Lab 2

Amanda Sandberg amasa869 & Philip Norberg phino637

Uppgift 3.1.1

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

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

value <- llgamma(x=x1, alpha=2, beta=2)
value
```

```
## [1] -75.18981
```

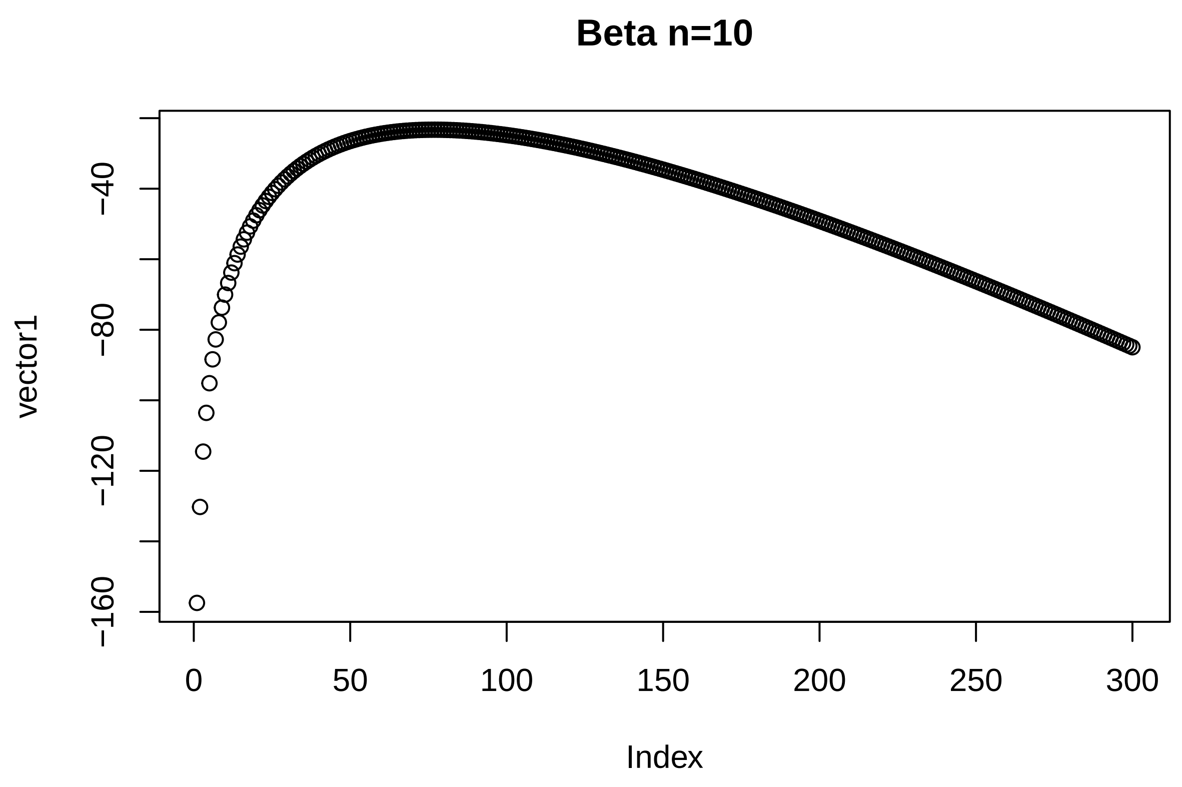
```r
# (2)

alpha <- 4
beta_step_vector = seq(0.01, 3, 0.01)

vector1 = c()
vector2 = c()

for (step in beta_step_vector) {
  vector1 <- append(vector1, llgamma(x1, alpha, step))
  vector2 <- append(vector2, llgamma(x2, alpha, step))

}

beta_maxX1 <- beta_step_vector[which.max(vector1)]
beta_maxX2 <- beta_step_vector[which.max(vector2)]
beta_maxX1
```

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

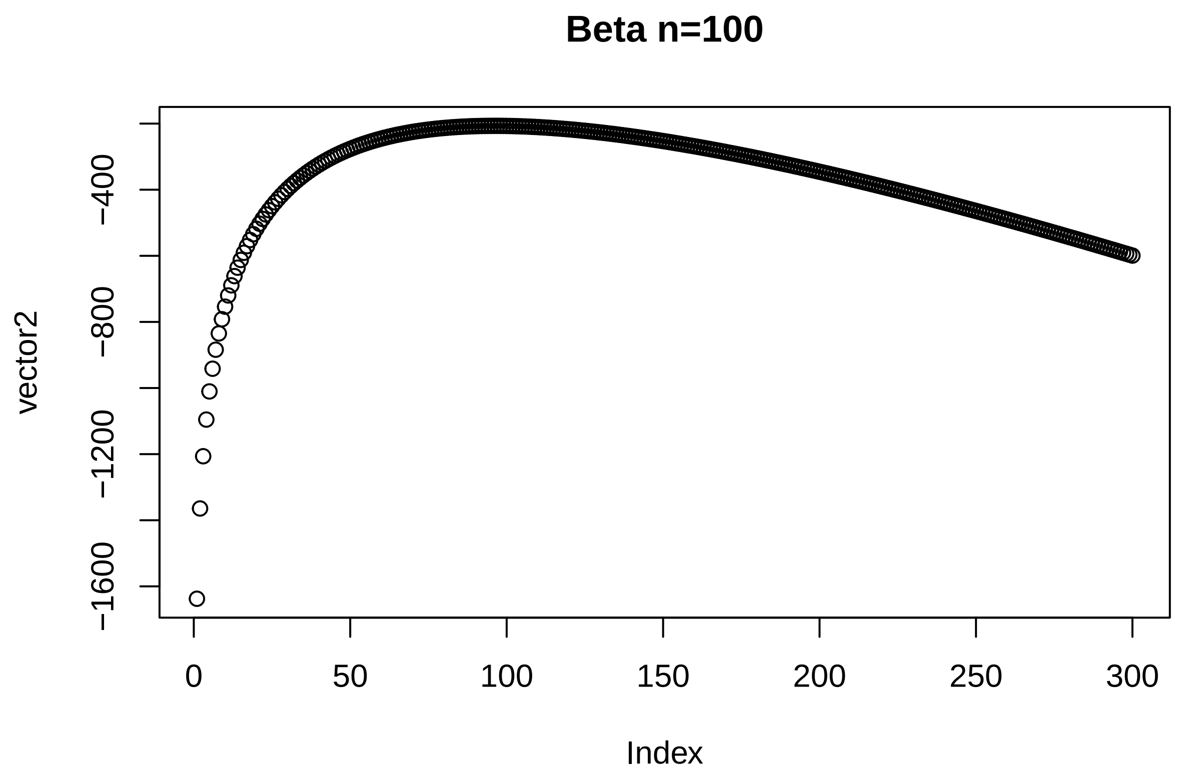
```r
beta_maxX2
```

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

```r
plot(vector1, main="Beta n=10")
```



```r
plot(vector2, main="Beta n=100")
```



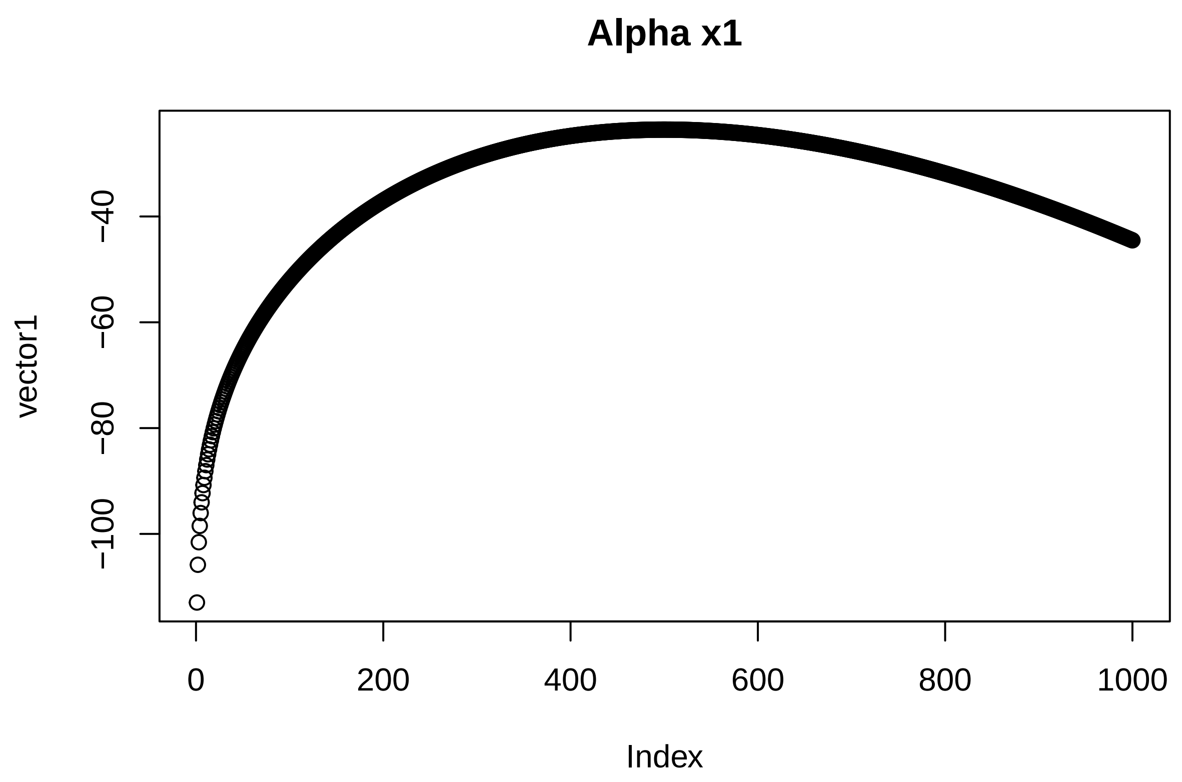
```
# (3)

beta <- 1
alpha_step_vector = seq(0.01, 10, 0.01)

vector1 = c()
vector2 = c()

for (step in alpha_step_vector) {
  vector1 <- append(vector1, llgamma(x1, step, beta))
  vector2 <- append(vector2, llgamma(x2, step, beta))
}

