a graph by setting global aesthetics

Global means aesthetic mappings are set when the graph is initialized with the `ggplot()` function

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

If we map aesthetics `ggplot()`, all the following `geom_*()` layers will inherit these aesthetics.

Recall what goes into each layer from Part 1



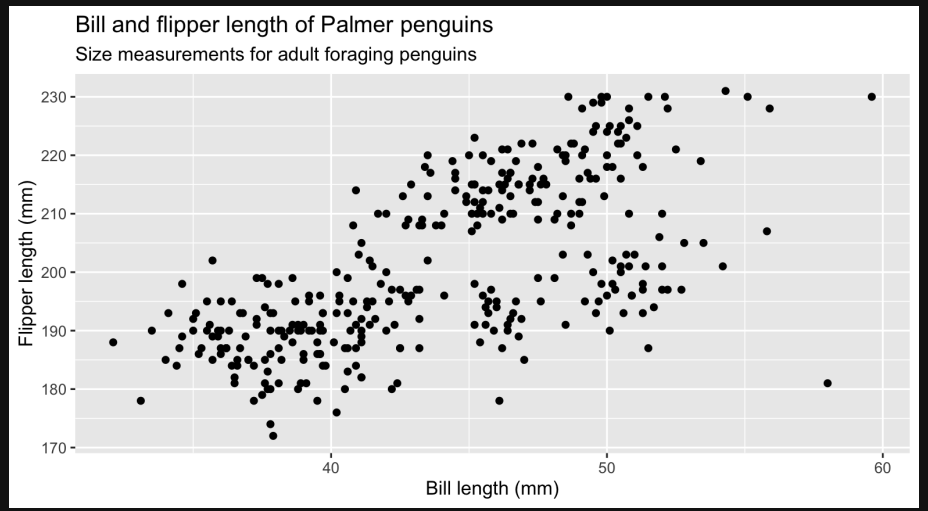
https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

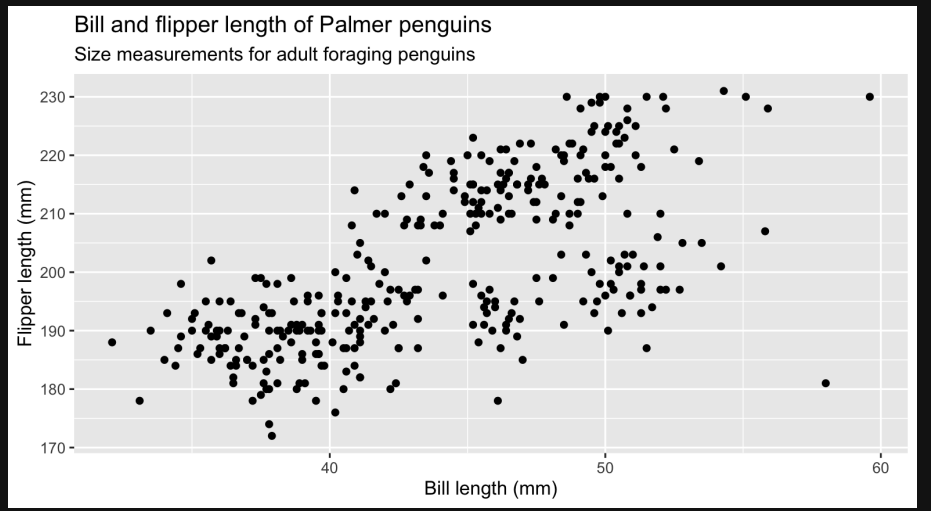
Global vs. local mapping

Mapping aesthetics globally and then adding the `geom_*()` function results in the same graph as when we map aesthetics locally (*inside* the `geom_*()` function)

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

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





https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Our ggplot2 templates

The template from part 1 uses local mappings (i.e. aesthetic mappings are set *inside* the geom_* function).

```
# Recall our template from Part 1
ggplot(data = <DATA>) +
     geom_*(mapping = aes(<AESTHETIC MAPPINGS>))
```

We could adjust this template to include global mappings (and the option to include aesthetic mappings locally)

```
# Adjusted template
ggplot(data = <DATA>,
       mapping = aes(<AESTHETIC MAPPINGS>)) + # global mappings
       geom_*(mapping = aes(<AESTHETIC MAPPINGS>)) # local mappings
```

Read more here.

graph 05 Convert global to local mappings

For graph 05.R, convert the global aesthetics to local aesthetics inside the `geom_point()` function

Global

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

Local

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

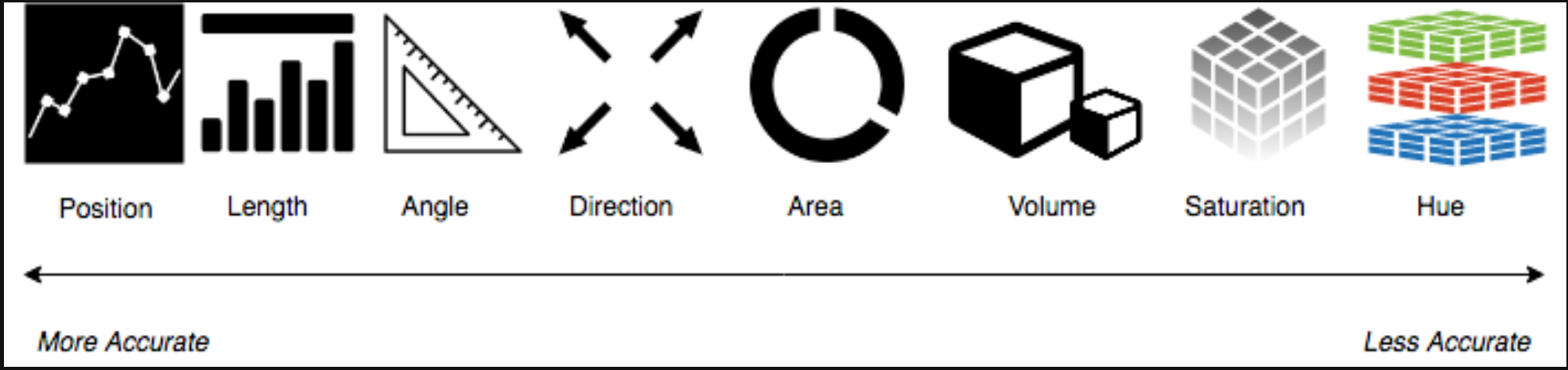
Visual encodings

What are visual encodings?

Visual encodings are what we see on the graph

Things like position, size, shape, color, etc.

