Computational Statistics. Chapter 4: Classical simulation. Solution of exercises

Thierry Denoeux

2024-03-07

Exercise 1

Question 1a

The cdf is

$$F(x) = \int_{-\infty}^{x} f(t)dt = \begin{cases} 0 & x \leq 0 \\ \int_0^x \frac{2}{a^2} t dt = \frac{2}{a^2}\left[\frac{t^2}{2}\right]_0^x = \frac{x^2}{a^2} & 0 < x \leq a \\ 1 & x > a. \end{cases}$$

Its inverse is

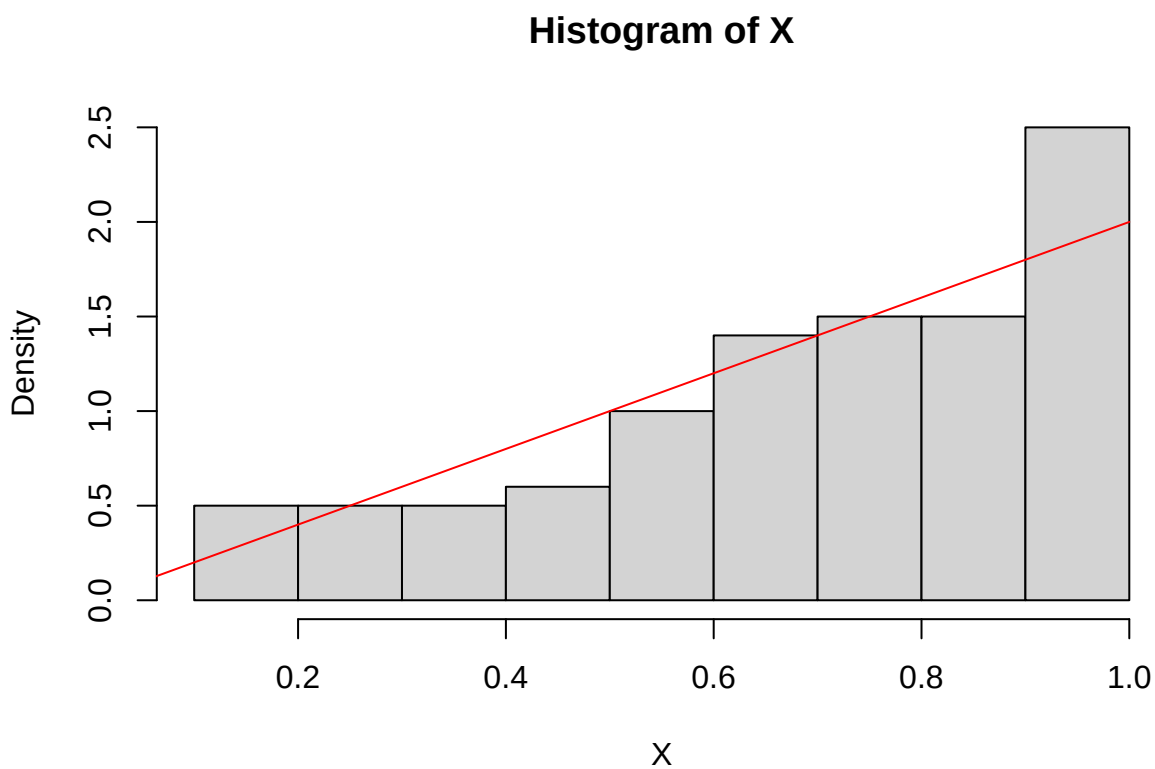
$$F^{-1}(u) = a\sqrt{u}.$$

The probability integral transform (PIT) method consists in generating U from a standard uniform distribution, and setting $X = a\sqrt{U}$. Let us set $a = 1$ and generate a sample of size $n = 100$:

```
a<-1
n<-100
U<-runif(n)
X <- a*sqrt(U)
```

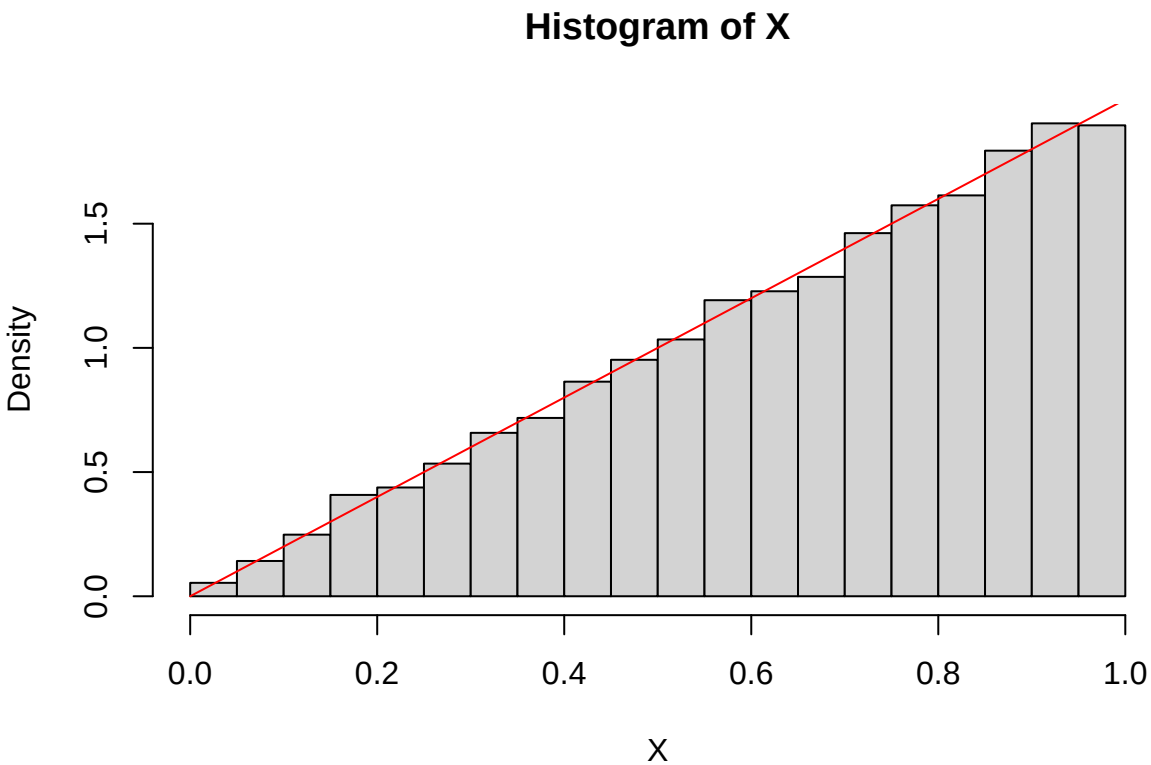
Let us compare the histogram to the population density:

```
hist(X,freq=FALSE)
lines(c(0,a),c(0,2*a),col="red")
```



