FLINDERS
UNIVERSITY
ADELAIDE
AUSTRALIA

Bayesian Inference of Non-homogeneous Gaussian Hidden Markov Models

by

Shin SATO, *BSc (Hons)*
College of Science and Engineering

April 2019

A thesis presented to the
Flinders University of South Australia
in total fulfilment of the requirements for the degree of
Master of Science (Mathematics)

Adelaide, South Australia, 2019

Contents

List of Figures

List of Tables

List of Symbols

Below are some details of the symbols and functions used throughout this thesis unless stated otherwise.

$\{\}$	Empty set
$\mathbb{N} = \mathbb{N}_{>0}$	The set of natural numbers
$\mathcal{Y}$	Observation space
Ω	Hidden state space
$\mathbb{R}$	The set of real numbers
$\mathbb{R}^m$	Euclidean m column space
$\mathbb{R}^{m\times n}$	Vector space of $m \times n$ matrix
$\mathbb{R}_{>0}$	The set of positive real numbers
Θ	Parameter space
$\mathrm{Cov}(\cdot)$	Covariance
$\det(A)$	The determinant of a $d \times d$ square matrix A
$\mathrm{E}[\cdot]$	The expected value
$\exp(\cdot)$	Exponential function
$\Gamma_d(\cdot)$	The multivariate gamma function of order d
$\gcd\{\cdot\}$	The greatest common divisor
$\log(\cdot)$	The natural logarithm
$\log_2(\cdot)$	The logarithm with base 2
$\|\| \cdot \|\|$	The Euclidean norm
$\Pr(\cdot)$	Probability measure
$\mathrm{tr}(A)$	The trace of a $d \times d$ square matrix A

$\mathcal{G}(\alpha, \beta)$	Gamma distribution with shape and rate parameters α and β, respectively
$\mathcal{N}(\mu, \sigma^2)$	Univariate normal/Gaussian distribution with mean μ and variance σ^2
$\mathcal{N}_d(\boldsymbol{\mu}, \Sigma)$	Multivariate normal/Gaussian distribution with mean $\boldsymbol{\mu}$ and positive-definite covariance matrix Σ
$\mathcal{U}(a, b)$	Uniform distribution over the interval a to b such that $a, b \in \mathbb{R}$
$\mathcal{W}^{-1}(\gamma, \Phi)$	Inverse Wishart distribution with degrees of freedom γ and scale matrix Φ

Abstract

This thesis specifically aims at investigating a non-homogeneous Gaussian hidden Markov model (NHGHMM) under both univariate and multivariate settings. The model is known to be able to capture non-homogeneity of transition probabilities between the hidden states. This advantage for the model can provide more flexibility on a predictive ability in statistical inference.

A Bayesian approach was implemented for parameter estimation by proposing several advanced Markov chain Monte Carlo (MCMC) algorithms to contribute to an estimation aspect of the models. From a Bayesian perspective, the current literature on multivariate extensions of an NHGHMM is scarce.

In terms of efficiency and convergence, I discovered that the three proposed MCMC algorithms, the *adaptive Metropolis*, *symmetric delayed rejection adaptive Metropolis*, and *multiple-try adaptive Metropolis* algorithms, were more efficient and achieved faster convergence than the standard Metropolis-Hastings algorithm under the univariate setting. For performances of parameter estimation, those three MCMC algorithms were able to provide more reliable estimates compared to the conventional methods. In addition, a case study was conducted by using the multiple-try adaptive Metropolis algorithm, which was the most efficient MCMC algorithm amongst the three algorithms.

Likewise, the three proposed MCMC algorithms were more efficient compared to the standard Metropolis-Hastings algorithm through the simulation studies within the multivariate setting. For this particular simulation study, those proposed algorithms achieved satisfactory convergence in the same number of iterations. In terms of performances of parameter estimation, those three proposed algorithms generated reliable

estimates. Overall, the case studies were mainly satisfactory, but one of them left a conundrum where the MCMC convergence was unachievable, creating a limitation with the model.

The methodologies for such models have not been fully explored in the literature, and thus, my original contributions are comprised of the proposed MCMC algorithms and a research contribution to the multivariate NHGHMMs. A set of R$^{©}$ and C++ code was specifically written to validate the proposed MCMC algorithms through the extensive simulation studies which were conducted in this thesis, making up part of my original contributions.

Certification

I certify that this thesis does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due reference is made in the text.

As requested under Clause 14 of Appendix D of the *Flinders University Research Higher Degree Student Information Manual* I hereby agree to waive the conditions referred to in Clause 13(b) and (c), and thus

- Flinders University may lend this thesis to other institutions or individuals for the purpose of scholarly research;
- Flinders University may reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

SSato

Shin Sato
2019.04.27
16:04:41 +09'00'

Acknowledgements

This thesis represents the culmination of my studies as a Master of Science by research student in the College of Science and Engineering at Flinders University.

I am indebted to my supervisors, Dr Darfiana Nur, and Dr Shaowen Qin for their encouragement, patience, guidance, and readily accessible advice.

Finally, I am ever grateful for the support of my family and friends.

Chapter 1

Introduction

REAL-world time series processes normally generate observable values which can be characterised as signals, consisting of three main conditions. First, the signals can be categorised into either discrete or continuous variables. Examples of these are the discrete observable values to the extent of four bases in a DNA sequence: adenine, cytosine, guanine, and thymine (Boys and Henderson, 2004); or the continuous observable values as the monthly US 3-month treasury bill rates (Hamilton, 1988; Meligkotsidou and Dellaportas, 2011). Second, the signal source can be either stationary (i.e. the signal properties do not vary over time) or non-stationary (i.e. the signal properties vary over time). And finally, the signal can be comprised of either a single source or multiple distinct sources (Rabiner, 1989).

Characterising such real-world signals has been a fundamental problem of interest in terms of signal models. One possible signal model is called an *observable Markov model* (Rabiner, 1989). This type of model assumes a system which may be described as being in one of a set of distinct states. These states follow a Markov process, and the stochastic process of observed values is equivalent to the process of states. Although an observable Markov model is able to capture some characteristics of the signals, it is too restrictive for more complicated signal modelling (Rabiner, 1989).

Due to the limitation of an observable Markov model, an extension of the model to a more intricate one was attempted by allowing the observation to be a probabilistic function of the state (Rabiner, 1989). Hence, the resulting model is called a *hidden Markov model* (HMM). According to Rabiner (1989), first of all, the model is far richer in mathematical structure, and therefore, it can form the theoretical base in a wide range of applications. Secondly, the model works extremely well with applications to several important areas such as signal processing (Rabiner, 1989; Rabiner and Juang, 1986).

An ordinary HMM is referred to as a *homogeneous hidden Markov model.* This type of HMM has a restrictive attribute due to constant transition probabilities of the model. To relax this restrictive characteristic, the first *non-homogeneous* hidden Markov model was proposed by Diebold et al. (1994). The time-varying transition probabilities of such a model are able to depend on the underlying stochastic process of interest.

Filardo and Gordon (1998) also discovered that constant transition probabilities of hidden states are too restrictive for estimating the expected duration of a business cycle. By allowing non-homogeneity for the transition probabilities, the result of expected duration calculations was found to be very intuitive, as when the end of a recession approaches, the expected duration of the recession plummets (Filardo and Gordon, 1998).

According to Spezia (2006), non-homogeneous hidden Markov models can be applied to many different real-world data sets as long as exogenous variables for time-varying transition probabilities are available. In other words, non-homogeneous hidden Markov models can provide more meaningful results than the standard homogeneous HMMs.

In this thesis, I specifically deal with a non-homogeneous hidden Markov model whose distribution of the observed values is Gaussian. Hence, a *non-homogeneous Gaussian hidden Markov model* (NHGHMM) will be the core for the remainder of this thesis.

There are two main methodologies in statistical inference, namely frequentist and Bayesian approaches. For HMMs, both approaches have been extensively used to estimate parameters of interest in the literature.

The very first practical application of frequentist inference on an HMM was developed by Rabiner and Juang (1986). Analyses of speech recognition have seen remarkable improvements since Rabiner and Juang (1986)'s work. Ever since Rabiner and Juang (1986) demonstrated the potential of the approach, other fields such as signal processing (Crouse et al., 1998), finance (Diebold et al., 1994), meteorology (Bellone et al., 2000; Hughes et al., 1999; Robertson et al., 2004), and array comparative genomic hybridisation analysis (Fridlyand et al., 2004) also utilised frequentist inference for parameter estimation of their own HMMs.

Although the frequentist approach had been considered to be effective for an HMM, Filardo and Gordon (1998) raised its limitations to the model where frequentist inference may be faced with technical issues. For instance, *all* possible permutations of unobserved states must be considered in frequentist methods. In addition, assessing of the uncertainty of the measure of the unobserved state is quite complicated (Filardo and Gordon, 1998). Hence, the other main methodology, a Bayesian approach, was proposed for parameter estimation of HMMs to avoid these problems.

The two primary differences between Bayesian and frequentist approaches are as follows:

- Use of prior distributions such as informative priors in a Bayesian approach is able to generate more intuitive results as opposed to a frequentist approach.
- A Bayesian approach treats parameters of interest as random variables whereas the frequentist approach aims at estimating the parameters as unknown constants. Bayesian inference provides the uncertainties of the parameters through their marginal posterior distributions.

To estimate parameters of interest, the expectation-maximisation (EM) algorithm is often implemented as maximum likelihood estimation (MLE) in frequentist inference. As for HMMs, in addition to mixture models, the Baum-Welch algorithm is especially used for parameter estimation, which follows the fashion of the EM algorithm (Rabiner, 1989; Rabiner and Juang, 1986).

A class of algorithms formulated within a Bayesian framework is called a *Markov chain Monte Carlo* (MCMC) algorithm. The most common MCMC algorithm is known as the Metropolis-Hastings algorithm which was invented by Metropolis et al. (1953) and was later generalised by Hastings (1970). In spite of its popularity, the impact of the Metropolis-Hastings algorithm on statistics had been minute until the early 1990's (Brooks et al., 2011). Nevertheless, the algorithm has opened the door for Bayesian inference to allow for the evaluation of more sophisticated statistical models. According to Brooks et al. (2011), however, the Metropolis-Hastings algorithm requires a choice of proposal distributions, and this choice defines the algorithm's efficiency in terms of autocorrelation functions and effective sample size.

Hence, more advanced MCMC algorithms will be applied in this thesis to improve the choice of proposal distributions. The adaptive Metropolis algorithm is one of the MCMC algorithms where it is able to learn about an optimal scaling of the proposal distribution (Brooks et al., 2011). The algorithm was invented by Haario et al. (2001) and allowed the computer to "learn" better parameter values automatically. This autonomous adaptation of proposal distributions is also able to improve convergence of the MCMC algorithm which is a crucial concept for parameter estimation of the model.

In addition, the delayed rejection Metropolis-Hastings algorithm plays an important role for a Bayesian analysis of NHGHMMs in this thesis. If acceptance probabilities of a series of proposed values are too low, then a chain of MCMC samples stays in the same parameter for a long period of time. The delayed rejection Metropolis-Hastings aims at postponing this rejection process for a proposed parameter (Mira et al., 2001).

As Mira et al. (2001) stated, the delayed rejection Metropolis-Hastings algorithm itself is not necessarily better than the Metropolis-Hastings algorithm; however, the adaptation of proposal distributions is able to take advantage of newly acquired information about parameters of interest. Hence, the symmetric delayed rejection adaptive Metropolis algorithm which was proposed by Haario et al. (2006); Mira et al. (2001) will be implemented in the hope of improving the acceptance probabilities.

Finally, the multiple-try Metropolis-Hastings algorithm will be introduced for greater improvement in terms of efficiency. Liu et al. (2000) stated that the standard Metropolis-Hastings algorithm can easily get trapped in a local mode, and hence, slow convergence of the algorithm. Thus, increasing "searching region" of proposal distributions directly leads to them being less localised. On the other hand, this "brute-force" way results in a very small acceptance probability (Liu et al., 2000). Therefore, the multiple-try Metropolis-Hastings algorithm was invented to tackle this conflict of interest by proposing multiple draws at one time.

With a collection of the proposed MCMC algorithms, I investigate whether Bayesian analyses of both the univariate and the multivariate NHGHMMs be able to generate more fruitful results in the field of parameter estimation of NHGHMMs.

Having discussed the above, the main objective of this thesis is the development and implementation of several proposed algorithms for parameter estimation of NHGHMMs. The detailed aims of this thesis are comprised of the following items:

(a) To present Bayesian inference of NHGHMMs under a univariate setting.

(b) To present Bayesian inference of NHGHMMs under a multivariate setting.

(c) For each of the models mentioned in (a) and (b), to propose more efficient MCMC algorithms to estimate the parameters of the models.

(d) To demonstrate:

   (1) the MCMC convergence, and

   (2) the performance of parameter estimation

   of each combination of the algorithms mentioned in (c) in parameter estimation against both simulation and case studies.

It is crucial to emphasise that the current literature about a multivariate NHGHMM is scarce. Moreover, a Bayesian analysis of such a model using Metropolis-Hastings type algorithms for parameter estimation has been avoided due to technical difficulties, except for Heaps et al. (2015) and Spezia et al. (2017). Those Metropolis-Hastings type algorithms are usually replaced by Pólya-Gamma data augmentation for inferring transition probabilities (Holsclaw et al., 2016, 2017; Koki et al., 2018). Finally, a combination of the model with advanced MCMC algorithms such as the adaptive Metropolis, the symmetric delayed rejection adaptive Metropolis, and the multiple-try adaptive Metropolis algorithms is currently unavailable in the literature. Hence, this thesis introduces a new Bayesian framework for the multivariate NHGHMM.

Having stated the aims above, the original contributions of this thesis are the following:

(i) The main algorithm (Algorithm 6) which consists of the forward filtering backward sampling algorithm (Algorithm 7), the Gibbs sampler (Algorithm 11), and either the adaptive Metropolis (Algorithm 8), the symmetric delayed rejection adaptive Metropolis (Algorithm 9), or the multiple-try adaptive Metropolis (Algorithm 10) algorithms for the univariate NHGHMM (Section 3.5).

(ii) The main algorithm (Algorithm 12) which consists of the forward filtering backward sampling algorithm (Algorithm 13), the Gibbs sampler (Algorithm 17), and either the adaptive Metropolis (Algorithm 14), the symmetric delayed rejection adaptive Metropolis (Algorithm 15), or the multiple-try adaptive Metropolis (Algorithm 16) algorithms for the multivariate NHGHMM (Section 4.5).

(iii) Comparisons of (i) Algorithm 8, Algorithm 9 & Algorithm 10 (Section 3.6), and (ii) Algorithm 14, Algorithm 15 & Algorithm 16 (Section 4.6) to the standard Metropolis-Hastings algorithm by using:

  (1) Geweke's diagnostics, effective sample size, and autocorrelation functions for the MCMC convergence, and

  (2) point estimates, 95% credible intervals, and coverage for the performances of parameter estimation.

(iv) A set of R© (R Core Team, 2016) and C++ code to validate the proposed algorithms in extensive simulation studies. The source code is listed in Appendices E and F for univariate and multivariate settings, respectively.

The contents of this thesis are organised as follows. Chapter 2 contains a comprehensive literature review of the models of interest, an HMM, GHMM, and NHGHMM, in addition to a discussion of the MCMC algorithms and their algorithmic efficiencies. Chapters 3 and 4 are composed of similar contents, but under univariate and multivariate settings, respectively. Both chapters begin with the introduction of each model and then describe the models in detail along with establishing prior distributions, likelihood functions, and both joint and conditional posterior distributions. Most importantly, the proposed MCMC algorithms are implemented under both univariate and multivariate settings for simulation studies. After confirming the validity of the MCMC algorithms, case studies of real-world data sets are conducted. Finally, Chapter 5 presents the conclusion of this thesis, as well as a discussion of future work.

Chapter 2

Literature Review

Hidden Markov model (HMM) is an extremely useful statistical model which is applicable to a wide range of time series problems such as speech recognition (Rabiner, 1989; Rabiner and Juang, 1986), signal processing (Crouse et al., 1998), DNA sequence segmentation (Boys and Henderson, 2004; Boys et al., 2000; Churchill, 1989), array comparative genomic hybridisation analysis (Fridlyand et al., 2004; Rueda et al., 2013), electrocardiogram analysis (Coast et al., 1990), protein folding (Stigler et al., 2011), precipitation model (Ailliot et al., 2009; Heaps et al., 2015; Hughes et al., 1997, 1999), business cycle analysis (Diebold et al., 1994; Filardo and Gordon, 1998), and many others. Whilst the model has been applied in science, engineering, finance, econometrics, and some other fields, the model's mathematical structure is surprisingly straightforward.

An HMM, in general, is explained by introducing a few applications, and other types of the HMMs are also described in Section 2.1. Then, the core of statistical inference in this thesis, Bayesian inference, is explained in Section 2.2. In Section 2.3, a collection of algorithms which attempt to estimate parameters of interest is explained in detail where the algorithms are to be constructed in a Bayesian framework. Given the aforementioned algorithms, Section 2.4 aims at introducing more advanced algorithms to accelerate convergence in a specific model. A collection of these proposed algorithms is fabricated by my original contribution in this thesis. Section 2.5 describes an intrinsic problem of the HMM, a label switching phenomenon. This issue needs to be addressed in order to validate parameter estimation of HMMs in a Bayesian framework. Furthermore, convergence of parameter estimation is discussed in Section 2.6 since it is as important as dealing with the label switching phenomenon. Finally, the methodology of measuring performance for each proposed algorithm is presented in Section 2.7 to argue the validity of the proposed algorithms.

2.1 Hidden Markov Model

A mathematical concept regarding the HMM was first developed by Baum and Eagon (1967); Baum and Petrie (1966); Baum et al. (1970). They investigated statistical estimation of probabilistic functions of Markov chains. The mathematics behind the HMM naturally gives rise to different types of HMMs, and each type of the HMM is defined by different mathematical architecture.

In fact, speech signals are impossible to be modelled meaningfully without considering such temporal variation. In the late 1980's, Rabiner and Juang (1986) and Rabiner (1989) studied characteristics of speech signals by considering a more complicated signal where a sinusoidal wave was embedded with noise and temporal variations of the signal. They successfully implemented the HMM on the problem in order to estimate the sine wave parameters (amplitude, frequency, and phase). In an attempt to model the utterance of speech signals, the HMM was adequately effective to describe the progressive change in signal characteristics in each state (Rabiner and Juang, 1986).

HMMs have also been applied in genetics. Since the Human Genome Project was initiated, rapid growth in the database of human DNA sequences was seen. The base composition of most DNA sequences was known to be heterogeneous, and sophisticated statistical techniques were required to analyse the vast amount of such DNA sequence data (Boys et al., 2000; Churchill, 1989). Observations in the model are four discrete bases: adenine, cytosine, guanine, and thymine (i.e. $\{a, c, g, t\}$). Markov chains with stationary transition probabilities have been used to model such naturally occurring DNA sequences, although the Markov chain approach is unable to describe local structure of heterogeneity of the sequences. Churchill (1989), Boys et al. (2000), and Boys and Henderson (2004) investigated the data to model the type of homogeneous segment at each location in the DNA sequence by using the HMM. They showed that the HMM was able to model the hidden label process which governed base transitions. As opposed to the Markov chain approach, the HMM was able to capture the heterogeneous structure of DNA sequences.

Not only was an HMM implemented in computational biology, but the model was also used in other engineering fields. In signal processing, it was known that the wavelet-based statistical signal processing techniques lacked usefulness for many real-world signals since such models treated wavelet coefficients as though they were independent. Such techniques simply disregarded the statistical dependencies between the coefficients. Crouse et al. (1998) developed a new framework for statistical signal processing based on wavelet-domain HMMs to tackle the problem where the statistical dependencies were also considered. Their newly developed framework proved its utility and flexibility with applications in many types of signal modelling problems. In light of this new approach, they believed that other related fields such as speech recognition and artificial intelligence would lead to even more sophisticated and accurate, yet still tractable and robust wavelet-domain HMMs for signal processing (Crouse et al., 1998).

As such, applications of the HMM are ubiquitous in all kinds of disciplines. Zucchini and MacDonald (2009) states that HMMs are incredibly flexible models for especially discrete-valued time series data such as categorical series or series of counts. Nevertheless, observations of the model can take a form of either discrete or continuous values. In fact, some applications mentioned above are classified as a specific type of the HMMs. Hereafter, an HMM whose distribution of observed variables is the Gaussian distribution will be the centre of this research interest, and thus, the mathematical structure behind such a model is described in Section 2.1.1.

2.1.1 Gaussian Hidden Markov Model

In this section, several papers about Gaussian HMMs were reviewed. Array comparative genomic hybridisation is a technique to measure DNA copy number aberrations quantitatively and map them onto genome sequences directly (Fridlyand et al., 2004). The methodology in their study was an HMM approach where the observations were continuous random variables, $\log_2$ ratios of a test intensity T, and reference intensity

R such that $\log_2(T/R)$. Their method proved to be useful by finding real alterations, and it could be applied to investigate breast cancer data for the purpose of revealing the relationship between the frequencies and types of copy number alterations with defects in the mechanisms (Fridlyand et al., 2004).

Likewise, Rueda et al. (2013) used the HMM to study the mechanisms of DNA copy number alterations, but they let the number of the hidden states remain unknown. In other words, the number of hidden states was regarded as a random variable instead of the fixed number in other array comparative genomic hybridisation analysis literature. The HMM with an unknown number of hidden states was more effective from a biological point of view in terms of parameter estimation (Rueda et al., 2013).

Cardiac arrhythmia is an event where a change in cardiac rhythms which causes irregular heartbeats and deteriorates pumping efficiency. Certain ventricular arrhythmias are known to show vulnerability to life-threatening conditions. Arrhythmia is identifiable in the electrocardiogram, although P waves are hardly detectable in practice (Coast et al., 1990). There is a system which is able to detect P waves described by Jenkins et al. (1979). The system attempts to insert an oesophageal electrode with attached lead wire into a patient's throat and position it near the atria. It is possible to obtain high-amplitude P waves by this method; however, the system has been limited due to the discomfort of the oesophageal electrode and lead wire. Coast et al. (1990) used the HMM approach in electrocardiogram arrhythmia analysis to detect low-amplitude P waves as well as the QRS complexes. Their results showed that the accurate detection of the low-amplitude P waves in ambulatory electrocardiogram recordings (Coast et al., 1990).

Protein folding is a complex mechanism to understand in terms of size and structure of proteomes (protein genomes), and unfolding the mystery of the specific mechanism requires a more intricate experimental method (Stigler et al., 2011). Stigler et al. (2011) used ultrastable high-resolution optical tweezers incorporated with an HMM to investigate the folding transitions of single calmodulin (calcium-modulated protein) molecules. It follows that Stigler et al. (2011) concluded the model with 4 states provided the greater level of detail about protein folding.

In fact, these applications utilise a statistical model called *Gaussian hidden Markov model* (GHMM), and the model especially revolutionised modelling computational biology as well as an HMM for discrete observed values. Hence, the GHMM is an essential stochastic model from that perspective. The model of interest is described as follows.

Mathematical structure

A GHMM is a stochastic model in which a sequence of real-valued observations is dependent on the contemporary sequence of finite hidden states (Rabiner and Juang, 1986; Spezia, 2006). Here, let $S = (S_1, S_2, \dots, S_n)$ be a sequence of m finite hidden states, where $m, n \in \mathbb{N} \setminus \{1\}$ and $S_t \in \Omega = \{1, 2, \dots, m\}$, for any $t \in \{1, 2, \dots, n\}$. Moreover, let $Y = (Y_1, Y_2, \dots, Y_n)$ be a sequence of conditionally independent real-valued random variables given all hidden states (Rydén et al., 2008).

The stochastic process of the hidden states S_t, for any $t \in \{1, 2, \dots, n\}$, follows a Markov process. For the stochastic process, there are two assumptions considered in general.

The first assumption of the Markov process is that transitions between the hidden states are determined by the underlying Markov chain. Following Zucchini and MacDonald (2009), the discrete-time p^{th} order Markov chain is given as follows:

$$\Pr(S_t = s_t \,|\, S_{t-1} = s_{t-1}, \dots, S_1 = s_1) = \Pr(S_t = s_t \,|\, S_{t-1} = s_{t-1}, \dots, S_{t-p} = s_{t-p}), \tag{2.1}$$

where $S_t \in \Omega$ for any $t \in \{2, 3, \dots, n\}$. Equation (2.1) states that a probability of the state at time t is dependent on the previous p states such that $\{s_{t-1}, s_{t-2}, \dots, s_{t-p}\}$. The GHMM is based on the assumption where a Markov chain satisfies *Markov property* if Equation (2.1) holds (Zucchini and MacDonald, 2009). For this thesis, the 1st order Markov chain will be explained as follows:

$$\Pr(S_t = s_t \,|\, S_{t-1} = s_{t-1}, \dots, S_1 = s_1) = \Pr(S_t = s_t \,|\, S_{t-1} = s_{t-1}), \tag{2.2}$$

where $S_t \in \Omega$ for any $t \in \{2, 3, \dots, n\}$. By Equation (2.2), it is possible to construct a transition probability matrix Q as follows:

$$Q = \begin{pmatrix} q_{1,1} & q_{1,2} & \cdots & q_{1,m} \\ q_{2,1} & q_{2,2} & \cdots & q_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ q_{m,1} & q_{m,2} & \cdots & q_{m,m} \end{pmatrix} \in \mathbb{R}^{m\times m}.$$

In this case, Q is an $m \times m$ right stochastic matrix such that

$$\begin{aligned} q_{i,j} &:= \Pr(S_t = j \mid S_{t-1} = i), \text{ for all } (i,j) \in \Omega^2 \\ q_{i,j} &\geq 0, \text{ for all } (i,j) \in \Omega^2 \\ \sum_{j=1}^{m} q_{i,j} &= 1, \text{ for all } i \in \Omega. \end{aligned}$$

The second assumption of a GHMM is a *state dependent process* where a probability distribution of a realisation y_t is only dependent on the current state s_t, regardless of the history of a hidden state sequence (Zucchini and MacDonald, 2009). It can be mathematically represented as follows:

$$\Pr(y_t \mid y_{1:t-1}, s_{1:t}) = \Pr(y_t \mid s_t) =: f(y_t \mid s_t), \quad \text{for any } t \in \{1, 2, \ldots, n\}, \tag{2.3}$$

where $y_{u:v} = (y_u, y_{u+1}, \ldots, y_v)$ for any $u, v \in \{1, 2, \ldots, n\}$ such that $u \leq v$. Moreover, $y_{u:v} = \{\}$ whenever $u > v$.

With the two assumptions altogether, the characteristic of a GHMM is defined as a bivariate stochastic process, $\{(Y_t)_{t=1}^n, (S_t)_{t=1}^n\}$ (Rydén et al., 2008). This can be graphically shown by Figure 2.1. This probabilistic graphical model simply represents the observed random variable Y_t which is conditionally independent on the corresponding hidden state S_t, for any $t \in \{1, 2, \ldots\}$.

The transition probability matrix for all states $i, j \in \Omega$ is denoted by Q, and the $m \times m$ matrix of emission densities at each $t \in \{1, 2, \ldots, n\}$ is denoted by $F(t)$. In addition, the initial distribution is denoted by ρ such that

$$\rho = (\rho_1, \rho_2, \ldots, \rho_m) = (\Pr(S_1 = 1), \Pr(S_1 = 2), \ldots, \Pr(S_1 = m)).$$

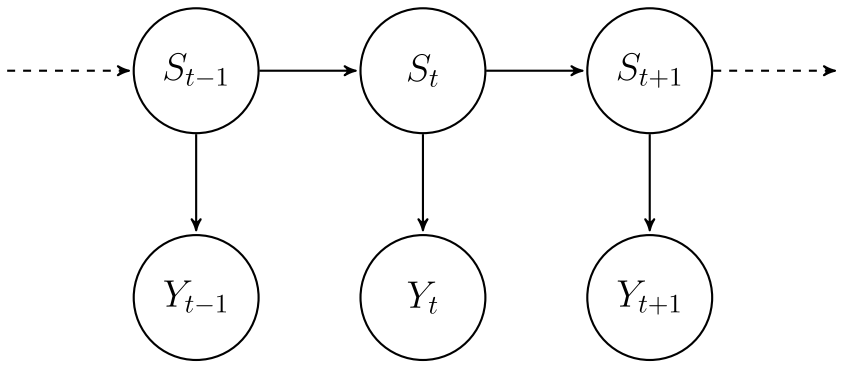


Figure 2.1: Depiction of an HMM

Equation (2.3) is sometimes called an *emission density* of the GHMM. Again, it is possible to construct a time-dependent matrix $F(t)$ whose dimension is $m \times m$ such that

$$F(t) = \begin{pmatrix} f(y_t \mid S_t = 1) & 0 & \cdots & 0 \\ 0 & f(y_t \mid S_t = 2) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & f(y_t \mid S_t = m) \end{pmatrix},$$

for any $t \in \{1, 2, \dots, n\}$ (Zucchini and MacDonald, 2009). Formally, a GHMM is defined as a triple of the following parameters:

$$\mathcal{M} = (Q, F(t), \rho).$$

It is assumed that the probability density function of every observed value y_t, for any $t \in \{1, 2, \dots, n\}$ is Gaussian distributed with a pair of state specific parameters. Hence, the model is called *Gaussian* hidden Markov model (Spezia, 2006, 2010).

Since the hidden state sequence is *unobservable*, it can only be made "observation" through the stochastic process which produces a sequence of observed values (Rabiner, 1989; Rabiner and Juang, 1986; Rydén et al., 2008; Zucchini and MacDonald, 2009).

Given the model $\mathcal{M}$, it remains to contemplate whether the observation of the GHMM is either univariate or multivariate. Hereafter, two types of observed values are considered and formulated into mathematical expressions.

Univariate GHMM

Define $y = (y_1, y_2, \dots, y_n)$ as a sequence of observed values $y_t \in \mathbb{R}$, for any $t \in \{1, 2, \dots, n\}$. Mathematically, a stochastic process of the observed values y_t, for any $t \in \{1, 2, \dots, n\}$, follows that

$$y_t = \mu_i + \varepsilon_i, \tag{2.4}$$

where $y_t \in \mathbb{R}$ denotes the observations at time t, for any $t \in \{1, 2, \dots, n\}$; for any state $i \in \Omega$, $\mu_i \in \mathbb{R}$ denotes the i^{th} state specific mean, and $\varepsilon_i \sim \mathcal{N}(0, \sigma_i^2)$ denotes the Gaussian noise with the i^{th} state specific variance σ_i^2 (Spezia, 2006).

For example, if a GHMM is defined as a 3-state model (i.e. $m = 3$), then Figure 2.2 simulates the bivariate stochastic process $\{(y_t)_{t=1}^n, (s_t)_{t=1}^n\}$. The resultant is called *trellis diagram* (Rabiner, 1989). The solid arrows represent realisations of the Markov process, and it is called *Viterbi path*. Figure 2.2 is able to help visualise a graphical representation of the GHMM.

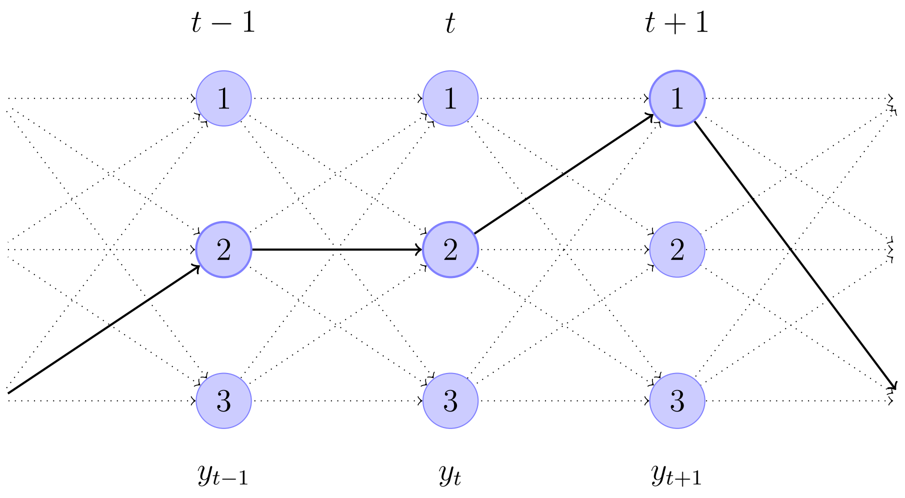


Figure 2.2: Trellis diagram of a 3-state HMM. The thick arrows represent the Viterbi path.

The fundamental assumption of an HMM where observations are governed by hidden/latent discrete states can achieve the following main objectives in the analysis (Box et al., 2015):

(i) *Description of the Dynamics*

to comprehend the features of the time series by inspecting intervals of observations,

(ii) *Modelling*

to explore an appropriate tool to reproduce the original data in order to obtain a fitted series closely approximated to the actual one, and

(iii) *Prediction*

to investigate what will happen if time shifts forwards and aim at predicting possible future values by a proposed probability distribution.

Multivariate GHMM

A univariate GHMM can indisputably be extended to a multivariate GHMM for which the observations take multivariate continuous random variables such that $\mathbf{Y}_t = (Y_t^1, Y_t^2, \ldots, Y_t^d) \in \mathbb{R}^d$, for any $t \in \{1, 2, \ldots, n\}$. The stochastic process of the observations in multivariate GHMMs preserves the conditional independence of the observations on the hidden states which is identical to that of a univariate GHMM. Spezia (2010) stated that this doubly stochastic process $\{(\mathbf{Y}_t)_{t=1}^n, (S_t)_{t=1}^n\}$ is considered as a "signal plus noise" model as follows:

$$\mathbf{Y}_t = \boldsymbol{\mu}_i + \boldsymbol{\varepsilon}_i, \tag{2.5}$$

where $\mathbf{Y}_t \in \mathbb{R}^d$ denotes observations when $S_t = i \in \Omega$ for any $t \in \{1, 2, \ldots, n\}$; then, $\boldsymbol{\mu}_i \in \mathbb{R}^d$ denotes a vector of m state specific means. Furthermore, $\boldsymbol{\varepsilon}_i \sim \mathcal{N}_d(\mathbf{0}, \Sigma_i)$ denotes the Gaussian noise, $\mathbf{0}$ is a d-dimensional zero vector and positive definite covariance matrices $\boldsymbol{\Sigma} = (\Sigma_1, \Sigma_2, \ldots, \Sigma_m)$ where $\Sigma_i \in \mathbb{R}^{d\times d}$, for any state $i \in \Omega$. A pair of parameters for the multivariate Gaussian distribution (i.e. $\boldsymbol{\mu}_i$ and $\boldsymbol{\varepsilon}_i$) corresponds to signal and noise, respectively. Hence, it follows that $\mathbf{Y}_t \,|\, S_t = i \sim \mathcal{N}_d(\boldsymbol{\mu}_i, \Sigma_i)$ (Spezia, 2006).

So far, the hidden states of these models are assumed to be independent of discrete-time t. They are called *homogeneous* GHMMs. Howbeit, they lack the predictive ability since constant transition probabilities are restrictive in many applications (Diebold et al., 1994; Filardo and Gordon, 1998). In practice, transitions of hidden states are considered to be time-varying. Thus, more sophisticated models whose transition probabilities are treated as time-varying are introduced.

2.1.2 Non-homogeneous Gaussian Hidden Markov Model

Regional weather types can be regarded as hidden states, and in practice the HMM was implemented by Ailliot et al. (2009) to model the spatiotemporal evolution of daily rainfall. Hughes et al. (1997, 1999) were the pioneers to formulate the framework of an HMM in precipitation. They developed a stochastic model which conditions local precipitation on broad-scale atmospheric circulation (i.e. the downscaling model). The model is a generalisation of existing weather state models, and one assumes that the transition matrix of the hidden states depends on atmospheric circulation data, namely a set of observed covariates (Hughes et al., 1997). A crucial advantage of their model is that one can incorporate those atmospheric data which are thought to affect the observations (Hughes et al., 1997, 1999). This feature makes the model different from most existing weather state models.

Estimation of expected business cycle durations is the centre of attention for economic agents and macroeconomists since it is pivotal for them to forecast when a recession is going to end (Filardo and Gordon, 1998). In fact, the underlying business cycle is unobservable. Hence, Filardo and Gordon (1998) implemented the HMM as an appropriate statistical tool to forecast expected business cycle durations. Diebold et al. (1994) pointed out that those models with constant transition probabilities failed to forecast expected durations accurately. In other words, the HMM with time-varying transition probabilities is more capable of capturing the complex behaviour of underlying economic fundamentals (Diebold et al., 1994; Filardo and Gordon, 1998).

Spezia (2006) introduced model selection with a Bayesian framework to determine the unknown number of states in a non-homogeneous Gaussian hidden Markov model (NHGHMM) via a logistic function. The framework allowed for the time-varying transition probabilities to be dependent on stochastic variables. It was claimed that whenever dynamic covariates are available, the method shows robust results for parameter estimation and model selection in various types of NHGHMMs.

A non-homogeneous hidden Markov model in the multivariate observable random variable setting gives rise to Hughes et al. (1999) for which the conditional distribution of the observation given the hidden state was the Bernoulli distribution. The Bernoulli distribution's support is either 0 or 1, and hence, the observed values are not continuous random variables. Nevertheless, Hughes et al. (1999)'s model had a framework of non-homogeneity in the hidden Markov chain. This leads to modifying the model to the Bayesian framework of the multivariate *Gaussian* non-homogeneous hidden Markov models.

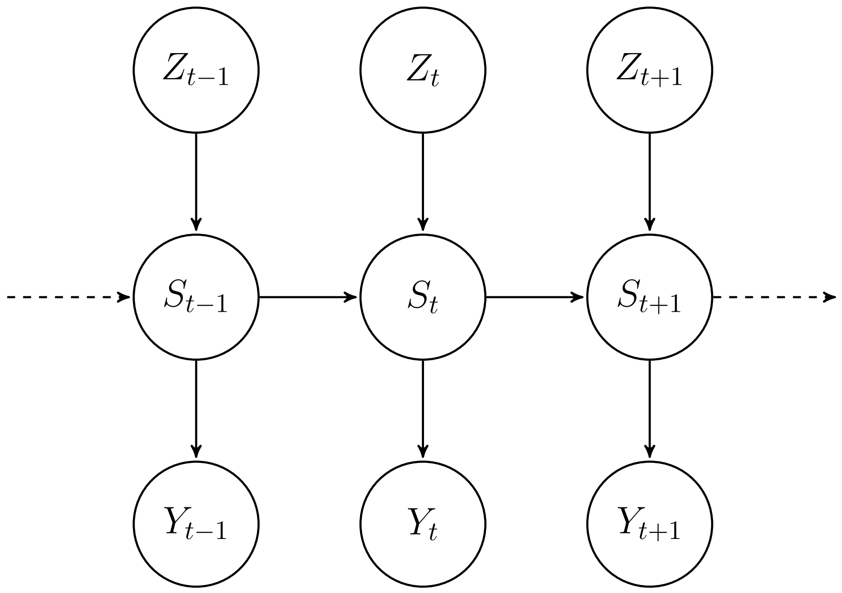


Figure 2.3: Depiction of a non-homogeneous HMM

Figure 2.3 shows a graphical model of a non-homogeneous HMM. Unlike Figure 2.1, a set of exogenous variables $\{Z_t\}$ governs the transition probabilities of the hidden states, namely $\Pr(S_t \mid S_{t-1})$, for any $t \in \{2, 3, \ldots\}$, and hence, the non-homogeneity of the model.

Given the graphical model of a non-homogeneous HMM, the mathematics behind the model is elaborated as follows.

Mathematical structure

The crucial difference between homogeneous and non-homogeneous GHMMs is a Markov process of the hidden states. Recall Equation (2.2). Instead of the transition probability matrix Q, the NHGHMM introduces another type of a transition probability matrix Q^t whose probabilities are also dependent on discrete-time $t \in \{2, 3, \ldots, n\}$.

Mathematically, it is possible to define such a matrix Q^t as follows:

$$Q^t = \begin{pmatrix} q^t_{1,1} & q^t_{1,2} & \cdots & q^t_{1,m} \\ q^t_{2,1} & q^t_{2,2} & \cdots & q^t_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ q^t_{m,1} & q^t_{m,2} & \cdots & q^t_{m,m} \end{pmatrix} \in \mathbb{R}^{m \times m}.$$

Furthermore, Q^t is a right stochastic matrix for any $t \in \{2, 3, \ldots, n\}$ such that

$$\begin{aligned} &\Pr(S_t = j \mid S_{t-1} = i) = q^t_{i,j}, \text{ for all } (i,j) \in \Omega^2 \\ &q^t_{i,j} \geq 0, \text{ for all } (i,j) \in \Omega^2 \text{ and } t \in \{2, 3, \ldots, n\} \\ &\sum_{j=1}^{m} q^t_{i,j} = 1, \text{ for all } i \in \Omega \text{ and } t \in \{2, 3, \ldots, n\}. \end{aligned}$$

Univariate NHGHMM

As for the univariate NHGHMM, the stochastic process of the model follows Equation (2.4). It is identical to that of a univariate GHMM, except a Markov process of the hidden states is now dependent on discrete-time $t \in \{2, 3, \ldots, n\}$. This dependence is driven by a set of exogenous variables $\{Z_t\}$ which pertains to the non-homogeneity of the model.

In finance, for example, the fundamentals such as revenue, earnings, assets, liabilities, and growth are often regarded as exogenous variables for longitudinal data of interest (Diebold et al., 1994; Filardo and Gordon, 1998).

Multivariate NHGHMM

Undeniably, multivariate observed values may also be introduced in an NHGHMM such that a stochastic process of the observation follows Equation (2.5).

Given the bivariate stochastic processes, a generic model $\mathcal{L}$ for the NHGHMM is defined as follows:

$$\mathcal{L} = (Q^t, F(t), \rho).$$

In fact, the literature about *multivariate non-homogeneous Gaussian* hidden Markov models is scarce. The aforementioned authors, Ailliot et al. (2009); Hughes et al. (1997, 1999), intensively worked on non-homogeneous hidden Markov models, but none of their work was related to the multivariate Gaussian distributions for the observed values. Heaps et al. (2015) attempted to model rainfall data in the UK by introducing a collection of Lamb weather types as *discrete* exogenous variables. Their work is much more relevant to this thesis since the model introduced the multivariate log-normal distribution, in addition to a Bayesian approach. Henceforth, Heaps et al. (2015)'s work will be looked at closely.

Equation (2.5) is able to represent a multivariate NHGHMM where the observations are generated by unknown parameters $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma}, Q^t)$. The methodologies of its parameter estimation include: a two-state HMM whose transition probabilities depend on exogenous variables through a logistic function (Diebold et al., 1994), the model then demonstrates a correlation between the transition probabilities and the covariates via probit models (Raymond and Rich, 1997), and the highly complex HMM with non-homogeneous Markov chains defined on a state space with an unknown number of hidden states whose parameters described by autologistic models (Hughes et al., 1999).

2.2 Bayesian Inference

In this thesis, Bayesian inference will be used to infer parameters of interest in NHGHMMs since it is capable of handling technical problems encountered by frequentist inference (Filardo and Gordon, 1998). Bayesian inference has been implemented in a wide range of applications such as artificial intelligence (Russell et al., 1995), machine learning (Bishop, 2006; Murphy, 2012), robotics (Thrun et al., 2005), econometrics (Putnam and Quintana, 1994; Quintana, 1992), psychology (Edwards et al., 1963; Myung and Pitt, 1997; Rouder et al., 2009), and many others. Recently, it has successfully drawn much attention from various studies since the advent of high-performance computing machines realised the practicality of Bayesian inference.

That being said, the fundamental theorem in Bayesian inference is reviewed in the following section.

2.2.1 Bayes' Theorem

The Reverend Thomas Bayes, an English clergyman, is known for the eponym who formulated the beta-binomial posterior distribution, which is a special case of the well-known theorem in probability and statistics, *Bayes' Theorem* (Stigler, 1982). Bayes left an obscure scholium, although he never published it during his lifetime. Bayes' Theorem had not been disputed at the time Richard Price editing and publishing Bayes' obscure scholium, *An Essay Towards Solving a Problem in the Doctrine of Chances* (Bayes et al., 1763), until Pierre-Simon Laplace formalised the theorem in *Théorie analytique des probabilités* (de Laplace, 1820; Laplace, 1986). There had been major critiques made by Sir Ronald Fisher, Karl Pearson, and Harold Jeffreys about the principal defect in Bayes' assumption since his wording was opaque and difficult to fathom. Nonetheless, his justification for the assumption, an unknown probability was uniformly distributed *a priori*, was interpreted very intuitively by Stigler (1982). Consequently, Bayes' much disputed assumption has now been renowned and accepted by modern statistical literature.

Consider two events A and B. Then, the conditional probability of A given B is defined as

$$\Pr(A \mid B) = \frac{\Pr(A \cap B)}{\Pr(B)}, \tag{2.6}$$

where $\Pr(B) \neq 0$ is assumed, and the joint probability of events A and B is denoted by $\Pr(A \cap B)$. Furthermore, the conditional probability of B given A is defined as

$$\Pr(B \mid A) = \frac{\Pr(A \cap B)}{\Pr(A)}. \tag{2.7}$$

Given Equations (2.6) and (2.7), the joint probability of A and B is given by

$$\Pr(A \cap B) = \Pr(A \mid B)\Pr(B) = \Pr(B \mid A)\Pr(A),$$

and therefore,

$$\Pr(A \mid B) = \frac{\Pr(B \mid A)\Pr(A)}{\Pr(B)}. \tag{2.8}$$

Equation (2.8) is now known as *Bayes' Theorem* or *Bayes' Rule*.

In general, the conditional distribution of parameters of interest given observations is defined as follows:

$$\pi(\theta \,|\, y) = \frac{f(y \,|\, \theta)p(\theta)}{f(y)} = \frac{f(y \,|\, \theta)p(\theta)}{\int_{\Theta} f(y \,|\, \theta')p(\theta')\, \mathrm{d}\theta'}, \tag{2.9}$$

where $\theta \in \Theta$ denotes the parameter of interest, and $y \in \mathcal{Y}$ denotes the observation.

In Equation (2.9), the term $\pi(\theta \,|\, y)$ is called a *joint posterior distribution*. It plays a significant role in Bayesian inference for parameter estimation. The term $f(y \,|\, \theta)$ is called a *joint likelihood function*. It is the core of statistics, especially frequentist statistics, for MLE which includes a class of algorithms such as the EM algorithm (Dempster et al., 1977). Also, note that a likelihood function need not be a probability distribution. The term $p(\theta)$ is called a *prior distribution*, and this gives prior beliefs about the parameters of interest. In Bayesian inference, the choice of this density is crucial as it influences the posterior distribution as well as the methodologies of inference. Finally, the term $f(y)$ is called a *marginal likelihood* or *normalising constant*.

The term $f(y)$ is a normalising constant of the joint posterior distribution, and in fact, the integration part follows the law of total probability. For more intricate statistical models, calculating marginal likelihoods involves multidimensional integration, and hence, computationally intensive. For this reason, Bayes' Theorem was historically impractical for statistical inference.

2.3 Markov Chain Monte Carlo Algorithms

The rise of computing machines opened the door for the Bayesian approach to enter revolutionary inferential statistics in the century and overcame the impracticality of Bayes' Theorem. Computing power incorporated with tools of random number generation and simulations made Bayesian inference to solve many intractable problems in inferential statistics. Metropolis and Ulam (1949) were the pioneers who challenged the dominant frequentist methods. They developed a novel method to evaluate multiple integrals approximately over multidimensional variables within intractable problems of physical sciences where classical and statistical mechanics had the limitations. The method is known as *Monte Carlo method* (Metropolis et al., 1953; Metropolis and Ulam, 1949).

As such, the Bayesian approach has been fundamentally entangled with the power of computing machines. Bayesian inference certainly allowed for statisticians to tackle intractable problems in many cases and sometimes it is preferred over frequentist methods in some areas.

Nevertheless, the standard Monte Carlo methods sometimes suffer from poor approximation to values of interest (Brooks et al., 2011; Metropolis et al., 1953; Metropolis and Ulam, 1949). Thus, one needs to define more sophisticated methodologies to achieve greater accuracy. Furthermore, to evaluate the joint posterior distribution in Equation (2.9) is often intractable since a closed form is unknown. Hence, a new numerical methodology which is different from the standard Monte Carlo methods is essential to facilitate Bayesian inference to maximise the joint posterior distribution in Equation (2.9).

The following sections are to present *Markov chain Monte Carlo* (MCMC) algorithms. The justification with regard to implementing a collection of MCMC algorithms is the development of underpinning theories such as rates of convergence, laws of large numbers and central limit theorems (Tierney, 1994). Indeed, Markov chain methods proved to be more efficient in exploring posterior distributions than the ordinary Monte Carlo methods (Brooks et al., 2011).

Furthermore, Mengersen et al. (1996) worked on providing necessary and sufficient conditions for the MCMC algorithms to converge at a geometric rate. They evaluated several different algorithms to explore the potential of the methods. Hereafter, a few of the most well-known MCMC algorithms are introduced as follows.

2.3.1 Metropolis-Hastings Algorithm

The most famous MCMC algorithm and probably one of the most essential statistical tools amongst scientists is called *Metropolis-Hastings algorithms*. The name of the algorithm was coined by Metropolis et al. (1953) and Hastings (1970). The methodology is considered to be originated from Metropolis et al. (1953), dealing with the problem where equations of state for chemical substances consisting of individual particles are described. In general, the method is suitable for fast computing machines to explore such chemical properties, and later the method proved useful in computational statistics and stochastic simulation (Gamerman and Lopes, 2006).

The standard Monte Carlo methods can be computationally expensive since the methods may require sampling from high dimensional probability distributions. Hence, Hastings (1970) and his student, Peskun (1973), worked on generalising the Metropolis algorithm to address the issue which was also pointed out by Metropolis et al. (1953).

A generalisation of the Metropolis algorithm involves utilising an irreducible Markov chain to obtain a sequence of samples. With a current parameter θ, a new parameter θ^* proposed by a proposal distribution $q(\cdot \mid \theta)$, and the target distribution $\pi(\cdot)$ as a stationary distribution are defined on the Markov chain. Note that this target distribution is proportional to a multiplicative constant of the joint posterior distribution in Equation (2.9). Then, the acceptance probability of the current parameter to the new parameter, $\theta \to \theta^*$, is defined as follows:

$$r = \frac{s(\theta^* \mid \theta)}{1 + \pi(\theta)q(\theta^* \mid \theta)/\pi(\theta^*)q(\theta \mid \theta^*)}, \tag{2.10}$$

where $s(\theta^* \mid \theta) = s(\theta^* \mid \theta)$ is a symmetric function so that $0 \leq r \leq 1$ (Brooks et al., 2011; Hastings, 1970). That is,

$$s(\theta^* \mid \theta) = \begin{cases} 1 + \pi(\theta)q(\theta^* \mid \theta)/\pi(\theta^*)q(\theta \mid \theta^*)\ , \text{ if } \pi(\theta^*)q(\theta \mid \theta^*)/\pi(\theta)q(\theta^* \mid \theta) \geq 1 \\ 1 + \pi(\theta^*)q(\theta \mid \theta^*)/\pi(\theta)q(\theta^* \mid \theta)\ , \text{ if } \pi(\theta^*)q(\theta \mid \theta^*)/\pi(\theta)q(\theta^* \mid \theta) \leq 1 \end{cases}.$$

Consequently, this ratio in Equation (2.10) is known as *Metropolis-Hastings ratio*, and it is considered as a generalised acceptance probability. Alternatively, the acceptance probability in Equation (2.10) may be expressed by

$$r = \min\left\{1, \frac{\pi(\theta^*)q(\theta \mid \theta^*)}{\pi(\theta)q(\theta^* \mid \theta)}\right\}.$$

The algorithm is known as *Metropolis-Hastings algorithm*. Nevertheless, such a sequence of samples obtained by the Metropolis-Hastings algorithm is usually correlated, and thus, more careful assessment of estimation error may be required than independent samples (Hastings, 1970).

The aforementioned algorithm is classified as the *random walk Metropolis-Hastings algorithm* since the proposal distribution is centred at the previous value of MCMC iterations. For example, at τ^{th} MCMC iteration, a proposed parameter of interest $\theta^{(\tau)}$ is distributed in the proposal distribution $q(\cdot \mid \cdot)$ such that $\theta^{(\tau)} \sim q(\cdot \mid \theta^{(\tau-1)})$. With that being said, the distribution of the proposed parameter is dependent on the previous parameter $\theta^{(\tau-1)}$.

There is another type of the Metropolis-Hastings algorithm called the *independent Metropolis-Hastings algorithm*. This type of the Metropolis-Hastings algorithm's proposal distribution is independent of the previous value of MCMC iterations. As for updating the parameter of interest, a proposed parameter is distributed in the proposal distribution $q(\cdot \mid \theta^{(\tau-1)}) = q(\cdot)$, which is independent of the previous parameter.

Each algorithm differs in terms of convergence properties. A review of these algorithms can be found in Mengersen et al. (1996).

It has been known for a while that the choice of the proposal distribution is crucial to rapid convergence of the Metropolis-Hastings algorithm (Brooks et al., 2011). As Metropolis et al. (1953) pointed out, if the maximum displacement of a proposed parameter from the previous one is too large, most moves will be rejected. Moreover, if too small, the configuration of the parameters will be insufficient (Brooks et al., 2011). Hence, the Metropolis-Hastings algorithm has its own limitation in terms of MCMC convergence.

Having said that, if conditional posterior distributions of parameters of interest have standard forms, then the Gibbs sampler can be implemented to simulate the posterior distribution.

2.3.2 Gibbs Sampler

A special case of the Metropolis-Hastings algorithm, *Gibbs sampler*, was developed by Geman and Geman (1984). The Gibbs sampler has gained its popularity since the paper was published by Geman and Geman (1984) for the study in image processing models (Casella and George, 1992). The original concept of Gibbs sampler is considered to be originated from Metropolis et al. (1953), and it was then combined with the improvement by Hastings (1970).

The algorithm aims at generating random variables from a marginal distribution indirectly to avoid cumbersome calculations of the density. Despite the description of the algorithm is simple and straightforward to illustrate, the mechanism which achieves the sampling scheme has been left unexplained to its full extent (Casella and George, 1992). Gibbs sampler allows avoiding expensive computations of the density and generating an easier sequence of Markov chain in which the posterior means converge to the stationary distribution (Casella and George, 1992).

The algorithm is extremely useful in Bayesian inference due to its ease of implementation. Furthermore, it has widely been applied in complex stochastic models with a very large number of variables (Casella and George, 1992; Gelfand and Smith, 1990). As for Bayesian inference, the method is used for generating posterior distributions.

To illustrate how the Gibbs sampler works, consider a given joint density function $f(\theta, \eta_1, \eta_2, \ldots, \eta_p)$. Calculating a marginal density of θ is performed by integrating out other variables, that is,

$$f(\theta) = \int \cdots \int f(\theta, \eta_1, \eta_2, \ldots, \eta_p)\, \mathrm{d}\eta_1 \mathrm{d}\eta_2 \ldots \mathrm{d}\eta_p, \tag{2.11}$$

where $\eta_1, \eta_2, \ldots, \eta_p \in \Theta$ is a set of dummy variables. Equation (2.11) is calculated in order to obtain the desired characteristics of the marginal density such as the mean or variance (Casella and George, 1992). Nevertheless, the integrations in Equation (2.11) are often difficult to compute in practice regardless of analytical or numerical methods. The Gibbs sampler avoids computing or approximating the difficult integrations to obtain $f(\theta)$. With a sufficient sample size, the Gibbs sampler effectively generates a sequence of $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(N)}$ from $f(\theta)$ without performing the integrations in Equation (2.11).

Let $\pi(\theta)$ be a joint posterior distribution of a vector of parameters $\theta = (\theta_1, \theta_2, \ldots, \theta_p)$. Then, define $\theta_{-i} = \{\theta_j : j \neq i\}$ for $i, j \in \{1, 2, \ldots, p\}$. Assuming full conditional densities for each component θ_i are known and will be given by $\pi(\theta_i \,|\, \theta_{-i})$ (Smith and

Roberts, 1993). Thus, a general scheme of the Gibbs sampler which samples from the joint posterior distribution of parameters and given observations can be outlined as follows.

Algorithm 1 Gibbs Sampler

```
1: Initialise a vector of parameters $\theta^{(1)}$
2: for $\tau = 2, 3, \dots, N$ do
3:     Draw $\theta_1^{(\tau)} \sim \pi(\theta_1 \,|\, \theta_{-1}^{(\tau-1)}, y)$.
4:     Draw $\theta_2^{(\tau)} \sim \pi(\theta_2 \,|\, \theta_1^{(\tau)}, \theta_3^{(\tau-1)}, \theta_4^{(\tau-1)}, \dots, \theta_p^{(\tau-1)}, y)$.
5:     Draw $\theta_3^{(\tau)} \sim \pi(\theta_3 \,|\, \theta_1^{(\tau)}, \theta_2^{(\tau)}, \theta_4^{(\tau-1)} \dots, \theta_p^{(\tau-1)}, y)$.
6:     $\vdots$
7:     Draw $\theta_p^{(\tau)} \sim \pi(\theta_p \,|\, \theta_{-p}^{(\tau)}, y)$.
8: end for
```

In theory, the distribution of a Gibbs sequence $(\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(N)})$ converges to the stationary distribution of interest as $N \to \infty$. Nevertheless, it is crucial to determine the number of iterations sufficiently large to achieve the convergence. A few methods (refer to Section 2.6 for the details) are suggested to assess the convergence of such a sequence (Cowles and Carlin, 1996) such as:

(i) treating the resulting sequence drawn by the Gibbs sampler as a time series (Geweke et al., 1991), and

(ii) detecting non-stationarity in the resulting sequence and eliminating "initial transient" in which the simulation does not start off in its stationary distribution (Heidelberger and Welch, 1983).

The Gibbs sampler appears to have slow convergence if highly correlated components are sampled individually. This slow convergence can be avoided and more efficient sampling may be performed by updating those correlated components in a block, although it requires a multivariate conditional distribution (Smith and Roberts, 1993).

One of the approaches to improve the convergence of a Gibbs sequence is to determine the initial k iterations to discard for which the Gibbs sequence to converge to the stationary distribution. This is called *burn-in* and the duration taken for the sequence in order to converge is known as *burn-in time* (Rosenthal, 1995).

For statistical problems in which a computationally expensive method is required, the empirical distribution of initial k iterations may converge to the stationary distribution in theory. From this point of view, one can determine the efficiency of the Gibbs sampler by monitoring this rate of convergence and the rate is dependent on how fast the Gibbs sampler explores the sample space (Casella and George, 1992).

Gelfand and Smith (1990) demonstrated a close relationship between the Gibbs sampler and the data augmentation algorithm proposed by Tanner and Wong (1987). Provided that the Gibbs sampling has access to full conditional distributions of $\theta_i \mid \theta_{-i}$ for each $i \in \{1, 2, \ldots, p\}$, a generalisation of the data augmentation algorithm may prove to be more efficient than the Gibbs sampler since the data augmentation algorithm allows sampling from correct $p \times (p-1)$ distributions, including p conditional posterior distributions at each iteration, on the other hand, the Gibbs sampler utilises only full conditionals for sampling. Thus, the data augmentation algorithm is equivalent to the Gibbs sampler in terms of the rate of convergence when some correction distributions, apart from full conditionals, are accessible (Gelfand and Smith, 1990).

Latent variables are an important concept in HMMs, and the data augmentation algorithm deals with the latent variables for its inference. The following describes a brief structure of the algorithm.

2.3.3 Data Augmentation

The data augmentation algorithm, also known as *the substitution algorithm* (Gelfand and Smith, 1990), is a standard mathematical tool to compute certain types of integral equations. Tanner and Wong (1987) demonstrated that algorithm could potentially be applied to statistical problems where observed data y might be augmented with hidden states s, that is, (y, s). The augmented data (y, s) sometimes are referred to as *complete data* whereas $\{y\}$ is called *incomplete data* in HMMs. Complete data consist of a set of missing and observed values (Tanner and Wong, 1987).

The data augmentation algorithm is essentially implemented for maximum likelihood estimation (MLE) or maximum a posteriori (MAP) estimation. Both methods are similar in terms of computing a point estimate. The MAP is based on a posterior distribution whereas the MLE is only based on a likelihood function. The MLE is usually performed by *the EM algorithm* further developed by Dempster et al. (1977)

where the algorithm involves two iterative steps such as:

(i) calculating expected values of the log likelihood function with the complete data, and

(ii) maximising the log likelihood function to update parameters of interest.

On the other hand, the data augmentation algorithm introduces substitutions of posterior and conditional densities involving integrable functions and integral transformation.

Consider $\theta \in \Theta$ as parameters of interest. The joint posterior distribution of interest is given by the following:

$$\pi(\theta \,|\, y) = \int_{\Omega} \pi(\theta \,|\, s, y) p(s \,|\, y) \, \mathrm{d}s, \tag{2.12}$$

where $\pi(\theta \,|\, y)$ denotes the joint posterior distribution of the parameters of interest θ given observed data y. Moreover, $\pi(\theta \,|\, s, y)$ denotes the conditional density of parameters given the hidden states s and observed data y, and $p(s \,|\, y)$ denotes the predictive density of the hidden states s given observed data y. In the same fashion as Equation (2.12), the predictive density of the hidden state can be expressed as follows:

$$p(s \,|\, y) = \int_{\Theta} p(s \,|\, \xi, y) p(\xi \,|\, y) \, \mathrm{d}\xi, \tag{2.13}$$

where $\xi \in \Theta$. Substitution of Equation (2.13) into Equation (2.12) yields the following equation:

$$h(\theta) = \int K(\theta, \xi) h(\xi) \, \mathrm{d}\xi, \tag{2.14}$$

where $K(\theta, \xi) = \int \pi(\theta \,|\, s, y) p(s \,|\, \xi, y) \, \mathrm{d}s$. Following the definition of Equation (2.14), suppose the integral transformation T which transforms any integrable function $g(\cdot)$ into an integrable function $Tg(\cdot)$ such that

$$Tg(\theta) = \int K(\theta, \xi) g(\xi) \, \mathrm{d}\xi.$$

With an initial approximation $h_1(\theta)$ to the posterior distribution $\pi(\theta \,|\, y)$, the iterative method for solving Equation (2.14) follows the following manner:

$$h_{t+1}(\theta) = (Th_t)(\theta). \tag{2.15}$$

In practice, an implementation of analytical calculation for solving Equation (2.15) is unavailable since the integrations in Equations (2.12) and (2.13) are difficult to be performed (Tanner and Wong, 1987). Nevertheless, the MCMC method offers the iterative method which allows the calculation of Equation (2.12) to be performed.

A data augmentation sampling scheme starts with an initial approximation $h_1(\theta)$ to $\pi(\theta \mid y)$ in the following manner.

Algorithm 2 Data Augmentation

```
1: for $t = 1, 2, \dots, T$ do
2:     Draw $\theta^* \sim h_t(\theta)$.
3:     Draw $s^* \sim p(s \mid \theta^*, y)$.
4:     Update the approximation $h_{t+1}$ to $\pi(\theta \mid y)$ by the following iteration:
           $h_{t+1}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \pi(\theta \mid s_i^*, y).$
5: end for
```

The calculation of $\pi(\theta \mid s, y)$ with any augmented data (y, s) is assumed to be tractable as it is a prerequisite for data augmentation sampling scheme (Tanner and Wong, 1987). With the larger number of n, generating θ^* and s^* which incorporates with updating h_{t+1} provides a closer approximation to Equation (2.15). Furthermore, the algorithm is insensitive to the number of samples since the average of $\pi(\theta \mid s, y)$ over the augmented data patterns will converge to the $\pi(\theta \mid y)$ by the iterative method above.

Howbeit, Tanner and Wong (1987) made a remark about the rate of convergence. It is linear and depends on the initial value $h_1(\theta)$ whenever the parameter space Θ is unbounded. Although this should not be regarded as a weakness of the method, data augmentation is not a desirable algorithm unless the parameter space Θ is compact (Tanner and Wong, 1987).

2.4 Advanced MCMC Algorithms

In the earlier sections, standard MCMC simulation techniques such as the Metropolis-Hastings algorithm, the Gibbs sampler, and the data augmentation algorithm were introduced. Reducing the correlation of MCMC samples, in addition to accelerating the convergence, often requires implementation of the following strategies (Gamerman and Lopes, 2006):

- long MCMC iterations to be run,
- sampling parameters of interest in a block, or
- reparametrisation.

Nonetheless, MCMC convergence for more complex statistical models is difficult to achieve with the standard MCMC algorithms mentioned above. For instance, it is essential for the standard random walk Metropolis-Hastings algorithm to adjust the proposal distribution in order to achieve the convergence in a Markov chain (Haario et al., 2001; Liu et al., 2000). As long as the reversibility of the Markov chain is preserved, it is a robust strategy to incorporate several MCMC schemes with the Metropolis-Hastings algorithm (Robert and Casella, 1999; Tierney, 1994).

That being said, there is one crucial theorem for MCMC algorithms to be included in Bayesian inference. *The Ergodic Theorem* is a fundamental theorem where it is able to justify the convergence of an MCMC algorithm (Tweedie, 1975). It shows that an MCMC sample average converges to the unique stationary distribution $\pi(\cdot)$, if it exists (Birkhoff, 1931). That is,

$$\lim_{N\to\infty} \frac{1}{N} \sum_{\tau=1}^{N} q\left(\theta^{(\tau)}\right) = \int q(\theta)\pi(\theta)\,\mathrm{d}\theta,$$

where $q(\cdot)$ is a proposal distribution. This theorem states that the following conditions must satisfy in order to achieve convergence to its unique stationary distribution (Tweedie, 1975) such that

- aperiodicity,
- positive recurrence, and
- irreducibility.

Aperiodicity

The *period* of a state i is defined as

$$d(i) = \gcd\{\nu \geq 1 : \Pr(\theta^{(\nu)} = i \,|\, \theta^{(1)} = i) > 0\}, \tag{2.16}$$

where gcd is the greatest common divisor, and $\nu \in \mathbb{N} \setminus \{1\}$ is the number of steps. Equation (2.16) indicates that the chain can only return to the same state i where the chain started from with the period of $d(i)$. A Markov chain is said to be *aperiodic* if all states have the period of 1 (Meyn and Tweedie, 2012). This is because it is not ideal for the chain to be stuck in some cycle between certain states.

Positive recurrence

A Markov chain is said to be *positive recurrent* if the mean recurrence time for state i, that is, t_i is finite (Meyn and Tweedie, 2012). It follows that

$$t_i = \sum_{\nu=1}^{\infty} \nu \cdot f_{ii}^{(\nu)} < \infty,$$

where $f_{ii}^{(\tau)}$ is the probability of first visit to state i at the ν^{th} step given that starting from state i. In other words, the chain will return to every state in a finite number of steps with probability 1.

Irreducibility

A Markov chain is said to be *irreducible* if all states communicate. That is,

$$i \to j \text{ and } j \to i, \text{ for all } i, j \in \mathcal{X},$$

where $\mathcal{X}$ is a state space. This means that, with a finite number of steps, any state can be reached from any other states (Meyn and Tweedie, 2012).

These three conditions allow the chain to converge to its unique stationary distribution $\pi(\cdot)$ from any given initial point where the chain starts. These properties lead to statistical efficiency in both theoretical and practical points of view, namely the

MCMC convergence.

In this thesis, the following three advanced MCMC algorithms are to be reviewed specifically.

2.4.1 Adaptive Metropolis Algorithm

The definition of the covariance matrix for parameters of interest $\{\theta^{(\nu)} \in \mathbb{R}^K : \nu \in \{1, 2, \ldots, \tau\}\}$ is as follows:

$$\operatorname{Cov}(\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(\tau)}) = \frac{1}{\tau - 1}\left(\sum_{\nu=1}^{\tau} \theta^{(\nu)}(\theta^{(\nu)})^\intercal - \tau\bar{\theta}^{(\tau)}(\bar{\theta}^{(\tau)})^\intercal\right), \tag{2.17}$$

where $\bar{\theta}^{(\tau)} = \sum_{\nu=1}^{\tau} \theta^{(\nu)}/\tau$. Here, the recursive formula for Equation (2.17) is derived in order to implement the adaptive method. It is easy to observe that an arithmetic mean is reduced to the recursive formula such that

$$\bar{\theta}^{(\tau)} = \frac{\tau - 1}{\tau}\bar{\theta}^{(\tau-1)} + \frac{1}{\tau}\theta^{(\tau)}. \tag{2.18}$$

With Equation (2.18) and algebraic manipulations, the recursive formula for Equation (2.17) will be as follows:

$$\begin{aligned}
&\operatorname{Cov}(\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(\tau)}) \\
&= \frac{1}{\tau - 1}\left(\sum_{\nu=1}^{\tau} \theta^{(\nu)}(\theta^{(\nu)})^\intercal - \tau\bar{\theta}^{(\tau)}(\bar{\theta}^{(\tau)})^\intercal\right) \\
&= \frac{1}{\tau - 1}\left[\sum_{\nu=1}^{\tau-1} \theta^{(\nu)}(\theta^{(\nu)})^\intercal + \theta^{(\tau)}(\theta^{(\tau)})^\intercal - \tau\left(\frac{\tau-1}{\tau}\bar{\theta}^{(\tau-1)} + \frac{1}{\tau}\theta^{(\tau)}\right)\right. \\
&\qquad\qquad\qquad\qquad\qquad\qquad \left.\times\left(\frac{\tau-1}{\tau}(\bar{\theta}^{(\tau-1)})^\intercal + \frac{1}{\tau}(\theta^{(\tau)})^\intercal\right)\right] \\
&= \frac{1}{\tau - 1}\left[\sum_{\nu=1}^{\tau-1} \theta^{(\nu)}(\theta^{(\nu)})^\intercal + \theta^{(\tau)}(\theta^{(\tau)})^\intercal - \tau\left(\frac{(\tau-1)^2}{\tau^2}\bar{\theta}^{(\tau-1)}(\bar{\theta}^{(\tau-1)})^\intercal\right.\right. \\
&\qquad\qquad \left.\left.+\frac{\tau-1}{\tau^2}\left(\bar{\theta}^{(\tau-1)}(\theta^{(\tau)})^\intercal + \theta^{(\tau)}(\bar{\theta}^{(\tau-1)})^\intercal\right) + \frac{1}{\tau^2}\theta^{(\tau)}(\theta^{(\tau)})^\intercal\right)\right] \\
&= \frac{1}{\tau - 1}\sum_{\nu=1}^{\tau-1} \theta^{(\nu)}(\theta^{(\nu)})^\intercal + \frac{1}{\tau-1}\theta^{(\tau)}(\theta^{(\tau)})^\intercal - \frac{\tau-1}{\tau}\bar{\theta}^{(\tau-1)}(\bar{\theta}^{(\tau-1)})^\intercal \\
&\qquad\qquad - \frac{1}{\tau}\left(\bar{\theta}^{(\tau-1)}(\theta^{(\tau)})^\intercal + \theta^{(\tau)}(\bar{\theta}^{(\tau-1)})^\intercal\right) - \frac{1}{\tau(\tau-1)}\theta^{(\tau)}(\theta^{(\tau)})^\intercal
\end{aligned}$$

$$= \frac{1}{\tau-1}\sum_{\nu=1}^{\tau-1}\theta^{(\nu)}(\theta^{(\nu)})^\intercal - \bar{\theta}^{(\tau-1)}(\bar{\theta}^{(\tau-1)})^\intercal + \frac{1}{\tau}\theta^{(\tau)}(\theta^{(\tau)})^\intercal + \frac{1}{\tau}\bar{\theta}^{(\tau-1)}(\bar{\theta}^{(\tau-1)})^\intercal$$
$$- \frac{1}{\tau}\left(\bar{\theta}^{(\tau-1)}(\theta^{(\tau)})^\intercal + \theta^{(\tau)}(\bar{\theta}^{(\tau-1)})^\intercal\right)$$
$$= \frac{1}{\tau-1}\left(\sum_{\nu=1}^{\tau-1}\theta^{(\nu)}(\theta^{(\nu)})^\intercal - (\tau-1)\bar{\theta}^{(\tau-1)}(\bar{\theta}^{(\tau-1)})^\intercal\right) + \frac{1}{\tau}\left[\bar{\theta}^{(\tau-1)}(\bar{\theta}^{(\tau-1)})^\intercal\right.$$
$$\left. - \left(\bar{\theta}^{(\tau-1)}(\theta^{(\tau)})^\intercal + \theta^{(\tau)}(\bar{\theta}^{(\tau-1)})^\intercal\right) + \theta^{(\tau)}(\theta^{(\tau)})^\intercal\right]$$
$$= \frac{\tau-2}{\tau-1}\operatorname{Cov}(\theta^{(1)},\theta^{(2)},\dots,\theta^{(\tau-1)}) + \frac{1}{\tau}\left(\bar{\theta}^{(\tau-1)} - \theta^{(\tau)}\right)\left(\bar{\theta}^{(\tau-1)} - \theta^{(\tau)}\right)^\intercal. \tag{2.19}$$

Following Haario et al. (2001), by Equation (2.19), the covariance matrix for the Gaussian proposal with the proposed sample $\theta^{(\tau)}$ as the mean is defined as follows:

$$C_\tau = \begin{cases} C_0, & \text{if } \tau \le \tau_0 \\ s_K \operatorname{Cov}(\theta^{(1)},\theta^{(2)},\dots,\theta^{(\tau-1)}) + s_K\varepsilon I_K, & \text{if } \tau > \tau_0 \end{cases},$$

where C_0 is a covariance matrix of a choice, $\tau_0 > 0$ is the initial period, $s_K = 2.4^2/K$, $\varepsilon > 0$, and I_K is a $K \times K$ identity matrix. It remains to derive the recursive formula for C_τ whenever $\tau > \tau_0$ such that

$$C_\tau = s_K \operatorname{Cov}(\theta^{(1)},\theta^{(2)},\dots,\theta^{(\tau-1)}) + s_K\varepsilon I_K$$
$$= s_K\left[\frac{\tau-3}{\tau-2}\operatorname{Cov}(\theta^{(1)},\theta^{(2)},\dots,\theta^{(\tau-2)})\right.$$
$$\left. + \frac{1}{\tau-1}(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})^\intercal\right] + s_K\varepsilon I_K$$
$$= s_K\frac{\tau-3}{\tau-2}\operatorname{Cov}(\theta^{(1)},\theta^{(2)},\dots,\theta^{(\tau-2)})$$
$$+ \frac{s_K}{\tau-1}(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})^\intercal + s_K\varepsilon I_K$$
$$= \frac{\tau-3}{\tau-2}\left(s_K\operatorname{Cov}(\theta^{(1)},\theta^{(2)},\dots,\theta^{(\tau-2)}) + \frac{\tau-2}{\tau-3}s_K\varepsilon I_K\right)$$
$$+ \frac{s_K}{\tau-1}(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})^\intercal$$
$$= \frac{\tau-3}{\tau-2}\left(\underbrace{s_K\operatorname{Cov}(\theta^{(1)},\theta^{(2)},\dots,\theta^{(\tau-2)}) + s_K\varepsilon I_K}_{C_{\tau-1}} + \frac{1}{\tau-3}s_K\varepsilon I_K\right)$$
$$+ \frac{s_K}{\tau-1}(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})^\intercal$$
$$= \frac{\tau-3}{\tau-2}C_{\tau-1} + \frac{1}{\tau-2}s_K\varepsilon I_K + \frac{s_K}{\tau-1}(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})^\intercal$$

$$
\begin{aligned}
&= \frac{\tau-3}{\tau-2}C_{\tau-1} + \frac{s_K}{\tau-2}\left[\frac{\tau-2}{\tau-1}(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})(\bar{\theta}^{(\tau-2)} - \theta^{(\tau-1)})^\intercal + \varepsilon I_K\right] \\
&= \frac{\tau-3}{\tau-2}C_{\tau-1} + \frac{s_K}{\tau-2}\Bigg[\frac{\tau-2}{\tau-1}\Big(\bar{\theta}^{(\tau-2)}(\bar{\theta}^{(\tau-2)})^\intercal - (\bar{\theta}^{(\tau-2)}(\theta^{(\tau-1)})^\intercal + \theta^{(\tau-1)}(\bar{\theta}^{(\tau-2)})^\intercal) \\
&\qquad\qquad + \theta^{(\tau-1)}(\theta^{(\tau-1)})^\intercal\Big) + \varepsilon I_K\Bigg] \\
&= \frac{\tau-3}{\tau-2}C_{\tau-1} + \frac{s_K}{\tau-2}\Bigg[\frac{1}{\tau-1}\Big((\tau-2)\bar{\theta}^{(\tau-2)}(\bar{\theta}^{(\tau-2)})^\intercal - \Big(\Big((\tau-1)\bar{\theta}^{(\tau-1)} - \theta^{(\tau-1)}\Big) \\
&\quad \times(\theta^{(\tau-1)})^\intercal + \theta^{(\tau-1)}\Big((\tau-1)(\bar{\theta}^{(\tau-1)})^\intercal - (\theta^{(\tau-1)})^\intercal\Big)\Big) + (\tau-2)\theta^{(\tau-1)}(\theta^{(\tau-1)})^\intercal\Big) + \varepsilon I_K\Bigg] \\
&= \frac{\tau-3}{\tau-2}C_{\tau-1} + \frac{s_K}{\tau-2}\Bigg[\frac{1}{\tau-1}\Big((\tau-2)\bar{\theta}^{(\tau-2)}(\bar{\theta}^{(\tau-2)})^\intercal \\
&\qquad - (\tau-1)(\bar{\theta}^{(\tau-1)}(\theta^{(\tau-1)})^\intercal + \theta^{(\tau-1)}(\bar{\theta}^{(\tau-1)})^\intercal) + \tau\theta^{(\tau-1)}(\theta^{(\tau-1)})^\intercal\Big) + \varepsilon I_K\Bigg] \\
&= \frac{\tau-3}{\tau-2}C_{\tau-1} + \frac{s_K}{\tau-2}\Bigg[\frac{1}{\tau-1}\Big((\tau-2)\bar{\theta}^{(\tau-2)}(\bar{\theta}^{(\tau-2)})^\intercal \\
&\qquad - (\tau-1)(\bar{\theta}^{(\tau-1)}(\theta^{(\tau-1)})^\intercal + (\bar{\theta}^{(\tau-1)}(\theta^{(\tau-1)})^\intercal)^\intercal) + \tau\theta^{(\tau-1)}(\theta^{(\tau-1)})^\intercal\Big) + \varepsilon I_K\Bigg].
\end{aligned}
\tag{2.20}
$$

The Gaussian proposal distribution is now able to contain the knowledge about the sequence of the random variables, $(\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(\tau-1)})$ such that $q_\tau(\cdot \mid \theta^{(1)}, \theta^{(2)}, \dots, \theta^{(\tau-1)})$. Then, suppose that $\theta' \sim q(\cdot \mid \theta^{(1)}, \theta^{(2)}, \dots, \theta^{(\tau-1)})$ was drawn. It follows that the acceptance probability of θ' given $\theta^{(\tau-1)}$ is given as follows:

$$
r = a(\theta' \mid \theta^{(\tau-1)}) = \min\left\{1, \frac{\pi(\theta')}{\pi(\theta^{(\tau-1)})}\right\}. \tag{2.21}
$$

For the current iteration $\theta^{(\tau)}$, θ' is accepted by the following steps such that

$$
\theta^{(\tau)} = \begin{cases} \theta', & \text{if } u \le r \\ \theta^{(\tau-1)}, & \text{if } u > r \end{cases},
$$

where $u \sim \mathcal{U}(0,1)$. It is important to note that Equation (2.21) is not based on the detailed balance condition (reversibility), and hence, the chain of the random variables is of a non-Markov property (Haario et al., 2001). Equation (2.20) allows one to compute the covariance C_τ that contains all the information of the random variables $\{\theta^{(\nu)} : \nu \in \{1, 2, \dots, \tau-1\}\}$.

Tierney and Mira (1999) discussed and conjectured how developing adaptive algorithms to specific important classes of problems will play an important role in the greatest progress in MCMC methods as a tool for Bayesian inference.

The following summarises the adaptive Metropolis algorithm, Algorithm 3.

Algorithm 3 The Adaptive Metropolis Algorithm

```
 1: Initialise $\theta^{(1)}$, $s_K$, $\varepsilon > 0$, $\tau_0$, and $C_0$.
 2: for $\tau = 2, 3, \ldots, N$ do
 3:     if $\tau <= \tau_0$ then
 4:         Simulate $\theta' \sim \mathcal{N}_K(\theta^{(\tau-1)}, C_0)$.
 5:     else
 6:         Simulate $\theta' \sim \mathcal{N}_K(\theta^{(\tau-1)}, C_{\tau-1})$.
 7:     end if
 8:     Calculate
            $$r = \min\left\{1, \frac{\pi(\theta')}{\pi(\theta^{(\tau-1)})}\right\}.$$
 9:     Draw $u \sim \mathcal{U}(0, 1)$.
10:     if $u \leq r$ then
11:         $\theta^{(\tau)} = \theta'$
12:     else
13:         $\theta^{(\tau)} = \theta^{(\tau-1)}$
14:     end if
15:     if $\tau > \tau_0$ then
16:         Update $C_\tau$ by Equation (2.20).
17:     end if
18: end for
```

2.4.2 Multiple-try Metropolis-Hastings Algorithm

It was well known that the standard Metropolis-Hastings algorithm might suffer from slow convergence (Mengersen et al., 1996). Metropolis-type local moves often get trapped in a local mode easily, and thus, it triggers slow convergence (Liu et al., 2000). The multiple-try Metropolis-Hastings (MTM) algorithm was proposed by Liu et al. (2000) in order to allow one for having a large searching region by simulating multiple draws from a proposal distribution $q(\cdot \mid \cdot)$ at once. It is often challenging for the Metropolis-Hastings algorithm to explore the entire target distribution of interest effectively since the sampler can be trapped in the local mode (Gamerman and Lopes, 2006; Liu et al., 2000; Roberts et al., 1997).

To implement an MTM algorithm, the weight of a proposed draw θ' given a current draw $\theta^{(\tau-1)}$, that is, $w(\theta' \mid \theta)$ is defined as follows:

$$w(\theta' \mid \theta^{(\tau-1)}) = \pi(\theta')q(\theta' \mid \theta^{(\tau-1)})\lambda(\theta' \mid \theta^{(\tau-1)}), \tag{2.22}$$

where $\lambda(\theta' \mid \theta^{(\tau-1)})$ is a non-negative symmetric function in $(\theta^{(\tau-1)}, \theta') \in \Theta^2$ whilst $\lambda(\theta' \mid \theta^{(\tau-1)}) > 0$ whenever $q(\theta' \mid \theta^{(\tau-1)}) > 0$, for any $\tau \in \{2, 3, \ldots, N\}$. The simplest form of the non-negative symmetric function is to set the function to be 1, that is, $\lambda(\theta' \mid \theta^{(\tau-1)}) = 1$. Another possible form is as follows:

$$\lambda(\theta' \mid \theta^{(\tau-1)}) = \left(\frac{q(\theta' \mid \theta^{(\tau-1)}) + q(\theta^{(\tau-1)} \mid \theta')}{2} \right)^{-1}. \tag{2.23}$$

Note that the symmetric proposal distribution in Equation (2.23) allows the weighting function in Equation (2.22) to be reduced to the posterior distribution itself (Liu et al., 2000). One draws multiple random samples, say $\{\theta_1^*, \theta_2^*, \ldots, \theta_k^*\}$, from a proposal distribution $q(\cdot \mid \theta^{(\tau-1)})$. Then, the proposed value θ' is drawn from $\{\theta_1^*, \theta_2^*, \ldots, \theta_k^*\}$ with probability which is proportional to $\{w(\theta_j^* \mid \theta^{(\tau-1)})\}$, for $j \in \{1, 2, \ldots, k\}$. Finally, one draws multiple random samples $\{\theta_1^{**}, \theta_2^{**}, \ldots, \theta_{k-1}^{**}\}$ from the proposal distribution $q(\cdot \mid \theta')$, and set $\theta_k^{**} = \theta^{(\tau-1)}$. Hence, a new acceptance ratio is defined as follows (Liu et al., 2000):

$$r = a(\theta' \mid \theta^{(\tau-1)}) = \min \left\{ 1, \frac{w(\theta_1^* \mid \theta^{(\tau-1)}) + w(\theta_2^* \mid \theta^{(\tau-1)}) + \cdots + w(\theta_k^* \mid \theta^{(\tau-1)})}{w(\theta_1^{**} \mid \theta') + w(\theta_2^{**} \mid \theta') + \cdots + w(\theta_k^{**} \mid \theta')} \right\}. \tag{2.24}$$

The basic scheme of the MTM algorithm is summarised as follows.

2.4.3 Delayed Rejection Metropolis-Hastings Algorithm

Tierney and Mira (1999) and Mira and Tierney (2002) introduced the idea of *delaying a rejection step* in the acceptance probability. It is an inherent property that a proposed draw from a proposal distribution for the target distribution is rejected by the Metropolis-Hastings ratio; however, the rejected draw may suggest that the proposal distribution is of a *bad fit* for the density of interest. Thus, an adapted proposal distribution based on the information of the rejected draw is able to help the MCMC

Algorithm 4 The Multiple-try Metropolis-Hastings Algorithm

```
1: Draw multiple random samples, $\{\theta_1^*, \theta_2^*, \ldots, \theta_k^*\}$, from $q(\cdot \mid \theta^{(\tau-1)})$.
2: Sample $\theta'$ from $\{\theta_1^*, \theta_2^*, \ldots, \theta_k^*\}$ with probability which is proportional to
   $\{w(\theta_j^* \mid \theta^{(\tau-1)})\}$, where $j \in \{1, 2, \ldots, k\}$.
3: Draw $\{\theta_1^{**}, \theta_2^{**}, \ldots, \theta_{k-1}^{**}\}$ from $q(\cdot \mid \theta')$, and set $\theta_k^{**} = \theta^{(\tau-1)}$.
4: Calculate the acceptance probability as follows:
   $$r = a(\theta' \mid \theta^{(\tau-1)}) = \min\left\{1, \frac{w(\theta_1^* \mid \theta^{(\tau-1)}) + w(\theta_2^* \mid \theta^{(\tau-1)}) + \cdots + w(\theta_k^* \mid \theta^{(\tau-1)})}{w(\theta_1^{**} \mid \theta') + w(\theta_2^{**} \mid \theta') + \cdots + w(\theta_k^{**} \mid \theta')}\right\}.$$
5: Draw $u \sim \mathcal{U}(0, 1)$.
6: if $u \leq r$ then
7:     $\theta^{(\tau)} = \theta'$
8: else
9:     $\theta^{(\tau)} = \theta^{(\tau-1)}$
10: end if
```

algorithm be more efficient (Tierney and Mira, 1999).

Suppose a current draw $\theta^{(\tau-1)}$, for any $\tau \in \{2, 3, \ldots, N\}$, and a proposed draw $\theta_1^* \sim q_1(\cdot \mid \theta^{(\tau-1)})$ is rejected. Then, a proposed draw $\theta_2^* \sim q_2(\cdot \mid \theta_1^*, \theta^{(\tau-1)})$ is generated. It follows that θ_2^* is accepted according to the following acceptance probability such that

$$a(\theta_2^* \mid \theta_1^*, \theta^{(\tau-1)}) = \min\left\{1, \frac{\pi(\theta_2^*) q_1(\theta_1^* \mid \theta_2^*) q_2(\theta^{(\tau-1)} \mid \theta_1^*, \theta_2^*)[1 - a(\theta_1^* \mid \theta_2^*)]}{\pi(\theta^{(\tau-1)}) q_1(\theta_1^* \mid \theta^{(\tau-1)}) q_2(\theta_2^* \mid \theta_1^*, \theta^{(\tau-1)})[1 - a(\theta_1^* \mid \theta^{(\tau-1)})]}\right\}, \tag{2.25}$$

where $q_1(\cdot \mid \theta^{(\tau-1)})$ and $q_2(\cdot \mid \theta_1^*, \theta^{(\tau-1)})$ are arbitrary proposal distributions.

Equation (2.25) preserves the detailed balance condition and proves the target distribution $\pi(\cdot)$ invariant (Tierney and Mira, 1999). Mira et al. (2001) made a remark on generalising Equation (2.25) to ν^{th} try such that

$$\begin{aligned} & a(\theta_\nu^* \mid \theta_{\nu-1}^*, \ldots, \theta_1^*, \theta^{(\tau-1)}) \\ & = \min\left\{1, \frac{\pi(\theta_\nu^*) q_1(\theta_{\nu-1}^* \mid \theta_\nu^*) q_2(\theta_{\nu-2}^* \mid \theta_{\nu-1}^*, \theta_\nu^*) \cdots q_\nu(\theta^{(\tau-1)} \mid \theta_1^*, \theta_2^*, \ldots, \theta_{\nu-1}^*, \theta_\nu^*)}{\pi(\theta^{(\tau-1)}) q_1(\theta_1^* \mid \theta^{(\tau-1)}) q_2(\theta_2^* \mid \theta_1^*, \theta^{(\tau-1)}) \cdots q_\nu(\theta_\nu^* \mid \theta_{\nu-1}^*, \theta_{\nu-2}^*, \ldots, \theta_1^*, \theta^{(\tau-1)})}\right. \\ & \quad \left. \times \frac{[1 - a(\theta_{\nu-1}^* \mid \theta_\nu^*)][1 - a(\theta_{\nu-2}^* \mid \theta_{\nu-1}^*, \theta_\nu^*)] \cdots [1 - a(\theta_1^* \mid \theta_2^*, \theta_3^*, \ldots, \theta_\nu^*)]}{[1 - a(\theta_1^* \mid \theta^{(\tau-1)})][1 - a(\theta_2^* \mid \theta_1^*, \theta^{(\tau-1)})] \cdots [1 - a(\theta_{\nu-1}^* \mid \theta_{\nu-2}^*, \theta_{\nu-3}^*, \ldots, \theta_1^*, \theta^{(\tau-1)})]}\right\}. \end{aligned} \tag{2.26}$$

By Equation (2.26), the crucial difference between the delayed rejection Metropolis-Hastings (DRMH) and MTM algorithms is that not only does some proposal at ν^{th} try of the DRMH algorithm $q_\nu(\theta^*_\nu \,|\, \theta^*_{\nu-1}, \theta^*_{\nu-2}, \ldots, \theta^*_1, \theta^{(\tau-1)})$ depend on the current draw $\theta^{(\tau-1)}$, but it also relies on the sequence of rejected tries $(\theta^*_1, \theta^*_2, \ldots, \theta^*_{\nu-1})$. A simple example of implementing different proposals is to define the initial proposal as an independent Metropolis-Hastings proposal such that $q_1(\theta^*_1 \,|\, \theta^{(\tau-1)}) = \mathcal{N}(0, \sigma^2)$, where $\sigma > 0$ is arbitrary, and to define the second proposal $q_2(\theta^*_2 \,|\, \theta^*_1, \theta^{(\tau-1)})$ as the random walk Metropolis-Hastings proposal such that $\theta^*_2 \sim \mathcal{N}(\theta^*_1, \sigma^2)$ (Tierney and Mira, 1999).

Furthermore, delayed rejection strategy can be considered as a means of incorporating two or more different proposals for Metropolis-Hastings moves. For example, either the first proposal which is computationally inexpensive to simulate can be tried in order to enhance the computational time. Alternatively, the first kernel can take a form of global search, namely with a large variance, to explore the state space more efficiently, and then, the second kernel can be more timid approach to allow the sampler to implement a local search (Haario et al., 2006).

2.4.4 Symmetric Delayed Rejection Metropolis-Hastings Algorithm

Suppose for any ν^{th} stage, the proposal distribution is symmetric and is only dependent on the last rejected draw $\theta^*_{\nu-1}$ such that

$$q_\nu(\theta^*_\nu \,|\, \theta^*_{\nu-1}, \theta^*_{\nu-2}, \ldots, \theta^*_1, \theta^{(\tau-1)}) = q(\theta^*_\nu \,|\, \theta^*_{\nu-1}) = q(\theta^*_{\nu-1} \,|\, \theta^*_\nu). \tag{2.27}$$

By Equation (2.27), the acceptance ratio in Equation (2.26) is reduced to the following:

$$\begin{aligned}
&a(\theta^*_\nu \,|\, \theta^*_{\nu-1}, \ldots, \theta^*_1, \theta^{(\tau-1)}) \\
&= \frac{\pi(\theta^*_\nu) q_1(\theta^*_{\nu-1} \,|\, \theta^*_\nu) q_2(\theta^*_{\nu-2} \,|\, \theta^*_{\nu-1}, \theta^*_\nu) \cdots q_\nu(\theta^{(\tau-1)} \,|\, \theta^*_1, \theta^*_2, \ldots, \theta^*_{\nu-1}, \theta^*_\nu)}{\pi(\theta^{(\tau-1)}) q_1(\theta^*_1 \,|\, \theta^{(\tau-1)}) q_2(\theta^*_2 \,|\, \theta^*_1, \theta^{(\tau-1)}) \cdots q_\nu(\theta^*_\nu \,|\, \theta^*_{\nu-1}, \theta^*_{\nu-2}, \ldots, \theta^*_1, \theta^{(\tau-1)})} \\
&\qquad \times \frac{[1 - a(\theta^*_{\nu-1} \,|\, \theta^*_\nu)][1 - a(\theta^*_{\nu-2} \,|\, \theta^*_{\nu-1}, \theta^*_\nu)] \cdots [1 - a(\theta^*_1 \,|\, \theta^*_2, \theta^*_3, \ldots, \theta^*_\nu)]}{[1 - a(\theta^*_1 \,|\, \theta^{(\tau-1)})][1 - a(\theta^*_2 \,|\, \theta^*_1, \theta^{(\tau-1)})] \cdots [1 - a(\theta^*_{\nu-1} \,|\, \theta^*_{\nu-2}, \theta^*_{\nu-3}, \ldots, \theta^*_1, \theta^{(\tau-1)})]} \\
&= \frac{\pi(\theta^*_\nu) \cancel{q(\theta^*_{\nu-1} \,|\, \theta^*_\nu)} \cancel{q(\theta^*_{\nu-2} \,|\, \theta^*_{\nu-1})} \cdots \cancel{q(\theta^*_1 \,|\, \theta^*_2)} \cancel{q(\theta^{(\tau-1)} \,|\, \theta^*_1)}}{\pi(\theta^{(\tau-1)}) \cancel{q(\theta^*_1 \,|\, \theta^{(\tau-1)})} \cancel{q(\theta^*_2 \,|\, \theta^*_1)} \cdots \cancel{q(\theta^*_{\nu-1} \,|\, \theta^*_{\nu-2})} \cancel{q(\theta^*_\nu \,|\, \theta^*_{\nu-1})}} \\
&\qquad\qquad \times \frac{[1 - a(\theta^*_{\nu-1} \,|\, \theta^*_\nu)][1 - a(\theta^*_{\nu-2} \,|\, \theta^*_{\nu-1})] \cdots [1 - a(\theta^*_1 \,|\, \theta^*_2)]}{[1 - a(\theta^*_1 \,|\, \theta^{(\tau-1)})][1 - a(\theta^*_2 \,|\, \theta^*_1)] \cdots [1 - a(\theta^*_{\nu-1} \,|\, \theta^*_{\nu-2})]}
\end{aligned}$$

$$= \frac{\pi(\theta_{\nu}^{*})[1 - a(\theta_{\nu-1}^{*} \,|\, \theta_{\nu}^{*})][1 - a(\theta_{\nu-2}^{*} \,|\, \theta_{\nu-1}^{*})] \cdots [1 - a(\theta_{1}^{*} \,|\, \theta_{2}^{*})]}{\pi(\theta^{(\tau-1)})[1 - a(\theta_{1}^{*} \,|\, \theta^{(\tau-1)})][1 - a(\theta_{2}^{*} \,|\, \theta_{1}^{*})] \cdots [1 - a(\theta_{\nu-1}^{*} \,|\, \theta_{\nu-2}^{*})]}. \tag{2.28}$$

Equation (2.28) relies on the assumption where the current proposal distribution only depends on the last rejected draw in the MCMC samples. This has a drawback and loses its inferential capability against Equation (2.26) in which the proposal utilises all the rejected draws (Mira et al., 2001).