Ranked by accuracy ("generally speaking"):

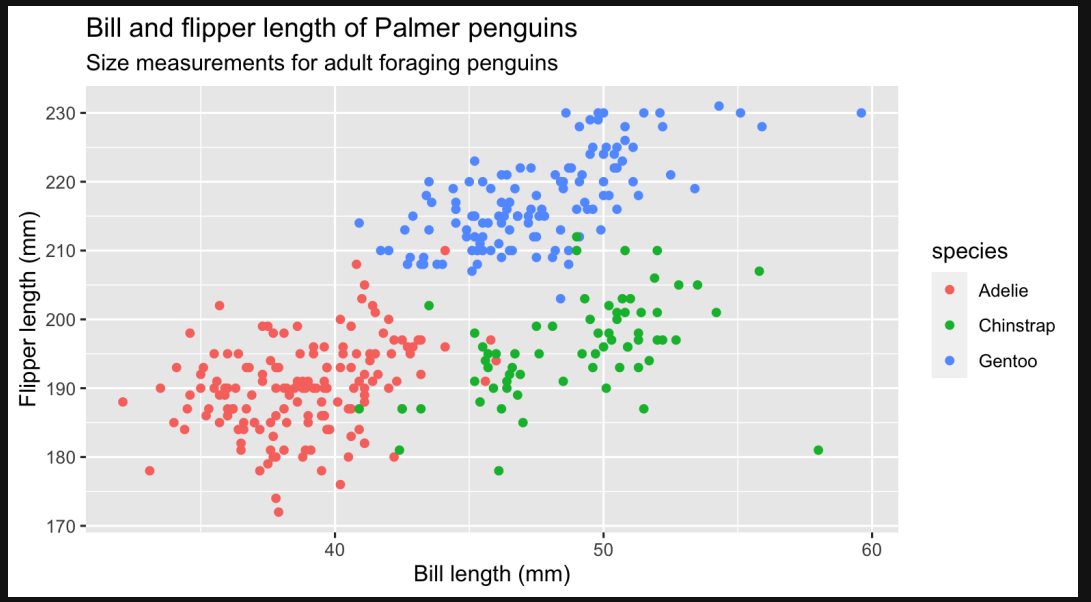


https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

graph 06 Adding color (global)

Map `color` to the `species` variable in the scatter plot using global aesthetic mapping

```
ggplot(data = penguins,
       mapping =
         aes(x = bill_length_mm,
             y = flipper_length_mm,
             color = species)) +
  geom_point() +
  labs_bill_vs_flippper
```

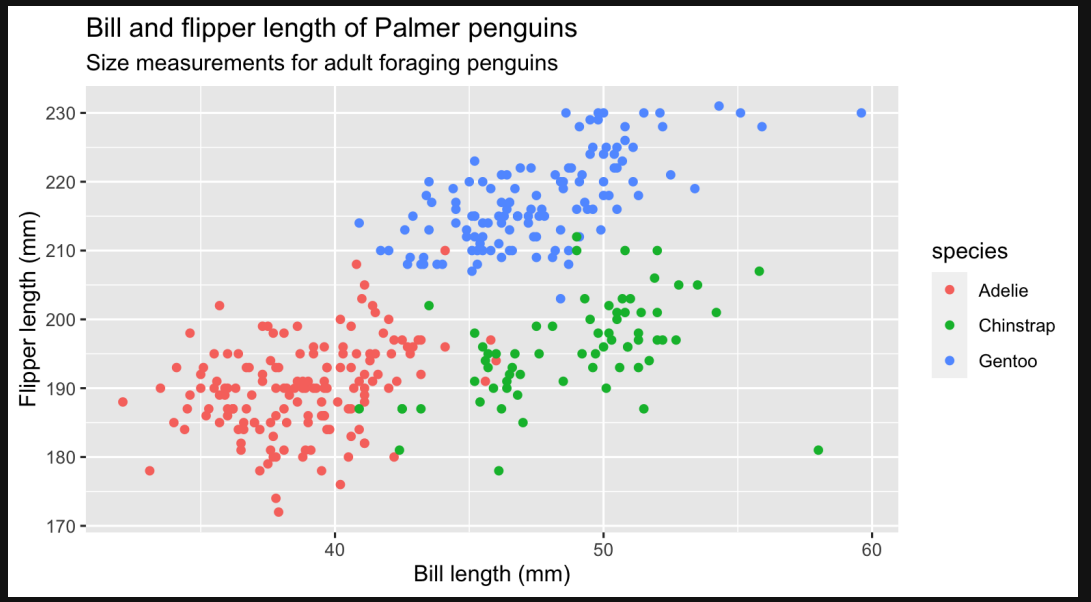


https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

graph 07 Adding color (local)

Map `color` to the `species` variable in the scatter plot using local aesthetic mapping

```
ggplot(data = penguins,
       mapping =
         aes(x = bill_length_mm,
             y = flipper_length_mm)) +
  geom_point(aes(color = species)) +
  labs_bill_vs_flippper
```



The `x` and `y` aesthetics are inherited from the `ggplot()` function, but the `color` aesthetic comes from the `geom_point()` layer

graph 08 Color vs. Fill

Below we'll look at the counts of `sex` vs. `species` of Palmer penguins

Create labels

```
labs_sex_vs_species <- ggplot2::labs(
  title = "Sex by species of Palmer penguins",
  subtitle = "Counts for adult foraging penguins",
  x = "Sex",
  fill = "Species")
```

Create `penguins_no_miss` by removing missing values

```
penguins_no_miss <- drop_na(data = penguins)
```

```
penguins_no_miss
```

species <fct>	island <fct>	bill_length_mm <dbl>
Adelie	Torgersen	39.1
Adelie	Torgersen	39.5
Adelie	Torgersen	40.3
Adelie	Torgersen	36.7
Adelie	Torgersen	39.3
Adelie	Torgersen	38.9
Adelie	Torgersen	39.2
Adelie	Torgersen	41.1
Adelie	Torgersen	38.6
Adelie	Torgersen	34.6

