Population-based reversible jump Markov chain Monte Carlo

By AJAY JASRA, DAVID A. STEPHENS

Department of Mathematics, Imperial College London, SW7 2AZ, London, UK

ajay.jasra@imperial.ac.uk d.stephens@imperial.ac.uk

AND CHRISTOPHER C. HOLMES

Department of Statistics, University of Oxford, OX1 3TG, Oxford, UK

cholmes@stats.ox.ac.uk

Summary

In this paper we present an extension of population-based Markov chain Monte Carlo (MCMC) to the variable dimension case. One of the main challenges in MCMC based inference is that of simulating from high and trans-dimensional target measures, where standard reversible jump MCMC methods do not adequately traverse the support of the target. We develop population methods to overcome these difficulties, and give a result proving the uniform ergodicity of these population algorithms. We use this result to demonstrate the superiority in terms of convergence rate of the transition kernel over a reversible jump sampler for a Bayesian variable selection problem. We also give an example of a population algorithm for a Bayesian multivariate mixture model with an unknown number of components. We apply the model to a gene expression data set of size 1000 points in 6 dimensions and show that our algorithm out performs some competing Markov chain samplers.

Some key words: Reversible jump Markov chain Monte Carlo, Uniform ergodicity, Bayesian variable selection, Mixture models

1. Introduction

The Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970) and its adaptation to the trans-dimensional case (Green, 1995) has provided a general framework to perform statistical inference for complex target probability measures $\pi(x)\lambda(\mathrm{d}x)$, on

measure space $(\mathsf{X}, \mathcal{B}(\mathsf{X}), \lambda)$, using (with $\mathcal{B}(\mathsf{X})$ denoting a countably generated $\sigma-$algebra and λ a $\sigma-$finite measure on X) a Markov transition kernel K. In this article we focus on reversible jump Markov chain Monte Carlo (RJMCMC) for state space $\mathsf{X} = \bigcup_{k\in\mathcal{K}} \left(\{k\} \times \mathsf{X}_k\right)$, $\mathcal{K} \subseteq \mathbb{N}$, $\mathsf{X}_k \subseteq \mathbb{R}^k$. Recent applications of RJMCMC include classification and regression (Denison et al., 2002) and mixture modelling (Richardson and Green, 1997), with particular emphasis on model determination. However, it is generally the case, for multimodal, variable dimension distributions that naive (vanilla) samplers entirely fail to move around the support of the target; see Brooks et al. (2003) for example.

To deal with these problems, several MCMC approaches have been suggested, including: simulated tempering (Brooks et al., 2002), auxiliary variable methods (Brooks et al., 2003) and tempered transitions (Jennison et al., 2003); see Green (2003b) for an up-to-date review. Methods other than MCMC, that may be used for difficult simulation problems, include dynamic weighting (Liu et al., 2001) and sequential Monte Carlo (Del Moral et al., 2004), but the discussion of such (non-MCMC) methods is beyond the scope of this paper. One method used for difficult sampling problems in fixed dimensional spaces, that have not been used in the variable dimension case, are population-based methods (Liang & Wong, 2001; Liu, 2001).

1·1 *Population-based Markov chain Monte Carlo*

Population-based Markov chain Monte Carlo operates by embedding the target into a sequence of N independent distributions and simulating the N parallel chains, as in parallel tempering (Geyer, 1991; Hukushima & Nemoto, 1996), whilst allowing the chains to interact via various crossover moves; we give a summary of the population MCMC approach in Section 3 - see Liu (2001) for an extensive review. It is straightforward, from a theoretical standpoint, to extend this approach to the variable dimension case and we consider whether such an approach is worthwhile.

The main advantage of population-based simulation over other methods is the fact that the population *simultaneously* represent many properties of the target distribution. This is particularly useful in trans-dimensional simulation, when it can be difficult to construct efficient dimension changing proposals. Green (2003b) notes that some MCMC

methods retain information about which states have been visited (e.g. the product space approach (Carlin & Chib, 1995; Godsill, 2001)), whilst standard reversible jump 'forgets' where it once was. The first approach can provide a mode jumping property not present in standard reversible jump, namely, the ability to jump a large number of dimensions that would take a substantial time under the standard approach. Conversely, the second approach will have greater capacity to discover new states that are consistent with the target. It is clear that an algorithm which can combine *both* properties is likely to perform well; the objective of this paper is to construct such an algorithm.

1·2 *Contribution and Structure of Paper*

One aspect of population-based MCMC that is rarely considered in the literature (an exception is Madras and Zheng (2003)), are the theoretical improvements, if any, over standard MCMC methods. In this article we present a result which ensures the uniform ergodicity of a population transition kernel and allows the construction of population algorithms which are preferable, in theory, to their single chain counterparts. We demonstrate this with an example in Bayesian variable selection.

In investigations of difficult sampling problems, one is quickly drawn to the conclusion that there is not often one single method that is guaranteed to work. However, population methods naturally accommodate multiple strategies in MCMC simulation which improve the ability of the sampler to traverse the state space. In our main example we show how to combine the methods of parallel chains (Geyer, 1991), tempering (e.g. Geyer & Thompson, 1995), snooker algorithms (Gilks, et. al., 1994), partitioning of the state space (Atachade & Liu, 2004)) and delayed rejection (Green & Mira, 2001). We believe that such methods will not necessarily perform adequately individually, but together can provide a superior MCMC sampler.

This paper is organised as follows. In Section 2 we provide an illustrative example related to the clustering of gene expression data via Bayesian multivariate mixture models. In Section 3 we introduce population-based reversible jump. In Section 4 we present new theoretical results that indicate why population methods can lead to superior performance compared to single chain algorithms. In Section 5, we present population MCMC

moves for the mixture model example. In Section 6 we provide a comparison of vanilla, simulated tempering (Geyer & Thompson, 1995) and population samplers for the mixture model example. In Section 7 we conclude with a discussion, detailing extensions to our approach.

2. An Illustrative Example: Finite Mixture Modelling

Mixture models are typically used to model heterogeneous data, or as a simple means of density estimation; see McLachlan & Peel (2000) for an overview. Bayesian analysis using mixtures with an unknown number of components has only fairly recently been implemented (see Richardson & Green (1997), and in the multivariate context Stephens (2000) and Dellaportas & Papageorgiou (2004)).

2·1 *Model*

Let $\mathbf{y}_1, \ldots, \mathbf{y}_n$ denote observed data that lie on support $\mathbf{y}_i \in \mathsf{Y} \subseteq \mathbb{R}^r$. We assume that the $\mathbf{y}_i,\ i = 1, \ldots, n$, are i.i.d with density:

$$p(\mathbf{y}_i|\boldsymbol{\eta}, \mathbf{w}, k) \quad = \quad \sum_{j=1}^{k} w_j f(\mathbf{y}_i; \boldsymbol{\eta}_j)$$

where $\boldsymbol{\eta} = (\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_k)$ are component specific parameters, the weights $\mathbf{w} = (w_1, \ldots, w_k)$ are such that $\sum_{j=1}^{k} w_j = 1$, $w_j \geq 0\ \forall j$, $p(\cdot)$ denotes an arbitrary probability mass/density function and $f(\cdot)$ is the component density. For our model, we restrict ourselves to the cases where $f(\cdot)$ is either multivariate normal, $\mathcal{N}_r(\boldsymbol{\mu}, \boldsymbol{\Lambda})$ or multivariate t, $\mathcal{T}_r(\boldsymbol{\mu}, \boldsymbol{\Lambda}, s)$, where $(\boldsymbol{\mu}, \boldsymbol{\Lambda})$ are the location, scale parameters and s is the degrees of freedom for the t-density.

In specifying the priors, we follow Stephens (2000). The mean vectors for each component are taken to be independent $\mathcal{N}_r(\boldsymbol{\xi}, \boldsymbol{\kappa}^{-1})$. The $\boldsymbol{\Lambda}_j$ are independently $\mathcal{IW}_r(2\alpha', 2\boldsymbol{\beta})$, where $\mathcal{IW}_r(\cdot, \cdot)$ is the inverse Wishart distribution. We take

$$\begin{aligned}
\beta &\sim \mathcal{W}(2g, (2\mathbf{h})^{-1}) && \text{where } \mathcal{W}_r(\cdot, \cdot) \text{ is the Wishart distribution} \\
\mathbf{w} &\sim \mathcal{D}(\delta) && \text{where } \mathcal{D}(\cdot) \text{ is the symmetric Dirichlet distribution} \\
k &\sim \mathcal{U}_{\{1,\ldots,k_{\max}\}} && \text{where } \mathcal{U}_S \text{ is the discrete uniform distribution on countable set } S.
\end{aligned}$$

When using the multivariate t-distribution, the degrees of freedom are assumed known. Thus, we have a prior structure as follows

$$p(\mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\beta}, k) \;=\; \left[\prod_{j=1}^{k} p(\boldsymbol{\mu}_j)p(\boldsymbol{\Lambda}_j|\boldsymbol{\beta})\right] p(\boldsymbol{\beta})p(\mathbf{w}|k)p(k).$$

2·2 *Data Processing and Prior Distributions*

For our example we consider the problem of clustering gene expression data as in Yeung et al. (2001) and Heard et al. (2004). The data consist of $n = 4221$ genes, with the level of gene expression in the parasite *Plasmodium* measured at $r = 46$ time points. The data are discussed in Bozdech et al. (2003). Even with modern computing power applying a fully Bayesian analysis to such data is not practical, we thus propose to preprocess the data, and to reduce the (n, r)-dimensional data to (l, q) dimensions. We achieve this by adopting a K-means partitioning approach to reduce n to l, and then principal components to reduce r to q. We selected $l = 1000$ and $q = 6$.

Our priors are set in a similar way to Stephens (2000). We set $\boldsymbol{\xi}$ to be the midpoint of the observed data in its corresponding dimension. $\boldsymbol{\kappa}$ is taken to be $\mathrm{diag}(1/R_1^2, \ldots, 1/R_r^2)$ where R_q is the range of the data in dimension $q = 1, \ldots, r$. Additionally $g = r/2$, $\delta = 1$ and $\alpha' = \alpha + (r+1)/2$, where $\alpha = 3$. Finally $\mathbf{h}$ is $\mathrm{diag}(100r/(2\alpha' R_1^2), \ldots, 100r/(2\alpha' R_r^2))$.

