Efficient construction of reversible jump MCMC proposal distributions

S.P. Brooks†
University of Cambridge, UK
P. Giudici
University of Pavia, Italy
G.O. Roberts
Lancaster University, UK

Summary. The major implementational problem for reversible jump MCMC is that there is commonly no natural way to choose jump proposals since there is no Euclidean structure in the parameter space to guide our choice. In this paper we consider mechanisms for guiding the proposal choice. The first group of methods is based upon an analysis of acceptance probabilities for jumps. Essentially, these methods involve a Taylor series expansion of the acceptance probability around certain canonical jumps, and turns out to have close connections to Langevin algorithms. The second group of methods generalises the reversible jump algorithm using the so-called dual space approach. These allow the chain to retain some degree of memory so that when proposing to move from a smaller to a larger model, information is borrowed from the last time that the reverse move performed.

The main motivation for this paper is that, in very high dimensional examples, the probability that the Markov chains moves between such spaces may be prohibitively small, as the probability mass is very thinly spread across the space. Finding reasonable jump proposals becomes extremely important. We will illustrate the procedure using several examples of reversible jump MCMC applications including the analysis of autoregressive time series, graphical Gaussian modelling and mixture modelling.

Keywords: Bayesian model selection, Langevin algorithms, Optimal scaling, Autoregressive time series, Mixture modelling, Graphical models

1. Introduction

The reversible jump algorithm (Green 1995) is a Metropolis-Hastings algorithm designed for allowing movement between different dimensional spaces. Commonly these spaces are just competing models with different sets of parameters. Constructing appropriate classes of jumps is often difficult since there is no natural neighbourhood structure between models to guide us. Jumping into higher dimensional spaces involves the introduction of additional degrees of freedom, and it is necessary to impose some kind of constraints upon the possible jumps in order to compensate for this. The introduction of these constraints provides an opportunity to restrict the class of permissible jumps to preserve certain intuitively plausible quantities, such as moments, between the two models. This idea is known as moment-matching.

Even where appropriate moment-matching constructions have been successfully identified, there is still considerable freedom for choosing the proposal distribution.

1.1. Reversible jump MCMC

In this subsection, we shall introduce reversible jump MCMC in a fairly general setting. Though the introduction of the models is not strictly necessary at this stage, we shall suppose (in order to keep in mind the motivating example in Bayesian inference) that we have models $M_1, \ldots M_k, \ldots$, where model $M_i$ has a continuous parameter space. Note that the ideas that we develop can be easily extended to

†Address for correspondence: Steve Brooks, Statistical Laboratory, University of Cambridge, Wilberforce Road, Cambridge, CB3 0WB, UK
E-mail: steve@statslab.cam.ac.uk
the partially or totally discrete cases but we shall restrict our attention to only the continuous case in this paper.

We write \( \pi(M_i, \theta_i) \) for the density part of our target distribution \( \pi \) restricted to \( M_i \). Thus, for an arbitrary set \( B \),

\[
\pi(B) = \sum_i \int_{B \in \Theta_i} \pi(M_i, \theta_i) \, d\theta_i.
\]

We shall write the parameter space for \( M_i \) as \( \Theta_i \), and with a minor abuse of notation, we shall write \( \theta_i \) for a typical element of \( \Theta_i \). We shall focus upon moves between \( M_i \) and \( M_j \) with \( n_i < n_j \). By reversibility, this also characterises the reverse move, and moves between all collections of pairs of models can be dealt with similarly.

To move from model \( M_i \) to \( M_j \), we generate a random vector \( U \) of length \( n_j - n_i \) consisting of \( \hat{u} \) variables from some proposal density \( q(\cdot) \). We shall write the joint density of \( U \) as \( q_{n_j - n_i}(u) = \prod_{k=1}^{n_j - n_i} q(u_i) \). Having generated \( U \), we now propose the move from \( \theta_i \) to \( \theta_j = h_{i,j}(\theta_i, U) \), where the so-called jump function \( h_{i,j} : \Theta_i \times \mathbb{R}^{n_j - n_i} \rightarrow \Theta_j \) denotes an injection, mapping the current state of the chain together with the generated random vector to a point in the higher dimensional space. This move is then accepted with probability

\[
a((M_i, \theta_i), (M_j, \theta_j)) = \min[1, A_{i,j}(\theta_i, \theta_j)],
\]

where \( A \) takes the familiar form (Green 1995):

\[
A_{i,j}(\theta_i, \theta_j) = \frac{\pi(M_j, \theta_j) r_{j}(i)}{\pi(M_i, \theta_i) r_{i}(j) q_{n_j - n_i}(u)} \left| \frac{\partial h_{i,j}(\theta_i, u)}{\partial \theta_i(u)} \right|
\]

and \( r_{i}(j) \) denotes the probability that a proposed jump to model \( j \) is attempted at any particular iteration, starting from anywhere in \( \Theta_i \). In the case where \( n_i > n_j \), we just take

\[
A_{i,j}(\theta_i, \theta_j) = [A_{j,i}(\theta_j, \theta_i)]^{-1}
\]

In general, there may be several move types and a term for this would appear in the ratio.

Now consider our motivating example where \( \pi \) is the posterior distribution over a collection of models \( M_1, \ldots, M_k \), with prior model probabilities \( p(M_1), \ldots, p(M_k) \), and within model prior densities \( \{p(\theta_i), \theta_i \in \Theta_i \} \). Assume that within each model, \( M_i \), the likelihood is given by \( L_i(\text{data}|\theta_i) \). Then, if the target density is the relevant posterior density,

\[
\pi(M_i, \theta_i) \propto L_i(\text{data}|\theta_i)p(\theta_i)p(M_i)
\]

Many modifications, extensions and variations of reversible jump exist. For instance, a convenient mathematical framework for jumping between different dimensional spaces is to let the union of all state spaces be described as a marked Poisson processes. To move around such a space, it is natural to use birth and death processes (see for example Preston 1977, and Ripley 1977) and this is the approach used to simulate interacting spatial point processes in Geyer and Møller (1994) and the considerable body of work that leads from this (see for example the review of Kendall and Thonnes 1999). This approach was also used in Stephens (2000) and applied effectively to the problem of Bayesian inference for mixtures with an unknown number of components.

An approach to trans-model dynamics, in which the algorithm stores a vector for each model at every iteration has been introduced by Carlin and Chib (1995), and recently extended to a general framework which also includes the reversible jump algorithm as described above by Godsill (2000). As a direct extension of reversible jump, Green and Mira (2000) introduce a procedure for reassigning rejected moves by sequentially attempting further proposals using a modification of the usual accept reject mechanism. This procedure allows some level of adaptation in the sense that later proposals at any particular iteration can be allowed to use information gained from earlier rejections within that iteration. These and other related advances are reviewed in more detail in Green (2000).

All of these extended methodologies offer alternative frameworks in which to construct Markov chain dynamics to explore across models. However beyond that, there has been little progress in the problem of exactly how to construct proposal distributions. For algorithms on Euclidean spaces, the metric structure of the state space guides proposal construction. For instance for the random walk
Efficient RJMCMC Proposals

Metropolis algorithm on a continuous target density, using a proposal distribution with a variance \( \sigma^2 \), very small values of \( \sigma^2 \) will lead to small jumps which are almost all accepted, while large values will lead to an excessively high rejection probability for proposed moves. Thus, the scaling problem will typically have an optimal value for the proposal scale \( \sigma \) which lies in between these two extremes. Whilst, proposal distributions can be refined in more subtle ways than by variance alterations, the restriction of the proposal choice problem to this one dimensional problem is appealing, works well in practice, and is supported in part by currently available theory (see for example Roberts et al 1997). The problem for reversible jump moves, is that there is no direct analogue of this kind of scaling problem, since there is no natural notion of a “local” move with an arbitrary high acceptance probability. The approach of this paper, is to try to translate natural ideas for proposal construction (such as the scaling problem) from their natural Euclidean environment to the union of model spaces.

1.2. Example: Auto-regressive model choice

In order to illustrate and motivate the methodology that we shall propose, we will use a simple example of model choice for auto-regressive time series models of unknown order. Here we will describe a standard implementation of reversible jump to this problem (see for example Godsill 2000; and Ehlers and Brooks 2000).

Suppose that we have data \( x_1, \ldots, x_T \) from an autoregressive process of unknown order. Let model \( M_k \) correspond to the \( k \)th order AR process which is specified by the relational formula

\[ X_t = \sum_{\tau=1}^{k} a_k X_{t-\tau} + \epsilon_t, \quad t = k_{max}, \ldots, T \]

for \( k = 1, 2, \ldots, k_{max} \), where \( \epsilon_t \sim N(0, \sigma^2) \). In order to model the data, we assume a uniform prior for \( k \) and within model \( M_k \) we take independent \( N(0, \sigma_k^2) \) priors for the coefficients \( a_i, i = 1, \ldots, k \) and we assume that \( \sigma^2 \) is known a priori. Therefore, the number of parameters in model \( M_k \) is given by \( n_k = k \) and we can write \( \theta_k = (a_1, a_2, \ldots, a_k) \). The posterior therefore comprises three terms, the likelihood:

\[ L_k(x|\theta) = \prod_{t=k_{max}+1}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left(-\frac{1}{2} \frac{\left[x_t - \sum_{\tau=1}^{k} a_\tau x_{t-\tau}\right]^2}{\sigma^2}\right), \]

the prior for the parameters:

\[ p(\theta_i) = (2\pi\sigma_0^2)^{k/2} \exp(-\frac{1}{2\sigma_0^2}\theta_i^T \theta_i), \]

and the prior for the models:

\[ p(M_i) = \frac{1}{k_{max}}, \quad i = 1, \ldots, k_{max}. \]

In this particular example, jumps only take place between nested models, i.e., when jumping between model \( M_k \) and model \( M_{k+1} \) say, the value and interpretation of the first \( k \) parameters may remain unchanged. This nesting structure is often but not always present and the natural function linking the two parameter spaces, which we shall term the canonical jump function, is the identity, so that

\[ h_{k, k+1}(\theta_i, u) = (\theta_i, u) \]

that is \( a_{k+1} = u \). In this case, the determinant term in the definition of \( A_{i,j} \) in (1) is simply 1.

