ST516: Computational Statistics Exam (Part A)

Katrine Eriksen and Katrine Bach

10 juni 2016

```
#install.packages("devtools")
#install.packages("roxygen2")
#library(devtools)

#install_github("Katrinebch/ExamST516")
#library(ExamST516)
```

Task 1

In this task we want to compare the effect of daily sport activity on the semester's grade avarage. To do this we use the data set sport.txt to calculate the following:

1. The correlation between the variables of interest.
2. The bootstrap estimate of correlation,
3. The bootstrap estimation of the standard error,
4. The bias
5. The 95% considence interval.

To begin with, we will explain the theory behind the concepts that is needed.

Correlation:

This section is based on (Sheldon 2013a).

$E[X]$ is the expected value of the random variable X, this value is the weighted average of the possible values of X. The value of $E[X]$ does not say anything about the variation of the these values. One way to measure this variance is to consider the average value of the squares of the difference between X and $E[X]$.

Definition: If X is a random variable with mean μ, then the variance of X, denoted by $Var(X)$, is defined by

$$Var(X) = E[(X - \mu)^2]$$

The covariance between to random varibles can be defined by the following

Definition: The covariance of two random variables X and Y, denoted $Cov(X, Y)$, is defined by

$$Cov(X, Y) = E[(X - \mu_x)(Y - \mu_y)]$$

where $\mu_x = E[X]$ and $\mu_y = E[Y]$.

The correlation between two random variables X and Y, denoted as $Cor(X, Y)$, is defined by

$$Cor(X, Y) = \frac{Cov(X, Y)}{\sqrt{Var(X)Var(Y)}}$$

Boostrap:

This chapter is based on (Sheldon 2013b).

The Boostrapping Technique was developed by Efron in 1979 and was inspired by the Jacknife method which is a method especially useful for variance and bias estimation. The boostrap technique is a nonparametric Monte Carlo method that estimate the population distribution by resampling from an observed sample. The resampling method allows us to estimate population charachteristics and make inference about them. We use this method when the populations mean μ is unknown.

Suppose that $X_1, \ldots, X_n$ are independent random variables that have a common distribution F. We are interested in using the variables to estimate some parameter $\theta(F)$ which could be the mean of F. Suppose further that an estimator of $\theta(F)$, where this estimator is called $g(X_1, \ldots, X_n)$, has been proposed. In order to judge its worth as an estimator of $\theta(F)$ we are interested in estimationg its mean square error.

$$MSE(F) \equiv E_F[(g(X_1, \ldots X_n) - \theta(F))^2]$$

As an immediate estimator of the above MSE which is S^2/n when $\theta = E[X_i]$ and $g(X_1, \ldots, X_n) = \bar{X}$, it is not that obvious how it can be estimated otherwise. We therefore present the boostrap technique for estimation the mean square error. We first note that if the distribution function F were known it would be possible to compute the mean square error. Based on the observed data points X_i it is possible to estimate the underlying distribution function F by a so called empirical distribution function F_e

$$F_e(x) = \frac{\text{number of i:} X_i \leq x}{n}$$

If the empirical distribution function F_e is close to F which it should be in the case where n is large. Then $\theta(F_e)$ probably be close to $\theta(F)$ when it is assumed that θ is a continous function of the distribution. The $MSE(F)$ should approximately be equal to

$$MSE(F_e) = E_{F_e}[(g(X_1, \ldots, X_n) - \theta(F_e))^2]$$

In this expression the X_i are to be regarded as being independent random variables having the distribution function F_e. This $MSE(F_e)$ is called the *Boostrap approximation to the mean square error* $MSE(F)$.

For the boostrap technique we can estimate the standard error by the following

$$\hat{se} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{n}(\hat{x_i} - \bar{\hat{x}})^2}$$

To make the bootstrap estimate of the bias, we use the following

$$\text{bias}(\hat{\theta}) = E[\hat{\theta} - \theta] = E[\hat{\theta}] - \theta$$

