Plotting in R Programming

A gentle Introduction to Basic Plots and ggplot

David Raj M

Contents

1 Introduction

Data visualization is a **core component of statistical analysis**. Before performing formal inference or modeling, visual exploration helps us:

- Understand the structure of the data
- Detect outliers and anomalies
- Check model assumptions
- Communicate results effectively

In this session, we learn **plotting in R from fundamentals to advanced statistical graphics**, using both **Base R** and **ggplot2**.

2 Base R Graphics

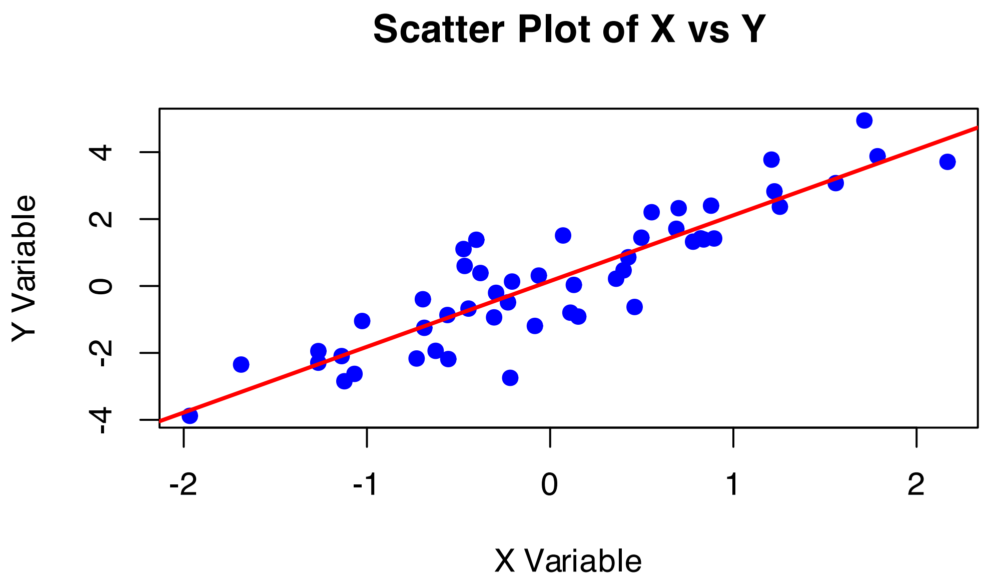
Base R graphics follow a **procedural approach**, where plots are built step by step. This system is simple, lightweight, and excellent for understanding the mechanics of plotting.

2.1 Scatter Plot

Scatter plots are used to visualize the relationship between two numerical variables.

```r
set.seed(123)
x = rnorm(50)
y = 2*x + rnorm(50)

plot(x, y,
     main = "Scatter Plot of X vs Y",
     xlab = "X Variable",
     ylab = "Y Variable",
     pch = 19,
     col =  "blue")
abline(lm(y ~ x), col = "red", lwd = 2)
```

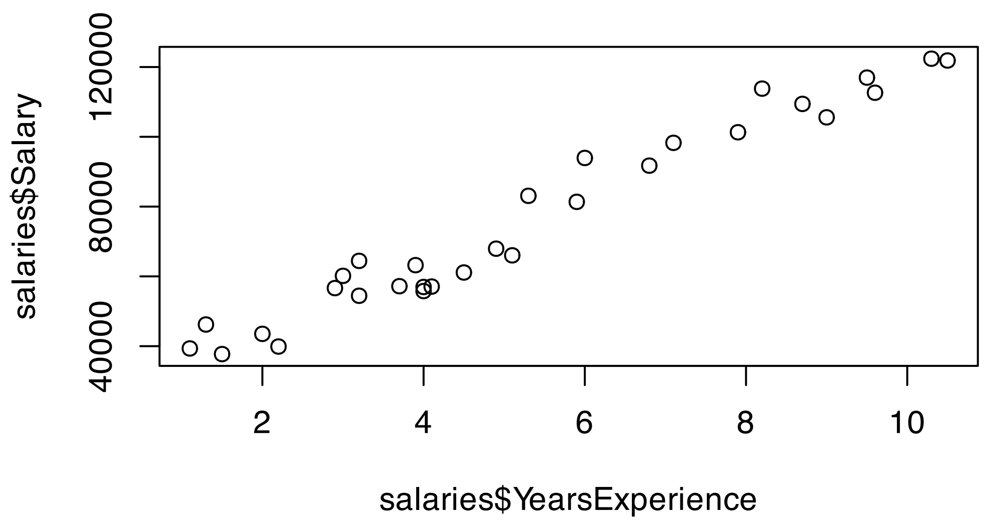


Example: Load the `Salary_Data.csv` and plot the `experience vs salary`.

```r
library(readr)
salaries = read.csv('./data/Salary_Data.csv')
head(salaries)
```

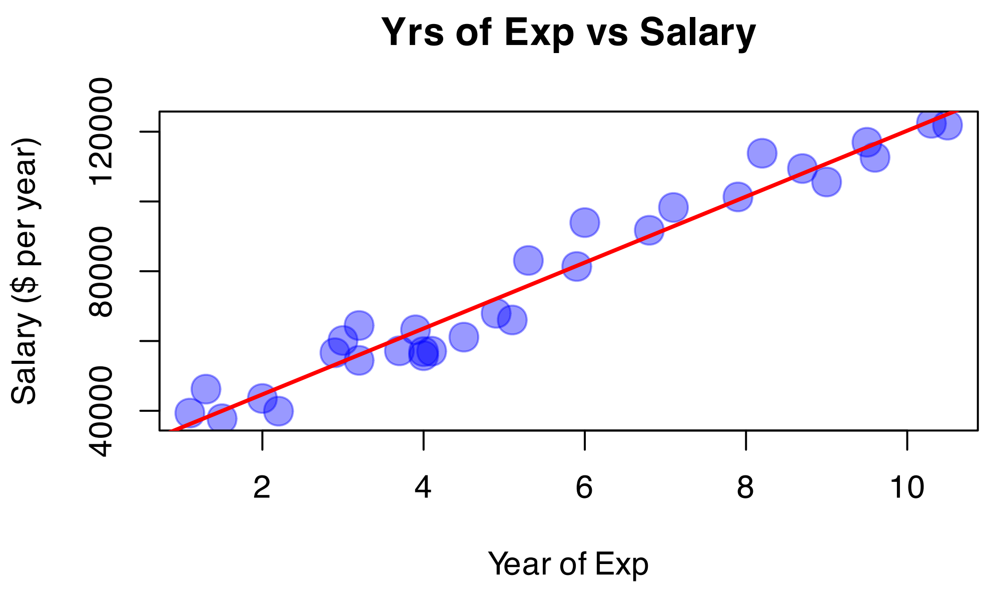
```
  YearsExperience Salary
1             1.1  39343
2             1.3  46205
3             1.5  37731
4             2.0  43525
5             2.2  39891
6             2.9  56642
```

```
plot(salaries$YearsExperience, salaries$Salary)
```



Note that, there is a positive correlation between the experience of the person and the salary that they receive.

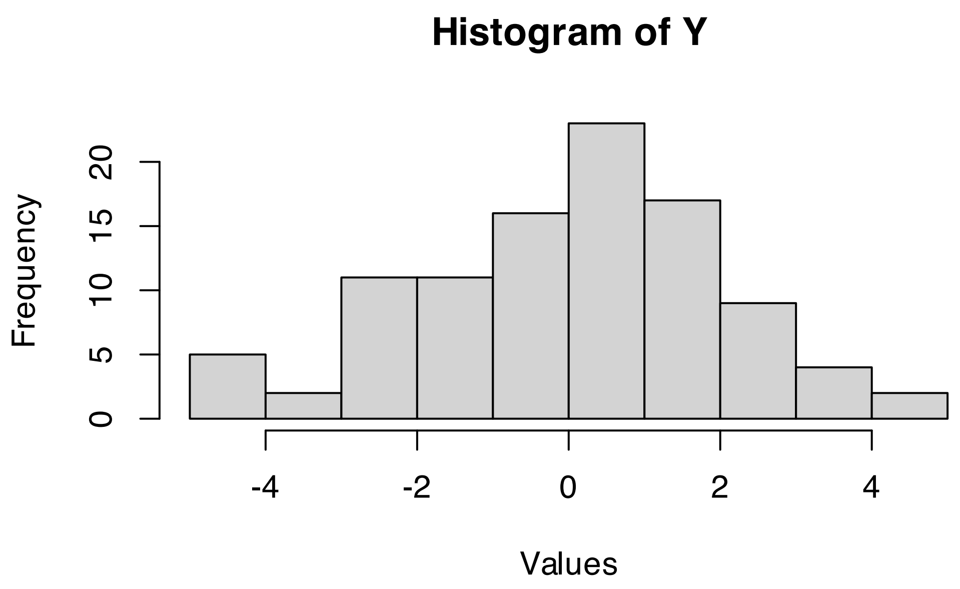
```
plot(salaries$YearsExperience, salaries$Salary,
     xlab = "Year of Exp",
     ylab = "Salary ($ per year)",
     pch = 19,
     col = adjustcolor("blue", alpha.f = 0.4),
     cex = 2,
     main = "Yrs of Exp vs Salary"
)
abline(lm(salaries$Salary ~ salaries$YearsExperience), col = "red", lwd = 2)
```



2.2 Histograms

Histograms show the distribution of a numerical variable.