1-10 of 333 row... Previous 1 2 3 4 5 6 ... 34 Next

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

graph 08 Color vs. Fill

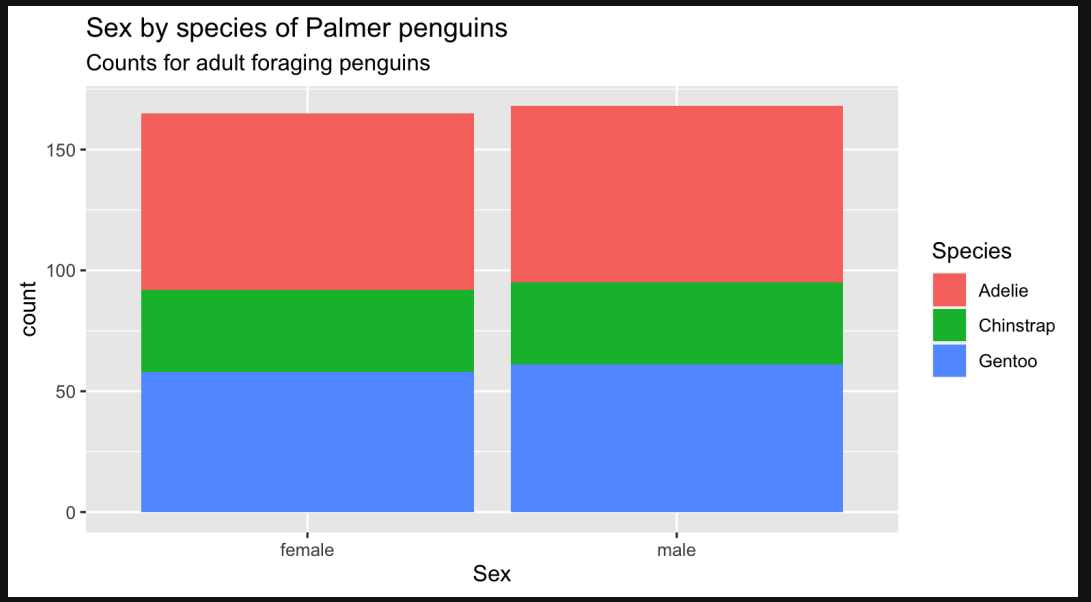
Some `geom_*()` functions take the `fill` argument instead of `color`

Build a bar-graph using `geom_bar()` using local aesthetic mapping

Map `sex` to the `x` axis and `y` to `fill`

```
ggplot(data = penguins_no_miss) +
  geom_bar(mapping = aes(x = sex,
                         fill = species)) +
  labs_sex_vs_species
```

Don't forget the labels!



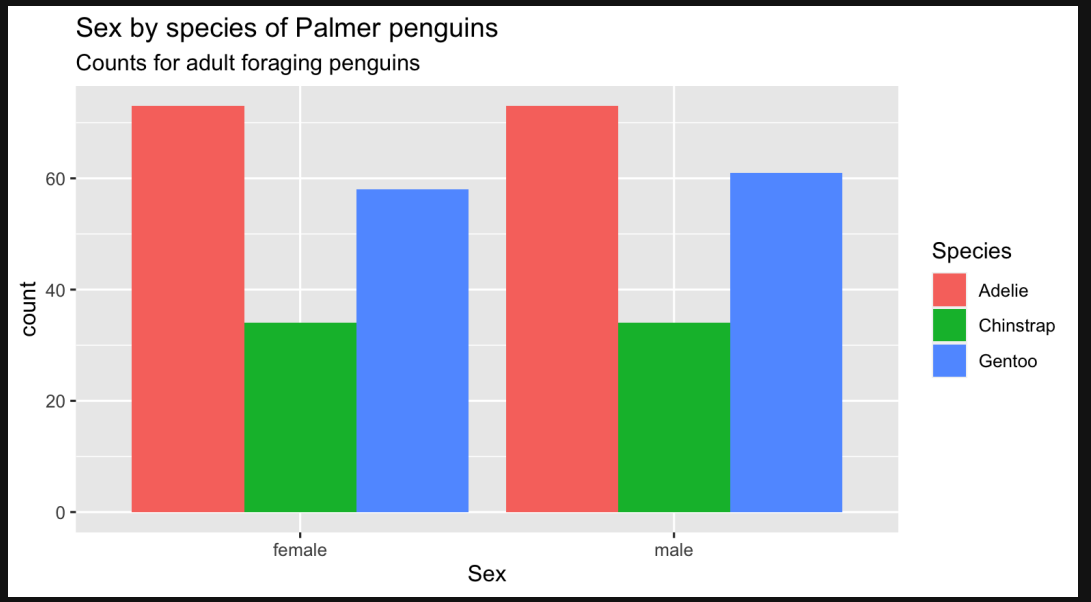
https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

graph 09 Bar position

Stacked bar-graphs make it difficult to do side-by-side comparisons using the y axis

Using the same code as graph 08, add the `position = "dodge"` argument *outside* the `aes()` function

```
ggplot(data = penguins_no_miss) +
  geom_bar(mapping = aes(x = sex,
                         fill = species),
           position = "dodge") +
  labs_sex_vs_species
```



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

graph 10 Histograms (special bar-graphs)

The `geom_histogram()` function uses 'bins' to represent counts for each value

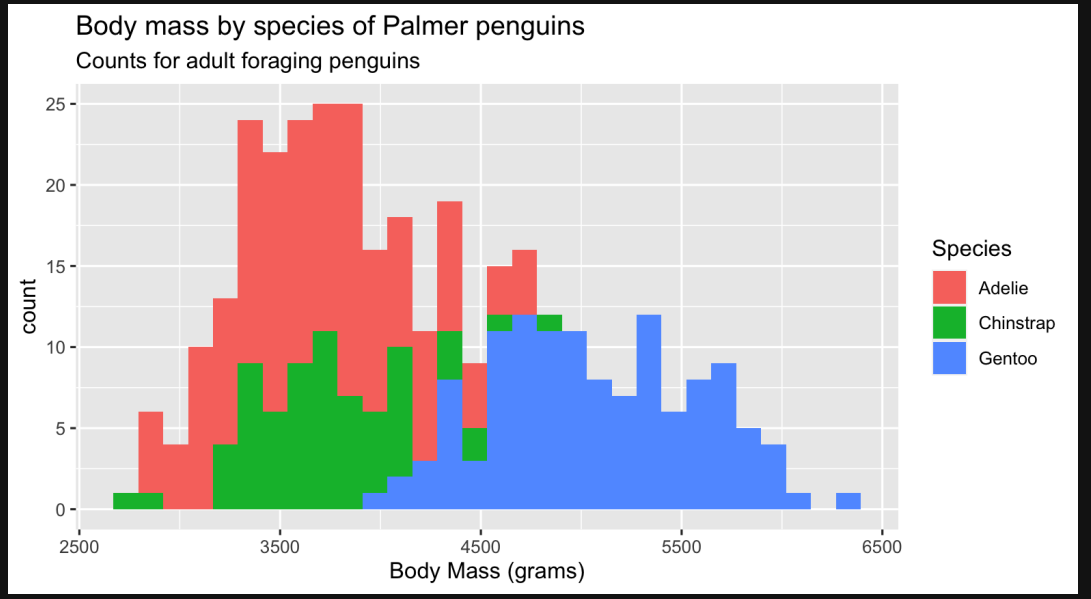
Create new labels

```
labs_bodymass_vs_species <- ggplot2::labs(
  title = "Body mass by species of Palmer penguins",
  subtitle = "Counts for adult foraging penguins",
  x = "Body Mass (grams)",
  fill = "Species")
```

