NORGES TEKNISK-NATURVITENSKAPELIGE UNIVERSITET

On the use of local optimisations within Metropolis-Hastings

by

Håkon Tjelmeland and Jo Eidsvik

PREPRINT
STATISTICS NO. 5/2002

NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY
TRONDHEIM, NORWAY

This report has URL `http://www.math.ntnu.no/preprint/statistics/2002/S5-2002.ps`
Håkon Tjelmeland has homepage: `http://www.math.ntnu.no/~haakont`
E-mail: `haakont@stat.ntnu.no`
Address: Department of Mathematical Sciences, Norwegian University of Science and Technology, N-7491 Trondheim, Norway.

On the use of local optimisations within Metropolis-Hastings

Håkon Tjelmeland and Jo Eidsvik
Department of Mathematical Sciences
Norwegian University of Science and Technology
Trondheim, Norway

Abstract

Markov chain Monte Carlo algorithms generate samples from a specified target distribution by simulating a Markov chain. In the Metropolis-Hastings set-up, each iteration consists of two parts. First, a potential new state is proposed from a proposal distribution and, second, the potential new state is accepted with a given probability. If a prior approximation to the target distribution is available, this can be used to generate potential new states independent of the current value, but, if the approximation is poor in parts of the sample space, the resulting convergence will be extremely slow.

In the present paper we limit attention to continuous distributions on $\Re^n$ and study how optimisation based mode jumping proposals can be combined with proposals from a prior approximation. Thereby one gets the computational efficiency associated with a fixed proposal distribution, but forestall the possibly severe consequences of a poor approximation in parts of the sample space. Next, we explore how optimisation results can be used to update the prior approximation without destroying the detailed balance of the chain. Thereby we get an adaptive Markov chain algorithm. We evaluate the effectiveness of the proposed algorithms in two simulation examples. The first considers a simple mixture of bivariate normal distributions, whereas the last samples from a posterior distribution based on a previously analysed data set.

Key words: Markov chain Monte Carlo, mode jumping proposals, adaptation, optimisation, prior approximation, multi-modal distribution, Metropolis-Hastings

1 Introduction

In Markov chain Monte Carlo (MCMC) samples from a specified target distribution, $\pi(x)$, are generated by simulating a Markov chain with limiting distribution equal to $\pi(x)$, see, for example, Besag et al. (1995) and references therein. After a fixed period, known as "burn-in", the chain has effectively converged and produces realisations from the prescribed distribution. A recipe for construction of Markov chains with the desired limiting distribution is given by the Metropolis-Hastings set-up.

For many target distributions quite simple Metropolis-Hastings algorithms works satisfactory. Numerous such cases can be found in the statistical literature, see for example Gilks et al. (1996). For more complex target distributions, careful tuning of the proposal mechanisms in Metropolis-Hastings may be necessary to obtain a Markov chain with an acceptable mixing rate. This is, for example, the situation if $\pi(x)$ contains several modes separated by low probability areas. One possibility is then to construct an approximation to the target density, from which efficient exact sampling is possible, and use this to propose potential new values in a Metropolis-Hastings setting. As discussed in Gelman and Rubin (1992), the approximation can be a mixture of multivariate t-distributions fitted to the local maxima for $\pi(x)$ found in a number of prior optimisation searches. If the approximation is reasonably good in the entire sample space, such an independent proposal Metropolis-Hastings algorithm will have good convergence and mixing properties. However, if the approximation is poor in some regions, the resulting convergence can become extremely slow. With the approximation strategy in Gelman and Rubin (1992), this may happen if one or more modes are missed in the prior search.

As an alternative to using optimisation searches to construct a prior approximation to the target distribution, Chib and Greenberg (1994) discusses how to use optimisations within Metropolis-Hastings steps. This strategy is further developed in Tjelmeland and Hegstad (2001), where optimisations are used to define direct mode jumping proposals. By continual use of optimisations within Metropolis-Hastings steps, one avoids the risk of missing modes as with a finite number of prior searches. However, this does not come without a computational cost, as running optimisations require a lot more computing resources than just generating proposals from some fixed proposal density.

In the present paper, we limit the attention to continuous probability distributions on $\Re^n$ and study how optimisation based mode jumping proposals can be combined with proposals from an available prior approximation. Thereby we get some of the computational efficiency associated with a fixed proposal distribution, but forestall the possibly severe consequences of a poor prior approximation in parts of the sample space. To take full advantage of the two types of proposals, the Metropolis-Hastings kernel should be constructed so that the number of proposals of each type automatically adjust to the quality of the approximation available.

The paper is organised as follows. Section 2 briefly reviews the MCMC strategy and the Metropolis-Hastings set-up. In Section 3 we discuss how to combine proposals from a prior approximation with optimisation based mode jumping proposals. In Section 4 we consider how the optimisations within Metropolis-Hastings also can be used to update the prior approximation and thereby define an adaptive Markov chain algorithm. Section 5 evaluates the effectiveness of the proposed algorithms in two examples and, finally, Section 6 provides conclusions.

2 Markov chain Monte Carlo

A general introduction to MCMC can be found in the references mentioned above. Here, we first define a few basic concepts in Markov chain theory and then briefly review the Metropolis-Hastings set-up. In the following we assume the target distribution to be continuous on $\Re^n$ and let $\pi(x)$ denote its density with respect to the Lebesgue measure. The normalising constant of $\pi(x)$ is possibly unknown and we let $h(x)$ be a corresponding unnormalised density function.

2.1 Detailed balance condition

The idea in MCMC is to construct a Markov chain with a stationary transition kernel P with limiting distribution equal to $\pi(x)$. Thus, $\pi(\cdot)$ must be an invariant distribution for P. In a Metropolis-Hastings algorithm, P is furthermore restricted to be time reversible, i.e. P must fulfil the detailed balance condition,

$$\int_A \pi(x)\mathrm{P}(B|x)\mathrm{d}x = \int_B \pi(x)\mathrm{P}(A|x)\mathrm{d}x \quad \forall\ A, B \in \mathcal{F}_n. \tag{1}$$

In MCMC combinations of transition kernels satisfying equation (1) are frequently used. If two kernels P_1 and P_2 both fulfil equation (1) it follows directly that $a\mathrm{P}_1 + (1-a)\mathrm{P}_2$ also does so for any $a \in [0,1]$. Likewise, if P^φ satisfy equation (1) for all $\varphi \in \Re^d$, then

$$\mathrm{P}(A|x) = \int_{\Re^d} \mathrm{P}^\varphi(A|x)a(\varphi)\mathrm{d}\varphi \tag{2}$$

also fulfils detailed balance with respect to $\pi(\cdot)$ for any density $a(\cdot)$ on $\Re^d$.

2.2 Metropolis-Hastings

Metropolis-Hastings is a recipe for construction of transition kernels satisfying equation (1). The set-up given in the following is a generalisation of the scheme used in Tjelmeland and Hegstad (2001).

Let $\mathrm{Q}_0, \mathrm{Q}_1, \ldots, \mathrm{Q}_{N-1}$ be N transition kernels on $\Re^n$, which we assume to have corresponding densities $q_i(\cdot|x); i = 0, 1, \ldots, N-1$ with respect to the Lebesgue measure. Furthermore, for each $x \in \Re^n$ we let $p(f, b|x)$ denote a joint probability distribution for a pair (f, b) for $f, b = 0, 1, \ldots, N-1$. Then, a corresponding Metropolis-Hastings transition kernel P is defined by

$$\mathrm{P}(A|x) = \sum_{f=0}^{N-1} \sum_{b=0}^{N-1} \left[p(f, b|x) \int_A q_f(y|x) \alpha(y|x, f, b) \mathrm{d}y \right] + I(x \in A) r(x), \tag{3}$$

where the acceptance probabilities $\alpha(y|x, f, b)$ are

$$\alpha(y|x, f, b) = \min\left\{1, \frac{h(y)}{h(x)} \frac{q_b(x|y)}{q_f(y|x)} \frac{p(b, f|y)}{p(f, b|x)}\right\} \quad \text{for} \quad f, b = 0, \ldots, N-1 \tag{4}$$

and $r(x)$ is the rejection probability, given as

$$r(x) = \sum_{f=0}^{N-1} \sum_{b=0}^{N-1} \left[p(f, b|x) \int_{\Re^n} q_f(y|x)(1 - \alpha(y|x, f, b)) \mathrm{d}y \right]. \tag{5}$$

Thus, with x denoting the current state, one iteration of the corresponding Metropolis-Hastings algorithm becomes; 1) draw a pair (f, b) from the distribution $p(f, b|x)$, 2) generate a potential new state y from $q_f(\cdot|x)$, 3) sample u from the uniform distribution on $[0, 1]$, and 4) if $u < \alpha(y|x, f, b)$ let y be the new state and otherwise leave the state vector unchanged. The Q_f is thereby used forward in time by proposing to go from x to y. The Q_b only appears in the acceptance probability, where it enters as the probability for the corresponding backward proposal from y to x.

