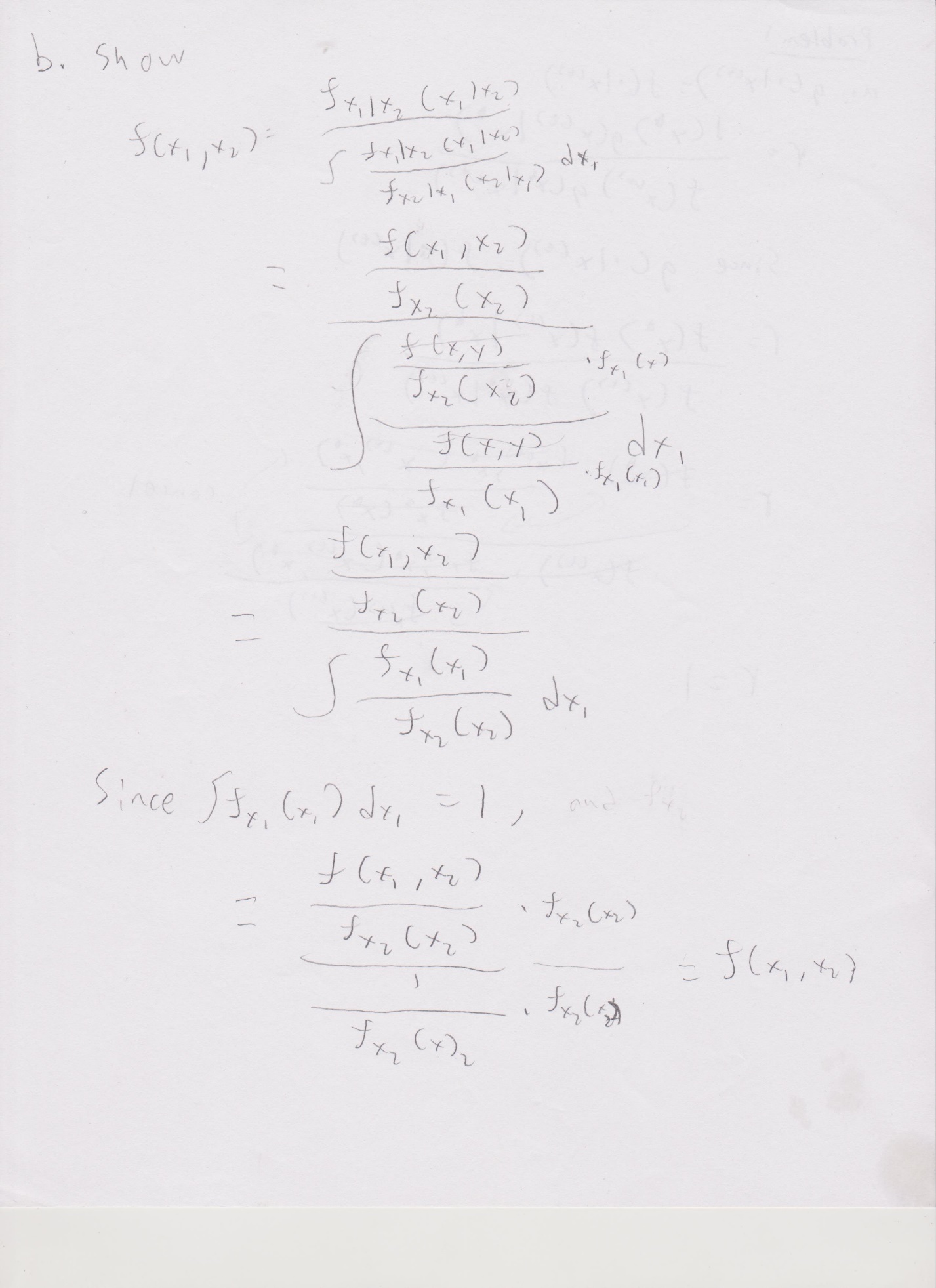
