Jared Yu

Computational Statistics

Problem Set 6

Note: Code for each of the problems is at the end under the appendix.

1. (a) Show that if in the Metropolis-Hastings algorithm the proposal distribution is $g\left(\cdot\,|x^{(t)}\right) = f(\cdot\,|x^{(t)})$, then the Metropolis-Hastings Ratio is always equal to one. (this is a first step to showing that the M-H ratio of a Gibbs sampler is always 1).

The Metropolis-Hastings Ratio is as follows:

$$R\left(x^{(t)}, x^*\right) = \frac{f(x^*)g\left(x^{(t)}|x^*\right)}{f(x^{(t)})g(x^*|x^{(t)})}$$

where a candidate $\mathbf{X}^*$ is sampled from $g\left(\cdot\,|x^{(t)}\right)$. Then, given that $g\left(\cdot\,|x^{(t)}\right) = f(\cdot\,|x^{(t)})$, the ratio can be rewritten as follows:

$$R\left(x^{(t)}, x^*\right) = \frac{f(x^*)f\left(x^{(t)}|x^*\right)}{f(x^{(t)})f(x^*|x^{(t)})}$$

For clarity, this will be expressed as follows:

$$R(\vec{u}, \vec{v}) = \frac{f(\vec{v})f(\vec{u}|\vec{v})}{f(\vec{u})f(\vec{v}|\vec{u})}$$

Using the formula for conditional densities, the following can be written:

$$f(\vec{v}|\vec{u}) = \frac{f(\vec{u},\vec{v})}{f(\vec{u})}, f(\vec{u}|\vec{v}) = \frac{f(\vec{u},\vec{v})}{f(\vec{v})}$$

Then the ratio can be rewritten as:

$$R(\vec{u}, \vec{v}) = \frac{f(\vec{v})f(\vec{u}|\vec{v})}{f(\vec{u})f(\vec{v}|\vec{u})} = \frac{f(\vec{v})\left[\frac{f(\vec{u},\vec{v})}{f(\vec{v})}\right]}{f(\vec{u})\left[\frac{f(\vec{u},\vec{v})}{f(\vec{u})}\right]} = \frac{f(\vec{u},\vec{v})}{f(\vec{u},\vec{v})} = 1 \blacksquare$$

(b) Let (X_1, X_2) be a two-dimensional random variable with joint density $f(x_1, x_2)$. Denote by $f_{X_1|X_2}(x_1|x_2)$ and $f_{X_2|X_1}(x_2|x_1)$ the corresponding conditional densities. Show that

$$f(x_1, x_2) = \frac{f_{X_1|X_2}(x_1|x_2)}{\int \frac{f_{X_1|X_2}(x_1|x_2)}{f_{X_2|X_1}(x_2|x_1)} dx_1}$$

For clarity, it will be noted that $f(x_1, x_2) = f_{X_1,X_2}(x_1, x_2)$.

The above will be shown from right to left.

$$\frac{f_{X_1|X_2}(x_1|x_2)}{\int \frac{f_{X_1|X_2}(x_1|x_2)}{f_{X_2|X_1}(x_2|x_1)}dx_1} = \frac{\frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_2}(x_2)}}{\int \frac{\frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_2}(x_2)}}{\frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_1}(x_1)}}dx_1}$$

$$\because f_{X_1|X_2}(x_1|x_2) = \frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_2}(x_2)} \text{ and } f_{X_2|X_1}(x_2|x_1) = \frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_1}(x_1)}$$

$$\cdots = \frac{\frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_2}(x_2)}}{\int \frac{f_{X_1,X_2}(x_1,x_2)f_{X_1}(x_1)}{f_{X_1,X_2}(x_1,x_2)f_{X_2}(x_2)}dx_1} = \frac{\frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_2}(x_2)}}{\frac{1}{f_{X_2}(x_2)}\int f_{X_1}(x_1)dx_1} = \frac{\frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_2}(x_2)}}{\frac{1}{f_{X_2}(x_2)}} = f_{X_1,X_2}(x_1,x_2) \blacksquare$$

2. (a) Derive the conditional distributions that are required to create a Gibbs sampler for drawing samples from a bivariate normal distribution with $\mu = \begin{bmatrix}\mu_1 \\ \mu_2\end{bmatrix}$ and $\Sigma = \begin{pmatrix}\sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2\end{pmatrix}$. (Hint: Look at the class notes.)

$$f_{X_1|X_2}(x_1|x_2) = \frac{f_{X_1,X_2}(x_1,x_2)}{f_{X_2}(x_2)}$$

$$= \frac{1}{\frac{1}{\sqrt{2\pi}\sigma_2}\exp\left\{-\frac{1}{2\sigma_2^2}(x_2-\mu_2)^2\right\}} \cdot \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}\exp\left\{-\frac{1}{2(1-\rho^2)}\left[\frac{(x_1-\mu_1)^2}{\sigma_1^2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2} - \frac{2\rho(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1\sigma_2}\right]\right\}$$

$$= \frac{1}{\frac{1}{\sqrt{2\pi}\sigma_2}\exp\left\{-\frac{1}{2\sigma_2^2}(x_2-\mu_2)^2\right\}} \cdot \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}\exp\left\{-\frac{1}{2(1-\rho^2)}\left[\frac{(x_1-\mu_1)^2}{\sigma_1^2}\right]\right\} \cdot \exp\left\{-\frac{1}{2(1-\rho^2)}\left[\frac{(x_2-\mu_2)^2}{\sigma_2^2} - \frac{2\rho(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1\sigma_2}\right]\right\}$$

$$\rightarrow \frac{(x_2-\mu_2)^2}{\sigma_2^2} - \frac{2\rho(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1\sigma_2}$$

$$= \frac{1}{\sigma_2\sigma_1^2}\left[\sigma_2(x_1^2 - 2\mu_1 x_1 + \mu_1^2) - 2\rho\sigma_1(x_1x_2 - x_1\mu_2 - x_2\mu_1 + \mu_1\mu_2)\right]$$

$$= \frac{1}{\sigma_2\sigma_1^2}\{\sigma_2 x_1^2 - 2x_1[\mu_1\sigma_2 + \rho\sigma_1 x_2 - \rho\sigma_1\mu_2] + \sigma_2\mu_1^2 + 2\rho\sigma_1 x_2\mu_1 - 2\rho\sigma_1\mu_1\mu_2\}$$

$$= \frac{1}{\sigma_1^2}\left\{x_1^2 - 2x_1\left[\mu_1 + \frac{\rho\sigma_1}{\sigma_2}(x_2-\mu_2)\right] + \mu_1^2 + \frac{2\rho\sigma_1\mu_1}{\sigma_2}(x_2-\mu_2)\right\}$$

