Lab1 : Density estimation

AMA-G5: Gerard Gómez, Rduio Fida, Cecilia Pérez

2023-09-20

Question 1

Relation between the histogram estimator

$$\hat{f}_{hist}(x)$$

and its leave one out version

$$\hat{f}_{hist,(-i)}(x)$$

$$\hat{f}_{hist,(-i)}(x) = \frac{n}{n-1}\hat{f}_{hist}(x) - \frac{1}{(n-1)b}$$

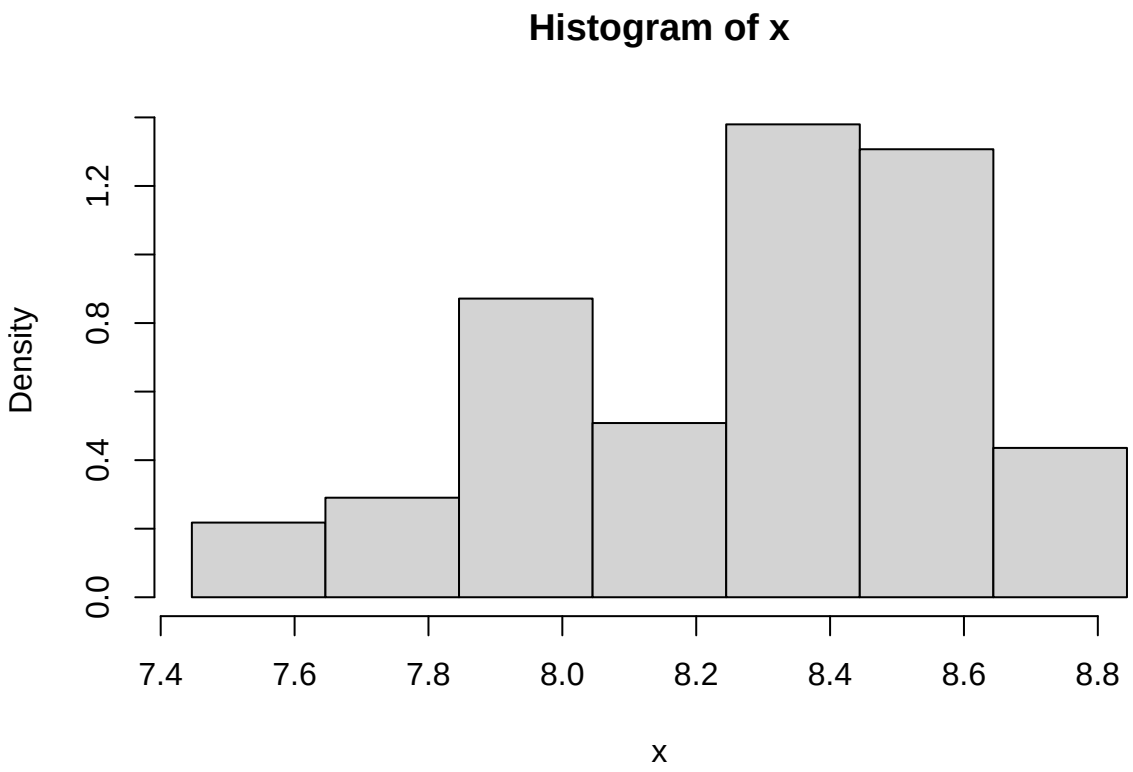
Question 2

The data is given by the dataframe cdrate.dat.

```
cdrate.df <-read.table("cdrate.dat")
head(cdrate.df)
```

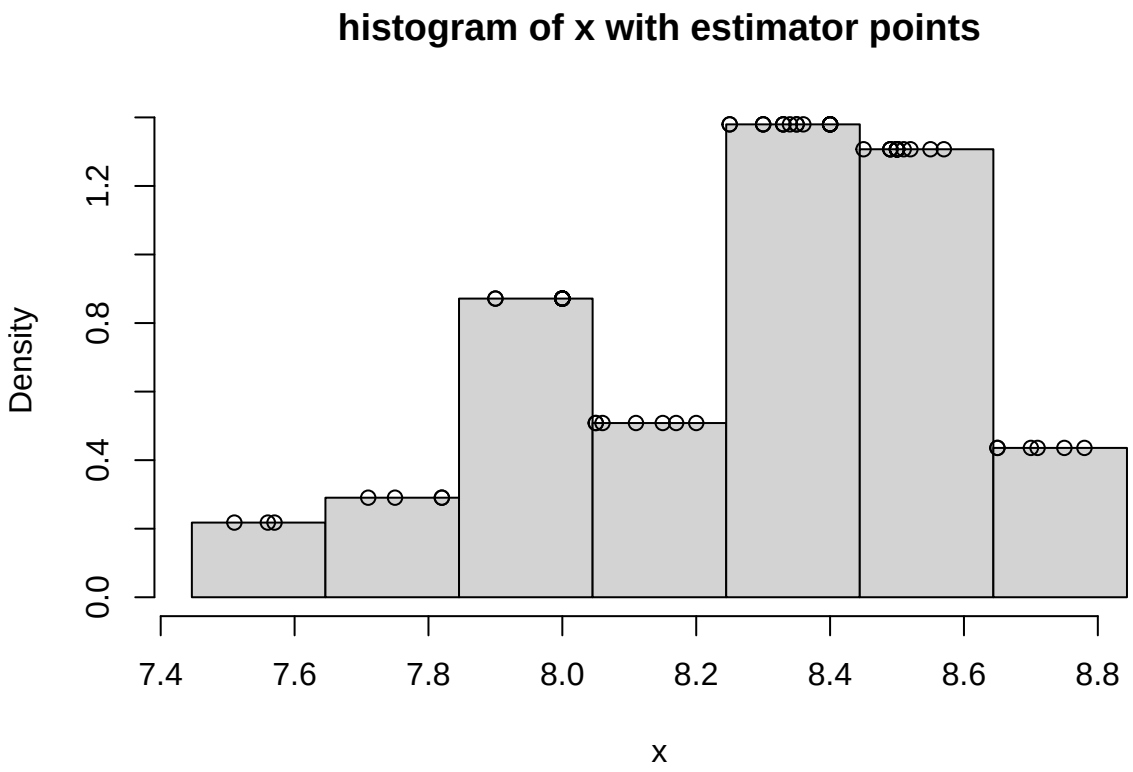
```
##     V1 V2
## 1 7.56  0
## 2 7.57  0
## 3 7.71  0
## 4 7.82  0
## 5 7.82  0
## 6 7.90  0
```

Once defined the minimum and maximum values for the dataset and number of bins for the histogram, a histogram(hx) is created, where values 8.4-8.6 of x have a higher density in the dataset.



