Exercise 9. Confidence Interval estimates (one sample)

Michal Béreš, Martina Litschmannová, Veronika Kubíčková

Demonstration - what is interval estimation?

Consider a random variable following the normal distribution with a mean value of μ and a standard deviation of σ. We will work with selections from this random variable and using them we try to estimate the mean value of the distribution(here we know its true value, but in practice its value is unknown).

In [1]:

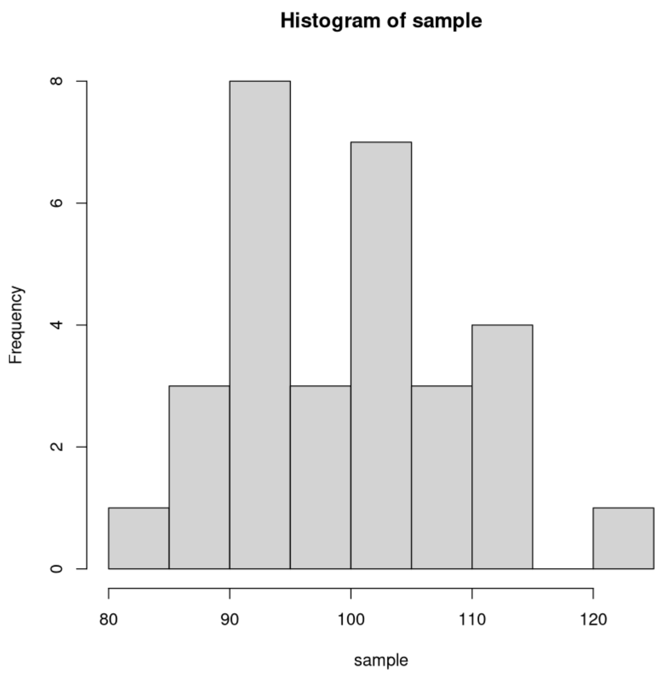
```
n = 30          # selection size
mu = 100        # mean value
sigma = 10      # st. deviation

# simulation of random selection from a given random variable
sample = rnorm(n = n, mean = mu, sd = sigma)

X = mean(sample) # sampling average as a point estimate
S = sd(sample)   # st. dev. of the sample
X
S
```

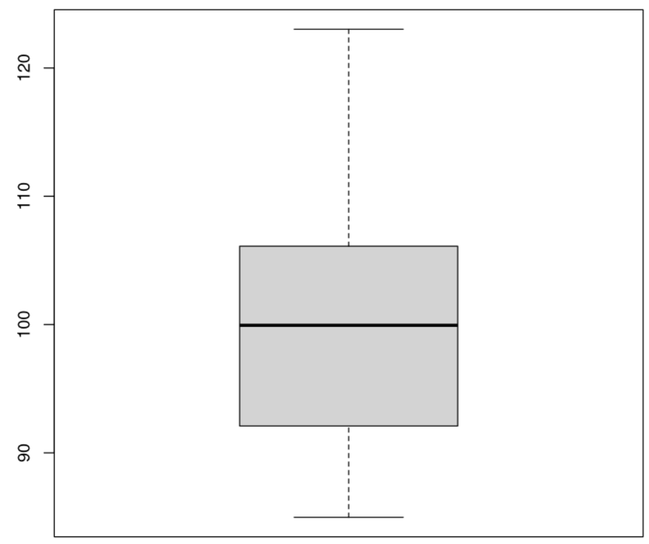
```
100.123187743497
9.44367119582114
```

For clarity, we can visualize the selection.

In [2]:

```
hist(sample)
boxplot(sample)
```





The construction of the confidence interval estimation using a selection characteristic

We will use this selection characteristic:(we assume that we do not know any real parameters of the distribution, only that it is following the normal distribution)

$$Y = \frac{\bar{X}-\mu}{S}\sqrt{n} \sim t_{n-1}$$

Since we know the distribution of Y, we are able to compute a a b in the following expression:

$$P(a < Y < b) \geq 1 - \alpha$$

- α is called the significance level(the probability that the searched value is outside our range)
- $1 - \alpha$ is called the interval estimation reliability

we pick a a b so that they are symmetric in probability, ie:

- $P(Y < a) \leq \alpha/2 \rightarrow a = t_{\alpha/2;n-1}$
- $P(b < Y) \leq \alpha/2 \rightarrow P(Y \leq b) \geq 1 - \alpha/2 \rightarrow b = t_{1-\alpha/2;n-1}$

In [3]:

```
# maximum probability with which we allow
# observations to lay outside the constructed interval
alpha = 0.05

# relevant quantiles of the student's distribution
t_low = qt(alpha/2, df = n-1)
t_high = qt(1 - alpha/2, df = n-1)

t_low
t_high
```

```
-2.0452296421327
2.0452296421327
```

Next we just add to the expression and modify:

$$P(t_{\alpha/2;n-1} < \frac{\bar{X}-\mu}{S}\sqrt{n} < t_{1-\alpha/2;n-1}) \geq 1 - \alpha$$

$$P(\bar{X} - t_{1-\alpha/2;n-1}\frac{S}{\sqrt{n}} < \mu < \bar{X} - t_{\alpha/2;n-1}\frac{S}{\sqrt{n}}) \geq 1 - \alpha$$

In [4]:

```
I_lower = X - t_high*S/sqrt(n)
I_upper = X - t_low*S/sqrt(n)
paste("P(", I_lower, " < μ < ", I_upper, ") ≥ ", 1-alpha)
```

```
'P( 96.5968629654515  < μ <  103.649512521542 ) ≥  0.95'
```

This particular estimate can also be obtained using the R function t.test:

In [5]:
```
t.test(sample, alternative = 'two.sided', conf.level = 1-alpha)$conf.int
```