$$= \frac{1}{\sigma_1^2}\left\{\left[x_1 - \left(\mu_1 + \frac{\rho\sigma_1}{\sigma_2}(x_2-\mu_2)\right)\right]^2 - \left(\mu_1 + \frac{\rho\sigma_1}{\sigma_2}(x_2-\mu_2)\right)^2 + \mu_1^2 + \frac{2\rho\sigma_1\mu_1}{\sigma_2}(x_2-\mu_2)\right\}$$

$$= \frac{1}{\sigma_1^2}\left\{\left[x_1 - \left(\mu_1 + \frac{\rho\sigma_1}{\sigma_2}(x_2-\mu_2)\right)\right]^2 - \left(\mu_1^2 + 2\frac{\rho\sigma_1}{\sigma_2}\mu_1(x_2-\mu_2) + \frac{\rho^2\sigma_1^2}{\sigma_2^2}(x_2-\mu_2)^2\right) + \mu_1^2\right.$$
$$\left. + \frac{2\rho\sigma_1\mu_1}{\sigma_2}(x_2-\mu_2)\right\}$$

$$= \frac{1}{\sigma_1^2}\left\{\left[x_1 - \left(\mu_1 + \frac{\rho\sigma_1}{\sigma_2}(x_2-\mu_2)\right)\right]^2 - \frac{\rho^2\sigma_1^2}{\sigma_2^2}(x_2-\mu_2)^2\right\}$$

$$\rightarrow \frac{1}{\frac{1}{\sqrt{2\pi}\sigma_2}\exp\left\{-\frac{1}{2\sigma_2^2}(x_2-\mu_2)^2\right\}}$$
$$\cdot \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}\exp\left\{-\frac{1}{2(1-\rho^2)}\left[\frac{(x_1-\mu_1)^2}{\sigma_1^2}\right]\right\}$$
$$\cdot \exp\left\{-\frac{1}{2(1-\rho^2)}\frac{1}{\sigma_1^2}\left\{\left[x_1 - \left(\mu_1 + \frac{\rho\sigma_1}{\sigma_2}(x_2-\mu_2)\right)\right]^2 - \frac{\rho^2\sigma_1^2}{\sigma_2^2}(x_2-\mu_2)^2\right\}\right\}$$

$$= \frac{1}{\sqrt{2\pi\sigma_1^2(1-\rho^2)}}\exp\left[-\frac{(x_2-\mu_2)^2}{2\sigma_2^2}\left(\frac{1}{1-\rho^2} - 1\right)\right.$$
$$\left. - \frac{1}{2\sigma_1^2(1-\rho^2)}\left\{\left[x_1 - \left(\mu_1 + \rho\left(\frac{\sigma_1}{\sigma_2}(x_2-\mu_2)\right)\right)\right]^2 - \frac{\rho^2\sigma_1^2}{\sigma_2^2}(x_2-\mu_2)^2\right\}\right]$$

$$\rightarrow -\frac{(x_2-\mu_2)^2}{2\sigma_2^2}\left(\frac{1}{1-\rho^2} - 1\right) + \frac{\frac{\rho^2\sigma_1^2}{\sigma_2^2}(x_2-\mu_2)^2}{2\sigma_1^2(1-\rho^2)}$$

$$= -\frac{(x_2-\mu_2)^2}{2\sigma_2^2}\left(\frac{\rho^2}{1-\rho^2}\right)+\frac{1}{2}(x_2-\mu_2)^2\frac{\rho^2}{1-\rho^2}=0$$

$$\Rightarrow \frac{1}{\sqrt{2\pi\sigma_1^2(1-\rho^2)}}\exp\left[-\frac{1}{2\sigma_1^2(1-\rho^2)}\left\{\left[x_1-\left(\mu_1+\rho\frac{\sigma_1}{\sigma_2}(x_2-\mu_2)\right)\right]^2\right\}\right]$$

$$\therefore X_1|X_2 \sim N\left(\mu_1+\rho\frac{\sigma_1}{\sigma_2}(x_2-\mu_2),\sigma_1^2(1-\rho^2)\right)$$

Similarly, it also follows that,

$$X_2|X_1 \sim N\left(\mu_2+\rho\frac{\sigma_1}{\sigma_2}(x_1-\mu_1),\sigma_2^2(1-\rho^2)\right) \blacksquare$$

(b) Now assume that $\mu_1=\mu_2=0$ and $\sigma_1^2=\sigma_2^2=1$ in the above distribution. For $\rho = 0,.1,.2,.3,.4,.5$, generate Gibbs samples with varying lengths of burn-in. Do you think that Gibbs sampling is a viable method for this problem?

Below in Figures 1-6 are 3×2 grids of plots, where the left-hand side represents the values for X_1 and the right-hand side represents values for X_2 when using Gibbs sampling for various $\rho = 0,.1,.2,\cdots,.5$. The corresponding burn-in values tested are: 100, 500, 1,000, 5,000, 10,000, and 7,000 from a set of 100,000 generated samples. The sample paths for all the different sets of values look roughly identical and they seem to show that the values of X_1 and X_2 are sampled well within the range of the distribution. From the sample paths alone, there seems to be good mixing, since there is vigorous wiggling about without noticeable slow movement or rejections. The cusum plots also look roughly similar, but it could be argued that some are better than others. If there are small excursions about zero with vigorous wiggling around then it's a good sign. However, an example would be X_1 from Figure 1 and X_2 in Figure 2. The former is considerably smoother in appearance than the other. This could indicate that there's some slow mixing taking place in the former example. Another issue is seen in X_2 of Figure 3, where the movement is largely to the upper portion above zero showing that there's some undesirable tendency towards one side. Overall, the cusum plots don't look too bad, as it would be strange for them to be too wiggly as if they looked like sample paths considering that they're generated from a Normal distribution. Also, in the autocorrelation function plots for each of the examples they look identical. There seems to be no dependency on previous values and therefore the sampling seems to be independent. This is a good sign and shows that there's good mixing. Overall, it seems that Gibbs sampling is a viable method, but from the below plots it seems that some tweaking needs to be done to optimize for certain values of ρ and different values for burn-in.

