Assignment – 3

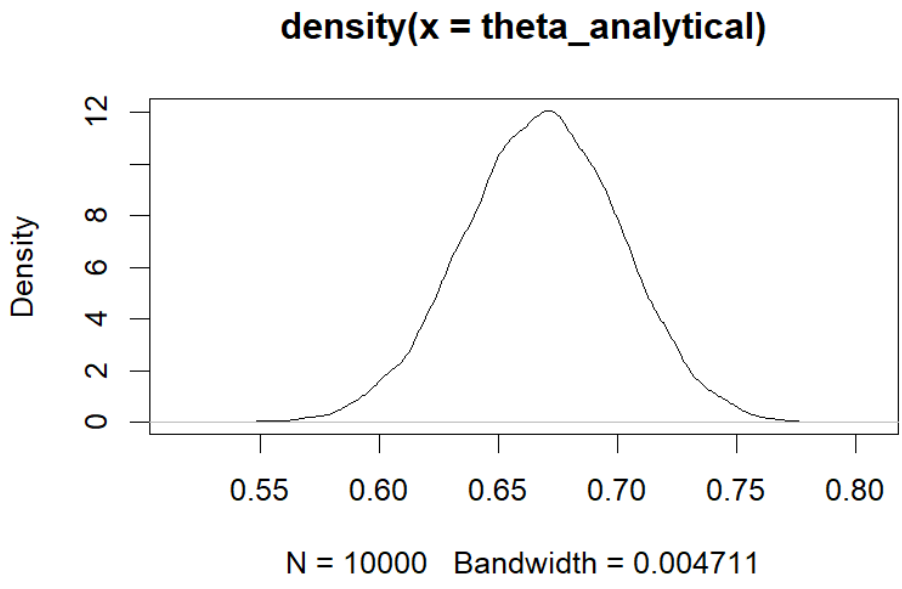
Name: Pritam Priyadarshi

Roll No.: 230793

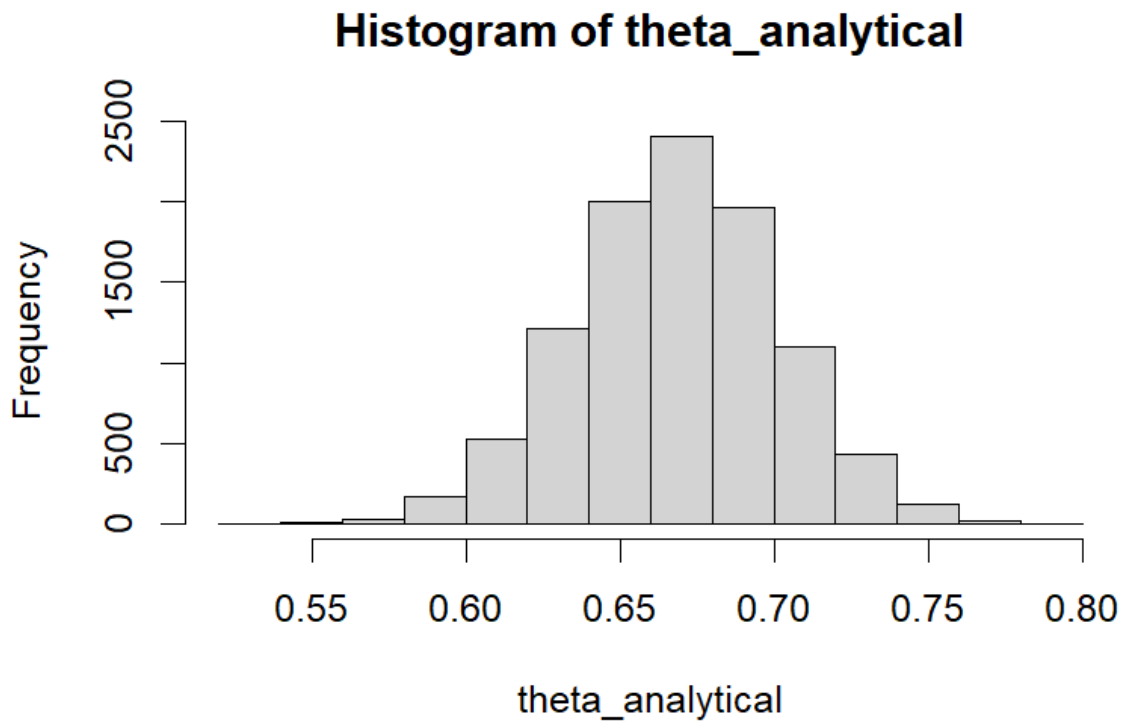
Part: 1 Estimating the posterior distribution using different computational methods

(1). Analytically

```
# ASSIGNMENT 3


################################

##### Part 1



### 1 (Analytically)

# Let yi be the ith datapoint
# yi ~ Binomial(n = 20,θ)
# θ ~ Beta(1,1)
# The analytically-derived posterior distribution of θ is:
# θ|y ~ Beta(135,67)

# Data and prior
n <- 20
y <- c(10, 15, 15, 14, 14, 14, 13, 11, 12, 16)

a <- 135
b <- 67

theta_analytical <- rbeta(10000, 135, 67)
plot(density(theta_analytical))

```

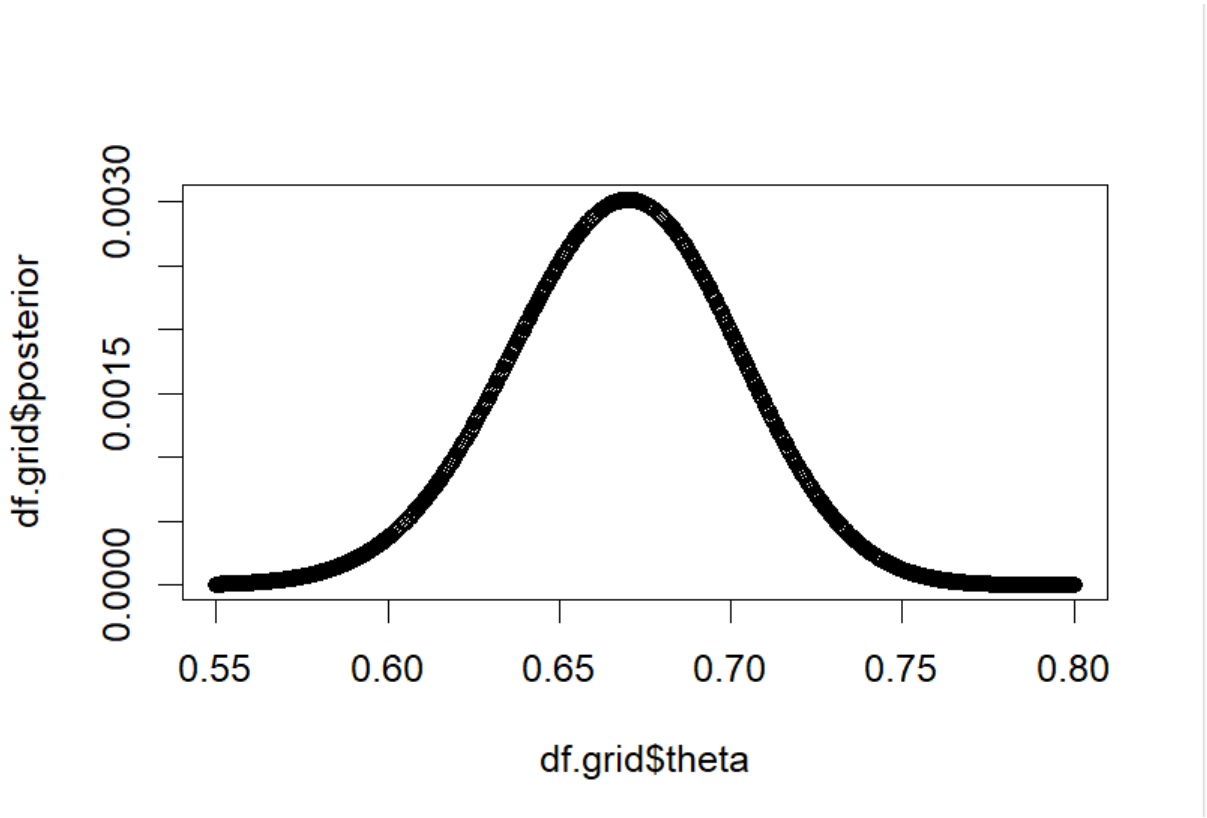


```r
hist(theta_analytical)
```



(2). Grid approximation

```r
### 2 (Grid Approximation)

# grid of theta values
theta_grid <- seq(0.55, 0.80, length=1000)

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$lkl[i] <- prod(dbinom(y, size=n, prob=theta_grid[i]))
  df.grid$prior[i] <- dbeta(theta_grid[i],1, 1)
}

df.grid$ML <- rep(sum(df.grid$lkl*df.grid$prior), 1000)

# Normalized posterior
df.grid$posterior <- (df.grid$lkl*df.grid$prior/df.grid$ML)

# Plot posterior density
plot(df.grid$theta, df.grid$posterior)

```

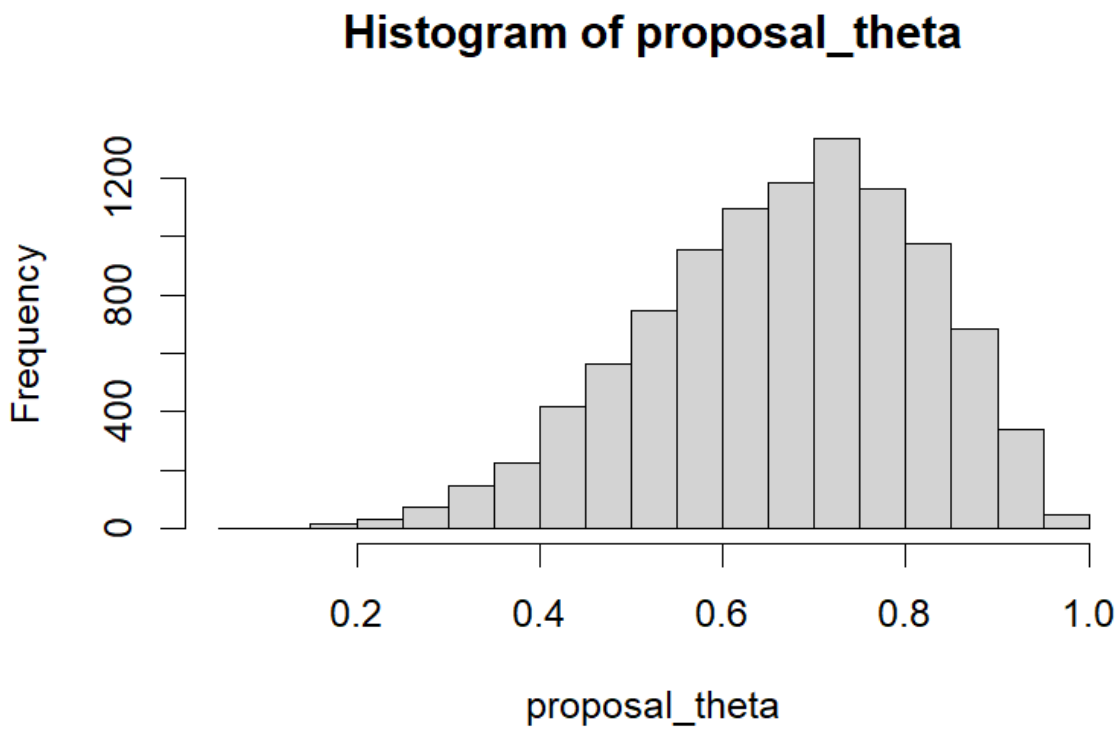


(3). Monte carlo integration

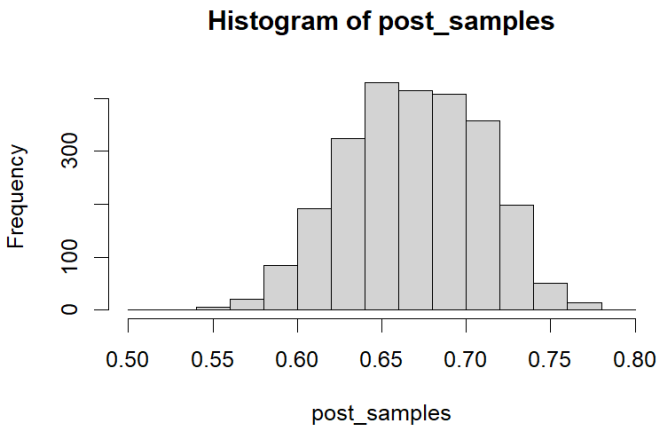
```r
### 3 (Monte Carlo Integration)

df.monte_carlo <- data.frame(matrix(ncol=2, nrow=10000))
colnames(df.monte_carlo) <- c("theta_sample", "lkl")

for (i in 1:10000) {
  theta_i <- rbeta(1,1,1) # independent sample from the prior
  lkl <- prod(dbinom(y, size=n, prob=theta_i))
  df.monte_carlo[i,] <- c(theta_i, lkl)
}

# Marginal likelihood
ML_mc <- mean(df.monte_carlo$lkl)
ML_mc

## [1] 1.344892e-10

```

(4). Importance sampling

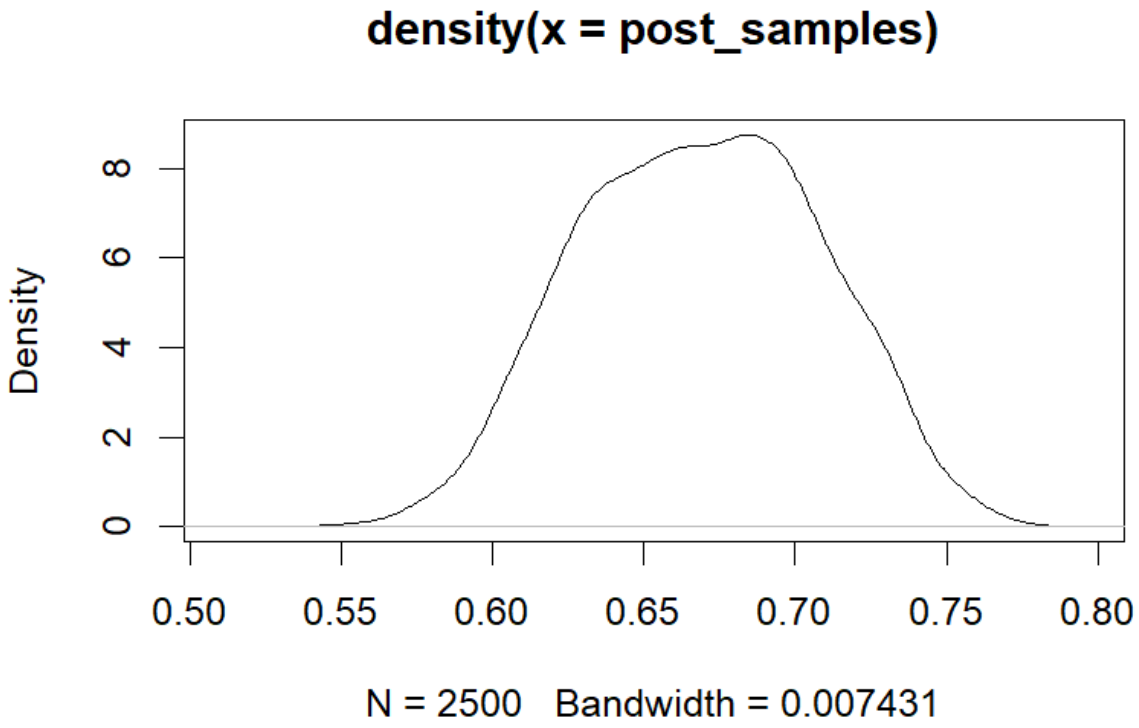
```r
### 4 (Importance Sampling)

nsamp <- 10000

# proposal density
proposal_theta <- rbeta(nsamp, 6, 3)
hist(proposal_theta)
```




```r
weights <- rep(NA, nsamp)

for (i in 1:nsamp) {
  lkl_i <- prod(dbinom(y, size=20, prob=proposal_theta[i]))
  prior_i <- dbeta(proposal_theta[i], 1, 1)
  proposal_density_i <- dbeta(proposal_theta[i], 6, 3)
  weights[i] <- lkl_i*prior_i/proposal_density_i
}

df.imp_samp <- data.frame(theta=proposal_theta,
                          weights=weights)
post_samples <- sample(df.imp_samp$theta, size=nsamp/4,
                       prob=df.imp_samp$weights)
hist(post_samples)

