Gibbs sampling and Reversible jump Markov Chain Monte Carlo with applications

Despina Skilogianni

Supervisor:
Assistant Prof. Spyridon J. Hatjispyros
Co-Supervisors:
Associate Prof. John Tsimikas
Assistant Prof. Stelios Zimeras

Department of Mathematics
(Track in Statistics and Actuarial - Financial Mathematics)
University of the Aegean
Greece
June 2017

Abstract

Markov Chain Monte Carlo (MCMC) methods are widely used in Bayesian statistics, computational physics, finance, computational biology and computational linguistics to solve numerical problems. These methods involve performing experiments on a computer as a means of random sampling.

The evolution of MCMC techniques is rapid and there is a growing interest in studying and comprehending these techniques, as the development of new methods, concepts and algorithms is constant and overwhelming.

We focus our interest on some well known MCMC samplers: the Metropolis-Hastings, the Gibbs sampler and the reversible jump MCMC. We attempt to make estimates as accurate as possible through the implementation of these samplers.

Acknowledgements

The current master's thesis was developed under the supervision of Mr. S.J. Hatjispyros who I need to thank for all his help and guidance throughout my postgraduate studies.

Furthermore, I am grateful for my family and friends since their faith and support are enough for me to accomplish any goal.

To my family, Christos, Vasiliki and Vana.

Contents

Introduction

The purpose of this master's thesis is to present Bayesian theory as the framework for implementing Monte Chain Monte Carlo methods. We approach Markov Chain Monte Carlo (MCMC) methods theoretically and perform simulations of various models by using MCMC samplers. All applications are implemented in R language and with the use of package coda.

In the first chapter, we present the basic Bayesian structure, we define exchangeability and sequential analysis. Also, we refer to Bayesian prediction and the pros of Bayesian inference.

In the second chapter, we present necessary elements of Markov chain theory and Monte Carlo integration. We make spacial reference to reversible Markov chains and the detailed balance condition. This section is completed with the theory behind MCMC methods.

In the third chapter, we analyse the Metropolis-Hastings (MH) algorithm for both joint and single-site updating and we present the derivation of the acceptance probability. Furthermore, we investigate the challenges that arise when implementing MH and provide some techniques for monitoring convergence. We simulate from a Normal distribution by using three different proposal steps.

In the fourth chapter, we present a special case of the MH algorithm, the Gibbs sampler, and we prove that all proposed moves are accepted with probability 1. We introduce the fully conditional distributions, which constitute the most important step for the implementation of this method. We present and simulate the Normal hierarchical model.

In the fifth and last chapter, we introduce the reversible jump MCMC (rjMCMC) sampler. We derive the acceptance probability and we prove that it is actually the general case of the MH algorithm. We present the necessary conditions for the implementation of the sampler. The sampler is used in a regression model selecting problem.

Chapter 1

Bayesian Inference

1.1 Basic Bayesian structure

In frequentist statistics, we draw conclusions about the unknown parameter under study by performing multiple repetitions of an experiment, assuming that the realizations we produce are statistically independent. The unknown parameter is considered constant and the inference is only dependent on the sample, which practically means that we rely on the likelihood of the data for our inferences. Methodologies like confidence intervals and hypothesis testing are based on frequentist statistics.

In Bayesian inference, we treat all unknown parameters as random variables and in order to update the probability of a hypothesis as more information is obtained, we use Bayes' theorem. All unknown quantities, namely the unknown parameter and the data before observation, have a probability distribution. For the data, the distribution, given θ, comes from a model that arises from past experience in handling similar data as well as subjective judgement. The distribution of θ arises as a quantification of our knowledge and belief. If though, this knowledge and belief are weak, we may fall back on a common objective distribution.

- $x = (x_1, \ldots, x_n)$: the observed data, where $x_j \in \mathbb{R}^m,\ m \geq 1$.
- $\theta = (\theta_1, \ldots, \theta_d) \in \Theta,\ \Theta \subseteq \mathbb{R}^d,\ d \geq 1$: the unknown model parameters
- $\pi(\theta)$: the prior density of θ. The sample $x = (x_1, \ldots, x_n)$ is used for updating $\pi(\theta)$
- $\pi(x_i|\theta)$: the likelihood function of the data
- $\pi(\theta|x)$: the posterior density of θ. Our inferences derive from the posterior distribution. The transition from $\pi(\theta)$ to $\pi(\theta|x)$ is what we have learnt from the data.

Some descriptive measures associated with the posterior distribution that will prove very useful further on, are the posterior mean and variance. For a real valued parameter θ:

$$E(\theta|x) = \int_{-\infty}^{+\infty} \theta\, \pi(\theta|x)\, d\theta$$

$$Var(\theta|x) = E\{(\theta - E(\theta|x))^2|x\} = \int_{-\infty}^{+\infty} (\theta - E(\theta|x))^2 \pi(\theta|x)\, d\theta$$

By using Baye's rule, we combine prior knowledge and the data for our estimations. The results are presented through the posterior distribution.

$$\pi(\theta|x) = \frac{\pi(\theta, x)}{\pi(x)} = \frac{\pi(x|\theta)\,\pi(\theta)}{\pi(x)} = \frac{\pi(x|\theta)\,\pi(\theta)}{\int\limits_{\Theta} \pi(x|\theta)\,\pi(\theta)\,d\theta}$$
$$\overset{\theta}{\propto} \pi(\theta, x) = \pi(x|\theta)\,\pi(\theta)$$

where $\pi(x)$ is the mixture of the likelihood and the model for the prior and is known as the prior predictive distribution. The prior predictive is the distribution of the data $x = (x_1, \ldots, x_n)$ and as its name suggests, it predicts the distribution of the data according to the probability model that we applied as a prior for θ. Note that $\pi(x)$ is also known as normalizing constant of the posterior kernel. The numerator is the joint density of θ and x and the denominator is the marginal density of x. When the parameter θ is discrete, the integral in the denominator is replaced by a sum.

Posterior Predictive Distribution

Let us assume that we have observed the data $x = (x_1, \ldots, x_n)$ and also that we know the posterior distribution of $\theta|x$. If we wish to make a prediction about a new observation x_{n+1} (which comes right after x) we need to compute the posterior predictive density $\pi(x_{n+1}|x)$. If $x_1, \ldots, x_n, x_{n+1}$ are conditionally independent given θ such that $x_i|\theta \sim \pi(\cdot|\theta)$, then:

$$\pi(x_{n+1}|x) = \int\limits_{\Theta} \pi(x_{n+1}, \theta|x)\,d\theta = \int\limits_{\Theta} \pi(\theta|x)\,\pi(x_{n+1}|\theta, x)\,d\theta$$
$$= \int\limits_{\Theta} \pi(\theta|x)\,\pi(x_{n+1}|\theta)\,d\theta$$

where, due to $x_1, \ldots, x_n, x_{n+1}$ independence,

$$\pi(x_{n+1}|\theta, x) = \frac{\pi(x_{n+1}, x, \theta)}{\pi(x, \theta)} = \frac{\pi(x_{n+1}, x|\theta)\pi(\theta)}{\pi(x|\theta)\pi(\theta)} = \frac{\pi(x_{n+1}, x, \theta)}{\pi(x, \theta)} = \frac{\pi(x_{n+1}|\theta)\pi(x|\theta)}{\pi(x|\theta)} = \pi(x_{n+1}|\theta).$$

$$E(x_{n+1}|x) = E(E(x_{n+1}|\theta)|x) = \int_X x_{n+1}\pi(x_{n+1}|\theta)dx_{n+1} = E(x_{n+1}|\theta)$$

similarly

$$Var(x_{n+1}|x) = E(Var(x_{n+1}|\theta)|x) + Var(E(x_{n+1}|\theta)|x)$$

Suppose that we have computed the posterior distribution $\pi(\theta|x_1, \ldots, x_m)$ for $(x_1, \ldots, x_m)$ observations and prior on θ, $\pi(\theta)$. Suppose now, that a new set of $n - m$ observations, $(x_{m+1}, \ldots, x_n)$ emerges, then we can use the knowledge we have on $(x_1, \ldots, x_m)$ as a prior to compute the posterior $\pi(\theta|x_1, \ldots, x_m, x_{m+1}, \ldots, x_n)$. Under the assumption of conditional independence:

$$\pi(\theta|x_1, \ldots, x_m, x_{m+1}, \ldots, x_n) \propto \pi(\theta|x_1, \ldots, x_m)\,\pi(x_{m+1}, \ldots, x_n|\theta)$$

where $\pi(\theta|x_1, \ldots, x_m)$ is the updated prior and $\pi(x_{m+1}, \ldots, x_n|\theta)$ is the likelihood of the new set of observations.

1.2 Exchangeability

An exchangeable sequence of random variables is a sequence such that future samples behave like earlier samples. The idea is that past experience is used as a basis for making predictions about the future. It is closely related to the use of independent and identically distributed random variables in statistical models. The concept of exchangeability was introduced by de Finetti. De Finetti's theorem states that exchangeable observations are conditionally independent given some latent variable to which a probability distribution would then be assigned.

An infinite sequence of real random variables $\{X_i\}_{i=1}^{\infty}$ is infinitely exchangeable, if for each finite sequence $\{X_i\}_{i=1}^{n}$ the joint probability distribution $\pi(x_1, \ldots, x_n)$ satisfies the following

$$\pi(x_1, \ldots, x_n) = \pi(x_{p(1)}, \ldots, x_{p(n)})$$

for all permutations $p \in \text{Perm}\{1, 2, \ldots, n\}$.

De Finetti's Theorem. If an infinite sequence of real random variables $\{X_i\}_{i=1}^{\infty}$ is infinitely exchangeable for all n, the joint distribution of $X_1, \ldots, X_n$, can be represented as

$$\pi(x_1, \ldots, x_n) = \int_{\Theta} \left\{ \prod_{i=1}^{n} \pi_s(x_i|\theta) \right\} \pi(\theta)\, d\theta$$

1.3 Advantages of Bayesian Inference

The Bayesian approach provides a fairly clear solution to common problems of statistical inference, new problems of high-dimensional data analysis, as well as complex decision problems of real life. It can handle presence of prior knowledge or partial prior knowledge. In some cases, a subjective prior can be used, though in most other cases one can choose an objective prior. Of course, in all cases one would wish to study the robustness of various aspects of the posterior with respect to modest variation in prior. However, we still use mostly objective priors, because it is difficult to elicit fully subjective priors in most problems in practice. If a fully subjective prior is available we would most certainly use it.

The subjective Bayesian approach is free from violation of some principles that are associated with classical statistics. These negative properties are due to the fact that classical statistics provides either data dependent measures like P-values which are difficult to interpret or confidence coefficients that are obtained by integrating over the whole sample space which may not make sense when a particular data set is under study. The objective Bayesian approach is not completely free from violation of some of these principles and often has frequentist validation, for example, consider the concept of equal probability priors.

Finally, basic Bayesian ideas and measures are easy to interpret and therefore easy to communicate. It might seem however, that in spite of all these advantages, a major growth and spread of the Bayesian approach has occurred only recently. A really important factor has been the arrival of MCMC (Markov chain Monte Carlo) methods in a big way and thereafter the advances in computation of posteriors for high-dimensional spaces and many real-life applications.

Chapter 2

Sampling from the Posterior Distribution

Before we analyse Markov chain Monte Carlo methods, some essential definitions are presented, so that the reader is familiarised with the theoretical background which is needed to comprehend those methods. Discrete time Markov chains with continuous state spaces are considered here, that is the random variables are assumed to be distributed continuously.

2.1 Markov Chain Theory essentials

A Markov chain is a sequence of random variables $\{X_n\}_{n\geq 1}$ where the conditional distribution of X_n given all the previous states $X_{n-1}, X_{n-2}, \ldots, X_0$ is the same as the conditional distribution of X_n given only the previous state X_{n-1}:

$$P(X_{n+1} \in A | X_0, X_1, X_2, \ldots, X_n) = P(x_{n+1} \in A | X_n).$$

Assume that X is a d-dimensional stochastic vector with density f on $\mathbb{R}^d$ and $A \subset \mathbb{R}^d$. Then, if $X_n = x$ is a given current state, the **transition kernel** specifies the conditional probability that X_{n+1} falls in A

$$K(x, A) = P(X_{n+1} \in A | X_n = x)$$

The chain is stationary or time-homogeneous if the transition kernel does not change over time, so for all n:

$$P(X_{n+1} \in A | X_n = x) = P(X_1 \in A | X_0 = x)$$

Stationarity: A probability measure π representing a possible equilibrium for the Markov chain is called a stationary or invariant distribution if

$$P(X_{n+1} \in A | X_n = x) = \int_A K(x, y) dy$$

where $K(x, y)$ is the transition kernel associated with the chain. This means that if we are in a state with distribution π, $X_n \sim \pi(x)$ then also the subsequent states will have distribution π, $X_{n+1} \sim \pi(x)$:

$$P(X_{n+1} \in A) = \int_X P(X_{n+1} \in A | X_n = x)\pi(x)\, dx = \int_X \int_A K(x,y)\, dy\, \pi(x)\, dx$$
$$= \int_A \int_X K(x,y)\, \pi(x)\, dx\, dy = \int_A \pi(y)\, dy$$
$$\Rightarrow X_{n+1} \sim \pi(x).$$

If this is satisfied, the chain will stay within the invariant distribution provided that it enters it at some point.

Ergodicity: An ergodic Markov chain converges to a unique stationary distribution π, for all legal initial probability distributions:

$$P(X_n \in A | X_0) \longrightarrow \int_A \pi(x)\, dx, \;\; n \to \infty.$$

According to Markov chain theory, the following requirements on the random chain need to be satisfied in order for the chain to be ergodic, hence having an invariant or stationary distribution π as its limiting distribution regardless of the starting point.

Irreducibility: A Markov chain is considered irreducible when all its states communicate in a finite number of transitions. The states of an irreducible chain all have the same period. A condition for irreducibility is: every state can be accessed from every other state, so that the complete sample space may be visited by the chain.

Aperiodicity: The period of a state $j \in S$ is defined as

$$d(j) = gcd\{n \in \mathbb{N} : p_{jj}(n) > 0\}$$

- if $d(j) > 1 \Rightarrow j \in X$ is periodic with period $d(j)$
- if $d(j) = 1 \Rightarrow j \in X$ is aperiodic

A Markov chain is aperiodic if all states have period 1. All states of an irreducible Markov chain, have the same period. A sufficient condition for a state to have period 1 is

$$P(X_n = j | X_0 = j) > 0 \quad \text{and} \quad P(X_{n+1} = j | X_0 = j) > 0$$

for some $n \geqslant 0$ and some state $j = 0, 1, ..., N-1$.

A discrete Markov chain with continuous state space, apart from having the properties of irreducibility and aperiodicity, must be Harris recurrent in order to ensure convergence to a unique stationary distribution. Harris recurrence is required to avoid the possibility of starting points from which convergence is not assured.

Recurrency: If the average number of visits to an arbitrary set A is infinite, the chain is said to be recurrent. Otherwise, if the average number of visits to A is finite, the chain is transient. A chain is said to be Harris recurrent if the probability of an infinite number of returns to A is 1, and Harris recurrence ensures that the chain has the same limiting distribution for every starting point.

Assume that τ_{ii} is the moment of first return at the i - state and is defined as

$$\tau_{ii} = \min\{t > 0 : \; X_t = i | X_0 = i\}.$$

State i is recurrent if $P(\tau_{ii} < \infty) = 1$ and positively recurrent if $E(\tau_{ii} < \infty)$. Note that an irreducible Markov chain is positively recurrent if all its states are positively recurrent.

2.2 Reversible Markov Chains

Consider an ergodic Markov chain with state space S that converges to an invariant distribution π. Let $x \in S$ denote the current state of the system, and let $y \in S$ denote the state of the next step. Let $p(x, y)$ be the probability of a transition from x to y and let $p(y, x)$ denote the probability of a transition in the opposite direction. A Markov chain is considered reversible if it satisfies the condition:

$$\begin{aligned} P\{X_n \in A, X_{n+1} \in B\} &= P\{X_n \in B, X_{n+1} \in A\} \\ \Rightarrow \pi(x)\, q(x, y) &= \pi(y)\, q(y, x) \end{aligned}$$

for all $x, y \in X$ and $A = (x, x + dx], B = (y, y + dy]$. This condition is known as the **detailed balance equation**. An ergodic chain in equilibrium satisfying this equation and has π as its unique stationary distribution.

If the transition kernel $K(\cdot, \cdot)$ of the Markov chain preserves π, that is if $X_i \sim \pi$ implies that $X_{i+1} \sim \pi$ or if $P(X_{n+1} \in A | X_n = x) = \int_A K(x, y)dy$ the density π is invariant for the Markov chain. Our goal is to verify that π is the invariant density, which is difficult because we have to integrate with respect to π. That is why we use MCMC: instead of proving that π is invariant, we simply choose the transition kernel that best satisfies the reversibility condition with respect to π. Reversibility holds if (X_n, X_{n+1}) has the same distribution as the time-reversed subchain (X_{n+1}, X_n) whenever X_n has density π.

Varying Dimension between States

Reversibility is not always easy to ascertain. Some algorithms, for example reversible jump samplers, allow the move from s current state to the next state even if the dimensions do not match. The following example shows a way to obtain an appropriate transition kernel for this case.

Consider computing the transition kernel

$$P(x, B) = P(Y \in B | X = x)$$

for the vector $Y = (Y_1, Y_2)'$ and the scalar variable X. Consider that given $X = x$, Y is defined by the deterministic mapping g:

$$(Y_1, Y_2) = g(x, U) = (x + U, x - U)$$

where U is a random variable with density q on $\mathbb{R}$. Given $X = x$, Y does not have a density on $\mathbb{R}^2$ because once Y_2 is known, the value of Y_1 is determined completely. For $A \subset \mathbb{R}$ and $B \subset \mathbb{R}^2$:

$$P(x, B) = P(Y \in B | X = x) = \int I((x + U, x - U) \in B)\, q(u)\, du$$

and

$$P(X \in A, Y \in B) = \int_A \int I((x + U, x - U) \in B)\, q(u)\, f(x)\, du\, dx$$

where f is the density of X on $\mathbb{R}$.

2.3 Monte Carlo Integration

Standard Monte Carlo integration is a method used for numerically solving analytically challenging integrals using random numbers. This method is based on repeated random sampling and is particularly useful for high-dimensional integrals.

Strong Law of Large Numbers. Let $X_1, X_2, \ldots$ be an infinite sequence of i.i.d. random variables with expected value $E(X_1) = E(X_2) = \ldots = \mu$. The strong law of large numbers states that the sample average converges almost surely to the expected value:

$$\overline{X}_n = \frac{1}{n}(X_1 + \cdots + X_n), \;\; \overline{X}_n \to \mu, \; n \to \infty,$$

that is

$$P(\lim_{n\to\infty} \overline{X}_n = \mu) = 1.$$

Note that this result holds for $\mu < \infty$.

Central Limit Theorem Lindeberg-Levy. Suppose $X_1, X_2, \ldots$ is a sequence of i.i.d. random variables with $E(X_i) = \mu$ and $Var(X_i) = \sigma^2 < \infty$. Then as n approaches infinity, the random variables $\sqrt{n}(S_n - \mu)$ converge in distribution to a normal $N(0, \sigma^2)$:

$$\sqrt{n}((\frac{1}{n}\sum_{i=1}^{n} X_i) - \mu) \xrightarrow{d} N(0, \sigma^2).$$

Monte Carlo method. Suppose we want to evaluate an integral of the form

$$I = \int_X h(x)\, f(x)\, dx = E_f[h(x)]$$

where $f(x)$ is a density. This is then recognized as the expected value of $h(x)$ with respect to the density $f(x)$, as shown above. If it is possible to simulate or generate a number of iid $x_i \sim f(x), \; i = 1, \ldots, m$, then the standard Monte Carlo estimator for the integral I is the empirical average, which converges towards the integral for m sufficiently high (strong law of large numbers):

$$\overline{h}_m = \frac{1}{m}\sum_{i=1}^{m} h(x_i) \longrightarrow I, \; m \to \infty$$

This technique of estimating the integral I using generation of iid random variables $x_i \sim f(x)$ is referred to as classical Monte Carlo integration. However, the challenge to generate random variables from $f(x)$ remains, and for some densities direct simulation is difficult. An alternative method to direct sampling for simulating from the distribution $f(x)$ is Markov Chain Monte Carlo.

Monte Carlo variance. The Monte Carlo estimate converges asymptotically to the true value of the integral of interest as the number of simulations increases. In order to evaluate the precision of the Monte Carlo estimate, we will compare the variance of the Monte Carlo estimate to the variance of the integral function $h(x)$. The variance of the Monte Carlo estimate is

$$Var[\overline{h}_m] = Var[\frac{1}{m}\sum_{i=1}^{m} h(x_i)] = \frac{1}{m^2} Var[\sum_{i=1}^{m} h(x_i)] = \frac{1}{m} Var_f[h(x)]$$

where $Var_f[h(x)]$ is the variance of the function $h(x)$ with respect to $f(x)$, since $x_1, \ldots, x_m$ are iid samples from density $f(x)$. Therefore, as $m \to \infty$, $Var_f[h(x)] \to 0$. In general, Monte Carlo variance will increase for dependent samples with a positive correlation structure. If $\overline{h}_m$ is an estimator based on dependent samples of $f(x)$ the corresponding variance becomes

$$\begin{aligned} Var[\overline{h}_m] &= \frac{1}{m^2} Var[\sum_{i=1}^{m} h(x_i)] \\ &= \frac{1}{m^2}[mVar_f[h(x)] + \sum_{i=1}^{m}\sum_{i\neq j} Cov((h(x_i), h(x_j))] \\ &= \frac{1}{m} Var_f[h(x)][1 + \frac{1}{m}\sum_{i=1}^{m}\sum_{i\neq j} Cov((h(x_i), h(x_j))] \end{aligned}$$

Hence, it can be seen that using dependent samples with a positive correlation structure will result in increased variance of the Monte Carlo estimate.

By using the Central Limit Theorem Lindeberg-Levy, we can show that the distribution of the estimator $\overline{h}_m$ is:

$$\begin{aligned} &\frac{\overline{h}_m - E[\overline{h}_m]}{\sqrt{Var[\overline{h}_m]}} \overset{d}{\approx} N(0,1) \\ &\Leftrightarrow N(E[\overline{h}_m], Var[\overline{h}_m]) = N(E_f[h(x)], \frac{1}{m} Var_f[h(x)]) \\ &\Leftrightarrow \overline{h}_m \overset{d}{\approx} N(I, \frac{1}{m} Var_f[h(x)]) \end{aligned}$$

2.4 Markov Chain Monte Carlo

We are going to illustrate an overview of the philosophy of Markov Chain Monte Carlo (MCMC) methods. The basic idea is to choose the transition kernel that generates the Markov chain with the desired stationary distribution. The transition kernel which imposes the strongest reversibility condition with respect to π, is sufficient to guarantee that π is invariant for the Markov chain.

Let X denote a real stochastic vector of unknown parameters or unobservable variables associated with some model, and assume X has a distribution with density π on $\mathbb{R}^d$. Density π could represent a posterior density. However, this density is required in order to evaluate necessary expectations with respect to π and has, in most cases, a complex form that may only be known up to an unknown normalising constant. Thus, it is not possible to calculate these expectations with numerical integration or analytically and simulating directly from the posterior density may be challenging. We are going to show further on, that this problem can be dealt with the construction of a Markov chain whose invariant distribution has density given by π.

In order to construct the Markov chain $\{X_n\}_{n=1}^{\infty}$ we must know:

1. The distribution for the initial state X_1
2. The transition kernel $K(\cdot, \cdot)$ of the conditional distribution for $X_{n+1}|X_n$

3. If $X_n = x$ is the value of the current state, the probability that X_{n+1} is in a set $A \subseteq \mathbb{R}^d$ is

$$K(x, A) = P(X_{n+1} \in A | X_n = x).$$

If the generated Markov chain is irreducible with invariant distribution π, it can be used for Monte Carlo estimation of various expectations $E(h(X))$ with respect to π. That is, for any function h on $\mathbb{R}^d$ with $E(h(X)) < \infty$

$$E(h(X)) = \int h(x)\,\pi(x)\,dx = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} h(X_i)$$

Thus, $E(h(X))$ can be approximated by

$$E(h(X)) \longrightarrow \frac{1}{N} \sum_{i=1}^{N} h(X_i)$$

the sample average, for some large N. Here h could be the indicator function of a set $A \subseteq \mathbb{R}^d$, so that $E(h(X))$ equals the probability

$$P(X \in A) = E(I(X \in A)) \approx \frac{1}{N} \sum_{i=1}^{N} I(X_i \in A)$$

Note that the convergence of the sample average to $E(h(X))$ for all starting values, requires the assumption of Harris recurrence. Even though the invariant distribution π is unique given a transition kernel K, this is not true the other way around. There might be several different transition kernels having the same invariant distribution, which practically means there are different choices of transition kernels to choose from corresponding to varying quality and efficiency of the implementation of the method.

In the following sections we will delve into some algorithms that are implemented via the construction of a Markov chain and Monte Carlo integration. The Metropolis-Hastings algorithm, the Gibbs sampler and the reversible-jump MCMC are presented. Gibbs sampling is a special case of the Metropolis-Hastings algorithm. However it can be considered as a general framework for sampling from a large set of variables, by sampling each variable (or a group of variables) in turn, and can incorporate the Metropolis-Hastings algorithm to implement one or more sampling steps. Reversible-jump Markov chain Monte Carlo (rjMCMC) is an extension of the Metropolis-Hastings algorithm that allows simulation of the posterior distribution on spaces of varying dimension. The proposal and the target distributions have densities on spaces of different dimensions. Thus, the simulation is possible even if the number of parameters in the model is unknown.

Chapter 3

The Metropolis-Hastings Algorithm

In the process of constructing a Markov chain, one of our main concerns is to confirm the existence and uniqueness of an equilibrium distribution for iterations of a given transition kernel. Contrary to that strategy, MCMC methods suggest the opposite: the equilibrium distribution is known, usually up to a constant multiple, but the transition kernel is unknown. Thus, the idea underlying these methods is to generate a Markov chain via iterative Monte Carlo simulation that has the desired posterior distribution as its stationary distribution, at least in an asymptotic sense.

The objective is to sample from the target distribution, however direct sampling in most cases is impossible. The initial step of the Metropolis-Hastings algorithm is to produce candidate draws from a proposal distribution. Under the frame of Markov chain theory, we allow the proposal density to depend on the current state of the process. These candidate draws are thereafter updated in a way that they tend to operate (asymptotically) as random observations from the desired invariant or target distribution. The process by which the Markov chain is generated by the Metropolis-Hastings algorithm at each stage, consists of two basic steps: the proposal step and the acceptance step. The former is associated with the proposal distribution and the latter with the acceptance probability.

The success of the method requires a fair acceptance ratio and a proper proposal density choice. These two features are very important when it comes to the convergence of the chain to its invariant distribution and will be elaborated further on. The implementation of the Metropolis-Hastings algorithm can be made in two different ways: one can update all random variables of the model jointly or update the random variables independently.

The Metropolis sampling algorithm can draw samples from a complex or unnormalized target probability distributions using a Markov chain. The Metropolis algorithm first proposes a possible new state X^* in the Markov chain, based on the previous state X_n, according to the proposal distribution $q(x^*|x_n)$. The algorithm accepts or rejects the proposed state based on the density of the the target distribution $\pi(\cdot)$ evaluated at X^*.

One constraint of the Metropolis sampler is that the proposal distribution $q(x^*|x_n)$ must be symmetric. This constraint originates from the detailed balance condition. However, a symmetric proposal distribution may be a poor fit for many problems. In order to be able to use an asymmetric proposal distributions, the Metropolis-Hastings algorithm implements an additional correction factor c, defined from the proposal distribution as

$$c = \frac{q(x_n|x^*)}{q(x^*|x_n)}$$

The correction factor ensures that the probability of moving from $X_n \to X_{n+1}$ is equal to the probability of moving from $X_{n+1} \to X_n$, no matter the proposal distribution.

3.1 Joint Updating M-H Algorithm

Draw N samples using the Metropolis-Hastings algorithm

1. Set $n = 0$
2. Generate an initial state $X_0 \sim \pi_0$
3. Repeat until $n = N$

   Set $n = n + 1$
   Generate a proposal state X^* from $q(x|x_n)$
   Calculate the proposal correction factor $c = \dfrac{q(x_n|x^*)}{q(x^*|x_n)}$
   Calculate the acceptance probability $\alpha = \min\left(1, \dfrac{\pi(x^*)}{\pi(x_n)} \times c\right)$
   Draw $u \sim U(0,1)$
   If $u \leq \alpha$ accept the proposal state X^* and set $X_{n+1} = X^*$
   Else set $X_n = X^*$

Deriving the acceptance probability

Let X_n denote the n-th state of the Markov chain $X_1, X_2, ...$ and let Y_{n+1} denote the proposal for the next state of the chain. The random vector (X_n, Y_{n+1}), consisting of the current Markov chain state and the proposal, has joint density g on $\mathbb{R}^{2d}$ given by

$$g(x, y) = q(x, y)\, \pi(x)$$

where π is the d-dimensional target density and $q(x, \cdot)$ is the d-dimensional proposal density of Y_{n+1}, given that $X_n = x$, $x \in \mathbb{R}^d$. The acceptance probability of the simultaneous updating Metropolis-Hastings algorithm is derived subject to the reversibility condition

$$P(X_n \in A, X_{n+1} \in B) = P(X_n \in B, X_{n+1} \in A)$$

for all $A, B \subseteq \mathbb{R}^d$.

$$P(X_n \in A, X_{n+1} \in B) = \int_A P(X_{n+1} \in B|X_n = x)\, \pi(x)\, dx.$$

For any $B \subseteq \mathbb{R}^d$ define the proposal distribution as

$$Q(x, B) = P(Y_{n+1} \in B|X_n = x) = \int I(y \in B)\, q(x, y)\, dy$$

which is the conditional probability that Y_{n+1} belongs in a set B given that $X_n = x$.

