Additional tools

Giovanni Saraceno

Contents


```
library(tidyverse)
library(dplyr)
library(ggplot2)
```

Assumptions in hypothesis testing

The methods for computing the confidence intervals and hypothesis testing via the t-test are correct assuming that the variable under study follows a normal distribution. However, in practice it can happen that the frequency distribution does not follow a normal distribution or that the variances of groups are not equal. Hence, it is fundamental in practice to check that data satisfy this assumption. In general, we can list the following possible approaches.

- If the sample size is sufficiently large and the violations of hypothesis are not extreme, the statistical procedures could be still applied and give reasonable results. Note that this is not suggesting to blindly ignore the violations of assumptions.
- Transformations of variables: applying some function to the data can satisfy the assumptions (logaritmic, arcsin, square root, inverse, ...).
- *Non-parameteric* methods: these methods do not require model assumptions, such as the normality assumtion.

Normality assumption

Consider the `penguin` data set

```
library(palmerpenguins)
data(penguins)
str(penguins)
```

```
## 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 ...
```

Consider that we want to compare the bill length of penguins from the species “Adelie” and “Gentoo”, assuming that they have equal variance.

```
id_a <- which(penguins$species=="Adelie" & !is.na(penguins$bill_length_mm))
mu_a <- mean(penguins$bill_length_mm[id_a], na.rm = TRUE)
sd_a <- sd(penguins$bill_length_mm[id_a], na.rm = TRUE)
id_g <- which(penguins$species=="Gentoo"& !is.na(penguins$bill_length_mm))
mu_g <- mean(penguins$bill_length_mm[id_g], na.rm = TRUE)
sd_g <- sd(penguins$bill_length_mm[id_g], na.rm = TRUE)
print(paste0("Adelie: ",mu_a, " (", sd_a, ")"))
```

```
## [1] "Adelie: 38.7913907284768 (2.66340484836862)"
```

```
print(paste0("Chinstrap: ", mu_g, " (", sd_g, ")"))
```

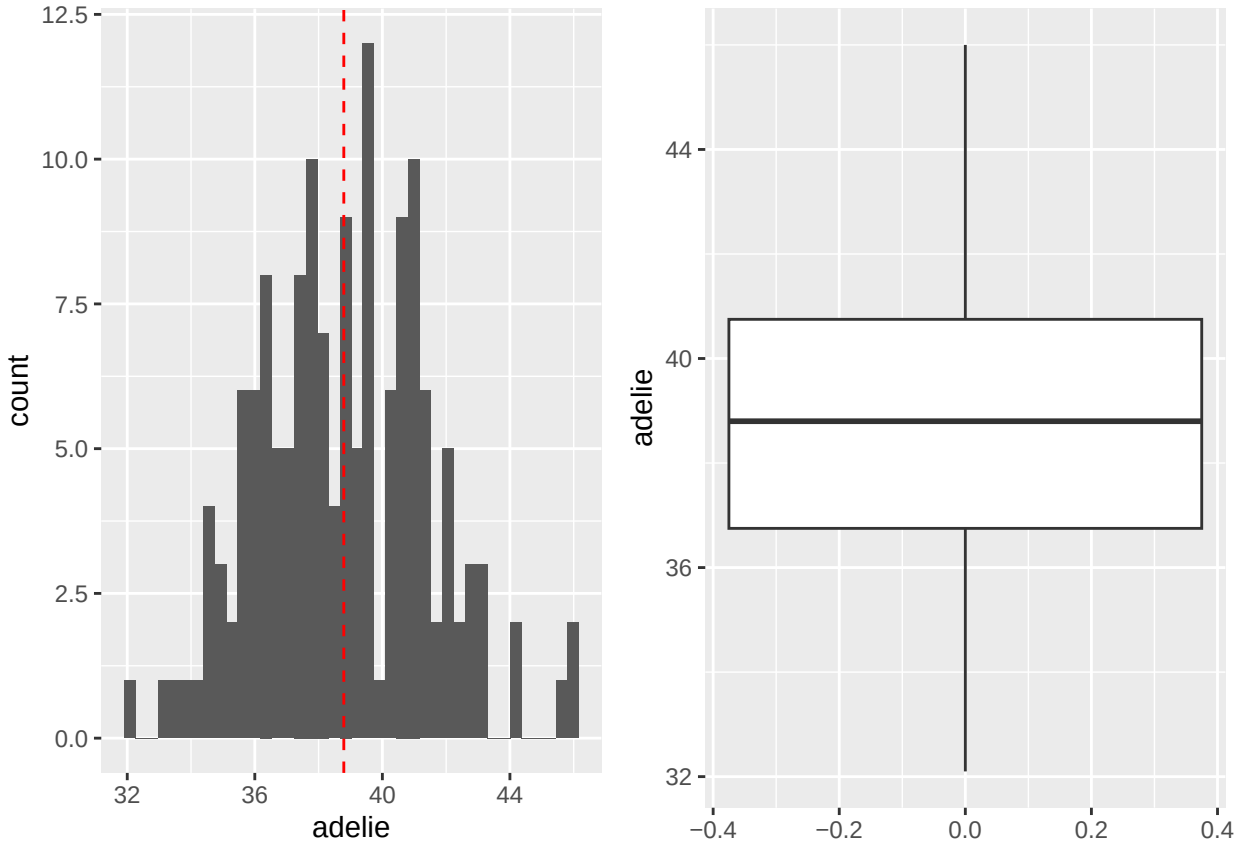
```
## [1] "Chinstrap: 47.5048780487805 (3.08185737211429)"
```

In this case, it is appropriate to perform a t.test considering as null hypothesis $H_0 : \mu_A = \mu_C$ versus $H_0 : \mu_A \neq \mu_C$. This test assumes that the two populations are normally distributed.

```
adelie <- penguins$bill_length_mm[id_a]
gentoo <- penguins$bill_length_mm[id_g]
```

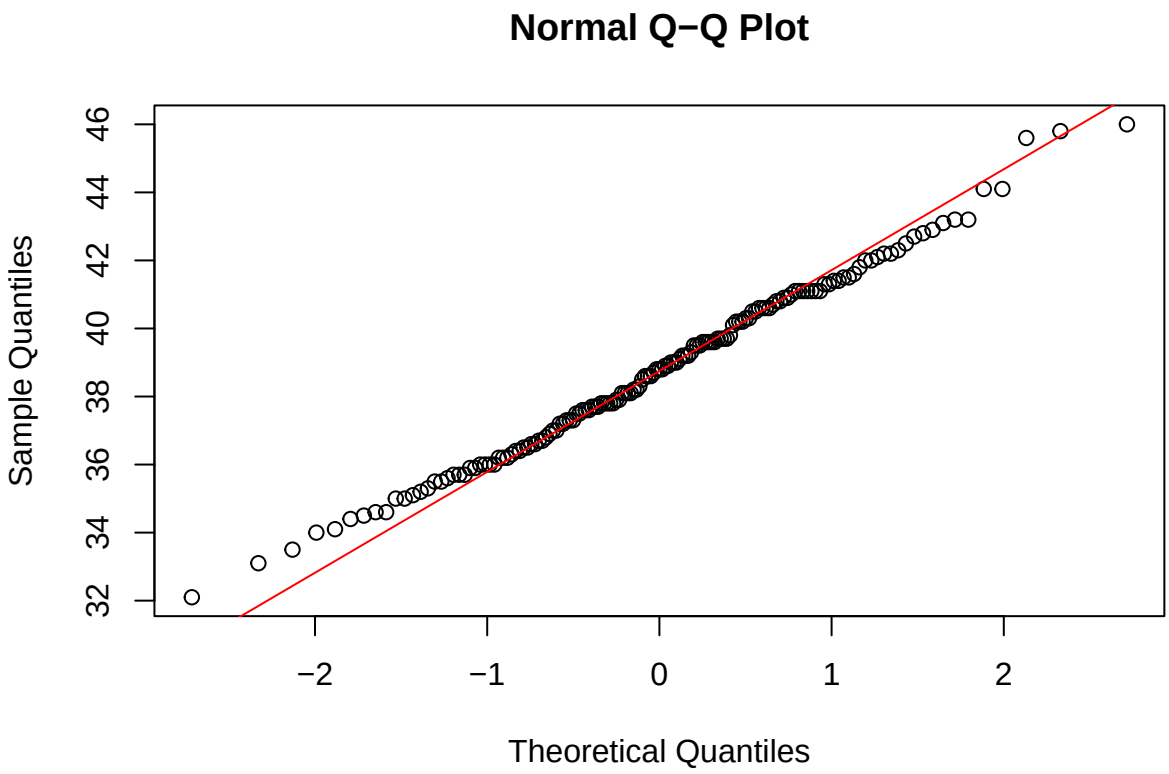
In order to evaluate the normality assumption we can investigate the distribution of data using graphical tools, such as boxplots and histograms, to verify that there are no departures from the normality, such as asymmetry, bimodality or presence of outliers.

```
n_a <- length(adelie)
p1 <- ggplot(mapping= aes(x=adelie))+
  geom_histogram(bins = 40) +
  geom_vline(xintercept = mu_a, color="red", linetype="dashed")
p2 <- ggplot(mapping= aes(y=adelie))+
  geom_boxplot(outliers = TRUE)
gridExtra::grid.arrange(p1, p2, nrow=1)
```