Algorithm 5 summarises both the DRMH and the symmetric delayed rejection Metropolis algorithms.

Algorithm 5 Delayed Rejection Metropolis-Hastings Algorithm

```
1: Initialise $\theta^{(1)}$ and the number of rejection stages, $\nu$.
2: for $\tau = 2, 3, \ldots, N$ do
3:     Set $\nu' = 0$.
4:     repeat
5:         $\nu' \leftarrow \nu' + 1$
6:         Simulate $\theta_{\nu'}^{*} \sim q(\cdot \,|\, \theta_{\nu'-1}^{*}, \theta_{\nu'-2}^{*}, \ldots, \theta_{1}^{*}, \theta^{(\tau-1)})$.
7:         Calculate $r = a(\theta_{\nu'}^{*} \,|\, \theta_{\nu'-1}^{*}, \theta_{\nu'-2}^{*}, \ldots, \theta_{1}^{*}\theta^{(\tau-1)})$ by either Equations (2.26) or
   (2.28), depending on symmetricity of the proposal distribution, $q(\cdot \,|\, \cdot)$.
8:         Draw $u \sim \mathcal{U}(0, 1)$.
9:     until Set $\theta^{(\tau)} = \theta_{\nu'}^{*}$ whenever $u \leq r$. Otherwise, set $\theta^{(\tau)} = \theta^{(\tau-1)}$ when $\nu' = \nu$.
10: end for
```

2.5 Label Switching

In Bayesian inference, it is quite common to encounter an issue where estimating parameters by posterior means or clustering using mixture models may not sometimes lead to desired MCMC convergence. This problem is referred to as the *label switching* problem caused by symmetry in the likelihood of the model parameters and its invariance to permutations of the labels (Scott, 2011; Sperrin et al., 2010; Stephens, 2000). A prior which is same for all the permutations of parameters has no information about identifying the components. Such a prior also symmetrises the posterior distribution, and thus, inference by their posterior means it provides unclear information.

The mixing of labels implies that MCMC convergence will be impossible to achieve since estimation of all the parameters will be as close as each other (Celeux et al., 2000). To circumvent this type of problem, there have been several strategies which can be divided into three categories (Sperrin et al., 2010) such as

- identifiability constraints,
- deterministic relabelling algorithms, and
- probabilistic relabelling algorithms.

Identifiability constraints relabels the resulting MCMC samples for which the relabeling satisfies a constraint on the state specific parameters. It sets a restriction on the parameter space $\Theta_{\mathfrak{X}}$, so that there exists a unique permutation of parameters $\chi^* \in \mathfrak{X}$, which satisfies $(\theta^1_{\chi^*}, \theta^2_{\chi^*}, \dots, \theta^p_{\chi^*}) \in \Theta$. A simple example of constraints in an HMM is component means $\mu_1 < \mu_2 < \cdots < \mu_m$ forcing a unique label on the MCMC output at each iteration (Rydén et al., 2008; Stephens, 2000).

2.6 MCMC Convergence

It is always required to analyse whether the MCMC samples converge to their true values. Otherwise, the sampled sequences will be unfit for the analysis. There are several methods in order to assess MCMC convergence. A comprehensive review on MCMC convergence diagnostics is available in Cowles and Carlin (1996). Of the MCMC diagnostic tests in the literature, Geweke's diagnostics was selected for this thesis. The diagnostics are available in the statistical package 'coda' in R©. The following sections are dedicated to describing Geweke's diagnostics.

Geweke et al. (1991) proposed a procedure based on the spectral analysis where a Gibbs sequence being treated as time series. The convergence analysis proceeds to estimate the mean of some function of parameters $f(\theta)$ in which the Gibbs sampler is supposed to simulate the parameters θ. The nature of MCMC iterations and the function $f(\cdot)$ are assumed to imply that there exists a spectral density $S_f(\omega)$, for the resulting sequence as a time series. This time series is considered to have no discontinuity at zero frequency, that is, $\omega = 0$ (Cowles and Carlin, 1996). It follows that the estimator $\mathrm{E}[f(\theta)]$ with N iterations of the Gibbs sampler is given as follows:

$$\bar{f}_N(\theta) = \frac{\sum_{\nu=1}^{N} f(\theta^{(\nu)})}{N}.$$

Also, the asymptotic variance is given by $S_f(0)/N$. Thus, the square root of this asymptotic variance gives an estimate of the standard error of the mean. This standard error was defined as a *numeric standard error* by Geweke et al. (1991).

Geweke's diagnostics assesses MCMC convergence by calculating the mean difference of $f_{N_A}(\theta)$ with the first N_A iterations and $f_{N_B}(\theta)$ with the last N_B iterations, and dividing by the numeric standard error of that mean difference. Then, a statistic of Geweke's diagnostics is given as follows:

$$\frac{\bar{f}_{N_B}(\theta) - \bar{f}_{N_A}(\theta)}{\sqrt{S_f(0)/N}}, \tag{2.29}$$

where $\bar{f}_{N_A} = \sum_{\nu=1}^{N_A} f(\theta^{(\nu)}) \Big/ N_A$, and $\bar{f}_{N_B} = \sum_{\nu=N-N_B+1}^{N} f(\theta^{(\nu)}) \Big/ N_B$.

By the central limit theorem, the distribution of Equation (2.29) should approach a standard normal distribution as N tends to infinity under the constraint $(N_A + N_B)/N < 1$. In practice, N_A and N_B are set to be $N_A = N/10$ and $N_B = N/2$ so that the assumption of the central limit theorem can be met (Geweke et al., 1991). If the value in Equation (2.29) lies between the 95% confidence interval about 0 in a standard normal distribution, then the convergence diagnostic meets the criterion where the resulting sequence has converged to the true value (Cowles and Carlin, 1996; Geweke et al., 1991). Therefore, if a statistic of Geweke's diagnostics lies between approximately -2 and 2, then it may be claimed that the convergence is achieved.

This method was selected for several reasons. First of all, it is comparatively easier to implement than other methods. Second of all, it may be applied with any MCMC method (Cowles and Carlin, 1996). Although Geweke's method is essentially for univariate MCMC samples, it might also be used for assess convergence of the joint posterior distributions whenever $f(\theta)$ is regarded as -2 times the log of the posterior distribution (Cowles and Carlin, 1996).

Despite the fact that Geweke's diagnostics is sensitive to the specification of the spectral window and does not have a procedure for applying the method, I will adhere to the default window spectral window to assess convergence of the joint posterior distributions.

2.7 MCMC Algorithm Efficiency

A requirement of assessing the efficiencies of MCMC algorithms is considered to be crucial since different algorithms generate different results. In the following, the measure of an algorithm's efficiency, *effective sample size*, is reviewed.

Effective sample size (ESS) represents *the corresponding number of uncorrelated samples* in MCMC iterations (Gamerman and Lopes, 2006). An estimate of the ESS for MCMC samples $\theta = (\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(N)})$ is given as follows:

$$\mathrm{ESS}(\theta) = \frac{N}{1 + 2\sum_{k=1}^{\infty} L_k},$$

where N is the number of MCMC samples, and L_k is the autocorrelation at the k^{th} lag. Note that the ESS can take a value that is greater than the original number of samples whenever $\sum_{k=1}^{\infty} L_k < 0$. The ESS can also be thought of as the size of a random sample with the same posterior variance (Liu and Chen, 1995).

It is also one of the computations where one can assess whether chains of a sampler mix well, and hence, the algorithm is robust in a sense of convergence (Del Moral et al., 2012). The ESS takes values between 1 and N. Its interpretation is understood in a way of how inference based on the N weighted samples is approximately equivalent to inference based on $\mathrm{ESS}(W_t^{(\tau)})$ perfect samples from the posterior distribution $\pi_t(\cdot)$, where $W_t^{(\tau)}$ denotes the weighted samples of population t at τ^{th} MCMC iteration. It also gives an indication of how the estimator is accurate and assesses whether re-sampling schemes are required whenever its value fails to be above the threshold (Del Moral et al., 2012).

The equation for an ESS is given as follows:

$$\mathrm{ESS}(W_t^{(\tau)}) = \frac{1}{\sum_{\tau=1}^{N} \left(W_t^{(\tau)}\right)^2}. \tag{2.30}$$

At the end of N^{th} iteration, Equation (2.30) can easily be computed for each population t since the calculation of $\{W_t^{(\tau)}\}$ for all $\tau \in \{1, 2, \ldots, N\}$ is complete.

Chapter 3

Univariate NHGHMM

Although the ordinary homogeneous GHMM has gained popularity in various disciplines, it restricts transition probabilities to be constant and fails to incorporate other important underlying variables such as economic fundamentals (Diebold et al., 1994; Filardo and Gordon, 1998). Therefore, the first univariate *non-homogeneous* GHMM was introduced in the econometric literature to overcome the restrictive nature of the constant transition probabilities (Diebold et al., 1994). Unlike a homogeneous GHMM, an NHGHMM consists of time-varying transition probabilities between the hidden states to accommodate different transition probabilities of segments in the GHMM.

Filardo and Gordon (1998) investigated the NHGHMM to model a business cycle. In addition, they used a Bayesian approach which has differed from Diebold et al. (1994)'s frequentist approach to tackle technical problems caused by classical techniques. The development of conventional homogeneous Markov switching models was motivated by the problem where the conditional duration of a phase is constant due to the constant transition probabilities (Filardo and Gordon, 1998). An NHGHMM incorporated with a Bayesian approach was applied to the US financial data. Their model was able to overcome the technical problems, and successfully evaluated its ability to explain observed business cycle durations (Filardo and Gordon, 1998).

Spezia (2006) conducted Bayesian parameter estimation of an NHGHMM whose transition probabilities depend on exogenous variables through a logistic function. Spezia (2006) implemented the MCMC algorithms through the case study of ozone data. Bayesian inference proved to be useful and performed well by introducing time-varying transition probabilities; however, Spezia (2006) used the standard Metropolis-Hastings algorithm for parameter estimation of the transition probabilities.

One of the aims in this chapter is to extend Spezia (2006)'s work by introducing more advanced MCMC algorithms in order to accelerate the MCMC convergence.

Then, I attempt to observe how effective the proposed MCMC algorithms are in the model and make an appropriate comparison of those algorithms in terms of simulation efficiency and MCMC convergence. To validate these algorithms, I generate 100 independent simulated data sets of Diebold et al. (1994) where the true values of parameters of interest were set to be identical. Finally, I compare my simulation study result to that of Diebold et al. (1994) as I use Bayesian inference, not frequentist.

As we are aware, the first comprehensive *Bayesian* inference of NHGHMM was conducted by Spezia (2006). He also developed several variants of HMMs such as the homogeneous GHMM with the multivariate periodic component as a random effect (Spezia et al., 2011), and the NHHMM whose observations being dichotomous variables (Spezia et al., 2014). Another interesting type of the HMM, a spatial hidden Markov model, was developed by Spezia et al. (2018), introducing an autologistic model in the framework of the HMM. Nevertheless, the transition probabilities in these types of the HMMs are not time-varying. Hence, the model is outside of the scope and I specifically follow Spezia (2006) to focus on the aims of this thesis.

In this chapter, my original contribution is to implement those proposed MCMC algorithms in NHGHMMs, namely the adaptive Metropolis, the symmetric delayed rejection adaptive Metropolis, and the multiple-try adaptive Metropolis algorithms. The approaches are essentially different from that of Spezia (2006).

This chapter is organised as follows. In Section 3.1, an NHGHMM is mathematically defined in detail so that the time-varying probabilities can properly be introduced. In Section 3.2, prior distributions of parameters of interest for the NHGHMM are specified. In Section 3.3, a joint likelihood function is defined, followed by formal derivations of the joint and conditional posterior distributions in Section 3.4. In Section 3.5, explanation of the MCMC algorithms as part of my original contributions is presented. Lastly, simulation and case studies are presented in Sections 3.6 and 3.7, respectively.

3.1 The Model

Recall the NHGHMM discussed in Section 2.1.2. First of all, it is required to specify each parameter, prior distributions on the parameters of interest, a likelihood function, and a joint posterior distribution. This is then followed by the proposed MCMC algorithms.

Following Spezia (2006), let $S = (S_1, S_2, \dots, S_n)$ be a discrete-time, first-order, non-homogeneous Markov chain and S_t belong to a state space $\Omega = \{1, 2, \dots, m\}$, for any $t \in \{1, 2, \dots, n\}$, where $n, m \in \mathbb{N} \setminus \{1\}$. Furthermore, let $Y = (Y_1, Y_2, \dots, Y_n)$ be a sequence of conditionally independent real-valued random variables $Y_t \in \mathbb{R}$, for any $t \in \{1, 2, \dots, n\}$, on the contemporary hidden state S_t. That is,

$$Y_t \,|\, S_t = i \sim \mathcal{N}(\mu_i, \lambda_i^{-1}),$$

where $i \in \Omega$. Now, let $\rho = (\rho_1, \rho_2, \dots, \rho_m)$ be a vector of initial distributions such that $\rho_i = \Pr(S_1 = i)$, for each $i \in \Omega$. Then, define $z_t = (1, z_t^1, z_t^2, \dots, z_t^{K-1})$ as a vector of K deterministic exogenous variables, where $K \in \mathbb{N} \setminus \{1\}$ for any $t \in \{2, 3, \dots, n\}$. Furthermore, let $\alpha_{i,j}$ be a vector of K coefficients, that is, $\alpha_{i,j} = (\alpha_{i,j}^1, \alpha_{i,j}^2, \dots, \alpha_{i,j}^K)$ whenever $i \neq j$, and be a vector of K zeros when $i = j$.

Moreover, the transition matrices Q^t, for all $t \in \{2, 3, \dots, n\}$, are computed as logistic functions of $\alpha_{i,j}$ and z_t (Spezia, 2006) such that

$$\Pr(S_t = j \,|\, S_{t-1} = i) := q_{i,j}^t = \frac{\exp(z_t^\intercal \alpha_{i,j})}{1 + \sum\limits_{\substack{j \in \Omega \\ j \neq i}} \exp(z_t^\intercal \alpha_{i,j})}. \tag{3.1}$$

When at least one of $\{z_t^k\}$, for $k \in \{1, 2, \dots, K-1\}$, is non-zero, it implies that the Markov chain is *non-homogeneous.*

In addition, the model deals with time-varying stochastic matrices Q^t, for all $t \in \{2, 3, \dots, n\}$, since it assumes non-homogeneity on the Markov chain. Hence, parameter estimation of the transition probabilities differs from that of homogeneous GHMM.

Consider that the distribution of observed values in a univariate NHGHMM is Gaussian, the pairs of state specific parameters μ_i and λ_i, for any $i \in \Omega$, and the vector of K coefficients for transition probabilities, $\alpha_{i,j}$, for any $\{(i,j) \in \Omega^2 : i \neq j\}$, in addition to the hidden state sequence $s = (s_1, s_2, \ldots, s_n)$ are comprised of all parameters of interest to be estimated in the model.

It is required to specify prior distributions for each parameter of interest. The prior specification is carried out in the following section.

3.2 Prior Distributions

Following Spezia (2006), define $\theta = (\mu, \lambda, \alpha, s)$ as a vector of the parameters of interest, where $\mu = (\mu_1, \mu_2, \ldots, \mu_m)$, $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ such that $\lambda^{-1} = \sigma^2$, $\alpha = (\alpha_{i,j})$, for any $(i,j) \in \Omega^2$, and $s = (s_1, s_2, \ldots, s_n)$.

The prior specification on the three vectors of the parameters, namely μ, λ, and α is as follows (Spezia, 2006):

(i) *Prior for μ*: A truncated normal prior $\mathcal{N}(\mu_{i-1}, \lambda_{M_i}^{-1})$ over the interval (μ_{i-1}, ∞) is placed on μ_i, for any $i \in \{2, 3, \ldots, m\}$. Define $\lambda_{M_i} = 2M^2/(\lambda_{i-1} + \lambda_i)$ with some positive constant M. Then, μ_1 is drawn from the normal prior $\mathcal{N}(\mu_{M_1}, \lambda_{M_1}^{-1})$. The truncated interval (μ_{i-1}, ∞) is proposed to overcome unidentifiability of labels by placing μ_i in an increasing order such that $\mu_i < \mu_j$, for any $i, j \in \Omega$, so that $i < j$ (Spezia, 2006). Due to the truncated interval, the prior distribution of μ_i, for any $i \in \{2, 3, \ldots, m\}$ is conditioned on μ_{i-1}. Thus,

$$p(\mu) = p(\mu_1) \prod_{i=2}^{m} p(\mu_i \,|\, \mu_{i-1}), \tag{3.2}$$

where $\mu_1 \sim \mathcal{N}(\mu_{M_1}, \lambda_{M_1}^{-1})$ and $\mu_i \,|\, \mu_{i-1} \sim \mathcal{N}(\mu_{i-1}, \lambda_{M_i}^{-1})$, for any $i \in \{2, 3, \ldots, m\}$. Therefore,

$$\mu_i \sim \mathcal{N}(\mu_{M_1}, \lambda_{M_1}^{-1}) = \sqrt{\frac{\lambda_{M_1}}{2\pi}} \exp\left(-\frac{1}{2}\frac{(\mu_i - \mu_{M_1})^2}{\lambda_{M_1}^{-1}}\right), \text{ if } i = 1$$

$$\mu_i \,|\, \mu_{i-1} \sim \mathcal{N}(\mu_{i-1}, \lambda_{M_i}^{-1}) = 2\sqrt{\frac{\lambda_{M_i}}{2\pi}} \exp\left(-\frac{1}{2}\frac{(\mu_i - \mu_{i-1})^2}{\lambda_{M_i}^{-1}}\right), \text{ if } i > 1,$$

where $\operatorname{supp}\{p(\mu_i \mid \mu_{i-1})\} = (\mu_{i-1}, \infty)$, for any $i \in \{2, 3, \ldots, m\}$.

(ii) *Prior for* λ: A gamma prior $\mathcal{G}(\alpha_\Lambda, \beta_\Lambda)$ is placed on λ_i, for any $i \in \Omega$. Each parameter is independent, and therefore,

$$p(\lambda) = \prod_{i=1}^{m} p(\lambda_i), \tag{3.3}$$

where $\lambda_i \sim \mathcal{G}(\alpha_A, \beta_A)$, for any $i \in \Omega$. Henceforth,

$$p(\lambda_i) = \frac{\beta_\Lambda^{\alpha_\Lambda}}{\Gamma(\alpha_\Lambda)} \lambda_i^{\alpha_\Lambda - 1} \exp(-\beta_\Lambda \lambda_i),$$

for any $i \in \{1, 2, \ldots, m\}$.

(iii) *Prior for* α: A multivariate normal prior of dimension K such that $\mathcal{N}_K(\mu_A, \Lambda_A^{-1})$ is placed on $\alpha_{i,j}$, for $\{(i,j) \in \Omega^2 : i \neq j\}$, and the parameters are independent of each other as follows:

$$p(\alpha) = \prod_{\substack{(i,j)\in\Omega^2 \\ i \neq j}} p(\alpha_{i,j}), \tag{3.4}$$

where $\alpha_{i,j} \sim \mathcal{N}_K(\mu_A, \Lambda_A^{-1})$, for $\{(i,j) \in \Omega^2 : i \neq j\}$, $\mu_A \in \mathbb{R}^K$ is a vector of means and $\Lambda_A \in \mathbb{R}^{K \times K}$ is a positive-definite precision matrix. Thus,

$$\begin{aligned} p(\alpha_{i,j}) &= [(2\pi)^K \det(\Lambda_A^{-1})]^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\alpha_{i,j} - \mu_A)^\intercal \Lambda_A (\alpha_{i,j} - \mu_A)\right) \\ &= \det(2\pi\Lambda_A^{-1})^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\alpha_{i,j} - \mu_A)^\intercal \Lambda_A (\alpha_{i,j} - \mu_A)\right), \end{aligned}$$

for any $\{(i,j) \in \Omega^2 : i \neq j\}$.

Note that the prior distributions for the parameters μ and λ are designed to be conjugate priors. As for the parameter α, the multivariate Gaussian distribution is chosen for the prior distribution (Spezia, 2006). In an application of the NHGHMM to ozone data, Spezia (2006) set the prior distribution of the parameter α to be non-informative. Following him, the non-informative prior distribution for the parameter α is used to avoid perturbing the posterior distribution of interest.

Now, it is required to derive the joint likelihood function of all the parameters in the model.

3.3 Likelihood Function

Within the framework of a univariate GHMM (Spezia, 2006), the joint likelihood function of the hidden states and the observed data (s, y) given other variables is given as follows:

$$\begin{aligned} f(s, y \mid \mu, \lambda, \alpha, z, \rho) &= f(s \mid \mu, \lambda, \alpha, z, \rho) f(y \mid \mu, \lambda, s, \alpha, z, \rho) \\ &= f(s \mid \alpha, z, \rho) f(y \mid \mu, \lambda, s) \qquad (3.5) \\ &= f(s_1 \mid \rho) \prod_{t=2}^{n} f(s_t \mid s_{t-1}, \alpha, z_t) \prod_{t=1}^{n} f(y_t \mid \mu_{s_t}, \lambda_{s_t}, s_t) \qquad (3.6) \\ &= \rho_{s_1} \prod_{t=2}^{n} q^t_{s_{t-1},s_t} \prod_{t=1}^{n} \frac{1}{\sqrt{2\pi\lambda^{-1}_{s_t}}} \exp\left(-\frac{1}{2} \frac{(y_t - \mu_{s_t})^2}{\lambda^{-1}_{s_t}} \right), \qquad (3.7) \end{aligned}$$

where $z = (z_1, z_2, \ldots, z_n)$. The joint density of the hidden states is *not* conditioned on (μ, λ), but is conditioned on (α, z, ρ) since $q^t_{i,j}$ is calculated according to Equation (3.1). Moreover, the joint density of the observed data is *not* conditioned on those variables (α, z, ρ). Thus, Equation (3.5) is true. From Equation (3.5) to Equation (3.6), the density of s_1 given the initial density ρ is simply the initial distribution. That is,

$$f(s_1 \mid \rho) = \Pr(S_1 = s_1) = \rho_{s_1}.$$

Note that the non-homogeneity of the model is reflected on the term $q^t_{s_{t-1},s_t}$, for any $t \in \{2, 3, \ldots, n\}$.

3.4 Posterior Distributions

Bayesian inference requires deriving a joint posterior distribution of the parameters given the observed data. Given the prior distributions in Equations (3.2), (3.3), and (3.4), and the likelihood function in Equation (3.7), the joint posterior distribution of the parameters $\theta = (\mu, \lambda, \alpha, s)$ given the observed data y, and other deterministic variables is expressed as follows:

$$
\begin{aligned}
\pi(\theta \mid y, z, \rho) &= \pi(\mu, \lambda, \alpha, s \mid y, z, \rho) \\
&\propto f(s, y \mid \mu, \lambda, \alpha, z, \rho) p(\mu, \lambda, \alpha, z, \rho) \\
&= f(s \mid \alpha, z, \rho) f(y \mid \mu, \lambda, s) p(\mu, \lambda, \alpha, z, \rho) \\
&= f(s \mid \alpha, z, \rho) f(y \mid \mu, \lambda, s) p(\mu, \lambda, \alpha \mid z, \rho) p(z, \rho) \\
&\propto f(s_1 \mid \rho) \prod_{t=2}^{n} f(s_t \mid s_{t-1}, \alpha, z_t) \prod_{t=1}^{n} f(y_t \mid \mu_{s_t}, \lambda_{s_t}, s_t) p(\mu, \lambda, \alpha) \\
&= \rho_{s_1} \prod_{t=2}^{n} q^t_{s_{t-1}, s_t} \prod_{t=1}^{n} \sqrt{\frac{\lambda_{s_t}}{2\pi}} \exp\left(-\frac{1}{2} \frac{(y_t - \mu_{s_t})^2}{\lambda^{-1}_{s_t}} \right) p(\mu) p(\lambda) p(\alpha) \\
&= \rho_{s_1} \prod_{t=2}^{n} q^t_{s_{t-1}, s_t} \prod_{t=1}^{n} \sqrt{\frac{\lambda_{s_t}}{2\pi}} \exp\left(-\frac{1}{2} \frac{(y_t - \mu_{s_t})^2}{\lambda^{-1}_{s_t}} \right) \\
&\qquad \times \left[p(\mu_1) \prod_{i=2}^{m} p(\mu_i \mid \mu_{i-1}) \right] \left[\prod_{i=1}^{m} p(\lambda_i) \right] \left[\prod_{\substack{(i,j) \in \Omega^2 \\ i \neq j}} p(\alpha_{i,j}) \right] \\
&= \rho_{s_1} \prod_{t=2}^{n} q^t_{s_{t-1}, s_t} \prod_{t=1}^{n} \sqrt{\lambda_{s_t}} \exp\left(-\frac{1}{2} \frac{(y_t - \mu_{s_t})^2}{\lambda^{-1}_{s_t}} \right) \\
&\quad \times \left[\sqrt{\frac{\lambda_{M_1}}{2\pi}} \exp\left(-\frac{1}{2} \frac{(\mu_1 - \mu_{M_1})^2}{\lambda^{-1}_{M_1}} \right) \prod_{i=2}^{m} \sqrt{\frac{\lambda_{M_i}}{2\pi}} \exp\left(-\frac{1}{2} \frac{(\mu_i - \mu_{i-1})^2}{\lambda^{-1}_{M_i}} \right) \right] \\
&\quad \times \left[\prod_{i=1}^{m} \frac{\beta_\Lambda^{\alpha_\Lambda}}{\Gamma(\alpha_\Lambda)} \lambda_i^{\alpha_\Lambda - 1} \exp(-\beta_\Lambda \lambda_i) \right] \\
&\quad \times \left[\prod_{\substack{(i,j) \in \Omega^2 \\ i \neq j}} \det(2\pi \Lambda_A^{-1})^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (\alpha_{i,j} - \mu_A)^\intercal \Lambda_A (\alpha_{i,j} - \mu_A) \right) \right] \\
&\propto \rho_{s_1} \prod_{t=2}^{n} q^t_{s_{t-1}, s_t} \prod_{t=1}^{n} \sqrt{\lambda_{s_t}} \exp\left(-\frac{1}{2} \frac{(y_t - \mu_{s_t})^2}{\lambda^{-1}_{s_t}} \right) \\
&\quad \times \left[\exp\left(-\frac{1}{2} \left(\frac{(\mu_1 - \mu_{M_1})^2}{\lambda^{-1}_{M_1}} + \sum_{i=2}^{m} \frac{(\mu_i - \mu_{i-1})^2}{\lambda^{-1}_{M_i}} \right) \right) \right] \\
&\quad \times \left[\left(\prod_{i=1}^{m} \lambda_i \right)^{\alpha_\Lambda - 1} \exp(-\beta_\Lambda \sum_{i=1}^{m} \lambda_i) \right] \\
&\quad \times \left[\prod_{\substack{(i,j) \in \Omega^2 \\ i \neq j}} \exp\left(-\frac{1}{2} (\alpha_{i,j} - \mu_A)^\intercal \Lambda_A (\alpha_{i,j} - \mu_A) \right) \right]. \qquad (3.8)
\end{aligned}
$$

Given the joint posterior distribution of the parameters of interest θ in Equation (3.8), Bayesian inference of the parameters $(\mu^{(\tau)}, \lambda^{(\tau)}, \alpha^{(\tau)})$ requires conditional posterior distributions of μ_i, λ_i for each $i \in \Omega$, and $\alpha_{i,j}$ for all $(i,j) \in \Omega^2$. Alternatively, to estimate a hidden state sequence at the τ^{th} MCMC iteration, $s^{(\tau)} = (s_1^{(\tau)}, s_2^{(\tau)}, \ldots, s_n^{(\tau)})$, the forward filtering backward sampling algorithm can be performed (Chib, 1996). The derivations of the full conditional posterior distributions of the parameters of interest are given below.

The conditional posterior distribution of s

The forward filtering backward sampling algorithm is implemented in order to realise a sequence of the hidden states, namely $s^{(\tau)} = (s_1^{(\tau)}, s_2^{(\tau)}, \ldots, s_n^{(\tau)})$. The simulation proceeds with estimating the hidden states from the joint posterior distribution of all the states given the observed data and the remaining parameters (Chib, 1996; Spezia, 2006). It follows that

$$\pi(s \,|\, y, \theta_{-s}) = \pi(s_n \,|\, y, \theta_{-s}) \prod_{t=1}^{n-1} \pi(s_t \,|\, s_{t+1:n}, y, \theta_{-s}),$$

where $\theta_{-s} = (\mu, \lambda, \alpha, \cancel{s}) = (\mu, \lambda, \alpha)$, $s_{i:j} = (s_i, s_{i+1}, \ldots, s_j)$, if $i < j$, and $s_{i:j} = s_i$, if $i = j$. Moreover, $s_{i:j} = \{\}$ whenever $i > j$.

By Bayes' Theorem, $\pi(s_t \,|\, s_{t+1:n}, y, \theta_{-s})$, for $t \in \{1, 2, \ldots, n-1\}$, can be reduced to the following expression:

$$\begin{aligned}
\pi(s_t \,|\, s_{t+1:n}, y, \theta_{-s}) &\propto f(s_{t+1:n}, y_{t+1:n} \,|\, s_t, y_{1:t}, \theta_{-s}) \pi(s_t \,|\, y_{1:t}, \theta_{-s}) \\
&\propto \underbrace{f(s_{t+2:n}, y_{t+1:n} \,|\, s_{t+1}, \cancel{s_t}, y_{1:t}, \theta_{-s})}_{1^{\text{st}} \text{ order Markov chain}} \underbrace{\pi(s_{t+1} \,|\, s_t, \cancel{y_{1:t}}, \theta_{-s})}_{\text{Markov property}} \pi(s_t \,|\, y_{1:t}, \theta_{-s}) \\
&= \underbrace{f(s_{t+2:n}, y_{t+1:n} \,|\, s_{t+1}, y_{1:t}, \theta_{-s})}_{\text{independent of } s_t} \pi(s_{t+1} \,|\, s_t, \theta_{-s}) \pi(s_t \,|\, y_{1:t}, \theta_{-s}) \\
&\propto \pi(s_{t+1} \,|\, s_t, \theta_{-s}) \pi(s_t \,|\, y_{1:t}, \theta_{-s}). \qquad (3.9)
\end{aligned}$$

Thus, the posterior distribution of the state at time t is reduced to a product of the two terms in Equation (3.9), and hence, the recursive simulation of each state from that posterior distribution is able to be executed (Chib, 1996; Scott, 2002).

The conditional posterior distribution of α

For any $\{(i,j) \in \Omega^2 : i \neq j\}$, the full conditional posterior distribution of $\alpha_{i,j}$ such that $\pi(\alpha_{i,j} \,|\, \theta_{-\alpha})$ is given as follows (Spezia, 2006):

$$\pi(\alpha_{i,j} \,|\, \theta_{-\alpha}) \propto \prod_{\{t\geq 2:\, s_{t-1}=i \,\wedge\, s_t=j\}} q^t_{s_{t-1},s_t} \exp\left(-\frac{1}{2}(\alpha_{i,j}-\mu_A)^\intercal \Lambda_A (\alpha_{i,j}-\mu_A)\right)$$

(by Equation (3.8))

$$= \prod_{\{t\geq 2:\, s_{t-1}=i \,\wedge\, s_t=j\}} \frac{\exp(z_t^\intercal \alpha_{i,j})}{1+\sum_{\substack{j\in\Omega\\ j\neq i}} \exp\left(z_t^\intercal \alpha_{i,j}\right)} \exp\left(-\frac{1}{2}(\alpha_{i,j}-\mu_A)^\intercal \Lambda_A (\alpha_{i,j}-\mu_A)\right)$$

$$= \frac{\exp\left(\sum_{\{t\geq 2:\, s_{t-1}=i \,\wedge\, s_t=j\}} z_t^\intercal \alpha_{i,j} - \frac{1}{2}(\alpha_{i,j}-\mu_A)^\intercal \Lambda_A (\alpha_{i,j}-\mu_A)\right)}{\prod_{\{t\geq 2:\, s_{t-1}=i \,\wedge\, s_t=j\}} \left(1+\sum_{\substack{j\in\Omega\\ j\neq i}} \exp(z_t^\intercal \alpha_{i,j})\right)}. \tag{3.10}$$

The joint posterior distribution in Equation (3.10) has a non-standard form, and hence, Gibbs sampler cannot be used for the parameter estimation. Therefore, Metropolis-Hastings algorithm which is one of the MCMC algorithms is applied.

At each Metropolis-Hastings step, $\alpha_i^{(\tau)} = (\alpha_{i,1}^{(\tau)}, \alpha_{i,2}^{(\tau)}, \ldots, \alpha_{i,i-1}^{(\tau)}, \mathbf{0}, \alpha_{i,i+1}^{(\tau)}, \ldots, \alpha_{i,m}^{(\tau)}) \in \mathbb{R}^{K\times m}$ is proposed from a proposal distribution (i.e. the multivariate Gaussian distribution), where $\mathbf{0}$ is a vector of K zeros. Thus, the full conditional posterior distribution of $\alpha_i^{(\tau)}$, for each $i \in \Omega$ is as follows:

$$\pi(\alpha_i^{(\tau)} \,|\, \theta_{-\alpha}) \propto \frac{\exp\left(\sum_{\{t\geq 2:\, s_{t-1}=i\}} z_t^\intercal \alpha_{i,j}^{(\tau)} - \frac{1}{2}\sum_{\substack{j\in\Omega\\ j\neq i}} (\alpha_{i,j}^{(\tau)}-\mu_A)^\intercal \Lambda_A (\alpha_{i,j}^{(\tau)}-\mu_A)\right)}{\prod_{\{t\geq 2:\, s_{t-1}=i\}} \left(1+\sum_{\substack{j\in\Omega\\ j\neq i}} \exp(z_t^\intercal \alpha_{i,j}^{(\tau)})\right)}. \tag{3.11}$$

The conditional posterior distribution of λ

For $\lambda_i \,|\, \theta_{-\lambda}$, for any $i \in \Omega$, it follows that

$$\pi(\lambda_i \,|\, \theta_{-\lambda}) \propto \prod_{\{t\geq 1:\, s_t=i\}} \lambda_i^{\frac{1}{2}} \exp\left(-\frac{1}{2}\frac{(y_t-\mu_i)^2}{\lambda_i^{-1}}\right) \lambda_i^{\alpha_\Lambda - 1} \exp\left(-\beta_\Lambda \lambda_i\right)$$

$$
\begin{aligned}
&= \lambda_i^{\frac{\nu_i^{(\tau)}}{2}} \lambda_i^{\alpha_\Lambda - 1} \exp\left(-\frac{1}{2}\lambda_i \sum_{\{t\geq 1:\, s_t=i\}} (y_t-\mu_i)^2\right) \exp\left(-\beta_\Lambda \lambda_i\right) \\
&= \lambda_i^{\frac{\nu_i^{(\tau)}}{2}+\alpha_\Lambda - 1} \exp\left(-\left(\frac{1}{2} \sum_{\{t\geq 1:\, s_t=i\}} (y_t-\mu_i)^2 + \beta_\Lambda\right)\lambda_i\right). \qquad (3.12)
\end{aligned}
$$

Therefore, $\lambda_i^{(\tau)} \,|\, \theta_{-\lambda}$ is drawn from the full conditional posterior distribution in Equation (3.12), that is,

$$
\lambda_i^{(\tau)} \,|\, \theta_{-\lambda} \sim \mathcal{G}\left(\frac{\nu_i^{(\tau)}}{2} + \alpha_\Lambda, \frac{1}{2} \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} (y_t - \mu_i^{(\tau-1)})^2 + \beta_\Lambda\right), \qquad (3.13)
$$

where $\nu_i^{(\tau)}$ is the number of observations in state i of $s^{(\tau)}$ generated at the τ^{th} iteration, namely $\nu_i^{(\tau)} = \#\{t \geq 1 :\, s_t^{(\tau)} = i\}$ (Spezia, 2006).

The conditional posterior distribution of μ

For $\mu_i \,|\, \theta_{-\mu}$, for any $i \in \{2, 3, \ldots, m\}$, it follows that

$$
\begin{aligned}
\pi(\mu_i \,|\, \theta_{-\mu}) &\propto \left[\prod_{\{t\geq 1:\, s_t=i\}} \exp\left(-\frac{1}{2}\frac{(y_t-\mu_i)^2}{\lambda_i^{-1}}\right)\right] \exp\left(-\frac{1}{2}\frac{(\mu_i-\mu_{i-1})^2}{\lambda_{M_i}^{-1}}\right) \\
&\propto \exp\left(-\frac{1}{2}\frac{\sum_{\{t\geq 1:\, s_t=i\}}(y_t-\mu_i)^2}{\lambda_i^{-1}}\right) \exp\left(-\frac{1}{2}\frac{(\mu_i-\mu_{i-1})^2}{\lambda_{M_i}^{-1}}\right) \\
&= \exp\left(-\frac{1}{2}\left(\lambda_i \sum_{\{t\geq 1:\, s_t=i\}} (y_t^2 - 2y_t\mu_i + \mu_i^2) + \lambda_{M_i}(\mu_i^2 - 2\mu_i\mu_{i-1} + \mu_{i-1}^2)\right)\right) \\
&= \exp\left(-\frac{1}{2}\left(\lambda_i \sum_{\{t\geq 1:\, s_t=i\}} y_t^2 - 2\lambda_i\mu_i \sum_{\{t\geq 1:\, s_t=i\}} y_t + \lambda_i\nu_i\mu_i^2 + \lambda_{M_i}\mu_i^2\right.\right. \\
&\qquad\qquad\qquad\qquad \left.\left. -2\lambda_{M_i}\mu_i\mu_{i-1} + \lambda_{M_i}\mu_{i-1}^2\right)\right) \\
&\propto \exp\left(-\frac{1}{2}\left((\lambda_i\nu_i + \lambda_{M_i})\mu_i^2 - 2\left(\lambda_i \sum_{\{t\geq 1:\, s_t=i\}} y_t + \lambda_{M_i}\mu_{i-1}\right)\mu_i\right)\right) \\
&= \exp\left(-\frac{1}{2}(\lambda_i\nu_i + \lambda_{M_i})\left(\mu_i^2 - 2\left(\frac{\lambda_i \sum_{\{t\geq 1:\, s_t=i\}} y_t + \lambda_{M_i}\mu_{i-1}}{\lambda_i\nu_i + \lambda_{M_i}}\right)\mu_i\right)\right) \\
&= \exp\left(-\frac{1}{2}(\lambda_i\nu_i + \lambda_{M_i})\left(\mu_i - \frac{\lambda_i \sum_{\{t\geq 1:\, s_t=i\}} y_t + \lambda_{M_i}\mu_{i-1}}{\lambda_i\nu_i + \lambda_{M_i}}\right)^2\right.
\end{aligned}
$$

$$- \frac{(\lambda_i \sum_{\{t\geq 1:\, s_t=i\}} y_t + \lambda_{M_i}\mu_{i-1})^2}{\lambda_i\nu_i + \lambda_{M_i}}\Bigg)$$

$$\propto \exp\left(-\frac{1}{2}(\lambda_i\nu_i + \lambda_{M_i})\left(\mu_i - \frac{\lambda_i \sum_{\{t\geq 1:\, s_t=i\}} y_t + \lambda_{M_i}\mu_{i-1}}{\lambda_i\nu_i + \lambda_{M_i}}\right)^2\right). \tag{3.14}$$

Therefore, $\mu_i^{(\tau)} \,|\, \theta_{-\mu}$ is drawn from the full conditional posterior distribution in Equation (3.14). It follows that

$$\mu_i^{(\tau)} \,|\, \theta_{-\mu} \sim \mathcal{N}\left(\frac{\lambda_i^{(\tau)} \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} y_t + \lambda_{M_i}\mu_{i-1}^{(\tau)}}{\lambda_i^{(\tau)}\nu_i + \lambda_{M_i}}, \frac{1}{\lambda_i^{(\tau)}\nu_i + \lambda_{M_i}}\right), \tag{3.15}$$

for any $i \in \{2, 3, \ldots, m\}$. Assume without loss of generality, the full conditional posterior distribution of $\mu_1 \,|\, \theta_{-\mu}$ replaces $\exp(-(\mu_i - \mu_{i-1})^2/2\lambda_{M_i}^{-1})$ with $\exp(-(\mu_1 - \mu_{M_1})^2/2\lambda_{M_1}^{-1})$ (Spezia, 2006). It follows that

$$\pi(\mu_1 \,|\, \theta_{-\mu}) \propto \left\{\prod_{\{t\geq 1:\, s_t=i\}} \exp\left(-\frac{1}{2}\frac{(y_t-\mu_i)^2}{\lambda_i^{-1}}\right)\right\} \exp\left(-\frac{1}{2}\frac{(\mu_1-\mu_{M_1})^2}{\lambda_{M_1}^{-1}}\right)$$

$$\vdots$$

$$\propto \exp\left(-\frac{1}{2}(\lambda_i\nu_i + \lambda_{M_1})\left(\mu_1 - \frac{\lambda_i \sum_{\{t\geq 1:\, s_t=i\}} y_t + \lambda_{M_1}\mu_{M_1}}{\lambda_i\nu_i + \lambda_{M_1}}\right)^2\right).$$

Thus, $\mu_1^{(\tau)} \,|\, \theta_{-\mu}$ is drawn as follows:

$$\mu_1^{(\tau)} \,|\, \theta_{-\mu} \sim \mathcal{N}\left(\frac{\lambda_i^{(\tau)} \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} y_t + \lambda_{M_1}\mu_{M_1}}{\lambda_i^{(\tau)}\nu_i + \lambda_{M_1}}, \frac{1}{\lambda_i^{(\tau)}\nu_i + \lambda_{M_1}}\right). \tag{3.16}$$

As the conditional posterior distributions of the parameters have been detailed, it remains to construct a main algorithm (Algorithm 6) which executes each MCMC sub-algorithm regarding the corresponding parameter.

Upon completing the theoretical components of a Bayesian analysis, the MCMC algorithms to approximate the joint posterior distribution of the parameters in Equation (3.8) are detailed as follows.

3.5 The MCMC Algorithms

Since the joint posterior distribution in Equation (3.8) is analytically intractable, numerical methods such as the MCMC algorithms are used for parameter estimation. To be more specific, a proposal density for the Metropolis-Hastings algorithm can be any multivariate probability density functions of choice to carry out MCMC iterations (Brooks et al., 2011). In this thesis, however, the multivariate Gaussian distribution is used as the proposal density when updating the parameter α_i, for any $i \in \{1, 2, \ldots, m\}$. As the dimension of the multivariate Gaussian distribution increases (i.e. K increases), there is an exponential increase in sampling space for approximating the conditional posterior distribution in Equation (3.11) (Bellman, 2015; Donoho et al., 2000).

The standard Metropolis-Hastings algorithm in such an exhaustive sampling scheme is likely to become computationally inefficient, in addition to the difficulty of determining proposal distributions (Brooks et al., 2011).

Hence, in this thesis, I propose the adaptive MCMC algorithms to tackle the curse of dimensionality and improve the convergence optimally. By recalling Section 2.4, I have proposed and implemented the following algorithms to accelerate the convergence and to become more efficient:

Algorithm 8 the adaptive Metropolis (AM) algorithm,

Algorithm 9 the symmetric delayed rejection adaptive Metropolis (DRAM) algorithm, and

Algorithm 10 the multiple-try adaptive Metropolis (MTAM) algorithm.

The motivations of proposing these algorithms are as follows.

The AM algorithm can be considered as the global adaptive strategy (Haario et al., 2006). Based on the past MCMC samples of the Metropolis-Hastings algorithms, online tuning of the proposal distribution is done so that one can choose the optimal choice of proposed values (Haario et al., 2001). Nevertheless, the algorithm is classified as neither Markovian nor reversible due to the form of its adaptation (Haario et al., 2006, 2001). This adaptive method is designed to scale the covariance matrix of the multivariate Gaussian proposal distribution, so that more efficient exploration of sample space with regard to the parameter α_i, for any $i \in \{1, 2, \ldots, m\}$, can be carried out.

Then, the DRAM algorithm is able to combine different proposal distributions at each rejection stage regardless of symmetry of the proposals. It aims at integrating partial local adaptation of the DRMH algorithm at each step of the Markov chain and the global adaptive strategy of the AM algorithm (Haario et al., 2006). For the sake of simplicity, a symmetric proposal distribution was used for the algorithm. As for the univariate NHGHMM, the symmetric DRAM algorithm is implemented. This algorithm attempts to delay rejection of a proposed parameter whose acceptance probability was relatively low. By delaying the rejection, it is able to increase the acceptance probability, in addition to treating those MCMC chains which get trapped in local modes.

Finally, the MTAM algorithm is proposed to maintain much larger searching regions at each Metropolis-Hastings step due to the means of an MTM algorithm's brute-force by exploiting multiple draws at once (Liu et al., 2000). The scheme of the MTM algorithm is incorporated with that of the AM algorithm. Hence, the MTAM algorithm is now able to propose more plausible values at each Metropolis-Hastings step for more improved MCMC convergence. For the model of interest, the MTAM algorithm is able to maintain both the large searching region and the high acceptance probability. Hence, this algorithm may be ideal for sampling the parameter α_i, for any $i \in \{1, 2, \ldots, m\}$, from the sample space which is potentially vast to cover.

These proposed algorithms implemented for the model essentially differ from that method of Spezia (2006)'s in terms of updating the parameter α.

In the following, Algorithm 6 outlines procedural steps of MCMC iterations, which consists of three sub-algorithms.

Algorithm 6 The Main Algorithm (Univariate NHGHMM)

```
1: Initialise $\theta^{(1)} = (\mu^{(1)}, \lambda^{(1)}, \alpha^{(1)}, s^{(1)})$.
2: for $\tau = 2, 3, \ldots, N$ do
3:     Forward Filtering Backward Sampling Step:
4:     Update $s^{(\tau)} \mid \mu^{(\tau-1)}, \lambda^{(\tau-1)}, \alpha^{(\tau-1)}$ by Algorithm 7.
5:     AM/Symmetric DRAM/MTAM Step:
6:     Update $\alpha^{(\tau)} \mid s^{(\tau)}, \mu^{(\tau-1)}, \lambda^{(\tau-1)}$ by either Algorithm 8, Algorithm 9, or Algorithm 10.
7:     Gibbs Sampling Step:
8:     Update $\lambda^{(\tau)} \mid s^{(\tau)}, \alpha^{(\tau)}, \mu^{(\tau-1)}$ by Gibbs sampler.
9:     Update $\mu^{(\tau)} \mid s^{(\tau)}, \alpha^{(\tau)}, \lambda^{(\tau)}$ by Gibbs sampler with embedded label switching.
10: end for
```

Given the main algorithm (Algorithm 6), each step within the algorithm is described.

Initialisation of the Parameters

The procedure of initialising the parameters is deterministic and closely follows that of Rydén et al. (2008)'s. This initialisation method generates a reasonable sequence of the hidden states which is crucial to calculation of subsequent transition probabilities.

For each $i \in \Omega$, the parameter $\mu_i^{(1)}$ is initialised as follows:

$$\mu_i^{(1)} = \min\{y\} + \frac{R}{2m} + (i-1)\frac{R}{m},$$

where $R = \max\{y\} - \min\{y\}$.

For each $t \in \{1, 2, \ldots, n\}$, the parameter $s_t^{(1)}$ is initialised as follows:

$$s_t^{(1)} = \arg\min_{i \in \Omega} (y_t - \mu_i^{(1)})^2.$$

For each $i \in \Omega$, the parameter $\lambda_i^{(1)}$ is initialised as follows:

$$\lambda_i^{(1)} = \left(\frac{1}{\nu_i^{(1)}} \sum_{\{t \geq 1 :\, s_t^{(1)} = i\}} (y_t - \mu_i^{(1)})^2 \right)^{-1},$$

where $\nu_i^{(1)} = \#\{t \geq 1 : \; s_t^{(1)} = i\}$.

For each $\{(i,j) \in \Omega^2 : i \neq j\}$, define $\alpha_{i,j}^{\circ} = (\alpha_{i,j}^2, \alpha_{i,j}^3, \ldots, \alpha_{i,j}^K)$. Then, the parameter

$\alpha_{i,j}^{(1)}$ is initialised as follows:

$$\alpha_{i,j}^{(1)} = (\log(\nu_{i,j}^{(1)}/\nu_i^{(1)}), \alpha_{i,j}^{\circ}),$$

where $\nu_{i,j}^{(1)} = \#\{t \geq 2 : \ s_{t-1}^{(1)} = i \wedge s_t^{(1)} = j\}$, and

$$\alpha_{i,j}^{\circ} \sim \mathcal{N}_{K-1}(\mu_A^{\circ}, (\Lambda_A^{\circ})^{-1}),$$

where $\mu_A^{\circ} = (\mu_A^2, \mu_A^3, \ldots, \mu_A^K) \in \mathbb{R}^{K-1}$ and $\Lambda_A^{\circ} = (\Lambda_A^{i,j})_{i,j=2}^K \in \mathbb{R}^{(K-1)\times(K-1)}$ are the hyperparameters defined in Equation (3.4).

Forward Filtering Backward Sampling Step

Now, the first sub-algorithm (Algorithm 7) within Algorithm 6 is described. The latter term in Equation (3.9) is the transition probability from s_t to s_{t+1} where its normalising constant is the sum of all $s_t \in \Omega$. It remains to calculate the former term recursively in order to estimate the joint posterior distribution of all the states, $(s_1^{(\tau)}, s_2^{(\tau)}, \ldots, s_n^{(\tau)})$, by the following procedure (Chib, 1996).

(i) At $t = 1$, initialise $\pi(s_t \,|\, y_{1:t}, \theta_{-s}^{(\tau-1)}) = \pi(s_1 \,|\, y_1, \theta_{-s}^{(\tau-1)}) = \rho_i = 1/m$, for all $i \in \Omega$.

(ii) By the law of total probability, it follows that

$$\begin{aligned}\pi(s_{t+1} \,|\, y_{1:t}, \theta_{-s}^{(\tau-1)}) &= \sum_{s_t \in \Omega} \pi(s_{t+1} \,|\, s_t, y_{1:t}, \theta_{-s}^{(\tau-1)}) \pi(s_t \,|\, y_{1:t}, \theta_{-s}^{(\tau-1)}) \\ &= \sum_{s_t \in \Omega} \pi(s_{t+1} \,|\, s_t, \theta_{-s}^{(\tau-1)}) \pi(s_t \,|\, y_{1:t}, \theta_{-s}^{(\tau-1)}); \text{ by Markov property.}\end{aligned}$$

(iii) By Bayes' Theorem, it follows that

$$\pi(s_{t+1} \,|\, y_{1:t+1}, \theta_{-s}^{(\tau-1)}) \propto \pi(s_{t+1} \,|\, y_{1:t}, \theta_{-s}^{(\tau-1)}) f(y_{t+1} \,|\, y_{1:t}, s_{t+1}, \theta_{-s}^{(\tau-1)}),$$

and its normalising constant is given by

$$\sum_{s_{t+1} \in \Omega} \pi(s_{t+1} \,|\, y_{1:t}, \theta_{-s}^{(\tau-1)}) f(y_{t+1} \,|\, y_{1:t}, s_{t+1}, \theta_{-s}^{(\tau-1)}).$$

(iv) Alternate the computations between $\pi(s_{t+1}\,|\,y_{1:t},\theta_{-s}^{(\tau-1)})$ and $\pi(s_{t+1}\,|\,y_{1:t+1},\theta_{-s}^{(\tau-1)})$ in Steps (ii) and (iii), respectively, until obtaining $\pi(s_n\,|\,y,\theta_{-s}^{(\tau-1)})$. Then, realise $s_n^{(\tau)}$ from $\pi(s_n\,|\,y,\theta_{-s}^{(\tau-1)})$.

(v) By Equation (3.9), it is possible to realise $s_t^{(\tau)}$ from $\pi(s_t\,|\,s_{t+1:n},y,\theta_{-s}^{(\tau-1)})$ such that

$$\pi(s_t\,|\,s_{t+1:n},y,\theta_{-s}^{(\tau-1)}) \propto \pi(s_t\,|\,y_{1:t},\theta_{-s}^{(\tau-1)})\pi(s_{t+1}\,|\,s_t,\theta_{-s}^{(\tau-1)}),$$

and its normalising constant is given by $\sum_{s_t\in\Omega}\pi(s_t\,|\,y_{1:t},\theta_{-s}^{(\tau-1)})\pi(s_{t+1}\,|\,s_t,\theta_{-s}^{(\tau-1)})$.

Thus, the realisation of hidden states $(s_1^{(\tau)},s_2^{(\tau)},\ldots,s_n^{(\tau)})$ at τ^{th} iteration can be obtained through the simulation in Step (v) for $t\in\{n-1,n-2,\ldots,1\}$.

The following summarises the steps in the first sub-algorithm (Algorithm 7).

Algorithm 7 Forward Filtering Backward Sampling Algorithm

```
1: Forward Filtering Step:
2: Set $\rho_i = 1/m$, and calculate $\mathcal{F}(S_1 = i) = \rho_i\phi(y_1;\mu_i^{(\tau-1)},\lambda_i^{(\tau-1)})$ for all $i\in\Omega$,
   where the forward variable is denoted by $\mathcal{F}(\cdot)$, and the Gaussian density function
   is denoted by $\phi(\cdot\,;\mu_i,\lambda_i)$, for any $i\in\Omega$.
3: Normalise $\mathcal{F}(S_1=i) \leftarrow \mathcal{F}(S_1=i)/\sum_{i\in\Omega}\mathcal{F}(S_1=i)$, for all $i\in\Omega$.
4: for $t = 2,3,\ldots,n$ do
5:     Calculate $\mathcal{F}(S_t=i)=\sum_{j\in\Omega}\mathcal{F}(S_{t-1}=j)q_{j,i}^t\phi(y_t;\mu_i^{(\tau-1)},\lambda_i^{(\tau-1)})$, for all $i\in\Omega$.
6:     Normalise $\mathcal{F}(S_t=i) \leftarrow \mathcal{F}(S_t=i)/\sum_{i\in\Omega}\mathcal{F}(S_t=i)$, for all $i\in\Omega$.
7: end for
8: Backward Sampling Step:
9: Update $s_n^{(\tau)} = \arg\max_{i\in\Omega}\mathcal{B}(S_n=i) \equiv \arg\max_{i\in\Omega}\mathcal{F}(S_n=i)$, where the backward
   variable is denoted by $\mathcal{B}(\cdot)$.
10: for $t = n-1,n-2,\ldots,1$ do
11:     Calculate $\mathcal{B}(S_t=i)=\mathcal{F}(S_t=i)q^{t+1}_{S_t=i,S_{t+1}=s_{t+1}^{(\tau)}}$.
12:     Normalise $\mathcal{B}(S_t=i) \leftarrow \mathcal{B}(S_t=i)/\sum_{i\in\Omega}\mathcal{B}(S_t=i)$.
13:     Update $s_t^{(\tau)}$ from $\{1,2,\ldots,m\}$ with probability $\mathcal{B}(S_t=i)$.
14: end for
```

Updating the Parameter, $\alpha^{(\tau)}$

The conditional posterior distribution of the parameter $\alpha^{(\tau)}$ in Equation (3.11) has a non-standard form. Thus, it is required to use the Metropolis-Hastings algorithm to update the parameter (recall Section 2.3.1 for a review of the Metropolis-Hastings algorithm). Firstly, it is required to introduce some proposal distribution of $\alpha'_{i,j}$ given

$\alpha_{i,j}^{(\tau-1)}$. Following Spezia (2006), a proposal random walk is defined as follows:

$$\alpha'_{i,j} \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E),$$

where $E \in \mathbb{R}^{K\times K}$ is a constant covariance matrix. In this thesis, the constant covariance matrix E is defined as follows:

$$\varepsilon_{i,j} = \begin{cases} 13, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases},$$

where $\varepsilon_{i,j} \in E$, for any $(i,j) \in \{1,2,\ldots,K\}^2$. This ensures a random walk covers the entire posterior distribution of interest (Spezia, 2006). Thus, the proposal distribution $q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)})$ is given by

$$\begin{aligned} q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}) &= (2\pi)^{-\frac{K}{2}} \det(E)^{\frac{1}{2}} \exp\left(-\frac{1}{2}(\alpha'_{i,j} - \alpha_{i,j}^{(\tau-1)})^\intercal E^{-1}(\alpha'_{i,j} - \alpha_{i,j}^{(\tau-1)})\right) \\ &\propto \exp\left(-\frac{1}{2}(\alpha'_{i,j} - \alpha_{i,j}^{(\tau-1)})^\intercal E^{-1}(\alpha'_{i,j} - \alpha_{i,j}^{(\tau-1)})\right). \end{aligned}$$

Define $\alpha_i^{(\tau)} = (\alpha_{i,1}^{(\tau)}, \alpha_{i,2}^{(\tau)}, \ldots, \alpha_{i,i-1}^{(\tau)}, \mathbf{0}, \alpha_{i,i+1}^{(\tau)}, \ldots, \alpha_{i,m}^{(\tau)}) \in \mathbb{R}^{K\times m}$. Recall the definition of an acceptance probability (Brooks et al., 2011; Metropolis et al., 1953). Then, the acceptance probability of α'_i given $\alpha_i^{(\tau-1)}$ is given as follows:

$$a(\alpha'_i \,|\, \alpha_i^{(\tau-1)}) = \pi(\alpha'_i \,|\, \theta_{-\alpha}) \prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}).$$

Similarly, the acceptance probability of $\alpha_i^{(\tau-1)}$ given α'_i is expressed by

$$a(\alpha_i^{(\tau-1)} \,|\, \alpha'_i) = \pi(\alpha_i^{(\tau-1)} \,|\, \theta_{-\alpha}) \prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha_{i,j}^{(\tau-1)} \,|\, \alpha'_{i,j}),$$

where $\pi(\cdot \,|\, \theta_{-\alpha})$ is the full conditional posterior distribution in Equation (3.11) (Spezia, 2006). Assume the detailed balance condition in which there exists the stationary distribution for the full conditional posterior distribution of α_i, for any $i \in \Omega$. Then, it follows that

$$\frac{a(\alpha_i' \,|\, \alpha_i^{(\tau-1)})}{a(\alpha_i^{(\tau-1)} \,|\, \alpha_i')} = \frac{\pi(\alpha_i' \,|\, \theta_{-\alpha}) \prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha_{i,j}' \,|\, \alpha_{i,j}^{(\tau-1)})}{\pi(\alpha_i^{(\tau-1)} \,|\, \theta_{-\alpha}) \prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha_{i,j}^{(\tau-1)} \,|\, \alpha_{i,j}')}.$$

Given that the proposal distribution is symmetric, the equation above is now reduced to the following expression such that

$$\begin{aligned}\frac{a(\alpha_i' \,|\, \alpha_i^{(\tau-1)})}{a(\alpha_i^{(\tau-1)} \,|\, \alpha_i')} &= \frac{\pi(\alpha_i' \,|\, \theta_{-\alpha}) \cancel{\prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha_{i,j}' \,|\, \alpha_{i,j}^{(\tau-1)})}}{\pi(\alpha_i^{(\tau-1)} \,|\, \theta_{-\alpha}) \cancel{\prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha_{i,j}^{(\tau-1)} \,|\, \alpha_{i,j}')}} \\ &= \frac{\pi(\alpha_i' \,|\, \theta_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \theta_{-\alpha})} \\ \Rightarrow a(\alpha_i' \,|\, \alpha_i^{(\tau-1)}) &:= \min\left\{1, \frac{\pi(\alpha_i' \,|\, \theta_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \theta_{-\alpha})}\right\}. \qquad (3.17)\end{aligned}$$

By Equation (3.17), a Metropolis-Hastings move is accepted such that one sets $\alpha_i^{(\tau)} = \alpha_i'$ with probability $a(\alpha_i' \,|\, \alpha_i^{(\tau-1)})$ at τ^{th} iteration. Otherwise, one rejects it with probability $\left(1 - a(\alpha_i' \,|\, \alpha_i^{(\tau-1)})\right)$ such that $\alpha_i^{(\tau)} = \alpha_i^{(\tau-1)}$.

Based on the acceptance probability in Equation (3.17), one can choose whether α_i' be accepted or not.

During the MCMC iterations, the covariance matrix E of a proposal distribution is designed to be tuned so that an acceptance rate, $A(\tau)$ at the τ^{th} iteration, is maintained optimal. The acceptance rate is given as follows:

$$A(\tau) = (\tau-1)^{-1}\left(a(\alpha_i' \,|\, \alpha_i^{(\tau-1)}) + \sum_{\nu=2}^{\tau-1} a(\alpha_i^{*(\nu)} \,|\, \alpha_i^{(\nu-1)})\right), \qquad (3.18)$$

where $\alpha_i^{*(\nu)}$ denotes the parameter proposed at ν^{th} iteration, for any $\nu \in \{1, 2, \dots, \tau - 1\}$.

Regardless of tuning the covariance matrix E the Metropolis-Hastings algorithm itself may suffer from slow convergence. Haario et al. (2001) demonstrated a problem where the tuning of the proposal distribution according to the acceptance rate only may lead to difficulties. Additionally, the problem becomes more prominent as the number of dimensions of the target distribution increases (Haario et al., 2001). Furthermore, one cannot find more complicated improvements through this tuning method, for example making the proposal covariance matrix proportional to the optimal one (Brooks et al., 2011). I especially deal with updating the parameter α which potentially involves sampling from the high dimensional multivariate Gaussian distribution.

On the other hand, the AM algorithm demonstrated the potential where it can "learn" the target covariance matrix, as well as approach an optimal acceptance rate, even in very high dimensions (Brooks et al., 2011; Haario et al., 2001).

As such, I need to introduce the AM algorithm to adapt the covariance matrix of the proposal distribution so that the algorithm's efficiency can be improved more.

The AM Step

In an attempt to propose a different sub-algorithm as an alternative to the Metropolis-Hasting algorithm, I now discuss the AM algorithm.

An adequate choice of a proposal distribution for the MCMC algorithm is crucial for fast convergence (Haario et al., 2001). This algorithm aims at accumulating the information of the MCMC iterations and updating the multivariate Gaussian proposal distribution along the process (recall Section 2.4.1 on Page 34).

Following Mira et al. (2001), let the covariance matrix, E, be a function of $(\alpha_i^{(1)}, \alpha_i^{(2)}, \ldots, \alpha_i^{(\tau)})$ such that

$$E(\alpha_i^{(1)}, \alpha_i^{(2)}, \ldots, \alpha_i^{(\tau)}) = E_i^{(\tau)} \in \mathbb{R}^{K \times K},$$

where $\alpha_i^{(\nu)} = (\alpha_{i,1}^{(\nu)}, \alpha_{i,2}^{(\nu)}, \ldots, \alpha_{i,i-1}^{(\nu)}, \mathbf{0}, \alpha_{i,i+1}^{(\nu)}, \ldots, \alpha_{i,m}^{(\nu)}) \in \mathbb{R}^{K \times m}$, for any $i \in \Omega$, for each $\nu \in \{1, 2, \ldots, \tau\}$. Thus, $E_i^{(\tau)}$ is now able to store the history of the parameter up to τ^{th} MCMC iteration. By Equation (2.19) on Page 35, the covariance matrix for the Gaussian proposal with mean of a current draw $\alpha_i^{(\tau)}$ is defined as follows:

$$E_i^{(\tau)} = \begin{cases} E_0, & \text{if } \tau \leq \tau_0 \\ s_K \operatorname{Cov}(\alpha_i^{(1)}, \alpha_i^{(2)}, \ldots, \alpha_i^{(\tau-1)}) + s_K \varepsilon I_K, & \text{if } \tau > \tau_0 \end{cases}, \tag{3.19}$$

where E_0 is a constant covariance matrix of choice, $\tau_0 > 0$ is the initial period, $s_K = 2.4^2/K$, $\varepsilon > 0$ is an infinitesimal value, and I_K is a $K \times K$ identity matrix.

Following Equation (2.20) on Page 36, the derivation of a recursive formula for the MCMC iterations of $\alpha_i^{(\tau)}$ is given as follows:

$$E_i^{(\tau)} = \frac{\tau-3}{\tau-2}E_i^{(\tau-1)} + \frac{s_K}{\tau-2}\left[\frac{1}{\tau-1}\left((\tau-2)\bar{\alpha}_i^{(\tau-2)}\bar{\alpha}_i^{(\tau-2)\intercal}\right.\right.$$
$$\left.\left.-(\tau-1)\left(\bar{\alpha}_i^{(\tau-1)}\alpha_i^{(\tau-1)\intercal} + \left(\bar{\alpha}_i^{(\tau-1)}\alpha_i^{(\tau-1)\intercal}\right)^\intercal\right) + \tau\alpha_i^{(\tau-1)}\alpha_i^{(\tau-1)\intercal}\right) + \varepsilon I_K\right]. \tag{3.20}$$

The choice for the length of the initial segment $\tau_0 > 0$ is unspecified. Nonetheless, an faster effect of the adaptation is anticipated with a smaller value of τ_0 (Haario et al., 2001).

Following that initial value setting of Haario et al. (2001)'s, the algorithm proceeds as follows.

(i) Initialise $s_K = 2.4^2/K$ and $\varepsilon = 10^{-6}$. Set the initial period, $\tau_0 = 50$.

(ii) If the current MCMC iteration $\tau = \tau_0$, then initialise arithmetic means $\bar{\alpha}_i^{(\tau-2)}$ and $\bar{\alpha}_i^{(\tau-1)}$, for all $i \in \Omega$.

(iii) If the current MCMC iteration, $\tau > \tau_0$, then update the arithmetic means $\bar{\alpha}_i^{(\tau-2)}$ and $\bar{\alpha}_i^{(\tau-1)}$ such that

$$\bar{\alpha}_i^{(\tau-2)} \leftarrow \bar{\alpha}_i^{(\tau-2)} + \frac{\alpha_i^{(\tau-2)} - \bar{\alpha}_i^{(\tau-2)}}{\tau-2}$$
$$\bar{\alpha}_i^{(\tau-1)} \leftarrow \bar{\alpha}_i^{(\tau-1)} + \frac{\alpha_i^{(\tau-1)} - \bar{\alpha}_i^{(\tau-1)}}{\tau-1},$$

respectively, for all $i \in \Omega$.

(iv) Now, it is required to compute the covariance matrices of the multivariate Gaussian distribution by Equation (3.20), for all $i \in \Omega$.

(v) By monitoring the acceptance rate, $A(\tau)$ in Equation (3.18), tune the covariance matrices $E_i^{(\tau)}$, for all $i \in \Omega$. If $A(\tau) < 0.25$, then tune $E_i^{(\tau)} \leftarrow 0.95^2 E_i^{(\tau)}$. If $A(\tau) > 0.5$, then tune $E_i^{(\tau)} \leftarrow 1.05^2 E_i^{(\tau)}$.

(vi) It remains to propose a new parameter for $\alpha_i^{(\tau)}$, for each $i \in \Omega$, from the proposal distribution such that

$$\alpha'_{i,j} \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E_i^{(\tau)}), \tag{3.21}$$

for all $j \in \Omega$. Then, calculate the acceptance probability of α'_i and $\alpha_i^{(\tau-1)}$, for each $i \in \Omega$, as in Equation (3.17). Since the proposal distribution $q(\alpha'_{i,j} \mid \alpha_{i,j}^{(\tau-1)})$ is symmetric, for all $j \in \Omega$. Therefore, the acceptance probability of α'_i given $\alpha_i^{(\tau-1)}$ is the same as Equation (3.17).

The following summarises the steps in the sub-algorithm (Algorithm 8).

Algorithm 8 The Adaptive Metropolis Algorithm

```
1: Initialise the parameters of the AM algorithm, $s_K = 2.4^2/K$ and $\varepsilon = 10^{-6} > 0$.
2: Set the initial period of the MCMC iterations, $\tau_0$ and $E_i^{(\tau_0)} = E_0$, for all $i \in \{1, 2, \dots, m\}$.
3: Define $\alpha_i = (\alpha_{i,1}, \alpha_{i,2}, \dots, \alpha_{i,i-1}, \mathbf{0}, \alpha_{i,i+1}, \dots, \alpha_{i,m})$, for all $i \in \Omega$.
4: for $i = 1, 2, \dots, m$ do
5:     if $\tau = \tau_0$ then
6:         Initialise $\bar{\alpha}_i^{(\tau-2)} = \sum_{\kappa=1}^{\tau-2} \alpha_i^{(\kappa)} \Big/ (\tau-2)$ and $\bar{\alpha}_i^{(\tau-1)} = \sum_{\kappa=1}^{\tau-1} \alpha_i^{(\kappa)} \Big/ (\tau-1)$.
7:     end if
8:     if $\tau > \tau_0$ then
9:         Compute $\bar{\alpha}_i^{(\tau-2)} \leftarrow \bar{\alpha}_i^{(\tau-2)} + (\alpha_i^{(\tau-2)} - \bar{\alpha}_i^{(\tau-2)}) \Big/ (\tau-2)$ and $\bar{\alpha}_i^{(\tau-1)} \leftarrow \bar{\alpha}_i^{(\tau-1)} + (\alpha_i^{(\tau-1)} - \bar{\alpha}_i^{(\tau-1)}) \Big/ (\tau-1)$.
10:        Compute
           $$E_i^{(\tau)} = \frac{\tau-3}{\tau-2} E_i^{(\tau-1)} + \frac{s_K}{\tau-2} \left[ \frac{1}{\tau-1} \left( (\tau-2) \bar{\alpha}_i^{(\tau-2)} \bar{\alpha}_i^{(\tau-2)\intercal} - (\tau-1) \left( \bar{\alpha}_i^{(\tau-1)} \alpha_i^{(\tau-1)\intercal} + \left( \bar{\alpha}_i^{(\tau-1)} \alpha_i^{(\tau-1)\intercal} \right)^\intercal \right) + \tau \alpha_i^{(\tau-1)} \alpha_i^{(\tau-1)\intercal} \right) + \varepsilon I_K \right],$$
           where $I_K$ denotes the identity matrix of $K$ dimensions.
11:        if $A(\tau) < 0.25$ then
12:            $E_i^{(\tau)} \leftarrow 0.95^2 E_i^{(\tau)}$
13:        end if
14:        if $A(\tau) > 0.5$ then
15:            $E_i^{(\tau)} \leftarrow 1.05^2 E_i^{(\tau)}$
16:        end if
17:        Random Walk Metropolis-Hastings Step: The random walk Metropolis-Hastings algorithm, the DRMH algorithm or the MTM algorithm can be implemented in this step to update $\alpha_i^{(\tau)}$.
18:    end if
19: end for
```

Naturally, the adaptation of the covariance matrix becomes slower as the dimension increases and may be more sensitive to an inadequate choice of the initial covariance

(Haario et al., 2001). To overcome this problem with high dimensionality, Haario et al. (2006) constructed the algorithm where the DRMH and AM algorithms were combined. Such a means of proposing a new parameter can prevent both over- and under-calibrated covariance matrices (Haario et al., 2006). Thus, faster convergence is achievable in higher dimensional target distributions.

Step (vi) of the AM algorithm can take several forms of proposing a new parameter for $\alpha_i^{(\tau)}$. Following Haario et al. (2006), I have combined the symmetric delayed rejection Metropolis and the AM algorithms to construct the ensure faster convergence and more efficiency. The resulting algorithm, the symmetric DRAM algorithm, is described below.

The Symmetric DRAM Step

This algorithm is a combination of the symmetric delayed rejection Metropolis (recall Section 2.4.4 on Page 40) and the adaptive Metropolis algorithms. The algorithm differs from the random walk Metropolis-Hastings algorithm for which rejection of a new parameter is delayed by subsequent rejection stages (Mira and Tierney, 2002; Tierney and Mira, 1999). It modifies the manner of proposing a new parameter in Equation (3.21).

Figure 3.1 shows the comparison between the standard Metropolis-Hastings algorithm and the DRMH algorithm. The target distribution is as follows:

$$\pi(\theta) = \frac{9}{10}\varphi(\theta; 0, 1) + \frac{1}{10}\varphi(\theta; 10, 1),$$

where $\varphi(\cdot\,; \mu, \sigma^2)$ denotes the Gaussian density function with the mean μ and the variance σ^2. This toy example follows that of Gamerman and Lopes (2006)'s. This suggests that the MCMC iterations in the DRMH algorithm mix better than those of the Metropolis-Hastings algorithm. In this thesis, the *symmetric delayed rejection Metropolis algorithm* was specifically used since the proposal distribution is the multivariate Gaussian density which is symmetric (Mira et al., 2001). The symmetric

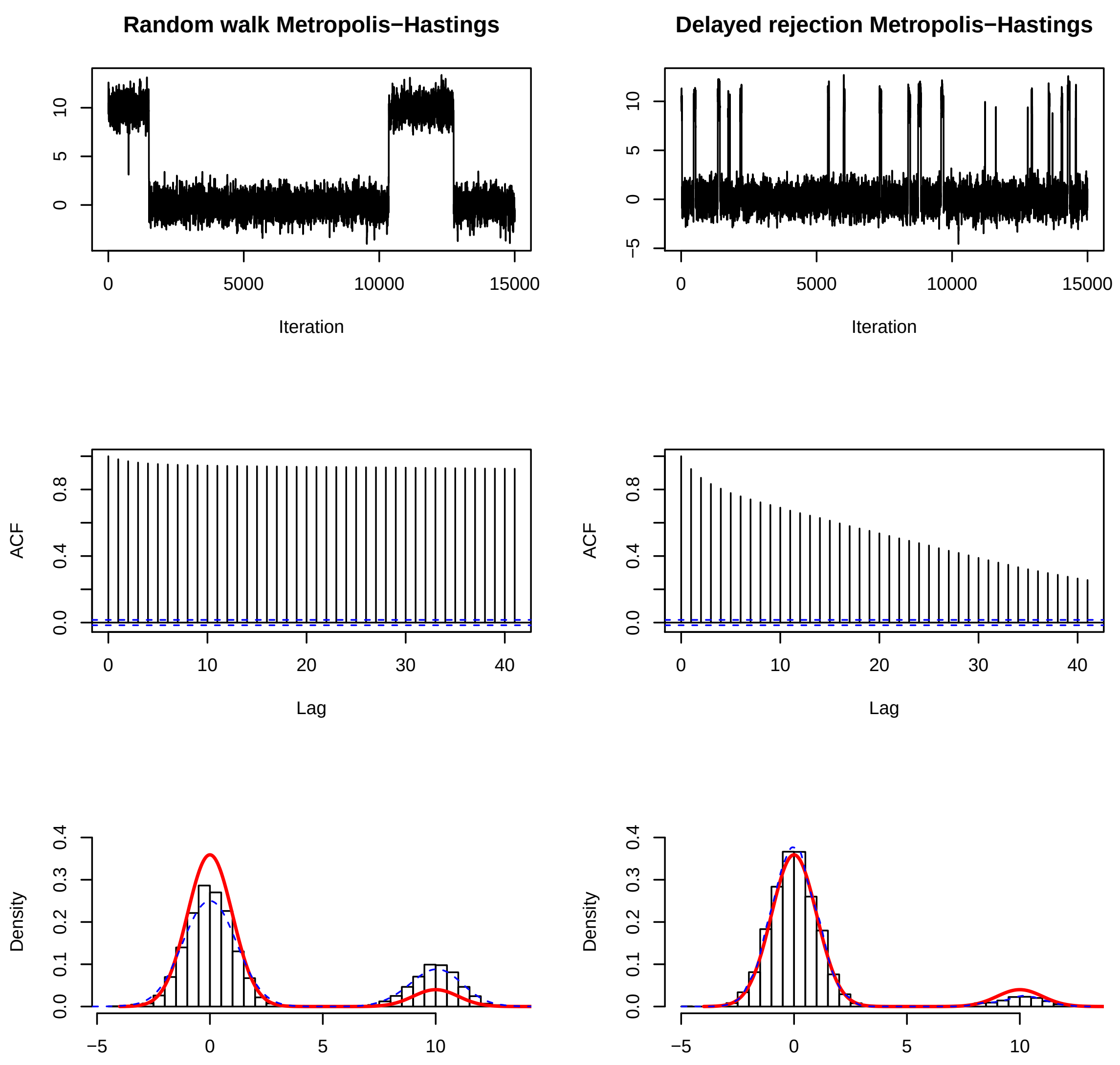


Figure 3.1: Trace plots (top), autocorrelation functions (middle), and marginal posterior densities (bottom) of the MCMC samples with respect to the parameter, θ, for a comparison between the standard random walk Metropolis-Hastings and DRMH algorithms. The red curved line represents the target distribution.

proposal distribution only depends on the last rejected sample such that

$$q(\alpha'_{i,j(\kappa)}\,|\,\alpha'_{i,j(\kappa-1)},\alpha'_{i,j(\kappa-2)},\ldots,\alpha'_{i,j(1)},\alpha_{i,j}^{(\tau-1)}) = q(\alpha'_{i,j(\kappa)}\,|\,\alpha'_{i,j(\kappa-1)}) = q(\alpha'_{i,j(\kappa-1)}\,|\,\alpha'_{i,j(\kappa)}),$$

where $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha'_{i,j(\kappa-1)}, E_i^{(\tau)})$, for any $\kappa \in \{2,3,\ldots,\nu\}$ and any $\{(i,j) \in \Omega^2 : i \neq j\}$. The algorithm proceeds as follows (Haario et al., 2006, 2001; Mira et al., 2001).

(i) Set a rejection stage indicator, $\kappa = 1$, and the maximum rejection stage, ν.

(ii) If $\kappa < 2$, then propose $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E_i^{(\tau)})$, for all $\{(i,j) \in \Omega^2 : i \neq j\}$. Then, define $\alpha'_{i(\kappa)} = (\alpha'_{i,1(\kappa)}, \alpha'_{i,2(\kappa)}, \ldots, \alpha'_{i,i-1(\kappa)}, \mathbf{0}, \alpha'_{i,i+1(\kappa)}, \ldots, \alpha'_{i,m(\kappa)})$, and set $\alpha^*_i = \alpha'_{i(\kappa)}$.

(iii) Since the symmetric proposal distribution was implemented, the acceptance probability is given as follows:

$$r_1 = \min\left\{1, \frac{\pi(\alpha'_{i(\kappa)} \,|\, \theta_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \theta_{-\alpha})}\right\},$$

where $\pi(\alpha_i \,|\, \theta_{-\alpha})$ is the conditional posterior distribution, for any $i \in \Omega$ in Equation (3.11). Draw $u \sim \mathcal{U}(0,1)$. If $u \leq r_1$, then accept $\alpha_i^{(\tau)} = \alpha'_{i(\kappa)}$ and terminate the algorithm. Otherwise, set $\kappa \leftarrow \kappa + 1$ and move to Step (iv).

(iv) If $\kappa \geq 2$, then propose $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha'_{i,j(\kappa-1)}, E_i^{(\tau)})$, for all $\{(i,j) \in \Omega^2 : i \neq j\}$.

(v) Calculation of the acceptance probability is now different from that of Step (iii). Therefore,

$$r_2 = \min\left\{1, \frac{\pi(\alpha'_{i(\kappa)} \,|\, \theta_{-\alpha}) - \pi(\alpha^*_i \,|\, \theta_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \theta_{-\alpha}) - \pi(\alpha^*_i \,|\, \theta_{-\alpha})}\right\}.$$

(vi) If $\pi(\alpha'_{i(\kappa)} \,|\, \theta_{-\alpha}) > \pi(\alpha^*_i \,|\, \theta_{-\alpha})$, then set $\alpha^*_i = \alpha'_{i(\kappa)}$.

(vii) Draw $u \sim \mathcal{U}(0,1)$. If $u \leq r_2$, set $\alpha_i^{(\tau)} = \alpha'_{i(\kappa)}$ and terminate the algorithm. Otherwise, set $\kappa \leftarrow \kappa + 1$ and move to Step (ii). If $\kappa > \nu$, then the algorithm is terminated.

Algorithm 9 summarises the steps in the symmetric DRAM algorithm. The symmetric delayed rejection Metropolis algorithm is schematically different from the MTM algorithm. The symmetric delayed rejection Metropolis algorithm preserves the last rejected sample whilst the random walk resumes at that candidate.

On the other hand, the MTM algorithm draws multiple candidates about the last MCMC iteration from the multivariate Gaussian distribution, and then, a proposed value for the next MCMC iteration is selected from a pool of multiple candidates (Liu et al., 2000). The multiple candidates can be seen as a means of preserving a wide searching region so that each Metropolis-Hastings step become less localised.

Several modifications of the MTM algorithm were proposed by Bédard et al. (2012); Craiu and Lemieux (2007). The original MTM algorithm utilised a pool of *independent*

Algorithm 9 The Symmetric Delayed Rejection Adaptive Metropolis Algorithm

```
1: Initialise a rejection stage indicator, $\kappa = 1$, and the maximum rejection stage, $\nu$.
2: for $i = 1, 2, \ldots, m$ do
3:     repeat
4:         if $\kappa < 2$ then
5:             for $j \in \Omega$ such that $j \neq i$ do
6:                 Simulate $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E_i^{(\tau)})$.
7:             end for
8:             Define $\alpha'_{i(\kappa)} = (\alpha'_{i,1(\kappa)}, \alpha'_{i,2(\kappa)}, \ldots, \alpha'_{i,i-1(\kappa)}, \mathbf{0}, \alpha'_{i,i+1(\kappa)}, \ldots, \alpha'_{i,m(\kappa)})$, and set $\alpha_i^* = \alpha'_{i(\kappa)}$.
9:             Calculate
                   $$r_1 = \min\left\{1, \frac{\pi(\alpha'_{i(\kappa)} \,|\, \theta_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \theta_{-\alpha})}\right\}.$$
10:            Draw $u \sim \mathcal{U}(0,1)$.
11:            if $u \leq r$ then
12:                $\alpha_i^{(\tau)} = \alpha'_{i(\kappa)}$
13:            else
14:                $\kappa \leftarrow \kappa + 1$, and move onto the next rejection stage.
15:            end if
16:        else
17:            for $j \in \Omega$ such that $j \neq i$ do
18:                Simulate $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha'_{i,j(\kappa-1)}, E_i^{(\tau)})$.
19:            end for
20:            Calculate
                   $$r_2 = \min\left\{1, \frac{\pi(\alpha'_{i(\kappa)} \,|\, \theta_{-\alpha}) - \pi(\alpha_i^* \,|\, \theta_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \theta_{-\alpha}) - \pi(\alpha_i^* \,|\, \theta_{-\alpha})}\right\}.$$
21:            if $\pi(\alpha'_{i(\kappa)} \,|\, \theta_{-\alpha}) > \pi(\alpha_i^* \,|\, \theta_{-\alpha})$ then
22:                Set $\alpha_i^* = \alpha'_{i(\kappa)}$.
23:            end if
24:            Draw $u \sim \mathcal{U}(0,1)$.
25:            if $u \leq r$ then
26:                $\alpha_i^{(\tau)} = \alpha'_{i(\kappa)}$
27:            else
28:                $\kappa \leftarrow \kappa + 1$, and move onto the next rejection stage.
29:            end if
30:        end if
31:    until $\alpha'_{i(\kappa)}$ is accepted or the loop reaches the $\nu^{\text{th}}$ stage of rejection.
32: end for
```

multiple candidates, whereas Craiu and Lemieux (2007)'s modified MTM algorithm, *multiple correlated-try Metropolis algorithm*, aimed at adapting the MTM algorithm to *dependent* multiple candidates. Furthermore, Bédard et al. (2012) extended Craiu and Lemieux (2007)'s work to even more extreme form of dependence in which all the candidates are drawn using a common random variable, *multiple-try Metropolis with* *common random variables algorithm.*

Thus, the MTM and AM algorithms can be undeniably combined to obtain a larger searching region and adapt the covariance matrix of a proposal distribution simultaneously. Hence, I hope to obtain a larger searching region than that of the symmetric DRAM algorithm's. In the following, the resulting algorithm, the MTAM algorithm, is described in detail.

The MTAM Step

Likewise, this algorithm is another combination of the two algorithms, namely the MTM (recall Section 2.4.2 on Page 37) and AM algorithms. Firstly, it is required to define a weighting function $w(\cdot\,|\,\cdot)$ in order to perform the MTAM algorithm. Recall Equations (2.22) and (2.23) on Page 38. Then, the weighting function is given as follows:

$$\begin{aligned}
w(\alpha'_i\,|\,\alpha_i^{(\tau-1)}) &= \pi(\alpha'_i\,|\,\theta_{-\alpha})\left[\prod_{\substack{j\in\Omega\\ j\neq i}} q(\alpha'_{i,j}\,|\,\alpha_{i,j}^{(\tau-1)})\lambda(\alpha'_{i,j}\,|\,\alpha_{i,j}^{(\tau-1)})\right]\\
&= \pi(\alpha'_i\,|\,\theta_{-\alpha})\left[\prod_{\substack{j\in\Omega\\ j\neq i}} q(\alpha'_{i,j}\,|\,\alpha_{i,j}^{(\tau-1)})\left(\frac{q(\alpha'_{i,j}\,|\,\alpha_{i,j}^{(\tau-1)})+q(\alpha_{i,j}^{(\tau-1)}\,|\,\alpha'_{i,j})}{2}\right)^{-1}\right]\\
&= \pi(\alpha'_i\,|\,\theta_{-\alpha})\left[\prod_{\substack{j\in\Omega\\ j\neq i}} q(\alpha'_{i,j}\,|\,\alpha_{i,j}^{(\tau-1)})\left(\frac{\cancel{2}q(\alpha'_{i,j}\,|\,\alpha_{i,j}^{(\tau-1)})}{\cancel{2}}\right)^{-1}\right]\\
&\qquad (\text{since } q(\alpha'_{i,j}\,|\,\alpha_{i,j}^{(\tau-1)}) = q(\alpha_{i,j}^{(\tau-1)}\,|\,\alpha'_{i,j}), \text{ for any } \{j\in\Omega:\ j\neq i\})\\
&= \pi(\alpha'_i\,|\,\theta_{-\alpha})\left[\prod_{\substack{j\in\Omega\\ j\neq i}} \cancel{q(\alpha'_{i,j}\,|\,\alpha_{i,j}^{(\tau-1)})}\frac{1}{\cancel{q(\alpha'_{i,j}\,|\,\alpha_{i,j}^{(\tau-1)})}}\right]\\
&= \pi(\alpha'_i\,|\,\theta_{-\alpha}). \qquad (3.22)
\end{aligned}$$

An acceptance probability of the MTAM algorithm is computed by Equation (2.24) on Page 38. Since the weighting function is merely equal to the posterior distribution as shown in Equation (3.22), the acceptance probability is given as follows:

$$r_3 = a(\alpha_i' \,|\, \alpha_i^{(\tau-1)}) := \min\left\{1, \frac{\pi(\alpha_{i(1)}^{*} \,|\, \theta_{-\alpha}) + \pi(\alpha_{i(2)}^{*} \,|\, \theta_{-\alpha}) + \cdots + \pi(\alpha_{i(\varkappa)}^{*} \,|\, \theta_{-\alpha})}{\pi(\alpha_{i(1)}^{**} \,|\, \theta_{-\alpha}) + \pi(\alpha_{i(2)}^{**} \,|\, \theta_{-\alpha}) + \cdots + \pi(\alpha_{i(\varkappa)}^{**} \,|\, \theta_{-\alpha})}\right\}, \tag{3.23}$$

where $\varkappa \in \mathbb{N}$ is the number of multiple tries. It is apparent that computational costs will increase as the number of multiple tries increases (Liu et al., 2000). It was found that 10 multiple tries make the sufficient number of trials, yet still computationally inexpensive. Thus, I set $\varkappa = 10$ in the main algorithm. The algorithm is then described as follows (Liu et al., 2000; Mira et al., 2001).

(i) For $j \in \{1, 2, \ldots, i-1, i+1, \ldots, m\}$, simulate multiple samples, $\{\alpha_{i,j(1)}^{*}, \alpha_{i,j(2)}^{*}, \ldots, \alpha_{i,j(\varkappa)}^{*}\}$, by drawing i.i.d. samples $\alpha_{i,j(\kappa)}^{*} \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E_i^{(\tau)})$, for each $\kappa \in \{1, 2, \ldots, \varkappa\}$.

(ii) Define $\alpha_{i(\kappa)}^{*} = (\alpha_{i,1(\kappa)}^{*}, \alpha_{i,2(\kappa)}^{*}, \ldots, \alpha_{i,i-1(\kappa)}^{*}, \mathbf{0}, \alpha_{i,i+1(\kappa)}^{*}, \ldots, \alpha_{i,m(\kappa)}^{*})$, and draw a proposed value, α_i', from $\{\alpha_{i(1)}^{*}, \alpha_{i(2)}^{*}, \ldots, \alpha_{i(\varkappa)}^{*}\}$ with probability proportional to $\{\pi(\alpha_{i(\kappa)}^{*} \,|\, \theta_{-\alpha})\}$, for any $\kappa \in \{1, 2, \ldots, \varkappa\}$.

(iii) For $j \in \{1, 2, \ldots, i-1, i+1, \ldots, m\}$, simulate multiple samples, $\{\alpha_{i,j(1)}^{**}, \alpha_{i,j(2)}^{**}, \ldots, \alpha_{i,j(\varkappa-1)}^{**}\}$, by drawing i.i.d. samples, $\alpha_{i,j(\kappa)}^{**} \sim \mathcal{N}_K(\alpha_{i,j}', E_i^{(\tau)})$, for each $\kappa \in \{1, 2, \ldots, \varkappa - 1\}$. Then, set $\alpha_{i(\varkappa)}^{**} = \alpha_i^{(\tau-1)}$, and hence, it follows that $\{\alpha_{i(1)}^{**}, \alpha_{i(2)}^{**}, \ldots, \alpha_{i(\varkappa)}^{**}\}$.

(iv) Calculate the acceptance probability provided by Equation (3.23). That is,

$$r_3 = \min\left\{1, \frac{\pi(\alpha_{i(1)}^{*} \,|\, \theta_{-\alpha}) + \pi(\alpha_{i(2)}^{*} \,|\, \theta_{-\alpha}) + \cdots + \pi(\alpha_{i(\varkappa)}^{*} \,|\, \theta_{-\alpha})}{\pi(\alpha_{i(1)}^{**} \,|\, \theta_{-\alpha}) + \pi(\alpha_{i(2)}^{**} \,|\, \theta_{-\alpha}) + \cdots + \pi(\alpha_{i(\varkappa)}^{**} \,|\, \theta_{-\alpha})}\right\}.$$

(v) Draw $u \sim \mathcal{U}(0,1)$. If $u \leq r_3$, then set $\alpha_i^{(\tau)} = \alpha_i'$. Otherwise, set $\alpha_i^{(\tau)} = \alpha_i^{(\tau-1)}$, for all $i \in \Omega$.

(vi) Repeat Steps (i)–(v), for all $i \in \Omega$.

Now, Algorithm 10 summarises the steps in the MTAM algorithm. After updating the parameter, $\alpha^{(\tau)}$, it remains to update $\lambda^{(\tau)}$ and $\mu^{(\tau)}$ by the Gibbs sampler.

Algorithm 10 The Multiple-try Adaptive Metropolis Algorithm

```
1: for $i = 1, 2, \ldots, m$ do
2:     for $j \in \Omega$ such that $j \neq i$ do
3:         Simulate multiple random samples, $\{\alpha^*_{i,j(1)}, \alpha^*_{i,j(2)}, \ldots, \alpha^*_{i,j(\varkappa)}\}$, by proposing $\alpha^*_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha^{(\tau-1)}_{i,j}, E^{(\tau)}_i)$, for any $\kappa \in \{1, 2, \ldots, \varkappa\}$.
4:     end for
5:     Define $\alpha^*_{i(\kappa)} = (\alpha^*_{i,1(\kappa)}, \alpha^*_{i,2(\kappa)}, \ldots, \alpha^*_{i,i-1(\kappa)}, \mathbf{0}, \alpha^*_{i,i+1(\kappa)}, \ldots, \alpha^*_{i,m(\kappa)})$, and draw $\alpha'_i$ from $\{\alpha^*_{i(1)}, \alpha^*_{i(2)}, \ldots, \alpha^*_{i(\varkappa)}\}$ with probability which is proportional to $\{\pi(\alpha^*_{i(\kappa)} \,|\, \theta_{-\alpha})\}$, where $\kappa \in \{1, 2, \ldots, \varkappa\}$.
6:     for $j \in \Omega$ such that $j \neq i$ do
7:         Simulate multiple random samples, $\{\alpha^{**}_{i,j(1)}, \alpha^{**}_{i,j(2)}, \ldots, \alpha^{**}_{i,j(\varkappa-1)}\}$, by proposing $\alpha^{**}_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha'_{i,j}, E^{(\tau)}_i)$, for any $\kappa \in \{1, 2, \ldots, \varkappa - 1\}$.
8:     end for
9:     Set $\alpha^{**}_{i(\varkappa)} = \alpha^{(\tau-1)}_i$, and calculate the acceptance probability as follows:
           $$r_3 = \min\left\{1, \frac{\pi(\alpha^*_{i(1)} \,|\, \theta_{-\alpha}) + \pi(\alpha^*_{i(2)} \,|\, \theta_{-\alpha}) + \cdots + \pi(\alpha^*_{i(\varkappa)} \,|\, \theta_{-\alpha})}{\pi(\alpha^{**}_{i(1)} \,|\, \theta_{-\alpha}) + \pi(\alpha^{**}_{i(2)} \,|\, \theta_{-\alpha}) + \cdots + \pi(\alpha^{**}_{i(\varkappa)} \,|\, \theta_{-\alpha})}\right\}.$$
10:    Draw $u \sim \mathcal{U}(0, 1)$.
11:    if $u \leq r_3$ then
12:        $\alpha^{(\tau)}_i = \alpha'_i$
13:    else
14:        $\alpha^{(\tau)}_i = \alpha^{(\tau-1)}_i$
15:    end if
16: end for
```

Gibbs Sampling Step

In Section 3.4, the full conditional posterior distributions are obtained in Equation (3.13) for $\lambda^{(\tau)}$, and Equations (3.16) & (3.15) for $\mu^{(\tau)}_1$ and $\{\mu^{(\tau)}_i\}^m_{i=2}$, respectively. All those full conditional posterior distributions have standard forms. Therefore, the Gibbs sampler is considered as an appropriate MCMC algorithm for these full conditionals. The Gibbs sampler proceeds as follows (Spezia, 2006).

