Density estimation

Group 1: Pariente Antonio, Bosch Guillem, Ebner Lena

27/09/2023

Histogram

1.

At the slides we have seen the following relationship

$$\hat{f}_{h,(-i)}(x_i) = \frac{n}{n-1}\left(\hat{f}_h(x_i) - \frac{K(0)}{nh}\right)$$

between the leave-one-out kernel density estimator $\hat{f}_{h,(-i)}(x)$ and the kernel density estimator using all the observations $\hat{f}_h(x)$, when both are evaluated at x_i, one of the observed data. Find a similar relationship between the histogram estimator of the density function $\hat{f}_{\text{hist}}(x)$ and its leave-one-out version, $\hat{f}_{\text{hist},(-i)}(x)$, when both are evaluated at x_i.

The new equation we have derived is

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

2.

Read the CD rate data set and call `x` the first column.

```
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
```

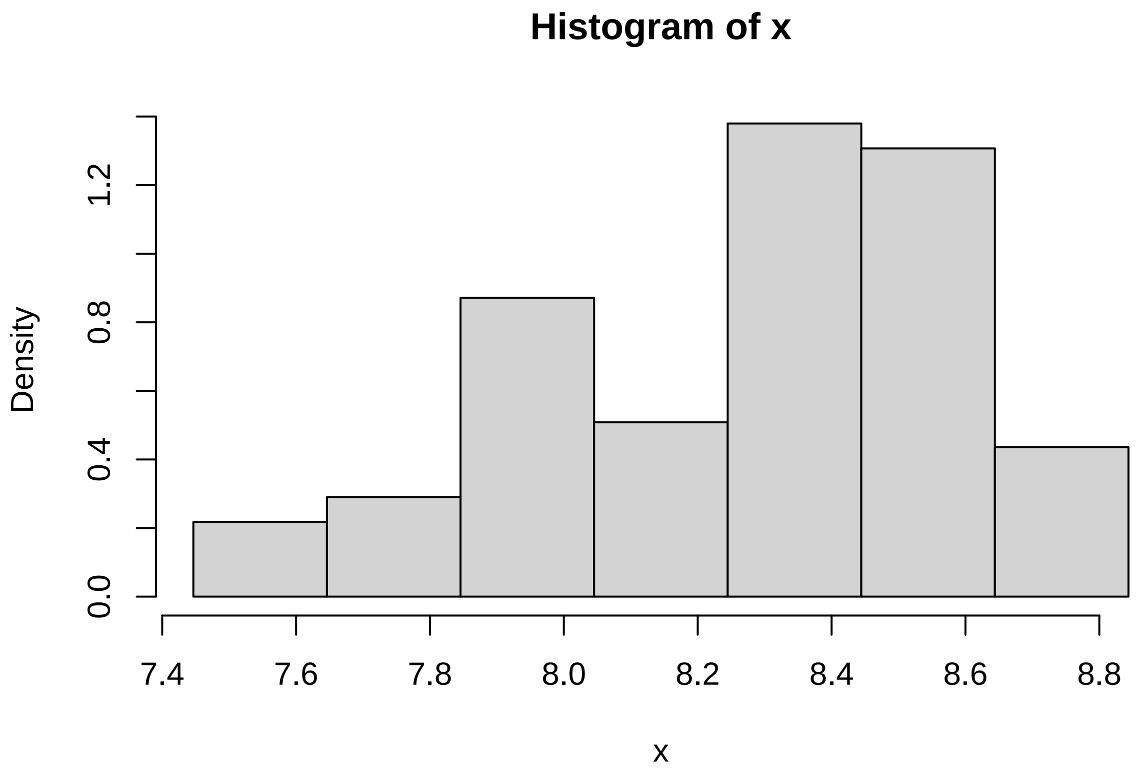
```
x <- cdrate.df[,1]
```

Then define

```
A <- min(x)-.05*diff(range(x))
Z <- max(x)+.05*diff(range(x))
nbr <- 7
```

and plot the histogram of x as

```
hx <- hist(x,breaks=seq(A,Z,length=nbr+1),freq=F)
```

