Lecture 03: Exploring and visualising data

ENVX1002 Introduction to Statistical Methods

Januar Harianto
The University of Sydney

Mar 2025

THE UNIVERSITY OF SYDNEY

CRICOS 00026A TEQSA PRV12057

Learning outcomes

After this **week**, you will be able to:

1. Understand the importance of data exploration before analysis
2. Apply exploratory functions to explore and summarise datasets
3. Identify different data types and structures in datasets
4. Select appropriate visualization types based on data characteristics
5. Understand the Grammar of Graphics approach to data visualisation
6. Create and customise visualisations in R using the ggplot2 package
7. Build plots layer-by-layer using the ggplot2 framework
8. Interpret distributions, including skewness, kurtosis, and outliers

Quick checklist

By now you should have...

- [ ] Installed **R** and **RStudio**
- [ ] Completed Lecture 2 content and read the ENVX1002 R guide
- [ ] A basic understanding of measures of central tendency and spread
- [ ] A basic understanding of what a `function(argument = value)` is in R
- [ ] Rendered a few Quarto documents in RStudio

Core concepts

Data exploration

Why explore data before analysis?

- Identify patterns, outliers, and relationships
- Detect data quality issues
- Guide selection of appropriate statistical methods
- Avoid incorrect conclusions from flawed data

The data exploration workflow

1. Understand data structure and types
2. Examine distributions and summary statistics
3. Visualise relationships between variables
4. Identify patterns and anomalies

Types of data: recap from Week 1

Data in R can be broadly categorized as either **categorical** or **continuous**.

- Different data types require different analysis approaches
- Understanding data types helps select appropriate visualisations
- **R stores different data types in specific formats** (which is why we need to know what they are when we import data!)

Categorical data

- **Nominal**: no natural order, e.g.
  - Species (dog, cat, fish)
  - Hair colour (black, brown, blonde)
  - Blood type (A, B, AB, O)
- **Ordinal**: natural order exists, e.g.
  - Education (primary, secondary, tertiary)
  - Pain scale (mild, moderate, severe)
  - T-shirt sizes (S, M, L, XL)

Continuous data

- **Interval**: equal intervals, no true zero, e.g.
  - Temperature in °C (0°C isn't "no temperature")
  - Calendar dates
  - pH scale
- **Ratio**: equal intervals with true zero, e.g.
  - Height (0 cm = no height)
  - Weight (0 kg = no weight)
  - Age (0 years = birth)

Different types of data are distributed differently

Understanding how data is distributed is crucial for selecting appropriate ways to explain it to others.

Normal distribution: Introduction

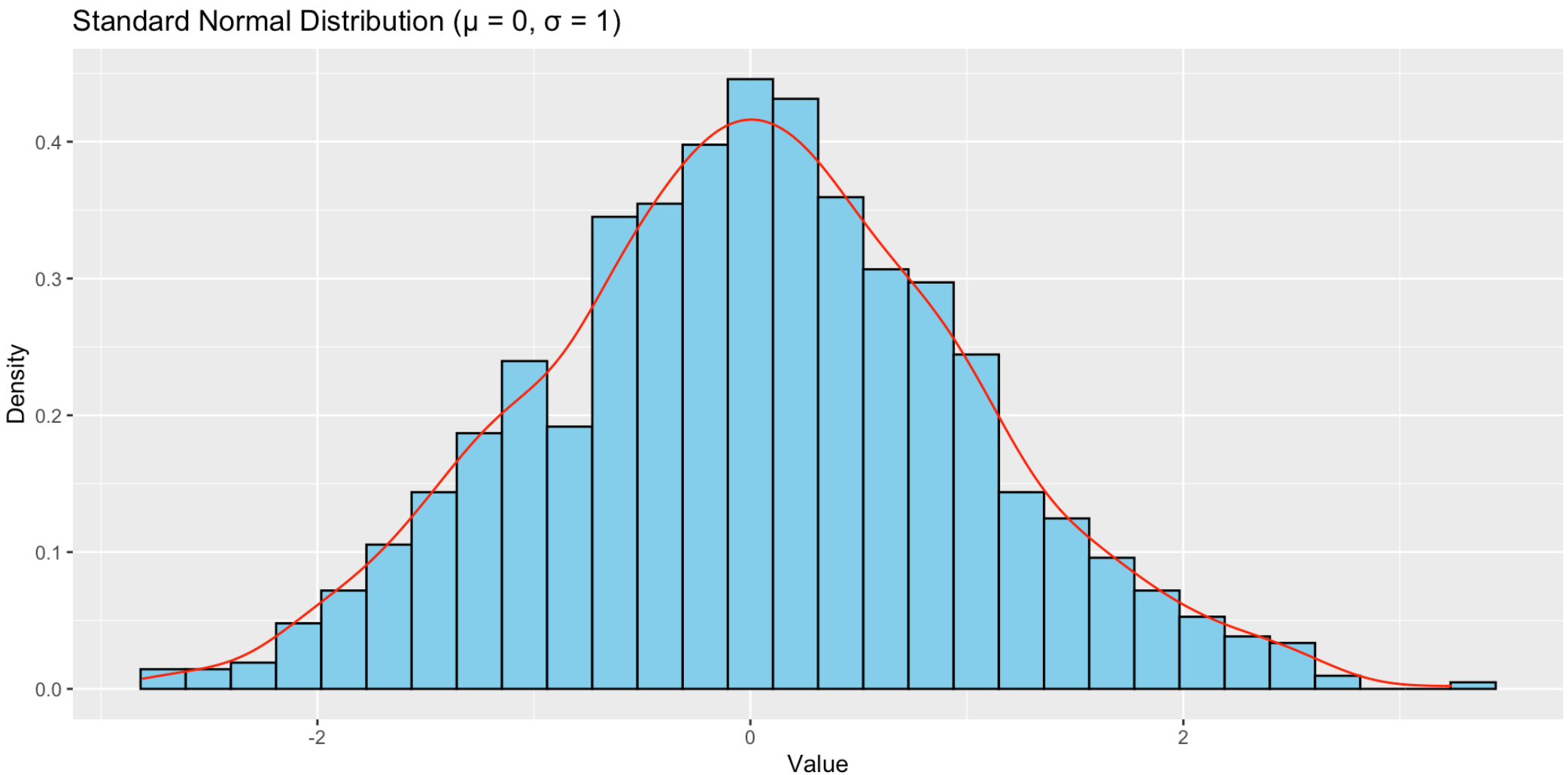
- Bell-shaped, symmetric curve
- Defined by mean (μ) and standard deviation (σ)
- Many natural phenomena follow this distribution
  - ➟ Heights of individuals in a population
  - ➟ Measurement errors
  - ➟ Many physiological traits

$$X \sim N(\mu, \sigma^2)$$

> The random variable X follows a normal distribution with mean μ and variance σ^2

What does a normal distribution look like?

▶ Code

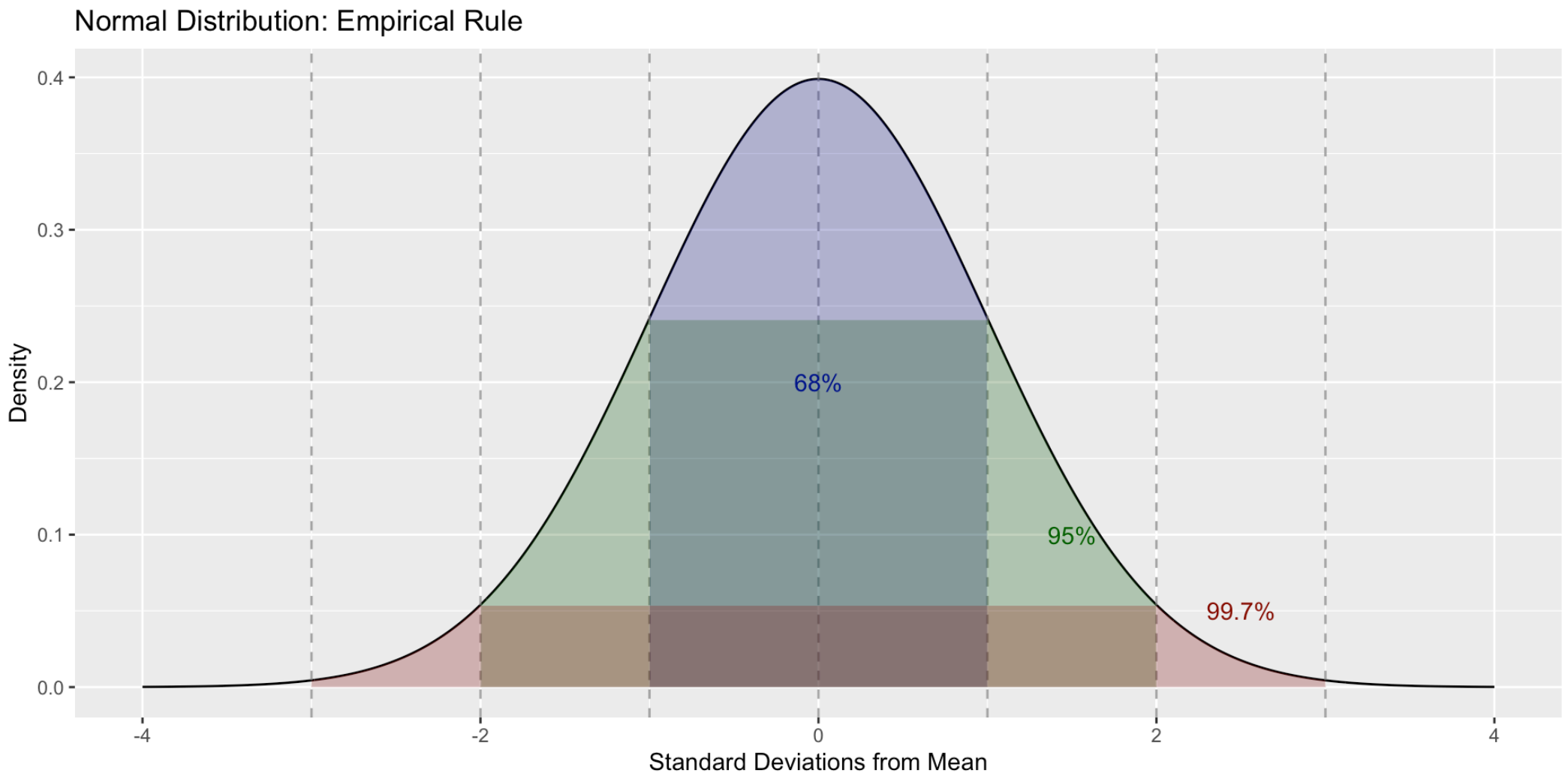


Properties of normal distribution: The empirical rule

- Mean = median = mode
- ~68% of data within 1σ of mean
- ~95% of data within 2σ of mean
- ~99.7% of data within 3σ of mean

