Bayesian_Final_Project

STAT 357

Zeqiu.Yu

2023-03-14

Question 1

a.

1.

a. The question is : $X_1, X_2, X_3, X_4 = (125, 18, 20, 34)$ with prob $= (\frac{1}{2}+\frac{\theta}{4}, \frac{1}{4}(1-\theta), \frac{1}{4}(1-\theta), \frac{\theta}{4})$

Then the augmented data are given by $(y_1, y_2, y_3, y_4, y_5)$ with $y_1+y_2=125$.

Under a flat prior, the posterior $p(\theta|Y) \propto (2+\theta)^{y_1}(1-\theta)^{y_2+y_3}\theta^{y_4}$ $\quad Y_2$

$$p(\theta|Y,z) \propto \theta^{y_2+y_5}(1-\theta)^{y_3+y_4} \sim \text{Beta}, \quad p(\overset{(y_2)}{z}|Y,\theta^i) = Bi(125, \frac{\theta^i}{\theta^i+2})$$

Given a initialization z^*, then we can sample $\theta^* \sim p(\theta|z^*,Y) \sim \text{Beta}(z^*+y_5+1,\ y_3+y_4+1)$

and then sample $z^* \sim p(z|\theta^*,Y) \sim Bi(125, \frac{\theta^*}{\theta^*+2})$

In this way, we can form a chain and get θ^i as i

We can use Gelman&Rubin method to access convergence of each initialization. For a given num of chains M, each with N samples.

$\mu_j = \frac{1}{N}\sum\theta_j$ $\quad s_j^2 = \frac{1}{N-1}\sum(\theta_j-\mu_j)^2$, $j=1,\cdots M$, θ_j is the samples from chain j.

Overall mean and variance $\mu = \frac{1}{MN}\sum_{j=1}^{M}(\sum\theta_j)$ $\quad s^2 = \frac{1}{MN-1}\sum(\sum(\theta_j-\mu)^2)$

$B = \frac{N}{M-1}\sum_{j=1}^{M}(\mu_j-\mu)^2$

$W = \frac{1}{M}\cdot\sum s_j^2$

$\hat{V} = \frac{N-1}{N}\cdot W + \frac{1}{N}B$

$\hat{R} = \sqrt{\frac{\hat{V}}{W}}$

According to the simulation in Python, I find the posterior mean is : 0.623 , the variance is: 0.0026

(I take $M=1000$ and $N=10000$)

I don't toss out any initial values according to Gelman& Rubin Methods. I find they all have the $\hat{R}$ that is close to 1 and less than 1.1, which means the chains converge for these initial values.

For this example, I take $z_0=0$ and get $\hat{R}=1.0044$

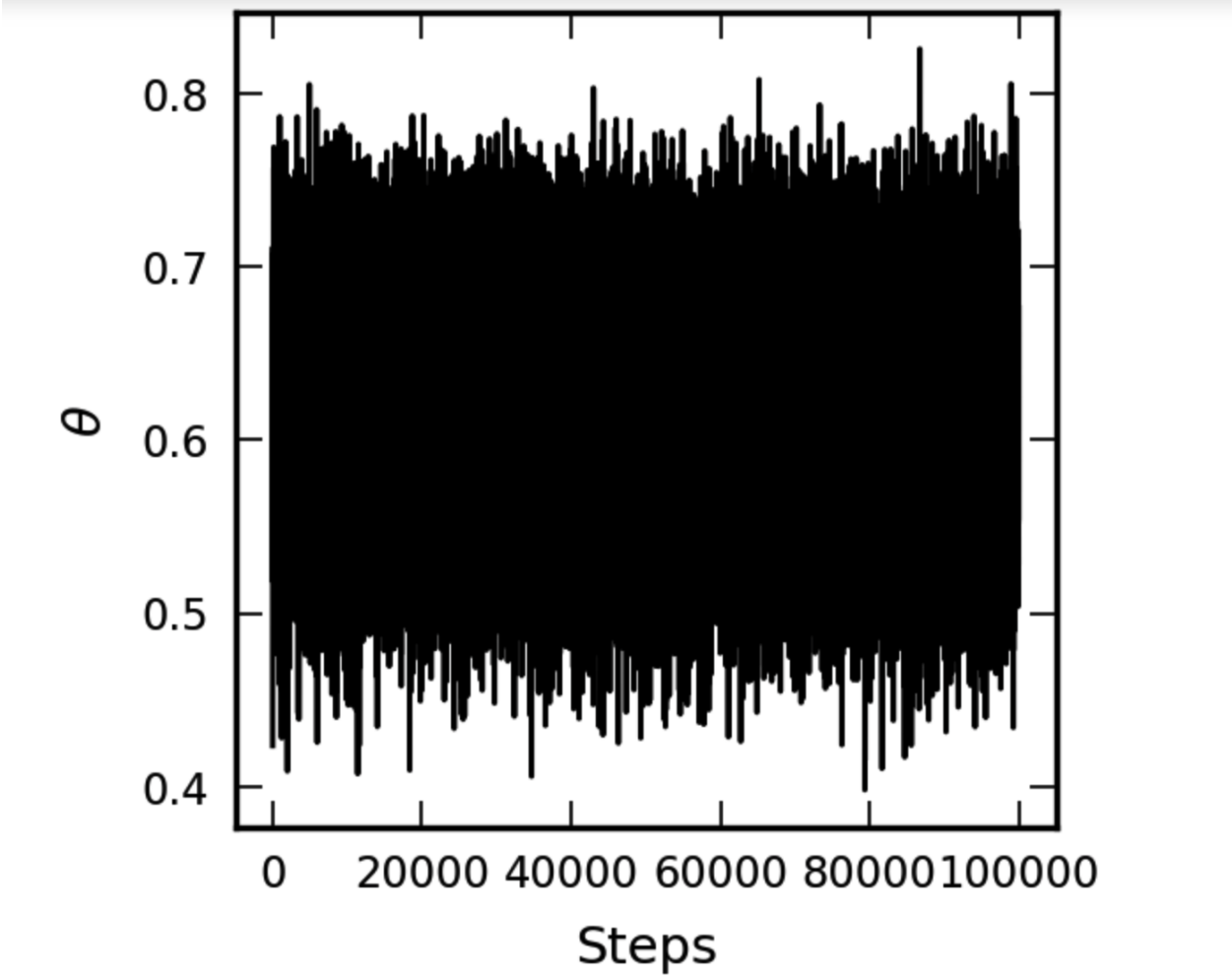
For the likelihood $p(\theta|Y)$ and normalized likelihood below.

(For the emergent environmental problem, I have to move the codes to R from Python).

```
# Give the initialization and iterNum M and sampleSz N
M = 1000
N = 10000
thetas = []
thetaMs = []
thetaSVs = []


z0Value = 0
z0 = np.array([z0Value for i in range(N)])
x1, x2, x3, x4 = (125, 18, 20, 34)
y1, y2, y3, y4, y5 = (125-z0Value, z0Value, 18, 20, 34)
# 1. iterations

for _ in range(M):
    z = z0
    thetai = np.random.beta(z0 + y5 + 1, y3 + y4 + 1, N)
    z0 = np.random.binomial(125, thetai/(2 + thetai), N)
    Muj = np.mean(thetai)
    SV = 1/(N-1)*np.sum((thetai - Muj)**2)
    thetaMs.append(Muj)
    thetaSVs.append(SV)
    thetas.append(thetai.tolist())

# Calculate Gelman-Rubin:
mu = 1/(M*N)*np.sum(thetas)
SV = 1/(M*N-1)*np.sum((N-1)*np.array(thetaSVs))
B = N/(M-1)*sum([(muj - mu)**2 for muj in thetaMs])
W = 1/M*sum(thetaSVs)
V_hat = (N-1)/N*W + 1/N*B
R_hat = np.sqrt(V_hat/W)
print(R_hat)
```

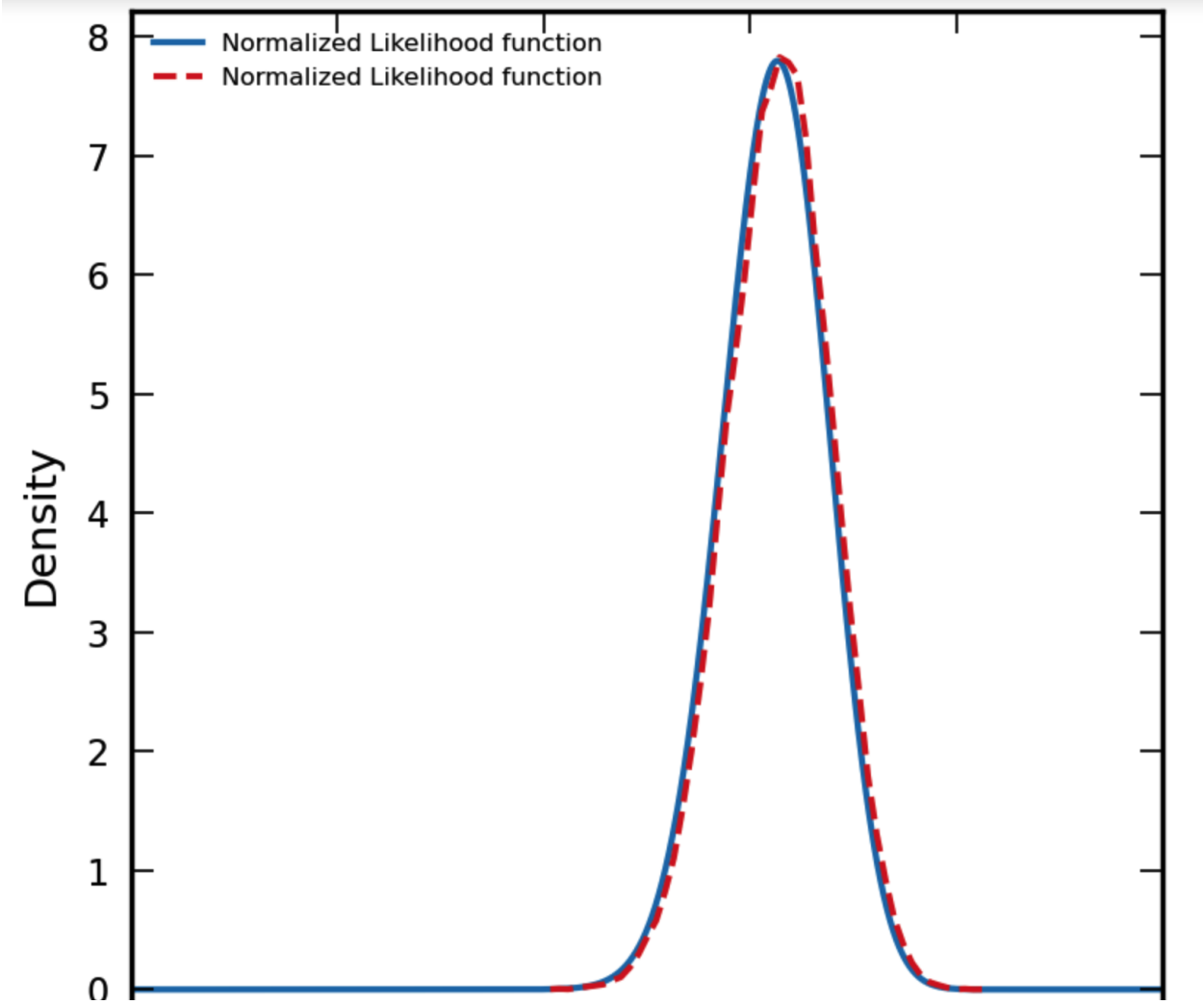
```
1.0044005147977342
```

```
np.mean(thetaMs[:]),np.var(thetaMs[:])

# Give the initialization and iterNum M and sampleSz N
M = 100000
N = 1
thetas = []
thetaMs = []

z0Value = 0
z0 = np.array([z0Value for i in range(N)])
x1, x2, x3, x4 = (125, 18, 20, 34)
y1, y2, y3, y4, y5 = (125-z0Value, z0Value, 18, 20, 34)
# 1. iterations

for _ in range(M):
    z = z0
    thetai = np.random.beta(z0 + y5 + 1, y3 + y4 + 1, N)
    z0 = np.random.binomial(125, thetai/(2 + thetai), N)
    Muj = np.mean(thetai)
    thetaMs.append(Muj)
    thetas.append(thetai.tolist())

f, ax = plt.subplots(1, 1, figsize=(2, 2), facecolor='white', dpi=300, gridspec_kw={'hspace': 0., 'wspace': 0.})
ax.plot(np.arange(M), thetaMs, color = "black",ls = '-', lw=.7, zorder = 1)

ax.tick_params(axis='both', which='both', labelsize='xx-small', right=True, top=True, direction='in', width=.4)
ax.set_xlabel("Steps", size='x-small')
ax.set_ylabel(r"$\theta$", size='x-small')
plt.show()

np.mean(thetaMs), np.var(thetaMs)
```




```
(0.6230044821354126, 0.002571009087313842)
```

```
def likelihood(theta, n):
    n1, n2, n3, n4 = n
    return (2+theta)**n1 * (1-theta)**n2 * (1-theta)**n3 * (theta)**n4
n = (125, 18, 20, 34)
theta = np.linspace(0, 1, 1000000)
L = likelihood(theta, n)
I = integrate.quad(likelihood, 0, 1, args=[n[i] for i in range(4)])
L_norm = L/(I[0] - I[1])

f, ax = plt.subplots(1, 1, figsize=(3, 3), facecolor='white', dpi=300, gridspec_kw={'hspace': 0., 'wspace': 0.})
ax.plot(theta, L_norm, color = "tab:blue",ls = '-', lw=1, label='Normalized Likelihood function', zorder = 1)
density = np.histogram(thetaMs, bins=50, density=True)
plt.plot(density[1][1:], density[0], color = "tab:red",ls = '--', lw=1, label='Normalized Likelihood function', zorder



ax.tick_params(axis='both', which='both', labelsize='xx-small', right=True, top=True, direction='in', width=.4)
ax.set_xlabel(r"$\theta$", size='x-small')
ax.set_ylabel("Density", size='x-small')
ax.set_xlim([0, 1])
ax.legend(loc = 2 ,fontsize = 4,markerscale = 3,ncol = 1,scatterpoints= 1,frameon = True,framealpha =0.).get_frame().se
plt.show()
```



b.

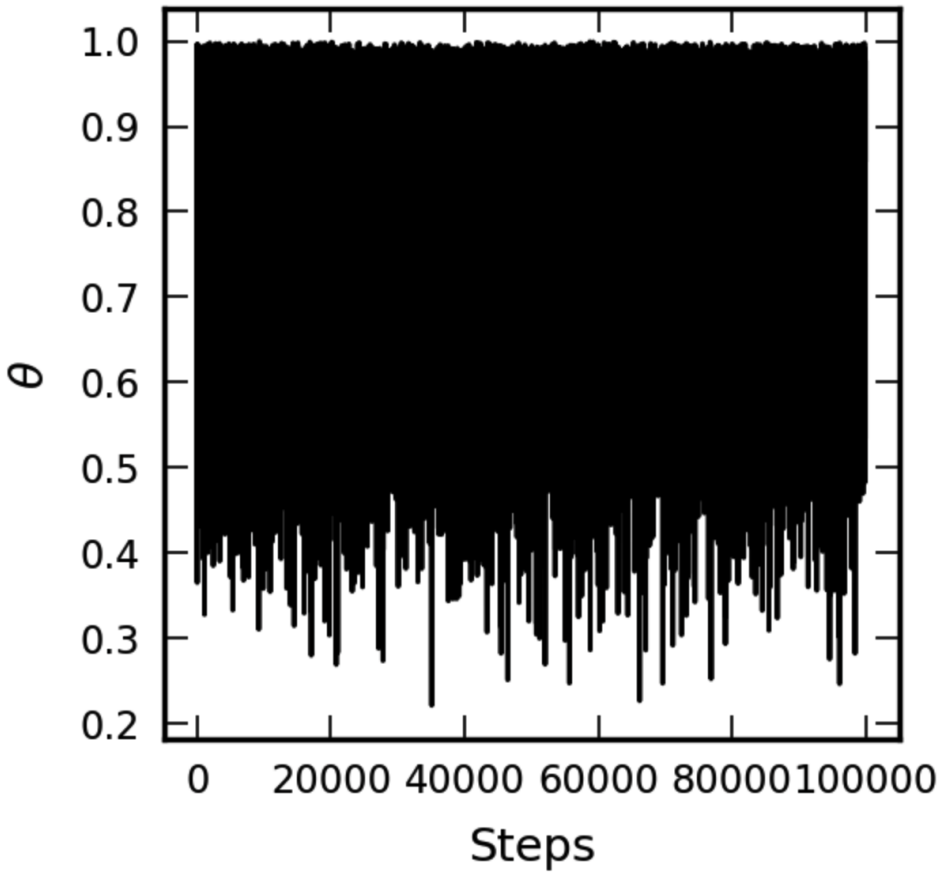
In the same way, for $(x_1, x_2, x_3, x_4) = (14, 0, 1, 5)$ with the same corresponding probability. Repeat the steps above.

```
# Give the initialization and iterNum M and sampleSz N
M = 1000
N = 10000
thetas = []
thetaMs = []
thetaSVs = []

z0Value = 0
z0 = np.array([z0Value for i in range(N)])
x1, x2, x3, x4 = (14, 0, 1, 5)
y1, y2, y3, y4, y5 = (14-z0Value, z0Value, 0, 1, 5)
# 1. iterations

for _ in range(M):
    z = z0
    thetai = np.random.beta(z0 + y5 + 1, y3 + y4 + 1, N)
    z0 = np.random.binomial(14, thetai/(2 + thetai), N)
    Muj = np.mean(thetai)
    SV = 1/(N-1)*np.sum((thetai - Muj)**2)
    thetaMs.append(Muj)
    thetaSVs.append(SV)
    thetas.append(thetai.tolist())

# Calculate Gelman-Rubin:
mu = 1/(M*N)*np.sum(thetas)
SV = 1/(M*N-1)*np.sum((N-1)*np.array(thetaSVs))
B = N/(M-1)*sum([(muj - mu)**2 for muj in thetaMs])
W = 1/M*sum(thetaSVs)
V_hat = (N-1)/N*W + 1/N*B
R_hat = np.sqrt(V_hat/W)
print(R_hat)
```

```
1.0002912285820016
```

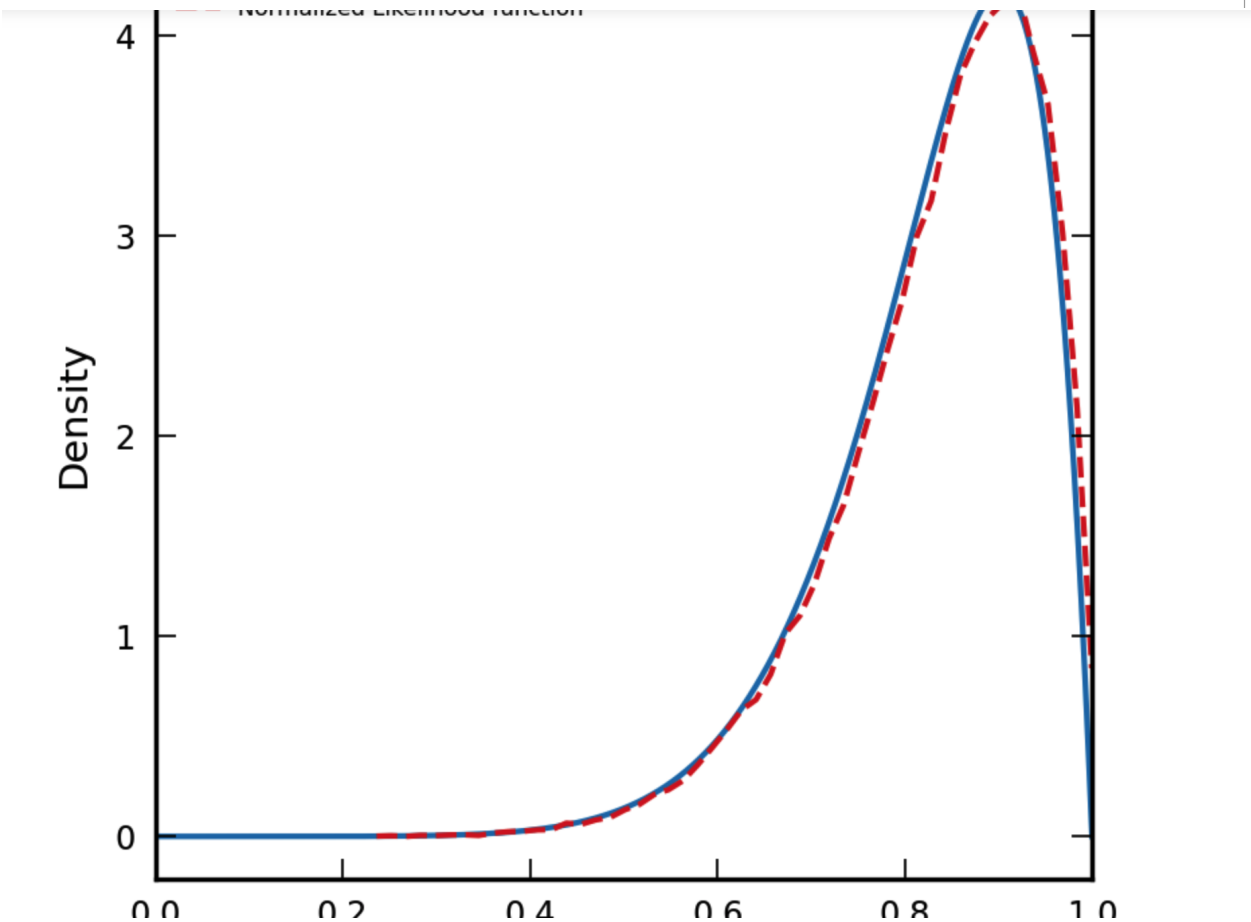
After trying some other initialization, we can still find the similar results with a, which shows the convergence of the chains.

```
np.mean(thetaMs[:]),np.var(thetaMs[:])

# Give the initialization and iterNum M and sampleSz N
M = 100000
N = 1
thetas = []
thetaMs = []

z0Value = 0
z0 = np.array([z0Value for i in range(N)])
x1, x2, x3, x4 = (14, 0, 1, 5)
y1, y2, y3, y4, y5 = (14-z0Value, z0Value, 0, 1, 5)
# 1. iterations

for _ in range(M):
    z = z0
    thetai = np.random.beta(z0 + y5 + 1, y3 + y4 + 1, N)
    z0 = np.random.binomial(14, thetai/(2 + thetai), N)
    Muj = np.mean(thetai)
    thetaMs.append(Muj)
    thetas.append(thetai.tolist())

f, ax = plt.subplots(1, 1, figsize=(2, 2), facecolor='white', dpi=300, gridspec_kw={'hspace': 0., 'wspace': 0.})
ax.plot(np.arange(M), thetaMs, color = "black",ls = '-', lw=.7, zorder = 1)

ax.tick_params(axis='both', which='both', labelsize='xx-small', right=True, top=True, direction='in', width=.4)
ax.set_xlabel("Steps", size='x-small')
ax.set_ylabel(r"$\theta$", size='x-small')
plt.show()

np.mean(thetaMs), np.var(thetaMs)
```




```
(0.8301818704732893, 0.011772538866183763)
```

```
: def likelihood(theta, n):
      n1, n2, n3, n4 = n
      return (2+theta)**n1 * (1-theta)**n2 * (1-theta)**n3 * (theta)**n4
  n = (14, 0, 1, 5)
  theta = np.linspace(0, 1, 1000000)
  L = likelihood(theta, n)
  I = integrate.quad(likelihood, 0, 1, args=[n[i] for i in range(4)])
  L_norm = L/(I[0] - I[1])

  f, ax = plt.subplots(1, 1, figsize=(3, 3), facecolor='white', dpi=300, gridspec_kw={'hspace': 0., 'wspace': 0.})
  ax.plot(theta, L_norm, color = "tab:blue",ls = '-', lw=1, label='Normalized Likelihood function', zorder = 1)
  density = np.histogram(thetaMs, bins=50, density=True)
  plt.plot(density[1][1:], density[0], color = "tab:red",ls = '--', lw=1, label='Normalized Likelihood function', zorder



  ax.tick_params(axis='both', which='both', labelsize='xx-small', right=True, top=True, direction='in', width=.4)
  ax.set_xlabel(r"$\theta$", size='x-small')
  ax.set_ylabel("Density", size='x-small')
  ax.set_xlim([0, 1])
  ax.legend(loc = 2 ,fontsize = 4,markerscale = 3,ncol = 1,scatterpoints= 1,frameon = True,framealpha =0.).get_frame().se
  plt.show()
```



