Assignment 3

Amipriya Anand (220122)

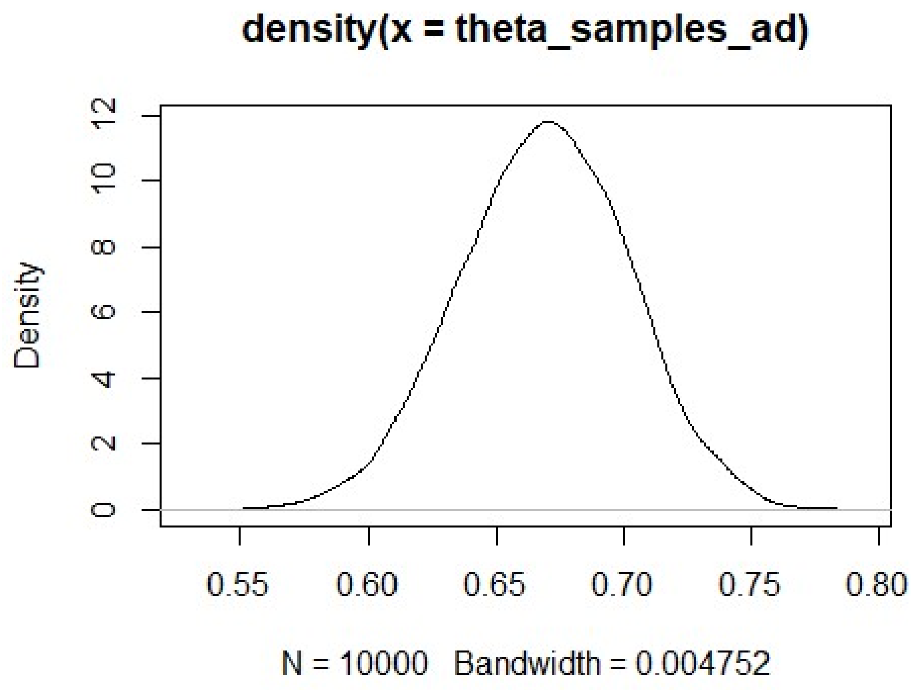
2024-06-16

❖ Part 1: Estimating the posterior distribution using different computational methods

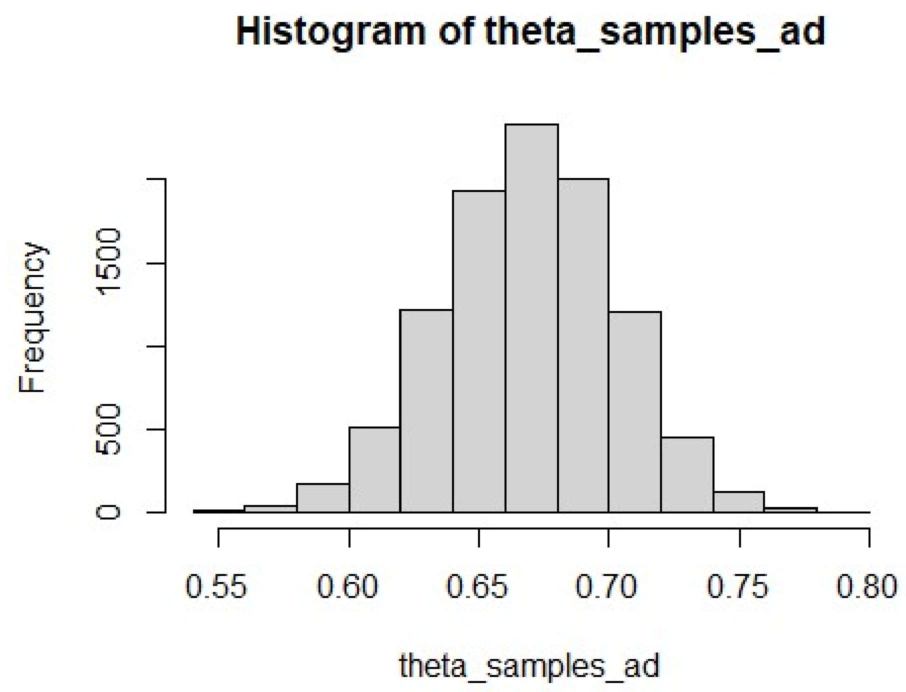
Let yi be the ith datapoint, yi ~ Binomial(n = 20,θ), θ ~ Beta(1,1). The analytically-derived posterior distribution of θ is a Beta distribution with shape parameters 135 and 67. θ|y ~ Beta(135,67)

1. Graph of the analytically-derived posterior distribution of ϑ.

```r
# analytically derived posterior distribution.
n = 20
y = c( 10, 15, 15, 14, 14, 14, 13, 11, 12, 16)
# As derived analytically and given
a = 135
b =  67

theta_samples_ad = rbeta(10000, a,b)
plot(density(theta_samples_ad))
```

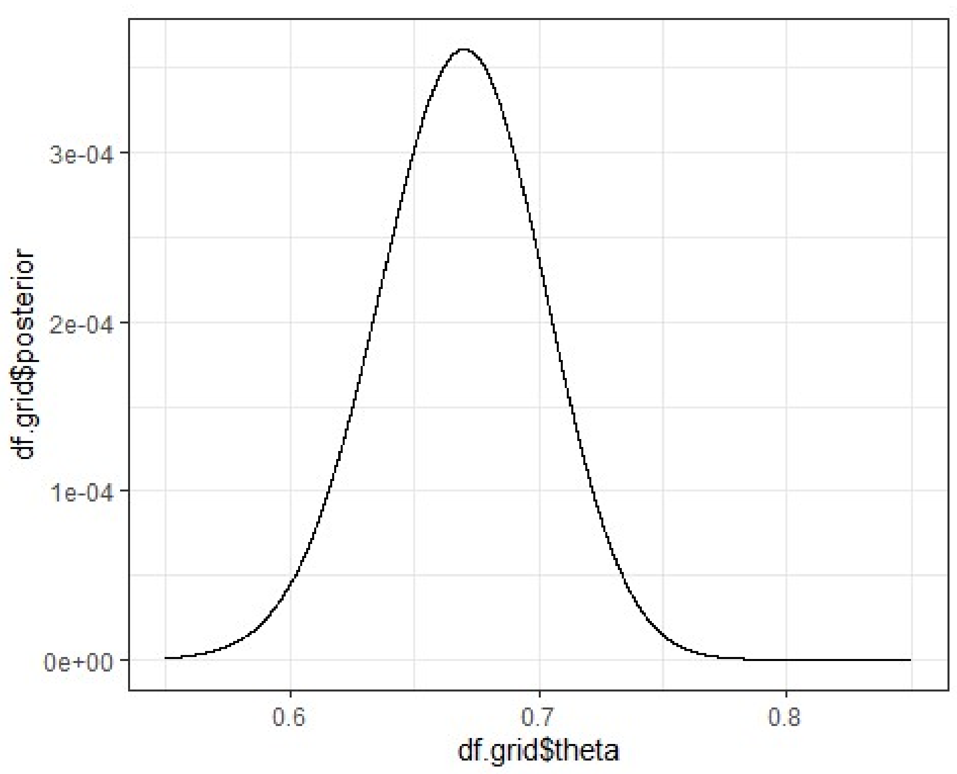


```r
hist(theta_samples_ad)
```



2. Using grid approximation to estimate the posterior density function of ϑ and graph the estimated posterior density function.

```r
library(ggplot2)

theta_grid = seq(0.55,0.85,length = 10000)
df.grid = data.frame(theta = theta_grid)
df.grid$lkl = rep(NA,length(theta_grid))
df.grid$prior = rep(NA,length(theta_grid))
for(i in 1:length(theta_grid))
{
  df.grid$lkl[i] = prod(dbinom(y,prob = theta_grid[i],size = 20))
  df.grid$prior[i] = dbeta(theta_grid[i],1,1)
}
ML = sum((df.grid$lkl)*(df.grid$prior))
ML

## [1] 4.686083e-06

df.grid$posterior = (df.grid$lkl)*(df.grid$prior)/ML
ggplot(df.grid,aes(x = df.grid$theta, y = df.grid$posterior))+
  geom_line()+
  theme_bw()

## Warning: Use of `df.grid$theta` is discouraged.
## ℹ Use `theta` instead.
## Warning: Use of `df.grid$posterior` is discouraged.
## ℹ Use `posterior` instead.
```



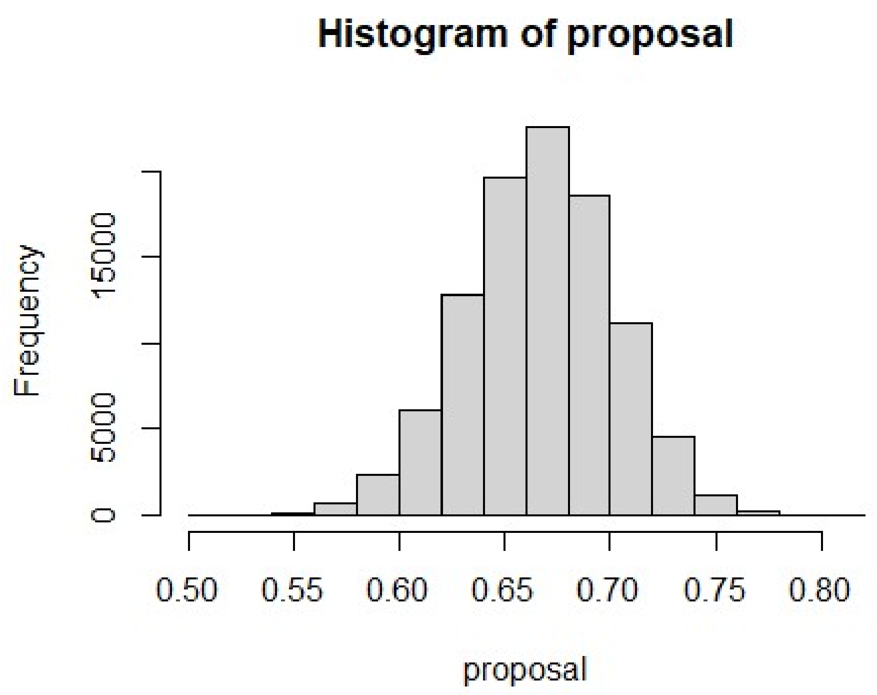
3. Using Monte Carlo integration to estimate the marginal likelihood of the model.

```
# for Monte Carlo Integration (considering sample of theta as above)
df.monte_carlo = data.frame(matrix(ncol = 2,nrow = 20000))
colnames(df.monte_carlo) = c("theta_Samples","likelihood")

for(i in 1:20000)
{
  theta_i =  rbeta(1,1,1)
  lkl = prod(dbinom(y,size = 20,prob = theta_i))
  df.monte_carlo[i,] = c(theta_i,lkl)
}

ML_mc = mean(df.monte_carlo$likelihood)
ML_mc

## [1] 1.429159e-10
```

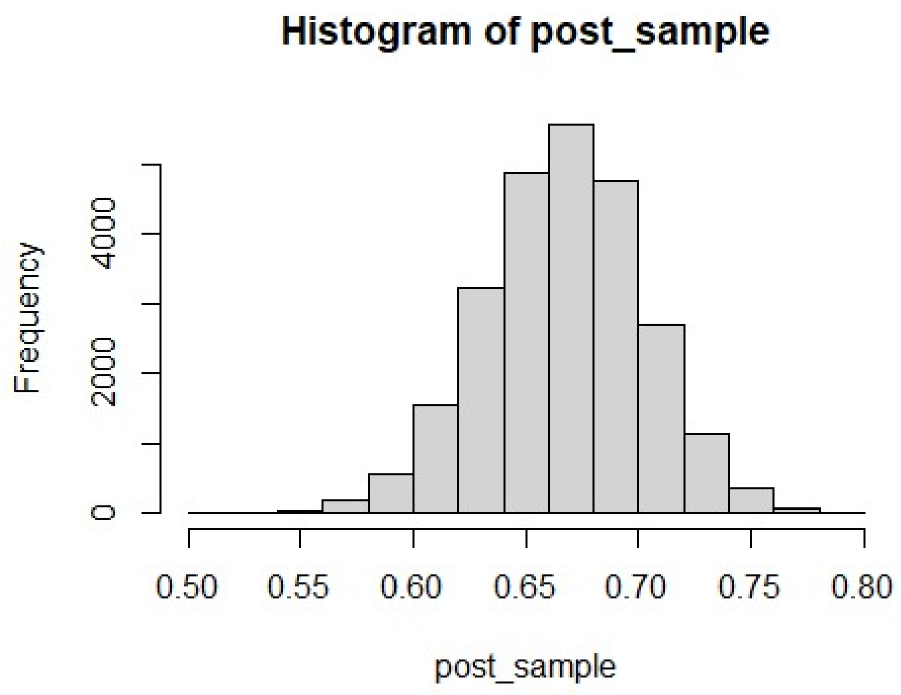
4. Using importance sampling to draw samples from the posterior distribution of ϑ.

```
## IMPORTANCE SAMPLING
#creating Samples
nsamp = 100000
y

##  [1] 10 15 15 14 14 14 13 11 12 16
```

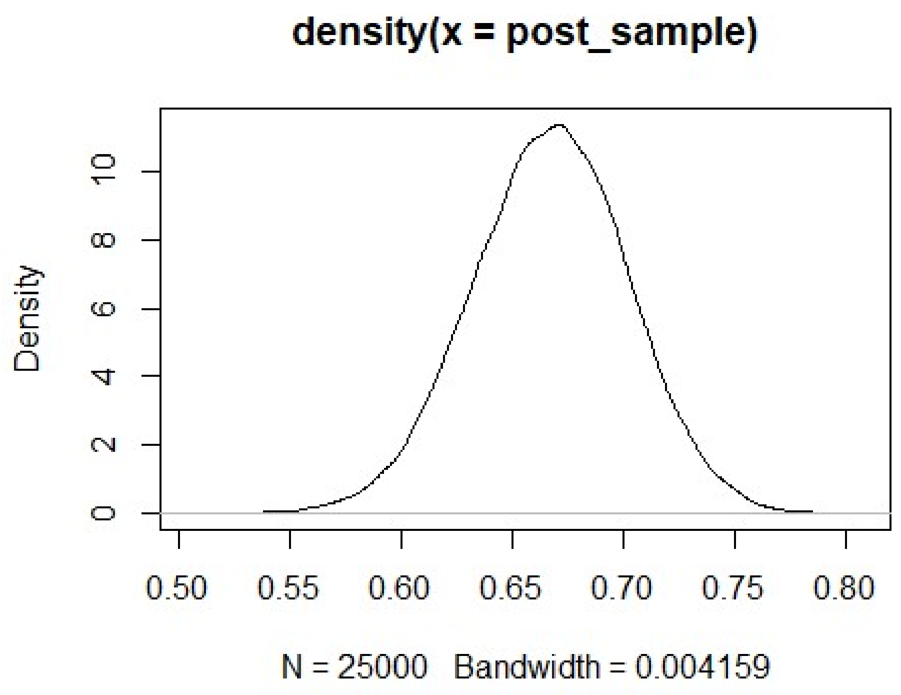
```r
##proposal distribution

proposal = rbeta(nsamp,120,60)
hist(proposal)
```



```r
library(reshape2)
library(ggplot2)

# these parameters of proposal distribution best aligns with the target
distribution
#drawing samples from proposal and plotting

theta_proposal = rbeta(nsamp,120,60)
weights_i = rep(NA,nsamp)
for(i in 1:nsamp)
{
  lkl_i = prod(dbinom(y,size = 20,prob = theta_proposal[i]))
  prior_i = dbeta(theta_proposal[i],1,1)
  proposal_i = dbeta(theta_proposal[i],120,60)
  weights_i = lkl_i*prior_i/proposal_i
}
df = data.frame(theta = theta_proposal,weights = weights_i)
# sample of posterior distribution
post_sample = sample(df$theta,size = (nsamp/4),prob = df$weights)
# plotting histogram
hist(post_sample)
```



```r
# plotting density plot
plot(density(post_sample))
```



