lab2_danhe178_rical803

Daniel Herzegh & Richard Friberg

2017-10-03

Uppgift 1 Likelihoodfunktioner

```
set.seed(4711)
x1 <- rgamma(n = 10, shape = 4, scale = 1)
x2 <- rgamma(n = 100, shape = 4, scale = 1)
```

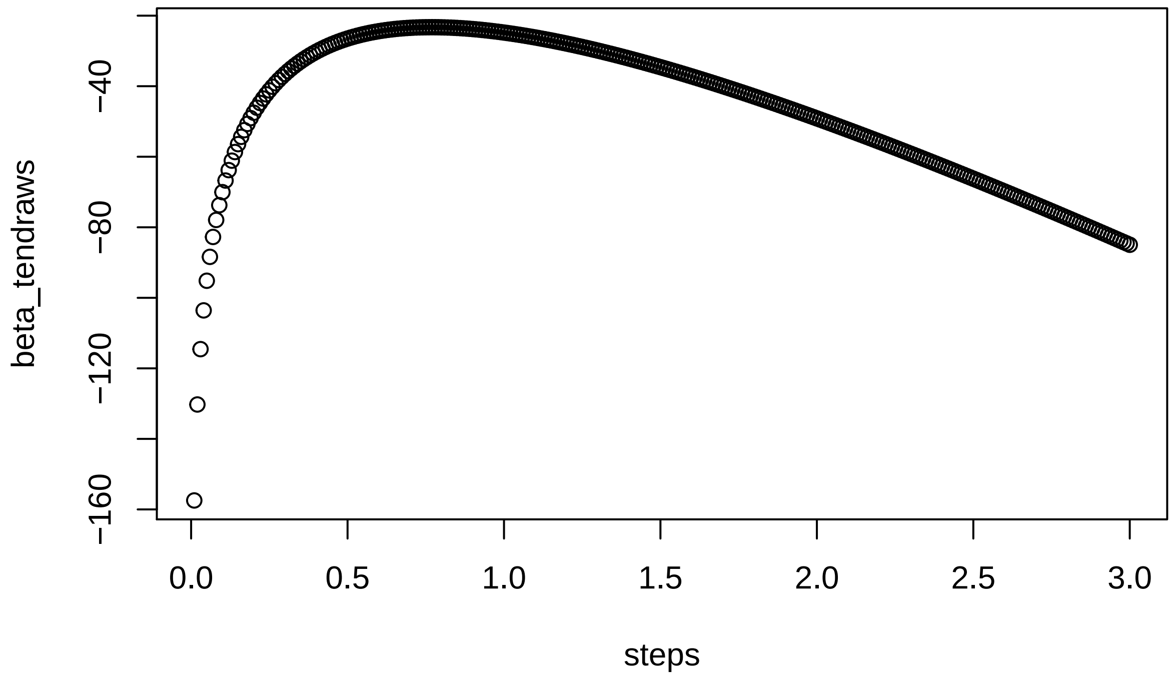
a)

```
llgamma <- function(x, alpha, beta) {
  return(length(x) * (alpha * log(beta) - lgamma(alpha)) + (alpha -1) * sum(log(x)) - beta * sum(x))
}
```

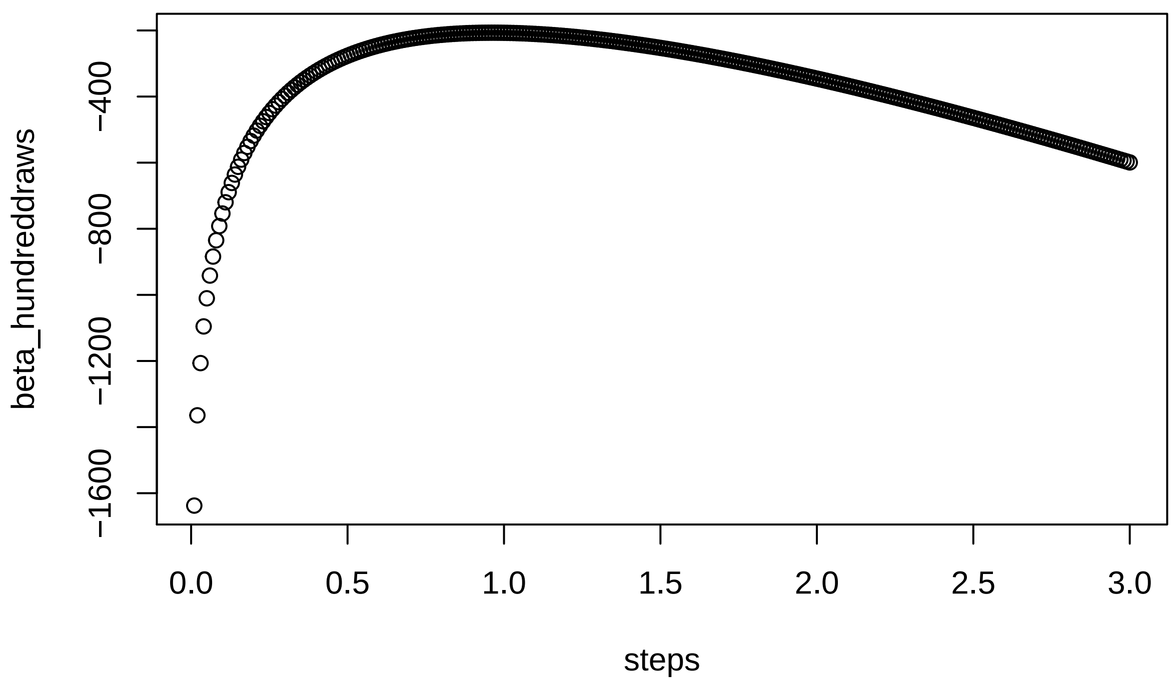
b)

```
beta_tendraws <- c()
beta_hundreddraws <- c()
steps <- c()

i = 0.01
while(i <= 3) {
  beta_tendraws <- c(beta_tendraws, llgamma(x1, alpha = 4, beta = i))
  beta_hundreddraws <- c(beta_hundreddraws, llgamma(x2, alpha = 4, beta = i))
  steps <- c(steps, i)
  i <- i + 0.01
}
```

```
# plot for ten draws
plot(steps, beta_tendraws)
```




```
# plot for hundred draws
plot(steps, beta_hundreddraws)
```




```
findMax <- function(vect) {
  i <- NULL
  currentMax <- -Inf
  x <- 1
  while (x < length(vect)) {
    if (vect[x] > currentMax) {
      currentMax <- vect[x]
```

```
      i <- x
    }
    x <- x + 1
  }
  return(i/100)
}
```

```
# Undersöker och returnerar vilket betavärde som loglikelihoodfunktionen får sitt maxvärde på
findMax(beta_tendraws)
```

```
## [1] 0.77
```

```
findMax(beta_hundreddraws)
```