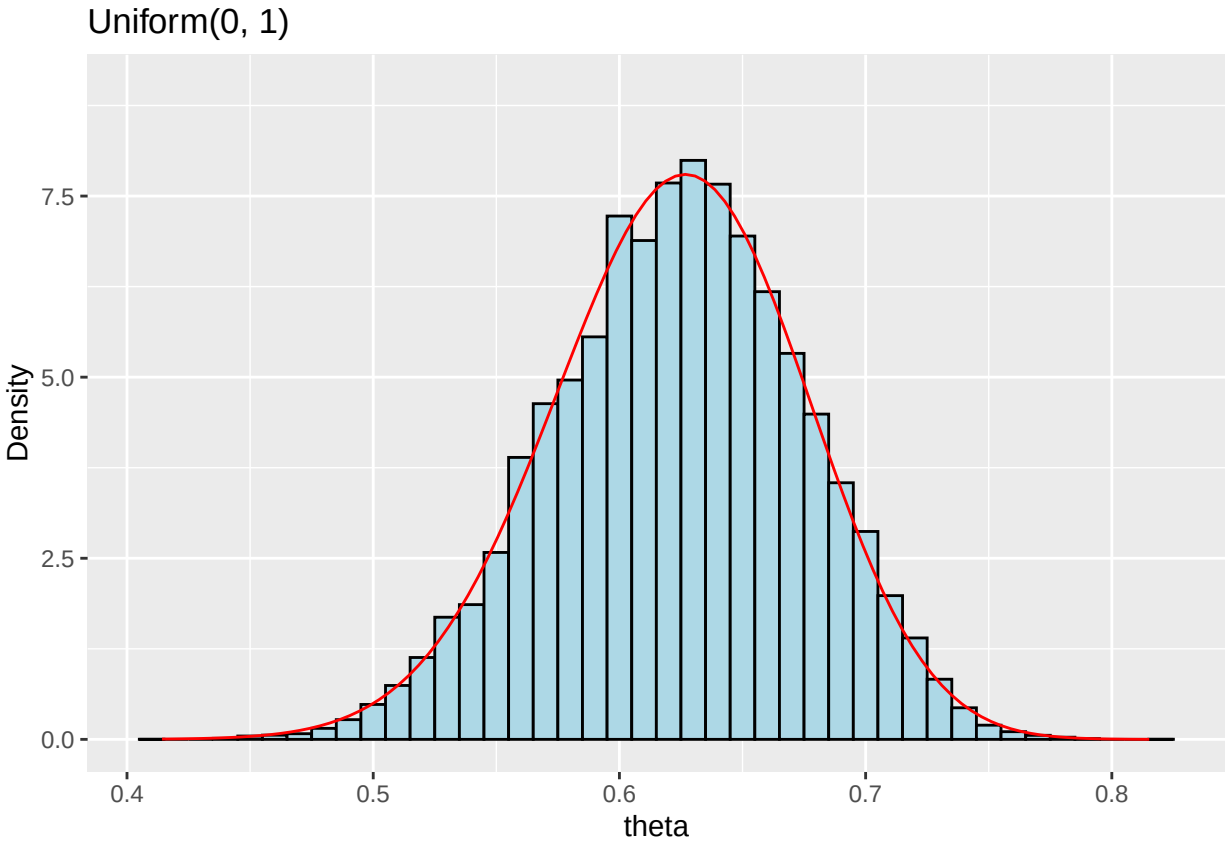
We can use Gibbs sampler to get theta_i and find the convergence traces and use Gelman-Rubin method to check convergence after choosing M and N. We can find both situations will conver in the end and plot the normalized likelihood and simulation density above.

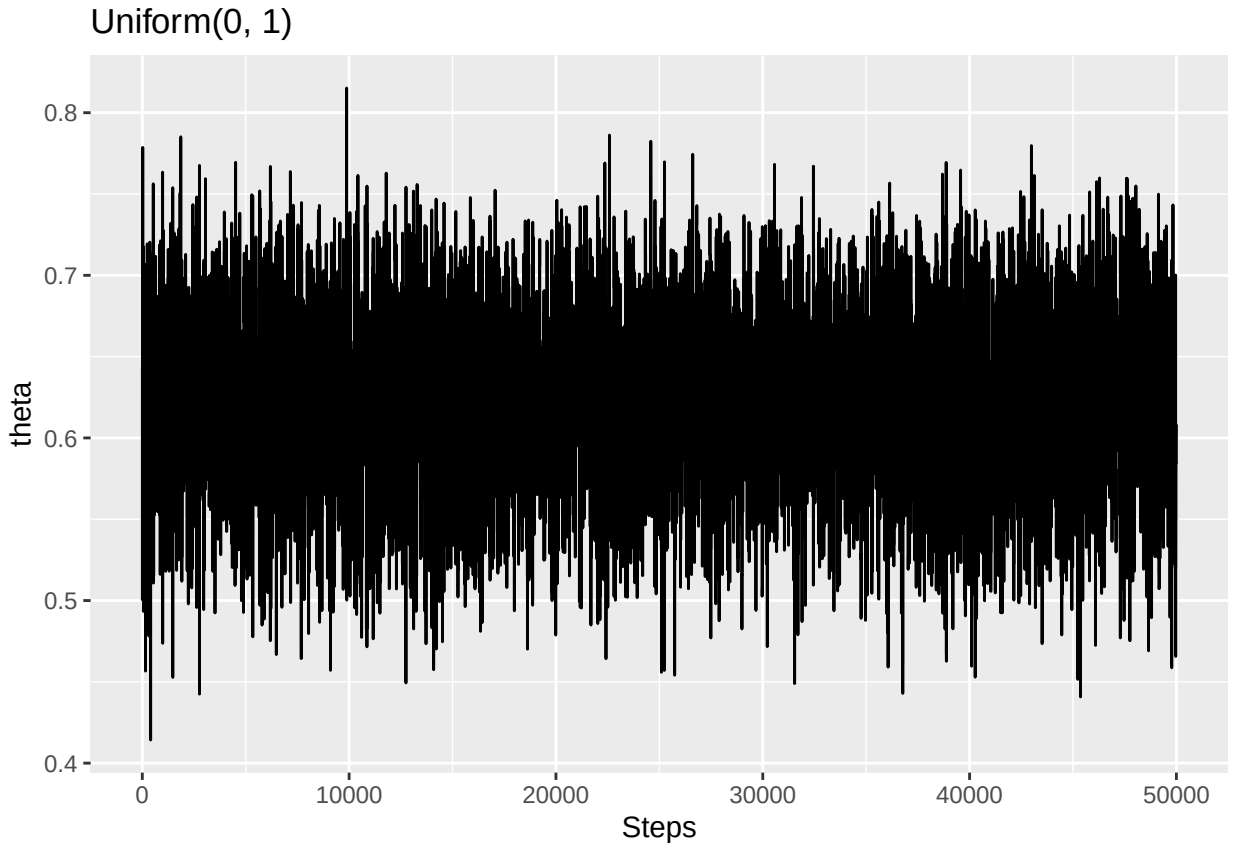
Question2

a

(1)

For the Uniform on (0,1):

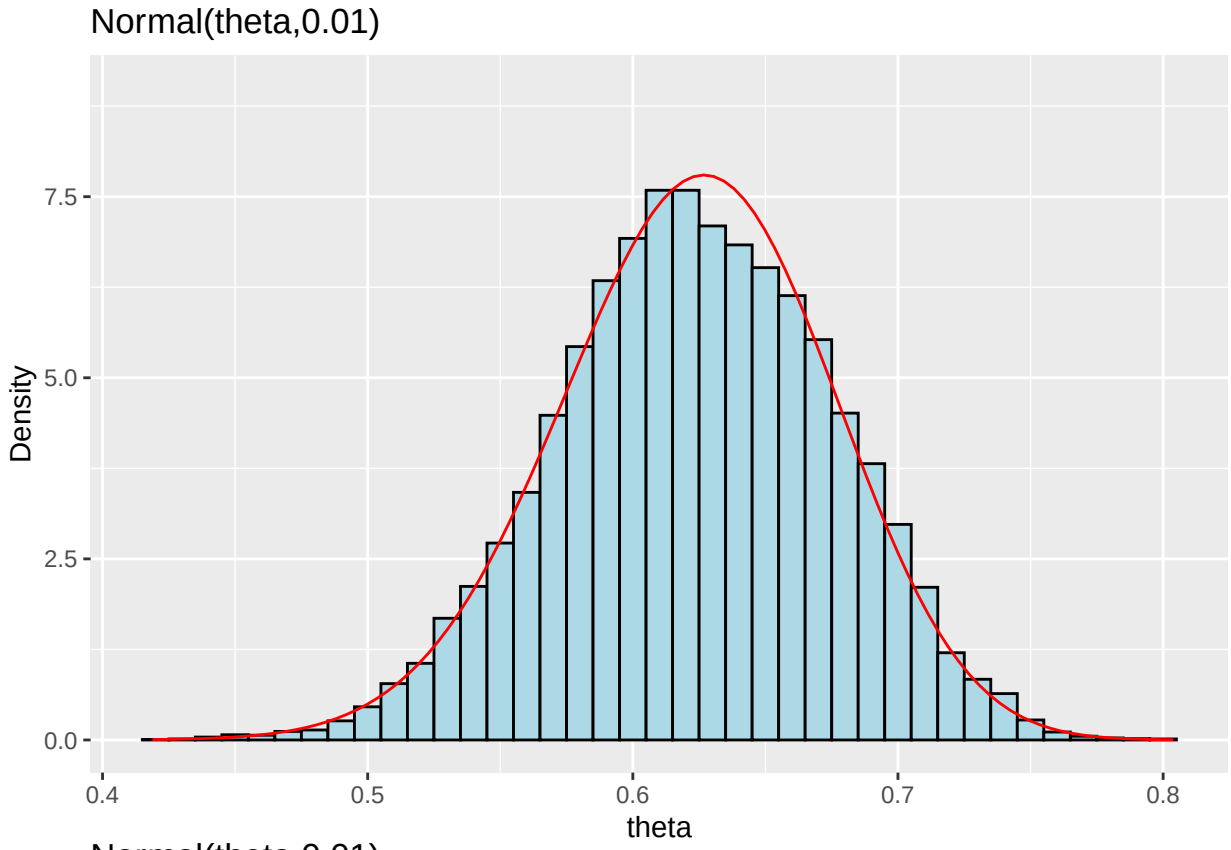


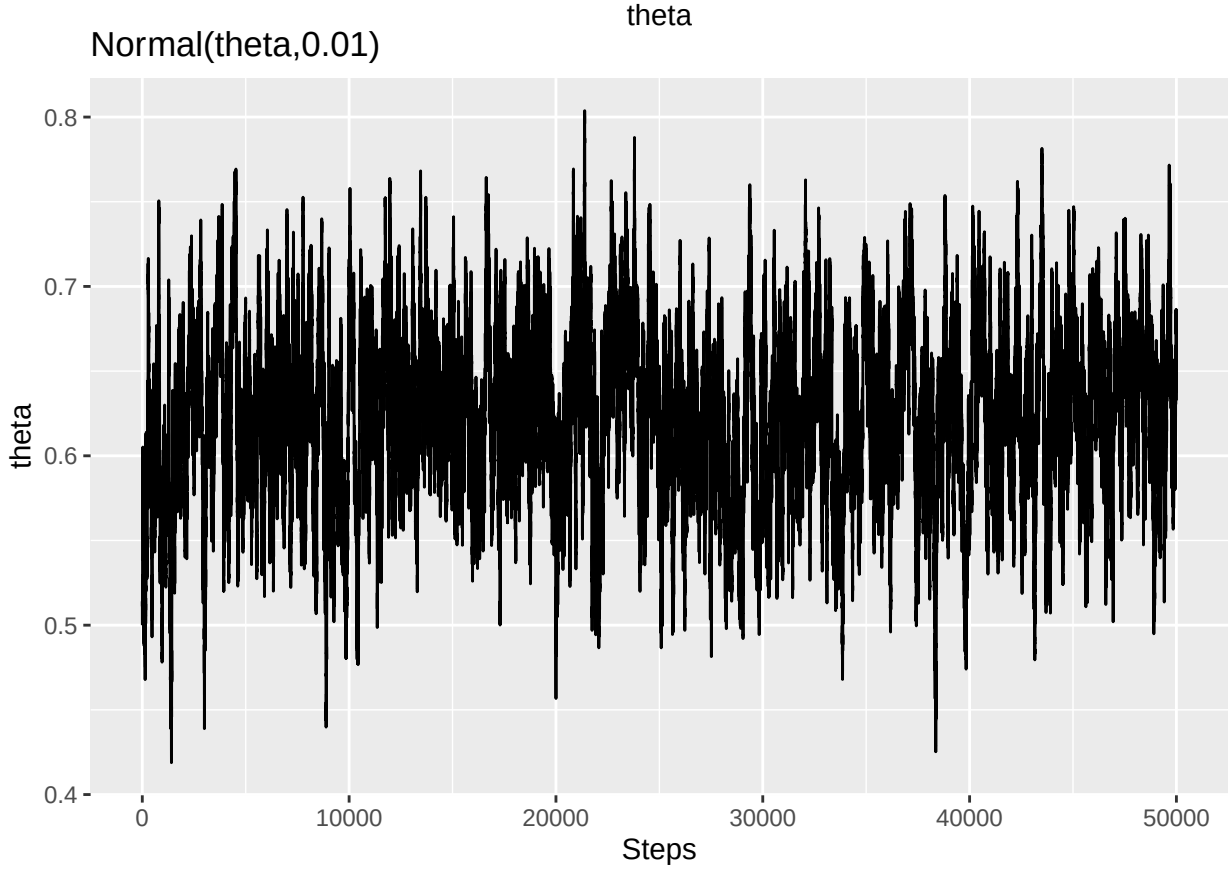



```
## [1] 0.6235314
```

(2).

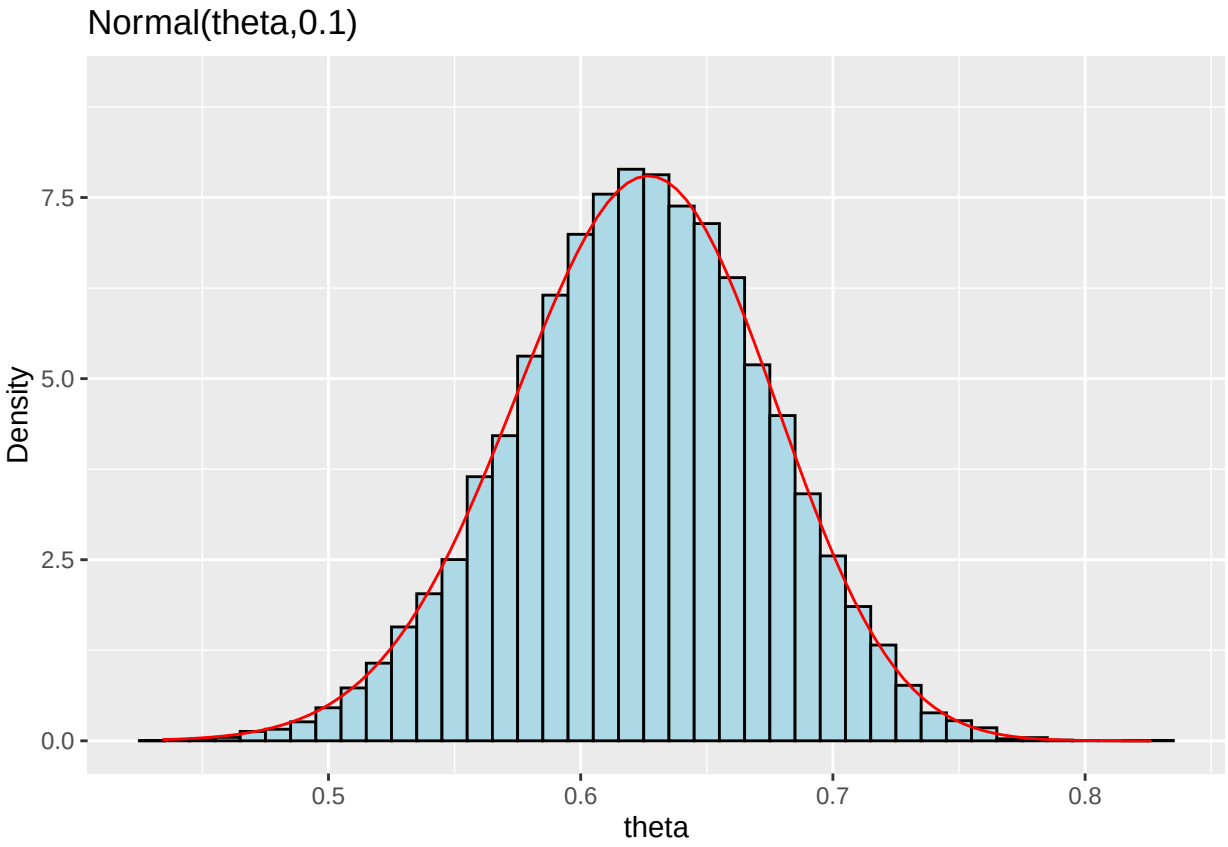
Then for the Normal cenetered at the current point in the chain and standard deviation 0.1:

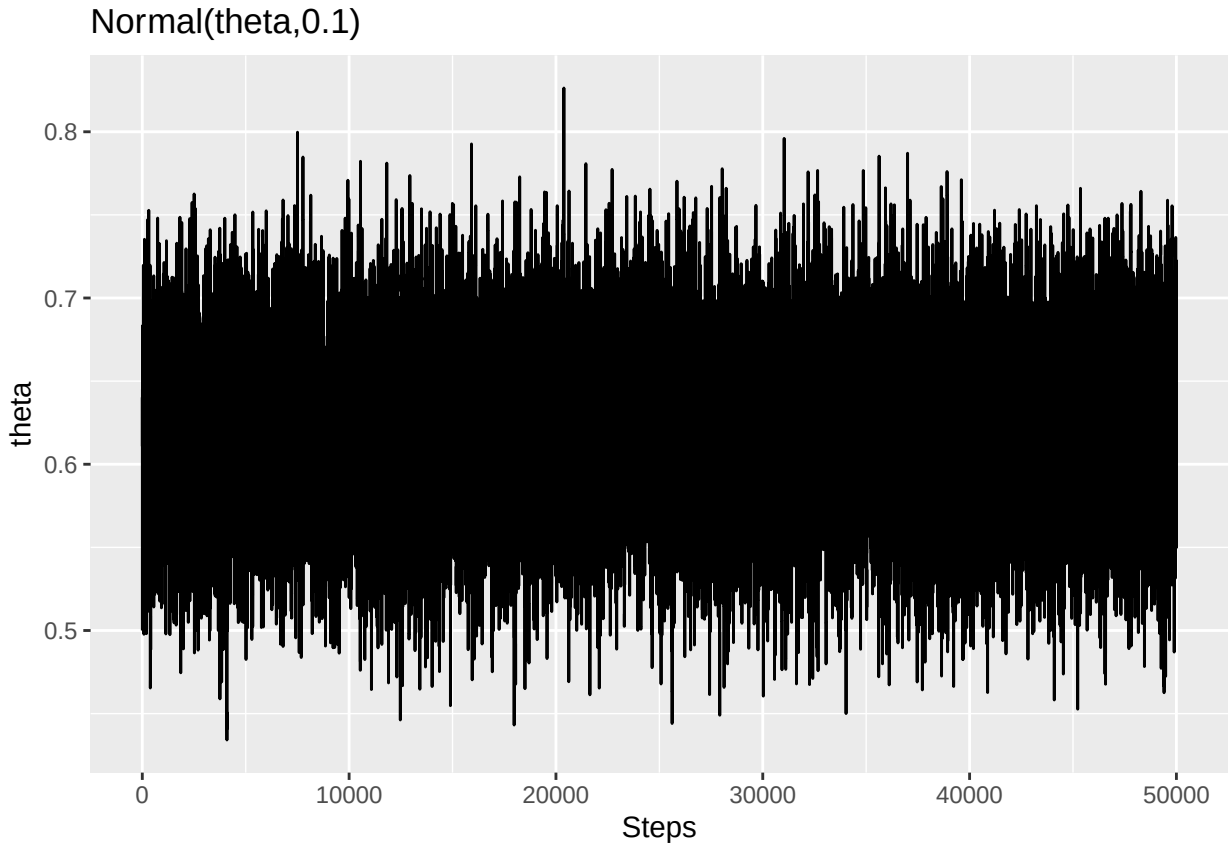

Normal(theta,0.01)
Density
7.5
5.0
2.5
0.0
0.4
0.5
0.6
0.7
0.8
theta


Normal(theta,0.01)
theta
0.8
0.7
0.6
0.5
0.4
0
10000
20000
30000
40000
50000
Steps

(3).