alpha_maxX1 <- alpha_step_vector[which.max(vector1)]
alpha_maxX2 <- alpha_step_vector[which.max(vector2)]
alpha_maxX1
```

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

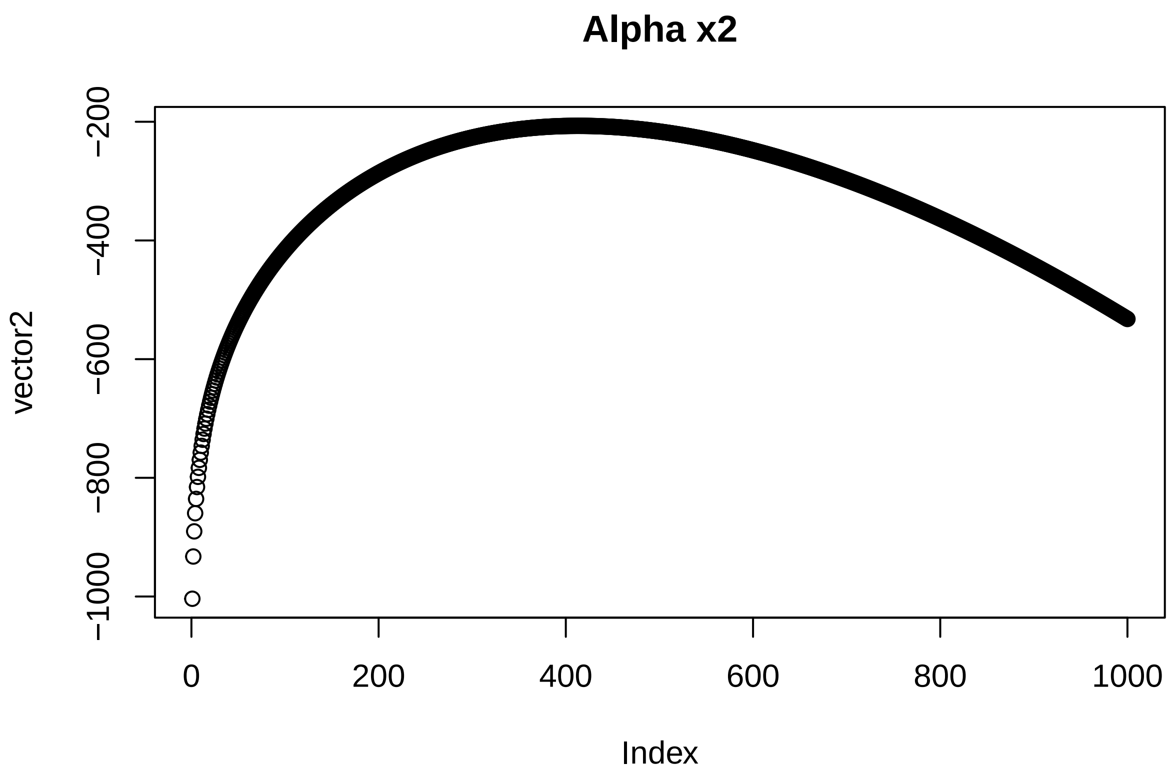
```
alpha_maxX2
```

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

```
plot(vector1, main="Alpha x1")
```



```
plot(vector2, main="Alpha x2")
```



(4) härledning av log-likelihood. Utgick ifrån https://www.statlect.com/fundamentals-of-statistics/normal-distribution-maximum-likelihood/

Likelihood-funktionen för en normalfördelning är:
$(2\pi\sigma\check{s})^{-n/2}exp\left(-\frac{1}{2\sigma\check{s}}\sum_{j=1}^{n}(x_j-\mu)\check{s}\right)$

Ta sedan logaritmen av likelihood-funktionen:
$= ln\left((2\pi\sigma\check{s})^{-n/2}exp\left(\frac{-1}{2\sigma\check{s}}\sum_{j=1}^{n}(x_j-\mu)\check{s}\right)\right)$

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

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

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

```
# (4)

llnorm <- function(x, mu, sigma2) {
  -(length(x)/2)*log(2*pi)-(length(x)/2)*log(sigma2)-(1/(2*sigma2))*sum((x-mu)*(x-mu))
}

llnorm(x1,2,1)
```

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

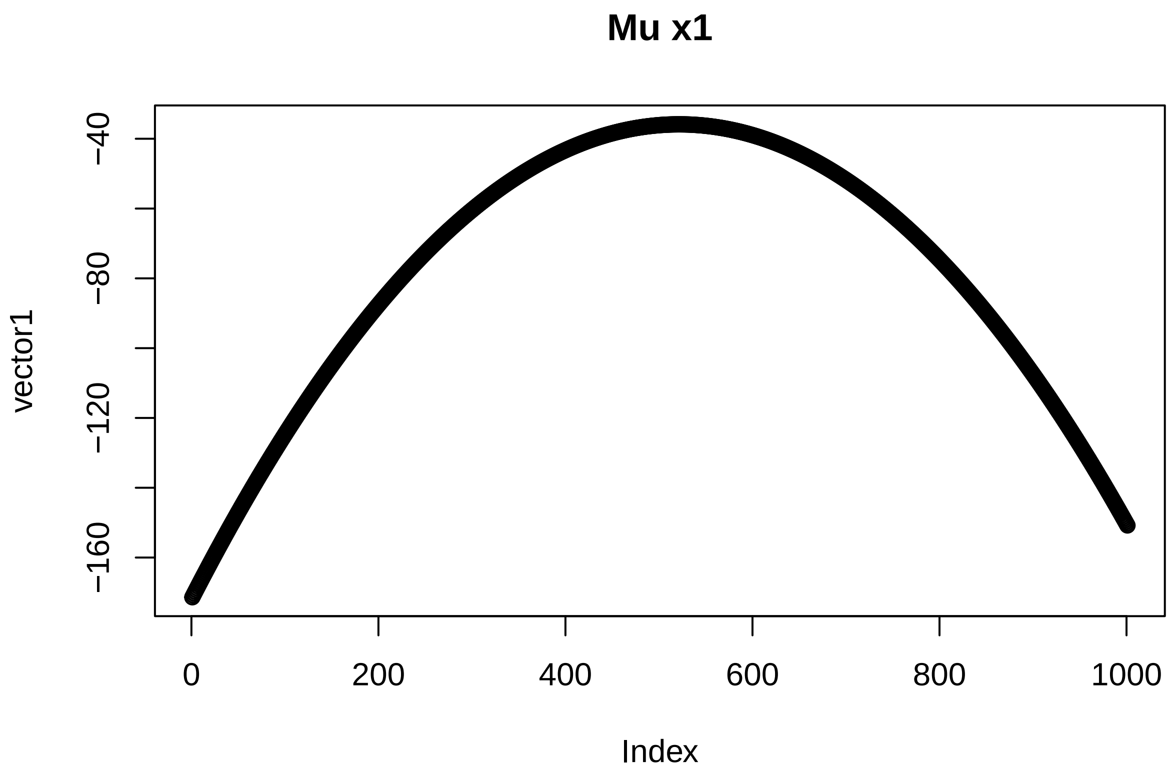
```r
# (5)

sigma2 <- 1

mu_step_vector = seq(0, 10, 0.01)

vector1 = c()
vector2 = c()

for (step in mu_step_vector) {
  vector1 <- append(vector1, llnorm(x1, step, sigma2))
  vector2 <- append(vector2, llnorm(x2, step, sigma2))
}

mu_maxX1 <- mu_step_vector[which.max(vector1)]
mu_maxX2 <- mu_step_vector[which.max(vector2)]
mu_maxX1
```