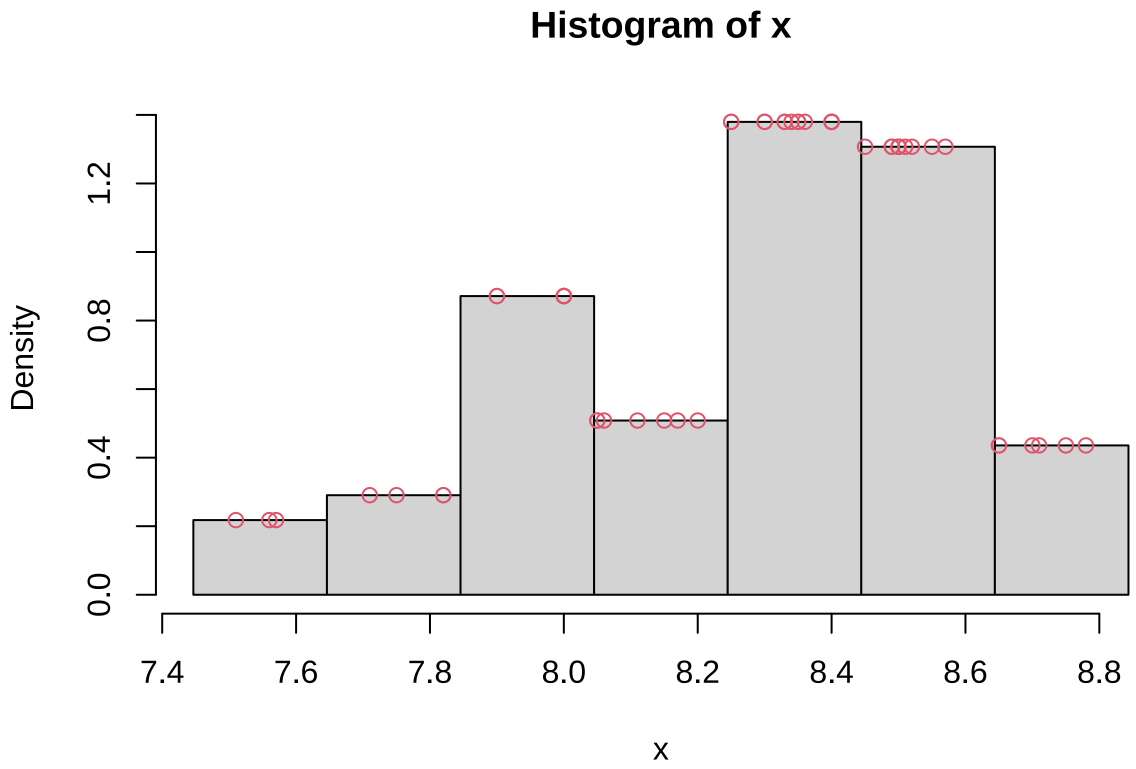


The following sentence converts this histogram into a function that can be evaluated at any point of R, or at a vector of real numbers:

```
hx_f <- stepfun(hx$breaks,c(0,hx$density,0))
```

Use `hx_f` to evaluate the histogram at the vector of observed data x. Then add the points (xi,f^hist(xi)), i=1,. . . ,n, to the histogram you have plotted before.

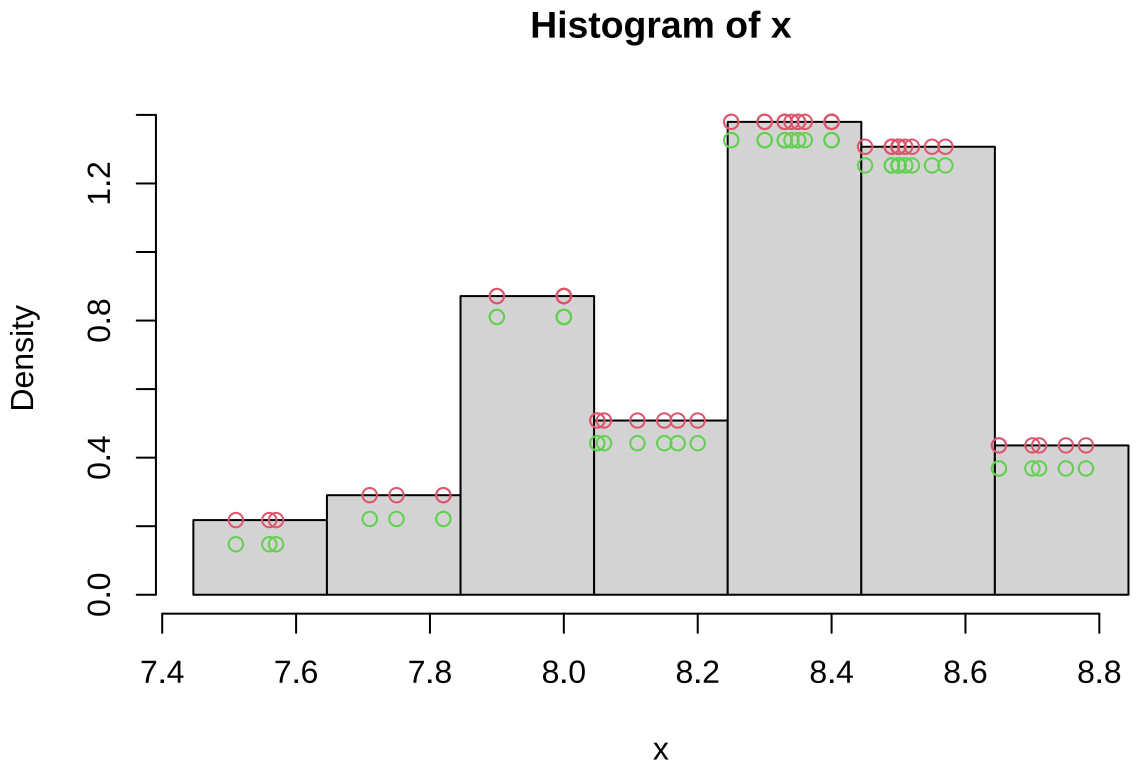
```
y=hx_f(x)
hist(x,breaks=seq(A,Z,length=nbr+1),freq=F)
points(x,y,col=2)
```



3.

Use the formula you have found before relating $\hat{f}_{\text{hist}}(xi)$ and $\hat{f}_{\text{hist},(-i)}(x_i)$ to compute $\hat{f}_{\text{hist},(-i)}(x_i)$, $i = 1, \ldots, n$. Then add the points $(x_i, \hat{f}_{\text{hist},(-i)}(x_i))$, $i = 1, \ldots, n$, to the previous plot.

```
b = hx$breaks[2] - hx$breaks[1]
n=length(x)
f_loovc = function(p) (hx_f(p)*n*b-1)/(b*(n-1))
hx_loocv=lapply(x,f_loovc)

hist(x,breaks=seq(A,Z,length=nbr+1),freq=F)
points(x,y,col=2)
points(x,hx_loocv,col=3)
```



4.

