551hw3

Zhijun Cai

3/10/2020

```
library(ggplot2)
library(MASS)
library(coda)
library(invgamma)
```

#1.1

```
set.seed(1)

#wx = fx(x)/gx
#sd(wx)/mean(wx)

ESS = function(m){
  set.seed(1)
  x = rt(m,df=2)
  f = dnorm(x)
  g = dt(x,df=2)
  w = f/g
  return(m/(1+(sd(w)/mean(w))^2))
}

m = c(50,100,200,500,1000)
for (i in m){
  print(ESS(i))
}
```

```
## [1] 42.15979
## [1] 83.93739
## [1] 161.3791
## [1] 429.7618
## [1] 869.9553
```

#1.2

```
ESS2 = function(m){
  x = rnorm(m,0,1)
  fx = dt(x,2)
  g = dnorm(x,0,1)
  w = fx/g
  return(m/(1+(sd(w)/mean(w))^2))
}
for (i in m){
  print(ESS2(i))
}
```

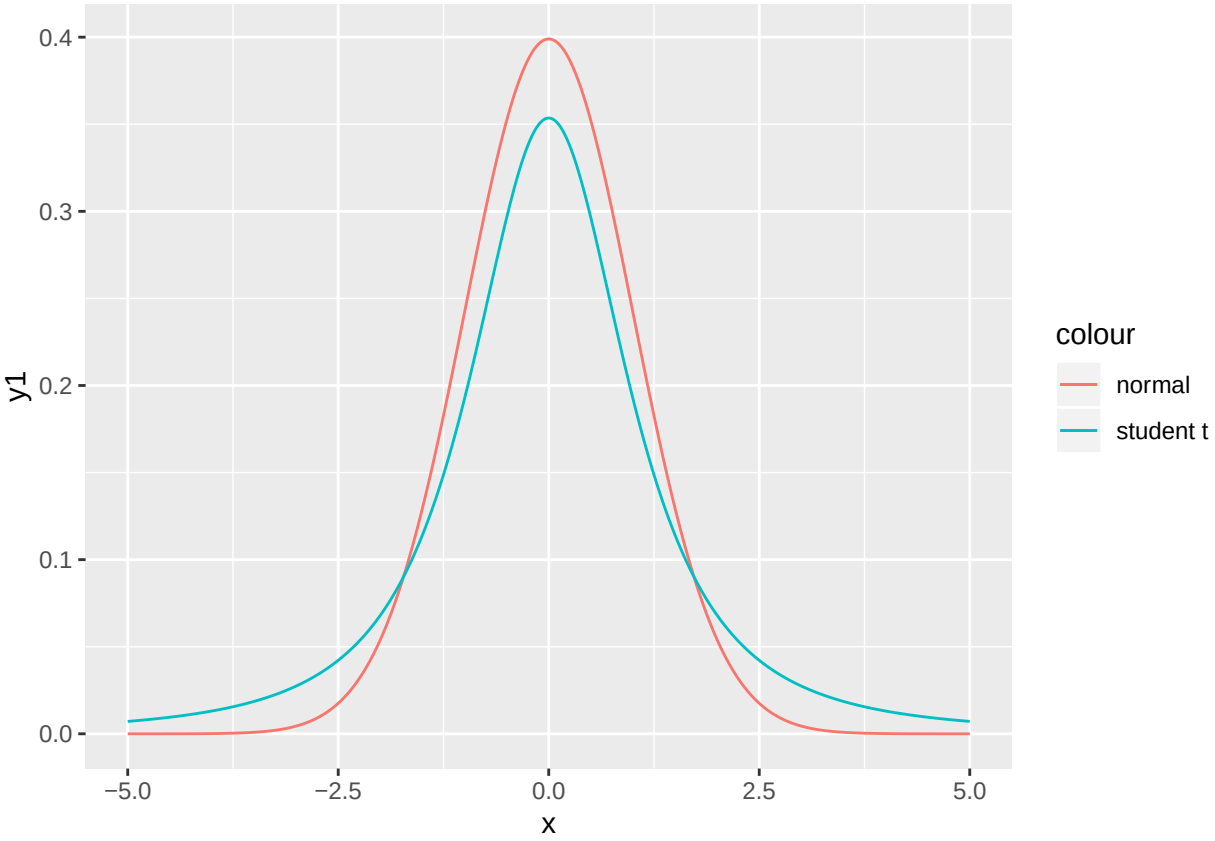
```
## [1] 47.11086
## [1] 79.37989
## [1] 190.8804
## [1] 460.3246
```

```
## [1] 369.6607
```

It is not a sensible choice because under standard normal distribution, many points contribute little to the expectation.

3

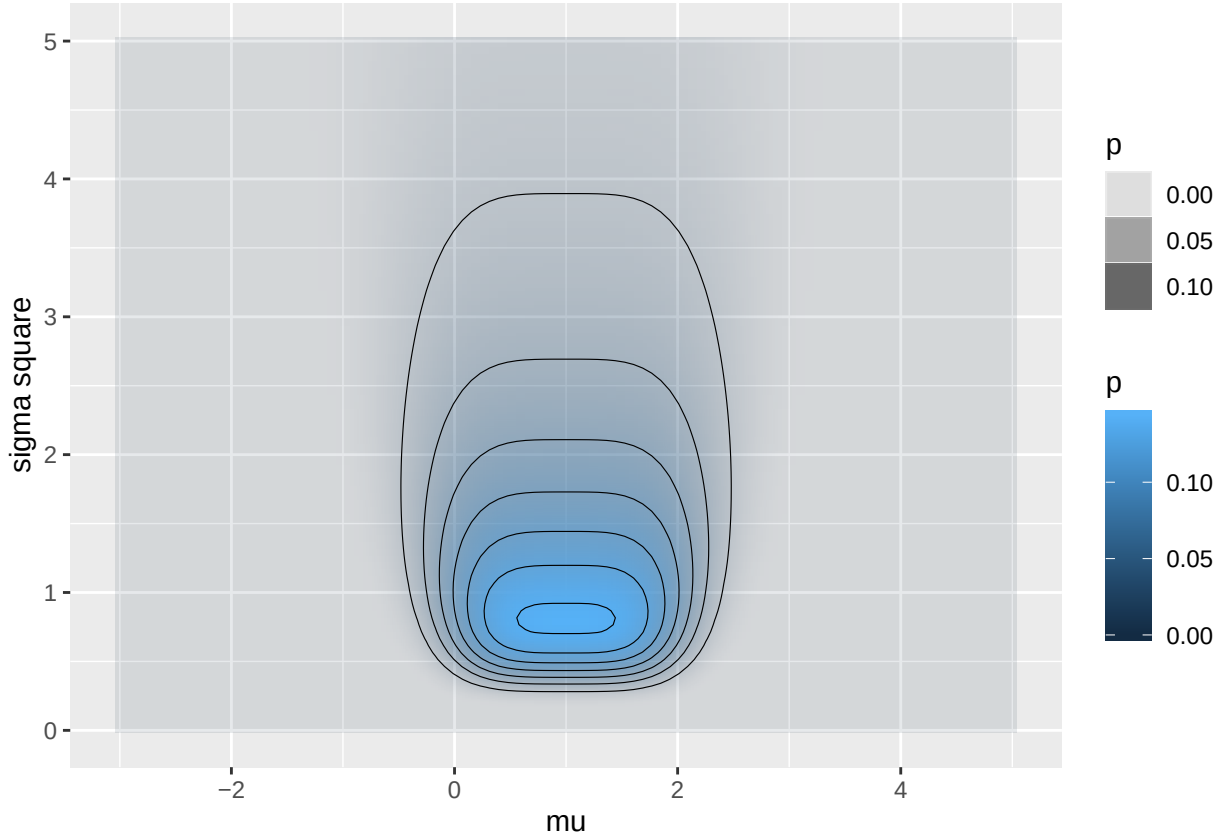
```
x = seq(from = -5, to = 5, by = 0.01)
a = dnorm(x,0,1)
b = dt(x,df=2)


ggplot(data = data.frame('x' = x,'y1' = a,'y2' = b))+
  geom_line(aes(x = x, y = y1, color = 'normal'))+
  geom_line(aes(x = x, y = y2, color = 'student t'))
```



