Ex 3_a

Christoffer Mondrup Kramer

2023-05-19

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.2     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.2     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to be
come errors
```

Ex. 3a - Access normality

QQ-plot

The goal is to access normality of a single variable, here is how it is done.

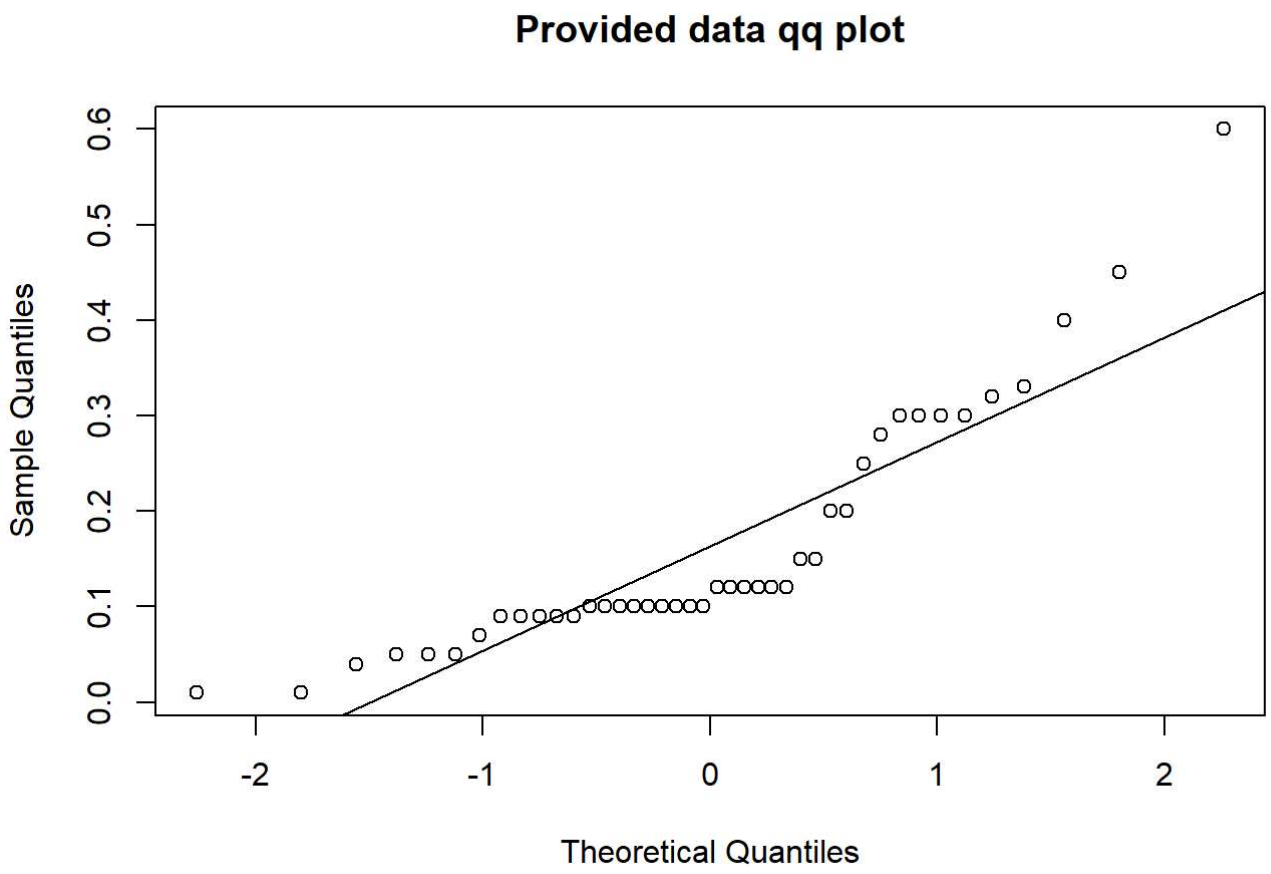
We start by reading the data frame:

```
df <- read.table("t4-5.dat", header = FALSE)
df
```

```
##      V1
## 1  0.30
## 2  0.09
## 3  0.30
## 4  0.10
## 5  0.10
## 6  0.12
## 7  0.09
## 8  0.10
## 9  0.09
## 10 0.10
## 11 0.07
## 12 0.05
## 13 0.01
## 14 0.45
## 15 0.12
## 16 0.20
## 17 0.04
## 18 0.10
## 19 0.01
## 20 0.60
## 21 0.12
## 22 0.10
## 23 0.05
## 24 0.05
## 25 0.15
## 26 0.30
## 27 0.15
## 28 0.09
## 29 0.09
## 30 0.28
## 31 0.10
## 32 0.10
## 33 0.10
## 34 0.30
## 35 0.12
## 36 0.25
## 37 0.20
## 38 0.40
## 39 0.33
## 40 0.32
## 41 0.12
## 42 0.12
```

We start out by using qqnorm to create the qqplot, and then we put up the qqline, to see how much the observed data differ from a theoretical normal distribution:

```
qqc <-qqnorm(df$V1, main = "Provided data qq plot")
qqline(df$V1)
```



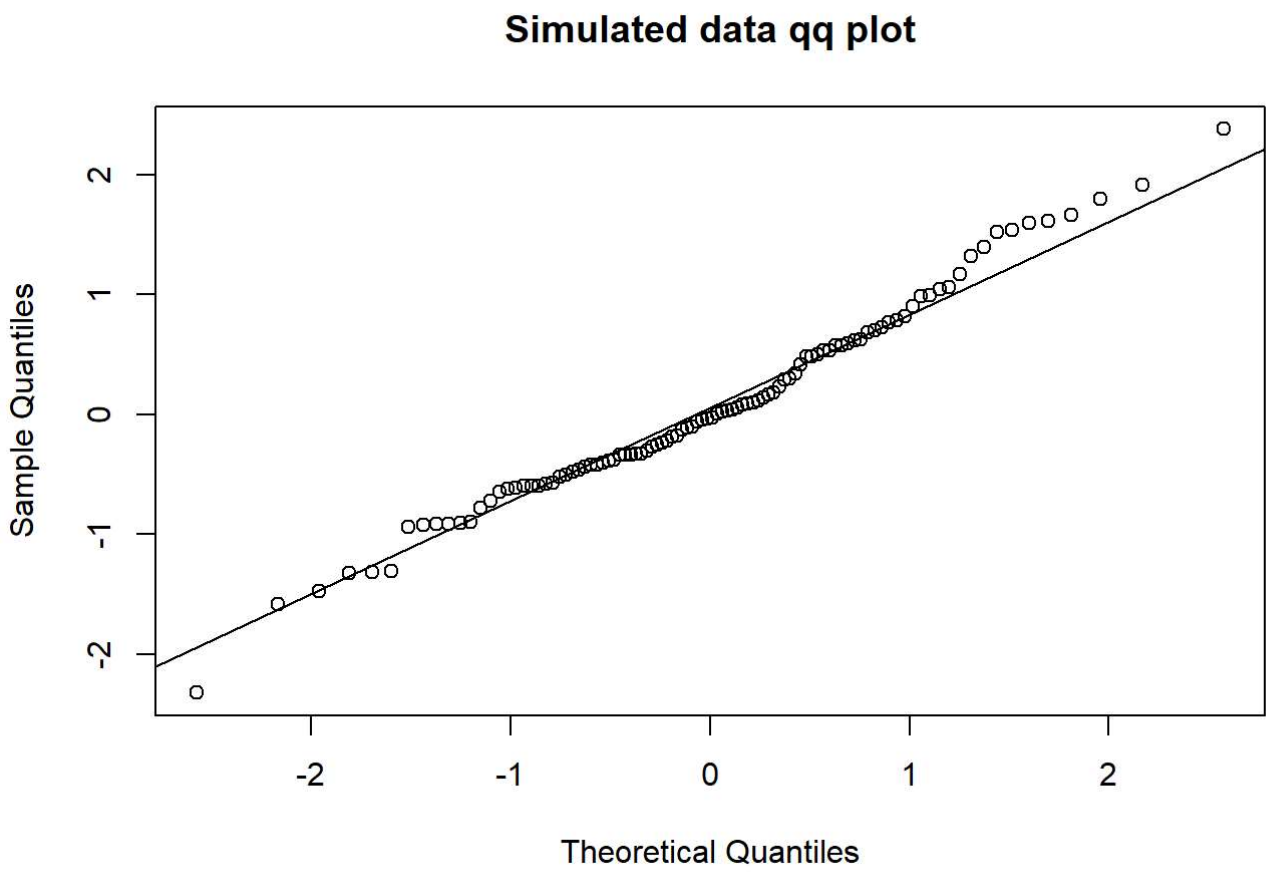
We then read the correlation between the theoretical points and the observed points. The closer this is to 1 the better.

```
cor(qqc$x, qqc$y)
```

```
## [1] 0.9090253
```

0.9 This might seem good, but let's compare it to a simulation:

```
    simu <- rnorm(100)
    qqs <- qqnorm(simu, main = "Simulated data qq plot")
    qqline(simu)
```



This align much better, which is reflected in the correlation:

```
cors <- cor(qqs$x, qqs$y)
cors
```

```
## [1] 0.9920936
```

It is 0.99 which is much stronger correlation

Hypothesis test

We have two hypothesis:

H0: Data IS normally distributed H1: Data is NOT normally distributed

For Hypothesis testing we will define a function, which will compare the correlation to the one at p. 181.

