Joshua French

Joshua French

2021-09-23

Contents

Preliminaries **5**

1 R Foundations **7**
1.1 What is R? 9
1.2 Where to get R (and R Studio Desktop) 9
1.3 R Studio Layout 9
1.4 Running code, scripts, and comments 9
1.5 Packages 9
1.6 Getting help 9
1.7 Data types and structures 9
1.8 Assignment 9
1.9 Vectors 9
1.10 Helpful functions 9
1.11 Data Frames 9
1.12 Logical statements 9
1.13 Subsetting with logical statements 9
1.14 Ecosystem debate 9

2 Data exploration **11**
2.1 Data analysis process 12
2.2 Data exploration 12
2.3 Kidney Example 12
2.4 Visualizing data with **base** graphics 12
2.5 Visualizing data with **ggplot2** 12
2.6 Summary of data exploration 12

3 Review of probability, random variables, and random vectors **13**
3.1 Probability Basics 15
3.2 Random Variables 15
3.3 Multivariate distributions 15
3.4 Random vectors 15
3.5 Properties of transformations of random vectors 15
3.6 Multivariate normal (Gaussian) distribution 15

3.7 Example 1 15
3.8 Example 2 15

4 Useful matrix facts 17
4.1 Notation 18
4.2 Basic mathematical properties 18
4.3 Transpose and related properties 18
4.4 Special matrices 18
4.5 Matrix inverse 18
4.6 Matrix derivatives 18

5 Linear model definition and properties 19
5.1 Background and terminology 20
5.2 Goals of regression 20
5.3 Regression for Pearson's height data 20
5.4 Definition of a linear model 20
5.5 Summarizing the components of a linear model 20
5.6 Types of regression models 20
5.7 Standard linear model assumptions and implications 20
5.8 Mathematical interpretation of coefficients 20
5.9 Exercises 20

6 Fitting a linear model 21
6.1 OLS estimation of the simple linear regression model 21
6.2 Penguins simple linear regression example 27
6.3 Fitting a linear model using R 31

