E.Kim
I6073884

Report: Independence Analysis in a Markov Chain Model

Maastricht University	
School of Business & Economics	
Place & date:	Maastricht, 25/01/2018
Name, initials:	E.Kim
ID number:	I6073884
Study:	Econometrics & OR
Course code:	EBS2043
Team mate:	W.Xu
Tutor name:	Nalan Bastürk
Assignment:	Introduction to Software in Econometrics

eunji.kim@maastrichtuniversity.nl

E.Kim
I6073884

Table of contents

1. Introduction

For the assignment of the course 'Bayesian econometrics', with my partner (Wen Xu), we have decided to work on the second topic 'Independence Analysis in a Markov Chain Model'. To have different results, we have set different random seeds (123 for me) as well as different conditional probabilities of the first order Markov chain model p_{ij}. The purpose of this paper is to analyze if Metropolis-Hasting algorithm works well on the simulated data from the first order Markov chain model.

The structure of this paper is as follows. The first part explains how the data simulation on a first order Markov Chain works. Next, the paper examines how the log-likelihood function, prior density, posterior density and candidate function are formulated for the model. Then, the application of the Metropolis-Hasting algorithm is done by the programming language R and the result from the MH algorithm is compared with the true value. To conclude, it investigates whether the occurrence of events is independent of their past as well as the result of our research question is explained in detail.

2. Data simulation on the first order Markov Chain model

First order Markov Chain model for observations y_t, t = 1,...,T is $pr(y_t = j \mid y_{t-1} = i) = p_{ij}$. To start with, I randomly set $p_{11} = 0.8$, $p_{12} = 0.2$, $p_{21} = 0.2$, $p_{22} = 0.8$. Note that $p_{i1} + p_{i2} = 1$. Hence, there are two free parameters and we will use p_{11} and p_{21} for our analysis and calculate p_{12} and p_{22} at the end. Then, to simulate 1000 observations, each draw will be selected randomly from the uniform distribution and if the new draw is less or equal to 0.8, we assign the value of the observation same as old draw and different otherwise. Since the simulated data is applied in loglikelihood, posterior and MH algorithm, we keep them in the vector.

3. Selection of prior, candidate density and deriving likelihood, posterior density

Prior is a probability function that represents the beliefs of the data before the observed value of the data. We used flat prior as it is recommended to use in the question as well as it makes the probability neutral. Firstly, we draw the values of p_{11} and p_{21} from the uniform (0,1) distribution. Then, if p_{11} and p_{21} are between 0 and 1, the value of the prior is 1 and 0 otherwise.

Next, the likelihood function and posterior density are formulated. Firstly, we derive the conditional probability of each observation (likelihood of single observation) then derive the likelihood function of the entire sample. We take the log of likelihood as the value of the likelihood is close to zero. Since posterior density is proportional to the product of prior density and likelihood function and we take log-likelihood, posterior is formulated simply as the sum

of log of prior and log-likelihood. The equation of likelihood function of single observation is as follows.

$$p\left(y_t\middle|p_{ij}\right)=\begin{cases}p_{11}^{2-y_t}(1-p_{11})^{y_t-1} & if\ y_{t-1}=1\\ p_{21}^{2-y_t}(1-p_{21})^{y_t-1} & if\ y_{t-1}=2\end{cases}.$$

Lastly, we need to select candidate density for the application of the Metropolis-Hasting algorithm later. Our initial thought on the candidate density selection was the uniform distribution. However, the MH algorithm rejects a lot of part of the draw if it is uniform distribution as the candidate values are not always close to true value. Hence, we decided to draw from the normal distribution where the mean is the approximate Maximum Likelihood Estimate. That is, the frequency of the event where 1 jumps to 1 and 1 jumps to 2 based on the simulated data (The algorithm is explained in detail under part 4.1).

Since the draw from the normal distribution candidate will be more likely to be the true value, we conclude that it makes more sense to draw from the normal distribution instead of uniform distribution.