Compute the leave-one-out log-likelihood function corresponding to the previous histogram, at which `nbr=7` has been used.

```
log_likelihood=sum(log(unlist(hx_loocv)))
log_likelihood
```

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

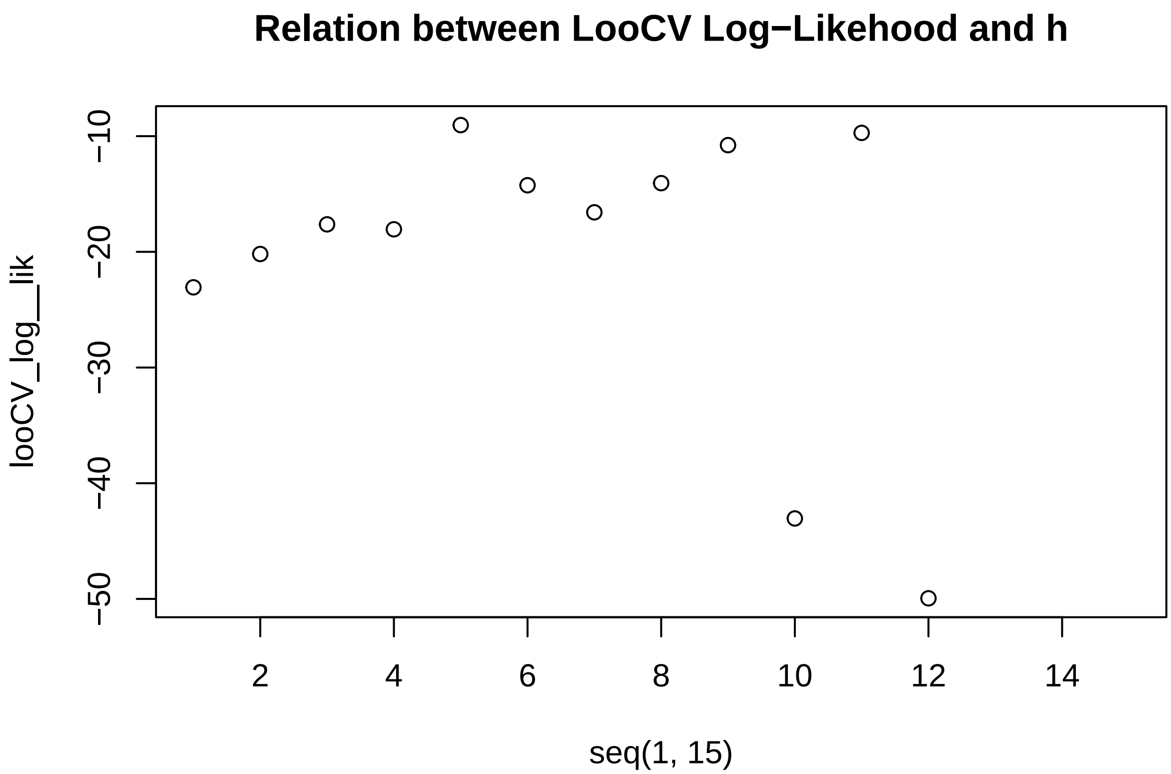
5.

Choosing `nbr` by leave-one-out Cross Validation (looCV). Consider now the set `seq(1,15)` as possible values for `nbr`, the number of intervals of the histogram. For each of them compute the leave-one-out log-likelihood function (`looCV_log_lik`) for the corresponding histogram. Then plot the values of `looCV_log_lik` against the values of `nbr` and select the optimal value of `nbr` as that at which `looCV_log_lik` takes its maximum. Finally, plot the histogram of x using the optimal value of `nbr`.

```
n=length(x)
hx_loocv=list()
looCV_log__lik=list()
y=list()
for (i in seq(1,15)){
  hx <- hist(x,breaks=seq(A,Z,length=i+1),plot=F)
  hx_f <- stepfun(hx$breaks,c(0,hx$density,0))
  b = hx$breaks[2] - hx$breaks[1]
  f_loocv = function(p) (hx_f(p)*n*b-1)/(b*(n-1))
  y[length(y)+1] <- list(hx_f(x))
  hx_loocv = lapply(x,f_loocv)
  looCV_log__lik = append(looCV_log__lik,sum(log(unlist(hx_loocv))))
}
```

```
## Warning in log(unlist(hx_loocv)): NaNs produced

## Warning in log(unlist(hx_loocv)): NaNs produced
```

```
plot(seq(1,15),looCV_log__lik)
title("Relation between LooCV Log-Likehood and h")
```