7 Interpreting a fitted linear model 35

8 Categorical predictors 37
8.1 Indicator/dummy variables 37
8.2 Common of linear models with categorical predictors 37

9 Assessing and addressing collinearity 39

Preliminaries

Placeholder

Chapter 1

R Foundations

Placeholder

1.1 What is R?

1.2 Where to get R (and R Studio Desktop)

1.3 R Studio Layout

1.4 Running code, scripts, and comments

1.4.1 Example

1.5 Packages

1.5.1 Example

1.6 Getting help

1.6.1 Example

1.7 Data types and structures

1.7.1 Basic data types

1.7.2 Other important object types

1.7.2.1 Numeric

1.7.2.2 NULL

1.7.2.3 NA

1.7.2.4 Functions

1.7.3 Data structures

1.8 Assignment

1.8.1 Example

1.9 Vectors

1.9.1 Creation

1.9.2 Creating patterned vectors

1.9.3 Example

1.9.4 Example

1.9.5 Categorical vectors

1.9.6 Example

1.9.7 Extracting parts of a vector

1.10 Helpful functions

1.10.1 General functions

1.10.2 Example

Chapter 2

Data exploration

Placeholder

2.1 Data analysis process

2.1.1 Problem Formulation

2.1.2 Data collection

2.2 Data exploration

2.2.1 Numerical summaries of data

2.2.2 Visual summaries of data

2.2.3 What to look for

2.3 Kidney Example

2.3.1 Numerically summarizing the data

2.3.2 Cleaning the data

2.4 Visualizing data with base graphics

2.4.1 Histograms

2.4.2 Density plots

2.4.3 Index plots

2.4.4 Bivariate scatter plots

2.4.5 Bivariate boxplots

2.4.6 Multiple plots in one figure

2.5 Visualizing data with ggplot2

2.5.1 A ggplot2 histogram

2.5.2 A ggplot2 density plot

2.5.3 A ggplot2 scatter plot

2.5.4 Scaling ggplot2 plots

2.5.5 Facetting in `ggplot2`

2.5.6 Summary of ggplot2

2.6 Summary of data exploration

Chapter 3

Review of probability, random variables, and random vectors

Placeholder

3.1 Probability Basics

3.2 Random Variables

3.2.1 Discrete random variables

3.2.1.1 Example (Bernoulli)

3.2.2 Continuous random variables

3.2.3 Useful facts for transformation of random variables

3.3 Multivariate distributions

3.3.1 Basic properties

3.3.2 Marginal distributions

3.3.3 Independence of random variables

3.3.4 Conditional distributions

3.3.5 Covariance

3.3.6 Useful facts for transformations of multiple random variables

3.4 Random vectors

3.4.1 Definition

3.4.2 Mean, variance, and covariance

3.5 Properties of transformations of random vectors

3.6 Multivariate normal (Gaussian) distribution

3.6.1 Definition

3.6.2 Useful facts

3.7 Example 1

3.7.1 Bernoulli distribution

3.7.2 Binomial distribution

3.7.3 Poisson Distribution

3.8 Example 2

3.8.1 Problem 1

3.8.2 Problem 2

3.8.3 Problem 3

3.8.4 Problem 4

Chapter 4

Useful matrix facts

Placeholder

4.1 Notation

4.2 Basic mathematical properties

4.2.1 Addition and subtraction

4.2.2 Scalar multiplication

4.2.3 Matrix multiplication

4.2.4 Associative property

4.2.5 Distributive property

4.2.6 No commutative property

4.3 Transpose and related properties

4.3.1 Definition

4.3.2 Transpose and mathematical operations

4.4 Special matrices

4.4.1 Square matrices

4.4.2 Identity matrix

4.4.3 Symmetric

4.4.4 Idempotent

4.5 Matrix inverse

4.6 Matrix derivatives

Chapter 5

Linear model definition and properties

Placeholder

5.1 Background and terminology

5.2 Goals of regression

5.3 Regression for Pearson's height data

5.4 Definition of a linear model

5.4.1 Basic construction and relationships

5.4.2 As a system of equations

5.4.3 Using matrix notation

5.5 Summarizing the components of a linear model

5.6 Types of regression models

5.7 Standard linear model assumptions and implications

5.8 Mathematical interpretation of coefficients

5.8.1 Coefficient interpretation in simple linear regression

5.8.2 Coefficient interpretation in multiple linear regression

5.9 Exercises

Chapter 6

Fitting a linear model

A linear model for a response variable Y using $p-1$ predictor variables $X_1, \dots, X_{p-1}$ may be described by the formula

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_{p-1} X_{p-1} + \epsilon,$$

where $\beta_0, \beta_1, \dots, \beta_{p-1}$ are regression coefficients and ϵ is the error.

In this chapter we focus on estimating the regression coefficients.

Fitting a regression model is the same thing as estimating the parameters of a simple linear regression model.

There are many different methods of parameter estimation in statistics: method-of-moments, maximum likelihood, Bayesian, etc.

The most common estimation method for linear models is known as **Ordinary Least Squares (OLS)** estimation. OLS estimation estimates the parameters with the values that minimize the residuals sum of squares (RSS).

6.1 OLS estimation of the simple linear regression model

In a simple linear regression context, we have n observed responses $Y_1, Y_2, \dots, Y_n$ and predictor values $x_1, x_2, \dots, x_n$.

Recall that for a simple linear regression model

$$Y = \beta_0 + \beta_1 X + \epsilon = E(Y|X) + \epsilon$$

with

$$E(Y|X) = \beta_0 + \beta_1 X.$$

We need to define some new notation and objects to define the RSS.

Let $\hat{\beta}_j$ denote the estimated value of β_j and the estimated mean response as a function of the predictor X is

$$\hat{E}(Y|X) = \hat{\beta}_0 + \hat{\beta}_1 X.$$

The **ith fitted value** is defined as

$$\hat{Y}_i = \hat{E}(Y|X = x_i) = \hat{\beta}_0 + \hat{\beta}_1 x_i.$$

Thus, the ith fitted value is the estimated mean of Y when the predictor $X = x_i$. More specifically, the ith fitted value is the estimated mean response of the ith observation.

The **ith residual** is defined as

$$\hat{\epsilon}_i = Y_i - \hat{Y}_i.$$

The ith residual is the difference between the response and estimated mean response of observation i.

The RSS of a regression model is the sum of its squared residuals.

The RSS for a simple linear regression model, as a function of the estimated regression coefficients, is

$$\begin{aligned}
RSS(\hat{\beta}_0, \hat{\beta}_1) &= \sum_{i=1}^{n} \hat{\epsilon}_i^2 \\
&= \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \\
&= \sum_{i=1}^{n} (Y_i - \hat{E}(Y|X = x_i))^2 \\
&= \sum_{i=1}^{n} (Y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i))^2.
\end{aligned}$$

The **fitted model** is the estimated model that minimizes the RSS. In a simple linear regression context, the fitted model is known as the **line of best fit**.