Now, with $n = 10^4$:

```
n<-10000
U<-runif(n)
X <- a*sqrt(U)
hist(X,freq=FALSE)
lines(c(0,a),c(0,2*a),col="red")
```



Question 1b

To apply the rejection sampling (RS) method, we need an envelope $e(x) = g(x)/\alpha \geq f(x)$, where $g(x)$ is a density we can sample from. We can take for $g(x)$ the uniform density in $[0, a]$: $g(x) = 1/a$ and $\alpha = 0.5$, which gives us the envelope $e(x) = 2/a$.

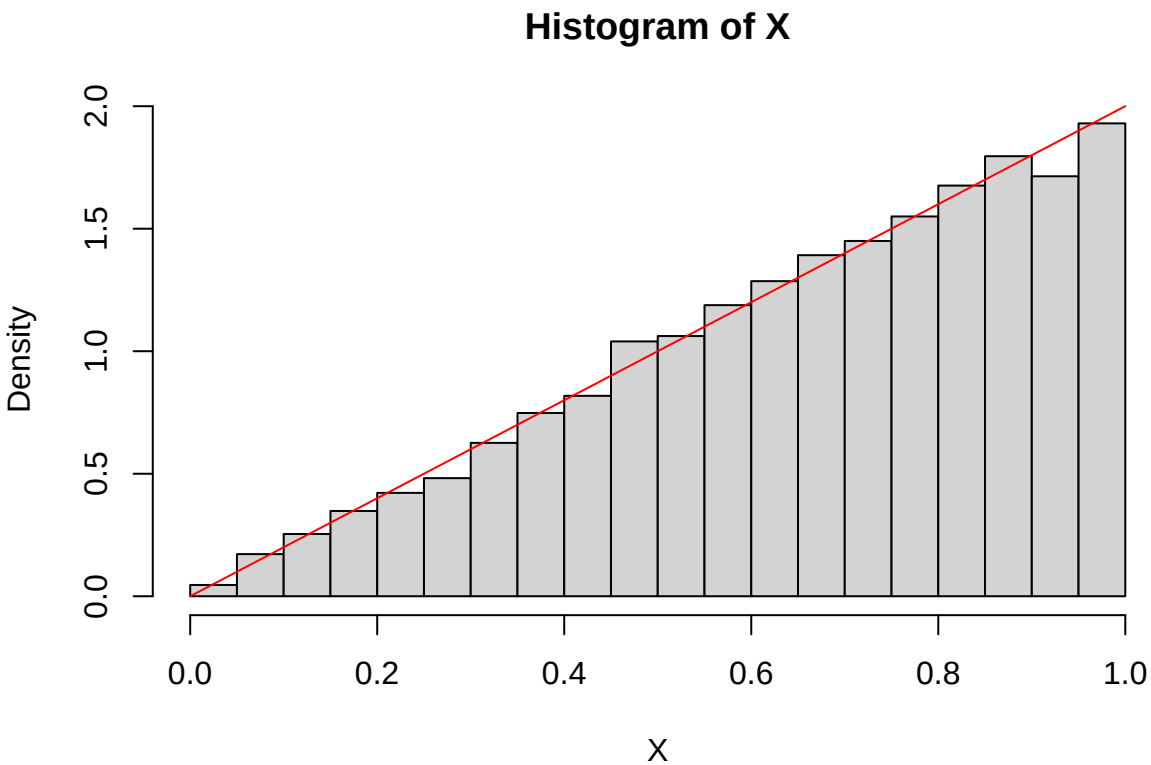
The RS algorithm can then be coded as follows:

```
n<-10000
X<-rep(0,n)
i<-0
j<-0
f <- function(x,a) 2/a^2*x

while(i<n){
  j <- j+1
  Y <- runif(1,0,a)
  U <- runif(1)
  if(U<f(Y,a)/(2/a)){
    i <- i+1
    X[i]<-Y
  }
}
```

Again, let us plot the histogram with the population density:

```
hist(X,freq=FALSE)
lines(c(0,a),c(0,2*a),col="red")
```



The rejection rate is

```
(j-n)/j
```

```
## [1] 0.5023638
```

It is close to the theoretical value $\alpha = 0.5$.