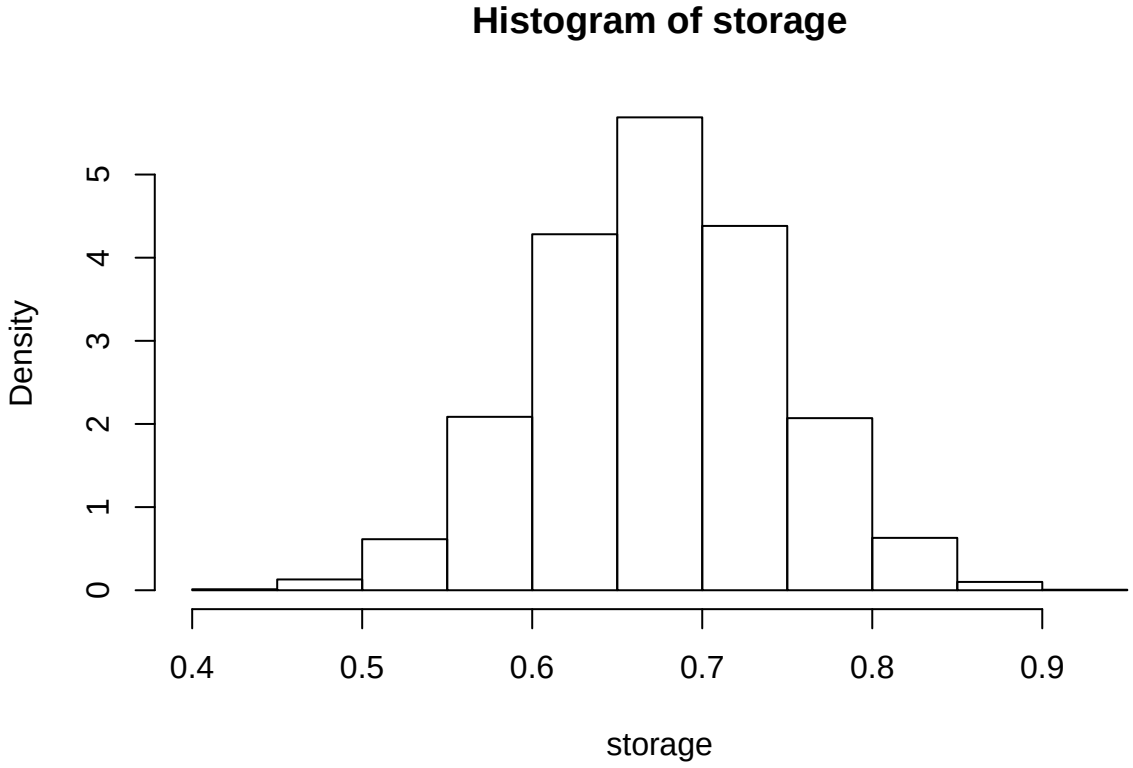
The simplest form of the confidence interval relies on the central limit theorem. This implies that it requires a large sample to be effective. It is assumed that $\hat{\theta}$ is unbiased and we have a normal distribution. Then θ is in the Z interval

$$\hat{\theta} \pm \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)$$

α is the p-value and because we want a 95% confidence interval, then α should be 0.05.

Now we use the explained theory above to write a function, that answer the questions giving in the task.

```r
URL <- "https://raw.githubusercontent.com/haghish/ST516/master/data/sport.txt"
data <- read.table(URL, header = TRUE)
x <- data$Sport
y <- data$Grades

bootstrap <- function(n,x,y){
  #Warnings and stop
  if(n<1){stop('n must be larger or equal to 1')}
  if(!is.numeric(n)){stop("n must be numeric")}
  if(length(x)<2 ){stop("x must be a single row vector")}
  if(length(y)<2 ){stop("y must be a single row vector")}
  if(length(x)!=length(y)){stop("x and y must have the same length")}

  #Calculation the correlation between the variables og interest.
  theta.hat <- cor(x,y)

  #Standard Error
  set.seed(516)
  N <- nrow(data) # sample size (number of rows)
  storage <- numeric(n) #Store the variables
  for (i in 1:n) {
    k <- sample(1:N, size = N, replace = TRUE) # random indice
    storage[i] <- cor(x[k],y[k])
  }
  se <- sd(storage) #standard error
  hist(storage, probability = TRUE)

  #bias of sample correlation
  theta.hat.boot <- mean(storage)
  bias <- theta.hat.boot - theta.hat

  #95% confidence interval
  alfa = 0.05
  plus <- theta.hat.boot + qnorm(1-(alfa/2))*se
  minus <- theta.hat.boot - qnorm(1-(alfa/2))*se
  return(list("exact_corr"=theta.hat,"boot_corr"=theta.hat.boot, "se"=se, "bias"=bias, "confidence_inte
}


bootstrap(10000, x, y)
```




```
## $exact_corr
## [1] 0.6672733
##
## $boot_corr
## [1] 0.6749195
##
## $se
## [1] 0.06964434
##
## $bias
## [1] 0.007646232
##
## $confidence_interval
## [1] 0.8114199 0.5384191
```

By using the R build-in function *cor* we found that the correlation between sport and grades are $cor(x, y) = 0.6673$.

From our function we see that the bootstrap estimate of correlation is 0.6754. It is possible to see that the two values of correlation lies close to each other, hence the bootstrap method for estimating the correlation is valid. It should not be expected that these values are exactly the same, as the bootstrap method only approximate the correlation.

The standard error of bootstrap estimation is found by using the formula from the theory. We get the standard error of the bootsrap estimation is 0.0698. This value can be interpreted as a measure of how good the approximation is. If the value is low this indicates that the approximation fits the observed values. On the other hand if the standard error is large the approximated values fits the observed values poorly. From this our standard error indicates that the bootstrap method approximate the values well, but there is still room for improvement.

The bias estimates the difference between the expected values of the approximation and the exact value. For a good approximation this implies that the difference between the two values must be small. Our value is 0.0081 which indicates that the estimator is close to being unbiased.