The conditional probability that Y_{n+1} belongs in a set B and Y_{n+1} is accepted, given that $X_n = x$ is defined as

$$Q^a(x, B) = P(Y_{n+1} \in B \text{ and } Y_{n+1} \text{ is accepted}|X_n = x) = \int I(y \in B)\, q(x, y)\, a(x, y)\, dy.$$

The conditional probability of rejecting the proposal given that $X_n = x$, is defined as

$$r(x) = P(Y_{n+1} \text{ is rejected}|X_n = x)$$

Then the **transition kernel** $K(x, B) = P(X_{n+1} \in B|X_n = x)$ can be written as

$$K(x,B) = Q^a(x,B) + r(x)\, I(x \in B)$$

This expression for the transition kernel, is derived due to the fact that there are two ways in which $X_{n+1} \in B$ (law of total probability). Given that the current state of the chain is $X_n = x$, one generates a proposal Y_{n+1} that belongs in B and therefore $Q^a(x,B)$ is the probability that this candidate draw is accepted, so that if $Y_{n+1} = y$ the chain will move to the new state $X_{n+1} = y$. The other one rejects the proposal, with probability $r(x)$, so that the chain will not move to a new state, that is $X_{n+1} = X_n = x$ and that $x \in B$. Hence:

$$\begin{aligned} &P(X_n \in A, X_{n+1} \in B) \\ &= \int_A Q^a(x,B)\, \pi(x)\, dx + \int_A r(x)\, I(x \in B)\, \pi(x)\, dx \\ &= \int_A Q^a(x,B)\, \pi(x)\, dx + \int r(x)\, I(x \in B \cap A)\, \pi(x)\, dx \end{aligned}$$

and by virtue of symmetry

$$\begin{aligned} &P(X_n \in B, X_{n+1} \in A) \\ &= \int_B Q^a(x',A)\, \pi(x')\, dx' + \int_B r(x')\, I(x' \in A)\, \pi(x')\, dx' \\ &= \int_B Q^a(x',A)\, \pi(x')\, dx' + \int r(x')\, I(x' \in B \cap A)\, \pi(x')\, dx' \end{aligned}$$

The reversibility condition is satisfied if:

$$\int_A Q^a(x,B)\, \pi(x)\, dx = \int_B Q^a(x',A)\, \pi(x')\, dx'$$

$$\int_A \int I(y \in B)\, q(x,y)\, a(x,y)\, \pi(x)\, dx\, dy = \int_B \int I(y' \in A)\, q(x',y')\, a(x',y')\, \pi(x')\, dx'\, dy'$$

$$\int \int I(x \in A, y \in B)\, q(x,y)\, a(x,y)\, \pi(x)\, dx\, dy = \int \int I(x' \in B, y' \in A)\, q(x',y')\, a(x',y')\, \pi(x')\, dx'\, dy'$$

In order to write this expression entirely as a function of the same variables we can set $y = x'$ and $x = y'$. This particular change of variables is feasible due to the following procedure: In the move from $X_n = x$ to $X_{n+1} = x'$, a proposal with realized value y is generated from $q(x,\cdot)$. If the proposal is accepted, $x' = y$. In the opposite move, from $X_n = x'$ to $X_{n+1} = x$, a proposal with realized value y' is generated from $q(x',\cdot)$. If the proposal is accepted, $x = y'$. Also, by taking into consideration that the Jacobian of the transformation is 1, the last expression yields the reversibility condition:

$$\begin{aligned} &\int \int I(x \in A, y \in B)\, q(x,y)\, a(x,y)\, \pi(x)\, dx\, dy \\ &= \int \int I(y \in B, x \in A)\, q(y,x)\, a(y,x)\, \pi(y)\, dy\, dx \end{aligned}$$

Equality is satisfied if

$$q(x,y)\, a(x,y)\, \pi(x) = q(y,x)\, a(y,x)\, \pi(y).$$

By setting $a(y,x) = 1$, the acceptance probability that is derived is the largest possible, subject to the detailed balance equation

$$\alpha(y,x) = 1 \Rightarrow \alpha(x,y) < 1$$

$$\alpha(x,y) = \frac{q(y,x)\,\pi(y)}{q(x,y)\,\pi(x)} < 1$$

therefore

$$a(x,y) = \min\left(1, \frac{q(y,x)\,\pi(y)}{q(x,y)\,\pi(x)}\right)$$

By setting $\alpha(x,y) = 1$, we get the inverse of that:

$$a(y,x) = \min\left(1, \frac{q(x,y)\,\pi(x)}{q(y,x)\,\pi(y)}\right).$$

3.1.1 Sampling from a Beta distribution with MH

Assume that we want to sample from a Beta(a,b) distribution by using the MH algorithm. Note that the support of the target density is $[0,1]$. Let z be the current state. We simulate a candidate point z' from a normal proposal distribution:

$$z'|z \sim N(z, \sigma^2), \;\; \sigma : \text{unknown}$$

Since the proposal distribution is a symmetric normal distribution, for the proposal density $q(\cdot)$ we get:

$$q(z'|z) = q(z|z')$$

thus, in the acceptance probability ratio, proposals cancel out. The corresponding acceptance probability $a(z,z') = \min\{1, A\}$ where

$$\begin{aligned} A &= \frac{\pi(z'|x)\,q(z|z')}{\pi(z|x)\,q(z'|z)} = \frac{\pi(z'|x)}{\pi(z|x)} \\ &= \begin{cases} \frac{z'^{a-1}(1-z')^{b-1}}{z^{a-1}(1-z)^{b-1}}, & z' \in [0,1] \\ 0, & z' \notin [0,1] \end{cases} \\ &= \begin{cases} \left(\frac{z'}{z}\right)^{a-1}\left(\frac{1-z'}{1-z}\right)^{b-1}, & z' \in [0,1] \\ 0, & z' \notin [0,1] \end{cases} \end{aligned}$$

An alternative way for reducing the proportion of proposed moves that are rejected is to ensure that the proposed values always lie in the interval $[0,1]$. This can be done by specifying another proposal distribution that takes values inside $[0,1]$. Two suitable choices would be:

1. A uniform distribution

$$z'|z \sim U(-\epsilon_z, \epsilon_z), \quad \epsilon_z = \min\{\epsilon, z, 1-z\}$$

   which ensures that $z' \in [0,1]$.

2. A truncated normal distribution

$$z'|z \sim TN(z, \sigma^2, 0, 1)$$

   constrained to the interval [0,1].

Thus the acceptance probability is $a(z, z') = \min\{1, A\}$ where

$$A = \frac{\pi(z'|x)\, q(z|z')}{\pi(z|x)\, q(z'|z)} = \left(\frac{z'}{z}\right)^{a-1} \left(\frac{1-z'}{1-z}\right)^{b-1} \frac{q(z|z')}{q(z'|z)}.$$

The proposal densities do not cancel out in these cases, which typically means that more proposed states will be accepted for posterior distributions that take values in the constrained interval, like in this case the beta distribution.

3.2 Single-Site Updating M-H Algorithm

Draw N samples using the Metropolis-Hastings algorithm

1. Set $n = 0$
2. Generate an initial state $X_0 \sim \pi_0$
3. Repeat until $n = N$

   Set $n = n + 1$
   For each dimension $i = 1, \ldots, d$
   Generate a proposal state X_i^* from $q(x_i|x_i)$
   Calculate the proposal correction factor $c = \dfrac{q(x_i|x_i^*)}{q(x_i^*|x_i)}$
   Calculate the acceptance probability $\alpha = \min\left(1, \dfrac{\pi(x_i^*, x_j)}{\pi(x_i, x_j)} \times c\right)$
   Draw $u \sim U(0, 1)$
   If $u \leq \alpha$ accept the proposal state X_i^* and set $X_i^{(n+1)} = X_i^*$
   Else set $X_i^{(n)} = X_i^*$

Note that a sample for the i - th dimension is proposed, then accepted or rejected while all other dimensions, $j \neq i$, are held fixed. We then move on to the next dimension $i + 1$, and repeat the process while holding all other variables fixed ($j \neq i + 1$).

Deriving the acceptance probability

Given that $X_n = x$, $Y_{n+1} = x$ except at the i-th component, where x_i is replaced by a random variable Z_i generated from a one-dimensional proposal density $q_i(x, \cdot)$, which may or may not depend on x or a subset of x. Since

$$Y_{n+1} \in B \Leftrightarrow (x_1, \ldots, x_{i-1}, Z_i, x_{i+1}, \ldots, x_d) \in B,$$

the probability that Y_{n+1} belongs in $B \subseteq \mathbb{R}^d$, given $X_n = x$, is given by the proposal distribution

$$\begin{aligned} Q(x, B) &= P(Y_{n+1} \in B | X_n = x) \\ &= \int I((x_1, \ldots, x_{i-1}, z_i, x_{i+1}, \ldots, x_d) \in B)\, q_i(x, z_i)\, dz_i \end{aligned}$$

which is a one-dimensional integral. Notice that the target density π is on $\mathbb{R}^d$, while the proposal density $q_i(x, \cdot)$ is on $\mathbb{R}$.

Consider the move from a state

$$x = (x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_d)$$

to the state

$$x' = (x_1, \ldots, x_{i-1}, z_i, x_{i+1}, \ldots, x_d).$$

The probability that X_{n+1} belongs in $B|X_n = x$, is given by

$$\begin{aligned} P(X_{n+1} \in B | X_n = x) &= Q^a(x, B) + r(x)\, I(x \in B) \\ &= \int I(x' \in B)\, a(x, x')\, q_i(x, z_i)\, dz_i + r(x)\, I(x \in B) \end{aligned}$$

The reversibility condition can be written as:

$$\begin{gathered} P(X_n \in A, X_{n+1} \in B) = P(X_n \in B, X_{n+1} \in A) \\ \int_A Q^a(x, B)\, \pi(x)\, dx + \int_A r(x)\, I(x \in B)\, \pi(x)\, dx = \int_B Q^a(x', A)\, \pi(x')\, dx' + \int_B r(x')\, I(x' \in A)\, \pi(x')\, dx' \\ \int_A Q^a(x, B)\, \pi(x)\, dx = \int_B Q^a(x', A)\, \pi(x')\, dx' \\ \int_A \int I(x' \in B)\, a(x, x')\, q_i(x, z_i)\, \pi(x)\, dz_i\, dx = \int_B \int I(x \in A)\, a(x', x)\, q_i(x', x_i)\, \pi(x')\, dx_i\, dx' \\ \int_{\mathbb{R}^d} \int_{\mathbb{R}} I(x \in A, x' \in B)\, a(x', x_i)\, q_i(x', x_i)\, \pi(x')\, dx'\, dx_i = \int_{\mathbb{R}^d} \int_{\mathbb{R}} I(x \in A, x' \in B)\, a(x, x')\, q_i(x, z_i)\, \pi(x)\, dz_i\, dx \end{gathered}$$

which are equal if:

$$q_i(x, z_i)\, a(x, x')\, \pi(x) = q_i(x', x_i)\, a(x', x_i)\, \pi(x').$$

Setting $a(x', x_i) = 1$, yields the acceptance probability

$$a(x, x') = \min\left(1, \frac{q_i(x', x_i)\, \pi(x')}{q_i(x, z_i)\, \pi(x)}\right)$$

and similarly we get the inverse by setting $a(x, x') = 1$:

$$a(x', x_i) = \min\left(1, \frac{q_i(x, z_i)\, \pi(x)}{q_i(x', x_i)\, \pi(x')}\right).$$

The arguments above also hold when the updating variable, rather than being a scalar, is a vector and a subset of x.

3.3 Implementation Issues

Proper Choice of the Proposal Distribution $q(\cdot)$

For the implementation of the Metropolis-Hastings algorithm, a main issue is to choose a proper density for the candidate draws. This density is chosen from a family of distributions and requires proper tuning of the location and scale parameters.

1. *Random walk update.* Letting $q(x, y) = q(y, x)$ yields the acceptance rate

$$a(x, y) = \min\{1, \frac{\pi(y)}{\pi(x)}\}.$$

The usual definition of random walk for $q(x, y)$ in $\mathbb{R}^d$ assumes $q(x, y) = q(y - x)$ or equivalently that the proposal random variable Y_{n+1} satisfies

$$(Y_{n+1}|X_n = x) \stackrel{d}{=} x + W_{n+1}, \text{ such that } W_{n+1} \stackrel{iid}{\sim} f_W(\cdot).$$

Because $q(x, y) = q(y, x)$ we have

$$f_W(y - x) = f_W(-(y - x)) \Leftrightarrow f_W(w) = f_W(-w) \Leftrightarrow W \stackrel{d}{=} -W$$

or in words, that the distribution of W will be symmetric around 0.

While the behaviour of the Markov chain generated by $f_W(\cdot)$ and $\pi(\cdot)$ is less sensitive to the shape of the distribution of $f_W(\cdot)$, it is sensitive to the size of the steps. Usually the equation

$$(Y_{n+1}|X_n = x) \stackrel{d}{=} x + W_{n+1}$$

is written in the form

$$(Y_{n+1}|X_n = x) \stackrel{d}{=} x + h\, W_{n+1},\ h > 0$$

then the value of h is chosen so that the Markov chain has good convergence properties. As an example consider the case where $W_{n+1} \stackrel{iid}{\sim} N(\cdot|0, 1)$ and $h = \sigma$ then

$$(Y_{n+1}|X_n = x) \stackrel{d}{=} x + \sigma\, N(0, 1) \stackrel{d}{=} N(x, \sigma^2)$$

As a rule of thumb, conventional wisdom is to adjust h empirically so that the acceptance ratio is in the range 30-50%, with 30-40% tending to work better than 40-50 %. Metropolis random walks with a smaller acceptance ratio (generally due to a larger value of h) are said to take "fewer but higher quality steps".

2. *Independence chain.* A second family of candidate generating densities, is given by $q(x, y) = q_2(y)$, as it appears in Hastings (1970). In this case, the candidate observation is drawn independently of the current state of the chain. The acceptance probability can be written as:

$$a(x, y) = \min\left\{1, \frac{w(y)}{w(x)}\right\}$$

where $w(y) = \pi(y)/q_2(y)$ is the importance weight function that would be used in importance sampling given observations generated from $q_2(\cdot)$. As in the random walk case, we can let q_2 be a multivariate normal or a multivariate-t density, with the difference that the location parameter as well as the spread need to be specified.

3. According to Chib and Greenberg (1994), we can define a candidate-generating density, using the known form of $\pi(\cdot)$. Let us assume that if $\pi(t)$ can be written as $\pi(t) \propto \psi(t)h(t)$, where $h(t)$ is a density of a known distribution that we can sample from, and $\psi(t)$ is uniformly bounded, then as in the independence chain case, we set $q(x, y) = h(y)$ to draw candidates. Thus the probability of move requires only the computation of the ψ function and reduces to

$$a(x, y) = \min\left\{1, \frac{\psi(y)}{\psi(x)}\right\}.$$

4. *Gibbs sampler.* A special case of the single-update MH algorithm arises if we set the proposal distribution for any parameter to be the conditional posterior distribution of that parameter given the current value of the others. In this case, the acceptance probability is always exactly 1. This is known as the Gibbs sampler (Casella and George 1992) and in the next chapter we discuss it thoroughly.

Acceptance Rate

The acceptance probability of a move for the Metropolis-Hastings algorithm is generally defined as

$$a(x, y) = \min\left\{1, \frac{q(y, x)\,\pi(y)}{q(x, y)\,\pi(x)}\right\}, \qquad \pi(x)\,q(x, y) > 0$$

and reduces to

$$a(x, y) = \min\left\{1, \frac{\pi(y)}{\pi(x)}\right\}, \qquad \pi(x) > 0$$

when symmetrical proposal densities are considered.

Let us assume the initial state of the chain x_0 with $\pi(x_0) > 0$, then for every n, $\pi(x_n) > 0$ since values of the proposal Y for which $\pi(y) = 0$ lead to $a(x, y) = 0$, these proposal moves are rejected. Achieving a fair rate of acceptance is an issue of great importance for the successful implementation of the method. Parameterization and the proper proposal density choice play a key role. Although we want to achieve a high acceptance ratio, acceptance rates in the neighbourhood of 1 imply strong similarity between current and proposed states. This would cause really slow movement of the chain. Note that the only exception is in the case where the proposal density is the invariant distribution. The opposite scenario is that the rejection rate can be too high. Here the proposed relocation is too large and the potential move falls outside the support of the posterior. A high rejection rate practically means that the chain will remain in the same state for many iterations. In these cases, convergence is not ensured.

Convergence

1. *Run length.* There are two elements to be considered when determining the simulation length: the time required for convergence, and the post-convergence sample size required for small Monte Carlo errors. We want to determine how long it takes for the Markov chain to converge to the target distribution. In practice, we discard observations from the start of the chain, during the burn-in period and just use observations from the chain once it has converged. The simplest method to determine the length of the burn-in period is to look at trace plots. You can often see the individual parameters converging from their starting position to values based around a constant mean. The use of trace plots is a fairly efficient method, but it is not robust.

   Running several replications from different over-dispersed starting points provides additional reassurance when one is trying to check that convergence has been achieved. Basically, if you run the

chain several times from different starting points and they all give you the same posterior estimates then this suggests that no major modes have been missed in any one simulation and that each has probably converged. This approach is formalised in the Brooks-Gelman-Rubin diagnostic (Brooks and Gelman 1998). There are various implementations of this diagnostic procedure, all based upon the idea of using an analysis of variance to determine whether or not there are any differences in estimates from different replications.

2. *Monte Carlo error.* MCMC integration is a method of estimating statistics of interest. It is a simulation-based estimation technique and, as such, is subject to what we call Monte Carlo error, which decreases with increasing sample size. Monte Carlo error essentially measures the variation you would expect to see in your estimates if you ran multiple replications of your MCMC chain. It is related to the autocorrelation of your chain, but if the sample size increases by a factor n, then the Monte Carlo error decreases by a factor $\sqrt{n}$ (Ripley 1987).

   As we have shown earlier, the ergodic average satisfies the central limit theorem. Suppose we wish to estimate the posterior mean of a parameter θ. By using Monte Carlo integration we estimate the posterior mean $E_\pi(\theta|x)$ by the sample mean $\overline{\theta} = 1/n \sum_{i=1}^{n} \theta_i$, thus

$$\overline{\theta} \sim N\left(E_\pi(\theta|x), \frac{\sigma^2}{n}\right)$$

   and $\sigma/\sqrt{n}$ is the Monte Carlo error we wish to estimate.

3. *Pilot tuning.* With the exception of the Gibbs update, most MCMC updates require a degree of so-called pilot tuning in order to ensure adequate convergence and acceptable Monte Carlo errors. Firstly, the proposal distributions for the parameters in the model are specified. Once the proposal distribution is defined, pilot tuning typically involves adjusting the relevant proposal variances so as to obtain MH acceptance probabilities of between 20 and 40% (Gelman et al. 1996). In many cases this can be automated by running the algorithm for some iterations, calculating the average acceptance ratio for each parameter during that time and then increasing the corresponding proposal variance if the acceptance probabilities are too high and decreasing the variances if the probabilities are too low.

4. *Autocorrelation Function.* The performance of the chain, in terms of mixing and moving around the parameter space, is often initially monitored visually with trace plots, or by computing the mean acceptance rate. However, another useful tool is the autocorrelation function (ACF). This is simply defined as the correlation between the given parameter value in the Markov chain separated by k iterations. Ideally, for good mixing chains, there should be a fast decrease in the value of the autocorrelation function as the lag increases. In an ACF plot, this would be represented by a sharp gradient at low values of k. This would imply that there is little relationship between values of the Markov chain within a small number of iterations. Conversely, poorly mixing chains will have a very shallow gradient in the ACF plot, with high autocorrelation values for even relatively large values of k.

5. *Thinning.* In order to make the sampler more efficient computationally and avoid autocorrelations within the chain, we use only every i - th step of the chain, a process known as thinning. The thinned values have reduced autocorrelation, but a large number of sampled values is discarded, which, even if they are highly correlated, still provide information concerning the posterior distribution.

3.3.1 Simulation from a Normal distribution with different proposals

Suppose that the target distribution that we want to sample from, is the weighted sum of two normal distributions:

$$X \sim p\, N(\mu_1, {\sigma_1}^2) + (1-p)\, N(\mu_2, {\sigma_2}^2)$$

with probability density function:

$$\begin{aligned} f(x) &= p\, \frac{1}{\sqrt{2\pi {\sigma_1}^2}} \exp\left\{-\frac{(x-\mu_1)^2}{2{\sigma_1}^2}\right\} + (1-p)\, \frac{1}{\sqrt{2\pi {\sigma_2}^2}} \exp\left\{-\frac{(x-\mu_2)^2}{2{\sigma_2}^2}\right\} \\ &\propto p\, \exp\left\{-\frac{(x-\mu_1)^2}{2{\sigma_1}^2}\right\} + (1-p)\, \exp\left\{-\frac{(x-\mu_2)^2}{2{\sigma_2}^2}\right\} \end{aligned}$$

Now, let us define a proposal that samples from a normal distribution centred on the current point, with standard deviation σ_p:

$$X^* \sim N(x, \sigma_p^2)$$

and density:

$$q(x^*) = \frac{1}{\sqrt{2\pi {\sigma_p}^2}} \exp\left\{-\frac{(x^*-x)^2}{2{\sigma_p}^2}\right\}.$$

The acceptance probability yields:

$$A = \min\{1, a\} = \min\left\{1, \frac{f(x^*)}{f(x)}\right\}.$$

Notice that there is no proposal density ratio, because the normal distribution is symmetrical.

We set $\mu_1 = -2, \sigma_1^2 = 0.5$, $\mu_2 = 1.5, \sigma_2^2 = 1.5$ and also set as an initial state $X = -10$. We perform the move from the current state X to a proposed state X^*, through three different proposals.

1. *(slow step)* $X^* \sim N(x, \sigma_p^2 = 0.2^2)$
2. *(normal step)* $X^* \sim N(x, \sigma_p^2 = 4^2)$
3. *(fast step)* $X^* \sim N(x, \sigma_p^2 = 30^2)$

The aim is to inspect the best proposal we can use to generate samples that come from the target density at hand.

We run the Metropolis Hastings algorithm for 100000 iterations. We want to check the chain's rate of convergence to the stationary distribution (posterior), as well as how it is mixing and moving around the parameter space. We do so graphically:

Traceplot: A plot of the iteration number against the value of the draw of the parameter at each iteration. The traceplot indicates the chain's mixing.

Histogram: A histogram of the posterior draws.

Running mean: A plot of the iterations against the cumulative sum of the draws up to each iteration. We can check the convergence of the draws to the ergodic average.

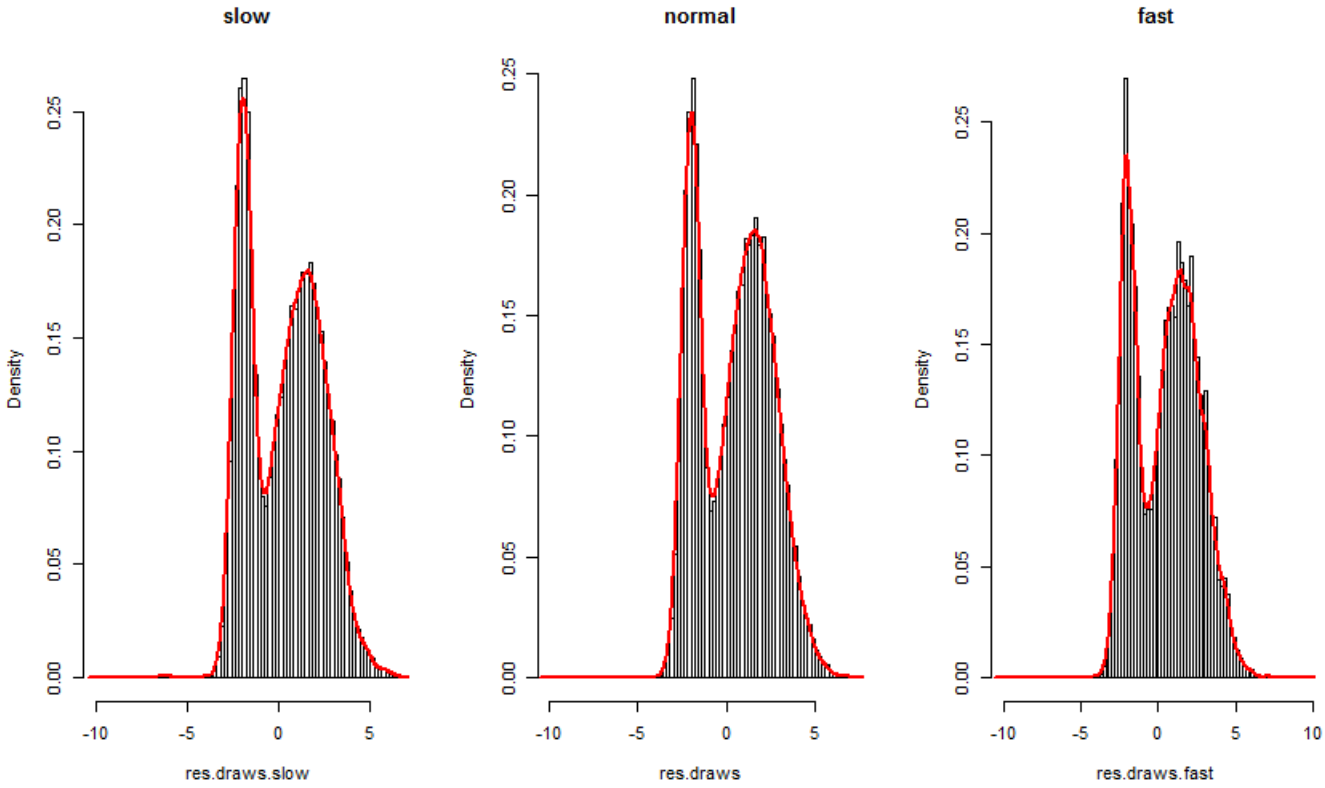

Figure 3.1: Histogram for each proposal.

The first two histograms seem to approximate the target distribution quite well. However, we should also check how the chains are mixing.

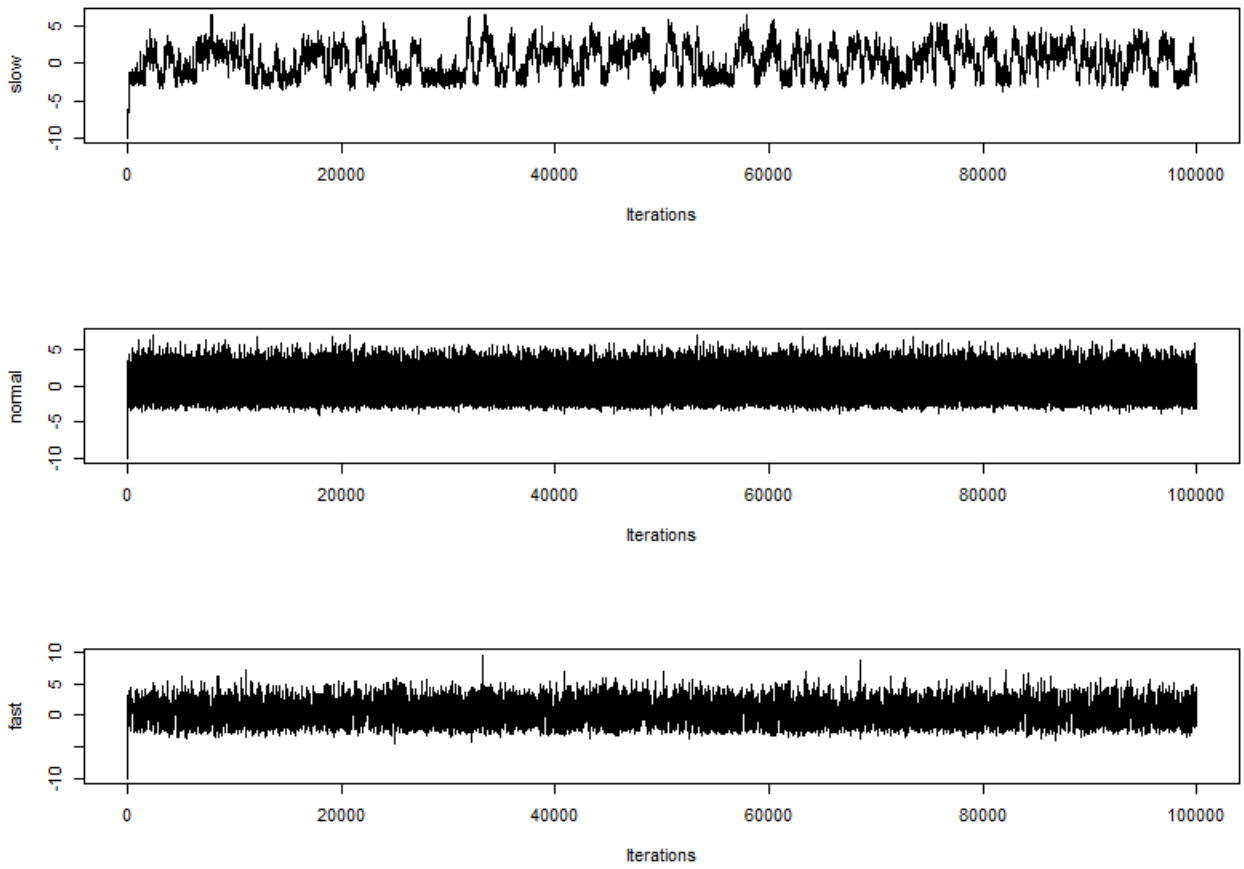

Figure 3.2: Traceplot for each proposal.

The traceplot of the first chain indicates worse mixing than that of the second chain. Up until know, we could say that quite possibly the second proposal is a better fit for our model.

We can observe the effect of different proposal steps in the autocorrelation among subsequent parameters. The autocorrelation function (ACF) plots show the decay in autocorrelation coefficient between steps of different lags, with the blue lines indicating statistical independence. High autocorrelations within chains indicate slow mixing and, usually, slow convergence.