Question 1c

To apply the sampling importance resampling (SIR) method, we can use the uniform distribution on $[0, a]$ as the sampling distribution. The unstandardized weights are then

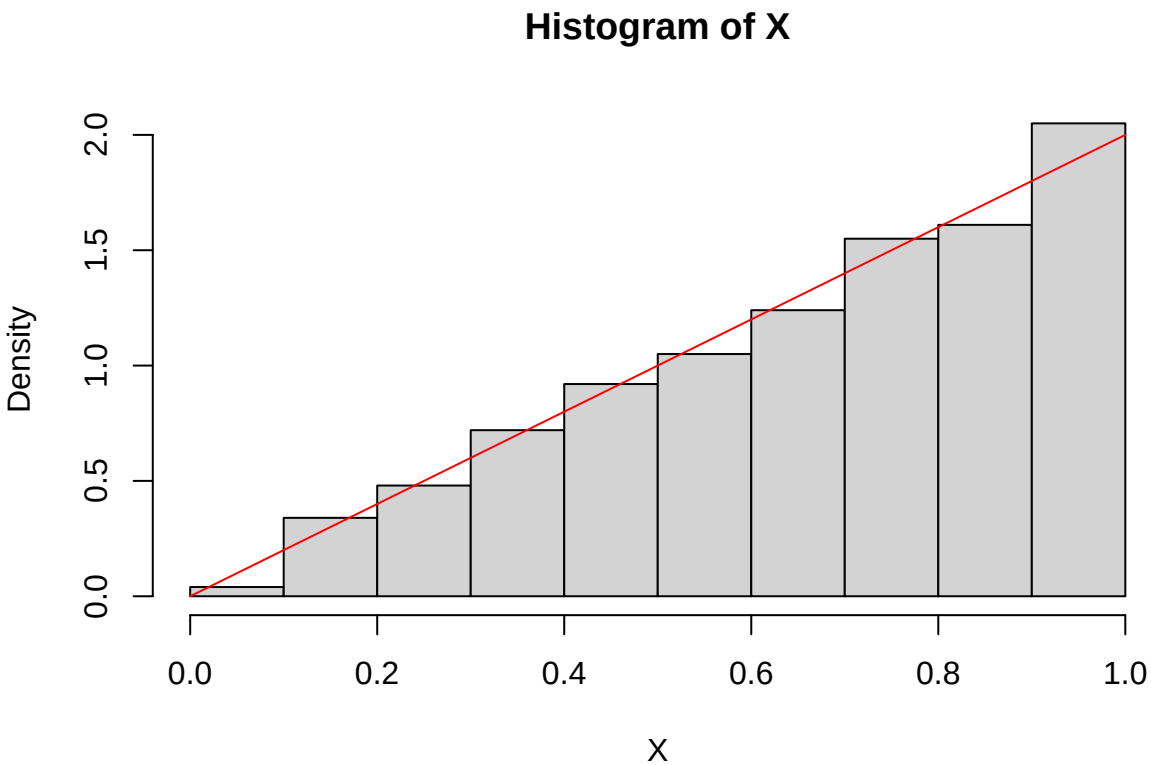
$$w^*(Y) = \frac{f(Y)}{g(Y)} = \frac{2}{a^2} Y a = \frac{2Y}{a}.$$

To generate a sample of size n, we need a primary sample of size $m \geq 10n$:

```
n<-1000
m <- 10*n
Y <- runif(m,0,a)
w <- 2*Y/a
w<- w/sum(w)
X <- sample(Y,size=n,replace=TRUE,prob=w)
```

Let us plot the histogram with the population density:

```
hist(X,freq=FALSE)
lines(c(0,1),c(0,2),col="red")
```



Question 1d

Here, the PIT method is obviously the best one. To obtain a sample of size n, we only need to generate n values from the standard uniform distribution using this method, against $4n$ values on average using the RS method, and at least $10n$ values for the SIR method (which is only approximate). When it is applicable, the PIT method is always the best one. However, the RS and SIR methods have wider applicability.

Exercise 2

Question 2a

We first write a function that computes the likelihood for an i.i.d. sample from the Poisson distribution:

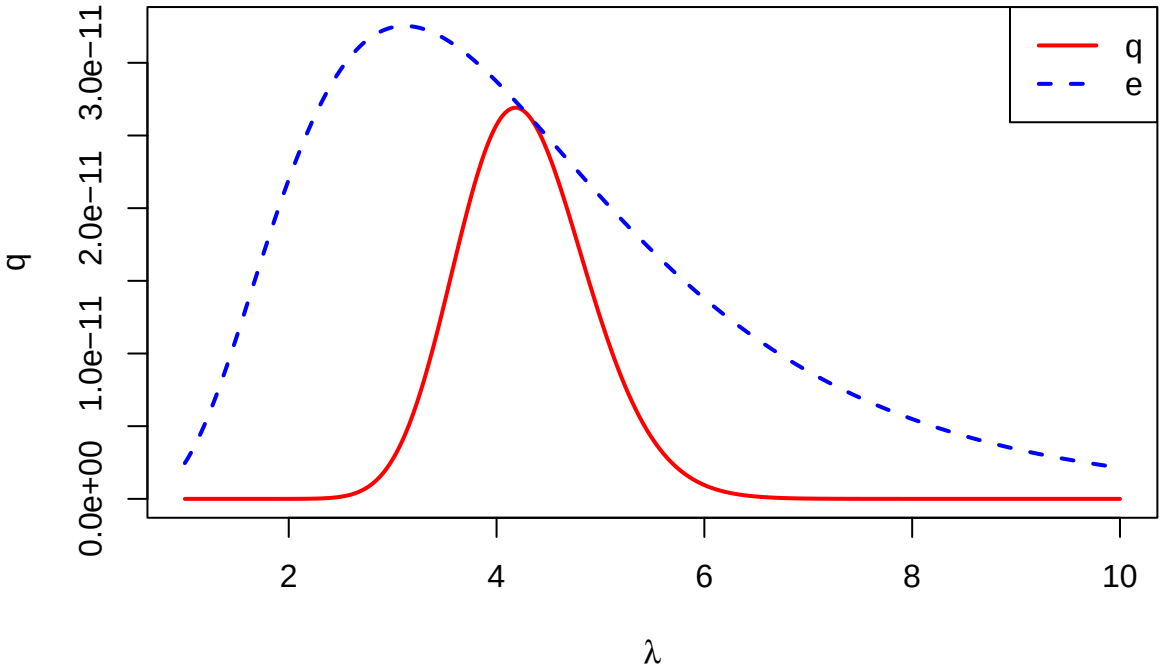
```
lik <- function(lambda,x) return(prod(sapply(x,dpois,lambda=lambda)))
```