A confidence interval of 95% reflects a significance level at 0.05. When it is stated that we are 95% confident that the true value of the parameter is in our confidence interval, we express that 95% of the hypothetically observed confidence intervals will hold the true value of the parameter. In our case we found that the 95% confidence interval is $0.53852490 - 0.8122820$ which implicates that if the parameter lies within this interval we are 95% confident that the paramter estimate is true, hence it can be accepted. Our bootstrap estimate is 0.0698 which is in the confidence interval, hence the parameter can be accepted.

Task 2

In this task we have to simulate two problems "Buffon's Needle Problem" and "Lazzarini's experiment". First we want to explain the idea of "Buffon's Needle Problem" and cover the theory we use.

Buffon's Needle Problem:

Buffon's needle problem is a question that was first posed in the 18th century by Georges-Luis Leclerc. The problem was the following. Suppose we have an infinit floor with parallel lines and we let d be the distance between the lines on the floor. If we then drop a needle onto the floor, what is the probability that the needle will lie across a line? For solving this problem let the needle have a length l with y being the distance from the lower end of the needle to the nearest gridline above it. Further more θ measures the smallest clockwise angle form the grid direction to the needle. In the case where $l \leq d$, then the probability that the needle is crossing a line on the floor is

$$P(hit) = \frac{\int_0^\pi l \sin\theta d\theta}{\pi d} = \frac{2l}{\pi d} \approx 0.6366$$

The needle hits one of the gridlines if and only iff $y < l \sin\theta$

Monte Carlo:

This section is based on (Sheldon 2013c).

Suppose we want to compute θ where

$$\theta = \int_0^1 g(x)dx$$

To compute the value of θ then we can express θ as

$$\theta = E[g(U)]$$

where U is uniformly distributed over (0,1) for which it follows that $g(U_1), \ldots, g(U_n)$ are independant and identically distributed with the mean θ. By the strong law of large numbers it follows that

$$\sum_{i=1}^{n} \frac{g(U_i)}{n} \to E[g(U)] = \theta \text{ as } n \to \infty$$

Hence it is possible to approximate θ by generating a large number of random numbers u_i and taking as our approximation the avarge value of $g(u_i)$. This approach is called *the Monte Carlo* method.

In the case where we want to compute

$$\theta = \int_a^b g(x)dx$$

then by the use of substitution $y = \frac{x-a}{b-a}$, $dy = \frac{dx}{b-a}$ form which we can see that

$$\begin{aligned}\theta &= \int_0^1 g(a + (b-a)y)(b-a)dy \\ &= \int_0^1 h(y)dy\end{aligned}$$

thus we can approximate θ by continually generating random numbers and then taking the average value of h evaluated at these random numbers.

In our case we can, by the use of the *Monte Carlo* method, write

$$\int_0^\pi l \sin\theta d\theta$$

as

$$\frac{\sum_{i=1}^n f((b-a) * U_i + a)}{n}(b-a)$$

Standard Deviation:

The standard deviation expresses how a stochastic variable is located around its mean value.

$$\sigma = \sqrt{E[(X - E[X])^2]}$$

Standard Error:

The standard error is calculated by the use of the standard deviation σ. It is usually estimated by the standard deviation divided by the square root of the sample size.

$$\text{se} = \frac{\sigma}{\sqrt{(n)}}$$

Lazzarini's experiment:

In 1901 the italian mathematician Mario Lazzarini perfomed "Buffon's Needle experiment". He tossed a needle 3408 times and obtained the well-known estimate $\frac{355}{133}$ for π which was accurate to the six significant digits. Lazzarini chose needles of a length 5/6 of the width of the stripes. In this case he found that the probability that the needles will cross the line is $\frac{5}{3\pi}$.

In the first subtask we have to estimate the "Buffon's Needle experiment" using the Monte Carlo method and report the **P(hit)**. In addtion we have to compute and report the variance, standard deviation, standard error and the 95% confidence interval. This have we done in the following program.

```
#opgave 1
integralet <- 0
N <- 10000
l <- 1
d <- 1
```

```r
# estimating the integration
for (i in 1:N) {
  a <- 0
  b <- pi
  U <- runif(N)
  theta <- (b-a)*U+a
  f <- l*sin(theta)
  integralet[i] <- sum((f/N))*(b-a) #the estiamtion
  Phit<- integralet[i]/(pi*d) #calculate phit by the formel given in the theory.
  pie <- (2*l)/(Phit*d)
}
#Variance
varians <- var(integralet)
print(paste("Variance:", varians))
```

```
## [1] "Variance: 9.19971777535335e-05"
```

```r
#Standard Deviation
deviation <- sd(integralet)
print(paste("Standard deviation:", deviation))
```

```
## [1] "Standard deviation: 0.00959151592573007"
```

```r
#Standard Error
error <- deviation/(sqrt(N))
print(paste("Standard Error:", error))
```

```
## [1] "Standard Error: 9.59151592573007e-05"
```

```r
#the 95% confidence interval
alfa = 0.05
plus <- integralet + qnorm(1-(alfa/2))*error
minus <- integralet - qnorm(1-(alfa/2))*error
print(c(mean(minus),mean(plus)))
```

```
## [1] 1.999910 2.000286
```