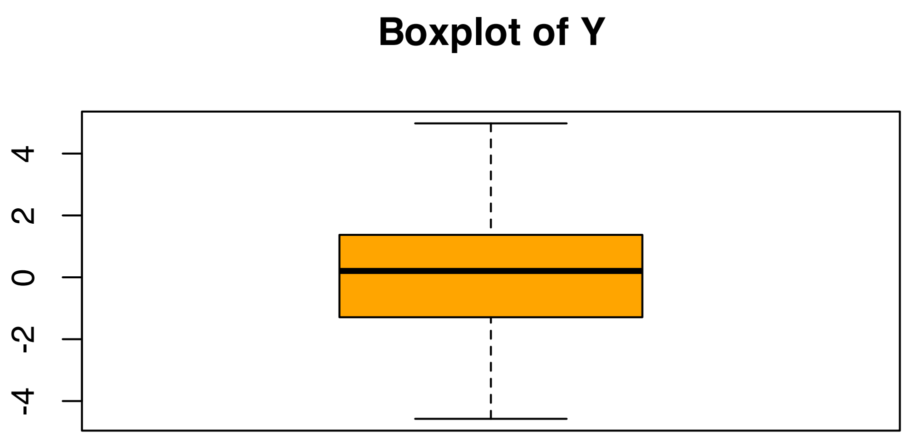
```r
set.seed(123)
x = rnorm(100)
y = 2*x + rnorm(100)
hist(y,
     breaks = 10,
     col = "lightgray",
     main = "Histogram of Y",
     xlab = "Values")
```



2.3 Boxplots

Boxplots provide a compact summary of distribution.

```r
boxplot(y,
        col = "orange",
        main = "Boxplot of Y")
```

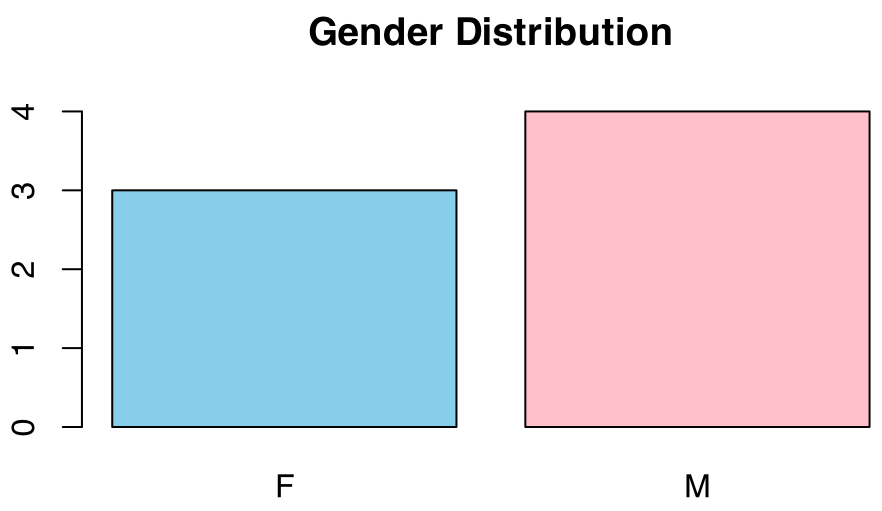


2.4 Bar Charts (Categorical Data)

```
gender <- c("M", "F", "M", "F", "F", "M","M")
gender_data = table(gender)
gender_data
```

```
gender
F M
3 4
```

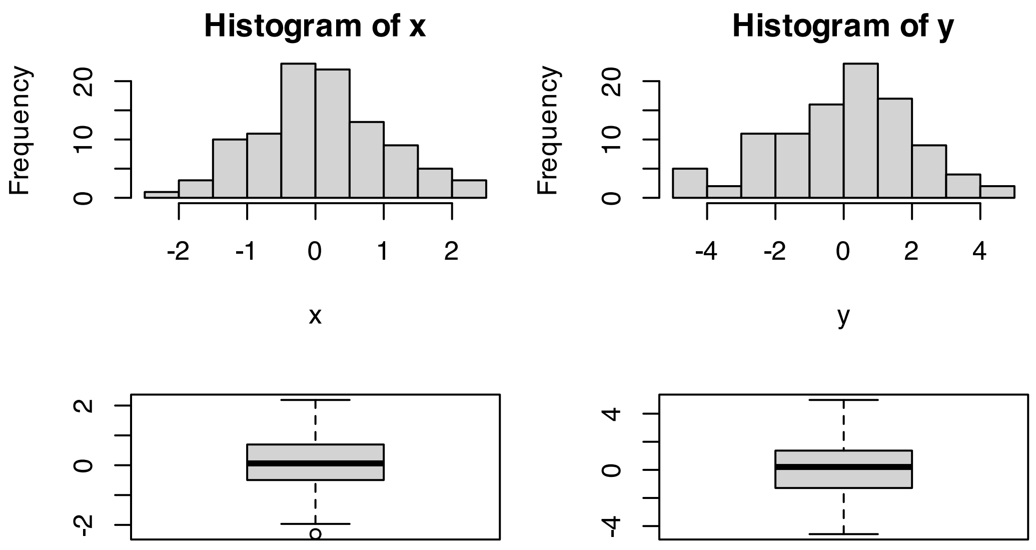
```
barplot(gender_data,
        col = c("skyblue", "pink"),
        main = "Gender Distribution")
```



Bar charts are used for **categorical variables**, unlike histograms which are for numerical data.

2.5 Multiple Plots in One Window

```r
par(mfrow = c(2,2),mar = c(4,4,2,1))
hist(x)
hist(y)
boxplot(x)
boxplot(y)
```



```r
dev.off()
```

```
null device 
          1 
```

This layout is useful for **diagnostic analysis**.

2.6 Exporting Plots

```r
png("scatter_plot.png", width = 800, height = 600)
plot(x, y)
dev.off()
```

```
svg 
  2 
```

Common formats:

- PNG (presentations)
- PDF (journals)
- SVG (best for vector supported documents, such latex, typst, html)

3 Grammar of Graphics - ggplot2

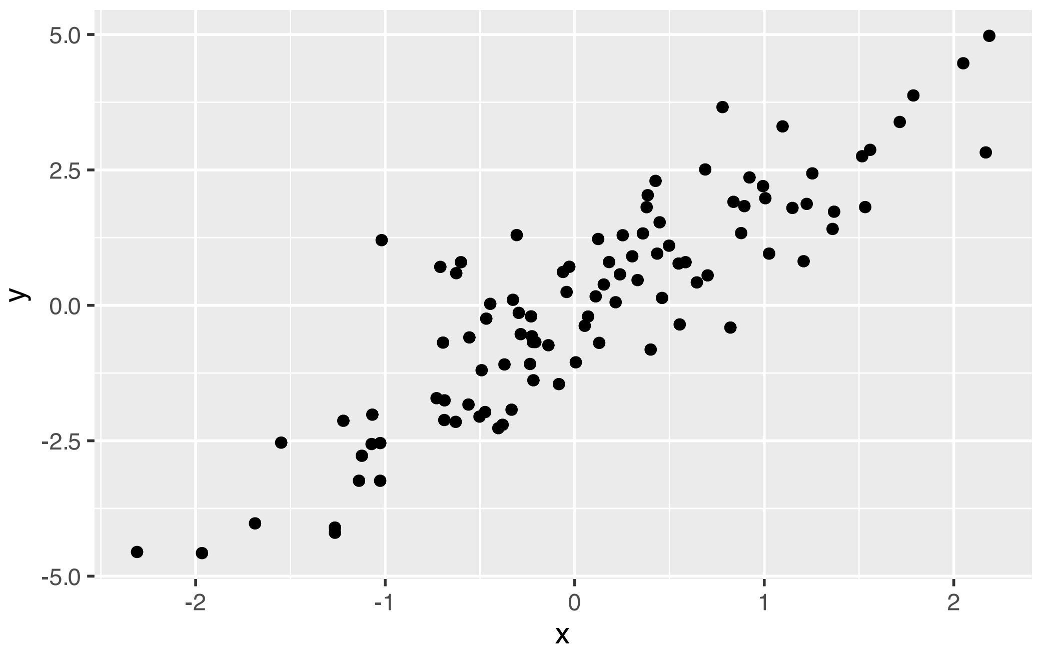
The **ggplot2** package follows the *Grammar of Graphics*, enabling structured and layered visualizations. With ggplot2, you can do more and faster by learning one system and applying it in many places.

```r
library(ggplot2)

df <- data.frame(x = x, y = y)
```