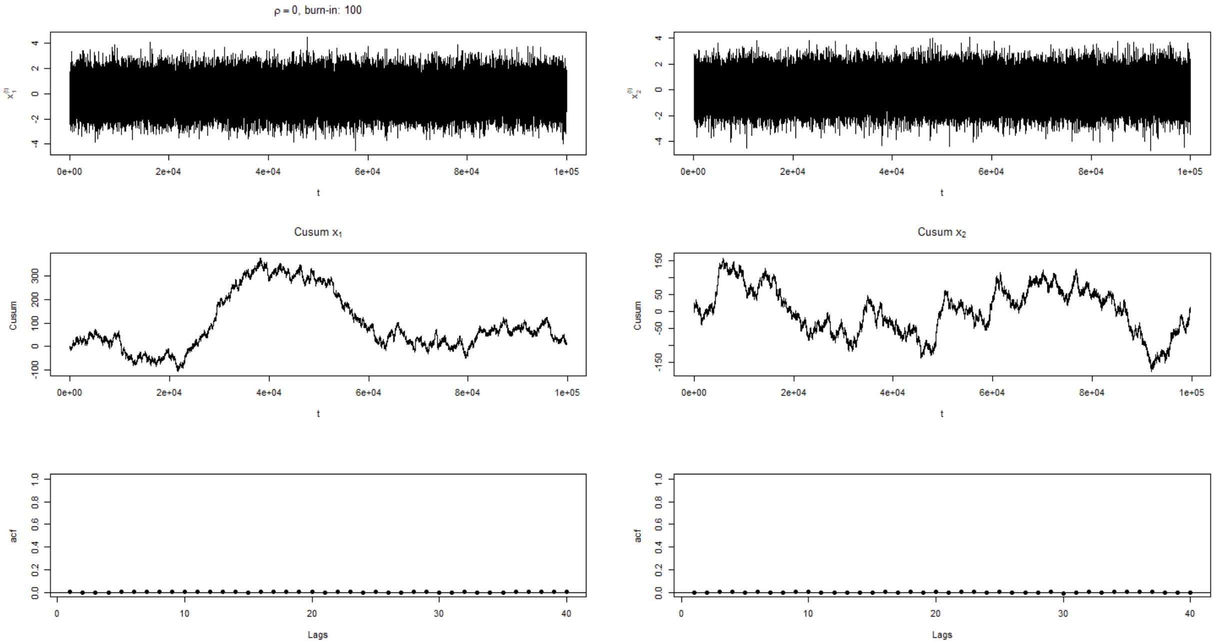


Figure 1 rho=0, burn-in: 100

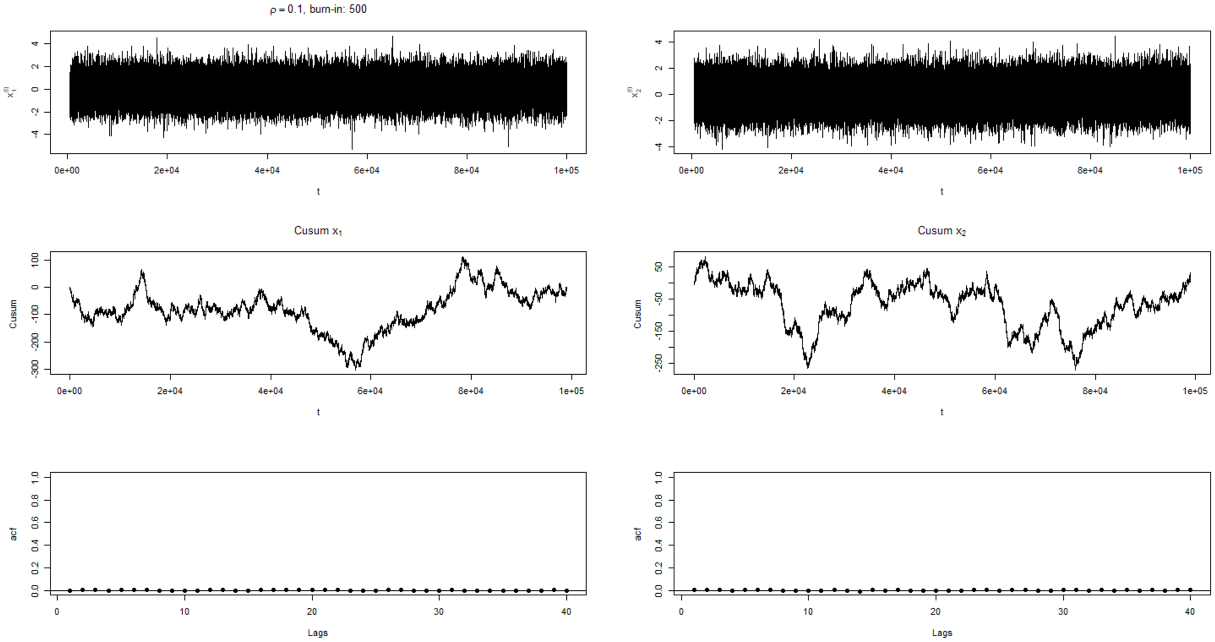


Figure 2 rho=0.1, burn-in: 500

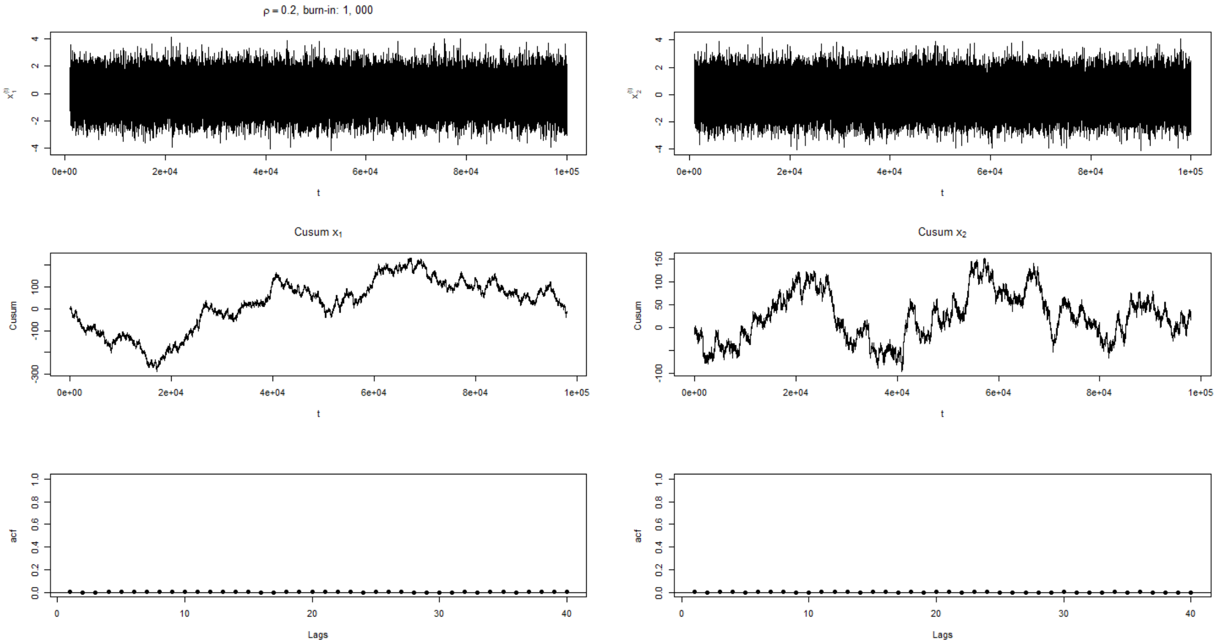


Figure 3 rho=0.2, burn-in: 1,000

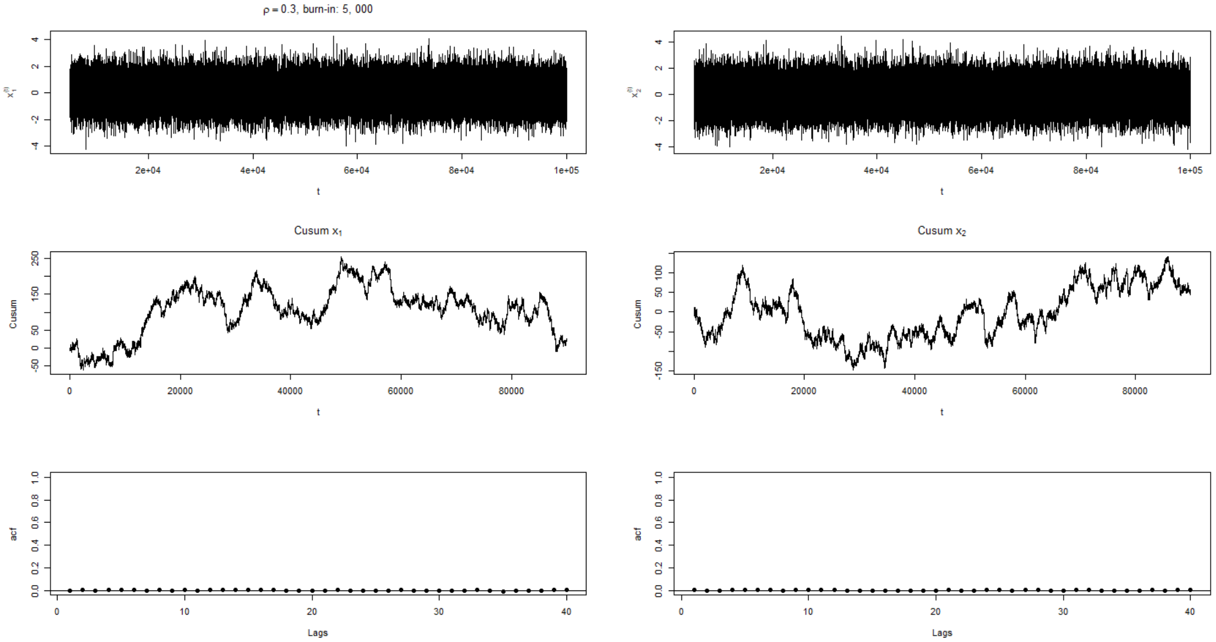


Figure 4 rho=0.3, burn-in: 5,000