In subtask 2 we should estimate **P(hit)** by performing "Buffon's Needle experiment". To do this we write a function that takes 3 arguments (N,l,d) and return the estimated value π. We have been given that $N = 10000$ and $l = d = 1$

```r
#Opgave 2
buffon <- function(N,l,d){
  set.seed(520)
  #warning and stop
  if(l>d){warning("l must be smaller than or equal to d")}
  if(!is.numeric(N)){stop("N must be a numeric")}
  if(!is.numeric(l)){stop("l must be a numeric")}
  if(!is.numeric(d)){stop("d must be a numeric")}
  if(N<1){stop("N must be larger or equal to 1")}
```

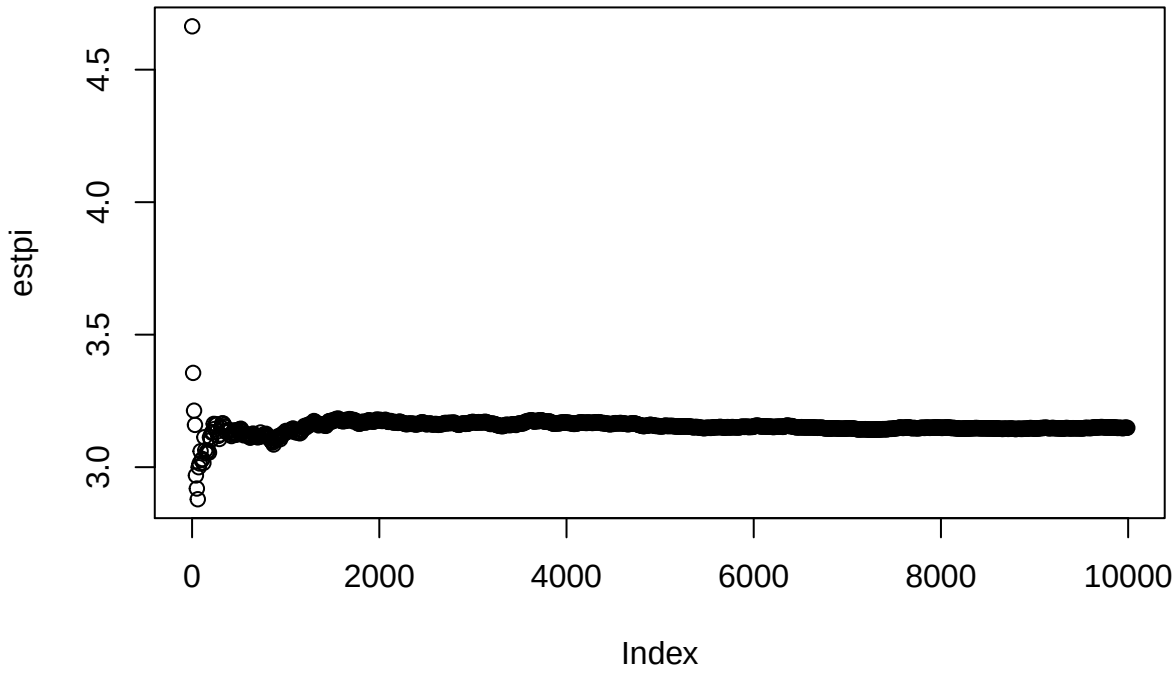
```
  if(d==0){warning("d should be bigger than zero")}
  if (d<1){stop("You cannot have a negative distance")}
  if (l==0){warning("It seems illogical to have a zero length")}
  if (l<0){stop("Not possible to have a negative length")}

  a <- 0
  b <- pi
  U <- runif(N)
  theta <- (b-a)*U+a
  f <- l*sin(theta)
  integralet <- sum((f/N))*(b-a) #Calculate the uppet part of the integral.
  Phit<- integralet/(pi*d) #calculate phit by the formel given in the theory.
  pie <- (2*l)/(Phit*d) #estimates pi
  return(pie)
}

buffon(10000,1,1)
```

```
## [1] 3.148795
```

Here we see that the estimated π we get is pretty close to the actual π which is 3.14159265. Obviously our estimate could be better, but we think it is an acceptable estimating of π.

In subtask 3 we have to repeat subtask 2 for values ranging from 1 to 10000 in intervals of 10. To do this we make a for-loop so we can see what the estimation for π is in every 10th interval.