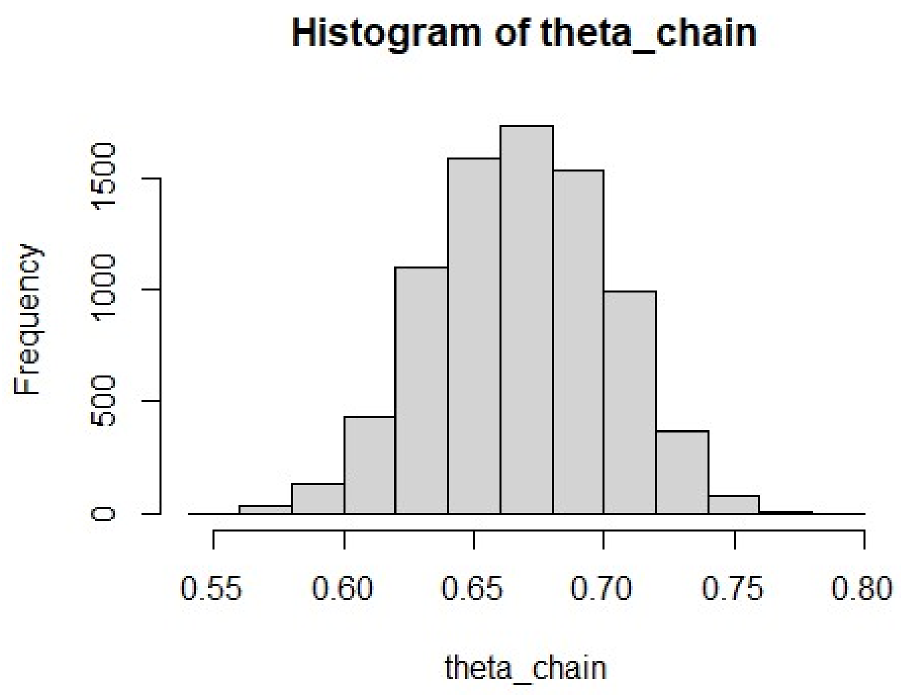
5. Using Markov chain Monte Carlo(MCMC) method to estimate the posterior distribution of ϑ.

```
y

##  [1] 10 15 15 14 14 14 13 11 12 16

nsamp = 8000
theta_chain = rep(NA,nsamp)

theta_chain[1] = rbeta(1,1,1)

i = 1
reject = 0
ML = 0
step = 0.055

while(i<nsamp)
{
  proposal_thata = rnorm(1,theta_chain[i],step)
  if(proposal_thata > 0 && proposal_thata < 1)
  {
    lkl_proposal = prod(dbinom(y,size = 20, prob = proposal_thata))
    lkl_current = prod(dbinom(y,size = 20, prob = theta_chain[i]))
    prior_proposal = dbeta(proposal_thata,1,1)
    prior_current = dbeta(theta_chain[i],1,1)
    backward_density = dnorm(theta_chain[i],proposal_thata,step)
    forward_density = dnorm(proposal_thata,theta_chain[i],step)
    # calculating the Hastings Ratio(H)
    H =
(lkl_proposal*prior_proposal*backward_density)/(lkl_current*prior_current*for
ward_density)
    #acceptance Criteria
    p_str = min(1,H)
    if(p_str > runif(1,0,1))
    {
      theta_chain[i+1] = proposal_thata
      i = i+1
      ML = (lkl_proposal*prior_proposal/forward_density)+ML
    }
    else
    {
      reject = reject + 1
    }
  }
}

# rate of rejection
rate_rej = reject*100/(reject+nsamp)
rate_rej

## [1] 43.47488
```

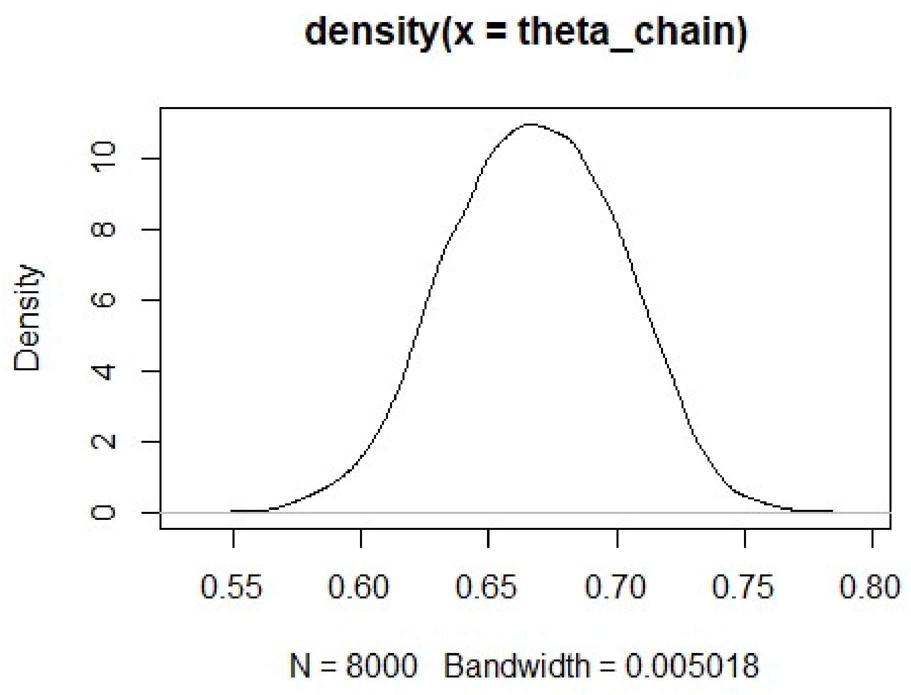
```r
#Marginal Likelihood
ML_estimate = ML/nsamp
ML_estimate
```

```
## [1] 2.052479e-10
```

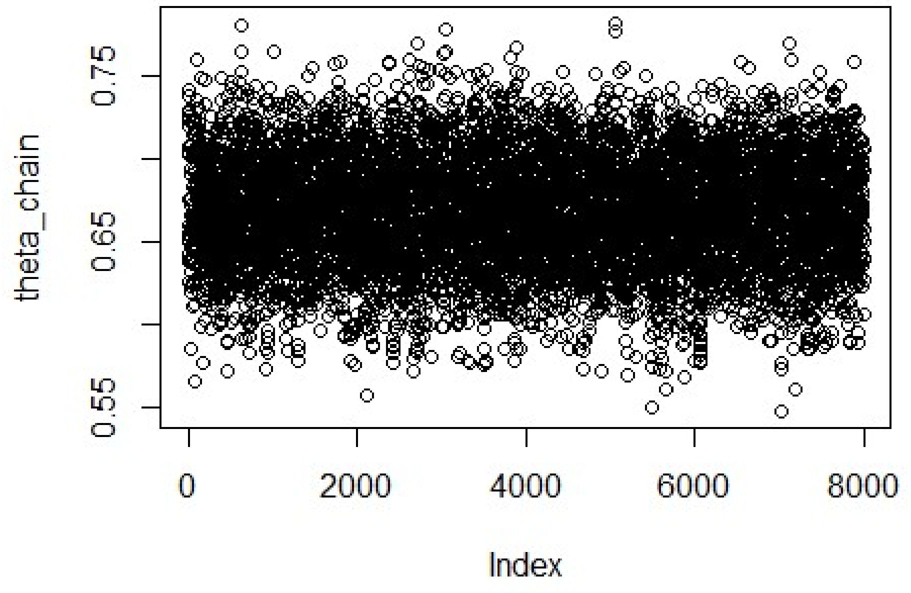
```r
##posterior distribution using MCMC
#histogram plot
hist(theta_chain)
```



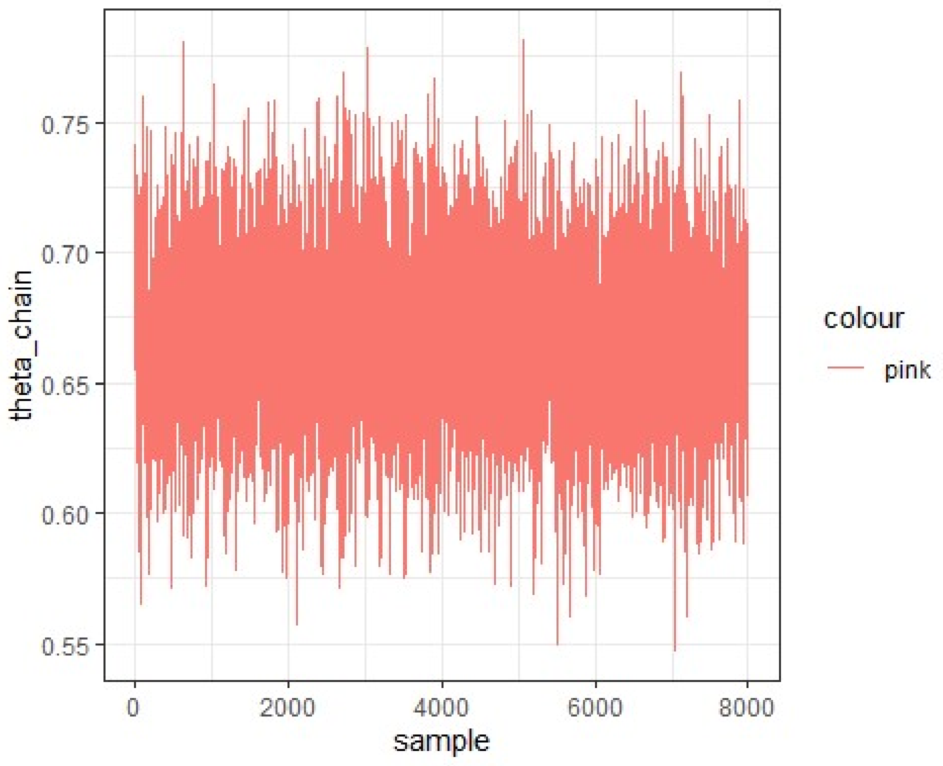
```r
#density plot
plot(density(theta_chain))
```



```
#plotting theta_chain
plot(theta_chain)
```



```
dat.MCMC = data.frame(theta_chain = theta_chain,sample = 1:nsamp)
ggplot(dat.MCMC,aes(y = theta_chain,x = sample,colour = "pink"))+
  geom_line()+
  theme_bw()
```

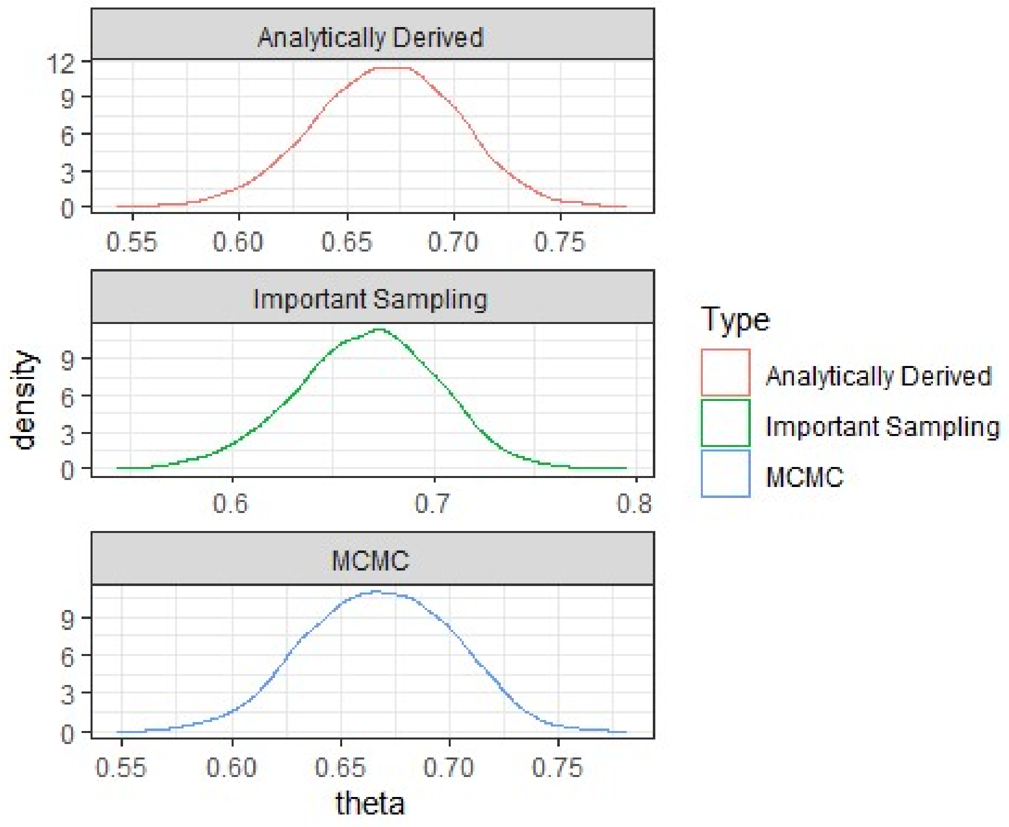


6. Graphical comparison of the posterior distributions of ϑ obtained using:

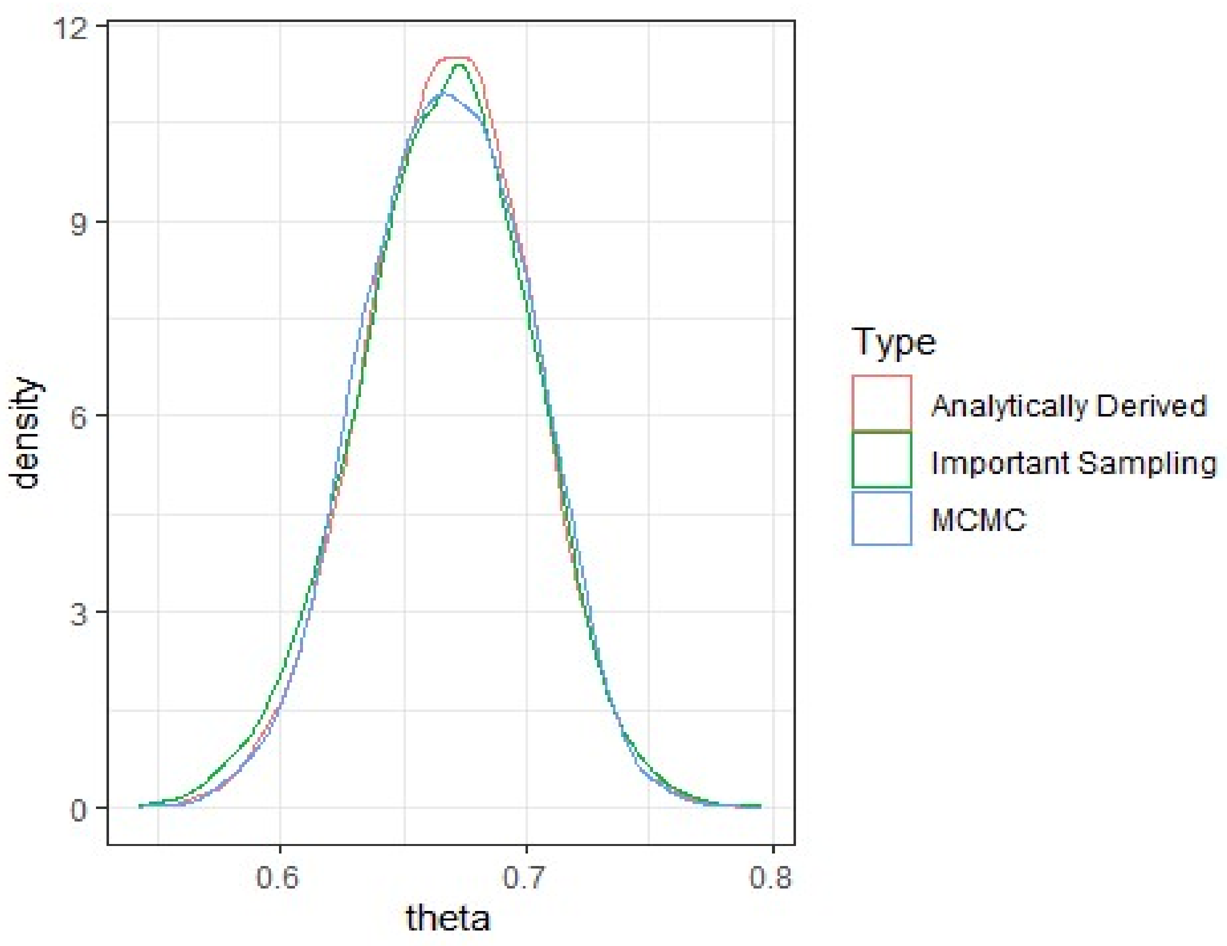
• Importance sampling • Markov chain Monte Carlo • Analytical derivation

```
#from MCMC samples
dat.posterior = data.frame(matrix(nrow = 8000,ncol = 3))
colnames(dat.posterior) = c("sample","theta","Type")
dat.posterior$sample = 1:8000
dat.posterior$theta = dat.MCMC$theta_chain
dat.posterior$Type = "MCMC"

#from importance sampling
post_sample = sample(df$theta,size = 8000,prob = df$weights)

#from analytically derived samples
a = 135
b =  67
theta_samples_ad = rbeta(8000,a,b)

dat.posterior = rbind(dat.posterior,data.frame(sample = 1:8000,theta = 
post_sample,Type = "Important Sampling"),data.frame(sample = 1:8000,theta = 
theta_samples_ad,Type = "Analytically Derived"))
```

```r
#altogether separately
ggplot(dat.posterior,aes(x = theta,group = Type,colour = Type))+
  geom_density()+
  facet_wrap(~Type,scales = "free", nrow = 3)+
  theme_bw()
```



```r
#in the same graph
ggplot(dat.posterior,aes(x = theta,group = Type,colour = Type))+
  geom_density()+
  theme_bw()
```