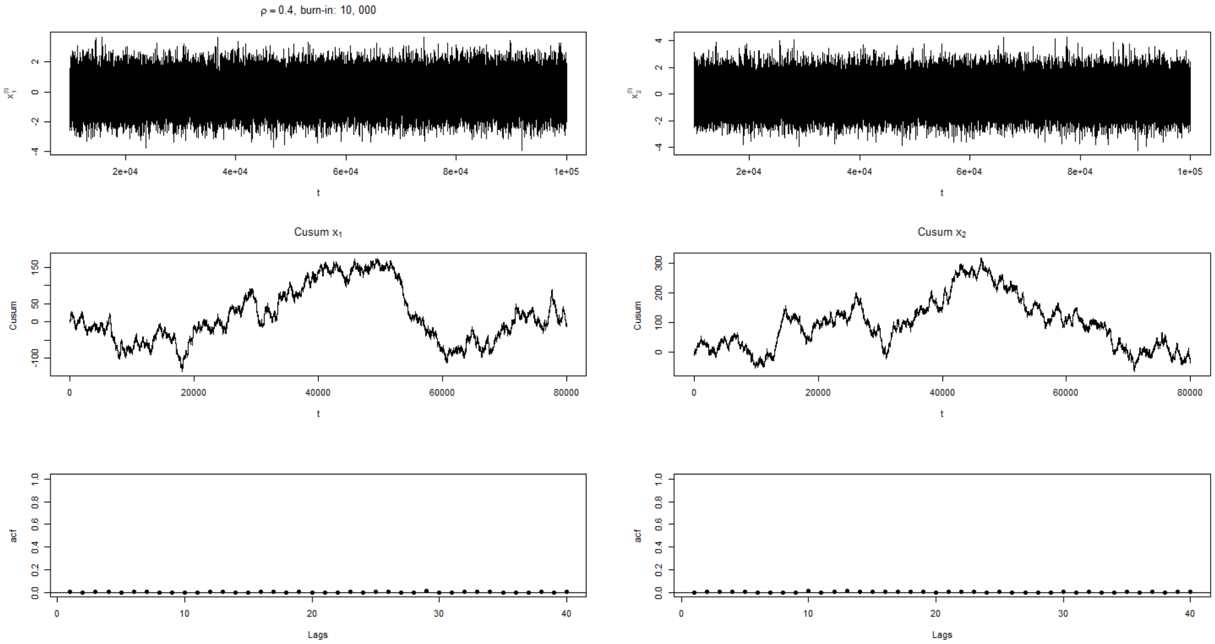


Figure 5 rho=0.4, burn-in: 10,000

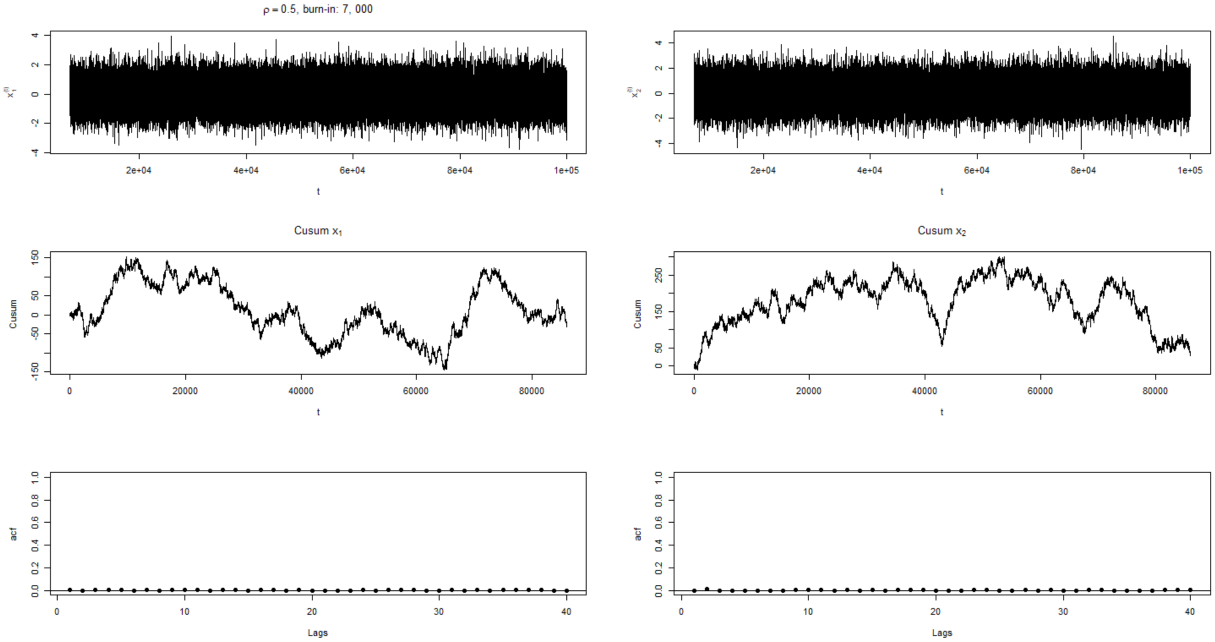
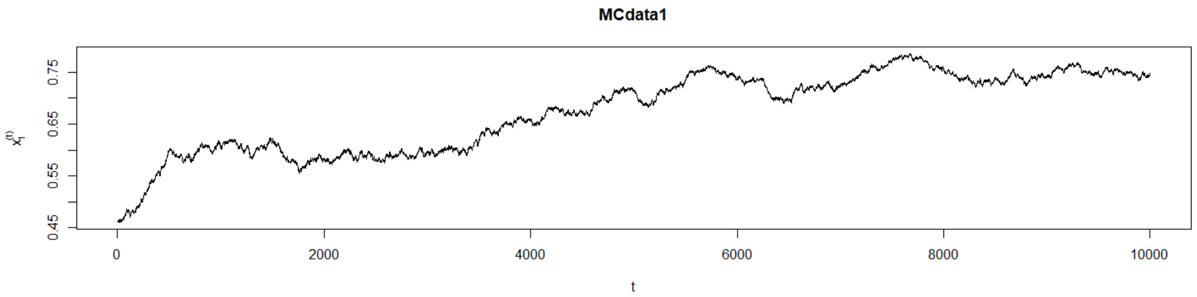
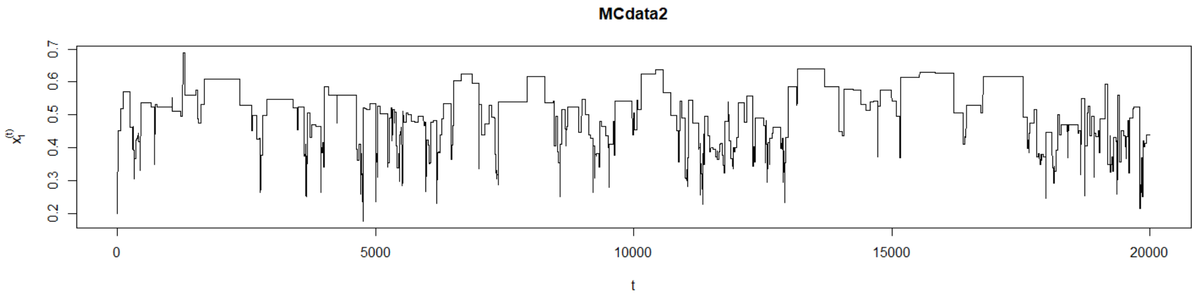


Figure 6 rho=0.5, burn-in: 7,000

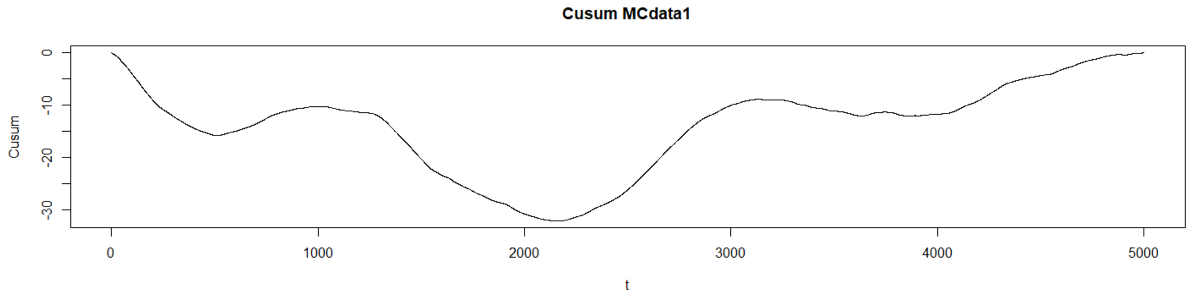