Though larger jumps are possible, we shall continue to concentrate upon jumps which only alter dimensionality by 1. Suppose that at any iteration we propose a jump from \( \theta_k \) in \( M_k \) to \( \theta_{k+1} = (\theta_k, u) \) in \( M_{k+1} \) with probability \( \beta \), otherwise we propose a jump which decreases the dimension of the model by 1. If we propose to increase dimension, we might take as our proposal \( u \sim N(0,1) \) we will look at more general proposals later on) and applying (2) to (1) we obtain

\[ A_{k, k+1}(\theta_k, \theta_{k+1}) = \frac{L_{k+1}(data|\theta_{k+1}, u) p_{k+1}(\theta_{k+1}) (1 - \beta)}{L_k(data|\theta_k) p_k(\theta_k)} \frac{1}{\beta} \frac{1}{q_1(u)}, \]
where \( q_1(u) = (2\pi)^{-1/2} \exp[-u^2/2] \). Clearly, since we have independent priors for the model parameters, the prior term in the denominator will cancel with the corresponding terms in the numerator.

Though repeated application of reversible jump moves of this type are sufficient to generate a Markov chain with the correct stationary distribution, a typical implementation would also involve the usual Metropolis-Hastings move to update the parameters within the current model. Clearly the performance of this chain will depend upon our choice of proposal parameters. In particular the ability to mix between models will depend heavily upon our choice for \( \sigma_n^2 \). If we set the proposal variance to be too small we will typically find it difficult to jump from larger models to smaller ones, since the \( q \) term (which would lie in the numerator for this move) would generally be small. By reversibility, the algorithm would show an equal reluctance to perform the reverse move also. Similarly, if \( \sigma_n^2 \) is too large, we will find it hard to accept moves from smaller models to larger ones since the proposed values of new parameters will tend to lie far from areas of high posterior support. In practice, the proposal variance is “tuned” on the basis of short pilot runs but for more complex problems, this can be a difficult and time-consuming process.

1.3. General perspective
Reversible jump MCMC is a very general and widely applicable technique. Markov chain convergence is assured under very weak conditions on the jump function \( h_{i,j} \) and the proposal density \( q \). However, very little is known about how to do this efficiently in a generic way. Whilst there has been considerable progress in this area for fixed-dimension sampling problems (see e.g. Gelman et al 1996; Roberts and Rosenthal 1998), most of the available statistical applications of reversible jump techniques rely upon various strategies of empirical tuning, as discussed above. In this paper we discuss a variety of general recipes for choosing the jump function \( h_{i,j} \) and for automatically scaling the proposal distributions, \( q \).

In Section 2, we develop a framework which allows us to discuss current and new methods for constructing reversible jumps. In particular, we will consider the concept of moment matching (Green 1995), and we introduce the idea of “centering” reversible jumps upon moves which take advantage of weak non-identifiability (where \( \theta_i \) and \( \theta_j \) give rise to the same probability model for \( \theta_i \in \Theta_i \) and \( \theta_j \in \Theta_j \)).

One of the main contributions of this paper is the introduction of an automatic method for determining proposal scales, based upon an analysis of acceptance probabilities for these jumps. The ideas of locating and scaling proposal distributions is described in detail, and all the ideas are illustrated using the simple auto-regressive model choice example introduced above. In Section 3 we extend this approach to consider more general analyses of the acceptance probabilities for determining automatic scales and provide some technical discussion on the advantages of adopting these higher order approaches. We also introduce some alternative centering methods.

Throughout the paper, we make connections to more familiar algorithms on Euclidean spaces, such as the random walk Metropolis algorithm and Langevin algorithms. This is because we are able to describe such algorithms in terms of conditions upon the acceptance probability formula, and these notions can be easily transported to the reversible jump context. Specifically, the random walk Metropolis algorithm on a Euclidean space has accept-reject ratio, \( A(x, y) \) (which equals \( \pi(y)/\pi(x) \) in this case) equal to 1 for \( y = x \). We call such an algorithm a zeroth order algorithm. In the reversible jump context, we can similarly attempt to obtain \( A(x, y) = 1 \) for a certain canonical jump from a point \( x \) in one space, and \( y \) in another. Similarly, Langevin algorithms can be characterised by the additional condition that \( \frac{\delta A}{\delta y} \bigg|_{y=x} = 0 \). We will define such an algorithm as a first order algorithm, and the definition extends naturally to the reversible jump case as in the first order case.

In Section 4 we provide a generalisation of the reversible jump algorithm using the so-called dual space approach. This method allows the random seeds generated in order to increase the dimensionality of the current model, to be retained on the subsequent simplification of the model. Through artificially introduced dependences between these random seeds, algorithms with a kind of momentum through model space can be generated, potentially aiding mixing in hard problems with discrete multi-modality in model space. We provide examples of the implementation of these methods and illustrate the dramatic improvement in performance that we observe in the context of several real applications.
There are three distinct settings which together describe the vast majority of applications of reversible jump MCMC: variable selection, where we decide which parameters should be included in a model; association selection, where we decide which interactions exist between a fixed number of model parameters; and finally classification problems, where we decide how to assign observations to different groups within a model. As discussed above we adopt a running example based upon variable selection for autoregressive models. The implementational details for this example are discussed throughout the text and in Section 5.1 we report the results of the application of these algorithms in the context of a particular data set. To further illustrate the applicability and utility of these methods we also introduce more applications. In Section 6 we introduce an example of association selection in the context of graphical Gaussian modelling and, in Section 7 we introduce a classification problem by re-examining the mixture modelling problem of Richardson and Green (1997) showing how sampler performance can be improved through the adoption of these methods.

2. Centering and Scaling Proposals

Reversible jump algorithms offer no real mathematical generalisation over traditional Metropolis-Hastings methods. They can be thought of as ways of constructing proposals to move around a rather more complicated space than the usual $\mathbb{R}^k$ (see for example Tierney 1998). So what makes it intrinsically more difficult than the traditional approach?

To answer this, consider the case where, for this example only, $\pi$ is a continuous density on $\mathbb{R}^k$. The vanilla (or default) algorithm we might use to explore $\pi$ is the random walk Metropolis algorithm, perhaps proposing a move from $\mathbf{x}$ to $\mathbf{x} + \mathbf{U}$ where $\mathbf{U} \sim N(0, \sigma^2)$ say. The analogue of the acceptance ratio (1) in this context is simply

$$A(\mathbf{x}, \mathbf{x} + \mathbf{U}) = \frac{\pi(\mathbf{x} + \mathbf{U})}{\pi(\mathbf{x})}.$$ 

Here $A$ takes the value 1 at a central move corresponding to $\mathbf{U}$ taking the value zero and therefore proposing to remain at $\mathbf{x}$. By continuity therefore, small jumps (that is where $|\mathbf{U}|$ is small) are accepted with a very high probability, and importantly so is the reverse move. Thus, depending upon $\pi$, a sufficiently small $\sigma^2$ defines an algorithm with a high enough acceptance rate to allow the chain to move around the state space (though sometimes perhaps rather slowly). See Gelman et al (1996) and Roberts et al (1996) for results describing the relationship between the optimal choice of $\sigma^2$ (in terms of the corresponding convergence rate of the chain) and the acceptance rate. Whilst none of this guarantees a useful algorithm in practice, this all makes the tuning of algorithms (choosing $\sigma^2$) relatively straightforward.

The existence of a central move where $A = 1$ is certainly not guaranteed in the reversible jump framework, and the aims of the simplest algorithms we shall introduce will be to ensure that it does.

More sophisticated algorithms for target densities on $\mathbb{R}^k$ can improve efficiency over random walk Metropolis algorithms drastically. So called Langevin algorithms (see for example Roberts and Tweedie 1996) use gradient information about $\pi$ to propose candidate moves which are more likely to be accepted. Specifically, given $\mathbf{x}$ the algorithm proposes a move to $\mathbf{x} + \mathbf{U}$ where

$$\mathbf{U} \sim N(0 + \sigma^2 \nabla \log \pi(\mathbf{x}), \sigma^2).$$

The usual way to motivate this choice of proposal is to consider the discretisation of a suitable Langevin diffusion which has $\pi$ as its stationary distribution. In the context of choosing sensible proposals, we note that this proposal leads to an acceptance ratio $A$ which satisfies $A(\mathbf{x}, \mathbf{x}) = 1$ and

$$\frac{\partial A(\mathbf{x}, \mathbf{y})}{\partial \mathbf{y}} \bigg|_{\mathbf{y} = \mathbf{x}} = 0.$$ 

Intuitively, this allows $A$ to be close to one further away from $\mathbf{x}$ than for the random walk Metropolis case, therefore allowing more ambitious jumps to be proposed and accepted. This leads to improved mixing (see Roberts and Rosenthal 1998 for theoretical results on this).

In the reversible jump context, we shall show that it is possible to construct algorithms which mimic the behaviour of these examples in terms of the characterisation given in (3). In fact even higher
order Langevin algorithms have analogues in the reversible jump context and we shall demonstrate how these can be constructed in Section 3.

2.1. Constructing the jump function
We return to the general setting of Section 1.1. Within this framework, the jump function $h_{i,j}$ can be chosen arbitrarily though often some advantage is obtained by exploiting features such as nesting and moment matching between models.

Example: nested jumps.
Suppose $\theta_i = \mu_1, \ldots, \mu_n$, for a collection of parameters $\mu_i$ whose interpretation is unaltered in models of increased complexity. Then the natural constraints when proposing a move to a more complex model, are to preserve the values of all current parameters. In this case, the jump function is simply the identity.

