Group 12 Lab 4

varsi146, aswma317

2022-12-06

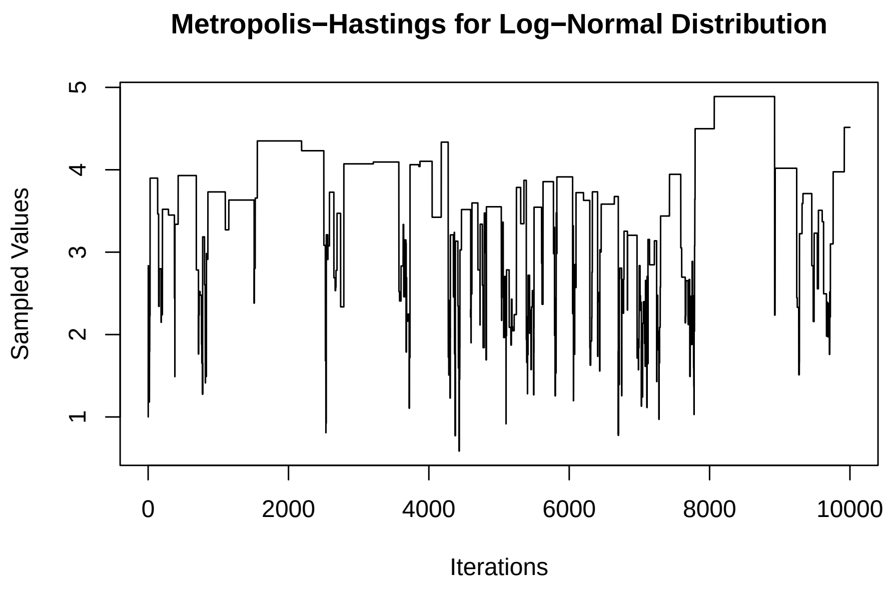
Question 1: Computations with Metropolis Hastings

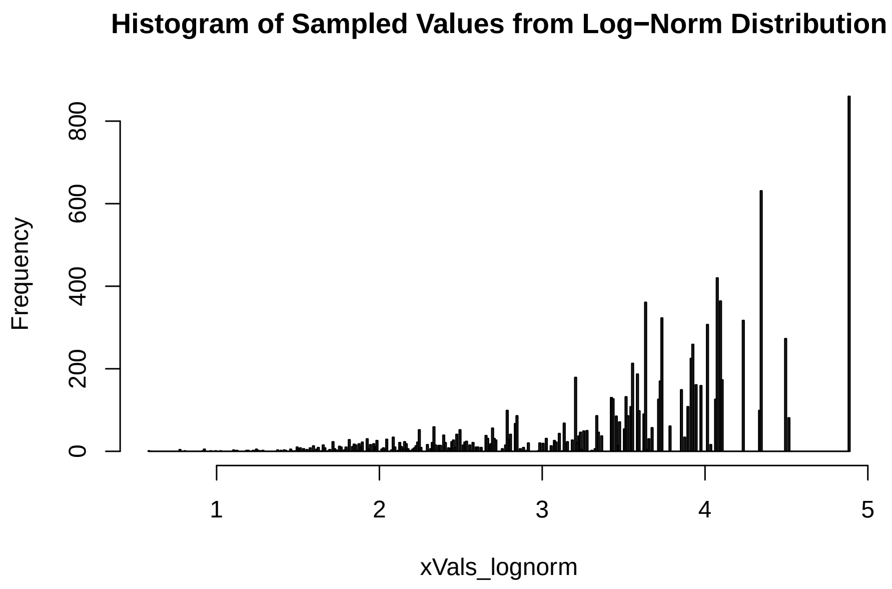
Sub Question 1

The given density function after including the constant of proportionality can be written as:

$$f(x) = \frac{x^5 \cdot e^{-x}}{120}, x > 0 \tag{1}$$

We then apply the Metropolis-Hastings algorithm to generate samples from the above distribution by using the log-normal, $LN(X_t, 1)$ as the proposal distribution. Below is the corresponding chain plotted as a time series plot:

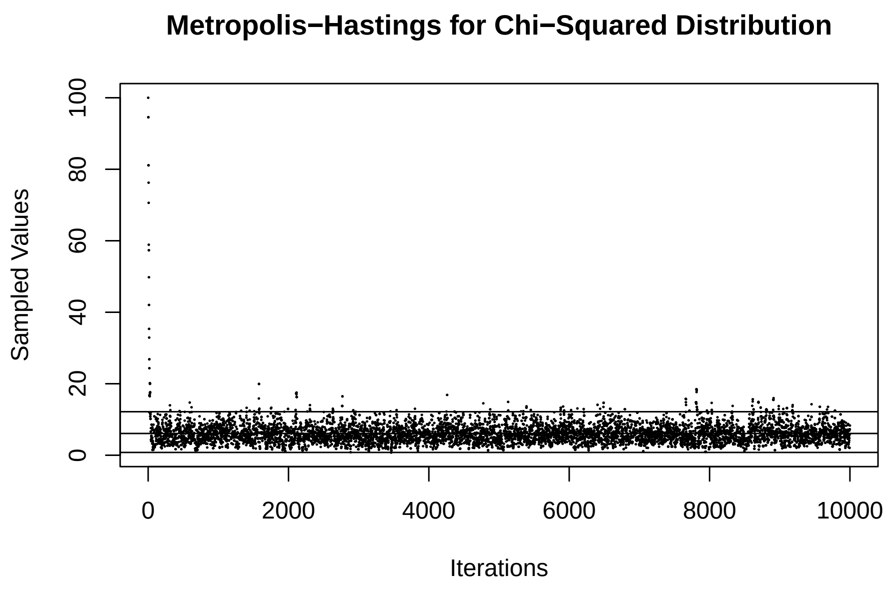


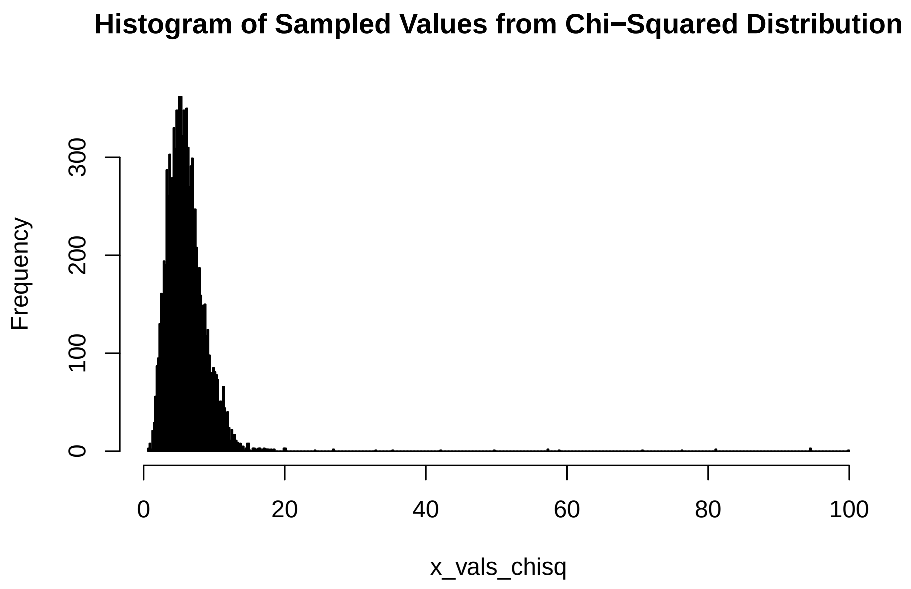


From the chain that is represented as a time series plot above, we can see that there is no convergence of the chain to a stationary distribution. We can then call the above chain as a non-stationary chain, and due to the absence of convergence to a stationary state, we cannot calculate the burn-in period. But according to Markov chain theory, it may converge at some point in time.

Sub Question 2

Now we are asked to use $\chi^2(\lfloor X_t + 1 \rfloor)$ as the proposal distribution in the Metropolis Hastings algorithm and generate samples from the target distribution. Below is the corresponding chain plotted as a time series plot:





Sub Question 3

It is evident from the above plots that when we use $\chi^2(\lfloor X_t + 1 \rfloor)$ as the proposal distribution the chain converges to a stationary distribution with a visible burn-in period that is, the transient state before it reaches the stationary distribution state.