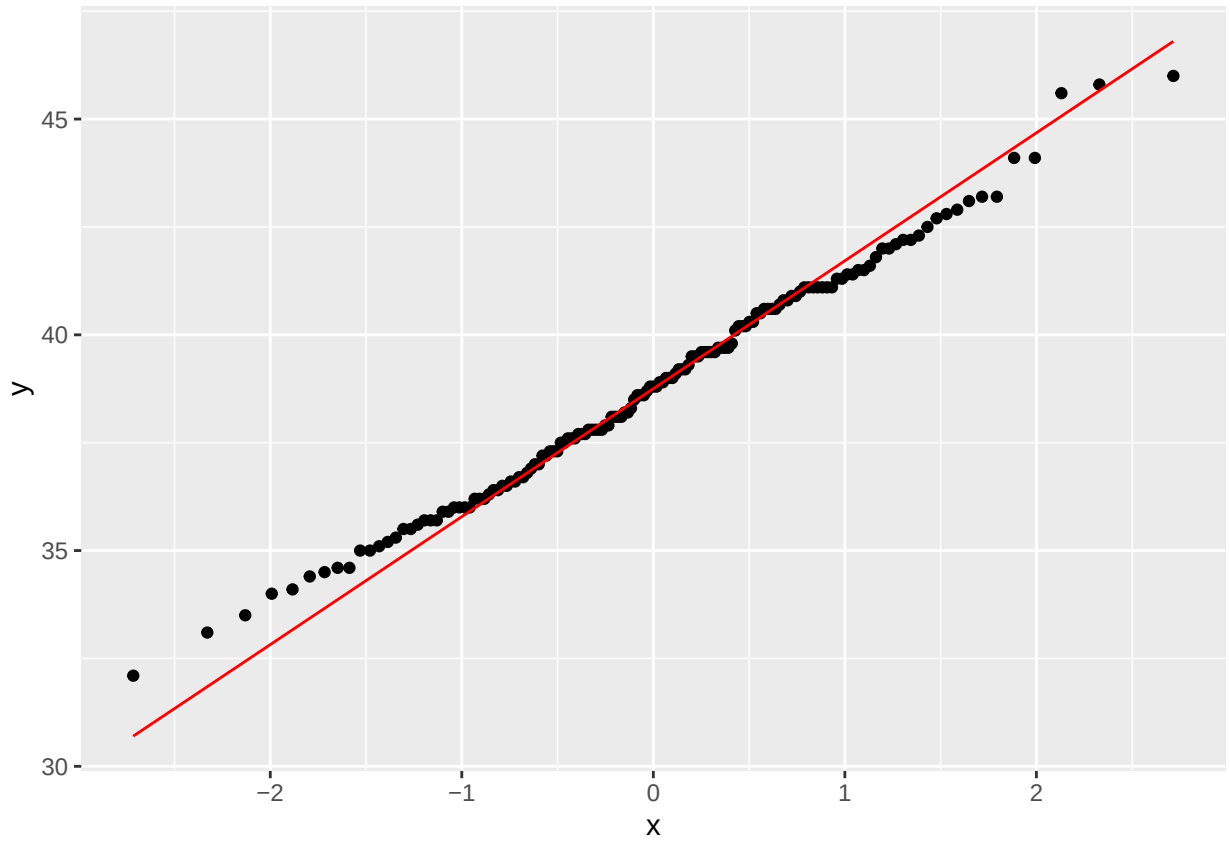


The distribution of `adelie` seems symmetric around the mean and the median, and unimodal. From the boxplot no outliers are clearly visible. Additionally, we can compare the sample quantiles with the quantiles of the normal distribution.

```
qqnorm(adelie, plot.it = TRUE)
qqline(adelie, col="red")
```




```
ggplot(mapping = aes(sample=adelie)) +
  geom_qq()+
  geom_qq_line(color="red")
```



We have added the line of theoretical quantile-quantile plot. The data is approximately normal, as the points largely follow the qqline. There are minor deviations at the tails and there are no extreme points (outliers) far away from the line. Finally, we can also perform a statistical test with null hypothesus H_0: the sample follows a normal distribution. Testing is data points followsì a specific distribution is referred to as *Goodness-of-fit* testing. One of the most used method for testing normality is the *Shapiro-Wilk* test.

```
shapiro.test(adelie)
```

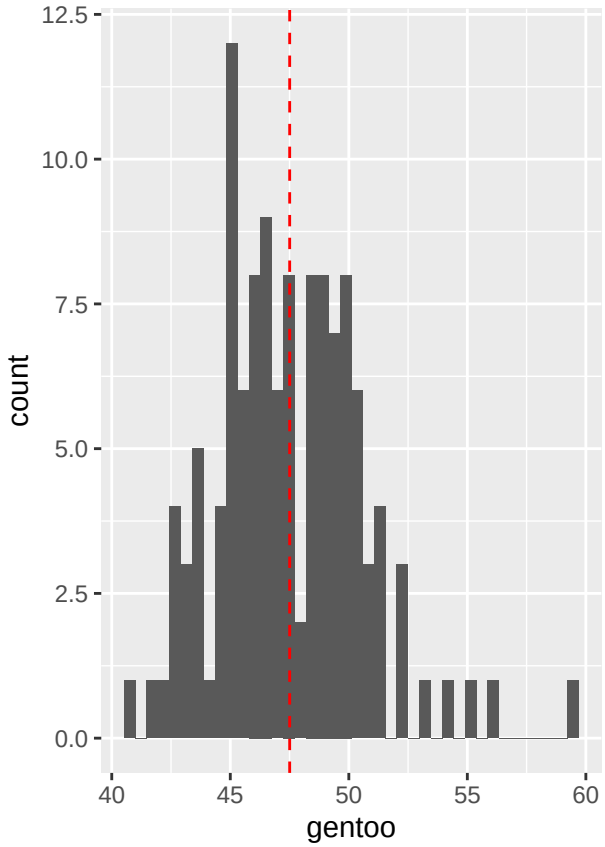
```
##
##  Shapiro-Wilk normality test
##
## data:  adelie
## W = 0.99336, p-value = 0.7166
```

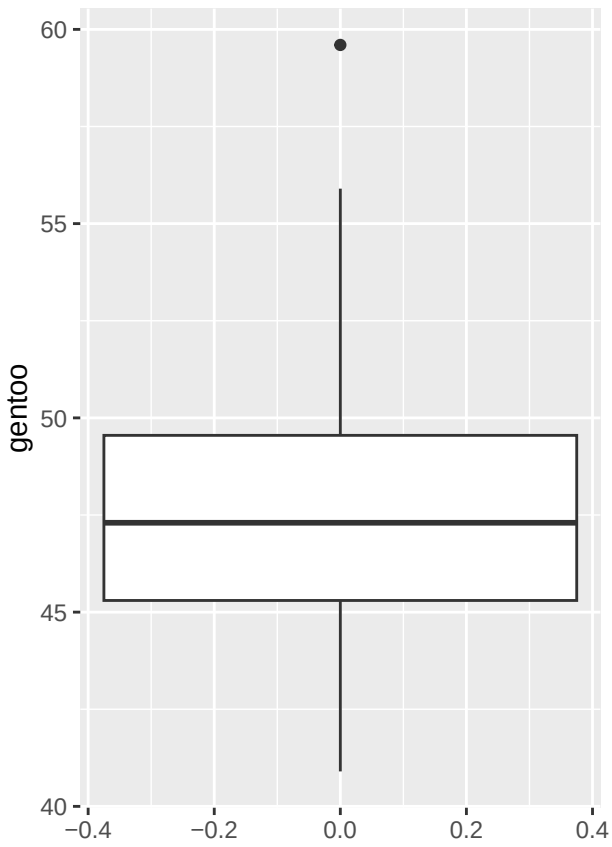
The obtained p-value corroborates the previous conclusions.

> Test for normality shoul be used with caution. A low sample size could not provide enough evidence for rejecting the null hypothesis, even if data are not normally distributed. On the other side, a very large sample size can reject the null hypothesis even in the cases where the deviation from the normal distribution is not so large.

We can do the same for the `gentoo` sample.

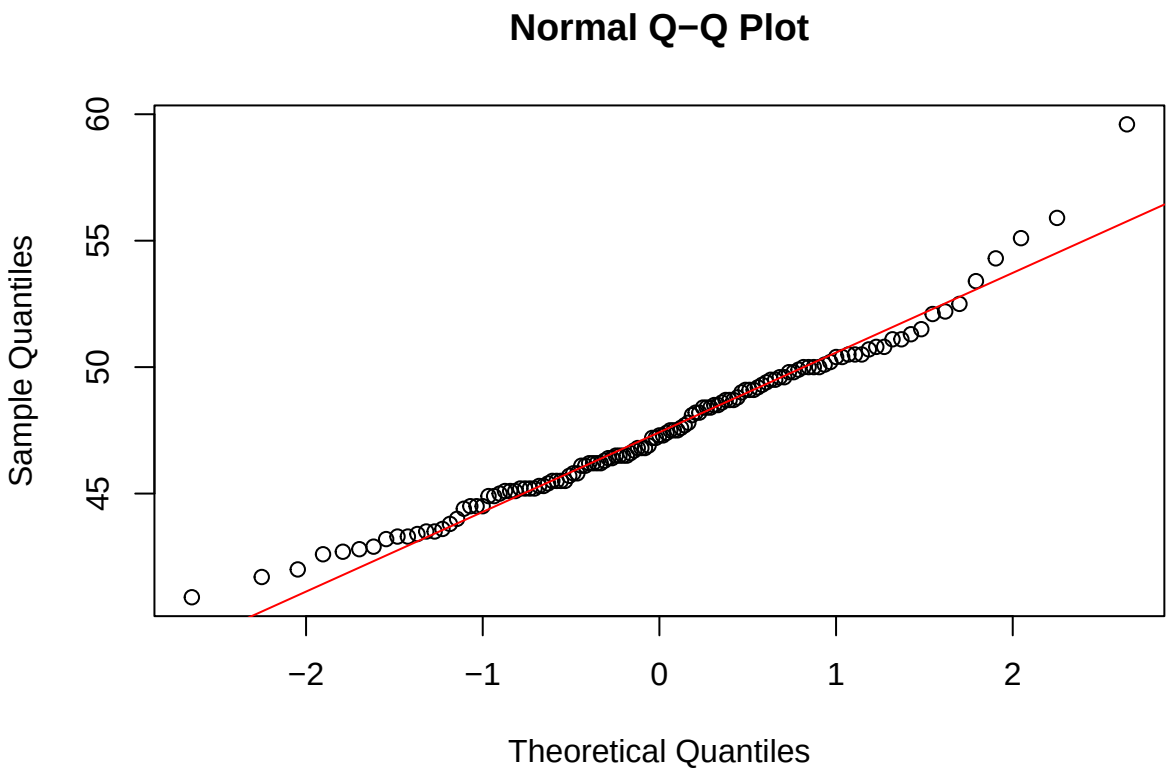
```
p1 <- ggplot(mapping= aes(x=gentoo))+
  geom_histogram(bins = 40) +
  geom_vline(xintercept = mu_g, color="red", linetype="dashed")
p2 <- ggplot(mapping= aes(y=gentoo))+
  geom_boxplot(outliers = TRUE)
gridExtra::grid.arrange(p1, p2, nrow=1)
```