```



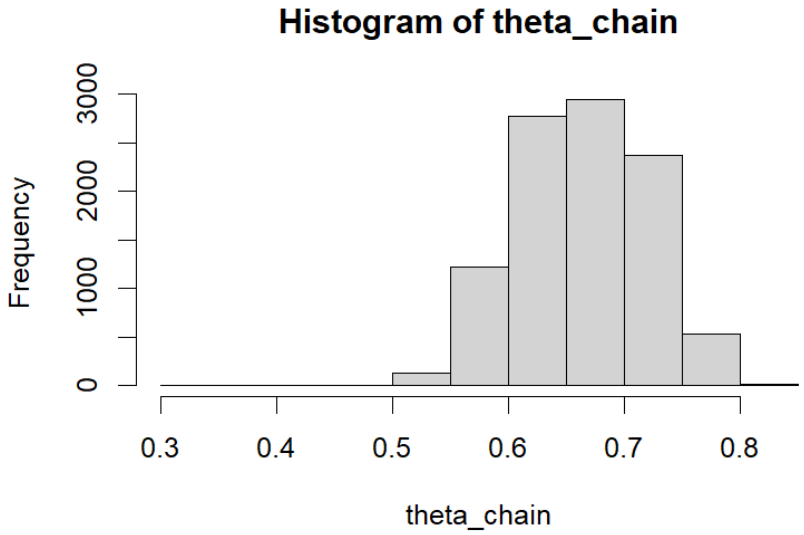
```
1. plot(density(post_samples))
```



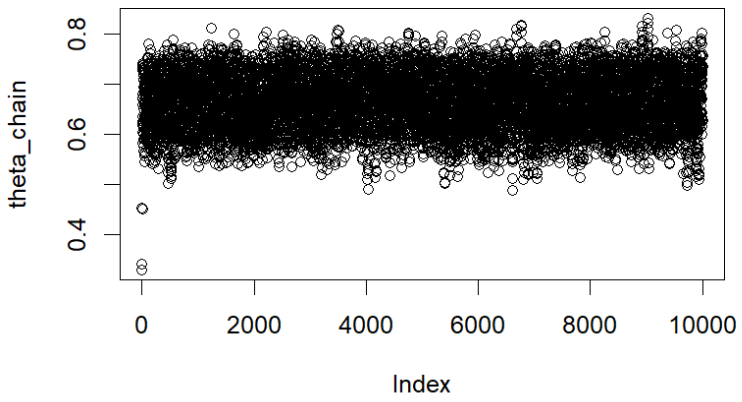
(5). MCMC method to estimate the posterior distribution of θ

```
 1. ### 5 (Markov Chain Monte Carlo)
 2.
 3. y
 4. ## [1] 10 15 15 14 14 14 13 11 12 16
 5.
 6. # Markov chain
 7. nsamp <- 10000
 8. theta_chain <- rep(NA, nsamp)
 9.
10. # Initialization of Markov chain
11. theta_chain[1] <- rbeta(1, 1, 1)
12.
13. # Evolution of Markov chain
14. i <- 1
15. step <- 0.09
16. reject <- 0
17. ML <- 0
18.
19. while (i<nsamp) {
20.   proposal_theta <- rnorm(1, theta_chain[i], step)
21.
22.   if(proposal_theta>0 & proposal_theta<1) {
23.     post_new <- prod(dbinom(y, 20, proposal_theta))*
24.       dbeta(proposal_theta,1,1)
25.     post_prev <- prod(dbinom(y,20,theta_chain[i]))*
26.       dbeta(theta_chain[i], 1, 1)
27.     # computing Hasting ratio
```

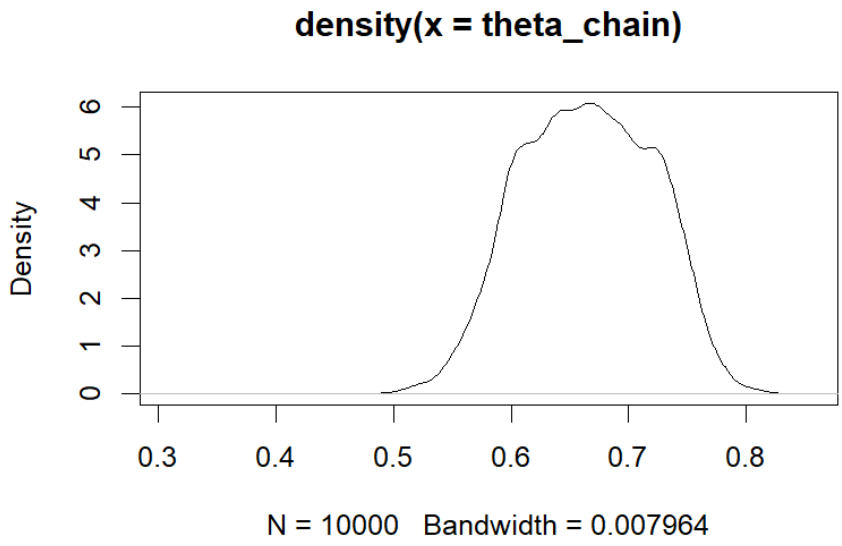
```r
    Hasting_ratio <- post_new*dnorm(theta_chain[i],proposal_theta,step)/
      post_prev*dnorm(proposal_theta,theta_chain[i],step)
    # acceptance criteria
    accep_p <- min(Hasting_ratio,1)
    if(accep_p>runif(1,0,1)){
      theta_chain[i+1] <- proposal_theta
      i <- i+1
    }else{
      reject <- reject+1
    }
  }
}

rejection_rate <- reject*100/(reject+nsamp)
rejection_rate

## [1] 36.79687

hist(theta_chain)

```



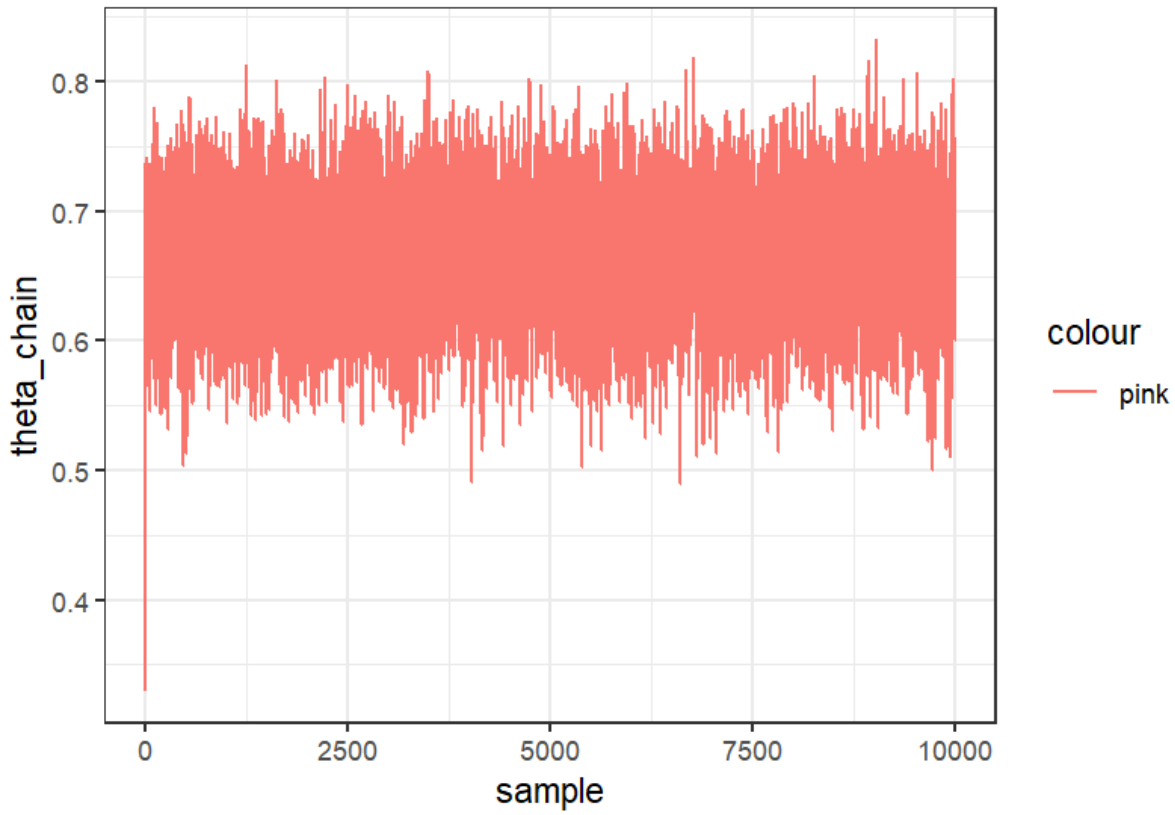

```r
plot(theta_chain)
```



```
1. plot(density(theta_chain))
```



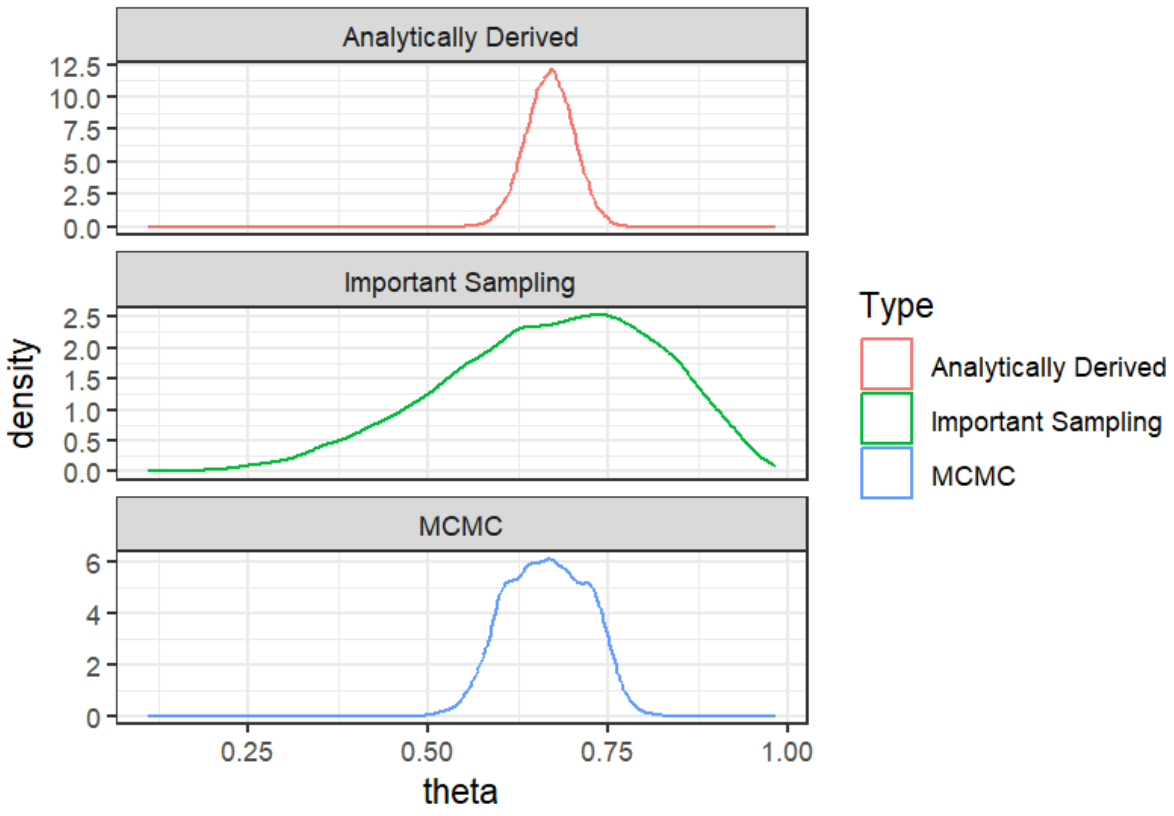

```
1. df.MCMC <- data.frame(theta_chain=theta_chain, sample=1:nsamp)
2. library(ggplot2)
3. ggplot(df.MCMC, aes(y=theta_chain, x=sample, color="pink"))+
4.   geom_line()+theme_bw()
5.
```



(6). Graphically compare the posterior distributions of θ obtained using:

- Importance sampling
- Markov chain Monte Carlo
- Analytical derivation

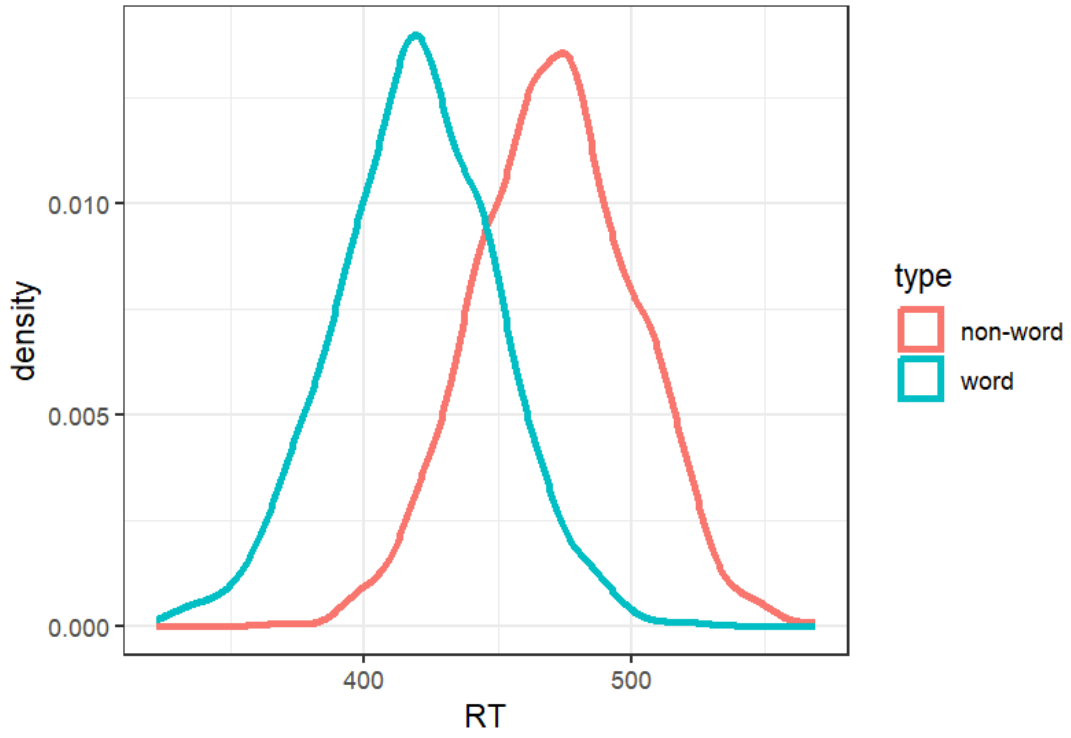
```
### 6

df.posterior <- data.frame(matrix(nrow=10000,ncol=3))
colnames(df.posterior)=c("sample", "theta", "Type")
df.posterior$sample <- 1:10000
df.posterior$theta <- df.MCMC$theta_chain
df.posterior$Type <- "MCMC"

df.posterior <- rbind(df.posterior,
                      data.frame(sample=1:10000,
                                 theta=df.imp_samp$theta,
                                 Type="Important Sampling"),
                      data.frame(sample=1:10000,
                                 theta=theta_analytical,
                                 Type="Analytically Derived"))

ggplot(df.posterior, aes(x=theta, group = Type, colour = Type))+
  geom_density()+facet_wrap(~Type, scales="free_y", nrow=3)+
  theme_bw()