3. Posted on the course Blackboard site are the data sets MCdata1.txt and MCdata2.txt each containing Markov chains of length 10000 and 20000 respectively. For each of these Markov chains:
   (a) Plot the Sample Path

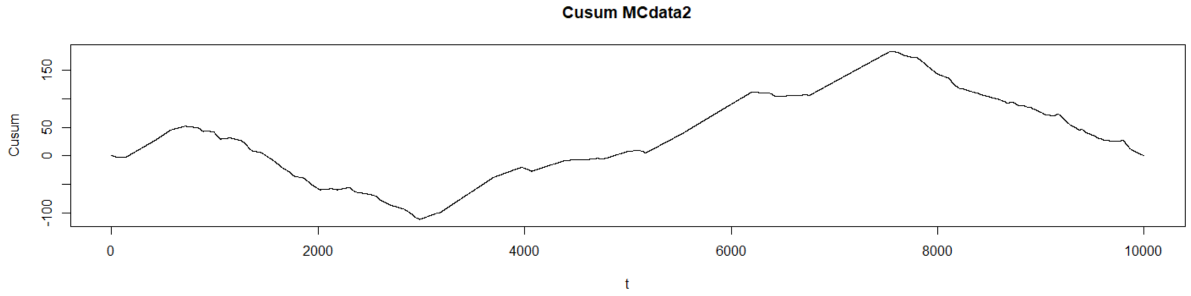




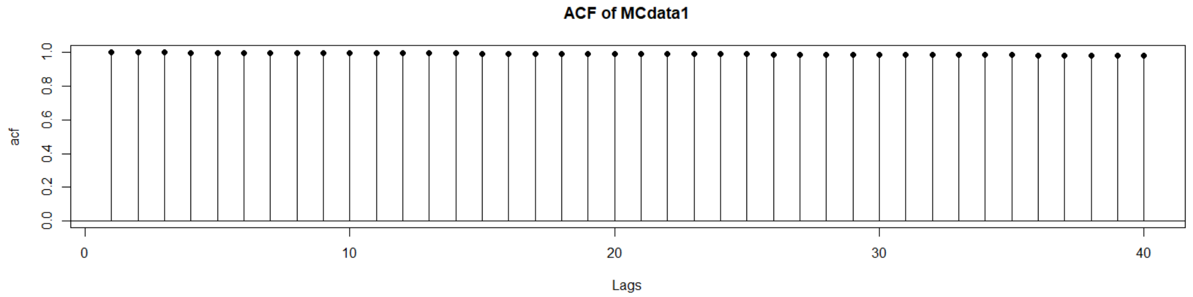
(b) Plot the Cusum diagnostic Plot for the mean