4. Metropolis-Hasting algorithm

4.1. General comment on the MH algorithm

Metropolis-Hasting algorithm is a Markov Chain Monte Carlo(MCMC) method that the candidates are drawn from a given probability distribution. It is recommended to use for our first order Markov Chain model assignment since the draw of candidate of MH algorithm is based on the previous draw in each iteration and the candidate draws converge to the target distribution. The steps of this algorithm are as follows:

Once we have the draw from the candidate distribution, we calculate the acceptance probability of the new draw and we accept or reject the new draw depending on the probability alpha. The function of the acceptance probability is written below.

$$\alpha(X_{t-1},y)=\ \min\left\{\frac{\pi(y)q(y,X_{t-1})}{\pi(X_{t-1})q(X_{t-1},y)}\ ,1\right\}.$$

Then, we draw a random value from the uniform (0,1) distribution and compare with the alpha. If the random value is smaller than alpha, we accept the new draw. Otherwise, we keep the old draw.

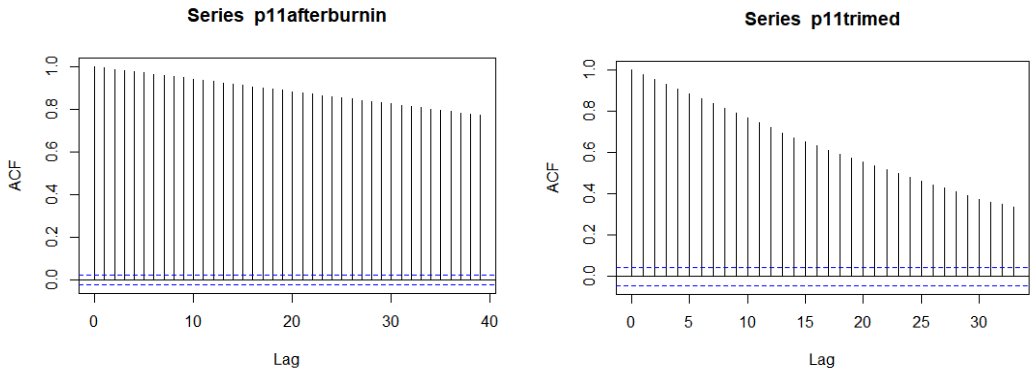
As it is mentioned above in the paper, we have selected the normal distribution for our candidate density. Now, we need begin with choosing the starting value of the candidate. We had a discussion whether we should choose the initial (starting) value of the candidate arbitrarily such as $p_{11} = 0.6$, $p_{12} = 0.4$ or choose based on the value that it is most likely to

get from the true value. We tried both method and decided to work with the second idea. To do so, we created a vector to store the difference between past and current observation from our simulated data. Hence, the length of the vector is 999. If the difference is 1, it means that the previous value is 2 and current value is 1. In a same manner, if the difference is -1, the previous value is 1 and current value is 2. Then, we calculated the frequency where 1 shows up in the simulated data as well as where 2 shows up. Based on the previous steps, we have found the frequency of the event where 1 jumps to 1 and 2 jumps to 1 and use it as the starting value of p_{ij} in the MH algorithm. As a result, our starting value of p11 is 0.7920168 and p21 is 0.1889313. Note that we name this algorithm as 'an approximate MLE' in this paper.

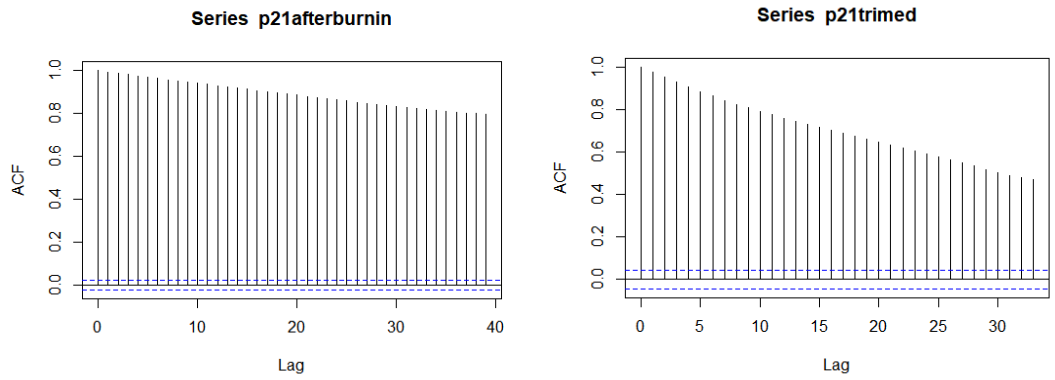
Total simulation of the MH algorithm is 10,000 and the number of accepted draws in the application of MH algorithm is 53. Therefore, the acceptance rate is 0.0053. The acceptance rate is very low because the starting point is too accurate to start and therefore the algorithm accepts the new draws for few times then starts to reject the new draws as the past draws are good enough.