Hx_f is the step function that allows to evaluate the histogram density at any point through the dataset. Points are added on the histogram to represent the histogram estimator values of x.

```
# histogram function
hx_f = stepfun(hx$breaks,c(0,hx$density,0))
y = hx_f(x)

hist(x,breaks=seq(A,Z,length=nbr+1),freq=F,main="histogram of x with estimator points")
points(x,y)
```



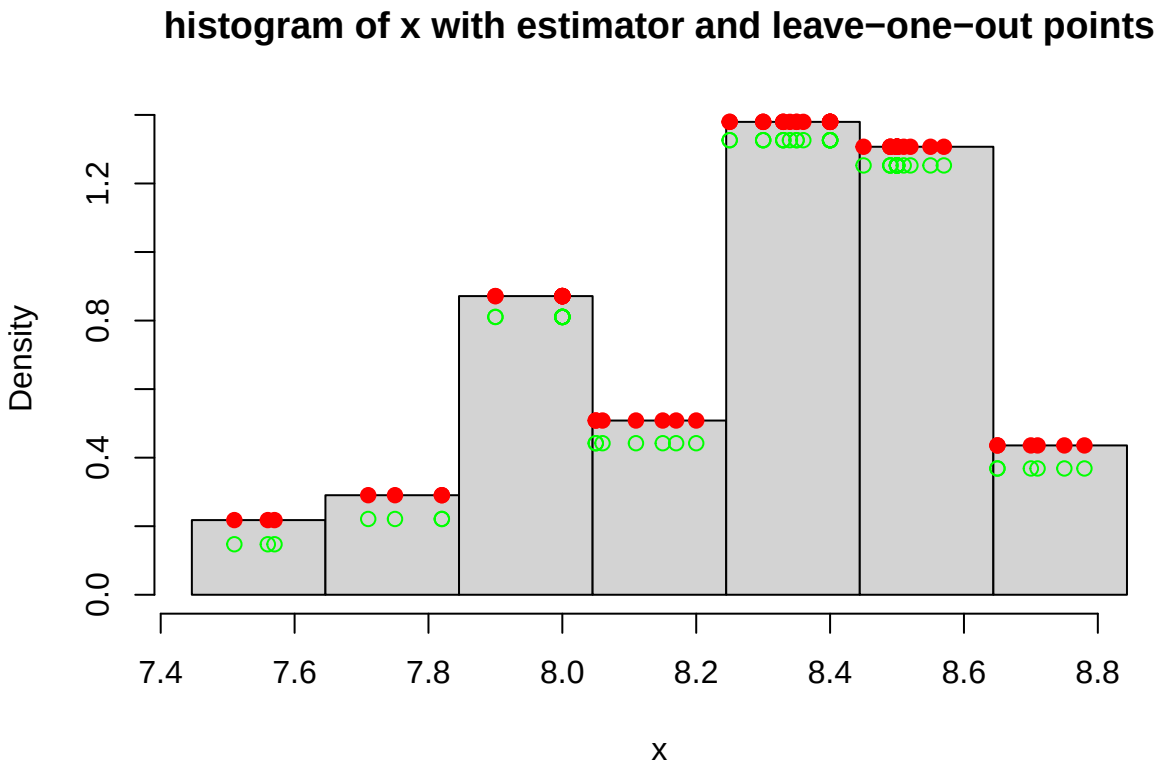
Question 3

```
fhist_loo = function(x,hist,f_h)
{
  # Computes the leave-one-out histogram estimator from data
  # Input :
  #   x : points
  #   hist : histogram object
  #   f_h : histogram estimator function
  # Output :
  #   f_i : Leave one out histogram estimator for the points of x

  b <- hist$breaks[2]-hist$breaks[1]
  n <- length(x)
  f_i = 1:n

  for (i in 1:n)
  {
    f_i[i] = n/(n-1)*f_h(x[i])-(1/((n-1)*b))
  }

  return(f_i)
}
```

```
y2 = fhist_loo(x,hx,hx_f)
```

The following histogram visualizes two density estimators for x. The red dots represent the histogram's density estimator and the green dots represent the leave-one-out histogram's density estimator. It can be observed that the density estimated is lower when individual points are excluded.

```
# Plots : histogram, histogram estimator and leave-one-out histogram estimator
hist(x,breaks=seq(A,Z,length=nbr+1),freq=F, main="histogram of x with estimator and leave-one-out points
points(x,y,pch=19,col="red")
points(x,y2,col="green")
legend(100,100,legend=c('f_hi','f_hi_loo'),col=c('red','green'),lty=1:2,cex=0.8)
```



Question 4

The leave-one-out log likelihood value for the histogram estimator with $nbr = 7$ is -16.58432. the fact that this value is negative can lead to conclude that the model's fit to the data is not good giving the performance of a histogram estimator of 7 bins

```
log_likelihood = function(x,hist,f_h)
{
  # Returns the leave-one-out log likelihood functions for the points x if the leave-one-out is >0. Els
  # Input :
```

