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

Thierry Denoeux

2023-01-05

Exercise 1

Question 1a

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

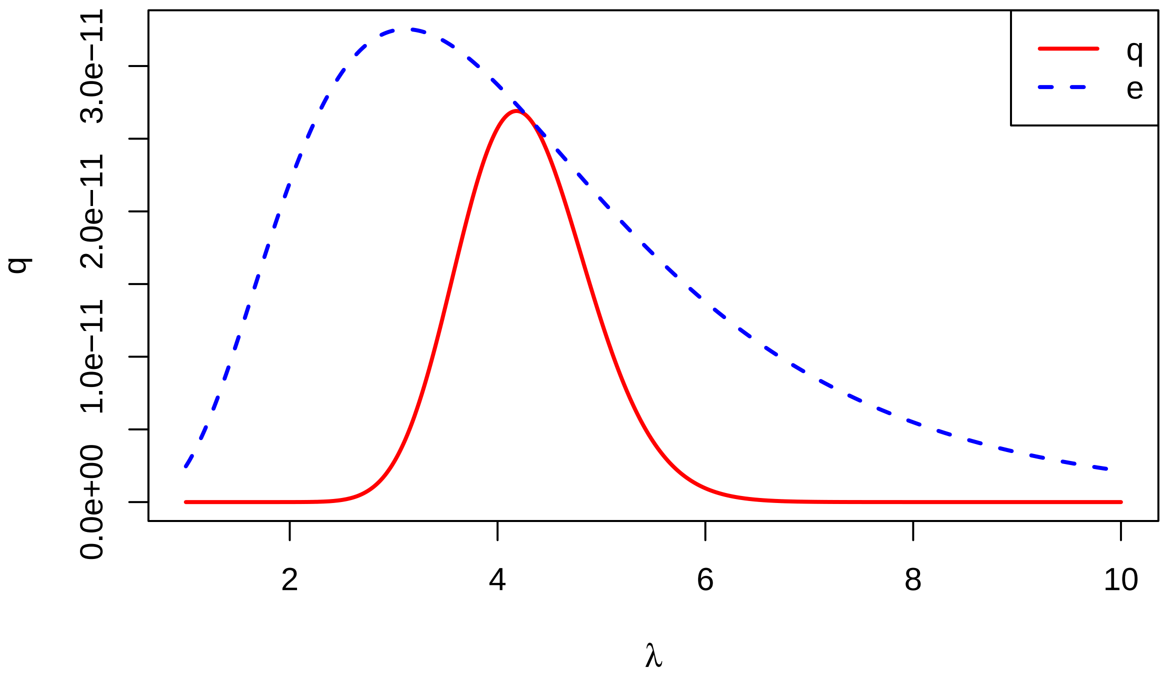
```
lik <- function(lambda,x) return(prod(sapply(x,dpois,lambda=lambda)))
```