To prove that the transition kernel in equation (3) is time reversible with respect to $\pi(x)$ is easily done by direct calculations, or one may rewrite the kernel and show how it originates for one specific choice of $s(x, y)$ in the original set-up of Hastings (1970). The latter implies that the above set-up is sub-optimal, unless $N = 1$, in terms of asymptotic variance. According to Peskun (1973), it is better to define the overall proposal density

$$q(y|x) = \sum_{f=0}^{N-1} \left[\left(\sum_{b=0}^{N-1} p(f, b|x) \right) q_f(y|x) \right], \tag{6}$$

to propose y from $q(\cdot|x)$ and accept y with probability $\min\{1, (h(y)q(x|y))/(h(x)q(y|x)\}$. This algorithm has exactly the same proposal mechanism as above, but with higher acceptance probabilities. However, Peskun measures efficiency in terms of number of iterations and in the Peskun optimal algorithm, $q_f(y|x)$ and $q_b(x|y)$ have to be computed for all $f, b = 0, 1, \ldots, N-1$, whereas in the Peskun sub-optimal algorithm it is sufficient to have $q_f(y|x)$ and $q_b(x|y)$ for one value of f and b. Thus, when efficiency is measured in terms of required computer resources, the Peskun sub-optimal algorithm may be preferable. In the next section we discuss one such situation.

It should be noted that one may use the above scheme also to construct a transition kernel indexed with a parameter φ, P^φ, to be used as in equation (2). Then, both the proposal kernels and the probabilities $p(f, b|x)$ may depend on φ, and as a consequence also the acceptance probabilities and the rejection probability become functions of φ. To indicate the dependence on φ, we then use the notation Q_i^φ, $q_i^\varphi(y|x)$, $p^\varphi(f, b|x)$, $\alpha^\varphi(y|x, f, b)$ and $r^\varphi(x)$.

3 Mode jumping proposals

Tjelmeland and Hegstad (2001) discusses how to sample from a multi-modal target distribution by use of so-called mode jumping proposals. Here, we use the set-up from Section 2.2 to define a refined version of their approach.

Assume that we prior to the MCMC simulation have available an approximation to the target distribution and let $g(x); x \in \Re^n$ denote its (normalised) density function. As suggested in Gelman and Rubin (1992), $g(x)$ may be a mixture of multivariate t-densities fitted to modes located in a prior optimisation search. Thus, up to an unknown multiplicative constant, we assume $g(x)$ to be a good approximation to $\pi(x)$ in some parts of $\Re^n$, but perhaps not in others. If $g(x)$ is the only proposal distribution used we get an independent proposal Metropolis-Hastings algorithm. If $\beta = \inf_{x \in \Re^n} [g(x)/\pi(x)] > 0$ the convergence becomes geometric with rate $1 - \beta$ (Mengersen and Tweedie, 1996). In the following, we require this condition to hold. If $\pi(x)$ is bounded, a sufficient condition is that $g(x)$ has heavier tails than $\pi(x)$, which can often be accomplished with a mixture of t-distributions. However, even geometric convergence is extremely slow if β is very close to zero, which will be the case if the prior optimisation search missed some modes. Thus, intuitively one should expect convergence to be improved by combining $g(x)$ with other proposal kernels which concentrates most of the probability mass in regions where $g(x)/\pi(x)$ is small. Such an area can be located by first drawing a $\varphi \in \Re^n$ from some wide distribution and, starting at φ, running a local minimisation algorithm for $U(x) = \ln(g(x)) - \ln(h(x))$. Let $\mu(\varphi)$ denote the location of the minimum found when starting at φ and let $\Sigma(\varphi)$ be the inverse of the Hessian at this minimum, i.e. $\Sigma(\varphi) = [(\nabla^2 U)(\mu(\varphi))]^{-1}$. One may then propose a potential new state from a non-central t-distribution with some fixed ν degrees of freedom, centered at $\mu(\varphi)$ and with correlation governed by $\Sigma(\varphi)$.

Following the ideas discussed above, we define a transition kernel P^{φ} according to the set-up in Section 2.2 with index $\varphi = (\varphi_1, \varphi_2)$, where $\varphi_1, \varphi_2 \in \Re^n$. We combine the following $N = 3$ proposal kernels

$$\begin{aligned} q_0^{\varphi}(y|x) &= g(y) && (7) \\ q_1^{\varphi}(y|x) &= \mathrm{T}_{\nu}(\mu(\varphi_1), \Sigma(\varphi_1))(y) && (8) \\ q_2^{\varphi}(y|x) &= \mathrm{T}_{\nu}(\mu(\varphi_2), \Sigma(\varphi_2))(y), && (9) \end{aligned}$$

where $\mathrm{T}_{\nu}(\mu, \Sigma)(y)$ denotes the density of a multivariate t-distribution with ν degrees of freedom, centered at μ and with covariance matrix defined by Σ.

To discuss the choice of $p^{\varphi}(f, b|x)$, we call it a (f, b)-move whenever specified values for f and b are used. We first set $p^{\varphi}(0, 2|x) = p^{\varphi}(2, 0|x) = 0$ as $(1, 0)$- and $(0, 1)$-moves have similar effects to $(2, 0)$- and $(0, 2)$-moves, respectively. We also set $p^{\varphi}(1, 1|x) = p^{\varphi}(2, 2|x) = 0$ as $(1, 2)$- and $(2, 1)$-moves correspond to $(1, 1)$- and $(2, 2)$-moves whenever $\mu(\varphi_1) = \mu(\varphi_2)$. To specify the remaining elements of $p^{\varphi}(f, b|x)$, we write

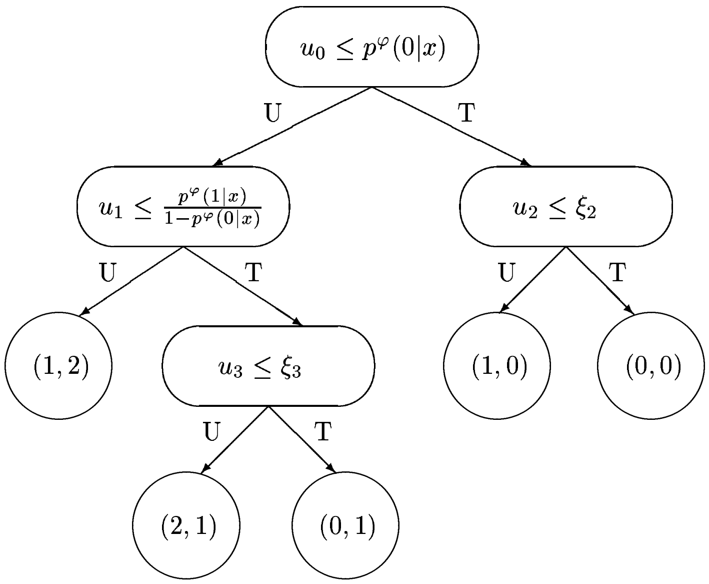
$$p^{\varphi}(f, b|x) = p^{\varphi}(b|x) p^{\varphi}(f|x, b), \tag{10}$$

where $p^{\varphi}(b|x)$ and $p^{\varphi}(f|x, b)$ are the marginal distribution for b and the conditional distribution for f given b, respectively. We first focus on $p^{\varphi}(b|x)$. As $g(x)$ appears in the numerator in the expression for the acceptance probability for a $(f, 0)$-move, we want $p^{\varphi}(0|x)$ to be high only if x is in a region where $g(x)$ provides a good approximation to $\pi(x)$. Later we will also need to bound $p^{\varphi}(0|x)$ away from zero, so a natural choice for $p^{\varphi}(0|x)$ becomes

$$p^{\varphi}(0|x) = \varepsilon + (1 - \varepsilon) \left[1 + \xi_0 \frac{h(x)}{kg(x)} \right]^{-1}, \tag{11}$$

where $\varepsilon \in (0, 1)$ and $\xi_0 > 0$ are constants to be specified and k should be chosen so that $kg(x)$ and $h(x)$ are of similar size in high density regions of $g(x)$. To set a reasonable value for k one may, prior to running the Metropolis-Hastings algorithm, sample $z_1, \ldots, z_s$ independently from $g(x)$ and set $k = \mathrm{median}_{i=1,\ldots,s}[h(z_i)/g(z_i)]$. It should be noted that k is not, and should not be, an estimate of the unknown normalising constant in $\pi(x)$. By similar arguments as for $p^{\varphi}(0|x)$, the probability $p^{\varphi}(1|x)$ should be high only when $T_{\nu}(\mu(\varphi_1), \Sigma(\varphi_1))(x)$ is high, i.e. if x is within the mode located by the optimisation search started at φ_1. Thus, a natural choice is to set

$$\frac{p^{\varphi}(1|x)}{1 - p^{\varphi}(0|x)} = \left[1 + \frac{\xi_1}{n} (x - \mu(\varphi_1))^T \Sigma(\varphi_1)^{-1} (x - \mu(\varphi_1)) \right]^{-1}, \tag{12}$$



$u_0 \le p^{\varphi}(0|x)$
U
T
$u_1 \le \frac{p^{\varphi}(1|x)}{1-p^{\varphi}(0|x)}$
$u_2 \le \xi_2$
U
T
U
T
$(1,2)$
$u_3 \le \xi_3$
$(1,0)$
$(0,0)$
U
T
$(2,1)$
$(0,1)$

Figure 1: Algorithm for sampling (f,b) from $p^{\varphi}(f,b|x)$. If an inequality in an oval is true follow the right arrow (T), and otherwise choose the left alternative (U). The variables u_0, u_1, u_2 and u_3 are independently and uniformly sampled on $[0,1]$.

where ξ_1 is a constant to be specified. Then, of course, $p^{\varphi}(2|x) = 1 - p^{\varphi}(1|x) - p^{\varphi}(0|x)$.