We then compute functions q and e for the range $\lambda \in [1, 10]$:

```
x<- c(8,3,4,3,1,7,2,6,2,7)
meanlog<-log(4)
sdlog<-0.5
lambda<-seq(from=1,to=10,by=0.01)
L<-sapply(lambda,lik,x=x) # Likelihood
prior<-sapply(lambda,dlnorm,meanlog = meanlog, sdlog = sdlog) # Prior
q<-L*prior
e<-lik(mean(x),x)*prior
```

We plot the two curves:

```
plot(lambda,q,type="l",ylim=range(q,e),col="red",lwd=2,xlab=expression(lambda))
lines(lambda,e,lty=2,col="blue",lwd=2)
legend("topright",legend=c("q","e"),col=c("red","blue"),lty=c(1,2),lwd=c(2,2))
```



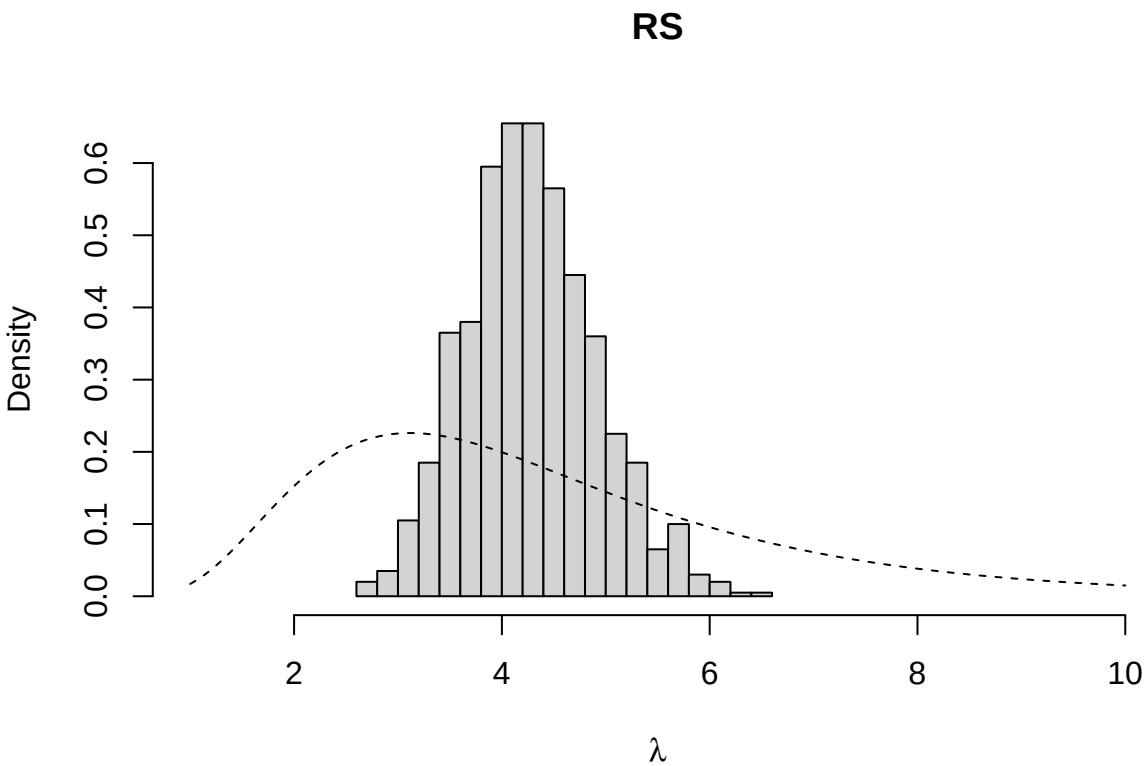
Question 2b

This is an implementation of the rejection sampling algorithm:

```
n<-1000  # desired sample size
Lambda<-vector(n,mode="numeric")
i<-0
likh<-lik(mean(x),x) # maximum likelihood
while(i<n){
    l<-rlnorm(1,meanlog = meanlog, sdlog = 0.5)
    u <- runif(1)
    if(u<lik(l,x)/likh){
        i<-i+1
        Lambda[i]<-l
    }
}
```

We plot the histogram of generated values together with the prior density:

```
h<-hist(Lambda,breaks=20,plot=FALSE)
ylim=range(0,prior,h$density)
hist(Lambda,freq=FALSE,breaks=20,ylim=ylim,xlim=range(lambda),
     xlab=expression(lambda),main="RS")
lines(lambda,prior,lty=2)
```



Question 2c

```
Linf=mean(Lambda)-sd(Lambda)*qnorm(0.975)/sqrt(n)
Lsup=mean(Lambda)+sd(Lambda)*qnorm(0.975)/sqrt(n)
print(c(Linf,Lsup))
```

```
## [1] 4.244516 4.322023
```