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))
```



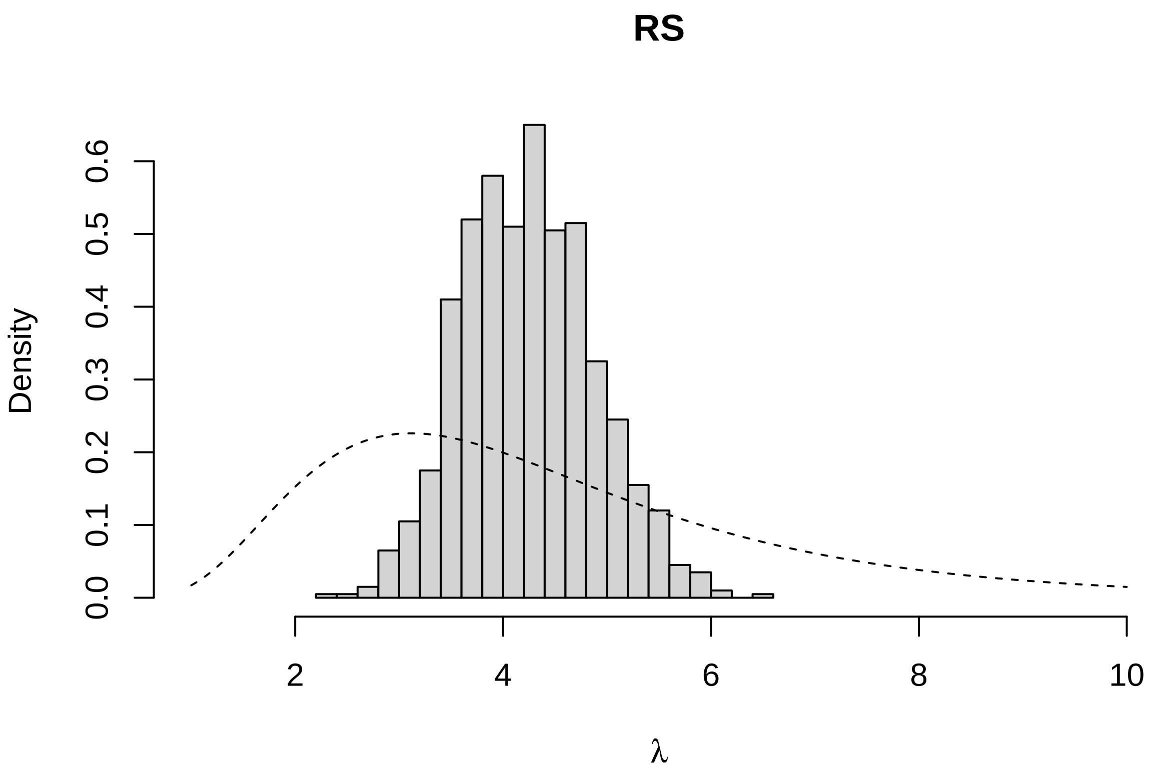
Question 1b

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 1c

```r
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] 4.208324 4.286728
```

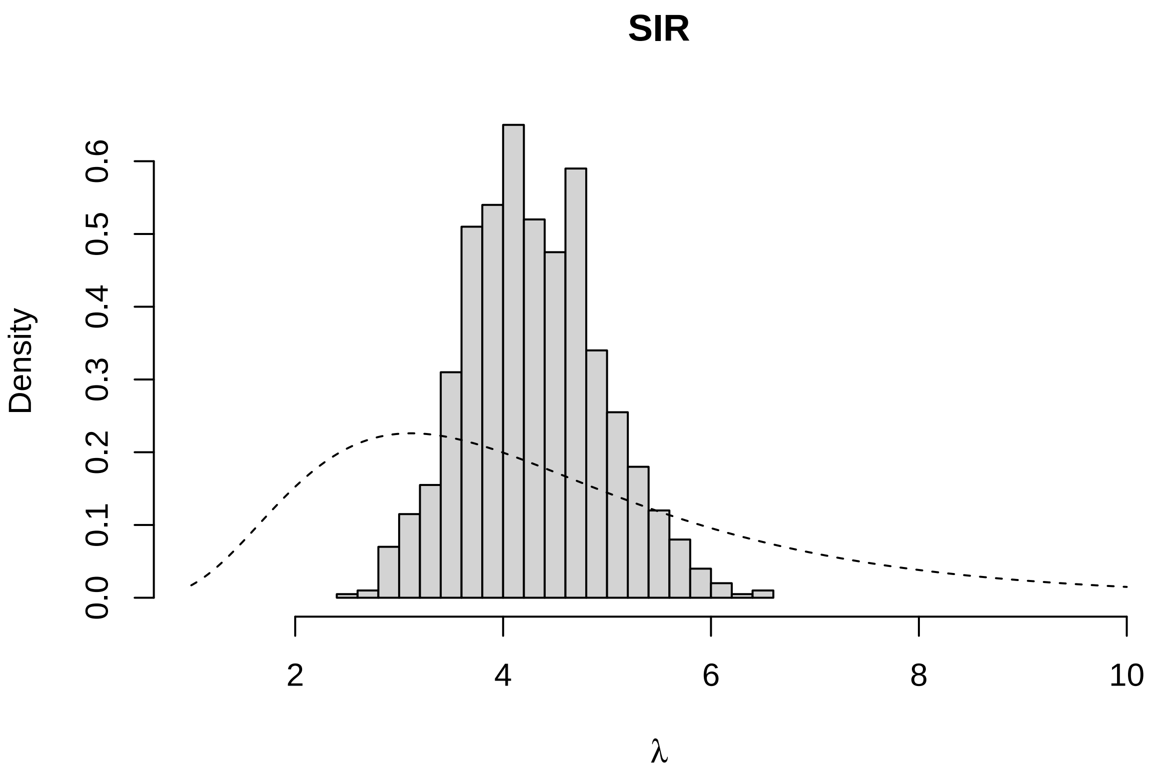
Question 1d

R implementation of the SIR algorithm:

```r
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:

```r
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:

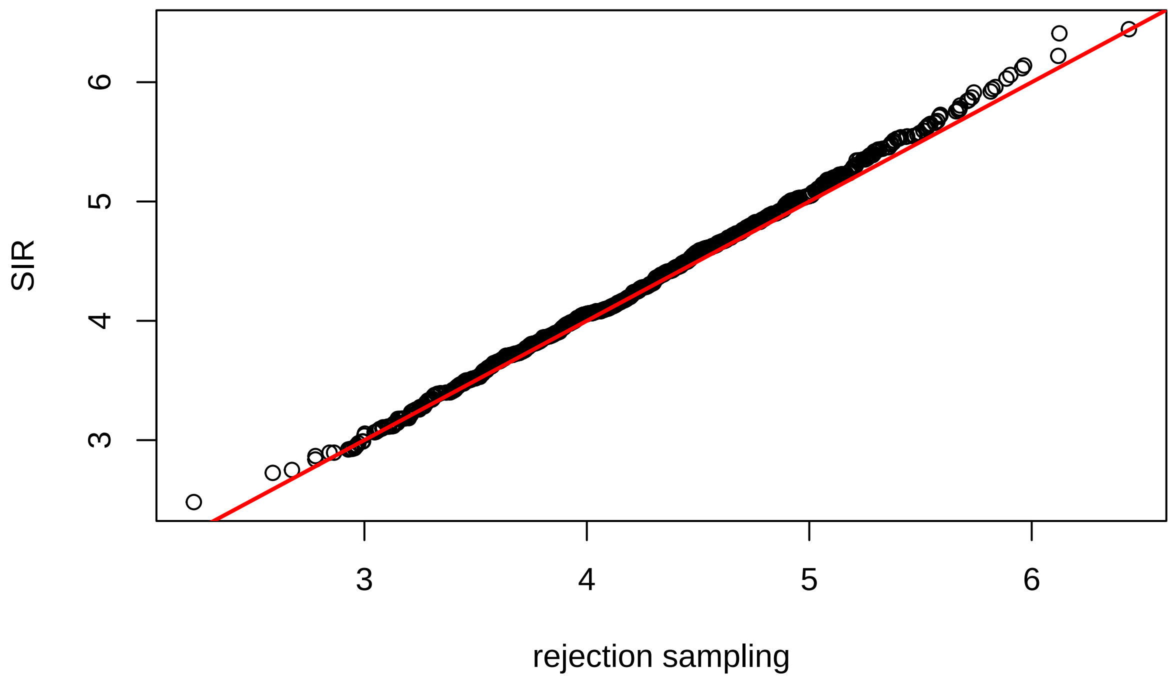
```
Linf1=mean(Lambda1)-sd(Lambda1)*qnorm(0.975)/sqrt(n)
Lsup1=mean(Lambda1)+sd(Lambda1)*qnorm(0.975)/sqrt(n)
print(c(Linf1,Lsup1))
```

