Introduction to Regression Analysis with R

Sophie Lee Dean Marchiori

Table of contents

Welcome **4**

How to use this book . . . 4
Data used in the course . . . 5
Feedback and issues . . . 6
Contact Me . . . 6
Licence and Attribution . . . 6

1 Introduction **7**

1.1 What is a model? . . . 7
1.2 Which type of regression should I use? . . . 7
1.3 Why regression? . . . 8
1.4 Notes on R coding style . . . 8

2 Preparing for linear regression **9**

2.1 Setting up R . . . 9
2.1.1 Packages . . . 9
2.2 Loading the data . . . 10
2.2.1 Tidy Data . . . 11
2.3 Exploring the data . . . 13
2.3.1 Data visualisation . . . 13
2.3.2 Summarising trends . . . 16
Exercise 1 . . . 17

3 Linear regression **18**

3.1 Simple linear regression . . . 18
3.2 Fitting a simple linear regression in R . . . 20
3.3 Multiple linear regression . . . 24
3.3.1 Dummy variables . . . 25
3.4 Model comparisons . . . 29
3.4.1 Adjusted R-squared value . . . 29
3.4.2 Information criterion . . . 30
3.4.3 Prediction metrics . . . 31
3.5 Model diagnostics . . . 32
3.5.1 Linearity . . . 33
3.5.2 Normally distributed residuals . . . 35

3.5.3 Multicollinearity . . . 37
3.5.4 Homoskedasticity . . . 39
3.5.5 Influential observations . . . 39
Exercise 2 . . . 42

4 Generalised linear models **43**
4.1 Generalised linear models in R . . . 43
4.2 Poisson regression . . . 45
4.2.1 Logarithm and exponential transformations . . . 45
4.2.2 Poisson regression for count data . . . 47
4.3 Poisson regression for rates . . . 50
4.4 GLM model diagnostics . . . 53
4.4.1 Poisson regression assumptions . . . 53
4.4.2 Equidispersion . . . 57
Exercise 3 . . . 58

5 Discussion **60**

Appendices **61**

A Data description **61**
A.1 Palmer Penguins . . . 61
A.2 Cancer Registry . . . 62

B Exercise solutions **64**
B.1 Exercise 1 . . . 64
Solution . . . 64
B.2 Exercise 2 . . . 68
Solution . . . 68
B.3 Exercise 3 . . . 74
Solution . . . 74

C Mathematical Derivation of OLS **78**

D Setup **81**
D.1 Setting up R . . . 81
D.1.1 Step 1: Install R . . . 81
D.1.2 Step 2: Install RStudio . . . 81
D.1.3 Packages . . . 81

Welcome

Welcome to the course materials for the Introduction to Regression Analysis with R short course.

This course provides a comprehensive understanding of regression analysis, including the theory behind these models, their application in R, validation techniques, and the interpretation of results. The course begins with an introduction to linear regression models, before advancing to the more flexible family of generalised linear models.

Topics covered as part of this course include:

- Linear regression: concepts, assumptions, application, and interpretations
- Diagnostics and validation of linear regression models
- Generalised linear models: beyond continuous outcomes
- Poisson regression: how to model counts and rates, and how this differs from linear regression
- Best practices in communicating results of regression analysis

How to use this book

This book provides a combination of written explanations, code examples, and practical exercises to allow you to practice what you have learned.

Code examples will be provided in code blocks, such as this one:

```
1 + 1
```

Code in these blocks can be copied and pasted into your R session to save time when coding. We recommend typing the code yourself to familiarise yourself with the coding process and use the copy option if you are really stuck!

Throughout the book, you will see colour-coded boxes which are used to highlight important points, give warnings, or give tips such as keyboard shortcuts.

Note

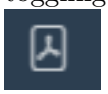
These boxes will be used to highlight important messages, supplementing the main text.

Hint

These boxes will contain useful hints, such as keyboard shortcuts, that can make your coding life a little easier!

Warning

These boxes will contain warnings and highlight areas where you need to be more cautious in your coding or analysis.

To make these notes as accessible as possible, they are available to view in dark mode by toggling the button. They are also available to download as a PDF file using the  button.

All exercise solutions are available in the appendices. Please attempt the exercises yourself first, making full use of R's built in help files, cheatsheets (where available), and example R code in this book. Going straight to the solutions to copy and paste the code without thinking will not help you after the course!

Some exercises contain expandable hints, such as functions required to complete them, that can be viewed when needed. For example:

Exercise hint

The functions you will need for this exercise are `filter` and `count`.

Data used in the course

The examples and exercises in these materials are based on real world data.

Data for this course can be downloaded from the `data` folder of this course's repository.

For more information about this data, including variable descriptions and sources, see the appendix.

Feedback and issues

If you spot a bug or mistake in these notes, please let me know by raising an issue.

Contact Me

If you enjoyed this course and would like me to run a session for your organisation, or if you would like to engage me as a consultant on a project get in touch at deanmarchiori.com

Licence and Attribution

I believe that science should not be behind a paywall, that is why these materials are available for free online, in accordance with the licence.

This work is an adaptation of original work "Regression with R". The material has been modified, to compare against the original works see here.

Regressions with R short course by Sophie Lee is licensed under Creative Commons Attribution-ShareAlike 4.0 International

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International.

1 Introduction

1.1 What is a model?

Modelling is a process that is carried out across many different fields for a wide variety of reasons. Models aim to explain complex processes in as simple terms as possible. The goal of modelling may be to make predictions based on observed values, or to gain insights into the process, while accounting for multiple pieces of data.

In statistics, regression models aim to quantify the relationship between an outcome and one or more explanatory variables using a mathematical equation. They are a powerful and widely used tool that can allow us to make inferences about these underlying relationships whilst accounting for background factors.

1.2 Which type of regression should I use?

This course will focus mostly on **linear models**: models with a single continuous outcome variable that assume the process can be described using a linear equation.

> **Note**
>
> This does not mean that the relationships between variables must be linear. We will see later in the course how models can be extended to account for nonlinear relationships.

We will use linear regression models to address a research question with real-world data. Through the course, you will learn how to fit linear regression models, interpret their outcomes, ways in which models can be extended and improved, how to check models are valid, and finally how to answer the initial research question using regression.

Later in the course, we will see how these linear models can be generalised to outcome variables beyond continuous measures, and how these model interpretations differ. Finally, we will end with a discussion about alternative models that are available beyond those covered in the course.

1.3 Why regression?

As with any other type of statistical analysis, we must always keep in mind the reason for carrying it out. Research questions are an often overlooked but fundamental part of any analysis plan, and should be fully defined before we carry out any analysis, or even collect any data!

> **Note**
>
> Research questions should be clear, concise and *answerable*! For a more detailed introduction to research question generation, including the **PICO** approach and worked examples, check out these notes.

Throughout most of this course, we will be trying to answer questions about penguins in the Palmer Archipelago, Antarctica. Our research question for this course will be:

Is body mass of penguins in the Palmer Archipelago related to their flipper size?

1.4 Notes on R coding style

To ensure that this course is as useful as possible to those attending, all theory will be supplemented with worked example using the R programming language. If you have never used R before, please refer to the setup instructions at the end of these materials.

There are many approaches to coding within R. In this course, we will be using the `tidyverse` approach. This approach requires the `tidyverse` suite of packages to be installed and loaded into the current R session, which we will cover in the next chapter.

2 Preparing for linear regression

2.1 Setting up R

To get started, ensure you have a recent version or R and RStudio installed.

- **Step 1: Install R:** To install R head to https://cran.rstudio.com/ and follow the instructions for your operating system.
- **Step 2: Install RStudio:** Next, install RStudio Desktop IDE at https://posit.co/download/rstudio-desktop/.

2.1.1 Packages

To install the required packages, we run the `install.packages("packagename")` function.

```
# Install the required R packages for our analysis (first time use only)
install.packages("tidyverse")
install.packages("palmerpenguins")
install.packages("Metrics")
install.packages("car")
install.packages("skimr")
install.packages("AER")
install.packages("here")
```

> **Note**
>
> The command `install.packages()` is only required the first time loading a new package or following any substantial updates. The `library()` command must be run every time you start an R session. To save potential issues arising from unloaded packages, put any `library()` commands at the beginning of any script file.

You should be able to now run the following commands:

```r
# Load the installed packages at the start of each session
library(tidyverse)
library(palmerpenguins)
library(Metrics)
library(car)
library(skimr)
library(AER)
library(here)
```

2.2 Loading the data

From our research question, we know that we require data about penguins in the Palmer Archipelago in Antarctica, and that this data must contain information about their body mass and flipper size. This data can be loaded into R using the `{palmerpenguins}` package. More information about the data and its collection can be found on the package website or the original publication.

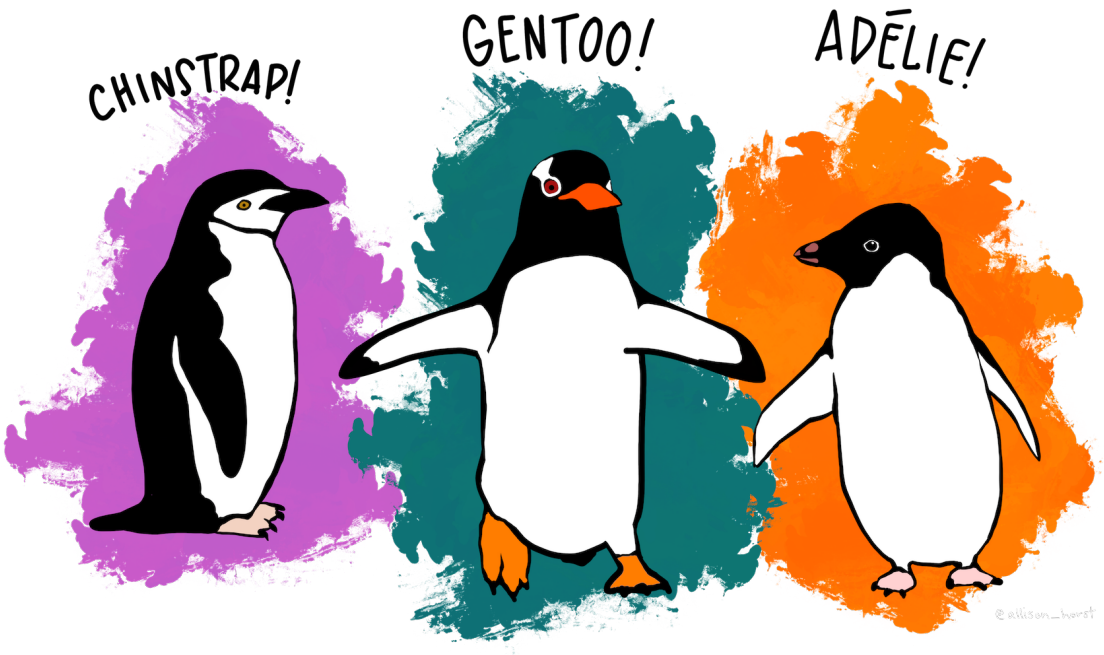


Figure 2.1: Artwork by @allison_horst

While most data will be contained in some external file, in this case the data set is available

within the `palmerpenguins` R package itself. First we load the R package and then we can load the required dataset called `penguins` into our session using the `data()` function.

```
library(palmerpenguins)                                    ①

data(penguins, package = "palmerpenguins")                 ②
```

① Load the package into the current session of R.
② Load the penguin data from this package to our environment.

When loading any data into R, we must run some checks to ensure it has been read in correctly. This includes checking all variables we expect are present, variable names are in a tidy format, and that variables have been recognised as the correct type.

2.2.1 Tidy Data

As defined in R for Data Science:

1. Each variable must have its own column.
2. Each observation must have its own row.
3. Each value must have its own cell.

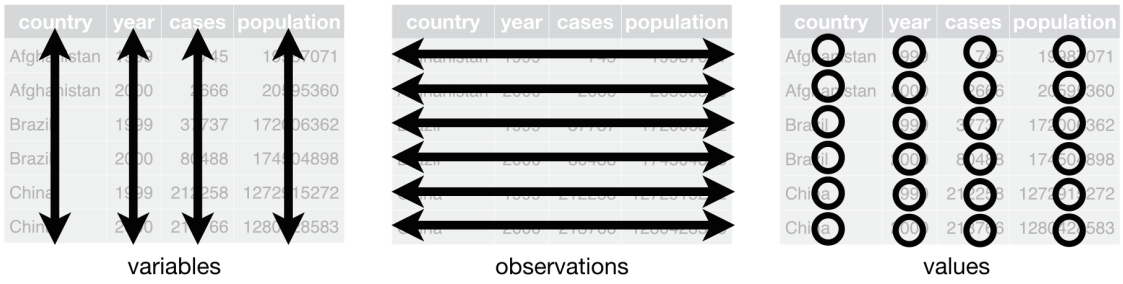


Figure 2.2: Source: https://r4ds.had.co.nz/tidy-data.html

> **Tip**
>
> Variable names should contain only lower case letters, numbers and underscores `_`. They should be clear and descriptive. If you are reading data from a particularly messy source, the `janitor` R package contains the `clean_names` function that converts existing variable names into a 'tidy' alternative.

```
View(penguins)                                                    ①
```

① Preview the dataset in RStudio. Note: Captial V

```
names(penguins)                                                   ②

str(penguins)                                                     ③
```

② Return variable names.
③ Display the structure of the data, including the object type, variable types, and a preview of each variable.

```
[1] "species"           "island"            "bill_length_mm"
[4] "bill_depth_mm"     "flipper_length_mm" "body_mass_g"
[7] "sex"               "year"
tibble [344 x 8] (S3: tbl_df/tbl/data.frame)
 $ species          : Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ island           : Factor w/ 3 levels "Biscoe","Dream",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ bill_length_mm   : num [1:344] 39.1 39.5 40.3 NA 36.7 39.3 38.9 39.2 34.1 42 ...
 $ bill_depth_mm    : num [1:344] 18.7 17.4 18 NA 19.3 20.6 17.8 19.6 18.1 20.2 ...
 $ flipper_length_mm: int [1:344] 181 186 195 NA 193 190 181 195 193 190 ...
 $ body_mass_g      : int [1:344] 3750 3800 3250 NA 3450 3650 3625 4675 3475 4250 ...
 $ sex              : Factor w/ 2 levels "female","male": 2 1 1 NA 1 2 1 2 NA NA ...
 $ year             : int [1:344] 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

The `penguins` dataset contains observations made on 344 penguins. There are 8 variables in the data, including body mass and flipper length, which would need to be included in our final model to answer our research question.

The data consists of a mixture of numeric, binary (`sex`) and nominal (`species`, `island`) variables which appear to be correctly specified within R.

> **Note**
>
> If the data contains ordered categorical variables, ensure they are recognised as `factor` with the correct order assigned. If this is not the case by default, correct this before proceeding, using the `mutate` function to add the converted variable and the `factor` function with `levels` defined in the correct order.

2.3 Exploring the data

When we are sure that the data have been read in correctly and tidied into a useable format, we can begin to explore the data. Data exploration can include

- Data visualisations, used to identify potential outliers, check variable distributions, etc.
- Summarising variables in the sample, to quantify aspects of the variables such as the center and spread (for numeric variables) or the distribution of observations between groups (for categorical variables)
- Quantifying bivariate relationships and differences between groups, using values such as abolute or relative differences, and correlation coefficients

Although data exploration will not allow us to answer the research question, it is a necessary step in the analysis process to build the best possible model. It allows us to identify potential issues that may arise before we encounter them.

2.3.1 Data visualisation

Data visualisation can be an effective method of exploring the data and generating hypotheses. In our example, we are interested in understanding the relationship between penguin's body mass and flipper size. Therefore, it makes sense to begin by visualising these variables. As both variables are continuous, we can use a histogram to visualise them:

> ⚠ Warning
>
> From this point on, we will be using the `{ggplot2}` package which is part of `{tidyverse}` to generate visualisations. Make sure you have loaded the `{tidyverse}` package to your current session of R using code from the previous section.

```
ggplot(data = penguins) +
    geom_histogram(aes(x = body_mass_g)) +
    labs(x = "body mass (g)") +                                    ①
    theme_light(base_size = 12)                                    ②

ggplot(data = penguins) +
    geom_histogram(aes(x = flipper_length_mm)) +
    labs(x = "flipper length (mm)") +
    theme_light(base_size = 12)
```

① Add a tidier label to the x-axis
② Change the default theme and ensure text is at least 12pt in size.

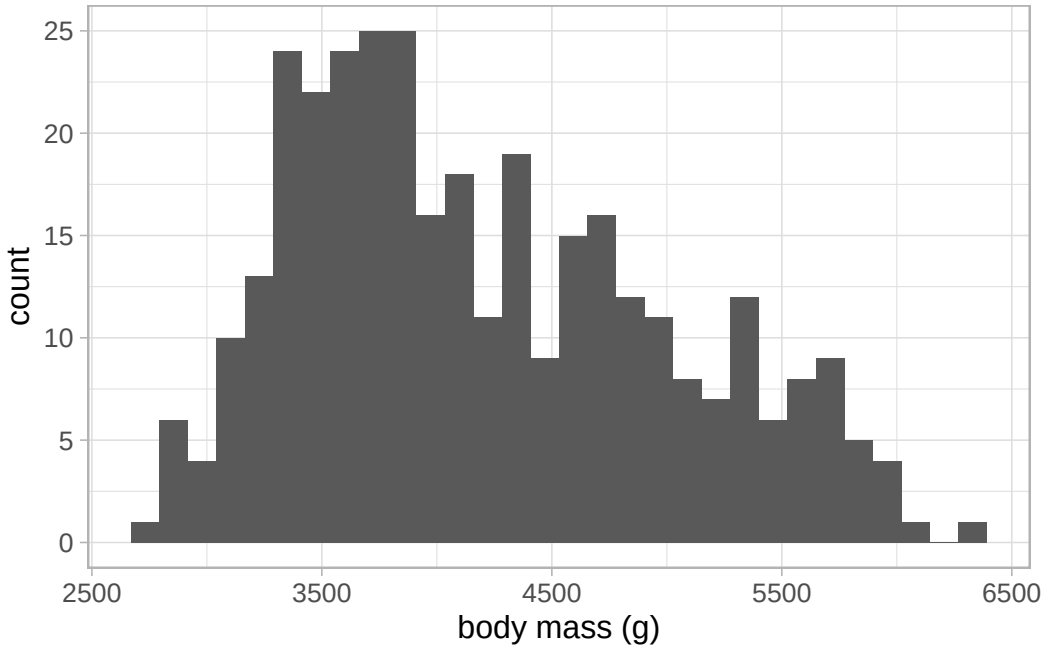


Figure 2.3: Body mass

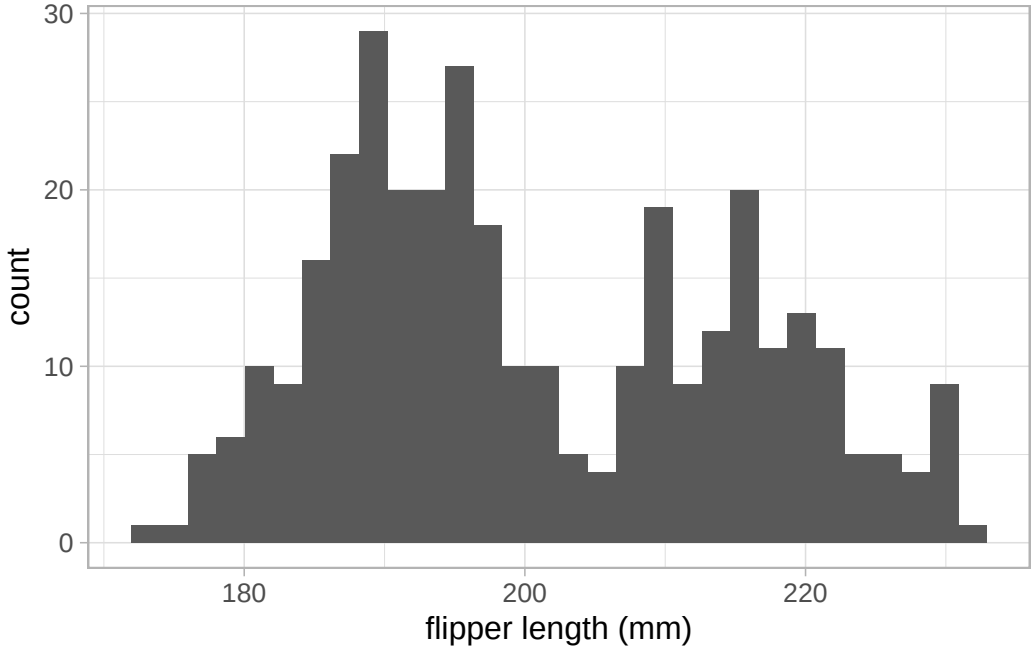


Figure 2.4: Flipper length

These histograms show that neither variable have any outliers of concern. The flipper length variable follows a bi-modal distribution, suggesting that there may be groupings in the data that may be important to explain differences in the sample. The outcome variable, body mass, follows a slightly positively skewed distribution.

> **Note**
>
> There is no requirement that our outcome must follow a normal distribution. A normal distribution is a naturally occurring distribution in many settings, this is why we often use this as a comparison.

We may also want to visualise the relationship between body mass and flipper length in our sample to generate a hypothesis regarding the answer to our research question. A scatterplot is an appropriate visualisation to investigate the relationship between two numeric variables:

```
ggplot(data = penguins) +
    geom_point(aes(y = body_mass_g, x = flipper_length_mm)) +                1
    labs(y = "body mass (g)", x = "flipper length (mm)") +
    theme_light(base_size = 12)
```

① The outcome variable should be displayed on the y-axis, the explanatory variable on the x-axis.

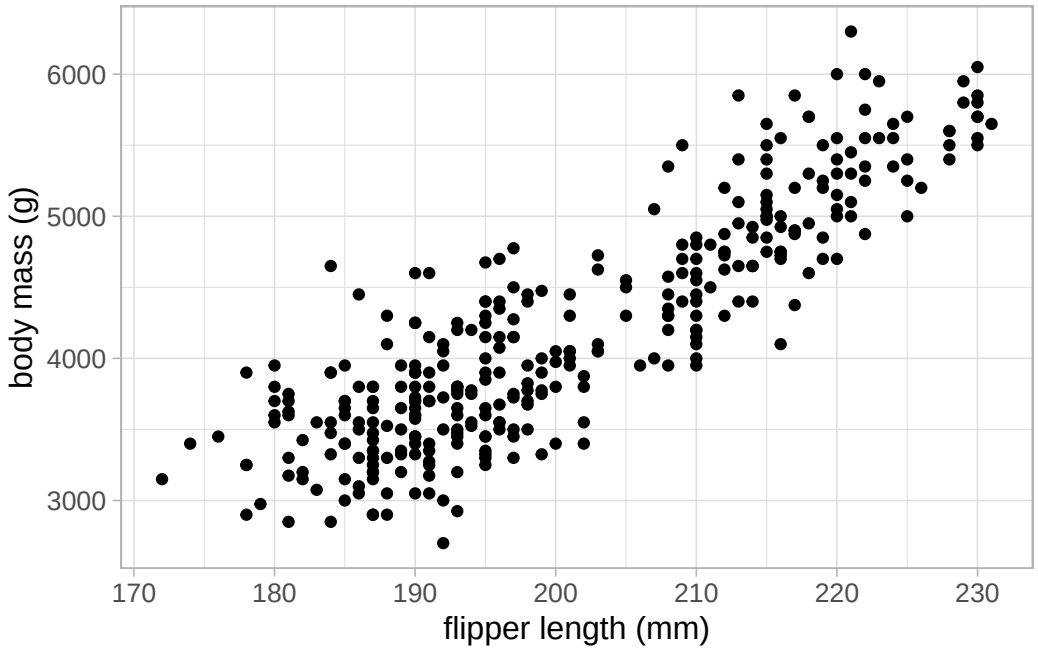


Figure 2.5: Scatterplot showing the relationship between penguins' body mass and flipper length

The scatterplot shows a strong, positive, linear relationship between body mass and flipper length: as flipper length increases, body mass tended to also increase.

2.3.2 Summarising trends

Summary statistics are useful for quantifying different aspects of a sample. As they relate only to the sample, they cannot make inferences about a target population, nor can they answer our research question. They can be used in data exploration though to quantify trends between variables and differences between categories to generate hypotheses about how we may answer our research question.

We saw in Figure 2.5 that there was a strong, linear relationship between penguins' body mass and flipper length. This relationship can be quantified using **Pearson's correlation coefficient**, a measure of linear association between numeric variables.

> **Note**
>
> Correlation coefficients take values between -1 and 1, with 0 representing no association and positive/negative results representing positive/negative associations. The closer the

result is to $\pm$ 1, the stronger an association is.

```
cor(penguins$body_mass_g, penguins$flipper_length_mm,
    use = "complete.obs"
)                                                          (1)
```

(1) As there are missing values in the `penguins` dataset, we must specify the function use only complete observations to avoid an `NA` result.

```
[1] 0.8712018
```

As expected, there is a very strong, positive association between body mass and flipper length. Correlation coefficients can be presented with p-values to make inferences on a target population. However, they provide very little information about the nature of the relationship between variables, for example the magnitude of the relationship.

That is where linear regression comes in!

Exercise 1

Using appropriate visualisations, investigate whether there are other variables that may explain differences in body mass. Consider whether any of these variables may be confounding the relationship between body mass and flipper length, and whether they should be included in the model.

Exercise hint

Consider changing the colour of points in Figure 2.5 to investigate whether the relationship between body mass and flipper length differs between species or sex.
Replace flipper length with other continuous variables to consider whether they may also contribute to differences in body mass.
Use `facet_wrap` to create plots facetted by categorical variables in the data to compare relationships without overlapping points.
If you are REALLY stuck, an example solution can be found here.

3 Linear regression

Linear regression aims to explain the relationship between a single continuous outcome variable and one (or more) explanatory variable(s).

It does this by fitting a line (or plane) that best models the relationship between the outcome variable and the explanatory variables.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \epsilon \tag{3.1}$$

In this equation, Y is the outcome being modelled, X_i are explanatory variables, and β_i are known as regression **coefficients**. ϵ is the error term and accounts for the difference between our observed values of the outcome variable and our model.

> **Note**
>
> As with many other areas of statistics, you may hear some of the elements of a regression model referred to by different names. These all have the same meanings and can be used interchangeably.
> The outcome variable may also be referred to as the dependent or response variable.
> Explanatory variables are also known as independent or predictor variables, or covariates.

3.1 Simple linear regression

Simple linear regression refers to a model with a single continuous outcome and a single explanatory variable. The regression model will assume the relationship between the outcome Y and explanatory variable X:

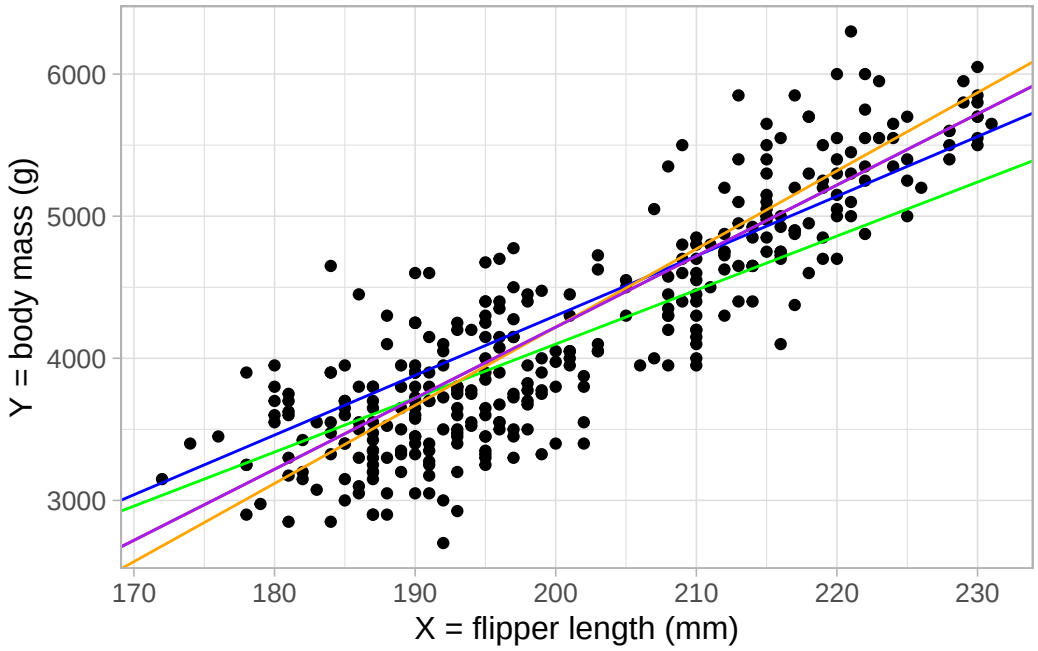
$$Y = \beta_0 + \beta_1 X_1 + \epsilon \tag{3.2}$$

The regression model that we will fit takes the form:

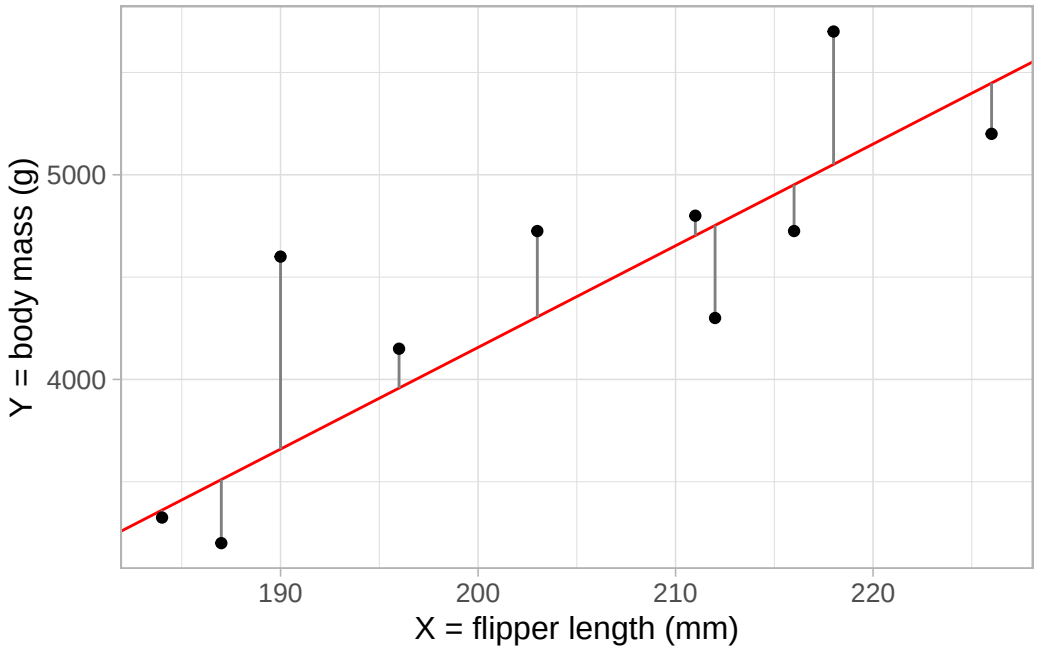
$$\hat{y_i} = \beta_0 + \beta_1 x_i \tag{3.3}$$

This represents the equation of a straight line, where β_0 is the intercept and β_1 is the gradient. The results of this linear regression model will provide the equation of the line of best fit.

We can imagine many straight lines through our data. How do we know which one best models the relationship?



Linear regression works by measuring the distance between each point and its 'predicted' value on the line of best fit. This distance is known as a residual. If we square this distance for each record and sum them all up, we get the sum of squared residuals. The choice of β_0 and β_1 that gives us a line that minimizes the sum of squared residuals is our 'best' fitting line. The full mathematical derivation of this is in the appendix however we don't need to do this by hand. In the next section we will use R to calculate the optimal values of β_0 and β_1.



3.2 Fitting a simple linear regression in R

To fit a simple linear regression to our data in R, we use the `lm()` function and return the model results using `summary()`.

The `lm()` function takes the form of a formula `lm(response ~ explantory)`

```
lm_flipper <- lm(body_mass_g ~ flipper_length_mm, data = penguins)          ①

summary(lm_flipper)                                                         ②
```

① First, we define the regression model equation.
② Summary provides model output, including coefficient estimates, p-values, and other model summaries.

```
Call:
lm(formula = body_mass_g ~ flipper_length_mm, data = penguins)

Residuals:
     Min       1Q   Median       3Q      Max
-1058.80  -259.27   -26.88   247.33  1288.69
```

```
Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       -5780.831    305.815  -18.90   <2e-16 ***
flipper_length_mm    49.686      1.518   32.72   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 394.3 on 340 degrees of freedom
  (2 observations deleted due to missingness)
Multiple R-squared:  0.759, Adjusted R-squared:  0.7583
F-statistic:  1071 on 1 and 340 DF,  p-value: < 2.2e-16
```

There is a lot of information given here, a quick summary of the important points (for now) are:

- `(Intercept)`: this returns the estimate of β_0 from Equation 3.3, the expected value of the outcome where all covariates are 0. Often, the intercept value will not have a meaningful interpretation. For example, here it tells us the expected body mass of penguins with flipper length 0mm is [-5780.83g]
- `flipper_length_mm`: this returns the estimate of β_1 from Equation 3.3, the gradient. This can interpreted as the expected change in the outcome for every unit increase of the associated covariate. In this example, penguins' body mass is expected to be 49.69g higher for every millimeter longer their flippers were.
- `Pr(>|t|)`: the p-value associated with each coefficient estimate, testing the null hypothesis of no association ($\beta_i = 0$). In this model, the p-value associated with β_1 is so small that it cannot be written in its entirety. Therefore we can state that there was a **statistically significant** association between flipper length and body mass.

> **Note**
>
> `<2e-16` is scientific notation for < 0.0000000000000002.

- `Multiple R-squared`: the R^2 value represents the proportion of variance in the outcome variable explained by the model. In this case, the proportion is 0.759 or, if we convert it into a percentage, ($0.759 \times 100 =$) 75.9% of the variation in body mass has been explained by flipper length. The `p-value` under this estimate relates to the R-squared value and tests the null hypothesis that R-squared = 0 (i.e. the model does not explain any of the outcome). In this case, the p-value is too small to be printed (given as `<2e-16`), indicating the model explains a significant amount of the variation in body mass.

Note

Where there is a single continuous explanatory variable, the R^2 value is the Pearson correlation squared. If we take the square root of this value, it will give us the same as the correlation value estimated earlier:

```
r_sq <- summary(lm_flipper)$r.squared

cor(penguins$body_mass_g, penguins$flipper_length_mm,
    use = "complete.obs")
```

```
[1] 0.8712018
```

```
sqrt(r_sq)
```

```
[1] 0.8712018
```

Using the model output gives us the equation of the line of best fit:

$$bodymass = -5780.83 + 49.69 \times flipperlength$$

As the linear equation assumes an additive relationship between the outcome and covariate, we can use this to make estimates about differences in the outcome based on the difference in covariate. For example, if there were two penguins and one had flippers that were 1cm (10mm) longer, we would expect their body mass to be $(49.69 \times 10 =)$ 496.9 heavier.

This line can be added to the scatterplot to visualise the results:

```
ggplot(data = penguins) +
  geom_point(aes(x = flipper_length_mm, y = body_mass_g)) +
  geom_abline(intercept = coefficients(lm_flipper)[1],
              slope = coefficients(lm_flipper)[2],
              colour = "red", linewidth = 2) +
  labs(x = "flipper length (mm)", y = "body mass (g)") +
  theme_light(base_size = 12)
```

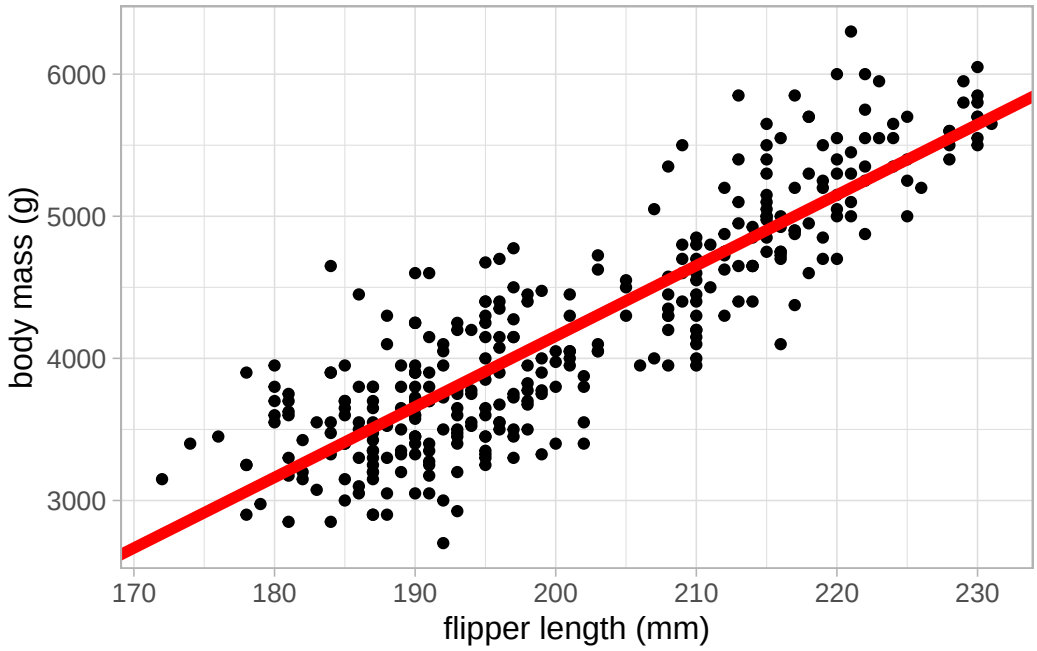


Figure 3.1: Scatterplot of body mass and flipper length with a line of best fit added

To obtain confidence intervals for the coefficient estimates, use the `confint` function:

```
confint(lm_flipper)
```
①

① By default, this returns the 95% confidence interval. This can be adjusted using the `level` argument. For example, `level = 0.9` would return the 90% confidence interval.