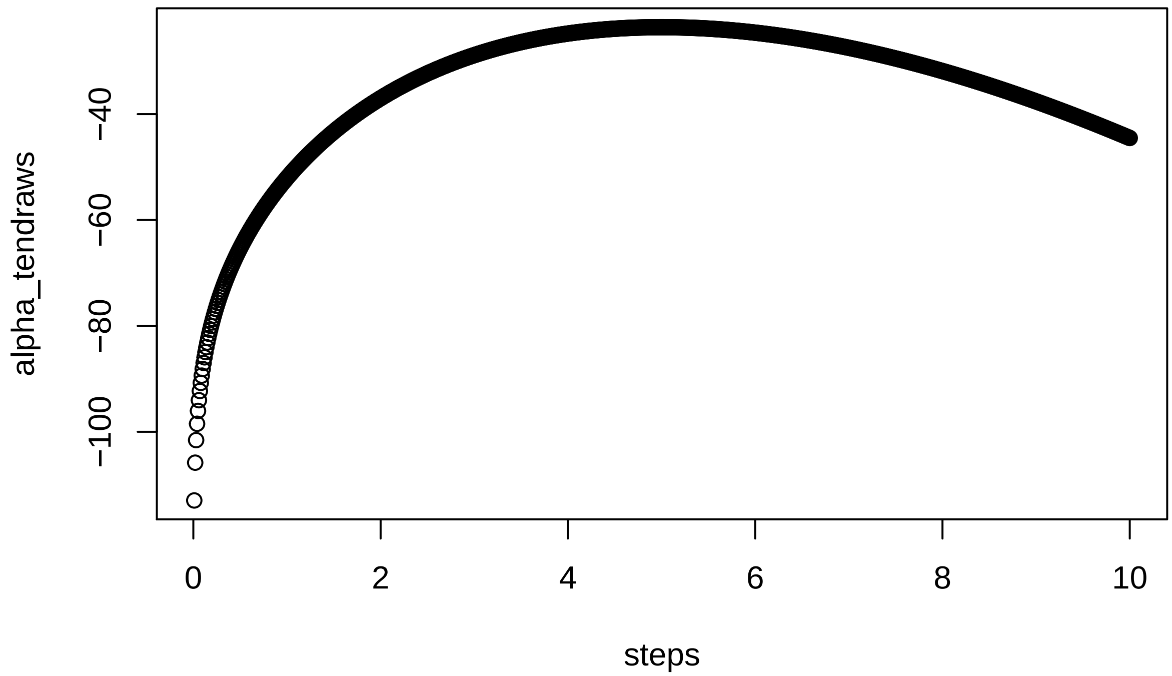
```
## [1] 0.96
```

Det varierar vilket av de upprepade värdena för beta som ger maximala värdet på loglikelihoodfunktionen, men ökar man antalet dragningar går denna siffra mot 1.0.

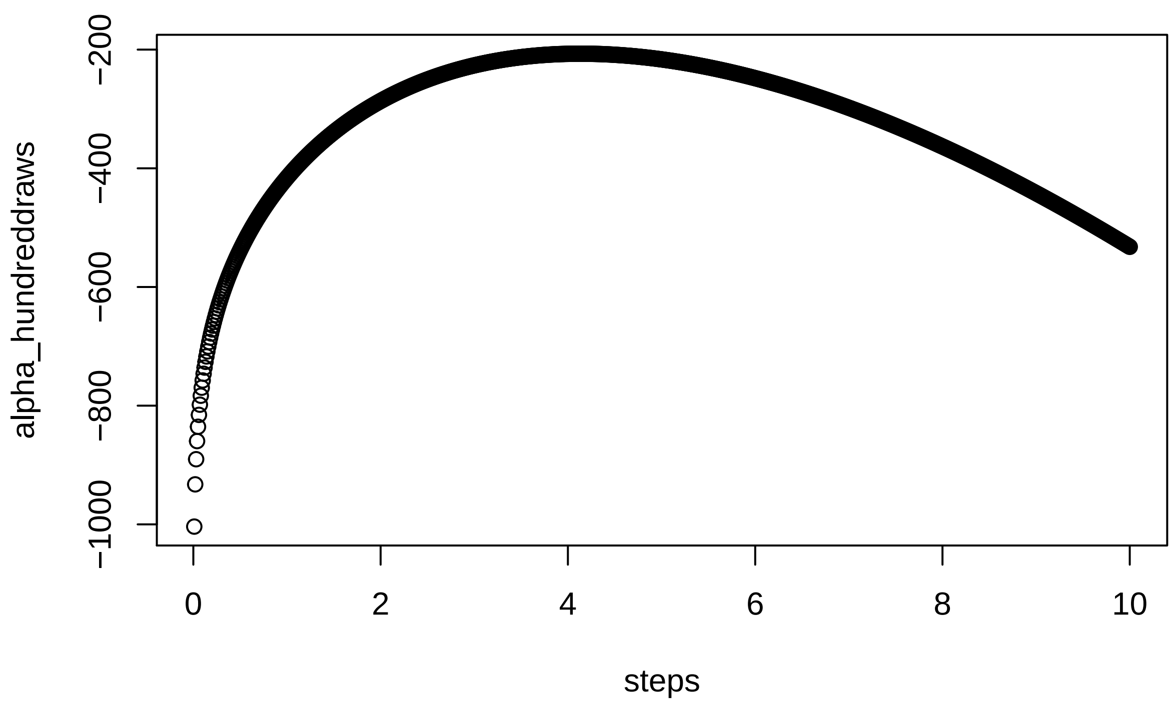
c)

```
alpha_tendraws <- c()
alpha_hundreddraws <- c()
steps <- c()

i = 0.01
while(i <= 10) {
  alpha_tendraws <- c(alpha_tendraws, llgamma(x1, alpha = i, beta = 1))
  alpha_hundreddraws <- c(alpha_hundreddraws, llgamma(x2, alpha = i, beta = 1))
  steps <- c(steps, i)
  i <- i + 0.01
}
```

```
# plot for ten draws
plot(steps, alpha_tendraws)
```




```
# plot for hundred draws
plot(steps, alpha_hundreddraws)
```




```
# Undersöker och returnerar vilket alphavärde som loglikelihoodfunktionen får sitt maxvärde på

findMax(alpha_tendraws)
```

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

```r
findMax(alpha_hundreddraws)
```

```
## [1] 4.13
```

Det varierar vilket av de upprepade värdena för alpha som ger maximala värdet på loglikelihoodfunktionen, men ökar man antalet dragningar går denna siffra mot 4.0.

d)

Härledning av log-likelihood för normalfördelning:

1

$$L(\mu,\sigma^2) = \prod_{j=1}^{n} f(x_j \mid \mu,\sigma^2) = \prod_{j=1}^{n} \overbrace{(2\pi\sigma^2)^{-1/2}\, e\left(-\frac{1}{2}\frac{(x_j-\mu)^2}{\sigma^2}\right)}^{\text{PDF}}$$

$$= \underbrace{(2\pi\sigma^2)^{-n/2}\, e\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{n}(x_j-\mu)^2\right)}_{\text{likelihood för normalfördelning}}$$

Härledning av log-likelihood för normaldistribution:

$$\ln\overbrace{\left[(2\pi\sigma^2)^{-n/2}\exp\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{n}(x_j-\mu)^2\right)\right]}^{\text{likelihood-funktion för normaldistribution}} =$$

$$= \ln\left((2\pi\sigma^2)^{-n/2}\right) - \frac{1}{2\sigma^2}\sum_{j=1}^{n}(x_j-\mu)^2$$

$$= -\frac{n}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{j=1}^{n}(x_j-\mu)^2$$

$$= -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}\sum_{j=1}^{n}(x_j-\mu)^2$$