12
9
6
3
0
density
0.6
0.7
0.8
theta
Type
Analytically Derived
Important Sampling
MCMC

❖ Part 2: Writing your own sampler for Bayesian inference

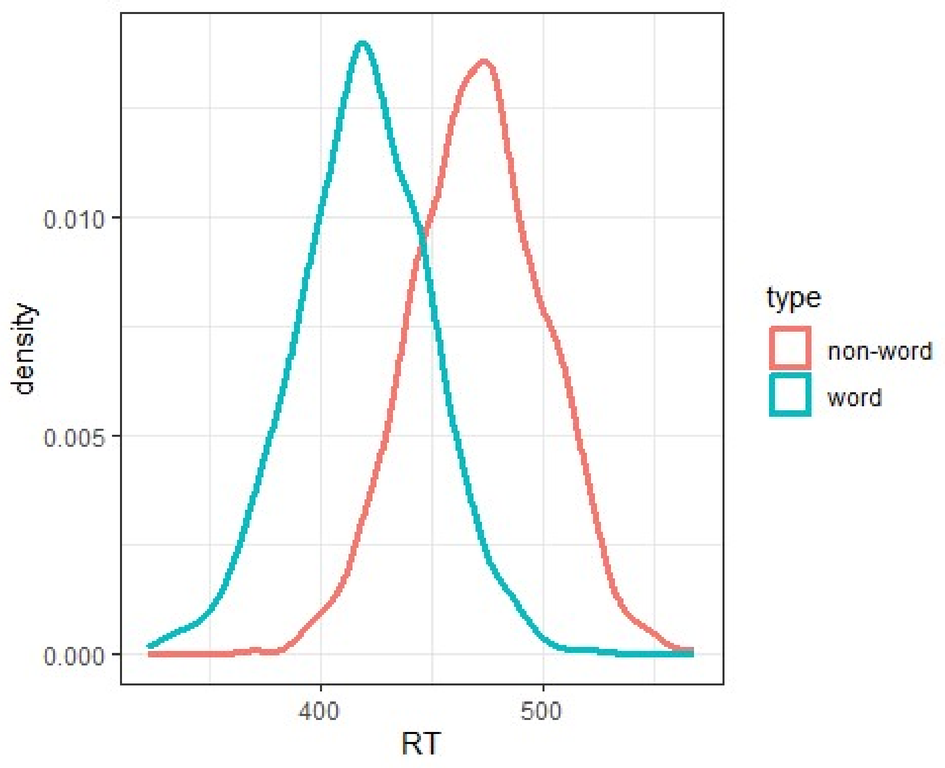
Likelihood assumption: RTi ~ Normal(μi,σ), such that, μi = α+β·typei where typei indicate whether the ith string is a word or a non-word. If the string i is a non word, the typei will be equal to 1. Priors: α ~ Normal(400,50) σ =30 β ~ Normal+(0,50)

```
library(reshape2)
library(ggplot2)
library(truncnorm)
#importing Data
dat <- read.table(

"https://raw.githubusercontent.com/yadavhimanshu059/CGS698C/main/notes/Data/w
ord-recognition-times.csv",
 sep=",",header = T)[,-1]
head(dat)
```

```
##       type       RT
## 1     word 423.1019
## 2     word 429.9432
## 3 non-word 486.9959
## 4 non-word 451.4400
## 5 non-word 482.2657
## 6 non-word 470.8003
```

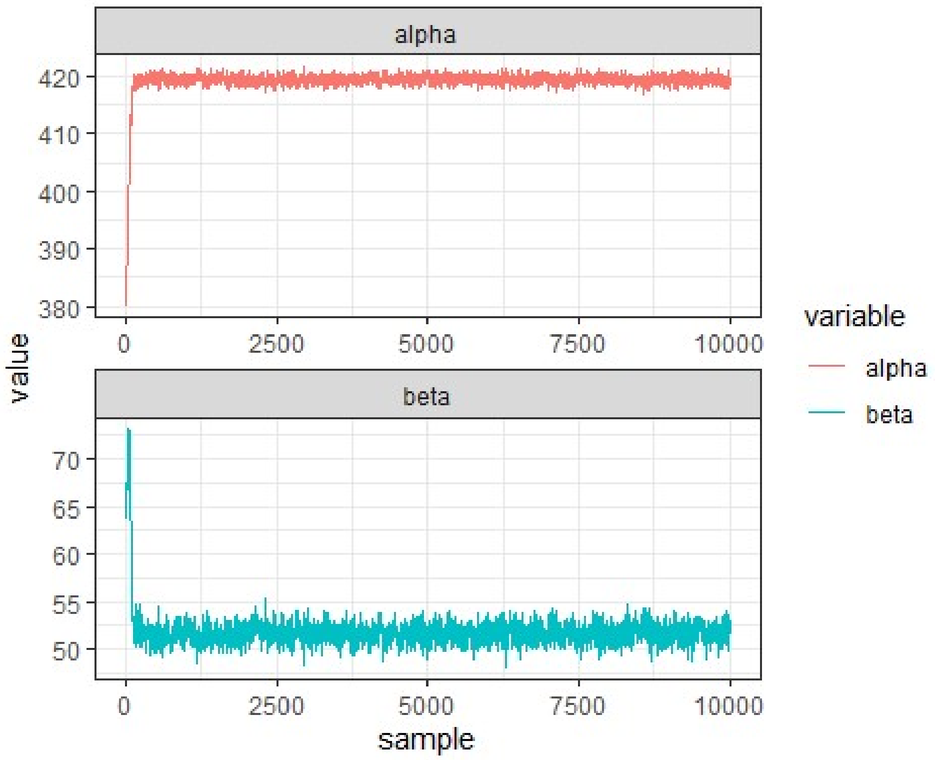
```
ggplot(dat,aes(x=RT,color=type))+geom_density(size=1.2)+theme_bw()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```r
RT_w = dat$RT[which(dat$type == "word")]
RT_nw = dat$RT[which(dat$type != "word")]

# implementing the MCMC using Metropolis Hasting Method

#sampling
nsamp = 10000
alpha_chain = rep(NA,nsamp)
beta_chain = rep(NA,nsamp)

#initialisation
sigma = 30
alpha_chain[1] = rnorm(1,400,50)
beta_chain[1] = rtruncnorm(1,a = 0,b=Inf,mean = 0,sd = 50)

#Evolution of Markov Chain
i = 1
reject = 0
step = 0.6
while(i<nsamp)
{
  proposed_alpha = rnorm(1,alpha_chain[i],step)
  proposed_beta = rtruncnorm(1,mean = beta_chain[i],sd = step,a = 0,b = Inf)

  #computing Likelihood and Prior
  #likelihood
  log_proposal_lkl = sum(dnorm(RT_w,proposed_alpha,sigma,log = TRUE))+
    sum(dnorm(RT_nw,(proposed_alpha +proposed_beta),sigma,log = TRUE))
  log_current_lkl = sum(dnorm(RT_w,alpha_chain[i],sigma,log = TRUE))+
    sum(dnorm(RT_nw,(alpha_chain[i] +beta_chain[i]),sigma,log = TRUE))
  #prior
  log_proposal_alpha = dnorm(proposed_alpha,400,50,log = TRUE)
  log_current_alpha = dnorm(alpha_chain[i],400,50,log = TRUE)

  log_proposal_beta = log(dtruncnorm(x = proposed_beta,mean = 0,sd = 50,
                                   a = 0,b = Inf))
  log_current_beta = log(dtruncnorm(x = beta_chain[i],mean = 0,sd = 50,
                                  a = 0,b = Inf))
  #proposal density
  log_backward_density_alpha =  dnorm(alpha_chain[i],proposed_alpha,
                                    step,log = TRUE)
  log_forward_density_alpha =  dnorm(proposed_alpha,alpha_chain[i],
                                   step,log = TRUE)

  log_backward_density_beta = log(dtruncnorm(x = beta_chain[i],a = 0,b = Inf,
                                          mean = proposed_beta,sd = step))
  log_forward_density_beta = log(dtruncnorm(x = proposed_beta,a = 0,b = Inf,
                                         mean = beta_chain[i],sd = step))

  #computing Hastings Ratio
  H = exp((log_proposal_lkl+log_proposal_alpha+log_proposal_beta+
             log_backward_density_alpha+log_backward_density_beta)-
            (log_current_lkl+log_current_alpha+log_current_beta+
               log_forward_density_alpha+log_forward_density_beta))
  #acceptance Crieteria
  p_str = min(1,H)
  if(p_str > runif(1,0,1))
  {
    alpha_chain[i+1] = proposed_alpha
    beta_chain[i+1] = proposed_beta
    i = i+1
  }
  else
  {
    reject = reject+1
  }
}

#rejection Rate
rate_rej = reject*100/(reject+nsamp)
rate_rej

## [1] 45.31634

#plotting alpha and beta parameters
params = data.frame(sample = 1:nsamp,alpha = alpha_chain,beta = beta_chain)
```

```r
ggplot(melt(params,id = "sample"),aes(y = value, x = sample,group = variable,
                                      colour = variable))+
  geom_line()+
  facet_wrap(~variable,scales = "free",nrow = 2)+
  theme_bw()
```

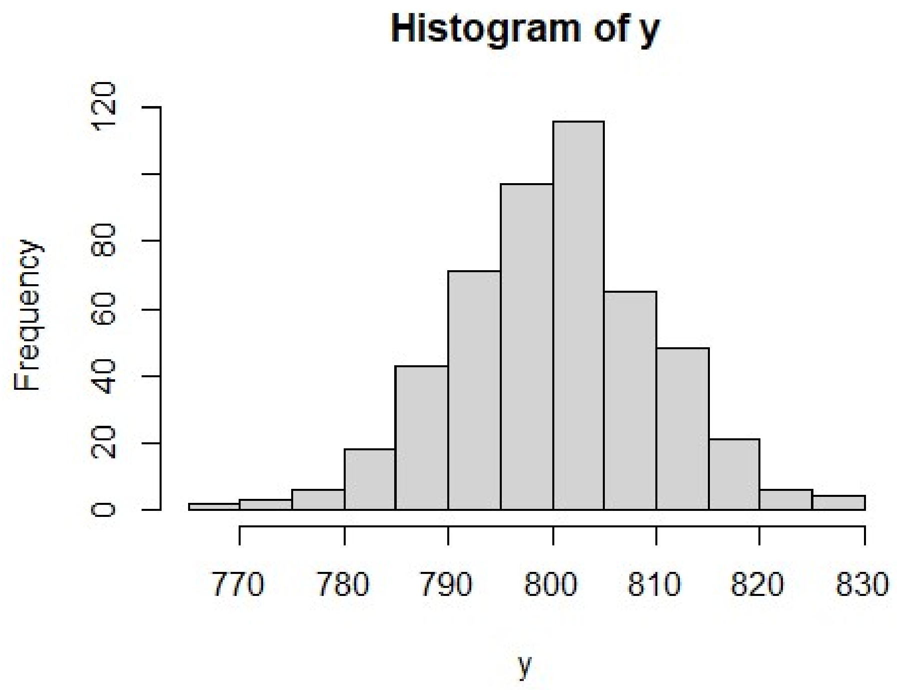


```r
##credible intervals ----
#for alpha
quantile(params$alpha,probs=c(.025,0.975))

##     2.5%    97.5%
## 417.9126 420.6792

#for beta
quantile(params$beta,probs=c(.025,0.975))

##     2.5%    97.5%
## 49.92405 53.73867

#mean of each parameters
mean(alpha_chain)

## [1] 419.2099

mean(beta_chain)

## [1] 51.82691
```

❖ Part 3: Hamiltonian Monte Carlo sampler

Given 500 data points that are assumed to come from a normal distribution with mean μ and variance σ2

```
true_mu <- 800
true_var <- 100 #sigma^2
y <- rnorm(500,mean=true_mu,sd=sqrt(true_var))
hist(y)
```



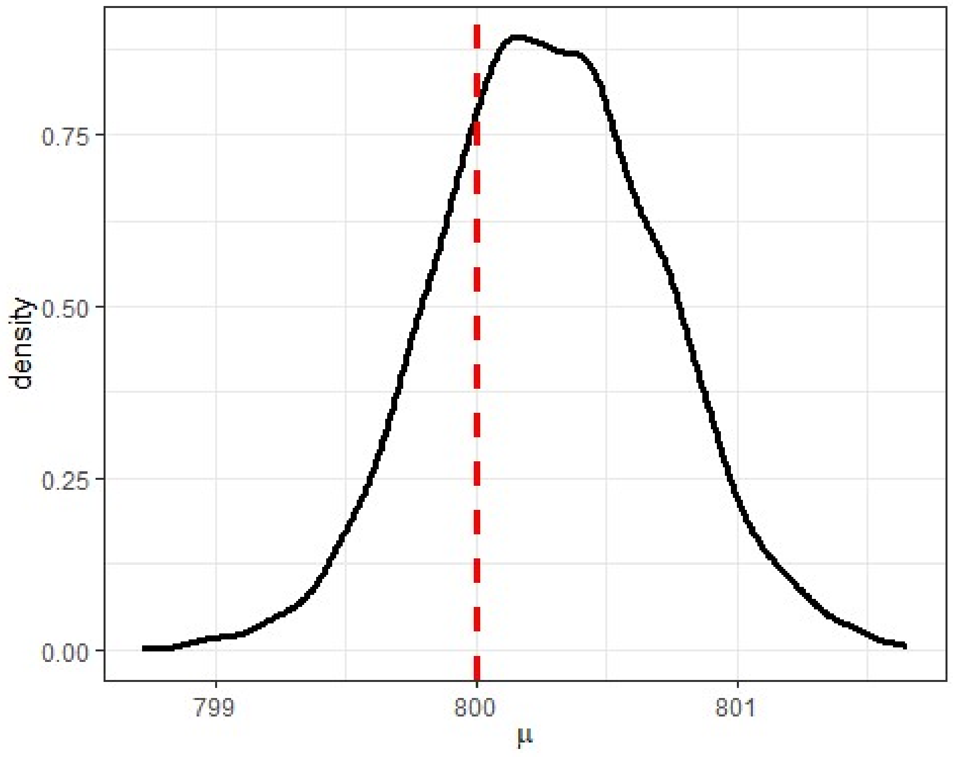
Model: Let yi be ith data point, yi ~ Normal(μ,σ2) μ ~Normal(m = 1000,s = 100) σ ~ Normal(a = 10,b = 2) Using the sample code given in the question

```
 #Gradient functions
gradient <- function(mu,sigma,y,n,m,s,a,b)
{
  grad_mu <- (((n*mu)-sum(y))/(sigma^2))+((mu-m)/(s^2))
  grad_sigma <- (n/sigma)-(sum((y-mu)^2)/(sigma^3))+((sigma-a)/(b^2))
  return(c(grad_mu,grad_sigma))
}

 #Potential energy function
V <- function(mu,sigma,y,n,m,s,a,b)
{
  nlpd = -
(sum(dnorm(y,mu,sigma,log=T))+dnorm(mu,m,s,log=T)+dnorm(sigma,a,b,log=T))
  nlpd
}
```