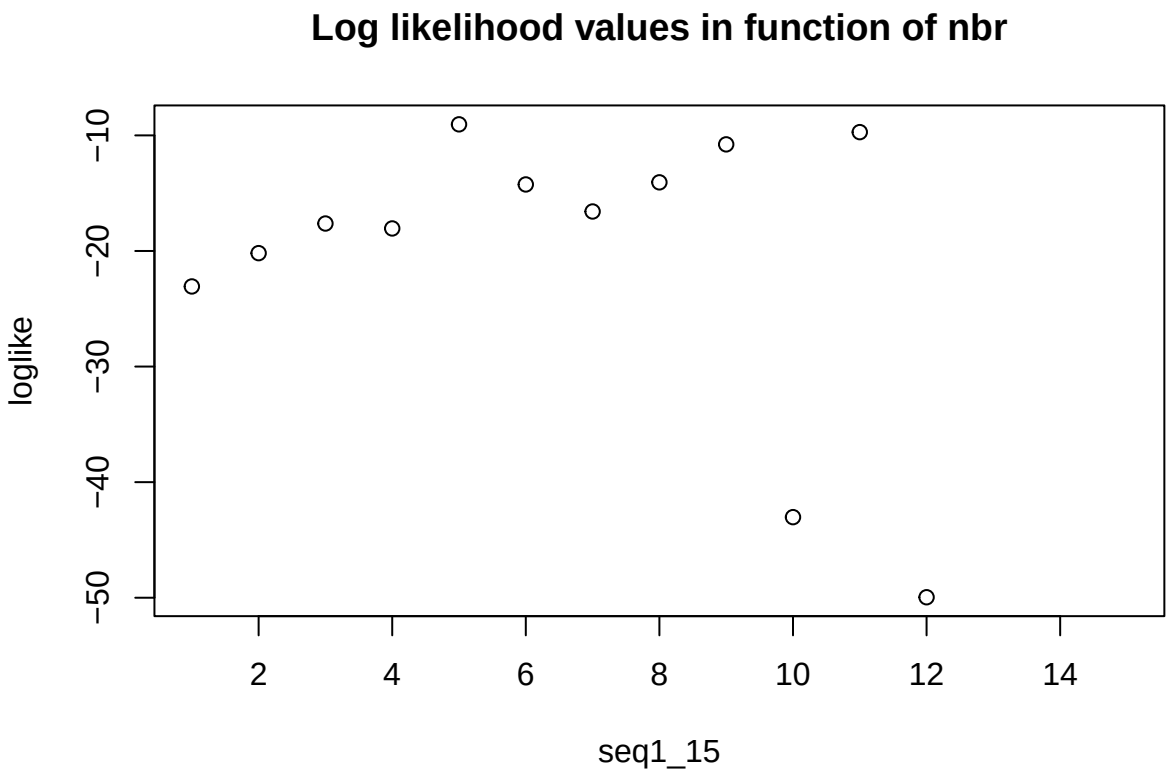
```
  #   x : points
  #   hist : histogram object
  #   f_h : histogram estimator function
  # Output :
  #   leave-one-out log likelihood

  loo = fhist_loo(x,hist,f_h)
  uzero = loo[loo<0]

  if(length((uzero))==0)
  {
    return(sum(log(loo)))
  }
  else{
    return(-Inf)
  }
}

log_likelihood(x,hx,hx_f)
```

```
## [1] -16.58432
```

Question 5

The following figure shows the values of leave-one-out Cross Validation (looCV) against nbr. We can see on it that the optimal value is $nbr = 5$ out of the sequence between 1 and 15. in addition, a plot of the log-likelihood values for each nbr value of the sequence has been shown in order to back up the result of the optimal value of nbr.

```
# Computing the leave-one-out log likelihood values with variating nbr
loglike <- c()
seq1_15<-seq(1,15)
A = min(x) - .05*diff(range(x))
Z = max(x) + .05*diff(range(x))

for (nbr in seq1_15)
{
  hx = hist(x,breaks=seq(A,Z,length=nbr+1),freq=F,plot=F)
  hx_f = stepfun(hx$breaks,c(0,hx$density,0))
  log_like <- log_likelihood(x,hx,hx_f)
  loglike <- append(loglike,log_like)
}

plot(seq1_15,loglike,main="Log likelihood values in function of nbr")
```

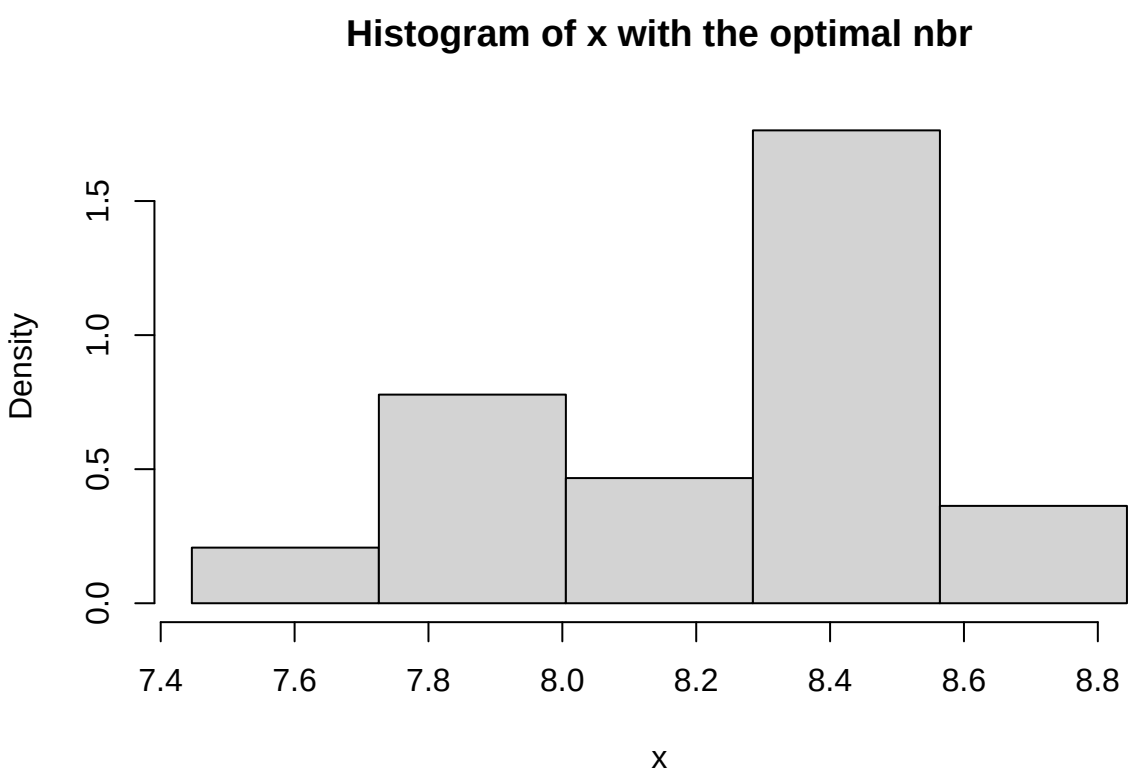



```
nbr_opt = which(loglike==max(loglike))
print(paste0("The optimal nbr is ",nbr_opt))
```

```
## [1] "The optimal nbr is 5"
```

Next, is the histogram of the data's density according to the optimal nbr found.