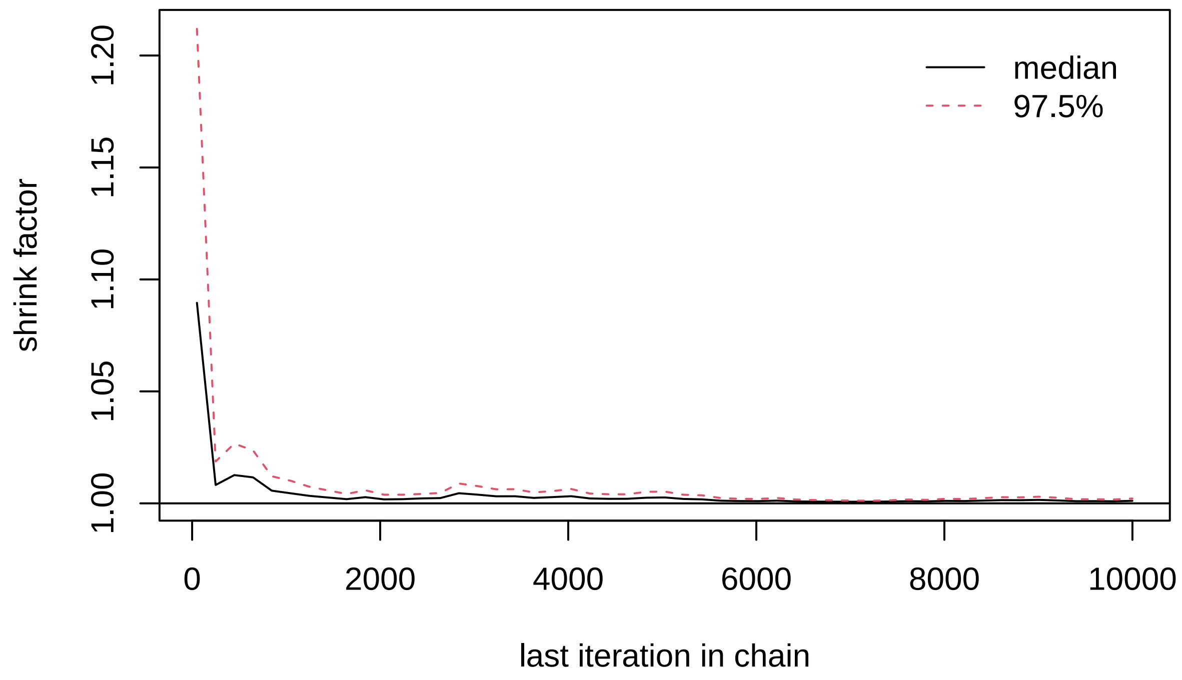
Hence, we can conclude that using $\chi^2(\lfloor X_t + 1 \rfloor)$ as the proposal distribution is a better choice as compared to the $LN(X_t, 1)$ distribution.

Sub Question 4

The below matrix is produced when a markov chain object is passed into the ***'gelman.diag'*** function that is the Gelman and Rubin convergence diagnostic. It gives the scale reduction factors for each variable along with upper and lower confidence limits. The function diagnoses an approximate convergence when the upper limit value is approximately equal to 1.

```
## Potential scale reduction factors:
##
##      Point est. Upper C.I.
## [1,]          1          1
```

The plot below shows the evolution of the scale reduction factor over time/iterations, and that as the iterations increase, the chains eventually converge to a stationary distribution.



Sub Question 5

Estimating the given integral , that is,

$$\int_0^\infty xf(x)dx$$

amounts to calculating the expected value of a given continuous function, that is,

And the expected value of a continuous random variable X is equal to its mean,

$$E(X) = \int_{-\infty}^{\infty} xf(x)dx = \mu$$

Therefore, the expected value of samples from the $LN(X_t, 1)$ distribution is:

```
## [1] 3.649884
```

The expected value of samples from the $\chi^2(\lfloor X_t + 1 \rfloor)$ distribution is:

```
## [1] 6.08567
```

Sub Question 6

The gamma function for continuous random variable X can be written as:

$$f(x) = [\frac{1}{\Gamma(\alpha)\beta^{\alpha}}]x^{\alpha-1}e^{-\frac{y}{\beta}};$$
$$0 < x < \infty$$

When we compare the Gamma distribution density function with our target distribution in our assignment, we can say that $\alpha = 6$ and $\beta = 1$.

Using the above values, we can find the value of constant that is:

$$[\frac{1}{\Gamma(\alpha)\beta^{\alpha}}] = [\frac{1}{\Gamma(6)\cdot 1^{6}}] = \frac{1}{120}$$

This is also the same constant of proportionality we use in our target distribution(equation 1) Furthermore, the expected value i.e, mean of a Gamma distribution density function can be written as:

$$E(X) = \mu = \alpha \cdot \beta$$

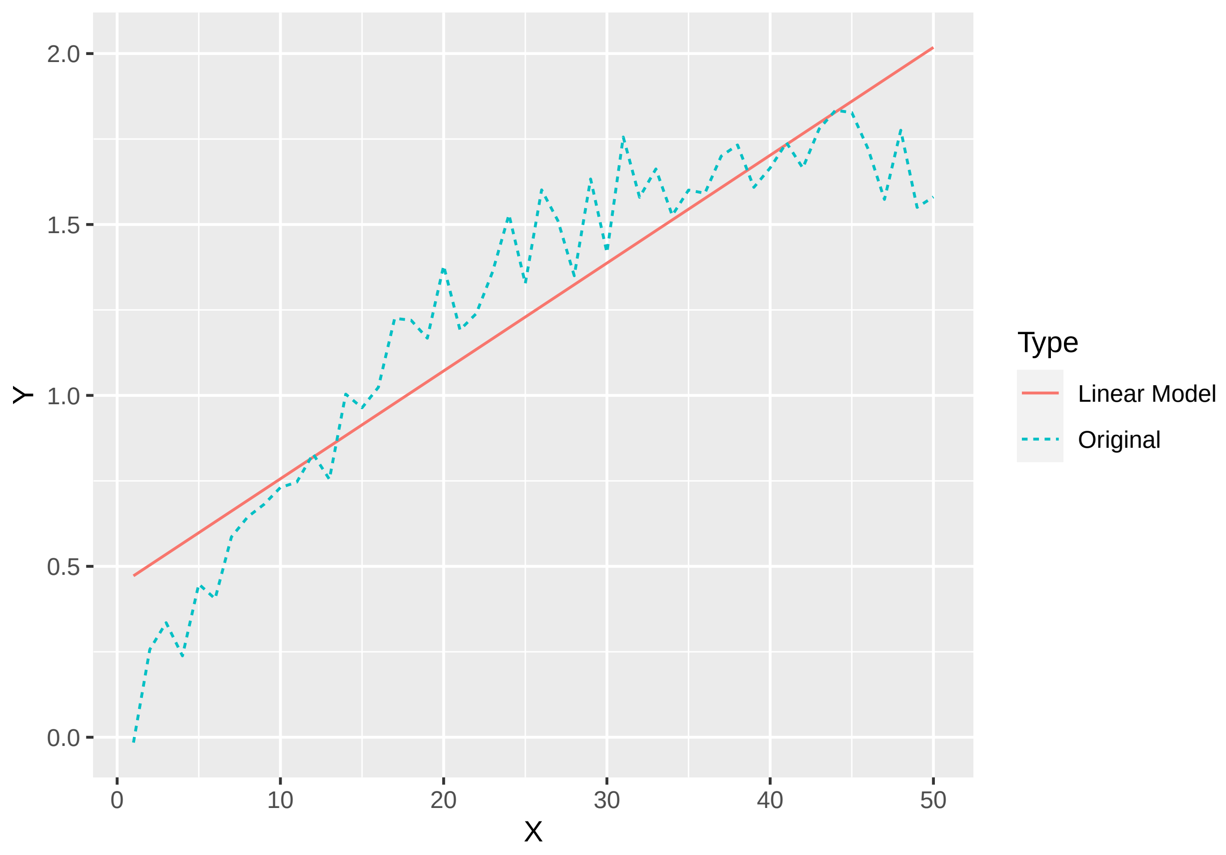
When we evaluate this, we get the mean, $\mu = 6$ and when we compare this value with the expected values of samples generated(see Sub Question 5) when $LN(X_t, 1)$ and $\chi^2(\lfloor X_t + 1 \rfloor)$ are chosen as proposal distributions, it is evident that, the Chi-squared distribution is a better choice of proposal distribution since value of mean is approximately equal to the mean of the gamma distribution.

Question 2: Gibbs sampling

Sub Question 1

Import the data to R and plot the dependence of Y on X. What kind of model is reasonable to use here?

This is a regression problem and since Y is linearly increasing with X, a reasonable model to start with would be a Linear Regression model.



Sub Question 2

Present the formulae showing the likelihood $p(\vec{Y}|\vec{\mu})$ and the prior $p(\vec{\mu})$

Since it is given that $Y_i \sim \mathsf{N}(\mu_i, var = 0.1), i = 1...n$, we can derive the likelihood as shown below:

$$p(\vec{Y}|\vec{\mu}) = \prod_{i=1}^{n} f(y_i), where f(y_i) = \frac{exp^{-\frac{(y-\mu)^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}$$

Substituing $\sigma = \sqrt{0.2}$, we get the likelihood as:

$$p(\vec{Y}|\vec{\mu}) = \frac{exp^{-\sum_{i=1}^{n} \frac{(y_i-\mu_i)^2}{0.04}}}{(\sqrt{0.4\pi})^n}$$

For finding the prior, we have can use the chain rule:

$$p(\vec{\mu}) = p(\mu_1)p(\mu_2|\mu_1)p(\mu_3|\mu_2)...p(\mu_n|\mu_{n-1})$$

where,

$$p(\mu_1) = 1$$

$$p(\mu_{i+1}|\mu) \sim \mathsf{N}(\mu_i, var = 0.1), i = 1...n$$

Applying this in the chain rule:

$$p(\vec{\mu}) = 1 * \frac{exp^{-\sum_{i=2}^{n} \frac{(\mu_i-\mu_{i-1})^2}{0.04}}}{(\sqrt{0.4\pi})^n}$$