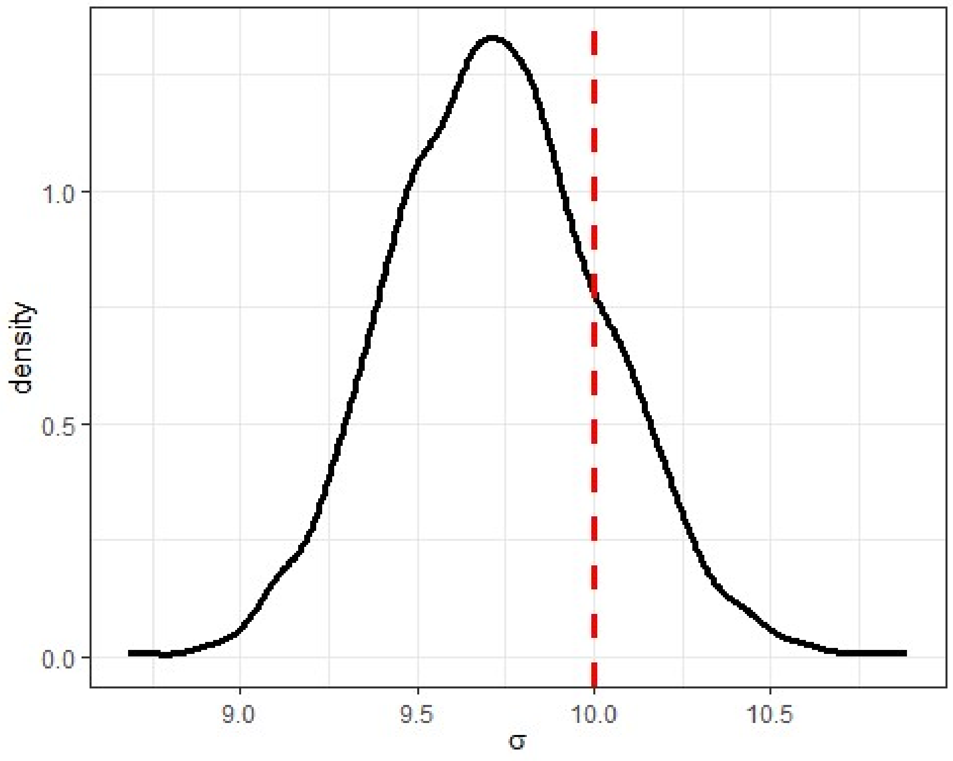
```
 #HMC sampler
HMC <- function(y,n,m,s,a,b,step,L,initial_q,nsamp,nburn)
{
  mu_chain <- rep(NA,nsamp)
  sigma_chain <- rep(NA,nsamp)
  reject <- 0
  #Initialization of Markov chain
  mu_chain[1] <- initial_q[1]
  sigma_chain[1] <- initial_q[2]
  #Evolution of Markov chain
  i <- 1
  while(i < nsamp)
  {
    q <- c(mu_chain[i],sigma_chain[i]) # Current position of the particle
    p <- rnorm(length(q),0,1)
    current_q <- q
    current_p <- p
    # Generate random momentum at the current position
    current_V = V(current_q[1],current_q[2],y,n,m,s,a,b) # Current potential
energy
    current_T = sum(current_p^2)/2
    #   Current kinetic energy
    # Take L leapfrog steps
    for(l in 1:L)
    {
      # Change in momentum in 'step/2' time
      p <- p-((step/2)*gradient(q[1],q[2],y,n,m,s,a,b))
      # Change in position in 'step' time
      q <- q + step*p
      # Change in momentum in 'step/2' time
      p <- p-((step/2)*gradient(q[1],q[2],y,n,m,s,a,b))
    }
    proposed_q <- q
    proposed_p <- p
    proposed_V = V(proposed_q[1],proposed_q[2],y,n,m,s,a,b) # Proposed
potential energy
    proposed_T = sum(proposed_p^2)/2
    # Proposed kinetic energy
    accept.prob <- min(1,exp(current_V+current_T-proposed_V-proposed_T))
    # Accept/reject the proposed position q
    if(accept.prob>runif(1,0,1))
    {
      mu_chain[i+1] <- proposed_q[1]
      sigma_chain[i+1] <- proposed_q[2]
      i <- i+1
    }
    else
    {
      reject <- reject+1
    }
  }
  posteriors <- data.frame(mu_chain,sigma_chain)[-(1:nburn),]
  posteriors$sample_id <- 1:nrow(posteriors)
  return(posteriors)
}
```

Exercise 3.1: Estimate and plot the posteriors for μ and σ.

```
library(ggplot2)
df.posterior = HMC(y=y,n=length(y),          # data
                   m=1000,s=20,a=10,b=2,     # priors
                   step=0.02,                # step-size
                   L=12,                     # no. of leapfrog steps
                   initial_q=c(1000,11),     # Chain initialization
                   nsamp=6000,               # total number of samples
                   nburn=2000)               # number of burn-in samples
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



```
#plot of sigma_chain
ggplot(df.posterior,aes(x = sigma_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept=10,size=1.2,colour="red",linetype = "dashed")
```

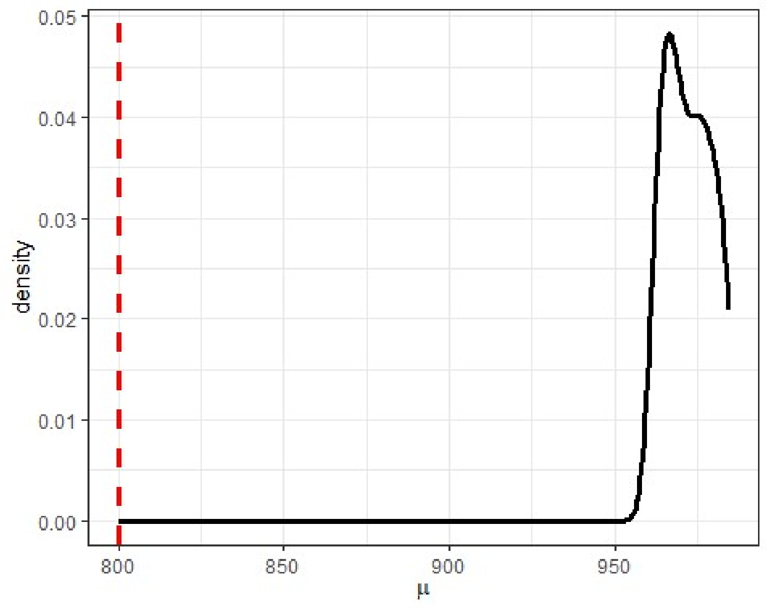


Exercise 3.2: Check posterior sensitivity to the total number of samples.

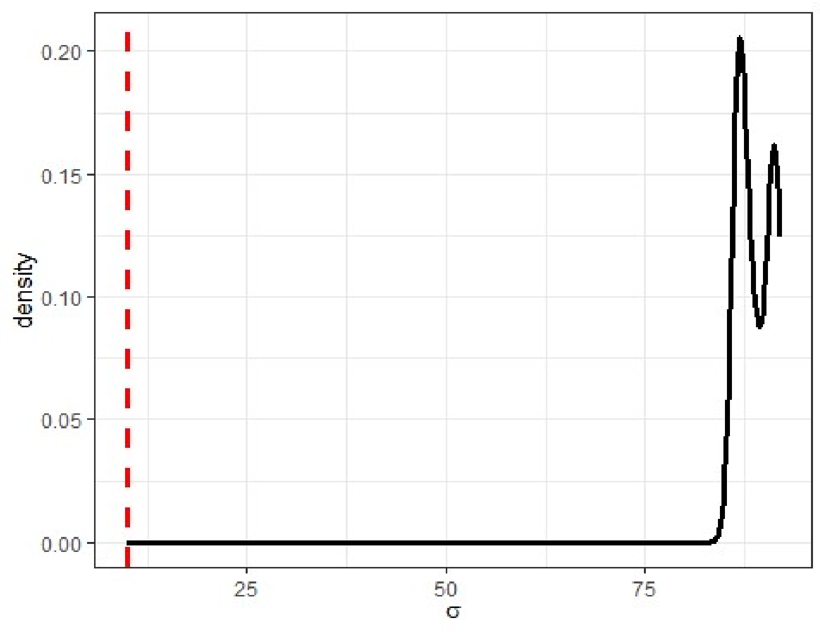
To estimate and compare the posteriors obtained for the following values of total number of samples, nsamp

• nsamp = 100

```
#when nsamp = 100
nsamp = 100
df.posterior = HMC(y=y,n=length(y),           # data
                   m=1000,s=20,a=10,b=2,      # priors
                   step=0.02,                 # step-size
                   L=12,                      # no. of leapfrog steps
                   initial_q=c(1000,11),      # Chain initialization
                   nsamp=nsamp,               # total number of samples
                   nburn=nsamp/3)             # number of burn-in samples

df.posterior$type = "nsamp = 100"
posterior = df.posterior
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```

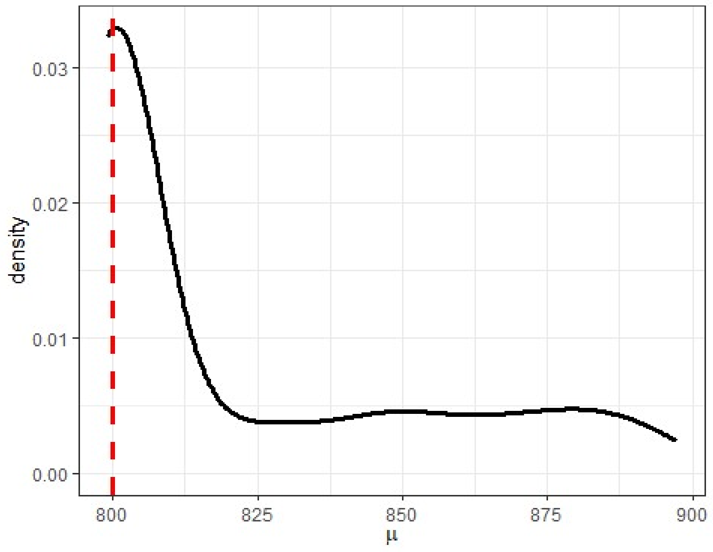


```
#plot of sigma_chain
ggplot(df.posterior,aes(x = sigma_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept=10,size=1.2,colour="red",linetype = "dashed")
```

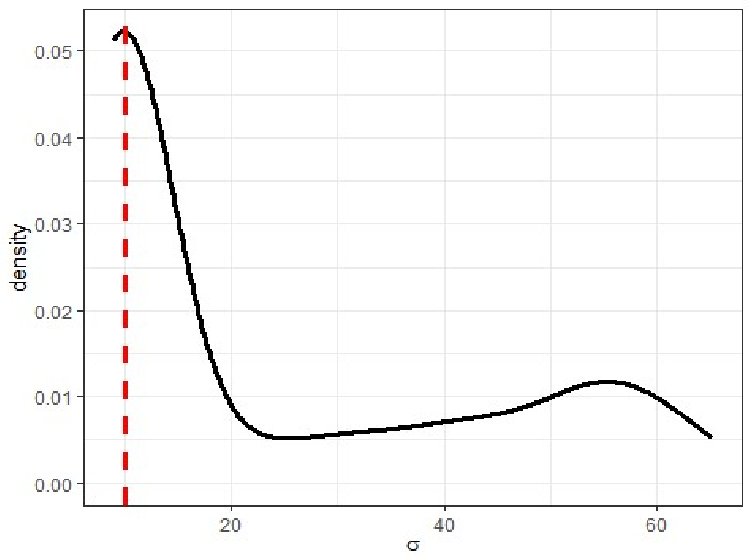


- nsamp = 1000

```r
#when nsamp = 1000
nsamp = 1000
df.posterior = HMC(y=y,n=length(y),            # data
                   m=1000,s=20,a=10,b=2,       # priors
                   step=0.02,                  # step-size
                   L=12,                       # no. of leapfrog steps
                   initial_q=c(1000,11),       # Chain initialization
                   nsamp=nsamp,                # total number of samples
                   nburn=nsamp/3)              # number of burn-in samples

df.posterior$type = "nsamp = 1000"
posterior = rbind(posterior,df.posterior)
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```

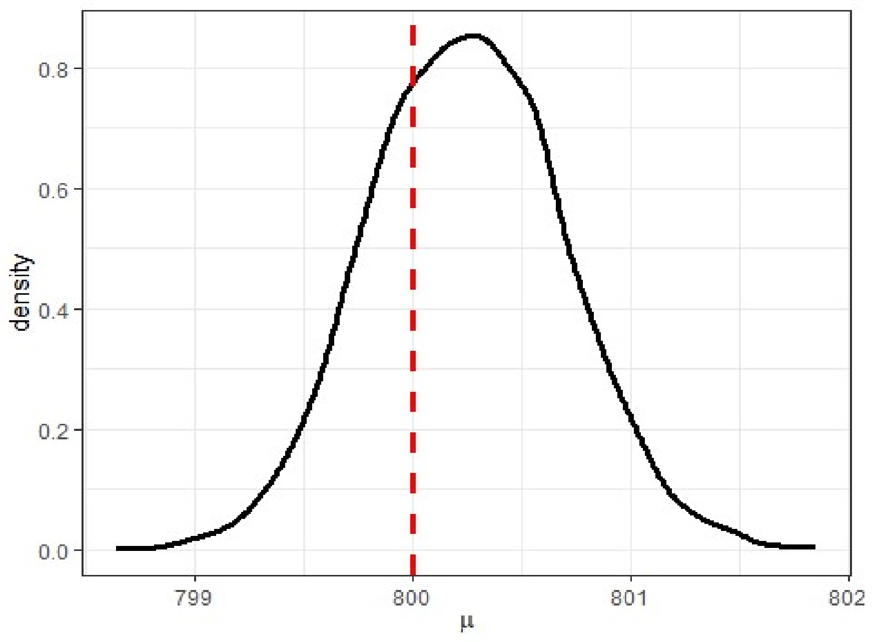