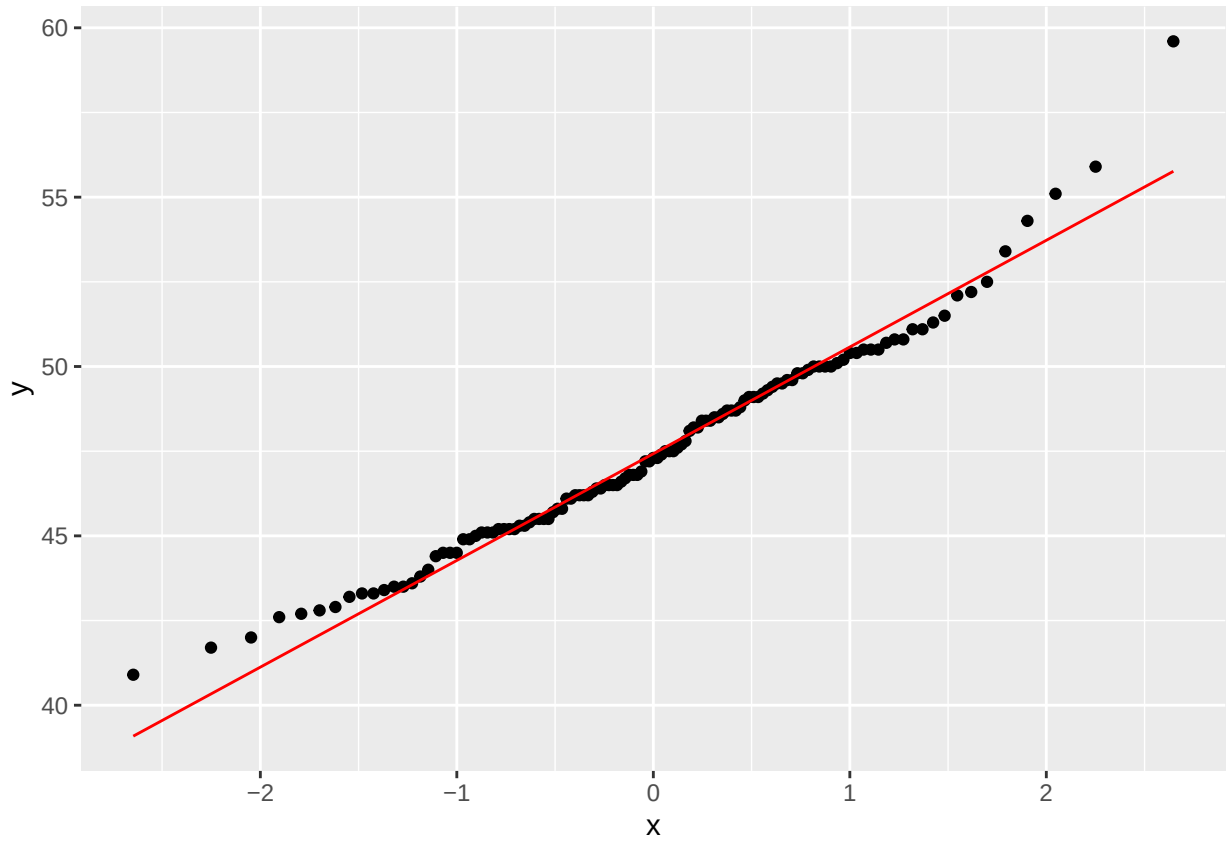


The distribution of `gentoo` seems symmetric around the mean and the median, and unimodal. From the boxplot one outlier is displayed on the right tail of the distribution. The havier right tail is also suggested by the histogram. Additionally, we compare the sample quantiles with the quantiles of the normal distribution.

```
qqnorm(gentoo, plot.it = TRUE)
qqline(gentoo, col="red")
```




```
ggplot(mapping = aes(sample=gentoo)) +
  geom_qq()+
  geom_qq_line(color="red")
```



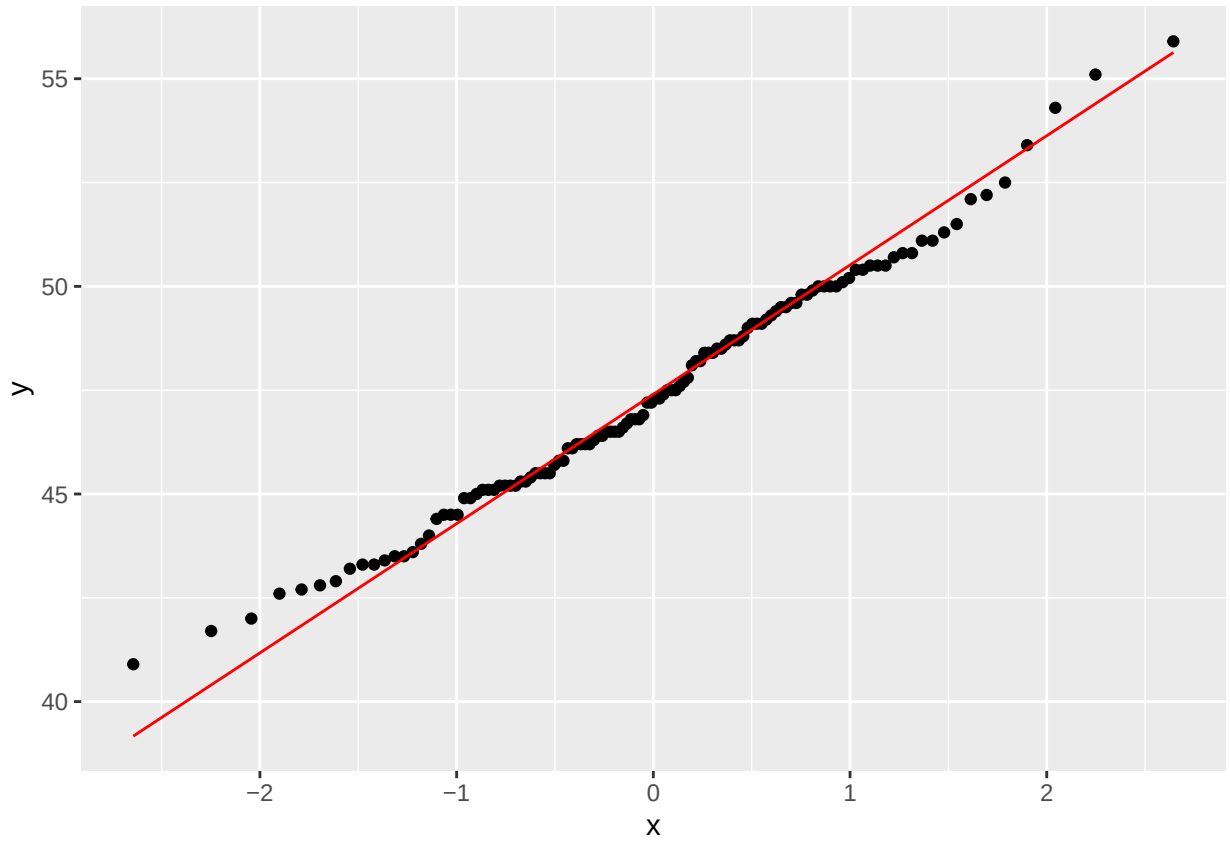
The deviations in the tails suggest that the data has heavier tails. The extreme deviations on both ends could be indicative of outliers or values that are unusually large/small relative to what is expected under normality. Finally, we can also perform the Shapiro-Wilk test.

```
shapiro.test(gentoo)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  gentoo
## W = 0.97272, p-value = 0.01349
```

Considering the 0.05 confidence level used for computing the test, the obtained p-value corroborates the previous conclusions, suggesting to reject the null hypothesis. Notice that, if we remove the outlying oservation, the sample is more likely following a normal distribution.

```
gentoo0 <- gentoo[-which.max(gentoo)]
ggplot(mapping = aes(sample=gentoo0)) +
  geom_qq()+
  geom_qq_line(color="red")
```




```
shapiro.test(gentoo0)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  gentoo0
## W = 0.98862, p-value = 0.4068
```

Finally, we apply the t.test for comparing their mean

```
t.test(adelie, gentoo, var.equal = TRUE, paired = FALSE)
```

```
##
##  Two Sample t-test
##
## data:  adelie and gentoo
## t = -25.095, df = 272, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -9.397060 -8.029915
## sample estimates:
## mean of x mean of y
##  38.79139  47.50488
```

Here, the difference between is clearly different from zero and this difference is statistically significant. In this case, the presence of the outlier does not affect the result.

Transformations of data

One possible technique is to transform data points to improve the goodness of fit to the assumed distribution. Notice that it is not a good idea to try several transformations for findng the one that has the best fit with the model assumptions.

Non-parametric approach

In case of violations of the model assumption, we can use non-parametric tests, which have less assumptions on the distributions of the variables. For example the *permutation-based test* (with the function `independence_test` from the `coin` package) and the *sign test* are alternatives to the t.test for the mean. When we want to compare two means and the normality assumptions are not satisfied, we can use the *Mann-Whitney U test* which only assumes independent samples.

```
wilcox.test(adelie, gentoo)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  adelie and gentoo
## W = 224.5, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```

Robust statistics

We have seen that assumptions are often violated due to the presence of non-normality or outliers and then tests canbecome unreliable. To address these limitations, robust alternatives have been developed in order to provide more accurate p-values and confidence intervals while being less sensitive to violations of model assumptions. Notice that non-parametric tests, even if not explicitly robust, offer alternatives that avoid the need for normality assumptions.

For comparing the means of two groups we can use the *Yuen's trimmed-mean test* (trims a percentage of extreme values from both tails before performing a t-test)

```
library(WRS2)
```

```
## Warning: il pacchetto 'WRS2' è stato creato con R versione 4.3.3
```

```
dat <- data.frame(bill_length = c(adelie,gentoo), species =rep(c(1,0),times=c(151,123)))
yuen(bill_length ~ species,data =  dat, tr = 0.2) # tr = trimming percentage
```

```
## Call:
## yuen(formula = bill_length ~ species, data = dat, tr = 0.2)
##
## Test statistic: 22.6846 (df = 154.36), p-value = 0
##
## Trimmed mean difference:  8.60983
## 95 percent confidence interval:
## 7.8601     9.3596
##
## Explanatory measure of effect size: 0.96
```

or the *permutation test*.