Sub Question 3

Part a: Use Bayes' Theorem to get the posterior up to a constant proportionality

$$p(\vec{\mu}) \propto exp^{-\sum_{i=1}^{n} \frac{(y_i-\mu_i)^2}{0.04}} * exp^{-\sum_{i=2}^{n} \frac{(\mu_i-\mu_{i-1})^2}{0.04}}$$

Part b: Using the Posterior, find the the distributions for $(\mu_1|\vec{\mu}_{-1}, \vec{Y}), (\mu_n|\vec{\mu}_{-n}, \vec{Y})$ and $(\mu_i|\vec{\mu}_{i-i}, \vec{Y})$

These full conditional distibutions can be found from the posterior we have derived above.

For Case1: $(\mu_1|\vec{\mu}_{-1}, \vec{Y})$,

$$\begin{aligned} p(\mu_1|\vec{\mu}_{-1}, \vec{Y}) &\propto exp^{-\frac{(y_1-\mu_1)^2}{0.04}} * exp^{-\frac{(\mu_2-\mu_1)^2}{0.04}} \\ &\propto exp^{-\frac{(\mu_1-y_1)^2+(\mu_1-\mu_2)^2}{0.04}} \\ &\propto exp^{-\frac{(\mu_1-\frac{(y_1+\mu_2)}{2})^2}{0.04}} \end{aligned}$$

The above looks like a Normal distribution, hence:

$$p(\mu_1|\vec{\mu}_{-1}, \vec{Y}) \sim \mathsf{N}(\frac{(y_1+\mu_2)}{2}, 0.01)$$

For Case2: $(\mu_n|\vec{\mu}_{-n}, \vec{Y})$,

$$\begin{aligned} p(\mu_n|\vec{\mu}_{-n}, \vec{Y}) &\propto exp^{-\frac{(y_n-\mu_n)^2}{0.04}} * exp^{-\frac{(\mu_n-\mu_{n-1})^2}{0.04}} \\ &\propto exp^{-\frac{(\mu_n-y_n)^2}{0.04}} * exp^{-\frac{(\mu_n-\mu_{n-1})^2}{0.04}} \\ &\propto exp^{-\frac{(\mu_n-\frac{(y_n+\mu_{n-1})}{2})^2}{0.04}} \end{aligned}$$

The above looks like a Normal distribution, hence:

$$p(\mu_n|\vec{\mu}_{-n}, \vec{Y}) \sim \mathsf{N}(\frac{(y_n+\mu_{n-1})}{2}, 0.01)$$

For Case2: $(\mu_i|\vec{\mu}_{i-i}, \vec{Y})$,

$$p(\mu_i|\vec{\mu}_{i-i}, \vec{Y}) \propto exp^{-\frac{(y_i-\mu_i)^2}{0.04}} * exp^{-\frac{(\mu_i-\mu_{i-1})^2}{0.04}} * exp^{-\frac{(\mu_{i+1}-\mu_i)^2}{0.04}}$$

$$\propto exp^{-\frac{(\mu_i-y_i)^2}{0.04}} * exp^{-\frac{(\mu_i-\mu_{i-1})^2}{0.04}} * exp^{-\frac{(\mu_i-\mu_{i+1})^2}{0.04}}$$