Basic ggplot Structure

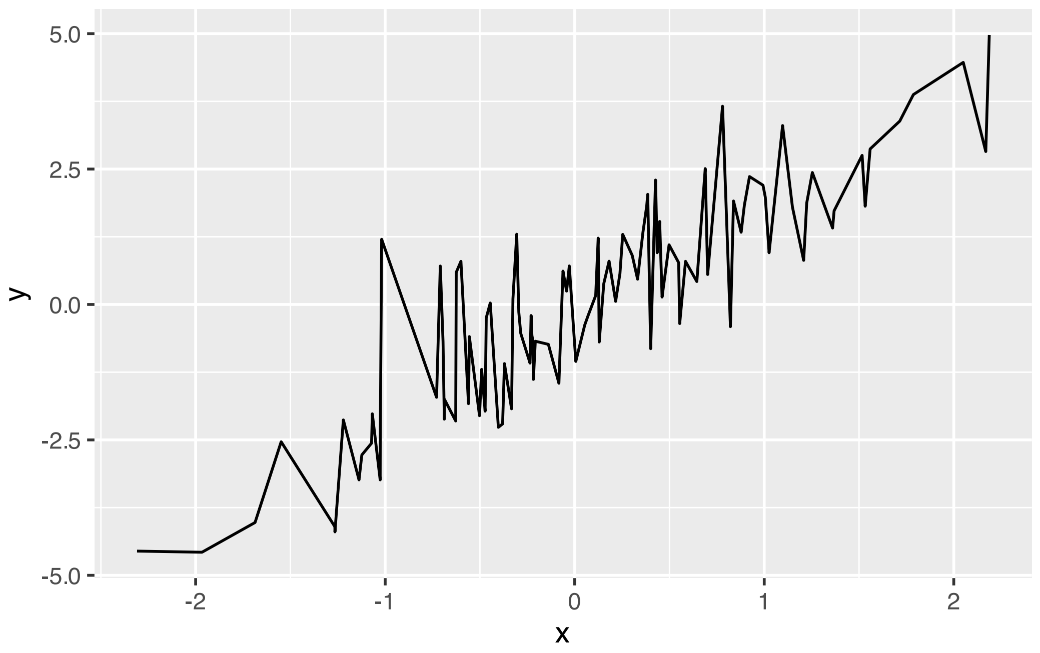
```r
ggplot(df, aes(x, y)) +
  geom_point()
```



Here:

- `data` defines the dataset
- `aes()` maps variables to aesthetics
- `geom_*()` defines the plot type

```r
ggplot(df, aes(x, y)) +
  geom_line()
```



In recent years, **`ggplot2`** has become a core package within the **`tidyverse`** ecosystem. The `tidyverse` is a collection of powerful and user-friendly R packages, including `dplyr`, `tidyr`, `readr`, `forcats`, and `lubridate`, all designed to work together seamlessly.

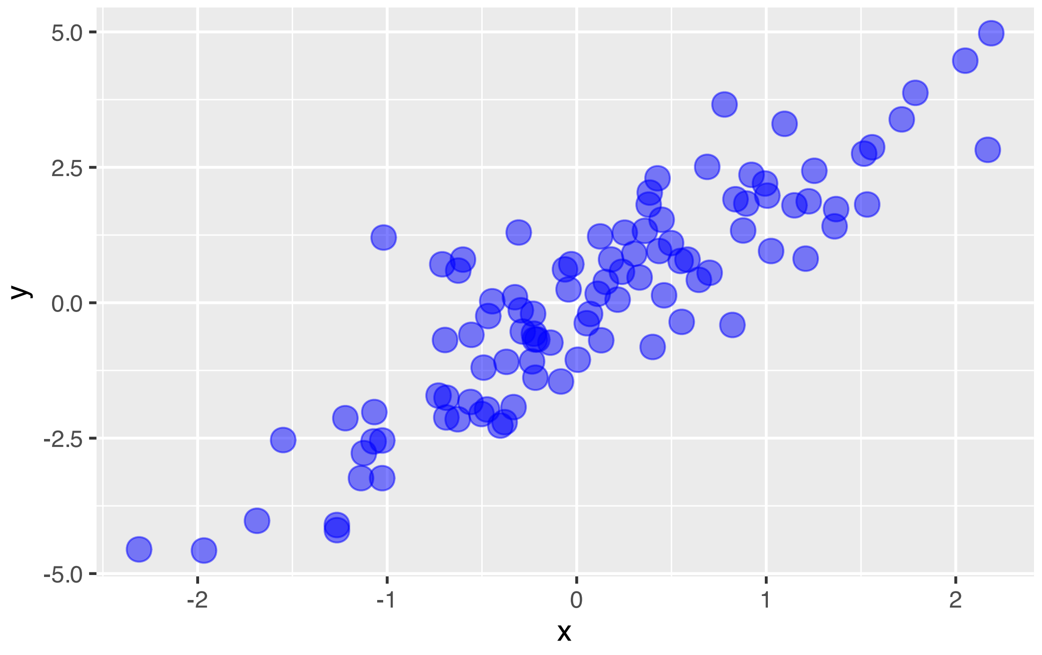
By installing and loading the `tidyverse`, we can take advantage of a consistent syntax, **pipe-based workflows**, efficient data import, and systematic data tidying, which greatly simplify data analysis and visualization.

3.1 Enhanced Scatter Plot

```r
# install.packages("tidyverse")
library(tidyverse)
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0
──
✔ dplyr     1.1.4     ✔ stringr   1.6.0
✔ forcats   1.0.1     ✔ tibble    3.3.1
✔ lubridate 1.9.4     ✔ tidyr     1.3.2
✔ purrr     1.2.1
── Conflicts ──────────────────────────────────────────── tidyverse_conflicts()
──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
```

```r
ggplot(df, aes(x, y)) +
  geom_point(color = "blue", size = 4, alpha=.5)
```



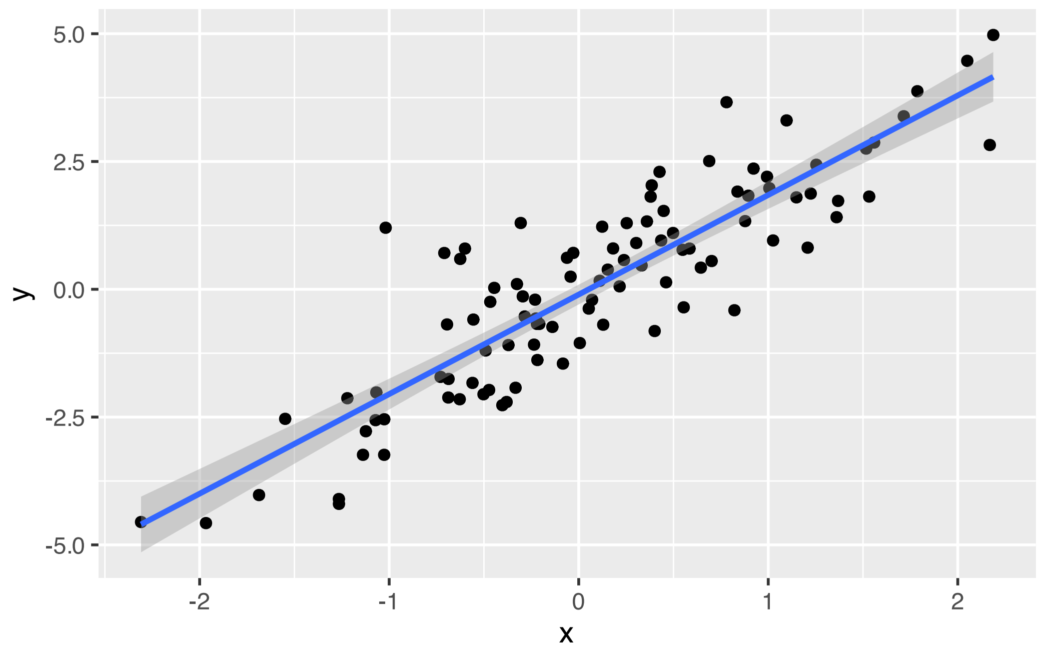
```r
  labs(title = "Scatter Plot",
       x = "X Variable",
       y = "Y Variable") +
  theme_bw()
```

```
NULL
```

