# computational_statistic lab4

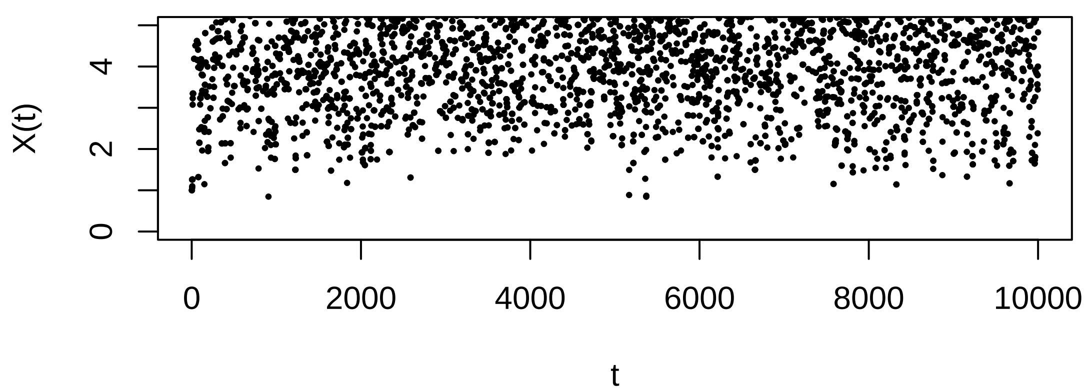
Group 8

11/21/2021

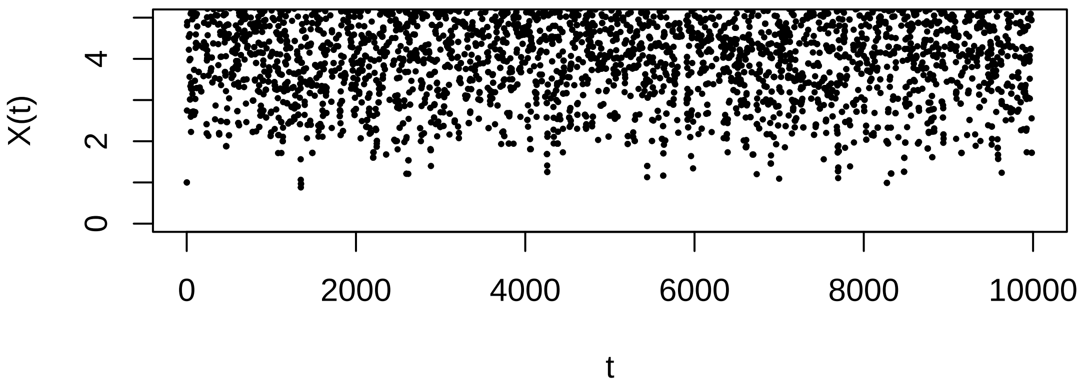
## Question 1 Computations with Metropolis–Hastings

**Q1:** Use Metropolis–Hastings algorithm to generate samples from this distribution by using proposal distribution as log–normal LN($X_t$, 1), take some starting point. Plot the chain you obtained as a time series plot. What can you guess about the convergence of the chain?
Solution :

The plot*(start points 1)* are not converged.




**Q2:** Perform Step 1 by using the chi-square distribution as a proposal distribution. What can you guess about the convergence of the chain?
Solution :




when $X_t < 1$, the plot will converge, when $x > 1$, plot will not converge. The start point is the same question1.

**Q3:** Generate 10 MCMC sequences using the generator from Step 2 and starting points 1, 2, . . . , or 10. Use the Gelman–Rubin method to analyze convergence of these sequences.
Solution :

```
## Potential scale reduction factors:
##
##      Point est. Upper C.I.
## [1,]       1.01       1.02
```

the upper limit is less than 1.2.so the chains are converged

**Q4:** Estimate $\int_0^\infty xf(x)dx$

solution :

The sample from step 1 and 2 are stored in q1_chains and q2_chains,we are going to use the theory from course slides. Devide f(x) into two parts : $f(x) = g(x) * p(x)$.