```
reject_hypothesis <- function(data_vector){
  # Create table from book
  n <- c(5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 75, 100, 150, 200, 300)
  one <- c(0.8299, 0.8801, 0.9126, 0.9269, 0.9410,
        0.9479, 0.9538, 0.9599, 0.9632, 0.9671,
        0.9695, 0.9720, 0.9771, 0.9822, 0.9879, 0.9905, 0.9935)

  five <- c(0.8788, 0.9198, 0.9389,
        0.9508, 0.9591, 0.9652,
        0.9682, 0.9726, 0.9749,
        0.9768, 0.9787, 0.9801,
        0.9838, 0.9873, 0.9913, 0.9931, 0.9953)

  ten <- c(0.9032, 0.9351, 0.9503,
        0.9604, 0.9665, 0.9715,
        0.9740, 0.9771, 0.9792,
        0.9809, 0.9822, 0.9836,
        0.9866, 0.9895, 0.9928, 0.9942, 0.9960)
  testing_tbl <- data.frame(
  n,
  one,
  five,
  ten
  )

  # Find out where to look in the table
  exact_sample_size <- FALSE # If the sample size fits perfectly with N in the table this wil
l be true
  sample_size <- length(data_vector)
  i <- 1
  prev_value = NaN
  for (n in testing_tbl$n){

    if (sample_size > testing_tbl$n[length(testing_tbl$n)]){
      i <- length(testing_tbl$n)
      break
    }

    if (n == sample_size){
      exact_sample_size <- TRUE
      break
    }

    else if ( n > sample_size & prev_value < sample_size){
      break
    }
    i <- i + 1
    prev_value <- n
  }

  # Return the intervals we need to look at:
  if (exact_sample_size) {
  print(testing_tbl[i, ])
  }
  else {
```

```
    print(testing_tbl[(i - 1) : i, ])
  }

}
```

We then look at the values we are interested in with our n.

```
reject_hypothesis(df$V1)
```

```
##    n    one   five    ten
## 8 40 0.9599 0.9726 0.9771
## 9 45 0.9632 0.9749 0.9792
```

As we can see our correlation is lower than any, so we have to reject the null hypothesis. THe current data set is NOT normally distributed.

Transforming data with box cox

Since the data is not normally distributed, we can use a data transformation to make i normal distributed. This is NOT a manipulation, but a transformation.

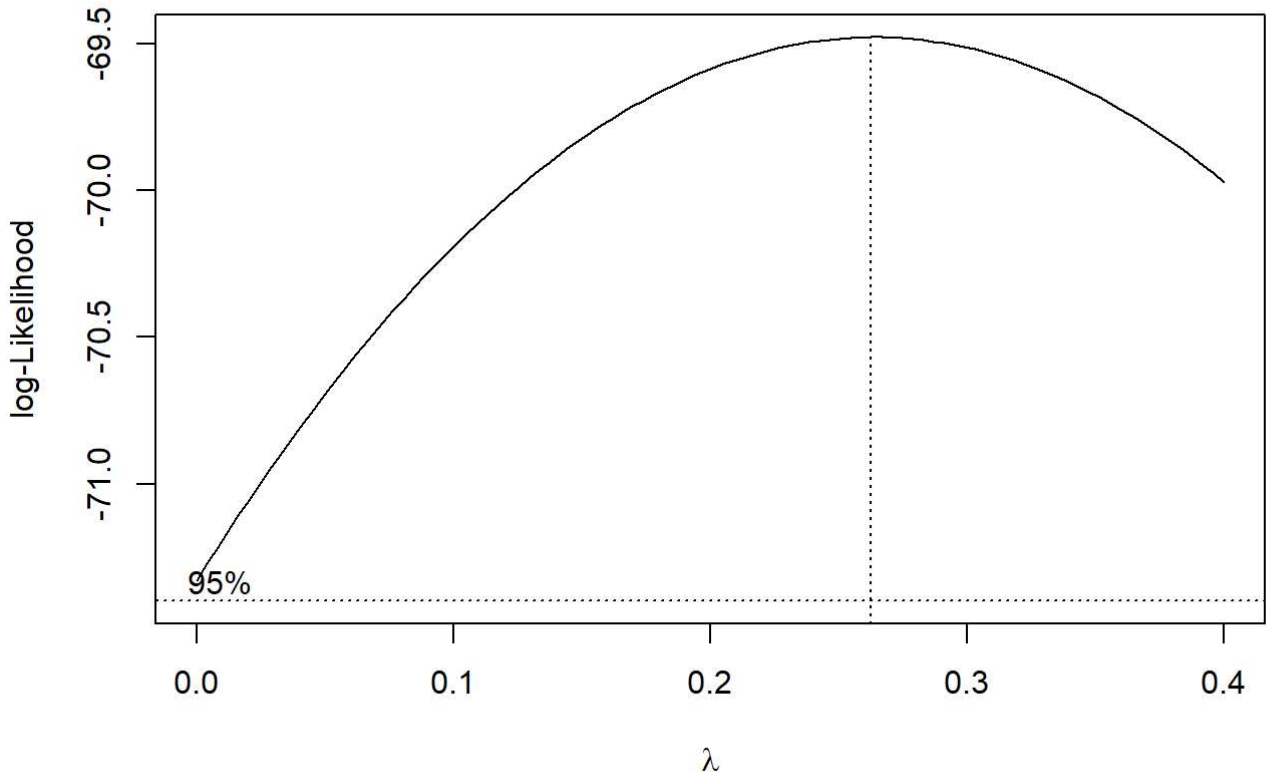
We first need to find the optimal value

```
library(MASS)
```

```
##
## Vedhæfter pakke: 'MASS'
```

```
## Det følgende objekt er maskeret fra 'package:dplyr':
##
##     select
```

```
# Make box cox plot on our data
df_box_cox<- boxcox(df$V1~1,lambda=seq(0, 0.5, 2/10))
```




```
# Get best lambda
max_lambda <- df_box_cox$x[which.max(df_box_cox$y)]
max_lambda # 0.2626263
```

```
## [1] 0.2626263
```

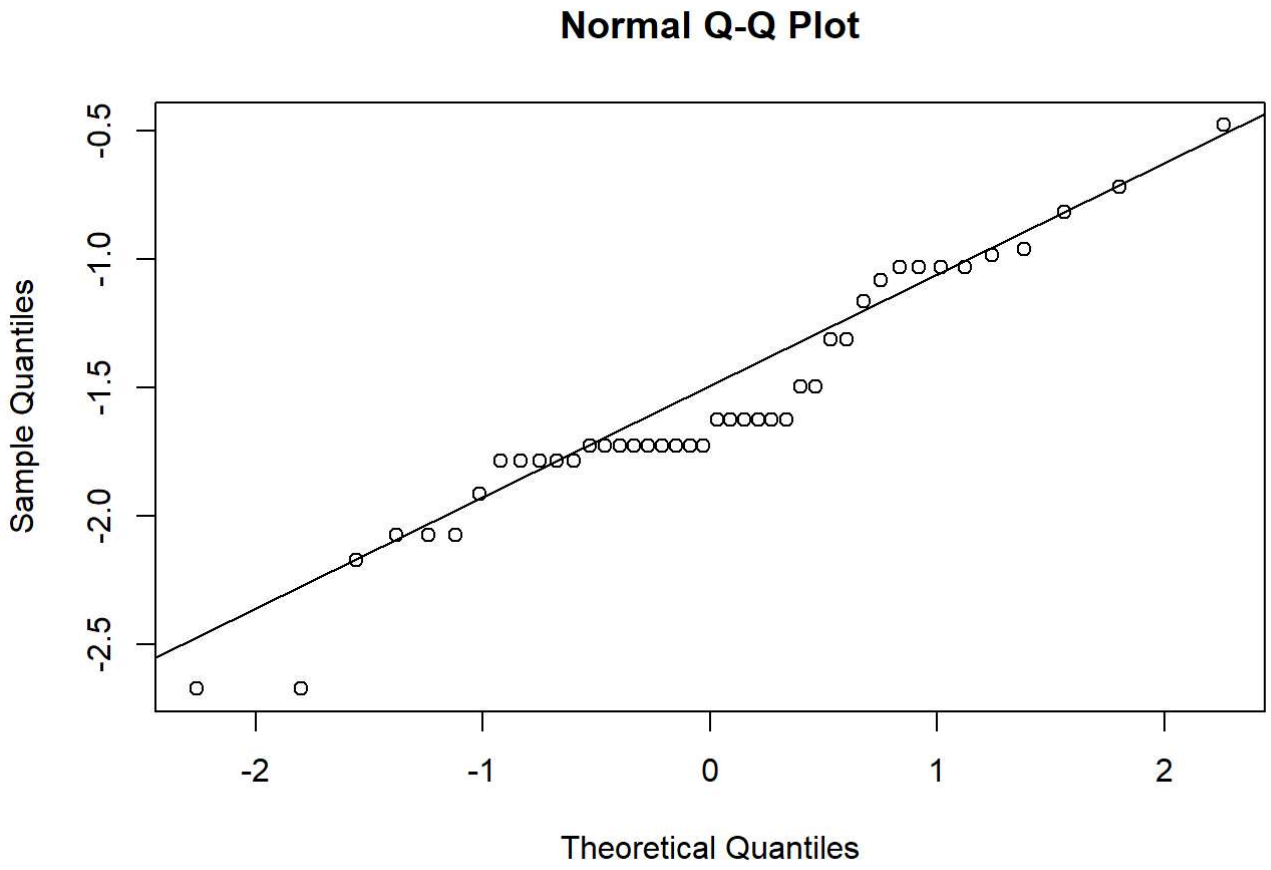
Now we just need to transform the data:

```
box_cox_transformation <- function(data_vector, lambda){
  if (lambda == 0){
    transformed_vector <- log(data_vector)
  }

  else {
  transformed_vector <- ((data_vector^lambda) - 1)/lambda
  }

  return(transformed_vector)
}

z_df <- box_cox_transformation(df$V1, max_lambda)
print(z_df)
```

```
##  [1] -1.0321992 -1.7845871 -1.0321992 -1.7278252 -1.7278252 -1.6258133
##  [7] -1.7845871 -1.7278252 -1.7845871 -1.7278252 -1.9138044 -2.0739793
## [13] -2.6716112 -0.7203389 -1.6258133 -1.3125578 -2.1726609 -1.7278252
## [19] -2.6716112 -0.4780427 -1.6258133 -1.7278252 -2.0739793 -2.0739793
## [25] -1.4941269 -1.0321992 -1.4941269 -1.7845871 -1.7845871 -1.0820364
## [31] -1.7278252 -1.7278252 -1.7278252 -1.0321992 -1.6258133 -1.1619650
## [37] -1.3125578 -0.8143778 -0.9618491 -0.9847550 -1.6258133 -1.6258133
```

We can then perform QQ plot:

```
qqz <- qqnorm(z_df)
qqline(z_df)
```



They seem closer but we need to know if they pass the hypothesis test:

```
reject_hypothesis(z_df)
```

```
##    n    one   five    ten
## 8 40 0.9599 0.9726 0.9771
## 9 45 0.9632 0.9749 0.9792
```

```
cor(qqz$x, qqz$y)
```

```
## [1] 0.9704366
```

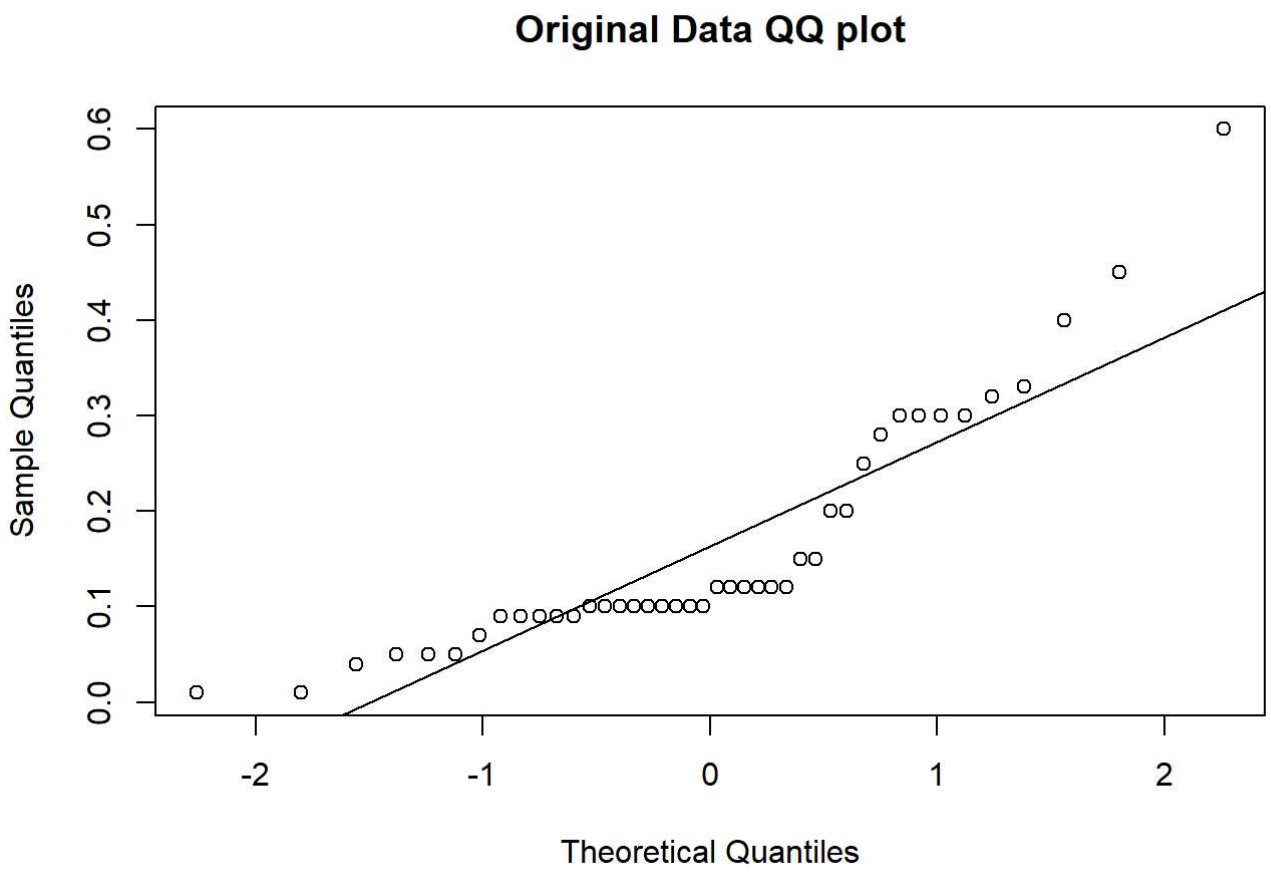
Full pipeline

```
box_cox_transformation <- function(data_vector, lambda){
  # Helper function
  if (lambda == 0){
    transformed_vector <- log(data_vector)
  }

  else {
  transformed_vector <- ((data_vector^lambda) - 1)/lambda
  }

  return(transformed_vector)
}


test_norm <- function(data_vector, signigicance = 0.05) {
  # You can chose the following significance levels
  # 0.01
  # 0.05
  # 0.10

  if (signigicance == 0.01){
    signigicance_col <- 2
  }

  else if (signigicance == 0.05){
    signigicance_col <- 3
  }

  else if (signigicance == 0.1){
    signigicance_col <- 4
  }


  # ----- QQ plot -----
  qq <- qqnorm(data_vector, main = "Original Data QQ plot")
  qqline(data_vector)

  print(paste("Length of data is ", length(data_vector)))
  # ----- Hypothesis test ------
  # Create Testing table
  n <- c(5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 75, 100, 150, 200, 300)
  one <- c(0.8299, 0.8801, 0.9126, 0.9269, 0.9410,
        0.9479, 0.9538, 0.9599, 0.9632, 0.9671,
        0.9695, 0.9720, 0.9771, 0.9822, 0.9879, 0.9905, 0.9935)

  five <- c(0.8788, 0.9198, 0.9389,
        0.9508, 0.9591, 0.9652,
        0.9682, 0.9726, 0.9749,
        0.9768, 0.9787, 0.9801,
        0.9838, 0.9873, 0.9913, 0.9931, 0.9953)

  ten <- c(0.9032, 0.9351, 0.9503,
        0.9604, 0.9665, 0.9715,
```

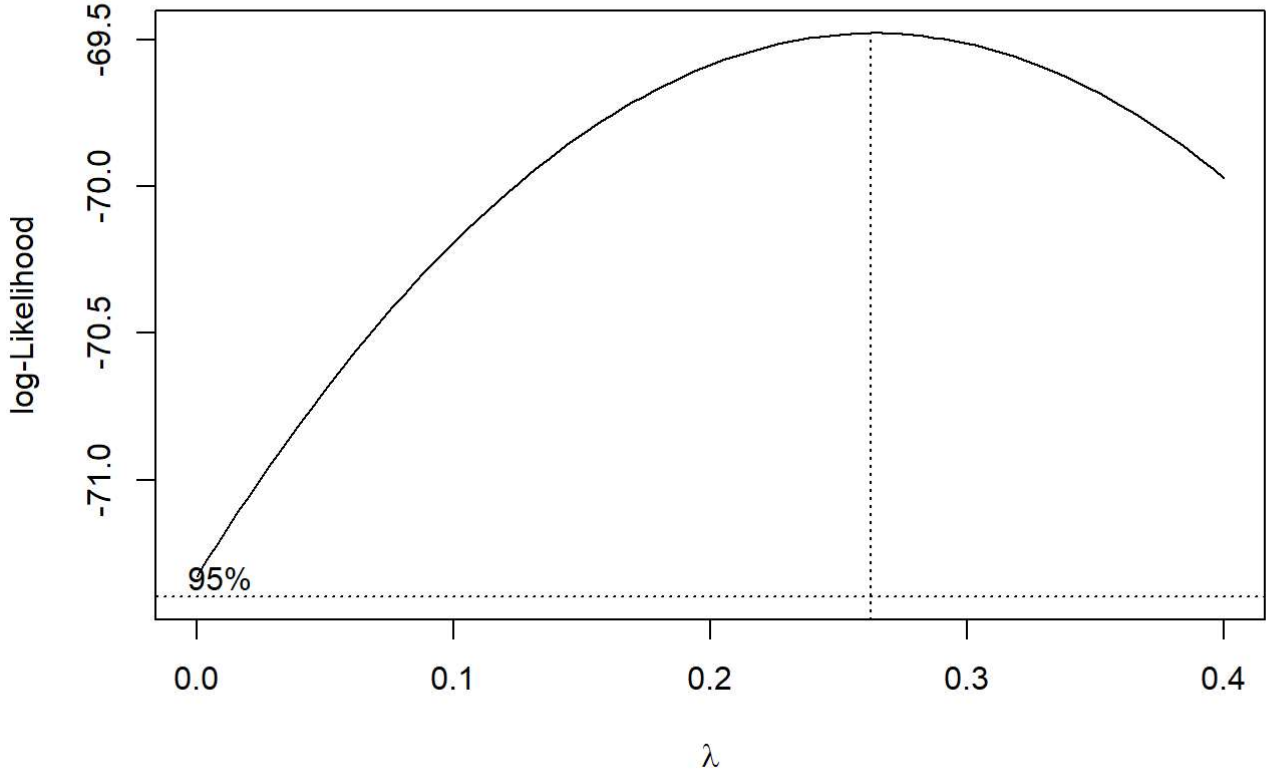
```
        0.9740, 0.9771, 0.9792,
        0.9809, 0.9822, 0.9836,
        0.9866, 0.9895, 0.9928, 0.9942, 0.9960)
  testing_tbl <- data.frame(
  n,
  one,
  five,
  ten
  )