```



Part 2: Writing your own sampler for Bayesian inference

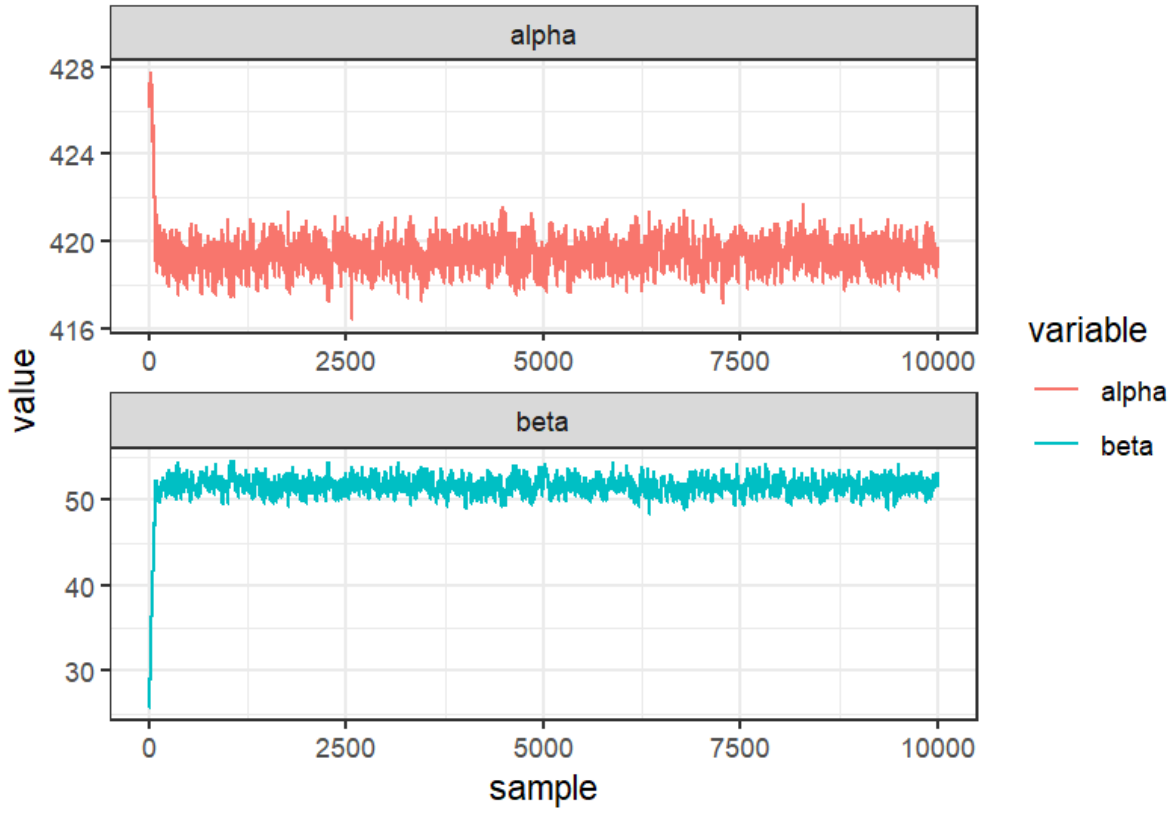
(2.1). Estimate the parameters α and β using Markov Chain Monte Carlo

```
################################

##### Part 2



# 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

# Priors:
# α ~ Normal(400,50)
# σ =30
# β ~ Normal+(0,50)

library(truncnorm)
library(reshape2)

dat <- read.table(
  "https://raw.githubusercontent.com/yadavhimanshu059/CGS698C/main/notes/Data/word-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()
```




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


# Markov chain
nsamp <- 10000
alpha_chain <- rep(NA, nsamp)
beta_chain <- rep(NA, nsamp)

# Initialization of Markov chain
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
step <- 0.5
reject <- 0
while (i<nsamp) {
  proposed_alpha <- rnorm(1, alpha_chain[i], step)
  proposed_beta <- rtruncnorm(1, a=0, b=Inf, mean=beta_chain[i], sd=step)

  # 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(proposed_beta, mean=0, sd=50,
                                      a=0, b=Inf))
  log_current_beta <- log(dtruncnorm(beta_chain[i], mean=0, sd=50,
                                      a=0, b=Inf))
  # proposal density
```

```r
  log_fwd_density_alpha <- dnorm(proposed_alpha, alpha_chain[i],
                                 step, log = TRUE)
  log_bcwd_density_alpha <- dnorm(alpha_chain[i], proposed_alpha,
                                  step, log = TRUE)
  log_fwd_density_beta <- dtruncnorm(proposed_beta, a=0, b=Inf,
                                     mean=beta_chain[i], sd=step)
  log_bcwd_density_beta <- dtruncnorm(beta_chain[i], a=0, b=Inf,
                                      mean=proposed_beta, sd=step)
  # Hasting ratio
  Hasting_ratio <- exp((log_proposal_lkl+log_proposal_alpha+
                          log_proposal_beta-log_fwd_density_alpha-
                          log_fwd_density_beta)-
                         (log_current_lkl+log_current_alpha+
                          log_current_beta-log_bcwd_density_alpha-
                          log_bcwd_density_beta))
  # acceptance criteria
  accep_p <- min(Hasting_ratio, 1)
  if(accep_p>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
rej_rate <- reject*100/(reject+nsamp)
rej_rate

## [1] 39.08753

# plotting parameters
df.parameters <- data.frame(sample =1:nsamp, alpha=alpha_chain,
                            beta=beta_chain)

ggplot(melt(df.parameters, id="sample"), aes(x=sample, y=value,
                                             group=variable, colour = variable))+
  geom_line()+theme_bw()+
  facet_wrap(~variable, scales="free", ncol=1)

```

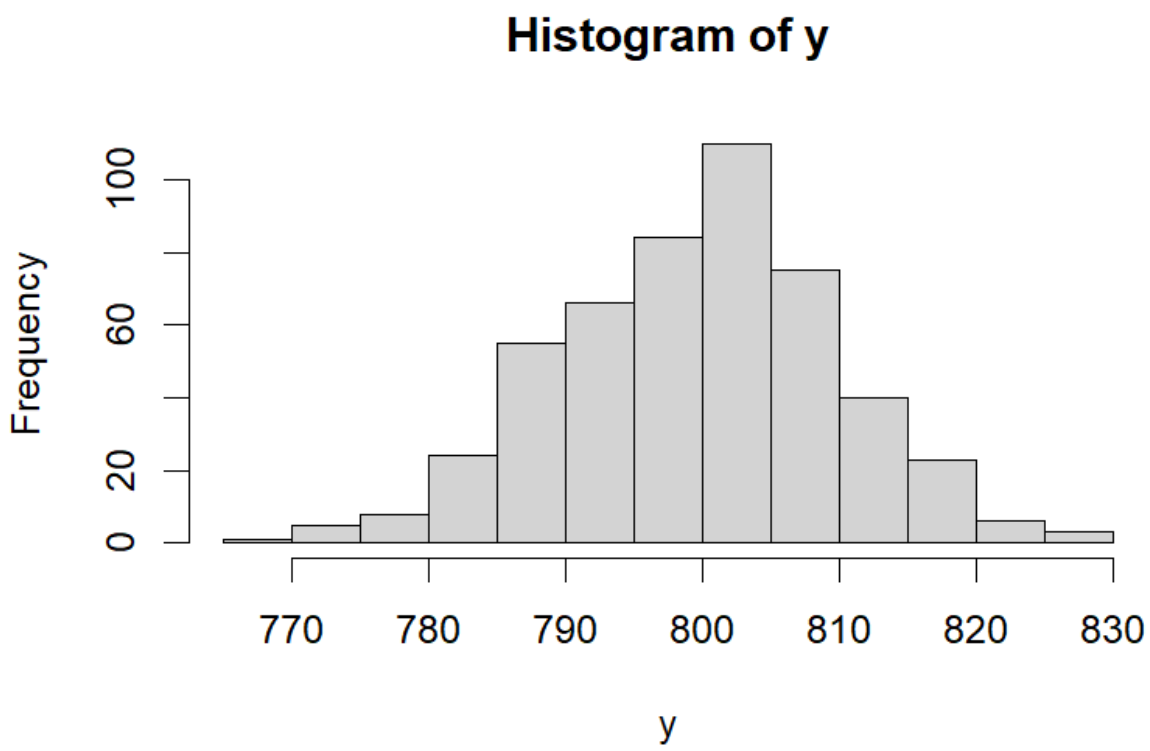


(2.2). Credible interval for each parameter

```
### Credible interval

# for alpha
quantile(df.parameters$alpha, probs = c(0.025,0.975))

##     2.5%    97.5%
## 418.1195 420.7264

# for beta
quantile(df.parameters$beta, probs = c(0.025,0.975))

##     2.5%    97.5%
## 49.83511 53.50223

# mean of each parameters
mean(alpha_chain)

## [1] 419.4338

mean(beta_chain)

## [1] 51.62699

```

Part 3: Hamiltonian Monte Carlo sampler

```
 1. ################################
 2.
 3. ##### Part 3 (Hamiltonian Monte Carlo sampler)
 4.
 5.
 6.
 7. true_mu <- 800
 8. true_var <- 100 #sigma^2
 9. y <- rnorm(500,mean=true_mu,sd=sqrt(true_var))
10. hist(y)
11.
```

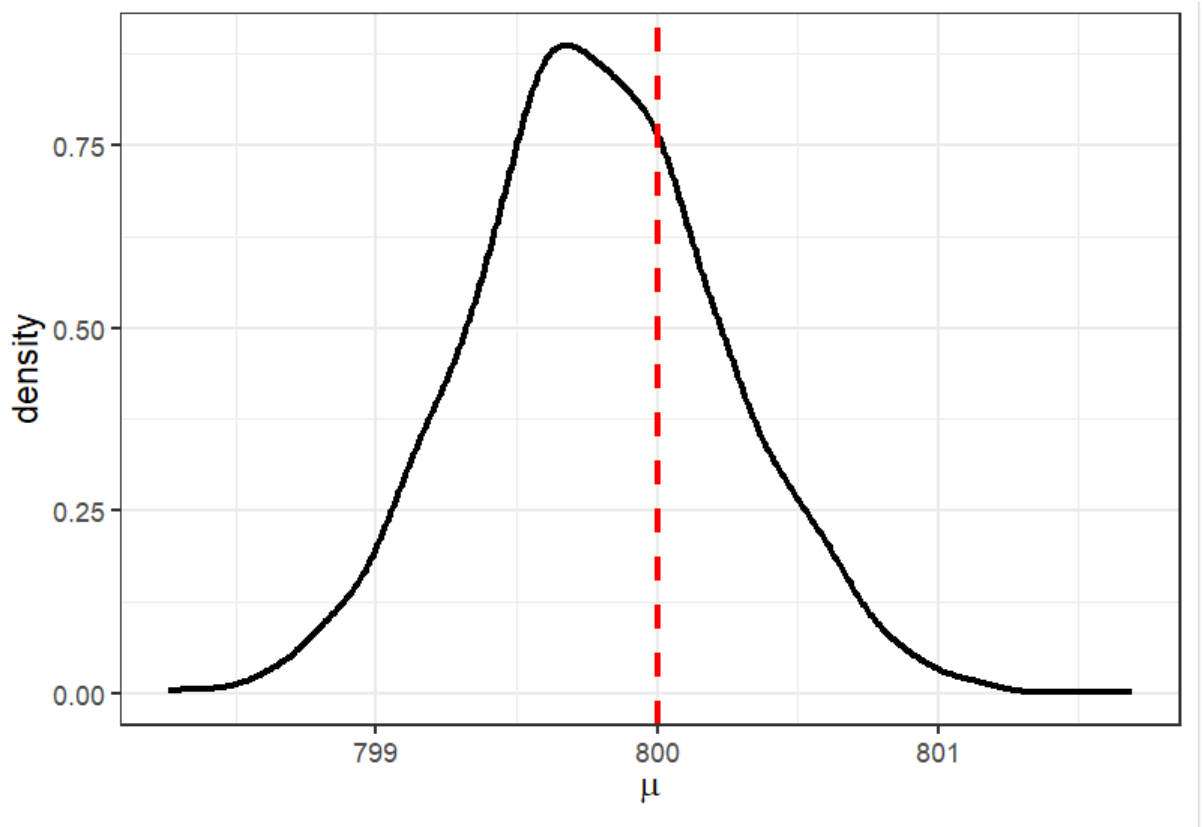



```
 1. #Gradient functions
 2. gradient <- function(mu,sigma,y,n,m,s,a,b){
 3.   grad_mu <- (((n*mu)-sum(y))/(sigma^2))+((mu-m)/(s^2))
 4.   grad_sigma <- (n/sigma)-(sum((y-mu)^2)/(sigma^3))+((sigma-a)/(b^2))
 5.   return(c(grad_mu,grad_sigma))
 6. }
 7. #Potential energy function
 8. V <- function(mu,sigma,y,n,m,s,a,b){
 9.   nlpd <--(sum(dnorm(y,mu,sigma,log=T))+dnorm(mu,m,s,log=T)+dnorm(sigma,a,b,log=T))
10.   nlpd
11. }
12. #HMC sampler
13. HMC <- function(y,n,m,s,a,b,step,L,initial_q,nsamp,nburn){
14.   mu_chain <- rep(NA,nsamp)
15.   sigma_chain <- rep(NA,nsamp)
16.   reject <- 0
17.   #Initialization of Markov chain
18.   mu_chain[1] <- initial_q[1]
19.   sigma_chain[1] <- initial_q[2]
```

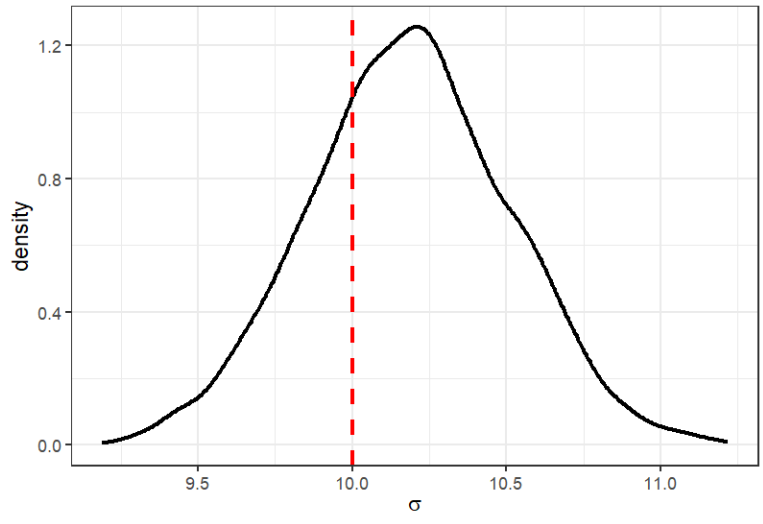
```r
  #Evolution of Markov chain
  i <- 1
  while(i < nsamp){
    q <- c(mu_chain[i],sigma_chain[i]) # Current position of the particle
    p <- rnorm(length(q),0,1)
    current_q <- q
    current_p <- p
    # Generate random momentum at the current position
    current_V = V(current_q[1],current_q[2],y,n,m,s,a,b) # 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)
  posteriors
}

```

(3.1).

