Aspects of Population Markov Chain Monte Carlo and Reversible Jump Markov Chain Monte Carlo

Eleni Bakra

A Dissertation Submitted to the
Faculty of Information and Mathematical Sciences
at the University of Glasgow
for the degree of
Doctor of Philosophy

Department of Statistics

October 2009

Abstract

Over the past decade Bayesian methodology has gained in popularity because of the development of very powerful computational algorithms known as Markov chain Monte Carlo (MCMC). The idea of MCMC was first introduced by Metropolis et al. (1953) as a simulation method of energy levels of atoms in a crystalline structure. Later on, Hastings (1970) applied the idea to statistical problems. Since then, MCMC methods have been used to integrate complex and high dimensional functions as well as sample from complex and highly structured real life models. The main idea is to draw a sample from the distribution of interest by running a Markov chain for a long time while an accept/reject mechanism is used to correct the arbitrary proposal mechanism and simulate from the invariant distribution of interest. Then, the unknown normalising constant, the posterior expectation or the marginal distribution can be estimated.

Several MCMC techniques have been developed through the years, such as the Metropolis-Hastings algorithm (Hastings (1970)), the Gibbs sampler (Geman and Geman (1984), Gelfand and Smith (1990)), adaptive Monte Carlo methods (Gilks et al. (1994), Gilks et al. (1995), Gilks et al. (1998), Tierney and Mira (1999), Liu et al. (2001), Warns (2001)). The most widely used MCMC methods simulate a single chain that is approximately distributed from a stationary

distribution for long time. Population Monte Carlo methods such as the pinball sampler (Robert and Mengersen (2003)), the Adaptive Direction Sampling (ADS) - snooker sampler (Gilks et al. (1994)), the Real parameter Evolutionary Monte Carlo algorithm (Liang and Wong (2001)) and the tempered transition method (Neal (1996)) simulate a collection of chains instead of a single one that may be trapped under a single mode depending on the starting position of the chain.

In the first part of this thesis, sampling techniques of unknown target distributions are considered. Two new population MCMC samplers, the simplex sampler and tempered simplex sampler, are presented. The simplex sampler is based on the Nelder and Mead (1965) simplex method and performs well in cases of highly correlated target distributions. In the case of a multi-modal target distribution, another population MCMC sampler, the tempered simplex sampler, is introduced to sample efficiently from them. The tempered simplex sampler uses the simplex sampler to improve mixing under the modes as well as a tempering ladder to explore the sampling space without getting trapped under a single mode.

In the second part of this thesis, model selection problems are considered. In general, model selection is a significant problem in statistics. The aim is to choose which model out of a number of possible ones best describes the data. From a Bayesian point of view, a prior distribution is used to describe the prior beliefs for each model. Then, the posterior model probabilities quantitatively discriminate between competing models providing a more informative comparison of models. In addition, obtaining posterior model probabilities also permits model averaging of parameters. Unfortunately, the posterior model probability requires knowledge of the marginal likelihood which, in most cases, cannot be estimated precisely because it is not analytically tractable.

Several techniques have been introduced in the literature attempting estimation of the unknown posterior model probabilities, see for instance the exponentiated Bayesian information criterion known as BIC (Schwarz (1978), Kass and Raftery (1995), Kass and Wasserman (1995), Raftery (1996)), the Laplace approximation (Tierney and Kadane (1986)), the Laplace-Metropolis estimator (Lewis and Raftery (1997)), the power posterior method (Friel and Pettitt (2008)). Additional methods for obtaining posterior model probabilities also include, for example, birth-death process (Stephens (2000)), Carlin and Chib's method (Carlin and Chib (1995)) and MCMCMC (Markov chain Monte Carlo model composition - for graphical models) (Madigan and York (1997)).

Reversible Jump Markov Chain Monte Carlo (RJMCMC) (Green (1995), Richardson and Green (1997)) is another approach to Bayesian model selection problems. It is used to explore the sampling space that consists of several models of different dimension. Thus, a crucial choice for the performance of the algorithm is to choose the right proposal mechanism. Hence, various methods have been proposed in the literature to choose the proposal mechanism (Al-Awadhi et al. (2004), Brooks and Ehlers (2008), Brooks et al. (2003), Dellaportas et al. (2002)). Here, the aim is to create an automatic RJMCMC algorithm so that no tuning is necessary and for this reason, the performance of the algorithm does not depend on the skills of the user. Therefore, the full-conditional posterior distribution of the parameters is used as the proposal distribution to simulate the proposed parameter vector. Then, one has to decide whether to condition on all, some or none of the common parameters between the current and proposed model. It turns out that the best strategy is to implement block updates so that the proposed parameter vector is generated from the posterior distribution of the

parameters.

Unfortunately, in most cases the posterior distribution of the parameters is intractable. For this reason, deterministic approximations are used to estimate it in a one-off sampler. Moreover, it can also be shown that the posterior model probabilities can be approximated off-line. Hence, these approximations to the posterior model probabilities can be used as the proposal distribution to change the model dimension inside the RJMCMC algorithm. Therefore, the mixing of the algorithm is improved as the models with higher posterior probability are proposed more often. Furthermore, it may be worthwhile to investigate whether the RJMCMC algorithm improves any bad approximations to the posterior model probabilities.

Chapter 1 provides an introduction to population MCMC and RJMCMC methods and motivates the thesis.

Chapter 2 introduces a new population MCMC sampler, the simplex sampler which is designed to sample from highly correlated target distributions.

Chapter 3 introduces another population MCMC sampler, the tempered simplex sampler that is designed to sample efficiently from multi-modal target distributions.

Chapter 4 describes the construction of automatic proposal distributions to simulate the proposed parameter vector inside the RJMCMC algorithm.

Chapter 5 describes the construction of automatic proposal distributions to change the model dimension inside the RJMCMC algorithm.

Chapter 6 summarises the research in the thesis and provides areas of possible future research.

Acknowledgements

I would like to take this opportunity to thank everyone who has helped me complete this thesis. Firstly, my supervisor, Professor Nial Friel, who contributed his time and expertise to this thesis. I would also like to thank my second supervisor Professor Mike Titterington for the helpful comments he has given on this work. Thanks must also go to Dr Agostino Nobile for his useful comments over the first two years of my PhD and to all the other members of the Statistics department who have helped make the last few years a very enjoyable experience. I am very grateful for all the opportunities the department gave me. Also, I must thank the Faculty of Information and Mathematical Sciences for funding me throughout this research. Finally, I would like to thank my family for their love and support over these years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis consists of two different parts. In the first part, two new population Markov chain Monte Carlo (MCMC) samplers are introduced. The first sampler, named the simplex sampler is based on the Nelder and Mead (1965) simplex method and samples efficiently from highly correlated target distributions while the second one, named the tempered simplex sampler, uses a tempering ladder and, in addition, samples efficiently from multi-modal targets. In the second part, Reversible jump Markov chain Monte Carlo (RJMCMC) methods are considered in model selection problems. The aim is to improve the proposal mechanism of RJMCMC by constructing automatic proposal distributions that do not depend on the skills of the user since no tuning is necessary. First, a proposal distribution to generate the proposed parameter vector is suggested considering the number of parameters to be updated in the proposed state. Then, an independent proposal distribution is also used to change the model dimension so that each of the models is visited with the correct frequency.

The two main ideas of the thesis can be brought together by using population

MCMC techniques inside the RJMCMC algorithm. Hence, the target for future research is to follow the work of Jasra et al. (2007a,b) and to consider variable selection problems and use the RJMCMC algorithm to explore the sampling space while a population of chains is used instead of just a single one. For instance, a population of Markov chains may visit each of the several models and every time a jump is proposed to a new one, the current population of the chains could be used to choose the proposed model as well as to explore the current one.

In the following text, a quick introduction is given to several population MCMC techniques. Then, the main idea of the simplex sampler and tempering simplex sampler is presented. In addition, an introduction to model selection problems is made and some of the already existing techniques are considered. Then, our contribution to improving the proposal mechanism of the RJMCMC algorithm by constructing automatic proposal distributions, is presented.

In Bayesian inference the unknown parameters $\boldsymbol{\theta}$ are treated as random variables and a prior distribution $p(\boldsymbol{\theta})$ is used to express the beliefs about them prior to seeing the data $\boldsymbol{y}$. Then, the information that the data provide through the likelihood $L(\boldsymbol{y}|\boldsymbol{\theta})$ is combined with the prior distribution of the parameters through Bayes' rule so that

$$\pi(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{p(\boldsymbol{\theta})L(\boldsymbol{y}|\boldsymbol{\theta})}{\int\limits_{\boldsymbol{\theta}} p(\boldsymbol{\theta})L(\boldsymbol{y}|\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}}$$

where $\int\limits_{\boldsymbol{\theta}} p(\boldsymbol{\theta})L(\boldsymbol{y}|\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}$ is the normalising constant. See, for example, Gilks et al. (1996), Cappé and Robert (2000), Robert and Casella (2004) for a discussion of standard MCMC methods and implementation issues.

Bayesian methodology has been widely used thanks to MCMC algorithms. Metropolis et al. (1953) first introduced the MCMC idea in physics as a simulation method of energy levels of atoms in a crystalline structure. Later on, Hastings (1970) applied the idea to statistical problems and since then, MCMC methods have been used to integrate complex and high dimensional functions and sample from complex and highly structured real life models. The MCMC methods simulate a sample from the target distribution by running a Markov chain for sufficient time using an accept/reject mechanism to correct the arbitrary proposal mechanism. Then, this sample can be used to evaluate the unknown normalising constant, the posterior expectation or the marginal distribution.

The main idea of MCMC sampling is very simple. Suppose that the distribution of interest $\pi(\boldsymbol{x})$ with $\boldsymbol{x} \in \mathcal{A} \subseteq \mathbb{R}^n$ is known up to a normalising constant and it is impossible to sample directly from it because $\pi(\cdot)$ is too complex. For this reason, an aperiodic and irreducible Markov chain is constructed with state space $\mathcal{A}$ and stationary distribution $\pi(\cdot)$. If the chain runs for sufficient time, the simulated values are considered to be a representative sample from the target distribution after disregarding an initial phase. Hence, this sample can be used to summarise the distribution of interest.

It is not always easy to construct a Markov chain with the desired properties. For this reason, the user should be very careful with the implementation of the MCMC algorithms. One of the most important choices is the transition mechanism of the chain. Any irreducible and aperiodic chain has a stationary distribution where at time t the transition kernel $\mathcal{K}$ will converge to the actual target distribution as time $t \to \infty$ so that $\pi\mathcal{K} = \pi$ (Robert and Casella (2004)). For instance, given an observation $\boldsymbol{x}^t$ at time t that comes from the distribution of

interest so that $\boldsymbol{x}^t \sim \pi(\boldsymbol{x}^t)$ then at time $t+1$ another observation $\boldsymbol{x}'$ is simulated as $\boldsymbol{x}' \sim q(\boldsymbol{x}^t, \boldsymbol{x}')$ where $q(\cdot,\cdot)$ is the proposal distribution. Thus, the proposed move is accepted with probability $\min\left\{1, \frac{\pi(\boldsymbol{x}')}{\pi(\boldsymbol{x}^t)} \frac{q(\boldsymbol{x}',\boldsymbol{x}^t)}{q(\boldsymbol{x}^t,\boldsymbol{x}')}\right\}$ or it is rejected with probability $1 - \min\left\{1, \frac{\pi(\boldsymbol{x}')}{\pi(\boldsymbol{x}^t)} \frac{q(\boldsymbol{x}',\boldsymbol{x}^t)}{q(\boldsymbol{x}^t,\boldsymbol{x}')}\right\}$. If the proposed move is accepted then the new state of the chain is $\boldsymbol{x}^{t+1} = \boldsymbol{x}'$ otherwise, the chain stays where it currently is so that $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$. If the process runs for sufficiently long enough, then eventually, the obtained sample, after considering an initial phase, is an approximation to the distribution of interest $\pi(\cdot)$.

Moreover, it is essential for a Markov chain to be time reversible so that

$$\pi(\boldsymbol{x}^t)\mathcal{K}(\boldsymbol{x}^t, \boldsymbol{x}^{t+1}) = \pi(\boldsymbol{x}^{t+1})\mathcal{K}(\boldsymbol{x}^{t+1}, \boldsymbol{x}^t)$$

which implies that the chain runs forward and backward in time. Any transition kernel, $\mathcal{K}$, that satisfies the above equation will converge to the stationary distribution $\pi(\cdot)$.

Furthermore, there are a few issues that one should consider when MCMC methods are used, like the choice of the transition mechanism, the choice of the initial state, the number of chains to be used and the number of iteration for the initial transient phase. For an irreducible chain, the starting state does not affect stationarity because the chain will forget the initial state and will eventually converge to the stationary distribution. However, a good starting position for the chain may possibly lead to better mixing and faster convergence resulting in a shorter initial transient phase. The initial transient phase or burn in defines how many iterations should be used until the chain reaches the stationary distribution. In practice, a pilot run is used to define the number of iterations. Another issue

may be the number of chains to be used, i.e one long chain or several shorter ones. One long chain should ensure that the stationary distribution is reached, but it may not fully explore the posterior distribution. Whereas for several shorter chains, there is the issue of having to discard the burn-in of each chain. Thus, The number of samples from the posterior distribution is reduced compared to a single long chain.

Standard MCMC methods simulate a single component which is accepted or rejected with the usual Metropolis rule. If the candidate component lands in a lower probability area, it is likely to be rejected and the chain stays where it currently is. Thus, the mixing of the chain may be slow if a lot of rejections take place. Tierney and Mira (1999) show that it is possible to propose a second candidate component that takes into account the first rejection step at the current state. Then, the second candidate is accepted or rejected with probability that also takes into account the first rejection. The procedure can be continued until a new candidate is accepted. Therefore, the mixing is improved because the acceptance probability increases and the algorithm explores efficiently the sampling space. Then again, there are some disadvantages with the implementation of the method such as the complexity of the MCMC updating step and the additional computational effort that is needed every time that the proposed move is rejected and another one is applied given that rejection.

Population MCMC methods (Gilks et al. (1994), Gilks et al. (1995), Gilks et al. (1998), Liu et al. (2001), Warns (2001), Robert and Mengersen (2003)) run various Markov chains in parallel that interact with each other at the current state to propose the new one. In the following text, some of these techniques are presented. The main idea of the pinball sampler (Robert and Mengersen

(2003)) is to use a collection of Markov chains to explore the sampling space by producing iid samples. Here, the candidate components tend not to be proposed in areas where other components already exist. For this reason, a pseudo reference distribution is introduced as the product of the actual target distribution and some exponential terms. These exponential terms cause a repulsive effect around the other components thanks to zero probability areas (holes) that are created around the current components preventing in this way, the candidate ones from landing close to them. Consequently, the components are not likely to be gathered under one mode. The candidate components are accepted or rejected with the usual Metropolis-Hastings rule.

The candidate component may be rejected if it lands in a low probability area or it may be too close to one of the other current components. For this reason, it may be reasonable to propose another component further away than the already rejected one and then, estimate the new acceptance probability which takes into account the first rejection. If the second candidate is also rejected then another move can be implemented by taking into account the previous rejection. Here, the main difference with the Tierney and Mira (1999) algorithm is that the move does not only depend on the previous moves but it also depends on the other components.

Parallel Tempering (Geyer (1991)) is another population MCMC method that uses a temperature ladder. Here, different states are exchanged between different temperatures. Hence, the algorithm does not get trapped under a single maximum and explores the distribution of interest.

Adaptive Direction Sampling (Gilks et al. (1994)) uses also a collection of chains to sample from the unknown target distribution while the proposed moves

take into account the local state of the chain. A special case of Adaptive Direction Sampling is the snooker algorithm. Assume that one of the components, called the current component is chosen to be moved. Then, another one named the anchor component is used to define the direction of the movement. The direction of the movement is through the line joining the current and anchor component following Figure 1.1. The candidate component is simulated from an

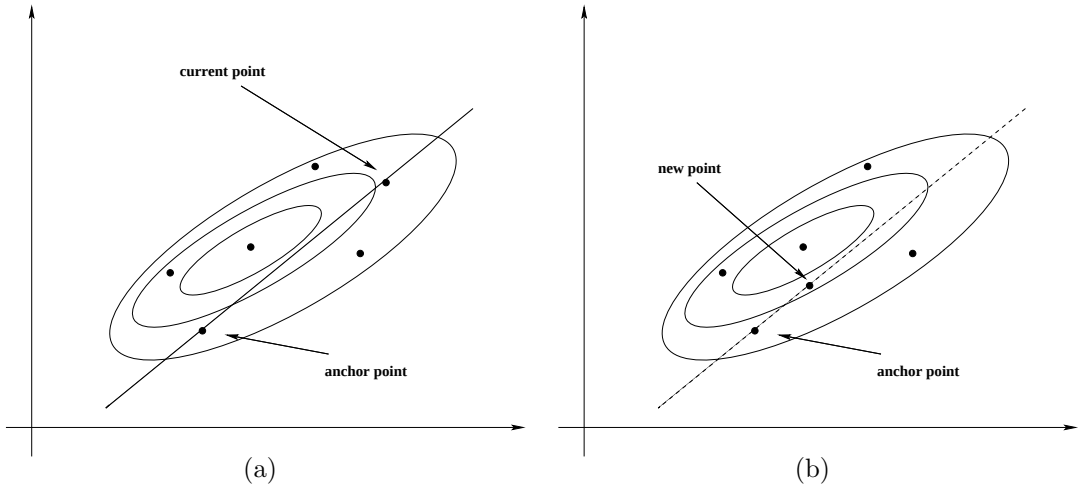


Figure 1.1: (a) A new point is sampled from an adjusted full-conditional distribution along the line which joins the current and the anchor point. (b) The proposed state.

adjusted full-conditional distribution along this line. If the process is repeated for sufficient time, the points will eventually traverse the whole sampling space and gather under high probability areas. The main disadvantage of Adaptive Direction Sampling is that it can not deal with discrete distributions.

Our intention is to design a new population MCMC algorithm, the simplex sampler, that is based on ideas of the Nelder and Mead (1965) simplex method. The simplex method is a deterministic optimisation algorithm which finds a local

maximum of a target function using three different type of moves, the so called reflection, expansion and contraction moves. Based on this algorithm, the simplex sampler consists of several chains that run in parallel and interact using three different stochastic instead of deterministic moves, the reflection, expansion and contraction moves as well as a Metropolis-Hastings update.

The simplex sampler samples efficiently from highly correlated target distributions. However, when the distribution of interest is multi-modal, the sampler does not sample with the correct frequency under the modes or it gets trapped under a local maximum unable to explore the whole sampling area. The problem is simply that the simplex method of the Nelder and Mead (1965) is an optimisation algorithm and for this reason, the simplex sampler tends to sample under a local maximum by design, unwilling to search for another one.

The tempered simplex sampler is designed to overcome this problem. The tempered simplex sampler is another new population MCMC sampler that aims to sample from multi-modal distributions. The tempered simplex sampler uses a population of chains that interact applying the simplex sampler and regarding a tempering ladder that allows sampling between several intermediate temperatures. Hence, the tempered simplex sampler uses the good mixing of the simplex sampler in cases of highly correlated distributions, while it also uses a tempering ladder that allows the sampler to explore the sampling area without getting trapped under a single maxima.

Model selection is a significant problem in statistics. It aims to find which of the $M_1, \ldots, M_N$ possible models describes best data $\boldsymbol{y}$, where each of the M_k possible models is defined by a parameter vector $\boldsymbol{\theta}_k$. From a Bayesian point of

view, the prior beliefs about the models are expressed through the prior distributions for each model such as $p(M_1), \ldots, p(M_N)$ and for the parameters within each of them through $p(\boldsymbol{\theta}_1|M_1), \ldots, p(\boldsymbol{\theta}_N|M_N)$. Then, using Bayes' formula, the posterior distribution of the parameters conditioned on model M_k is

$$\pi(\boldsymbol{\theta}_k|\boldsymbol{y}, M_k) = \frac{L(\boldsymbol{y}|\boldsymbol{\theta}_k, M_k)p(\boldsymbol{\theta}_k|M_k)}{\int\limits_{\boldsymbol{\theta}_k} L(\boldsymbol{y}|\boldsymbol{\theta}_k, M_k)p(\boldsymbol{\theta}_k|M_k)\mathrm{d}\boldsymbol{\theta}_k}$$

where the normalising constant is $p(\boldsymbol{y}|M_k) = \int_{\boldsymbol{\theta}_k} L(\boldsymbol{y}|\boldsymbol{\theta}_k, M_k)p(\boldsymbol{\theta}_k|M_k)\mathrm{d}\boldsymbol{\theta}_k$ and $L(\boldsymbol{y}|\boldsymbol{\theta}_k, M_k)$ is the likelihood of model M_k. Then, using Bayes' theorem the posterior model probability for each model M_k is

$$\pi(M_k|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|M_k)p(M_k)}{\sum\limits_{k=1}^{N} p(\boldsymbol{y}|M_k)p(M_k)}. \tag{1.1}$$

The posterior model probability requires knowledge of the marginal likelihood $p(\boldsymbol{y}|M_k)$. The marginal likelihood is the predictive probability of the data or, in other words, it is the predictive probability conditional on model M_k. It is not always feasible to achieve a precise estimation of the above integral as in most cases it is not analytically tractable. Several methods have been proposed in the literature to estimate the posterior model probabilities, see for example Green (1995), Carlin and Chib (1995), Godsill (2001), Sisson (2005).

In a Bayesian approach to the model selection problem, the Bayes factor is widely used since it provides evidence in favour of model M_k compared with model $M_{k'}$, (Jeffreys (1935), Jeffreys (1961), Kass and Raftery (1995), Lavine and Schervish (1999)). The Bayes factor is estimated as the ratio of the marginal

likelihood of model M_k to the marginal likelihood of model $M_{k'}$ so that

$$B_{k,k'} = \frac{p(\boldsymbol{y}|M_k)}{p(\boldsymbol{y}|M_{k'})}.$$

Using formula (1.1), the Bayes Factor can also be expressed as the ratio of the posterior model odds to the prior odds,

$$B_{k,k'} = \frac{\pi(M_k|\boldsymbol{y})}{\pi(M_{k'}|\boldsymbol{y})} \Big/ \frac{p(M_k)}{p(M_{k'})}.$$

In general, if $B_{k,k'} > 1$ then model M_k is supported more by the data than model $M_{k'}$. However, if $B_{k,k'} < 1$, model $M_{k'}$ is supported more by the data under consideration. Jeffreys (1961) grouped the interpretation of $B_{k,k'}$ according to the strength of evidence, see Table 1.1

$B_{k,k'}$	$\log_{10}(B_{k,k'})$	Evidence against H_0
1 to 3.2	0 to 1/2	Barely worth mentioning
3.2 to 10	1/2 to 1	Substantial
10 to 32	1 to 3/2	Strong
32 to 100	3/2 to 2	Very strong
> 100	> 2	Decisive

Table 1.1: Interpretation of $B_{k,k'}$ according to the strength of evidence against H_0, i.e model M_k is supported more by the data than model $M_{k'}$.

Meng and Wong (1996) try to evaluate the Bayes factors using the bridge sampling identity. Several other methods to estimate the Bayes factor using MCMC methods have also been proposed, see Meng and Wong (1996), Chen and Shao (1997b), Chen and Shao (1997a), Gelman and Meng (1998), Han and Carlin (2001), Meng and Schilling (2002), Green et al. (2003), Bartolucci et al. (2006),

Gramacy et al. (2009).

Moreover, a popular approximation to the posterior model probabilities is the exponentiated Bayesian information criterion, BIC (Kass and Raftery (1995), Kass and Wasserman (1995), Raftery (1996)). Schwarz (1978) proved that the BIC is a logarithmic approximation to the Bayes factor when model M_k is compared to the model under the null hypothesis. The BIC gives a penalty to models that include many parameters avoiding overfitting and it is independent of the choice of the prior distribution. Thus, the BIC for model M_k is estimated as

$$BIC_k = \log L(\boldsymbol{y}|\hat{\boldsymbol{\theta}}_k, M_k) - \frac{1}{2} d \log n$$

where $\log L(\boldsymbol{y}|\hat{\boldsymbol{\theta}}_k, M_k)$ is the maximum of the log likelihood, d is the dimension of model M_k and n is the sample size. Then, the posterior model probability is approximated by

$$\pi(M_k|\boldsymbol{y}) \propto \exp\{BIC_k\}.$$

Tierney and Kadane (1986) used the Laplace approximation to evaluate the marginal likelihood so that

$$p(\boldsymbol{y}|M_k) \approx (2\pi)^{d/2} |\mathbf{H}|^{1/2} p(\hat{\boldsymbol{\theta}}_k|M_k) L(\boldsymbol{y}|\hat{\boldsymbol{\theta}}_k, M_k)$$

where $\hat{\boldsymbol{\theta}}_k$ is the value of the parameter vector at which the intractable distribution takes its maximum value and $\mathbf{H}$ equals the negative of the inverse Hessian matrix of the intractable distribution at $\hat{\boldsymbol{\theta}}_k$.

Lewis and Raftery (1997) estimate the Bayes factors through posterior simulation using the Laplace-Metropolis estimator. Here, the Laplace method is used

to approximate the intractable marginal likelihood. However, it is not always easy to evaluate the Laplace parameter values analytically. In this case, they suggest to use the Metropolis-Hastings algorithm to estimate them.

Friel and Pettitt (2008) use ideas from path sampling to estimate the marginal likelihood. A power posterior distribution is defined to be proportional to the product of the prior and the likelihood raised to a power T where $T \in [0, 1]$ so that $\pi(\boldsymbol{\theta}_k|\boldsymbol{y}, M_k) \propto L\boldsymbol{y}|\boldsymbol{\theta}_k, M_k)^T p(\boldsymbol{\theta}_k)$. Then, a sample is drawn from the power posterior distribution and it is used to estimate the intractable marginal likelihood.

Reversible Jump Markov Chain Monte Carlo (Green (1995), Carlin and Chib (1995), Richardson and Green (1997), Godsill (2001), Green and Mira (2001), Sisson (2005)) is an extension of the Metropolis-Hastings algorithm and it can be used as another approach to the Bayesian model selection problem. It is used to explore the sampling space that consists of several models of different dimension. The algorithm performs in two steps. First, a fixed dimensional move allows exploration of the current model while a jumping move makes changes to dimensionality. Moreover, crucial to the performance of the algorithm is the choice of the proposal mechanism. Therefore, various methods have been proposed on how to choose the proposal mechanism of the RJMCMC algorithm.

Brooks et al. (2003) choose the proposal mechanism in an automatic way by optimising the acceptance probability based on the choice of the proposal distribution. Here, the aim is to choose the parameters of the proposal distribution so that every time a move is proposed to a new model, it will be accepted more often because various models will be visited more often and the mixing will be improved.

Brooks and Ehlers (2008) consider jumping proposal distributions for autoregressive time series models. They construct efficient sampling steps using the Brooks et al. (2003) method. It is also shown that applying the Brooks et al. (2003) method is similar to using the full-conditional posterior distribution of the parameters as the jump proposal distribution. Three different schemes are considered. In the first scheme, when a jump step is proposed to a new model, the proposed parameter vector is defined by adding or deleting parameter values from the current parameter vector to match the proposed model dimension. In the second scheme, some of the current parameter values are retained while the rest of them are updated together with the new ones. Finally, in the third scheme the whole proposed parameter vector is updated. They conclude that the last two schemes may give better efficient sample size but they are computationally more expensive since block updates are implemented. For this reason, the first scheme is concluded to be the best strategy.

Dellaportas et al. (2002) assume a Normal proposal distribution with mean and covariance matrix estimated in an off-line step. In the off-line step, a long Markov chain runs for the saturated model where all parameters are presented. In the on-line step, when a jump is proposed to a new model, the proposed parameter vector is generated from the full-conditional posterior distribution of the parameters. The mean and covariance matrix of the full-conditional posterior distribution are evaluated using the estimated mean and covariance matrix of the saturated model that has already been calculated in the off-line step.

In the case of a multi-modal target distribution when the proposals land away from the modes, they have almost no chance to be accepted. Furthermore, when the multi-modal target distribution is not analytically tractable, it is almost

impossible to design proposal steps that allow jumps between different modes. Al-Awadhi et al. (2004) overcome this problem by introducing a secondary Markov chain to modify the proposed moves. The basic idea is to use the RJMCMC algorithm to propose a new value and then move it closer to a mode before the accept/reject decision is taken. Here, irreducibility holds in each step.

Our target is to create an automatic RJMCMC algorithm so that no tuning is necessary and for this reason, the performance of the algorithm does not depend on the skills of the user, as well as to improve the proposal mechanism of the RJMCMC. In our analyses, the full-conditional posterior distribution of the parameters is used as the proposal distribution. Then, three different updating schemes are considered to generate the proposed parameter vector depending on how many of the common parameter values are conditioned on between the proposed and the current model. One might choose to condition on all, some or none of the common parameters. It turns out that the best strategy is when the proposed parameter vector is simulated conditioning on none of the common parameter values. In this case, the proposal distribution is the posterior one which is assumed to be tractable.

Unfortunately, the posterior distribution of the parameters is intractable in most cases. Hence, it has to be estimated quite fast and accurately before the proposed parameter vector is drawn from it. Therefore, stochastic methods may not be suitable to estimate it because that will be too computationally expensive and time consuming. For this reason, deterministic approximations like the Laplace approximation or the BIC are used to estimate the intractable posterior distribution of the parameters in an off-line step. Then, these estimates are used to approximate the posterior model probabilities in an also off-line step. Thus,

the approximated posterior model probabilities and the approximated posterior distribution of the parameters can be used as the proposal distribution inside the RJMCMC algorithm. The estimated posterior model probabilities can be used as an independent proposal to change the model dimension and the estimated posterior distribution of the parameters as a proposal distribution to draw the new parameter vector respectively.

The motivation is to improve the mixing of the RJMCMC algorithm by efficiently choosing the proposal distribution to make changes to model dimension. Hence, models with large posterior probability mass will be visited more frequently compared to models with smaller posterior probability mass. Thus, it is expected that the models are visited with the correct frequency. Moreover, it will be worthwhile to see whether the RJMCMC algorithm improves any bad approximations to the posterior model probabilities.

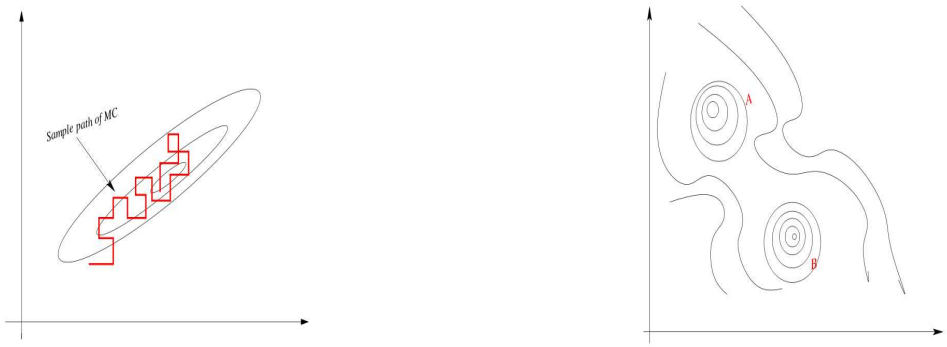
The future work involves the construction of a RJMCMC algorithm that consists of a selection of Markov chains defined on different dimensional models, following the work of Jasra et al. (2007a) and Jasra et al. (2007b). Hence, a population of several Markov chains could be used to choose the proposed model as well as to explore the current visited ones. The Markov chains have no memory and for this reason, one can not learn from the already rejected steps. Thus, if a population of Markov chains visits each of the possible models then, the proposed model may be chosen using that information.

Chapter 2

The simplex sampler

Usual Markov chain Monte Carlo (MCMC) methods like the Metropolis-Hastings algorithm (Hastings (1970)) and the Gibbs sampler (Geman and Geman (1984), Gelfand and Smith (1990)) use a single Markov chain to sample efficiently from the target distribution but there are a few cases where they lead to poor performances. If the variables are highly correlated then the methods may give highly correlated output and result in slow mixing of the Markov chain. Likewise, if the target distribution is multi-modal then the chain may get trapped under a single local maxima and consequently, be unable to traverse the whole sampling space, see Figure 2.1.

Moreover, the proposal mechanism has to be chosen carefully since it influences the performance of the algorithm. Suppose that the Metropolis-Hastings algorithm is used with a Gaussian proposal distribution. Then, the variance of the Gaussian distribution has to be tuned carefully as different values of the proposal variance may lead to different performances. If the proposal steps are not large enough then the Markov chain may get trapped under a local maxima or



(a) (b)

Figure 2.1: (a) In the Gibbs sampler case, the proposed moves are proposed in co-ordinate directions. If the variables are highly correlated, the mixing of the chain is slow. (b) In the case of a multi-modal target, it might not be easy to travel between the modes for instance, to move from area A to area B.

it may traverse slowly the sampling space resulting in slow mixing. Conversely, if the proposal steps are too large, the proposals are likely to land in regions of lower probability mass. In this case, the ratio of the target probability at the proposed configuration to the target probability at the current configuration will be very small and for this reason, the rejection probability will be too high. Thus, the chain may remain at a particular area for a long time resulting in slow mixing and give a high correlated sample. Furthermore, the algorithm performs better if the proposal density matches the shape of the target distribution but in most cases, this is impossible since the distribution of interest is not known.

Population MCMC methods use more than one Markov chain to explore the

sampling space. The chains run in parallel and interact with each other in various ways (Gilks et al. (1994), Neal (1996), Liang and Wong (2001)). Hence, a population of chains explores the current state and defines the proposed one.

Our aim is to introduce a new population MCMC sampler, the simplex sampler, designed to overcome the problem of highly correlated and multi-modal target distributions. The simplex sampler uses a proposal mechanism similar to the deterministic simplex method of Nelder and Mead (1965) for function optimisation. The different type of moves are now stochastic using an accept/reject mechanism to correct the arbitrary distribution. The sampler consists of three different updating moves, the so called reflection, expansion and contraction moves as well as a Metropolis-Hastings update. The idea is simple and easy to implement. In designed moves, detailed balance is preserved at each step in such a way that there is always a positive probability that enables the chain to move from the proposed state back to the current one and ensures convergence to the stationary distribution.

The work which is described in this chapter was carried out independently of Strens et al. (2002). Strens et al. (2002) also introduced a population MCMC sampler based on the Nelder and Mead (1965) simplex method. Our simplex sampler is more developed, includes more move types and it is tested more rigorously on challenging examples.

In the following text, the simplex method of Nelder and Mead (1965) is presented together with some of the existing techniques. Then, the proposal mechanism of the simplex sampler is introduced. The sampler is also tested on different examples and its performance is compared to the performance of some of the existing techniques. Finally, a discussion is included outlining the advantages and

disadvantages of the sampler.

2.1 Simplex method for function minimisation

Nelder and Mead (1965) introduced a deterministic minimization algorithm, the simplex method. The simplex method is designed to find a local minimum of an n dimensional function. Hence, the simplex is defined as a polytope of $n+1$ vertices in n dimensional space. For instance, the simplex is a line segment in one dimension, a triangle in two dimensions, a tetrahedron in three dimensions, etc. The local minimum is reached by comparing the function values of each of the vertices of the simplex and replacing the vertex with the highest function value by another one with smaller function value. The simplex method achieves the minimum by implementing three different types of move, the so called reflection, expansion and contraction moves.

Suppose that one is interested in the minimum of an n-dimensional function. Then, the current simplex is defined as $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n+1}\}$ where each $\boldsymbol{x}_i \in \Re^n$, for $i = 1, \ldots, n+1$. Let $\boldsymbol{y}_i$ denote the function value at $\boldsymbol{x}_i$ and let $\boldsymbol{y}_h$ be the maximum value of all function values of the vertices of the simplex so that $\boldsymbol{y}_h = \max_i \boldsymbol{y}_i$. Likewise, let $\boldsymbol{y}_\ell$ be the minimum value of all function values so that $\boldsymbol{y}_\ell = \min_i \boldsymbol{y}_i$.

Every time, the vertex with the highest function value, say $\boldsymbol{x}_c$, is chosen to be reflected through the centroid of the simplex according to the following equation

$$\boldsymbol{x}^* = (1+\alpha)\bar{\boldsymbol{x}} - \alpha \boldsymbol{x}_c$$

where $\boldsymbol{x}^*$ is the reflected vertex, α is a positive constant which is called the

reflection coefficient and $\bar{\boldsymbol{x}}$ is the centroid of the simplex. If a two dimensional function is assumed then the reflected point is on the line joining the point with the highest target function value and the centroid of the simplex on the far side of the centroid from $\boldsymbol{x}_c$, see Figure 2.2. Notice that the distance from $\boldsymbol{x}^*$ to the centroid is equal to α times the distance from the centroid to the current vertex $\boldsymbol{x}_c$, that is $|\boldsymbol{x}^* - \bar{\boldsymbol{x}}| = \alpha|\bar{\boldsymbol{x}} - \boldsymbol{x}_c|$.

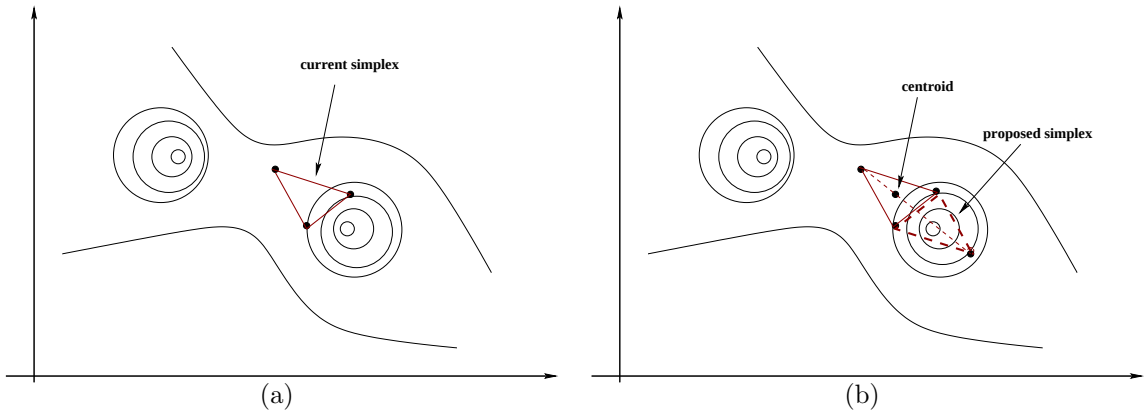


Figure 2.2: (a) Simplex in the current state. The point with the lower target probability is going to be moved. (b) Move the point through the line joining the current point and the centroid in a deterministic way.

If the reflection has produced a vertex with smaller function value than $\boldsymbol{y}_h$ then $\boldsymbol{x}_c$ is replaced by $\boldsymbol{x}^*$ and the process is started again with the new simplex. If a new minimum has been produced then $\boldsymbol{x}^*$ is expanded to $\boldsymbol{x}^{**}$ according to the following equation

$$\boldsymbol{x}^{**} = \gamma\boldsymbol{x}^* + (1-\gamma)\bar{\boldsymbol{x}}$$

where γ is the expansion coefficient and it should be greater than one. The expansion coefficient determines how far away from the centroid, $\boldsymbol{x}^*$ is expanded

to $\boldsymbol{x}^{**}$. The distance from $\boldsymbol{x}^{**}$ to $\bar{\boldsymbol{x}}$ is equal to γ times the distance from $\boldsymbol{x}^{*}$ to $\bar{\boldsymbol{x}}$, that is $|\boldsymbol{x}^{**} - \bar{\boldsymbol{x}}| = \gamma|\boldsymbol{x}^{*} - \bar{\boldsymbol{x}}|$.

If $\boldsymbol{x}^{**}$ has smaller target value than the maximum one then $\boldsymbol{x}_c$ is replaced by $\boldsymbol{x}^{**}$. Otherwise, the expansion move has failed so $\boldsymbol{x}_c$ is replaced by $\boldsymbol{x}^{*}$. If a new maximum is produced then a new vertex, say $\boldsymbol{x}'_c$, is defined to be either the old one, $\boldsymbol{x}_c$, or the reflected, $\boldsymbol{x}^{*}$, depending on which of the two has the lower function value. Then, it is expanded to $\boldsymbol{x}^{***}$ according to the following formula

$$\boldsymbol{x}^{***} = \beta\boldsymbol{x}'_c + (1-\beta)\bar{\boldsymbol{x}}$$

where β is the contraction coefficient and takes values between 0 and 1. The contraction coefficient β also defines how far way from $\boldsymbol{x}'_c$, $\boldsymbol{x}^{***}$ is going to be. The distance from $\boldsymbol{x}^{***}$ to $\bar{\boldsymbol{x}}$ is equal to β times the distance from $\boldsymbol{x}'_c$ to $\bar{\boldsymbol{x}}$, that is $|\boldsymbol{x}^{***} - \bar{\boldsymbol{x}}| = \beta|\boldsymbol{x}'_c - \bar{\boldsymbol{x}}|$. The contracted $\boldsymbol{x}^{***}$ replaces $\boldsymbol{x}'_c$ and the process restarts again unless the constructed $\boldsymbol{x}^{***}$ is worse than the better of $\boldsymbol{x}'_c$ and $\boldsymbol{x}^{*}$. In this case, each point is transformed as $(\boldsymbol{x}_i + \boldsymbol{x}_\ell)/2$ and then, the process is restarted. The procedure is stopped when a local minimum is achieved, for instance, when the points are too close to each other or when the standard error of the function is less than a predefined small value.

Here, the values of α, β and γ are defined at the beginning of the algorithm. Standard value for the reflection coefficient α is 1, for the contraction coefficient, β, is 1/2 and for the expansion coefficient, γ, is 2.

2.2 Markov chain Monte Carlo techniques

2.2.1 Metropolis-Hastings algorithm

The idea of Markov chain Monte Carlo (MCMC) was first introduced by Metropolis et al. (1953) as a simulation method of energy levels of atoms in a crystalline structure. Hastings (1970) was the first to apply the idea to statistical problems. Since then, MCMC methods have been widely used to integrate complex and high dimensional functions as well as to sample from complex and highly structured real life models.

Assume that $\pi(\cdot)$ is a complex and high dimensional distribution so that it is not possible to simulate directly from it. For this reason, the proposal distribution $q(\cdot,\cdot)$ is used to draw a sample from $\pi(\cdot)$ using an accept/reject mechanism. Simulation from $q(\cdot,\cdot)$ should be possible and straightforward. Thus, the proposal distribution $q(\boldsymbol{x}^t, \boldsymbol{y})$ is the probability of moving from the current state, $\boldsymbol{x}^t$, to the proposed one, $\boldsymbol{y}$, at time t where $\boldsymbol{x} \in \mathbb{R}^n$. At time t, a candidate component $\boldsymbol{y}$ is simulated from $q(\boldsymbol{x}^t, \boldsymbol{y})$ and it is accepted with probability

$$\alpha(\boldsymbol{x}^t, \boldsymbol{y}) = \min\left\{1, \frac{\pi(\boldsymbol{y})}{\pi(\boldsymbol{x}^t)}\frac{q(\boldsymbol{y}, \boldsymbol{x}^t)}{q(\boldsymbol{x}^t, \boldsymbol{y})}\right\}. \tag{2.1}$$

If the candidate component is accepted with probability given in (2.1), the next state of the chain becomes $\boldsymbol{x}^{t+1} = \boldsymbol{y}$. Otherwise, the candidate component is rejected and the Markov chain stays where it currently is so that $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$. If the Markov chain is run long enough, i.e the chain has reached the stationary distribution, realisations of the chain can be regarded as a dependent sample

from the posterior distribution $\pi(\cdot)$. The algorithmic version of the Metropolis-Hastings method can be seen in Algorithm 1.

The choice of the proposal distribution is crucial for the performance of the algorithm since it explores the intractable distribution and it should ensure that irreducibility holds at each step. The mixing of the chain also depends on the proposal distribution. If the proposed move is accepted more often then the chain results in fast mixing otherwise, the chain results in slow mixing. For instance, if a Normal proposal distribution is used then the variance of the Normal distribution must be carefully tuned. If the proposed steps are too large, it is possible that the chain may remain stuck at a particular area for a long time since the proposed moves may be rejected and the chain results in slow mixing. Then again, if the proposed steps are too small, the chain tends to make small changes and for this reason, it moves inefficiently. In practice, the proposal variance is chosen on the basis of a pilot run. One way is to tune the algorithm according to the acceptance probability since acceptance probability values between 20% and 40% imply that the algorithm has been tuned efficiently (Gelman et al. (1996)).

In general, the algorithm may result in poor performance when the parameters are highly correlated since it may give a highly correlated output. In the case of a multi-modal target distribution, the chain may get trapped under a mode and for this reason, it does not traverse the whole sampling space especially, when the modes are well separated from each other.

Algorithm 1 Metropolis - Hastings algorithm

At time t, the current state of the chain is $\boldsymbol{x}^t$.

1. Propose the new state of the chain $\boldsymbol{y}$ such as $\boldsymbol{y} \sim q(\boldsymbol{x}^t, \boldsymbol{y})$.
2. Calculate the acceptance probability $\alpha(\boldsymbol{x}^t, \boldsymbol{y})$.
   - Accept $\boldsymbol{y}$ with probability $\alpha(\boldsymbol{x}^t, \boldsymbol{y})$ given in (2.1) and set $\boldsymbol{x}^{t+1} = \boldsymbol{y}$ or
   - reject $\boldsymbol{y}$ with probability $1 - \alpha(\boldsymbol{x}^t, \boldsymbol{y})$ and set $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$.
3. Return to step 1.