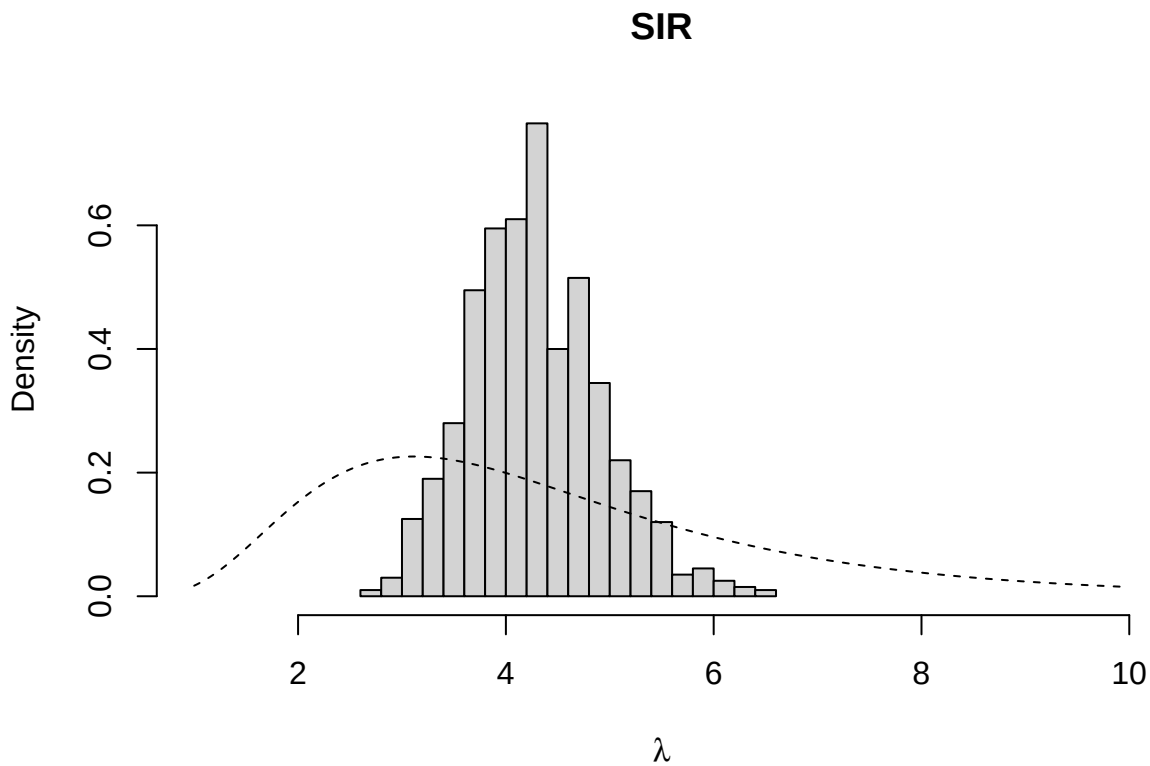
Question 2d

R implementation of the SIR algorithm:

```
m<-50000
n<-1000
Lambda0<-rlnorm(m,meanlog = meanlog, sdlog = sdlog) # sample from prior
L<-sapply(Lambda0,lik,x=x)
w<-L/sum(L) # standardized importance weights
Lambda1<-sample(Lambda0,size=n,replace=TRUE,prob=w) # resampling
```

Plot of the histogram together with the prior density:

```
h<-hist(Lambda1,breaks=20,plot=FALSE)
ylim=range(0,prior,h$density)
hist(Lambda1,freq=FALSE,breaks=20,ylim=ylim,xlim=range(lambda),
     xlab=expression(lambda),main="SIR")
lines(lambda,prior,lty=2)
```



We can see that the obtained distribution is very similar to the one found using the rejection sampling algorithm. The confidence interval is close to the previous one:

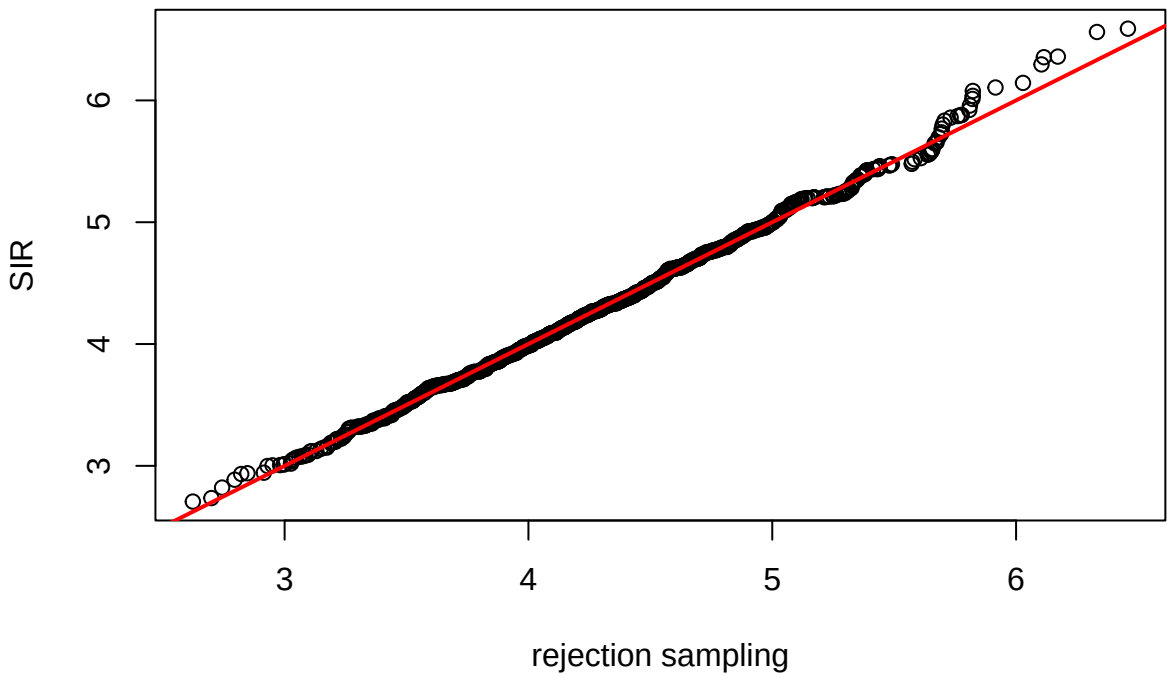
```
Linf1=mean(Lambda1)-sd(Lambda1)*qnorm(0.975)/sqrt(n)
Lsup1=mean(Lambda1)+sd(Lambda1)*qnorm(0.975)/sqrt(n)
print(c(Linf1,Lsup1))
```

```
## [1] 4.245822 4.324187
```