```r
## 3.1
# Estimate and plot the posteriors for mu and sigma
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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

```



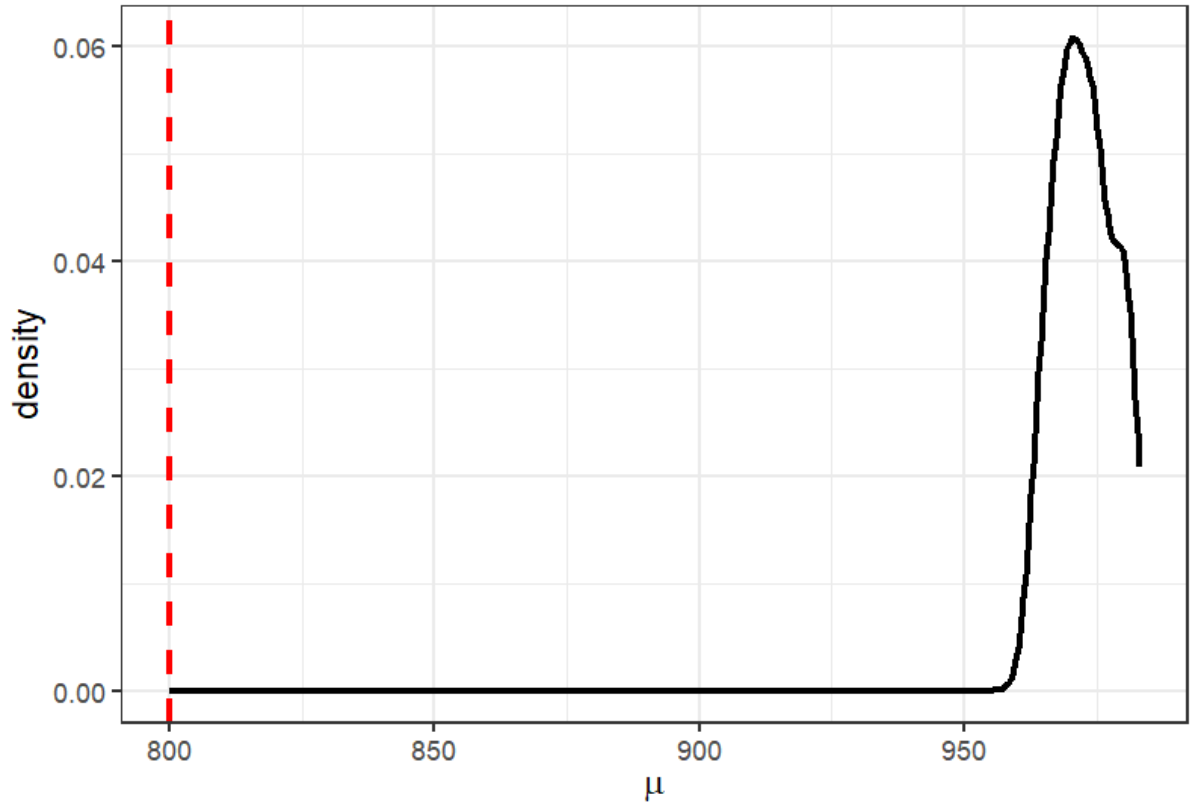

```
1. # plot of sigma_chain
2. ggplot(df.posterior, aes(x=sigma_chain))+
3.   geom_density(size=1)+theme_bw()+
4.   theme(legend.title = element_blank(),legend.position = "top")+
5.   xlab(expression(sigma))+
6.   geom_vline(xintercept = 10,size=1,colour="red",linetype="dashed")
7.
```



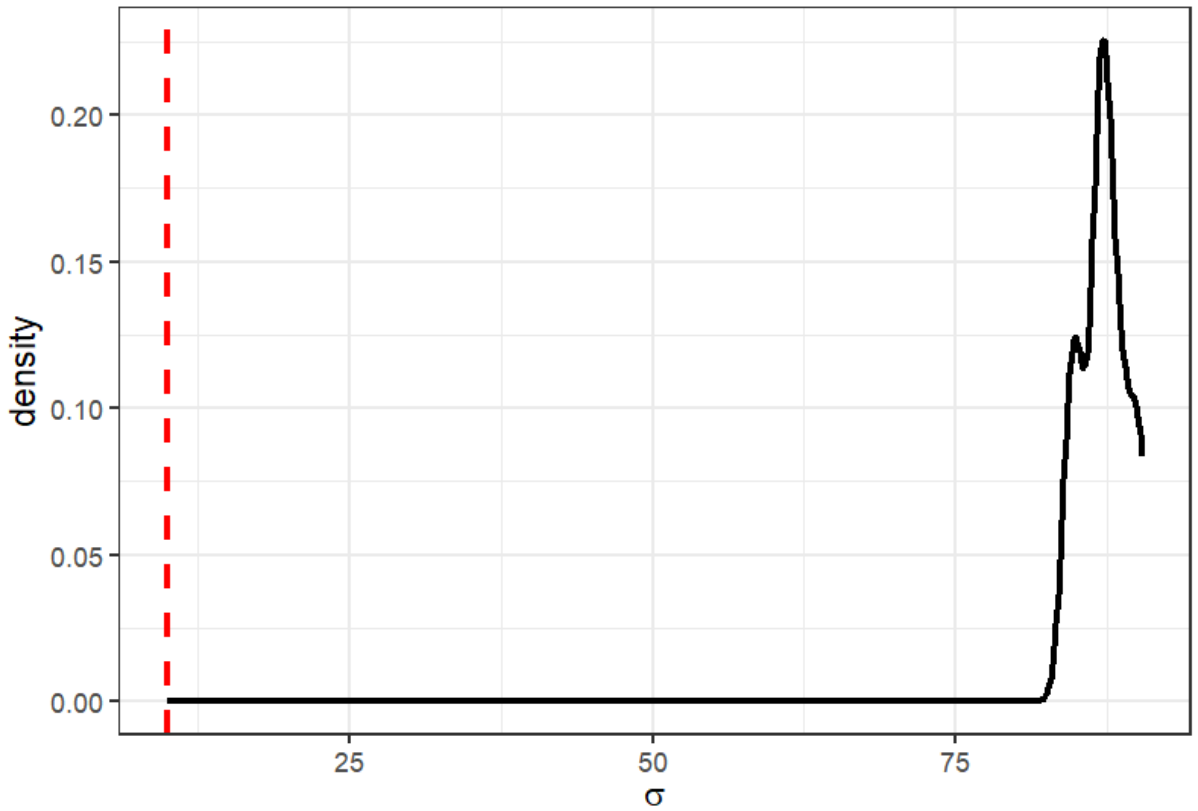
(3.2).

• nsamp =100

```r
### 3.2 Check posterior senstivity to the total no of samples
# 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=33)              # 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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

```

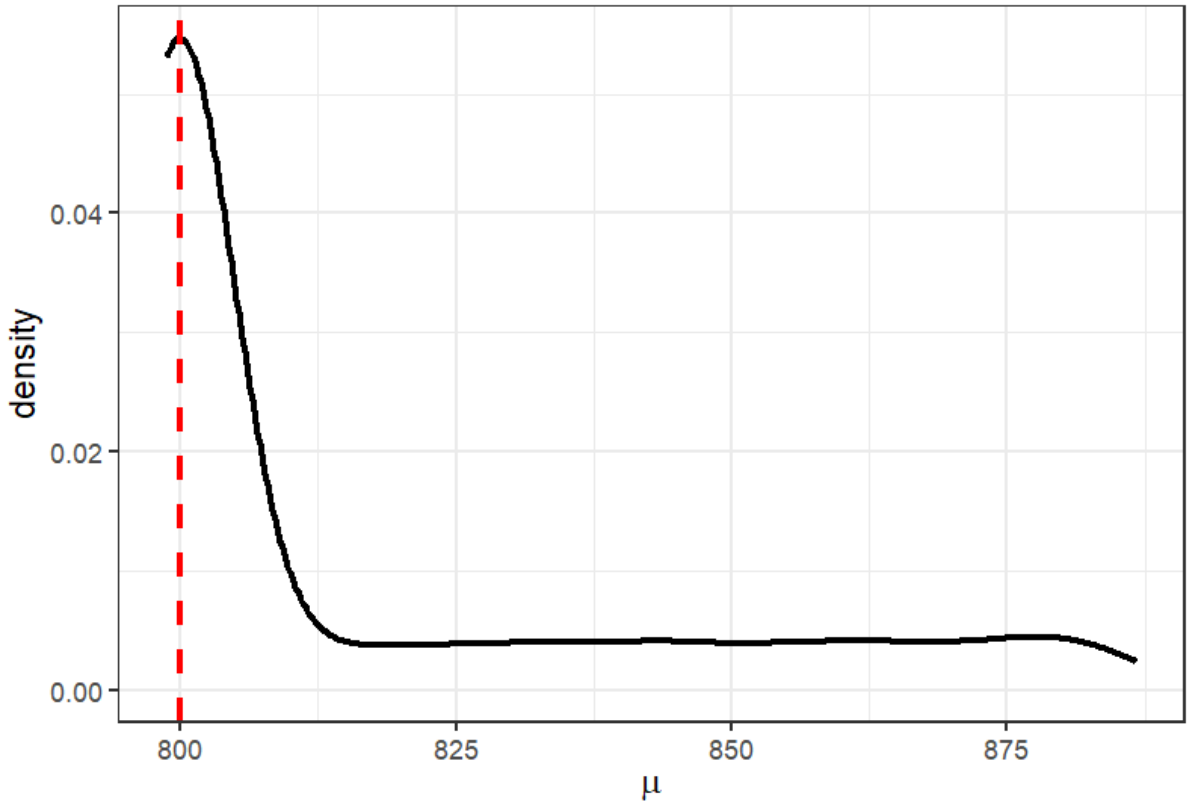



```r
# plot of sigma_chain
ggplot(df.posterior, aes(x=sigma_chain))+
  geom_density(size=1)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept = 10,size=1,colour="red",linetype="dashed")

```

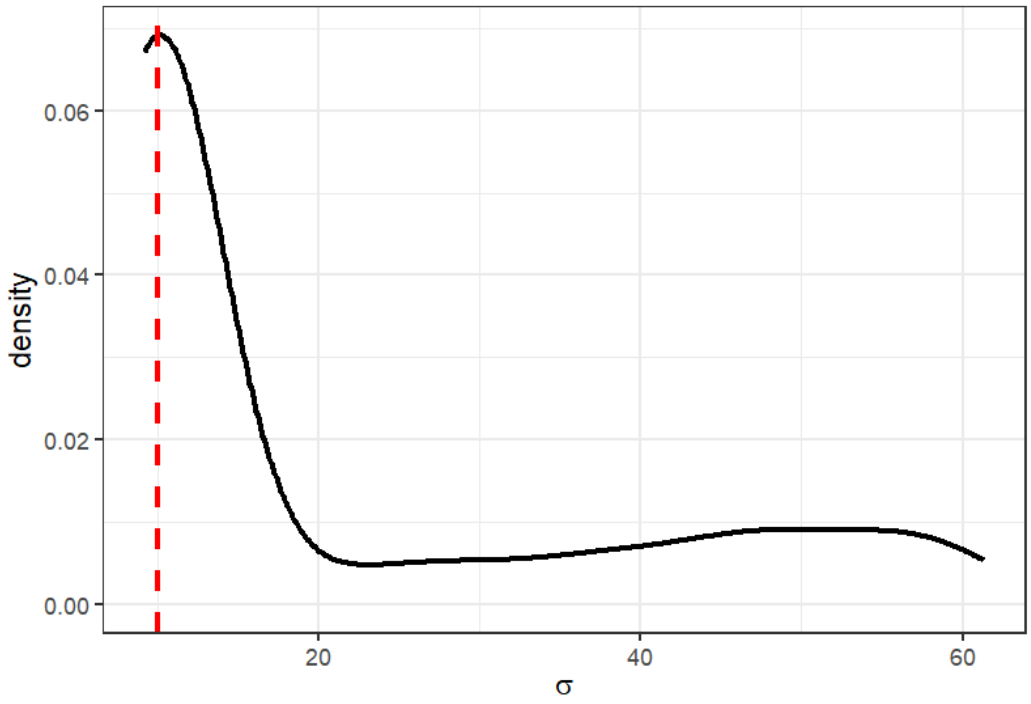


• nsamp =1000

```
# 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=333)               # 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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

```

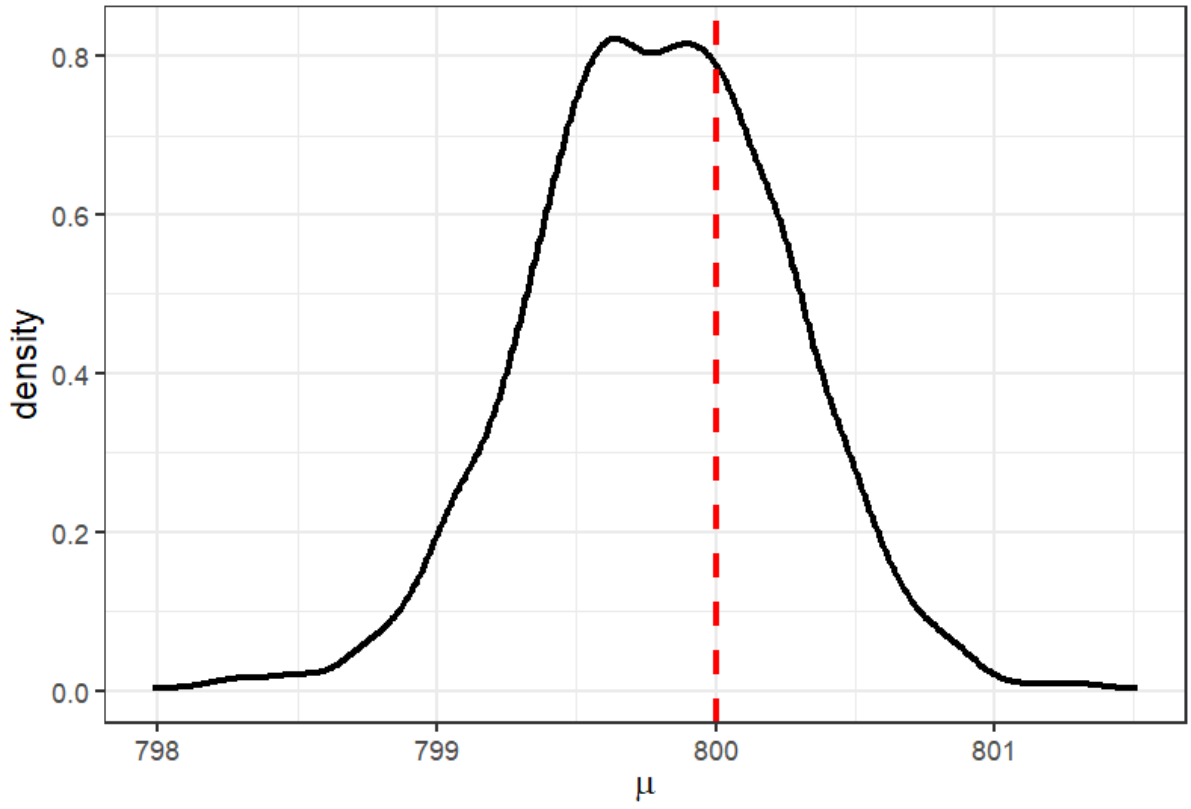