Adding Regression Line with Confidence Interval

```r
ggplot(df, aes(x, y)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE)
```

```
`geom_smooth()` using formula = 'y ~ x'
```



This plot visually represents:

- Estimated regression line
- 95% confidence band

Example: In addition to the `tidyverse`, we will also use the `palmerpenguins` package, which provides the `penguins` dataset. This dataset contains body measurements of penguins collected from three islands in the Palmer Archipelago, Antarctica.

```r
# install.packages("palmerpenguins")
library(palmerpenguins)
```

```


Attaching package: 'palmerpenguins'
```

```
The following objects are masked from 'package:datasets':

    penguins, penguins_raw
```

```r
head(penguins)
```

```
# A tibble: 6 × 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
# ℹ 2 more variables: sex <fct>, year <int>
```

The penguins data frame consists of 8 variables.
To view a compact summary of all variables, use `glimpse()`

```r
glimpse(penguins)
```

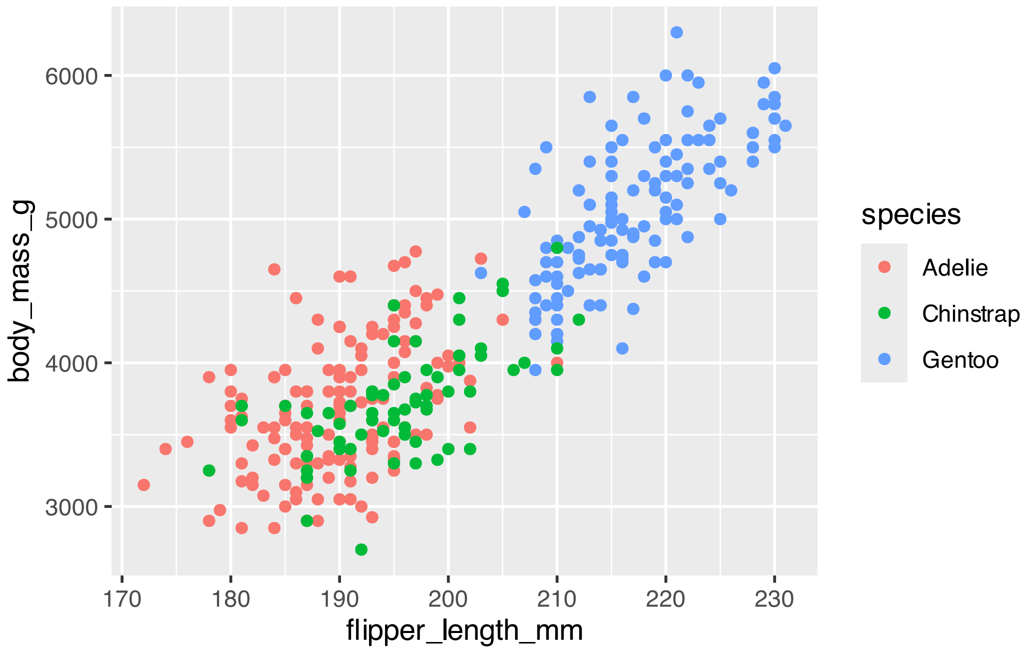
```
Rows: 344
Columns: 8
$ species           <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adelie,
Adel…
$ island            <fct> Torgersen, Torgersen, Torgersen, Torgersen,
Torgerse…
$ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.9, 39.2, 34.1,
…
$ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.8, 19.6, 18.1,
…
$ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195, 193, 190,
186…
$ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 3625, 4675, 3475,
…
$ sex               <fct> male, female, female, NA, female, male, female,
male…
$ year              <int> 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007,
2007…
```

Important variables include:

- `species`: Penguin species (Adelie, Chinstrap, Gentoo)
- `flipper_length_mm`: Flipper length in millimeters
- `body_mass_g`: Body mass in grams

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

```
Warning: Removed 2 rows containing missing values or values outside the scale
range
(`geom_point()`).
```



When a categorical variable is mapped to an aesthetic, `ggplot2` automatically assigns distinct colors to each category and adds a legend.

Now let's add one more layer: a smooth curve displaying the relationship between body mass and flipper length. Before you proceed, refer back to the code above, and think about how we can add this to our existing plot.

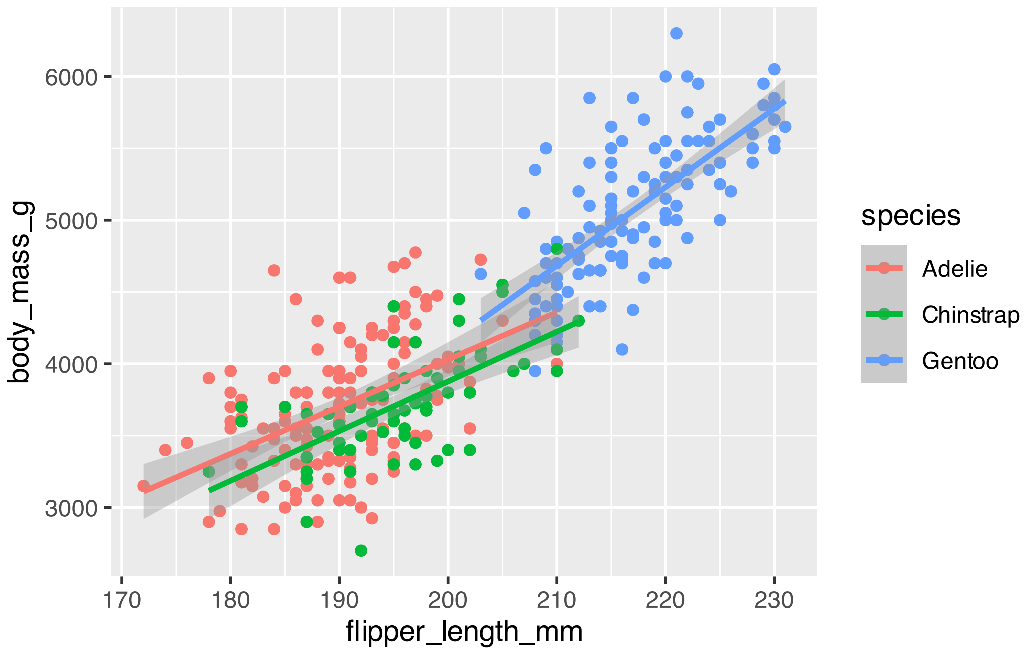
Since this is a new geometric object representing our data, we will add a new geom as a layer on top of our point geom: geom_smooth(). And we will specify that we want to draw the line of best fit based on a linear model with method = "lm".

```r
ggplot(
  data = penguins,
  mapping = aes(x = flipper_length_mm, y = body_mass_g, color = species)
) +
  geom_point() +
  geom_smooth(method = "lm")
```

```
`geom_smooth()` using formula = 'y ~ x'
```

```
Warning: Removed 2 rows containing non-finite outside the scale range
(`stat_smooth()`).
```

```
Warning: Removed 2 rows containing missing values or values outside the scale
range
(`geom_point()`).
```