We can see no good arguments for letting $p^{\varphi}(f|x,b)$ be a function of x or φ. Thereby, the two parameters $\xi_2 = p^{\varphi}(0|x,0)$ and $\xi_3 = p^{\varphi}(0|x,1)$ gives all the probabilities $p^{\varphi}(f|x,b); f,b = 0,1,\dots,N-1$. This is so because, to get the vanishing probabilities $p^{\varphi}(f,b|x)$ discussed above, one must have $p^{\varphi}(2|x,0) = p^{\varphi}(1|x,1) = p^{\varphi}(0|x,2) = p^{\varphi}(2|x,2) = 0$. To sample a pair (f,b) can thereby be performed via a series of binary decisions as shown in Figure 1. It should be noted that with this choice of $p^{\varphi}(f,b|x)$ no optimisations need to be run for $(0,0)$-moves. Optimisations are necessary neither to decide whether $(f,b) = (0,0)$ nor to propose from $g(x)$ or to compute $\alpha^{\varphi}(y|x,0,0)$. Likewise, for $(0,1)$- and $(1,0)$-moves, it is sufficient with one optimisation. This property is obviously extremely important for computational efficiency and explains why the proposed algorithm is preferable to the corresponding Peskun optimal alternative.

The Metropolis-Hastings kernel P^{φ} discussed above may be modified in several respects, some of which we briefly discuss in the following. First, the P^{φ} should not be used alone, but be combined with standard Metropolis-Hastings kernels doing small local changes within one mode more efficiently. Second, the $(1,2)$- and $(2,1)$-moves are included to enable jumps between two high $\pi(x)$-density regions which are not reflected in $g(x)$. Such moves can also be indirectly obtained via an acceptance of a $(1,0)$-move followed by acceptance of a $(0,1)$-move. Thus, if it is known that the regions where $g(x)$ provides a good approximation to $\pi(x)$ contain a significant amount of the probability mass under $\pi(x)$, it is natural to simplify the graph in Figure 1 by setting $p^{\varphi}(1|x,2) = p^{\varphi}(2|x,1) = 0$, i.e. defining P^{φ} from $q_0^{\varphi}(y|x)$ and $q_1^{\varphi}(y|x)$ only. Third, if no approximative density $g(x)$ is available, one may define P^{φ} from $q_1^{\varphi}(y|x)$ and $q_2^{\varphi}(y|x)$ only, where the function to be optimised should now be $U(x) = -\ln(h(x))$.

In the Metropolis-Hastings algorithm defined here, any $(0,1)$, $(1,0)$, $(1,2)$ or $(2,1)$-move includes optimisations of $U(x) = \ln(h(x)) - \ln(g(x))$. Thus, regions where $g(x)$ is too low compared to $\pi(x)$ are identified. It is then tempting to use this information to define an improved approximation to $\pi(x)$ and replace the old $g(x)$ by this updated $g(x)$ in later iterations. However, with such a procedure the resulting chain is no longer Markov. Even if φ is sampled independently of everything else, whether or not an optimisation is actually performed depends on the current state

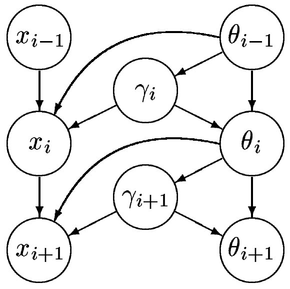


Figure 2: Conditional independence graph for the three sequences $\{x_i\}_{i=0}^{\infty}$, $\{\theta_i\}_{i=0}^{\infty}$ and $\{\gamma_i\}_{i=1}^{\infty}$.

vector x via the probabilities $p^{\varphi}(f,b|x)$. Thereby, an updated $g(x)$ will also depend on the current state and, in turn, the $g(x)$ used in a later iteration will not only depend on the current value of x, but also previously visited states via $g(x)$. Instead of discussing this non-Markovian alternative any further, in the next section we consider a simplified situation which ensures a Markov property also for the adaptive chain.

4 An adaptive Markov chain

In this section we first define a Markov chain with an adaptive structure and evaluates its convergence properties. Thereafter we describe how an adaptation of the mode jumping proposal Metropolis-Hastings algorithm defined in Section 3 can be formulated within this framework.

4.1 Adaptation structure

Let $\{x_i\}_{i=0}^{\infty}$, $\{\theta_i\}_{i=0}^{\infty}$ and $\{\gamma_i\}_{i=1}^{\infty}$ be three stochastic sequences where $x_i \in \Re^n$, $\theta_i \in \Theta$ (for some sample space Θ) and $\gamma_i \in \Re^m$ (for some integer $m > 0$) for all i and assume the distribution of the sequences to have the conditional independence graph given in Figure 2. Thus, the joint distribution of the three sequences is defined by the initial joint distribution for x_0 and θ_0 and three conditional distributions; (i) the conditional distribution for γ_i given θ_{i-1}, (ii) the conditional distribution for θ_i given θ_{i-1} and γ_i, and (iii) the conditional distribution for x_i given θ_{i-1}, γ_i and x_{i-1}. To keep notation simple we also do the following assumptions; Let x_0 have a continuous marginal distribution and denote its density with respect to the Lebesgue measure by $\nu_0(\cdot)$. Assume the conditional distribution (i) to be continuous and denote the conditional density by $\tau(\cdot|\theta)$. Let conditional distribution (iii) by defined by a Metropolis-Hastings transitional kernel as defined in Section 2.2, but where the proposal kernels and probability distributions for the pair (f,b) now may be functions of γ and θ. We use the obvious notation $q_i(\cdot|x,\gamma,\theta)$, $p(f,b|x,\gamma,\theta)$ and $\mathrm{P}(\cdot|x,\gamma,\theta)$. Thus, for each fixed $\gamma \in \Re^m$ and $\theta \in \Theta$, $\mathrm{P}(\cdot|x,\gamma,\theta)$ defines a Markov chain with invariant distribution $\pi(\cdot)$. Moreover, as we still require the proposal distributions to be continuous, x_i has a continuous marginal distribution for each i. The corresponding density we denote by $\nu_i(\cdot)$.

In the above construction, conditional on $\{\theta_i\}_{i=0}^{\infty}$ and $\{\gamma_i\}_{i=1}^{\infty}$, $\{x_i\}_{i=1}^{\infty}$ becomes a Markov chain with (non-stationary) transition kernel $\mathrm{P}(\cdot|x,\gamma,\theta)$. The algorithm can then be interpreted as an adaptive Metropolis-Hastings algorithm. The adaptation of the transition kernel is via the value of θ and, at any iteration, the generation of γ may uncover new information about the target distribution, which can then be used both to propose a new state vector and to update the value of θ. To guarantee convergence of the algorithm to $\pi(x)$, it is not sufficient that $\mathrm{P}(\cdot|x,\gamma,\theta)$ has $\pi(\cdot)$ as invariant distribution for any fixed γ and θ. One must also avoid the adaptation of θ to make the mixing arbitrarily slow. A natural alternative, used in the following theorem, is to include a non-adaptive term in one proposal density.

Theorem 1 *Let the joint distribution for $\{x_i\}_{i=1}^{\infty}$, $\{\theta_i\}_{i=0}^{\infty}$ and $\{\gamma_i\}_{i=1}^{\infty}$ be defined as specified above. Then the following holds:*

(a) If the marginal distribution for x_0 follows $\pi(\cdot)$, then the marginal distribution for x_i follows $\pi(\cdot)$ for all $i \geq 1$.

(b) Let $q_0(\cdot|x,\gamma,\theta)$ be given as

$$q_0(y|x,\gamma,\theta) = c(\gamma,\theta)q_0^*(y|x) + (1 - c(\gamma,\theta))\widetilde{q}_0(y|x,\gamma,\theta), \tag{13}$$

where $c(\gamma,\theta) \in [0,1]$ for all $\gamma \in \Re^m$ and $\theta \in \Theta$, and $q_0^*(\cdot|x)$ and $\widetilde{q}_0(\cdot|x,\gamma,\theta)$ are proposal densities on $\Re^n$. Furthermore, write

$$\beta^* = \inf_{x,y\in\Re^n}\left[\frac{q_0^*(y|x)}{\pi(y)}\right] \quad \textit{and} \quad \kappa^* = \inf_{x\in\Re^n,\theta\in\Theta}\left\{E[p(0,0|x,\gamma,\theta)c(\gamma,\theta)|\theta]\right\}, \tag{14}$$

where the expectation is with respect to conditional distribution (i) above, $\tau(\cdot|\theta)$ Then, if $\beta^* > 0$ and $\kappa* > 0$

$$||\nu_i(\cdot) - \pi(\cdot)|| \leq (1 - \beta^*\kappa^*)^i. \tag{15}$$

The requirement $\beta^* > 0$ ensures geometric convergence when $q_0^*(\cdot|x)$ is used alone and $\kappa^* > 0$ gives that $q_0^*(\cdot|x)$ is used infinitely often. A detailed proof of the theorem is given in Appendix A. Similarly to convergence theorems for other adaptive chains (Gilks et al., 1998; Haario et al.,2001; Sahu and Zhigljavsky, 1999; Holden, 1998), the above result does not guarantee any benefit of the adaptation process, it just prevents convergence and mixing from becoming arbitrarily slow. In fact, the convergence rate guaranteed by the theorem, $1 - \beta^*\kappa^*$, is lower than what $q_0^*(\cdot|x)$ is required to produce alone. In practice, construction of the adaptation process should obviously aim at obtaining faster convergence and better mixing. In the next section we discuss adaptation strategies for the Metropolis-Hastings algorithm defined in Section 3 and we empirically demonstrate improved mixing for two examples in Section 5.