(i) For each $i \in \Omega$, simulate

$$\lambda^{(\tau)}_i \,|\, \theta_{-\lambda} \sim \mathcal{G}\left(\frac{\nu^{(\tau)}_i}{2} + \alpha_\Lambda, \frac{1}{2} \sum_{\{t \geq 1:\, s^{(\tau)}_t = i\}} (y_t - \mu^{(\tau-1)}_i)^2 + \beta_\Lambda\right),$$

where $\mathcal{G}(\cdot, \cdot)$ denotes the gamma distribution, $\nu^{(\tau)}_i = \#\{t \geq 1 :\, s^{(\tau)}_t = i\}$, and α_Λ & β_Λ are hyperparameters.

(ii) Simulate

$$\mu_1^{(\tau)} \,|\, \theta_{-\mu} \sim \mathcal{N}\left(\frac{\lambda_i^{(\tau)} \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} y_t + \lambda_{M_1}\mu_{M_1}}{\lambda_i^{(\tau)}\nu_i^{(\tau)} + \lambda_{M_1}}, \frac{1}{\lambda_i^{(\tau)}\nu_i^{(\tau)} + \lambda_{M_1}}\right),$$

where $\mu_{M_1} = \min\{y\} + (\max\{y\} - \min\{y\})/2m$, and $\lambda_{M_1} = \nu_1^{(1)} \Big/ \sum_{\{t\leq 1:\, s_t^{(1)}=1\}} (y_t - \mu_1^{(1)})^2$.

(iii) For each $i \in \{2, 3, \ldots, m\}$, simulate

$$\mu_i^{(\tau)} \,|\, \theta_{-\mu} \sim \mathcal{N}\left(\frac{\lambda_i^{(\tau)} \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} y_t + \lambda_{M_i}\mu_{i-1}^{(\tau)}}{\lambda_i^{(\tau)}\nu_i + \lambda_{M_i}}, \frac{1}{\lambda_i^{(\tau)}\nu_i + \lambda_{M_i}}\right),$$

where a normal prior, $\mathcal{N}(\mu_{i-1}^{(\tau)}, \lambda_{M_i})$, is truncated to the interval, (μ_{i-1}, ∞), and $\lambda_{M_i} = 2M^2/(\lambda_{i-1}^{(\tau)}+\lambda_i^{(\tau)})$ such that M is a positive constant, for any $i \in \{2, 3, \ldots, m\}$.

The scheme of the Gibbs sampler is summarised in the following (Algorithm 11).

Algorithm 11 Gibbs Sampler (Univariate NHGHMM)

```
1: for $i = 1, 2, \ldots, m$ do
2:     Simulate
           $$\lambda_i^{(\tau)} \,|\, \theta_{-\lambda} \sim \mathcal{G}\left(\frac{\nu_i^{(\tau)}}{2} + \alpha_\Lambda, \frac{1}{2}\sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} (y_t - \mu_i^{(\tau-1)})^2 + \beta_\Lambda\right).$$
3: end for
4: Simulate
           $$\mu_1^{(\tau)} \,|\, \theta_{-\mu} \sim \mathcal{N}\left(\frac{\lambda_i^{(\tau)} \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} y_t + \lambda_{M_1}\mu_{M_1}}{\lambda_i^{(\tau)}\nu_i^{(\tau)} + \lambda_{M_1}}, \frac{1}{\lambda_i^{(\tau)}\nu_i^{(\tau)} + \lambda_{M_1}}\right).$$
5: for $i = 2, 3, \ldots, m$ do
6:     Simulate
           $$\mu_i^{(\tau)} \,|\, \theta_{-\mu} \sim \mathcal{N}\left(\frac{\lambda_i^{(\tau)} \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} y_t + \lambda_{M_i}\mu_{i-1}^{(\tau)}}{\lambda_i^{(\tau)}\nu_i + \lambda_{M_i}}, \frac{1}{\lambda_i^{(\tau)}\nu_i + \lambda_{M_i}}\right),$$
       where the prior distribution, $\mathcal{N}(\mu_{i-1}^{(\tau)}, \lambda_{M_i})$, is truncated to the interval, $(\mu_{i-1}, \infty)$,
       and $M$ is some positive constant for $\lambda_{M_i} = 2M^2/(\lambda_{i-1}^{(\tau)} + \lambda_i^{(\tau)})$.
7: end for
```

This concludes a single τ^{th} MCMC iteration for $\tau \in \{2, 3, \ldots, N\}$. The main algorithm iterates this single iteration N times until all the parameters of interest are believed to reach the convergence jointly.

By applying a series of these three proposed MCMC algorithms, I anticipate there should exist some improvements in algorithms' efficiencies and MCMC convergence as a result. My original contribution in this chapter is comprised of incorporating these MCMC algorithms into the main algorithm and constructing a set of R© & C++ code for the algorithms. The simulation was run on an 8-core 3.60 GHz Intel(R) Core(TM) i7-4790 central processor unit with 8 GB of random access memory.

A univariate NHGHMM is now simulated to validate the MCMC algorithms through simulation studies in the next section.

3.6 Simulation Study

The following model was simulated by Diebold et al. (1994) to demonstrate how time-varying transition probabilities are more effective to estimate a sequence of the hidden states than constant transition probability. That simulation study of Diebold et al. (1994)'s did not provide 95% confidence intervals and coverage for the parameters of interest (Diebold et al., 1994). Moreover, the point estimates for one of the two transition probability matrices in the identical model was quite biased. Thus, a series of the proposed MCMC algorithms was implemented for the model. I expect to observe that Bayesian framework is able to estimate the parameters more accurately than that of Diebold et al. (1994)'s by the following reasons:

(i) the framework is of a Bayesian approach, and
(ii) Bayesian inference works better for data sets with small sample size as long as the prior sensitivity is heeded (McNeish, 2016).

There are two aspects regarding the proposed MCMC algorithms being considered:

(a) the MCMC convergence, and
(b) the performance of parameter estimation.

The model is a two-state model (i.e. $m = 2$), and its parameters are as follows:

$$
\begin{gathered}
Y_t \,|\, s_t = i \sim f(y_t \,|\, \theta_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\frac{(y_t - \mu_i)^2}{\sigma^2}\right), \; t \in \{1, 2, \ldots, 100\}, \; i \in \{1, 2\} \\
(\mu_1, \mu_2) = (-1, 1) \\
(\sigma_1, \sigma_2) = (2, 2) \\
\alpha_{1,1} = (0, 0), \; \alpha_{1,2} = (3.529, -5.205) \\
\alpha_{2,1} = (-5.320, 5.205), \; \alpha_{2,2} = (0, 0) \\
z_{t_1} = (1.0, 0.6), \; z_{t_2} = (1.0, 1.1) \\
Q^{t_1} = \begin{pmatrix} 0.4 & 0.6 \\ 0.1 & 0.9 \end{pmatrix}, \; Q^{t_2} = \begin{pmatrix} 0.9 & 0.1 \\ 0.6 & 0.4 \end{pmatrix},
\end{gathered}
\tag{3.24}
$$

where

$$
\begin{gathered}
t_1 \in \{2, 3, \ldots, 20\} \cup \{41, 42, \ldots, 60\} \cup \{81, 82, \ldots, 100\} \\
t_2 \in \{21, 22, \ldots, 40\} \cup \{61, 62, \ldots, 80\}.
\end{gathered}
$$

Figure 3.2 graphically represents a simulated data from Model (3.24) where s_t is a sequence of hidden states, and y_t is a sequence of univariate observed values, for any $t \in \{1, 2, \ldots, 100\}$. This particular simulated data set was chosen since it is a good example to demonstrate Bayesian inference may outperform a frequentist approach. Increasing the sample size of this data set was not attempted in order to maintain the identical parameter set.

In the simulation study, a standard deviation σ is preferred over a precision λ since statistical software R© deals with the former. Note that Diebold et al. (1994) set the parameters α_{11} and α_{22} as non-zero values. Nonetheless, Model (3.24) is able to estimate transition probabilities by following Spezia (2006)'s algorithm.

As for the prior distributions, the framework of Spezia (2006) is followed by setting:

(i) normal priors, $\mathcal{N}(0, 10/3)$, for any μ_i in Equation (3.2),

(ii) gamma priors, $\mathcal{G}(1/2, 1/2)$, for any $\lambda_i = 1/\sigma_i^2$ in Equation (3.3), and

(iii) multivariate Gaussian priors, $\mathcal{N}_{(2)}(\mu_A, \Sigma_A)$, in Equation (3.4) where $\mu_A = \mathbf{0}$ and

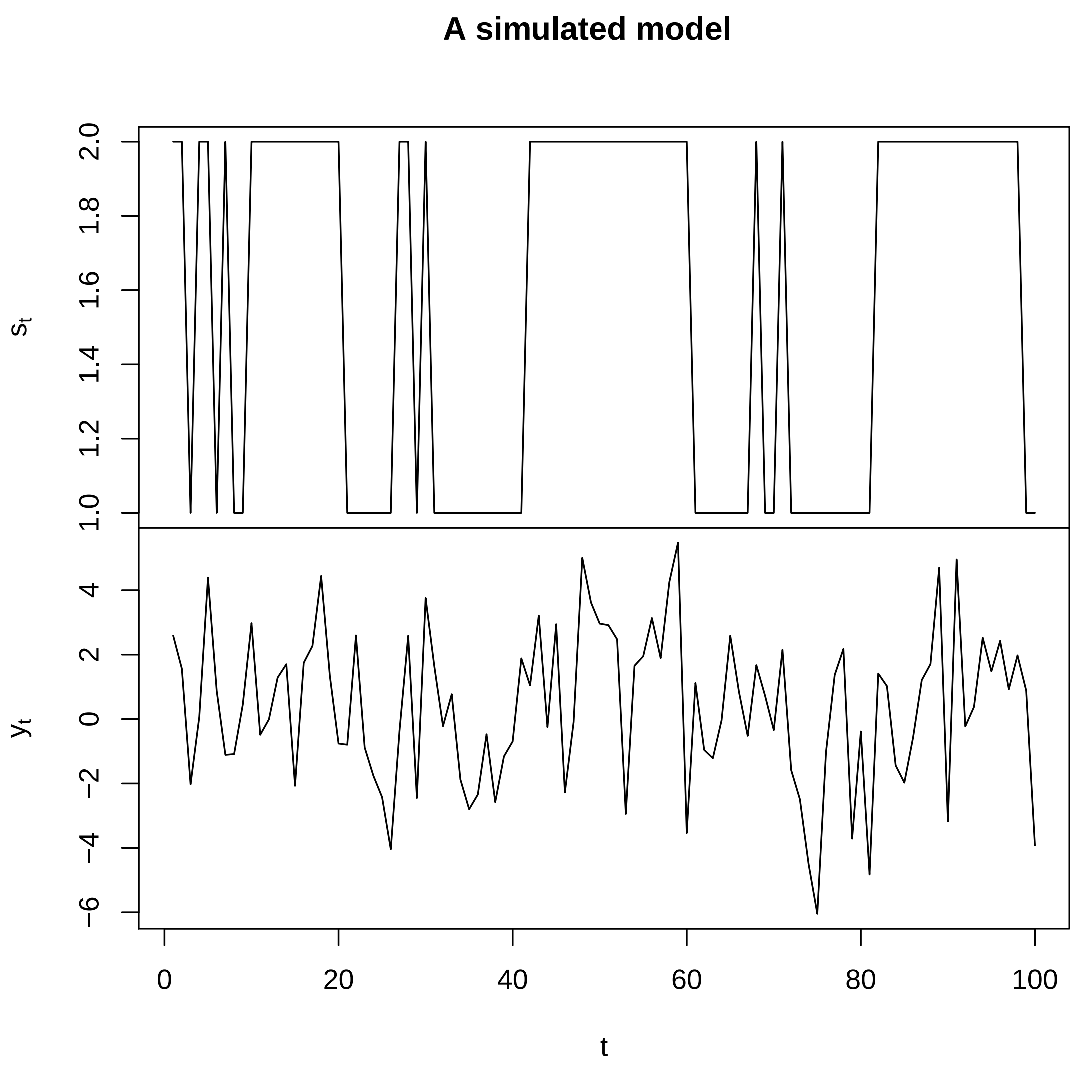


Figure 3.2: Time series plot of Model (3.24): The hidden states s_t (top), and observed values y_t (bottom).

$\Sigma_A^{i,j} = 10$, for any $(i, j) \in \{1, 2\} \times \{1, 2\}$ whenever $i = j$. Otherwise, $\Sigma_A^{i,j} = 0$, whenever $i \neq j$.

As Spezia (2006) stated that the prior distribution for the parameter α is quite vague. Thus, a non-informative prior distribution for that parameter of interest is also used here.

3.6.1 MCMC Convergence

It is a common strategy to reduce autocorrelations of MCMC samples by taking every ν^{th} iteration, where $\nu \in \mathbb{N} \setminus \{1\}$, and this is called *thinning* (Owen, 2017). It is also known to be useful for memory storage management. Note that some authors, however, are against the thinning technique by stating “usually unnecessary and always inefficient” (Link and Eaton, 2012; MacEachern and Berliner, 1994).

Therefore, I investigate the MCMC iterations in two ways, without and with thinning, to observe successive improvements by this autocorrelation reducing technique and the proposed MCMC algorithms.

The MCMC algorithms without thinning

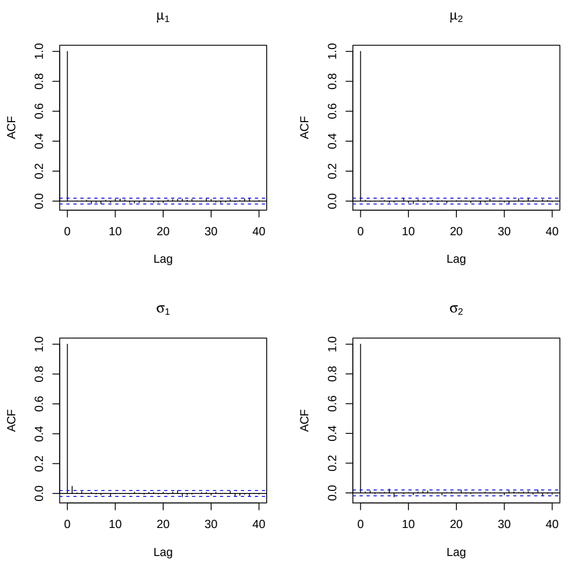
Firstly, I implemented the AM, symmetric DRAM, and MTAM algorithms within the main algorithm (Algorithm 6) without thinning. The total number of MCMC iterations was $N = 1,500,000$ and burn-in time of 1,490,000. Thus, the last 10,000 MCMC iterations for every parameter were taken into account. These particular numbers were chosen to ensure the MCMC convergence since the sample size of the simulated data set is considerably small.

Although MCMC algorithms have gained their enormous popularity in many applications, it is often difficult to claim MCMC samples are true representatives of target distributions of interest. Therefore, it is crucial for a Bayesian analysis to conclude when an MCMC algorithm has reached to the convergence (Cowles and Carlin, 1996). As such, Algorithms 8, 9, and 10 described in Section 3.5 are proposed to ensure the convergence of parameters of interest, especially the parameter $\alpha_{i,j}$, for any $(i, j) \in \{(1, 2), (2, 1)\}$.

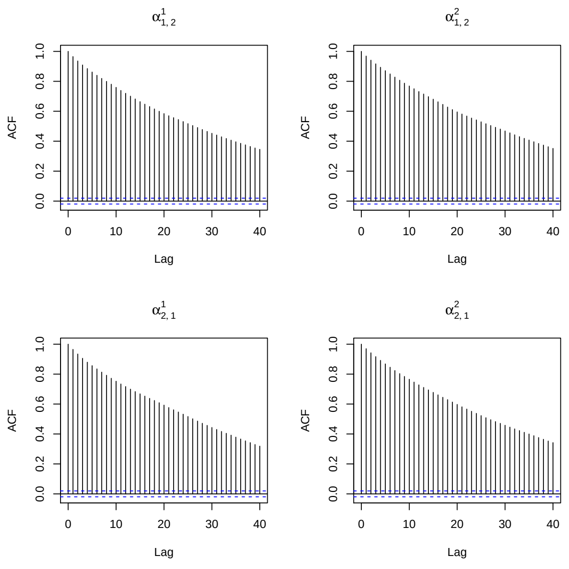
In this simulation study, Geweke’s diagnostics (Section 2.6), ESS and autocorrelation functions are used to monitor the MCMC convergence.

Table 3.1: Summary of Geweke's diagnostics and ESS for the random walk Metropolis-Hastings algorithm without thinning

Parameter	Geweke's diagnostics	ESS
μ_1	-0.288	10 000
μ_2	1.241	10 000
σ_1	-0.648	9 082
σ_2	-0.885	9 594
$\alpha^1_{1,2}$	-0.721	134
$\alpha^2_{1,2}$	0.591	130
$\alpha^1_{2,1}$	1.418	135
$\alpha^2_{2,1}$	-1.289	140



(a)



(b)

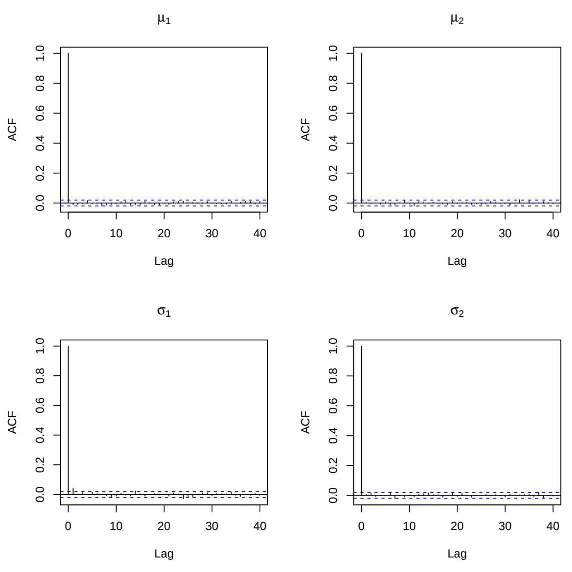
Figure 3.3: Autocorrelation functions of μ & σ (a), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (b), for the random walk Metropolis-Hastings algorithm without thinning

Table 3.1 shows the result of Geweke's diagnostics and ESS for the Metropolis-Hastings algorithm. The ESS of the parameters $\alpha_{1,2}$ and $\alpha_{2,1}$ are significantly low. Moreover, the corresponding autocorrelation functions of the parameters also indicate the dependencies between the MCMC samples in Figure 3.3. Therefore, the convergence for these resulting MCMC samples was not met.

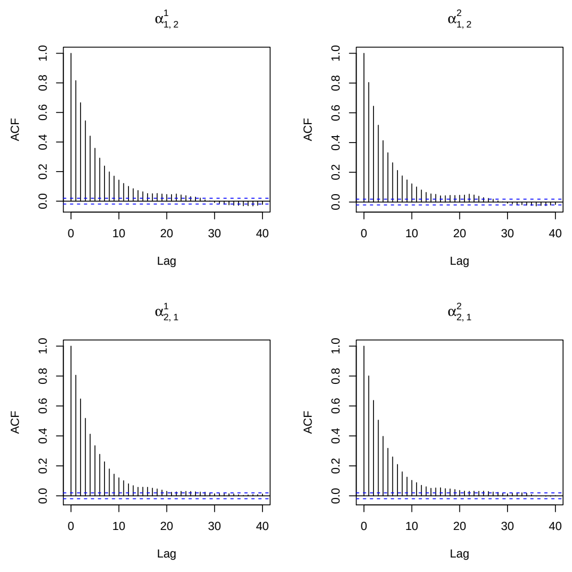
To improve mixing, the proposed MCMC algorithms were implemented on the simulated model.

Table 3.2: Summary of Geweke's diagnostics and ESS for the AM algorithm without thinning

Parameter	Geweke's diagnostics	ESS
μ_1	-0.924	10 000
μ_2	1.141	10 000
σ_1	-1.552	9 226
σ_2	-1.131	10 000
$\alpha^1_{1,2}$	-1.780	1 017
$\alpha^2_{1,2}$	1.578	1 087
$\alpha^1_{2,1}$	0.783	1 077
$\alpha^2_{2,1}$	-0.929	1 037



(a)



(b)

Figure 3.4: Autocorrelation functions of μ & σ (a), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (b), for the AM algorithm without thinning

Table 3.2 represents the result of the AM algorithm, and shows Geweke's diagnostics and ESS for the parameters of interest.

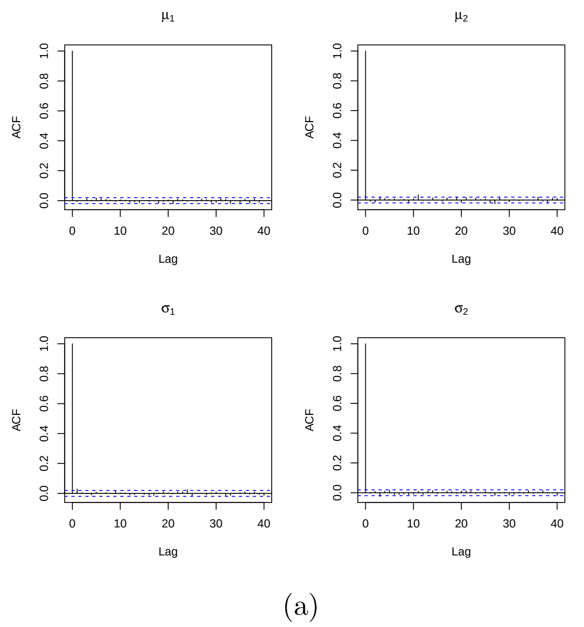
It can be observed that the parameters σ_1 and $\alpha^1_{1,2}$ almost failed Geweke's diagnostics although every parameter passed the test. In general, it is a quite difficult task to confirm the convergence. Therefore, analyses of autocorrelation functions for the parameters of interest are conducted. The ESS of Gibbs samples μ and σ show that those MCMC samples have converged. Having said that, the ESS of the AM algorithm performed very poorly and it was claimed that those MCMC samples are invalid to be used for the parameter estimation.

By Figure 3.4, the result of convergence tests suggests that the MCMC samples have not converged jointly and the convergence is not confirmed yet as the autocorrelation functions of the parameters $\alpha_{1,2}$ and $\alpha_{2,1}$ are fairly correlated. Hence, parameter estimation for these MCMC samples is invalid for the inference.

The next attempted MCMC algorithm is the symmetric DRAM algorithm which is described in Section 2.4.4. The number of stages was set to 10, and after the rejection of the 10th try, the MCMC chain stays in the current draw. Summary statistics of the convergence test for the symmetric DRAM algorithm is shown in Table 3.3.

Table 3.3: Summary of Geweke's diagnostics and ESS for the symmetric DRAM algorithm without thinning

Parameter	Geweke's diagnostics	ESS
μ_1	1.110	10 000
μ_2	0.998	9 276
σ_1	−0.077	9 421
σ_2	−0.930	10 405
$\alpha^1_{1,2}$	0.002	2 253
$\alpha^2_{1,2}$	0.017	2 259
$\alpha^1_{2,1}$	−0.568	2 184
$\alpha^2_{2,1}$	0.766	2 216



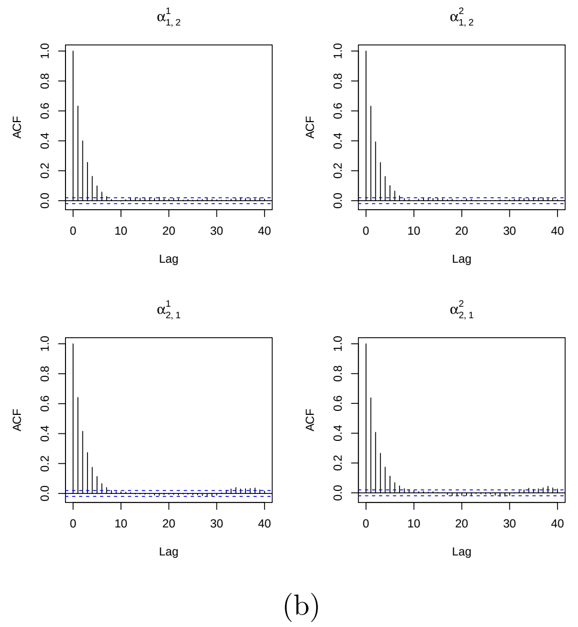


Figure 3.5: Autocorrelation functions of μ & σ (a), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (b), for the symmetric DRAM algorithm without thinning

An improvement of the convergence can be seen clearly compared to the AM algorithm above. The ESS of the parameters $\alpha_{1,2}$ and $\alpha_{2,1}$ in Table 3.3 slightly increased. More importantly, all the parameters of interest have passed Geweke's diagnostics.

In addition, the autocorrelation functions of the parameters $\alpha_{1,2}$ and $\alpha_{2,1}$ especially improved as shown in Figure 3.5. Nevertheless, the ESS may improve more by another scheme of a distinct MCMC algorithm.

Therefore, the MTAM algorithm was implemented, which is described in Section 2.4, to obtain even better results of Geweke's diagnostics and to reduce the dependencies of the samples as much as possible to ensure the convergence. In the multiple-try sampling scheme, 10 tries at each MCMC iteration were attempted. The following table shows the result of the algorithm.

Table 3.4: Summary of Geweke's diagnostics and ESS for the MTAM algorithm without thinning

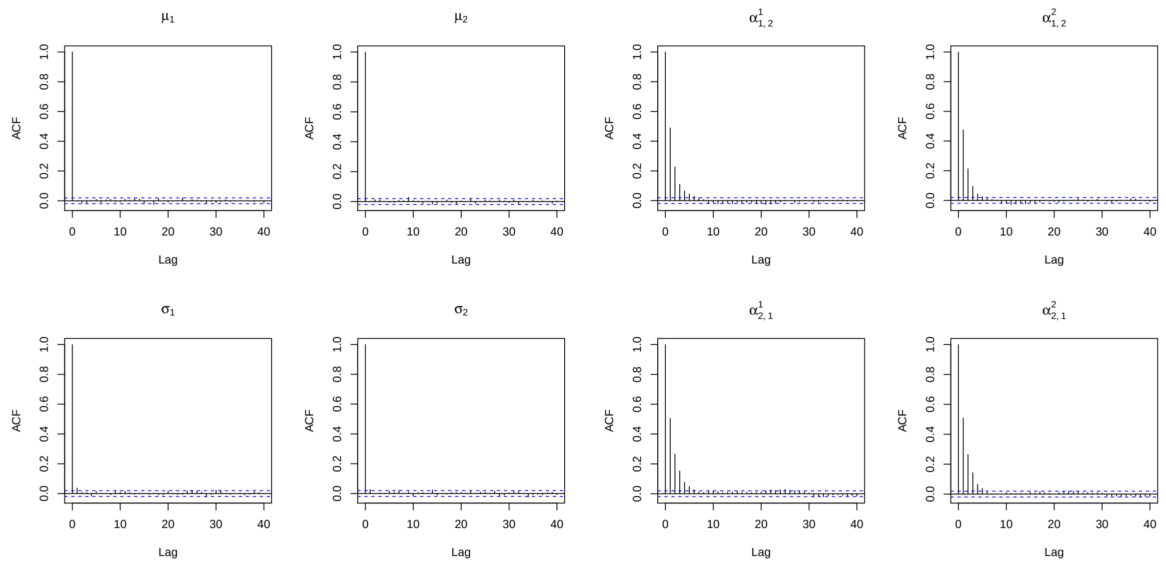
Parameter	Geweke's diagnostics	ESS
μ_1	−0.139	10 000
μ_2	1.771	10 000
σ_1	−0.076	9 281
σ_2	1.015	9 480
$\alpha^1_{1,2}$	−0.162	3 544
$\alpha^2_{1,2}$	−0.438	3 677
$\alpha^1_{2,1}$	1.423	3 086
$\alpha^2_{2,1}$	−0.924	3 257

As observed in Table 3.4, all the parameters have passed Geweke's diagnostics. Moreover, it can be seen an improvement on the ESS of the parameter α. Therefore, it may be claimed that the MTAM algorithm is more effective at improving the ESS than the symmetric DRAM algorithm is.

Figure 3.6 suggests that the dependencies of all the parameters are almost non-existent, although the autocorrelation functions of the parameters $\alpha_{1,2}$ and $\alpha_{2,1}$ still show somewhat dependencies between the MCMC samples.

The MCMC algorithms with thinning

Hereafter, the *thinning* technique is used to reduce the dependencies and enhance ESS as much as possible. Hence, the MCMC samples were simulated by setting burn-in



(a) (b)

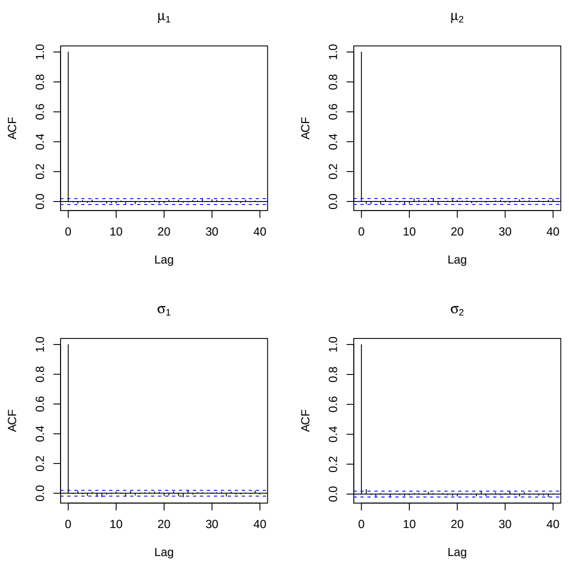
Figure 3.6: Autocorrelation functions of μ & σ (a), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (b), for the MTAM algorithm without thinning

time to be 1,000,000 and then every 50th iteration was thinned, so that the number of MCMC iterations for the inference is consistent with the last 10,000 MCMC iterations in the earlier inference. This scheme was attempted for the random walk Metropolis-Hastings, AM, symmetric DRAM, and MTAM algorithm in order.

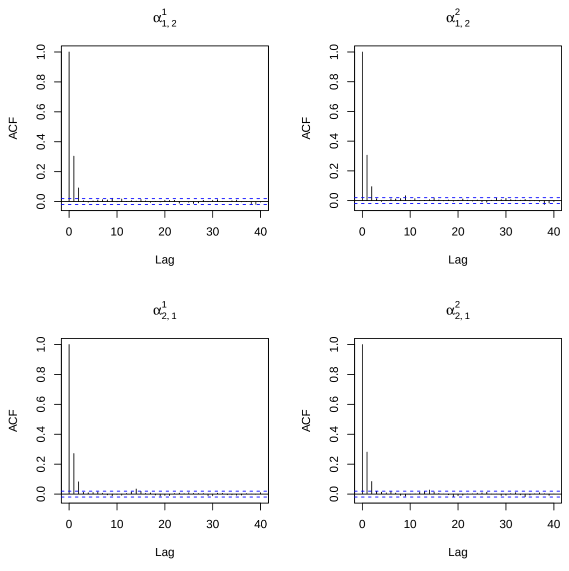
Table 3.5: Summary of Geweke's diagnostics and ESS for the random walk Metropolis-Hastings algorithm with thinning

Parameter	Geweke's diagnostics	ESS
μ_1	−0.837	10 000
μ_2	0.269	10 650
σ_1	−2.461	10 965
σ_2	−0.545	9 390
$\alpha_{1,2}^1$	−0.311	5 614
$\alpha_{1,2}^2$	0.253	4 950
$\alpha_{2,1}^1$	0.180	5 731
$\alpha_{2,1}^2$	−0.509	5 607

Table 3.5 shows that Geweke's diagnostics of the parameter σ_1 indicates a failure for the MCMC convergence. Thus, it is concluded that the algorithm has failed to converge every parameter jointly. In addition, the ESS of the parameters $\alpha_{1,2}$ and $\alpha_{2,1}$ are relatively small. By Figure 3.7, the corresponding autocorrelation functions of those parameters show the existence of dependencies and explain the small ESS.



(a)



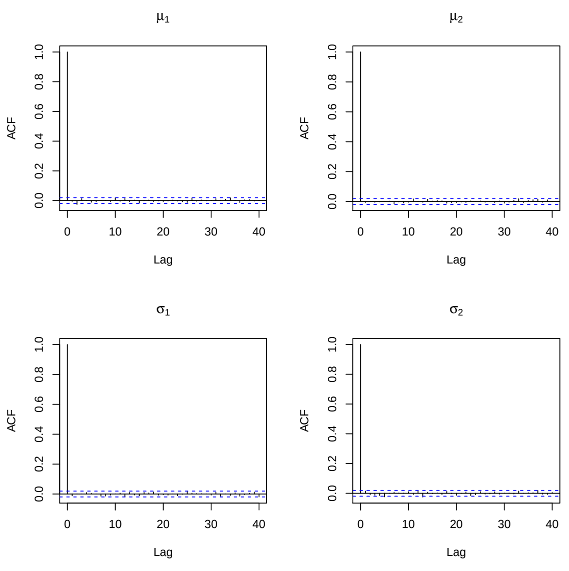
(b)

Figure 3.7: Autocorrelation functions of μ & σ (a), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (b), for the random walk Metropolis-Hastings algorithm with thinning

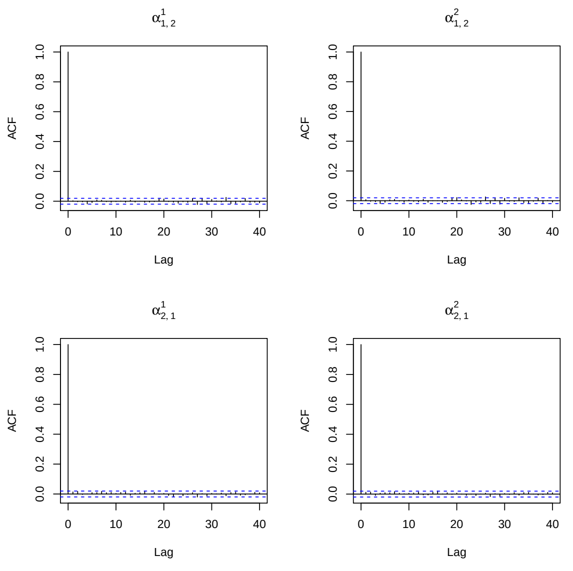
Given that the random walk Metropolis-Hastings algorithm was unsuccessful in the convergence assessment, the AM algorithm is then examined for the test.

Table 3.6: Summary of Geweke's diagnostics and ESS for the AM algorithm with thinning

Parameter	Geweke's diagnostics	ESS
μ_1	0.510	10 635
μ_2	0.002	10 000
σ_1	−1.498	10 000
σ_2	−0.909	10 981
$\alpha^1_{1,2}$	−0.175	10 000
$\alpha^2_{1,2}$	−0.621	10 000
$\alpha^1_{2,1}$	0.144	9 400
$\alpha^2_{2,1}$	−0.235	10 000



(a)



(b)

Figure 3.8: Autocorrelation functions of μ & σ (a), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (b), for the AM algorithm with thinning

Table 3.6 shows all the parameters of interest passed Geweke's diagnostics. In addition, the ESS of every parameter is on a satisfactory level, which is very close to the number, $N = 10,000$.

Furthermore, Figure 3.8 represents that dependencies between each MCMC sample are non-existent. The results for the AM algorithm with thinning showed that the MCMC samples have converged and the statistical inference is acceptable to be used.

Given the result of the AM algorithm with thinning above was successful in terms of every test, the MCMC samples of the symmetric DRAM and MTAM algorithms are also thinned by the same manner. Thus, a comparison between the AM and the aforementioned algorithms is able to be made. The result of the symmetric DRAM algorithm is as follows.

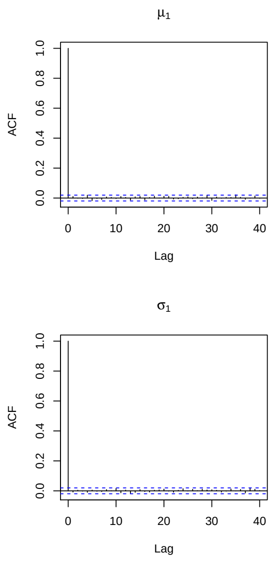
Given Table 3.7, all the parameters passed Geweke's diagnostics and the ESS of every parameter is highly satisfactory.

By Figure 3.9, autocorrelation functions also corroborate the conclusion in which the MCMC samples have converged.

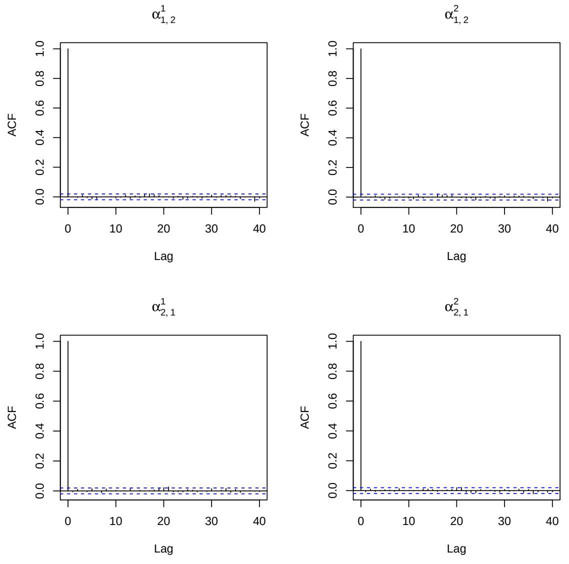
Finally, the assessment of the MCMC convergence for the MTAM algorithm concludes all the tests. The number of the multiple tries remains the same as the one without thinning. The result is presented as follows.

Table 3.7: Summary of Geweke's diagnostics and ESS for the symmetric DRAM algorithm with thinning

Parameter	Geweke's diagnostics	ESS
μ_1	-0.733	10 000
μ_2	-0.988	10 000
σ_1	-1.374	10 000
σ_2	-0.743	9 287
$\alpha_{1,2}^1$	1.006	10 000
$\alpha_{1,2}^2$	-0.289	10 000
$\alpha_{2,1}^1$	0.539	10 000
$\alpha_{2,1}^2$	0.067	10 000



(a)

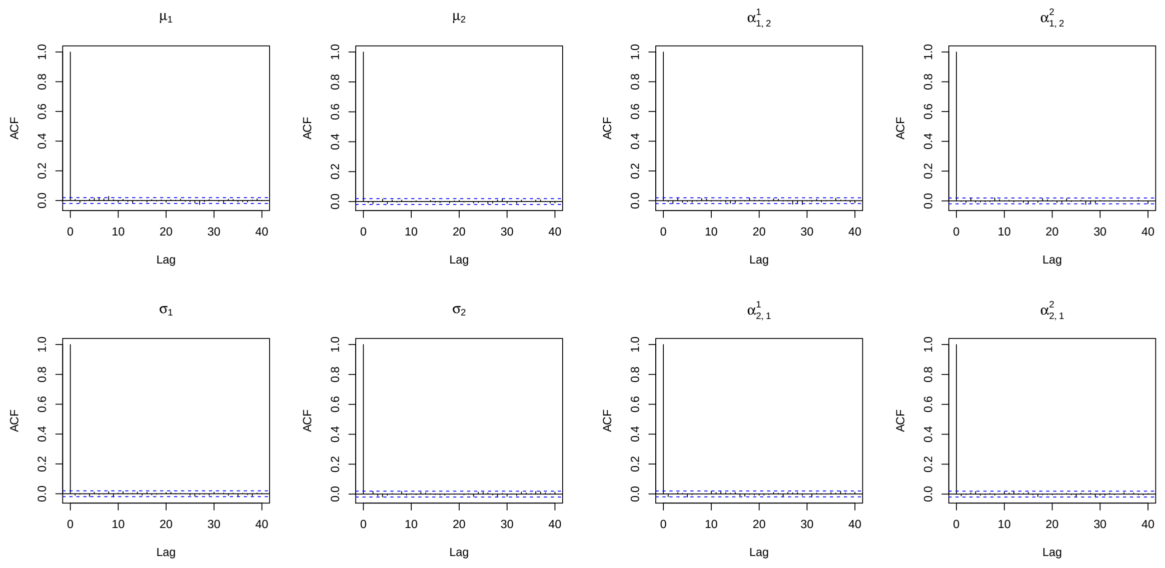


(b)

Figure 3.9: Autocorrelation functions of μ & σ (a), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (b), for the symmetric DRAM algorithm with thinning

Table 3.8: Summary of Geweke's diagnostics and ESS for the MTAM algorithm with thinning

Parameter	Geweke's diagnostics	ESS
μ_1	0.636	8 600
μ_2	0.450	10 000
σ_1	-0.617	10 000
σ_2	-0.599	10 257
$\alpha_{1,2}^1$	0.869	10 173
$\alpha_{1,2}^2$	-0.953	10 000
$\alpha_{2,1}^1$	-0.333	10 369
$\alpha_{2,1}^2$	0.563	10 000



(a) (b)

Figure 3.10: Autocorrelation functions of μ & σ (a), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (b), for the MTAM algorithm with thinning

3.6.2 Performance of Parameter Estimation

As expected, all the convergence tests confirm that the MCMC samples of the parameters have converged since Table 3.8 shows the quite satisfactory values of Geweke's diagnostics and ESS. By Figure 3.10, it is evident that the dependencies between the MCMC samples are non-existent.

Prior to the simulation study, several pilot runs were carried out for model selection of the model by the random walk Metropolis-Hastings, AM, symmetric DRAM, and MTAM algorithms.

The case where the number of hidden states is equal to 1 (i.e. $m = 1$), is omitted since such a model is insensible in the non-homogeneous setting (Spezia, 2006). The case where $m \geq 4$ is also not considered due to the small sample size. Therefore, the number of hidden states where $m \in \{2, 3\}$ is considered. The results of model selection are shown as follows.

Table 3.9: Summary of log marginal likelihoods for the Metropolis-Hastings (MH), AM, symmetric DRAM, and MTAM algorithms

	MH	AM	DRAM	MTAM
$m = 2$	-209.961	-209.991	-209.940	-209.736
$m = 3$	-213.555	-214.042	-214.050	-214.020

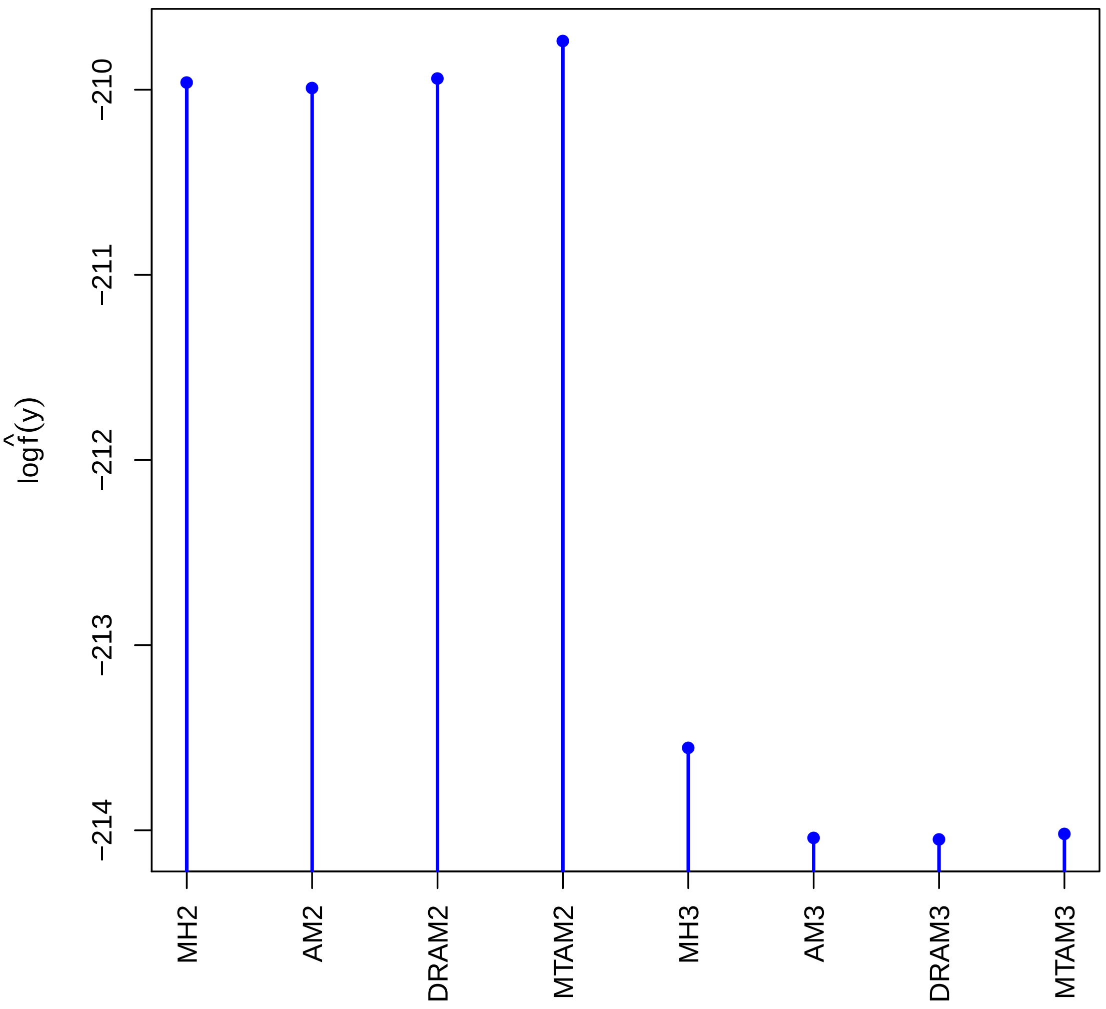


Figure 3.11: Bar plot of log marginal likelihoods for model selection; the suffix number of each algorithm in the plot corresponds to the number of hidden states.

According to Table 3.9, the MTAM algorithm with the number of hidden states with $m = 2$, evaluates the maximum log marginal likelihood amongst all the other competing models. By Figure 3.11, it is conclusive that the model selection has been successful. The result is consistent with the true number of hidden states in the simulated model.

Provided that the successful result of convergence assessment and model selection, the AM, symmetric DRAM, and MTAM algorithms are implemented for the following parameter estimation.

Therefore, I now simulate $R = 100$ replications in order to complete my simulation studies with the AM algorithm, the symmetric DRAM algorithm, and the MTAM algorithm. Then, I compare the results of the three algorithms in terms of the mean squared error (MSE) and the mean absolute error (MAE) of the estimated hidden state sequences and the performances of parameter estimation.

The performance describes the proportion of the replications for which the 95% credible intervals (CrI) of the parameters include the true values. The number of the MCMC iterations was set to $N = 1,500,000$ and the burn-in time of 1,000,000. Then, the MCMC samples of the parameters for all the three proposed algorithms were thinned by every 50th iteration. Table 3.10 shows the summary of parameter estimation for the AM algorithm, the symmetric DRAM algorithm, the MTAM algorithm, respectively. As the standard Metropolis-Hastings algorithm was unable to generate the convergent MCMC chains, the estimates of the algorithm are not presented here.

Overall, the performances of the three algorithms are reasonably reliable; however, some parameters such as $\alpha^2_{1,2}$, $\alpha^1_{2,1}$, $q^{t_1}_{2,1}$, $q^{t_1}_{2,2}$, $q^{t_2}_{1,1}$ and $q^{t_2}_{1,2}$, indicated quite poor coverage. Such results may be caused by the large values for σ and small sample size (i.e. $n = 100$).

In Appendix C, all the trace plots and histograms of the parameters for the AM, symmetric DRAM, and MTAM algorithms are shown in Figures C.1 & C.2, C.3 & C.4, and C.5 & C.6, respectively. Each simulated data set is chosen by the lowest MSE and MAE of the estimated state sequence out of $R = 100$ replications during the course of the simulation study.

By Figures C.1a and C.1b, it was observed for the AM algorithm that the parameters μ_1 and μ_2 were under- and over-estimated, respectively. These biases were driven by the small sample size of the simulated data set. Most importantly, the estimation of the transition parameters is quite reasonable as the central tendencies were very close to their true values. Conversely, the parameters $q^{t_2}_{2,1}$ and $q^{t_2}_{2,2}$ were slightly biased. Nevertheless, all the other parameters reasonably covered the true values.

As for the symmetric DRAM algorithm, the biases of the parameters μ_1 and σ_1 are quite remarkable as observed in Figures C.3a and C.3b. Hence, it might be claimed that the parameter estimation for Gaussian parameters of State 1 was unsuccessful. In addition, by Figures C.4c and C.4d, the MCMC samples of Q^{t_2} were found to be biased. This bias might have been caused by the small number of sample size.

Table 3.10: Summary statistics for each algorithm where $R = 100$ replications were taken into account. A comparison of the three proposed MCMC algorithms includes: point estimates, 95% CrI and coverage of the true values.

Parameter	True value	AM			DRAM			MTAM			Diebold et al.
		Estimate	95% CrI	Coverage	Estimate	95% CrI	Coverage	Estimate	95% CrI	Coverage	Estimate
μ_1	−1	−1.482	(−2.147, −0.821)	54%	−1.516	(−2.164, −0.865)	49%	−1.381	(−2.017, −0.747)	73%	−1.62
μ_2	1	1.284	(0.728, 1.835)	74%	1.314	(0.716, 1.908)	72%	1.340	(0.782, 1.895)	62%	1.27
σ_1	2	1.783	(1.361, 2.344)	76%	1.741	(1.326, 2.291)	80%	1.785	(1.379, 2.318)	74%	1.68
σ_2	2	1.788	(1.434, 2.232)	81%	1.837	(1.459, 2.323)	84%	1.758	(1.403, 2.205)	70%	1.72
$\alpha^1_{1,2}$	3.529	1.684	(−1.423, 4.952)	85%	1.568	(−1.677, 4.987)	85%	1.639	(−1.303, 4.730)	89%	NA
$\alpha^2_{1,2}$	−5.205	−1.978	(−5.573, 1.533)	60%	−1.987	(−5.789, 1.737)	64%	−2.101	(−5.578, 1.295)	61%	NA
$\alpha^1_{2,1}$	−5.320	−2.352	(−5.173, 0.370)	47%	−2.296	(−5.206, 0.512)	50%	−2.360	(−5.073, 0.282)	45%	NA
$\alpha^2_{2,1}$	5.205	2.238	(−1.247, 5.852)	70%	2.114	(−1.496, 5.835)	69%	2.313	(−1.111, 5.857)	72%	NA
$q^{t_1}_{1,1}$	0.4	0.412	(0.185, 0.684)	84%	0.432	(0.200, 0.708)	81%	0.439	(0.214, 0.692)	72%	0.04
$q^{t_1}_{1,2}$	0.6	0.588	(0.316, 0.815)	84%	0.568	(0.292, 0.800)	81%	0.561	(0.308, 0.786)	72%	0.96
$q^{t_1}_{2,1}$	0.1	0.308	(0.169, 0.479)	51%	0.299	(0.153, 0.477)	57%	0.305	(0.164, 0.481)	53%	0.10
$q^{t_1}_{2,2}$	0.9	0.692	(0.521, 0.831)	51%	0.701	(0.523, 0.847)	57%	0.695	(0.519, 0.836)	53%	0.90
$q^{t_2}_{1,1}$	0.9	0.618	(0.399, 0.811)	43%	0.619	(0.407, 0.815)	51%	0.640	(0.445, 0.811)	53%	0.62
$q^{t_2}_{1,2}$	0.1	0.382	(0.189, 0.601)	43%	0.381	(0.185, 0.593)	51%	0.360	(0.189, 0.555)	53%	0.38
$q^{t_2}_{2,1}$	0.6	0.514	(0.235, 0.769)	78%	0.491	(0.201, 0.766)	78%	0.524	(0.239, 0.802)	83%	0.62
$q^{t_2}_{2,2}$	0.4	0.486	(0.231, 0.765)	78%	0.509	(0.234, 0.799)	78%	0.476	(0.198, 0.761)	83%	0.38

Finally, coverage of the parameters μ, σ, and α for the MTAM algorithm was superior to the other two MCMC algorithms as seen in Figure C.5. Although the central tendencies of the transition probabilities $q^{t_1}_{2,1}$, $q^{t_1}_{2,2}$, $q^{t_2}_{1,1}$, and $q^{t_2}_{1,2}$ were quite accurate to their true values, those of the other transition probabilities were slightly biased by Figure C.6.

Overall, the simulation study in this thesis generated the more meaningful result than that of Diebold et al. (1994)'s given the same sample size (i.e. $n = 100$).

Table 3.11: Summary of the lowest bounds of MSE and MAE for the AM, symmetric DRAM, and MTAM algorithms along with Diebold et al.'s model

	AM	DRAM	MTAM	Diebold et al.
MSE	0.09	0.08	0.10	0.11
MAE	0.18	0.14	0.20	NA

Table 3.11 shows the lowest MSE and MAE for each of the particular simulated data sets. According to the resultant, all of the proposed algorithms in this thesis generated lower MSE than that of Diebold et al. (1994)'s. The corresponding estimated state and the actual sequences are shown in Figures 3.12d, 3.12e, and 3.12f for the AM, symmetric DRAM, and MTAM algorithms, respectively.

Furthermore, by exploiting a set of exogenous variables, it was observed that the estimation of the hidden state sequences in NHGHMMs proved more effective than homogeneous GHMMs in Figure 3.12. As Diebold et al. (1994) pointed out, NHGHMMs were able to relax the restrictive constant transition probabilities in my simulation study.

Also, a comparison of the proposed MCMC algorithms was carried out to examine each algorithm's computational cost by using 'rbenchmark' package in R©. A test model was chosen to be Model (3.24).

For each algorithm, the number of MCMC iterations was set to be $N = 500$. The symmetric DRAM and MTAM algorithms were specifically the focus of this comparison with regard to their number of tries and the corresponding computational expenses. The MCMC run was repeated 100 times for each algorithm. The comparison is detailed in the following table.

Table 3.12: Comparison of the proposed MCMC algorithms for the univariate case

Algorithm	No. of tries	Replications	Elapsed time (s)	Relative
AM	1	100	80.94	1.00
DRAM	10	100	267.03	3.30
DRAM	25	100	552.86	6.83
DRAM	50	100	1 019.75	12.60
MTAM	10	100	220.75	2.73
MTAM	25	100	348.64	4.31
MTAM	50	100	591.97	7.31

As the AM algorithm being a reference algorithm, computational costs of the symmetric DRAM and MTAM algorithms will increase linearly with respect to the number of tries for each MCMC iteration. More importantly, the symmetric DRAM algorithm was found to be a more computationally expensive algorithm than the MTAM algorithm whilst the latter has been the most efficient algorithm in terms of autocorrelation functions and ESS.

Last but not least, the computational time for the simulation study of each algorithm was 2.75 days for the AM algorithm, 8.20 days for the symmetric DRAM algorithm, and 7.16 days for the MTAM algorithm.

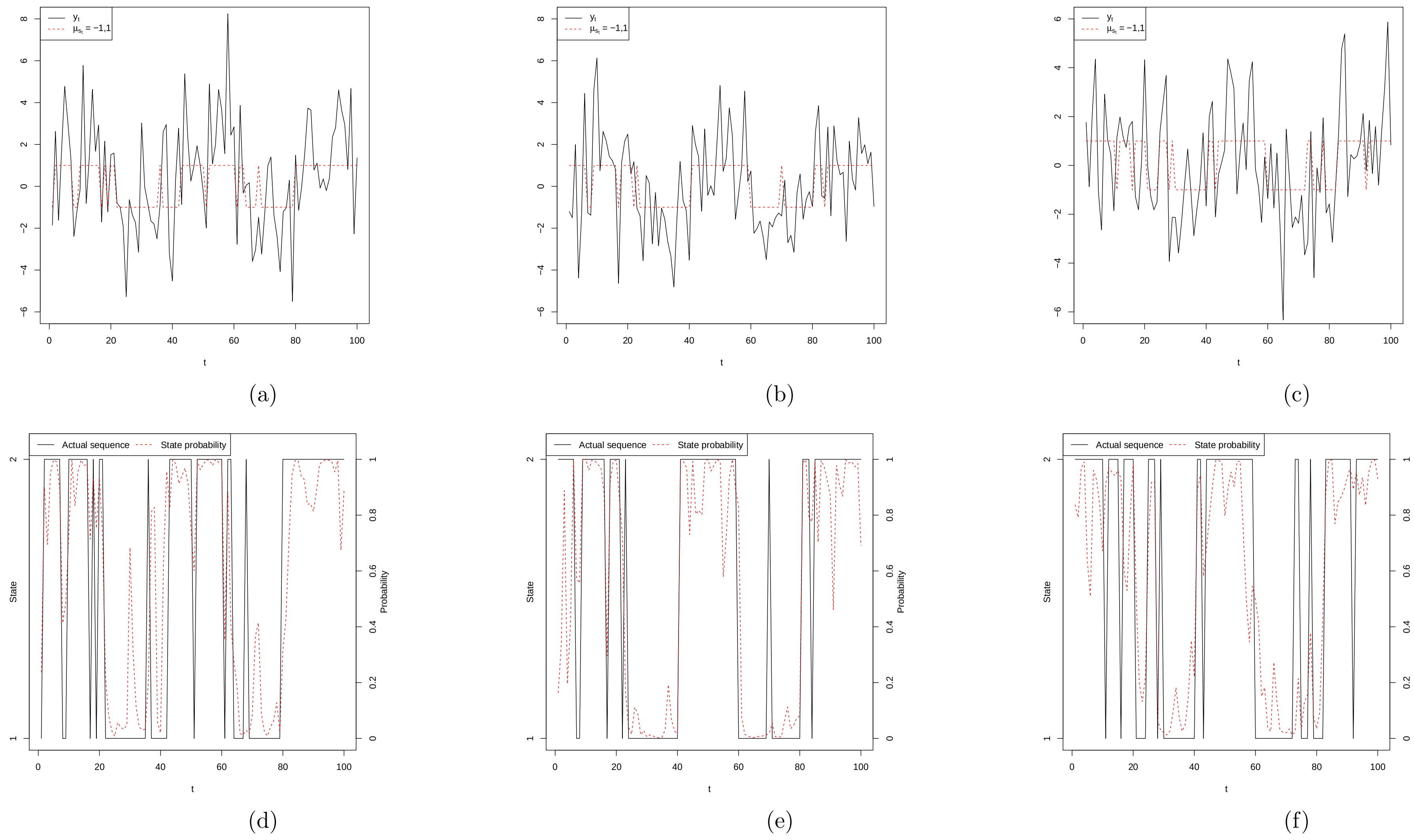


Figure 3.12: Time series plots of the observations (top row) and the estimated hidden state sequences (bottom row) for the AM algorithm, (a) & (d); the symmetric DRAM algorithm, (b) & (e); and the MTAM algorithm, (c) & (f).

3.7 Case Study

Having completed the simulation studies in the previous section, a case study of the univariate NHGHMM from the following real-world financial data set is now presented in this section. The data set analysed in this case study can be obtained from Quandl (https://www.quandl.com/) by using 'Quandl' package in R©.

3.7.1 The US Treasury Bill Rates

Economic and financial time series modelling through HMMs has been proposed by Hamilton (1994). One of the crucial applications of the HMMs is forecasting, but a homogeneous HMM lacks the capability of predicting hidden state sequence as mentioned in the earlier sections.

Considering the limitation of the homogeneous HMM's predictive ability, Meligkotsidou and Dellaportas (2011) applied an NHGHMM to the data set of the monthly US 3-month treasury bill rates from January 1962 to December 1999. In addition to the time-varying transition probabilities, Bayesian inference was taken into account for parameter estimation as Filardo and Gordon (1998) suggested the methodology to avoid the technical problems pertaining to a frequentist approach (Meligkotsidou and Dellaportas, 2011).

As for the data set of the monthly US 3-month treasury bill rates, Dellaportas et al. (2007) utilised regime switching models with the reversible jump MCMC algorithm and identified different regimes in the data set. Moreover, they introduced six financial variables to describe the episodes of mean-reverting tendencies which appear in interest rate levels (Meligkotsidou and Dellaportas, 2011).

Following Meligkotsidou and Dellaportas (2011), a two-state NHGHMM was proposed in the case study and the six financial variables were used as exogenous variables within the model. The financial variables include:

z_t^1 : the US annual inflation rate (INF),

z_t^2 : the trade-weighted US dollar index against other major currencies (CUR),

z_t^3 : the US producer price index (PPI),

z_t^4 : the national association of purchasing management index (NAPM),

z_t^5 : the US consumer price index (CPI), and

z_t^6 : the 10-year US treasury yield (LTR).

All the exogenous variables $z_t = (1, z_t^1, z_t^2, z_t^3, z_t^4, z_t^5, z_t^6)$, for any $t \in \{2, 3, \ldots, 456\}$, were differenced before the statistical analysis except for LTR (Meligkotsidou and Dellaportas, 2011). Hence, I compute the first difference of those exogenous variables and leave the economic variable, LTR, intact. The variable, CUR, only consists of data points ranging from January 1973 to December 1999. A two-state GHMM is a very common model in such a type of time series. It was found that an NHGHMM with the six exogenous variables on the data set of a monthly interest rates return possesses the better predictive ability compared to a homogeneous GHMM (Meligkotsidou and Dellaportas, 2011).

I now consider the identical data set, but a slightly different model from that of Meligkotsidou and Dellaportas (2011)'s. In addition, the model does not take the forecasting ability of the NHGHMM into consideration since it is simply beyond the scope of the research aims in this thesis. Thus, the observed data from January 1973 to December 1999 were used whereas Meligkotsidou and Dellaportas (2011) applied their NHGHMM to the truncated observed data up to December 1997 for generating the fitted observations. The parameters of interest in this case study are:

- μ_i, for all $i \in \{1, 2\}$,
- σ_i, for all $i \in \{1, 2\}$,
- $\alpha_{i,j} = (\alpha_{i,j}^1, \alpha_{i,j}^2, \alpha_{i,j}^3, \alpha_{i,j}^4, \alpha_{i,j}^5, \alpha_{i,j}^6, \alpha_{i,j}^7)$, for $(i, j) \in \{(1, 2), (2, 1)\}$,
- $Q_{i,j}^t$, for all $(i, j) \in \{1, 2\} \times \{1, 2\}$ and $t \in \{2, 3, \ldots, 456\}$, namely from January 1962 to December 1999, and
- S_t, where $S_t \in \{1, 2\}$, for any $t \in \{1, 2, \ldots, 456\}$, namely the estimated hidden state sequence.

In this model, the sequence of the observations in the monthly US 3-month treasury bill rates, y_t, was differenced such that

$$\Delta y_t = \mu_{s_t} + \varepsilon_t,\ \varepsilon_t \sim \mathcal{N}(0, \sigma_{s_t}^2), \tag{3.25}$$

where $\Delta y_t = y_t - y_{t-1}$, for any $t \in \{1, 2, \dots, 456\}$ and y_0 denotes the observation in December 1961. Note that it is equivalent to state as follows:

$$\Delta y_t \sim \mathcal{N}(\mu_{s_t}, \sigma^2_{s_t}), \text{ for any } t \in \{1, 2, \dots, 456\}.$$

The MCMC method includes the Gibbs sampler and the MTAM algorithm. The total number of the MCMC iterations was 5,000,000, and the burn-in time was set to be 4,000,000. After the burn-in time, the thinning of every 100th iteration was taken into account so that the last 10,000 iterations are the parameters of interest.

The parameter $\alpha_{i,j}$, for any $(i, j) \in \{(1, 2), (2, 1)\}$, is distributed in the 7-dimensional Gaussian distribution. Therefore, following the finding of Heaps et al. (2015), the thinning of every 100th iteration was carried out to ensure that every parameter of interest converge jointly, instead of every 10th iteration in the simulation study.

Firstly, it is crucial to assess convergence of the parameters. Table 3.13 shows Geweke's diagnostics, ESS, and relative numerical efficiency of each parameter.

Table 3.13: Summary for Geweke's diagnostics and ESS of each parameter

Parameter	Geweke's diagnostics	ESS
μ_1	−0.851	10 000
μ_2	1.282	10 609
σ_1	0.074	9 068
σ_2	−0.344	10 000
$\alpha^1_{1,2}$	−0.213	10 460
$\alpha^2_{1,2}$	−0.786	10 000
$\alpha^3_{1,2}$	0.194	10 000
$\alpha^4_{1,2}$	0.685	10 972
$\alpha^5_{1,2}$	0.090	9 621
$\alpha^6_{1,2}$	0.602	10 000
$\alpha^7_{1,2}$	0.483	10 476
$\alpha^1_{2,1}$	−0.121	10 712
$\alpha^2_{2,1}$	−1.087	10 439
$\alpha^3_{2,1}$	0.274	10 000
$\alpha^4_{2,1}$	−0.465	10 000
$\alpha^5_{2,1}$	0.469	10 000
$\alpha^6_{2,1}$	−0.724	10 000
$\alpha^7_{2,1}$	0.383	10 709

All the parameters of interest passed Geweke's diagnostics. Furthermore, the ESS of each parameter is at a satisfactory level, and is showing no correlation between the MCMC samples. The case study found the identifiability of σ such that $\sigma_1 < \sigma_2$, and thus, the other parameters are relabelled as such. That being said, no label switching phenomenon was detected. Likewise, Meligkotsidou and Dellaportas (2011) observed no label switching amongst the parameters of interest.

Figure 3.15 shows the differenced US 3-month bill rates and the grey shaded areas. The shaded areas indicate the hidden state sequence is in the state where the observations with large fluctuations. The case study identified the duration between November 1970 & August 1971, June 1973 & February 1975, September 1979 & November 1982, and September 1984 & June 1985 as the period in which the observations with the large fluctuations. The estimates and 95% CrI of the parameters for this case study are given by Table 3.14.

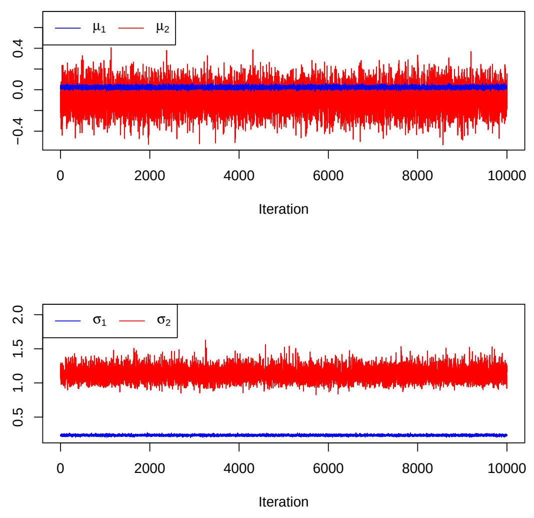
Table 3.14: Parameter estimation of the MCMC iterations for the case study of the US 3-month treasury bill rates

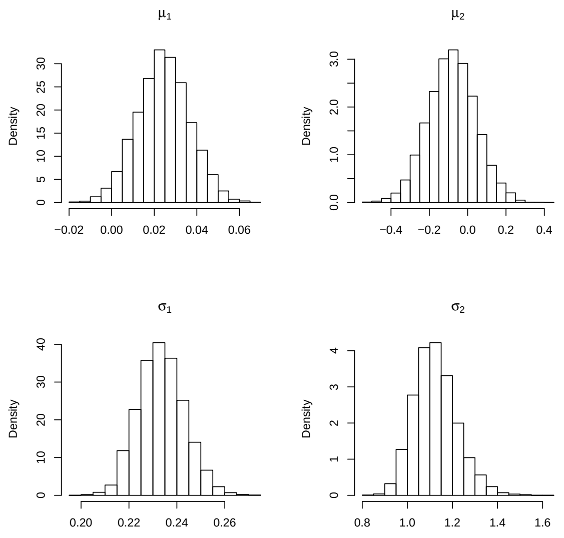
Parameter	Estimate	95% CrI
μ_1	0.024	(0.000, 0.048)
μ_2	−0.081	(−0.326, 0.165)
σ_1	0.233	(0.216, 0.253)
σ_2	1.124	(0.958, 1.332)
$\alpha_{1,2}^1$	−5.637	(−9.229, −2.301)
$\alpha_{1,2}^2$	1.490	(−2.401, 5.421)
$\alpha_{1,2}^3$	0.247	(−0.580, 1.238)
$\alpha_{1,2}^4$	0.774	(−3.991, 5.489)
$\alpha_{1,2}^5$	−0.192	(−0.732, 0.349)
$\alpha_{1,2}^6$	−0.230	(−6.469, 6.034)
$\alpha_{1,2}^7$	0.035	(−0.471, 0.489)
$\alpha_{2,1}^1$	0.689	(−4.355, 6.161)
$\alpha_{2,1}^2$	−4.247	(−8.820, −0.248)
$\alpha_{2,1}^3$	−1.392	(−2.818, −0.258)
$\alpha_{2,1}^4$	−2.895	(−8.628, 2.468)
$\alpha_{2,1}^5$	0.132	(−0.308, 0.650)
$\alpha_{2,1}^6$	−2.228	(−8.506, 3.944)
$\alpha_{2,1}^7$	−0.778	(−1.741, −0.064)

The results differ from that of Meligkotsidou and Dellaportas (2011)'s since their model is essentially different from the model being proposed in this thesis. Nevertheless, the model was able to estimate similar transition probabilities to Meligkotsidou and

Dellaportas (2011)'s estimation by Figure 3.14.

The statistical summary of MCMC iterations are shown in Figure 3.14, and the trace plots and histograms of μ and σ are shown in Figure 3.13.





(a) (b)

Figure 3.13: Trace plots (a) and histograms (b) of μ & σ

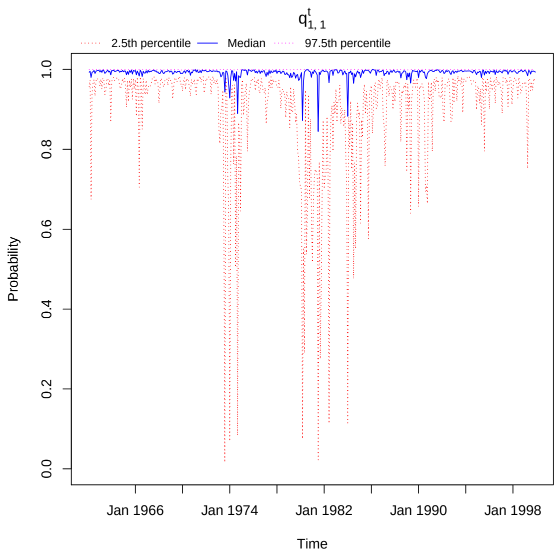
The model is a non-homogeneous GHMM, and hence, there are $2 \times 2 \times (456 - 1) = 1820$ transition probabilities to be estimated. Figure 3.14 summarises the 95% CrI of each transition probability at time $t \in \{2, 3, \ldots, 456\}$. A combination of the parameter α and the exogenous variable z defines the transition probabilities, and its acceptance rates of MCMC iterations for α_{12} and α_{21} were approximately 47.63% and 49.66%, respectively.

The assessment of convergence was a particularly difficult task since MCMC samples in the MTAM algorithm were in the 7-dimensional space. It was necessary to increase the size of tries in the MCMC algorithm or other techniques to accelerate the convergence. This will be part of my future work.

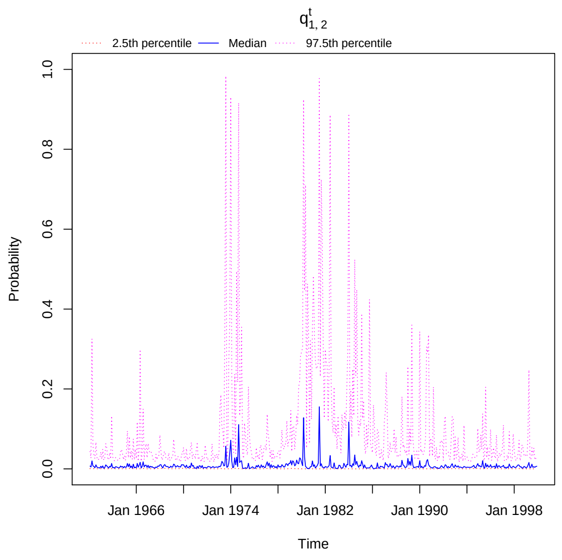
In this case study, I used Model (3.25) which is different from Meligkotsidou and Dellaportas (2011)'s. That fitted model of Meligkotsidou and Dellaportas (2011)'s is given as follows:

$$\Delta y_t = \mu_{s_t} + b_{s_t} y_{t-1} + \varepsilon_t, \;\; \varepsilon_t \sim \mathcal{N}(0, \sigma_{s_t}^2),$$

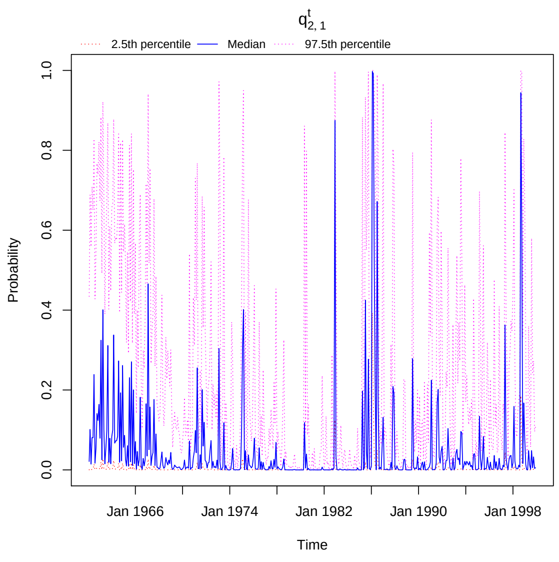
where b_{s_t} is the state specific parameter of the autoregressive model of order 1.



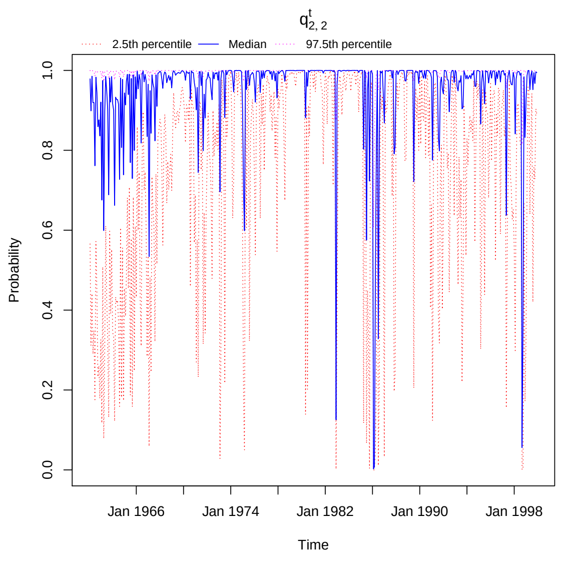
(a)



(b)



(c)



(d)

Figure 3.14: Time series plots of medians and 95% CrI for the transition matrices, $q^t_{1,1}$ (a), $q^t_{1,2}$ (b), $q^t_{2,1}$ (c), and $q^t_{2,2}$ (d), for each $t \in \{2, 3, \ldots, 456\}$

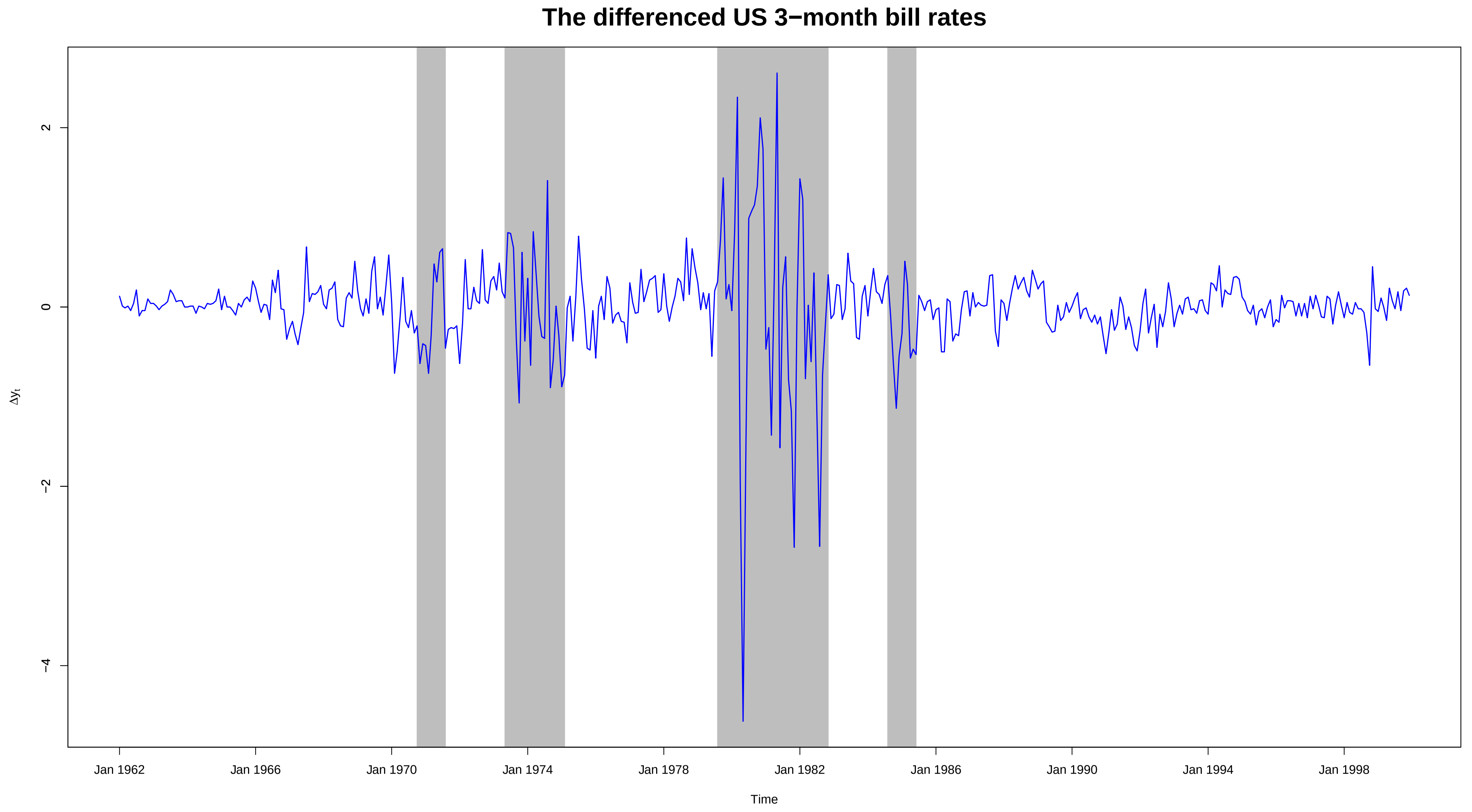


Figure 3.15: Time series plot of the differenced US 3-month treasury bill rates. The grey shaded areas represent the time duration for which the hidden states are in State 2, namely the state with large fluctuations.

Chapter 4

Multivariate NHGHMM

PRIOR to discussing a multivariate NHGHMM, it is necessary to mention the first multivariate non-homogeneous hidden Markov model. The model gives rise to Hughes et al. (1997, 1999)'s work where it was fitted to broad scale atmospheric circulation patterns, namely a downscaling problem. The model dealt with precipitation occurrences, which were not exactly Gaussian random variables. Nevertheless, they established the stochastic process of hidden states being non-homogeneous, and consequently relaxed the restrictive attribute of constant transition probabilities.

Ailliot et al. (2009) fitted an NHGHMM to daily rainfall data from a small network of stations in New Zealand. They successfully modelled regional weather types that govern the temporal dependence of the rainfall, although the parameter estimation was carried out by a frequentist approach.

As such, a collection of literature about *Bayesian analyses of multivariate non-homogeneous Gaussian* HMMs is indeed scarce. With that being said, Heaps et al. (2015) studied rainfall data in the UK using an NHGHMM through a Bayesian approach. They found that Lamb weather types in climatology had a large impact on the atmospheric status, and thus, the NHGHMM was an appropriate stochastic model for spatiotemporal analysis of the observed rainfall data (Heaps et al., 2015).

This section is largely motivated by the scarcity of detailed literature about Bayesian analyses of multivariate NHGHMMs. Hence, the aim in this chapter is to develop various MCMC algorithms within a Bayesian framework which is extended from the univariate setting in Chapter 3.

This chapter is organised as follows. In Section 4.1, the multivariate NHGHMM is mathematically defined, which completely differs from the univariate model in Chapter 3. In Section 4.2, prior distributions for the parameters of interest are specified in order to conduct a Bayesian analysis. Then, a joint likelihood function for the multivariate NHGHMM is defined in Section 4.3. In Section 4.4, a joint posterior distribution is defined, and subsequently, conditional posterior distributions for all the parameters of interest are defined. In Section 4.5, I propose the MCMC algorithms, the AM, the symmetric DRAM, and the MTAM algorithms for the multivariate NHGHMM. This section also contributes to highlighting my original contributions to Bayesian analyses of multivariate NHGHMMs. Finally, the results for a simulation study and case studies are presented in Sections 4.6 and 4.7, respectively.

4.1 The Model

This section is dedicated to introducing a multivariate NHGHMM whose observations are a sequence of d-dimensional continuous random variables, $\mathbf{Y}_t = (Y_t^1, Y_t^2, \ldots, Y_t^d) \in \mathbb{R}^d$, where $d \in \mathbb{N} \setminus \{1\}$ with discrete-time $t \in \{1, 2, \ldots, n\}$ (Spezia, 2010). The observations are assumed to be Gaussian distributed with a vector of state specific means $\boldsymbol{\mu}_i \in \mathbb{R}^d$ and a state specific positive-definite covariance matrix $\Sigma_i \in \mathbb{R}^{d \times d}$, for any $i \in \Omega = \{1, 2, \ldots, m\}$. Hence, the observations are conditionally independent of other observed variables on a sequence of hidden states $S = (S_1, S_2, \ldots, S_n)$ such that

$$\mathbf{Y}_t \mid S_t = i \sim \mathcal{N}_d(\boldsymbol{\mu}_i, \Sigma_i), \tag{4.1}$$

where $S_t \in \Omega$, for any $t \in \{1, 2, \ldots, n\}$. Following Spezia (2006), the initial distribution of the hidden state is defined as $\rho = (\rho_1, \rho_2, \ldots, \rho_m)$, where $\rho_i = \Pr(S_1 = i)$ for any $i \in \Omega$. Then, the transition probabilities of $S_{t-1} = i$ to $S_t = j$, where $i, j \in \Omega$, are defined as follows:

$$\Pr(S_t = j \mid S_{t-1} = i) := q_{i,j}^t = \frac{\exp(z_t^\intercal \alpha_{i,j})}{1 + \sum\limits_{\substack{j \in \Omega \\ j \neq i}} \exp(z_t^\intercal \alpha_{i,j})}, \tag{4.2}$$

where $z_t = (1, z_t^1, z_t^2, \ldots, z_t^{K-1})$ is a vector of K deterministic exogenous variables for any $t \in \{2, 3, \ldots, n\}$, and the multivariate Gaussian distributed random variable $\alpha_{i,j} = (\alpha_{i,j}^1, \alpha_{i,j}^2, \ldots, \alpha_{i,j}^K) \in \mathbb{R}^K$, for any $\{(i,j) \in \Omega^2 : i \neq j\}$ (Spezia, 2006).

Let the parameters of interest be a vector of the parameters for the multivariate Gaussian distributions $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the parameter α for Equation (4.2), and a sequence of the estimated hidden states $s = (s_1, s_2, \ldots, s_n)$ such that

$$\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, s),$$

where $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_m)$, $\boldsymbol{\Sigma} = (\Sigma_1, \Sigma_2, \ldots, \Sigma_m)$, and $\alpha = (\alpha_{i,j})$, for any $\{(i,j) \in \Omega^2 : i \neq j\}$.

4.2 Prior Distributions

Bayesian inference for the multivariate NHGHMM requires the important step to specify prior distributions for the parameters of interest. A set of those parameters is comprised of $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, and α. Following Richardson and Green (1997)'s advice, non-informative prior density functions on the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ were chosen to allow the priors to vary on the interval of the data (Spezia, 2010). A prior density function on the parameter α is also chosen to be non-informative by following Spezia (2006). Therefore, the prior specification is as follows.

(i) *Prior for* $\boldsymbol{\mu}$: A vector of m means which are d-dimensional real-valued vectors $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_m)$. The multivariate Gaussian distribution is defined as the prior for $\boldsymbol{\mu}_i$, for any $i \in \{1, 2, \ldots, m\}$, such that

$$p(\boldsymbol{\mu}) = \prod_{i=1}^{m} p(\boldsymbol{\mu}_i), \tag{4.3}$$

where $\boldsymbol{\mu}_i \sim \mathcal{N}_d(\boldsymbol{\varphi}, \Psi)$, and $\boldsymbol{\varphi}$ is a vector of m midpoints for each sequence of univariate observations y_t^j, $j \in \{1, 2, \ldots, d\}$. The positive-definite covariance matrix

Ψ is defined as follows (Spezia, 2010):

$$\Psi = \begin{pmatrix} \langle y_t^1 \rangle & \frac{\delta_{1,2}}{2}\sqrt{\langle y_t^1 \rangle \langle y_t^2 \rangle} & \cdots & \frac{\delta_{1,d}}{2}\sqrt{\langle y_t^1 \rangle \langle y_t^d \rangle} \\ \frac{\delta_{2,1}}{2}\sqrt{\langle y_t^2 \rangle \langle y_t^1 \rangle} & \langle y_t^2 \rangle & \cdots & \frac{\delta_{2,d}}{2}\sqrt{\langle y_t^2 \rangle \langle y_t^d \rangle} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\delta_{d,1}}{2}\sqrt{\langle y_t^d \rangle \langle y_t^1 \rangle} & \frac{\delta_{d,2}}{2}\sqrt{\langle y_t^d \rangle \langle y_t^2 \rangle} & \cdots & \langle y_t^d \rangle \end{pmatrix}, \tag{4.4}$$

where $\langle y_t^j \rangle = \max\{y_t^j\} - \min\{y_t^j\}$ denotes the range of the observed data y_t^j, for any $j \in \{1, 2, \ldots, d\}$, and $\delta_{j,k}$ denotes the sign function such that

$$\delta_{j,k} = \begin{cases} 1, & \text{when } y_t^j \text{ and } y_t^k \text{ are positively correlated} \\ -1, & \text{when } y_t^j \text{ and } y_t^k \text{ are negatively correlated} \end{cases}, \tag{4.5}$$

for any $\{(j,k) \in \{1, 2, \ldots, d\}^2 : j \neq k\}$. Hence,

$$\begin{aligned} p(\boldsymbol{\mu}_i) &= [(2\pi)^d \det(\Psi)]^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\varphi})^\intercal \Psi^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\varphi})\right) \\ &= \det(2\pi\Psi)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\varphi})^\intercal \Psi^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\varphi})\right), \end{aligned}$$

for any $i \in \{1, 2, \ldots, m\}$.

(ii) *Prior for* $\boldsymbol{\Sigma}$: A vector of m positive-definite covariance matrices which are $(d \times d)$-dimensional $\boldsymbol{\Sigma} = (\Sigma_1, \Sigma_2, \ldots, \Sigma_m)$. The inverse Wishart distribution is defined as the prior for Σ_i, for any $i \in \{1, 2, \ldots, m\}$, such that

$$p(\boldsymbol{\Sigma}) = \prod_{i=1}^{m} p(\Sigma_i), \tag{4.6}$$

where $p(\Sigma_i) = \mathcal{W}^{-1}(\gamma, \Phi)$, $\gamma = \text{int}\{(d+1)/2\} + 1$ in which $\text{int}\{\cdot\}$ denotes the integer part of an arbitrary argument. Furthermore, $\Phi = (\phi_{j,k})$ is a $(d \times d)$-dimensional positive-definite matrix such that

$$\phi_{j,k} = \begin{cases} \gamma, & \text{if } j = k \\ \delta_{j,k}\dfrac{\gamma}{2}, & \text{if } j \neq k \end{cases}, \tag{4.7}$$

for which $\delta_{j,k}$ is defined as in Equation (4.5). Thus,

$$\Sigma_i \sim \mathcal{W}^{-1}(\gamma, \Phi) = \frac{\det(\Phi)^{\frac{\gamma}{2}}}{2^{\frac{\gamma d}{2}}\Gamma_d\left(\frac{\gamma}{2}\right)} \det(\Sigma_i)^{-\frac{\gamma+d+1}{2}} \exp\left(-\frac{1}{2}\operatorname{tr}(\Phi\Sigma_i^{-1})\right),$$

for any $i \in \{1, 2, \ldots, m\}$ where $\Gamma_d(\cdot)$ is the multivariate gamma function of order d, and $\operatorname{tr}(\cdot)$ is the trace function.

(iii) *Prior for* α: A matrix of $\alpha_{i,j} = (\alpha^1_{i,j}, \alpha^2_{i,j}, \ldots, \alpha^K_{i,j}) \in \mathbb{R}^K$ such that $\alpha = (\alpha_{i,j})$, where $\{(i,j) \in \Omega^2 : i \neq j\}$. The multivariate Gaussian distribution is the prior distribution for the parameter $\alpha_{i,j}$ such that

$$p(\alpha) = \prod_{\substack{(i,j)\in\Omega^2 \\ i\neq j}} p(\alpha_{i,j}), \tag{4.8}$$

where $\alpha_{i,j} \sim \mathcal{N}_K(\mu_A, \Lambda_A^{-1})$, for $\{(i,j) \in \Omega^2 : i \neq j\}$, and $\mu_A \in \mathbb{R}^K$ & $\Lambda_A \in \mathbb{R}^{K\times K}$ are the hyperparameters of $\alpha_{i,j}$. Hence,

$$\begin{aligned} p(\alpha_{i,j}) &= [(2\pi)^K \det(\Lambda_A^{-1})]^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\alpha_{i,j} - \mu_A)^\intercal \Lambda_A (\alpha_{i,j} - \mu_A)\right) \\ &= \det(2\pi\Lambda_A^{-1})^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\alpha_{i,j} - \mu_A)^\intercal \Lambda_A (\alpha_{i,j} - \mu_A)\right), \end{aligned}$$

for any $\{(i,j) \in \Omega^2 : i \neq j\}$.

4.3 Likelihood Function

The following notations will be used in this chapter:

$$\begin{aligned} \boldsymbol{\theta}_{-\boldsymbol{\mu}} &= (\cancel{\boldsymbol{\mu}}, \boldsymbol{\Sigma}, \alpha, s) = (\boldsymbol{\Sigma}, \alpha, s) \\ \boldsymbol{\theta}_{-\boldsymbol{\Sigma}} &= (\boldsymbol{\mu}, \cancel{\boldsymbol{\Sigma}}, \alpha, s) = (\boldsymbol{\mu}, \alpha, s) \\ \boldsymbol{\theta}_{-\alpha} &= (\boldsymbol{\mu}, \boldsymbol{\Sigma}, \cancel{\alpha}, s) = (\boldsymbol{\mu}, \boldsymbol{\Sigma}, s) \\ \boldsymbol{\theta}_{-s} &= (\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, \cancel{s}) = (\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha). \end{aligned}$$

The observations are assumed to be distributed in the multivariate Gaussian distribution given the hidden state of a Markov chain. The joint likelihood function of sequences of the observations and the hidden states given the rest of parameters, $f(\mathbf{y}, s \,|\, \boldsymbol{\theta}_{-s}, z, \rho)$, for the model is defined as follows (Spezia, 2010):

$$\begin{aligned}
f(\mathbf{y}, s \,|\, \boldsymbol{\theta}_{-s}, z, \rho) &= f(\mathbf{y}, s \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, z, \rho) \\
&= f(s \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, z, \rho) f(\mathbf{y} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, s, z, \rho) \\
&= f(s_1 \,|\, \rho) \prod_{t=2}^{n} f(s_t \,| s_{t-1}, \alpha, z_t) \prod_{t=1}^{n} f(\mathbf{y}_t \,|\, \boldsymbol{\mu}_{s_t}, \Sigma_{s_t}, s_t) \\
&= \rho_{s_1} \prod_{t=2}^{n} q^{t}_{s_{t-1}, s_t} \prod_{t=1}^{n} \det(2\pi\Sigma_{s_t})^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (\mathbf{y}_t - \boldsymbol{\mu}_{s_t})^{\intercal} \Sigma_{s_t}^{-1} (\mathbf{y}_t - \boldsymbol{\mu}_{s_t}) \right),
\end{aligned} \tag{4.9}$$

where $q^{t}_{s_{t-1}, s_t}$, for any $t \in \{2, 3, \ldots, n\}$ is defined as in Equation (4.2).