The first one using student t distribution as an importance function is more senisble than the other because in the plot student t covers more weighted points (points far from 0) than the other way. ## 4(a)

```
mu = seq(-3,5,length.out=100)
sigmasq = seq(0.01,5,length.out = 100)
cmu = rep(mu,each = length(sigmasq))
csig = rep(sigmasq, length(mu))
fun = function(a,b){
  return(b^(-2.5)*exp(-((a-1)^4+4)/(2*b)))
}

p = fun(cmu,csig)
```

```
ggplot(data = data.frame(cmu,csig,p),aes(x=cmu,y=csig))+
  geom_raster(aes(fill = p,alpha = p),interpolate = T)+
  geom_contour(aes(z=p),colour = 'black',size = 0.2)+
  labs(x = 'mu',y = 'sigma square')
```



#4(b)

$$\pi(\mu,\sigma^2) \propto \sigma^{-5}\exp\left(-\frac{(\mu-1)^4+4}{2\sigma^2}\right)$$

$$= \frac{\sqrt{2\pi}}{\sqrt{2\pi}}\sigma^{-1}\sigma^{-4}\exp\left(-\frac{(\mu-1)^4}{2\sigma^2}\right)\exp\left(-\frac{4}{2\sigma^2}\right)$$

$$= \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(\mu-1)^4}{2\sigma^2}\right)\sqrt{2\pi}\sigma^{-4}\exp\left(-\frac{4}{2\sigma^2}\right)$$

$\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(\mu-1)^4}{2\sigma^2}\right) \sim N(1,\sigma^2)$ amd $\sqrt{2\pi}\sigma^{-4}\exp\left(-\frac{4}{2\sigma^2}\right) \propto Inv-Gamma(1,2)$

I can draw σ^2 from Inv-Gamma(1,2) and draw μ based on simulated σ^2

$$g(\mu,\sigma^2) = \frac{2}{\sqrt{2\pi}}\frac{1}{\Gamma(1)}\sigma^{-5}\exp\left(-\frac{(\mu-1)^2+4}{2\sigma^2}\right)$$

```
gfun = function(m, s2){
  return (2/(sqrt(2*pi)*gamma(1)*s2^(-5/2)) * exp(-((m-1)^2+4)/(2*s2)))
}
ESS_3 = function(m){
  sigma2_sim = rinvgamma(m,1,2)
  mu_sim = sapply(sigma2_sim, function(x){rnorm(1, 1, sqrt(x))})
```

```
  w = fun(mu_sim,sigma2_sim)/gfun(mu_sim,sigma2_sim)
  return(m/(1+(sd(w)/mean(w))^2))
}

for (i in m){
  print( ESS_3(i))
}
```

```
## [1] 5.740989
## [1] 2.820496
## [1] 3.192966
## [1] 7.165052
## [1] 19.05638
```

```
m = c(50,100,200,500,1000)

## Gird
cv = sd(p)/mean(p)
ESS_gird = length(p)/(1+cv^2)
ESS_gird
```

```
## [1] 2278.291
```

ESS for gird sampling is 2278.291.

Gibbs Sammpling

2(a) see the note

2(b)

```
library(MCMCpack)
```

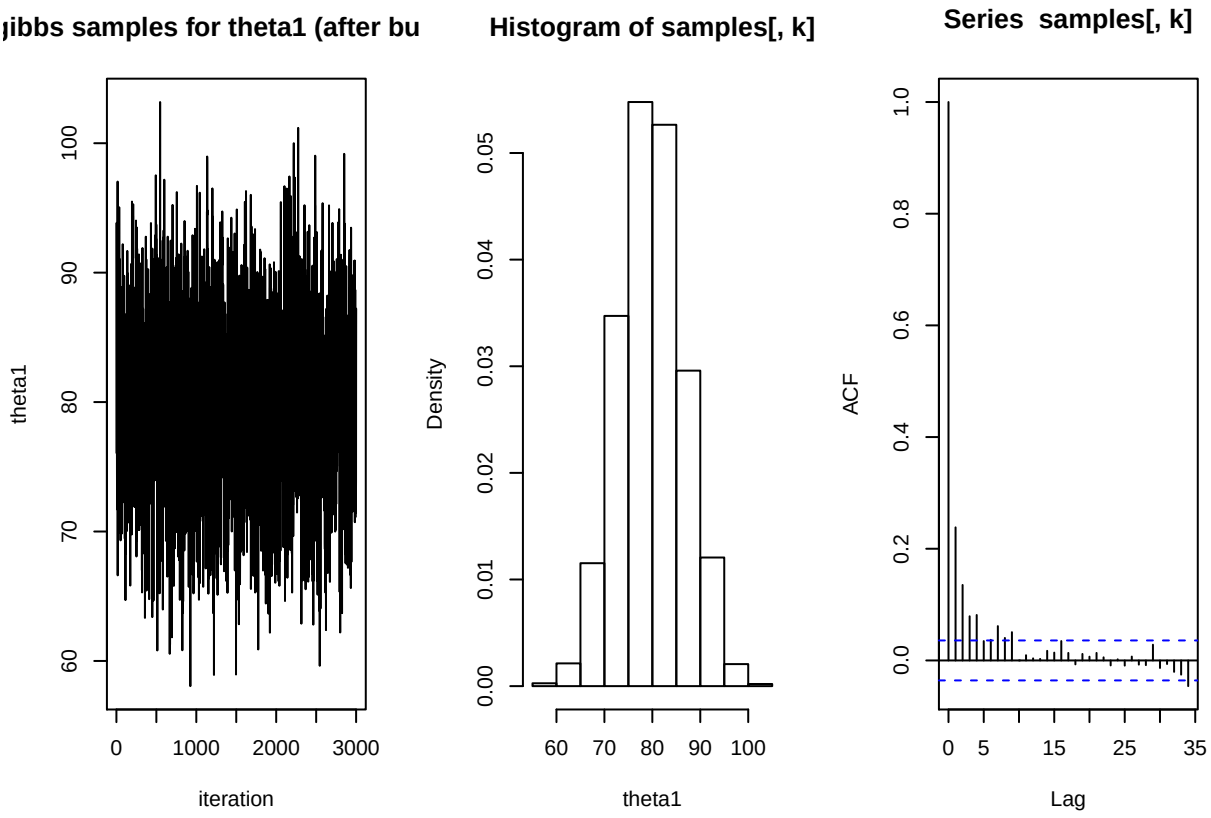
```
## ##
## ## Markov Chain Monte Carlo Package (MCMCpack)
## ## Copyright (C) 2003-2020 Andrew D. Martin, Kevin M. Quinn, and Jong Hee Park
## ##
## ## Support provided by the U.S. National Science Foundation
## ## (Grants SES-0350646 and SES-0350613)
## ##
##
## Attaching package: 'MCMCpack'
## The following objects are masked from 'package:invgamma':
##
##     dinvgamma, rinvgamma
```