4.2 Adaptation of mode jumping proposals

To generalise the mode jumping proposal Metropolis-Hastings algorithm in Section 3 to an adaptive algorithm, we assume the approximate distribution, $g(x)$, to be parameterised with a parameter $\theta \in \Theta$. Thus, in the algorithm defined in Section 3 we replace $g(x)$ by $g(x|\theta)$. Consequently, all quantities derived from $g(x)$, like $U(x)$, $\mu(\varphi)$, $\Sigma(\varphi)$, $q_i^{\varphi}(y|x)$, $\alpha_{ij}^{\varphi}(y|x)$ and $\mathrm{P}^{\varphi}(A|x)$, also become functions of θ. However, except for $g(x|\theta)$, we retain our notation from Section 3 and thereby suppress the dependence on θ in the notation.

Reconsidering the adaptation strategy briefly discussed in the end of Section 3 and putting it into the adaptive Markov chain framework defined above, we can now better understand what is going wrong. We only want to use the modes found by optimisation starting at φ_1 or φ_2 to update θ if the corresponding optimisations have to be performed to run the Metropolis-Hastings iteration. Thus, dependent on the pair (f,b), we would like to set $\gamma = (\varphi_1,\varphi_2)$, $\gamma = \varphi_1$ or $\gamma = 0$. However, the (f,b) sampled depends on the state vector x, so this alternative violates the conditional independence structure of Figure 2. As an alternative adaptation strategy one could try to set $\gamma = (\varphi_1, u_2)$, where u_2 is the uniform random variate used to decide (f,b) in Figure 1, update θ only if $u_2 > \xi_2$ and when updating θ only use $\mu(\varphi_1)$ and $\Sigma(\varphi_1)$. The event $u_2 > \xi_2$ is independent of x and whenever this event occurs, the optimisation from φ_1 must be run to do the Metropolis-Hastings update. However, also this alternative violates the conditions set for the adaptive Markov chain because the P^{φ} defined in Section 3 does not fulfil detailed balance when conditioned on u_2. To fix this problem, the acceptance probabilities need to be changed.

At any iteration i, let $\gamma_i = (\varphi_1, u_4)$, where u_4 is uniformly sampled on the unit interval, independently of everything else. Define a new transition kernel, $\mathrm{P}(\cdot|x,\gamma,\theta)$ by

$$P(A|x,\gamma,\theta) = \begin{cases} \mathrm{P}_1^{\varphi}(A|x,\gamma,\theta) & \text{if } u_4 \leq \xi_4 \\ \mathrm{P}_2^{\varphi}(A|x,\gamma,\theta) & \text{otherwise,} \end{cases} \tag{16}$$

where $\xi_4 \in (\xi_2, 1)$ and where both P_1^φ and P_2^φ are identical to the P^φ defined in Section 3 and Figure 2, except that ξ_2 is replaced with ξ_2/ξ_4 for P_1^φ and ξ_2 is set equal to zero for P_2^φ. One should note that, when randomised over the value of u_4, the resulting proposal mechanism for $\mathrm{P}(\cdot|x,\gamma,\theta)$ is identical to what is used in Section 3. However, the acceptance probabilities differ (and is lower) because the acceptance probabilities for $\mathrm{P}(\cdot|x,\gamma,\theta)$ is computed conditional on the value of u_4. With this choice for γ and $\mathrm{P}(\cdot|x,\gamma,\theta)$, γ is sampled independently of x and whenever the event $u_4 > \xi_4$ occurs, $\mu(\varphi_1)$ and $\Sigma(\varphi_1)$ must be computed to perform the Markov chain update and can thereby also be used to improve the approximation $g(x|\theta)$ by modifying θ. Moreover, $\mathrm{P}(\cdot|x,\gamma,\theta)$ fulfils detailed balance for each fixed $\gamma \in \Re^{2n+1}$ and $\theta \in \Theta$, as required.

To complete the definition of the adaptive Markov chain, it remains to specify Θ, $g(x|\theta)$ and the updating rule for θ. Different choices are again possible, but we pursue the idea briefly discussed in Section 3 and let $g(x|\theta)$ be a mixture of multivariate t-distributions and have found this to work well in several situations. Appendix B gives the exact updating procedure we have used and show how the resulting simulation algorithm is within the class defined by Theorem 1(b). It should be noted that also the parameters $\xi_0, \xi_1, \ldots, \xi_4$ may be functions of θ and we use this to get more frequent adaptations in the first part of a simulation.

5 Simulation examples

In the first example in this section we let the target distribution be given as a mixture of nine Gaussian distributions on $\Re^2$. The second example revisit a Bayesian model for mixtures.

5.1 Mixture of Gaussian distributions

Let the target distribution be

$$\pi(x) = \sum_{i=1}^{9} w_i \mathrm{N}(\mu_i, \Sigma_i)(x) \quad \text{for } x \in \Re^2, \tag{17}$$

where $\mathrm{N}(\mu,\Sigma)(\cdot)$ denotes the density function of a bivariate normal distribution with mean μ and covariance matrix Σ. Let $\mu_1, \ldots, \mu_9$ be equal to $(0,0)$, $(-0.1,-0.1)$, $(0.1,-0.1)$, $(-0.1,0.1)$, $(0.1,0.1)$, $(-1,-1)$, $(1,-1)$, $(-1,1)$ and $(1,1)$, respectively, set $\Sigma_i = \mathrm{diag}(0.005^2, 0.005^2)$ for $i = 1,6,7,8,9$ and $\Sigma_i = \mathrm{diag}(0.1^2, 0.1^2)$ for $i = 2,3,4,5$, and $w_1 = 1/3$, $w_2 = w_3 = w_4 = w_5 = 1/9$ and $w_6 = w_7 = w_8 = w_9 = 1/18$. Thus, the target distribution has 9 well separated modes. It should be noted how modes 2, 3, 4 and 5 lie around and have larger variances than mode 1, Thereby, mode 1 is hidden by these four modes in the optimisations and reduce the "attraction area" of this mode.

We simulate from the specified target distribution with six different simulation algorithms, the three first of which are within the class defined in Section 3. Algorithm *i*) uses no approximation $g(x)$, i.e. it has $p^\varphi(0|x) = \xi_3 = 0$. Algorithm *ii*) has a $g(x)$ with three of the modes in $\pi(x)$ represented, namely modes number 4, 5 and 7. Algorithm *iii*) uses a $g(x)$ where seven modes are represented, all except modes number 1 and 6. All other parameters are identical for algorithm *i*), *ii*) and *iii*). Ideally, at least parameters ξ_3 and ξ_4 should be tuned according to the quality of the $g(x)$ available, but our focus here is to show that the same parameter values can give reasonable results for approximations of different quality. In practice, the quality of the available $g(x)$ will typically be unknown. The parameter values used are $\xi_0 = 1/100$, $\xi_1 = 1/4$, $\xi_2 = 0.9$, $\xi_3 = 0.5$, $\varepsilon = 0.01$ and $\nu = 10$. In each iteration, φ_1 and φ_2 are sampled independently from a $\mathrm{N}(0, \mathrm{diag}(2^2, 2^2))$ density and for optimisation we use a Newton algorithm with line search. Algorithm *iv*) is an independent proposal Metropolis-Hastings algorithm where new states are proposed from a $\mathrm{N}(0, \mathrm{diag}(2^2, 2^2))$ distribution. Algorithm *v*) uses tempered transitions (Neal, 1996) to propose new values. To define the details of this algorithm, we follows the principles used in the article mentioned. We use 120 temperature levels, $T_i = 1.1^i$ for $i = 1, 2, \ldots, 120$ and on each level we do ten random walk proposals. On temperature level i, the potential new state is generated from the current one by adding a $\mathrm{N}(0, \mathrm{diag}(0.005^2 \cdot T_i, 0.005^2 \cdot T_i))$-variate. Algorithm *vi*) is the adaptive Markov chain algorithm defined in Section 4.2. For ξ_0, ξ_1 and ε we use the same values as for algorithms *i*), *ii*) and *iii*), whereas ξ_2, ξ_3 and ξ_4 are adaptive as specified in Appendix B. For $g_0(x)$ we use a multivariate t-distribution centered at the origin and with 0.5 degrees of freedom, start with $g(x|\theta) = g_0(x)$ and have $\alpha^* = 0.01$. Again we do not claim these parameter values to be optimal in any sense. Our focus is rather to demonstrate the robustness of the proposed algorithms.