```
1. # plot of sigma_chain
2. ggplot(df.posterior, aes(x=sigma_chain))+
3.   geom_density(size=1)+theme_bw()+
4.   theme(legend.title = element_blank(),legend.position = "top")+
5.   xlab(expression(sigma))+
6.   geom_vline(xintercept = 10,size=1,colour="red",linetype="dashed")
7.
```

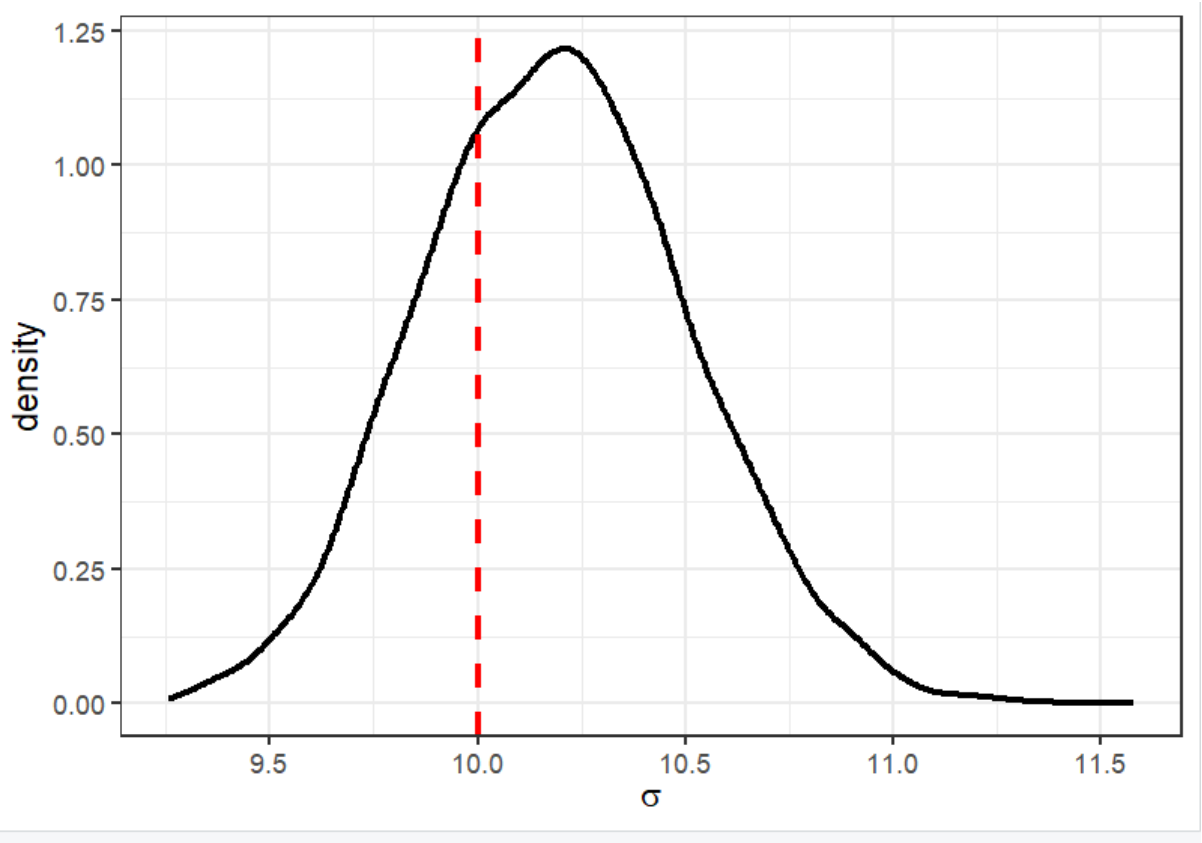


• nsamp =6000

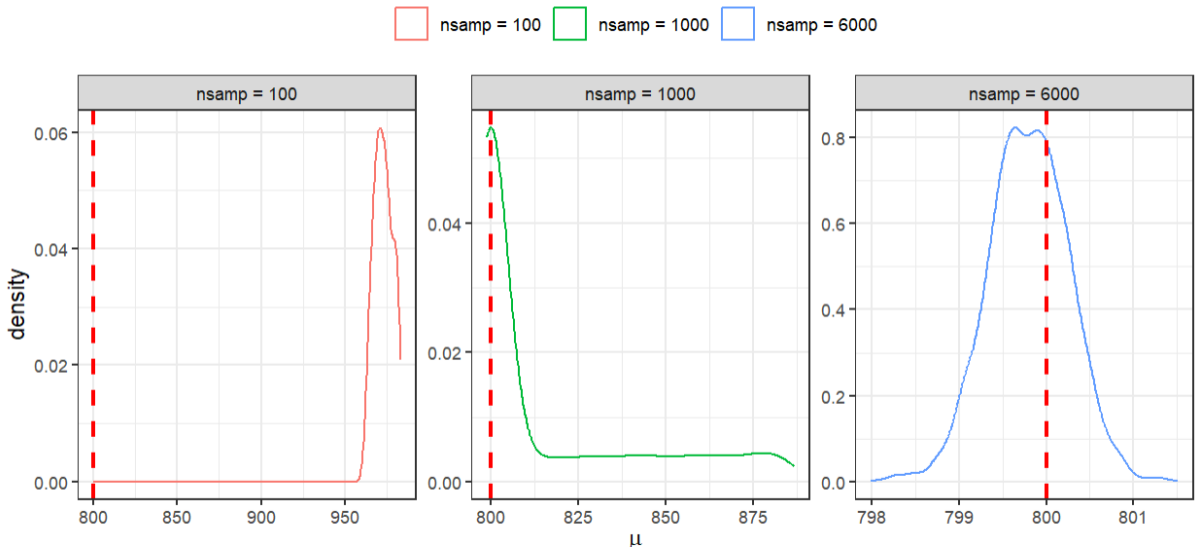
```r
# nsamp = 6000
nsamp <- 6000
df.posterior <- HMC(y=y,n=length(y),        # data
                    m=1000,s=20,a=10,b=2,   # priors
                    step=0.02,              # step-size
                    L=12,                   # no. of leapfrog steps
                    initial_q=c(1000,11),   # Chain initialization
                    nsamp=nsamp,            # total number of samples
                    nburn=2000)               # 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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

```



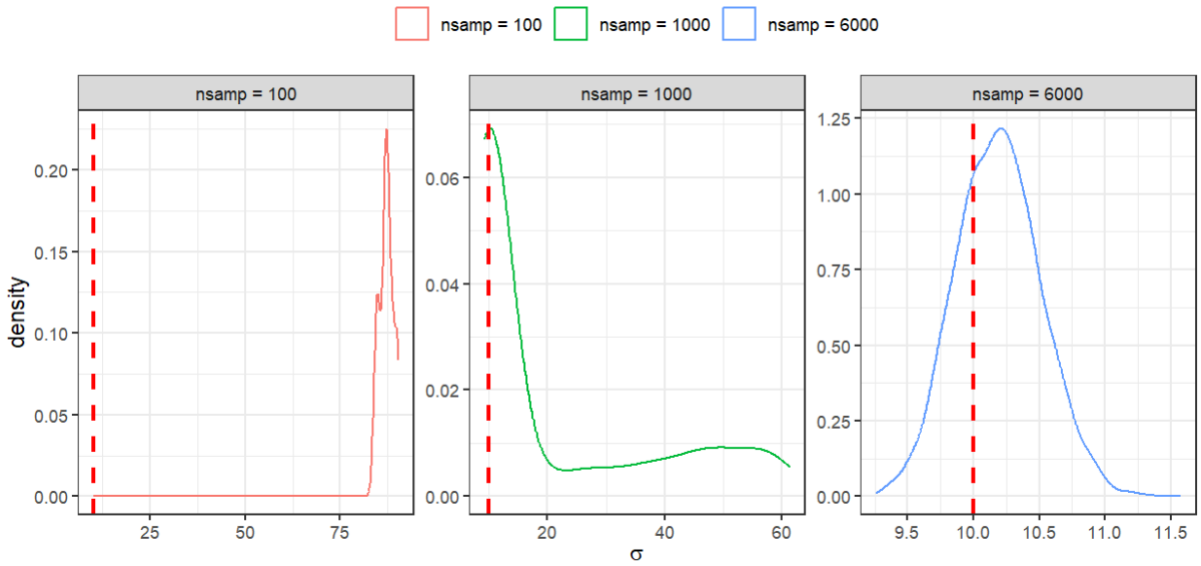

```r
# plot of sigma_chain
ggplot(df.posterior, aes(x=sigma_chain))+
  geom_density(size=1)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept = 10,size=1,colour="red",linetype="dashed")

```




```
1. # plotting altogether of mu_chain
2. ggplot(posterior, aes(x=mu_chain,group=type,colour = type))+
3.   geom_density()+theme_bw()+
4.   theme(legend.title = element_blank(),legend.position = "top")+
5.   xlab(expression(mu))+
6.   geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")+
7.   facet_wrap(~type,scales="free")
8.
```



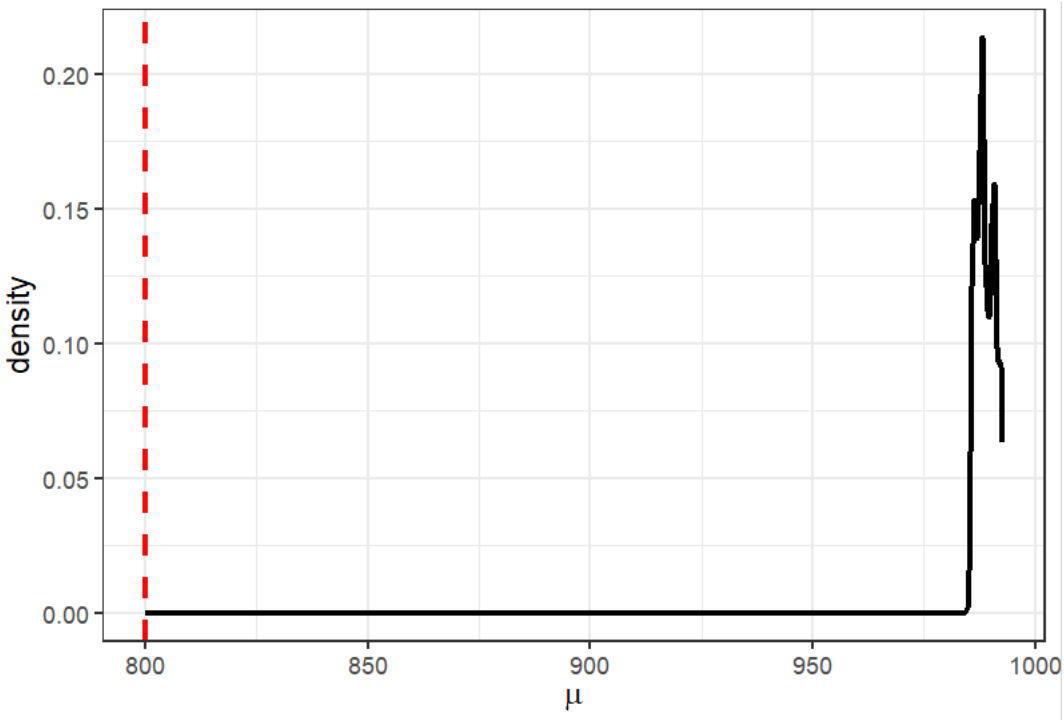
```r
# plotting altogether of sigma_chain
ggplot(posterior, aes(x=sigma_chain,group=type,colour = type))+
  geom_density()+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept = 10,size=1,colour="red",linetype="dashed")+
  facet_wrap(~type,scales="free")

```



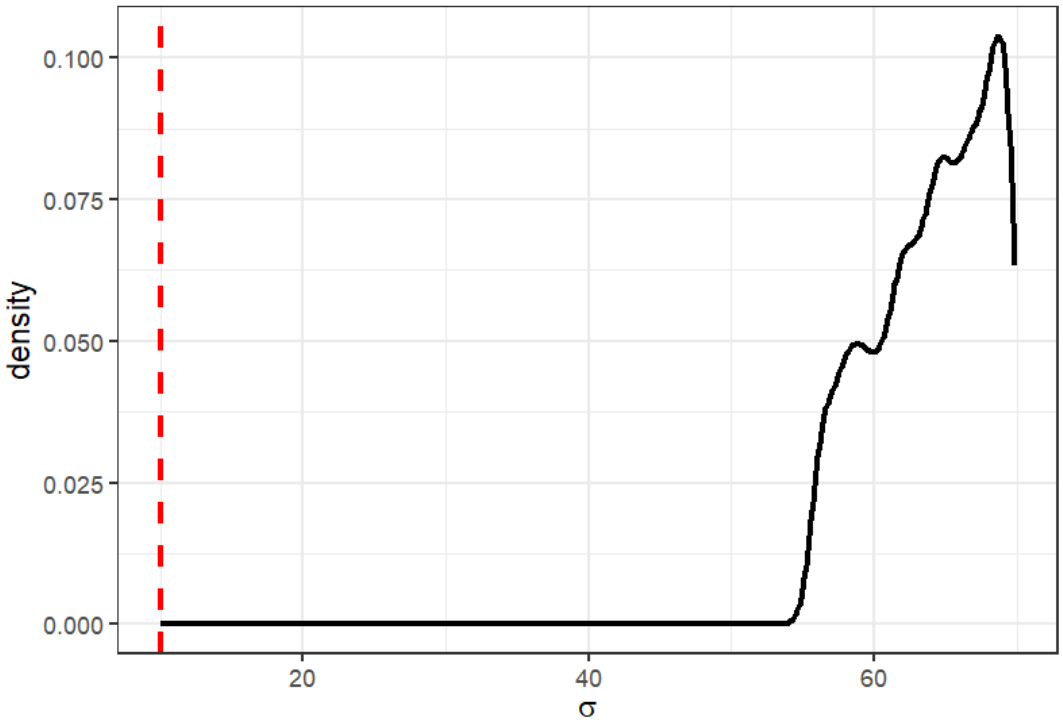
(3.3). Estimate and compute the posteriors obtained when step-size changes

• step = 0.001

```r
### Estimate and compute the posteriors obtained when step-size changes
# 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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

```

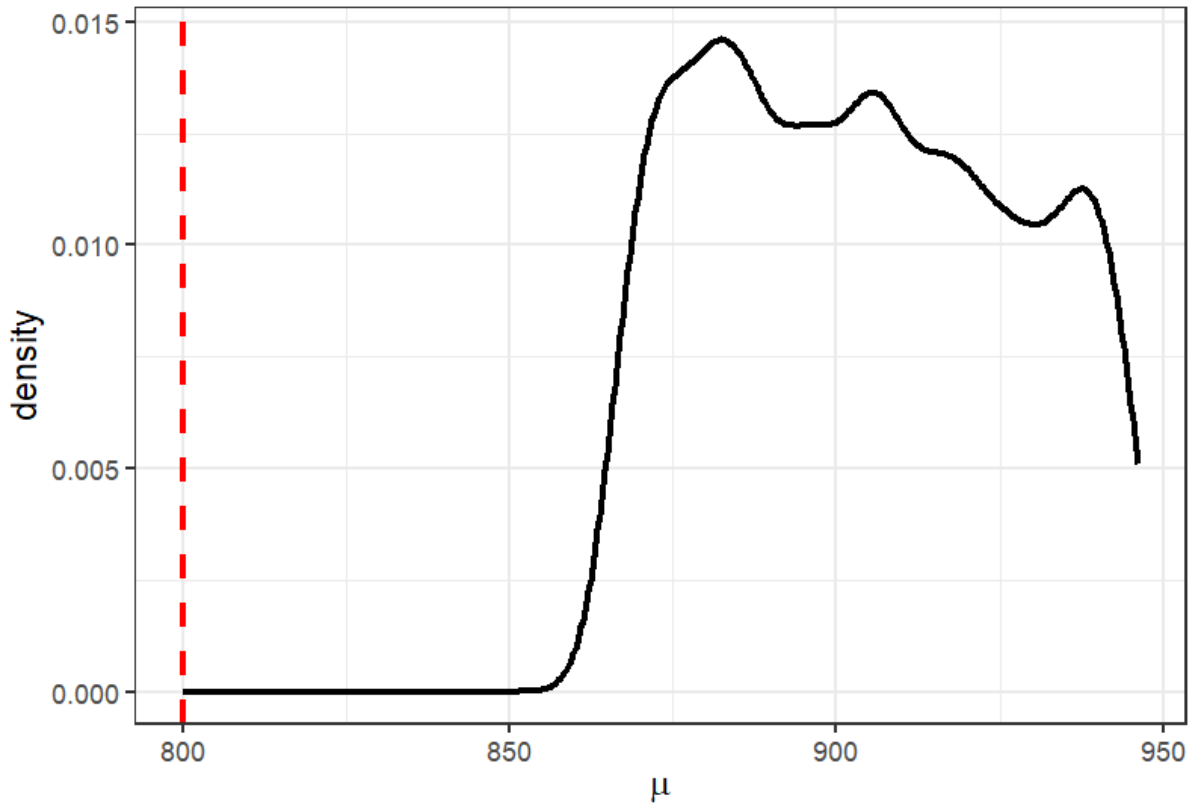