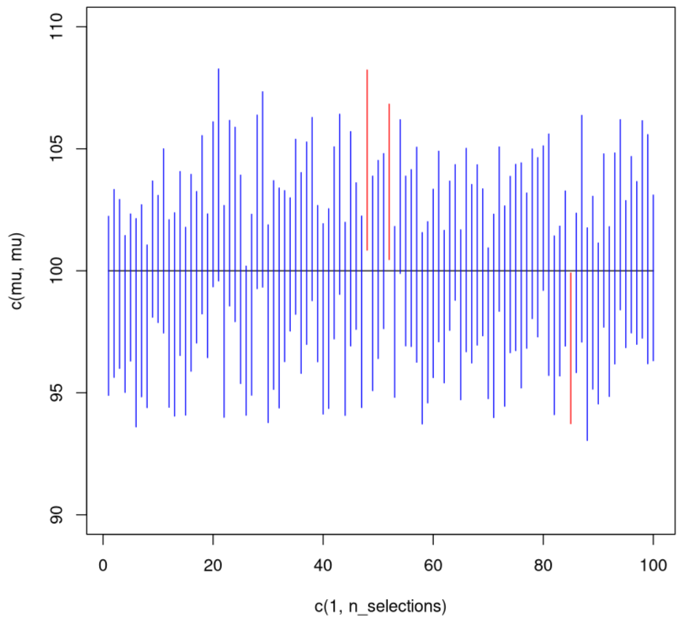
96.5968629654515 · 103.649512521542

Testing confidence intervals on multiple samples

We demonstrate what confidence level means.

In [6]:
```
n_selections = 100 # number of selections

n = 30              # selection size
mu = 100            # mean value
sigma = 10          # guided. deviation.

alpha = 0.05         # significance level

# relevant quantiles of the student's distribution
t_low = qt(alpha/2, df = n-1)
t_high = qt(1 - alpha/2, df = n-1)

# plot of the actual mean value
plot(c(1, n_selections), c(mu, mu), type = 'l', ylim = c(90,110))

count_failed = 0
# cycle through individual selections
for(i in 1:n_selections){
    vyber = rnorm(n = n, mean = mu, sd = sigma)
    X = mean(vyber)
    S = sd(vyber)
    I_lower = X - t_high*S/sqrt(n)
    I_upper = X - t_low*S/sqrt(n)

    # select the plot color, depending on whether the CI contains the mean
    if( I_lower<mu & mu<I_upper){color = "blue"}
    else{color = "red"
         count_failed = count_failed + 1}
    # plot the CI as a vertical line
    lines(c(i, i), c(I_lower, I_upper), col=color)
}
paste('alpha = ', alpha, ', ratio of failures of CI = ',
      count_failed/n_selections)
```

'alpha = 0.05 , ratio of failures of CI = 0.03'



Types of interval estimates

Examples of estimating the mean value of data from a normal distribution.

Bottom/Left IC

- $P(M_D^* < \mu) = 1 - \alpha$
- in R **alternative="greater"**

In [7]:
```
sample = rnorm(n = 30, mean = 100, sd = 10)
alpha = 0.05
t.test(sample, alternative = 'greater', conf.level = 1-alpha)$conf.int
```

```
96.8951724871622 · Inf
```

Top/Right IO

- $P(\mu < M_H^*) = 1 - \alpha$
- in R **alternative="less"**

In [8]:
```
sample = rnorm(n = 30, mean = 100, sd = 10)
alpha = 0.05
t.test(sample, alternative = 'less', conf.level = 1-alpha)$conf.int
```

```
-Inf · 102.698491434627
```

Double-sided IC

- $P(M_D < \mu < M_H) = 1 - \alpha$
- in R **alternative="two.sided"**

In [9]:
```
sample = rnorm(n = 30, mean = 100, sd = 10)
alpha = 0.05
t.test(sample, alternative = 'two.sided', conf.level = 1-alpha)$conf.int
```

```
97.2220113040179 · 104.947262376058
```

Overview of selection parameters and their point/interval estimates

We usually have more CI constructions(functions in R that will do this for us), but each construction has different data requirements and produces different "quality"(in terms of IO size) estimates. We will always select the "best quality" CI that **has met** the prerequisites for use.

The order of the various CIs below will always be from "best" to most robust.

Position measures of one selection

By position measures we mean measures that determines the position of the data. For data from the normal distribution we can estimate the mean value, for others the median.

a) student's CI using t-test

- we estimate the mean value - the point estimate is the sample average
- the data must come from a normal distribution
- exploratory: skewness and sharpness lie in(-2,2)
- exploratory: The QQ graph has points approximately on the line
- exact: using a statistical test, eg Shapiro-Wilk test(shapiro.test(data))

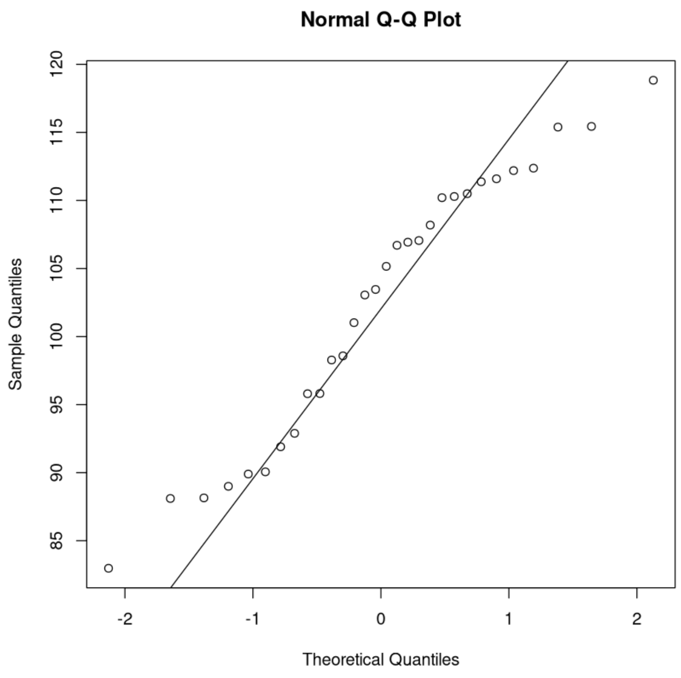
In [10]:
```
sample = rnorm(n = 30, mean = 100, sd = 10)
alpha = 0.05
```