In Figure 6.1, we attempt to visualize the response values, fitted values, residuals, and line of best fit in a simple linear regression context. Notice that:

- The fitted values are the value returned by the line of best fit when it is evaluated at the observed predictor values. Alternatively, the fitted value for each observation is the y-value obtained when intersecting the line of best fit with a vertical line drawn from each observed predictor value.

- The residual is the vertical distance between each response value and the fitted value.
- The RSS seeks to minimize the sum of the squared vertical distances between the response and fitted values.

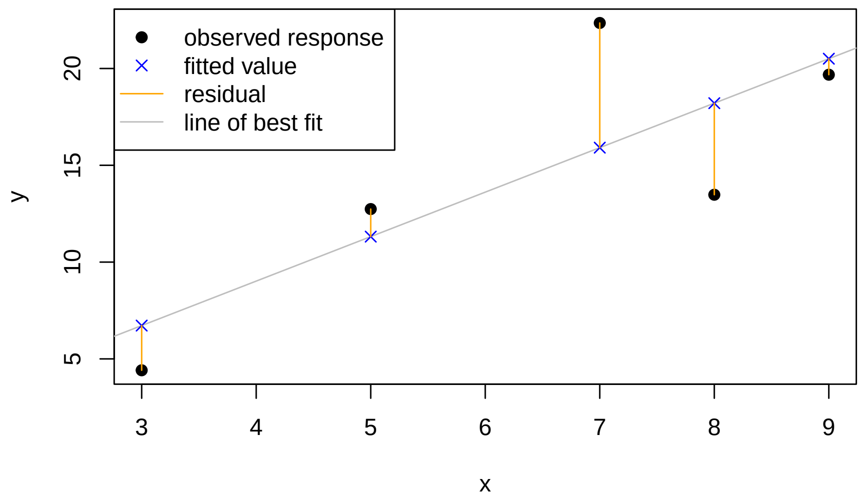


Figure 6.1: Visualization of the response values, fitted values, residuals, and line of best fit.

6.1.1 Visualizing the RSS as a function of the estimated coefficients

As we have attempted to emphasize through its notation, $RSS(\hat{\beta}_0, \hat{\beta}_1)$ is a function of $\hat{\beta}_0$ and $\hat{\beta}_1$. OLS estimation for the simple linear regression model seeks to find the values of the estimated coefficients that minimize the $RSS(\hat{\beta}_0, \hat{\beta}_1)$. In the example below, we visualize this three-dimensional surface to see how difficult it would be to optimize the RSS computationally .

Consider the Pearson and Lee's height data (`PearsonLee` in the **HistData** package) previously discussed. For that data set, we tried to model the child's height (`child`) based on the height of the child's parents (`parent`). Thus, our response variable is `child` and our predictor variable is `parent`. We seek to estimate the regression equation

$$E(\texttt{child} \mid \texttt{parent}) = \beta_0 + \beta_1 \texttt{parent}$$

with the values of $\hat{\beta}_0$ and $\hat{\beta}_1$ that minimize the associated RSS.

We first load the height data, extract the response and predictor and assign them the names `y` and `x`.

```r
# load height data
data(PearsonLee, package = "HistData")
# extract response and predictor variables from data set
y <- PearsonLee$child
x <- PearsonLee$parent
```

We now create a function that computes the RSS as a function of $\hat{\beta}_0$ and $\hat{\beta}_1$ (called `b0` and `b1`, respectively in the code below). The function takes the vector `b = c(b0, b1)`, extracts `b0` and `b1` from this vector, computes the fitted values (`yhat`) for the provided `b0` and `b1`, computes the corresponding residuals (`ehat`), and the returns the sum of the squared residuals, i.e., the RSS.