Question 2e

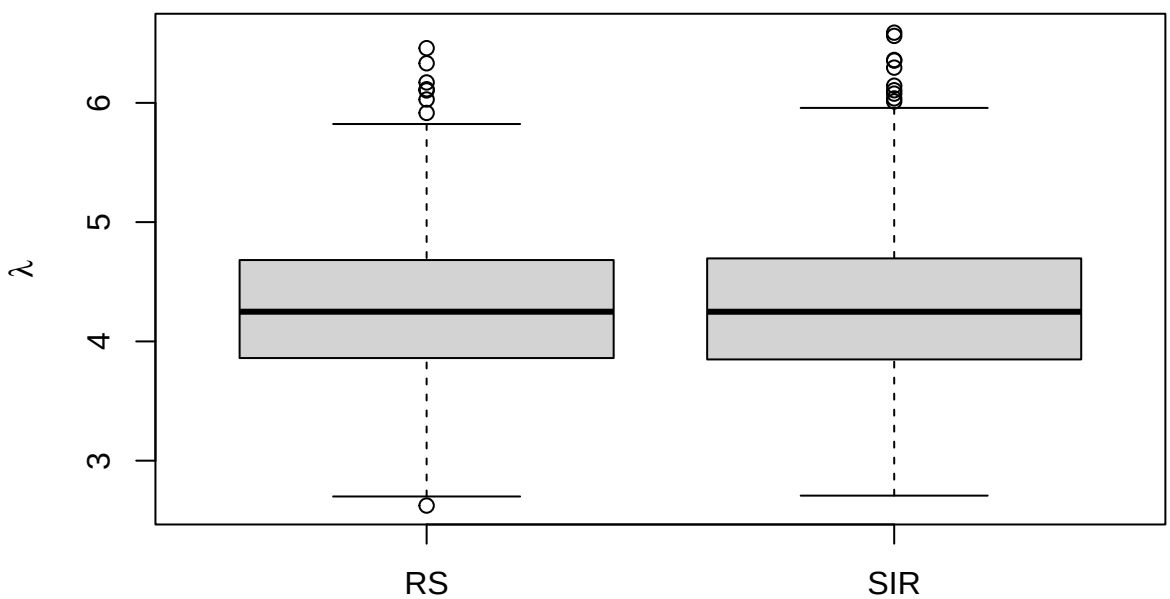
We can compare the two distributions graphically using a QQ-plot:

```
qqplot(Lambda,Lambda1,xlab="rejection sampling",ylab="SIR")
abline(0,1,col="red",lwd=2)
```



We can also plot boxplots of the two distributions side by side:

```
boxplot(Lambda,Lambda1,ylab=expression(lambda),names=c("RS","SIR"))
```



Exercise 3

Question 3a

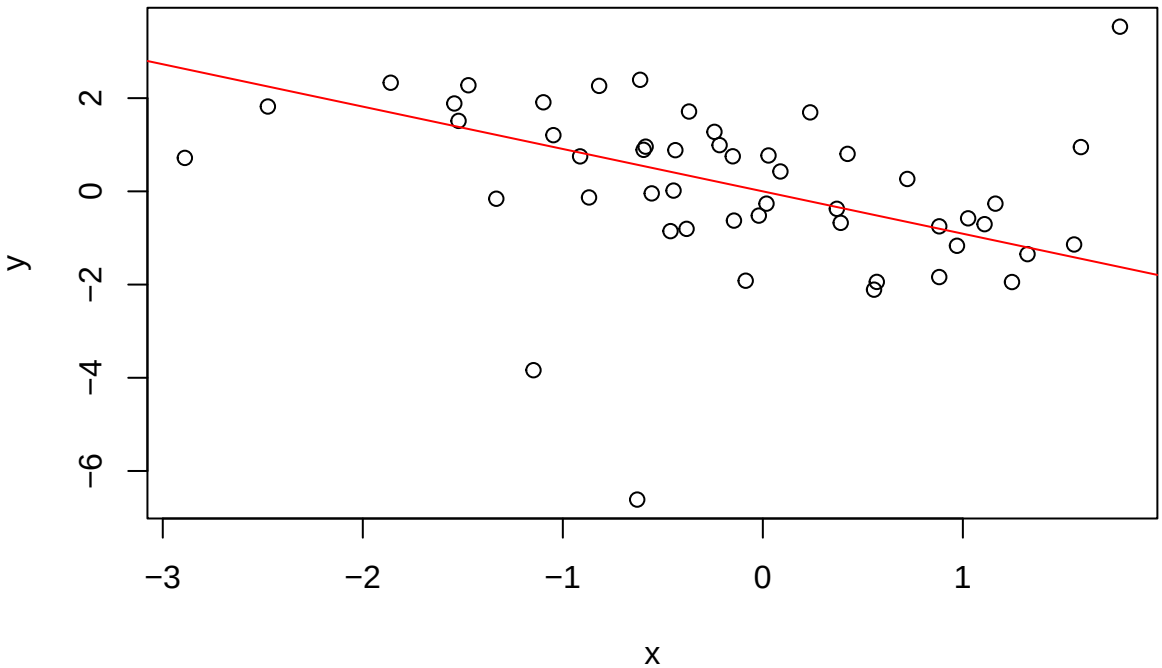
We generate a dataset of size $n = 50$:

```
set.seed(20)
n<-50
```

```
x<-rnorm(n)
beta<-rcauchy(1,location=0,scale=2)
y<-beta*x+rt(n,df=2)
```