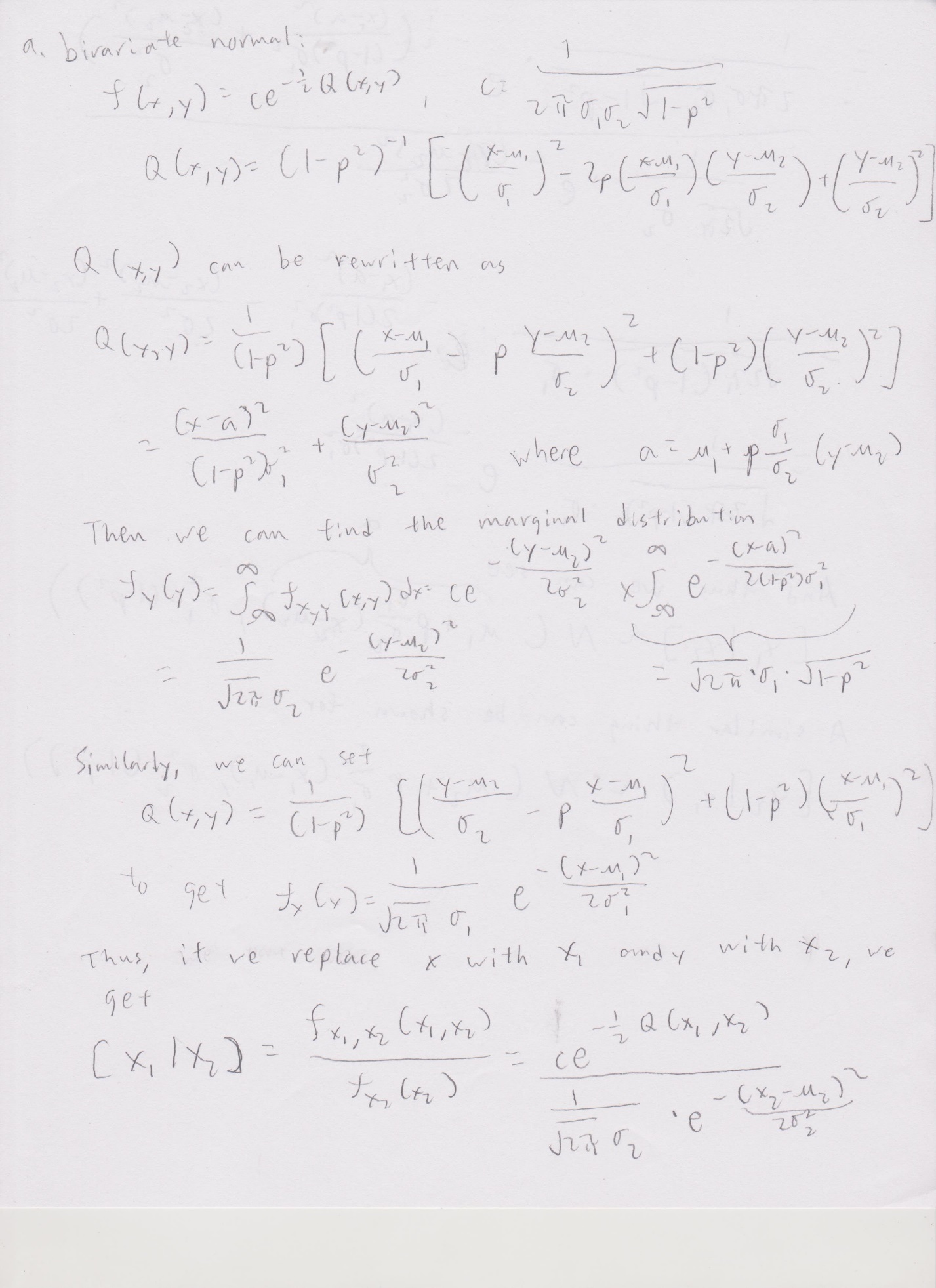
Yichen Dong HW 6

