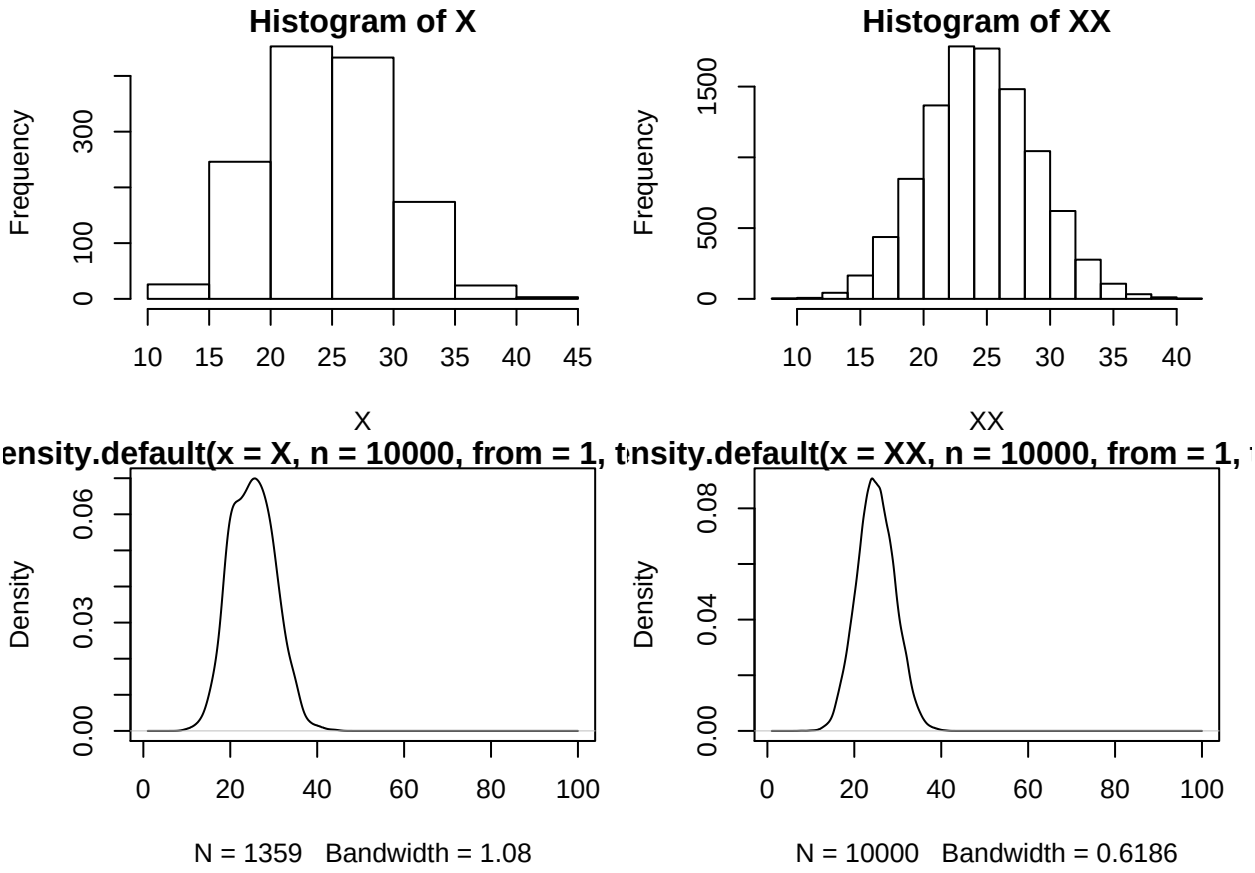
Assessment1-ST5222

Name: Zhu Xu ; User ID: E0337988 ; Student ID: A0191344H

15 September 2018

proposal–Unif , Binomial distribution(n=100, p=0.25)

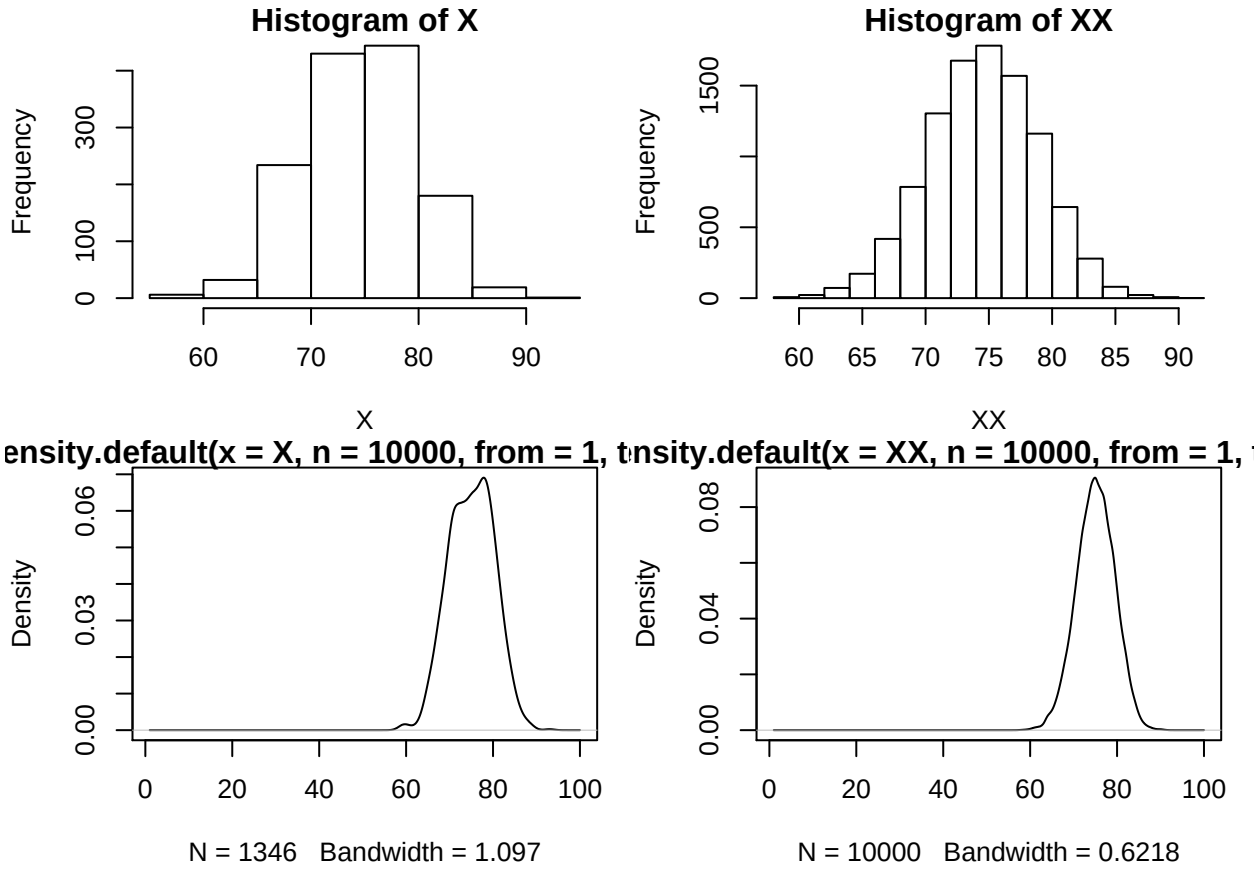
```
f_0 <- function(x){(factorial(100)/(factorial(x)*factorial(100-x)))*0.25^(x)*(1-0.25)^(100-x)}
# define the Binomial distribution , n=100, p=0.25
q_0 <- function(x){1/100}
# define the proposal function q_0 (Uniform)
n <- 100
X <- numeric() # store samples
count <- 1
x_old <- 0 # initialize the Markov chain
for(i in 1:10000) # main loop to obtain samples
{
  x_new <- sample.int(101, 1)-1 # sample a integer from q_0, as a candidate
  u <- runif(1,0,1)
  acc_1 <- min((f_0(x_new)*q_0(x_old))/(f_0(x_old)*q_0(x_new)), 1)
# calculate the acceptance prob
  if(u < acc_1) # accept
  {
    X[count] <- x_new
    count <- count +1
    x_old <- x_new
  }
  else # reject the sample
  {
    x_old <- x_old
  }
}
par(mfrow = c(2, 2), mar = c(4, 4, 1, 1))