2.2.2 Gibbs sampler

The Gibbs sampler was introduced by Geman and Geman (1984). It is a special case of the single-component Metropolis-Hastings algorithm (Hastings (1970)). Suppose that $\pi(\cdot)$ is the intractable target distribution and at the current state t, the chain is $\boldsymbol{x}^t = (x_1^t, \ldots, x_n^t)$. Thus, at time $t+1$ the i^{th} component is simulated from the full-conditional distribution $\pi(x_i^{t+1} | x_1^{t+1}, \ldots, x_{i-1}^{t+1}, x_{i+1}^t, \ldots, x_n^t)$. If the algorithm runs for a long time the simulated values $\{(x_1^i, \ldots, x_n^i)\}_{i=k+1}^N$ are considered to be a representative sample from the target distribution $\pi(\cdot)$, where k is the burn in and N is the total number of iterations. The algorithmic version of the Gibbs sampler can be seen in Algorithm 2.

The Gibbs sampler is typically used when the posterior conditional distribution is of standard form, and hence generally easy to sample from. In general, the sampler is easy to implement and most of the time, it performs quite well in practice. However, there are cases where the performance of the algorithm is not so good. For instance, if the components are highly correlated, the algorithm converges slowly and gives a highly correlated output.

Algorithm 2 Gibbs sampler

At time t, the current state of the chain is $\boldsymbol{x}^t = (x_1^t, \ldots, x_n^t)$.

1. Simulate x_1^{t+1} from the conditional distribution $\pi(x_1^{t+1}|x_2^t, \ldots, x_n^t)$.
2. Simulate x_2^{t+1} from the conditional distribution $\pi(x_2^{t+1}|x_1^{t+1}, x_2^t, \ldots, x_n^t)$.
   $\vdots$
3. Simulate x_i^{t+1} from the conditional distribution $\pi(x_i^{t+1}|x_1^{t+1}, \ldots, x_{i-1}^{t+1}, x_{i+1}^t, \ldots, x_n^t)$.
   $\vdots$
4. Simulate x_n^{t+1} from the conditional distribution $\pi(x_n^{t+1}|x_1^{t+1}, \ldots, x_{n-1}^{t+1})$.
5. Return to step 1.

2.2.3 Population Markov chain Monte Carlo

Usual MCMC methods simulate a single Markov chain $\boldsymbol{x}_i$ with stationary distribution exactly equal to the posterior distribution of interest $\pi_i(\cdot)$ on space $\mathcal{X}$. The general idea in population MCMC methods is to use a collection of chains $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ instead of a single one $\boldsymbol{x}_i$ that can be trapped under a single mode, depending on the starting position of the chain. Here, the target distribution is

$$\pi(\boldsymbol{x}) = \prod_{i=1}^{n} \pi_i(\boldsymbol{x}_i) \tag{2.2}$$

on space $\mathcal{X}^n$.

Hence, at the current state, a population of different chains $\boldsymbol{x}$ is used to propose a new one, $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$. One can either accept each of the proposed components, $\boldsymbol{y}_i$, individually or the whole collection of them, $\boldsymbol{y}$, globally. The

latter approach suffers from the problem of dimensionality. Therefore, it is recommended to propose a new component, $\boldsymbol{y}_i$, at each time because the average acceptance probability decreases with the number of components.

2.2.4 Real-Parameter Evolutionary Monte Carlo

The Real-parameter Evolutionary Monte Carlo method of Liang and Wong (2001) is a population MCMC based algorithm. The method borrows the learning ability of genetic algorithms and the fast mixing of parallel tempering. At time t, the current state consists of a collection $\boldsymbol{x}^t = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$, with $\boldsymbol{x}_i \in \mathbb{R}^d$ so that $\boldsymbol{x}_i = (\boldsymbol{\beta}_1^i, \ldots, \boldsymbol{\beta}_d^i)$ and a temperature ladder $T = (T_1, \ldots, T_n)$, with $T_1 = 0 \leq T_2 \leq \ldots \leq T_n = 1$. The distribution of interest is defined by replacing $\pi_i(\boldsymbol{x}_i) = \pi^{T_i}(\boldsymbol{x}_i)$ in (2.2) so that

$$\pi(\boldsymbol{x}^t) = \prod_{i=1}^{n} \pi(\boldsymbol{x}_i)^{T_i}$$

Thus, when the temperature value T_i is close to zero then the distribution $\pi(\cdot)^{T_i}$ is similar to the uniform one while when T_i takes the value one, the tempered distribution, $\pi(\cdot)^{T_i}$, is the actual target one. The rest of the temperature values correspond to different tempered distributions. The main idea is that movement in tempered distributions is easier than movement in the target distribution. Therefore, proposing to swap values from tempered to target distributions should promote mixing.

Three different types of move are implemented, the so called mutation, crossover and exchange operators. In the mutation operator, one of the components $\boldsymbol{x}_i$ is chosen at the current state and a new one $\boldsymbol{y}_i$ is generated under the same tempered distribution $\pi(\cdot)^{T_i}$ by adding a random vector $\mathbf{e}_i$ to the current one such

as $\boldsymbol{y}_i = \boldsymbol{x}_i + \mathbf{e}_i$. Then, the proposed move is accepted with probability

$$\alpha_m = \min\left\{1, \frac{\pi(\boldsymbol{y}_i)^{T_i}}{\pi(\boldsymbol{x}_i)^{T_i}}\right\}$$

and the new state of the Markov chain is $\boldsymbol{x}^{t+1} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \boldsymbol{y}_i, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n)$ or it is rejected with probability $1 - \alpha_m$ and the chain stays where it currently is, so that $\boldsymbol{x}^{t+1} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$.

In the crossover operator, a pair of components is randomly selected from the current population and two candidate components are generated by implementing a real crossover or a snooker crossover move. The real crossover operator chooses one component $\boldsymbol{x}_i = (\boldsymbol{\beta}_1^i, \ldots, \boldsymbol{\beta}_d^i)$ under temperature T_i and another one $\boldsymbol{x}_j = (\boldsymbol{\beta}_1^j, \ldots, \boldsymbol{\beta}_d^j)$ under temperature T_j to propose two new candidate components in the following way. First, a crossover point c is chosen from $\{1, \ldots, d\}$ and then, two new candidate components are generated as

$$\boldsymbol{y}_i = (\boldsymbol{\beta}_1^i, \ldots, \boldsymbol{\beta}_{c-1}^i, \boldsymbol{\beta}_c^j, \ldots, \boldsymbol{\beta}_d^j)$$

and

$$\boldsymbol{y}_j = (\boldsymbol{\beta}_1^j, \ldots, \boldsymbol{\beta}_{c-1}^j, \boldsymbol{\beta}_c^i, \ldots, \boldsymbol{\beta}_d^i).$$

The proposed move is accepted with probability

$$\alpha_c = \min\left\{1, \frac{\pi(\boldsymbol{y}_i)^{T_i}}{\pi(\boldsymbol{x}_i)^{T_i}} \frac{\pi(\boldsymbol{y}_j)^{T_j}}{\pi(\boldsymbol{x}_j)^{T_j}}\right\}$$

otherwise, it is rejected and the chain stays where it currently is.

The snooker crossover operator is based on ideas from the snooker sampler

of Gilks et al. (1994). Here, the component $\boldsymbol{x}_i$ is randomly chosen from the current population under temperature T_i and the other component $\boldsymbol{x}_j$ is chosen either randomly or with probability proportional to its Boltzmann weight under temperature T_j. Thus, the candidate component is generated by first calculating $\mathbf{e} = \frac{\boldsymbol{x}_j - \boldsymbol{x}_i}{|\boldsymbol{x}_j - \boldsymbol{x}_i|}$ and then setting $\boldsymbol{y}_i = \boldsymbol{x}_i + r\mathbf{e}$ where the random variable $r \in (-\infty, +\infty)$ and it is sampled from the density

$$f(r) \propto \mid r\mid^{d-1} \pi(\boldsymbol{y}_i).$$

Notice that the density $f(r)$ is usually not known in closed form. In this case, standard MCMC methods like Metropolis-Hastings algorithm can be used to draw the r value from it.

The proposed move is accepted or rejected with the usual Metropolis rule. The algorithm proceeds by applying the mutation operator with probability p_m, the real crossover with probability $(1 - p_m)/2$ and the snooker crossover with probability $(1 - p_m)/2$. The probability p_m is called the mutation rate.

The exchange operator swaps points under different temperatures. Suppose that the current state is defined as $\boldsymbol{x}^t = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_j, \ldots, \boldsymbol{x}_n)$ and that component $\boldsymbol{x}_i$ is chosen under temperature T_i to be exchanged with another component $\boldsymbol{x}_j$ under temperature T_j. Thus, the components $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are exchanged by randomly selecting i from $1, \ldots, n$ and then setting $j = i \pm 1$ with probability $p(j = i + 1) = p(j = i - 1) = 0.5$ for $i = 2, \ldots, n - 1$, $p(j = 2) = 1$ for $i = 1$ and $p(j = n - 1) = 1$ for $i = n$. Then, the proposed configuration

$\boldsymbol{y} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_n)$ is accepted with probability

$$\alpha_e = \min\left\{1, \frac{\pi(\boldsymbol{x}_j)^{T_i}}{\pi(\boldsymbol{x}_i)^{T_i}} \frac{\pi(\boldsymbol{x}_i)^{T_j}}{\pi(\boldsymbol{x}_j)^{T_j}}\right\}$$

or equivalently,

$$\alpha_e = \min\left\{1, \left(\frac{\pi(\boldsymbol{x}_i)}{\pi(\boldsymbol{x}_j)}\right)^{T_j - T_i}\right\}.$$

Otherwise, the proposed configuration is rejected and the chain stays where it currently is, so that $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$. Notice that when the different temperature values are close then the difference $T_j - T_i$ tends to zero. Thus, the acceptance probability for the exchange move, α_e, is almost equal to one. For this reason, the proposed move tends to be accepted more often.

The Real-parameter Evolutionary Monte Carlo method performs very well most of the time even though there are a lot of parameters that have to be tuned depending on the skills of the user. The tuning also influences the mixing of the algorithm. The number of the intermediate temperatures has to be defined and the way to draw the r value. Another issue is that different tempered distributions are used to sample from the distribution of interest but only the sample under temperature one which corresponds to the target distribution, is stored. The method is quite complex to implement and for this reason, it requires a skillful user.

2.2.5 Tempered Transitions Method

Neal (1996) suggests to sample from multimodal distributions using the tempered transitions method. The method is based on simulated tempering, (Geyer and

Thompson (1995)) and uses $\pi^1(\cdot), \ldots, \pi^n(\cdot)$ distributions in order to sample from the distribution of interest $\pi(\cdot)$. A transition probability $\hat{Q}_i(\boldsymbol{x}^t, \boldsymbol{x}^{t+1})$ is used to move from the current state $\boldsymbol{x}^t$ to the proposed one $\boldsymbol{x}^{t+1}$ and $\check{Q}_i(\boldsymbol{x}^{t+1}, \boldsymbol{x}^t)$ is used to move from the proposed state $\boldsymbol{x}^{t+1}$ to the current one $\boldsymbol{x}^t$so that detailed balance is presented at each step, that is

$$\pi(\boldsymbol{x}^t)\hat{Q}_i(\boldsymbol{x}^t, \boldsymbol{x}^{t+1}) = \pi(\boldsymbol{x}^{t+1})\check{Q}_i(\boldsymbol{x}^{t+1}, \boldsymbol{x}^t).$$

Notice that if $\hat{Q}_i$ consists of several sub-transitions such as $\hat{Q}_i = S_1 \cdots S_k$ then $\check{Q}_i = S_k \cdots S_1$. The method finds a candidate state by applying the transition probabilities in turn, i.e $\hat{Q}_1 \ldots \hat{Q}_n \check{Q}_n \ldots \check{Q}_1$. Then, the candidate state is accepted or rejected based on the intermediate acceptance probabilities.

Assume that the current state is $\boldsymbol{x}^t = \hat{x}_0$ then, the candidate state is $\check{x}_0$ estimated as follows

Generate $\hat{x}_1$ from $\hat{x}_0$ using $\hat{Q}_1$

Generate $\hat{x}_2$ from $\hat{x}_1$ using $\hat{Q}_2$

$\ldots$

Generate $\bar{x}_n$ from $\hat{x}_{n-1}$ using $\hat{Q}_n$

Generate $\check{x}_{n-1}$ from $\bar{x}_n$ using $\check{Q}_n$

$\ldots$

Generate $\check{x}_1$ from $\check{x}_2$ using $\check{Q}_2$

Generate $\check{x}_0$ from $\check{x}_1$ using $\check{Q}_1$

The candidate state $\check{x}_0$ is accepted with probability

$$\alpha_e = \min\left\{1, \prod_{i=1}^{n-1} \frac{\pi^{i+1}(\hat{x}_i)}{\pi^i(\hat{x}_i)} \frac{\pi^i(\check{x}_i)}{\pi^{i+1}(\check{x}_i)}\right\}$$

and the next state of the chain is $\boldsymbol{x}^{t+1} = \check{x}_0$ otherwise, it is rejected and the next state of the chain is $\boldsymbol{x}^{t+1} = \hat{x}_0$. It should be mentioned that a crucial choice for the performance of the algorithm is the number of intermediate distributions to be used and how these are defined.

2.3 Proposal mechanism of the simplex sampler

The proposal mechanism of the simplex sampler consists of three different types of move, the reflection, expansion and contraction moves. The reflection move is the core of the simplex sampler while the expansion and contraction moves tend to keep the shape of the simplex stable so that the algorithm does not lose its local ability. A Metropolis-Hastings move is also used to update the vertices of the simplex. All designed moves promote good mixing. At each sweep, each of the designed moves can be applied in turn or a kernel of them can be used. The algorithmic version of the sampler is presented in Algorithm 3.

2.3.1 Reflection move

Suppose that at the current state, the simplex is defined as $\boldsymbol{x}^t = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$. The proposal mechanism of the simplex sampler proceeds by first choosing one of the components of the simplex with some probability. The chosen component or the current component, $\boldsymbol{x}_c$, tends to be the one with the smallest target probability

compared to the rest of the population. The current component is chosen with discrete probability density function $\ell_t(i)$, for $i = 1, \ldots, n$ with

$$\ell_t(i) \propto 1 - \frac{\pi(\boldsymbol{x}_i)}{\sum_{i=1}^{n} \pi(\boldsymbol{x}_i)}$$

so that

$$\ell_t(i) = \frac{P(\boldsymbol{x}_i)}{\sum_{j=1}^{n} P(\boldsymbol{x}_j)} \tag{2.3}$$

where $P(\boldsymbol{x}_i)$ is defined as

$$P(\boldsymbol{x}_i) = 1 - \frac{\pi(\boldsymbol{x}_i)}{\sum_{j=1}^{n} \pi(\boldsymbol{x}_j)}.$$

Then, the candidate component, $\boldsymbol{y}$, is reflected through the line joining the centroid of the current population, $\bar{\boldsymbol{x}}^t$, and the current component, $\boldsymbol{x}_c$, see Figure 2.3, so that

$$\boldsymbol{y} = (1 + \alpha)\, \bar{\boldsymbol{x}}^t - \alpha \boldsymbol{x}_c \tag{2.4}$$

where α is the reflection coefficient which determines how far away from the centroid the new candidate component is.

The absolute value of the reflection coefficient, $|\alpha|$, is defined by equation 2.4. The sign of alpha is positive when the $\boldsymbol{x}_c$ and $\boldsymbol{y}$ are on different sides of $\bar{x}_t$ while it is negative when they are on the same side of $\bar{x}_t$. When $\alpha = 0$ then $\boldsymbol{y} = \bar{\boldsymbol{x}}^t$, i.e they are on either side of $\bar{\boldsymbol{x}}^t$.

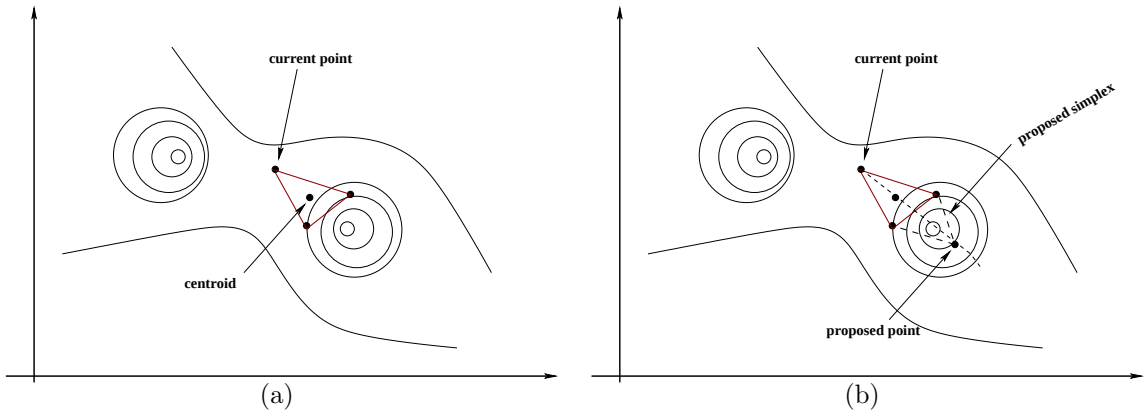


Figure 2.3: (a) Simplex in the current state. The centroid is estimated and the current point is chosen. (b) A new point is reflected through the line joining the current point and the centroid of the simplex. The proposed simplex will be accepted or rejected with the usual Metropolis-Hastings rule.

Hence, different α values determine different candidate components. According to equation (2.4), if $\alpha = 0$ then $\boldsymbol{y} = \bar{\boldsymbol{x}}^t$ meaning that the candidate component is the centroid of the simplex at the current state. Likewise, if $\alpha = -1$ then $\boldsymbol{y} = \boldsymbol{x}_c$ implying that the candidate component is the current one. In addition, if $\alpha > 1$, the candidate component is proposed uphill of the simplex while if $\alpha < -1$ then it is proposed downhill of the simplex, see Figure 2.4.

The simplex method of Nelder and Mead (1965) is a local optimisation algorithm. For this reason, the components of the simplex tend to be proposed uphill most of the time. Thus, the sampler may get trapped under a local maxima unable to traverse the whole sampling surface. Therefore, α values less than -1 ensure that the sampler proposes candidate components downhill and explores the whole sampling space without getting trapped under a local maxima. Notice also that when $\alpha = 0.5$, the proposed component is on the line (or Hyper plane)

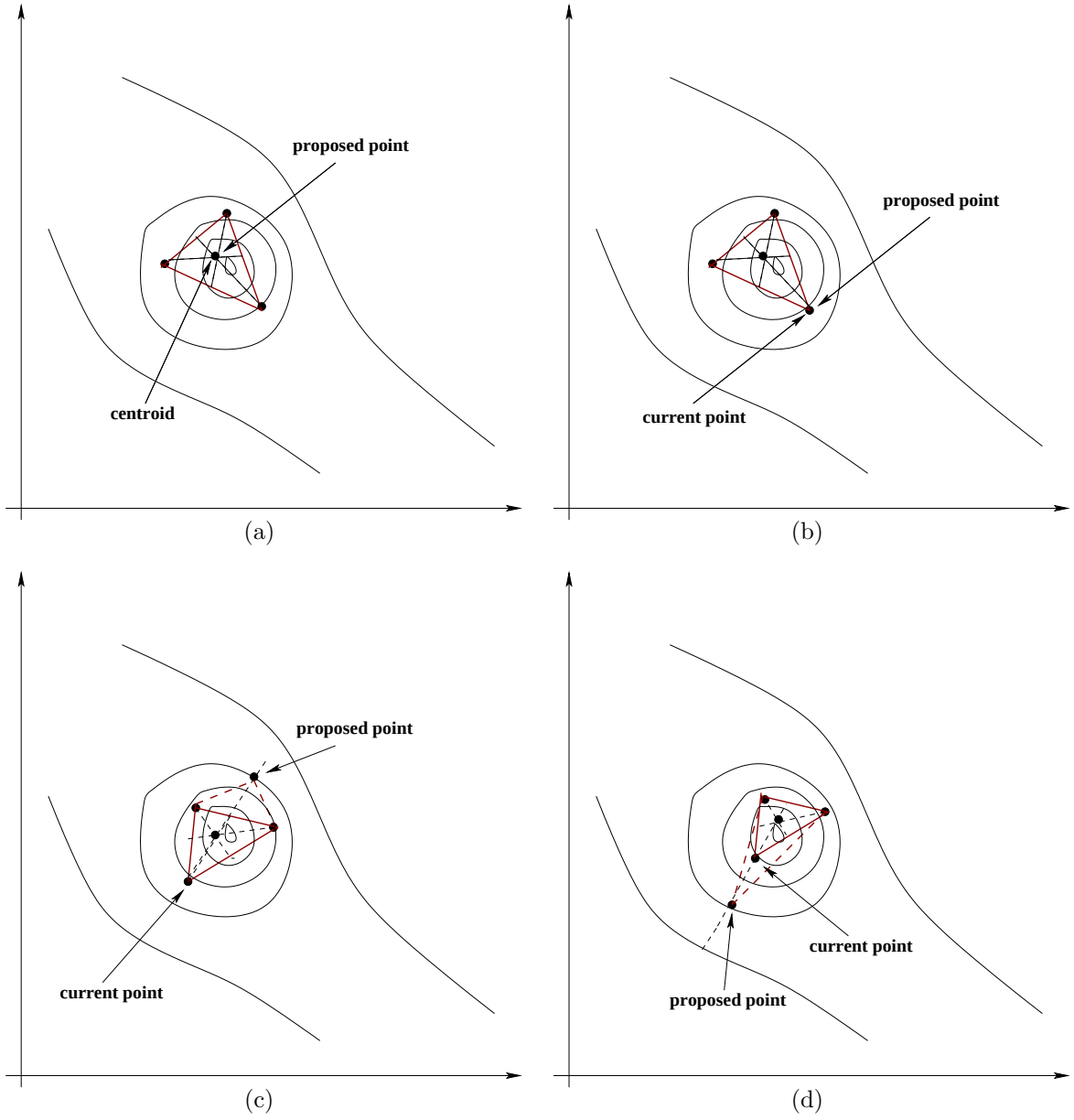


Figure 2.4: (a) For $\alpha = 0$ the proposed component is the centroid. (b) For $\alpha = -1$ the proposed component is the current component. (c) For $\alpha > 1$ the candidate component is proposed uphill of the simplex. (d) For $\alpha < -1$ the candidate component is proposed downhill of the simplex

joining the other vertices of the simplex. For this reason, this value should be avoided as the sampler may collapse if the simplex consists of only three components.

The proposal mechanism of the simplex sampler uses the current configuration of the Markov chain to propose the new one. Then, the proposed component $\boldsymbol{y}$ is accepted with probability

$$\alpha_{ref}\left(\boldsymbol{x}_c, \boldsymbol{y}\right) = \min\left\{1, \frac{\pi(\boldsymbol{y})}{\pi(\boldsymbol{x}_c)} \frac{q(\boldsymbol{y}, \boldsymbol{x}_c)}{q(\boldsymbol{x}_c, \boldsymbol{y})}\right\} \tag{2.5}$$

where $q(\cdot, \cdot)$ is the proposal distribution. The proposal distribution $q(\boldsymbol{x}_c, \boldsymbol{y})$ is defined as the probability of choosing $\boldsymbol{x}_c^t$ as the current component and proposing $\boldsymbol{y}$ to be the candidate one. Hence, $q(\boldsymbol{x}_c, \boldsymbol{y}) = \ell_t(c)p_\alpha(\alpha)$ where $p_\alpha(\alpha)$ is the probability of choosing the value α for the reflection coefficient from the $p_\alpha(\cdot)$ distribution. A good choice of the proposal distribution to choose the reflection coefficient, α, value is important for the mixing of the algorithm. If the proposed component is accepted more often, i.e it is proposed to a higher probability area, then the mixing of the chain is better and the target distribution is explored faster. Experimentation showed that a $N(1, 2^2)$ is a good choice of the proposal distribution to draw the reflection coefficient value.

If the candidate component is accepted with probability $\alpha_{ref}\left(\boldsymbol{x}_c, \boldsymbol{y}\right)$ then the next state of the Markov chain is

$$\boldsymbol{x}^{t+1} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{c-1}, \boldsymbol{y}, \boldsymbol{x}_{c+1}, \ldots, \boldsymbol{x}_n\}.$$

Otherwise, the chain stays where it currently is so that $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$. The procedure

is repeated for sufficient time to ensure convergence to the stationary distribution.

In all designed moves, detailed balance is preserved at each step. Thus, there is always a positive probability that enables the proposed component to move back from the proposed state to the current one and ensures convergence to the stationary distribution. For this reason, the proposal distribution $q(\boldsymbol{y}, \boldsymbol{x}_c)$ is defined as $q(\boldsymbol{y}, \boldsymbol{x}_c) = \ell_{t+1}(c)p_\alpha(\alpha^*)$ where $\ell_{t+1}(c)$ is the probability of choosing the candidate component $\boldsymbol{y}$ as the one to be reflected and $p_\alpha(\alpha^*)$ is the probability of generating the α^* value from $p_\alpha(\cdot)$. The reflection coefficient for the reverse move, α^*, ensures that the chain moves from the proposed state back to the current one. Thus, irreducibility holds at each step of the reflection move and for this reason, the chain runs forward and backward in time.

The reflection coefficient α^* is estimated according to the following formula

$$\begin{aligned} \boldsymbol{x}_c &= (1+\alpha^*)\bar{\boldsymbol{x}}^{t+1} - \alpha^*\boldsymbol{y} \\ \boldsymbol{x}_c &= \bar{\boldsymbol{x}}^{t+1} + \alpha^*(\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}) \\ (\boldsymbol{x}_c - \bar{\boldsymbol{x}}^{t+1}) &= \alpha^*(\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}) \end{aligned}$$

so that

$$|\alpha^*| = \frac{||\boldsymbol{x}_c - \bar{\boldsymbol{x}}^{t+1}||}{||\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}||}$$

where $||\boldsymbol{x}_c - \bar{\boldsymbol{x}}^{t+1}||$ is the euclidean distance between the current component, $\boldsymbol{x}_c$, and the centroid of the simplex at the proposed state, $\bar{\boldsymbol{x}}^{t+1}$. The sign of α^* is defined so that detailed balanced holds and the Markov chain runs forward and

backward in time at each step.

2.3.2 Expansion and Contraction moves

The vertices of the simplex may be far apart or too close to each other. In this case, the simplex sampler loses its local ability and performs poorly. For this reason, two additional moves are introduced, the expansion and contraction moves to keep the shape of the simplex stable. In the contraction and expansion moves each of the components of the simplex is updated according to the following formula

$$\boldsymbol{y}_i = \beta(\boldsymbol{x}_i - \bar{\boldsymbol{x}}^t) + \bar{\boldsymbol{x}}^t, \text{ for } i = 1, \ldots, n.$$

The expansion/contraction coefficient β is drawn from some distribution $p_\beta(.)$. Notice that β is defined as the ratio of the distance between the candidate component, $\boldsymbol{y}_i$, and the centroid of the simplex at the current state, $\bar{\boldsymbol{x}}^t$, and the distance of the component to be updated, $\boldsymbol{x}_i$, and $\bar{\boldsymbol{x}}^t$, that is

$$\beta = \frac{||\boldsymbol{y}_i - \bar{\boldsymbol{x}}^t||}{||\boldsymbol{x}_i - \bar{\boldsymbol{x}}^t||}.$$

If β takes a value less than one then a contraction move is proposed while if β is greater than one, an expansion move is proposed, see Figure 2.5. The shape of the simplex stays unchanged when the coefficient β takes values close to unity. The proposal distribution to choose the expansion/contraction coefficient value is important for the mixing of the chain. Experimentation showed that a $Ga(1, 2)$ is a good choice of the proposal distribution to draw the β value.

The designed expansion and contraction moves ensure that the chain moves

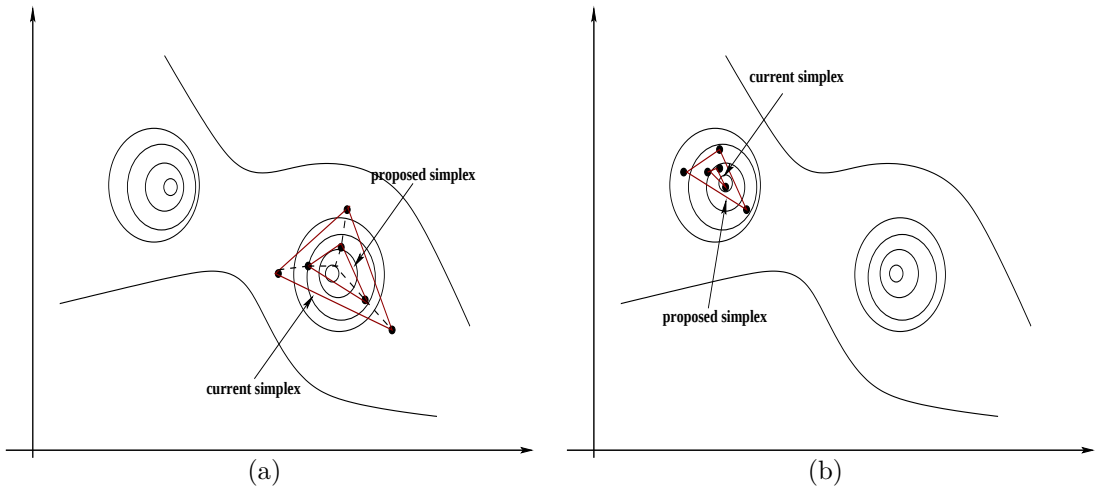


Figure 2.5: (a) The vertices are too far away so the simplex is contracted. (b) The vertices are too close to each other so the simplex is expanded.

from the proposed configuration back to the current one by defining β^*. Hence, β^* is the expansion/contraction coefficient for the reverse move which ensures that the chain traverses forwards and backwards in time. The coefficient β^* is evaluated as

$$\boldsymbol{x}_i = \beta^*(\boldsymbol{y}_i - \bar{\boldsymbol{x}}^{t+1}) + \bar{\boldsymbol{x}}^{t+1}$$

so that

$$\beta^* = \frac{||\boldsymbol{x}_i - \bar{\boldsymbol{x}}^{t+1}||}{||\boldsymbol{y}_i - \bar{\boldsymbol{x}}^{t+1}||}.$$

Hence, β^* is the ratio of the distance between the component to be updated in the current state, $\boldsymbol{x}_i$, and the centroid of the simplex in the proposed state, $\bar{\boldsymbol{x}}^{t+1}$, versus the distance between the proposed component, $\boldsymbol{y}_i$, and $\bar{\boldsymbol{x}}^{t+1}$.

The proposed configuration $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$ is accepted with probability

$$\alpha_{exp/con}\left(\boldsymbol{x}^t, \boldsymbol{y}\right) = \min\left\{1, \frac{\pi(\boldsymbol{y})p_\beta(\beta^*)}{\pi(\boldsymbol{x}^t)p_\beta(\beta)}\right\} \tag{2.6}$$

and the new state of the simplex is $\boldsymbol{x}^{t+1} = \boldsymbol{y}$. Otherwise, it is rejected and the chain stays where it currently is.

2.3.3 Metropolis-Hastings move

A Metropolis-Hastings move is carried out to update each of the vertices of the simplex. Hence, each of the components of the simplex is generated from $\boldsymbol{y}_i \sim N(\boldsymbol{x}_i, \sigma^2\mathbf{I}_d)$, for $i = 1, \ldots, n$ where σ^2 is the proposal variance. Here, small values of σ^2 lead to small changes of the simplex and for this reason, the proposed moves tend to be accepted more often. In this case, the acceptance probability is

$$\alpha_{mh}\left(\boldsymbol{x}^t, \boldsymbol{y}\right) = \min\left\{1, \frac{\pi(\boldsymbol{y})}{\pi(\boldsymbol{x}^t)}\right\}. \tag{2.7}$$

Here, it should be mentioned that depending on the dimensionality of the problem, one can use a single update or a global one. If the dimension is big then global updates will not be accepted as often resulting in slow mixing. In this case, single updates are recommended.

2.3.4 Relation between the reflection coefficients α and α^*

In general, if three points a, b and c are in the same line then the distance between points a and c can be defined as $||a, c|| = ||a, b|| + ||b, c||$, for $a \leq b \leq c$. Hence,

the reflection coefficient α is

$$\begin{aligned} \alpha &= \frac{||\boldsymbol{y} - \bar{\boldsymbol{x}}^t||}{||\bar{\boldsymbol{x}}^t - \boldsymbol{x}_c||} \\ \alpha + 1 &= \frac{||\boldsymbol{y} - \bar{\boldsymbol{x}}^t|| + ||\bar{\boldsymbol{x}}^t - \boldsymbol{x}_c||}{||\bar{\boldsymbol{x}}^t - \boldsymbol{x}_c||} \\ \alpha + 1 &= \frac{||\boldsymbol{y} - \boldsymbol{x}_c||}{||\bar{\boldsymbol{x}}^t - \boldsymbol{x}_c||} \\ ||\boldsymbol{y} - \boldsymbol{x}_c|| &= (\alpha + 1)||\bar{\boldsymbol{x}}^t - \boldsymbol{x}_c|| \end{aligned} \tag{2.8}$$

and the reflection coefficient α^* is

$$\begin{aligned} \alpha^* &= \frac{||\boldsymbol{x}_c - \bar{\boldsymbol{x}}^{t+1}||}{||\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}||} \\ \alpha^* + 1 &= \frac{||\boldsymbol{x}_c - \bar{\boldsymbol{x}}^{t+1}|| + ||\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}||}{||\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}||} \\ \alpha^* + 1 &= \frac{||\boldsymbol{x}_c - \boldsymbol{y}||}{||\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}||}. \end{aligned} \tag{2.9}$$

Thus, inserting equation (2.8) in equation (2.9), the relationship between α and α^* can be defined as

$$\begin{aligned} \alpha^* + 1 &= \frac{(\alpha + 1)||\bar{\boldsymbol{x}}^t - \boldsymbol{x}_c||}{||\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}||} \\ \alpha^* &= \alpha\frac{||\bar{\boldsymbol{x}}^t - \boldsymbol{x}_c||}{||\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}||} + \frac{||\bar{\boldsymbol{x}}^t - \boldsymbol{x}_c||}{||\bar{\boldsymbol{x}}^{t+1} - \boldsymbol{y}||} - 1. \end{aligned}$$

Algorithm 3 Simplex sampler

Assume that at time t, the current configuration of the simplex is $\boldsymbol{x}^t = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, for $\boldsymbol{x}_i \in \mathbb{R}^d$ and the target distribution is $\pi(\cdot)$ given in (2.2).

Reflection move:

1. Choose the current component $\boldsymbol{x}_c$ of the simplex with probability $\ell_t(c)$ given in (2.3).
2. Calculate the centroid of the simplex $\bar{\boldsymbol{x}}^t$.
3. Draw the reflection coefficient α from some distribution $p_\alpha(\cdot)$.
4. Calculate the proposed component as $\boldsymbol{y} = (1+\alpha)\,\bar{\boldsymbol{x}}^t - \alpha \boldsymbol{x}_c$.
5. Accept the proposed component with probability $\alpha_{ref}(\boldsymbol{x}_c, \boldsymbol{y})$ given in (2.5) and set $\boldsymbol{x}^{t+1} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{y}, \ldots, \boldsymbol{x}_n\}$ otherwise, set $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$.

Expansion and Contraction move:

1. Calculate the centroid of the simplex $\bar{\boldsymbol{x}}^t$.
2. Draw the expansion/contraction coefficient β from some distribution $p_\beta(\cdot)$.
3. Calculate the proposed simplex as $\boldsymbol{y}_i = \beta(\boldsymbol{x}_i - \bar{\boldsymbol{x}}^t) + \bar{\boldsymbol{x}}^t$, for $i = 1, \ldots, n$.
4. Accept the proposed simplex with probability $\alpha_{exp/con}(\boldsymbol{x}^t, \boldsymbol{y})$ given in (2.6) and set $\boldsymbol{x}^{t+1} = \boldsymbol{y}$ otherwise, set $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$.

Metropolis - Hastings update:

1. Draw the proposed simplex as $\boldsymbol{y} \sim N(\boldsymbol{x}^t, \sigma^2 \mathbf{I}_d)$, where σ^2 is the proposal variance.
2. Accept the proposed configuration with probability $\alpha_{mh}(\boldsymbol{x}^t, \boldsymbol{y})$ given in (2.7) and set $\boldsymbol{x}^{t+1} = \boldsymbol{y}$ or otherwise, set $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$.

2.4 Examples

In this section, the simplex sampler is tested on different examples and its performance is compared to the Metropolis-Hastings (M-H) algorithm (Hastings (1970)), Real-parameter Evolutionary Monte Carlo (EMC) algorithm (Liang and Wong (2001)) and tempered transitions method (Neal (1996)).

For each of the different methods, the mean estimates and their standard deviation (std), the standard errors, the bias of the estimates and the Monte Carlo standard error are evaluated. The standard error (ste) is estimated as $\text{ste} = \text{std}/\sqrt{N}$, where N is the sample size from the MCMC algorithm. The bias of the estimates is calculated as $|E(\hat{\boldsymbol{\theta}}) - \boldsymbol{\theta}|$, which is the absolute value of the difference between the estimated posterior expectation, $E(\hat{\boldsymbol{\theta}})$, and the parameter values, $\boldsymbol{\theta}$. The Monte Carlo standard error (MCste) is evaluated as the standard error of the average estimates divided by the square root of the sample size.

In addition, the integrated autocorrelation (IA) is calculated using the method proposed by Sokal (1997). Then, using the measure of integrated autocorrelation, the efficient sample size (ESS) is defined as $N/(\text{IA}\,\mathtt{t})$ where N is the sample size generated from the MCMC algorithm and $\mathtt{t}$ is the computation time to execute the algorithm. Thus, high values of efficient sample size give a measure of performance of the algorithm which accounts for both computational time and number of MCMC draws.

In the following examples, the proposal parameter values are chosen based on the acceptance probability. Acceptance probability values between 0.20 and 0.40 may indicate that the algorithm converges to the distribution of interest (Gelman et al. (1996)). The Real-parameter Evolutionary Monte Carlo algorithm

and tempered transition method use ten different temperatures that are equally spaced in $[0, 1]$. Notice that only one out of the ten is the temperature of interest. Consequently, at the end of each sweep only the sample under temperature one is stored while the rest of them are disregarded. In order to make the algorithms comparable, the total number of parameter updating steps was kept constant as each algorithm will typically have a different number of updating steps within each iteration of the Markov chain, accounting for the different number of updating steps per MCMC iteration.

2.4.1 Mixture of two Normal distributions

The first example is a simple mixture problem which for badly designed algorithms can result in poor performance. Consider the mixture of two Normal distributions such as

$$f(\boldsymbol{x}) = wN(\boldsymbol{x}; \mu_1, \sigma^2) + (1 - w)N(\boldsymbol{x}; \mu_2, \sigma^2)$$

The parameters w and σ are known while μ_1 and μ_2 are assumed unknown. Thus, the prior beliefs about the means are expressed through a Normal distribution such as $p(\mu_1, \mu_2) = N(\mu_1|\theta, \sigma^2/\lambda)N(\mu_2|\theta, \sigma^2/\lambda)$. Then, the target distribution is the posterior which is defined as $\pi(\mu_1, \mu_2|\boldsymbol{x}) \propto L(\boldsymbol{x}|\mu_1, \mu_2)\, p(\mu_1, \mu_2)$ where $L(\boldsymbol{x}|\mu_1, \mu_2)$ is the likelihood.

Ten different data sets are generated consisting of 100 responses each from the mixture $0.2N(0, 1) + 0.8N(2, 1)$ so that $w = 0.2$, $\mu_1 = 0$, $\mu_2 = 2$ and $\sigma = 1$. The parameters of the prior distribution are $\theta = 1$ and $\lambda = 0.1$. The contour plot of the above mixture is presented in Figure 2.6(a). It is obvious that the coordinates are

highly correlated and consequently, the Metropolis-Hastings algorithm is likely to suffer from slow mixing. The initial position of the chain is chosen under two different schemes. First, the initial position of the chain is chosen under the tails of the posterior distribution. The aim is to figure out how fast the algorithm can escape and sample under the whole sampling area. Then, the initial position of the chain is also randomly selected for each data set.

For the simplex sampler case, the simplex is defined by three vertices in the two-dimensional space. The reflection coefficient is drawn from a Normal distribution, $N(0, 4^2)$, and the expansion/contraction coefficient is generated from a Gamma distribution, $\text{Ga}(1, 2)$, with mean 2 and variance 4. A Metropolis-Hastings move is carried out to update the vertices of the simplex using a Normal proposal distribution, $N(0, 0.1^2)$.

For the Real-parameter Evolutionary Monte Carlo algorithm, a Normal proposal distribution, $N(0, 0.1^2)$, is used in the mutation operator and the Metropolis-Hastings algorithm is applied in the snooker crossover operator considering a Normal proposal distribution, $N(0, 0.1^2)$ and ten iterations in total. Liang and Wong (2001) mention that even one iteration may be enough to draw the r value. Here, the mutation probability is 0.20.

For the tempered transition method, the transition probability is $N(0, 1)$ using 100 different iterations under each temperature. The tempered transitions method stores only the sample under temperature one. The simplex sampler runs for $90,000$ iterations regarding the first $10,000$ as burn in.

The contour plot of the posterior distribution on the log scale is presented in Figure 2.6. From the contour plot, it seems that there is high correlation between the variables. Here, the starting position of the chain is chosen under the tails.

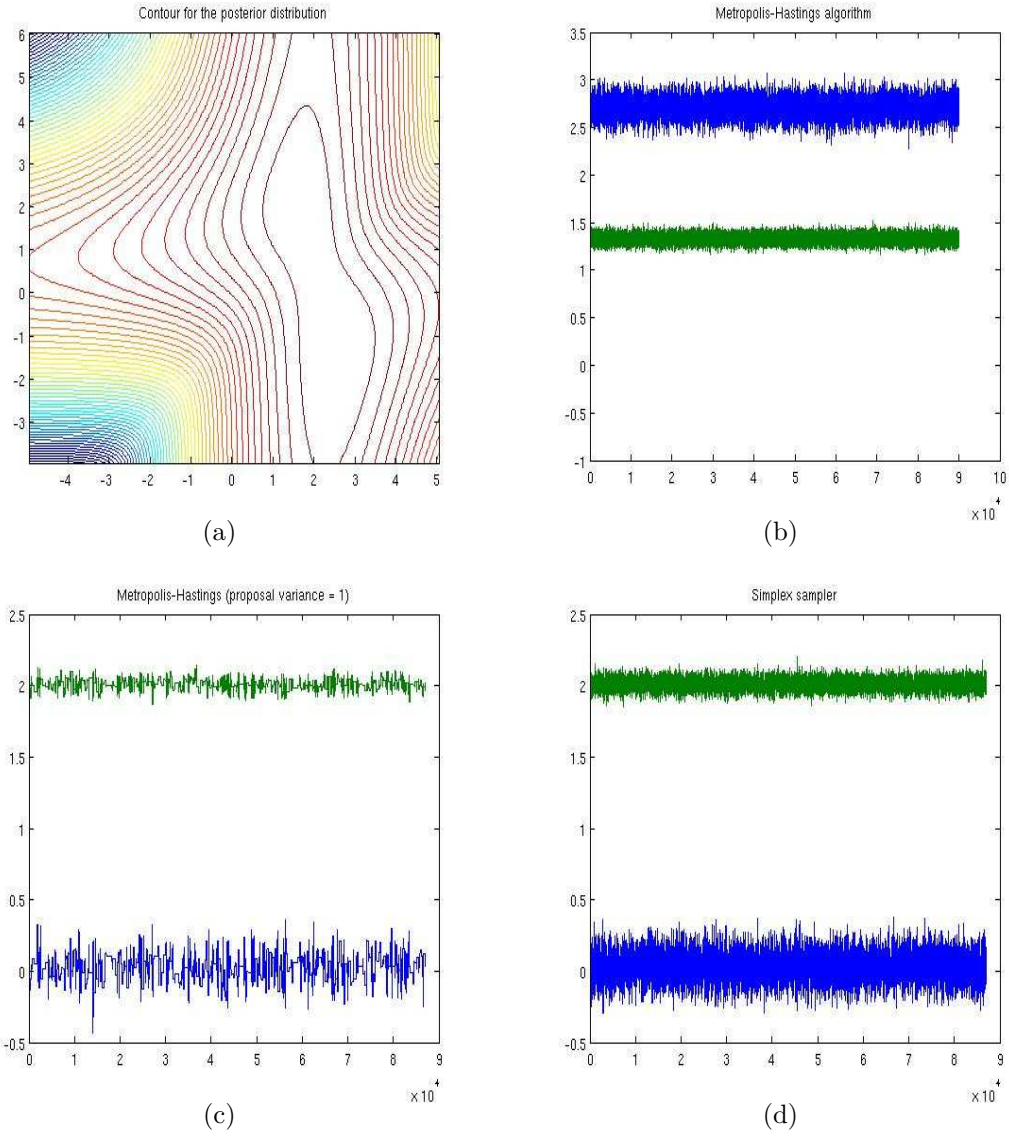


Figure 2.6: (a) Contour plot of the target distribution in log scale. (b) Metropolis-Hastings algorithm with proposal variance 0.1. It fails to sample from the actual target values. (c) Metropolis-Hastings algorithm with proposal variance 1. It converges to the actual target values but the mixing of the chain is poor. (d) The simplex sampler converges from the beginning to the actual target values.