```r
#plot of sigma_chain
ggplot(df.posterior,aes(x = sigma_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept=10,size=1.2,colour="red",linetype = "dashed")
```

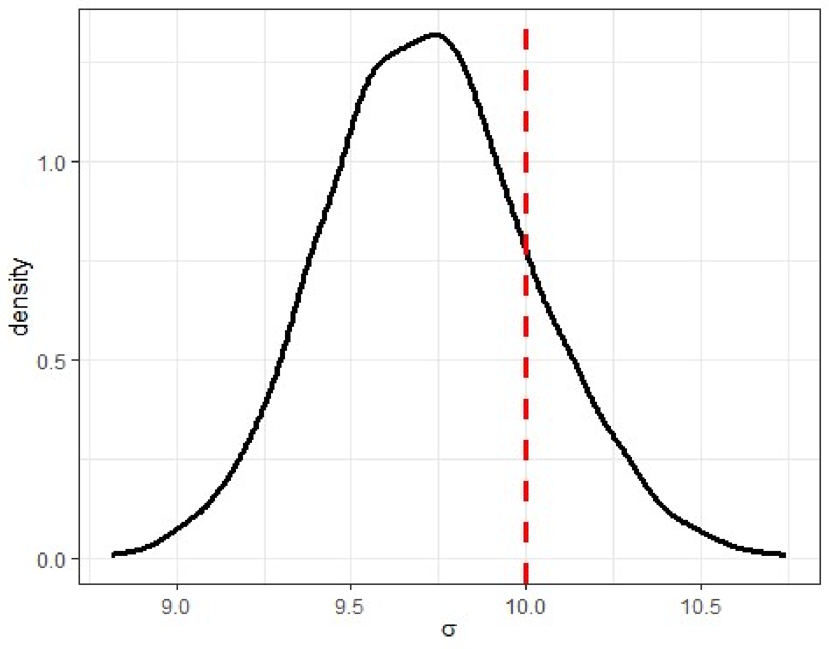


- nsamp = 6000

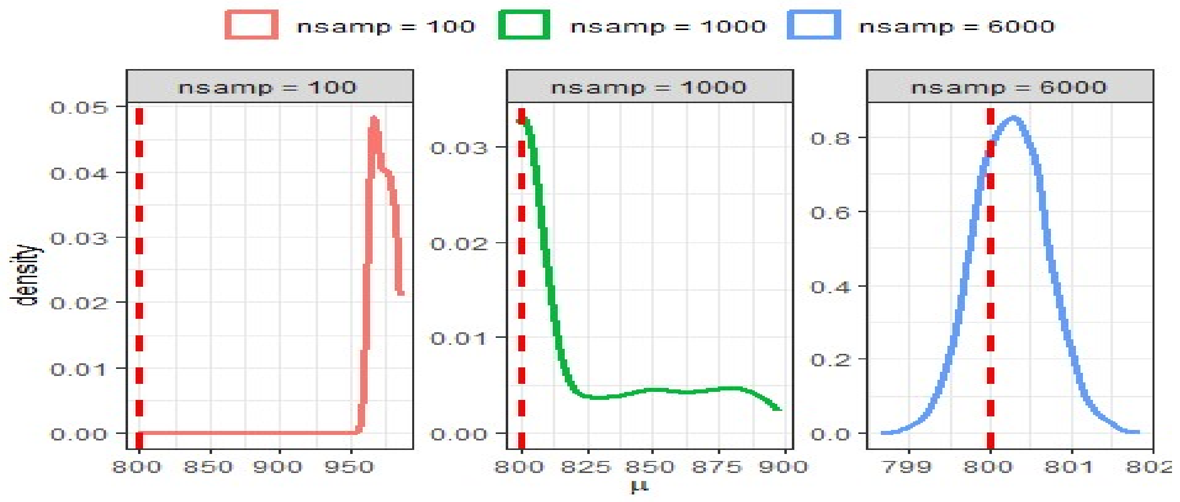
```
#when nsamp = 6000
nsamp = 6000
df.posterior = HMC(y=y,n=length(y),            # data
                    m=1000,s=20,a=10,b=2,     # priors
                    step=0.02,                # step-size
                    L=12,                     # no. of leapfrog steps
                    initial_q=c(1000,11),     # Chain initialization
                    nsamp=nsamp,              # total number of samples
                    nburn=nsamp/3)            # number of burn-in samples

df.posterior$type = "nsamp = 6000"
posterior = rbind(posterior,df.posterior)
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



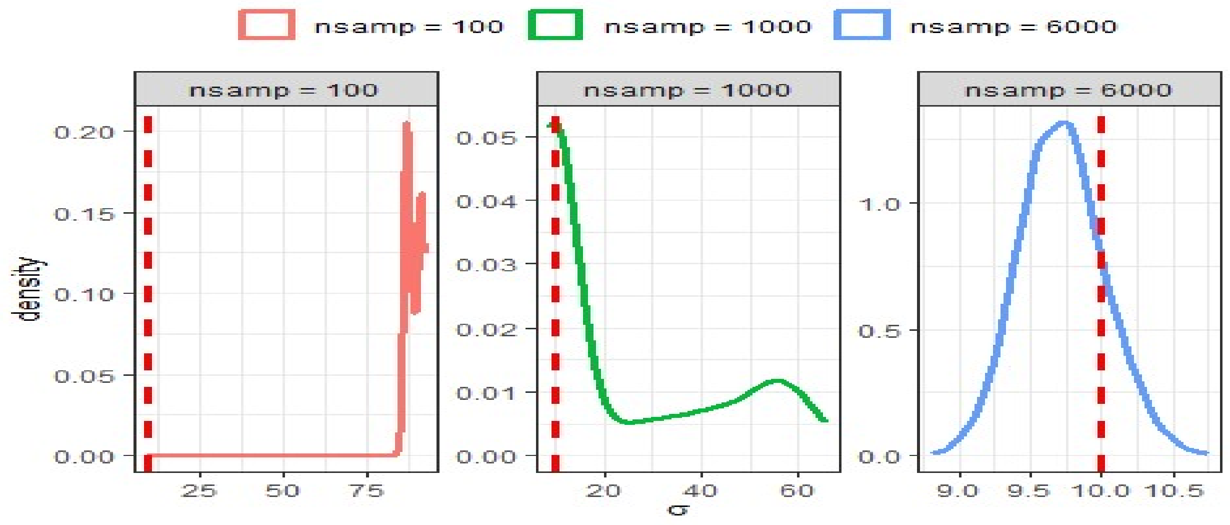
```r
#plot of sigma_chain
ggplot(df.posterior,aes(x = sigma_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept=10,size=1.2,colour="red",linetype = "dashed")
```



```
#plotting altogether of mu_chain
ggplot(posterior,aes(x = mu_chain,group = type,colour = type))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")+
  facet_wrap(~type,scales = "free")
```



```
#plot of sigma_chain
ggplot(posterior,aes(x = sigma_chain,group = type,colour = type))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept=10,size=1.2,colour="red",linetype = "dashed")+
  facet_wrap(~type,scales = "free")
```

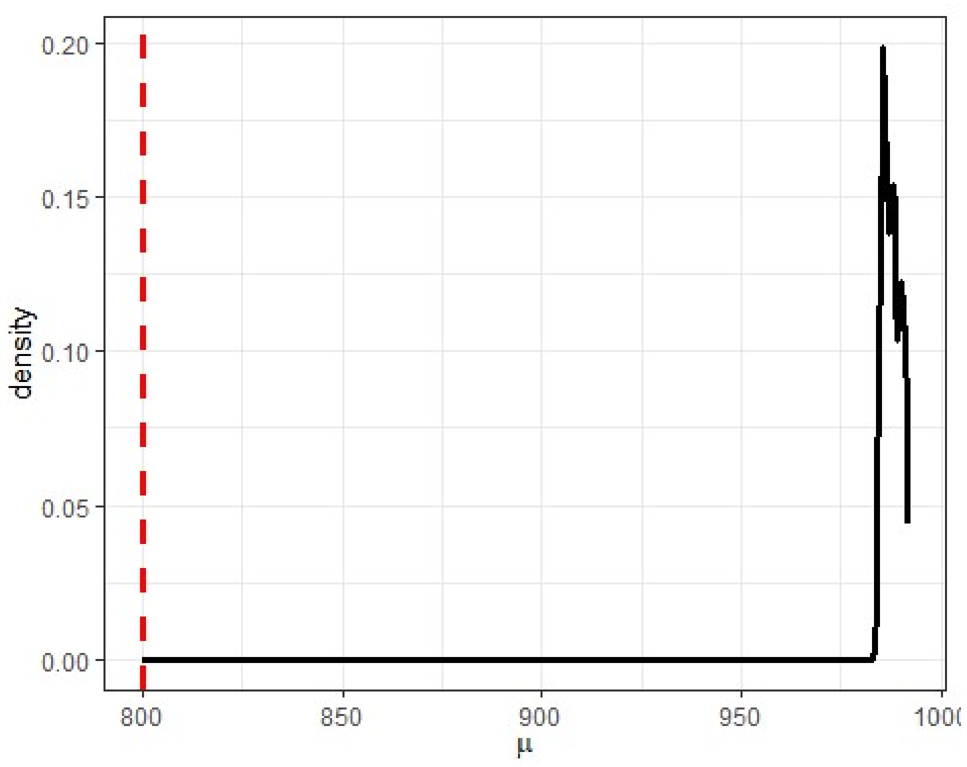


As can be seen from the above graph that as the sample size decreases the posterior distribution for both the parameters leads to less reliable estimates.

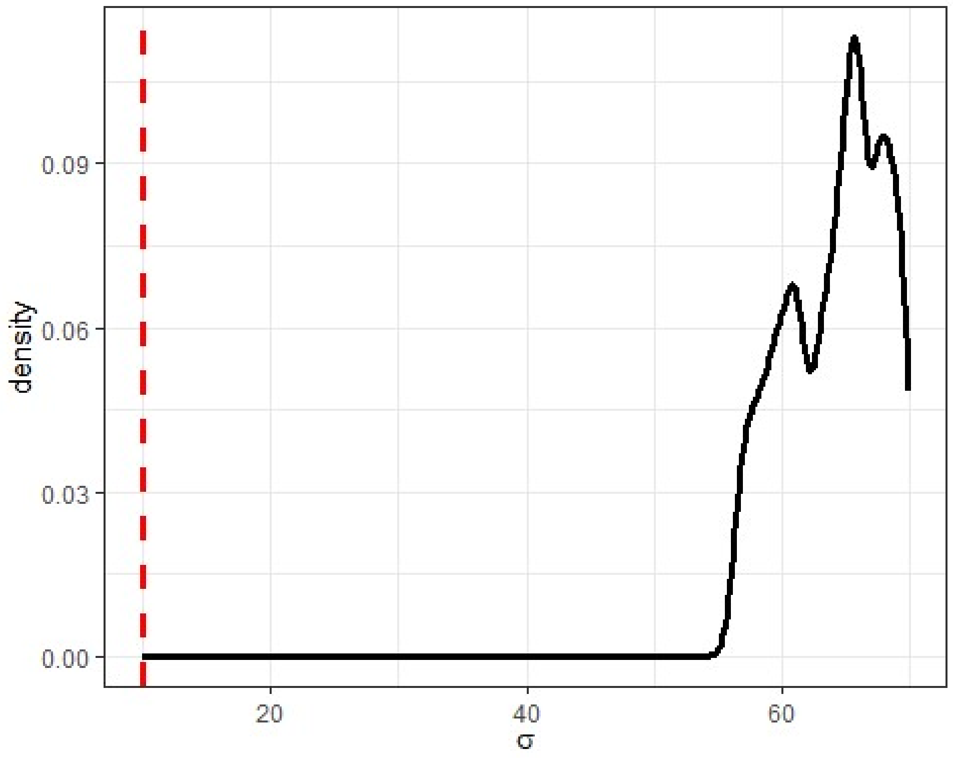
Exercise 3.3: Estimate and compare the posteriors obtained when step-size changes.

• step = 0.001 • step = 0.005 • step = 0.02

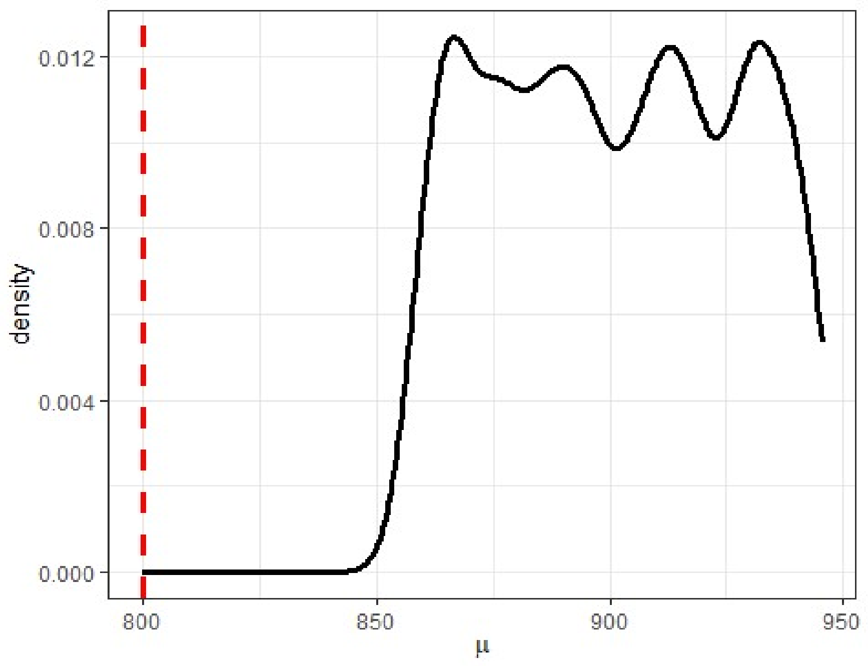
```
#when step size = 0.001
step_size = 0.001
df.posterior = HMC(y=y,n=length(y),                   # data
                    m=1000,s=20,a=10,b=2,              # priors
                    step=step_size,                    # step-size
                    L=12,                              # no. of leapfrog steps
                    initial_q=c(1000,11),              # Chain initialization
                    nsamp=6000,                        # total number of samples
                    nburn=2000)                        # number of burn-in samples

df.posterior$type = "step size = 0.001"
posterior = df.posterior
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



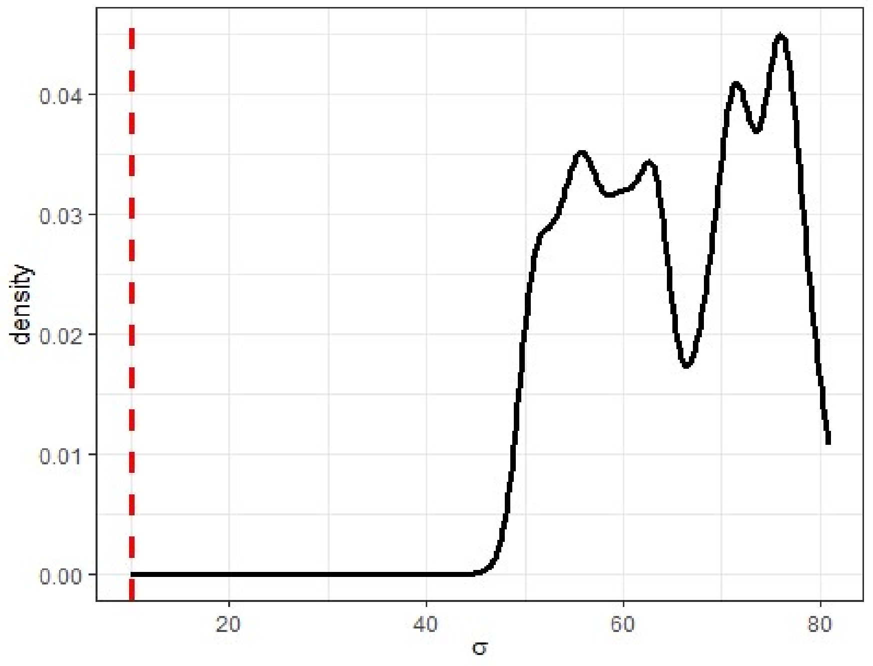
```r
#plot of sigma_chain
ggplot(df.posterior,aes(x = sigma_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept=10,size=1.2,colour="red",linetype = "dashed")
```



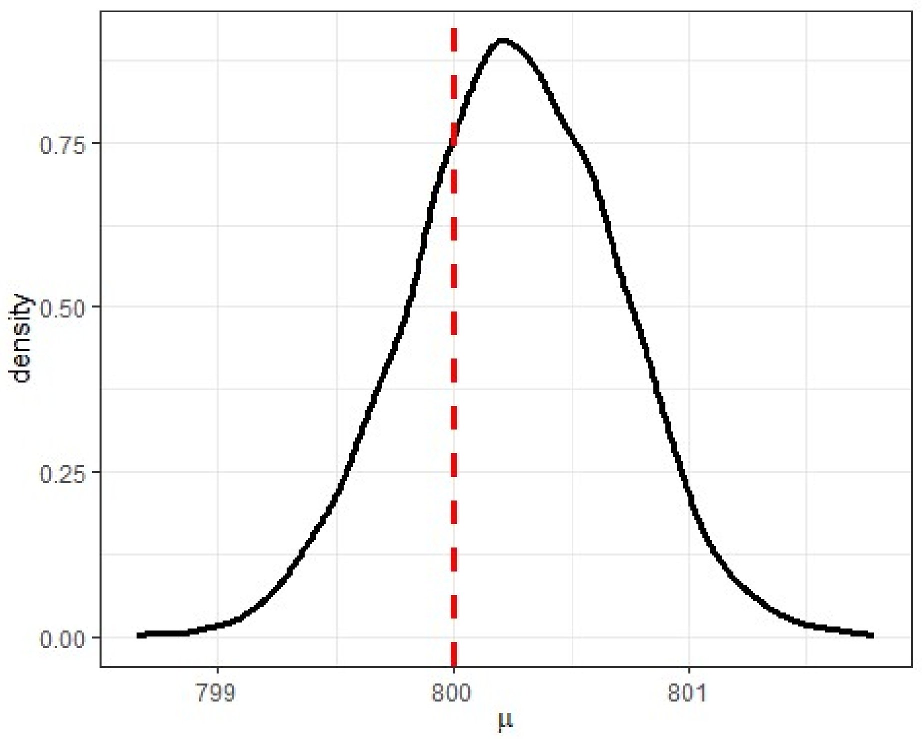
```r
#when step size = 0.005
step_size = 0.005
df.posterior = HMC(y=y,n=length(y),          # data
                    m=1000,s=20,a=10,b=2,    # priors
                    step=step_size,               # step-size
                    L=12,                    # no. of leapfrog steps
                    initial_q=c(1000,11),    # Chain initialization
                    nsamp=6000,             # total number of samples
                    nburn=2000)           # number of burn-in samples