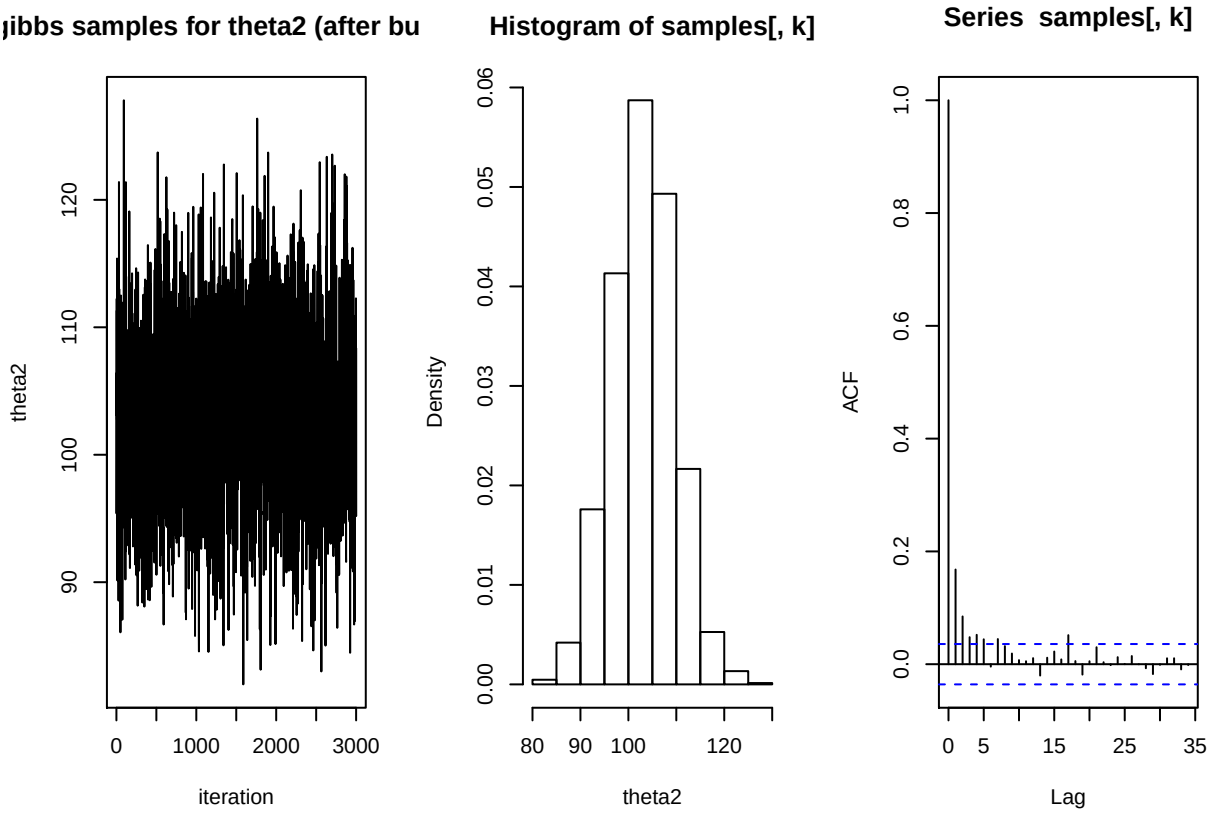
```
y = matrix(NA, nrow = 6, ncol = 5)
y[1,] = c(83,92,92,46,67)
y[2,] = c(117,109,114,104,87)
y[3,] = c(101,93,92,86,67)
y[4,] = c(105,119,116,102,116)
y[5,] = c(79,97,103,79,92)
y[6,] = c(57,92,104,77,100)
```

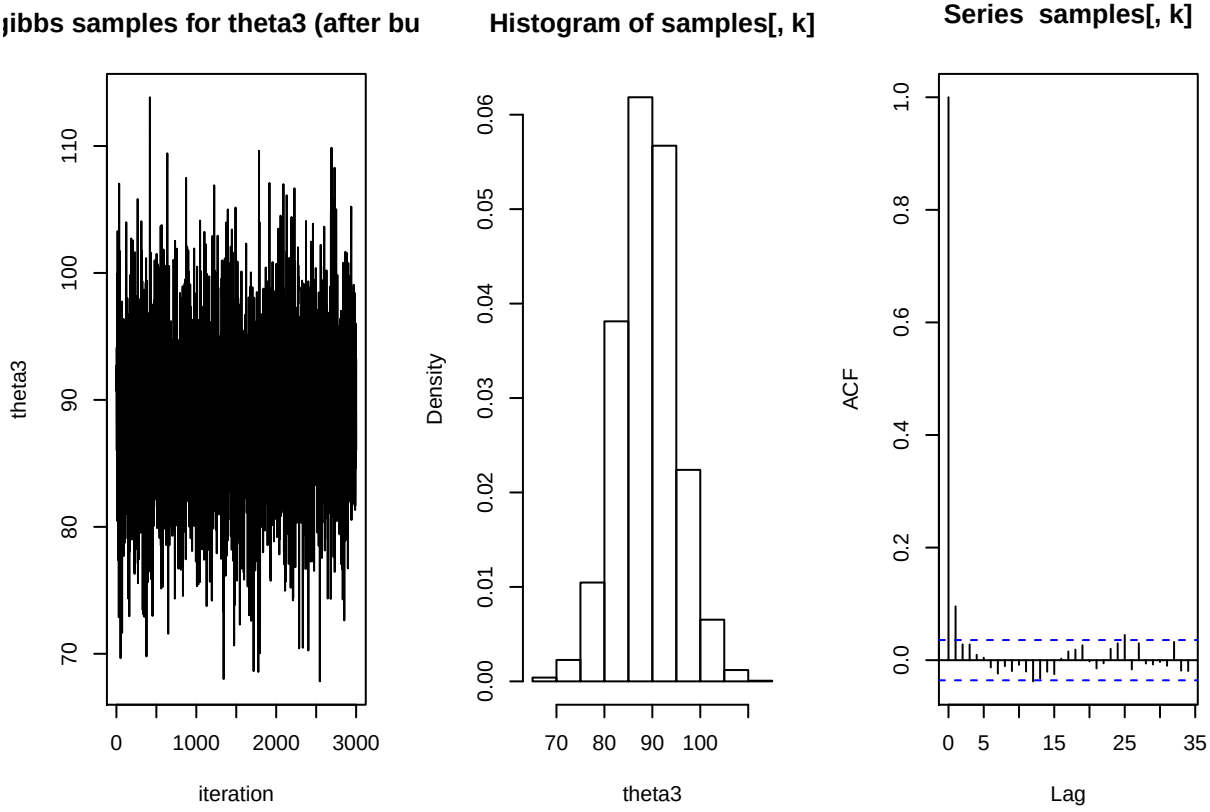
```r
sample_theta = function(mu, tau2, sigma2, Y){
  n_j = ncol(Y)
  J = nrow(Y)
  v = 1/((1/tau2)+(n_j/sigma2))
  sd = sqrt(v)
  m = ((mu/tau2)+(rowSums(Y)/sigma2))*v
  thetas = rnorm(J,m,sd)
  return(thetas)
}

sample_mu = function(theta,tau2,Y){
  n_j = ncol(Y)
  J = nrow(Y)
  m = sum(theta)/J
  sd = sqrt(tau2/J)
  mu_new = rnorm(1,m,sd)
  return(mu_new)
}

sample_sigma2 = function(theta, Y){
  n = length(Y)
  m = n
  v = sum((Y-theta)^2)/n
  sigma2_new = rinvgamma(1,m/2,v*n/2)
  return(sigma2_new)
}

sample_tau2 = function(theta,mu,Y){
  J = nrow(Y)
  m = J-1
  v = sum((theta-mu)^2)/m
  tau2_new = rinvgamma(1,m/2,(m*v)/2)
  return(tau2_new)
}

burn_in = 0.3
M = ceiling(3000/(1-burn_in))

#initial values
theta = rowMeans(y)
mu = mean(rowMeans(y))
sigma2 = sum((y-theta)^2)/length(y)
tau2 = sum((theta-mu)^2)/(nrow(y)-1)

gibbs_samples = matrix(NA, nrow = 9, ncol = M)
rownames(gibbs_samples) = c(sapply(1:6, function(i) paste('theta',i, sep = '')), 'mu','sigma2','tau2')
gibbs_samples[,1] = c(theta,mu,sigma2,tau2)

for (i in 2:M){
  theta = sample_theta(mu, tau2, sigma2, y)
  mu = sample_mu(theta,tau2,y)
  sigma2 = sample_sigma2(theta, y)
```