hist(X) # display histogram of accepted samples
XX <- rbinom(10000, 100, 0.25)
hist(XX)
P <- density(X, n = 10000, from = 1, to  = 100)
plot(P)
B <- density(XX, n = 10000, from = 1, to  = 100)
plot(B)
```



proposal–Unif , Binomial distribution(n=100, p=0.75)

```
f_2 <- function(x){(factorial(100)/(factorial(x)*factorial(100-x)))*0.75^(x)*(1-0.75)^(100-x)}
# define the Binomial distribution , n=100, p=0.75
q_0 <- function(x){1/100}
# define the proposal function q_0 (Uniform)
n <- 100
X <- numeric() # store samples
count <- 1
x_old <- 0 # initialize the Markov chain
for(i in 1:10000)
{
  x_new <- sample.int(101, 1)-1 # sample a integer from q_0, as a candidate
  u <- runif(1,0,1)
  acc_1 <- min((f_2(x_new)*q_0(x_old))/(f_2(x_old)*q_0(x_new)), 1)
# calculate the acceptance prob
  if(u < acc_1) # accept
  {
    X[count] <- x_new
    count <- count +1
    x_old <- x_new
  }
  else # reject
  {
    x_old <- x_old
```

```
  }
}
par(mfrow = c(2, 2), mar = c(4, 4, 1, 1))