Create a histogram of `body_mass_g`, colored (filled) by `species`

```
ggplot(data = penguins) +
  geom_histogram(
    mapping = aes(x = body_mass_g,
                  fill = species)) +
  labs_bodymass_vs_species
```

Notice the overlapping distributions of `body_mass_g`



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

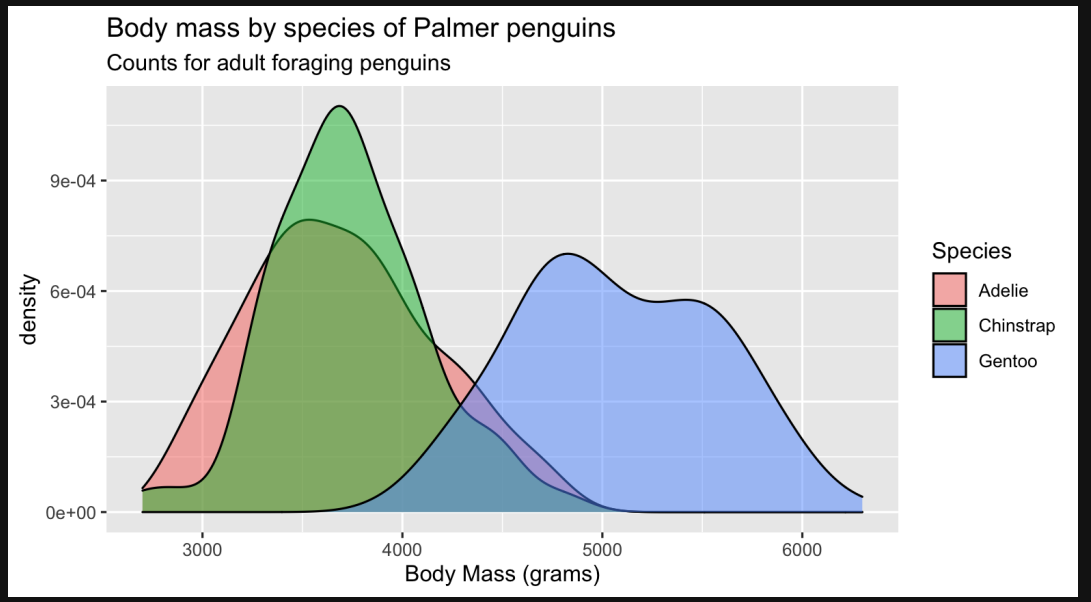
graph 11 Density plots

Density plots are also great for comparing overlapping distributions

Create a density plot with `geom_density()`

Change the color saturation by setting alpha to 1/2

```
ggplot(data = penguins) +
  geom_density(
    mapping = aes(x = body_mass_g,
                  fill = species),
    alpha = 1/2) +
  labs_bodymass_vs_species
```



Also check out ridgeline plots

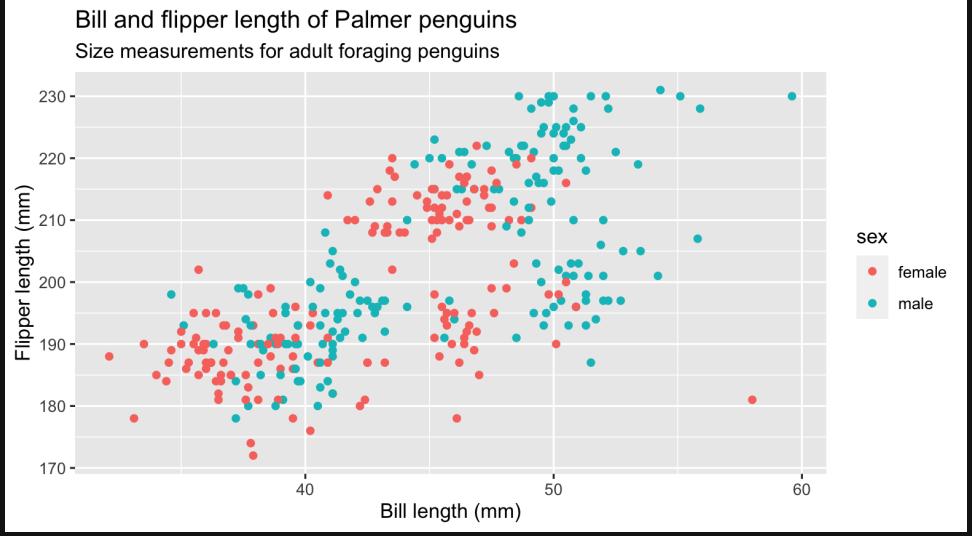
https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Mapping vs. setting aesthetics

Mapping vs. setting

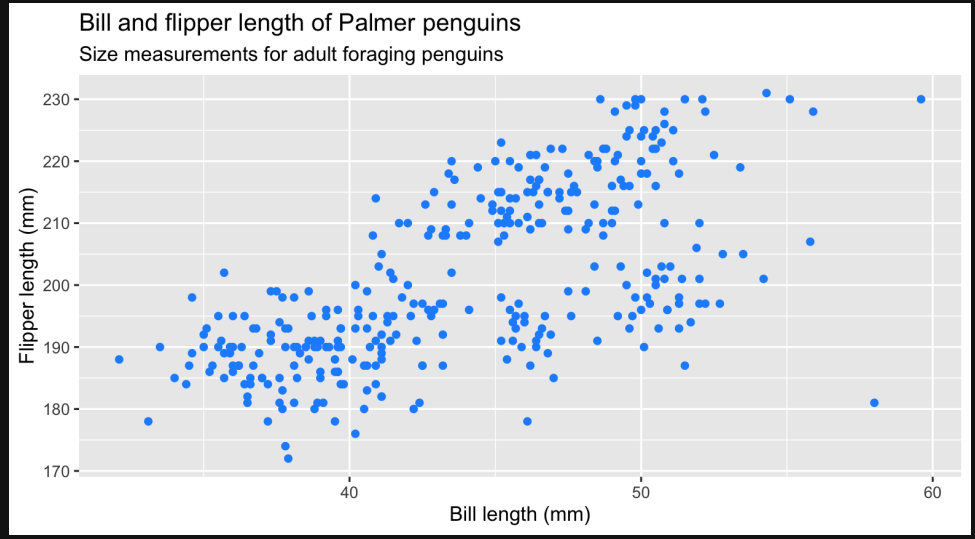
The last few graphs have mapped aesthetics inside and outside the `aes()` function

```
ggplot(data = penguins_no_miss) +
  geom_point(
    mapping = aes(x = bill_length_mm,
                  y = flipper_length_mm,
                  color = sex)) + # inside
  labs_bill_vs_flippper
```



Inside the `aes()` function is 'mapping', outside the `aes()` function is 'setting'

```
ggplot(data = penguins_no_miss) +
  geom_point(
    mapping = aes(x = bill_length_mm,
                  y = flipper_length_mm),
                  color = "dodgerblue") + # outside
  labs_bill_vs_flippper
```