Different proposal variances are used for the Metropolis-Hastings algorithm, see Figure 2.6. When the proposal variance is 0.1, the Metropolis-Hastings algorithm fails to converge to the actual target values even though the acceptance probability is around 0.35. Increasing the proposal variance to 1, the algorithm samples the actual target values but then the acceptance probability is less than 1% which results in slow mixing of the chain. If one chooses to increase the number of iterations then it is possible to get better estimates. The acceptance probabilities for the simplex sampler are 0.36 for the reflection move, 0.43 for the expansion/contraction move and 0.33 for the Metropolis-Hastings update. Usually, the proposal variance is tuned according to the acceptance ratio. According to Gelman et al. (1996) acceptance probability values between 0.20 and 0.40 indicate that the algorithm has been tuned effectively.

Next, the initial position of the chain is randomly chosen for each data set. The algorithms give in general quite good estimates of the unknown parameter values. The estimations together with their standard deviations and standard errors are presented in Table 2.1. In this case, the actual posterior means are $\mu_1 = 0.2524$ and $\mu_2 = 1.8263$. The simplex sampler gives more accurately estimates of the posterior means compared to the other methods. The tempered transitions method estimates the posterior means more accurately compared to the Metropolis-Hastings and Real-parameter Evolutionary Monte Carlo algorithm.

The standard deviation and standard error of the estimates is similar to each method and for this reason, the simplex sampler has an advantage since it gives more accurate estimates. Moreover, the simplex sampler gives also less biased estimates compared to the other methods, see Table 2.2. Then again, the Metropolis-Hastings algorithm has the most biased estimates. The Monte Carlo

	simplex sampler		M-H		EMC		tempered transitions	
Target	μ_1	μ_2	μ_1	μ_2	μ_1	μ_2	μ_1	μ_2
Mean	0.2576	1.8291	-0.1463	1.8639	0.3856	1.7967	0.1337	1.8499
Std	0.5998	0.1926	0.5480	0.1743	0.8035	0.2532	0.4770	0.1591
Ste	0.0012	0.0004	0.0011	0.0004	0.0016	0.0005	0.0010	0.0003

Table 2.1: Mean, standard deviation and standard error for the estimated parameters for the mixture of two Normal distributions using each of the different methods.

standard error is very small meaning that the estimates of each method are similar between the different simulated data sets, see Table 2.3. Table 2.4 presents the integrated autocorrelation and efficient sample size for each of the different methods. The integrated autocorrelation for the Metropolis-Hastings algorithm is the biggest one. Then again, the tempered transitions method gives the smaller integrated autocorrelation. The efficient sample size is similar for each method indicating that the computational time for the tempered transitions method and the Real-parameter Evolutionary Monte Carlo algorithm is higher compared to the simplex sampler one. In conclusion, the simplex sampler performs better.

Method	μ_1	μ_2	Average
simplex sampler	0.0052	0.2297	0.1175
M-H	0.3987	0.1949	0.2968
EMC	0.1332	0.2621	0.1977
tempered transitions	0.1187	0.2089	0.1638

Table 2.2: Bias of the estimated posterior means using each of the different methods for the mixture of two Normal distributions.

The acceptance probabilities for the simplex sampler are 0.3794 for the reflection move, 0.4227 for the expansion/contraction move and 0.3471 for the

Method	μ_1	μ_2	Average
simplex sampler	0.0012	0.0003	0.0008
M-H	0.0009	0.0002	0.0006
EMC	0.0012	0.0003	0.0007
tempered transitions	0.0016	0.0001	0.0009

Table 2.3: Monte Carlo standard error using each of the different methods for the mixture of two Normal distributions.

Methods	IA	ESS	N	time
simplex sampler	514	0.52	36,333	135,94
M-H	617	0.73	1,000,000	2220,20
EMC	224	0.46	28,572	277,29
Tempered transitions	113	0.82	50,000	539,61

Table 2.4: Integrated autocorrelation, efficient sample size, number of iterations and time for the mixture of two Normal distributions using each of the different methods.

Metropolis-Hastings update. The acceptance probability for the Metropolis-Hastings algorithm is 0.3523 and for the tempered transitions method is 0.4576. The acceptance probabilities for the Real-parameter Evolutionary Monte Carlo algorithm are 0.5287 for the mutation operator, 0.4594 for the real crossover operator, 0.3023 for the snooker crossover and 0.7396 for the exchange move.

2.4.2 Autoregressive model of order three

The next example is an autoregressive process of order three with unknown parameter values and residual variance. The autoregressive process is modeled as

$$x_t = \sum_{i=1}^{3} \alpha_i x_{t-1} + \epsilon_t,$$

where the residual ϵ_t follow a Normal distribution, $N(0, \sigma_\epsilon^2)$.

Each process is generated with parameters $\alpha_1 = 0.5$, $\alpha_2 = 0.3333$, $\alpha_3 = -0.1667$ and $\sigma_\epsilon^2 = 1$. Ten different data sets are generated according to the above process where each of them contains 500 observations. A randomly selected data set is presented in Figure 2.7. A Normal prior distribution is assumed for the autoregressive coefficients, i.e $\alpha_i \sim N(0, 1)$, for $i = 1, 2, 3$ and a Gamma distribution for the variance of the residual, i.e $\sigma_\epsilon^2 \sim \text{Ga}(1, 1)$.

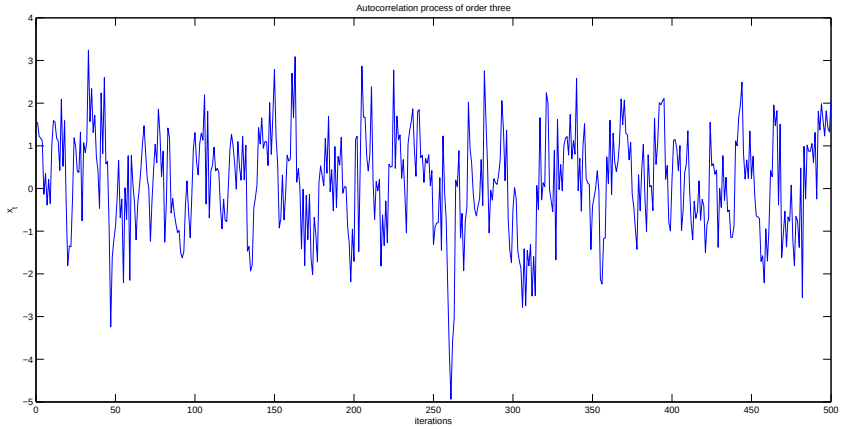


Figure 2.7: Autocorrelation process of order three for a randomly selected data set.

In this case, the full-conditional posterior distributions are tractable and therefore, Gibbs updates (Geman and Geman (1984)) are implemented instead of Metropolis-Hastings ones. Thus, the performance of the simplex sampler is compared to the Gibbs sampler one instead.

Here, the simplex is defined by five vertices in four dimensional space. The reflection coefficient value is simulated from a $N(0, 4^2)$ while the expansion/contraction coefficient is drawn from a $\text{Ga}(1, 2)$. Moreover, for the Real-parameter Evolutionary Monte Carlo algorithm, in the mutation operator a Normal proposal

distribution, $N(0, 0.1^2)$, is considered and in the snooker crossover operator, the Metropolis-Hastings algorithm. The Metropolis-Hastings algorithm uses a Normal proposal distribution, $N(0, 0.1^2)$. The mutation probability is 0.20. Furthermore, for the tempered transitions method, the transition probability is $N(0, 1)$ using 100 different iterations under each temperature. The simplex sampler runs for $90,000$ iterations regarding the first $10,000$ as burn in.

The estimated parameter values together with their standard deviation are presented in Table 2.5. All methods give quite good estimates of the parameter values. The standard deviation is similar for each method apart from the Real-parameter Evolutionary Monte Carlo algorithm which has higher standard deviation compared to the other ones. This may imply that the algorithm should be ran for longer time or it needs a longer burn in.

The bias of the estimates, the Monte Carlo standard error across the different data sets, the integrated autocorrelation and the efficient sample size are presented in Table 2.6. The bias is similar to all methods. Hence, the autoregressive coefficients and the residual error are estimated quite well. Moreover, the Monte Carlo standard error is small for all methods meaning that the various estimates do not differ a lot between the different simulated data sets. The simplex sampler has the highest efficient sample size followed by the Gibbs sampler, the Real-parameter Evolutionary Monte Carlo algorithm and the tempered transition method. Hence, the simplex sampler is more efficient and uses less computational power to provide a better efficient sample size.

The acceptance probabilities for the simplex sampler are 0.2623 for the reflection move and 0.3860 for the expansion/contraction move. The acceptance probability for the tempered transitions method is 0.9249 which might explain

why the efficient sample size is quite small and the integrated autocorrelation time is quite high. The Real-parameter Evolutionary Monte Carlo algorithm is 0.1555 for the mutation operator, 0.1623 for the real crossover operator, 0.2355 for the snooker crossover and 0.3160 for the exchange move. Hence, the mixing of the methods is quite good.

True parameter values		0.5	0.3333	-0.1667	1
simplex sampler	mean	0.5224	0.3367	-0.1822	0.9747
	std	0.0433	0.0474	0.0439	0.0619
Gibbs sampler	mean	0.5324	0.3167	-0.1822	0.9744
	std	0.0443	0.0483	0.0443	0.0619
EMC	mean	0.5308	0.3088	-0.1775	0.9867
	std	0.3546	0.3943	0.3592	0.3822
Tempered transitions	mean	0.5602	0.3076	-0.2154	0.9095
	std	0.0431	0.0472	0.0427	0.0560

Table 2.5: Estimated parameter values and standard deviations for the autoregressive example using each of the different methods.

Method	Bias	MCste	IA	ESS	N	time
simplex sampler	0.0166	0.0008	12	37	36,333	81,83
Gibbs	0.0225	0.0008	22	34	1,000,000	1336,84
EMC	0.0198	0.0009	55	3	28,572	173,16
tempered transitions	0.0563	0.0008	50	3	50,000	333,33

Table 2.6: Bias, Monte Carlo standard error across the different data sets, integrated autocorrelation, efficient sample size, number of iterations and time for the estimated parameters using each of the different techniques.

2.4.3 Linear Regression Problem

A linear regression model with unknown parameters is considered next. Ten different data sets are simulated where each of them consists of 100 independent observations. The relationship between the predictor variables, $\boldsymbol{x} = (x_{i1}, x_{i2}, \ldots, x_{i10})$ and the response, y_i, is given below

$$y_i = \sum_{k=1}^{10} \beta_k x_{ik} + \epsilon_i$$

for $i = 1, \ldots, 100$, where the residual ϵ_i follow a Normal distribution, $N(0, 2.5^2)$. For each data set, the predictor variables are generated from a multivariate Normal distribution, $N_n(\mathbf{0}, \Sigma)$, with $\mathbf{0} = (0, \ldots, 0)$ and $\Sigma = \begin{bmatrix} 1 & & 0.9 \\ & \ddots & \\ 0.9 & & 1 \end{bmatrix}$.

The prior distribution for the parameter $\boldsymbol{\beta}$ is assumed to be $N(\mathbf{0}, 10^2\mathbf{I})$. Thus, the distribution of interest is the posterior one, $\pi(\boldsymbol{\beta}|\mathbf{x}, \mathbf{y}) \propto L(\mathbf{y}|\mathbf{x}, \boldsymbol{\beta})\, p(\boldsymbol{\beta})$ where $L(\mathbf{y}|\mathbf{x}, \boldsymbol{\beta})$ is the likelihood.

In this case, the simplex is defined by eleven vertices in the ten dimensional space. The reflection coefficient is drawn from a $N(0, 4^2)$, while the expansion/contraction coefficient is generated from a $\text{Ga}(1, 2)$. The Metropolis-Hastings move uses a $N(0, 0.1^2)$ distribution to update each of the components of the simplex. Moreover, the Real-parameter Evolutionary Monte Carlo algorithm uses a $N(0, 0.1^2)$ distribution for the mutation operator and the Metropolis-Hastings algorithm in the snooker crossover operator. The Metropolis-Hastings algorithm uses a $N(0, 0.1^2)$ proposal distribution. The mutation probability is 0.20. Likewise, for the tempered transitions method a $N(0, 1)$ proposal distribution is considered with 100 different iterations under each temperature.

In the case of the Real-parameter Evolutionary Monte Carlo algorithm and the tempered transitions method, only the sample under the temperature of interest is stored at the end of each sweep using extra computational power. The simplex sampler runs for $90,000$ iterations regarding the first $10,000$ as burn in.

All methods give quite good estimates of the parameter values but the simplex sampler gives the best estimates, see Table 2.7. The standard deviation for the simplex sampler and the Real-parameter Evolutionary Monte Carlo algorithm is smaller compared to the other ones. The standard error of the estimates is also very small for each method.

The bias of the estimates, the Monte Carlo standard error across the different data sets, the integrated autocorrelation time and the efficient sample size can be seen in Table 2.8. The simplex sampler gives the smallest bias to the estimates. Moreover, the Monte Carlo standard error across the different data sets is quite small for each method. The simplex sampler gives similar integrated autocorrelation to the tempered transitions method, while the Metropolis-Hastings algorithm gives the highest integrated autocorrelation. In addition, the simplex sampler gives the highest efficient sample size, with the tempered transitions method having the next highest one. The simplex sampler is the most efficient of the other methods because it gives less bias estimates while it also uses less computational time.

The acceptance probabilities for the simplex sampler are 0.3474 for the reflection move, 0.4243 for the expansion/contraction move and 0.3741 for the Metropolis-Hastings update. The acceptance probability for the tempered transitions method is 0.1905 while for the Real-parameter Evolutionary Monte Carlo

algorithm is 0.263 for the mutation operator, 0.3583 for the real crossover operator, 0.2193 for the snooker crossover and 0.7656 for the exchange move. Thus, the methods result in good mixing according to their acceptance probabilities.

True	1	0.5	-0.5	0	0	0	0	0	0	0
Simplex sampler										
Mean	0.9463	0.4730	-0.5402	-0.0354	-0.1442	-0.1357	0.3282	-0.0953	0.0511	0.0123
Std	0.3886	0.4803	0.4201	0.4389	0.4567	0.4627	0.4764	0.4740	0.4523	0.4342
stde	0.0008	0.0010	0.0009	0.0009	0.0009	0.0009	0.0009	0.0010	0.0009	0.0009
M-H										
Mean	0.9188	0.3810	-0.2303	-0.0313	-0.0145	-0.2549	0.1746	-0.0226	0.0307	-0.0725
Std	0.4851	0.6211	0.6649	0.6666	0.6151	0.4629	0.4842	0.6176	0.4632	0.6676
stde	0.0010	0.0013	0.0014	0.0014	0.0013	0.0009	0.0010	0.0013	0.0009	0.0014
EMC										
Mean	0.9110	0.3950	-0.2334	-0.0384	-0.0156	-0.2651	0.1729	-0.0180	0.0331	-0.0758
Std	0.2031	0.2422	0.2145	0.2146	0.2376	0.2278	0.2390	0.2322	0.2301	0.2321
stde	0.0004	0.0005	0.0004	0.0004	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005
Tempered transitions										
Mean	0.9451	0.3526	-0.6445	-0.0318	-0.1445	-0.2345	0.3356	-0.0895	0.0519	0.0189
Std	0.5905	0.7376	0.6383	0.6587	0.6996	0.6987	0.7174	0.7099	0.6908	0.6516
stde	0.0012	0.0015	0.0013	0.0013	0.0014	0.0014	0.0015	0.0014	0.0014	0.0013

Table 2.7: Estimated parameter values together with their standard deviation and standard error for the linear regression model.

Method	Bias	MCste	IA	ESS	N	time
simplex sampler	0.0923	0.0002	278	3.20	36,333	40,84
M-H	0.1071	0.0003	1592	0.94	1,000,000	668,23
EMC	0.1079	0.0002	400	0.86	28,570	83,06
tempered transitions	0.1253	0.0002	250	1.23	50,000	162,60

Table 2.8: Bias, Monte Carlo error across the different data sets, integrated autocorrelation, efficient sample size, number of iterations and time for the linear regression example using each of the different methods.

2.4.4 A Bimodal example

The target distribution in this case is a mixture of two five-dimensional Normal distributions described in Liang and Wong (2001). The distribution of interest is defined as

$$\pi(\boldsymbol{x}) = 1/3N_5(\boldsymbol{x}; \mathbf{0}, \mathbf{I_5}) + 2/3N_5(\boldsymbol{x}; \mathbf{5}, \mathbf{I_5})$$

where $\boldsymbol{x} \in \mathbb{R}^5$, $\mathbf{0} = (0, \ldots, 0)$ and $\mathbf{I_5}$ is the identity matrix.

The simplex is defined by six vertices in five-dimensional space. The reflection coefficient is drawn from a Normal distribution such as $N(0, 4^2)$ and the expansion/contraction coefficient is generated from a Gamma distribution that is $\text{Ga}(1, 2)$. A Metropolis-Hastings move is carried out to update the vertices of the simplex using a Normal proposal distribution, $N(0, 1)$. The proposal parameter values are again chosen based on the acceptance probability.

In addition, for the Real-parameter Evolutionary Monte Carlo algorithm a $N(0, 0.1^2)$ distribution is used in the mutation operator and the Metropolis-Hastings algorithm in the snooker crossover operator with a $N(0, 1)$ proposal distribution. The mutation probability is again 0.20. Furthermore, the tempered transitions method uses a $N(0, 1)$ proposal distribution and 100 iterations under each temperature. The simplex sampler runs for $90,000$ iterations regarding the first $10,000$ as burn in.

The algorithms should sample under the two modes with the correct frequency to get a representative sample. The simplex sampler does not sample with the correct frequency under the modes even though, it does not get trapped under a single one, see Figure 2.8. The reason is that the simplex method of Nelder and Mead (1965) is an optimization algorithm that aims to find a local maxima.

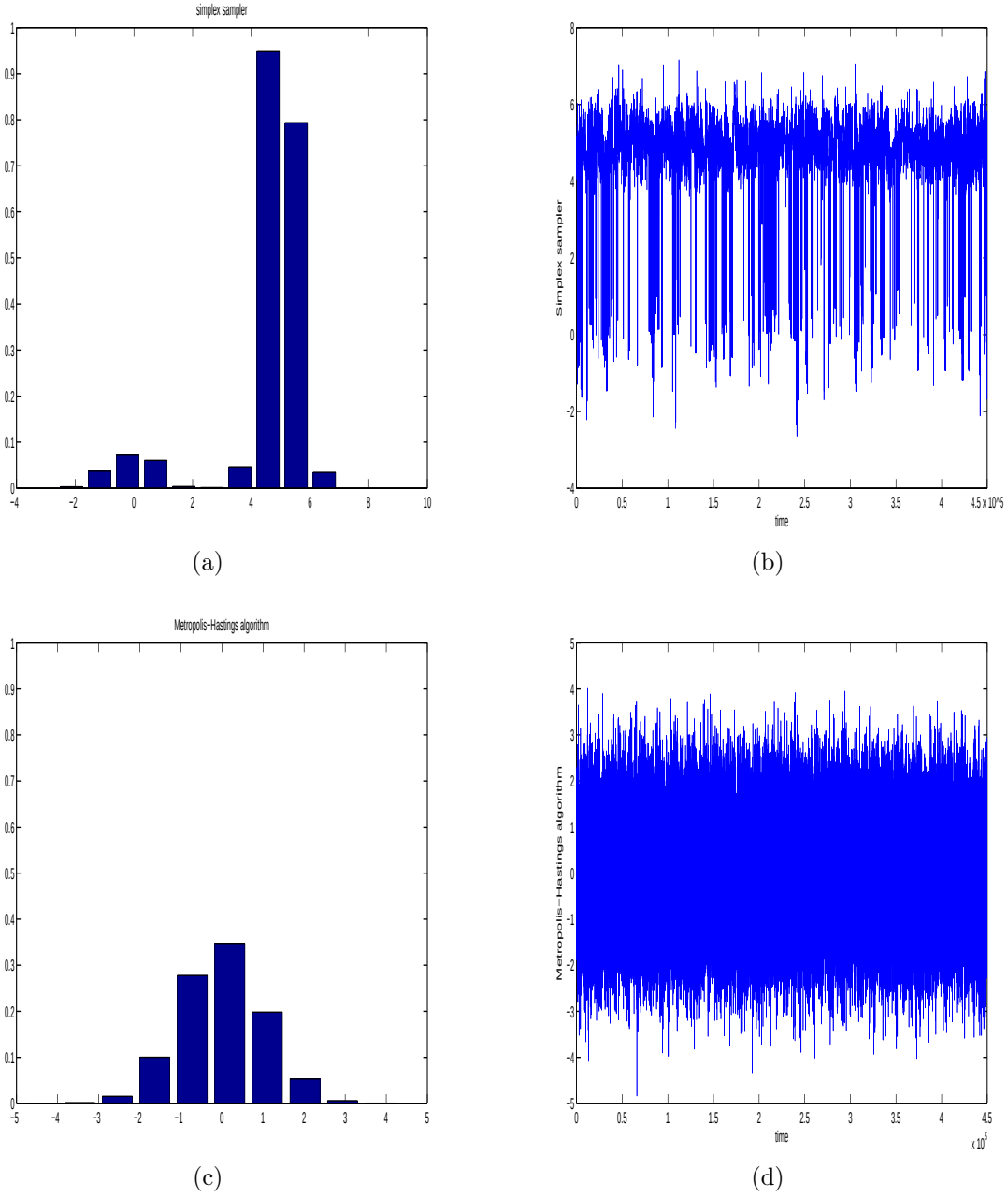
Hence, the simplex sampler tends, by design, to sample under a single local maxima unwilling to search for another one. The Metropolis-Hastings algorithm also does not traverse the whole sampling space and gets trapped under a single mode, see Figure 2.8 .

The tempered transitions method and Real-parameter Evolutionary Monte Carlo algorithm sample under both modes, see Figure 2.9. The tempered transition method samples with better frequency under both modes compared to the Real-parameter Evolutionary Monte Carlo algorithm. Hence, in the case of a multi-modal target distribution, it may be useful to assume a tempering vector in order to sample from the distribution of interest without being trapped under a single maxima.

The efficient sample size and integrated autocorrelation are presented in Table 2.9. The tempered transition method gives the smaller integrated autocorrelation. The simplex sampler has the highest efficient sample size while the tempered transitions method and the Real-parameter Evolutionary Monte Carlo algorithm give a fairly similar but lower efficient sample size.

The Metropolis-Hastings algorithm do not traverse the whole sampling space and gets trapped under a single maximum. The overall mixing for the Metropolis-Hastings algorithm seems to be poor even though the mixing within mode appears to be quite good. The simplex sampler tends to spend more time sampling under one mode and for this reason, the posterior density estimate of the bimodal distribution is poor.

Thus, the acceptance probabilities are 0.2755 for the simplex sampler, 0.3142 for the Metropolis-Hastings, 0.3214 for the Real-parameter Evolutionary Monte Carlo algorithm and finally, 0.1954 for the tempered transitions method.



(a) (b)

(c) (d)

Figure 2.8: (a) Histogram and (b) trace plot for the simplex sampler. (c) Histogram and (d) trace plot for the Metropolis-Hastings algorithm.

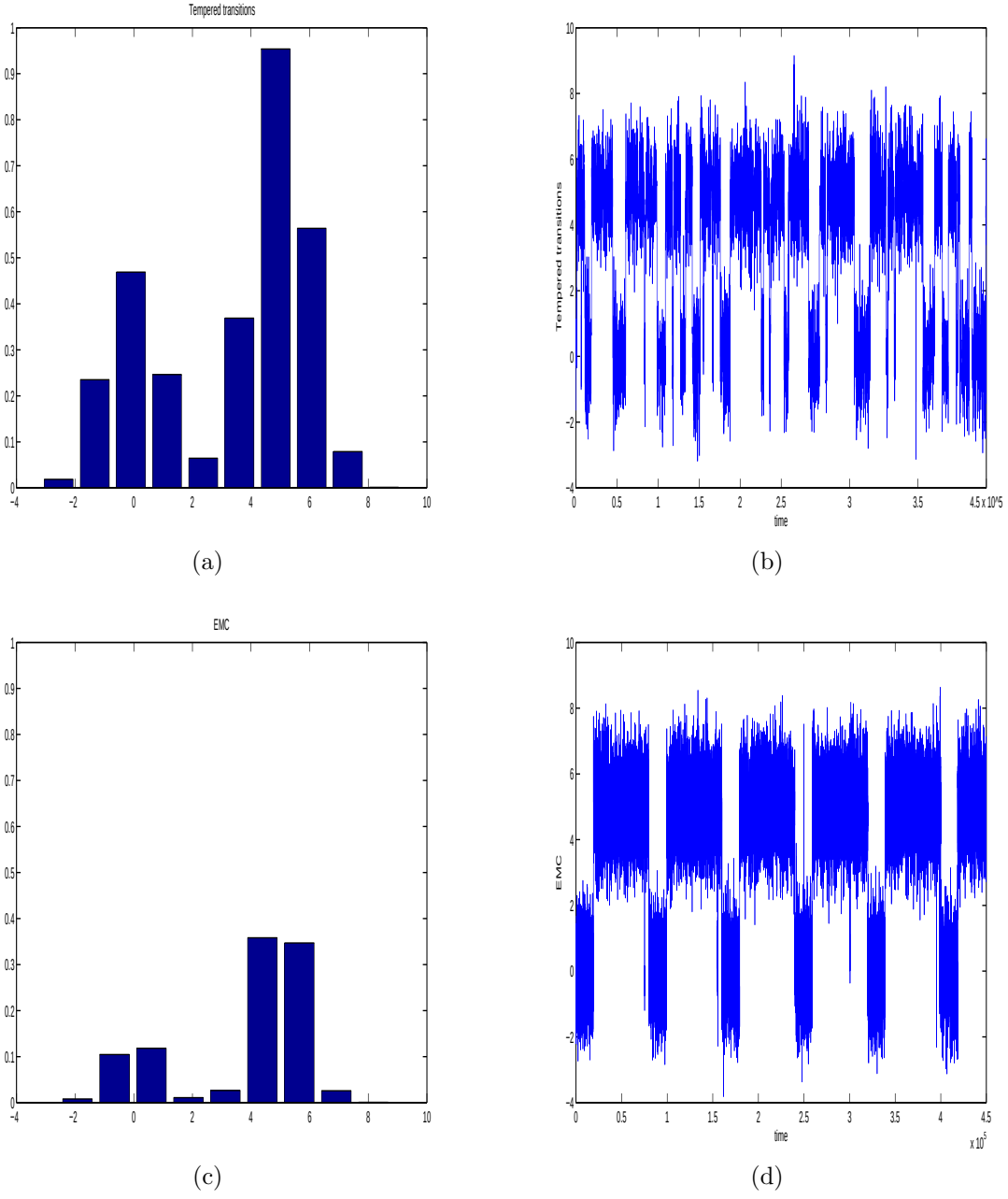


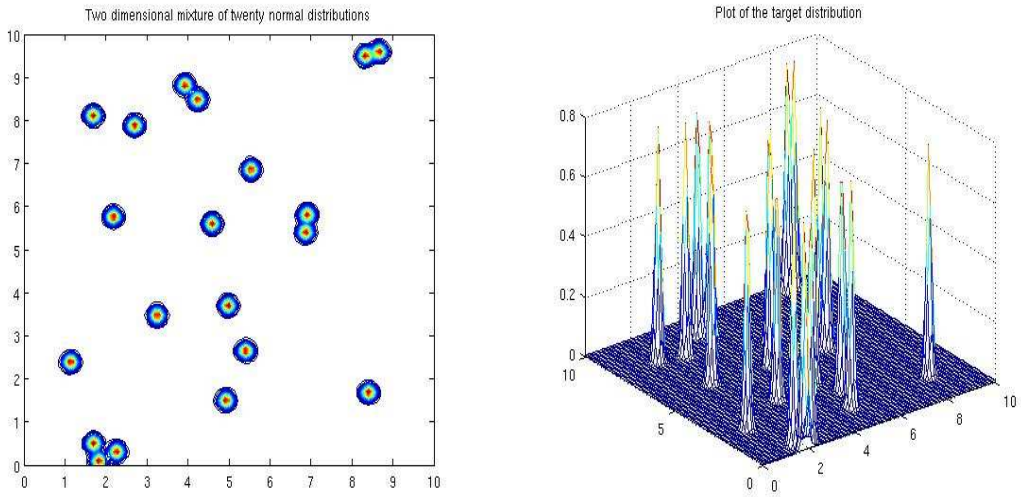
Figure 2.9: (a) Histogram and (b) trace plot for the tempered transitions method. (c) Histogram and (d) trace plot for the Real-parameter Evolutionary Monte Carlo method.

Methods	IA	ESS
simplex sampler	53	11
EMC	45	5
tempered transitions	31	7

Table 2.9: Integrated autocorrelation and efficient sample size for the bimodal example using each of the different methods.

2.4.5 Mixture of twenty Normal distributions

A mixture of twenty bivariate Normal distributions is used to test the performance of the algorithms as presented in Liang and Wong (2001). The distribution of interest is multi-modal and it is described by high and essentially zero probability areas. The modes are well separated from each other, see Figure 2.10. Therefore,



(a) (b)

Figure 2.10: (a) Contour plot and (b) plot for the mixture of twenty Normal distributions.

it is a challenge for the algorithms to jump between the different modes and

explore the whole sampling space. The distribution of interest, $\pi(\boldsymbol{x})$, is described in the formula below.

$$\pi(\mathbf{x}) = \frac{1}{2\pi\sigma^2} \sum_{i=1}^{20} w_i \exp\left(-\frac{1}{2\sigma^2}(\mathbf{x}-\mu_i)'(\mathbf{x}-\mu_i)\right)$$

where $\sigma = 0.1$, $w_1 = \ldots = w_{20} = 0.05$ and $\mu_1, \mu_2, \ldots, \mu_{20}$ are known.

The simplex is defined by three vertices in two-dimensional space. The reflection coefficient value is simulated from a $N(0, 4^2)$, and the expansion/contraction coefficient is drawn from a $\text{Ga}(1, 2)$. The Metropolis-Hastings move uses a Normal proposal distribution, $N(0, 2^2)$, to update each of the components of the simplex. Moreover, the Real-parameter Evolutionary Monte Carlo algorithm uses a $N(0, 0.1^2)$ proposal distribution in the mutation operator and the Metropolis-Hastings algorithm in the snooker crossover operator. For the Metropolis-Hastings algorithm, a $N(0, 2^2)$ proposal distribution is considered. The mutation probability is 0.20. Furthermore, the tempered transitions method uses a $N(0, 1)$ transition probability and 100 iterations under each temperature.

In the Real-parameter Evolutionary Monte Carlo algorithm and tempered transitions method, only one out of the ten is the temperature of interest. The simplex sampler runs for $100,000$ iterations regarding the first $10,000$ as burn in. In order to make the algorithms comparable, the total number of iterations for each algorithm was kept constant accounting for the different number of iterations per sweep of each algorithm.

The Metropolis-Hastings algorithm and the simplex sampler seem to get trapped under the modes without being able to escape and explore the whole sampling space. However, the tempered transitions method and the Real-parameter

Evolutionary Monte Carlo algorithm seem to be able to traverse the whole sampling space and visit all modes without getting trapped under a single one, see Figure 2.11.

This is an extreme example where there are high and low probability areas connected by essentially zero probability mass, it is perhaps unrealistic. Nevertheless, in the case of a single Markov chain like Metropolis-Hastings algorithm, it is easy to get trapped under a local maxima. On the other hand, population MCMC techniques do not get trapped so easily because they run several MCMC chains in parallel allowing them to interact in order to propose the new state of the Markov chain. But then, the simplex sampler though is based on ideas of a local optimization algorithm and for this reason, it does not escape easily from one maxima to search for another one. Nevertheless, the performance of the algorithm is good for highly correlated distributions.

2.5 Discussion

The simplex sampler generally performed better than the Metropolis-Hastings algorithm in the considered examples. In general, population MCMC methods are more powerful than standard MCMC ones that use only one chain to sample from the distribution of interest, especially when the target is multi-modal or the variables are highly correlated. Therefore it could be postulated that population MCMC techniques are in general more powerful because they use more than one chain to sample from the target distribution using at every sweep, the current state to define the proposed one.

If the variables are highly correlated then the simplex sampler performs quite

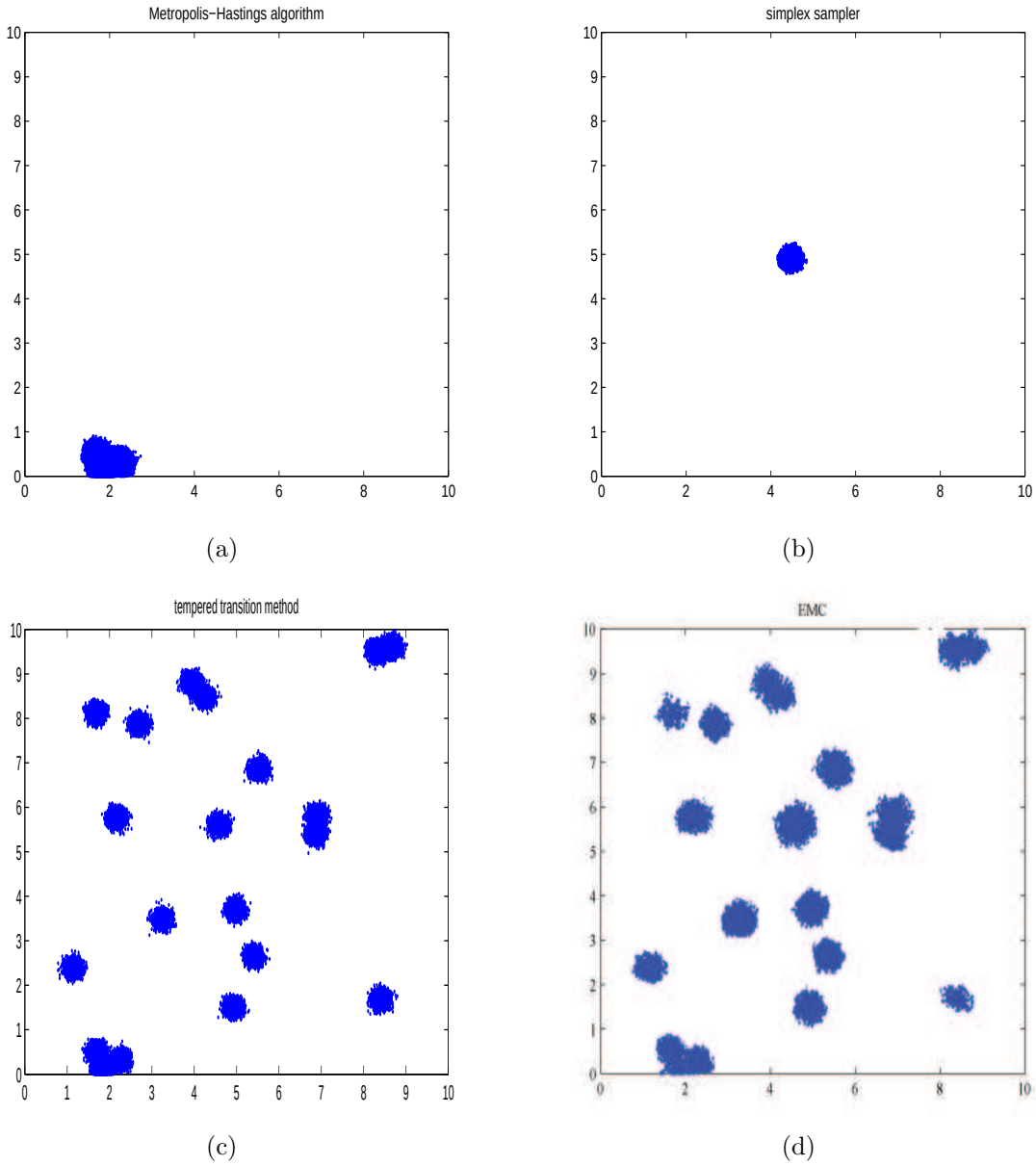


Figure 2.11: (a) Metropolis-Hastings algorithm, (b) simplex sampler, (c) tempered transitions method and (d) Real-parameter Evolutionary Monte Carlo algorithm for the mixture of the twenty Normal distributions.

well. In most cases, it gives more accurate and less biased estimations to the parameter values. Furthermore, for most population MCMC algorithm, there is no objective way to decide the number of chains to be used to sample from the distribution of interest in order to promote mixing. For this reason, the choice is left to the skills of the user. In the simplex sampler case, if the target distribution is defined in $\mathbb{R}^d$, then $d+1$ vertices are used to define the simplex sampler, and hence $d+1$ Markov chains to be run.

On the other hand, the Real-parameter Evolutionary Monte Carlo algorithm and tempered transitions methods use a temperature ladder to define the intermediate distributions to sample from. Thus, they move from "cooler" distributions to the target one without getting trapped under a local maxima. Then again, all these intermediate distributions are disregarded at the end of each sweep using a lot of computational power. There is also another issue with their implementation. In particular, the number of intermediate distributions should be consider for sampling from the distribution of interest, which has to be tuned by the user depending on his skills. Then again, the simplex sampler stores and uses every chain that is considered to sample from the distribution of interest since it does not consider any intermediate tempered distributions. Thus, no computational power is wasted and the sampler performs efficiently especially when there is strong correlation between the variables.

However, the simplex sampler does not sample efficiently from multi-modal target distributions. In this case, the simplex sampler gets trapped under a single local maximum without being able to escape and search for another one or if it doesn't get trapped, it does not sample with the correct frequency under each mode. The problem is that the simplex method of Nelder and Mead (1965) is

a local optimization technique and for this reason, the simplex sampler tends to spend more time sampling under a single local maximum unwilling to search for another one.

In conclusion, the simplex sampler performs well when the parameters are highly correlated since it gives more accurate and less biased estimates. It also has the advantage of using all iterations without disregarding any of them. On the other hand, when the distribution of interest is multi-modal then for the reasons described above, it tends to get trapped under a single local maxima or not sampling under each mode with the correct frequency. Therefore, Chapter 3 introduces a new population MCMC sampler, the tempered simplex sampler, where a tempering ladder is used to overcome the multi-modal problem and allow the sampler to move efficiently between the different modes. Hence, instead of having only one Markov chain under each tempering ladder, several of them are used. The Markov chains run in parallel and interact with each other using the simplex sampler idea.

Chapter 3

Tempered simplex sampler

The simplex sampler introduced in Chapter 2 performs well when the parameters of the distribution of interest are strongly correlated. In these cases, it samples better than usual MCMC methods like the Metropolis-Hastings algorithm and also has an advantage on tempering techniques like the Real-parameter Evolutionary Monte Carlo algorithm and the tempered transitions method because it does not discard any of the Markov chains at intermediate temperatures and for this reason, it represents a more economical use of computational resources.

Nevertheless, when the target distribution is multi-modal, the simplex sampler fails to sample efficiently from the distribution of interest as it samples under a local maxima unwilling to move to another one. The problem is related to the fact that the simplex sampler is based on ideas of the simplex method of Nelder and Mead (1965) which is a local optimization technique. Therefore, it is not easy for the sampler to escape the currently visited local maximum and search for another one. However, techniques that use a tempering ladder like the Real-parameter Evolutionary Monte Carlo algorithm and tempered transitions

method tend not to get trapped under a single local maximum and they are able to explore the whole sample space because they sample from intermediate tempered distributions.

Therefore, the main idea of this chapter is to create a new population MCMC sampler, the tempered simplex sampler, using a tempering ladder that will sample efficiently from multi-modal target distributions. Usual tempering methods use a population of chains, one for each temperature to explore the target distribution. In the tempered simplex sampler case, a whole population of chains is considered under each temperature and not just a single one. Thus, a population of chains is exchanged under different temperatures. First, the chains interact by applying the simplex sampler idea to explore the distribution of interest under each temperature. Then, these populations of chains are exchanged under different temperatures. In the following section, the proposal mechanism of the tempered simplex sampler is introduced. Conclusions are made on its performance compared to the one of the Real-parameter Evolutionary Monte Carlo algorithm and the tempered transitions method using different multi-modal examples.

3.1 Proposal mechanism of the tempered simplex sampler

Assume that $\pi(\cdot)$ is a d dimensional distribution of interest and the current population is defined as $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ where each of the different sub populations $\boldsymbol{x}_i$ consists of m different vertices so that $\boldsymbol{x}_i = (\boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{im})$, with $\boldsymbol{x}_{ij} \in \mathbb{R}^d$ with $i = 1, \ldots, n$ and $j = 1, \ldots, m$. The temperature ladder is defined as

$T = (T_1, \ldots, T_n)$ with $T \in [0, 1]$ and each of the T_i being equally spaced from each other. Thus, each of the $\boldsymbol{x}_i$ is defined under temperature T_i so that the distribution of interest is

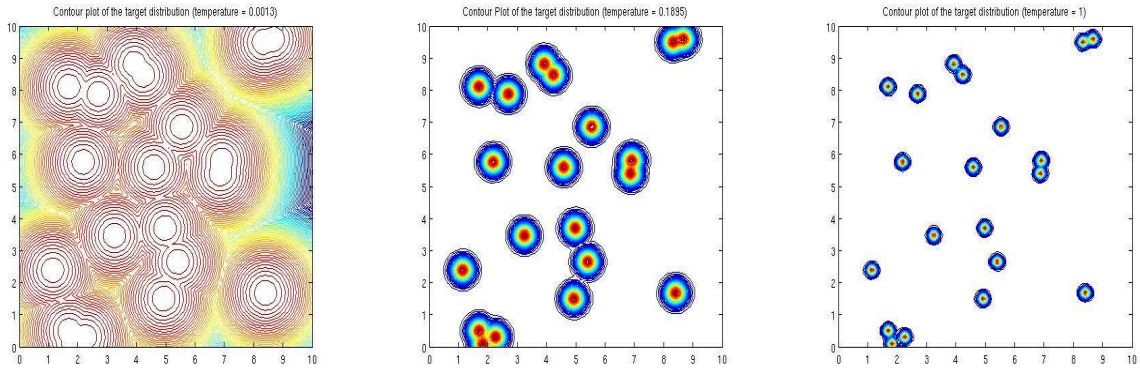
$$\pi(\boldsymbol{X}) = \prod_{i=1}^{n} \pi^{T_i}(\boldsymbol{x}_i).$$

and

$$\pi^{T_i}(\boldsymbol{x}_i) = \prod_{j=1}^{m} \pi^{T_i}(\boldsymbol{x}_{ij}).$$

The sampler traverses the whole sampling space without getting trapped under a local maxima thanks to the tempering ladder. The main idea is to start sampling from distributions that are similar to a uniform distribution and then, slowly increasing the tempering parameter values and eventually sample from the distribution of interest, see Figure 3.1. At the end of each sweep, only the values that correspond to the distribution of interest are stored and the rest of them are discarded. This may be computationally expensive but, it produces a representative sample from the distribution of interest because the tempering parameter allows the chains to traverse the whole sampling space without getting trapped under a local maxima. Then again, it is not necessary to discard all of the sampled values when the temperature is not equal to 1, importance sampling can be used instead Gramacy et al. (2009).

The tempered simplex sampler consists of two steps. First, each of the populations $\boldsymbol{x}_i$ is updated under each tempering ladder T_i using the simplex sampler. Hence, there are m different chains that run in parallel and interact with each other under temperature T_i using the simplex sampler to propose the new state. Then, the different populations say $\boldsymbol{x}_i$ and $\boldsymbol{x}_l$ are chosen to be exchanged under



(a) (b) (c)

Figure 3.1: (a) Target distribution at temperature 0.0013. (b) Target distribution at temperature 0.1895. (c) The actual target distribution at temperature 1.

different temperatures say T_i and T_l so that the chains also interact under different temperatures. The two steps of the sampler are described in detail below.

3.1.1 First step: Updating the simplex within each temperature.

In the first step of the tempered simplex sampler, the chains within each temperature interact with each other. The simplex sampler is applied to each sub population $\boldsymbol{x}_i$ under temperature T_i. Thus, under each temperature the simplex consists of m different vertices in d dimensional space and the distribution of interest is

$$\pi^{T_i}(\boldsymbol{x}_i) = \prod_{j=1}^{m} \pi^{T_i}(\boldsymbol{x}_{ij}).$$

Three different types of move are carried out, the reflection, expansion/contraction move and a Metropolis-Hastings update. Similar to the simplex sampler case, the

core of the tempered simplex sampler is the reflection move. The other moves are used to update the shape of the simplex. Hence, at each sweep, one can carry out all of the designed moves in turn or use a distribution of them.