hist(X) # display histogram of accepted samples
XX <- rbinom(10000, 100, 0.75)
hist(XX)
P <- density(X, n = 10000, from = 1, to  = 100)
plot(P)
B <- density(XX, n = 10000, from = 1, to  = 100)
plot(B)
```

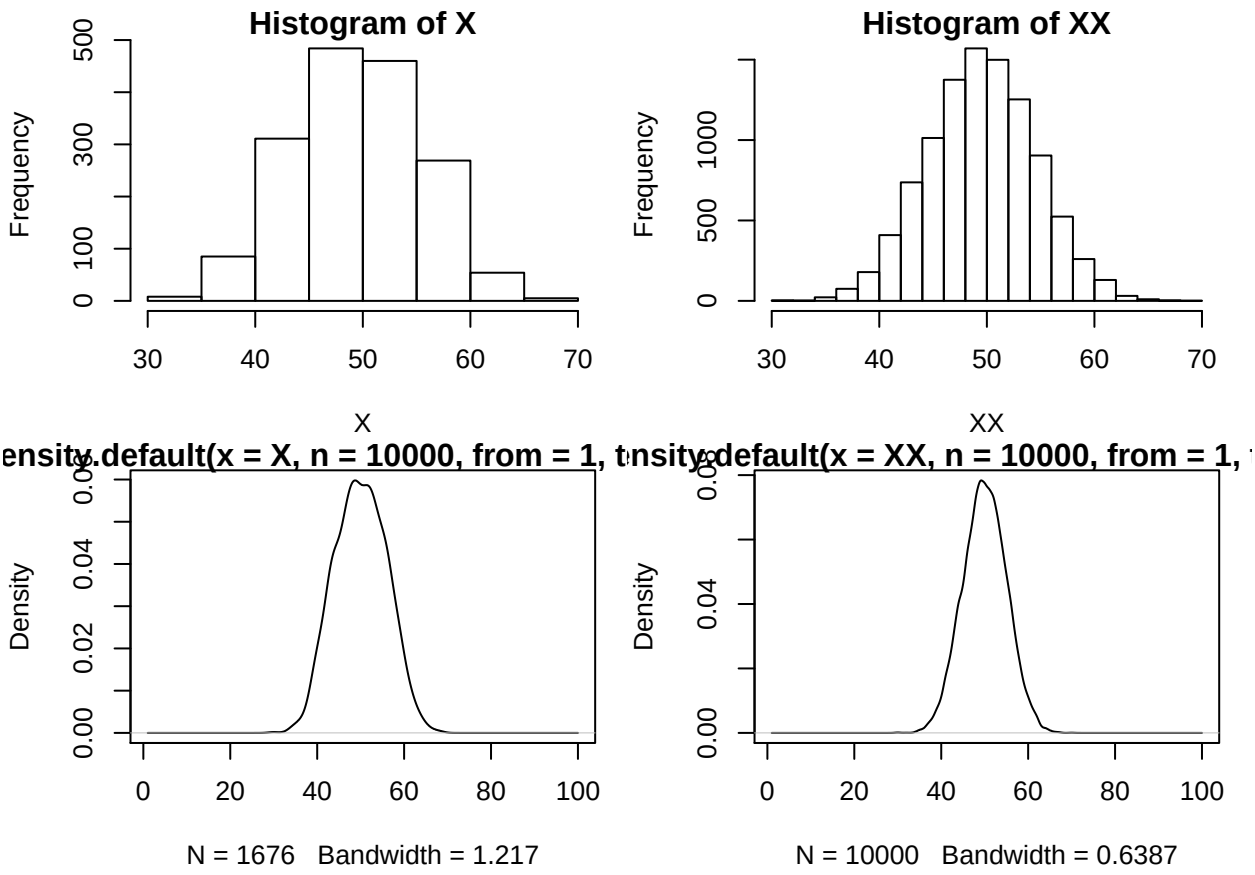


proposal — unif 3, p=0.5

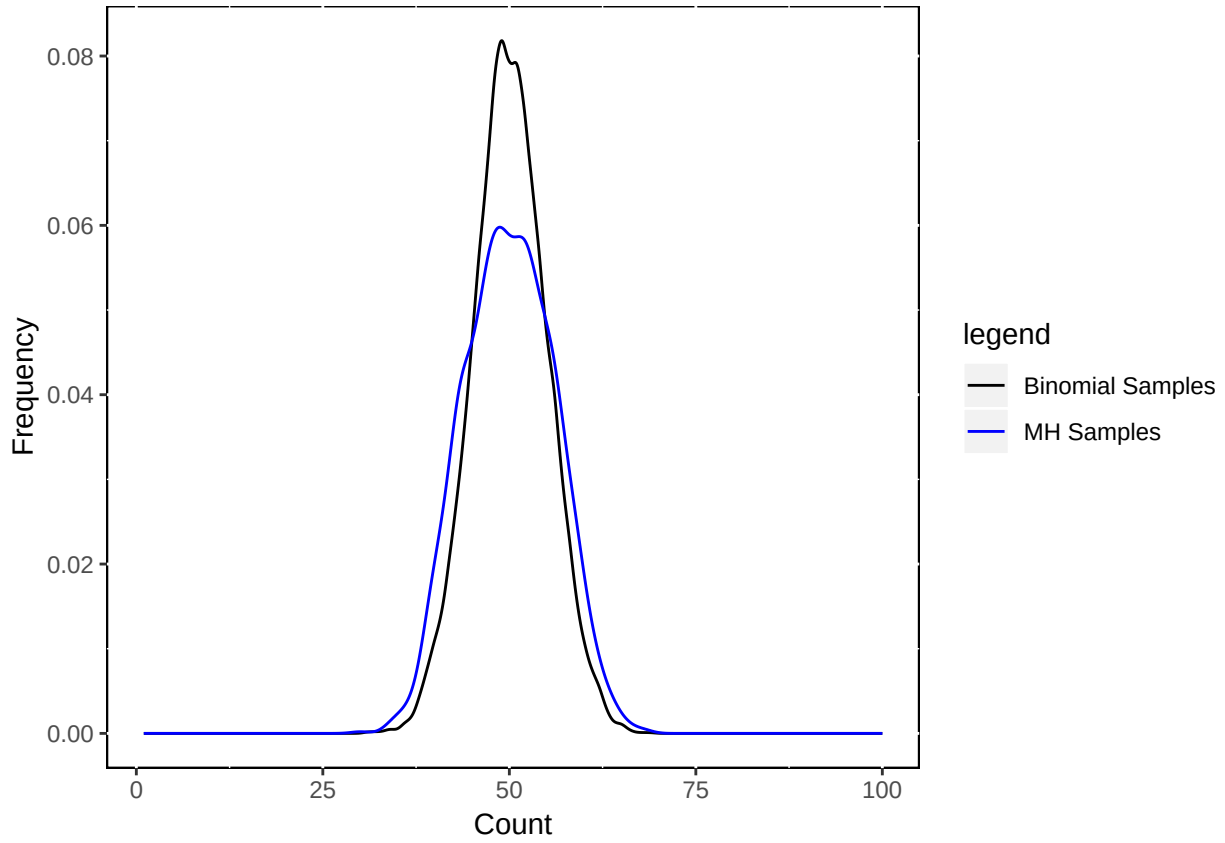
```
f_1 <- function(x){(factorial(100)/(factorial(x)*factorial(100-x)))*0.5^(x)*(1-0.5)^(100-x)}
# define the Binomial distribution , n=100, p=0.5
q_0 <- function(x){1/100}
# define the proposal function q_0 (Uniform)
n <- 100
X <- numeric() # store samples
count <- 1
x_old <- 0 # initialize the Markov chain
for(i in 1:10000)
{
  x_new <- sample.int(101, 1)-1 # sample a integer from q_0, as a candidate
  u <- runif(1,0,1)
```

```
  acc_1 <- min((f_1(x_new)*q_0(x_old))/(f_1(x_old)*q_0(x_new)), 1)
# calculate the acceptance prob
  if(u < acc_1) # accept
  {
    X[count] <- x_new
    count <- count +1
    x_old <- x_new
  }
  else # reject
  {