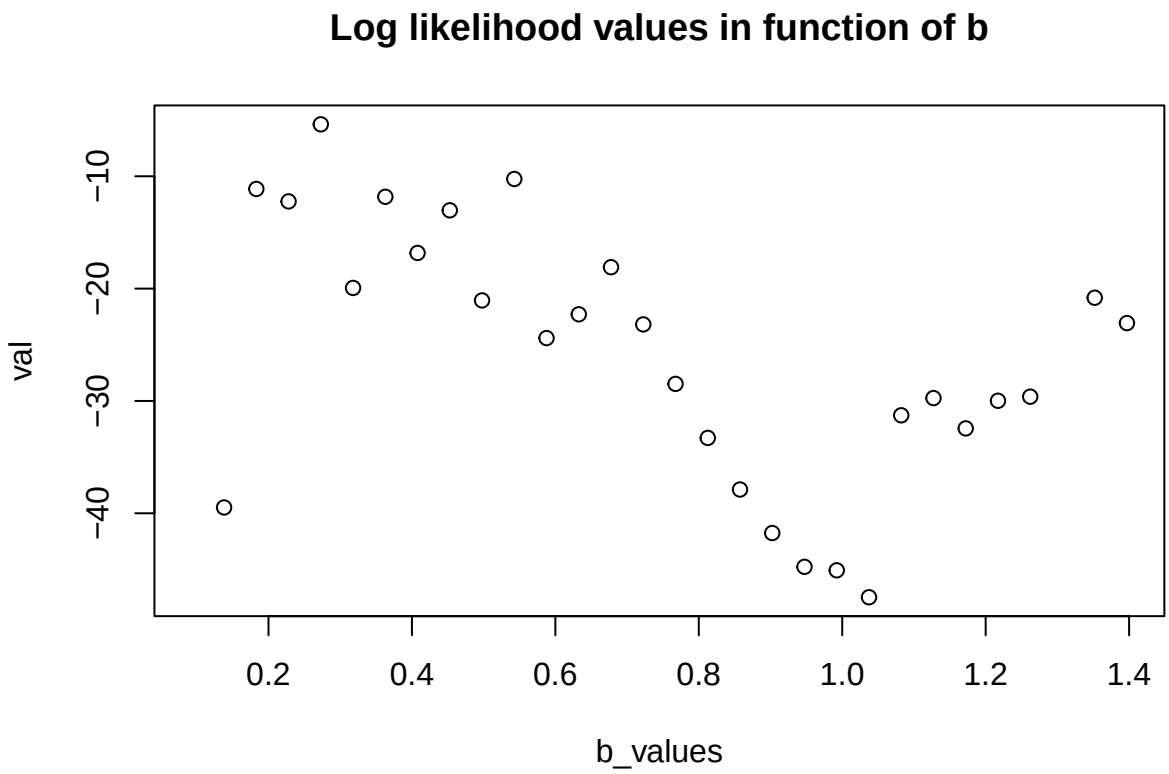
```
hist(x,breaks=seq(A,Z,length=nbr_opt+1),freq=F,main="Histogram of x with the optimal nbr")
```



Question 6

We applied the LOOCV to find the optimal value of the bins of the histogram $b = 0.2729$. Optimal b found equals to 0.272977 and it was determined with LOOCV ensuring that the histogram would align with data. A plotting of the log-likelihood values according to b is added in demostration of the obtained results. In addition, the histogram shows the representation of the data's distribution.

```
# Computing the optimal value of b
b_values  <- seq((Z-A)/15, (Z-A)/1, length= 30)
val = c()
optimal_b <-NULL

for (b in b_values){
  hx <- hist(x,breaks=seq(A,Z+b,by=b), plot=F)
  hx_f = stepfun(hx$breaks,c(0,hx$density,0))
  log_like <- log_likelihood(x,hx,hx_f)
  val <- append(val,log_like)
}

ind = which.max(val)
optimal_b = b_values[ind]

plot(b_values,val,main="Log likelihood values in function of b")
```

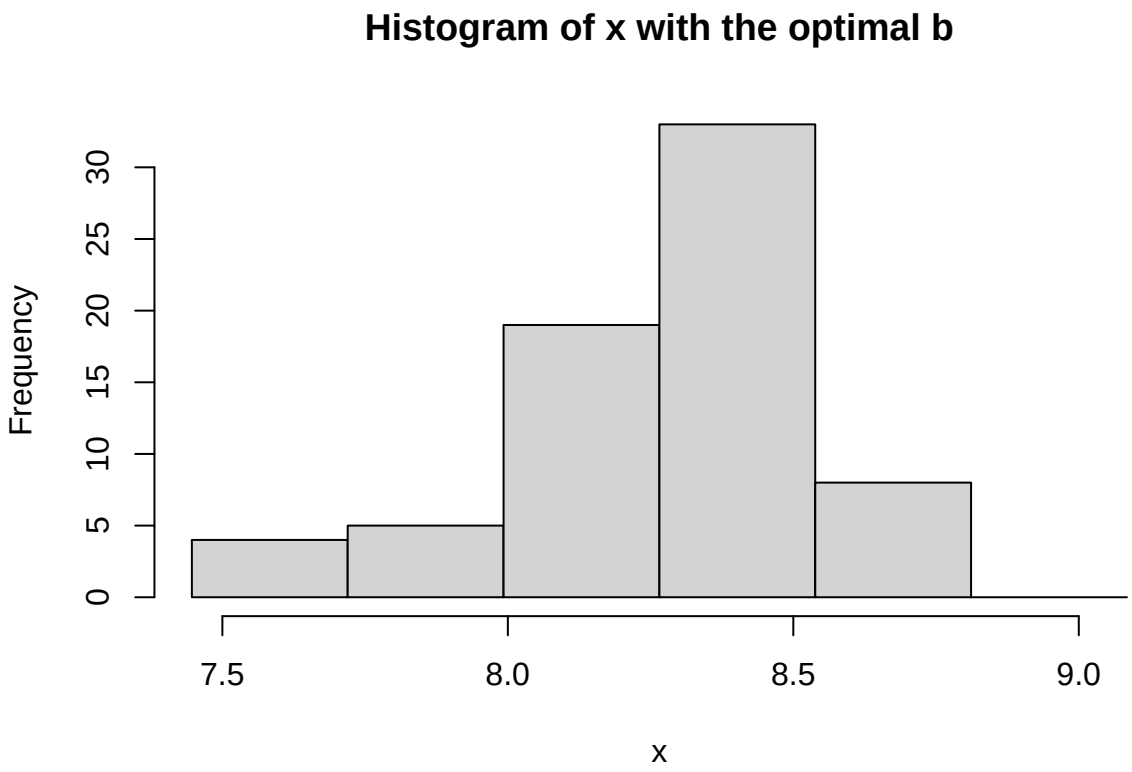



```
print(paste0("The optimal value of b is ",optimal_b))
```

```
## [1] "The optimal value of b is 0.272977011494253"
```