Global vs Local Aesthetic Mappings

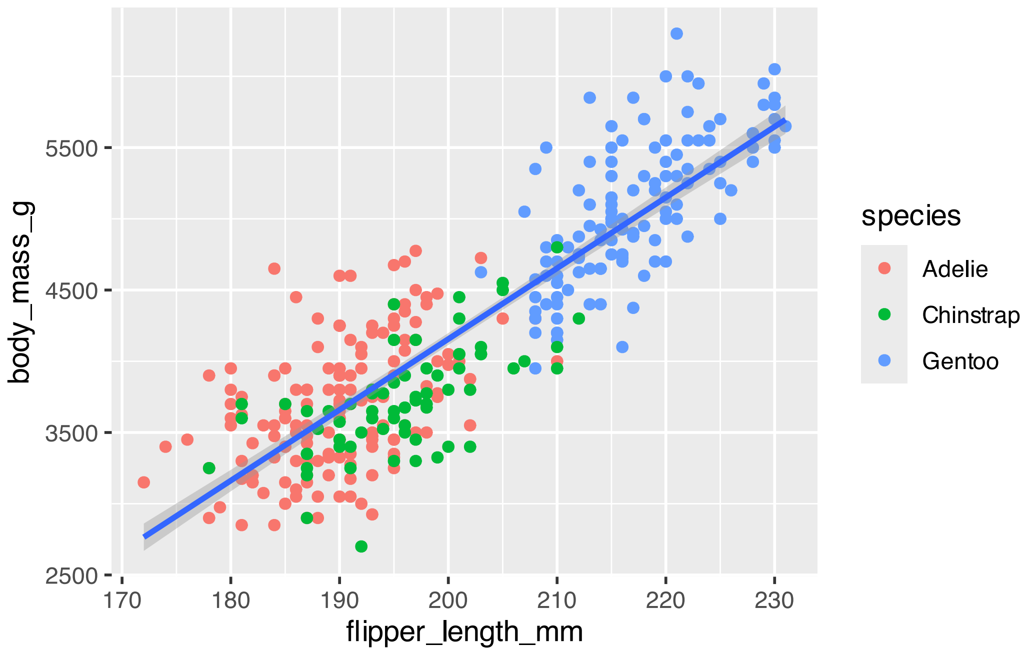
When aesthetic mappings are defined in ggplot(), at the global level, they're passed down to each of the subsequent geom layers of the plot. However, each geom function in ggplot2 can also take a mapping argument, which allows for aesthetic mappings at the local level that are added to those inherited from the global level. Since we want points to be colored based on species but don't want the lines to be separated out for them, we should specify color = species for geom_point() only.

```
ggplot(
  data = penguins,
  mapping = aes(x = flipper_length_mm, y = body_mass_g)
) +
  geom_point(mapping = aes(color = species)) +
  geom_smooth(method = "lm")
```

```
`geom_smooth()` using formula = 'y ~ x'
```

```
Warning: Removed 2 rows containing non-finite outside the scale range
(`stat_smooth()`).
```

```
Warning: Removed 2 rows containing missing values or values outside the scale
range
(`geom_point()`).
```



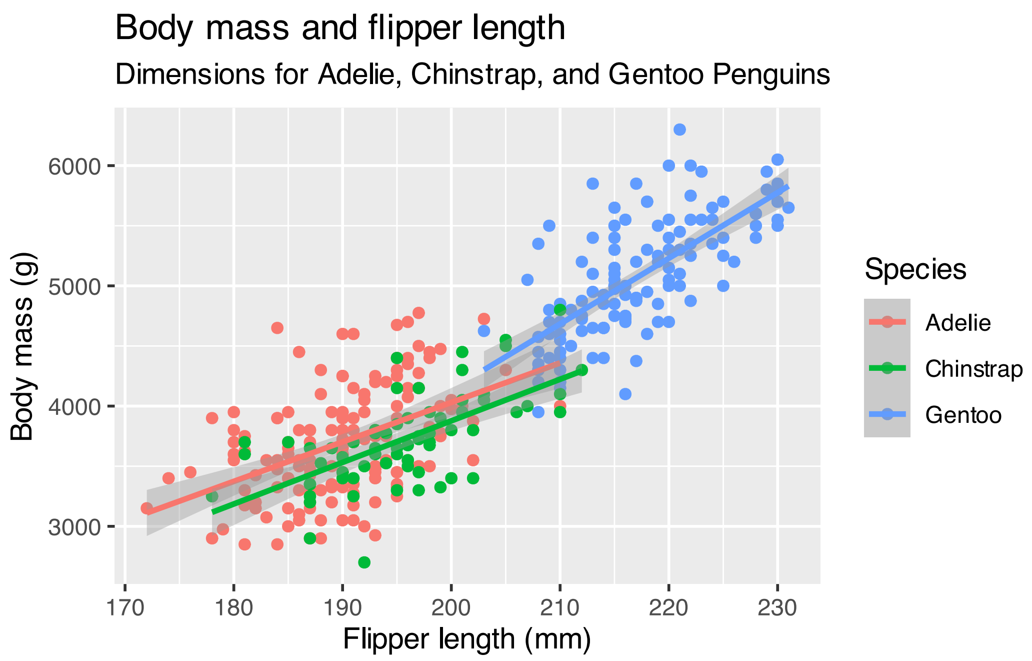
And, finally lets add some labels properly.

```
ggplot(
  data = penguins,
  mapping = aes(x = flipper_length_mm, y = body_mass_g, color = species)
) +
  geom_point() +
  geom_smooth(method = "lm")+
  labs(
    title = "Body mass and flipper length",
    subtitle = "Dimensions for Adelie, Chinstrap, and Gentoo Penguins",
    x = "Flipper length (mm)", y = "Body mass (g)",
    color = "Species", shape = "Species"
  )
```

```
Ignoring unknown labels:
• shape : "Species"
`geom_smooth()` using formula = 'y ~ x'
```

```
Warning: Removed 2 rows containing non-finite outside the scale range
(`stat_smooth()`).
```

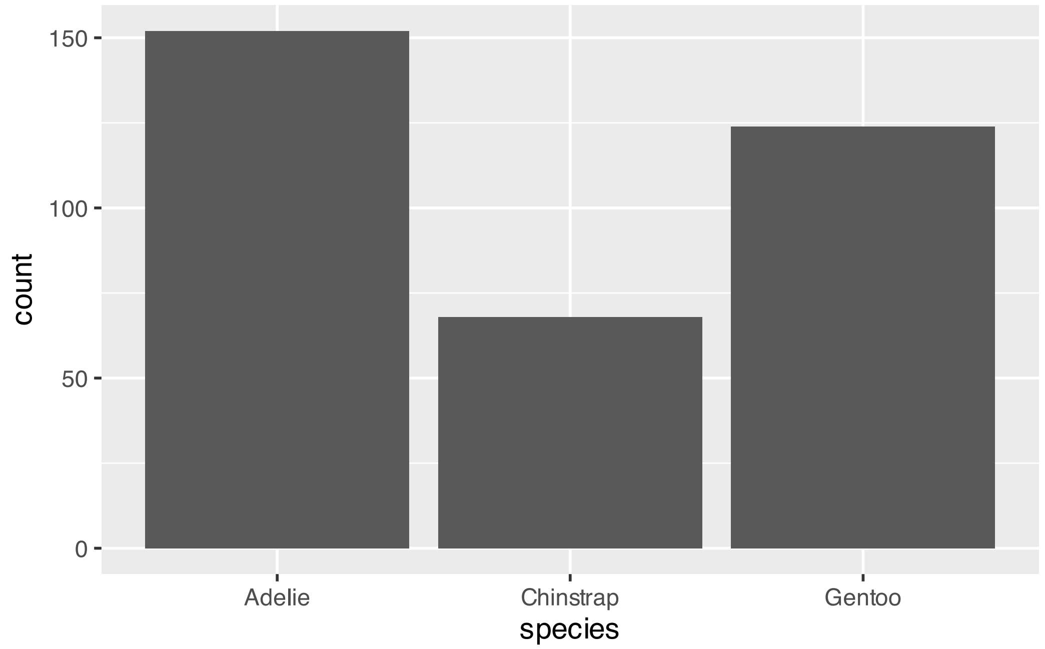
```
Warning: Removed 2 rows containing missing values or values outside the scale
range
(`geom_point()`).
```



3.2 Bar Charts

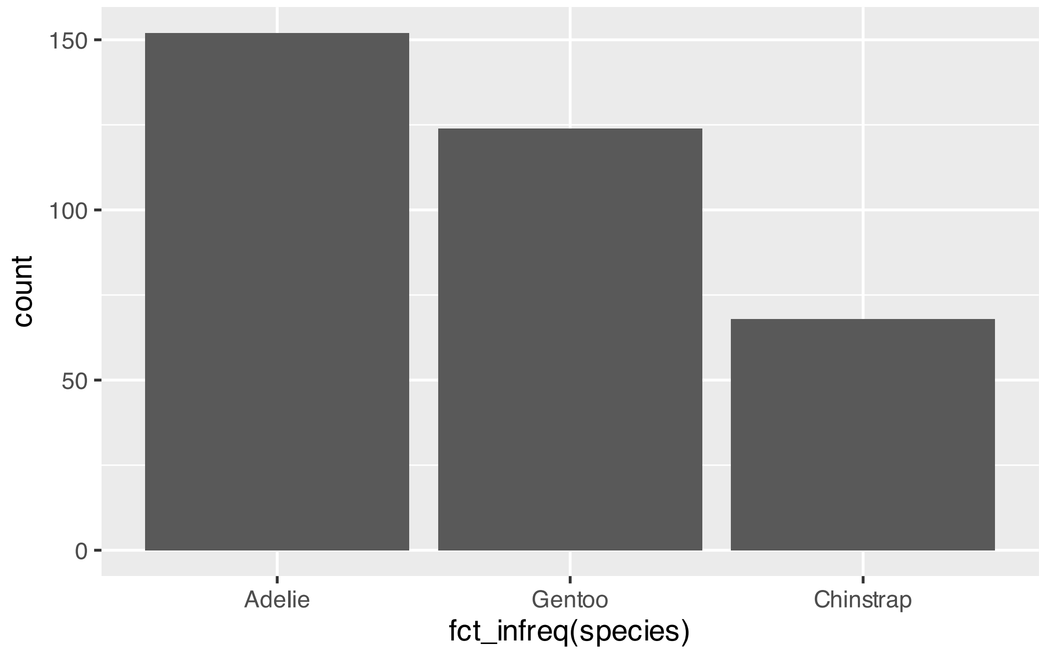
A variable is categorical if it can only take one of a small set of values. To examine the distribution of a categorical variable, you can use a bar chart. The height of the bars displays how many observations occurred with each x value.

```r
ggplot(penguins, aes(x = species)) +
  geom_bar()
```