```
## [1] 4.257473 4.338179
```

Question 1d

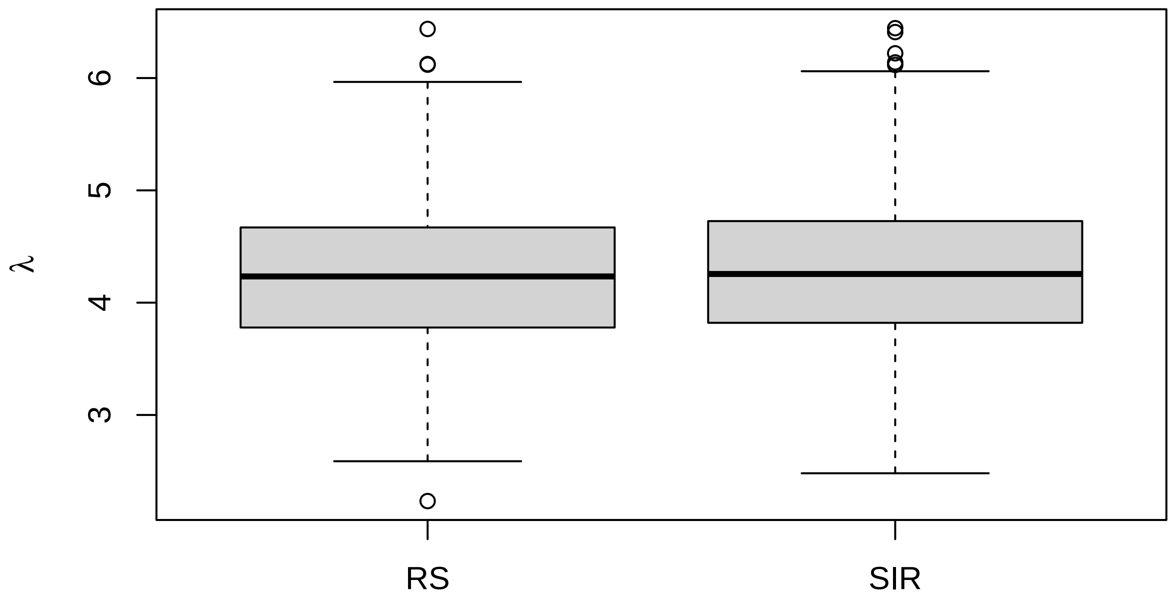
We can compare the two distributions graphically using a QQ-plot:

```
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 2

Question 2a

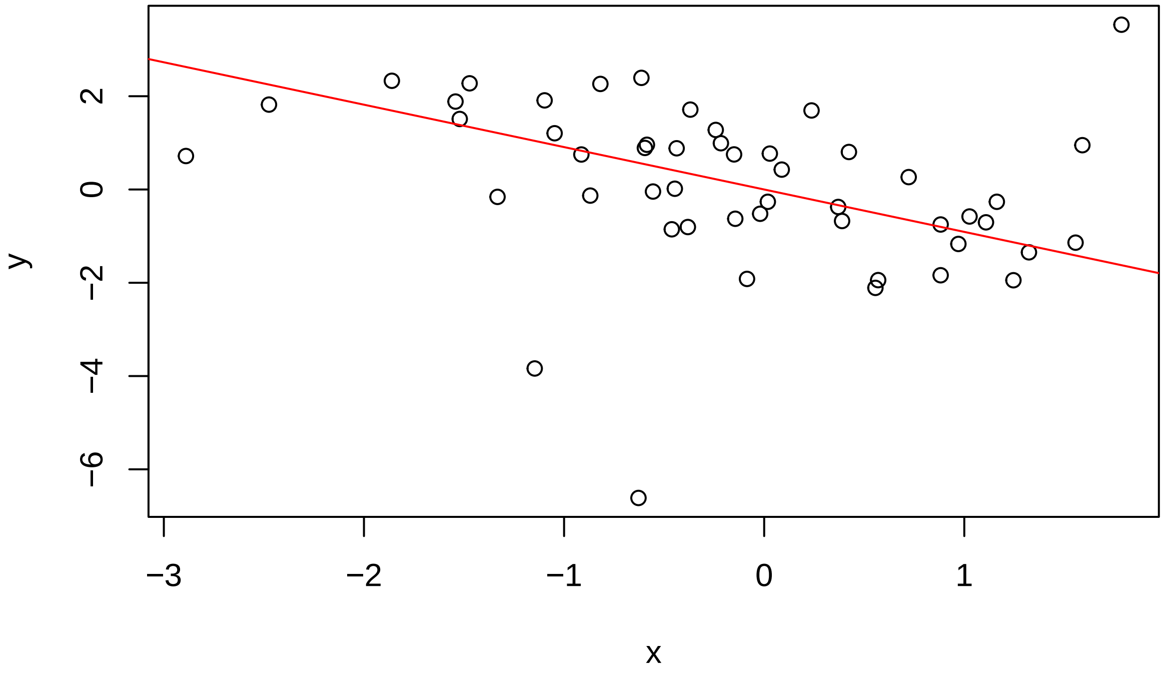
We generate a dataset of size $n = 50$:

```
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')
```