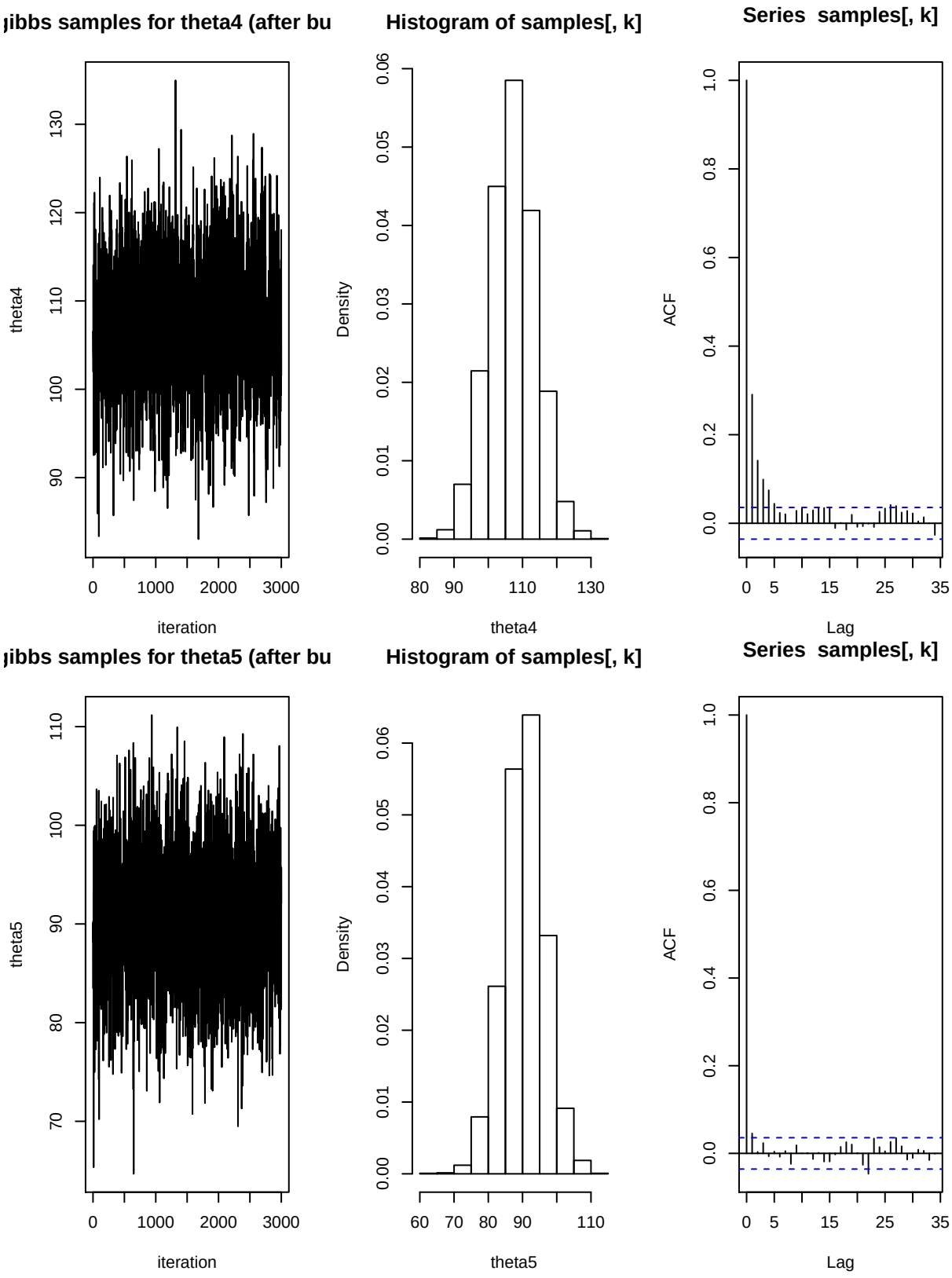
```
  tau2 = sample_tau2(theta,mu,y)
  gibbs_samples[,i] = c(theta,mu,sigma2,tau2)
}

#discard burn in
samples = t(gibbs_samples[,ceiling(M*burn_in):M])
```

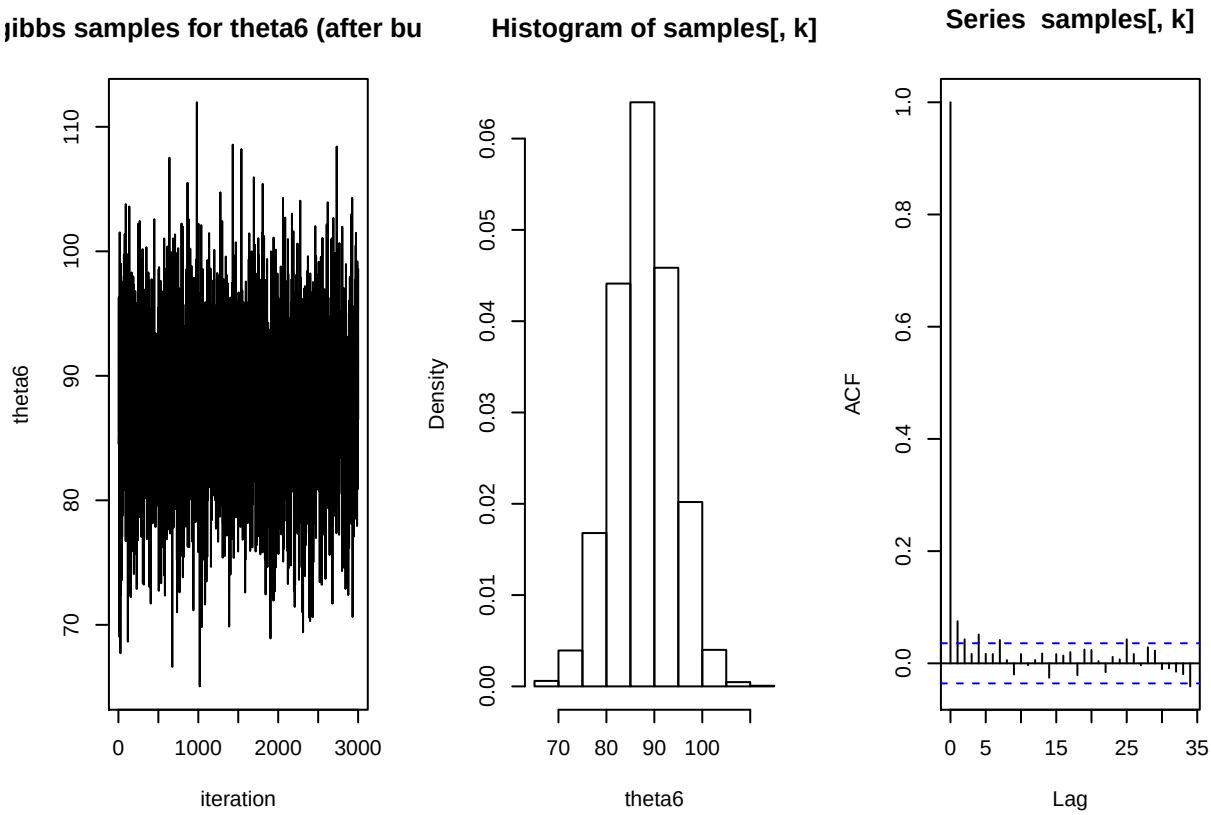
```
for(k in 1:9){
  par(mfrow = c(1,3))
  plot(samples[,k], type = 'l', xlab = "iteration",ylab=colnames(samples)[k], main = paste("gibbs sample
  hist(samples[,k], xlab=colnames(samples)[k], freq = FALSE)
  lines(seq(-3, 3, length.out = 100), dnorm(seq(-3, 3, length.out = 100)), col = "red")
  acf(samples[,k])
}
```