4.2. Burn in and trimming

After running the MH algorithm for 10,000 times, we burn in the first 2,000 observations to eliminate the effect of dependence on the starting value. Then, we use the trim method to eliminate the autocorrelation due to the property of the Markov Chain model. In this step, we only select 1 observation out of 4. Notice that from the ACF graph before trimming, autocorrelation with 40 lags are about 0.8. But it is notable that the level of autocorrelation between lags are decreasing as the number of lags increases after trimming. However, it is not possible to eliminate them.



Graph 1&2: ACF of p_{11} before and after trimming



Graph 3&4: ACF of p_{21} before and after trimming

4.3.Calculation of p_{11}, p_{21}, p_{12} and p_{21} and the comparison with the true value

We can conclude from the mean and variance of p_{ij} that MH algorithm works well in our case as they are very close to the true value. The following tables indicate the results and the quantile can be found in the appendix.

	Mean of p_{11}	Variance of p_{11}	Mean of p_{21}	Variance of p_{21}
MH algorithm	0.7915882	0.0003031576	0.1887277	0.0003258232
True value	0.8	0	0.2	0

	Mean of p_{12}	Variance of p_{12}	Mean of p_{22}	Variance of p_{22}
MH algorithm	0.2084118	0.0003031576	0.8112723	0.0003258232
True value	0.2	0	0.8	0

5. Investigation if the event is independent of its past

The last part of this assignment, we are asked to create variables $w_1 = \frac{p_{11}}{p_{21}}$, $w_2 = \frac{p_{12}}{p_{22}}$ to check whether the occurrence of events is independent of their past. If $w_1 = w_2 = 1$, then we conclude that it is indeed independent of the past and dependent otherwise.

In our case, it is observable that the mean of both w_1 and w_2 are significantly different from one and credible intervals do not include one. Therefore, we can conclude that the occurrence of events is dependent of their past.

The following table shows the result of our case.

	Mean	Variance	Credible interval
w_1	4.235082	0.1908048	(3.65223, 5.001851)
w_2	0.2570706	0.0005239063	(0.2262293, 0.2952902)

6. Conclusion

Throughout the analysis on our research question, the paper concludes that Metropolis-Hasting algorithm indeed works well on the simulated data from the first order Markov Chain model and we have observed the effects of burn-in and trimming method in the MH algorithm. Furthermore, it is concluded that the occurrence of events in dependent of their past on the first order Markov Chain model.