In [11]:

```
# exploratory normality test
# library(moments) - we can avoid this by calling moments ::
# it's safer - we're sure we're calling a feature from this package
moments::skewness(sample)
moments::kurtosis(sample) - 3
qqnorm(sample)
qqline(sample)
```

-0.273507993503118

-1.13668944702712



In [12]:

```
# exact data normality test
shapiro.test(sample)$p.value
# the resulting p-value must be greater than significance level(eg 0.05)
```

0.132111001792141

In [13]:

```
# point estimate
mean(sample)
# IO
t.test(sample, alternative = 'two.sided', conf.level = 1-alpha)$conf.int
```

102.374139362453

98.6692481907854 · 106.07903053412

b) Wilcoxn CI test

- we estimate the median - the point estimate is the sample median
- the data must come from a symmetric distribution
- exploratory: the skewness lies in(-2,2)
- exploratory: the histogram looks approximately symmetrical
- function in Rk requires additional parameter(conf.int=TRUE)

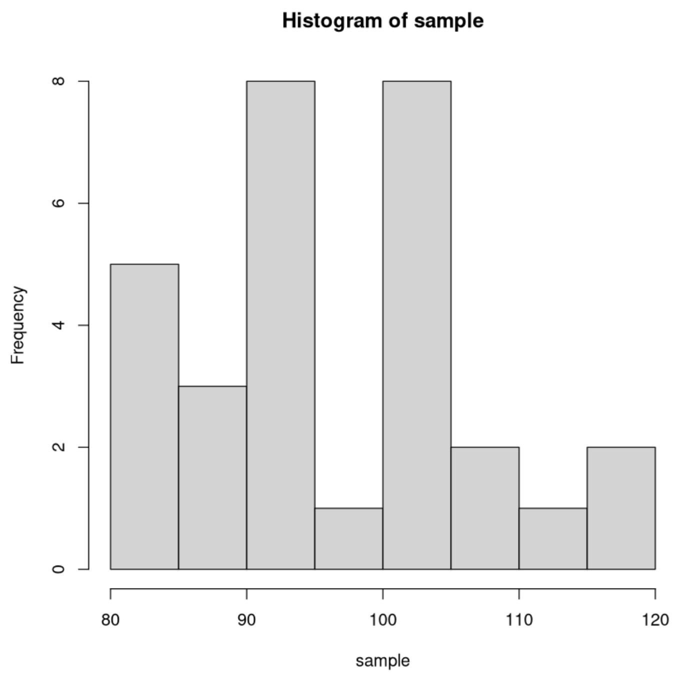
In [14]:

```
sample = runif(n = 30, min = 80, max = 120)
alpha = 0.05
```

In [15]:

```
# exploratory
moments::skewness(sample)
hist(sample, breaks = 6)
```

0.343665267915399



In [16]:
```
# point estimate
quantile(sample, probs = 0.5)
# or median(sample)
# IO
wilcox.test(sample, alternative = 'two.sided', conf.level = 1-alpha,
            conf.int = TRUE)$conf.int
```

50%: 94.8295037681237

92.652215026319 · 100.765396570787

c) sign test IO test

- we estimate the median - the point estimate is the sample median
- if we cannot use previous tests (no normality, no symmetry)
- function in R requires additional parameter(conf.int=TRUE)
- requires "BSDA" library
- as the most robust test, it can also be used for discontinuous data - eg order in a list

In [17]:
```
sample = rexp(n = 30, rate = 1/100)
alpha = 0.05
```

In [18]:
```
# true median
qexp(p = 0.5, rate = 1/100)
```

69.3147180559945

In [19]:
```
# point estimate
# quantile(select, probs=0.5)
median(sample)
# IO
# install.packages("BSDA")
BSDA::SIGN.test(sample, alternative = 'two.sided', conf.level = 1-alpha,
          conf.int = TRUE)$conf.int
```

51.7451141728088

20.5193844012165 · 83.8890039950543

Measures of variability of one selection

By measures of variability we mean measures determining the dispersion/variability of the data. For data from the normal distribution, we can estimate the standard deviation.

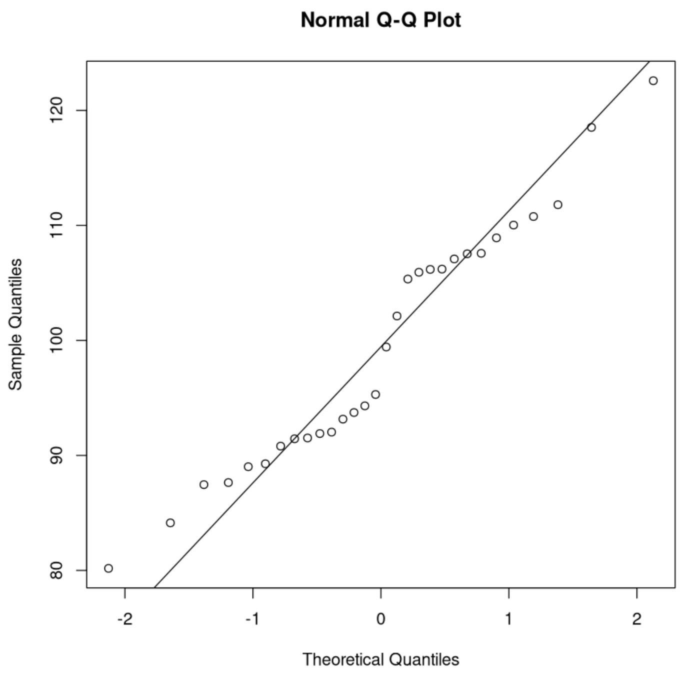
IO standard deviations

- we estimate the standard deviation - the point estimate is the sample standard deviation
- the data must come from a normal distribution
- exploratory: skewness and kurtosis lie in(-2,2)
- exploratory: The QQ graph has points approximately on the line
- exactly: using a statistical test, eg Shapiro-Wilk test(shapiro.test(data))
- requires "EnvStats" package
- function in R, gives the calculation of variance - the square root of the result is necessary