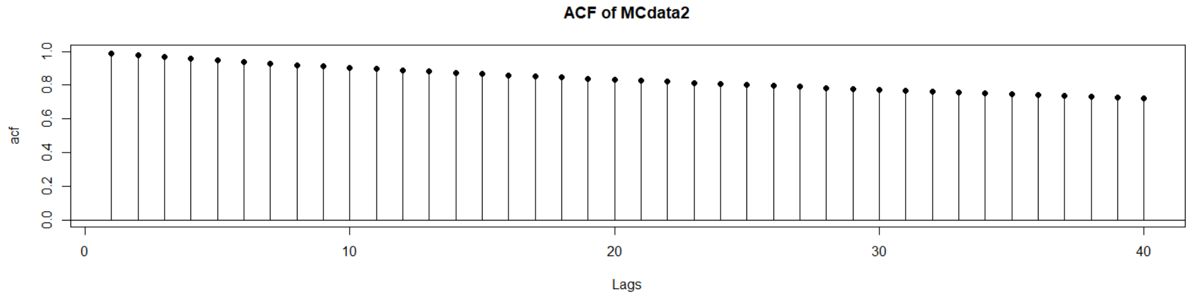
In the following plots, the first 5000 / 10000 iterations (i.e., half of the data or a conservative approach) were removed from each of the datasets.





(c) Plot the Autocorrelation Plot





(d) Discuss the mixing and convergence of the chain based on those three diagnostics.

In MCdata1the sample path shows that the chain is moving slowly from around 0.45 towards 0.7. It utilizes many iterations to edge slowly upwards before finding a range that it begins to settle down in. Even as it gets closer to this range, the sample path shows that it will still move around slowly rather than bouncing back and forth quickly around a certain range. It seems from the sample path that the chain itself is having trouble finding a good balance and there’s too much dependency on past values. This suggests poor mixing. The cusum plot shows problems also for MCdata1, where there’s a great deal of negative values and an inability to bounce around between positive and negative values about zero. Also, the curve is quite smooth and not wiggly as is ideal. This suggests slower mixing. In the autocorrelation plot there are many high values for many lags suggesting poor mixing with extremely slow decay.

The MCdata1 sample path shows that the chain moves quickly and can bounce between a range, but it’s evident from the many horizontal lines that there’s a great deal of rejection going on. In a univariate distribution such as this one, this is not a good sign and indicates that there’s some issue with the mixing and that it’s poor. Ideally, there’d be a greater deal of acceptance and hence fewer horizontal lines in the sample path. This would be depicted by wiggling about vigorously. The cusum plot it’s evident that the curve is going back and forth about zero, but the issue is that the curve is quite smooth and not wiggly. Also, ideally it would have small excursions around zero rather than the large ones seen in the plot. This suggests slower mixing speeds. In the autocorrelation plot, it shows that the lags are decaying, but it’s doing so at an extremely slow pace and the decay is not great enough. There’s still a high dependency on lags even up to 40 and so this shows that there’s poor mixing going on in the chain.

4. Posted on the course Blackboard site is the file MulitpleChains.txt. It contains 7 Markov Chains each of length 1000. For each of the following subsets of chains, calculate B, W

and $\sqrt(R)$ as described in Section 7.3.1.2 of the text and discuss your results. Be careful when you load the chains in. You may need to separate them into different files first depending on your method.

(a) The entire chains with $D = 0$ and $L = 1000$.

B	W	$\sqrt{R}$
0.0952192	0.00623490	1.007111

In this part, $\sqrt{R} < 1.2$ and so both D and L can be thought of as acceptable. This indicates that good mixing has occurred and $\sqrt{R} \to 1$ as $L \to \infty$. In this case, both D and L are large enough.

(b) The entire chains with $D = 500$ and $L = 500$.

B	W	$\sqrt{R}$
0.084016304	0.004562567	1.017265217

In this part, $\sqrt{R} < 1.2$ and so both D and L can be thought of as acceptable. This indicates that good mixing has occurred and $\sqrt{R} \to 1$ as $L \to \infty$. In this case, both D and L are large enough.

(c) The first 500 elements of the chains with $D = 0$ and $L = 500$.

B	W	$\sqrt{R}$
0.06485166	0.00775841	1.00733201

In this part, $\sqrt{R} < 1.2$ and so both D and L can be thought of as acceptable. This indicates that good mixing has occurred and $\sqrt{R} \to 1$ as $L \to \infty$. In this case, both D and L are large enough.

(d) The first 500 elements of the chains with $D = 250$ and $L = 250$.

B	W	$\sqrt{R}$
0.05675384	0.00542501	1.01874731

In this part, $\sqrt{R} < 1.2$ and so both D and L can be thought of as acceptable. This indicates that good mixing has occurred and $\sqrt{R} \to 1$ as $L \to \infty$. In this case, both D and L are large enough.

(e) The first 50 elements of the chains with $D = 0$ and $L = 50$.

B	W	$\sqrt{R}$
0.1269782	0.0216535	1.0475123

In this part, $\sqrt{R} < 1.2$ and so both D and L can be thought of as acceptable. This indicates that good mixing has occurred and $\sqrt{R} \to 1$ as $L \to \infty$. In this case, both D and L are large enough.

(f) The first 50 elements of the chains with $D = 25$ and $L = 25$.

B	W	$\sqrt{R}$
0.058073729	0.002523006	1.371388744

In this part, $\sqrt{R} > 1.2$ and so the chosen values of D and L are unacceptable. This indicates that poor mixing has occurred and both D and L aren't large enough.