```
coin::independence_test(bill_length ~ species, dat)
```

```
##
##  Asymptotic General Independence Test
##
```

```
## data:  bill_length by species
## Z = -13.808, p-value < 2.2e-16
## alternative hypothesis: two.sided
```

How to choose the appropriate test?

The tests used along these notes are not intended to be the suggested ones in practice. Mostly are the firstly introduced methods in statistical hypothesis testing and are considered here to show the reasoning behind the translation of the research question to a null hypothesis and how the confidence intervals and statistical tests can help in making some statistically significant inference on the data. In order to choose the appropriate test to use in our analysis, it is important to consider the following steps.

- Use the graphical methods introduced in the Exploratory analysis section in order to *know* your data.

Type of data	**graphical method**
Categorical	barplot
Numerical	Histogram
	boxplot
	density
Two Categorical	Grouped Barplot
	Contingency table
One Numerical and one categorical	Grouped Histograms
	Grouped Boxplots
Two Numerical	Scatter plot
	Line plot

- Are you studying a single variable or the association of two or more variables?
- The considered variables are Categorical or Numerical?
- What is the research question that we are investigating? It is crucial to define the correct null hypothesis for understanding the appropriate tests.
- What are the assumptions of the available tests? The assumptions of the method must be satisfied by our data to obtain reliable results.
- Search the literature for the most innovative procedures that show the best fit with the complexities in the data in hand. Do not limit yourself to the use of standard and well-known methods. In the statistical literature, new methods are developed to address specific situations, to be robust against model mispecifications or overcome sources of errors in specific applications. Know the performance in terms of level (significance level α) and power of the test in different situations.

Analysis of Variance (ANOVA)

In this section, we consider the case where we have two random variables, X and Y, where one of them, say X is qualitative. In this case, we want to compare the conditional distributions, or in other words we want to verify if observations (Y) in different groups (levels of X) are different on average. The Analysis of Variance, or ANOVA, is the most famous tool for evaluating simultaneously the equality of means of k groups. In general, we say that Y is independent on average from X when X does not influence the mean of Y. Hence, we test

$$\begin{cases} H_0 & : \mu_1 = \ldots = \mu_k \\ H_1 & : \text{at least one } \mu_i \text{ is different from the others} \end{cases}$$

Let us consider the following example.
To determine whether and to what extent the type of meat used to prepare hot dogs influences their calorie content, the calories of 54 packages from different brands were measured. These data are stored in the data frame hot_dog:

```
load("hotdog.Rdata")
colnames(hot_dog) <- c("meat", "calories")
summary(hot_dog)
```

```
##       meat       calories
##  Bovina :20   Min.   : 86.0
##  Mista  :17   1st Qu.:132.0
##  Pollame:17   Median :145.0
##               Mean   :145.4
##               3rd Qu.:172.8
##               Max.   :195.0
```

First, we compute the main descriptive statistics and represent the distributions, based on the type of meat, using boxplots. In this dataset, we have a single numeric vector, calories, whose elements are grouped according to the levels of the factor variable meat. We can use the by function, specifying the numeric vector, the grouping factor, and the operation to apply. For example

```
attach(hot_dog)
group_sizes <- by(calories, meat, length)
n <- sum(group_sizes)
sigma2 <- function(x){
  v <- mean((x - mean(x))^2)
  return(v)
}
means <- by(calories, meat, mean)
variances <- by(calories, meat, sigma2)
print(cbind(means, variances))
```

```
##            means variances
## Bovina  156.8500  487.0275
## Mista   158.7059  599.3841
## Pollame 118.7647  478.6505
```

The overall mean and variance are

```
mu <- mean(calories)
v <- sigma2(calories)
print(c(mu, v))
```

```
## [1] 145.4444 847.3951
```

The total number of groups k is

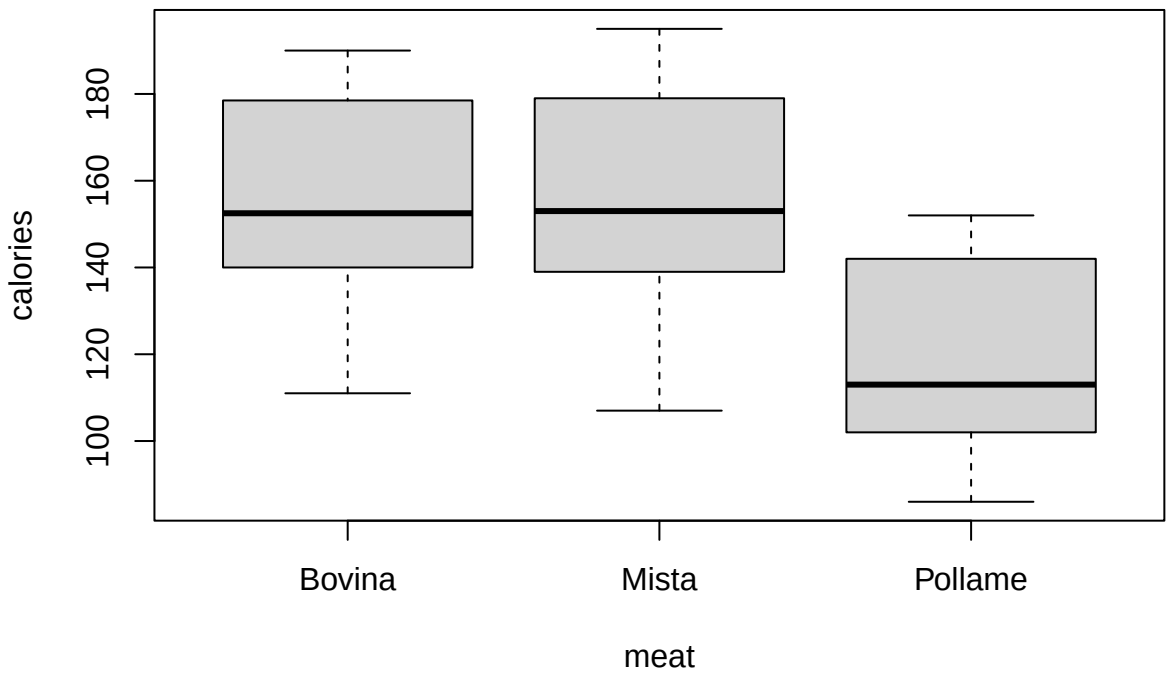
```
k <- length(levels(meat))
k
```

```
## [1] 3
```

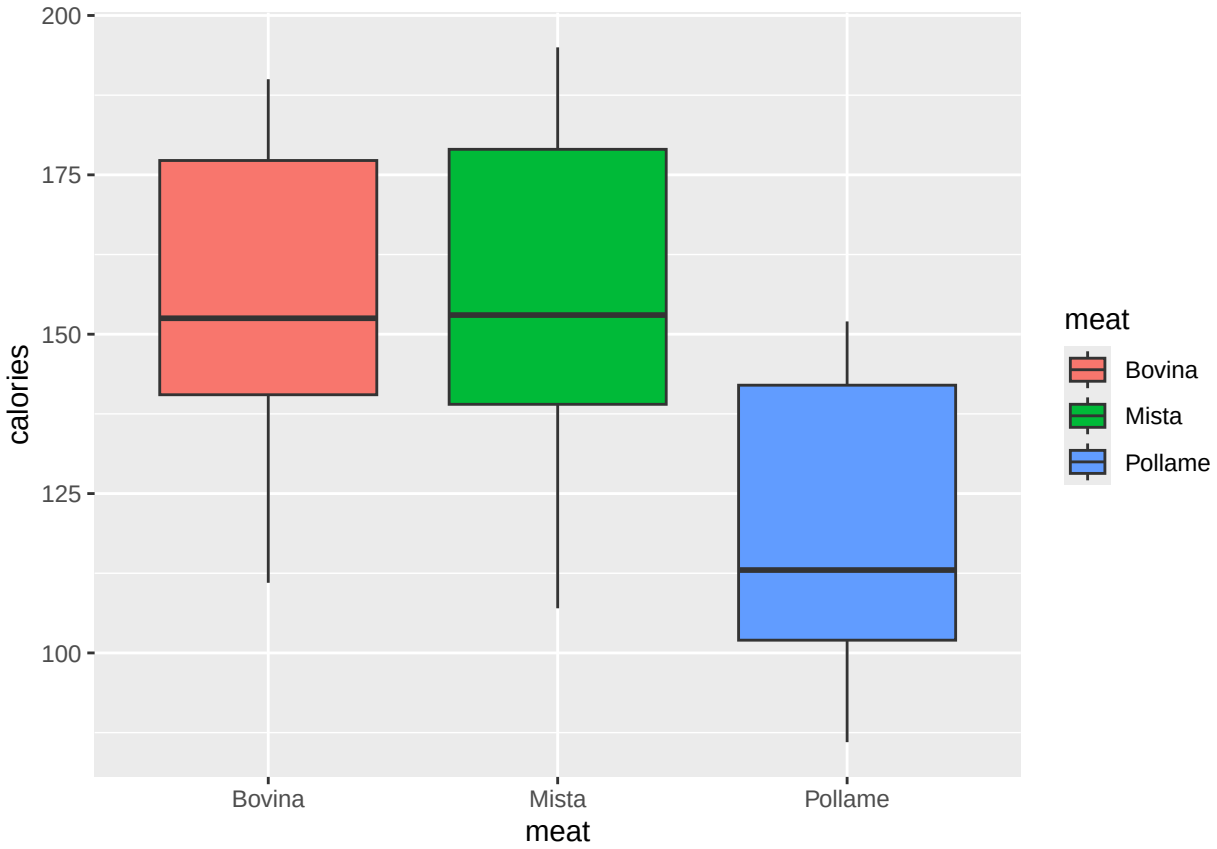
```
detach(hot_dog)
```

The boxplot for calorie distributions by meat type can be plotted using

```r
boxplot(calories ~ meat, hot_dog)
```




```r
ggplot(hot_dog, aes(x=meat,y=calories,fill=meat))+
  geom_boxplot()
```



Note that the custom function `sigma2` computes the variance as the mean of squared deviations (dividing by n, not $n-1$ as in the sample variance). The values and the plot suggest a clear difference for the third group (Poultry). To test if this difference is significant, we conduct a hypothesis test with $H_0 : \mu_1 = \mu_2 = \mu_3$.

We compute the sum of squared deviations from the group means (SS error) and the sum of squared deviations of group means from the overall mean (SS groups)

$$SS_{\text{err}} = \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{ij} - \mu_i)^2, \quad SS_{\text{groups}} = \frac{1}{n}\sum_{i=1}^{k} n_i(\mu_i - \mu)^2$$

```
sse <- sum(group_sizes * variances)
ssg <- sum(group_sizes * (means - mu)^2)
print(c(ssg, sse))
```

```
## [1] 17692.20 28067.14
```

Variance Decomposition These two quantities are also known as within-group variance (SS error) and between-group variance (SS groups). Note that

$$\frac{SS_{\text{err}}}{n} + \frac{SS_{\text{groups}}}{n} = S^2$$

where S^2 is the total variance of all observations with respect to the overall mean.