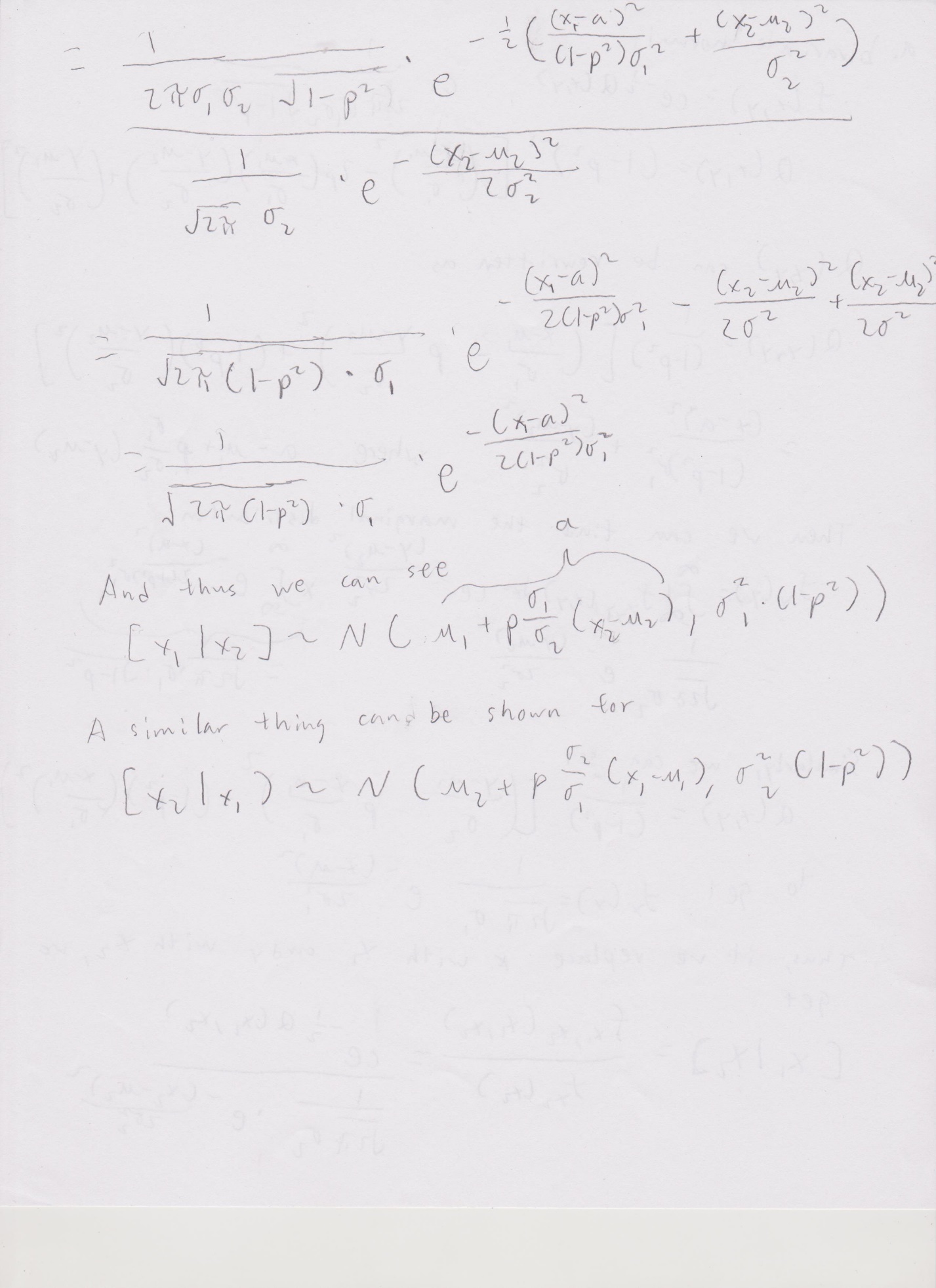
Yichen Dong

October 10, 2018

## C:\Users\Yichen\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Image (141).jpg

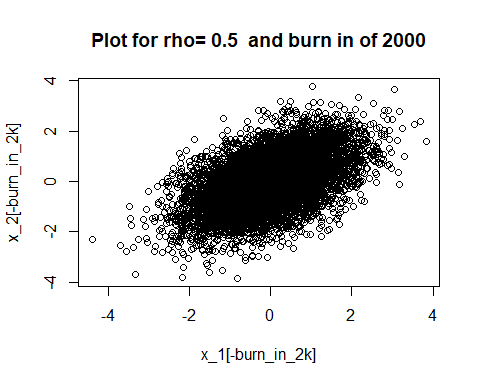
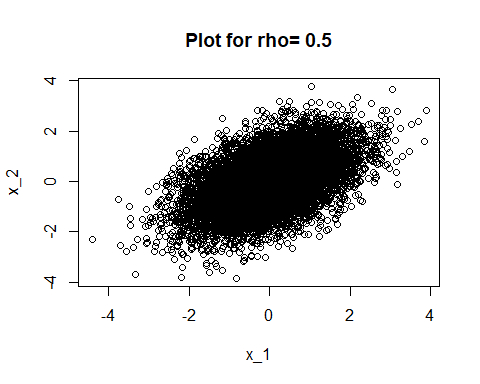
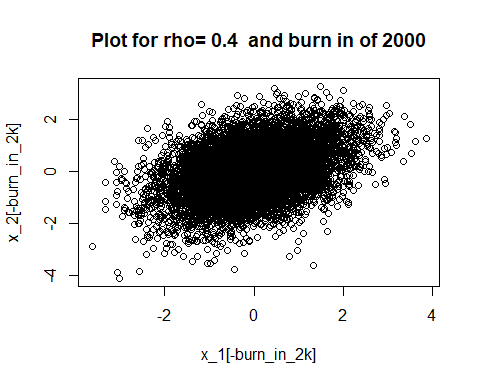
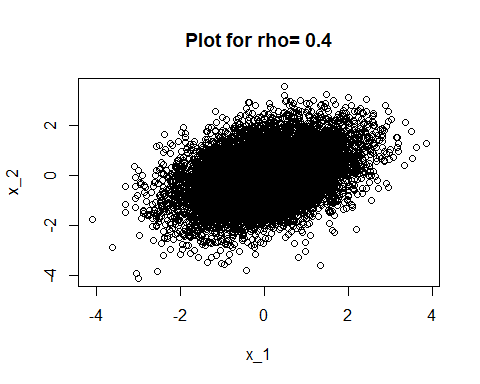