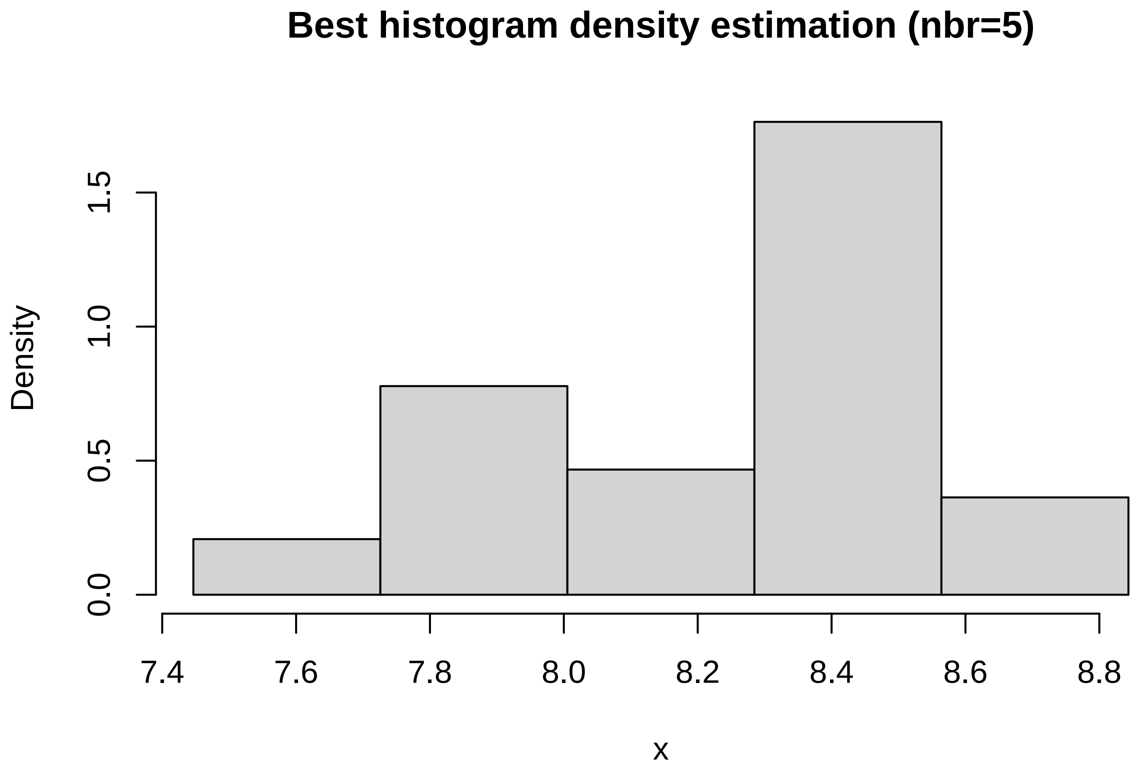


The best result can be found with `nbr=5`

```
best=seq(1,15)[which.max(looCV_log__lik)]
best
```

```
## [1] 5
```

```
hist(x,breaks=seq(A,Z,length=best+1), main="Best histogram density estimation (nbr=5)", freq=F)
```



6.

Choosing `b` by looCV. Let b be the common width of the bins of a histogram. Consider the set

```
b=seq((Z-A)/15,(Z-A)/1,length=30)
```

as possible values for `b`. Select the value of `b` maximizing the leave-one-out log-likelihood function, and plot the corresponding histogram. *NOTE*: To avoid errors, use the following syntax for computing a histogram with bin width `b`

```
hx <- hist(x,breaks=seq(A,Z+b,by=b), plot=F)
```

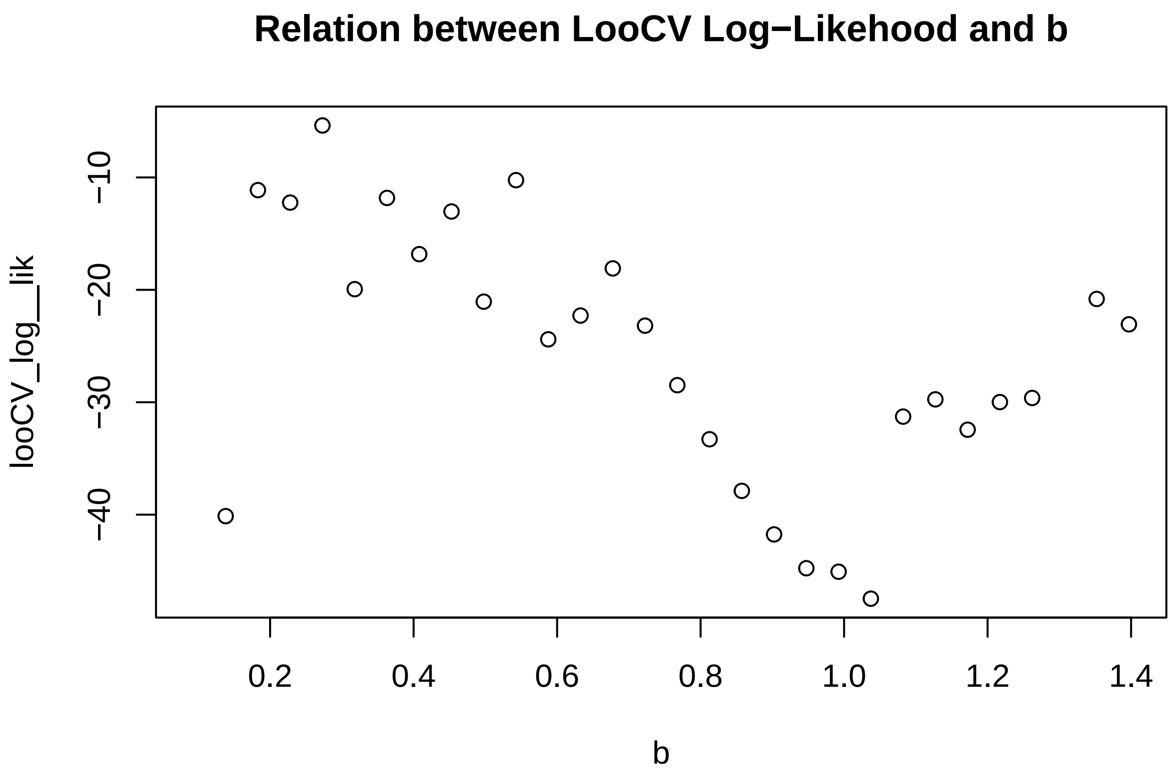
and this sentence to plot it:

```
plot(hx,freq = FALSE)
```

```
b=seq((Z-A)/15,(Z-A)/1,length=30)
n=length(x)
y=list()
looCV_log__lik=list()
for (i in b){
  hx <- hist(x,breaks=seq(A,Z+i,by=i),plot=F)
  hx_f <- stepfun(hx$breaks,c(0,hx$density,0))
  f_loocv = function(p) (hx_f(p)*n*i-1)/(i*(n-1))
  y[length(y)+1]<-list(hx_f(x))
  hx_loocv=lapply(x,f_loocv)
  looCV_log__lik=append(looCV_log__lik,sum(log(unlist(hx_loocv))))
}
```

```
## Warning in log(unlist(hx_loocv)): NaNs produced

## Warning in log(unlist(hx_loocv)): NaNs produced
```

```
plot(b,looCV_log__lik)
title("Relation between LooCV Log-Likehood and b")
```