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Mapping vs. setting

From ggplot2 book

> *If you want appearance to be governed by a variable, put the specification inside `aes()`; if you want override the default size or colour, put the value outside of `aes()`.*

graph 12 Setting graph aesthetics

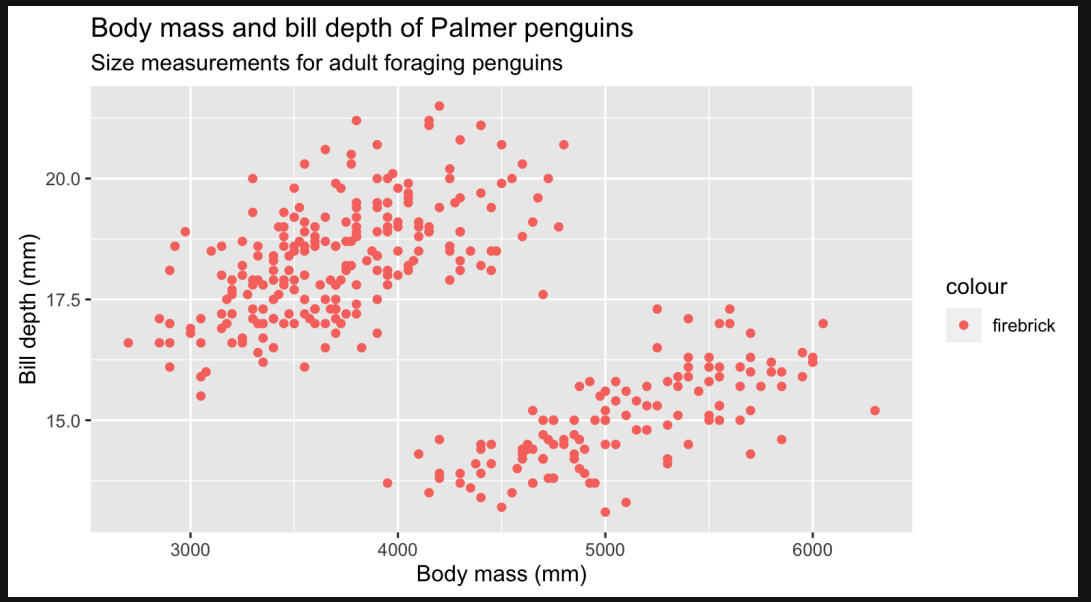
Change the code below to make the points `"firebrick"` red

Create labels

```
labs_body_mass_vs_bill_depth <- ggplot2::labs(
  title = "Body mass and bill depth of Palmer penguins",
  subtitle = "Size measurements for adult foraging penguins",
  x = "Body mass (mm)",
  y = "Bill depth (mm)")
```

What color will the points be on this graph?

```
ggplot(data = penguins) +
  geom_point(
    mapping = aes(x = body_mass_g,
                  y = bill_depth_mm,
                  color = "firebrick")) +
  labs_body_mass_vs_bill_depth
```



TIP: the legend tells us `geom_point()` is looking for a mapped variable in the penguins dataset named `"firebrick"`

Combining layers

graph 13 New layer, new data, no problem

Each `geom_*()` function also has a `data` argument, so we can supply new data at each layer

Create a small dataset with max values from `bill_length_mm`, `bill_depth_mm`, `flipper_length_mm` and `body_mass_g`

Create a dataset of the max body mass and bill depth (`big_penguins`)

```
big_penguins <- bind_rows(
  slice_max(.data = penguins, bill_length_mm, n = 1),
  slice_max(.data = penguins, bill_depth_mm, n = 1),
  slice_max(.data = penguins, flipper_length_mm, n = 1),
  slice_max(.data = penguins, body_mass_g, n = 1))
```

Create data `label` and `source`

```
big_penguins <- mutate(.data = big_penguins,
       label = case_when(
   bill_length_mm == 59.6 ~ paste0("long bill = ", bill_length_mm),
   bill_depth_mm == 21.5 ~ paste0("deep bill = ", bill_depth_mm),
   flipper_length_mm == 231 ~ paste0("big flipper = ", flipper_length_mm),
   body_mass_g == 6300 ~ paste0("big bird = ", body_mass_g)),
       source = case_when(
   bill_length_mm == 59.6 ~ "max bill length",
   bill_depth_mm == 21.5 ~ "max bill depth",
   flipper_length_mm == 231 ~ "max flipper length",
   body_mass_g == 6300 ~ "max body mass"))
```


https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Our label dataset

View `label` and `source` from `big_penguins`

label	source
<chr>	<chr>
long bill = 59.6	max bill length
deep bill = 21.5	max bill depth
big flipper = 231	max flipper length
big bird = 6300	max body mass

4 rows

Objective: Create a scatter-plot to show the relationship between body mass, flipper length, and bill length.

graph 13 Layer 1

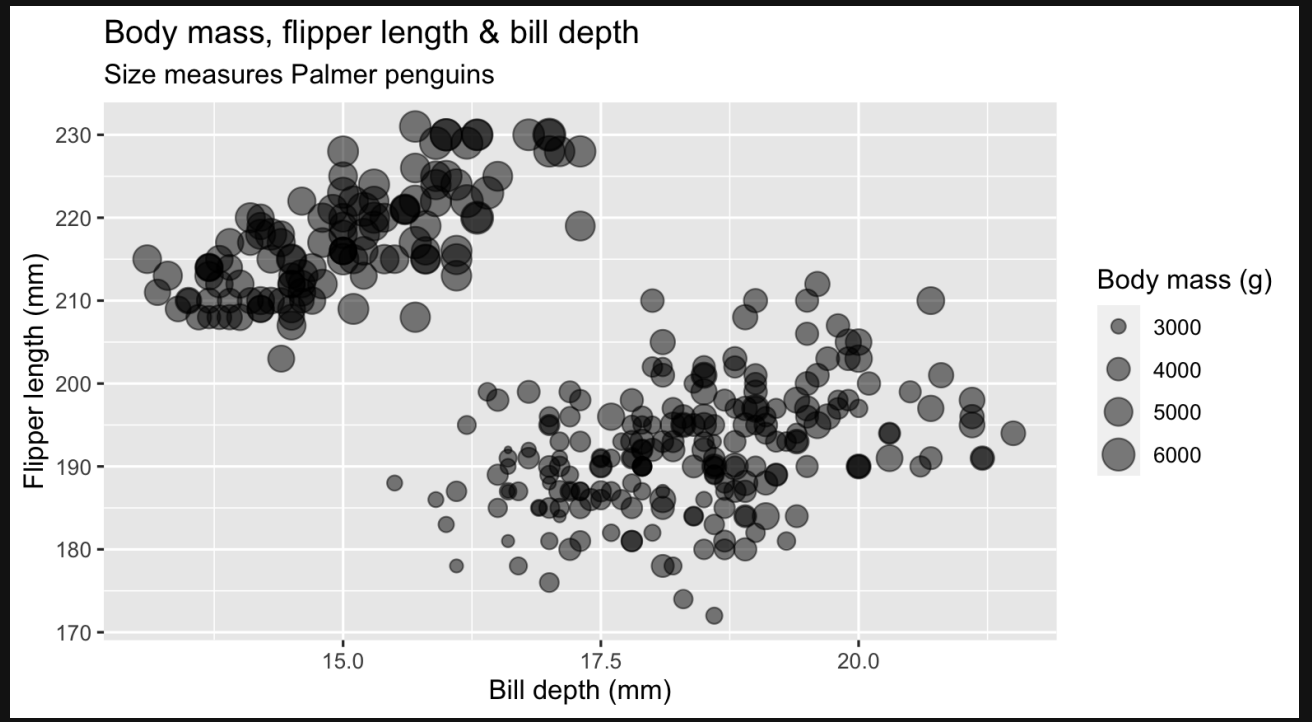
Create layer 1 with `geom_point()` using `size`