2·3 *Performance of Vanilla Sampler*

The vanilla reversible jump sampler outlined in Appendix 1 was implemented for the data above. For illustration we use a t-distribution on four degrees of freedom as the component density in our mixture model and set $k_{\max} = 20$.

We ran the reversible jump algorithm from two different starting points, dispersed with respect to the dimensionality. The code (written in C) was run on a Pentium 4, 3 Ghz machine and took about three hours on average. The sampled values of k can be seen in Figure 1; we can observe extremely poor mixing (all variable dimension acceptance rates below 1%). We can see that there appears to be support between $k = 3 - 5$ components, but also at $k = 9 - 11$ (note for longer runs of the sampler, the same poor mixing behaviour was observed). The main problem is that the vanilla sampler cannot

jump between these two modes. (We note that any potential lack of convergence cannot be attributed to the absence of Harris recurrence: we started the algorithm with a draw from an absolutely continuous distribution).

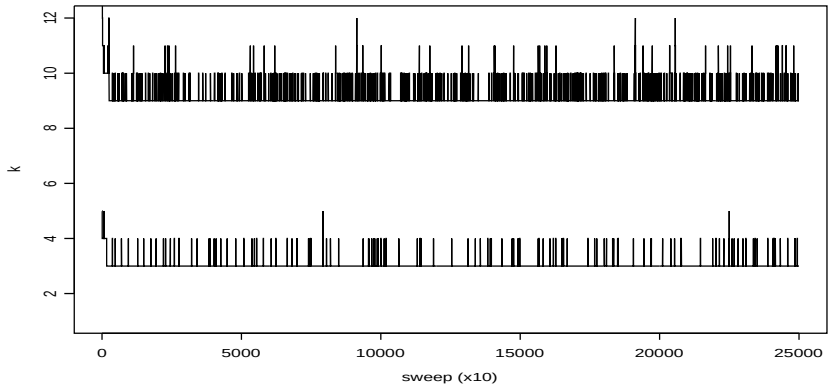


Figure 1: Sampled values of k using the vanilla reversible jump algorithm, for two chains started at different points. The algorithm was run for 250000 sweeps, every 10^{th} sample was taken.

Some of the poor performance of the vanilla algorithm can be attributed to the difference in dimensionality between different mixture models. To jump from a k to a $k+1$ component mixture model, we need to draw $1 + r + r(r+1)/2 = 28$ random variables, so we will need proposals/jump functions that are more tailored than those used in the vanilla sampler. In addition, since we can seldom jump between models, the state of the chain spends a long time adapting to the data, making it even less likely that we jump. Whilst *global* moves might be constructed (as noted by Green (2003a), they are more likely to produce better mixing than the local moves attempted here), dimension matching dictates that this will be difficult to achieve efficiently.

2·4 *Alternative Algorithms*

Possible MCMC methods that might be used to deal with the problems encountered here may be the auxiliary variable method of Brooks et al. (2003) or product space methods, but both are problematic, as they require significant user-input.

Constructing proposal distributions by creating an approximation of the target in each dimension using fixed dimensional MCMC is complicated by the label-switching problem (see Jasra et al. (2005)). Additionally, to achieve adequate mixing, the fixed dimensional simulation will need to be carried out using some non-standard MCMC approach; see Jasra et al. (2005) for example.

Delayed rejection (Green & Mira, 2001) and tempered transitions (Jennison et al., 2003) also often do not provide a solution to the problems highlighted by this example. For the former method, insufficient information is learnt at the first stage rejection to provide a significantly improved second stage proposal, whereas for the latter, it often takes a large number of intermediate simulations (e.g. 100) to provide a reasonable proposal, but even so, the performance gain is not always substantial.

3. Population-Based Reversible Jump

We now consider population-based reversible jump algorithms. First we give details of the population MCMC method, and then study the theoretical properties of the algorithm, in particular its uniform ergodicity.

3·1 *The Population MCMC Method*

Consider a sequence of densities $\{\pi_N\}$ with respect to measure λ on measure space $(\mathsf{X}^i, \mathcal{B}(\mathsf{X}^i))$, $\mathsf{X}^i = \bigcup_{k\in\mathcal{K}} \left(\{k\} \times \mathsf{X}_k^i\right)$, $i = 1, \ldots, N$. X_k^i is the support of the ith distribution in dimension k, for which in most cases will be identical to the original target and for the remainder of the paper $\pi_1 \equiv \pi$.

Now suppose we have a modified target density π^* (with respect to measure $\lambda \times \cdots \times \lambda$ (product N times) on measurable space $(\mathsf{X}^1 \times \cdots \times \mathsf{X}^N, \mathcal{B}(\mathsf{X}^1) \times \cdots \times \mathcal{B}(\mathsf{X}^1)))$, such that

$$\pi^*(\mathbf{x}, \mathbf{k}) \quad = \quad \prod_{i=1}^{N} \pi_i(x_i, k_i) \quad (x_i, k_i) \in \mathsf{X}^i$$

where $\mathbf{x} = (x_1, \ldots, x_N)$ and $\mathbf{k} = (k_1, \ldots, k_N)$.

For the auxiliary distributions (i.e. π_i, $i = 2, \ldots, N$) we take $\pi_i \propto \pi^{\zeta_i}$, $1 = \zeta_1 > \cdots > \zeta_N > 0$, where $\{\zeta_N\}$ are inverse temperature parameters. This is the standard approach,

but other settings include taking $\zeta_i = 1$ for each i; see Del Moral et al. (2004) for further discussion.

We generate N parallel (variable dimension) chains, in order that we can explore the target correctly. We seek to use the extra information of N chains at different temperatures to allow large moves in dimension of the chain of interest as well as allowing improved performance in more local moves (within and between dimensions). One of the main problems of parallel tempering is the minimal interactions between the chains. Thus we seek a more population-based approach to justify the increased cost in computation.

Since we need only make sure that all moves used satisfy detailed balance, and that any dimension changing move satisfies the dimension matching constraints of Green (1995), the population approach is valid.

We now give a theoretical justification for using the population approach. In particular, we show that population MCMC algorithms can be constructed so that the Markov chain is uniformly ergodic.

4. Some Theory for Population Samplers

We now consider some theory for population samplers.

4·1 *Uniform Ergodicity and Small Sets*

We will concentrate on the concept of uniform ergodicity (see Roberts & Rosenthal (2004) for further details).

Definition 1. *A Markov chain (X_n) with invariant measure π is uniformly ergodic if:*

$$||K^n(x,\cdot) - \pi(\cdot)||_{TV} \quad \leq \quad M\psi^n$$

where $||\cdot||_{TV}$ is the total variation distance, $M < \infty$ and $\psi \in (0,1)$.

Definition 2. *A set $A \in \mathcal{B}(\mathsf{X})$ is small if there exists $n_0 \in \mathbb{N}$, $\epsilon > 0$ and a non-trivial probability measure $\nu(dx)$ such that*

$$K^{n_0}(x,A) \quad \geq \quad \epsilon\nu(A) \qquad \forall x \in A. \tag{4.1}$$

Inequality (4.1) is known as the minorization condition. A set is termed (n_0, ϵ, ν) small if the minorization condition is satisfied. The following theorem is a well-known result for Markov chains (see Roberts & Rosenthal (2004) for example).

Theorem. *Consider a Markov transition kernel K with invariant probability measure $\pi(\mathrm{d}x)$, $x \in \mathsf{X}$. If the state space X is small then K is uniformly ergodic. Furthermore the total variation distance $||K^n(x,\cdot) - \pi(\cdot)||_{TV}$ is bounded by:*

$$R \quad = \quad \big(1-\epsilon\big)^{\left\lfloor \frac{n}{n_0} \right\rfloor}$$

with n_0 and ϵ the parameters in the minorization condition (4.1).

Thus to compare convergence speed, if we can establish that both algorithms are uniformly ergodic then we have a way to compare the algorithms by computing (n_0, ϵ).

4·2 *Main Result*

We now demonstrate that population MCMC approaches can be constructed to exhibit uniform ergodicity. For the following Theorem we denote a mutation (Markov transition) kernel as K_M and an exchange kernel K_E. A *mutation* move is a Metropolis-Hastings kernel (or cycle/mixture of such kernels) which attempts to change a single member of the population, dependent only on the current state of the chain for that member. An *exchange* move is a Metropolis-Hastings kernel which proposes to swap the current states of two different members of the population.

We set $\chi^*(\mathrm{d}\mathbf{x}) = \prod_{i=1}^{N} \chi_i(\mathrm{d}x_i)$, $x_i \in \mathsf{X}\ \forall i$, and assume that $\chi_i \ll \lambda$. We denote the density for χ_i as π_i and to avoid difficulties $\pi_i(x) > 0\ \forall x \in \mathsf{X}$, $i = 1, \ldots, N$ (i.e. that each density has the same support). Additionally, we denote a vector with its i^{th} element missing as $\mathbf{x}_{-i}$ and with only its i^{th} and j^{th} elements as $\mathbf{x}_{i,j}$. We now give our main theoretical result;

Theorem. *Consider $\widetilde{K}$ an aperiodic, χ^*-irreducible Markov transition kernel with in-*

variant measure χ^* *and*

$$\widetilde{K}(\mathbf{x}, \mathrm{d}\mathbf{x}') = \Big(\tau K_M + (1-\tau)K_E\Big)(\mathbf{x}, \mathrm{d}\mathbf{x}')$$

$$K_M(\mathbf{x}, \mathrm{d}\mathbf{x}') = \begin{cases} \prod_{i=1}^{N} K_{x_i}(x_i, \mathrm{d}x_i')\delta_{\mathbf{x}_{-i}}(\mathrm{d}\mathbf{x}_{-i}) & (i) \\ \sum_{i=1}^{N} \tau_i K_{x_i}(x_i, \mathrm{d}x_i')\delta_{\mathbf{x}_{-i}}(\mathrm{d}\mathbf{x}_{-i}) & (ii) \end{cases}$$

$$K_E(\mathbf{x}, \mathrm{d}\mathbf{x}') = \sum_{i=1}^{N-1}\sum_{l=i+1}^{N} \varepsilon_{il} K_E(\mathbf{x}_{i,l}, \mathrm{d}\mathbf{x}_{i,l}')\delta_{\mathbf{x}_{-(i,l)}}(\mathrm{d}\mathbf{x}_{-(i,l)})$$

with

$$\tau, \tau_i, \varepsilon_{il} \in (0,1), \ \sum_{i=1}^{N} \tau_i = 1, \ \sum_{i=1}^{N-1}\sum_{l=i+1}^{N} \varepsilon_{il} = 1$$

and K_{x_i} *is an aperiodic,* χ_i*-irreducible mutation (Markov) kernel with invariant measure* χ_i*,* $i = 1, \ldots, N$*. Suppose that,* $K_{x_{j^*}}$ *is uniformly ergodic* ($j^* \in \{1, \ldots, N\}$) *and for each* $i \neq j^*$ $\exists \varrho_i \in (0, \infty)$ *such that* $\pi_i(x) \leq \varrho_i \pi_{j^*}(x)$ $\forall x \in \mathsf{X}$*. Then* $\widetilde{K}$*, under either (i) or (ii), is uniformly ergodic.*

See Appendix 2 for the proof.