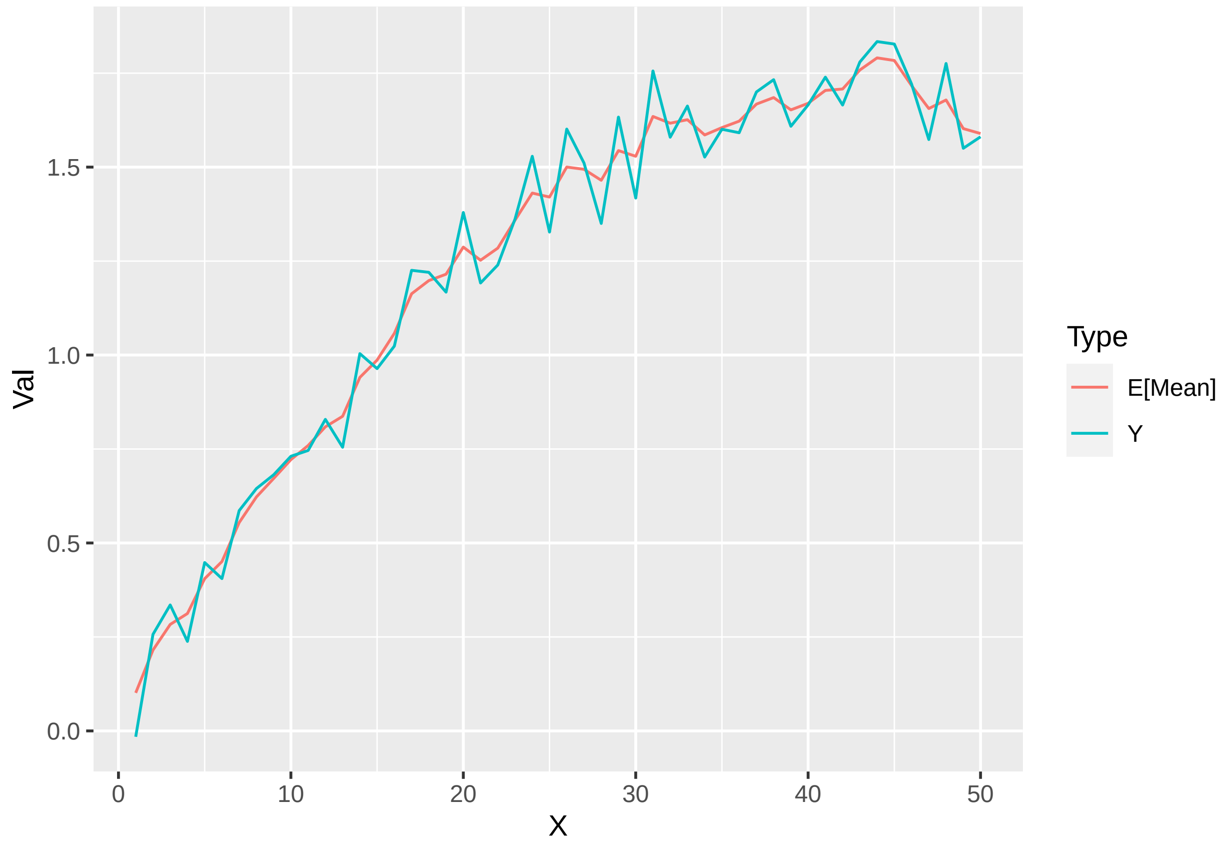
$$\propto exp^{-\frac{(\mu_i-\frac{y_i+\mu_{i-1}+\mu_{i+1}}{3})^2}{0.013}}$$

The above looks like a Normal distribution, hence:

$$p(\mu_i|\vec{\mu}_{i-i}, \vec{Y}) \sim \mathsf{N}(\frac{y_i+\mu_{i-1}+\mu_{i+1}}{3}, 0.006)$$

Sub Question 4

Part a: Run Gibbs sampler to get 1000 values for $\vec{\mu}$ and compute $E[\vec{\mu}]$. Plot $E[\vec{\mu}]$ vs X and Y vs X in the same graph.



Part b: Does it seem that you have managed to remove the noise?

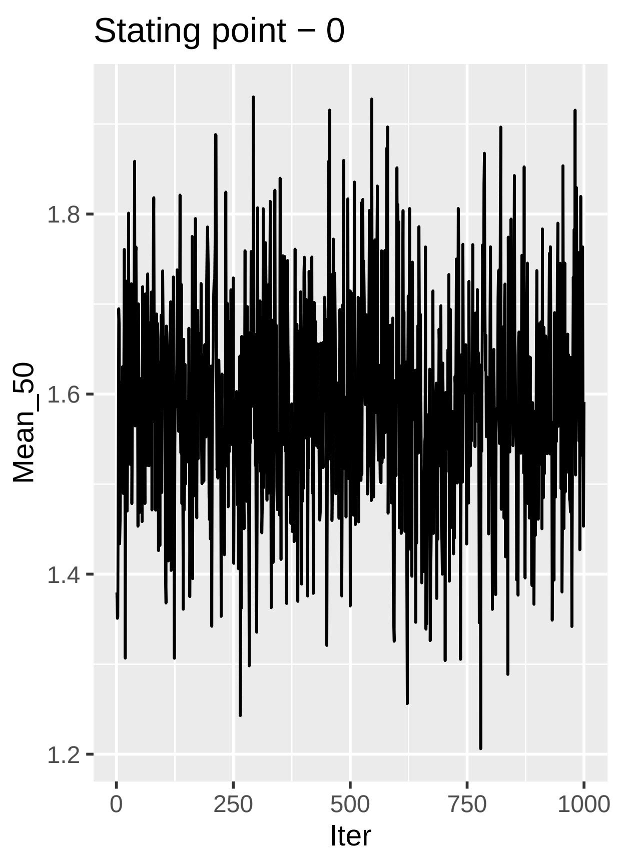
Yes. As it can be seen, the line representing Y is more jagged at some places, which represents the noise, while the line representing the $E[\vec{\mu}]$ is comparatively much smoother at these places. This is because the μ_i we calculated has a dependency on the μ before it and the one after it(the Markhov chain), except for the μ_1 & μ_{50} which depends only on the one adjacent to it. This helps in bringing down the noise in the data.