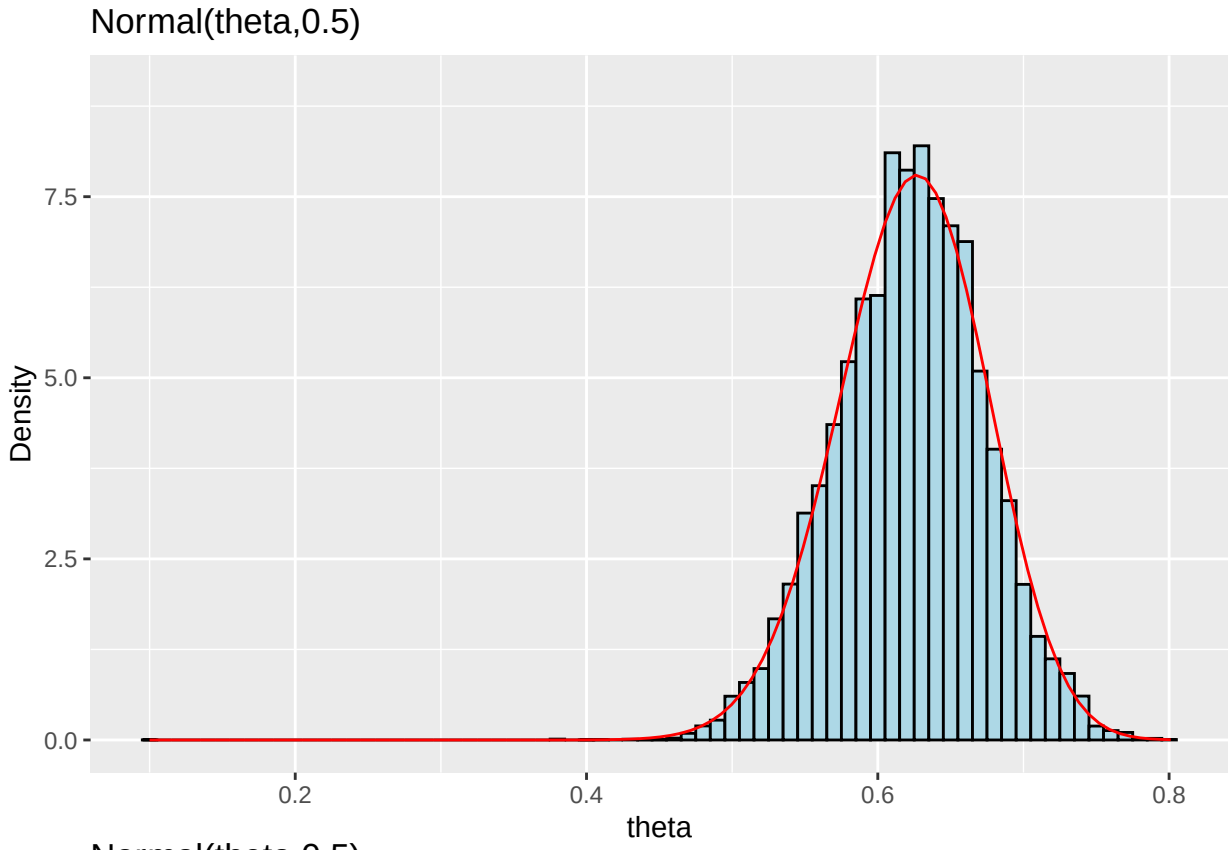
Then for the Normal with same mean as in (2) and standard deviation of 0.1:

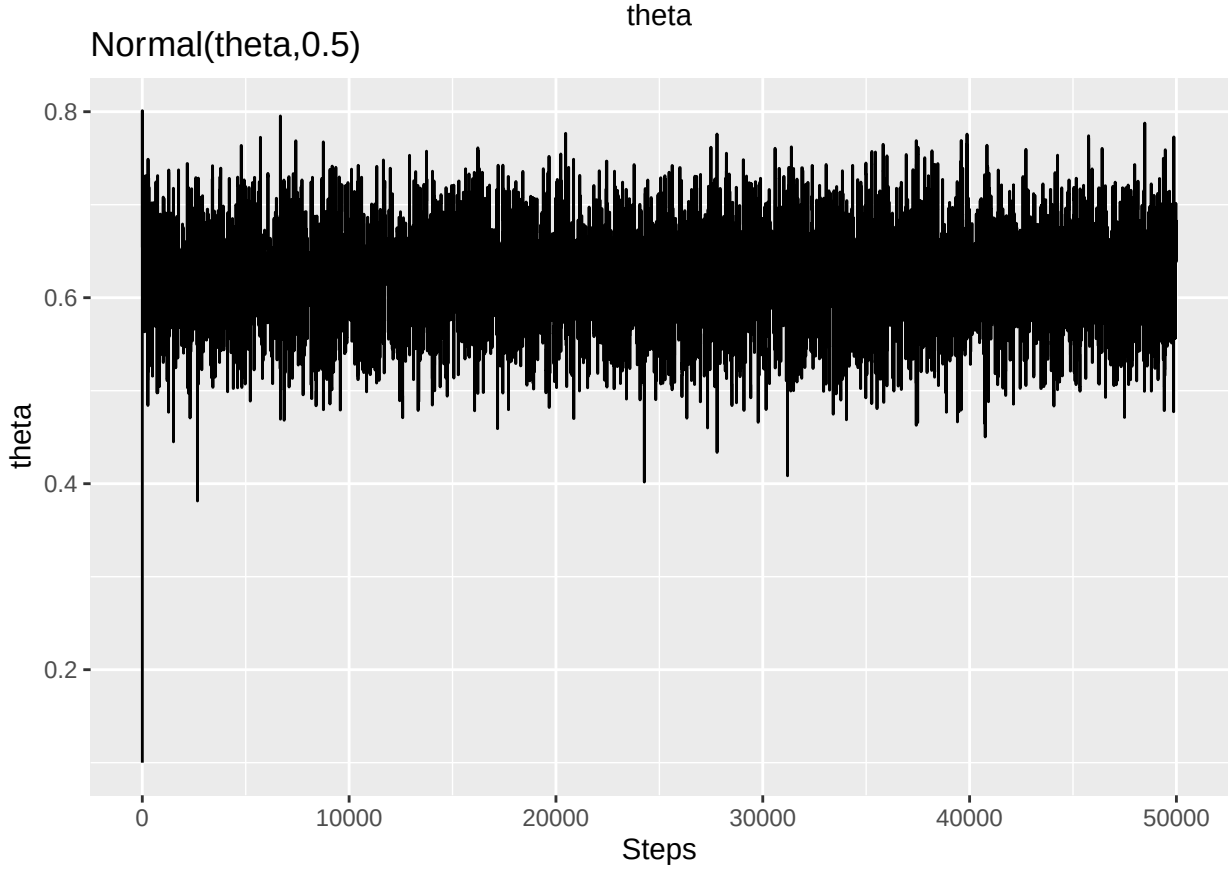




(4).

Then for the Normal with same mean as in (3) and standard deviation of 0.5:

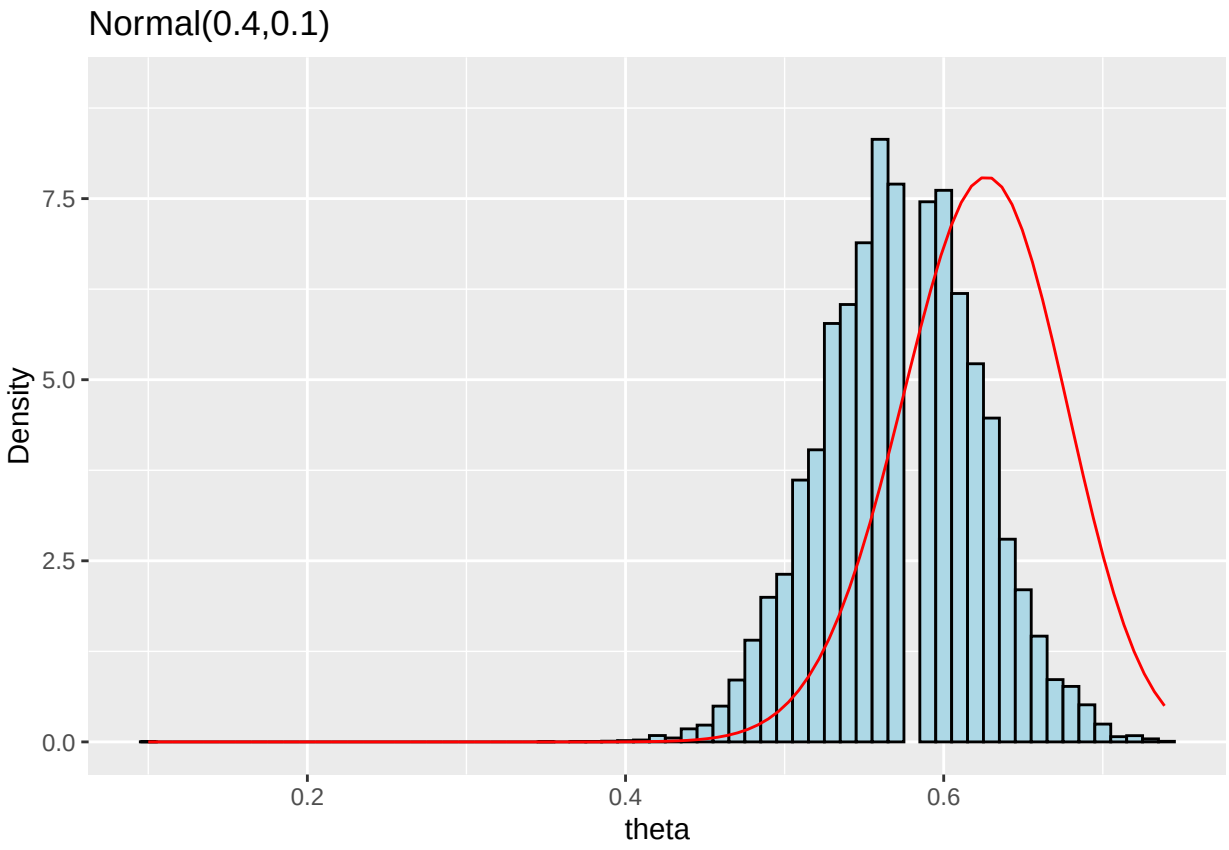

Normal(theta,0.5)
Density
7.5
5.0
2.5
0.0
0.2
0.4
0.6
0.8
theta

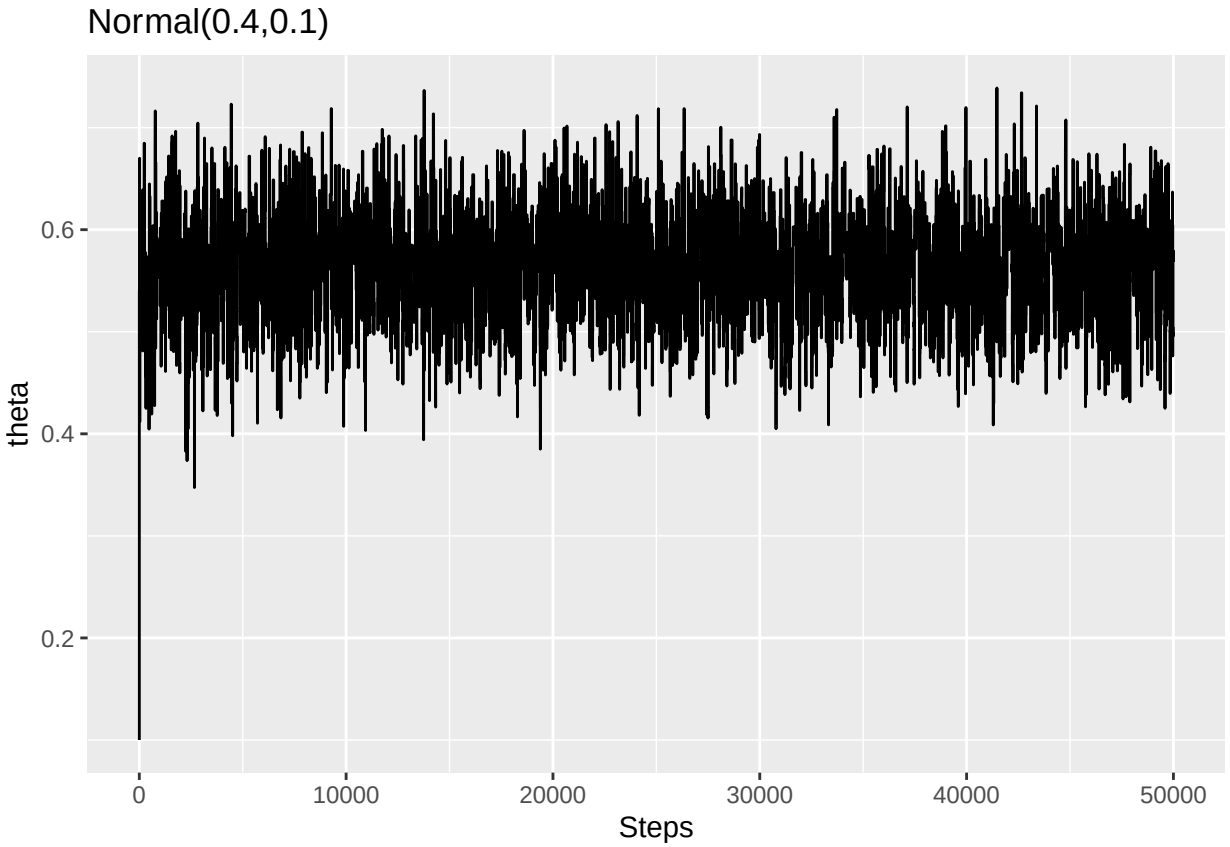

Normal(theta,0.5)
theta
0.8
0.6
0.4
0.2
0
10000
20000
30000
40000
50000
Steps

(5).

Then for the Normal with same mean 0.4 and standard deviation of 0.1:

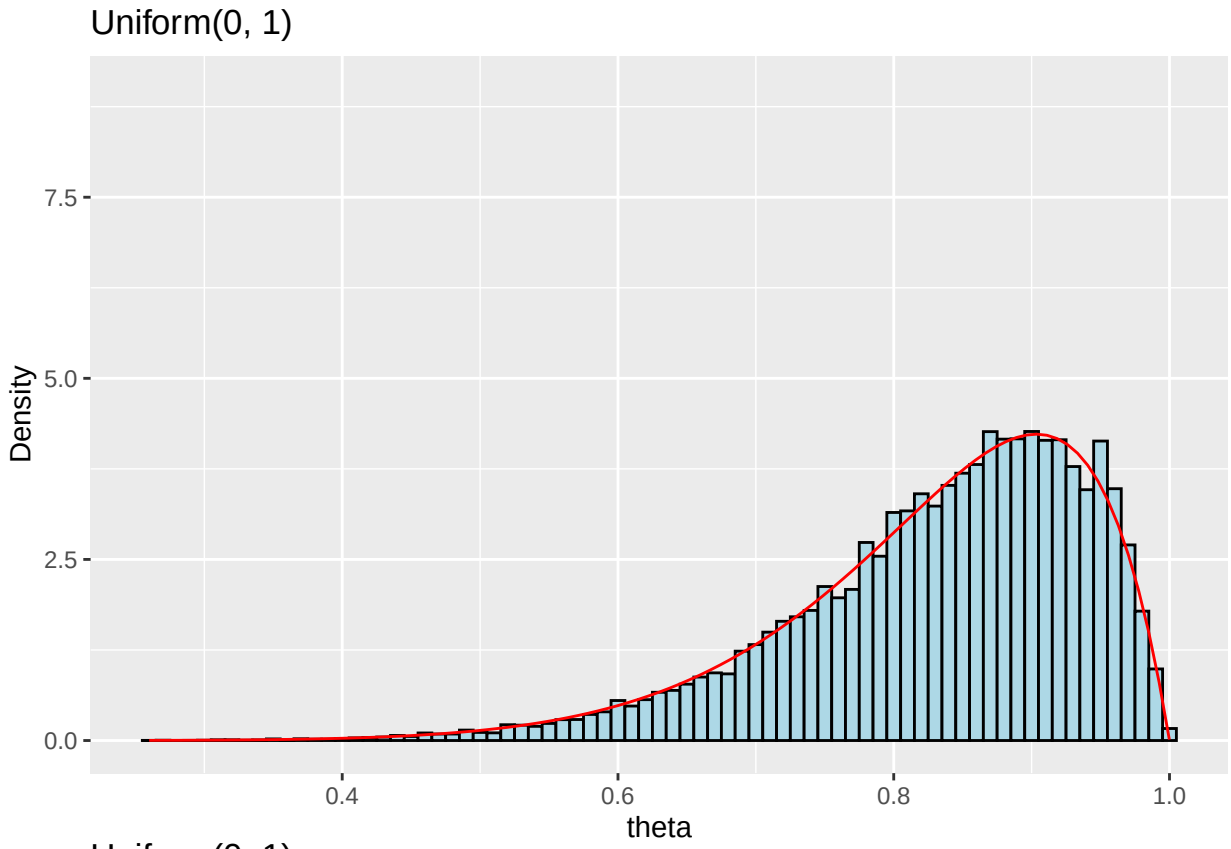
```
## Warning: Removed 1 rows containing missing values (`geom_bar()`).
```

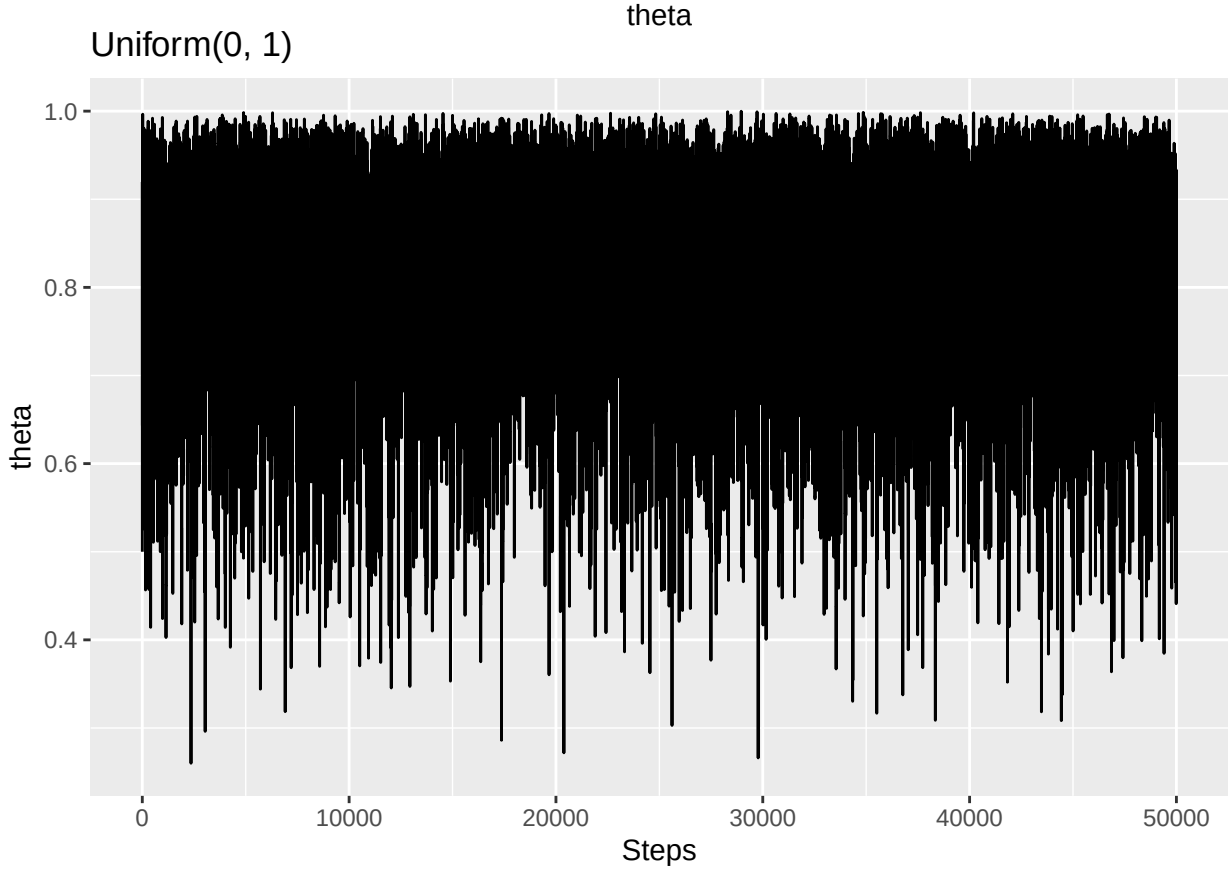




We can assess the convergence as the method used in Question 1 using the Gelman-Rubin method. From the results, former four methods converge well and fast but for the fifth chain, it does not converge well. In contrast, the case with std = 0.1, mean = θ converges fastes. I did not toss some intial values out. For the fifth chain, Metropolis or generalized Metropolis may not behave well.