Here is the histogram corresponding to the optimal b. We can observe that we have the same number of rectangles than for the optimal nbr but the shape of histogram is a little different, having the highest value at approximately 8.4

```
hist(x,breaks=seq(A,Z+optimal_b,by=optimal_b),main="Histogram of x with the optimal b")
```



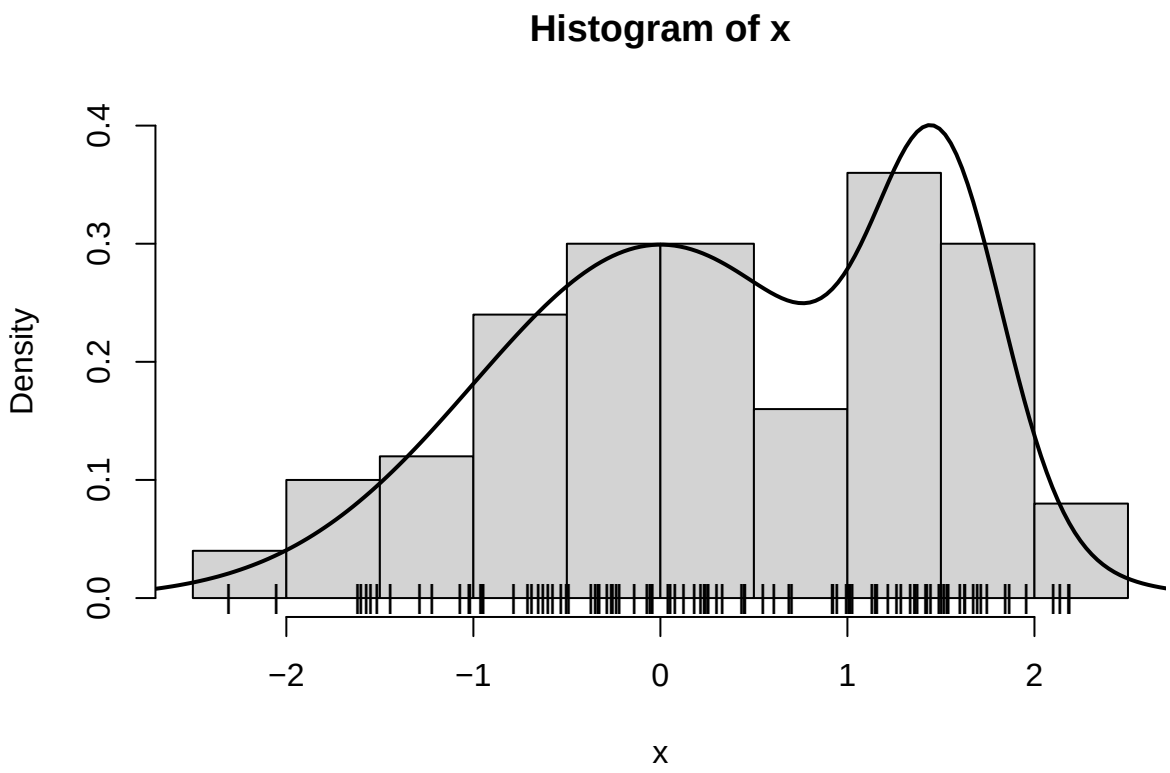
Question 7

First, the functions graph.mixt and sim.mixt are gotten from the density_estimation.Rmd. With these functions we generate new data by using the mixture of two normals. We then calculates the optimal b by LOOCV and compare it with the bin given by Scott's formulas : $b_{scott} = 3.49St.Dev(x)n^{-\frac{1}{3}}$ Scott's formula relies on the statistical properties of the data while the LOOCV method assesses the fit of the histogram to the data by the maximization of the log-likelihood. Yet again the results obatined differ between methods. Being the LOOCv method, the one providing the best result for b.

```
# graph.mixt
# Input:
#    k: number mixture components
#    mu: vector of length k with the mean values of the k normals
#    sigma: vector of length k with the st.dev. values of the k normals
#    alpha: vector of length k with the weights of each normal
#    graphic: logical value indicating if the mixture density must be plotted
#    ...: Other parameters passed to plot()
#
# Output:
#    L, U: extremes of the interval where the mixture density is plotted
#    x: points at which the mixture density is evaluated
#    fx: value of the mixture density at x
#
graph.mixt<-
function(k=1, mu=seq(-2*(k-1),2*(k-1),length=k), sigma=seq(1,1,length=k), alpha=seq(1/k,1/k,length=k),
{
   L<-min(mu-3*sigma)
```