  # Find index of testing (n)
  sample_size <- length(data_vector)
  i <- 1
  prev_value = NaN
  for (n in testing_tbl$n){

    if (sample_size > testing_tbl$n[length(testing_tbl$n)]){
      i <- length(testing_tbl$n)
      break
    }

    if (n == sample_size){
      exact_sample_size <- TRUE
      break
    }

    else if ( n > sample_size & prev_value < sample_size){
      break
    }
    i <- i + 1
    prev_value <- n
  }

  print(testing_tbl[(i - 1) : i, ])
  # ----- Normal -------
  cor_coef <- cor(qq$x, qq$y)
  normality = FALSE
  if (cor_coef > testing_tbl[i, signigicance_col]){
    print(paste("With a correlation coefficient of ", cor_coef, "The data is normal within si
gnificance levels of", signigicance))
    normality = TRUE
    return(normality)
  }


  # ----- Not normal -----
  else {



    print(paste("With a correlation coefficient of ", cor_coef,
                "The data is not normal within significance levels of", signigicance, "Transf
orming data ... "))

    # Transform data
    # Make box cox plot on our data
    df_box_cox<- boxcox(df$V1~1,lambda=seq(0, 0.5, 2/10))
```

```
    df_box_cox

    # Get best lambda
    max_lambda <- df_box_cox$x[which.max(df_box_cox$y)]
    print(paste("The best lambda is ", max_lambda))

    # Transform data
    z_vector <- box_cox_transformation(data_vector, max_lambda)

    # QQ plot on transformed data
    qqz <- qqnorm(z_vector, main = "Transformed Data QQ plot")
    qqline(z_vector)

    # Hypothesis test on transformed data
    z_cor_coef <- cor(qqz$x, qqz$y)
    if (z_cor_coef > testing_tbl[i, signigicance_col]){
      print(paste("With a correlation coefficient of ", z_cor_coef,
                  "The data is normal within significance levels of", signigicance,
                  "For the box cox transformed data"))
      normality = TRUE
      return(normality)
    }

    else {
        print(paste("With a correlation coefficient of ", z_cor_coef,
              "The data is normal within significance levels of", signigicance,
              "For the box cox transformed data"))
    }


  }




}
```

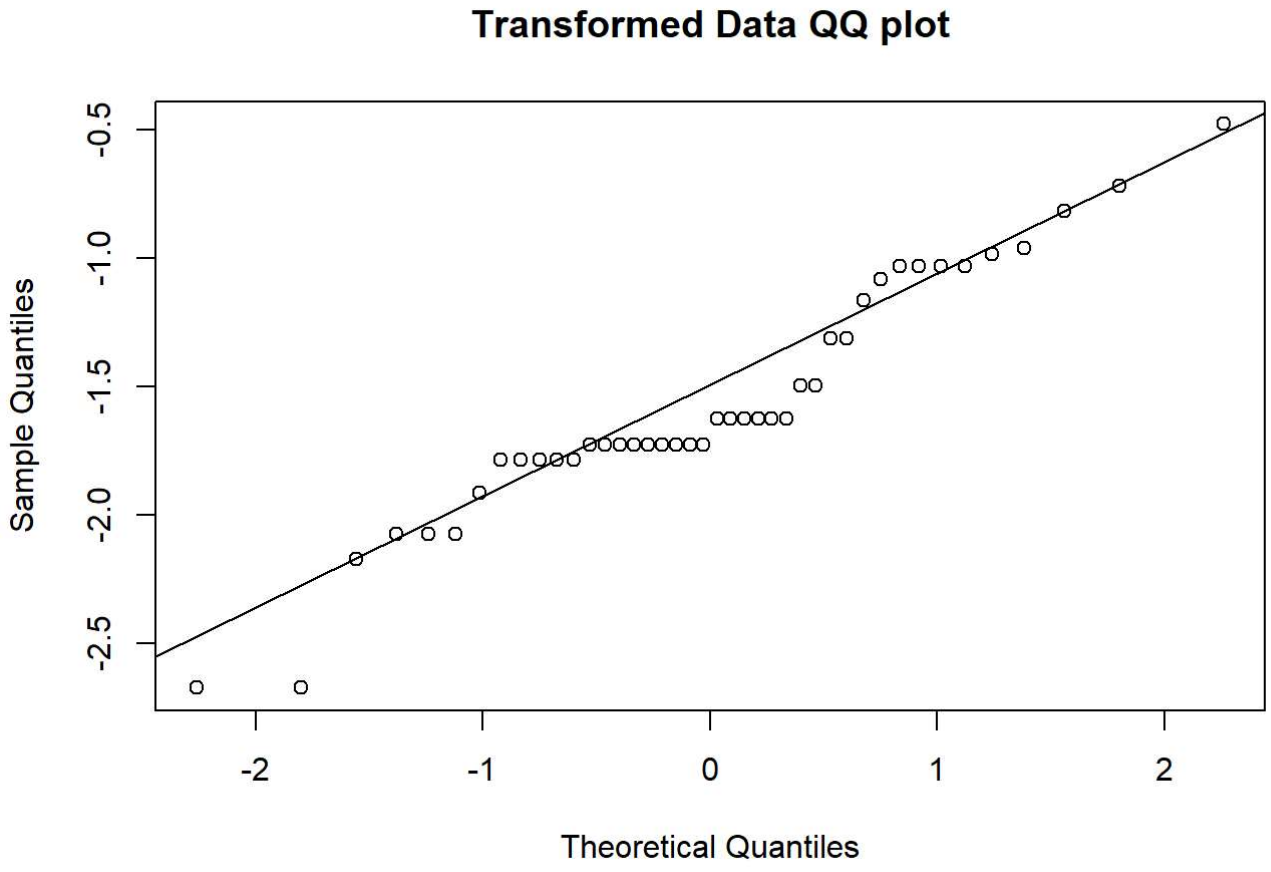
```
test_norm(df$V1, signigicance = 0.01)
```




```
## [1] "Length of data is  42"
##    n    one   five    ten
## 8 40 0.9599 0.9726 0.9771
## 9 45 0.9632 0.9749 0.9792
## [1] "With a correlation coefficient of  0.909025275553708 The data is not normal within si
gnificance levels of 0.01 Transforming data ... "
```



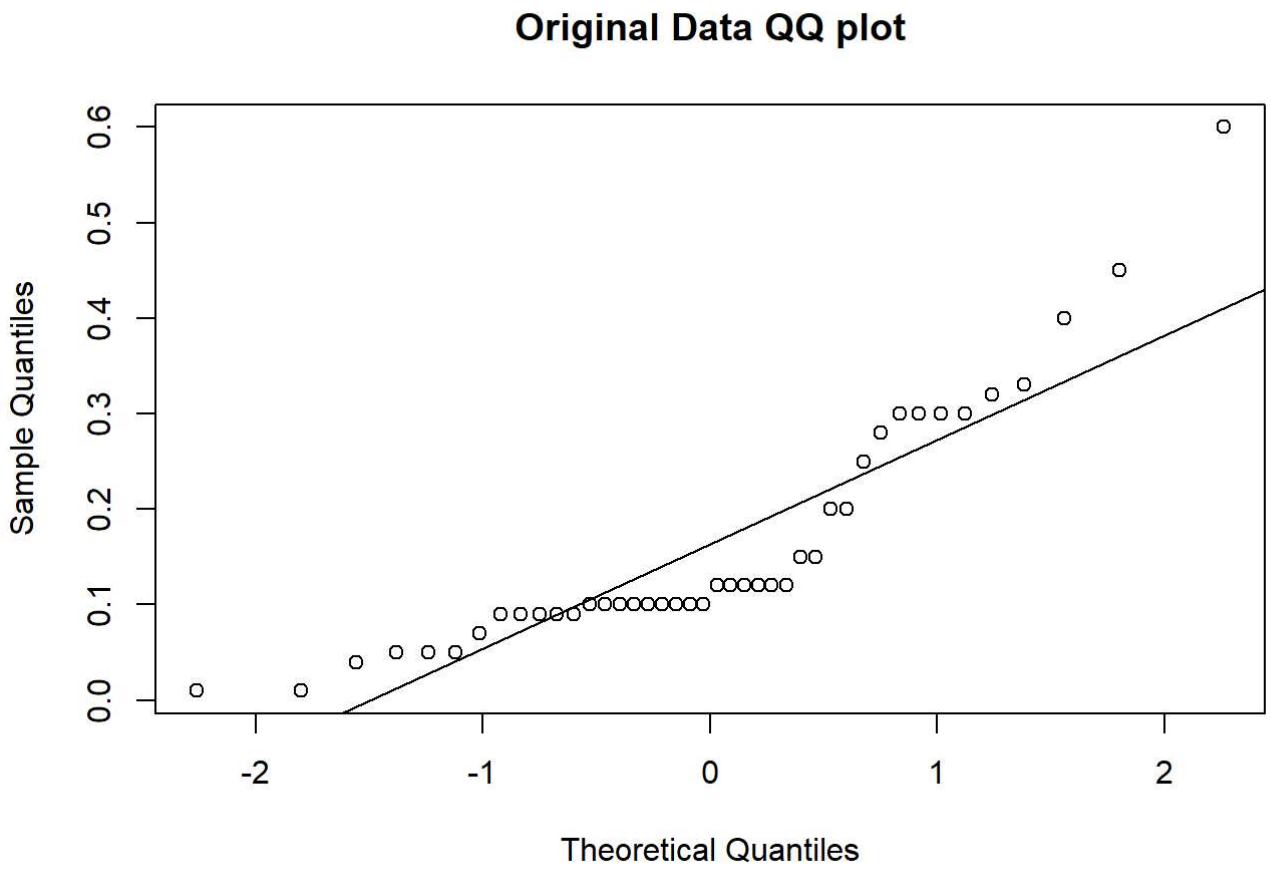