Then let p(x) be the target density function.Since our generated sample is in the form of target density function, so X ~ p(x) and integral p(x) = 1. As a result, what we need to compute is actually the estimation of g(x), so what we need to do is using mean() function to get sample estimation.

```
## [1] "for sample from question1: 6.04488796129457"
```

```
## [1] "for sample from question1: 6.02347759332851"
```

**Q5:** The distribution generated is in fact a gamma distribution (6, 1). Determine the actual value of the integral. Compare it with the one you obtained in the previous step.

solution:

The actual value of integral is estimation of gamma(6,1), which equals 6. From the results presented above, they are very closed to 6.

## Question 2 Gibbs sampling

**Q1:** Present the formula.

solution:

$$Y_i = \mathcal{N}(\mu, \sigma = 0.2), \quad i = 1, ..., n$$

where the prior is

$$p(\mu_1) = 1$$
$$p(\mu_{i+1} \mid \mu_i) = \mathcal{N}(\mu_i, 0.2) \quad i = 1, ..., n-1$$

$p(\vec{Y} \mid \vec{\mu})$ and $p(\vec{\mu})$ are:

$$\begin{aligned}
\mathcal{L}[p(\vec{Y}|\vec{\mu}, 0.2)] = &\prod_{i=1}^{n} \frac{1}{\sqrt{0.4\pi}} \exp(-\frac{(y_i-\mu_i)^2}{0.4}) \\
=&\Big(\frac{1}{\sqrt{0.4\pi}}\Big)^n \exp\Big(-\frac{\sum_{i=1}^{n}(y_i-\mu_i)^2}{0.4}\Big)
\end{aligned}$$

$$\begin{aligned}
p(\vec{\mu}) &= p(\mu_1)\cdot p(\mu_2|\mu_1)\cdot p(\mu_3|\mu_2)\cdots p(\mu_n|\mu_{n-1}) \\
&= 1\cdot\prod_{i=2}^{n} p(\mu_n|\mu_{n-1}) \\
&= \frac{1}{\sqrt{0.4\pi}}\exp(-\frac{(\mu_2-\mu_1)^2}{0.4})\cdots\exp(-\frac{(\mu_n-\mu_{n-1})^2}{0.4}) \\
&=\Big(\frac{1}{\sqrt{0.4\pi}}\Big)^{n-1}\exp(-\frac{\sum_{i=2}^{n}(\mu_i-\mu_{i-1})^2}{0.4})
\end{aligned}$$

**Q2:** Bayes's theorem.

solution:

$$p(\mu_1|\mu_{-1}, \vec{Y}) = \frac{p(\mu, Y)}{p(\mu_{-1}, \vec{Y})} = \frac{p(\mu, Y)}{p(\vec{Y}|\mu_{-1}) * p(\mu_{-1})}$$

So,

$$\begin{aligned}
p(\mu_1|\vec{\mu}_{-1}, \vec{Y}) = &\ \frac{p(\vec{\mu}, \vec{Y})}{p(\vec{\mu}_{-1}, \vec{Y})} \\
\propto &\ \exp\Big(-\frac{(y_1-\mu_1)^2+(\mu_2-\mu_1)^2}{2\sigma^2}) \\
\propto &\ \exp(-\frac{(\mu_1-(y_1+\mu_2)/2)^2}{2\sigma^2/2}) \ \ (\textit{this is accorind to the pdf})
\end{aligned}$$

$$p(\mu_n|\mu_{-n}, \vec{Y}) = \frac{p(\mu, Y)}{p(\mu_{-n}, \vec{Y})} = \frac{p(\mu, Y)}{p(\vec{Y}|\mu_{-n}) * p(\mu_{-n})}$$

Similar to $\mu_1$, we can compute that :

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

So, we can see this process is actually discard some part within 'exp'. For i(not 1 or n), the part need to be discarded are $(\mu_{i+1}-\mu_i)^2+(\mu_i-\mu_{i-1})^2$.As a result, for i(not 1 or n), we have:

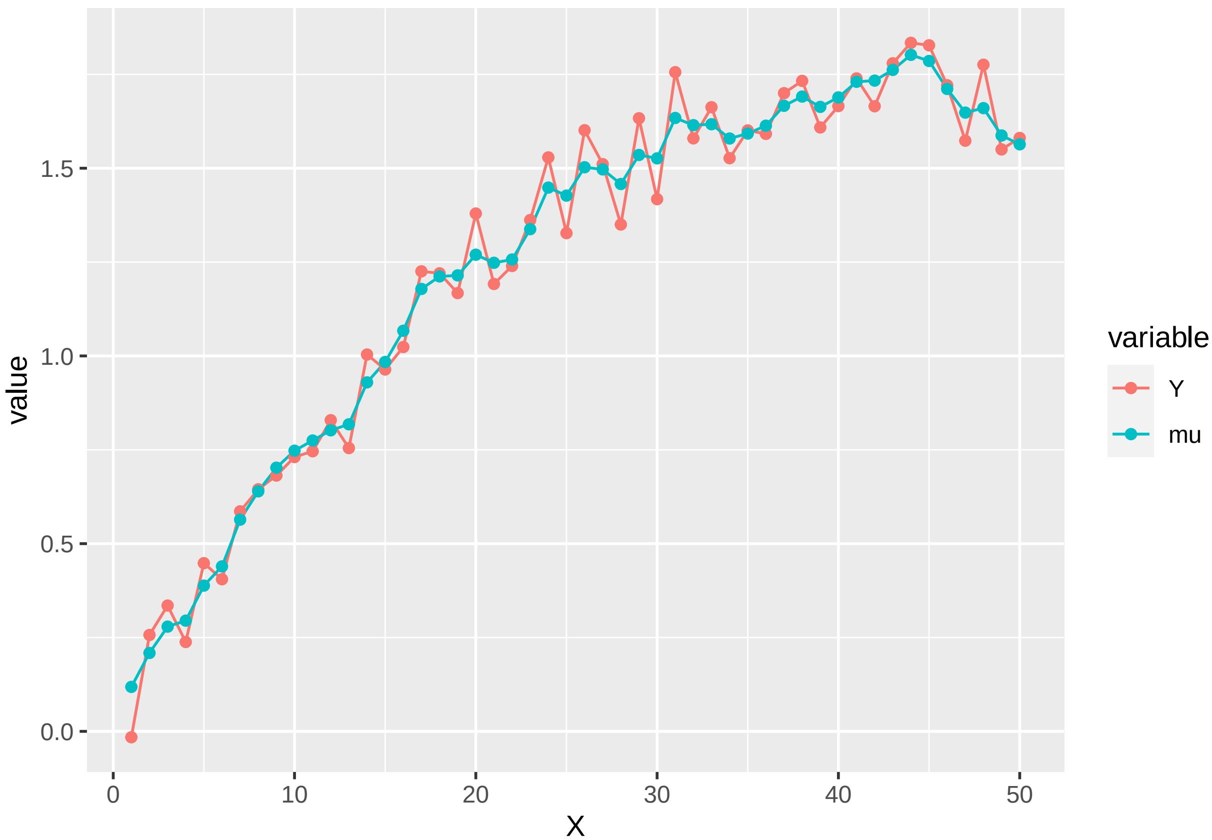
$$\begin{aligned} p(\mu_i|\vec{\mu}_{-i},\vec{Y}) = & \frac{p(\vec{\mu},\vec{Y})}{p(\vec{\mu}_{-i},\vec{Y})} \\ \propto & \exp\Big(-\frac{(y_i-\mu_i)^2+(\mu_{i+1}-\mu_i)^2+(\mu_i-\mu_{i-1})^2}{2\sigma^2}\Big) \\ \propto & \exp(-\frac{(\mu_i-(y_i+\mu_{i-1}+\mu_{i+1})/3)^2}{2\sigma^2/3}), \quad i\in(1,n). \end{aligned}$$