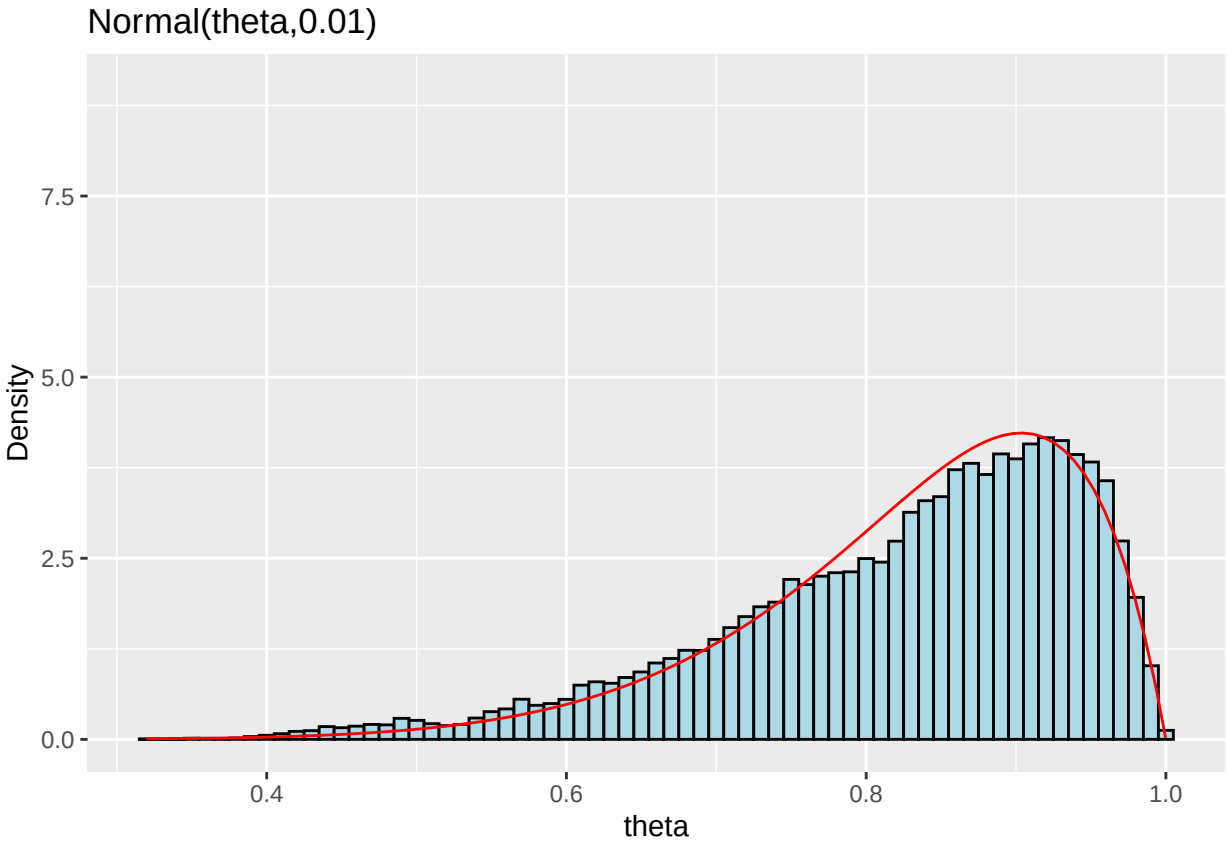
b. Repeat the process above for the data Y = (14, 0, 1, 5) and I get following results: For the Uniform on (0,1):

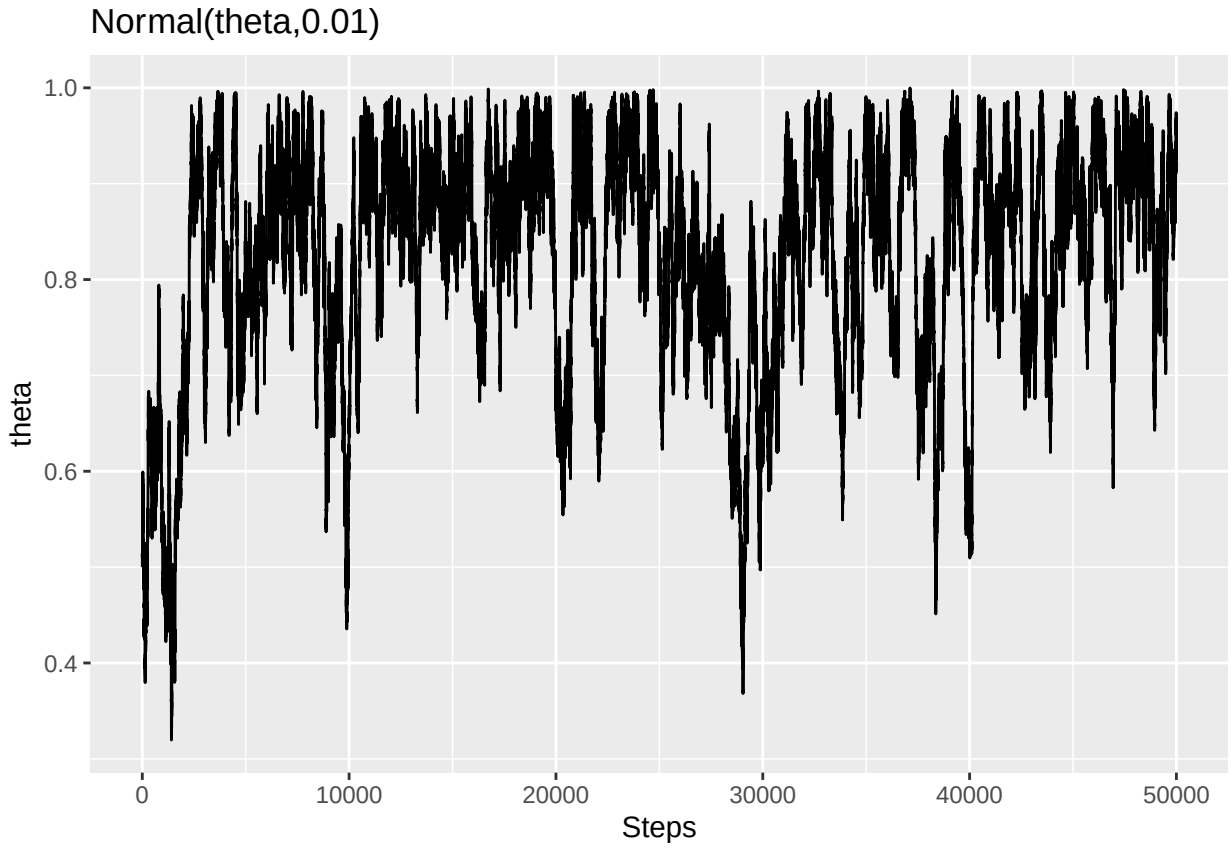




(2).

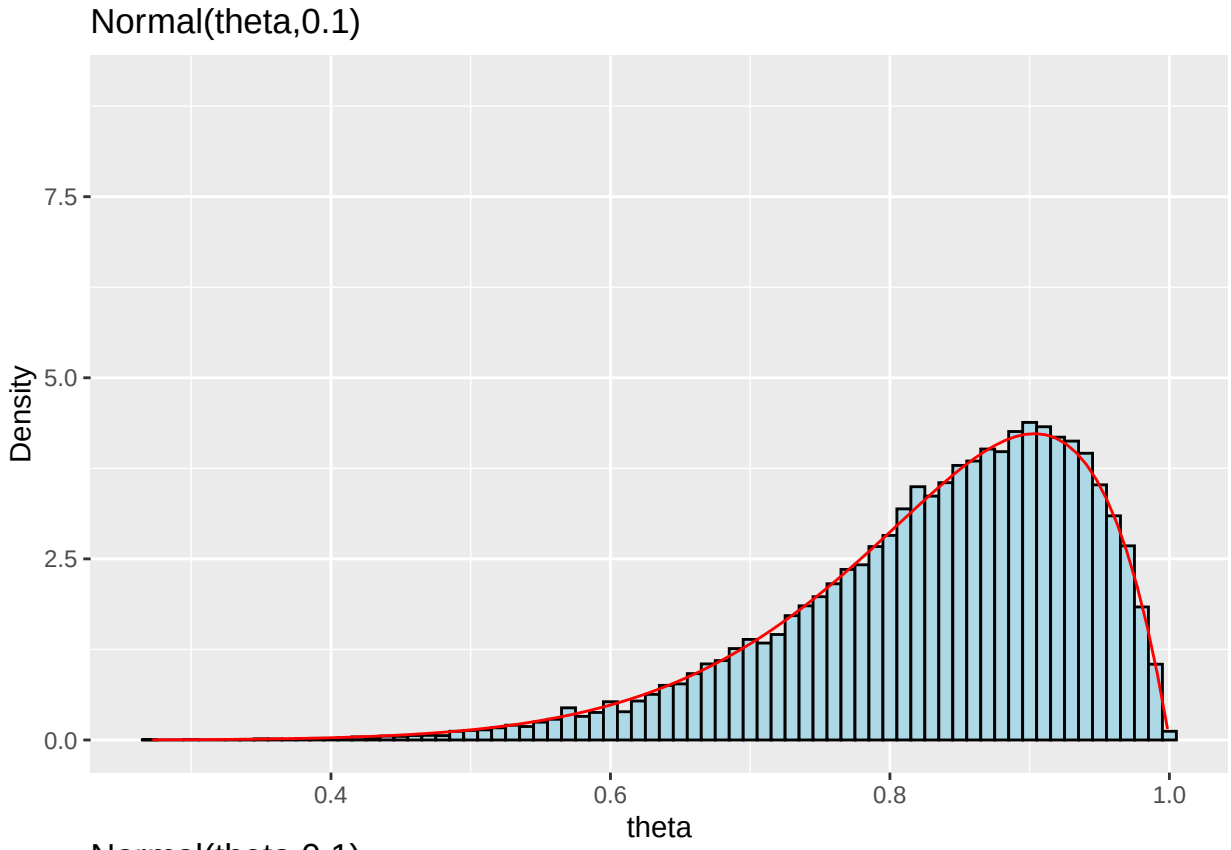
Then for the Normal cenetered at the current point in the chain and standard deviation 0.1:

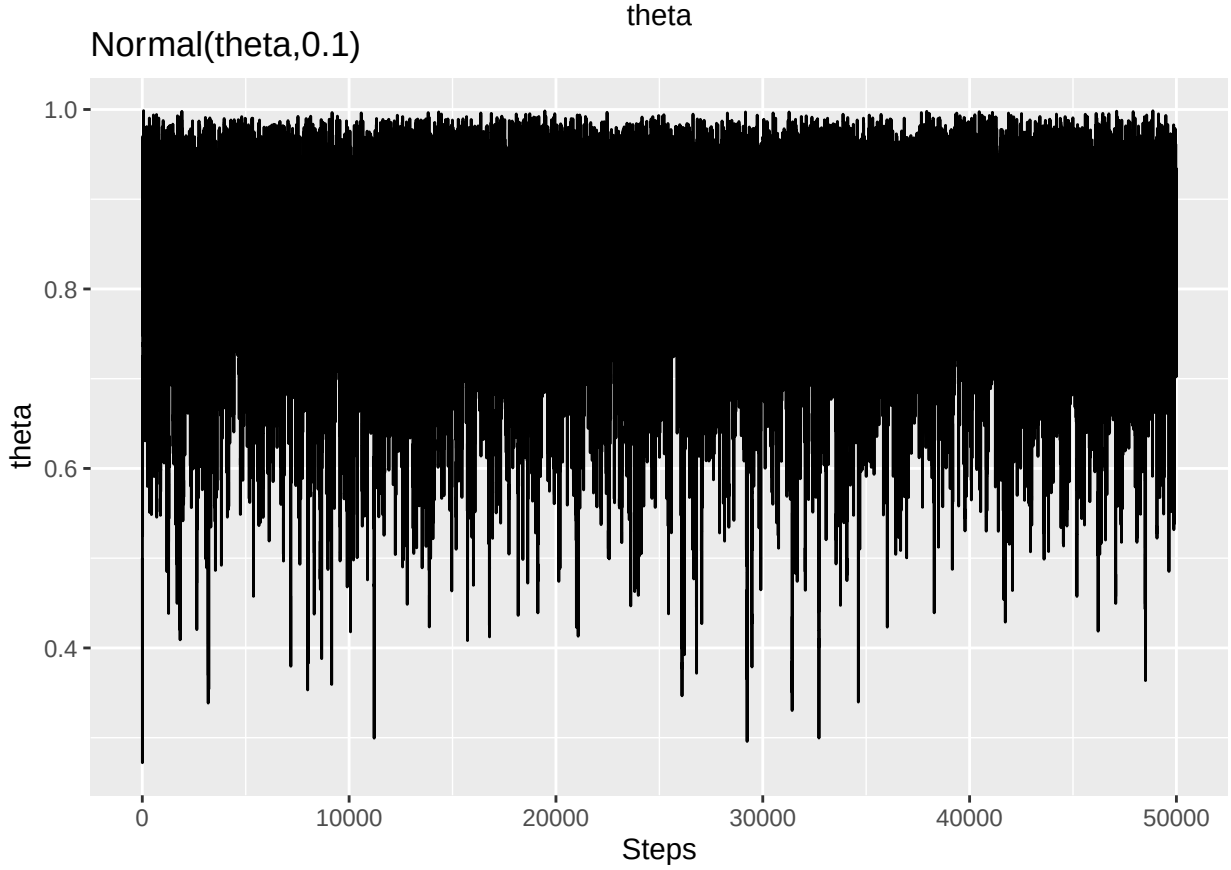




(3).

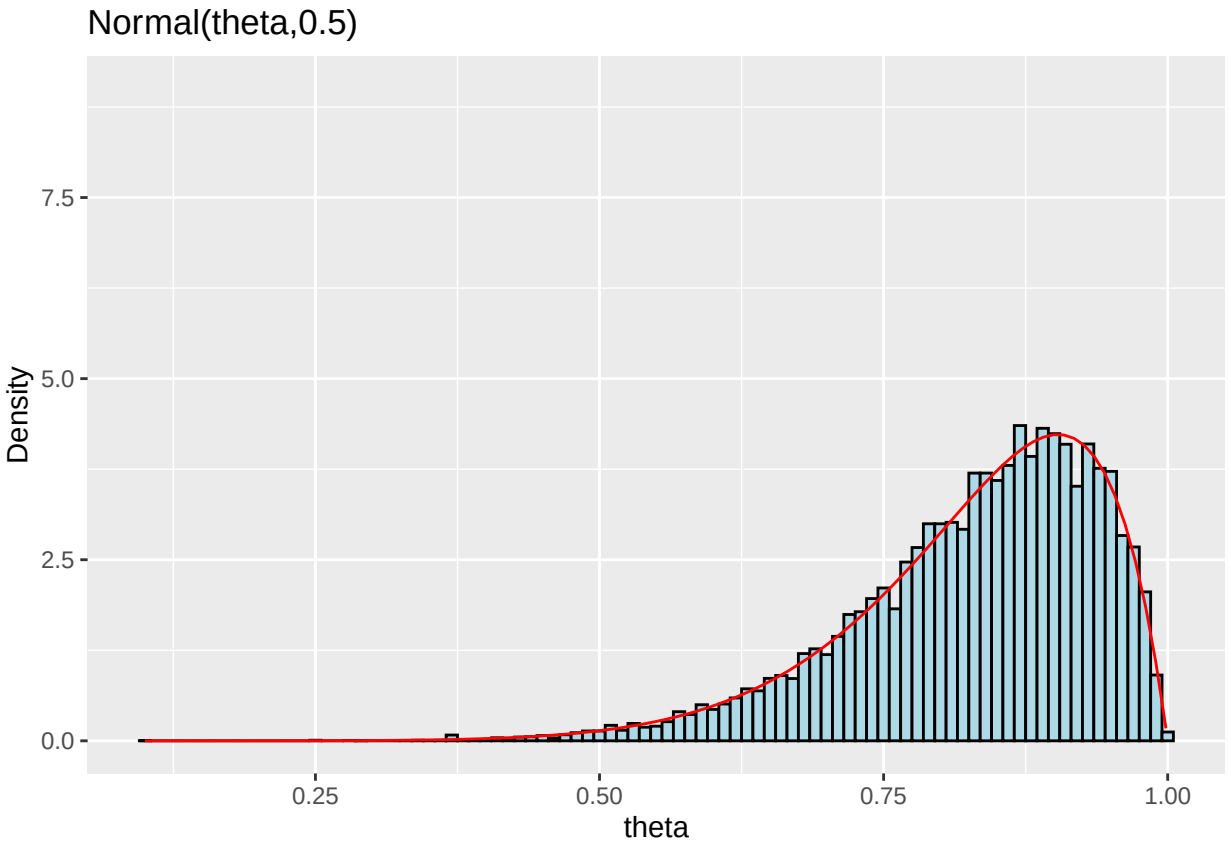
Then for the Normal with same mean as in (2) and standard deviation of 0.1:

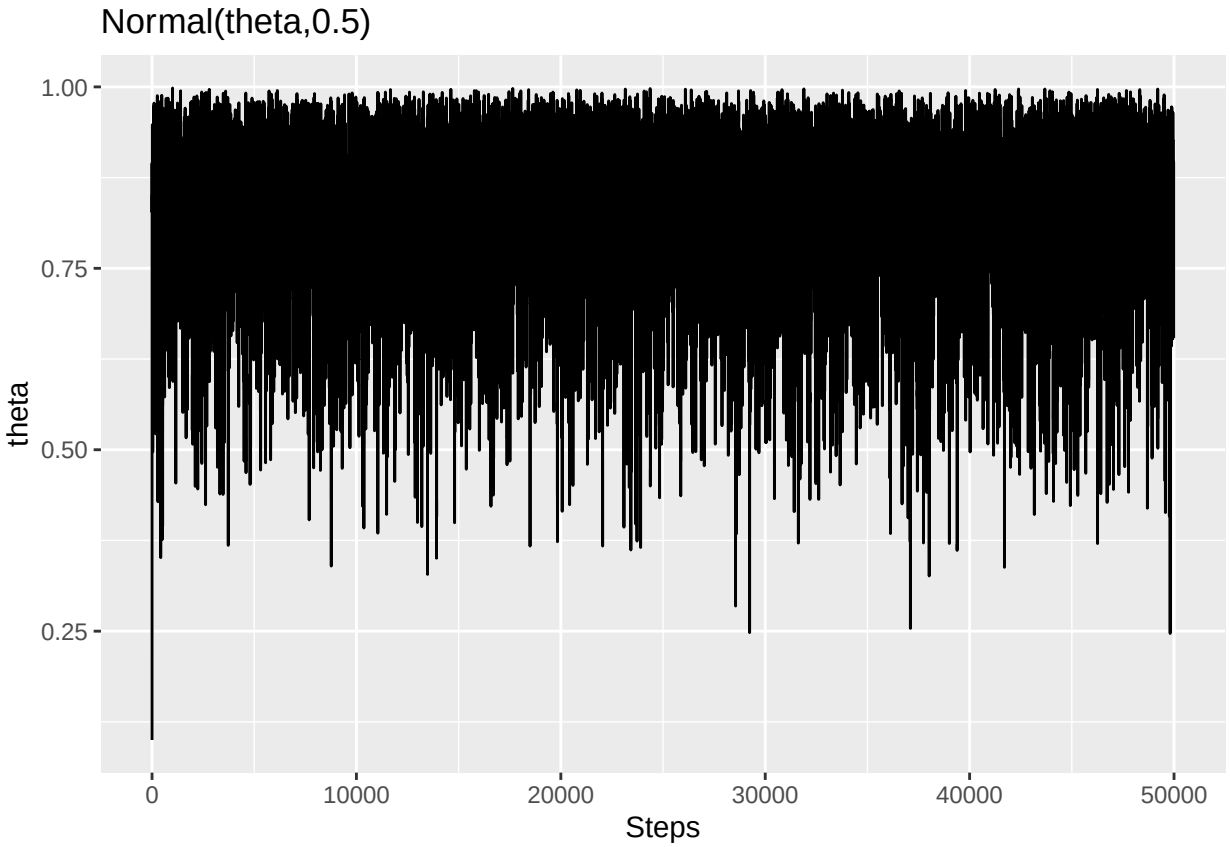

Normal(theta,0.1)
Density
7.5
5.0
2.5
0.0
0.4
0.6
0.8
1.0
theta


Normal(theta,0.1)
theta
1.0
0.8
0.6
0.4
0
10000
20000
30000
40000
50000
Steps

(4).