```
                       2.5 %      97.5 %
(Intercept)       -6382.35801 -5179.30471
flipper_length_mm    46.69892    52.67221
```

The 95% confidence interval for the flipper length coefficient it [46.7, 52.67]. Therefore we would expect the true effect of flipper length on body mass to be between 46.7 and 52.67 95% of the time under repeated sampling and modelling.

> **Note**
>
> We cannot make any causal statements about increases in flipper length causing increases in body mass as there may be underlying factors confounding these results.

3.3 Multiple linear regression

Multiple, or multivariable, linear regression is a powerful extension which allows models to take account of other observed variables. This is important as **confounding** variables can cause misleading results where they mask or even create spurious associations between variables.

Although the previous model appears to explain a large proportion of the variation in body mass, we want to ensure that this association is not influenced by other variables. For example, we may wish to account for differences in species which is likely to be associated with body mass:

```
ggplot(data = penguins) +
  geom_point(aes(x = flipper_length_mm, y = body_mass_g,
                 colour = species)) +
  scale_colour_brewer(palette = "Dark2") +
  labs(x = "flipper length (mm)", y = "body mass (g)") +
  theme_light(base_size = 12)
```

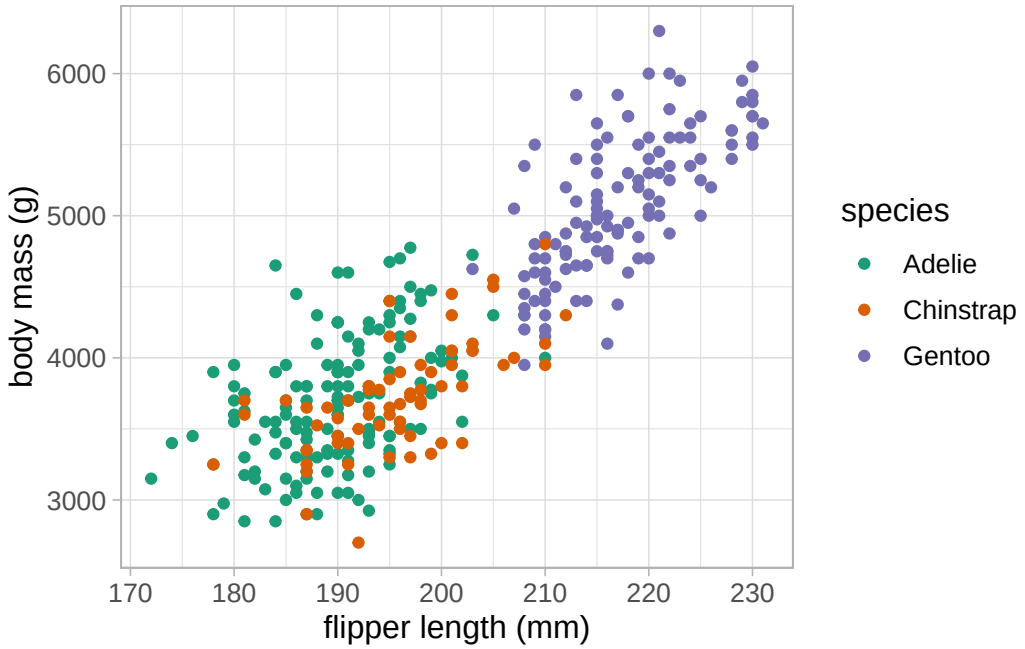


Figure 3.2: Scatterplot investigating the relationship between body mass, flipper length, and species in penguins.

The scatterplot clearly shows that species is highly associated with both body mass and flipper length, making it a potential confounding variable, and therefore a variable we should consider

including in our model. As species is a categorical variable, we must include it into the model as dummy variables.

3.3.1 Dummy variables

Dummy variables take the value 1 or 0, with 1 representing inclusion in a group. Categorical variables require one less dummy variable than the number of categories the variable represents to be included in a regression model. For example, binary variables require one dummy variable, the `species` variable requires two. The category which does not have an associated dummy variable is implicitly included in the model as the reference group.

For example, the species variable would be converted from a single nominal variable to two dummy variables:

species	chinstrap	gentoo
Gentoo	0	1
Gentoo	0	1
Adelie	0	0
Adelie	0	0
Adelie	0	0
Gentoo	0	1
Adelie	0	0
Adelie	0	0
Adelie	0	0
Adelie	0	0

R converts `factor` variables into dummy variables automatically when they are included in the model formula. It is important to check that variables are classified as `factor` before adding them to the model.

```
class(penguins$species)
```

```
[1] "factor"
```

```
levels(penguins$species)
```

```
[1] "Adelie"    "Chinstrap" "Gentoo"
```

The `class` function returns the type of variable and `levels` lists the levels of `factor` variables in the order that they have been specified. R uses the first level of a factor variable as the reference group in a linear model. To change this order, we can use the `fct_relevel` function from tidyverse's `forcats` package. For example, if we want to set the Gentoo species as the reference group, we would use the following:

```
penguins_new <- mutate(penguins,
                       species_gentoo = fct_relevel(species, "Gentoo"))

levels(penguins_new$species_gentoo)
```

```
[1] "Gentoo"    "Adelie"    "Chinstrap"
```

> **Note**
>
> The choice of reference group depends on the data being analysed. If there is a clear reference, for example some control group that we would like to compare all others to, this should be set as the first level. If there is no clear choice of reference, sometimes people choose the largest group.

The factor is then added into the model formula within the `lm` function:

```
lm_flipper_spec <- lm(body_mass_g ~ flipper_length_mm + species,
                      data = penguins)

summary(lm_flipper_spec)
```

```
Call:
lm(formula = body_mass_g ~ flipper_length_mm + species, data = penguins)

Residuals:
    Min      1Q  Median      3Q     Max
-927.70 -254.82  -23.92  241.16 1191.68

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       -4031.477    584.151  -6.901 2.55e-11 ***
flipper_length_mm    40.705      3.071  13.255  < 2e-16 ***
speciesChinstrap   -206.510     57.731  -3.577 0.000398 ***
speciesGentoo       266.810     95.264   2.801 0.005392 **
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 375.5 on 338 degrees of freedom
  (2 observations deleted due to missingness)
Multiple R-squared:  0.7826,    Adjusted R-squared:  0.7807
F-statistic: 405.7 on 3 and 338 DF,  p-value: < 2.2e-16
```

The output contains two additional coefficient estimates (one for each of the dummy variables). The updated linear equation estimated from this model is as follows:

$$bodymass = -4031.48 + 40.71 \times flipperlength + -206.51 \times chinstrap + 266.81 \times gentoo$$

Notice that the coefficient estimate for flipper length has changed. This is because coefficient estimates in multiple regression models give the estimated change in the outcome per unit increase in the associated covariate after adjusting for everything else in the model. Therefore, body mass is expected to increase by 40.71 for every 1mm increase in flipper length *after adjusting for species differences*.

Coefficients associated with dummy variables give the average difference between that group and the reference group, assuming all other variables are equal. For example, Gentoo penguins with the same flipper length as an Adelie penguin were expected to weigh 266.81 more on average.

If we were to visualise this, dummy variables add parallel lines of best fit, one for each group:

```
ggplot(data = penguins) +
  geom_point(aes(x = flipper_length_mm, y = body_mass_g, colour = species)) +
  geom_abline(intercept = coefficients(lm_flipper_spec)[1],                  ①
              slope = coefficients(lm_flipper_spec)[2],
              colour = "#1B9E77", linewidth = 2) +
  geom_abline(intercept = (coefficients(lm_flipper_spec)[1] +                ②
                           coefficients(lm_flipper_spec)[3]),
              slope = coefficients(lm_flipper_spec)[2],
              colour = "#D95F02", linewidth = 2) +
  geom_abline(intercept = (coefficients(lm_flipper_spec)[1] +                ③
                           coefficients(lm_flipper_spec)[4]),
              slope = coefficients(lm_flipper_spec)[2],
              colour = "#7570B3", linewidth = 2) +
  scale_colour_brewer(palette = "Dark2") +
  labs(x = "flipper length (mm)", y = "body mass (g)") +
  theme_light(base_size = 12)
```

① The equation of the line for Adelie penguins will be body mass = -4031.48 + 40.71 × flipper length + -206.51 × 0 + 266.81 × 0 = -4031.48 + 40.71 × flipper length

② The equation of the line for Chinstrap penguins will be body mass = -4031.48 + 40.71 × flipper length + -206.51 × 1 + 266.81 × 0 = (-4031.48 + -206.51) + 40.71 × flipper length

③ The equation of the line for Gentoo penguins will be body mass = -4031.48 + 40.71 × flipper length + -206.51 × 0 + 266.81 × 1 = (-4031.48 + 266.81) + 40.71 × flipper length

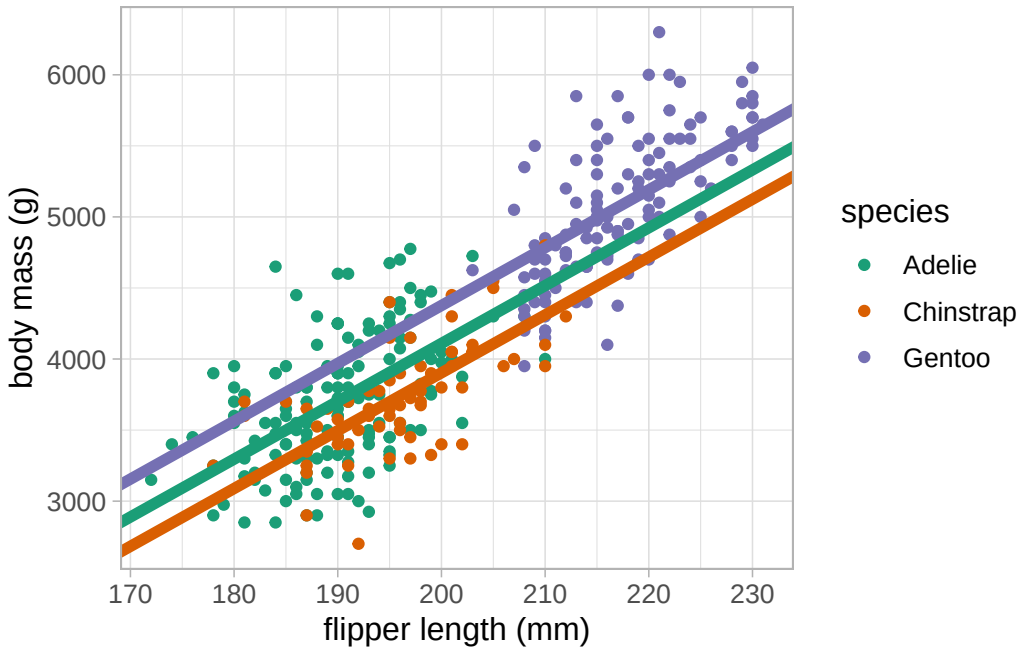


Figure 3.3: Scatterplot of body mass, flipper length and species, with lines of best fit added

All three coefficients had very low p-values and are therefore significantly different from 0, or no association. We can obtain the confidence intervals for the coefficient estimates using `confint`:

```
confint(lm_flipper_spec)
```

```
                       2.5 %      97.5 %
(Intercept)       -5180.50685 -2882.44693
flipper_length_mm    34.66468    46.74612
speciesChinstrap   -320.06672   -92.95352
speciesGentoo        79.42513   454.19408
```

Notice that none of these confidence intervals contain 0, supporting the p-values that we are not compatible with no association between these variables and body mass at the target population level.

> **Warning**
>
> When including categorical variables, we must include all associated dummy variables or none. This is the case even when some coefficients are considered significant and some are not.

3.4 Model comparisons

When choosing the best possible model to address a research question, the aim is usually to find the most parsimonious model for the job. That means the simplest model to explain as much as possible. Model choice should first and foremost be driven by the motivation for the analysis (the research question), our prior knowledge of what other factors are important, and common sense based on preliminary checks. Model choice should not be determined solely by p-values.

There are a number of tools available to help us select the most parsimonious model but these should be considered after motivation, prior knowledge and common sense.

3.4.1 Adjusted R-squared value

The `Multiple R-squared value` provided by the `summary()` function provides a measure of the proportion of the variation of the outcome explained by the model. As we add covariates to the model this value will increase, even by a tiny amount, regardless of whether the model addition is 'worthwhile'. That is why the `Adjusted R-squared` value is also provided.

The `Adjusted R-squared` penalises the R-squared value based on the complexity of the model: the more complex a model is, the higher the penalty. Although the `Multiple R-squared` increases with every model addition, the `Adjusted R-squared` will only increase where the model is considered improved. Therefore, the adjusted value can be used to compare between models to identify the most parsimonious.

We can compare the two models fitted in this section using the Adjusted R-squared to identify the most parsimonious. The higher the Adjusted R-squared, the better the fit:

```
summary(lm_flipper)$adj.r.squared
```

```
[1] 0.7582837
```

```
summary(lm_flipper_spec)$adj.r.squared
```

```
[1] 0.7807187
```

The model including species had a higher Adjusted R-squared and is therefore considered the most parsimonious in this case.

> **Note**
>
> The `Adjusted R-squared` value should be used as a model comparison to identify the most parsimonious model. When the best possible model has been chosen, the results should be presented with the `Multiple R-squared` as a summary.

3.4.2 Information criterion

Another method to assess model goodness of fit is the deviance (also known as -2 log-likelihood). This is a measure of how much a model deviates from a hypothetical full model that predicts each point perfectly. This full model would not be useful to make inferences from as it would only describe the sample it is based on but the deviance can compare similar models to help find the most parsimonious.

The deviance alone is not useful as there is no value where a deviance is 'small enough' to represent a good fit. However, the deviance can be transformed into a score known as an information criterion. These provide a measure of how parsimonious models are by penalising their deviance based on the number of variables included. If the information criterion is lower after adding extra variables, this means the extra complexity explains enough to be worthy of inclusion.

There are a number of information criterions available with different penalties. Two of the most common are the Akaike information criterion (AIC) and the Bayesian information criterion (BIC). These scores will usually give similar results but may differ slightly as they attach different penalties (the BIC usually prefers simpler models to the AIC).

```
AIC(lm_flipper)
```

```
[1] 5062.855
```

```
AIC(lm_flipper_spec)
```

```
[1] 5031.523
```

```
BIC(lm_flipper)
```

```
[1] 5074.359
```

```
BIC(lm_flipper_spec)
```

```
[1] 5050.697
```

The model including species had the lowest value for both information criterion, indicating that the addition of species has improved the model enough to consider it worthwhile.

3.4.3 Prediction metrics

Root mean squared error (RMSE) and mean absolute error (MAE) are model comparison metrics that compare model predictions to the observed values. The smaller the RMSE or MAE, the better the model is at predicting the outcome. There are other similar metrics that can be used on place of these, however the RMSE and MAE are useful as they give the result on the same scale as the outcome.

As the name suggests, the RMSE is estimated by finding the root mean squared error (difference between the observed outcome, y_i and the predicted outcome from the model, $\hat{y}_i$:

$$\sqrt{\frac{1}{n}\sum(y_i - \hat{y}_i)^2}$$

The MAE returns the mean absolute error between the observed and predicted outcome:

$$\frac{1}{n}\sum|y_i - \hat{y}_i|$$

Both these metrics can be estimated using the `{Metrics}` package in R, inputting the observed outcome (here, `body_mass_g`) and the predicted outcome from the model (obtained using the `predict()` function):