```
## [1] 5.21
```

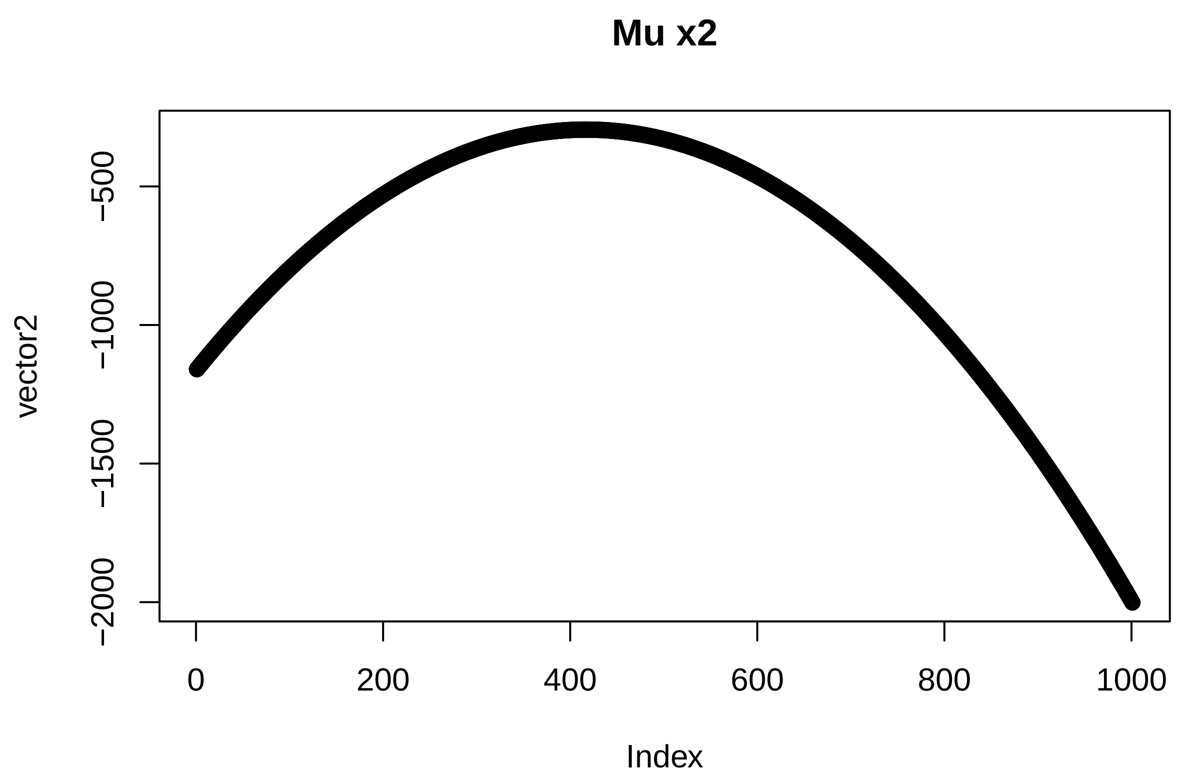
```r
mu_maxX2
```

```
## [1] 4.16
```

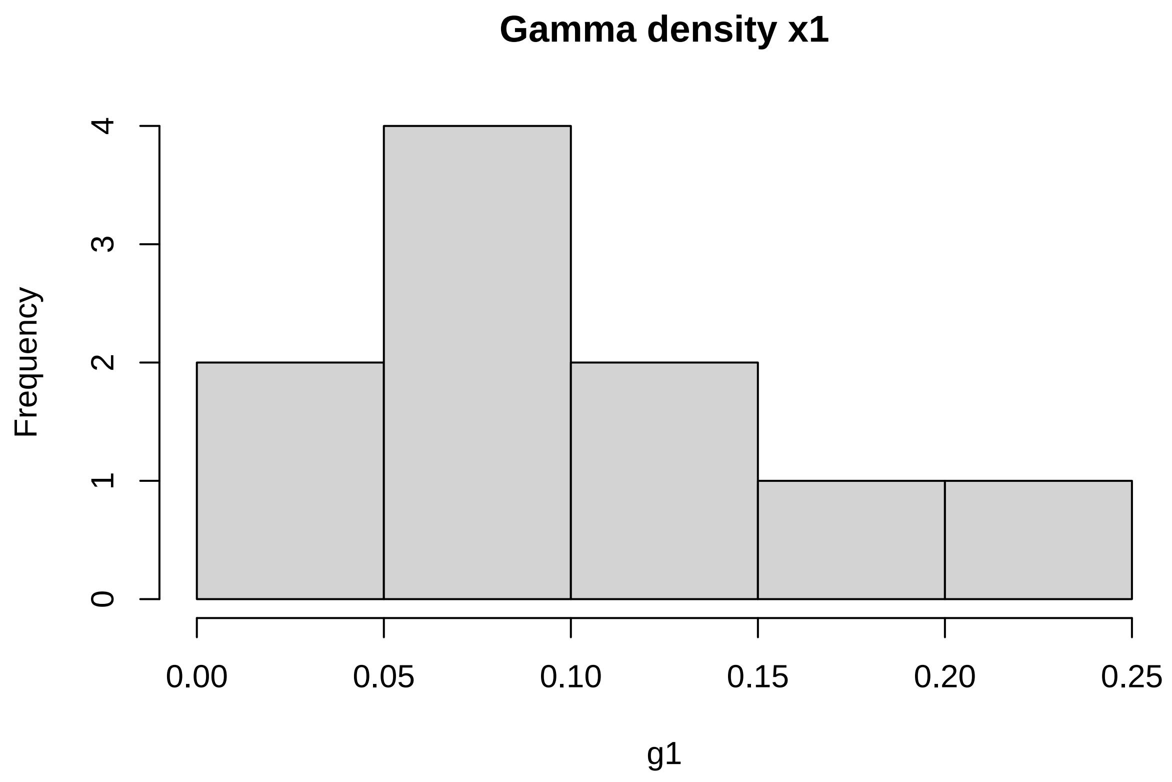
```r
plot(vector1, main="Mu x1")
```



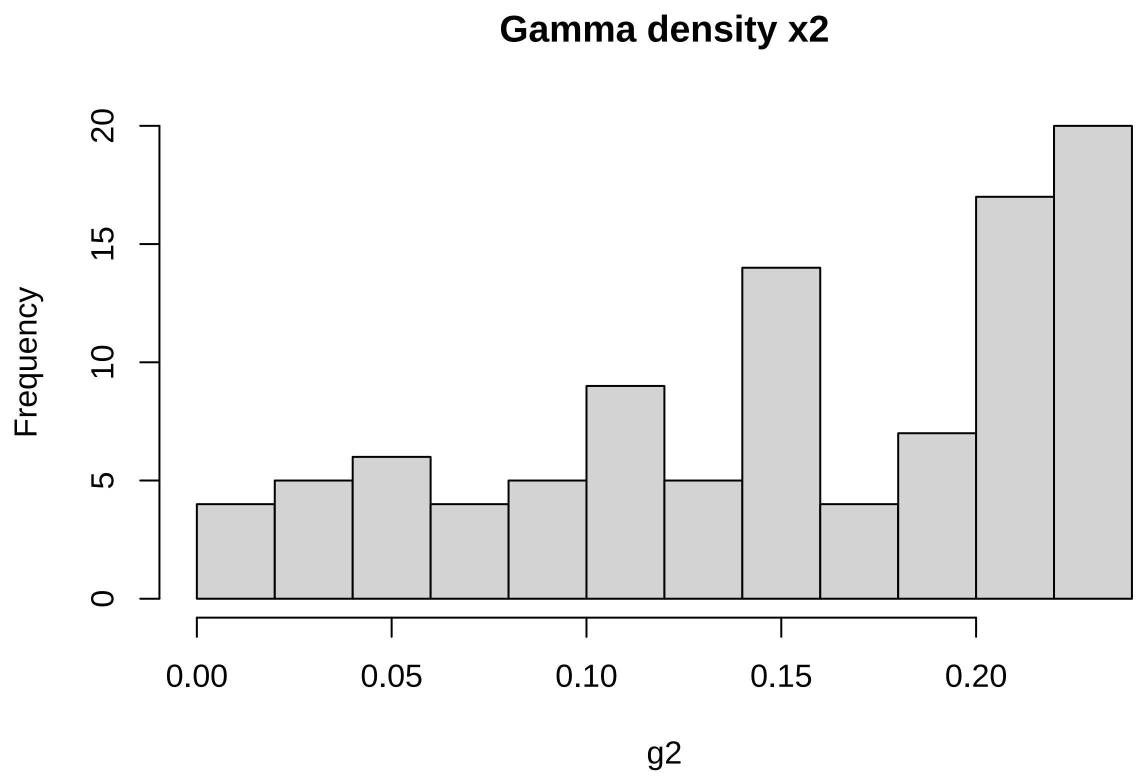
```
plot(vector2, main="Mu x2")
```



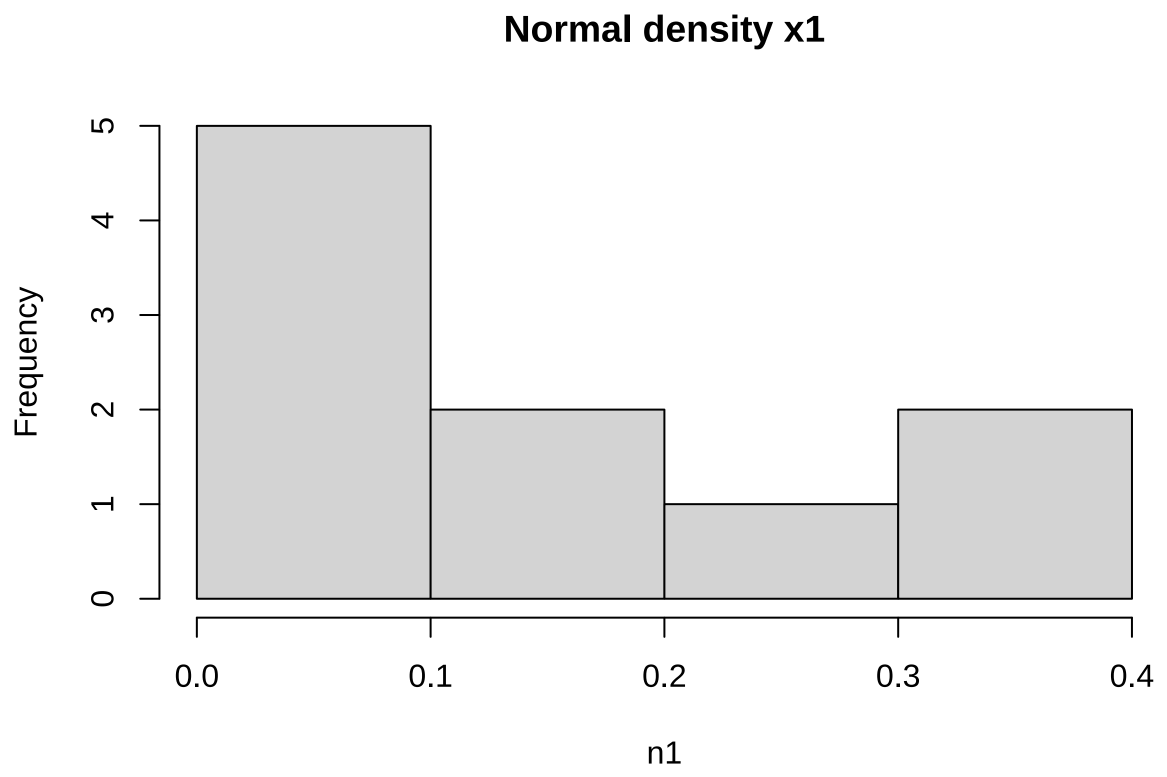
```
g1 <- dgamma(x1, shape = alpha_maxX1, scale=beta_maxX1)
g2 <- dgamma(x2, shape=alpha_maxX2, scale = beta_maxX2)