```
## [1] "The best lambda is  0.262626262626263"
```



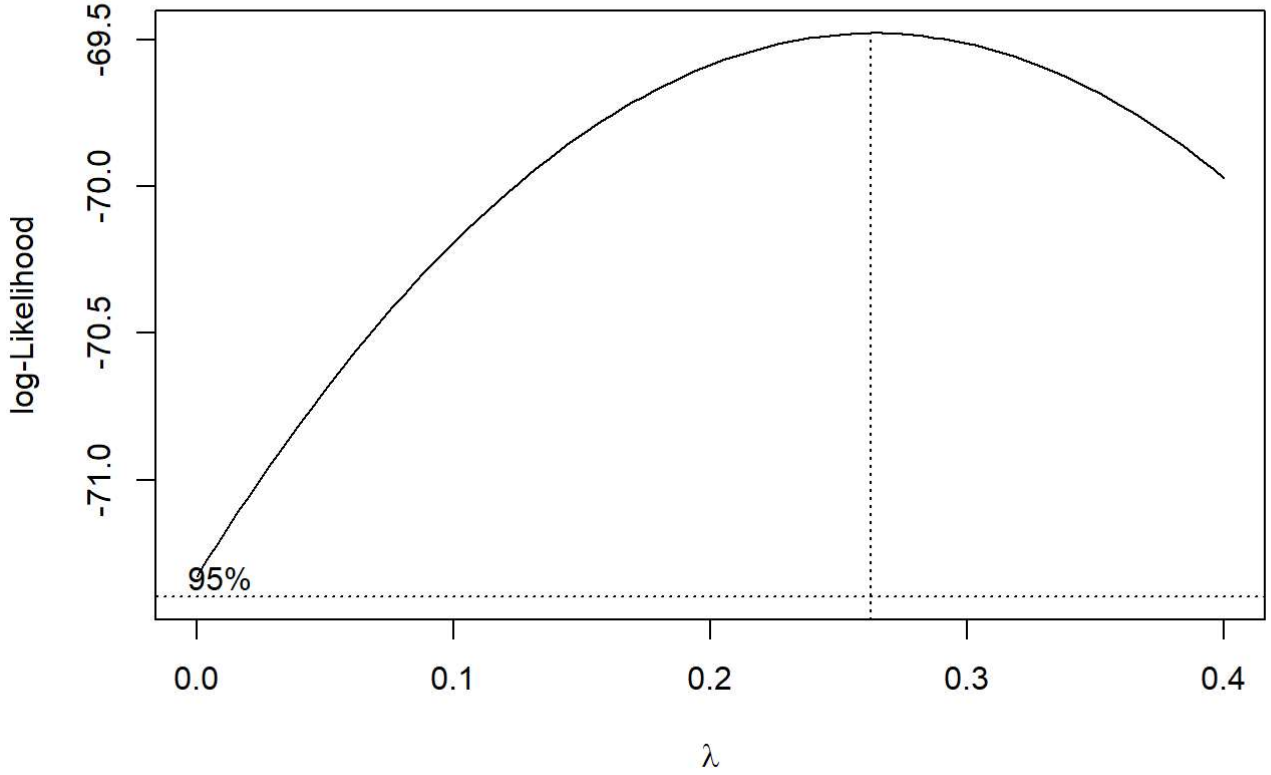
```
## [1] "With a correlation coefficient of  0.970436639721784 The data is normal within signif
icance levels of 0.01 For the box cox transformed data"
```

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

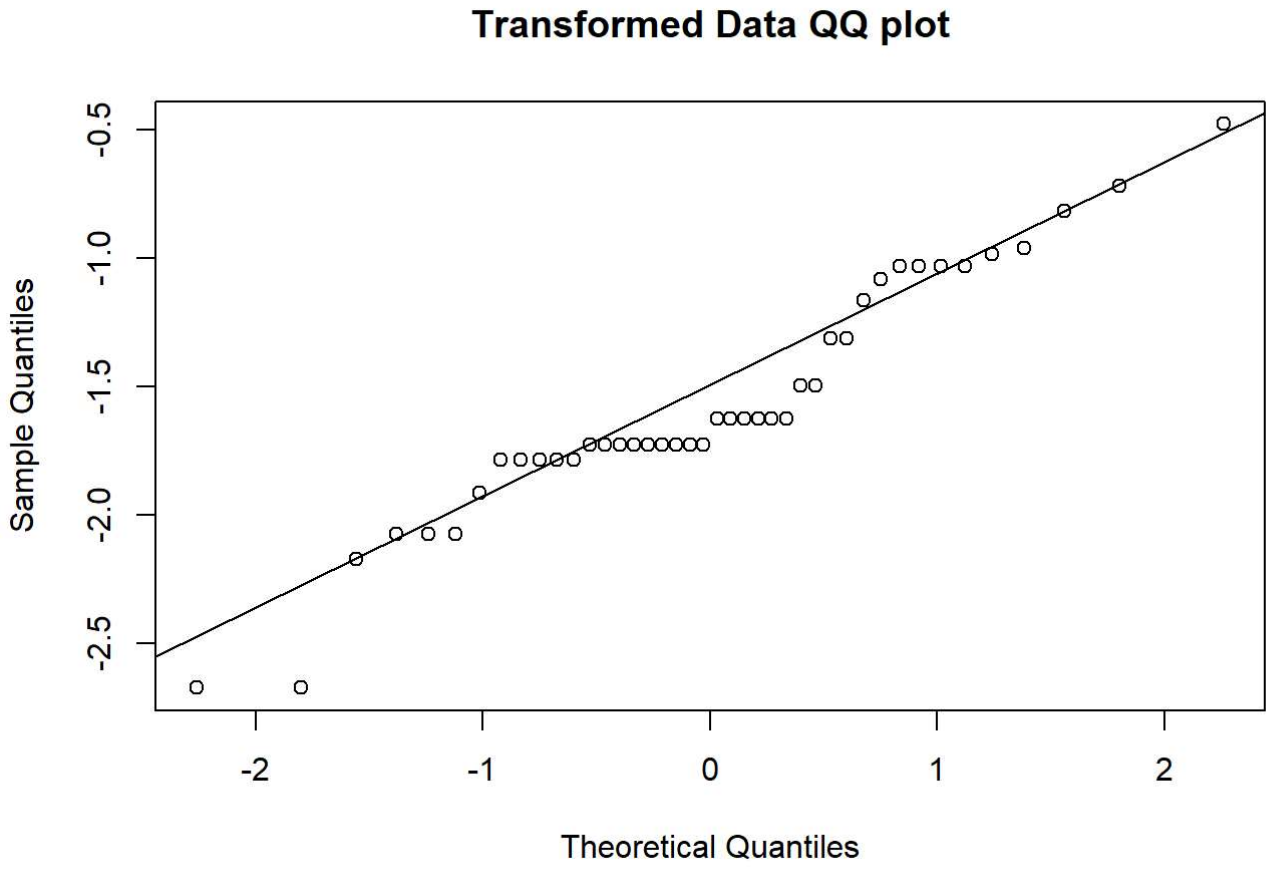
```
test_norm(df$V1, signigicance = 0.05)
```