```
rmse(lm_flipper$model$body_mass_g, predict(lm_flipper))
```

```
[1] 393.1236
```

```
rmse(lm_flipper_spec$model$body_mass_g, predict(lm_flipper_spec))
```

```
[1] 373.3325
```

```
mae(lm_flipper$model$body_mass_g, predict(lm_flipper))
```

```
[1] 313.0018
```

```
mae(lm_flipper_spec$model$body_mass_g, predict(lm_flipper_spec))
```

```
[1] 296.4359
```

> ⚠ Warning
>
> We have extracted the observed outcome from the model object rather than the original data. This is because there is missing data in the `penguins` data which is removed when we fit the models. Using the original data will lead to variables of different length and produce an error in the functions.

Both metrics agree that the model containing species was better at predicting the body mass of penguins than the model just containing flipper length. Usually we would just choose one of these metrics rather than displaying both. In most cases, they will agree, where they don't it is because the RMSE is more sensitive to outliers, unlike the MAE which treats all values equally.

3.5 Model diagnostics

Before communicating the results of a model, we must ensure that the model we have used is valid and appropriate for the data. Linear regression is a parametric method which means there are assumptions that must be checked to ensure that the model we are using is appropriate. Linear regression assumptions can be remembered using the **LINE** acronym:

- **L**inearity: the relationship between the (mean) outcome and explanatory variable(s) can be described using a linear equation
- **I**ndependence of errors: The residuals (errors) are independent
- **N**ormally distributed errors: The residuals should be approximately normally distributed.
- **E**quivalent variance: The variance of the residuals is constant across all levels of the independent variables. (also known as **homoskedasticity**)

3.5.1 Linearity

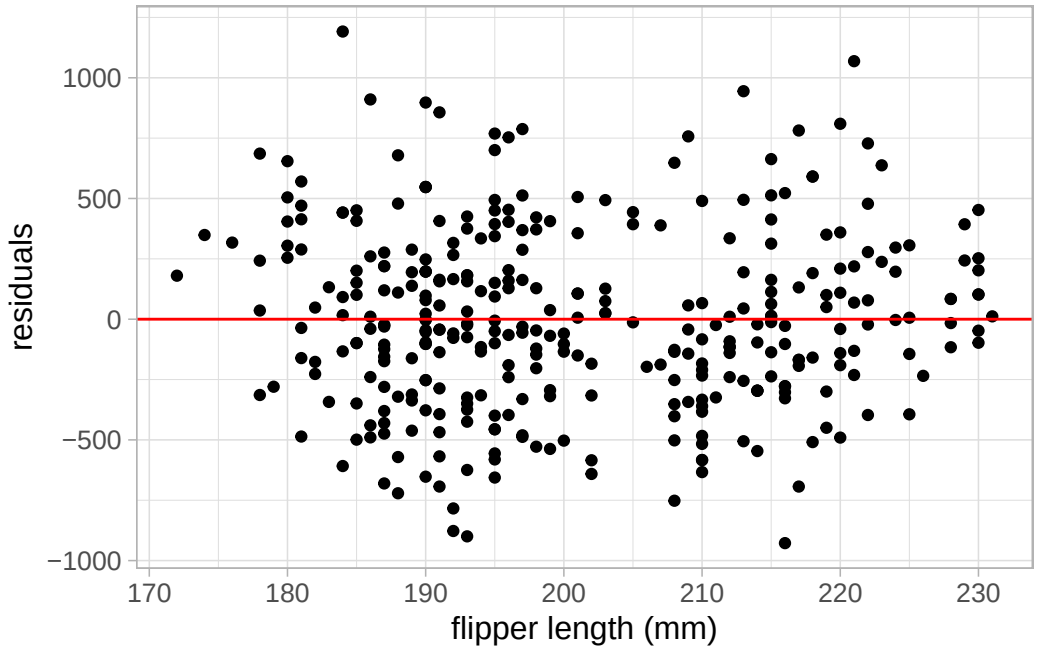
One of the major assumptions underpinning linear models is that the mean outcome can be described as a linear equation of the covariates present in the model. Where there is only a single numeric variable or one numeric and one categorical variable, this assumption can be checked using a scatterplot. However, when models become more complex, we are no longer able to check this assumption graphically.

A better approach to checking the linearity assumption of regression models is to plot a scatterplot of model residuals against each covariate. If the assumption of linearity is valid, residual points should be randomly scattered around 0 without any obvious patterns. To produce these plots, we simply extract the residuals using `resid` and produce a scatterplot using `ggplot` and `geom_point`:

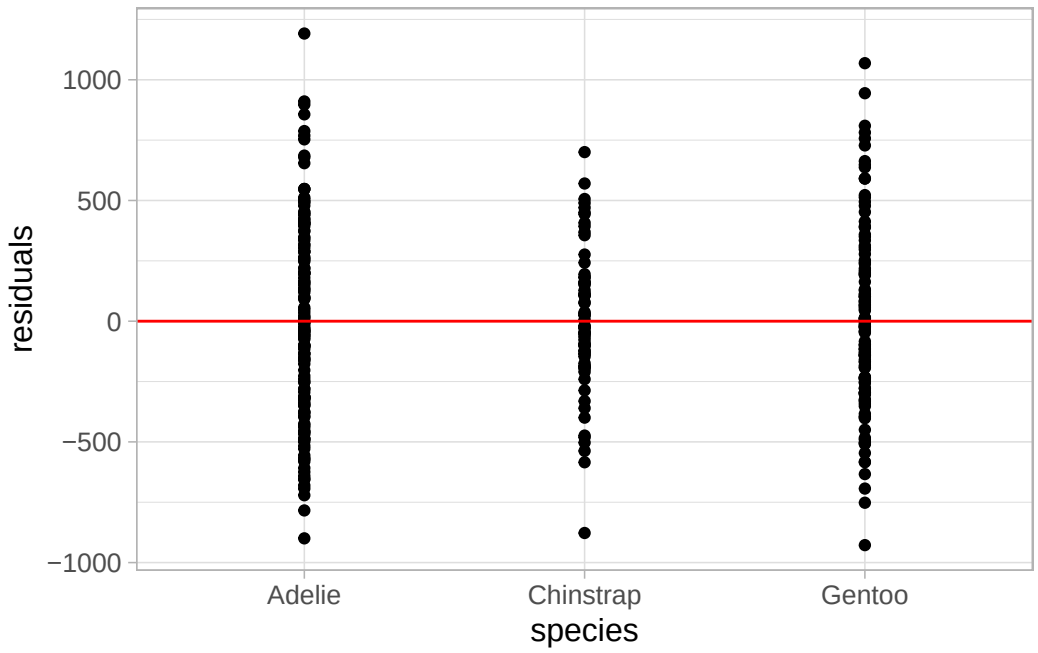
```
penguins_resid <- lm_flipper_spec$model %>%                                    ①
    mutate(
        residuals_flipper_spec = resid(lm_flipper_spec),
        fitted_flipper_spec = predict(lm_flipper_spec)
    )

ggplot(data = penguins_resid) +
    geom_point(aes(x = flipper_length_mm, y = residuals_flipper_spec)) +
    labs(y = "residuals", x = "flipper length (mm)") +
    geom_hline(yintercept = 0, colour = "red") +                               ②
    theme_light(base_size = 12)
```

③ Adding a reference line where residuals = 0 can help check this assumption




```
ggplot(data = penguins_resid) +
    geom_point(aes(x = species, y = residuals_flipper_spec)) +
    labs(y = "residuals", x = "species") +
    geom_hline(yintercept = 0, colour = "red") +                    ③
    theme_light(base_size = 12)
```



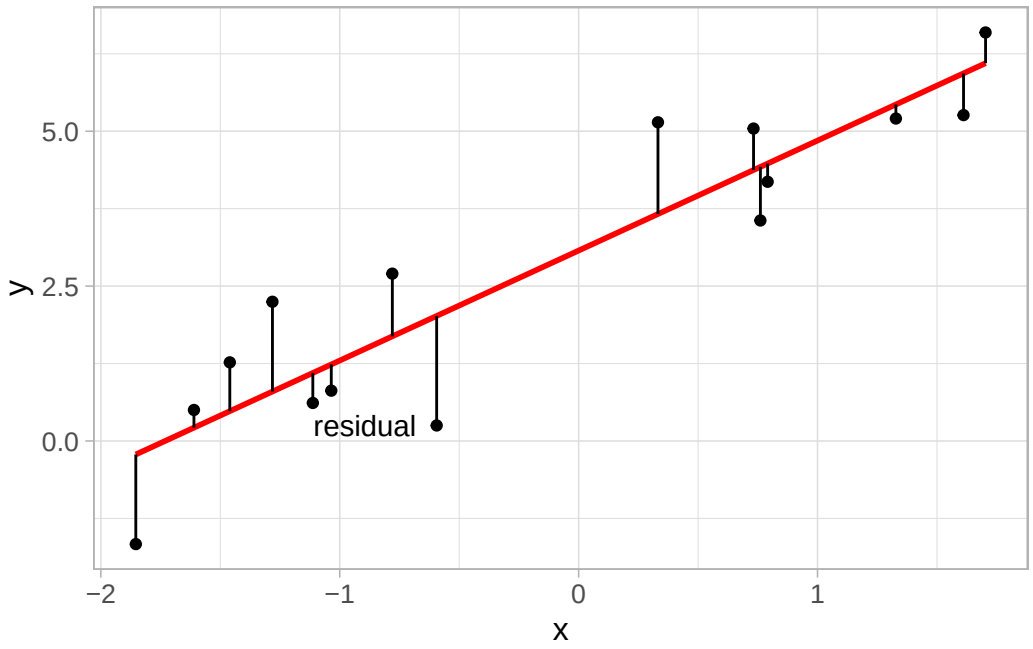
Both plots show residuals randomly scattered around the reference line of 0, with no patterns in the points. This indicates that the assumption of linearity is valid in this case.

> ⚠ Warning
>
> Where the assumption of linearity is not appropriate, steps must be taken to resolve this or another method must be used.
> One way to overcome lack of linearity is to transform the covariate that does not adhere to this assumption. For example, if the relationship between the outcome and a covariate is more curved than linear, a polynomial term (x^2) may be considered. Be cautious when applying transformations as this will change the interpretation of model coefficients (they will no longer be on the same scale as the original data).
> If simple transformations will not overcome a lack of linearity, or where we need coefficients to be interpretable on the data scale, we could consider generalised additive models, an alternative statistical modelling approach that can be used to model nonlinear relationships.

3.5.2 Normally distributed residuals

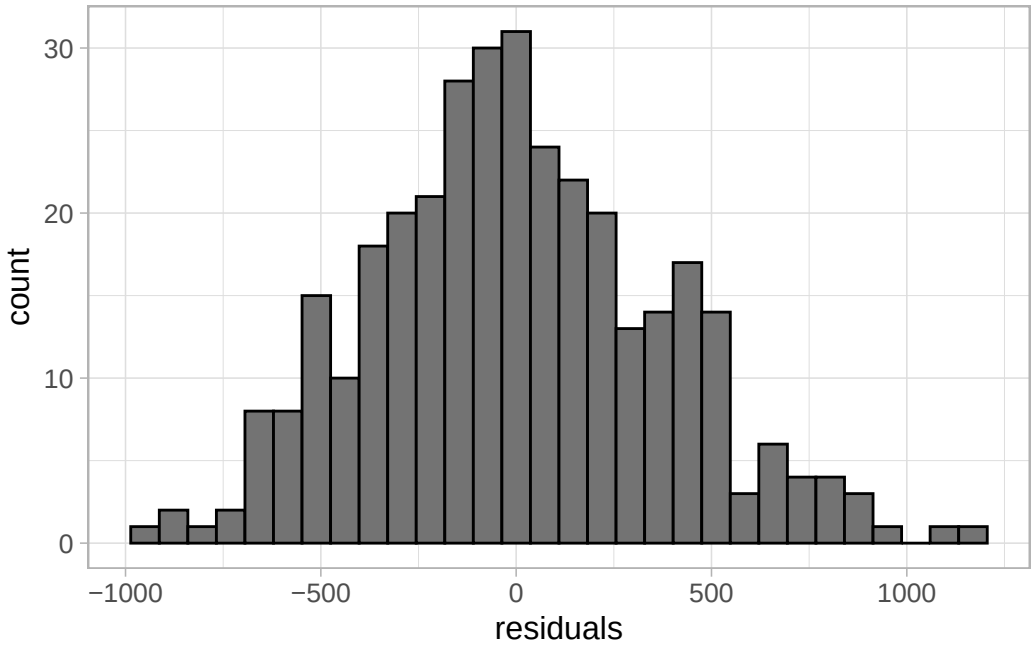
Residuals are calculated by finding the difference between the observed outcome from the data and the predicted outcome using the model. Large residuals are an indication of poor model

fit and can be used to improve a model. Residuals can be obtained from a model object in R using the `resid` function.



A common misconception about linear regression is that the outcome variable must be normally distributed. This is not the case, but the residuals must be. This can be easily checked using a histogram:

```
ggplot(data = penguins_resid) +
  geom_histogram(aes(x = residuals_flipper_spec),
                 colour = "black", fill = "grey45") +
  labs(x = "residuals") +
  theme_light(base_size = 12)
```



The histogram shows residuals follow an approximately normal distribution, centred around 0. Another method is to use the defaul QQ plots in the model diagnostic output.

3.5.3 Multicollinearity

Multicollinearity occurs when one or more of the explanatory variables can be explained by other covariates in the model. In other words, the explanatory variables are not independent of one another. When multicollinearity exists within a model, coefficient estimates become unstable and results may no longer be valid. The level of dependence between continuous covariates can be quantified using the variance inflation factor (VIF).

The VIF is estimated for each covariate, X_i, by fitting a linear model where X_i is set as the outcome, with all other covariates included as explanatory variables. The VIF of this coefficient is estimated using the R-squared value of that model:

$$VIF_i = \frac{1}{1 - R_i^2}$$

The larger the VIF, the more collinearity present in a model. For example, a VIF of 10 occurs when 90% of the covariate is explained by other variables in the model (where $R^2 = 0.9$). There is no agreed level above which multicollinearity causes an issue in the model, although 5 is a common cut-off above which a VIF is considered high.

> **Note**
>
> Although correlation between covariates may give some indication of potential multicollinearity, regression is able to account for some amount of correlation. Although correlated covariates can be included, model interpretations may become more complex as the coefficients are estimated *after adjusting* for other variables in the model, often producing unexpected results.

Where a model contains categorical variables, the generalised VIF (GVIF) can be used. However, as some categorical variables are included into a model with multiple dummy variables, the GVIF must be adjusted to account for differences in the degrees of freedom (df). To make GVIF comparable across variables with different degrees of freedom, we apply the correction:

$$GVIF^{\frac{1}{2df}}$$

This value is also known as the generalised standard error inflation factor (GSEIF). For numeric or binary covariates, this is the square root of the GVIF. Therefore, the rule of thumb cut-offs for problematic levels of multicollinearity when considering GSEIF will be the square root of the VIF cut-offs ($\sqrt{5} = 2.236$).

VIFs, GVIFs and (when the model contains categorical variables) GSEIFs are estimated in R using the `{car}` package:

```
library(car)

vif(lm_flipper_spec)
```

```
                    GVIF Df GVIF^(1/(2*Df))
flipper_length_mm 4.509154  1        2.123477
species           4.509154  2        1.457215
```

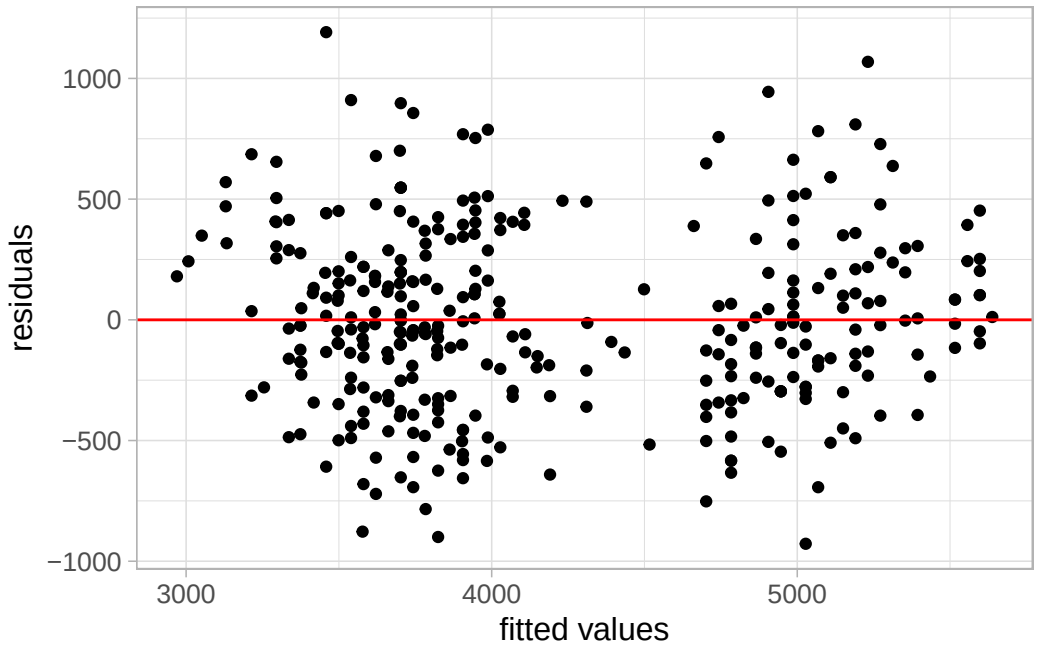
Both GSEIF values are below the cut-off value, indicating there is no issue with multicollinearity in this model.

> **Note**
>
> Where there is evidence of multicollinearity in a model, one or more of the covariates causing the issue must be removed from the model. Otherwise, coefficient estimates will be unstable and inferential results may be invalid.

3.5.4 Homoskedasticity

The final assumption to check when using linear regression is that the residuals have a constant variance across all observations. This can be checked using a scatterplot of the residuals against the predicted values. We would hope to see points scattered randomly around 0. If the variance is not constant, for example if we observe a funnel shape, we must rethink our model:

```
ggplot(data = penguins_resid) +
    geom_point(aes(x = fitted_flipper_spec, y = residuals_flipper_spec)) +
    geom_hline(yintercept = 0, colour = "red") +
    labs(x = "fitted values", y = "residuals") +
    theme_light(base_size = 12)
```



This plot shows evidence of heteroskedasticity: as body mass increases, the variance of the residuals also appears to increase. Therefore, this model could not be used without improvements.

3.5.5 Influential observations

Another consideration when fitting regression models is the existence (and impact) of influential observations. Observations may be influential if they are outliers or behave differently to

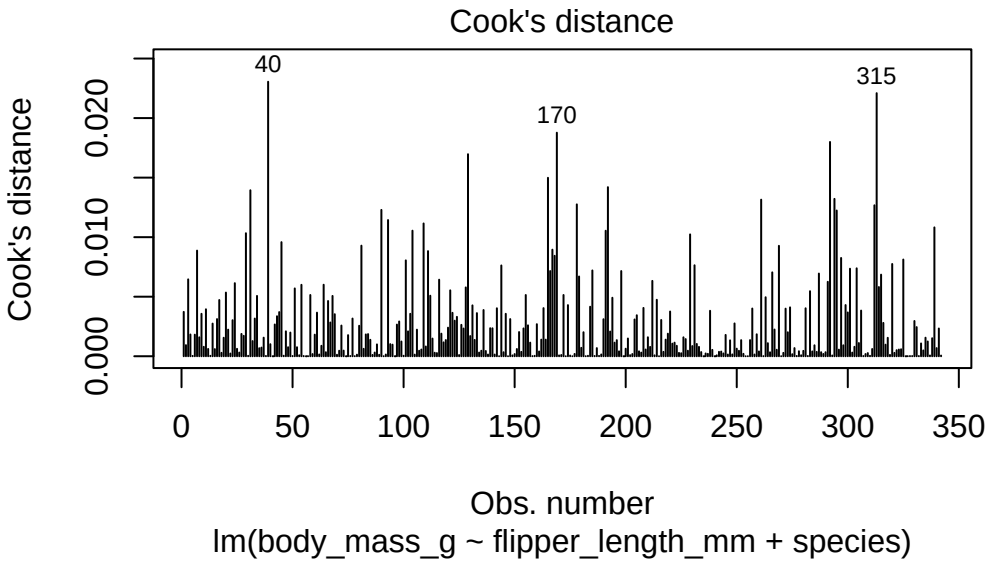
other points, which can lead them pulling the model away from the majority of the sample. This can lead to models that are not representative of the majority of the data.

> **Note**
>
> As with other outliers, influential observations should not necessarily be removed from an analysis if they are part of the target population we would like to address. They may be part of an underrepresented part of the population that was not captured in the random sample.

One of the most common measures of influence is known as Cook's distance. Cook's distances provide a measure of how much the removal of each observation would change the model. The larger the Cook's distance, the more the observation changes the model, making the point more influential.

There are no agreed guidelines giving a cut-off value above which an observation becomes 'influential' (although some have stated around 0.5). The best approach is best to plot Cook's distances and identify extreme values by eye. The Cook's distance can be plotted as follows:

```
plot(lm_flipper_spec, which = 4)
```



By default, R returns the row number of observations it considers 'influential' which can then be used to improve the model.

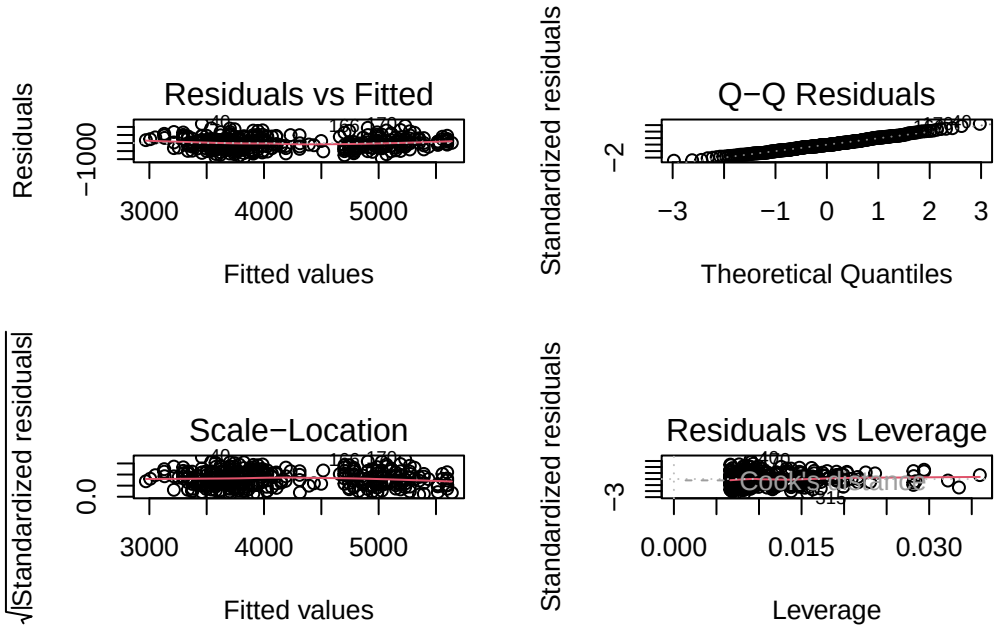
R uses an arbitrary method of identifying 'influential' observations. This means these observations are not necessarily problematic. In this case, all Cook's distances are below 0.025, making their level of influence very low.

R recognised rows 40, 170 and 315 as potentially influential. In some cases, viewing these rows can give us ideas about variables that are not in the current model but explain the differences in these observations that may improve the model:

species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	year
Adelie	Dream	36.5	18.0	182	3150	female	2007
Gentoo	Biscoe	46.2	14.5	209	4800	female	2007
Chinstrap	Dream	49.0	19.5	210	3950	male	2008

> **Hint**
>
> R contains a generic `plot` function that returns model diagnostic plots, including residual plots and Cook's distance when applied to a linear model object.

```
par(mfrow = c(2, 2))
plot(lm_flipper_spec)
```



Exercise 2

Using everything you have learned up to this point, use linear regression to answer the research question posed earlier:

Is body mass of penguins in the Palmer Archipelago related to their flipper size?

> Exercise hint
>
> Explore the data to identify variables that are likely to be related to body mass that could be confounders. This includes visualising and summarising the sameple.
> Add variables into the model, using model comparisons such as the adjusted R-squared, information criterions and RMSE/MAE to understand whether these improve the model.
> When you have chosen what you consider to be the best possible model, check the linear regression assumptions are met and present your answer.
> If you are REALLY stuck, an example solution can be found here.

4 Generalised linear models

Linear regression is a powerful tool but is only appropriate where the outcome of interest is continuous and the residuals are normally distributed. However, we may be interested in answering research questions about a binary outcome, or predicting a rate based on observed variables. This is where generalised linear models (GLMs) are useful.

GLMs extend the linear regression framework to allow other types of outcomes to be modelled using a linear equation. They are appropriate to use for any outcome that is assumed to follow a distribution from the exponential family. The most commonly used GLMs are:

- Linear regression: continuous outcome
- Logistic regression: binary outcome
- Poisson regression: count/rate outcome
- Ordinal logistic regression: ordinal outcome
- Multinomial regression: nominal outcome

GLM generalises the linear model framework by fitting the linear model to some transformation of the outcome. This transformation is known as the link function. Each regression type has an associated link function, $g(Y)$, making the updated model formula:

$$g(Y) = \beta_0 + \beta_1 X_1 + ... \beta_k X_k$$

Linear regression is the simplest form of GLMs as the link function is the identity function $g(Y) = Y$. Where another link function has been used, coefficient estimates must be transformed to put them onto the same scale as the data before they are interpretable.

4.1 Generalised linear models in R

To fit a GLM in R, we use the `glm()` function. The model specification is the same as the `lm()` function but with an optional `family` argument where the outcome is not continuous.

We can fit the linear model from the previous section using the following code:

```r
glm_penguin <- glm(body_mass_g ~ flipper_length_mm + species,
                   data = penguins, family = gaussian)                    ①

summary(glm_penguin)
```

① Gaussian distribution is another term for normal distribution.

```
Call:
glm(formula = body_mass_g ~ flipper_length_mm + species, family = gaussian,
    data = penguins)

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       -4031.477    584.151  -6.901 2.55e-11 ***
flipper_length_mm    40.705      3.071  13.255  < 2e-16 ***
speciesChinstrap   -206.510     57.731  -3.577 0.000398 ***
speciesGentoo       266.810     95.264   2.801 0.005392 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 141026.6)

    Null deviance: 219307697  on 341  degrees of freedom
Residual deviance:  47666988  on 338  degrees of freedom
  (2 observations deleted due to missingness)
AIC: 5031.5

Number of Fisher Scoring iterations: 2
```

> **Note**
>
> For a full list of `family` options, open the `?family` help file.

The `lm()` and `glm()` functions will provide the same results when fitting a linear model, although some of the output provided in the `summary` is slightly different (`lm()` will provide the R-squared values by default, whereas `glm()` provides deviance and the AIC).

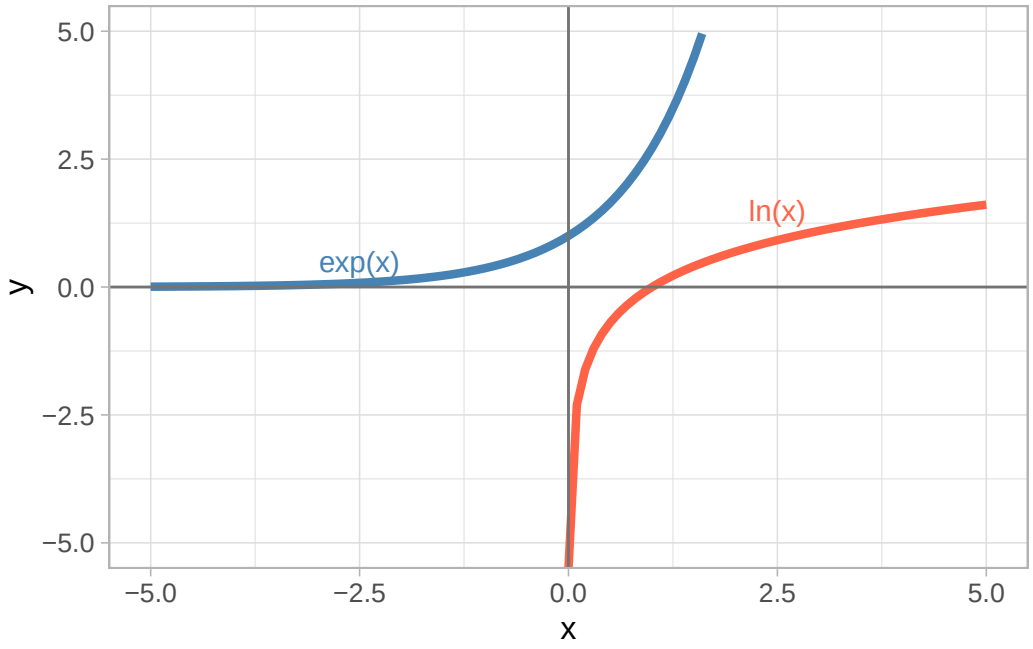
4.2 Poisson regression

Poisson regression is used where the model outcome is either a count or rate. Linear regression would not be appropriate for count data as predictions could be negative or non-integer values. To overcome this, the model is fitted to a transformed version of the outcome. In theory, any mathematical function could be used as a link function as long as there is an opposite function that would return values to their original scale, making model results interpretable.

> **Note**
>
> We do not apply any transformation to the data, this is done by the computer when we are fitting the model. The coefficients will be estimated in relation to the log-transformed outcome rather than the original variable.

4.2.1 Logarithm and exponential transformations

The transformation applied to the outcome when fitting a poisson regression is the natural log function (also known as ln or log_e (log to the base e). The opposite of this transformation is the exponential function (sometimes written as exp or e).



The natural log function is useful when dealing with counts as it takes non-negative variables and 'stretches' them, allowing them to take any value between $\pm\infty$. To return to the original data, we just apply the exponential function to the transformation:

Original values (x)	ln(x)	exp(x)
-0.5	-	-
0.0	-	-
0.5	-0.69	0.5
1.0	0	1
2.0	0.69	2
5.0	1.61	5
10.0	2.3	10
100.0	4.61	100
1000.0	6.91	1000

The exponential and natural log transformations have some interesting properties that we must be aware of before interpreting poisson regression results. This is because model results may need to be transformed before they can be interpreted.

For example, the exponential function takes an additive relationship and converts it into a multiplicative one:

$e^{a+b} = e^a \times e^b$

e.g., $e^5 = e^{2+3} = e \times e \times e \times e \times e = (e \times e) \times (e \times e \times e) = e^2 \times e^3$

Multiplicative relationships become exponential relationships:

$e^{a \times b} = (e^a)^b = (e^b)^a$

e.g., $e^6 = e^{2 \times 3} = e \times e \times e \times e \times e \times e = (e \times e) \times (e \times e) \times (e \times e) = (e \times e)^3 = (e^2)^3$

The natural log also has some important properties that we must be aware of during the poisson regression process:

$ln(a + b) = ln(a) \times ln(b)$

e.g., $ln(8) = ln(6 + 2) = ln(6) \times ln(2)$

$ln(a - b) = ln(a) \div ln(b)$

e.g., $ln(5) = ln(9 - 4) = ln(9) \div ln(4)$

4.2.2 Poisson regression for count data

To demonstrate poisson regression, we will be using data from the cancer registry and census in the USA from 2015. This data can be downloaded from the course repository and more information about the dataset can be found in the course appendix.

The research question we will aim to answer using this data is:

Is cancer mortality associated with poverty levels in the USA?

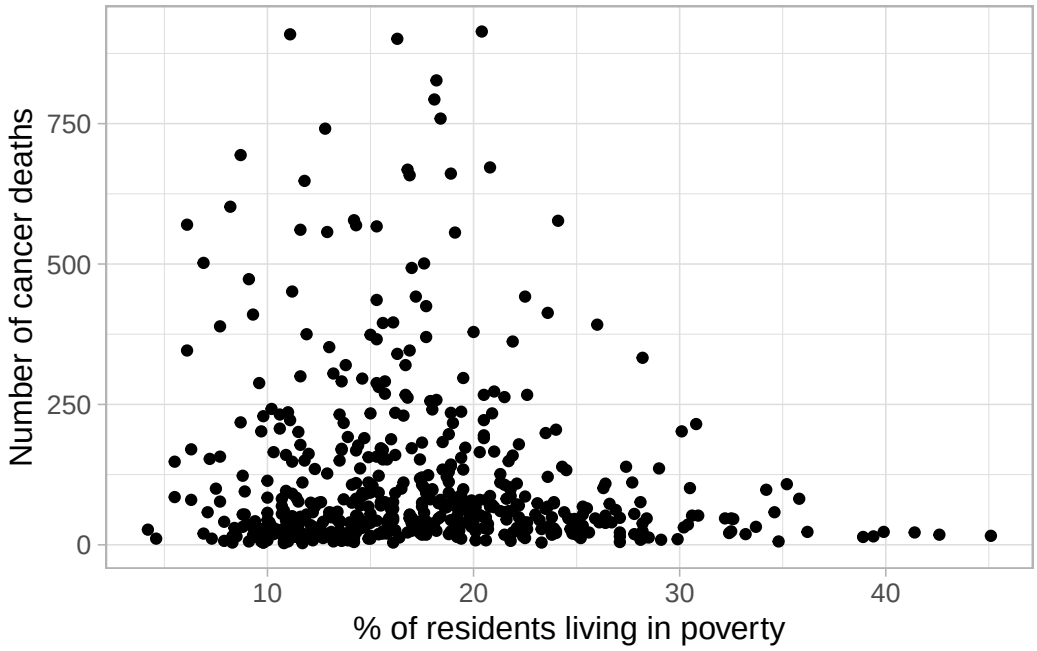
Regardless of what model comparison statistics find, our final model must have number of cancer deaths as the outcome and some measure of poverty included as an explanatory variable.

We begin be exploring the bivariate relationship between these variables by loading the data and producing a scatterplot:

```
cancer_reg <- read_csv(here("data/cancer_reg.csv"))                    ①

ggplot(data = cancer_reg) +
  geom_point(aes(x = poverty, y = number_death)) +
  labs(x = "% of residents living in poverty",
       y = "Number of cancer deaths") +
  theme_light(base_size = 12)
```

① This code requires the `cancer_reg.csv` file to be saved in a folder within your working directory named `data`. If the data are saved in the working directory, remove the `data/` prefix.



The scatterplot does not appear to show any clear relationship between the poverty levels in US counties and the number of cancer deaths. However, this may be due to the smaller number of counties with high levels of poverty in the sample.

We can quantify the level of this relationship using a simple[1] poisson regression model:

```
glm_cancer_pov <- glm(number_death ~ poverty, data = cancer_reg,
                      family = poisson)

summary(glm_cancer_pov)
```

```
Call:
glm(formula = number_death ~ poverty, family = poisson, data = cancer_reg)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  5.1946575  0.0114207  454.85   <2e-16 ***
poverty     -0.0243758  0.0006472  -37.66   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

[1] simple = only one covariate

```
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 74152  on 526  degrees of freedom
Residual deviance: 72668  on 525  degrees of freedom
AIC: 75821

Number of Fisher Scoring iterations: 5
```

The model results show a significant association between the number of cancer deaths and rates of poverty in counties. The coefficient estimates from the model output can be used to construct the linear equation fit to this data:

$$\ln(\text{number of deaths}) = 5.19 - 0.02 \times poverty$$

However, we are not able to interpret these effects on this scale, we must back-transform the outcome using the exponential function. If we apply the exponential function to the left-hand side of this equation, we must do the same to the right side to ensure equality. Therefore, the model equation becomes:

$$\text{number of deaths} = e^{5.19 - 0.02 \times poverty}$$

$$= e^{5.19} \times e^{-0.02 \times poverty}$$

$$= 180.31 \times 0.98^{poverty}$$

The transformed intercept value, 180.31, is the expected number of cancer deaths where no one in the county lived in poverty.

The coefficient associated with poverty level, 0.98, now describes the multiplicative relationship between poverty and cancer deaths. For every 1 percentage point increase in poverty, the number of deaths is expected to decrease (as the transformed coefficient is below 1, no difference). This decrease can be converted into the percentage change to make it easier to communicate:

First, we find the difference between the multiplicative change and no difference (1 in this case): 1 - 0.9759 = 0.0241.

This represents the proportion change in the outcome. To convert a proportion into a percentage, we simply multiply it by 100%:

$0.0241 \times 100\% = 2.41$ %.

For every 1 percentage point increase in poverty, the number of deaths is expected to decrease by 2.41 %. To find the expected difference in counties where the poverty level was 10% higher, we exponentiate the transformed coefficient by itself 10 times and convert it into a percentage change:

$0.9759^{10} = 0.7837$

1 - 0.7837 = 0.2163

$0.2163 \times 100\% = 21.63$ %.

The confidence intervals for coefficient estimates can be obtained using the `confint` function we used before. However, the interval will be presented on the transformed scale and so we need to apply the exponential function before interpreting the results:

```
round(exp(confint(glm_cancer_pov)), 2)
```

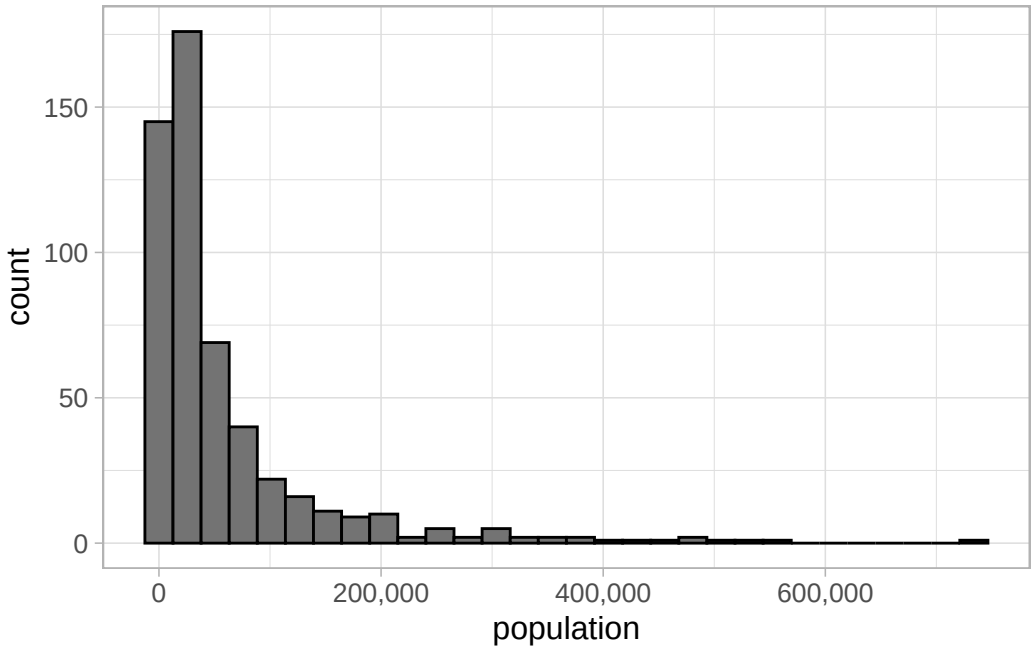
```
             2.5 % 97.5 %
(Intercept) 176.31 184.39
poverty       0.97   0.98
```

Therefore, we can state the true effect of number of deaths in a county with a poverty level 1 percentage point higher will be between 0.97 and 0.98 at a 95% level of confidence.