n1 <- dnorm(x1, mean=mu_maxX1, sd=1)
n2 <- dnorm(x2, mean=mu_maxX2, sd=1)
hist(g1, main="Gamma density x1")
```



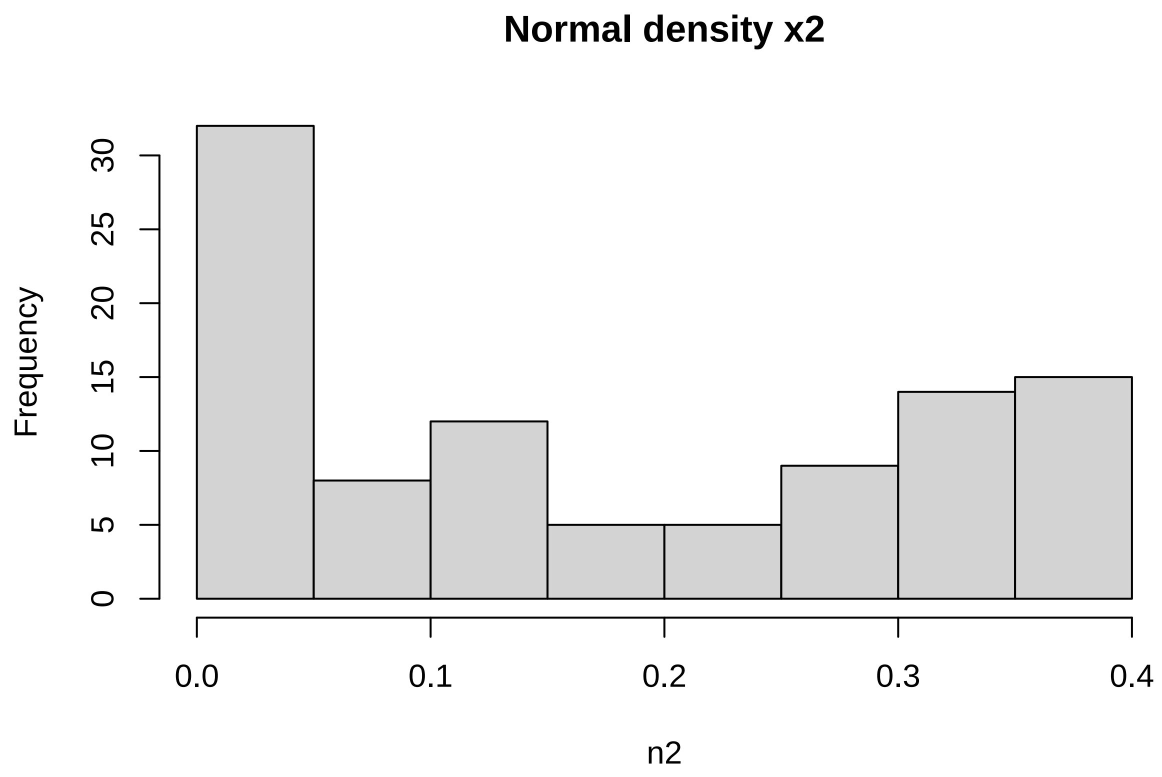
```r
hist(g2, main="Gamma density x2")
```



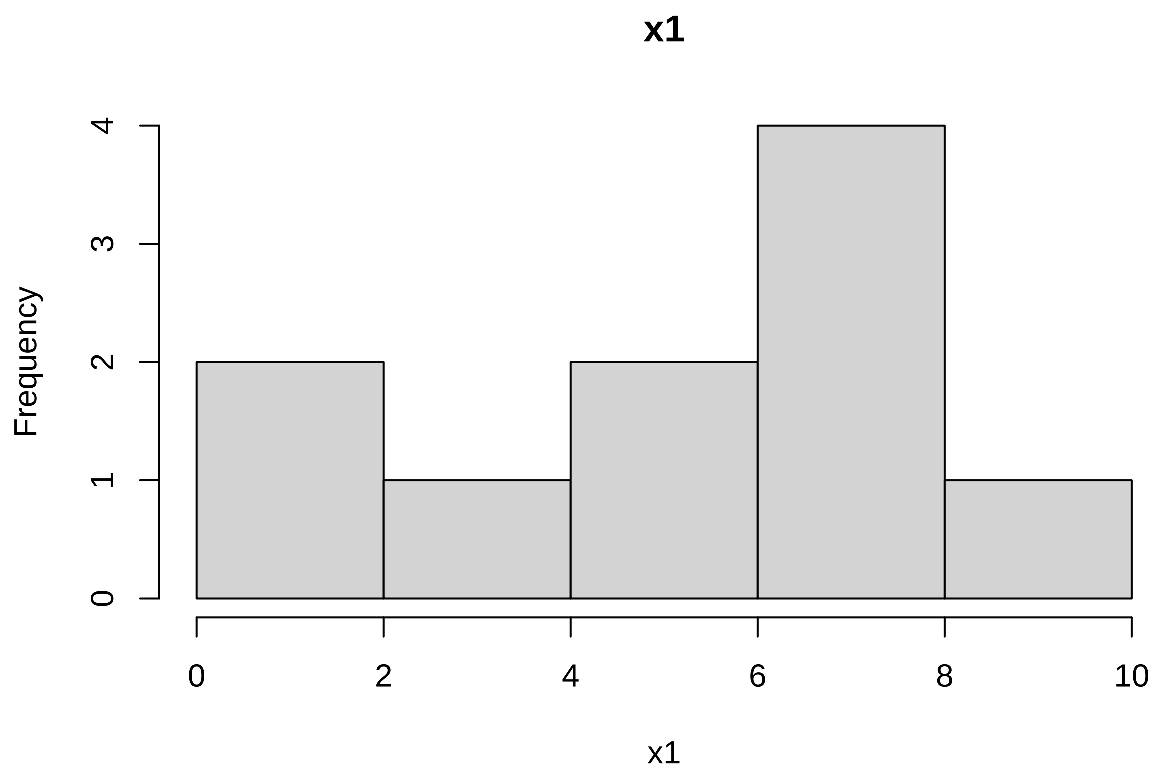
```
hist(n1, main="Normal density x1")
```



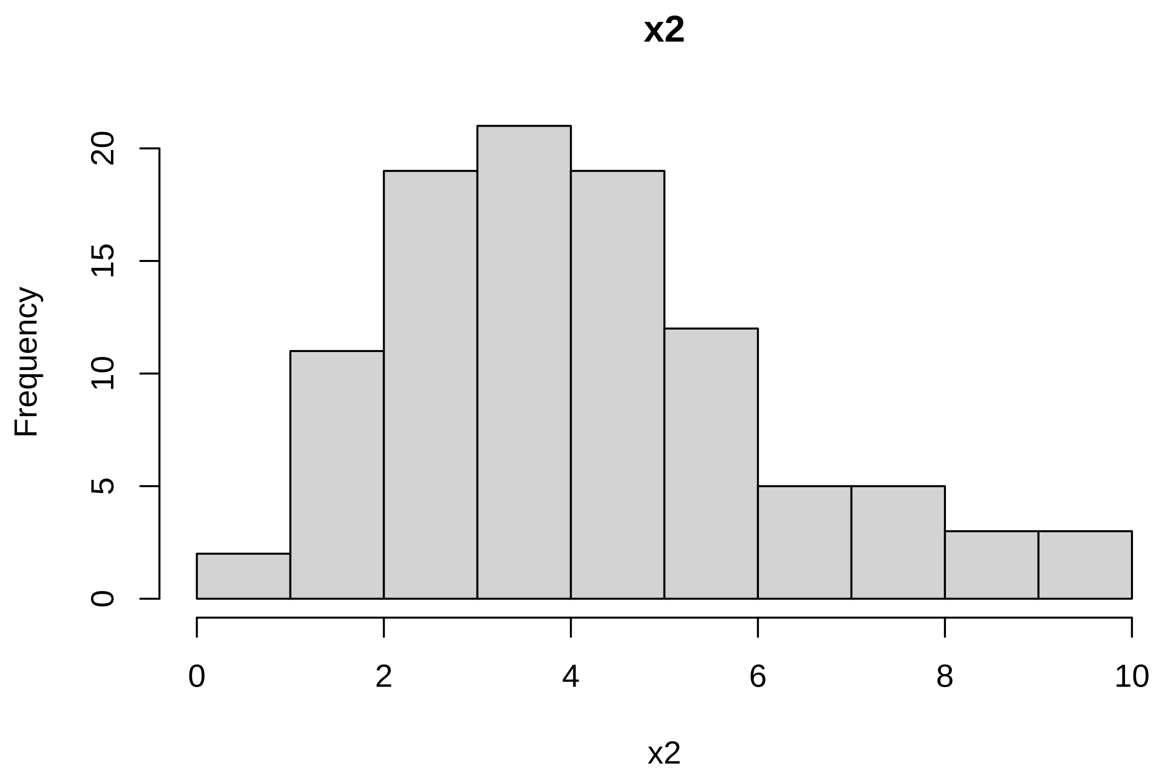
```r
hist(n2, main="Normal density x2")
```