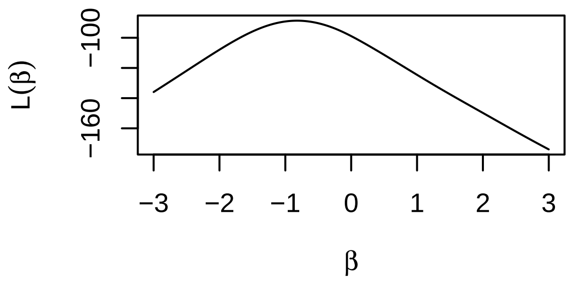
Question 2b

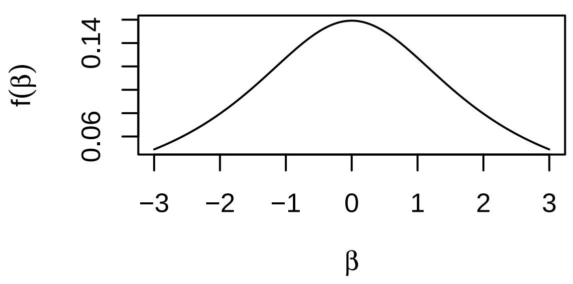
We first write a function that computes the log-likelihood:

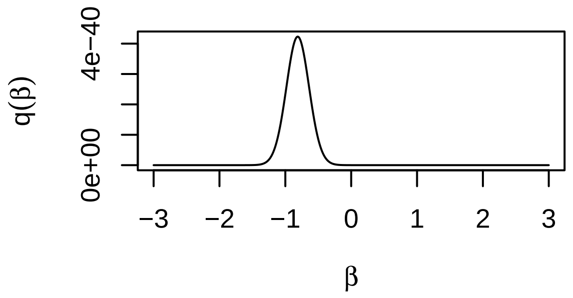
```
loglik<- function(beta,x,y) return(sum(log(dt(y-beta*x,df=2))))
```