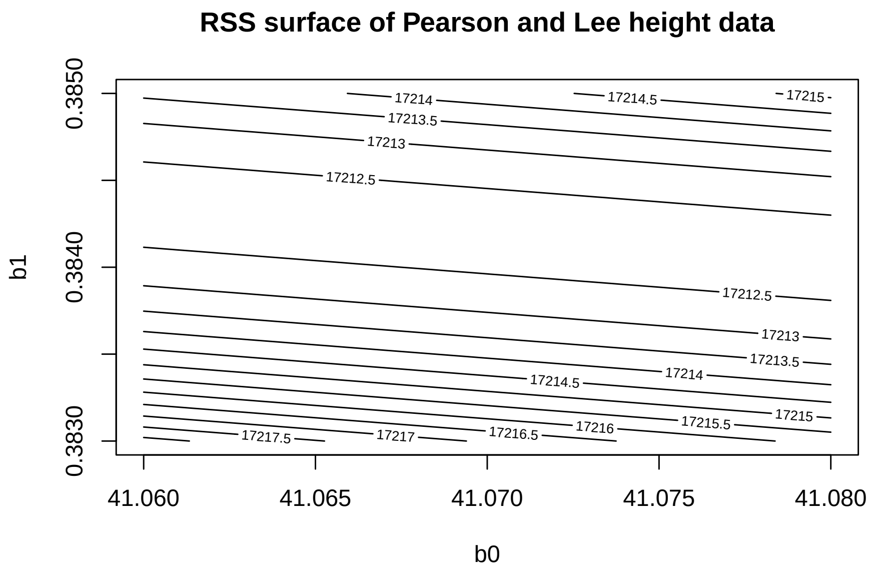
```r
# function to compute the RSS
# b = c(b0, b1)
compute_rss <- function(b) {
  b0 = b[1] # extract b0 from b
  b1 = b[2] # extract b1 from b
  yhat <- b0 + b1 * x # compute fitted values
  ehat <- y - yhat # compute residuals
  return(sum(ehat^2)) # return RSS
}
```

Next, we specify sequences of `b0` and `b1` values to consider for optimizing the RSS. We create a matrix, `rss_mat` to store the computed RSS for each combination of `b0` and `b1`. We then use a double `for` loop to evaluate the RSS for each combination of `b0` and `b1` in our sequences.

```r
# sequences of candidate b0 and b1 values
b0_seq <- seq(41.06, 41.08, len = 101)
b1_seq <- seq(0.383, 0.385, len = 101)
# matrix to store rss values
rss_mat <- matrix(nrow = length(b0_seq), ncol = length(b1_seq))
# use double loop to compute RSS for all combinations of b0_seq and b1_seq
# seq_along(b0_seq) returns the vector 1:length(b0_seq), but is safer
for (i in seq_along(b0_seq)) {
  for (j in seq_along(b1_seq)) {
    rss_mat[i, j] <- compute_rss(c(b0_seq[i], b1_seq[j]))
  }
}
```

We draw a contour plot of the RSS surface using the `contour` function.

```r
# draw a contour plot of the RSS surface
contour(x = b0_seq, y = b1_seq, z = rss_mat, xlab = "b0", ylab = "b1")
title("RSS surface of Pearson and Lee height data")
```



A contour plot uses contour lines to describe the height of the z dimension of a 3-dimensional (x, y, z) surface. Each line/contour indicates the height of the surface along that line. Note that in the graphic above, the contours are basically straight lines. There's no easily identifiable combinations of `b0` and `b1` the produce the minimum RSS.

We can approximate the optimal values of `b0` and `b1` that minimize the RSS through the `optim` function. The optim function takes two main arguments:

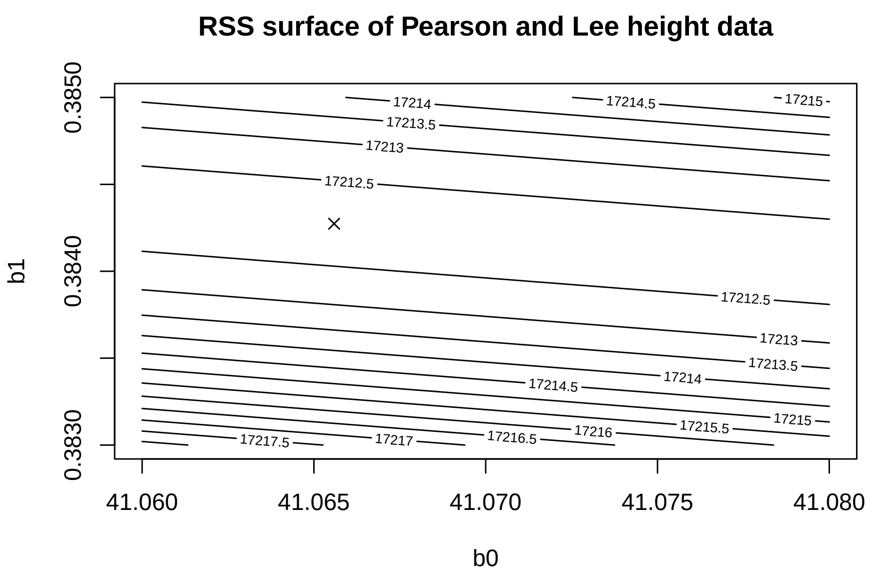
- `par`: a vector of starting values for the optimization algorithm. In our case, this will be the starting values for `b0` and `b1`.
- `fn`: a function of `par` to minimize.