```
hist(x1, main="x1")
```



```r
hist(x2, main="x2")
```



Vi tycker att gammafördelningen passar datamaterialet bäst.

Uppgift 3.2.1

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


gamma_beta_mle <- function(x, alpha){
  length(x)*alpha/sum(x)
}

bmle1 <- gamma_beta_mle(x1, 4)
bmle2 <- gamma_beta_mle(x2, 4)
bmle1
```

```
## [1] 1.261754
```

```
bmle2
```

```
## [1] 0.9585935
```

Det faktista beta-värdet är 1. Med 10 dragningar så estimeras beta-värdet vara 0.88 och för 100 dragningar estimeras beta-värdet vara 1.01. Det visar på att ju fler datapunkter som finns, desto bättre punktskattning sker för beta-värdet.

Uppgift 3.2.2

```r
# (1)
test_x <- 1:10
norm_mu_mle <- function(x){
  (1/length(x))*sum(x)
}

norm_sigma2_mle <- function(x){
  xhat = norm_mu_mle(x)
  (1/length(x))*sum((x-xhat)^2)
}

# (2)
set.seed(42)

x1 <- rnorm(10, 10, sqrt(4))
x2 <- rnorm(10000, 10, sqrt(4))

mu10 <- norm_mu_mle(x1)
mu10k <- norm_mu_mle(x2)
sigma10 <- norm_sigma2_mle(x1)
sigma10k <- norm_sigma2_mle(x2)
mu10
```

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

```r
mu10k
```

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

```r
sigma10
```

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

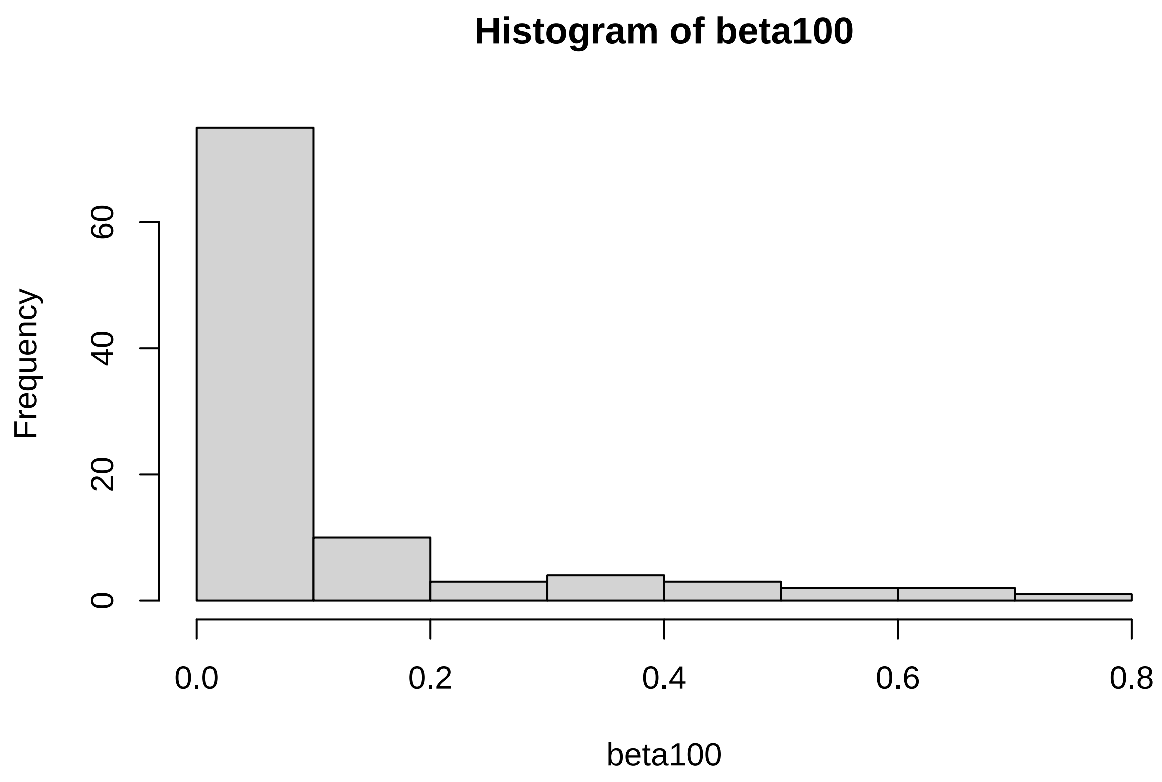
```r
sigma10k
```

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

Skillnaden mellan 10 och 10000 dragningar är att det estimerade värdet går mot det faktiska värdet ju fler dragningar som görs.

Uppgift 3.3.1

```r
# (1)

# https://www.youtube.com/watch?v=8nogLkirA3I
# L(theta | x ) -> L(theta | x1, x2, x3, ...) where x1 etc is independent ->
# -> L(theta | x1, x2, x3, ..) = L(theta | x1) * L(theta | x2)* ...
# Then go log(L(theta | x)) which results in a sum of the log of the likelihoods


llbeta <- function(par, x){
  return(-(sum(dbeta(x, par[1], par[2], log=TRUE))))
}

# (2)

beta100 <- rbeta(100, 0.2, 2)
hist(beta100)
```



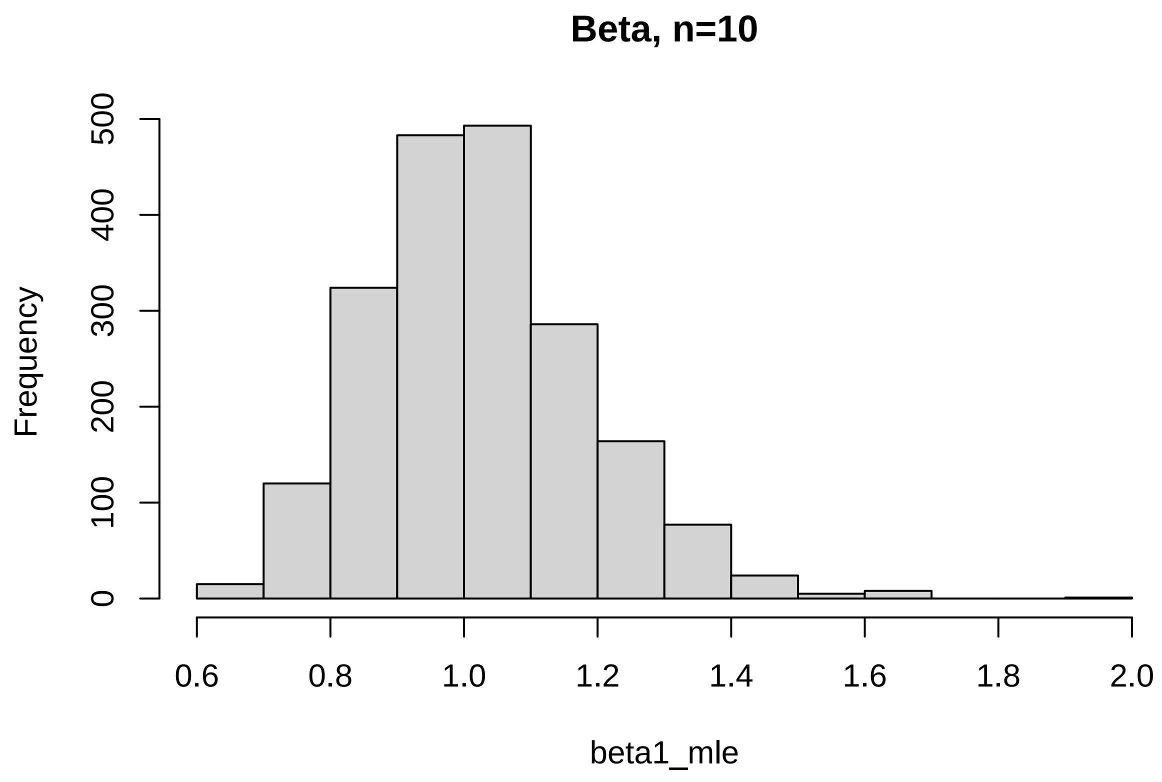
```r
# (3)

opt_res <- optim(par = c(0,1), fn = llbeta, x=beta100, method="L-BFGS-B", lower=0.00000001)
opt_res$par
```

```
## [1] 0.2211382 2.1439118
```

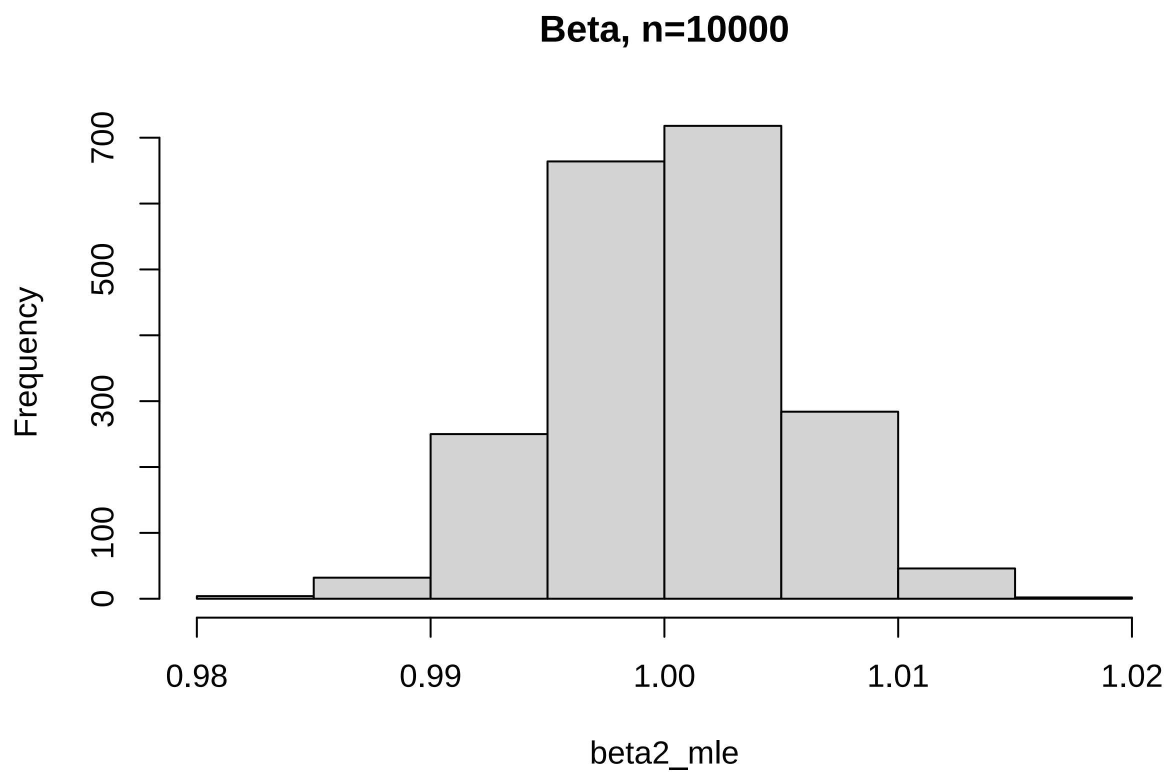
Uppgift 3.4.1

```r
# (1)
beta1_mle <- c()
```