Remark 1. The assumption of uniform ergodicity for $K_{x_{j^*}}$ is not too restrictive. In many applications (e.g. Bayesian analyses for which the MLE exists) a proposal under the independence sampler can be found which ensures uniform ergodicity, but at the cost of being a poor proposal for π. However, if applied to a flatter (related) distribution, this proposal will perform quite well, that is, applied to a flat distribution in the population. The assumption $\pi_i(x) \leq \varrho_i \pi_{j^*}(x)$ $\forall x \in \mathsf{X}$ is quite reasonable and would apply in the framework described here, when π is bounded, for the case $j^* = N$. The idea is that the uniform rate of convergence for one of the densities in the system is propagated through it.

Remark 2. The Theorem shows in simple cases, where we can design uniformly ergodic chains for the target π (which are likely to perform well in practice), how to compare population and single chain approaches. In other words we can *construct* a population sampler which has a faster rate of convergence to π^* (and hence π) which

justifies the increased cost in computation. Additionally, we can observe the population kernels which are likely to provide good mixing for more complex examples. We demonstrate the above in the following example.

4·3 *Example: Bayesian Variable Selection*

Consider the statistical model:

$$y_i \quad = \quad \gamma_0 + \sum_{j=1}^{k_{\max}} \vartheta_j \gamma_j B(x_{ij}) + \varpi_i$$

with ϖ_i i.i.d $\mathcal{N}(0, \sigma^2)$, $\vartheta_j \in \{0, 1\}$, $\gamma_j \in \mathbb{R}$ and $B(\cdot)$ a basis function. If we consider the conjugate prior specification: $p(\boldsymbol{\gamma}|\sigma, k) = \mathcal{N}_k(m, \sigma^2 V)$, $p(\sigma^2) = \mathcal{IG}(a, b)$ (where $\mathcal{IG}(\cdot, \cdot)$ is the inverted Gamma distribution) and

$$p(\boldsymbol{\vartheta}) = \frac{1}{k_{\max} + 1} \binom{k_{\max}}{k}^{-1} \quad k = 0, \ldots, k_{\max}$$

then we can integrate out the parameters and sample from a distribution on a finite state space.

We generated 100 data points from a linear model, with $k_{\max} = 8$ (i.e. 256 states) and $z_{ij} \sim \mathcal{N}(0, 0.001^2)$, $i = 1, \ldots, 100$, $j = 1, \ldots, k_{\max} - 1$, $z_{ik_{\max}} \sim \mathcal{N}(0, 1000^2)$ and set $x_{ij} = z_{ij} + z_{ik_{\max}}$, $j = 1, \ldots, k_{\max} - 1$ and $x_{ik_{\max}} = -7 z_{ik_{\max}}$. Finally we simulated $y_i \sim \mathcal{N}(1 + \sum_{j=1}^{7} z_{ij}, 0.001^2)$. The extreme variances were chosen to ensure that there was model uncertainty on either no or all of the covariates. The posterior probability of a null model was 0.55 and 0.33 for the saturated model. This is a typical situation for which a standard MCMC sampler would fail to move around the state space easily.

To sample from the posterior distribution we use an MCMC algorithm similar to that detailed in Denison et al. (2002), that is, the reversible jump algorithm comprises of three moves: birth, death and flip.

The birth move comprises of proposing to add a covariate not currently in the model, chosen uniformly among those available. The reverse death move seeks to remove a covariate, currently in the model, uniformly at random. Finally the flip move which seeks to add a covariate not currently in the model, whilst removing an existing one.

Moves are selected, conditional on the current state of the chain. See Denison et al. (2002) for further details.

Since the state space is finite, it is clear that an ergodic Markov chain with appropriate stationary distribution is uniformly ergodic. To construct an appropriate ν (i.e. that leads to a large ϵ), we used the approach discussed in chapter 6 of Robert & Casella (2004). Let K_{ij}^n $(i, j \in \mathsf{X})$ denote the n step transition probability and suppose that $\inf_i\{K_{ij}^n\} > 0$ for some j, then $\forall j$

$$K_{ij}^n \geq \inf_l\{K_{lj}^n\} = \epsilon \upsilon_j$$

say, where

$$\upsilon_j = \frac{\inf_i\{K_{ij}^n\}}{\sum_{l\in\mathsf{X}} \inf_i\{K_{il}^n\}} \quad \text{and} \quad \epsilon = \sum_{l\in\mathsf{X}} \inf_i\{K_{il}^n\}.$$

For the algorithm discussed above, we found that the bound on the rate of convergence was reasonably similar for $n_0 = 1000$ to $n_0 = 5000$; thus we focus upon the pair $(1000, 3.63 \times 10^{-3})$.

For a population sampler, suppose we take a single auxiliary distribution:

$$\pi_2(\boldsymbol{\vartheta}_2|\mathbf{x}, \mathbf{y}) \propto L(\mathbf{x}, \mathbf{y}; \boldsymbol{\vartheta}_2)^\zeta p(\boldsymbol{\vartheta}_2)$$

with $\zeta = 0.01$. We concentrate upon a kernel which updates $\boldsymbol{\vartheta}_1$ and $\boldsymbol{\vartheta}_2$ via the RJ algorithm described above and after every 10 iterations proposes an exchange (in the spirit of a restart distribution in Tierney (1994)). (Note that the Theorem can still be applied and we will use the uniform ergodicity of the RJ kernel for the auxiliary distribution).

It can be shown that the (n_0, ϵ) pair for the population sampler (using the approach above to construct a measure in the minorization condition) is $(21, 6.01 \times 10^{-4})$. A graph comparing the bounds on total variation distance can be seen in Figure 2.

In Figure 2 we observe that the bound on the total variation distance suggests a much faster rate of convergence for the population algorithm, indeed $M_{0.01} = 5.27 \times 10^6$ (the number of iterations to achieve a bound on the total variation distance less than 0.01) for the vanilla algorithm and $M_{0.01} = 159552$ for the population algorithm, that is, it is significantly faster.

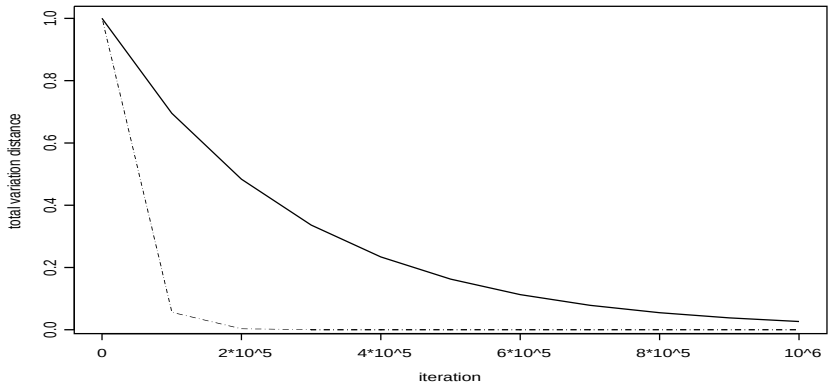


Figure 2: Plot of the Bound on the Total Variation Distance for Sampling from a Bayesian Linear Model. We used a reversible jump algorithm similar to that in Denison et al. (2002) (unbroken) and a Population algorithm (dash-dot).

We can see, for this example, a simple extension of the original algorithm to include an auxiliary distribution that provides a good proposal (for the original target) and allows efficient movement around the state space, leads to substantially improved convergence properties. For this example, there would be little extra coding effort and the CPU time would not be substantially longer than for the original RJ algorithm.

5. Population Moves for Mixture Example

Now that we have established, for difficult problems, that population methods can lead to faster convergence, we discuss how to implement population moves for our mixture example. Our notation is such that $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N)$, with $\boldsymbol{\theta}_i = (\boldsymbol{\eta}^i, \mathbf{w}^i, \boldsymbol{\beta}^i, k_i)$, $i = 1, \ldots, N$ and $(\boldsymbol{\eta}^i, \mathbf{w}^i)$ refers to all of the component specific parameters and weights for chain i. We will temper the likelihoods only, to avoid any possible integrability problems.

We now proceed to combine several MCMC methods to improve the mixing ability of the chain.

5·1 *Exchange Moves*

An exchange step is often used to swap information between parallel tempered chains. At iteration t we select two adjacent chains (in terms of the temperature parameter) uniformly at random and propose to swap their values. To merit reasonable interaction, the temperature ladder is set so that this move is accepted about half of the time (Liu, 2001). One way to improve this step is to use the delayed rejection method, as suggested by Green & Mira (2001).

At iteration t select any two chains i_1 and i_2 to swap, accepting or rejecting with the normal Hastings ratio, that is, with probability

$$\rho_1(\boldsymbol{\theta}, \boldsymbol{\theta}') = \min\left\{1, \frac{\pi_{i_1}(\boldsymbol{\theta}_{i_2})\pi_{i_2}(\boldsymbol{\theta}_{i_1})}{\pi_{i_1}(\boldsymbol{\theta}_{i_1})\pi_{i_2}(\boldsymbol{\theta}_{i_2})}\right\}$$

where the labelling of the chains is with respect to the current state of the chain and $\boldsymbol{\theta}'$ denotes the new configuration of chains. If rejected, select two adjacent chains i_3 and i_4 to swap, denoting this configuration $\boldsymbol{\theta}''$. To ensure reversibility with respect to the target (i.e. as part of the delayed rejection method) we perform a pseudo move that means that at the proposed state of the chain we rejected a move which tried to swap chains labelled as i_1 and i_2. The second stage move is accepted with probability

$$\rho_2(\boldsymbol{\theta}, \boldsymbol{\theta}'') = \min\left\{1, \frac{\pi_{i_3}(\boldsymbol{\theta}_{i_4})\pi_{i_4}(\boldsymbol{\theta}_{i_3})\{1-\rho_1(\boldsymbol{\theta}'', \boldsymbol{\theta}^*)\}}{\pi_{i_3}(\boldsymbol{\theta}_{i_3})\pi_{i_4}(\boldsymbol{\theta}_{i_4})\{1-\rho_1(\boldsymbol{\theta}, \boldsymbol{\theta}')\}}\right\}$$

where $\boldsymbol{\theta}^*$ denotes the configuration under the pseudo move.