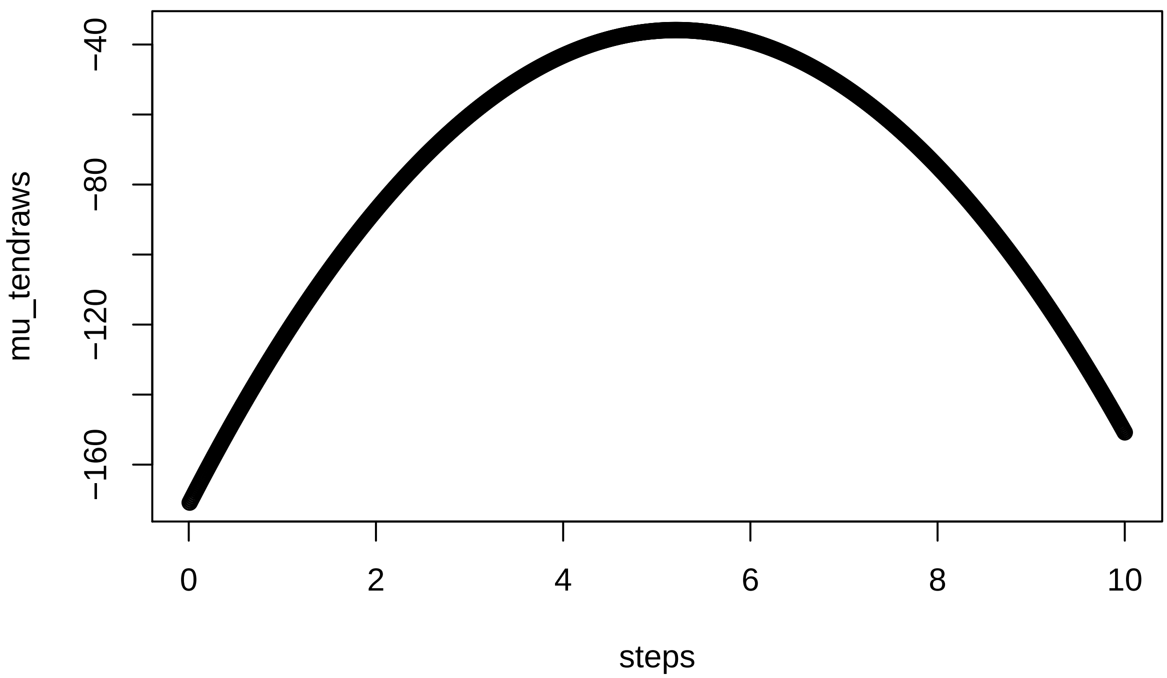
```r
llnormal <- function(x, mu, sigma2) {
  xsum <- sum((x - mu)**2)
  return(-length(x)/2*log(2*pi) - length(x)/2 * log(sigma2) - 1/(2 * sigma2) * xsum)
}

llnormal(x = x1, mu = 2, sigma2 = 1) #Fråga om okej
```

```
## [1] -87.25743
```

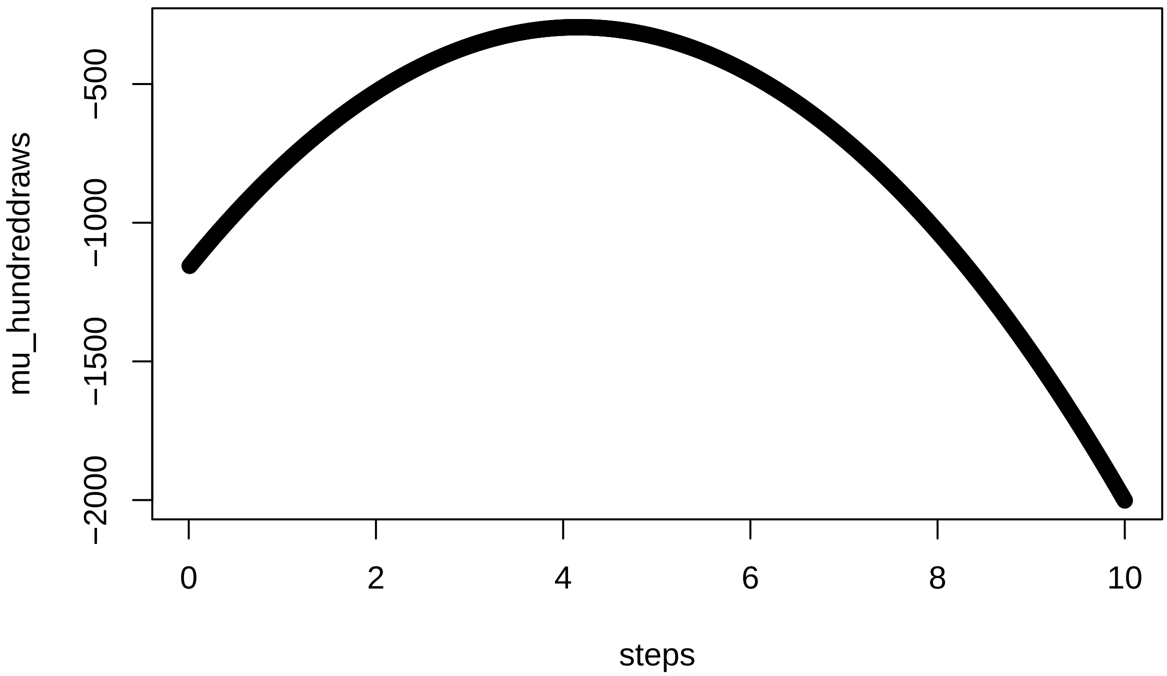
e)

```r
mu_tendraws <- c()
mu_hundreddraws <- c()
steps <- c()

i = 0.01
while(i <= 10) {
  mu_tendraws <- c(mu_tendraws, llnormal(x1, mu = i, sigma2 = 1))
  mu_hundreddraws <- c(mu_hundreddraws, llnormal(x2, mu = i, sigma2 = 1))
  steps <- c(steps, i)
  i <- i + 0.01
}
```

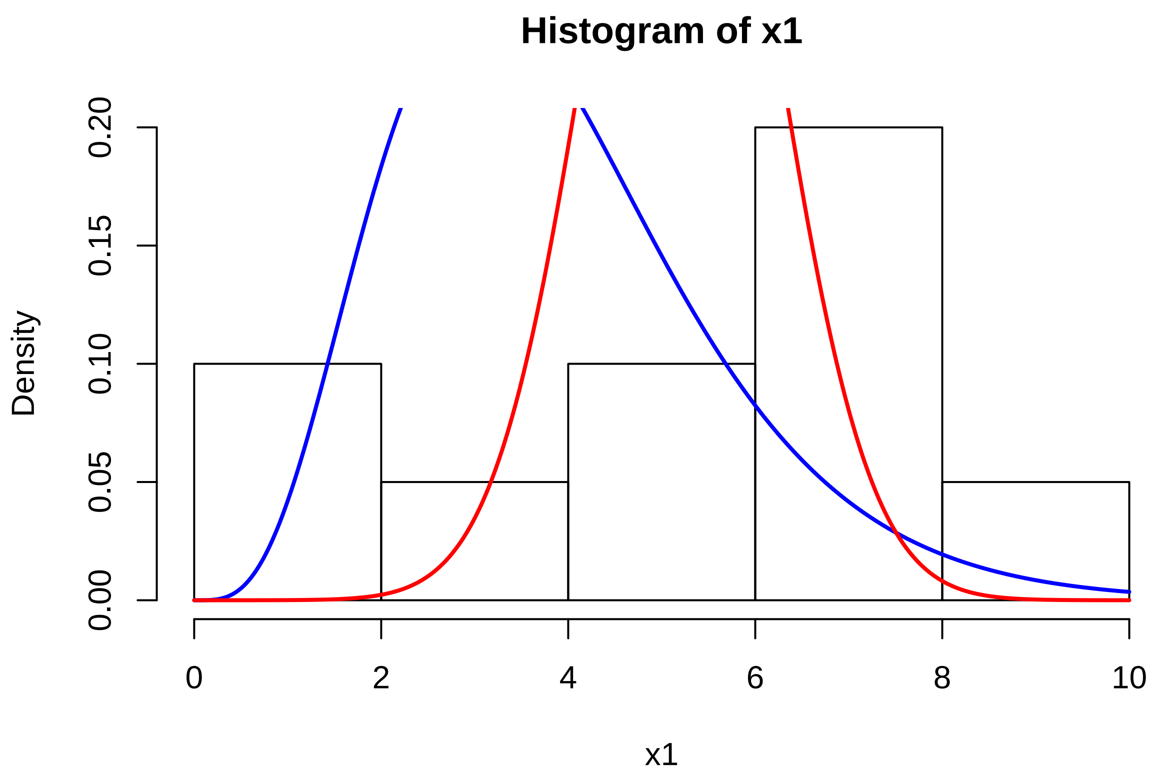
```r
# plot for ten draws
plot(steps, mu_tendraws)
```