```
# plot of sigma_chain
ggplot(df.posterior, aes(x=sigma_chain))+
  geom_density(size=1)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept = 10,size=1,colour="red",linetype="dashed")

```

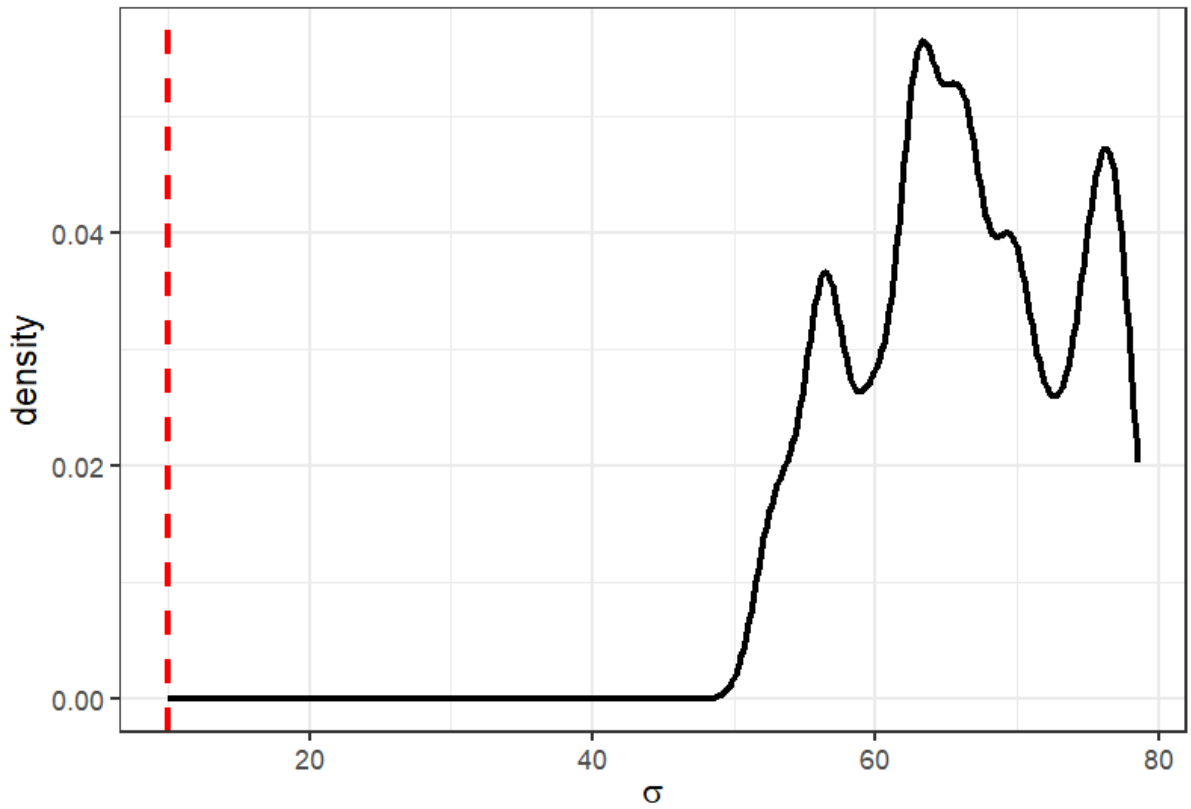


• step = 0.005

```r
# 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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

```

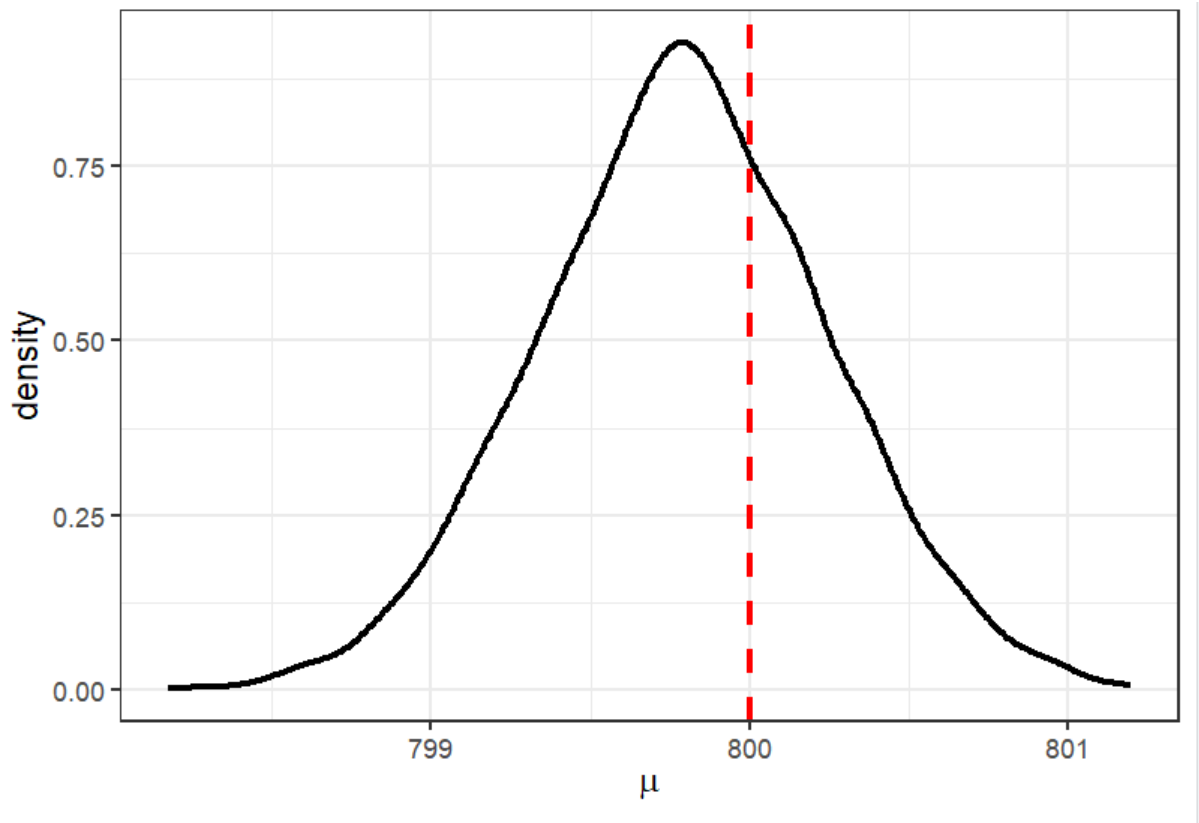



```r
# plot of sigma_chain
ggplot(df.posterior, aes(x=sigma_chain))+
  geom_density(size=1)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept = 10,size=1,colour="red",linetype="dashed")

```

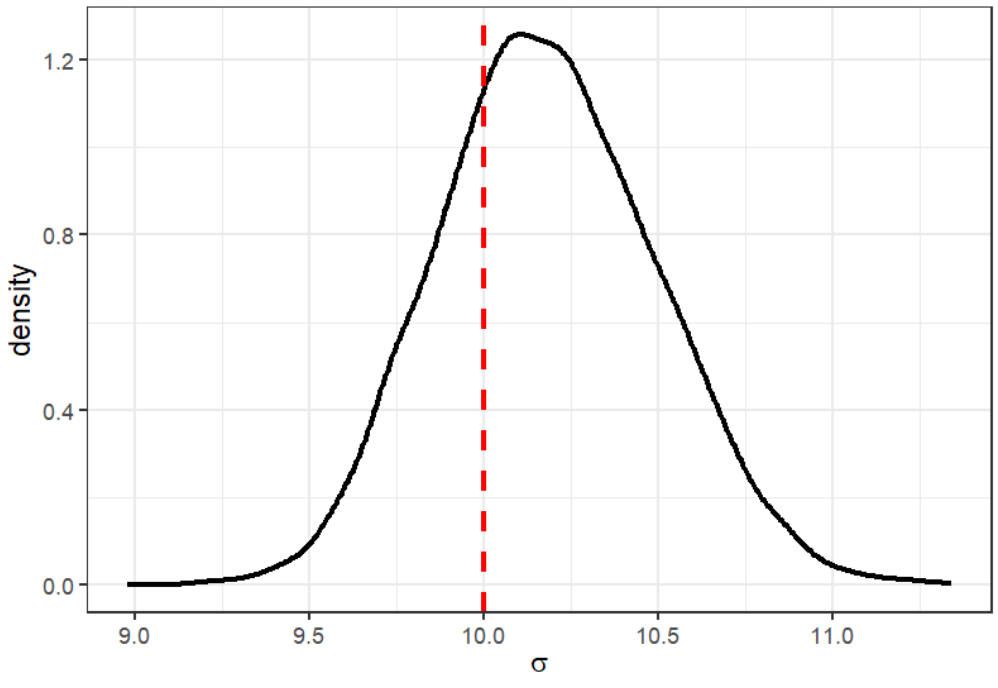


• step = 0.02

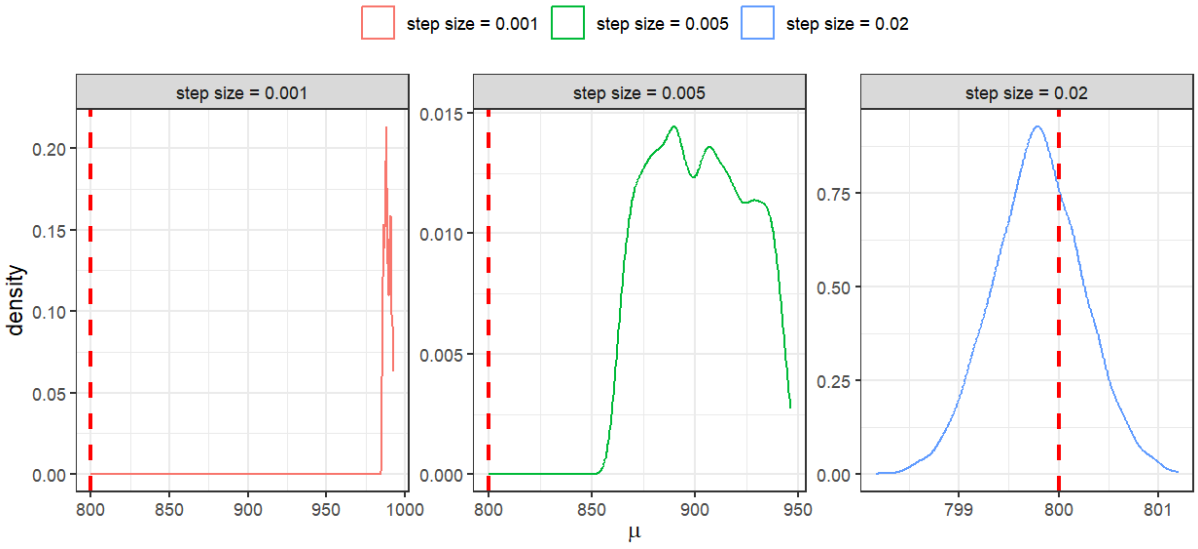
```r
# step size = 0.02
step_size <- 0.02
df.posterior <- HMC(y=y,n=length(y),        # data
                    m=1000,s=20,a=10,b=2,   # priors
                    step=step_size,             # step-size
                    L=12,                   # no. of leapfrog steps
                    initial_q=c(1000,11),   # 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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

```



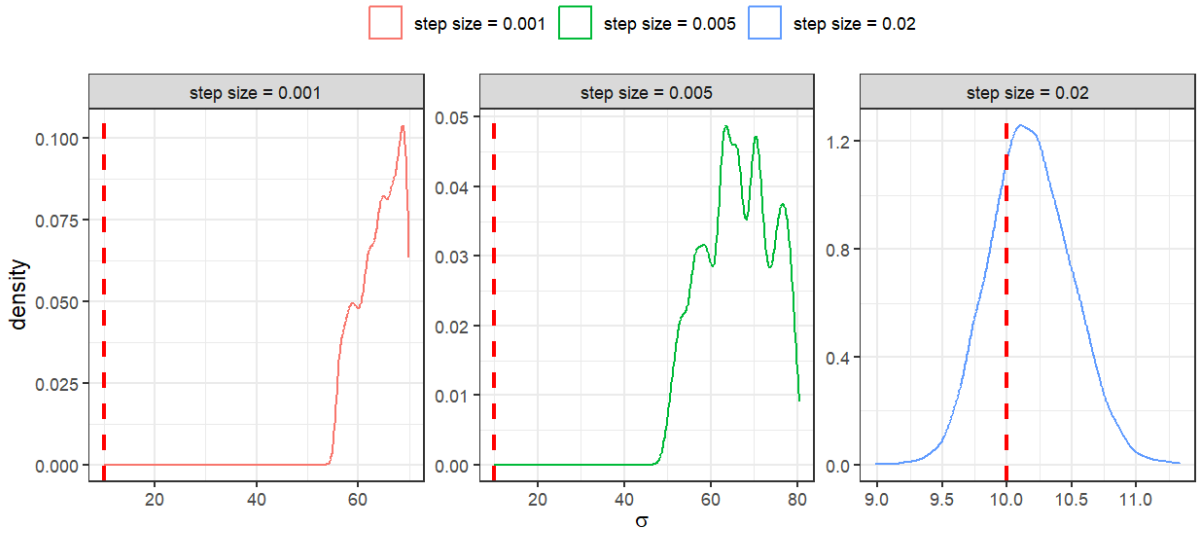

```
1. # plot of sigma_chain
2. ggplot(df.posterior, aes(x=sigma_chain))+
3.   geom_density(size=1)+theme_bw()+
4.   theme(legend.title = element_blank(),legend.position = "top")+
5.   xlab(expression(sigma))+
6.   geom_vline(xintercept = 10,size=1,colour="red",linetype="dashed")
7.
```



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

```

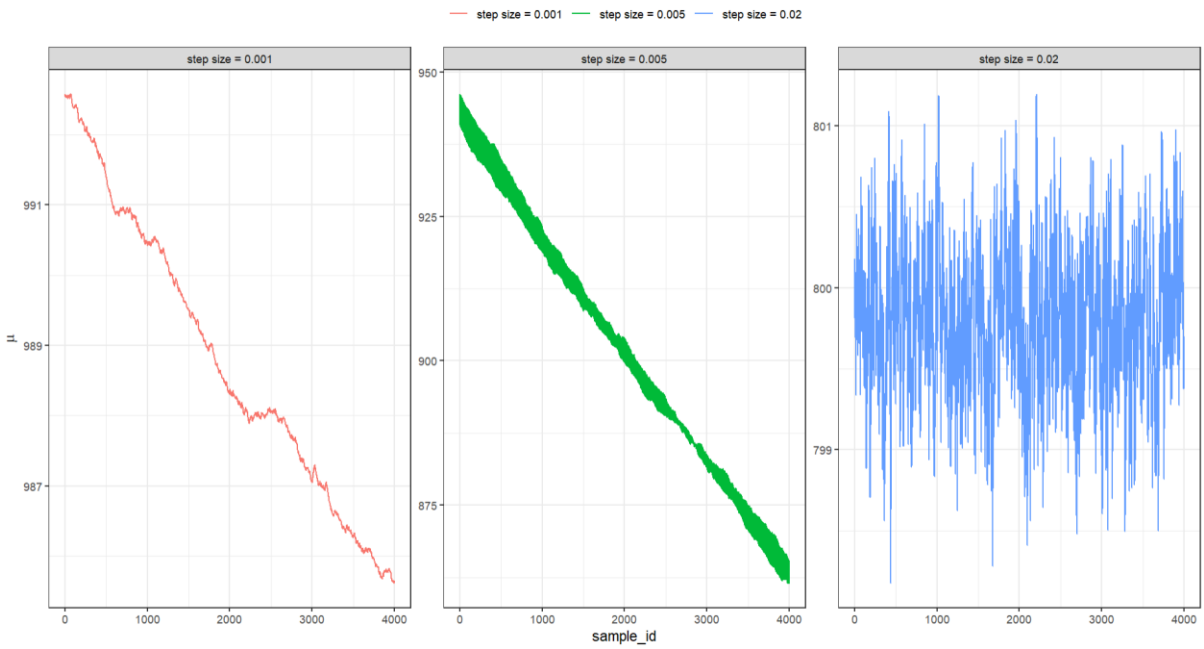



```
# plotting altogether of sigma_chain
ggplot(posterior, aes(x=sigma_chain,group=type,colour = type))+
  geom_density()+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(sigma))+
  geom_vline(xintercept = 10,size=1,colour="red",linetype="dashed")+
  facet_wrap(~type,scales="free")