Example: moment matching.
Suppose $\theta_i = (\mu_{i,1}, \ldots, \mu_{i,n})$ and $\theta_j = (\mu_{j,1}, \ldots, \mu_{j,n})$. $h_{i,j}$ is often chosen with the aid of an $(n_j - n_i)$ dimensional constraint. For example suppose $A$ is an $n_i \times n_j$ matrix, then we might wish to impose the condition that $A\theta_j = \theta_i$. The simplest example of this occurs where $n_j = n_i + 1$ and the dimension matching constraints are $\mu_{i,k} = \mu_{j,k}$, $k = 1, 2, \ldots, n_i - 1$ and $(\mu_{j,n_i} + \mu_{j,n_i+1})/2 = \mu_{i,n_i}$. The corresponding jump function would then be to set

$$h_{i,j}(\theta_i, u) = (\mu_{i,1}, \ldots, \mu_{i,n_i-1}, \mu_{i,n_i} + u, \mu_{i,n_i} + u)$$

As it happens, even for a given collection of dimension matching constraints, there is still some flexibility in how to choose the jump function. We shall assume here, that we are given some collection of canonical jump functions $\{f_{i,j}\}$, $f_{i,j}: \Theta_i \times \mathbb{R}^{n_j-n_i} \rightarrow \Theta_j$. For instance in the nested case above, it is natural to take $f_{i,j}$ to be the identity function. We then set

$$h_{i,j}(\theta_i, u) = f_{i,j}(\theta_i, k_{i,j}\theta_i(u))$$

where $k_{i,j}\theta_i$ plays the role of a general scaling function. We shall typically choose $k_{i,j}\theta_i$ from a small dimensional family of functions, all we shall insist upon is that $k_{i,j}\theta_i$ is invertible.

Let us quickly return to our autoregressive example. Suppose we wish to move from $\theta_k$ to $\theta_{k+1}$ by generating a random value $u \sim N(0,1)$. Since the models here are nested, the natural choice is to adopt the identity as the jump function $f_{i,j}$ and to take a simple linear scaling function $k_{i,j}\theta_i(u) = \sigma_u u$, say equivalent to taking a proposal of the form $U \sim N(0, \sigma_u^2)$. This provides the same jump transition as described in Section 1.2 if $\sigma_u = 1$.

The function $f_{i,j}$ will be considered fixed in our search for an automatic proposal mechanism. However, we still retain considerable freedom for the choice of $k_{i,j}\theta_i$. This and the following section will largely concentrate upon strategies for the automatic choice of $k_{i,j}\theta_i$, which can be adopted without additional computation and complexity inherent in the pilot tuning required for many existing reversible jump applications.

2.2. Centering proposals
For the introduction of our mechanism for constructing reversible jump algorithms outside the simple nested cases, we require the introduction of a collection of centering functions. A centering function $c_{i,j}: \Theta_i \rightarrow \Theta_j$ can be specified by the equation

$$c_{i,j}(\theta_i) = f_{i,j}(\theta_i, b_{i,j}(\theta_i))$$

where $b_{i,j}(\theta_i)$ is some pre-determined value often taken to be zero. Essentially, we are identifying a “special” value $b_{i,j}(\theta_i)$ for the proposal vector $u$ and associating this with a point in the higher dimensional space, $c_{i,j}$. We shall require such a function between each collection of models for which we might attempt to jump. Intuitively the centering function should choose a “representative point” on the image of $h_{i,j}(\theta_i, \theta_j)$ from which to extract information for the construction of $k_{i,j}\theta_i$. 

---

*Brooks, Giudici and Roberts*
2.4. The zeroth order approximation

2.3. State dependent proposal density

There are a number of possible strategies for constructing the centering functions, using statistical or mathematical principles. For instance the centering function can often be defined according to weak non-identifiability, where the probability model described by \( \theta_i \) in \( \Theta_i \) is identical to that described by \( c_{i,j}(\theta_i) \). As an illustration, in the auto-regressive example of Section 1.2, the weak non-identifiability centering for a move between \( M_k \) and \( M_{k+1} \) is characterised by \( b_{k,k+1} = 0 \) so that \( c_{k,k+1}(\theta_i) = (\theta_i,0) \), since the \( k \)-dimensional model with parameters \((a_1, \ldots, a_k)\) is identical (in terms of likelihood contribution) to the \( k+1 \) dimensional model with parameters \((a_1, \ldots, a_k, 0)\). Thus, it would be natural to center any proposal for the parameter vector in the \( k+1 \)-dimensional model about this point which corresponds to the \( k \)-dimensional model in the higher dimensional space.

In order to choose \( \{k_{i,j}(\theta_i)\} \) for any particular problem, we propose a hierarchy of strategies in terms of both ease of implementation and accuracy (ease decreases with increasing accuracy). Before we discuss these strategies in detail, let us first introduce an alternative formulation of the approach described above.

2.3. State dependent proposal density

There is an alternative (and essentially equivalent) formulation of (4) which replaces the notion of modifying the canonical jump function with the idea of a modification of the proposal distribution itself. Let

\[
k_{i,j}(\theta_i, U_1, \ldots, U_{n_j-n_i}) = (F_{r;i,j}^{-1}(F_{r;i,j}(U_1)), \ldots, F_{r;i,j}^{-1}(F_{r;i,j}(U_{n_j-n_i}))),
\]

where \( F_{r;i,j}(\theta_i) \) is the distribution function of a one dimensional distribution, which can depend upon \( \theta_i \). If \( U \) is drawn from \( \eta_{n_j-n_i} \), then \( k_{i,j}(\theta_i, U) \) consists of independent components \((V_1, \ldots, V_{n_j-n_i})\) with \( V_r \) having distribution function \( F_{r;i,j}(\theta_i) \). In other words the algorithm now merely inputs differently distributed random variables into the canonical jump function.

In order to apply (1) we need to evaluate the Jacobian term. It is easily verified that in this case,

\[
\left| \frac{\partial h_{i,j}(\theta_i, u)}{\partial (\theta_i, v)} \right| = \left| \frac{\partial f_{i,j}(\theta_i, v)}{\partial (\theta_i, v)} \right|_{v=k} \times \left| \frac{\partial k_{i,j}(\theta_i)(u)}{\partial u} \right|.
\]  

Note that the first term on the right hand side of (6) is exactly that used in ordinary reversible jump using the canonical jump functions \( \{f_{i,j}\} \). Investigating the second term using (5),

\[
\left| \frac{\partial k_{i,j}(\theta_i)(u)}{\partial (u)} \right| = \left( \prod_{r=1}^{n_j-n_i} f_{r;i,j}(F_{r;i,j}(F_{r;i,j}(U_r))) \right)^{-1}
\]

where \( f_{r;i,j}(\theta_i) \) denotes the density corresponding to the distribution function \( F_{r;i,j}(\theta_i) \). Thus, in this context we can rewrite (1) as

\[
A_{i,j}(\theta_i, \theta_j) = \frac{\pi(M_i, \theta_j) n_j(i)}{\pi(M_i, \theta_i) n_i(j) \prod_{r=1}^{n_j-n_i} f_{r;i,j}(\theta_i)(v)} \left| \frac{\partial f_{i,j}(\theta_i, v)}{\partial (\theta_i, v)} \right|.
\]

where \( v \) has initially been drawn from the distribution with independent components with respective distribution functions \( F_{r;i,j}(\theta_i) \), and \( \theta_j \) is constructed from (4) and (5).

2.4. The zeroth order approximation

The simplest of our methods allows only one degree of freedom and would usually be used to fix a scale parameter for \( q_j \) having first fixed the location. Suppose that we are currently in state \( \theta_i \) then we wish to choose a scale for the proposal \( q \) that we will use to generate a state in the new model, \( \theta_j \). We choose the proposal scale so that for the jump between \( \theta_i \) and its image in \( \Theta_j \) under the centering function \( c_{i,j}(\theta_i) \), the acceptance ratio given in (1) is identically equal to one i.e.,

\[
A_{i,j}(\theta_i, c_{i,j}(\theta_i)) = 1,
\]
More specifically, suppose we constrain \( k_{i,j} \theta_i \) to belong to the scaling family

\[
k_{i,j} \theta_i (u) = R_{i,j} \theta_i \times u.
\]

where \( R_{i,j} \theta_i \) is a state dependent scale parameter. Then, re-arranging (7) using the definition of \( A_{i,j} \) in (1) and \( k_{i,j} \theta_i \) in (8), we get

\[
R_{i,j} = \frac{\pi(M_j, \theta_j) r_{i,j}(j) q(b_{i,j}^{-1}(\theta_i))}{\pi(M_j, c_{i,j}(\theta_i)) r_{i,j}(i)} \frac{1}{J_i(\theta_i, u)}
\]

(9)
giving a solution \( R(= R_{i,j} \theta_i) \). Here we write \( J_i(\theta_i, u) \) for \( \begin{vmatrix} \frac{\partial h_{i,j}(\theta_i, u)}{\partial \theta_i} \\ \frac{\partial h_{i,j}(\theta_i, u)}{\partial \theta_i} \end{vmatrix} \bigg|_{u = b_{i,j}(\theta_i)/k_{i,j} \theta_i} \).

Setting \( A_{i,j}(\theta_i, \theta_j) = 1 \) for certain “central” jumps is a sound heuristic principle. For general MCMC algorithms, it is often automatically satisfied. For instance, in Euclidean spaces the random walk Metropolis algorithm on a continuous density will have \( A_{i,j} \) close to 1 for all small enough jumps. Moreover, more sophisticated algorithms (for instance the hybrid Monte Carlo procedure of Duane et al. 1987) can often be motivated in terms of arguments that try to fix \( A_{i,j} \) to be equal to, or approximately equal to 1, for appropriately chosen jumps. For reversible jump algorithms, the lack of Euclidean structure in the state space means that obtaining \( A_{i,j}(\theta_i, \theta_j) = 1 \) for appropriate jumps is not automatic. The zeroth order method described above ensures that the acceptance probability does equal one for centered jumps between \( \theta_i \) and \( c_{i,j}(\theta_i) \).

2.5. Applying the zeroth order method to Bayesian applications

To apply our procedure, we need to specify the functions \( h_{i,j} \) and \( c_{i,j}(\theta_i) \), or equivalently the proposal distributions \( F_{r_{i,j} \theta_i} \), for all collections of models \( M_i \) and \( M_j \) between which jumps might conceivably be proposed. A useful strategy for specifying \( c_{i,j}(\cdot) \) takes advantage of weak non-identifiability in the following way. Suppose there exists \( b_{i,j}(\theta_i) \) such that \( L(\text{data}|\theta_i) = L(\text{data}|h_{i,j}(\theta_i, b_{i,j}(\theta_i))) \), then set \( c_{i,j}(\cdot) = f_{i,j}(\theta_i, b_{i,j}(\theta_i)). \) A location shift in \( h_{i,j} \) ensures that there is no loss of generality in taking \( b_{i,j}(\theta_i) = 0 \). We define this method for choosing the centering functions as non-identifiability centering.