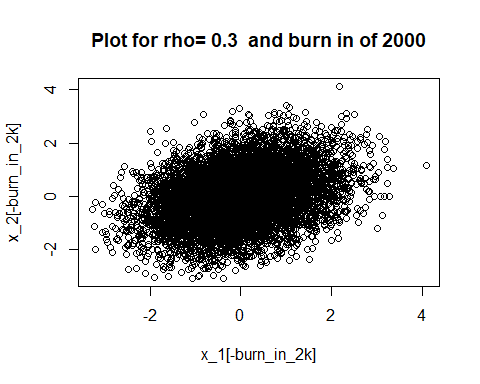
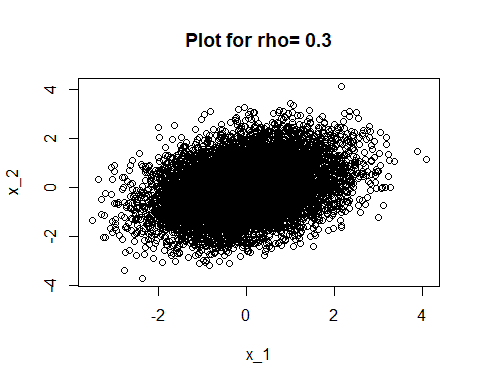
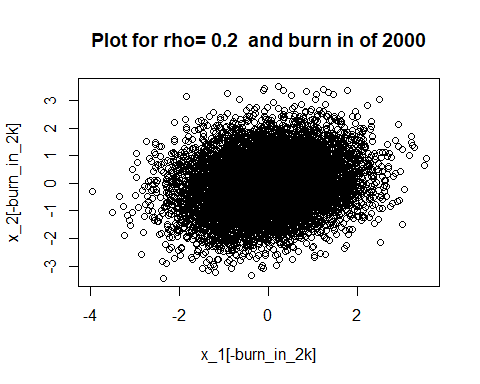
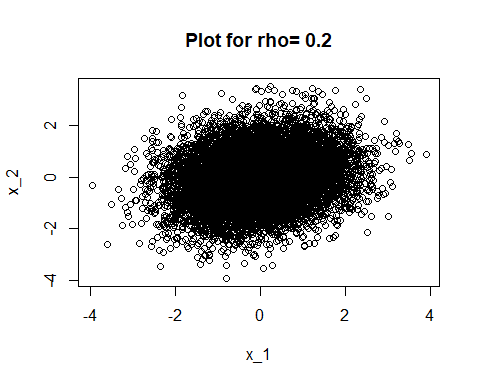
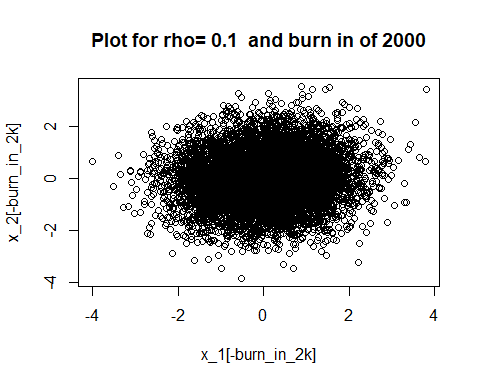
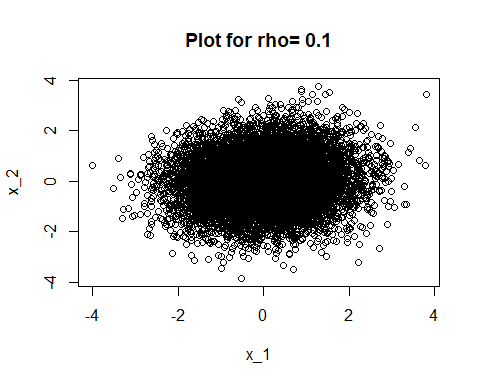
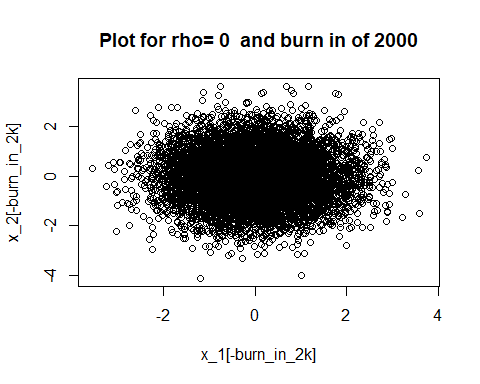
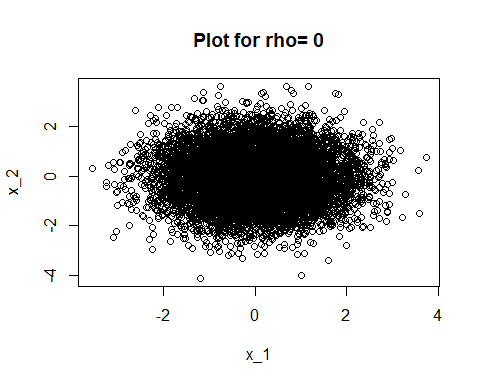