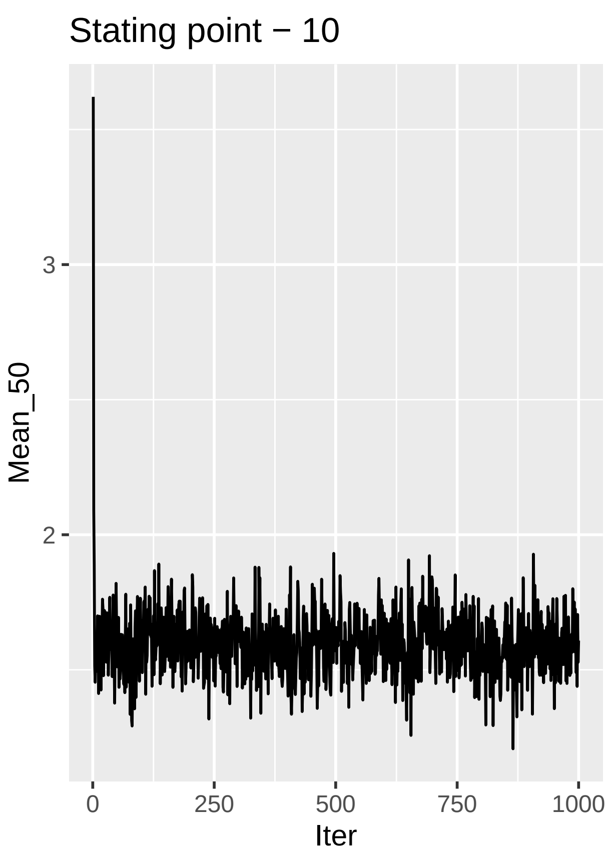
Part c: Does it seem that the expected value of $\vec{\mu}$ can catch the true underlying dependence between Y and X?

Yes. As seen in the graph, $E[\vec{\mu}]$ follows the line Y without any deviation and doesn't overfit as well.

Sub Question 5

Make a trace plot for $\vec{\mu_n}$ and comment on the burn–in period and convergence.





As seen in the graph, there is negligble burn-in when we start from 0 and the convergence to the stationary distrubition is immediate. When we change the starting point to 10, we see a couple of points before reaching the stationary, but this is negligible.

Note: I have not included the starting points as seen in the graph.