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 2c

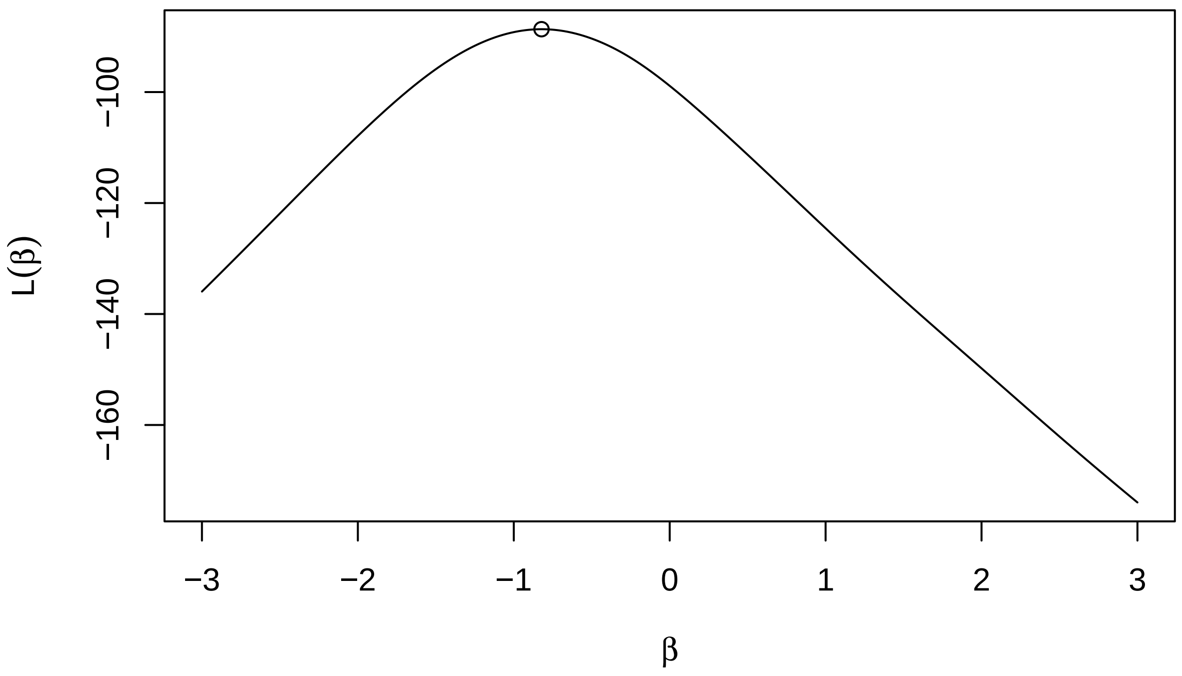
We use the R function `optimize`:

```
opt<-optimize(loglik,c(-5,0),x,y,maximum=TRUE)
opt
```

```
## $maximum
## [1] -0.8221408
##
## $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 2d

Let us first use rejection sampling:

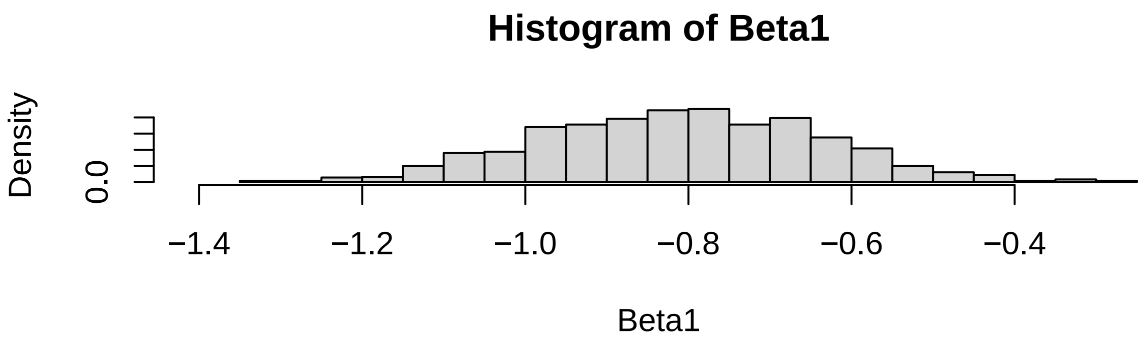
```
N<-1000
Beta1<-vector(N,mode="numeric")
i<-0
while(i<N){
  b<-rcauchy(1,location=0,scale=2)  # sample from prior
  u <- runif(1)
  if(u<exp(loglik(b,x,y)-loglik(betah,x,y))){
    i<-i+1
    Beta1[i]<-b
  }
}
```

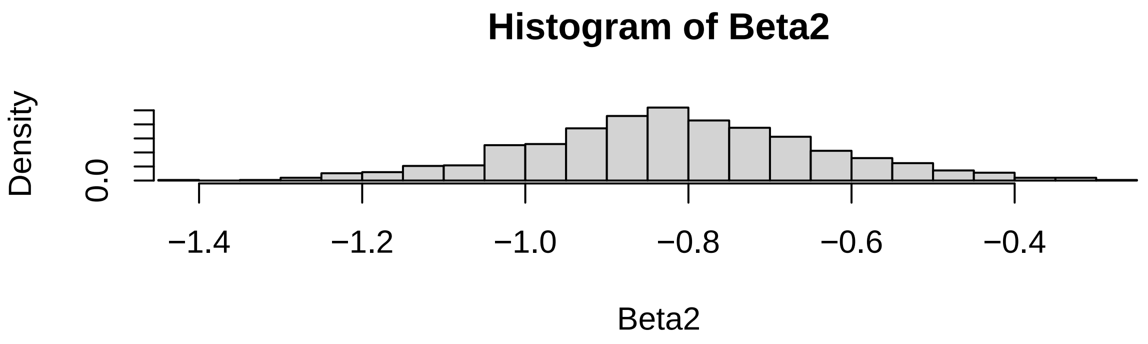
Then we use SIR:

```
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))
```

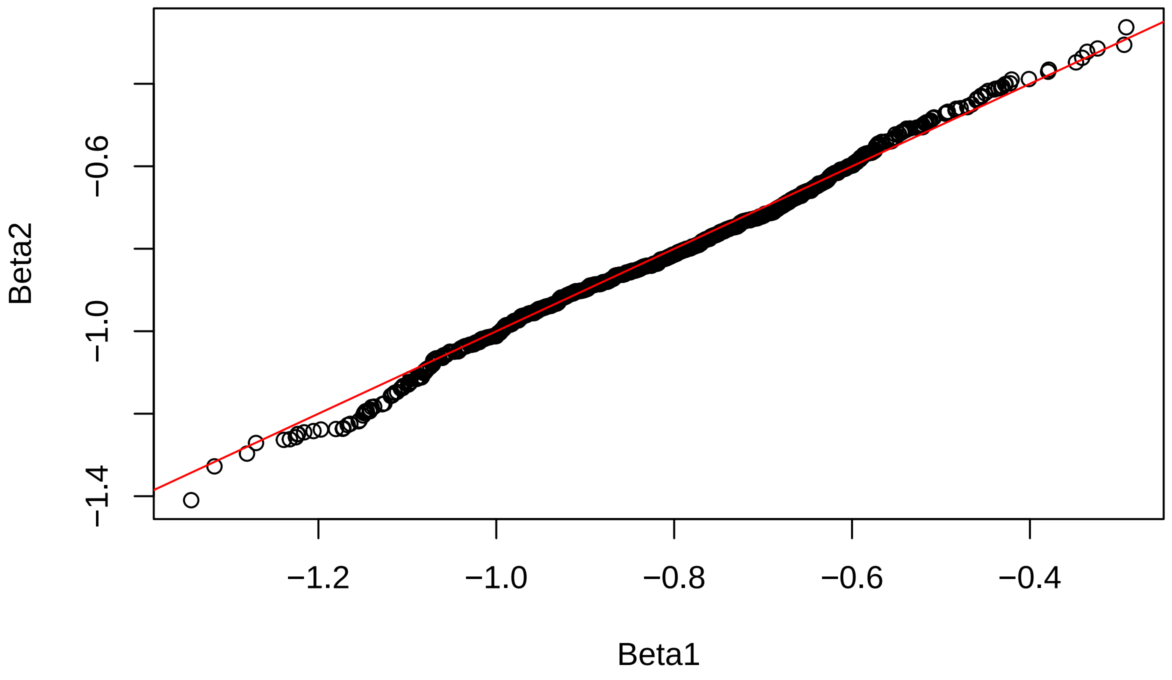




```
par(mfrow=c(1,1))
```

QQ-plot:

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



Question 2e

To get 95% credibility intervals, we simply compute the 2.5% and 97.5% quantiles of the posterior distribution of β. For the sample generated by the rejection sampling method, we get:

```
print(quantile(Beta1,c(0.025,0.975)),2)
```

```
##  2.5% 97.5%
## -1.14 -0.47
```

With the SIR method, we get:

```
print(quantile(Beta2,c(0.025,0.975)),2)
```

```
##  2.5% 97.5%
## -1.19 -0.45
```

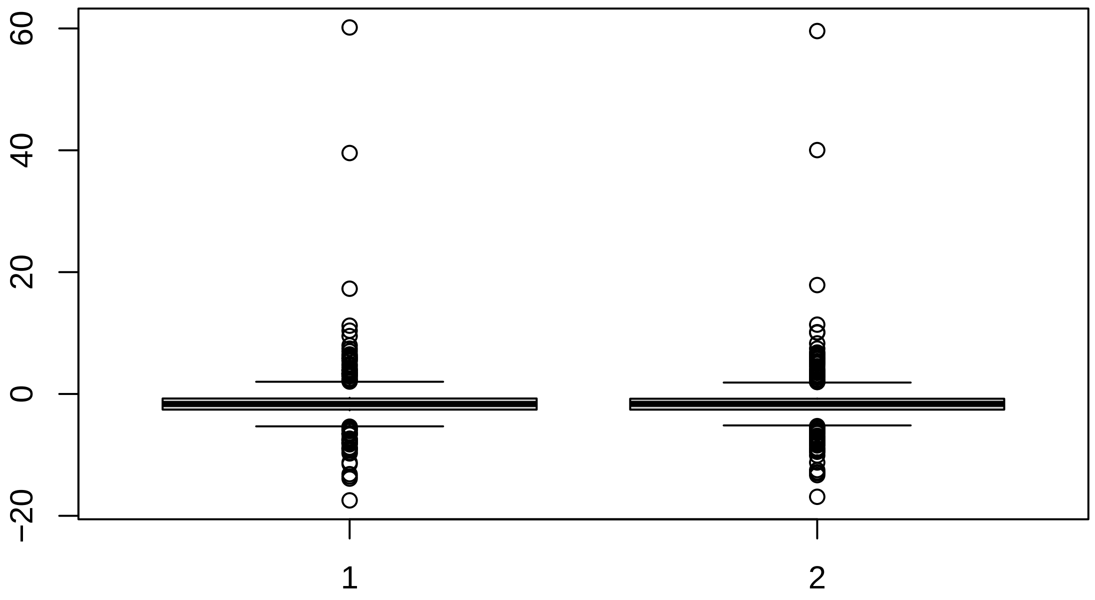
Question 2f

To generate an observation from the predictive distribution of $y_0 = \beta x_0 + u$, we independently draw β from its posterior distribution, and u from the error distribution:

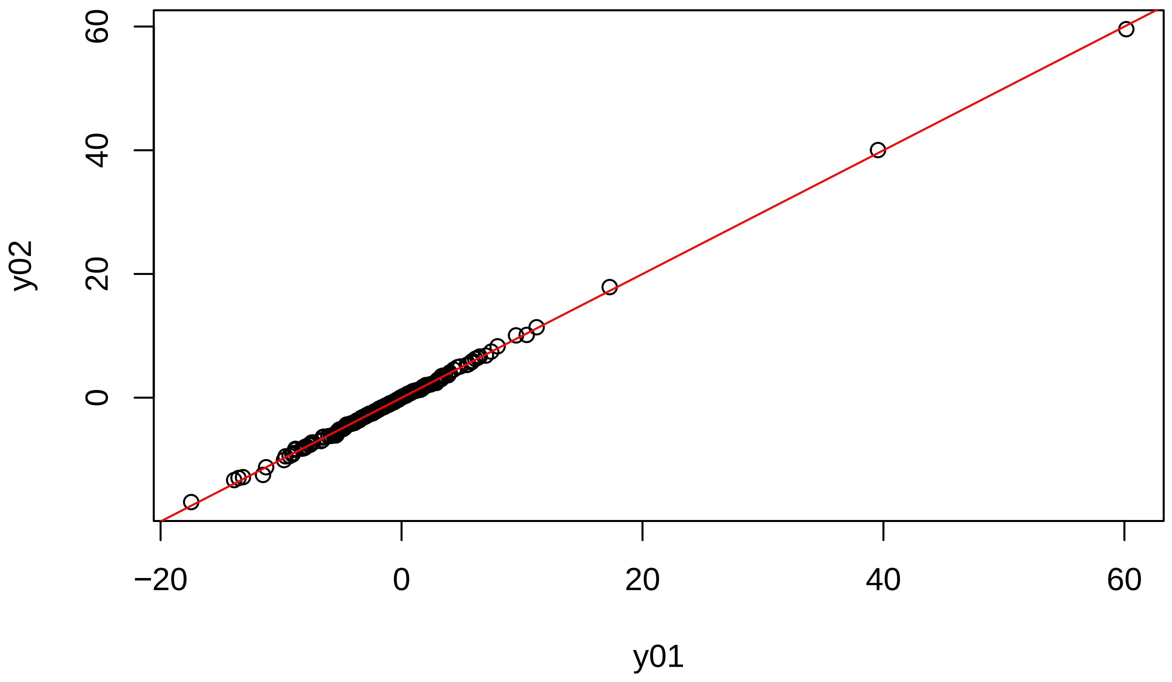
```
x0<-2
u<-rt(N,df=2)
y01<-Beta1*x0+u
y02<-Beta2*x0+u
```

We compare the two samples of size $N = 1000$ obtained from the two methods:

```
boxplot(y01,y02)
```



```
qqplot(y01,y02)
abline(0,1,col='red')
```


y02
y01
−20
0
20
40
60
0
20
40
60