## Problem 2 Part 2

set.seed(1234)  
summary = data.frame("mean\_x\_1" = numeric(),"var\_x\_1" = numeric(), "mean\_x\_2" = numeric(),"var\_x\_2" = numeric(), "corr" =numeric(), "rho" = numeric())  
counter = 1  
burn\_in\_2k = 1:2000  
for(rho in c(0,.1,.2,.3,.4,.5)){  
 itr = 10000  
 u\_1 = 0  
 u\_2 = 0  
 s\_1 = 1  
 s\_2 = 1  
 x\_1 = NULL  
 x\_2 = NULL  
 x\_1[1] = 0  
 x\_2[1] = 0  
   
 for(i in 1:itr){  
 x\_1[i+1] = rnorm(1,u\_1+rho\*s\_1/s\_2\*(x\_2[i]-u\_2),sqrt(s\_1^2\*(1-rho^2)))  
 x\_2[i+1] = rnorm(1,u\_2+rho\*s\_2/s\_1\*(x\_1[i+1]-u\_1),sqrt(s\_2^2\*(1-rho^2)))  
 }  
   
 plot(x\_1,x\_2)  
 title(paste("Plot for rho=", rho))  
 plot(x\_1[-burn\_in\_2k],x\_2[-burn\_in\_2k])  
 title(paste("Plot for rho=", rho, " and burn in of 2000"))  
   
 summary[counter,1] = mean(x\_1)  
 summary[counter,2]=var(x\_1)  
 summary[counter,3]=mean(x\_2)  
 summary[counter,4]=var(x\_2)  
 summary[counter,5]=cor(x\_1,x\_2)  
 summary[counter,6]=rho  
 counter = counter + 1  
}



summary

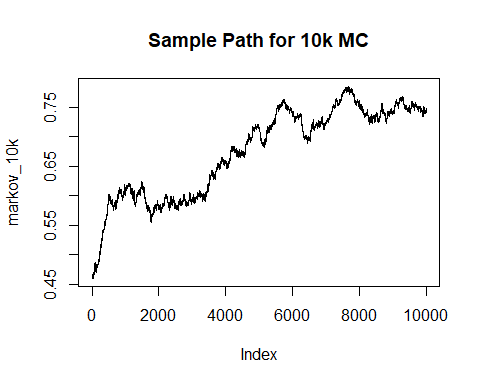
## mean\_x\_1 var\_x\_1 mean\_x\_2 var\_x\_2 corr rho  
## 1 0.007790845 0.9932918 -0.009421811 0.9936796 -0.007484543 0.0  
## 2 0.011437529 1.0218916 0.005248704 0.9834821 0.098044435 0.1  
## 3 -0.006625190 1.0014226 -0.006211513 1.0042913 0.199493907 0.2  
## 4 -0.008315305 1.0081808 0.010501223 1.0006561 0.311769035 0.3  
## 5 0.019291907 1.0027863 0.018648848 0.9833818 0.392822570 0.4  
## 6 0.013711359 0.9902929 0.001580508 0.9992635 0.499122714 0.5

We can see that the Gibbs sampling appears to be an appropriate method for this problem. The mean and variance of x\_1 and x\_2 were both very close to their actual values for all values of rho, and the correlation is also very close to the given correlation of rho.

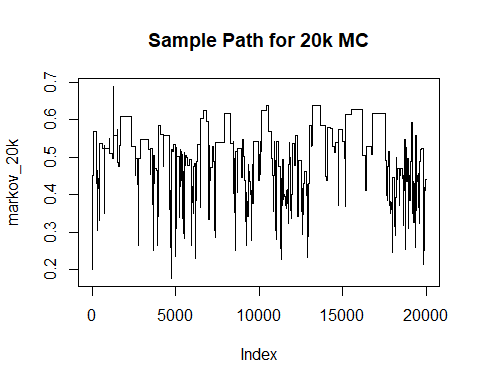
## Problem 3

markov\_10k = scan("Module 6 Data Sets/MCdata1.txt")  
markov\_20k = scan("Module 6 Data Sets/MCdata2.txt")