plot of the data with the regression line:

```
plot(x,y)
abline(0,beta,col='red')
```



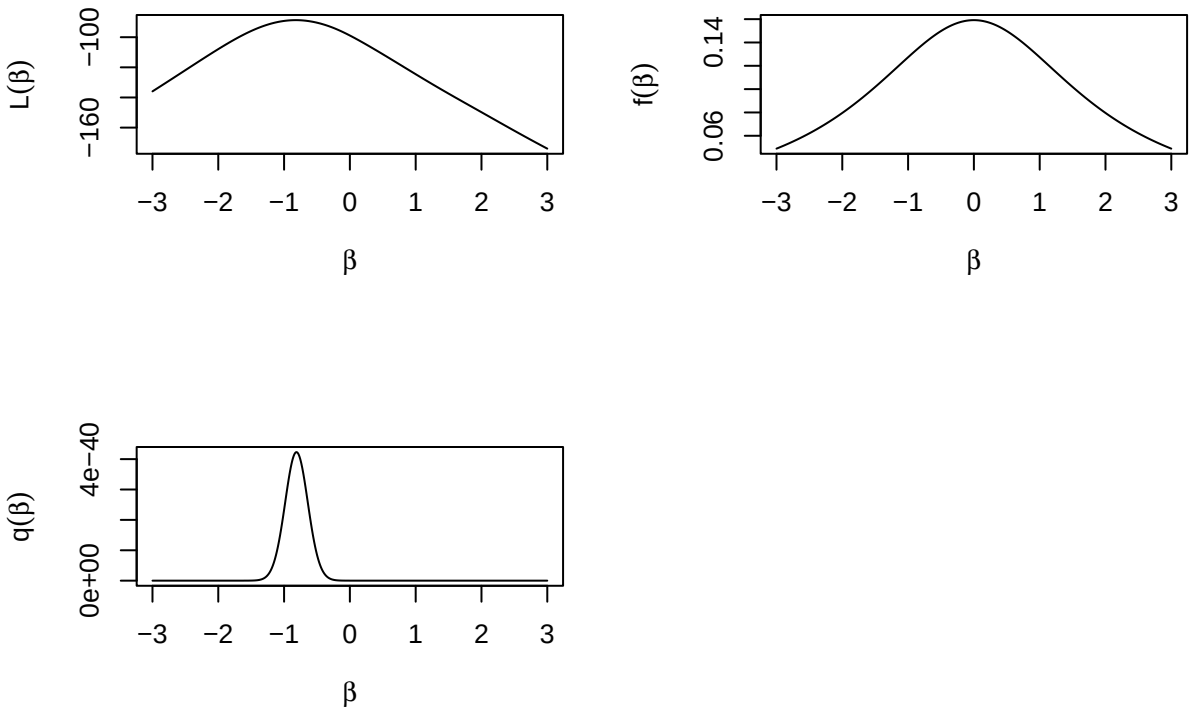
Question 3b

We first write a function that computes the log-likelihood:

```
loglik<- function(beta,x,y) return(sum(log(dt(y-beta*x,df=2))))
```

We then plot the three functions:

```
BETA<-seq(-3,3,0.001)
LOGL<-sapply(BETA,loglik,x,y)
par(mfrow=c(2,2))
plot(BETA,LOGL,type="l",xlab=expression(beta),ylab=expression(L(beta)))
plot(BETA,dcauchy(BETA,0,2),type="l",
     xlab=expression(beta),ylab=expression(f(beta)))
plot(BETA,exp(LOGL)*dcauchy(BETA,0,2),type="l",
     xlab=expression(beta),ylab=expression(q(beta)))
par(mfrow=c(1,1))
```