3.1.1.1 Reflection move

In the reflection move, the current vertex, $\boldsymbol{x}_{ic}$ given in (3.1) is chosen with probability $\ell_i(c)$ which is inversely proportional to its target probability. Hence, the probability $\ell_i(j)$ is evaluated as

$$\ell_i(j) = \frac{P(\boldsymbol{x}_{ij})}{\sum_{z=1}^{m} P(\boldsymbol{x}_{iz})}, \tag{3.1}$$

where $P(\boldsymbol{x}_{ij})$ is defined as

$$P(\boldsymbol{x}_{ij}) = 1 - \frac{\pi^{T_i}(\boldsymbol{x}_{ij})}{\sum_{z=1}^{m} \pi^{T_i}(\boldsymbol{x}_{iz})}.$$

Then, the euclidean distance between the current vertex $\boldsymbol{x}_{ic}$ and each vertex $\boldsymbol{x}_{ij}$ for $j \neq c$ is calculated. The two vertices that are relatively close to the current one define the current simplex. Hence, the vertices are chosen with probability

$$\phi_{ri}(j) = \frac{\sum_{z=1}^{m} d(\boldsymbol{x}_{iz}, \boldsymbol{x}_{ic}) - d(\boldsymbol{x}_{ij}, \boldsymbol{x}_{ic})}{\sum_{z=1}^{m} d(\boldsymbol{x}_{iz}, \boldsymbol{x}_{ic})}$$

where $d(\boldsymbol{x}_{ij}, \boldsymbol{x}_{ic}) = \sqrt{(\boldsymbol{x}_{ij} - \boldsymbol{x}_{ic})^2} = |\boldsymbol{x}_{ij} - \boldsymbol{x}_{ic}|$ is the euclidean distance between the current vertex, $\boldsymbol{x}_{ic}$, and each vertex $\boldsymbol{x}_{ij}$. Assume now that the vertices $\boldsymbol{x}_{ik}$ and $\boldsymbol{x}_{ir}$ are chosen with probability $\phi_{ri}(k)$ and $\phi_{ri}(r)$ respectively. Then, the current

simplex is defined as $\boldsymbol{x} = \{\boldsymbol{x}_{ic}, \boldsymbol{x}_{ik}, \boldsymbol{x}_{ir}\}$ and the reflection move is carried out as described in section 2.3.1. Hence, the centroid of the current simplex is evaluated and the candidate vertex $\boldsymbol{y}_{ic}$ is proposed through the line joining the current vertex and the centroid of the simplex. The proposed vertex $\boldsymbol{y}_{ic}$ is accepted with probability

$$\alpha_{ref}\left(\boldsymbol{x}_{ic}, \boldsymbol{y}_{ic}\right) = \min\left\{1, \frac{\pi^{T_i}(\boldsymbol{y}_{ic})}{\pi^{T_i}(\boldsymbol{x}_{ic})} \frac{q(\boldsymbol{y}_{ic}, \boldsymbol{x}_{ic})}{q(\boldsymbol{x}_{ic}, \boldsymbol{y}_{ic})}\right\}.$$

The proposal distribution $q(\boldsymbol{x}_{ic}, \boldsymbol{y}_{ic})$ is evaluated as the probability of choosing vertex $\boldsymbol{x}_{ic}$ as the current one and proposing $\boldsymbol{y}_{ic}$ to be the candidate one. Hence, $q(\boldsymbol{x}_{ic}, \boldsymbol{y}_{ic}) = \ell_i(c)\phi_{ri}(k)\phi_{ri}(r)p_\alpha(\alpha)$ where $p_\alpha(\alpha)$ is the probability of choosing the value α for the reflection coefficient from the proposal distribution $p_\alpha(\cdot)$.

If the candidate component is accepted with probability $\alpha_{ref}\left(\boldsymbol{x}_{ic}, \boldsymbol{y}_{ic}\right)$ then the current vertex $\boldsymbol{x}_{ic}$ is replaced by $\boldsymbol{y}_{ic}$ otherwise, the proposed move is rejected with probability $1 - \alpha_{ref}\left(\boldsymbol{x}_{ic}, \boldsymbol{y}_{ic}\right)$ and the Markov chain stays where it currently is, see Figure 3.2.

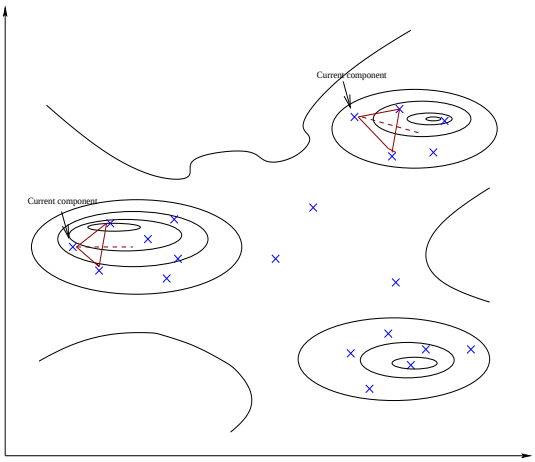


Figure 3.2: Reflection move of the tempered simplex sampler.

In addition, detailed balance is preserved throughout each step and for this reason, the Markov chain should be able to move from the proposed state back to the current one. Hence, the proposal distribution $q(\boldsymbol{y}_{ic}, \boldsymbol{x}_{ic})$ is defined as the probability of moving from the proposed vertex, $\boldsymbol{y}_{ic}$, to the current one, $\boldsymbol{x}_{ic}$. It is evaluated as $q(\boldsymbol{y}_{ic}, \boldsymbol{x}_{ic}) = \ell_i^*(c)\phi_{ri}^*(k)\phi_{ri}^*(r)p_\alpha(\alpha^*)$ where $\ell_i^*(c)$ is the probability of choosing $\boldsymbol{y}_{ic}$ as the current vertex in the proposed state, $\phi_{ri}^*(k)$ and $\phi_{ri}^*(r)$ are the probabilities of choosing vertices $\boldsymbol{x}_{ik}$ and $\boldsymbol{x}_{ir}$ respectively to define the simplex for the reverse move. Hence, the probability of choosing the j^{th} vertex in the proposed state is

$$\phi_{ri}^*(j) = \frac{\sum_{z=1}^{m} d(\boldsymbol{x}_{iz}, \boldsymbol{y}_{ic}) - d(\boldsymbol{x}_{ij}, \boldsymbol{y}_{ic})}{\sum_{z=1}^{m} d(\boldsymbol{x}_{iz}, \boldsymbol{y}_{ic})}.$$

Finally, $p_\alpha(\alpha^*)$ is the probability of choosing the α^* value from the $p_\alpha(\cdot)$ distribution which ensures that the chain moves from vertex $\boldsymbol{y}_{ic}$ back to vertex $\boldsymbol{x}_{ic}$.

When the tempered simplex sampler is applied, it depends on the skills of the user to decide how many vertices are going to be updated under temperature T_i. For instance, one may choose to update one, more than one or all of the vertices of the $\boldsymbol{x}_i$ population. It is obvious that the more vertices are updated the better the mixing of the sampler will be considering also the computational power that is used.

3.1.1.2 Expansion/Contraction move

It is possible for the vertices of the simplex to be far apart or too close to each other, which results in inefficiently sampling because the sampler loses its local ability. For this reason, every time a vertex is chosen another two are selected

to define the current simplex depending on how far away or how close they are from the chosen one.

For instance, suppose that one of the $\boldsymbol{x}_{ij}$ vertices is randomly selected, say $\boldsymbol{x}_{il}$, then an expansion or contraction move will be applied. If an expansion move is carried out then two more vertices, say $\boldsymbol{x}_{ik}$ and $\boldsymbol{x}_{ir}$ that are relatively close to $\boldsymbol{x}_{il}$ are selected. The vertices are chosen with probability

$$\phi_i(j) = \frac{\sum_{z=1}^{m} d(\boldsymbol{x}_{iz}, \boldsymbol{x}_{il}) - d(\boldsymbol{x}_{ij}, \boldsymbol{x}_{il})}{\sum_{z=1}^{m} d(\boldsymbol{x}_{iz}, \boldsymbol{x}_{il})},$$

for $j = l$ or $j = r$. On the other hand, if a contraction move is carried out then the vertices, say $\boldsymbol{x}_{ik}$ and $\boldsymbol{x}_{ir}$ that are relatively far away from $\boldsymbol{x}_{il}$ are selected with probability

$$\phi_i(j) = \frac{d(\boldsymbol{x}_{ij}, \boldsymbol{x}_{il})}{\sum_{z=1}^{m} d(\boldsymbol{x}_{iz}, \boldsymbol{x}_{il})},$$

for $j = k$ or $j = l$. Hence, assume that the current simplex is defined as $\boldsymbol{x} = \{\boldsymbol{x}_{ic}, \boldsymbol{x}_{ik}, \boldsymbol{x}_{ir}\}$. Then, the proposed expansion or contraction move is implemented as described in section 2.3.2. The proposed move is accepted with probability

$$\alpha_{exp/con}(\boldsymbol{x}, \boldsymbol{y}) = \min\left\{1, \frac{\pi^{T_i}(\boldsymbol{y}) q(\boldsymbol{y}, \boldsymbol{x})}{\pi^{T_i}(\boldsymbol{x}) q(\boldsymbol{x}, \boldsymbol{y})}\right\}, \tag{3.2}$$

where $q(\boldsymbol{x}, \boldsymbol{y})$ is the probability of moving from the current simplex, $\boldsymbol{x}$, to the proposed one, $\boldsymbol{y}$. Hence, the proposal probability $q(\boldsymbol{x}, \boldsymbol{y}) = \nu_i(c)\phi_i(k)\phi_i(r)p_\beta(\beta)$ where $\nu_i(c)$ is the probability of selecting vertex $\boldsymbol{x}_{ic}$, $\phi_i(k)$ and $\phi_i(r)$ are the probabilities of choosing $\boldsymbol{x}_{ik}$ and $\boldsymbol{x}_{ir}$ respectively and $p_\beta(\beta)$ is the probability

of choosing the contraction/expansion coefficient β from the $p_\beta(\cdot)$ distribution. Note that the probability of carrying out an expansion or contraction move is 0.5 and consequently, it disappears in the acceptance probability given in (3.2).

The proposal probability $q(\boldsymbol{y}, \boldsymbol{x})$ ensures that the chain is able to move from the proposed state back to the current one and for this reason, detailed balance is preserved at each step of the expansion and contraction moves. Hence, the proposal probability $q(\boldsymbol{y}, \boldsymbol{x}) = \nu_i^*(l)\phi_i^*(k)\phi_i^*(r)p_\beta(\beta^*)$, where $\nu_i^*(l)$ is the probability of choosing vertex $\boldsymbol{y}_{il}$ from the proposed population, $\phi_i^*(k)$ and $\phi_i^*(r)$ are the probabilities of selecting vertices $\boldsymbol{y}_{ik}$ and $\boldsymbol{y}_{ir}$ respectively, after choosing vertex $\boldsymbol{y}_{il}$ to define the simplex in the proposed state. Finally, β^* is the contraction/expansion coefficient value for the reverse move.

3.1.1.3 Metropolis-Hastings update

Here, every vertex of the $\boldsymbol{x}_i$ population is updated under each temperature T_i using a Metropolis-Hastings update. Hence, each of the components of the simplex is generated as $\boldsymbol{y}_{ij} \sim N(\boldsymbol{x}_{ij}, \sigma^2)$ under temperature T_i where σ^2 is the proposal variance. The acceptance probability for the Metropolis-Hastings update is given in the following formula (3.3)

$$\alpha_{mh}\left(\boldsymbol{x}_{ij}, \boldsymbol{y}_{ij}\right) = \min\left\{1, \frac{\pi^{T_i}(\boldsymbol{y}_{ij})}{\pi^{T_i}(\boldsymbol{x}_{ij})}\right\}. \tag{3.3}$$

3.1.2 Second step: Exchange simplexes between different temperatures.

Different simplexes are exchanged under different temperatures in order to promote mixing and avoid getting trapped under a single mode. Assume that the population of Markov chains $\boldsymbol{x}_i$ under temperature T_i is chosen to be exchanged with another population $\boldsymbol{x}_l$ under temperature T_l. Then, set the proposed population $\boldsymbol{y}_i = \boldsymbol{x}_l$ under temperature T_i and $\boldsymbol{y}_l = \boldsymbol{x}_i$ under temperature T_l. Thus, the proposed move is accepted with probability

$$\alpha_e = \min\left\{1, \frac{\pi(\boldsymbol{x}_l)^{T_i}}{\pi(\boldsymbol{x}_i)^{T_i}} \frac{\pi(\boldsymbol{x}_i)^{T_l}}{\pi(\boldsymbol{x}_l)^{T_l}}\right\}$$

or equivalently,

$$\alpha_e = \min\left\{1, \left(\frac{\pi(\boldsymbol{x}_i)}{\pi(\boldsymbol{x}_l)}\right)^{T_l - T_i}\right\}. \tag{3.4}$$

If the proposed move is accepted with probability α_e given in (3.4) then the next state is $\boldsymbol{y}_i$ under temperature T_i and $\boldsymbol{y}_l$ under temperature T_l. Otherwise, it is rejected with probability $1 - \alpha_e$ and the chain stays where it currently is, i.e at the next state of the chain, the population $\boldsymbol{x}_i$ is under temperature T_i and the population $\boldsymbol{x}_l$ is under temperature T_l.

The populations are exchanged by randomly selecting i from $1, \ldots, n$ and then setting $l = i \pm 1$ with probability $p(j = i + 1) = p(j = i - 1) = 0.5$ for $i = 2, \ldots, n-1$, $p(j = 2) = 1$ for $i = 1$ and $p(j = n - 1) = 1$ for $i = n$. One can choose each of the $\boldsymbol{x}_i$ populations in turn to be exchanged or randomly select a few of them.

When the exchange move is applied, it works best if the chosen populations

have similar temperature values because there are more chances for the proposed move to be accepted, i.e if $T_i \approx T_j$ then $\left(\frac{\pi(\boldsymbol{x}_i)}{\pi(\boldsymbol{x}_l)}\right)^{T_l - T_i} \approx 1$. This characteristic is important in the efficiency of the sampler.

The algorithmic version of the tempered simplex sampler is given Algorithm in 4.

Algorithm 4 Tempered simplex sampler

At the current state a population $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ is defined under a temperature ladder $T = (T_1, \ldots, T_n)$ with $T_1 = 0 \leq T_2 \leq \ldots \leq T_n = 1$.

STEP A: Each population $\boldsymbol{x}_i = (\boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{im})$, with $\boldsymbol{x}_{ij} \in \mathbb{R}^d$, explores the target distribution within temperature T_i.

Each of the different chains in the population, $\boldsymbol{x}_i$, interacts with each other using the simplex sampler.

(i) Apply a reflection move.

(ii) Apply an expansion/contraction move.

(iii) Apply a Metropolis-Hastings move.

STEP B: Exchange the population $\boldsymbol{x}_i$ under temperature T_i with another population $\boldsymbol{x}_l$ under temperature T_l.

1. Set $\boldsymbol{y}_i = \boldsymbol{x}_l$ under temperature T_i and $\boldsymbol{y}_l = \boldsymbol{x}_i$ under temperature T_l.
2. Estimate the acceptance probability probability α_e given in (3.4).
3. Accept the proposed move with probability α_e and the new population is $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{y}_i, \ldots, \boldsymbol{y}_l, \ldots, \boldsymbol{x}_n)$ or reject it with probability $1 - \alpha_e$ and the chain stays where it currently is, so that $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_l, \ldots, \boldsymbol{x}_n)$.

* Store only the population that corresponds to the distribution of interest at the end of each sweep.

* Repeat the procedure for an sufficiently long time.

3.2 Examples

In this section, only multi-modal target distributions are considered because of their characteristics such as large areas of zero probability. Hence, it is a challenge for the sampler to jump between the modes. The performance of the tempered simplex sampler is compared to the Real-parameter Evolutionary Monte Carlo algorithm (Liang and Wong (2001)) and tempered transitions method (Neal (1996)) through, the integrated autocorrelation (IA) and the efficient sample size (ESS). The integrated autocorrelation is calculated using the method proposed by Sokal (1997). Then, the efficient sample size is evaluated as $N/(\mathrm{IA}\mathtt{t})$, where N is the sample size generated from the MCMC algorithm and $\mathtt{t}$ is the computation time to execute the algorithm. Thus, high values of the efficient sample size give a measure of the performance of the algorithm which accounts for both the computational time and the number of MCMC draws. Moreover, the coverage probability under each mode is estimated and compared to the true one.

In the following examples, the algorithms run for $1,000,000$ iterations in total using a burn in of $20,000$ iterations. The computational cost was kept equal. The parameter values of the proposal distribution are chosen based on the acceptance probabilities. Here, the tempered simplex sampler uses ten populations under ten different temperatures that are equally spaced in $[0, 1]$. Likewise, the Real-parameter Evolutionary Monte Carlo algorithm and the tempered transitions method use ten different temperatures equally spaced in $[0, 1]$.

3.2.1 Mixture of twenty Normal distributions

In the first example, the distribution of interest is a mixture of twenty Normal distributions described by high and low probability areas. This example was also explored in section 2.4.5. Here, the modes are well separated from each other as shown in Figure 3.3(a). The challenge for the algorithms is to visit each mode and traverse the whole sampling space while they also spend the right amount of time sampling under each mode.

For the tempered simplex sampler, the simplex sampler is implemented by drawing the reflection coefficient value from a Normal distribution $N(0, 4^2)$, the expansion/contraction coefficient value is simulated from a Gamma distribution $\text{Ga}(1, 2)$ and for the Metropolis-Hastings update, a Normal proposal distribution $N(0, 0.1^2)$ is used. Moreover, the Real-parameter Evolutionary Monte Carlo algorithm uses a $N(0, 0.1^2)$ as the proposal distribution in the mutation operator and the Metropolis-Hastings algorithm in the snooker crossover operator. The Metropolis-Hastings algorithm uses a $N(0, 0.1^2)$. Here, the mutation probability is 0.20. Furthermore, for the tempered transitions method a $N(0, 1)$ proposal distribution is used considering 100 iterations under each temperature.

As it turns out, all the different methods explore the entire sampling space without getting trapped under a single maxima, see Figure 3.3. Hence, the tempering ladder enables the methods to sample efficiently from multi-modal target distributions. Then again, no significant difference can be seen between the different methods following Figure 3.3. For this reason, the proportion of realisations from the Markov chain under each mode is counted and compared to the target one to test whether the algorithms sample with the correct frequency

under each mode.

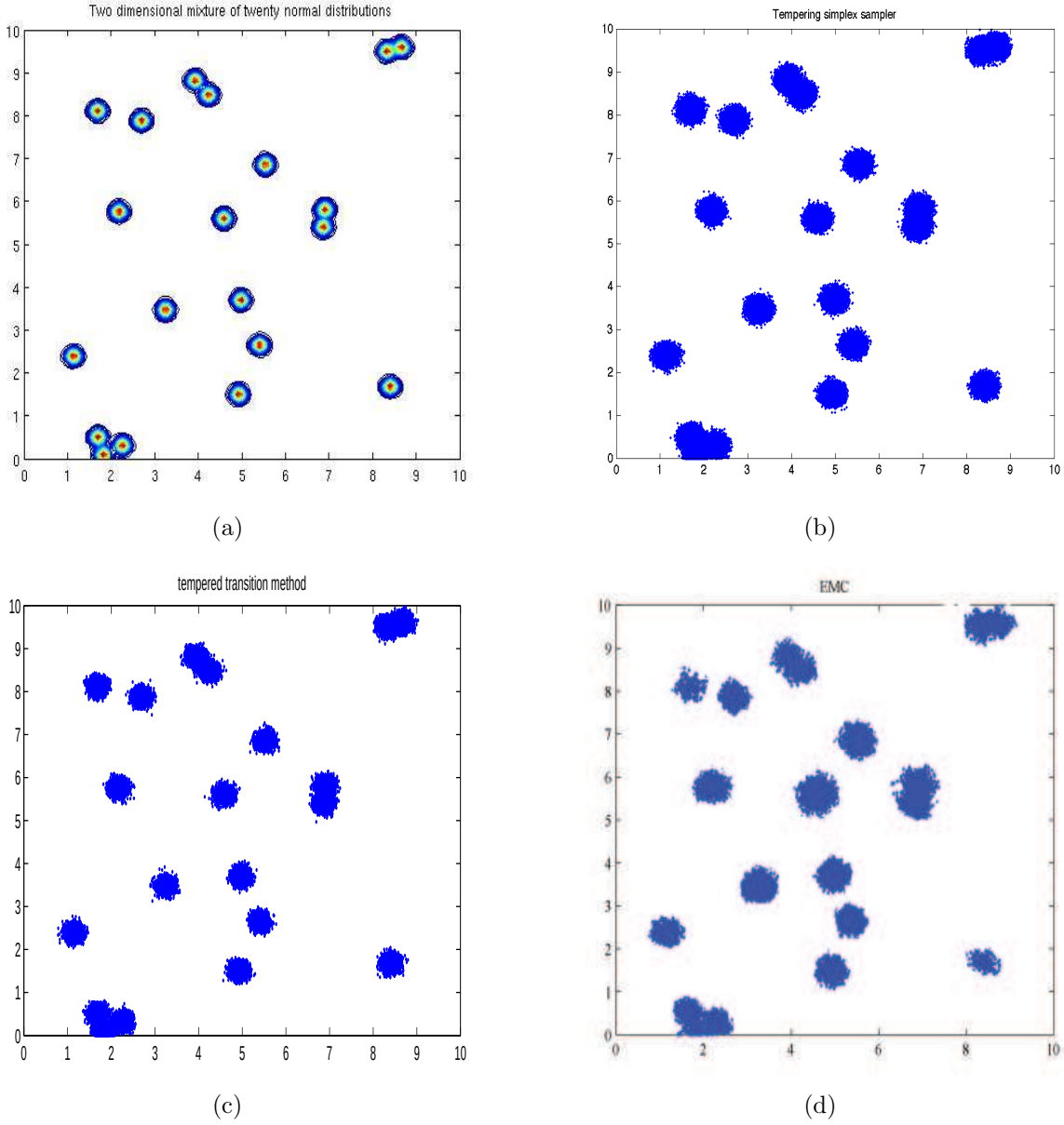


Figure 3.3: (a)Mixture of twenty Normal distributions (b) tempered simplex sampler (c) tempered transitions method and (d) Real-parameter Evolutionary Monte Carlo algorithm for the mixture of twenty Normal distributions.

Table 3.1 indicates which coordinates define the chosen modes that are presented in Table 3.2. The coverage of these modes is presented in Table 3.2. It

Modes	A	B	C	D
x-axis $\in$	[8,9]	[1,3]	[0,2]	[8,9]
y-axis $\in$	[1,2]	[0,1]	[2,3]	[9,10]

Table 3.1: Evaluation of the modes presented in Table 3.2

seems that the tempered simplex sampler and the tempered transitions method perform better than the Real-parameter Evolutionary Monte Carlo algorithm. The tempered simplex sampler gives slightly better proportions than the tempered transitions method. However, the Real-parameter Evolutionary Monte Carlo algorithm has the worst proportion compared to the other methods.

Modes	A	B	C	D
true	5%	15%	5%	10%
tempered simplex sampler	4.81%	14.18%	5.03%	10.57 %
EMC	1.17%	7.82%	5.33%	8.24%
tempered transitions	2.48%	16.48%	6.88%	3.68%

Table 3.2: Percentage of observations under different modes using the different methods.

The integrated autocorrelation and the efficient sample size are presented in Table 3.3. The tempered simplex sampler gives the smallest integrated autocorrelation while the Real-parameter Evolutionary Monte Carlo algorithm and tempered transitions method have the biggest integrated autocorrelation. The tempered simplex sampler on the other hand, has the biggest efficient sample size. Hence, the tempered simplex sampler is more efficient compared to the other methods.

Methods	IA	ESS	N	time
tempered simplex sampler	60	17	32,260	31.63
EMC	90	5	28,572	63,49
tempered transitions	67	6	50,000	124,38

Table 3.3: Integrated autocorrelation, efficient sample size, number of iterations and time for the mixture of twenty normal distributions using each of the different methods.

The total acceptance probability for the tempered simplex sampler is 0.3923, for the Real-parameter Evolutionary Monte Carlo algorithm is 0.4059 and for the tempered transitions method is 0.6015. The acceptance probabilities for the between temperature moves are 0.6143 for the tempered simplex sampler, 0.5378 for the Real-parameter Evolutionary Monte Carlo algorithm and 0.730 for the tempered transitions method.

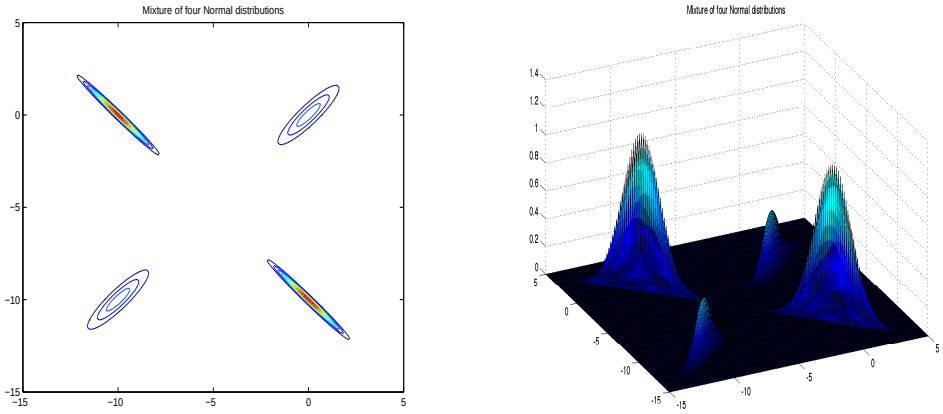
3.2.2 Mixture of four bivariate Normal distributions

The next target is a mixture of four bivariate Normal distributions defined as

$$\pi(\boldsymbol{x}) = w_1 N(\boldsymbol{x}; \mu_1, \Sigma_1) + w_2 N(\boldsymbol{x}; \mu_2, \Sigma_1) + w_3 N(\boldsymbol{x}; \mu_3, \Sigma_2) + w_4 N(\boldsymbol{x}; \mu_4, \Sigma_2)$$

where $\mathbf{x} \in \mathbb{R}^2$, $w_i = 0.25$, for $i = 1, \ldots, 4$, $\mu_1 = [0, 0]$, $\mu_2 = [-10, -10]$, $\mu_3 = [0, -10]$, $\mu_4 = [-10, 0]$, $\Sigma_1 = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix}$ and $\Sigma_2 = \begin{bmatrix} 1 & -0.99 \\ -0.99 & 1 \end{bmatrix}$.

The contour plot and density plot of the target distribution are presented in Figure 3.4. The distribution of interest consists of high and low probability areas with highly correlated components. Hence, the aim is to find out how efficiently the algorithms sample under these high correlated modes.



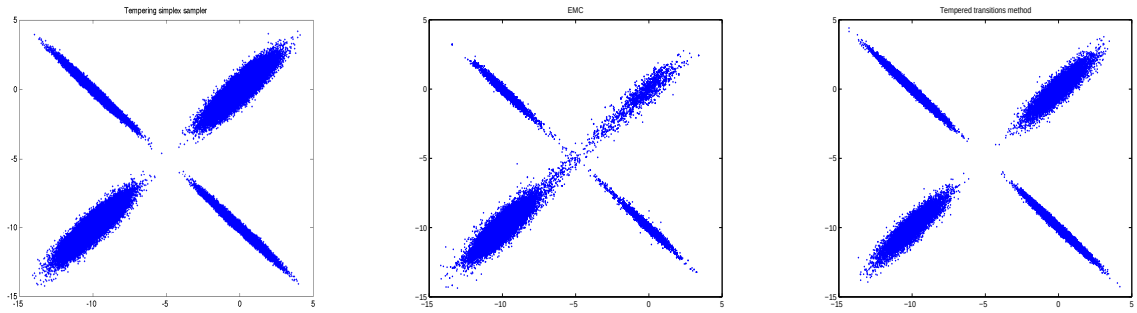
(a) (b)

Figure 3.4: (a) Contour plot and (b) density plot for the mixture of the four Normal distributions.

The tempered simplex sampler is implemented by drawing the reflection coefficient value from a $N(0, 4^2)$, the expansion/contraction coefficient value is simulated from a Ga$(1, 2)$. The Metropolis-Hastings update uses a $N(0, 1)$. For the Real-parameter Evolutionary Monte Carlo algorithm, a $N(0, 1)$ is used for the the mutation operator and in the snooker crossover operator. The mutation probability is 0.20. For the tempered transitions method, the transition probability is $N(0, 1)$ using 100 different iterations under each temperature.

The different techniques visit all modes and they sample efficiently from the target distribution following Figure 3.5. However, the tempered simplex sampler samples more efficiently under the modes compared to the other algorithms. It seems that it may be worthwhile to use a population of chains under each temperature to sample from the target distribution. In the tempered simplex sampler case, the sampler uses the good mixing of the simplex sampler for high

correlated targets and also the ability to sample from the distribution of interest using different intermediate tempered distributions for multi-modal targets.



(a) (b) (c)

Figure 3.5: (a)Tempered simplex sampler, (b) Real-parameter Evolutionary Monte Carlo algorithm and (c) tempered transitions method for the mixture of four Normal distributions.

In order to quantify the differences between the techniques, the coverage under each mode is estimated. Each of the modes is selected and the percentage of observations under them is evaluated. Table 3.4 presents the coordinates that are used to define the modes in Table 3.5. It seems that the tempered simplex sampler performs better than the other methods and samples more efficiently under the modes following Table 3.5.

Modes	A	B	C	D
x-axis $\in$	[-5,5]	[-15,-5]	[-15,-5]	[-5,5]
y-axis $\in$	[-15,-5]	[-15,-5]	[-5,5]	[-5,5]

Table 3.4: Evaluation of the modes presented in Table 3.5.

The tempered simplex sampler has the smallest integrated autocorrelation and the highest efficient sample size following Table 3.6. Therefore, it seems that

Modes	A	B	C	D
target	25%	25%	25%	25%
tempered simplex sampler	26.68%	22.72%	26.30%	24.31 %
EMC	28.73%	51.34%	14.41%	5.52%
tempered transitions	32.67%	20.44%	24.36%	22.54%

Table 3.5: Percentage of observations under each mode for the mixture of four Normal distributions using the different methods.

the tempered simplex sampler is sampling more efficiently than the other two methods.

Methods	IA	ESS	N	time
tempered simplex sampler	41	24	32,260	32,78
EMC	66	6	28,572	72,15
tempered transitions	54	7	50,000	132,27

Table 3.6: Integrated autocorrelation, efficient sample size, number of iterations and time for the mixture of four normal distributions using each of the different methods.

The acceptance probability for the tempered simplex sampler is 0.3845, for the Real-parameter Evolutionary Monte Carlo algorithm is 0.7823 and for the tempered transitions method is 0.8763 in total.

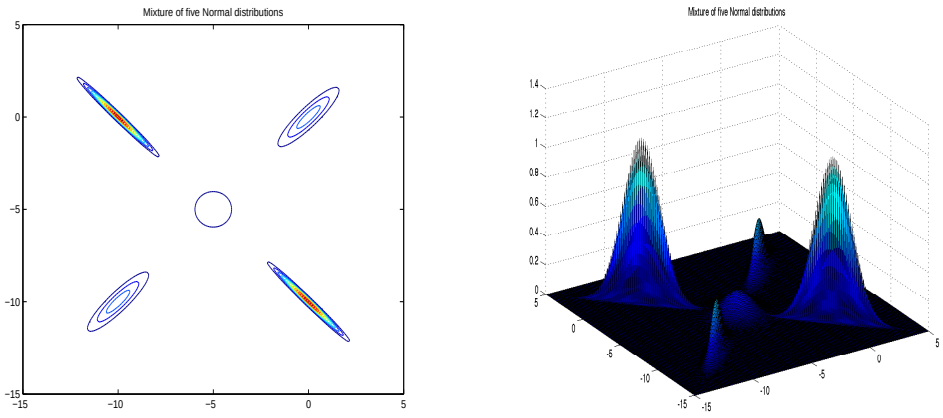
3.2.3 Mixture of five bivariate Normal distributions

Here, a mixture of five bivariate Normal distributions is considered. The distribution of interest is given in the following formula.

$$\pi(\boldsymbol{x}) = w_1 N(\boldsymbol{x}; \mu_1, \Sigma_1) + w_2 N(\boldsymbol{x}; \mu_2, \Sigma_2) + w_3 N(\boldsymbol{x}; \mu_3, \Sigma_1) + w_4 N(\boldsymbol{x}; \mu_4, \Sigma_2) + w_5 N(\boldsymbol{x}; \mu_5, \Sigma)$$

for $\mathbf{x} \in \mathbb{R}^2$, where $w_i = 0.20$, for $i = 1, \ldots, 5$, $\mu_1 = [0, 0]$, $\mu_2 = [-10, -10]$, $\mu_3 = [0, -10]$, $\mu_4 = [-10, 0]$, $\mu_5 = [-5, 5]$, $\Sigma_1 = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix}$, $\Sigma_2 = \begin{bmatrix} 1 & -0.99 \\ -0.99 & 1 \end{bmatrix}$ and $\Sigma_3 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$.

The contour plot and density plot for the mixture of five Normal distributions is presented in Figure 3.6. There are high and low probability areas where the covariates under the four of the modes are highly correlated, while for one of them they are not so strongly correlated. Once more, the aim is to find out how well the algorithms sample under each of the modes and whether the different correlation structures can be picked up by the sampling methods.



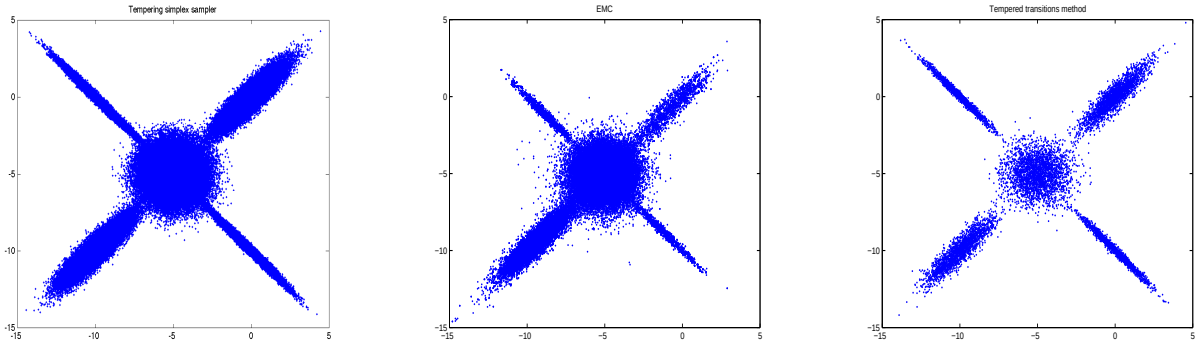
(a) (b)

Figure 3.6: (a) Contour plot and (b) density plot for the mixture of five dimensional Normal distributions.

Thus, the tempered simplex sampler is implemented by drawing the reflection coefficient value from a $N(0, 4^2)$ and the expansion/contraction coefficient

from a Ga$(1, 2)$ while the Metropolis-Hastings update uses a $N(0, 1)$. The Real-parameter Evolutionary Monte Carlo algorithm uses a $N(0, 1)$ for the the mutation operator. In the snooker crossover operator, the Metropolis-Hastings algorithm is used with a $N(0, 1)$ proposal distribution. The mutation probability is again 0.20. For the tempered transitions method, the transition probability is $N(0, 1)$ and 100 iterations are used under each temperature.

The different methods traverse the whole sample space and sample from each mode without getting trapped under a single one, see Figure 3.7. Nevertheless, it is not easy to draw conclusions based on the plots. For this reason, the coverage under the modes is estimated in order to compare the different methods.



(a) (b) (c)

Figure 3.7: (a) Tempered simplex sampler (b) Real-parameter Evolutionary Monte Carlo and (c) tempered transitions method for the mixture of the five bivariate Normal distributions.

Table 3.7 presents the way that each of the modes was evaluated before being presented in Table 3.8. Notice that mode E is estimated as the area which is not covered by the other modes. The tempered simplex sampler estimates the area under each mode very well. Then again, the Real-parameter Evolutionary Monte

Modes	A	B	C	D
x-axis $\in$	[-3,5]	[-15,-7]	[-15,-8]	[-3,2.5]
y-axis $\in$	[-2,5]	[-3,5]	[-15,8]	[-15,-7]

Table 3.7: Evaluation of the modes presented in Table 3.5.

Carlo algorithm does not estimate as well the area under each mode.

Modes	A	B	C	D	E
true	20%	20%	20%	20%	20%
tempering simplex sampler	18.90%	21.20%	17.99%	20.41 %	21.50 %
EMC	17.58%	16.21%	22.28%	17.41%	25.02%
tempered transitions	18.59%	20.06%	22.51%	18.37%	20.47%

Table 3.8: Percentage of observations under each mode for the mixture of five Normal distributions using the different methods.

The integrated autocorrelation and the efficient sample size are calculated and presented in Table 3.9. The tempered simplex sampler has the largest efficient sample size, and the Real-parameter Evolutionary Monte Carlo algorithm gives the smallest one. Thus, the tempered simplex sampler performs better than the other methods.

Methods	IA	ESS	N	time
tempering simplex sampler	77	13	32,260	32,23
EMC	87	5	28,572	65,68
tempered transitions	59	7	50,000	121,06

Table 3.9: Integrated autocorrelation, efficient sample size, number of iterations and time for the mixture of five normal distributions using each of the different methods.

In this example, the acceptance probability for the tempered simplex sampler

is 0.3453, for the Real-parameter Evolutionary Monte Carlo algorithm is 0.4162 and for the tempered transitions method is 0.3945 in total. Thus, the acceptance probabilities indicate that the algorithms have been tuned efficiently.

3.2.4 Mixture of two ten-dimensional Bimodal distributions

The last example is a mixture of two ten-dimensional bimodal distributions aiming to test the performance of the algorithms when a high dimensional target is considered. If the dimension is high then the algorithm becomes more computationally expensive. Thus, the distribution of interest is given in the following formula:

$$\pi(\boldsymbol{x}) = 1/3N_{10}(\boldsymbol{x}; \mathbf{0}, \mathbf{I_{10}}) + 2/3N_{10}(\boldsymbol{x}; \mathbf{5}, \mathbf{I_{10}})$$

where $\mathbf{0} = (0, \ldots, 0)$, $\mathbf{5} = (5, \ldots, 5)$ and $\mathbf{I_{10}}$ is the identity matrix. In this case, the modes are well separated, see Figure 3.8(a).

Hence, the tempered simplex sampler is implemented by drawing the reflection coefficient value from a $N(0, 4^2)$ and the expansion/contraction coefficient from a $\text{Ga}(1, 2)$ while the Metropolis-Hastings update uses a $N(0, 1)$. The Real-parameter Evolutionary Monte Carlo algorithm uses a $N(0, 1)$ for the the mutation operator and the Metropolis-Hastings algorithm for the snooker crossover operator. The Metropolis-Hastings algorithm is used with a $N(0, 1)$ proposal distribution. Here, the mutation probability is 0.20. In the case of the tempered transitions method, a $N(0, 1)$ proposal distribution is used and 100 iterations.

In Figure 3.8, a randomly selected variable is presented because the rest of the variables are similar to the presented ones. It seems that the algorithms

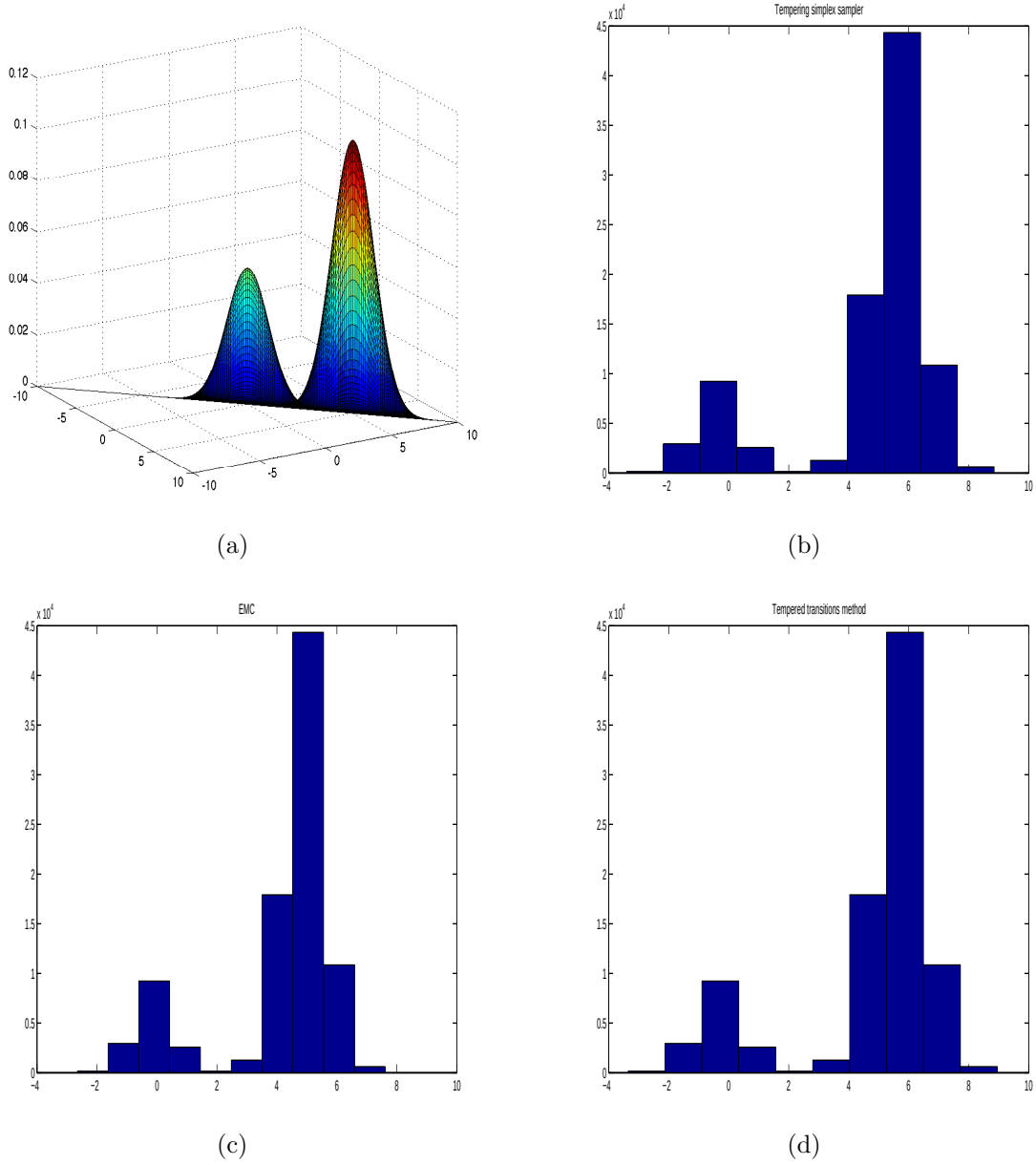
sample under both modes with the correct frequency. Thus, the tempered simplex sampler does not get trapped under a single one.

The integrated autocorrelation and efficient sample size are also evaluated and presented in Table 3.10. The tempered simplex sampler gives the higher efficient sample size compared to the other methods, while the tempered transitions method has the smallest one. The reason may be that the tempered transitions method has very small acceptance probability resulting in slow mixing of the chain and a highly correlated output. Therefore, the integrated autocorrelation for the tempered transition method is the largest one.

Methods	IA	ESS	N	time
tempering simplex sampler	4106	0.24	32,260	32,74
EMC	5002	0.09	28,572	63,47
tempered transitions	5459	0.07	50,000	130,84

Table 3.10: Integrated autocorrelation, efficient sample size, number of iterations and time for the mixture of the two bimodal ten-dimensional Normal distributions using each of the different methods.

In this case, the acceptance probability for the tempered simplex sampler is 0.4949, for the Real-parameter Evolutionary Monte Carlo algorithm is 0.5754 and for the tempered transitions method is 0.0245 in total. Notice that the acceptance probability for the tempered transitions method is very low. Hence, the algorithm does not mix well resulting in slow mixing of the chain.



(a) (b)

(c) (d)

Figure 3.8: (a) Target distribution (b) tempered simplex sampler (c) Real-parameter Evolutionary Monte Carlo and (d) tempered transitions method for the mixture of the two bimodal ten-dimensional Normal distributions.

3.3 Discussion

Usual MCMC methods use a single Markov chain to explore the sampling area and for this reason, they may not perform so well in the case of highly correlated or multi-modal target distributions. For this reason, population MCMC algorithms use several Markov chains that run in parallel and interact in various ways in order to traverse the sampling space. The simplex sampler performs well in the case of highly correlated target distributions but it tends to be trapped under a single maxima. For this reason, the tempered simplex sampler is designed to overcome this problem.

Regular tempering methods use a population of chains one for each temperature. On the other hand, the tempered simplex sampler uses a population of chains for each temperature. Hence, the tempered simplex sampler uses the idea of the simplex sampler to explore the sampling space under each temperature while a population of chains is exchanged under different temperatures. Thus, the tempered simplex sampler uses the good mixing ability of the simplex sampler in the cases of highly correlated target distributions and the good mixing of the tempering methods that are able to traverse the entire sampling space without being trapped under a single maxima. For this reason, the mixing of the tempered simplex sampler is better compared to the simplex sampler described in Chapter 2.