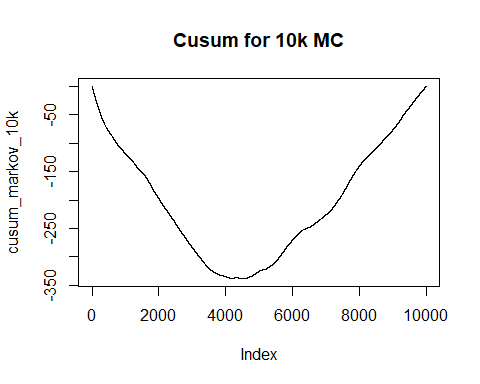
plot(markov\_10k,type = "l")  
title("Sample Path for 10k MC")

 This does not look like it’s wiggling vigorously about a single region

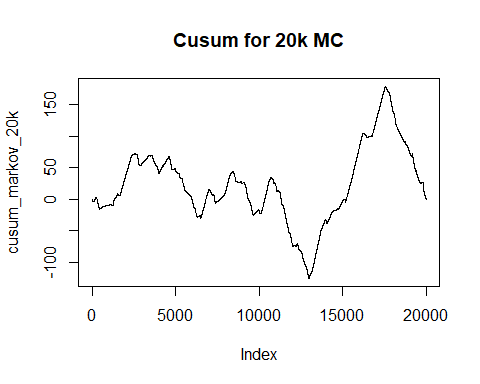
plot(markov\_20k,type = "l")  
title("Sample Path for 20k MC")

 This looks like it’s wiggling more around the same region, but I wouldn’t exactly call it vigorous in some places.

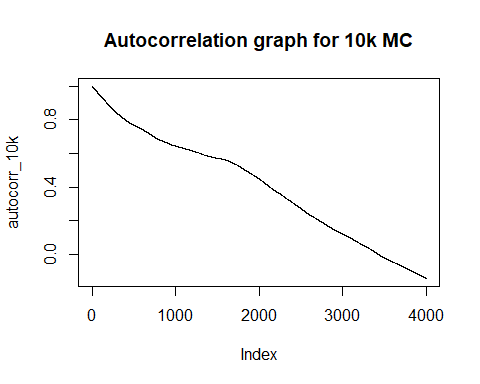
theta\_10k = mean(markov\_10k)  
theta\_20k = mean(markov\_20k)  
  
cusum\_markov\_10k = NULL  
for(i in 1:length(markov\_10k)){  
 cusum\_markov\_10k[i] = sum(markov\_10k[1:i] - theta\_10k)  
}  
plot(cusum\_markov\_10k, type = "l")  
title("Cusum for 10k MC")

 This isn’t wiggly or only contain small excursions from 0. In fact, the excursion it makes is only negative until about the 5000 mark, then it’s only positive!

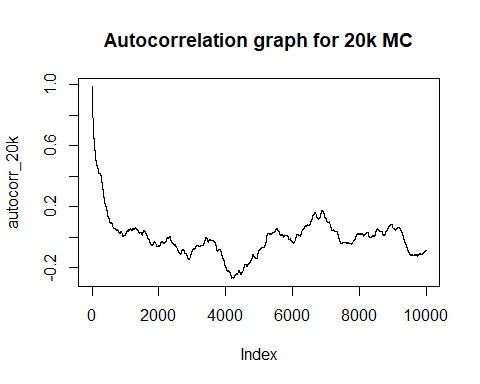
cusum\_markov\_20k = NULL  
for(i in 1:length(markov\_20k)){  
 cusum\_markov\_20k[i] = sum(markov\_20k[1:i] - theta\_20k)  
}  
plot(cusum\_markov\_20k, type = "l")  
title("Cusum for 20k MC")

 This looks like it’s somewhat centered around 0 in the beginning, and has a sort of wiggly shape to it, but it starts losing its wiggliness after 10000 and has periods of going only down or up. It also starts getting larger deviations from 0.

autocorr\_10k = NULL  
for(i in 1:4000){  
 R\_i = NULL  
 C\_i = 0  
 C\_0 = 0  
 x\_bar = mean(markov\_10k)  
 n = length(markov\_10k)  
 for(t in 1:(length(markov\_10k)-i)){  
 C\_i = C\_i + 1/n\*(markov\_10k[t] - x\_bar)\*(markov\_10k[t+i] - x\_bar)  
 }  
 for(t in 1:length(markov\_10k)){  
 C\_0 = C\_0 + 1/n\*(markov\_10k[t] - x\_bar)^2  
 }  
 R\_i = C\_i/C\_0  
 autocorr\_10k[i] = R\_i  
}  
plot(autocorr\_10k,type = "l")  
title("Autocorrelation graph for 10k MC")

 While the correlation is going down linearly, I’m not sure if the pace of this is quick enough to say that this chain has good mixing properties. I think we want to see the autocorrelation decrease rapidly initially.