4.4 Posterior Distributions

Given the prior density functions in Equations (4.3), (4.6), and (4.8), and the joint likelihood function in Equation (4.9), the joint posterior distribution of the parameters given the observed data and the rest of variables of interest is derived as follows:

$$\begin{aligned}
\pi(\boldsymbol{\theta} \,|\, \mathbf{y}, z, \rho) &= \pi(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, s \,|\, \mathbf{y}, z, \rho) \\
&\propto f(\mathbf{y}, s \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, z, \rho) p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, z, \rho) \\
&= f(s \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, z, \rho) f(\mathbf{y} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, s, z, \rho) p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha \,|\, z, \rho) p(z, \rho) \\
&\propto f(s \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, z, \rho) f(\mathbf{y} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, s, z, \rho) p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha) \\
&= f(s \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, z, \rho) f(\mathbf{y} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, s, z, \rho) p(\boldsymbol{\mu}) p(\boldsymbol{\Sigma}) p(\alpha) \\
&= f(s_1 \,|\, \rho) \prod_{t=2}^{n} f(s_t \,| s_{t-1}, \alpha, z_t) \prod_{t=1}^{n} f(\mathbf{y}_t \,|\, \boldsymbol{\mu}_{s_t}, \Sigma_{s_t}, s_t) \\
&\qquad \times \prod_{i=1}^{m} p(\boldsymbol{\mu}_i) \prod_{i=1}^{m} p(\Sigma_i) \prod_{\substack{(i,j)\in\Omega^2 \\ i\neq j}} p(\alpha_{i,j}) \\
&= \rho_{s_1} \prod_{t=2}^{n} q^{t}_{s_{t-1}, s_t} \prod_{t=1}^{n} (2\pi)^{-\frac{d}{2}} \det(\Sigma_{s_t})^{-\frac{1}{2}} \exp\left(-\frac{1}{2} (\mathbf{y}_t - \boldsymbol{\mu}_{s_t})^{\intercal} \Sigma_{s_t}^{-1} (\mathbf{y}_t - \boldsymbol{\mu}_{s_t}) \right)
\end{aligned}$$

$$
\begin{aligned}
&\times \left[\prod_{i=1}^{m} \det(2\pi\Psi)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\varphi})^\intercal \Psi^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\varphi})\right)\right] \\
&\quad \times \left[\prod_{i=1}^{m} \frac{\det(\Phi)^{\frac{\gamma}{2}}}{2^{\frac{\gamma d}{2}}\Gamma_d\left(\frac{\gamma}{2}\right)} \det(\Sigma_i)^{-\frac{\gamma+d+1}{2}} \exp\left(-\frac{1}{2}\operatorname{tr}(\Phi\Sigma_i^{-1})\right)\right] \\
&\quad\quad \times \left[\prod_{\substack{(i,j)\in\Omega^2 \\ i\neq j}} \det(2\pi\Lambda_A^{-1})^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\alpha_{i,j} - \mu_A)^\intercal \Lambda_A(\alpha_{i,j} - \mu_A)\right)\right] \\
\propto\ & \rho_{s_1} \prod_{t=2}^{n} q^t_{s_{t-1},s_t} \prod_{t=1}^{n} \det(\Sigma_{s_t})^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{y}_t - \boldsymbol{\mu}_{s_t})^\intercal \Sigma_{s_t}^{-1}(\mathbf{y}_t - \boldsymbol{\mu}_{s_t})\right) \\
&\times \left[\prod_{i=1}^{m} \exp\left(-\frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\varphi})^\intercal \Psi^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\varphi})\right)\right] \\
&\quad \times \left[\prod_{i=1}^{m} \det(\Sigma_i)^{-\frac{\gamma+d+1}{2}} \exp\left(-\frac{1}{2}\operatorname{tr}(\Phi\Sigma_i^{-1})\right)\right] \\
&\quad\quad \times \left[\prod_{\substack{(i,j)\in\Omega^2 \\ i\neq j}} \exp\left(-\frac{1}{2}(\alpha_{i,j} - \mu_A)^\intercal \Lambda_A(\alpha_{i,j} - \mu_A)\right)\right]. \qquad (4.10)
\end{aligned}
$$

Likewise, Bayesian inference of the multivariate NHGHMM proceeds in a similar manner as the univariate NHGHMM does. The parameters $(\boldsymbol{\mu}^{(\tau)}, \boldsymbol{\Sigma}^{(\tau)}, \alpha^{(\tau)}, s^{(\tau)})$ are updated by the Gibbs sampler and Metropolis-Hastings algorithms at τ^{th} iteration, depending on their conditional posterior distributions. Given the joint posterior distribution in Equation (4.10), the derivation of the conditional posterior distributions of the parameters is detailed as follows.

The conditional posterior distribution of s

The forward filtering backward sampling algorithm was implemented anew in order to estimate a sequence of the hidden states at τ^{th} iteration in the multivariate NHGHMM such that $s^{(\tau)} = (s_1^{(\tau)}, s_2^{(\tau)}, \dots, s_n^{(\tau)})$. The joint posterior distribution of s is given as follows:

$$
\begin{aligned}
\pi(s \mid \mathbf{y}, \boldsymbol{\theta}_{-s}) &= \pi(s_1 \mid s_2, s_3, \dots, s_n, \mathbf{y}, \boldsymbol{\theta}_{-s}) \\
&\quad \times \pi(s_2 \mid s_3, s_4, \dots, s_n, \mathbf{y}, \boldsymbol{\theta}_{-s}) \\
&\quad\quad \vdots
\end{aligned}
$$

$$
\begin{aligned}
&\times \pi(s_{n-1} \,|\, s_n, \mathbf{y}, \boldsymbol{\theta}_{-s}) \\
&\times \pi(s_n \,|\, \mathbf{y}, \boldsymbol{\theta}_{-s}) \\
= {}& \pi(s_n \,|\, \mathbf{y}, \boldsymbol{\theta}_{-s}) \prod_{t=1}^{n-1} \pi(s_t \,|\, s_{t+1:n}, \mathbf{y}, \boldsymbol{\theta}_{-s}).
\end{aligned}
$$

For any $t \in \{1, 2, \ldots, n-1\}$, the conditional posterior distribution of s_t will be reduced by Bayes' Theorem as follows:

$$
\begin{aligned}
\pi(s_t \,|\, s_{t+1:n}, \mathbf{y}, \boldsymbol{\theta}_{-s}) &\propto f(s_{t+1:n}, \mathbf{y}_{t+1:n} \,|\, s_t, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}) \pi(s_t \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}) \\
&\propto \underbrace{f(s_{t+2:n}, \mathbf{y}_{t+1:n} \,|\, s_{t+1}, \cancel{s_t}, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s})}_{1^{\text{st}} \text{ order Markov chain}} \underbrace{\pi(s_{t+1} \,|\, s_t, \cancel{\mathbf{y}_{1:t}}, \boldsymbol{\theta}_{-s})}_{\text{Markov property}} \pi(s_t \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}) \\
&= \underbrace{f(s_{t+2:n}, \mathbf{y}_{t+1:n} \,|\, s_{t+1}, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s})}_{\text{independent of } s_t} \pi(s_{t+1} \,|\, s_t, \boldsymbol{\theta}_{-s}) \pi(s_t \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}) \\
&\propto \pi(s_{t+1} \,|\, s_t, \boldsymbol{\theta}_{-s}) \pi(s_t \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}). \qquad (4.11)
\end{aligned}
$$

Therefore, the conditional posterior distribution of the hidden state at time t is given as above. Calculation of filtered probabilities up to time $t = n$ is performed, and it is followed by sampling hidden states from the full conditional posterior distribution in Equation (4.11) in reverse time $t \in \{n, n-1, \ldots, 1\}$ (Chib, 1996; Scott, 2002).

The conditional posterior distribution of α

For any $\{(i, j) \in \Omega^2 : \; i \neq j\}$, the full conditional posterior distribution of $\alpha_{i,j}$ is given as follows:

$$
\begin{aligned}
\pi(\alpha_{i,j} \,|\, \boldsymbol{\theta}_{-\alpha}) &\propto \left[\prod_{\{t \geq 2 :\, s_{t-1} = i \,\wedge\, s_t = j\}} q_{i,j}^t \right] \exp\left(-\frac{1}{2} (\alpha_{i,j} - \mu_A)^\intercal \Lambda_A (\alpha_{i,j} - \mu_A) \right) \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(by Equation (4.10))} \\
&= \left[\prod_{\{t \geq 2 :\, s_{t-1} = i \,\wedge\, s_t = j\}} \frac{\exp(z_t^\intercal \alpha_{i,j})}{1 + \sum_{\substack{j \in \Omega \\ j \neq i}} \exp(z_t^\intercal \alpha_{i,j})} \right] \exp\left(-\frac{1}{2} (\alpha_{i,j} - \mu_A)^\intercal \Lambda_A (\alpha_{i,j} - \mu_A) \right) \\
&= \frac{\exp\left(\sum_{\{t \geq 2 :\, s_{t-1} = i \,\wedge\, s_t = j\}} z_t^\intercal \alpha_{i,j} \right)}{\prod_{\{t \geq 2 :\, s_{t-1} = i \,\wedge\, s_t = j\}} \left(1 + \sum_{\substack{j \in \Omega \\ j \neq i}} \exp(z_t^\intercal \alpha_{i,j}) \right)} \exp\left(-\frac{1}{2} (\alpha_{i,j} - \mu_A)^\intercal \Lambda_A (\alpha_{i,j} - \mu_A) \right)
\end{aligned}
$$

$$= \frac{\exp\left(\sum_{\{t\geq 2:\, s_{t-1}=i\,\wedge\, s_t=j\}} z_t^\intercal \alpha_{i,j} - \frac{1}{2}(\alpha_{i,j}-\mu_A)^\intercal \Lambda_A(\alpha_{i,j}-\mu_A)\right)}{\prod_{\{t\geq 2:\, s_{t-1}=i\,\wedge\, s_t=j\}}\left(1+\sum_{\substack{j\in\Omega\\ j\neq i}}\exp(z_t^\intercal \alpha_{i,j})\right)}. \tag{4.12}$$

The conditional posterior distribution in Equation (4.12) has a non-standard form, and hence, the Gibbs sampler can no longer be implemented for parameter estimation. Thus, the Metropolis-Hastings algorithm will be used. The multivariate Gaussian distribution is used for $\alpha_i^{(\tau)} = (\alpha_{i,1}^{(\tau)}, \alpha_{i,2}^{(\tau)}, \ldots, \alpha_{i,i-1}^{(\tau)}, \mathbf{0}, \alpha_{i,i+1}^{(\tau)}, \ldots, \alpha_{i,m}^{(\tau)}) \in \mathbb{R}^{K\times m}$. The conditional posterior distribution of α_i, for any $i \in \{1,2,\ldots,m\}$, is given as follows:

$$\pi(\alpha_i^{(\tau)}\,|\,\boldsymbol{\theta}_{-\alpha}) \propto \frac{\exp\left(\sum_{\{t\geq 2:\, s_{t-1}=i\}} z_t^\intercal \alpha_{i,j}^{(\tau)} - \frac{1}{2}\sum_{\substack{j\in\Omega\\ j\neq i}}(\alpha_{i,j}^{(\tau)}-\mu_A)^\intercal \Lambda_A(\alpha_{i,j}^{(\tau)}-\mu_A)\right)}{\prod_{\{t\geq 2:\, s_{t-1}^{(\tau)}=i\}}\left(1+\sum_{\substack{j\in\Omega\\ j\neq i}}\exp(z_t^\intercal \alpha_{i,j}^{(\tau)})\right)}, \tag{4.13}$$

for any $\tau \in \{2,3,\ldots,N\}$.

The conditional posterior distribution of $\boldsymbol{\mu}$

For $\boldsymbol{\mu}_i$ for any $i\in\{1,2,\ldots,m\}$, define $\nu_i^{(\tau)} = \#\{t \geq 1:\ s_t = i\}$. Then,

$$\begin{aligned}
\pi(\boldsymbol{\mu}_i\,|\,\boldsymbol{\theta}_{-\mu}) &\propto \left[\prod_{\{t\geq 1:\, s_t=i\}} \exp\left(-\frac{1}{2}(\mathbf{y}_t-\boldsymbol{\mu}_i)^\intercal \Sigma_i^{-1}(\mathbf{y}_t-\boldsymbol{\mu}_i)\right)\right] \\
&\qquad \times \exp\left(-\frac{1}{2}(\boldsymbol{\mu}_i-\boldsymbol{\varphi})^\intercal \Psi^{-1}(\boldsymbol{\mu}_i-\boldsymbol{\varphi})\right) \\
&= \exp\left(-\frac{1}{2}\sum_{\{t\geq 1:\, s_t=i\}}(\mathbf{y}_t-\boldsymbol{\mu}_i)^\intercal \Sigma_i^{-1}(\mathbf{y}_t-\boldsymbol{\mu}_i)\right) \\
&\qquad \times \exp\left(-\frac{1}{2}(\boldsymbol{\mu}_i-\boldsymbol{\varphi})^\intercal \Psi^{-1}(\boldsymbol{\mu}_i-\boldsymbol{\varphi})\right) \\
&= \exp\left(-\frac{1}{2}\sum_{\{t\geq 1:\, s_t=i\}}\left[\mathbf{y}_t^\intercal \Sigma_i^{-1}\mathbf{y}_t - \mathbf{y}_t^\intercal \Sigma_i^{-1}\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal \Sigma_i^{-1}\mathbf{y}_t + \boldsymbol{\mu}_i^\intercal \Sigma_i^{-1}\boldsymbol{\mu}_i\right]\right) \\
&\qquad \times \exp\left(-\frac{1}{2}\left[\boldsymbol{\mu}_i^\intercal \Psi^{-1}\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal \Psi^{-1}\boldsymbol{\varphi} - \boldsymbol{\varphi}^\intercal \Psi^{-1}\boldsymbol{\mu}_i + \boldsymbol{\varphi}^\intercal \Psi^{-1}\boldsymbol{\varphi}\right]\right) \\
&\propto \exp\left(-\frac{1}{2}\sum_{\{t\geq 1:\, s_t=i\}}\left[-\mathbf{y}_t^\intercal \Sigma_i^{-1}\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal \Sigma_i^{-1}\mathbf{y}_t + \boldsymbol{\mu}_i^\intercal \Sigma_i^{-1}\boldsymbol{\mu}_i\right]\right) \\
&\qquad \times \exp\left(-\frac{1}{2}\left[\boldsymbol{\mu}_i^\intercal \Psi^{-1}\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal \Psi^{-1}\boldsymbol{\varphi} - \boldsymbol{\varphi}^\intercal \Psi^{-1}\boldsymbol{\mu}_i\right]\right)
\end{aligned}$$

$$= \exp\left(-\frac{1}{2}\left[\sum_{\{t\geq 1:\, s_t=i\}} \boldsymbol{\mu}_i^\intercal \Sigma_i^{-1}\boldsymbol{\mu}_i + \boldsymbol{\mu}_i^\intercal \Psi^{-1}\boldsymbol{\mu}_i - \sum_{\{t\geq 1:\, s_t=i\}} \mathbf{y}_t^\intercal \Sigma_i^{-1}\boldsymbol{\mu}_i \right.\right.$$
$$\left.\left. - \sum_{\{t\geq 1:\, s_t=i\}} \boldsymbol{\mu}_i^\intercal \Sigma_i^{-1}\mathbf{y}_t - \boldsymbol{\mu}_i^\intercal \Psi^{-1}\boldsymbol{\varphi} - \boldsymbol{\varphi}^\intercal\Psi^{-1}\boldsymbol{\mu}_i\right]\right)$$
$$= \exp\left(-\frac{1}{2}\left[\boldsymbol{\mu}_i^\intercal(\nu_i\Sigma_i^{-1}+\Psi^{-1})\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal\left(\Sigma_i^{-1}\left(\sum_{\{t\geq 1:\, s_t=i\}}\mathbf{y}_t\right) + \Psi^{-1}\boldsymbol{\varphi}\right)\right.\right.$$
$$\left.\left. - \left(\left(\sum_{\{t\geq 1:\, s_t=i\}}\mathbf{y}_t^\intercal\right)\Sigma_i^{-1} - \boldsymbol{\varphi}^\intercal\Psi^{-1}\right)\boldsymbol{\mu}_i\right]\right).$$

Define

$$A_i = \nu_i\Sigma_i^{-1}+\Psi^{-1}, \text{ and } \mathbf{b}_i = \Sigma_i^{-1}\left(\sum_{\{t\geq 1:\, s_t=i\}}\mathbf{y}_t\right) + \Psi^{-1}\boldsymbol{\varphi}.$$

Since Ψ^{-1} and Σ_i^{-1}, for any $i \in \{1,2,\ldots,m\}$, are positive definite and symmetric matrices, it follows that

$$(\Sigma_i^{-1})^\intercal = (\Sigma_i^\intercal)^{-1} = \Sigma_i^{-1}, \text{ and } (\Psi^{-1})^\intercal = (\Psi^\intercal)^{-1} = \Psi^{-1}.$$

Furthermore, $A_i = \nu_i\Sigma_i^{-1}+\Psi^{-1}$ is invertible since it is the weighted sum of two symmetric, full-rank covariance matrices. Therefore, there exists A_i^{-1} such that $A_i^{-1}A_i = A_iA_i^{-1} = I$. It follows that

$$\pi(\boldsymbol{\mu}_i \,|\, \boldsymbol{\theta}_{-\boldsymbol{\mu}}) \propto \exp\left(-\frac{1}{2}\left[\boldsymbol{\mu}_i^\intercal A_i\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal\mathbf{b}_i - \mathbf{b}_i^\intercal\boldsymbol{\mu}_i\right]\right)$$
$$= \exp\left(-\frac{1}{2}\left[\boldsymbol{\mu}_i^\intercal A_i\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal\mathbf{b}_i - \mathbf{b}_i^\intercal\boldsymbol{\mu}_i\right]\right)\exp\left(-\frac{1}{2}\mathbf{b}_i^\intercal A_i^{-1}\mathbf{b}_i + \frac{1}{2}\mathbf{b}_i^\intercal A_i^{-1}\mathbf{b}_i\right)$$
$$\propto \exp\left(-\frac{1}{2}\left[\boldsymbol{\mu}_i^\intercal A_i\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal\mathbf{b}_i - \mathbf{b}_i^\intercal\boldsymbol{\mu}_i\right]\right)\exp\left(-\frac{1}{2}\mathbf{b}_i^\intercal A_i^{-1}\mathbf{b}_i\right)$$
$$= \exp\left(-\frac{1}{2}\left[\boldsymbol{\mu}_i^\intercal A_i\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal\mathbf{b}_i - \mathbf{b}_i^\intercal\boldsymbol{\mu}_i + \mathbf{b}_i^\intercal A_i^{-1}\mathbf{b}_i\right]\right)$$
$$= \exp\left(-\frac{1}{2}\left[\boldsymbol{\mu}_i^\intercal A_i\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal A_iA_i^{-1}\mathbf{b}_i - \mathbf{b}_i^\intercal A_i^{-1}A_i\boldsymbol{\mu}_i + \mathbf{b}_i^\intercal A_i^{-1}A_iA_i^{-1}\mathbf{b}_i\right]\right).$$

Define $\Sigma_{\varpi(i)} = A_i^{-1}$, and $\boldsymbol{\mu}_{\varpi(i)} = A_i^{-1}\mathbf{b}_i$. Then,

$$\pi(\boldsymbol{\mu}_i \,|\, \boldsymbol{\theta}_{-\boldsymbol{\mu}}) \propto \exp\left(-\frac{1}{2}[\boldsymbol{\mu}_i^\intercal\Sigma_{\varpi(i)}^{-1}\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^\intercal\Sigma_{\varpi(i)}^{-1}\boldsymbol{\mu}_{\varpi(i)} - \boldsymbol{\mu}_{\varpi(i)}^\intercal\Sigma_{\varpi(i)}^{-1}\boldsymbol{\mu}_i + \boldsymbol{\mu}_{\varpi(i)}^\intercal\Sigma_{\varpi(i)}^{-1}\boldsymbol{\mu}_{\varpi(i)}]\right)$$

$$= \exp\left(-\frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_{\varpi(i)})^\intercal \Sigma_{\varpi(i)}^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_{\varpi(i)})\right). \tag{4.14}$$

Equation (4.14) is in a standard form expression, and hence, the Gibbs sampler is used for updating the parameter. Given that the standard form is known, the parameter of interest is distributed as follows:

$$\boldsymbol{\mu}_i^{(\tau)} \,|\, \boldsymbol{\theta}_{-\boldsymbol{\mu}} \sim \mathcal{N}_d(\boldsymbol{\mu}_{\varpi(i)}, \Sigma_{\varpi(i)}), \tag{4.15}$$

where $\boldsymbol{\mu}_{\varpi(i)} = \left(\nu_i\left(\Sigma_i^{(\tau-1)}\right)^{-1} + \Psi^{-1}\right)^{-1}\left[\left(\Sigma_i^{(\tau-1)}\right)^{-1}\left(\sum_{\{t\geq 1:\, s_t=i\}} \mathbf{y}_t\right) + \Psi^{-1}\boldsymbol{\varphi}\right]$, and $\Sigma_{\varpi(i)} = \left(\nu_i\left(\Sigma_i^{(\tau-1)}\right)^{-1} + \Psi^{-1}\right)^{-1}$, for any $\tau \in \{2, 3, \ldots, N\}$.

The conditional posterior distribution of $\boldsymbol{\Sigma}$

With respect to Σ_i for any $i \in \{1, 2, \ldots, m\}$,

$$\begin{aligned}
\pi(\Sigma_i \,|\, \boldsymbol{\theta}_{-\boldsymbol{\Sigma}}) &\propto \left[\prod_{\{t\geq 1:\, s_t=i\}} \det(\Sigma_i)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{y}_t - \boldsymbol{\mu}_i)^\intercal \Sigma_i^{-1}(\mathbf{y}_t - \boldsymbol{\mu}_i)\right)\right] \\
&\qquad \times \det(\Sigma_i)^{-\frac{\gamma+d+1}{2}} \exp\left(-\frac{1}{2}\operatorname{tr}(\Phi\Sigma_i^{-1})\right) \\
&= \det(\Sigma_i)^{-\frac{\nu_i}{2}} \det(\Sigma_i)^{-\frac{\gamma+d+1}{2}} \\
&\qquad \times \exp\left(-\frac{1}{2}\sum_{\{t\geq 1:\, s_t=i\}} (\mathbf{y}_t - \boldsymbol{\mu}_i)^\intercal \Sigma_i^{-1}(\mathbf{y}_t - \boldsymbol{\mu}_i)\right) \exp\left(-\frac{1}{2}\operatorname{tr}(\Phi\Sigma_i^{-1})\right) \\
&= \det(\Sigma_i)^{-\frac{\nu_i+\gamma+d+1}{2}} \\
&\qquad \times \exp\left(-\frac{1}{2}\left[\sum_{\{t\geq 1:\, s_t=i\}} (\mathbf{y}_t - \boldsymbol{\mu}_i)^\intercal \Sigma_i^{-1}(\mathbf{y}_t - \boldsymbol{\mu}_i) + \operatorname{tr}(\Phi\Sigma_i^{-1})\right]\right) \\
&= \det(\Sigma_i)^{-\frac{\nu_i+\gamma+d+1}{2}} \\
&\qquad \times \exp\left(-\frac{1}{2}\left[\operatorname{tr}\left(\sum_{\{t\geq 1:\, s_t=i\}} (\mathbf{y}_t - \boldsymbol{\mu}_i)(\mathbf{y}_t - \boldsymbol{\mu}_i)^\intercal \Sigma_i^{-1}\right) + \operatorname{tr}(\Phi\Sigma_i^{-1})\right]\right) \\
&= \det(\Sigma_i)^{-\frac{\nu_i+\gamma+d+1}{2}} \\
&\qquad \times \exp\left(-\frac{1}{2}\operatorname{tr}\left(\sum_{\{t\geq 1:\, s_t=i\}} (\mathbf{y}_t - \boldsymbol{\mu}_i)(\mathbf{y}_t - \boldsymbol{\mu}_i)^\intercal \Sigma_i^{-1} + \Phi\Sigma_i^{-1}\right)\right) \\
&= \det(\Sigma_i)^{-\frac{\nu_i+\gamma+d+1}{2}}
\end{aligned}$$

$$\times \exp\left(-\frac{1}{2}\operatorname{tr}\left(\left(\sum_{\{t\geq 1:\, s_t=i\}} (\mathbf{y}_t-\boldsymbol{\mu}_i)(\mathbf{y}_t-\boldsymbol{\mu}_i)^\intercal + \Phi\right)\Sigma_i^{-1}\right)\right). \tag{4.16}$$

Equation (4.16) now has a standard form expression. Therefore, the Gibbs sampler is used for updating the parameter. The parameter of interest is distributed as follows:

$$\Sigma_i^{(\tau)} \mid \boldsymbol{\theta}_{-\boldsymbol{\Sigma}} \sim \mathcal{W}^{-1}\left(\nu_i^{(\tau)}+\gamma, \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} (\mathbf{y}_t-\boldsymbol{\mu}_i^{(\tau)})(\mathbf{y}_t-\boldsymbol{\mu}_i^{(\tau)})^\intercal + \Phi\right), \tag{4.17}$$

for any $\tau \in \{2,3,\ldots,N\}$.

The conditional posterior distributions of the parameters derived in Equations (4.11), (4.13), (4.14), and (4.16) have been defined. It remains to outline the algorithmic structure for sampling from those conditional posterior distributions of the multivariate NHGHMM in the following section.

4.5 The MCMC Algorithms

Recall that the parameters of interest comprise state specific parameters for the multivariate Gaussian distribution $(\boldsymbol{\mu}_i, \Sigma_i)$, for any $i \in \Omega$, the parameter for the transition probability matrices $\alpha_{i,j}$, for any $\{(i,j) \in \Omega^2 : j \neq i\}$, and hidden states s_t, for all $t \in \{1,2,\ldots,n\}$. Therefore, a vector $(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \alpha, s)$ is estimated in the parameter estimation.

In this section, the main algorithm (Algorithm 12) is performed, followed by the forward filtering backward sampling algorithm (Algorithm 13) to update a sequence of the hidden states.

As for updating the parameter α, three MCMC algorithms were proposed to add my original contributions to the literature about Bayesian analyses of *multivariate non-homogeneous Gaussian* HMM. The algorithms include:

Algorithm 14 the AM algorithm,

Algorithm 15 the symmetric DRAM algorithm, and

Algorithm 16 the MTAM algorithm.

The main objective of proposing Algorithms 14, 15, and 16 is to achieve faster convergence since the standard Metropolis-Hastings algorithm may suffer from poor mixing in high dimensions (Brooks et al., 2011). These algorithms are motivated by the fact that it is optimal to have a proposal covariance matrix for the multivariate Gaussian distributed parameter α (Brooks et al., 2011).

In addition, the AM, symmetric DRAM, and MTAM algorithms are indeed able to adapt the target covariance matrix even in very high dimensions such as the multivariate Gaussian distribution of the parameter α (Roberts and Rosenthal, 2009).

The main algorithm for parameter estimation is described in Algorithm 12 as follows.

Algorithm 12 The Main Algorithm (Multivariate NHGHMM)

```
1: Initialise $\boldsymbol{\theta}^{(1)} = (\boldsymbol{\mu}^{(1)}, \boldsymbol{\Sigma}^{(1)}, \alpha^{(1)}, s^{(1)})$.
2: for $\tau = 2, 3, \ldots, N$ do
3:     Forward Filtering Backward Sampling Step:
4:     Update $s^{(\tau)} \mid \boldsymbol{\mu}^{(\tau-1)}, \boldsymbol{\Sigma}^{(\tau-1)}, \alpha^{(\tau-1)}$ by Algorithm 13.
5:     AM/Symmetric DRAM/MTAM Step:
6:     Update $\alpha^{(\tau)} \mid s^{(\tau)}, \boldsymbol{\mu}^{(\tau-1)}, \boldsymbol{\Sigma}^{(\tau-1)}$ by either Algorithm 14, Algorithm 15, or Algorithm 16.
7:     Gibbs Sampling Step:
8:     Update $\boldsymbol{\mu}^{(\tau)} \mid s^{(\tau)}, \alpha^{(\tau)}, \boldsymbol{\Sigma}^{(\tau-1)}$ by the Gibbs sampler.
9:     Update $\boldsymbol{\Sigma}^{(\tau)} \mid s^{(\tau)}, \alpha^{(\tau)}, \boldsymbol{\mu}^{(\tau)}$ by the Gibbs sampler.
10: end for
11: Label switching algorithm is implemented to relabel the parameters by identifiability such that $\boldsymbol{\mu}_1 \prec \boldsymbol{\mu}_2 \prec \cdots \prec \boldsymbol{\mu}_m$ in a lexicographic order.
```

Initialisation of the Parameters

For each $i \in \Omega$, the parameter $\boldsymbol{\mu}_i^{(1)}$ is initialised by using Hartigan-Wong algorithm to minimise the following expression:

$$\arg\min_{\mathcal{C}} \sum_{i\in\Omega} \sum_{\mathbf{y}\in C_i} \|\mathbf{y} - \boldsymbol{\mu}_i^{(1)}\|^2,$$

where $\mathcal{C} = (C_1, C_2, \ldots, C_m)$ is a collection of clusters C_i, and the centroid for the corresponding cluster is denoted by the parameter $\boldsymbol{\mu}_i^{(1)}$. The function, `kmeans`, is available from 'stats' package in R©.

For each $t \in \{1, 2, \ldots, n\}$, the parameter $s_t^{(1)}$ is initialised as follows:

$$s_t^{(1)} = \arg\min_{i \in \Omega} (\mathbf{y}_t - \boldsymbol{\mu}_i^{(1)})^2.$$

For each $i \in \Omega$, the parameter $\Sigma_i^{(1)}$ is initialised as follows:

$$\Sigma_i^{(1)} = \frac{1}{\nu_i^{(1)} - 1} \sum_{\{t \geq 1 :\, s_t^{(1)} = i\}} (\mathbf{y}_t - \boldsymbol{\mu}_i^{(1)})(\mathbf{y}_t - \boldsymbol{\mu}_i^{(1)})^\intercal,$$

where $\nu_i^{(1)} = \#\{t \geq 1 :\ s_t^{(1)} = i\}$.

For each $\{(i, j) \in \Omega^2 : i \neq j\}$, define $\alpha_{i,j}^\circ = (\alpha_{i,j}^2, \alpha_{i,j}^3, \ldots, \alpha_{i,j}^K)$. Then, the parameter $\alpha_{i,j}^{(1)}$ is initialised as follows:

$$\alpha_{i,j}^{(1)} = (\log(\nu_{i,j}^{(1)} / \nu_i^{(1)}), \alpha_{i,j}^\circ),$$

where $\nu_{i,j}^{(1)} = \#\{t \geq 2 :\ s_{t-1}^{(1)} = i \wedge s_t^{(1)} = j\}$, and

$$\alpha_{i,j}^\circ \sim \mathcal{N}_{K-1}(\mu_A^\circ, (\Lambda_A^\circ)^{-1}),$$

where $\mu_A^\circ = (\mu_A^2, \mu_A^3, \ldots, \mu_A^K) \in \mathbb{R}^{K-1}$ and $\Lambda_A^\circ = (\Lambda_A^{i,j})_{i,j=2}^K \in \mathbb{R}^{(K-1)\times(K-1)}$ are the hyperparameters defined in Equation (4.8).

Forward Filtering Backward Sampling Step

This algorithm aims to reconstruct a hidden state sequence in the multivariate NHGHMM. The observed values are now multivariate normally distributed. Therefore, the algorithm is required to take that distribution into account. It is intrinsically different from Algorithm 7 in that sense. The forward filtering backward sampling algorithm for the multivariate NHGHMM is described as follows (Chib, 1996).

(i) At $t = 1$, initialise $\pi(s_t \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)}) = \pi(s_1 \,|\, \mathbf{y}_1, \boldsymbol{\theta}_{-s}^{(\tau-1)}) = \rho_i = 1/m$, for all $i \in \Omega$.

(ii) By the law of total probability, it follows that

$$\pi(s_{t+1} \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)}) = \sum_{s_t \in \Omega} \pi(s_{t+1} \,|\, s_t, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)}) \pi(s_t \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)})$$

$$= \sum_{s_t \in \Omega} \pi(s_{t+1} \,|\, s_t, \boldsymbol{\theta}_{-s}^{(\tau-1)}) \pi(s_t \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)}).$$

(by Markov property)

(iii) By Bayes' Theorem, it follows that

$$\pi(s_{t+1} \,|\, \mathbf{y}_{1:t+1}, \boldsymbol{\theta}_{-s}^{(\tau-1)}) \propto \pi(s_{t+1} \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)}) f(\mathbf{y}_{t+1} \,|\, \mathbf{y}_{1:t}, s_{t+1}, \boldsymbol{\theta}_{-s}^{(\tau-1)}),$$

and its normalising constant is given by

$$\sum_{s_{t+1} \in \Omega} \pi(s_{t+1} \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)}) f(\mathbf{y}_{t+1} \,|\, \mathbf{y}_{1:t}, s_{t+1}, \boldsymbol{\theta}_{-s}^{(\tau-1)}).$$

(iv) Alternate the computations between $\pi(s_{t+1} \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)})$ and $\pi(s_{t+1} \,|\, \mathbf{y}_{1:t+1}, \boldsymbol{\theta}_{-s}^{(\tau-1)})$ in Steps (ii) and (iii), respectively, until obtaining $\pi(s_n \,|\, \mathbf{y}, \boldsymbol{\theta}_{-s}^{(\tau-1)})$. Then, sample $s_n^{(\tau)}$ from $\pi(s_n \,|\, \mathbf{y}, \boldsymbol{\theta}_{-s}^{(\tau-1)})$.

(v) By Equation (4.11), it is possible to sample $s_t^{(\tau)}$ from $\pi(s_t \,|\, s_{t+1:n}, \mathbf{y}, \boldsymbol{\theta}_{-s}^{(\tau-1)})$ such that

$$\pi(s_t \,|\, s_{t+1:n}, \mathbf{y}, \boldsymbol{\theta}_{-s}^{(\tau-1)}) \propto \pi(s_t \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)}) \pi(s_{t+1} \,|\, s_t, \boldsymbol{\theta}_{-s}^{(\tau-1)}),$$

and its normalising constant is given by $\sum_{s_t \in \Omega} \pi(s_t \,|\, \mathbf{y}_{1:t}, \boldsymbol{\theta}_{-s}^{(\tau-1)}) \pi(s_{t+1} \,|\, s_t, \boldsymbol{\theta}_{-s}^{(\tau-1)})$.

Thus, the realisation of hidden states $(s_1^{(\tau)}, s_2^{(\tau)}, \ldots, s_n^{(\tau)})$ at τ^{th} iteration can be obtained through the simulation in Step (v), for $t \in \{n-1, n-2, \ldots, 1\}$.

Algorithm 13 Forward Filtering Backward Sampling Algorithm

```
1: Forward Filtering Step:
2: Set $\rho_i = 1/m$, and calculate $\mathcal{F}(S_1 = i) = \rho_i \phi_{(d)}(\mathbf{y}_1; \boldsymbol{\mu}_i^{(\tau-1)}, \Sigma_i^{(\tau-1)})$ for all $i \in \Omega$, where the forward variable is denoted by $\mathcal{F}(\cdot)$, and the $d$-dimensional Gaussian density function is denoted by $\phi_{(d)}(\cdot\,; \boldsymbol{\mu}_i, \Sigma_i)$ for any $i \in \Omega$.
3: Normalise $\mathcal{F}(S_1 = i) \leftarrow \mathcal{F}(S_1 = i) / \sum_{i\in\Omega} \mathcal{F}(S_1 = i)$, for all $i \in \Omega$.
4: for $t = 2, 3, \ldots, n$ do
5:     Calculate $\mathcal{F}(S_t = i) = \sum_{j\in\Omega} \mathcal{F}(S_{t-1} = j) q_{j,i}^t \phi_{(d)}(\mathbf{y}_t; \boldsymbol{\mu}_i^{(\tau-1)}, \Sigma_i^{(\tau-1)})$, for all $i \in \Omega$.
6:     Normalise $\mathcal{F}(S_t = i) \leftarrow \mathcal{F}(S_t = i) / \sum_{i\in\Omega} \mathcal{F}(S_t = i)$, for all $i \in \Omega$.
7: end for
8: Backward Sampling Step:
9: Update $s_n^{(\tau)} = \arg\max_{i\in\Omega} \mathcal{B}(S_n = i) \equiv \arg\max_{i\in\Omega} \mathcal{F}(S_n = i)$, where the backward variable is denoted by $\mathcal{B}(\cdot)$.
10: for $t = n-1, n-2, \ldots, 1$ do
11:     Calculate $\mathcal{B}(S_t = i) = \mathcal{F}(S_t = i) q^{t+1}_{S_t = i, S_{t+1} = s_{t+1}^{(\tau)}}$.
12:     Normalise $\mathcal{B}(S_t = i) \leftarrow \mathcal{B}(S_t = i) / \sum_{i\in\Omega} \mathcal{B}(S_t = i)$.
13:     Update $s_t^{(\tau)}$ from $\{1, 2, \ldots, m\}$ with probability $\mathcal{B}(S_t = i)$.
14: end for
```

Updating the Parameter, $\alpha^{(\tau)}$

Following Spezia (2006), define a proposal distribution of the parameter $\alpha_{i,j}$ given $\alpha_{i,j}^{(\tau-1)}$, for any $\{(i,j) \in \Omega^2 : j \neq i\}$, as the K-dimensional multivariate Gaussian distribution. Therefore, a proposed value $\alpha'_{i,j}$ is distributed as follows:

$$\alpha'_{i,j} \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E),$$

where $E \in \mathbb{R}^{K\times K}$ is a constant covariance matrix of choice. In this thesis, the constant covariance matrix E is defined as follows:

$$\varepsilon_{i,j} = \begin{cases} 13, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases}, \tag{4.18}$$

where $\varepsilon_{i,j} \in E$, for any $(i,j) \in \{1, 2, \ldots, K\}^2$. This ensures a random walk covers the entire posterior distribution of interest (Spezia, 2006). Hence, the proposal distribution $q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)})$ is given by

$$q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}) = (2\pi)^{-\frac{K}{2}} \det(E)^{\frac{1}{2}} \exp\left(-\frac{1}{2}(\alpha'_{i,j} - \alpha_{i,j}^{(\tau-1)})^\intercal E^{-1} (\alpha'_{i,j} - \alpha_{i,j}^{(\tau-1)})\right)$$

$$\propto \exp\left(-\frac{1}{2}(\alpha'_{i,j} - \alpha_{i,j}^{(\tau-1)})^{\intercal} E^{-1}(\alpha'_{i,j} - \alpha_{i,j}^{(\tau-1)})\right).$$

Define $\alpha_i^{(\tau)} = (\alpha_{i,1}^{(\tau)}, \alpha_{i,2}^{(\tau)}, \ldots, \alpha_{i,i-1}^{(\tau)}, \mathbf{0}, \alpha_{i,i+1}^{(\tau)}, \ldots, \alpha_{i,m}^{(\tau)}) \in \mathbb{R}^{K\times m}$, and an acceptance probability of α'_i given $\alpha_i^{(\tau-1)}$ is given as follows (Spezia, 2006):

$$a(\alpha'_i \,|\, \alpha_i^{(\tau-1)}) = \pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha}) \prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}).$$

Moreover, an acceptance probability of $\alpha_i^{(\tau-1)}$ given α'_i is given by

$$a(\alpha_i^{(\tau-1)} \,|\, \alpha'_i) = \pi(\alpha_i^{(\tau-1)} \,|\, \boldsymbol{\theta}_{-\alpha}) \prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha_{i,j}^{(\tau-1)} \,|\, \alpha'_{i,j}),$$

where $\pi(\cdot \,|\, \boldsymbol{\theta}_{-\alpha})$ is the conditional posterior distribution in Equation (4.13). The detailed balance condition is assumed where there exists the stationary distribution, $\pi(\alpha_i \,|\, \boldsymbol{\theta}_{-\alpha})$, for any $i \in \Omega$. Hence, it follows that

$$\frac{a(\alpha'_i \,|\, \alpha_i^{(\tau-1)})}{a(\alpha_i^{(\tau-1)} \,|\, \alpha'_i)} = \frac{\pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha}) \prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)})}{\pi(\alpha_i^{(\tau-1)} \,|\, \boldsymbol{\theta}_{-\alpha}) \prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha_{i,j}^{(\tau-1)} \,|\, \alpha'_{i,j})}. \tag{4.19}$$

Since the proposal distribution of $\alpha_{i,j}$ is symmetric, Equation (4.19) will be reduced to the following expression (Spezia, 2006):

$$\begin{aligned}
\frac{a(\alpha'_i \,|\, \alpha_i^{(\tau-1)})}{a(\alpha_i^{(\tau-1)} \,|\, \alpha'_i)} &= \frac{\pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha}) \cancel{\prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)})}}{\pi(\alpha_i^{(\tau-1)} \,|\, \boldsymbol{\theta}_{-\alpha}) \cancel{\prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha_{i,j}^{(\tau-1)} \,|\, \alpha'_{i,j})}} \\
&= \frac{\pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \boldsymbol{\theta}_{-\alpha})} \\
\Rightarrow a(\alpha'_i \,|\, \alpha_i^{(\tau-1)}) &= \min\left\{1, \frac{\pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \boldsymbol{\theta}_{-\alpha})}\right\}. \qquad (4.20)
\end{aligned}$$

By Equation (4.20), one accepts the proposed value α'_i and sets $\alpha_i^{(\tau)} = \alpha'_i$ with probability $a(\alpha'_i \,|\, \alpha_i^{(\tau-1)})$ at τ^{th} iteration. Otherwise, one rejects it with probability $1 - a(\alpha'_i \,|\, \alpha_i^{(\tau-1)})$, and sets $\alpha_i^{(\tau)} = \alpha_i^{(\tau-1)}$.

As discussed earlier, the standard Metropolis-Hastings algorithm may be faced with poor mixing. Additionally, sampling from the K-dimensional Gaussian distribution may lead to a problem with high dimensionality as the dimension K increases (Brooks et al., 2011). Having said that, the AM algorithm showed highly impressive performance in high dimensions (Roberts and Rosenthal, 2009), and thus, the AM algorithm is preferred over the standard Metropolis-Hastings algorithm in this thesis. Therefore, I propose an alternative algorithm to updating the parameter α by constructing the AM algorithm for the multivariate NHGHMM in the following section.

The AM Step

Since hidden states of the multivariate NHGHMM are univariate, it follows that the AM algorithm for the model is equivalent to that of the univariate model. Once again, recall Section 2.4.1 on Page 34 for a review of the algorithm.

Let the covariance matrix E be a function of $(\alpha_i^{(1)}, \alpha_i^{(2)}, \dots, \alpha_i^{(\tau)})$ such that

$$E(\alpha_i^{(1)}, \alpha_i^{(2)}, \dots, \alpha_i^{(\tau)}) = E_i^{(\tau)} \in \mathbb{R}^{K \times K},$$

where $\alpha_i^{(\nu)} = (\alpha_{i,1}^{(\nu)}, \alpha_{i,2}^{(\nu)}, \dots, \alpha_{i,i-1}^{(\nu)}, \mathbf{0}, \alpha_{i,i+1}^{(\nu)}, \dots, \alpha_{i,m}^{(\nu)}) \in \mathbb{R}^{K \times m}$, for any $i \in \Omega$, and for each $\nu \in \{1, 2, \dots, \tau\}$. By Equation (2.19) on Page 35, the covariance matrix for the Gaussian proposal with mean of a current draw $\alpha_i^{(\tau)}$ is defined as follows (Haario et al., 2001):

$$E_i^{(\tau)} = \begin{cases} E_0, & \text{if } \tau \leq \tau_0 \\ s_K \operatorname{Cov}(\alpha_i^{(1)}, \alpha_i^{(2)}, \dots, \alpha_i^{(\tau-1)}) + s_K \varepsilon I_K, & \text{if } \tau > \tau_0 \end{cases}, \tag{4.21}$$

where E_0 is a constant covariance matrix of choice, $\tau_0 > 0$ is the initial period, $s_K = 2.4^2/K$, $\varepsilon > 0$ is an infinitesimal value, and I_K is a $K \times K$ identity matrix. Following Equation (2.20) on Page 36, the derivation of a recursive formula for the MCMC iterations of $\alpha_i^{(\tau)}$ is given as follows (Haario et al., 2001):

$$\begin{aligned} E_i^{(\tau)} = \frac{\tau-3}{\tau-2} E_i^{(\tau-1)} + \frac{s_K}{\tau-2} \Bigg[\frac{1}{\tau-1} \Big((\tau-2) \bar{\alpha}_i^{(\tau-2)} \bar{\alpha}_i^{(\tau-2)\intercal} \\ -(\tau-1) \left(\bar{\alpha}_i^{(\tau-1)} \alpha_i^{(\tau-1)\intercal} + \left(\bar{\alpha}_i^{(\tau-1)} \alpha_i^{(\tau-1)\intercal} \right)^\intercal \right) + \tau \alpha_i^{(\tau-1)} \alpha_i^{(\tau-1)\intercal} \Big) + \varepsilon I_K \Bigg], \end{aligned} \tag{4.22}$$

where $\bar{\alpha}_i^{(\tau)} = \tau^{-1}\sum_{\nu=1}^{\tau}\alpha_i$, for any $i \in \Omega$, and for any $\tau \in \{1, 2, \ldots, N\}$. The algorithm is then described as follows (Haario et al., 2001).

(i) Initialise $s_K = 2.4^2/K$ and $\varepsilon = 10^{-6}$. Set the initial period, $\tau_0 = 50$.

(ii) If the current MCMC iteration, $\tau = \tau_0$, then initialise arithmetic means $\bar{\alpha}_i^{(\tau-2)}$ and $\bar{\alpha}_i^{(\tau-1)}$, for all $i \in \Omega$.

(iii) If the current MCMC iteration $\tau > \tau_0$, then update the arithmetic means $\bar{\alpha}_i^{(\tau-2)}$ and $\bar{\alpha}_i^{(\tau-1)}$ such that

$$\begin{aligned}\bar{\alpha}_i^{(\tau-2)} &\leftarrow \bar{\alpha}_i^{(\tau-2)} + \frac{\alpha_i^{(\tau-2)} - \bar{\alpha}_i^{(\tau-2)}}{\tau-2}\\ \bar{\alpha}_i^{(\tau-1)} &\leftarrow \bar{\alpha}_i^{(\tau-1)} + \frac{\alpha_i^{(\tau-1)} - \bar{\alpha}_i^{(\tau-1)}}{\tau-1},\end{aligned}$$

respectively, for all $i \in \Omega$.

(iv) Now, it is required to compute the covariance matrices of the multivariate Gaussian distribution by Equation (4.22), for all $i \in \Omega$.

(v) By monitoring an acceptance rate at τ^{th} iteration, $A(\tau)$, tune the covariance matrices, $E_i^{(\tau)}$, for all $i \in \Omega$. If the acceptance rate is less than 0.25, then update $E_i^{(\tau)} \leftarrow 0.95^2 E_i^{(\tau)}$. If the acceptance rate is greater than 0.5, then update $E_i^{(\tau)} \leftarrow 1.05^2 E_i^{(\tau)}$.

(vi) It remains to propose a new parameter for $\alpha_i^{(\tau)}$, for each $i \in \Omega$, from the proposal distribution such that

$$\alpha'_{i,j} \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E_i^{(\tau)}), \tag{4.23}$$

for all $j \in \Omega$. Then, calculate the acceptance ratio of α'_i and $\alpha_i^{(\tau-1)}$, for each $i \in \Omega$. Since the proposal distribution $q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)})$ is symmetric, for all $j \in \Omega$, the acceptance probability of α'_i given $\alpha_i^{(\tau-1)}$ is same as Equation (4.20).

The performance of the AM algorithms in a problem with high dimensionality indeed surpasses that of the standard Metropolis-Hastings algorithm, although optimal scaling of the covariance matrix may still be improved by calibrating proposal distributions at each MCMC iteration (Haario et al., 2006). Hence, Haario et al. (2006) suggested combining the DRMH and the AM algorithms to prevent both over- and under-calibrated proposal distributions.

Since the proposal distribution of interest is symmetric, I attempt to implement a special case of the DRMH algorithm, namely the symmetric delayed rejection Metropolis algorithm.

Algorithm 14 The Adaptive Metropolis Algorithm

```
 1: Initialise the parameters of the AM algorithm, $s_K = 2.4^2/K$ and $\varepsilon = 10^{-6} > 0$.
 2: Set the initial period of the MCMC iterations $\tau_0$, and $E_i^{(\tau_0)} = E_0$, for all $i \in \{1, 2, \ldots, m\}$.
 3: Define $\alpha_i = (\alpha_{i,1}, \alpha_{i,2}, \ldots, \alpha_{i,i-1}, \mathbf{0}, \alpha_{i,i+1}, \ldots, \alpha_{i,m})$, for all $i \in \{1, 2, \ldots, m\}$.
 4: repeat
 5:     for $i = 1, 2, \ldots, m$ do
 6:         if $\tau = \tau_0$ then
 7:             Initialise $\bar{\alpha}_i^{(\tau-2)} = \sum_{\kappa=1}^{\tau-2} \alpha_i^{(\kappa)} \Big/ (\tau-2)$ and $\bar{\alpha}_i^{(\tau-1)} = \sum_{\kappa=1}^{\tau-1} \alpha_i^{(\kappa)} \Big/ (\tau-1)$.
 8:         end if
 9:         if $\tau > \tau_0$ then
10:             Compute $\bar{\alpha}_i^{(\tau-2)} \leftarrow \bar{\alpha}_i^{(\tau-2)} + (\alpha_i^{(\tau-2)} - \bar{\alpha}_i^{(\tau-2)}) \Big/ (\tau-2)$ and $\bar{\alpha}_i^{(\tau-1)} \leftarrow \bar{\alpha}_i^{(\tau-1)} + (\alpha_i^{(\tau-1)} - \bar{\alpha}_i^{(\tau-1)}) \Big/ (\tau-1)$.
11:             Compute
                $$E_i^{(\tau)} = \frac{\tau-3}{\tau-2} E_i^{(\tau-1)} + \frac{s_K}{\tau-2} \left[ \frac{1}{\tau-1} \left( (\tau-2) \bar{\alpha}_i^{(\tau-2)} \bar{\alpha}_i^{(\tau-2)\intercal} - (\tau-1) \left( \bar{\alpha}_i^{(\tau-1)} \alpha_i^{(\tau-1)\intercal} + \left( \bar{\alpha}_i^{(\tau-1)} \alpha_i^{(\tau-1)\intercal} \right)^{\intercal} \right) + \tau \alpha_i^{(\tau-1)} \alpha_i^{(\tau-1)\intercal} \right) + \varepsilon I_K \right],$$
                where $I_K$ denotes the identity matrix of $K$ dimensions.
12:             if $A(\tau) < 0.25$ then
13:                 $E_i^{(\tau)} \leftarrow 0.95^2 E_i^{(\tau)}$
14:             end if
15:             if $A(\tau) > 0.5$ then
16:                 $E_i^{(\tau)} \leftarrow 1.05^2 E_i^{(\tau)}$
17:             end if
18:             Random Walk Metropolis-Hastings Step: The random walk Metropolis-Hastings algorithm, the DRMH algorithm or the MTM algorithm can be implemented in this step to update $\alpha_i^{(\tau)}$.
19:         end if
20:     end for
21:     $\tau \leftarrow \tau + 1$
22: until $N^{\text{th}}$ iteration is reached.
```

The Symmetric DRAM Step

Recall Section 2.4.4 on Page 40 for the symmetric delayed rejection Metropolis algorithm. This algorithm aims to enhance the way of proposing a new parameter in Equation (4.23). The symmetric proposal distribution is used in the algorithm, and only depends on the last rejected sample, that is,

$$q(\alpha'_{i,j(\kappa)} \,|\, \alpha'_{i,j(\kappa-1)}, \alpha'_{i,j(\kappa-2)}, \ldots, \alpha'_{i,j(1)}, \alpha_{i,j}^{(\tau-1)}) = q(\alpha'_{i,j(\kappa)} \,|\, \alpha'_{i,j(\kappa-1)}) = q(\alpha'_{i,j(\kappa-1)} \,|\, \alpha'_{i,j(\kappa)}),$$

where $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha'_{i,j(\kappa-1)}, E_i^{(\tau)})$, for any $\kappa \in \{2, 3, \ldots, \nu\}$ and any $\{(i,j) \in \Omega^2 : i \neq j\}$. The algorithm is described as follows.

(i) Set a rejection stage indicator $\kappa = 1$, and the maximum rejection stage ν.

(ii) If $\kappa < 2$, then propose $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E_i^{(\tau)})$, for all $\{(i,j) \in \Omega^2 : i \neq j\}$. Then, define $\alpha'_{i(\kappa)} = (\alpha'_{i,1(\kappa)}, \alpha'_{i,2(\kappa)}, \ldots, \alpha'_{i,i-1(\kappa)}, \mathbf{0}, \alpha'_{i,i+1(\kappa)}, \ldots, \alpha'_{i,m(\kappa)})$, and set $\alpha_i^* = \alpha'_{i(\kappa)}$.

(iii) Since the symmetric proposal distribution is implemented, the acceptance probability is given as follows:

$$r_1 = \min\left\{1, \frac{\pi(\alpha'_{i(\kappa)} \,|\, \boldsymbol{\theta}_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \boldsymbol{\theta}_{-\alpha})}\right\},$$

where the $\pi(\cdot \,|\, \boldsymbol{\theta}_{-\alpha})$ is the conditional posterior distribution in Equation (4.13). Then, accept $\alpha_i^{(\tau)} = \alpha'_{i(\kappa)}$ with probability r and terminate the algorithm. Otherwise, set $\kappa \leftarrow \kappa + 1$ and move to Step (iv).

(iv) If $\kappa \geq 2$, then propose $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha'_{i,j(\kappa-1)}, E_i^{(\tau)})$, for all $\{(i,j) \in \Omega^2 : i \neq j\}$.

(v) Calculation of the acceptance probability is now different from that of Step (iii). Therefore,

$$r_2 = \min\left\{1, \frac{\pi(\alpha'_{i(\kappa)} \,|\, \boldsymbol{\theta}_{-\alpha}) - \pi(\alpha_i^* \,|\, \boldsymbol{\theta}_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \boldsymbol{\theta}_{-\alpha}) - \pi(\alpha_i^* \,|\, \boldsymbol{\theta}_{-\alpha})}\right\}.$$

(vi) If $\pi(\alpha'_{i(\kappa)} \,|\, \boldsymbol{\theta}_{-\alpha}) > \pi(\alpha_i^* \,|\, \boldsymbol{\theta}_{-\alpha})$, then set $\alpha_i^* = \alpha'_{i(\kappa)}$.

(vii) Draw $u \sim \mathcal{U}(0,1)$. If $u \leq r$, set $\alpha_i^{(\tau)} = \alpha'_{i(\kappa)}$ and terminate the algorithm. Otherwise, set $\kappa \leftarrow \kappa + 1$ and move to Step (ii). If $\kappa > \nu$, then the algorithm is terminated.

As such, the symmetric DRAM algorithm indeed improves the MCMC convergence by suppressing both over- and under-calibrated proposal distributions (Haario et al., 2006). On the other hand, the MTM algorithm utilises a brute-force strategy to increase a searching region of the target distribution (Liu et al., 2000). Furthermore, Liu et al. (2000) showed that combining the MTM algorithm with the adaptive direction sampling method was able to produce a better sampler. As Gilks et al. (1994, 1998) pointed out, the adaptive direction sampling leaves two issues such as

(i) how one can select a meaningful direction, and

(ii) how one can sample from the target distribution effectively.

By using the MTM algorithm, Liu et al. (2000) showed that the novel method was able to perform significantly better.

Thus, it is indisputably rational in this thesis to combine the MTM algorithm and the AM algorithm. By doing so, the MTAM algorithm is expected to achieve a larger searching region than that of the symmetric DRAM algorithm.

The MTAM Step

Recall Section 2.4.2 on Page 37 for the review. It is necessary to construct a weighting function $w(\cdot \,|\, \cdot)$, for the calculation of an acceptance ratio in the algorithm. Given Equations (2.22) and (2.23) on Page 38, the weighting function is given as follows:

$$
\begin{aligned}
w(\alpha'_i \,|\, \alpha_i^{(\tau-1)}) &= \pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha}) \left[\prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}) \lambda(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}) \right] \\
&= \pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha}) \left[\prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}) \left(\frac{q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}) + q(\alpha_{i,j}^{(\tau-1)} \,|\, \alpha'_{i,j})}{2} \right)^{-1} \right] \\
&= \pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha}) \left[\prod_{\substack{j\in\Omega \\ j\neq i}} q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}) \left(\frac{\cancel{2} q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)})}{\cancel{2}} \right)^{-1} \right] \\
&\qquad \text{(since } q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)}) = q(\alpha_{i,j}^{(\tau-1)} \,|\, \alpha'_{i,j}) \text{, for any } \{j \in \Omega : \ j \neq i\})
\end{aligned}
$$

Algorithm 15 The Symmetric Delayed Rejection Adaptive Metropolis Algorithm

```
1: Initialise a rejection stage indicator $\kappa = 1$, and the maximum rejection stage $\nu$.
2: for $i = 1, 2, \ldots, m$ do
3:     repeat
4:         if $\kappa < 2$ then
5:             for $j \in \Omega$ such that $j \neq i$ do
6:                 Simulate $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E_i^{(\tau)})$.
7:             end for
8:             Define $\alpha'_{i(\kappa)} = (\alpha'_{i,1(\kappa)}, \alpha'_{i,2(\kappa)}, \ldots, \alpha'_{i,i-1(\kappa)}, \mathbf{0}, \alpha'_{i,i+1(\kappa)}, \ldots, \alpha'_{i,m(\kappa)})$, and set $\alpha_i^* = \alpha'_{i(\kappa)}$.
9:             Calculate
                   $$r_1 = \min\left\{1, \frac{\pi(\alpha'_{i(\kappa)} \,|\, \boldsymbol{\theta}_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \boldsymbol{\theta}_{-\alpha})}\right\}.$$
10:            Draw $u \sim \mathcal{U}(0,1)$.
11:            if $u \leq r_1$ then
12:                $\alpha_i^{(\tau)} = \alpha'_{i(\kappa)}$
13:            else
14:                $\kappa \leftarrow \kappa + 1$, and move onto the next rejection stage.
15:            end if
16:        else
17:            for $j \in \Omega$ such that $j \neq i$ do
18:                Simulate $\alpha'_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha'_{i,j(\kappa-1)}, E_i^{(\tau)})$.
19:            end for
20:            Calculate
                   $$r_2 = \min\left\{1, \frac{\pi(\alpha'_{i(\kappa)} \,|\, \boldsymbol{\theta}_{-\alpha}) - \pi(\alpha_i^* \,|\, \boldsymbol{\theta}_{-\alpha})}{\pi(\alpha_i^{(\tau-1)} \,|\, \boldsymbol{\theta}_{-\alpha}) - \pi(\alpha_i^* \,|\, \boldsymbol{\theta}_{-\alpha})}\right\}.$$
21:            if $\pi(\alpha'_{i(\kappa)} \,|\, \boldsymbol{\theta}_{-\alpha}) > \pi(\alpha_i^* \,|\, \boldsymbol{\theta}_{-\alpha})$ then
22:                Set $\alpha_i^* = \alpha'_{i(\kappa)}$.
23:            end if
24:            Draw $u \sim \mathcal{U}(0,1)$.
25:            if $u \leq r_2$ then
26:                $\alpha_i^{(\tau)} = \alpha'_{i(\kappa)}$
27:            else
28:                $\kappa \leftarrow \kappa + 1$, and move onto the next rejection stage.
29:            end if
30:        end if
31:    until $\alpha'_{i(\kappa)}$ is accepted or the loop reaches the $\nu$th stage of rejection.
32: end for
```

$$= \pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha}) \left[\prod_{\substack{j\in\Omega \\ j\neq i}} \cancel{q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)})} \frac{1}{\cancel{q(\alpha'_{i,j} \,|\, \alpha_{i,j}^{(\tau-1)})}} \right]$$

$$= \pi(\alpha'_i \,|\, \boldsymbol{\theta}_{-\alpha}). \tag{4.24}$$

Therefore, By Equation (4.24), the weighting function is equivalent to the full conditional posterior distribution. An acceptance probability of the MTAM algorithm is calculated by Equation (2.24) on Page 38, and hence, it is given as follows:

$$r_3 = a(\alpha_i' \,|\, \alpha_i^{(\tau-1)}) := \min\left\{1, \frac{\pi(\alpha_{i(1)}^* \,|\, \boldsymbol{\theta}_{-\alpha}) + \pi(\alpha_{i(2)}^* \,|\, \boldsymbol{\theta}_{-\alpha}) + \cdots + \pi(\alpha_{i(\varkappa)}^* \,|\, \boldsymbol{\theta}_{-\alpha})}{\pi(\alpha_{i(1)}^{**} \,|\, \boldsymbol{\theta}_{-\alpha}) + \pi(\alpha_{i(2)}^{**} \,|\, \boldsymbol{\theta}_{-\alpha}) + \cdots + \pi(\alpha_{i(\varkappa)}^{**} \,|\, \boldsymbol{\theta}_{-\alpha})}\right\}, \tag{4.25}$$

where $\varkappa \in \mathbb{N}$ is the number of multiple tries. The maximum number of multiple tries was set to be 10 such that $\varkappa = 10$, so that the algorithm is executed in a reasonable amount of time. The MTAM algorithm proceeds as follows.

(i) For $j \in \{1, 2, \ldots, i-1, i+1, \ldots, m\}$, simulate multiple samples, $\{\alpha_{i,j(1)}^*, \alpha_{i,j(2)}^*, \ldots, \alpha_{i,j(\varkappa)}^*\}$, by drawing i.i.d. samples, $\alpha_{i,j(\kappa)}^* \sim \mathcal{N}_K(\alpha_{i,j}^{(\tau-1)}, E_i^{(\tau)})$, for each $\kappa \in \{1, 2, \ldots, \varkappa\}$.

(ii) Define $\alpha_{i(\kappa)}^* = (\alpha_{i,1(\kappa)}^*, \alpha_{i,2(\kappa)}^*, \ldots, \alpha_{i,i-1(\kappa)}^*, \mathbf{0}, \alpha_{i,i+1(\kappa)}^*, \ldots, \alpha_{i,m(\kappa)}^*)$, and draw a proposed value, α_i', from $\{\alpha_{i(1)}^*, \alpha_{i(2)}^*, \ldots, \alpha_{i(\varkappa)}^*\}$ with probability proportional to $\{\pi(\alpha_{i(\kappa)}^* \,|\, \boldsymbol{\theta}_{-\alpha})\}$, for any $\kappa \in \{1, 2, \ldots, \varkappa\}$.

(iii) For $j \in \{1, 2, \ldots, i-1, i+1, \ldots, m\}$, simulate multiple samples, $\{\alpha_{i,j(1)}^{**}, \alpha_{i,j(2)}^{**}, \ldots, \alpha_{i,j(\varkappa-1)}^{**}\}$, by drawing i.i.d. samples, $\alpha_{i,j(\kappa)}^{**} \sim \mathcal{N}_K(\alpha_{i,j}', E_i^{(\tau)})$, for each $\kappa \in \{1, 2, \ldots, \varkappa - 1\}$. Then, set $\alpha_{i(\varkappa)}^{**} = \alpha_i^{(\tau-1)}$, and hence, it follows that $\{\alpha_{i(1)}^{**}, \alpha_{i(2)}^{**}, \ldots, \alpha_{i(\varkappa)}^{**}\}$.

(iv) Calculate the acceptance probability provided by Equation (4.25). That is,

$$r_3 = \min\left\{1, \frac{\pi(\alpha_{i(1)}^* \,|\, \boldsymbol{\theta}_{-\alpha}) + \pi(\alpha_{i(2)}^* \,|\, \boldsymbol{\theta}_{-\alpha}) + \cdots + \pi(\alpha_{i(\varkappa)}^* \,|\, \boldsymbol{\theta}_{-\alpha})}{\pi(\alpha_{i(1)}^{**} \,|\, \boldsymbol{\theta}_{-\alpha}) + \pi(\alpha_{i(2)}^{**} \,|\, \boldsymbol{\theta}_{-\alpha}) + \cdots + \pi(\alpha_{i(\varkappa)}^{**} \,|\, \boldsymbol{\theta}_{-\alpha})}\right\}.$$

(v) Draw $u \sim \mathcal{U}(0, 1)$. If $u \leq r$, then set $\alpha_i^{(\tau)} = \alpha_i'$. Otherwise, set $\alpha_i^{(\tau)} = \alpha_i^{(\tau-1)}$, for all $i \in \Omega$.

(vi) Repeat Steps (i)–(v), for all $i \in \Omega$.

After updating the parameter $\alpha^{(\tau)}$, it remains to update the rest of parameter of interest $(\boldsymbol{\mu}^{(\tau)}, \boldsymbol{\Sigma}^{(\tau)})$.

Algorithm 16 The Multiple-try Adaptive Metropolis Algorithm

```
1: for $i = 1, 2, \ldots, m$ do
2:     for $j \in \Omega$ such that $j \neq i$ do
3:         Simulate multiple random samples $\{\alpha^*_{i,j(1)}, \alpha^*_{i,j(2)}, \ldots, \alpha^*_{i,j(\varkappa)}\}$, by proposing $\alpha^*_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha^{(\tau-1)}_{i,j}, E^{(\tau)}_i)$ for any $\kappa \in \{1, 2, \ldots, \varkappa\}$.
4:     end for
5:     Define $\alpha^*_{i(\kappa)} = (\alpha^*_{i,1(\kappa)}, \alpha^*_{i,2(\kappa)}, \ldots, \alpha^*_{i,i-1(\kappa)}, \mathbf{0}, \alpha^*_{i,i+1(\kappa)}, \ldots, \alpha^*_{i,m(\kappa)})$, and draw $\alpha'_i$ from $\{\alpha^*_{i(1)}, \alpha^*_{i(2)}, \ldots, \alpha^*_{i(\varkappa)}\}$ with probability which is proportional to $\{\pi(\alpha^*_{i(\kappa)} \,|\, \boldsymbol{\theta}_{-\alpha})\}^{\varkappa}_{\kappa=1}$.
6:     for $j \in \Omega$ such that $j \neq i$ do
7:         Simulate multiple random samples $\{\alpha_{i,j(1)}, \alpha_{i,j(2)}, \ldots, \alpha_{i,j(\varkappa-1)}\}$, by proposing $\alpha_{i,j(\kappa)} \sim \mathcal{N}_K(\alpha'_{i,j}, E^{(\tau)}_i)$ for any $\kappa \in \{1, 2, \ldots, \varkappa - 1\}$, and set $\alpha_{i(\varkappa)} = \alpha^{(\tau-1)}_i$.
8:     end for
9:     Calculate the acceptance probability as follows:
           $$r_3 = \min\left\{1, \frac{\pi(\alpha^*_{i(1)} \,|\, \boldsymbol{\theta}_{-\alpha}) + \pi(\alpha^*_{i(2)} \,|\, \boldsymbol{\theta}_{-\alpha}) + \cdots + \pi(\alpha^*_{i(\varkappa)} \,|\, \boldsymbol{\theta}_{-\alpha})}{\pi(\alpha_{i(1)} \,|\, \boldsymbol{\theta}_{-\alpha}) + \pi(\alpha_{i(2)} \,|\, \boldsymbol{\theta}_{-\alpha}) + \cdots + \pi(\alpha_{i(\varkappa)} \,|\, \boldsymbol{\theta}_{-\alpha})}\right\}.$$
10:    Draw $u \sim \mathcal{U}(0, 1)$.
11:    if $u \leq r$ then
12:        $\alpha^{(\tau)}_i = \alpha'_i$
13:    else
14:        $\alpha^{(\tau)}_i = \alpha^{(\tau-1)}_i$
15:    end if
16: end for
```

Gibbs Sampling Step

I implemented the Gibbs sampler on the parameters $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, rather than the Metropolis-Hastings algorithm or its variants, similar to the Metropolis-Hastings-within-Gibbs algorithms used by Heaps et al. (2015). The full conditional posterior distributions are obtained in Equation (4.14) for $\boldsymbol{\mu}^{(\tau)}_i$, and Equation (4.17) for $\Sigma^{(\tau)}_i$, respectively, for all $i \in \Omega$, and any $\tau \in \{2, 3, \ldots, N\}$. Then, the Gibbs sampler proceeds as follows.

(i) For each $i \in \Omega$, simulate

$$\boldsymbol{\mu}^{(\tau)}_i \,|\, \boldsymbol{\theta}_{-\boldsymbol{\mu}} \sim \mathcal{N}_d(\boldsymbol{\mu}_{\varpi(i)}, \Sigma_{\varpi(i)}),$$

where $\boldsymbol{\mu}_{\varpi(i)} = \left(\nu_i \left(\Sigma^{(\tau-1)}_i\right)^{-1} + \Psi^{-1}\right)^{-1} \left[\left(\Sigma^{(\tau-1)}_i\right)^{-1} \left(\sum_{\{t\geq 1:\, s_t = i\}} \mathbf{y}_t\right) + \Psi^{-1}\boldsymbol{\varphi}\right]$,

and $\Sigma_{\varpi(i)} = \left(\nu_i \left(\Sigma^{(\tau-1)}_i\right)^{-1} + \Psi^{-1}\right)^{-1}$, for any $\tau \in \{2, 3, \ldots, N\}$. Ψ and $\boldsymbol{\varphi}$ are the priors defined in Equations (4.4) and (4.7), respectively.

(ii) For each $i \in \Omega$, simulate

$$\Sigma_i^{(\tau)} \,|\, \boldsymbol{\theta}_{-\boldsymbol{\Sigma}} \sim \mathcal{W}^{-1}\left(\nu_i^{(\tau)} + \gamma, \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} (\mathbf{y}_t - \boldsymbol{\mu}_i^{(\tau)})(\mathbf{y}_t - \boldsymbol{\mu}_i^{(\tau)})^\intercal + \Phi\right),$$

where $\mathcal{W}^{-1}(\cdot,\cdot)$ denotes the inverse Wishart distribution.

After completing N iterations, the label switch algorithm was implemented on the MCMC samples to relabel the parameters. It was carried out by identifiability of the parameter $\boldsymbol{\mu}$ with a lexicographic order such that $\boldsymbol{\mu}_1 \prec \boldsymbol{\mu}_2 \prec \cdots \prec \boldsymbol{\mu}_m$. Hence, the statistical inference of the model should be meaningful.

Algorithm 17 Gibbs Sampler (Multivariate NHGHMM)

```
1: for $i = 1, 2, \ldots, m$ do
2:     Simulate $\boldsymbol{\mu}_i^{(\tau)} \,|\, \boldsymbol{\theta}_{-\boldsymbol{\mu}} \sim \mathcal{N}_d(\boldsymbol{\mu}_{\varpi(i)}, \Sigma_{\varpi(i)})$,
       where $\boldsymbol{\mu}_{\varpi(i)} = \left(\nu_i \left(\Sigma_i^{(\tau-1)}\right)^{-1} + \Psi^{-1}\right)^{-1} \left[\left(\Sigma_i^{(\tau-1)}\right)^{-1} \left(\sum_{\{t\geq 1:\, s_t=i\}} \mathbf{y}_t\right) + \Psi^{-1}\boldsymbol{\varphi}\right]$,
       and
       $\Sigma_{\varpi(i)} = \left(\nu_i \left(\Sigma_i^{(\tau-1)}\right)^{-1} + \Psi^{-1}\right)^{-1}$.
3: end for
4: for $i = 1, 2, \ldots, m$ do
5:     Simulate $\Sigma_i^{(\tau)} \,|\, \boldsymbol{\theta}_{-\boldsymbol{\Sigma}} \sim \mathcal{W}^{-1}\left(\nu_i^{(\tau)} + \gamma, \sum_{\{t\geq 1:\, s_t^{(\tau)}=i\}} (\mathbf{y}_t - \boldsymbol{\mu}_i^{(\tau)})(\mathbf{y}_t - \boldsymbol{\mu}_i^{(\tau)})^\intercal + \Phi\right)$.
6: end for
```

As for the computational task, the identical computer was used for the simulation. The specification is exactly the same; an 8-core 3.60 GHz Intel(R) Core(TM) i7-4790 central processor unit with 8 GB of random access memory.

4.6 Simulation Study

To validate the proposed MCMC algorithms, an extensive simulation study of the multivariate NHGHMM is conducted in this section. The number of hidden states is defined as $m = 3$, and the dimension of the process being as $d = 2$. Hence, the observations are bivariate normally distributed. The length of observations is set to be $n = 300$. By Equation (4.1), the stochastic process of bivariate Gaussian distributed observations is defined as follows:

$$\mathbf{Y}_t \,|\, S_t = i \sim \mathcal{N}_2(\boldsymbol{\mu}_i, \Sigma_i),$$

where $\mathbf{Y}_t, \boldsymbol{\mu}_i \in \mathbb{R}^2$ and $\Sigma_i \in \mathbb{R}^{2\times 2}$, for any $i \in \{1, 2, 3\}$ and for any $t \in \{1, 2, \ldots, 300\}$. Since the model assumes non-homogeneity, two transition matrices are introduced where every 60th observations, the two matrices alternate. The model is stated below and Figure 4.1 shows its time series plots of a hidden state sequence and observed values.

$$\begin{gathered}
\boldsymbol{\mu}_1 = (1, 2),\ \boldsymbol{\mu}_2 = (3, 0),\ \boldsymbol{\mu}_3 = (5, 4) \\
\Sigma_1 = \begin{pmatrix} 1.5 & 0.5 \\ 0.5 & 1.0 \end{pmatrix},\ \Sigma_2 = \begin{pmatrix} 2.0 & 0.6 \\ 0.6 & 1.0 \end{pmatrix},\ \Sigma_3 = \begin{pmatrix} 1.5 & -0.5 \\ -0.5 & 2.0 \end{pmatrix} \\
\alpha_{1,1} = (0, 0),\ \alpha_{1,2} = (-1.109, 0.555),\ \alpha_{1,3} = (-3.039, 1.664) \\
\alpha_{2,1} = (-3.193, 1.485),\ \alpha_{2,2} = (0, 0),\ \alpha_{2,3} = (-2.361, 0.376) \\
\alpha_{3,1} = (-2.867, 1.433),\ \alpha_{3,2} = (-1.282, 0.785),\ \alpha_{3,3} = (0, 0) \\
z_{t_1} = (1.00, 0.75),\ z_{t_2} = (1.00, 2.00) \\
Q^{t_1} = \begin{pmatrix} 0.6 & 0.3 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.3 & 0.6 \end{pmatrix},\ Q^{t_2} = \begin{pmatrix} 0.3 & 0.3 & 0.4 \\ 0.4 & 0.5 & 0.1 \\ 0.3 & 0.4 & 0.3 \end{pmatrix},
\end{gathered} \tag{4.26}$$

where

$$\begin{gathered}
t_1 = \{2, 3, \ldots, 60\} \cup \{121, 122, \ldots, 180\} \cup \{241, 242, \ldots, 300\} \\
t_2 = \{61, 62, \ldots, 120\} \cup \{181, 182, \ldots, 240\}.
\end{gathered}$$

4.6.1 MCMC Convergence

Firstly, the number of MCMC iterations for each parameter was set to 1,500,000 and the burn-in time of 1,490,000 was chosen without thinning.

In terms of Geweke's diagnostics and ESS, the MCMC convergence assessments of

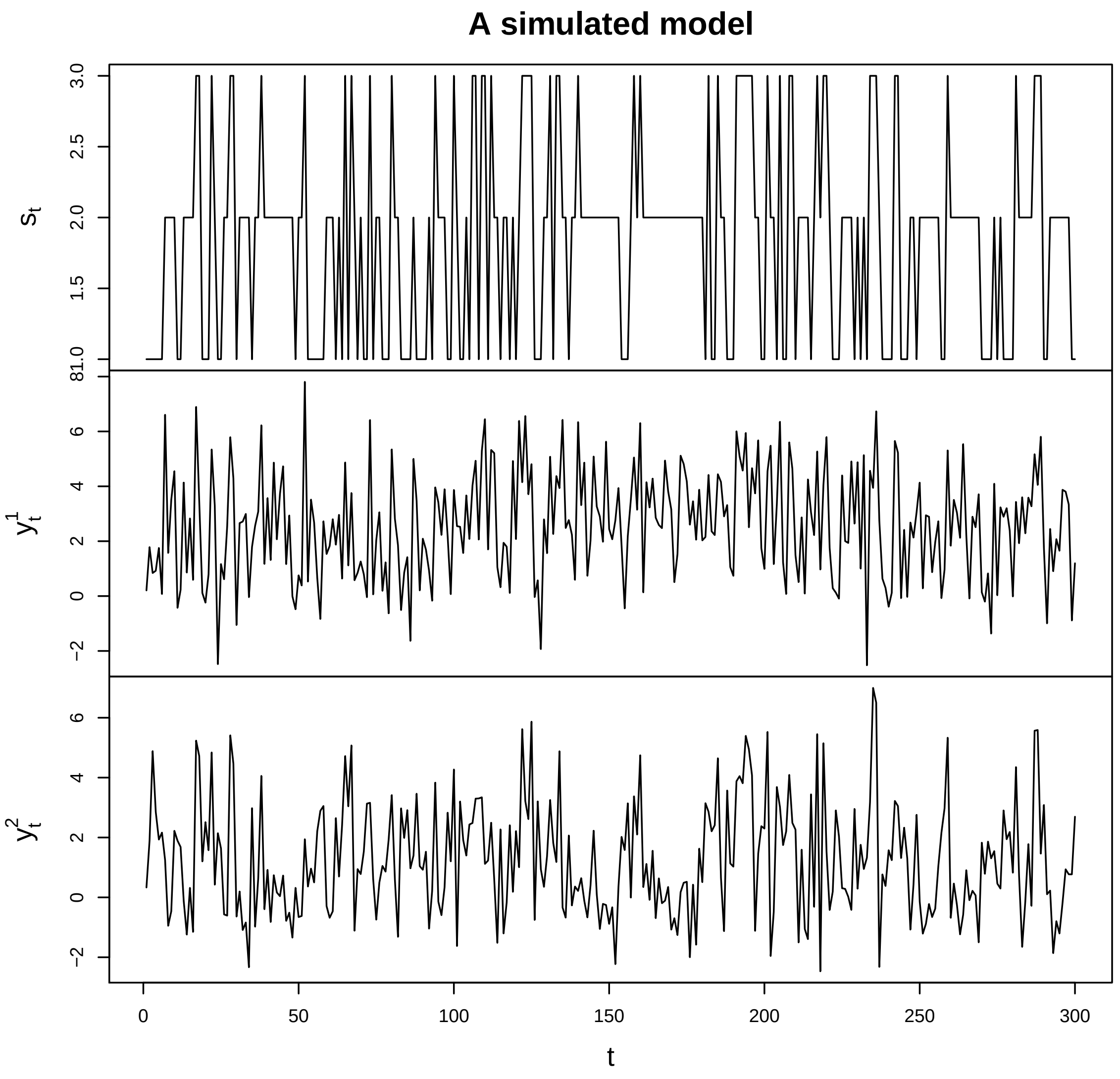


Figure 4.1: Time series plot of Model (4.26). The realisations of hidden states and multivariate observed values are denoted by s_t, y_t^1 and y_t^2, respectively.

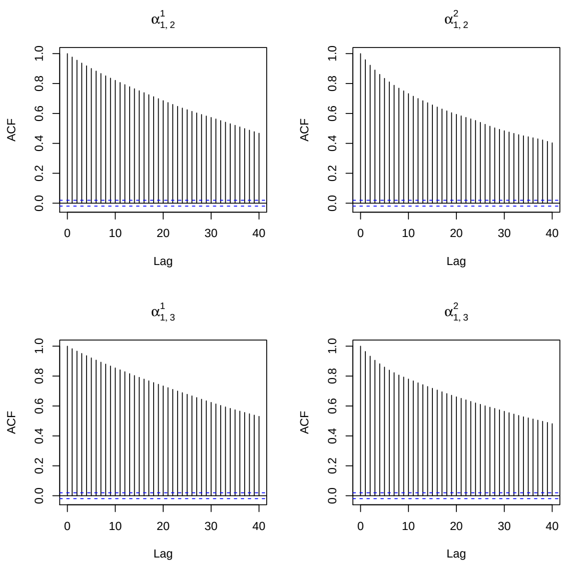
the standard Metropolis-Hastings, AM, symmetric DRAM, and MTAM algorithms are shown in Table 4.1 as follows.

Table 4.1: Summary statistics for Geweke's diagnostics and ESS of the random walk Metropolis-Hastings, AM, symmetric DRAM, and MTAM algorithms without thinning

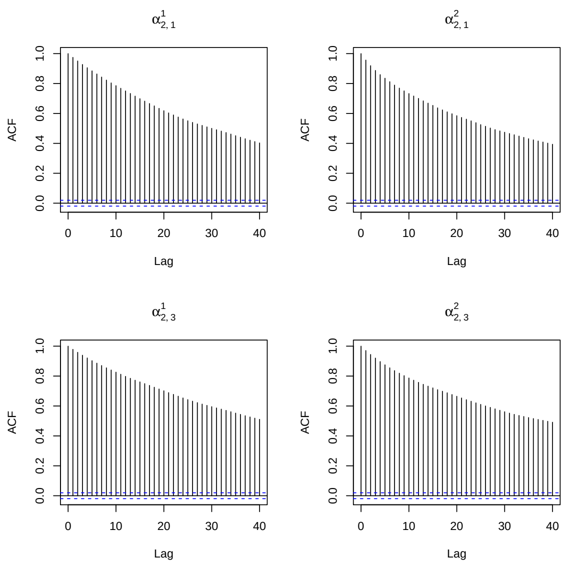
Parameter	Metropolis-Hastings		AM		DRAM		MTAM	
	Geweke's diagnostics	ESS	Geweke's diagnostics	ESS	Geweke's diagnostics	ESS	Geweke's diagnostics	ESS
μ_1^1	−0.931	10 000	−3.651	8 923	1.265	9 708	0.812	10 647
μ_1^2	−0.305	10 000	−2.166	10 338	0.268	11 621	0.208	9 845
μ_2^1	−1.378	10 000	0.550	10 000	−1.342	10 000	1.928	9 710
μ_2^2	0.341	10 000	0.900	11 055	−1.454	9 630	0.285	10 000
μ_3^1	2.137	10 000	0.480	9 027	−0.253	10 000	−1.187	10 000
μ_3^2	−0.678	10 000	−0.636	10 000	0.186	9 558	0.637	10 000
$\Sigma_1^{1,1}$	0.790	10 000	−0.404	9 719	0.125	10 000	−0.072	9 113
$\Sigma_1^{1,2}$	−0.165	10 329	0.194	9 993	−0.324	10 000	1.304	10 000
$\Sigma_1^{2,1}$	−0.165	10 329	0.194	9 993	−0.324	10 000	1.304	10 000
$\Sigma_1^{2,2}$	0.664	9 512	−0.391	11 618	−0.333	9 552	−0.695	10 000
$\Sigma_2^{1,1}$	−1.266	10 000	−1.249	9 499	−0.618	9 380	0.474	10 000
$\Sigma_2^{1,2}$	−0.665	10 000	−0.159	9 781	−0.785	10 000	−0.183	10 000
$\Sigma_2^{2,1}$	−0.665	10 000	−0.159	9 781	−0.785	10 000	−0.183	10 000
$\Sigma_2^{2,2}$	0.411	10 000	1.184	10 201	−2.129	10 000	−0.590	10 047
$\Sigma_3^{1,1}$	−1.029	9 456	0.653	10 000	−0.925	10 000	1.074	10 000
$\Sigma_3^{1,2}$	0.888	9 614	1.059	9 595	−0.537	10 000	−0.292	9 552
$\Sigma_3^{2,1}$	0.888	9 614	1.059	9 595	−0.537	10 000	−0.292	9 552
$\Sigma_3^{2,2}$	−0.422	9 527	−0.776	10 000	−1.188	9 535	0.384	10 000
$\alpha_{1,2}^1$	0.154	100	−0.465	566	−0.831	1 021	0.536	1 893
$\alpha_{1,2}^2$	−0.190	127	−0.237	491	0.202	907	−0.341	1 895
$\alpha_{1,3}^1$	1.022	83	−0.862	630	−0.581	976	−1.057	2 005
$\alpha_{1,3}^2$	−1.117	97	0.351	640	0.013	1 068	1.032	2 092
$\alpha_{2,1}^1$	2.329	128	−0.471	573	0.144	990	−0.010	1 990
$\alpha_{2,1}^2$	−2.393	124	0.607	643	−0.050	1 222	0.249	2 129
$\alpha_{2,3}^1$	2.733	95	0.042	497	−1.024	983	0.031	1 626
$\alpha_{2,3}^2$	−2.774	107	0.445	446	1.103	895	−0.525	1 612
$\alpha_{3,1}^1$	0.583	75	−1.437	476	−1.525	712	0.476	1 340
$\alpha_{3,1}^2$	−0.530	101	1.642	480	1.805	738	−0.651	1 611
$\alpha_{3,2}^1$	0.966	134	−0.418	920	−2.899	1 333	−1.471	2 053
$\alpha_{3,2}^2$	−0.691	144	1.289	838	2.100	1 244	1.591	1 985

The values in red represent that the corresponding parameters have failed to pass Geweke's diagnostics. As such, the standard Metropolis-Hastings, AM, and symmetric DRAM algorithms failed to converge jointly. In addition, each of the ESS of those algorithms is remarkably low. This suggests that there exist dependencies between the MCMC samples.

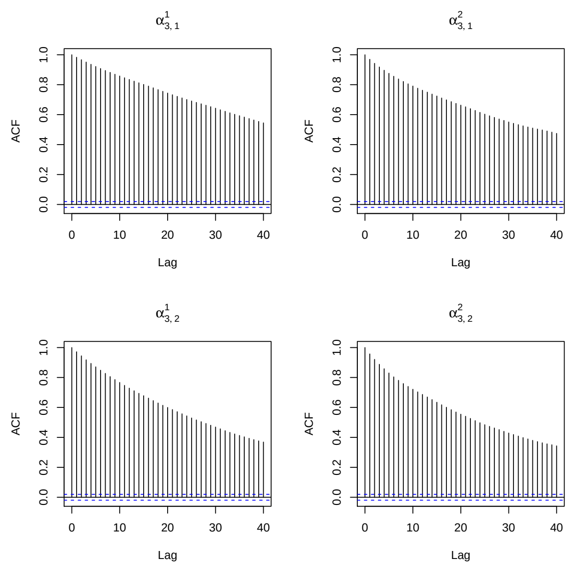
On the other hand, only the MTAM algorithm passed Geweke's diagnostics although the ESS of the parameters $\alpha_{1,2}$, $\alpha_{1,3}$, $\alpha_{2,1}$, $\alpha_{2,3}$, $\alpha_{3,1}$, and $\alpha_{3,2}$ remain low. Therefore, it is claimed that all the MCMC algorithms are invalid for parameter estimation. Figures 4.2–4.5 show autocorrelation functions of those parameters with low ESS for each algorithm. As for the rest of the parameters with respect to autocorrelation functions, see Figures D.1–D.8 in Appendix D. Now, the burn-in time is chosen to



(a)



(b)

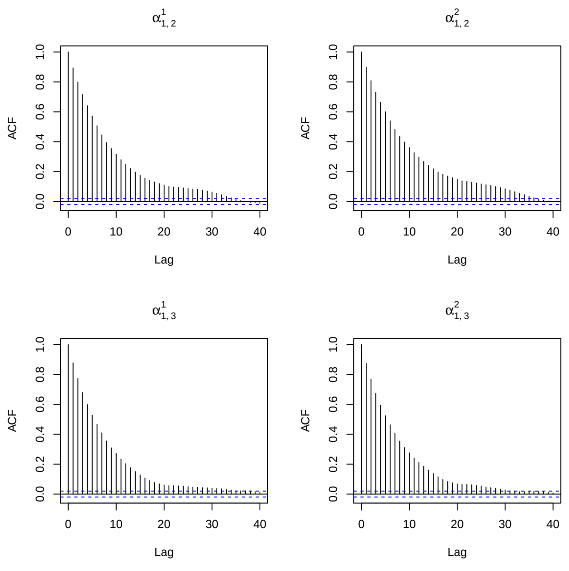


(c)

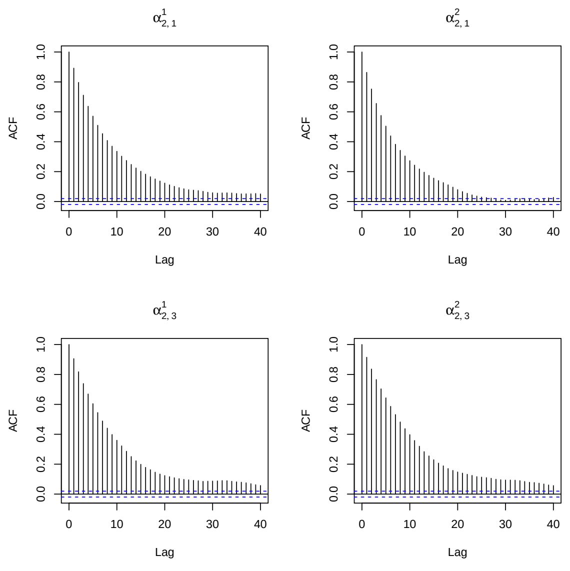
Figure 4.2: Autocorrelation functions of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the standard Metropolis-Hastings algorithm without thinning

be 1,000,000. Then, thinning of every 50th iteration was performed for the last 500,000 iterations, and hence, the 10,000 MCMC samples were taken into account.

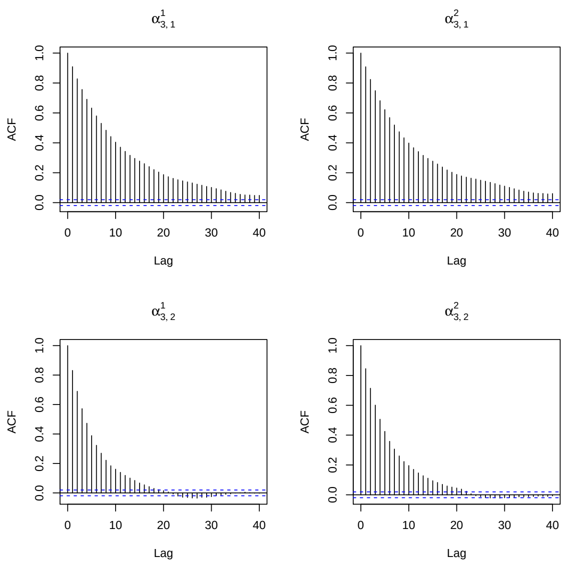
The MCMC convergence assessment using Geweke's diagnostics, ESS, and autocorrelation functions was performed for the standard Metropolis-Hastings, and all the three modified MCMC algorithms, the AM, symmetric DRAM, and MTAM algorithms on Metropolis updates. The results are shown in Table 4.2. As observed, every algo-



(a)



(b)

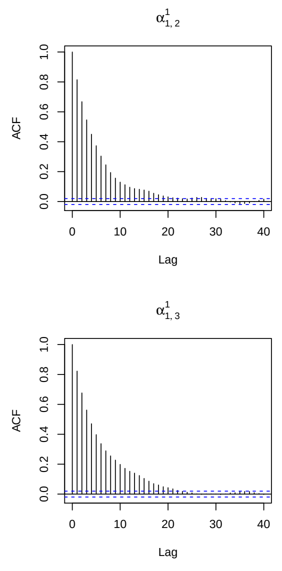


(c)

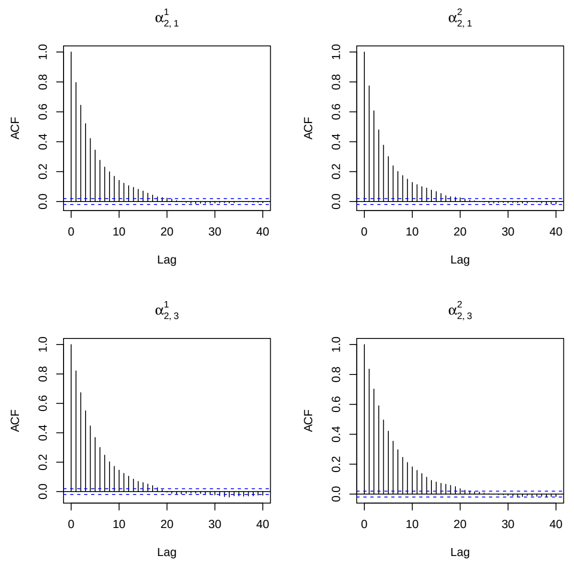
Figure 4.3: Autocorrelation functions of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the AM algorithm without thinning

rithm has passed Geweke's diagnostics by performing thinning. Nevertheless, the ESS of the parameters $\alpha_{1,2}$, $\alpha_{1,3}$, $\alpha_{2,1}$, $\alpha_{2,3}$, $\alpha_{3,1}$, and $\alpha_{3,2}$ for the standard Metropolis-Hastings algorithm are relatively low. There must exist much dependency between those MCMC samples. Hence, the standard Metropolis-Hastings algorithm is now disregarded for the simulation study. Nevertheless, the result of the algorithm for model selection is still presented in the following.