4.3 Poisson regression for rates

The previous model showed there was a negative relationship between the number of deaths from cancer in the US and poverty. This is the opposite of what we would expect to see and could be an indication that there are confounding factors. As with linear regression, we must consider whether there are background factors that may be distorting the results of our model that must be included to obtain valid results.

The counties in our sample vary greatly in size:



Using the number of deaths as the outcome does not account for the differences in county populations. The higher the population, the more people that are at risk. Failure to adjust for this means we may just be measuring the relationship between poverty and population.

To avoid masking the relationship we are interested in modelling, we can introduce the population into the model. This can be done by modelling the rate of deaths rather than the raw count. We don't aim to model the rate of death directly, we still leave our response as `number_death` however we add an 'offset' term to the right hand side. For this we will use `log(population_2015)`.

```
glm_mort_rate <- glm(number_death ~ poverty + offset(log(population_2015)),
    data = cancer_reg,
    family = "poisson"
)

summary(glm_mort_rate)
```

```
Call:
glm(formula = number_death ~ poverty + offset(log(population_2015)),
    family = "poisson", data = cancer_reg)
```

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -6.3846438  0.0125010 -510.73   <2e-16 ***
poverty      0.0105205  0.0007165   14.68   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 4101.4  on 526  degrees of freedom
Residual deviance: 3888.2  on 525  degrees of freedom
AIC: 7041.7

Number of Fisher Scoring iterations: 4
```

> **Note**
>
> The offset term does not have a coefficient estimated by the model as it is considered a constant with a coefficient of 1.

We can view the new model implicitly modelling the rates of death as follows:

$$\ln(\text{number of deaths}) = -6.38 + 0.01 \times \textit{poverty} + \ln(\text{population 2015})$$

$$\ln(\frac{\text{number of deaths}}{\text{population 2015}}) = -6.38 + 0.01 \times \textit{poverty}$$

Although this model uses the same data as the first, the coefficient of the model now shows a positive association between poverty levels and mortality rate (as poverty increases by a percentage point, the mortality rate is expected to increase $e^{0.0105 \times 1} = 1.0106$. This indicates that the previous model, using count as the outcome, was modelling differences in population rather than cancer deaths.

The 95% confidence interval for poisson models of rates is produced in the same way as those for counts. Remember to transform the interval using the exponential function before interpreting these values.

```
round(exp(confint(glm_mort_rate)), 2)                                    ①
```

① I have rounded the interval to ensure the output is tidier.

```
             2.5 % 97.5 %
(Intercept)  0.00   0.00
poverty      1.01   1.01
```

4.4 GLM model diagnostics

Each form of generalised linear model has a different set of assumptions that must be checked to ensure results are valid. The assumptions will depend on the link function that is applied to the outcome, and the distribution that the outcome is expected to come from (for linear models, this is the normal distribution, for poisson models it is poisson, etc.).

For example, GLMs have a linearity assumption, but that the transformed outcome can be described using a linear equation containing covariates in the model.

4.4.1 Poisson regression assumptions

As with linear models, observations used to fit a poisson model must be independent of one another, and explanatory variables included in the model must not be dependent on one another. However, there are some important differences between assumptions made about the model response and residuals.

One of the main differences between the assumptions underpinning linear and poisson regression models is that the count (or rate) outcome is assumed to follow a poisson distribution. A key assumption of the poisson distribution is that the mean and variance are equal. Therefore, the assumption that residuals have a constant variance is not appropriate.

Rather than considering the raw residuals (the difference between the observed and expected outcomes) for poisson models, Pearson residuals and deviance residuals give more insight into model validity and fit.

4.4.1.1 Pearson residuals

Pearson residuals (r_i^p) standardise raw residuals by dividing the difference between observed (y_i) and predicted ($\widehat{y}_i$) outcomes by the standard deviation. For poisson regression, if the model (and poisson distribution assumption) is valid, the standard deviation will be the square root of the mean, or predicted outcome of the model:

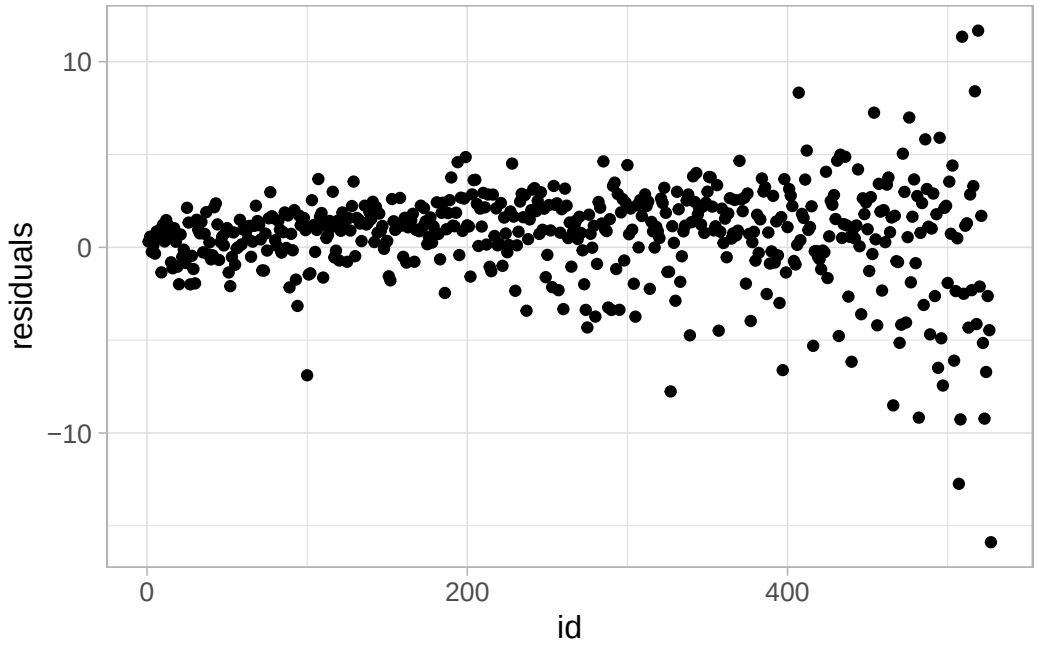
$$r_i^p = \frac{y_i - \widehat{y}_i}{\sqrt{\widehat{y}_i}}$$

If the outcome (and therefore the residuals) followed a poisson distribution, we would expect these standardised Pearson residuals to follow a normal distribution, with a constant variance, and a mean of 0.

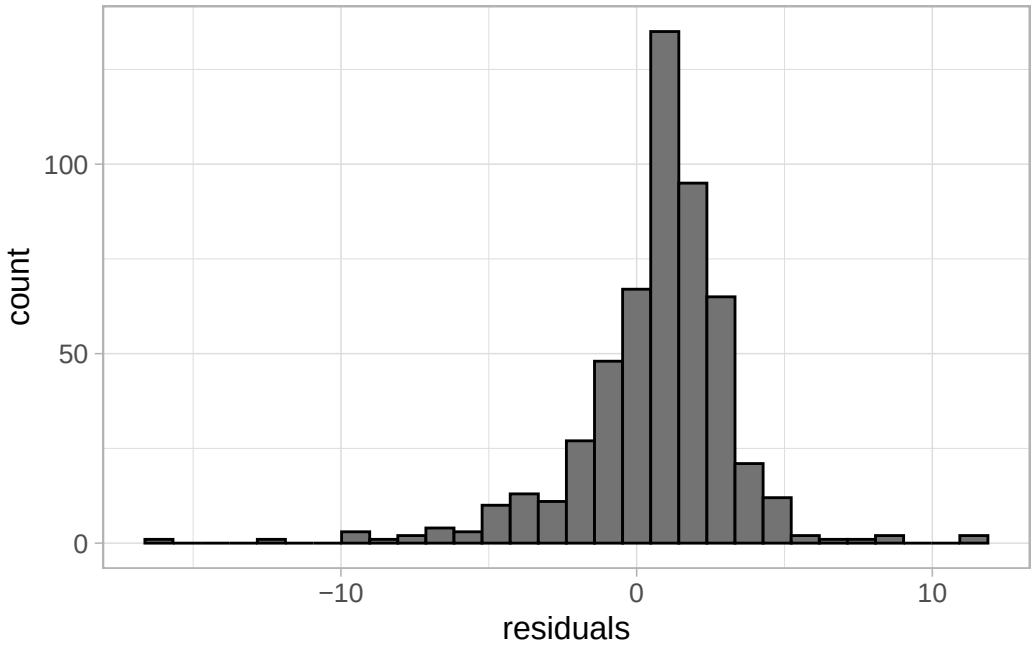
In R, the Pearson residuals can be calculated and plotted using the `residuals` function and specifying the `type` argument:

```
pearson_resid <- tibble(glm_mort_rate$model) %>%
  mutate(residuals = residuals(glm_mort_rate, type = "pearson"),
         id = row_number(.))

ggplot(data = pearson_resid) +
  geom_point(aes(y = residuals, x = id)) +
  theme_light(base_size = 12)
```




```
ggplot(data = pearson_resid) +
  geom_histogram(aes(x = residuals), colour = "black", fill = "grey45") +
  theme_light(base_size = 12)
```



Both plots show a 'funnel' shape, showing that the poisson assumption is not valid for this data. We can refer to the data to try and examine why these observations are not as well represented by the model as other points.

4.4.1.2 Deviance residuals

The deviance was introduced in Section 3.4.2 as a model comparison tool. Deviance quantifies how much the current model deviates from a hypothetical (and totally useless) full model. The lower the deviance, the better the model fits the sample data. The deviance alone is not particularly useful as the full model overfits the sample and is not able to provide inferences to the target population it is drawn from. However, it can be combined with other information to give insights about the model fit.

Deviance residuals are estimated by multiplying the square root of the deviance contribution, d_i of an observation by

- +1 if the observed count is higher than the predicted count,
- 0 if the observed and predicted counts are equal, or
- -1 if the observed count is lower than the predicted count.

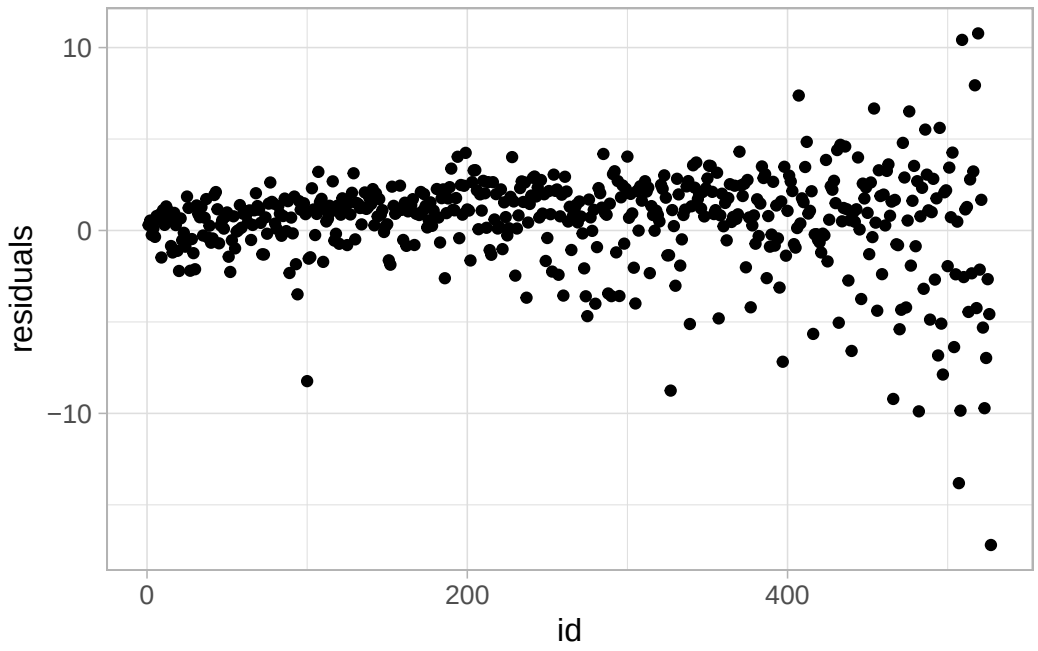
For poisson regression, the deviance contribution for observation i is:

$$d_i = 2\left[y_i log\left(\frac{y_i}{\hat{y}_i}\right) - (y_i - \hat{y}_i)\right]$$

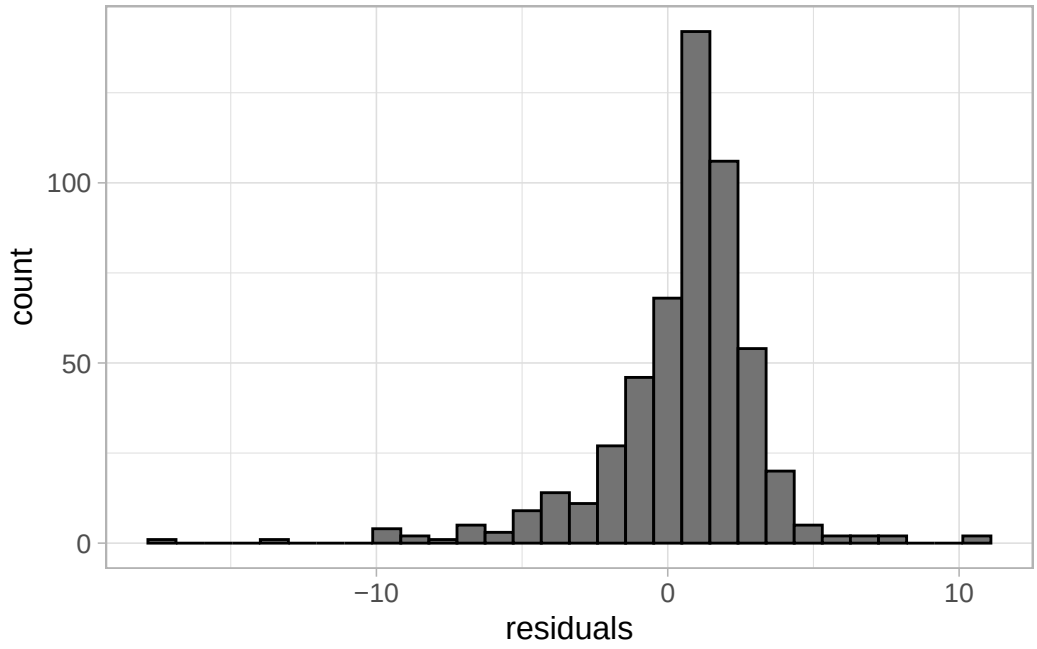
Large deviance residuals indicate that the model is not fitting an observation well. Plotting these deviance residuals can help identify potential outliers, influential values, or could help improve a model by highlighting a group that is not represented by the current model. In R, the deviance residuals are calculated (and plotted) as follows:

```
dev_resid <- tibble(glm_mort_rate$model) %>%
  mutate(residuals = residuals(glm_mort_rate, type = "deviance"),
         id = row_number(.))

ggplot(data = dev_resid) +
  geom_point(aes(y = residuals, x = id)) +
  theme_light(base_size = 12)
```




```
ggplot(data = dev_resid) +
  geom_histogram(aes(x = residuals), colour = "black", fill = "grey45") +
  theme_light(base_size = 12)
```



Deviance residuals produce similar results as Pearson residuals when using poisson regression. However, they provide a more generalised model check for regression types other than poisson.

> **Note**
>
> In reality, we would not use both the Pearson and deviance residuals when checking a regression model. The choice of residual depends on the intention of the check and whether we are communicating the results. The deviance residuals are presented here to give a generalised tool that can be applied to other GLMs.

4.4.2 Equidispersion

Both the Pearson and deviance residual plots showed non-equal variance of residuals. This could be an indication of overdispersion. Overdispersion occurs when the poisson assumption that the mean and variance of the outcome are equal is not valid. When this is the case, other more flexible models may be required.

To test for overdispersion, we can use the `dispersiontest` function which is part of the `{AER}` package in R. This function tests the hypothesis that the outcome mean (μ) and variance ($var(y)$) are equal, against an alternative that the variance takes the form:

$$var(y) = (1 + \alpha) \times \mu = dispersion \times \mu$$

The output provides an estimate of the dispersion parameter (which would take the value 0 if the mean and variance are equal) and a p-value testing the null hypothesis of equidispersion.