autocorr\_20k = NULL  
for(i in 1:10000){  
 R\_i = NULL  
 C\_i = 0  
 C\_0 = 0  
 x\_bar = mean(markov\_20k)  
 n = length(markov\_20k)  
 for(t in 1:(length(markov\_20k)-i)){  
 C\_i = C\_i + 1/n\*(markov\_20k[t] - x\_bar)\*(markov\_20k[t+i] - x\_bar)  
 }  
 for(t in 1:length(markov\_20k)){  
 C\_0 = C\_0 + 1/n\*(markov\_20k[t] - x\_bar)^2  
 }  
 R\_i = C\_i/C\_0  
 autocorr\_20k[i] = R\_i  
}  
  
plot(autocorr\_20k,type = "l")  
title("Autocorrelation graph for 20k MC")

 This autocorrelation graph looks pretty good initally, as there is a steep decline in correlation initally. However, it appears that around a lag of 1000 the correlation stops decreasing and starts going back up and then starts being correlated in the negative direction. It does seem to hang around 0, so I would say that the mixing properties are ok.

Based on these graphs, it seems that Markov Chain 1 has both decent mixing but very poor convergence. The correlation in the autocorrelation plot is going down linearly over time, but it doesn’t have the sleep decline in the beginning. The sample path is wiggling vigorously, but not around a stable distribution. The cusum graph is not centered around 0 at all, and is not wiggly. Thus, convergence is pretty bad.

Markov Chain 2 appears to have good mixing initially, but eventually the mixing becomes worse past the 10000th mark. This can be seen in the later stages of the sample path, where it stays at the same value for large periods of time. The convergence also seems good initially, but in the later stages diverges from 0 in the Cusum plot.

## Problem 4

multi\_markov\_1 = scan("Module 6 Data Sets/MultiMC1.txt")  
multi\_markov\_2 = scan("Module 6 Data Sets/MultiMC2.txt")  
multi\_markov\_3 = scan("Module 6 Data Sets/MultiMC3.txt")  
multi\_markov\_4 = scan("Module 6 Data Sets/MultiMC4.txt")  
multi\_markov\_5 = scan("Module 6 Data Sets/MultiMC5.txt")  
multi\_markov\_6 = scan("Module 6 Data Sets/MultiMC6.txt")  
multi\_markov\_7 = scan("Module 6 Data Sets/MultiMC7.txt")  
multi\_markov\_all = cbind(multi\_markov\_1,multi\_markov\_2,multi\_markov\_3,multi\_markov\_4,multi\_markov\_5,multi\_markov\_6,multi\_markov\_7)

##Entire chain with D=0 and L=1000  
D=0  
L=1000  
J=7  
x\_bar\_j = NULL  
for(i in 1:J){  
 x\_bar\_j[i]=sum(multi\_markov\_all[(D+1):(D+L),i])/L  
}  
  
x\_bar\_dot = mean(x\_bar\_j)  
B = L/(J-1)\*sum((x\_bar\_j-x\_bar\_dot)^2)  
s\_j\_squared = NULL  
for(i in 1:J){  
 s\_j\_squared[i] = 1/(L-1) \* sum((multi\_markov\_all[(D+1):(D+L),i] - x\_bar\_j[i])^2)  
}  
W = mean(s\_j\_squared)  
R = (((L-1)/L)\*W+(1/L)\*B)/W  
sqrt\_R = sqrt(R)  
paste("Sqrt\_R for D =",D,"and L =",L,"is", round(sqrt\_R,4), ",B is", round(B,4), ",W is",round(W,4))

## [1] "Sqrt\_R for D = 0 and L = 1000 is 1.0082 ,B is 0.1137 ,W is 0.0065"

##Entire chain with D=500 and L=500  
D=500  
L=500  
J=7  
x\_bar\_j = NULL  
for(i in 1:J){  
 x\_bar\_j[i]=sum(multi\_markov\_all[(D+1):(D+L),i])/L  
}  
  
x\_bar\_dot = mean(x\_bar\_j)  
B = L/(J-1)\*sum((x\_bar\_j-x\_bar\_dot)^2)  
s\_j\_squared = NULL  
for(i in 1:J){  
 s\_j\_squared[i] = 1/(L-1) \* sum((multi\_markov\_all[(D+1):(D+L),i] - x\_bar\_j[i])^2)  
}  
W = mean(s\_j\_squared)  
R = (((L-1)/L)\*W+(1/L)\*B)/W  
sqrt\_R = sqrt(R)  
paste("Sqrt\_R for D =",D,"and L =",L,"is", round(sqrt\_R,4), ",B is", round(B,4), ",W is",round(W,4))