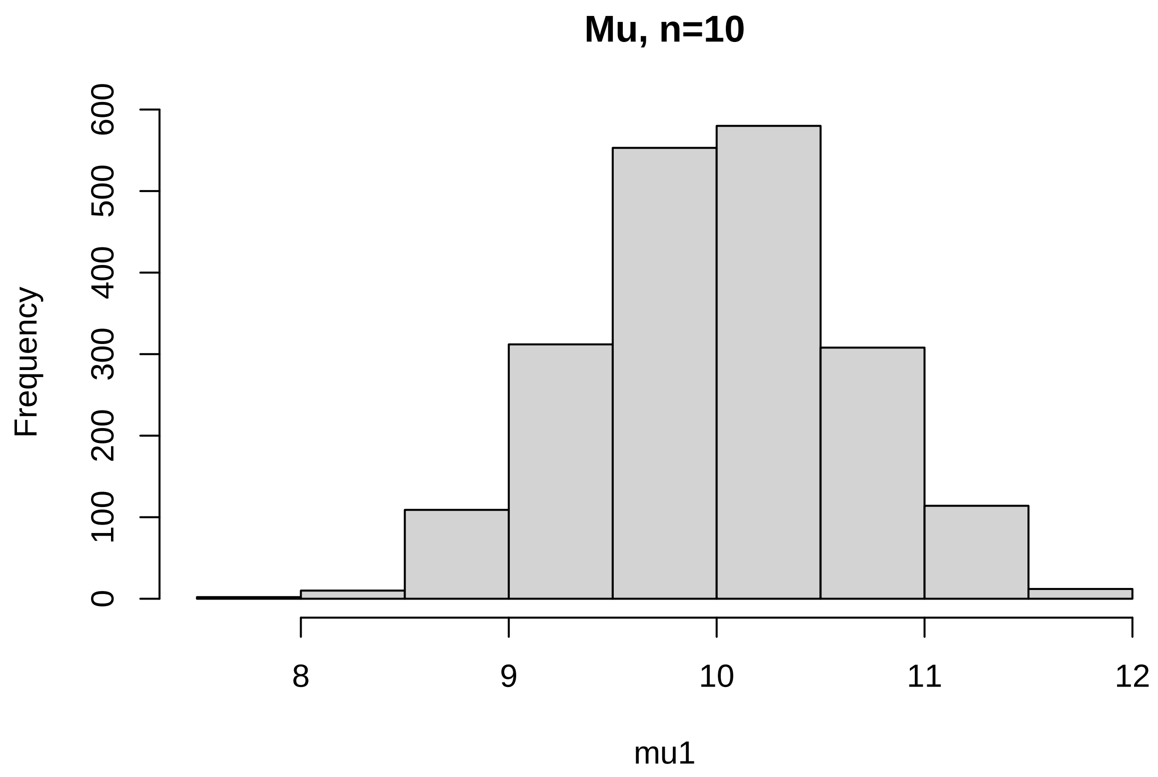
```r
beta2_mle <- c()
mu1 <- c()
mu2 <- c()
sigma1 <- c()
sigma2 <- c()

for (i in 1:2000) {

  x1 <- rgamma(10, 4, 1)
  x2 <- rgamma(10000, 4, 1)
  beta1_mle[i] <- gamma_beta_mle(x1, 4)
  beta2_mle[i] <- gamma_beta_mle(x2, 4)

  y1 <- rnorm(10, 10, 2)
  y2 <- rnorm(10000, 10, 2)
  mu1[i] <- norm_mu_mle(y1)
  mu2[i] <- norm_mu_mle(y2)
  sigma1[i] <- norm_sigma2_mle(y1)
  sigma2[i] <- norm_sigma2_mle(y2)

}
hist(beta1_mle, main="Beta, n=10")
```



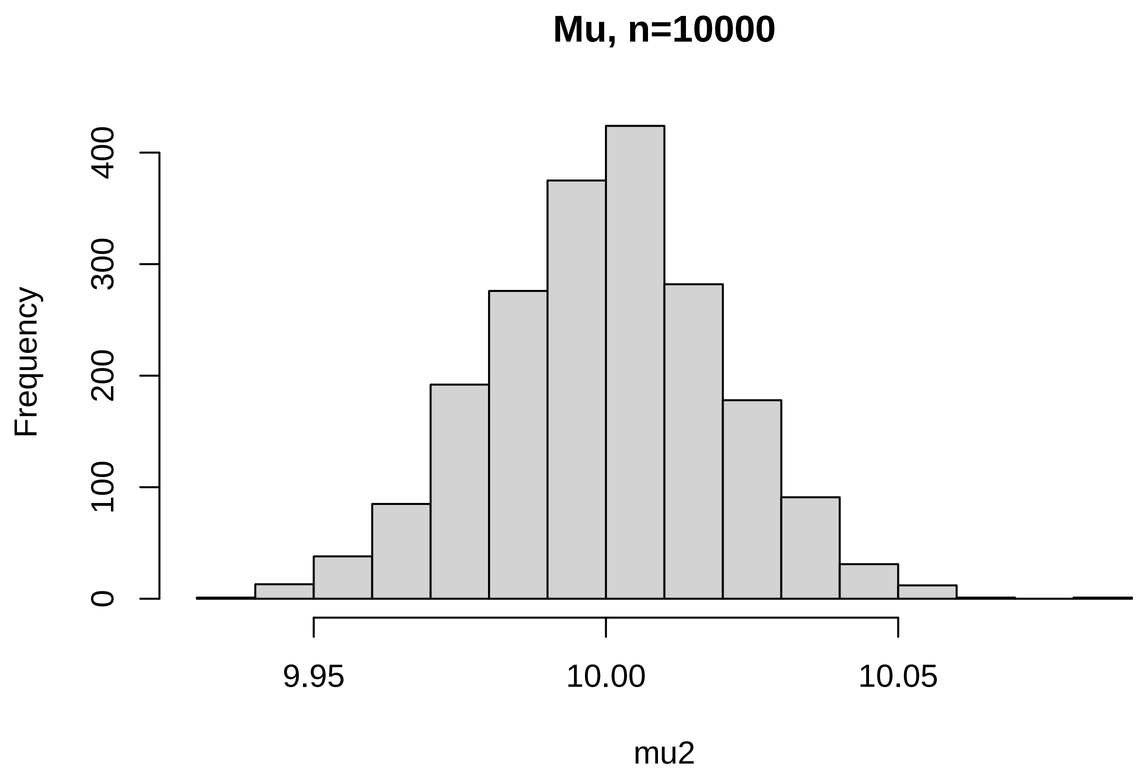
```r
hist(beta2_mle, main="Beta, n=10000")
```



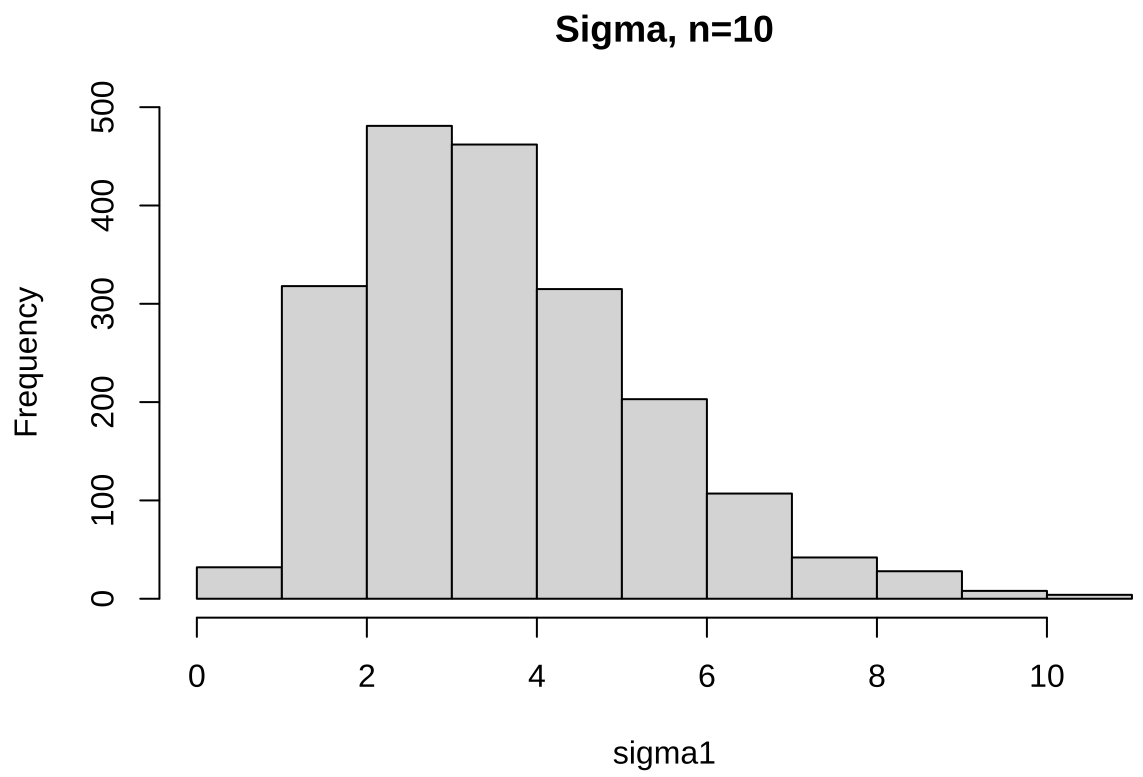
```r
hist(mu1, main="Mu, n=10")
```



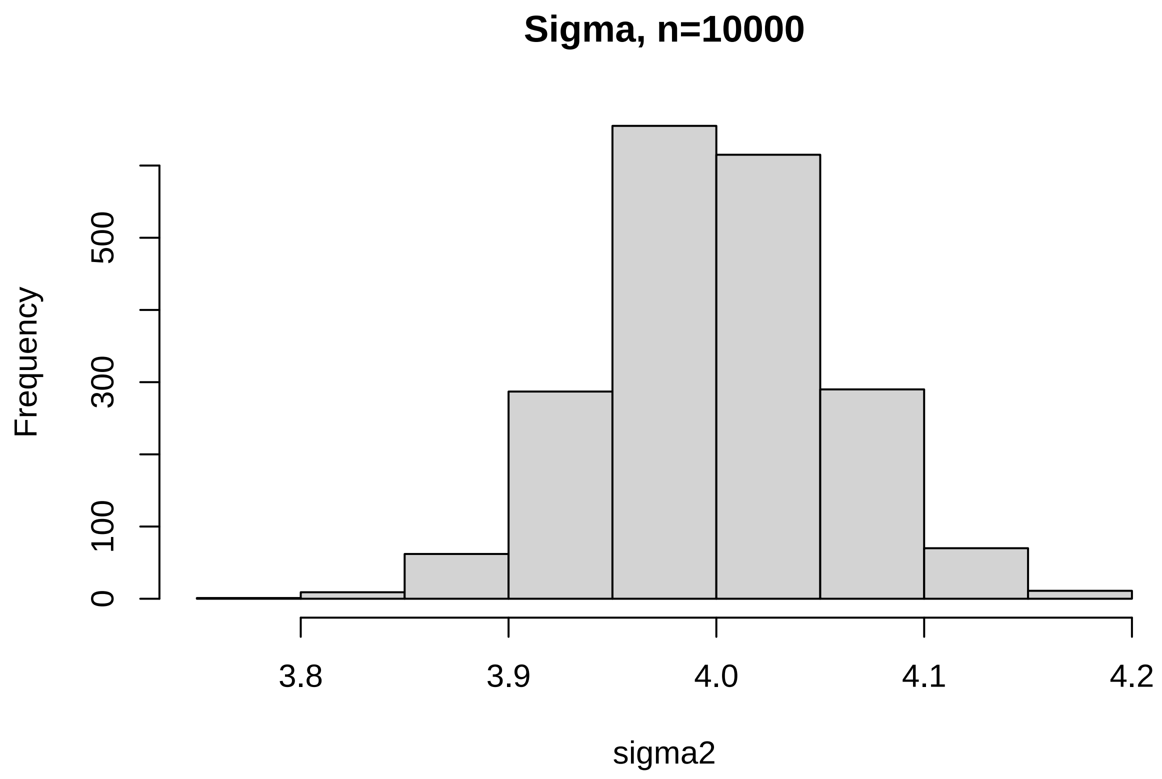
```r
hist(mu2, main="Mu, n=10000")
```



```r
hist(sigma1, main="Sigma, n=10")
```