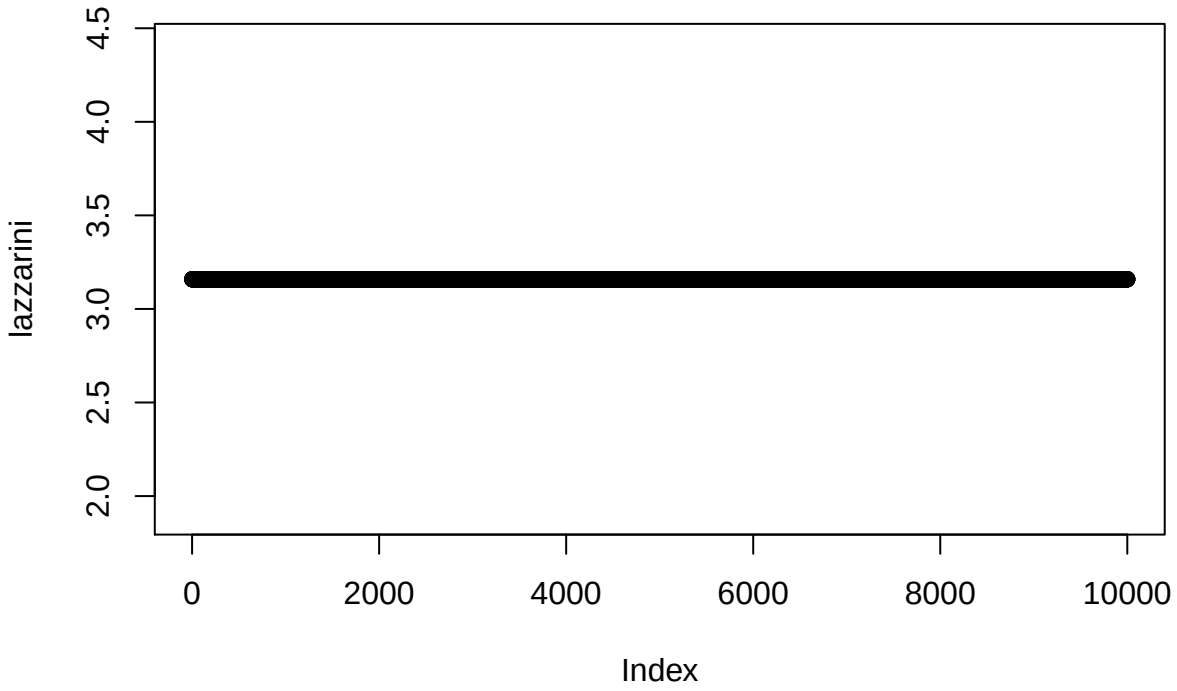
```
#Opgave 3
estpi <- 0
for (i in seq(1, 10000, 10)) {
  estpi[i]<-buffon(i,1,1) #estimates pi by using our function buffon
}
plot(estpi)
```



It can be seen that for a small value of N the estimate of π is very poorly, while these values shows a big diversity. From this plot we see that when N goes to infity then our estimation of π goes towards the exact π.

The last subtask is about Lazzarini's famous estimation of π using "Buffon's Needle experiment", which is accurate to the 6th decimal. We have to replicate his experiment a 10000 times where $N = 3408$, $l = 2.5$ and $d = 3$, and compare them to our own results.

```
#Opgave 4
lazzarini <- 0
for(i in seq(1,10000,1)){
  lazzarini[i] <- buffon(3408,2.5,3)
}
plot(lazzarini)
```



From the plot we see that the Lazzarini estimation of π is centered in the interval $3.10 - 3.17$. We get a plot that is more detailed by the Lazzarini estimation of π than our own. This implies that the Lazzarini experiment is better than our estimation of π. In our model we have estimations that varies from $2.7 - 4.4$ but most of them is centered in the interval $3.0 - 3.3$ which is not as good at the Lazzarini's experiment. We can therefore conclude that because Lazzarini have chosen $N = 3408$, $l = 2.5$ and $d = 3$, his estimation of π is much better than our estimation.

We think Lazzarini's choice of N is a little inapproriate because of the fact that it seems to specific. It would be more reliable because seems as if Lazzerini only chose this value because he already knew that he should get something like $\frac{355}{113}$ for running the eksperiment multiple times.

Task 3

A company creates gum for children. These gums comes with a photo of a soccerplayer. Gitte has purchased 301 cards.

This task is about whether we reject or accept a p-value. Again we first start by explaining the teory we use, which is χ^2-distribution and Goodness of fit test.

χ^2-distribution:

In the case where Y_i are independt, standard normal vairables with mean 0 and variance 1 then χ^2 distribution can be defined as

$$\chi^2 = \sum_{i=1}^{df} Y_i^2$$

where *df* is the *degrees of freedom*. The *degree og freedom* is the number of parameters which may be independently varied minus 1(Sheldon 2013d)

Goodness of fit:

This section is based on (Sheldon 2013d)

The idea of the Pearson chi-square test is to compare the observed sample points with the expected sample points. The hypothesis H_0 is rejected if the observed and expected numbers are too different.

We let N_j denote the observed number of sample points in A_j

$$N_j = \sum_{i=0}^{n} I_{A_j}(X_i)$$

The expected number under H_0 is given by:

$$E_{H_0} = \sum_{i=1}^{n} E_{H_0} I_{A_j}(X_i) = nP_{H_0}(X_i \in A_j)$$

The Pearson chi-square statistic combine N_j and E_{H_0} into a single expression in the following way

$$\chi^2 = \sum_{j=1}^{k} \frac{(N_j - E_{H_0}(N_j))^2}{E_{H_0}(N_j)}$$

We reject H_0 when the observed value of χ^2 is too large.