Using non-identifiability centering, (9) reduces to

\[
R_{i,j} = \frac{p_i(\theta_i) p(M_j) r_{i,j}(j) q(c_{i,j}(\theta_i))}{p_j(\theta_j) p(M_i) r_{i,j}(i)} \frac{1}{J_i(\theta_i)}
\]

Note that this is independent of the likelihood function entirely. Essentially, the zeroth order method is approximating the posterior distribution by the prior in order to determine a sensible scale for the proposal. The zeroth order method is therefore unable to adapt to the data. However, the advantage of the zeroth order approach over some of the more complex approaches developed in the next section is its inherent simplicity. Clearly, when the priors are informative, either in terms of precision or structural complexity, the zeroth order may perform well. However, the method may perform poorly when we adopt more vague priors as in the case, the prior and posterior distributions will typically be quite different. In such cases, the method could be improved if we can also incorporate information from the data in choosing the proposal scales. A natural way to do this is to consider higher order approximations. Alternatively, if the simplicity of the zeroth order method is to be maintained, we suggest using the conditional maximisation approach. Both of these approaches are described in the next section.

Before we describe these extensions, let us return to the autoregressive example to illustrate how the zeroth order method may be applied.

Example: the autoregressive example revisited

Applying non-identifiable centering and the zeroth order algorithm to the auto-regressive example introduced in Section 1.2, and adopting the identity jump function \( f_{i,j} \) and simple linear scale function \( k_{i,j} \theta_i (u) = u \sigma_u \) as discussed in Section 2.1, the acceptance ratio in (1) reduces to

\[
A_{k,k+1} (\theta_i, \theta_j) = \frac{p_{k+1}(\theta_{k+1})}{p_k(\theta_k)} \frac{p_k(M_k)}{p_{k+1}(M_{k+1})} \frac{1}{r_k(k)} \frac{r_{k+1}(k+1)}{r_{k+1}(k)} \frac{q(u)}{q(u)}
\]

(10)
where
\[
\frac{p_{k+1}(\theta_k, u)}{p_k(\theta_k)} = \left(2\pi\sigma^2_u\right)^{-1/2} \exp\{-u^2/(2\sigma^2_u)\},
\]
\[
p(M_k) = p(M_{k+1}) = \frac{1}{k_{max}},
\]
\[
r_{k+1}(k) = \begin{cases} 
0 & k = 0 \\
\frac{1}{\beta} & 0 < k < k_{max} - 1 \\
1 & k = k_{max} - 1
\end{cases}
\]
\[
r_k(k + 1) = \begin{cases} 
0 & k = k_{max} \\
\frac{1}{\beta} & 0 < k < k_{max} \quad \text{and}
1 & k = 1
\end{cases}
\]
\[
q(u) = \left(2\pi\sigma^2_u\right)^{-1/2} \exp\{-u^2/(2\sigma^2_u)\}.
\]

Therefore, equation (10) becomes
\[
A_{k,k+1}(\theta_i, \theta_j) = \frac{(2\pi\sigma^2_u)^{-1/2} \exp\{-u^2/(2\sigma^2_u)\} r_{k+1}(k)}{(2\pi\sigma^2_u)^{-1/2} \exp\{-u^2/(2\sigma^2_u)\} r_k(k+1)}
\]  

From Section 2.2 we know that we should centre at \( u = b_{ij}(\theta_i) = 0 \) so that the general zeroth order equation (7) becomes
\[
A_{k,k+1}[\theta_i, (\theta_i, 0)] = \frac{(2\pi\sigma^2_u)^{-1/2} r_{k+1}(k)}{(2\pi\sigma^2_u)^{-1/2} r_k(k+1)} = 1,
\]
which can be solved to obtain
\[
\sigma^2_u = \sigma^2_u \left(\frac{r_{k+1}(k)}{r_k(k+1)}\right)^2.
\]

Clearly, \( \sigma^2_u \) is independent of the data and so only information from the prior is used to tune the proposal distribution. The further the prior distribution is from the posterior, the lower the reliability of this automatic procedure for increasing the MCMC mixing due to the tuning of the jump proposal distribution. Thus, higher order approximations which involve the likelihood term at the centering point may provide better performance.

3. Higher order approximations

The zeroth order method can naturally be extended by considering higher order expansion terms. The idea of trying to obtain \( A_{i,j} \) as close to unity as possible can be used to motivate more sophisticated proposal choices. One natural way to extend the method is to not only fix \( A_{i,j} \) to take the value 1 at some chosen "central" value, but also to stipulate that some of its derivatives be zero at that central value also.

3.1. The first order method

We now describe the simplest possible approximation of this type, namely, a first order approximation which we shall call the first order algorithm. With \( A_{i,j} \) defined as in (1), this method satisfies both equation (7) and
\[
\nabla A_{i,j}[\theta_i, c_{i,j}(\theta_i)] = 0
\]  

for all possible choices of \( i, j \) and \( \theta_i \). Here \( \nabla \) is taken with respect to \( \theta \), and therefore (13) imposes a \( n_j - n_i \)-dimensional constraint upon the proposal. In practice it is often easier to specify the derivative constraints in terms of the log of the acceptance ratio, so it may be easier to impose the constraint that
\[
\nabla \log A_{i,j}[\theta_i, c_{i,j}(\theta_i)] = 0
\]

This method can be thought of as the reversible jump analogue of Langevin algorithms. Such algorithms on a Euclidean space are characterised by (13) with \( c_{i,j}(\theta_i) = \theta_i \). Langevin algorithms
tend to have considerably superior convergence properties than simpler zeroth order methods (see for example Roberts and Rosenthal 1998). This is because the algorithm takes into account local fluctuations in the shape of the target density and adjusts the target as a result. In addition, since the acceptance ratio is 1 except for a quadratic error (as opposed to linear in the zeroth order case), larger jumps can be attempted without leading to acceptance rates close to zero.

As with the zeroth order algorithm, we shall fix $h_{i,j}$ and try to find a proposal density to satisfy both (7) and (14) simultaneously. Of course there are many different ways to do that. We give the following approach as an example.

**Example: triangular proposals**

Suppose that $n_{j} - n_{i} = 1$. Then (7) and (14) together introduce a two-dimensional constraint upon the proposal density. To satisfy these two conditions we only need to consider a class of distributions with two degrees of freedom. There are many possible choices for this, but we shall illustrate the idea with one of the most natural choices.

Recall first of all that $h_{i,j}(\theta_{i}, u) = f_{i,j}[\theta_{i}, k_{i,j}\theta_{i}(u)]$, so that the value of $u$ corresponding to the centering point is that which satisfies the condition that $k_{i,j}\theta_{i}(u) = b_{i,j}(\theta_{i})$. Without loss of generality, let us assume that this value of $u$ is zero since there is always a simple linear transformation of the state space that ensures this to be so.

Now, consider triangular densities, satisfying

$$q(u) = R^{-1} + \gamma u, \quad u \in [a, b]$$

(15)

where $R$ is chosen from the zeroth order analysis in order that $q$ satisfies (7). Clearly, at the centering point (corresponding to $u = 0$), the proposal density in (15) is independent of the value of $\gamma$. Thus, the value of $R$ can be obtained from the zeroth order formula which will hold for all values of $\gamma$. The value of $\gamma$ can then be chosen to satisfy the first order formula given this value of $R$.

Since

$$\log A_{i,j}(\theta_{i}, \theta_{j}) = \log \pi(M_{j}, \theta_{j}) + \log J_{i,j}(\theta_{i}, u) + q(u) + \text{const.},$$

it is easy to show that

$$\gamma = \frac{1}{R} \nabla \left( \log(p_{j}(M_{j}, h_{i,j}(\theta_{j}, u))) \left| \frac{\partial h_{i,j}(\theta_{i}, u)}{\partial \theta_{i}} \right|_{u=0} \right)$$

satisfies (14).

We will choose $a$ and $b$ so that the interval $[a, b]$ contains the centering point i.e., $a \leq 0 \leq b$. If $|\gamma| \geq 1/(2R^2)$, then we can choose one of $a$ and $b$ (a say, which makes $\gamma$ positive) and fix the proposal density at that point to be zero i.e., $q(a) = 0$. Then the other endpoint is chosen uniquely in order to make $q$ a probability density function i.e.,

$$b = \frac{\sqrt{2}\gamma R - 1}{\gamma R}.$$

If $|\gamma| < 1/(2R^2)$ the above procedure is not possible since in this case, $a < b < 0$ so that the centering point is not even in the region to be sampled. This is clearly undesirable, and since in this case, the gradient has a small magnitude, a sensible alternative is to just replace the proposal by a uniform proposal centered about the centering point, that is $q \sim U(-R/2, R/2)$

We remark that higher order methods will often increase the complexity of the implementation in that several equations need to be solved simultaneously in order to obtain the proposal parameters. The triangular density approach presented above provides a simple generic method which would work for arbitrarily complex problems for which the derivatives of $A_{i,j}$ are available. Essentially, the stepwise nature of this approach allows us to consider the equations one at a time and can be extended for higher order approximations (where higher order derivatives of $A_{i,j}$ are considered).
3.2. Other approaches

Of course the extension to higher order methods need not stop at the first derivative. We may also set higher order derivatives to zero. Clearly as we set more and more derivatives to zero we get acceptance probabilities which become closer and closer to one. Thus, the infinite order method in which all derivatives are set to zero would be equivalent to the Gibbs sampler in the model jumping case i.e., the acceptance probability would be identically one and we would no longer need to perform the accept-reject step.

Of course, higher order methods will often increase the complexity of the implementation in that several equations need to be solved simultaneously in order to obtain the proposal parameters. However, the triangular density approach presented above provides a simple generic method which would work for arbitrarily complex problems for which the derivatives of $A_{i,j}$ are available. Essentially, the stepwise nature of this approach allows us to consider the equations one at a time and can be extended for higher order approximations (where higher order derivatives of $A_{i,j}$ are considered). Of course this approach is not the only one and, in some cases, more direct approaches are possible as we shall when we return to the autoregressive example in Section 3.3.