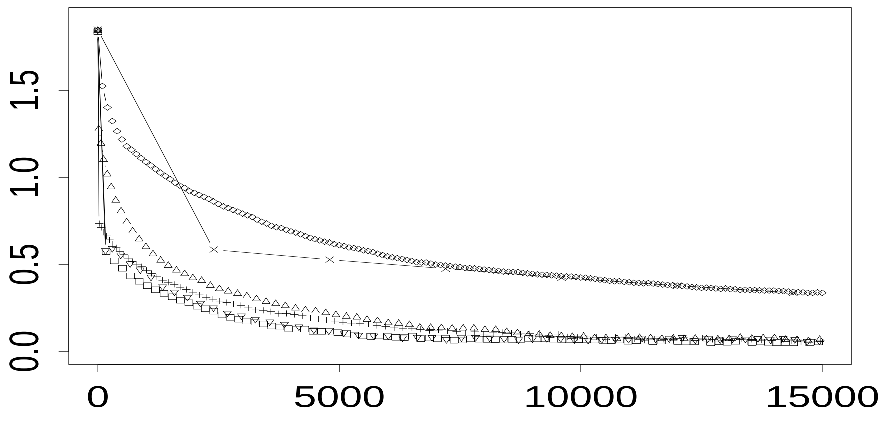
To compare the six algorithms we first run 10,000 runs for each of them, starting from a $\mathrm{N}(0, \mathrm{diag}(2^2, 2^2))$ density, and estimate the convergence by dividing $\Re^2$ into 15^2 regions and calculating, as function of iteration number,

$$d = \sum_{j=1}^{15^2} |\widehat{\pi}_j - \pi_j|, \tag{18}$$

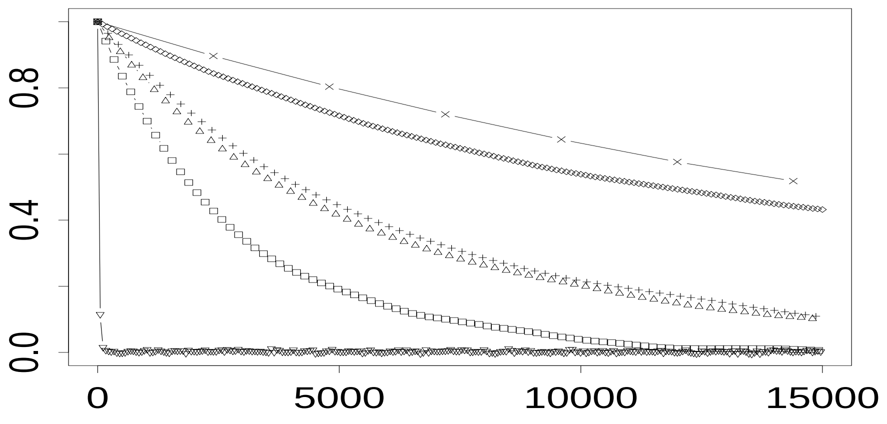
where $\widehat{\pi}_j$ is the fraction of the 10,000 states within region number j and π_j is the corresponding correct probability (found by numerical integration). The 15^2 regions used are defined by dividing each of the two coordinate directions at -1.005, -0.995, -0.55, -0.11, -0.09, -0.05, -0.005, 0.005, 0.05, 0.09, 0.11, 0.55, 0.995 and 1.005. For a fair comparison of the algorithms one must take into account the computer resources required for each of them. For typical target densities where MCMC are used, evaluation of $h(x)$ will dominate the cpu time of our algorithms. Thus, even if evaluation of $g(x)$ also contributes significantly to the computation time for our very simple target distribution, we use number of evaluations of $h(x)$ (or first or second derivatives) to measure the computer resources for each algorithm. The mean number of target evaluations varies with iteration number, but are around *i*) 170, *ii*) 215, *iii*) 186, *iv*) 2286, *v*) 1 and *vi*) 192. For algorithm *vi*) the given figure is only representative for the beginning of a simulation. As the number of iterations increases and $g(x)$ stabilises, the number reduces to about 57 evaluations per iteration. Figure 3(a) shows the estimated convergences as function of the (mean) number of evaluations of $h(x)$. We observe that algorithms *iv*) and *v*) converge much slower than the others. For the independent proposal algorithm, this should come as no surprise. The slow convergence of the tempered transition algorithm is because of the very high number of target evaluations per iteration. Measured in number of iterations, the tempered transition algorithm has far the best convergence. The difference in convergence between algorithms *i*), *ii*), *iii*) and *vi*) is quite small, with algorithm *i*) in fact being the best. For more realistic target densities we do not expect algorithm *i*) to have the best convergence. For our very simple target density, $\ln(h(x))$ is essentially quadratic close to each of the modes and this give a fast convergence of the optimisations, which will not be generally true.

To evaluate also the mixing properties after convergence we run one more (long) run, where we initiate the chains correctly according to the target density and estimate the resulting autocorrelation functions for the indicator function $I(||x|| \leq 0.025^2)$. Again we scale the results corresponding to the mean number of target density evaluations used, see Figure 3(b). For algorithm *vi*), we only used the last portion of the run, after $g(x)$ seemed to have stabilised, for the estimation. It should be noted that the mode centered at the origin is included in $g(x)$ for none of the algorithms used. After convergence the adaptive algorithm use a $g(x)$ where all 9 modes are well represented and the autocorrelation of this algorithm becomes far better than the others. Again algorithm *i*) is the best of the non-adaptive algorithms and we believe the explanation to be as discussed above.

To get a better understanding of how the algorithms work it is also of interest to monitor the fraction of different moves used for the different algorithms and the corresponding acceptance probabilities. Table 1 gives these figures for algorithms *i*), *ii*), *iii*) and *vi*), where the figures for the adaptive algorithm are after $g(x)$ has stabilised. Algorithms *iv*) and *v*) have acceptance rates of 0.014% and 29%, respectively. From the figures in Table 1, one can observe how the fraction of $(0,0)$-moves increases as the approximation $g(x)$ improves. The much higher fraction of $(1,2)$-moves relative to $(2,1)$-moves is a consequence of our choice of $p^{\varphi}(f, b|x)$. Whenever no other good alternative can be found, we choose a $(1,2)$-move with high probability. This also explains the



(a)



(b)

Figure 3: For each of the six simulation algorithms: (a) estimated convergence and (b) estimated autocorrelation function after convergence. The horizontal axis is scaled and shows the (mean) number of evaluations of the target density. The symbols used are $\square$, $\triangle$, $+$, $\times$, $\Diamond$ and $\triangledown$ for algorithms $i)$ to $vi)$, respectively.

(f, b)	without $g(x)$	$g(x)$ with three modes	$g(x)$ with seven modes	adaptive algorithm
$(0, 0)$	—	25.3% (90.0%)	53.3% (91.6%)	95.9% (92.3%)
$(0, 1)$	—	4.9% (33.5%)	3.6% (25.1%)	0.01% (50.0%)
$(1, 0)$	—	2.8% (58.8%)	5.7% (15.2%)	2.0% (30.0%)
$(1, 2)$	89.1% (7.5%)	61.9% (2.8%)	33.8% (0.4%)	1.7% (10.0%)
$(2, 1)$	10.9% (61.1%)	5.1% (32.9%)	3.6% (5.4%)	0.3% (0.0%)

Table 1: Fraction of (f, b)-moves in algorithms $i)$, $ii)$, $iii)$ and $vi)$ and corresponding acceptance rates in parenthesis.

quite low acceptance rates for $(1,2)$-moves. The reduced acceptance rates for $(2,1)$-moves when the quality of $g(x)$ improve can also be understood. The modes in the approximation $g(x)$ do not fit exactly with the modes in $h(x)$ and in $U(x)$ this results in small local minima. As more modes are included in $g(x)$, $U(x)$ contains more such "fake" minima and in turn increase the rejection rate for $(2,1)$-moves.

5.2 Bayesian analysis of mixtures

In this section we study the performance of the simulation algorithms for the Bayesian model for mixtures defined in Richardson and Green (1997). We present simulation results for the acidicity data set used in this article, see also Crawford et al. (1992) and Crawford (1994). The data set concerns an acidity index measured in 155 lakes in Wisconsin. We limit our attention to a model with a fixed number of components in the mixture, whereas Richardson and Green (1997) consider the more general problem when also the number of components is stochastic. Except for the fixed number of components, we adopt the exact same model as in Richardson and Green (1997), including values for hyper-parameters. Therefore, we give here only a very brief description of the model and refer to the article mentioned for a detailed presentation and discussion of the model.

The data set consists of $N = 155$ observations, $y_1, \ldots, y_N$, which in the model are assumed to be generated independently from a K-terms Gaussian mixture,

$$y_i|\{w_k, \mu_k, \sigma_k^2\}_{k=1}^K \sim \sum_{k=1}^K w_k \mathrm{N}(\mu_k, \sigma_k^2)(y_i), \tag{19}$$

where $\sum_{k=1}^K w_k = 1$. The prior for $\{w_k, \mu_k, \sigma_k^2\}_{k=1}^K$ is defined by independent priors for the three groups $(w_1, \ldots, w_K)$, $(\mu_1, \ldots, \mu_K)$ and $(\sigma_1^2, \ldots, \sigma_K^2)$. The weights $(w_1, \ldots, w_K)$ are assign a symmetric Dirichlet distribution, $\mathrm{D}(1, \ldots, 1)$. The prior distribution for $\mu_1, \ldots, \mu_K$ assumes them to be independently sampled from a $\mathrm{N}(\xi, \kappa^{-1})$ density, restricted to the ordering constraint $\mu_1 < \mu_2 < \ldots < \mu_K$. The hyper-parameters ξ and κ are fixed and their values are chosen to give a rather flat prior over the interval of the observed values. The $\sigma_1^2, \ldots, \sigma_K^2$ are assigned independent $\Gamma(2, \beta)$-priors, where β is again assigned a gamma distribution.