This move allows for increased interaction between the population. At the first stage, we allow any pair of chains to be swapped, thus if a chain at a high temperature is consistent with one of the distributions at a lower temperature, it is allowed to quickly jump down the ladder. It is likely that this move will only rarely be accepted (in fact not too often), thus our second stage move attempts a proposal that is far more likely to be accepted - thus still allowing reasonable rate of exchanging information.

5·2 *Crossover Moves*

Liang & Wong (2001) employ various crossover moves to increase the interaction of the population. We use two move types:

Variable dimension crossover

When in state $\boldsymbol{\theta}$, we select a variable dimension crossover with probability $v(\boldsymbol{\theta})$;

$$v(\boldsymbol{\theta}) = \begin{cases} 1 & \text{if } k_i \neq k_j \text{ for some } i \neq j \\ 0 & \text{otherwise.} \end{cases}$$

Note the case $v(\boldsymbol{\theta}) = 0$ corresponds to a 'do nothing' move. We select a pair of chains with differing dimension with probability inversely proportional to the squared absolute value of the difference between the dimensions. We then permute the chains to obey the identifiability constraint which orders on the weights, then propose the new state of the chains, by swapping k's and the weights. We take the lowest weighted component specific parameters of the higher dimensional chain to the lower dimensional chain, i.e. if $k_{i_1} > k_{i_2}$ for the selected chains i_1, i_2 we propose:

$$\begin{aligned} \boldsymbol{\eta}'_{i_1} &= (\boldsymbol{\eta}^{i_1}_{k_{i_1}-k_{i_2}}, \ldots, \boldsymbol{\eta}^{i_1}_{k_{i_1}}) \\ \boldsymbol{\eta}'_{i_2} &= (\boldsymbol{\eta}^{i_1}_{1}, \ldots, \boldsymbol{\eta}^{i_1}_{k_{i_1}-k_{i_2-1}}, \boldsymbol{\eta}^{i_2}_{1}, \ldots, \boldsymbol{\eta}^{i_2}_{k_{i_2}}) \end{aligned}$$

where $\boldsymbol{\eta}^{i_1}_j$ denotes an element of $\boldsymbol{\eta}^{i_1}$. The acceptance probability is easily calculated and thus omitted. After the move has been accepted or rejected we propose a random permutation (all permutations have uniform probability of being proposed) of the labels of the parameters for all chains, thus ensuring invariance with respect to the target.

Fixed Dimension Crossover

When in state $\boldsymbol{\theta}$, we select a fixed dimension crossover with probability 1, if it can be selected (i.e. there are at least two chains with the same dimensionality) otherwise we select a 'do nothing' move. Select a pair of chains (i_1, i_2) with the same dimensionality, with probability

$$p(i_1, i_2|\boldsymbol{\theta}) \propto |\zeta_{i_1} - \zeta_{i_2}|^{-1} \mathbb{I}_{k_{i_1}=k_{i_2}}. \tag{5.1}$$

We then permute the chains by ordering on the first dimension of the means. We select a position $j = 1, \ldots, k_{l_1-1}$ to crossover, this selection made with probability proportional to $1/j$ and switch all component specific parameters to the left of j inclusive (note that

if the identifiability constraint is not satisfied in the proposed state of the chain we immediately reject) and accept or reject on the basis of the Hastings ratio. After the accept reject decision has been made, we again propose a random permutation of the labels of the parameters.

5·3 *Snooker Jumps*

One of the most important ways we can use the information in the population is by targeting variable dimension jumps by using another chain. This idea is linked to the snooker algorithm of Gilks et al. (1994) and is performed in the following way.

When in state $\boldsymbol{\theta}$ select a birth with probability $b(\boldsymbol{\theta})$, where

$$b(\boldsymbol{\theta}) = \begin{cases} 1 & \text{if } k_i = 1 \ \forall i \\ 0 & \text{if } k_i = k_{\max} \ \forall i \\ 1/2 & \text{otherwise} \end{cases}$$

then select a chain (the current point $\boldsymbol{\theta}_c$) for which a birth is possible (let $m_b(\boldsymbol{\theta})$ be the number of chains such that a birth can occur when in state $\boldsymbol{\theta}$) with uniform probability and select an anchor point ($\boldsymbol{\theta}_a$) with probability inversely proportional to the absolute value of the difference between the inverse temperatures. Generate $w \sim \mathcal{B}e(1, k_c)$, with $\mathcal{B}e(\cdot,\cdot)$ the beta density, and draw a new $\boldsymbol{\mu}, \boldsymbol{\Lambda}$ from:

$$q(\boldsymbol{\mu}, \boldsymbol{\Lambda}) \quad = \quad \sum_{j=1}^{k_a} \bar{h}(\boldsymbol{\eta}_j) \mathcal{N}_r(\boldsymbol{\mu}_j^a, \sigma) \mathcal{IW}(2r+3, \boldsymbol{\Lambda}_j^a)$$

where

$$\bar{h}(\boldsymbol{\eta}_j) \quad \propto \quad \frac{1}{k_c} \sum_{l=1}^{k_c} h((\boldsymbol{\mu}_j^a, \boldsymbol{\Lambda}_j^a), (\boldsymbol{\mu}_l^c, \boldsymbol{\Lambda}_l^c))$$

and $h(\cdot,\cdot)$ is the Mahalanobis distance. We then perform the rest of the move as for the birth in Appendix 1. In the death we perform much the same as for Appendix 1, except we select a current point with probability $1/m_d(\boldsymbol{\theta})$ (where $m_d(\boldsymbol{\theta})$ is the number of chains for which a death can occur when in state $\boldsymbol{\theta}$) and (redundantly) select an anchor point (which is used in the reverse birth). The birth move is accepted with probability

$\min\{1, A\}$ with:

$$A = \frac{p(\mathbf{y}|\boldsymbol{\eta}'^{c}, \mathbf{w}'^{c}, k_c+1)^{\zeta_c} p(\boldsymbol{\mu}) p(\boldsymbol{\Phi}) p(k_c+1)}{p(\mathbf{y}|\boldsymbol{\eta}^{c}, \mathbf{w}^{c}, k_c)^{\zeta_c} p(k_c)} B(k_c\delta, \delta)^{-1} w^{\delta-1} (1-w)^{k_c(\delta-1)} \frac{(k_c+1)!}{k_c!} \times \frac{d(\boldsymbol{\theta}') m_b(\boldsymbol{\theta})}{(k_c+1) b(\boldsymbol{\theta}) m_d(\boldsymbol{\theta}')} \frac{(1-w)^{k_c-1}}{\mathcal{B}e(w; 1, k_c) q(\boldsymbol{\mu}, \boldsymbol{\Phi})}$$

where $\boldsymbol{\Phi}$ is the Cholesky decomposition of $\boldsymbol{\Lambda}$ (see Appendix 1 for details), $p(\mathbf{y}|\boldsymbol{\eta}^{c}, \mathbf{w}^{c}, k_c)^{\zeta_c}$ is the tempered likelihood (for the current point), $B(\cdot,\cdot)$ is the beta function and $\mathcal{B}e(x;\cdot,\cdot)$ is the beta density evaluated at x. The objective of this move is to propose new component specific parameters which are likely to be consistent with the data, but are markedly different from the current components. It also provides an adaptive element to the birth proposal, as it relies on current information that is being continuously updated.

5·4 *Partitioning*

One aspect of population-based simulation that is apparent, is the need for diversity of the population (also the case in sequential Monte Carlo - see Del Moral et. al. (2004)). In many cases for which it is difficult to traverse the state space, it is often the case that the members at higher inverse temperatures can become trapped as in single chain MCMC methods (i.e. as in Section 2·3). As a result, we will tend to derive incorrect quantities of interest. To avoid this problem, we propose to partition some of the members of the population. That is, for some subset $\mathsf{I} = \{l, \ldots, N\}$ $(l \geq 2)$, π_i with $i \in \mathsf{I}$ is a density constrained to $\mathsf{X}_i \subset \mathsf{X}$.

Choosing the partition is a difficult problem which needs to be determined in a problem specific manner. See Atachade & Liu (2004) for some discussion.

5·5 *The Algorithm*

To sample from the augmented distribution we use the algorithm below; we use the genetic algorithm terminology of Liang & Wong (2001).

0. *Initialise the chain* $\boldsymbol{\theta}$.

For $t = 1, \ldots, M$ *sweep over the following:*

1. *Mutation. Select a chain i with probability τ_i and then perform one sweep of the reversible jump algorithm in Appendix A for this chain.*
2. *Make a random choice between performing steps 3 or 4.*
3. *Crossover. Propose a variable dimension crossover move with probability* $1/2$*, else propose a fixed dimension crossover.*
4. *Snooker Jump. Propose a birth with probability $b(\boldsymbol{\theta})$, else propose a death.*
5. *Exchange. Perform the delayed rejection exchange move.*

Note that, for partitioned chains, we only allow them to be involved in fixed dimensional crossovers and a special exchange move that we now describe (which is added to 5).

Propose to exchange a partitioned and non-partitioned chain, selecting the move only if such a move may be performed. All selections are made with uniform probability and no delayed rejection is used.

6. Gene expression example revisited

6·1 *Specification of Simulation Parameters*

Population Size

To run the population algorithm in Section 5·5 we used $N = 25$ with 5 chains partitioned. We recommend a large population size in general, so that results are reasonably similar for separate runs of the algorithm. We note that this may lead to slower convergence to the target density π^*, in terms of CPU time, but that there is more information to improve exploration of the original target density π.