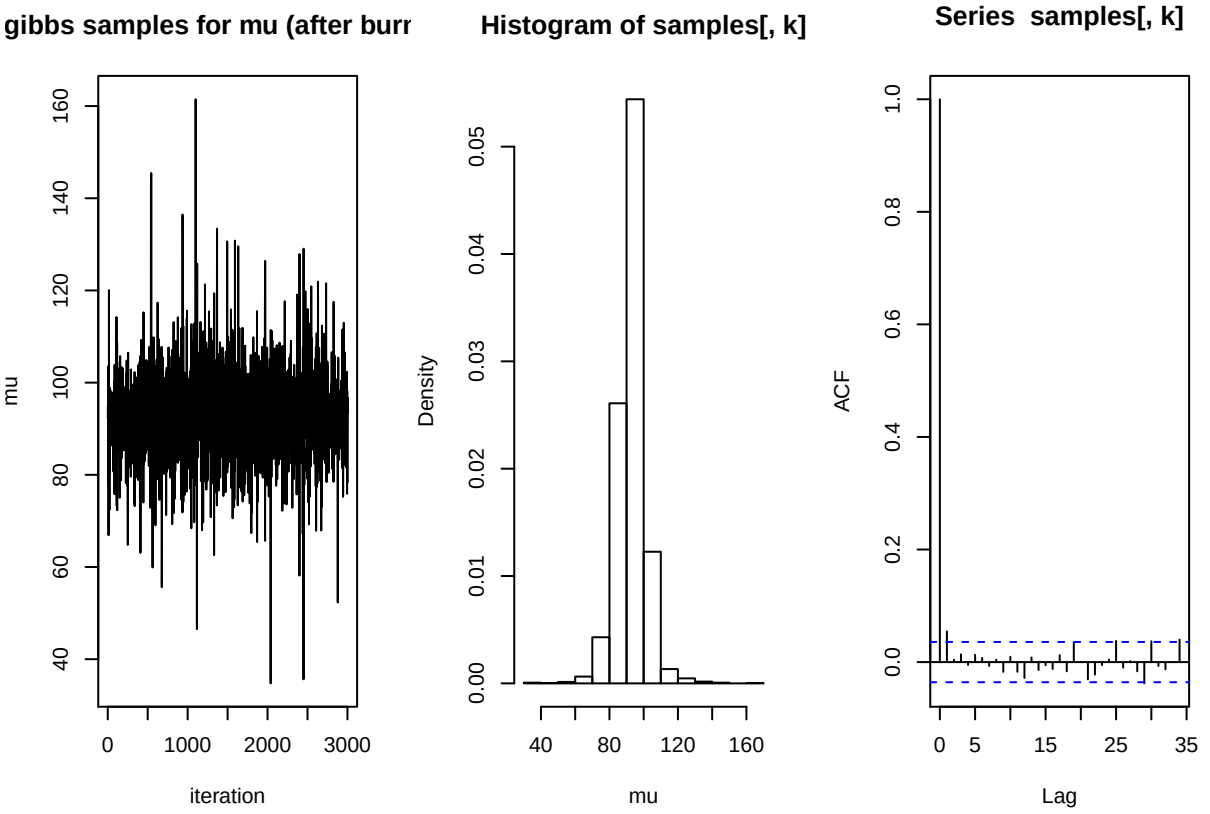


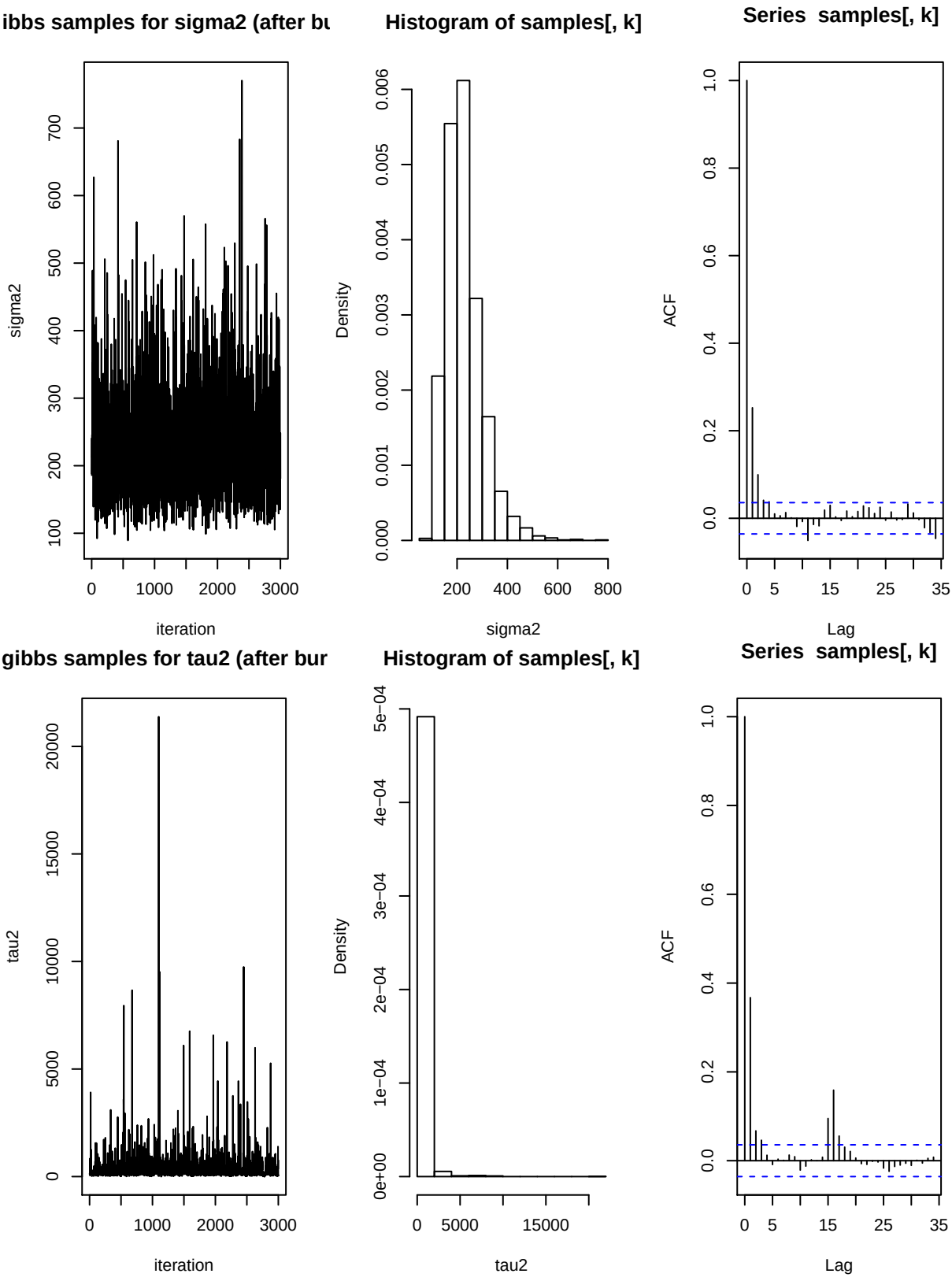

gibbs samples for theta2 (after bu
theta2
iteration
Histogram of samples[, k]
Density
theta2
Series samples[, k]
ACF
Lag