## [1] "Sqrt\_R for D = 500 and L = 500 is 1.0212 ,B is 0.1083 ,W is 0.0048"

##First 500 elements with D=0 and L=500  
D=0  
L=500  
J=7  
x\_bar\_j = NULL  
for(i in 1:J){  
 x\_bar\_j[i]=sum(multi\_markov\_all[(D+1):(D+L),i])/L  
}  
  
x\_bar\_dot = mean(x\_bar\_j)  
B = L/(J-1)\*sum((x\_bar\_j-x\_bar\_dot)^2)  
s\_j\_squared = NULL  
for(i in 1:J){  
 s\_j\_squared[i] = 1/(L-1) \* sum((multi\_markov\_all[(D+1):(D+L),i] - x\_bar\_j[i])^2)  
}  
W = mean(s\_j\_squared)  
R = (((L-1)/L)\*W+(1/L)\*B)/W  
sqrt\_R = sqrt(R)  
paste("Sqrt\_R for D =",D,"and L =",L,"is", round(sqrt\_R,4), ",B is", round(B,4), ",W is",round(W,4))

## [1] "Sqrt\_R for D = 0 and L = 500 is 1.0069 ,B is 0.0637 ,W is 0.008"

##First 500 elements with D=250 and L=250  
D=250  
L=250  
J=7  
x\_bar\_j = NULL  
for(i in 1:J){  
 x\_bar\_j[i]=sum(multi\_markov\_all[(D+1):(D+L),i])/L  
}  
  
x\_bar\_dot = mean(x\_bar\_j)  
B = L/(J-1)\*sum((x\_bar\_j-x\_bar\_dot)^2)  
s\_j\_squared = NULL  
for(i in 1:J){  
 s\_j\_squared[i] = 1/(L-1) \* sum((multi\_markov\_all[(D+1):(D+L),i] - x\_bar\_j[i])^2)  
}  
W = mean(s\_j\_squared)  
R = (((L-1)/L)\*W+(1/L)\*B)/W  
sqrt\_R = sqrt(R)  
paste("Sqrt\_R for D =",D,"and L =",L,"is", round(sqrt\_R,4), ",B is", round(B,4), ",W is",round(W,4))

## [1] "Sqrt\_R for D = 250 and L = 250 is 1.0192 ,B is 0.057 ,W is 0.0053"

##First 50 elements with D=0 and L=50  
D=0  
L=50  
J=7  
x\_bar\_j = NULL  
for(i in 1:J){  
 x\_bar\_j[i]=sum(multi\_markov\_all[(D+1):(D+L),i])/L  
}  
  
x\_bar\_dot = mean(x\_bar\_j)  
B = L/(J-1)\*sum((x\_bar\_j-x\_bar\_dot)^2)  
s\_j\_squared = NULL  
for(i in 1:J){  
 s\_j\_squared[i] = 1/(L-1) \* sum((multi\_markov\_all[(D+1):(D+L),i] - x\_bar\_j[i])^2)  
}  
W = mean(s\_j\_squared)  
R = (((L-1)/L)\*W+(1/L)\*B)/W  
sqrt\_R = sqrt(R)  
paste("Sqrt\_R for D =",D,"and L =",L,"is", round(sqrt\_R,4), ",B is", round(B,4), ",W is",round(W,4))

## [1] "Sqrt\_R for D = 0 and L = 50 is 1.0452 ,B is 0.1282 ,W is 0.0228"

##First 50 elements with D=25 and L=25  
D=25  
L=25  
J=7  
x\_bar\_j = NULL  
for(i in 1:J){  
 x\_bar\_j[i]=sum(multi\_markov\_all[(D+1):(D+L),i])/L  
}  
  
x\_bar\_dot = mean(x\_bar\_j)  
B = L/(J-1)\*sum((x\_bar\_j-x\_bar\_dot)^2)  
s\_j\_squared = NULL  
for(i in 1:J){  
 s\_j\_squared[i] = 1/(L-1) \* sum((multi\_markov\_all[(D+1):(D+L),i] - x\_bar\_j[i])^2)  
}  
W = mean(s\_j\_squared)  
R = (((L-1)/L)\*W+(1/L)\*B)/W  
sqrt\_R = sqrt(R)  
paste("Sqrt\_R for D =",D,"and L =",L,"is", round(sqrt\_R,4), ",B is", round(B,4), ",W is",round(W,4))

## [1] "Sqrt\_R for D = 25 and L = 25 is 1.4162 ,B is 0.0618 ,W is 0.0024"

It seemed that for the most part, the sqrt(R) was close to 1, except for the last one with only 25 elements. It seems that 50 elements might be too small. However, it appears that the sqrt(r) is practically the same for L=500 and L=1000 when D=0. It also seems that between chain variance is the smallest for D=250 and L=250.