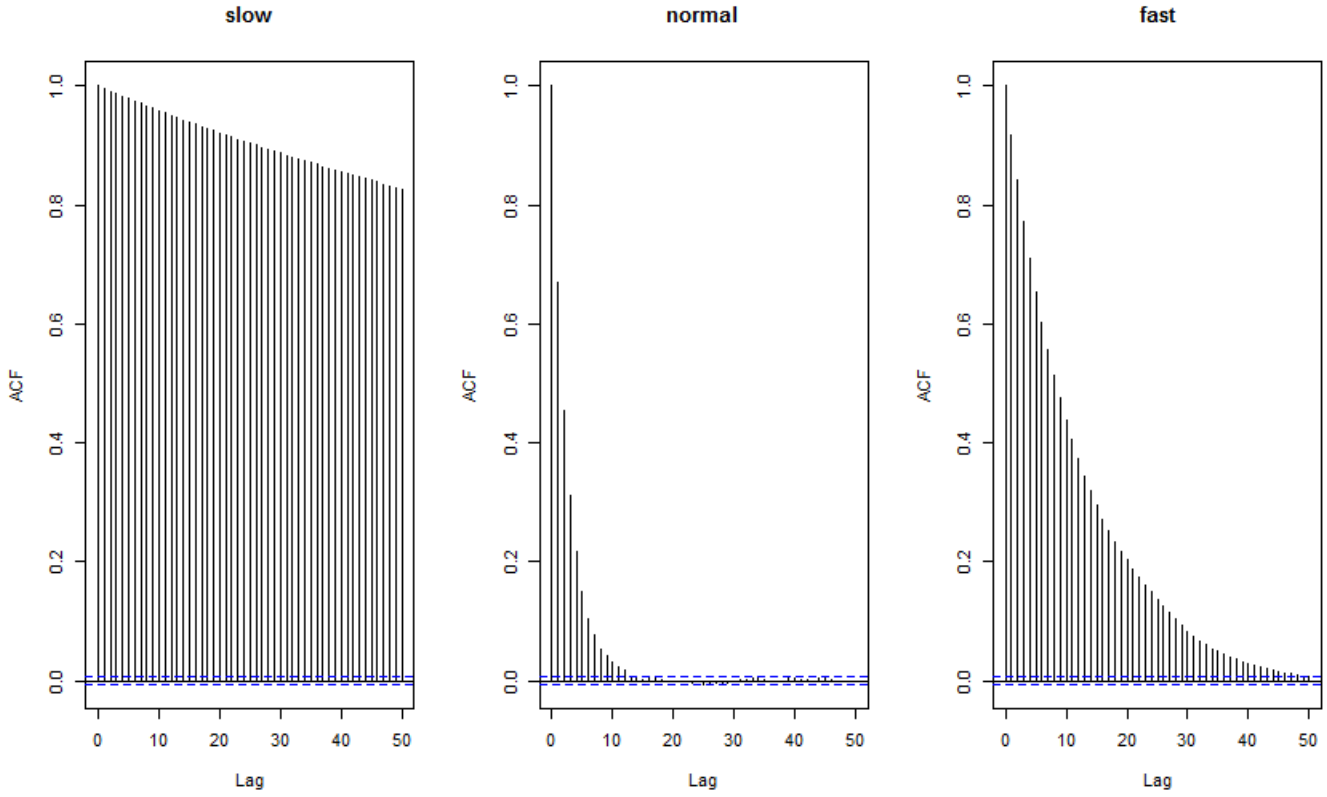

Figure 3.3: ACF plot for each proposal.

Chain	Lag 0	Lag 1	Lag 5	Lag 10	Lag 50
Slow step Proposal Chain	1.00	0.995	0.978	0.958	0.826
Normal step Proposal Chain	1.00	0.671	0.151	0.031	-0.002
Fast step Proposal Chain	1.00	0.917	0.654	0.438	0.007

The second chain has a faster decaying gradient than the other two chains, which indicates faster convergence. The other two chains mix worse than the normal step proposal chain.

The effective sample size for an accurate estimate of the posterior means for each chain is:

Chain	Effective Sample Size
Slow step Proposal Chain	196
Normal step Proposal Chain	18940
Fast step Proposal Chain	4125

Values over 4000 are generally acceptable, so we can conclude that the normal step and fast step chains probably provide a fair estimate of the posterior mean. Nonetheless, the normal step proposal seems to be more effective.

The acceptance rates for each chain are:

Chain	Acceptance Rate
Slow step Proposal Chain	0.94
Normal step Proposal Chain	0.47
Fast step Proposal Chain	0.078

Although we want to achieve a high acceptance ratio, acceptance rates in the neighbourhood of 1 imply strong similarity between current and proposed states. This would cause really slow movement of the chain. The only exception is if the proposal density is the invariant distribution. On the contrary, a high rejection rate, indicates that the proposed relocation is too large and the potential move falls outside the support of the posterior. A high rejection rate practically means that the chain will remain in the same state for many iterations. In these cases, convergence is not ensured. Eventually, the best proposal is the normal step proposal in our case.

Now that we have come to a conclusion about the best proposal density choice, we will run some further diagnostics. First, we compute the empirical means, the standard deviation and the standard error of the mean, for the posterior mean μ:

Variable	Mean	SD	SE
μ	0.458	2.067	0.007

The running average plot shows fast convergence of the draws to the ergodic average (red line):

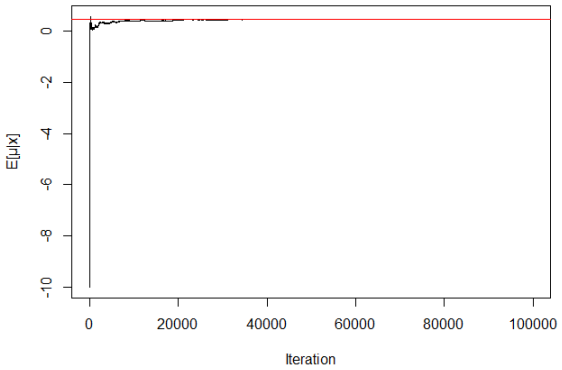

Figure 3.4: Running average tranjectory of μ.

Now, we monitor convergence with the use of some diagnostic tests.

Geweke: The null hypothesis that we test with Geweke diagnostic, is whether the means of two non-overlapping parts of the MC come from the same distribution.

Heidelberg and Welch: The null hypothesis is that the MC comes from a stationary distribution.

Raferty and Lewis: A run length control diagnostic based on a criterion of accuracy of estimation of the quantile q. It is intended for use on a short pilot run of a Markov chain. Note that values of the dependence factor larger than 5 indicate strong autocorrelation which may be due to a poor choice of starting value, high posterior correlations or 'stickiness' of the MCMC algorithm.

Gelman and Rubin: We run $m \geq 2$ chains of length $2n$ from overdispersed starting values. Then, we discard the first n draws in each chain. We calculate the within-chain and between-chain variance and the estimated variance of the parameter as a weighted sum of the within-chain and between-chain variance. Finally, we calculate the potential scale reduction factor (psrf). Note that, necessarily, prsf ≤ 1 to ensure convergence to the posterior distribution.

	Geweke	Heidelberg and Welch					
Variable	Z-score	Stationarity test	Start	P-value	Halfwidth	Mean	Halfwidth test
μ	-1.2	✓	1	0.684	0.029	0.458	✓

As we see in the diagnostic results, the posterior means satisfies the null hypotheses of both tests.

For the quantile $q = 0.025$, within an accuracy of $r + / - 0.005$ with probability $s = 0.95$, the Raferty and Lewis diagnostic yields:

	Raferty and Lewis			
Variable	Burn-in	Required SS	Minimum SS	Dependence Factor
μ	12	12837	3746	3.43

Note that high dependence factors ($I > 5$) are worrisome and may be due to influential starting values, high correlations between coefficients, or poor mixing.

We run 5 chains with 30000 iterations each, at different (overdispersed) starting values. The results give us the median potential scale reduction factor and its 97.5% quantile:

1. psrf=1
2. 97.5% quantile=1

We can see how the psrf changes through the iterations:

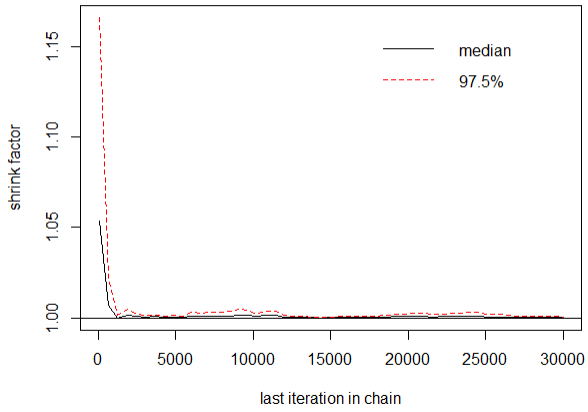

Figure 3.5: The evolution of Gelman and Rubin's shrink factor as the number of iterations increases.

Chapter 4

The Gibbs Sampler

The Gibbs sampler is an MCMC method used to generate random variables, without necessarily computing the probability density function. The characteristic of this algorithm, is that it concentrates as much information as possible from the density of interest. In that way it allows the deconstruction of a complex problem, into simpler ones. For example, the Gibbs sampler makes it possible to draw from several low-dimensional distributions instead of drawing from a single high-dimensional distribution. However, there is a chance that it may take a lot of time for this sequence of low-dimensional distributions to converge but the Gibbs sampler is designed so that the draws from the conditional distributions approximate the joint distribution in an appointed time.

Suppose that the distribution of interest is $\pi(x)$ with $x = (x_1, x_2, ..., x_d)$. The idea is to construct a Markov chain that converges to the distribution $\pi(x)$, and has a transition kernel formed by the fully conditional distributions. In our first encounter with the Gibbs sampler we are going to show that it actually is a special case of the Metropolis-Hastings algorithm. Let us denote the move from x to x', with the proposal being generated from

$$q_i(x, z_i) = \pi(z_i|x_{-i})$$

where $x_{-i} = x$, with the i-th component omitted, that is $x_{-i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_d)$. In the move from x' to x, the proposal is generated from

$$q_i(x', x_i) = \pi(x_i|x_{-i}).$$

The conditional densities that appear are known as the fully conditional posterior distributions (FC). These equal to

$$\pi(z_i|x_{-i}) = \frac{\pi(x')}{\pi(x_{-i})}$$

and

$$\pi(x_i|x_{-i}) = \frac{\pi(x)}{\pi(x_{-i})}.$$

Substituting the FCs on the Metropolis-Hastings acceptance probability $a(x, x')$ yields:

$$\frac{q_i(x', x_i)\,\pi(x')}{q_i(x, z_i)\,\pi(x)} = \frac{\pi(x)\,\pi(x')\,\pi(x_{-i})}{\pi(x)\,\pi(x')\,\pi(x_{-i})} = 1$$

which shows that if we generate a Metropolis-Hastings proposal from an appropriate fully conditional distribution, this proposal is accepted with probability 1. This is the scheme known as Gibbs sampling.

The transition kernel of the Gibbs sampler preserves π, which means that the density π is invariant for the Markov Chain produced. The updating of all elements of x in terms of the transition kernel involves the product

$$\pi(z_i|x_2, x_3, \ldots, x_d)\, \pi(z_2|z_1, x_3, \ldots, x_d) \ldots \pi(z_d|z_1, z_2, \ldots, z_{d-1})$$

and according to the equation $\int_{\mathbb{R}^d} K(x, B)\, \pi(x)\, dx = \int_B \pi(x)\, dx$, by setting $d = 3$ for simplicity, we get

$$K(x, B) = \int I(z_1, z_2, z_3 \in B)\, \pi(z_1|z_1, x_3)\, \pi(z_2|z_1, x_3)\, \pi(z_3|z_1, z_2)\, dz_1\, dz_2\, dz_3.$$

Hence we result in an integral that is 3-dimensional. Integrating over the distribution of x_1, x_2 and x_3 we get

$$\begin{aligned} &K(x, B)\, \pi(x_1, x_2, x_3)\, dx_1\, dx_2\, dx_3 = \\ &= \int I(z_1, z_2, z_3 \in B)\, \pi(z_1, z_2, z_3)\, dz_1\, dz_2\, dz_3 \\ &= P(X_1, X_2, X_3 \in B) \end{aligned}$$

The abovementioned equation is still satisfied if we define the transition kernel with respect to only one of the elements of x. If we want to update only the first element of x in the 3-dimensional example, the transition kernel is:

$$K(x, B) = \int I(z_1, x_2, x_3 \in B)\, \pi(z_1|x_2, x_3)\, dz_1$$

and

$$\int K(x, B)\, \pi(x)\, dx = \int \int I(z_1, x_2, x_3 \in B)\, \pi(z_1|x_2, x_3)\, \pi(x_1, x_2, x_3)\, dz_1\, dx_1\, dx_2\, dx_3.$$

Integrating over the distribution of x_1 yields

$$\int I(z_1, x_2, x_3 \in B)\, \pi(z_1, x_2, x_3)\, dz_1\, dx_2\, dx_3 = P(X_1, X_2, X_3 \in B).$$

4.1 Fully Conditional Posterior Distributions

Let $(\theta_1, \theta_2, ..., \theta_d)$ denote the vector of parameters and $\theta_{-i} = (\theta_1, \theta_2, ..., \theta_{i-1}, \theta_{i+1}, ..., \theta_d)$ denote the vector equal to θ with its i-th component θ_i omitted. Note that θ_{-i}, is of dimension $(d-p)$, $d > p$, $p \geq 1$, where p is the number of elements in θ_i. The form of θ_i can be either scalar, vector or matrix. The fully conditional posterior distribution of θ_i is

$$\begin{aligned} \pi\left(\theta_i \mid \theta_{-i}, y\right) &= \frac{\pi\left(\theta_1, ..., \theta_{i-1}, \theta_i, \theta_{i+1}, ..., \theta_d \mid y\right)}{\int \pi\left(\theta_1, ..., \theta_{i-1}, \theta_i, \theta_{i+1}, ..., \theta_d \mid y\right)\, d\theta_i} \\ &\propto \pi\left(\theta_1, ..., \theta_{i-1}, \theta_i, \theta_{i+1}, ..., \theta_d \mid y\right) \end{aligned}$$

The Gibbs Sampling Algorithm

1. Consider a user-defined vector of starting values

$$\theta^{(0)} = \left(\theta_1^{(0)}, \theta_2^{(0)}, ..., \theta_d^{(0)}\right).$$

These values should be 'legal' in the sense that their posterior distribution exists and is greater than zero, that is $\pi\left(\theta_1^{(0)}, \theta_2^{(0)}, ..., \theta_d^{(0)} \mid y\right) > 0$.

2. The Gibbs sampler is the realization of iterating through the loop

$$\begin{gathered}
\text{draw } \theta_1^{(1)} \text{ from } \pi\left(\theta_1 \mid \theta_2^{(0)}, ..., \theta_d^{(0)}, y\right) \\
\text{draw } \theta_2^{(1)} \text{ from } \pi\left(\theta_2 \mid \theta_1^{(1)}, \theta_3^{(0)}, ..., \theta_d^{(0)}, y\right) \\
\text{draw } \theta_3^{(1)} \text{ from } \pi\left(\theta_3 \mid \theta_1^{(1)}, \theta_2^{(1)}, \theta_4^{(0)}, ..., \theta_d^{(0)}, y\right) \\
\vdots \\
\text{draw } \theta_d^{(1)} \text{ from } \pi\left(\theta_d \mid \theta_1^{(1)}, ..., \theta_{d-1}^{(1)}, y\right) \\
\text{draw } \theta_1^{(2)} \text{ from } \pi\left(\theta_1 \mid \theta_2^{(1)}, ..., \theta_d^{(1)}, y\right) \\
\vdots \\
\text{and so on.}
\end{gathered}$$

The idea is to let the sampler iterate through that loop until convergence is reached, which means that the result of the j-th iteration $\theta_i^{(j)}$ after convergence, is regarded as a draw from its marginal posterior distribution with density $\pi(\theta_i \mid y)$. We should take into consideration, that there is an initial period during which the samples drawn are highly correlated and dependent on the starting values. This is known as the burn-in period and its length can be approximately determined by monitoring the rate of convergence of the Markov chain.

An important conclusion, is the fact that the form of the joint distribution is uniquely shaped by the form of the fully conditional distributions. The draws from all fully conditional posterior distributions are the ones that constitute the sample from the joint distribution via the Gibbs sampling algorithm. We are going to present a proof of that assumption for the two-dimensional case. Consider the density

$$\pi(x, y) = \pi(y \mid x)\ \pi(x) = \pi(x \mid y)\ \pi(y)$$

it follows that

$$\pi(y) = \frac{\pi(y \mid x)}{\pi(x \mid y)}\,\pi(x) \propto \frac{\pi(y \mid x)}{\pi(x \mid y)}$$

hence the normalized marginal density is

$$\pi(y) = \frac{\pi(y \mid x)/\pi(x \mid y)}{\int \pi(y \mid x)/\pi(x \mid y)\ dy}.$$

Eventually it yields

$$\pi(x, y) = \frac{\pi(y \mid x)}{\int \pi(y \mid x)/\pi(x \mid y)\ dy}.$$

Assuming of course that the joint distribution of random variables X, Y exists, it follows that it can be written in terms of the fully conditional posterior distributions. Note that the form of the joint distribution may be impossible to be expressed analytically.

4.1.1 A Zero-Inflated Poisson Model

A zero-inflated Poisson model concerns a random event containing excess zero-count data in unit time, for example, the number of insurance claims within a population for a certain type of risk would be zero-inflated by those people who have not taken out insurance against the risk and thus are unable to claim. The zero-inflated Poisson model employs two components that correspond to two zero generating processes. The first process is governed by a binary distribution that generates structural zeros. The second process is governed by a Poisson distribution that generates counts, some of which may be zero.

We assume that random observations $X_1, \ldots, X_n$ are of the form

$$X_i = R_i\, Y_i, \text{ where } Y_i \overset{iid}{\sim} P(\lambda),\ R_i \overset{iid}{\sim} Bernoulli(p),\ \ 1 \le i \le n$$

and R_i, Y_i are independent. Thus, the data $X_1, \ldots, X_n$ have the following distribution:

$$x_i|\mathbf{r}, \lambda, p \sim P(\lambda\, r_i) \ \text{ independently, } \ 1 \le i \le n$$

and vector $r_i = (r_1, \ldots, r_n)$ has distribution:

$$r_i|p, \lambda \sim Bernoulli(p) \ \text{ independently, } \ 1 \le i \le n.$$

Given an outcome $x = (x_1, \ldots, x_n)$, the objective is to estimate both λ and p. We assign a-priori distributions for parameters λ, p:

$$\begin{aligned} p &\sim U(0,1) \\ \lambda|p &\sim Ga(a,b) \end{aligned}$$

where a, b are hyperparameters and assumed known. The joint probability density function of all parameters and the data x of the hierarchical model is:

$$\begin{aligned} \pi(x, r_i, \lambda, p) &= \pi(x|r_i, \lambda, p)\, \pi(r_i|p, \lambda)\, \pi(\lambda|p)\, \pi(p) \\ &= \prod_{i=1}^{n} \frac{e^{-\lambda r_i}(\lambda r_i)^{x_i}}{x_i!} \times p^{r_i}(1-p)^{1-r_i} \times \frac{b^a}{\Gamma(a)}\lambda^{a-1}e^{-b\lambda} \times 1 \\ &= \frac{b^a \lambda^{a-1} e^{-\lambda b}}{\Gamma(a)} e^{-\lambda n\overline{r}} p^{n\overline{r}}\, (1-p)^{n-n\overline{r}} \lambda^{n\overline{x}} \prod_{i=1}^{n} \frac{r_i^{x_i}}{x_i!} \end{aligned}$$

where $n\overline{r} = \sum_i r_i, n\overline{x} = \sum_i x_i$. The posterior density of the model is:

$$\pi(\mathbf{r}, \lambda, p|x) \propto \pi(x, \mathbf{r}, \lambda, p) = \frac{b^a \lambda^{a-1} e^{-\lambda b}}{\Gamma(a)}\, e^{-\lambda n\overline{r}}\, p^{n\overline{r}}\, (1-p)^{n-n\overline{r}}\, \lambda^{n\overline{x}} \prod_{i=1}^{n} \frac{r_i^{x_i}}{x_i!}$$

which is the kernel of a nonstandard high-dimenional distribution, very challenging to solve analytically. We will use the Gibbs sampling algorithm, thus, we need to compute the fully conditionals of the posterior.

Full Conditional Distribution of λ

$$\begin{aligned}\pi(\lambda|\ldots) \propto \pi(r,\lambda,p|x) &\overset{\lambda}{\propto} \lambda^{a-1}e^{-\lambda b}\, e^{-\lambda n\overline{r}}\, \lambda^{n\overline{x}} \\ &\propto \lambda^{a+n\overline{x}-1}e^{-\lambda(b+n\overline{r}} \\ &\propto Ga\left(a+n\overline{x}, b+n\overline{r}\right)\end{aligned}$$

Full Conditional Distribution of p

$$\begin{aligned}\pi(p|\ldots) \propto \pi(r,\lambda,p|x) &\overset{p}{\propto} p^{n\overline{r}}\left(1-p\right)^{n-n\overline{r}} \\ &\propto Be\left(n\overline{r}+1, n-n\overline{r}+1\right)\end{aligned}$$

Full Conditional Distribution of $r_i,\ i=1,\ldots,n$

$$\begin{aligned}\pi(r_i|\ldots) \propto \pi(r,\lambda,p|x) &\overset{r_i}{\propto} e^{-\lambda n\overline{r}}\, p^{n\overline{r}}\left(1-p\right)^{n-n\overline{r}} \prod_{i=1}^{n} r_i^{x_i} \\ &\overset{(i=k)}{\propto} e^{-\lambda r_k} p^{r_k}(1-p)^{n-r_k} r_k^{x_k} \propto \frac{p^{r_k}}{e^{\lambda r_k}(1-p)^{r_k}} r_k^{x_k} \\ &\propto \left(\frac{p\, e^{-\lambda}}{1-p}\right)^{r_k} r_k^{x_k} \propto Bernoulli\left(\frac{pe^{-\lambda}}{pe^{-\lambda}+(1-p)I_{\{x_k=0\}}}\right)\end{aligned}$$

The full conditional distribution of r_i is this Bernoulli, since:

$$\begin{cases} P\{r_k=1|\ldots\} = \frac{pe^{-\lambda}}{pe^{-\lambda}+(1-p)I_{\{x_k=0\}}} \\ P\{r_k=0|\ldots\} = \frac{pe^{-\lambda}+(1-p)I_{\{x_k=0\}}-pe^{-\lambda}}{pe^{-\lambda}+(1-p)I_{\{x_k=0\}}} \end{cases} \Rightarrow \begin{cases} P\{r_k=1|\ldots\}=1 \\ P\{r_k=0|\ldots\}=0 \end{cases}, \quad x_k \neq 0$$

$$\Rightarrow \begin{cases} P\{r_k=1|\ldots\} = \frac{pe^{-\lambda}}{pe^{-\lambda}+(1-p)} \\ P\{r_k=0|\ldots\} = \frac{(1-p)}{pe^{-\lambda}+(1-p)} \end{cases}, \quad x_k = 0$$

- $x_k = 0$:

$$f(r_k|\ldots) \propto \left(\frac{p\, e^{-\lambda}}{1-p}\right)^{r_k} {r_k}^{x_k} \overset{(x_k=0)}{\Rightarrow} f(r_k|x_k=0,\ldots) \propto \left(\frac{p\, e^{-\lambda}}{1-p}\right)^{r_k}$$

$$\Rightarrow \begin{cases} f\left(r_k=1|x_k=0,\ldots\right) = \frac{\frac{pe^{-\lambda}}{1-p}}{1+\frac{pe^{-\lambda}}{1-p}} = \frac{pe^{-\lambda}}{pe^{-\lambda}+(1-p)} \\ f\left(r_k=0|x_k=0,\ldots\right) = \frac{(1-p)}{pe^{-\lambda}+(1-p)} \end{cases}$$

- $x_k \neq 0$:

$$\begin{aligned} x_k &\in \{1,2,\ldots\} \\ r_k &\in \{0,1\} \end{aligned}$$

$$r_k = 0 \Rightarrow f(r_k=0|x_k\neq 0,\ldots) \propto 0 \Rightarrow f(r_k=1|x_k\neq 0,\ldots) = 1$$

Since we computed the full conditionals, we will use these distributions to sample draws by implementing the Gibbs sampler. The chain converges after a burn-in period, and the sample is considered to be a sample from the posterior distribution.

4.2 A Hierarchical Normal Model

We assume having m similar processes (production lines) that generate products which share a certain qualitative characteristic (valve diameter). The i - th process generates n_i objects. We set y_{ij} to be the measurement on the i -th group and on the j - th object of the group. We assume that the observations are normally distributed according to the following model:

$$y_{ij}|\mu_i, \tau \overset{iid}{\sim} N(\mu_i, \tau^{-1}),\ 1 \leq i \leq m,\ 1 \leq j \leq n_i$$

We also assume that the mean vector $\mu_i = (\mu_1, \ldots, \mu_m)$ is normally distributed, thus:

$$\mu_i|\mu, \nu \overset{iid}{\sim} N(\mu, \nu^{-1}),\ 1 \leq i \leq m$$

We consider the parameters τ, μ, ν unknown. We are going to estimate the parameter vector

$$\theta = (((\mu_1, \ldots, \mu_m), \tau), (\mu, \nu)) \in \Theta = (\mathbb{R}^m \times \mathbb{R}^+) \times (\mathbb{R} \times \mathbb{R}^+)$$

where $(\mu_1, \ldots, \mu_m)$ is the latent mean vector. Variables $(\mu_1, \ldots, \mu_m)$ and τ are the mixture components of the hierarchical model. The model illustrated, is a first order hierarchical model.

We assume independent a-priori distributions for variables τ, μ, ν:

$$\begin{aligned}
\tau &\sim Ga(c, d) \\
\mu &\sim N(a, b^{-1}) \\
\nu &\sim Ga(e, f)
\end{aligned}$$

where a, b, c, d, e, f are hyperparameters. By setting $\theta = ((\mu_1, \ldots, \mu_m), \tau)$, $\theta_p = (\mu, \nu)$ and $y = \{y_{ij},\ 1 \leq i \leq m,\ 1 \leq j \leq n_i\}$ we get:

$$\begin{aligned}
y|\theta &\sim \pi(\cdot|\theta) \\
\theta|\theta_p &\sim \pi(\cdot|\theta_p) \\
\theta_p &\sim \pi(\cdot)
\end{aligned}$$

The posterior distribution of the hierarchical model is:

$$\pi(\theta|y) = \pi(\theta, \theta_p|y) \overset{\theta,\theta_p}{\propto} \pi(\theta, \theta_p, y) = \pi(\theta, \theta_p)\, \pi(y|\theta, \theta_p) = \pi(\theta_p)\, \pi(\theta|\theta_p)\, \pi(y|\theta, \theta_p)$$

The prior components are given by:

$$\pi(\theta_p) = \pi(\mu, \nu) = \pi(\mu)\, \pi(\nu)$$

$$\begin{aligned}
\pi(\theta|\theta_p) &= \pi((\mu_1, \ldots, \mu_m), \tau|\mu, \nu) \\
&= \pi(\tau|\mu, \nu)\, \pi(\mu_1, \ldots, \mu_m|\tau, \mu, \nu) \\
&= \pi(\tau)\, \pi(\mu_1, \ldots, \mu_m|\mu, \nu) = \pi(\tau) \prod_{i=1}^{m} \pi(\mu_i|\mu, \nu)
\end{aligned}$$

The component of the likelihood is given by:

$$\begin{aligned}\pi(y|\theta,\theta_p) = \pi(y|\theta) &= \pi(\{y_{ij},\ 1\le i\le m,\ 1\le j\le n_i\}|(\mu_i),\tau)\\ &= \prod_{i=1}^{m}\pi(\{y_{ij},\ 1\le j\le n_i\}|\mu_i,\tau)\\ &= \prod_{i=1}^{m}\prod_{j=1}^{n_i}\pi(y_{ij}|\mu_i,\tau)\end{aligned}$$

Therefore the posterior distribution yields:

$$\pi((\mu_i),\tau,\mu,\nu|(y_{ij})) \propto \pi(\mu)\,\pi(\nu)\times\pi(\tau)\prod_{i=1}^{m}\pi(\mu_i|\mu,\nu)\times\prod_{m=1}^{m}\prod_{j=1}^{n_i}\pi(y_{ij}|\mu_i,\tau)$$

We will calculate some of the quantities that appear in the posterior distribution, so that we can come to a conclusion on whether the posterior is a known distribution or whether it yields a nonstandard form.

$$\pi(y_{ij}|\mu_i,\tau) = N(y_{ij}|\mu_i,\tau^{-1}) = \frac{\sqrt{\tau}}{2\pi}\exp\{-\frac{\tau}{2}(y_{ij}-\mu_i)^2\} \propto \tau^{1/2}\exp\{-\frac{\tau}{2}(y_{ij}-\mu_i)^2\}$$

thus

$$\prod_{j=1}^{n_i}\pi(y_{ij}|\mu_i,\tau) \propto \prod_{j=1}^{n_i}\tau^{1/2}\exp\{-\frac{\tau}{2}(y_{ij}-\mu_i)^2\} = \tau^{n_i/2}\exp\{-\frac{\tau}{2}\sum_{j=1}^{n_i}(y_{ij}-\mu_i)^2\}$$

We set:

- $y_{i.} = \dfrac{1}{n_i}\displaystyle\sum_{j=1}^{n_i} y_{ij}$, the sample mean of the i - th group of observations
- $(n_i-1)\,S_i^2 = \displaystyle\sum_{j=1}^{n_i}(y_{ij}-y_{i.})^2$, the sample variance of the i - th group of observations

$$\begin{aligned}\sum_{j=1}^{n_i}(y_{ij}-\mu_i)^2 &= \sum_{j=1}^{n_i}((y_{ij}-y_{i.})-(\mu_i-y_{i.}))^2\\ &= \sum_{j=1}^{n_i}(y_{ij}-y_{i.})^2 - 2\sum_{j=1}^{n_i}(y_{ij}-y_{i.})(\mu_i-y_{i.}) + \sum_{j=1}^{n_i}(\mu_i-y_{i.})^2\\ &= \sum_{j=1}^{n_i}(y_{ij}-y_{i.})^2 - 2(\mu_i-y_{i.})\sum_{j=1}^{n_i}(y_{ij}-y_{i.}) + n_i(\mu_i-y_{i.})^2\\ &= \sum_{j=1}^{n_i}(y_{ij}-y_{i.})^2 - 2(\mu_i-y_{i.})(n_iy_{i.}-n_iy_{i.}) + n_i(\mu_i-y_{i.})^2\\ &= \sum_{j=1}^{n_i}(y_{ij}-y_{i.})^2 + n_i(\mu_i-y_{i.})^2 = (n_i-1)\,S_i^2 + n_i(\mu_i-y_{i.})^2\end{aligned}$$

Substituting that on $\displaystyle\prod_{j=1}^{n_i}\pi(y_{ij}|\mu_i,\tau)$, yields

$$\prod_{j=1}^{n_i} \pi(y_{ij}|\mu_i,\tau) \propto \tau^{n_i/2} \exp\{-\frac{\tau}{2}\left[(n_i-1)\,S_i^2 + n_i(\mu_i - y_{i.})^2\right]\}$$

By setting $n \stackrel{\triangle}{=} \sum_{i=1}^{m} n_i$, the likelihood becomes:

$$\begin{aligned}
\pi(y|\theta) &= \prod_{m=1}^{m}\prod_{j=1}^{n_i} \pi(y_{ij}|\mu_i,\tau) \\
&\propto \prod_{m=1}^{m} \tau^{n_i/2} \exp\{-\frac{\tau}{2}\left[(n_i-1)\,S_i^2 + n_i(\mu_i - y_{i.})^2\right]\} \\
&= \tau^{\sum n_i/2} \exp\{-\frac{\tau}{2}\sum_{m=1}^{m}\left[(n_i-1)\,S_i^2 + n_i(\mu_i - y_{i.})^2\right]\} \\
&= \tau^{n/2} \exp\{-\frac{\tau}{2}\sum_{m=1}^{m}\left[(n_i-1)\,S_i^2 + n_i(\mu_i - y_{i.})^2\right]\}
\end{aligned}$$

The prior components become:

$$\begin{aligned}
\pi(\theta_p) &= \pi(\mu)\,\pi(\nu) = N(\mu|a,b^{-1}) \times Ga(\nu|e,f) \\
&= \frac{\sqrt{b}}{2\pi}\exp\{-\frac{b}{2}(\mu-a)^2\} \times \frac{f^e}{\Gamma(e)}\nu^{e-1}\exp\{-f\nu\} \\
&\propto \nu^{e-1}\exp\{-\frac{b}{2}(\mu-a)^2 - f\nu\}
\end{aligned}$$

$$\begin{aligned}
\pi(\theta|\theta_p) &= \pi(\tau)\prod_{i=1}^{m}\pi(\mu_i|\mu,\nu) = Ga(\tau|c,d)\prod_{i=1}^{m} N(\mu_i|\mu,\nu^{-1}) \\
&= \frac{d^e}{\Gamma(d)}\tau^{c-1}\exp\{-d\tau\}\prod_{i=1}^{m}\frac{\sqrt{\nu}}{2\pi}\exp\{-\frac{\nu}{2}(\mu_i-\mu)^2\} \\
&\propto \tau^{c-1}\exp\{-d\tau\}\prod_{i=1}^{m}\nu^{1/2}\exp\{-\frac{\nu}{2}(\mu_i-\mu)^2\} \\
&= \tau^{c-1}\,\nu^{m/2}\exp\{-d\tau - \frac{\nu}{2}\sum_{i=1}^{m}(\mu_i-\mu)^2\}
\end{aligned}$$

By substituting the computed expressions for likelihood and prior components, we get the posterior distribution of the hierarchical model:

$$\begin{aligned}
\pi(\theta|y) = \pi((\mu_i), \tau, \mu, \nu|(y_{ij})) &\propto \{\pi(\mu)\,\pi(\nu)\}\{\pi(\tau)\prod_{i=1}^{m}\pi(\mu_i|\mu,\nu)\}\{\prod_{m=1}^{m}\prod_{j=1}^{n_i}\pi(y_{ij}|\mu_i,\tau)\} \\
&\propto \nu^{e-1}\exp\left\{-\frac{b}{2}(\mu-a)^2 - f\nu\right\} \times \tau^{c-1}\,\nu^{m/2}\exp\left\{-d\tau - \frac{\nu}{2}\sum_{i=1}^{m}(\mu_i-\mu)^2\right\} \\
&\times \tau^{n/2}\exp\left\{-\frac{\tau}{2}\sum_{m=1}^{m}\left[(n_i-1)\,S_i^2 + n_i(\mu_i - y_{i\cdot})^2\right]\right\} \\
&= \tau^{n/2+c-1}\,\nu^{m/2+e-1} \\
&\times \exp\left\{-\frac{1}{2}\left[b(\mu-a)^2 + 2f\nu + 2d\tau + \sum_{i=1}^{m}(\nu(\mu_i-\mu)^2 + \tau(n_i-1)\,S_i^2 + \tau n_i(\mu_i - y_{i\cdot})^2)\right]\right\}
\end{aligned}$$

which is the kernel of a nonstandard, $m+3,\ m \geq 2$ dimensional distribution, with support $\Theta = (\mathbb{R}^m \times \mathbb{R}^+) \times (\mathbb{R} \times \mathbb{R}^+)$.