gibbs samples for theta3 (after bu
theta3
iteration
Histogram of samples[, k]
Density
theta3
Series samples[, k]
ACF
Lag


gibbs samples for theta4 (after bu
theta4
iteration
Histogram of samples[, k]
Density
theta4
Series samples[, k]
ACF
Lag
gibbs samples for theta5 (after bu
theta5
iteration
Histogram of samples[, k]
Density
theta5
Series samples[, k]
ACF
Lag


gibbs samples for theta6 (after bu
theta6
iteration
Histogram of samples[, k]
Density
theta6
Series samples[, k]
ACF
Lag


gibbs samples for mu (after burr
mu
iteration
Histogram of samples[, k]
Density
mu
Series samples[, k]
ACF
Lag


ibbs samples for sigma2 (after bu
sigma2
iteration
Histogram of samples[, k]
Density
sigma2
Series samples[, k]
ACF
Lag
gibbs samples for tau2 (after bur
tau2
iteration
Histogram of samples[, k]
Density
tau2
Series samples[, k]
ACF
Lag

(c) Compute the posterior expectation of the mean of the quality measurements of the sixth machine.

```
mean_theta6 = mean(samples[,'theta6'])
mean_theta6
```

```
## [1] 87.70101
```

(d) Suppose a new measurement is taken for the sixth machine. Give a 95%posterior predictive interval for this new measurement.

```
sigma2 = samples[,'sigma2']
theta6 = samples[,'theta6']
new_measurement = sapply(1:length(theta6), function(x){rnorm(1,theta6[x],sqrt(sigma2[x]))})
ci = quantile(new_measurement,c(0.025,0.975))
ci
```

```
##      2.5%     97.5%
##  56.28133 118.49957
```

(e) Give a 95% posterior interval for the mean of the quality control measurements of the seventh machine.

```
mu = samples[,'mu']
tau2 = samples[,'tau2']
theta_7 = sapply(1:length(theta6), function(x){rnorm(1,mu[x],sqrt(tau2[x]))})
ci_7 = quantile(theta_7,c(0.025,0.975))
ci_7
```

```
##      2.5%     97.5%
##  51.33601 132.88945
```

(f) Is it reasonable to assume a model with common variance for this data?Why or why not?

It is not reasonable to assune a model with common variance because the variance for tau2 and sigma2 are very large in this case.

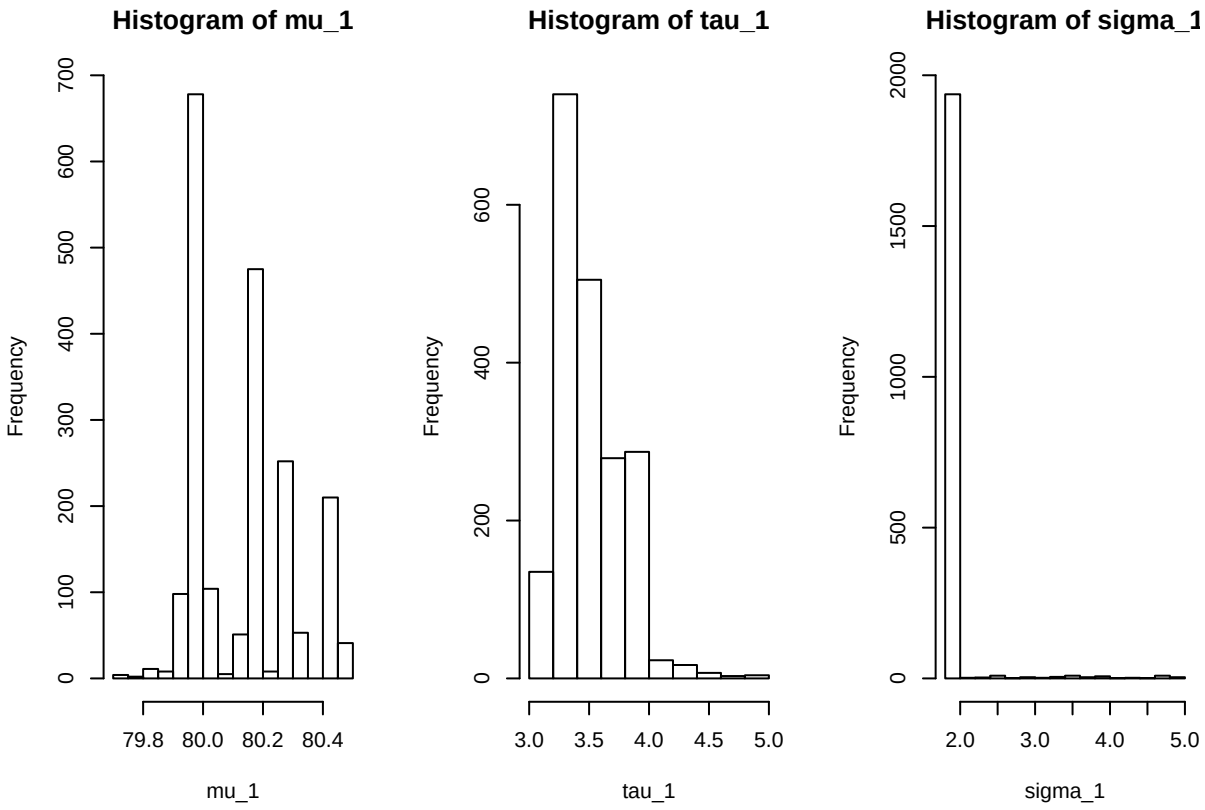
3 (b)

Niter = 2000, start value 80, 5, 5

```
post_density = function(mu, logsigma, logtau){
  sigma2 = (exp(logsigma))^2
  tau2 = (exp(logtau))^2
  y_sum = apply(y,1,sum)
  y_sq_sum = apply(y,1, function(x) (sum(x))^2)

  f1 = exp(-0.5*(mu^2/tau2 + y_sq_sum/sigma2))
  f2 = exp(0.5*(mu/tau2 + y_sum/sigma2)^2 / (1/tau2 + 5/sigma2))* sqrt(2*pi/(1/tau2 + 5/sigma2))

  fx = 1/(sigma2^15 * tau2^2.5) * sum(f1,f2)
  return (fx)
}
```

Variance = 0.01

```
mu_1 = rep(0,2000)
sigma_1 = rep(0,2000)
tau_1 = rep(0,2000)
mu_1[1] = 80
sigma_1[1] = 5
tau_1[1] = 5

for (i in 1:(1999)){
  mu_c = rnorm(1, mu_1[i] , sqrt(0.01))
  s_c = rnorm(1, sigma_1[i] , sqrt(0.01))
  t_c = rnorm(1,tau_1[i] , sqrt(0.01))

  fx_c = post_density(mu_c, s_c, t_c)

  fx_i = post_density(mu_1[i], sigma_1[i], tau_1[i])

  if (fx_c == Inf){
    fx_c = 0
  }
  if (fx_i == Inf){
    fx_i = 0
  }
  ratio = fx_c/fx_i

  alpha = min(1, ifelse(fx_i>0,ratio,0))

  a = runif(1)

  if(a<alpha){
    mu_1[i+1] = mu_c
    sigma_1[i+1] = s_c
    tau_1[i+1] = t_c
  }else{
    mu_1[i+1] = mu_1[i]
    sigma_1[i+1] = sigma_1[i]
    tau_1[i+1] = tau_1[i]
  }

}
par(mfrow = c(1,3))
hist(mu_1)
hist(tau_1)
hist(sigma_1)
```