Appendix

```r
library(latex2exp)
### Problem 2
g1 <- function(x2, mu1, mu2, sigma1, sigma2, rho) {
  rnorm(n = 1, mean = mu1 + rho * (sigma1 / sigma2) * (x2 - mu2),
        sd = sqrt((sigma1^2) * (1 - rho^2)))
}

g2 <- function(x1, mu1, mu2, sigma1, sigma2, rho) {
  rnorm(n = 1, mean = mu2 + rho * (sigma1 / sigma2) * (x1 - mu1),
        sd = sqrt((sigma2^2) * (1 - rho^2)))
}

gibbs_sampling <- function(x1_init, x2_init, mu1_init = 0, mu2_init = 0,
                           sigma1_init = 1, sigma2_init = 1, rho_init, iterations = 1e5) {
  sampling_matrix <- matrix(NA, nrow = iterations, ncol = 2)

  x1_t <- x1_init; x2_t <- x2_init; mu1 <- mu1_init; mu2 <- mu2_init
  sigma1 <- sigma1_init; sigma2 <- sigma2_init; rho <- rho_init

  for (i in 1:iterations) {
    x1_star <- g1(x2 = x2_t, mu1 = mu1, mu2 = mu2,
                  sigma1 = sigma1, sigma2 = sigma2, rho = rho)
    x2_star <- g2(x1 = x1_star, mu1 = mu1, mu2 = mu2,
                  sigma1 = sigma1, sigma2 = sigma2, rho = rho)
    sampling_matrix[i,] <- c(x1_star, x2_star)
  }
 return(sampling_matrix)
}

rho0_sample <- gibbs_sampling(x1_init = 0, x2_init = 0, rho_init = 0)
rho1_sample <- gibbs_sampling(x1_init = 0, x2_init = 0, rho_init = 0.1)
rho2_sample <- gibbs_sampling(x1_init = 0, x2_init = 0, rho_init = 0.2)
rho3_sample <- gibbs_sampling(x1_init = 0, x2_init = 0, rho_init = 0.3)
rho4_sample <- gibbs_sampling(x1_init = 0, x2_init = 0, rho_init = 0.4)
rho5_sample <- gibbs_sampling(x1_init = 0, x2_init = 0, rho_init = 0.5)

# Sampling Paths
burn_in1 <- 100
burn_in2 <- 500
burn_in3 <- 1e3
burn_in4 <- 5e3
burn_in5 <- 1e4
burn_in6 <- 7e3
# Reference: https://www.r-bloggers.com/setting-graph-margins-in-r-using-the-par-function-and-lots-of-cow-milk/
par(mfrow = c(3,2), mar=c(5,6,4,2) + 0.1)
plot(burn_in1:1e5, rho0_sample[burn_in1:1e5,1],
     type = 'l', ylab = TeX('$x_1^{(t)}$'), xlab = TeX('t'))
title(main = TeX('$\\rho = 0$, burn-in: 100'))
plot(burn_in1:1e5, rho0_sample[burn_in1:1e5,2],
     type = 'l', ylab = TeX('$x_2^{(t)}$'), xlab = TeX('t'))
cs1 <- cusum(theta_hat = mean(rho0_sample[burn_in1:1e5,1]),
      mcdata = rho0_sample[burn_in1:1e5,1], burn = burn_in1)
cs2 <- cusum(theta_hat = mean(rho0_sample[burn_in1:1e5,2]),
      mcdata = rho0_sample[burn_in1:1e5,2], burn = burn_in1)
plot(1:nrow(cs1), cs1, main = TeX('Cusum $x_1$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
plot(1:nrow(cs2), cs2, main = TeX('Cusum $x_2$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
```

```r
lags <- 40
autocorrelation(mcdata = rho0_sample[burn_in1:1e5,1], lag_i = lags)
autocorrelation(mcdata = rho0_sample[burn_in1:1e5,2], lag_i = lags)

plot(burn_in2:1e5, rho1_sample[burn_in2:1e5,1],
     type = 'l', ylab = TeX('$x_1^{(t)}$'), xlab = TeX('t'))
title(main = TeX('$\\rho = 0.1$, burn-in: 500'))
plot(burn_in2:1e5, rho1_sample[burn_in2:1e5,2],
     type = 'l', ylab = TeX('$x_2^{(t)}$'), xlab = TeX('t'))
cs3 <- cusum(theta_hat = mean(rho1_sample[burn_in2:1e5,1]),
             mcdata = rho1_sample[burn_in2:1e5,1], burn = burn_in2)
cs4 <- cusum(theta_hat = mean(rho1_sample[burn_in2:1e5,2]),
             mcdata = rho1_sample[burn_in2:1e5,2], burn = burn_in2)
plot(1:nrow(cs3), cs3, main = TeX('Cusum $x_1$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
plot(1:nrow(cs4), cs4, main = TeX('Cusum $x_2$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
autocorrelation(mcdata = rho1_sample[burn_in2:1e5,1], lag_i = lags)
autocorrelation(mcdata = rho1_sample[burn_in2:1e5,2], lag_i = lags)

plot(burn_in3:1e5, rho2_sample[burn_in3:1e5,1],
     type = 'l', ylab = TeX('$x_1^{(t)}$'), xlab = TeX('t'))
title(main = TeX('$\\rho = 0.2$, burn-in: 1,000'))
plot(burn_in3:1e5, rho2_sample[burn_in3:1e5,2],
     type = 'l', ylab = TeX('$x_2^{(t)}$'), xlab = TeX('t'))
cs5 <- cusum(theta_hat = mean(rho2_sample[burn_in3:1e5,1]),
             mcdata = rho2_sample[burn_in3:1e5,1], burn = burn_in3)
cs6 <- cusum(theta_hat = mean(rho2_sample[burn_in3:1e5,2]),
             mcdata = rho2_sample[burn_in3:1e5,2], burn = burn_in3)
plot(1:nrow(cs5), cs5, main = TeX('Cusum $x_1$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
plot(1:nrow(cs6), cs6, main = TeX('Cusum $x_2$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
autocorrelation(mcdata = rho2_sample[burn_in3:1e5,1], lag_i = lags)
autocorrelation(mcdata = rho2_sample[burn_in3:1e5,2], lag_i = lags)

plot(burn_in4:1e5, rho3_sample[burn_in4:1e5,1],
     type = 'l', ylab = TeX('$x_1^{(t)}$'), xlab = TeX('t'))
title(main = TeX('$\\rho = 0.3$, burn-in: 5,000'))
plot(burn_in4:1e5, rho3_sample[burn_in4:1e5,2],
     type = 'l', ylab = TeX('$x_2^{(t)}$'), xlab = TeX('t'))
cs7 <- cusum(theta_hat = mean(rho3_sample[burn_in4:1e5,1]),
             mcdata = rho3_sample[burn_in4:1e5,1], burn = burn_in4)
cs8 <- cusum(theta_hat = mean(rho3_sample[burn_in4:1e5,2]),
             mcdata = rho3_sample[burn_in4:1e5,2], burn = burn_in4)
plot(1:nrow(cs7), cs7, main = TeX('Cusum $x_1$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
plot(1:nrow(cs8), cs8, main = TeX('Cusum $x_2$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