In order to implement the Gibbs sampler, we need to compute the full conditionals (FCs) of the posterior distribution. The transition kernel of the Gibbs sampler consists of these FCs. The draws from all fully conditional posterior distributions are the ones that constitute the sample from the joint distribution via the Gibbs sampling algorithm. Practically, we will compute the distribution for each of the parameters $(\mu_i), \tau, \mu, \nu$, conditional on all the remaining parameters and observations, for instance the FC of μ is given by $\pi(\mu|(\mu_i), \tau, \nu, (y_{ij}))$.

Full Conditional Distribution of μ

$$\begin{aligned}
\pi(\mu|\cdots) \propto \pi((\mu_i), \tau, \mu, \nu|(y_{ij})) &\overset{\mu}{\propto} \exp\left\{-\frac{1}{2}\left[b(\mu-a)^2 + \nu\sum_{i=1}^{m}(\mu_i-\mu)^2\right]\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left[b\mu^2 - 2ab\mu + ba^2 + \nu\sum_{i=1}^{m}(\mu_i^2 - 2\mu\mu_i + \mu^2)\right]\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left[b\mu^2 - 2ab\mu + ba^2 + \nu(-2\mu m\mu_{\cdot} + m\mu^2)\right]\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left[(b+\nu m)\mu^2 - 2(ab + m\mu_{\cdot})\mu\right]\right\} \\
&\propto \exp\left\{-\frac{b+\nu m}{2}\left[\mu - (\frac{ab+m\mu_{\cdot}}{b+\nu m})\right]^2\right\} \\
&\propto N\left(\mu|\frac{ab+m\mu_{\cdot}}{b+\nu m}, (b+\nu m)^{-1}\right), \quad \mu_{\cdot} = \frac{1}{m}\sum_{i=1}^{m}\mu_i
\end{aligned}$$

Full Conditional Distribution of τ

$$\begin{aligned}
\pi(\tau|\cdots) \propto \pi((\mu_i), \tau, \mu, \nu|(y_{ij})) &\overset{\tau}{\propto} \tau^{n/2+c-1}\exp\left\{-\frac{\tau}{2}\left[2d + \sum_{i=1}^{m}((n_i-1)\,S_i^2 + n_i(\mu_i - y_{i\cdot})^2)\right]\right\} \\
&\propto Ga\left(\tau|n/2 + c, d + \frac{1}{2}\sum_{i=1}^{m}\left[(n_i-1)\,S_i^2 + n_i(\mu_i - y_{i\cdot})^2\right]\right)
\end{aligned}$$

Full Conditional Distribution of ν

$$\begin{aligned}\pi(\nu|\cdots) &\propto \pi((\mu_i),\tau,\mu,\nu|(y_{ij})) \overset{\nu}{\propto} \nu^{m/2+e-1}\exp\left\{-\frac{\nu}{2}\left[2f+\sum_{i=1}^{m}(\mu_i-\mu)^2\right]\right\}\\ &\propto Ga\left(\nu|m/2+e, f+\frac{1}{2}\sum_{i=1}^{m}(\mu_i-\mu)^2\right)\end{aligned}$$

Full Conditional Distribution of $\mu_i,\ \ i=1,\ldots,m$

$$\begin{aligned}\pi(\mu_i|\cdots) &\propto \pi((\mu_i),\tau,\mu,\nu|(y_{ij})) \overset{\mu_i}{\propto} \exp\left\{-\frac{1}{2}\left[\nu(\mu_i-\mu)^2+\tau n_i(mu_i-y_{i.})^2\right]\right\}\\ &\propto \exp\left\{-\frac{1}{2}\left[\mu_i^2(\nu+\tau n_i)-2\mu_i(\tau n_i y_{i.}+\nu\mu)\right]\right\}\\ &\propto \exp\left\{-\frac{\nu+\tau n_i}{2}\left[(\mu_i-(\frac{\tau n_i y_{i.}+\nu\mu}{\nu+\tau n_i}))^2\right]\right\}\\ &\propto N\left(\mu_i|\frac{\tau n_i y_{i.}+\nu\mu}{\nu+\tau n_i},(\nu+\tau n_i)^{-1}\right),\ \ 1\le i\le m\end{aligned}$$

So this is a Gibbs sampler in $m+3$ dimensions, which is also the dimension of the Markov chain

$$G=\{(\mu_i^{(k)}),\tau^{(k)},\mu^{(k)},\nu^{(k)}\}_{k\in\mathbb{N}_0}$$

Before Implementing the Gibbs Sampler

1. Initialize hyperparameters $(a,b),(c,d),(e,f)$, by putting to use our a-priori knowledge. If there isn't any, we should use a non-informative (flat) prior.
2. Compute the data summaries $n, y_{i.}, S_i^2$ for given $m, n_i, (y_{ij})$
3. Initialize the sampler

In order to accomplish step 3, we observe the dependence of the posterior full conditionals on the parameters.

$$\begin{aligned}\pi(\mu|\ldots) &= \pi(\mu|(\mu_i),\nu)\\ \pi(\tau|\ldots) &= \pi(\tau|(\mu_i))\\ \pi(\nu|\ldots) &= \pi(\nu|(\mu_i),\mu)\\ \pi(\mu_i|\ldots) &= \pi(\mu_i|\nu,\mu,\tau)\end{aligned}$$

The first iteration of the Gibbs sampler will be:

$$\begin{aligned}[\mu^{(1)}|\ldots] &\sim \pi(\cdot|(\mu_i^{(0)}),\nu^{(0)})\\ [\tau^{(1)}|\ldots] &\sim \pi(\cdot|(\mu_i^{(0)}))\\ [\nu^{(1)}|\ldots] &\sim \pi(\cdot|(\mu_i^{(0)}),\mu^{(1)})\\ [\mu_i^{(1)}|\ldots] &\sim \pi(\cdot|\nu^{(1)},\mu^{(1)},\tau^{(1)}),\ \ 1\le i\le m\end{aligned}$$

which means that we need to set initial values for $(\mu_i^{(0)}),\nu^{(0)}$. There are two ways in which we can achieve that:

1. Considering that the hyperparameters are constant, we simulate values from the parameters' prior distributions

$$\begin{aligned}
&\nu^{(0)} \sim Ga(\cdot|e,f)\\
&\mu^{(0)} \sim N(\cdot|a,b^{-1})\\
&[(\mu_i^{(0)})|\mu^{(0)},\nu^{(0)}] \overset{iid}{\sim} N(\cdot|\mu^{(0)},(\nu^{(0)})^{-1}),\ \ 1\le i\le m
\end{aligned}$$

2. Assigning on μ,ν, the values of the prior means

$$\begin{aligned}
&\nu^{(0)} = e/f\\
&\mu^{(0)} = a\\
&[(\mu_i^{(0)})|\mu^{(0)},\nu^{(0)}] \overset{iid}{\sim} N(\cdot|\mu^{(0)},(\nu^{(0)})^{-1}),\ \ 1\le i\le m
\end{aligned}$$

Running the Gibbs Sampler - Monitor Convergence

Subsequently, we run the sampler for a large number of iterations N, and as a result, we get draws from the invariant joint posterior distribution. We sample values from the sequence

$$\{(\mu_i^{(k)}),\tau^{(k)},\mu^{(k)},\nu^{(k)}\}_{k=1}^{\mathbb{N}}$$

through the recursive type

$$\begin{aligned}
&\mu^{(k)} \sim \pi(\cdot|(\mu_i^{(k-1)}),\nu^{(k-1)})\\
&\tau^{(k)} \sim \pi(\cdot|(\mu_i^{(k-1)}))\\
&\nu^{(k)} \sim \pi(\cdot|(\mu_i^{(k-1)}),\mu^{(k)})\\
&\mu_i^{(k)} \sim \pi(\cdot|\nu^{(k)},\mu^{(k)},\tau^{(k)}),\ \ 1\le i\le m
\end{aligned}$$

In some cases the model is complex enough, to make it hard to decide on the burn in period that is proper in order to practically obtain a sample of the stationary joint posterior distribution. Most of the times it is "safe" to discard one third of the chain points (burn in period), while sometimes it is preferable to run the sampler multiple times by altering the starting conditions.

4.2.1 Simulation

We simulate data from the normal distribution

$$y_{ij}|\mu_i,\tau \overset{iid}{\sim} N(\mu_i,\tau^{-1}),\ \ 1\le i\le m,\ 1\le j\le n_i$$

for $m=6$ and $n_i=1000$. We want to estimate $\mu,\tau,\nu,\{\mu_i\}_{i=1}^6$. We run the Gibbs sampler for 25000 iterations and 5000 draws are discarded as burn-in period. For all parameters θ, we gain estimators of the form

$$\frac{1}{N-N_{\mathrm{b}}}\sum_{i=N_{\mathrm{b}}+1}^{N}\theta_i \longrightarrow E[\theta|x], \text{ namely the ergodic average.}$$

where N_{b} is the burn-in period. We also compute the standard error of the mean (which captures simulation error of the mean rather than posterior uncertainty)

$$SE = \frac{posterior\,SD}{\sqrt{N-N_{\mathrm{b}}}}.$$

Variable	Mean	SD	SE
μ	3.5	0.97	0.007
τ	0.96	0.02	0.0001
ν	0.28	0.18	0.001
μ_1	0.99	0.03	0.0002
μ_2	1.98	0.03	0.0002
μ_3	3.02	0.03	0.0002
μ_4	4.02	0.03	0.0002
μ_5	4.98	0.03	0.0002
μ_6	5.99	0.03	0.0002

We check convergence and mixing of the chains graphically:

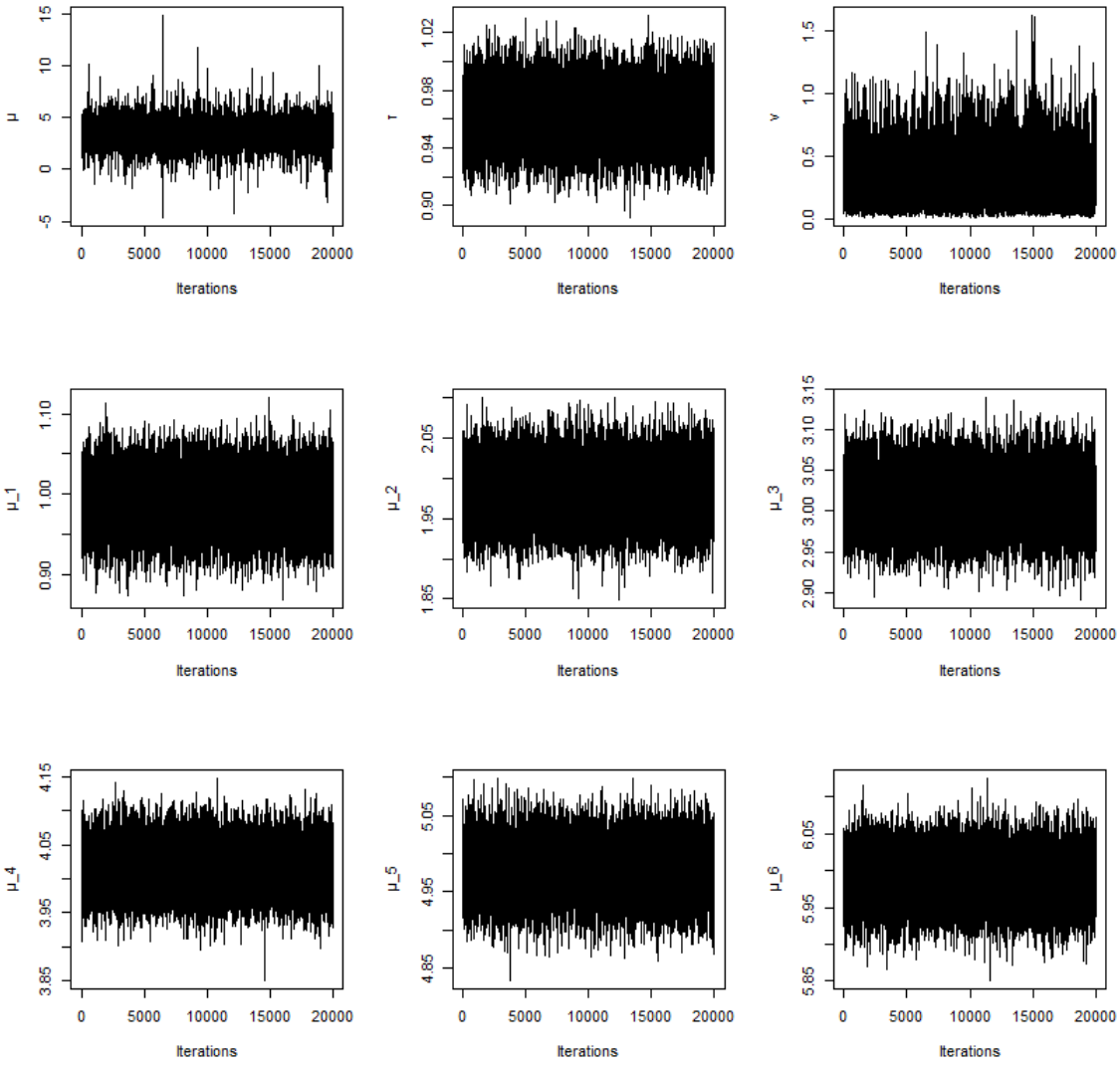

Figure 4.1: Traceplots

The chains for all variables come of good mixing.

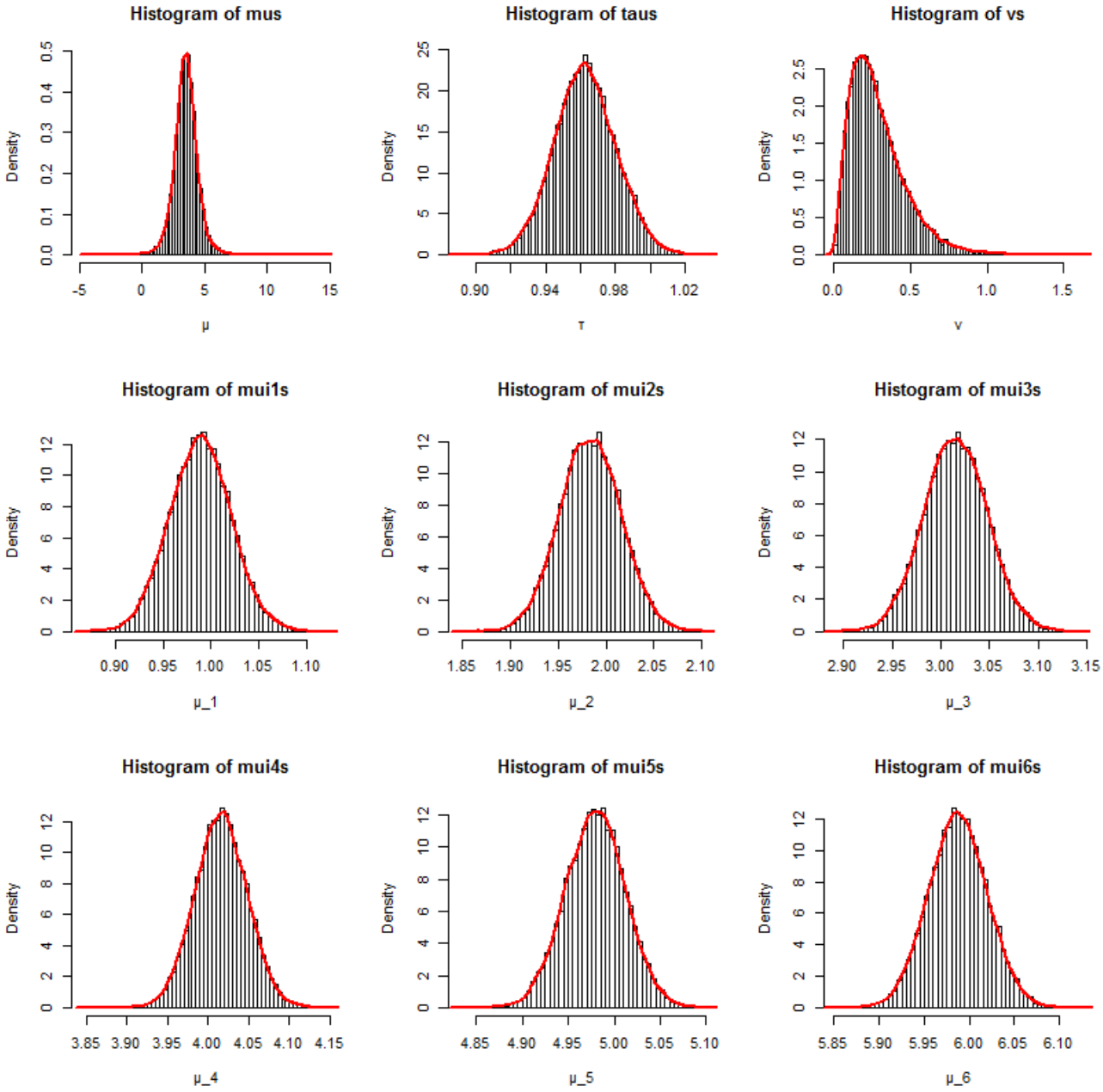

Figure 4.2: Histograms

Densities of all variables are well approximated by the posterior draws.

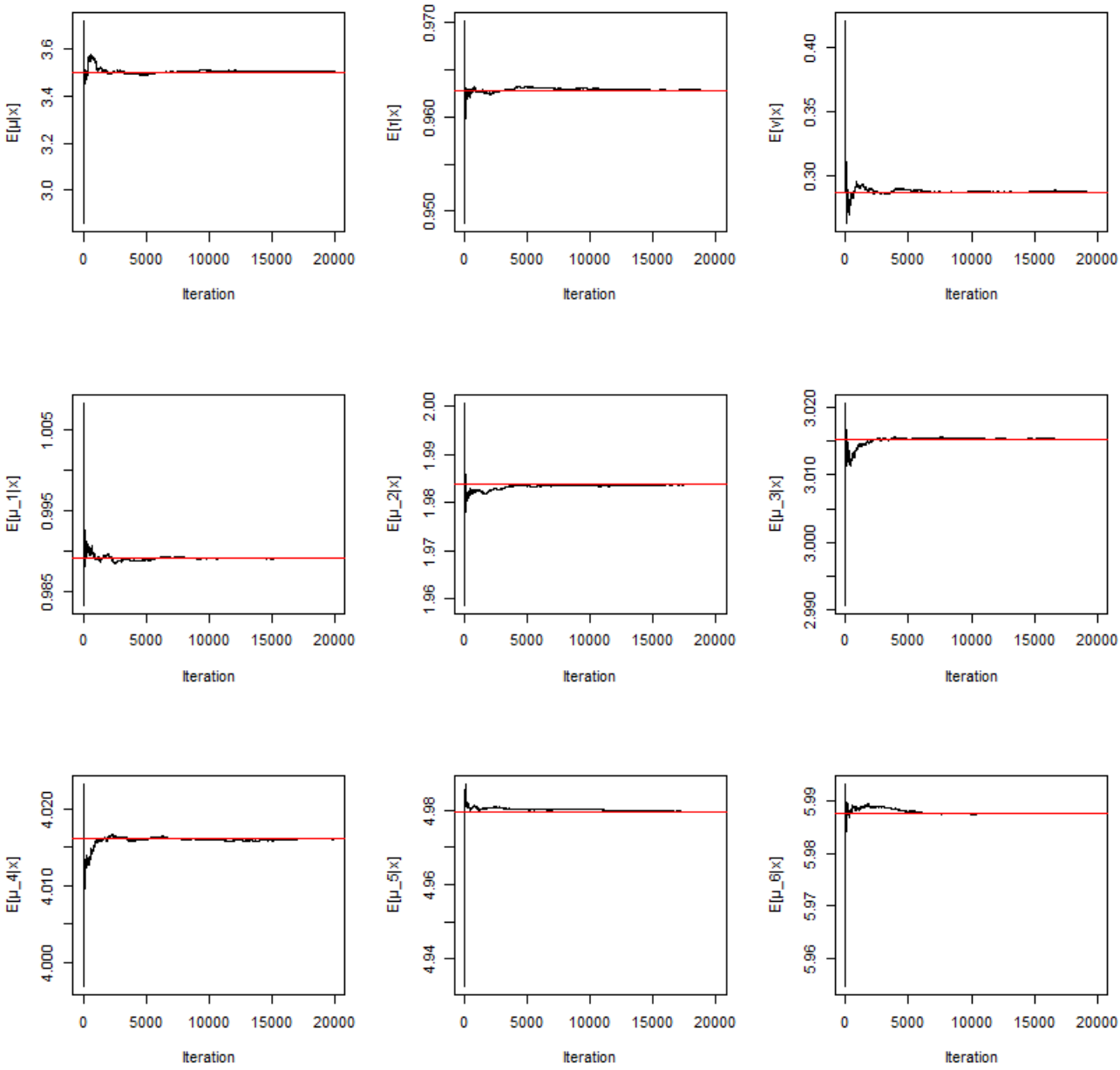

Figure 4.3: Running average plots

Running averages for all variables indicate satisfactory convergence to the ergodic means (red line).

Now, we monitor convergence with the use of some diagnostic tests.

	Geweke	Heidelberg and Welch					
Variable	Z-score	Stationarity test	Start	P-value	Halfwidth	Mean	Halfwidth test
μ	-0.04	✓	1	0.67	0.014	3.5	✓
τ	-0.56	✓	1	0.12	0.001	1	✓
ν	0.69	✓	1	0.57	0.003	0.3	✓
μ_1	0.55	✓	1	0.98	0.001	1	✓
μ_2	-2.34	✓	2001	0.18	0.001	2	✓
μ_3	0.03	✓	1	0.49	0.001	3	✓
μ_4	0.22	✓	1	0.54	0.001	4	✓
μ_5	2.25	✓	2001	0.11	0.001	5	✓
μ_6	1.85	✓	1	0.36	0.001	6	✓

As we see in the diagnostic results, all variables satisfy the null hypotheses of both tests.

For the quantile $q = 0.025$, within an accuracy of $r + / - 0.005$ with probability $s = 0.95$, the Raferty and Lewis diagnostic yields:

	Raferty and Lewis			
Variable	Burn-in	Required SS	Minimum SS	Dependence Factor
μ	3	4267	3746	1.14
τ	2	3787	3746	1.01
ν	4	4732	3746	1.26
μ_1	2	3787	3746	1.01
μ_2	2	3665	3746	0.98
μ_3	2	3665	3740	0.99
μ_4	2	3665	3635	0.97
μ_5	2	3710	3746	0.99
μ_6	2	3710	3746	0.99

Chapter 5

Reversible Jump MCMC

When it comes to implementing MCMC algorithms, we frequently face challenges regarding parameter inference and model selection. A more general scheme than Metropolis-Hastings introduced by Green (1995), the reversible jump algorithm, can deal more efficiently with such challenges.

The reversible jump algorithm provides the potentiality to simulate a sample from a posterior distribution on space of varying dimension; consider the case when the number of parameters is not fixed, and can be treated as a random variable. Reversible jump can also be used when there is a number of competing models that have the same number of parameters. Note however, that in simpler problems other approaches may be preferable, due to the algorithm's high computational cost. We are going to present in detail the derivation of the acceptance probability of the reversible jump algorithm which practically, is a similar procedure as for the Metropolis-Hastings acceptance probability.

The Stationary Distribution

Assume that (M, Z) represents a state of the Markov chain, containing two components; $M \in \{1, 2, \ldots, I\}$ is a model indicator and Z is a real stochastic vector, possibly of varying dimension. The differentiation between the models may be due to their different parametric form rather than a different number of parameters. The stochastic vector Z is defined in the set $C = \cup_{m=1}^{I} C_m$, $C_m = \mathbb{R}^{n_m}$, $n_m > 1$. Assume now that the joint probability distribution of (M, Z) is π. Given $M = m$, Z can only take values in C_m, so that π is specified by the probability $p_m = P(M = m)$ and densities $f(\cdot \mid M = m)$ on C_m, for $m = 1, 2, \ldots, I$. The joint probability distribution of (M, Z) for $A_m \subseteq C_m$ is

$$\begin{aligned} P(M = m, Z \in A_m) &= P(M = m)\ P(Z \in A_m \mid M = m) \\ &= p_m \int_{A_m} f(z \mid M = m)\ dz. \end{aligned}$$

A representation of the **joint posterior distribution** for each state (M, Z), and some given data y is the following

$$p_m\, f_m = c^{-1}\, \tilde{p}_m\, h(z|m)\, l(y|m, z)$$

$\boldsymbol{p_m}$: the posterior probability of model m
$\boldsymbol{f_m}$: the posterior density of vector Z of parameters associated with model m
$\boldsymbol{c^{-1}}$: normalizing constant (typically unknown)

$$c = \sum_{m=1}^{I} \tilde{p}_m \int_{C_m} h(z|m)\, l(y|m, z)\, dz$$

$\tilde{\boldsymbol{p}}_{\boldsymbol{m}}$: the prior probability of model m
$\boldsymbol{h(z|m)}$: the prior density of Z given $M = m$
$\boldsymbol{l(y|m, z)}$: the likelihood of the data y given $(M, Z) = (m, z)$.

The Proposal Density

In the Metropolis-Hastings scheme depending on whether the parameter updating is simultaneous or single-site, there are two different kinds of proposal densities generated. In the first case, the candidate point Y_{n+1} for the new state X_{n+1} is generated from the d-dimensional proposal density $q(x, \cdot)$. In the second case, the candidate point $Y_{n+1} = (x_1, x_2, \ldots, x_{i-1}, Z_i, x_{i+1}, \ldots, x_d)$ is generated by drawing the random variable Z_i from the one-dimensional proposal density $q_i(x, \cdot)$.

In the reversible jump scheme, each $X_i = (M_i, Z_i)$ represents a state of the chain. Assume that the Markov chain is currently on the state $X_n = (m, z)$ and a move to the state $X_{n+1} = (m', z')$ is considered next. Below we present the procedure of generating a proposal state for the chain's next move.

Let $Y_{n+1} = \left(Y_{n+1}^{ind}, Y_{n+1}^{par}\right)$, where ind, par represent the proposal for the model indicator M_{n+1} and for the vector Z_{n+1} respectively, be the proposal state for X_{n+1}.