In [20]:
```
sample = rnorm(n = 30, mean = 100, sd = 10)
alpha = 0.05
```

In [21]:
```
# exploratory normality test
moments::skewness(sample)
moments::kurtosis(sample) - 3
qqnorm(sample)
qqline(sample)
```

```
0.244608734600265
-0.822982393579184
```



In [22]:
```
# exact data normality test
shapiro.test(sample)$p.value
# the resulting p-value must be greater than (eg 0.05)
```

```
0.204244233507958
```

In [23]:
```
# point estimate
sd(sample)
# CI
# install.packages("EnvStats")
result = EnvStats::varTest(sample, alternative = 'two.sided', conf.level = 1-alpha)$conf.int
sqrt(result)
```

10.6300938373495

LCL: 8.46588042798249 **UCL:** 14.2901965022992

Probability of occurrence with one selection

CI probability

- we estimate the probability - the point estimate is the relative frequency

- we need enough data: $n > \frac{9}{p(1-p)}$
- we have a lot of different options:
  - Clopper - Pearson estimate(binom.test) **preferred one**
    - does not take data as a parameter, but the number of successes and the number of observations
  - Wald's - from selection characteristics using normal distribution

In [24]:
```
pi = 0.3
n = 60
alpha = 0.05
sample = runif(n = n, min = 0, max = 1) < pi
```

In [25]:
```
# verification of assumptions
p = mean(sample)
p
9/(p*(1-p))
```

```
0.266666666666667
46.0227272727273
```

In [26]:
```
# point estimate
p
# Clopper - Pearson interval estimation
sample_size = length(sample)
n_successes = sum(sample)
binom.test(x = n_successes, n = sample_size, alternative = 'two.sided',
           conf.level = 1 - alpha)$conf.int
```

```
0.266666666666667
0.160746451590415 · 0.396610340025496
```

In [27]:
```
# Wald's interval estimation
dol_q = qnorm(alpha/2)
hor_q = qnorm(1-alpha/2)

p - hor_q*sqrt(p*(1-p)/n)   # lower IO limit
p - dol_q*sqrt(p*(1-p)/n)       # upper IO limit
```

```
0.15477247454567
0.378560858787663
```

In [28]:
```
# Calculation of the 11 most frequently used confidence intervals param. bin. distribution
# using binom package
# install.packages("binom")
binom::binom.confint(n = sample_size, x = n_successes)
```

A data.frame: 11 × 6

	method	x	n	mean	lower	upper
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	agresti-coull	16	60	0.2666667	0.1704828	0.3909308
2	asymptotic	16	60	0.2666667	0.1547725	0.3785609
3	bayes	16	60	0.2704918	0.1630277	0.3821541
4	cloglog	16	60	0.2666667	0.1627391	0.3820386
5	exact	16	60	0.2666667	0.1607465	0.3966103
6	logit	16	60	0.2666667	0.1702598	0.3918804
7	probit	16	60	0.2666667	0.1676588	0.3888210
8	profile	16	60	0.2666667	0.1660967	0.3867770
9	lrt	16	60	0.2666667	0.1660827	0.3867702
10	prop.test	16	60	0.2666667	0.1645226	0.3989020
11	wilson	16	60	0.2666667	0.1713264	0.3900871

Examples

In [29]:
```
# we may need theese
library(dplyr)
library(rstatix)
```

```
Attaching package: ‘dplyr’


The following objects are masked from ‘package:stats’:

    filter, lag


The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union



Attaching package: ‘rstatix’


The following object is masked from ‘package:stats’:

    filter
```

Example 1.

During control tests of 16 light bulbs, an estimate of the mean value of $\bar{x}$=3,000 hours and the standard deviation s=20 hours of their service life were determined. Assuming that the lamp life has a normal distribution, determine a 90% interval estimate for the mean value μ.

In [30]:
```
# We estimate the mean value of the lamp life
# Part of the input is information about data normality

n = 16          # sample size
x.bar = 3000    # hours.... average(point estimate of mean value)
s = 20          # hours.... sample standard deviation(point estimate of standard deviation)
alpha = 0.1     # significance level(reliability 1-alpha=0.9)
```

In [31]:
```
# two sided interval estimate of the mean
dol_q = qt(alpha/2,n-1)
hor_q = qt(1 - alpha/2,n-1)

x.bar - hor_q*s/sqrt(n)    # lower limit of IO
x.bar - dol_q*s/sqrt(n)    # upper limit of IO
```

```
2991.23474822154
3008.76525177846
```

Example 2.

The depth of the sea is measured with an instrument whose systematic error is zero and the random errors have a normal distribution with a standard deviation of 20 m. How many measurements do we need to take if we need 95% confidence interval of maximum size 20m = $< \bar{X} - 10, \bar{X} + 10 >$.

Remember:

Confidence interval have the form of: $P(\bar{X} - z_{1-\alpha/2}\frac{S}{\sqrt{n}} < \mu < \bar{X} - z_{\alpha/2}\frac{S}{\sqrt{n}}) \geq 1 - \alpha$

In [32]:
```
# We determine the estimate of the required selection range(number of required measurements)
# We assume data normality, with known variance(according to assignment)

sigma = 20    # meters.... known standard deviation
alpha = 0.05 # significance level(reliability 1-alpha=0.95)
delta = 10    # meters... permissible measurement error

# Estimate selection range
# we need to find n such z*S/sqrt(n)>10,
# where z is 1-alpha/2 quantile of the normal distribution
z_alpha = qnorm(1 - alpha/2,0,1)
(z_alpha*sigma/delta)^2
```

```
15.3658352827765
```

Example 3.

Suppose that in a random selection of 200 young men, 120 of them have higher than recommended serum cholesterol levels. Determine a 95% confidence interval for the percentage of young men with higher cholesterol levels in the population.