The empirical rule visualised

▶ Code



Why normal distributions matter in data exploration

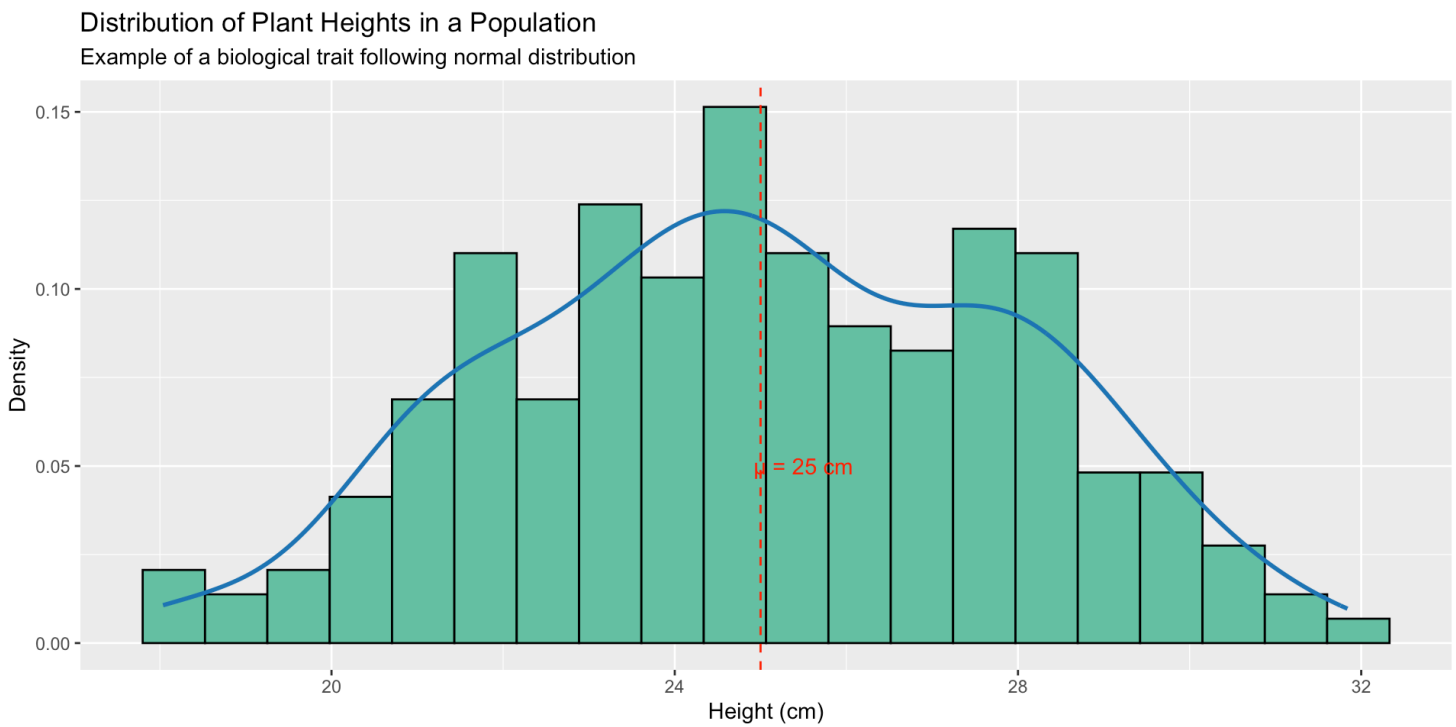
When exploring data, understanding distributions helps you:

1. **Identify patterns and anomalies**
    - Is your data normally distributed as expected?
    - Are there unexpected skews or outliers?
2. **Choose appropriate analysis methods**
    - Many statistical tests assume normality
    - Non-normal data may require different approaches
3. **Interpret results correctly**
    - Context for understanding how unusual a value is
    - Framework for making statistical inferences

Example

Many biological traits follow normal distributions. For example, plant heights within a species:

▶ Code



This example shows how plant heights cluster around the mean (25 cm) following a normal distribution pattern. This helps researchers identify outliers, establish experimental categories, and detect environmental effects on growth patterns.

Skewness

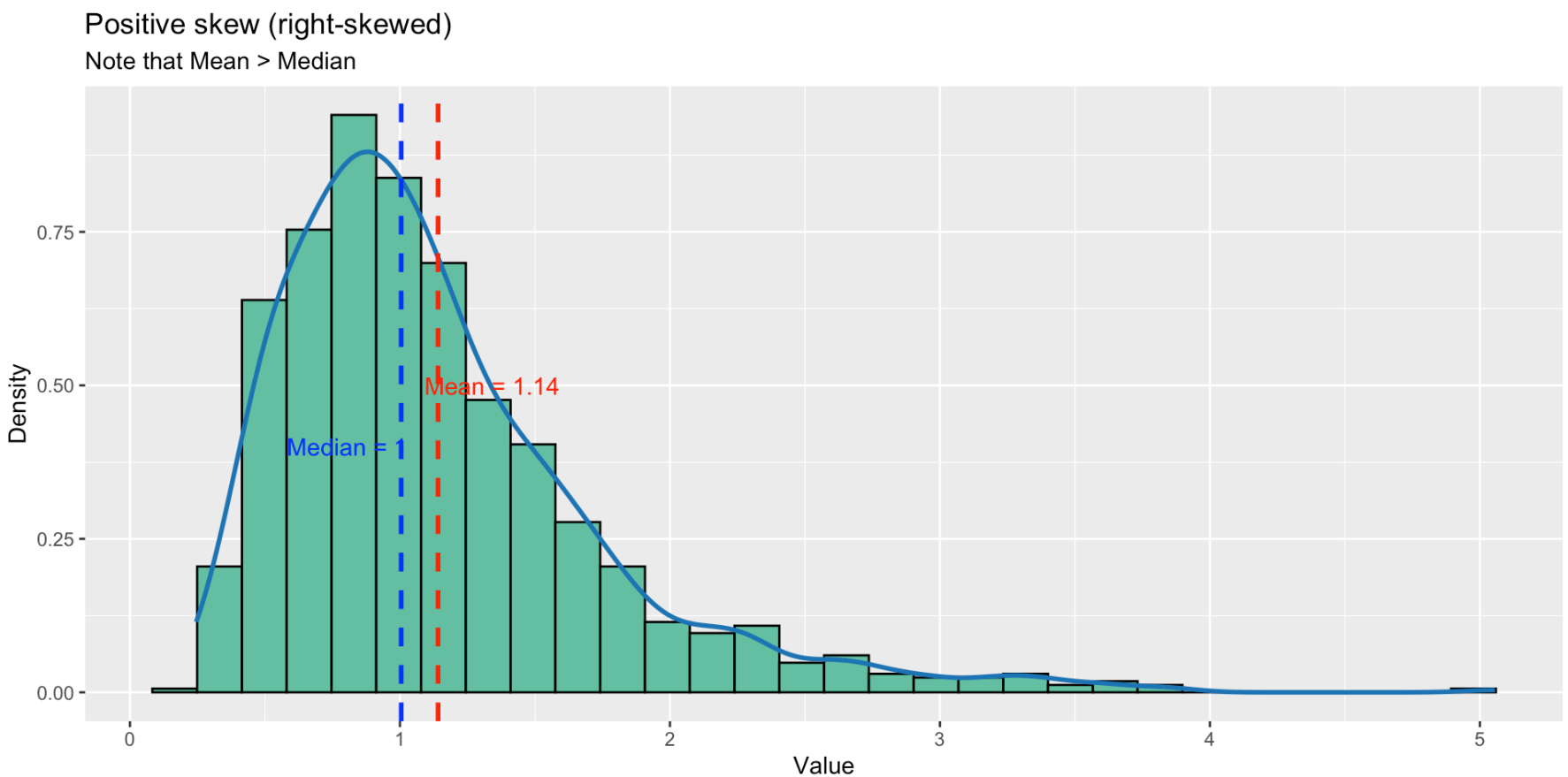
What is skewness?

- Measure of asymmetry in a distribution
- Indicates which side of the distribution has a longer tail
- Important for selecting appropriate statistical tests

Positive skew (right-skewed)

- Long tail on right side
- Mean > median

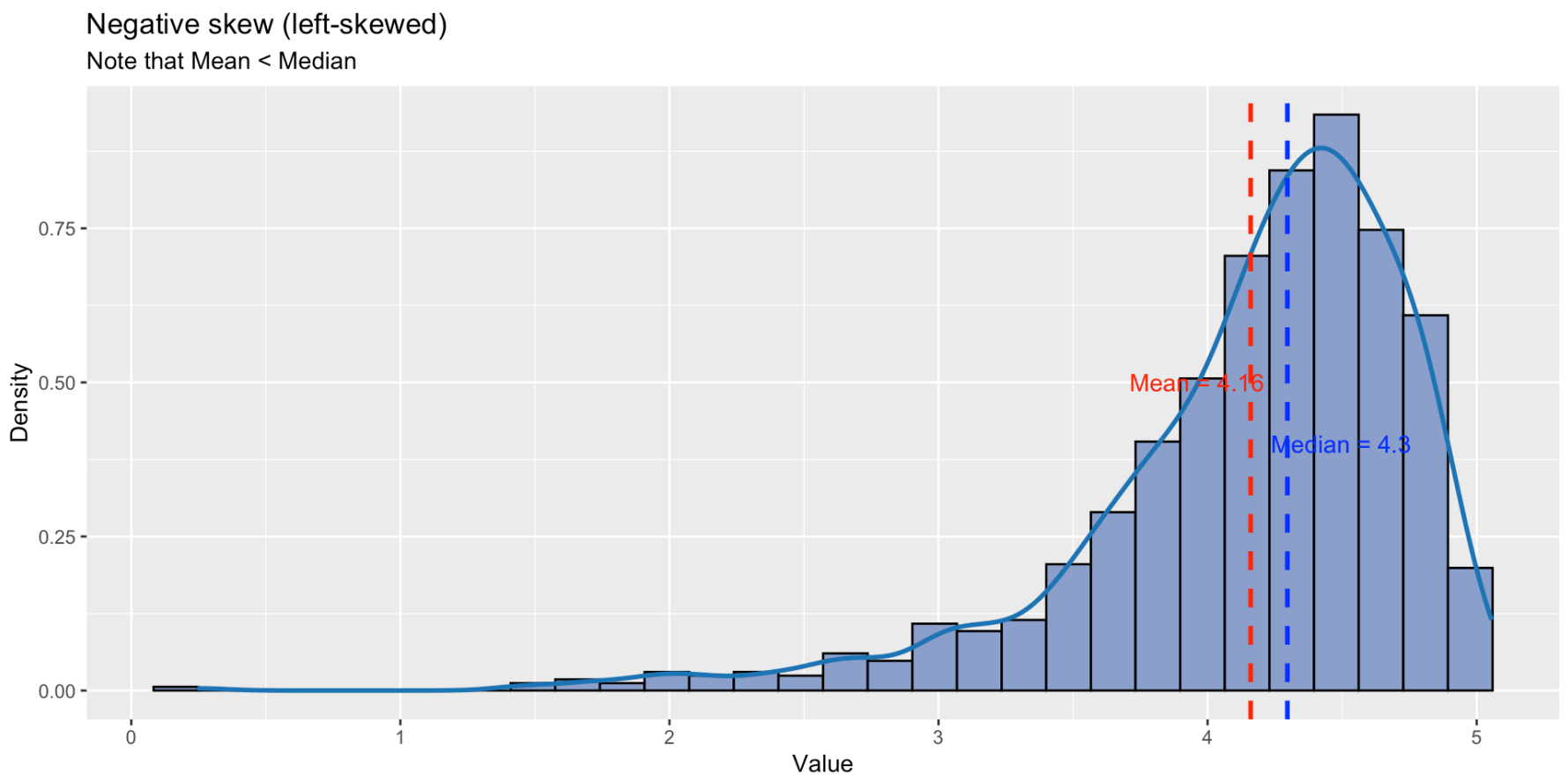
▶ Code



Negative skew (left-skewed)