Jumps using the rjMCMC algorithm

1. Set the proposal $Y_{n+1}^{ind} = m'$ with probability $p_{mm'}$ (this probability is user-defined, and satisfies the condition $\left(\sum_{m'=1}^{I} pmm' = 1\right)$)

2. The proposal Y_{n+1}^{par} is generated in $C_{m'}$ given that $Y_{n+1}^{ind} = m'$, then $Y_{n+1}^{par} = z' = g_{1mm'}(z, U)$, where $g_{1mm'}$ is a deterministic mapping applied to the previous value z and to a random component U

3. Draw $U \sim q_{mm'}(z, \cdot)$

4. The proposal Y_{n+1} is accepted with probability $\alpha_{mm'}\left(z, Y_{n+1}^{par}\right) = \alpha_{mm'}(z, g_{1mm'}(z, U)) = \alpha_{mm'}(z, z')$

5.1 The Dimension Matching Condition

In the context of reversible jump, since the Markov chain is allowed to move between spaces of varying dimension, it must be ensured that the current state and the proposed state have matching dimensions for the jump to make sense.

Consider a move from the state (m, z) to the state (m', z') and back, that is the opposite move from the state (m', z') to the state (m, z)

$$(m, z) \longmapsto (m', z') = (m', g_{1mm'}(z, u))$$
$$(m', z') \longmapsto (m, z) = (m, g_{1m'm}(z', u'))$$

where $m, m' \in \{1, 2, \ldots, I\}$, $z \in A_m \subseteq C_m = \mathbb{R}^{n_m}$, $z' \in B_{m'} \subseteq C_{m'} = \mathbb{R}^{n_{m'}}$ and $u \in \mathbb{R}^{n_{mm'}}, u' \in \mathbb{R}^{n_{m'm}}$. Vectors (z, u) and (z', u') must be of equal dimension. The dimension matching condition is:

$$n_m + n_{mm'} = n_{m'} + n_{m'm}.$$

Mapping Properties

To construct the proposal parameter vector Y_{n+1}^{par}, we need to apply a mapping to the vector (z, u). This mapping can be either deterministic or in some cases we prefer the identity mapping. Recall that in the single-site updating Metropolis-Hastings scheme, the candidate point $Y_{n+1} = (x_1, x_2, \ldots, x_{i-1}, Z_i, x_{i+1}, \ldots, x_d)$ is generated by drawing the random variable Z_i from the one-dimensional proposal density $q_i(x, \cdot)$, which with some abuse of notation, can be written

$$Y_{n+1} = g\left(x_1, x_2, \ldots, x_{i-1}, Z_i, x_{i+1}, \ldots, x_d\right)$$

where the function g is the identity mapping.

A deterministic mapping $g_{mm'}$ should satisfy the following properties

1. $g_{mm'}$ is differentiable
2. $g_{mm'}$ is one-to-one with $g_{m'm}$
3. The one-to-one mapping exists only if the dimension matching condition holds.

Given that a deterministic mapping has the properties mentioned above, the following transformations are feasible

$$(z', u') = g_{mm'}(z, u) = (g_{1mm'}(z, u), g_{2mm'}(z, u))$$

$$(z, u) = g_{mm'}^{-1}(z', u') = g_{m'm}(z', u') = \left(g_{1m'm}(z', u'), g_{2m'm}(z', u')\right).$$

5.2 Deriving the acceptance probability

Suppose that $X_n = (M_n, Z_n,) \sim \pi$. The reversibility condition is

$$\begin{aligned} &P\left(M_n = m, Z_n \in A_m, M_{n+1} = m', Z_{n+1} \in B_{m'}\right) \\ &= P\left(M_n = m', Z_n \in B_{m'}, M_{n+1} = m, Z_{n+1} \in A_m\right) \end{aligned}$$

where $m, m' \in \{1, 2, \ldots, I\}$, $A_m \subseteq C_m$ and $B_{m'} \subseteq C_{m'}$.

$$\begin{aligned} &P\left(M_n = m, Z_n \in A_m, M_{n+1} = m', Z_{n+1} \in B_{m'}\right) \\ &= \int_{A_m} P\left(M_{n+1} = m', Z_{n+1} \in B_{m'} \mid X_n = (m, z)\right) p_m \, f_m(z) \, dz. \end{aligned}$$

The **transition kernel** is $P\left(M_{n+1} = m', Z_{n+1} \in B_{m'} | X_n = (m, z)\right)$.

Given that the current state of the Markov chain is $X_n = (m, z)$, the probability of generating and accepting the proposal $Y_{n+1} = \left(Y_{n+1}^{ind}, Y_{n+1}^{par}\right) = \left(m', Y_{n+1}^{par} \in B_{m'}\right)$ is:

$$Q_{mm'}^{a}\left(z, B_{m'}\right) = P\left(Y_{n+1}^{ind} = m', Y_{n+1}^{par} \in B_{m'} \text{ and } Y_{n+1} \text{ is accepted} | X_n = (m, z)\right)$$

and the probability of rejecting the proposal is

$$r_m(z) = P\left(Y_{n+1} \text{ is rejected} \mid X_n = (m, z)\right).$$

Thus

$$P\left(M_{n+1}=m', Z_{n+1}\in B_{m'} \mid X_n=(m,z)\right) = Q^a_{mm'}\left(z, B_{m'}\right) + r_m(z)\, I\left(m=m', z\in B_{m'}\right)$$

which by substituting, yields:

$$\begin{aligned}
&P\left(M_n=m, Z_n\in A_m, M_{n+1}=m', Z_{n+1}\in B_{m'}\right)\\
&= p_m \int\limits_{A_m} Q^a_{mm'}\left(z, B_{m'}\right)\, f_m(z)\, dz + p_m \int\limits_{A_m} r_m(z)\, I\left(m=m', z\in B_{m'}\right)\, f_m(z)\, dz\\
&= p_m \int\limits_{A_m} Q^a_{mm'}\left(z, B_{m'}\right)\, f_m(z)\, dz + p_m \int r_m(z)\, I\left(m=m', z\in A_m\cap B_{m'}\right)\, f_m(z)\, dz
\end{aligned}$$

and by symmetry

$$\begin{aligned}
&P\left(M_n=m', Z_n\in B_{m'}, M_{n+1}=m, Z_{n+1}\in A_m\right)\\
&= p'_m \int\limits_{B_{m'}} Q^a_{m'm}\left(z', A_m\right)\, f'_m(z')\, dz' + p'_m \int r'_m(z')\, I\left(m=m', z'\in B_{m'}\cap A_m\right)\, f_m,(z')\, dz'.
\end{aligned}$$

The second terms are equal $\forall m$. More specifically

. if $m=m'$ the chain does not move to a new state, therefore it stays within the same model and both expressions are identical

. if $m\neq m'$ the indicator function is zero, which means that the second integrals are zero.

A sufficient condition for reversibility to hold

$$p_m \int\limits_{A_m} Q^a_{mm'}\left(z, B_{m'}\right)\, f_m(z)\, dz = p_{m'} \int\limits_{B_{m'}} Q^a_{m'm}\left(z', A_m\right)\, f_{m'}(z')\, dz'.$$

for all m, m'.

We described the procedure of generating a proposal and the probability of accepting that proposal. Taking that into account, and given that the current state of the Markov chain is $X_n=(m,z)$, the joint probability of generating and accepting the proposal $Y_{n+1}=\left(Y^{ind}_{n+1}, Y^{par}_{n+1}\right)=\left(m', Y^{par}_{n+1}\in B_{m'}\right)$ now takes the form

$$Q^a_{mm'}\left(z, B_{m'}\right) = p_{mm'} \int I(z'\in B_{m'})\, \alpha_{mm'}(z,z')\, q_{mm'}(z,u)\, du.$$

Therefore the reversibility condition yields

$$p_m \int\limits_{A_m} Q^a_{mm'}\left(z, B_{m'}\right)\, f_m(z)\, dz \;= p'_m \int\limits_{B_{m'}} Q^a_{m'm}\left(z', A_m\right)\, f_{m'}(z')\, dz'$$

for all m, m'.

$$p_m \int\int I(z \in A_m, z' \in B_{m'})\, p_{mm'}\, \alpha_{mm'}(z, z')\, q_{mm'}(z, u)\, f_m(z)\, dz\, du$$
$$= p_{m'} \int\int I(z' \in B_{m'}, z \in A_m)\, p_{m'm}\, \alpha_{m'm}(z', z)\, q_{m'm}(z', u')\, f_{m'}(z')\, dz'\, du'.$$

In order to ensure that reversibility holds, we need to examine the conditions under which these are equal. For that matter, we need to write the abovementioned equations as functions of the same variables.

$$dz'\, du' = |\det\left(g'_{mm'}(z, u)\right)|\, dz\, du$$

where $|\det\left(g'_{mm'}(z, u)\right)|$ is the absolute value of the Jacobian of the transformation and

$$g'_{mm'}(z, u) = \frac{\partial g_{mm'}(z, u)}{\partial(z, u)} = \begin{bmatrix} \frac{\partial g_{1mm'}(z,u)}{\partial z} & \frac{\partial g_{2mm'}(z,u)}{\partial z} \\ \frac{\partial g_{1mm'}(z,u)}{\partial u} & \frac{\partial g_{2mm'}(z,u)}{\partial u} \end{bmatrix}$$

$$P\left(M_n = m', Z_n \in B_{m'}, M_{n+1} = m, Z_{n+1} \in A_m\right)$$
$$= \int\int I(z' \in B_{m'}, z \in A_m)\, p_{m'm}\, \alpha_{m'm}(z', z)\, q_{m'm}(z', u')\, p_{m'}\, f_{m'}(z')\, |\det\left(g'_{mm'}(z, u)\right)|\, dz\, du.$$

Equality is satisfied if

$$p_{mm'}\, \alpha_{mm'}(z, z')\, q_{mm'}(z, u)\, p_m\, f_m(z)$$
$$= p_{m'm}\, \alpha_{m'm}(z', z)\, q_{m'm}(z', u')\, p_{m'}\, f_{m'}(z')\, |\det\left(g'_{mm'}(z, u)\right)|.$$

By setting $\alpha_{m'm}(z', z) = 1$, the acceptance probability that is derived is the largest possible, subject to the detailed balance condition

$$\alpha_{m'm}(z', z) = 1 \Rightarrow \alpha_{mm'}(z, z') < 1$$

$$\alpha_{mm'}(z, z') = \frac{p_{m'm}\, q_{m'm}(z', u')\, p_{m'}\, f_{m'}(z')}{p_{mm'}\, q_{mm'}(z, u)\, p_m\, f_m(z)} \left|\det \frac{\partial g_{mm'}(z, u)}{\partial(z, u)}\right| < 1$$

therefore

$$\alpha_{mm'}(z, z') = \min\left(1, \frac{p_{m'm}\, q_{m'm}(z', u')\, p_{m'}\, f_{m'}(z')}{p_{mm'}\, q_{mm'}(z, u)\, p_m\, f_m(z)} \left|\det \frac{\partial g_{mm'}(z, u)}{\partial(z, u)}\right|\right)$$

By setting $\alpha_{mm'}(z, z') = 1$, we get the inverse:

$$\alpha_{m'm}(z', z) = \min\left(1, \frac{p_{mm'}\, q_{mm'}(z, u)\, p_m\, f_m(z)}{p_{m'm}\, q_{m'm}(z', u')\, p_{m'}\, f_{m'}(z')} \left|\det \frac{\partial g_{mm'}(z, u)}{\partial(z, u)}\right|^{-1}\right).$$

Note that generally $p_{mm'}\, q_{mm'}(z, u)\, p_m\, f_m(z) > 0$ but $p_{mm'}\, q_{mm'}(z, u)\, p_m\, f_m(z) = 0$ only if $p_m\, f_m(z) = 0$ in the initial state (m, z) of the Markov chain.

5.3 Generating Proposals via an appropriate Mapping

Assume we use a deterministic proposal for a move from the state X_n to the state X_{n+1} and a stochastic proposal for a move in the opposite direction. Let $Y_{n+1} = (m', g_{1mm'}(z))$, then the dimension matching condition equals

$$n_m = n_{m'} + n_{m'm}$$

since there is no random component generated in the move from m to m', which means that the move is deterministic and $n_{mm'} = 0$. The move in the opposite direction however is stochastic, and requires generating a random variable U'. Therefore:

$$(z', u') = g_{mm'}(z) = (g_{1mm'}(z), g_{2mm'}(z))$$
$$(z) = g^{-1}_{mm'}(z', u') = g_{m'm}(z', u') = g_{1m'm}(z', u').$$

Let us assume the case where we use a deterministic proposal for a move from the state X_n to the state X_{n+1} and a stochastic proposal for a move in the opposite direction. Let $Y_{n+1} = (m', g_{1mm'}(z))$, then the dimension matching condition equals

$$n_m = n_{m'} + n_{m'm}$$

since there is no random component generated in the move from m to m', which means that the move is deterministic and $n_{mm'} = 0$. The move in the opposite direction however is stochastic, and requires generating a random variable U'. Therefore:

$$(z', u') = g_{mm'}(z) = (g_{1mm'}(z), g_{2mm'}(z))$$
$$(z) = g^{-1}_{mm'}(z', u') = g_{m'm}(z', u') = g_{1m'm}(z', u').$$

Then the proposal acceptance probability takes the form:

$$Q^a_{mm'}(z, B_{m'}) = p_{mm'}\, I(g_{1mm'}(z) \in B_{m'})\, \alpha_{mm'}(z, g_{1mm'}(z)) = p_{mm'}\, I(z' \in B_{m'})\, \alpha_{mm'}(z, z')$$

and

$$p_m \int_{A_m} Q^a_{mm'}(z, B_{m'})\ f_m(z)\, dz = \int I(z \in A_m, z' \in B_{m'})\, p_{mm'}\, \alpha_{mm'}(z, z')\, p_m\, f_m(z)\, dz.$$

$$dz'\, du' = |\det\left(g'_{mm'}(z)\right)|dz, \qquad \text{where } \ g'_{mm'}(z) = \frac{\partial g_{mm'}(z)}{\partial z}$$

Consequently, the detailed balance equation takes the form:

$$\int I(z \in A_m, z' \in B_{m'})\, p_{mm'}\, \alpha_{mm'}(z, z')\, p_m\, f_m(z)\, dz$$
$$= \int I(z' \in B_{m'}, z \in A_m)\, p_{m'm}\, \alpha_{m'm}(z', z)\, q_{m'm}(z', u')\, p_{m'}\, f_{m'}(z')\, |\det\left(g'_{mm'}(z)\right)|dz, \text{ where } u' = g_{2mm'}(z).$$

The acceptance probability now yields:

$$\alpha_{mm'}(z, z') = \min\left(1, \frac{p_{m'm}\, q_{m'm}(z', u')\, p_{m'}\, f_{m'}(z')}{p_{mm'}\, p_m\, f_m(z)}\ \left|\det\left(g'_{mm'}(z)\right)\right|\right).$$

Identity Mapping (FF Strategy)

Assume that $g_{1mm'}(z, U)$ is the identity mapping, and by setting $U = Z'$ we get

$$Y_{n+1}^{par} = Z'.$$

Variable Z' is generated from the proposal density $q_{mm'}(z, \cdot)$, on $\mathbb{R}^{n_{m'}}$ and may depend on the value z of the current state of the Markov chain. The probability of accepting the proposal becomes:

$$Q_{mm'}^{a}(z, B_{m'}) = p_{mm'} \int I(z' \in B_{m'})\, \alpha_{mm'}(z, z')\, q_{mm'}(z, z')\, dz'.$$

This strategy is known as FF strategy, and yields the acceptance probability:

$$\alpha_{mm'}(z, z') = \min\left(1, \frac{p_{m'm}\, q_{m'm}(z', z)\, p_{m'}\, f_{m'}(z')}{p_{mm'}\, q_{mm'}(z, z')\, p_m\, f_m(z)}\right).$$

Notice that there is no Jacobian term, due to the use of the identity mapping. Also, $q_{m'm}$ is a density on $\mathbb{R}^{n_m}$.

The acceptance rate of the FF strategy, is quite similar to the acceptance rate of the Metropolis-Hastings scheme with an extra term, $p_{m'm}/p_{mm'}$. Another difference between them, is that the proposal densities $q(x, y)$ and $q(y, x)$ are both densities on $\mathbb{R}^d$, where d is the dimension of X, Y. By inspection of the reversible jump with proposals generated via the identity mapping scheme, its relationship with the MH algorithm is more obvious. The FF strategy provides a better understanding of the way rjMCMC generalizes the MH scheme.

5.4 Model selection with reversible jump MCMC

5.4.1 Poisson vs Negative Binomial

When modelling count data that are characterized by overdispersion, often arises the issue of which distribution will have a better fit on the data: Poisson or negative binomial distribution? We use the reversible jump scheme to decide on which is the best fitting model.

Let $Y = (y_1, \ldots, y_n)$ be a vector of observations. Suppose that data Y is distributed independently and identically under a Poisson model with parameter $\lambda > 0$:

$$y_i \overset{iid}{\sim} P(\lambda),\ \ \lambda > 0$$

or alternatively distributed under a negative binomial model with parameters $\lambda > 0,\ \kappa > 0$:

$$y_i \overset{iid}{\sim} NB(\lambda, \kappa),\ \ \lambda > 0,\ \kappa > 0$$

The likelihood for the Poisson data is:

$$P(y_i|\lambda) = \prod_{i=1}^{n} \frac{\lambda^{y_i}}{y_i!} \exp(-\lambda)$$

whereas the likelihood for the negative binomial data is:

$$P(y_i|\lambda, \kappa) = \prod_{i=1}^{n} \frac{\lambda^{y_i}}{y_i!} \frac{\Gamma(1/\kappa + y_i)}{\Gamma(1/\kappa)(\kappa + \lambda)^{y_i}} (1 + \kappa\lambda)^{-1/\kappa}$$

We assume that $\theta_1 = \lambda$ is the state in which data belong to the first model (Poisson, $k = 1$) and $\theta_2 = (\theta_{2,1}, \theta_{2,2}) = (\lambda, \kappa)$ is the state in which the data belong to the second model (negative binomial, $k = 2$). We set the following prior probabilities on the models, assuming there is an equal probability of choosing either model:

$$p(k = 1) = p(k = 2) = \frac{1}{2}$$

and for $\theta_1, \theta_{2,1}, \theta_{2,2}$ we set as priors

$$\begin{aligned} \theta_1, \theta_{2,1} &\sim G\left(a_\lambda, b_\lambda\right) \\ \theta_{2,2} &\sim G\left(a_\kappa, b_\kappa\right) \end{aligned}$$

For $k = 1$, the posterior distribution for the first model is:

$$\begin{aligned} \pi(k = 1, \theta_1|Y) &\propto \frac{1}{2}\, p(\theta_1|k = 1)\, p(Y|\theta_1) \propto \frac{1}{2}\, G(\theta_1|a_\lambda, b_\lambda)\, P(Y|\lambda) \\ &\propto \frac{1}{2}\, \frac{b_\lambda^{a_\lambda}}{\Gamma(a_\lambda)}\, \theta_1^{a_\lambda - 1}\, e^{-b_\lambda \theta_1} \prod_{i=1}^{n} \frac{\lambda^{y_i}}{y_i!}\, e^{-\lambda} \\ &\propto \theta_1^{a_\lambda - 1}\, e^{-b_\lambda \theta_1}\, \lambda^{n\bar{y}}\, e^{-\lambda} \propto \theta_1^{(a_\lambda + n\bar{y}) - 1}\, e^{-\theta_1 (b_\lambda + 1)} \\ &\propto G\left(a_\lambda + n\bar{y}, b_\lambda + 1\right) \end{aligned}$$

where $n\bar{y} = \sum y_i$.

For $k = 2$, the posterior distribution for the second model is:

$$\begin{aligned} \pi(k = 2, \theta_2|Y) &\propto \frac{1}{2}\, p(\theta_{2,1}, \theta_{2,2}|k = 2)\, p(Y|\theta_{2,1}, \theta_{2,2}) \propto \frac{1}{2}\, G(\theta_{2,1}|a_\lambda, b_\lambda)\, G(\theta_{2,2}|a_\kappa, b_\kappa)\, P(Y|\lambda, \kappa) \\ &\propto \frac{1}{2}\, \frac{b_\lambda^{a_\lambda}}{\Gamma(a_\lambda)}\, \theta_{2,1}^{a_\lambda - 1}\, e^{-b_\lambda \theta_{2,1}}\, \frac{1}{2}\, \frac{b_\kappa^{a_\kappa}}{\Gamma(a_\kappa)}\, \theta_{2,2}^{a_\kappa - 1}\, e^{-b_\kappa \theta_{2,2}} \prod_{i=1}^{n} \frac{\lambda^{y_i}}{y_i!}\, \frac{\Gamma(1/\kappa + y_i)}{\Gamma(1/\kappa)(\kappa + \lambda)^{y_i}} (1 + \kappa\lambda)^{-1/\kappa} \\ &\propto \theta_{2,1}^{a_\lambda - 1}\, \theta_{2,2}^{a_\kappa - 1}\, e^{-(b_\lambda \theta_{2,1} + -b_\kappa \theta_{2,2})}\, (1 + \kappa\lambda)^{-n/\kappa} \lambda^{n\bar{y}} \prod_{i=1}^{n} \frac{\Gamma(1/\kappa + y_i)}{\Gamma(1/\kappa)} \end{aligned}$$

Consider the move from model 1 to model 2. Let $x = (1, \theta)$ be the current state of the chain. Since there is no equivalent to the parameter κ in model 1, we generate a random variable u,

$$u \sim N(0, \sigma^2), \;\; \sigma \text{ fixed.}$$

Let $x' = (2, \theta')$, and

$$\theta' = (\theta_1', \theta_2') = h_{12}(\theta, u) = (\theta, \mu \exp(u)), \;\; \mu \text{ fixed.}$$

We compute the Jacobian of the transformation $|\det\left(h_{12}(\theta, u)\right)|$:

$$h_{12}(\theta, u) = \frac{\partial h_{12}(\theta, u)}{\partial(\theta, u)} = \begin{bmatrix} \frac{\partial \theta_1'}{\partial \theta_1} & \frac{\partial \theta_1'}{\partial u} \\ \frac{\partial \theta_2'}{\partial \theta_1} & \frac{\partial \theta_2'}{\partial u} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \mu \exp(u) \end{bmatrix} = \mu \exp(u)$$

Consider now the reverse move, from model 2 to model 1. This move does not require generating a random variable u'. We set:

$$(\theta, u) = h'_{21}(\theta') = (\theta'_1, \log(\theta'_2/\mu))$$

We compute the Jacobian of the transformation $|\det(h'_{21}(\theta'))|$:

$$h'_{21}(\theta'_1, \theta'_2) = \frac{\partial h'_{21}(\theta'_1, \theta'_2)}{\partial(\theta'_1, \theta'_2)} = \begin{bmatrix} \frac{\partial\theta}{\partial\theta'_1} & \frac{\partial\theta}{\partial\theta'_2} \\ \frac{\partial u}{\partial\theta'_1} & \frac{\partial u}{\partial\theta'_2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{\theta'_2} \end{bmatrix} = \frac{1}{\theta'_2}$$

Now we can compute the acceptance probability for the move from model 1 to model 2, $A_{1,2} = \min\{1, a_{12}\}$ where

$$a_{12} = \frac{p(k=2)\,\pi(k=2,\theta_2|Y)}{p(k=1)\,\pi(k=1,\theta_1|Y)\,q(u)} \times |J|$$
$$a_{12} = \frac{\theta_1^{a_\lambda-1}\,e^{-b_\lambda\theta_1}\,\lambda^{n\bar{y}}\,e^{-\lambda} \propto \theta_1 a_\lambda + n\bar{y} - 1\,e^{-\theta_1(b_\lambda+1)}}{\theta_{2,1}^{a_\lambda-1}\,\theta_{2,2}^{a_\kappa-1}\,e^{-(b_\lambda\theta_{2,1}+-b_\kappa\theta_{2,2})}\,(1+\kappa\lambda)^{-n/\kappa}\lambda^{n\bar{y}}\prod_{i=1}^{n}\frac{\Gamma(1/\kappa+y_i)}{\Gamma(1/\kappa)}}\left\{\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left[\frac{-u^2}{2\sigma^2}\right]\right\}^{-1} \times \mu\exp(u)$$

and the acceptance probability for the reverse move, $A_{2,1} = \min\{1, a_{21}\}$ where

$$a_{21} = \frac{p(k=1)\,\pi(k=1,\theta_1|Y)\,q'(\theta'_2)}{p(k=2)\,\pi(k=2,\theta_2|Y)} \times |J|$$
$$a_{21} = \frac{\theta_{2,1}^{a_\lambda-1}\,\theta_{2,2}^{a_\kappa-1}\,e^{-(b_\lambda\theta_{2,1}+-b_\kappa\theta_{2,2})}\,(1+\kappa\lambda)^{-n/\kappa}\lambda^{n\bar{y}}\prod_{i=1}^{n}\frac{\Gamma(1/\kappa+y_i)}{\Gamma(1/\kappa)}}{\theta_1^{a_\lambda-1}\,e^{-b_\lambda\theta_1}\,\lambda^{n\bar{y}}\,e^{-\lambda} \propto \theta_1 a_\lambda + n\bar{y} - 1\,e^{-\theta_1(b_\lambda+1)}}\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left[\frac{-(\log(\theta'_2/\mu))^2}{2\sigma^2}\right] \times \frac{1}{\theta'_2}$$

5.4.2 Comparing differences between two treatments

Consider that data are identically and independently distributed according to model 1:

$$\left(y_j|M=1,t,\sigma^2\right) \overset{iid}{\sim} N(t,\sigma^2),\ \ 1 \le j \le 2n$$

or alternatively according to model 2:

$$\begin{cases} \left(y_{1i}|M=2,t_1,\sigma^2\right) \overset{iid}{\sim} N(t_1,\sigma^2), & 1 \le i \le n \\ \left(y_{2i}|M=2,t_2,\sigma^2\right) \overset{iid}{\sim} N(t_2,\sigma^2), & 1 \le i \le n \end{cases}$$

The problem is discriminating between these two models and for that matter we will use the reversible jump MCMC. It is convenient to introduce the stochastic variable T:

$$T = tI(M=1) + (t_1,t_2)I(M=2)$$

The target distribution takes the form:

$$f(M=i,T,\sigma^2|\underline{y}) = \frac{f(M=i,T,\sigma^2,y)}{f(y)} \propto f(M=i,T,\sigma^2)\,f(\underline{y}|M=i,T,\sigma^2)$$
$$\propto f(M=i)\,f(T,\sigma^2|M=i)\,f(\underline{y}|M=i,T,\sigma^2)$$

where $f(M=i)\,f(T,\sigma^2|M=i)$ is the prior distribution and $f(\underline{y}|M=i,T,\sigma^2)$ is the likelihood of the data.