In practice of course, our proposal density will typically have only a few parameters which need to be selected. Given a proposal with $r$ parameters we therefore only need $r$ constraints in order to specify those parameters. If we add additional constraints, then it may not be possible to solve all of them. As we shall see in the context of the autoregressive example in Section 3.3, given a proposal with two parameters, these parameters may be set by taking the zeroth and first order constraints or by taking the first and second order constraints for example. In fact any combination of two constraints could be used and there is evidence to suggest that the flatness of the acceptance ratio is perhaps more important than it being closer to zero. We return to this point in greater detail in Section 3.4. In practice the choice of constraints may depend upon analytic tractability and/or computational complexity.

Before we return to our autoregressive example, we illustrate an alternative approach which extends the first order method described above by adopting an alternative to the zeroth order constraint for fixing the location parameter of the proposal.

The conditional maximisation approach

The conditional maximisation approach may be used to augment any of the methods discussed above, removing an additional degree of freedom by finding a sensible location for the proposal distribution. The idea behind all of these methods is to try and find proposal parameters that improve the chances of the chain actually jumping from one model to the next. When considering where to jump to in the higher dimensional space an obvious place to locate the proposal is around the mode of the posterior distribution for the higher dimensional model. The conditional maximisation approach proceeds by first taking the derivative of the log-conditional-posterior in the higher dimensional model and setting it to zero to locate the mode of the conditional distribution of the parameters that vary with $u$ under the jump jump $h_{i,j}(\theta_i, u)$ conditioning upon the value of those that don’t. For example, with a nested jump going from $\theta_i$ to $\theta_j = (\theta_i, \theta)$, we maximise the conditional distribution of $\theta$ given $\theta_i$ to obtain the conditional mode $\hat{\theta}_j = (\hat{\theta}_i, \hat{\theta})$. The proposal is then centered about the posterior mode $\hat{\theta}_j$ i.e., we set $c_{i,j}(\theta_i) = \hat{\theta}_j$. To derive values for any remaining proposal parameters (typically, the scale at least) we can use the zeroth or higher order methods described above, centering at the posterior mode.

Thus, the conditional maximisation method provides an alternative to the weak non-identifiability centering described above. There are, of course, other centering methods. For example, an alternative and promising approach in the context of linear models is described in Green (2000). In order to provide an illustration of the use of conditional maximisation and the other higher order methods in practice, we return to our autoregressive example.

3.3. Example: the auto-regressive example revisited

We begin with the basic first-order method described by the simultaneous solution of equations (7) and (14) to obtain both a location and scale for the proposal distribution so that $u \sim N(\mu_u, \sigma_u^2)$.  

Setting $A_{k,k+1} = 1$ and the derivative of $\log A_{k,k+1}$ to zero at the non-identifiability centering point $u = 0$, we obtain

$$\frac{\mu_u}{\sigma_u^2} = \frac{1}{\sigma_u^2} \sum_{t=k+k+1}^T \left( x_t - \sum_{t=1}^k a_t x_{t-k-1} \right) x_{t-k-1} \text{ and}$$

$$\frac{1}{\sigma_u} r_{k+1} (k) = \frac{1}{\sigma_u} \exp \left( -\frac{\mu_u^2}{2\sigma_u^2} \right) r_k (k+1)$$

These are clearly analytically intractable presenting the drawback that the equations require numerical solution and therefore incur additional computational expense. (In fact, in this case, the additional expense is fairly minimal - see Section 5.1. However, this may not generally be the case.)

An alternative is to consider a second order term and to set the first and second order derivatives of $A_{k,k+1}$ to zero at the non-identifiability centering point $u = 0$, ignoring the zeroth order term. Simultaneously solving these two equations, we obtain

$$\mu_u = \frac{\sum_{t=k+k+1}^T \left( x_t - \sum_{t=1}^k a_t x_{t-k-1} \right) x_{t-k-1}}{\sum_{t=k+k+1}^T x_{t-k-1}^2 + \frac{\sigma_u^2}{\sigma_u^2}}$$

and

$$\sigma_u^2 = \frac{\sum_{t=k+k+1}^T x_{t-k-1}^2 + \frac{\sigma_u^2}{\sigma_u^2}}{\sum_{t=k+k+1}^T \left( x_t - \sum_{t=1}^k a_t x_{t-k-1} \right) x_{t-k-1}}$$

Clearly, since the likelihood, proposal and prior are all Gaussian, the third order derivatives of $\log A_{k,k+1}$ and higher are all identically zero. Thus, these higher order constraints are automatically satisfied. Unfortunately, since the Gaussian distribution has only two parameters the zeroth, first and second order equations cannot normally be simultaneously satisfied so that an infinite order method is not possible.

Finally, if we wish to consider the conditional maximisation method, this involves taking $\mu_u$ to be point maximising the posterior distribution $L_{k+1}(\theta, u) p_{k+1}(\theta, u)$ obtained by setting the derivative of its logarithm to zero. We then centre at $u = \mu_u$ and, taking the zeroth order method, solve the equation $A_{k,k+1} = 1$ at $u = \mu_u$ to find $\sigma_u^2$. In this way, we obtain

$$\mu_u = \frac{\sum_{t=k+k+1}^T \left( x_t - \sum_{t=1}^k a_t x_{t-k-1} \right) x_{t-k-1}}{\sum_{t=k+k+1}^T x_{t-k-1}^2 + \frac{\sigma_u^2}{\sigma_u^2}}$$

and

$$\sigma_u^2 = \frac{\sum_{t=k+k+1}^T x_{t-k-1}^2 + \frac{\sigma_u^2}{\sigma_u^2}}{\sum_{t=k+k+1}^T \left( x_t - \sum_{t=1}^k a_t x_{t-k-1} \right) x_{t-k-1}}$$

Thus, we obtain the same mean as for the second order method and a variance term similar to the zeroth order solution.

All of these forms for the mean and variance of the proposal distribution have plausible interpretations. For the second order method, $\mu_u$ is a function of the estimated squared correlation coefficient, of order $k$, between the residuals from the fitted AR($k$) model. Recall that the squared correlation coefficient determines the MLE $a_{k+1}$ and so the proposal is approximately centered at the MLE. On the other hand, the variance $\sigma_u^2$ is the ratio between the model variance and the estimated variance. If the data are not particularly informative then the proposal variance increases.

For the conditional maximisation, the proposal mean is the same as before. For the variance we can see that the zeroth order solution is corrected by a factor that is positively related to the model variance and negatively related to the estimated covariance between the residuals from the fitted $k$th order model.
3.4. In support of first and higher order methods

It is difficult to prove rigorous results of any level of generality to support the use of the first and second order methods. However it is possible to make some progress, at least in simple stylised examples. Even for these toy examples, the results do not appear to be totally intuitive, and give support to the notion of attempting to construct algorithms which have little variation in acceptance probabilities (that is having first and higher order properties but not necessarily zeroth order properties). This idea is verified empirically in the autoregressive case by the results provided in Section 5.1.

The analysis will use a notion called capacitance (Lawler and Solal 1988), which is well-known to be closely related to the rate of convergence of a Markov chain by Cheeger’s inequality as we shall see. We define the capacitance of a reversible Markov chain:

$$\kappa = \inf_{A \subset B \text{ s.t. } \pi(A) \leq 1/2} \kappa(A)$$

where

$$\kappa(A) = \frac{\int_A \pi(dx) P(x, A^c)}{\pi(A)}. \quad (16)$$

Cheeger’s inequality tells us that if $\rho$ is the supremum of the spectrum of the Markov chain transition operator (that is usually its rate of convergence), then

$$1 - 2\kappa \leq \rho \leq 1 - \kappa^2/2.$$ 

See Brooks and Roberts (1999), for example.

This result tells us that a surrogate for convergence of a Markov chain is its capacitance, although we cannot precisely identify a chains rate of convergence from its capacitance. Therefore, it is natural to attempt to construct the algorithm with the largest possible capacitance (decreasing both the lower and upper bounds on the convergence rate) and in simple reversible jump settings we can identify these algorithms and characterise them in terms of first order properties of the Markov chain.

The following result is the simplest situation in which we can non-trivially discuss reversible jump, where transitions are being constructed between spaces of dimension 0 and 1.

**Proposition 3.1.** Suppose that $\Theta = \Theta_1 \cup \Theta_2$ with $\Theta_1 = \{e\}$ and $\Theta_2 = [0, 1]$. Let $\pi(e) = p$, and $\pi(u) = (1 - p)f(u)$ for some probability density function $f$ on $[0, 1]$. Consider the algorithm which always attempts to jump from model 1 to model 3 - i (i.e., without any within model moves). Let the proposal density be $q$ if we are currently in $M_i$, attempting a move to the 1-dimensional space $M_j$, and all moves from $M_2$ just attempt to jump to $e$. Then amongst all possible choices of $q$, the capacitance of the algorithm is maximised by the choice $q = f$.

**Remark.** In the language of this paper, the result can be restated as saying that the algorithm which maximises the rate of convergence is $r$th order for all $r \geq 1$, but not for $r = 0$. This is because the maximizing algorithm (with $q = f$) leads to an acceptance ratio of $A_{1,2} = (1 - p)/p$. Clearly, the derivatives of this are all zero, but the acceptance rate itself will only be zero if $p = 1/2$. Thus, the optimal algorithm is $r$th order for $r \geq 1$, but not zeroth order. A first and/or second order method may well be a good approximation to this optimal algorithm in specific cases.