    x_old <- x_old
  }
}
par(mfrow = c(2, 2), mar = c(4, 4, 1, 1))
hist(X)
XX <- rbinom(10000, 100, 0.5)
hist(XX)
P <- density(X, n = 10000, from = 1, to  = 100)
plot(P)
B <- density(XX, n = 10000, from = 1, to  = 100)
plot(B)
p <- 0.5
real <- rbinom(10000, 100, p)
right_boundary <- max( c( (median(real)-1)*2.5, (median(X)-1)*2.5) )
  if(right_boundary > 100){right_boundary <- 100}


  d <- density(real, n = 10000, from = 1, to = right_boundary)
  d1 <- density(X, n = 10000, from = 1, to = right_boundary)
  Data_up <- data.frame(count = d$x, y = d$y, legend = rep(c("Binomial Samples"), 10000, 1))
  Data_down <- data.frame(count = d1$x, y = d1$y, legend = rep(c("MH Samples"), 10000, 1))
  Data <- rbind(Data_up, Data_down)
  acceptance_rate <- (count-1) / 100000
  library(ggplot2)
```




```
g <- ggplot( Data, aes(x = count, y = y, group = legend) ) +
  geom_line( aes(color = legend), size = 0.5 ) +
  xlab("Count") + ylab("Frequency") +
  scale_color_manual(values = c("black", "blue")) +