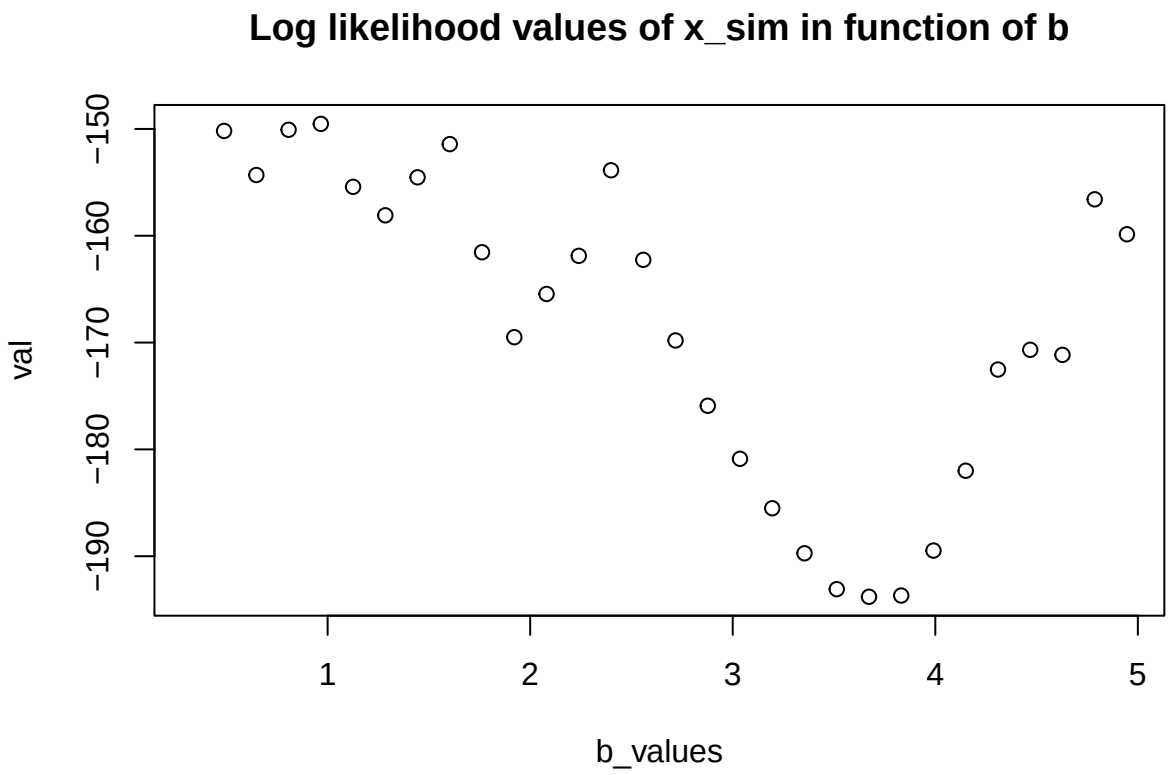
```
   U<-max(mu+3*sigma)

   x<- seq(from=L,to=U,length=200)
   fx<- 0*x
   Salpha<-sum(alpha)
   for(i in 1:k){
    p<-alpha[i]/Salpha
#       fx <- fx + p*exp(-.5*((x-mu[i])/sigma[i])^2)/(sqrt(2*pi)*sigma[i])
    fx <- fx + p*dnorm(x,mu[i],sigma[i])
   }
   if (graphic){
      plot(x,fx,type="l",...)
   }
   return(list(L = L, U = U, x = x, fx = fx))
}

# sim.mixt
# Input:
#    n: number of simulated data
#    k: number mixture components
#    mu: vector of length k with the mean values of the k normals
#    sigma: vector of length k with the st.dev. values of the k normals
#    alpha: vector of length k with the weights of each normal
#    graphic: logical value indicating if the mixture density and the
#              histogram of the simulated data must be plotted
#    ...: Other parameters passed to plot()
#
# Output:
#    x: simulated data
#
# Requires:
#    graph.mixt
sim.mixt <- function(n=1,k=1,
         mu=seq(-2*(k-1),2*(k-1),length=k),
         sigma=seq(1,1,length=k),
         alpha=seq(1/k,1/k,length=k), graphic=FALSE,...)
{
   csa<-cumsum(alpha)
   x<-runif(n)

   for (i in 1:n){
      comp<-sum(csa<=x[i])+1
      x[i]<-rnorm(1,mu[comp],sigma[comp])
   }
   if(graphic) {
      out<-graph.mixt(k, mu, sigma, alpha, gr=FALSE)
      hist(x,freq = FALSE,
           ylim=c(0,max(c(max(out$fx),max(hist(x,plot=FALSE)$density)))))
      lines(out$x,out$fx,lty=1,lwd=2)
   }
   return(x)
}
```

```r
set.seed(123)
n <- 100
mu <- c(0,3/2)
sigma <- c(1,1/3)
alpha <- c(3/4,1/4)

# Generating data from the mixture of two normals
x_sim <- sim.mixt(n=n, k=2, mu=mu, sigma=sigma, alpha=alpha, gr=T)
points(x_sim,0*x_sim,pch="|")
```




```r
# Calculating the optimal b for the data generated from the mixture of two normals
A = min(x_sim) - .05*diff(range(x_sim))
Z = max(x_sim) + .05*diff(range(x_sim))
b_values <- seq((Z-A)/15,(Z-A)/1,length=30)
val = c()
optimal_b <-NULL

for (b in b_values){
  hx <- hist(x_sim,breaks=seq(A,Z+b,by=b), plot=F)
  hx_f = stepfun(hx$breaks,c(0,hx$density,0))
  log_like <- log_likelihood(x_sim,hx,hx_f)
  val <- append(val,log_like)
}

ind = which.max(val)
```

```
optimal_b = b_values[ind]

plot(b_values,val,main="Log likelihood values of x_sim in function of b")
```




```
print(paste0("The optimal b for the generated data is: ", optimal_b))
```

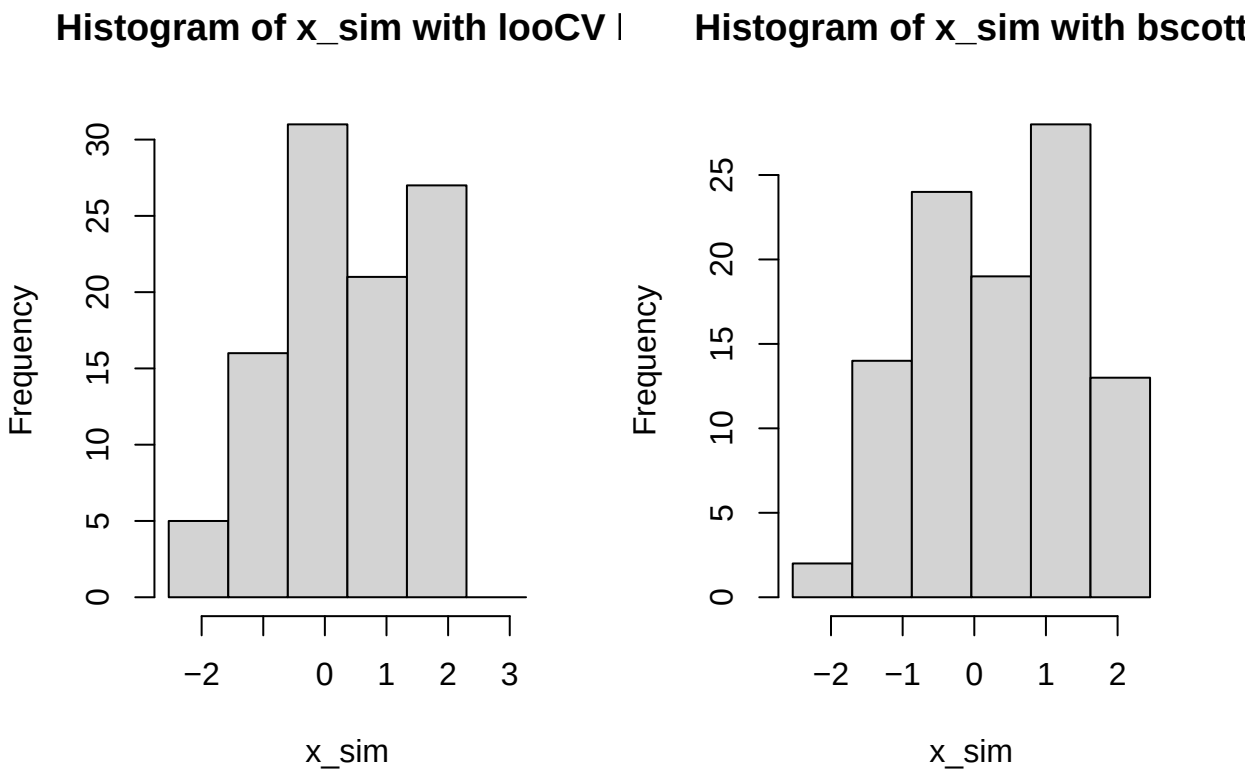
```
## [1] "The optimal b for the generated data is: 0.966489482113715"
```

```
scottform_b <- 3.49 * sd(x_sim) * n^(-1/3)
print(paste0("The optimal b for Scott's formula is: ", scottform_b))
```

```
## [1] "The optimal b for Scott's formula is: 0.831198505080386"
```