Create labels

```
labs_bodymass_bill_depth_flipper_length <- labs(
  title = "Body mass, flipper length & bill depth",
  subtitle = "Size measures Palmer penguins",
  x = "Bill depth (mm)",
  y = "Flipper length (mm)",
  size = "Body mass (g)")
```

Add new layer with new data

```
ggplot(data = penguins_no_miss) +
  # layer 1
  geom_point(
    mapping = aes(x = bill_depth_mm,
                  y = flipper_length_mm,
                  size = body_mass_g),
    alpha = 1/2)  +
  # labels
  labs_bodymass_bill_depth_flipper_length
```



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

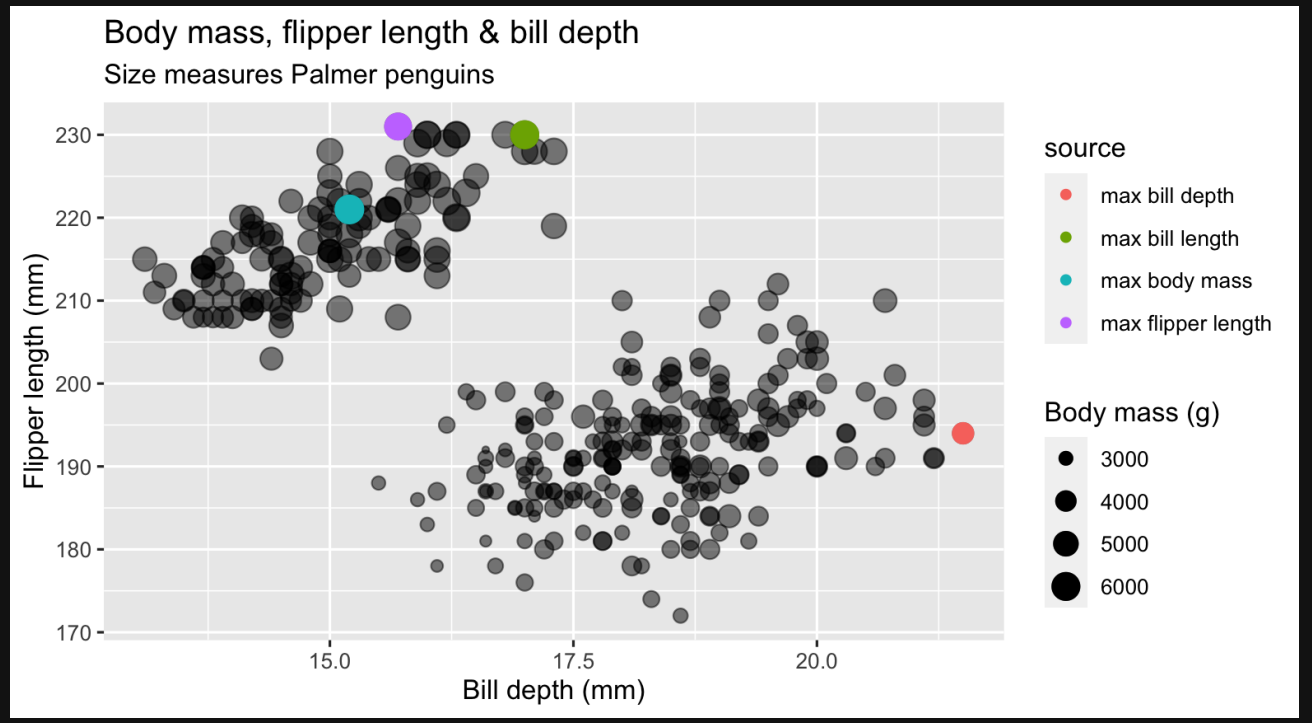
graph 14 Layer 2

Create layer 2 with another `geom_point()` using `color` and `size`

Set the `data` argument to `big_penguins`

Use `scale_size()` to adjust point scaling

```
ggplot(data = penguins_no_miss) +
  geom_point(
    mapping = aes(x = bill_depth_mm,
                  y = flipper_length_mm,
                  size = body_mass_g),
    alpha = 1/2) +
  # layer 2
  geom_point(data = big_penguins,
             mapping = aes(x = bill_depth_mm,
                           y = flipper_length_mm,
                           # color by source
                           color = source,
                           size = body_mass_g)) +
  # re-scale
  scale_size(range = c(1, 5)) +
  # labels
  labs_bodymass_bill_depth_flipper_length
```