```

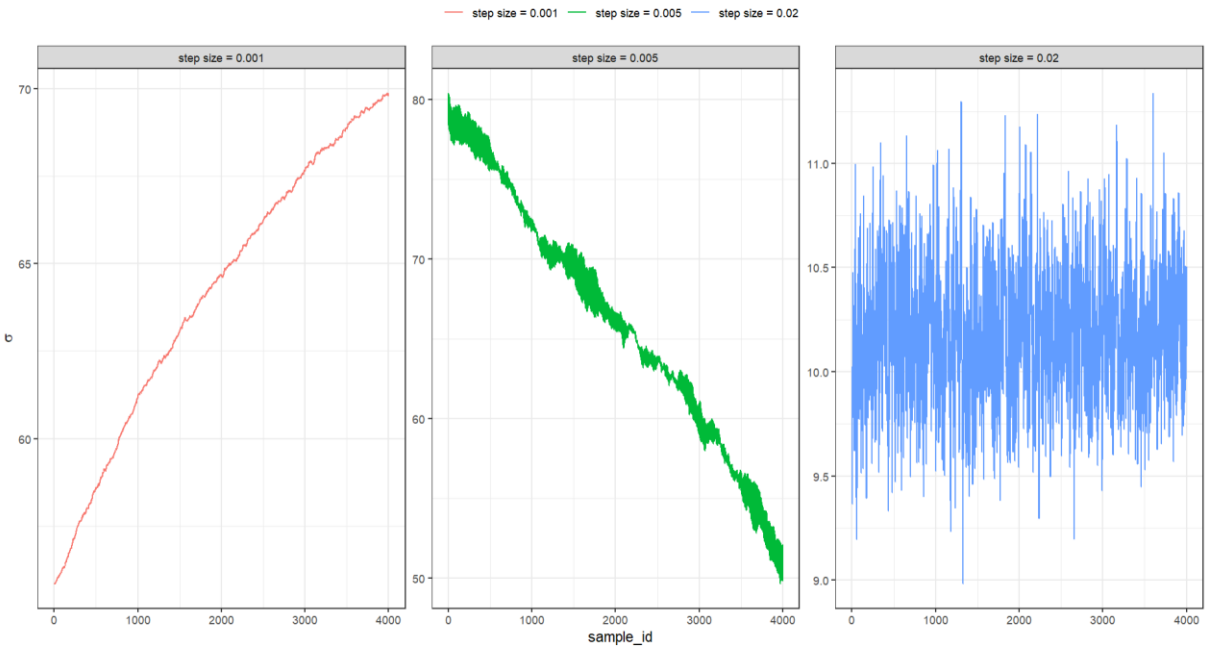


(3.4).

```
### 3.4
# 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:

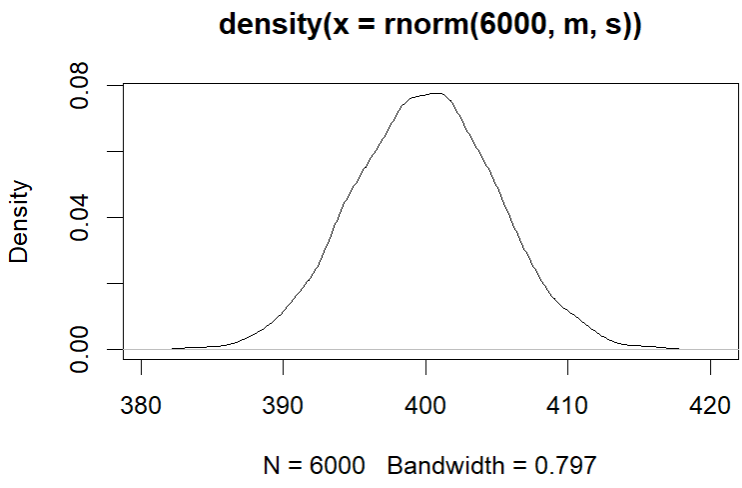
Smaller value of step size results in shorter trajectories as compared to larger value of step size. Due to this short trajectories may not explore the entire posterior distribution for both the parameters.

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.

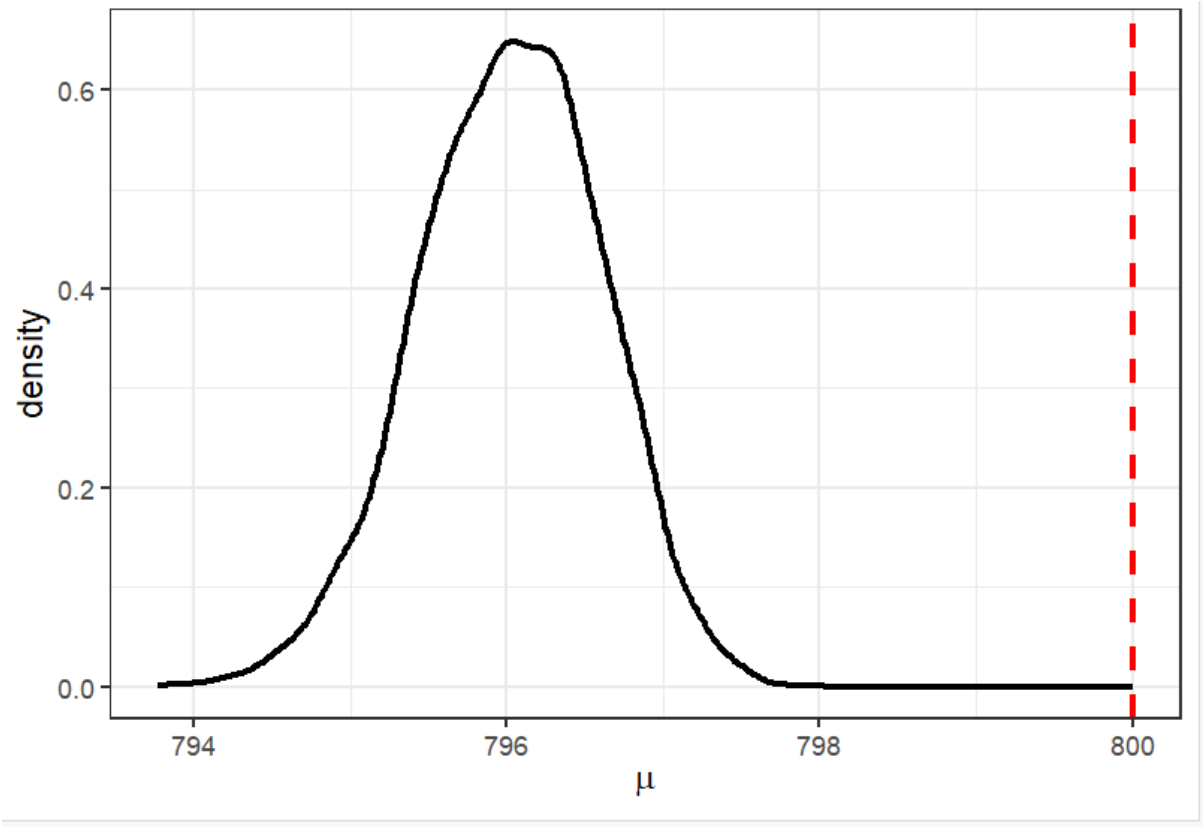
(3.5).

• μ~Normal(m =400,s = 5)

```
### 3.5
# for μ~Normal(m =400,s = 5)
m <- 400
s <- 5
plot(density(rnorm(6000,m,s)))

```

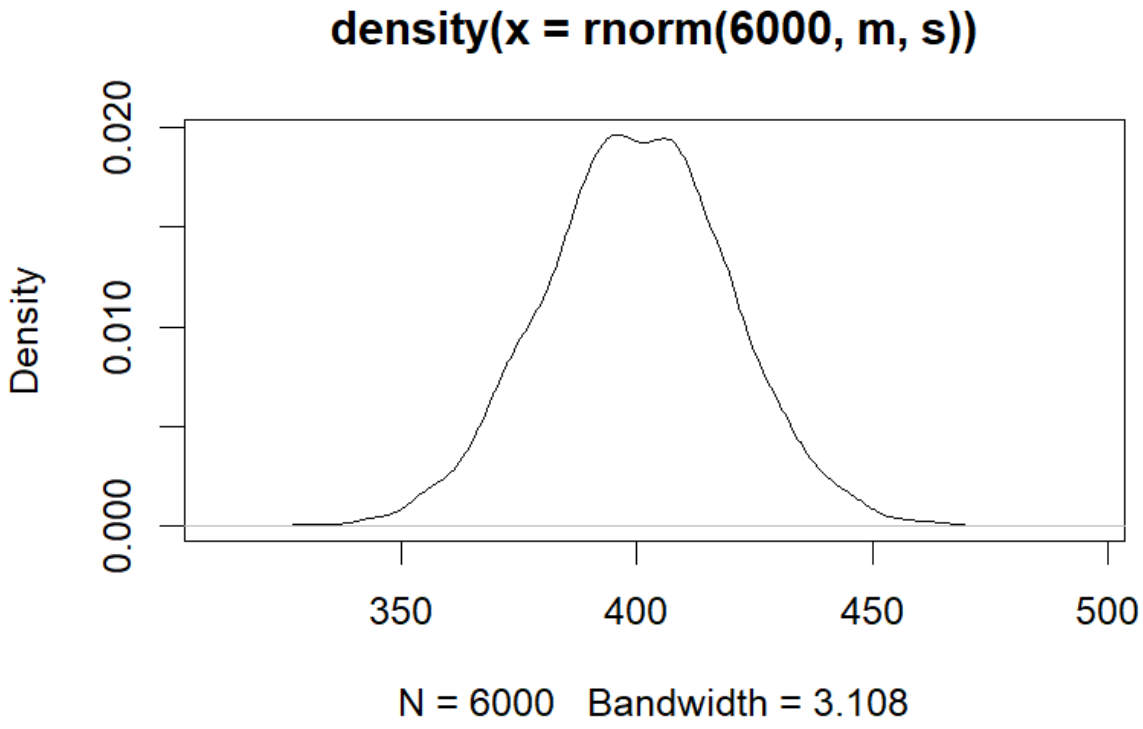


```r
df.posterior <- HMC(y=y,n=length(y),        # data
                    m=m,s=s,a=10,b=2,       # priors
                    step=step_size,         # step-size
                    L=12,                   # no. of leapfrog steps
                    initial_q=c(1000,11),   # Chain initialization
                    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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

```

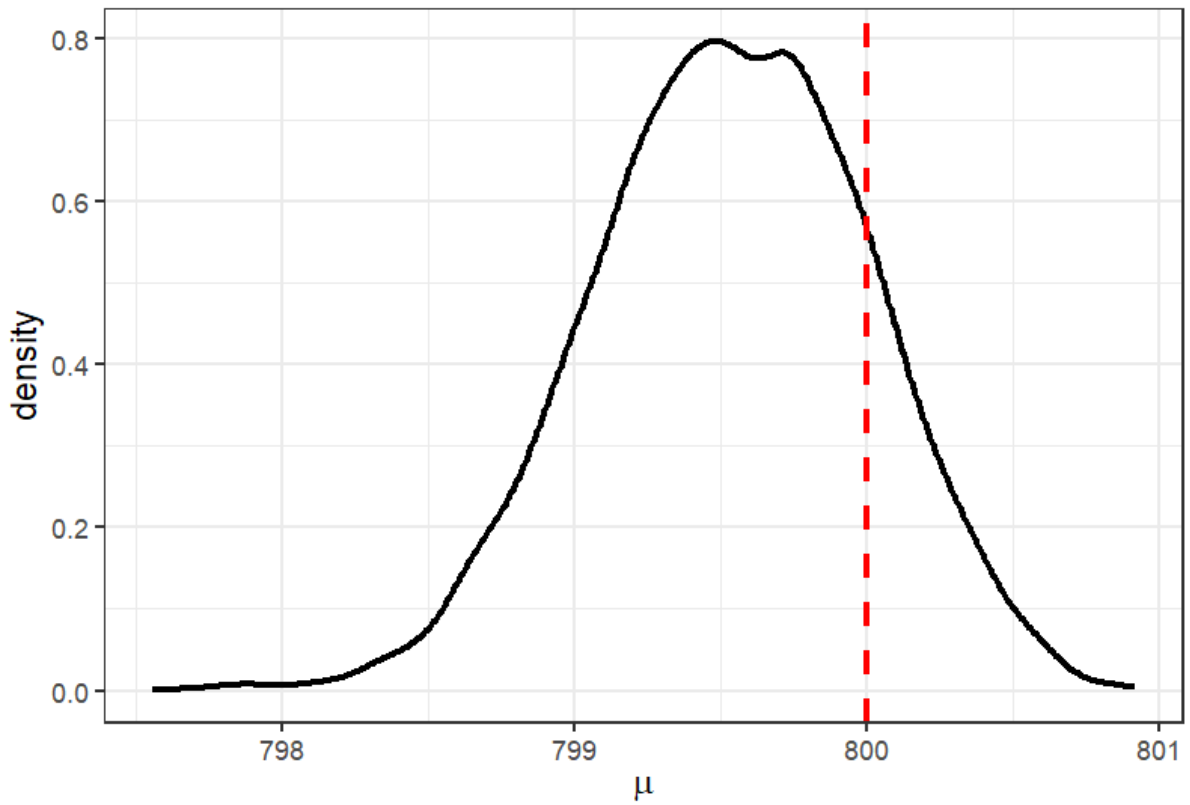


• μ~Normal(m =400,s = 20)

```r
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)))

```

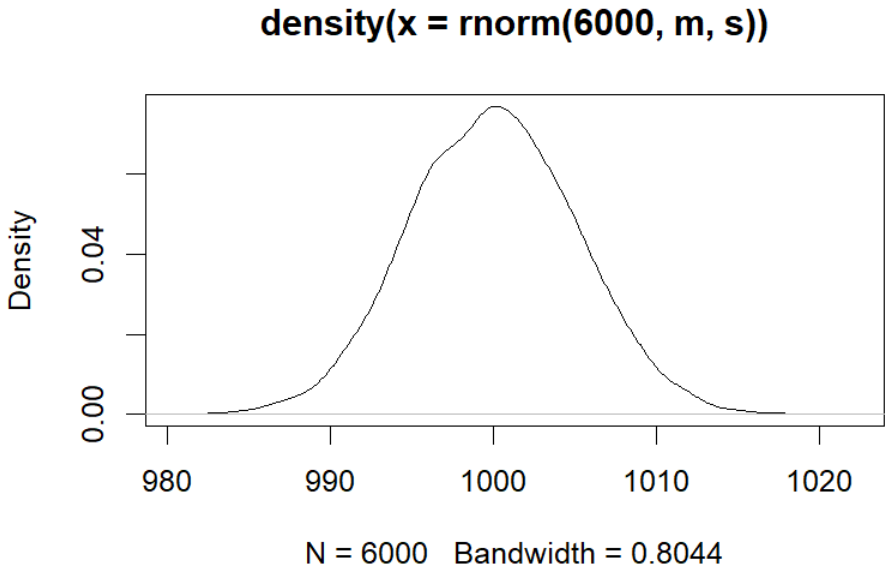



```r
df.posterior <- HMC(y=y,n=length(y),        # data
                    m=m,s=s,a=10,b=2,       # priors
                    step=step_size,         # step-size
                    L=12,                   # no. of leapfrog steps
                    initial_q=c(1000,11),   # Chain initialization
                    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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

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

```

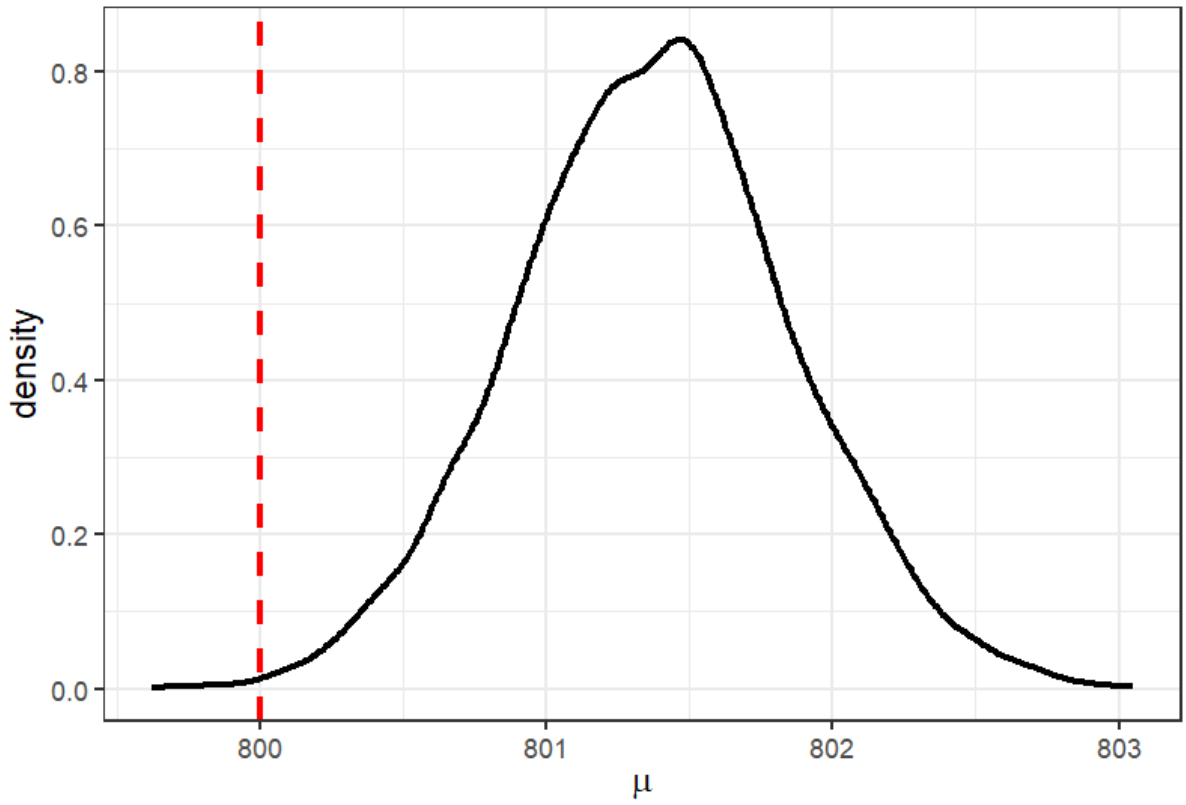


• μ~Normal(m =1000,s = 5)