```
library(AER)

dispersiontest(glm_mort_rate, trafo = 1)
```

```
    Overdispersion test

data:  glm_mort_rate
z = 7.6555, p-value = 9.627e-15
alternative hypothesis: true alpha is greater than 0
sample estimates:
   alpha
6.235632
```

The results show clear overdispersion. On average, the variance is 7.24 times higher than the mean. The p-value is too small to be printed in its entirety, indicating that this overdispersion is statistically significant.

A poisson model would not be appropriate in this case. Other models, such as a quasipoisson or negative binomial model, which allow for unequal mean and variance, may be more appropriate.

Exercise 3

Using the data, fit an appropriate model to answer the research question:

Is cancer mortality associated with poverty levels in the USA?

Ensure that the mode contains any variables you consider necessary, and check that it is valid before using it to answer the research question.

> Exercise hint
>
> As with previous questions, plot the outcome and variables you believe may be important covariates to generate hypotheses about best fitting models.
> Use model comparison techniques such as information criterions and prediction errors to

identify the most parsimonious model, ensuring that all variables that need to be in the model are present.
Test model assumption, including checking for multicollinearity using `vif`, plotting Pearson residuals, and checking for equidispersion.
If the model is still overdispersed, try using `family = quasipoisson` instead, which assumes the variance is proportional to the mean, rather than equal. The `summary` of this model will include a dispersion parameter estimate, but model output is interpreted in the same way as poisson regression.
If you are REALLY stuck, an example solution can be found here.

5 Discussion

Generalised linear models are powerful statistical tools that can be applied to a wide range of data and situations. The choice of the most appropriate model to address a research question will depend on the type of outcome, but also:

- The intention of the model: which variables must be included to answer the research question?
- Common sense and background knowledge: what do we know about the context of the data and what are the known confounders?
- Parsimony: which model provides the simplest solution to our problem without losing any information?

Do not choose a regression model solely based on p-values!!

All models must be checked to ensure that any assumptions are met and the results are valid. All GLMs require observations to be independent of one another. This means that there is no clustering, repeated measures, or autocorrelation within the data. Where this assumption is not valid, multilevel models (also known as mixed effect, random effect, GLMMs, or hierarchical models) should be considered.

GLMs assume that the relationships between covariates and the (link-transformed) outcome are linear. Where this is not the case, covariates can be transformed before they are included into the model, for example polynomial regression. Where the relationship is more complex or unknown, consider generalised additive models (GAMs), which are able to for non-linear data.

Finally, note that the models shown in these notes and exercise solutions are not definitive. Choice of model is often subjective and context-specific.

A Data description

A.1 Palmer Penguins

Information about this data are available from palmerpenguins

```
data(penguins, package = "palmerpenguins")
```

```
skimr::skim(penguins)
```

Table A.1: Data summary

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
species	0	1.00	FALSE	3	Ade: 152, Gen: 124, Chi: 68
island	0	1.00	FALSE	3	Bis: 168, Dre: 124, Tor: 52
sex	11	0.97	FALSE	2	mal: 168, fem: 165

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
bill_length_mm	2	0.99	43.92	5.46	32.1	39.23	44.45	48.5	59.6	
bill_depth_mm	2	0.99	17.15	1.97	13.1	15.60	17.30	18.7	21.5	
flipper_length_mm	2	0.99	200.92	14.06	172.0	190.00	197.00	213.0	231.0	
body_mass_g	2	0.99	4201.75	801.95	2700.0	3550.00	4050.00	4750.0	6300.0	
year	0	1.00	2008.03	0.82	2007.0	2007.00	2008.00	2009.0	2009.0	

A.2 Cancer Registry

```
cancer_reg <- readr::read_csv(here::here("data/cancer_reg.csv"))
```

```
Rows: 527 Columns: 20
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (3): County, state, group
dbl (17): mortality_rate, number_death, cancer_incidence, number_cases, popu...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
skimr::skim(cancer_reg)
```

Table A.4: Data summary

Name	cancer_reg
Number of rows	527
Number of columns	20
_____	
Column type frequency:	
character	3
numeric	17
_____	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
County	0	1	16	38	0	527	0
state	0	1	4	14	0	48	0
group	0	1	7	7	0	3	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
mortality_rate	0	1	181.19	26.75	66.30	163.35	181.80	197.75	292.50	
number_death	0	1	119.08	155.31	3.00	28.00	58.00	141.50	914.00	
cancer_incidence	0	1	450.87	50.24	211.10	421.55	453.55	487.15	630.40	
number_cases	0	1	409.08	568.85	6.00	76.00	158.00	453.50	2841.00	
population_2015	0	1	59408.13	90106.98	1130.00	11343.50	25594.00	64463.00	734871.00	
age	0	1	41.00	5.02	22.30	38.10	40.90	43.80	56.60	
income	0	1	45918.68	11779.18	23047.00	38017.50	44065.00	51414.50	108477.00	
poverty	0	1	17.53	6.61	4.20	12.60	16.70	21.30	45.10	
household	0	1	2.47	0.46	0.02	2.37	2.50	2.63	3.97	
married	0	1	51.56	6.91	23.10	48.10	52.30	56.30	69.20	
unemployed	0	1	8.09	3.25	0.70	5.90	8.00	9.70	22.60	
medicare	0	1	19.63	5.91	2.60	15.50	19.40	23.45	41.40	
white	0	1	83.10	17.16	11.01	75.81	90.24	95.41	100.00	
black	0	1	10.36	16.07	0.00	0.68	2.28	12.45	84.87	
asian	0	1	0.99	1.67	0.00	0.24	0.52	1.13	21.28	
BirthRate	0	1	5.58	1.98	0.00	4.54	5.38	6.42	17.88	
random_n	0	1	0.19	0.10	0.00	0.10	0.19	0.28	0.35	

B Exercise solutions

All the exercise solutions here are suggestions and are not exhaustive. Data exploration and model building are often subjective processses which are determined by a person's prior experience and the context of a project.

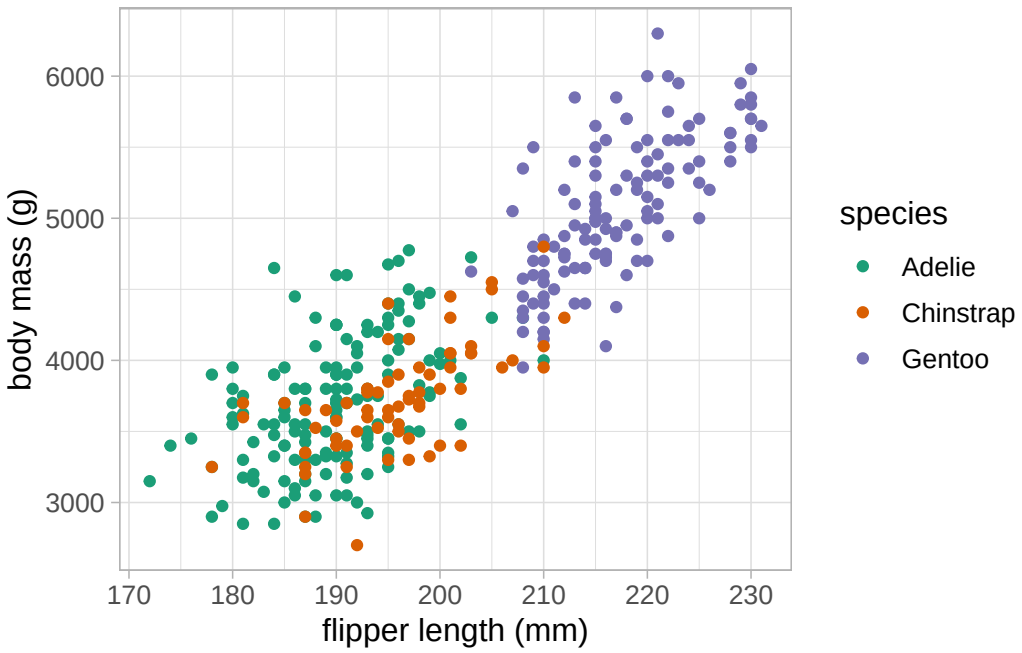
B.1 Exercise 1

Using appropriate visualisations, investigate whether there are other variables that may explain differences in body mass. Consider whether any of these variables may be confounding the relationship between body mass and flipper length, and whether they should be included in the model.

Solution

Body mass and flipper length are both likely to differ between penguin species. Changing the colour of points for each species will allow us to visualise these differences:

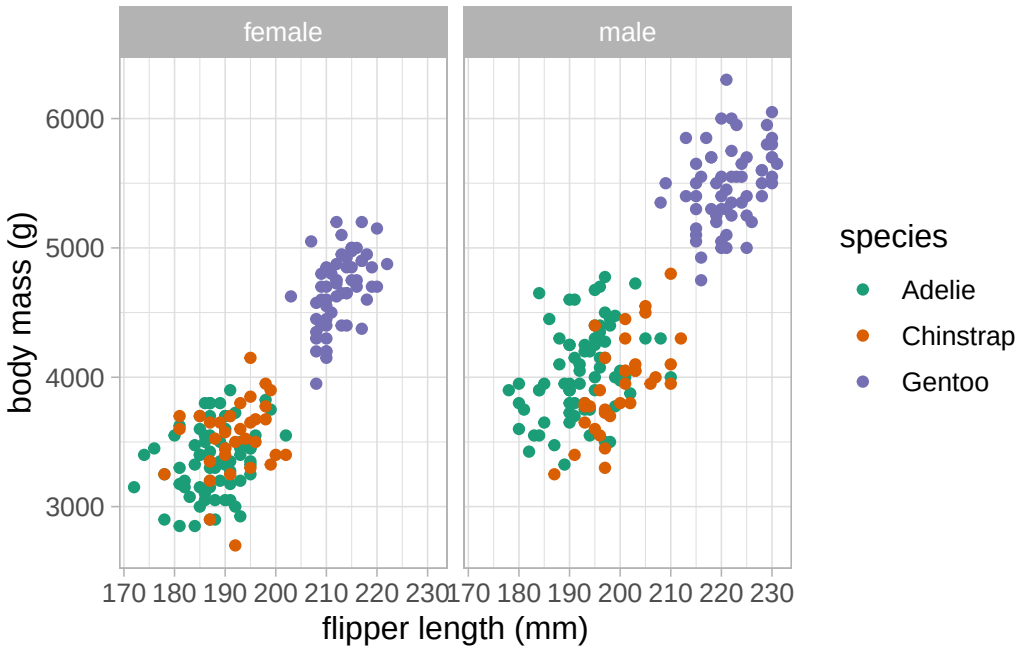
```
ggplot(data = penguins) +
  geom_point(aes(x = flipper_length_mm, y = body_mass_g, colour = species)) +
  scale_colour_brewer(palette = "Dark2") +
  labs(x = "flipper length (mm)", y = "body mass (g)") +
  theme_light(base_size = 12)
```



Species is clearly strongly associated with both body mass and flipper length, although the gradient of these associations appear similar across species.

This scatterplot could be extended to investigate whether these trends differ between sexes. Adding an additional variable to the previous scatterplot may overload it, making the relationships difficult to interpret. Instead, we could facet the graphs, showing a scatterplot per sex on the same graph area, with the same axes:

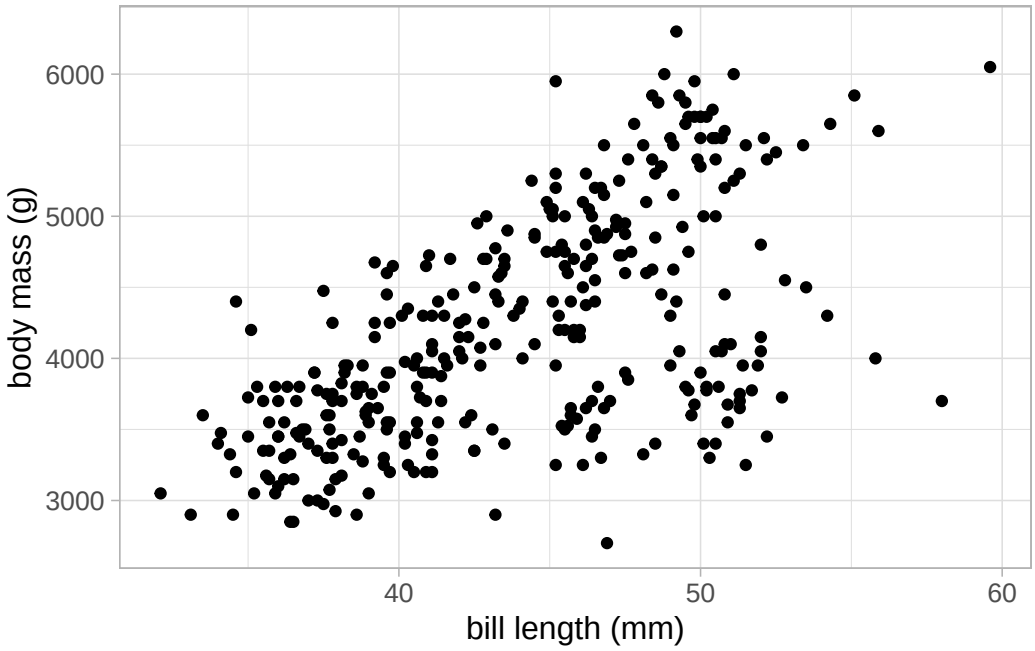
```
ggplot(data = na.omit(penguins)) +
  geom_point(aes(x = flipper_length_mm, y = body_mass_g, colour = species)) +
  scale_colour_brewer(palette = "Dark2") +
  labs(x = "flipper length (mm)", y = "body mass (g)") +
  facet_wrap(vars(sex), ncol = 2) +
  theme_light(base_size = 12)
```



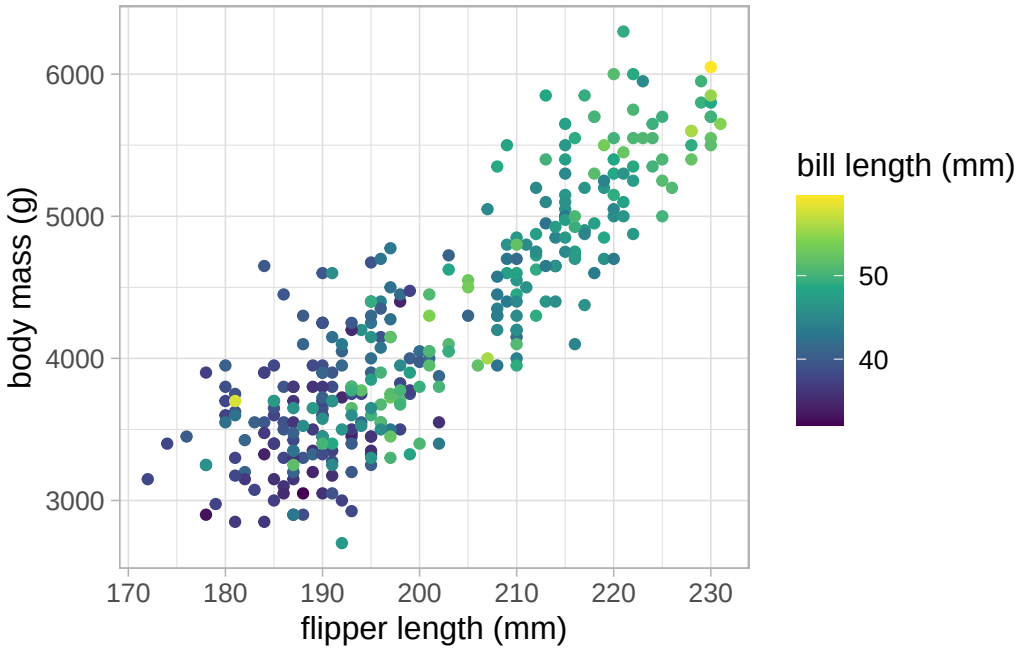
Here, it appears that the male pengions are larger on average than the females. The relationships between body mass, flipper length and species appear equal between sexes.

The data also contains information about penguins' bill length and depth which may also by a predictor of body mass. These can be plotted against body mass in a scatterplot, replacing flipper length, or could be included into the original scatterplot by using a continuous colour scale.

```
ggplot(data = penguins) +
  geom_point(aes(x = bill_length_mm, y = body_mass_g)) +
  labs(x = "bill length (mm)", y = "body mass (g)") +
  theme_light(base_size = 12)
```




```
ggplot(data = penguins) +
  geom_point(aes(x = flipper_length_mm, y = body_mass_g,
                 colour = bill_length_mm)) +
  scale_colour_viridis_c(name = "bill length (mm)") +
  labs(x = "flipper length (mm)", y = "body mass (g)") +
  theme_light(base_size = 12)
```



There appears to be a positive association between bill length and body mass, but it is not as strong as the one between flipper length and body mass.

B.2 Exercise 2

Using everything you have learned up to this point, use linear regression to answer the research question posed earlier:

Is body mass of penguins in the Palmer Archipelago related to their flipper size?

Solution

From our research question, we know that our model must have body mass as the outcome and flipper length as an explanatory variable. Previous exploratory analysis showed that sex and bill length were also associated to body mass. We can add these variables into a linear model and consider whether it improves the model fit. We may also try removing species from the model as this appeared to lead to heteroskedasticity in the residuals:

```
lm_flipper <- lm(body_mass_g ~ flipper_length_mm, data = penguins)          ①

lm_flipper_sex <- lm(body_mass_g ~ flipper_length_mm + sex,
                     data = penguins)                                       ②
```

```r
lm_flipper_bill <- lm(body_mass_g ~ flipper_length_mm +
                        bill_length_mm, data = penguins)                    ③

lm_full <- lm(body_mass_g ~ flipper_length_mm + sex +
                bill_length_mm, data = penguins)                            ④
```

① We will begin with the simplest possible model for comparison, one containing just body mass and flipper length.
② A model with sex instead of species which was found to be related to body mass.
③ Add bill length to see if this improves the initial model.
④ A model with all potential exploratory variables (besides species).

We can compare these models in various ways, including the adjusted R-squared, information criterions, and prediction errors. Below is a table containing these comparisons for each model.

model	adjusted R-squared	AIC	RMSE
flipper only	0.7582837	5062.855	393.1236
flipper + sex	0.8046607	4862.484	354.2762
flipper + bill length	0.7585415	5063.482	392.3357
flipper + bill length + sex	0.8047466	4863.327	353.6612

The model containing flipper length and sex slightly outperformed the full momdel according to the adjusted R-squared and AIC, but had a slightly lower RMSE. As the bill length is not important to our research question and the model is not being used for prediction, we will choose the simplest possible model and remove bill length.

Before we use this model to answer our research question, we must ensure that the model is valid. Remember, the assumptions we need to check are **L**inearity, **I**ndependent covariates, **N**ormally distributed residuals, with **E**qual variance.