```
print(c(sse/n + ssg/n, v))
```

```
## [1] 847.3951 847.3951
```

This result shows that dividing the data into groups explains part of the variability. Specifically, the total variance can be decomposed into a portion that describes differences between group means and a portion describing differences within groups.

Pearson Correlation Ratio Using these quantities, we can calculate the Pearson correlation ratio

```
eta2 <- (ssg / n) / v
eta2
```

```
## [1] 0.3866358
```

In general, it ranges between 0 and 1, where 0 indicates independence (the mean does not depend on grouping), and 1 indicates perfect dependence.

Normalizing the sums, we compute

$$MS_{\text{err}} = \frac{SS_{\text{err}}}{n-k}, \quad MS_{\text{groups}} = \frac{SS_{\text{groups}}}{k-1}.$$

Then, the test statistic is

$$F = \frac{MS_{\text{groups}}}{MS_{\text{err}}}.$$

Under H_0 this follows an F-distribution with $k-1$ and $n-k$ degrees of freedom. We reject H_0 if the observed F-value exceeds the $(1-\alpha)$-quantile of the F-distribution for $\alpha = 0.05$.

```
alpha <- 0.05
msg <- ssg / (k - 1)
mse <- sse / (n - k)
print(c(msg, mse))
```

```
## [1] 8846.098  550.336
```

```
f_oss <- msg / mse
pf(f_oss, k - 1, n - k, lower.tail = FALSE)
```

```
## [1] 3.862072e-06
```

In R, ANOVA can be performed directly using the `aov` function:

```
model <- aov(calories ~ meat, hot_dog)
model
```

```
## Call:
##    aov(formula = calories ~ meat, data = hot_dog)
##
## Terms:
##                       meat Residuals
## Sum of Squares  17692.19  28067.14
## Deg. of Freedom        2        51
##
## Residual standard error: 23.45924
## Estimated effects may be unbalanced
```

```
summary(model)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## meat         2  17692    8846   16.07 3.86e-06 ***
## Residuals   51  28067     550
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

For the F-test results, use

```
anova(model)
```

```
## Analysis of Variance Table
##
## Response: calories
##           Df Sum Sq Mean Sq F value    Pr(>F)
## meat       2  17692  8846.1  16.074 3.862e-06 ***
## Residuals 51  28067   550.3
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In practice, the `aov` function is a special case of the `lm` function, which can be used in the same way.

```
attach(hot_dog)
modello2 <- lm(calories ~ meat, hot_dog)
anova(modello2)
```

```
## Analysis of Variance Table
##
## Response: calories
##           Df Sum Sq Mean Sq F value    Pr(>F)
## meat       2  17692  8846.1  16.074 3.862e-06 ***
## Residuals 51  28067   550.3
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Using the `anova` function, we obtain the same results, but the `modello2` object contains more information. If the test indicates that there is a difference between groups, we are naturally interested in identifying where this difference lies. For this purpose, pairwise comparisons between the groups are useful. These can be extracted using the `summary` function

```
summary(modello2)
```

```
##
## Call:
## lm(formula = calories ~ meat, data = hot_dog)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -51.706 -18.492  -5.278  22.500  36.294
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  156.850      5.246  29.901  < 2e-16 ***
## meatMista      1.856      7.739   0.240    0.811
## meatPollame  -38.085      7.739  -4.921 9.39e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.46 on 51 degrees of freedom
## Multiple R-squared:  0.3866, Adjusted R-squared:  0.3626
## F-statistic: 16.07 on 2 and 51 DF,  p-value: 3.862e-06
```

The aspect of multiple comparisons is not treated here.

A non parametric alternative is the *Kruskal-Wallis test* with the assumption that the distribution of the

variable must be the same in each population.

```
kruskal.test(calories~meat, hot_dog)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  calories by meat
## Kruskal-Wallis chi-squared = 19.251, df = 2, p-value = 6.601e-05
```

Hypothesis Verification

The ANOVA test assumes that the variable is normally distributed in each of the k populations and the variance is the same in all the populations. So it is important to verify whether this assumption holds. Since we do not know the true variances of each group, we can perform a test to determine (with a given confidence level) whether the assumption is satisfied. For this, Bartlett's test can be used, where the null hypothesis corresponds to the equality of variances

```
bartlett.test(calories~meat)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  calories by meat
## Bartlett's K-squared = 0.26732, df = 2, p-value = 0.8749
```

```
detach(hot_dog)
```

Note: It is crucial to verify that the assumptions hold for the given study. Otherwise, the test results are meaningless!

Correlation and Regression Analysis

In this section, we consider situations where we want to describe the relationship between two continuous variables.

Let X and Y be two continuous random variables with samples $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$, respectively. To assess the relationship between these two variables, we can start with a scatter plot, which displays the points (x_i, y_i) on a Cartesian plane. Consider the following example:

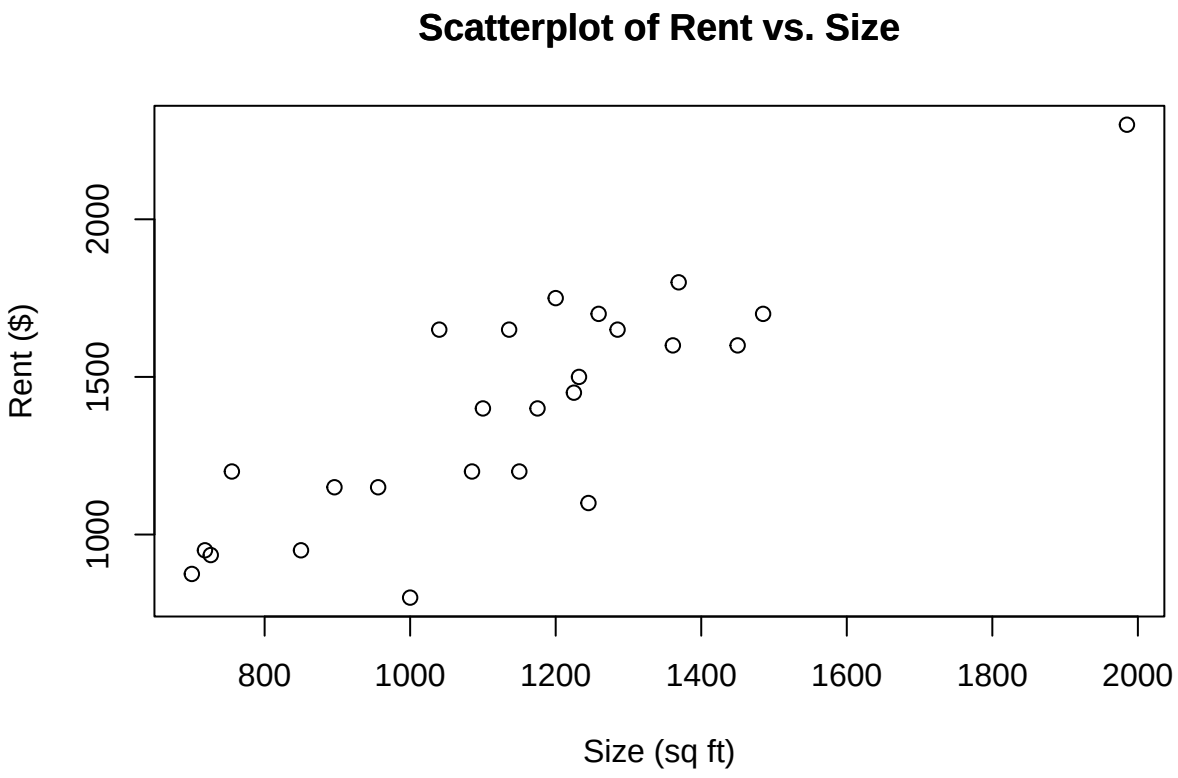
A real estate agent wants to predict the monthly rent of apartments based on their size. For this purpose, they conduct a survey and collect data on 25 apartments in a residential area (rent in dollars, size in square feet).