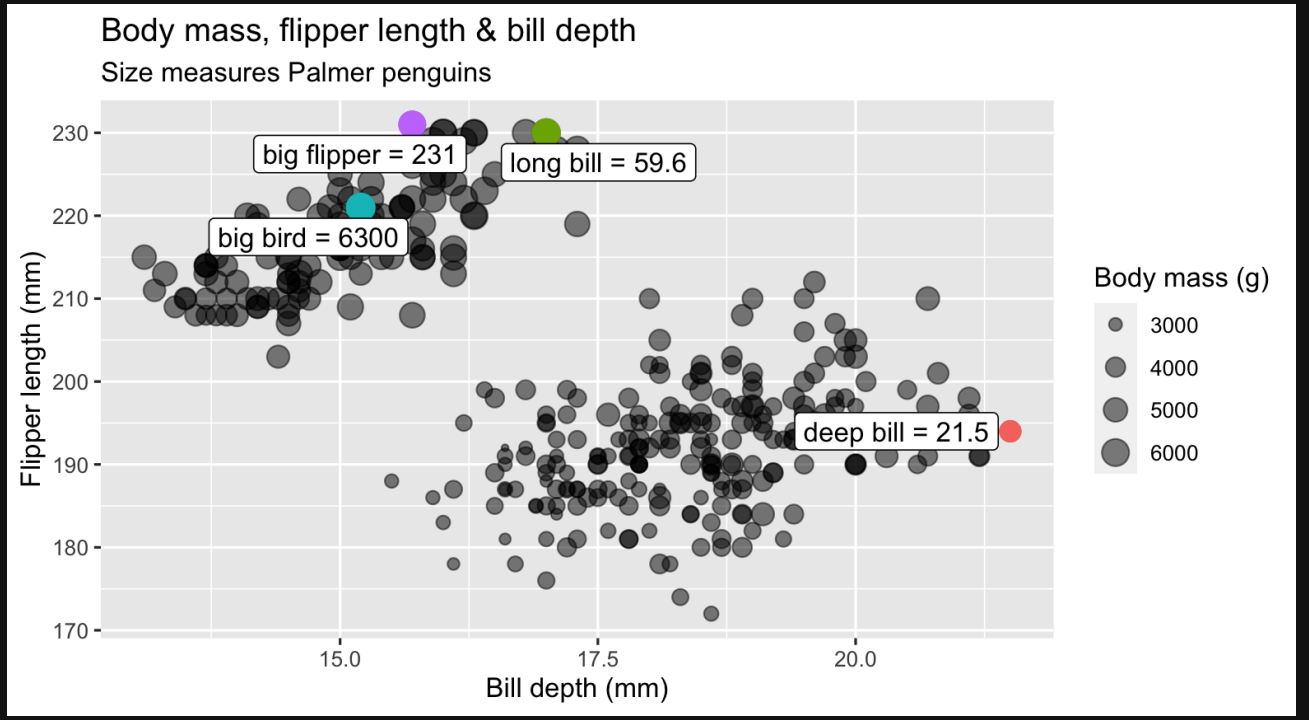
https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

graph 15 Label 3 (max values)

Add layer 3 with the `geom_label_repel()` function from `ggrepel`

Add layer for `labels` in `big_penguins`

```
ggplot(data = penguins_no_miss) +
  geom_point(
    mapping = aes(x = bill_depth_mm,
                  y = flipper_length_mm,
                  size = body_mass_g),
    alpha = 1/2) +
  geom_point(data = big_penguins,
             mapping = aes(x = bill_depth_mm,
                           y = flipper_length_mm,
                           color = source,
                           size = body_mass_g),
             # remove legend
             show.legend = FALSE) +
  # rescale
  scale_size(range = c(1, 5)) +
  # layer 3
  ggrepel::geom_label_repel(
    data = big_penguins,
    mapping = aes(x = bill_depth_mm,
                  y = flipper_length_mm,
                  label = label)) +
  # labels
  labs_bodymass_bill_depth_flipper_length
```



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Facets

Facets

From ggplot2 book

> *Small multiples are a powerful tool for exploratory data analysis: you can rapidly compare patterns in different parts of the data and see whether they are the same or different.*

Facets = small multiples

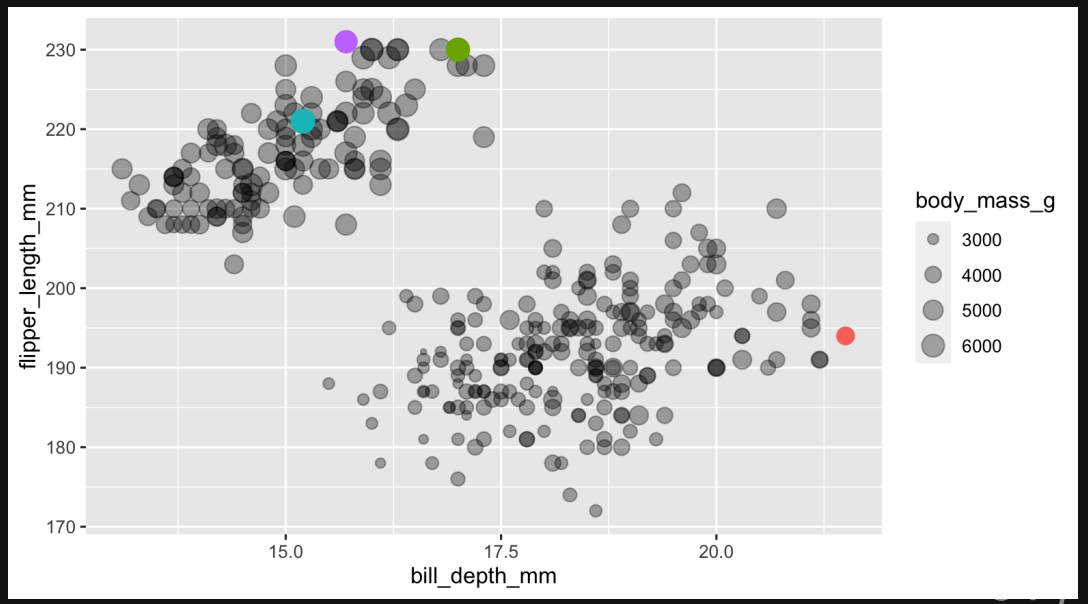
In the previous graph, we used multiple aesthetics (`color`, `size`, `shape`)

Can we explore these relationships by `sex` or `species`?

Store graph 15 in `ggp_penguin_measures`

```
ggp_penguin_measures <- ggplot(data =
penguins_no_miss) +
  geom_point(
    mapping = aes(x = bill_depth_mm,
                  y = flipper_length_mm,
                  size = body_mass_g),
    alpha = 1/3) +
  geom_point(data = big_penguins,
             mapping = aes(x = bill_depth_mm,
                           y = flipper_length_mm,
                           color = source,
                           size = body_mass_g),
             show.legend = FALSE) +
  scale_size(range = c(1, 5))
```

```
ggp_penguin_measures
```