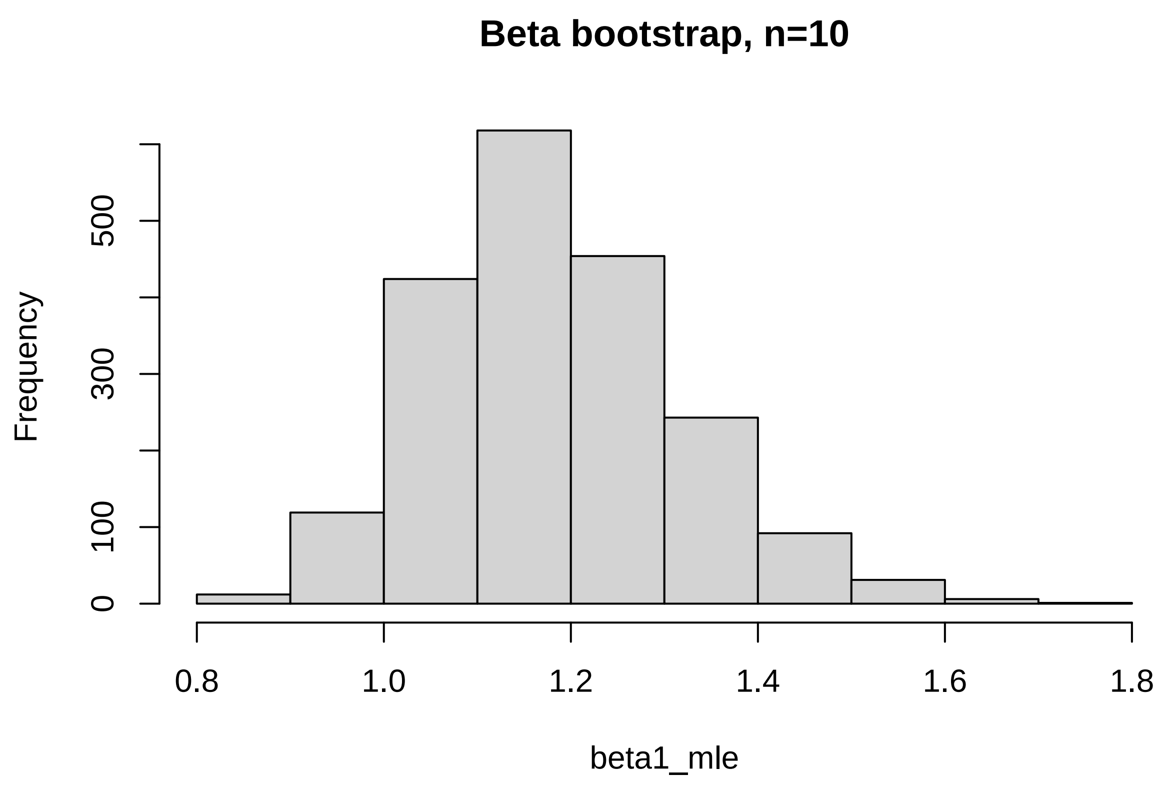


```
hist(sigma2, main="Sigma, n=10000")
```

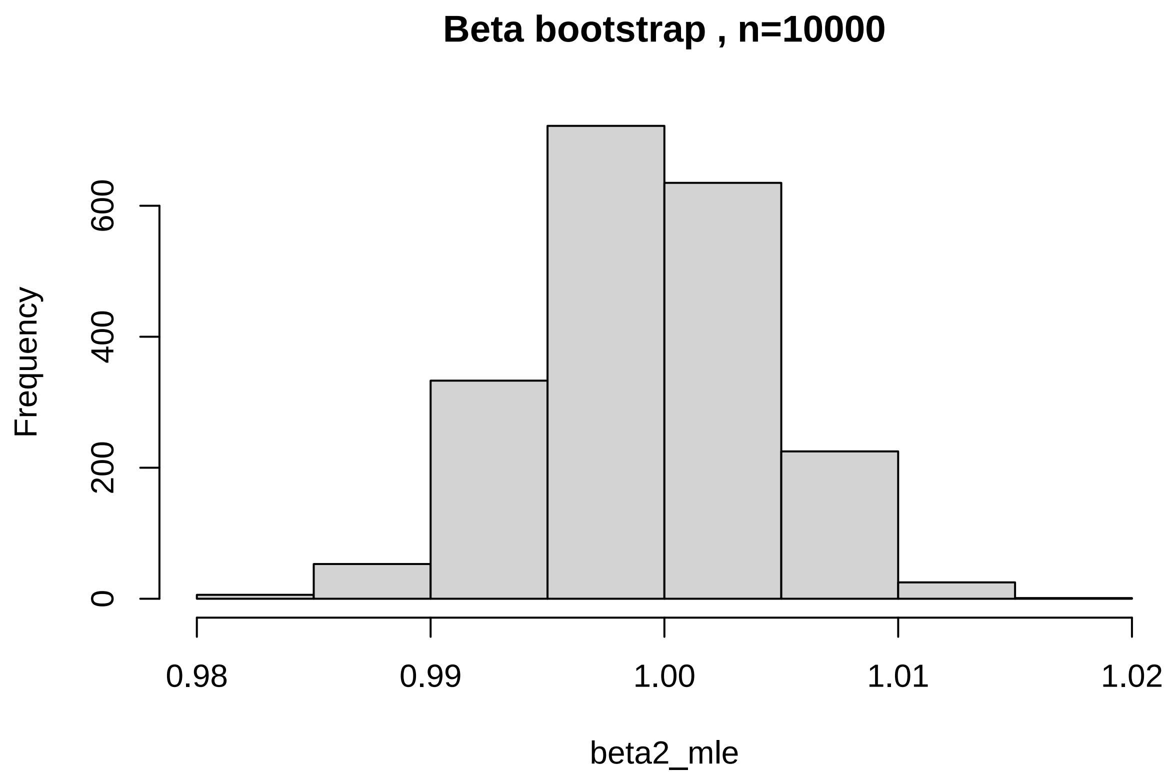


Mu fördelar sig OK oavsett n=10 eller n=10000 och topparna i båda histogramen ligger kring värdet vi vill ha.
Beta då n=10 har OK topp men lite skev fördelning. Beta då n=10000 är OK både fördelning- och toppvis.
Sigma är som Beta fast skevheten då n=10 är starkare.

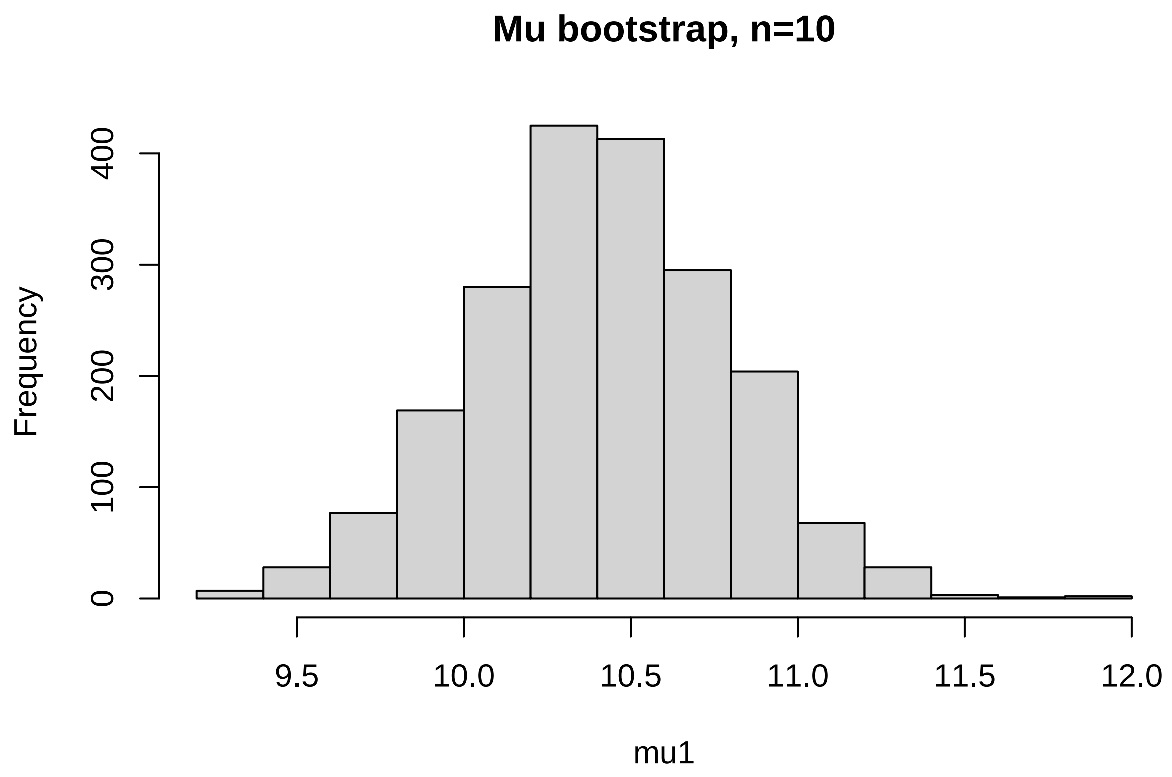
```r
# (2)

beta1_mle <- c()
beta2_mle <- c()
mu1 <- c()
mu2 <- c()
sigma1 <- c()
sigma2 <- c()

x1 <- rgamma(10, 4, 1)
x2 <- rgamma(10000, 4, 1)
y1 <- rnorm(10, 10, 2)
y2 <- rnorm(10000, 10, 2)

for (i in 1:2000) {
  beta1_mle[i] <- gamma_beta_mle(x = sample(x1, 10, replace = TRUE), alpha = 4)
  beta2_mle[i] <- gamma_beta_mle(x = sample(x2, 10000, replace = TRUE), alpha = 4)
  mu1[i] <- norm_mu_mle(x = sample(y1, 10, replace = TRUE))
  mu2[i] <- norm_mu_mle(x = sample(y2, 10000, replace = TRUE))
  sigma1[i] <- norm_sigma2_mle(x = sample(y1, 10, replace = TRUE))
  sigma2[i] <- norm_sigma2_mle(x = sample(y2, 10000, replace = TRUE))
}
hist(beta1_mle, main="Beta bootstrap, n=10")
```