The p-value gives the probability of obtaining an outcome that is more extreme than the observed one if H_0 is true

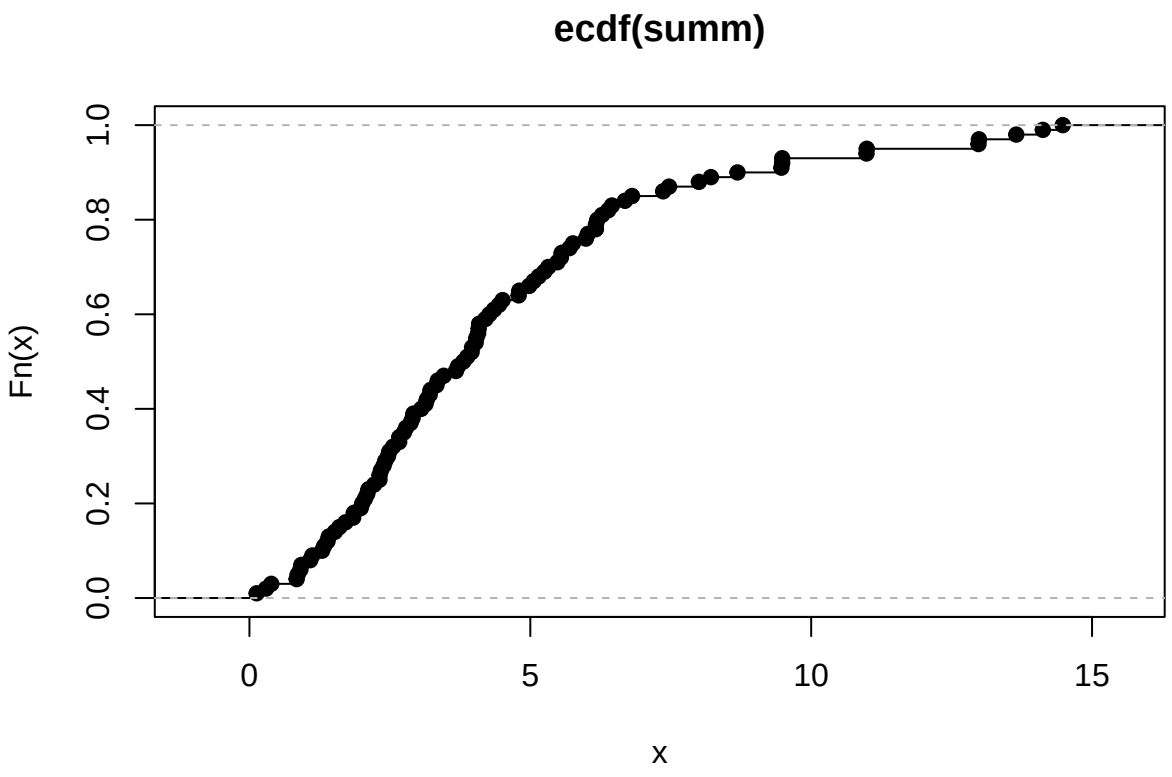
$$p - value = P(\chi^2_{k-r-1} > \chi^2_{obs})$$

where χ^2_{k-r-1} is denoting a chi-square random variables with $k - r - 1$ degrees of freedom. We reject H_0 if the p-value$< \alpha$.

In the first subtask we have to write a function that applies the Monte Carlo method and generate a χ^2 distribution. The function we make should return the probability that the χ^2 distribution is larger than the value x.

```
chi.probability <-function(x, df, n){
  #WARNING/STOPS
  if (!is.numeric(x)){stop("x must be numeric")}
  if (x<1){warning("Generates wrong chi-probability")}

  if(df==0){stop("The degrees of freedom must be greater than zero")}
  if (df<0){stop("The degrees of freedom must be positive")}
  if (!is.numeric(df)){stop("The degrees of freedom must be numeric")}
```

```r
  if(n==0){warning("n must be positive")}
  if (n<0){stop("Impossible to generate a amount of variables")}
  if (!is.numeric(n)){stop("n must be numeric")}


  set.seed(13)
  summ <- 0
  Y <- 0
  for (i in 1:df) { # makes a for loop that goes from 1 to degree of freedom.
    Y <- rnorm(n, mean=0, sd=1)
    Yianden <- Y^2 #Sets Y-square
    summ <- summ + Yianden # makes the sum of all the Yianden.
  }
  blup <- ecdf(summ)
  plot(blup)
  p <-1-blup(x) #calculates the p-value.
  return(p)

}
chi.probability(10,4,100)
```




```
## [1] 0.07
```

In the second subtask we have to write a function that takes two variables (x,p). This function should return the χ^2 goodness of fit value.

```r
GoodnessOfFit <- function(x,p=c(rep(1/length(x),length(x)))){
  #Warning/stops
  if(sum(p)!=1){stop("p must sum to 1 and be a vector of probabilities")}
  if(any(p<0)){stop("you can't have a negative probability")}
  if(length(x)!=length(p)){stop("x and p must have the same length")}

  set.seed(13)
  n<-301
  Exp<-n*p
  Pearson<-((x-Exp)^2/Exp) #calculates the sum for pearson chi-square
  GOF<-sum(Pearson,na.rm = FALSE) # Sum the pearson
  return(GOF)
}
```

In this function we have made the precautions that the if sum of p is not equal to 1 then the function should stop. This is because the sum of all probabilities must be one. We also check whether p is bigger than zero or not. A probabilitiy cannot be negative, and therefore we have to have this precaution. Also if the length of x is not equal to p then the function should stop because x and p must have the same values.