Our simulation algorithms with optimisations and t-proposals are most easily used when the sample space is all of $\Re^n$, i.e. without restrictions on legal values. We therefore define a reparameterisation of the model. For $w_1, \ldots, w_K$ we use the logistic transformation, $v_i = \ln(w_i) - \ln(w_K)$ for $i = 1, \ldots, K-1$ and transform $\mu_1, \ldots, \mu_K$ by setting $\mu_1 = m_1$ and $\mu_i = \mu_{i-1} + \exp\{m_i\}$ for $i = 2, \ldots, K$. Finally, for $\sigma_1^2, \ldots, \sigma_K^2$ and β we use a log-transformation and set $\tau_i = \ln(\sigma_i^2)$ for $i = 1, \ldots, K$ and $b = \ln(\beta)$. Thus, the vector to simulate is $x = (v_1, \ldots, v_{K-1}, m_1, \ldots, m_K, \tau_1, \ldots, \tau_K, b)$ and its posterior density is strictly positive in all of $\Re^{3K}$.

We compare simulation results with $K = 3$ for six different simulation algorithms. The two first are within the class defined in Section 3. Algorithm *i*) has no approximation $g(\cdot)$, i.e. $p^\varphi(0|x) = \xi_3 = 0$, whereas algorithm *ii*) has a $g(\cdot)$ which is generated via five optimisations and corresponding adaptations of θ as defined in Appendix B. The resulting $g(\cdot)$ is as given by equation (20) with $i = 2$. For both algorithms *i*) and *ii*), φ_1 and φ_2 are sampled independently from a Gaussian distribution centered at $(0,0,5,0,0,0,0,0,0)$ and with covariance matrix $\mathrm{diag}(2^2, 2^2, 3^2, 2^2, 2^2, 2^2, 2^2, 2^2, 2^2)$. These variances are larger than we expect necessary, but are chosen to reduce the risk of missing some posterior modes. Note that the third component, $m_1 = \mu_1$, is assigned a different mean and variance from the rest because the reparameterisation leaves this element unchanged. The remaining algorithmic parameters are identical to the one used for algorithm *i*), *ii*) and *iii*) in Section 5.1. Algorithm *iii*) is an independent proposal Metropolis-Hastings algorithm, where potential new states are generated from the same Gaussian distribution used to sample φ_1 and φ_2 for algorithms *i*) and *ii*). Algorithm *iv*) is a random walk proposal Metropolis-Hastings algorithm where potential new states are produced by adding a $\mathrm{N}(0, 0.05^2)$-variate to each element of the state vector. Algorithm *v*) is an adaptive Markov chain algorithm defined as in Section 4.2, where

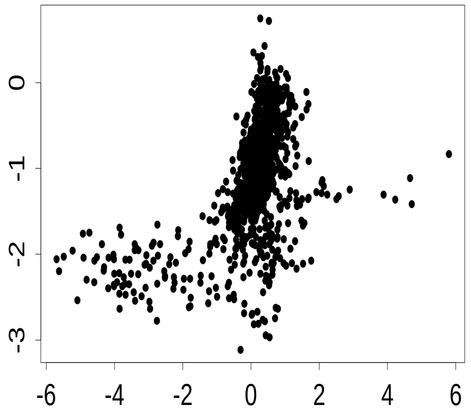
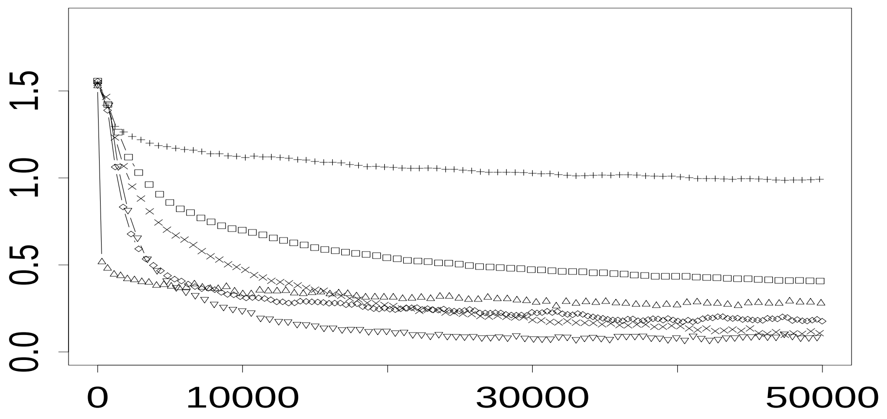


Figure 4: Biplot with 1,000 (independent) pairs (v_1, τ_2) simulated via one very long run with algorithm *vi*)
.

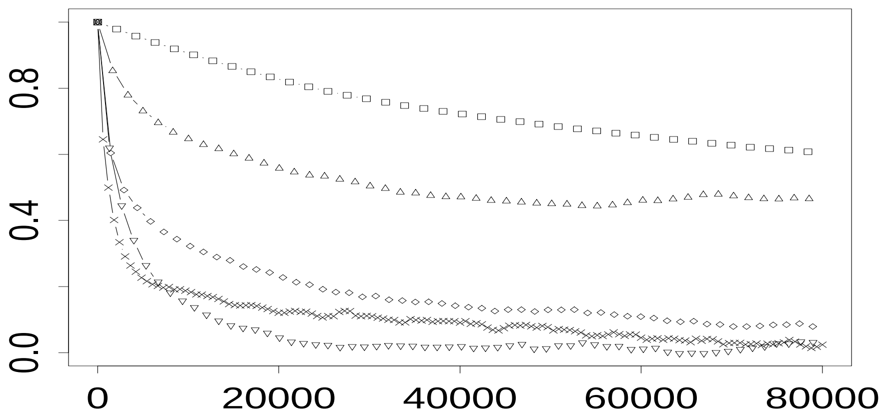
φ_1 and φ_2 are generated as for algorithms *i*) and *ii*) and the remaining parameters are identical to the values used for the adaptive algorithm in Section 5.1. Finally, algorithm *vi*) is a mixture of algorithms *iv*) and *v*), where each iteration consists of one (adaptive) update followed by 100 random walk proposal updates. We also tried to use tempered transitions to generate potential new states, but did not succeed in getting this to work satisfactory and therefore omit this algorithm from the simulation study.

To compare the five algorithms we first run 5,000 runs for each of them, starting from a $\mathrm{N}((0,0,5,0,0,0,0,0,0), \mathrm{diag}(2^2,2^2,3^2,2^2,2^2,2^2,2^2,2^2,2^2))$ density. Preliminary runs indicated v_1 and τ_2 to be two of the elements with the longest autocorrelations ranges and we therefore estimate focus on the bivariate distribution for these two elements when estimating convergence. We define $13 \times 11 = 143$ regions in the (v_1, τ_2)-plane by dividing at $-4 + z; z = 0, 1, \ldots, 11$ for v_1 and for $-2.5 + z/2; z = 0, 1, \ldots, 9$ for τ_2, and estimate the convergence with a formula similar to the one in equation (18). However, computation of π_j by numerical integration is now infeasible and we instead estimate π_j from one very long run with algorithm *vi*). A biplot of 1,000 simulated pairs (v_1, τ_2) are shown in Figure 4. For each of the six algorithms used, Figure 5(a) shows the estimated convergences as function of the (mean) number of density evaluations. We observe that the random walk proposal algorithm, *v*), performs quite well, but have an overall slower convergence than the mixture algorithm, *vi*). A closer study of the optimisations in algorithm *i*) shows several local maxima for the target distribution. Thus, to be able to explain the merits of algorithms *v*) and *vi*), the local maxima must be separated with areas with a density low enough to make a hinder for the movement of the random walk proposal algorithm, but the density in the low density areas is not sufficiently low to totally block the motion between modes in algorithm *iv*). The quite slow convergence of algorithms *i*) and *ii*) should also be noted, even if we expect somewhat better performances also here if we combine them with local walk proposals.

To evaluate the mixing properties after convergence we again run one long run for each of the six algorithms and estimate the autocorrelation for τ_2, see Figure 5(b). To initiate these runs, we start with 1,000 iterations of algorithm *vi*). The independent proposal algorithm had no accepted moves in 50,000 iterations and is therefore omitted in Figure 5(b). The mean number of density evaluations, used to scale the autocorrelation functions plotted, are *i*) 706, *ii*) 557, *iii*) 1, *iv*) 1, *v*) 243 and *vi*) 446. It should be noted that for the adaptive algorithms these numbers will vary from one run to another, because one will end up with different approximations $g(x)$ in different runs. This explains why the difference between the figures for algorithms *v*) and *vi*) is not one hundred



(a)



(b)

Figure 5: For each of the six simulation algorithms: (a) estimated convergence and (b) estimated autocorrelation function after convergence. The horizontal axis is scaled and shows the (mean) number of evaluations of the target density. The symbols used are □, △, +, ×, ◊ and ▽ for algorithms *i*) to *vi*), respectively.

(f,b)	without $g(x)$	with $g(x)$ non-adaptive	adaptive algorithm	mixture algorithm
$(0,0)$	—	14.8% (30.9%)	66.9% (31.7%)	52.5% (32.1%)
$(0,1)$	—	6.2% (1.6%)	0.2% (6.8%)	0.3% (2.0%)
$(1,0)$	—	9.7% (1.0%)	1.4% (1.0%)	1.1% (0.9%)
$(1,2)$	80.9% (0.5%)	59.7% (0.007%)	27.2% (0.1%)	39.2% (0.1%)
$(2,1)$	19.1% (2.2%)	6.0% (0.2%)	3.4% (0.8%)	5.1% (0.7%)

Table 2: Fraction of (f,b)-moves in algorithms *i*), *ii*), *iii*) and *vi*) and corresponding acceptance rates in parenthesis.

as one should expect. In Figure 5(b) we can observe how the autocorrelation function for the random walk proposal algorithm drops quickly for small lags, indicating good mixing properties within each mode, but also has small positive correlations for quite large lags, induced by the low density hinders between modes. Combining random walk proposals with the adaptive moves gives an algorithm which both jumps efficiently between modes and have good mixing properties within each mode. Table 2 gives fractions of different moves and corresponding acceptance probabilities for algorithms *i*), *ii*), *v*) and *vi*), from which we also can see that our runs for algorithms *v*) and *vi*) end up with different $g(x)$. Acceptance rates for random walk proposals were 40.5% for both algorithms *iv*) and *vi*).