- Long tail on left side
- Mean < median

▶ Code



Kurtosis

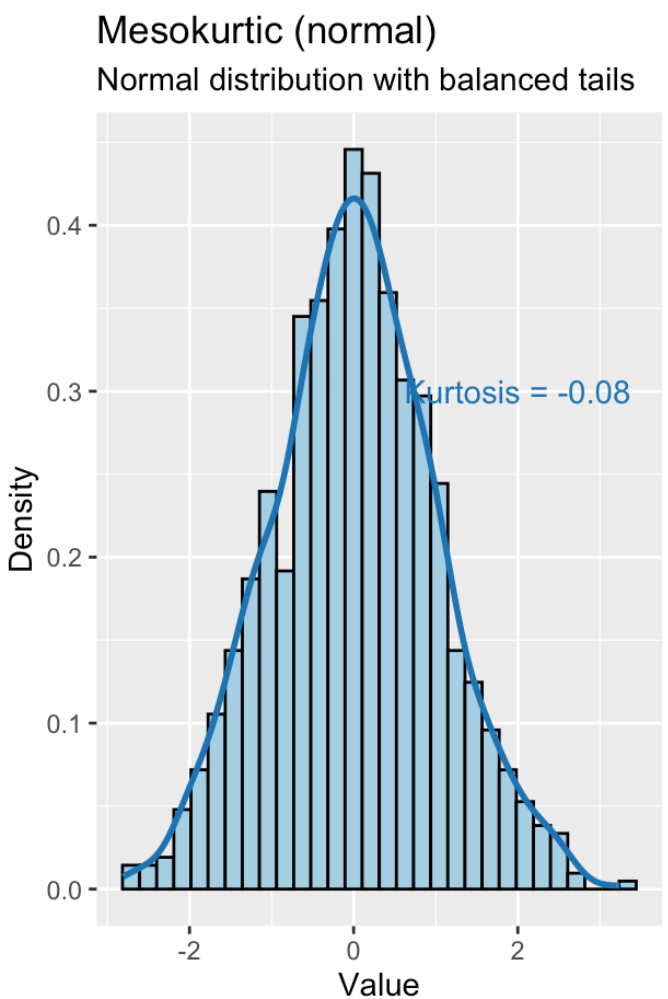
- Measure of “tailedness” of a distribution
- Describes the shape of a distribution’s tails relative to its overall shape
- Affects the choice of statistical methods

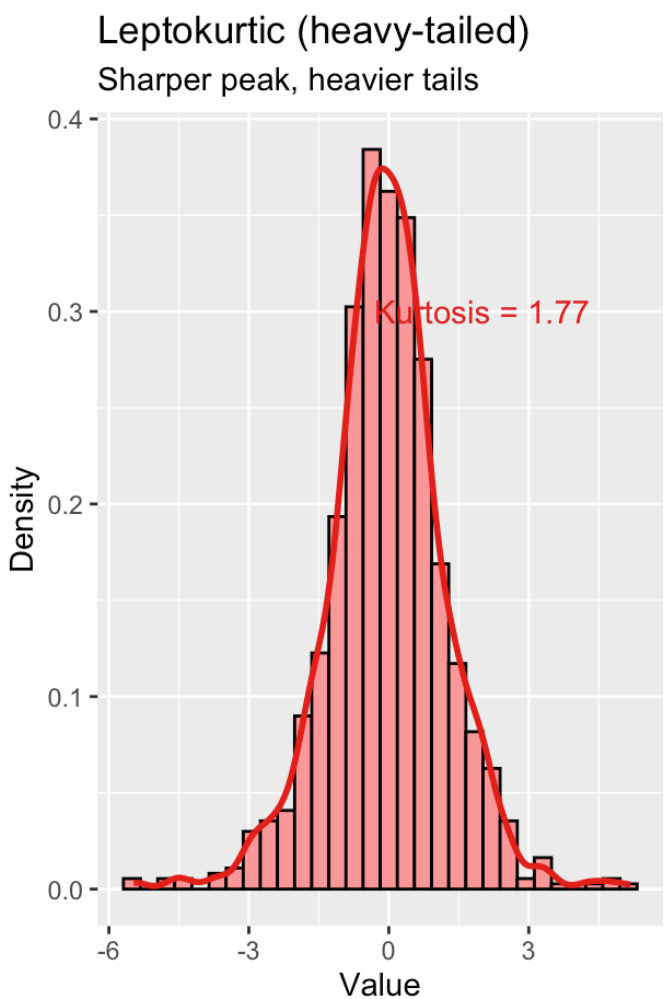
Types of kurtosis

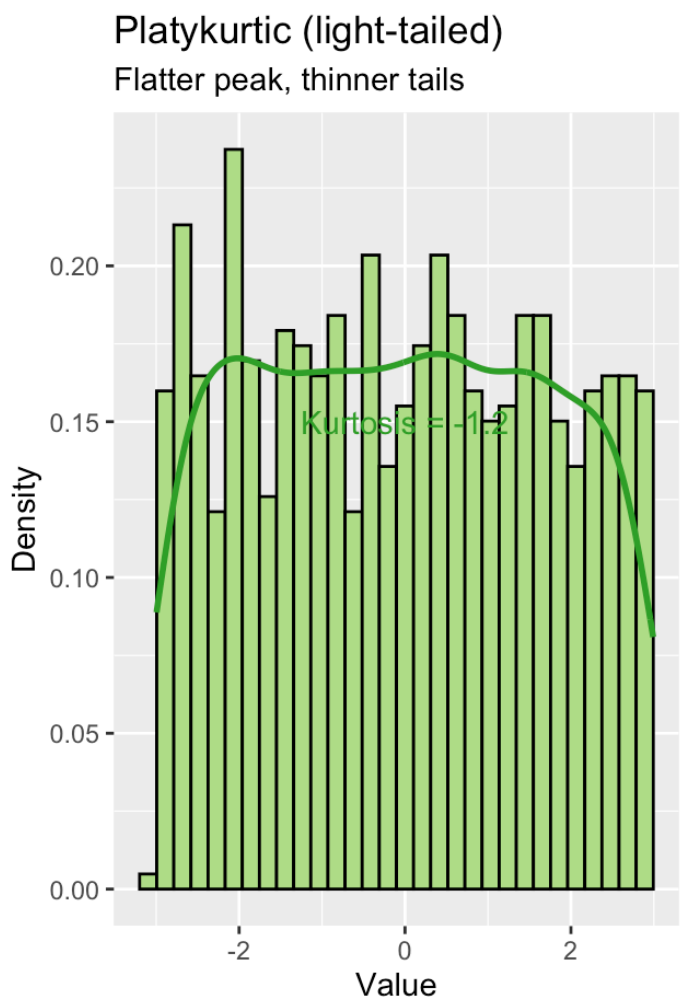
- **Mesokurtic**: Normal distribution (kurtosis = 3)
- **Leptokurtic**: Sharper peak, heavier tails (kurtosis > 3)
- **Platykurtic**: Flatter peak, thinner tails (kurtosis < 3)

Visualising kurtosis

▶ Code







Exploring data in R

Data structures in R

Data is stored in R in various structures, each with specific purposes:

Vectors: 1-dimensional collection of elements

▶ Code

```
[1] 1.65 1.70 1.75 1.80 1.85
```

Data frames: 2-dimensional tables with rows and columns

▶ Code

```
  species height weight
1       A   1.65     60
2       B   1.70     65
3       C   1.75     70
4       A   1.80     75
5       B   1.85     80
```

Other data structures include lists, matrices, arrays, and factors, but these are less common at your level.

Common functions

Use these essential functions to understand your data structure and summary statistics:

```r
# Core function 1: Structure overview
str(df)
```

```
'data.frame':   5 obs. of  3 variables:
 $ species: chr  "A" "B" "C" "A" ...
 $ height : num  1.65 1.7 1.75 1.8 1.85
 $ weight : num  60 65 70 75 80
```

```r
# Core function 2: Statistical summary
summary(df)
```

```
  species              height         weight
 Length:5           Min.   :1.65   Min.   :60
 Class :character   1st Qu.:1.70   1st Qu.:65
 Mode  :character   Median :1.75   Median :70
                    Mean   :1.75   Mean   :70
                    3rd Qu.:1.80   3rd Qu.:75
                    Max.   :1.85   Max.   :80
```

The `summary()` function provides a quick overview of your data and can help identify skewness, outliers, and missing values…but it isn't always enough.