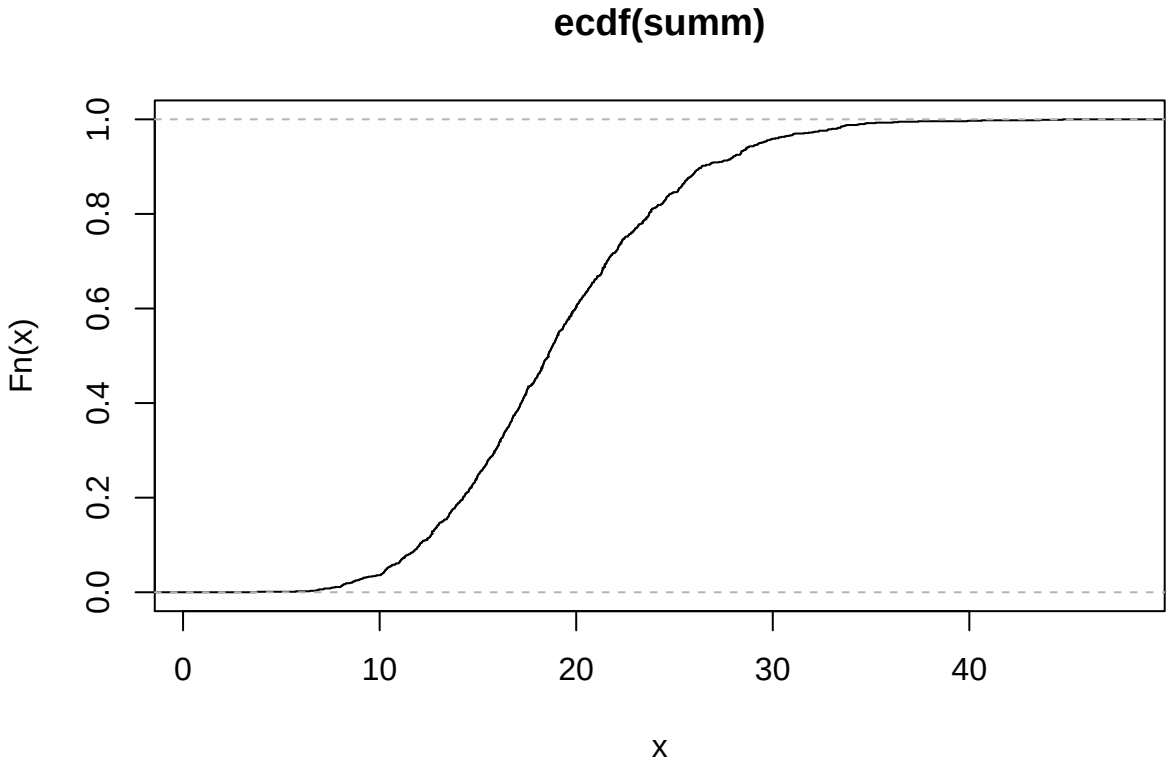
In this function we have made the precautions that the amount of observable values cannot be equal to zero, while this would imply that the probabilities within the p-vector was undefined. The length of the vector containing the observed values cannot be zero, since it is not possible to have a negative amount of observations. This also counts for the p-vector. One of the probabilities within the p-vector can not be zero because then the *Pearson* step will be undefined.

For the fourth subtask the company aclaims that all cards are equally likely to get in any shop. For validating this we make the following H_0 together with H_1 :

H_0: It is equally likely to get each of the different cards. This implies that the probability for getting each card is 1/20.

H_1: It is not equally likely to get each of the different cards. This is just the opposite of our null-hypothesis, so the H1-hypothesis implies that the probability for getting each card is not 1/20.

```r
URL <- "https://raw.githubusercontent.com/haghish/ST516/master/data/soccer.txt"
data <- read.table(URL, header = TRUE)
x<-data$Number
GOFit<-GoodnessOfFit(x)
chi.probability(GOFit,length(x)-1,1000)
```




```
## [1] 0
```

It can be seen that the p-value is zero. By interpreting this value the null-hypothesis can be rejected. Hence all cards are not equally distributed. This is also consistent with the observed values from Gitte.

For subtask five we want to show the difference between the χ^2 provided by R in contrast to the function for χ^2 by Monte Carlo made in one of the previous subtasks.

```
1-pchisq(GOFit,length(x)-1)
```

```
## [1] 1.257277e-08
```

It can be seen that these two p-values are not the same. Eventhough these values are not the same, they both reach the same conclusion that the H_0 hypothesis must be rejected. The difference is that the build-in function within R calculates the intregral numerically, while our function only makes an approximation of the integral using the Monte Carlo method.