7. Appendix

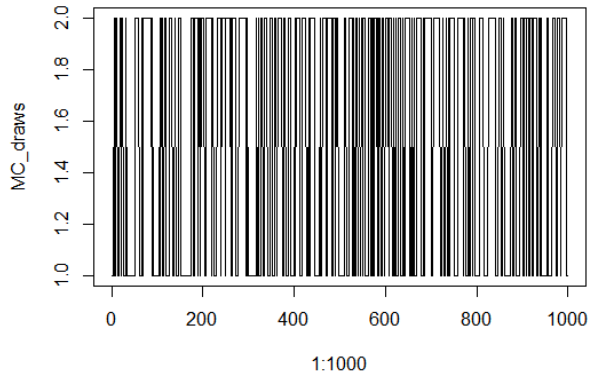


Figure 1: 1000 simulated data on the first order of Markov Chain model

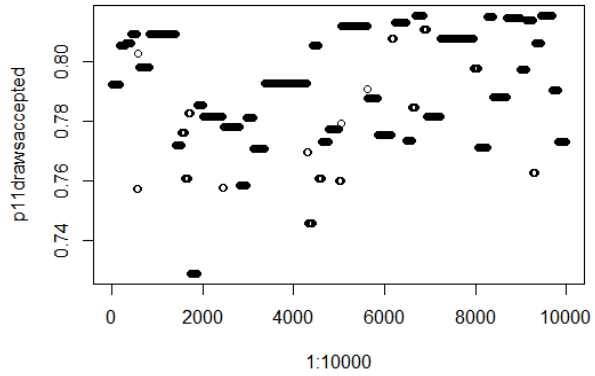


Figure 2: plotted p_{11} after the application of MH algorithm

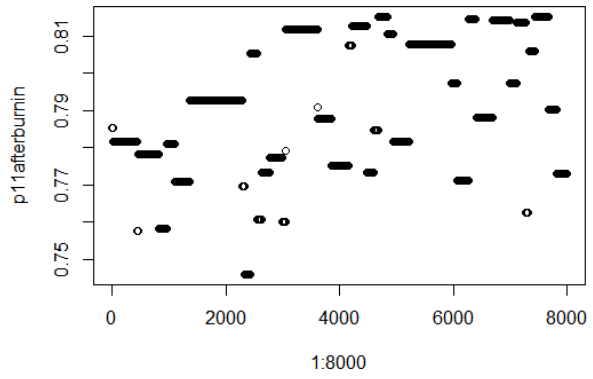


Figure 3: plotted p_{11} after the burn-in

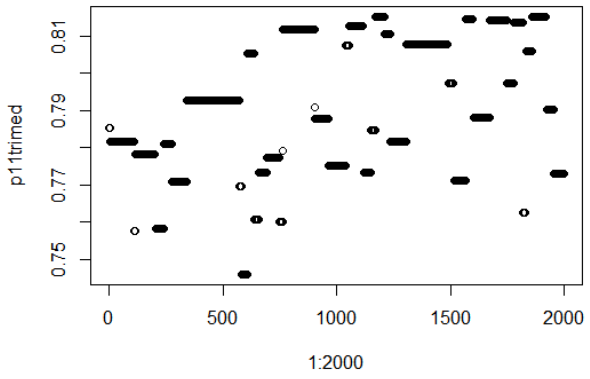


Figure 4: plotted p_{11} after burn-in & trimming

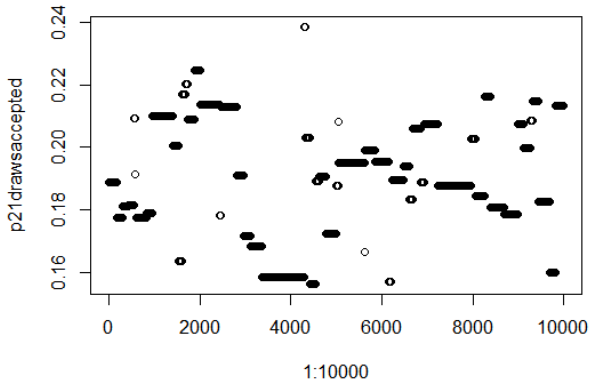


Figure 5: plotted p_{21} after the application of MH algorithm

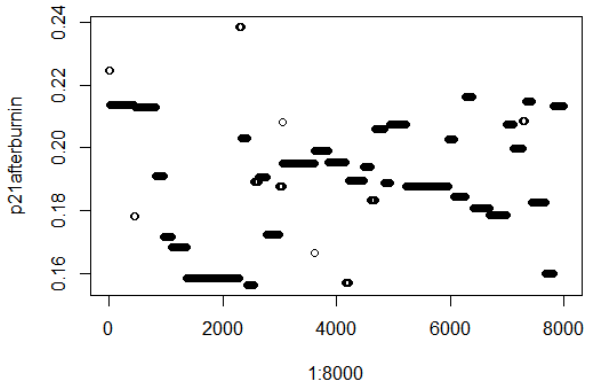