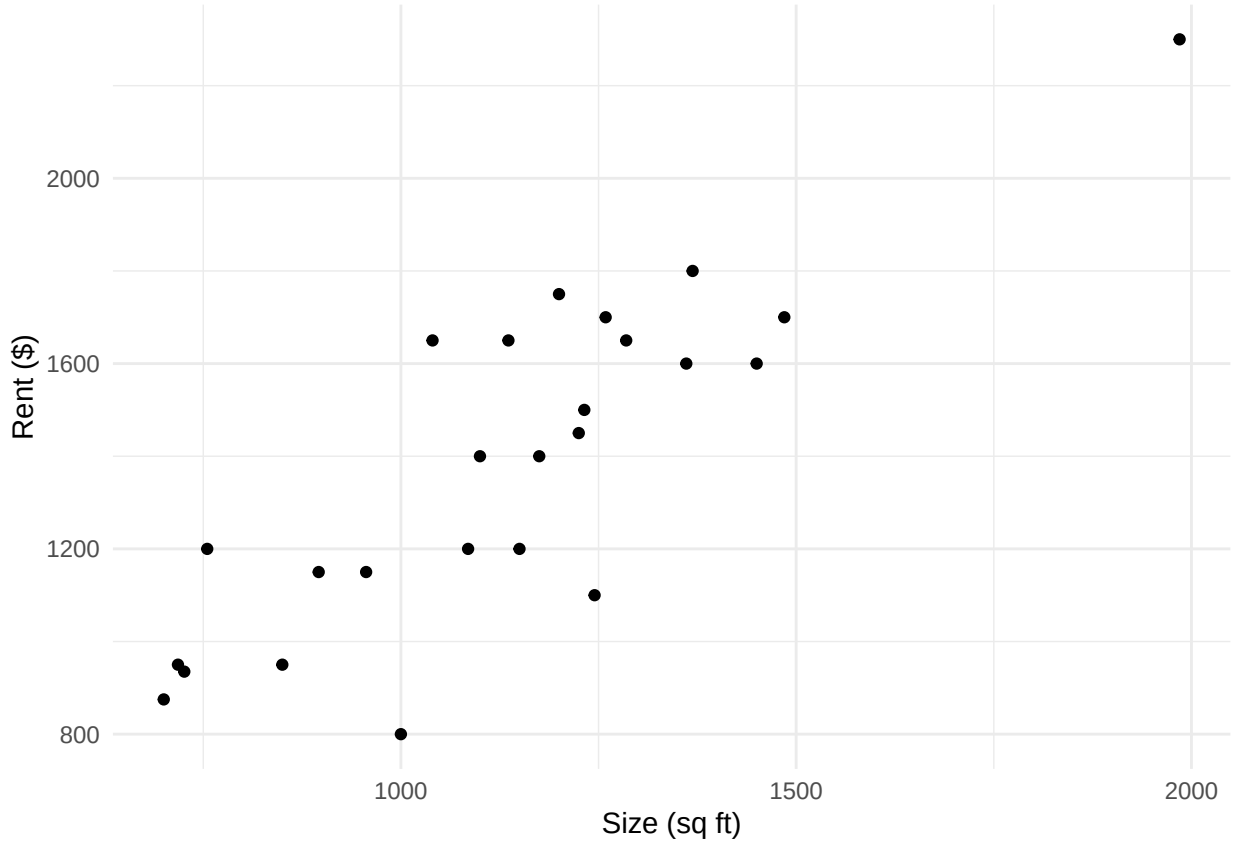
```
data <- data.frame(
  rent = c(950, 1600, 1200, 1500, 950, 1700, 1650, 935, 875, 1150, 1400, 1650,
           2300, 1800, 1400, 1450, 1100, 1700, 1200, 1150, 1600, 1650, 1200,
           800, 1750),
  size = c(850, 1450, 1085, 1232, 718, 1485, 1136, 726, 700, 956, 1100, 1285,
           1985, 1369, 1175, 1225, 1245, 1259, 1150, 896, 1361, 1040, 755, 1000,
           1200)
)

# Scatterplot
plot(data$size, data$rent, main = "Scatterplot of Rent vs. Size",
     xlab = "Size (sq ft)", ylab = "Rent ($)")

ggplot(data, aes(x=size, y=rent)) +
```

```
  geom_point() +
  xlab("Size (sq ft)") +
  ylab("Rent ($)") +
  title("Scatterplot of Rent vs. Size") +
  theme_minimal()
```





The scatterplot visually inspects whether the points exhibit any pattern or regularity.
Remark: The order of the variables in the scatter plot, i.e., which one is X and which one is Y, is crucial in regression analysis. This choice depends on what we want to study or verify. In the example, the agent wants to predict rent based on size, so `size` will be the *independent variable* X, and `rent` will be the *dependent variable* Y.

The relationship between the two variables can be quantified using the *correlation coefficient* ρ. In general, the correlation coefficient is a measure of association between variables that ranges from -1 to 1. Values close to -1 or 1 indicate a strong linear relationship, while values near 0 indicate no linear relationship.

The Pearson correlation coefficient ρ can be calculated in R using the cor function

```
cor(data$size, data$rent)
```

```
## [1] 0.8500608
```

Other types of correlation, such as Spearman or Kendall, can also be computed:

```
cor(data$size, data$rent, method = "spearman")
```

```
## [1] 0.7865843
```

```
cor(data$size, data$rent, method = "kendall")
```

```
## [1] 0.6282929
```

We can set the hypothesis test with $H_0 : \rho = 0$ vs $H_1 : \rho \neq 0$ to verify that the correlation is significant. We can use the `cor.test` function

```
cor.test(data$size, data$rent)
```

```
##
##  Pearson's product-moment correlation
##
## data:  data$size and data$rent
## t = 7.7404, df = 23, p-value = 7.518e-08
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6850170 0.9321097
## sample estimates:
##       cor
## 0.8500608
```

If the correlation is significantly different from zero, we can consider the possibility of a linear relationship between the variables.

Remark: some of the methods for estimating and testing the correlation coefficient are based on the assumptions that the measures follow a *bivariate normal distribution*, the distributions of X and Y separately are normal, and the points in the scatter plot show an elliptical shape.

Simple Linear Regression

The Linear Regression model is commonly used to predict the value of a numeric variable with respect to the value of another variable and it is given as

$$Y = a + bX,$$

where a is called *intercept* and b is the *slope*. Note that the intercept correspond to the value of Y when $X = 0$ and the slope indicates the relative variation of Y for each unit of X. In R, the `lm` function is used to fit a linear regression model using the *least squares method*. The syntax is `y ~ x`, where y is the dependent variable and x is the independent variable.

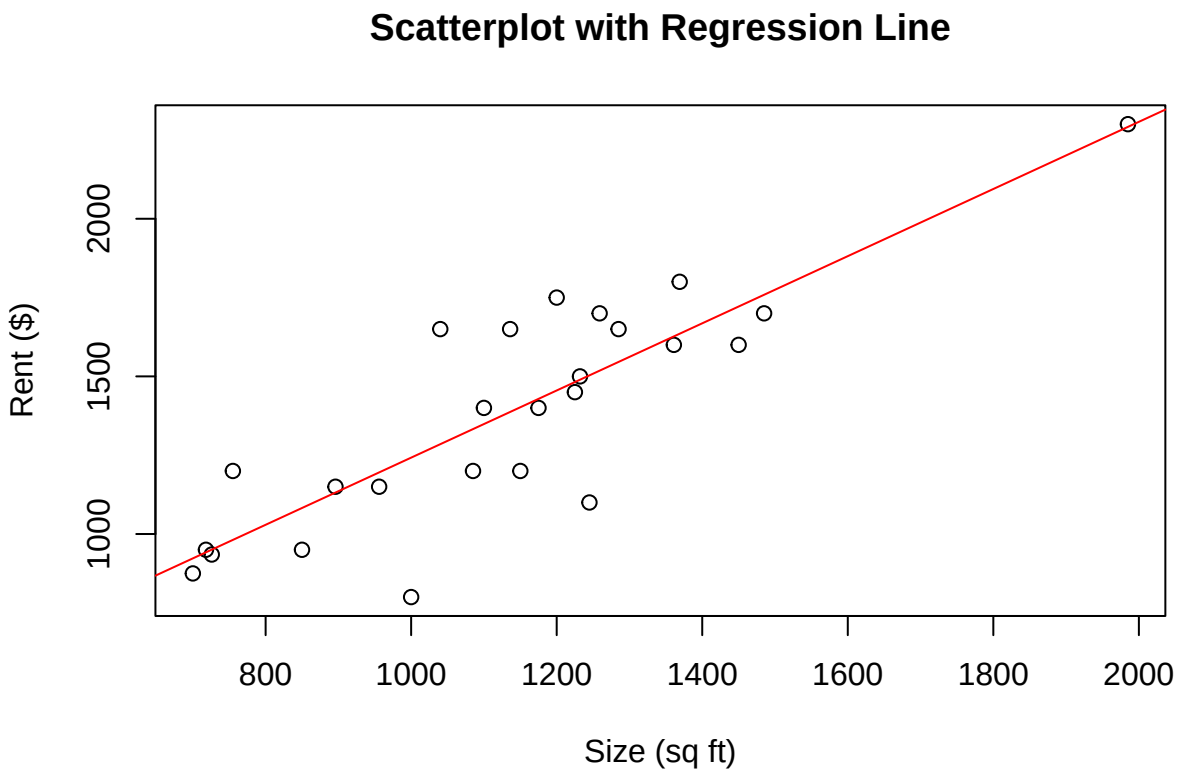
```
model <- lm(rent ~ size, data = data)
summary(model)
```

```
##
## Call:
## lm(formula = rent ~ size, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -442.26  -58.86  -15.42  104.17  365.13
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 177.1208   161.0043    1.10    0.283
## size          1.0651     0.1376    7.74 7.52e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 194.6 on 23 degrees of freedom
## Multiple R-squared:  0.7226, Adjusted R-squared:  0.7105
## F-statistic: 59.91 on 1 and 23 DF,  p-value: 7.518e-08
```