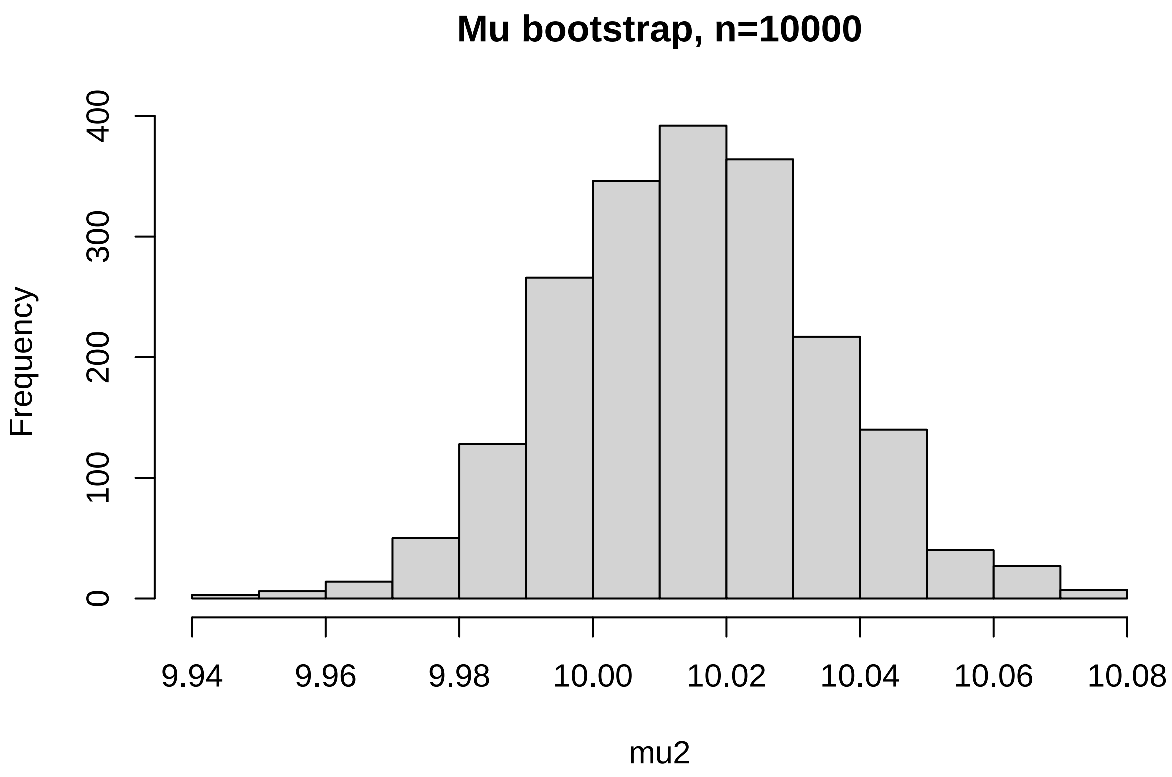
```r
hist(beta2_mle, main="Beta bootstrap , n=10000")
```



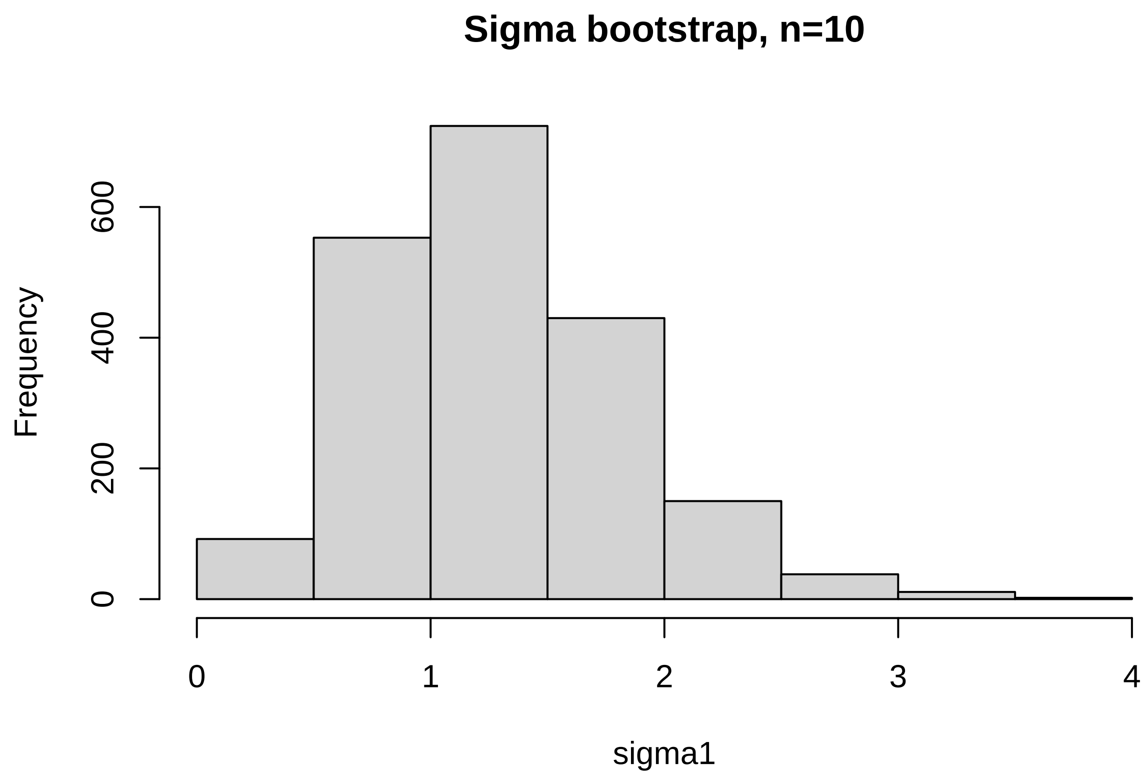
```
hist(mu1, main="Mu bootstrap, n=10")
```



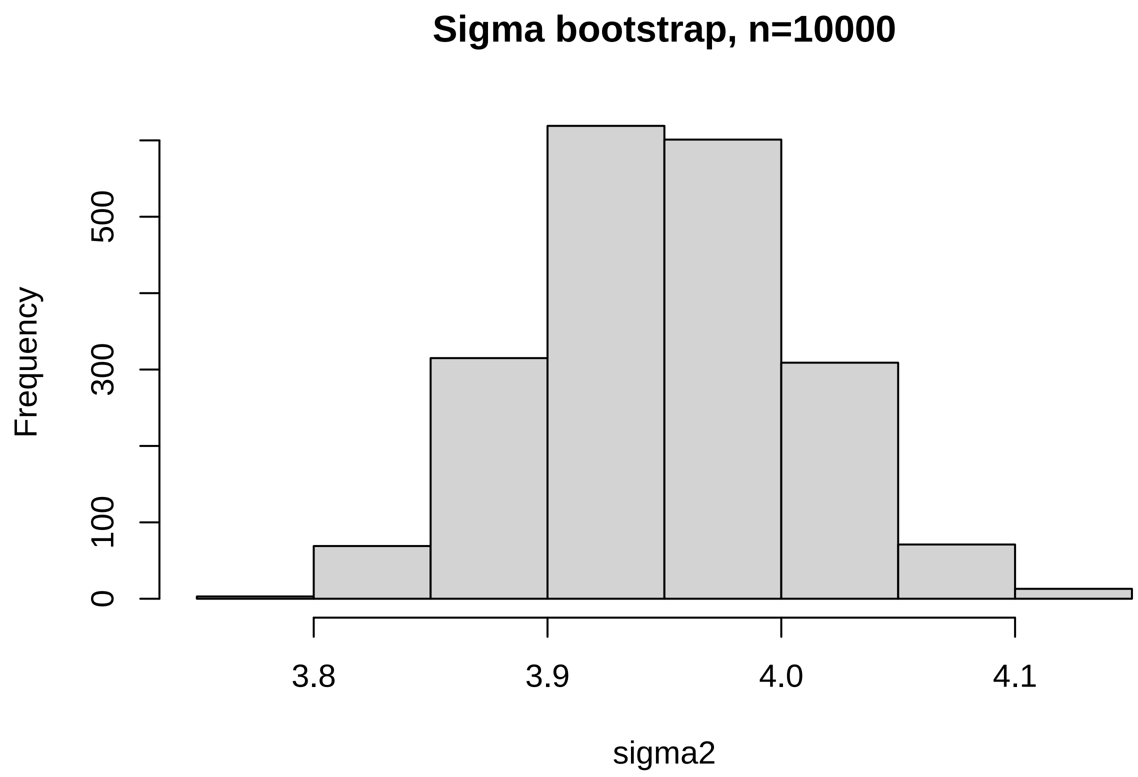
```
hist(mu2, main="Mu bootstrap, n=10000")
```



```r
hist(sigma1, main="Sigma bootstrap, n=10")
```



```
hist(sigma2, main="Sigma bootstrap, n=10000")
```



Samtliga n=10 histogram för bootstrap-metoden visar antingen på starkare skevhet eller ingen korrelation alls (sigma2) gentemot testet i (1).