The `optim` function will return a list with several pieces of information (see `?stats::optim`) for details. We want the `par` component of the returned list, which is the `par` vector that (approximately) minimizes `fn`. We then use the `points` function to plot the "optimal" values of `b0` and `b1` that minimize the RSS.

```
# use the optim function to find the values of b0 and b1 that minimize the RSS
# par is the vector of initial values
# fn is the function to minimize
# $par extracts the values found by optim to minimize fn
optimal_b <- optim(par = c(41, 0.4), fn = compute_rss)$par
# print the optimal values of b
optimal_b
```

```
## [1] 41.0655877  0.3842737
```

```r
# plot optimal value as an X on the contour plot
contour(x = b0_seq, y = b1_seq, z = rss_mat, xlab = "b0", ylab = "b1")
title("RSS surface of Pearson and Lee height data")
points(x = optimal_b[1], y = optimal_b[2], pch = 4)
```



What is our takeaway from this example? It's probably not ideal to numerically search for the values of $\hat{\beta}_0$ and $\hat{\beta}_1$ that minimize $RSS(\hat{\beta}_0, \hat{\beta}_1)$. Instead, we should seek an exact solution using mathematics.

6.1.2 OLS estimators of the simple linear regression coefficents

Define $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i$.

The OLS estimators of the regression coefficients for a simple linear regression coefficients are

$$\begin{aligned}
\hat{\beta}_1 &= \frac{\sum_{i=1}^n x_i Y_i - \frac{1}{n}\left(\sum_{i=1}^n x_i\right)\left(\sum_{i=1}^n Y_i\right)}{\sum_{i=1}^n x_i^2 - \frac{1}{n}\left(\sum_{i=1}^n x_i\right)^2} \\
&= \frac{\sum_{i=1}^n (x_i - \bar{x})(Y_i - \bar{Y})}{\sum_{i=1}^n (x_i - \bar{x})^2} \\
&= \frac{\sum_{i=1}^n (x_i - \bar{x})Y_i}{\sum_{i=1}^n (x_i - \bar{x})x_i}
\end{aligned}$$

and

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{x}. \tag{6.1}$$

Thought it's already been said, we state once again that the OLS estimators of β_0 and β_1 shown above are the estimators that minimize the RSS.

The other parameter we've discussed is the error variance, σ^2. The most common estimator of the error variance is

$$\hat{\sigma}^2 = \frac{RSS}{n-p}, \tag{6.2}$$

where p is the number of regression coefficients. In general, $n-p$ is the degrees of freedom of the RSS. In a simple linear regression context, the denominator of (6.2) is $n-2$.

6.2 Penguins simple linear regression example

We will use the `penguins` data set in the **palmerpenguins** package (Horst, Hill, and Gorman 2020) to illustrate a very basic simple linear regression analysis.

The `penguins` data set provides data related to various penguin species measured in the Palmer Archipelago (Antarctica), originally provided by Gorman, Williams, and Fraser (2014). We start by loading the data into memory.

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

The data set includes 344 observations of 8 variables. The variables are:

- `species`: a `factor` indicating the penguin species
- `island`: a `factor` indicating the island the penguin was observed
- `bill_length_mm`: a `numeric` variable indicating the bill length in millimeters
- `bill_depth_mm`: a `numeric` variable indicating the bill depth in millimeters
- `flipper_length_mm`: an `integer` variable indicating the flipper length in millimeters
- `body_mass_g`: an `integer` variable indicating the body mass in grams

- `sex`: a `factor` indicating the penguin sex (`female`, `male`)
- `year`: an integer denoting the study year the penguin was observed (2007, 2008, or 2009)

We begin by creating a scatter plot of `bill_length_mm` versus `body_mass_g` (y-axis versus x-axis) in Figure 6.2. We see a clear positive association between body mass and bill length: as the body mass increases, the bill length tends to increase. The pattern is linear, i.e., roughly a straight line.

```
plot(bill_length_mm ~ body_mass_g, data = penguins,
     ylab = "bill length (mm)", xlab = "body mass (g)",
     main = "Penguin size measurements")
```