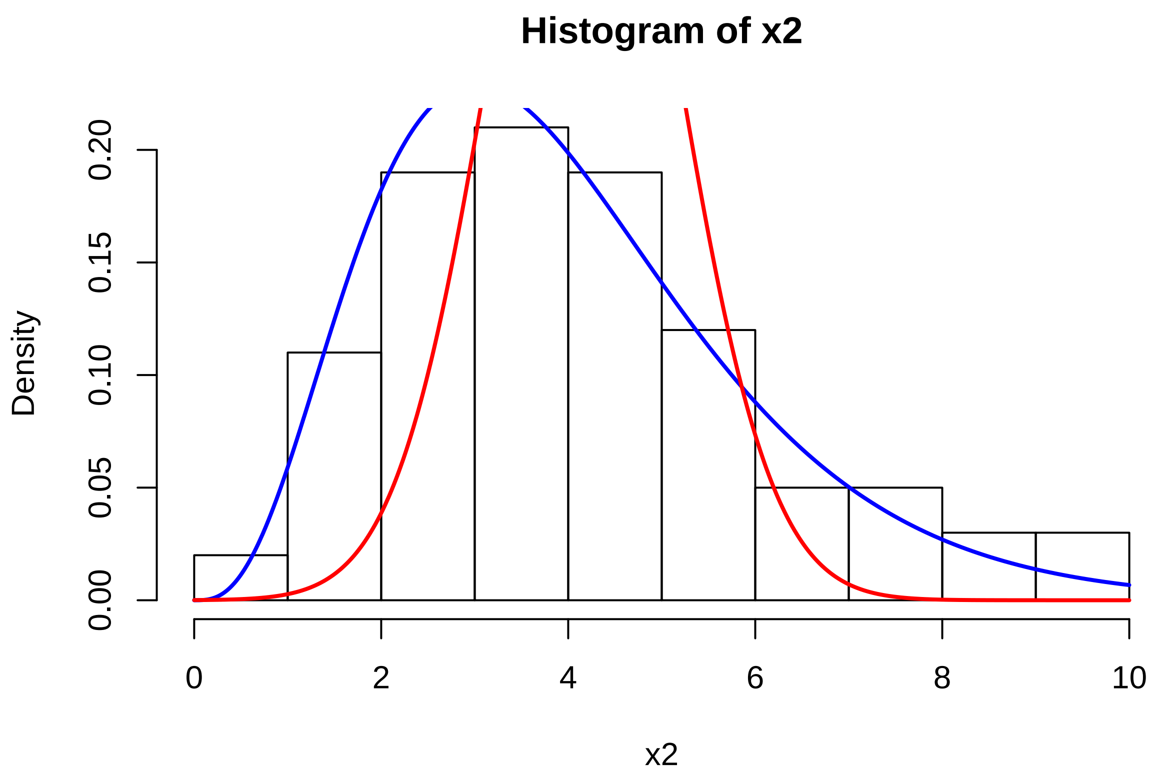

```r
# plot for hundred draws
plot(steps, mu_hundreddraws)
```




```
mu_max_ten <- findMax(mu_tendraws)
alpha_max_ten <- findMax(alpha_tendraws)
beta_max_ten <- findMax(beta_tendraws)
hist(x1, probability = TRUE)
xfit <- seq(0, 10, 0.01)
yfitgamma <- dgamma(xfit, shape = alpha_max_ten, scale = beta_max_ten)
lines(xfit, yfitgamma, col="blue", lwd=2)
yfitnorm <- dnorm(xfit, mean = mu_max_ten, sd = 1)
lines(xfit, yfitnorm, col="red", lwd = 2)
```




```
mu_max_hundred <- findMax(mu_hundreddraws)
alpha_max_hundred <- findMax(alpha_hundreddraws)
beta_max_hundred <- findMax(beta_hundreddraws)
hist(x2, probability = TRUE)
xfit <- seq(0, 10, 0.01)
yfitgamma <- dgamma(xfit, shape = alpha_max_hundred, scale = beta_max_hundred)
lines(xfit, yfitgamma, col="blue", lwd=2)
yfitnorm <- dnorm(xfit, mean = mu_max_hundred, sd = 1)
lines(xfit, yfitnorm, col="red", lwd = 2)
```



Gammafördelningen tycks passa datamaterialet bäst.

Uppgift 2 Punktskattning med MLE i en gammafördelning

```
gamma_beta_mle <- function(x, alpha) {
  return(length(x)*alpha*1/sum(x))
}
gamma_beta_mle(x1, 4)
```

```
## [1] 0.7683785
```

```
gamma_beta_mle(x2, 4)
```

```
## [1] 0.9619473
```

Vi testade att öka antalet dragningar och drar slutsatsen att estimatet går mot 1.0.

Uppgift 3 Punktskattning med MLE i en normalfördelning

a)

```r
norm_mu_mle <- function(x) {
  return(sum(x)/length(x))
}


norm_sigma2_mle <- function(x) {
  mean_x <- norm_mu_mle(x)
  sumhelp <- sum((x - mean_x)**2)
  return(sumhelp/length(x))
}
```

```r
test_x <- 1:10

norm_mu_mle(x = test_x)
```

```
## [1] 5.5
```

```r
norm_sigma2_mle(x = test_x)
```

```
## [1] 8.25
```

b)

```r
set.seed(42)
# Skattning med n = 10
y1 <- rnorm(n = 10, mean = 10, sd = 2)

norm_mu_mle(x = y1)
```

```
## [1] 11.09459
```

```r
norm_sigma2_mle(x = y1)
```

```
## [1] 2.512709
```

```r
# Skattning med n = 10000
y2 <- rnorm(n = 10000, mean = 10, sd = 2)

norm_mu_mle(x = y2)
```

```
## [1] 9.9762
```

```r
norm_sigma2_mle(x = y2)
```

```
## [1] 4.048198
```