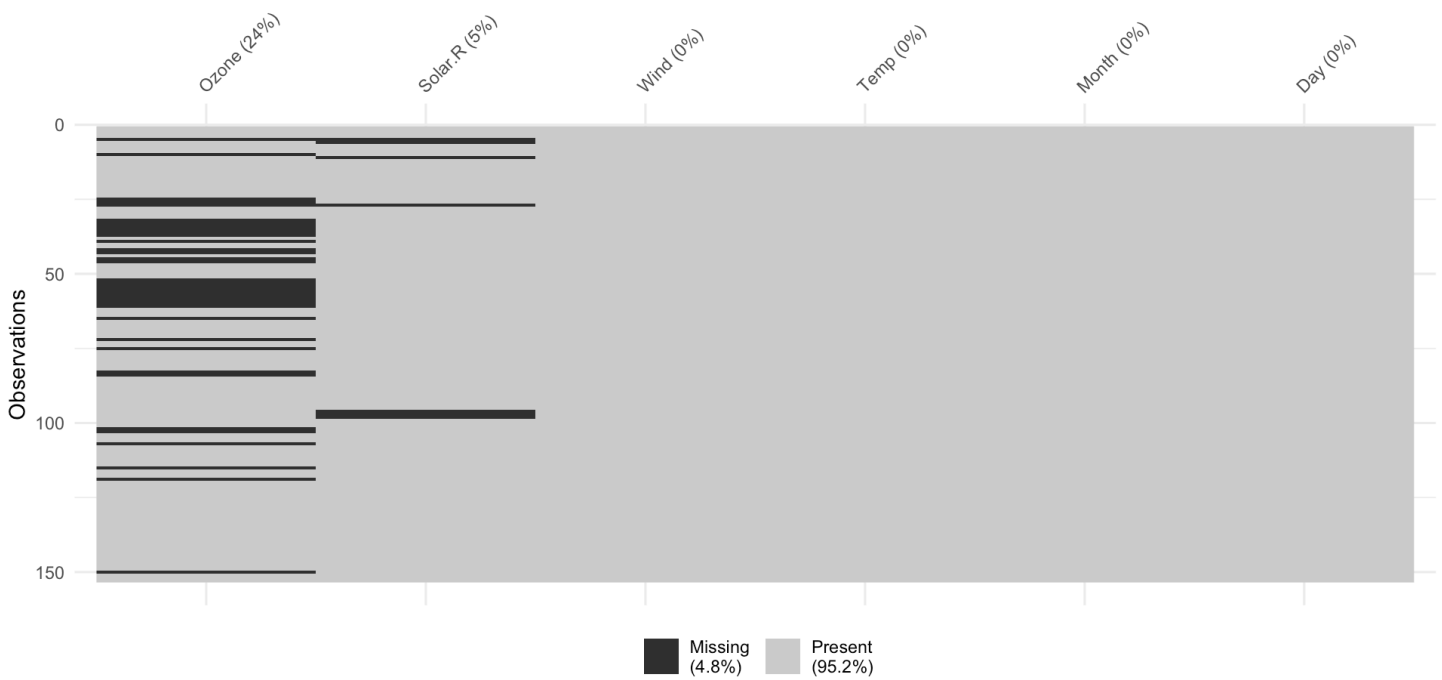
Your options are endless (almost)

There are many specialised functions for exploring different aspects of your data:

```
# Check for unique values in categorical variables
unique(df$species)
```

```
[1] "A" "B" "C"
```

```
# Visualise missing data patterns
library(naniar)
vis_miss(airquality)
```



The value of data visualisation

The output of `vis_mis()` cleary demonstrates the advantage of a **visual** approach to data exploration.

Compare the visualisation to looking at the raw data or a summary of the raw data

```
1  airquality$Ozone
```

```
  [1]  41  36  12  18  NA  28  23  19   8  NA   7  16  11  14  18  14  34   6
 [19]  30  11   1  11   4  32  NA  NA  NA  23  45 115  37  NA  NA  NA  NA  NA
 [37]  NA  29  NA  71  39  NA  NA  23  NA  NA  21  37  20  12  13  NA  NA  NA
 [55]  NA  NA  NA  NA  NA  NA  NA 135  49  32  NA  64  40  77  97  97  85  NA
 [73]  10  27  NA   7  48  35  61  79  63  16  NA  NA  80 108  20  52  82  50
 [91]  64  59  39   9  16  78  35  66 122  89 110  NA  NA  44  28  65  NA  22
[109]  59  23  31  44  21   9  NA  45 168  73  NA  76 118  84  85  96  78  73
[127]  91  47  32  20  23  21  24  44  21  28   9  13  46  18  13  24  16  13
[145]  23  36   7  14  30  NA  14  18  20
```

```
1  summary(airquality$Ozone)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   1.00   18.00   31.50   42.13   63.25  168.00      37
```

Common plot types and their applications

Different types of data require different visualisation approaches. Let’s explore the most common plot types and when to use them.

Histograms

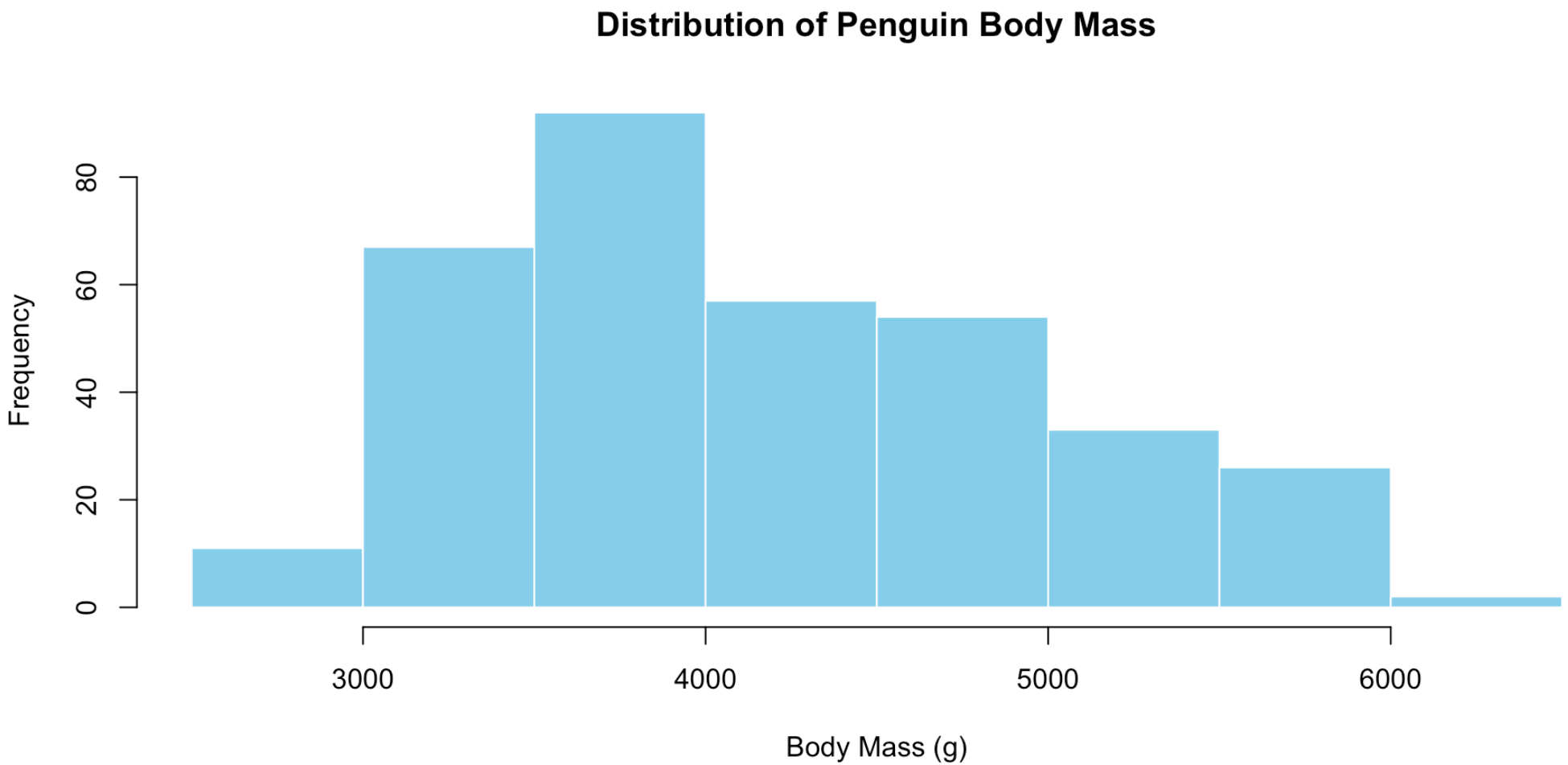
Purpose and applications:

- Visualise distribution of continuous data
- Identify central tendency, spread, outliers, and skewness
- Examine distributions of measurements in biological data

When to use:

- For continuous variables (interval or ratio data)
- When you want to understand the shape of a distribution
- Examples: heights, weights, temperatures, measurements

▶ Code


Distribution of Penguin Body Mass
Frequency
0
20
40
60
80
3000
4000
5000
6000
Body Mass (g)

Bar plots

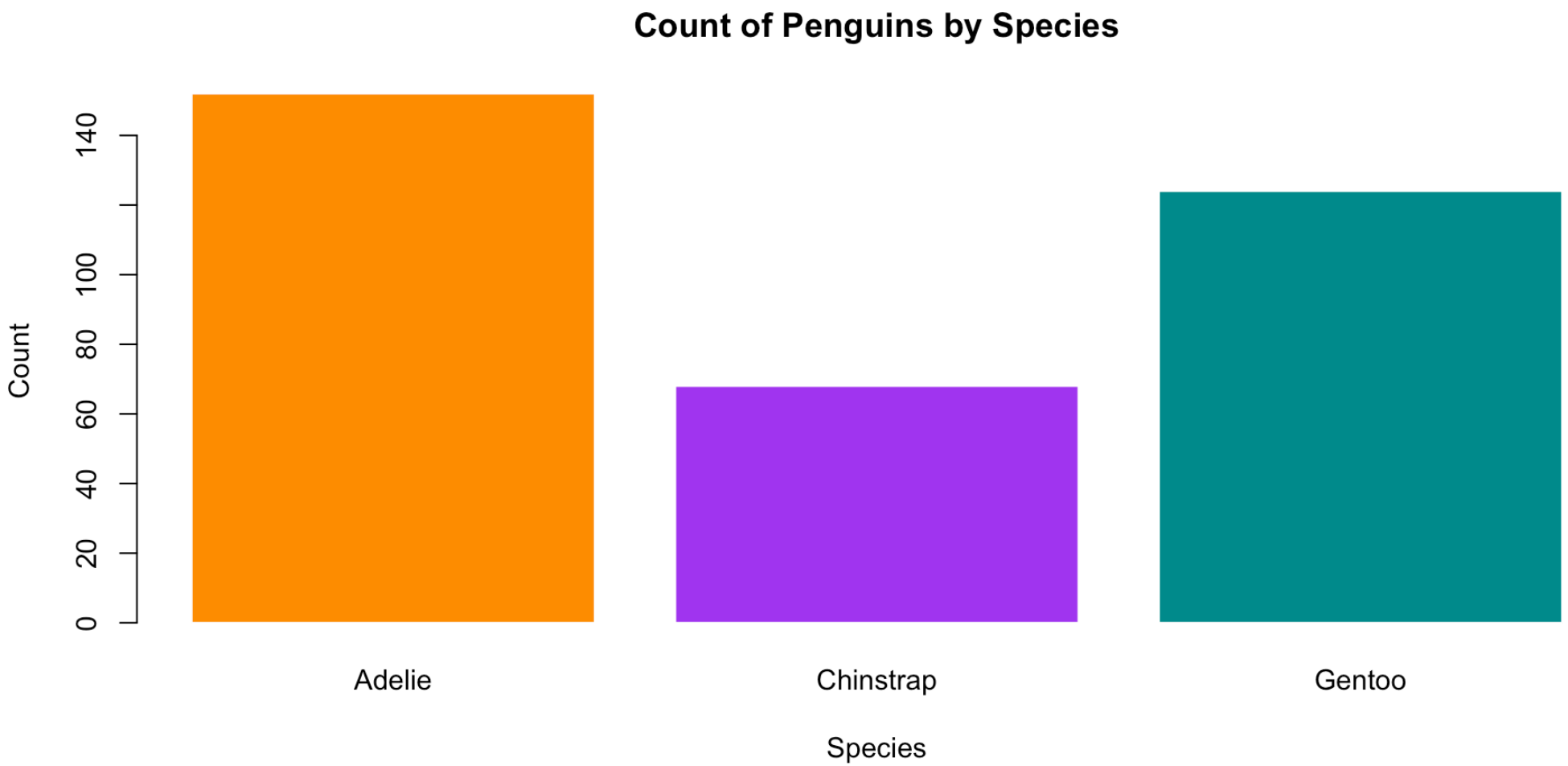
Purpose and applications:

- Compare values across categories
- Show proportions or counts in categorical data
- Visualise species abundance, treatment effects

When to use:

- For categorical variables (nominal or ordinal data)
- When comparing frequencies or counts across groups
- Examples: species counts, treatment groups, survey responses

▶ Code


Count of Penguins by Species
Count
0
20
40
60
80
100
140
Adelie
Chinstrap
Gentoo
Species

Scatterplots

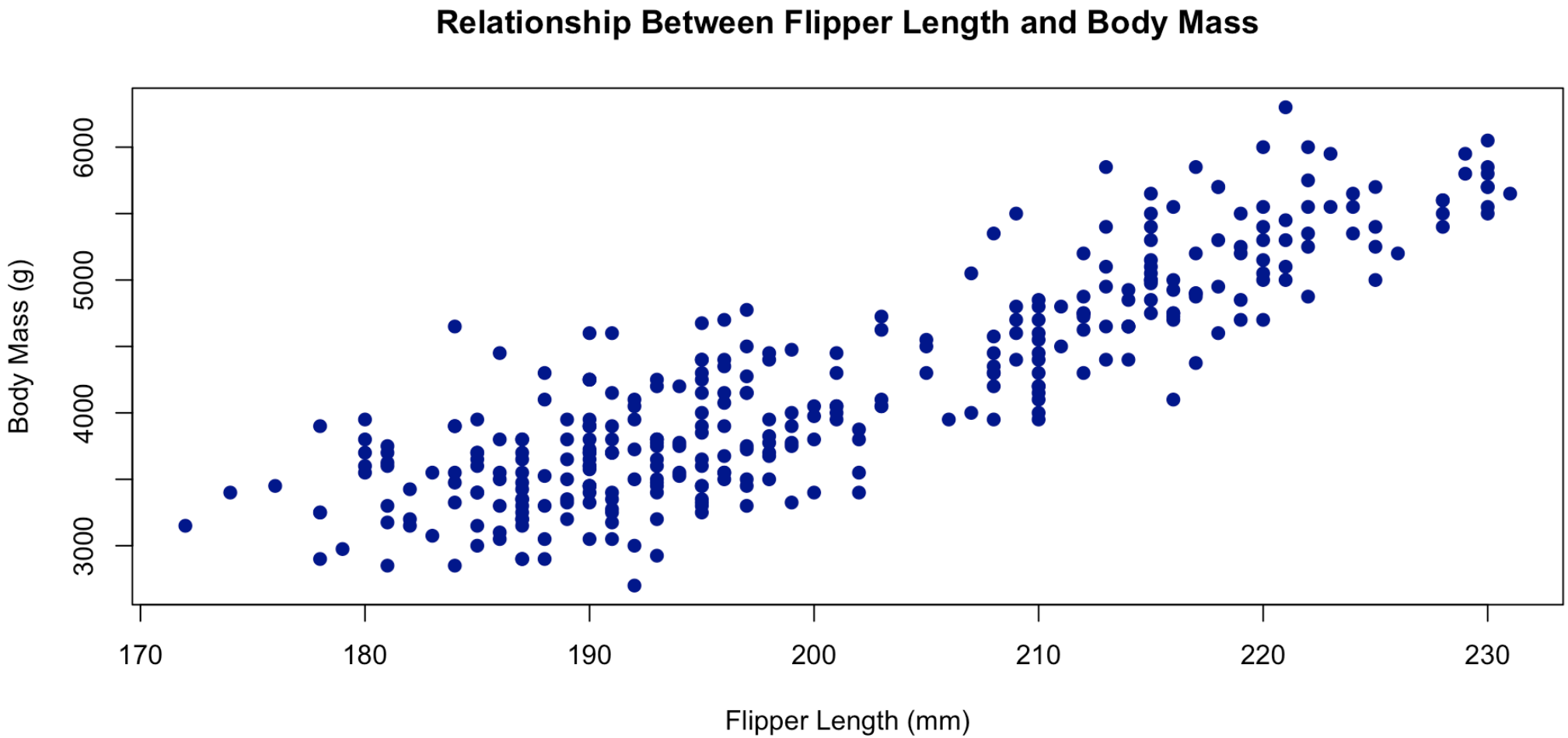
Purpose and applications:

- Examine relationships between continuous variables
- Identify correlations, patterns, and outliers
- Explore relationships between measurements

When to use:

- When examining relationships between two continuous variables
- When looking for correlations or patterns
- Examples: height vs. weight, temperature vs. growth rate

▶ Code


Relationship Between Flipper Length and Body Mass
Body Mass (g)
3000
4000
5000
6000
170
180
190
200
210
220
230
Flipper Length (mm)

Boxplots

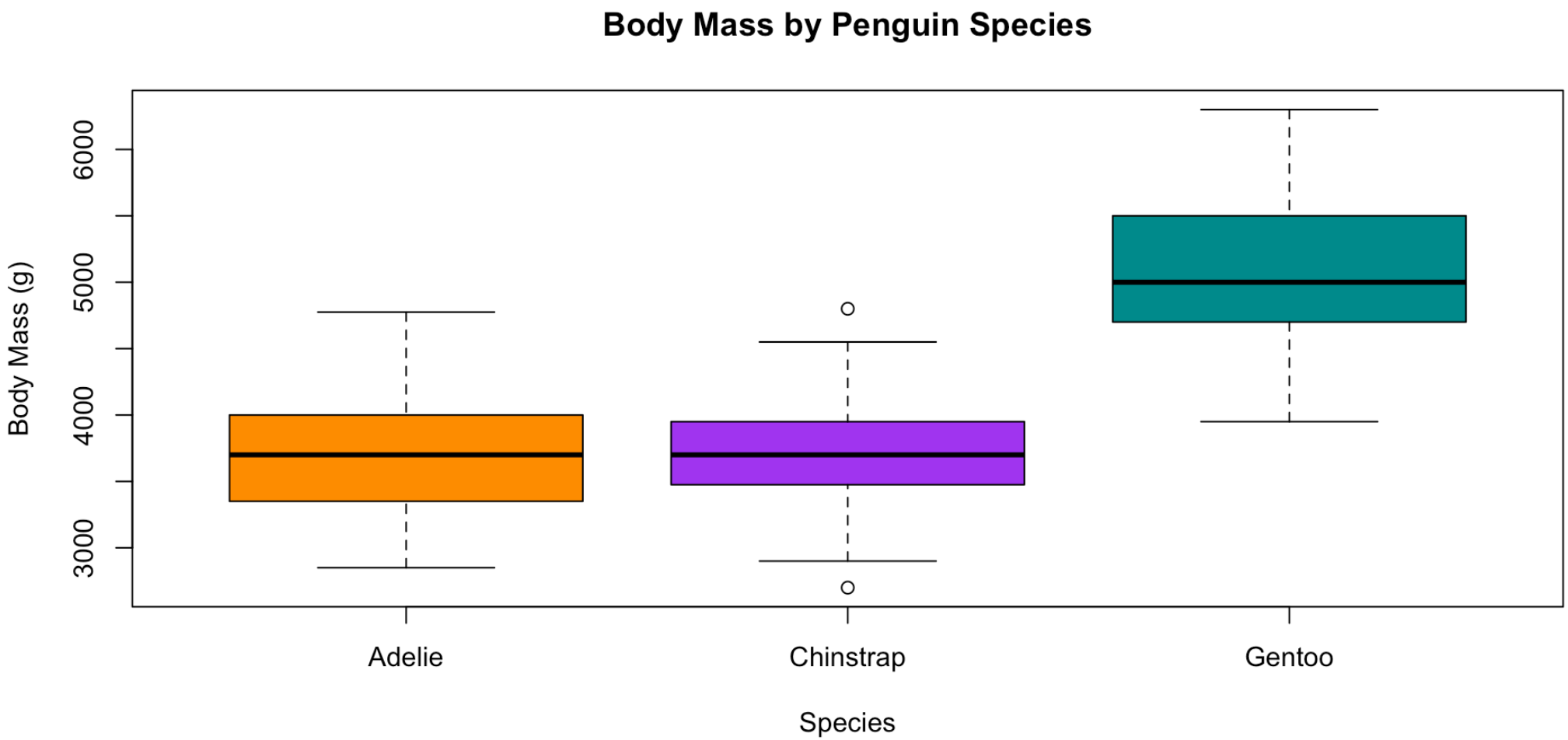
Purpose and applications:

- Compare distributions across groups
- Visualise median, quartiles, and outliers
- Compare measurements across treatments

When to use:

- When comparing a continuous variable across categorical groups
- When you need to show the spread and central tendency
- Examples: comparing heights across species, measurements across treatments

▶ Code


Body Mass by Penguin Species
Body Mass (g)
3000
4000
5000
6000
Adelie
Chinstrap
Gentoo
Species

Introduction to ggplot2

The Grammar of Graphics

ggplot2 is based on the Grammar of Graphics, a systematic approach to creating visualisations by combining different components:

1. **Data**: The dataset you want to visualise
2. **Aesthetics**: Mapping variables to visual properties (position, colour, size, etc.)
3. **Geometries**: The shapes used to represent the data (points, lines, bars, etc.)
4. **Scales**: How values are mapped to visual properties
5. **Facets**: How to split the data into subplots
6. **Coordinates**: The coordinate system to use
7. **Themes**: Visual styling of the plot

This grammar allows you to build complex visualisations layer by layer.

Why use ggplot2?

- **Consistent syntax** across different plot types
- **Layered approach** makes it easy to build complex visualisations
- **Excellent defaults** that produce publication-quality graphics
- **Highly customisable** with extensive options for fine-tuning
- **Large community** with extensive documentation and examples

Building a plot: Step 1 - Start with data

Let's build a scatterplot of penguin flipper length vs. body mass using the palmerpenguins dataset.