The performance of the tempered simplex sampler is compared to that of the Real-parameter Evolutionary Monte Carlo algorithm and the tempered transitions method. The mixing of the tempered simplex sampler is better than the

Real-parameter Evolutionary Monte Carlo algorithm and the tempered transitions method in the considered examples. Moreover, the Real-parameter Evolutionary Monte Carlo algorithm does not perform as well as the tempered simplex sampler or the tempered transitions method. Furthermore, the tempered simplex sampler gives better efficient sample size in most of the examples.

In the case of a high dimensional target distribution, the tempered simplex sampler performs again better and gives the highest efficient sampling size. Moreover, for the specific example that is was examined, the tempered transitions method has very small acceptance probability resulting in slow mixing of the chain. For this reason, the efficient sample size was very small.

The tempered simplex sampler is computationally very intensive particularly for high dimensional cases. The sampler is much easier to be tuned compared to the Real-parameter Evolutionary Monte Carlo algorithm. When the target is a multi-modal distribution, then there is no natural way to define the simplex. In this case, the number of vertices depends on the complexity of the target.

In conclusion, the tempered simplex sampler samples efficiently from multi-modal distributions. It mixes well under the modes and also traverse the whole sampling area. In future work, the aim is to try different ways of exchanging the population of the several Markov chains under different temperatures. One may be able to use the the tempered transitions method by considering a population of Markov chains under each temperature and not just a single Markov chain.

Chapter 4

Automatic RJMCMC for variable selection - tractable posterior distribution

Variable selection is an important problem in statistics. Given a collection of variables, the question is, which variables best describe the data $\boldsymbol{y}$. Hence, the aim is to find a model that contains enough variables to explain the data and yet is sufficiently parsimonious to avoid over-fitting. Consider the setting of a variable selection problem where there are N possible covariates to choose from. In this case, there are 2^N posssible models. Thus, for even a small number of covariates, the total number of possible models can be considerably large and this presents an enormous problem.

Suppose that $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_N)$ is the parameter vector for the saturated model. If the i^{th} predictor variable is excluded from the model then the corresponding component of the parameter vector is $\theta_i = 0$ while if the i^{th} covariate is included

then $\theta_i \neq 0$. A vector of latent binary variables $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_N)$ can therefore be used to denote whether a particular variable is included or excluded from the model. Thus, if $\theta_i = 0$, then $\gamma_i = 0$ meaning that the i^{th} covariate is not included in the model while if $\gamma_i = 1$ then the i^{th} covariate is included in the model. Hence, each model is characterised by different $\boldsymbol{\gamma}$ binary vectors.

4.1 Reversible Jump Markov chain Monte Carlo

Reversible jump Markov chain Monte Carlo (RJMCMC) introduced by Green (1995) is an extension of the Metropolis-Hastings algorithm for stationary distributions of variable dimension. RJMCMC has been successfully applied in a wide variety of settings, including the challenging problem of determining the number of components in a finite mixture (Richardson and Green (1997)) and determining the number of states in a hidden Markov model (Robert et al. (2000)). RJMCMC usually considers of a selection of move types, some of which explore the parameter space within a model, and others which propose changes to the dimensionality of the model. Thus, the chain explores both the model and parameter space. Assume that the sampling space consists a collection of Λ possible models where each model $M_{\boldsymbol{\gamma}}$ is described by an unknown parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$. The model dimension may vary from model to model. The target distribution is the posterior given by the following formula:

$$\pi(M_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}} | \boldsymbol{y}) \propto L(\boldsymbol{y} | \boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})\, p(\boldsymbol{\theta}_{\boldsymbol{\gamma}} | M_{\boldsymbol{\gamma}})\, p(M_{\boldsymbol{\gamma}}), \tag{4.1}$$

where $L(\boldsymbol{y}|M_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}})$ is the likelihood of the data $\boldsymbol{y}$ described by model $M_{\boldsymbol{\gamma}}$, $p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}})$ is the prior density for parameters $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ within model $M_{\boldsymbol{\gamma}}$ and $p(M_{\boldsymbol{\gamma}})$ indicates the prior probability of model $M_{\boldsymbol{\gamma}}$.

At each iteration of the Markov chain, the sample space is explored by first applying regular Markov chain Monte Carlo methods such as the Metropolis-Hastings algorithm or the Gibbs sampler to explore within a model and then by proposing to change the dimension of the model. Suppose that at the current state, the Markov chain visits model $M_{\boldsymbol{\gamma}}$ with parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ and a jump is proposed to a different dimensional model $M_{\boldsymbol{\gamma}'}$ with parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$. First generate a vector $\mathbf{v}$ from some proposal distribution $q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\cdot)$ and then the proposed parameter vector is chosen as $(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, \mathbf{v}') = f_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \mathbf{v})$ where $f_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\cdot,\cdot)$ is some deterministic invertible function. For the reverse move, a vector $\mathbf{v}'$ is generated from some proposal distribution $q_{\boldsymbol{\gamma}',\boldsymbol{\gamma}}(\cdot)$ where $\dim(\boldsymbol{\theta}_{\boldsymbol{\gamma}}) + \dim(\mathbf{v}) = \dim(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}) + \dim(\mathbf{v}')$. The proposed jump is either accepted as the new state of the Markov chain with probability $A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'})$ where $A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min(1, A)$, with

$$A = \frac{\pi(M_{\boldsymbol{\gamma}'}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{\pi(M_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}})}{p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})} \frac{q_{\boldsymbol{\gamma}',\boldsymbol{\gamma}}(\mathbf{v}')}{q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\mathbf{v})} \left| \frac{\partial f_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \mathbf{v})}{\partial f_{\boldsymbol{\gamma}',\boldsymbol{\gamma}}(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, \mathbf{v}')} \right| \tag{4.2}$$

or rejected with probability $1 - A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'})$ and the Markov chain remains at model $M_{\boldsymbol{\gamma}}$.

Here, $p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})$ is the probability of proposing a jump from model $M_{\boldsymbol{\gamma}}$ to model $M_{\boldsymbol{\gamma}'}$. The last term of equation (4.2) is the Jacobian for the transformation of the parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ to parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$.

In the case where models are nested or the interpretation of the parameters

is similar between different models, a popular approach to define the proposed parameter vector is to add or delete parameters in the current one. Hence, the jumps are deterministic in one direction, in this case. Every time that a higher dimensional jump is proposed, the new parameter vector is estimated as $\boldsymbol{\theta}_{\boldsymbol{\gamma}'} = f_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_k, \mathbf{v})$ where $\mathbf{v}$ is generated from $q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\cdot)$ and $\dim(\boldsymbol{\theta}_{\boldsymbol{\gamma}}) + \dim(\mathbf{v}) = \dim(\boldsymbol{\theta}_{\boldsymbol{\gamma}'})$. The acceptance probability is $A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min(1, A)$, with

$$A = \frac{\pi(M_{\boldsymbol{\gamma}'}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{\pi(M_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}})}{p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})} \frac{1}{q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\mathbf{v})} \left| \frac{\partial f_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \mathbf{v})}{\partial(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \mathbf{v})} \right|.$$

The reverse move to a smaller dimensional space involves moving from $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$ to $(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \mathbf{v})$ and the acceptance probability is equal to $\min(1, A)$.

RJMCMC allows the parameters of the proposed model to depend on the parameters in the current model in a general way through the function $f_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\cdot,\cdot)$. The problem of choosing $f_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\cdot,\cdot)$ involves choosing the structural aspect of the proposal mechanism as Peter Green addresses in a discussion of Brooks et al. (2003). He suggests that this is often the difficult and crucial part in implementing RJMCMC.

If the function $f_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\cdot,\cdot)$ is the identity matrix then the Jacobian term in formula (4.2) is one. In this case, the full parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$ is sampled from a proposal distribution $q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\cdot)$ and the acceptance probability is

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{\pi(M_{\boldsymbol{\gamma}'}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{\pi(M_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}})}{p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})} \frac{q_{\boldsymbol{\gamma}',\boldsymbol{\gamma}}(\boldsymbol{\theta}_{\boldsymbol{\gamma}})}{q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}'})}\right).$$

4.2 Proposal distribution of the parameters - Different updating schemes

Depending on the structure of the problem, it is probable that some of the models will have variables in common. For instance, if the models are nested then it may be reasonable to assume that the interpretation of the parameters between different models may be the same meaning that the parameter values are similar between different models. On the other hand, considering regression problems, it may not be so natural to assume that the parameters have the same interpretation between different models.

In this chapter, it is assumed that the posterior distribution of the parameters in each model is of a standard form. In this case, our strategy is to use the full-conditional posterior distribution of the parameters as a proposal distribution. Thus, assume that model $M_{\boldsymbol{\gamma}}$ is indicated by the parameter vector $\boldsymbol{\gamma}$. When a jump is proposed from model $M_{\boldsymbol{\gamma}}$ to model $M_{\boldsymbol{\gamma}'}$ then the proposed parameter vector is generated from the full-conditional posterior distribution of the parameters of model $M_{\boldsymbol{\gamma}'}$. Our primary question of interest is to decide whether it is better to condition on one, some or all of the parameters that are in common to both models.

It is assumed that the posterior distribution of the parameters is of standard form, and the marginal likelihood is available, since the posterior model probabilities are calculated *exactly*, and these estimates are compared to estimates gathered using the RJMCMC algorithm.

Assume that at the current state, model $M_{\boldsymbol{\gamma}}$ is visited with parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ and a jump is proposed to model $M_{\boldsymbol{\gamma}'}$ with parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$. Let $\boldsymbol{\psi}$

indicate which variables are in common between models $M_{\boldsymbol{\gamma}}$ and $M_{\boldsymbol{\gamma}'}$ so that $\boldsymbol{\psi} = \{i : \gamma_i = \gamma_i' = 1, 1 \leq i \leq N\}$. Suppose that $\boldsymbol{\psi}$ is not the empty set and let $\boldsymbol{\psi}'$ be a non-empty subset of $\boldsymbol{\psi}$ which denotes the parameter values that are conditioned on. Hence, $\boldsymbol{\theta}_{\boldsymbol{\psi}'}$ is a collection of θ_i parameters for which the corresponding variables are common in both the current and proposed models so that $\boldsymbol{\theta}_{\boldsymbol{\psi}'} = \{\theta_i : i \in \boldsymbol{\psi}'\}$.

To illustrate this, consider the following situation. Suppose that we are in a variable selection setting with data $\boldsymbol{y}$ and a selection of explanatory variables x_1, x_2, x_3, x_4, x_5. Suppose that the Markov chain visits model $M_{\boldsymbol{\gamma}}$, where $\boldsymbol{\gamma} = (1, 1, 1, 0, 1)$ and with parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}} = (\theta_1, \theta_2, \theta_3, \theta_5)$. Furthermore, suppose that a move is proposed to model $M_{\boldsymbol{\gamma}'}$, with $\boldsymbol{\gamma}' = (1, 1, 0, 1, 1)$. Hence, $\boldsymbol{\psi} = \{1, 2, 5\}$ and this indicates that variables x_1, x_2 and x_5 are common to model $M_{\boldsymbol{\gamma}}$ and $M_{\boldsymbol{\gamma}'}$. Our interest is to decide which, if any of the corresponding parameters, θ_1, θ_2 and θ_5 to condition on in our proposal distribution. For example, suppose that we choose to update parameters θ_1 and θ_5 as well as the newly introduced parameter θ_4. Therefore, the proposal distribution $q(\boldsymbol{\theta}^*_{\boldsymbol{\gamma}'}|\boldsymbol{\theta}_{\boldsymbol{\gamma}})$ is the full-conditional posterior distribution, $\pi(\theta_1^*, \theta_4^*, \theta_5^*|\theta_2, M_{\boldsymbol{\gamma}'})$, so that $\boldsymbol{\theta}_{\boldsymbol{\gamma}'} = (\theta_1^*, \theta_2, \theta_4^*, \theta_5^*)$. In general, let $\boldsymbol{\psi}' \subseteq \boldsymbol{\psi}$ denote the variables from among the collection of variables that are in common with $M_{\boldsymbol{\gamma}}$ and $M_{\boldsymbol{\gamma}'}$ that are chosen to condition on. Three different strategies are considered depending on the $\boldsymbol{\psi}'$ choice so that

$$\boldsymbol{\psi}' = \begin{cases} \boldsymbol{\psi}, & \textit{Scheme A} \\ \emptyset, & \textit{Scheme C} \\ \text{otherwise}, & \textit{Scheme B} \end{cases}$$

4.2.1 Scheme A: Condition on all common variables

Brooks and Ehlers (2008) used the second order method of Brooks et al. (2003) for the autoregressive model choice in the difficult problem of choosing the model order. It turns out that in this case, the optimal proposal distribution is the full-conditional posterior distribution of the parameters. For a linear regression variable selection problem, it can be proven that the proposal distribution which is estimated with the second order method of Brooks et al. (2003) is also the full-conditional posterior distribution of the parameters when only the newly introduced parameters are generated from it and the other parameters retain the current parameter values in the proposed model.

Thus, when the models are nested or the interpretation of the parameters does not change between models, it might be reasonable to generate only the newly introduced parameters $\mathbf{v}$ and retain the common parameter values in the proposed state. This is actually the most popular choice in the implementation of the RJMCMC algorithm. Hence, the vector $\mathbf{v}$ is drawn from the full-conditional posterior distribution $q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\mathbf{v}) = \pi(\mathbf{v}|\boldsymbol{\theta}_{\boldsymbol{\psi}}, \boldsymbol{y}, M_{\boldsymbol{\gamma}'})$ where $\dim(\mathbf{v}) + \dim(\boldsymbol{\theta}_{\boldsymbol{\gamma}}) = \dim(\boldsymbol{\theta}_{\boldsymbol{\gamma}'})$. In this case, the acceptance probability is given by the following formula:

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, M_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}})}{p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})} \frac{1}{q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\mathbf{v})}\right).$$

Here, the proposed parameter vector $\mathbf{v}$ is generated by conditioning on the current state of the chain, i.e the current parameters $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$. In the case of nested models like the autoregressive model choice, the jumping move is deterministic in one direction. Hence, if a jump is proposed to a higher dimension then only the newly introduced parameters $\mathbf{v}$ are added in the current parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ to

match the proposed dimension. On the other hand, when a jump is proposed to a lower dimension then some of the current parameter values are deleted for the same reason.

Scheme A is the most popular choice to generate the proposed parameter vector inside the RJMCMC algorithm even though, it does not seem to be the best strategy. For instance, consider the data set that appears in Figure 4.1. One might want to decide which of the two models, the constant or the two dimensional linear model, describes the data better. Suppose that the RJMCMC algorithm is used to find out the best model. At the current state of the Markov chain, model K_1 is visited with parameters $\boldsymbol{\theta}_1 = a$ and a jump is proposed to model K_2 with parameter vector $\boldsymbol{\theta}_2 = (a^*, b^*)$. If the current parameter values are retained in the proposed state of the Markov chain such that $a^* = a$, the best model may never be visited. Then again, if both parameters are updated in the proposed state, then there may be a better chance of finding the best model. Hence, retaining the current parameters in the proposed state may not always be a good strategy. It seems that proposing a totally new parameter vector or adjusting the current parameters after generating the new ones may be a better choice.

4.2.2 Scheme B: Condition on some subset of the common variables

Following the previous section, it may be reasonable to update some of the current parameters together with the newly introduced ones conditioning on some subset of the common variables. Thus, assume that at the current state model $M_{\boldsymbol{\gamma}}$ is

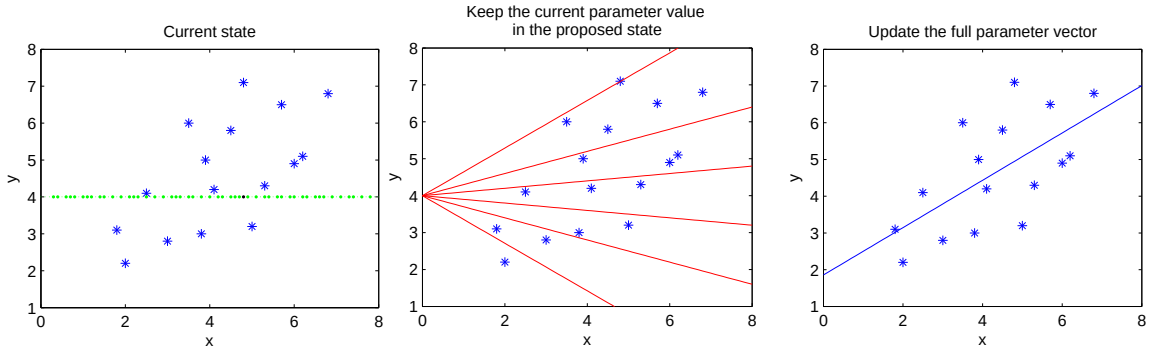


Figure 4.1: (a) Current state of the Markov chain, $y = a$. (b) Proposed state of the Markov chain, $y = a + b^*x$, when the current parameter value a is retained in the proposed state. (c) One of the possible proposed states of the Markov chain, $y = a^* + b^*x$, when the full parameter vector is updated in the proposed state.

visited with parameters $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ and a jump is proposed to model $M_{\boldsymbol{\gamma}'}$ with parameters $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$. Recall that $\boldsymbol{\psi}$ consists of all the common variables, while $\boldsymbol{\psi}'$ is a subset of $\boldsymbol{\psi}$. Then, $\boldsymbol{\theta}_{\boldsymbol{\psi}'}$ is the parameter vector of all the common variables to be conditioned on such as $\boldsymbol{\theta}_{\boldsymbol{\psi}'} = \{\theta_i : i \in \boldsymbol{\psi}'\}$. Hence, a subset of some common variables is generated together with the newly introduced ones conditioning on $\boldsymbol{\theta}_{\boldsymbol{\psi}'}$. In this case, the proposal distribution is $q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\mathbf{v}) = \pi(\mathbf{v}|\boldsymbol{\theta}_{\boldsymbol{\psi}'}, \boldsymbol{y}, M_{\boldsymbol{\gamma}'})$ where $\dim(\mathbf{v}) + \dim(\boldsymbol{\theta}_{\psi'}) = \dim(\boldsymbol{\theta}_{\boldsymbol{\gamma}'})$ and the acceptance probability is

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{\pi(M_{\boldsymbol{\gamma}'}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{\pi(M_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}})}{p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})} \frac{q_{\boldsymbol{\gamma}',\boldsymbol{\gamma}}(\boldsymbol{\theta}_{\psi'})}{q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\mathbf{v})}\right) \tag{4.3}$$

Unfortunately, there is no natural way to lead the choice of how many and which of the common variables should be conditioned on. Hence, there is no obvious

way to choose $\boldsymbol{\psi}'$. Therefore, the elements of $\boldsymbol{\psi}'$ are randomly chosen each time.

4.2.3 Scheme C: Condition on no common variables

In this scheme, the full parameter vector is updated in the proposed state of the Markov chain conditioning on no common parameter values. For this reason, the proposal mechanism is independent of the current state. The proposal distribution is now the posterior distribution of model $M_{\boldsymbol{\gamma}'}$, $q_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\cdot) = \pi(\mathbf{v}|\boldsymbol{y}, M_{\boldsymbol{\gamma}'})$, where $dim(\mathbf{v}) = dim(\boldsymbol{\theta}_{\boldsymbol{\gamma}'})$ and the acceptance probability is

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{\pi(M_{\boldsymbol{\gamma}'}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{\pi(M_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}})}{p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})} \frac{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})}{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y}, M_{\boldsymbol{\gamma}'})}\right).$$

Notice that the joint posterior model probability is proportional to the likelihood times the prior distribution of the parameters and the prior distribution for the model so that $\pi(M_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}) \propto \ell(\boldsymbol{y}|\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}}) p(M_{\boldsymbol{\gamma}})$. Hence, the acceptance probability takes the following form:

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{\ell(\boldsymbol{y}|\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, M_{\boldsymbol{\gamma}'}) p(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}|M_{\boldsymbol{\gamma}'}) p(M_{\boldsymbol{\gamma}'})}{\ell(\boldsymbol{y}|\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}}) p(M_{\boldsymbol{\gamma}})} \frac{p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}})}{p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})} \frac{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})}{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y}, M_{\boldsymbol{\gamma}'})}\right).$$

Furthermore, the marginal likelihood of the data assuming model $M_{\boldsymbol{\gamma}}$ is

$$p(\boldsymbol{y}|M_{\boldsymbol{\gamma}}) = \ell(\boldsymbol{y}|\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})\, p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}}) \big/ \pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}}).$$

Therefore, the acceptance probability can be written as

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{p(\boldsymbol{y}|M_{\boldsymbol{\gamma}'})}{p(\boldsymbol{y}|M_{\boldsymbol{\gamma}})} \frac{p(M_{\boldsymbol{\gamma}'})}{p(M_{\boldsymbol{\gamma}})} \frac{p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}})}{p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})}\right).$$

Moreover, the posterior model probability is $p(M_{\boldsymbol{\gamma}}|\boldsymbol{y}) \propto p(\boldsymbol{y}|M_{\boldsymbol{\gamma}})p(M_{\boldsymbol{\gamma}})$ so that the acceptance probability is defined as

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{p(M_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{p(M_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}})}{p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})}\right). \tag{4.4}$$

Hence, the acceptance probability given in (4.4) is independent of the current parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ or the proposed one $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$. It is the ratio of the posterior model probability for model $M_{\boldsymbol{\gamma}'}$ to the posterior model probability for model $M_{\boldsymbol{\gamma}}$ multiplied by the ratio of the probability to jump from model $M_{\boldsymbol{\gamma}'}$ to model $M_{\boldsymbol{\gamma}}$ versus the probability for the reverse move. When the probability of moving from model $M_{\boldsymbol{\gamma}'}$ to model $M_{\boldsymbol{\gamma}}$ equals the probability of moving from model $M_{\boldsymbol{\gamma}}$ to model $M_{\boldsymbol{\gamma}'}$, that is $p(M_{\boldsymbol{\gamma}'} \to M_{\boldsymbol{\gamma}}) = p(M_{\boldsymbol{\gamma}} \to M_{\boldsymbol{\gamma}'})$, the acceptance probability for Scheme C reduces to the odds ratio of the posterior model probability of model $M_{\boldsymbol{\gamma}}$ to model $M_{\boldsymbol{\gamma}'}$, i.e

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{p(M_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{p(M_{\boldsymbol{\gamma}}|\boldsymbol{y})}\right).$$

Suppose that at the current state, model $M_{\boldsymbol{\gamma}}$ is visited with parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$. Then, at each iteration the proposal mechanism of the RJMCMC algorithm using each of the updating schemes is described in Algorithm 5.

If the full-conditional posterior distribution of the parameters has a tractable form then it is straightforward to generate the proposed parameter vector from it and for this reason, no tuning is necessary. In the next section, a normal linear regression model is used to test the performance of the algorithm applying each

Algorithm 5 Proposal mechanism of RJMCMC

Jump move: Change in dimension

1. Choose the proposed model $M_{\boldsymbol{\gamma}'}$ using a local proposal distribution that jumps up or down one or more dimensions.

   If Scheme A: Generate the proposed parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$ so that $\boldsymbol{\theta}_{\boldsymbol{\gamma}'} = (\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \mathbf{v})$, where $\mathbf{v} \sim \pi(\mathbf{v}|\boldsymbol{\theta}_{\boldsymbol{\psi}}, \boldsymbol{y}, M_{\boldsymbol{\gamma}'})$.

   If Scheme B: Generate the proposed parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$ so that $\boldsymbol{\theta}_{\boldsymbol{\gamma}'} = (\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \mathbf{v})$, where $\mathbf{v} \sim \pi(\mathbf{v}|\boldsymbol{\theta}_{\boldsymbol{\psi}'}, \boldsymbol{y}, M_{\boldsymbol{\gamma}'})$.

   If Scheme C: Generate the proposed parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$ so that $\boldsymbol{\theta}_{\boldsymbol{\gamma}'} \sim \pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y}, M_{\boldsymbol{\gamma}'})$.

2. Accept or reject the proposed move with the corresponding probability $A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'})$.

Exchange move: No change in dimension

1. Choose a model of the same dimension, M_k where $dim(\boldsymbol{\theta}_k) = dim(\boldsymbol{\theta}_{\boldsymbol{\gamma}})$.

2. **If Scheme A:** Generate the proposed parameter vector $\boldsymbol{\theta}_k$ so that $\boldsymbol{\theta}_k = (\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \mathbf{v})$, where $\mathbf{v} \sim \pi(\mathbf{v}|\boldsymbol{\theta}_{\boldsymbol{\psi}}, \boldsymbol{y}, M_k)$.

   If Scheme B: Generate the proposed parameter vector $\boldsymbol{\theta}_k$ so that $\boldsymbol{\theta}_k = (\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \mathbf{v})$, where $\mathbf{v} \sim \pi(\mathbf{v}|\boldsymbol{\theta}_{\boldsymbol{\psi}'}, \boldsymbol{y}, M_k)$.

   If Scheme C: Generate the proposed parameter vector $\boldsymbol{\theta}_k$ so that $\boldsymbol{\theta}_k \sim \pi(\boldsymbol{\theta}_k|\boldsymbol{y}, M_k)$.

3. Accept or reject the proposed move with the corresponding probability $A_{\boldsymbol{\gamma},k}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_k)$.

Gibbs updates: No change in dimension

If Scheme A or Scheme B: Single component updates using the full-conditional posterior distribution of the parameters.

If Scheme C: Block updates using the posterior distribution of the parameters.

of the three different updating schemes since the marginal likelihoods, $p(\boldsymbol{y}|M_k)$, are tractable. For this reason, the actual posterior model probabilities are known and can be compared to the estimated posterior model probabilities using the RJMCMC algorithm for each of the three different updating schemes. Then, conclusions can be made on the performance of the three different algorithms.

An autoregressive example is also used to test the performance of the algorithm. Here, the full-conditional posterior distribution of the parameters is also tractable but the marginal likelihood, $p(\boldsymbol{y}|M_k)$, is not known in closed form. In this case, the power posterior method of Friel and Pettitt (2008) is used to estimate the posterior model probabilities. The posterior model probabilities are estimated very accurately using the power posterior method by running a very long Markov chain. Then, they are compared to the ones estimated with the RJMCMC algorithm using each of the three updating schemes. Therefore, conclusions can be made on the performance of the different RJMCMC algorithms.

4.3 Examples

4.3.1 Linear regression model choice

In this section, the performance of the RJMCMC algorithm using each of the updating schemes is tested on a linear regression problem. This is a teaching example since the marginal likelihoods are known in closed form. Thus, the posterior model probabilities are also tractable and there is no need for the RJMCMC algorithm to be implemented. On the other hand, the estimated posterior model probabilities with the RJMCMC method can be compared to the actual target

ones. Then, conclusions can be made on the performance of the algorithms and consequently, it can be decided which of the different updating schemes is best.

Consider a variable selection problem where $\boldsymbol{y} = (y_1, \ldots, y_n)^T$ is a collection of n observed responses and $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})^T$ is the $p \times 1$ vector of predictors for y_i, with $i = 1, \ldots, n$. $\mathbf{X}$ is a $n \times p$ matrix where its i^{th} row is $\boldsymbol{x}_i^T$. Hence, the Normal linear regression model is given in the following formula

$$\boldsymbol{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_p)^T$ is a $p \times 1$ vector of unknown parameters and the residuals $\boldsymbol{\epsilon}$ are Normally distributed according to $N_n(0, \frac{1}{\tau}\mathbf{I}_n)$ where τ is the precision of the residual and $\mathbf{I}_n$ is the identity matrix.

In Bayesian variable selection problems, a prior distribution is used to describe the prior beliefs about the vector of unknown parameters $\boldsymbol{\beta}$. Notice that if the component of the parameter vector $\beta_j = 0$ then it is implied that the j^{th} predictor variable is not included in the model. On the other hand, when $\beta_j \neq 0$ then the j^{th} predictor variable is included in the model. A vector of latent binary variables $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_n)$ indicates which variable is included or excluded from the model. Thus, each of the components of the binary vector is $\gamma_j = 0$ when $\beta_j = 0$ which implies that the j^{th} predictor variable is not included in the model and $\gamma_j = 1$, otherwise. There are various models indicated by different $\boldsymbol{\gamma}$ values depending on whether the j^{th} predictor variable is used to estimate the mean of $\boldsymbol{y}$ or not.

Given a vector $\boldsymbol{\gamma}$, the corresponding parameter vector $\boldsymbol{\beta}_{\boldsymbol{\gamma}}$ is obtained by extracting the non-zero components of $\boldsymbol{\beta}$. For this reason, $\boldsymbol{\beta}_{\boldsymbol{\gamma}}$ is a subset of $\boldsymbol{\beta}$. Likewise, $\mathbf{X}_{\boldsymbol{\gamma}}$ includes all columns of $\mathbf{X}$ that correspond to $\gamma_i = 1$. Assume that

for known $\boldsymbol{\gamma}$ values, $\boldsymbol{y}$ is defined as

$$\boldsymbol{y} = \mathbf{X}_{\boldsymbol{\gamma}}\,\boldsymbol{\beta}_{\boldsymbol{\gamma}} + \boldsymbol{\epsilon}.$$

The conjugate prior on $\boldsymbol{\beta}_{\boldsymbol{\gamma}}$ is denoted by $p(\boldsymbol{\beta}_{\boldsymbol{\gamma}}|\tau, M_{\boldsymbol{\gamma}})$ and the conjugate prior on τ is $p(\tau|M_{\boldsymbol{\gamma}})$. The marginal likelihood is

$$p(\boldsymbol{y}|M_{\boldsymbol{\gamma}}) = \int\int p(\boldsymbol{\beta}_{\boldsymbol{\gamma}}|\tau, M_{\boldsymbol{\gamma}})p(\tau|M_{\boldsymbol{\gamma}})d\boldsymbol{\beta}_{\boldsymbol{\gamma}}d\tau$$

and the prior distribution for the unknown parameters is

$$p(\boldsymbol{\beta}_{\boldsymbol{\gamma}}, \tau, M_{\boldsymbol{\gamma}}) = p(\boldsymbol{\beta}_{\boldsymbol{\gamma}}|\tau, M_{\boldsymbol{\gamma}})p(\tau|M_{\boldsymbol{\gamma}})p(M_{\boldsymbol{\gamma}}).$$

Following Bernardo and Smith (1994), it is assumed that the prior distribution for the parameter vector $\boldsymbol{\beta}_{\boldsymbol{\gamma}}$, $p(\boldsymbol{\beta}_{\boldsymbol{\gamma}}|\tau, M_{\boldsymbol{\gamma}})$, is $N_p(\mathbf{0}, \tau^{-1}\mathbf{I}_p)$ and for the precision τ, $p(\tau|M_{\boldsymbol{\gamma}})$, is $\mathrm{Ga}(a, b^{-1})$ so that the prior distribution for the parameters within model $M_{\boldsymbol{\gamma}}$ is a Normal-Gamma defined as $p(\boldsymbol{\beta}_{\boldsymbol{\gamma}}, \tau|M_{\boldsymbol{\gamma}}) = p(\boldsymbol{\beta}_{\boldsymbol{\gamma}}|\tau, M_{\boldsymbol{\gamma}})p(\tau|M_{\boldsymbol{\gamma}}) = N_p\mathrm{Ga}(\mathbf{0}, \tau^{-1}, \mathbf{I}_p, a, b^{-1})$. The prior distribution for each model $M_{\boldsymbol{\gamma}}$, $p(M_{\boldsymbol{\gamma}})$, is a Uniform distribution. Hence, the data $\boldsymbol{y}$ are modelled as

$$\boldsymbol{y} \sim N_n(\mathbf{X}_{\boldsymbol{\gamma}}\,\boldsymbol{\beta}_{\boldsymbol{\gamma}}, \tau^{-1}\mathbf{I}_n).$$

The likelihood is thus given by

$$L(\boldsymbol{y}|\mathbf{X}_{\boldsymbol{\gamma}},\, \boldsymbol{\beta}_{\boldsymbol{\gamma}},\, \tau, M_{\boldsymbol{\gamma}}) = \frac{1}{|\tau^{-1}\mathbf{I}_n|^{\frac{1}{2}}\,(2\,\pi)^{\frac{n}{2}}} \exp\left\{-\frac{\tau}{2}\left(\boldsymbol{y} - \mathbf{X}_{\boldsymbol{\gamma}}\,\boldsymbol{\beta}_{\boldsymbol{\gamma}}\right)'\left(\boldsymbol{y} - \mathbf{X}_{\boldsymbol{\gamma}}\,\boldsymbol{\beta}_{\boldsymbol{\gamma}}\right)\right\}.$$

The priors are

$$p(\boldsymbol{\beta}_{\boldsymbol{\gamma}}|\tau, M_{\boldsymbol{\gamma}}) = \frac{1}{|\tau^{-1}\mathbf{I}_p|^{\frac{1}{2}}\,(2\,\pi)^{\frac{p}{2}}} \exp\left\{-\frac{\tau}{2}\,\boldsymbol{\beta}'_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}}\right\}$$

and

$$p(\tau|M_{\boldsymbol{\gamma}}) = \frac{\tau^{a-1}}{\frac{1}{b^a}\Gamma(a)} \exp\{-\tau b\}.$$

The posterior probability of the parameters is then

$$\begin{aligned}
\pi(\boldsymbol{\beta}_{\boldsymbol{\gamma}}, \tau|\boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) &\propto L(\boldsymbol{y}|\mathbf{X}_{\boldsymbol{\gamma}},\, \boldsymbol{\beta}_{\boldsymbol{\gamma}},\, \tau, M_{\boldsymbol{\gamma}})\, p(\boldsymbol{\beta}_{\boldsymbol{\gamma}}, \tau|M_{\boldsymbol{\gamma}}) \\
&\propto \tau^{a-1+\frac{n}{2}+\frac{p}{2}} \exp\left\{-\frac{\tau}{2}\left(\boldsymbol{y}' - \boldsymbol{\beta}'_{\boldsymbol{\gamma}}\mathbf{X}'_{\boldsymbol{\gamma}}\right)\left(\boldsymbol{y} - \mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}}\right) - \right. \\
&\quad \left. \frac{\tau}{2}\boldsymbol{\beta}'_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \tau\, b\right\} \\
&= \tau^{a+\frac{n}{2}+\frac{p}{2}-1} \exp\left\{-\frac{\tau}{2}\left[\boldsymbol{y}'\boldsymbol{y} - \boldsymbol{y}'\mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\beta}'_{\boldsymbol{\gamma}}\mathbf{X}'_{\boldsymbol{\gamma}}\boldsymbol{y} + \right.\right. \\
&\quad \left.\left. \boldsymbol{\beta}'_{\boldsymbol{\gamma}}\mathbf{X}'_{\boldsymbol{\gamma}}\mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}} + \boldsymbol{\beta}'_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}} + 2b\right]\right\}. \\
&= \tau^{a+\frac{n}{2}+\frac{p}{2}-1} \exp\left\{-\frac{\tau}{2}\left[\boldsymbol{\beta}'_{\boldsymbol{\gamma}}\left(\mathbf{X}'_{\boldsymbol{\gamma}}\mathbf{X}_{\boldsymbol{\gamma}} + \mathbf{I}_p\right)\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \right.\right. \\
&\quad \left.\left. \boldsymbol{\beta}'_{\boldsymbol{\gamma}}\mathbf{X}'_{\boldsymbol{\gamma}}\boldsymbol{y} - \boldsymbol{y}'\mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}} + \boldsymbol{y}'\boldsymbol{y} + 2b\right]\right\}. \qquad (4.5)
\end{aligned}$$

In general, it can be written that if $\mathbf{z} \sim N_n(\boldsymbol{\mu}, \Sigma)$ then

$$\begin{aligned}
f(\mathbf{z}|\boldsymbol{\mu}, \Sigma) &= \frac{1}{|\Sigma|^{\frac{1}{2}}\,(2\,\pi)^{\frac{n}{2}}} \exp\left\{-\frac{1}{2}\,(\mathbf{z} - \boldsymbol{\mu})'\,\Sigma^{-1}\,(\mathbf{z} - \boldsymbol{\mu})\right\} \\
&= \frac{1}{|\Sigma|^{\frac{1}{2}}\,(2\,\pi)^{\frac{n}{2}}} \exp\left\{-\frac{1}{2}\,(\mathbf{z}' - \boldsymbol{\mu}')\,\Sigma^{-1}\,(\mathbf{z} - \boldsymbol{\mu})\right\} \\
&= \frac{1}{|\Sigma|^{\frac{1}{2}}\,(2\,\pi)^{\frac{n}{2}}} \exp\left\{-\frac{1}{2}\left(\mathbf{z}'\Sigma^{-1}\mathbf{z} - \mathbf{z}'\Sigma^{-1}\boldsymbol{\mu} - \boldsymbol{\mu}'\Sigma^{-1}\mathbf{z} + \boldsymbol{\mu}'\Sigma^{-1}\boldsymbol{\mu}\right)\right\}. \qquad (4.6)
\end{aligned}$$

From (4.5) and (4.6), the posterior probability can be written as

$$\begin{aligned}\pi(\boldsymbol{\beta_\gamma}, \tau|\boldsymbol{y}, \mathbf{X_\gamma}, M_{\boldsymbol{\gamma}}) &\propto \tau^{a+\frac{n}{2}+\frac{p}{2}-1} \exp\left\{-\frac{\tau}{2}\left[\left(\boldsymbol{\beta_\gamma}-\boldsymbol{\mu}_*\right)' \Sigma_*^{-1}\left(\boldsymbol{\beta_\gamma}-\boldsymbol{\mu}_*\right) - \right.\right. \\ &\quad \left.\left. \boldsymbol{\mu}_*\Sigma_*^{-1}\boldsymbol{\mu}_* + \boldsymbol{y}'\boldsymbol{y} + 2b\right]\right\} \\ &\propto \tau^{\frac{p}{2}} \exp\left\{-\frac{\tau}{2}\left(\boldsymbol{\beta_\gamma}-\boldsymbol{\mu}_*\right)' \Sigma_*^{-1}\left(\boldsymbol{\beta_\gamma}-\boldsymbol{\mu}_*\right)\right\} \\ &\quad \tau^{a+\frac{n}{2}-1} \exp\left\{-\tau\left[\frac{\boldsymbol{y}'\boldsymbol{y}-\boldsymbol{\mu}_*\Sigma_*^{-1}\boldsymbol{\mu}_*}{2}+b\right]\right\}\end{aligned}$$

where $\boldsymbol{\mu}_* = \Sigma_*(\mathbf{X\gamma}'\boldsymbol{y})$, $\Sigma_* = (\mathbf{X\gamma}'\mathbf{X\gamma} + \mathbf{I}_p)^{-1}$, $a_* = a + \frac{n}{2}$ and $b_* = \frac{1}{2}(\boldsymbol{y}'\boldsymbol{y} - \mu_*'\Sigma_*^{-1}\mu_*) + b$.

Hence, the posterior probability of the parameters is a Normal-Gamma distribution

$$\pi(\boldsymbol{\beta_\gamma}, \tau|\boldsymbol{y}, \mathbf{X_\gamma}, M_{\boldsymbol{\gamma}}) = N_p\mathrm{Ga}(\boldsymbol{\mu}_*, \Sigma_*, a_*, b_*).$$

Then, the posterior distribution can be written as

$$\pi(\boldsymbol{\beta_\gamma}, \tau|\boldsymbol{y}, \mathbf{X_\gamma}, M_{\boldsymbol{\gamma}}) = \frac{p(\boldsymbol{\beta_\gamma}, \tau)\, \ell(\boldsymbol{y}|\mathbf{X_\gamma}, \boldsymbol{\beta_\gamma}, \tau, M_\gamma)}{p(\boldsymbol{y}|M_{\boldsymbol{\gamma}})}.$$

Therefore, the marginal likelihood appears as

$$\begin{aligned}p(\boldsymbol{y}|M_{\boldsymbol{\gamma}}) &= \frac{p(\boldsymbol{\beta_\gamma}|\tau)\, p(\tau)\, \ell(\boldsymbol{y}|\mathbf{X_\gamma}, \boldsymbol{\beta_\gamma}, \tau, M_\gamma)}{\pi(\boldsymbol{\beta_\gamma}, \tau|\boldsymbol{y}, \mathbf{X_\gamma}, M_{\boldsymbol{\gamma}})} \\ &= \frac{\frac{b^a\tau^{a-1}}{\tau^{-p/2}|\mathbf{I}_p|^{1/2}(2\pi)^{p/2}\Gamma(a)\tau^{-n/2}(2\pi)^{n/2}}}{\frac{\tau^{a_*-1}b_*^{a_*}}{|\tau^{-1}\Sigma_*|^{1/2}(2\pi)^{p/2}\Gamma(a_*)}} \exp\{q\} \\ &= \frac{b^a\tau^{a-1}\tau^{-p/2}|\Sigma_*|^{1/2}(2\pi)^{p/2}\Gamma(a_*)}{\tau^{-p/2}|\mathbf{I}_p|^{1/2}(2\pi)^{p/2}\Gamma(a)\tau^{-n/2}(2\pi)^{n/2}\tau^{a_*-1}b_*^{a_*}} \exp\{q\}.\end{aligned}$$

Recall that a_* is defined as $a_* = a + \frac{n}{2}$. Hence, the marginal likelihood is

$$\pi(\boldsymbol{y}|M_{\boldsymbol{\gamma}}) = \frac{b^a|\Sigma_*|^{1/2}\Gamma(a_*)}{|\mathbf{I}_p|^{1/2}\Gamma(a)(2\pi)^{n/2}b_*^{a_*}}\exp\{q\}$$

where the exponential term, $\exp\{q\}$, is

$$\begin{aligned}
\exp\{q\} &= \exp\left\{-\tau\left[b - b_* + \frac{1}{2}\left[\boldsymbol{\beta}_{\boldsymbol{\gamma}}'\boldsymbol{\beta}_{\boldsymbol{\gamma}} + \left(\boldsymbol{y} - \mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}}\right)'\left(\boldsymbol{y} - \mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}}\right) - \right.\right.\right.\\
&\quad \left.\left.\left.\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)'\Sigma_*^{-1}\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)\right]\right]\right\}\\
&= \exp\left\{-\tau\left[b - b_* + \frac{1}{2}\left[\boldsymbol{\beta}_{\boldsymbol{\gamma}}'\boldsymbol{\beta}_{\boldsymbol{\gamma}} + (\boldsymbol{y}' - \boldsymbol{\beta}_{\boldsymbol{\gamma}}'\mathbf{X}_{\boldsymbol{\gamma}}')(\boldsymbol{y} - \mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}}) - \right.\right.\right.\\
&\quad \left.\left.\left.\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)'\Sigma_*^{-1}\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)\right]\right]\right\}\\
&= \exp\left\{-\tau\left[b - b_* + \frac{1}{2}\left[\boldsymbol{\beta}_{\boldsymbol{\gamma}}'\boldsymbol{\beta}_{\boldsymbol{\gamma}} + \boldsymbol{y}'\boldsymbol{y} - \boldsymbol{\beta}_{\boldsymbol{\gamma}}'\mathbf{X}_{\boldsymbol{\gamma}}'\boldsymbol{y} - \boldsymbol{y}'\mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}} + \right.\right.\right.\\
&\quad \left.\left.\left.\boldsymbol{\beta}_{\boldsymbol{\gamma}}'\mathbf{X}_{\boldsymbol{\gamma}}'\mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)'\Sigma_*^{-1}\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)\right]\right]\right\}\\
&= \exp\left\{-\tau\left[b - b_* + \frac{1}{2}\left[\boldsymbol{\beta}_{\boldsymbol{\gamma}}'(\mathbf{I}_p + \mathbf{X}_{\boldsymbol{\gamma}}'\mathbf{X}_{\boldsymbol{\gamma}})\boldsymbol{\beta}_{\boldsymbol{\gamma}} + \boldsymbol{y}'\boldsymbol{y} - \boldsymbol{\beta}_{\boldsymbol{\gamma}}'\mathbf{X}_{\boldsymbol{\gamma}}'\boldsymbol{y} - \right.\right.\right.\\
&\quad \left.\left.\left.\boldsymbol{y}'\mathbf{X}_{\boldsymbol{\gamma}}\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)'\Sigma_*^{-1}\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)\right]\right]\right\}.
\end{aligned}$$

From equation (4.6), $\boldsymbol{\mu}_* = \Sigma_*(\mathbf{X}_{\boldsymbol{\gamma}}'\boldsymbol{y})$ and $\Sigma_* = (\mathbf{X}_{\boldsymbol{\gamma}}'\mathbf{X}_{\boldsymbol{\gamma}} + \mathbf{I}_p)^{-1}$. If these are

replaced in the above equation, the exponential term then becomes

$$\begin{aligned}
\exp\{q\} &= \exp\left\{-\tau\left[b - b_* + \frac{1}{2}\left[\boldsymbol{\beta}'_{\boldsymbol{\gamma}}\Sigma_*^{-1}\boldsymbol{\beta}_{\boldsymbol{\gamma}} + \boldsymbol{y}'\boldsymbol{y} - \boldsymbol{\beta}'_{\boldsymbol{\gamma}}\Sigma_*^{-1}\boldsymbol{\mu}_* - \boldsymbol{\mu}'_*\Sigma_*^{-1}\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \right.\right.\right.\\
&\quad \left.\left.\left.\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)'\Sigma_*^{-1}\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)\right]\right]\right\}\\
&= \exp\left\{-\tau\left[b - b_* + \frac{1}{2}\left[\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)'\Sigma_*^{-1}\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right) + \boldsymbol{y}'\boldsymbol{y} - \boldsymbol{\mu}'_*\Sigma_*^{-1}\boldsymbol{\mu}_* - \right.\right.\right.\\
&\quad \left.\left.\left.\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)'\Sigma_*^{-1}\left(\boldsymbol{\beta}_{\boldsymbol{\gamma}} - \boldsymbol{\mu}_*\right)\right]\right]\right\}\\
&= \exp\left\{-\tau\left[b - b_* + \frac{1}{2}\left[\boldsymbol{y}'\boldsymbol{y} - \boldsymbol{\mu}'_*\Sigma_*^{-1}\boldsymbol{\mu}_*\right]\right]\right\}\\
&= \exp\left\{-\tau\left[b - b_* + b_* - b\right]\right\}\\
&= \exp\{0\}\\
&= 1.
\end{aligned}$$

Notice that $b_* = \frac{1}{2}(\boldsymbol{y}'\boldsymbol{y} - \boldsymbol{\mu}'_*\Sigma_*^{-1}\boldsymbol{\mu}_*) + b$ so that q is zero and for this reason, $\exp\{q\} = 1$. Hence, the marginal likelihood takes the following form

$$p(\boldsymbol{y}|M_{\boldsymbol{\gamma}}) = \frac{b^a\Gamma(a_*)|\Sigma_*|^{1/2}}{b_*^{a_*}\Gamma(a)|\mathbf{I}_p|^{1/2}(2\pi)^{n/2}}. \tag{4.7}$$

Therefore, the marginal likelihood, given in (4.7), is available in closed form. In this case, everything is tractable and for this reason, the estimated posterior model probabilities,

$$\pi(M_{\boldsymbol{\gamma}}|\boldsymbol{y}) = \frac{p(M_{\boldsymbol{\gamma}})\,p(\boldsymbol{y}|M_{\boldsymbol{\gamma}})}{\sum\limits_{\boldsymbol{\gamma}} p(M_{\boldsymbol{\gamma}})\,p(\boldsymbol{y}|M_{\boldsymbol{\gamma}})},$$

that are calculated using the RJMCMC algorithm can then be compared to the exact target posterior model probabilities. Thus, this example can teach us how well the algorithm performs and which of the three strategies is best.