df.posterior$type = "step size = 0.005"
posterior = rbind(posterior,df.posterior)
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



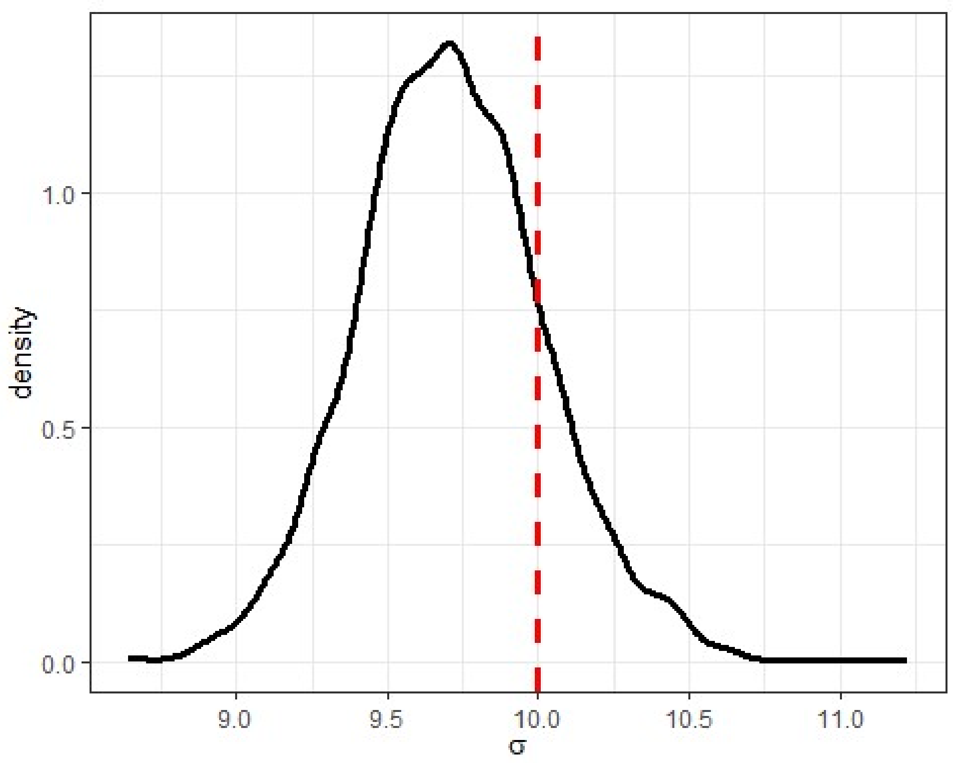
```
#plot of sigma_chain
ggplot(df.posterior,aes(x = sigma_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept=10,size=1.2,colour="red",linetype = "dashed")
```



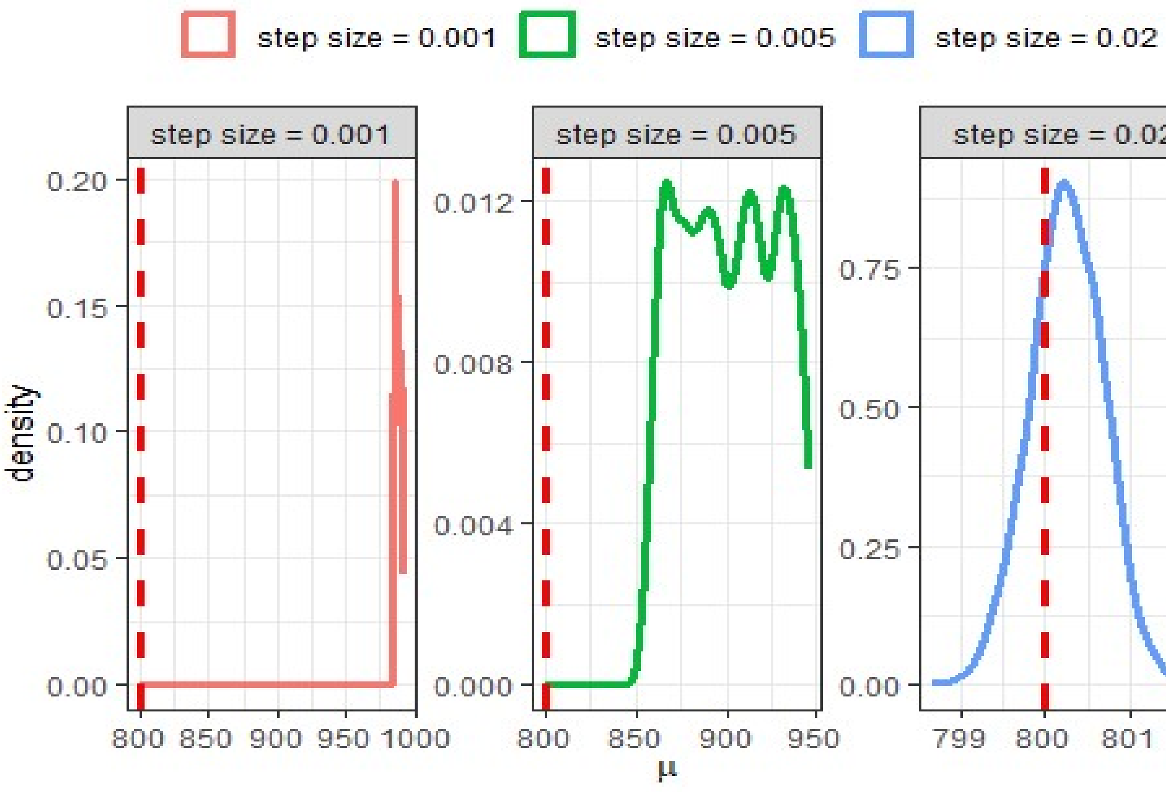
```
#when step size = 0.02
step_size = 0.02
df.posterior = HMC(y=y,n=length(y),          # data
                    m=1000,s=20,a=10,b=2,     # priors
                    step=step_size,               # step-size
                    L=12,                     # no. of leapfrog steps
                    initial_q=c(1000,11),     # Chain initialization
                    nsamp=6000,              # total number of samples
                    nburn=2000)            # number of burn-in samples

df.posterior$type = "step size = 0.02"
posterior = rbind(posterior,df.posterior)
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



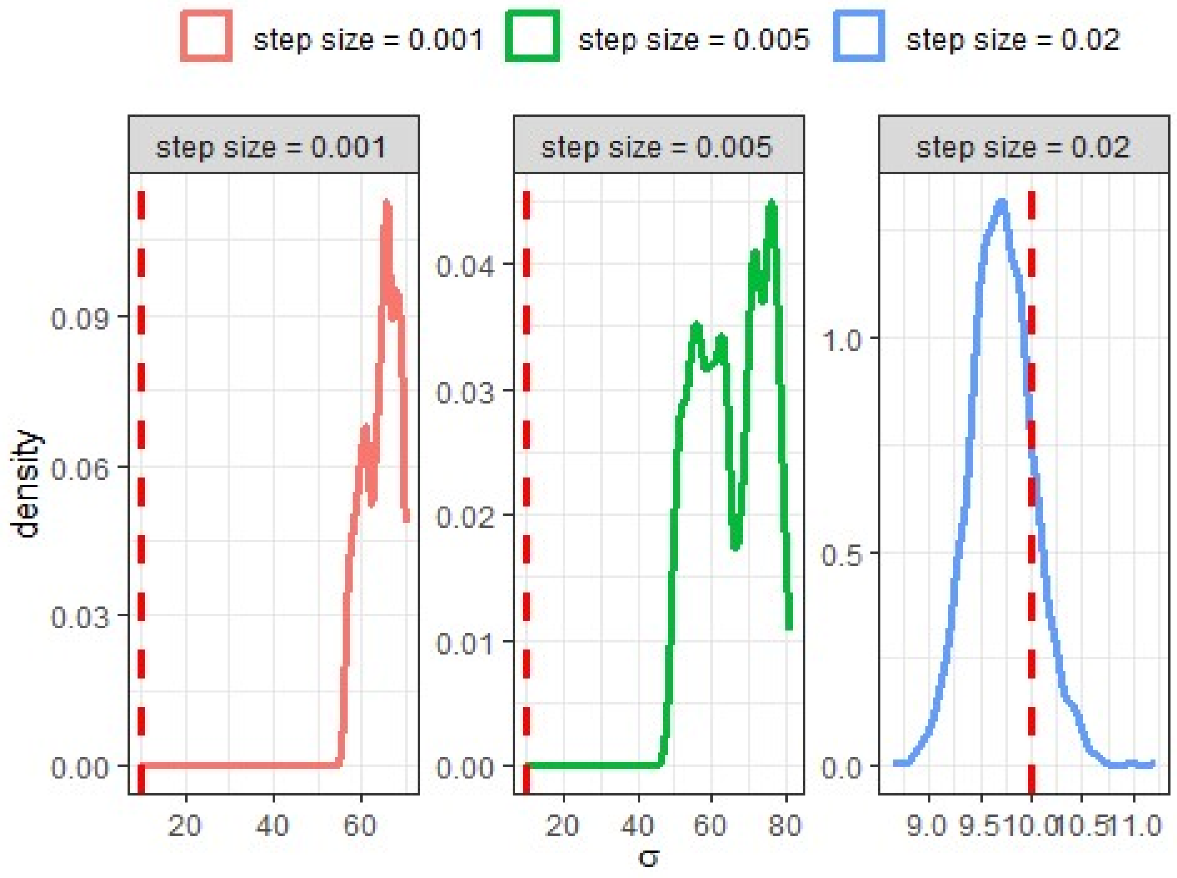
```
#plot of sigma_chain
ggplot(df.posterior,aes(x = sigma_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept=10,size=1.2,colour="red",linetype = "dashed")
```



```
#plotting altogether
ggplot(posterior,aes(x = mu_chain,group = type,colour = type))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")+
  facet_wrap(~type,scales = "free")
```

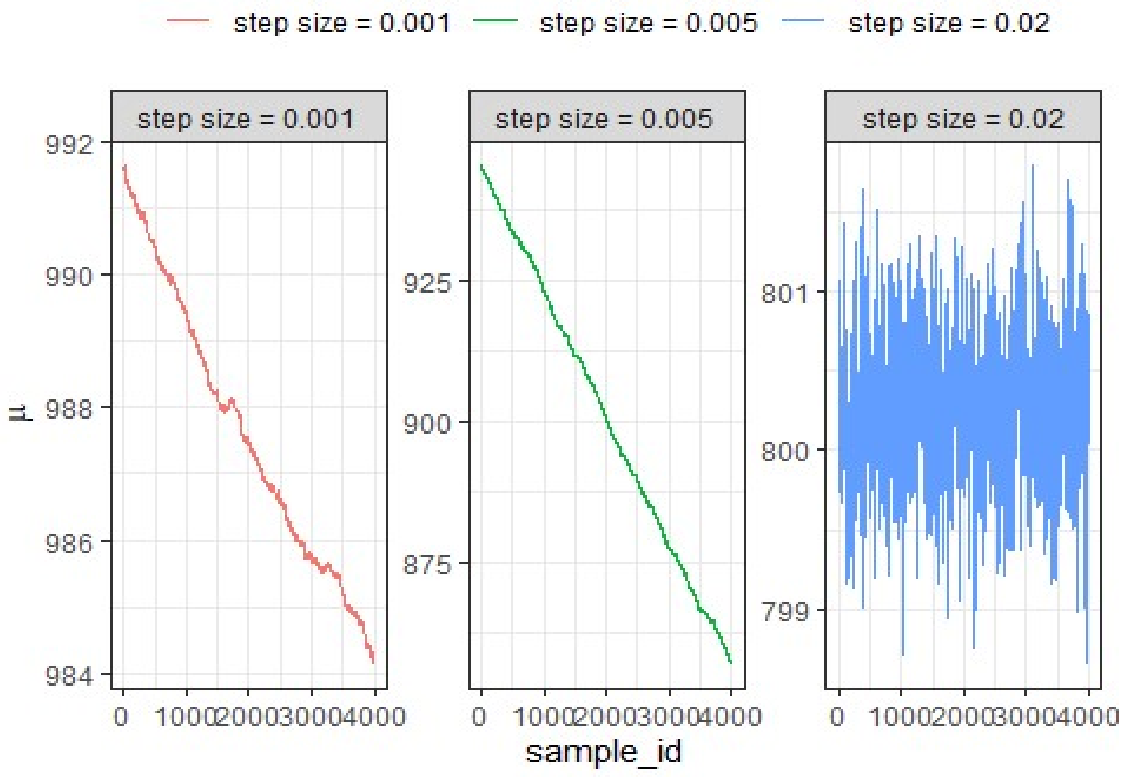