Then for the Normal with same mean as in (3) and standard deviation of 0.5:

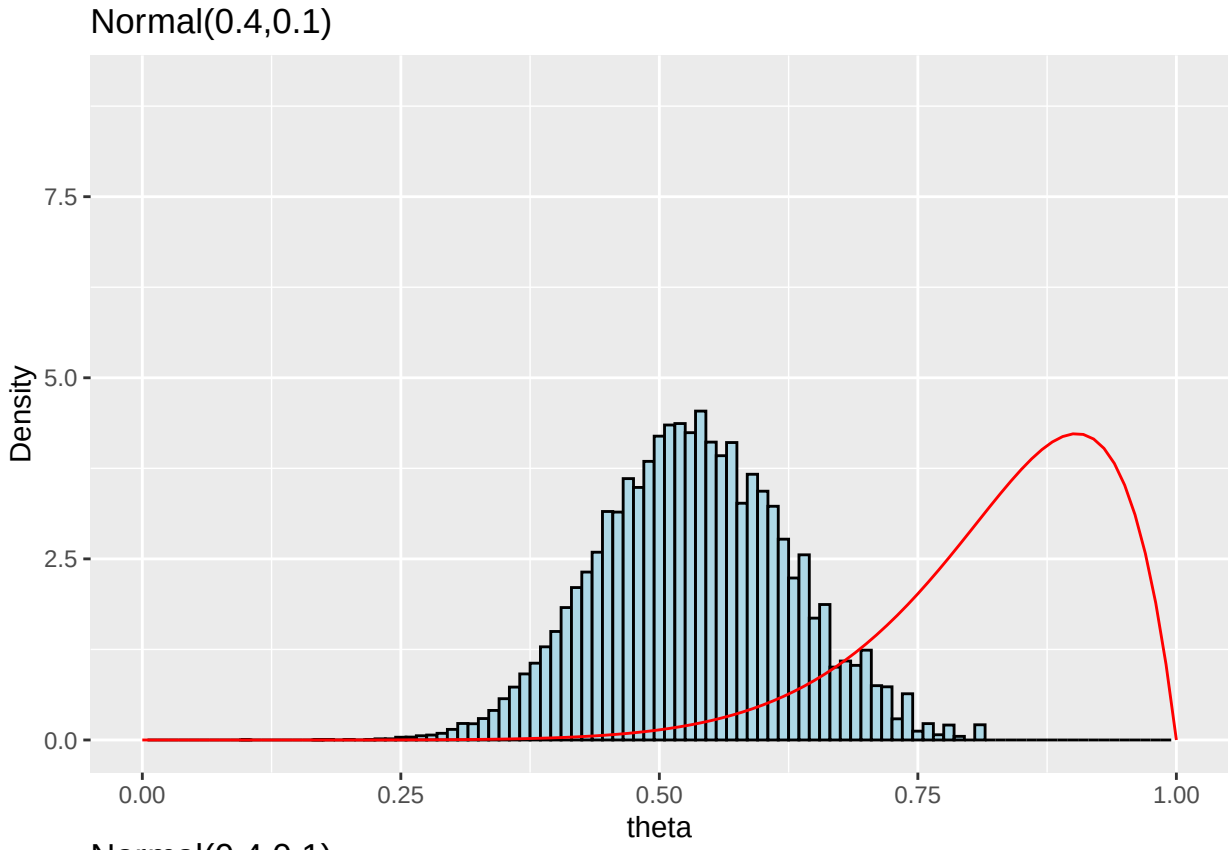


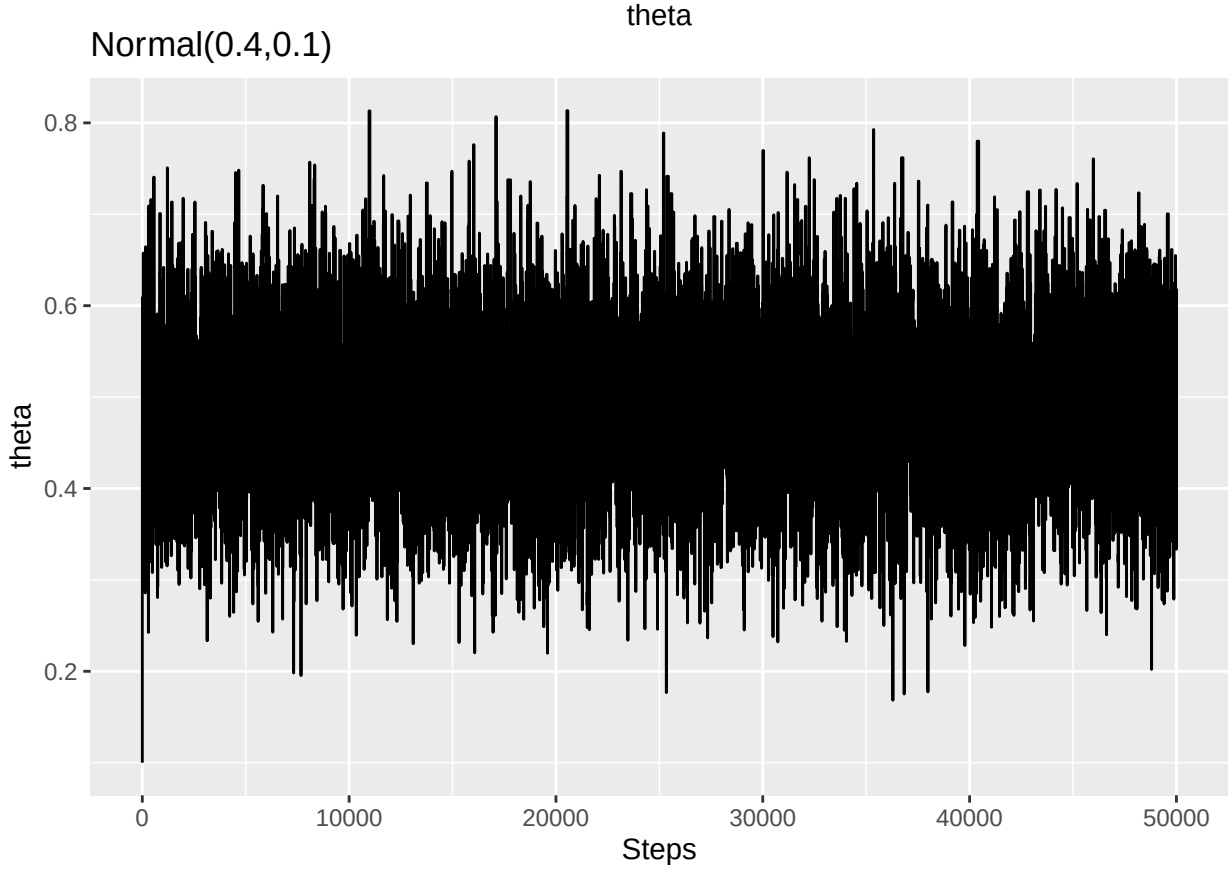


(5).

Then for the Normal with same mean 0.4 and standard deviation of 0.1:

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```





Similarly, we can assess the convergence as the method used in Question 1 using the Gelman-Rubin method.

From the results, former four methods converge well but for the fifth chain, it does not converge well. However, for the second chain, it shows the problem of slow convergence.

3.

```
## [1] "Following the order of Uniform(0,1), Normal(theta, 0.01), , Normal(theta, 0.1), , Normal(theta,
```

```
## [1] "The means of the first dataset:"
```

```
## [1] 0.6235314 0.6233531 0.6232230 0.6218123 0.5732043
```

```
## [1] "The variances of the first dataset:"
```

```
## [1] 0.002584313 0.002671752 0.002529258 0.002536835 0.002239955
```

```
## [1] "The means of the second dataset:"
```

```
## [1] 0.8331044 0.8239688 0.8335206 0.8308692 0.5340515
```

```
## [1] "The variances of the seconnd dataset:"
```

```
## [1] 0.01141151 0.01403171 0.01112360 0.01146826 0.00835545
```

3. a

(i)

$$p(\mu,\theta,\sigma_\varepsilon^2,\sigma_\theta^2,Y) \propto p(\mu)p(\theta|\mu,\sigma_\theta^2,\sigma_\varepsilon^2,Y)$$
$$\propto \exp\{-\frac{1}{2\sigma_0^2}(\mu-\mu_0)^2\}\exp\{-\frac{1}{2\sigma_\theta^2}\sum_{i=1}^{K}(\theta_i-\mu)^2\}$$
$$\propto \exp\{\frac{\sigma_\theta^2\mu^2-2\sigma_\theta^2\mu_0\mu-2\mu\sigma_0^2\sum_{i=1}^{K}\theta_i+K\sigma_0^2\mu^2}{2\sigma_0^2\sigma_\theta^2}\}$$
$$\propto \exp\{-\frac{(\mu-\hat{\mu})^2}{\hat{\sigma}^2}\} \sim N(\hat{\mu},\hat{\sigma}^2) \text{ with } \hat{\mu}=\frac{\sigma_\theta^2\mu_0+\sigma_0^2\sum\theta_i}{\sigma_\theta^2+K\sigma_0^2},\ \hat{\sigma}^2=\frac{\sigma_0^2\sigma_\theta^2}{\sigma_\theta^2+K\sigma_0^2}$$

(ii) $$p(\theta_i|\mu,\sigma_\varepsilon^2,\sigma_\theta^2,Y) \propto p(Y_i|\mu,\sigma_\varepsilon^2,\theta_i)p(\theta_i|\mu,\sigma_\theta^2)$$
$$\propto \exp\{-\frac{(\theta_i-\mu)^2}{2\sigma_\theta^2}\}\exp\{-\frac{\sum_{j=1}^{J}Y_{ij}}{2\sigma_\varepsilon^2}\}$$
$$\propto \exp\left\{\frac{(\theta_i-\frac{\sigma_\theta^2\sum_{j=1}^{J}Y_{ij}}{J\sigma_\theta^2+\sigma_\varepsilon^2})^2}{2\frac{\sigma_\theta^2\sigma_\varepsilon^2}{J\sigma_\theta^2+\sigma_\varepsilon^2}}\right\}$$
$$\propto \exp\left\{\frac{(\theta_i-\hat{\mu})^2}{2\hat{\sigma}^2}\right\} \sim N(\hat{\mu},\hat{\sigma}^2) \text{ with } \hat{\mu}=\frac{\sigma_\theta^2\bar{Y}}{J\sigma_\theta^2+\sigma_\varepsilon^2},\ \hat{\sigma}^2=\frac{\sigma_\theta^2\sigma_\varepsilon^2}{J\sigma_\theta^2+\sigma_\varepsilon^2}$$

(iii) $$p(\sigma_\varepsilon^2|\mu,\theta,\sigma_\theta^2,Y) \propto p(\sigma_\varepsilon^2)p(Y|\sigma_\varepsilon^2,\mu,\theta,\sigma_\theta^2,Y)$$
$$\propto \sigma_\varepsilon^{2(-a_\varepsilon-1)}\exp\{-\frac{b_\varepsilon}{\sigma_\varepsilon^2}\}\sigma_\varepsilon^{-KJ}\exp\{-\frac{\sum_{i=1}^{K}\sum_{j=1}^{J}(Y_{ij}-\theta_i)^2}{2\sigma_\varepsilon^2}\}$$
$$\propto \sigma_\varepsilon^{2(-a_\varepsilon-1-KJ/2)}\exp\{-\frac{2b_\varepsilon+\sum_{i=1}^{K}\sum_{j=1}^{J}(Y_{ij}-\theta_i)^2}{2\sigma_\varepsilon^2}\} \sim IG(a_\varepsilon+\frac{KJ}{2},\ b_\varepsilon+\frac{1}{2}\sum_{i=1}^{K}\sum_{j=1}^{J}(Y_{ij}-\theta_i)^2)$$

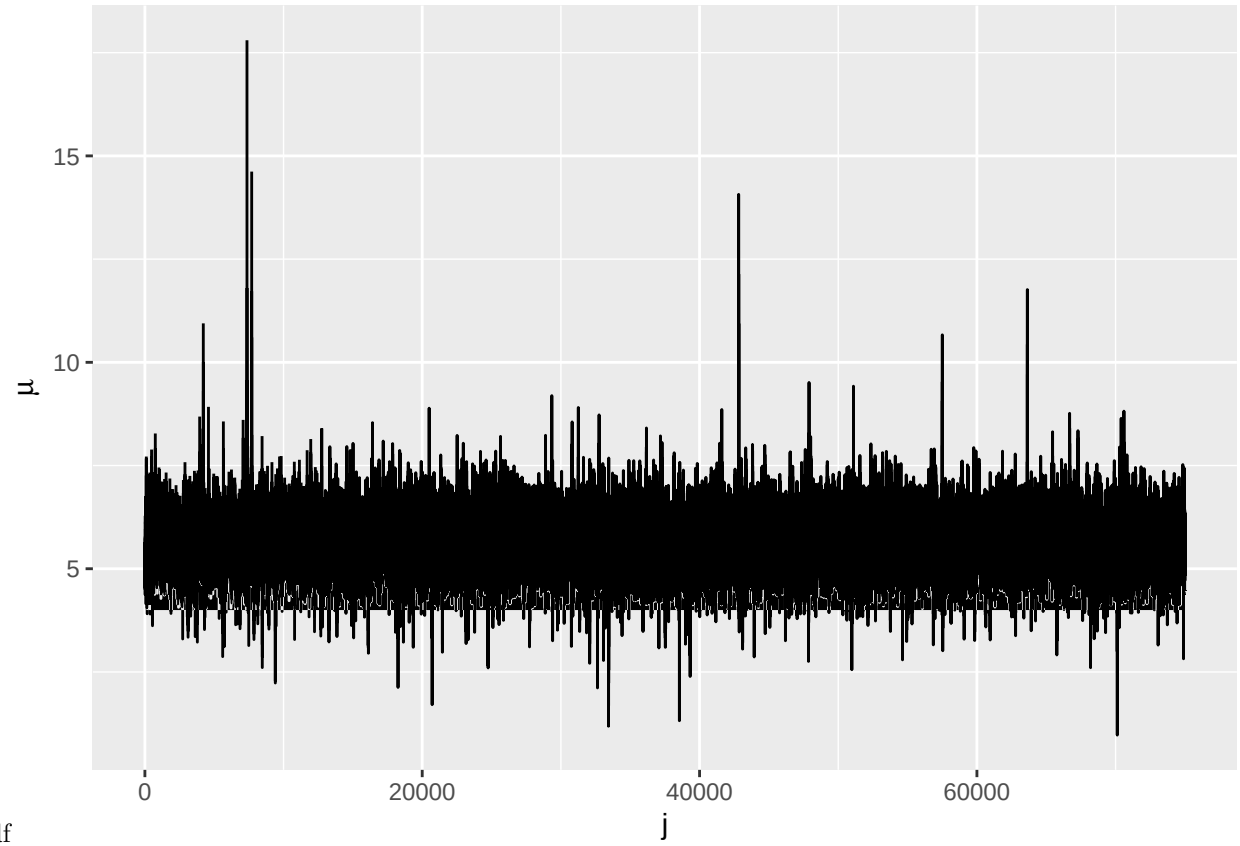
(iv) $$p(\sigma_\theta^2|\mu,\theta,\sigma_\varepsilon^2,Y) \propto p(\sigma_\theta^2)p(\theta|\mu,\sigma_\theta^2,Y,\sigma_\varepsilon^2)$$
$$\propto \sigma_\theta^{2(-a_1-1)}\exp\{-\frac{b_1}{\sigma_\theta^2}\}\sigma_\theta^{-K}\exp\{-\frac{\sum_{i=1}^{K}(\theta_i-\mu)^2}{2\sigma_\theta^2}\}$$
$$\propto \sigma_\theta^{2(-a_1-1-\frac{K}{2})}\exp\{-\frac{2b_1+\sum_{i=1}^{K}(\theta_i-\mu)^2}{2\sigma_\theta^2}\} \sim IG(a_1+\frac{K}{2},\ b_1+\frac{1}{2}\sum_{i=1}^{K}(\theta_i-\mu)^2)$$

Figure 1: Q3

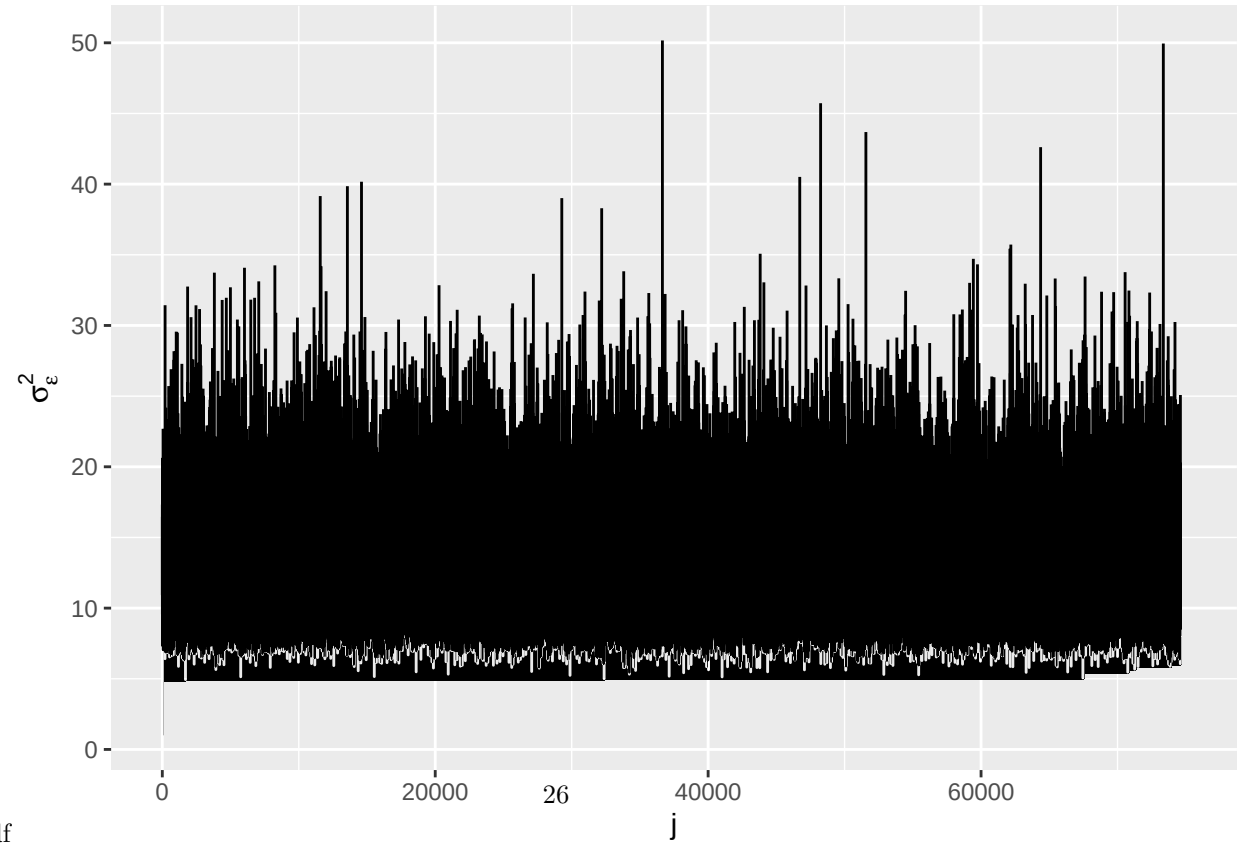
Question 3

a.

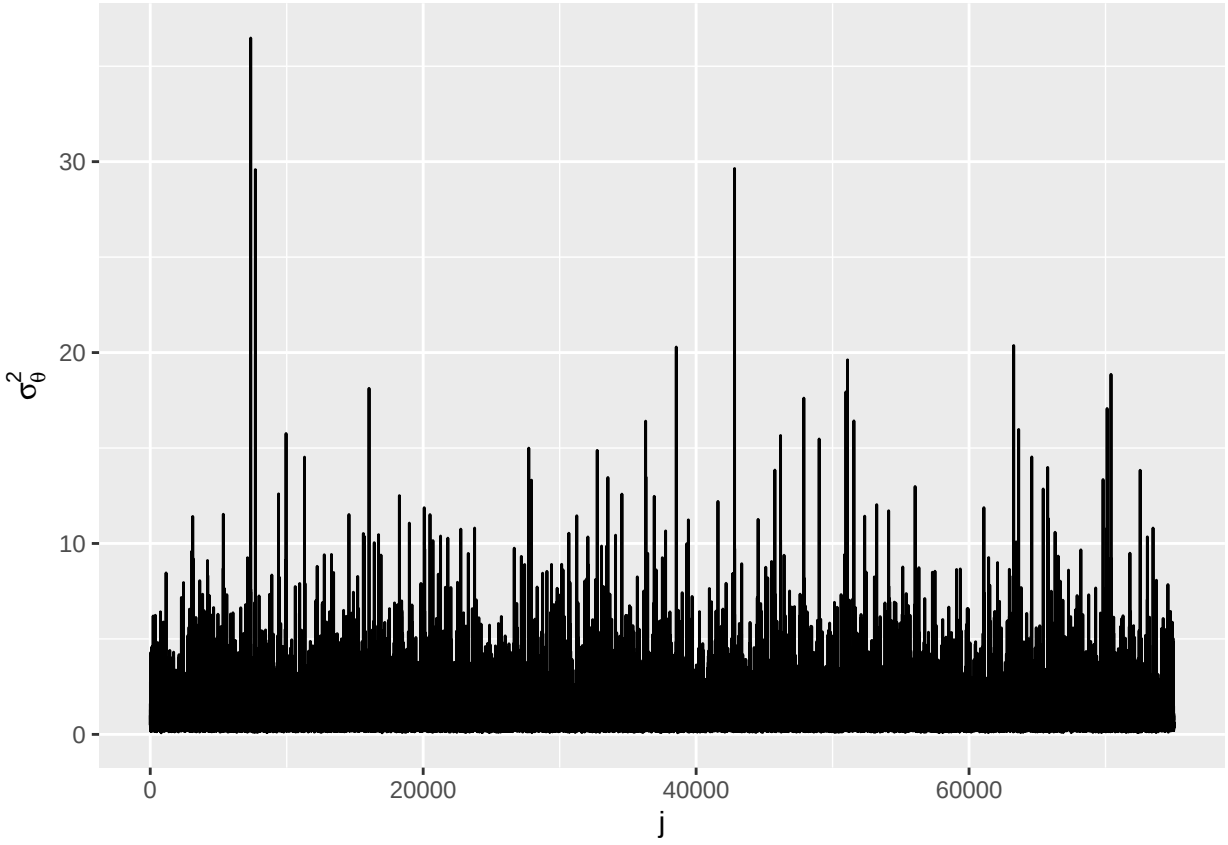
b.