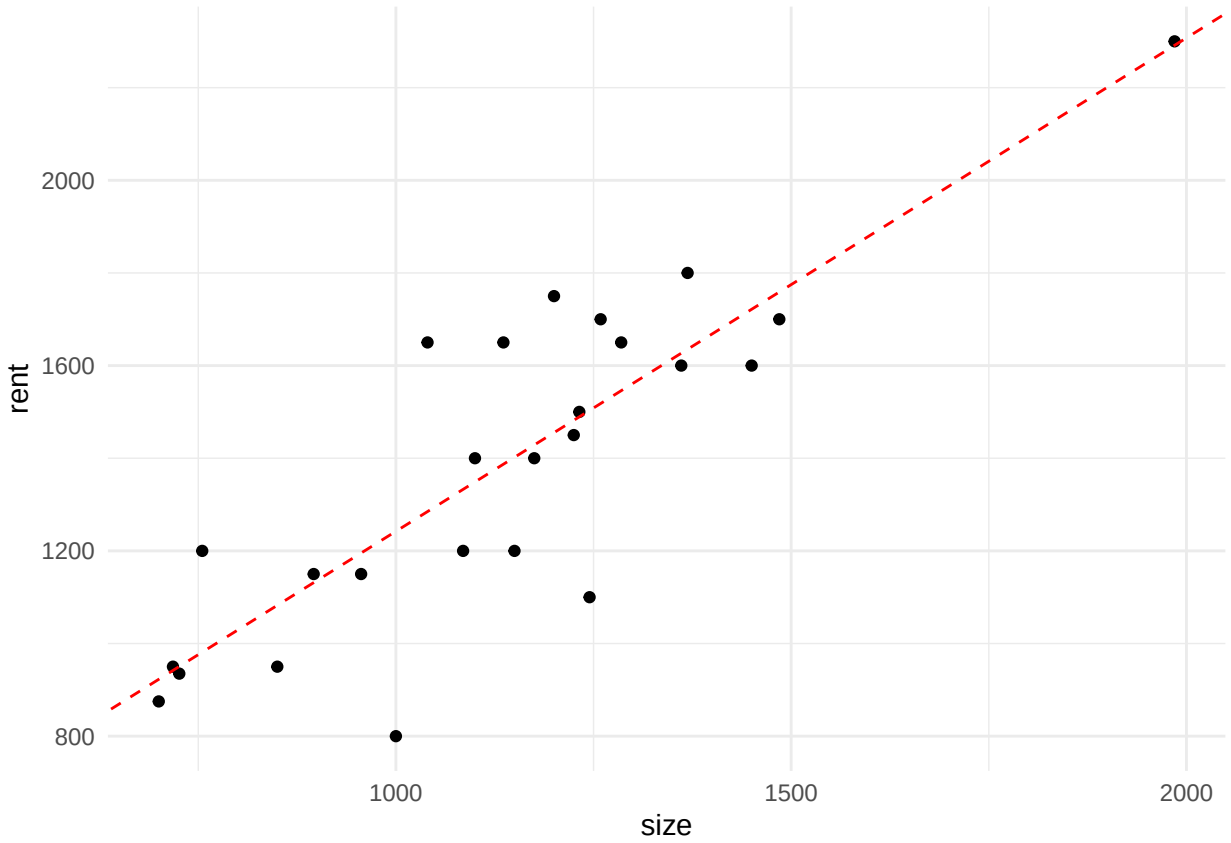
The output provides the estimated intercept and slope of the regression line. Additionally, it performs a t.test on the slope parameter with $H_0 : b = 0$ and $H_1 : b \neq 0$.
The summary also report the R^2 coefficient which measure the proportion of variation in Y which is explained

by X. If it is close to 1, then X predict the majority of the variability of Y.
We can visualize the regression line on the scatter plot:

```
plot(data$size, data$rent, main = "Scatterplot with Regression Line",
     xlab = "Size (sq ft)", ylab = "Rent ($)")
abline(model, col = "red")
```




```
ggplot(data, aes(x=size, y=rent))+
  geom_point() +
  theme_minimal() +
  geom_abline(intercept = model$coefficients[1],
              slope = model$coefficients[2], color="red",
              linetype="dashed")
```



Once we have the regression line, we can use the computed estimates to predict the values of Y for specific X. The `predict` function can be used to make predictions

```
predict(model, newdata = data.frame(size = 1800))
```

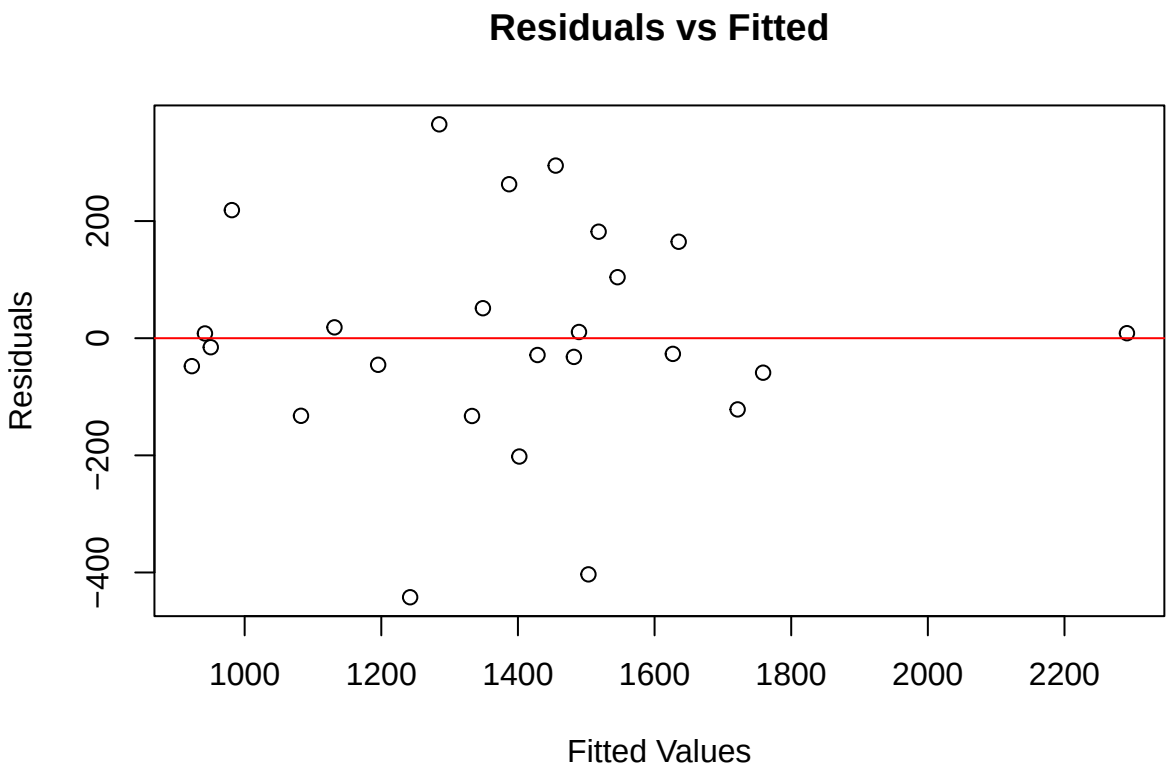
```
##       1
## 2094.38
```

Residuals measure the dispersion of points around the regression line and are crucial to evaluate the quality of the regression model. Considering the predicted value $\hat{Y}_i = \hat{a} + \hat{b}X_i$ for observation X_i, the i-th residual e_i is computed as $e_i = Y_i - \hat{Y}_i$.
When using linear regression model, the following assumptions are considered

- X and Y have a linear relationship.
- The distribution of the possible values of Y is normal.
- The variance of the values of Y is the same for each X.

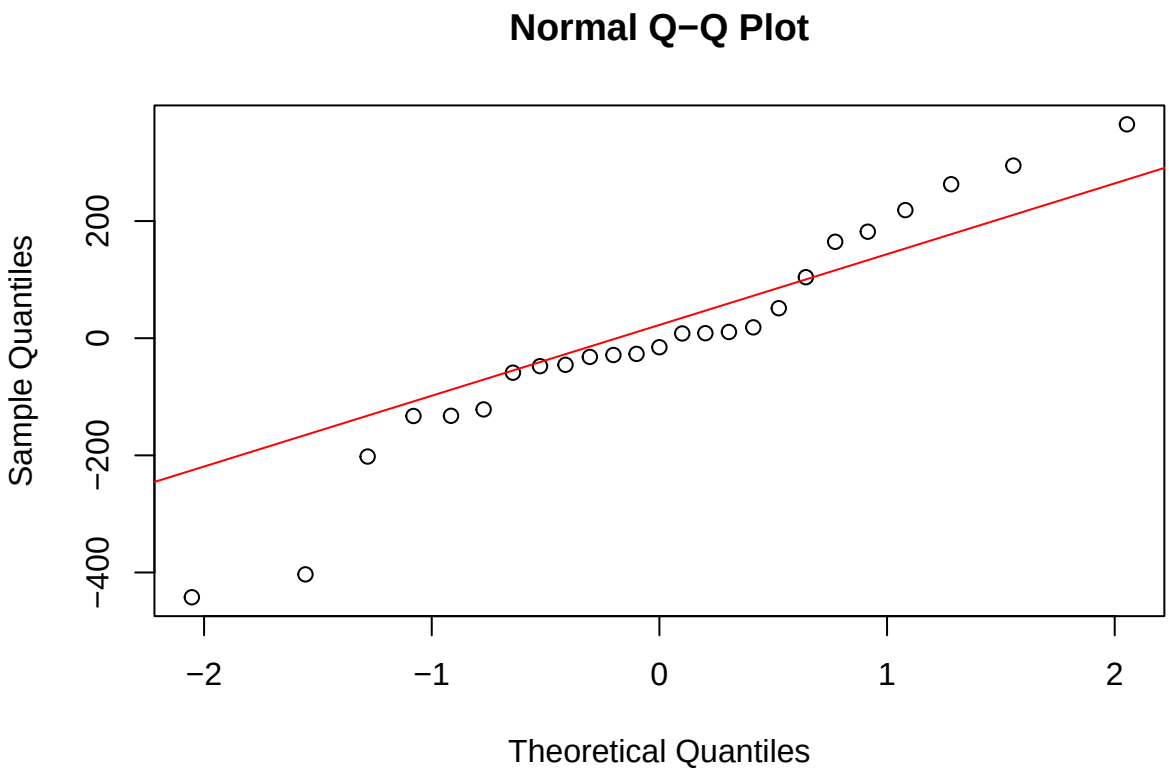
These can be verified analyzing the residuals.

```
plot(model$fitted.values, model$residuals, main = "Residuals vs Fitted",
     xlab = "Fitted Values", ylab = "Residuals")
abline(h = 0, col = "red")
```



Alternatively, we can also investigate the quantiles of the residuals using a qqplot.

```
qqnorm(model$residuals)
qqline(model$residuals, col = "red")
```