```
1. # μ~Normal(m =1000,s = 5)
2. m <- 1000
3. s <- 5
4. plot(density(rnorm(6000,m,s)))
5.
```

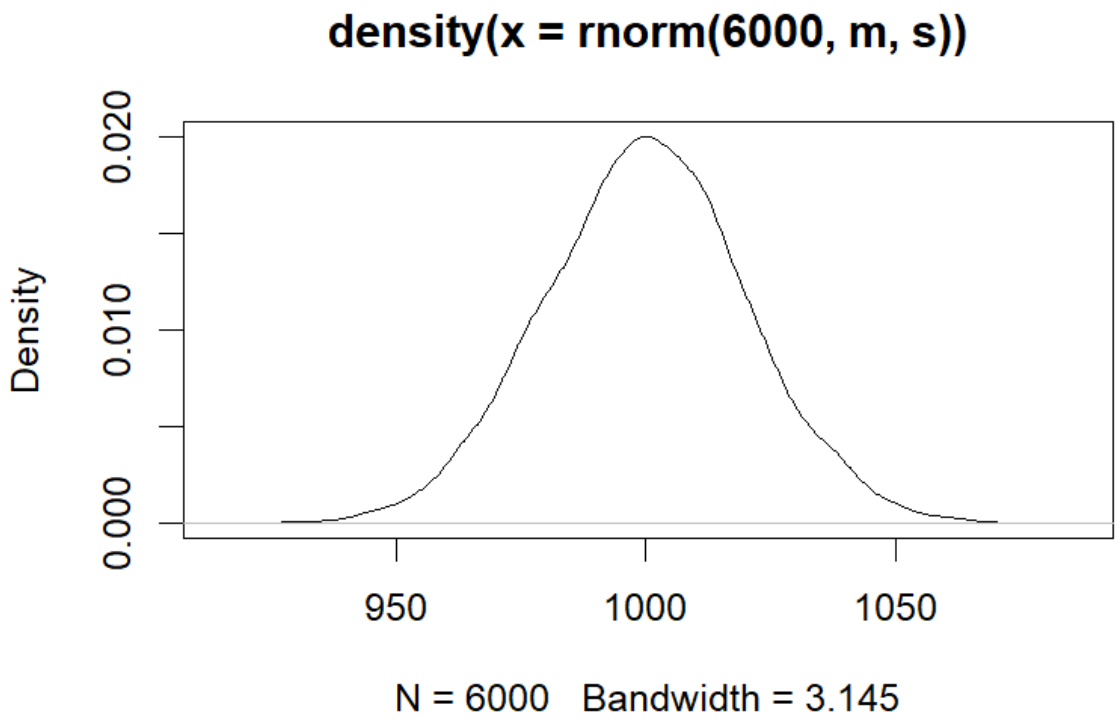


```r
df.posterior <- HMC(y=y,n=length(y),        # data
                    m=m,s=s,a=10,b=2,       # priors
                    step=step_size,         # step-size
                    L=12,                   # no. of leapfrog steps
                    initial_q=c(1000,11),   # Chain initialization
                    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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

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

```

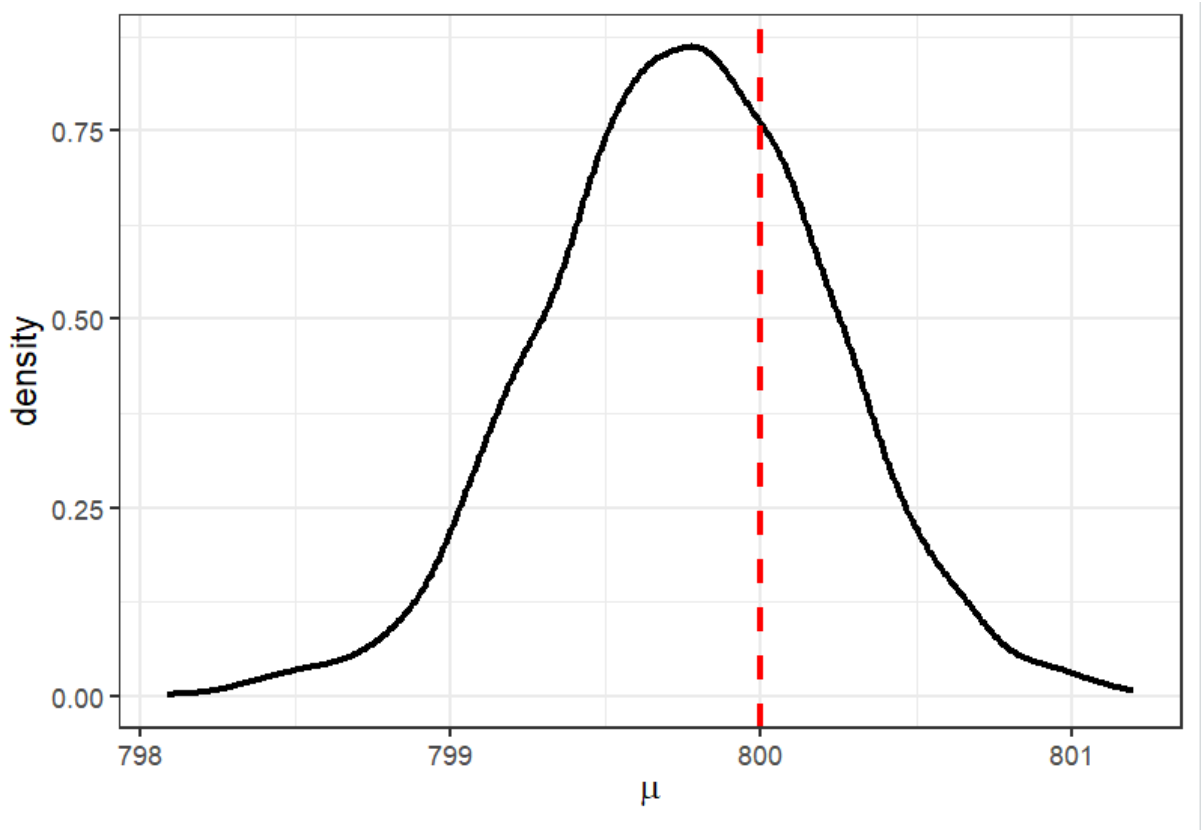


- μ~Normal(m =1000,s = 20)

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

```

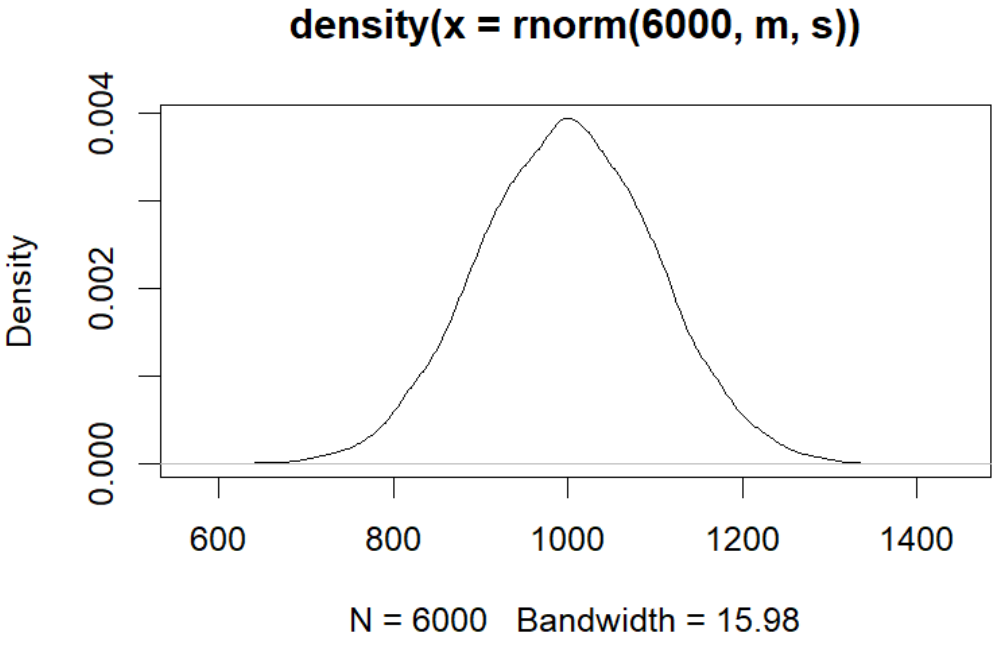



```r
df.posterior <- HMC(y=y,n=length(y),        # data
                    m=m,s=s,a=10,b=2,       # priors
                    step=step_size,         # step-size
                    L=12,                   # no. of leapfrog steps
                    initial_q=c(1000,11),   # Chain initialization
                    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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

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

```

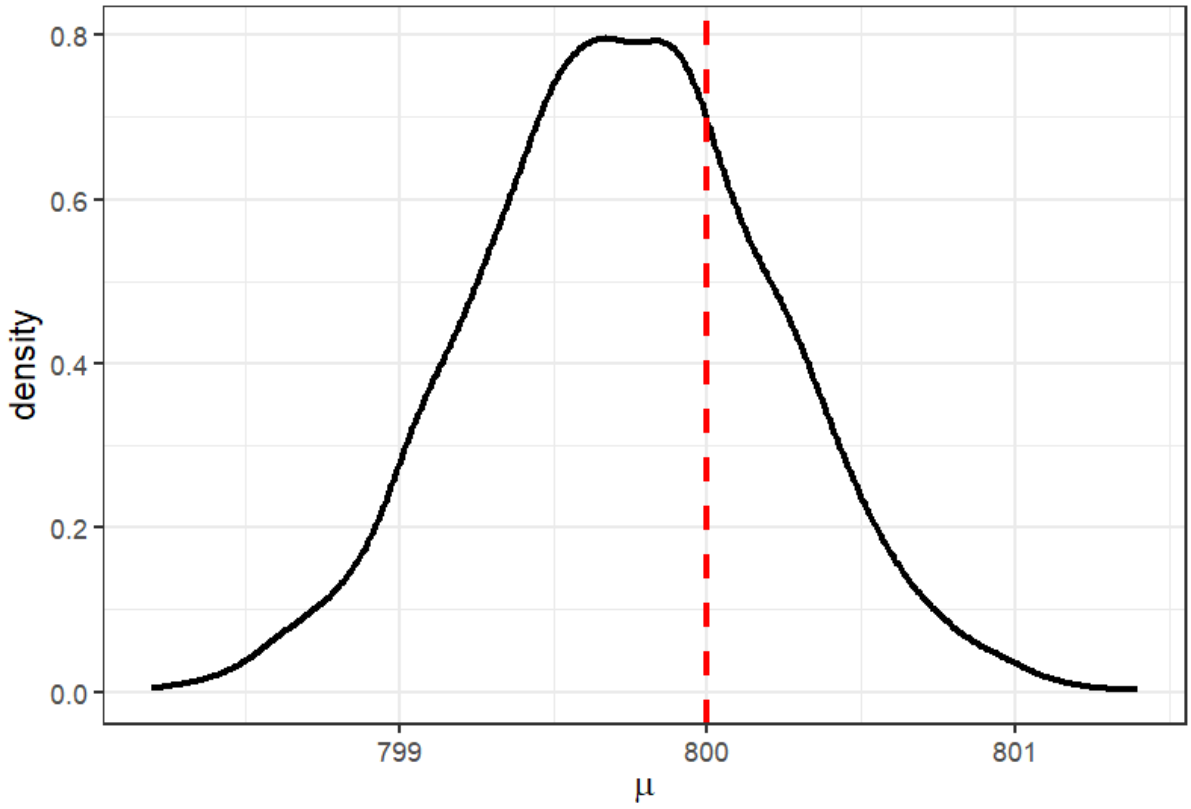


• μ~Normal(m =1000,s = 100)

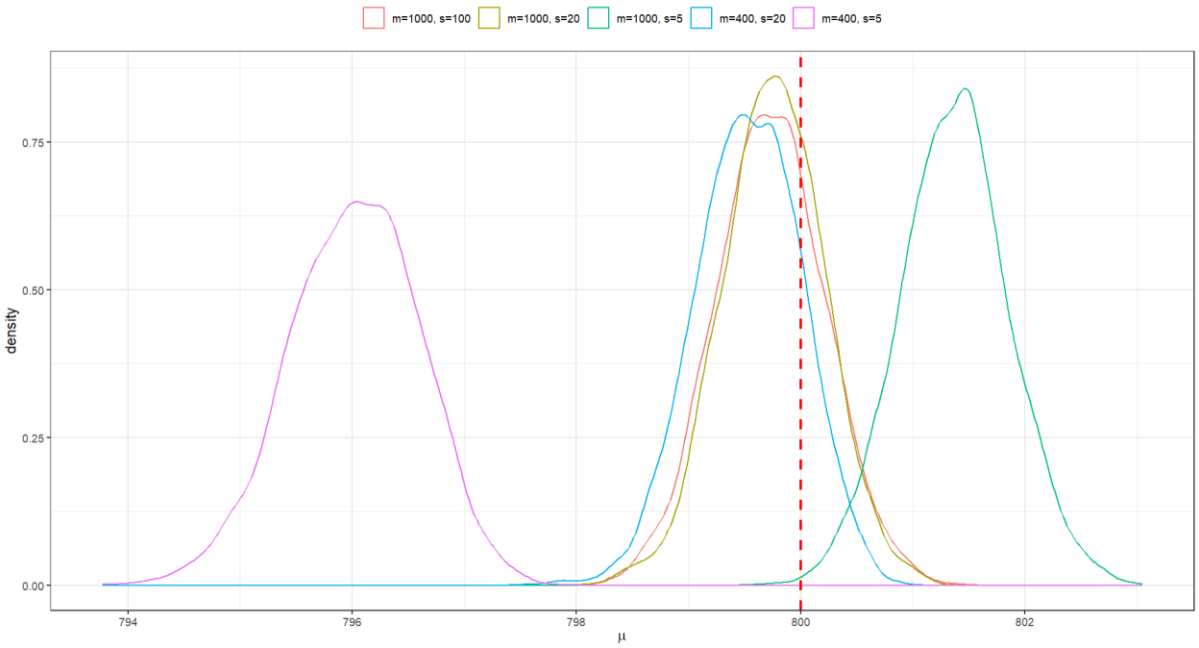
```
1. # μ~Normal(m =1000,s = 100)
2. m <- 1000
3. s <- 100
4. plot(density(rnorm(6000,m,s)))
5.
```



```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)+theme_bw()+
  theme(legend.title = element_blank(),legend.position = "top")+
  xlab(expression(mu))+
  geom_vline(xintercept = 800,size=1,colour="red",linetype="dashed")

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

```




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

```


m=1000, s=100
m=1000, s=20
m=1000, s=5
m=400, s=20
m=400, s=5
density
0.75
0.50
0.25
0.00
794
796
798
800
802
μ