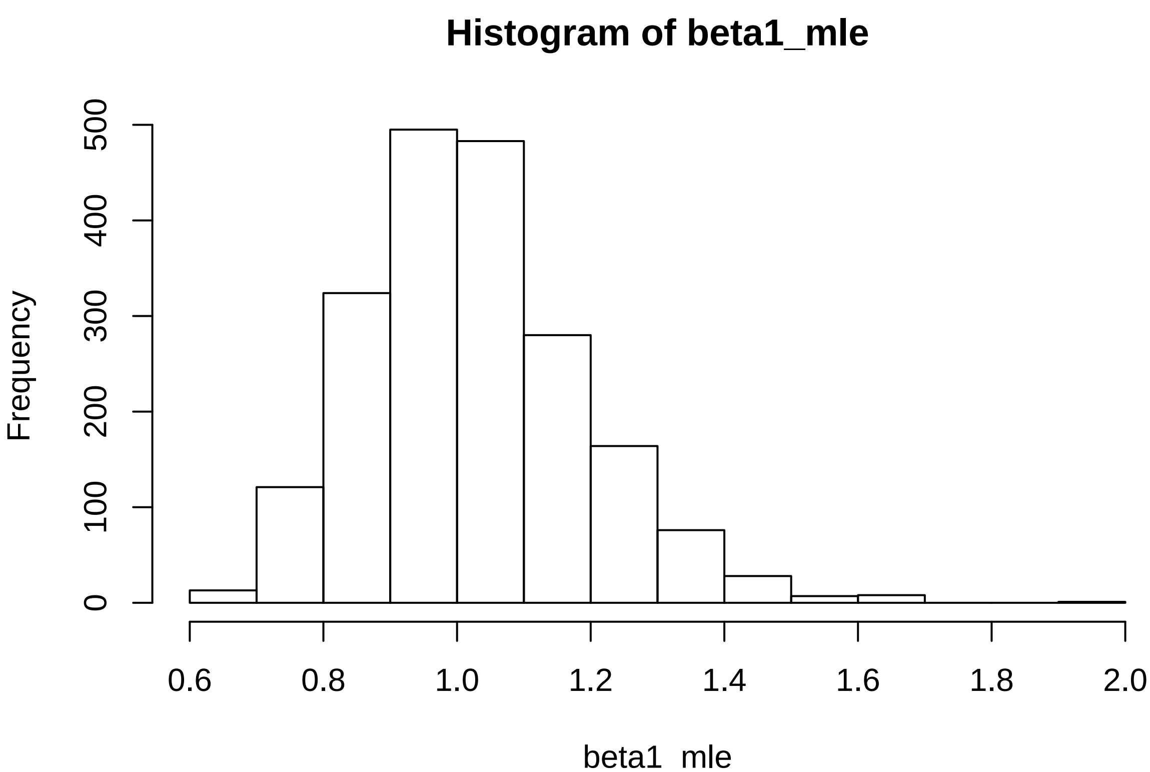
Desto större antal dragningar som görs, desto närmare kommer vi mu och sigma2, med respektive norm_mu_mle och norm_sigma2_mle. Detta följer av centralagärnsvärdessatsen som ger oss ett y som går mot normalfördelning och därmed tydligare väntevärde samt varians.

Uppgift 4 Samplingfördelningen för Bmle, MUmle och sigma2mle

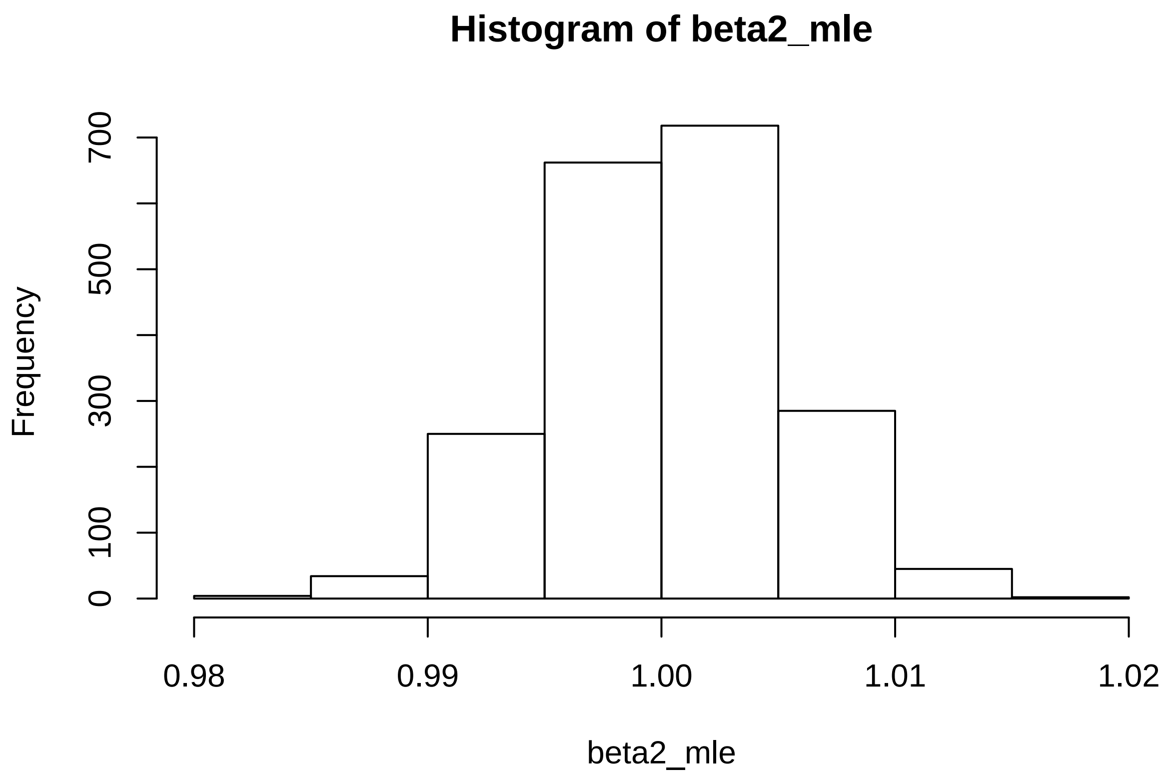
a)

```
beta1_mle <- c(1:2000)
beta2_mle <- c(1:2000)
mu1 <- c(1:2000)
mu2 <- c(1:2000)
sigma1 <- c(1:2000)
sigma2 <- c(1:2000)

i <- 1
while (i <= 2000) {
  x1 <- rgamma(n = 10, shape = 4, rate = 1)
  x2 <- rgamma(n = 10000, shape = 4, rate = 1)
  beta1_mle[i] <- gamma_beta_mle(x = x1, alpha = 4)
  beta2_mle[i] <- gamma_beta_mle(x = x2, alpha = 4)

  y1 <- rnorm(n = 10, mean = 10, sd = 2)
  y2 <- rnorm(n = 10000, mean = 10, sd = 2)
  mu1[i] <- norm_mu_mle(x = y1)
  mu2[i] <- norm_mu_mle(x = y2)
  sigma1[i] <- norm_sigma2_mle(x = y1)
  sigma2[i] <- norm_sigma2_mle(x = y2)
  i <- i + 1
}
```