First, we need to load the ggplot2 package and prepare our data:

▶ Code

```
# A tibble: 6 × 3
  species flipper_length_mm body_mass_g
  <fct>               <int>       <int>
1 Adelie                181        3750
2 Adelie                186        3800
3 Adelie                195        3250
4 Adelie                193        3450
5 Adelie                190        3650
6 Adelie                181        3625
```

Building a plot: Step 2 - Create a blank canvas

The `ggplot()` function initialises a plot with data:

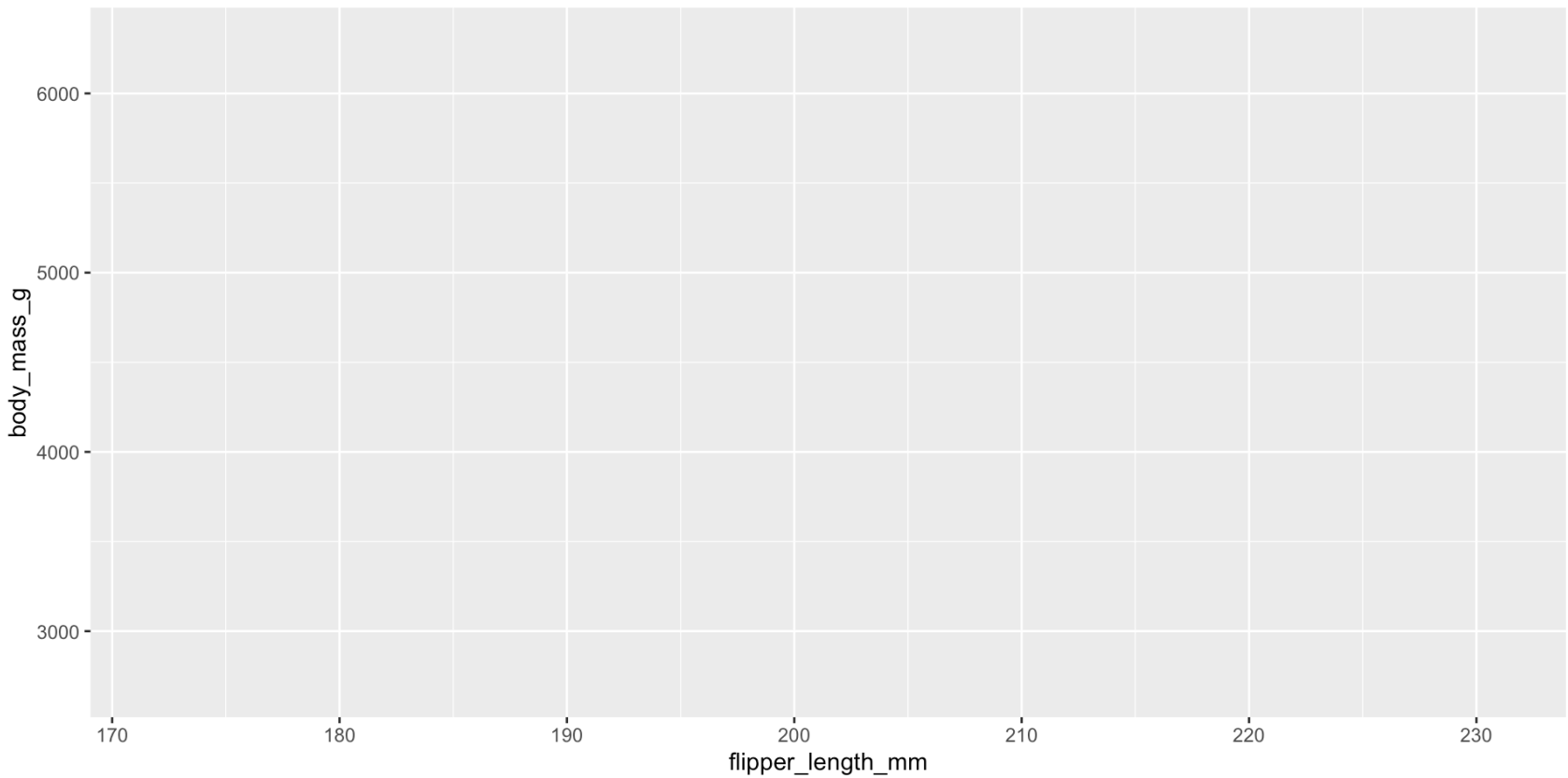
▶ Code

This creates an empty plot. We need to add layers to visualise our data.

Building a plot: Step 3 - Add aesthetics

Aesthetics map variables in the data to visual properties:

▶ Code

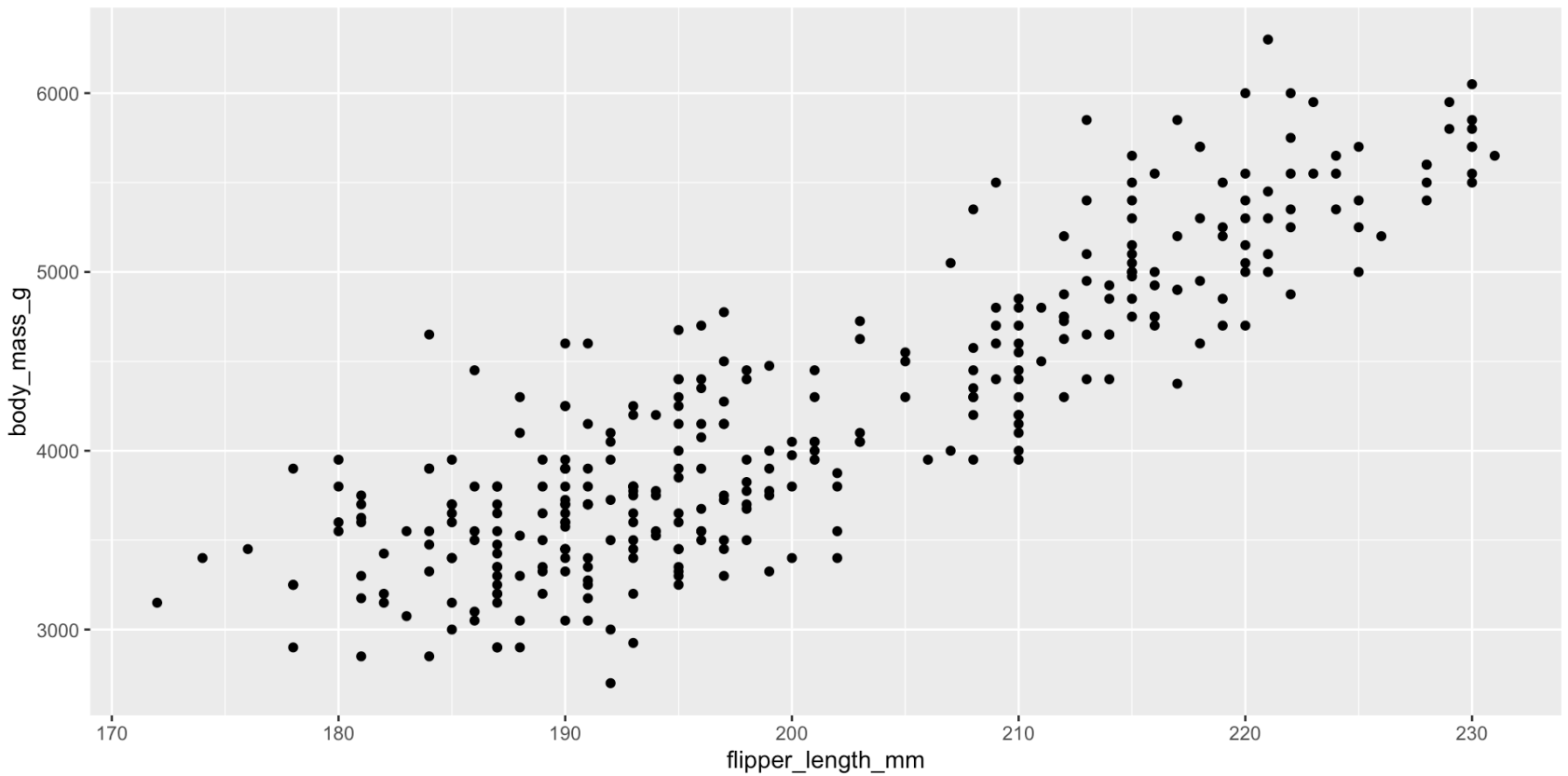


We've defined which variables go on which axes, but we still need to specify how to represent the data.

Building a plot: Step 4 - Add a geometry

Geometries define how the data is represented visually:

▶ Code

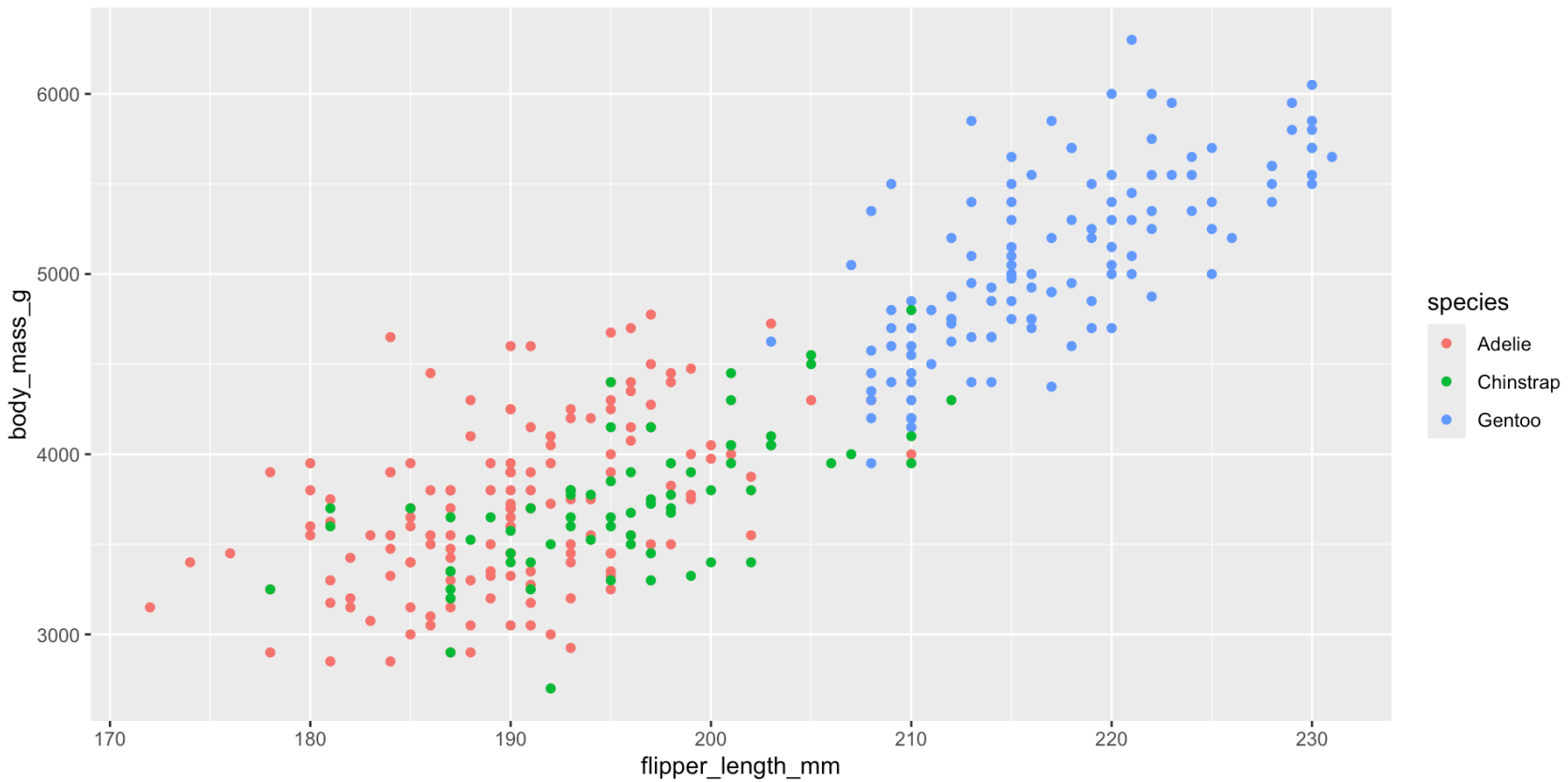


Now we can see the relationship between flipper length and body mass!