```
## [1] "Length of data is  42"
##    n    one   five    ten
## 8 40 0.9599 0.9726 0.9771
## 9 45 0.9632 0.9749 0.9792
## [1] "With a correlation coefficient of  0.909025275553708 The data is not normal within si
gnificance levels of 0.05 Transforming data ... "
```

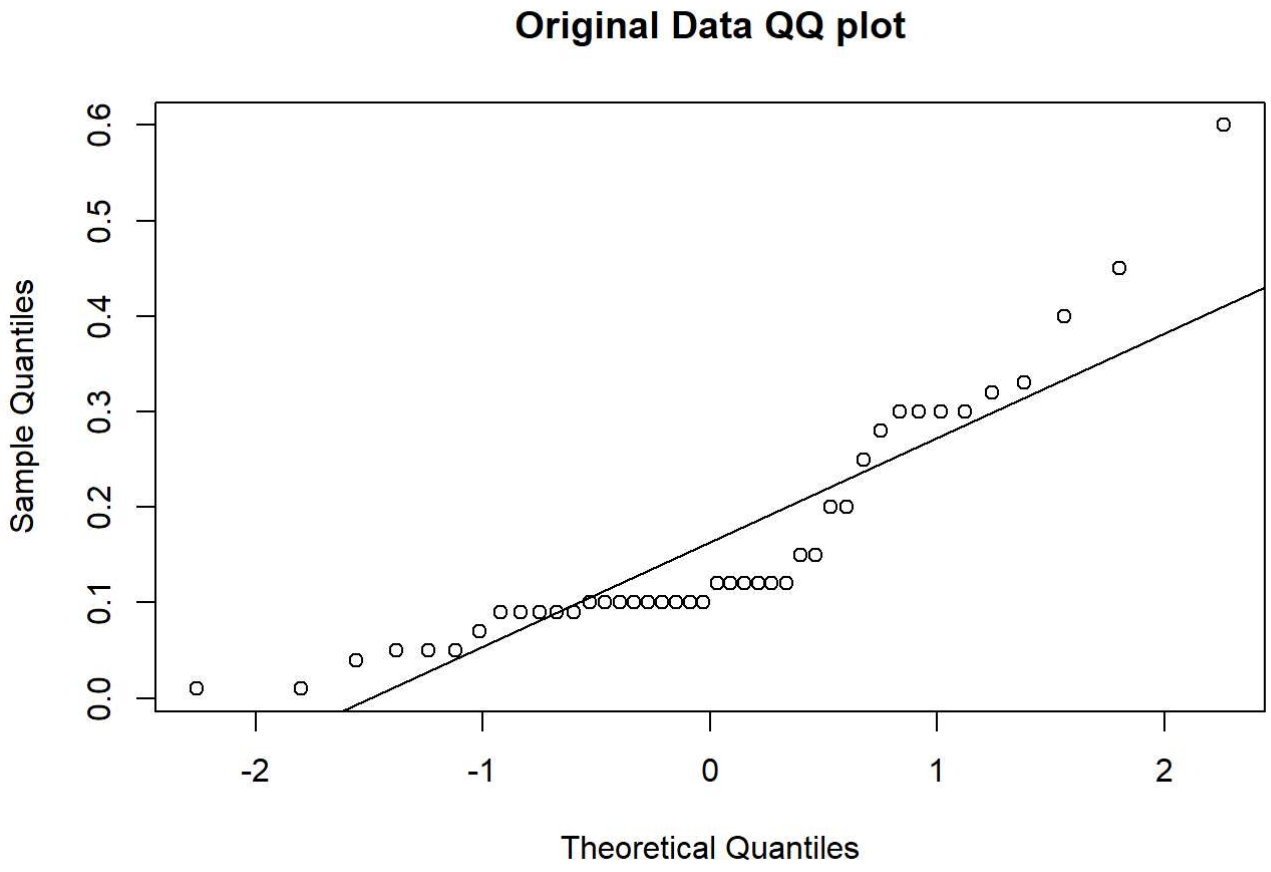



```
## [1] "The best lambda is  0.262626262626263"
```

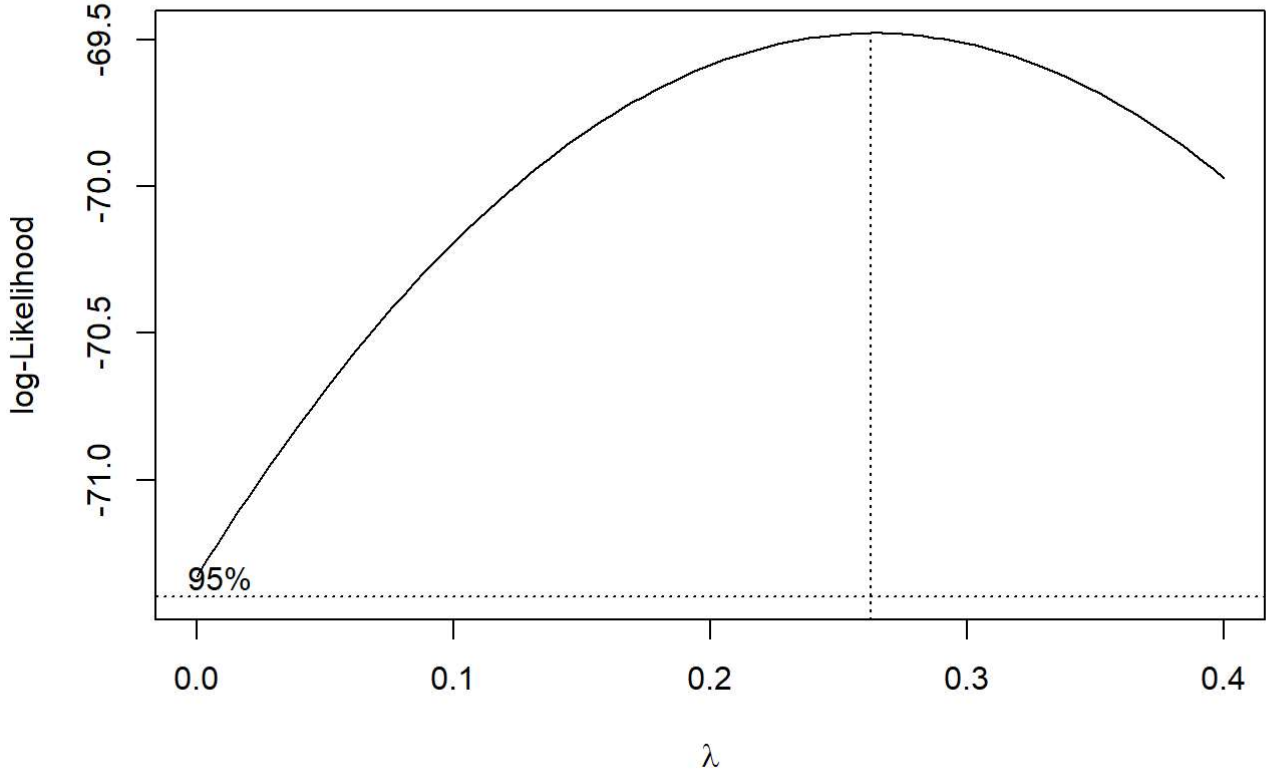


```
## [1] "With a correlation coefficient of  0.970436639721784 The data is normal within signif
icance levels of 0.05 For the box cox transformed data"
```

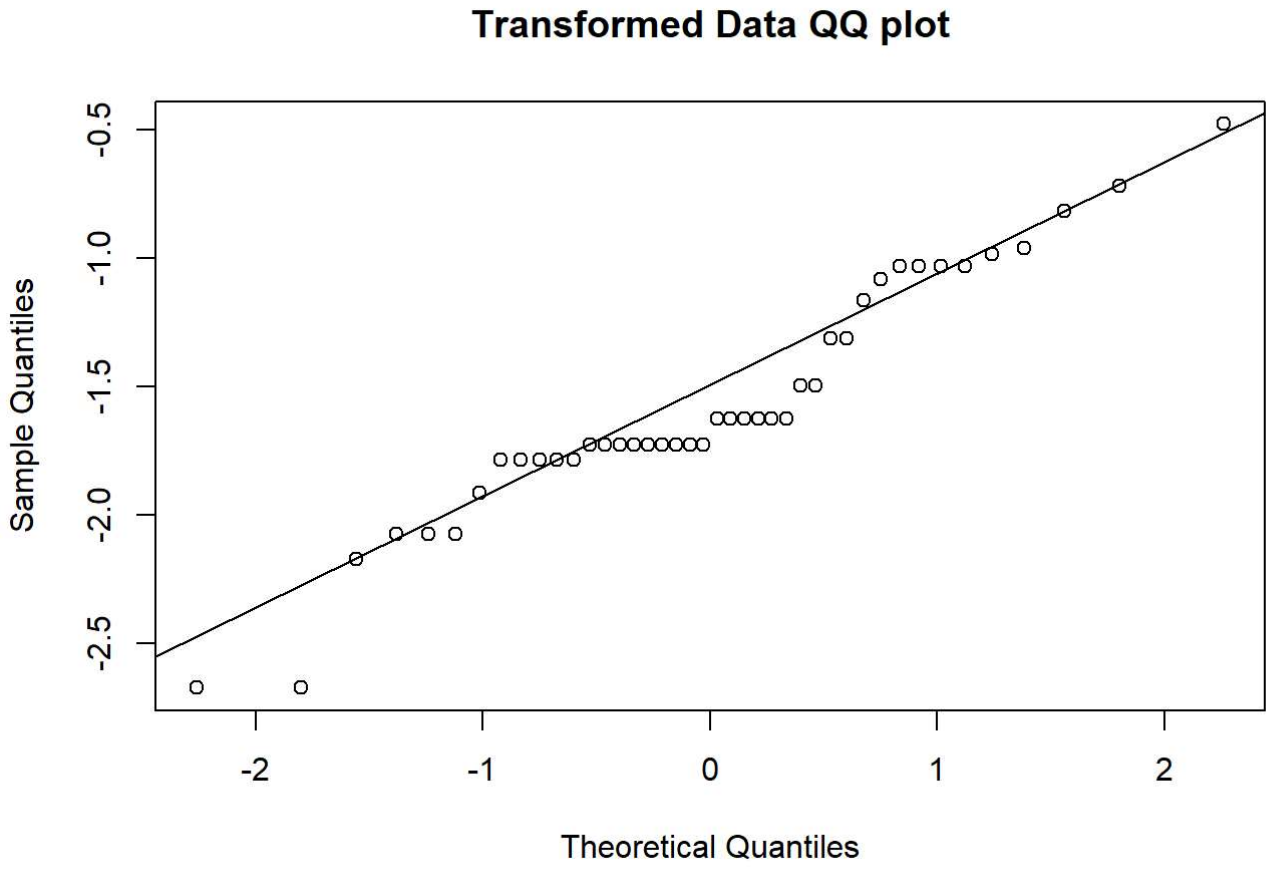
```
test_norm(df$V1, signigicance = 0.10)
```