A point of interest, is that with a large population, significant CPU time may be spent on updating variables that may never add any information for the original target density; thus the method of target orientated evolutionary Monte Carlo has been developed: see Goswami & Liu (2005).

Temperature Parameters

For the main population (i.e. non-partitioned chains) the following inverse temperatures were selected:

$$\begin{aligned} \zeta_1 &= 1 \\ \zeta_i &= \zeta_{i-1} - \varsigma\varphi^{i-1} \; i = 2, \ldots, 20 \end{aligned}$$

for constants $\varsigma > 0$, $\varphi > 1$. We selected $\varsigma = 10^{-6}$, $\varphi = 1.85$.

Our choice of heating schedule, and population size was based upon pilot tuning. We selected a slowly decreasing sequence of ζ's since we observed a poor acceptance rate for the exchange move for distributions that were further away. We found that the inverse temperature at which the reversible jump algorithm performed best (that is reasonable acceptance rates along with regions of high support that were similar to the target) was $\zeta = 0.75$. Thus we attempt to include a distribution with this temperature. We note that we need to be careful when specifying temperatures, since for low temperatures (low depending on the problem at hand) the distribution starts to favour dimensionalities that are small, although this may be alleviated by specifying priors for k which penalise small values. For more discussion on temperature selection see Goswami & Liu (2005).

Partitioned Chains

To select the partitions we used a prior simulation. We ran the algorithm with the $N = 25$ and the inverse temperature parameters discussed above, only 9 chains in the main population and 16 partitioned chains (given inverse temperature parameter 0.999). We selected partitions with respect to the dimensionality, that is we had 10 chains constrained to lie $k \in \{1, 2\}, \ldots, \{19, 20\}$ then six other chains constrained to lie in $k \in \{3, \ldots, 6\}$, $\{6, \ldots, 9\}$, $\{9, \ldots, 12\}$, $\{12, \ldots, 15\}$ and $\{15, \ldots, 18\}$. This was adopted in order to determine whether there was any support outside $\{3, \ldots, 11\}$ found in Section 2·3. Based upon a short run we took the five partitioned chains to lie in (subsets of $\{1, \ldots, k_{\max}\}$) $\{2, 3, 4\}$, $\{4, 5, 6\}$, $\{5, 6, 7\}$ and $\{7, 8, 9\}$, $\{9, 10, 11\}$. The idea of the prior tuning is to avoid wasting CPU time on population members constrained to lie

in areas of support that have low density with respect to the original target density. Additionally, the partitioned chains need to be able interact with the main population and the prior tuning allows us to make this choice. The inverse temperature parameters for the partitioned chains were 0.999, since we seek to maintain diversity with respect to the population at colder temperatures.

For further discussion in the setting of partitions, especially in the context of reversible jump, see Atchade & Liu (2004).

We ran the algorithm in Section 5·5 (with the addition that if a crossover move was selected, we also propose an exchange for the chain of interest) for 1 million sweeps which took approximately $9\frac{1}{2}$ hours.

6·2 *Comparison with Vanilla Sampler*

In Figure 3 (a) we can see the sampled values of k for the original target density. The improvement over the vanilla sampler is substantial, on average, the chain of interest took 6.75 sweeps to jump between a mixture model with less than 4 components to a mixture with more than 6.

The inability of the original reversible jump algorithm to move around the state space (i.e. from $k = 3 - 5$ to $k = 9 - 11$) is not the case for the population sampler, since it may represent both states. We note that a move which proposes to add and remove four mixture components (e.g. from 3 to 7 components) may be constructed, but the difference in dimensionalities is 112, which will mean that it will be very difficult to design an efficient move. Note that due to the large state space that we are sampling from, we do not claim that the sampler has converged and visited all possible regions of interest in the state space.

In Figure 3 (b) we can observe the sample path for a population-based algorithm that did not use partitioning. We can observe that, despite the substantially improved performance when compared to the vanilla sampler, we have missed the region $k \in \{8, 9, 10\}$. This is due to the reasons discussed earlier, and demonstrates that partitioning can help guard against such problems.

The effective sample size (ESS) (see Liu (2001) for example) for k was 59578 (60000

samples, using a lag of 10 in the autocorrelation calculation, which more than enough based upon inspection of the plots), compared with 176 for the vanilla algorithm (averaged over two chains for a 50000 sample size). Taking

$$E = \frac{\mathrm{ESS}_{\mathrm{pop}}}{M_{\mathrm{pop}}T_{\mathrm{pop}}} / \frac{\mathrm{ESS}_{\mathrm{van}}}{M_{\mathrm{van}}T_{\mathrm{van}}}$$

where the subscripts refer to the population and vanilla algorithms respectively, M is the sample size and T is the CPU time, we obtain $E = 70.21$. Therefore there is little contest between using population-based reversible jump and the vanilla counterpart, for this example (note all coincidental simulation parameters are the same between algorithms).

6·3 *The Efficiency of Sampler*

The exchange move was accepted 44% of the time at the first stage and 75% at the second. This indicates that delayed rejection helps to ensure that we are constantly swapping information between the chains (i.e. for 86% of the sweeps there is at least one exchange).

The snooker and variable dimension crossover moves have acceptance rate less than 1%. That this occurs is to be expected. Liang & Wong (2001) report fairly small acceptance rates for their crossover moves and that our rates are smaller is because we are working in a more complex space than for their examples. Our experience with the snooker and variable dimension crossover (as we have implemented them) in more simple examples, is that they are generally not worth the extra coding effort given their performance. However, we were satisfied with the fixed dimensional crossover which was accepted 2.9% of the time. The snooker birth is accepted more often than the standard birth, but the reverse snooker death move is rarely accepted (c.f. a birth move with a proposal that has low variance). Hence the move is less successful overall than the standard birth.

The variable dimension move acceptance rates (averaged over all chains) were still below 1% with the split/combine move being less effective.

6·4 *Comparison with Simulated Tempering*

A more appropriate sampler to compare with the population method is a simulated tempering algorithm (see Hodgson (1999) for an example of another variable dimension simulated tempering algorithm). Here the target distribution is:

$$\pi(\boldsymbol{\theta}, \zeta, k|\mathbf{y}) \quad \propto \quad p(\mathbf{y}|\boldsymbol{\theta}, k)^{\zeta_i} p(\boldsymbol{\theta}, k) p(\zeta_i)$$

where $p(\zeta_i)$ is known as the *pseudo prior* and the temperatures lie in some finite set. To sample from this distribution we use the reversible jump algorithm in Appendix 1, and to update the temperature parameter we used a delayed rejection move as follows.

- Propose a temperature uniformly at random and accept or reject with the Hastings ratio. If rejected, select an adjacent temperature, performing a pseudo move that selects to go from the proposed temperature at the second stage, to the proposed temperature at the first stage.

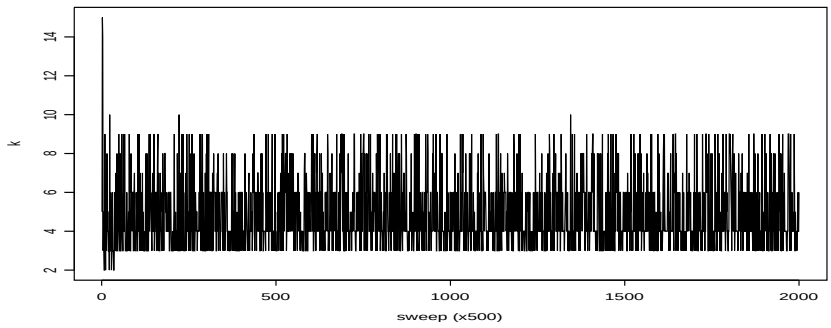
We could not find a pseudo prior so that for a reasonable number of temperatures (e.g. 25), we could jump between the target and the inverse temperature 0.75 (recall the reversible jump algorithm performs well at this temperature), thus for any sensible number of distributions the performance of this approach is as the vanilla algorithm (slightly improved). An example of a run of the simulated tempering algorithm was setting $p(\zeta_i) \propto 1/i$, with $\zeta_1 = 1$ and having a difference of 1×10^{-4} between each temperature. We found, with 25 distributions, the algorithm only visited the distribution of interest 10% of the time in a run of 250000 sweeps.

This justifies why we used many chains that are similar to the target for the population approach; they contain information relevant to the target, with slightly improved mixing. That the distribution with inverse temperature 0.75 is difficult to reach under the simulated tempering approach does not matter in the population algorithm. This is because we use this distribution to (once in a while) discover states with high posterior support, whilst the distributions close to the target retain the places we have been.

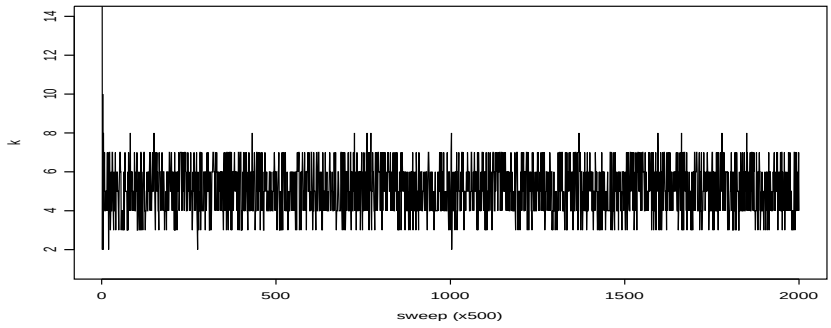
6·5 *Comments*

In early simulations of the algorithm, states with low probability under target π may be accepted into the chain of interest, especially for a sequence of distributions with significant overlap.

Recall that, since we may not update any samples directly related to the distribution of interest (on a particular sweep of the algorithm), it is not advisable to record samples for every sweep.



(a) Sampled k.



(b) Sampled k without Partitioining.

Figure 3: Sampled k (a) and Sampled k without partitioning (b) from the population-based reversible jump algorithm in Section 5·5 We ran the algorithm for 1 million sweeps and every 500^{th} sample was taken.

7. Discussion

To summarize, we have demonstrated the following. We found that a vanilla reversible jump algorithm failed completely to jump around a multimodal model space, meaning we were unsure even of the support of the target. We introduced population-based reversible jump MCMC and gave some theoretical justification for why these methods can be preferable to standard MCMC methods. We then saw that population reversible jump is a means to improving variable dimension simulation.