We assume that the probability of choosing either of the models is equal:

$$P(M=1) = P(M=2) = \frac{1}{2}.$$

We set prior distributions for model parameters:

$$\begin{aligned} t &\sim N(\mu_0, \sigma_0^2) \\ t_1 &\sim N(\mu_1, \sigma_1^2) \\ t_2 &\sim N(\mu_2, \sigma_2^2) \\ \sigma^2 &\sim IG(a_0, b_0) \end{aligned}$$

The posterior distribution for model 1 is:

$$\begin{aligned} f(M=1, T, \sigma^2|\underline{y}) &\propto f(M=1)\, f(t,\sigma^2)\, f(y_1,\ldots,y_{2n}|M=1,t,\sigma^2) \\ &\propto f(M=1)\, f(t)\, f(\sigma^2) \prod_{j=1}^{2n} f(y_j|M=1,t,\sigma^2) \\ &\propto f(M=1)\, N(t|\mu_0,\sigma_0^2)\, IG(\sigma^2|a_0,b_0) \prod_{j=1}^{2n} N(y_i|t,\sigma^2) \\ &\propto \frac{1}{2}\frac{1}{\sqrt{2\pi\sigma_0^2}} e^{-\frac{(t-\mu_0)^2}{2\sigma_0^2}} \frac{b_0^{a_0}}{\Gamma(a_0)} (\sigma^2)^{-a_0-1} e^{-b_0/\sigma^2} \times \prod_{j=1}^{2n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_j-t)^2}{2\sigma^2}} \\ &\propto \frac{1}{\sigma^{2(n+a_0+1)}} \exp\left\{ -\frac{1}{2\sigma^2\sigma_0^2} \left(\sigma^2(t-\mu_0)^2 + 2\sigma_0^2 b_0 + \sigma_0^2 \sum_{j=1}^{2n} (y_j - t)^2 \right) \right\} \end{aligned}$$

The posterior distribution for model 2 is:

$$\begin{aligned} f(M=2, T, \sigma^2|\underline{y}) &\propto f(M=2)\, f(t_1,t_2,\sigma^2)\, f(y_{1j}|M=2,t_1,\sigma^2)\, f(y_{2j}|M=2,t_2,\sigma^2) \\ &\propto f(M=2)\, f(t_1)\, f(t_2)\, f(\sigma^2) \prod_{j=1}^{n} f(y_{1j}|M=2,t_1,\sigma^2) \prod_{j=1}^{n} f(y_{2j}|M=2,t_2,\sigma^2) \\ &\propto f(M=2)\, N(t_1|\mu_1,\sigma_1^2)\, N(t_2|\mu_2,\sigma_2^2)\, IG(\sigma^2|a_0,b_0) \prod_{j=1}^{n} N(y_{1j}|t_1,\sigma^2) \prod_{j=1}^{n} N(y_{2j}|t_2,\sigma^2) \\ &\propto \frac{1}{2}\frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(t_1-\mu_1)^2}{2\sigma_1^2}} \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(t_2-\mu_2)^2}{2\sigma_2^2}} \frac{b_0^{a_0}}{\Gamma(a_0)} (\sigma^2)^{-a_0-1} e^{-b_0/\sigma^2} \\ &\times \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_{1j}-t_1)^2}{2\sigma^2}} \times \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_{2j}-t_2)^2}{2\sigma^2}} \\ &\propto \frac{1}{\sigma^{2(n+a_0+1)}} \exp\left\{ -\frac{1}{2\sigma^2\sigma_1^2\sigma_2^2} \left[\sigma_2^2\sigma^2(t_1-\mu_1)^2 + \sigma_1^2\sigma^2(t_2-\mu_2)^2 + 2\sigma_1^2\sigma_2^2 b_0 \right] \right\} \\ &\times \exp\left\{ -\frac{1}{2\sigma^2} \left[\sigma_1^2\sigma_2^2 \sum_{j=1}^{n} (y_{1j}-t_1)^2 + \sigma_1^2\sigma_2^2 \sum_{j=1}^{n} (y_{2j}-t_2)^2 \right] \right\} \end{aligned}$$

Stochastic proposals in both directions

Let the current state of the chain be (m, z) and a move to state (m', z') is proposed with probability $p_{mm'}$. This probability is user defined subject to $p_{mm'} + p_{m'm} = 1$.

$$(m = 1, z = (t, \sigma^2)) \leftrightarrow (m = 2, z' = (t_1, t_2, \sigma^2)).$$

Due to the dimension matching condition

$$n_m + n_{mm'} = n_{m'} + n_{m'm} \Leftrightarrow 2 + n_{mm'} = 3 + n_{m'm}$$

we need to generate two stochastic variables in the move from m to m' and one stochastic variable in the move from m' to m in order to have stochastic proposals in both directions.

$$\begin{aligned} &n_m = 2 : \text{associated with } t, \sigma^2 \\ &n_{mm'} : \text{associated with } u = (v_1, v_2) \\ &n_{m'} : \text{associated with } t_1, t_2, \sigma^2 \\ &n_{m'm} : \text{associated with } u' = v \end{aligned}$$

The stochastic variable $u' = v$ is generated through the proposal $q(z, \cdot)$

$$q(z, v) = q(v|t, \sigma^2) = N(v|t, \sigma^2)$$

and $u = (v_1, v_2)$ is generated through the proposal $q(z', \cdot)$

$$q(z', v_1, v_2) = q(v_1, v_2|t_1, t_2, \sigma^2) = N(v_1|t_1, \sigma^2)\, N(v_2|t_2, \sigma^2) = N_2\left(\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} | \begin{pmatrix} t_1 \\ t_2 \end{pmatrix}, \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix}\right)$$

The deterministic mapping in this case, yields the transformation:

$$(z', u') = (t_1, t_2, \sigma^2, v) = g_{12}(t, \sigma^2, (v_1, v_2)) = (g_{1(12)}(z, u), g_{2(12)}(z, u))$$

where

$$g_{12}\begin{pmatrix} t \\ \sigma^2 \\ v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} t_1 \\ t_2 \\ \sigma^2 \\ v \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/2 & 1/2 \end{pmatrix} \begin{pmatrix} t \\ \sigma^2 \\ v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} t + v_1 \\ t + v_2 \\ \sigma^2 \\ \frac{v_1 + v_2}{2} \end{pmatrix}$$

thus

$$(z', u') = (t_1, t_2, \sigma^2, v) = (g_{1(12)}(z, u), g_{2(12)}(z, u)) = ((t + v_1, t + v_2, \sigma^2), (\frac{v_1 + v_2}{2})).$$

The absolute value of the Jacobian of the transformation for the move from m to m' is:

$$|\det Jac(T)| = \left|\det\left(\frac{\partial g_{12}(z, u)}{\partial(z, u)}\right)\right| = \left|\det\left(\frac{\partial(t + v_1, t + v_2, \sigma^2, \frac{v_1 + v_2}{2})}{\partial(t, \sigma^2, v_1, v_2)}\right)\right| = \left|\det\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1/2 \\ 0 & 1 & 0 & 1/2 \end{bmatrix}\right| = 1$$

and for the reverse move is:

$$|\det Jac(T^{-1})| = |\det Jac(T)|^{-1} = 1.$$

The acceptance probability from m to m' is $A_{12} = \min\{1, a_{12}\}$ where

$$\begin{aligned}
a_{12}(z,z') = a_{12}((t,\sigma^2),(t_1,t_2,\sigma^2)) &= \frac{p_{21}\, q(z,v)\, c^{-1}\, f(M=2,T,\sigma^2|\underline{y})}{p_{12}\, q(z',v_1,v_2)\, c^{-1}\, f(M=1,T,\sigma^2|\underline{y})} \times |J| \\
&= \frac{p_{21}\, N(v|t,\sigma^2)\, \exp\left\{-\frac{1}{2\sigma^2\sigma_1^2\sigma_2^2}\left[\sigma_2^2\sigma^2(t_1-\mu_1)^2 + \sigma_1^2\sigma^2(t_2-\mu_2)^2 + 2\sigma_1^2\sigma_2^2 b_0\right]\right\}}{p_{12}\, N(v_1|t_1,\sigma^2)\, N(v_2|t_2,\sigma^2)\, \exp\left\{-\frac{1}{2\sigma^2\sigma_0^2}\left(\sigma^2(t-\mu_0)^2 + 2\sigma_0^2 b_0 + \sigma_0^2 \sum_{j=1}^{2n}(y_j - t)^2\right)\right\}} \\
&\times \exp\left\{-\frac{1}{2\sigma^2}\left[\sigma_1^2\sigma_2^2\sum_{j=1}^{n}(y_{1j}-t_1)^2 + \sigma_1^2\sigma_2^2\sum_{j=1}^{n}(y_{2j}-t_2)^2\right]\right\} \\
&= \frac{p_{21}\, \exp\left\{-\frac{1}{2\sigma^2\sigma_1^2\sigma_2^2}\left[\sigma_2^2\sigma^2(t_1-\mu_1)^2 + \sigma_1^2\sigma^2(t_2-\mu_2)^2 + 2\sigma_1^2\sigma_2^2 b_0\right]\right\}}{p_{12}\, \exp\left\{-\frac{1}{2\sigma^2\sigma_0^2}\left(\sigma^2(t-\mu_0)^2 + 2\sigma_0^2 b_0 + \sigma_0^2 \sum_{j=1}^{2n}(y_j - t)^2\right)\right\}} \\
&\times \frac{\sqrt{2\pi\sigma^2}\, \exp\{(v-t)^2/2\sigma^2\}}{\exp\{((v_1-t_1)^2 + (v_2-t_2)^2)/2\sigma^2\}} \times \exp\left\{-\frac{1}{2\sigma^2}\left[\sigma_1^2\sigma_2^2\sum_{j=1}^{n}(y_{1j}-t_1)^2 + \sigma_1^2\sigma_2^2\sum_{j=1}^{n}(y_{2j}-t_2)^2\right]\right\}
\end{aligned}$$

and the acceptance probability for the reverse move, $A_{21} = \min\{1, a_{21}\}$ is the inverse of A_{12}

$$a_{21}(z',z) = a_{12}(z,z')^{-1}$$

Stochastic proposal in one direction - Deterministic in the other

Let us consider the move from $m = 1$ to $m' = 2$. The dimension matching condition now is

$$n_m + n_{mm'} = n_{m'} \Leftrightarrow 2 + n_{mm'} = 3$$

thus, it is required to generate one stochastic variable in the move from m to m'

$$\begin{aligned}
&n_m = 2 : \text{associated with } \ t, \sigma^2 \\
&n_{mm'} = 1 : \text{associated with } \ u \\
&n_{m'} = 3 : \text{associated with } \ t_1, t_2, \sigma^2
\end{aligned}$$

The stochastic variable u is generated through the proposal $q(z,\cdot)$

$$q(z,u) = q(u|t,\sigma^2) = N(u|t,\sigma^2)$$

The mapping in this case is:

$$z' = (t_1, t_2, \sigma^2) = g_{12}(z,u) = g_{1(12)}$$

$$g_{12}\begin{pmatrix} t \\ u \\ \sigma^2 \end{pmatrix} = \begin{pmatrix} t_1 \\ t_2 \\ \sigma^2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} t \\ u \\ \sigma^2 \end{pmatrix} = \begin{pmatrix} t+u \\ t-u \\ \sigma^2 \end{pmatrix}$$

thus

$$z' = (t_1, t_2, \sigma^2) = g_{12}(z, u) = (t + u, t - u, \sigma^2)$$

The absolute value of the Jacobian of the transformation for the move from m to m' is:

$$|\det Jac(T)| = \left|\det\left(\frac{\partial g_{12}(z,u)}{\partial(z,u)}\right)\right| = \left|\det\left(\frac{\partial(t+u, t-u, \sigma^2)}{\partial(t, \sigma^2, u)}\right)\right| = \left|\det\begin{bmatrix}1 & 1 & 0\\ 0 & 0 & 1\\ 1 & -1 & 0\end{bmatrix}\right| = 2.$$

The acceptance probability from m to m' is $A_{12} = \min\{1, a_{12}\}$ where

$$\begin{aligned} a_{12}(z, z') = a_{12}((t, \sigma^2), (t_1, t_2, \sigma^2)) &= \frac{p_{21}\, c^{-1}\, f(M = 2, T, \sigma^2|\underline{y})}{p_{12}\, q(z, u)\, c^{-1}\, f(M = 1, T, \sigma^2|\underline{y})} \times |J| \\ &= \frac{p_{21}\, \exp\left\{-\frac{1}{2\sigma^2\sigma_1^2\sigma_2^2}\left[\sigma_2^2\sigma^2(t_1 - \mu_1)^2 + \sigma_1^2\sigma^2(t_2 - \mu_2)^2 + 2\sigma_1^2\sigma_2^2 b_0\right]\right\}}{p_{12}\, \exp\left\{-\frac{1}{2\sigma^2\sigma_0^2}\left(\sigma^2(t - \mu_0)^2 + 2\sigma_0^2 b_0 + \sigma_0^2 \sum_{j=1}^{2n}(y_j - t)^2\right)\right\}} \\ &\times \frac{2\sqrt{2\pi\sigma^2}}{\exp\{(u - t)^2/2\sigma^2\}} \times \exp\left\{-\frac{1}{2\sigma^2}\left[\sigma_1^2\sigma_2^2 \sum_{j=1}^{n}(y_{1j} - t_1)^2 + \sigma_1^2\sigma_2^2 \sum_{j=1}^{n}(y_{2j} - t_2)^2\right]\right\} \end{aligned}$$

Now the move from $m' = 2$ to $m = 1$ is deterministic. The mapping in this case is:

$$g_{21}\begin{pmatrix}t_1\\ t_2\\ \sigma^2\end{pmatrix} = \begin{pmatrix}t\\ u\\ \sigma^2\end{pmatrix} = \begin{pmatrix}1/2 & 1/2 & 0\\ 1/2 & -1/2 & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}t_1\\ t_2\\ \sigma^2\end{pmatrix} = \begin{pmatrix}\frac{1}{2}(t_1 + t_2)\\ \frac{1}{2}(t_1 - t_2)\\ \sigma^2\end{pmatrix}$$

The absolute value of the Jacobian of the transformation for the move from m' to m is:

$$|\det Jac(T)| = \left|\det\left(\frac{\partial g_{21}(z)}{\partial(z)}\right)\right| = \left|\det\left(\frac{\partial(\frac{1}{2}(t_1 + t_2), \sigma^2, \frac{1}{2}(t_1 - t_2))}{\partial(t_1, t_2, \sigma^2)}\right)\right| = \left|\det\begin{bmatrix}1/2 & 0 & 1/2\\ 1/2 & 0 & -1/2\\ 0 & 1 & 0\end{bmatrix}\right| = \frac{1}{2}$$

which is the inverse as expected. Furthermore, the acceptance probability from m' to m, $A_{21} = \min\{1, a_{21}\}$ is equal to the inverse of $A_{12} = \min\{1, a_{12}\}$:

$$a_{21}(z', z) = \frac{p_{12}\, q(z, u)\, c^{-1}\, f(M = 1, T, \sigma^2|\underline{y})}{p_{21}\, c^{-1}\, f(M = 2, T, \sigma^2|\underline{y})} \times \frac{1}{2}$$

Deterministic proposals in both directions - FF strategy

We assume that σ^2 is fixed, since it is common in both models. Let $(m = 1, z = t)$ and $(m = 2, z' = (t_1, t_2))$. Consider the move from m to m'.

We use as a proposal for the move from m to m' the density $q_{12}(\cdot)$:

$$q(t_1, t_2) = N_2\left(\begin{pmatrix} t \\ t \end{pmatrix}, \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix}\right)$$

and for the move from m' to m the density $q_{21}(\cdot)$:

$$q(t) = N\left(\frac{t_1 + t_2}{2}, \sigma^2\right).$$

The acceptance probability now is $A_{12} = \min\{1, a_{12}\}$, where

$$a_{12}(z, z') = \frac{p_{21}\, q(t)\, c^{-1}\, f(M = 2, T, \sigma^2|\underline{y})}{p_{12}\, q(t_1, t_2)\, c^{-1}\, f(M = 1, T, \sigma^2|\underline{y})}$$

and the acceptance probability of the reverse move, $A_{21} = \min\{1, a_{21}\}$ is the inverse of that.

5.5 Model Choice in Regression

Let us consider the independent random variables $\{Z_i\}$ expressed in the form

$$Z_i = \sum_{k=0}^{m-1} \beta_k\, {a_i}^k + \epsilon_i, \qquad \epsilon_i \sim N(0, 1), \quad i = 1, \cdots, N = 101$$

thus

$$z_i \sim N\left(\sum_{k=0}^{m-1} \beta_k\, {a_i}^k, 1\right).$$

Parameters $a_1, \ldots, a_N$ are known, while parameters $m \in \{1, 2, 3, 4\}$ (model indicator) and $\beta^{(m)} = (\beta_0, \ldots, \beta_{m-1})$ (regression coefficients) are unknown. Note that the data $\mathbf{z} = (z_1, \ldots, z_N)$, for $a_i = \frac{i-1}{20}$, $i = 1, \ldots, 101$ and $m = 4$ are constructed as follows:

$$A = \begin{bmatrix} a_1^0 & a_1^1 & a_1^2 & a_1^3 \\ a_2^0 & a_2^1 & a_2^2 & a_2^3 \\ \vdots & \vdots & \ddots & \vdots \\ a_i^0 & a_i^1 & a_i^2 & a_i^3 \end{bmatrix}$$

thus

$$\mathbf{z} = (z_1, \ldots, z_{101}) = \begin{pmatrix} a_1^0 & a_1^1 & a_1^2 & a_1^3 \\ a_2^0 & a_2^1 & a_2^2 & a_2^3 \\ \vdots & \vdots & \ddots & \vdots \\ a_{101}^0 & a_{101}^1 & a_{101}^2 & a_{101}^3 \end{pmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_{101} \end{bmatrix}$$

The likelihood of the data is:

$$f\left(z|\beta^{(m)}, m\right) = \prod_{i=1}^{N} N\left(z_i|\sum_{k=0}^{m-1} \beta_k\, {a_i}^k, 1\right) \propto \exp\left(-\frac{1}{2}\sum_{i=1}^{N}\left(z_i - \sum_{k=0}^{m-1} \beta_k\, {\alpha_i}^k\right)^2\right)$$

Suppose that the current state of the chain is $(m, x = \beta^{(m)})$ and we propose a move to a next state $(m', y = \beta^{(m')})$.

Assigning uniform priors for both m and β, yields the posterior density

$$f\left(\beta^{(m)}, m|z\right) \propto f(m)\, f(\beta^{(m)}|m)\, f\left(z|\beta^{(m)}, m\right) \propto f\left(z|\beta^{(m)}, m\right) \propto \exp\left(-\frac{1}{2}\sum_{i=1}^{N}\left(z_i - \sum_{k=0}^{m-1}\beta_k\,{\alpha_i}^k\right)^2\right)$$

A move is proposed from the density:

$$q(m', y|x) = q(m'|x)\, q(y|x, m')$$

where typically

$$q(m'|x) = q(m'|m), \quad m = \dim(x).$$

We generate m' and y from the proposals:

$$\begin{aligned} m' &\sim DU(1, \ldots, m) \\ y &\sim N_m(0, I) \end{aligned}$$

which means that q depends on the current dimension only. Note that the proposal from $q_{mm'}$ is a density on $\mathbb{R}^{m'}$ and $q_{m'm}$ is a density on $\mathbb{R}^m$.

The acceptance probability for the move from model m to model m' is $A = \min\{1, a_{mm'}\}$, where

$$\begin{aligned} a_{mm'} &= \frac{f(\beta^{(m')}, m'|z)\, q(m|m')\, q(\beta^{(m)}|\beta^{(m')}, m)}{f\left(\beta^{(m)}, m|z\right)\, q(m'|m)\, q(\beta^{(m')}|\beta^{(m)}, m')} \\ &= \frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{N}\left(z_i - \sum_{k=0}^{m'-1}\beta_k\,{\alpha_i}^k\right)^2\right) \times (2\pi)^{-m/2}\, \exp\left\{-0.5\sum_{k=0}^{m-1}(\beta_k^{(m)2})\right\} \times m'}{\exp\left(-\frac{1}{2}\sum_{i=1}^{N}\left(z_i - \sum_{k=0}^{m-1}\beta_k\,{\alpha_i}^k\right)^2\right) \times (2\pi)^{-m'/2}\, \exp\left\{-0.5\sum_{k=0}^{m'-1}(\beta_k^{(m')2})\right\} \times m} \end{aligned}$$

For the move from m' to m, the acceptance ratio is the inverse of that.

For the same move from $(m, x = \beta^{(m)})$ to $(m', y = \beta^{*(m')})$, consider using stochastic proposals:

- For $m' \leq m$ the dimension matching condition is:

$$n_m = n_{m'} + n_{m'm}, \quad m = m' + (m - m')$$

We generate m', y and u from the proposals:

$$\begin{aligned} m' &\sim DU(1, \ldots, m) \\ y &\sim N_m(0, I) \\ u &\sim N_{m'}(0, I) \end{aligned}$$

The mapping is:

$$\left(\beta^{*(m')}, u^{(m-m')}\right) = g_{mm'}(\beta^{(m)}) = \left(g_{1mm'}(\beta^{(m)}), g_{2mm'}(\beta^{(m)})\right)$$

where $\beta^{(m)} = \left(\beta^{(m')}, \beta^{(m-m')}\right)$,

$$g_{mm'}\begin{bmatrix}\beta^{(m')}\\ \beta^{(m-m')}\end{bmatrix} = \begin{bmatrix}\beta^{*(m')}\\ u^{(m-m')}\end{bmatrix} = \begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}\begin{bmatrix}\beta^{(m')}\\ \beta^{(m-m')}\end{bmatrix} = \begin{bmatrix}\beta^{(m')} + \beta^{(m-m')}\\ \beta^{(m-m')}\end{bmatrix}$$

thus

$$\left(\beta^{*(m')}, u^{(m-m')}\right) = \left(\beta^{(m')} + \beta^{(m-m')}, \beta^{(m-m')}\right)$$

The absolute value of the Jacobian of the transformation for the move from m to m' is:

$$|\det Jac(T)| = \left|\det\left(\frac{\partial g_{mm'}(\beta^{(m')}, \beta^{(m-m')})}{\partial(\beta^{(m')}, \beta^{(m-m')})}\right)\right| = \left|\det\left(\frac{\partial(\beta^{(m')} + \beta^{(m-m')}, \beta^{(m-m')})}{\partial(\beta^{(m')}, \beta^{(m-m')})}\right)\right| = \left|\det\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}\right| = 1.$$

The acceptance probability for the move from m to m' is $A_{mm'} = \min\{1, a_{mm'}\}$, where:

$$\begin{aligned} a_{mm'} &= \frac{f(\beta^{(m')}, m'|z)\, q(m|m')\, q(\beta^{(m)}|\beta^{(m')}, m) \times q(u)}{f\left(\beta^{(m)}, m|z\right)\, q(m'|m)\, q(\beta^{(m')}|\beta^{(m)}, m')} \times |J| \\ &= \frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{N}\left(z_i - \sum_{k=0}^{m'-1}\beta_k\,{\alpha_i}^k\right)^2\right) \times (2\pi)^{-m/2}\, \exp\left\{-0.5\sum_{k=0}^{m-1}({\beta_k^{(m)}}^2)\right\} \times \exp\left\{-0.5\sum_{k=0}^{m'-1}u^2\right\} \times m'}{\exp\left(-\frac{1}{2}\sum_{i=1}^{N}\left(z_i - \sum_{k=0}^{m-1}\beta_k\,{\alpha_i}^k\right)^2\right) \times (2\pi)^{-m'}\, \exp\left\{-0.5\sum_{k=0}^{m'-1}({\beta_k^{(m')}}^2)\right\} \times m} \end{aligned}$$

and for the move from m' to m it is the inverse of that.

- For $m' > m$ the dimension matching condition is:

$$n_m + n_{mm'} = n'_m, \quad m' = m + (m' - m)$$

We generate m', y and v from the proposals:

$$\begin{aligned} m' &\sim DU(1, \ldots, m) \\ y &\sim N_m(0, I) \\ v &\sim N_{m'}(0, I) \end{aligned}$$

The mapping is:

$$\beta^{*(m')} = g_{mm'}(\beta^{(m)}, v) = \left(g_{1mm'}(\beta^{(m)}, v), g_{2mm'}(\beta^{(m)}, v)\right)$$

$$g_{mm'}\begin{bmatrix}\beta^{(m)}\\ v^{(m'-m)}\end{bmatrix}=\begin{bmatrix}\beta^{*(m)}\\ \beta^{*(m'-m)}\end{bmatrix}=\begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}\begin{bmatrix}\beta^{(m)}\\ v^{(m'-m)}\end{bmatrix}=\begin{bmatrix}\beta^{(m)}+v^{(m'-m)}\\ v^{(m'-m)}\end{bmatrix}$$

thus

$$\beta^{*(m')}=\left((\beta^{(m)}+v^{(m'-m)}),v^{(m'-m)}\right)$$

The absolute value of the Jacobian of the transformation for the move from m to m' is:

$$|\det Jac(T)|=\left|\det\left(\frac{\partial g_{mm'}(\beta^{(m)},v^{(m'-m)})}{\partial(\beta^{(m)},v^{(m'-m)})}\right)\right|=\left|\det\left(\frac{\partial(\beta^{(m)}+v^{(m'-m)},v^{(m'-m)})}{\partial(\beta^{(m)},v^{(m'-m)})}\right)\right|=\left|\det\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}\right|=1.$$

The acceptance probability for the move from m to m' is $A_{mm'}=\min\{1,a_{mm'}\}$, where:

$$\begin{aligned}a_{mm'}&=\frac{f(\beta^{(m')},m'|z)\,q(m|m')\,q(\beta^{(m)}|\beta^{(m')},m)}{f\left(\beta^{(m)},m|z\right)\;q(m'|m)\,q(\beta^{(m')}|\beta^{(m)},m')\times q(v)}\times|J|\\&=\frac{\exp\left(-\frac{1}{2}\sum_{i=1}^{N}\left(z_i-\sum_{k=0}^{m'-1}\beta_k\,{\alpha_i}^k\right)^2\right)\times(2\pi)^{-m}\,\exp\left\{-0.5\sum_{k=0}^{m-1}(\beta_k^{(m)^2})\right\}\times m'}{\exp\left(-\frac{1}{2}\sum_{i=1}^{N}\left(z_i-\sum_{k=0}^{m-1}\beta_k\,{\alpha_i}^k\right)^2\right)\times(2\pi)^{-m'/2}\,\exp\left\{-0.5\sum_{k=0}^{m'-1}(\beta_k^{(m')^2})\right\}\times\exp\left\{-0.5\sum_{k=0}^{m-1}v^2\right\}\times m}\end{aligned}$$

and for the move from m' to m it is the inverse of that.

5.5.1 Simulation

For initial values $\beta=(1,0.3,0.15,0.005)$, $m\in\{1,2,3,4\}$, we run the reversible jump sampler for 1000000 iterations:

$\underline{\beta^{(m)}\text{ estimates:}}$
Note: For the stochastic proposals, we run the sampler for two cases:

1.
$$\begin{aligned}z_i&\sim N\left(\sum_{k=0}^{m-1}\beta_k\,{a_i}^k,1\right)\\y&\sim N_m(0,I)\\u&\sim N_{m'}(0,I)\\v&\sim N_{m'}(0,I)\end{aligned}$$

2.
$$\begin{aligned}z_i&\sim N\left(\sum_{k=0}^{m-1}\beta_k\,{a_i}^k,\sigma_z\right)\\y&\sim N_m(0,I*\sigma_y)\\u&\sim N_{m'}(0,I*\sigma_u)\\v&\sim N_{m'}(0,I*\sigma_v)\end{aligned}$$

Beta	FF strategy	Stochastic $\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$	Stochastic $\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$
$\beta^{(1)}$	0.047	0.94	2.62
$\beta^{(2)}$	(0.38, 1.159)	(0.41, 1.14)	(0.38, 1.15)
$\beta^{(3)}$	(1.07, 0.318, 0.17)	(1.07, 0.34, 0.16)	(1.001, 0.39, 0.154)
$\beta^{(4)}$	(1.07, 0.016, 0.34, -0.023)	(1.3, -0.039, 0.37, -0.03)	(0.87, 0.44, 0.17, -0.005)

m estimates (model selecting):

Model probability	FF strategy	Stochastic $\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$	Stochastic $\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$
$p^{(m=1)}$	0.00001	0.00001	0.00001
$p^{(m=2)}$	0.029	0.211	0.354
$p^{(m=3)}$	0.814	0.773	0.419
$p^{(m=4)}$	0.157	0.014	0.227

In all three cases, model $m = 3, \beta^{(m=3)}$ is selected. We set as a thinning interval, thin = 50 and we obtain the following running average plots and histograms. Convergence to the estimated values for $\beta^{(m=3)}$ is fulfilled. Furthermore, convergence of the model selection probabilities is also fulfilled.