Figure 6: plotted p_{21} after the burn-in

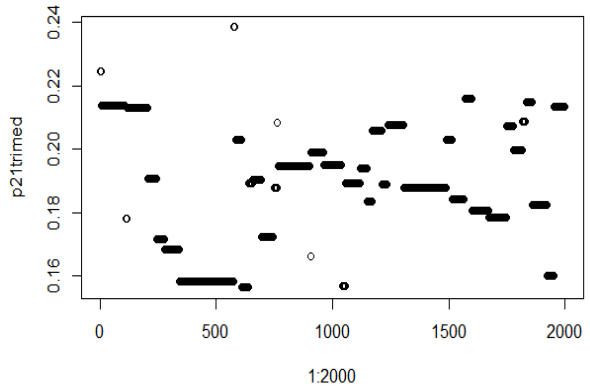


Figure 7: plotted p_{21} after burn-in & trimming

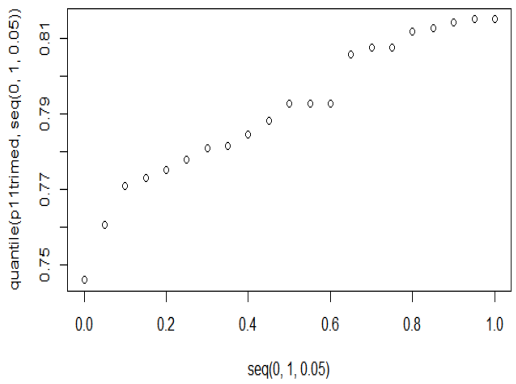


Figure 8: quantile of p_{11}

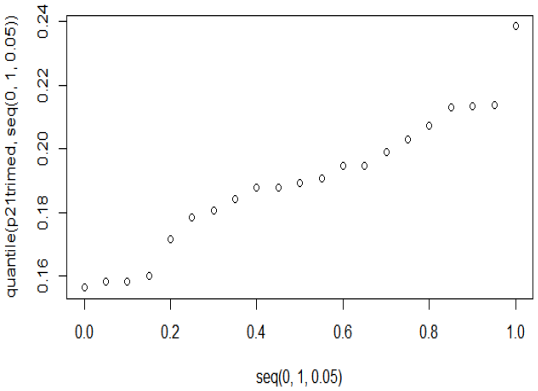


Figure 9: quantile of p_{21}

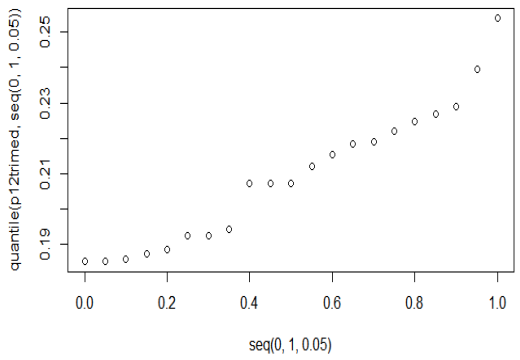


Figure 10: quantile of p_{12}

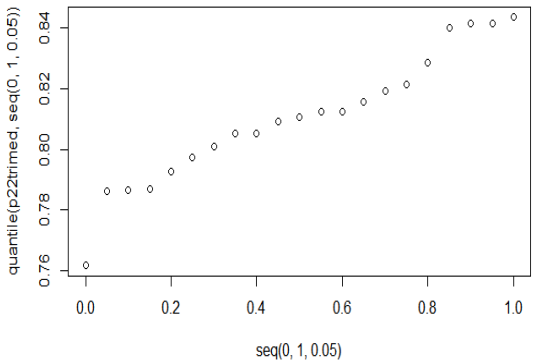


Figure 11: quantile of p_{22}

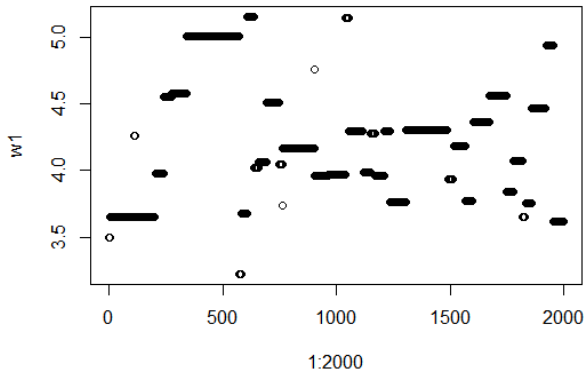


Figure 12: plotted w_1

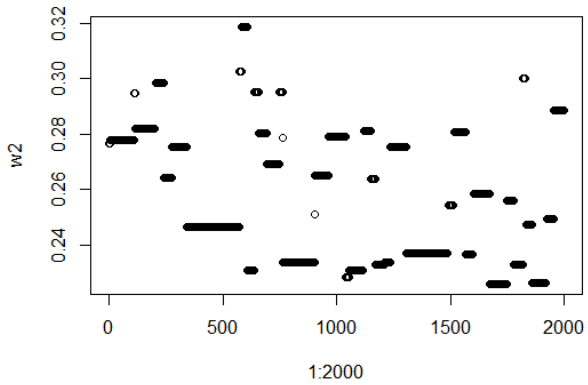


Figure 13: plotted w_2