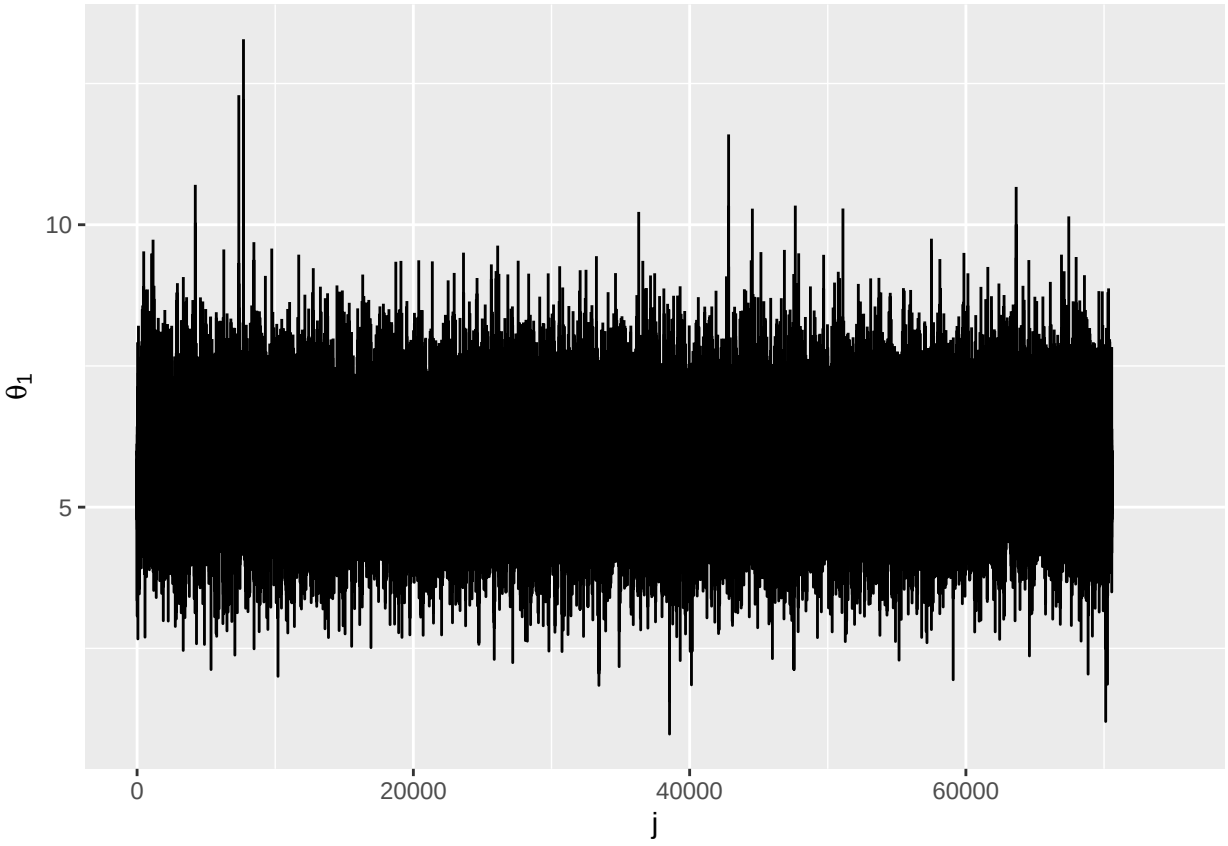
3-1.pdf



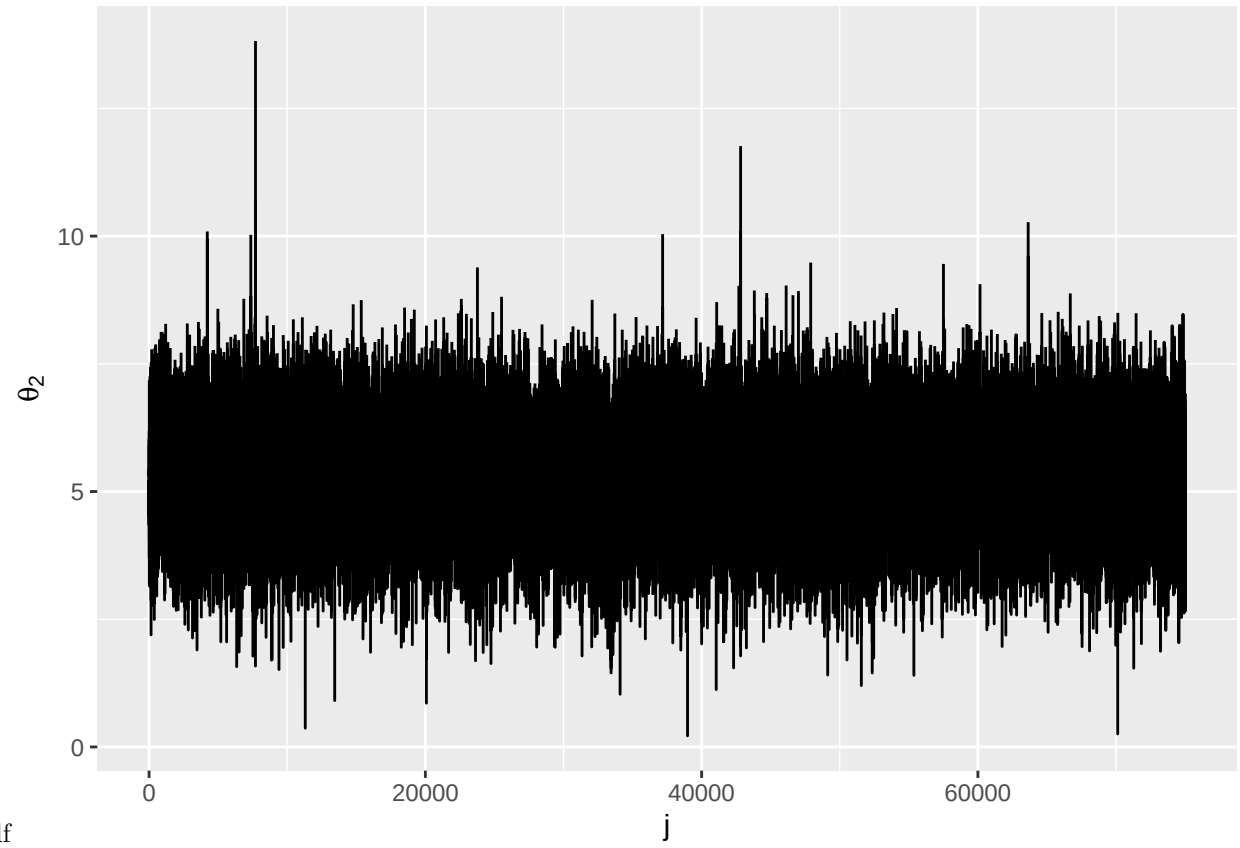
3-2.pdf


30
20
10
0
σ^2_θ
0
20000
40000
60000
j

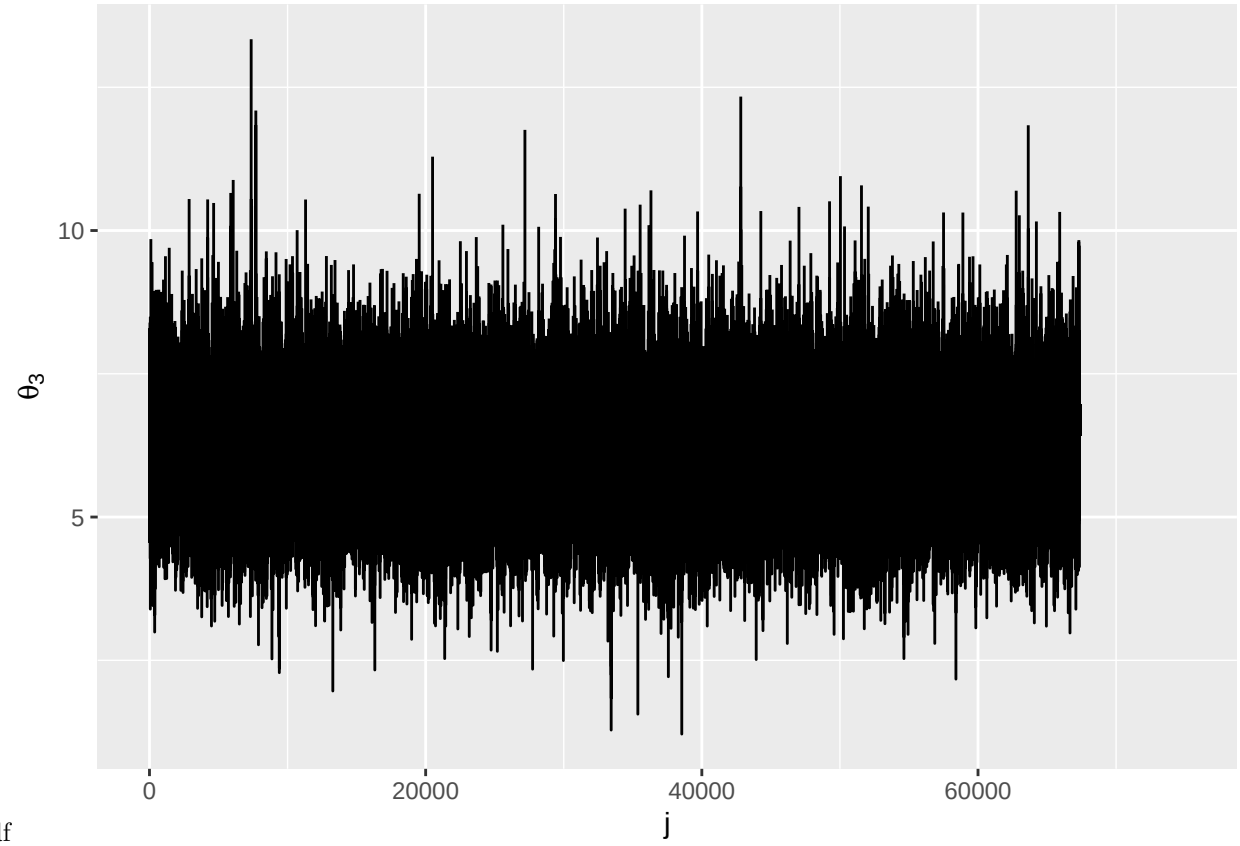
3-3.pdf


10
5
θ_1
0
20000
40000
60000
j

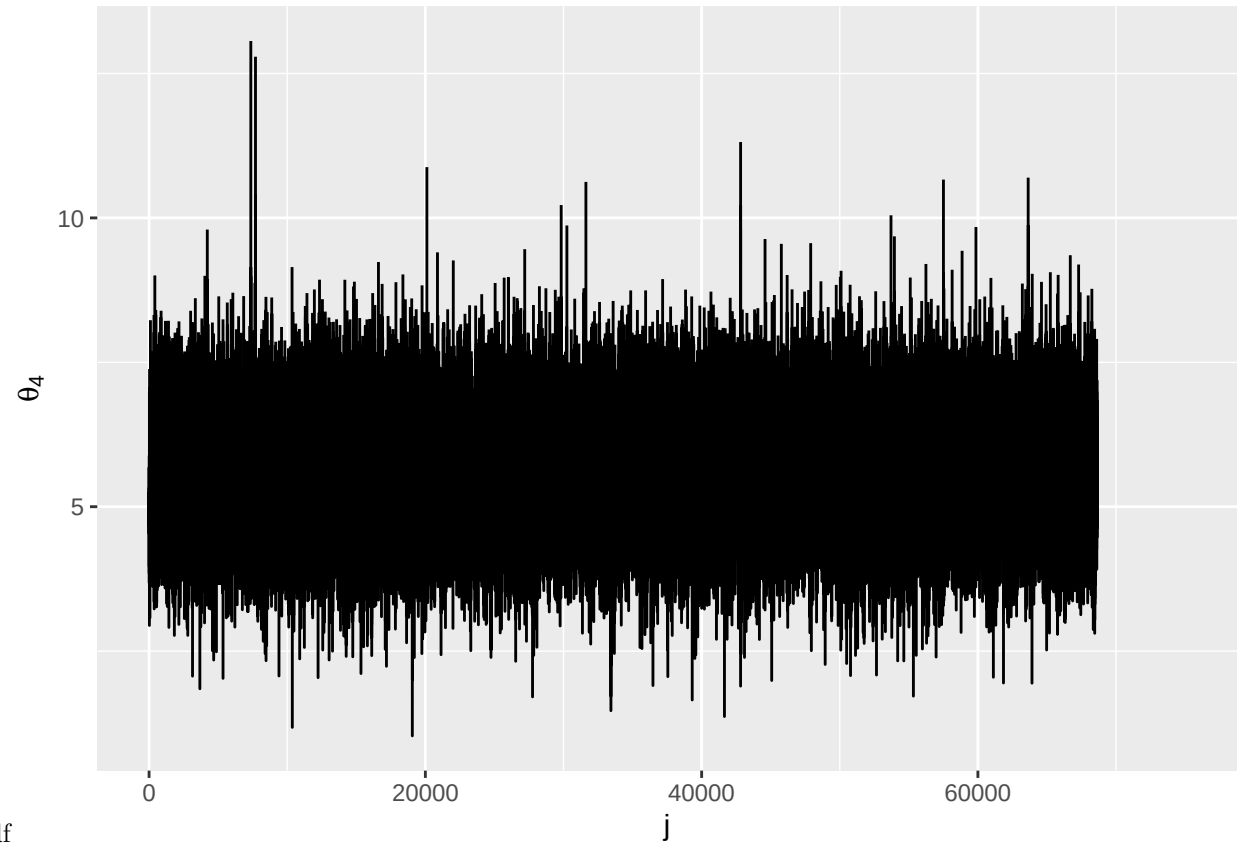
3-4.pdf



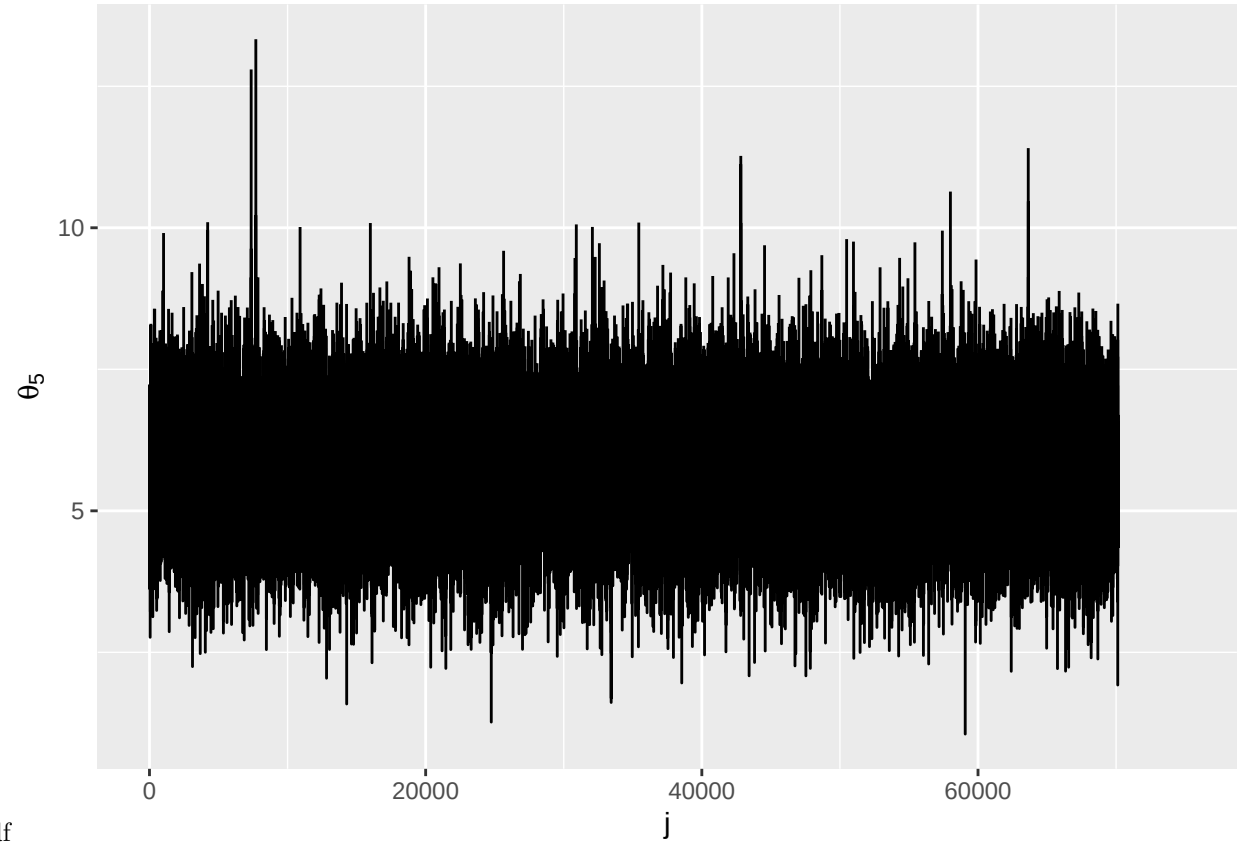