```r
vif(lm_flipper_sex)
```

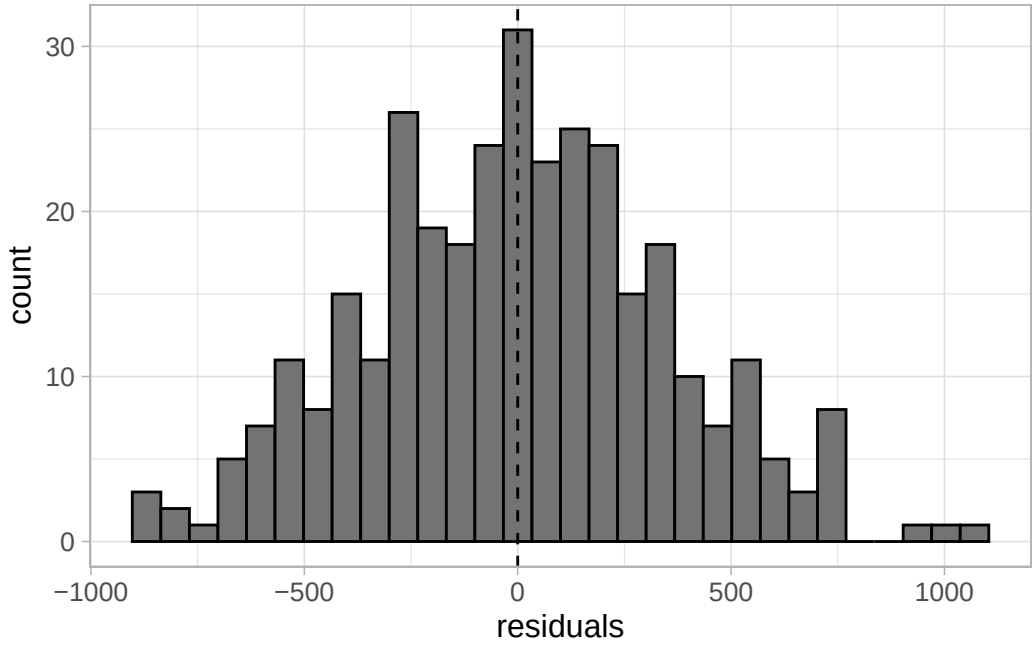
```
flipper_length_mm               sex
         1.069646          1.069646
```

All VIFs are very low, indicating no issues with multicollinearity.

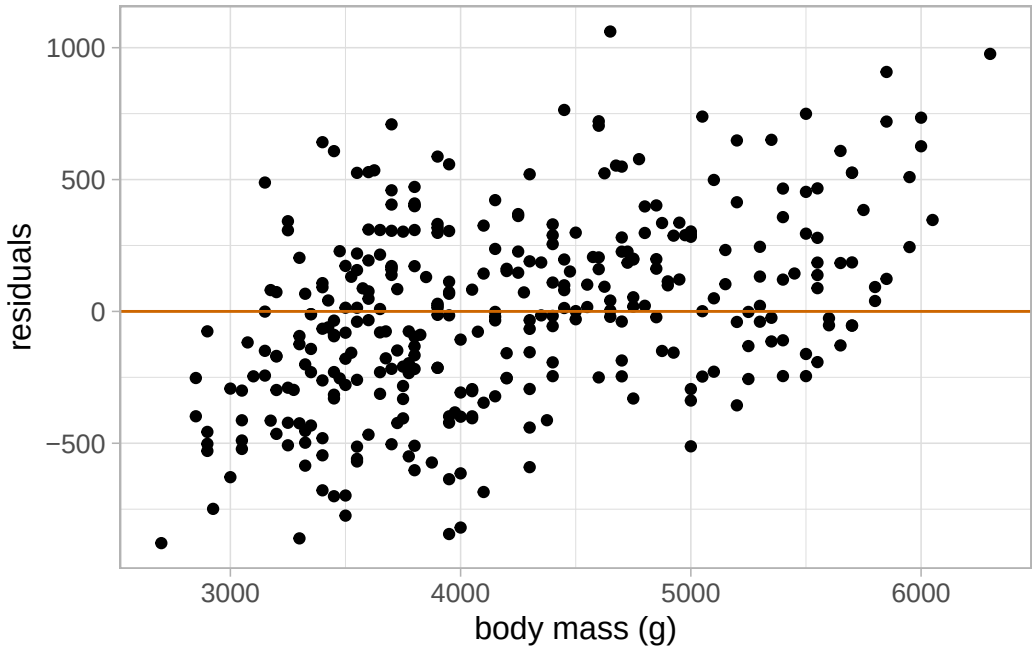
```r
lm_flipper_sex_resid <- lm_flipper_sex$model %>%
  mutate(residuals = residuals(lm_full))

ggplot(data = lm_flipper_sex_resid) +                                    ①
  geom_histogram(aes(x = residuals), colour = "black", fill = "grey45") +
  geom_vline(xintercept = 0, linetype = "dashed") +
  theme_light(base_size = 12)
```

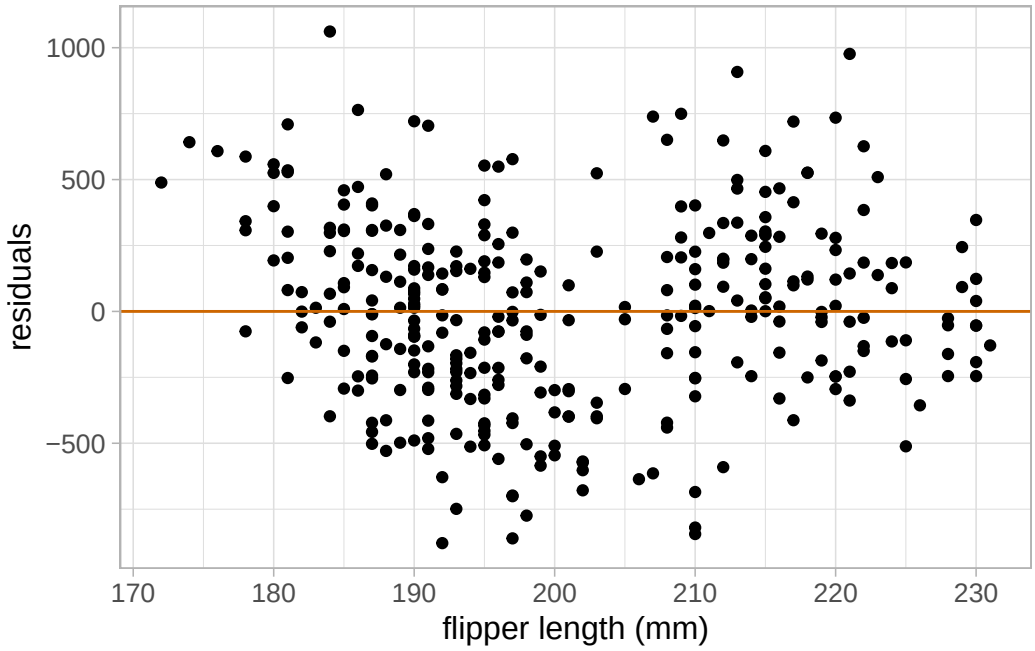
③ Plot the residuals against each covariate to check the linearity assumption.



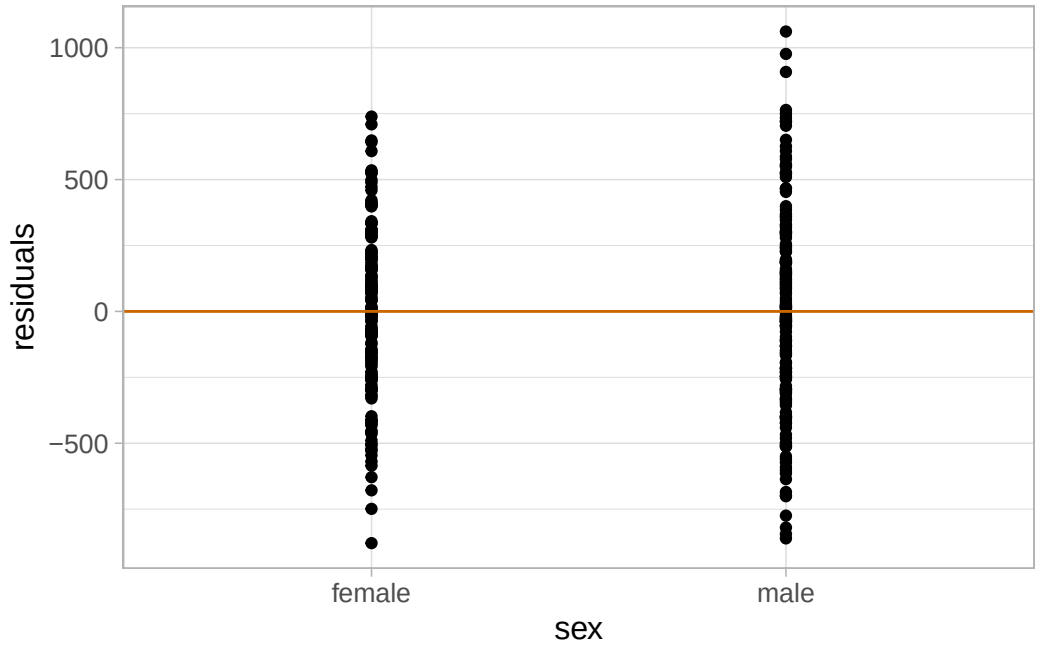

```r
ggplot(data = lm_flipper_sex_resid) +                                    ②
  geom_point(aes(x = body_mass_g, y = residuals)) +
  geom_hline(yintercept = 0, colour = "darkorange3") +
  labs(x = "body mass (g)", y = "residuals") +
  theme_light(base_size = 12)
```




```ggplot(data = lm_flipper_sex_resid) +  ③
  geom_point(aes(x = flipper_length_mm, y = residuals)) +
  geom_hline(yintercept = 0, colour = "darkorange3") +
  labs(x = "flipper length (mm)", y = "residuals") +
  theme_light(base_size = 12)
```




```
ggplot(data = lm_flipper_sex_resid) +
  geom_point(aes(x = sex, y = residuals)) +
  geom_hline(yintercept = 0, colour = "darkorange3") +
  labs(x = "sex", y = "residuals") +
  theme_light(base_size = 12)
```



The residuals are approximately normal, their variance is approximately constant, and there is no evidence to suggest that the linearity assumption would not be valid. Therefore, we can use this model to answer our research question.

```
summary(lm_flipper_sex)
```

```
Call:
lm(formula = body_mass_g ~ flipper_length_mm + sex, data = penguins)

Residuals:
    Min      1Q  Median      3Q     Max
-910.28 -243.89   -2.94  238.85 1067.73

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       -5410.300    285.798 -18.931  < 2e-16 ***
flipper_length_mm    46.982      1.441  32.598  < 2e-16 ***
sexmale             347.850     40.342   8.623 2.78e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 355.9 on 330 degrees of freedom
```

```
  (11 observations deleted due to missingness)
Multiple R-squared:  0.8058,    Adjusted R-squared:  0.8047
F-statistic: 684.8 on 2 and 330 DF,  p-value: < 2.2e-16
```

```
confint(lm_flipper_sex)
```

```
                       2.5 %      97.5 %
(Intercept)       -5972.51535 -4848.08510
flipper_length_mm    44.14697    49.81738
sexmale             268.49120   427.20930
```

Based on these results, we can infer that there is a significantly positive association between flipper length and body mass of the Palmer penguins. On average, body mass is expected to increase by 46.98g for every 1mm increase in flipper length, We are 95% confident that this increase is between 44.15g and 49.82g in the target population.

B.3 Exercise 3

Using the data, fit an appropriate model to answer the research question:

Is cancer mortality associated with poverty levels in the USA?

Ensure that the mode contains any variables you consider necessary, and check that it is valid before using it to answer the research question.

Solution

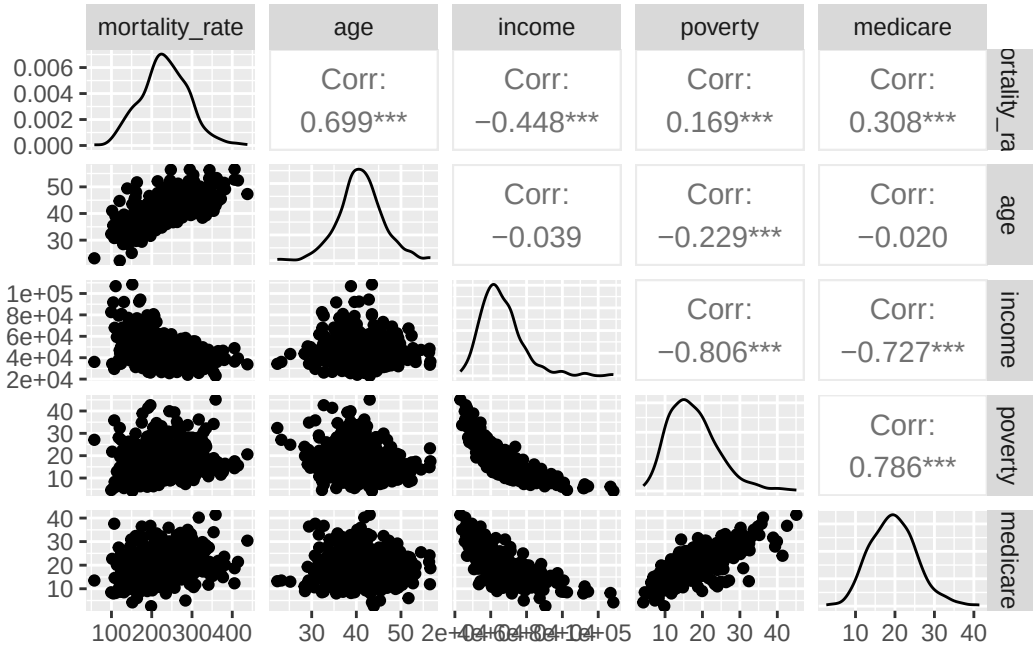
The model we require will have cancer mortality (number of deaths with population as an offset) as an outcome and must contain a measure of poverty to answer the research question. Other variables from the data that may be important include the average age of a county, their access to medicare, and possibly income (although this will likely be highly correlated to poverty).

First, we should explore the data and plot these variables to understand their bivariate relationships. Rather than do this manually, we could use the `ggpair` function from the `GGally` package:

```
library(GGally)

cancer_reg_clean <- cancer_reg %>%
  mutate(mortality_rate = (number_death / population_2015) * 10^5,
         offset_rate = log(population_2015 / 10^5)) %>%
  select(mortality_rate, number_death, population_2015, age, income, poverty,
         medicare, offset_rate)

cancer_reg_clean %>%
  select(-number_death, -population_2015, -offset_rate) %>%
  ggpairs()
```



As expected, both income and medicare access are highly correlated to poverty. Although this does not necessarily make them **dependent** on each other, the interpretation of mode coefficients can be complicated by their inclusion. To remove this issue, we can add age to the original model to see if it improves the fit.

```
pois_pov <- glm(number_death ~ poverty + offset(offset_rate),
                data = cancer_reg_clean, family = poisson)

pois_pov_age <- glm(number_death ~ poverty + age + offset(offset_rate),
                    data = cancer_reg_clean, family = poisson)
```

The adjusted R-squared measure is only appropriate for linear models. However, we can still use information criterions and prediction errors to compare models:

model	AIC	RMSE
poverty only	7041.657	192.3616
poverty + age	4330.177	192.3497

Adding age appears to vastly improve the AIC but only slightly improve prediction according to the RMSE. The poisson model containing just poverty showed strong evidence of overdispersion, therefore we must check this model to find whether the addition of age has removed the issue:

```
dispersiontest(pois_pov_age, trafo = 1)
```

```
    Overdispersion test

data:  pois_pov_age
z = 4.0437, p-value = 2.63e-05
alternative hypothesis: true alpha is greater than 0
sample estimates:
   alpha
1.141831
```

Although the dispersion parameter is lower than the poverty only model, there is still evidence of overdispersion. Therefore, a quasipoisson model may be more appropriate:

```
quasi_pov_age <- glm(number_death ~ poverty + age  + offset(offset_rate),
                     data = cancer_reg_clean, family = quasipoisson)

summary(quasi_pov_age)
```

```
Call:
glm(formula = number_death ~ poverty + age + offset(offset_rate),
    family = quasipoisson, data = cancer_reg_clean)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 3.134120   0.058920   53.19   <2e-16 ***
```

```
poverty     0.018326   0.001076   17.02   <2e-16 ***
age         0.047598   0.001326   35.89   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasipoisson family taken to be 2.175137)

    Null deviance: 4101.4  on 526  degrees of freedom
Residual deviance: 1174.7  on 524  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 4
```

The `summary` of quasipoisson models contains an estimate of the dispersion parameter. Quasipoisson and Poisson models are equivalent when the dispersion parameter is 1. As the parameter was estimated above that, this is a clear indication the poisson model was not appropriate for this data.

C Mathematical Derivation of OLS

The coefficient values for simple linear regression can be derived using a procedure known as Ordinary Least Squares (OLS).

For an assumed model:

$$y_i = \beta_0 + x_i\beta_1 + \epsilon_i$$

We can generate predictions in the form:

$$\hat{y_i} = \beta_0 + x_i\beta_i$$

The residuals for our model $\epsilon_i = y_i - \hat{y_i}$ can be written as:

$$\epsilon_i = y_i - \beta_0 - \beta_1 x_i$$

To find the sum of squared residuals (RSS) for all records we have:

$$\sum_{i=1}^{N} \epsilon_i^2 = \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_i)^2$$

We want to find β_0 and β_1 such that we minimise the RSS. This occurs when we set the partial derivative with respect to both β_0 and β_1 to zero.

$$\frac{\partial}{\partial \beta_0} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_i)^2 = 0$$

$$\frac{\partial}{\partial \beta_1} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_i)^2 = 0$$

This gives us:

$$-2\sum_{i=1}^{N}(y_i - \beta_0 - \beta_1 x_i) = 0 \tag{C.1}$$

$$-2x_i\sum_{i=1}^{N}(y_i - \beta_0 - \beta_1 x_i) = 0 \tag{C.2}$$

From Equation C.1 we can ignore the constant -2 and redistribute the summation.

$$\sum_{i=1}^{N} y_i - \sum_{i=1}^{N}\beta_0 - \sum_{i=1}^{N}\beta_1 x_i = 0$$

Giving:

$$N\bar{y} - N\beta_0 - N\beta_1\bar{x} = 0$$

Which simplifies to:

$$\beta_0 = \bar{y} - \beta_1\bar{x} \tag{C.3}$$

From Equation C.2 we can ignore the -2 and distribute the x_i:

$$\sum_{i=1}^{N} y_i x_i - \beta_0 x_i - \beta_1 x_i^2 = 0$$

Substituting Equation C.3 and distributing the summation we get:

$$\sum_{i=1}^{N} x_i y_i - \bar{y}\sum_{i=1}^{N} x_i + \beta_1\bar{x}\sum_{i=1}^{N} x_i - \beta_1\sum_{i=1}^{N} x_i^2 = 0$$

Solving for β_1 gives:

$$\beta_1 = \frac{\sum_{i=1}^{N} x_i y_i - N\bar{x}\bar{y}}{\sum_{i=1}^{N} x_i^2 - N\bar{x}^2}$$

Which simplifies to:

$$\beta_1 = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{C.4}$$

Now we have a closed-form expression for both β_0 Equation C.3 and β_1 Equation C.4.

> **Note**
>
> This can extend to more than simple linear regression but would involve an expression in matrix notation. For more complex regression models, other methods are used such as maximum likelihood estimation (MLE). This is beyond the scope of this course.

Let's try to manually verify this by calculating the regression coefficients by hand. We can compare this to the R output in Section 3.1.

```
library(dplyr)
library(palmerpenguins)

dat <- penguins |>
    select(body_mass_g, flipper_length_mm) |>
    na.omit()

x <- dat$flipper_length_mm
y <- dat$body_mass_g

b_1 <- sum((x - mean(x)) * (y - mean(y))) / sum((x - mean(x))^2)
b_0 <- mean(y) - b_1 * mean(x)

b_0
```

```
[1] -5780.831
```

```
b_1
```

```
[1] 49.68557
```

D Setup

D.1 Setting up R

To get started, ensure you have a recent version or R and RStudio installed.

D.1.1 Step 1: Install R

To install R head to https://cran.rstudio.com/ and follow the instructions for your operating system.

D.1.2 Step 2: Install RStudio

Next, install RStudio Desktop IDE at https://posit.co/download/rstudio-desktop/.

D.1.3 Packages

To install the required packages, we run the `install.packages("packagename")` function.

```
# Install the required R packages for our analysis (first time use only)
install.packages("tidyverse")
install.packages("palmerpenguins")
install.packages("Metrics")
install.packages("car")
install.packages("skimr")
install.packages("AER")
```

> **Note**
>
> The command `install.packages()` is only required the first time loading a new package or following any substantial updates. The `library()` command must be run every time you start an R session. To save potential issues arising from unloaded packages, put any `library()` commands at the beginning of any script file.

You should be able to now run the following commands:

```
# Load the installed packages at the start of each session
library(tidyverse)
library(palmerpenguins)
library(Metrics)
library(car)
library(skimr)
library(AER)
```