An aspect of interest is the fact that the basic algorithm (without population moves) can easily be coded given a vanilla sampler. Therefore our method provides a simple way to check the performance of MCMC algorithms.

One of the problems of our approach is the limited amount of success we had with our crossover moves. Whilst this was observed for simpler problems in Liang & Wong (2001), we would still hope that the population can still provide more information when proposing moves. We note, however, that we do not want too many snooker type moves (see Gilks et al. (1994)) to be accepted. This is because it will reduce the diversity of the population. Another area that needs to be investigated, is developing general guidelines for constructing partitions and finding efficient ways to make them interact with the population.

Theoretical extensions that may be considered in the future are as follows. First, to develop coupling and perfect simulation methods in the spirit of Brooks et al. (2002). Secondly, to use adaptive MCMC methods (see Andrieu & Robert (2001) for example and Chauveau & Vandekerkhove (2002) in the population context). This is likely to be superior to standard adaptive algorithms, since we have more information to update proposals. Furthermore, we have more information in terms of where the chain *has not* been, i.e. we may search (fewer) regions of the support of π for states with high density. One area that we did investigate was Peskun ordering (Peskun, 1973; Tierney, 1998), but found that results we derived were difficult to apply in practice.

Overall, the success of our method helps open up the possibility of fully Bayesian analyses in other problems for which simulation is prohibitively slow.

Acknowledgement

The first author was supported by an Engineering and Physical Sciences Research Council Studentship. The first author acknowledges many useful conversations with Nick Heard about reversible jump and also on the Bayesian variable selection and gene expression examples. Additionally Matthew Gander and Zhicheng Zhang for valuable coding advice.

Appendix 1: *Reversible Jump Sampler*

One of the drawbacks of the model we have selected is the need for $\mathbf{\Lambda}$ to be positive definite. As a result, moves in MCMC simulation will be difficult to construct such that this constraint is satisfied. To deal with this problem we consider the Cholesky decomposition $\mathbf{\Phi}$ (see Dellaportas & Papageorgiou (2004) for an analysis using the spectral decomposition). That is, $\mathbf{\Lambda} = \mathbf{\Phi}\mathbf{\Phi}'$ where $\mathbf{\Phi}$ is lower triangular with positive diagonal elements (recall the Jacobian is $2^r \prod_{l=1}^{r} \phi_{ll}^{r-l+1}$). Thus in our MCMC moves we need only ensure that the diagonal elements of $\mathbf{\Phi}$ are positive to guarantee positive definiteness of $\mathbf{\Lambda}$.

To draw approximate samples from our posterior distribution we use RJMCMC. Following Cappé et. al. (2003) (among others) we do not complete the missing data (i.e. the class labels for each data point) and rely upon Metropolis-Hastings updates (unless otherwise stated), as follows.

Firstly the fixed dimensional moves. The component specific means ($\boldsymbol{\mu}_j$) and component specific lower triangular part of $\mathbf{\Phi}_j$ are both updated via an additive cauchy random walk, independent in each dimension. The component specific diagonals of $\mathbf{\Phi}_j$ are updated via a multiplicative log-normal random walk, independent in each dimension. The weights are proposed using an additive normal random walk on the logit scale. Finally, $\boldsymbol{\beta}$, is generated using a Gibbs kernel; the full conditional is $\mathcal{W}(2(g+k\alpha), (2\mathbf{h} + 2\sum_{j=1}^{k} \mathbf{\Lambda}_j^{-1})^{-1})$.

Secondly a birth/death of a component, largely following Richardson & Green (1997). Briefly, we draw a new $\boldsymbol{\mu}$ and $\mathbf{\Phi}$ from the prior and $w \sim \mathcal{B}e(1,k)$, setting the new weights as $(w_1(1-w), \ldots, w_k(1-w), w)$, selecting the move with probability b_k (when in state k). The death, selected with probability d_k, is performed by selecting a component to die with uniform probability and inverting the jump function.

Finally, a split/combine of a component. We select a split with probability s_k and choose a component j^* uniformly at random to split into components labelled as (j_1, j_2). The split requires the following actions:

(i) Split the weight by drawing $u_1 \sim \mathcal{B}e(\gamma, \gamma)$ and set

$$\begin{aligned} w_{j_1} &= u_1 w_{j^*} \\ w_{j_2} &= (1-u_1) w_{j^*}. \end{aligned}$$

(ii) Split the mean vector by drawing $u_{1(2)}, \ldots, u_{r(2)} \sim \mathcal{N}(0, \sigma_\mu)$ and take

$$\begin{aligned} \mu_{l(j_1)} &= \mu_{l(j^*)} + u_{l(2)} \\ \mu_{l(j_2)} &= \mu_{l(j^*)} - u_{l(2)}. \end{aligned}$$

(iii) Split the off diagonals of $\mathbf{\Phi}$ by drawing $u_{21(3)}, \ldots, u_{r(r-1)(3)} \sim \mathcal{N}(0, \sigma_\phi)$ and take

$$\begin{aligned} \phi_{lm(j_1)} &= \phi_{lm(j^*)} + u_{lm(3)} \\ \phi_{lm(j_2)} &= \phi_{lm(j^*)} - u_{lm(3)} \end{aligned}$$

where $l = 2, \ldots, r$, $m = 1, \ldots, l-1$.

(iv) Split the diagonals of $\mathbf{\Phi}$ by drawing $u_{11(3)}, \ldots, u_{rr(3)} \sim \mathcal{LN}(0, \sigma)$ and take

$$\phi_{ll(j_1)} = \frac{\phi_{ll(j^*)}}{u_{ll(3)}} \qquad \phi_{ll(j_2)} = \phi_{ll(j^*)} u_{ll(3)}.$$

In order to combine we select the move with probability c_k and invert the jump function above. We note that due to the symmetry constraint imposed on the jump function it does not matter which way we combine the components. We choose two components to combine, when in state k, with probability inversely proportional to the Mahalanobis distance between them, that is:

$$p_k(j_1, j_2) \propto \left[(\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})' \mathbf{\Lambda}_{j_1}^{-1} (\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2}) + (\boldsymbol{\mu}_{j_2} - \boldsymbol{\mu}_{j_1})' \mathbf{\Lambda}_{j_2}^{-1} (\boldsymbol{\mu}_{j_2} - \boldsymbol{\mu}_{j_1}) \right]^{-1}.$$

The split in state k is accepted with probability $\min\{1, A\}$ where

$$\begin{aligned} A = {} & (\text{likelihood ratio}) \frac{p(\mathbf{\Phi}_{j_1}) p(\mathbf{\Phi}_{j_2})}{p(\mathbf{\Phi}_{j^*})} \frac{p(\boldsymbol{\mu}_{j_1}) p(\boldsymbol{\mu}_{j_2})}{p(\boldsymbol{\mu}_{j^*})} B(k\delta, \delta)^{-1} \left(w_{j^*} u_1 (1 - u_1) \right)^{\delta - 1} \times \\ & \frac{p(k+1)}{p(k)} \frac{(k+1)!}{k!} \frac{k c_{k+1} p_{k+1}(j_1, j_2)}{s_k} \frac{|J|}{2 q_1(u_1) q_2(\boldsymbol{u}_2) q_3(\boldsymbol{u}_3)} \end{aligned}$$

where $|J|$ is the Jacobian:

$$|J| = 2^{\frac{r(r+3)}{2}} w_{j^*} \prod_{l=1}^{r} \frac{\phi_{ll(j^*)}}{u_{ll(3)}}$$

and obvious notation for the prior and proposal densities.

More complex split/combine moves may be employed, (see for example Dellaportas & Papageorgiou (2004)) however, it is unlikely for problems such as those considered in this paper, that such jump functions will lead to substantially faster mixing. We thus prefer the simple move described above.

The algorithm is performed in a deterministic sweep over all fixed dimension moves followed by a random choice of birth, death, split or merge selected with uniform probability (assuming we allow a move, i.e. no birth or split when $k = k_{\max}$ or death or combine when $k = 1$).

APPENDIX 2: *Proof of Theorem*

We prove the Theorem for $N = 2$, both for clarity and that its proof will be the basis for general N. We concentrate on the case (i) since (ii) follows in a similar manner. The strategy of the proof is to use

the uniform ergodicity of $K_{x_{j^*}}$ (which we take to be K_{x_2}) and the acceptance of an exchange move. We denote $x_i^{(l)}$ as the value of x_i after l steps.

Let $A_1, A_2 \in \mathcal{B}(\mathsf{X})$, $A = A_1 \times A_2 \in \mathcal{B}(\mathsf{X}) \times \mathcal{B}(\mathsf{X})$. Since we have that X is (n_0, ϵ, ν) small under K_{x_2}, consider:

$$\begin{aligned}
\widetilde{K}^{2n_0+1}(\mathbf{x}, A) &= \int_{\mathbf{x}^{(n_0)} \in \mathsf{X}^2} \widetilde{K}^{n_0}(\mathbf{x}, \mathrm{d}\mathbf{x}^{(n_0+1)}) \widetilde{K}^{n_0+1}(\mathbf{x}^{(n_0+1)}, A) \\
&= \int_{\mathbf{x}^{(n_0)} \in \mathsf{X}^2} \int_{\mathbf{x}^{(n_0+1)} \in \mathsf{X}^2} \widetilde{K}^{n_0}(\mathbf{x}, \mathrm{d}\mathbf{x}^{(n_0)}) \widetilde{K}(\mathbf{x}^{(n_0)}, \mathrm{d}\mathbf{x}^{(n_0+1)}) \widetilde{K}^{n_0}(\mathbf{x}^{(n_0+1)}, A) \\
&\geq \int_{\mathbf{x}^{(n_0)} \in \mathsf{X}^2} \int_{\mathbf{x}^{(n_0+1)} \in \mathsf{X}^2} \widetilde{K}^{n_0}(\mathbf{x}, \mathrm{d}\mathbf{x}^{(n_0)}) \widetilde{K}(\mathbf{x}^{(n_0)}, \mathrm{d}\mathbf{x}^{(n_0+1)}) \tau^{n_0} K_M^{n_0}(\mathbf{x}^{(n_0+1)}, A)
\end{aligned}$$

where we have applied Chapman-Kolmogorov twice and used $\widetilde{K}^n(\mathbf{x}, \cdot) \geq \tau^n K_M^n(\mathbf{x}, \cdot)$ $\forall \mathbf{x} \in \mathsf{X}^2$ and all $n \in \mathbb{N}$ (note that the integral (resp. $\mathbf{x}^{(n_0+1)}$) is a measurable function of $\mathbf{x}^{(n_0)}$).