Three different types of move are used inside the RJMCMC in order to improve mixing. The first type of move jumps between models of different dimension changing dimensionality. Then, a fixed dimension move, called the exchange move is applied between models of the same dimension. In the exchange move, some of the variables in the current model are exchanged with some of the variables that are not included in the current model. For instance, if the model indicator of the current model is $\boldsymbol{\gamma} = (1, 0, 0, 0, 1, 1, 1)$ then the model indicator of the proposed model may be $\boldsymbol{\gamma}' = (1, 1, 0, 0, 1, 0, 1)$. Hence, variable x_6 is proposed to be exchanged with variable x_2. The proposed parameter vector for the exchange move is also generated from the full-conditional posterior distribution of the parameters according to each of the different updating schemes. The third type of move is also a fixed dimensional move that explores the current model and updates the current parameters using MCMC methods like the Metropolis-Hastings algorithm or the Gibbs sampler.

4.3.1.1 Proposal distributions for each of the updating schemes

It has already been shown that the full-conditional posterior distribution of the parameters is a Normal-Gamma distribution

$$\pi(\boldsymbol{\beta}_{\boldsymbol{\gamma}}, \tau | \boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) = N\text{Ga}(\boldsymbol{\mu}_*, \Sigma_*, a_*, b_*) \tag{4.8}$$

where $\boldsymbol{\mu}_* = \Sigma_* \mathbf{X}\boldsymbol{\gamma}' \boldsymbol{y}$, $\Sigma_* = (\mathbf{X}\boldsymbol{\gamma}' \mathbf{I}_p \mathbf{X}\boldsymbol{\gamma} + \mathbf{I}_p)^{-1}$, $a_* = a + \frac{n}{2}$ and $b_* = \frac{1}{2}(\boldsymbol{y}' \mathbf{I}_n \boldsymbol{y} - \boldsymbol{\mu}_*' \Sigma_*^{-1} \boldsymbol{\mu}_*) + b$. In the first scheme, scheme A, only the newly introduced parameter values $\mathbf{v}$ are generated from the full-conditional posterior distribution given in (4.9) while the common parameter values are retained in the proposed state.

$$\pi(\cdot|\boldsymbol{\beta}_{\boldsymbol{\psi}}, \tau, \boldsymbol{y}, M_{\boldsymbol{\gamma}'}) \sim N\left(\frac{\sum_{i=1}^{n} y_i x_{im} - \sum_{i=1}^{n} x_{im} \sum_{j \neq m}^{m} x_{ij} \beta_j}{\sum_{i=1}^{n} x_{im}^2 + 1}, \frac{\tau^{-1}}{\sum_{i=1}^{n} x_{im}^2 + 1} \right). \quad (4.9)$$

In the second scheme, scheme B, some of the common parameter values are retained in the proposed state of the Markov chain while the rest of them together with the new parameter values $\mathbf{v}$ are generated from the full-conditional posterior distribution given in (4.10).

$$\pi(\boldsymbol{\beta}_{\boldsymbol{\phi}}|\boldsymbol{\beta}_{\boldsymbol{\psi}'}, \tau, \boldsymbol{y}, \mathbf{X}\boldsymbol{\gamma}') \sim N\left(\bar{\boldsymbol{\mu}}, \frac{1}{\tau}\bar{\Sigma}\right) \quad (4.10)$$

where $\boldsymbol{\beta}_{\boldsymbol{\psi}'} = \{\beta_i : i \in \boldsymbol{\psi}'\}$ are the common parameter values that are conditioned on with $\dim(\boldsymbol{\beta}_{\boldsymbol{\gamma}'}) = \dim(\boldsymbol{\beta}_{\boldsymbol{\phi}}) + \dim(\boldsymbol{\beta}_{\boldsymbol{\psi}'})$ and $\boldsymbol{\phi}$ indicates which parameter values are going to be updated so that $\boldsymbol{\gamma}' = \boldsymbol{\phi} \cup \boldsymbol{\psi}'$.

If the posterior mean $\boldsymbol{\mu}_*$ and covariance matrix Σ_* of model $M_{\boldsymbol{\gamma}'}$ can be partitioned as

$$\boldsymbol{\mu}_* = (\boldsymbol{\mu}_{\boldsymbol{\phi}}, \boldsymbol{\mu}_{\boldsymbol{\psi}'})$$

and

$$\Sigma_* = \begin{bmatrix} \lambda_{\boldsymbol{\phi},\boldsymbol{\phi}} & \lambda_{\boldsymbol{\phi},\boldsymbol{\psi}'} \\ \lambda_{\boldsymbol{\psi}',\boldsymbol{\phi}} & \lambda_{\boldsymbol{\psi}',\boldsymbol{\psi}'} \end{bmatrix}.$$

where $\boldsymbol{\mu}_{\boldsymbol{\phi}}$ corresponds to posterior mean values of the parameters to be updated

and $\boldsymbol{\mu}_{\boldsymbol{\psi}'}$ to the posterior mean values of the parameters that are conditioned on, then $\bar{\boldsymbol{\mu}} = \boldsymbol{\mu}_{\boldsymbol{\phi}} - \lambda^{-1}_{\boldsymbol{\phi},\boldsymbol{\phi}}\lambda_{\boldsymbol{\phi},\boldsymbol{\psi}'}(\boldsymbol{\beta}_{\boldsymbol{\psi}'} - \boldsymbol{\mu}_{\boldsymbol{\psi}'})$ and $\bar{\Sigma} = \lambda_{\boldsymbol{\phi},\boldsymbol{\phi}} - \lambda_{\boldsymbol{\phi},\boldsymbol{\psi}'}\lambda^{-1}_{\boldsymbol{\psi}',\boldsymbol{\psi}'}\lambda_{\boldsymbol{\psi}',\boldsymbol{\phi}}$. Finally, in the third scheme, scheme C, the proposed parameter vector is generated from the posterior distribution of the parameters given in (4.8).

The performance of each of the algorithms is tested on 20 different simulated data sets where the response $\boldsymbol{y}$ is modeled as $\boldsymbol{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$. Each of the rows of the covariance matrix $\mathbf{X}$ is drawn from a $N(\mathbf{0}, \Sigma_\ell)$, for $\ell = 1, 2, 3$. When the covariance matrix $\Sigma_1 = \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{bmatrix}$, each of the data sets consists of uncorrelated predictive variables while when $\Sigma_2 = \begin{bmatrix} 1 & & 0.5 \\ & \ddots & \\ 0.5 & & 1 \end{bmatrix}$, it is the case of correlated predictive variables and when $\Sigma_3 = \begin{bmatrix} 1 & & 0.8 \\ & \ddots & \\ 0.8 & & 1 \end{bmatrix}$, it is the case of highly correlated predictive variables. The different degrees of covariance are used to test the performance of the algorithms in different structures of data. The error $\boldsymbol{\epsilon}$ is normally distributed so that $\boldsymbol{\epsilon} \sim N_n(\mathbf{0}, 1/2.5^2\mathbf{I}_n)$ and the parameter vector $\boldsymbol{\beta}$ is randomly generated for each data set from a $N(0.5, 0.1^2)$. Ten possible variables are considered and for this reason, the model space consists of $2^{10} - 1 = 1,023$ possible models. Here, the posterior model probabilities are known in closed form.

The RJMCMC algorithm is implemented in the following way. 40% of the time Gibbs updates are applied to update parameter values within different models since the full-conditional posterior distributions are tractable. Then, 20% of the time, the exchange move is implemented in order to promote mixing. Finally, the

remainder of the time, a jump move is used to make changes to dimensionality and thus, visiting models of different dimensions. Hence, the proposed parameter vector is generated according to one of the updating schemes. Note that no tuning is necessary since the proposal distribution is tractable and it is straightforward to generate the proposed parameter vector from it. An automatic algorithm has the advantage that its performance does not depend on the skills of the user. The algorithms run for $500,000$ iterations regarding the first $100,000$ as burn in. The estimated posterior model probabilities using the RJMCMC output are compared to the target ones through the following comparison criteria.

4.3.1.2 Comparison Criteria

Suppose that $p = \{p_1, \ldots, p_n\}$ is the target distribution and $q = \{q_1, \ldots, q_n\}$ is the estimated discrete distribution. Then the following metrics measure how different the estimated discrete distribution q and the true one are (Boone et al. (2005), Cripps et al. (2006)). For each data set the D_{L_1} and D_{L_2} metrics are weighted over the target distribution and they are calculated as

$$D_{L_1}(p, q) = \left[\sum_i (q_i - p_i)^2 p_i \right]^{1/2}$$

and

$$D_{L_2}(p, q) = \sum_i |q_i - p_i| p_i.$$

The weighted distances D_{L_1} and D_{L_2} take values between $[0, 1]$. If the estimated distribution is a good approximation to the target one, the distance values are

close to zero. However, poor approximations to the target distribution give distance values close to one. A non-weighted metric, the Hellinger distance, D_H, is also used. It is estimated as

$$D_H(p,q) = \left[2 - 2\sum_i [q_i p_i]^{1/2}\right]^{1/2}.$$

The Hellinger distance takes values between $[0, 2]$. Likewise, if the approximation to the target distribution is good, the Hellinger distance value is close to zero. On the other hand, if the estimated distribution is a poor approximation to the target one then the Hellinger distance takes values close to two.

Hence, if the estimated posterior model probabilities for each of the updating schemes are a good approximation to the target ones then each of the distances is close to zero. From Figure 4.2 and Figure 4.3, it seems that the three different updating Schemes give good approximations to the target distribution. Figure 4.2 indicates that Scheme C gives slightly better approximations compared to Scheme A and Scheme B. On the other hand, Scheme A and Scheme B seem to perform in a similar way. Recall that the weighted metrics give more importance to the target distribution while the Hellinger distance does not. In Figure 4.3, Scheme A and Scheme B still perform on a similar level while it is clear that Scheme C gives better approximations to the posterior model probabilities as the distance scores are closer to zero compared to the other two updating methods. Therefore, Scheme C appears to be the best strategy according to the metric distances.

Scheme C is sampling most directly from the posterior distribution over model

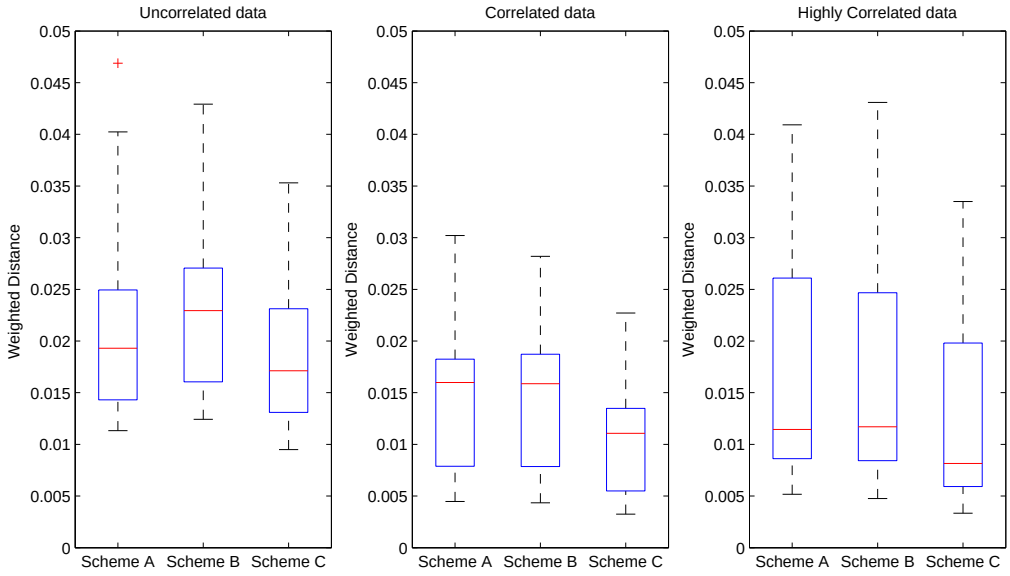


Figure 4.2: (a) Weighted distance for each of the updating schemes using uncorrelated data, (b) correlated data and (c) highly correlated data.

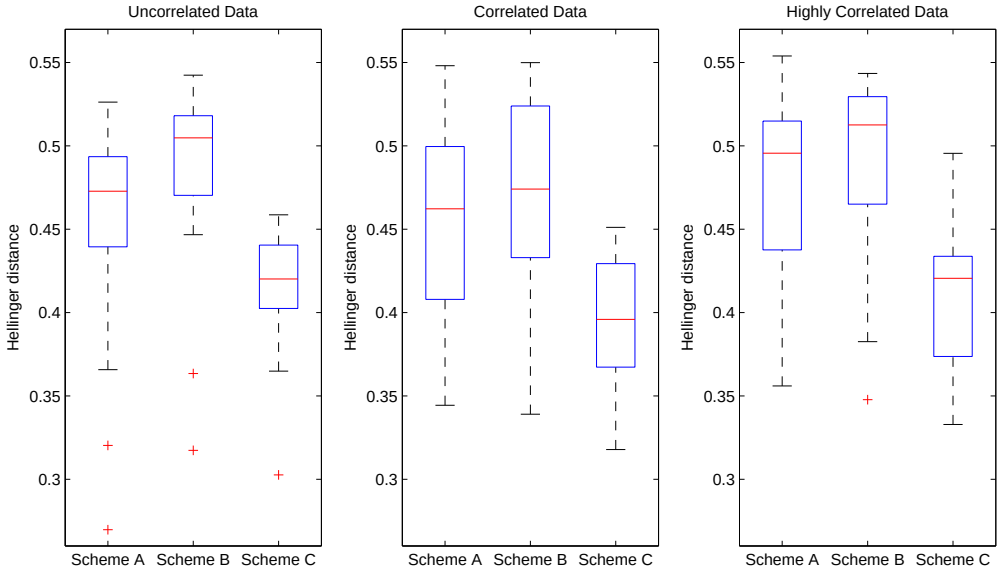


Figure 4.3: (a) Hellinger distance for each of the updating schemes using uncorrelated data, (b) correlated data and (c) highly correlated data.

space since the acceptance probability of the Reversible jump move is simply the ratio of posterior model probabilities of the proposed and current models. Thus, the method could be regarded as a Markov chain searching over model space alone since the parameters have been essentially integrated out. Thus, Scheme C should provide an idea of the efficiency of the proposed updating algorithm, i.e how model space is explored in terms of adding or removing a single parameter and the exchange steps.

The most probable models are presented for a randomly selected sample following Table 4.1, Table 4.2 and Table 4.3.

The schemes generally find the most probable models in the correct order and the Bayes factors appear to be estimated quite accurately. Longer simulations would appear to be necessary to ensure convergence of the posterior model probabilities to their correct values. This may be as a result of slow mixing over models with very low probability, particularly with a reasonably large number of possible models (currently 1023). Scheme C seems to be the best strategy compared to the other ones. When Scheme C is implemented, block updates are used to propose the new parameter vector. In this case, the acceptance probability does not depend on the dimension of the parameter vector since it is proportional to the Bayes factor and for this reason, the estimated posterior model probabilities are expected to be better. Nevertheless, Scheme A and Scheme B also give good estimates and most of the time they find the most probable models in the correct order.

For Table 4.1, Table 4.2 and Table 4.3, the target posterior model probabilities are calculated together with the estimated posterior model probabilities using the RJMCMC algorithms for uncorrelated, correlated and highly correlated predictor

variables. The Bayes factor is also evaluated for each of the data sets that are described in Table 4.1, Table 4.2 and Table 4.3. The three most probable models are compared using the Bayes factor, see Table 4.4 for uncorrelated, correlated and highly correlated data. In general, Scheme C gives better estimates to the Bayes factor compared of the other two updating schemes.

The acceptance probabilities for the exchange and jump move using each of the different scenarios are shown in Table 4.5. Looking at Scheme A and Scheme B, it appears that the acceptance probability decreases as the correlation in the predictors increases. On the other hand, for Scheme C the acceptance probability increases when the correlation between the data increases. The probability mass may be concentrated in just a few models for uncorrelated data, while, when the correlation between the data increases, the probability mass may be spread over more models. Hence, if the predictors are highly correlated, when a jump is proposed to a new model, there may be a higher probability that the jump will be accepted. This may be the reason why the acceptance probability for Scheme C increases together with the correlation of the predictors. Then again, Scheme A and Scheme B seem to be influenced by the structure of the data since as the correlation increases the proposed jumps are not accepted as often. However, it would be expected that the proposed jump steps would be accepted more frequently as the probability mass is spread over to different models.

In conclusion, this is just a teaching example since there is no need to use the RJMCMC algorithm to estimate the posterior model probabilities for a relatively small number of models because they are known in closed form. However, one can learn a lot from the performance of the algorithms because the estimated posterior model probabilities within the RJMCMC algorithm can be compared

True	Model	Scheme A	Scheme B	Scheme C
0.0625	x_9	0.1238	0.1370	0.1186
0.0518	x_4	0.1032	0.1133	0.0956
0.0436	x_{10}	0.0831	0.0952	0.0855
0.0340	x_6	0.0744	0.0768	0.0664
0.0334	x_1	0.0709	0.0747	0.0662
0.0309	x_8	0.0679	0.0668	0.0624
0.0291	x_5	0.0591	0.0640	0.0571
0.0252	x_9, x_{10}	0.0124	0.0121	0.0143
0.0252	x_7	0.0564	0.0553	0.0517
0.0227	x_3	0.0528	0.0503	0.0488
0.0199	x_2	0.0466	0.0418	0.0422
0.0194	x_4, x_9	0.0101	0.0097	0.0112
0.0157	x_4, x_{10}	0.0081	0.0080	0.0085
0.0144	x_1, x_9	0.0078	0.0068	0.0084
0.0111	x_8, x_9	0.0063	0.0053	0.0069
0.0109	x_6, x_9	0.0063	0.0053	0.0068
0.0104	x_5, x_9	0.0052	0.0052	0.0064
0.0102	x_4, x_6	0.0058	0.0051	0.0056
0.0100	x_7, x_9	0.0056	0.0054	0.0063
0.0097	x_4, x_5	0.0052	0.0049	0.0061
0.0097	x_4, x_9, x_{10}	0.0020	0.0018	0.0022
0.0095	x_5, x_{10}	0.0050	0.0045	0.0058
0.0093	x_1, x_4	0.0049	0.0042	0.0055
0.0092	x_4, x_8	0.0055	0.0047	0.0057
0.4722	Rest	0.1718	0.1419	0.2059

Table 4.1: Posterior model probabilities for each of the updating Schemes using uncorrelated predictive variables.

True	Model	Scheme A	Scheme B	Scheme C
0.1582	x_4, x_5	0.3128	0.3006	0.2517
0.0523	x_2, x_4, x_5	0.0356	0.0372	0.0941
0.0491	x_4, x_5, x_6	0.0377	0.0372	0.0339
0.0370	x_3, x_4, x_5	0.0260	0.0282	0.0273
0.0336	x_4, x_5, x_7	0.0270	0.0270	0.0227
0.0312	x_4, x_5, x_9	0.0214	0.0252	0.0216
0.0306	x_4, x_5, x_8	0.0215	0.0235	0.0223
0.0277	x_1, x_4, x_5	0.0218	0.0219	0.0220
0.0261	x_4, x_5, x_{10}	0.0207	0.0206	0.0201
0.0236	x_3, x_4, x_5, x_6	0.0109	0.0117	0.0100
0.0141	x_2, x_3, x_4, x_5	0.0069	0.0072	0.0063
0.0135	x_2, x_4, x_5, x_6	0.0056	0.0055	0.0057
0.0131	x_4, x_5, x_6, x_9	0.0049	0.0055	0.0064
0.0127	x_1, x_4, x_5, x_6	0.0067	0.0056	0.0067
0.0110	x_3, x_4, x_5, x_7	0.0053	0.0055	0.0051
0.0109	x_2, x_4, x_5, x_{10}	0.0049	0.0050	0.0057
0.0104	x_2, x_4, x_5, x_9	0.0043	0.0044	0.0052
0.0103	x_1, x_2, x_4, x_5	0.0050	0.0048	0.0056
0.0102	x_2, x_4, x_5, x_8	0.0047	0.0051	0.0050
0.0101	x_2, x_4, x_5, x_7	0.0044	0.0045	0.0048
0.0101	x_4, x_5, x_6, x_8	0.0050	0.0053	0.0053
0.0095	x_4, x_5, x_6, x_7	0.0055	0.0048	0.0047
0.0090	x_5	0.1005	0.1057	0.0680
0.0083	x_4, x_5, x_6, x_{10}	0.0043	0.0041	0.0046
0.3773	Rest	0.2966	0.2940	0.3952

Table 4.2: Posterior model probabilities for each of the updating Schemes using correlated predictive variables.

True	Model	Scheme A	Scheme B	Scheme C
0.0465	x_1, x_7	0.1184	0.0981	0.0648
0.0246	x_2, x_7	0.0422	0.0522	0.0367
0.0223	x_1, x_5, x_7	0.0186	0.0187	0.0334
0.0205	x_1, x_6, x_7	0.0193	0.0179	0.0132
0.0205	x_1, x_7, x_{10}	0.0184	0.0178	0.0129
0.0203	x_1, x_2, x_7	0.0130	0.0160	0.0131
0.0151	x_1, x_7, x_8	0.0123	0.0119	0.0099
0.0136	x_1, x_4, x_7	0.0103	0.0109	0.0091
0.0126	x_1, x_6	0.0320	0.0298	0.0200
0.0120	x_1, x_4, x_6, x_7	0.0059	0.0053	0.0053
0.0110	x_1, x_5	0.0219	0.0226	0.0175
0.0097	x_1, x_3, x_7	0.0079	0.0078	0.0057
0.0094	x_1, x_7, x_9	0.0087	0.0085	0.0071
0.0086	x_2, x_7, x_{10}	0.0062	0.0076	0.0065
0.0084	x_2, x_5, x_7	0.0069	0.0076	0.0058
0.0078	x_2, x_6, x_7	0.0060	0.0067	0.0061
0.0074	x_1, x_7, x_8	0.0063	0.0054	0.0049
0.0074	x_1, x_{10}	0.0183	0.0157	0.0125
0.0074	x_1, x_2, x_5, x_7	0.0025	0.0033	0.0036
0.0073	x_1, x_4, x_7, x_{10}	0.0038	0.0033	0.0035
0.0072	x_1, x_5, x_7, x_{10}	0.0042	0.0040	0.0036
0.0069	x_6, x_7	0.0184	0.0158	0.0120
0.0068	x_1, x_5, x_6, x_7	0.0043	0.0032	0.0035
0.0068	x_1, x_5, x_6	0.0063	0.0052	0.0051
0.6802	Rest	0.5879	0.6047	0.7041

Table 4.3: Posterior model probabilities for each of the updating Schemes using highly correlated predictive variables.

Bayes Factor	True	Scheme A	Scheme B	Scheme C
Uncorrelated Data				
$B_{1,2}$	1.2072	1.2005	1.2409	1.2092
$B_{2,3}$	1.1881	1.2418	1.1182	1.1903
Correlated Data				
$B_{1,2}$	3.0234	8.7898	8.0893	2.6748
$B_{2,3}$	1.0661	0.9450	0.9995	1.0060
High Correlated Data				
$B_{1,2}$	1.8923	2.8052	1.8793	1.7655
$B_{2,3}$	1.1035	2.2747	2.7860	1.0988

Table 4.4: Bayes Factor values for the three most probable models presented in Table 4.1, Table 4.2 and Table 4.3.

Data	Uncorrelated			Correlated			High Correlated		
Accept prob,	jump	exch	total	jump	exch	total	jump	exch	total
Scheme A	0.2130	0.4222	0.2827	0.2020	0.2306	0.2116	0.1812	0.1962	0.1862
Scheme B	0.2112	0.4621	0.2948	0.2311	0.3264	0.2629	0.2198	0.3022	0.2473
Scheme C	0.2571	0.4814	0.3318	0.3347	0.4168	0.3621	0.3385	0.4017	0.3596

Table 4.5: Acceptance probabilities for the exchange and jump moves using uncorrelated, correlated and highly correlated predictive variables for each of the updating schemes. The total acceptance probabilities set are also presented.

to the true ones. It seems that using the full-conditional posterior distribution of the parameters within the model as the proposal distribution to generate the proposed parameter vector may be the best strategy since the estimated posterior model probabilities are very close to the true ones especially when Scheme C is applied. When Scheme C is used, the acceptance ratio is independent of the parameter values and proportional to the Bayes factor for the proposed model versus the current one since it is assumed that each of the candidate models has the same probability of being selected. Thus, when the whole proposed

parameter vector is generated from the posterior distribution of the parameters within the model, the approximated posterior model probabilities are closer to the target ones compared to the other methods. Hence, we believe that applying Gibbs updates inside the RJMCMC algorithm is a good strategy, especially when block updates are implemented inside the RJMCMC algorithm regarding a linear regression variable selection problem.

4.3.2 Autoregressive model choice

The performance of each of the updating schemes is also gauged using an autoregressive time series model choice problem. Suppose that data $x_1, \ldots, x_N$ are simulated from an autoregressive process of unknown order. Thus, model M_k corresponds to the k^{th} order autoregressive process which is modelled as

$$x_t = \sum_{i=1}^{k} \alpha_i x_{t-i} + \epsilon_t$$

where $t = k+1, \ldots, N$ and $\epsilon \sim N(0, 1/\tau)$, with τ representing the precision of the residual. A uniform prior is assumed for k while within model M_k, an independent prior is assumed for each of the coefficients so that $\alpha_i \sim N(0, 1/\tau_\alpha)$ and a Gamma distribution for the precision τ that is $\tau \sim G(a, b)$. Thus, the prior distributions are

$$\begin{aligned} p(\tau) &= \frac{1}{b^a \Gamma(a)} \tau^{a-1} \exp\left\{-\frac{\tau}{b}\right\} \\ &\propto \tau^{a-1} \exp\left\{-\frac{\tau}{b}\right\}, \end{aligned}$$

and

$$\begin{aligned} p(\boldsymbol{\alpha}) &= \prod_{i=1}^{k} \left(\frac{\tau_\alpha}{2\pi}\right)^{1/2} \exp\left\{-\frac{\tau_\alpha}{2}\alpha_i^2\right\} \\ &\propto \exp\left\{-\frac{\tau_\alpha}{2}\sum_{i=1}^{k}\alpha_i^2\right\}. \end{aligned}$$

The likelihood of the data $\boldsymbol{x}$ for model M_k is

$$\begin{aligned} L(\boldsymbol{x}|\boldsymbol{\alpha},\tau,M_k) &= \prod_{t=k+1}^{N} \left(\frac{\tau}{2\pi}\right)^{1/2} \exp\left\{-\frac{\tau}{2}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2\right\} \\ &\propto \tau^{\frac{N-k}{2}} \exp\left\{-\frac{\tau}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2\right\}. \end{aligned}$$

Then, the posterior distribution within model is arrived at by using Bayes' Theorem

$$\begin{aligned} \pi(\boldsymbol{\alpha},\tau,M_k|\boldsymbol{x}) &\propto L(\boldsymbol{x}|\boldsymbol{\alpha},\tau,M_k)p(\boldsymbol{\alpha})p(\tau) \\ &\propto \tau^{\frac{N-k}{2}} \exp\left\{-\frac{\tau}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2\right\} \exp\left\{-\frac{\tau_\alpha}{2}\sum_{i=1}^{k}\alpha_i^2\right\} \\ &\quad \tau^{a-1}\exp\left\{-\frac{\tau}{b}\right\} \\ &= \tau^{\frac{N-k}{2}+a-1} \exp\left\{-\frac{\tau}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2 - \frac{\tau_\alpha}{2}\sum_{i=1}^{k}\alpha_i^2 - \frac{\tau}{b}\right\}. \end{aligned}$$

Thus, the full-conditional posterior probabilities are defined as

$$\pi(\tau|\boldsymbol{\alpha},\boldsymbol{x},M_k) \propto \tau^{\frac{N-k}{2}+a-1} \exp\left\{-\tau\left[\frac{1}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2 + \frac{1}{b}\right]\right\}$$

Therefore,

$$\tau|\boldsymbol{\alpha}, \boldsymbol{x}, M_k \sim G\left(\frac{N-k}{2} + a, \frac{1}{\frac{1}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2 + \frac{1}{b}}\right).$$

Likewise, the full-conditional posterior distribution for each of the autoregressive coefficients is

$$\begin{aligned}\pi(\alpha_i|\tau, \boldsymbol{x}, M_k) &\propto \exp\left\{-\frac{\tau}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2 - \frac{\tau_\alpha}{2}\sum_{i=1}^{k}\alpha_i^2\right\} \\ &= \exp\left\{-\frac{\tau}{2}\sum_{t=k+1}^{N}\left[-2x_t\alpha_i x_{t-i} + \left(\sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2\right] - \frac{\tau_\alpha}{2}\alpha_i^2\right\}. \quad (4.11)\end{aligned}$$

Notice that,

$$\begin{aligned}\left(\sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2 &= \left(\alpha_i x_{t-i} + \sum_{\substack{j=1 \\ j\neq i}}^{k}\alpha_j x_{t-j}\right)^2 \\ &= \alpha_i^2 x_{t-i}^2 + 2\alpha_i x_{t-i}\sum_{\substack{j=1 \\ j\neq i}}^{k}\alpha_j x_{t-j} + \left(\sum_{\substack{j=1 \\ j\neq i}}^{k}\alpha_j x_{t-j}\right)^2 \\ &\propto \alpha_i^2 x_{t-i}^2 + 2\alpha_i x_{t-i}\sum_{\substack{j=1 \\ j\neq i}}^{k}\alpha_j x_{t-j} \quad (4.12)\end{aligned}$$

If we use (4.12) in equation (4.11), the full-conditional posterior distribution of coefficient α_i is

$$\begin{aligned}
\pi(\alpha_i|\tau,\boldsymbol{x},M_k) &\propto \exp\left\{-\frac{\tau}{2}\sum_{t=k+1}^{N}\left[-2x_t\alpha_i x_{t-i}+\alpha_i^2x_{t-i}^2+2\alpha_i x_{t-i}\sum_{\substack{j=1\\j\neq i}}^{k}\alpha_j x_{t-j}\right]-\frac{\tau_\alpha}{2}\alpha_i^2\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[-2\tau\alpha_i\sum_{t=k+1}^{N}x_tx_{t-i}+\alpha_i^2\tau\sum_{t=k+1}^{N}x_{t-i}^2+\right.\right.\\
&\qquad \left.\left.2\alpha_i\tau\sum_{t=k+1}^{N}x_{t-i}\sum_{\substack{j=1\\j\neq i}}^{k}\alpha_jx_{t-j}+\frac{\tau_\alpha}{2}\alpha_i^2\right]\right\}\\
&= \exp\left\{-\frac{1}{2}\left[\alpha_i^2\left(\tau\sum_{t=k+1}^{N}x_{t-i}^2+\tau_\alpha\right)-2\alpha_i\left(\tau\sum_{t=k+1}^{N}x_tx_{t-i}-\right.\right.\right.\\
&\qquad \left.\left.\left.\tau\sum_{t=k+1}^{N}x_{t-i}\sum_{\substack{j=1\\j\neq i}}^{k}\alpha_jx_{t-j}\right)\right]\right\}.
\end{aligned}$$

Hence,

$$\alpha_i|\tau,\boldsymbol{\alpha}_{\setminus i},\boldsymbol{x},M_k\sim N\left(\frac{\tau\sum\limits_{t=k+1}^{N}x_tx_{t-i}-\tau\sum\limits_{t=k+1}^{N}x_{t-i}\sum\limits_{\substack{j=1\\j\neq i}}^{k}\alpha_jx_{t-j}}{\tau\sum\limits_{t=k+1}^{N}x_{t-i}^2+\tau_\alpha},\frac{1}{\tau\sum\limits_{t=k+1}^{N}x_{t-i}^2+\tau_\alpha}\right).$$

In this case, the full-conditional posterior densities of the parameters are tractable. Thus, it is straightforward to draw the proposed parameter vector from them. Unfortunately, the marginal likelihood is not known in closed form. The power posterior method of Friel and Pettitt (2008) is used to estimate the marginal likelihood for each model by running a very long Markov chain. Then,

these estimates are used in turn to approximate the posterior model probabilities. However, to estimate accurately the posterior model probabilities requires a large computational time. Clearly, both the power posterior and RJMCMC methods give estimates of the posterior model probabilities. However, considerably more computational time is given to the power posterior method, so that it may be viewed as a benchmark against which to compare the RJMCMC estimates. Hence, the posterior model probabilities that are estimated using the RJMCMC algorithm may be compared to the good estimates found using the power posterior method. For this reason, conclusions can be made on the performance of the algorithm using each of the updating schemes. The power posterior method is described below.

4.3.2.1 Marginal likelihood estimation using power posterior

Friel and Pettitt (2008) estimate the intractable marginal likelihood using path integration. The marginal likelihood is calculated based on samples from the so called power posterior distribution $\pi(\boldsymbol{\theta}|\boldsymbol{y}, T)$. The power posterior is proportional to the prior distribution of the parameters $p(\boldsymbol{\theta})$ times the likelihood raised to a power T, $L(\boldsymbol{y}|\boldsymbol{\theta})^T$ so that $\pi(\boldsymbol{\theta}|\boldsymbol{y}, T) \propto L(\boldsymbol{y}|\boldsymbol{\theta})^T \pi(\boldsymbol{\theta}|\boldsymbol{y}, T)$ where T is a tempering parameter and takes values in $[0, 1]$. Then, the normalising constant for $\pi(\boldsymbol{\theta}|\boldsymbol{y}, T)$ is $z(\boldsymbol{y}|T) = \int\limits_{\boldsymbol{\theta}} L(\boldsymbol{y}|\boldsymbol{\theta})^T p(\boldsymbol{\theta}) d\boldsymbol{\theta}$ so that

$$\pi(\boldsymbol{\theta}|\boldsymbol{y}, T) = \frac{L(\boldsymbol{y}|\boldsymbol{\theta})^T p(\boldsymbol{\theta})}{z(\boldsymbol{y}|T)}.$$

Notice that for $T = 0$, the marginal likelihood is $z(\boldsymbol{y}|T = 0) = \int\limits_{\boldsymbol{\theta}} p(\boldsymbol{\theta}) d\boldsymbol{\theta} = 1$

and for $T = 1$, it is $z(\boldsymbol{y}|T = 1) = \int_{\boldsymbol{\theta}} L(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta} = p(\boldsymbol{y})$ which is the marginal likelihood of the data. Thus, $z(\boldsymbol{y}|T = 1)$ is the integral of interest. It is shown in Friel and Pettitt (2008) that

$$\log p(\boldsymbol{y}) = \int_0^1 E_{\boldsymbol{\theta}|\boldsymbol{y},T} \left[\log L(\boldsymbol{y}|\boldsymbol{\theta})\right] dT. \tag{4.13}$$

Thus, the marginal likelihood is the integral over the temperature T of half the mean deviance, where the expectation is taken with respect to the power posterior. In practice, this integral is approximated by discretising the integral at points $T_i \in [0, 1]$ and different chains are sampled from the power posterior at temperature T_i then a sample is drawn from $E_{\boldsymbol{\theta}|\boldsymbol{y},T_i} \left[\log L(\boldsymbol{y}|\boldsymbol{\theta})\right]$. Hence, the log marginal likelihood is approximated as

$$\log p(\boldsymbol{y}) \approx \sum_{i=0}^{n-1} (T_{i+1} - T_i) \frac{E_{\boldsymbol{\theta}|\boldsymbol{y},T_{i+1}} \left[\log L(\boldsymbol{y}|\boldsymbol{\theta})\right] + E_{\boldsymbol{\theta}|\boldsymbol{y},T_i} \left[\log L(\boldsymbol{y}|\boldsymbol{\theta})\right]}{2}.$$

The partition is often chosen as $T_i = (i/n)^c$, where n is the number of points in the tempering scale in $[0, 1]$ and $c > 1$. This choice provides a better approximation to the integral through using quadrature.

The power posterior method described earlier on is now applied to an autoregressive model of order k. The first step is to estimate the power posterior

distribution such as

$$\begin{aligned}
\pi(\boldsymbol{\alpha}, \tau|\boldsymbol{x}, T, M_k) &\propto L(\boldsymbol{x}|\boldsymbol{\alpha})^T p(\boldsymbol{\alpha}) p(\tau) \\
&\propto \tau^{T\frac{N-k}{2}} \exp\left\{-\frac{T\tau}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2\right\} \exp\left\{-\frac{\tau_\alpha}{2}\sum_{i=1}^{k}\alpha_i^2\right\} \\
&\quad \tau^{a-1}\exp\left\{-\frac{\tau}{b}\right\} \\
&= \tau^{\frac{T(N-k)}{2}+a-1} \exp\left\{-\frac{T\tau}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2 - \frac{\tau_\alpha}{2}\sum_{i=1}^{k}\alpha_i^2 - \frac{\tau}{b}\right\}.
\end{aligned}$$

Thus, the full-conditional posterior distribution of the precision τ is

$$\pi(\tau|\boldsymbol{\alpha}, \boldsymbol{x}, T, M_k) \propto \tau^{\frac{T(N-k)}{2}+a-1} \exp\left\{-\tau\left[\frac{T}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2 + \frac{1}{b}\right]\right\}$$

and therefore,

$$\tau|\boldsymbol{\alpha}, \boldsymbol{x}, T, M_k \sim G\left(\frac{T(N-k)}{2} + a, \frac{1}{\frac{T}{2}\sum_{t=k+1}^{N}\left(x_t - \sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2 + \frac{1}{b}}\right).$$

Likewise the full-conditional posterior distribution for each of the i^{th} coefficients

is

$$\begin{aligned}
\pi(\alpha_i|\tau,\boldsymbol{x},T,M_k) &\propto \exp\left\{-\frac{T\tau}{2}\sum_{t=k+1}^{N}\left(x_t-\sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2-\frac{\tau_\alpha}{2}\sum_{i=1}^{k}\alpha_i^2\right\}\\
&= \exp\left\{-\frac{T\tau}{2}\sum_{t=k+1}^{N}\left[-2x_t\alpha_i x_{t-i}+\left(\sum_{i=1}^{k}\alpha_i x_{t-i}\right)^2\right]-\frac{\tau_\alpha}{2}\alpha_i^2\right\}\\
&\propto \exp\left\{-\frac{T\tau}{2}\sum_{t=k+1}^{N}\left[-2x_t\alpha_i x_{t-i}+\alpha_i^2x_{t-i}^2+2\alpha_i x_{t-i}\sum_{\substack{j=1\\j\neq i}}^{k}\alpha_j x_{t-j}\right]-\frac{\tau_\alpha}{2}\alpha_i^2\right\}\\
&= \exp\left\{-\frac{1}{2}\left[\alpha_i\left(-2T\tau\sum_{t=k+1}^{N}x_t x_{t-i}+2T\tau\sum_{t=k+1}^{N}x_{t-i}\sum_{\substack{j=1\\j\neq i}}^{k}\alpha_j x_{t-j}\right)-\right.\right.\\
&\qquad \left.\left.\alpha_i^2\left(T\tau\sum_{t=k+1}^{N}x_{t-i}^2+\tau_\alpha\right)\right]\right\}\\
&= \exp\left\{-\frac{1}{2}\left[\alpha_i^2\left(T\tau\sum_{t=k+1}^{N}x_{t-i}^2+\tau_\alpha\right)-2\alpha_i\left(T\tau\sum_{t=k+1}^{N}x_t x_{t-i}-\right.\right.\right.\\
&\qquad \left.\left.\left.T\tau\sum_{t=k+1}^{N}x_{t-i}\sum_{\substack{j=1\\j\neq i}}^{k}\alpha_j x_{t-j}\right)\right]\right\}.
\end{aligned}$$

Hence, the posterior full-conditional distribution of the parameter α_i is

$$\alpha_i|\tau,\boldsymbol{x},T,M_k \sim N\left(\frac{T\tau\sum\limits_{t=k+1}^{N}x_t x_{t-i}-T\tau\sum\limits_{t=k+1}^{N}x_{t-i}\sum\limits_{\substack{j=1\\j\neq i}}^{k}\alpha_j x_{t-j}}{T\tau\sum\limits_{t=k+1}^{N}x_{t-i}^2+\tau_\alpha},\frac{1}{T\tau\sum\limits_{t=k+1}^{N}x_{t-i}^2+\tau_\alpha}\right).$$

We simulate 20 different data sets from an autoregressive time series model of order three to test the performance of the three different algorithms. Each of the data sets consists of 100 responses. The coefficient values are randomly chosen from a Uniform distribution $U(-1, 1)$ and the error is Normally distributed as $\epsilon \sim N(0, 1)$ for the first data set while in the second data set the error is distributed as $\epsilon \sim N(0, 2^2)$. Then, it is assumed that the prior distribution for the precision τ is Gamma with shape and scale one. Two different prior distributions for the coefficients are used, a $N(0, 2^2)$ and a $N(0, 1)$. Each of the 20 different data sets is similar to the ones that are presented in Figure 4.4.

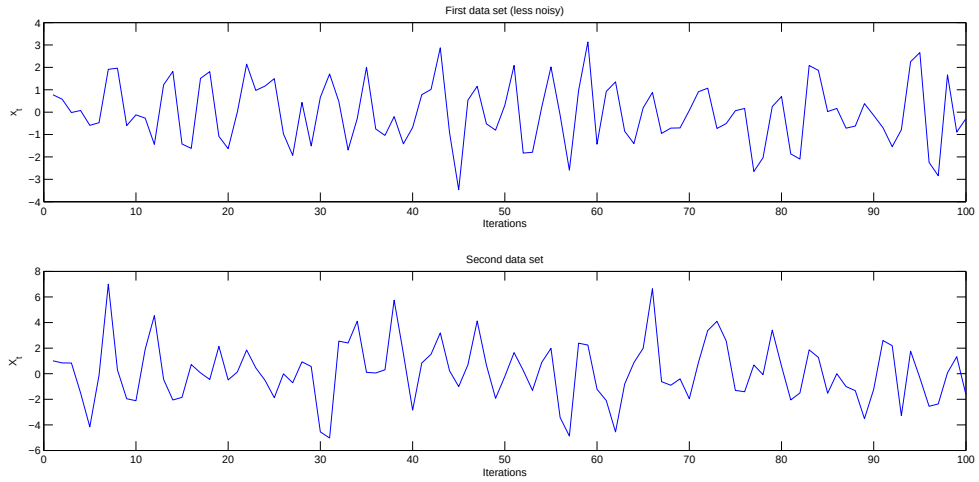


Figure 4.4: Autoregressive process of order three with error $\epsilon \sim N(0, 1)$ and with error $\epsilon \sim N(0, 4)$.