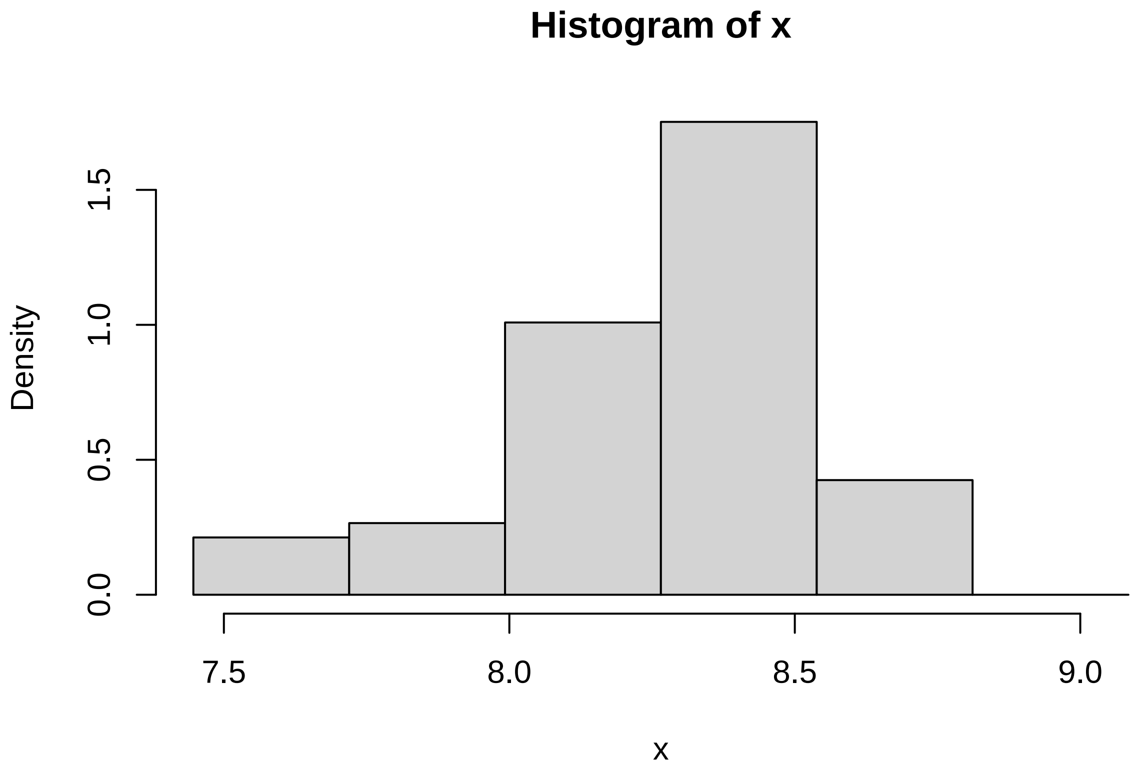


Best estimation can be found at $b = 0.272977$

```
best=b[which.max(looCV_log__lik)]
best
```

```
## [1] 0.272977
```

```
hist(x,breaks=seq(A,Z+best,by=best), freq=F)
```



```
hist(x,breaks=seq(A,Z+best,by=best), plot=F)
```

```
## $breaks
## [1] 7.446500 7.719477 7.992454 8.265431 8.538408 8.811385 9.084362
##
## $counts
## [1]  4  5 19 33  8  0
##
## $density
## [1] 0.2123659 0.2654574 1.0087381 1.7520188 0.4247318 0.0000000
##
## $mids
## [1] 7.582989 7.855966 8.128943 8.401920 8.674897 8.947874
##
## $xname
## [1] "x"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

7.

Recycle the functions `graph.mixt` and `sim.mixt` defined at `density_estimation.Rmd` to generate n=100 data from

$$f(x) = (3/4)N(x; m = 0, s = 1) + (1/4)N(x; m = 3/2, s = 1/3)$$

Let `b` be the bin width of a histogram estimator of f(x) using the generated data. Select the value of `b` maximizing the leave-one-out log-likelihood function, and plot the corresponding histogram. Compare with the results obtained using the Scott's formula:

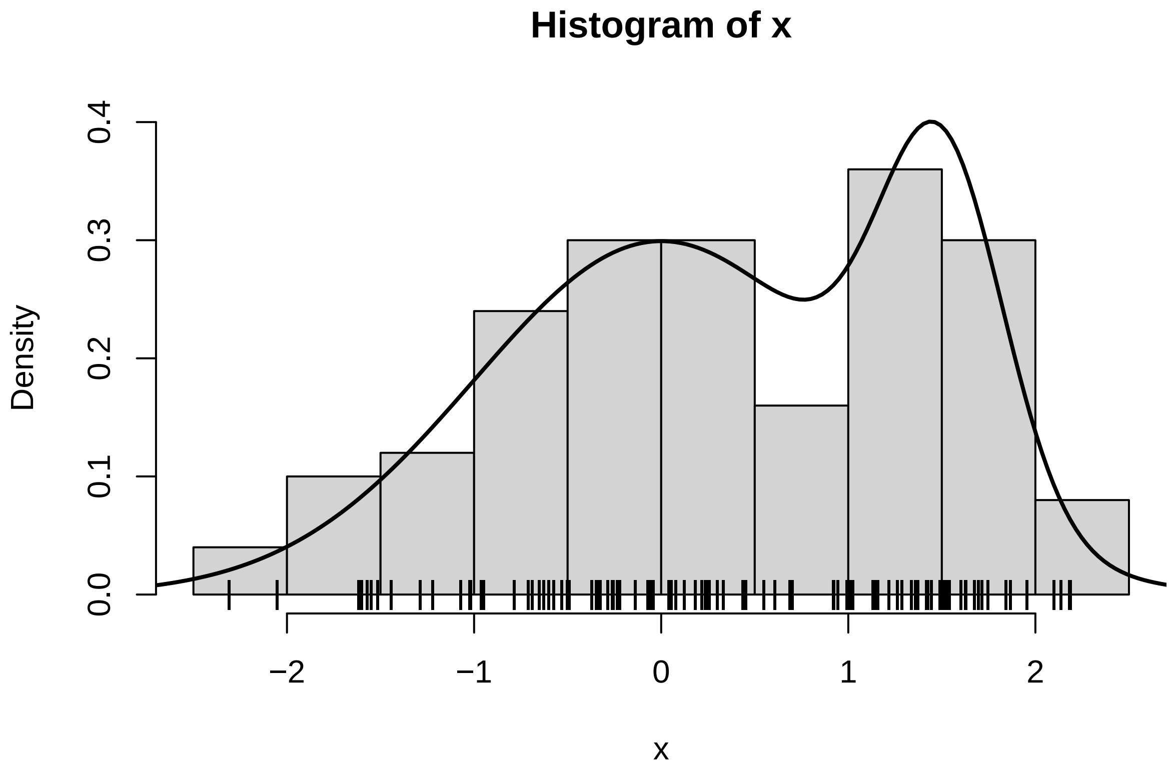
$$b_{\text{Scott}} = 3.49 \cdot \text{SD}(X)n^{-\frac{1}{3}}.$$