**Proof** Consider an arbitrary set $A$ such that $e \in A$, and $\pi(A) = a$. Then, since $e \in A$, the only way to move to a point in $A^c$ is to move from model $M_1$ to model $M_2$. Therefore, the formula for $\kappa(A)$ in (16) reduces to

$$\kappa(A) = \frac{p}{a} P(e, A^c)$$

$$= \frac{p}{a} \times \int_{A^c} \min \left\{ 1, \frac{1 - p f(y)}{p q(y)} \right\} q(y) dy$$

$$= \frac{p}{a} \mathbb{E}_{f} \left( \min \left\{ \frac{q(Y)}{f(Y)}, \frac{1 - p}{p} \right\} 1_{A^c}(Y) \right)$$

$$= \frac{p (1 - a)}{a (1 - p)} \mathbb{E}_{f|A^c} \left( \min \left\{ \frac{q(Y)}{f(Y)}, \frac{1 - p}{p} \right\} \right), \quad (17)$$

Efficient RJMCMC Proposals
where \( f|A^c \) is just the density proportional to \( f \) restricted to \( A^c \) so that \( f|A^c(y) = (1-p) f(y)/(1-a) \) \( \forall y \in A^c \) and 0 otherwise. (Note that the normalisation constant here is derived from the fact that \( \pi(A) = p + (1-p) f(A) \), since \( e \in A \).)

Let \( \Gamma(u) = \{ y : q(y)/f(y) < u \} \), and \( \lambda(u) = P_f(\Gamma(U)) \). We shall assume that \( \lambda \) is continuous. A minor modification of this argument is possible to cover the discontinuous case. Let \( a^* \) be such that \( \lambda(a^*) = 1-a \). Set \( S^c = \Gamma(a^*) \). Then the expectation under \( f \) can be split into two distinct components by restricting the support first to \( S^c \subseteq \Theta_2 \) and \( S \cap \Theta_2 \) as follows,

\[
E_f \left( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \right) = (1-a) E_f|_{S^c} \left( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \right) + a E_f|_{S \cap \Theta_2} \left( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \right).
\]

Furthermore, \( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \) on \( S^c \) is less than or equal to its value on \( S \cap \Theta_2 \). Therefore

\[
E_f|_{S^c} \left( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \right) \leq E_f|_{S \cap \Theta_2} \left( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \right)
\]

and so

\[
E_f|_{S^c} \left( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \right) \leq E_f \left( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \right).
\]

Therefore

\[
\inf_{A: \pi(A) = a, e \in A} \kappa(A) \leq \frac{p(1-a)}{a(1-p)} E_f|_{S^c} \left( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \right) \text{ by (17)}
\]

\[
\leq \frac{p(1-a)}{a(1-p)} E_f \left( \min \left\{ \frac{1-p}{p} \frac{q(Y)}{f(Y)} \right\} \right) \text{ by (18)}
\]

\[
\leq \frac{p(1-a)}{a(1-p)} \min \left\{ \frac{1-p}{p}, E_f \left( \frac{q(Y)}{f(Y)} \right) \right\}
\]

\[
= \frac{p(1-a)}{a(1-p)} \min \left\{ \frac{1-p}{p}, 1 \right\}
\]

since \( E_f[q(Y)/f(Y)] = \int q/f dy = 1 \). However the last term in these series of inequalities is the value for the capacitance for any set \( A \) such that \( \pi(A) = a \) and \( e \in A \) in the case where \( q = f \). Hence for this case, the capacitance is maximised when \( q = f \).

The argument for the case where \( e \notin A \) is easy since by reversibility, \( \kappa(A) = \pi(A^c) \kappa(A^c)/\pi(A) \) so we can consider instead \( A^c \) which is covered by the case considered in detail above. The result therefore follows.

An alternative the \( k \)th order approaches described so far can be developed by adopting a dual space approach which allows the chain to retain information when going from a larger to a smaller model which can be used when returning to that model later on. This approach is discussed in the next section.

4. The dual space approach

The dual space approach involves augmenting the state space of the Markov so that the dimension of the chain remains constant throughout the simulation. At any given time, some of the states of the chain will correspond to model parameters and the rest can be used to retain information about where the chain has been in the past. In particular, if we move from a larger to a smaller dimensional model, information can be retained so that when we return to the larger model later on in the simulation, we can use this information to ensure that we propose jumping to a sensible place.

4.1. Augmenting the state space

Suppose that \( \sup_i \tau_i = \tau_{\text{max}} < \infty \) and let \((M_i, \theta_i)\) be a random variable distributed according to \( \pi \). Define a collection of dual random variables conditional upon the value of \((M_i, \theta_i)\) in the following
way. Given \( (M_i, \theta_i) \), let \( u_r \), \( n_i + 1 \leq r \leq n_{\text{max}} \) be a collection of univariate random variables with joint density \( q_{n_{\text{max}} - n_i} \) with respect to Lebesgue measure and let \( u_i = (u_{i,n_i+1}, \ldots, u_{i,n_{\text{max}}}) \). Thus, given \( M_i, (\theta_i, u_i) \) describes an \( n_{\text{max}} \)-dimensional random vector with joint density with respect to \( n_{\text{max}} \)-dimensional Lebesgue measure given by

\[
\pi_{\text{aux}} = \pi(M_i, \theta_i) \times q_{n_{\text{max}} - n_i}(u_i) .
\] (19)

Under this dual space arrangement, the Markov chain simulation proceeds in three stages at each update. First we update the states corresponding to the model parameters under the current model i.e., \( \theta_i \). This can be done in the usual way using some form of Metropolis-Hastings update for example. Next we update the elements of \( u_i \) by any procedure which preserves the stationary distribution of \( u_i \) conditionally upon \( \theta_i \). Finally, we update the model using a reversible jump step.

We shall now describe two distinct all-purpose reversible jump algorithms in the context of these dual random variables. We begin with the case where the random variables \( u_r \) are distributed independently from one another, before considering the dependent case.

4.2. The independent auxiliary variable method

Here, we suppose that given \( (M_i, \theta_i) \), the \( u_r \), \( n_i + 1 \leq r \leq n_{\text{max}} \) are a collection of independent univariate random variables each with density \( q \) with respect to Lebesgue measure. Thus, the joint density in (19) becomes

\[
\pi_{\text{aux}}(M_i, \theta_i, u_i) = \pi(M_i, \theta_i) \times \prod_{k=n_i+1}^{n_{\text{max}}} q(u_{i,k}) .
\]

Suppose that we have a collection of injective maps \( \{h_{i,j} : (i, j) \in E\} \) where \( E = \{(i, j) : r_i(j) > 0\} \) denotes the set of pairs \( (i, j) \) for which jumps between models \( M_i \) and \( M_j \) are allowed. For a particular pair \( (i, j) \) such that \( n_i < n_j \), the jump function \( h_{i,j} : \Theta_i \times \mathbb{R}^{n_{\text{max}} - n_i} \rightarrow \Theta_j \times \mathbb{R}^{n_{\text{max}} - n_j} \) is a bijective map that fixes \( u_r \), \( n_j + 1 \leq r \leq n_{\text{max}} \) i.e., if \( h_{i,j}(\theta_i, u_i) = (\theta_j, u_j) \) then \( u_{i,r} = u_{j,r} \), \( \forall n_j + 1 \leq r \leq n_{\text{max}} \). Finally, we set \( h_{j,i} = h_{i,j}^{-1} \).

Given that the current state of the Markov chain is \( (M_i, \theta_i, u_i) \), then the algorithm chooses an element at random from \( \{l : (i, l) \in E\} \), \( j \) say, and proposes the move to \( h_{i,j}(\theta_i, u_i) \) according to the probabilities \( r_i(l) \). The move is accepted then with probability \( \alpha[(M_i, \theta_i, u_i), (M_j, \theta_j, u_j)] = \min\{1, A_{i,j}(\theta_i, u_i; \theta_j, u_j)\} \) where

\[
A_{i,j}(\theta_i, u_i; \theta_j, u_j) = \frac{\pi(M_j, \theta_j) r_j(i)}{\pi(M_i, \theta_i) r_i(j) \prod_{k=n_i+1}^{n_{\text{max}}} q(u_{i,k})} |J_{i,j}(\theta_i, u_{i,n_i+1}, \ldots, u_{i,n_{\text{max}}})| .
\] (20)

Here \( \theta_j = h_{i,j}(\theta_i, u_i) \). Note the strong similarity with the acceptance ratio given in (1). Otherwise, for \( n_i > n_j \) the acceptance probability is given by

\[
\alpha((M_i, \theta_i, x_i), (M_j, \theta_j, x_j)) = \min\{1, A_{i,j}^{-1}(\theta_j, u_j; \theta_i, u_i)\} .
\]

Coupled with a positive-recurrence property of all algorithms updating within each \( \Theta_i \), the overall algorithm is suitably positive recurrent. Here the \( u \) are playing the role of the random draws from the proposal distribution used to do reversible jump MCMC. However, including them explicitly in the target density provides us with additional flexibility in constructing algorithms depending upon how we update the \( u \).

Example: "vanilla" reversible jump

Suppose that we update the elements of \( u \) by iid sampling directly from their stationary distribution. In this case, there is no need to store the \( u \)-sequence (since each is independent of the previous one) and in fact \( \theta \) is itself marginally, a Markov chain. Clearly, this is just ordinary reversible jump MCMC as described in Section 1.1 with proposal distribution \( q(u_{i,n_i+1}, \ldots, u_{i,n_j}) \) when jumping from model \( M_i \) to \( M_j \).

However, we can move beyond the vanilla reversible jump algorithm by adopting a Markov updating scheme for the elements of \( u \). This can be done in any manner of ways. However, we shall illustrate one approach in the context of the autoregressive example.
Example: autoregressive example revisited

We may update the elements of $\mathbf{u}$ in any manner which ensures that their stationary distribution is that specified for $q$. An alternative to the vanilla algorithm above is to use any Markov scheme which produces the correct stationary distribution. Suppose, for example, that we choose the standard Gaussian distribution to be our stationary distribution $q$. In that case we might use an autoregressive process to update the elements of $\mathbf{u}$ as follows.

Suppose that we are currently in model $M_k$ (where $k \leq k_{\text{max}}$) and that we need to update each of the elements $u_{k,k+1}, \ldots, u_{k,k_{\text{max}}}$. We may consider each in turn (since we are assuming that they are independently - and identically - distributed). Let us consider the update for $u_{k,r}$. If we take a new value for this variable ($u_{k,r}^t$, say) such that

$$u_{k,r}^t = \lambda u_{k,r} + N(0,1 - \lambda^2),$$

for some $\lambda \in [-1,1]$ then the stationary distribution of this process is the standard normal which we would adopt as our density $q$.