The actual posterior model probabilities are not known in closed form and for this reason, the power posterior method of Friel and Pettitt (2008) is used to estimate them by running the Markov chain for sufficiently long time. Then, these estimates are considered as the "true" posterior model probabilities. For the power posterior method, the Gibbs sampler samples from the power posterior

distribution since in this case, the full-conditional posterior distributions are available. We run $10,000$ Gibbs updates for each temperature while the tempering scale is partitioned as $T_i = (i/55)^3$ where $i = 1, \ldots, 55$. In total, $550,000$ iterations are used to estimate the marginal likelihood for each model. The expected log power posterior values for a randomly selected model versus the temperature t are shown in Figure 4.5. Notice that Figure 4.5 is similar for each model and each data set using either of the two different priors. It also indicates that the algorithm converges to the actual target values. An informative prior will always give a less steep curve.

Hence, the posterior model probabilities that are estimated using the RJMCMC output for each of the three different updating schemes are compared to the posterior model probabilities that are calculated using the power posterior method. Then, conclusions on the performance of the algorithm can be made. The RJMCMC algorithm runs for $105,000$ iterations regarding the first $5,000$ as burn in. It is assumed that the maximum order of the process is 10 and for this reason, there are ten possible models. Notice that having a fairly small number of models allows us to compute the posterior model probabilities using the power posterior method. Otherwise, it will be computationally impossible to approximate them. The acceptance probabilities for each of the different schemes are presented in Table 4.6. It seems that Scheme C gives the higher acceptance ratio compared to the other updating schemes.

The estimated posterior model probabilities are compared to the target ones through the metrics that were described in section 4.3.1.2. Figure 4.6 presents the weighted distance between the estimated posterior model probabilities and the target ones. In general, the estimated posterior model probabilities are a good

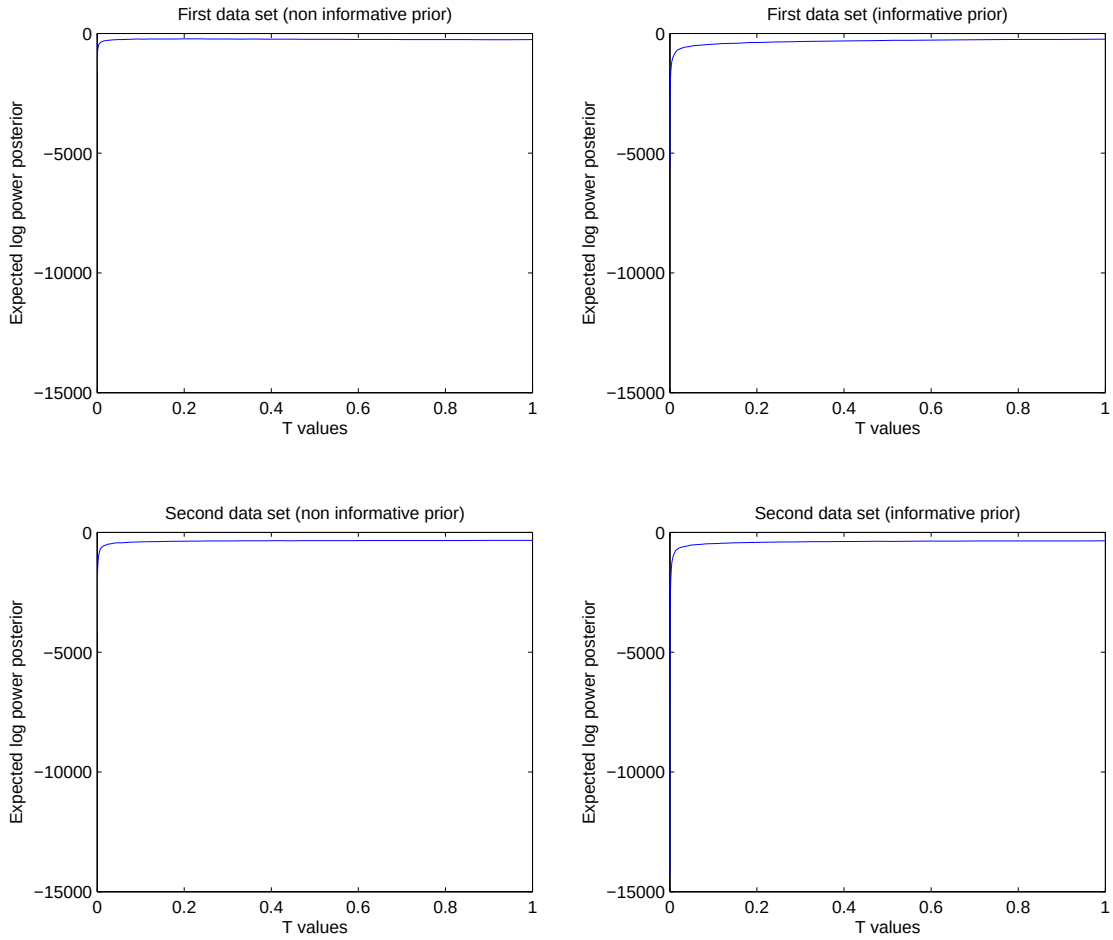


Figure 4.5: Expected values of the log power posterior versus temperature T for each data set using the different priors.

Data	First data set		Second data set	
Accept Prob	Non informative prior	Informative prior	Non informative prior	Informative prior
Scheme A	0.1813	0.1472	0.0877	0.1164
Scheme B	0.2578	0.2502	0.1213	0.1834
Scheme C	0.3386	0.3879	0.1612	0.2631

Table 4.6: Acceptance probabilities for each data set using different priors.

approximation to the target ones when the distance is close to zero. The Hellinger distance has also been transformed to take values in $[0, 1]$. From the boxplots, the three different updating schemes give good approximations to the posterior model probabilities although Scheme C gives slightly better approximations. In

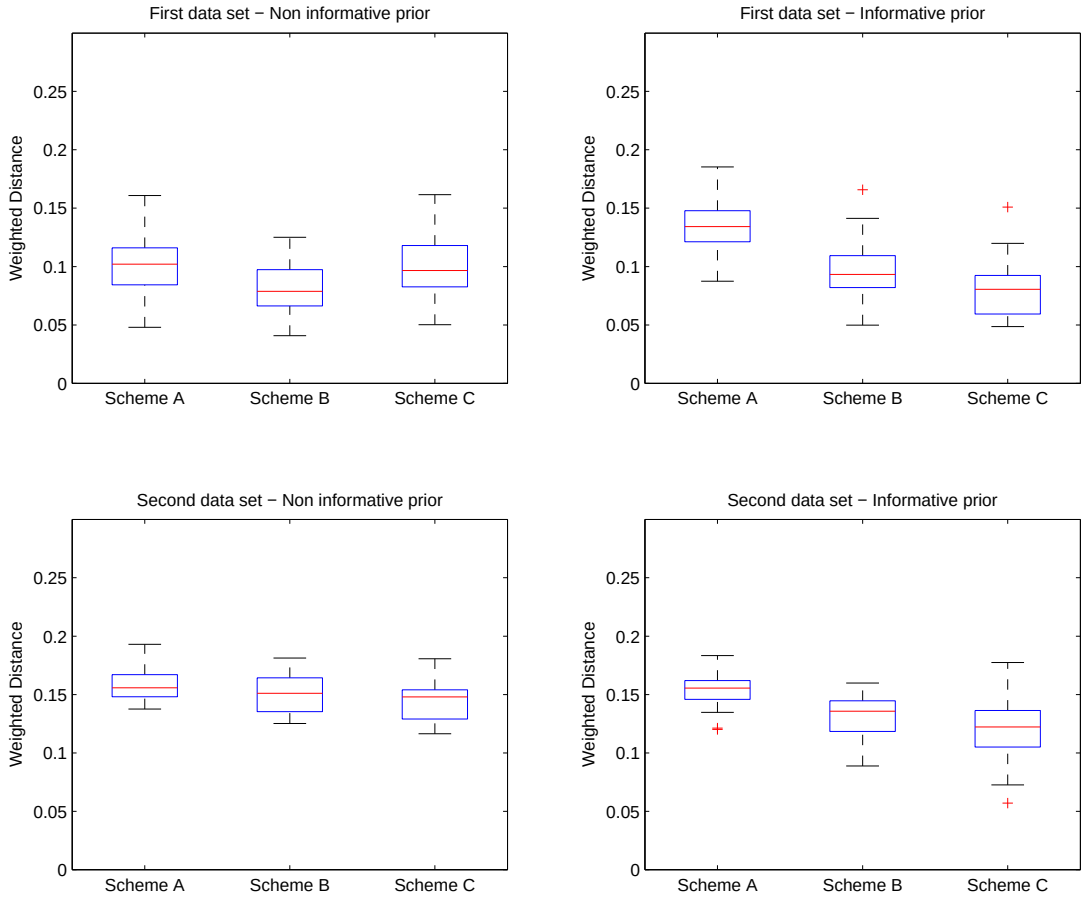


Figure 4.6: Weighted distance for each data set using different priors.

Figure 4.7, the Hellinger distance is presented. In general, all the schemes are good approximations to the target distribution but again Scheme C gives slightly better estimations compared to the other two. Scheme B also approximates quite

well the posterior model probabilities.

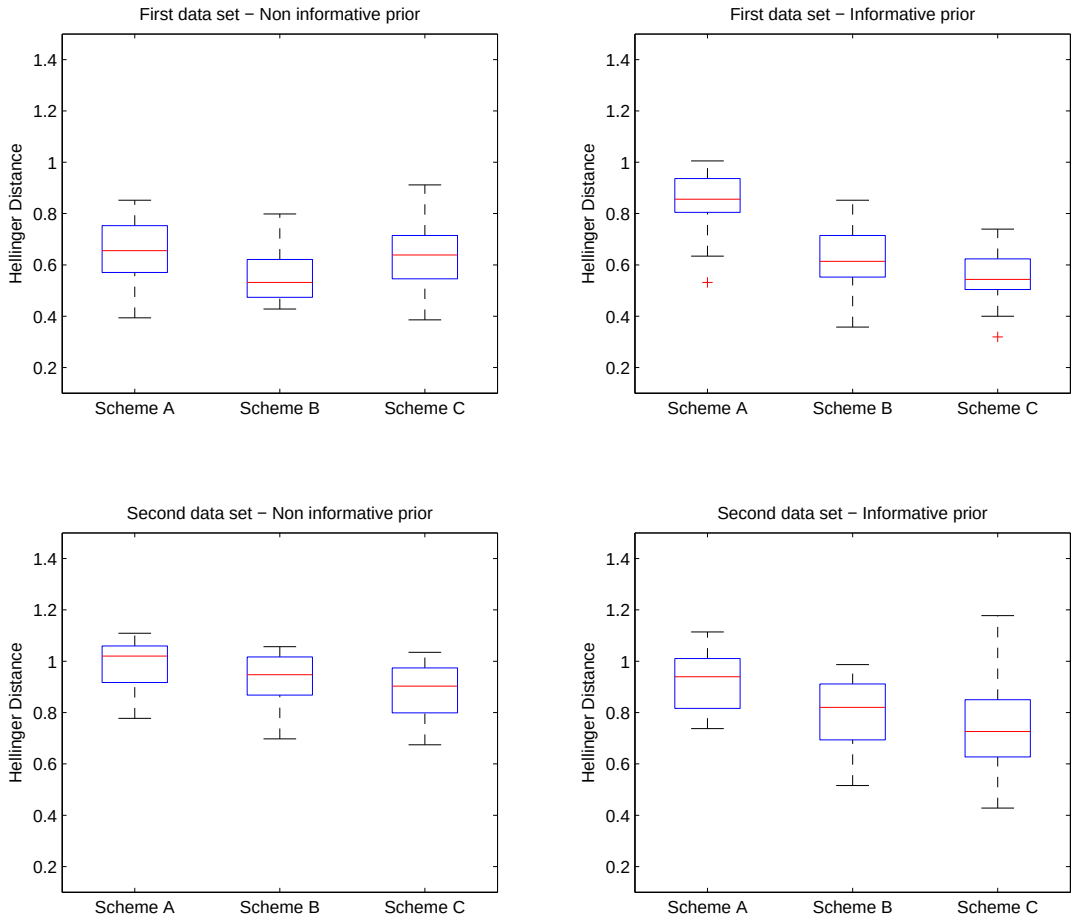


Figure 4.7: Hellinger distance for each data set using different priors.

The Bayes factor is estimated to compare the two most probable models of a randomly selected data set. Thus, in Table 4.7, the posterior model probabilities of the two most probable models are compared using informative and non informative priors and the estimated Bayes factor is compared to the target one. In this case, the models are nested and even though, both Scheme A and Scheme B

give good approximations to the posterior model probabilities, Scheme C approximates the Bayes factor better. Hence, it may be worthwhile to implement block updates inside the RJMCMC algorithm in an autoregressive time series variable selection problem.

Bayes Factor	Target	Scheme A	Scheme B	Scheme C
First data set				
$B_{1,2}$	1.1208	1.3066	0.6972	1.1080
$B_{1,2}$	1.0986	2.3574	1.5443	1.3468
Second data set				
$B_{1,2}$	0.7270	1.0693	0.2490	0.7259
$B_{1,2}$	1.0626	3.8048	2.5362	1.5467

Table 4.7: Bayes Factor values for the two most probable models when an informative and an non-informative prior is used respectively.

4.4 Discussion

Chapter 4 considered variable selection problems in linear regression and autoregression model problems. It is suggested that it may be worthwhile to use the full-conditional posterior distribution of the parameters as the proposal distribution to generate the new parameter vector inside the RJMCMC algorithm. Hence, Gibbs updates are proposed inside the RJMCMC algorithm. Then, the full-conditional posterior distribution conditions on all, some or none of the common variables to propose the new parameter vector.

If the models are nested then according to Scheme A only the newly introduced parameter vector is simulated from the full-conditional posterior distribution conditioning on the current parameter values. Brooks and Ehlers (2008)

used the second order method of Brooks et al. (2003) for an autoregressive model choice and they show that the best proposal distribution inside the RJMCMC algorithm is the full-conditional posterior distribution of the parameters. Recall that the method of Brooks et al. (2003) assumes a Normal proposal distribution where parameter values are chosen by maximizing the acceptance probability.

Even though Scheme A is the most popular strategy when the RJMCMC algorithm is applied, it is not always the best one. It turns out that for these specific examples that were examined, Scheme C is the best strategy. According to Scheme C, the entire proposed parameter vector is generated from the posterior distribution of the parameters without retaining any of the current parameter values in the proposed model. Hence, the best proposal distribution is the posterior distribution of the parameters within that model. In this case, the acceptance probability is shown to be proportional to the posterior model probability odds when jumps between different models happen with equal probability. In this case, it is also proportional to the Bayes factor of the proposed model to the current one. For this reason, it is independent of the parameter values and the dimension of the parameter vector. Consequently the mixing of the algorithm is better.

For the autoregressive model choice problem, when the models are nested, Scheme C and Scheme B approximate well the posterior model probabilities while Scheme C gives slightly better approximations to the Bayes factor compared to Scheme B. If the models are not nested, see the linear regression variable selection case, then Scheme C appears to be the best strategy because it approximates better the posterior model probabilities and the Bayes factors.

Here, we have considered cases where the posterior distribution of the parameters is tractable. In most examples, the posterior distribution of the parameters is not known in closed form and for this reason, it has to be estimated. Routine MCMC methods could be used to approximate the intractable posterior distribution but this would be too computationally expensive and time consuming especially when the approximation is used inside the proposal mechanism of the RJMCMC algorithm. Therefore, Chapter 5 aims to overcome this problem and considers cases where the posterior distribution of the parameters is intractable.

Chapter 5

Automatic RJMCMC for variable selection - intractable posterior distribution

In Chapter 4, it was concluded in a Reversible jump Markov chain Monte Carlo (RJMCMC) variable selection problem, that the best strategy when updating within a model is to update the whole proposed parameter vector from the posterior distribution of the parameters within that model if this is possible. Unfortunately, the posterior distribution of the parameters is not always tractable. Our approach here is to find a tractable approximation to the within model posterior distribution, and to use this tractable approximation as a proposal distribution. Thus, analytical approximations such as the Laplace approximation are considered to estimate the posterior distribution of the parameters within a model.

The choice of the proposal distribution to change the model dimension is very crucial. If only local jumps up or down of one or more dimensions are proposed

then the RJMCMC algorithm potentially needs to run for a long time in order to visit all possible models, especially if the sampling space is large. Typically for variable selection problems and for model selection problems, more generally, most of the posterior mass is distributed over a small subset of the models under consideration. Therefore, in such situations, a naive RJMCMC sampler which proposes random jumps to neighbouring models will be frequently rejected and consequently the Markov chain will not have good mixing properties. In practice, it is difficult to guide the Markov chain to propose models with high posterior probability.

Here, the model space is discrete and typically some of the models have high probability mass while most of them have low. Unfortunately, there is no natural way to determine which of them are more probable a priori. Moreover, the Markov property does not allow us to look back at previously visited models and learn from them. Then again, it is feasible to approximate the posterior model probabilities using the Bayesian information criterion (BIC) (Schwarz (1978)) or the Laplace approximation (Tierney and Kadane (1986), Tierney et al. (1989), Lewis and Raftery (1997)) for a relatively small number of parameters. Then, these estimated posterior model probabilities can be used as an independent proposal distribution inside the proposal mechanism of the RJMCMC algorithm to make changes to model dimension.

5.1 Proposal mechanism of RJMCMC

5.1.1 Strategy A: RJMCMC and the Laplace approximation

The Laplace approximation (Tierney and Kadane (1986), Tierney et al. (1989), Lewis and Raftery (1997)) is a Taylor series expansion to the intractable distribution which in this case, is the posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$. Here, the posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$ is approximated by a multivariate Normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix Σ. The parameter vector $\boldsymbol{\mu}$ equals the parameter values that maximize the posterior distribution within each model $M_{\boldsymbol{\gamma}}$, that is $\boldsymbol{\mu} = \arg\max_{\boldsymbol{\theta}_{\boldsymbol{\gamma}}} \pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$. The covariance matrix Σ is related to the Hessian matrix of the posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$ so that

$$\Sigma = -\left(\frac{\partial^2}{\partial \boldsymbol{\theta}_{\boldsymbol{\gamma}}^2}\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})\right)^{-1}.$$

The Laplace approximation is relatively easy to implement. One has only to evaluate the location and the scale of the multivariate Normal distribution and then draw the new parameter vector from it. When the Laplace approximation is used inside the RJMCMC algorithm, the algorithm still performs in an automatic way even though the posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$ is not known in closed form.

The estimated posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$, $\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$ is used to approximate the posterior model probability $\hat{\pi}(M_{\boldsymbol{\gamma}}|\boldsymbol{y})$ in the following way. The posterior distribution of the parameters within model

$M_{\boldsymbol{\gamma}}$ can be written as

$$\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}}) = \frac{L(\boldsymbol{y}|\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}})}{\pi(\boldsymbol{y}|M_{\boldsymbol{\gamma}})} \tag{5.1}$$

where $L(\boldsymbol{y}|\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})$ is the likelihood, $p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}})$ is the prior distribution for the parameters within model $M_{\boldsymbol{\gamma}}$ and $\pi(\boldsymbol{y}|M_{\boldsymbol{\gamma}})$ is the marginal likelihood. From equation (5.1), the marginal likelihood can be defined as

$$\pi(\boldsymbol{y}|M_{\boldsymbol{\gamma}}) = \frac{L(\boldsymbol{y}|\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}})}{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})}. \tag{5.2}$$

In fact, this is exactly the identity used in Chib (1995), to estimate $\pi(\boldsymbol{y}|M_{\boldsymbol{\gamma}})$. Our approximation results from replacing $\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$ in (5.2) with $\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$ and further by evaluating the right hand side of (5.2) at $\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}$ that is the maximum of $\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$. The marginal likelihood approximation is then

$$\pi(\boldsymbol{y}|M_{\boldsymbol{\gamma}}) \approx \hat{\pi}(\boldsymbol{y}|M_{\boldsymbol{\gamma}}) = \frac{\ell(\boldsymbol{y}|\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})p(\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}})}{\hat{\pi}(\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})}.$$

The posterior model probabilities are now approximated by

$$\pi(M_{\boldsymbol{\gamma}}|\boldsymbol{y}) \approx \hat{\pi}(M_{\boldsymbol{\gamma}}|\boldsymbol{y}) = \frac{\hat{\pi}(\boldsymbol{y}|M_{\boldsymbol{\gamma}})p(M_{\boldsymbol{\gamma}})}{\sum_i \hat{\pi}(\boldsymbol{y}|M_i)p(M_i)}$$

where $p(M_{\boldsymbol{\gamma}})$ is the prior probability of model $M_{\boldsymbol{\gamma}}$. The estimated posterior model probabilities are then used inside the proposal mechanism of the RJMCMC algorithm to propose changes to dimension of the model. Notice that in this case, the algorithm still performs in an automatic way since no tuning is necessary.

5.1.2 Strategy B: RJMCMC and the BIC approximation

The posterior model probability, $\hat{\pi}(M_{\boldsymbol{\gamma}}|\boldsymbol{y})$, can also be approximated using the Bayesian information criterion (BIC)(Schwarz (1978)). First, estimate the BIC for each model.

$$\text{BIC}_{M_{\boldsymbol{\gamma}}} = \ell(\boldsymbol{y}|\widetilde{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) - \frac{1}{2} d \log n$$

where $\ell(\boldsymbol{y}|\widetilde{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})$ is the maximized log likelihood at $\widetilde{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}$, the parameter vector $\widetilde{\boldsymbol{\theta}}_{\boldsymbol{\gamma}} = \max_{\boldsymbol{\theta}_{\boldsymbol{\gamma}}} \ell(\boldsymbol{y}|\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})$, d is the dimension of model $M_{\boldsymbol{\gamma}}$ and n is the sample size. Then, the posterior model probabilities are approximated as

$$\pi(M_{\boldsymbol{\gamma}}|\boldsymbol{y}) \approx \hat{\pi}(M_{\boldsymbol{\gamma}}|\boldsymbol{y}) = \frac{\exp\{BIC_{M_{\boldsymbol{\gamma}}}\} P(M_{\boldsymbol{\gamma}})}{\sum \exp\{BIC_{M_k}\} P(M_k)}.$$

The approximated posterior model probabilities, $\hat{\pi}(M_{\boldsymbol{\gamma}}|\boldsymbol{y})$, are now used as an independent proposal distribution in the model updating step. As before, the posterior distribution of the parameters is assumed to be approximated by a Normal distribution, $N(\widetilde{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}, \Sigma)$, with mean equal to the parameter values that maximize the log likelihood and covariance matrix $\Sigma = \sigma^2 \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The proposal parameter value, σ^2, has to be tuned and it should be relatively large reflecting the uncertainty. In this case, the algorithm does not perform in a completely automatic way since the variance of the proposal distribution of the parameters has to be tuned. Moreover, one should also take into account that if an independent proposal distribution is used then the algorithm could perform very poorly if the parameters are highly correlated since the parameters are simulated independently of each other in the proposal move; or the posterior

variance of the parameters are substantially different since the proposal variance is assumed to be the same over all parameters. One could also use a covariance matrix estimated via a pilot run and use the posterior variance and covariance of the parameters in the proposal distribution.

5.1.3 Acceptance probability

In both Strategy A and Strategy B, the main approach is to propose moves between the joint model and the parameter space as follows. Suppose that the Markov chain is currently visiting model $M_{\boldsymbol{\gamma}}$ with parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$. First, a proposed model $M_{\boldsymbol{\gamma}'}$ is selected with probability $\hat{\pi}(M_{\boldsymbol{\gamma}'}|\boldsymbol{y})$, and then the parameter vector, $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$ within this model is simulated from the estimated posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}'}$, $\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y}, M_{\boldsymbol{\gamma}'})$. The overall proposed probability is therefore,

$$q(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \hat{\pi}(M_{\boldsymbol{\gamma}'}|\boldsymbol{y})\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y}, M_{\boldsymbol{\gamma}'}).$$

This proposed distribution can be considered as an approximation to the joint model and parameter distribution

$$\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, M_{\boldsymbol{\gamma}'}|\boldsymbol{y}) \approx \hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, M_{\boldsymbol{\gamma}'}|\boldsymbol{y}) = \hat{\pi}(M_{\boldsymbol{\gamma}'}|\boldsymbol{y})\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y}, M_{\boldsymbol{\gamma}'}).$$

Hence, the proposed move is accepted with probability

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, M_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{q(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, \boldsymbol{\theta}_{\boldsymbol{\gamma}})}{q(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'})}\right)$$

or equivalently,

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}},\boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, M_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{\hat{\pi}(M_{\boldsymbol{\gamma}}|\boldsymbol{y})}{\hat{\pi}(M_{\boldsymbol{\gamma}'}|\boldsymbol{y})} \frac{\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})}{\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y}, M_{\boldsymbol{\gamma}'})}\right).$$

Notice that the acceptance probability reduces to

$$A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}},\boldsymbol{\theta}_{\boldsymbol{\gamma}'}) = \min\left(1, \frac{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, M_{\boldsymbol{\gamma}'}|\boldsymbol{y})}{\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}|\boldsymbol{y})} \frac{\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}|\boldsymbol{y})}{\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}, M_{\boldsymbol{\gamma}'}|\boldsymbol{y})}\right). \tag{5.3}$$

Hence, the above acceptance probability will be close to one when the proposal distribution to change the model dimension, $\hat{\pi}(M_{\boldsymbol{\gamma}'}|\boldsymbol{y})$, is a good approximation to the actual posterior model probability, $\pi(M_{\boldsymbol{\gamma}}|\boldsymbol{y})$, and the proposal distribution of the parameters, $\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$, is also a good approximation to the posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$, $\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$.

The method of Brooks et al. (2003) aims to automate the RJMCMC algorithm by choosing automatically the parameters from the proposal distribution. They consider a Taylor series expansion of the acceptance probability using the first, second or higher order expansions. Then, they choose the parameters of the proposal distribution that maximize the acceptance probability. In our method, one has to find a good approximation to the posterior distribution and the posterior model probabilities in order to achieve the maximum acceptance probability. If a poor approximation to the posterior model probability is made, then the RJMCMC algorithm should correct these approximations and give better estimates to the posterior model probabilities. Thus, it seems to be a good strategy to draw the proposed parameter vector from the approximated posterior distribution of the parameters as well as to choose the proposed model according to

the approximated posterior model probabilities.

The method performs in two steps. First, the posterior distributions within each model and the posterior model probabilities are estimated in an off-line step using the Laplace or the BIC approximation. Then, in the on-line step the RJMCMC algorithm uses these approximations in its proposal mechanisms to generate the proposed parameter vector and propose changes to the model dimension. There is a potential danger in using $\hat{\pi}(M_{\boldsymbol{\gamma}}|\boldsymbol{y})$ as the proposal distribution to jump between models. If this approximation is poor and incorrectly places close to zero probability on any model $M_{\boldsymbol{\gamma}}$, then the sampler will very rarely propose jumps to these models and this may adversely effect the mixing of the chain. Therefore, in the model jumping step, it is considered a mixture of an independent proposal, $\hat{\pi}(M_{\boldsymbol{\gamma}}|\boldsymbol{y})$ and a local proposal to models neighbouring the current model for example, only propose to move to models that differ by a single parameter. The proposed RJMCMC algorithm is described in Algorithm 6.

5.2 Examples

5.2.1 Logistic regression model choice

Here, a logistic regression model is used to test the performance of the algorithms because the posterior model probabilities and the posterior distribution of the parameters within a model are of a non-standard form. Therefore, they have to be approximated before they are used as the proposal distribution inside the RJMCMC algorithm. The performance of the algorithm depends on how well

Algorithm 6 Proposal mechanism of RJMCMC

STEP 1: Off-line

For each model $M_{\boldsymbol{\gamma}}$ estimate:

(i) the posterior distribution of the parameters $\hat{\pi}(\boldsymbol{\theta}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$

(i) the posterior model probabilities $\hat{\pi}(M_{\boldsymbol{\gamma}}|\boldsymbol{y})$

using Strategy A, see section 5.1.1 or Strategy B, see section 5.1.2.

STEP 2: On-line

At each iteration:

1. Within model updates:

   Within model $M_{\boldsymbol{\gamma}}$, $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ is proposed from $\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$.

2. Jump move:

   (i) Propose to move from model $M_{\boldsymbol{\gamma}}$ to model $M_{\boldsymbol{\gamma}'}$. A mixture proposal distribution is used where the proposed model is chosen either according to an independent proposal distribution, $\hat{\pi}(M_{\boldsymbol{\gamma}'}|\boldsymbol{y})$ or to a local proposal distribution where jumps up or down of one or more dimensions are implemented.

   (ii) $\boldsymbol{\theta}_{\boldsymbol{\gamma}'}$ is proposed from $\hat{\pi}(\boldsymbol{\theta}_{\boldsymbol{\gamma}'}|\boldsymbol{y}, M_{\boldsymbol{\gamma}'})$.

   (iii) Accept or reject with probability $A_{\boldsymbol{\gamma},\boldsymbol{\gamma}'}(\boldsymbol{\theta}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}'})$ given in (5.3).

---

the posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$, $\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, M_{\boldsymbol{\gamma}})$, and the posterior model probabilities, $\pi(M_{\boldsymbol{\gamma}}|\boldsymbol{y})$ are approximated. When the above distributions are close enough to the target ones, then the acceptance probability should be close to one.

Here, the power posterior method of Friel and Pettitt (2008) is used to get very accurate estimates of the posterior model probabilities by running a very

long Markov chain. In this way, these estimated posterior model probabilities are used as a benchmark to compare the performance of the RJMCMC algorithm using the approximated posterior model probabilities.

Consider the following logistic regression model

$$p(y = \pm 1|\boldsymbol{x}, \boldsymbol{\theta}) = \delta(y\boldsymbol{\theta}^T\boldsymbol{x}) = \frac{1}{1 + \exp(-y\boldsymbol{\theta}^T\boldsymbol{x})},$$

where y is the binary response, $\boldsymbol{x}$ is a collection of possible explanatory variables and $\boldsymbol{\theta}$ is the parameter vector. Note that if $y = 1$ then

$$p(y = 1|\boldsymbol{x}, \boldsymbol{\theta}) = \delta(\boldsymbol{\theta}^T\boldsymbol{x}) = \frac{\exp(\boldsymbol{\theta}^T\boldsymbol{x})}{1 + \exp(\boldsymbol{\theta}^T\boldsymbol{x})}$$

while if $y = -1$ then

$$p(y = -1|\boldsymbol{x}, \boldsymbol{\theta}) = \delta(-\boldsymbol{\theta}^T\boldsymbol{x}) = \frac{1}{1 + \exp(\boldsymbol{\theta}^T\boldsymbol{x})} = 1 - \delta(\boldsymbol{\theta}^T\boldsymbol{x}).$$

Given a data set $(\mathbf{X}, \boldsymbol{y}) = [(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)]$, assume that one wants to find out which of the explanatory variables $\mathbf{X}$ describes better the binary response $\boldsymbol{y}$. The likelihood function of each model $M_{\boldsymbol{\gamma}}$ can be written as

$$L(\boldsymbol{y}|\mathbf{X}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) = \prod_{i=1}^{n} \frac{1}{1 + \exp(-y_i\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T\boldsymbol{x}_i)}$$

and the log-likelihood function is

$$\ell(\boldsymbol{y}|\mathbf{X}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) = -\sum_{i=1}^{n} \log\left(1 + \exp(-y_i\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T\boldsymbol{x}_i)\right),$$

where $\mathbf{X}_{\boldsymbol{\gamma}}$ is a collection of explanatory variables, $\boldsymbol{x}_i$, that corresponds to model $M_{\boldsymbol{\gamma}}$ and $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ is the parameter vector of model $M_{\boldsymbol{\gamma}}$.

Here, two different prior distributions are used leading to two different analyses. In the first scenario, a Normal prior is assumed for $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ while in the second one, a double exponential (or Laplace) prior is used. In this case, the Laplace approximation may not be so accurate. The aim is to find out whether it is worthwhile to use Strategy A or Strategy B to approximate the posterior model probabilities and if the RJMCMC algorithm improves the already estimated posterior model probabilities.

5.2.1.1 Normal Prior Distribution

Within model $M_{\boldsymbol{\gamma}}$, the prior beliefs about the parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ are expressed through a Normal distribution that is $p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}}) = N(\mathbf{0}, \lambda^{-1}\mathbf{I})$, where $\mathbf{I}$ is the identity matrix and $\lambda \in \mathbb{R}$. Thus, the log-posterior distribution within model $M_{\boldsymbol{\gamma}}$ is

$$\begin{aligned} \log(\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})) &\propto \ell(\boldsymbol{y}|\mathbf{X}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) + \log(p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|M_{\boldsymbol{\gamma}})) \\ &\propto -\sum_{i=1}^{n} \log(1 + \exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) - \frac{\lambda}{2} \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{\theta}_{\boldsymbol{\gamma}}. \quad (5.4) \end{aligned}$$

Unfortunately, the posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$ described in (5.4) is not known in closed form. Therefore, according to Strategy A, the Laplace approximation is used to approximate it. A Normal distribution is assumed for the intractable posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$, $N(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}, -H^{-1})$ where $\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}$ is the parameter vector that maximizes the log-posterior distribution of the parameters within model $M_{\boldsymbol{\gamma}}$ and H is the

Hessian matrix estimated as described below. The first derivatives of the log posterior distribution is

$$\begin{aligned}\frac{\partial \log \pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})}{\partial \boldsymbol{\theta}_{\boldsymbol{\gamma}}} &= -\sum_{i=1}^{n} \frac{\exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)}{1+\exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)}(-y_i \boldsymbol{x}_i) - \lambda \boldsymbol{\theta}_{\boldsymbol{\gamma}} \\ &= \sum_{i=1}^{n}(1-\delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) y_i \boldsymbol{x}_i - \lambda \boldsymbol{\theta}_{\boldsymbol{\gamma}}.\end{aligned}$$

Note that

$$\begin{aligned}\frac{\partial \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T} &= -\frac{\exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)(-y_i \boldsymbol{x}_i)}{[1+\exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)]^2} \\ &= \frac{\exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i) y_i \boldsymbol{x}_i}{[1+\exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)]^2} \\ &= \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)(1-\delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) y_i \boldsymbol{x}_i.\end{aligned}$$

Therefore, the Hessian matrix is

$$\begin{aligned}H &= \frac{\partial^2 \log \pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})}{\partial \boldsymbol{\theta}_{\boldsymbol{\gamma}} \partial \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T} \\ &= \sum_{i=1}^{n} \frac{\partial}{\partial \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T} \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i) - \lambda \mathbf{I} \\ &= -\sum_{i=1}^{n} \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)(1-\delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) y_i \boldsymbol{x}_i (y_i \boldsymbol{x}_i)^T - \lambda \mathbf{I}.\end{aligned}$$

If $y = 1$, then the Hessian matrix takes the following form

$$H = -\sum_{i=1}^{n} \delta(\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)(1-\delta(\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) \boldsymbol{x}_i \boldsymbol{x}_i^T - \lambda \mathbf{I},$$

while if $y = -1$, then the Hessian matrix is

$$\begin{aligned} H &= -\sum_{i=1}^{n} \delta(-\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)(1 - \delta(-\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) \boldsymbol{x}_i \boldsymbol{x}_i^T - \lambda \mathbf{I} \\ &= -\sum_{i=1}^{n} (1 - \delta(\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i))(1 - 1 + \delta(\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) \boldsymbol{x}_i \boldsymbol{x}_i^T - \lambda \mathbf{I}. \end{aligned}$$

We may write the Hessian matrix as

$$H = -\mathbf{X}_{\boldsymbol{\gamma}} \mathbf{A} \mathbf{X}_{\boldsymbol{\gamma}}^T - \lambda \mathbf{I}$$

where the matrix $\mathbf{A}$ is diagonal with $a_{ii} = \delta(\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)(1 - \delta(\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i))$. One way to calculate $\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}$ is by using the Newton-Raphson method. The Newton-Raphson method is an optimization algorithm which finds out a local maximum. The local maximum is achieved by estimating the new parameter vector given the old one as

$$\begin{aligned} \boldsymbol{\theta}_{\boldsymbol{\gamma}_{new}} &= \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}} - H^{-1} \frac{\partial \ell(\boldsymbol{\theta}_{\boldsymbol{\gamma}})}{\partial \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}}} \\ &= \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}} + (\mathbf{X}_{\boldsymbol{\gamma}} \mathbf{A} \mathbf{X}_{\boldsymbol{\gamma}}^T + \lambda \mathbf{I})^{-1} \sum_{i=1}^{n} (1 - \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}}^T \boldsymbol{x}_i)) y_i \boldsymbol{x}_i - \lambda \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}} \\ &= (\mathbf{X}_{\boldsymbol{\gamma}} \mathbf{A} \mathbf{X}_{\boldsymbol{\gamma}}^T + \lambda \mathbf{I})^{-1} \left[(\mathbf{X}_{\boldsymbol{\gamma}} \mathbf{A} \mathbf{X}_{\boldsymbol{\gamma}}^T + \lambda \mathbf{I}) \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}} + \sum_{i=1}^{n} (1 - \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}}^T \boldsymbol{x}_i)) y_i \boldsymbol{x}_i - \lambda \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}} \right] \\ &= (\mathbf{X}_{\boldsymbol{\gamma}} \mathbf{A} \mathbf{X}_{\boldsymbol{\gamma}}^T + \lambda \mathbf{I})^{-1} \left[\mathbf{X}_{\boldsymbol{\gamma}} \mathbf{A} \mathbf{X}_{\boldsymbol{\gamma}}^T \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}} + \lambda \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}} + \sum_{i=1}^{n} (1 - \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}}^T \boldsymbol{x}_i)) y_i \boldsymbol{x}_i - \lambda \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}} \right] \\ &= (\mathbf{X}_{\boldsymbol{\gamma}} \mathbf{A} \mathbf{X}_{\boldsymbol{\gamma}}^T + \lambda \mathbf{I})^{-1} \left[\mathbf{X}_{\boldsymbol{\gamma}} \mathbf{A} \mathbf{X}_{\boldsymbol{\gamma}}^T \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}} + \sum_{i=1}^{n} (1 - \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}}^T \boldsymbol{x}_i)) y_i \boldsymbol{x}_i \right]. \end{aligned}$$

Notice that

$$\begin{aligned}(\mathbf{X}_{\boldsymbol{\gamma}}\mathbf{A})\left[(\mathbf{X}_{\boldsymbol{\gamma}}\mathbf{A})^{-1}\sum_{i=1}^{n}(1-\delta(y_i\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T\boldsymbol{x}_i))y_i\boldsymbol{x}_i\right] &= (\mathbf{X}_{\boldsymbol{\gamma}}\mathbf{A})\left[\mathbf{A}^{-1}\mathbf{X}_{\boldsymbol{\gamma}}^{-1}\sum_{i=1}^{n}(1-\delta(y_i\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T\mathbf{x}_i))y_i\boldsymbol{x}_i\right]\\ &= \mathbf{X}_{\boldsymbol{\gamma}}\mathbf{A}\frac{1-\delta(y_i\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T\boldsymbol{x}_i))y_i}{a_{ii}}.\end{aligned}$$

Hence,

$$\begin{aligned}\boldsymbol{\theta}_{new} &= (\mathbf{X}_{\boldsymbol{\gamma}}\mathbf{A}\mathbf{X}_{\boldsymbol{\gamma}}^T+\lambda\mathbf{I})^{-1}\mathbf{X}_{\boldsymbol{\gamma}}\mathbf{A}\left[\mathbf{X}_{\boldsymbol{\gamma}}^T\boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}}+\frac{1-\delta(y_i\boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}}^T\boldsymbol{x}_i)y_i}{a_{ii}}\right]\\ &= (\mathbf{X}_{\boldsymbol{\gamma}}\mathbf{A}\mathbf{X}_{\boldsymbol{\gamma}}^T+\lambda\mathbf{I})^{-1}\mathbf{X}_{\boldsymbol{\gamma}}\mathbf{A}\mathbf{z},\end{aligned}$$

where

$$z_i=\mathbf{X}_{\boldsymbol{\gamma}}^T\boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}}+\frac{1-\delta(y_i\boldsymbol{\theta}_{\boldsymbol{\gamma}_{old}}^T\boldsymbol{x}_i)y_i}{a_{ii}}.$$

The process is repeated long enough until no changes to the new parameter vector are observed.

5.2.1.2 Double exponential (or Laplace) Prior Distribution

Assume that within model $M_{\boldsymbol{\gamma}}$ the parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}} \sim \text{Laplace}(\mu, b)$, with location parameter $\mu = 0$ and shape parameter b, then the prior distribution of the parameters is

$$p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\mu=0,b)=\left(\frac{1}{2b}\right)^p\exp\left\{-\sum_{i=1}^{p}\frac{|\theta_i|}{b}\right\}$$

where p is the dimension of the parameter vector $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ and the components of $\boldsymbol{\theta}_{\boldsymbol{\gamma}}$ are independent. The log-prior distribution is

$$\log p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\mu = 0, b) = -p\log(2b) - \sum_{i=1}^{p} \frac{|\theta_i|}{b}.$$

The log-posterior distribution is then

$$\begin{aligned} \log(\pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})) &\propto \ell(\boldsymbol{y}|\mathbf{X}_{\boldsymbol{\gamma}}, \boldsymbol{\theta}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) + \log(p(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\mu, b)) \\ &\propto -\sum_{i=1}^{n} \log(1 + \exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) - p\log(2b) - \sum_{i=1}^{p} \frac{|\theta_i|}{b}. \end{aligned} \tag{5.5}$$

The above posterior distribution is not known in closed form and for this reason, the Laplace approximation is used to approximate it so that

$$(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}}) \sim N(\boldsymbol{\theta}_{\boldsymbol{\gamma}}; \hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}, -H^{-1}),$$

where $\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}} = \max_{\boldsymbol{\theta}_{\boldsymbol{\gamma}}} \log \pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})$. Hence, the first derivatives of the log posterior distribution are given by

$$\begin{aligned} \frac{\partial \log \pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})}{\partial \boldsymbol{\theta}_{\boldsymbol{\gamma}}} &= -\sum_{i=1}^{n} \frac{\exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)}{1 + \exp(-y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)}(-y_i \boldsymbol{x}_i) - \frac{1}{b} \\ &= \sum_{i=1}^{n} (1 - \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) y_i \boldsymbol{x}_i - \frac{1}{b}. \end{aligned}$$

The second order derivative of the posterior distribution is

$$\begin{aligned} H &= \frac{\partial^2 \log \pi(\boldsymbol{\theta}_{\boldsymbol{\gamma}}|\boldsymbol{y}, \mathbf{X}_{\boldsymbol{\gamma}}, M_{\boldsymbol{\gamma}})}{\partial \boldsymbol{\theta}_{\boldsymbol{\gamma}} \partial \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T} \\ &= \sum_{i=1}^{n} \frac{\partial}{\partial \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T} \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i) \\ &= -\sum_{i=1}^{n} \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)(1 - \delta(y_i \boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)) y_i \boldsymbol{x}_i (y_i \boldsymbol{x}_i)^T. \end{aligned}$$

Thus, the Hessian matrix is

$$H = -\mathbf{X}_{\boldsymbol{\gamma}} \mathbf{A} \mathbf{X}_{\boldsymbol{\gamma}}^T$$

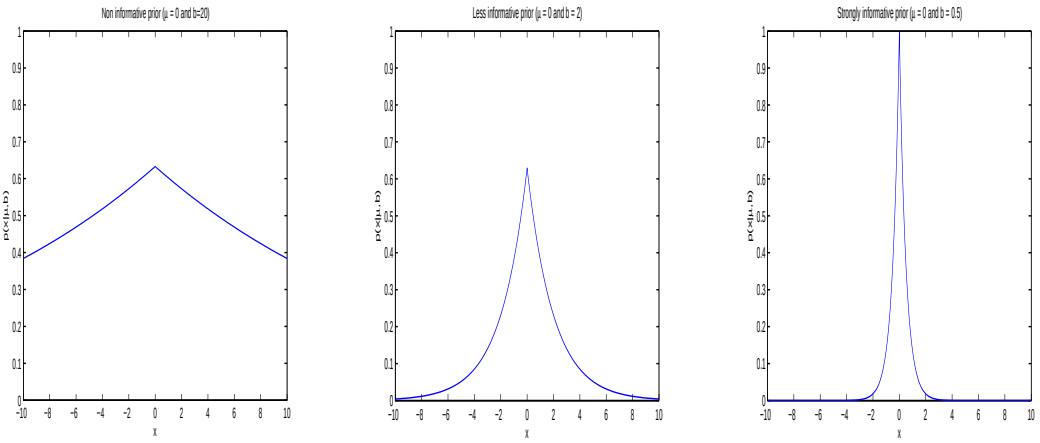
where $a_{ii} = \delta(\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i)(1 - \delta(\boldsymbol{\theta}_{\boldsymbol{\gamma}}^T \boldsymbol{x}_i))$.

Here, the Newton-Raphson method is also applied to estimate the parameter vector that maximizes the posterior distribution. For Strategy B, the simplex method of Nelder and Mead (1965) is used to find the maximum of the log likelihood. The method that was simplest to implement in each case was chosen to find the maximum.