Appendix : All code for this report.

```
library(coda)
library(ggplot2)
library(gridExtra)
knitr::opts_chunk$set(echo = TRUE)
######################Code for Metropolis Hastings######################
# Target Distribution

f_x <- function(x){
  res <- ((x^5)*exp(-x))/120
  return(res)
}

# Log Normal as Proposal distribution in Metropolis Hastings algorithm

mc_mh_lognorm <- function(X0, tmax){
  range_t <- 1:tmax
  X_t_1 <- rep(X0, tmax)
```

```r
  for (i in 2:tmax) {
    X_t <- X_t_1[i-1]
    Y <- rlnorm(1, meanlog = X_t, sdlog = 1)
    U <- runif(1)
    acc_prob <- min(c(1,
                      (f_x(Y)*dlnorm(X_t,
                                     meanlog = Y,
                                     sdlog = 1))/(f_x(X_t)*dlnorm(Y,
                                                                  meanlog = X_t,
                                                                  sdlog = 1))))
    if (U <= acc_prob) {
      X_t_1[i] <- Y
    }else{
      X_t_1[i] <- X_t
    }
  }
  plot(range_t, X_t_1, type = 'l', xlab = 'Iterations',
       ylab = 'Sampled Values',
       main = 'Metropolis-Hastings for Log-Normal Distribution')
  return(X_t_1)
}
xVals_lognorm <- mc_mh_lognorm(X0 = 1, tmax = 10000)
hist(xVals_lognorm, breaks = 500,
     main = 'Histogram of Sampled Values from Log-Norm Distribution')
# Chi Squared as Proposal distribution in Metropolis hastings algorithm

mc_mh_chisq <- function(X0, tmax){
  range_t <- 1:tmax
  X_t_1 <- rep(X0, tmax)
  for (i in 2:tmax) {
    X_t <- X_t_1[i-1]
    Y <- rchisq(1, df = floor(X_t+1))
    U <- runif(1)
    acc_prob <- min(c(1,(f_x(Y)*dchisq(X_t,
                                       df = floor(Y+1)))/
                        (f_x(X_t)*dchisq(Y, df = floor(X_t+1)))))
    if (U <= acc_prob) {
      X_t_1[i] <- Y
    }else{
      X_t_1[i] <- X_t
    }
  }
  plot(range_t, X_t_1, pch = 19, cex = 0.1, xlab = 'Iterations',
       ylab = 'Sampled Values',
       main = 'Metropolis-Hastings for Chi-Squared Distribution')
  abline(h=mean(X_t_1))
  abline(h=min(X_t_1))
  abline(h=2*mean(X_t_1))
  return(X_t_1)
}
x_vals_chisq <- mc_mh_chisq(X0 = 100, tmax = 10000)
hist(x_vals_chisq, breaks = 500,
     main = 'Histogram of Sampled Values from Chi-Squared Distribution')
```

```r
# Sequence of starting values
X0_vals <- seq(1,10,1)
mc_list <- list()

# Creating a list of sequences for different starting points
for (i in 1:length(X0_vals)) {
  mc_list[[i]] <- mc_mh_chisq(X0 = X0_vals[i], tmax = 10000)
}

# Conversion of list type to mcmc type
mc_list <- lapply(1:length(mc_list), function(x){as.mcmc(mc_list[[x]])})
# plot(mc_list[[9]])

# Calculating gelman diag factors
gelman_factor <- gelman.diag(mc_list)
gelman_factor
gelman.plot(mc_list)
expected_value_lognorm <- mean(xVals_lognorm)
expected_value_lognorm
expected_value_chisq <- mean(x_vals_chisq)
expected_value_chisq
######################Code for Gibbs Sampling#####################
load("chemical.RData")
data <- data.frame(x = X, y = Y)
lin_mod <- lm(Y ~ X, data = data)
lin_mod <- lm(Y ~ X, data = data)
predict <- predict(lin_mod,newdata = data)
lin_mod_df <- rbind(data.frame(X = data$x, Y = data$y, Type = 'Original'),
      data.frame(X = data$x, Y = predict, Type = 'Linear Model'))
ggplot(lin_mod_df, aes(x = X, y = Y)) +
  geom_line(aes(color = Type, linetype = Type))
gibbs_sampler <- function(steps, initial_mu, data){
  mu_mat <- matrix(NA, nrow = steps, ncol = length(data$y))
  step_mu <- initial_mu
  y <- data$y

  for (itr in 1:steps) {
    #Calculate mu1
    step_mu[1] <- rnorm(1, mean = (y[1] + step_mu[2])/2,
                        sd = sqrt(0.01))
    #Calculate mu2...mu49
    for (i in 2:49) {
      step_mu[i] <- rnorm(1, mean = (y[i] + step_mu[i-1] + step_mu[i+1])/3,
                          sd = sqrt(0.006))
    }
    #Calculate mu50
    step_mu[50] <- rnorm(1, mean = (y[50] + step_mu[49])/2,
                         sd = sqrt(0.01))
    # cat('Iteration:' , itr)
    # print(step_mu)
    #Populate mu_mat
    mu_mat[itr,] <- step_mu
  }
  return(mu_mat)
}
#Run the Gibbs sample to get 1000 values of mu
mu_mat <- gibbs_sampler(steps = 1000, initial_mu = rep(0, length(data$y)),
              data = data)
e_mu <- apply(mu_mat, MARGIN = 2, FUN = mean)#Calculate the expected mean
plot_df <- matrix(nrow = 0, ncol = 3)
colnames(plot_df) <- c('X', 'Type', 'Val')
plot_df <- rbind(plot_df,
                 data.frame(X = data$x, Type = 'Y', Val = data$y),
                 data.frame(X = data$x, Type = 'E[Mean]', Val = e_mu)
                 )
plot1 <- ggplot(plot_df, aes(x = X, y = Val)) +
  geom_line(aes(color = Type))
plot1
#Trace plot
mu_mat <- as.data.frame(mu_mat)
mu_mat$Iter <- seq(from = 1, to = 1000, by = 1)
plot2 <- ggplot(mu_mat, aes(x = Iter, y = mu_mat[,50])) +
  geom_line() +
  ylab('Mean_50') +
  ggtitle('Stating point - 0')
#Comment on the burn-in period & convergence
mu_mat_1 <- gibbs_sampler(steps = 1000, initial_mu = rep(10, length(data$y)),
                        data = data)
mu_mat_1 <- as.data.frame(mu_mat_1)
mu_mat_1$Iter <- seq(from = 1, to = 1000, by = 1)
plot3 <- ggplot(mu_mat_1, aes(x = Iter, y = mu_mat_1[,50])) +
  geom_line() +
  ylab('Mean_50') +
  ggtitle('Stating point - 10')
grid.arrange(plot2, plot3, ncol = 2)
```