```
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), g
{
   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 <- 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)
}
```

Generate $n = 100$ data from

$$f(x) = (3/4)N(x; m = 0, s = 1) + (1/4)N(x; m = 3/2, s = 1/3)$$

```
set.seed(123)
n <- 100
mu <- c(0,3/2)
sigma <- c(1,1/3)
alpha <- c(3/4,1/4)
x <- sim.mixt(n=n, k=2, mu=mu, sigma=sigma, alpha=alpha, gr=T)
points(x,0*x,pch="|")
```



Make the usual estimations together with an additional one using the `scott_b` parameter.

```
A <- min(x)-.05*diff(range(x))
Z <- max(x)+.05*diff(range(x))
b=seq((Z-A)/15,(Z-A)/1,length=30)
n=length(x)
y=list()
looCV_log__lik=list()
for (i in b){
  hx <- hist(x,breaks=seq(A,Z+i,by=i),plot=F)
  hx_f <- stepfun(hx$breaks,c(0,hx$density,0))
  f_loovc = function(p) (hx_f(p)*n*i-1)/(i*(n-1))
  y[length(y)+1]<-list(hx_f(x))
  hx_loocv=lapply(x,f_loovc)
  looCV_log__lik=append(looCV_log__lik,sum(log(unlist(hx_loocv))))
}
```

```
scott_b=3.49*sd(x)*length(x)^(-1/3)
scott_b
```

```
## [1] 0.8311985
```

```
scott_hx <- hist(x,breaks=seq(A,Z+scott_b,by=scott_b),plot=F)
scott_hx_f <- stepfun(scott_hx$breaks,c(0,scott_hx$density,0))
scott_f_loovc = function(p) (scott_hx_f(p)*n*scott_b-1)/(scott_b*(n-1))
scott_y=scott_hx_f(x)
scott_hx_loocv=lapply(x,scott_f_loovc)
scott_looCV_log__lik=sum(log(unlist(scott_hx_loocv)))
scott_looCV_log__lik
```

```
## [1] -150.3579
```

```
best=b[which.max(looCV_log__lik)]
# maybe it is better to put this as text in the markdown
print(paste("best b (LOOCV)",best," l-likelihood=",max(unlist(looCV_log__lik))))
```

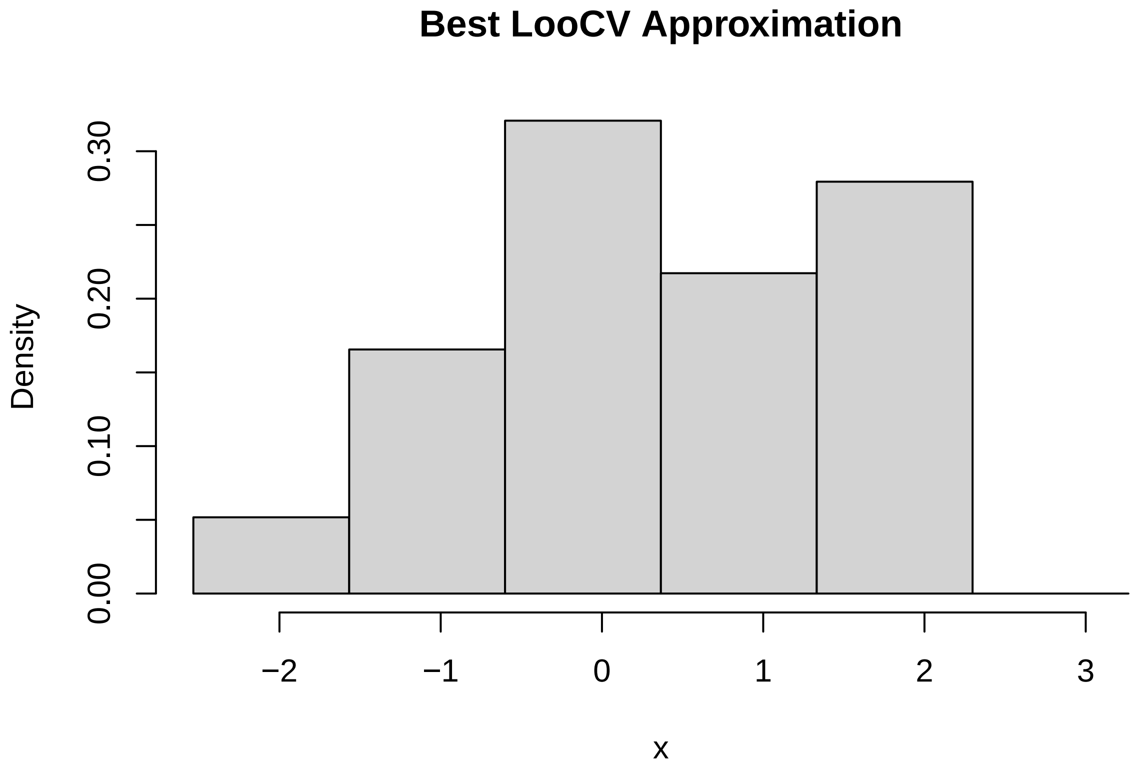
```
## [1] "best b (LOOCV) 0.966489482113715  l-likelihood= -149.527120603357"
```