In bar plots of categorical variables with non-ordered levels, like the penguin species above, it's often preferable to reorder the bars based on their frequencies. Doing so requires transforming the variable to a factor (how R handles categorical data) and then reordering the levels of that factor.

```r
ggplot(penguins, aes(x = fct_infreq(species))) +
  geom_bar()
```



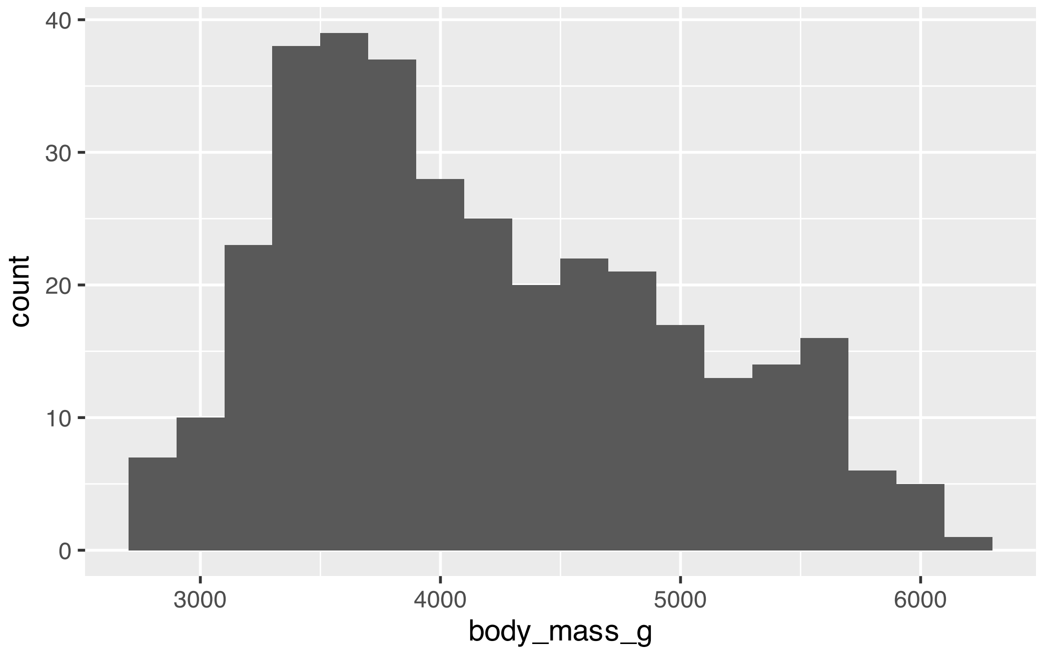
3.3 Histograms and Density Plots

A variable is numerical (or quantitative) if it can take on a wide range of numerical values, and it is sensible to add, subtract, or take averages with those values. Numerical variables can be continuous or discrete.

One commonly used visualization for distributions of continuous variables is a histogram.

```r
ggplot(penguins, aes(x = body_mass_g)) +
  geom_histogram(binwidth = 200)
```

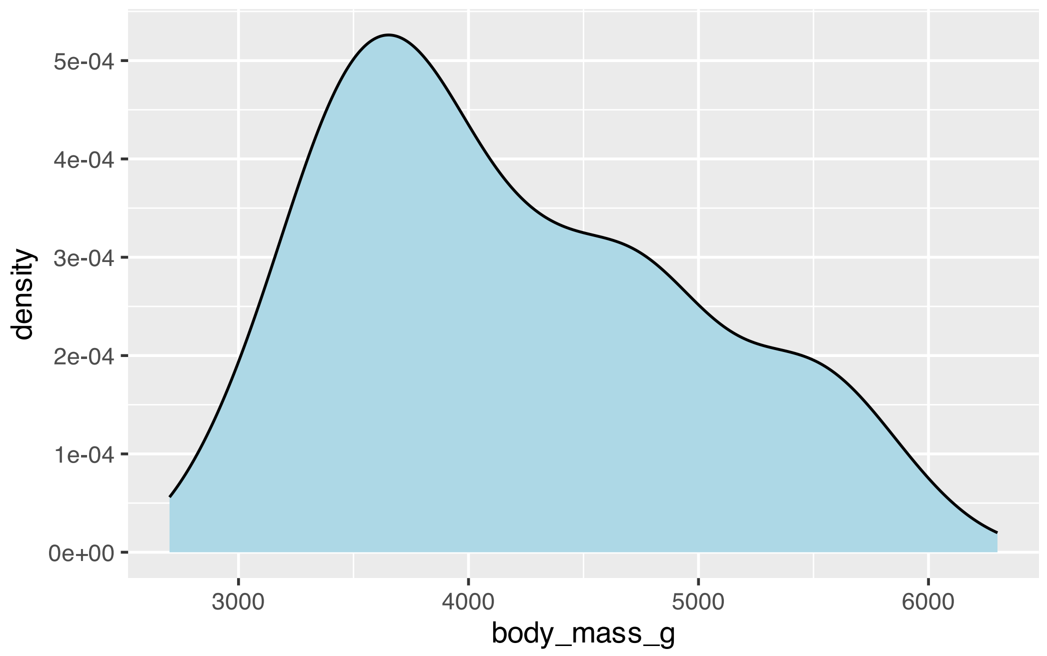
```
Warning: Removed 2 rows containing non-finite outside the scale range
(`stat_bin()`).
```



An alternative visualization for distributions of numerical variables is a density plot. A density plot is a smoothed-out version of a histogram and a practical alternative, particularly for continuous data that comes from an underlying smooth distribution.

```r
ggplot(penguins, aes(x = body_mass_g)) +
  geom_density(fill = "lightblue")
```

```
Warning: Removed 2 rows containing non-finite outside the scale range
(`stat_density()`).
```



3.4 Boxplots

To visualize the relationship between a numerical and a categorical variable we can use side-by-side box plots. A boxplot is a type of visual shorthand for measures of position (percentiles) that describe a distribution. It is also useful for identifying potential outliers.