In [33]:
```
# We estimate the proportion of men with higher cholesterol levels in the entire population,
# ie the probability that a randomly selected man will have a higher cholesterol level

n = 200  # file range
x = 120  # number of "successes"
p = x/n  # relative frequency(probability point estimate)
p
alpha = 0.05 # significance level(reliability 1-alpha=0.95)
```

```
0.6
```

In [34]:
```
# Verification of assumptions
9/(p*(1-p))
```

```
37.5
```

In [35]:
```
# two sided Clopper - Pearson(exact) int. Estimate param. binom. distribution
binom.test(x,n,alternative="two.sided",conf.level=0.95)$conf.int
```

```
0.528535672797046 · 0.668453736264361
```

Example 4.

In a research study, we are working with a random selection of 70 women from the Czech population. Hemoglobin was measured in each of the women with an accuracy of 0.1 g/100 ml. The measured values are listed in the Hemoglobin.xls file. Find 95% interval estimates of standard deviation and mean hemoglobin in the population of Czech women.(Check the normality based on the exploration graphs.)

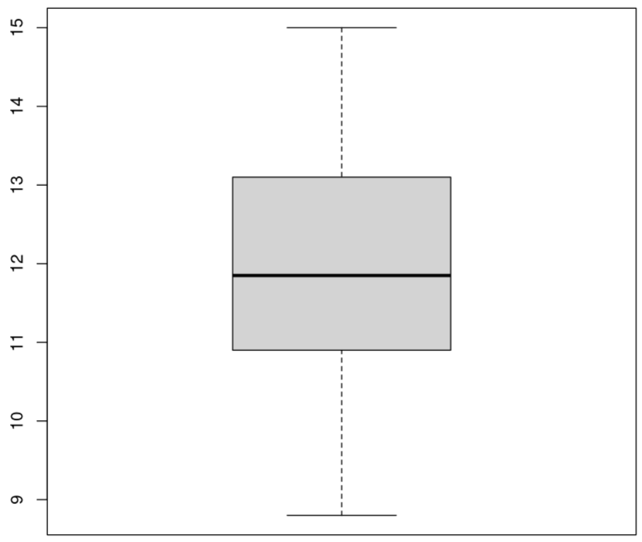
In [36]:
```
# We estimate the mean and standard deviation of hemoglobin in serum

# Read data from xlsx file(using readxl package)
hem = readxl::read_excel("data/intervalove_odhady.xlsx",
                  sheet = "Hemoglobin")
head(hem)
```

A tibble: 6 × 1

Hemoglobin [g/100 ml]
<dbl>
10.2
13.3
10.6
12.1
9.3
12.0

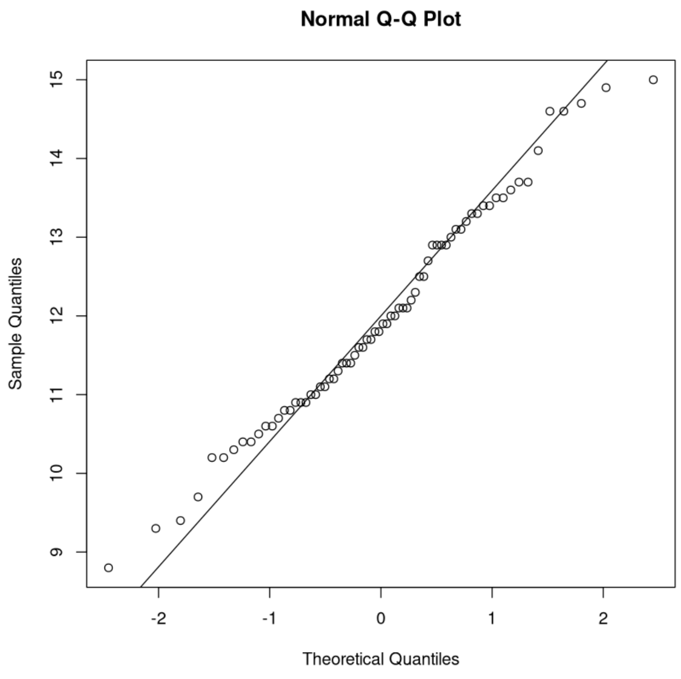
In [37]:
```
# lets rename the column for easier work
colnames(hem) = "value"
```

In [38]:
```
# Exploratory analysis
boxplot(hem$value)
# no outliers
```



In [39]:
```
# Verification of normality - exploratory
qqnorm(hem$value)
qqline(hem$value)

moments::skewness(hem$value)
moments::kurtosis(hem$value) - 3
# Both skew and sharpness meet the standards. distribution.
```

```
0.159328164635338
-0.504563833531858
```



In [40]:
```
# normality verification: exact - normality test.
# Shapirs. Wilk's test.
shapiro.test(hem$value)$p.value
# we cannot reject normality at significance 0.05
```

0.521873756408219

In [41]:
```
# 95% two sided interval estimate of the mean
mean(hem$value)
t.test(hem$value, altarnative="two.sided", conf.level=0.95)$conf.int
```

11.9828571428571

11.6459438718688 · 12.3197704138455

In [42]:
```
# # 95% two-way interval standard deviation estimate
sd(hem$value)
sqrt(EnvStats::varTest(hem$value, alternative = "two.sided", conf.level = 0.95)$conf.int)
```

1.41298034577871

LCL: 1.21151432999132 **UCL:** 1.69544380474259

Example 5.

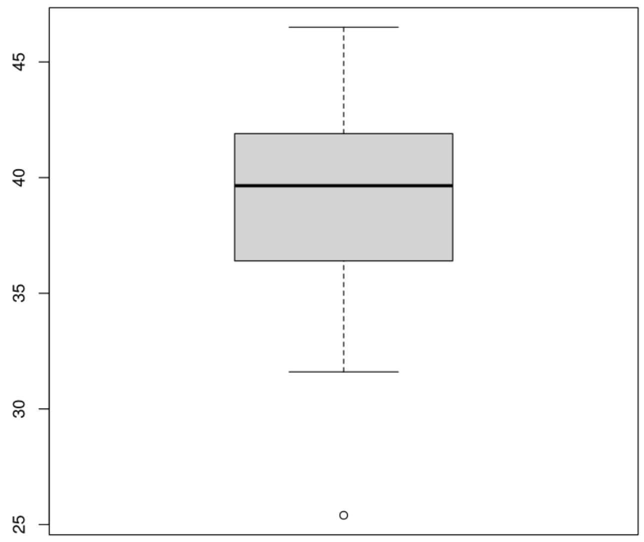
In the data file pr7.xlsx you will find the measurement of noise caused by the computer fan [dB]. Calculate the 95% interval estimate of the average noise and the 95% interval estimate of the noise variability.