6 Closing remarks

This paper discusses how local optimisation can be included in a Metropolis-Hastings algorithm to produce better mixing. Relative to most other proposal mechanisms commonly used in MCMC, optimisations are computer intensive and its use in Metropolis-Hastings can not be justified unless the target density is particularly difficult to sample from. Multi-modal distributions is the typical example of such a situation. We have discussed both a non-adaptive and an adaptive Metropolis-Hastings algorithm. As briefly discussed in Section 1, Gelman and Rubin (1992) has previously suggested to approximate $\pi(\cdot)$ with a density $g(\cdot)$ based on a number of prior optimisation runs and thereafter use $g(\cdot)$ as a proposal mechanism in Metropolis-Hastings. Our non-adaptive algorithm can be seen as a robustification of this procedure. It should be noted that even this non-adaptive Metropolis-Hastings algorithm has a flavour of adaptiveness, in that it automatically adjust the fraction fast $(0,0)$-moves to the quality of the approximation $g(\cdot)$ available. The examples in Section 5 clearly demonstrates this effect. The examples also demonstrates the importance of measuring convergence and mixing in terms of computer time and not number of iterations. Algorithms that are clearly Peskun sub-optimal can still be the best in simulation time.

The effect of including local optimisation in Metropolis-Hastings is most clear when the target density has several modes separated with very low density areas which effectively acts as barriers when using local moves only. Our example in Section 5.1 falls within this class and it is there unnecessary to include a random walk proposal algorithm in the simulation study as it is obvious that this will not be able to move between modes. However, the example in Section 5.2 shows that to include optimisation based moves can also give significantly better convergence and mixing properties when the density between the modes is not low enough to totally block the motion via local moves.

In addition to the examples presented, we have also used the algorithms for the model in Section 5.2 for other values of K and for two other data sets. The behaviour is essentially the same for all three data sets. Our proposed algorithms works satisfactory when K is low enough, but when K becomes large the acceptance rates for the mode jumping proposals becomes very low and the adaptive algorithm is not able to obtain a good approximation to the target density. We believe this is because the posterior becomes almost degenerate for large K's, so that approximation to it by t-densities is no longer possible.

A requirement for the proposed optimisation based algorithms is that we have available a prior approximation $g(x)$ (or $g_0(x)$ in the adaptive algorithm), for which $\inf[g(x)/h(x)] > 0$. In practice, we do not expect this to be a problem. If this requirement is not fulfilled this will quickly be revealed when starting to simulate, as then some optimisations will return $U(x)$ equal to infinity.

Appendix A

This appendix contains a proof for Theorem 1. By assumption, $\mathrm{P}(\cdot|x,\gamma,\theta)$ fulfils detailed balance and thereby has $\pi(\cdot)$ as an invariant distribution for any $\gamma \in \Re^m$ and $\theta \in \Theta$. Thus, result (a) follows by induction.

To prove (b), we start to consider the conditional transition kernel for x given γ and θ. Let $\mathrm{P}_{00}(\cdot|x,\gamma,\theta)$ be the transition kernel corresponding to a $(0,0)$-move. Our assumptions for $q_0(\cdot|x,\gamma,\theta)$ gives

$$\frac{q_0(y|x,\gamma,\theta)}{\pi(y)} = c(\gamma,\theta)\frac{q_0^*(y|x)}{\pi(y)} + (1-c(\gamma,\theta))\frac{\widetilde{q}_0(y|x,\gamma,\theta)}{\pi(y)} \geq \beta c(\gamma,\theta)$$

for all $x, y \in \Re^n$, $\gamma \in \Re^m$ and $\theta \in \Theta$. Thus, $q_0(\cdot|x,\gamma,\theta)$ can be rewritten as

$$q_0(y|x,\gamma,\theta) = \beta c(\gamma,\theta)\pi(y) + (1-\beta c(\gamma,\theta))\widehat{q}_0(y|x,\gamma,\theta)$$

for some density function $\widehat{q}_0(\cdot|x,\gamma,\theta)$. In obvious notation, this gives

$$\mathrm{P}_{00}(A|x,\gamma,\theta) = \int_A q_0(y|x,\gamma,\theta)\alpha(y|x,\gamma,\theta,0,0)\mathrm{d}y + I(x\in A)r(x|\gamma,\theta,0,0)$$

$$= \int_A \frac{1}{\pi(x)}\min\{\pi(x)(\beta c(\gamma,\theta)\pi(y) + (1-\beta c(\gamma,\theta))\widehat{q}_0(y|x,\gamma,\theta)),$$

$$\pi(y)(\beta c(\gamma,\theta)\pi(x) + (1-\beta c(\gamma,\theta))\widehat{q}_0(y|x,\gamma,\theta))\}\mathrm{d}y + I(x\in A)r(x|\gamma,\theta,0,0)$$

$$\geq \int_A \frac{1}{\pi(x)}\beta c(\gamma,\theta)\pi(x)\pi(y)\mathrm{d}y = \beta c(\gamma,\theta)\int_A \pi(y)\mathrm{d}y$$

Thus, for $\mathrm{P}(\cdot|x,\gamma,\theta)$ one gets

$$\mathrm{P}(A|x,\gamma,\theta) \geq p(0,0|x,\gamma,\theta)\beta c(\gamma,\theta)\int_A \pi(y)\mathrm{d}y,$$

which, when integrating out γ, gives

$$\mathrm{P}(A|x,\theta) = \int_{\Re^m} \mathrm{P}(A|x,\gamma,\theta)\tau(\gamma|\theta)\mathrm{d}\gamma$$

$$\geq \beta\ \mathrm{E}[p(0,0|x,\gamma,\theta)c(\gamma,\theta)|\theta]\int_A \pi(y)\mathrm{d}y \geq \beta \inf_{x\in\Re^n}\{\mathrm{E}[p(0,0|x,\gamma,\theta)c(\gamma,\theta)|\theta]\}\int_A \pi(y)\mathrm{d}y,$$

where the expectations are with respect to the conditional distribution for γ given θ, $\tau(\cdot|\theta)$. This implies that $\mathrm{P}(\cdot|x,\theta)$ can be expressed as

$$\mathrm{P}(A|x,\theta) = \beta\kappa(\theta)\int_A \pi(y)\mathrm{d}y + (1-\beta\kappa(\theta))\widetilde{P}(A|x,\theta),$$

where $\kappa(\theta) = \inf_{x\in\Re^n}\{\mathrm{E}[p(0,0|x,\gamma,\theta)c(\gamma,\theta)|\theta]\}$ and $\widetilde{P}(\cdot|x,\theta)$ is some transition kernel on $\Re^n$.

Now we consider the convergence of the x chain conditioned on the θ sequence. Thus, let $\boldsymbol{\theta} = \{\theta_i\}_{i=0}^{\infty}$ and let $\nu_i(\cdot|\boldsymbol{\theta})$ denote the conditional density of x_i given $\boldsymbol{\theta}$. Using the above, it then follows

$$||\nu_{i+1}(\cdot|\boldsymbol{\theta}) - \pi(\cdot)|| = \left|\left|\beta\kappa(\theta_i)\pi(\cdot) + (1-\beta\kappa(\theta_i))\int_{\Re^n}\widetilde{P}(\cdot|x,\theta_i)\nu_i(x|\boldsymbol{\theta})\mathrm{d}x - \pi(\cdot)\right|\right|$$

$$= (1-\beta\kappa(\theta_i))\left|\left|\int_{\Re^n}\widetilde{P}(\cdot|x,\theta_i)(\nu_i(x|\boldsymbol{\theta}) - \pi(x))\mathrm{d}x\right|\right| \leq (1-\beta\kappa(\theta_i))\,||\nu_i(\cdot|\boldsymbol{\theta}) - \pi(\cdot)||\,.$$

Thus, by induction on i it follows

$$||\nu_i(\cdot|\boldsymbol{\theta}) - \pi(\cdot)|| \leq \prod_{j=1}^{i}\left[1-\beta\kappa(\theta_j)\right].$$

The last step is to integrate out each θ_j in turn. To integrate over all $\theta_j, j \geq i$ is immediate as the independence graph in Figure 2 gives $\nu_i(\cdot|\boldsymbol{\theta}_{i-1}) = \nu_i(\cdot|\boldsymbol{\theta})$, where $\boldsymbol{\theta}_i = \{\theta_j\}_{j=0}^{i}$. Thus,

$$||\nu_i(\cdot|\boldsymbol{\theta}_{i-1}) - \pi(\cdot)|| \leq \prod_{j=1}^{i}\left[1-\beta\kappa(\theta_j)\right].$$

Next, integrating out θ_{i-1} gives

$$||\nu_i(\cdot|\boldsymbol{\theta}_{i-2}) - \pi(\cdot)|| \leq \prod_{j=1}^{i-1}\left[1-\beta\kappa(\theta_j)\right](1-\beta\inf_{\theta\in\Theta}\{\kappa(\theta)\}).$$

Continuing like this, integrating out θ_{j-1} for $j = i-1, i-2, \ldots, 1$ one gets

$$||\nu_i(\cdot) - \pi(\cdot)|| \leq (1-\beta\inf_{\theta\in\Theta}\{\kappa(\theta)\})^i$$

as required.