  theme(panel.background = element_rect(fill = 'white', colour = 'black'))
g
```

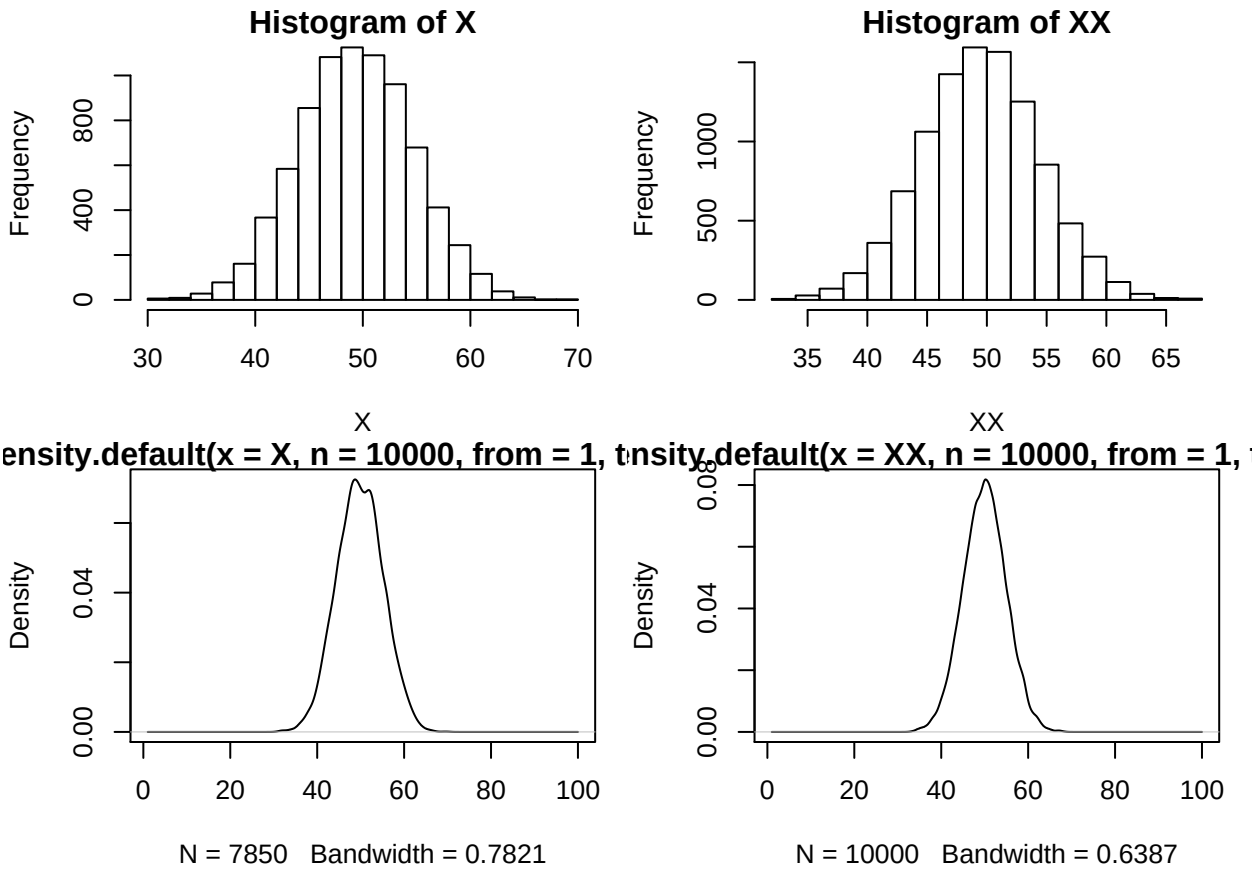


proposal — poisson lamdba=50

```
lambda <- 50
f_1 <- function(x){(factorial(100)/(factorial(x)*factorial(100-x)))*0.5^(x)*(1-0.5)^(100-x)}
# define the Binomial distribution , n=100, p=0.5
q_1 <- function(x){((lambda^x)*exp(-lambda))/factorial(x)}
# define the proposal function q_1 (Poisson), lambda=np=50
n <- 100
X <- numeric()  # store samples
count <- 1
x_old <- 0 # initialize the Markov chain
for(i in 1:10000)
{

  x_new <- rpois(1, lambda) # sample a candidate from q_1
  u <- runif(1,0,1)
  acc_1 <- min((f_1(x_new)*q_1(x_old))/(f_1(x_old)*q_1(x_new)), 1)
# calculate the acceptance prob
  if(u < acc_1) # accept
  {
    X[count] <- x_new
    count <- count +1
    x_old <- x_new
  }
  else # reject
```

```
  {
    x_old <- x_old
  }
}
par(mfrow = c(2, 2), mar = c(4, 4, 1, 1))
hist(X)
XX <- rbinom(10000, 100, 0.5)
hist(XX)
P <- density(X, n = 10000, from = 1, to  = 100)
plot(P)
B <- density(XX, n = 10000, from = 1, to  = 100)
plot(B)
```




```
p <- 0.5
real <- rbinom(10000, 100, p)
right_boundary <- max( c( (median(real)-1)*2.5, (median(X)-1)*2.5) )
  if(right_boundary > 100){right_boundary <- 100}


  d <- density(real, n = 10000, from = 1, to = right_boundary)
  d1 <- density(X, n = 10000, from = 1, to = right_boundary)
  Data_up <- data.frame(count = d$x, y = d$y, legend = rep(c("Binomial Samples"), 10000, 1))
  Data_down <- data.frame(count = d1$x, y = d1$y, legend = rep(c("MH Samples"), 10000, 1))
  Data <- rbind(Data_up, Data_down)
  acceptance_rate <- (count-1) / 100000
  library(ggplot2)
```

```
g <- ggplot( Data, aes(x = count, y = y, group = legend) ) +
  geom_line( aes(color = legend), size = 0.5 ) +
  xlab("Count") + ylab("Frequency") +
  scale_color_manual(values = c("black", "blue")) +
  theme(panel.background = element_rect(fill = 'white', colour = 'black'))
g
```