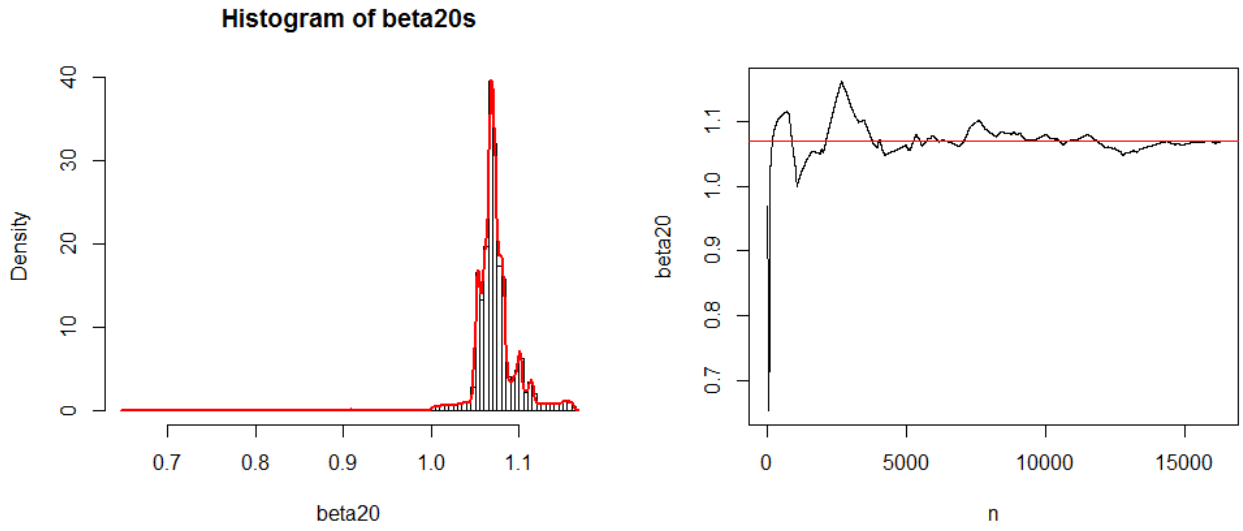

Figure 5.1: FF strategy: ${\beta_0}^{(3)}$

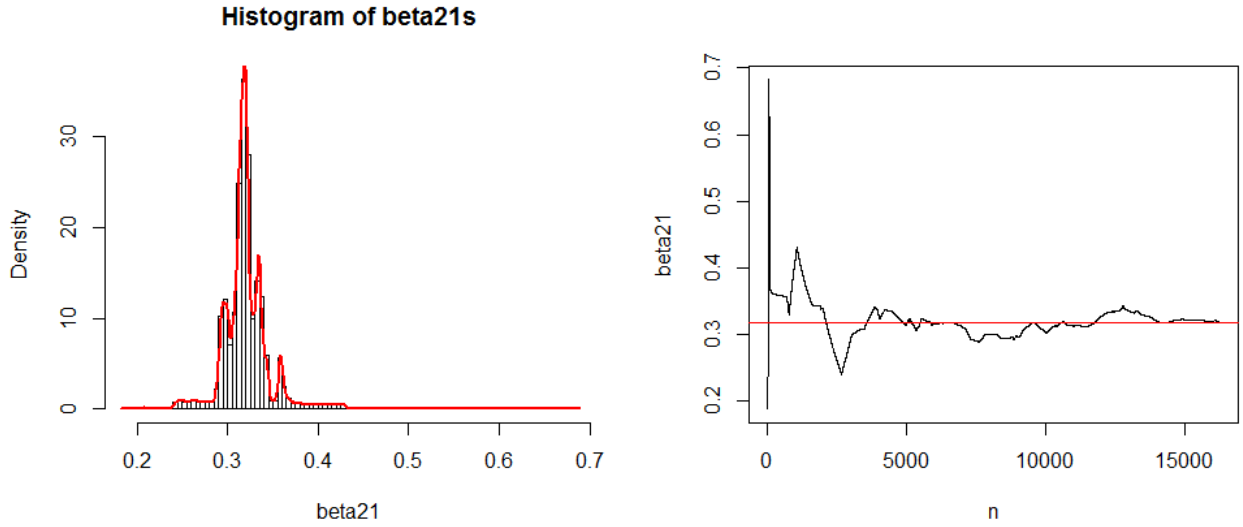

Figure 5.2: FF strategy: ${\beta_1}^{(3)}$

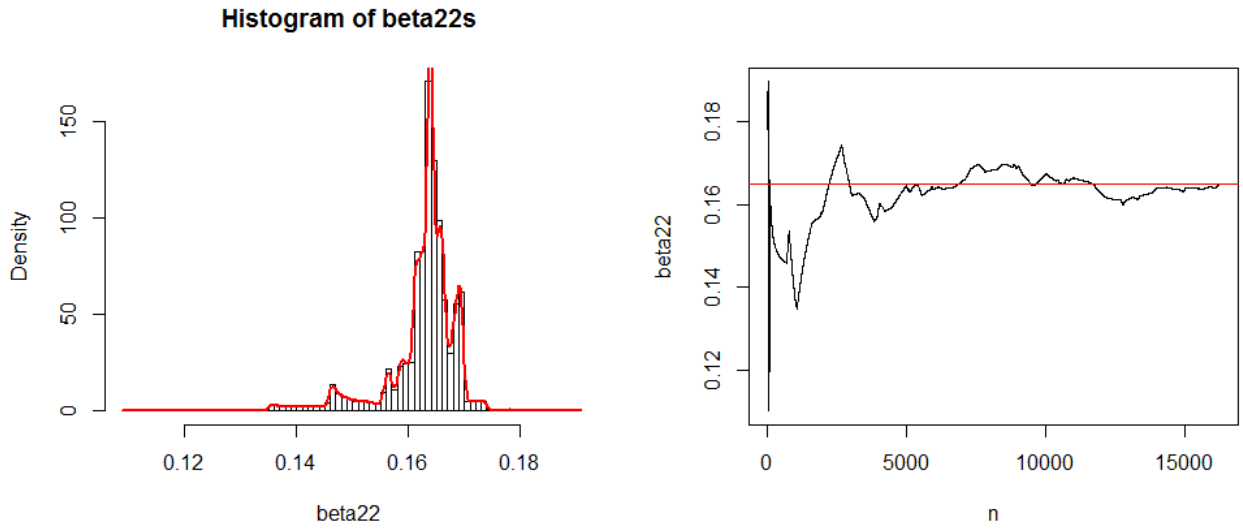

Figure 5.3: FF strategy: $\beta_2{}^{(3)}$

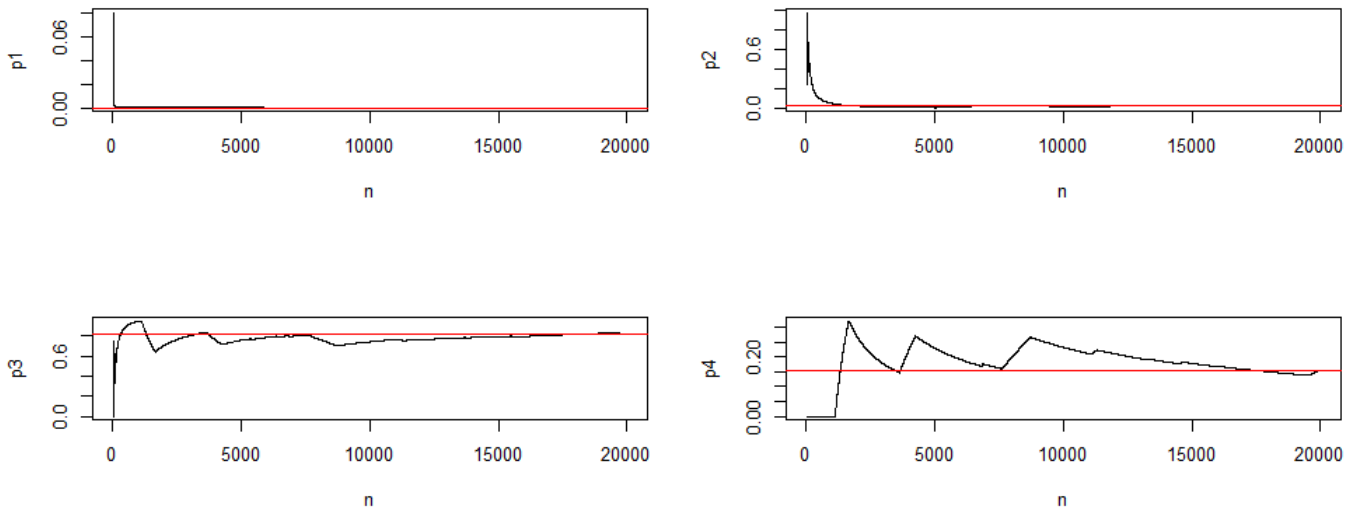

Figure 5.4: FF strategy: Probabilities of selecting each model

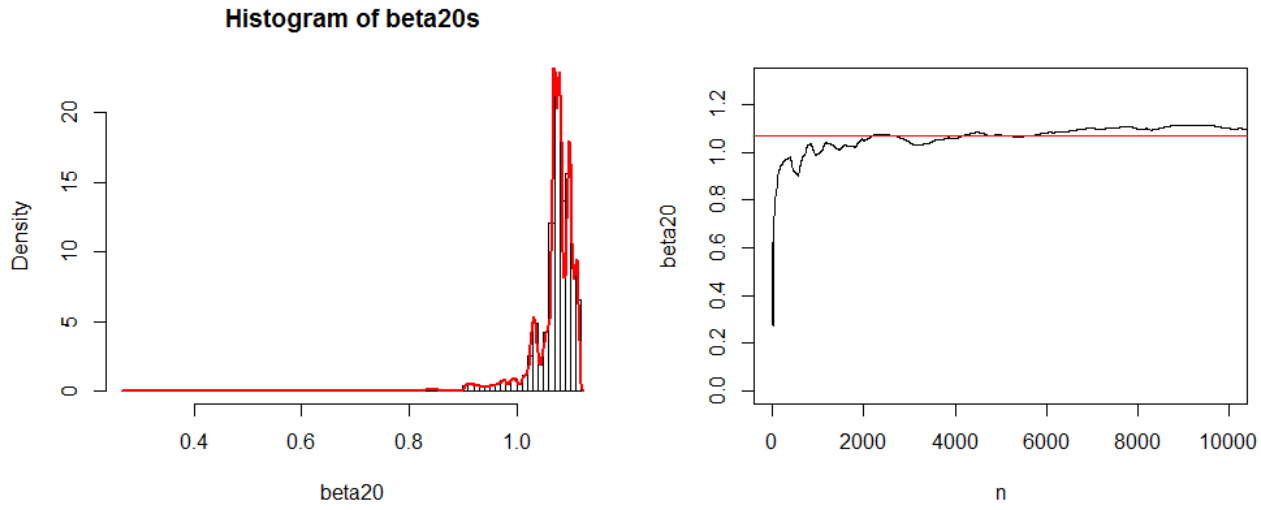

Figure 5.5: Stochastic $\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$: ${\beta_0}^{(3)}$

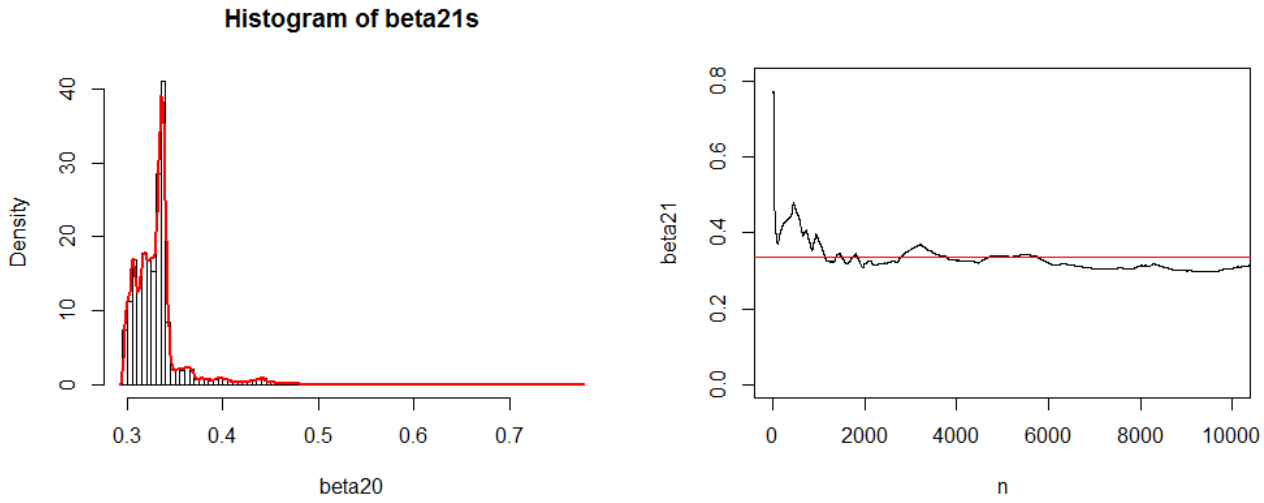

Figure 5.6: Stochastic $\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$: ${\beta_1}^{(3)}$

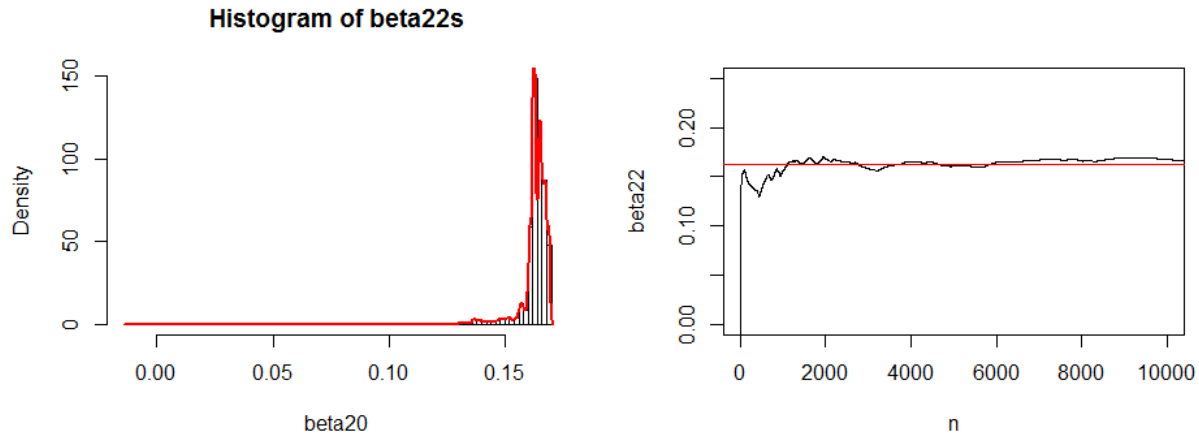

Figure 5.7: Stochastic $\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$: ${\beta_2}^{(3)}$

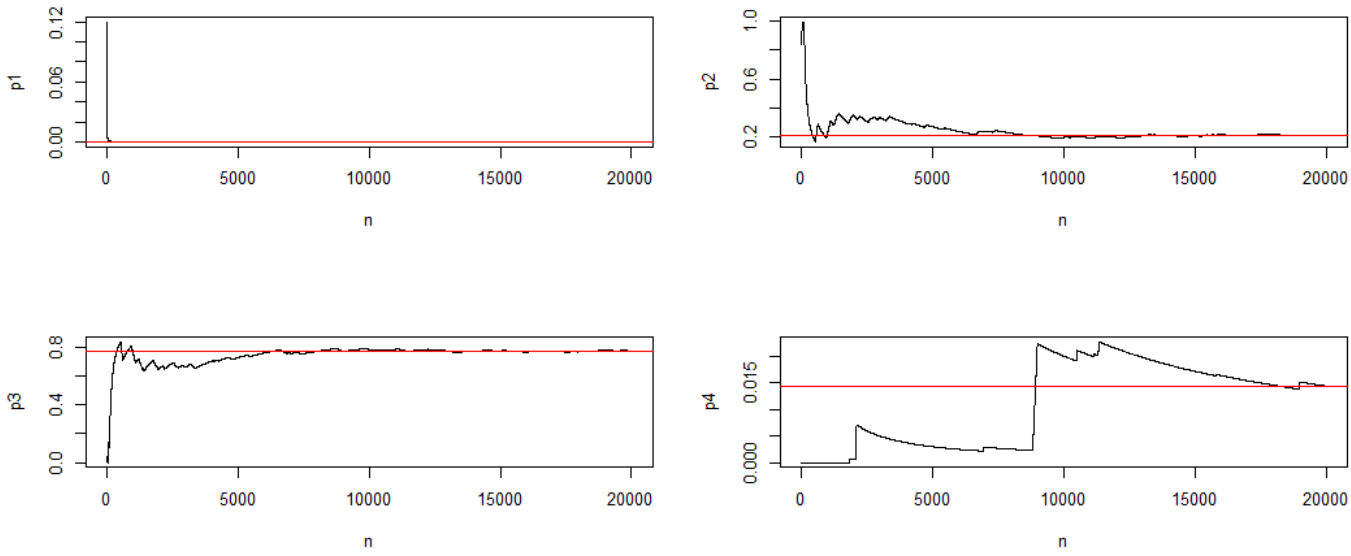

Figure 5.8: Stochastic $\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$: Probabilities of selecting each model

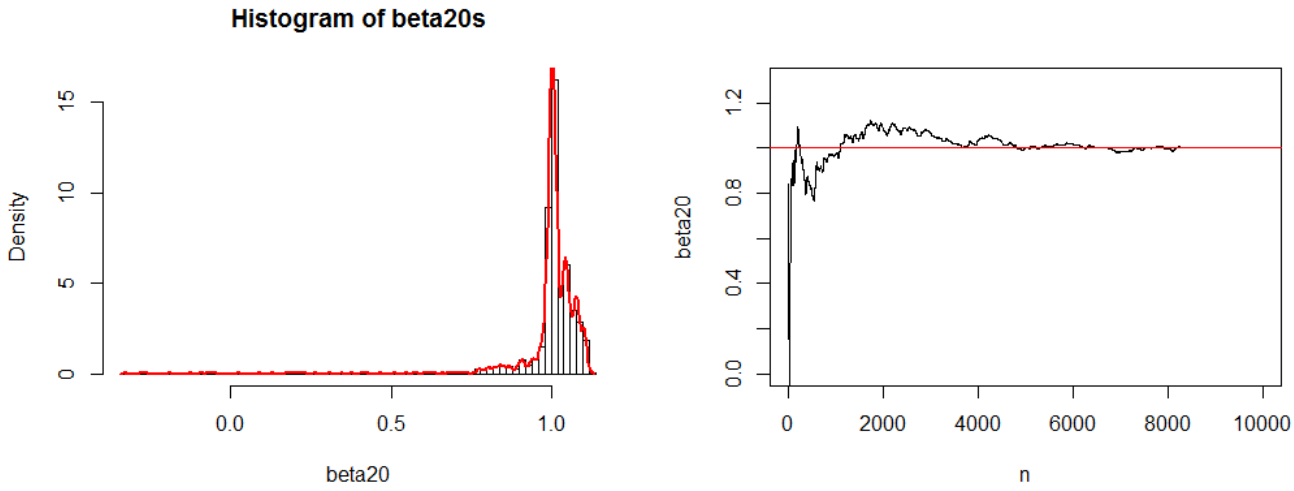

Figure 5.9: Stochastic $\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$: ${\beta_0}^{(3)}$

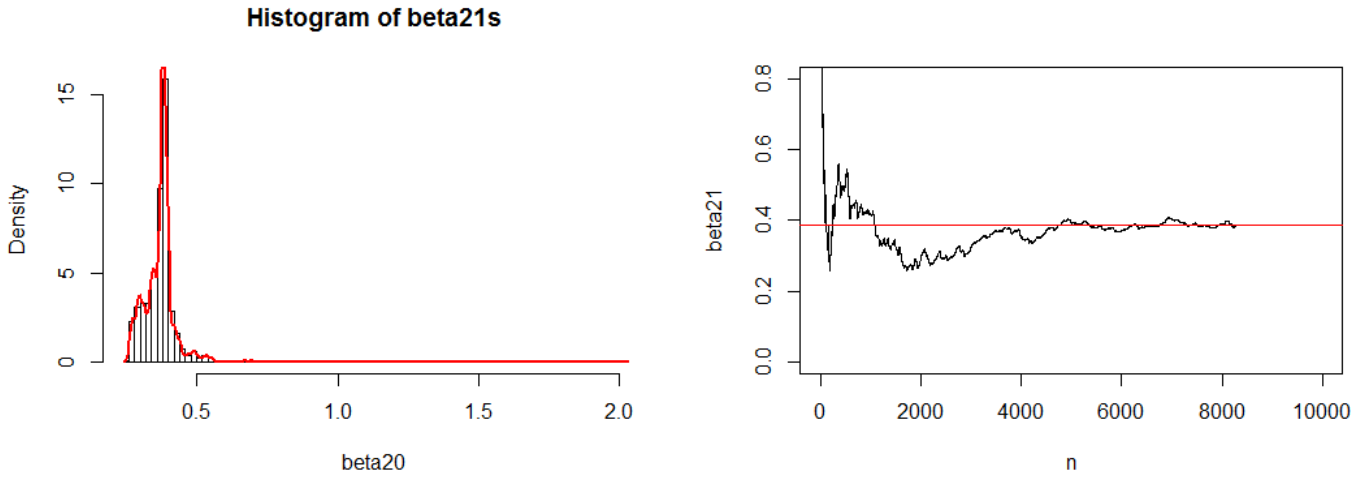

Figure 5.10: Stochastic $\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$: ${\beta_1}^{(3)}$

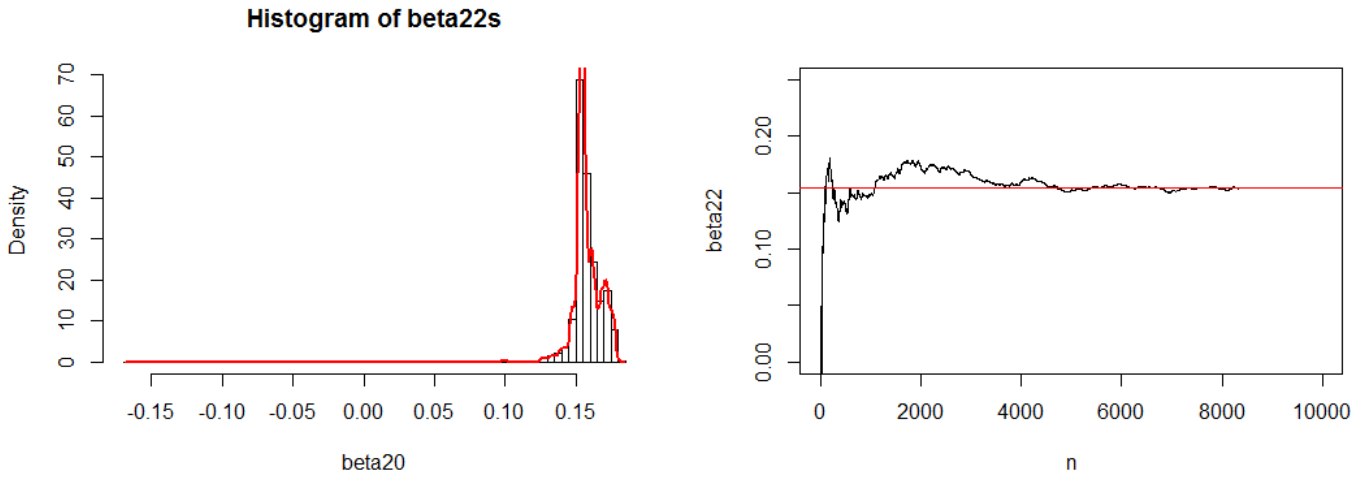

Figure 5.11: Stochastic $\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$: ${\beta_2}^{(3)}$

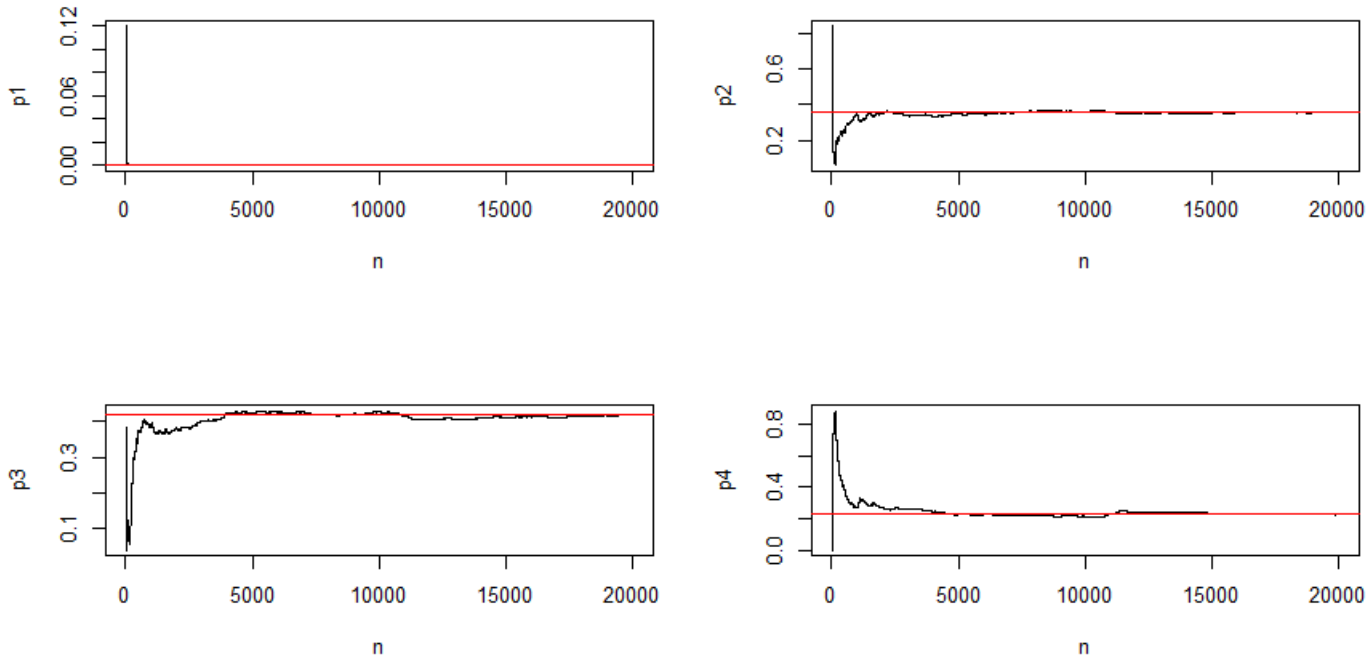

Figure 5.12: Stochastic $\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$: Probabilities of selecting each model

We notice that by decreasing the proposal variance (third case), the probabilities of selecting a model converge faster.

We run some further diagnostics for model 3. We get some idea on the computational efficiency of each method:

Method	Iterations	SS of the chain	Thinning interval
FF strategy	814051	16282	50
Stochastic $\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$	774101	15483	50
Stochastic $\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$	417401	8349	50

The performance of each method for model 3, can be monitored with some further diagnostic tests.

	Heidelberg and Welch					
FF strategy	Stationarity test	Start (iter)	P-value	Halfwidth	Mean	Halfwidth test
${\beta_0}^{(3)}$	✓	1	0.6	0.01	1.07	✓
${\beta_1}^{(3)}$	✓	1	0.7	0.01	0.32	✓
${\beta_2}^{(3)}$	✓	1	0.18	0.003	0.16	✓
$\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$	Stationarity test	Start (iter)	P-value	Halfwidth	Mean	Halfwidth test
${\beta_0}^{(3)}$	✓	1550	0.6	0.03	1.08	✓
${\beta_1}^{(3)}$	✓	1	0.4	0.01	0.33	✓
${\beta_2}^{(3)}$	✓	1	0.6	0.003	0.16	✓
$\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$	Stationarity test	Start (iter)	P-value	Halfwidth	Mean	Halfwidth test
${\beta_0}^{(3)}$	✓	3341	0.18	0.014	1.01	✓
${\beta_1}^{(3)}$	✓	3341	0.21	0.013	0.38	✓
${\beta_2}^{(3)}$	✓	3341	0.23	0.003	0.16	✓

As we see in the diagnostic results, all methods satisfy the null hypotheses of both tests.

For the quantile $q = 0.025$, within an accuracy of $r + / - 0.005$ with probability $s = 0.95$, the Raferty and Lewis diagnostic yields:

	Raferty and Lewis			
FF strategy	Burn-in	Required SS	Minimum SS	Dependence Factor
${\beta_0}^{(3)}$	69300	38745200	3746	10300
${\beta_1}^{(3)}$	69300	38745200	3746	10300
${\beta_2}^{(3)}$	68300	74546350	3746	19900
$\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$	Burn-in	Required SS	Minimum SS	Dependence Factor
${\beta_0}^{(3)}$	65900	36843350	3746	9840
${\beta_1}^{(3)}$	64950	70886850	3746	18900
${\beta_2}^{(3)}$	65900	36843350	3746	9840
$\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$	Burn-in	Required SS	Minimum SS	Dependence Factor
${\beta_0}^{(3)}$	13950	12276200	3746	3280
${\beta_1}^{(3)}$	17400	18936500	3746	5060
${\beta_2}^{(3)}$	17500	14477550	3746	3860

It is clear that the chains 'suffer' from high autocorrelations, which is confirmed by the ACF plots. The required sample size needed to achieve low correlations between draws and/or independence from the influence of starting values is enormous and computationally costly.

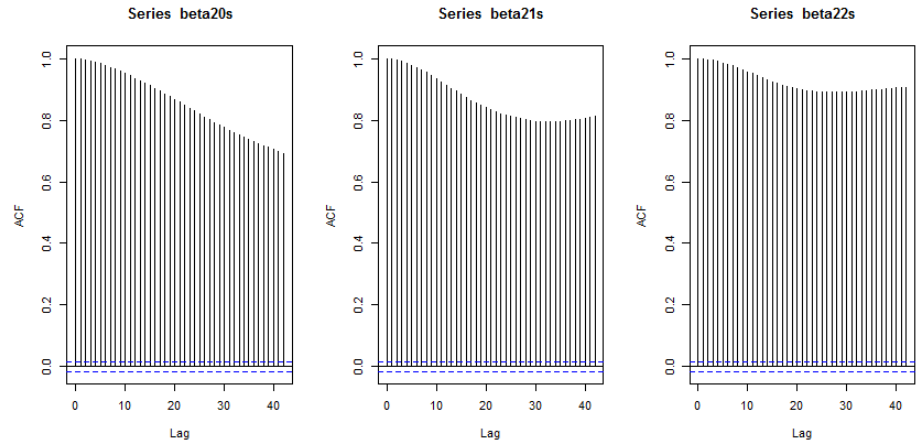

Figure 5.13: ACF plots for FF strategy.

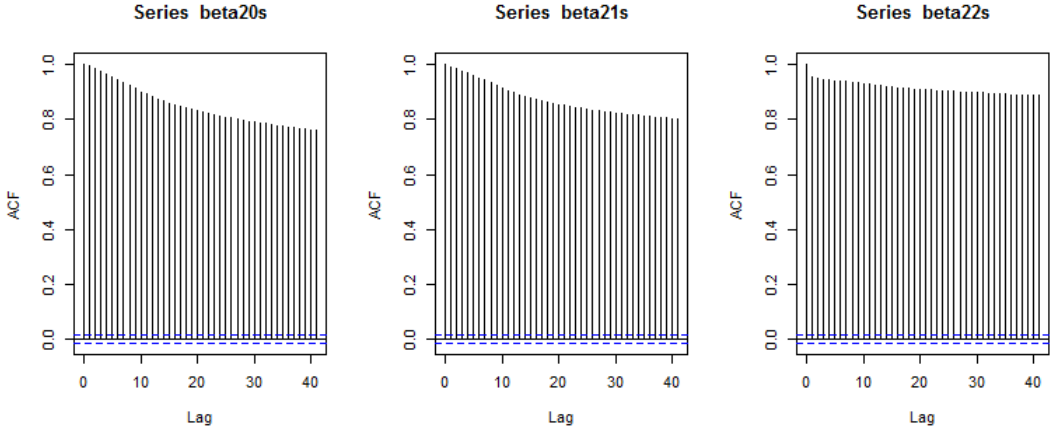

Figure 5.14: ACF plots for $\sigma_z = \sigma_y = \sigma_u = \sigma_v = 1$.

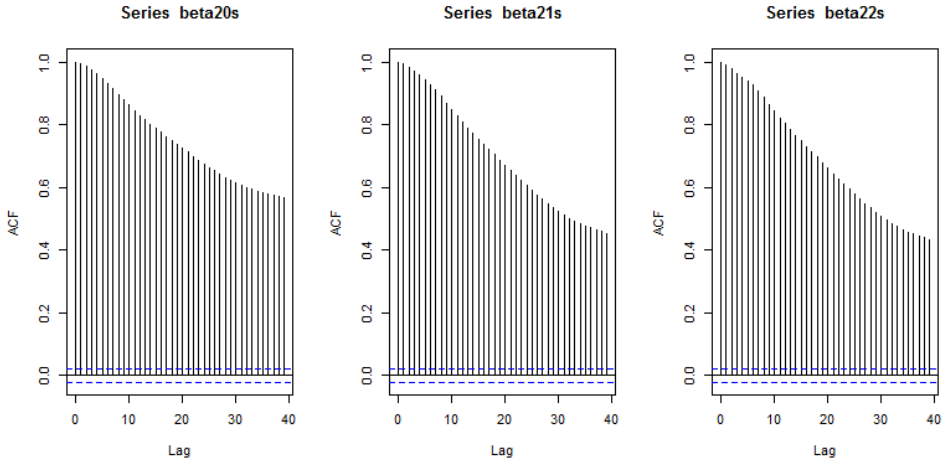

Figure 5.15: ACF plots for $\sigma_z = \sigma_u = \sigma_v = 0.25$, $\sigma_y = 0.35$.

Bibliography

[1] Spyridon J. Hatjispyros, Notes on RJMCMC-University of the Aegean

[2] David I. Hastie, Peter J. Green, Model Choice using Reversible Jump Markov Chain Monte Carlo-Imperial College London, University of Bristol (May 11, 2011)

[3] Daniel Sorensen, Daniel Gianola, Likelihood, Bayesian, and MCMC Methods in Quantitative Genetics-Springer

[4] Spyridon J. Hatjispyros, Notes on Bayesian Statistics (Greek)-University of the Aegean

[5] Patrick Lam, Convergence Diagnostics-Harvard University

[6] Dirk P. Kroese, Thomas Taimre, Zdravko I. Botev, Handbook of Monte Carlo Methods-Wiley

[7] Siddhartha Chib, Edward Greenberg, Understanding the Metropolis-Hastings Algorithm-The American Statistician, Vol.49, No.4., pp. 327-335 (Nov., 1995)

[8] Jayanta K. Ghosh, Mohan Delampady, Tapas Samanta, An Introduction to Bayesian Analysis Theory and Methods-Springer

[9] Spyridon J. Hatjispyros, Notes on Computational Bayesian Statistics (Greek)-University of the Aegean

[10] Erik Vanem, Bayesian Hierarchical Space–Time Models with Application to Significant Wave Height-Springer

[11] S. Richardson,P.J. Green, On Bayesian Analysis of Mixtures with an Unknown Number of Components-R. Statist. Soc. B59, No.4, pp. 731-792 (1997)

[12] Mary Kathryn Cowles, Bradley P. Carlin, Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review-Harvard School of Public Health, University of Minnesota

[13] Panagiotis Papastamoulis, George Iliopoulos, Computational Statistics and Data Analysis 53 900-911 (2009)

[14] package coda, r-project
`https://cran.r-project.org/web/packages/coda/index.html`

Appendix A

Diagnostic Tests

Gelman and Rubin

Gelman and Rubin (1992) propose a general approach to monitoring convergence of MCMC output in which $m > 1$ parallel chains are run with starting values that are overdispersed relative to the posterior distribution. Convergence is diagnosed when the chains have 'forgotten' their initial values, and the output from all chains is indistinguishable. The diagnostic is applied to a single variable from the chain. It is based a comparison of within-chain and between-chain variances, and is similar to a classical analysis of variance. There are two ways to estimate the variance of the stationary distribution: the mean of the empirical variance within each chain, W , and the empirical variance from all chains combined, which can be expressed as

$$\hat{\sigma}^2 = \frac{(n-1)W}{n} + \frac{B}{n}$$

where n is the number of iterations and B/n is the empirical between-chain variance.