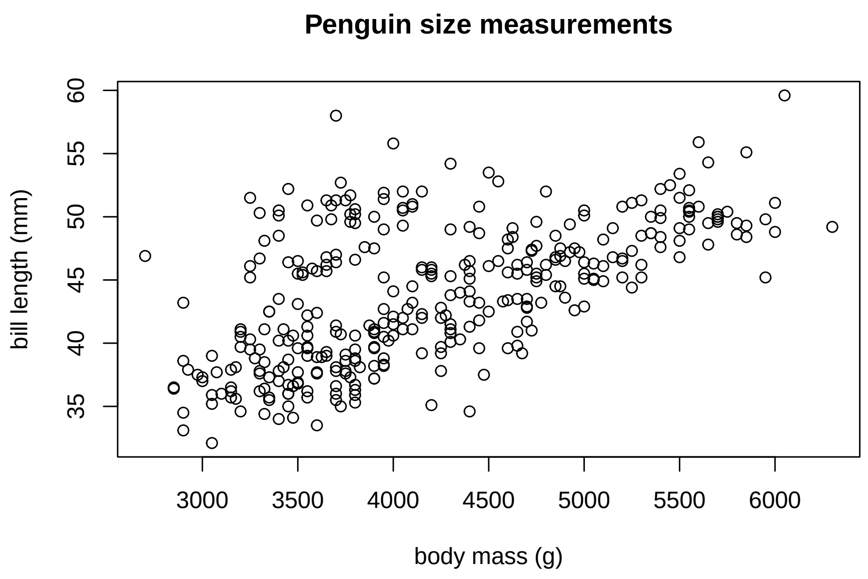


Figure 6.2: A scatter plot of penguin bill length (mm) versus body mass (g)

We first perform a single linear regression analysis manually using the equations previously provided by regressing `bill_length_mm` on `body_mass_g`.

Using the `summary` function on the `penguins` data frame, we see that both `bill_length_mm` and `body_mass_g` have `NA` values.

```
summary(penguins)
```

```
##       species          island    bill_length_mm
##  Adelie   :152   Biscoe   :168   Min.   :32.10
##  Chinstrap: 68   Dream    :124   1st Qu.:39.23
##  Gentoo   :124   Torgersen: 52   Median :44.45
##                                   Mean   :43.92
##                                   3rd Qu.:48.50
##                                   Max.   :59.60
##                                   NA's   :2
##  bill_depth_mm   flipper_length_mm  body_mass_g
##  Min.   :13.10   Min.   :172.0     Min.   :2700
##  1st Qu.:15.60   1st Qu.:190.0     1st Qu.:3550
##  Median :17.30   Median :197.0     Median :4050
##  Mean   :17.15   Mean   :200.9     Mean   :4202
##  3rd Qu.:18.70   3rd Qu.:213.0     3rd Qu.:4750
##  Max.   :21.50   Max.   :231.0     Max.   :6300
##  NA's   :2       NA's   :2         NA's   :2
##      sex           year
##  female:165   Min.   :2007
##  male  :168   1st Qu.:2007
##  NA's  : 11   Median :2008
##               Mean   :2008
##               3rd Qu.:2009
##               Max.   :2009
##
```

We want to remove the rows of `penguins` where either `body_mass_g` or `bill_length_mm` have `NA` values. We do that below using the `na.omit` function (selecting only the relevant variables) and assign the cleaned object the name `penguins_clean`.

```
# remove rows of penguins where bill_length_mm or body_mass_g have NA values
penguins_clean <- na.omit(penguins[,c("bill_length_mm", "body_mass_g")])
```

We extract the `bill_length_mm` variable from the `penguins` data frame and assign it the name `y` since it will be the response variable. We extract the `body_mass_g` variable from the `penguins` data frame and assign it the name `y` since it will be the predictor variable. We also determine the number of observations and assign that value the name `n`.

```
# extract response and predictor from penguins_clean
y <- penguins_clean$bill_length_mm
x <- penguins_clean$body_mass_g
# determine number of observations
n <- length(y)
```

We now compute $\hat{\beta}_1$ and $\hat{\beta}_0$. Note that placing `()` around the assignment operations will both perform the assign and print the results.

```
# compute OLS estimates of beta1 and beta0
(b1 <- (sum(x * y) - sum(x) * sum(y) / n)/(sum(x^2) - sum(x)^2/n))
```

```
## [1] 0.004051417
```

```
(b0 <- mean(y) - b1 * mean(x))
```

```
## [1] 26.89887
```