Finally, we find the exp part is similar to normal distribution. So, we conclude that:

$$(\mu_i|\vec{\mu}_{-i},\vec{Y}) \begin{cases} N(\frac{y_1+\mu_2}{2},0.1) & i=1 \\ N(\frac{y_i+\mu_{i-1}+\mu_{i+1}}{3},\frac{0.2}{3}) & Other \\ N(\frac{y_n+\mu_{n-1}}{2},0.1) & i=n \end{cases}$$

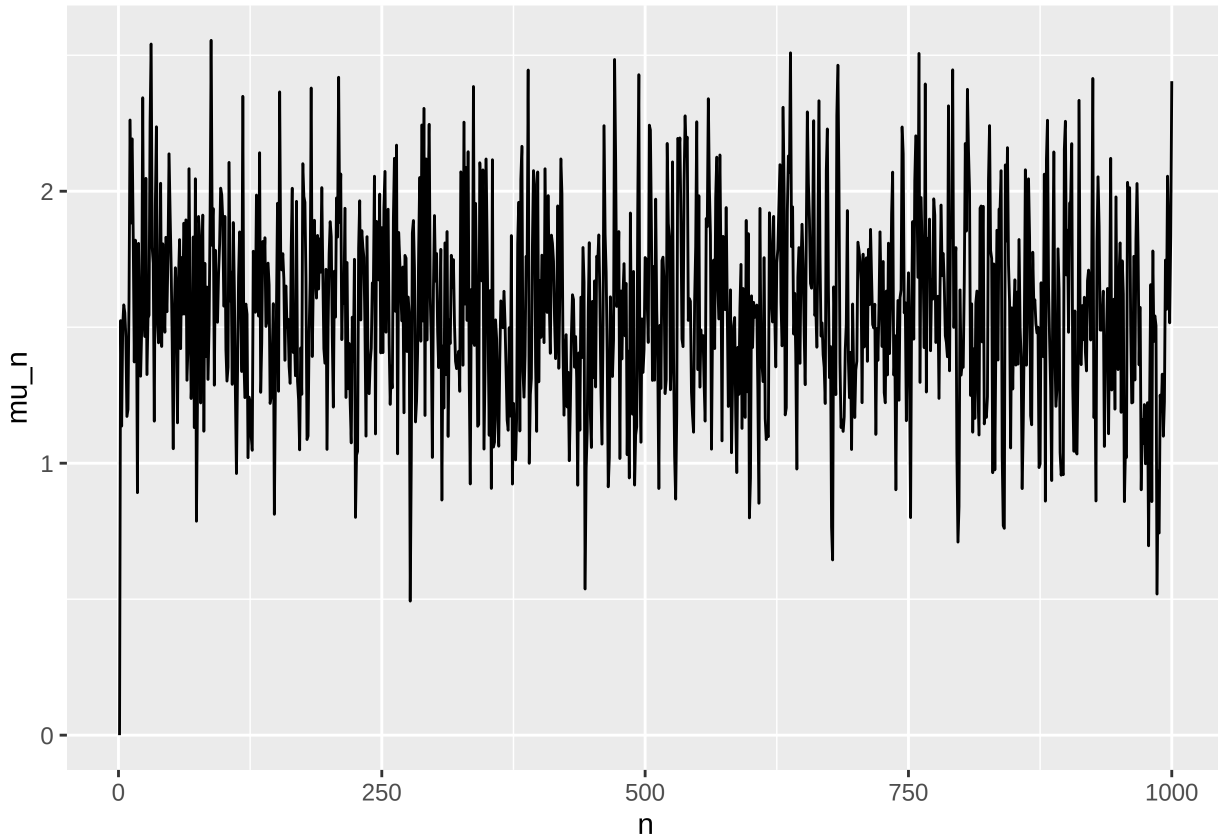
**Q3:** implement Gibbs sampler and Monte Carlo approach.
solution:

The curve seems to be more smooth so the noise should be removed. $\vec{\mu}$ can catch underlying dependence between Y and X.