```
## [1] "Length of data is  42"
##    n    one   five    ten
## 8 40 0.9599 0.9726 0.9771
## 9 45 0.9632 0.9749 0.9792
## [1] "With a correlation coefficient of  0.909025275553708 The data is not normal within si
gnificance levels of 0.1 Transforming data ... "
```



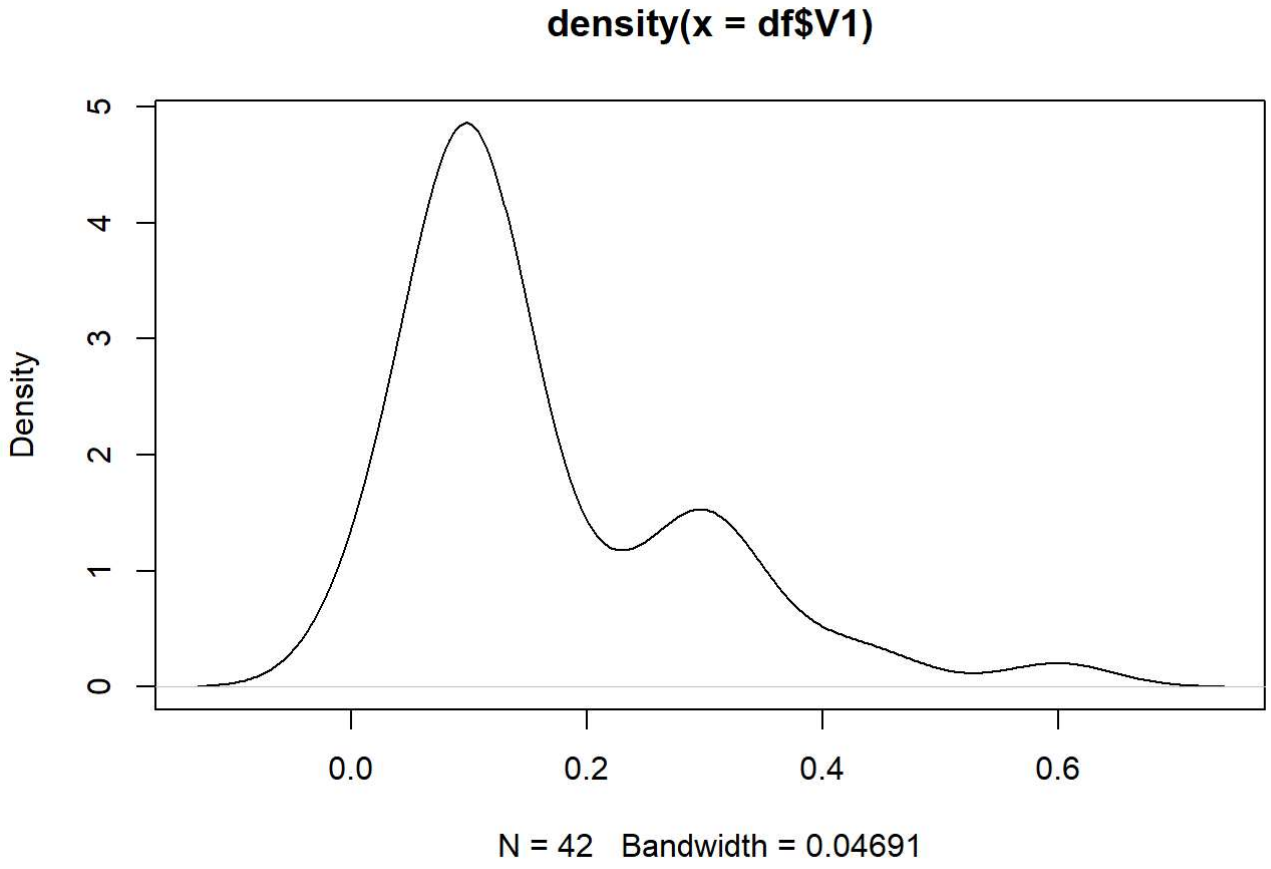

```
## [1] "The best lambda is  0.262626262626263"
```



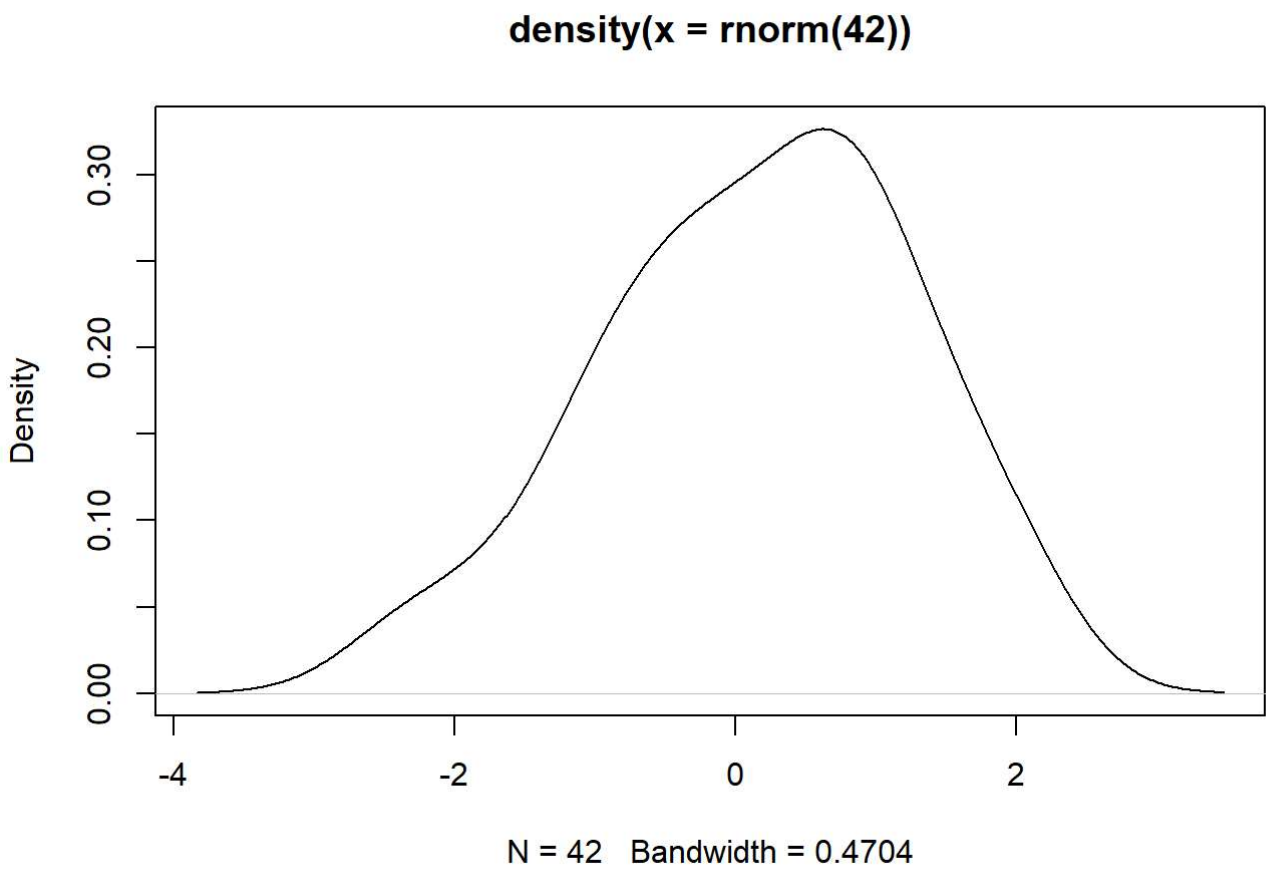
```
## [1] "With a correlation coefficient of  0.970436639721784 The data is normal within signif
icance levels of 0.1 For the box cox transformed data"
```

Other

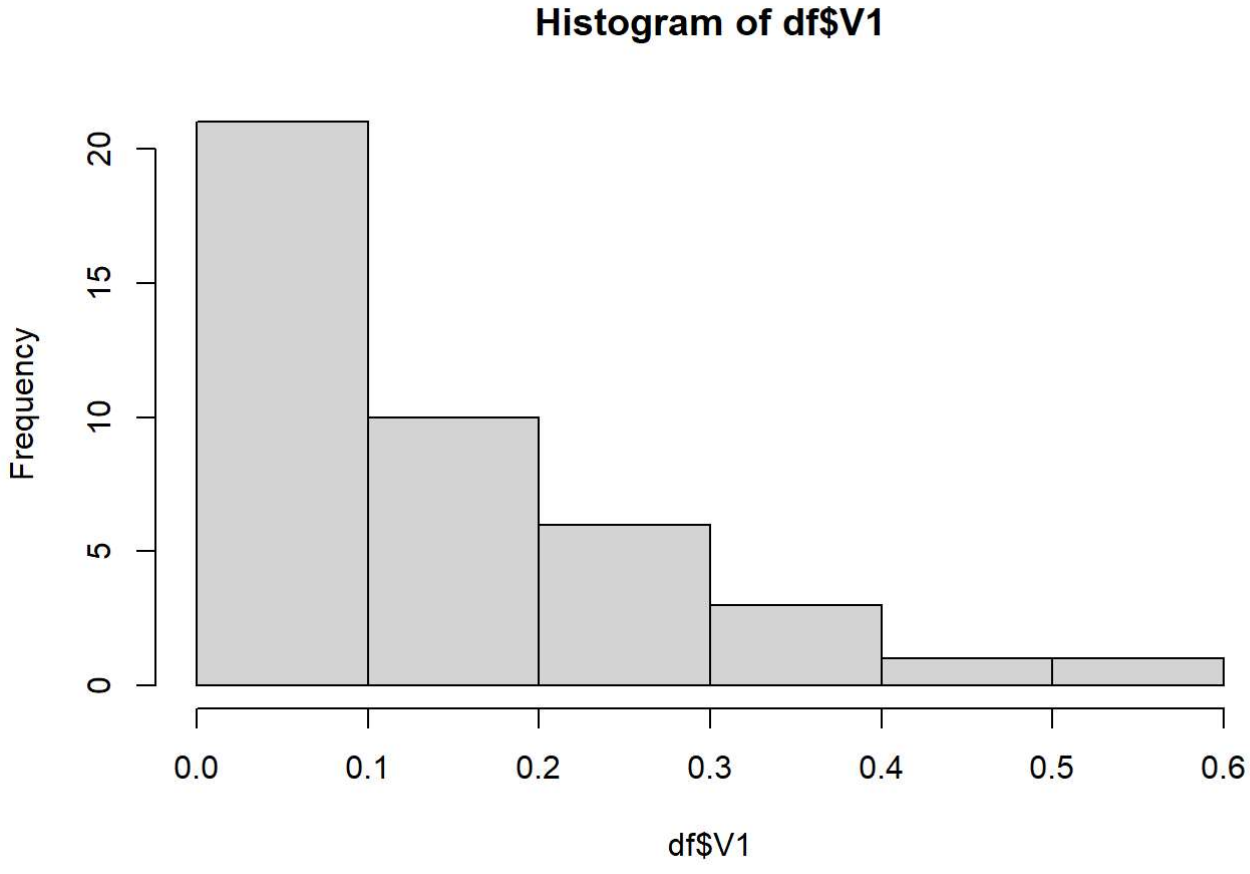
```
# Our data
d <- density(df$V1) # PDC
plot(d)
```



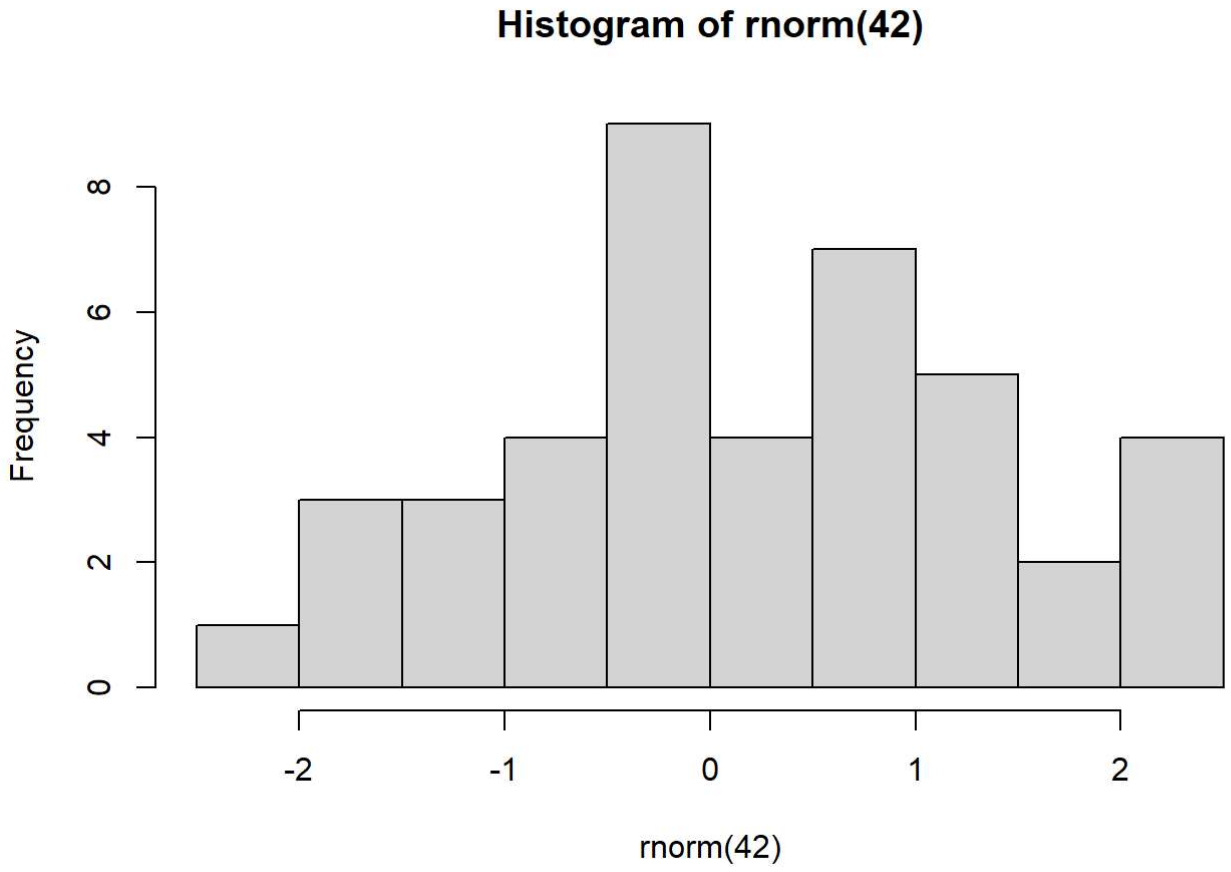