The performance of the algorithm is tested on twenty different simulated data sets that each consist of twenty responses and three explanatory variables. Each of the predictor variables is generated from a multivariate Normal distribution $N(\mathbf{1}, \mathbf{I}_{20})$ where $\mathbf{1} = (1, \ldots, 1)$ and $\mathbf{I}_{20}$ is the identity matrix. The parameter vector is drawn from a multivariate Normal distribution, $N_3(\mathbf{0}, 4\mathbf{I}_3)$, for each data set.

Two different scenarios are examined depending on the choice of the proposal distribution. In the first scenario, a Normal prior distribution is assumed to

describe the prior beliefs about the parameters, $N_3(\mathbf{0}, 1/\lambda\mathbf{I}_3)$, where $\lambda = 0.1$. In the second scenario, three different double exponential priors are used, a non-informative, $p(\cdot|\mu = 0, b = 20)$, a less informative, $p(\cdot|\mu = 0, b = 2)$ and a more informative, $p(\cdot|\mu = 0, b = 0.5)$, see Figure 5.1.



(a) (b) (c)

Figure 5.1: (a) Non-informative double exponential prior with $\mu = 0$ and $b = 20$. (b) Less informative double exponential prior with $\mu = 0$ and $b = 2$. (c) Informative double exponential prior with $\mu = 0$ and $b = 0.5$.

Every time the posterior model probabilities are estimated using the Laplace approximation, the BIC approximation, Strategy A, Strategy B and the vanilla RJMCMC. In the vanilla RJMCMC, the proposal jump steps are drawn from the local proposal distribution and the newly introduced parameters are updated while the rest of them are retained in the proposed state. The newly introduced ones are generated from $N(0, 1)$. For Strategy B, the proposal variance, σ^2, is 100.

Two different moves are implemented inside the RJMCMC algorithm in order to promote mixing. First, the parameters are updated in the current state of the Markov chain using the approximated posterior distribution of the parameters

within model. Then, a jump is used to propose changes to the model dimension. The proposal distribution to change the model dimension uses the independent proposal distribution 90% of the time and the remaining 10% of the time a local proposal to move to a model differing by ± 1 dimension. Each of the RJMCMC algorithms runs for $200,000$ iterations regarding the first $100,000$ as burn in.

The power posterior method runs for a big number of iterations so that the posterior model probabilities can be used a benchmark to compare to the estimated posterior model probabilities from the RJMCMC algorithms. Here, the power posterior algorithm runs using the Metropolis-Hastings algorithm to sample from the power posterior for each of the data sets applying $50,000$ iterations within each temperature with proposal variance 5. The temperature vector is defined as $t_i = (i/55)^3$ for $i = 1, \ldots, 55$. The power posterior method converges to the target values according to Figure 5.2 which is similar for each of the different models.

The weighted and Hellinger distances are calculated in Figure 5.3 and Figure 5.4 respectively. The distance is estimated between the approximated posterior model probabilities and the accurate ones estimated with the power posterior method. The Hellinger distance is not a weighted metric while the weighted distance is weighted over the true posterior model probabilities. The distances take values close to zero if the approximation is good and values close to one if the approximation is poor, see section 4.3.1.2 for more details. The Hellinger distance has also been transformed to take values in $[0, 1]$.

The Laplace approximation seems to give better approximations to the posterior model probabilities compared to the BIC one which is what Boone et al.

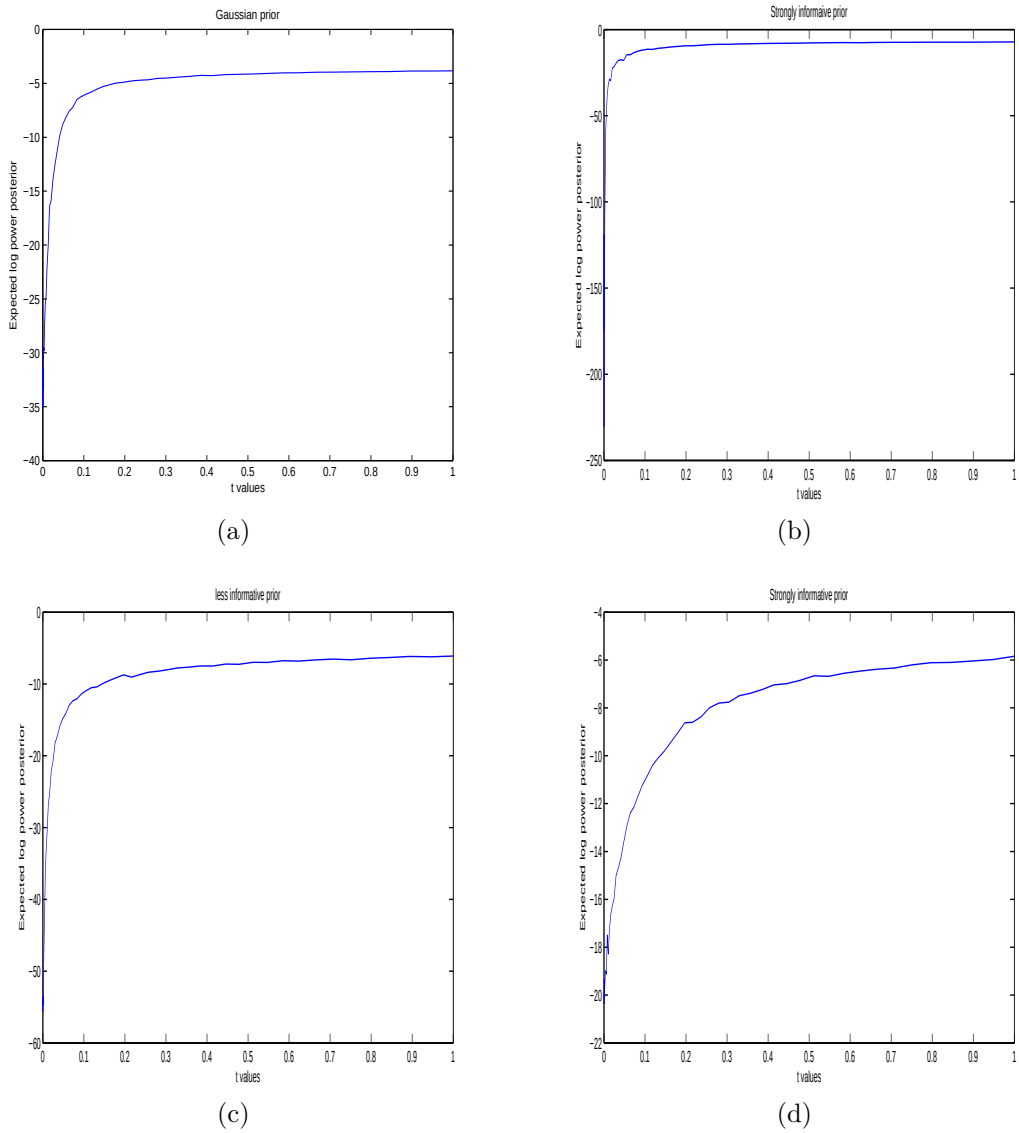


Figure 5.2: Expected values of the logarithmic version of the power posterior for each temperature t using (a) a Normal prior (b) a non-informative double exponential prior with $\mu = 0$ and $b = 20$ (c) a less informative double exponential prior with $\mu = 0$ and $b = 2$ (d) an informative double exponential prior with $\mu = 0$ and $b = 0.5$.

(2005) suggested. The Laplace approximation uses a Normal distribution to approximate the posterior distribution of the parameters within model and even

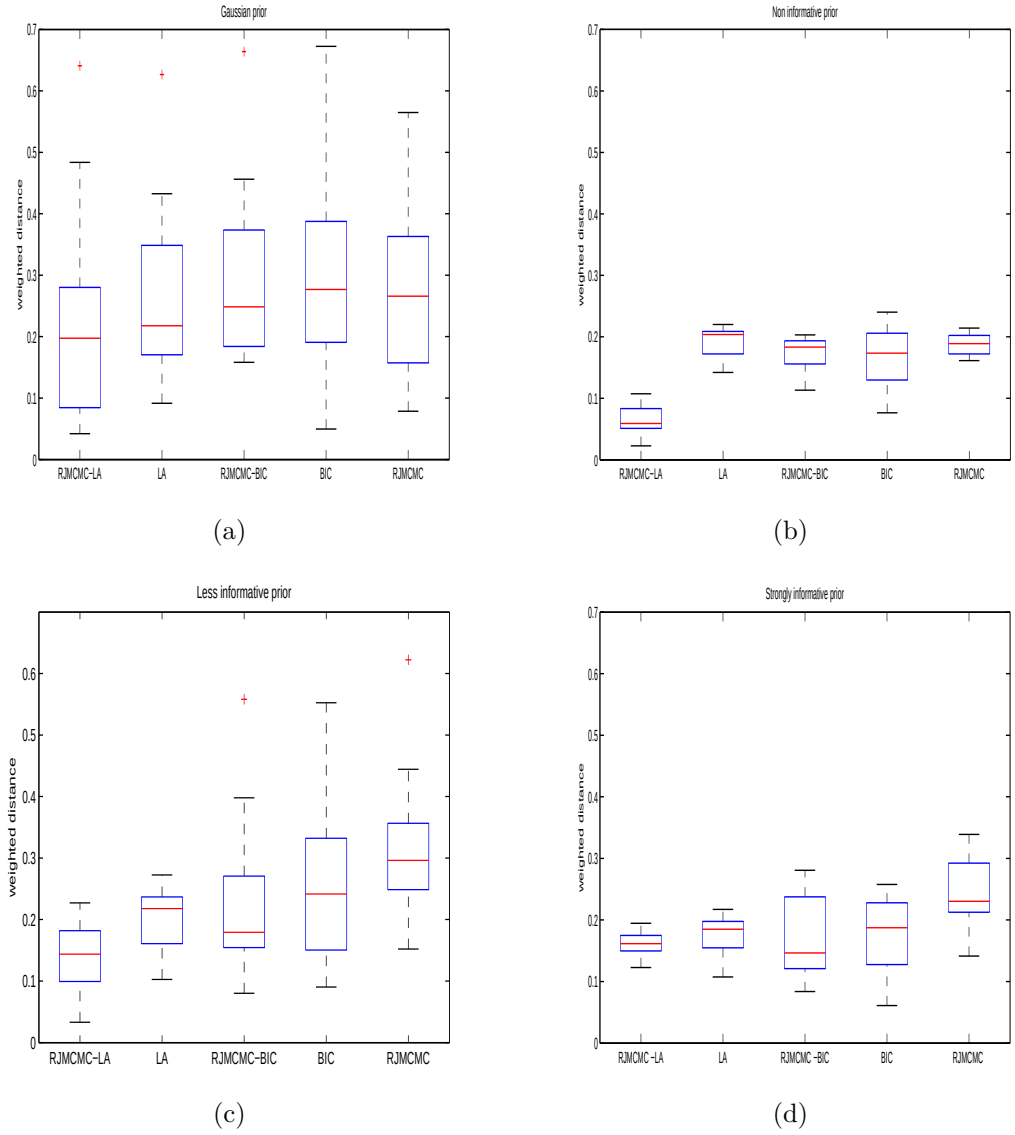


Figure 5.3: Weighted distance for each of the different scenarios using (a) a Normal prior (b) a non-informative double exponential prior with $\mu = 0$ and $b = 20$ (c) a less informative double exponential prior with $\mu = 0$ and $b = 2$ (d) an informative double exponential prior with $\mu = 0$ and $b = 0.5$.

in cases where the target can not be approximated by a Normal distribution, the Laplace approximation still performs quite well in practice, especially if the sample size is large. When a Normal prior is applied, the approximations are

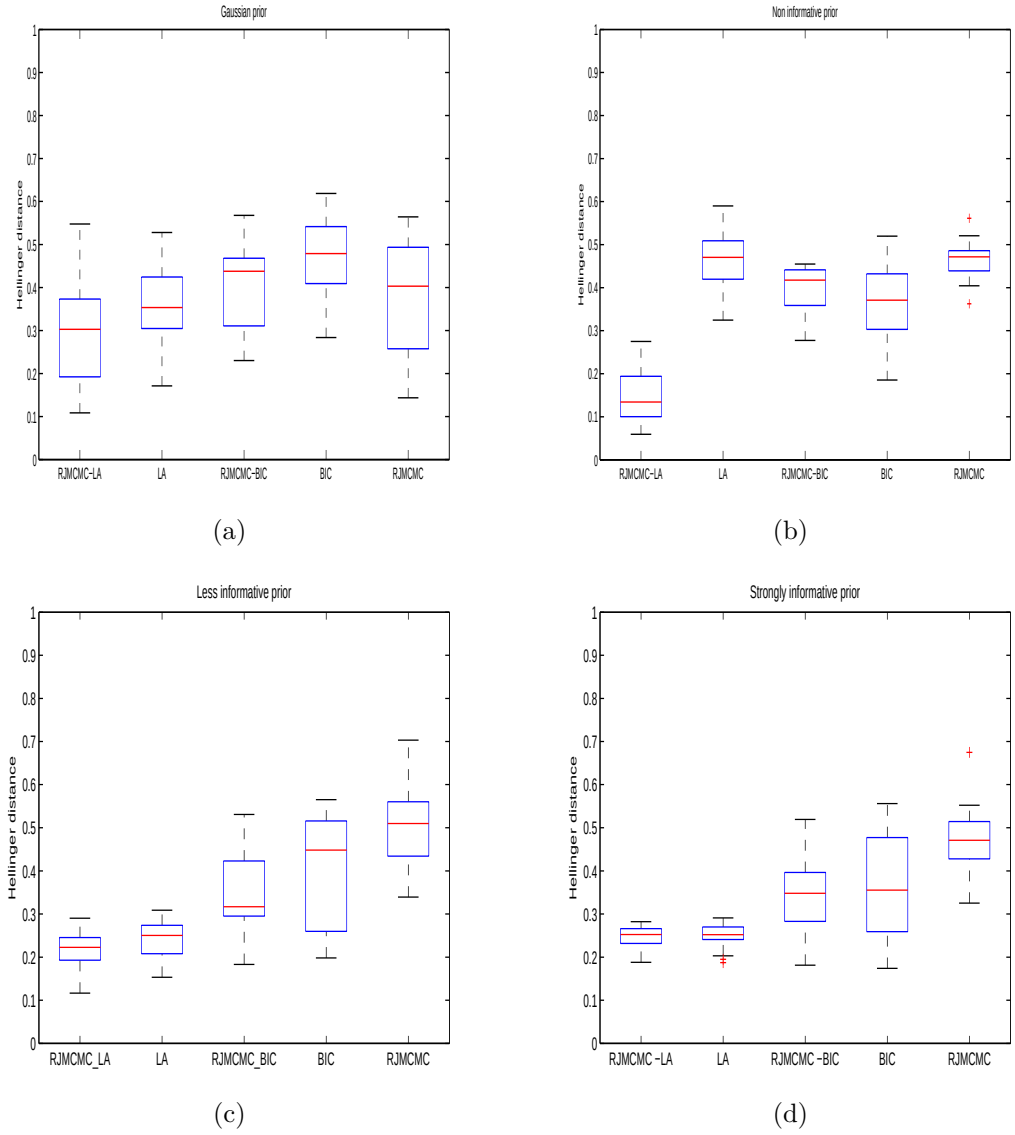


Figure 5.4: Hellinger distance for each of the different scenarios using (a) a Normal prior (b) a non-informative double exponential prior with $\mu = 0$ and $b = 20$ (c) a less informative double exponential prior with $\mu = 0$ and $b = 2$ (d) an informative double exponential prior with $\mu = 0$ and $b = 0.5$.

quite close to the target ones compared to cases where a double exponential prior is used. However, in the case of a non-informative double exponential prior, the Laplace approximations are not as good as the BIC ones. Notice also that the

BIC approximations estimate the posterior model probabilities by maximizing the log-likelihood. The RJMCMC algorithm seems to improve the posterior model probabilities when Strategy A or Strategy B is used. Moreover, the vanilla RJMCMC gives quite poor estimates of the posterior model probabilities following Figure 5.2 and Figure 5.4.

In the case of a Normal prior distribution, Strategy A and Strategy B improve slightly the estimates of the posterior model probabilities. The Laplace approximation also gives better approximations compared to the BIC method. Here, the estimations for each method seem to vary more when compared to the double exponential prior. Then, when a non-informative prior is used the Laplace approximation gives almost as good estimates as the BIC approximation. Then again, the RJMCMC algorithm using Strategy A and Strategy B improves these estimations. Moreover, the vanilla RJMCMC algorithm gives quite poor estimates. When a less informative and an informative prior is used, the Laplace approximation performs better than the BIC one. Once more, the RJMCMC algorithms using Strategy A and Strategy B slightly improves these estimates while the vanilla RJMCMC once more gives poor estimates of the posterior model probabilities.

Hence, it may be reasonable to use Strategy A or Strategy B since the estimated posterior model probabilities are slightly better when compared to the ones of the other methods especially the ones estimated using the vanilla RJMCMC. Overall, Strategy A performs better than Strategy B because of the more accurate estimations provided by the Laplace approximation. Recall that the mixing is better when the estimated posterior model probability and the estimated posterior distribution of the parameters within the model are close to the

target ones because the acceptance probability is maximized. The question is whether it is worthwhile to use the estimated posterior model probabilities inside the RJMCMC algorithms. The RJMCMC algorithm seems to improve any bad approximations but when these approximations are good then it does not seem to contribute anything extra to them. Hence, in this case, the estimated posterior model probabilities do not improve much inside the RJMCMC algorithm.

The acceptance probabilities for each of the RJMCMC algorithms are presented in Table 5.1. It seems that the proposal jumps are accepted more often when the independent proposal is applied. The acceptance probability for the vanilla RJMCMC is less than 1% for each different scenario which implies that the mixing of the chain is very slow.

5.2.2 Real data example

The next example is a real data set presented in Davison (2003). Table 5.2 presents the data of 53 patients with prostate cancer. There are five binary explanatory variables in total. The age in years which takes the value zero if it is less than 60, otherwise it takes the value one. The stage measures the seriousness of the tumour, the grade measures the pathology of the tumour and the xray is zero for less serious cases and it is one for more serious cases. Finally, the level of serum acid phosphatase takes the value zero if it is less than 0.6 otherwise, it takes the value one. The response, nodal involvement, indicates whether the cancer has spread to neighbouring lymph nodes.

The first row of Table 5.2 means that there was a nodal involvement for five out of six patients aged less than 60 that also had high levels of the other

Moves	RJMCMC - LA	RJMCMC - BIC
Normal prior		
exchange	0.4307	0.2326
independent prop	0.2950	0.1992
local prop	0.2727	0.1644
Total	0.3617	0.2144
Non-informative prior		
exchange	0.2087	0.4921
independent prop	0.7092	0.1320
local prop	0.3525	0.190
Total	0.4411	0.2972
Less informative prior		
exchange	0.3147	0.4679
local jump	0.4504	0.110
independent jump	0.7230	0.1259
Total	0.5052	0.2961
Informative prior		
exchange	0.4281	0.4747
local jump	0.6931	0.1020
independent jump	0.4812	0.1200
Total	0.5500	0.2964

Table 5.1: Acceptance probabilities for each of the different strategies using each of the different priors.

explanatory variables. It should be noticed that a case is an individual patient and not a row of the table. The aim is to find out which of the five binary explanatory variables are useful to predict nodal involvement. Hence, since each of the explanatory variables may or may not be included in a model, there are $2^5 - 1 = 31$ possible models to choose from.

A diffuse Normal prior, $N(0, 100^2)$, is used to express the prior beliefs about the parameter values. The RJMCMC algorithms run for $500,000$ iterations using the first $100,000$ as burn in. The algorithms run 20 times using different starting

m	r	age	stage	grade	xray	acid
6	5	0	1	1	1	1
6	1	0	0	0	0	1
4	0	1	1	1	0	0
4	2	1	1	0	0	1
4	0	0	0	0	0	0
3	2	0	1	1	0	1
3	1	1	1	0	0	0
3	0	1	0	0	0	1
3	0	1	0	0	0	0
2	0	1	0	0	1	0
2	1	0	1	0	0	1
2	1	0	0	1	0	0
1	1	1	1	1	1	1
1	1	1	1	0	1	1
1	1	1	0	1	1	1
1	1	1	0	0	1	1
1	0	1	0	1	0	0
1	1	0	1	1	1	0
1	0	0	1	1	0	0
1	1	0	1	0	1	0
1	1	0	0	1	0	1
1	0	0	0	0	1	1
1	0	0	0	0	1	0

Table 5.2: Data on nodal involvement of Miller et al. (1980).

positions. The proposal variance for the RJMCMC using Strategy B is again $\sigma^2 = 100$.

The power posterior method uses 60 different temperatures and the Metropolis-Hastings algorithm runs for $50,000$ iterations within each temperature in order to sample from the power posterior with proposal variance one. In total, $3,000,000$ iterations were produced and such a long MCMC implementation should give very accurate estimates of the posterior model probabilities with which to use as a benchmark to compare the other various methods. The algorithm converges to the true posterior model probabilities according to Figure 5.5 .

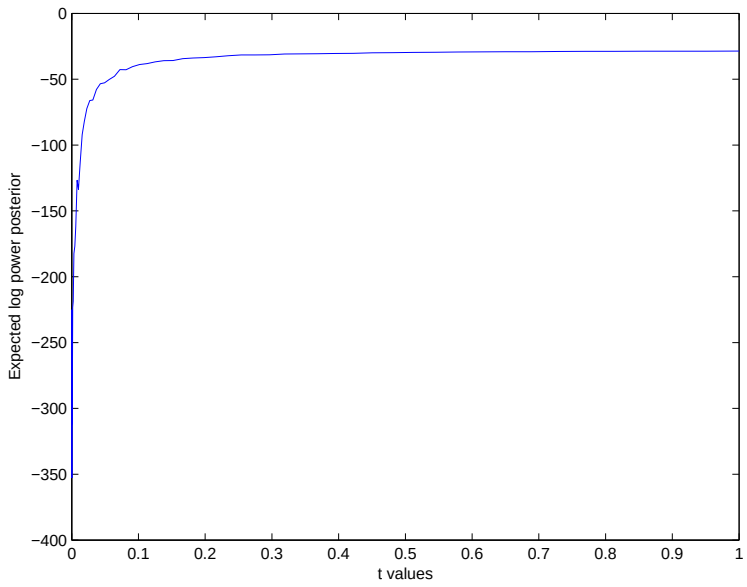


Figure 5.5: Expected log posterior over different temperature values for the real data.

The posterior model probabilities are estimated for each model applying the Laplace and BIC approximation as well as the RJMCMC using Strategy A, Strategy B and also the vanilla RJMCMC. The estimated posterior model probabilities

for each method are compared to the power posterior ones using the weighted and Hellinger distances, see Table 5.4. It seems it is beneficial to use Strategy A or Strategy B as the posterior model probabilities are approximated better comparing to the vanilla method. Then again, these strategies give similar estimates to the posterior model probabilities comparing to the Laplace and BIC approximations. Therefore, it is arguable whether the extra efforts of using the BIC or Laplace approximation inside RJMCMC is actually worthwhile.

Distance	RJMCMC - LA	LA	RJMCMC - BIC	BIC	Vanilla
Hellinger	0.3938	0.3968	0.4258	0.4390	0.6983
Weighted	0.1318	0.1399	0.1131	0.1228	0.1713

Table 5.3: Distance values for each method.

The acceptance probability for each of the RJMCMC algorithms can be seen in Table 5.4. Notice that the acceptance probability is less than 1% for the vanilla RJMCMC meaning that the mixing of the algorithm is poor. When Strategy A or Strategy B is used inside the RJMCMC algorithm, the mixing is improved as shown by the acceptance probability around 0.40 for Strategy A and 0.24 for Strategy B. Notice also that the acceptance probability for the independent proposal distribution is much higher compared to the local proposal distribution.

Accept prob	RJMCMC - LA	RJMCMC - BIC
exchange	0.500	0.3766
local jump	0.1621	0.1086
independent jump	0.3164	0.1646
Total	0.4005	0.2404

Table 5.4: Acceptance probabilities for each of the different methods.

5.3 Discussion

Following Chapter 4, a reasonable strategy to update the parameter values within a model in a variable selection problem considering Normal-Gamma linear models, is to draw the proposed parameter values from the tractable posterior distribution of the parameters within each model. If the posterior distribution of the parameters is tractable then it is straightforward to generate the proposed parameter vector. On the other hand, if it is intractable then it has to be estimated before the proposed parameter vector is drawn from it. The approximation has to be fast and accurate. Unfortunately, usual Markov chain Monte Carlo methods might not be efficiently applied here as they are too time consuming and computationally expensive especially if they are used inside the proposal mechanism of the RJMCMC algorithm. Thus, deterministic approximations should be used instead.

Moreover, when a jump is proposed to a new model, then a uniform proposal distribution is usually assumed for each model in a neighbourhood of the current model, i.e each such model is proposed with the same probability. Even though this strategy is the most popular, it may not be a good choice. Given that the sampling space is discrete, the posterior probability mass may be concentrated

in a few models. Thus, if a model with high probability mass in visited and a jump is proposed to another model with lower probability mass then the proposed jump is more likely to be rejected. In order to improve the proposal mechanism of the RJMCMC algorithm, a more informative proposal distribution is used to change the model dimension. For instance, if there was a prior knowledge on how the posterior model probability mass is spread between the models then most of the time, jumps should be proposed to models with higher posterior model probability and less of the time to models with lower posterior mass. Hence, it is important for the mixing of the algorithm to jump and explore the proposed models with the correct frequency.

Thus, the Laplace approximation can be used to estimate the intractable posterior distribution of the parameters in a deterministic way. It is easy to implement, fast and most of the time very accurate. The Laplace approximation assumes a Normal distribution to the intractable distribution. Hence, when the intractable distribution is similar to a Normal distribution the approximation is quite accurate. Then again, if the intractable distribution is not similar to a Normal one but the sample size is large then the Laplace approximation still performs quite well in practice. Hence, the Laplace approximation can be used to estimate the posterior distribution of the parameters as well as the posterior model probabilities when the sampling space is not large. Thus, these estimates can be used as an independent prior inside the proposal mechanism of RJMCMC to change the model dimension.

So, two different distributions are used to change the model dimension, a local and an independent one. The local distribution changes the model dimension

by applying jumps up and down by one or more dimensions while the independent one uses the estimated posterior model probabilities to choose the proposed model. The local distribution is used to avoid cases where the approximation to the posterior model probabilities is not very good or the estimated posterior model probabilities are close to zero. In this way, each model will be eventually explored.

Moreover, if the approximated posterior model probabilities and the approximated posterior distribution of the parameters are good approximations to the target ones inside the RJMCMC algorithm then the acceptance probability is almost one. Thus, the proposed jumps are more likely to be accepted improving the mixing of the RJMCMC algorithm.

Furthermore, the posterior model probabilities and the posterior distribution of the parameters are estimated using the Laplace and the BIC approximation. When these estimates are used inside the proposal mechanism of the RJMCMC algorithm then the performance of the algorithm improves compared to the vanilla one because the mixing of the algorithm is better. Then again, the estimated posterior model probabilities with the RJMCMC method do not improve much inside the RJMCMC. Hence, it is arguable whether the RJMCMC improves the already estimated posterior model probabilities with the Laplace and BIC approximation. Rue et al. (2009) use various types of approximations, for instance the Laplace approximation, to the posterior distribution and the marginal posterior distribution for a big sampling space. This paper generally agrees that deterministic approximations to the posterior are preferable to MCMC approximations for certain classes of statistical model, and the results of our analysis concur with this view.

The methods proposed here can only be applied when it is possible to individually fit each model to the data. When the sampling space is big then it might be difficult to implement these ideas. For this reason, other techniques have to be developed. For instance, a population of MCMC chains could be used to guide the proposed jump of the RJMCMC. The number of chains that is visiting each model does not have to be constant for each of the possible models. These ideas are discussed further in Chapter 6.

Chapter 6

Conclusions and Future Research

This chapter will state, summarise and discuss the conclusions that can be drawn from the work presented in this thesis. Also, some possible extensions of this work will be considered. This thesis concerns aspects of population Markov chain Monte Carlo (MCMC) and Reversible jump Markov chain Monte Carlo (RJMCMC) methods.

Chapter 1 explained the motivation of the thesis and reviewed some of the different attempts that have been made by different authors in order to solve existing problems on each of these two aspects. A small introduction is also made to the contribution of our research to solve these problems and possible connection between the two different aspects that will involve our future work motivating the thesis.

Chapter 2 explained the disadvantages of the usual MCMC algorithms that use only one chain to explore the sampling space and motivated the use of population MCMC algorithms which use several Markov chains to sample from the

target distribution. Then, a new population MCMC sampler, the simplex sampler was introduced to sample from highly correlated target distributions. The simplex sampler uses a population of Markov chains which interact with each other and is based on the Nelder and Mead (1965) simplex method. The mixing of the simplex sampler is quite good when the distribution of interest has highly correlated parameters. However, when the target distribution is multi-modal the sampler tends to get trapped under a local maxima unable to traverse the whole sampling space.

Chapter 3 introduced a new population MCMC sampler, the tempered simplex sampler, which samples from multi-modal distributions using a tempering ladder. The sampler considers a population of several Markov chains under each temperature and uses the simplex sampler to explore each of the intermediate tempered distributions while it exchanges different populations of Markov chains under different temperatures. In this way, the tempered simplex sampler borrows the good mixing of the simplex sampler in cases of highly correlated target distributions and the good mixing of the tempering methods when a multi-modal target distribution is considered.

Chapter 4 motivated the significance of the choice of a good proposal distribution for the RJMCMC algorithm. It was suggested to use the full-conditional posterior distribution of the parameters as the proposal distribution to generate the proposed parameter vector. There are several proposal distributions depending on how many of the common parameters between the proposed and the current model are conditioned on. These may be conditioned on none, some or all of the common parameters. It was concluded that the best strategy is to use the posterior distribution of the parameters as the proposal distribution so that none

of the common parameter values is conditioned or retained in the proposed state. Chapter 4 considered cases that the posterior distribution of the parameters is tractable.

Furthermore, Chapter 5 considered cases where the posterior distribution of the parameters is intractable. For this reason, deterministic approximations are used to estimate it before the proposed parameter vector is drawn from it in an off-line step. Then, it was shown that these approximations can be used to estimate the posterior model probabilities also in an off-line step. Thus, the main idea was to use these approximations to the posterior model probabilities as an independent proposal distribution inside the proposal mechanism of the RJMCMC algorithm to choose the proposed model. Then, the models tended to be visited with the correct frequency so that models with higher posterior model probability are visited more often. When these approximations were used inside the RJMCMC algorithm, the mixing of the algorithm was better compared to the vanilla one. Then again, the estimated posterior probabilities with the RJMCMC algorithm do not differ much from the ones estimated with the Laplace and the BIC approximation. Hence, it is questionable whether the RJMCMC algorithm improves any good approximations to the posterior model probabilities.

In future, the aim is to create population MCMC algorithms that aim to sample from multi-modal target distributions considering a population of Markov chains under each temperature ladder, for instance, using a population of Markov chains has visited and not just a single one when the tempered transitions method is applied. Then, the new sampler can use the good mixing of population MCMC algorithms as well as the good mixing of the tempered transitions method.

The aim is to connect these two different aspects. Our intention is to construct

a new RJMCMC algorithm based on a population of Markov chains and not just a single one. Then, the population of Markov chains can be used to choose the proposed model. It is known that Markov chains have no memory and for this reason, it is not possible to go back and learn from previous rejections. For this reason, a population of Markov chains can be used to explore each of the possible models and then, these chains can also be used to choose the model to jump to.

Bibliography

Akaike, H. (1974). A new look at statistical model identification. *IEEE Transactions on Automatic Control, 19*, 716–722.

Al-Awadhi, F., M. Hurn, and C. Jennison (2004). Improving the acceptance rate of reversible jump MCMC proposals. *Statistics and Probability letters, 69*, 189–198.

Andrieu, C., N. De Freitas, A. Doucet, and M. I. Jordan (2003). An introduction to MCMC for Machine Learning. *Machine Learning, 50*, 5–43.

Andrieu, C. and E. Moulines (2006). On the ergodicity properties of some adaptive Markov chain Monte Carlo algorithms. *Annals of Applied Probability, 16*, 1462–1505.

Andrieu, C., E. Moulines, and P. Priouret (2005). Stability of stochastic approximation under verifiable conditions. *SIAM Journal on Control and Optimization, 44*, 283–312.

Bartolucci, F., L. Scaccia, and A. Mira (2006). Efficient Bayes factor estimation from the Reversible jump output. *Biometrika, 93*, 41–52.

Bernardo, J. M. and A. F. Smith (1994). *Bayesian Theory*. Wiley.

Boone, E., K. Ye, and E. P. Smith (2005). Assessment of two approximation methods for computing posterior model probabilities. *Computational Statistics and Data Analysis, 48*, 221–234.

Brooks, S. P. (1998). Markov chain Monte Carlo method and its application. *The Statistician, 47*, 69–100.

Brooks, S. P. and R. S. Ehlers (2008). Efficient construction of Reversible jump MCMC proposals for autoregressive time series models. Technical report, Statistical Laboratory, University of Cambridge,.

Brooks, S. P., Y. Fan, and J. S. Rosenthal (2006). Perfect forward simulation via simulated tempering. *Communications in Statistics—Simulation and Computation 35*, 683–713.

Brooks, S. P., N. Friel, and R. King (2003). Classical Model Selection via Simulated Annealing. *Journal of Royal Statistical Society, Series B, 65*, 503–520.

Brooks, S. P., P. Giudici, and G. O. Roberts (2003). Efficient construction of reversible jump Markov chain Monte Carlo proposal distributions (with discussion). *Journal of Royal Statistical Society, Series B, 65*, 3–55.

Cappé, O. and C. Robert (2000). Markov chain Monte Carlo: 10 years and still running! *Journal of the American Statistical Association, 95*, 1282–1286.

Carlin, B. P. and S. Chib (1995). Bayesian model choice via Markov chain Monte Carlo methods. *Journal of Royal Statistical Society, Series B, 57*, 473–484.

Carlin, B. P. and T. A. Louis (2000). *Bayes and Empirical Bayes Methods for Data Analysis*. Chapman & Hall/CRC.

Chen, M. H. and Q. M. Shao (1997a). Estimating ratios of normalising constants for densities with different dimensions. *Statistica Sinica, 7*, 607–30.

Chen, M. H. and Q. M. Shao (1997b). On Monte Carlo methods for estimating ratios of normalizing constants. *Annals of Statistics, 25*, 1563–94.

Chib, S. (1995). Marginal Likelihood from the Gibbs Output. *Journal of the American Statistical Association, 90*, 1313–1321.

Chib, S. and B. P. Carlin (1999). On MCMC sampling in hierarchical longitudinal models. *Statistics and Computing, 9*, 17–26.

Chib, S. and I. Jeliazkov (2001). Marginal likelihood from the Metropolis-Hastings output. *Journal of the American Statistical Association, 96*, 270–281.

Consonni, G. and P. Veronese (1995). A Bayesian method for combining results from several binomial experiments. *Journal of the American Statistical Association, 90*, 935–944.

Cripps, E., R. Kohn, and D. Nott (2006). Bayesian subset selection and model averaging using a centred and dispersed prior for the error variance. *Australian and New Zealand Journal of Statistics, 48*, 237–252.

Davison, A. C. (2003). *Statistical Models*. Cambridge Series.

Dellaportas, P., J. J. Forster, and I. Ntzoufras (2002). On Bayesian model and variable selection using MCMC. *Statistics and Computing, 12*, 27–36.

Dellaportas, P. and I. Papageorgiou (2006). Multivariate mixtures of normals with an unknown number of components. *Statistics and Computing, 16*, 57–68.

Friel, N. and A. N. Pettitt (2008). Marginal likelihood estimation via power posteriors. *Journal of Royal Statistical Society, Series B, 70*(3), 589–607.

Gamerman, D. (1997). *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*. Chapman & Hall/CRC.

Gelfand, A. E. and A. F. M. Smith (1990). Sampling based approaches to calculating marginal densities. *Journal of the American Statistical Association, 85*, 398–409.

Gelman, A., J. B. Carlin, H. S. Stern, and D. B. Rubin (2003). *Bayesian Data Analysis*. Chapman & Hall/CRC.

Gelman, A., W. R. Gilks, and G. Roberts (1996). Efficient Metropolis jumping rules. In J. M. Bernardo, J. O. Berger, A. P. David, and A. F. M. Smith (Eds.), *Bayesian Statistics V*, pp. 599 – 608. Oxford University Press.

Gelman, A. and X. L. Meng (1998). Simulating normalising constants: from importance sampling to bridge sampling to path sampling. *Statistical Science, 13*, 163–185.

Geman, S. and D. Geman (1984). Stochastic Relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6*, 721–741.

Geyer, C. J. (1991). Markov chain Monte Carlo maximum likelihood. In *Computing Science and Statistics: The 23rd symposium on the interface*.

Geyer, C. J. (1992). Practical Markov chain Monte Carlo. *Statistical Science, 7*, 473–483.

Geyer, C. J. and E. A. Thompson (1995). Annealing Markov chain Monte Carlo with applications to ancestral inference. *Journal of the American Statistical Association, 90*, 909–920.

Gilks, W. R., N. G. Best, and K. K. C. Tan (1995). Adaptive rejection Metropolis sampling within Gibbs sampling. *Journal of Royal Statistical Society, Series C 44*, 455–472.

Gilks, W. R., S. Richardson, and D. J. Spiegelhalter (1996). *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC.

Gilks, W. R., G. O. Roberts, and E. I. George (1994). Adaptive direction sampling. *The Statistician, 43*, 179–189.

Gilks, W. R., G. O. Roberts, and S. K. Sahu (1998). Adaptive Markov chain Monte Carlo through regeneration. *Journal of the American Statistical Association, 93*, 1045–1054.

Godsill, S. J. (2001). On the relatonship between Markov chain Monte Carlo methods for model uncertainty methods. *Journal of Computational and Graphical Statistics, 10*, 230–248.

Gramacy, R. B., R. J. Samworth, and R. King (2009). Importance Tempering. To appear in Statistics and Computing.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika, 82*, 711–732.

Green, P. J., N. L. Hjort, and S. Richardson (2003). *Trans-dimensional Markov*

chain Monte Carlo. In Highly Structured Stochastic Systems. Oxford, U.K: Oxford University Press.

Green, P. J. and A. Mira (2001). Delayed rejection in Reversible jump Metropolis-Hastings. *Biometrika, 88*, 1035–1053.

Haario, H., E. Saksman, and J. Tamminen (1999). Adaptive proposal distribution for random walk Metropolis algorithm. *Computational Statistics, 14*, 375–395.

Haario, H., E. Saksman, and J. Tamminen (2001). An adaptive Metropolis algorithm. *Bernoulli, 7*, 223–242.

Han, C. and B. P. Carlin (2001). Markov chain Monte Carlo methods for computing Bayes Factors: a comparative review. *Journal of the American Statistical Association 96*, 1122–1132.

Hastie, D. (2005). *Towards an automatic reversible jump Markov Chain Monte Carlo.* Ph. D. thesis, University of Bristol.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their application. *Biometrika 57*, 97–109.

Jasra, A., D. Stephens, and C. Holmes (2007a). On population-based simulation for static inference. *Statistics and Computing 17*, 263–279.

Jasra, A., D. Stephens, and C. C. Holmes (2007b). Population-based reversible jump MCMC. *Biometrika 94*, 787–807.

Jeffreys, H. (1935). Some tests of significance, treated by the theory of probability. *Proceedings of the Cambridge Philosophical Society, 31*, 97–109.

Jeffreys, H. (1961). *Theory of Probability* (third ed.). Oxford.

Kass, R. and L. Wasserman (1995). A reference Bayesian test for nested hypotheses and its relationship with the Schwarz criterion. *Journal of the American Statistical Association, 90*, 928–934.

Kass, R. E. and A. E. Raftery (1995). Bayes Factors. *Journal of the American Statistical Association, 90*(430), 773–795.

Lavine, M. and M. J. Schervish (1999). Bayes factors: what they are and what they are not. *American Statistician, 53*, 119–122.

Lewis, S. M. and A. E. Raftery (1997). Estimating Bayes factors via posterior simulation with the Laplace-Metropolis estimator. *Journal of the American Statistical Association, 92*, 648–655.

Liang, F. and W. H. Wong (2001). Real-parameter Evolutionary Monte Carlo with applications to Bayesian mixture models. *Journal of the American Statistical Association, 96*, 653–666.

Liu, J. S., F. Liang, and W. H. Wong (2001). A theory of dynamic weighting in Monte Carlo computation. *Journal of the American Statistical Association, 96*, 561–573.

Madigan, D. and J. C. York (1997). Bayesian methods for estimation of the size of a closed population. *Biometrika, 84*, 19–31.

McLachlan, G. J. and D. Peel (2000). *Finite Mixture Models*. Wiley, New York.

Meng, X. and W. Wong (1996). Simulating ratios of normalising constants via a simple identity: a theoretical exploration. *Statistica Sinica, 6*, 831–860.

Meng, X. L. and S. Schilling (2002). Warp bridge sampling. *Journal of Computational and Graphical Statistics, 11*, 552–86.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). Equations of state Calculations by fast computing machines. *The Journal of Chemical Physics 21*, 1087–1092.

Miller, R., B. Efron, B. Brown, and L. E. Moses (1980). *Prediction analysis for binary data*. Biostatistics Casebook.

Minka, T. (2007). A comparison of numerical optimizers for logistic regression. Technical report, Microsoft Research Cambridge.

Neal, R. (1996). Sampling from multimodal distributions using tempered transitions. *Statistics and Computing, 6*, 353–366.

Nelder, J. A. and R. Mead (1965). A simplex method for function minimization. *The Computer Journal, 7*, 308–313.

Peskun, P. H. (1973). Optimum Monte-Carlo sampling using Markov chains. *Biometrika, 60*, 607–612.

Raftery, A. E. (1996). Approximate Bayes factors and accounting for model uncertainty in generalised linear models. *Biometrika, 83*, 251–266.

Raftery, A. E., D. Madigan, and J. A. Hoeting (1997). Bayesian model averaging for linear regression models. *Journal of the American Statistical Association, 92*, 179–191.

Richardson, S. and P. J. Green (1997). On Bayesian analysis of mixture models via an unknown number of components (with discussion). *Journal of Royal Statistical Society, Series B, 59*, 731–792.

Robert, C. P. and G. Casella (2004). *Monte Carlo statistical methods*. Springer-Verlag.

Robert, C. P. and K. L. Mengersen (2003). Iid sampling with self-avoiding particle filters: The Pinball sampler. In *Valencia 7*.

Robert, C. P., T. Rydén, and D. M. Titterington (2000). Bayesian inference in hidden Markov models through the reversible jump Markov chain Monte Carlo method. *Journal of Royal Statistical Society, Series B 62*, 57–75.

Roberts, G. O. and S. R. Jeffreys (2007). Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. *Journal of Applied Probability 44*(2), 458 –475.

Roberts, G. O. and J. S. Rosenthal (1999). Convergence of slice sampler Markov chains. *Journal of Royal Statistical Society, Series B, 61*, 643–600.

Roberts, G. O. and J. S. Rosenthal (2001). Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science, 16*, 351–367.

Roberts, G. O. and S. K. Sahu (1997). Updating schemes, correlation Structure, blocking and parametrization for the Gibbs sampler. *Journal of Royal Statistical Society, Series B, 59*, 291–317.

Rue, H., S. Martino, and N. Chopin (2009). Approximate Bayesian inference

using integrated nested Laplace approximations. *Journal of Royal Statistical Society, Series B, 71*, 1–35.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics, 6*, 461–464.

Sisson, S. A. (2005). Transdimensional Markov chains: A decade of progress and future perspectives. *Journal of the American Statistical Association, 100*, 1077–1089.

Sokal, A. (1997). Monte Carlo Methods in Statistical Mechanics: Foundations and New Algorithms. *Functional integration, 361*, 131–192.

Stephens, M. (2000). Bayesian analysis of mixture models with an unknown number of components - an alternative to Reversible jump methods. *The Annals of Statistics, 28*(40-74).

Stone, M. (1976). Corrigenda: Cross validatory choice and assessment of statistical predictions. *Journal of Royal Statistical Society, Series B, 38*, 102–102.

Strens, M. J. A., M. Bernhardt, and N. Everett (2002). Markov chain Monte Carlo sampling using direct search optimization. Proceedings of the Nineteenth International Conference on Machine Learning, Sydney, Australia.

Tierney, L. (1994). Markov chains for exploring posterior distributions. *Annals of Statistics, 22*, 1701–1762.

Tierney, L. and J. Kadane (1986). Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association, 81*, 82–86.

Tierney, L., R. Kass, and J. Kadane (1989). Fully exponential Laplace approximations to expectations and variances of non-positive functions. *Journal of the American Statistical Association, 84*, 710–716.

Tierney, L. and A. Mira (1999). Some adaptive Monte Carlo methdos for Bayesian inference. *Statistics in Medicine, 18*, 2507–2515.

Titterington, D. M., A. F. M. Smith, and U. E. Makov (1985). *Statistical Analysis of Finite Mixture Distributions*. Wiley, Chichester.

Warns, G. (2001). The Normal kernel coupler: An adaptive Markov chain Monte Carlo method for efficiently sampling from multi-modal distributions. Technical report, University of Washington.