If the chains have converged, then both estimates are unbiased. Otherwise the first method will underestimate the variance, since the individual chains have not had time to range all over the stationary distribution, and the second method will overestimate the variance, since the starting points were chosen to be overdispersed.

The convergence diagnostic is based on the assumption that the target distribution is normal. A Bayesian credible interval can be constructed using a t-distribution with mean

$$\hat{\mu} = \text{Sample mean of all chains combined}$$

and variance

$$\hat{V} = \hat{\sigma}^2 + \frac{B}{mn}$$

and degrees of freedom estimated by the method of moments

$$d = \frac{2\,\hat{V}^2}{Var(\hat{V})}$$

Use of the t-distribution accounts for the fact that the mean and variance of the posterior distribution are estimated.

The convergence diagnostic itself is

$$R = \sqrt{\frac{(d+3)\hat{V}}{(d+2)W}}$$

Values substantially above 1 indicate lack of convergence. If the chains have not converged, Bayesian credible intervals based on the t-distribution are too wide, and have the potential to shrink by this factor if the MCMC run is continued.

Geweke

Geweke (1992) proposed a convergence diagnostic for Markov chains based on a test for equality of the means of the first and last part of a Markov chain (by default the first 10% and the last 50%).

If the samples are drawn from the stationary distribution of the chain, the two means are equal and Geweke's statistic has an asymptotically standard normal distribution. The test statistic is a standard Z-score: the difference between the two sample means divided by its estimated standard error. The standard error is estimated from the spectral density at zero and so takes into account any autocorrelation. The Z-

score is calculated under the assumption that the two parts of the chain are asymptotically independent, which requires that the sum of the fraction to use from the beginning of the chain and fraction to use from the end of the chain be strictly less than 1.

Heidelberg and Welch

The convergence test uses the Cramer-von-Mises statistic to test the null hypothesis that the sampled values come from a stationary distribution. The test is successively applied, firstly to the whole chain, then after discarding the first 10%, 20%,... of the chain until either the null hypothesis is accepted, or 50% of the chain has been discarded. The latter outcome constitutes 'failure' of the stationarity test and indicates that a longer MCMC run is needed. If the stationarity test is passed, the number of iterations to keep and the number to discard are reported.

The half-width test calculates a 95% confidence interval for the mean, using the portion of the chain which passed the stationarity test. Half the width of this interval is compared with the estimate of the mean. If the ratio between the half-width and the mean is lower than some ϵ (target value for ratio of halfwidth to sample mean), the halfwidth test is passed. Otherwise the length of the sample is deemed not long enough to estimate the mean with sufficient accuracy.

If the half-width test fails then the run should be extended. In order to avoid problems caused by sequential testing, the test should not be repeated too frequently. Heidelberger and Welch (1981) suggest increasing the run length by a factor $I > 1.5$, each time, so that estimate has the same, reasonably large, proportion of new data.

Raferty and Lewis

The estimated sample size for variable U is based on the process

$$Z_t = d(U_t \leq u)$$

where d is the indicator function and u is the qth quantile of U. The process Z_t is derived from the Markov chain data by marginalization and truncation, but is not itself a Markov chain. However, Z_t may behave as a Markov chain if it is sufficiently thinned out. The diagnostic calculates the smallest value of thinning interval k which makes the thinned chain ${Z_t}^k$ behave as a Markov chain. The required sample

size is calculated from this thinned sequence. Since some data is 'thrown away' the sample size estimates are conservative.

The criterion for the number of 'burn in' iterations m to be discarded, is that the conditional distribution of ${Z_m}^k$ given Z_0 should be within the precision required for estimate of time to convergence of the equilibrium distribution of the chain ${Z_t}^k$.

Appendix B

R code

Simulation from a Normal distribution with different proposals

```
set.seed(1)
p <- 0.3
mu <- c(-2, 1.5)
sd <- c(1/2, 1.5)


f <- function(x) {
  return(p * dnorm(x, mu[1], sd[1]) + (1-p) * dnorm(x, mu[2], sd[2]))
}

proposal <- function(x) {
  return(rnorm(1, x, 4))
}

step <- function(x, f, proposal) {
  xp <- proposal(x)
  alpha <- min(1, f(xp) / f(x))
  if (runif(1) < alpha)
    x <- xp
  x
}

run <- function(x, f, proposal, nsteps) {
  res <- matrix(data=NA, nrow=nsteps, ncol=length(x))
  for (i in seq(1:nsteps))
    res[i,] <- x <- step(x, f, proposal)
  drop(res)
}

res <- run(-10, f, proposal, 100000)
res.fast <- run(-10, f, function(x) rnorm(1, x, 30), 100000)
res.slow <- run(-10, f, function(x) rnorm(1, x, 0.2), 100000)

#diagnostics
library(coda)
```

```
res.draws <- mcmc(res)
res.draws.fast <- mcmc(res.fast)
res.draws.slow <- mcmc(res.slow)


summary(res.draws)


v1 <- c(0*10^4, 2*10^4, 4*10^4, 6*10^4, 8*10^4, 10^5)
v2 <- c('0', '20000', '40000', '60000', '80000', '100000')

plot(cumsum(res)/1:length(res), type = 'l', xlab="Iteration", ylab="", xaxt='n')
axis(1, at=v1, labels=v2)
abline(c(0.457775,0), col='red')


effectiveSize(res.draws)
effectiveSize(res.draws.fast)
effectiveSize(res.draws.slow)

autocorr.diag(res.draws)
autocorr.diag(res.draws.fast)
autocorr.diag(res.draws.slow)

layout(matrix(c(1,2,3), 1, 3, byrow = TRUE))
acf(res.draws.slow, main='slow')
acf(res.draws, main='normal')
acf(res.draws.fast, main='fast')

layout(matrix(c(1,2,3), 3, 1, byrow = TRUE))
traceplot(res.draws.slow, xaxt='n', ylab = 'slow')
axis(1, at=v1, labels=v2)
traceplot(res.draws, xaxt='n', ylab = 'normal')
axis(1, at=v1, labels=v2)
traceplot(res.draws.fast, xaxt='n', ylab = 'fast')
axis(1, at=v1, labels=v2)

layout(matrix(c(1,2,3), 1, 3, byrow = TRUE))
hist(res.draws.slow, prob=T, breaks=100, main = 'slow')
lines(density(res.slow), col='red', lwd=2)
hist(res.draws, prob=T, breaks=100, main = 'normal')
lines(density(res), col='red', lwd=2)
hist(res.draws.fast, prob=T, breaks=100, main = 'fast')
lines(density(res.fast), col='red', lwd=2)



#accept-reject rate
rejectionRate(res.draws)
acceptance.rate <- 1 - rejectionRate(res.draws)
```

```
acceptance.rate

rejectionRate(res.draws.fast)
acceptance.rate <- 1 - rejectionRate(res.draws.fast)
acceptance.rate

rejectionRate(res.draws.slow)
acceptance.rate <- 1 - rejectionRate(res.draws.slow)
acceptance.rate

geweke.diag(res.draws)
raftery.diag(res.draws, q = 0.025, r = 0.005, s = 0.95)
heidel.diag(res.draws)

res1 <- run(-9, f, proposal, 30000)
res2 <- run(-7, f, proposal, 30000)
res3 <- run(-4, f, proposal, 30000)
res4 <- run(10, f, proposal, 30000)
res5 <- run(16, f, proposal, 30000)

res.draws1 <- mcmc(res1)
res.draws2 <- mcmc(res2)
res.draws3 <- mcmc(res3)
res.draws4 <- mcmc(res4)
res.draws5 <- mcmc(res5)

res.list <- mcmc.list(list(res.draws1, res.draws2, res.draws3, res.draws4,
res.draws5))
gelman.diag(res.list)
gelman.plot(res.list)
```

Simulation from a Normal Hierarchical model

```
library(coda)

#data
set.seed(1)
tau <- 1; mui <- c(1,2,3,4,5,6); mu <- 3.5; v <- 1
a <- 0.5; b <- 0.001; c <- 0.001; d <- 0.001; e <- 0.001; f <- 0.001
m <- 6; ni <- 1000


y1 <- matrix(rnorm(n=ni, mean=mui[1], sd=1/sqrt(tau)), nrow=1)
y2 <- matrix(rnorm(n=ni, mean=mui[2], sd=1/sqrt(tau)), nrow=1)
y3 <- matrix(rnorm(n=ni, mean=mui[3], sd=1/sqrt(tau)), nrow=1)
y4 <- matrix(rnorm(n=ni, mean=mui[4], sd=1/sqrt(tau)), nrow=1)
y5 <- matrix(rnorm(n=ni, mean=mui[5], sd=1/sqrt(tau)), nrow=1)
y6 <- matrix(rnorm(n=ni, mean=mui[6], sd=1/sqrt(tau)), nrow=1)
y <- rbind(y1,y2,y3,y4,y5,y6)

ydot <- .rowMeans(y, m, ni)
ysq <- (y-ydot)^2
ysqr <- .rowSums(ysq, m, ni)


n<-0
for(i in 1:m){
  n <- n+ni
}; n

mu_update <- function(mui, v){
  return(rnorm(n=1, mean=(a*b+m*mean(mui)*v)/(b+m*v), sd=1/sqrt((b+v*m))))
}

tau_update <- function(mui){
  return(rgamma(n=1, shape=c+n/2, scale=1/(d+0.5*sum(ysqr+ni*(mui-ydot)^2))))
}

v_update <- function(mui, mu){
  return(rgamma(n=1, shape=e+m/2, scale=1/(f+0.5*sum((mui-mu)^2))))
}


mui_update <- function(v, tau, mu){
  return(rnorm(n=m, mean=(v*mu+ni*tau*ydot)/(v+ni*tau), sd=1/sqrt(v+ni*tau)))
}


gibbs_sampler <- function(N=100000, burnin=25000, v_initial=v_initial,
mui_initial=mui_initial){
```

```
  mu_MC <- c(); tau_MC <- c(); v_MC <- c()
  mui_MC1 <- c(); mui_MC2 <- c(); mui_MC3 <- c(); mui_MC4 <- c(); mui_MC5 <- c();
   mui_MC6 <- c()


  mui_k <- mui_initial
  v_k <- v_initial

  for(i in 1:N){
    mu_k <- mu_update(mui=mui_k, v=v_k)
    tau_k <- tau_update(mui=mui_k)
    v_k <- v_update(mui=mui_k, mu=mu_k)
    mui_k <- mui_update(v=v_k, tau=tau_k, mu=mu_k)

    if(i>burnin){
      mu_MC[(i-burnin)] <- mu_k
      tau_MC[(i-burnin)] <- tau_k
      v_MC[(i-burnin)] <- v_k
      mui_MC1[(i-burnin)] <- mui_k[1]
      mui_MC2[(i-burnin)] <- mui_k[2]
      mui_MC3[(i-burnin)] <- mui_k[3]
      mui_MC4[(i-burnin)] <- mui_k[4]
      mui_MC5[(i-burnin)] <- mui_k[5]
      mui_MC6[(i-burnin)] <- mui_k[6]
    }

  }

  mu_MC.m <- sum(mu_MC)/(N-burnin)
  tau_MC.m <- sum(tau_MC)/(N-burnin)
  v_MC.m <- sum(v_MC)/(N-burnin)
  mui_MC1.m <- sum(mui_MC1)/(N-burnin)
  mui_MC2.m <- sum(mui_MC2)/(N-burnin)
  mui_MC3.m <- sum(mui_MC3)/(N-burnin)
  mui_MC4.m <- sum(mui_MC4)/(N-burnin)
  mui_MC5.m <- sum(mui_MC5)/(N-burnin)
  mui_MC6.m <- sum(mui_MC6)/(N-burnin)

  ergodic.means <- c(mu_MC.m, tau_MC.m, v_MC.m, mui_MC1.m, mui_MC2.m, mui_MC3.m,
   mui_MC4.m, mui_MC5.m, mui_MC6.m)

  parameters <- c(mu_MC, tau_MC, v_MC, mui_MC1, mui_MC2, mui_MC3, mui_MC4, mui_MC5,
  mui_MC6)

  return(list(parameters, ergodic.means))
}

posterior <- gibbs_sampler(N=25000, burnin=5000,
v_initial=1, mui_initial=c(0.8,1.3,2.5,3.4,4.3,5.7))
posterior
```

```
v1 <- posterior[[1]]
v2 <- posterior[[2]]

mus <- v1[(1:20000)]
taus <- v1[(20001:40000)]
vs <- v1[(40001:60000)]
mui1s <- v1[(60001:80000)]
mui2s <- v1[(80001:100000)]
mui3s <- v1[(100001:120000)]
mui4s <- v1[(120001:140000)]
mui5s <- v1[(140001:160000)]
mui6s <- v1[(160001:180000)]


mus.draws <- mcmc(mus)
summary(mus.draws)
geweke.diag(mus.draws)
heidel.diag(mus.draws)

taus.draws <- mcmc(taus)
summary(taus.draws)
geweke.diag(taus.draws)
heidel.diag(taus.draws)

vs.draws <- mcmc(vs)
summary(vs.draws)
geweke.diag(vs.draws)
heidel.diag(vs.draws)

mui1s.draws <- mcmc(mui1s)
summary(mui1s.draws)
geweke.diag(mui1s.draws)
heidel.diag(mui1s.draws)

mui2s.draws <- mcmc(mui2s)
summary(mui2s.draws)
geweke.diag(mui2s.draws)
heidel.diag(mui2s.draws)

mui3s.draws <- mcmc(mui3s)
summary(mui3s.draws)
geweke.diag(mui3s.draws)
heidel.diag(mui3s.draws)

mui4s.draws <- mcmc(mui4s)
summary(mui4s.draws)
geweke.diag(mui4s.draws)
heidel.diag(mui4s.draws)
```

```
mui5s.draws <- mcmc(mui5s)
summary(mui5s.draws)
geweke.diag(mui5s.draws)
heidel.diag(mui5s.draws)

mui6s.draws <- mcmc(mui6s)
summary(mui6s.draws)
geweke.diag(mui6s.draws)
heidel.diag(mui6s.draws)

layout(matrix(c(1,2,3,4,5,6,7,8,9),3,3, byrow = TRUE))
hist(mus, prob=T, breaks=70, xlab="")
lines(density(mus), col='red', lwd=2)
hist(taus, prob=T, breaks=70, xlab="")
lines(density(taus), col='red', lwd=2)
hist(vs, prob=T, breaks=70, xlab="")
lines(density(vs), col='red', lwd=2)
hist(mui1s, prob=T, breaks=70, xlab="")
lines(density(mui1s), col='red', lwd=2)
hist(mui2s, prob=T, breaks=70, xlab="")
lines(density(mui2s), col='red', lwd=2)
hist(mui3s, prob=T, breaks=70, xlab="")
lines(density(mui3s), col='red', lwd=2)
hist(mui4s, prob=T, breaks=70, xlab="")
lines(density(mui4s), col='red', lwd=2)
hist(mui5s, prob=T, breaks=70, xlab="")
lines(density(mui5s), col='red', lwd=2)
hist(mui6s, prob=T, breaks=70, xlab="")
lines(density(mui6s), col='red', lwd=2)

layout(matrix(c(1,2,3,4,5,6,7,8,9),3,3, byrow = TRUE))
plot(cumsum(mus)/1:length(mus), type = 'l', xlab="Iteration", ylab="")
abline(c(v2[1],0), col='red')
plot(cumsum(taus)/1:length(taus), type = 'l', xlab="Iteration", ylab="")
abline(c(v2[2],0), col='red')
plot(cumsum(vs)/1:length(vs), type = 'l', xlab="Iteration", ylab="")
abline(c(v2[3],0), col='red')
plot(cumsum(mui1s)/1:length(mui1s), type = 'l', xlab="Iteration", ylab="")
abline(c(v2[4],0), col='red')
plot(cumsum(mui2s)/1:length(mui2s), type = 'l', xlab="Iteration", ylab="")
abline(c(v2[5],0), col='red')
plot(cumsum(mui3s)/1:length(mui3s), type = 'l', xlab="Iteration", ylab="")
abline(c(v2[6],0), col='red')
plot(cumsum(mui4s)/1:length(mui4s), type = 'l', xlab="Iteration", ylab="")
abline(c(v2[7],0), col='red')
plot(cumsum(mui5s)/1:length(mui5s), type = 'l', xlab="Iteration", ylab="")
abline(c(v2[8],0), col='red')
plot(cumsum(mui6s)/1:length(mui6s), type = 'l', xlab="Iteration", ylab="")
abline(c(v2[9],0), col='red')
```

```
layout(matrix(c(1,2,3,4,5,6,7,8,9),3,3, byrow = TRUE))
traceplot(mus.draws, ylab="")
traceplot(taus.draws, ylab="")
traceplot(vs.draws, ylab="")
traceplot(mui1s.draws, ylab="")
traceplot(mui2s.draws, ylab="")
traceplot(mui3s.draws, ylab="")
traceplot(mui4s.draws, ylab="")
traceplot(mui5s.draws, ylab="")
traceplot(mui6s.draws, ylab="")
```

Model choice in regression - FF strategy

```
norm_vec <- function(x) sqrt(sum(x^2))

f <- function(b,z,A){
  if(!is.matrix(A)) stop("A must be a matrix")
  return(exp(-0.5*norm_vec(z-A[ ,c(1:length(b))]%*%b)^2))
}

g <- function(u){
  exp(-0.5*norm_vec(u)^2)/(2*pi)^(length(u)/2)
}

myregression <- function(T=1000000, myseed=1){
  set.seed(myseed)
  a <- seq(from=0, to=5, by=1/20)
  b <- matrix(c(1, 0.3, 0.15, 0.005), ncol=1)
  A <- matrix(c(a^0, a^1, a^2, a^3), nrow=length(a))
  noise <- matrix(rnorm(101), ncol=1)
  z <- A %*% b + noise

  m <- 1
  x <- matrix(rnorm(m), ncol=1)

  data <- matrix(rep(0, T), ncol=1)

  beta0 <- matrix(c(0), ncol=1)
  beta1 <- matrix(c(0,0), ncol=1)
  beta2 <- matrix(c(0,0,0), ncol=1)
  beta3 <- matrix(c(0,0,0,0), ncol=1)


  for(i in 1:T){
    n <- ceiling(4*runif(1))
    y <- matrix(rnorm(n), ncol=1)
    if(runif(1) < min(f(y,z,A)/f(x,z,A)*g(x)/g(y)*n/m,1)){
      x <- y; m <- n
    }
    data[i] <- m
    if(m==1){
      beta0 <- beta0 + x
    } else if(m==2){
      beta1 <- beta1 + x
    } else if(m==3){
      beta2 <- beta2 + x
    } else if(m==4){
      beta3 <- beta3 + x
    }
```

```
  }

  beta0 <- beta0/length(data[data==1])
  beta1 <- beta1/length(data[data==2])
  beta2 <- beta2/length(data[data==3])
  beta3 <- beta3/length(data[data==4])

  beta <- c(beta0, beta1, beta2, beta3)

  p1 <- length(data[data==1])/T
  p2 <- length(data[data==2])/T
  p3 <- length(data[data==3])/T
  p4 <- length(data[data==4])/T

  p <- c(p1,p2,p3,p4)

  return(list(beta,p))
}

K <- myregression(T=1000000, myseed=1)

myregression <- function(T=1000000, myseed=1){
  set.seed(myseed)
  a <- seq(from=0, to=5, by=1/20)
  b <- matrix(c(1, 0.3, 0.15, 0.005), ncol=1)
  A <- matrix(c(a^0, a^1, a^2, a^3), nrow=length(a))
  noise <- matrix(rnorm(101), ncol=1)
  z <- A %*% b + noise

  m <- 1
  x <- matrix(rnorm(m), ncol=1)

  data <- matrix(rep(0, T), ncol=1)

  beta0 <- matrix(c(0), ncol=1)
  beta1 <- matrix(c(0,0), ncol=1)
  beta2 <- matrix(c(0,0,0), ncol=1)
  beta3 <- matrix(c(0,0,0,0), ncol=1)

  beta20 <- c(); beta21 <- c(); beta22 <- c()
  for(i in 1:T){
    n <- ceiling(4*runif(1))
    y <- matrix(rnorm(n), ncol=1)
    if(runif(1) < min(f(y,z,A)/f(x,z,A)*g(x)/g(y)*n/m,1)){
      x <- y; m <- n
    }
    data[i] <- m
    if(m==1){
      beta0 <- beta0 + x
    } else if(m==2){
```

```
      beta1 <- beta1 + x
    } else if(m==3){
      beta2 <- beta2 + x
    } else if(m==4){
      beta3 <- beta3 + x
    }


    if(i%%50==0 && m==3){
      s0 <- beta2[1]/length(data[data==3])
      beta20 <- c(beta20, s0)
    }

    if(i%%50==0 && m==3){
      s1 <- beta2[2]/length(data[data==3])
      beta21 <- c(beta21, s1)
    }

    if(i%%50==0 && m==3){
      s2 <- beta2[3]/length(data[data==3])
      beta22 <- c(beta22, s2)
    }

  }
  return(list(beta20, beta21, beta22))
}

M <- myregression(T=1000000, myseed=1)

beta20s <- M[[1]]
beta21s <- M[[2]]
beta22s <- M[[3]]

layout(matrix(c(1,2),1,2,byrow = TRUE))
hist(beta20s, prob=T, breaks=100, xlab="beta20")
lines(density(beta20s), col='red', lwd=2)
plot(beta20s, type='l', pch=".", xlab="n", ylab="beta20")
abline(1.06972376,0,col='red')

layout(matrix(c(1,2),1,2,byrow = TRUE))
hist(beta21s, prob=T, breaks=100, xlab="beta21")
lines(density(beta21s), col='red', lwd=2)
plot(beta21s, type='l', pch=".", xlab="n", ylab="beta21")
abline(0.31733055,0,col='red')

layout(matrix(c(1,2),1,2,byrow = TRUE))
hist(beta22s, prob=T, breaks=100, xlab="beta22")
lines(density(beta22s), col='red', lwd=2)
plot(beta22s, type='l', pch=".", xlab="n", ylab="beta22")
abline(0.16497739,0,col='red')
```

```
library(coda)
beta20s.draws <- mcmc(beta20s, thin=50)
beta21s.draws <- mcmc(beta21s, thin=50)
beta22s.draws <- mcmc(beta22s, thin=50)

heidel.diag(beta20s.draws)
raftery.diag(beta20s.draws, q = 0.025, r = 0.005, s = 0.95)

summary(beta21s.draws)
heidel.diag(beta21s.draws)
raftery.diag(beta21s.draws, q = 0.025, r = 0.005, s = 0.95)

summary(beta22s.draws)
heidel.diag(beta22s.draws)
raftery.diag(beta22s.draws, q = 0.025, r = 0.005, s = 0.95)

layout(matrix(c(1,2,3),1,3,byrow=TRUE))
acf(beta20s)
acf(beta21s)
acf(beta22s)

myregression <- function(T=1000000, myseed=1){
  set.seed(myseed)
  a <- seq(from=0, to=5, by=1/20)
  b <- matrix(c(1, 0.3, 0.15, 0.005), ncol=1)
  A <- matrix(c(a^0, a^1, a^2, a^3), nrow=length(a))
  noise <- matrix(rnorm(101), ncol=1)
  z <- A %*% b + noise

  m <- 1
  x <- matrix(rnorm(m), ncol=1)

  data <- matrix(rep(0, T), ncol=1)

  beta0 <- matrix(c(0), ncol=1)
  beta1 <- matrix(c(0,0), ncol=1)
  beta2 <- matrix(c(0,0,0), ncol=1)
  beta3 <- matrix(c(0,0,0,0), ncol=1)

  ravp1 <- c(); ravp2 <- c(); ravp3 <- c(); ravp4 <- c()
  for(i in 1:T){
    n <- ceiling(4*runif(1))
    y <- matrix(rnorm(n), ncol=1)
    if(runif(1) < min(f(y,z,A)/f(x,z,A)*g(x)/g(y)*n/m,1)){
      x <- y; m <- n
    }
    data[i] <- m
    if(m==1){
      beta0 <- beta0 + x
```

```
    } else if(m==2){
      beta1 <- beta1 + x
    } else if(m==3){
      beta2 <- beta2 + x
    } else if(m==4){
      beta3 <- beta3 + x
    }

    if(i%%50==0){
      cp1 <- length(data[data==1])/i
      ravp1 <- c(ravp1,cp1)
    }

    if(i%%50==0){
      cp2 <- length(data[data==2])/i
      ravp2 <- c(ravp2,cp2)
    }


    if(i%%50==0){
      cp3 <- length(data[data==3])/i
      ravp3 <- c(ravp3,cp3)
    }

    if(i%%50==0){
      cp4 <- length(data[data==4])/i
      ravp4 <- c(ravp4,cp4)
    }

  }

  return(list(ravp1, ravp2, ravp3, ravp4))
}

L <- myregression(T=1000000, myseed=1)

ravp1 <- L[[1]]
ravp2 <- L[[2]]
ravp3 <- L[[3]]
ravp4 <- L[[4]]

layout(matrix(c(1,2,3,4), 2,2,byrow=TRUE))
plot(ravp1, type='l', pch=".", xlab="n", ylab="p1")
abline(0.000004,0, col='red')

plot(ravp2, type='l', pch=".", xlab="n", ylab="p2")
abline(0.029043,0, col='red')

plot(ravp3, type='l', pch=".", xlab="n", ylab="p3")
abline(0.814028,0, col='red')
```

```
plot(ravp4, type='l', pch=".", xlab="n", ylab="p4")
abline(0.156925,0, col='red')
```

Model choice in regression - Stochastic proposals

```
norm_vec <- function(x) sqrt(sum(x^2))

regression <- function(T=1000000, myseed=1, sigma0=0.2, sigmap=0.3, sigmar=0.2) {

  set.seed(myseed)
  a <- seq(from=0, to=5, by=1/20)
  bor <- matrix(c(1, 0.3, 0.15, 0.005), ncol=1)
  A <- matrix(c(a^0, a^1, a^2, a^3), nrow=length(a))
  noise <- matrix(rnorm(101,0,sigma0), ncol=1)
  z <- A %*% bor + noise


  n0 <- ceiling(4*runif(1))
  b <- matrix(rnorm(n0,0,sigmap), ncol=1)

  data <- matrix(rep(0, T), ncol=1)

  beta0 <- matrix(c(0), ncol=1)
  beta1 <- matrix(c(0,0), ncol=1)
  beta2 <- matrix(c(0,0,0), ncol=1)
  beta3 <- matrix(c(0,0,0,0), ncol=1)


  for (i in 1:T) {
    n <- ceiling(4*runif(1))
    if(n<n0) {
      u1 <- matrix(rnorm(n,0,sigma0), ncol = 1)
      b1 <- matrix(b[c(1:length(u1)), ], ncol = 1)
      b2 <- matrix(b[seq(length(u1)+1,n0,1)], ncol = 1)
      bstar <- b1+u1
      u <- matrix(c(u1,b2), ncol = 1)
      g1 <- exp(-1/(2*sigmar^2)*norm_vec(u1)^2)/(2*pi*sigmar^2)^(n/2)
      g2 <- exp(-1/(2*sigmar^2)*norm_vec(u)^2)/(2*pi*sigmar^2)^(n0/2)
    }
    else if(n>n0){
      u <- matrix(rnorm(n,0,sigma0), ncol = 1)
      u1 <- matrix(u[c(1:n0), ], ncol = 1)
      u2 <- matrix(u[seq(n0+1,n,1)], ncol = 1)
      bn <- matrix(c(b,rep(0,length(u2))), ncol = 1)
      bstar <- bn+u
      g1 <- exp(-1/(2*sigmar^2)*norm_vec(u)^2)/(2*pi*sigmar^2)^(n/2)
      g2 <- exp(-1/(2*sigmar^2)*norm_vec(u1)^2)/(2*pi*sigmar^2)^(n0/2)
    }
    else if(n==n0){
      u1 <- matrix(rnorm(n,0,sigma0), ncol = 1)
      b1 <- matrix(b[c(1:length(u1)), ], ncol = 1)
      bstar <- b1+u1
      g1 <- exp(-1/(2*sigmar^2)*norm_vec(u1)^2)/(2*pi*sigmar^2)^(n/2)
```

```
      g2 <- exp(-1/(2*sigmar^2)*norm_vec(u1)^2)/(2*pi*sigmar^2)^(n0/2)
    }

    e1 <- norm_vec(z-A[ ,c(1:length(bstar))]%*%bstar)^2
    e2 <- norm_vec(z-A[ ,c(1:length(b))]%*%b)^2
    z1 <- norm_vec(bstar)^2
    z2 <- norm_vec(b)^2
    f1 <- (2*pi*sigmap^2)^((n0-n)/2)
    f2 <- (2*pi*sigma0^2)^((n0-n)/2)

    accr <- (exp((-1/2*sigma0^2)*(e1-e2))*exp((-1/2*sigmap^2)*(z1-z2))*f1*f2*g2)/g1

    if(runif(1) < min(accr,1)){
      b <- bstar; n0 <- n
    }

    data[i] <- n0
    if(n0==1){
      beta0 <- beta0 + b
    } else if(n0==2){
      beta1 <- beta1 + b
    } else if(n0==3){
      beta2 <- beta2 + b
    } else if(n0==4){
      beta3 <- beta3 + b
    }

  }

  beta0 <- beta0/length(data[data==1])
  beta1 <- beta1/length(data[data==2])
  beta2 <- beta2/length(data[data==3])
  beta3 <- beta3/length(data[data==4])

  beta <- c(beta0, beta1, beta2, beta3)

  p1 <- length(data[data==1])/T
  p2 <- length(data[data==2])/T
  p3 <- length(data[data==3])/T
  p4 <- length(data[data==4])/T

  p <- c(p1,p2,p3,p4)

  return(list(beta,p))
}

stochastic <- regression(T=1000000, myseed=1, sigma0=1, sigmap=1, sigmar=1)
```