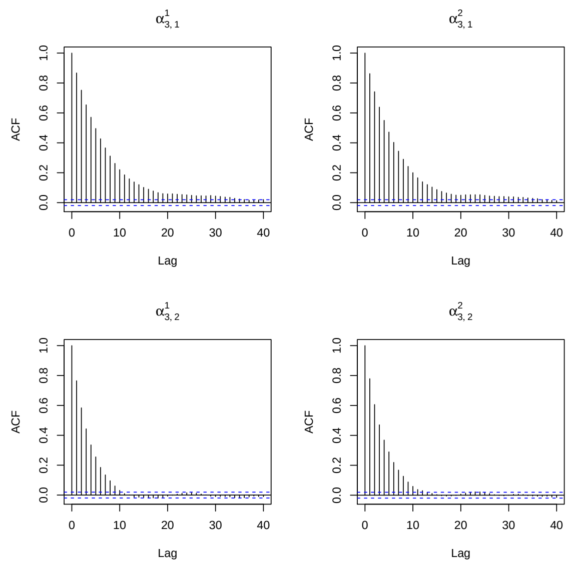
Even though all the proposed MCMC algorithms have passed Geweke's diagnostics,



(a)



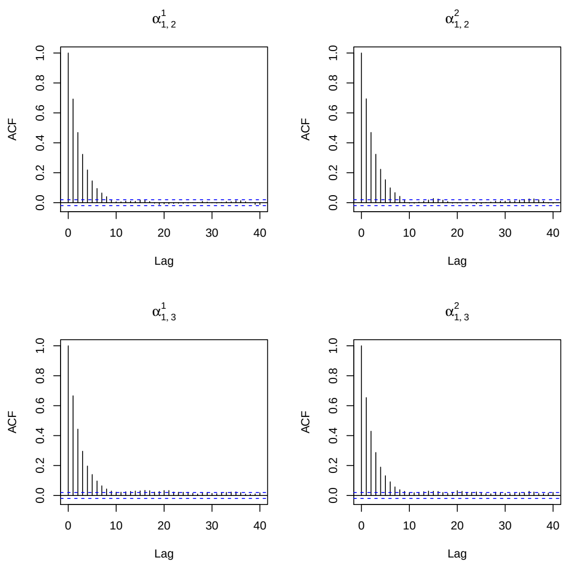
(b)



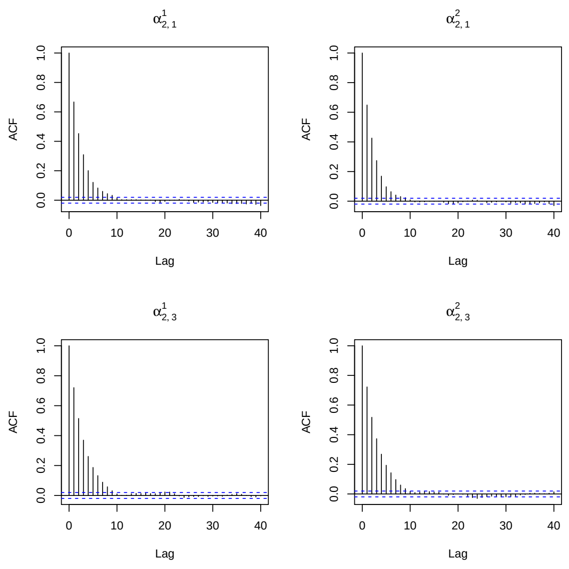
(c)

Figure 4.4: Autocorrelation functions of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the symmetric DRAM algorithm without thinning

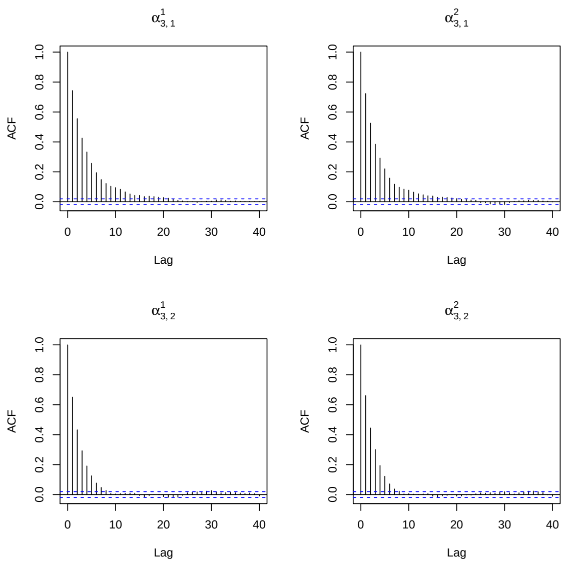
some of the ESS indicates possible dependencies between the MCMC samples, especially the ESS of the parameter $\alpha^2_{2,3}$ for the symmetric DRAM algorithm. Therefore, autocorrelation functions of the parameters, especially α, are investigated to ensure that the MCMC convergence has been met.



(a)



(b)

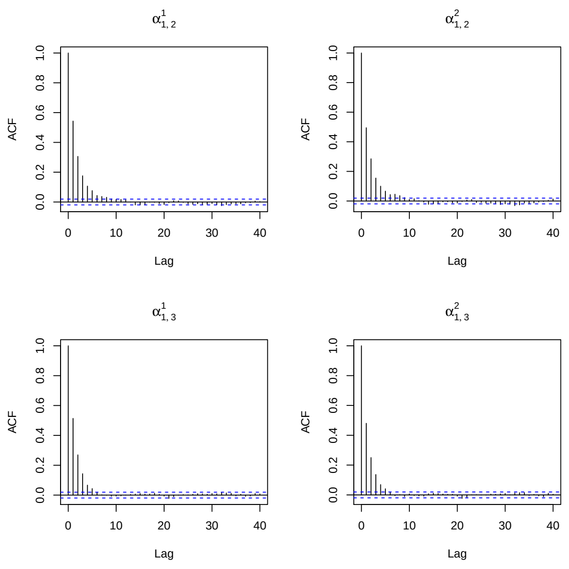


(c)

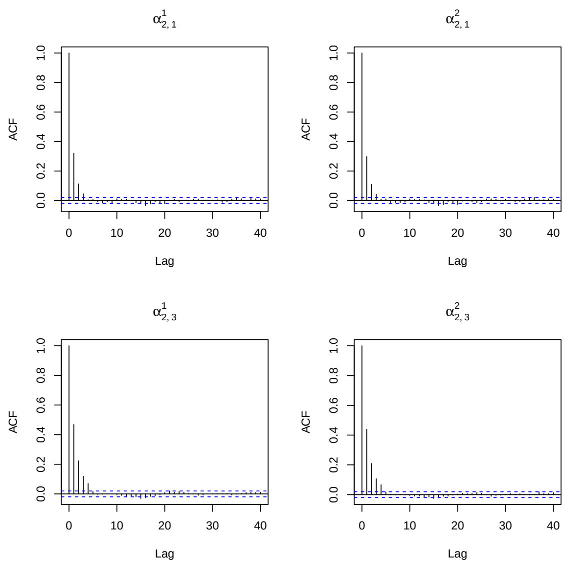
Figure 4.5: Autocorrelation functions of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the MTAM algorithm without thinning

Table 4.2: Summary statistics for Geweke's diagnostics and ESS of the random walk Metropolis-Hastings, AM, symmetric DRAM, and MTAM algorithms with thinning

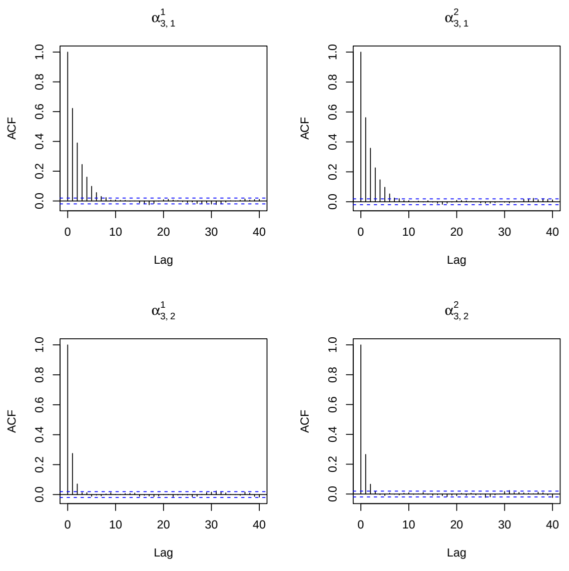
Parameter	Metropolis-Hastings		AM		DRAM		MTAM	
	Geweke's diagnostics	ESS	Geweke's diagnostics	ESS	Geweke's diagnostics	ESS	Geweke's diagnostics	ESS
μ_1^1	0.871	10 000	−0.470	10 000	0.601	9 744	−0.513	10 000
μ_1^2	0.078	8 883	−1.076	10 000	−0.007	10 000	−1.239	11 186
μ_2^1	−0.541	10 044	0.077	10 000	−0.187	10 000	0.788	9 669
μ_2^2	−0.498	10 000	−0.250	9 474	0.860	9 050	0.862	10 000
μ_3^1	−0.054	10 245	−0.864	10 000	−0.015	10 317	0.227	9 696
μ_3^2	0.219	9 674	−1.097	10 000	0.579	10 000	−0.475	10 000
$\Sigma_1^{1,1}$	−0.028	10 000	−1.139	10 000	−0.352	10 000	−0.514	10 000
$\Sigma_1^{1,2}$	−0.499	9 628	−1.059	9 751	−0.376	10 000	0.768	10 000
$\Sigma_1^{2,1}$	−0.499	9 628	−1.059	9 751	−0.376	10 000	0.768	10 000
$\Sigma_1^{2,2}$	−0.305	10 000	−1.710	9 626	1.021	10 339	0.936	10 000
$\Sigma_2^{1,1}$	−1.440	8 152	0.832	10 000	−1.354	10 000	−0.408	10 000
$\Sigma_2^{1,2}$	0.270	10 000	0.620	10 000	−1.820	9 654	0.273	10 000
$\Sigma_2^{2,1}$	0.270	10 000	0.620	10 000	−1.820	9 654	0.273	10 000
$\Sigma_2^{2,2}$	0.225	10 000	1.783	10 588	1.566	10 000	0.008	10 000
$\Sigma_3^{1,1}$	0.198	10 000	0.757	10 000	1.295	10 000	−0.194	10 000
$\Sigma_3^{1,2}$	−1.192	10 000	0.646	10 000	1.052	10 000	1.359	10 000
$\Sigma_3^{2,1}$	−1.192	10 000	0.646	10 000	1.052	10 000	1.359	10 000
$\Sigma_3^{2,2}$	−0.443	10 000	0.258	9 685	−0.726	9 286	0.845	10 000
$\alpha_{1,2}^1$	0.225	2 871	0.062	10 000	−0.052	10 000	−1.522	10 000
$\alpha_{1,2}^2$	0.085	2 770	0.183	10 000	0.734	10 600	1.045	10 000
$\alpha_{1,3}^1$	1.459	3 212	0.353	10 000	0.116	10 637	−0.155	10 000
$\alpha_{1,3}^2$	−1.403	3 335	−0.307	10 000	0.275	8 873	−0.286	9 581
$\alpha_{2,1}^1$	1.305	5 151	0.103	10 000	1.081	10 000	0.221	10 000
$\alpha_{2,1}^2$	−1.184	5 177	0.124	10 000	−0.717	10 000	0.779	10 000
$\alpha_{2,3}^1$	1.424	3 614	0.468	10 000	0.425	10 358	0.548	10 000
$\alpha_{2,3}^2$	−1.786	3 733	−0.621	9 620	0.241	7 906	0.145	10 000
$\alpha_{3,1}^1$	0.225	2 330	−0.914	9 632	0.711	9 148	0.478	10 000
$\alpha_{3,1}^2$	−0.427	2 479	0.744	10 000	−0.917	9 108	−0.825	9 834
$\alpha_{3,2}^1$	0.005	5 686	0.014	9 554	−0.039	10 000	1.696	10 310
$\alpha_{3,2}^2$	−0.077	5 803	−0.248	9 977	0.424	10 000	−1.479	10 961



(a)

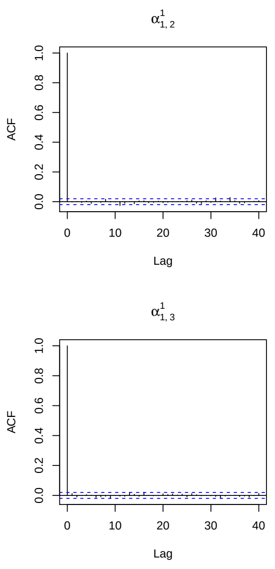


(b)

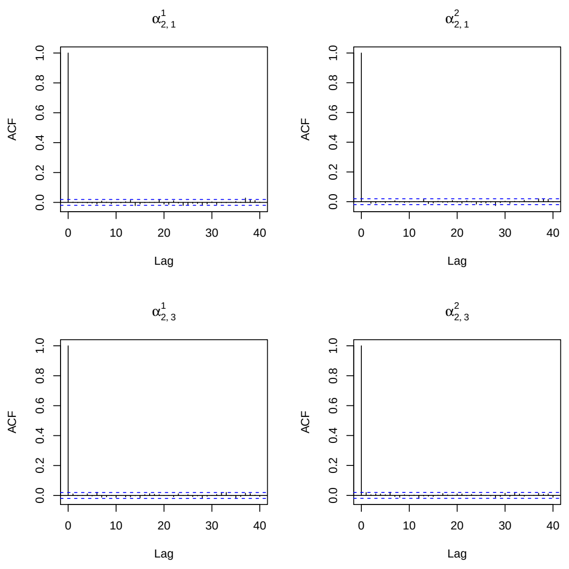


(c)

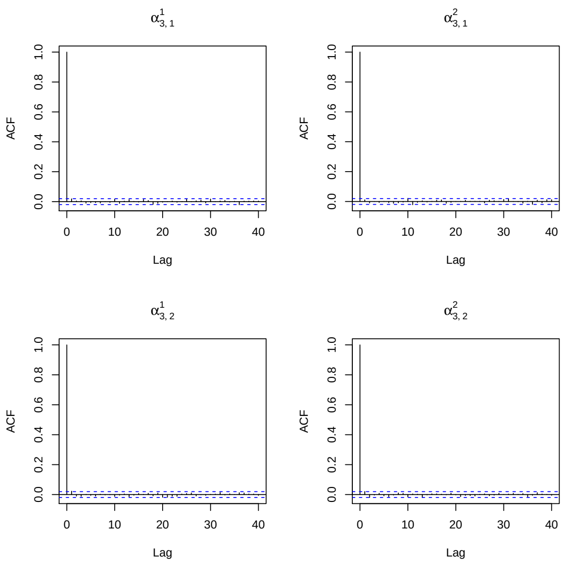
Figure 4.6: Autocorrelation functions of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the standard Metropolis-Hastings algorithm with thinning



(a)



(b)



(c)

Figure 4.7: Autocorrelation functions of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the AM algorithm with thinning

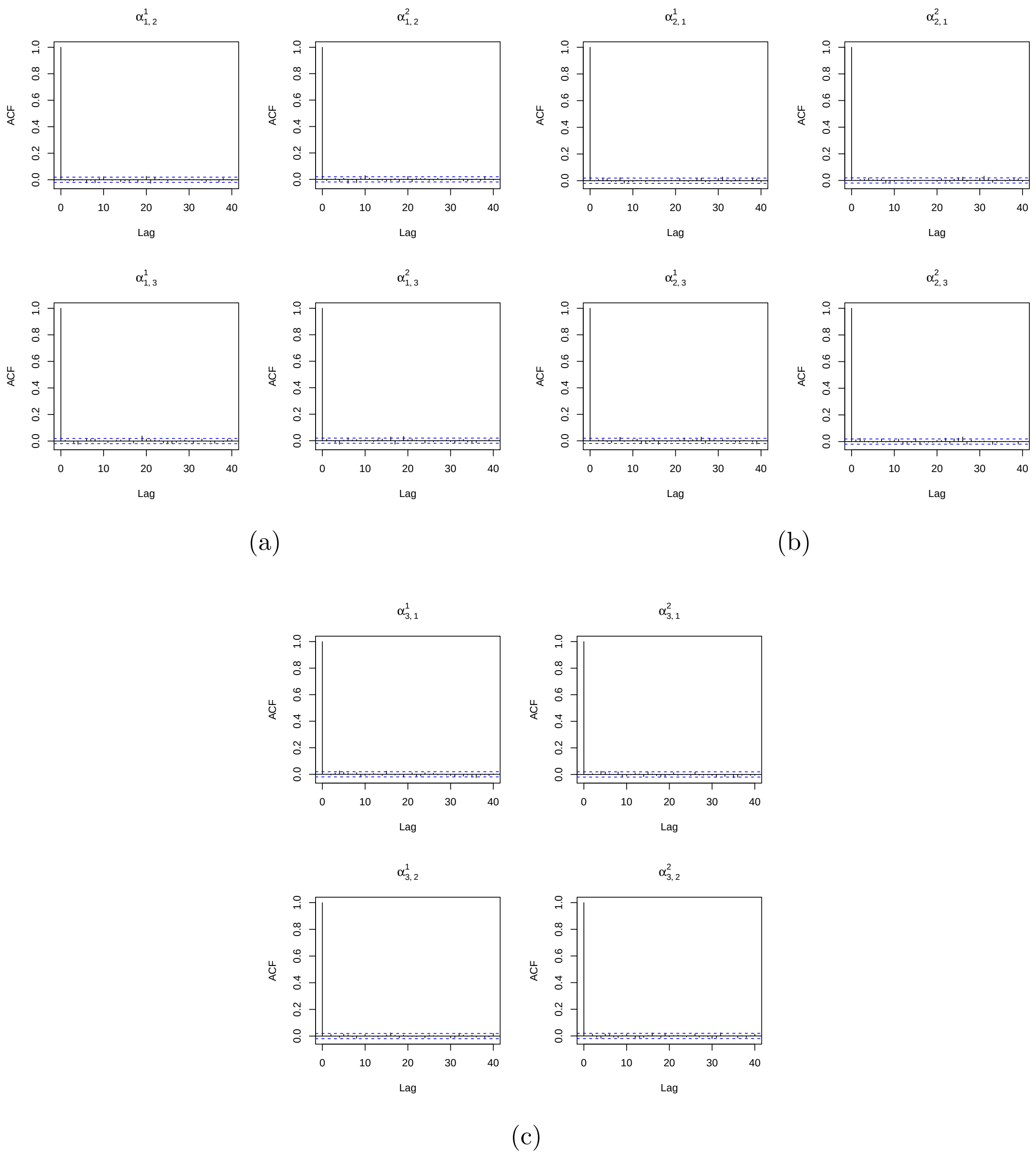
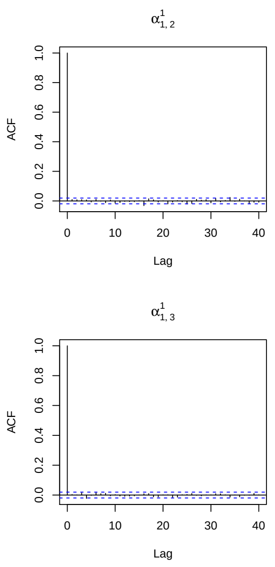
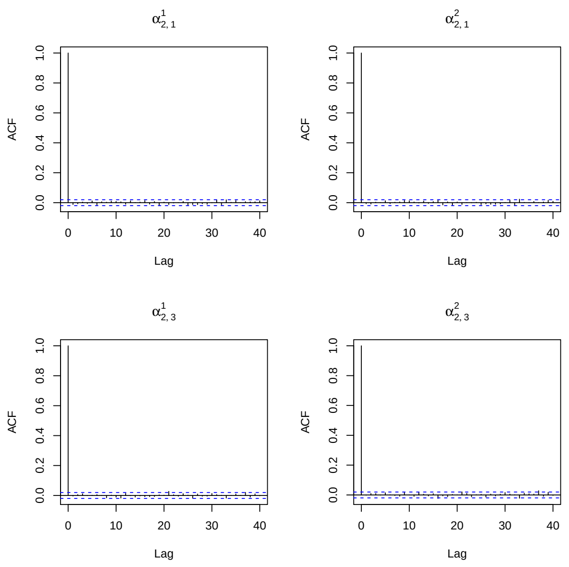


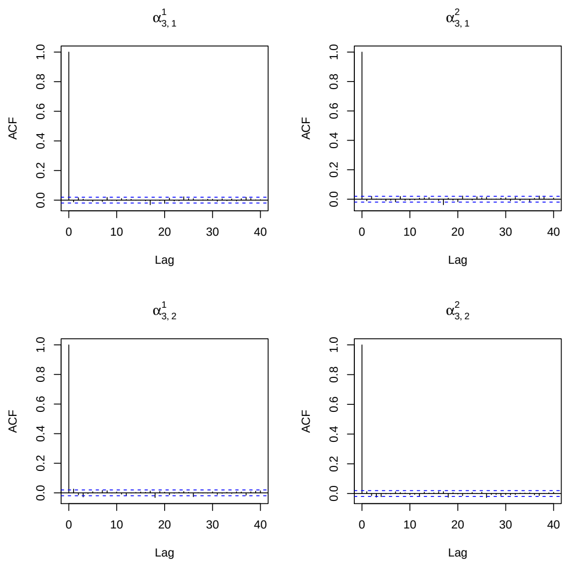
Figure 4.8: Autocorrelation functions of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the symmetric DRAM algorithm with thinning



(a)



(b)



(c)

Figure 4.9: Autocorrelation functions of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the MTAM algorithm with thinning

According to Figure 4.6, there are still some dependencies between the MCMC samples. Thus, it is claimed that the MCMC convergence of the standard Metropolis-Hastings algorithm has not been achieved.

On the other hand, by Figures 4.7–4.9 of autocorrelation functions, it is evident that the MCMC samples of all the parameters of interest are uncorrelated. Therefore, it is sufficient to claim that the MCMC convergence has been satisfied. As for the autocorrelation functions of the rest of the parameters, see Figures D.9–D.16 in Appendix D.

With the same setting of the convergence, pilot runs of the MCMC algorithms were performed for model selection. The number of hidden states was set to range from 2 to 5, namely $m \in \{2, 3, 4, 5\}$. For each competing model, the log marginal likelihood was estimated by Equation (B.8). For more details of model selection for multivariate NHGHMMs, see Appendix B. The results are shown in Table 4.3 and Figure 4.10.

Table 4.3: Summary of log marginal likelihoods for the Metropolis-Hastings (MH), AM, symmetric DRAM, and MTAM algorithms

	MH	AM	DRAM	MTAM
$m = 2$	$-1\,198.272$	$-1\,198.294$	$-1\,198.283$	$-1\,198.286$
$m = 3$	$-1\,170.641$	$-1\,170.489$	$-1\,170.581$	$-1\,170.462$
$m = 4$	$-1\,176.340$	$-1\,179.933$	$-1\,177.513$	$-1\,176.136$
$m = 5$	$-1\,254.376$	$-1\,255.862$	$-1\,259.930$	$-1\,252.532$

The maximum log marginal likelihood has been determined as the MTAM algorithm with the number of hidden states (i.e. $m = 3$). This result is consistent with the true simulated model.

4.6.2 Performance of Parameter Estimation

Considering the results of convergence assessment and model selection being satisfactory, it is now required to conduct a simulation study of all the proposed algorithms. The number of 1,500,000 MCMC iterations was run on each of 100 simulated data sets from Model (4.26). Burn-in time and the thinning were set identical to those of the single data set so that each parameter of interest comprises 10,000 MCMC iterations for statistical inference. The result of point estimates, 95% CrI and coverage of each

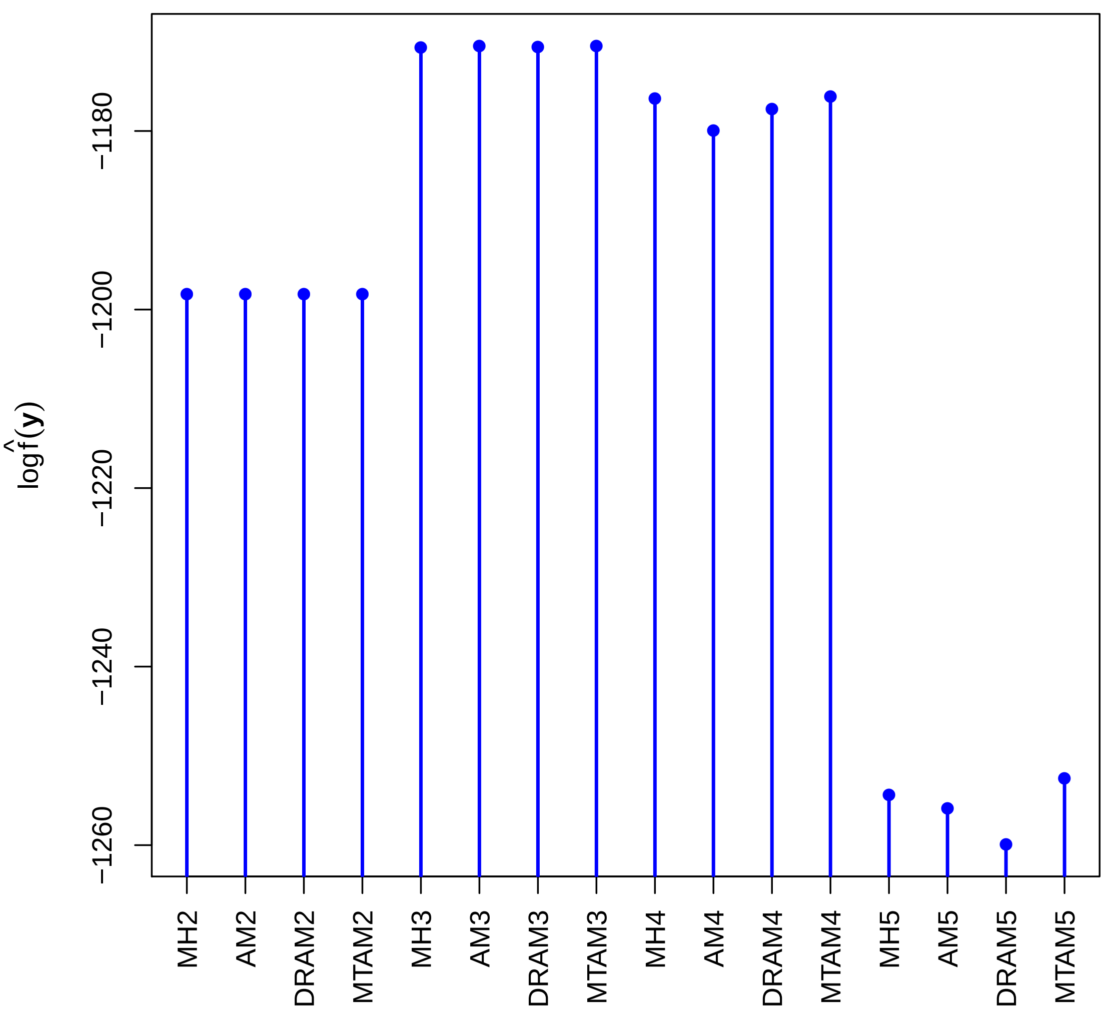


Figure 4.10: Bar plot of log marginal likelihoods for model selection; the suffix number of each algorithm in the plot corresponds to the number of hidden states.

parameter is shown in Table 4.4.

After completing the simulation study by taking arithmetic means of each parameter in 100 simulated data sets, the proposed algorithms consolidate the true values in terms of estimates, coverage and 95% CrI. All the three algorithms showed similar point estimates, 95% CrI, and coverage. Given that the number of observations was 300 for each simulated data set, all the algorithms demonstrated reasonable accuracy. The detailed findings for the AM, symmetric DRAM, and MTAM algorithms are as follows.

The AM algorithm was able to estimate the parameters of interest reasonably well although some parameters with regard to the Gibbs sampler showed relatively low cov-

Table 4.4: Summary statistics of estimates and coverage for random walk Metropolis-Hastings algorithm, AM, symmetric DRAM, and MTAM algorithms

Parameter	True value	AM			DRAM			MTAM		
		Estimate	95% CrI	Coverage	Estimate	95% CrI	Coverage	Estimate	95% CrI	Coverage
μ_1^1	1	1.070	(0.760, 1.381)	76%	1.025	(0.716, 1.334)	88%	1.087	(0.773, 1.401)	78%
μ_1^2	2	1.913	(1.651, 2.176)	80%	1.928	(1.666, 2.192)	83%	1.984	(1.723, 2.245)	85%
μ_2^1	3	3.022	(2.779, 3.265)	90%	3.006	(2.765, 3.249)	88%	3.007	(2.767, 3.249)	94%
μ_2^2	0	0.001	(−0.183, 0.196)	91%	−0.019	(−0.192, 0.154)	87%	−0.008	(−0.180, 0.165)	89%
μ_3^1	5	4.933	(4.611, 5.257)	92%	4.979	(4.663, 5.295)	93%	5.011	(4.690, 5.330)	89%
μ_3^2	4	3.901	(3.508, 4.287)	81%	3.970	(3.582, 4.359)	79%	3.959	(3.575, 4.346)	92%
$\Sigma_1^{1,1}$	1.5	1.614	(1.119, 2.297)	88%	1.625	(1.129, 2.310)	85%	1.674	(1.161, 2.379)	85%
$\Sigma_1^{1,2}$	0.5	0.456	(0.117, 0.869)	90%	0.511	(0.169, 0.931)	88%	0.514	(0.169, 0.938)	84%
$\Sigma_1^{2,1}$	0.5	0.456	(0.117, 0.869)	90%	0.511	(0.169, 0.931)	88%	0.514	(0.169, 0.938)	84%
$\Sigma_1^{2,2}$	1.0	1.134	(0.782, 1.617)	85%	1.150	(0.793, 1.640)	84%	1.145	(0.789, 1.631)	83%
$\Sigma_2^{1,1}$	2.0	2.013	(1.558, 2.596)	94%	2.047	(1.596, 2.609)	93%	1.965	(1.525, 2.514)	87%
$\Sigma_2^{1,2}$	0.6	0.607	(0.345, 0.922)	90%	0.630	(0.371, 0.934)	83%	0.577	(0.325, 0.872)	87%
$\Sigma_2^{2,1}$	0.6	0.607	(0.345, 0.922)	90%	0.630	(0.371, 0.934)	83%	0.577	(0.325, 0.872)	87%
$\Sigma_2^{2,2}$	1.0	1.041	(0.801, 1.352)	88%	1.032	(0.802, 1.322)	90%	0.993	(0.767, 1.277)	88%
$\Sigma_3^{1,1}$	1.5	1.622	(1.106, 2.335)	89%	1.485	(1.016, 2.148)	93%	1.584	(1.092, 2.277)	89%
$\Sigma_3^{1,2}$	−0.5	−0.419	(−0.956, 0.059)	86%	−0.452	(−0.989, 0.013)	95%	−0.461	(−1.011, 0.019)	88%
$\Sigma_3^{2,1}$	−0.5	−0.419	(−0.956, 0.059)	86%	−0.452	(−0.989, 0.013)	95%	−0.461	(−1.011, 0.019)	88%
$\Sigma_3^{2,2}$	2.0	2.087	(1.412, 3.026)	93%	2.120	(1.428, 3.097)	87%	2.194	(1.495, 3.171)	92%
$\alpha_{1,2}^1$	−1.109	−0.952	(−2.571, 0.592)	94%	−0.892	(−2.459, 0.604)	93%	−0.935	(−2.560, 0.623)	96%
$\alpha_{1,2}^2$	0.555	0.305	(−0.831, 1.424)	94%	0.348	(−0.707, 1.402)	90%	0.263	(−0.900, 1.393)	86%
$\alpha_{1,3}^1$	−3.039	−2.856	(−5.099, −0.956)	95%	−2.918	(−5.269, −0.951)	94%	−2.857	(−5.166, −0.927)	97%
$\alpha_{1,3}^2$	1.664	1.559	(0.405, 2.848)	92%	1.544	(0.362, 2.872)	95%	1.513	(0.339, 2.835)	95%
$\alpha_{2,1}^1$	−3.193	−3.211	(−4.578, −1.981)	95%	−3.279	(−4.665, −2.057)	91%	−3.167	(−4.567, −1.927)	93%
$\alpha_{2,1}^2$	1.485	1.502	(0.645, 2.389)	96%	1.555	(0.726, 2.435)	92%	1.481	(0.625, 2.384)	92%
$\alpha_{2,3}^1$	−2.361	−1.969	(−3.618, −0.247)	98%	−1.922	(−3.608, −0.117)	91%	−1.976	(−3.650, −0.183)	90%
$\alpha_{2,3}^2$	0.376	−0.245	(−2.063, 1.184)	87%	−0.351	(−2.291, 1.131)	85%	−0.184	(−2.103, 1.253)	90%
$\alpha_{3,1}^1$	−2.867	−2.773	(−5.134, −0.747)	97%	−2.701	(−5.055, −0.687)	98%	−2.540	(−4.931, −0.506)	91%
$\alpha_{3,1}^2$	1.433	1.328	(−0.081, 2.813)	95%	1.284	(−0.101, 2.749)	95%	1.113	(−0.309, 2.590)	93%
$\alpha_{3,2}^1$	−1.282	−1.127	(−2.676, 0.365)	96%	−1.279	(−2.910, 0.256)	90%	−1.148	(−2.744, 0.355)	96%
$\alpha_{3,2}^2$	0.785	0.678	(−0.463, 1.829)	89%	0.754	(−0.371, 1.916)	88%	0.675	(−0.410, 1.800)	92%
$q_{1,1}^{t_1}$	0.6	0.570	(0.393, 0.737)	89%	0.561	(0.383, 0.732)	90%	0.579	(0.399, 0.745)	98%
$q_{1,2}^{t_1}$	0.3	0.296	(0.151, 0.476)	91%	0.311	(0.161, 0.495)	92%	0.290	(0.144, 0.471)	95%
$q_{1,3}^{t_1}$	0.1	0.122	(0.040, 0.271)	96%	0.116	(0.036, 0.263)	96%	0.120	(0.038, 0.269)	97%
$q_{2,1}^{t_1}$	0.1	0.107	(0.053, 0.186)	88%	0.105	(0.051, 0.184)	93%	0.109	(0.052, 0.192)	95%
$q_{2,2}^{t_1}$	0.8	0.787	(0.690, 0.866)	93%	0.794	(0.698, 0.872)	92%	0.785	(0.684, 0.866)	92%
$q_{2,3}^{t_1}$	0.1	0.101	(0.049, 0.179)	95%	0.097	(0.046, 0.173)	96%	0.102	(0.049, 0.183)	97%
$q_{3,1}^{t_1}$	0.1	0.115	(0.037, 0.259)	94%	0.119	(0.036, 0.270)	97%	0.122	(0.038, 0.274)	94%
$q_{3,2}^{t_1}$	0.3	0.313	(0.164, 0.494)	98%	0.295	(0.149, 0.479)	94%	0.307	(0.157, 0.492)	96%
$q_{3,3}^{t_1}$	0.6	0.560	(0.385, 0.728)	94%	0.574	(0.393, 0.744)	93%	0.559	(0.379, 0.729)	95%
$q_{1,1}^{t_2}$	0.3	0.327	(0.187, 0.497)	88%	0.324	(0.182, 0.494)	95%	0.344	(0.195, 0.519)	87%
$q_{1,2}^{t_2}$	0.3	0.254	(0.128, 0.420)	82%	0.281	(0.149, 0.448)	85%	0.248	(0.120, 0.418)	86%
$q_{1,3}^{t_2}$	0.4	0.408	(0.248, 0.582)	91%	0.384	(0.232, 0.555)	89%	0.396	(0.236, 0.574)	92%
$q_{2,1}^{t_2}$	0.4	0.413	(0.251, 0.584)	94%	0.424	(0.269, 0.588)	91%	0.410	(0.249, 0.583)	92%
$q_{2,2}^{t_2}$	0.5	0.498	(0.334, 0.667)	97%	0.499	(0.339, 0.661)	90%	0.498	(0.331, 0.666)	90%
$q_{2,3}^{t_2}$	0.1	0.079	(0.026, 0.195)	79%	0.068	(0.020, 0.174)	79%	0.082	(0.025, 0.204)	84%
$q_{3,1}^{t_2}$	0.3	0.296	(0.144, 0.496)	81%	0.288	(0.134, 0.491)	88%	0.259	(0.114, 0.460)	90%
$q_{3,2}^{t_2}$	0.4	0.385	(0.211, 0.582)	81%	0.387	(0.207, 0.593)	88%	0.400	(0.216, 0.605)	88%
$q_{3,3}^{t_2}$	0.3	0.304	(0.152, 0.498)	86%	0.309	(0.152, 0.512)	94%	0.326	(0.163, 0.526)	91%

erage such as μ_1^1 and μ_1^2. On the other hand, estimates and coverage of the parameter α were relatively accurate. This means that the choice of the exogenous variables was adequate. Subsequently, estimation of the parameter Q^{t_1} was reasonably accurate although that of the transition probability Q^{t_2} was not as satisfactory as Q^{t_1}'s, especially $q_{2,3}^{t_2}$.

The symmetric DRAM algorithm also shows the similar result to that of the AM algorithm in terms of estimate and coverage; however, the Gibbs sampler has outperformed the AM algorithm's. In addition, every 95% credible interval includes the true values of the model. Likewise, coverage of the parameter $q_{2,3}^{t_2}$ was relatively poor.

Finally, the MTAM algorithm recorded the highest value of coverage in the parameter $q_{2,3}^{t_2}$, which was 84%, and thus, the MTAM algorithm is proven to be the most favourable algorithm in terms of statistical inference of the model. As for 95% CrI, all the parameters of interest in Model (4.26) are included within the intervals.

Trace plots and histograms of all the parameters of interest are shown by Figures D.17–D.46 in Appendix D.

With this simulation study, parameter estimation was completed with great accuracy for especially the parameters pertaining to the transition probabilities such as α, Q^{t_1}, and Q^{t_2}. That being said, it has implied that the choice of exogenous variables was appropriate, and hence, such an accurate result was obtained.

Compared to the simulation study of the univariate NHGHMM in Section 3.6, relatively small values of noise were selected for the parameters $\mathbf{\Sigma} = (\Sigma_1, \Sigma_2, \Sigma_3)$ for Model (4.26). Consequently, the MCMC samples were able to achieve higher values of coverage in the simulation study.

For a comparison of the proposed MCMC algorithms, Model (4.26) was used for a test model. The number of MCMC iterations were selected as $N = 500$ with replications of 100 for each of the algorithms. The results are shown in the following table.

Table 4.5: Comparison of the proposed MCMC algorithms for the multivariate case

Algorithm	No. of tries	Replications	Elapsed time (s)	Relative
AM	1	100	259.14	1.00
DRAM	10	100	1 276.00	4.92
DRAM	25	100	2 752.69	10.62
DRAM	50	100	5 195.16	20.05
MTAM	10	100	724.46	2.80
MTAM	25	100	1 453.94	5.61
MTAM	50	100	2 639.73	10.19

According to the table above, it was found to be more computationally expensive to run the algorithms in the multivariate setting than in the univariate setting. Overall, the symmetric DRAM algorithm is more computationally expensive than the MTAM algorithm with respect to the number of tries for each MCMC iteration. Moreover, the computational costs increase linearly as the number of tries increases for both the algorithms.

Finally, the computational time for each algorithm was 10.90 days for the AM algorithm, 30.39 days for the symmetric DRAM algorithm, and 30.08 days for the MTAM algorithm.

Despite the observations with small sample size of 300 for each simulated data set, the result for the multivariate model was considerably successful. Simulation study with 1,000 observations for each simulated data set may be considered and carried out to explore the validity of the algorithms.

4.7 Case Studies

Given that the successful simulation study of the multivariate NHGHMM, it remains to apply the model on real-world data sets. In this section, data sets of Icelandic river flow and rainfall in the UK were selected.

4.7.1 Icelandic river flow data

The multivariate NHGHMM was applied to the Icelandic daily river flow data from the 1st of January, 1972 to the 31st of December, 1974, where the year 1972 was a leap year. The daily river flow data with length of 1,096 were measured in cubic metres per second (m^3/s) at the two rivers, the Jökulsá and the Vatnsdalsá. In addition, daily precipitation (mm) and mean daily temperature (°C) over the same period were recorded in Hveravellir, and these measurements are used as a set of exogenous variables in this case study. The data are available at the the Hydrological Survey of the National Energy Authority of Iceland (`https://nea.is/the-national-energy-authority/map-servers/energy-resource-data-viewer/`).

The first statistical analysis of the data set was conducted by Tong et al. (1985). In their investigation, a threshold model was more adequate than conventional linear models to capture different hydrological characteristics of the rivers. The observed flow data of the two rivers were analysed individually.

Another investigation of the Icelandic river flow data was carried out by Tsay (1998). The model was designed to analyse the bivariate observed data jointly using the multivariate threshold autoregressive model. The residual analysis showed that the fitted model was appropriate, although several clusters of large residuals suggested there was the possibility of minor periodic behaviour in the river flow such as seasonal fluctuations (Tsay, 1998).

In this section, the multivariate NHGHMM was applied to analyse the flow data of the two rivers jointly. This model is considerably different from the threshold model applied by Tsay (1998).

According to Tsay (1998), a two-regime setting for the threshold model seemed to be feasible. Hence, the number of hidden states, where $m = 2$ is used for the case study in this section. Time series plots of the river flow, precipitation, and temperature data are shown in Figure 4.11.

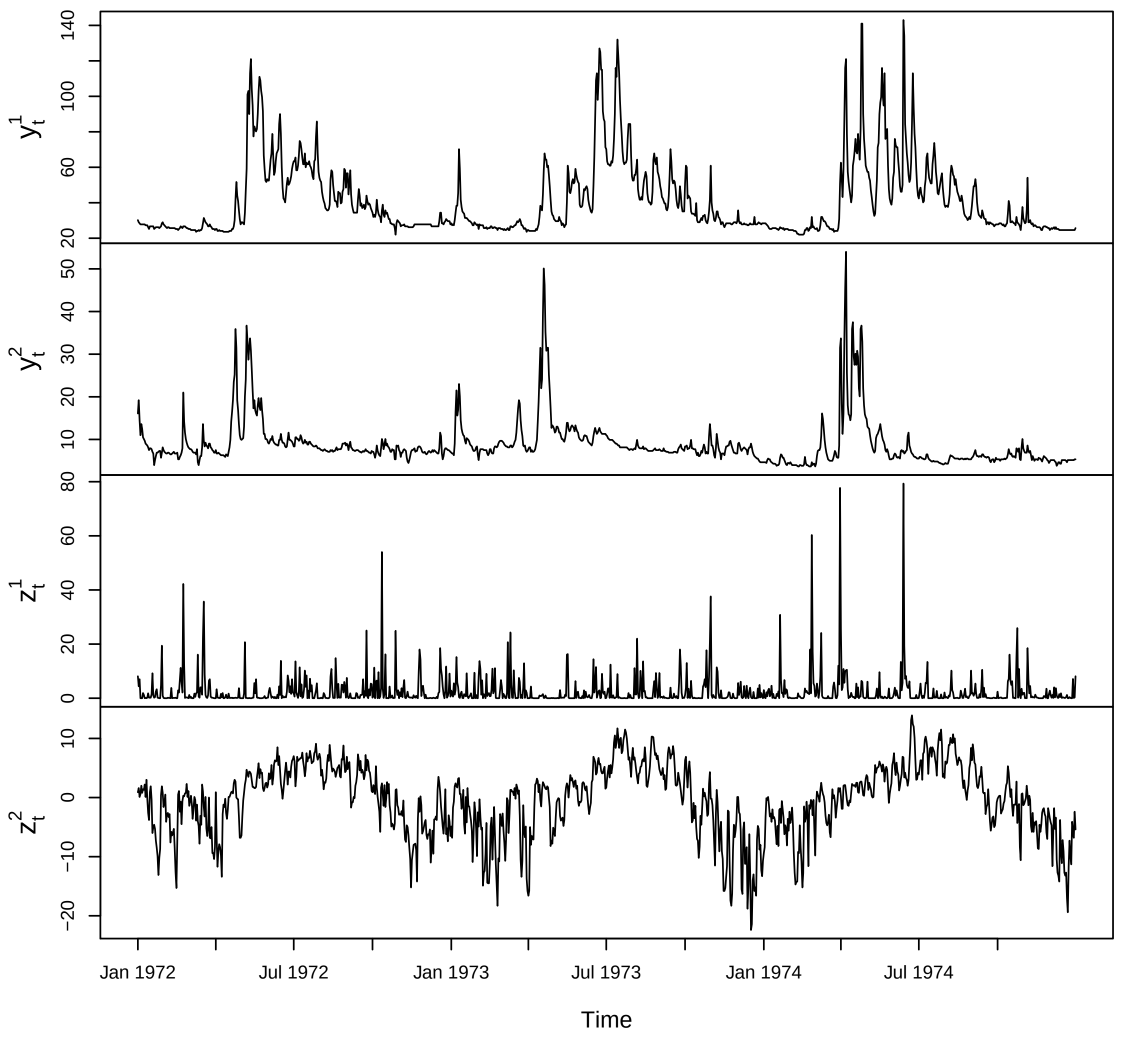


Figure 4.11: Time series plots of the Jökulsá river flow (top), Vatnsdalsá river flow (upper middle), precipitation (lower middle), and mean temperature (bottom)

Following Tong et al. (1985), the precipitation data are shifted forward by 1 due to the fact that the recorded values are the accumulated rain at 9:00 a.m. from the same time the day before. As a result, there are 1,095 observations for the precipitation after the shift.

For the case study, the following model was used. That is,

$$\mathbf{y}_t = \boldsymbol{\mu}_{s_t} + \boldsymbol{\varepsilon}_t,\ \boldsymbol{\varepsilon}_t \sim \mathcal{N}_2(\mathbf{0}, \Sigma_{s_t}), \tag{4.27}$$

where $\mathbf{y}_t = (y_t^1, y_t^2)$ denotes the bivariate observations at $t \in \{1, 2, \dots, 1096\}$, and $\mathbf{0} = (0, 0)$ is the 2-dimensional zero vector. In addition, y_t^1 indicates the Jökulsá river flow, and y_t^2 represents the Vatnsdalsá river flow.

Most importantly, the exogenous variables $z_t = (1, z_t^1, z_t^2)$, for any $t \in \{2, 3, \dots, 1096\}$, are also included implicitly in the model, where z_t^1 denotes the daily precipitation, and z_t^2 denotes the mean daily temperature.

The number of MCMC iterations was set to be $N = 350,000$ with the burn-in time of 50,000. Then, thinning of every 30th iteration was carried out so that the resulting 10,000 iterations were taken into account.

To ensure MCMC convergence of the parameters, especially $\alpha_{1,2}$ and $\alpha_{2,1}$, the MTAM algorithm was implemented with the number of multiple tries being 30.

Hereafter, the MCMC convergence assessments of the parameters were carried out. The results of Geweke's diagnostics and ESS are shown in the following table. As seen in Table 4.6, both Geweke's diagnostics and ESS of the parameters were satisfactory. To ensure the MCMC convergence of those parameters, the autocorrelation functions were also investigated and the results are shown below.

According to Figure 4.12, it is concluded that all the parameters of the model have converged since the dependencies are non-existent.

Provided that every convergence test was successful, parameter estimation of the parameters is then performed. The results are shown below.

There was no label switching phenomenon occurred during the MCMC iterations. Thus, relabelling the parameters was not required. Table 4.7 suggested that the observations, where $\{t \geq 1 : S_t = 2\}$ were pertaining to the higher mean river flow rate with

Table 4.6: Summary of Geweke's diagnostics and ESS of the parameters for the case study of Icelandic river flow data

Parameter	Geweke's diagnostics	ESS
μ_1^1	−0.520	10 000
μ_1^2	0.704	10 000
μ_2^1	1.781	10 000
μ_2^2	0.236	11 271
$\Sigma_1^{1,1}$	0.627	10 000
$\Sigma_1^{1,2}$	0.666	10 000
$\Sigma_1^{2,1}$	0.666	10 000
$\Sigma_1^{2,2}$	0.536	10 000
$\Sigma_2^{1,1}$	−0.044	10 348
$\Sigma_2^{1,2}$	0.989	10 000
$\Sigma_2^{2,1}$	0.989	10 000
$\Sigma_2^{2,2}$	−1.083	10 000
$\alpha_{1,2}^1$	−0.629	10 000
$\alpha_{1,2}^2$	−0.164	9 623
$\alpha_{1,2}^3$	0.890	10 000
$\alpha_{2,1}^1$	−1.089	10 000
$\alpha_{2,1}^2$	−0.871	10 000
$\alpha_{2,1}^3$	−0.786	10 000

Table 4.7: Summary of estimates and 95% CrI of the parameters of interest

Parameter	Estimate	95% CrI
μ_1^1	26.84	(26.65, 27.05)
μ_1^2	6.75	(6.60, 6.93)
μ_2^1	54.66	(52.78, 56.54)
μ_2^2	11.01	(10.43, 11.60)
$\Sigma_1^{1,1}$	4.50	(3.88, 5.26)
$\Sigma_1^{1,2}$	1.36	(0.99, 1.81)
$\Sigma_1^{2,1}$	1.36	(0.99, 1.81)
$\Sigma_1^{2,2}$	2.84	(2.42, 3.45)
$\Sigma_2^{1,1}$	473.60	(420.95, 532.74)
$\Sigma_2^{1,2}$	42.44	(29.30, 56.01)
$\Sigma_2^{2,1}$	42.44	(29.30, 56.01)
$\Sigma_2^{2,2}$	51.43	(45.52, 57.76)
$\alpha_{1,2}^1$	−3.25	(−4.44, −2.34)
$\alpha_{1,2}^2$	−0.06	(−0.27, 0.08)
$\alpha_{1,2}^3$	1.14	(0.70, 1.70)
$\alpha_{2,1}^1$	−3.15	(−3.86, −2.55)
$\alpha_{2,1}^2$	0.01	(−0.06, 0.06)
$\alpha_{2,1}^3$	−0.44	(−0.60, −0.29)

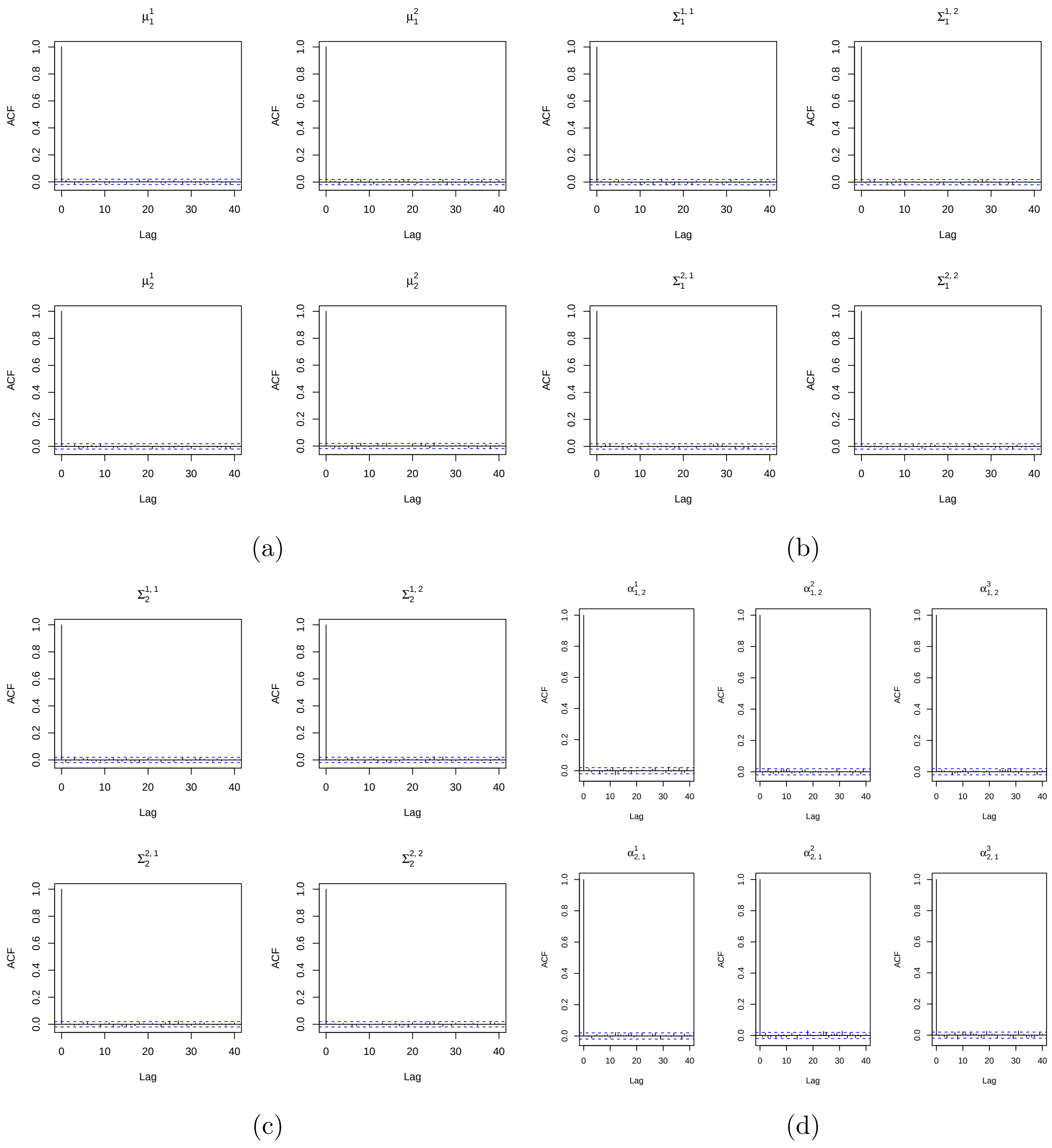


Figure 4.12: Autocorrelation functions of $\boldsymbol{\mu}$ (a), Σ_1 (b), Σ_2 (c), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (d)

more volatility in the river flow for both the rivers. Figures 4.13 and 4.14 represent trace plots and histograms of the parameters, respectively.

Except for the 95% CrI of the parameter, Σ_2, those of every other parameter indicated quite narrow credible intervals, and hence, great accuracy was achieved. More importantly, the 95% CrI of $\alpha_{1,2}$ and $\alpha_{2,1}$ were also narrow.

Here, the marginal posterior distributions are considered for the multivariate case. It is noted that the inference, however, is unable to be comprehensive with respect to correlations between each parameter.

Therefore, parameter estimation of the transition probabilities was quite satisfac-

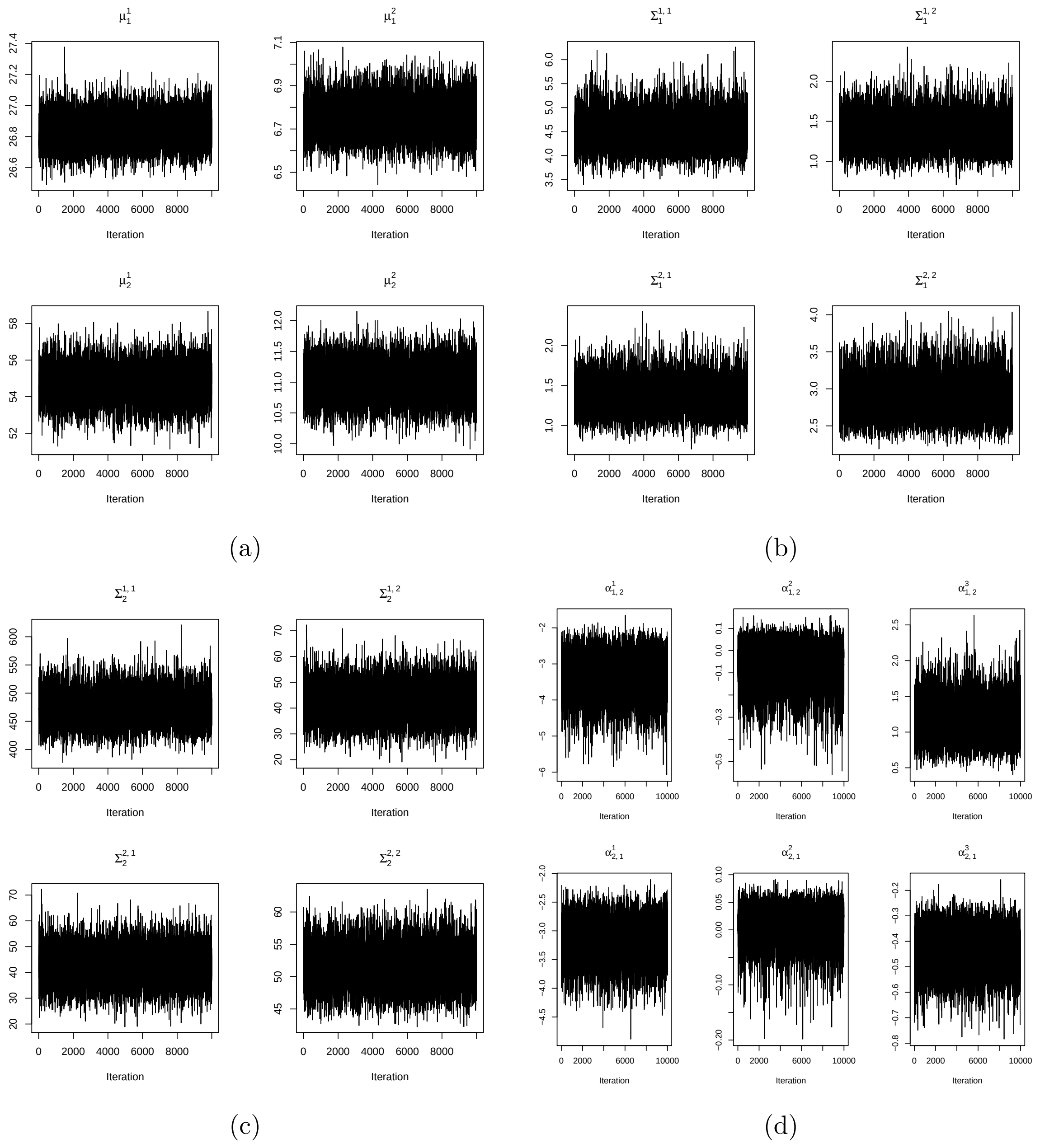
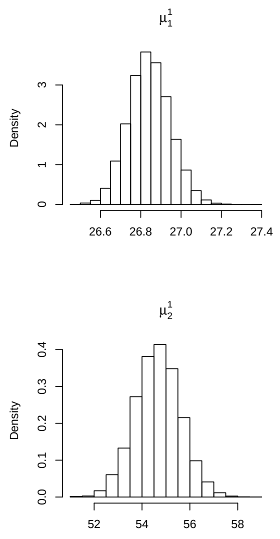


Figure 4.13: Trace plots of $\boldsymbol{\mu}$ (a), Σ_1 (b), Σ_2 (c), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (d)

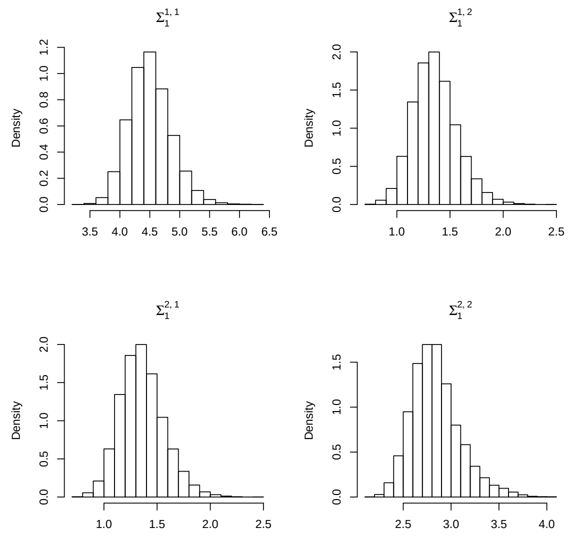
tory. The results of the transition probabilities are shown below.

By Figure 4.15, it is evident that periodic features are shown in the transition probabilities. For instance, transitions from the lower flow rate state to the higher flow rate state become more likely during the warm periods, roughly April through October each year.

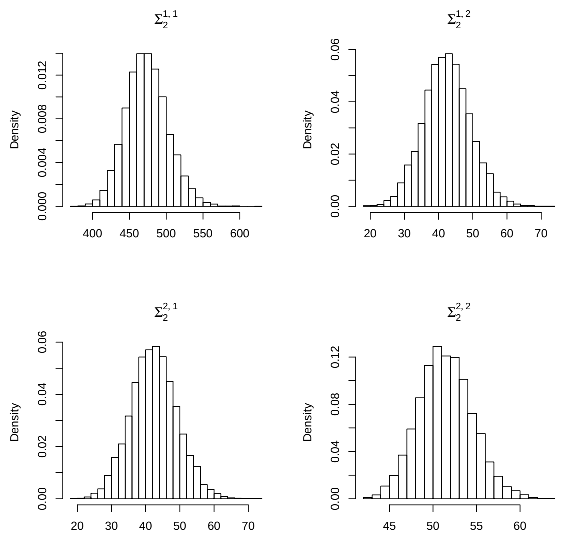
Due to this fact, the daily mean temperatures are a major factor in terms of the non-homogeneity. This reduces to the logical conclusion where higher mean temperature values are responsible for melting ice fields, and hence, more flow rates are induced in those rivers. Most importantly, the 95% CrI in Figure 4.15 indicate the great accuracy.



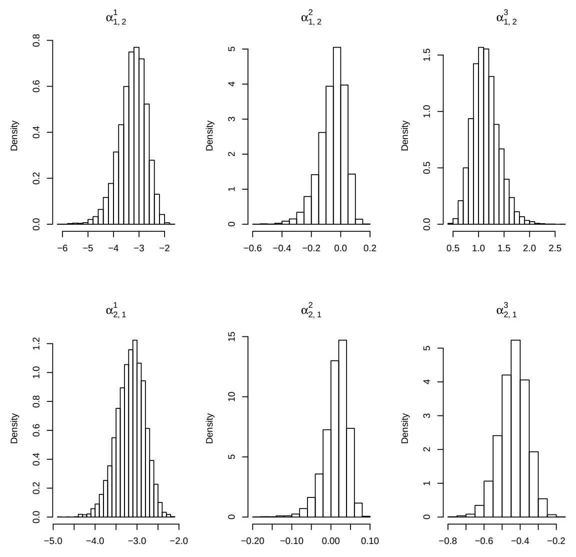
(a)



(b)



(c)



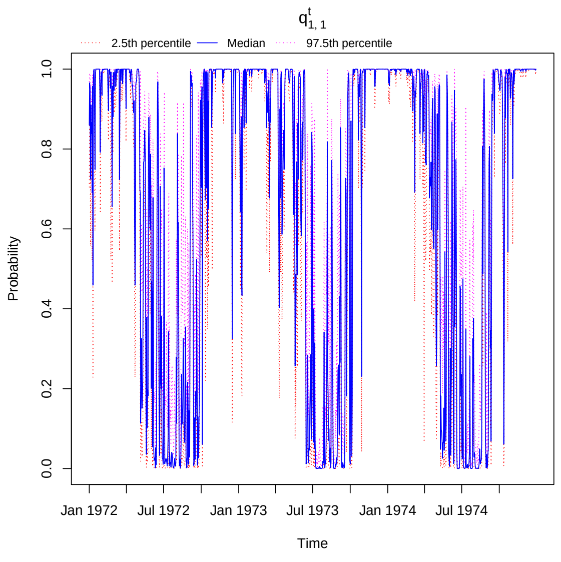
(d)

Figure 4.14: Histograms of $\boldsymbol{\mu}$ (a), Σ_1 (b), Σ_2 (c), and $\alpha_{1,2}$ & $\alpha_{2,1}$ (d)

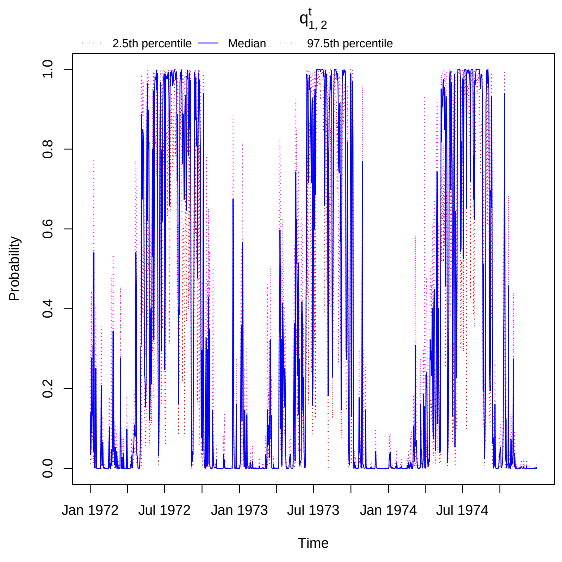
Finally, Figure 4.16 shows the estimated hidden state sequence where grey shaded areas indicate the time periods belonging to the higher river flow rates.

According to Figure 4.16, an interesting finding was observed where the higher river flow rates steadily took place, with several peaks, in the Jökulsá during the warm periods. Conversely, higher river flow rates occurred in the Vatnsdalsá in the early stage of the warm periods, although the relatively lower river flow rates were observed roughly after July each year. This may suggest that the residue of ice fields for the Vatnsdalsá was less than the Jökulsá whose source is the Vatnajökull glacier.[1]

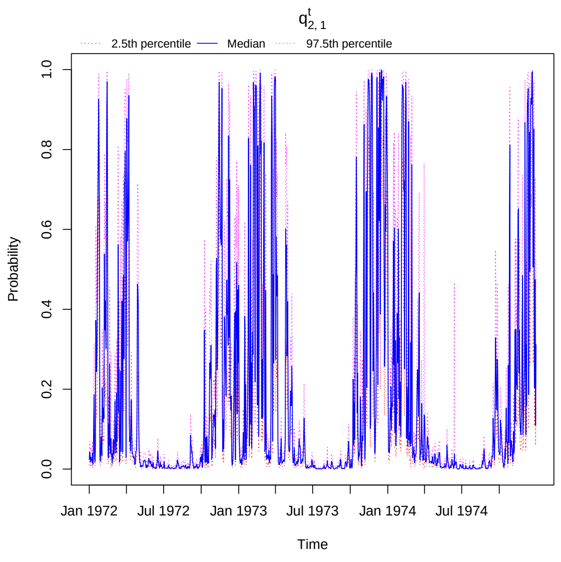
[1]Jökulsá literally means *glacial river* in Icelandic. The Vatnajökull glacier is the largest ice cap in



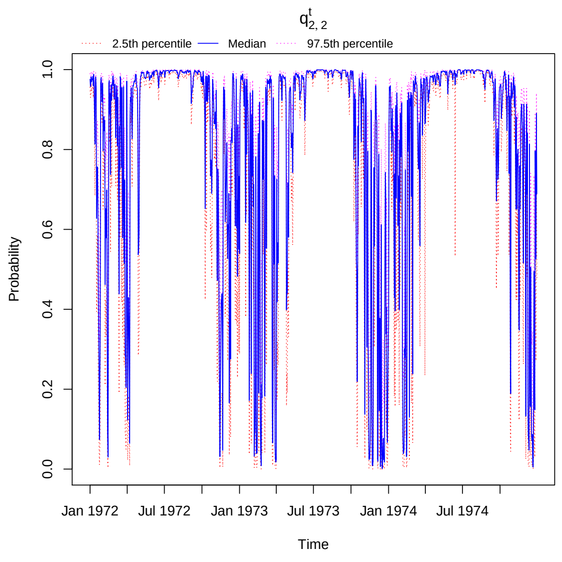
(a)



(b)



(c)



(d)

Figure 4.15: Trace plots of 95% CrI for $q^t_{1,1}$ (a), $q^t_{1,2}$ (b),$q^t_{2,1}$ (c), and $q^t_{2,2}$ (d) where $t \in \{2, 3, \ldots, 1096\}$

Therefore, the influence of the mean daily temperature values gets weaker after April.

The discussion about snow melting is consistent with that of Tsay (1998)'s.

Iceland.

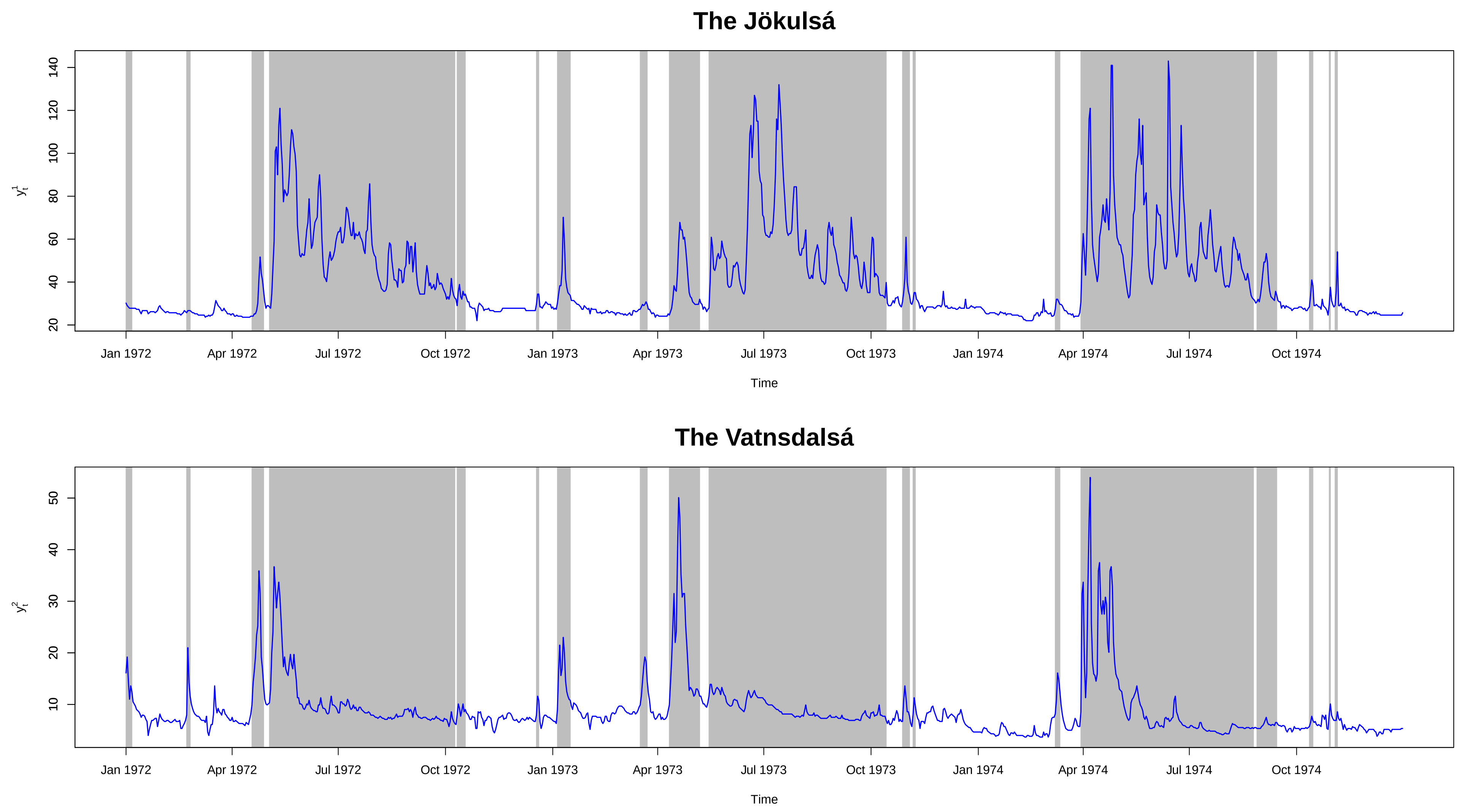


Figure 4.16: Time series plots of the Jökulsá (top) and the Vatnsdalsá (bottom). The grey shaded areas represent the time duration for which the hidden states are in State 2, namely the state with higher river flow rates.

4.7.2 Rainfall in the UK

The relationship between rainfall and climate has drawn much attention from climatologists to investigate the potential effects of climate change in recent years (Heaps et al., 2015). Conventional precipitation models are often referred to as circulation models. They are mathematical models of the circulation of the atmosphere. Nonetheless, the models often fail to capture local weather features due to the lack of fine resolutions in terms of longitude and latitude. To obtain much finer resolution, *statistical downscaling models* have been studied in hydrology and meteorology (Ailliot et al., 2009; Hughes et al., 1997, 1999).

Aggregated precipitation is in a mixed distribution, namely positively skewed density function on the positive real line $\mathbb{R}_{>0}$ with a point mass at zero. This attribute exacerbates a statistical analysis of models for precipitation which can accommodate spatiotemporal structure (Heaps et al., 2015). Hence, the development of statistical downscaling models has been faced with such a difficulty. Prior to Heaps et al. (2015)'s work, there had been no literature about a Bayesian approach to modelling precipitation by using *multivariate non-homogeneous Gaussian* HMMs (Heaps et al., 2015).

Bellone et al. (2000) extended an existing multivariate non-homogeneous hidden Markov model for precipitation occurrences to precipitation amounts. The observations at 24 rain gauge stations were assumed to be conditional on synoptic atmospheric patterns in Washington, the USA. For the parameter estimation, a frequentist approach was carried out by applying the EM algorithm (Bellone et al., 2000).

Since Heaps et al. (2015) formulated a Bayesian framework to a precipitation model using non-homogeneous HMMs, Holsclaw et al. (2016, 2017) also studied a Bayesian analysis of daily precipitation amounts.

Therefore, I present Bayesian inference of a multivariate NHGHMM for a precipitation model by using a set of the following exogenous variables:

(i) mean sea level pressure (MSLP),
(ii) 500 hPa geopotential height (GPH), and
(iii) geopotential thickness of 500–1000 hPa.

The data which are analysed in this case study are available from the Met Office integrated archive system MIDAS (http://catalogue.ceda.ac.uk). A summary of the data set is presented in Table 4.8. The period of precipitation amounts which were recorded was selected to be winter calendar periods. The specific 12 rainfall gauge stations were chosen to cover the UK, as observed in Figure 4.18. Before the analysis, the observed values of rainfall amounts were transformed by Box-Cox transform to generate Gaussian distributed random variables.

Table 4.8: Summary of data from the UK network of stations during calendar winter periods (December–February) from 1961/62 to 1988/89

	Station	County	Proportion wet days	Mean daily precipitation (mm)	Coefficient of variation
1	Starcross	Devon	53.5%	5.405	1.172
2	Faversham	Kent	50.9%	3.580	1.256
3	Wolterton Park	Norfolk	53.8%	3.447	1.129
4	Cheltenham	Gloucestershire	51.7%	3.877	1.320
5	Newtown Linford	Leicestershire	52.4%	3.470	1.149
6	Pen-Y-Ffridd	Gwynedd	62.4%	5.194	1.228
7	Bar Gap Farm	Durham	64.5%	5.384	1.231
8	Castle Kennedy	Wigtownshire	58.9%	6.083	1.049
9	Craibstone	Aberdeenshire	59.3%	3.826	1.387
10	Fortrose	Ross & Cromarty	52.5%	3.126	1.289
11	Toomebridge	Antrim	64.9%	3.919	1.129
12	Ardtalnaig	Perthshire	65.1%	6.527	1.215

A set of exogenous variables $z_t = (1, z_t^1, z_t^2, \dots, z_t^{21}) \in \mathbb{R}^{22}$ is defined as a 22-dimensional vector, for any $t \in \{1, 2, \dots, 2527\}$. The exogenous variables are comprised of comprehensive atmospheric data covering areas of North Atlantic Ocean, North Sea, in addition to the British Isles. Table 4.9 shows the details of the exogenous variables as follows.

Table 4.9: Atmospheric measures as exogenous variables for the precipitation model at daily discrete-time, t, during winter periods (December-February) from 1961/62 to 1988/89

Variable	Data type	Variable	Data type	Variable	Data type	Latitude	Longitude
z_t^1	MSLP	z_t^8	500 hPa GPH	z_t^{15}	Thickness	55°N	0°
z_t^2	MSLP	z_t^9	500 hPa GPH	z_t^{16}	Thickness	45°N	30°W
z_t^3	MSLP	z_t^{10}	500 hPa GPH	z_t^{17}	Thickness	50°N	20°W
z_t^4	MSLP	z_t^{11}	500 hPa GPH	z_t^{18}	Thickness	55°N	10°W
z_t^5	MSLP	z_t^{12}	500 hPa GPH	z_t^{19}	Thickness	60°N	0°
z_t^6	MSLP	z_t^{13}	500 hPa GPH	z_t^{20}	Thickness	60°N	10°E
z_t^7	MSLP	z_t^{14}	500 hPa GPH	z_t^{21}	Thickness	55°N	10°E

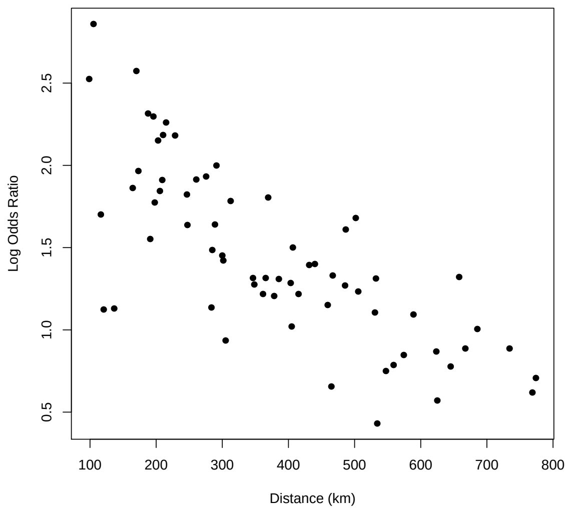
Prior to running the MCMC algorithm, preliminary analyses were conducted as

Heaps et al. (2015) attempted to reveal some statistical properties. Figure 4.17a shows the log odds ratios such that

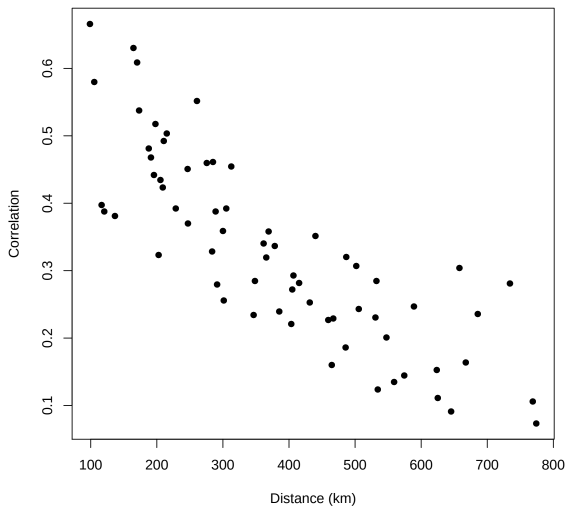
$$\log(\nu_{0,0}(i,j)\nu_{1,1}(i,j)) - \log(\nu_{0,1}(i,j)\nu_{1,0}(i,j)),$$

where $\nu_{d^i,d^j}(i,j)$ denotes the indicator function of observed number of days in which rainfall occurrence, d^i at station i and d^j at station j. Moreover, $d^k = 1$ whenever station k records rainfall amount of greater than or equal to 0.2 mm. Otherwise, $d^k = 0$.

By Figure 4.17, it is evident that both log odds ratio and correlation of those stations are negatively correlated to the distance between them. This is consistent with that result of Heaps et al. (2015).



(a)



(b)

Figure 4.17: Plots of log odds ratio (a), and correlation (b) against distance between the 12 stations

The main algorithm (Algorithm 12) using the MTAM algorithm (Algorithm 16) is now performed with the number of MCMC iterations, $N = 1,500,000$, and burn-in time of 1,000,000. Then, thinning of every 50th iteration is conducted so that the total 10,000 iterations are taken into account for the statistical analysis. As for the number of hidden states, Heaps et al. (2015) discovered that $m = 4$ is the most adequate number, which was supported by the continuous ranked probability score of their results. Hence, the same number such that $m = 4$ was chosen. The result of the convergence assessment is shown below.

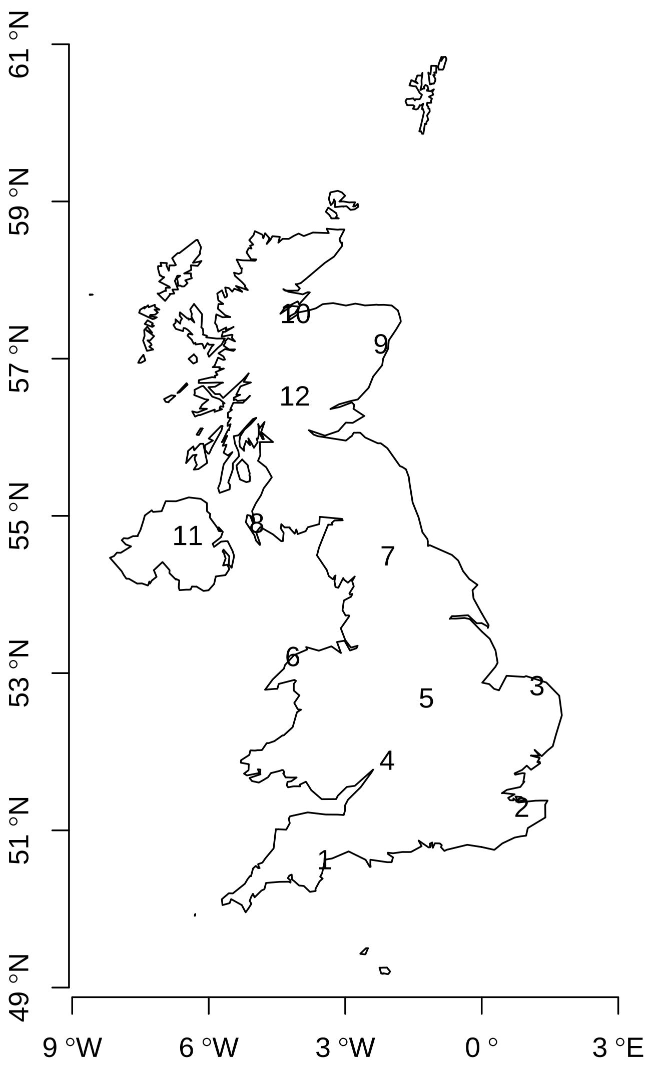


Figure 4.18: Map of the United Kingdom and the Channel Islands. Each number on the map corresponds to each station in Table 4.8.

The parameters listed in Table 4.10 are the ones which failed to pass Geweke's diagnostics. It is evident that the ESS of several parameters such as $\alpha^4_{3,1}$, $\alpha^5_{3,1}$, and so on, are quite small. As observed, there are numerous parameters of interest failing to pass the test, and hence, the convergence was unable to be met. The Bayesian inference of this case study is concluded to be inapplicable. Therefore, the model for this data set has reached its limitation.

One of the possible explanations to the model's limitations is the dimension of exogenous variables is too large and so is the length of the observations. The model

Table 4.10: Summary of Geweke's diagnostics and ESS for the parameters of interest which failed to pass the tests

Parameter	Geweke's diagnostics	ESS	Parameter	Geweke's diagnostics	ESS
μ_3^1	2.025	6 915	$\alpha_{1,2}^{19}$	−2.064	6 910
μ_2^4	−2.471	10 000	$\alpha_{1,3}^{4}$	2.371	6 311
μ_3^8	2.377	6 722	$\alpha_{1,4}^{5}$	2.344	7 041
μ_4^8	−2.884	10 000	$\alpha_{1,4}^{18}$	−3.306	6 677
μ_2^{10}	−3.225	10 000	$\alpha_{2,3}^{14}$	2.295	6 061
μ_4^{11}	−2.570	9 125	$\alpha_{2,4}^{8}$	2.513	7 025
$\Sigma_1^{7,2}$	−2.332	10 000	$\alpha_{2,4}^{18}$	−2.882	6 754
$\Sigma_1^{2,7}$	−2.332	10 000	$\alpha_{2,4}^{21}$	2.508	6 122
$\Sigma_2^{6,1}$	−2.021	9 657	$\alpha_{3,1}^{4}$	2.083	358
$\Sigma_2^{11,1}$	−3.249	10 000	$\alpha_{3,1}^{5}$	−2.260	318
$\Sigma_2^{8,2}$	2.227	10 000	$\alpha_{3,1}^{6}$	−3.383	252
$\Sigma_2^{8,3}$	2.164	10 912	$\alpha_{3,1}^{11}$	−4.746	220
$\Sigma_2^{12,3}$	2.076	10 000	$\alpha_{3,1}^{12}$	2.321	267
$\Sigma_2^{4,4}$	2.225	10 000	$\alpha_{3,1}^{17}$	2.584	301
$\Sigma_2^{11,5}$	2.545	10 000	$\alpha_{3,1}^{22}$	−4.538	260
$\Sigma_2^{1,6}$	−2.021	9 657	$\alpha_{3,2}^{2}$	−4.155	293
$\Sigma_2^{2,8}$	2.227	10 000	$\alpha_{3,2}^{11}$	−3.762	259
$\Sigma_2^{3,8}$	2.164	10 912	$\alpha_{3,2}^{12}$	2.556	293
$\Sigma_2^{11,9}$	2.567	10 000	$\alpha_{3,2}^{19}$	2.657	306
$\Sigma_2^{12,9}$	2.425	10 030	$\alpha_{3,2}^{21}$	2.014	326
$\Sigma_2^{10,10}$	−2.909	10 000	$\alpha_{3,2}^{22}$	−3.670	238
$\Sigma_2^{1,11}$	−3.249	10 000	$\alpha_{3,4}^{1}$	3.629	321
$\Sigma_2^{5,11}$	2.545	10 000	$\alpha_{3,4}^{4}$	−2.847	305
$\Sigma_2^{9,11}$	2.567	10 000	$\alpha_{3,4}^{6}$	−2.677	267
$\Sigma_2^{3,12}$	2.076	10 000	$\alpha_{3,4}^{19}$	2.928	269
$\Sigma_2^{9,12}$	2.425	10 030	$\alpha_{3,4}^{20}$	3.893	283
$\Sigma_3^{4,1}$	3.169	10 000	$\alpha_{3,4}^{22}$	−5.016	258
$\Sigma_3^{1,4}$	3.169	10 000	$\alpha_{4,1}^{1}$	−2.025	240
$\Sigma_3^{11,7}$	−2.584	10 000	$\alpha_{4,1}^{10}$	−1.973	255
$\Sigma_3^{7,11}$	−2.584	10 000	$\alpha_{4,1}^{11}$	2.537	208
$\Sigma_3^{12,12}$	−2.069	9 606	$\alpha_{4,1}^{16}$	−3.747	225
$\Sigma_4^{11,1}$	2.688	10 000	$\alpha_{4,1}^{19}$	2.242	233
$\Sigma_4^{7,3}$	−2.419	9 442	$\alpha_{4,1}^{22}$	2.209	253
$\Sigma_4^{3,7}$	−2.419	9 442	$\alpha_{4,2}^{4}$	−3.443	215
$\Sigma_4^{7,7}$	−2.135	8 994	$\alpha_{4,2}^{5}$	−2.220	212
$\Sigma_4^{12,7}$	−2.713	9 446	$\alpha_{4,2}^{12}$	2.237	139
$\Sigma_4^{1,11}$	2.688	10 000	$\alpha_{4,2}^{13}$	2.242	159
$\Sigma_4^{7,12}$	−2.713	9 446	$\alpha_{4,3}^{2}$	−2.002	252
			$\alpha_{4,3}^{19}$	3.008	438

simply is unable to estimate the parameters of interest well since floating point underflow will be caused by such a large number of samples in Equation (4.10) during MCMC sampling of the parameter, α. That is,

$$\prod_{\{t \geq 2:\, s_{t-1}=i \,\wedge\, s_t=j\}} q_{i,j}^t \to 0 \text{ as } \#\{t \geq 2 :\; s_{t-1} = i \,\wedge\, s_t = j\} \to n_0,$$

where $n_0 \in \mathbb{N}$ is sufficiently large, and $q_{i,j}^t < 1$ for any $\{(i, j) \in \Omega^2 :\; i \neq j\}$.

Chapter 5

Conclusion

The main objective of this thesis was to develop and implement the proposed MCMC algorithms for parameter estimation of both univariate and multivariate NHGHMMs. The primary motivation of proposing advanced MCMC algorithms was to improve the acceptance rate of Metropolis-Hastings type algorithms in the process of estimating the parameters in the models.

After the literature review of this thesis, Chapters 3 and 4 were presented for univariate and multivariate settings, respectively. Each chapter consisted of the prior specification, derivation of joint and conditional posterior distributions of the parameters, implementation of MCMC algorithms, and simulation & case studies.

In a Bayesian framework, it is necessary to specify prior distributions of parameters of interest. As for a univariate NHGHMM, the prior specification of Spezia (2006) was followed.

The theoretical component of the univariate NHGHMM differs from that of an ordinary homogeneous GHMM in terms of transition probabilities. The NHGHMM relaxes the restriction of constant probabilities by introducing a set of exogenous variables which are considered to affect the observed variable. Therefore, the resulting posterior distribution of the parameter, α, was in a non-standard form, and the Metropolis-Hastings type algorithms were necessary for parameter estimation.

Likewise, the parameter estimation of the multivariate NHGHMM was conducted by extending the univariate case. Since the dimension of the hidden states is the same as that of the univariate NHGHMM, the posterior distribution of the parameter α was also in a non-standard form. Hence, the proposed MCMC algorithms implemented earlier in Chapter 3 preserved the similar algorithmic scheme.

The proposed MCMC algorithms generally outperformed the Metropolis-Hastings algorithm in the univariate NHGHMM as the convergence assessment confirms. Even with the simulated data of length $n = 300$ in the simulation study, a set of MCMC convergence tests such as Geweke's diagnostics, ESS, and autocorrelation functions proved that the advanced MCMC algorithms indeed improve mixing.

The advanced MCMC algorithms also proved to be useful when it comes to improving mixing of the MCMC chains in the multivariate NHGHMM. In this setting, the nature of the model especially exacerbated the mixing of the MCMC samples via the standard random walk Metropolis-Hastings algorithm, even with thinning. On the other hand, the proposed MCMC algorithms were able to obtain satisfactory convergence assessment through the valid representative MCMC samples of the marginal posterior distributions.

The performances of the proposed algorithms, for both univariate and multivariate settings, were thoroughly investigated in terms of MCMC convergence and parameter estimation. The MCMC convergence tests, namely Geweke's diagnostics and autocorrelation functions of the parameters, are used in the simulation studies. In addition, model selection for the simulated models was also performed. As a Bayesian approach was adopted in this thesis, point estimates, 95% CrI, and coverage of the parameters were presented for the simulation studies. As for the case studies, the real-world data sets were used for practical applications after the MCMC convergence was achieved.

Furthermore, the log marginal likelihoods for both univariate and multivariate NHGHMMs confirm the number of hidden states in the model selection. Amongst all the competing models with different MCMC algorithms, the MTAM algorithm proved to be the most successful algorithm in estimating the number of hidden states in the simulation studies under both univariate and multivariate settings.

Ultimately, I made some observations through running the algorithms for each of the NHGHMMs as follows:

(i) The development of the more advanced MCMC algorithms on the models was successful regarding ease of implementation.

(ii) Autocorrelation functions and ESS of the parameters can be improved jointly by replacing an MCMC algorithm with more advanced ones.

(iii) The choice of exogenous variables was unsuitable for the models, and this suggests that some practical applications are required to transform the exogenous variables into feasible values before parameter estimation.

More detailed findings are listed as follows.

As for the univariate NHGHMM, the simulation study was conducted. The performance of each MCMC algorithm, namely the AM, symmetric DRAM, and MTAM algorithms, was found to be reasonable even though the sample size of the simulated data sets was relatively small. It was also observed that there was a successive improvement for the autocorrelation functions and ESS of the parameters.

The MTAM algorithm was implemented for parameter estimation of the differenced monthly US 3-month treasury bill rates from January 1973 to December 1999 since the algorithm is more efficient than any other proposed MCMC algorithm. Since the dimension of the exogenous variables was much larger than that of the simulation study, it was necessary for the MCMC algorithm to be run much longer. That being said, it is certain that the convergence was met under the model. Hence, the two hidden states, namely small and large fluctuation periods, were identified although lengths of those periods differ from Meligkotsidou and Dellaportas (2011)'s work since the random effect was not considered in this thesis.

In order to validate the proposed MCMC algorithms, an extensive simulation study for the multivariate NHGHMM was also conducted prior to a case study. In spite of the multivariate observation, the non-homogeneity preserves the mathematical structure of the univariate NHGHMM. Hence, the AM, symmetric DRAM, and MTAM algorithms were implemented without any modifications to them.

The data set of the Icelandic river flow was analysed in Section 4.7.1. The multivariate NHGHMM was able to capture the periodic characteristics in terms of transition probabilities. It was also found that the major factor for transitions between the hidden states to occur was daily mean temperatures.

In another case study of the multivariate NHGHMM, the rainfall data set was analysed in Section 4.7.2 in a Bayesian framework. The convergence assessment, however, showed that the MCMC chains of the parameters of interest failed to converge jointly. This leads to the conclusion where the multivariate NHGHMM was unfitted for the UK rainfall data set. As Heaps et al. (2015) pointed out, a combination of rainfall occurrences (Bernoulli variables) and rainfall amounts (Gaussian variables) is able to make the statistical analysis possible to conduct. Hence, more sophisticated models such as Heaps et al. (2015)'s are to be part of my future work.

As observed the simulation study in Chapter 3, the proposed MCMC algorithms with a Bayesian approach may outperform frequentist methods where sample size is considerably small. One of the drawbacks, however, is considered to be the incapability of handling huge sample size of data sets due to the possibility of inducing arithmetic underflow which has been seen in the case study of rainfall in the UK.

A set of normal R© code was faced with lengthy computational time during the course for my research. Hence, it was necessary to construct a set of C++ code within 'Rcpp' package developed by Eddelbuettel et al. (2011) to accelerate the computational time, as well as running MCMC iterations long enough to achieve convergence.

All the proposed MCMC algorithms showed that they were able to converge to the true values. In addition, a successive improvement of the autocorrelation functions and ESS of the parameters was observed.

Consequently, the aims of this thesis were mostly met. As a by-product, a set of C++ code was the most useful tool in both simulation and case studies. It certainly accelerated the required computational tasks. Without it, this research would have been impossible to complete.

Some aspects in statistical inference were disregarded in this thesis. Thus, it follows that the limitations of this research may be caused by the following perspectives such as:

- Apart from Geweke's diagnostics and autocorrelation functions, other possible MCMC convergence tests such as Heidelberger-Welch's diagnostics (Heidelberger and Welch, 1983) could be used.

- Similarly, only the ESS was used for measuring the efficiencies of the algorithms. Some other possible methods such as relative numerical efficiency (Geweke et al., 1991) might be implemented.
- To measure performances of parameter estimation, point estimates and 95% CrI were used. This may be extended to using the deviance information criterion, root-mean-square error or some other measures.
- The proposed MCMC algorithms have worked well in the specific simulation study of the multivariate NHGHMM where the values of $\mathbf{\Sigma}$ were relatively small. A comprehensive simulation study may be taken into account to ensure that the model works well with various values of $\mathbf{\Sigma}$.

With that being said, possible future work from this thesis may consist of:

(a) Comprehensive prior sensitivity analyses for both univariate and multivariate NHGHMMs.

(b) A fully Bayesian analysis in both univariate and multivariate settings where the number of hidden states is unknown.

(c) Different distributions of observations (i.e. exponential or Dirichlet distributions) for non-homogeneous hidden Markov models.

The theoretical component of the NHGHMM also has the potential to improve as Heaps et al. (2015) founded the framework of the multivariate NHGHMM to be able to model the rainfall data set which is usually quite difficult for this type of statistical inference.

Given that the non-homogeneity of transition probabilities is far superior to constant transition probabilities in terms of predictive ability, the NHGHMM has great applicability for predictive modelling.

Appendix A

Unknown Number of Hidden States: Univariate NHGHMM

Let $\theta^* = (\mu^*, \lambda^*, \alpha^*)$ be maximum a posteriori estimates of the joint posterior distribution. To evaluate a Bayes factor for each competing model in regards to the number of hidden states, m, it is practically required to compute the log marginal likelihood and is given as follows:

$$\log \hat{f}(y) = \log f(y \mid \theta^*) + \log p(\theta^*) - \log \hat{\pi}(\theta^* \mid y), \tag{A.1}$$

where $\hat{\pi}(\theta^* \mid y)$ represents a posterior estimate at θ^* realised by the MCMC samples from the simulation of conditional posterior distributions (i.e. either the Gibbs sampler or Metropolis-Hastings algorithm). Equation (A.1) enables the MCMC samples to be directly used for estimating the term, $\hat{\pi}(\theta^* \mid y)$, since a marginal likelihood is calculated by integrating a likelihood with respect to a prior instead of a posterior distribution (Chib and Jeliazkov, 2001).