In [43]:
```
# read data
data = readxl::read_excel("data/pr7.xlsx")
head(data)
```

A tibble: 6 × 2

ID	dB
<dbl>	<dbl>
1	36.4
2	35.5
3	40.1
4	41.8
5	43.5
6	36.4

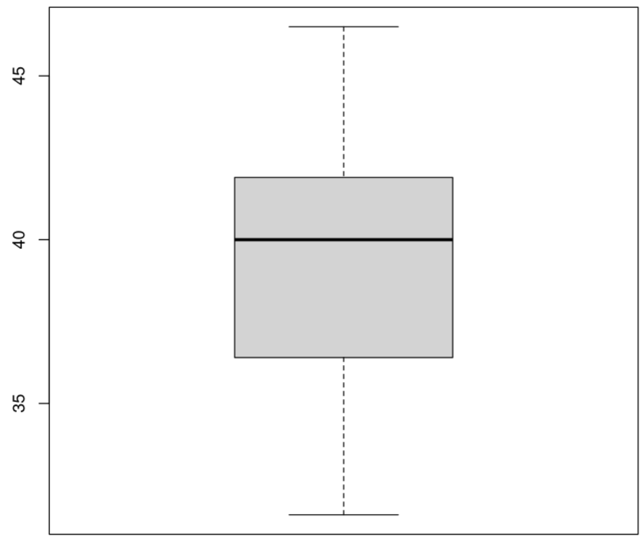
In [44]:
```
# visualization
boxplot(data$dB)
# there is an outlier!!
```



In [45]:
```
# removal of OP
outliers = data %>% identify_outliers(dB)
outliers

data$dB_no_outliar = ifelse(data$ID %in% outliers$ID,NA,data$dB)

boxplot(data$dB_no_outliar)
```

A tibble: 1 × 4

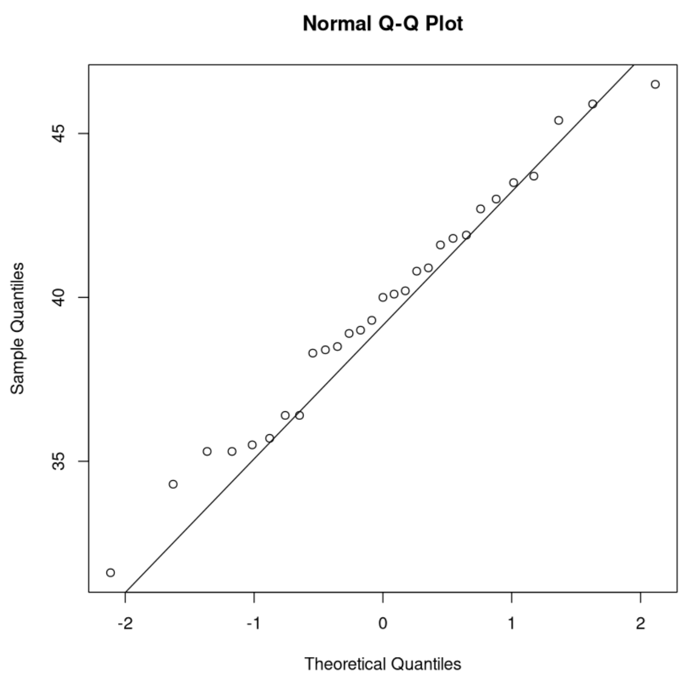
ID	dB	is.outlier	is.extreme
<dbl>	<dbl>	<lgl>	<lgl>
16	25.4	TRUE	FALSE



In [46]:
```
# data normality test exploratory
moments::skewness(data$dB_no_outliar, na.rm = TRUE)
moments::kurtosis(data$dB_no_outliar, na.rm = TRUE) - 3

qqnorm(data$dB_no_outliar)
qqline(data$dB_no_outliar)
```

-0.0813315523487833

-0.553070494607634



In [47]:
```
# normality test exactly
shapiro.test(data$dB_no_outliar)$p.value
```

0.914005694892269

In [48]:
```
# point and interval estimation of the mean
mean(data$dB_no_outliar, na.rm = TRUE)

t.test(data$dB_no_outliar, alternative = "two.sided", conf.level = 0.95)$conf.int
```

39.6862068965517

38.2871348190026 · 41.0852789741009

In [49]:
```
# point and interval estimation of the standard deviation
sd(data$dB_no_outliar, na.rm = TRUE)

res = EnvStats::varTest(data$dB_no_outliar,alternative = "two.sided", conf.level = 0.95)
sqrt(res$conf.int)
```

3.6780938520158

```
Warning message in is.not.finite.warning(x):
“There were 1 nonfinite values in x : 1 NA's”
Warning message in EnvStats::varTest(data$dB_no_outliar, alternative = "two.sided", :
“1 observations with NA/NaN/Inf in 'x' removed.”
```

LCL: 2.91885973658896 **UCL:** 4.97444431571921

Example 6.

In the data file pr8.xlsx you will find the measurement of the time to failure of the electrical component [h]. Calculate the 99% interval estimate of the average life of a given component type.