Note we have that $K_M^n(\mathbf{x}, \cdot)$ is equal to the product measure $K_{x_1}^n(x_1, \cdot) K_{x_2}^n(x_2, \cdot)$, using this and the above we can obtain:

$$\begin{aligned}
\widetilde{K}^{2n_0+1}(\mathbf{x}, A) \geq\ & \tau^{2n_0}(1-\tau) \int_{\mathsf{X}^2} K_{x_1}^{n_0}(x_1, \mathrm{d}x_1^{(n_0)}) K_{x_2}^{n_0}(x_2, \mathrm{d}x_2^{(n_0)}) \times \\
& \int_{\mathsf{X}^2} K_E(\mathbf{x}^{(n_0)}, \mathrm{d}\mathbf{x}^{(n_0+1)}) \times \\
& K_{x_1}^{n_0}(x_1^{(n_0+1)}, A_1) K_{x_2}^{n_0}(x_2^{(n_0+1)}, A_2)
\end{aligned} \tag{7.1}$$

which corresponds to selecting n_0 consecutive mutations followed by an exchange and then followed again by another n_0 consecutive mutations.

Applying the minorization condition for $K_{x_2}^{n_0}(x_2^{(n_0+1)}, A_2)$, equation (7.1) becomes

$$\begin{aligned}
\widetilde{K}^{2n_0+1}(\mathbf{x}, A) \geq\ & \tau^{2n_0}(1-\tau) \int_{\mathsf{X}^2} K_{x_1}^{n_0}(x_1, \mathrm{d}x_1^{(n_0)}) K_{x_2}^{n_0}(x_2, \mathrm{d}x_2^{(n_0)}) \times \\
& \int_{\mathsf{X}^2} \delta_{x_1^{(n_0)}}(\mathrm{d}x_2^{(n_0+1)}) \delta_{x_2^{(n_0)}}(\mathrm{d}x_1^{(n_0+1)}) \min\left\{1, \frac{\pi_1(x_2^{(n_0)}) \pi_2(x_1^{(n_0)})}{\pi_1(x_1^{(n_0)}) \pi_2(x_2^{(n_0)})}\right\} \times \\
& K_{x_1}^{n_0}(x_1^{(n_0+1)}, A_1) \epsilon \nu(A_2)
\end{aligned} \tag{7.2}$$

where we have ignored the rejection of an exchange move.

Since

$$\min\left\{1, \frac{\pi_1(x_2)\pi_2(x_1)}{\pi_1(x_1)\pi_2(x_2)}\right\} \geq \min\left\{1, \frac{\pi_1(x_2)}{\pi_2(x_2)\varrho_1}\right\} \ \forall (x_1, x_2) \in \mathsf{X}^2 \tag{7.3}$$

and using the measurability of the function (note the inequality may hold almost everywhere wrt some dominating measure), we can split the integrals in equation (7.2) into $I_1 \times I_2$, where:

$$\begin{aligned}
I_1 &= \int_{\mathsf{X}} K_{x_1}^{n_0}(x_1, \mathrm{d}x_1^{(n_0)}) \int_{\mathsf{X}} \delta_{x_1^{(n_0)}}(\mathrm{d}x_2^{(n_0+1)}) \\
I_2 &= \int_{\mathsf{X}^2} K_{x_2}^{n_0}(x_2, \mathrm{d}x_2^{(n_0)}) \delta_{x_2^{(n_0)}}(\mathrm{d}x_1^{(n_0+1)}) \min\left\{1, \frac{\pi_1(x_2^{(n_0)})}{\pi_2(x_2^{(n_0)})\varrho_1}\right\} K_{x_1}^{n_0}(x_1^{(n_0+1)}, A_1).
\end{aligned}$$

Clearly $I_1 = 1$. For I_2, integrating with respect to Dirac measure and then applying the minorization condition we obtain:

$$I_2 \geq \epsilon \int_{\mathsf{X}} \nu(\mathrm{d}x_2^{(n_0)}) \min\left\{1, \frac{\pi_1(x_2^{(n_0)})}{\pi_2(x_2^{(n_0)})\varrho_1}\right\} K_{x_1}^{n_0}(x_2^{(n_0)}, A_1).$$

Therefore equation (7.2) becomes:

$$\widetilde{K}^{2n_0+1}(\mathbf{x}, A) \geq \theta\nu^*(A) \tag{7.4}$$

where

$$\begin{aligned}
\theta &= \epsilon^2\tau^{2n_0}(1-\tau)\phi \\
\nu^*(A) &= K^*(A_1)\nu(A_2) \\
K^*(A_1) &= \frac{1}{\phi}\int_{\mathsf{X}} \nu(\mathrm{d}x_2^{(n_0)}) \min\left\{1, \frac{\pi_1(x_2^{(n_0)})}{\pi_2(x_2^{(n_0)})\varrho_1}\right\} K_{x_1}^{n_0}(x_2^{(n_0)}, A_1) \\
\phi &= \int_{\mathsf{X}} \nu(\mathrm{d}x_2) \min\left\{1, \frac{\pi_1(x_2)}{\pi_2(x_2)\varrho_1}\right\}
\end{aligned}$$

Since (7.4) holds $\forall \mathbf{x} \in \mathsf{X}^2$, $\forall A \in \mathcal{B}(\mathsf{X}) \times \mathcal{B}(\mathsf{X})$ and the product (probability) measure is non-trivial, we have that X^2 is $(2n_0+1, \theta, K^* \times \nu)$ small.

Now consider the case of general $N > 2$, we will use the $Nn_0 + N - 1$ step transition kernel and the set $A = A_1 \times \cdots \times A_N \in \mathcal{B}(\mathsf{X}) \times \cdots \times \mathcal{B}(\mathsf{X})$ (product N times). We will assume that $j^* = N$ and for clarity in the exchange steps we will swap in decreasing order of the indices. However, this need not be the case and thus the minorization condition has $(N-1)!$ representations of the below; we consider this at the end of the proof.

Following a similar argument for $N = 2$ we obtain (i.e. the equivalent of equation (7.2)):

$$\begin{aligned}
\widetilde{K}^{n_N-1}(\mathbf{x}, A) \geq\ & \tau^* \int\cdots\int \left[\prod_{j=1}^{N} K_{x_j}^{n_0}(x_j, \mathrm{d}x_j^{(n_0)})\right] \delta_{x_{N-1}^{(n_0)}}(\mathrm{d}x_N^{(n_1)}) \delta_{x_N^{(n_0)}}(\mathrm{d}x_{N-1}^{(n_1)}) \\
& \rho(x_{N-1}^{(n_0)}, x_N^{(n_0)}) \left[\prod_{j=1}^{N-2} \delta_{x_j^{(n_0)}}(\mathrm{d}x_j^{(n_1)})\right] \times \cdots \times \left[\prod_{j=1}^{N} K_{x_j}^{n_0}(x_j^{(n_N-2)}, \mathrm{d}x_j^{(n_{N-1}-1)})\right] \times \\
& \delta_{x_1^{(n_{N-1}-1)}}(\mathrm{d}x_N^{(n_N-1)}) \delta_{x_N^{(n_{N-1}-1)}}(\mathrm{d}x_1^{(n_N-1)}) \rho(x_1^{(n_{N-1}-1)}, x_N^{(n_{N-1}-1)}) \times \\
& \left[\prod_{j=2}^{N-1} \delta_{x_j^{(n_{N-1}-1)}}(\mathrm{d}x_j^{(n_N-1)})\right] \left[\prod_{j=1}^{N-1} K_{x_j}^{n_0}(x_j^{(n_N-1)}, A_j)\right] \epsilon\nu(A_N)
\end{aligned} \tag{7.5}$$

where

$$\rho(x_i, x_j) = \min\left\{1, \frac{\pi_i(x_j)\pi_j(x_i)}{\pi_i(x_i)\pi_j(x_j)}\right\}$$

with $i \neq j$, $i, j \in \{1, \ldots, N\}$, $n_l = l(n_0+1)$ (for some integer l) and $\tau^* = \tau^{Nn_0}(1-\tau)^{N-1}\{\prod_{i=1}^{N-1}\varepsilon_{iN}\}$.

Now, let us consider the integrals over $(x_1^{(n_1-1)}, x_1^{(n_1)}, \ldots, x_1^{(n_{N-1}-1)}, x_1^{(n_{N-1})})$ and $(x_N^{(n_{N-1}-1)}, x_N^{(n_{N-1})})$. Applying the inequality (7.3) (with suitable changes of subscripts) and splitting the integrals into $I_1 \times I_2$ as before, we obtain:

$$\begin{aligned}
I_1 &= \int_{\mathsf{X}} K_{x_1}^{n_0}(x_1, \mathrm{d}x_1^{(n_0)}) \int_{\mathsf{X}} \delta_{x_1^{(n_0)}}(\mathrm{d}x_1^{(n_1)}) \times \cdots \times \\
&\quad \int_{\mathsf{X}} K_{x_1}^{n_0}(x_1^{(n_N-2)}, \mathrm{d}x_1^{(n_{N-1}-1)}) \int_{\mathsf{X}} \delta_{x_1^{(n_{N-1}-1)}}(\mathrm{d}x_N^{(n_N-1)}) \\
I_2 &= \int_{\mathsf{X}^2} K_{x_N}^{(n_0)}(x_N^{(n_N-2)}, \mathrm{d}x_N^{(n_{N-1}-1)}) \delta_{x_N^{(n_{N-1}-1)}}(\mathrm{d}x_1^{(n_{N-1})}) \times \\
&\quad \min\left\{1, \frac{\pi_1(x_N^{(n_{N-1}-1)})}{\pi_N(x_N^{(n_{N-1}-1)})\varrho_1}\right\} K_{x_1}^{n_0}(x_1^{(n_{N-1})}, A_1).
\end{aligned}$$