autocorrelation(mcdata = rho3_sample[burn_in4:1e5,1], lag_i = lags)
autocorrelation(mcdata = rho3_sample[burn_in4:1e5,2], lag_i = lags)

plot(burn_in5:1e5, rho4_sample[burn_in5:1e5,1],
     type = 'l', ylab = TeX('$x_1^{(t)}$'), xlab = TeX('t'))
title(main = TeX('$\\rho = 0.4$, burn-in: 10,000'))
plot(burn_in5:1e5, rho4_sample[burn_in5:1e5,2],
     type = 'l', ylab = TeX('$x_2^{(t)}$'), xlab = TeX('t'))
cs9 <- cusum(theta_hat = mean(rho4_sample[burn_in5:1e5,1]),
             mcdata = rho4_sample[burn_in5:1e5,1], burn = burn_in5)
cs10 <- cusum(theta_hat = mean(rho4_sample[burn_in5:1e5,2]),
             mcdata = rho4_sample[burn_in5:1e5,2], burn = burn_in5)
```

```r
plot(1:nrow(cs9), cs9, main = TeX('Cusum $x_1$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
plot(1:nrow(cs10), cs10, main = TeX('Cusum $x_2$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
autocorrelation(mcdata = rho4_sample[burn_in5:1e5,1], lag_i = lags)
autocorrelation(mcdata = rho4_sample[burn_in5:1e5,2], lag_i = lags)

plot(burn_in6:1e5, rho5_sample[burn_in6:1e5,1],
     type = 'l', ylab = TeX('$x_1^{(t)}$'), xlab = TeX('t'))
title(main = TeX('$\\rho = 0.5$, burn-in: 7,000'))
plot(burn_in6:1e5, rho5_sample[burn_in6:1e5,2],
     type = 'l', ylab = TeX('$x_2^{(t)}$'), xlab = TeX('t'))
cs11 <- cusum(theta_hat = mean(rho5_sample[burn_in6:1e5,1]),
             mcdata = rho5_sample[burn_in6:1e5,1], burn = burn_in6)
cs12 <- cusum(theta_hat = mean(rho5_sample[burn_in6:1e5,2]),
              mcdata = rho5_sample[burn_in6:1e5,2], burn = burn_in6)
plot(1:nrow(cs11), cs11, main = TeX('Cusum $x_1$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
plot(1:nrow(cs12), cs12, main = TeX('Cusum $x_2$'),
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
autocorrelation(mcdata = rho5_sample[burn_in6:1e5,1], lag_i = lags)
autocorrelation(mcdata = rho5_sample[burn_in6:1e5,2], lag_i = lags)

### Problem 3
# part (a)
mcdata1 <- scan(file.choose())
mcdata2 <- scan(file.choose())

par(mfrow = c(2,1), mar=c(5,6,4,2) + 0.1)
plot(1:length(mcdata1), mcdata1, main = 'MCdata1',
     type = 'l', ylab = TeX('$x_1^{(t)}$'), xlab = TeX('t'))
plot(1:length(mcdata2), mcdata2, main = 'MCdata2',
     type = 'l', ylab = TeX('$x_1^{(t)}$'), xlab = TeX('t'))
dev.off()

# part (b)
burn_in1 <- 5001
burn_in2 <- 10001
theta_hat1 <- mean(mcdata1[burn_in1:length(mcdata1)])
theta_hat2 <- mean(mcdata2[burn_in2:length(mcdata2)])

cusum <- function(theta_hat=theta_hat1, mcdata=mcdata1, burn=burn_in) {
  cusum_data <- matrix(NA, nrow = length(mcdata) - burn + 1)
  cusum_diff <- matrix(NA, nrow = length(mcdata) - burn + 1)
  mcdata <- mcdata[burn:length(mcdata)]
  for (i in 1:length(mcdata)) {
    cusum_diff[i] <- mcdata[i] - theta_hat
    cusum_data[i] <- sum(cusum_diff, na.rm = TRUE)
  }
  return(cusum_data)
}

cs_1 <- cusum(theta_hat = theta_hat1, mcdata = mcdata1, burn = burn_in1)
cs_2 <- cusum(theta_hat = theta_hat2, mcdata = mcdata2, burn = burn_in2)

par(mfrow = c(2,1))
plot(1:nrow(cs_1), cs_1, main = 'Cusum MCdata1',
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
plot(1:nrow(cs_2), cs_2, main = 'Cusum MCdata2',
     type = 'l', ylab = 'Cusum', xlab = TeX('t'))
dev.off()
```

```r
# part (c)
C_i <- function(mcdata=mcdata1, i=40) {
  n <- length(mcdata)
  xbar <- mean(mcdata)
  autocovariance_matrix <- matrix(NA, nrow = (n - i))
  for (t in 1:(n - i)) {
    autocovariance_matrix[t] <- (mcdata[t] - xbar) *
      (mcdata[t + i] - xbar)
  }
  mean(autocovariance_matrix)
}

C_0 <- function(mcdata=mcdata1) {
  xbar <- mean(mcdata)
  mean((mcdata - xbar)^2)
}

autocorrelation <- function(mcdata=mcdata1, lag_i=40) {
  c_0 <- C_0(mcdata = mcdata)
  acf_matrix <- matrix(NA, nrow = lag_i)
  for (i in 1:lag_i) {
    c_i <- C_i(mcdata=mcdata, i=i)
    acf_matrix[i] <- c_i / c_0
  }
  plot(1:lag_i, acf_matrix, ylim = c(0,1),
       pch = 19, xlab = 'Lags', ylab = 'acf')
  segments(x0 = 1:lag_i, y0 = 0, x1 = 1:lag_i, y1 = acf_matrix)
  abline(h = 0)
}

lags <- 40
par(mfrow = c(2,1))
acf1 <- autocorrelation(mcdata = mcdata1, lag_i = lags)
title('ACF of MCdata1')
acf2 <- autocorrelation(mcdata = mcdata2, lag_i = lags)
title('ACF of MCdata2')
dev.off()

### Problem 4
# part (a)
mc1 <- scan(file.choose())
mc2 <- scan(file.choose())
mc3 <- scan(file.choose())
mc4 <- scan(file.choose())
mc5 <- scan(file.choose())
mc6 <- scan(file.choose())
mc7 <- scan(file.choose())

gelman_rubin <- function(D=0, L=1000, first_part=1000) {
  J <- 7
  mc <- cbind(mc1, mc2, mc3, mc4, mc5, mc6, mc7)
  mc <- mc[1:first_part,]
  mc <- mc[(D+1):(D+L),]

  xbar_j <- (1 / L) * colSums(mc)
  xbar <- mean(xbar_j)

  B <- (L / (J - 1)) * sum((xbar_j - xbar)^2)
  sj2 <- (1 / (L - 1)) * apply(matrix(1:J), 1, function(x) {
    sum((mc[,x] - xbar_j[x])^2)
  })
  W <- mean(sj2)
  R <- (((L - 1) / L) * W + (1 / L) * B) / W
  return(c(B, W, sqrt(R)))
}

# part(a-f)
gelman_rubin(D = 0, L = 1000)
gelman_rubin(D = 500, L = 500)
gelman_rubin(D = 0, L = 500, first_part = 500)
gelman_rubin(D = 250, L = 250, first_part = 500)
gelman_rubin(D = 0, L = 50, first_part = 50)
gelman_rubin(D = 25, L = 25, first_part = 50)
```