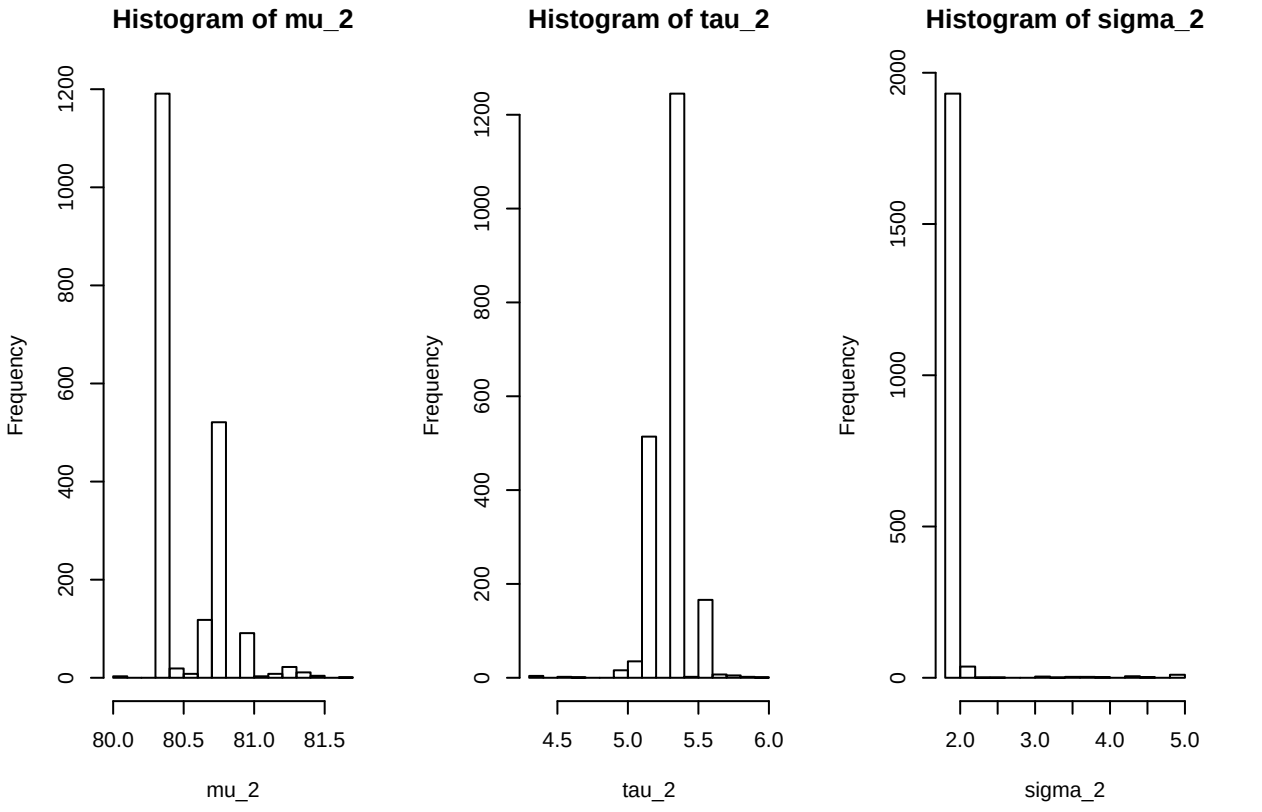

```
effsize = c(effectiveSize(mcmc.list(as.mcmc(mu_1))),effectiveSize(mcmc.list(as.mcmc(tau_1))),effectiveS
effsize
```

```
##      var1      var1      var1
##  4.423558  3.255376 22.979357
```

Variance = 0.1

```
mu_2 = rep(0,2000)
sigma_2 = rep(0,2000)
tau_2 = rep(0,2000)
mu_2[1] = 80
sigma_2[1] = 5
tau_2[1] = 5

for (i in 1:(1999)){
  mu_c = rnorm(1, mu_2[i] , sqrt(0.1))
  s_c = rnorm(1, sigma_2[i] , sqrt(0.1))
  t_c = rnorm(1,tau_2[i] , sqrt(0.1))

  fx_c = post_density(mu_c, s_c, t_c)

  fx_i = post_density(mu_2[i], sigma_2[i], tau_2[i])

  if (fx_c == Inf){
    fx_c = 0
  }
  if (fx_i == Inf){
```

```
    fx_i = 0
  }
  ratio = fx_c/fx_i

  alpha = min(1, ifelse(fx_i>0,ratio,0))

  a = runif(1)

  if(a<alpha){
    mu_2[i+1] = mu_c
    sigma_2[i+1] = s_c
    tau_2[i+1] = t_c
  }else{
    mu_2[i+1] = mu_2[i]
    sigma_2[i+1] = sigma_2[i]
    tau_2[i+1] = tau_2[i]
  }

}
par(mfrow = c(1,3))
hist(mu_2)
hist(tau_2)
hist(sigma_2)
```



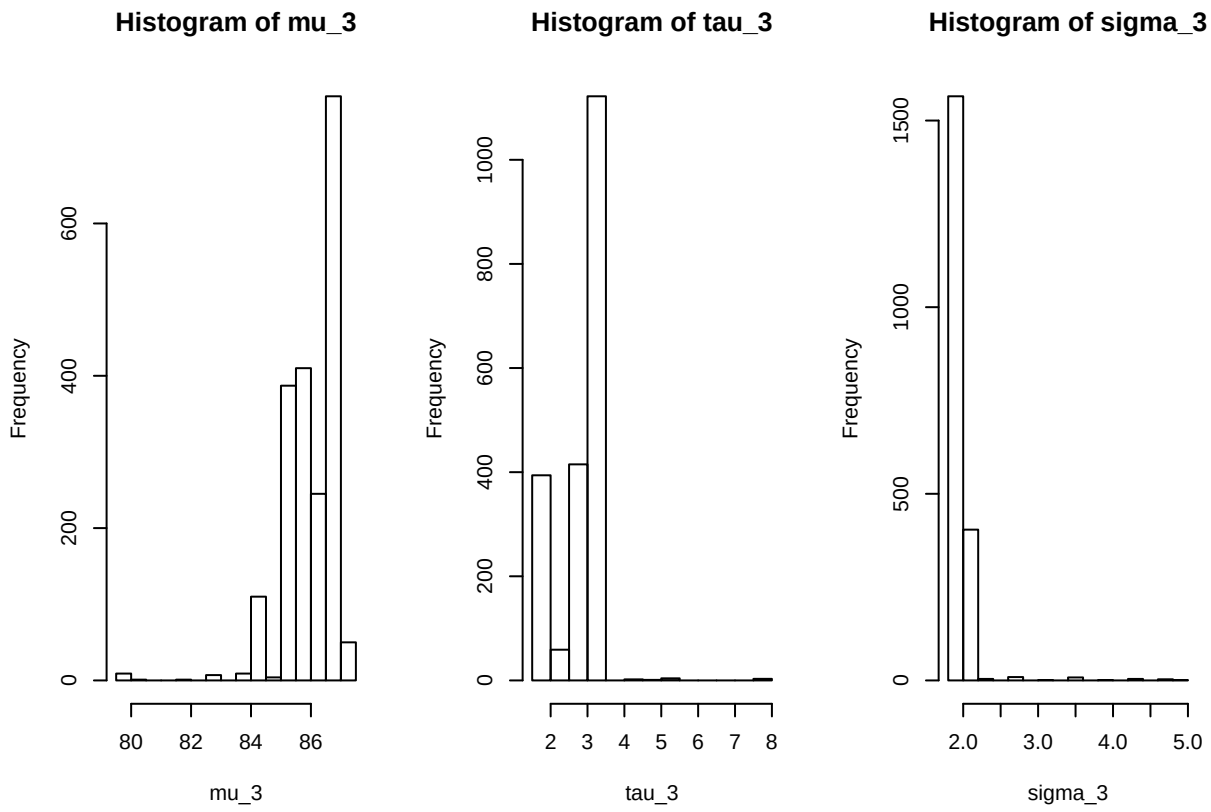

```
effsize_2 = c(effectiveSize(mcmc.list(as.mcmc(mu_2))),effectiveSize(mcmc.list(as.mcmc(tau_2))),effectiv
effsize_2
```

```
##      var1      var1      var1
```

```
##  7.080729 14.803983 50.739825
```