Again, $I_1 = 1$ and

$$I_2 \geq \epsilon \int_{\mathsf{X}} \nu(\mathrm{d}x_N^{(n_{N-1}-1)}) \min\left\{1, \frac{\pi_1(x_N^{(n_{N-1}-1)})}{\pi_N(x_N^{(n_{N-1}-1)})\varrho_1}\right\} K_{x_1}^{n_0}(x_N^{(n_{N-1}-1)}, A_1).$$

This leaves (7.5) as

$$\begin{aligned}\widetilde{K}^{n_N-1}(\mathbf{x}, A) \geq\ & \tau^* \int \cdots \int \left[\prod_{j=2}^{N} K_{x_j}^{n_0}(x_j, \mathrm{d}x_j^{(n_0)})\right] \delta_{x_{N-1}^{(n_0)}}(\mathrm{d}x_N^{(n_1)}) \delta_{x_N^{(n_0)}}(\mathrm{d}x_{N-1}^{(n_1)}) \\ & \rho(x_{N-1}^{(n_0)}, x_N^{(n_0)}) \left[\prod_{j=2}^{N-2} \delta_{x_j^{(n_0)}}(\mathrm{d}x_j^{(n_1)})\right] \times \cdots \times \left[\prod_{j=2}^{N} K_{x_j}^{n_0}(x_j^{(n_{N-3})}, \mathrm{d}x_j^{(n_{N-2^{-1}})})\right] \times \\ & \delta_{x_2^{(n_{N-2}-1)}}(\mathrm{d}x_N^{(n_{N-2})}) \delta_{x_N^{(n_{N-2}-1)}}(\mathrm{d}x_2^{(n_{N-2})}) \rho(x_2^{(n_{N-2}-1)}, x_N^{(n_{N-2}-1)}) \times \\ & \left[\prod_{j=3}^{N-1} \delta_{x_j^{(n_{N-2}-1)}}(\mathrm{d}x_j^{(n_{N-2})})\right] \left[\prod_{j=2}^{N-1} K_{x_j}^{2n_0}(x_j^{(n_{N-2})}, A_j)\right] \epsilon^2 K_{(n_0,x_1)}^*(A_1) \nu(A_N)\end{aligned}$$

with $K_{(n_0,x_1)}^*(A_1) = \int_{\mathsf{X}} \nu(\mathrm{d}x) \min\left\{1, \frac{\pi_1(x)}{\pi_N(x)\varrho_1}\right\} K_{x_1}^{n_0}(x, A_1)$.

Applying the above argument recursively yields:

$$\widetilde{K}^{n_N-1}(\mathbf{x}, A) \geq \tau^* \epsilon^N K_{(n_0,x_1)}^*(A_1) K_{(2n_0,x_2)}^*(A_2) \times \cdots \times K_{((N-1)n_0,x_{N-1})}^*(A_{N-1}) \nu(A_N)$$

which holds $\forall \mathbf{x} \in \mathsf{X}^N$, $A \in \mathcal{B}(\mathsf{X}) \times \cdots \times \mathcal{B}(\mathsf{X})$. However, since we can exchange any pair in the exchange kernel we have:

$$\widetilde{K}^{n_N-1}(\mathbf{x}, A) \geq \theta \nu^*(A)$$

with $\theta = \tau^* \epsilon^N \phi$, $\phi = \int_{\mathsf{X}^{N-1}} \sum_{j=1}^{(N-1)!} \prod_{l=1}^{N-1} K_{(\sigma_j(l)n_0,x_l)}^*(\mathrm{d}x_l)$ (note $\phi \leq (N-1)!$ and σ is used to denote permutations here) and $\nu^*(A) = \phi^{-1} \nu(A_N) \sum_{j=1}^{(N-1)!} \prod_{l=1}^{N-1} K_{(\sigma_j(l)n_0,x_l)}^*(A_l)$. We thus have that X^N is $(n_N - 1, \theta, \nu^*)$ small.□

REFERENCES

Andrieu, C. & Robert C. P. (2001). Controlled MCMC for optimal sampling, Technical Report, Universitie Paris Dauphine.

Atachade, Y. F. & Liu J. S. (2004). The Wang-Landau algorithm for Monte Carlo computation on general state spaces, Technical Report, University of Ottawa.

Bozdech, Z., Llinas, M., Pullium, B. L., Wong, E. D., Zhu, J. & DeRisi, J. L. (2003). The transcriptome of the intraerythrocytic developmental cycle of *Plasmodium falciparum*. *PLoS Biol.* **1**, 1–16.

Brooks, S. P., Fan, Y., & Rosenthal, J. S. (2002). Perfect forward simulation via simulated tempering, Technical Report, University of Cambridge.

BROOKS, S. P., GIUDICI, P. & ROBERTS, G. O. (2003). Efficient construction of reversible jump Markov chain Monte Carlo proposal distributions (with Discussion). *J. R. Statist. Soc. B* **65**, 1–37.

CAPPÉ, O., ROBERT, C. P. & RYDÉN, T. (2003). Reversible jump, birth-and-death and more general continuous time Markov chain Monte Carlo samplers. *J. R. Statist. Soc. B* **65**, 679–700.

CARLIN, B. & CHIB, S. (1995). Bayesian model choice via Markov chain Monte Carlo methods. *J. R. Statist. Soc. B* **57**, 473–84.

CHAUVEAU, D. & VANDEKERKHOVE, P. (2002). Improving convergence of the Hastings and Metropolis algorithm with an adaptive proposal. *Scand. J. Statist.* **29**, 13–29.

DEL MORAL, P., DOUCET, A., & PETERS, G. W. (2004). Sequential Monte Carlo samplers, Technical Report, University of Cambridge.

DELLAPORTAS, P. & PAPAGEORGIOU, I. (2004). Multivariate mixtures of normals with an unknown number of components, Technical Report, Athens University.

DENISON, D. G. T., HOLMES, C. C., MALLICK, B. K. & SMITH, A. F. M. (2002). *Bayesian Methods for Nonlinear Classification and Regression.* Chichester: Wiley.

GEYER, C. (1991), Markov chain maximum likelihood, In *Computing Science and Statistics: The 23rd symposium on the interface*, (E. Keramigas ed), 156-63 Fairfax: Interface Foundation.

GEYER, C. & THOMPSON, E. A. (1995). Annealing Markov chain Monte Carlo with applications to ancestral inference. *J. Am. Statist. Assoc.* **90**, 909–20.

GILKS, W. R., ROBERTS, G. O. & GEORGE, E. I. (1994). Adaptive direction sampling. *The Statistician* **43**, 179–89.

GODSILL, S. (2001). On the relationship between MCMC methods for uncertainty. *J. Comput. Graph. Statist.* **10**, 230–48.

Goswami, G. R. & Liu J. S. (2005). On real parameter evolutionary Monte Carlo algorithm, Technical Report, Harvard University.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika* **82**, 711–32.

Green, P. J. (2003a). Discussion of Efficient construction of reversible jump Markov chain Monte Carlo proposal distributions. *J. R. Statist. Soc. B* **65**, 48-9.

Green, P. J. (2003b). Trans-dimensional Markov chain Monte Carlo. In *Highly Structured Stochastic Systems*, (P.J. Green, N.L. Hjort & S. Richardson eds), 179-96 Oxford: Oxford University Press.

Green, P. J. & Mira, A. (2001). Delayed rejection in reversible jump Metropolis-Hastings. *Biometrika* **88**, 1035–53.

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

Heard, N. A., Holmes, C. C., & Stephens, D. A. (2004). A quantitative study of gene regulation involved in the immune response of anopheline mosquitoes: An application of Bayesian hierarchical clustering of curves, Technical Report, Imperial College London.

Hodgson, M. E. A. (1999). A Bayesian restoration of an ion channel signal. *J. R. Statist. Soc. B* **61**, 95–114.

Hukushima, K. & Nemoto, K. (1996). Exchange Monte Carlo method and application to spin glass simulations. *J. Phys. Soc. Japan* **65**, 1604-08.

Jasra, A., Holmes, C. C., & Stephens, D. A. (2005). Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modelling, *Statist. Sci.*, To appear.

Jennison, C., Hurn, M. A. & Al-Awadhi, F. (2003). Discussion of Efficient construction of reversible jump Markov chain Monte Carlo proposal distributions. *J.*

R. Statist. Soc. B **65**, 44-5.

LIANG, F. & WONG, W. H. (2001). Real parameter evolutionary Monte Carlo with applications to Bayesian mixture models. *J. Am. Statist. Assoc.* **96**, 653–66.

LIU, J. S. (2001). *Monte Carlo Strategies in Scientific Computing.* New York: Springer.

LIU, J. S., LIANG, F. & WONG, W. H. (2001). A theory for dynamic weighting in Monte Carlo computation. *J. Am. Statist. Assoc.* **96**, 561–73.

MADRAS, N. & ZHENG, Z. (2003). On the swapping algorithm. *Random Structures and Algorithms* **22**, 66–97.

MCLACHLAN, G. J. & PEEL, D. (2000). *Finite Mixture Models.* Chichester: Wiley.

METROPOLIS, N., ROSENBLUTH, A. W., ROSENBLUTH, M. N., TELLER, A. H. & TELLER, E. (1953). Equations of state calculations by fast computing machines. *J. Chem. Phys.* **21**, 1087–92.

PESKUN, P. H. (1973). Optimum Monte Carlo using Markov chains. *Biometrika* **60**, 607–12.

RICHARDSON, S. & GREEN, P. J. (1997). On Bayesian analysis of mixture models with an unknown number of components (with Discussion). *J. R. Statist. Soc. B* **59**, 731–92.

ROBERT, C. P. & CASELLA G. (2004). *Monte Carlo Statistical Methods.* Second edition. New York: Springer.

ROBERTS, G. O. & ROSENTHAL, J. S. (2004). General state space Markov chains and MCMC algorithms. *Prob. Surveys*, to appear.

STEPHENS, M. (2000). Bayesian analysis of mixture models with an unknown number of components - an alternative to reversible jump methods. *Ann. Statist.* **28**, 40–74.

Tierney, L. (1998). Markov chains for exploring posterior distributions (with Discussion). *Ann. Statist.* **22**, 1701–62.

Tierney, L. (1998). A note on Metropolis-Hastings kernels for general state spaces. *Ann. Appl. Prob.* **8**, 1–9.

Yeung, K. Y., Fraley, C., Raftery, A. E. & Ruzzo, W. L. (2001). Model-based clustering and data transformations for gene expression data. *Bioinformatics* **17**, 977–87.