The following shows how to decompose each term of the right hand side in Equation (A.1) so that the log marginal likelihood can be calculated.

The log likelihood

Since $y = (y_1, y_2, \dots, y_n)$ is a set of pairwise discrete-time observations, it follows that

$$\begin{aligned}
\log f(y \,|\, \theta^*) &= \log f(y_1, y_2, \dots, y_n \,|\, \theta^*) \\
&= \log(f(y_n \,|\, y_{1:n-1}, \theta^*) f(y_{n-1} \,|\, y_{1:n-2}, \theta^*) \cdots f(y_2 \,|\, y_1, \theta^*) f(y_1 \,|\, \theta^*)) \\
&= \log\left(\prod_{t=1}^{n} f(y_t \,|\, y_{1:t-1}, \theta^*)\right) \\
&= \sum_{t=1}^{n} \log f(y_t \,|\, y_{1:t-1}, \theta^*) \\
&= \sum_{t=1}^{n} \log\left(\sum_{i=1}^{m} f(y_t \,|\, y_{1:t-1}, S_t = i, \theta^*) f(S_t = i \,|\, y_{1:t-1}, \theta^*)\right) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{(by the law of total probability)} \\
&= \sum_{t=1}^{n} \log\left(\sum_{i=1}^{m} \underbrace{f(y_t \,|\, S_t = i, \mu^*, \lambda^*, \alpha^*)}_{\text{the Gaussian density}} \underbrace{\Pr(S_t = i \,|\, y_{1:t-1}, \mu^*, \lambda^*, \alpha^*)}_{\text{filtered probability}}\right).
\end{aligned} \tag{A.2}$$

The observations, $\{y_t\}$, are conditionally independent for all $t \in \{1, 2, \dots, n\}$. The filtered probability for each state $i \in \Omega$ at time $t \in \{1, 2, \dots, n\}$ is available from by-products of the forward filtering backward sampling algorithm.

The log priors

Since each prior is independent of the others, it follows that

$$\begin{aligned}
\log p(\theta^*) &= \log p(\mu^*, \lambda^*, \alpha^*) \\
&= \log(p(\mu^*) p(\lambda^*) p(\alpha^*)) \\
&= \log p(\mu^*) + \log p(\lambda^*) + \log p(\alpha^*) \\
&= \log\left(\prod_{i=1}^{m} p(\mu_i^*)\right) + \log\left(\prod_{i=1}^{m} p(\lambda_i^*)\right) + \log\left(\prod_{i=1}^{m} \prod_{\substack{j=1 \\ j\neq i}}^{m} p(\alpha_{i,j}^*)\right) \\
&= \sum_{i=1}^{m} \log p(\mu_i^*) + \sum_{i=1}^{m} \log p(\lambda_i^*) + \sum_{i=1}^{m} \sum_{\substack{j=1 \\ j\neq i}}^{m} \log p(\alpha_{i,j}^*).
\end{aligned} \tag{A.3}$$

The log posteriors

Following (Chib, 1995), the maximum a posteriori estimate, $(\mu^*, \lambda^*, \alpha^*)$, is selected for more accurate density estimation from which the fact that a high density point is attributed to the point, $(\mu^*, \lambda^*, \alpha^*)$. As such, the log posterior distribution is given by

$$\begin{aligned}
&\log \hat{\pi}(\theta^* \,|\, y) \\
&= \log \hat{\pi}(\mu^*, \lambda^*, \alpha^* \,|\, y) \\
&= \log(\hat{\pi}(\mu^* \,|\, y, \lambda^*, \alpha^*)\hat{\pi}(\lambda^* \,|\, y, \alpha^*)\hat{\pi}(\alpha^* \,|\, y)) \\
&= \log\left(N^{-1}\sum_{\tau=1}^{N}\prod_{i=1}^{m}\pi(\mu_i^* \,|\, y, \lambda^*, \alpha^*, s^{(\tau)})\right) + \log\left(N^{-1}\sum_{\tau=1}^{N}\prod_{i=1}^{m}\pi(\lambda_i^* \,|\, y, \alpha^*, \mu^{(\tau)}, s^{(\tau)})\right) \\
&\qquad + \log \hat{\pi}(\alpha^* \,|\, y).
\end{aligned} \tag{A.4}$$

For $\log \hat{\pi}(\alpha^* \,|\, y)$, the MCMC samples from the full conditional posterior distribution are Metropolis-Hastings updates, and thus, the log posterior distribution has a different form than those of the full conditional posterior distributions of parameters of interest (Chib and Jeliazkov, 2001).

Recall that the detailed balance condition between a new point, θ', and a current point, θ, is assumed. Then,

$$Q(\theta' \,|\, \theta)\pi(\theta) = Q(\theta \,|\, \theta')\pi(\theta'),$$

where $Q(\theta' \,|\, \theta)$ denotes the transition kernel from the current point, θ, to the proposed point, θ'. Define $Q(\theta' \,|\, \theta) = a(\theta' \,|\, \theta)q(\theta' \,|\, \theta)$, where $a(\theta' \,|\, \theta)$ denotes the acceptance probability from θ to θ', and $q(\theta' \,|\, \theta)$ denotes the proposal density for the random walk Metropolis-Hastings move from θ to θ'.

For the log posterior distribution of α^* given the observed data, y, the following expression is obtained such that

$$Q(\alpha^* \,|\, \alpha)\pi(\alpha \,|\, y) = Q(\alpha \,|\, \alpha^*)\pi(\alpha^* \,|\, y). \tag{A.5}$$

Then, integrate Equation (A.5) with respect to α. It follows that

$$\begin{aligned}\int Q(\alpha^* \,|\, \alpha)\pi(\alpha \,|\, y)\,\mathrm{d}\alpha &= \int Q(\alpha \,|\, \alpha^*)\pi(\alpha^* \,|\, y)\,\mathrm{d}\alpha \\ \Leftrightarrow \pi(\alpha^* \,|\, y)\int a(\alpha \,|\, \alpha^*)q(\alpha \,|\, \alpha^*)\,\mathrm{d}\alpha &= \int a(\alpha^* \,|\, \alpha)q(\alpha^* \,|\, \alpha)\pi(\alpha \,|\, y)\,\mathrm{d}\alpha \\ \Leftrightarrow \pi(\alpha^* \,|\, y) = \frac{\int a(\alpha^* \,|\, \alpha)q(\alpha^* \,|\, \alpha)\pi(\alpha \,|\, y)\,\mathrm{d}\alpha}{\int a(\alpha \,|\, \alpha^*)q(\alpha \,|\, \alpha^*)\,\mathrm{d}\alpha} &= \frac{\mathrm{E}_{\pi(\alpha \,|\, y)}[a(\alpha^* \,|\, \alpha)q(\alpha^* \,|\, \alpha)]}{\mathrm{E}_{q(\alpha \,|\, \alpha^*)}[a(\alpha \,|\, \alpha^*)]}. \end{aligned} \tag{A.6}$$

Using the Monte Carlo method, Equation (A.6) can be estimated as follows:

$$\pi(\alpha^* \,|\, y) \approx \hat{\pi}(\alpha^* \,|\, y) = \frac{N_1^{-1}\sum_{\tau=1}^{N_1} a(\alpha^* \,|\, \alpha^{(\tau)})q(\alpha^* \,|\, \alpha^{(\tau)})}{N_2^{-1}\sum_{\nu=1}^{N_2} a(\alpha^{(\nu)} \,|\, \alpha^*)},$$

where $\{\alpha^{(\tau)}\}$ is a set of realisations from the full conditional posterior distribution, $\pi(\alpha \,|\, y)$, and $\{\alpha^{(\nu)}\}$ is a set of realisations from the proposal distribution, $q(\alpha \,|\, \alpha^*)$. Normally, $N_1 \geq N_2$ since the time required for convergence of samples from the proposal distribution is less in comparison to the MCMC samples from the full conditional posterior distribution (Chib and Jeliazkov, 2001). Nevertheless, the condition such that $N_1 = N_2 = N$ is considered in this thesis. Moreover, for α_i where $\forall i \in \Omega$, it follows that

$$\hat{\pi}(\alpha_i^* \,|\, y) = \frac{N^{-1}\sum_{\tau=1}^{N}\prod_{i=1}^{m}\left(a(\alpha_i^* \,|\, \alpha_i^{(\tau)})\prod_{\substack{j=1\\ j\neq i}}^{m} q(\alpha_{i,j}^* \,|\, \alpha_{i,j}^{(\tau)})\right)}{N^{-1}\sum_{\tau=1}^{N}\prod_{i=1}^{m} a(\alpha_i^{(\tau)} \,|\, \alpha_i^*)},$$

and the log posterior distribution of α^* is given by

$$\begin{aligned}\log\hat{\pi}(\alpha^* \,|\, y) = \log\left(N^{-1}\sum_{\tau=1}^{N}\prod_{i=1}^{m}\left(a(\alpha_i^* \,|\, \alpha_i^{(\tau)})\prod_{\substack{j=1\\ j\neq i}}^{m} q(\alpha_{i,j}^* \,|\, \alpha_{i,j}^{(\tau)})\right)\right) \\ - \log\left(N^{-1}\sum_{\tau=1}^{N}\prod_{i=1}^{m} a(\alpha_i^{(\tau)} \,|\, \alpha_i^*)\right). \end{aligned} \tag{A.7}$$

Therefore, given Equations (A.2), (A.3), (A.4) and (A.7), the log marginal likelihood is estimated as follows:

$$\log\hat{f}(y) = \sum_{t=1}^{n}\log\left(\sum_{i=1}^{m} f(y_t \,|\, s_t = i, \mu^*, \lambda^*, \alpha^*)\Pr(s_t = i \,|\, y_{1:t-1}, \mu^*, \lambda^*, \alpha^*)\right)$$

$$
\begin{aligned}
&+ \sum_{i=1}^{m} \log p(\mu_i^*) + \sum_{i=1}^{m} \log p(\lambda_i^*) + \sum_{i=1}^{m} \sum_{\substack{j=1\\ j\neq i}}^{m} \log p(\alpha_{i,j}^*) \\
&\quad - \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} \pi(\mu_i^* \,|\, y, \lambda^*, \alpha^*, s^{(\tau)}) \right) \\
&\qquad - \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} \pi(\lambda_i^* \,|\, y, \alpha^*, \mu^{(\tau)}, s^{(\tau)}) \right) \\
&\qquad\quad - \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} \left(a(\alpha_i^* \,|\, \alpha_i^{(\tau)}) \prod_{\substack{j=1\\ j\neq i}}^{m} q(\alpha_{i,j}^* \,|\, \alpha_{i,j}^{(\tau)}) \right) \right) \\
&\qquad\qquad + \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} a(\alpha_i^{(\tau)} \,|\, \alpha_i^*) \right). \qquad \text{(A.8)}
\end{aligned}
$$

Appendix B

Unknown Number of Hidden States: Multivariate NHGHMM

Let $\boldsymbol{\theta}^* = (\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, \alpha^*)$ be maximum a posteriori estimates of the joint posterior distribution. To evaluate a Bayes factor for each competing model with regard to the number of hidden states, m, it is practically required to compute the log marginal likelihood and is given as follows:

$$\log \hat{f}(\mathbf{y}) = \log f(\mathbf{y} \,|\, \boldsymbol{\theta}^*) + \log p(\boldsymbol{\theta}^*) - \log \hat{\pi}(\boldsymbol{\theta}^* \,|\, \mathbf{y}), \tag{B.1}$$

where $\hat{\pi}(\boldsymbol{\theta}^* \,|\, \mathbf{y})$ denotes a posterior estimate at $\boldsymbol{\theta}^*$ realised by the MCMC samples from the simulation of conditional posterior distributions. Equation (B.1) enables the MCMC samples to be directly used for estimating the term, $\hat{\pi}(\boldsymbol{\theta}^* \,|\, \mathbf{y})$, since a marginal likelihood is calculated by integrating a likelihood with respect to a prior distribution instead of a posterior one (Chib and Jeliazkov, 2001).

The following shows how to decompose each term of the right hand side in Equation (B.1) so that the log marginal likelihood can be calculated.

The log likelihood

Since $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)$ is a set of pairwise discrete-time observations, it follows that

$$\log f(\mathbf{y} \,|\, \boldsymbol{\theta}^*) = \log f(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n \,|\, \boldsymbol{\theta}^*)$$

$$
\begin{aligned}
&= \log(f(\mathbf{y}_n \,|\, \mathbf{y}_{1:n-1}, \boldsymbol{\theta}^*) f(\mathbf{y}_{n-1} \,|\, \mathbf{y}_{1:n-2}, \boldsymbol{\theta}^*) \cdots f(\mathbf{y}_2 \,|\, \mathbf{y}_1, \boldsymbol{\theta}^*) f(\mathbf{y}_1 \,|\, \boldsymbol{\theta}^*)) \\
&= \log\left(\prod_{t=1}^{n} f(\mathbf{y}_t \,|\, \mathbf{y}_{1:t-1}, \boldsymbol{\theta}^*) \right) \\
&= \sum_{t=1}^{n} \log f(\mathbf{y}_t \,|\, \mathbf{y}_{1:t-1}, \boldsymbol{\theta}^*) \\
&= \sum_{t=1}^{n} \log \left(\sum_{i=1}^{m} f(\mathbf{y}_t \,|\, \mathbf{y}_{1:t-1}, S_t = i, \boldsymbol{\theta}^*) f(S_t = i \,|\, \mathbf{y}_{1:t-1}, \boldsymbol{\theta}^*) \right) \\
&\qquad \text{(by the law of total probability)} \\
&= \sum_{t=1}^{n} \log \left(\sum_{i=1}^{m} \underbrace{f(\mathbf{y}_t \,|\, S_t = i, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, \alpha^*)}_{\text{the Gaussian density}} \underbrace{\Pr(S_t = i \,|\, \mathbf{y}_{1:t-1}, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, \alpha^*)}_{\text{filtered probability}} \right),
\end{aligned}
\tag{B.2}
$$

where $\mathbf{y}_t = (y_t^1, y_t^2, \ldots, y_t^d) \in \mathbb{R}^d$, for any $t \in \{1, 2, \ldots, n\}$.

The observations, $\{\mathbf{y}_t\}$, are conditionally independent for all $t \in \{1, 2, \ldots, n\}$. The filtered probability for each state $i \in \Omega$ at time $t \in \{1, 2, \ldots, n\}$ is available from by-products of the forward filtering backward sampling algorithm.

The log priors

Since each prior distribution is independent of the others, it follows that

$$
\begin{aligned}
\log p(\boldsymbol{\theta}^*) &= \log p(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, \alpha^*) \\
&= \log(p(\boldsymbol{\theta}^*) p(\boldsymbol{\Sigma}^*) p(\alpha^*)) \\
&= \log p(\boldsymbol{\mu}^*) + \log p(\boldsymbol{\Sigma}^*) + \log p(\alpha^*) \\
&= \sum_{i=1}^{m} \log p(\boldsymbol{\mu}_i^*) + \sum_{i=1}^{m} \log p(\Sigma_i^*) + \sum_{i=1}^{m} \sum_{\substack{j=1 \\ j \neq i}}^{m} \log p(\alpha_{i,j}^*).
\end{aligned}
\tag{B.3}
$$

The log posteriors

Following Chib (1995), the maximum a posteriori estimate, $(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, \alpha^*)$, is selected for more accurate density estimation from which the fact that a high density point is attributed to the point, $(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, \alpha^*)$. As such, the log posterior distribution is given by

$$\log \hat{\pi}(\boldsymbol{\theta}^* \,|\, \mathbf{y})$$

$$\begin{aligned}
&= \log \hat{\pi}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, \alpha^* \,|\, \mathbf{y}) \\
&= \log(\hat{\pi}(\boldsymbol{\Sigma}^* \,|\, \mathbf{y}, \boldsymbol{\mu}^*, \alpha^*)\hat{\pi}(\boldsymbol{\mu}^* \,|\, \mathbf{y}, \alpha^*)\hat{\pi}(\alpha^* \,|\, \mathbf{y})) \\
&= \log\left(N^{-1}\sum_{\tau=1}^{N}\prod_{i=1}^{m}\pi(\Sigma_i^* \,|\, \mathbf{y}, \boldsymbol{\mu}^*, \alpha^*, s^{(\tau)})\right) + \log\left(N^{-1}\sum_{\tau=1}^{N}\prod_{i=1}^{m}\pi(\boldsymbol{\mu}_i^* \,|\, \mathbf{y}, \alpha^*, \boldsymbol{\Sigma}^{(\tau)}, s^{(\tau)})\right) \\
&\qquad + \log \hat{\pi}(\alpha^* \,|\, \mathbf{y}).
\end{aligned} \tag{B.4}$$

For $\log \hat{\pi}(\alpha^* \,|\, \mathbf{y})$, the MCMC samples from the full conditional posterior distribution are Metropolis-Hastings updates, and therefore, the log posterior distribution has a different form than those of the full conditional posterior distributions of parameters of interest (Chib and Jeliazkov, 2001).

Recall that the detailed balance condition between a new point, $\boldsymbol{\theta}'$, and a current point, $\boldsymbol{\theta}$, is assumed. Then,

$$Q(\boldsymbol{\theta}' \,|\, \boldsymbol{\theta})\pi(\boldsymbol{\theta}) = Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}')\pi(\boldsymbol{\theta}'),$$

where $Q(\boldsymbol{\theta}' \,|\, \boldsymbol{\theta})$ denotes the transition kernel from the current point, $\boldsymbol{\theta}$, to the proposed point, $\boldsymbol{\theta}'$. Define $Q(\boldsymbol{\theta}' \,|\, \boldsymbol{\theta}) = a(\boldsymbol{\theta}' \,|\, \boldsymbol{\theta})q(\boldsymbol{\theta}' \,|\, \boldsymbol{\theta})$, where $a(\boldsymbol{\theta}' \,|\, \boldsymbol{\theta})$ denotes the acceptance probability from $\boldsymbol{\theta}$ to $\boldsymbol{\theta}'$, and $q(\boldsymbol{\theta}' \,|\, \boldsymbol{\theta})$ denotes the proposal distribution for the random walk Metropolis-Hastings move from $\boldsymbol{\theta}$ to $\boldsymbol{\theta}'$.

For the log posterior distribution of α^* given the observations, $\mathbf{y}$, the following expression is obtained such that

$$Q(\alpha^* \,|\, \alpha)\pi(\alpha \,|\, \mathbf{y}) = Q(\alpha \,|\, \alpha^*)\pi(\alpha^* \,|\, \mathbf{y}). \tag{B.5}$$

Then, integrate Equation (B.5) with respect to α. It follows that

$$\begin{aligned}
\int Q(\alpha^* \,|\, \alpha)\pi(\alpha \,|\, \mathbf{y})\,\mathrm{d}\alpha &= \int Q(\alpha \,|\, \alpha^*)\pi(\alpha^* \,|\, \mathbf{y})\,\mathrm{d}\alpha \\
\Leftrightarrow \pi(\alpha^* \,|\, \mathbf{y})\int a(\alpha \,|\, \alpha^*)q(\alpha \,|\, \alpha^*)\,\mathrm{d}\alpha &= \int a(\alpha^* \,|\, \alpha)q(\alpha^* \,|\, \alpha)\pi(\alpha \,|\, \mathbf{y})\,\mathrm{d}\alpha \\
\Leftrightarrow \pi(\alpha^* \,|\, \mathbf{y}) = \frac{\int a(\alpha^* \,|\, \alpha)q(\alpha^* \,|\, \alpha)\pi(\alpha \,|\, \mathbf{y})\,\mathrm{d}\alpha}{\int a(\alpha \,|\, \alpha^*)q(\alpha \,|\, \alpha^*)\,\mathrm{d}\alpha} &= \frac{\mathrm{E}_{\pi(\alpha \,|\, \mathbf{y})}[a(\alpha^* \,|\, \alpha)q(\alpha^* \,|\, \alpha)]}{\mathrm{E}_{q(\alpha \,|\, \alpha^*)}[a(\alpha \,|\, \alpha^*)]}.
\end{aligned} \tag{B.6}$$

Using the Monte Carlo method, Equation (B.6) can be estimated as follows:

$$\pi(\alpha^* \,|\, \mathbf{y}) \approx \hat{\pi}(\alpha^* \,|\, \mathbf{y}) = \frac{N_1^{-1} \sum_{\tau=1}^{N_1} a(\alpha^* \,|\, \alpha^{(\tau)}) q(\alpha^* \,|\, \alpha^{(\tau)})}{N_2^{-1} \sum_{\nu=1}^{N_2} a(\alpha^{(\nu)} \,|\, \alpha^*)},$$

where $\{\alpha^{(\tau)}\}$ is a set of realisations from the full conditional posterior distribution, $\pi(\alpha \,|\, \mathbf{y})$, and $\{\alpha^{(\nu)}\}$ is a set of realisations from the proposal distribution, $q(\alpha \,|\, \alpha^*)$. Normally, $N_1 \geq N_2$ since the time required for convergence of samples from the proposal distribution is less in comparison to the MCMC samples from the full conditional posterior distribution (Chib and Jeliazkov, 2001).

For the sake of simplicity, the condition $N_1 = N_2 = N$ is considered. Then, for α_i where $\forall i \in \Omega$, it follows that

$$\hat{\pi}(\alpha_i^* \,|\, y) = \frac{N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} \left(a(\alpha_i^* \,|\, \alpha_i^{(\tau)}) \prod_{\substack{j=1 \\ j \neq i}}^{m} q(\alpha_{i,j}^* \,|\, \alpha_{i,j}^{(\tau)}) \right)}{N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} a(\alpha_i^{(\tau)} \,|\, \alpha_i^*)},$$

and the log posterior distribution of α^* is given by

$$\log \hat{\pi}(\alpha^* \,|\, \mathbf{y}) = \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} \left(a(\alpha_i^* \,|\, \alpha_i^{(\tau)}) \prod_{\substack{j=1 \\ j \neq i}} q(\alpha_{i,j}^* \,|\, \alpha_{i,j}^{(\tau)}) \right) \right) - \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} a(\alpha_i^{(\tau)} \,|\, \alpha_i^*) \right). \quad \text{(B.7)}$$

Therefore, given Equations (B.2), (B.3), (B.4) and (B.7), the log marginal likelihood is estimated as follows:

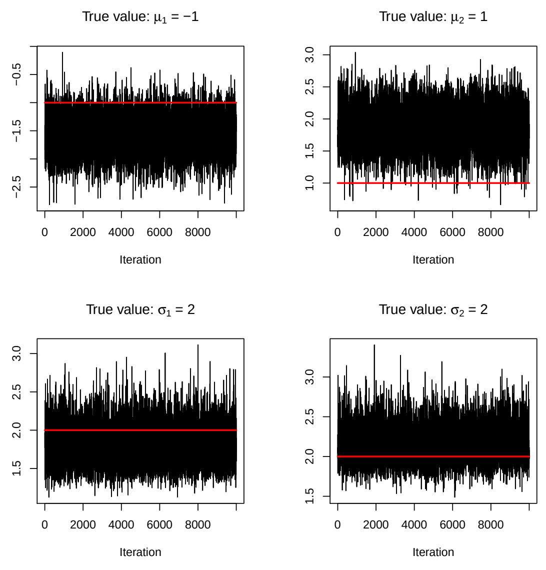
$$\begin{aligned}
\log \hat{f}(\mathbf{y}) = \sum_{t=1}^{n} \log \left(\sum_{i=1}^{m} f(\mathbf{y}_t \,|\, s_t = i, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, \alpha^*) \Pr(s_t = i \,|\, \mathbf{y}_{1:t-1}, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*, \alpha^*) \right) \\
+ \sum_{i=1}^{m} \log p(\boldsymbol{\mu}_i^*) + \sum_{i=1}^{m} \log p(\Sigma_i^*) + \sum_{i=1}^{m} \sum_{\substack{j=1 \\ j \neq i}}^{m} \log p(\alpha_{i,j}^*) \\
- \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} \pi(\Sigma_i^* \,|\, \mathbf{y}, \boldsymbol{\mu}^*, \alpha^*, s^{(\tau)}) \right) \\
- \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} \pi(\boldsymbol{\mu}_i^* \,|\, \mathbf{y}, \alpha^*, \boldsymbol{\Sigma}^{(\tau)}, s^{(\tau)}) \right)
\end{aligned}$$

$$- \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} \left(a(\alpha_i^* \mid \alpha_i^{(\tau)}) \prod_{\substack{j=1 \\ j \neq i}} q(\alpha_{i,j}^* \mid \alpha_{i,j}^{(\tau)}) \right) \right)$$

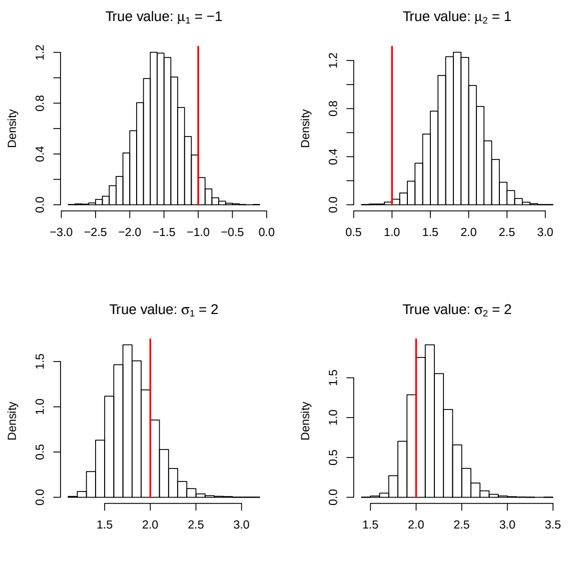
$$+ \log \left(N^{-1} \sum_{\tau=1}^{N} \prod_{i=1}^{m} a(\alpha_i^{(\tau)} \mid \alpha_i^*) \right). \tag{B.8}$$

Appendix C

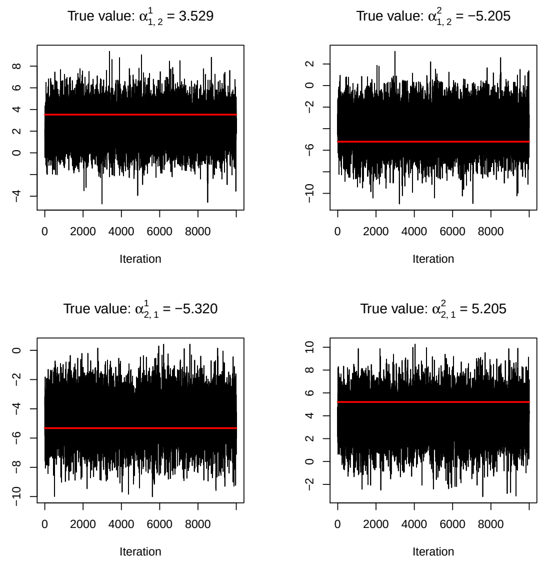
Figures: Univariate NHGHMM (Simulation Study)



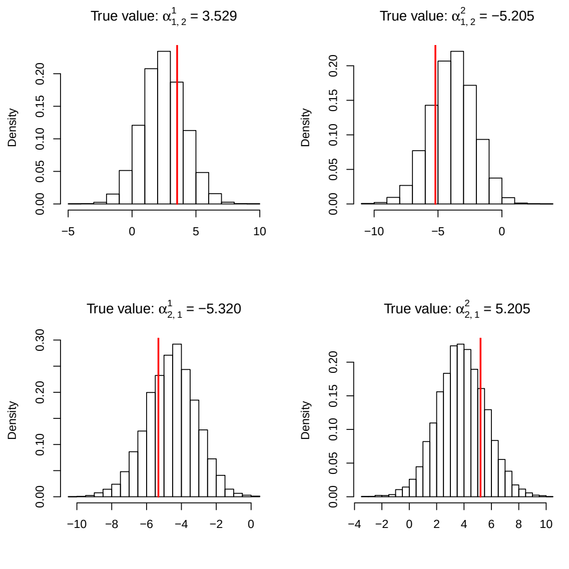
(a)



(b)

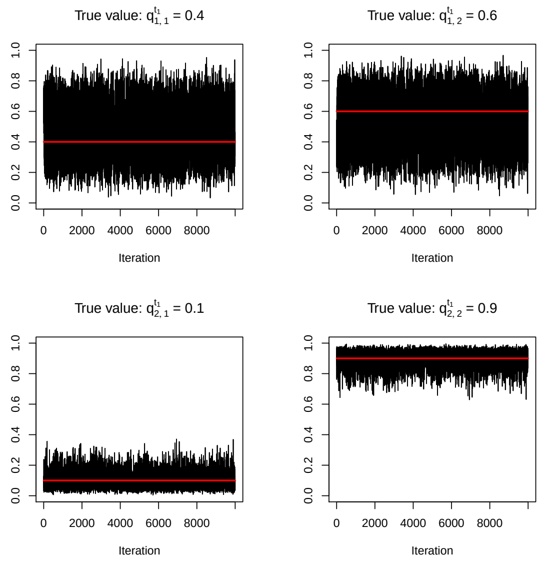


(c)

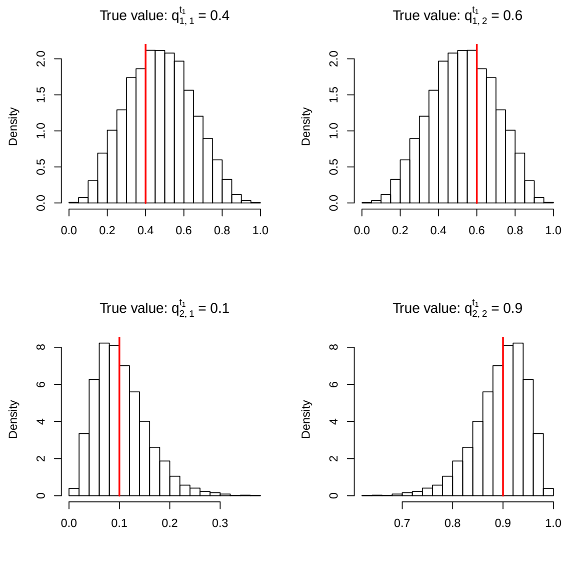


(d)

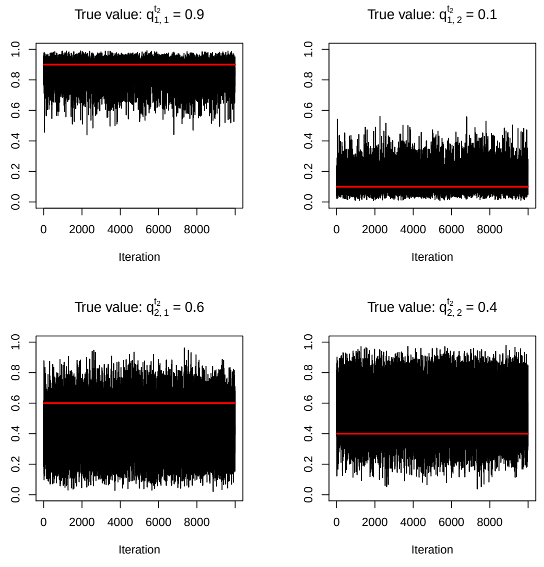
Figure C.1: Trace plots (a) and histograms (b) of μ & σ; trace plots (c) and histograms (d) of $\alpha_{1,2}$ & $\alpha_{2,1}$ for one of $R = 100$ replications using the AM algorithm. Each of the true values for the parameters is represented by the red line.



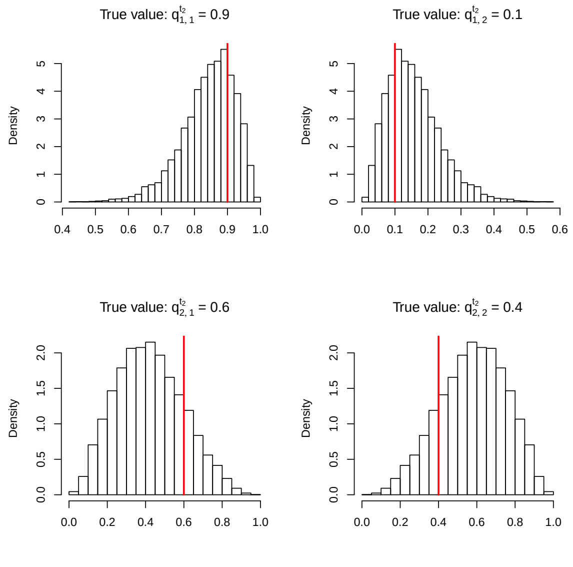
(a)



(b)

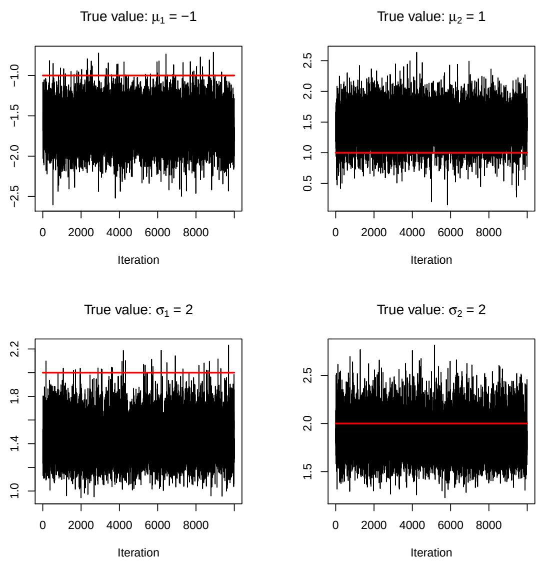


(c)

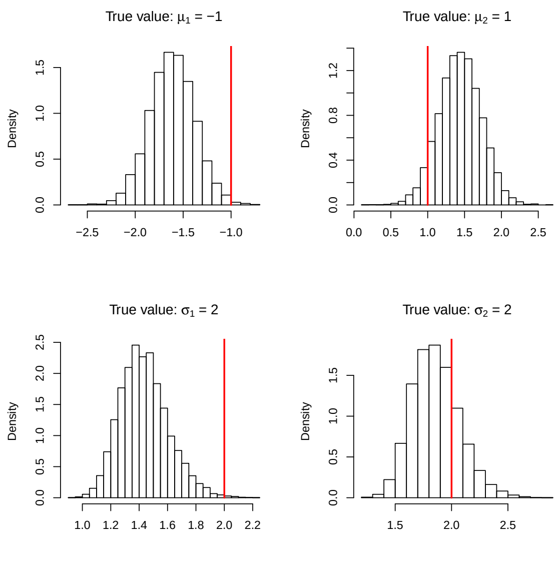


(d)

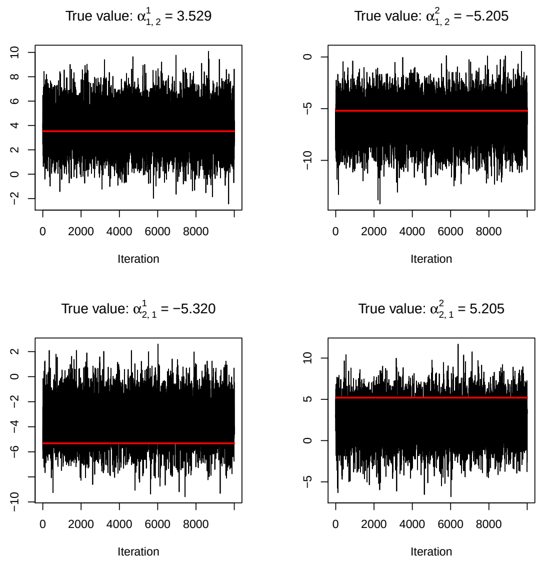
Figure C.2: Trace plots (a) and histograms (b) of Q^{t_1}; trace plots (c) and histograms (d) of Q^{t_2} for one of $R = 100$ replications using the AM algorithm. Each of the true values for the parameters is represented by the red line.



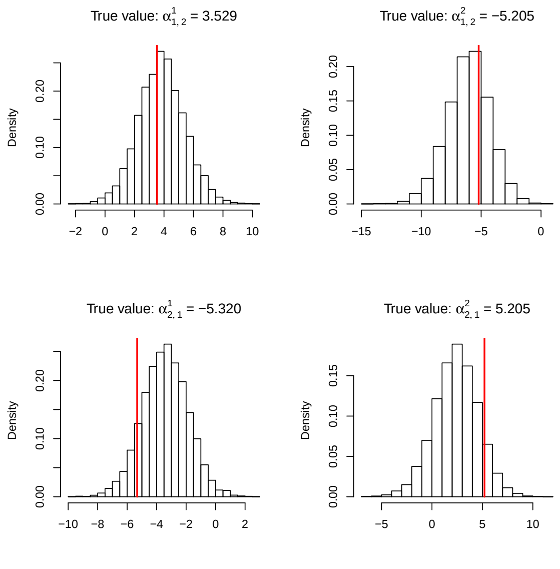
(a)



(b)

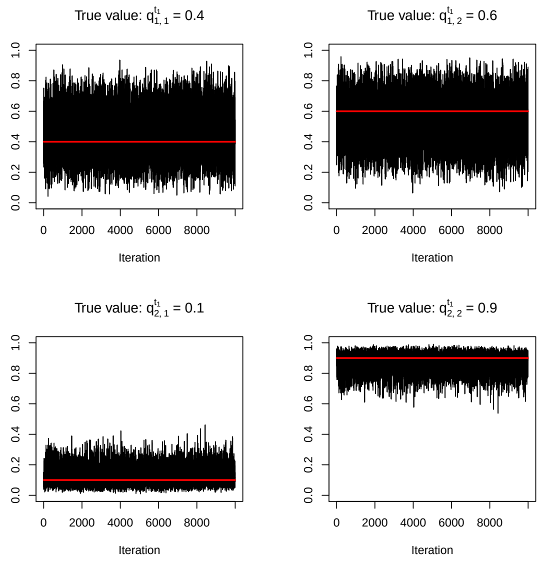


(c)

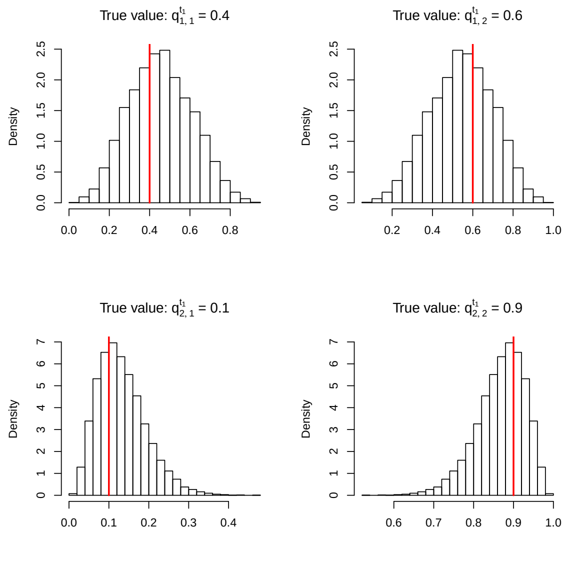


(d)

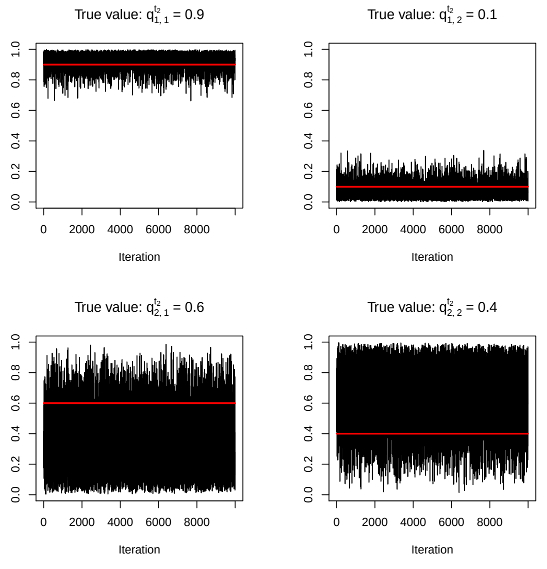
Figure C.3: Trace plots (a) and histograms (b) of μ & σ; trace plots (c) and histograms (d) of $\alpha_{1,2}$ & $\alpha_{2,1}$ for one of $R = 100$ replications using the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.



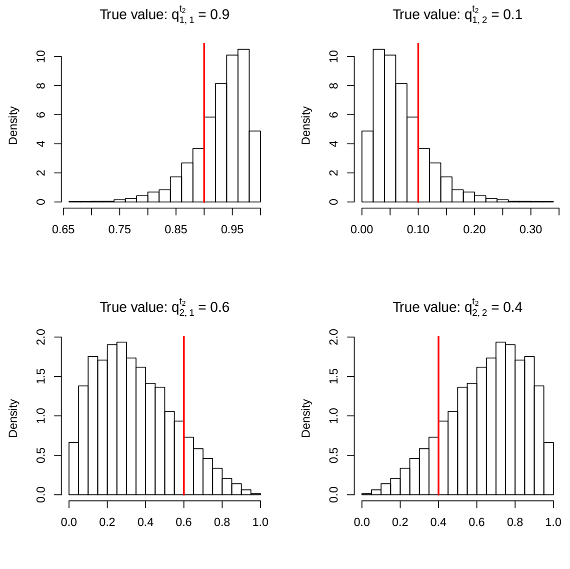
(a)



(b)

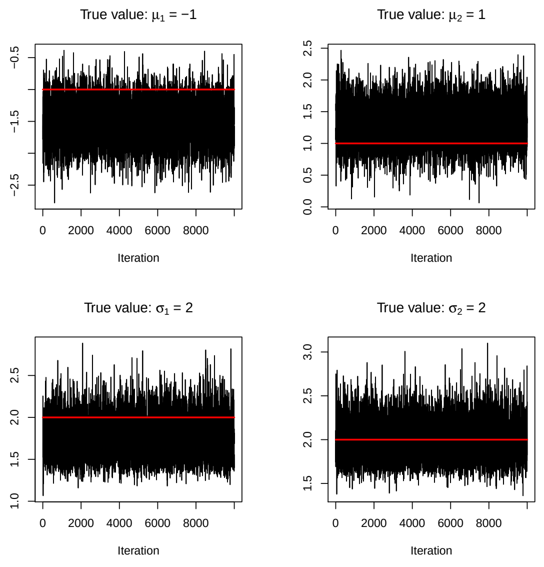


(c)

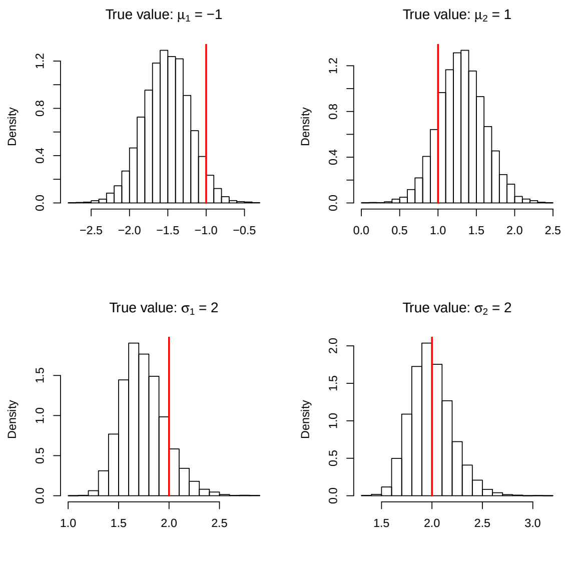


(d)

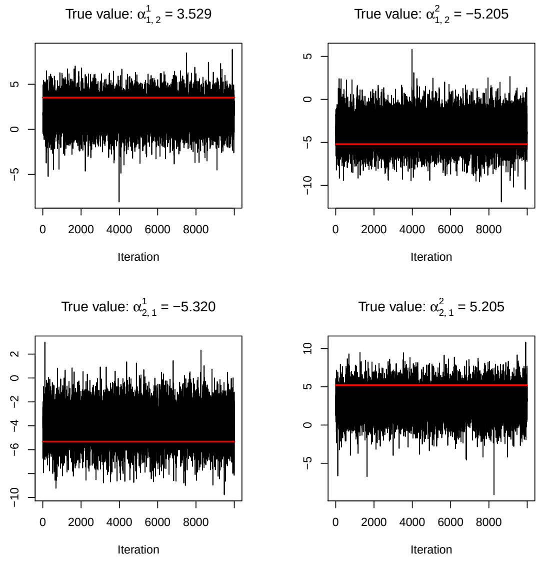
Figure C.4: Trace plots (a) and histograms (b) of Q^{t_1}; trace plots (c) and histograms (d) of Q^{t_2} for one of $R = 100$ replications using the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.



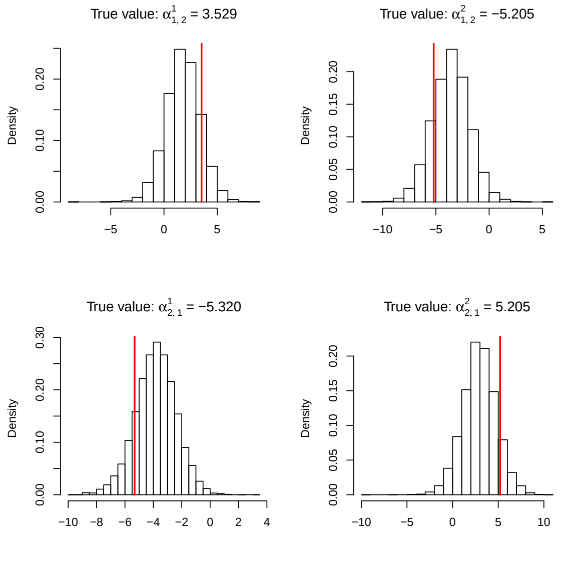
(a)



(b)

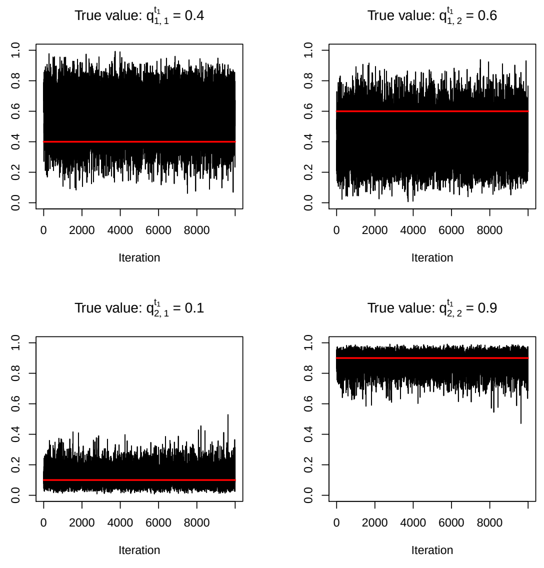


(c)

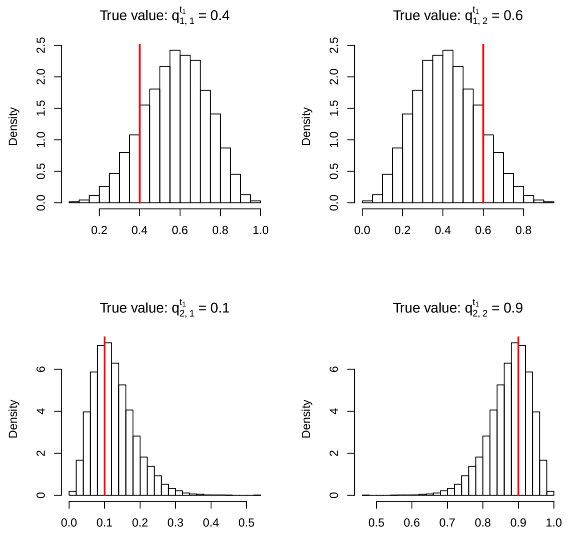


(d)

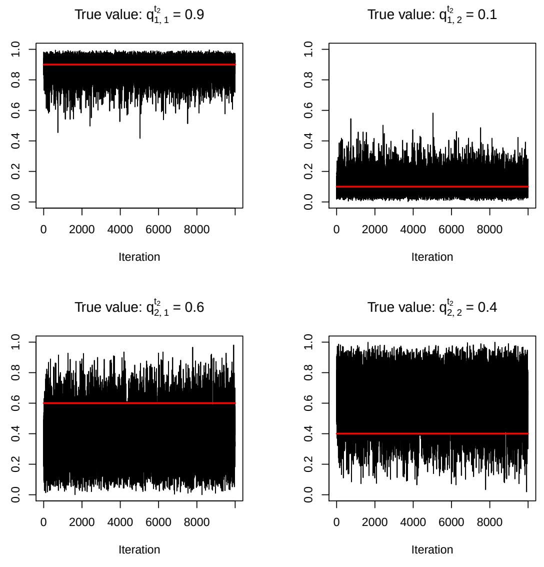
Figure C.5: Trace plots (a) and histograms (b) of μ & σ; trace plots (c) and histograms (d) of $\alpha_{1,2}$ & $\alpha_{2,1}$ for one of $R = 100$ replications using the MTAM algorithm. Each of the true values for the parameters is represented by the red line.



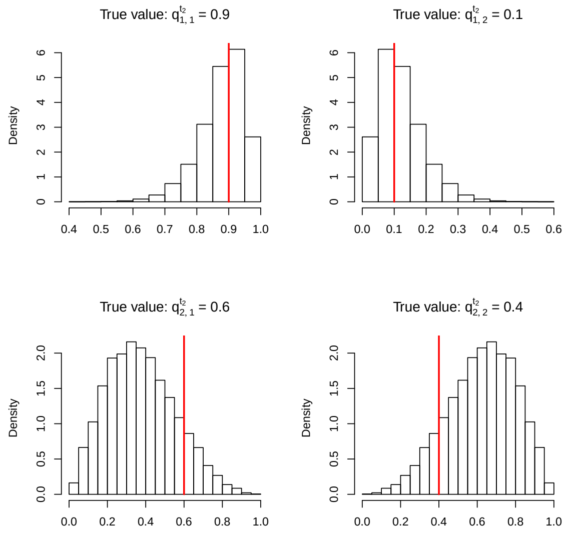
(a)



(b)



(c)



(d)

Figure C.6: Trace plots (a) and histograms (b) of Q^{t_1}; trace plots (c) and histograms (d) of Q^{t_2} for one of $R = 100$ replications using the MTAM algorithm. Each of the true values for the parameters is represented by the red line.

Appendix D

Figures: Multivariate NHGHMM (Simulation Study)

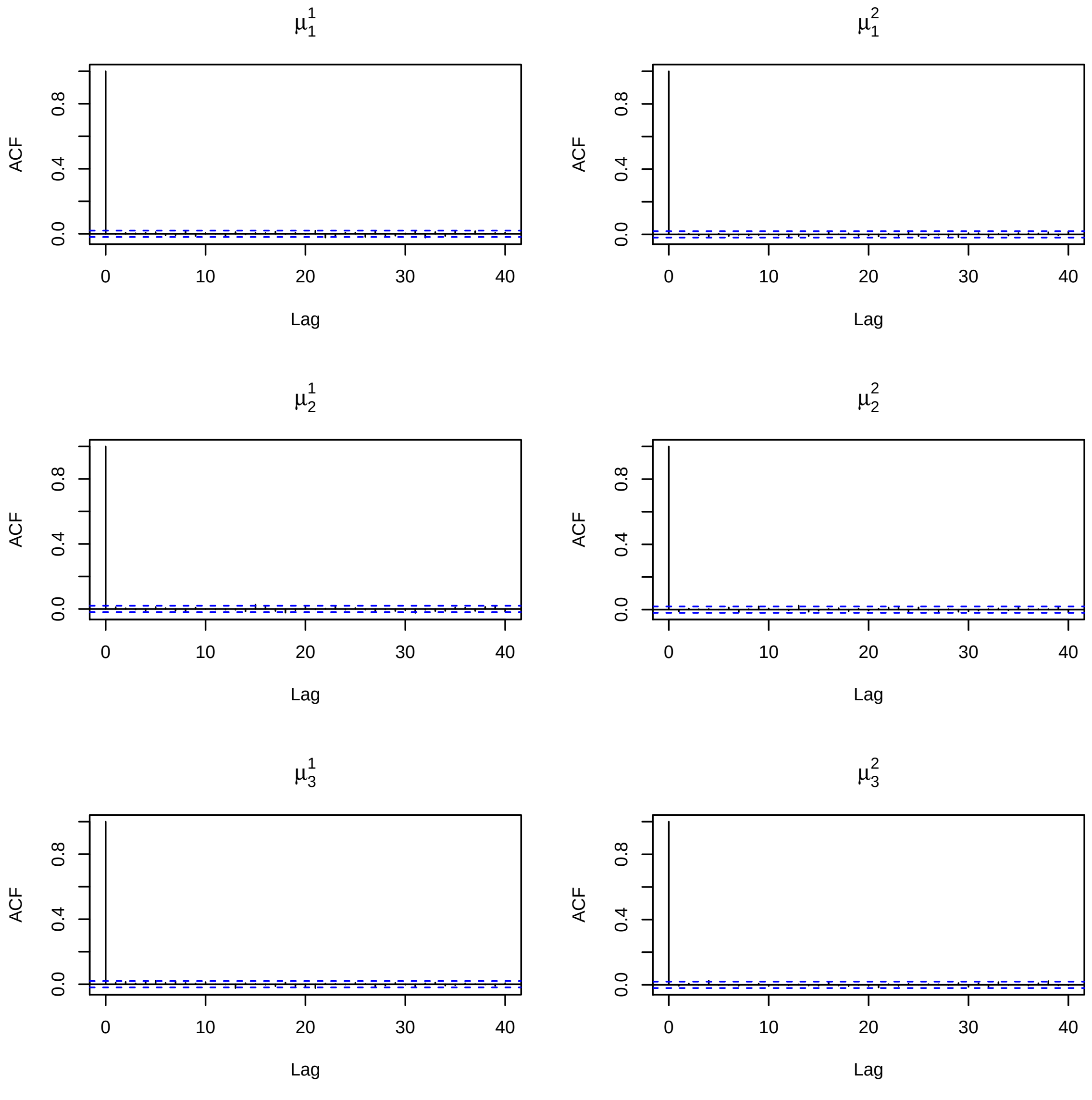
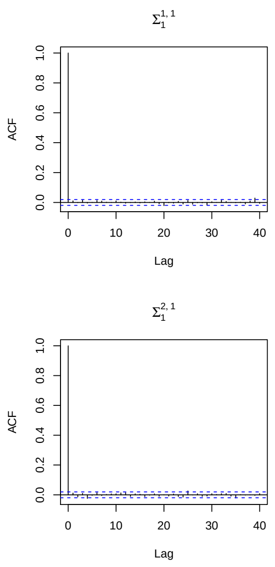
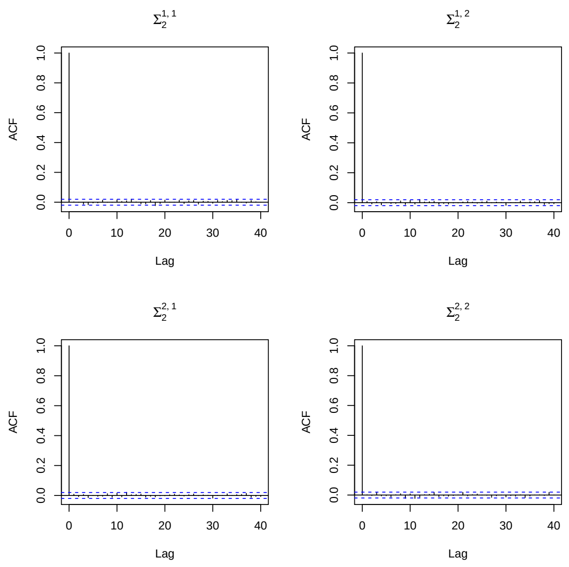


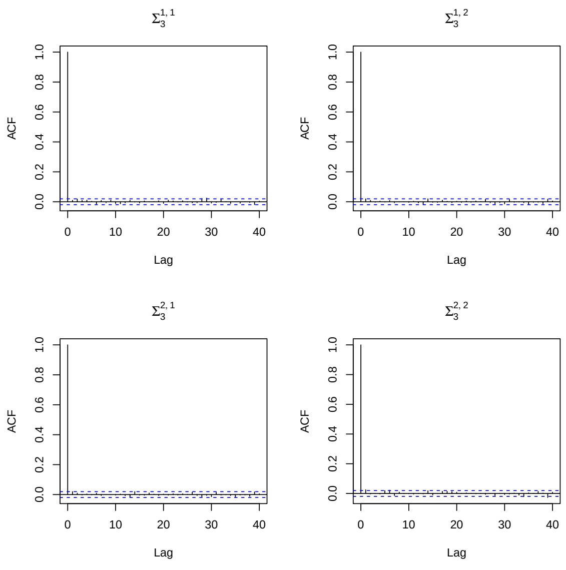
Figure D.1: Autocorrelation functions of $\boldsymbol{\mu}$ for the standard Metropolis-Hastings algorithm without thinning



(a)



(b)



(c)

Figure D.2: Autocorrelation functions of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the standard Metropolis-Hastings algorithm without thinning

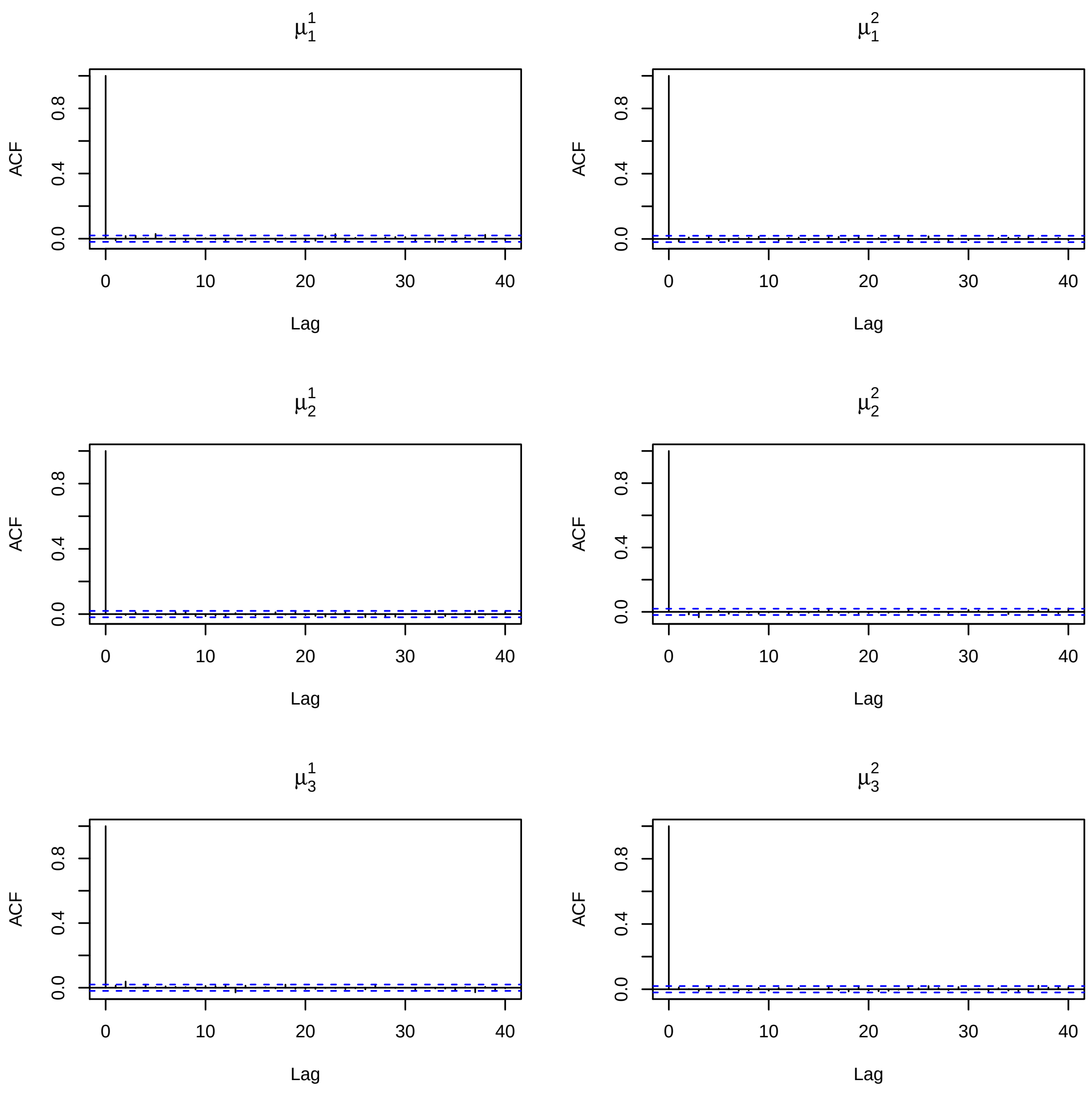
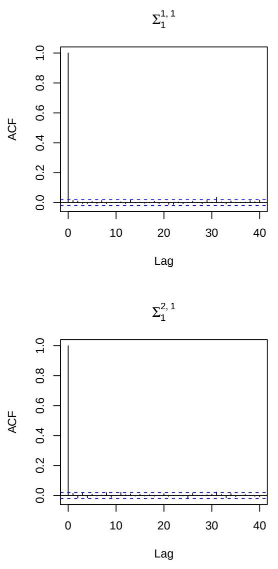
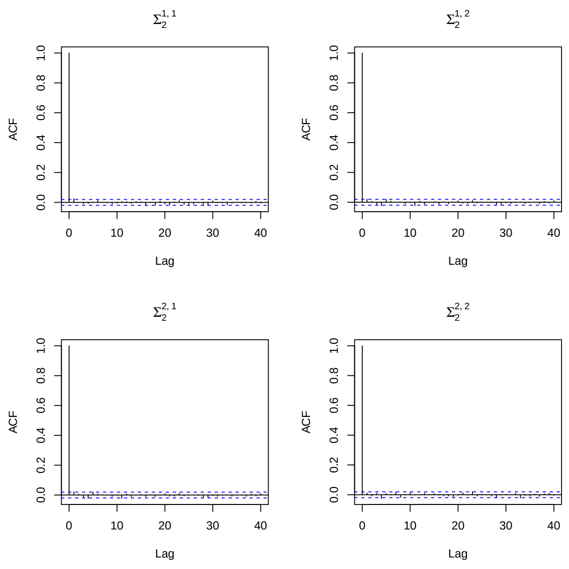
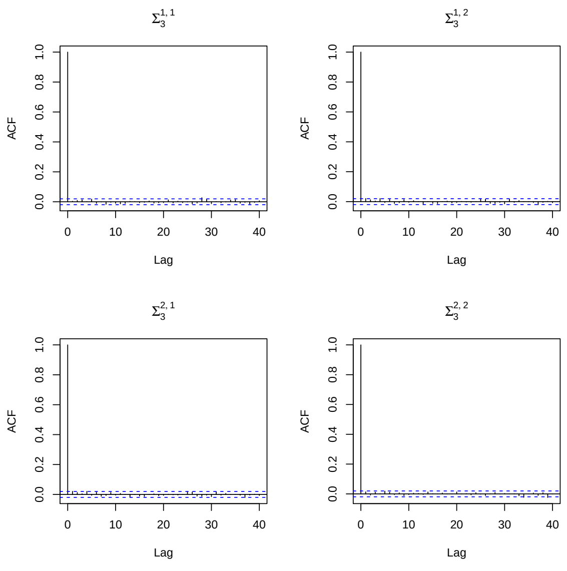


Figure D.3: Autocorrelation functions of $\boldsymbol{\mu}$ for the AM algorithm without thinning





(a) (b)



(c)

Figure D.4: Autocorrelation functions of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the AM algorithm without thinning

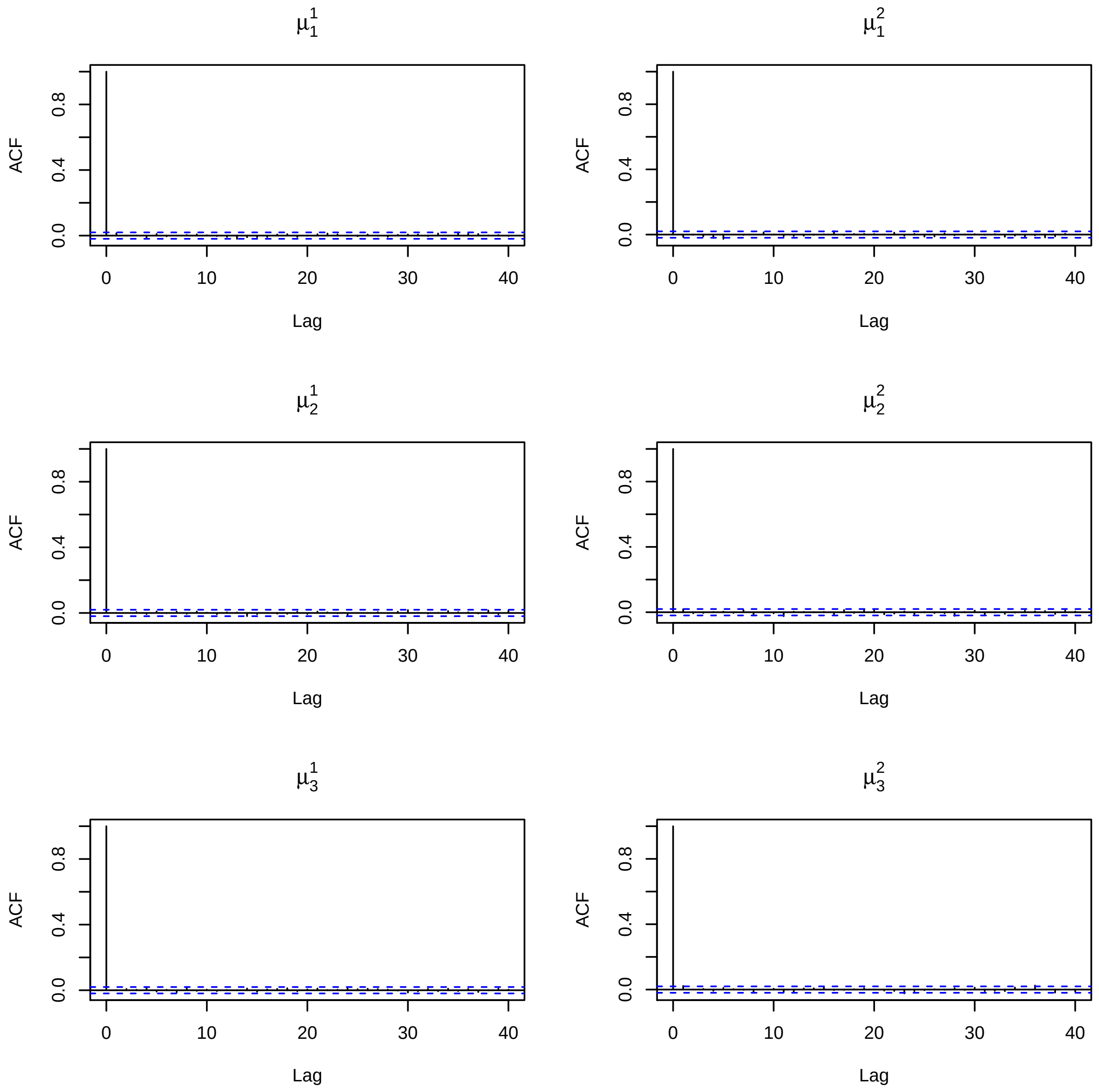
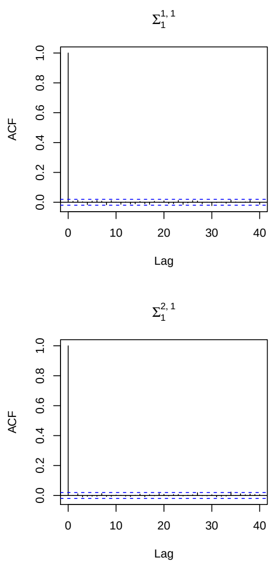
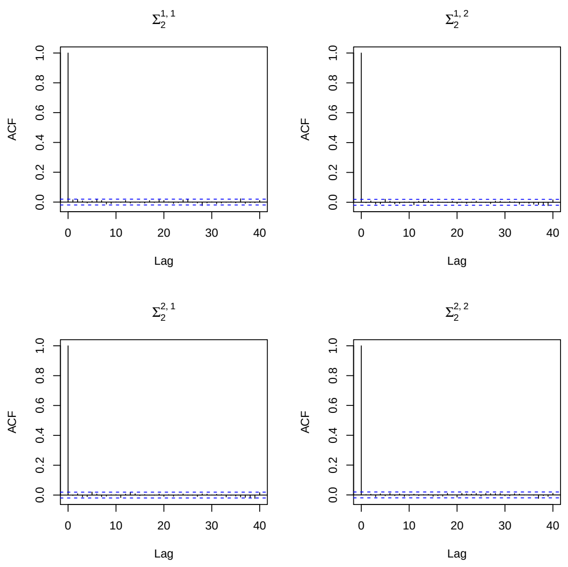


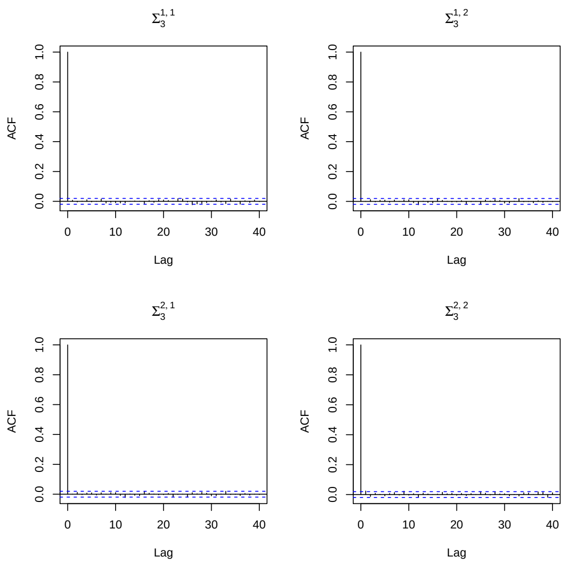
Figure D.5: Autocorrelation functions of $\boldsymbol{\mu}$ for the symmetric DRAM algorithm without thinning



(a)



(b)



(c)

Figure D.6: Autocorrelation functions of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the symmetric DRAM algorithm without thinning

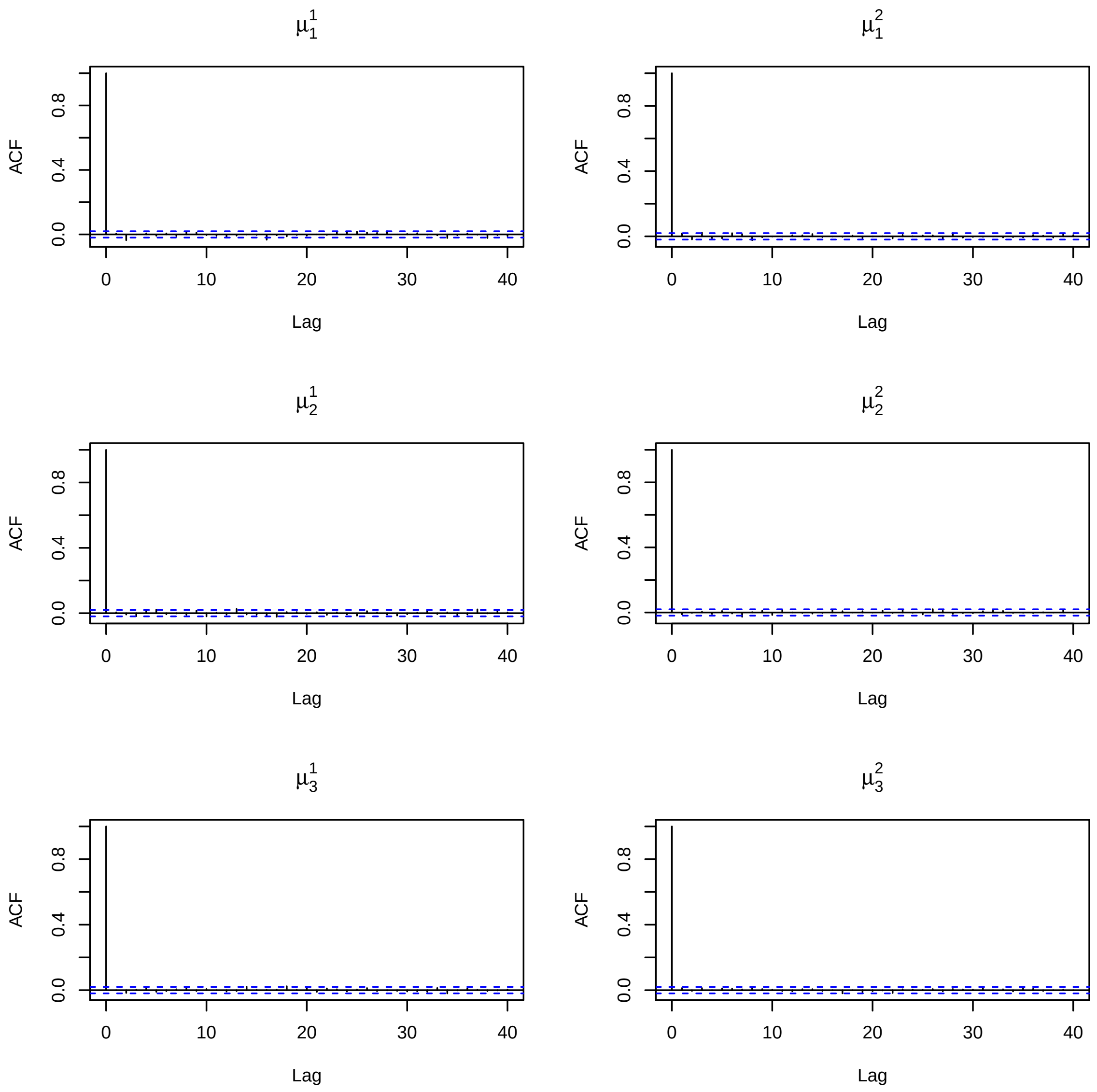
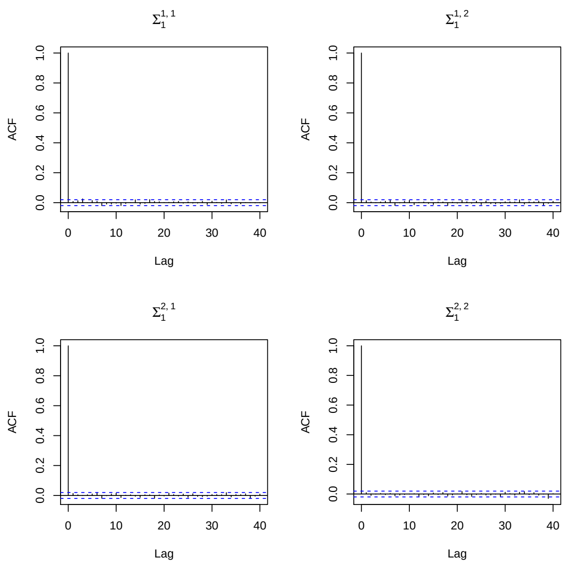
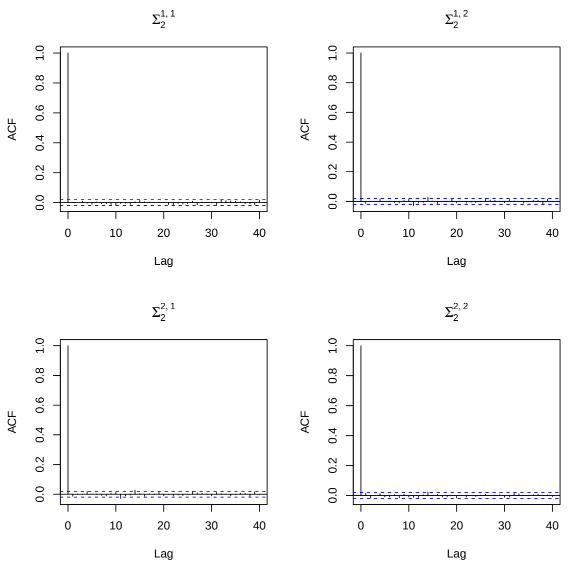


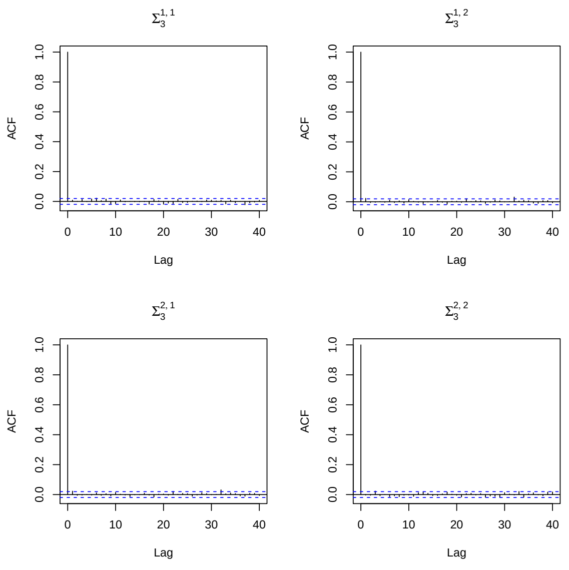
Figure D.7: Autocorrelation functions of $\boldsymbol{\mu}$ for the MTAM algorithm without thinning



(a)



(b)



(c)

Figure D.8: Autocorrelation functions of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the MTAM algorithm without thinning

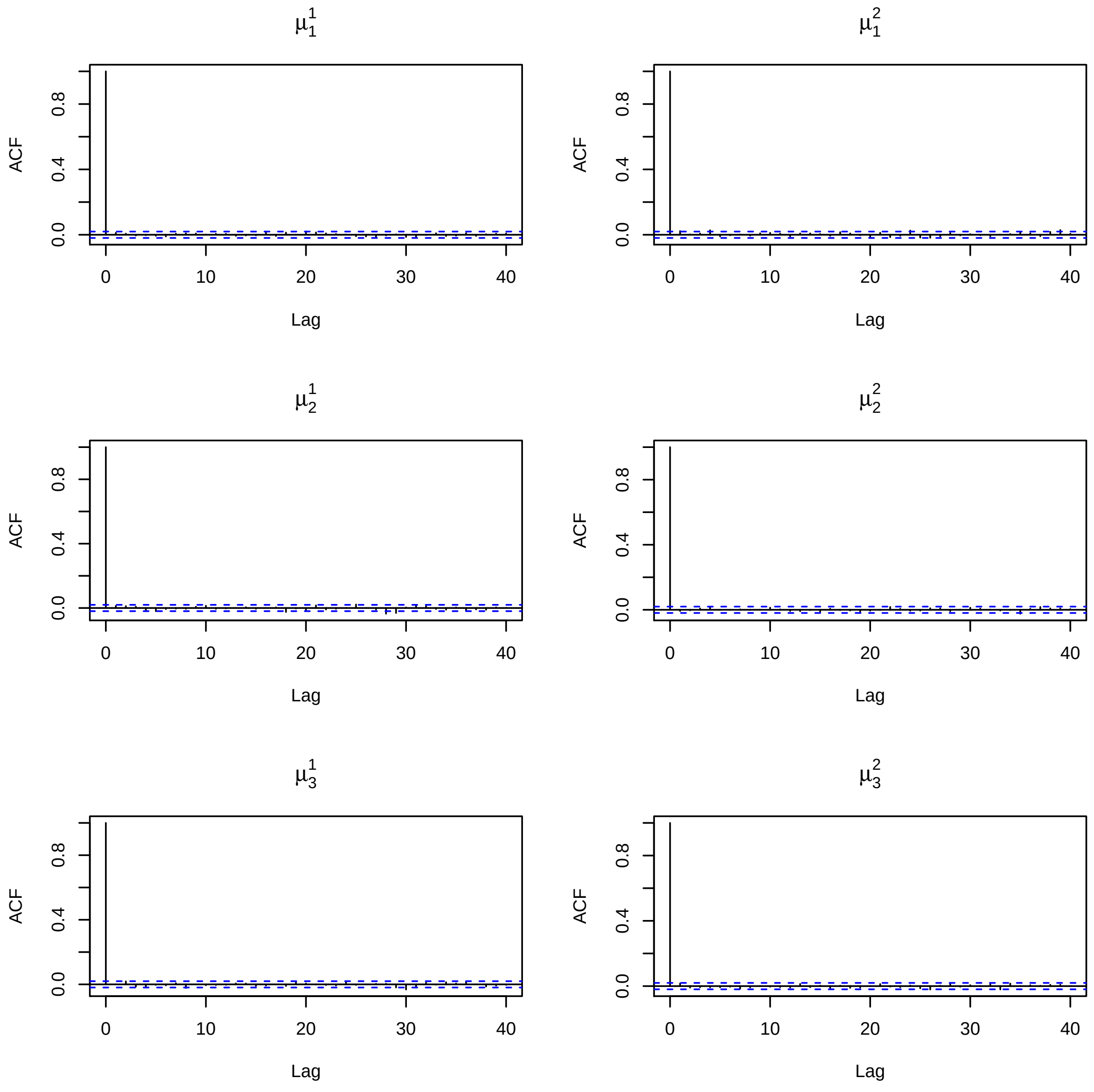
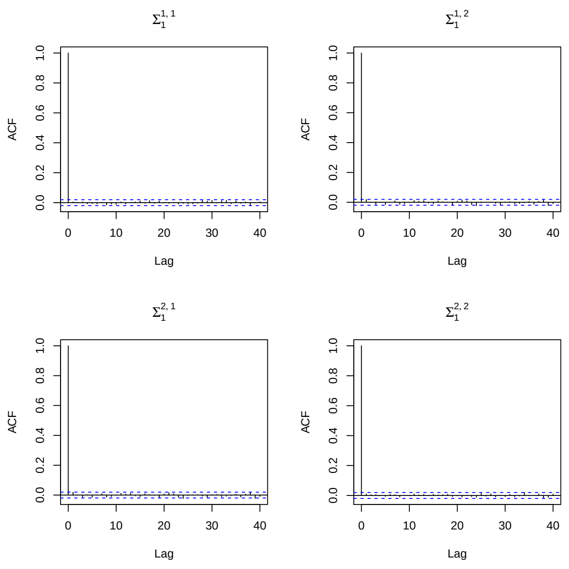
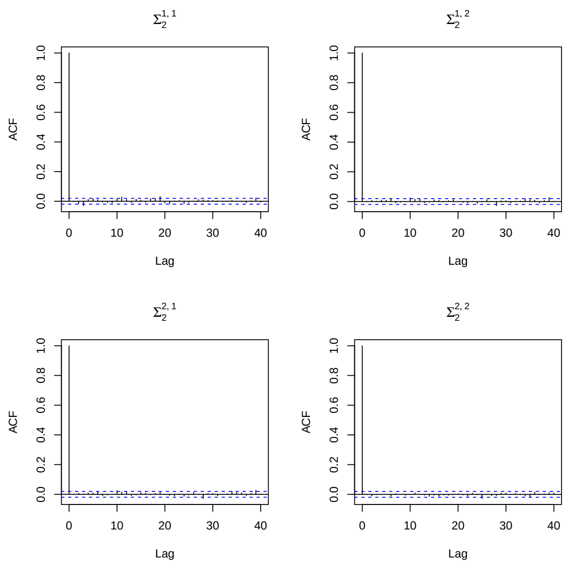


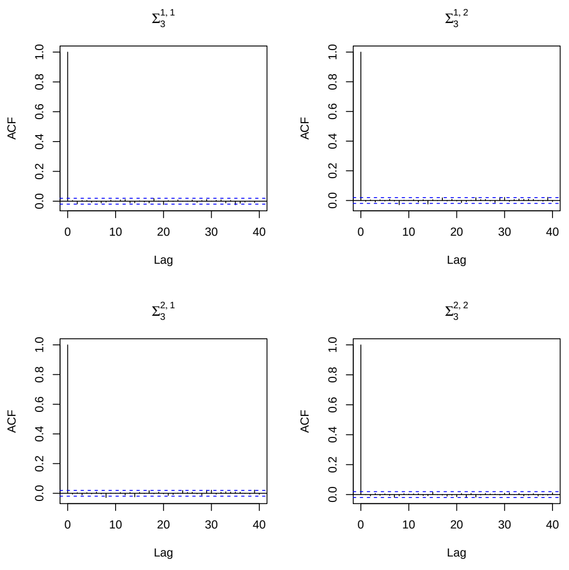
Figure D.9: Autocorrelation functions of $\boldsymbol{\mu}$ for the standard Metropolis-Hastings algorithm with thinning



(a)



(b)



(c)

Figure D.10: Autocorrelation functions of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the standard Metropolis-Hastings algorithm with thinning

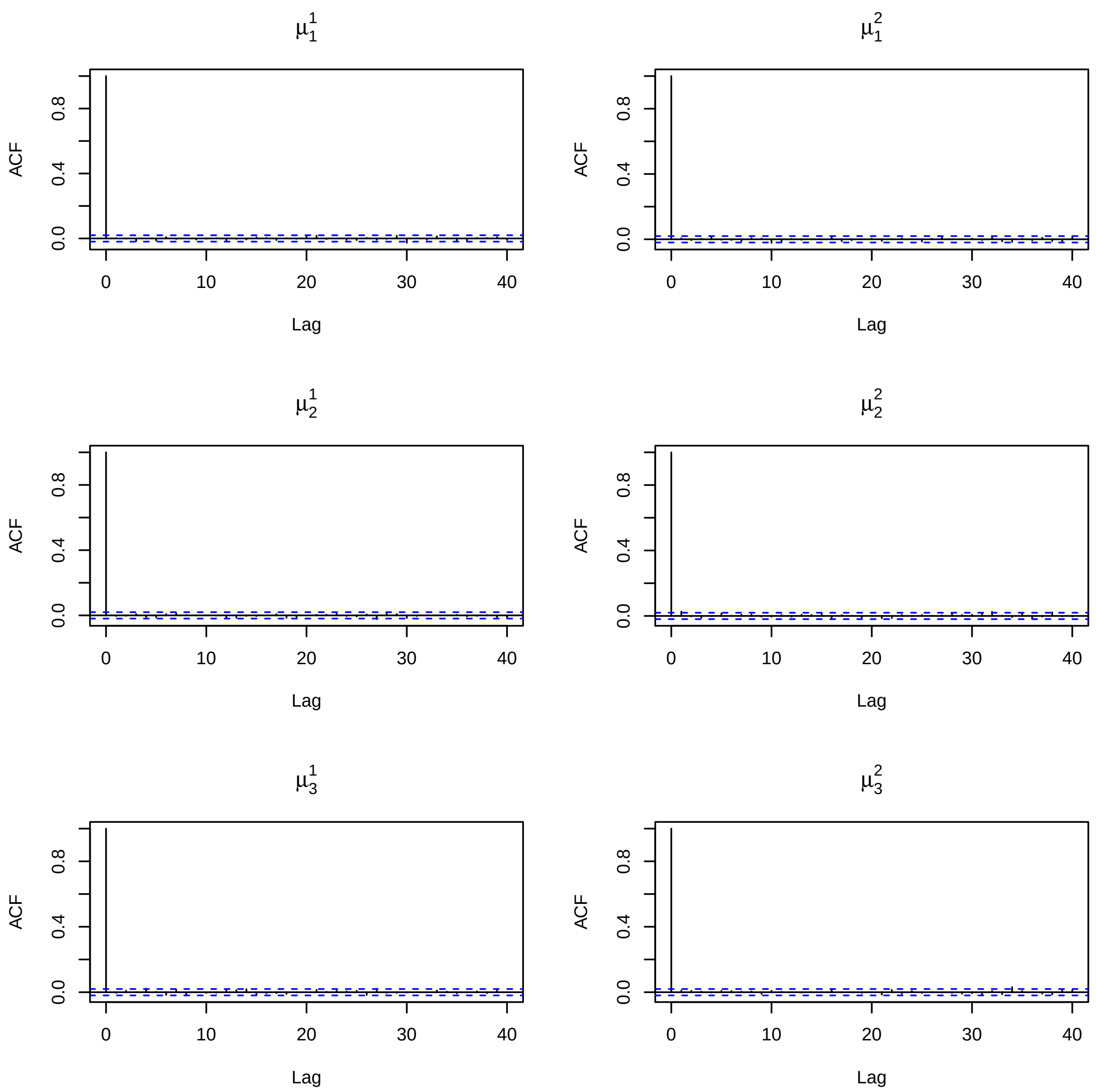
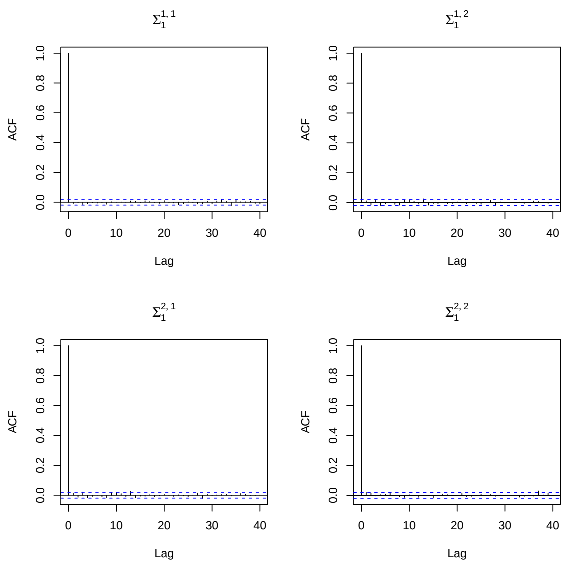
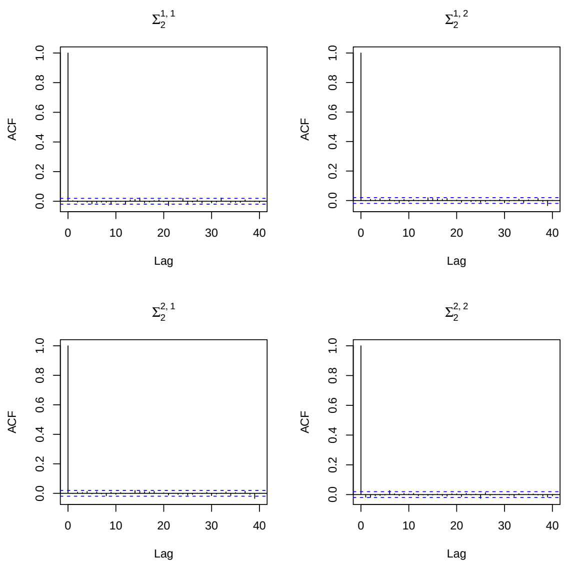


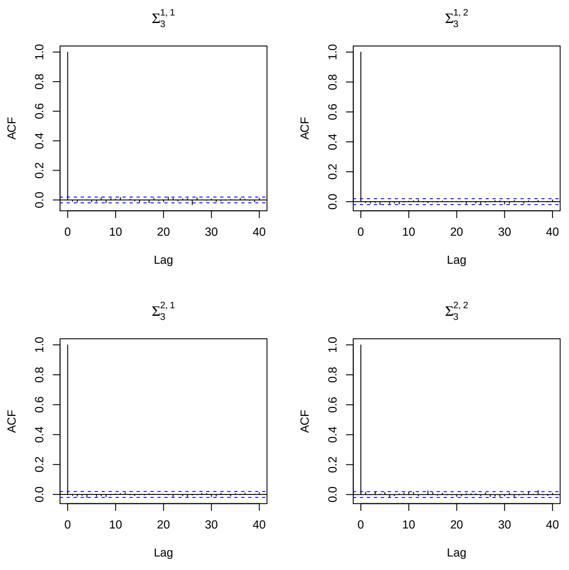
Figure D.11: Autocorrelation functions of $\boldsymbol{\mu}$ for the AM algorithm with thinning



(a)



(b)



(c)

Figure D.12: Autocorrelation functions of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the AM algorithm with thinning

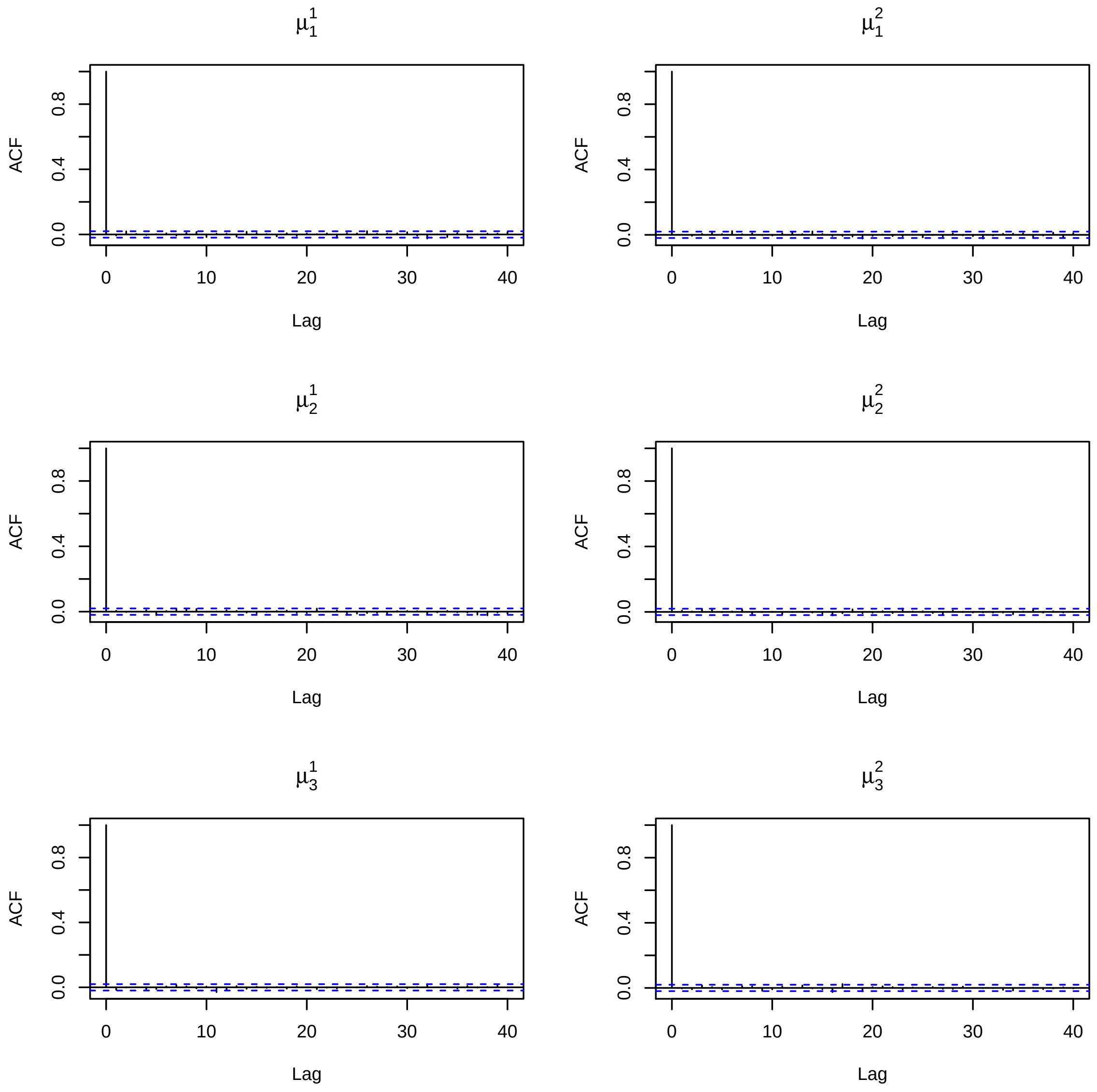
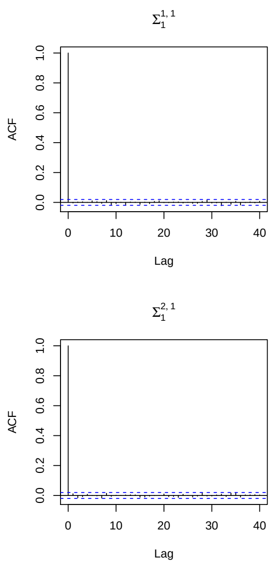
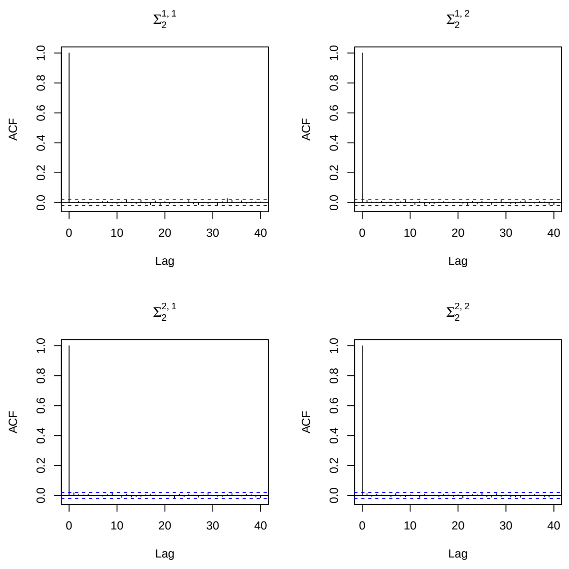