```
#plot of sigma_chain
ggplot(posterior,aes(x = sigma_chain,group = type,colour = type))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept=10,size=1.2,colour="red",linetype = "dashed")+
  facet_wrap(~type,scales = "free")
```

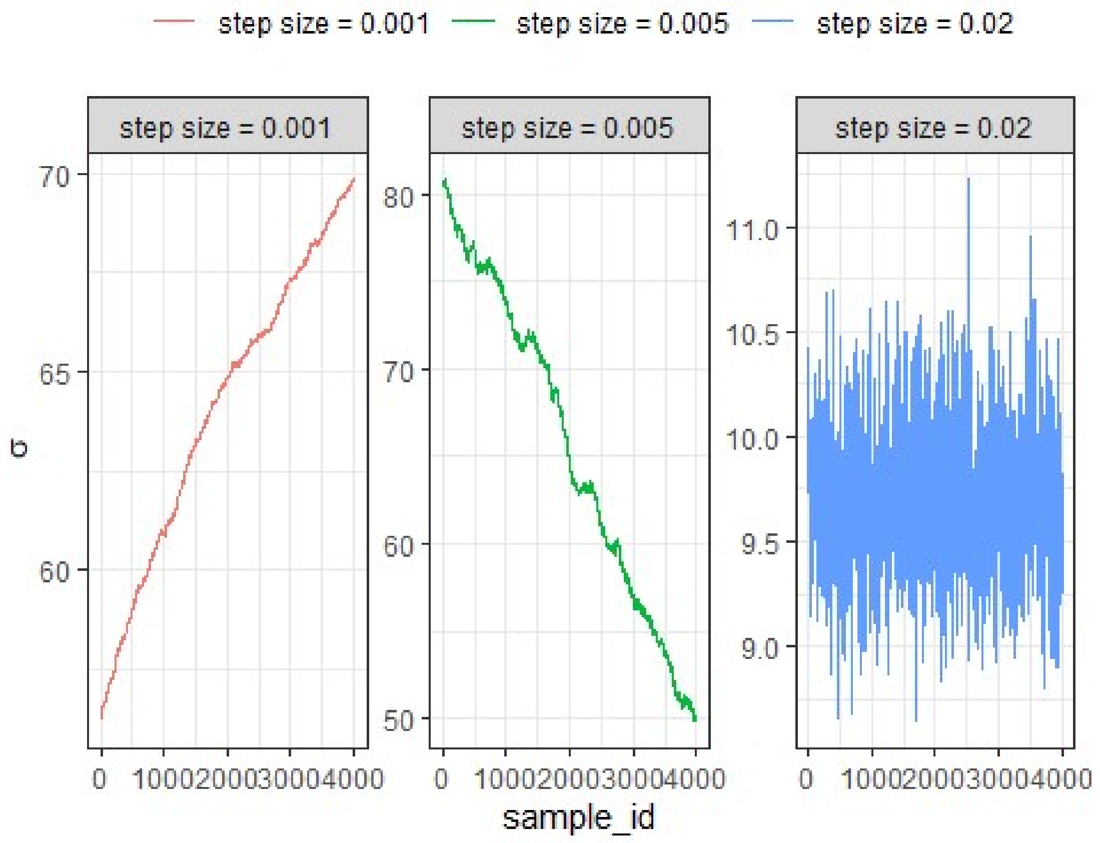


Exercise 3.4: Visually inspect the mu and sigma chains obtained in expercise 3.3. Do youfind anything problematic?

```
#from the last question
#for mu_chain
ggplot(posterior,aes(x = sample_id,y = mu_chain,group = type,colour = type))+
  geom_line()+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  ylab(expression(mu))+
  facet_wrap(~type,scales = "free")
```



```
#for sigma_chain
ggplot(posterior,aes(x = sample_id,y = sigma_chain,group = type,colour =
type))+
  geom_line()+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  ylab(expression(sigma))+
  facet_wrap(~type,scales = "free")
```



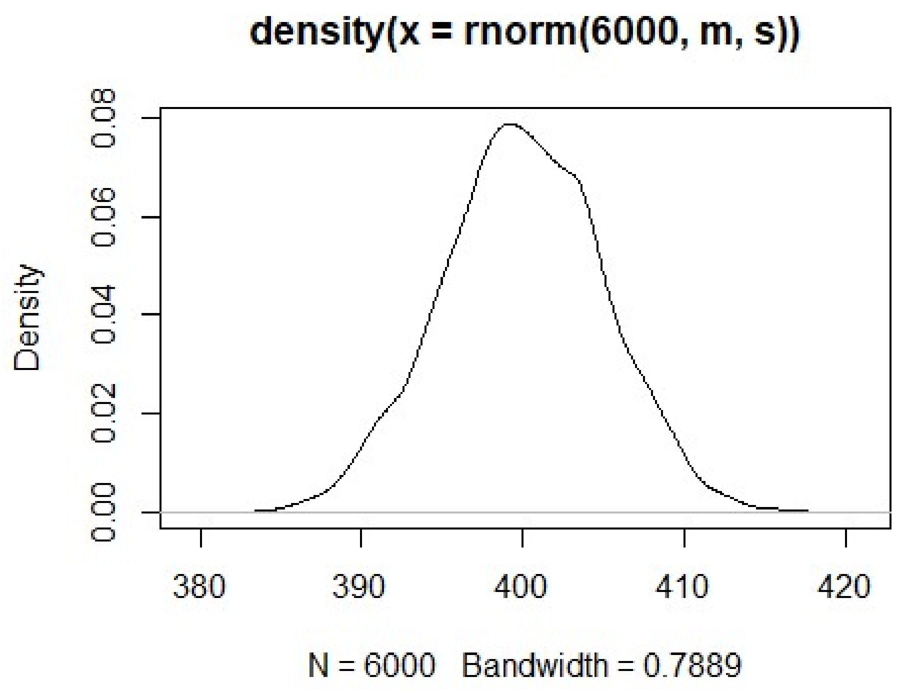
Problem with small step size--

Smaller value of step size results in shorter trajectories as compared to larger value of step size. Due to this short trajectories may not explore the entire posterior distribution for both the parameters.

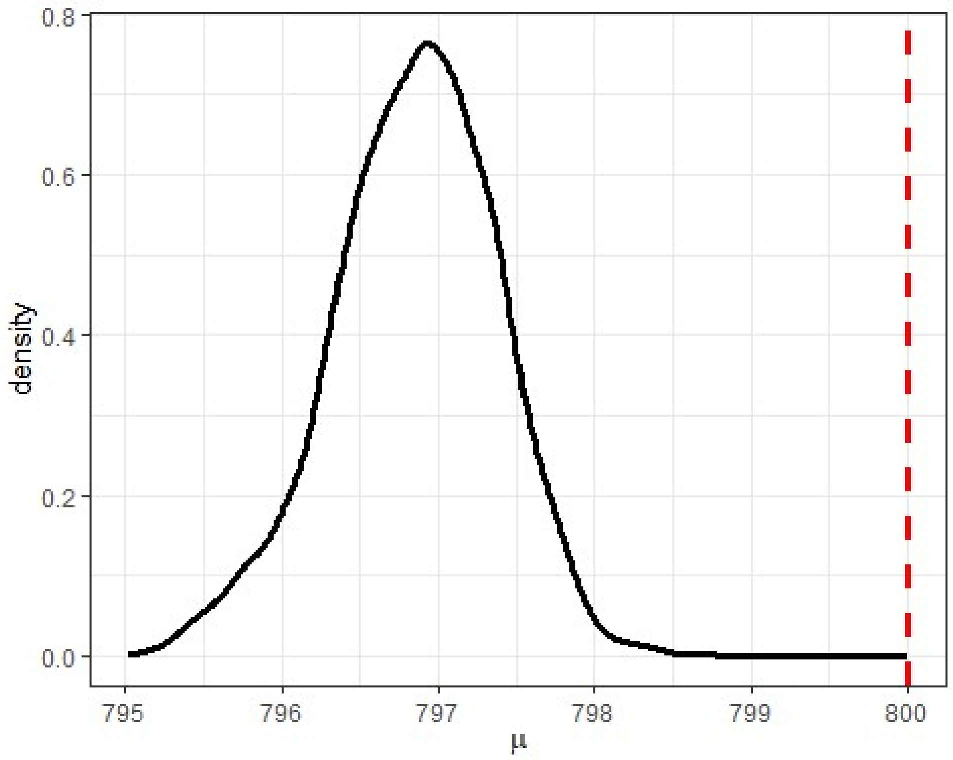
Also, having small step size results in slow convergence of chains requiring more iterations to reach the target distribution, and also small step size makes the sample highly correlated, leading to inefficient exploration

Exercise 3.5: Check the prior sensitivity for the μ parameter. Estimate and compare the posterior distribution of μ when the prior on μ are-

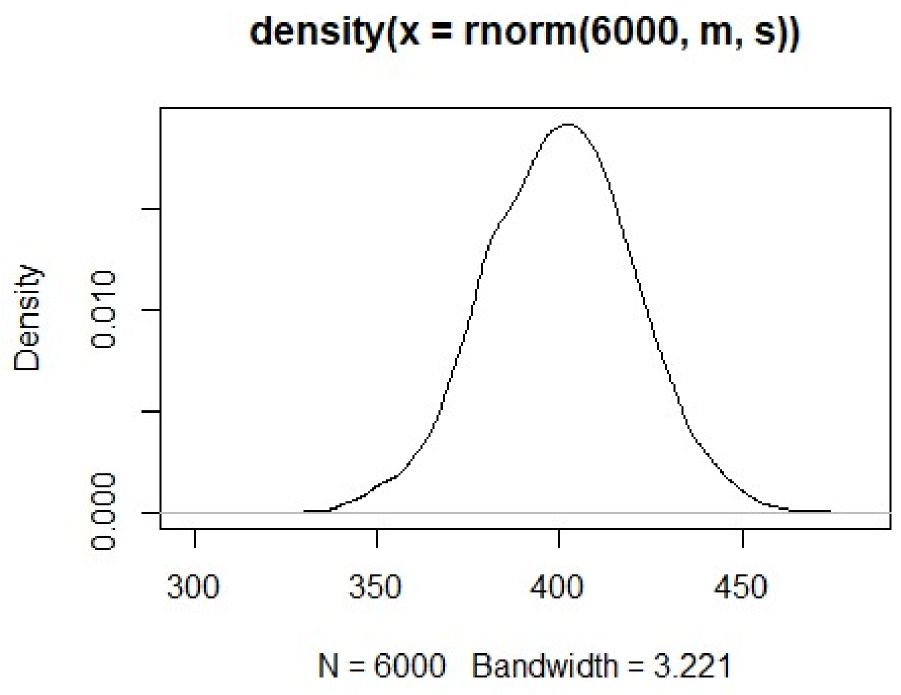
```
#μ~Normal(m =400,s = 5)
m = 400
s = 5
plot(density(rnorm(6000,m,s)))
```



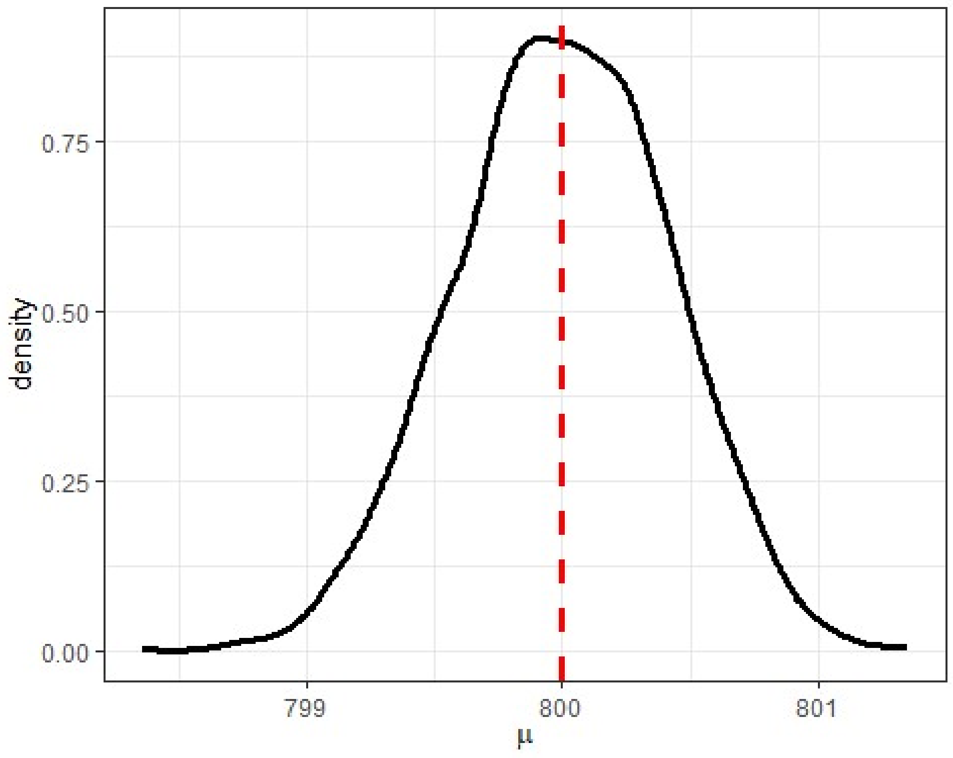
```r
df.posterior = HMC(y=y,n=length(y),                    # data
                    m=m,s=s,a=10,b=2,                   # priors
                    step=step_size,                     # step-size
                    L=12,                               # no. of leapfrog steps
                    initial_q=c(1000,11),               # Chain initialization
                    nsamp=6000,                         # total number of samples
                    nburn=2000)                         # number of burn-in samples
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



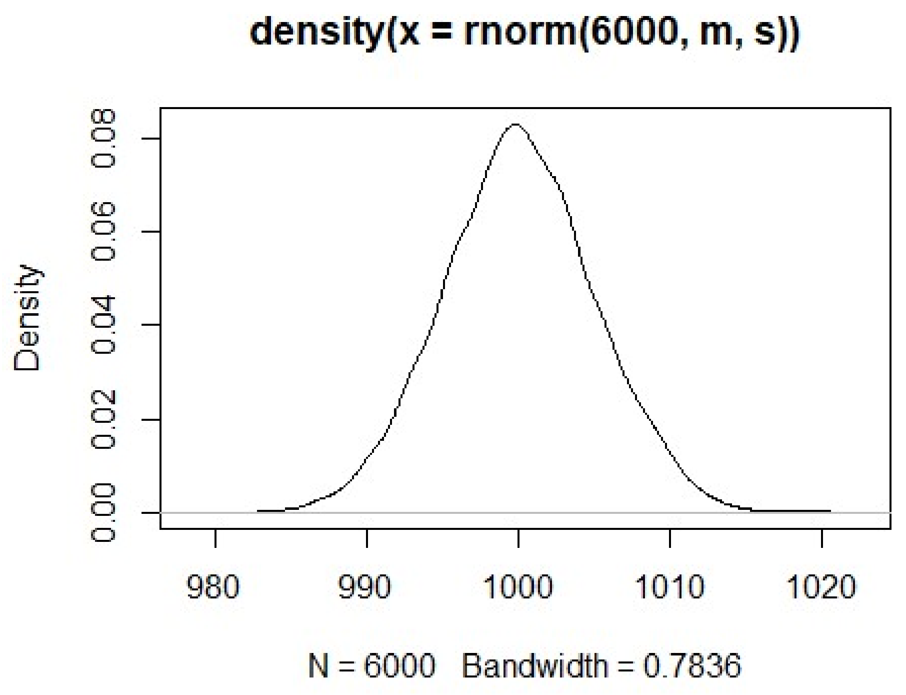
```
df.posterior$type = "m = 400,s = 5"
posterior = df.posterior

#μ~Normal(m =400,s = 20)
m = 400
s = 20
plot(density(rnorm(6000,m,s)))
```