3-5.pdf



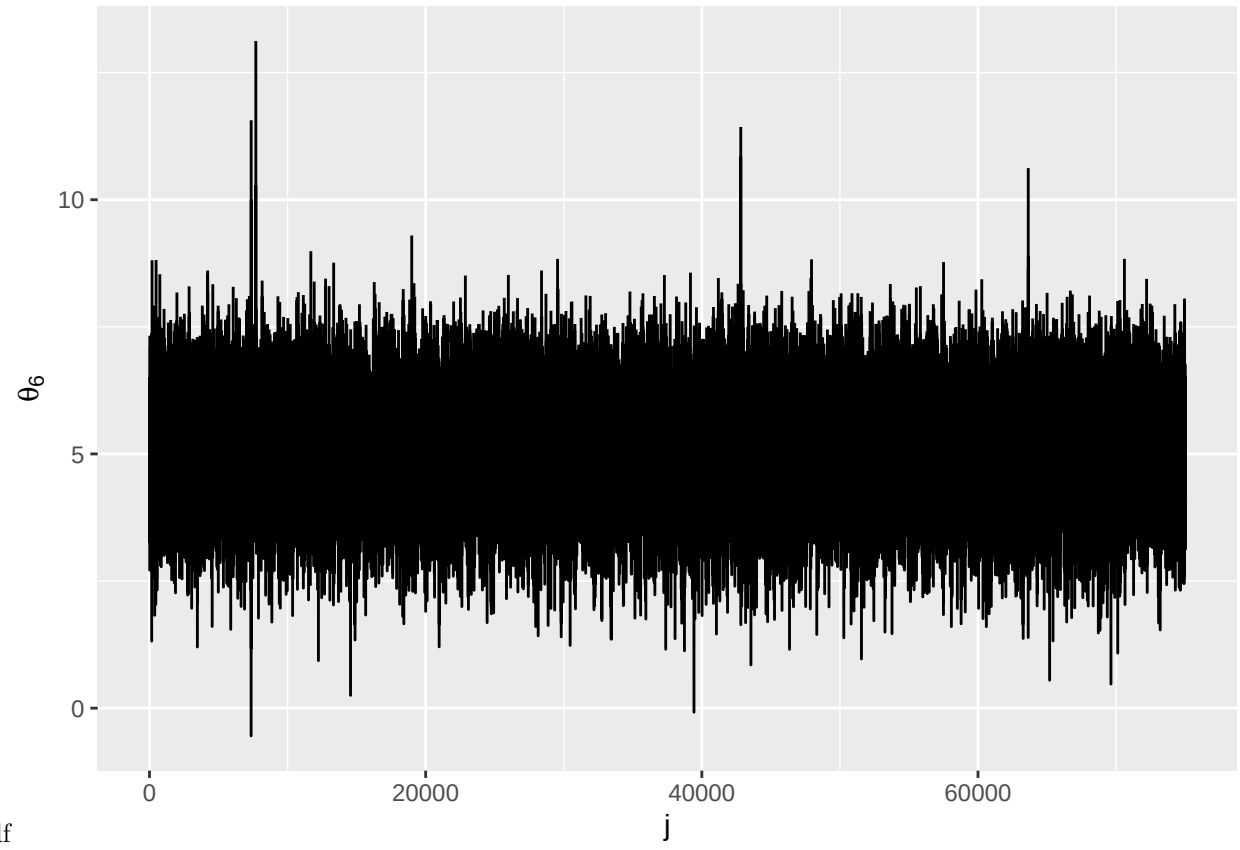
3-6.pdf



3-7.pdf

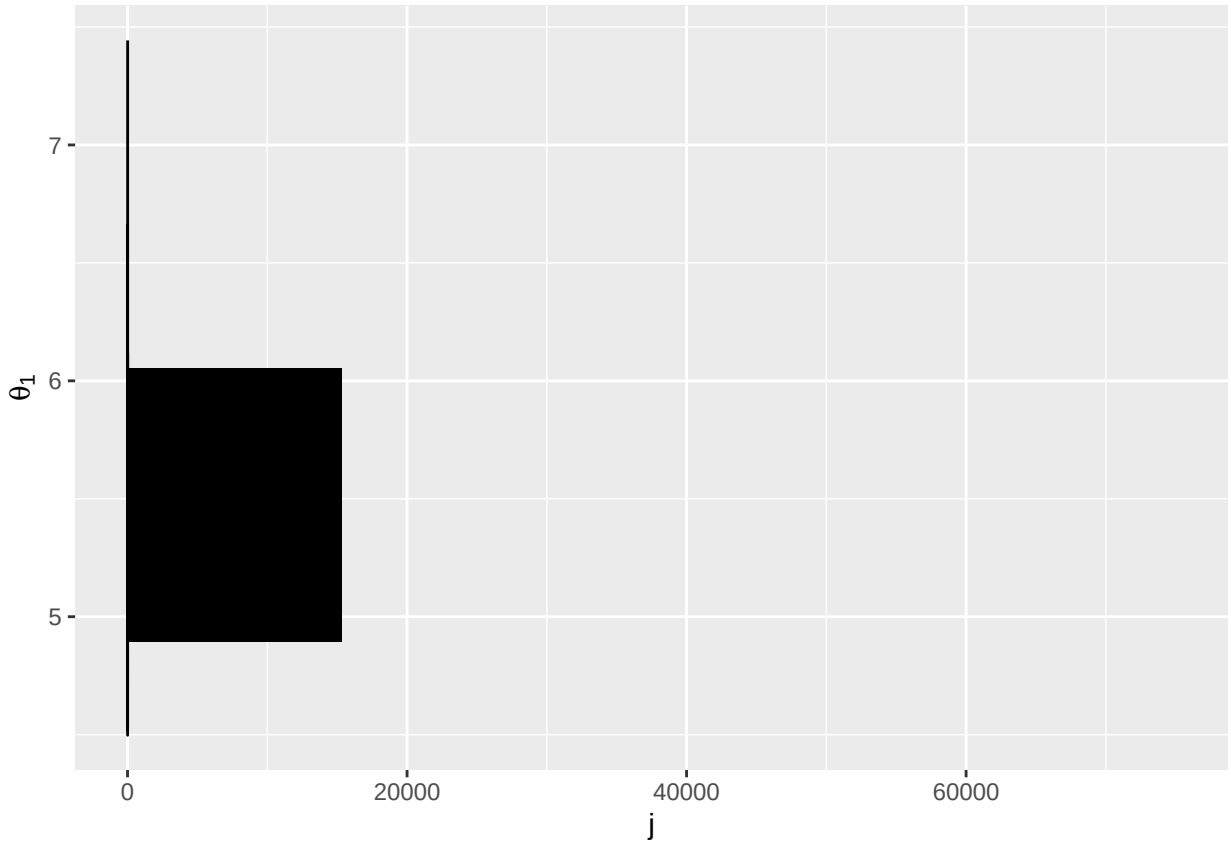


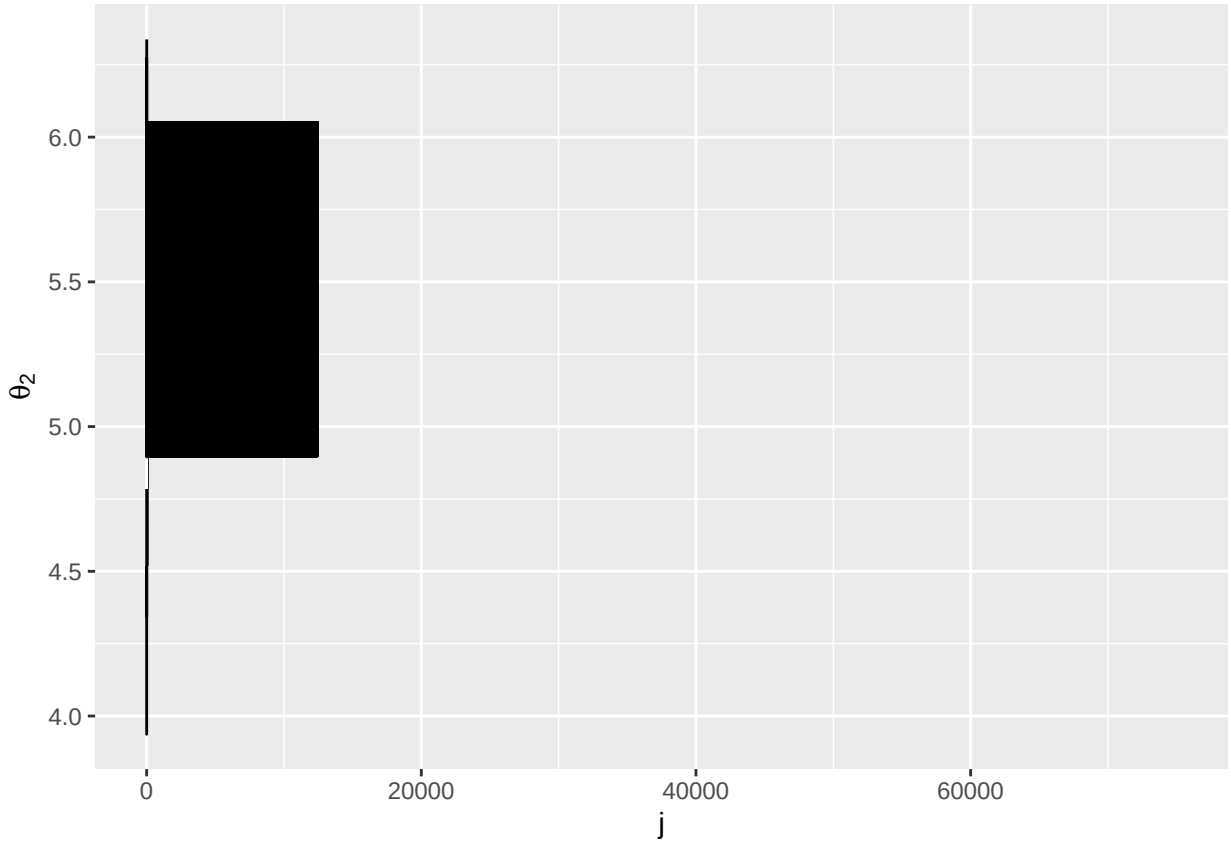
3-8.pdf

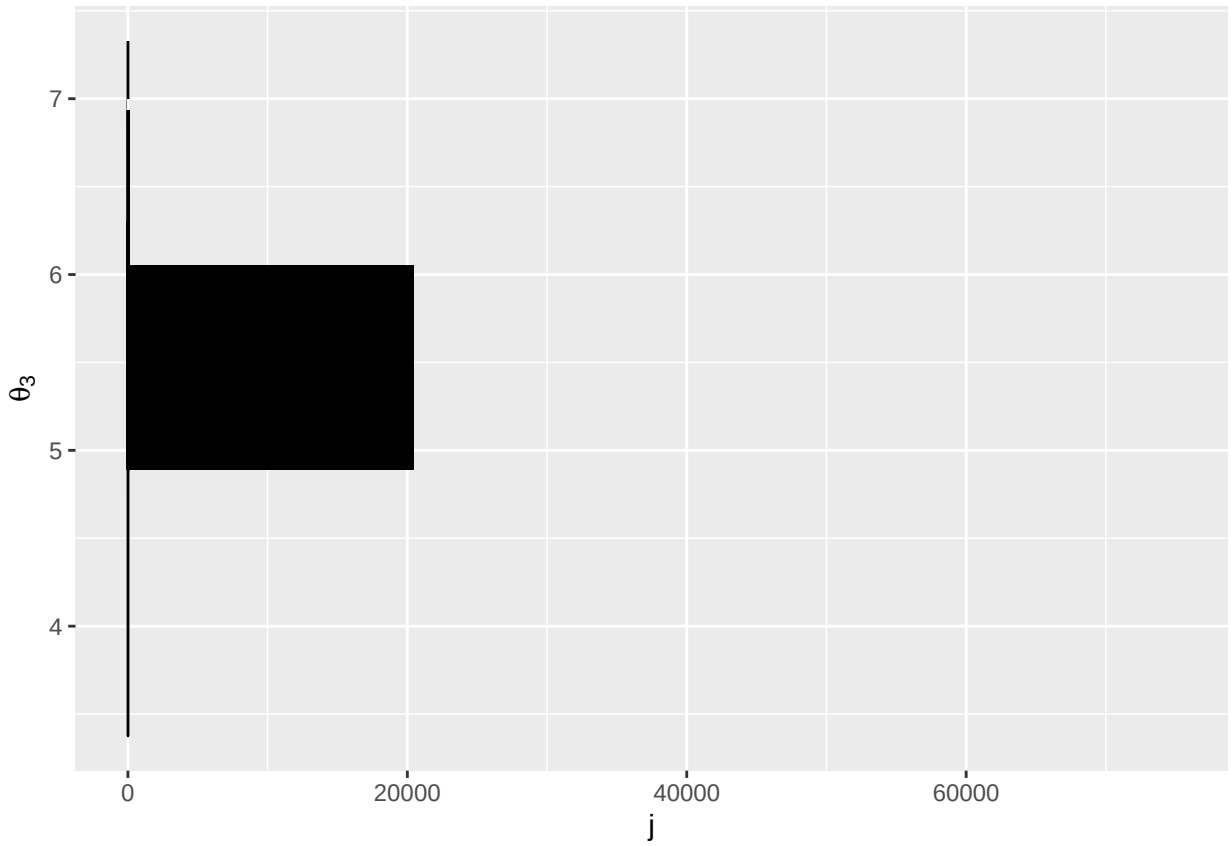


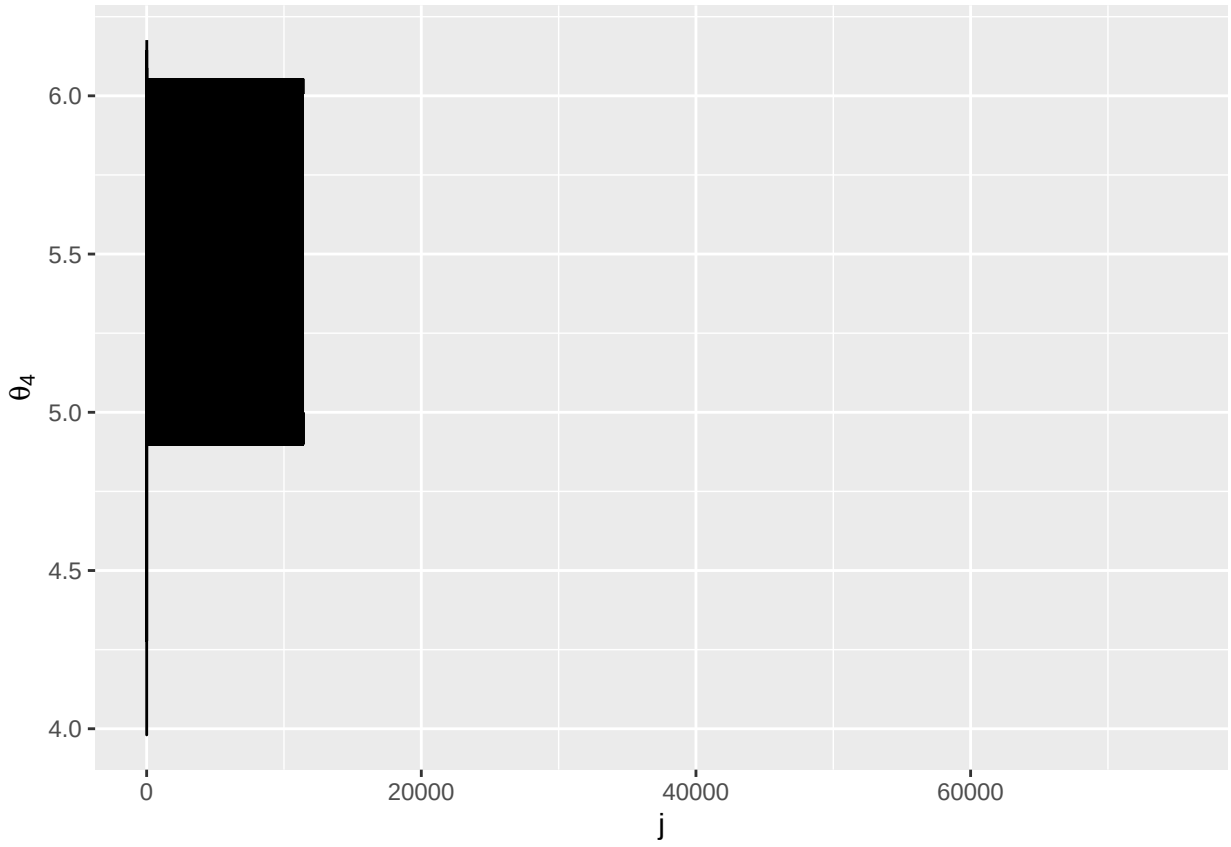
3-9.pdf

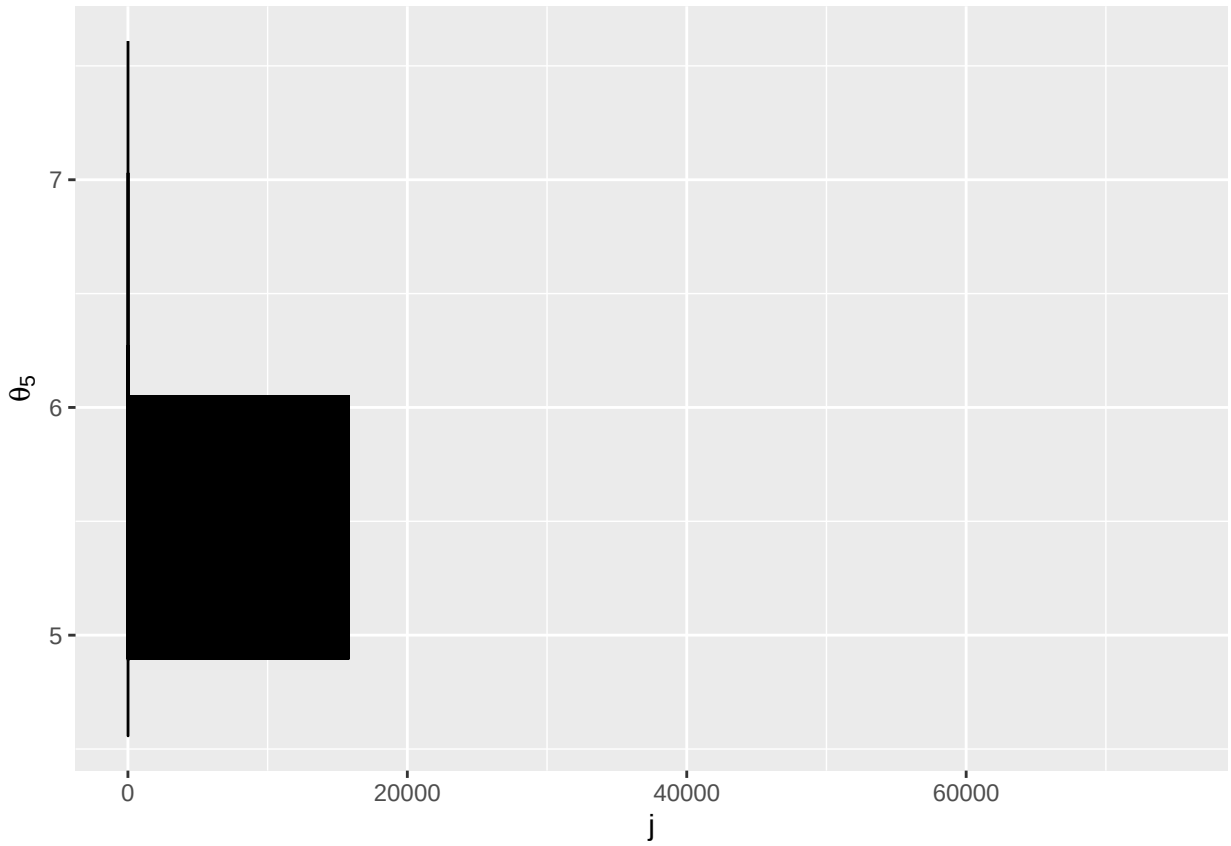
c.