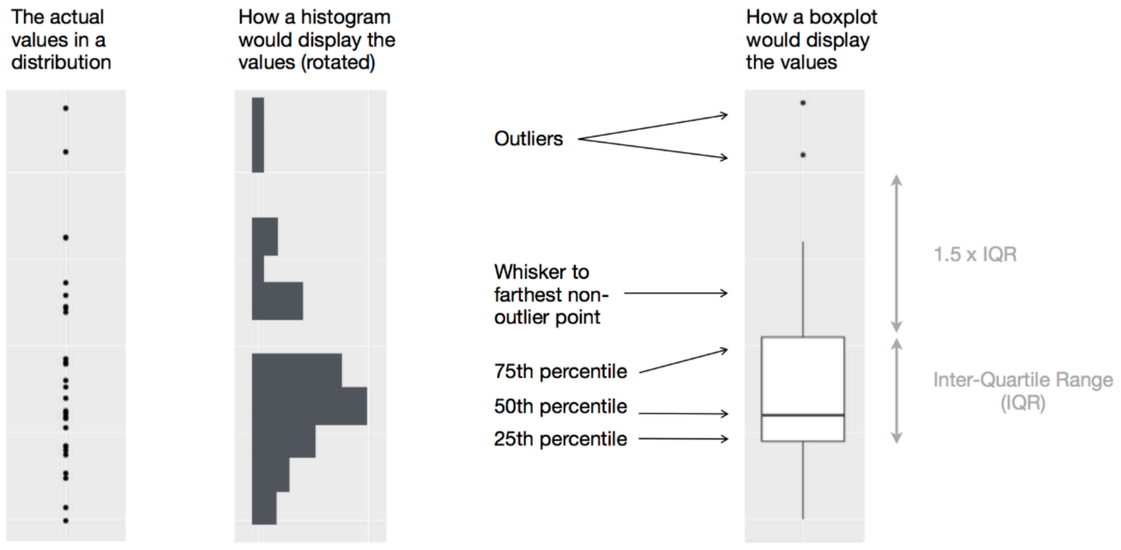
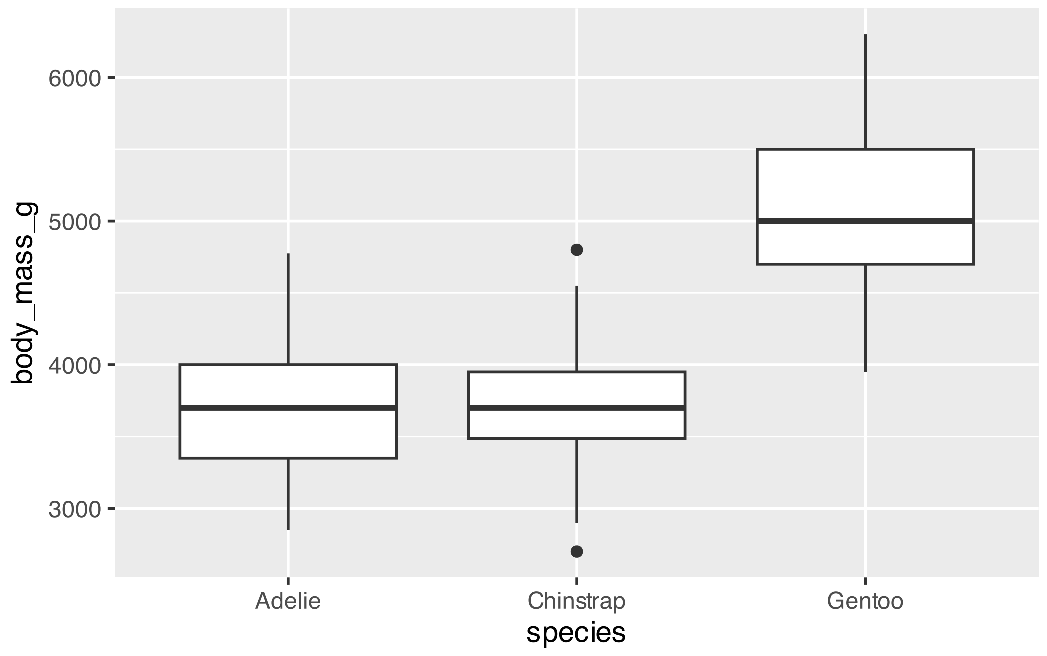

The actual values in a distribution
How a histogram would display the values (rotated)
How a boxplot would display the values
Outliers
Whisker to farthest non-outlier point
75th percentile
50th percentile
25th percentile
1.5 x IQR
Inter-Quartile Range (IQR)

Figure 1: How Box Plot is Created?

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

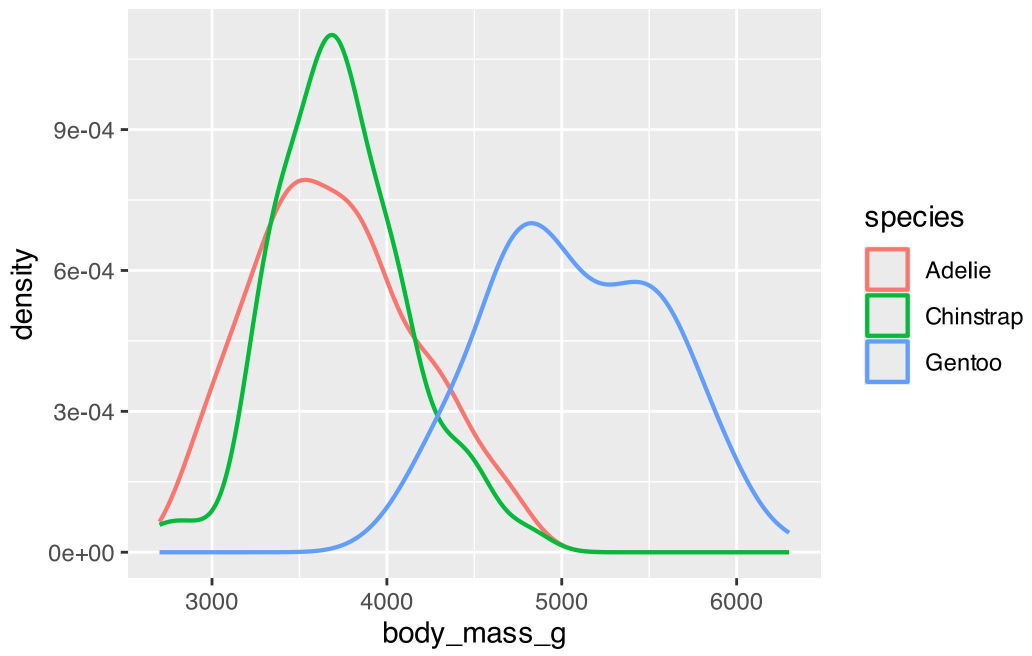
```
Warning: Removed 2 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```



Alternatively, we can make density plots with geom_density()

```r
ggplot(penguins, aes(x = body_mass_g, color = species)) +
  geom_density(linewidth = 0.75)
```

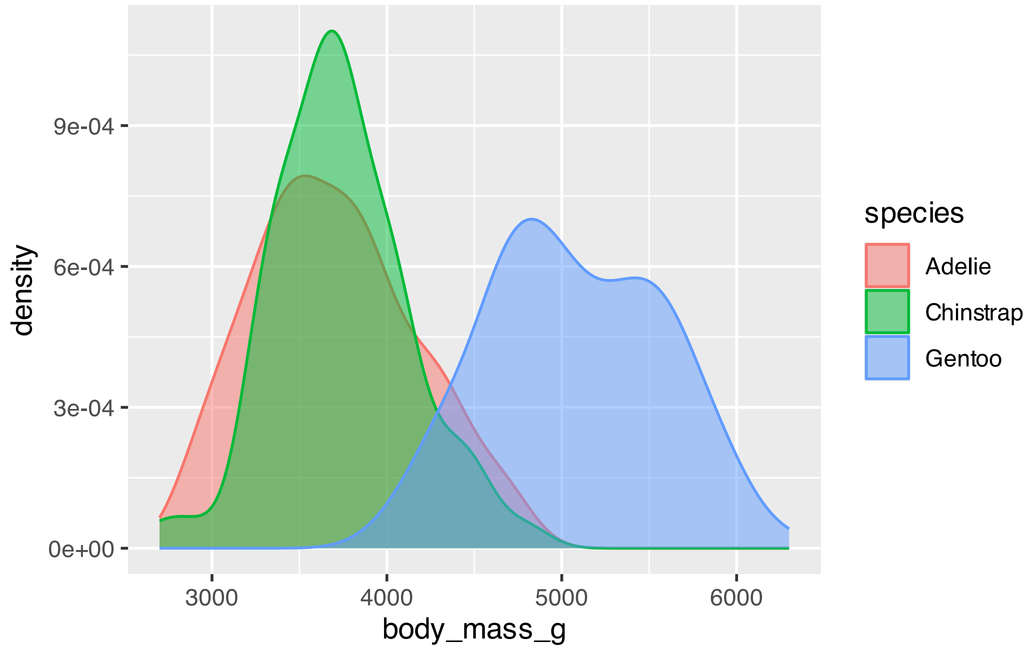
```
Warning: Removed 2 rows containing non-finite outside the scale range
(`stat_density()`).
```



Additionally, we can map species to both color and fill aesthetics and use the alpha aesthetic to add transparency to the filled density curves. This aesthetic takes values between 0 (completely transparent) and 1 (completely opaque). In the following plot it's set to 0.5.

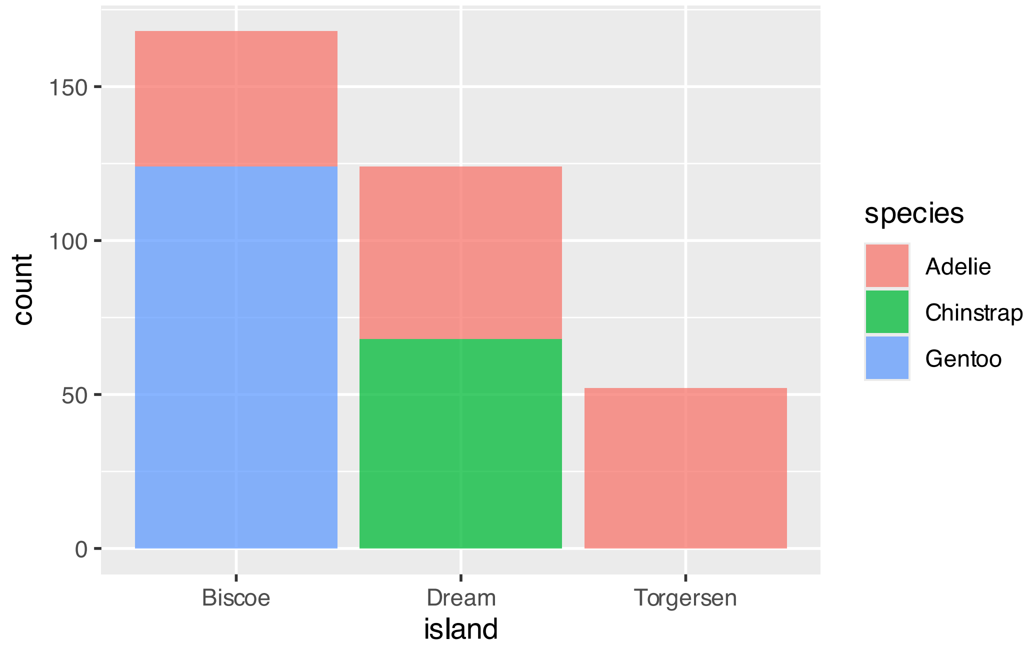
```
ggplot(penguins, aes(x = body_mass_g, color = species, fill = species)) +
  geom_density(alpha = 0.5)
```