Variance = 1

```
mu_3 = rep(0,2000)
sigma_3 = rep(0,2000)
tau_3 = rep(0,2000)
mu_3[1] = 80
sigma_3[1] = 5
tau_3[1] = 5

for (i in 1:1999){
  mu_c = rnorm(1, mu_3[i] , 1)
  s_c = rnorm(1, sigma_3[i] , 1)
  t_c = rnorm(1,tau_3[i] , 1)

  fx_c = post_density(mu_c, s_c, t_c)
  #print(fx_c)
  fx_i = post_density(mu_3[i], sigma_3[i], tau_3[i])
  #print(fx_i)
  if (fx_c == Inf){
    fx_c = 0
  }
  if (fx_i == Inf){
    fx_i = 0
  }
  ratio = fx_c/fx_i

  alpha = min(1, ifelse(fx_i>0,ratio,0))

  if(runif(1)<alpha){
    mu_3[i+1] = mu_c
    sigma_3[i+1] = s_c
    tau_3[i+1] = t_c
  }else{
    mu_3[i+1] = mu_3[i]
    sigma_3[i+1] = sigma_3[i]
    tau_3[i+1] = tau_3[i]
  }

}
par(mfrow = c(1,3))
hist(mu_3)
hist(tau_3)
hist(sigma_3)
```




```
effsize_3 = c(effectiveSize(mcmc.list(as.mcmc(mu_3))),effectiveSize(mcmc.list(as.mcmc(tau_3))),effective
effsize_3
```

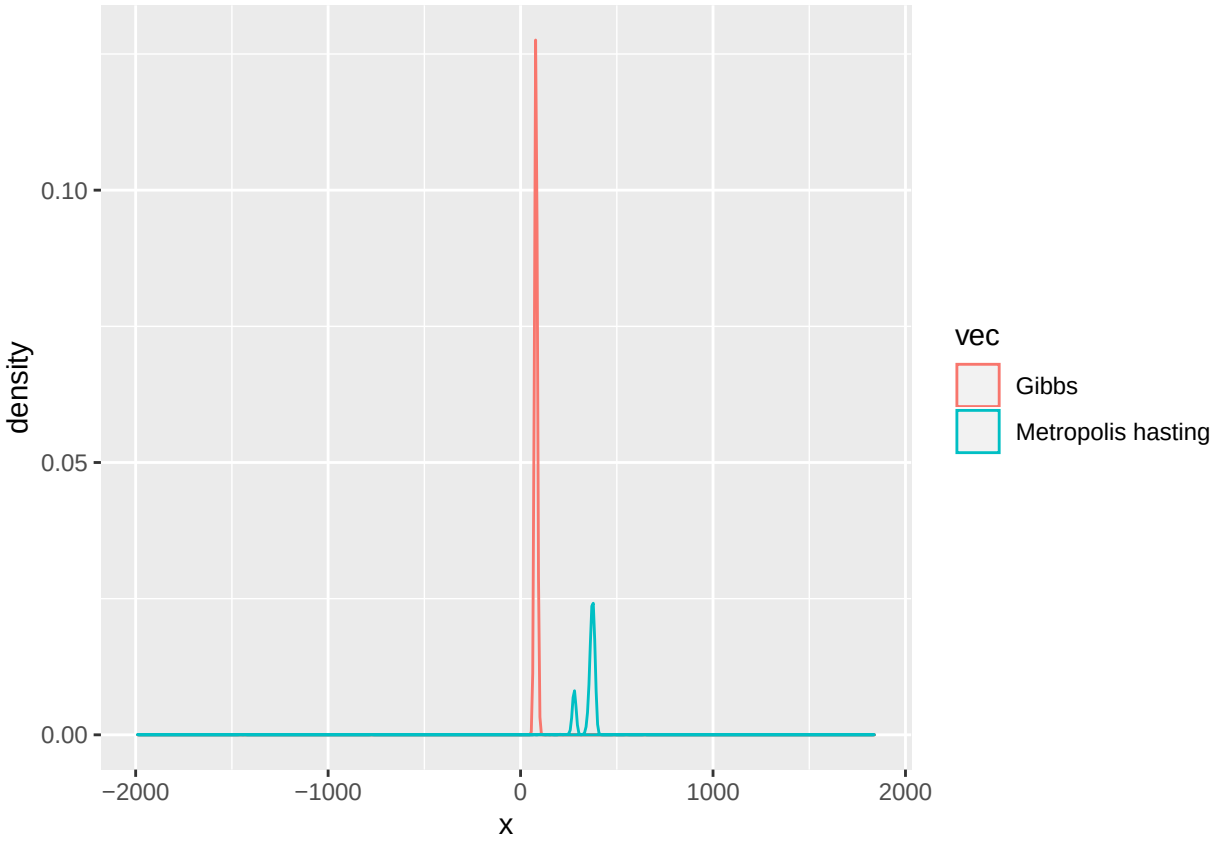
```
##      var1      var1      var1
##  7.794899  3.862845 59.037959
```

3(c)

```
theta_new = function(mu, log_sigma, log_tau){
  sigma2 = (exp(log_sigma))^2
  tau2 = (exp(log_tau))^2

  theta = apply(y,1, function(x) rnorm(1, (mu/tau2 +5*sum(x)/sigma2)/(1/tau2 + 5/sigma2), 1/(1/tau2 + 5/

}

mh_samples_1 = matrix(NA, nrow = 6, ncol = 2000)
mh_samples_2 = matrix(NA, nrow = 6, ncol = 2000)
mh_samples_3 = matrix(NA, nrow = 6, ncol = 2000)
for (i in 1:2000){
  mh_samples_1[,i] = theta_new(mu_1[i], sigma_1[i], tau_1[i])
  mh_samples_2[,i] = theta_new(mu_2[i], sigma_2[i], tau_2[i])
  mh_samples_3[,i] = theta_new(mu_3[i], sigma_3[i], tau_3[i])
}
theta_mh_1 = t(mh_samples_1)
theta_mh_2 = t(mh_samples_2)
theta_mh_3= t(mh_samples_3)
```

```r
df <- rbind(data.frame(x=samples[,1], vec='Gibbs'),
            data.frame(x=theta_mh_3[,1], vec='Metropolis hasting'))
ggplot(df, aes(x, group=vec, col=vec)) + geom_density(position='dodge')
```

```
## Warning: Width not defined. Set with `position_dodge(width = ?)`
```



Gibbs sample has a smaller variance than metroplois hasting sample.