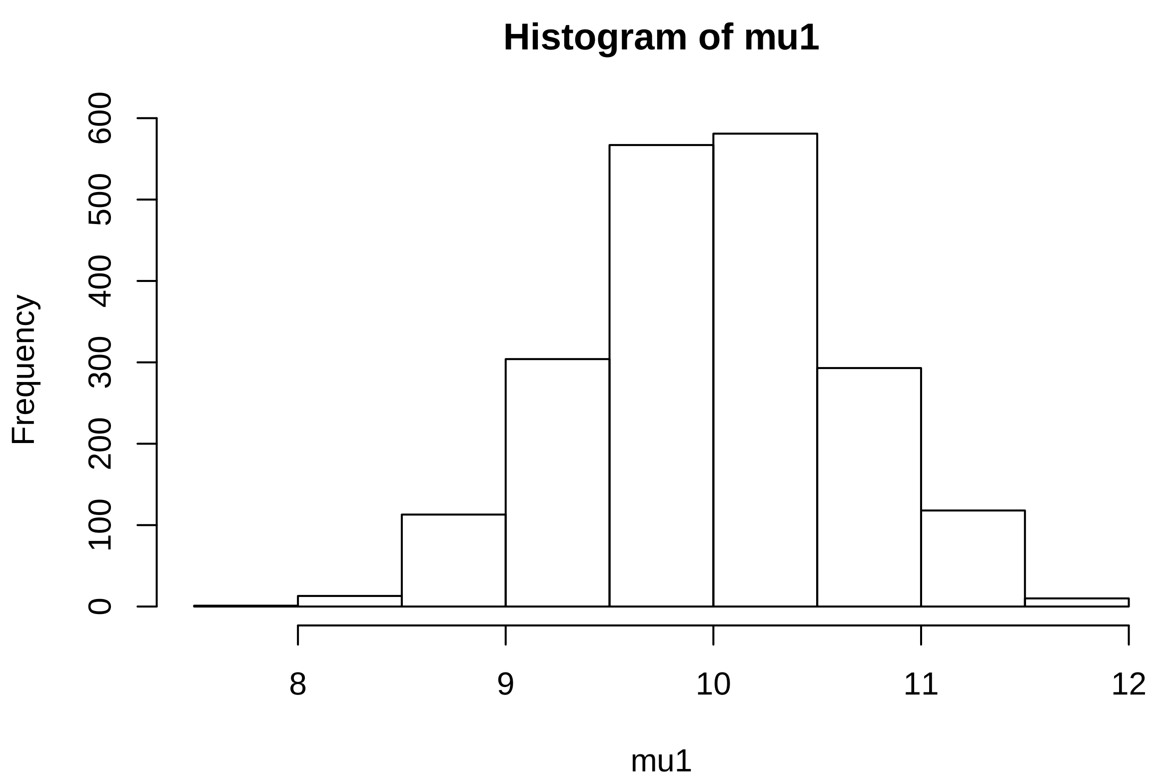
```
hist(beta1_mle)
```



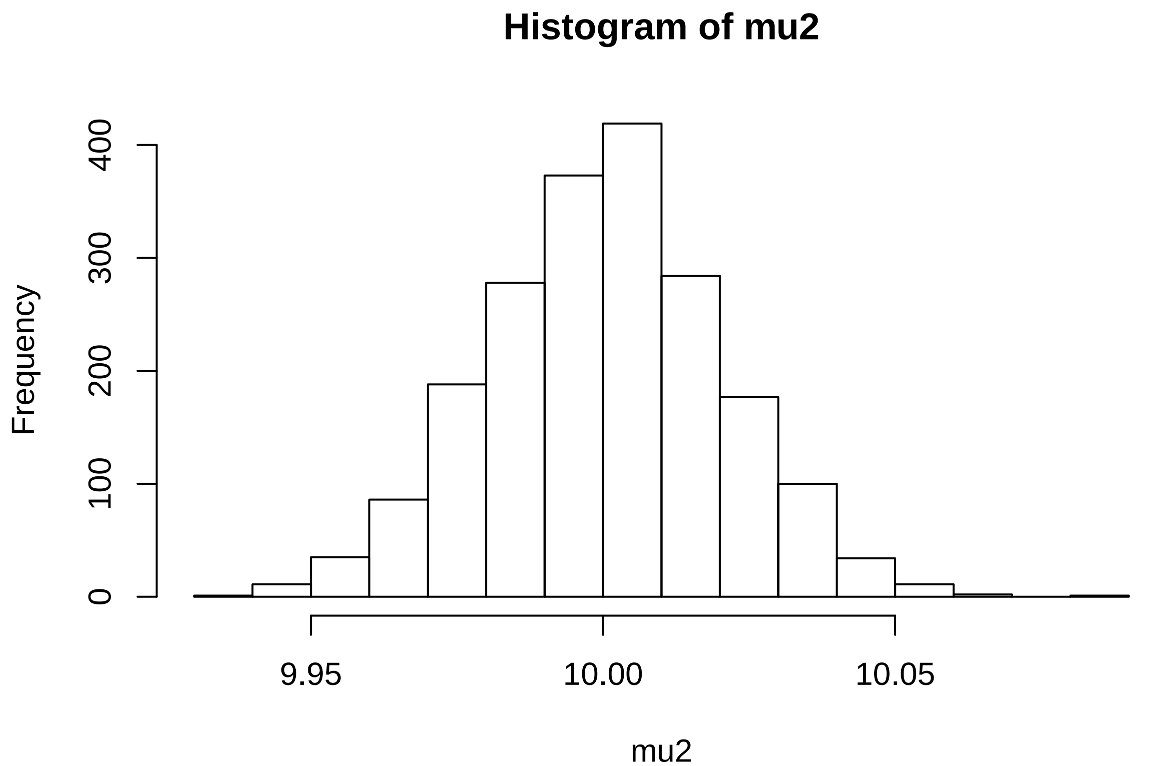
```r
hist(beta2_mle)
```



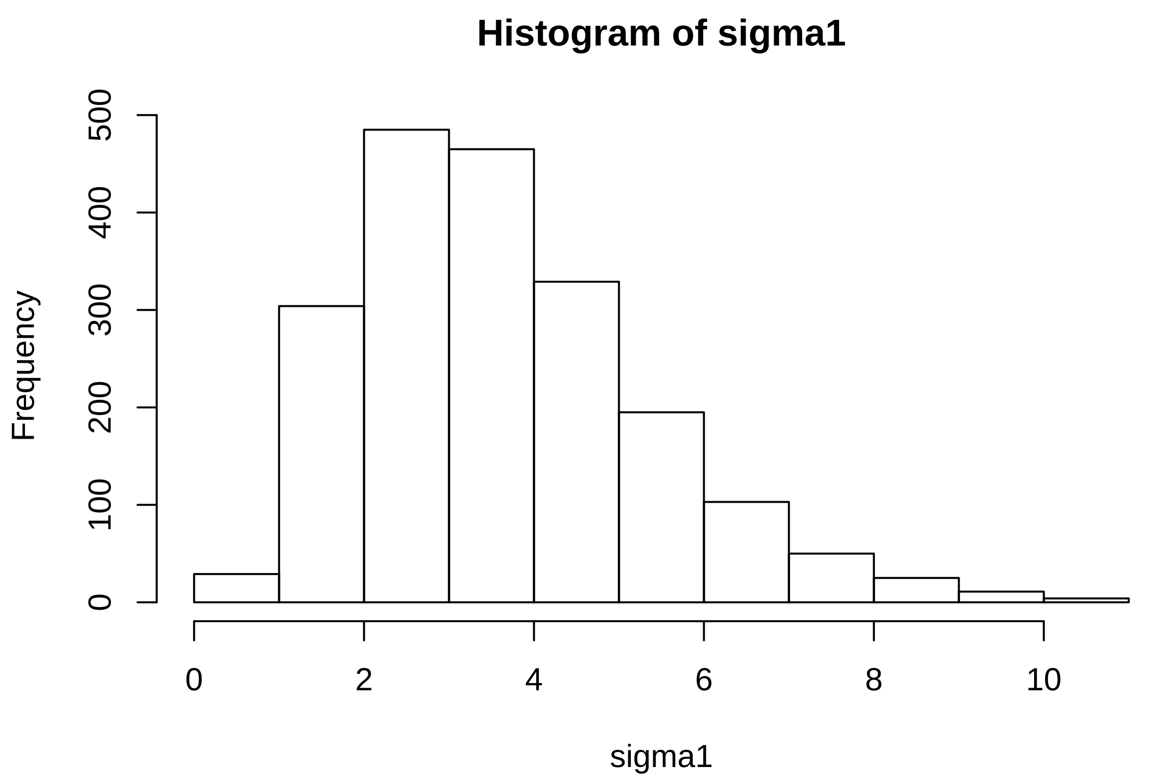

```r
hist(mu1)
```



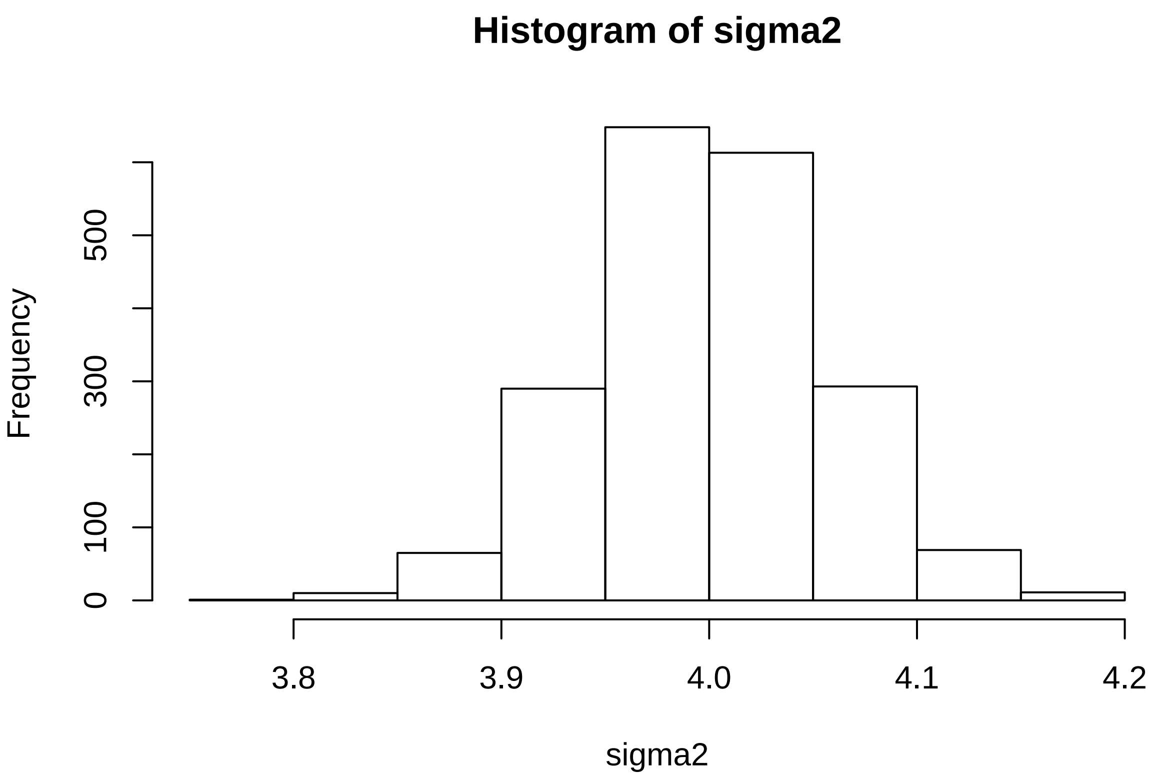
```r
hist(mu2)
```




```r
hist(sigma1)
```



```
hist(sigma2)
```



Precis som tidigare ser vi att ju fler dragningar så närmar sig histogrammen en normalfördelning vilket följer av den centrala gränsvärdessatsen.

Uppgift 5 Log-likelihoodfunktionen för betafördelning

a)

```
llbeta <- function(par, x){
  sum1 <- (par[1]-1)*sum(log(x))
  sum2 <- (par[2]-1)*sum(log(1-x))
  sum3 <- length(x)*log(gamma(par[1])*gamma(par[2])/gamma(par[1] + par[2]))
  return((sum1 + sum2 - sum3)*-1) # <- Varför multiplicera med -1??
}
llbeta(par = c(2, 2), x = c(0.01, 0.5, 0.99))
```

```
## [1] 5.241457
```