```
print(paste("best scott b",scott_b," l-likelihood=",scott_looCV_log__lik))
```

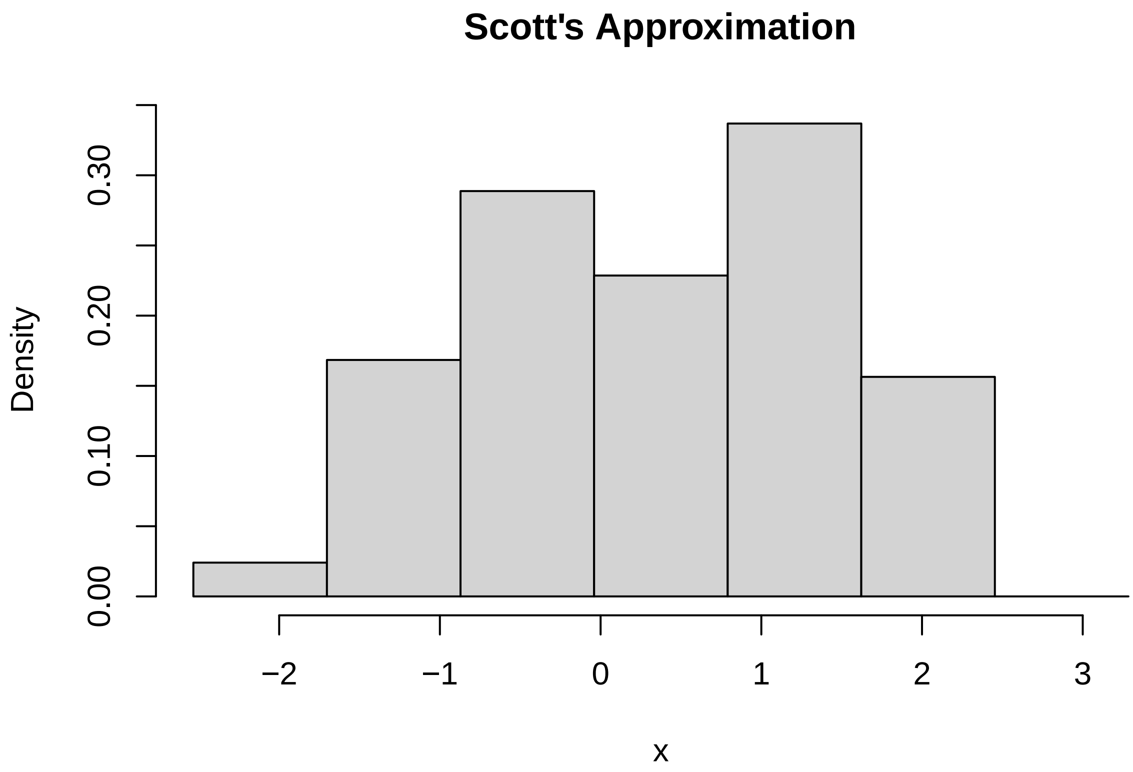
```
## [1] "best scott b 0.831198505080386  l-likelihood= -150.35788696066"
```

Best LOOCV b: 0.9665, Log-Likelihood: -149.527
Scott's Formula b: 0.8312, Log-Likelihood: -150.358

```
hist(x,breaks=seq(A,Z+best,by=best),main="Best LooCV Approximation",freq=F)
```



```
# plot(x,unlist(y[which.max(looCV_log__lik)]))

hist(x,breaks=seq(A,Z+best,by=scott_b),main="Scott's Approximation",freq=F)
```



```
# plot(x,unlist(scott_y))
```

Kernel density estimation

8.

Consider the vector `x` of data you have generated before from the mixture of two normals. Use the relationship

$$\hat{f}_{h,(-i)}(x_i) = \frac{n}{n-1}\left(\hat{f}_h(x_i) - \frac{K(0)}{nh}\right)\text{‘}$$

to select the value of `h` maximizing the leave-one-out log-likelihood function, and plot the corresponding kernel density estimator. *NOTE*: The following sentences converts the kernel density estimator obtained with the function `density` into a function that can be evaluated at any point of R or at a vector of real numbers:

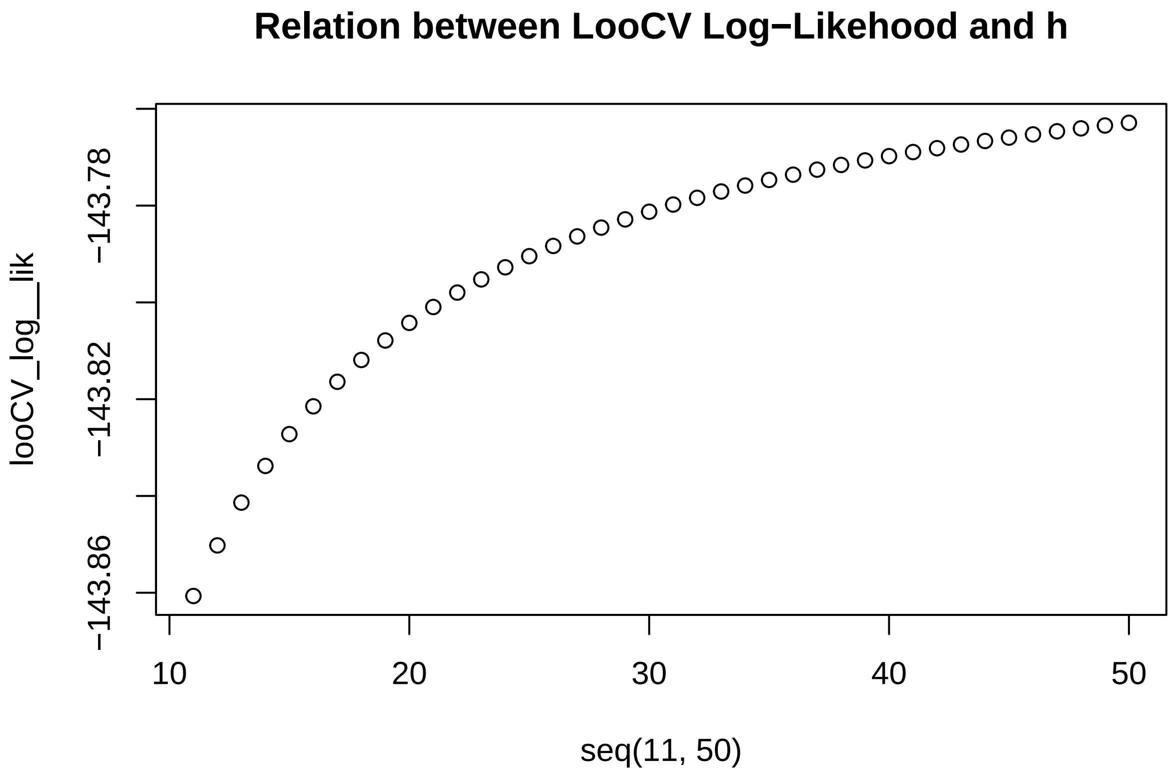
```
kx <- density(x)
x_f <- approxfun(x=kx$x, y=kx$y, method='linear', rule=2)
```

Here the `h` is pulled from a [11,50] interval, and the estimation is the kernel one performed in theory lessons. However, the value seems to improve in a logarithmic way with respect to h. This makes sense due to the kernel choice, since it is the one chosen by R. We considered that, since the scope of this practice is not to make a good fit (avoiding regression), 50 is a sufficient number.