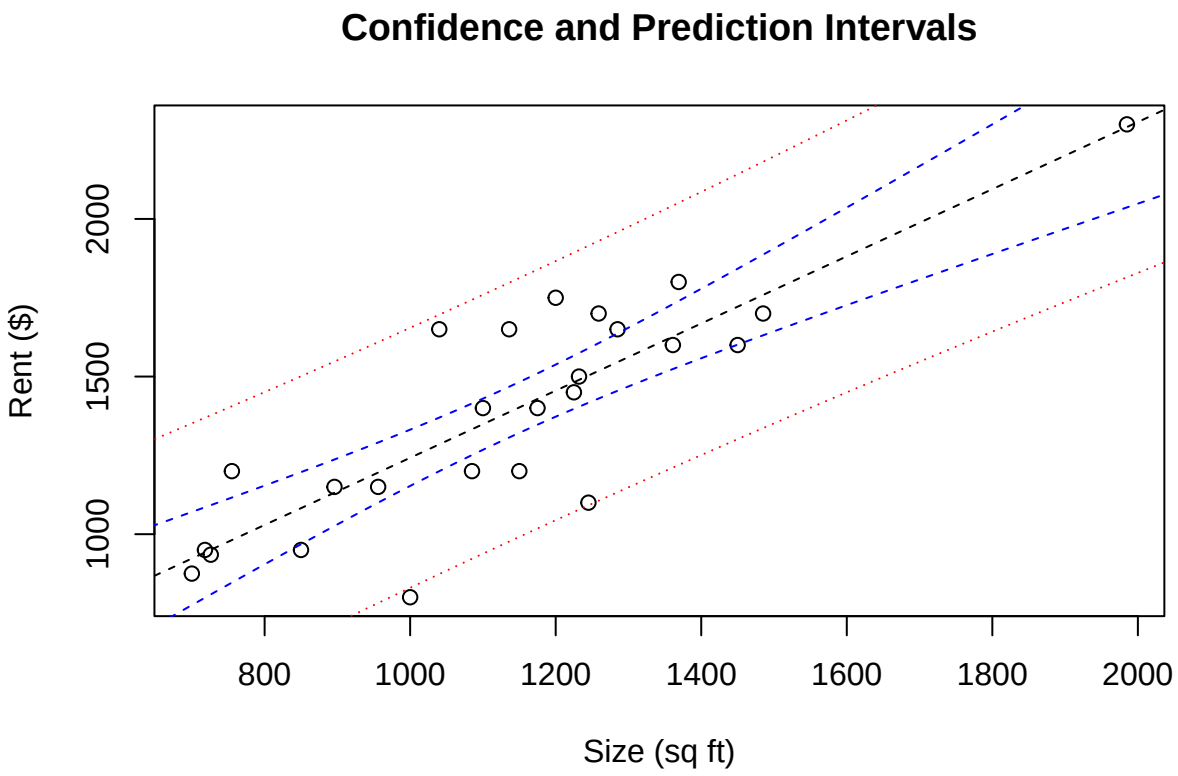
If the normality assumptions are satisfied, residuals should concentrate around the line $Y = 0$ and an equivalent dispersion of points above and below the line.

Confidence and Prediction Bands To assess uncertainty in the regression estimates, we calculate confidence and prediction intervals using the predict function with the interval argument

```
newdat <- data.frame(size=seq(250, 2200, 10))
# Confidence intervals
conf_int <- predict(model, interval = "confidence", newdata = newdat)

# Prediction intervals
pred_int <- predict(model, interval = "prediction", newdata = newdat)

# Visualizing intervals
plot(data$size, data$rent, main = "Confidence and Prediction Intervals",
     xlab = "Size (sq ft)", ylab = "Rent ($)")
abline(model, lty = 2)
lines(newdat$size, conf_int[, 2], col = "blue", lty = 2)
lines(newdat$size, conf_int[, 3], col = "blue", lty = 2)
lines(newdat$size, pred_int[, 2], col = "red", lty = 3)
lines(newdat$size, pred_int[, 3], col = "red", lty = 3)
```



Correlation and Cause-Effect

If two events are associated, one wonders if there is the possibility that one is the cause for the other. Notice that the presence of correlation between two variables does not imply a cause-effect relation, but for example it can be due to some common cause.

One possibility is the presence of *confounding variables*, that is a non-measured variable which is related to one or more of the measured variables.

Robust Statistics

The classical linear regression method, Ordinary Least Squares (OLS), is highly sensitive to violations of the assumptions. For example in presence of outliers, non-normality of errors, heteroscedasticity, that is constant variance of errors. Robust linear regression methods address these challenges by reducing the influence of outliers and when assumptions are violated. Furthermore, robust methods are designed such that they perform equivalently ot the OLS method if all the assumptions are satisfied.
Common approaches include

- *M-Estimators*: Generalization of maximum likelihood estimators, minimizing a robust loss function.

```
library(MASS)
modelM <- rlm(rent ~ size, data = data)
summary(modelM)
```

```
##
## Call: rlm(formula = rent ~ size, data = data)
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -447.43  -65.52  -19.74   98.12  359.84
##
## Coefficients:
##             Value    Std. Error t value
## (Intercept) 179.2019 157.2227     1.1398
## size          1.0682   0.1344     7.9495
##
## Residual standard error: 145.5 on 23 degrees of freedom
```

- *MM-Estimators*: Improves robustness by combining high breakdown point and high efficiency.

```
library(robustbase)
modelMM <- lmrob(rent ~ size, data = data)
summary(modelMM)
```

```
##
## Call:
## lmrob(formula = rent ~ size, data = data)
##  \--> method = "MM"
## Residuals:
##     Min      1Q  Median      3Q     Max
## -450.20  -67.30  -23.07   95.94  357.15
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 184.02581   82.29594   2.236   0.0353 *
## size          1.06618    0.06146  17.347 1.05e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Robust residual standard error: 155.4
## Multiple R-squared:  0.7708, Adjusted R-squared:  0.7608
## Convergence in 12 IRWLS iterations
##
## Robustness weights:
##  4 weights are ~= 1. The remaining 21 ones are summarized as
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3815  0.8391  0.9271  0.8608  0.9893  0.9980
## Algorithmic parameters:
##        tuning.chi                bb        tuning.psi        refine.tol
##         1.548e+00         5.000e-01         4.685e+00         1.000e-07
##           rel.tol         scale.tol         solve.tol          zero.tol
##         1.000e-07         1.000e-10         1.000e-07         1.000e-10
##       eps.outlier             eps.x warn.limit.reject warn.limit.meanrw
##         4.000e-03         3.611e-09         5.000e-01         5.000e-01
##      nResample         max.it       best.r.s       k.fast.s          k.max
##            500             50              2              1            200
##    maxit.scale      trace.lev            mts     compute.rd fast.s.large.n
##            200              0           1000              0           2000
##                  psi           subsampling                   cov
##            "bisquare"         "nonsingular"         ".vcov.avar1"
## compute.outlier.stats
##                  "SM"
## seed : int(0)
```

- *Least Trimmed Squares (LTS)*: Minimizes the sum of the smallest squared residuals, focusing on a

subset of data.

```
library(robustbase)
model <- ltsReg(rent ~ size, data = data)
summary(model)
```

```
##
## Call:
## ltsReg.formula(formula = rent ~ size, data = data)
##
## Residuals (from reweighted LS):
##     Min      1Q  Median      3Q     Max
## -223.95  -65.67  -11.98   81.12  344.15
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## Intercept 189.3257   127.3755   1.486    0.152
## size        1.0736     0.1085   9.891 2.35e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 152.3 on 21 degrees of freedom
## Multiple R-Squared: 0.8233,  Adjusted R-squared: 0.8149
## F-statistic: 97.83 on 1 and 21 DF,  p-value: 2.347e-09
```

- *Quantile Regression*: Models conditional quantiles (e.g., median or other percentiles) rather than the mean.

```
library(quantreg)
model <- rq(rent ~ size, data = data, tau = 0.5)  # Median regression
summary(model)
```

```
##
## Call: rq(formula = rent ~ size, tau = 0.5, data = data)
##
## tau: [1] 0.5
##
## Coefficients:
##             coefficients lower bd  upper bd
## (Intercept) 147.87530     96.42451 291.64525
## size          1.08419      0.95750   1.13032
```

- *Bayesian Robust Regression*: Bayesian approach to robust regression using heavy-tailed distributions (e.g., t-distribution).

```
library(brms)
model <- brm(rent ~ size, data = data, family = student())
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 3.8e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.38 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
```

```
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.128 seconds (Warm-up)
## Chain 1:                0.04 seconds (Sampling)
## Chain 1:                0.168 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 7e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.123 seconds (Warm-up)
## Chain 2:                0.044 seconds (Sampling)
## Chain 2:                0.167 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 7e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
```

```
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.123 seconds (Warm-up)
## Chain 3:                0.038 seconds (Sampling)
## Chain 3:                0.161 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 7e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.113 seconds (Warm-up)
## Chain 4:                0.042 seconds (Sampling)
## Chain 4:                0.155 seconds (Total)
## Chain 4:
```

```
summary(model)
```

```
##  Family: student
##   Links: mu = identity; sigma = identity; nu = identity
## Formula: rent ~ size
##    Data: data (Number of observations: 25)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept   181.40    158.37  -144.21   500.30 1.00     4132     2587
## size          1.06      0.13     0.79     1.33 1.00     4083     2739
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma   186.33     33.61   126.25   261.11 1.00     2272     1697
```

```
## nu       20.12     14.05     3.37    57.02 1.00     2957     2046
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

- *Nonparametric Regression* (Locally Weighted): Fits locally weighted regression models.

```
library(stats)
model <- loess(rent ~ size, data = data)
summary(model)
```

```
## Call:
## loess(formula = rent ~ size, data = data)
##
## Number of Observations: 25
## Equivalent Number of Parameters: 5.14
## Residual Standard Error: 210.4
## Trace of smoother matrix: 5.66  (exact)
##
## Control settings:
##   span     :  0.75
##   degree   :  2
##   family   :  gaussian
##   surface  :  interpolate      cell = 0.2
##   normalize:  TRUE
##  parametric:  FALSE
## drop.square:  FALSE
```

Exercises

1. Perform a linear regression analysis using the following data

```
height <- c(175, 168, 170, 171, 169, 165, 165, 160, 180, 186)
weight <- c(80, 68, 72, 75, 70, 65, 62, 60, 85, 90)
```

2. Regression of Carbon Content and Yield Strength. Conduct a linear regression analysis for the following data

```
carbon <- c(46, 27, 44, 35, 35, 25, 34, 29, 34)
yield <- c(71, 47, 63, 52, 55, 37, 49, 43, 48)
```

References

Wasserstein, RL. 2019. "Moving to a World Beyond 'p< 0.05'." Taylor & Francis.

Wilcox, Rand R, and Guillaume A Rousselet. 2018. "A Guide to Robust Statistical Methods in Neuroscience." *Current Protocols in Neuroscience* 82 (1): 8–42.

Yu, Zhaoxia, Michele Guindani, Steven F Grieco, Lujia Chen, Todd C Holmes, and Xiangmin Xu. 2022. "Beyond t Test and ANOVA: Applications of Mixed-Effects Models for More Rigorous Statistical Analysis in Neuroscience Research." *Neuron* 110 (1): 21–35.