In [50]:
```
# read data
data = readxl::read_excel("data/pr8.xlsx")
head(data)
```

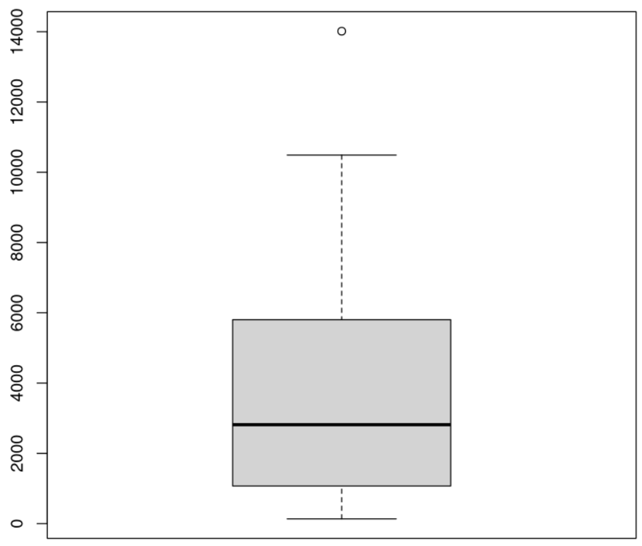
A tibble: 6 × 2

ID	cas_h
<dbl>	<dbl>
1	10488.8

ID	cas_h
<dbl>	<dbl>
2	4294.0
3	5804.5
4	830.3
5	2906.2
6	1821.3

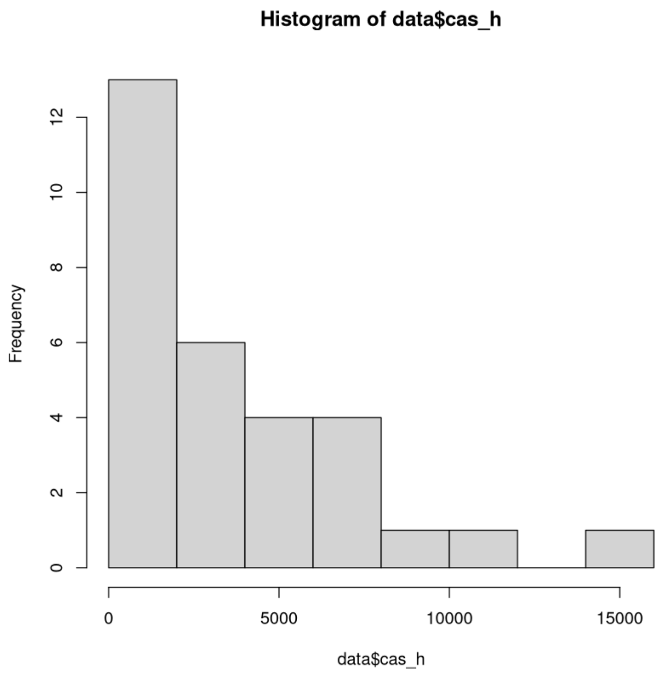
In [51]:

```
# visualization and verification of OP
boxplot(data$cas_h)
# there is an outliar, but is it really "bad" value?
# cannot we assume that the data came from exponential dist.?
```



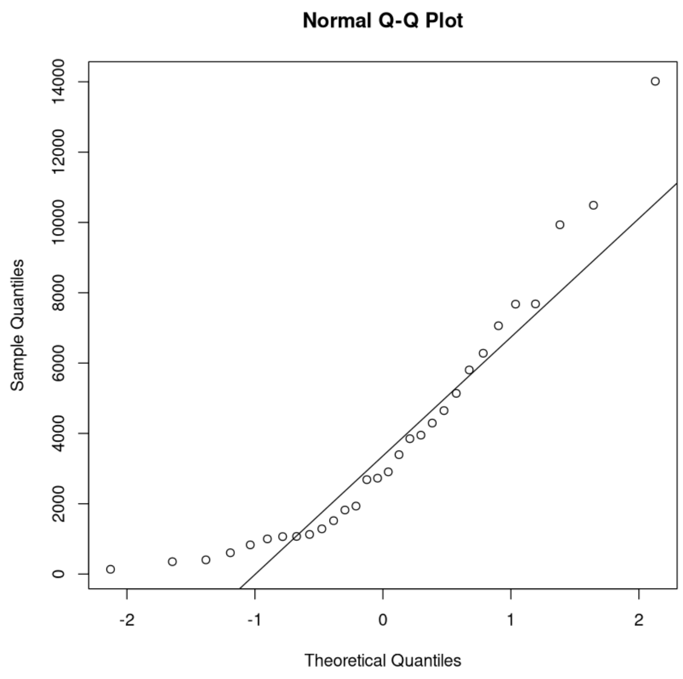
In [52]:

```
hist(data$cas_h)
# looks very likeli as exponential distr.
# we keep the outliar as its not really outliar
# we already know we dont have normality and symmetry
```



In [53]:
```
# data normality test exploratory
moments::skewness(data$cas_h)
moments::kurtosis(data$cas_h) - 3

qqnorm(data$cas_h)
qqline(data$cas_h)
```

```
1.17885927322967
0.847440261606931
```



In [54]:
```
# median point and interval estimation
median(data$cas_h)
# IO
# install.packages("BSDA")
alpha = 0.01
BSDA::SIGN.test(data$cas_h, alternative = 'two.sided', conf.level = 1-alpha,
          conf.int = TRUE)$conf.int
```

```
2816.5
1095.41958211663 · 5513.10320526916
```

Example from slides - 2

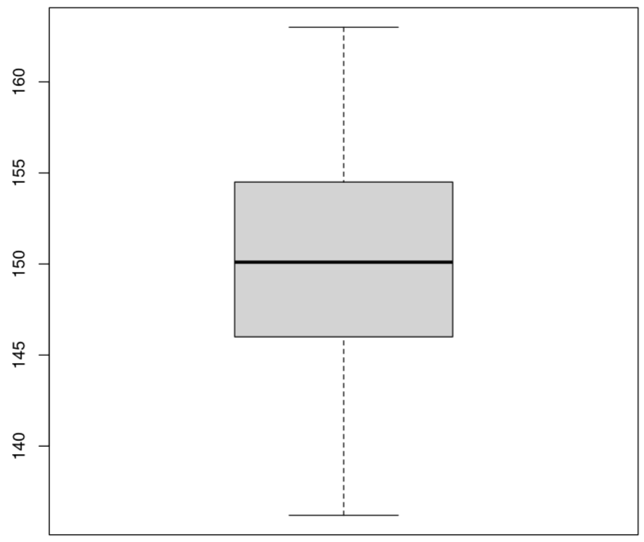
Company FactoryX produces packages with cocoa. The weight (in grams) of a randomly chosen packages is recorded in the dataset (cocoa.csv). Perform EDA and the two-sided 95% confidence interval for the mean weight of the packages from the whole production (or for median if necessary).