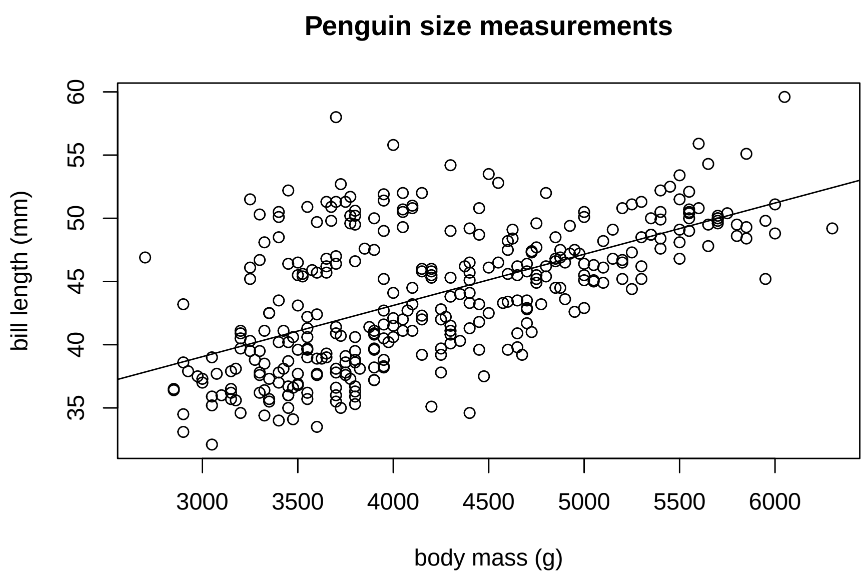
The estimated value of β_0 is $\hat{\beta}_0 = 26.90$ and the estimated value of β_1 is $\hat{\beta}_1 = 0.004$. The basic mathematical interpretation of our results is that:

- ($\hat{\beta}_1$): If a penguin has a body mass 1 gram larger than another penguin, we expect the larger penguins bill length to be 0.004 millimeters longer.
- ($\hat{\beta}_0$):A penguin with a body mass of 0 grams is expected to have a bill length of 26.9 millimeters.

The latter interpretation is clearly non-sensical and is caused by the fact that we are extrapolating far outside the observed body mass values. The relationship between body mass and bill length is different for penguin chicks versus adults.

We can use the `abline` function to overlay the fitted model on the observed data. Note that in simple linear regression, $\hat{\beta}_1$ corresponds to the slope of the fitted line and $\hat{\beta}_0$ will be the intercept.

```
plot(bill_length_mm ~ body_mass_g, data = penguins,
     ylab = "bill length (mm)", xlab = "body mass (g)",
     main = "Penguin size measurements")
# a is the intercept and b is the slope
abline(a = b0, b = b1)
```



The fit of the model to our observed data seems reasonable.

We can also compute the residuals, $\hat{\epsilon}_1, \ldots, \hat{\epsilon}_n$, the fitted values $\hat{y}_1, \ldots, \hat{y}_n$, and the associated RSS, $RSS = \sum_{i=1}^{n} \hat{\epsilon}_i^2$.

```
yhat <- b0 + b1 * x # compute fitted values
ehat <- y - yhat # compute residuals
(rss <- sum(ehat^2)) # sum of the squared residuals
```

```
## [1] 6564.494
```

```
(sigmasqhat <- rss/(n-2)) # estimated error variance
```

```
## [1] 19.30734
```

6.3 Fitting a linear model using R

We now describe how to use R to fit a linear model to data.

The `lm` function uses OLS to fit a linear model to data. The function has two major arguments:

- `data`: the data frame in which the model variables are stored. This can be omitted if the variables are already stored in memory.
- `formula`: a Wilkinson and Rogers (1973) style formula describing the linear regression model. Assuming the `y` is the response, `x`, `x1`, `x2`, `x3` are available numeric predictors:
  - `y ~ x` describes a simple linear regression model based on $E(Y|X) = \beta_0 + \beta_1 X$.
  - `y ~ x1 + x2` describes a multiple linear regression model based on $E(Y|X_1, X_2) = \beta_0 + \beta_1 X_1 + \beta_2 X_2$.
  - `y ~ x1 + x2 + x1:x2` and `y ~ x1 * x2` describe a multiple linear regression model based on $E(Y|X_1, X_2) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2$.
  - `y ~ -1 + x1 + x2` describes a multiple linear regression model without an intercept, in this case, $E(Y|X_1, X_2) = \beta_1 X_1 + \beta_2 X_2$.
  - `y ~ x + I(x^2)` describe a multiple linear regression model based on $E(Y|X) = \beta_0 + \beta_1 X + \beta_2 X^2$.