Appendix B

In this appendix we specify the precise adaptation procedure we have used in the simulation examples. Let $\Theta = \cup_{i=0}^{\infty}\Theta_i$, where $\Theta_0 = \{(k,\xi_2,\xi_3,\xi_4)|k>0,\xi_2,\xi_3\in(0,1),\xi_4\in(\xi_2,1)\}$ and

$$\Theta_i = \big\{(k,\xi_2,\xi_3,\xi_4,\alpha_0,\{\alpha_j,\mu_j,\Sigma_j\}_{j=1}^{i})\,|k>0,\xi_2,\xi_3\in(0,1),\xi_4\in(\xi_2,1),\alpha_0\in(\alpha^*,1],\alpha_j\in(0,1]$$
$$\text{for } j=1,2,\ldots,i,\alpha_0+\ldots+\alpha_i=1,\mu_j\in\Re^n,\Sigma_j\in\Re^{n\times n}\ \text{ positive definite for } j=1,2,\ldots,i\big\},$$

for $i = 1,2,\ldots$. The k in θ_i is the one used to define $p^{\varphi}(0|x)$ in equation (11), ξ_2,ξ_3 and ξ_4 are the parameters used to define $\mathrm{P}(\cdot|x,\gamma,\theta)$ in equation (16) and $\alpha^*>0$ is a fixed parameter. We let $g(x|\theta)$ be given as

$$g(x|\theta) = \begin{cases} g_0(x) & \text{if } i=0, \\ \alpha_0 g_0(x) + \sum_{j=1}^{i}\alpha_j t_\nu(\mu_j,\Sigma_j)(x) & \text{otherwise,}\end{cases} \tag{20}$$

where $\nu > 0$ is fixed parameter and $g_0(\cdot)$ is a fixed density function on $\Re^n$, for which we require

$$\inf_{x\in\Re^n}\left[\frac{g_0(x)}{\pi(x)}\right] > 0. \tag{21}$$

Whenever $u_4 > \xi_4$ in a Metropolis-Hastings update, $\mu(\varphi_1)$ and $\Sigma(\varphi_1)$ can be used to update θ. We do this by first increasing i by one and setting $\mu_i = \mu(\varphi_1)$ and $\Sigma_i = \Sigma(\varphi_1)$. Thereafter, to avoid i growing to large we reduce i again if this seems not to worsen the approximation too much. Our exact algorithm to update θ is

1. Set $i := i + 1$ and thereafter $\mu_i = \mu(\varphi_1)$ and $\Sigma_i = \Sigma(\varphi_1)$.

2. Set
$$\boldsymbol{\alpha} := \underset{\boldsymbol{\alpha}|\alpha_0 \geq \alpha^*}{\arg\min} \left[\min_{j=1,\dots,i} \left[\ln(g(\mu_j|\theta)) - \ln(h(\mu_j)) \right] \right],$$
where $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \dots, \alpha_i)$. Remember that $\boldsymbol{\alpha}$ is included in θ, so $g(x|\theta)$ is a function of $\boldsymbol{\alpha}$. Furthermore, let l be the minimum value attained, i.e. $l = \min_{j=1,\dots,i} \left[\ln(g(\mu_j|\theta)) - \ln(h(\mu_j)) \right]$.

3. Check if i can be reduced via the following algorithm

    (a) for $j = 1, \dots, i$, compute
    $$l(j) = \underset{\boldsymbol{\alpha}|\alpha_0 \geq \alpha^*, \alpha_j = 0}{\operatorname{argmin}} \left[\min_{j=1,\dots,i} \left[\ln(g(\mu_j|\theta)) - \ln(h(\mu_j)) \right] \right],$$
    and $j^* = \operatorname{argmin}_{j=1,\dots,i}[l(j)]$.

    (b) if $l(j^*) \leq l + 0.01$:

    i. set $i := i - 1$

    ii. for $j = i, i-1, \dots, j^*$, set $\mu_j := \mu_{j+1}$ and $\Sigma_j := \Sigma_{j+1}$

    iii. goto 2

    (c) compute $j^* = \operatorname{argmin}_{j=1,\dots,i}[\alpha_j]$

    (d) if $\alpha_{j^*} \leq 0.001$:

    i. set $i := i - 1$

    ii. for $j = i, i-1, \dots, j^*$, set $\mu_j := \mu_{j+1}$ and $\Sigma_j := \Sigma_{j+1}$

    iii. goto 2

4. To generate a new value for k, first generate $z_1, \dots, z_s$ independently from $g(\cdot|\theta)$ and thereafter set $k = \operatorname{median}_{j=1,\dots,s}[h(z_j)/g(z_j)]$.

5. If the resulting μ_j and Σ_j's are unchanged by the above steps, set $\xi_2 := \min\{\xi_2 + (1 - \xi_2)/10, 0.98\}$ and otherwise set $\xi_2 := \max\{\xi_2 - \xi_2/3, 0.02\}$. Finally, set $\xi_3 := \min\{5 \cdot (1 - \xi_2)/(1 + \xi_2), 0.98\}$ and $\xi_4 := \xi_2 + (1 - \xi_2)/2$.

It should be noted that this updating procedure ensures that $\alpha_0 \geq \alpha^*$ and $\xi_2 \leq 0.02$ as required above. Thus, our adaptive Markov chain is of the type specified in Theorem 1(b) with $q_0^*(y|x) = g_0(y)$,

$$c(\gamma, \theta) = \begin{cases} 0 & \text{if } u_4 > \xi_4, \\ 1 & \text{if } u_4 \leq \xi_4 \text{ and } i = 0, \\ \alpha_0 & \text{otherwise} \end{cases} \tag{22}$$

and

$$p(0,0|x,\gamma,\theta) = \xi_4 \cdot p^\varphi(0|x) \cdot \frac{\xi_2}{\xi_4} = \xi_4 p^\varphi(0|x). \tag{23}$$

For β^*, κ^* defined in Theorem 1 this gives $\beta^* > 0$ by equation (21) and

$$\kappa^* = 0.02\varepsilon\alpha^*, \tag{24}$$

where we have used that $p^\varphi(0|x) \geq \varepsilon$ for all $x \in \Re^n$ and $\theta \in \Theta$.

References

Besag, J., Green, P., Higdon, D. and Mengersen, K. (1995). Spatial statistics and Bayesian computation, *Statistical Science* **10**: 3–66.

Chib, S. and Greenberg, E. (1994). Bayes inference in regression models with arma (p,q) errors, *Journal of Econometrics* **64**: 183–206.

Crawford, S. (1994). An application of the Laplace method to finite mixture distributions, *Journal of the American Statistical Association* **89**: 259–267.

Crawford, S., DeGroot, M., Kadane, J. and Small, M. (1992). Modeling lake chemistry distributions: approximate Bayesian methods for estimating a finite mixture model, *Technometrics* **34**: 441–453.

Gelman, A. and Rubin, D. (1992). Inference from iterative simulation using multiple sequences (with discussion), *Statistical Science* **7**: 457–472.

Gilks, W. R., Richardson, S. and Spiegelhalter, D. J. (1996). *Markov Chain Monte Carlo in practice*, London: Chapman & Hall.

Gilks, W. R., Roberts, G. O. and Sahu, S. K. (1998). Adaptive Markov chain Monte Carlo through regeneration, *Journal of the American Statistical Association* **93**: 1045–1054.

Haario, H., Saksman, E. and Tamminen, J. (2001). An adaptive Metropolis algorithm, *Bernoulli* **7**(2): 223–242.

Hastings, W. K. (1970). Monte Carlo simulation methods using Markov chains and their applications, *Biometrica* **57**: 97–109.

Holden, L. (1998). Adaptive chains, *SAND/11/98*, Norwegian Computing Center, Oslo, Norway. Available from http://www.statslab.cam.ac.uk/~mcmc/.

Mengersen, K. L. and Tweedie, R. L. (1996). Rates of convergence of the Hastings and Metropolis algorithms, *The Annals of Statistics* **24**: 101–121.

Mykland, P., Tierney, L. and Yu, B. (1995). Regeneration in Markov chain samplers, *Journal of the American Statistical Association* **90**: 233–241.

Neal, R. M. (1996). Sampling from multimodal distributions using tempered transitions, *Statistics and Computing* **6**: 353–366.

Peskun, P. H. (1973). Optimum Monte-Carlo sampling using Markov chains, *Biometrica* **60**: 607–612.

Richardson, S. and Green, P. (1997). On Bayesian analysis of mixtures with an unknown number of components (with discussion), *Journal of the Royal Statistical Society, Series B* **59**: 731–791.

Sahu, S. K. and Zhigljavsky, A. A. (1999). Self regenerative Markov chain Monte Carlo with adaptation, *Technical report*, Faculty of Mathematical Studies, University of Southampton, UK. Available from http://www.maths.soton.ac.uk/staff/Sahu/.

Tjelmeland, H. and Hegstad, B. K. (2001). Mode jumping proposals in MCMC, *Scandinavian Journal of Statistics* **28**: 205–223.