```
Warning: Removed 2 rows containing non-finite outside the scale range
(`stat_density()`).
```



3.5 Bar Plots: Counts vs Values

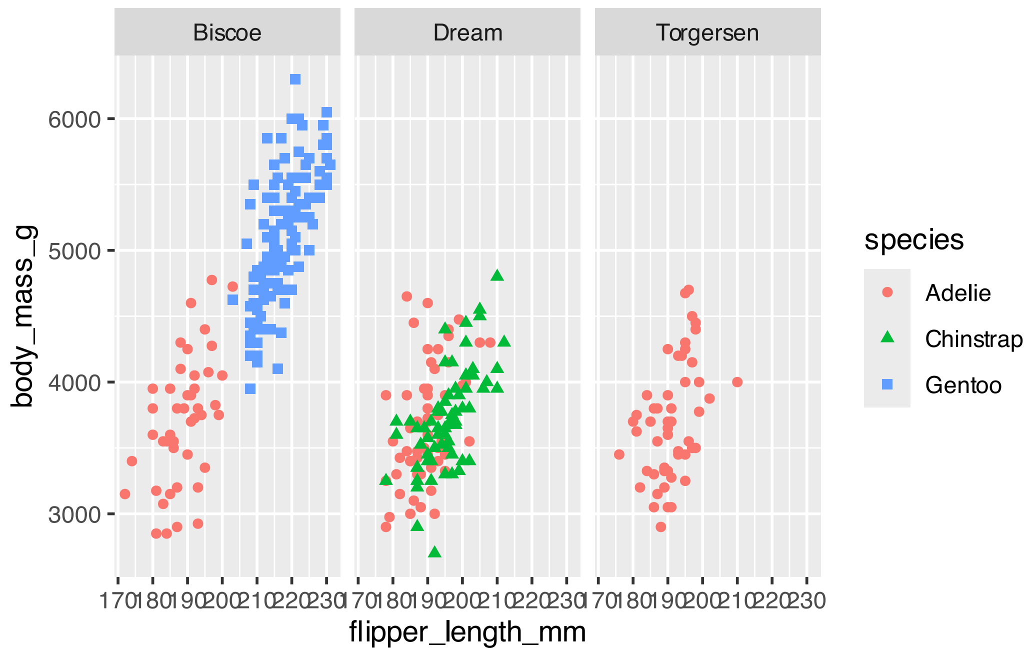
```r
ggplot(penguins, aes(x = island,fill=species)) +
  geom_bar(alpha=0.75)
```



3.6 Faceting (Small Multiples)

```r
ggplot(penguins, aes(x = flipper_length_mm, y = body_mass_g)) +
  geom_point(aes(color = species, shape = species)) +
  facet_wrap(~island)
```

```
Warning: Removed 2 rows containing missing values or values outside the scale
range
(`geom_point()`).
```



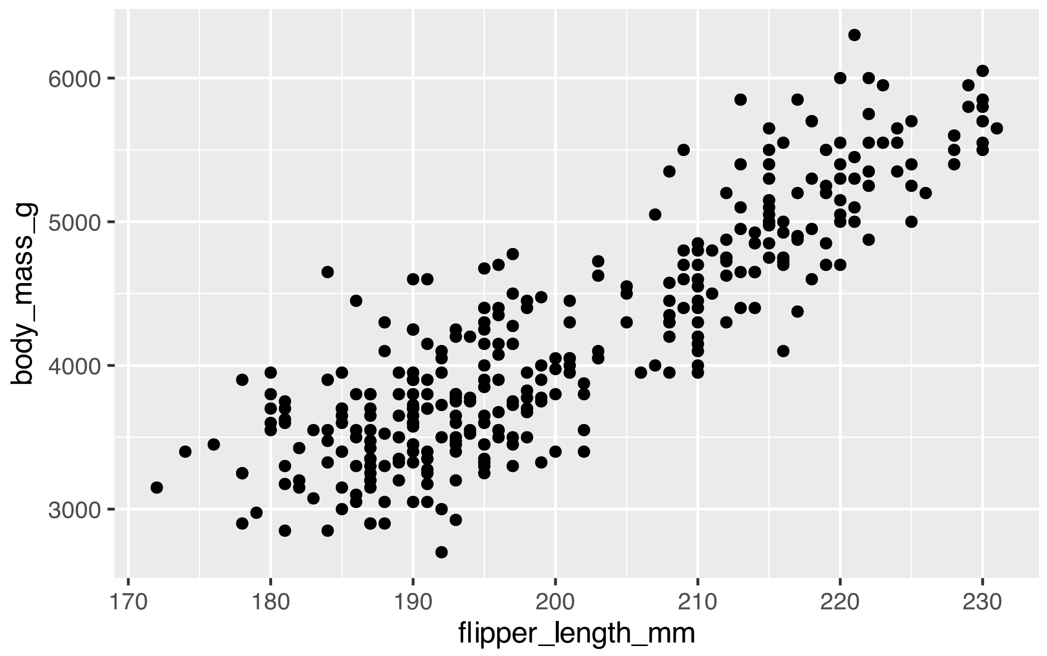
Faceting allows comparison across subgroups.

3.7 Saving your plots

Once you've made a plot, you might want to get it out of R by saving it as an image that you can use elsewhere. That's the job of ggsave(), which will save the plot most recently created to disk:

```r
ggplot(penguins, aes(x = flipper_length_mm, y = body_mass_g)) +
  geom_point()
```

```
Warning: Removed 2 rows containing missing values or values outside the scale
range
(`geom_point()`).
```



```r
ggsave(filename = "penguin-plot.png")
```

```
Saving 5.5 x 3.5 in image
```

```
Warning: Removed 2 rows containing missing values or values outside the scale
range
(`geom_point()`).
```

This will save your plot to your working directory.

If you don't specify the width and height they will be taken from the dimensions of the current plotting device. For reproducible code, you'll want to specify them.

3.8 Exercise: Airline Exposure vs Safety Incidents (1985–1999)

The dataset `airline-safety.csv` contains airline-wise safety statistics across two different time periods.

1. Load the dataset from the CSV file `airline-safety.csv`.
2. Create a scatter plot showing the relationship between `avail_seat_km_per_week` and `incidents_85_99`.
3. Use a logarithmic scale for the x-axis.
4. Add appropriate labels and a title.
5. Interpret whether airlines with higher exposure tend to have more incidents.

6. Do airlines with higher available seat kilometers show a higher number of incidents? What effect does the logarithmic scale have on interpretability?

3.9 Exercise: Distribution of Murder Counts (2014 vs 2015)

The file `murder.csv` contains full-year murder counts for U.S. cities with populations above 250,000.

1. Load the dataset `murder_2015_final.csv`.
2. Create a scatter plot comparing 2015 and 2016 murder counts.
3. Add a reference line where values are equal (`y = x`).
4. Compare the distributions of murder counts for 2014 and 2015.
5. Use box plots to visualize the distributions.
6. Comment on changes in median and variability.

Did the median murder count increase from 2014 to 2015? Are there more extreme outliers in one year?

3.10 Exercise: Analysis of Causes of Police Deaths

Load the data set `police_deaths.csv`, and do the following:

1. Group the dataset by `cause_short`.
2. Calculate the total number of deaths for each cause.
3. Create a **horizontal bar chart** displaying the frequency of deaths by cause.
4. Arrange the bars in descending order of frequency.
5. Add meaningful labels and an informative title.

Also, answer the following quesitons.

- Which cause of death is the most common?
- How does the frequency of deaths due to `Gunfire` compare with other causes?
- What insights can be drawn about occupational risks faced by police officers?