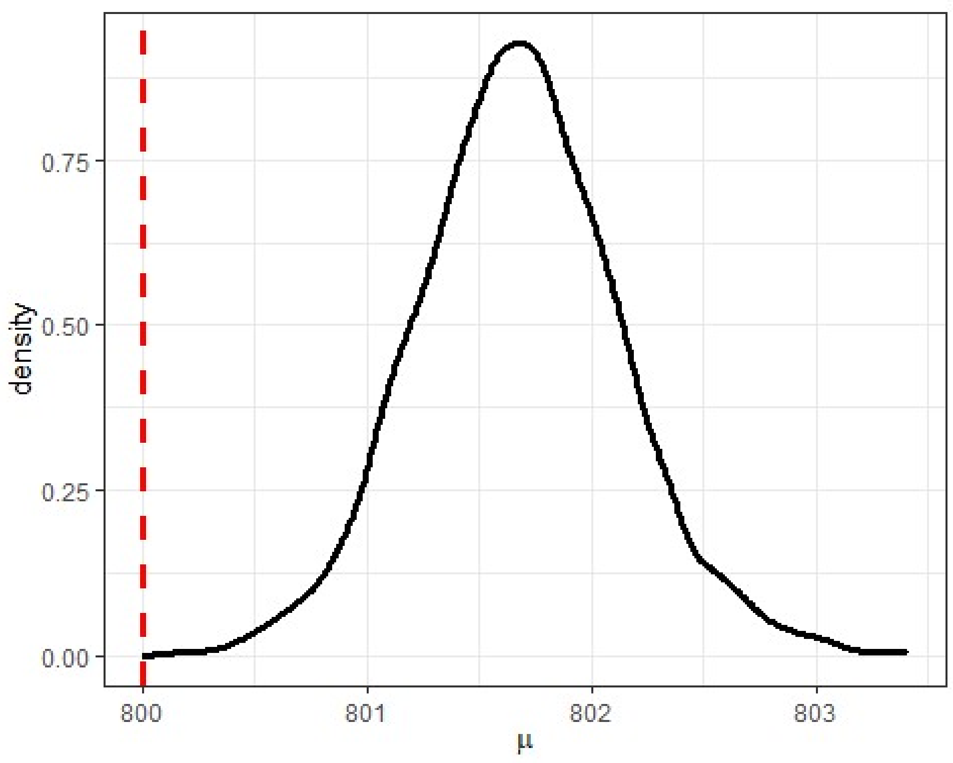
```
df.posterior = HMC(y=y,n=length(y),                  # data
                   m=m,s=s,a=10,b=2,                 # priors
                   step=step_size,                   # step-size
                   L=12,                             # no. of leapfrog steps
                   initial_q=c(1000,11),             # Chain initialization
                   nsamp=6000,                       # total number of samples
                   nburn=2000)                       # number of burn-in samples
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



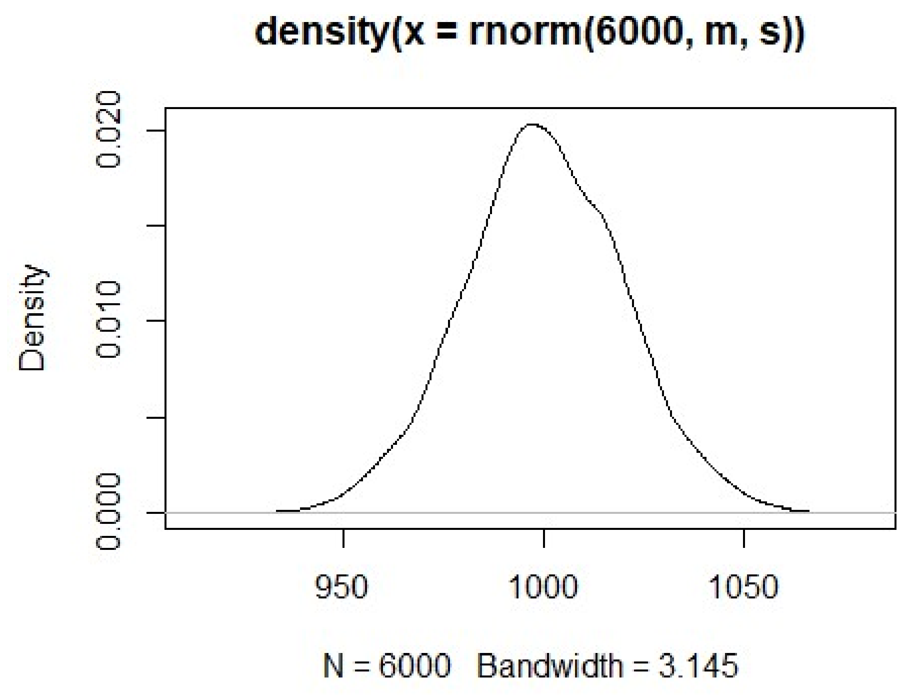
```
df.posterior$type = "m = 400,s = 20"
posterior = rbind(posterior,df.posterior)

#μ~Normal(m =1000,s = 5)
m = 1000
s = 5
plot(density(rnorm(6000,m,s)))
```



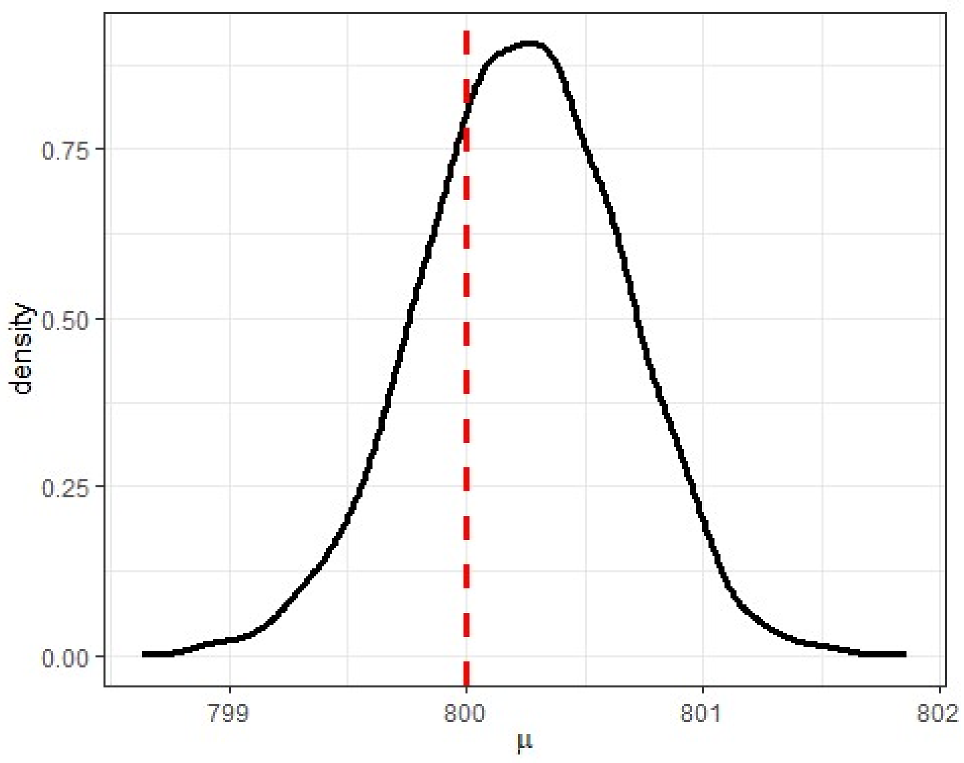
```
df.posterior = HMC(y=y,n=length(y),                    # data
                   m=m,s=s,a=10,b=2,                   # priors
                   step=step_size,                     # step-size
                   L=12,                               # no. of leapfrog steps
                   initial_q=c(1000,11),               # Chain initialization
                   nsamp=6000,                         # total number of samples
                   nburn=2000)                         # number of burn-in samples
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



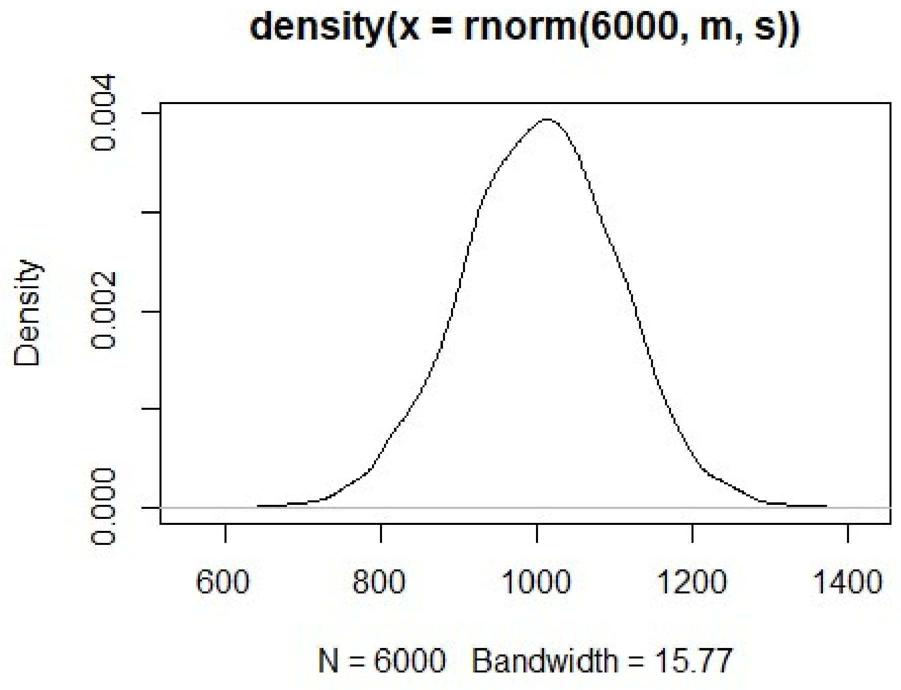
```
df.posterior$type = "m = 1000,s = 5"
posterior = rbind(posterior,df.posterior)

#μ~Normal(m =1000,s = 20)
m = 1000
s = 20
plot(density(rnorm(6000,m,s)))
```



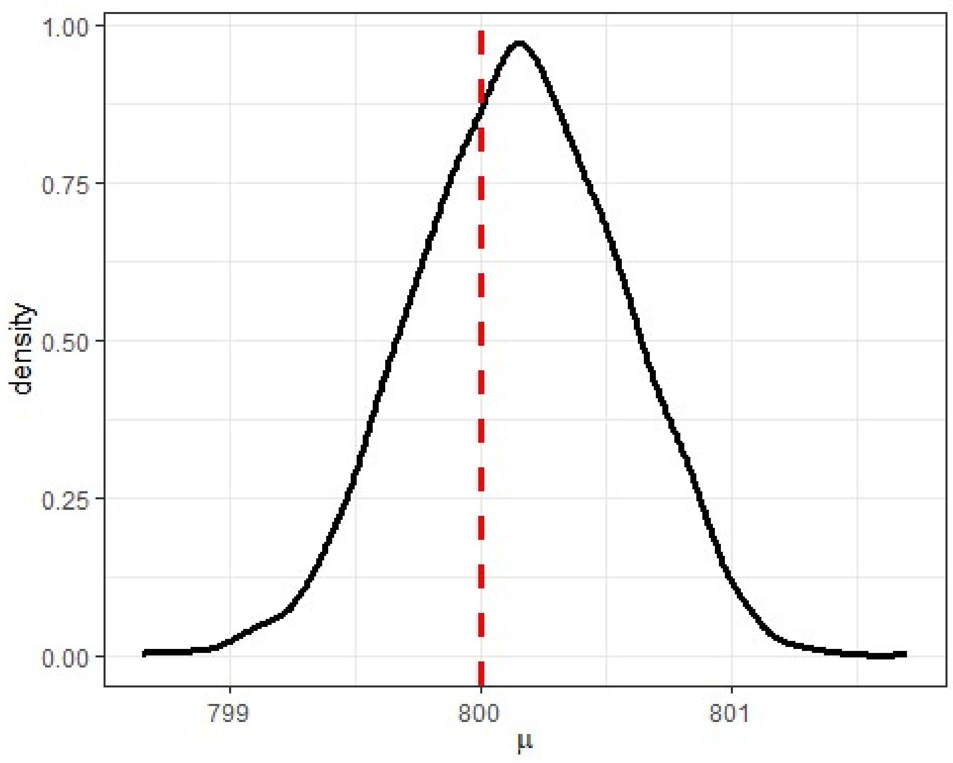
```
df.posterior = HMC(y=y,n=length(y),                # data
                    m=m,s=s,a=10,b=2,              # priors
                    step=step_size,                # step-size
                    L=12,                          # no. of leapfrog steps
                    initial_q=c(1000,11),          # Chain initialization
                    nsamp=6000,                    # total number of samples
                    nburn=2000)                    # number of burn-in samples
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



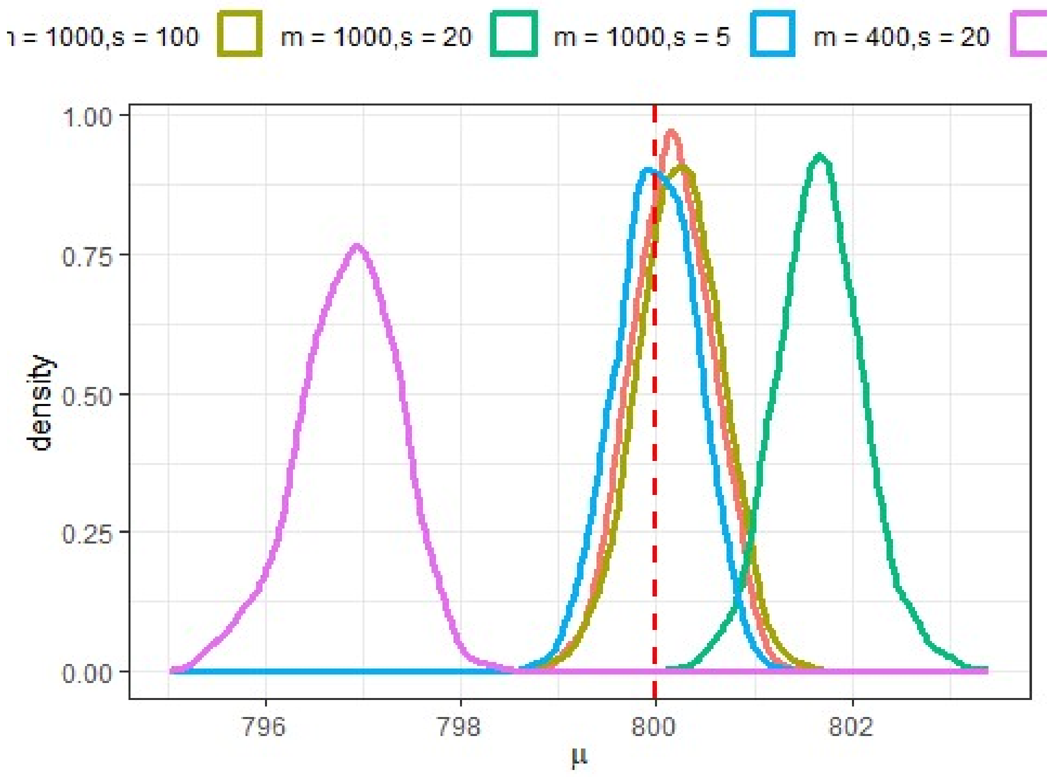
```
df.posterior$type = "m = 1000,s = 20"
posterior = rbind(posterior,df.posterior)

#μ~Normal(m =1000,s = 100)
m = 1000
s = 100
plot(density(rnorm(6000,m,s)))
```



```r
df.posterior = HMC(y=y,n=length(y),                 # data
                   m=m,s=s,a=10,b=2,                # priors
                   step=step_size,                  # step-size
                   L=12,                            # no. of leapfrog steps
                   initial_q=c(1000,11),            # Chain initialization
                   nsamp=6000,                      # total number of samples
                   nburn=2000)                      # number of burn-in samples
#plot of mu_chain
ggplot(df.posterior,aes(x = mu_chain))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1.2,colour="red",linetype = "dashed")
```



```
df.posterior$type = "m = 1000,s = 100"
posterior = rbind(posterior,df.posterior)


#plotting altogether
ggplot(posterior,aes(x = mu_chain,group = type,colour = type))+
  geom_density(size = 1.2)+
  theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept=800,size=1,colour="red",linetype = "dashed")
```


ı = 1000,s = 100
m = 1000,s = 20
m = 1000,s = 5
m = 400,s = 20
1.00
0.75
0.50
0.25
0.00
density
796
798
800
802
μ