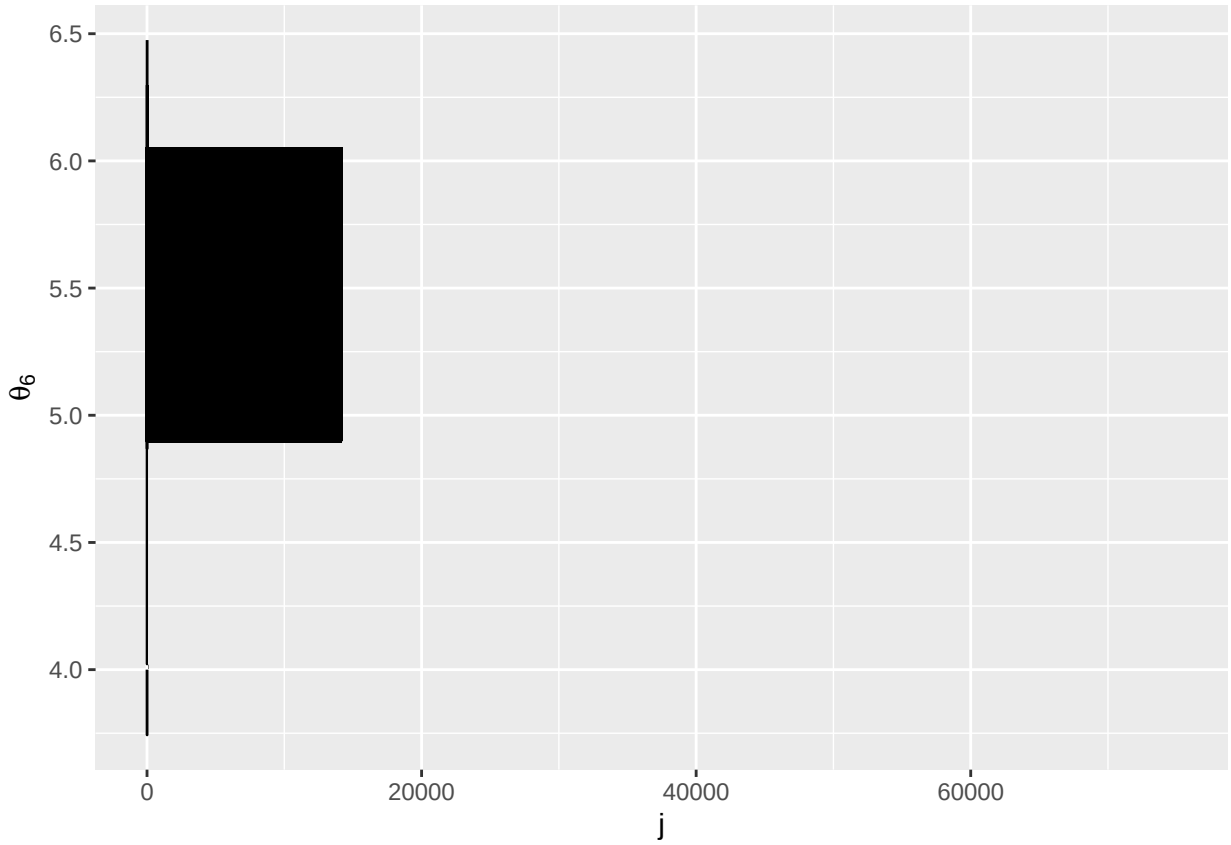

7
6
5
θ1
0
20000
40000
60000
j


6.0
5.5
5.0
4.5
4.0
θ2
0
20000
40000
60000
j


7
6
5
4
θ3
0
20000
40000
60000
j


6.0
5.5
5.0
4.5
4.0
θ4
0
20000
40000
60000
j



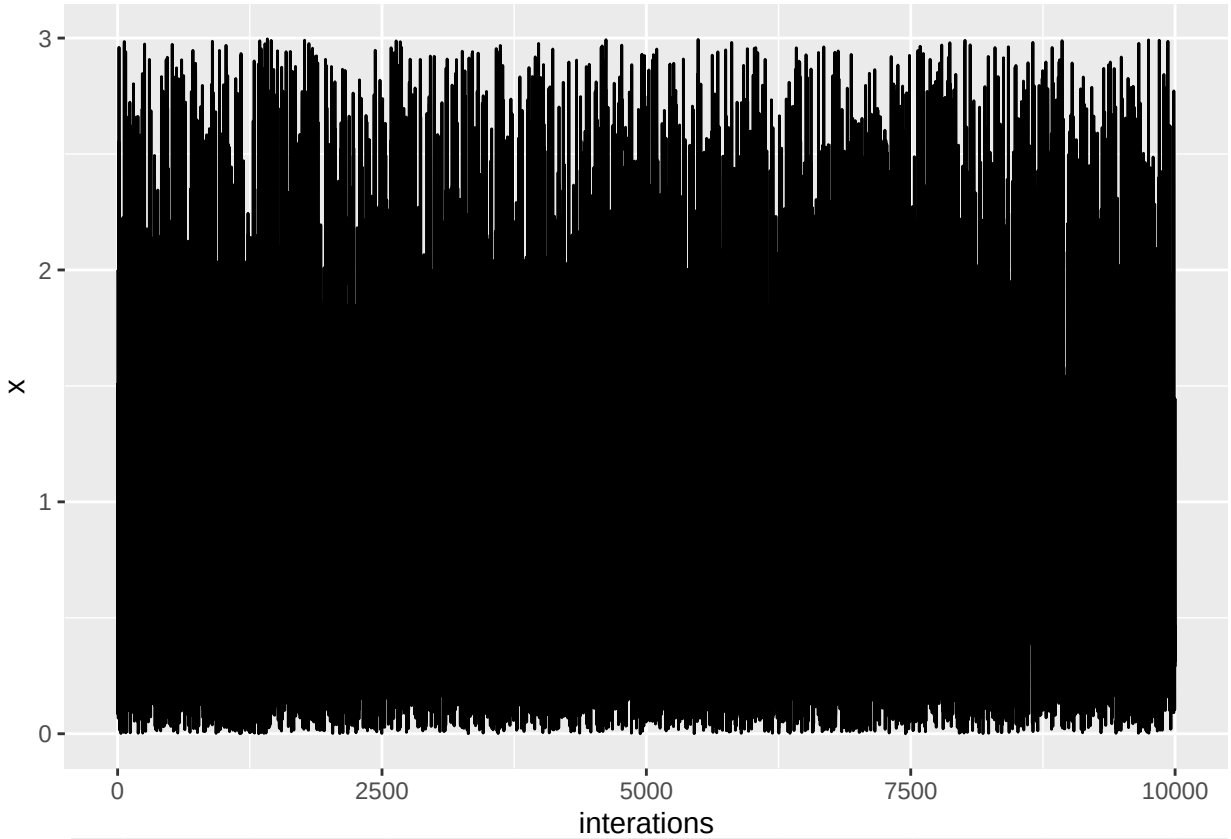


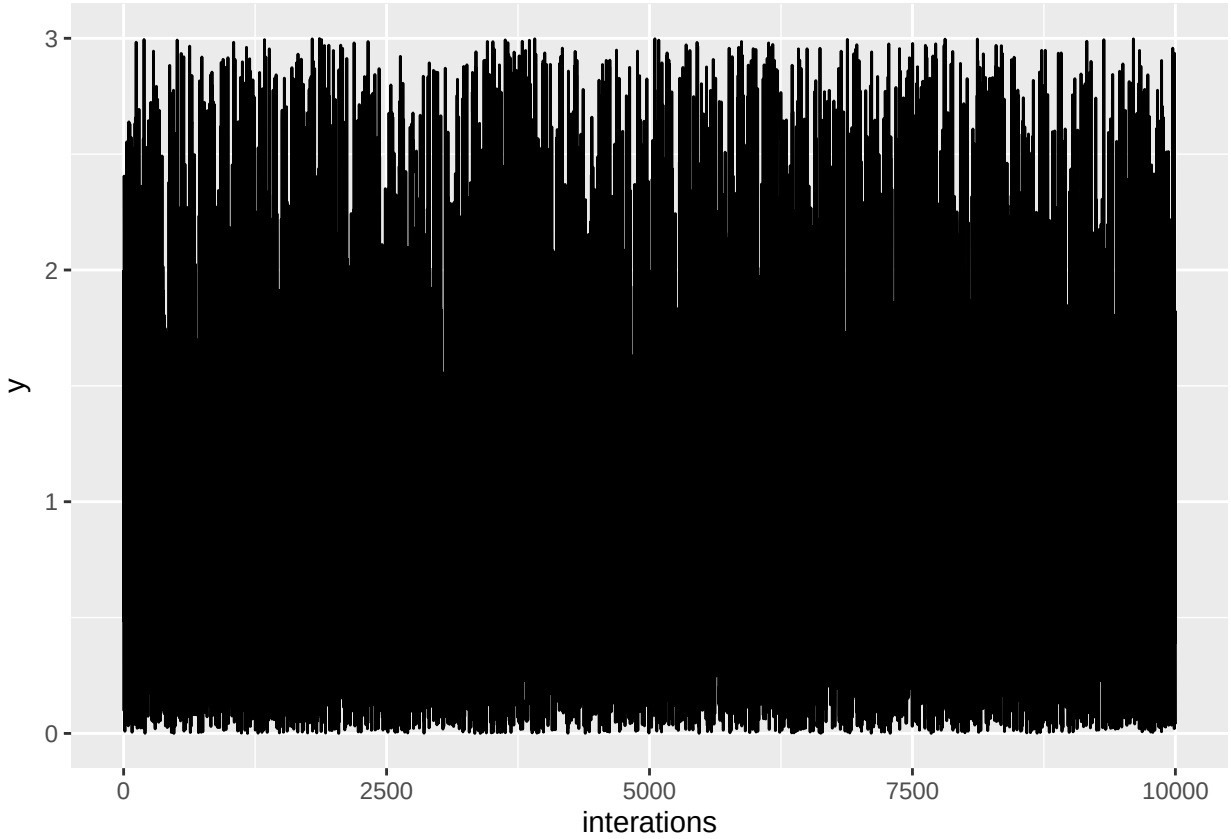
It violates Hobert-Casella conditions. Because of the Lack for prior information, all prior distributions are

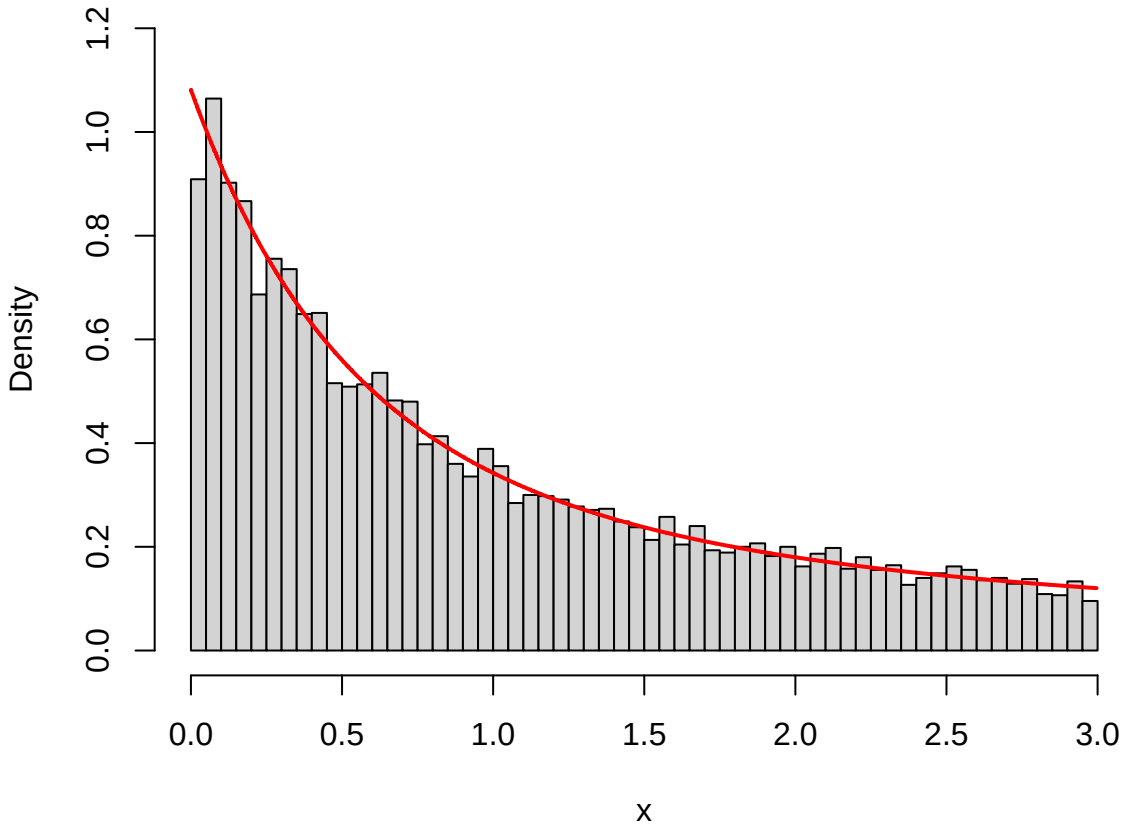
chosen to be uninformative, which leads to the disconvergence of the chains.

Question 5

a.

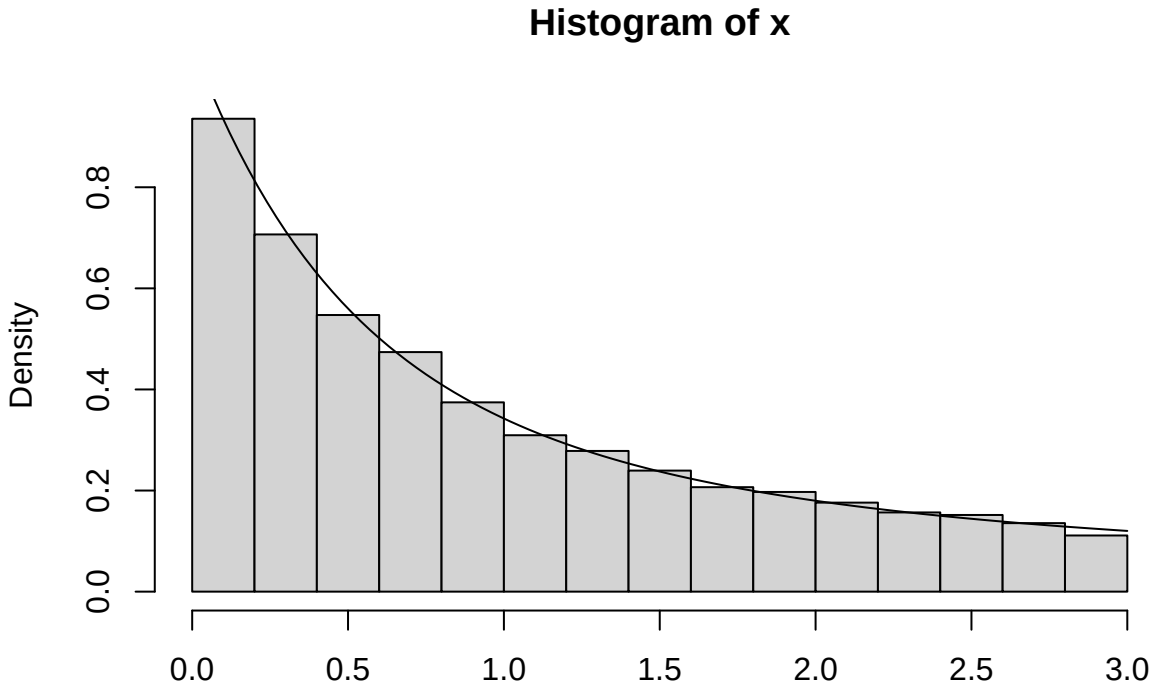

3
2
1
0
x
0
2500
5000
7500
10000
interations

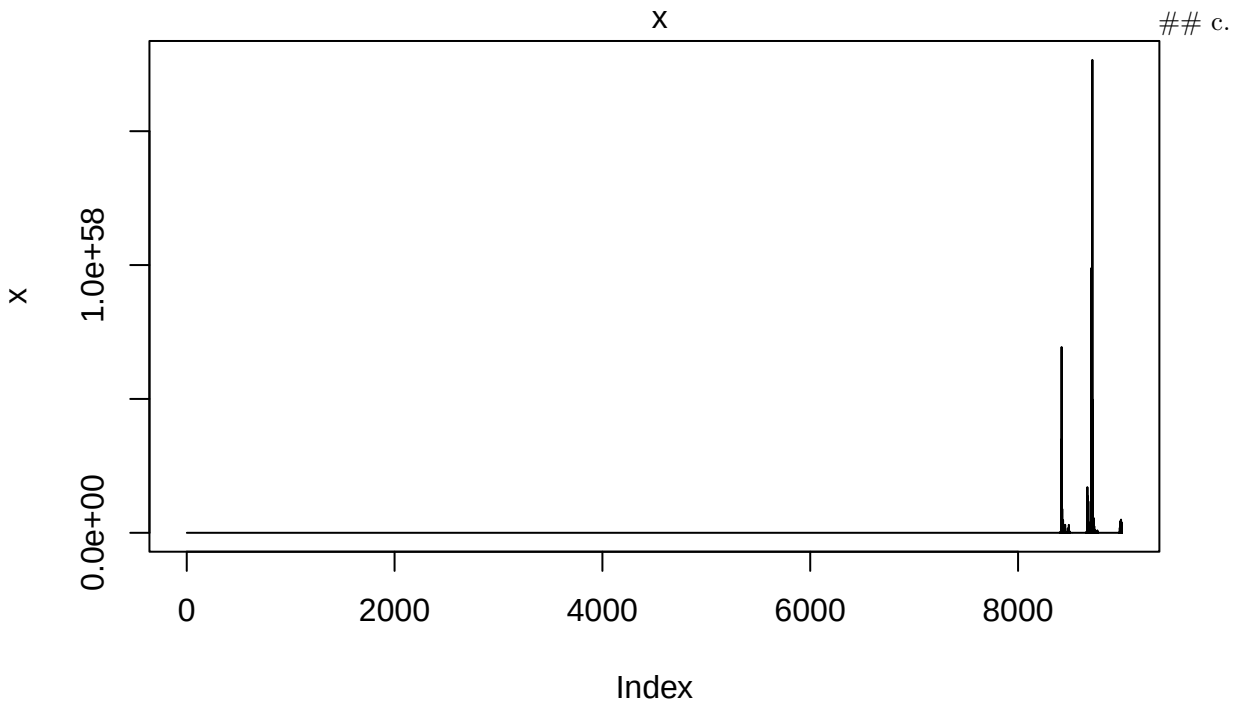

3
2
1
0
y
0
2500
5000
7500
10000
interations



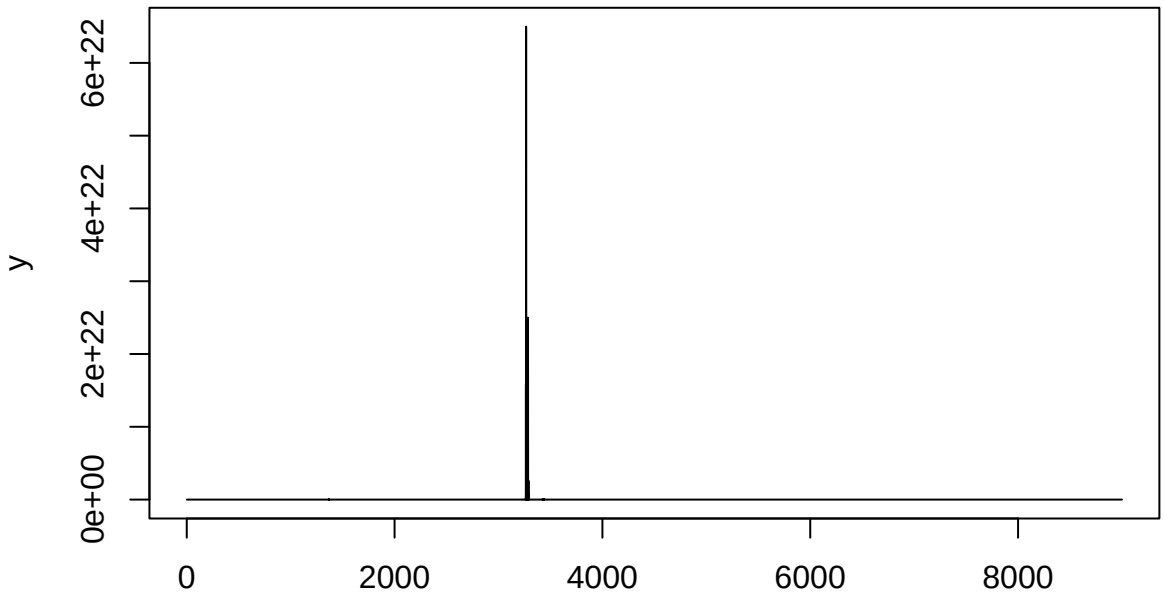
Evaluate the convergence using Galmen-Rubin method with R_hat close to 1 and less than 1.1, which means it converges well.

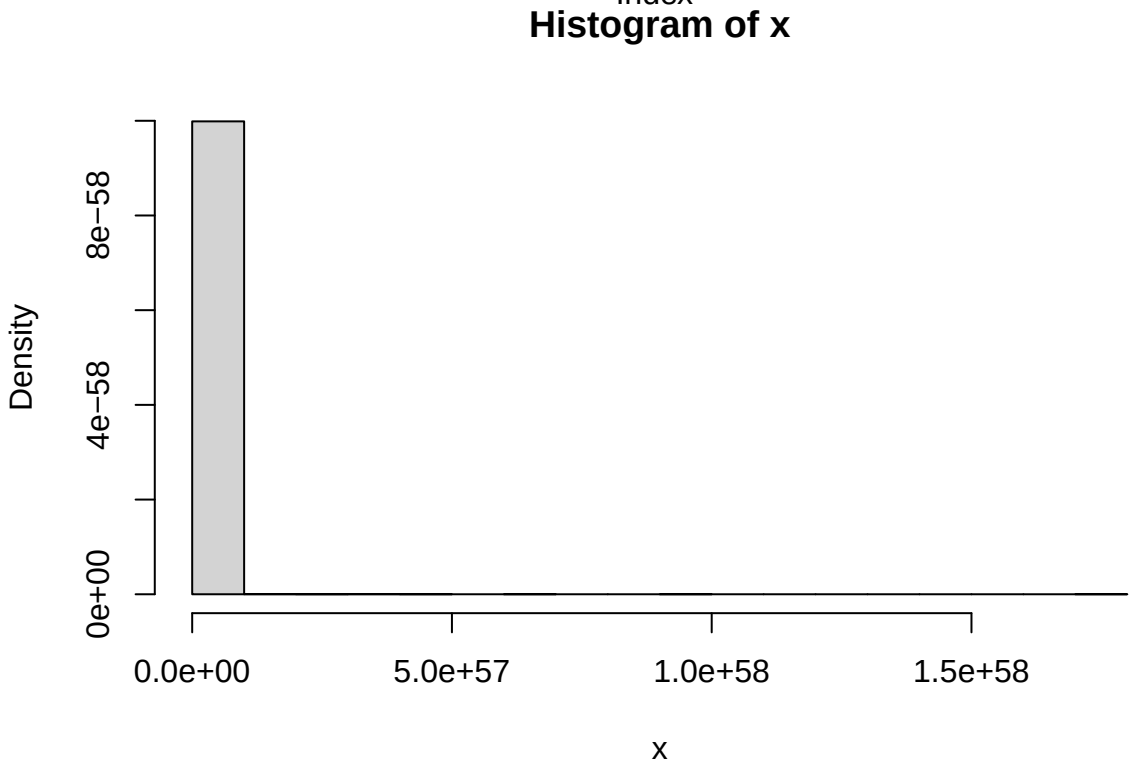
b.


Histogram of x
Density
0.0
0.2
0.4
0.6
0.8
0.0
0.5
1.0
1.5
2.0
2.5
3.0


x
x
0.0e+00
1.0e+58
0
2000
4000
6000
8000
Index

c.


y
0e+00
2e+22
4e+22
6e+22
0
2000
4000
6000
8000
Index


Histogram of x
Density
0e+00
4e−58
8e−58
0.0e+00
5.0e+57
1.0e+58
1.5e+58
x

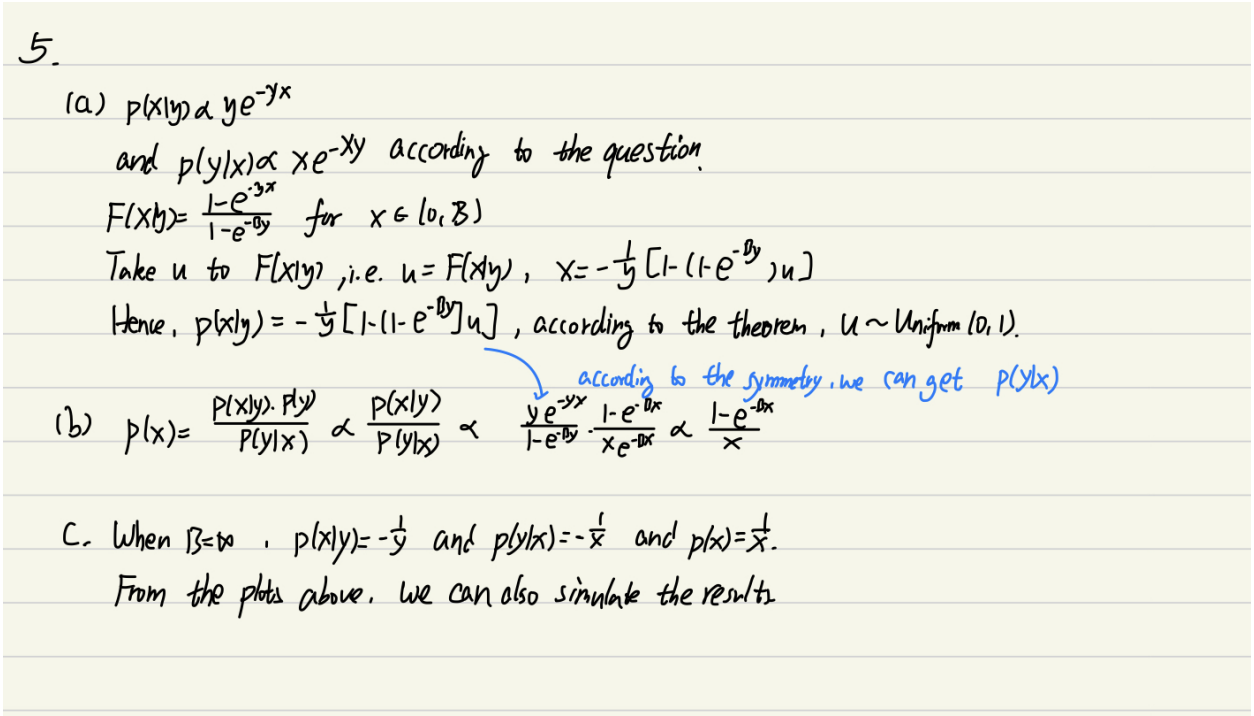


Figure 2: Question5