**Q4:** Make a trace plot for $\mu_n$ solution: burn-in period is very short at the beginning, then the curve converges.

solutions:




**Appendix:**
Some code related to output are like '## expression'

```
library(coda)
library(ggplot2)
library(reshape2)
##################################################################################
#Question 1
##################################################################################
#q1
##################################################################################
pix <- function(x)
{
  return(x^5*exp(-x))
}
fMH_q1<-function(nstep,X0,props){
  vN<-1:nstep
  vX<-rep(X0,nstep);
  for (i in 2:nstep){
    X<-vX[i-1]
    Y<-rlnorm(1,meanlog=log(X),sdlog =props)
    u<-runif(1)
    a<-min(c(1,(pix(Y)*dlnorm(X,meanlog=log(Y),sdlog=props))/(pix(X)*dlnorm(Y,meanlog=log(X),sdlog=1))))
    if (u <=a){vX[i]<-Y}else{vX[i]<-X}
  }
 ## plot(vN,vX,pch=19,cex=0.3,col="black",xlab="t",ylab="X(t)",main="",ylim=c(min(X0-0.5,0),max(5,X0+0..
```

```
  return(vX)
}
t <- 1
q1_chains <- fMH_q1(10000,1,1)
rm(t)
################################################################################
# q2
################################################################################
fMH_q2<-function(nstep,X0){
  vN<-1:nstep
  vX<-rep(X0,nstep);
  for (i in 2:nstep){
    X<-vX[i-1]
    Y<-rchisq(1,df=floor(X+1))
    u<-runif(1)
    a<-min(c(1,(pix(Y)*dchisq(X,df=floor(Y+1)))/(pix(X)*dchisq(Y,df=floor(X+1)))))
    if (u <=a){vX[i]<-Y}else{vX[i]<-X}
  }
  return(vX)
}
q2_chains <- fMH_q2(10000,1)
## plot(1:10000,q2_chains,pch=19,cex=0.3,col="black",xlab="t",ylab="X(t)",main="",ylim=c(min(1-0.5,0),ma
################################################################################
# q3
################################################################################
mc_list <- mcmc.list()
for (i in 1:10)
{
  fmh_vX <- fMH_q2(1000,i)
  mc_list[[i]] <- as.mcmc(fmh_vX)
}
## gelman.diag(mc_list)
################################################################################
# q4
################################################################################
mean_1 <- mean(q1_chains)
mean_2 <- mean(q2_chains)
## print(paste0('for sample from question1: ',mean_1))
## print(paste0('for sample from question1: ',mean_2))
################################################################################
# Q2 Gibbs sampling
################################################################################
#q3
################################################################################
load('chemical.RData')
nstep <- 1000
len <- length(Y)
m0 <- rep(0,len)
gibbs <- function(nstep,Y,m0)
{
  n <- length(m0)
  gbs_matrix_mu <- matrix(0, nrow=nstep, ncol=n)
  gbs_matrix_mu[1,] <- m0
```

```
  for (t in 2:nstep)
  {
      gbs_matrix_mu[t,1] <- rnorm(1,(Y[1]+gbs_matrix_mu[t-1,2])/2,sqrt(0.1))
      for (i in 2:(n-1))
      {
        gbs_matrix_mu[t,i] <- rnorm(1,(Y[i]+gbs_matrix_mu[t,i-1]+gbs_matrix_mu[t-1,i+1])/3,sqrt(0.2/3))
      }
      gbs_matrix_mu[t,n] <- rnorm(1,(Y[n]+gbs_matrix_mu[t,n-1])/2,sqrt(0.1))
  }
  return(gbs_matrix_mu)
}
sample_mu <- gibbs(nstep = nstep,Y=Y,m0 = m0)
for_plot <- data.frame(X=X,Y=Y,mu=colMeans(sample_mu))
for_plot_melt <- reshape2::melt(for_plot,id='X')
##ggplot(for_plot_melt,aes(x=X,y=value,color=variable))+
##  geom_line()+
##  geom_point()
################################################################################
#q4
################################################################################
plot_mu <- data.frame(n=1:1000,mu_n=sample_mu[,50])
## ggplot(plot_mu,aes(x=n,y=mu_n))+
## geom_line()
```