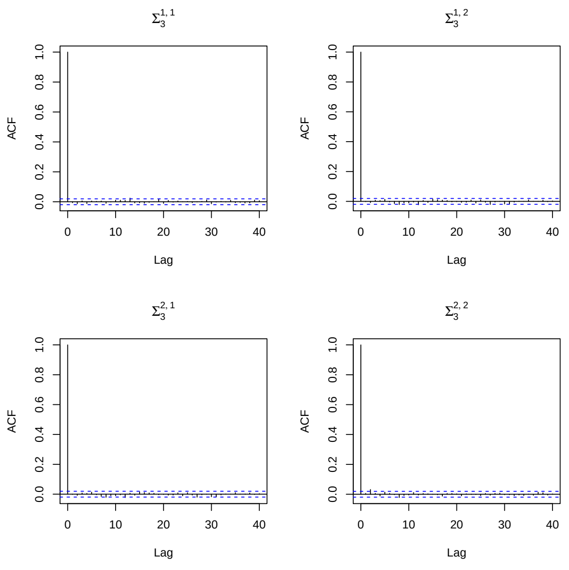
Figure D.13: Autocorrelation functions of $\boldsymbol{\mu}$ for the symmetric DRAM algorithm with thinning



(a)



(b)



(c)

Figure D.14: Autocorrelation functions of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the symmetric DRAM algorithm with thinning

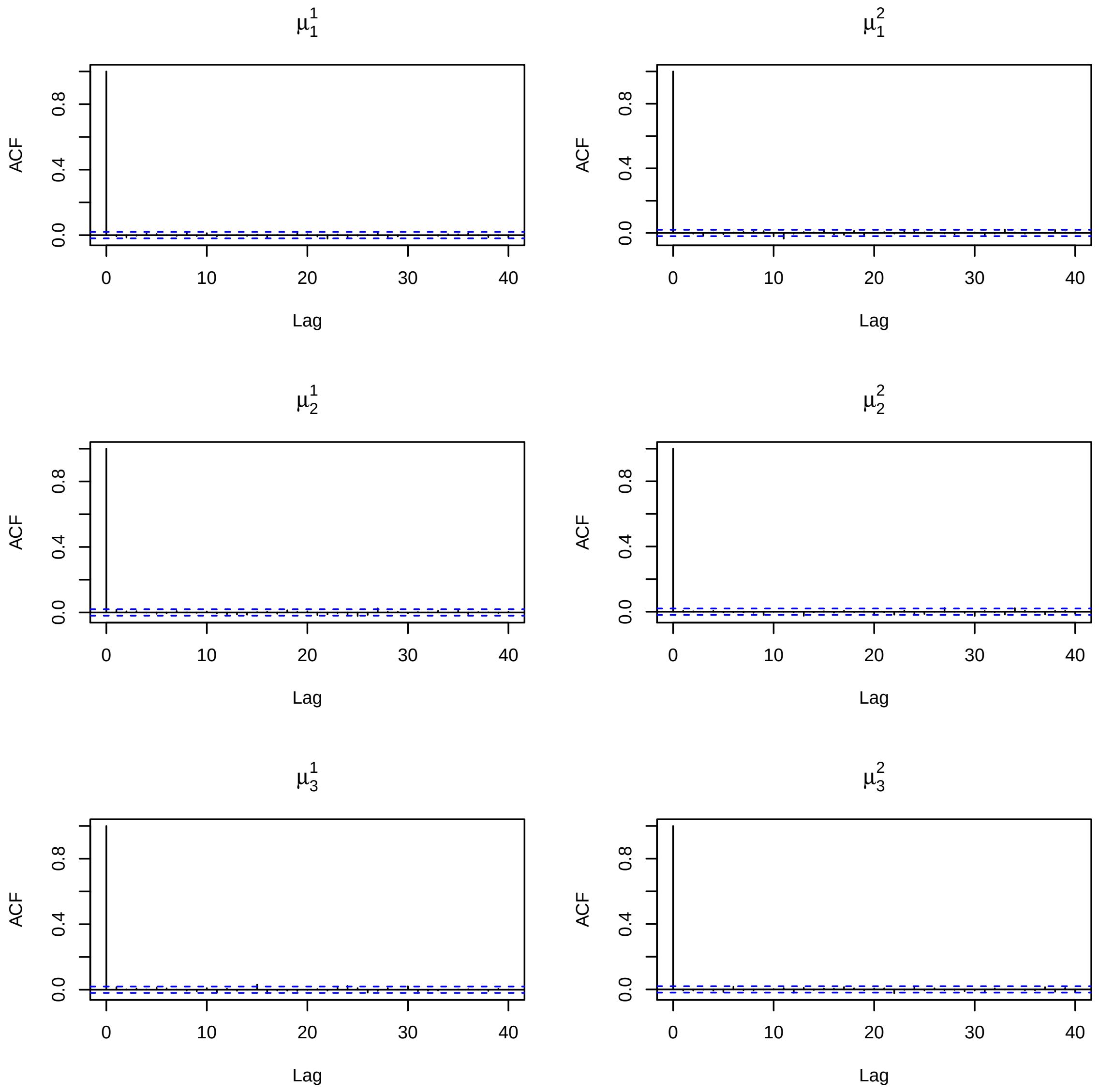
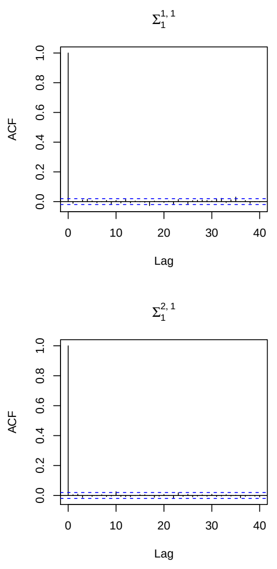
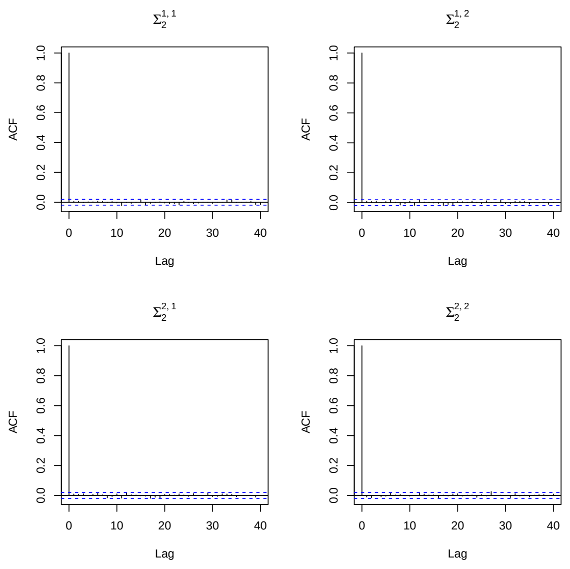


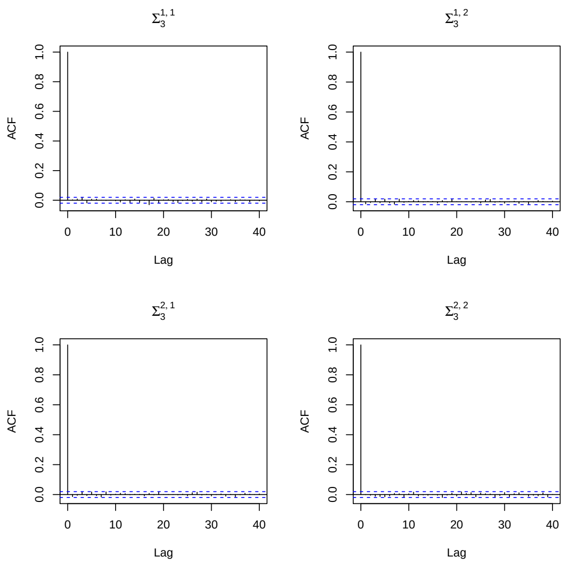
Figure D.15: Autocorrelation functions of $\boldsymbol{\mu}$ for the MTAM algorithm with thinning



(a)



(b)



(c)

Figure D.16: Autocorrelation functions of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the MTAM algorithm with thinning

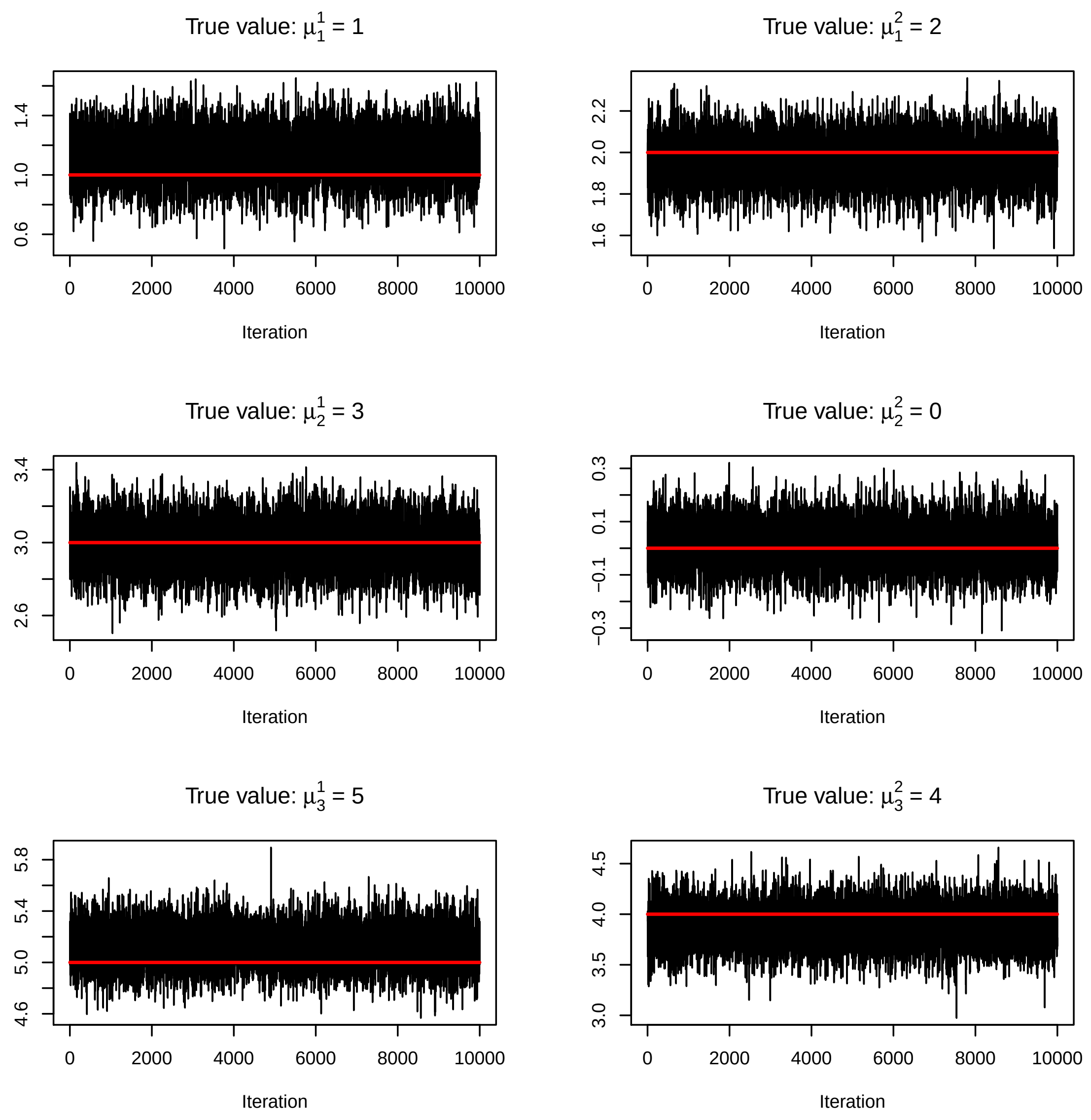


Figure D.17: Trace plots of $\boldsymbol{\mu}$ for the AM algorithm. Each of the true values for the parameters is represented by the red line.

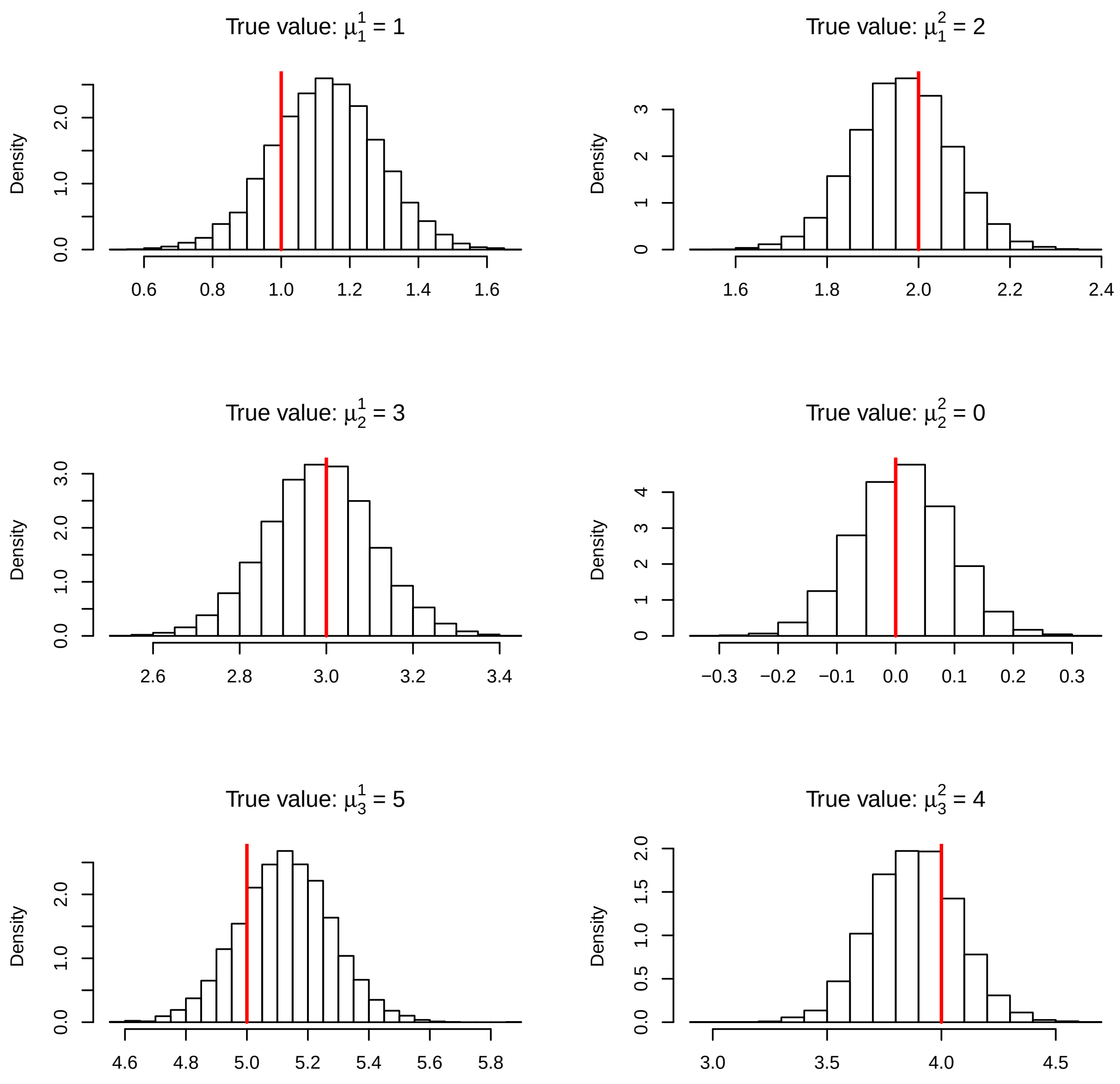
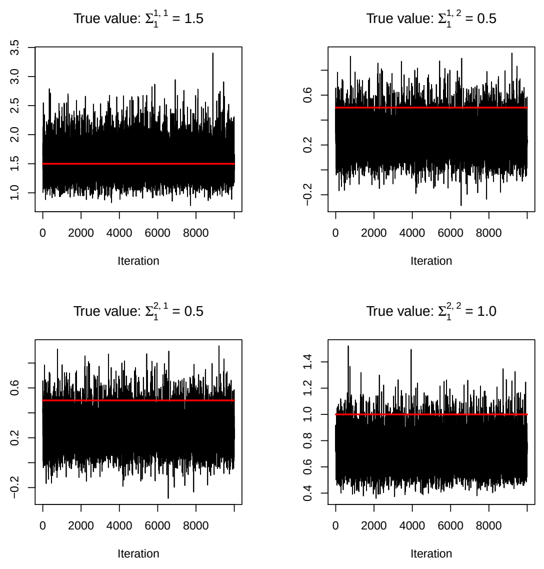
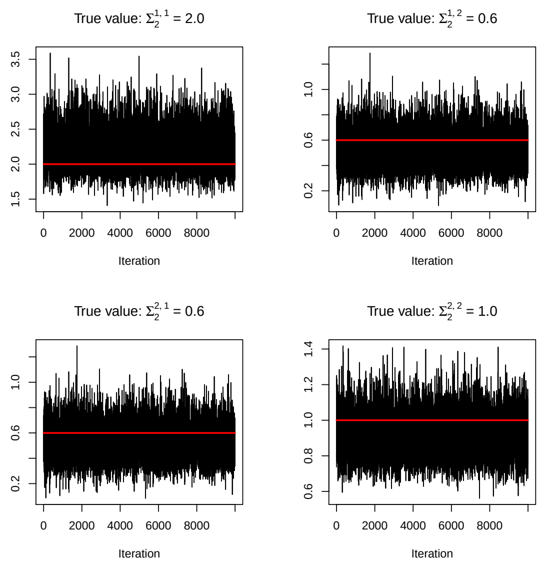


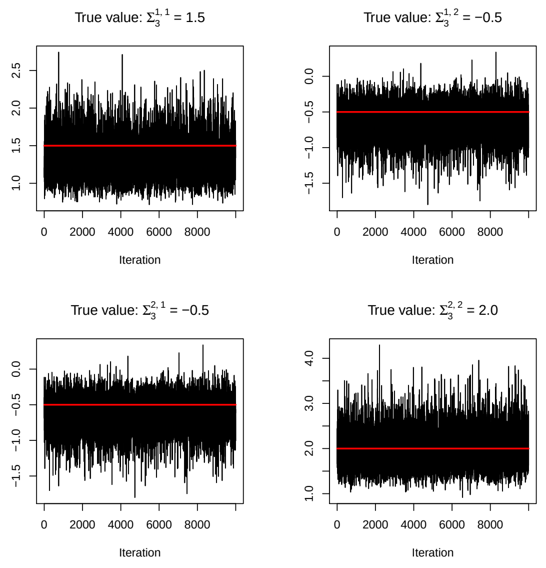
Figure D.18: Histograms of $\boldsymbol{\mu}$ for the AM algorithm.Each of the true values for the parameters is represented by the red line.



(a)

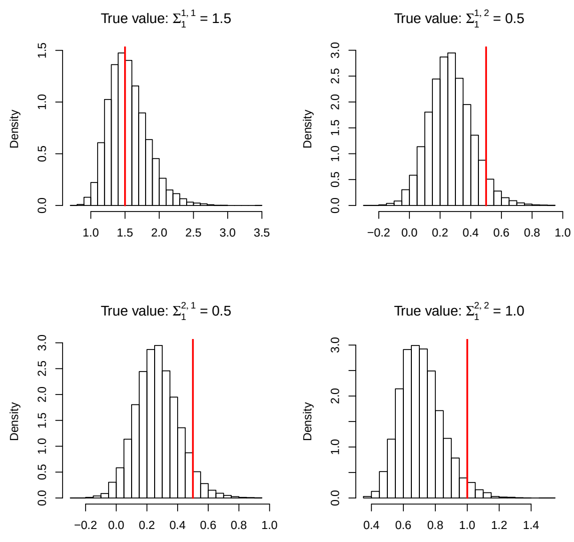


(b)

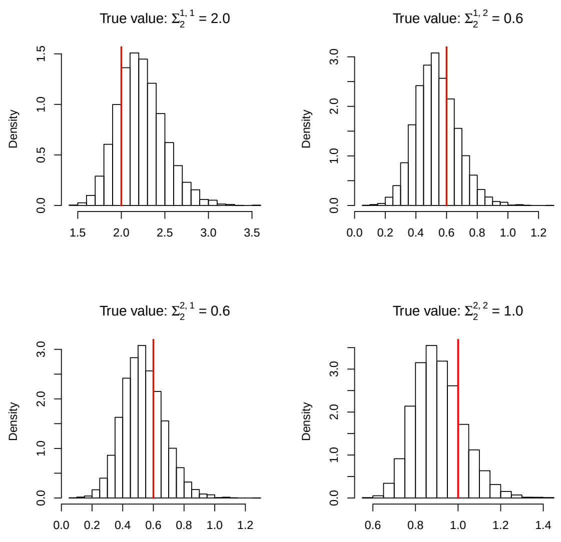


(c)

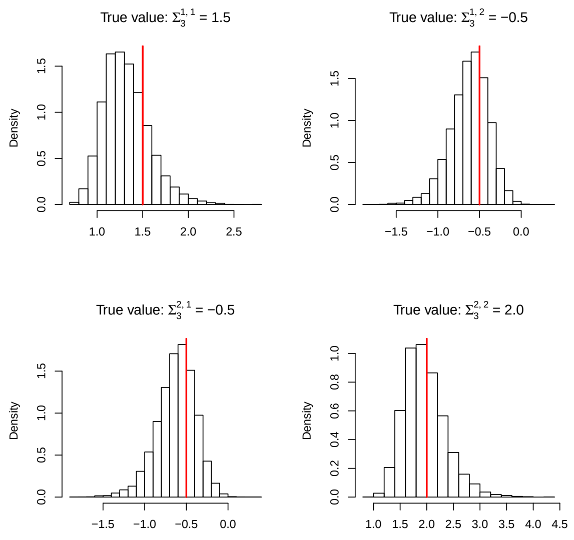
Figure D.19: Trace plots of Σ_1 (a), Σ_2 (b), and Σ_3 (a), for the AM algorithm. Each of the true values for the parameters is represented by the red line.



(a)

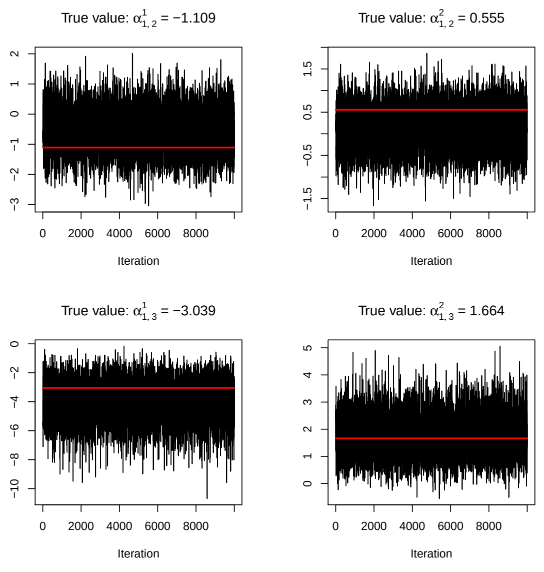


(b)

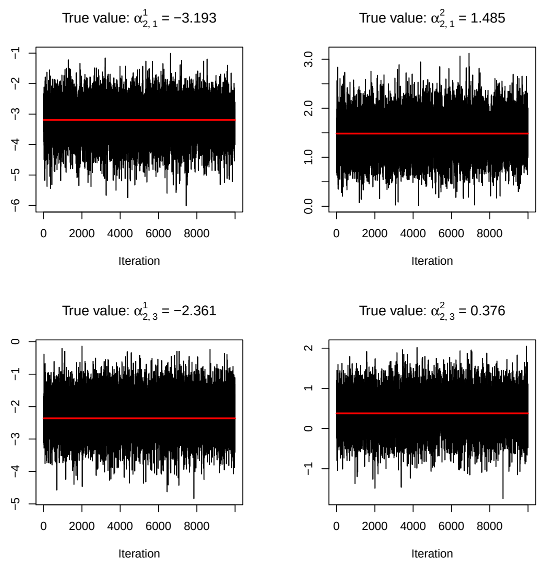


(c)

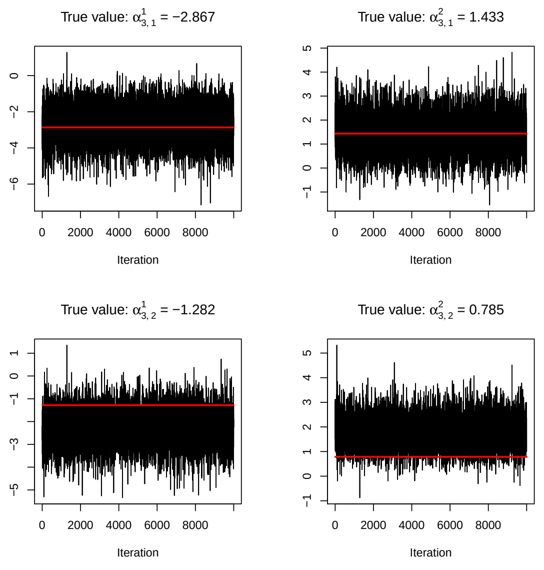
Figure D.20: Histograms of Σ_1 (a), Σ_2 (b), and Σ_3 (a), for the AM algorithm. Each of the true values for the parameters is represented by the red line.



(a)

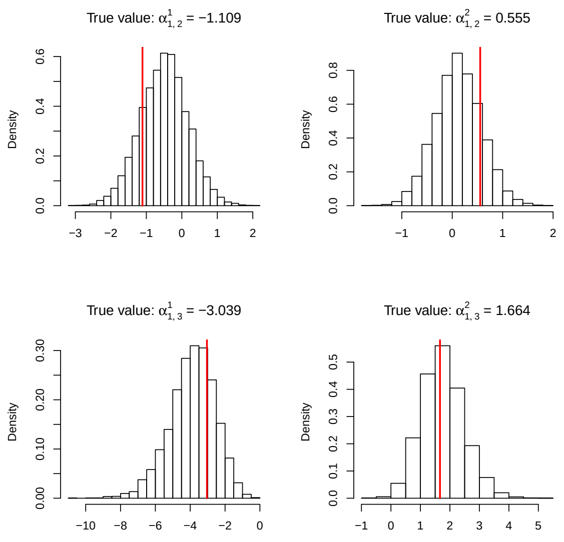


(b)

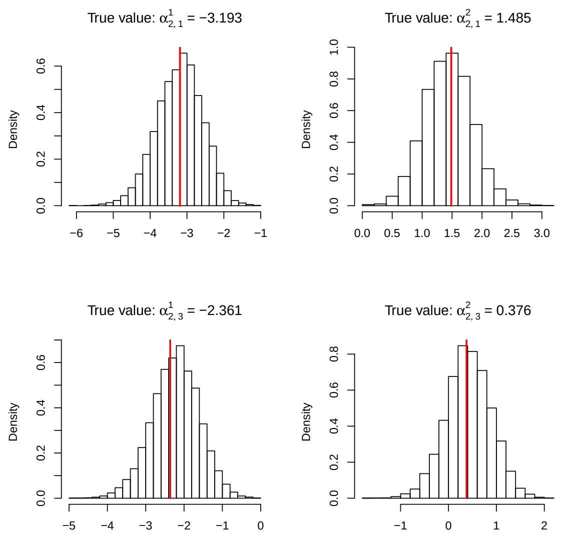


(c)

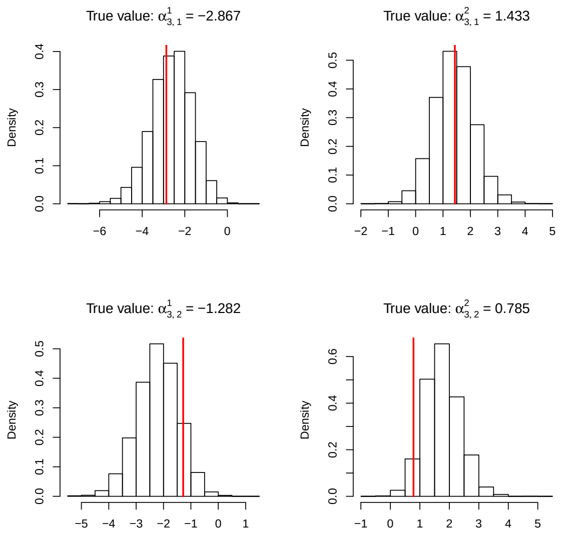
Figure D.21: Trace plots of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the AM algorithm. Each of the true values for the parameters is represented by the red line.



(a)



(b)



(c)

Figure D.22: Histograms of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the AM algorithm. Each of the true values for the parameters is represented by the red line.

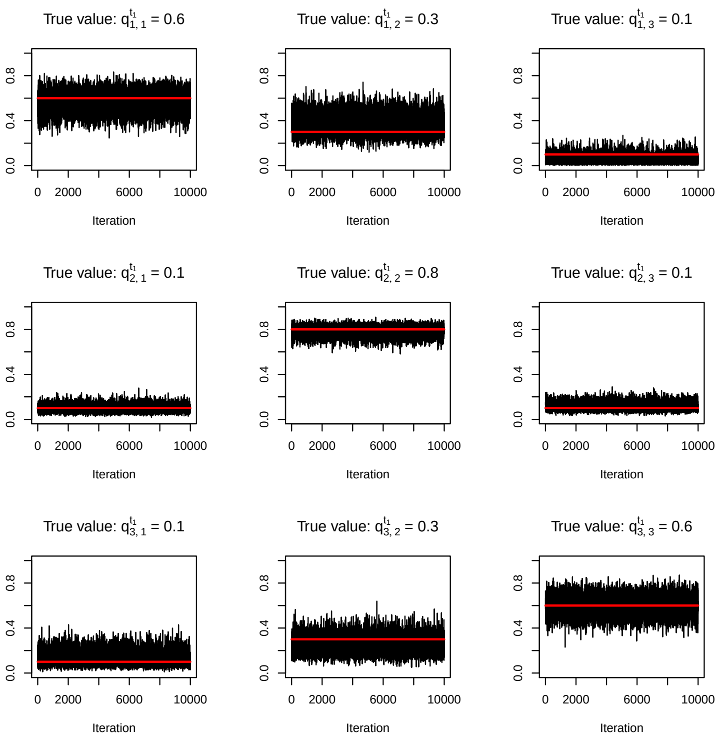


Figure D.23: Trace plots of Q^{t_1} for the AM algorithm. Each of the true values for the parameters is represented by the red line.

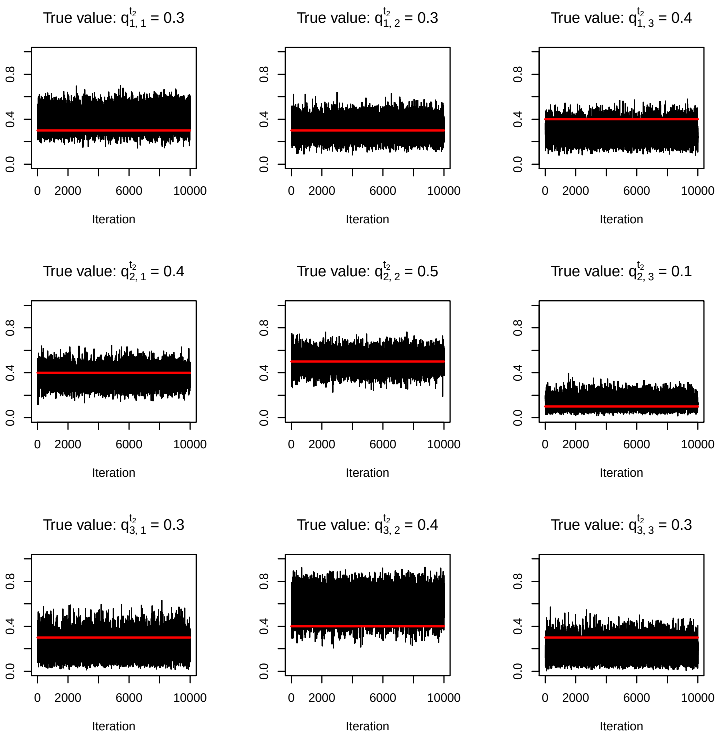


Figure D.24: Trace plots of Q^{t_2} for the AM algorithm. Each of the true values for the parameters is represented by the red line.

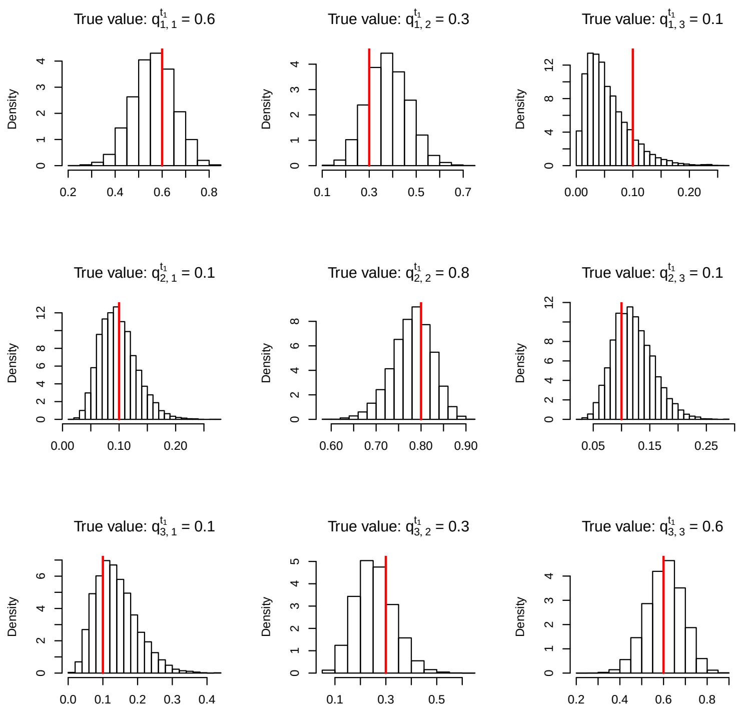


Figure D.25: Histograms of Q^{t_1} for the AM algorithm. Each of the true values for the parameters is represented by the red line.

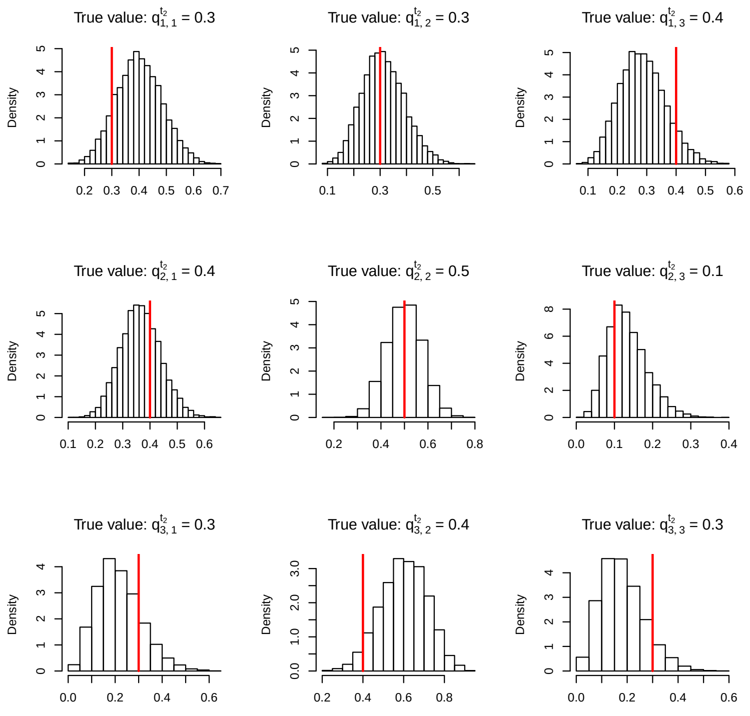


Figure D.26: Histograms of Q^{t_2} for the AM algorithm. Each of the true values for the parameters is represented by the red line.

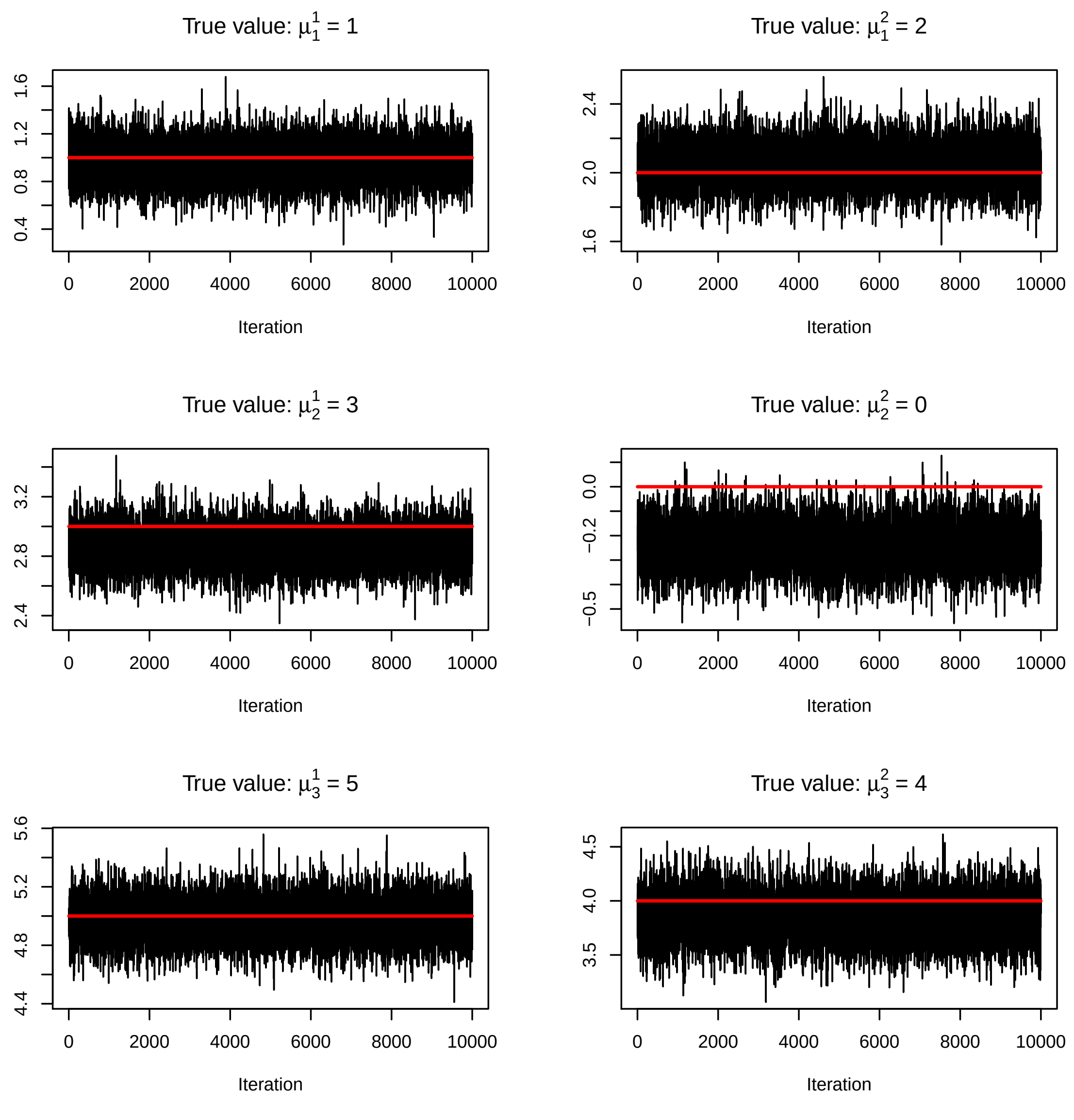


Figure D.27: Trace plots of $\boldsymbol{\mu}$ for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.

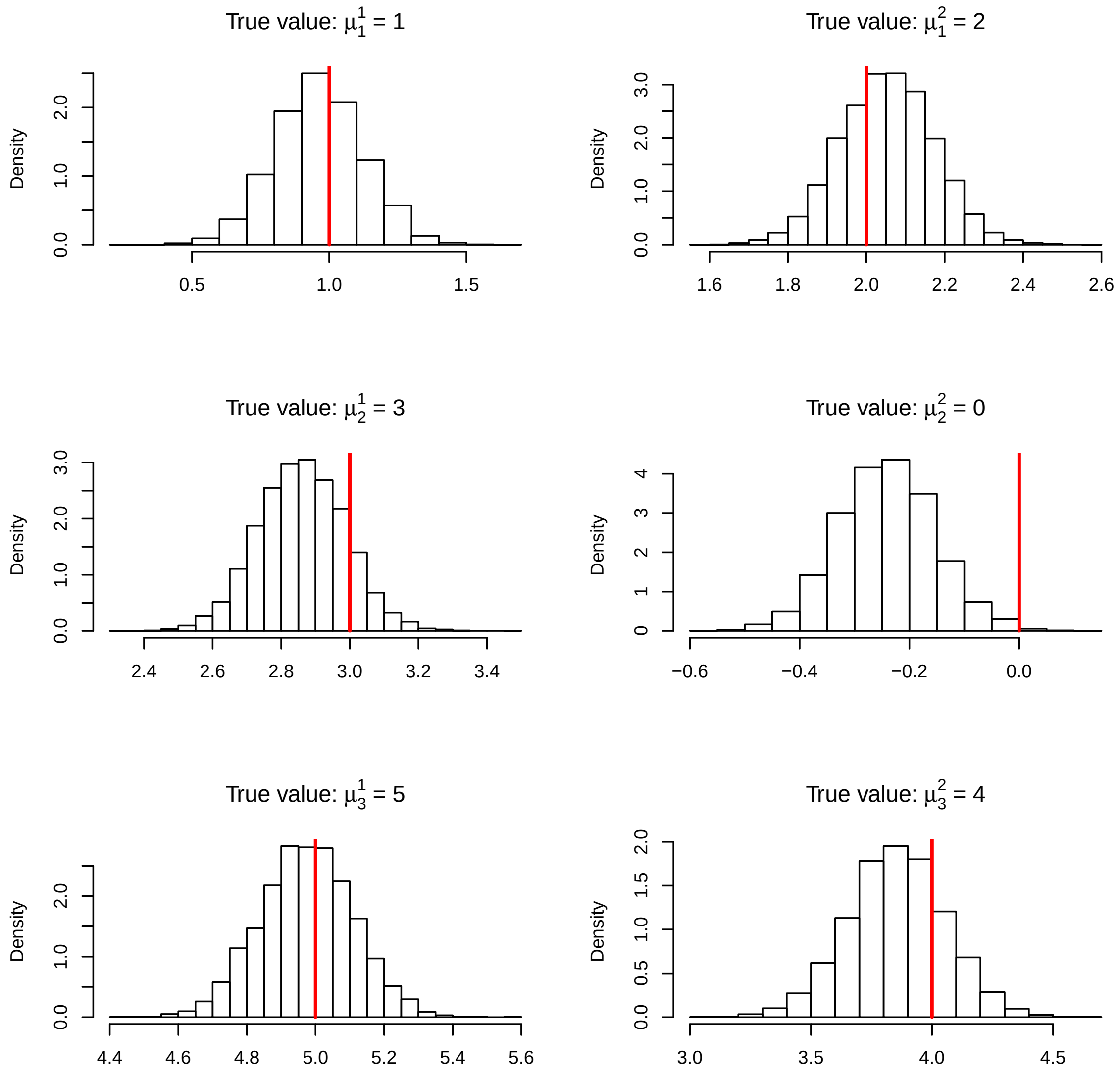
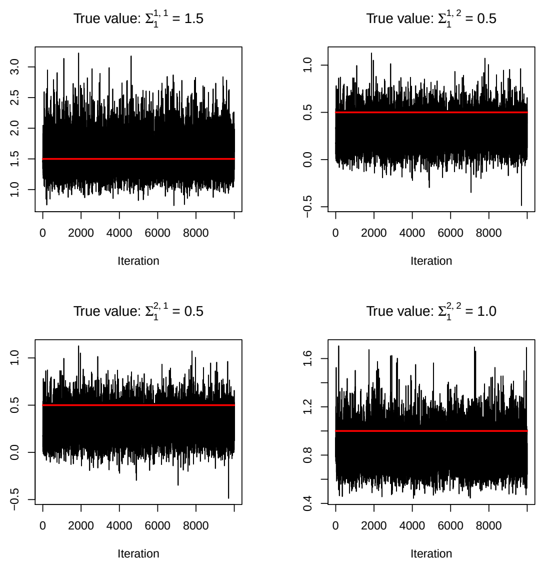
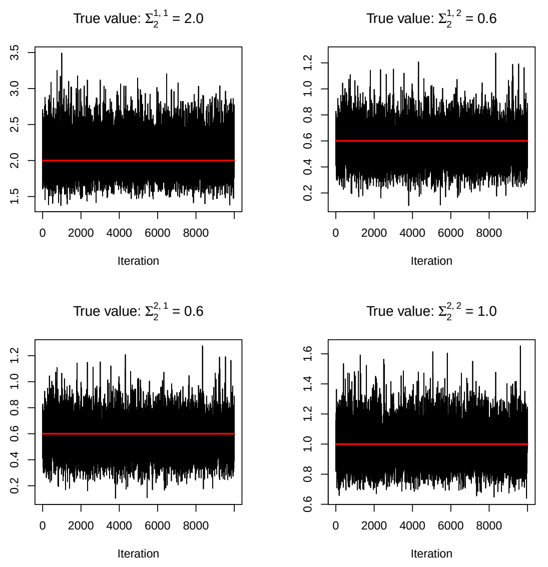


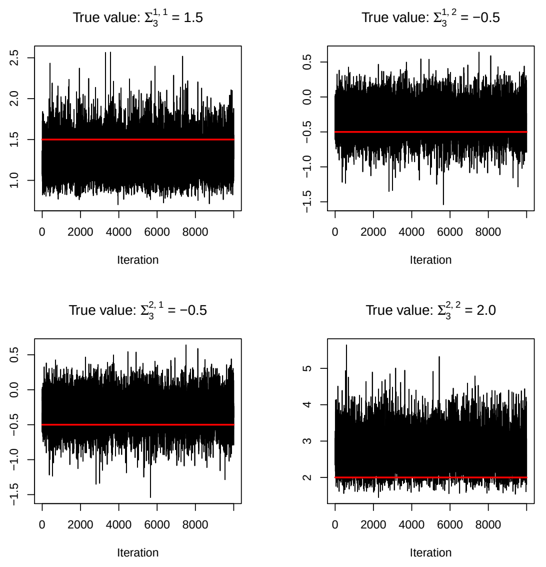
Figure D.28: Histograms of $\boldsymbol{\mu}$ for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.



(a)

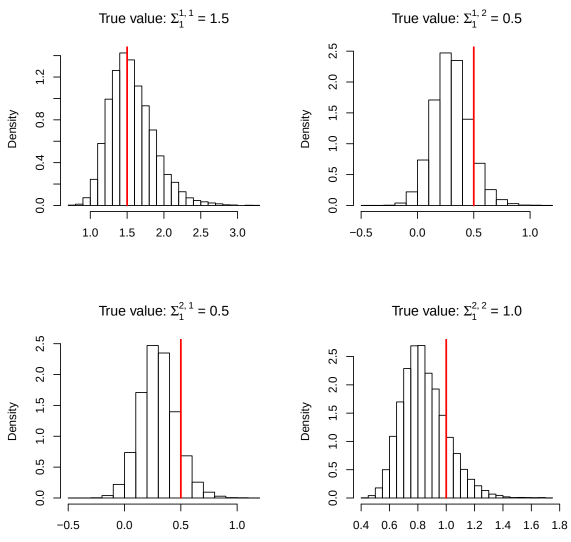


(b)

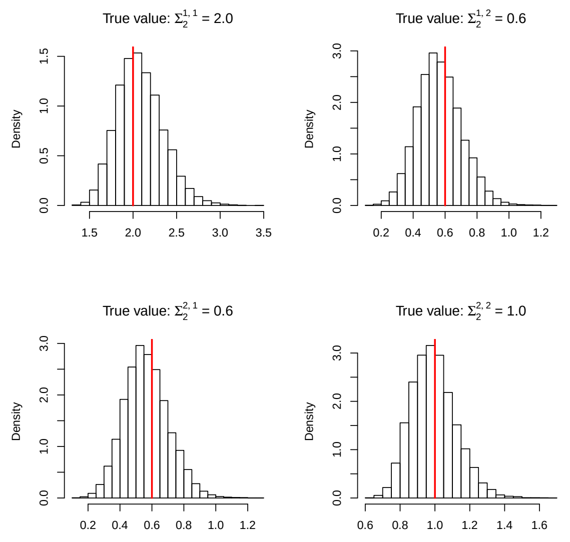


(c)

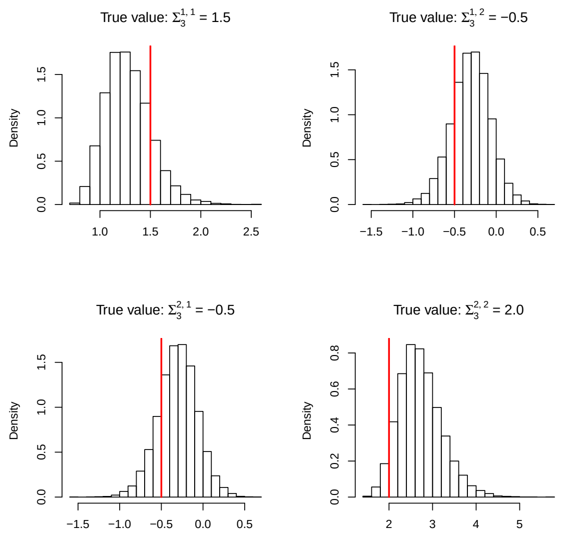
Figure D.29: Trace plots of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.



(a)

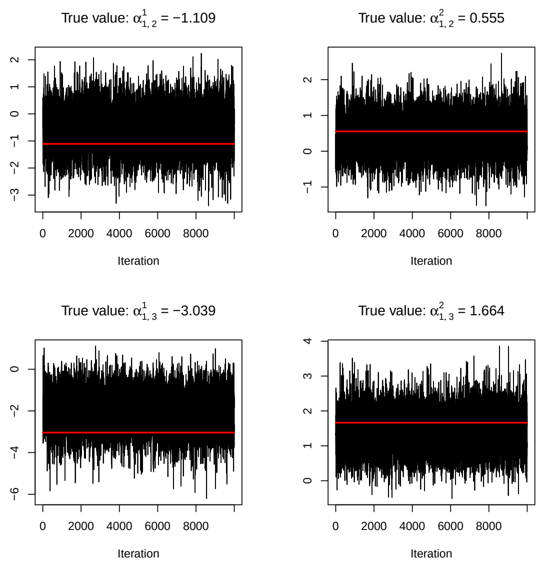


(b)

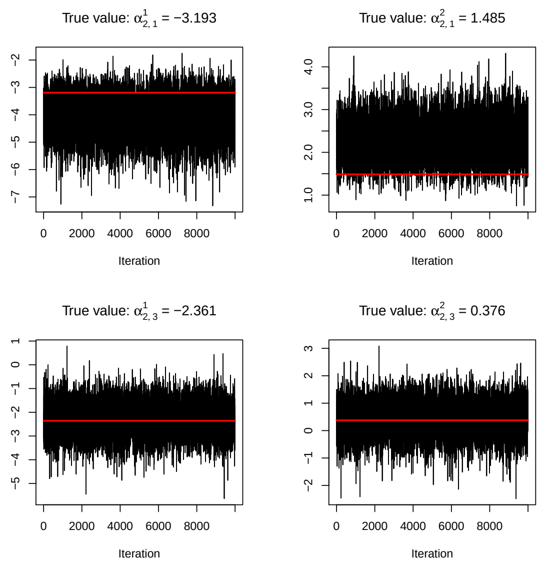


(c)

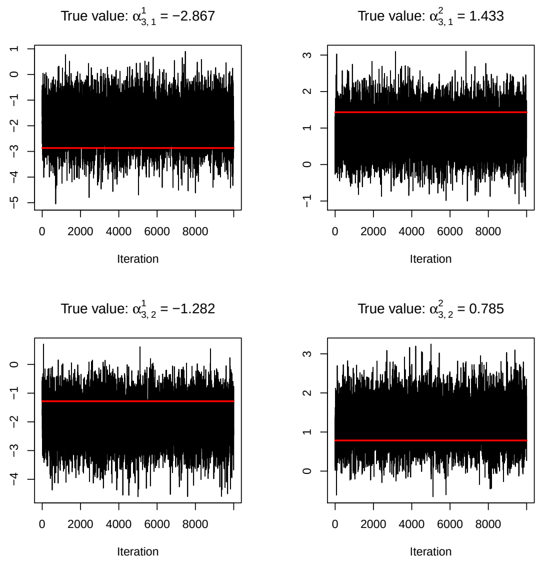
Figure D.30: Histograms of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.



(a)

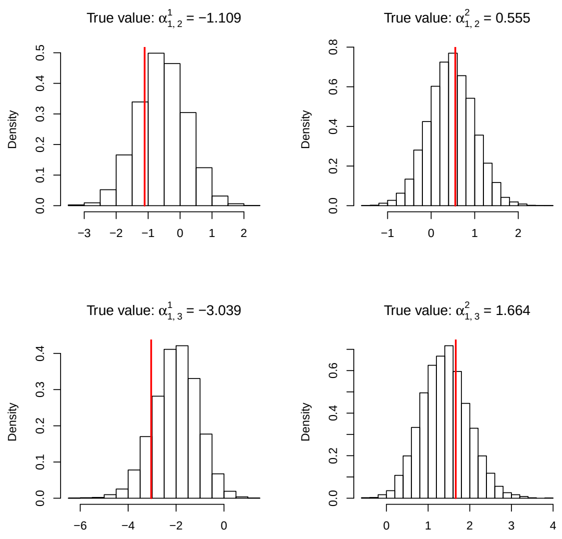


(b)

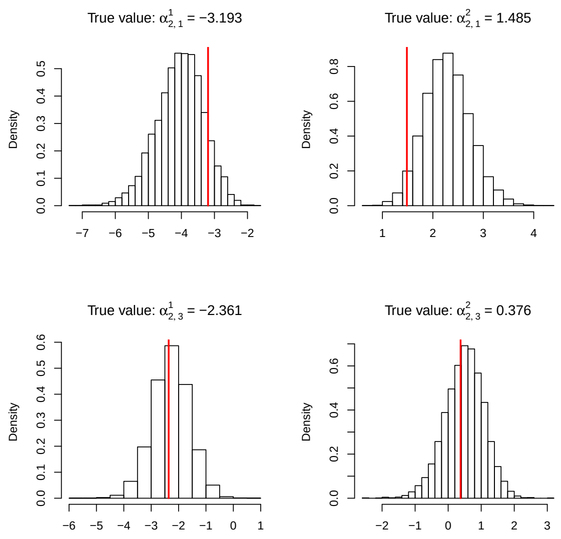


(c)

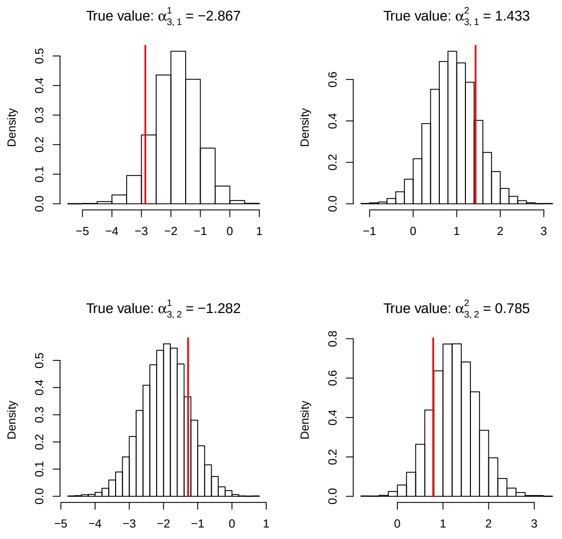
Figure D.31: Trace plots of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.



(a)



(b)



(c)

Figure D.32: Histograms of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), and $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.

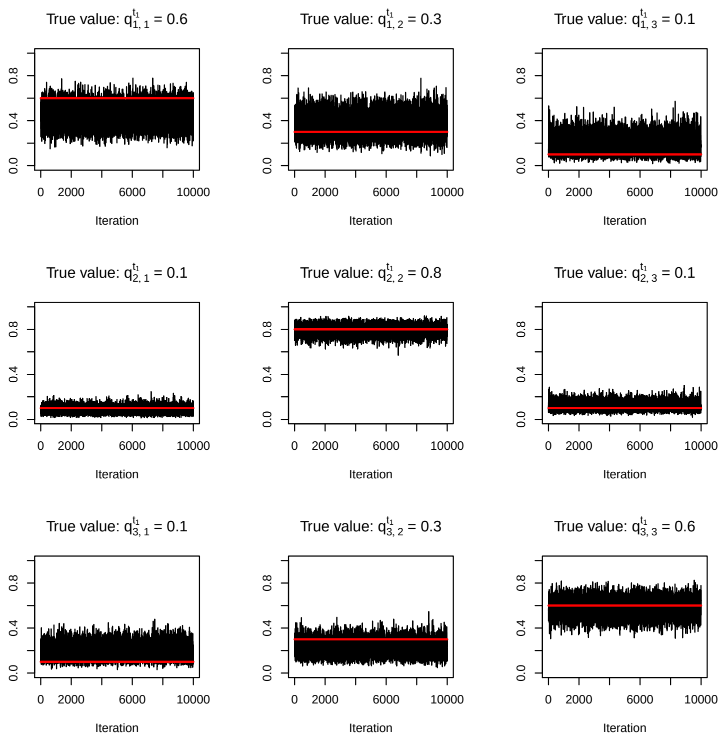


Figure D.33: Trace plots of Q^{t_1} for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.

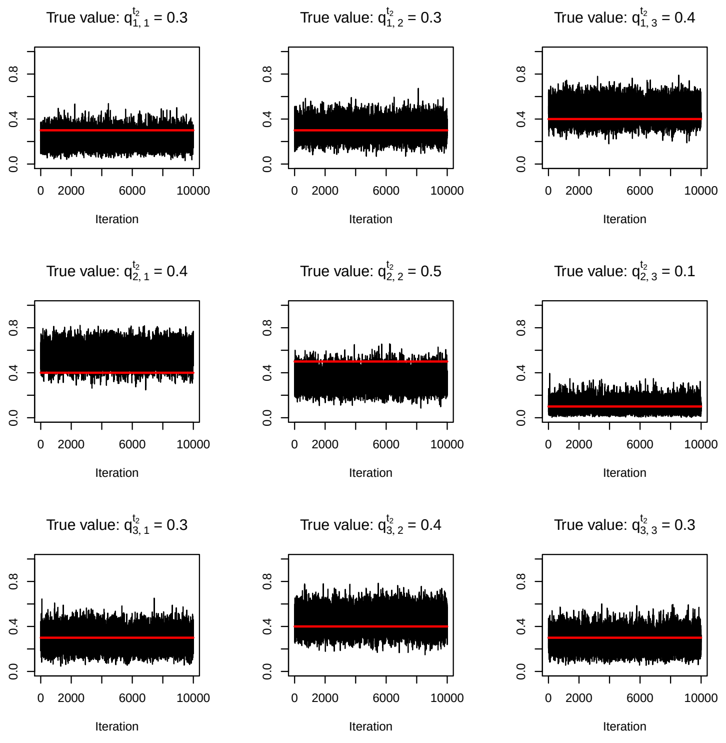


Figure D.34: Trace plots of Q^{t_2} for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.

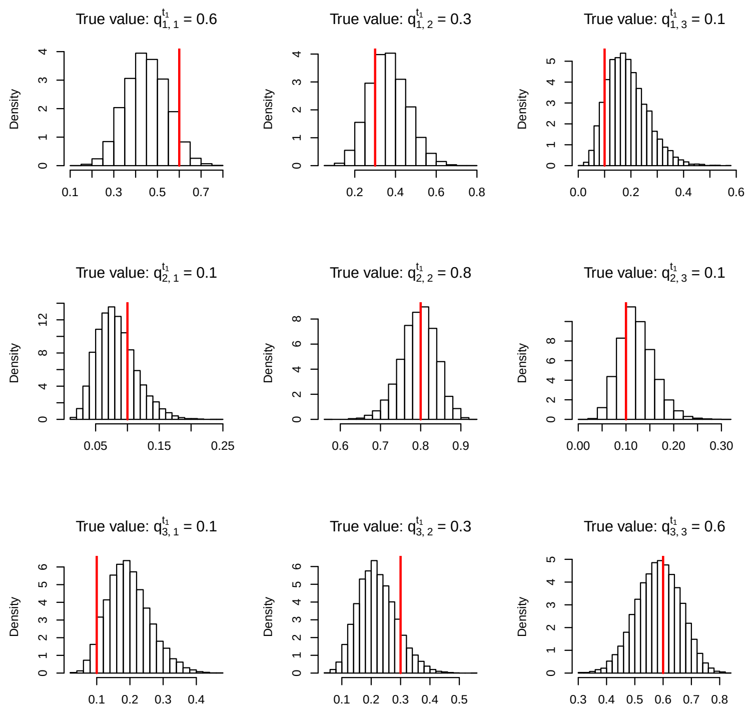


Figure D.35: Histograms of Q^{t_1} for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.

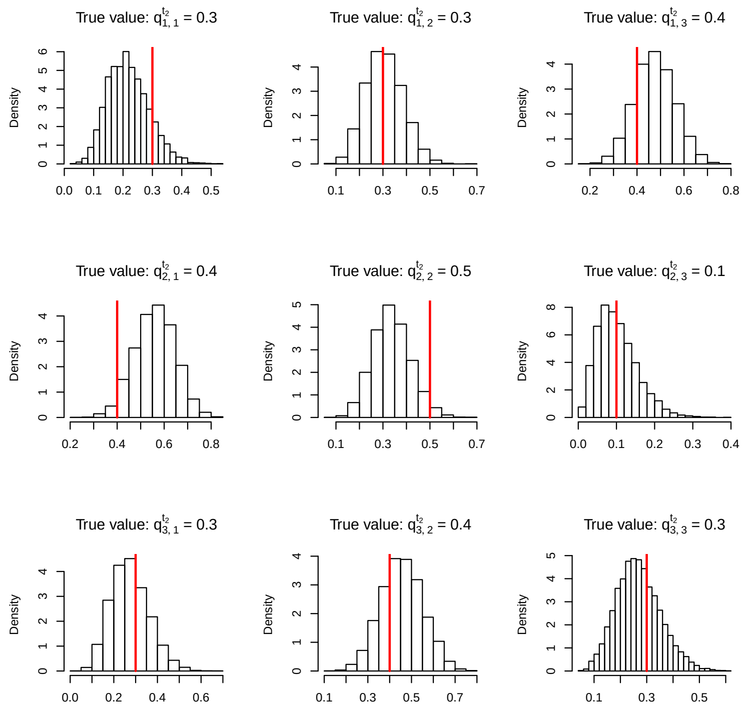


Figure D.36: Histograms of Q^{t_2} for the symmetric DRAM algorithm. Each of the true values for the parameters is represented by the red line.

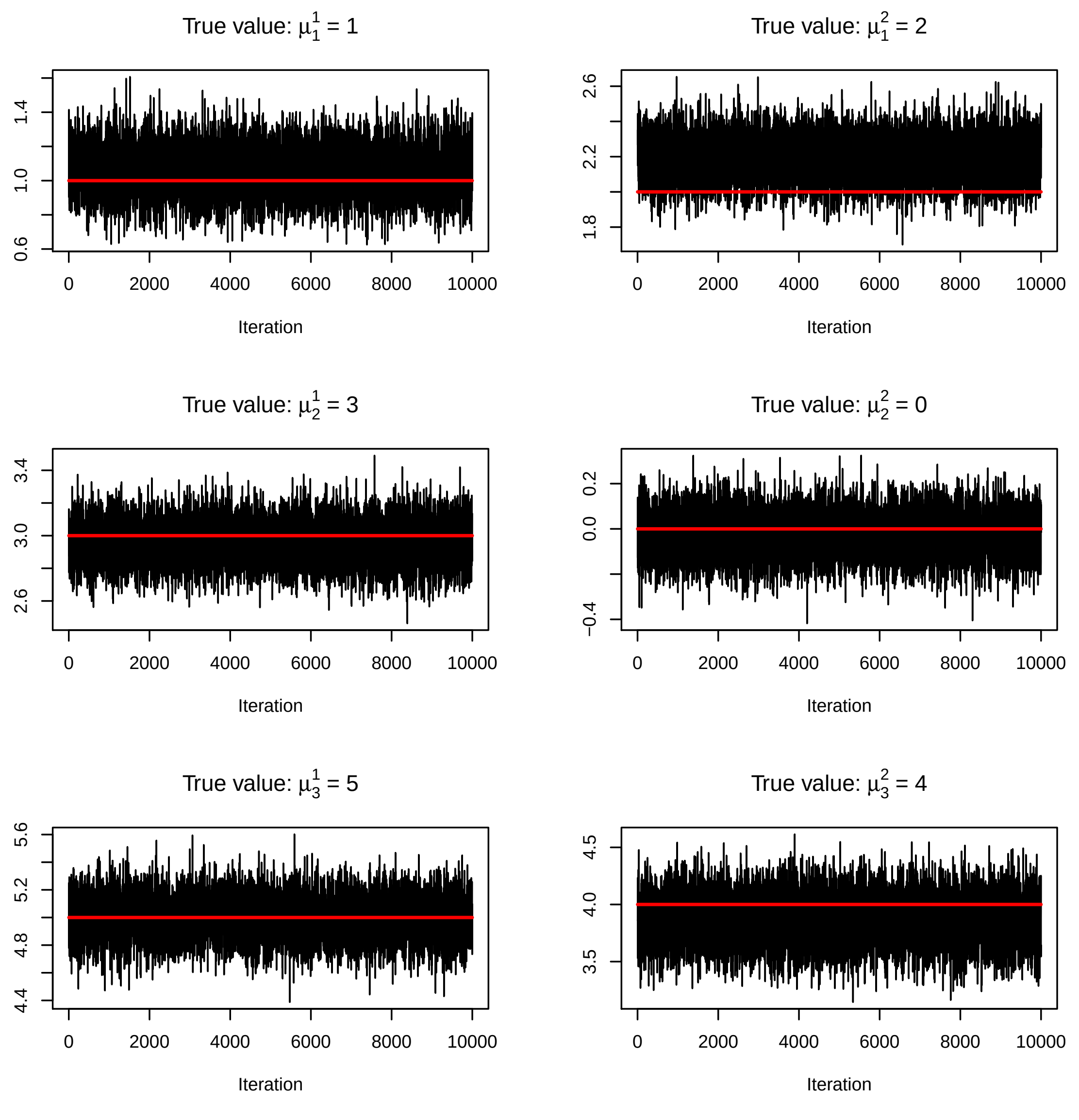


Figure D.37: Trace plots of $\boldsymbol{\mu}$ for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.

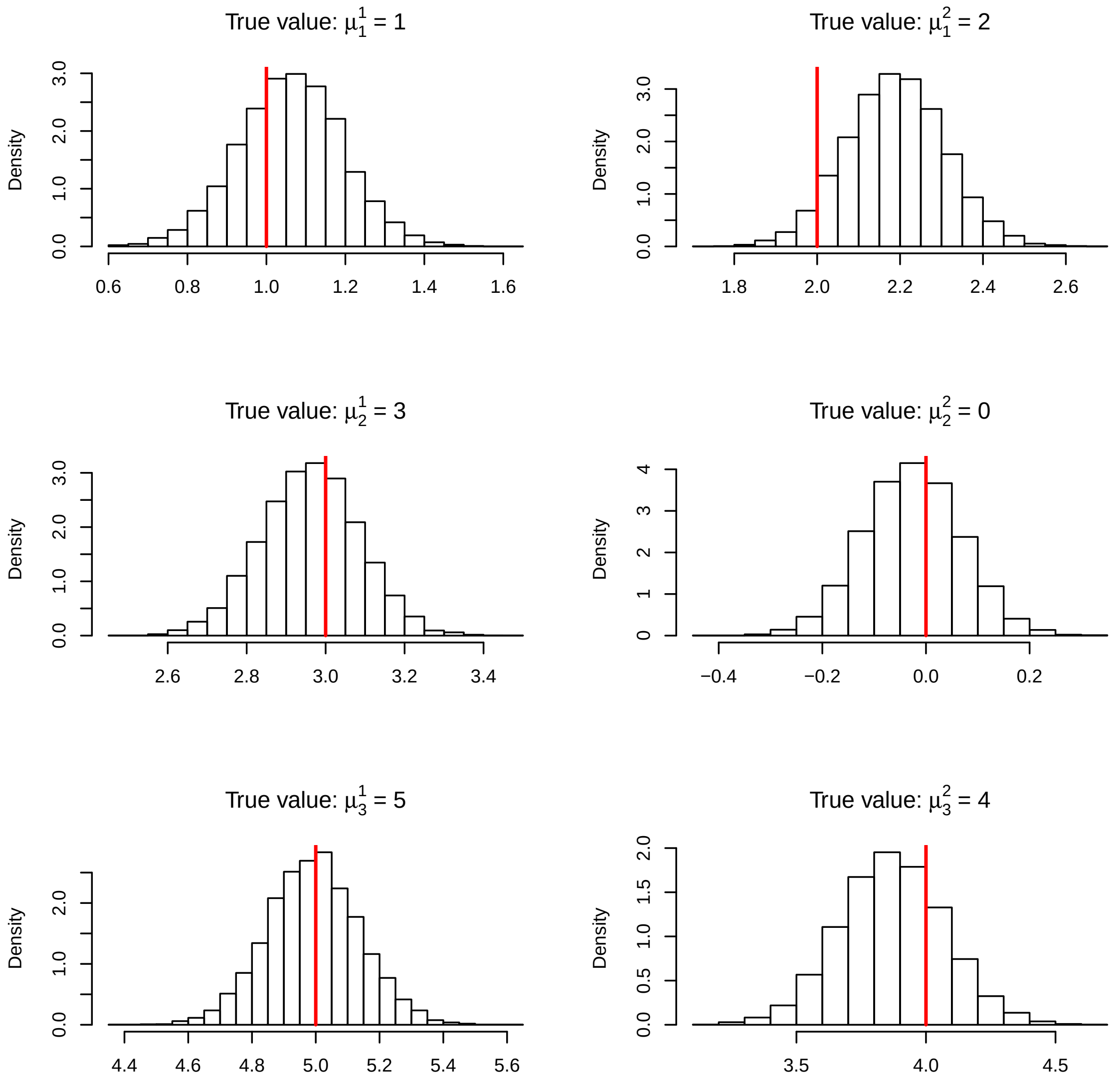
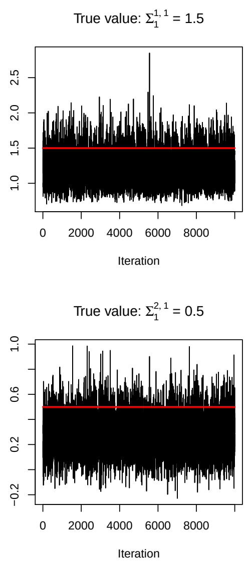
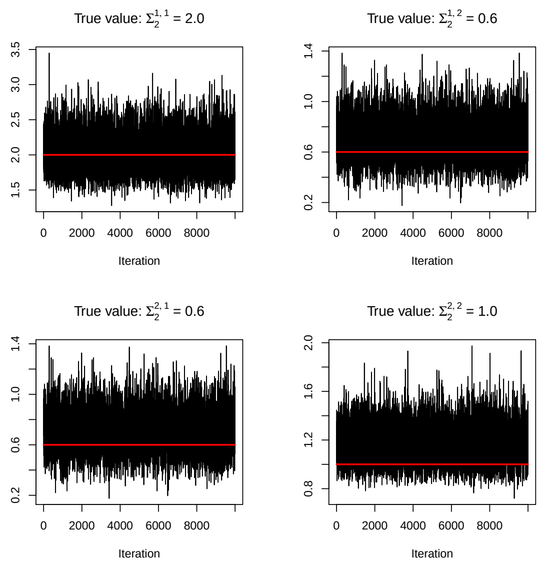


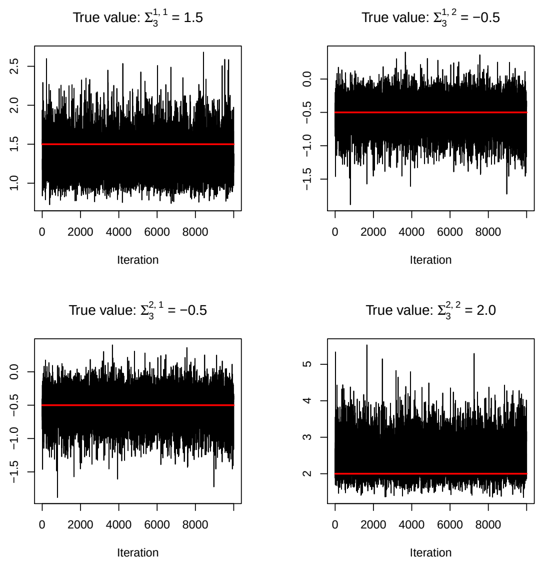
Figure D.38: Histograms of $\boldsymbol{\mu}$ for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.



(a)

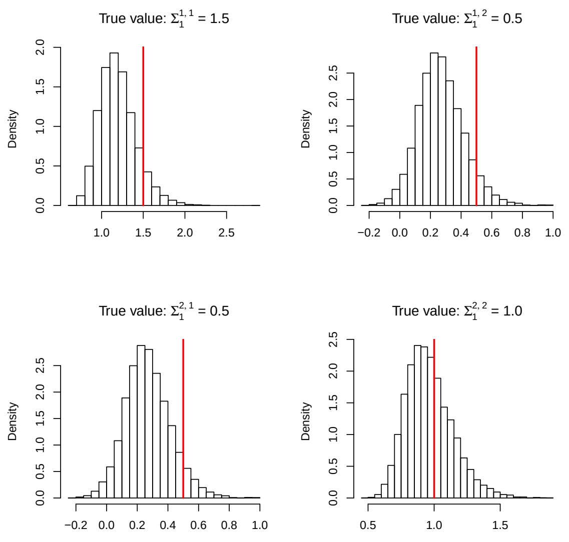


(b)

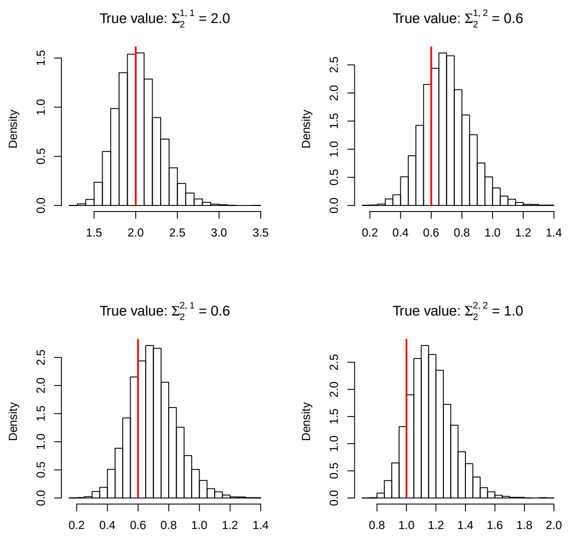


(c)

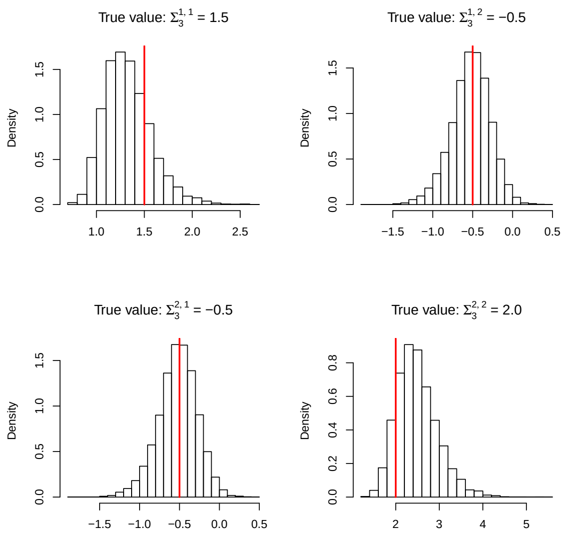
Figure D.39: Trace plots of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.



(a)

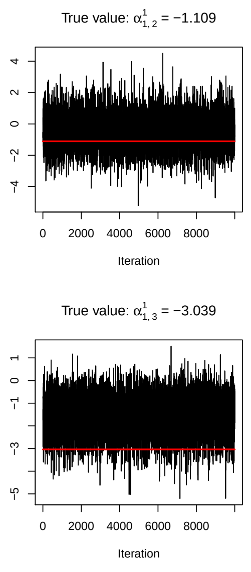


(b)

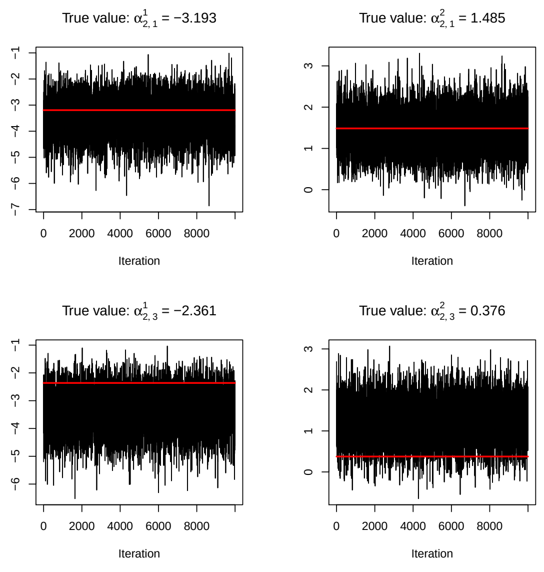


(c)

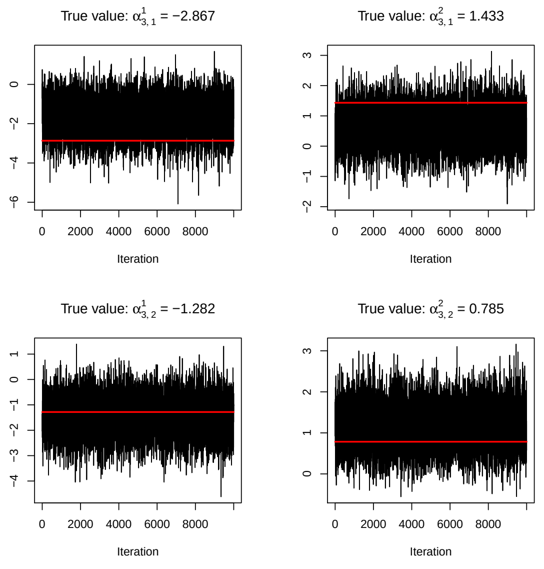
Figure D.40: Histograms of Σ_1 (a), Σ_2 (b), and Σ_3 (c), for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.



(a)

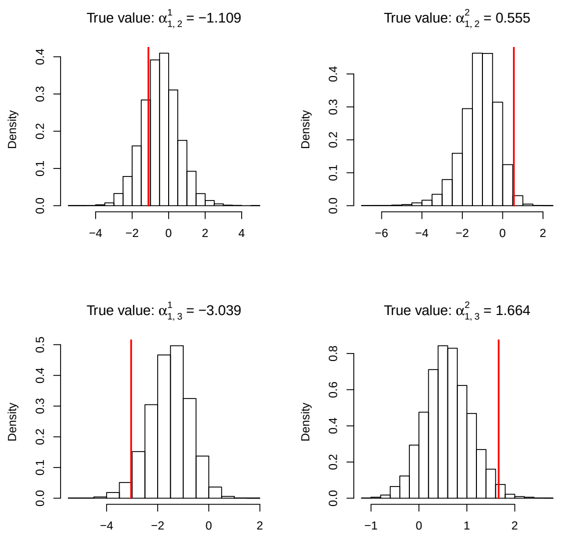


(b)

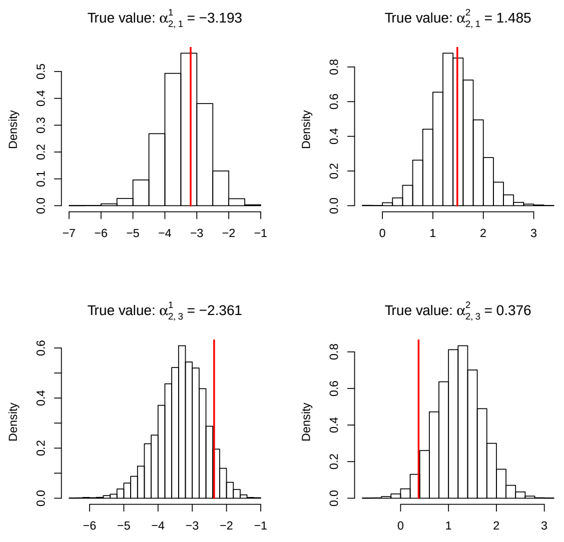


(c)

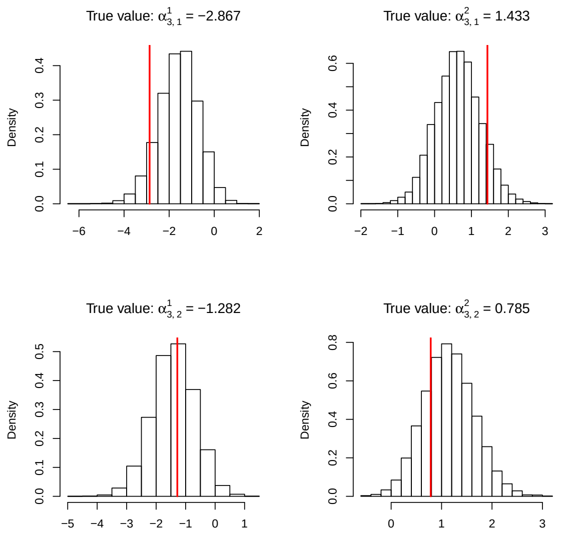
Figure D.41: Trace plots of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.



(a)



(b)



(c)

Figure D.42: Histograms of $\alpha_{1,2}$ & $\alpha_{1,3}$ (a), $\alpha_{2,1}$ & $\alpha_{2,3}$ (b), $\alpha_{3,1}$ & $\alpha_{3,2}$ (c), for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.

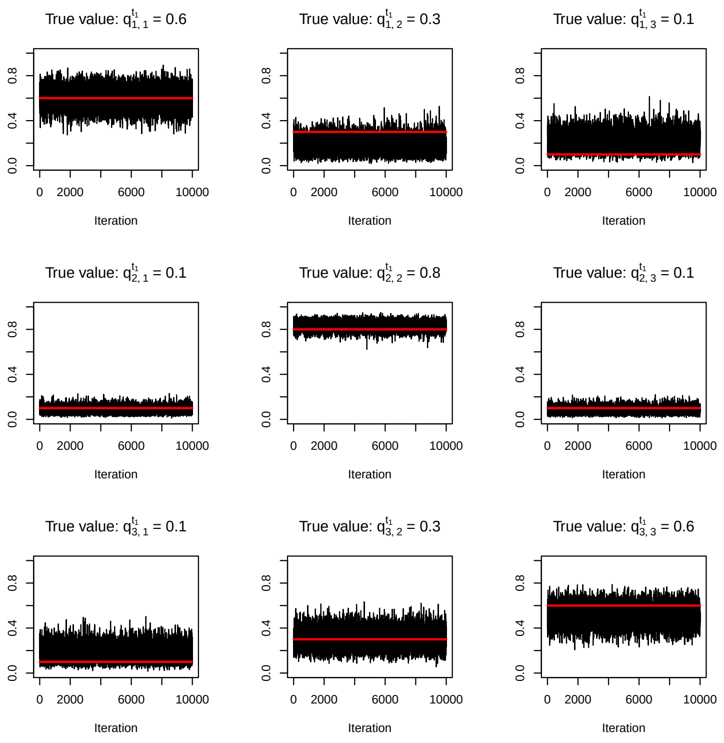


Figure D.43: Trace plots of Q^{t_1} for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.

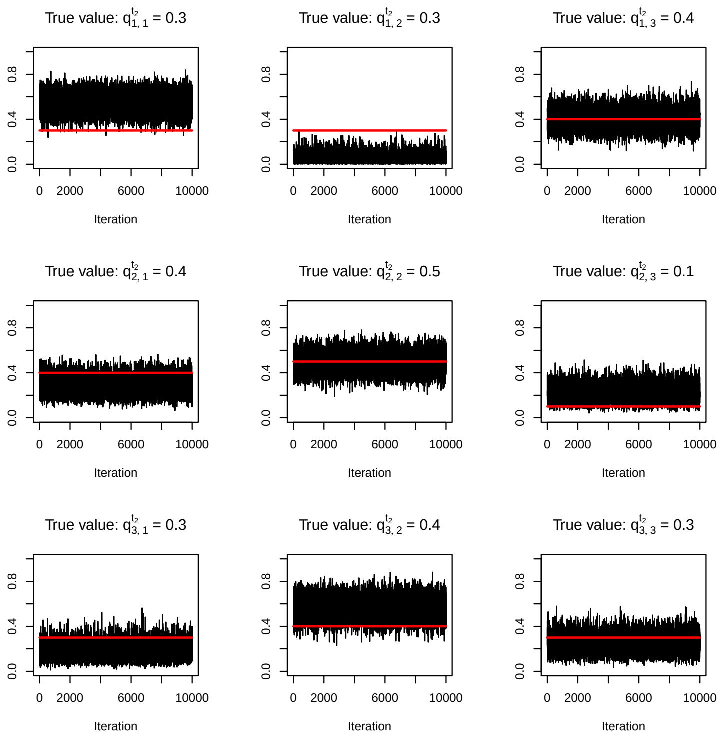


Figure D.44: Trace plots of Q^{t_2} for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.

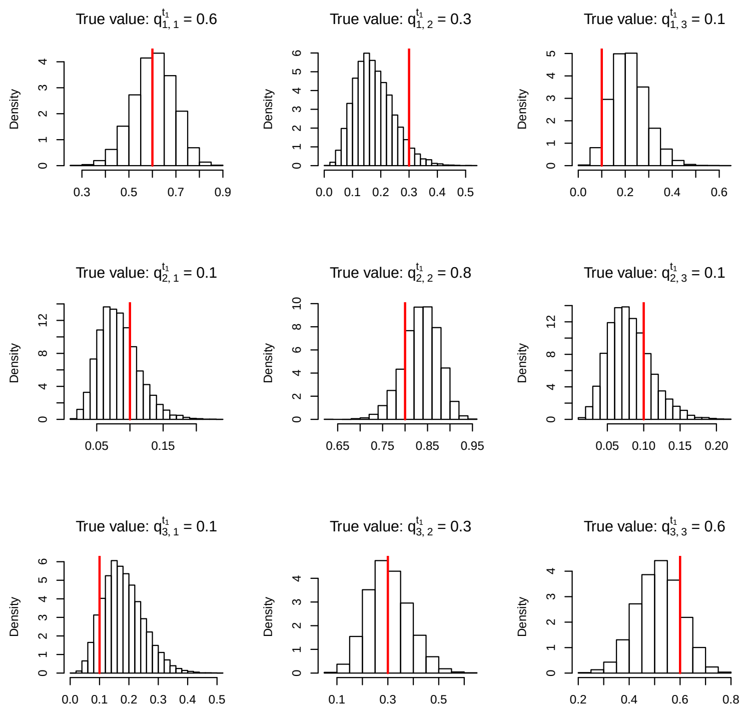


Figure D.45: Histograms of Q^{t_1} for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.

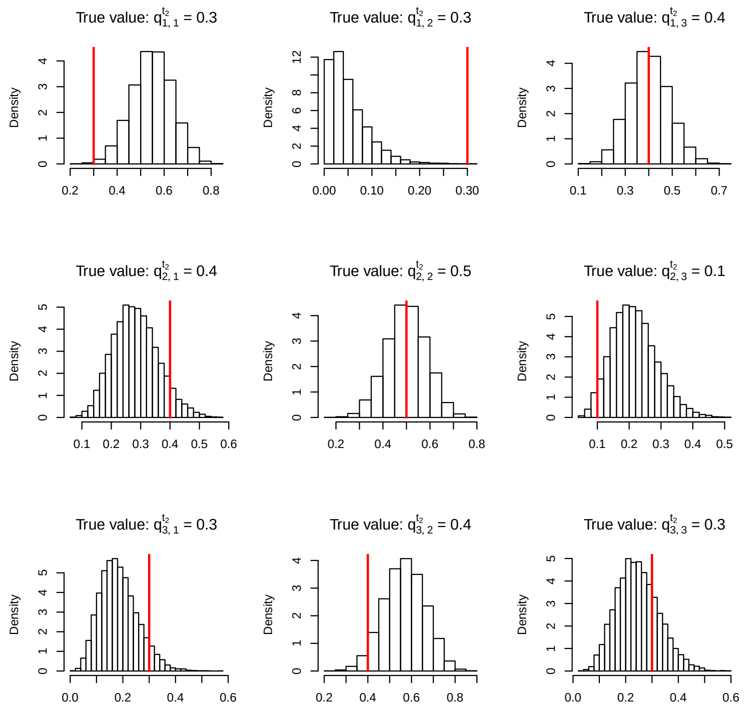


Figure D.46: Histograms of Q^{t_2} for the MTAM algorithm. Each of the true values for the parameters is represented by the red line.