The $P(\chi^2 > x)$ can be used to obtain the p-value, because this calculates the upper tail of the distribution.

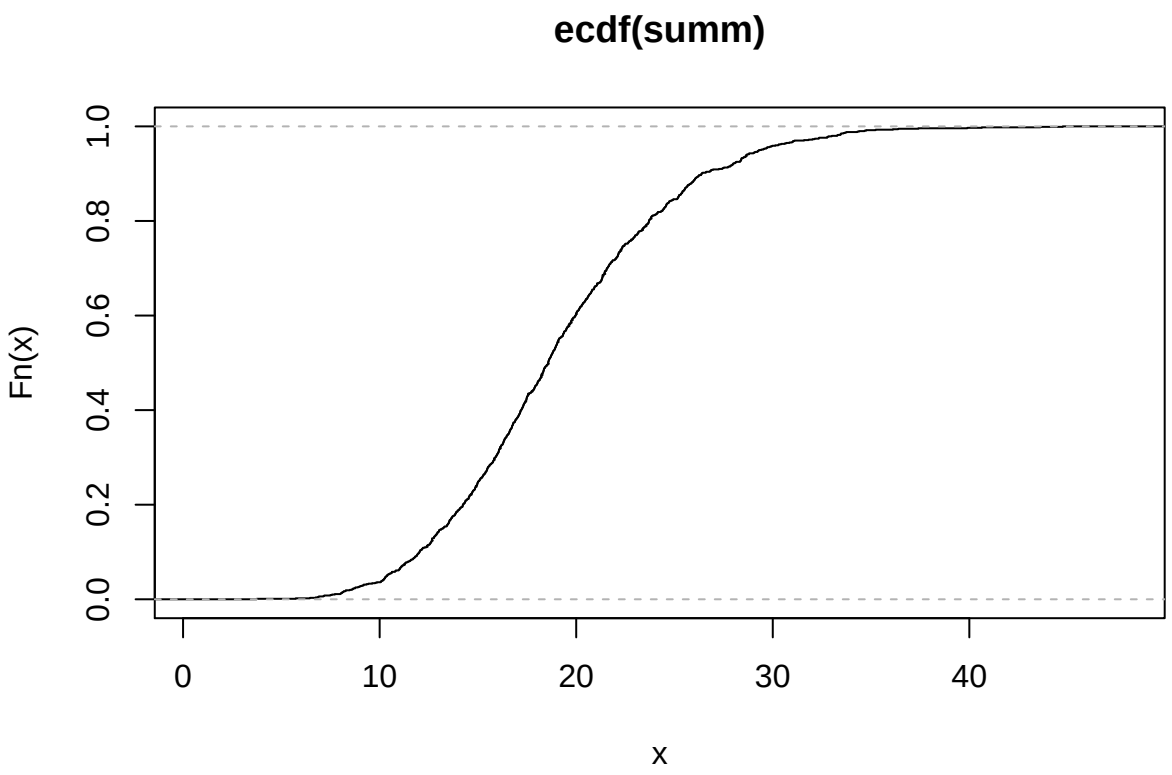
Since the model is a chi-squared Goodness of fit test the chi-square distribution is used to evaluate the observed data. The chi-squared distribution is used to find the lower tail, which is the sum of probability until x. The probability that $\chi^2 > x$ is then 1 minus the lower tail.

The p-value is the exact level of significance, which can also be stated as the probability of rejecting the H_0 in the case where it is actually true. Hence the p-value indicates how good the observed data fits the distribution. A high p-value indicate that we can have more confidence in the null hypothesis. If the p-value is low this indicate that the obsereved data is unlikely according to the null-hypothesis, which implies that the null-hypothesis must be wrong, hence it is rejected (n.d.).

In subtask 7 we assume that Gittes friend collects a large number of cards and calculates the probability of obtaining each player's card. We will make use of our functions *chi.probability* and *GoodnessOfFit* to check if

Gitte's cards fit the expected distribution. This implies that the null-hypothesis in this case is that Gitte's cards fits the expected distribution calculated by her friend.

```
URL <- "https://raw.githubusercontent.com/haghish/ST516/master/data/soccer.txt"
data <- read.table(URL, header = TRUE)
x<-data$Number
p<-data$expected
GOFit<-GoodnessOfFit(x,p)
chi.probability(GOFit,length(x)-1,1000)
```




```
## [1] 0.127
```

By looking at the output it can be seen that the expected distribution calculated by Gitte's friend fits the distribution of Gitte's cards better than the claim from the company. It can be seen from the p-value in this case that the null-hypothesis is not rejected.

Sheldon, Ross. 2013a. *Simulation*. Academic Press.

———. 2013b. *Simulation*. Academic Press.

———. 2013c. *Simulation*. Academic Press.

———. 2013d. *Simulation*. Academic Press.

n.d. http://www.uv.es/uriel/4%20Hypothesis%20testing%20in%20the%20multiple%20regression%20model.pdf.