The optimal values from LOOCV is $b_{opt} = 0.966489482113715$ and the one from Scott's formula is $b_{scott} = 0.831198505080386$. The values are different by 0.1 but for further comparison, the plot of the corresponding histograms is added. It can be seen that the histogram are quite similar but with different nbr values.

```
par(mfrow=c(1,2))
hist(x_sim, breaks=seq(A,Z+optimal_b,by=optimal_b), main = "Histogram of x_sim with looCV b")
hist(x_sim, breaks=seq(A,Z+scottform_b,by=scottform_b), main = "Histogram of x_sim with bscott")
```

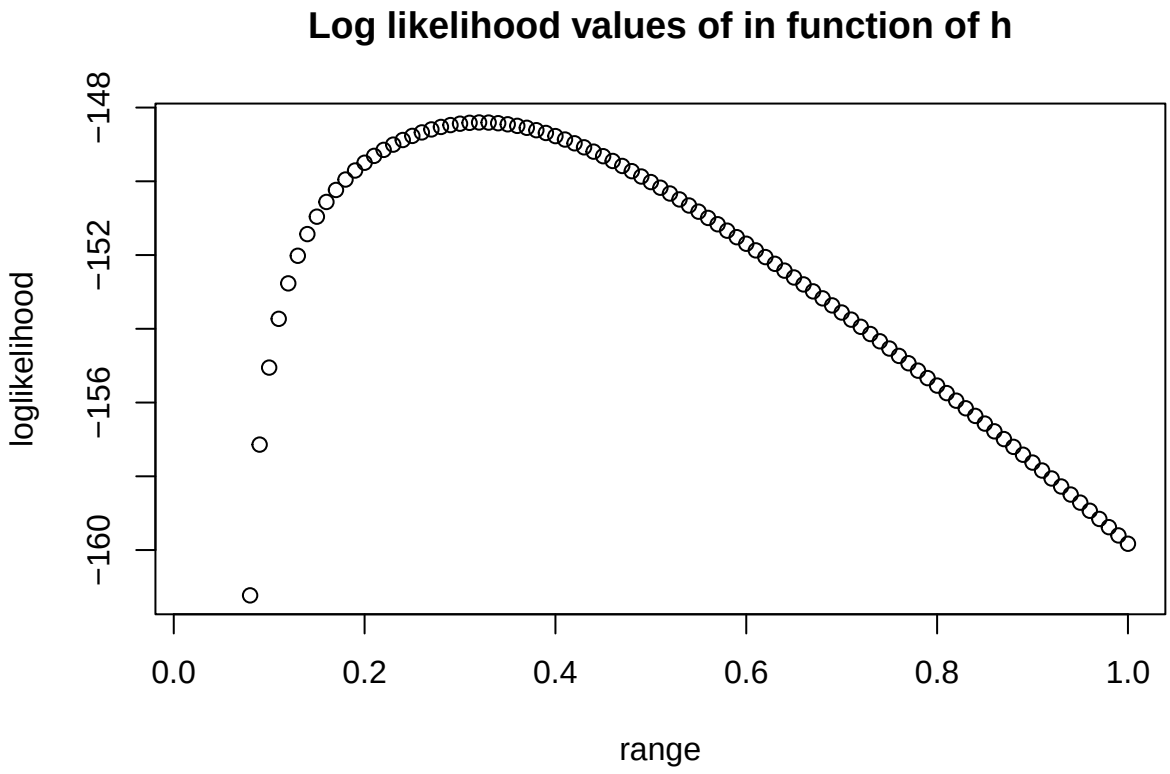


Question 8

LOOCV was used to evaluate the quality of the kernel density estimator and the formula provided was used to calculate the kernel density estimator and using approxfun allowed an evaluation of the estimated density over the values. The optimal bandwidth for the kernel density estimator is 0.01 and provided the best fit for the distribution of the data. Finally the plot shows the data distribution with the optimal bandwidth.

```
fkernel_loo <- function(x,kx_f,h){
  # Returns the leave-one-out log likelihood functions for the kernel density estimation with a gaussia
  # Input :
  #   x : points
  #   hist : kernel density estimator object
  #   f_h : kernel density estimator function
  # Output :
  #   leave-one-out log likelihood

  n <- length(x)
  f_i <- c()
  K_0 = dnorm(0) # Gaussian kernel
  for(i in 1:n){
    f_i<-append(f_i,(n/(n-1))*(kx_f(x[i])-(K_0/(n*h))))
  }
  return(f_i)
}
```

```r
# Calculating the optimal h by looCV
loglikelihood <- c()
range =  seq(0.02, 1.0, by = 0.01)
for(h in range){
  kx <- density(x_sim,bw=h,kernel ='gaussian')
  kx_f <- approxfun(x=kx$x, y=kx$y, method='linear', rule=2)
  loglikelihood <- append(loglikelihood,(sum(log(fkernel_loo(x_sim,kx_f,h)))))
}
optimal_h <- range[which.max(loglikelihood)]
plot(range,loglikelihood,main="Log likelihood values of in function of h")
```