In [55]:
```
data = read.csv2("data/cocoa.csv")
head(data)
```

A data.frame: 6 × 2

	ID	weight
	<int>	<dbl>
1	1	144.1
2	2	155.5
3	3	144.4
4	4	156.0
5	5	144.8
6	6	155.9

In [56]:
```
boxplot(data$weight)
```



In [57]:

```
shapiro.test(data$weight)
```

```
	Shapiro-Wilk normality test

data:  data$weight
W = 0.99128, p-value = 0.9714
```

In [58]:

```
t.test(data$weight, alternative = "two.sided")
```

```
	One Sample t-test

data:  data$weight
t = 186.58, df = 49, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 148.7623 152.0017
sample estimates:
mean of x 
  150.382 
```

Example from slides - 3

Let's assume, that 30 out of 100 asked students in our university are smokers. What is the two-sided 95% confidence interval of the proportion of smokers among all university students? (The source data are also available in smokers.csv.)

In [59]:

```
data = read.csv2("data/smokers.csv")
head(data)
```

A data.frame: 6 × 2

	ID	Smokers
	<int>	<chr>
1	1	Y
2	2	Y
3	3	Y
4	4	Y
5	5	Y
6	6	Y

In [60]:

```
tab = table(data$Smokers)
```

```
tab
```

```
 N  Y 
70 30 
```

In [61]:

```
n = sum(tab)
n
x = tab['Y']
x
```

100

Y: 30

In [62]:

```
p = x/n
p
9/(p*(1-p))
```

Y: 0.3

Y: 42.8571428571429

In [63]:

```
binom.test(x,n,alternative="two.sided",conf.level=0.95)
```

```
	Exact binomial test

data:  x and n
number of successes = 30, number of trials = 100, p-value = 7.85e-05
alternative hypothesis: true probability of success is not equal to 0.5
95 percent confidence interval:
 0.2124064 0.3998147
sample estimates:
probability of success 
                   0.3 
```

Example from slides - 4

The hospital observed 50 patients with lung cancer and recorded their survival time in years (time.csv). Find the left-sided 95% confidence interval for the mean time of survival (or for median if necessary).

In [64]:

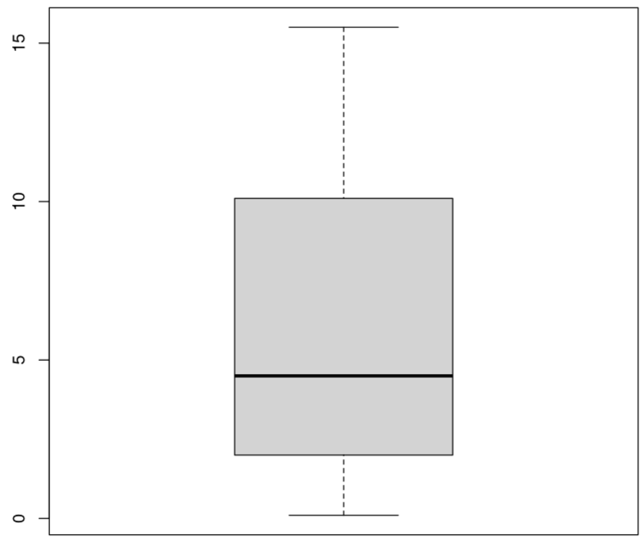
```
data = read.csv2("data/time.csv")
head(data)
```

A data.frame: 6 × 2

	ID	values
	<int>	<dbl>
1	1	3.5
2	2	3.2
3	3	2.6
4	4	0.2
5	5	7.2
6	6	5.1

In [65]:

```
boxplot(data$values)
```

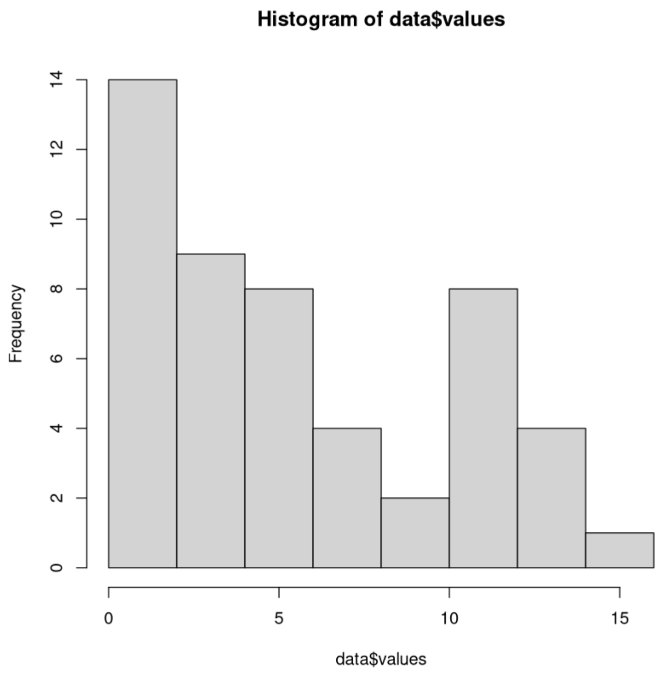


```
In [66]: shapiro.test(data$values)
```

```
	Shapiro-Wilk normality test

data:  data$values
W = 0.9132, p-value = 0.001351
```

```
In [67]: moments::skewness(data$values)
         hist(data$values)
         # no normality, but we can assume symmetry
```

0.603133016234218



```
In [68]: median(data$values)
         wilcox.test(data$values,alternative = "greater",conf.int = TRUE, conf.level = 0.95)$conf.int
```

4.5

4.20002257072764 · Inf