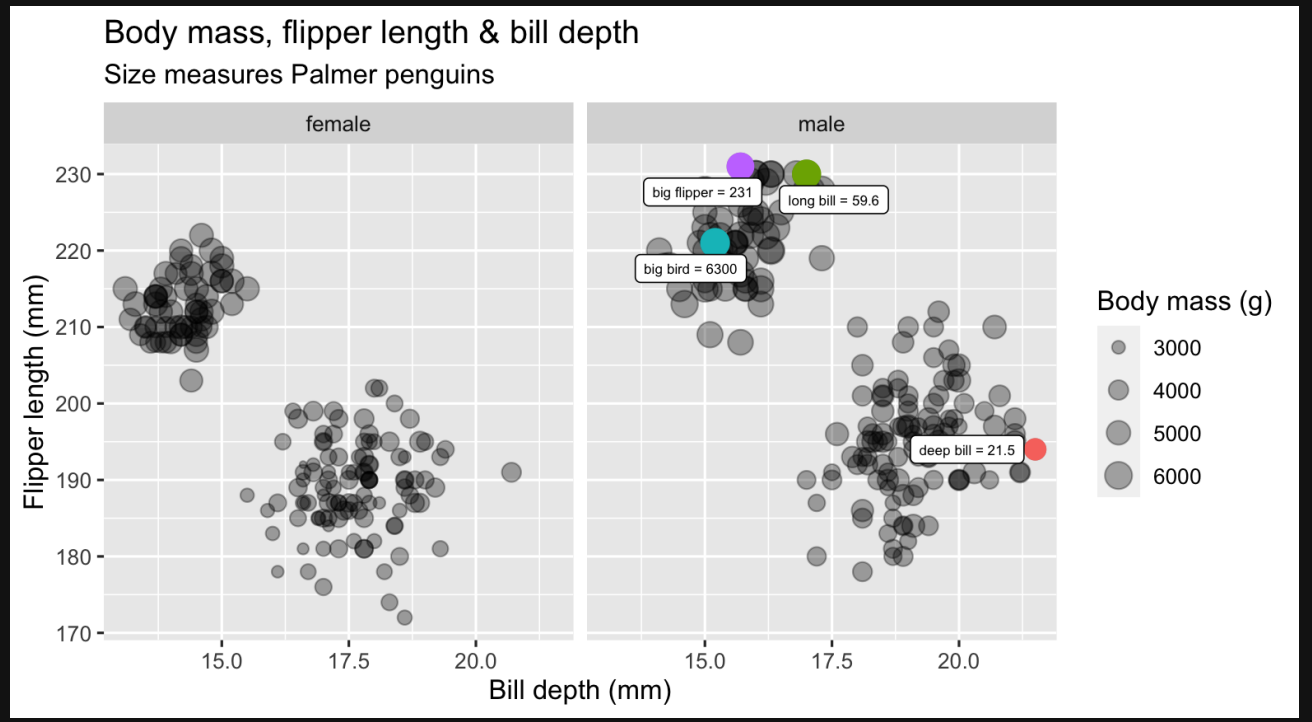


graph 16 Facet by sex

Use `facet_wrap()` to view our previous graph by `sex`

`facet_wrap()` uses `~ [var]`

```
ggp_penguin_measures +
  ggrepel::geom_label_repel(
    data = big_penguins,
    mapping = aes(x = bill_depth_mm,
                  y = flipper_length_mm,
                  label = label),
  # adjust size
                size = 2) +
  # facet by sex
  facet_wrap(. ~ sex) +
  # labels
  labs_bodymass_bill_depth_flipper_length
```



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

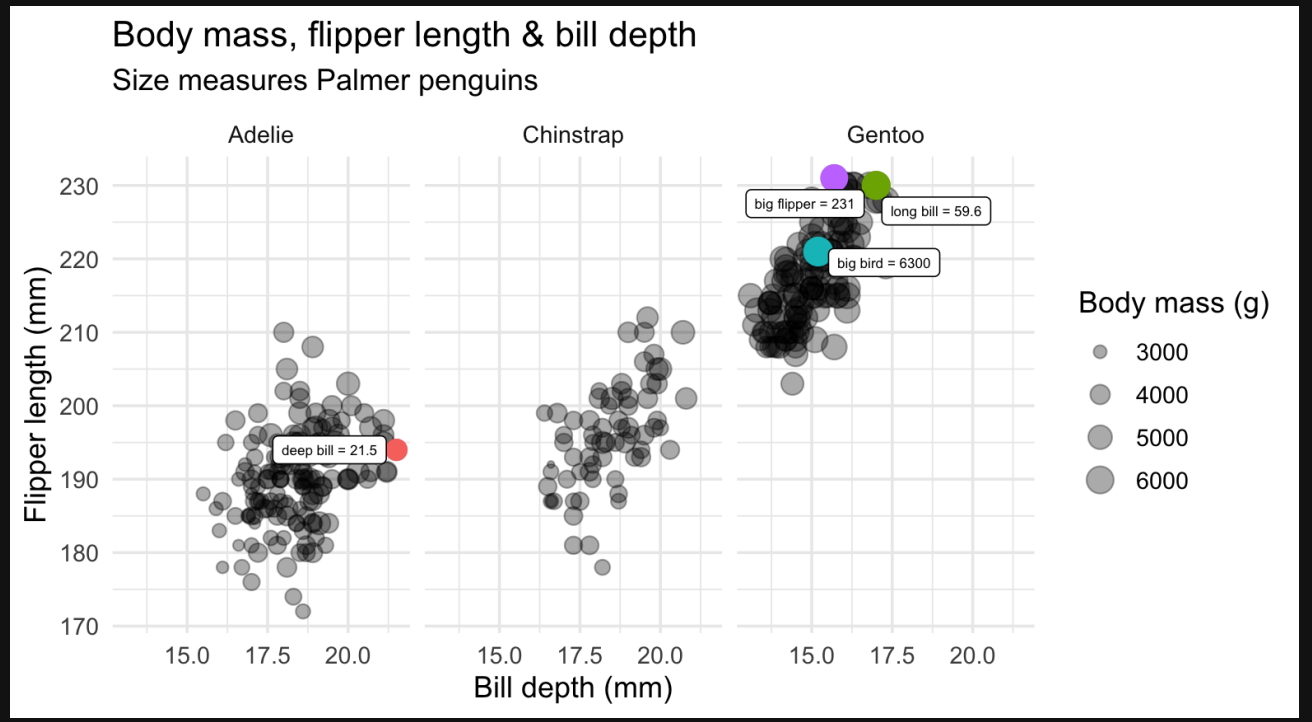
graph 17 Facet by species

Change facet_wrap() to build graphs by species and add theme

Change facet_wrap() to ~ species

Add theme_minimal()

```
ggp_penguin_measures +
  ggrepel::geom_label_repel(
    data = big_penguins,
    mapping = aes(x = bill_depth_mm,
                  y = flipper_length_mm,
                  label = label),
    size = 2) +
  # change to species
  facet_wrap(. ~ species) +
  # add theme
  theme_minimal() +
  # labels
  labs_bodymass_bill_depth_flipper_length
```



https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

Recap

What we've covered

1. Build labels (set your expectations)

2. View data before building any graphs

3. Building graphs layer-by-layer (data, mapping, geom)

4. Mapping variables to graph elements (color, position, size, etc)

5. Extending graphs by combining layers

6. Using facets to explore relationships

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04

ODSC
The Future of AI Gathers Here

Thanks!

@mjfrigaard
@mjfrigaard
mjfrigaard@pm.e
What does "λέξις" mean?

https://github.com/mjfrigaard/talks/tree/main/odsc-eda-2022-04