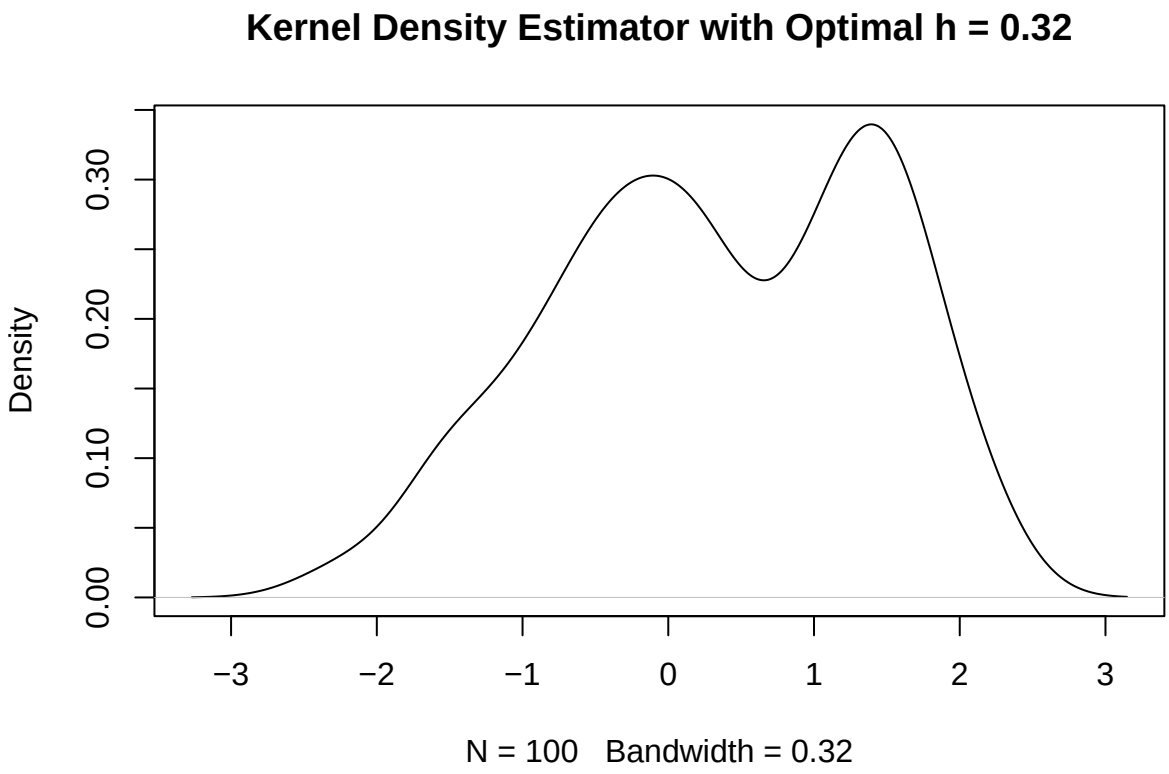



```r
print(paste0("The optimal h is ",optimal_h))
```

```
## [1] "The optimal h is 0.32"
```

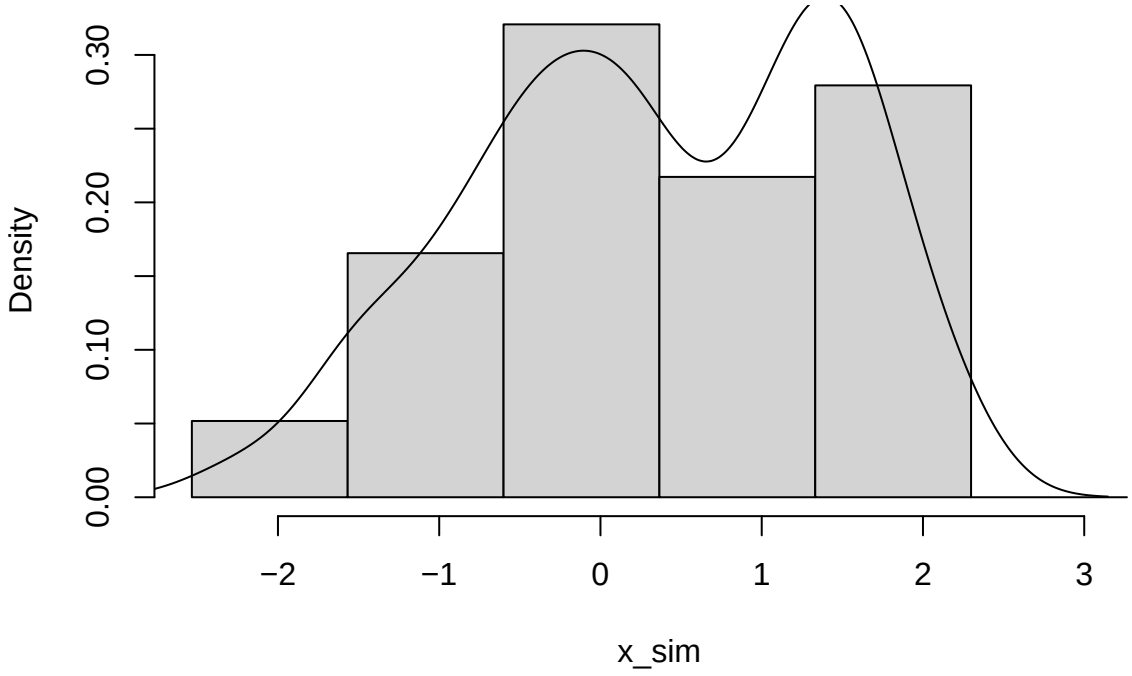
The following figure shows the kernel density estimator with the optimal value of h, $h = 0.32$.

```r
kx_optimal <- density(x_sim, bw = optimal_h)
plot(kx_optimal, main = paste("Kernel Density Estimator with Optimal h =", round(optimal_h, 2)))
```



We can compare the obtained kernel density estimation with the histogram obtained with the optimal b. It is clear that the two shapes are similar which firstly means that the obtained estimator make sense. We can also see that the kernel density estimation is more precise than the histogram.

```
hist(x_sim, breaks=seq(A,Z+optimal_b,by=optimal_b), main = "Histogram against density estimation",freq =
lines(kx_optimal, main = paste("Kernel Density Estimator with Optimal h =", round(optimal_h, 2)))
```


Histogram against density estimation
Density
0.00
0.10
0.20
0.30
−2
−1
0
1
2
3
x_sim