Question 3c

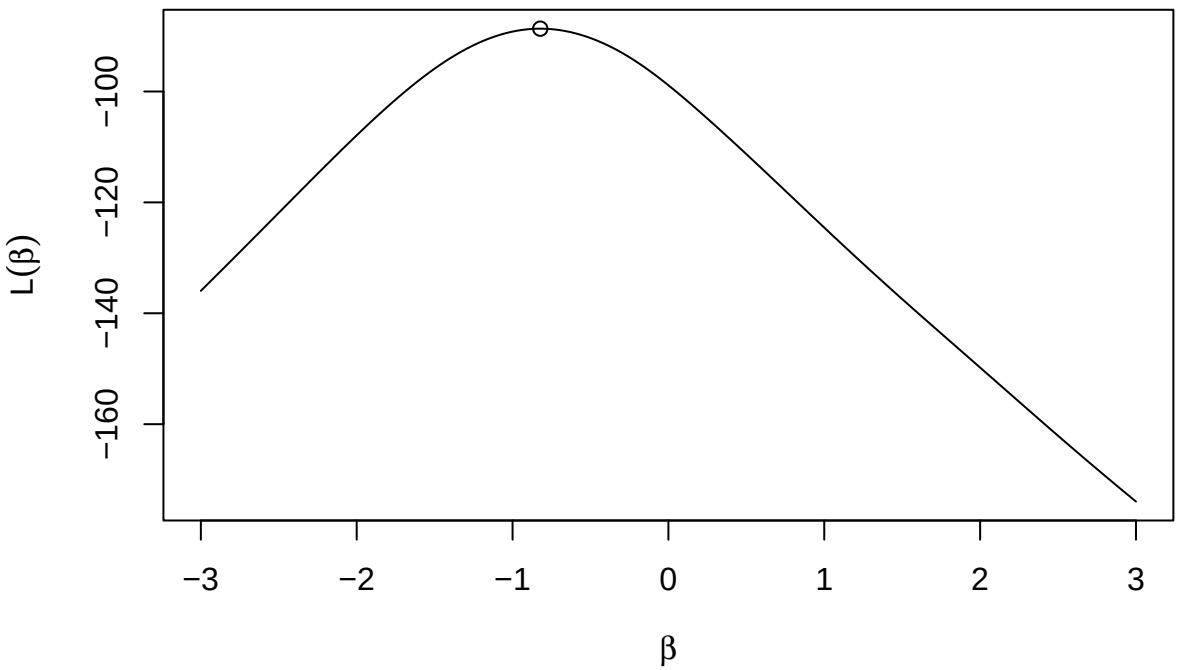
We use the R function `optimize`:

```
opt<-optimize(loglik,c(-2,0),x,y,maximum=TRUE)
opt
```

```
## $maximum
## [1] -0.8221405
##
## $objective
## [1] -88.66904
```

We plot the results:

```
plot(BETA,LOGL,type="l",xlab=expression(beta),ylab=expression(L(beta)))
points(opt$maximum,opt$objective)
```




```
betah<-opt$maximum
```

Question 3d

Let us first use rejection sampling:

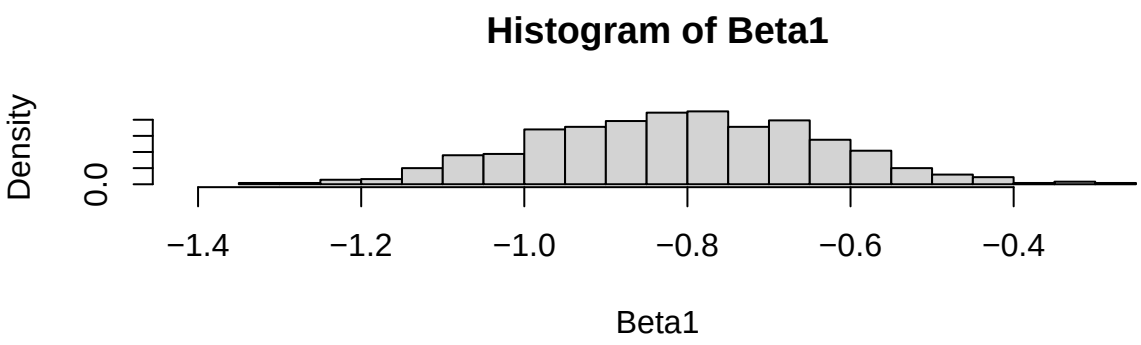
```
N<-1000
Beta1<-vector(N,mode="numeric")
i<-0
while(i<N){
  b<-rcauchy(1,location=0,scale=2)  # sample from prior
  u <- runif(1)
  if(u<exp(loglik(b,x,y)-loglik(betah,x,y))){
    i<-i+1
    Beta1[i]<-b
  }
}
```

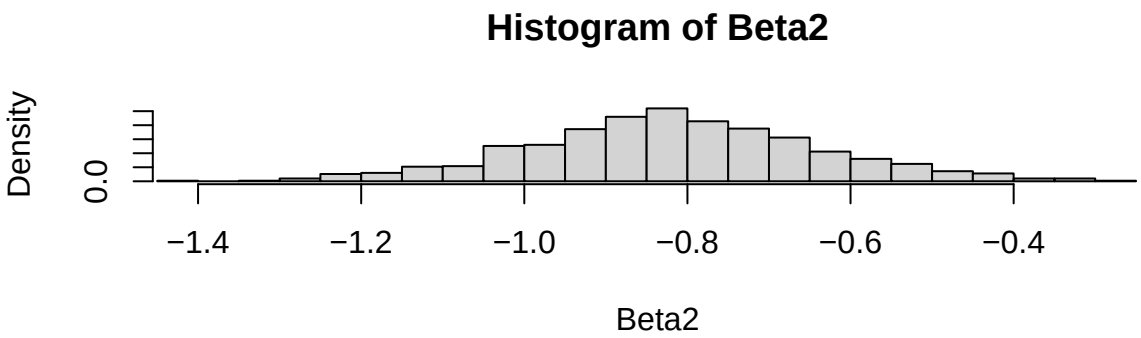
Then we use SIR:

```
m<-50000
Beta0<-rcauchy(m,location=0,scale=2) # sample from prior
L<-sapply(Beta0,loglik,x,y)
w<-exp(L)/sum(exp(L)) # Standardized importance weight
Beta2<-sample(Beta0,size=N,replace=TRUE,prob=w)
```

Plot of the two histograms side by side:

```
par(mfrow=c(2,1))
h1<-hist(Beta1,freq=FALSE,breaks=20,xlim=range(Beta1,Beta2))
h2<-hist(Beta2,freq=FALSE,breaks=20,xlim=range(Beta1,Beta2))
```






```
par(mfrow=c(1,1))
```

QQ-plot:

```
qqplot(Beta1,Beta2)
abline(0,1,col='red')
```