Thus, in the context of the autoregressive example and considering a jump from $(\theta_k, \mathbf{x}_k)$ to $(\theta_{k+1}, \mathbf{x}_{k+1})$, we might use a combination of the zeroth order method and the uncorrelated auxiliary variable method above, by setting $\theta_{k+1,k+1} = \sigma_u u_{k,k+1}$. Recall, that the zeroth order method suggests scaling the standard normal by $\sigma_u$, given in (12). Thus, the $q$ term in (11) is simply a normal density with zero mean and variance $\sigma_u^2$. Note also that when doing the reverse move we would set $u_{k,k+1} = \theta_{k+1,k+1}/\sigma_u$. Of course, this idea can easily be extended to the second order method by simply setting $\theta_{k+1,k+1} = \lambda u_{k,k+1}/\sigma_u$ for example.

To see how the auxiliary variable approach might improve the performance of the algorithm consider what happens as we move from a larger to a smaller model. Suppose we go from model $k+1$ to model $k$, then the old value of $a_{k+1}$ is stored as $u_{k,k+1}$. As the simulation continues, this value will continue to be updated. However, if $\lambda$ is close to 1, then movement will be very slow and if the reverse jump is proposed (from $k$ to $k+1$) relatively quickly, then the proposal will be to move to somewhere close (in terms of the value of $a_{k+1}$) to where it was when it left that model.

This provides the chain with a form of memory, making it easier to move between models. Of course, the length of this memory depends upon the value of $\lambda$. The larger the value, the longer the memory. Obviously, if we implement the $k$th order auxiliary variable method, but take $\lambda = 0$, we get just the $k$th order method. We shall examine the performance of these algorithms in Section 5.1 and also examine alternative implementations in the context of the mixtures problem in Section 7.

4.3. The correlated auxiliary variable algorithm

We turn now to an implementation of (19) which allows the elements of the $\mathbf{u}$ vector to be correlated. Though the method is more generally applicable, we shall restrict our attention to a Gaussian formulation for illustration. The basic idea is to assume that the $u_{i,r}$ random variables are exchangeable with distribution $N(0, \Sigma_i)$ where $\Sigma_i$ denotes the $i$ dimensional covariance matrix given by

$$\Sigma_i = \begin{pmatrix} 1 & \rho & \cdots & \rho \\ \rho & 1 & \cdots & \rho \\ \vdots & \vdots & \ddots & \vdots \\ \rho & \rho & \cdots & 1 \end{pmatrix}. $$

As before, the algorithm alternates between updating $u_i$ according to any Markov chain which preserves $q_{n_{\text{max}} - n_i}$, and proposing model jumps in the usual way. In this case (20) is modified slightly to give

$$A_{i,j}(\theta_i; \mathbf{u}_i, \theta_j; \mathbf{u}_j) = \frac{\pi(M_j, \theta_j) r_{ij}(i) q_{n_{\text{max}} - n_j} (u_{n_j+1}, \ldots, u_{n_{\text{max}}})}{\pi(M_i, \theta_i) r_{ij}(j) q_{n_{\text{max}} - n_i} (u_{n_i+1}, \ldots, u_{n_{\text{max}}})} |J_{i,j}(\theta_i; u_{i,n_i+1}, \ldots, u_{i,n_i})|. $$

The steps used to update $u_i$ can be carried out in a variety of ways. One natural scheme is to update the variables singly by Gibbs sampling. In this case,

$$u_{i,r} \mid u_{i,(r)} \sim N \left( \frac{\rho \sum_{s \neq r} u_{i,s}}{1 + (d-2) \rho}, \frac{(1-\rho)(1 + (d-1) \rho)}{1 + (d-2) \rho} \right).$$
where \( \mathbf{u}_{i,[r]} \) denotes the vector \( \mathbf{u}_i \) with the \( r \)th element removed and \( d \) is the dimension of the \( \mathbf{u}_i \) vector i.e., \( d = n_{\text{max}} - n_i \).

Both auxiliary variable methods are potentially of greatest advantage in the context of multimodality in model space. Specifically, there may be a number of models with appreciable probability mass separated by models of low probability through which the algorithm would need to travel in a journey between modes. The persistence introduced artificially into the \( U \)'s by the auxiliary variable methods allows certain choices of auxiliary variables to persist allowing the algorithm to sustain 'momentum' in moving between modes.

The correlated auxiliary variable method in particular allows the construction of algorithms where (for instance) values of the \( u \)'s which favour model complexity (or model parsimony), can be positively correlated, thus allowing the algorithm to favour complex models (or parsimonious ones) at particular times. (See Section 7.3 for an illustration of this effect in the context of the mixture example.) This provides a potential solution to the multimodality problem if there are a number of modal regions in model space all of similar complexity, and mixing of the Markov chain can be improved by briefly moving to a very complex (or simple) model before returning to the region of most highly supported model complexity, possibly reaching a different mode in the process.

4.4. The infinite auxiliary variable method
Since \( n_{\text{max}} \) in Section 4.3 can be replaced by any larger integer, a natural possibility is to consider the case where we have an infinite collection of auxiliary variables. This does allow us to consider the case where the dimensionality of the models under consideration is unbounded. By de Finetti’s theorem, in the exchangeable case, this can be reformulated hierarchically as follows.

Suppose that \( Y \) is a \( N(0, \rho) \) variable, and conditional upon \( Y \), we set each \( u_{ij} \) to be independently \( N(Y, 1 - \rho) \). This formulation is particularly attractive for the correlated auxiliary variable method, since we can assume that we have this infinite collection \( \mathbf{u} \), though only store as many as we need. If we need a new one, we just generate it from its distribution conditional upon \( Y \). In this case the acceptance ratio for a move from smaller dimension \( n_i \) to larger dimensional \( n_j \) is described by

\[
A_{i,j}(\theta, \mathbf{u}_i; \theta, \mathbf{u}_j) = \frac{\pi(M_j, \theta_j) r_j(i) \tilde{q}_d(n_{i+1}, \ldots, n_{i+j})}{\pi(M_i, \theta_i) r_i(j) \tilde{q}_d(n_{i-1}, \ldots, n_{i+j})} |J_{i,j}(\theta, \mathbf{u}_i; \theta, \mathbf{u}_j)| ,
\]

where \( \tilde{q}_d \) denotes the density of \( d \) independent random variables with distribution \( N(Y, 1 - \rho) \).

The algorithm therefore proceeds as follows. Suppose we are currently at model \( M_i \), and in state \((\theta_i, \mathbf{u}_i)\).

- Update \( Y \) according to any Markov chain dynamic preserving its distribution.
- Choose a model to try to move to according to \( r_i(\cdot) \), \( j \) say.
- Suppose that \( n_j > n_i \), then
  - generate \( u_{n_i+1}, \ldots, u_{n_j} \) according to \( N(Y, 1 - \rho) \);
  - accept the move to \( h_{i,j}(\theta, \mathbf{u}_i) \) with probability \( \min\{1, A_{i,j}(\theta_i, \mathbf{u}_i; \theta_j, \mathbf{u}_j)\} \) given in (21);
  - otherwise remain at \((M_i, \theta_i, \mathbf{u}_i)\).
- If \( n_j < n_i \), compute \( h_{i,j}^{-1}(\theta_i, \mathbf{u}_i) \), which gives us \((\theta_j, \mathbf{u}_j)\). Accept this move with probability \( \min\{1, A_{i,j}(\theta_j, \mathbf{u}_j; \theta_i, \mathbf{u}_i)\}^{-1} \). Otherwise stay at \((M_i, \theta_i, \mathbf{u}_i)\).

5. Comparing the methods
In this and the following sections, we shall illustrate our methodology with the aid of three examples chosen to represent the full range of problems to which RJMCMC is regularly applied. However, before we do so, we begin by discussing a variety of methods which can be used to assess sampler performance so that the different algorithms may be compared.

The performance assessment techniques can be split into two categories, the numerical and the graphical. Graphical techniques include raw trace plots, autocorrelation plots and cumulative plots.
Since these are to be plotted over all iterations, we must first find statistics to plot which retain a constant interpretation across all models. In most cases, the model number may be plotted. For example in the AR case the model number is simply the AR order; in the mixtures case it would be the number of components; and in the graphical models case it may the number of edges or some lexicographic representation of the current graph. An alternative is to plot the deviance over time as that too retains a constant interpretation. Examples of cumulative plots include the cumulative number of models visited within a simulation, and the cumulative occupancy fractions (see Richardson and Green 1997, Brooks and Giudici 2000 and Brooks et al 2000).

Numerical assessment techniques include monitoring acceptance rates for model-changing moves, noting the total number of models visited, effective sample size calculations and convergence rate estimates. Acceptance rates for RJMCMC moves are typically somewhat lower than those for Metropolis-Hastings, for example. Though, a high acceptance rate doesn't necessarily guarantee good sampler performance (Gelman et al 1996) an increase in the acceptance rate whilst retaining the same posterior inference would usually be viewed as an improvement. Effective sample size (ESS) calculations can be obtained for statistics that retain a coherent interpretation throughout the simulation. These tell us how many independent observations are equivalent (in terms of learning about specific statistics of interest) to the set of dependent observations actually obtained. Thus, the larger the ESS, the better the performance of the algorithm, see for example Hastings (1970). Perhaps most useful are comparisons of ESS per second (Brooks and Draper 2000) which also incorporates computational expense for a more practical comparison. Finally, convergence rate estimation (in terms of the marginal distribution over model space) may be obtained by examining the marginal distribution of the model number (or any other scalar statistic with constant interpretation, such as number of edges) and deriving an empirical transition matrix for this sub-Markov chain. The second largest eigenvalue of this matrix provides an indication of the convergence rate, see Brooks et al (2000).

None of the methods described above provide a reliable comparison in themselves, but together they provide sufficient information to begin making some general statements comparing two or more samplers. Obviously there is considerable scope for future work in this area.

5.1. Example: autoregressive model choice

Throughout the preceding sections, we have illustrated our methods with reference to the analysis of autoregressive time series. We begin our discussion, comparing the different methods, by examining their performance in the context of this example.