Building a plot: Step 5 - Add colour by species

We can map the species variable to the colour aesthetic:

▶ Code

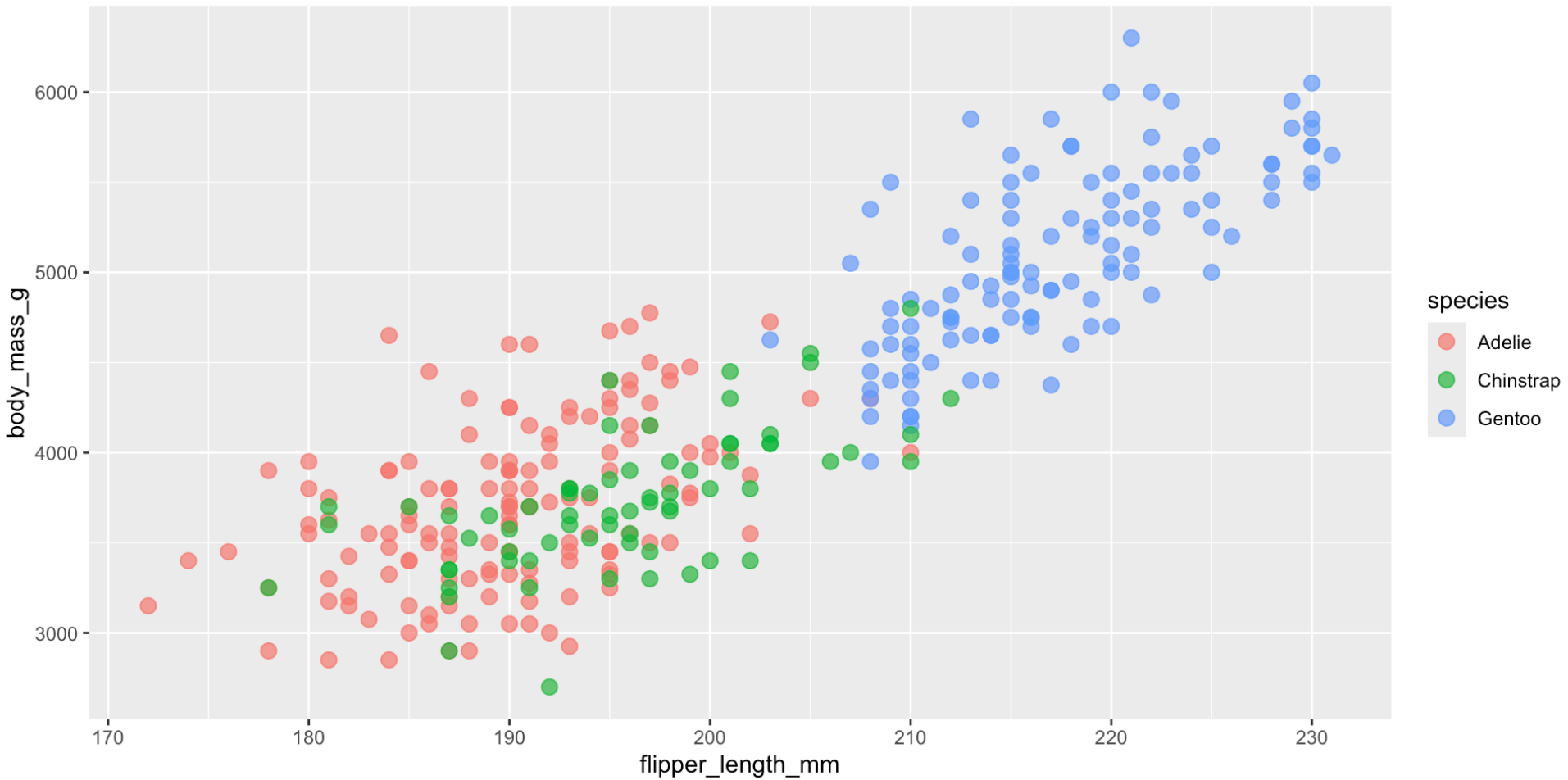


Notice how ggplot2 automatically creates a legend for the species colours.

Building a plot: Step 6 - Customise point appearance

We can adjust the size and transparency of points:

▶ Code



Building a plot: Step 7 - Add labels and title

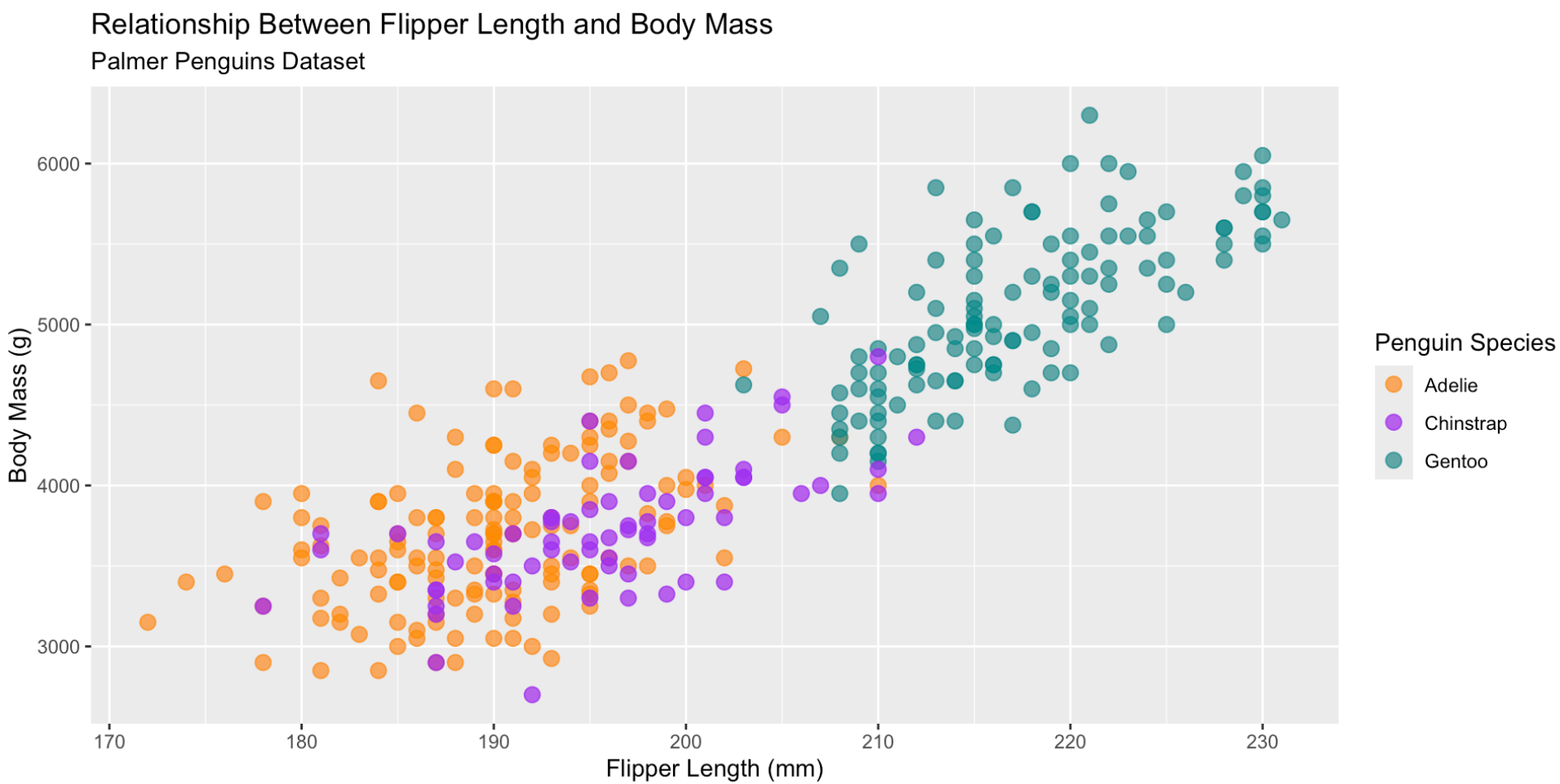
Let's add informative labels and a title:

▶ Code

Building a plot: Step 8 - Customise colours

We can use a custom colour palette:

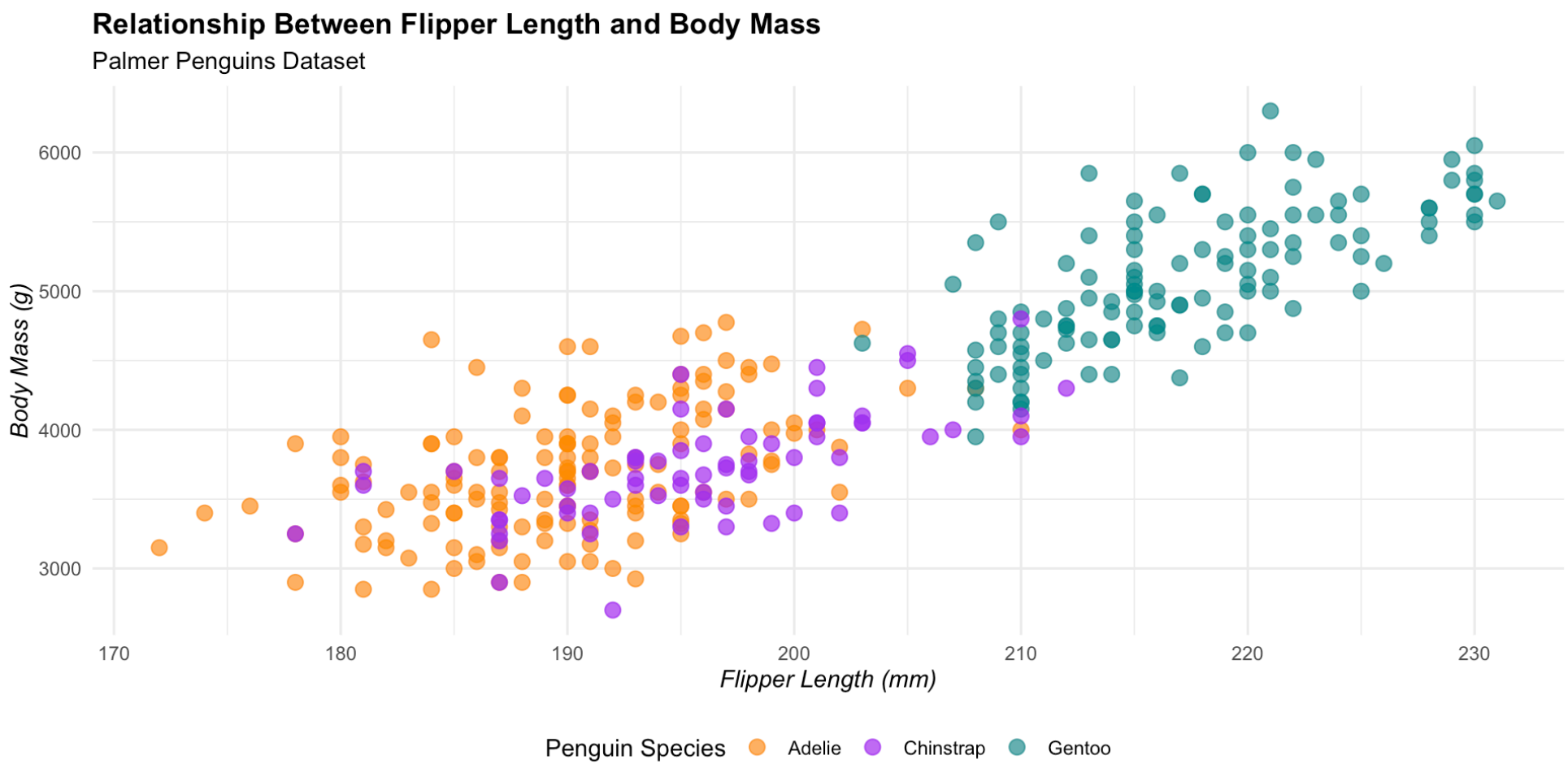
▶ Code



Building a plot: Step 9 - Apply a theme

Finally, let's apply a theme to change the overall appearance:

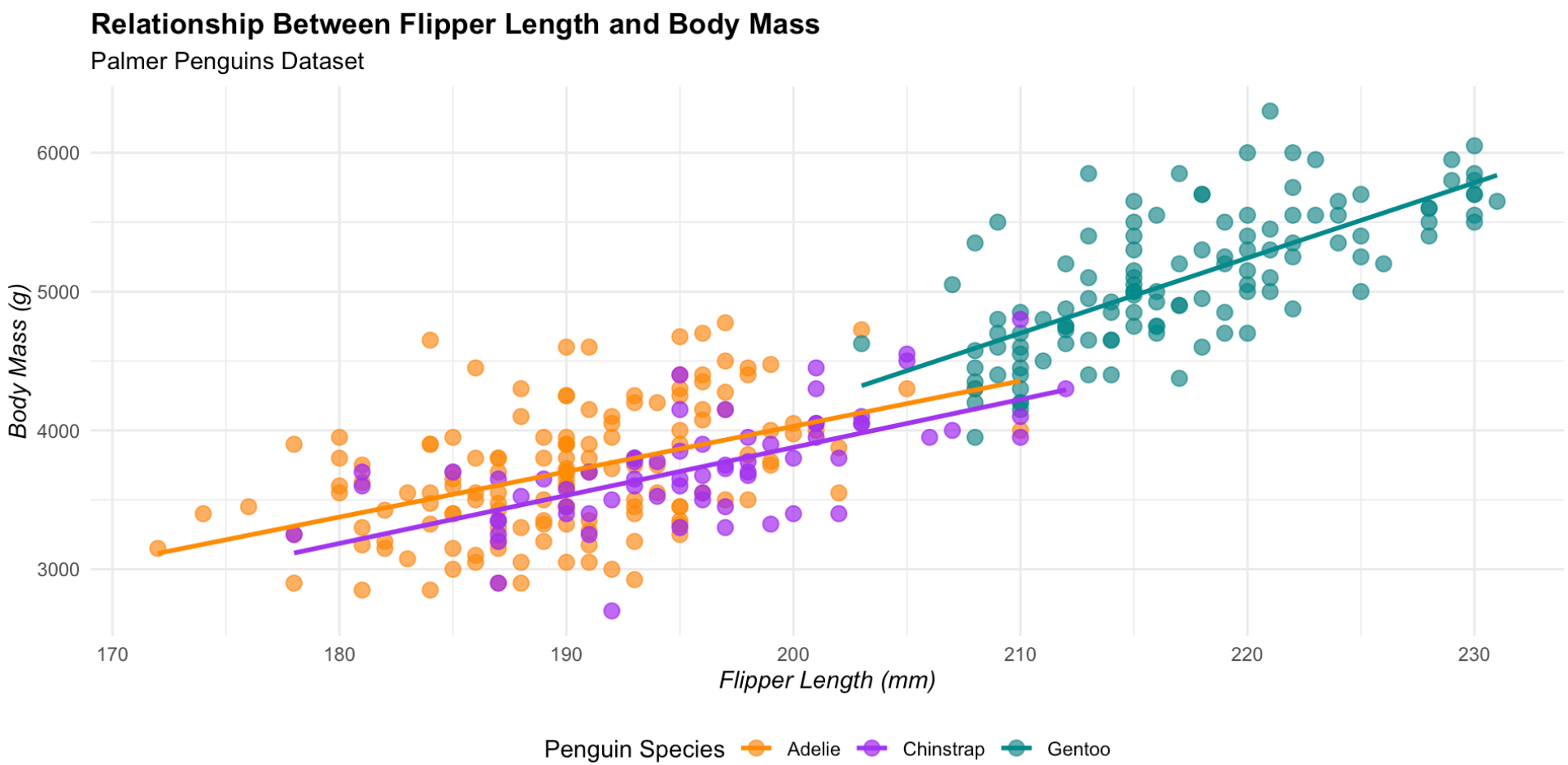
▶ Code



Adding more layers: Trend lines

One of the strengths of ggplot2 is the ability to add multiple layers:

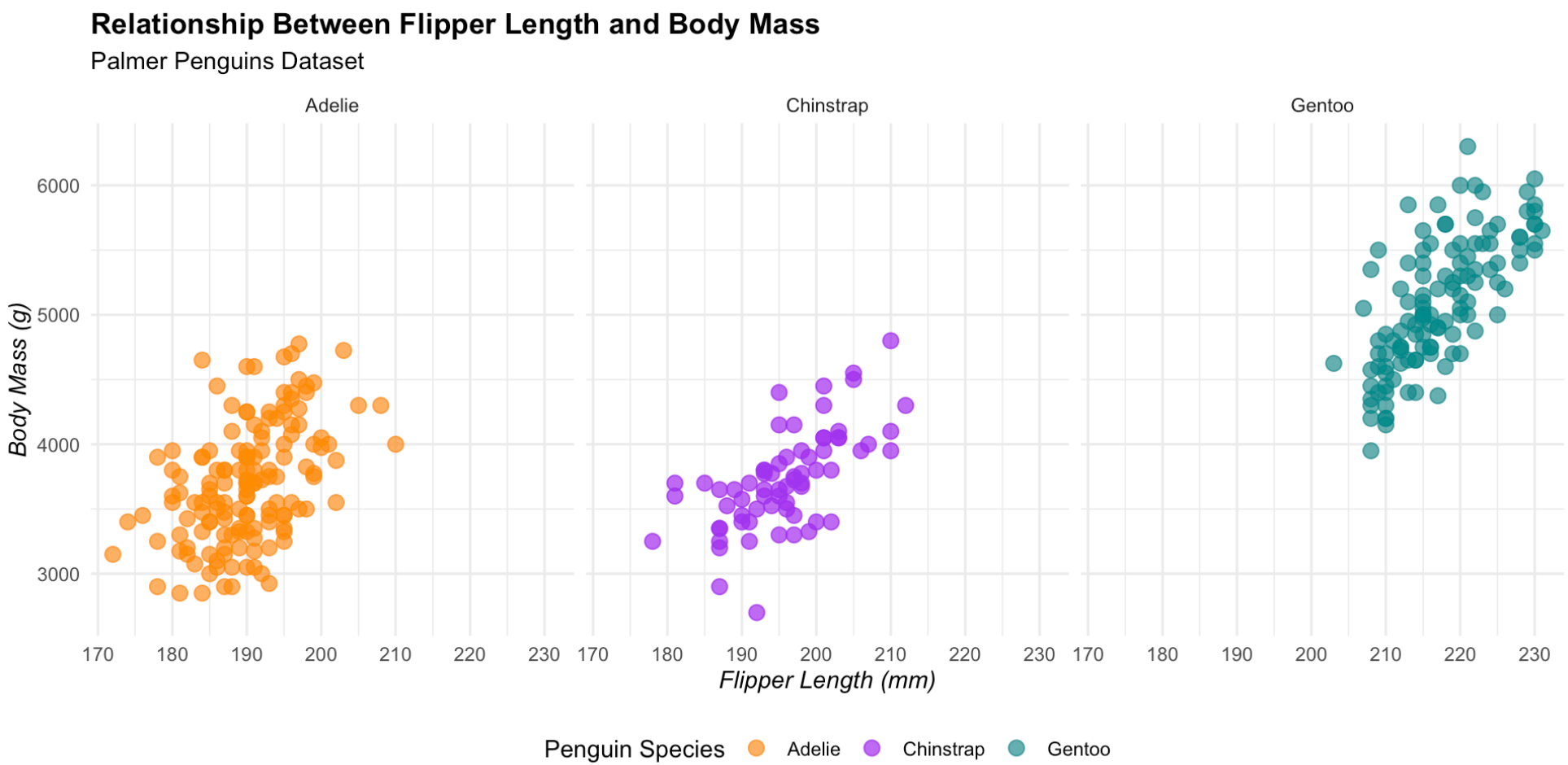
▶ Code



Faceting: Split by species

We can also split the plot into facets by species:

▶ Code



The complete ggplot2 code

Here's the complete code for our final plot:

▶ Code

Resources for further learning

- R Graphics Cookbook
- ggplot2 documentation
- R for Data Science

References and resources

Core reading

- Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer.
- Chang, W. (2018). *R Graphics Cookbook*. O'Reilly Media.
- Wickham, H., & Grolemund, G. (2017). *R for Data Science*. O'Reilly Media.

Online resources

- ggplot2 documentation
- R for Data Science - Data Visualization chapter
- The R Graph Gallery
- Cookbook for R - Graphs

Thanks!

This presentation is based on the SOLES Quarto reveal.js template and is licensed under a Creative Commons Attribution 4.0 International License.