Appendix E

Source Code: Univariate NHGHMM

E.1 R©

```
simulate_NHHMM <- function(n,mu,sigma,Q){
#------------------------------------------------------------
# This function simulates a non-homogeneous Gaussian hidden
    Markov model and returns a sequence of the hidden
   states and observations.
#------------------------------------------------------------
# Inputs
#------------------------------------------------------------
# - n: the number of observations
# - mu: a vector of mu
# - sigma: a vector of sigma
# - Q: the transition matrix/matrices
#------------------------------------------------------------
# Outputs
#------------------------------------------------------------
# - state: a sequence of the hidden states
# - obs: a sequence of the observations
#-----------------------------------------------------------
  if(dim(Q)[1] != dim(Q)[2]){
    stop("The transition matrix is not square.")
  }
  m <- dim(Q)[1]
  state <- numeric(n)
  obs <- numeric(n)
  state[1] <- sample(1:m,size=1,prob=rep(1/m,m))
  obs[1] <- rnorm(1,mean=mu[state[1]],sd=sigma[state[1]])
# Homogeneous transition probabilities
  if(is.matrix(Q)){
    for(t in 2:n){
      state[t] <- sample(1:m,size=1,prob=Q[state[t-1],])
      obs[t] <- rnorm(1,mean=mu[state[t]],sd=sigma[state[t
         ]])
    }
  }
# Non-homogeneous transition probabilities
  else{
    for(t in 2:n){
      state[t] <- sample(1:m,size=1,prob=Q[state[t-1],,t])
      obs[t] <- rnorm(1,mean=mu[state[t]],sd=sigma[state[t
         ]])
    }
  }
  result <- cbind(state,obs)
  return(result)
}
```

```
sampleAlpha <- function(j,state,Q,z,A,mu_A,Sigma_A,E){
#-----------------------------------------------------------
```

```
# This function samples the parameter, alpha, from the
   posterior distribution of interest, f.
#-----------------------------------------------------------
# Inputs
#-----------------------------------------------------------
# - j: the index from 1 to m
# - state: the estimated sequence of the hidden states at
   the current MCMC iteration
# - Q: the current estimate of the transition probabilities
# - z: the exogenous variables
# - A: the parameter, alpha, at the current MCMC iteration
# - mu_A: the vector of means, mu, for the multivariate
   Gaussian distribution
# - Sigma_A: the covariance matrix, Sigma, for the
   multivariate Gaussian distribution
# - E: the covariance matrix for random walk Metropolis
   updates
#-----------------------------------------------------------
# Outputs
#-----------------------------------------------------------
# - alpha: the new proposed parameter, alpha
# - prob: the acceptance probability of the proposed alpha
#-----------------------------------------------------------
# Define the posterior density function
  f <- function(A){
    prod((exp(z%*%A)/rowSums(exp(z%*%A)))[mat_dims])*exp(-1
       /2*sum(diag((A-mu_A)%*%t(A-mu_A)%*%Sigma_A)))
  }
  m <- dim(Q)[1]
  n <- length(state)
  ind <- which(state[-n]==j)
  mat_ind <- (1:m)[-j]
  mat_dims <- matrix(c(ind+1,state[ind+1]),ncol=2)
# Metropolis-Hastings update
  A_curr <- A
  if(length(mat_ind) < 2){
    A[,mat_ind] <- mvrnormArma(1,A_curr[,mat_ind],E)
  }
  else{
    A[,mat_ind] <- apply(A_curr[,mat_ind],2,function(x)
         mvrnormArma(1,x,E))
  }
# Calculate the acceptance ratio
  P <- f(A)/f(A_curr)
  acc_prob <- min(1,P)
  if(is.nan(acc_prob)){
    A_prop <- A
    acc_prob <- 1
  }
  else{
    if(runif(1) <= acc_prob){
      A_prop <- A
    }
    else{
      A_prop <- A_curr
    }
  }
  result <- list(alpha=A_prop,prob=acc_prob)
  return(result)
}
```

```
sampleAlphaDR <- function(j,state,Q,z,A,mu_A,Sigma_A,E,STG)
  {
#----------------------------------------------------------
# This function samples the parameter, alpha, from the
  posterior distribution of interest, f.
#----------------------------------------------------------
# Inputs
#----------------------------------------------------------
# - j: the index from 1 to m
# - state: the estimated sequence of the hidden states at
  the current MCMC iteration
# - Q: the current estimate of the transition probabilities
# - z: the exogenous variables
# - A: the parameter, alpha, at the current MCMC iteration
# - mu_A: the vector of means, mu, for the multivariate
  Gaussian distribution
# - Sigma_A: the covariance matrix, Sigma, for the
  multivariate Gaussian distribution
# - E: the covariance matrix for random walk Metropolis
  updates
# - STG: the number of delayed rejection stages
#----------------------------------------------------------
# Outputs
#----------------------------------------------------------
# - alpha: the new proposed parameter, alpha
# - prob: the acceptance probability of the proposed alpha
#----------------------------------------------------------
# Define the posterior density function
  f <- function(A){
    prod((exp(z%*%A)/rowSums(exp(z%*%A)))[mat_dims])*exp(-1
      /2*sum(diag((A-mu_A)%*%t(A-mu_A)%*%Sigma_A)))
  }
  m <- dim(Q)[1]
  n <- length(state)
  K <- dim(A)[1]
  ind <- which(state[-n]==j)
  mat_ind <- (1:m)[-j]
  mat_dims <- matrix(c(ind+1,state[ind+1]),ncol=2)
  X <- A
  Y <- array(0,dim=c(K,m,STG))
# Metropolis-Hastings update (a single column)
  if(length(mat_ind) < 2){
    Y[,mat_ind,1] <- mvrnormArma(1,X[,mat_ind],E)
    Y_star <- Y[,,1]
    P <- f(Y[,,1])/f(X)
    acc_prob <- min(1,P)
    if(is.nan(acc_prob)){
      A_prop <- Y[,,1]
      acc_prob <- 1
    }
    else{
      if(f(Y[,,1]) >= f(X) || runif(1) <= acc_prob){
        A_prop <- Y[,,1]
      }
      else{
        for(stg in 2:STG){
          Y[,mat_ind,stg] <- mvrnormArma(1,Y[,mat_ind,stg
             -1],E)
          P <- (f(Y[,,stg]) - f(Y_star))/(f(X) - f(Y_star))
          if(f(Y[,,stg]) > f(Y_star)){
```

```
          Y_star <- Y[,,stg]
        }
        acc_prob <- min(1,P)
        if(is.nan(acc_prob)){
          A_prop <- Y[,,stg]
          acc_prob <- 1
          break
        }
        else{
          if(f(Y[,,stg]) >= f(X) || runif(1) <= acc_prob)
             {
            A_prop <- Y[,,stg]
            break
          }
          if(stg == STG){
            A_prop <- X
            break
          }
          next
        }
      }
    }
  }
}
# Metropolis-Hastings update (mulptile columns)
  else{
    Y[,mat_ind,1] <- apply(X[,mat_ind],2,function(x)
        mvrnormArma(1,x,E))
    Y_star <- Y[,,1]
    P <- f(Y[,,1])/f(X)
  acc_prob <- min(1,P)
  if(is.nan(acc_prob)){
    A_prop <- Y[,,1]
    acc_prob <- 1
  }
  else{
    if(f(Y[,,1]) >= f(X) || runif(1) <= acc_prob){
      A_prop <- Y[,,1]
    }
    else{
      for(stg in 2:STG){
        Y[,mat_ind,stg] <- apply(Y[,mat_ind,stg-1],2,
           function(x) mvrnormArma(1,x,E))
        P <- (f(Y[,,stg]) - f(Y_star))/(f(X) - f(Y_star))
        if(f(Y[,,stg]) > f(Y_star)){
          Y_star <- Y[,,stg]
        }
        acc_prob <- min(1,P)
        if(is.nan(acc_prob)){
          A_prop <- Y[,,stg]
          acc_prob <- 1
        }
        else{
          if(f(Y[,,stg]) >= f(X) || runif(1) <= acc_prob)
             {
            A_prop <- Y[,,stg]
            break
          }
          if(stg == STG){
            A_prop <- X
```

```r
              break
            }
            next
          }
        }
      }
    }
  }
  result <- list(alpha=A_prop,prob=acc_prob)
  return(result)
}
```

```r
sampleAlphaMTM <- function(j,state,Q,z,A,mu_A,Sigma_A,E,d){
#----------------------------------------------------------
# This function samples the parameter, alpha, from the posterior distribution of interest, f.
#----------------------------------------------------------
# Inputs
#----------------------------------------------------------
# - j: the index from 1 to m
# - state: the estimated sequence of the hidden states at the current MCMC iteration
# - Q: the current estimate of the transition probabilities
# - z: the exogeneous variables
# - A: the parameter, alpha, at the current MCMC iteration
# - mu_A: the vector of means, mu, for the multivariate Gaussian distribution
# - Sigma_A: the covariance matrix, Sigma, for the multivariate Gaussian distribution
# - E: the covariance matrix for random walk Metropolis updates
# - d: the number of multiple tries for Metropolis updates
#------------------------------------------------------------
# Outputs
#------------------------------------------------------------
# - alpha: the new proposed parameter, alpha
# - prob: the acceptance probability of the proposed alpha
#------------------------------------------------------------
# Define the posterior density function
  f <- function(A){
    prod((exp(z%*%A)/rowSums(exp(z%*%A)))[mat_dims])*exp(-1
       /2*sum(diag((A-mu_A)%*%t(A-mu_A)%*%Sigma_A)))
  }
  m <- dim(Q)[1]
  n <- length(state)
  K <- ncol(z)
  ind <- which(state[-n]==j)
  mat_ind <- (1:m)[-j]
  mat_dims <- matrix(c(ind+1,state[ind+1]),ncol=2)
# Propose the multiple tries
  Y <- array(0,dim=c(K,m,d))
  X <- array(0,dim=c(K,m,d-1))
  if(length(mat_ind) < 2){
    Y[,mat_ind,] <- t(mvrnormArma(d,A[,mat_ind],E))
  }
  else{
    Y[,mat_ind,] <- replicate(d,apply(A[,mat_ind],2,
       function(x) mvrnormArma(1,x,E)))
  }
  yprob <- apply(Y,3,f)
  if(sum(yprob) <= 0 || sum(is.na(yprob)) > 0){
```

```r
    print("A sampling failure")
    result <- list(alpha=A,prob=NA)
    return(result)
    break
  }
  k <- sample(1:d,size=1,prob=yprob)
  y <- Y[,,k]
# Update the previous parameters based upon the proposed parameters
  if(length(mat_ind) < 2){
    X[,mat_ind,] <- t(mvrnormArma(d-1,y[,mat_ind],E))
  }
  else{
    X[,mat_ind,] <- replicate(d-1,apply(y[,mat_ind],2,function(x) mvrnormArma(1,x,E)))
  }
  x <- A
# Calculate the acceptance ratio
  P <- sum(apply(Y,3,f))/(sum(apply(X,3,f))+f(x))
  acc_prob <- min(1,P)
  if(is.nan(acc_prob)){
    A_prop <- y
    acc_prob <- 1
  }
  else{
    if(runif(1) <= acc_prob){
      A_prop <- y
    }
    else{
      A_prop <- x
    }
  }
  result <- list(alpha=A_prop,prob=acc_prob)
  return(result)
}
```

```
paramRcpp <- function(N,m,obs,z,burnin=0,thinning=1,iter0=N
    ,iter0Alpha=0,thinningAlpha=N,adaptive=FALSE,tau0=50,L
    =100,M=3/40,numDR=10,numMTM=10,alternative=c("metropolis
    ","delayed","multiple","HMC"),TRUNC=TRUE){
#------------------------------------------------------------
# This function calculates and returns the parameters of
    interest by executing several other sub-algorithms.
#------------------------------------------------------------
# Inputs
#------------------------------------------------------------
# - N: the number of MCMC iterations
# - m: the number of hidden states
# - obs: the sequence of observations
# - z: the exogenous variables
# - alternative: the string which defines the type of an MH
      algorithm
# - TRUNC: the logical indicator that determines the method
      of updating the parameter, mu
#------------------------------------------------------------
# Outputs
#------------------------------------------------------------
# - mu: the MCMC samples of mu
# - sigma: the MCMC samples of sigma
# - alpha: the MCMC samples of alpha
# - prob: the estimated probabilities of the hidden states
# - accept: the acceptance rate of the Metropolis
   algorithms
# - CM: the adaptive covariance matrices
#-----------------------------------------------------------
# Set the indices and initialise the parameters
  n <- length(obs)
  Tau <- (N - burnin)/thinning
  mu <- sigma <- matrix(0,nrow=Tau,ncol=m)
  St <- vector("list",Tau)
  R <- diff(range(obs))
  rho <- rep(1/m,m)
  ind <- rep(1:m,each=m)
  indices <- matrix(c(rep(1:m,each=m),rep(1:m,m)),ncol=2)
# Hyperparameters for the univariate normal distribution of
   mu
  Kap <- 1/R^(2)
  xi <- (max(obs)+min(obs))/2
# Hyperparameters for the gamma distribution of sigma
  A_sigma <- 1/2
  B_sigma <- 1/2
# Initialise the transition matrix/matrices and
   hyperparameters for the multivariate Gaussian
   distribution of alpha
  alternative <- match.arg(alternative)
  MHalg <- switch(alternative,metropolis=1,delayed=2,
     multiple=3,HMC=4)
  if(missing(z)){
    Q <- array(0,dim=c(m,m,Tau))
  }
  else{
    K <- ncol(z)
    Q <- Q0 <- array(0,dim=c(m,m,n))
    A <- array(0,dim=c(K,m*m,Tau))
    if(adaptive == TRUE){
      Ainit <- array(0,dim=c(K,m*m,tau0))
    }
    acc_prob <- numNA <- numeric(m)
    mu_A <- numeric(K)
    Sigma_A <- solve(diag(10,K))
    E <- sapply(1:m,function(x) list(diag(13,K)))
    sK <- 2.4^(2)/K
    eps <- 1e-6
  }
# Other parameters
  N0 <- N - iter0
  MAV <- SAV <- matrix(0,nrow=(iter0-iter0Alpha)/
     thinningAlpha,ncol=m)
  if(missing(z)){
    QAV <- array(0,dim=c(m,m,(iter0-iter0Alpha)/
       thinningAlpha))
  }
  else{
    AAV <- array(0,dim=c(K,m*m,(iter0-iter0Alpha)/
       thinningAlpha))
  }
# Execute the MCMC iteration (Metropolis-within-Gibbs)
  for(iter in 1:N){
    message(paste(iter,"iteration(s)"))
    if(iter < 2){
# Initialise the parameter, mu
```

```r
      mu_init <- mu_curr <- min(obs)+R/(2*m)+(1:m-1)*R/m
# Initialise a sequence of the hidden states
      state <- sapply(1:n,function(x) which.min((obs[x]-mu_curr)^(2)))
      tbl <- table(state[-n],state[-1])
# Initialise the parameter, sigma
      sigma_init <- sigma_curr <- sapply(1:m,function(x) sqrt(sum((obs[which(state == x)] - mu_curr[x])^(2))/length(which(state == x))))
# Initialise the parameters, alpha and Q
      if(missing(z)){
        Q_curr <- tbl/rowSums(tbl)
        state_prob <- ffbsCmat(obs,rho,Q_curr,mu_curr,sigma_curr)$prob
      }
      else{
        tbl <- tbl/diag(tbl)
        A_curr <- rbind(log(as.vector(t(tbl))),t(mvrnormArma(m*m,mu_A[-1],solve(Sigma_A[-1,-1]))))
        A_curr[,which(indices[,1]==indices[,2])] <- 0
        Mat <- z[-1,]%*%A_curr
        Q[,,-1] <- aperm(array(t(Mat),dim=c(m,m,n-1)),c(2,1,3))
        Q[,,-1] <- exp(Q[,,-1])/aperm(array(rep(apply(exp(Q[,,-1]),c(1,3),sum),each=m),dim=c(m,m,n-1)),c(2,1,3))
        state_prob <- ffbsCarr(obs,rho,Q,mu_curr,sigma_curr)$prob
      }
      mu_prev <- mu_curr
      sigma_prev <- sigma_curr
      if(missing(z)){
        Q_prev <- Q_curr
      }
      else{
        if(adaptive == TRUE){
          Ainit[,,1] <- A_prev <- A_curr
        }
        else{
          A_prev <- A_curr
        }
      }
    }
# Update a sequence of the hidden states
    else{
      if(iter <= iter0){
        if(missing(z)){
          FB <- ffbsCmat(obs,rho,Q_prev,mu_prev,sigma_prev)
          FB$state <- FB$state + 1
        }
        else{
          FB <- ffbsCarr(obs,rho,Q,mu_prev,sigma_prev)
          FB$state <- FB$state + 1
        }
      }
      else{
        if(missing(z)){
          FB <- ffbsCmat(obs,rho,(Q_prev*(N0-(iter-iter0))+
             QAV*(iter-iter0))/N0,(mu_prev*(N0-(iter-iter0)
             )+MAV*(iter-iter0))/N0,(sigma_prev*(N0-(iter-
             iter0))+SAV*(iter-iter0))/N0)
          FB$state <- FB$state + 1
        }
        else{
          FB <- ffbsCarr(obs,rho,(Q*(N0-(iter-iter0))+QAV*(
             iter-iter0))/N0,(mu_prev*(N0-(iter-iter0))+MAV
             *(iter-iter0))/N0,(sigma_prev*(N0-(iter-iter0)
             )+SAV*(iter-iter0))/N0)
          FB$state <- FB$state + 1
        }
      }
      state_prob <- state_prob + (FB$prob - state_prob)/
         iter
      state <- FB$state
      tbl <- table(state[-n],state[-1])
      while((nrow(tbl) != m) || (ncol(tbl) != m)){
        state <- sapply(1:n,function(x) sample(1:m,size=1,
           prob=state_prob[x,]))
        tbl <- table(state[-n],state[-1])
      }
      nu <- sapply(1:m,function(x) sum(state == x))
# Update the parameters, alpha
      if(missing(z)){
        Q_curr <- t(sapply(1:m,function(x) rdirichlet(1,tbl
           [x,]+1)))
      }
      else{
        if(iter == tau0 && adaptive == TRUE){
          Xbarr <- apply(Ainit[,,1:(iter-2)],1:2,mean)
          Xbar <- apply(Ainit[,,1:(iter-1)],1:2,mean)
        }
        if(iter > tau0 && adaptive == TRUE){
          Xbarr <- Xbarr + (A_prevv-Xbarr)/(iter-2)
          Xbar <- Xbar + (A_prev-Xbar)/(iter-1)
          XX <- lapply(1:m,function(j) Xbar[,which(ind==j)]
             %*%t(A_prev[,which(ind==j)]))
          E <- lapply(1:m,function(j) (iter-3)/(iter-2)*E[[
             j]]+sK/(iter-2)*(1/(iter-1)*((iter-2)*Xbarr[,
             which(ind==j)]%*%t(Xbarr[,which(ind==j)])-(
             iter-1)*(XX[[j]]+t(XX[[j]]))+iter*A_prev[,
             which(ind==j)]%*%t(A_prev[,which(ind==j)]))+
             diag(eps,K)))
        }
        rem <- (iter - numNA)/L - floor((iter - numNA)/L)
           == 0
        if(sum(rem) > 0){
          for(j in which(rem)){
            if(acc_prob[j] < .25){
              E[[j]] <- E[[j]]*0.95^(2)
            }
            if(acc_prob[j] > .5){
              E[[j]] <- E[[j]]*1.05^(2)
            }
          }
        }
        if(MHalg == 1){
          MH <- lapply(1:m,function(j) sampleAlpha(j,state,
             Q,z,A_prev[,which(ind==j)],mu_A,Sigma_A,E[[j
             ]]))
        }
        if(MHalg == 2){
          MH <- lapply(1:m,function(j) sampleAlphaDR(j,
             state,Q,z,A_prev[,which(ind==j)],mu_A,Sigma_A,
             E[[j]],numDR))
        }
        if(MHalg == 3){
          MH <- lapply(1:m,function(j) sampleAlphaMTM(j,
             state,Q,z,A_prev[,which(ind==j)],mu_A,Sigma_A,
             E[[j]],numMTM))
        }
        A_curr <- matrix(sapply(1:m,function(j) MH[[j]]$
           alpha),K,m*m)
# Update the transition matrices, Q
        Mat <- z[-1,]%*%A_curr
        Q[,,-1] <- aperm(array(t(Mat),dim=c(m,m,n-1)),c
           (2,1,3))
        Q[,,-1] <- exp(Q[,,-1])/aperm(array(rep(apply(exp(Q
           [,,-1]),c(1,3),sum),each=m),dim=c(m,m,n-1)),c
           (2,1,3))
        numNA <- numNA + sapply(1:m,function(j) as.numeric(
           is.na(MH[[j]]$prob)))
        acc_prob <- sapply(1:m,function(j) if(is.na(MH[[j]]
           $prob)){acc_prob[j]} else{acc_prob[j] + (MH[[j]]
           $prob - acc_prob[j])/(iter-(numNA[j] + 1))})
      }
# Update the parameter, sigma
      U <- sapply(1:m,function(x) sum((obs[which(state==x)]
          - mu_prev[x])^(2)))
      V <- sapply(1:m,function(x) sum(obs[which(state==x)])
          )
      sigma_curr <- rgamma(m,nu/2+A_sigma,rate=U/2+B_sigma)
      sigma_curr <- sqrt(1/sigma_curr)
# Update the parameter, mu
      if(TRUNC == TRUE){
        mu_curr[1] <- rnorm(1,mean=(sigma_init[1]^(2)*V[1]+
           sigma_curr[1]^(2)*mu_init[1])/(sigma_init[1]^(2)
           *nu[1]+sigma_curr[1]^(2)),sd=sqrt(sigma_curr
           [1]^(2)*sigma_init[1]^(2)/(sigma_init[1]^(2)*nu
           [1]+sigma_curr[1]^(2))))
        for(j in 2:m){
          har <- (sigma_curr[j]^(2)+sigma_curr[j-1]^(2))/(2
             *M^(2)*sigma_curr[j-1]^(2)*sigma_curr[j]^(2))
          mu_curr[j] <- rtruncnorm(1,mu_curr[j-1],Inf,(har*
             V[j]+sigma_curr[j]^(2)*mu_curr[j-1])/(har*nu[j
             ]+sigma_curr[j]^(2)),sqrt((sigma_curr[j]^(2)*
             har)/(har*nu[j]+sigma_curr[j]^(2))))
        }
      }
      else{
        mu_curr <- rnorm(m,mean=(V+Kap*xi*sigma_curr^(2))/(
           nu+Kap*sigma_curr^(2)),sd=sqrt(sigma_curr^(2)/(
           nu+Kap*sigma_curr^(2))))
      }
# Update the parameters of interest at the previous
   iteration
      mu_prev <- mu_curr
      sigma_prev <- sigma_curr
      if(missing(z)){
        Q_prev <- Q_curr
```

```r
      }
      else{
        if(iter <= tau0 && adaptive == TRUE){
          Ainit[,,iter] <- A_curr
        }
        A_prevv <- A_prev
        A_prev <- A_curr
      }
# Imputation of the means of the parameters
      if(iter < iter0 && iter >= iter0Alpha && (iter-
         iter0Alpha)/thinningAlpha - floor((iter-iter0Alpha
         )/thinningAlpha) == 0){
        MAV[(iter-iter0Alpha)/thinningAlpha,] <- mu_curr
        SAV[(iter-iter0Alpha)/thinningAlpha,] <- sigma_curr
        if(missing(z)){
          QAV[,,(iter-iter0Alpha)/thinningAlpha] <- Q_curr
        }
        else{
          AAV[,,(iter-iter0Alpha)/thinningAlpha] <- A_curr
        }
      }
# Initialise the means of the parameters
      if(iter == iter0){
        if(missing(z)){
          QAV <- apply(QAV,1:2,median)
        }
        else{
          AAV <- z[-1,]%*%apply(AAV,1:2,mean)
          Q0[,,-1] <- aperm(array(t(AAV),dim=c(m,m,n-1)),c
             (2,1,3))
          Q0[,,-1] <- exp(Q0[,,-1])/aperm(array(rep(apply(
            exp(Q0[,,-1]),c(1,3),sum),each=m),dim=c(m,m,n
            -1)),c(2,1,3))
          QAV <- Q0
        }
        MAV <- apply(MAV,2,mean)
        SAV <- apply(SAV,2,mean)
      }
# Update the medians by taking arithmetic means
      if(iter > iter0){
        if(missing(z)){
          QAV <- QAV + (Q_curr - QAV)/iter
        }
        else{
          QAV <- QAV + (Q - QAV)/iter
        }
        MAV <- MAV + (mu_curr - MAV)/iter
        SAV <- SAV + (sigma_curr - SAV)/iter
      }
    }
    if(iter >= burnin + thinning && (iter-burnin)/thinning
      - floor((iter-burnin)/thinning) == 0){
      mu[(iter - burnin)/thinning,] <- mu_curr
      sigma[(iter - burnin)/thinning,] <- sigma_curr
      if(missing(z)){
        Q[,,(iter - burnin)/thinning] <- Q_curr
      }
      else{
        A[,,(iter - burnin)/thinning] <- A_curr
      }
      St[[(iter-burnin)/thinning]] <- state
      message(paste(iter,"th iteration is stored.",sep=""))
    }
  }
  if(missing(z)){
    result <- list(mu=mu,sigma=sigma,Q=Q,state=St,prob=
        state_prob)
  }
  else{
    result <- list(mu=mu,sigma=sigma,alpha=A,state=St,prob=
        state_prob,accept=acc_prob,CM=E)
  }
  return(result)
}
```

```
label_switch <- function(mu,sigma,St,A,Q){
#-----------------------------------------------------------
# This function relabels the hidden states after the MCMC
    algorithm.
#-----------------------------------------------------------
# Inputs
#-----------------------------------------------------------
# - mu: the MCMC samples of mu
# - sigma: the MCMC samples of sigma
# - A: the MCMC samples of alpha
# - St: a sequence of the hidden states
# - Q: the MCMC samples of a transition matrix
#-----------------------------------------------------------
# Outputs
#-----------------------------------------------------------
# - mu: the relabelled MCMC samples of mu
# - sigma: the relabelled MCMC samples of sigma
# - alpha: the relabelled MCMC samples of alpha
# - state: the relabelled MCMC samples of state
# - Q: the relabelled MCMC samples of Q
#-----------------------------------------------------------
# Set the indices and initialise the parameters
  N <- dim(mu)[1]
  m <- dim(mu)[2]
  perm <- permn(1:m)
  M <- gamma(m+1)
  P <- matrix(1:(m*m),nrow=m,ncol=m)
# Relabel the MCMC samples of interest
  for(iter in 1:N){
    L <- sapply(1:M,function(i) mu[iter,perm[[i]]]==sort(mu
       [iter,]))
    perm_curr <- perm[[which.max(colSums(L))]]
    mu[iter,] <- mu[iter,perm_curr]
    sigma[iter,] <- sigma[iter,perm_curr]
    St[[iter]] <- mapvalues(St[[iter]],1:m,perm_curr)
    if(missing(Q)){
      A[,,iter] <- A[,as.vector(P[perm_curr,perm_curr]),
         iter]
    }
    else{
      Q[,,iter] <- Q[perm_curr,perm_curr,iter]
    }
  }
  if(missing(Q)){
    result <- list(mu=mu,sigma=sigma,alpha=A,state=St)
  }
  else{
    result <- list(mu=mu,sigma=sigma,Q=Q,state=St)
  }
  return(result)
}
```

```
marLik <- function(mu,sigma,state,obs,z,alpha=NULL,Q=NULL){
#------------------------------------------------------------
# This function calculates and returns a log marginal
   likelihood for the MCMC samples of interest.
#------------------------------------------------------------
# Inputs
#------------------------------------------------------------
# - mu: the MCMC samples of mu
# - sigma: the MCMC samples of sigma
# - state: the MCMC samples of hidden states
# - obs: the observations
# - z: the exogenous variables
# - alpha: the MCMC samples of alpha
# - Q: the MCMC samples of the transition matrix/matrices
#------------------------------------------------------------
# Outputs
#------------------------------------------------------------
# - fhat: the log likelihoods
# - phat: the log priors
# - Phat: the log posteriors
#------------------------------------------------------------
  m <- ncol(mu)
  n <- length(obs)
  N <- length(state)
  M <- 3/40
```

```r
  rho <- rep(1/m,m)
  if(!missing(z)){
    K <- ncol(z)
    mu_A <- numeric(K)
    Sigma_A <- solve(diag(10,K))
    CM <- diag(13,K)
  }
  R <- diff(range(obs))
  mu_init <- min(obs)+R/(2*m)+(1:m-1)*R/m
  St <- sapply(1:n,function(x) which.min((obs[x]-mu_init)^(2)))
  sigma_init <- sapply(1:m,function(x) sqrt(sum((obs[which(St==x)] - mu_init[x])^(2))/length(which(St==x))))
  A_sigma <- 1/2
  B_sigma <- 1/2
  muStar <- apply(mu,2,mean)
  sigmaStar <- apply(sigma,2,mean)
  lambdaStar <- apply(sigma^(-2),2,median)
  if(missing(z)){
    QStar <- apply(Q,1:2,mean)
  }
  else{
    alphaStar <- apply(alpha,1:2,mean)
  }
# Define the posterior density function
  f <- function(A){
    prod((exp(z%*%A)/rowSums(exp(z%*%A)))[mat_dims])*exp(-1/2*sum(diag((A-mu_A)%*%t(A-mu_A)%*%Sigma_A)))
  }
# Compute the log marginal likelihood
```

```
  fhat <- sapply(1:m,function(x) dnorm(obs,muStar[x],
     sigmaStar[x]))
  fhat[1,] <- fhat[1,]*rho
  for(i in 2:n){
    if(missing(z)){
      fhat[i,] <- (fhat[i-1,]/sum(fhat[i-1,]))%*%QStar*fhat
         [i,]
    }
    else{
      P <- lapply(1:dim(alpha)[3],function(x) matrix(exp(z[
         i,]%*%alpha[,,x]),m,m,TRUE))
      P <- lapply(1:length(P),function(x) P[[x]]/rowSums(P
         [[x]]))
      P <- array(unlist(P),dim=c(m,m,length(P)))
      fhat[i,] <- (fhat[i-1,]/sum(fhat[i-1,]))%*%apply(P
         ,1:2,median)*fhat[i,]
    }
  }
  fhat <- sum(log(apply(fhat,1,sum)))
# Compute the log priors
  phat <- sum(log(dnorm(muStar,0,sqrt(0.3^(-1))))) + sum(
     log(dgamma(lambdaStar,A_sigma,B_sigma)))
  if(missing(z)){
    phat <- phat + sum(sapply(1:m,function(x) log(
       ddirichlet(QStar[x,],rep(1,m)))))
  }
  else{
    phat <- phat + sum(sapply(which(diag(m)==0),function(x)
        log(dmvnorm(alphaStar[,x],mu_A,solve(Sigma_A)))))
  }
```

```
# Compute the log posteriors
  muPool <- alphaPool1 <- alphaPool2 <- numeric(m)
  estMu <- estLambda <- estQ <- estAlpha1 <- estAlpha2 <-
     numeric(N)
  for(iter in 1:N){
    nu <- sapply(1:m,function(x) sum(state[[iter]]==x))
    U <- sapply(1:m,function(x) sum((obs[which(state[[iter
       ]]==x)] - mu[iter,x])^(2)))
    V <- sapply(1:m,function(x) sum(obs[which(state[[iter
       ]]==x)]))
    estLambda[iter] <- prod(dgamma(lambdaStar,nu/2+A_sigma,
       U/2+B_sigma))
    muPool[1] <- dnorm(muStar[1],mean=(sigma_init[1]^(2)*V
       [1]+sigmaStar[1]^(2)*mu_init[1])/(sigma_init[1]^(2)*
       nu[1]+sigmaStar[1]^(2)),sd=sqrt(sigmaStar[1]^(2)*
       sigma_init[1]^(2)/(sigma_init[1]^(2)*nu[1]+sigmaStar
       [1]^(2))))
    for(j in 2:m){
      har <- (sigmaStar[j]^(2)+sigmaStar[j-1]^(2))/(2*M^(2)
         *sigmaStar[j-1]^(2)*sigmaStar[j]^(2))
      muPool[j] <- dtruncnorm(muStar[j],mu[iter,j-1],Inf,(
         har*V[j]+sigmaStar[j]^(2)*mu[iter,j-1])/(har*nu[j
         ]+sigmaStar[j]^(2)),sqrt((sigmaStar[j]^(2)*har)/(
         har*nu[j]+sigmaStar[j]^(2))))
    }
    estMu[iter] <- prod(muPool)
    if(missing(z)){
      tbl <- table(state[[iter]][-n],state[[iter]][-1])
      estQ[iter] <- prod(sapply(1:m,function(x) ddirichlet(
         QStar[x,],tbl[x,]+1)))
```

```r
    }
    else{
      for(k in 1:m){
        idx <- rep(1:m,each=m)
        ind <- which(state[[iter]][-n]==k)
        mat_ind <- (1:m)[-k]
        mat_dims <- matrix(c(ind+1,state[[iter]][ind+1]),ncol=2)
        alphaPool1[k] <- min(1,f(alphaStar[,which(idx==k)])/f(alpha[,which(idx==k),iter]))*prod(sapply(which(idx==k),function(x) dmvnorm(alphaStar[,x],alpha[,x,iter],CM))[mat_ind])
        alphaPool2[k] <- min(1,f(alpha[,which(idx==k),iter])/f(alphaStar[,which(idx==k)]))
      }
      estAlpha1[iter] <- prod(alphaPool1)
      estAlpha2[iter] <- prod(alphaPool2)
    }
  }
  if(missing(z)){
    Phat <- log(mean(estQ)) + log(mean(estLambda)) + log(mean(estMu))
  }
  else{
    Phat <- log(mean(estAlpha1)) - log(mean(estAlpha2)) + log(mean(estLambda)) + log(mean(estMu))
  }
  return(c(fhat,phat,-Phat))
}
```

E.2 C++

```
#include <RcppArmadillo.h>
// [[Rcpp::depends(RcppArmadillo)]]
using namespace Rcpp;
// [[Rcpp::export]]
arma::mat mvrnormArma(int n, arma::vec mu, arma::mat Sigma)
   {
  int ncols = Sigma.n_cols;
  arma::mat Y = arma::randn(n,ncols);
  return arma::repmat(mu,1,n).t() + Y * arma::chol(Sigma);
}
```

```
#include <RcppArmadilloExtensions/sample.h>
// [[Rcpp::depends(RcppArmadillo)]]
using namespace Rcpp;
// [[Rcpp::export]]
arma::rowvec dnormvec(double x, NumericVector means,
   NumericVector sds){
  int n = means.size();
  arma::rowvec res(n);
  for(int i = 0; i < n; i++){
    res[i] = R::dnorm(x,means[i],sds[i],false);
  }
  return res;
}
// [[Rcpp::export]]
List ffbsCmat(NumericVector obs, arma::rowvec rho, arma::
   mat Q, NumericVector mu, NumericVector sigma){
    int m = rho.n_elem;
    int n = obs.size();
  IntegerVector state(n);
 IntegerVector space = seq_len(m) - 1;
  arma::mat fwd(n,m);
  arma::mat bwd(n,m);
// Initialisation of the forward variable
  fwd.row(0) = rho%dnormvec(obs(0),mu,sigma);
  fwd.row(0) = fwd.row(0)/sum(fwd.row(0));
// Execute the forward filtering recursion
  for(int t = 1; t < n; t++){
    fwd.row(t) = (fwd.row(t-1)*Q)%dnormvec(obs(t),mu,
     sigma);
    fwd.row(t) = fwd.row(t)/sum(fwd.row(t));
  }
// Initialisation of the backward variable
  bwd.row(n-1) = fwd.row(n-1);
  state(n-1) = bwd.row(n-1).index_max();
// Execute the backward sampling recursion
  for(int t = n-1; t > 0; t--){
    bwd.row(t-1) = fwd.row(t-1)%arma::conv_to<arma::
     rowvec>::from(Q.col(state(t)));
    bwd.row(t-1) = bwd.row(t-1)/sum(bwd.row(t-1));
    state(t-1) = RcppArmadillo::sample(space,1,false,as
     <NumericVector>(wrap(arma::conv_to<arma::rowvec
     >::from(bwd.row(t-1)))))(0);
  }
  return List::create(Named("state",state),Named("prob",
   bwd));
}
```

```
#include <RcppArmadilloExtensions/sample.h>
// [[Rcpp::depends(RcppArmadillo)]]
```

```
using namespace Rcpp;
// [[Rcpp::export]]
arma::rowvec dnormvec(double x, NumericVector means,
    NumericVector sds){
  int n = means.size();
  arma::rowvec res(n);
  for(int i = 0; i < n; i++){
    res[i] = R::dnorm(x,means[i],sds[i],false);
  }
  return res;
}
// [[Rcpp::export]]
List ffbsCarr(NumericVector obs, arma::rowvec rho, arma::
    cube Q, NumericVector mu, NumericVector sigma){
    int m = rho.n_elem;
    int n = obs.size();
    IntegerVector state(n);
  IntegerVector space = seq_len(m) - 1;
    arma::mat fwd(n,m);
    arma::mat bwd(n,m);
// Initialisation of the forward variable
    fwd.row(0) = rho%dnormvec(obs(0),mu,sigma);
    fwd.row(0) = fwd.row(0)/sum(fwd.row(0));
// Execute the forward filtering recursion
    for(int t = 1; t < n; t++){
        fwd.row(t) = (fwd.row(t-1)*Q.slice(t))%dnormvec(obs
            (t),mu,sigma);
        fwd.row(t) = fwd.row(t)/sum(fwd.row(t));
    }
// Initialisation of the backward variable
```

```
    bwd.row(n-1) = fwd.row(n-1);
    state(n-1) = bwd.row(n-1).index_max();
// Execute the backward sampling recursion
    for(int t = n-1; t > 0; t--){
        bwd.row(t-1) = fwd.row(t-1)%arma::conv_to<arma::
           rowvec>::from(Q.slice(t).col(state(t)));
        bwd.row(t-1) = bwd.row(t-1)/sum(bwd.row(t-1));
        state(t-1) = RcppArmadillo::sample(space,1,false,as
           <NumericVector>(wrap(arma::conv_to<arma::rowvec
           >::from(bwd.row(t-1)))))(0);
    }
    return List::create(Named("state",state),Named("prob",
       bwd));
}
```

Appendix F

Source Code: Multivariate NHGHMM

F.1 R©

```
simHMM <- function(n,mu,Sigma,Q){
#------------------------------------------------------------
# This function simulates a non-homogeneous Gaussian hidden
    Markov model and returns a sequence of the hidden
   states and the observations.
#------------------------------------------------------------
# Inputs
#------------------------------------------------------------
# - n: the number of observations
# - mu: a matrix of m row vectors of mu
# - Sigma: matrices of m positive-definite matrices of
   Sigma
# - Q: the transition matrix/matrices
#------------------------------------------------------------
# Outputs
#------------------------------------------------------------
# - state: a sequence of the hidden states
```

```r
# - obs: a sequence of the observations
#-----------------------------------------------------------
  if(dim(Q)[1] != dim(Q)[2]){
    stop("The transition matrix is not square.")
  }
  d <- ncol(mu)
  m <- dim(Q)[1]
  state <- numeric(n)
  obs <- matrix(0,nrow=n,ncol=d)
# Initialise the first elements of states and observations
  state[1] <- sample(1:m,size=1,prob=rep(1/m,m))
  obs[1,] <- mvrnorm(1,mu[state[1],],Sigma[,,state[1]])
# Homogeneous transition probabilities
  if(is.matrix(Q)){
    for(t in 2:n){
      state[t] <- sample(1:m,size=1,prob=Q[state[t-1],])
      obs[t,] <- mvrnorm(1,mu[state[t],],Sigma[,,state[t]])
    }
  }
# Non-homogeneous transition probabilities
  else{
    for(t in 2:n){
      state[t] <- sample(1:m,size=1,prob=Q[state[t-1],,t])
      obs[t,] <- mvrnorm(1,mu[state[t],],Sigma[,,state[t]])
    }
  }
  result <- list(state=state,obs=obs)
  return(result)
}
```

```r
sampleAlpha <- function(j,state,Q,z,A,mu_A,Sigma_A,E){
```

```
#-----------------------------------------------------------
# This function samples the parameter, alpha, from the
   posterior distribution of interest, f.
#-----------------------------------------------------------
# Inputs
#-----------------------------------------------------------
# - j: the index from 1 to m
# - state: the estimated sequence of the hidden states at
   the current MCMC iteration
# - Q: the current estimate of the transition probabilities
# - z: the exogenous variables
# - A: the parameter, alpha, at the current MCMC iteration
# - mu_A: the vector of means, mu, for the multivariate
   Gaussian distribution
# - Sigma_A: the covariance matrix, Sigma, for the
   multivariate Gaussian distribution
# - E: the covariance matrix for random walk Metropolis
   updates
#-----------------------------------------------------------
# Outputs
#-----------------------------------------------------------
# - alpha: the new proposed parameter, alpha
# - prob: the acceptance probability of the proposed alpha
#-----------------------------------------------------------
# Define the posterior density function
  f <- function(A){
    prod((exp(z%*%A)/rowSums(exp(z%*%A)))[mat_dims])*exp(-1
       /2*sum(diag((A-mu_A)%*%t(A-mu_A)%*%Sigma_A)))
  }
  m <- dim(Q)[1]
  n <- length(state)
  ind <- which(state[-n]==j)
  mat_ind <- (1:m)[-j]
  mat_dims <- matrix(c(ind+1,state[ind+1]),ncol=2)
# Metropolis-Hastings update
  A_curr <- A
  if(length(mat_ind) < 2){
    A[,mat_ind] <- mvrnormArma(1,A_curr[,mat_ind],E)
  }
  else{
    A[,mat_ind] <- apply(A_curr[,mat_ind],2,function(x)
       mvrnormArma(1,x,E))
  }
# Calculate the acceptance ratio
  P <- f(A)/f(A_curr)
  acc_prob <- min(1,P)
  if(is.nan(acc_prob)){
    A_prop <- A
    acc_prob <- 1
  }
  else{
    if(runif(1) <= acc_prob){
      A_prop <- A
    }
    else{
      A_prop <- A_curr
    }
  }
  result <- list(alpha=A_prop,prob=acc_prob)
  return(result)
```

```
}
```

```
sampleAlphaDR <- function(j,state,Q,z,A,mu_A,Sigma_A,E,STG)
   {
#-----------------------------------------------------------
# This function samples the parameter, alpha, from the
   posterior distribution of interest, f.
#-----------------------------------------------------------
# Inputs
#-----------------------------------------------------------
# - j: the index from 1 to m
# - state: the estimated sequence of the hidden states at
   the current MCMC iteration
# - Q: the current estimate of the transition probabilities
# - z: the exogenous variables
# - A: the parameter, alpha, at the current MCMC iteration
# - mu_A: the vector of means, mu, for the multivariate
   Gaussian distribution
# - Sigma_A: the covariance matrix, Sigma, for the
   multivariate Gaussian distribution
# - E: the covariance matrix for random walk Metropolis
   updates
# - STG: the number of delayed rejection stages
#-----------------------------------------------------------
# Outputs
#-----------------------------------------------------------
# - alpha: the new proposed parameter, alpha
# - prob: the acceptance probability of the proposed alpha
#-----------------------------------------------------------
# Define the posterior density function
  f <- function(A){
    prod((exp(z%*%A)/rowSums(exp(z%*%A)))[mat_dims])*exp(-1
       /2*sum(diag((A-mu_A)%*%t(A-mu_A)%*%Sigma_A)))
  }
  m <- dim(Q)[1]
  n <- length(state)
  K <- dim(A)[1]
  ind <- which(state[-n]==j)
  mat_ind <- (1:m)[-j]
  mat_dims <- matrix(c(ind+1,state[ind+1]),ncol=2)
  X <- A
  Y <- array(0,dim=c(K,m,STG))
# Metropolis-Hastings update (a single column)
  if(length(mat_ind) < 2){
    Y[,mat_ind,1] <- mvrnormArma(1,X[,mat_ind],E)
    Y_star <- Y[,,1]
    P <- f(Y[,,1])/f(X)
    acc_prob <- min(1,P)
    if(is.nan(acc_prob)){
      A_prop <- Y[,,1]
      acc_prob <- 1
    }
    else{
      if(f(Y[,,1]) >= f(X) || runif(1) <= acc_prob){
        A_prop <- Y[,,1]
      }
      else{
        for(stg in 2:STG){
          Y[,mat_ind,stg] <- mvrnormArma(1,Y[,mat_ind,stg
             -1],E)
          P <- (f(Y[,,stg]) - f(Y_star))/(f(X) - f(Y_star))
```

```r
          if(f(Y[,,stg]) > f(Y_star)){
            Y_star <- Y[,,stg]
          }
          acc_prob <- min(1,P)
          if(is.nan(acc_prob)){
            A_prop <- Y[,,stg]
            acc_prob <- 1
            break
          }
          else{
            if(f(Y[,,stg]) >= f(X) || runif(1) <= acc_prob) {
              A_prop <- Y[,,stg]
              break
            }
            if(stg == STG){
              A_prop <- X
              break
            }
            next
          }
        }
      }
    }
  }
# Metropolis-Hastings update (mulptile columns)
  else{
    Y[,mat_ind,1] <- apply(X[,mat_ind],2,function(x) mvrnormArma(1,x,E))
    Y_star <- Y[,,1]
    P <- f(Y[,,1])/f(X)
    acc_prob <- min(1,P)
    if(is.nan(acc_prob)){
      A_prop <- Y[,,1]
      acc_prob <- 1
    }
    else{
      if(f(Y[,,1]) >= f(X) || runif(1) <= acc_prob){
        A_prop <- Y[,,1]
      }
      else{
        for(stg in 2:STG){
          Y[,mat_ind,stg] <- apply(Y[,mat_ind,stg-1],2,
             function(x) mvrnormArma(1,x,E))
          P <- (f(Y[,,stg]) - f(Y_star))/(f(X) - f(Y_star))
          if(f(Y[,,stg]) > f(Y_star)){
            Y_star <- Y[,,stg]
          }
          acc_prob <- min(1,P)
          if(is.nan(acc_prob)){
            A_prop <- Y[,,stg]
            acc_prob <- 1
          }
          else{
            if(f(Y[,,stg]) >= f(X) || runif(1) <= acc_prob)
              {
              A_prop <- Y[,,stg]
              break
            }
            if(stg == STG){
              A_prop <- X
              break
            }
            next
          }
        }
      }
    }
  }
  result <- list(alpha=A_prop,prob=acc_prob)
  return(result)
}
```

```
sampleAlphaMTM <- function(j,state,Q,z,A,mu_A,Sigma_A,E,d){
#----------------------------------------------------------
# This function samples the parameter, alpha, from the posterior distribution of interest, f.
#----------------------------------------------------------
# Inputs
#----------------------------------------------------------
# - j: the index from 1 to m
# - state: the estimated sequence of the hidden states at the current MCMC iteration
# - Q: the current estimate of the transition probabilities
# - z: the exogeneous variables
# - A: the parameter, alpha, at the current MCMC iteration
# - mu_A: the vector of means, mu, for the multivariate Gaussian distribution
# - Sigma_A: the covariance matrix, Sigma, for the multivariate Gaussian distribution
# - E: the covariance matrix for random walk Metropolis
```

```
   updates
# - d: the number of multiple tries for Metropolis updates
#-----------------------------------------------------------
# Outputs
#-----------------------------------------------------------
# - alpha: the new proposed parameter, alpha
# - prob: the acceptance probability of the proposed alpha
#-----------------------------------------------------------
# Define the posterior density function
  f <- function(A){
    prod((exp(z%*%A)/rowSums(exp(z%*%A)))[mat_dims])*exp(-1
       /2*sum(diag((A-mu_A)%*%t(A-mu_A)%*%Sigma_A)))
  }
  m <- dim(Q)[1]
  n <- length(state)
  K <- ncol(z)
  ind <- which(state[-n]==j)
  mat_ind <- (1:m)[-j]
  mat_dims <- matrix(c(ind+1,state[ind+1]),ncol=2)
# Propose the multiple tries
  Y <- array(0,dim=c(K,m,d))
  X <- array(0,dim=c(K,m,d-1))
  if(length(mat_ind) < 2){
    Y[,mat_ind,] <- t(mvrnormArma(d,A[,mat_ind],E))
  }
  else{
    Y[,mat_ind,] <- replicate(d,apply(A[,mat_ind],2,
       function(x) mvrnormArma(1,x,E)))
  }
  yprob <- apply(Y,3,f)
```

```r
  if(sum(yprob) <= 0 || sum(is.na(yprob)) > 0){
    print("A sampling failure")
    result <- list(alpha=A,prob=NA)
    return(result)
    break
  }
  k <- sample(1:d,size=1,prob=yprob)
  y <- Y[,,k]
# Update the previous parameters based upon the proposed
   parameters
  if(length(mat_ind) < 2){
    X[,mat_ind,] <- t(mvrnormArma(d-1,y[,mat_ind],E))
  }
  else{
    X[,mat_ind,] <- replicate(d-1,apply(y[,mat_ind],2,
        function(x) mvrnormArma(1,x,E)))
  }
  x <- A
# Calculate the acceptance ratio
  P <- sum(apply(Y,3,f))/(sum(apply(X,3,f))+f(x))
  acc_prob <- min(1,P)
  if(is.nan(acc_prob)){
    A_prop <- y
    acc_prob <- 1
  }
  else{
    if(runif(1) <= acc_prob){
      A_prop <- y
    }
    else{
      A_prop <- x
    }
  }
  result <- list(alpha=A_prop,prob=acc_prob)
  return(result)
}
```

```
paramRcppMvn <- function(N,m,obs,z,burnin=0,thinning=1,
   iter0=N,iter0Alpha=0,thinningAlpha=N,adaptive=FALSE,tau0
   =50,L=100,epsHMC=1e-1,numDR=10,numMTM=10,numHMC=25,
   sliceWD=6,alternative=c("metropolis","delayed","multiple
   ","HMC","slice")){
#-----------------------------------------------------------
# This function calculates and returns the parameters of
   interest.
#-----------------------------------------------------------
# Inputs
#-----------------------------------------------------------
# - N: the number of MCMC iterations
# - m: the number of hidden states
# - obs: the sequence of observations
# - z: the exogenous variables
#-----------------------------------------------------------
# Outputs
#-----------------------------------------------------------
# - mu: the MCMC samples of mu
# - Sigma: the MCMC samples of Sigma
# - alpha: the MCMC samples of alpha
# - prob: the estimated probabilities of the hidden states
# - accept: the acceptance rate of the Metropolis
   algorithms
```

```
# - CM: the adaptive covariance matrices
#------------------------------------------------------------
# Set the indices and initialise the parameters
  n <- dim(obs)[1]
  d <- dim(obs)[2]
  Tau <- (N - burnin)/thinning
  mu <- array(0,dim=c(m,d,Tau))
  Sigma <- array(0,dim=c(d,d,m,Tau))
  St <- vector("list",Tau)
  rho <- rep(1/m,m)
  ind <- rep(1:m,each=m)
  indices <- matrix(c(rep(1:m,each=m),rep(1:m,m)),ncol=2)
  signMat <- sign(cov(obs))
# Hyperparameters for the multivariate normal distribution
   of mu
  mu_M <- apply(obs,2,function(x) mean(range(x)))
  Sigma_M <- diag(apply(obs,2,function(x) diff(range(x))))
  idx_M <- which(Sigma_M == 0,arr.ind=TRUE)
  Sigma_M[idx_M] <- signMat[idx_M]*sapply(1:nrow(idx_M),
     function(x) sqrt(Sigma_M[idx_M[x,1],idx_M[x,1]]*Sigma_
     M[idx_M[x,2],idx_M[x,2]])/2)
# Hyperparameters for the inverse-Wishart distribution of
   Sigma
  gamma_S <- trunc((d+1)/2) + 1
  Lambda_S <- diag(gamma_S,d)
  idx_S <- which(Lambda_S == 0,arr.ind=TRUE)
  Lambda_S[idx_S] <- signMat[idx_S]*gamma_S/2
# Initialise the transition matrix/matrices and
   hyperparameters for the multivariate normal distribution
    of alpha
  alternative <- match.arg(alternative)
  MHalg <- switch(alternative,metropolis=1,delayed=2,
     multiple=3,HMC=4,slice=5)
  if(missing(z)){
    Q <- array(0,dim=c(m,m,Tau))
  }
  else{
    K <- ncol(z)
    Q <- Q0 <- array(0,dim=c(m,m,n))
    A <- array(0,dim=c(K,m*m,Tau))
    if(adaptive == TRUE){
      Ainit <- array(0,dim=c(K,m*m,tau0))
    }
    acc_prob <- numNA <- numeric(m)
    mu_A <- numeric(K)
    Sigma_A <- solve(diag(10,K))
    E <- sapply(1:m,function(x) list(diag(13,K)))
    sK <- 2.4^(2)/K
    eps <- 1e-6
  }
# Other parameters
  N0 <- N - iter0
  MAV <- array(0,dim=c(m,d,(iter0-iter0Alpha)/thinningAlpha
     ))
  SAV <- array(0,dim=c(d,d,m,(iter0-iter0Alpha)/
     thinningAlpha))
  if(missing(z)){
    QAV <- array(0,dim=c(m,m,(iter0-iter0Alpha)/
       thinningAlpha))
  }
  else{
    AAV <- array(0,dim=c(K,m*m,(iter0-iter0Alpha)/
        thinningAlpha))
  }
# Execute the MCMC iteration (FB-MH-GS)
  for(iter in 1:N){
    message(paste(iter,"iteration(s)"))
    if(iter < 2){
# Initialise the parameter, mu
      mu_curr <- kmeans(na.omit(obs),m)$centers
      mu_curr <- mu_curr[order(mu_curr[,1]),]
# Initialise a sequence of the hidden states
      state <- sapply(1:n,function(x) as.numeric(which.min(
          apply((matrix(obs[x,],m,d,TRUE) - mu_curr)^(2),1,
          sum))))
      tbl <- table(state[-n],state[-1])
# Initialise the parameter, Sigma
      Sigma_curr <- array(unlist(sapply(1:m,function(x) cov
          (obs[which(state == x),] - matrix(mu_curr[x,],sum(
          state == x),d,TRUE)))),dim=c(d,d,m))
# Initialise the parameters, alpha and Q
      if(missing(z)){
        Q_curr <- tbl/rowSums(tbl)
        state_prob <- ffbsCmatMvn(obs,rho,Q_curr,mu_curr,
            Sigma_curr)$prob
      }
      else{
        tbl <- tbl/diag(tbl)
        A_curr <- rbind(log(as.vector(t(tbl))),t(
            mvrnormArma(m*m,mu_A[-1],solve(Sigma_A[-1,-1])))
          )
        A_curr[,which(indices[,1]==indices[,2])] <- 0
        Mat <- z[-1,]%*%A_curr
        Q[,,-1] <- aperm(array(t(Mat),dim=c(m,m,n-1)),c
          (2,1,3))
        Q[,,-1] <- exp(Q[,,-1])/aperm(array(rep(apply(exp(Q
          [,,-1]),c(1,3),sum),each=m),dim=c(m,m,n-1)),c
          (2,1,3))
        state_prob <- ffbsCarrMvn(obs,rho,Q,mu_curr,Sigma_
          curr)$prob
      }
      mu_prev <- mu_curr
      Sigma_prev <- Sigma_curr
      if(missing(z)){
        Q_prev <- Q_curr
      }
      else{
        if(adaptive == TRUE){
          Ainit[,,1] <- A_prev <- A_curr
        }
        else{
          A_prev <- A_curr
        }
      }
    }
# Update a sequence of the hidden states
    else{
      if(iter <= iter0){
        if(missing(z)){
          FB <- ffbsCmatMvn(obs,rho,Q_prev,mu_prev,Sigma_
             prev)
          FB$state <- FB$state + 1
        }
        else{
          FB <- ffbsCarrMvn(obs,rho,Q,mu_prev,Sigma_prev)
          FB$state <- FB$state + 1
        }
      }
      else{
        if(missing(z)){
          FB <- ffbsCmatMvn(obs,rho,(Q_prev*(N0-(iter-iter0
             ))+QAV*(iter-iter0))/N0,(mu_prev*(N0-(iter-
             iter0))+MAV*(iter-iter0))/N0,(Sigma_prev*(N0-(
             iter-iter0))+SAV*(iter-iter0))/N0)
          FB$state <- FB$state + 1
        }
        else{
          FB <- ffbsCarrMvn(obs,rho,(Q*(N0-(iter-iter0))+
             QAV*(iter-iter0))/N0,(mu_prev*(N0-(iter-iter0)
             )+MAV*(iter-iter0))/N0,(Sigma_prev*(N0-(iter-
             iter0))+SAV*(iter-iter0))/N0)
          FB$state <- FB$state + 1
        }
      }
      state_prob <- state_prob + (FB$prob - state_prob)/
         iter
      state <- FB$state
      tbl <- table(state[-n],state[-1])
      nu <- sapply(1:m,function(x) sum(state == x))
      while((nrow(tbl) != m) || (ncol(tbl) != m) || sum(nu
          + gamma_S < d) > 0){
        state <- sapply(1:n,function(x) sample(1:m,size=1,
          prob=state_prob[x,]))
        tbl <- table(state[-n],state[-1])
        nu <- sapply(1:m,function(x) sum(state == x))
      }
# Update the parameters, alpha
      if(missing(z)){
        Q_curr <- t(sapply(1:m,function(x) rdirichlet(1,tbl
          [x,]+1)))
      }
      else{
        if(iter == tau0 && adaptive == TRUE){
          Xbarr <- apply(Ainit[,,1:(iter-2)],1:2,mean)
          Xbar <- apply(Ainit[,,1:(iter-1)],1:2,mean)
        }
        if(iter > tau0 && adaptive == TRUE){
          Xbarr <- Xbarr + (A_prevv-Xbarr)/(iter-2)
          Xbar <- Xbar + (A_prev-Xbar)/(iter-1)
          XX <- lapply(1:m,function(j) Xbar[,which(ind==j)]
            %*%t(A_prev[,which(ind==j)]))
          E <- lapply(1:m,function(j) (iter-3)/(iter-2)*E[[
            j]]+sK/(iter-2)*(1/(iter-1)*((iter-2)*Xbarr[,
            which(ind==j)]%*%t(Xbarr[,which(ind==j)])-(
            iter-1)*(XX[[j]]+t(XX[[j]]))+iter*A_prev[,
            which(ind==j)]%*%t(A_prev[,which(ind==j)]))+
            diag(eps,K)))
        }
        rem <- (iter - numNA)/L - floor((iter - numNA)/L)
          == 0
        if(sum(rem) > 0){
          for(j in which(rem)){
            if(acc_prob[j] < .25){
              E[[j]] <- E[[j]]*0.95^(2)
            }
            if(acc_prob[j] > .5){
              E[[j]] <- E[[j]]*1.05^(2)
            }
          }
        }
        if(MHalg == 1){
          MH <- lapply(1:m,function(j) sampleAlpha(j,state,
             Q,z,A_prev[,which(ind==j)],mu_A,Sigma_A,E[[j
             ]]))
        }
        if(MHalg == 2){
          MH <- lapply(1:m,function(j) sampleAlphaDR(j,
             state,Q,z,A_prev[,which(ind==j)],mu_A,Sigma_A,
             E[[j]],numDR))
        }
        if(MHalg == 3){
          MH <- lapply(1:m,function(j) sampleAlphaMTM(j,
             state,Q,z,A_prev[,which(ind==j)],mu_A,Sigma_A,
             E[[j]],numMTM))
        }
        if(MHalg == 4){
          MH <- lapply(1:m,function(j) sampleAlphaHMC(j,
             state,Q,z,A_prev[,which(ind==j)],mu_A,Sigma_A,
             epsHMC,numHMC))
        }
        if(MHalg == 5){
          MH <- lapply(1:m,function(j) sampleAlphaSS(j,
             state,Q,z,A_prev[,which(ind==j)],mu_A,Sigma_A,
             sliceWD))
        }
        A_curr <- matrix(sapply(1:m,function(j) MH[[j]]$
           alpha),K,m*m)
# Update the transition matrices, Q
        Mat <- z[-1,]%*%A_curr
        Q[,,-1] <- aperm(array(t(Mat),dim=c(m,m,n-1)),c
           (2,1,3))
        Q[,,-1] <- exp(Q[,,-1])/aperm(array(rep(apply(exp(Q
           [,,-1]),c(1,3),sum),each=m),dim=c(m,m,n-1)),c
           (2,1,3))
        numNA <- numNA + sapply(1:m,function(j) as.numeric(
           is.na(MH[[j]]$prob)))
        acc_prob <- sapply(1:m,function(j) if(is.na(MH[[j]]
           $prob)){acc_prob[j]} else{acc_prob[j] + (MH[[j]]
           $prob - acc_prob[j])/(iter-(numNA[j]+1))})
      }
# Update the parameter, mu
      mu_curr <- t(sapply(1:m,function(x) mvrnormArma(1,
         solve(nu[x]*solve(Sigma_prev[,,x])+solve(Sigma_M))
         %*%(solve(Sigma_prev[,,x])%*%matrix(apply(matrix(
         obs[which(state == x),],ncol=d),2,sum))+solve(
         Sigma_M)%*%matrix(mu_M)),solve(nu[x]*solve(Sigma_
         prev[,,x])+solve(Sigma_M)))))
# Update the parameter, Sigma
      Sigma_curr <- array(unlist(lapply(1:m,function(x)
         riwish(nu[x]+gamma_S,t(obs[which(state == x),] -
        matrix(mu_curr[x,],sum(state == x),d,TRUE))%*%(obs
        [which(state == x),] - matrix(mu_curr[x,],sum(
        state == x),d,TRUE))+Lambda_S))),dim=c(d,d,m))
# Imputation of the means of the parameters
    if(iter < iter0 && iter >= iter0Alpha && (iter-
        iter0Alpha)/thinningAlpha - floor((iter-iter0Alpha
        )/thinningAlpha) == 0){
      MAV[,,(iter-iter0Alpha)/thinningAlpha] <- mu_curr
      SAV[,,,(iter-iter0Alpha)/thinningAlpha] <- Sigma_
         curr
      if(missing(z)){
        QAV[,,(iter-iter0Alpha)/thinningAlpha] <- Q_curr
      }
      else{
        AAV[,,(iter-iter0Alpha)/thinningAlpha] <- A_curr
      }
    }
# Initialise the means of the parameters
    if(iter == iter0){
      if(missing(z)){
        QAV <- apply(QAV,1:2,median)
      }
      else{
        AAV <- z[-1,]%*%apply(AAV,1:2,mean)
        Q0[,,-1] <- aperm(array(t(AAV),dim=c(m,m,n-1)),c
           (2,1,3))
        Q0[,,-1] <- exp(Q0[,,-1])/aperm(array(rep(apply(
           exp(Q0[,,-1]),c(1,3),sum),each=m),dim=c(m,m,n
           -1)),c(2,1,3))
        QAV <- Q0
      }
      MAV <- apply(MAV,1:2,mean)
      SAV <- apply(SAV,1:3,mean)
    }
# Update the medians by taking arithmetic means
    if(iter > iter0){
      if(missing(z)){
        QAV <- QAV + (Q_curr - QAV)/iter
      }
      else{
        QAV <- QAV + (Q - QAV)/iter
      }
      MAV <- MAV + (mu_curr - MAV)/iter
      SAV <- SAV + (Sigma_curr - SAV)/iter
    }
# Update the parameters of interest at the previous iteration
    mu_prev <- mu_curr
    Sigma_prev <- Sigma_curr
    if(missing(z)){
      Q_prev <- Q_curr
    }
    else{
      if(iter <= tau0 && adaptive == TRUE){
        Ainit[,,iter] <- A_curr
      }
      A_prevv <- A_prev
      A_prev <- A_curr
    }
   }
```

```
    if(iter >= burnin + thinning && (iter-burnin)/thinning
       - floor((iter-burnin)/thinning) == 0){
      mu[,,(iter - burnin)/thinning] <- mu_curr
      Sigma[,,,(iter - burnin)/thinning] <- Sigma_curr
      if(missing(z)){
        Q[,,(iter - burnin)/thinning] <- Q_curr
      }
      else{
        A[,,(iter - burnin)/thinning] <- A_curr
      }
      St[[(iter-burnin)/thinning]] <- state
      message(paste(iter,"th iteration is stored.",sep=""))
    }
  }
  if(missing(z)){
    result <- list(mu=mu,Sigma=Sigma,Q=Q,state=St,prob=
       state_prob)
  }
  else{
    result <- list(mu=mu,Sigma=Sigma,alpha=A,state=St,prob=
       state_prob,accept=acc_prob,CM=E)
  }
  return(result)
}
```

```
labelSwitchMvn <- function(mu,Sigma,St,A,Q){
#----------------------------------------------------------
# This function relabels the hidden states after the MCMC
   algorithm.
#----------------------------------------------------------
# Inputs
```

```
#-----------------------------------------------------------
# - mu: the MCMC samples of mu
# - Sigma: the MCMC samples of sigma
# - St: a sequence of the estimated hidden states
# - A: the MCMC samples of alpha
# - Q: the MCMC samples of transition matrix
#-----------------------------------------------------------
# Outputs
#-----------------------------------------------------------
# - mu: the relabelled MCMC samples of mu
# - Sigma: the relabelled MCMC samples of sigma
# - state: the relabelled MCMC samples of state
# - alpha: the relabelled MCMC samples of alpha
# - Q: the relabelled MCMC samples of Q
#-----------------------------------------------------------
# Set the indices and initialise the parameters
  m <- dim(mu)[1]
  d <- dim(mu)[2]
  N <- dim(mu)[3]
  perm <- permn(1:m)
  M <- gamma(m+1)
  L <- numeric()
  P <- matrix(1:(m*m),nrow=m,ncol=m)
# Relabel the MCMC samples of interest
  for(iter in 1:N){
    L <- sapply(1:M,function(i) sum(mu[perm[[i]],,iter]==mu
       [order(mu[,1,iter]),,iter]))
    perm_curr <- perm[[which.max(L)]]
    mu[,,iter] <- mu[perm_curr,,iter]
    Sigma[,,,iter] <- Sigma[,,perm_curr,iter]
    St[[iter]] <- mapvalues(St[[iter]],1:m,perm_curr)
    if(missing(Q)){
      A[,,iter] <- A[,as.vector(P[perm_curr,perm_curr]),
         iter]
    }
    else{
      Q[,,iter] <- Q[perm_curr,perm_curr,iter]
    }
  }
  if(missing(Q)){
    result <- list(mu=mu,Sigma=Sigma,alpha=A,state=St)
  }
  else{
    result <- list(mu=mu,Sigma=Sigma,Q=Q,state=St)
  }
  return(result)
}
```

```
marLikMvn <- function(mu,Sigma,state,obs,z,alpha=NULL,Q=
    NULL){
#-----------------------------------------------------------
# This function calculates and returns a log marginal
    likelihood for the MCMC samples of interest.
#-----------------------------------------------------------
# Inputs
#-----------------------------------------------------------
# - mu: the MCMC samples of mu
# - Sigma:  the MCMC samples of Sigma
# - state: the MCMC samples of hidden states
# - obs: the observations
# - z: the exogenous variables
# - alpha: the MCMC samples of alpha
# - Q: the MCMC samples of the transition matrix/matrices
#-----------------------------------------------------------
# Outputs
#-----------------------------------------------------------
# - fhat: the log likelihoods
# - phat: the log priors
# - Phat: the log posteriors
#-----------------------------------------------------------
  m <- dim(mu)[1]
  n <- dim(obs)[1]
  d <- dim(obs)[2]
  N <- length(state)
  rho <- rep(1/m,m)
  if(!missing(z)){
    K <- ncol(z)
    mu_A <- numeric(K)
    Sigma_A <- solve(diag(10,K))
    CM <- diag(13,K)
  }
  signMat <- sign(cov(obs))
  mu_M <- apply(obs,2,function(x) mean(range(x)))
  Sigma_M <- diag(apply(obs,2,function(x) diff(range(x))))
  idx_M <- which(Sigma_M == 0,arr.ind=TRUE)
  Sigma_M[idx_M] <- signMat[idx_M]*sapply(1:nrow(idx_M),
     function(x) sqrt(Sigma_M[idx_M[x,1],idx_M[x,1]]*Sigma_
     M[idx_M[x,2],idx_M[x,2]])/2)
  gamma_S <- trunc((d+1)/2) + 1
  Lambda_S <- diag(gamma_S,d)
  idx_S <- which(Lambda_S == 0,arr.ind=TRUE)
```

```r
  Lambda_S[idx_S] <- signMat[idx_S]*gamma_S/2
  muStar <- apply(mu,1:2,mean)
  SigmaStar <- apply(Sigma,1:3,mean)
  if(missing(z)){
    QStar <- apply(Q,1:2,mean)
  }
  else{
    alphaStar <- apply(alpha,1:2,mean)
  }
# Define the posterior density function
  f <- function(A){
    prod((exp(z%*%A)/rowSums(exp(z%*%A)))[mat_dims])*exp(-1/2*sum(diag((A-mu_A)%*%t(A-mu_A)%*%Sigma_A)))
  }
# Compute the log likelihood
  fhat <- t(sapply(1:n,function(x) dmvnrm_arma(obs[x,],muStar,SigmaStar)))
  fhat[1,] <- fhat[1,]*rho
  for(i in 2:n){
    if(missing(z)){
      fhat[i,] <- (fhat[i-1,]/sum(fhat[i-1,]))%*%QStar*fhat[i,]
    }
    else{
      P <- lapply(1:dim(alpha)[3],function(x) matrix(exp(z[i,]%*%alpha[,,x]),m,m,TRUE))
      P <- lapply(1:length(P),function(x) P[[x]]/rowSums(P[[x]]))
      P <- array(unlist(P),dim=c(m,m,length(P)))
      fhat[i,] <- (fhat[i-1,]/sum(fhat[i-1,]))%*%apply(P
        ,1:2,median)*fhat[i,]
    }
  }
  fhat <- sum(log(apply(fhat,1,sum)))
# Compute the log priors
  phat <- sum(sapply(1:m,function(x) log(dmvnorm(muStar[x
     ,],mu_M,Sigma_M))) + sum(sapply(1:m,function(x) log(
     diwish(SigmaStar[,,x],gamma_S,Lambda_S))))
  if(missing(z)){
    phat <- phat + sum(sapply(1:m,function(x) log(
       ddirichlet(QStar[x,],rep(1,m)))))
  }
  else{
    phat <- phat + sum(sapply(which(diag(m)==0),function(x)
        log(dmvnorm(alphaStar[,x],mu_A,solve(Sigma_A)))))
  }
# Compute the log posteriors
  alphaPool1 <- alphaPool2 <- numeric(m)
  estMu <- estSigma <- estQ <- estAlpha1 <- estAlpha2 <-
     numeric(N)
  for(iter in 1:N){
    nu <- sapply(1:m,function(x) sum(state[[iter]]==x))
    estMu[iter] <- prod(sapply(1:m,function(x) dmvnorm(
       muStar[x,],solve(nu[x]*solve(Sigma[,,x,iter])+solve(
       Sigma_M))%*%(solve(Sigma[,,x,iter])%*%matrix(apply(
       matrix(obs[which(state[[iter]]==x),],ncol=d),2,sum))
       +solve(Sigma_M)%*%matrix(mu_M)),solve(nu[x]*solve(
       Sigma[,,x,iter])+solve(Sigma_M)))))
    estSigma[iter] <- prod(sapply(1:m,function(x) diwish(
       SigmaStar[,,x],nu[x]+gamma_S,t(obs[which(state[[iter
        ]] == x),] - matrix(muStar[x,],sum(state[[iter]] ==
        x),d,TRUE))%*%(obs[which(state[[iter]] == x),] -
        matrix(muStar[x,],sum(state[[iter]] == x),d,TRUE))+
        Lambda_S)))
    if(missing(z)){
      tbl <- table(state[[iter]][-n],state[[iter]][-1])
      estQ[iter] <- prod(sapply(1:m,function(x) ddirichlet(
        QStar[x,],tbl[x,]+1)))
    }
    else{
      for(k in 1:m){
        idx <- rep(1:m,each=m)
        ind <- which(state[[iter]][-n]==k)
        mat_ind <- (1:m)[-k]
        mat_dims <- matrix(c(ind+1,state[[iter]][ind+1]),
          ncol=2)
        alphaPool1[k] <- min(1,f(alphaStar[,which(idx==k)])
          /f(alpha[,which(idx==k),iter]))*prod(sapply(
          which(idx==k),function(x) dmvnorm(alphaStar[,x],
          alpha[,x,iter],CM))[mat_ind])
        alphaPool2[k] <- min(1,f(alpha[,which(idx==k),iter
          ])/f(alphaStar[,which(idx==k)]))
      }
      estAlpha1[iter] <- prod(alphaPool1)
      estAlpha2[iter] <- prod(alphaPool2)
    }
  }
  if(missing(z)){
    Phat <- log(mean(estQ)) + log(mean(estSigma)) + log(
      mean(estMu))
  }
  else{
    Phat <- log(mean(estAlpha1)) - log(mean(estAlpha2)) +
       log(mean(estSigma)) + log(mean(estMu))
  }
  return(c(fhat,phat,-Phat))
}
```

F.2 C++

```
#include <RcppArmadillo.h>
// [[Rcpp::depends(RcppArmadillo)]]
using namespace Rcpp;
// [[Rcpp::export]]
arma::mat mvrnormArma(int n, arma::vec mu, arma::mat Sigma)
   {
  int ncols = Sigma.n_cols;
  arma::mat Y = arma::randn(n,ncols);
  return arma::repmat(mu,1,n).t() + Y * arma::chol(Sigma);
}
```

```
#include <RcppArmadilloExtensions/sample.h>
// [[Rcpp::depends(RcppArmadillo)]]
using namespace Rcpp;
const double log2pi = std::log(2.0 * M_PI);
// [[Rcpp::export]]
arma::rowvec dmvnrm_arma(arma::rowvec x, arma::mat mean,
   arma::cube sigma, bool logd = false) {
    int m = mean.n_rows;
    int xdim = mean.n_cols;
    arma::rowvec out(m);
    double constants = -(static_cast<double>(xdim)/2.0) *
       log2pi;
    for (int i=0; i < m; i++) {
    arma::mat rooti = arma::trans(arma::inv(trimatu(arma::
       chol(sigma.slice(i)))));
    double rootisum = arma::sum(log(rooti.diag()));
        arma::vec z = rooti * arma::trans(x - mean.row(i))
           ;
        out(i)      = constants - 0.5 * arma::sum(z%z) +
           rootisum;
    }
    if (logd == false) {
        out = exp(out);
    }
    return(out);
}
// [[Rcpp::export]]
List ffbsCmatMvn(arma::mat obs, arma::rowvec rho, arma::mat
     Q, arma::mat mu, arma::cube Sigma){
    int m = rho.n_elem;
    int n = obs.n_rows;
    IntegerVector state(n);
  IntegerVector space = seq_len(m) - 1;
    arma::mat fwd(n,m);
    arma::mat bwd(n,m);
// Initialisation of the forward variable
    fwd.row(0) = rho%dmvnrm_arma(obs.row(0),mu,Sigma);
    fwd.row(0) = fwd.row(0)/sum(fwd.row(0));
// Execute the forward filtering recursion
```

```
    for(int t = 1; t < n; t++){
        fwd.row(t) = (fwd.row(t-1)*Q)%dmvnrm_arma(obs.row(t
           ),mu,Sigma);
        fwd.row(t) = fwd.row(t)/sum(fwd.row(t));
    }
// Initialisation of the backward variable
    bwd.row(n-1) = fwd.row(n-1);
    state(n-1) = bwd.row(n-1).index_max();
// Execute the backward sampling recursion
    for(int t = n-1; t > 0; t--){
        bwd.row(t-1) = fwd.row(t-1)%arma::conv_to<arma::
           rowvec>::from(Q.col(state(t)));
        bwd.row(t-1) = bwd.row(t-1)/sum(bwd.row(t-1));
        state(t-1) = RcppArmadillo::sample(space,1,false,as
           <NumericVector>(wrap(arma::conv_to<arma::rowvec
           >::from(bwd.row(t-1)))))(0);
    }
    return List::create(Named("state",state),Named("prob",
       bwd));
}
```

```
#include <RcppArmadilloExtensions/sample.h>
// [[Rcpp::depends(RcppArmadillo)]]
using namespace Rcpp;
const double log2pi = std::log(2.0 * M_PI);
// [[Rcpp::export]]
arma::rowvec dmvnrm_arma(arma::rowvec x, arma::mat mean,
   arma::cube sigma, bool logd = false) {
    int m = mean.n_rows;
    int xdim = mean.n_cols;
    arma::rowvec out(m);
    double constants = -(static_cast<double>(xdim)/2.0) *
       log2pi;
    for (int i=0; i < m; i++) {
    arma::mat rooti = arma::trans(arma::inv(trimatu(arma::
       chol(sigma.slice(i)))));
    double rootisum = arma::sum(log(rooti.diag()));
        arma::vec z = rooti * arma::trans(x - mean.row(i))
           ;
        out(i)      = constants - 0.5 * arma::sum(z%z) +
           rootisum;
    }
    if (logd == false) {
        out = exp(out);
    }
    return(out);
}
// [[Rcpp::export]]
List ffbsCmatMvn(arma::mat obs, arma::rowvec rho, arma::mat
    Q, arma::mat mu, arma::cube Sigma){
    int m = rho.n_elem;
    int n = obs.n_rows;
    IntegerVector state(n);
  IntegerVector space = seq_len(m) - 1;
    arma::mat fwd(n,m);
    arma::mat bwd(n,m);
// Initialisation of the forward variable
    fwd.row(0) = rho%dmvnrm_arma(obs.row(0),mu,Sigma);
    fwd.row(0) = fwd.row(0)/sum(fwd.row(0));
// Execute the forward filtering recursion
    for(int t = 1; t < n; t++){
        fwd.row(t) = (fwd.row(t-1)*Q)%dmvnrm_arma(obs.row(t),mu,Sigma);
        fwd.row(t) = fwd.row(t)/sum(fwd.row(t));
    }
// Initialisation of the backward variable
    bwd.row(n-1) = fwd.row(n-1);
    state(n-1) = bwd.row(n-1).index_max();
// Execute the backward sampling recursion
    for(int t = n-1; t > 0; t--){
        bwd.row(t-1) = fwd.row(t-1)%arma::conv_to<arma::rowvec>::from(Q.col(state(t)));
        bwd.row(t-1) = bwd.row(t-1)/sum(bwd.row(t-1));
        state(t-1) = RcppArmadillo::sample(space,1,false,as<NumericVector>(wrap(arma::conv_to<arma::rowvec>::from(bwd.row(t-1)))))(0);
    }
    return List::create(Named("state",state),Named("prob",bwd));
}
```

References

Ailliot, P., Thompson, C. and Thomson, P. (2009), ‘Space–time modelling of precipitation by using a hidden Markov model and censored Gaussian distributions’, *Journal of the Royal Statistical Society: Series C (Applied Statistics)* **58**(3), 405–426.

Baum, L. E. and Eagon, J. A. (1967), ‘An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology’, *Bulletin of the American Mathematical Society* **73**(3), 360–363.

Baum, L. E. and Petrie, T. (1966), ‘Statistical inference for probabilistic functions of finite state Markov chains’, *The Annals of Mathematical Statistics* **37**(6), 1554–1563.

Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970), ‘A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains’, *The Annals of Mathematical Statistics* **41**(1), 164–171.

Bayes, T., Price, R. and Canton, J. (1763), *An essay towards solving a problem in the doctrine of chances*, C. Davis, Printer to the Royal Society of London.

Bédard, M., Douc, R. and Moulines, E. (2012), ‘Scaling analysis of multiple-try MCMC methods’, *Stochastic Processes and their Applications* **122**(3), 758–786.

Bellman, R. E. (2015), *Adaptive control processes: a guided tour*, Princeton University Press.

Bellone, E., Hughes, J. P. and Guttorp, P. (2000), ‘A hidden Markov model for downscaling synoptic atmospheric patterns to precipitation amounts’, *Climate Research* **15**(1), 1–12.

Birkhoff, G. D. (1931), 'Proof of the ergodic theorem', *Proceedings of the National Academy of Sciences* **17**(12), 656–660.

Bishop, C. M. (2006), 'Pattern recognition', *Machine Learning* **128**, 1–58.

Box, G. E., Jenkins, G. M., Reinsel, G. C. and Ljung, G. M. (2015), *Time series analysis: forecasting and control*, John Wiley & Sons.

Boys, R. J. and Henderson, D. A. (2004), 'A Bayesian approach to DNA sequence segmentation', *Biometrics* **60**(3), 573–581.

Boys, R. J., Henderson, D. A. and Wilkinson, D. J. (2000), 'Detecting homogeneous segments in DNA sequences by using hidden Markov models', *Applied Statistics* pp. 269–285.

Brooks, S., Gelman, A., Jones, G. and Meng, X.-L. (2011), *Handbook of Markov chain Monte Carlo*, CRC Press.

Casella, G. and George, E. I. (1992), 'Explaining the Gibbs sampler', *The American Statistician* **46**(3), 167–174.

Celeux, G., Hurn, M. and Robert, C. P. (2000), 'Computational and inferential difficulties with mixture posterior distributions', *Journal of the American Statistical Association* **95**(451), 957–970.

Chib, S. (1995), 'Marginal likelihood from the Gibbs output', *Journal of the American Statistical Association* **90**(432), 1313–1321.

Chib, S. (1996), 'Calculating posterior distributions and modal estimates in Markov mixture models', *Journal of Econometrics* **75**(1), 79–97.

Chib, S. and Jeliazkov, I. (2001), 'Marginal likelihood from the Metropolis–Hastings output', *Journal of the American Statistical Association* **96**(453), 270–281.

Churchill, G. A. (1989), 'Stochastic models for heterogeneous DNA sequences', *Bulletin of Mathematical Biology* **51**(1), 79–94.

Coast, D. A., Stern, R. M., Cano, G. G. and Briller, S. A. (1990), 'An approach to cardiac arrhythmia analysis using hidden Markov models', *IEEE Transactions on Biomedical Engineering* **37**(9), 826–836.

Cowles, M. K. and Carlin, B. P. (1996), 'Markov chain Monte Carlo convergence diagnostics: a comparative review', *Journal of the American Statistical Association* **91**(434), 883–904.

Craiu, R. V. and Lemieux, C. (2007), 'Acceleration of the multiple-try Metropolis algorithm using antithetic and stratified sampling', *Statistics and Computing* **17**(2), 109–120.

Crouse, M. S., Nowak, R. D. and Baraniuk, R. G. (1998), 'Wavelet-based statistical signal processing using hidden Markov models', *IEEE Transactions on Signal Processing* **46**(4), 886–902.

de Laplace, P. S. (1820), *Théorie analytique des probabilités*, Vol. 7, Courcier.

Del Moral, P., Doucet, A. and Jasra, A. (2012), 'An adaptive sequential Monte Carlo method for approximate Bayesian computation', *Statistics and Computing* **22**(5), 1009–1020.

Dellaportas, P., Denison, D. G. and Holmes, C. (2007), 'Flexible threshold models for modelling interest rate volatility', *Econometric Reviews* **26**(2-4), 419–437.

Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977), 'Maximum likelihood from incomplete data via the EM algorithm', *Journal of the Royal Statistical Society: Series B (Methodological)* pp. 1–38.

Diebold, F. X., Lee, J.-H. and Weinbach, G. C. (1994), 'Regime switching with time-varying transition probabilities', *Business Cycles: Durations, Dynamics, and Forecasting* pp. 144–165.

Donoho, D. L. et al. (2000), 'High-dimensional data analysis: The curses and blessings of dimensionality', *AMS Math Challenges Lecture* **1**(2000).

Eddelbuettel, D., François, R., Allaire, J., Ushey, K., Kou, Q., Russel, N., Chambers, J. and Bates, D. (2011), 'Rcpp: Seamless R and C++ integration', *Journal of Statistical Software* **40**(8), 1–18.

Edwards, W., Lindman, H. and Savage, L. J. (1963), 'Bayesian statistical inference for psychological research.', *Psychological Review* **70**(3), 193–242.

Filardo, A. J. and Gordon, S. F. (1998), 'Business cycle durations', *Journal of Econometrics* **85**(1), 99–123.

Fridlyand, J., Snijders, A. M., Pinkel, D., Albertson, D. G. and Jain, A. N. (2004), 'Hidden Markov models approach to the analysis of array CGH data', *Journal of Multivariate Analysis* **90**(1), 132–153.

Gamerman, D. and Lopes, H. F. (2006), *Markov chain Monte Carlo: stochastic simulation for Bayesian inference*, CRC Press.

Gelfand, A. E. and Smith, A. F. (1990), 'Sampling-based approaches to calculating marginal densities', *Journal of the American Statistical Association* **85**(410), 398–409.

Geman, S. and Geman, D. (1984), 'Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images', *IEEE Transactions on Pattern Analysis and Machine Intelligence* (6), 721–741.

Geweke, J. et al. (1991), *Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments*, Vol. 196, Federal Reserve Bank of Minneapolis, Research Department Minneapolis, MN, USA.

Gilks, W. R., Roberts, G. O. and George, E. I. (1994), 'Adaptive direction sampling', *The Statistician* pp. 179–189.

Gilks, W. R., Roberts, G. O. and Sahu, S. K. (1998), 'Adaptive Markov chain Monte Carlo through regeneration', *Journal of the American Statistical Association* **93**(443), 1045–1054.

Haario, H., Laine, M., Mira, A. and Saksman, E. (2006), ‘DRAM: efficient adaptive MCMC’, *Statistics and Computing* **16**(4), 339–354.

Haario, H., Saksman, E. and Tamminen, J. (2001), ‘An adaptive Metropolis algorithm’, *Bernoulli* pp. 223–242.

Hamilton, J. D. (1988), ‘Rational-expectations econometric analysis of changes in regime: An investigation of the term structure of interest rates’, *Journal of Economic Dynamics and Control* **12**(2-3), 385–423.

Hamilton, J. D. (1994), *Time series analysis*, Vol. 2, Princeton, NJ: Princeton University Press.

Hastings, W. K. (1970), ‘Monte Carlo sampling methods using Markov chains and their applications’, *Biometrika* **57**(1), 97–109.

Heaps, S. E., Boys, R. J. and Farrow, M. (2015), ‘Bayesian modelling of rainfall data by using non-homogeneous hidden Markov models and latent Gaussian variables’, *Journal of the Royal Statistical Society: Series C (Applied Statistics)* **64**(3), 543–568.

Heidelberger, P. and Welch, P. D. (1983), ‘Simulation run length control in the presence of an initial transient’, *Operations Research* **31**(6), 1109–1144.

Holsclaw, T., Greene, A. M., Robertson, A. W. and Smyth, P. (2016), ‘A Bayesian hidden Markov model of daily precipitation over South and East Asia’, *Journal of Hydrometeorology* **17**(1), 3–25.

Holsclaw, T., Greene, A. M., Robertson, A. W., Smyth, P. et al. (2017), ‘Bayesian nonhomogeneous Markov models via Pólya-Gamma data augmentation with applications to rainfall modeling’, *The Annals of Applied Statistics* **11**(1), 393–426.

Hughes, J. P., Guttorp, P. and Charles, S. P. (1997), ‘A nonhomogeneous hidden Markov model for precipitation’.

Hughes, J. P., Guttorp, P. and Charles, S. P. (1999), ‘A non-homogeneous hidden Markov model for precipitation occurrence’, *Journal of the Royal Statistical Society: Series C (Applied Statistics)* **48**(1), 15–30.

Jenkins, J. M., Wu, D. and Arzbaecher, R. C. (1979), ‘Computer diagnosis of supraventricular and ventricular arrhythmias. A new esophageal technique.’, *Circulation* **60**(5), 977–987.

Koki, C., Meligkotsidou, L. and Vrontos, I. (2018), ‘Bayesian analysis of predictive Non-Homogeneous hidden Markov models using Polya-Gamma data augmentation’, *arXiv preprint arXiv:1802.02825* .

Laplace, P. S. (1986), ‘Memoir on the probability of the causes of events’, *Statistical Science* **1**(3), 364–378.

Link, W. A. and Eaton, M. J. (2012), ‘On thinning of chains in MCMC’, *Methods in Ecology and Evolution* **3**(1), 112–115.

Liu, J. S. and Chen, R. (1995), ‘Blind deconvolution via sequential imputations’, *Journal of the American Statistical Association* **90**(430), 567–576.

Liu, J. S., Liang, F. and Wong, W. H. (2000), ‘The multiple-try method and local optimization in Metropolis sampling’, *Journal of the American Statistical Association* **95**(449), 121–134.

MacEachern, S. N. and Berliner, L. M. (1994), ‘Subsampling the Gibbs sampler’, *The American Statistician* **48**(3), 188–190.

McNeish, D. (2016), ‘On using Bayesian methods to address small sample problems’, *Structural Equation Modeling: A Multidisciplinary Journal* **23**(5), 750–773.

Meligkotsidou, L. and Dellaportas, P. (2011), ‘Forecasting with non-homogeneous hidden Markov models’, *Statistics and Computing* **21**(3), 439–449.

Mengersen, K. L., Tweedie, R. L. et al. (1996), ‘Rates of convergence of the Hastings and Metropolis algorithms’, *The Annals of Statistics* **24**(1), 101–121.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953), ‘Equation of state calculations by fast computing machines’, *The Journal of Chemical Physics* **21**(6), 1087–1092.

Metropolis, N. and Ulam, S. (1949), ‘The Monte Carlo method’, *Journal of the American Statistical Association* **44**(247), 335–341.

Meyn, S. P. and Tweedie, R. L. (2012), *Markov chains and stochastic stability*, Springer Science & Business Media.

Mira, A. and Tierney, L. (2002), ‘Efficiency and convergence properties of slice samplers’, *Scandinavian Journal of Statistics* **29**(1), 1–12.

Mira, A. et al. (2001), ‘On Metropolis-Hastings algorithms with delayed rejection’, *Metron* **59**(3-4), 231–241.

Murphy, K. P. (2012), *Machine learning: a probabilistic perspective*, MIT Press.

Myung, I. J. and Pitt, M. A. (1997), ‘Applying Occams razor in modeling cognition: A Bayesian approach’, *Psychonomic Bulletin & Review* **4**(1), 79–95.

Owen, A. B. (2017), ‘Statistically efficient thinning of a Markov chain sampler’, *Journal of Computational and Graphical Statistics* **26**(3), 738–744.

Peskun, P. H. (1973), ‘Optimum Monte-Carlo sampling using Markov chains’, *Biometrika* **60**(3), 607–612.

Putnam, B. H. and Quintana, J. M. (1994), New Bayesian statistical approaches to estimating and evaluating models of exchange rates determination, *in* ‘Proceedings of the ASA Section on Bayesian Statistical Science’, Vol. 1994, American Statistical Association, pp. 232–237.

Quintana, J. (1992), ‘Optimal portfolios of forward currency contracts’, *Bayesian Statistics* **4**, 753–762.

R Core Team (2016), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.
URL: *https://www.r-project.org/*

Rabiner, L. R. (1989), ‘A tutorial on hidden Markov models and selected applications in speech recognition’, *Proceedings of the IEEE* **77**(2), 257–286.

Rabiner, L. R. and Juang, B.-H. (1986), ‘An introduction to hidden Markov models’, *ASSP Magazine, IEEE* **3**(1), 4–16.

Raymond, J. E. and Rich, R. W. (1997), ‘Oil and the macroeconomy: A Markov state-switching approach’, *Journal of Money, Credit, and Banking* pp. 193–213.

Richardson, S. and Green, P. J. (1997), ‘On Bayesian analysis of mixtures with an unknown number of components (with discussion)’, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **59**(4), 731–792.

Robert, C. P. and Casella, G. (1999), The Metropolis-Hastings Algorithm, *in* ‘Monte Carlo Statistical Methods’, Springer New York, pp. 231–283.

Roberts, G. O., Gelman, A., Gilks, W. R. et al. (1997), ‘Weak convergence and optimal scaling of random walk Metropolis algorithms’, *The Annals of Applied Probability* **7**(1), 110–120.

Roberts, G. O. and Rosenthal, J. S. (2009), ‘Examples of adaptive MCMC’, *Journal of Computational and Graphical Statistics* **18**(2), 349–367.

Robertson, A. W., Kirshner, S. and Smyth, P. (2004), ‘Downscaling of daily rainfall occurrence over northeast Brazil using a hidden Markov model’, *Journal of Climate* **17**(22), 4407–4424.

Rosenthal, J. S. (1995), ‘Minorization conditions and convergence rates for Markov chain Monte Carlo’, *Journal of the American Statistical Association* **90**(430), 558–566.

Rouder, J. N., Speckman, P. L., Sun, D., Morey, R. D. and Iverson, G. (2009), ‘Bayesian t tests for accepting and rejecting the null hypothesis’, *Psychonomic Bulletin & Review* **16**(2), 225–237.

Rueda, O. M., Rueda, C. and Diaz-Uriarte, R. (2013), ‘A Bayesian HMM with random effects and an unknown number of states for DNA copy number analysis’, *Journal of Statistical Computation and Simulation* **83**(1), 82–96.

Russell, S., Norvig, P. and Intelligence, A. (1995), 'A modern approach', *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs* **25**(27), 79–80.

Rydén, T. et al. (2008), 'EM versus Markov chain Monte Carlo for estimation of hidden Markov models: A computational perspective', *Bayesian Analysis* **3**(4), 659–688.

Scott, S. L. (2002), 'Bayesian methods for hidden Markov models: Recursive computing in the 21st century', *Journal of the American Statistical Association* **97**(457), 337–351.

Scott, S. L. (2011), 'Bayesian methods for hidden Markov models', *Journal of the American Statistical Association* .

Smith, A. F. and Roberts, G. O. (1993), 'Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods', *Journal of the Royal Statistical Society: Series B (Methodological)* pp. 3–23.

Sperrin, M., Jaki, T. and Wit, E. (2010), 'Probabilistic relabelling strategies for the label switching problem in Bayesian mixture models', *Statistics and Computing* **20**(3), 357–366.

Spezia, L. (2006), 'Bayesian analysis of non-homogeneous hidden Markov models', *Journal of Statistical Computation and Simulation* **76**(8), 713–725.

Spezia, L. (2010), 'Bayesian analysis of multivariate Gaussian hidden Markov models with an unknown number of regimes', *Journal of Time Series Analysis* **31**(1), 1–11.

Spezia, L., Brewer, M. J. and Birkel, C. (2017), 'An anisotropic and inhomogeneous hidden Markov model for the classification of water quality spatio-temporal series on a national scale: The case of Scotland', *Environmetrics* **28**(1).

Spezia, L., Cooksley, S. L., Brewer, M. J., Donnelly, D. and Tree, A. (2014), 'Mapping species distributions in one dimension by non-homogeneous hidden Markov models: the case of freshwater pearl mussels in the River Dee', *Environmental and Ecological Statistics* **21**(3), 487–505.

Spezia, L., Friel, N. and Gimona, A. (2018), 'Spatial hidden Markov models and species distributions', *Journal of Applied Statistics* **45**(9), 1595–1615.

Spezia, L., Futter, M. N. and Brewer, M. J. (2011), 'Periodic multivariate Normal hidden Markov models for the analysis of water quality time series', *Environmetrics* **22**(3), 304–317.

Stephens, M. (2000), 'Dealing with label switching in mixture models', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **62**(4), 795–809.

Stigler, J., Ziegler, F., Gieseke, A., Gebhardt, J. C. M. and Rief, M. (2011), 'The complex folding network of single calmodulin molecules', *Science* **334**(6055), 512–516.

Stigler, S. M. (1982), 'Thomas Bayes's Bayesian inference', *Journal of the Royal Statistical Society: Series A (General)* pp. 250–258.

Tanner, M. A. and Wong, W. H. (1987), 'The calculation of posterior distributions by data augmentation', *Journal of the American Statistical Association* **82**(398), 528–540.

Thrun, S., Burgard, W. and Fox, D. (2005), *Probabilistic robotics*, MIT Press.

Tierney, L. (1994), 'Markov chains for exploring posterior distributions', *The Annals of Statistics* pp. 1701–1728.

Tierney, L. and Mira, A. (1999), 'Some adaptive Monte Carlo methods for Bayesian inference', *Statistics in Medicine* **18**(1718), 2507–2515.

Tong, H., Thanoon, B. and Gudmundsson, G. (1985), 'Threshold time series modeling of two Icelandic riverflow systems', *JAWRA Journal of the American Water Resources Association* **21**(4), 651–662.

Tsay, R. S. (1998), 'Testing and modeling multivariate threshold models', *Journal of the American Statistical Association* **93**(443), 1188–1202.

Tweedie, R. L. (1975), 'Sufficient conditions for ergodicity and recurrence of Markov chains on a general state space', *Stochastic Processes and their Applications* **3**(4), 385–403.

Zucchini, W. and MacDonald, I. L. (2009), *Hidden Markov models for time series: an introduction using R*, CRC Press.