```
n=length(x)
hx_loocv=list()
```

```
looCV_log__lik=list()
y=list()
for (h in seq(11,50)){
  kx <- density(x)
  x_f <- approxfun(x=kx$x, y=kx$y, method='linear', rule=2)
  f_loovc = function(p) (n/(n-1))*(x_f(p)-(x_f(0)/(n*h)))
  y[length(y)+1]<-list(x_f(x))
  hx_loocv=lapply(x,f_loovc)
  looCV_log__lik=append(looCV_log__lik,sum(log(unlist(hx_loocv))))
}
```

```
plot(seq(11,50),looCV_log__lik)
title("Relation between LooCV Log-Likehood and h")
```

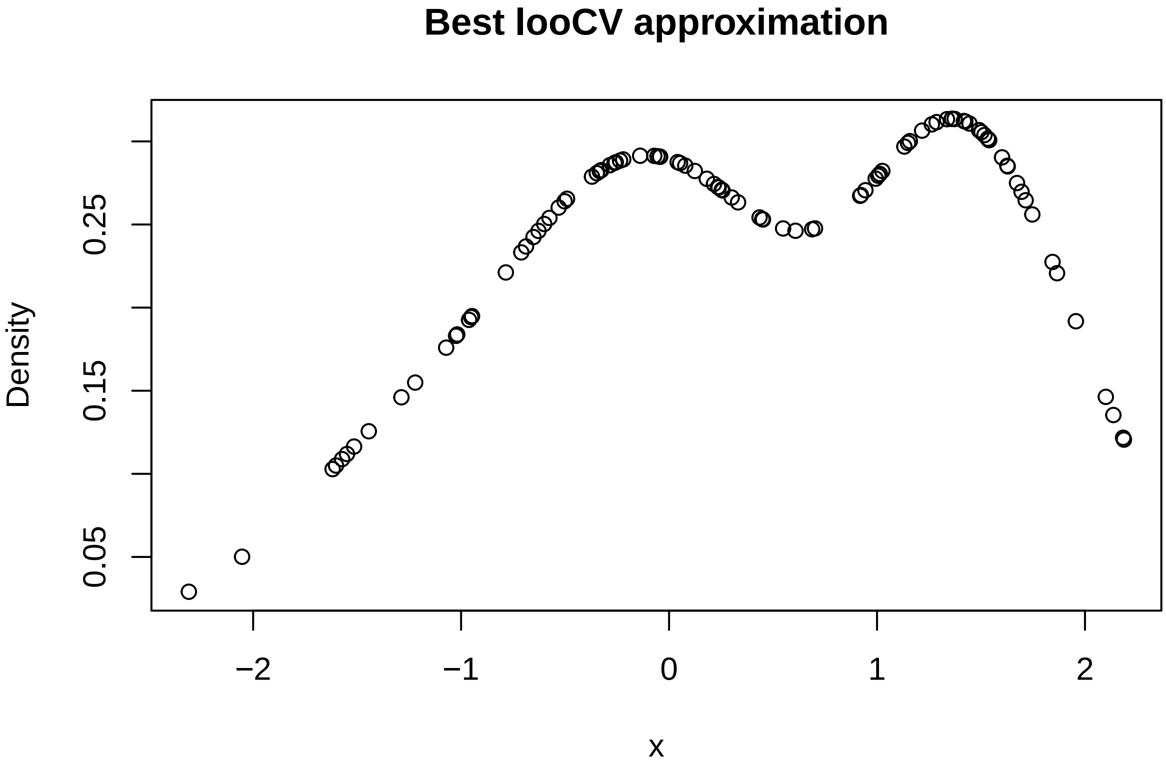


The best estimation is $h = 50$

```
best=seq(11,50)[which.max(looCV_log__lik)]
best
```

```
## [1] 50
```

```
plot(x,unlist(y[which.max(looCV_log__lik)]),ylab="Density")
title("Best looCV approximation")
```


Best looCV approximation
Density
0.05
0.15
0.25
−2
−1
0
1
2
x