```
# Simulated
simu <- density(rnorm(42))
plot(simu)
```


density(x = rnorm(42))
Density
0.00
0.10
0.20
0.30
-4
-2
0
2
N = 42 Bandwidth = 0.4704

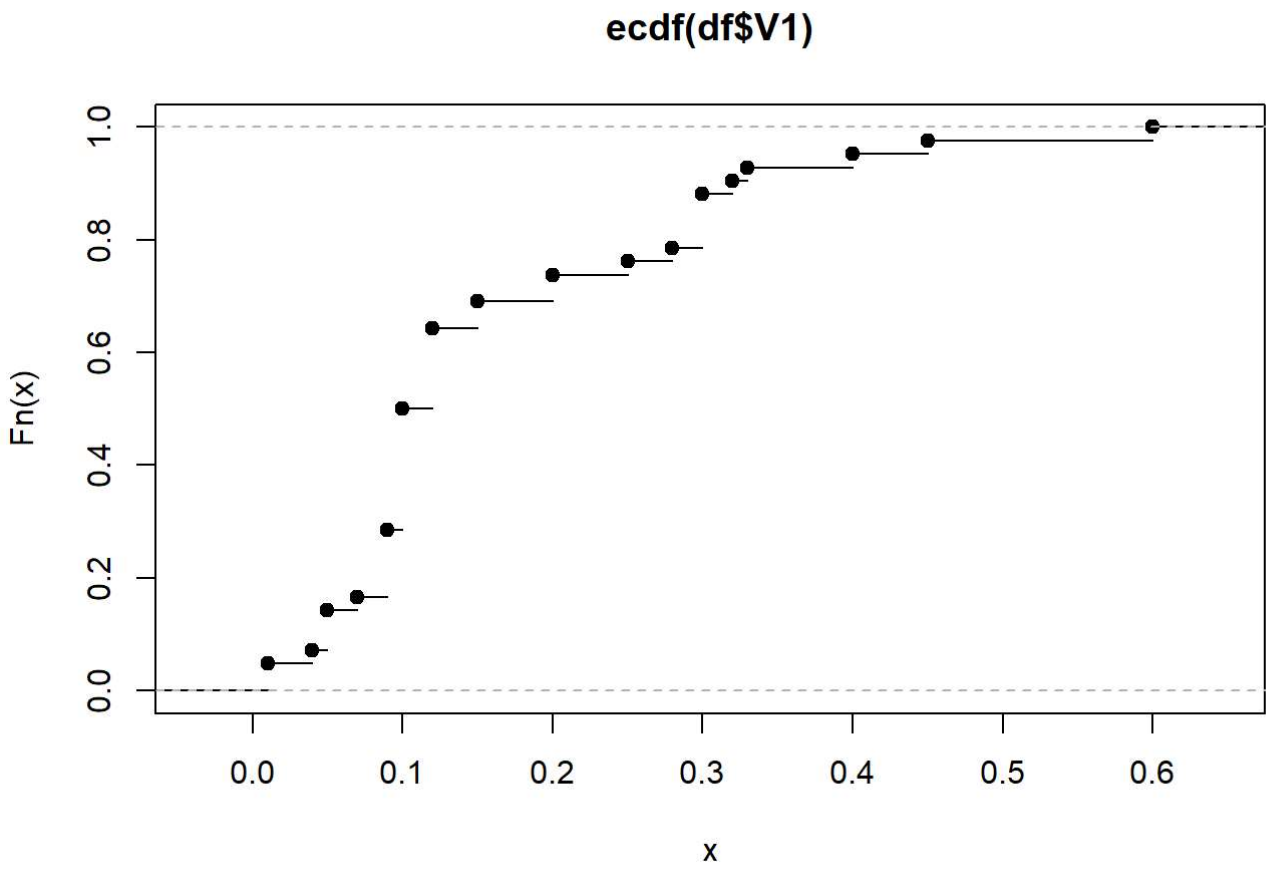
```
hist(df$V1)
```


Histogram of df$V1
Frequency
0
5
10
15
20
0.0
0.1
0.2
0.3
0.4
0.5
0.6
df$V1

```
hist(rnorm(42))
```




```
cdf <- ecdf(df$V1) # Cumulative density function
plot( cdf )
```




```
simu_cdf <- ecdf(rnorm(42))
plot(simu_cdf)
```