b)

```
x3 <- rbeta(100, 0.2, 2)
hist(x3)
```

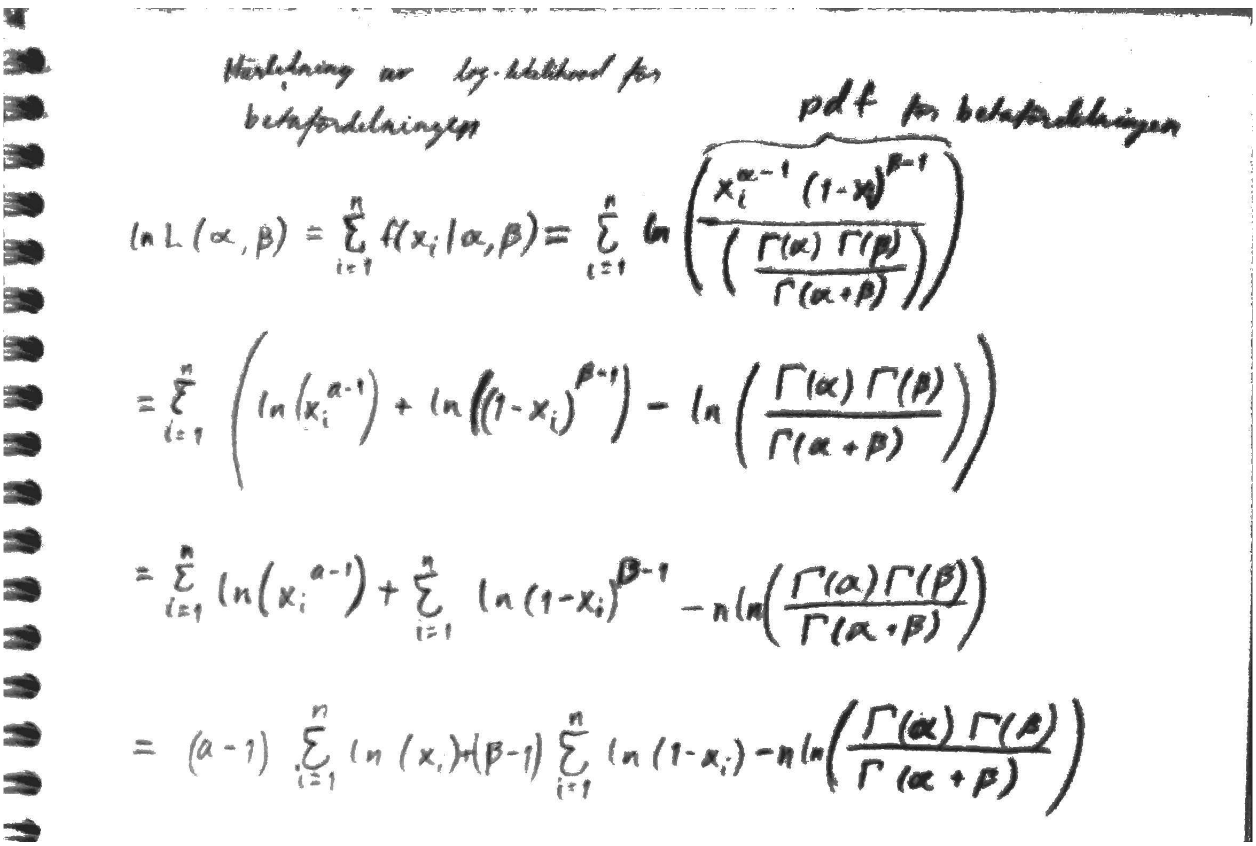
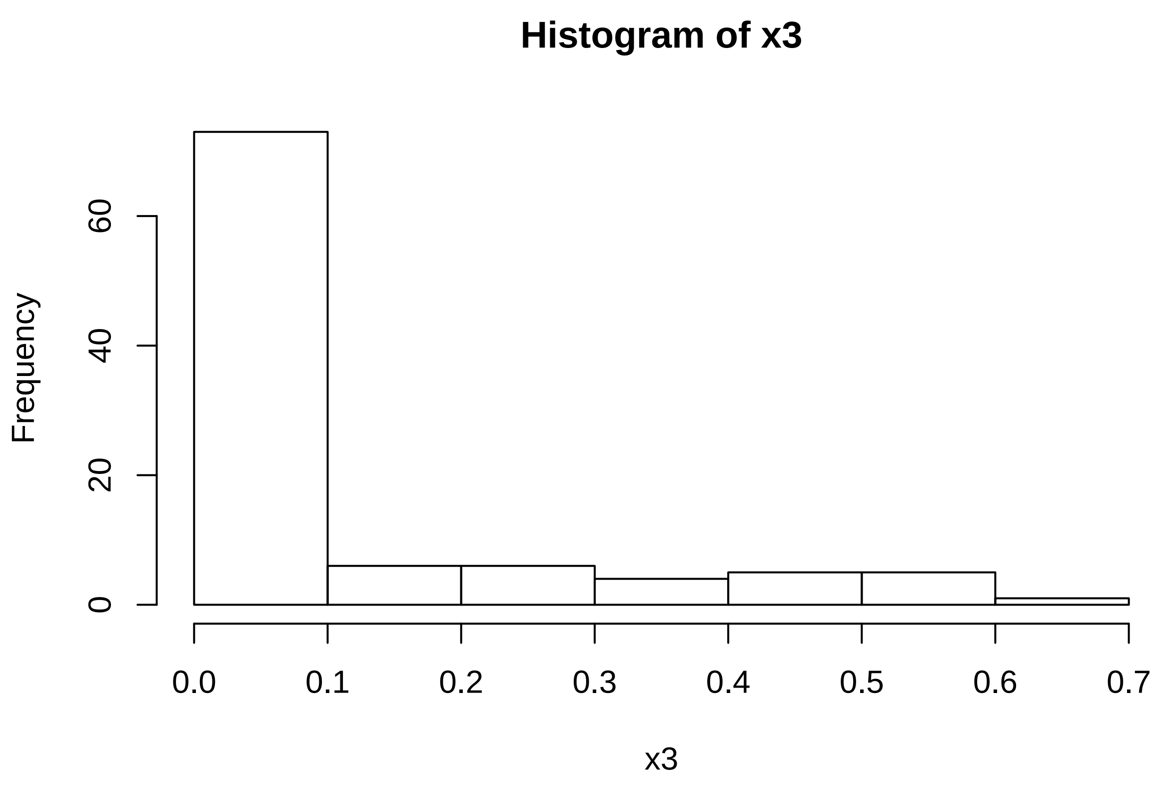


Figure 1: Härledning av log-likelihood för betafördelning



c)

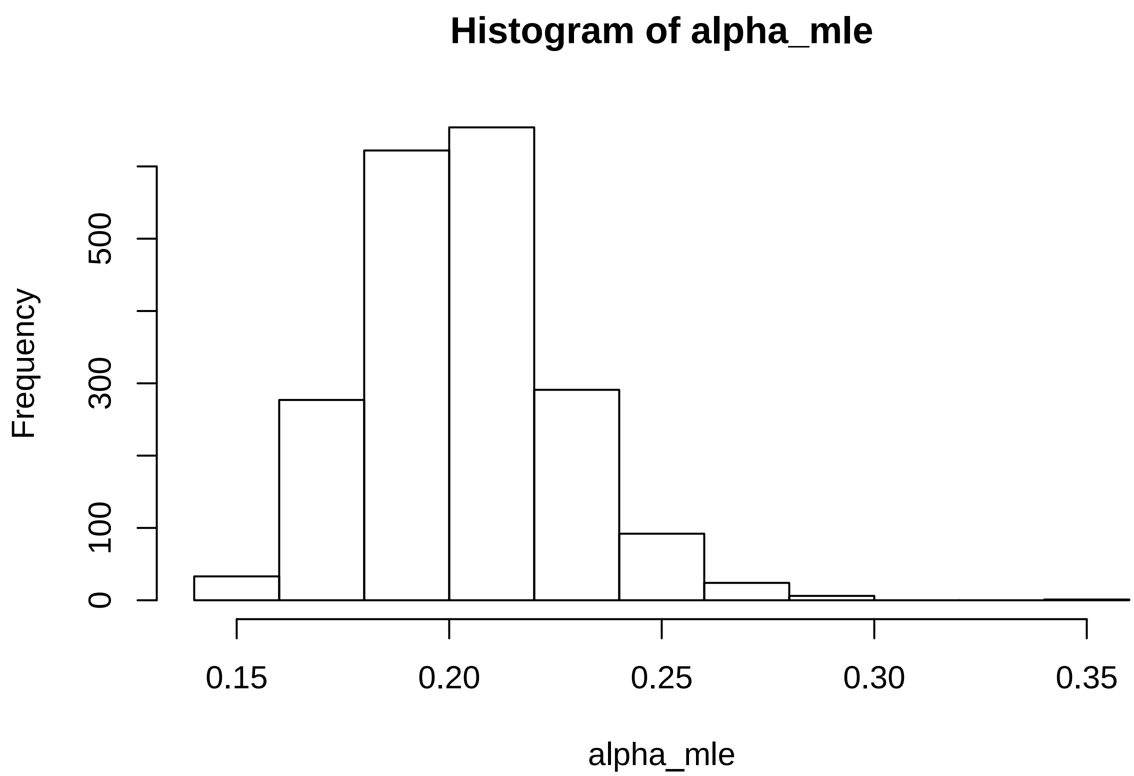
```
opt_res <- optim(par= c(0.2, 2), fn = llbeta, x=x3, upper=Inf,method ="L-BFGS-B", lower = .Machine$doubl
opt_res$par
```

```
## [1] 0.2008169 1.8445239
```

d)

```
alpha_mle <- c(1:2000)
beta_mle <- c(1:2000)
i <- 1
while(i <= 2000) {
  x3 <- rbeta(100, 0.2, 2)
  opt_res <- optim(par= c(0.2, 2), fn = llbeta, x=x3, upper=Inf,method ="L-BFGS-B", lower = .Machine$dou
  alpha_mle[i] <- opt_res$par[1]
  beta_mle[i] <- opt_res$par[2]
  i <- i + 1
}
```

```
hist(alpha_mle)
```




```r
hist(beta_mle)
```