We fit a linear model regressing `body_mass_g` on `bill_length_mm` using the `penguins` data frame and store it in the object `lmod`. `lmod` is an object of class `lm`.

```
lmod <- lm(bill_length_mm ~ body_mass_g, data = penguins) # fit model
class(lmod) # class of lmod
```

```
## [1] "lm"
```

There are a number of methods (generic function that do something specific when applied to a certain type of object). Commonly used ones include:

- `residuals`: extracts $\hat{\boldsymbol{\epsilon}}$ from an `lm` object.
- `fitted`: extracts $\hat{\mathbf{y}}$ from an `lm` object.

- `coef` or `coefficients`: extracts $\hat{\boldsymbol{\beta}}$ from an `lm` object.
- `deviance`: extracts the RSS from an `lm` object.
- `sigma`: extracts $\hat{\sigma}$ from an `lm` object.
- `df.residual`: extracts $n-p$, the degrees of freedom for the RSS, from an `lm` object.
- `summary`: provides:
  - A 5-number summary of the $\hat{\boldsymbol{\epsilon}}$
  - A table that lists the predictors, the `Estimate` of the associated coefficients, the **estimated** standard error of the estimates (`Std.Error`), the computed test statistic associated with testing $H_0 : \beta_j = 0$ versus $H_a : \beta_j \neq 0$ for $j = 0, 1, \ldots, p-1$ (`t value`), and the associated p-value of each test `Pr(>|t|)`.

We now use some of the methods to extract important characteristics of our fitted model. We then check whether the values obtained from these methods match our manual calculations.

```
(coeffs2 <- coefficients(lmod)) # extract, assign, and print coefficients
```

```
##  (Intercept)  body_mass_g
## 26.898872424  0.004051417
```

```
ehat2 <- residuals(lmod) # extract and assign residuals
yhat2 <- fitted(lmod) # extract and assign fitted values
rss2 <- deviance(lmod) # extract and assign rss
sigmasqhat2 <- rss2/df.residual(lmod) # estimated error variance
# compare to manually computed values
all.equal(c(b0, b1), coeffs2, check.attributes = FALSE)
```

```
## [1] TRUE
```

```
all.equal(ehat, ehat2, check.attributes = FALSE)
```

```
## [1] TRUE
```

```
all.equal(rss, rss2)
```

```
## [1] TRUE
```

```
all.equal(sigmasqhat, sigmasqhat2)
```

```
## [1] TRUE
```

```
# methods(class="lm")
```

6.3.1 Derivation of OLS simple linear regression estimators

Use calculus to derive the OLS estimator of the regression coefficients. Take the partial derivatives of $RSS(\hat{\beta}_0, \hat{\beta}_1)$ with respect to $\hat{\beta}_0$ and $\hat{\beta}_1$, set the derivatives equal to zero, and solve for $\hat{\beta}_0$ and $\hat{\beta}_1$ to find the critical points of $RSS(\hat{\beta}_0, \hat{\beta}_1)$.

Technically, you must show that the Hessian matrix of $RSS(\hat{\beta}_0, \hat{\beta}_1)$ is positive definite to verify that our solution minimizes the RSS, but we won't do that here.

6.3.2 Expected value of OLS simple linear regression estimators

Determine the expected value of the OLS simple linear regression estimators.

Chapter 7

Interpreting a fitted linear model

Placeholder

Chapter 8

Categorical predictors

Placeholder

8.1 Indicator/dummy variables

8.2 Common of linear models with categorical predictors

8.2.1 One-way ANOVA

8.2.1.1 Definition

8.2.1.2 Interpretation

8.2.2 Main effects models

8.2.3 Interaction models

8.2.4 Extensions

Chapter 9

Assessing and addressing collinearity

Gorman, Kristen B., Tony D. Williams, and William R. Fraser. 2014. "Ecological Sexual Dimorphism and Environmental Variability Within a Community of Antarctic Penguins (Genus Pygoscelis)." *PLOS ONE* 9 (3): 1–14. https://doi.org/10.1371/journal.pone.0090081.

Horst, Allison, Alison Hill, and Kristen Gorman. 2020. *Palmerpenguins: Palmer Archipelago (Antarctica) Penguin Data*. https://CRAN.R-project.org/package=palmerpenguins.

Wilkinson, GN, and CE Rogers. 1973. "Symbolic Description of Factorial Models for Analysis of Variance." *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 22 (3): 392–99.