We simulate a series of 500 observations from a stationary autoregressive process of order 2. For each method we take a vague prior over parameter and model spaces and run five independent replications of 1 million iterations, thinning to every 20th value to reduce computational overheads associated with storage. In all simulations, the within-model parameter estimates were extremely close to the true values. In the case of autoregressive models of order greater than two, the first two parameters were estimated to have values close to the true values under the AR(2) model and the remainder were all very close to zero, as would be expected. Similarly, the answers in terms of posterior model probabilities were also very reliable, though there was a greater degree of variability between the different methods. All methods attributed highest posterior probability to the AR(2) model with steadily decreasing probability assigned to higher order models, as we would expect.

For illustration, we ran the vanilla algorithm with a pilot-tuned proposal distribution for model moves fixed to be a normal distribution with mean zero and variance 0.001. For comparison, we also ran the zeroth, first, second and conditional maximisation methods. We also ran a series of auxiliary variable methods. We begin with the vanilla uncorrelated auxiliary variable method, taking a range of \( \lambda \) values. We then fixed \( \lambda \) to be 0.5 and took a combination of the uncorrelated AV method with the zeroth, first, second and conditional maximisation methods. Finally, we ran the correlated AV method with a range of \( \rho \) values. The results of these simulations are presented in Table 1.

We can see from Table 1 that all of the simulations put highest posterior probability on the true model and that all except the methods involving zeroth order visit the full range of models. The zeroth order methods appear to have performed poorly here. They observe only models 1-4 and spend too much time in the AR(2) model. This is because, we have taken a vague prior for the parameters and so the zeroth order method is tuning itself to a space which differs markedly from the posterior. The lack of movement for the zeroth order method throughout the simulation leads
Table 1. Summary statistics (acceptance rate \( \hat{\alpha} \), range of models visited, posterior probability of “true” model, effective sample size, computation time in seconds and estimated convergence rate, \( \hat{\rho} \)) for the performance (averaged over 5 replications) of the different methods for the thinned chains and the autoregressive example. * denotes that vanilla uncorrelated AV (UAV) and correlated AV (CAV) methods were used.

<table>
<thead>
<tr>
<th>Method</th>
<th>( \hat{\alpha} )</th>
<th>models</th>
<th>( \pi(M_c) )</th>
<th>ESS</th>
<th>Time(s)</th>
<th>( \hat{\rho} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vanilla</td>
<td>0.030</td>
<td>1-8</td>
<td>0.792</td>
<td>1070</td>
<td>139</td>
<td>0.975</td>
</tr>
<tr>
<td>Zeroth</td>
<td>0.006</td>
<td>1-4</td>
<td>0.974</td>
<td>5600</td>
<td>137</td>
<td>0.408</td>
</tr>
<tr>
<td>First</td>
<td>0.030</td>
<td>1-8</td>
<td>0.742</td>
<td>1094</td>
<td>252</td>
<td>0.979</td>
</tr>
<tr>
<td>Second</td>
<td>0.035</td>
<td>1-8</td>
<td>0.732</td>
<td>1136</td>
<td>214</td>
<td>0.979</td>
</tr>
<tr>
<td>CM</td>
<td>0.018</td>
<td>1-8</td>
<td>0.893</td>
<td>2188</td>
<td>217</td>
<td>0.974</td>
</tr>
<tr>
<td>UAV*</td>
<td>( \lambda = 0.2 )</td>
<td>0.030</td>
<td>1-8</td>
<td>0.740</td>
<td>818</td>
<td>141</td>
</tr>
<tr>
<td></td>
<td>( \lambda = 0.5 )</td>
<td>0.027</td>
<td>1-8</td>
<td>0.752</td>
<td>1186</td>
<td>140</td>
</tr>
<tr>
<td></td>
<td>( \lambda = 0.7 )</td>
<td>0.025</td>
<td>1-8</td>
<td>0.752</td>
<td>1158</td>
<td>141</td>
</tr>
<tr>
<td></td>
<td>( \lambda = 0.9 )</td>
<td>0.022</td>
<td>1-8</td>
<td>0.749</td>
<td>822</td>
<td>141</td>
</tr>
<tr>
<td></td>
<td>( \lambda = 0.95 )</td>
<td>0.021</td>
<td>1-8</td>
<td>0.747</td>
<td>782</td>
<td>141</td>
</tr>
<tr>
<td>UAV (0)</td>
<td>( \lambda = 0.5 )</td>
<td>0.006</td>
<td>1-5</td>
<td>0.969</td>
<td>4418</td>
<td>134</td>
</tr>
<tr>
<td>UAV (1)</td>
<td>( \lambda = 0.5 )</td>
<td>0.028</td>
<td>1-8</td>
<td>0.730</td>
<td>1280</td>
<td>268</td>
</tr>
<tr>
<td>UAV (2)</td>
<td>( \lambda = 0.5 )</td>
<td>0.031</td>
<td>1-8</td>
<td>0.735</td>
<td>1410</td>
<td>178</td>
</tr>
<tr>
<td>UAV (CM)</td>
<td>( \lambda = 0.5 )</td>
<td>0.018</td>
<td>1-8</td>
<td>0.873</td>
<td>1020</td>
<td>177</td>
</tr>
<tr>
<td></td>
<td>( \lambda = 0.2 )</td>
<td>0.030</td>
<td>1-8</td>
<td>0.767</td>
<td>1530</td>
<td>326</td>
</tr>
<tr>
<td></td>
<td>( \lambda = 0.5 )</td>
<td>0.028</td>
<td>1-8</td>
<td>0.743</td>
<td>1348</td>
<td>328</td>
</tr>
<tr>
<td>CAV*</td>
<td>( \lambda = 0.7 )</td>
<td>0.025</td>
<td>1-8</td>
<td>0.742</td>
<td>1082</td>
<td>327</td>
</tr>
<tr>
<td></td>
<td>( \lambda = 0.9 )</td>
<td>0.020</td>
<td>1-8</td>
<td>0.660</td>
<td>864</td>
<td>327</td>
</tr>
<tr>
<td></td>
<td>( \lambda = 0.95 )</td>
<td>0.018</td>
<td>1-8</td>
<td>0.569</td>
<td>1022</td>
<td>328</td>
</tr>
</tbody>
</table>

To inaccurate estimates of the convergence rate and effective sample size, so these should be ignored here.

In terms of acceptance rate, the second order method appears to perform well, the effective sample size for this method appears to compare well with the vanilla, though with additional computational expense. Of course the second order method adaptively selects the proposal parameters avoiding the hidden computational expense associated with the pilot tuning required for the vanilla method. The conditional maximisation method also appears to perform well in terms of ESS and estimated convergence rate. The acceptance rate is rather low but this should not, in itself, be taken as a negative indication of performance, as we shall see in the context of the mixtures example in Section 7.

Turning to the uncorrelated auxiliary variable method, we can see that increasing \( \lambda \) for the vanilla method results in a decreasing acceptance rate as we would expect, and that the convergence rate appears to change very little. The effective sample size seems to increase for \( \lambda \) values of 0.2 and 0.5, suggesting that some but not much dependence is most desirable for this example. It is worth noting here, that this is a particularly simple example. There is no multi-modality in model space and so the addition of the auxiliary variables would not be expected to improve upon the vanilla algorithm which itself performs very well on this example.

When we fix the dependence of the uncorrelated AV method and consider combining it with the \( k \)th order methods, we observe a similar pattern to those without AV. The second order and conditional maximisation methods appear to perform well and arguably slightly better than the pilot-tuned vanilla algorithm. Finally, when we examine the correlated AV method, we observe a similar performance to the uncorrelated AV method, much as we might have expected.

These results demonstrate that higher order methods appear to work at least as well as the pilot-tuned vanilla algorithm in this simple example. There appears to be no additional benefit to the introduction of auxiliary variable methods, but that would be expected in this most simple of cases. In the next two sections, we consider a correspondingly smaller ranges of methods applied to two more realistic problems and demonstrate how the \( k \)th order and auxiliary variable methods can make dramatic improvements in performance over the the vanilla algorithm (even when it is pilot-tuned).
6. Graphical Gaussian models

Let $\mathbf{X}$ be a $k$-dimensional vector of random variables. A conditional independence graph, $\mathcal{G} = (V, E)$, describes the association structure of $\mathbf{X}$ by means of a graph, specified by the vertex set $V$ and the edge set $E$. A graphical model is a family of probability distributions $P_\mathcal{G}$ which is Markov over $\mathcal{G}$ (see, for instance, Lauritzen 1996). A graphical Gaussian model is obtained when only continuous random variables are considered and assuming $P_\mathcal{G} = N(\mathbf{\mu}, \Sigma_\mathcal{G})$, with $\Sigma_\mathcal{G}$ positive definite and such that $P_\mathcal{G}$ is Markov over $\mathcal{G}$.

An exact Bayesian analysis of decomposable graphical Gaussian models, using hyper inverse Wishart priors, has been suggested in Dawid and Lauritzen (1993). A severe limitation of the exact approach is the necessity of considering simple one-stage conjugate priors, which are difficult to specify and, furthermore, may have a strong effect upon final inferences. Furthermore, the number of alternative models to be considered may be too large to permit full analytic computations for all of them. Recently, Giudici and Green (1999) have proposed a hierarchical class of prior distributions, and a reversible jump MCMC method to approximate both model selection and inference upon the quantities of interest.

Here we consider their proposed hierarchical prior. The dimension-changing moves are made within the class of all decomposable graphical models. Moves are performed, at each stage, by adding or deleting one edge from the conditional independence graph of the model and checking that the resulting graph is decomposable, so as to allow local computations.

When a new edge, say $(i, j)$ is proposed for insertion, the dimensionality of the parameter space increases by one; this is specified by an extra free element of $\Sigma_g$. This is done by drawing a random variable $u$ from a $N(0, \sigma_{ij}^2)$ distribution and then letting $\sigma_{ij} = u$. This is a blind proposal, which does not take into account the previous (constrained) state of $\sigma_{ij}$ as this ensures that updates involve only local computations.

Giudici and Green (1999) show that the ratio of the model posteriors localises to the four subsets $S$, $S \cup i$, $S \cup j$, $S \cup i \cup j$ (abbreviated below as $S, Si, Sj$ and $St$):

$$R_{post} = \frac{h(\Sigma_S)h(\Sigma_{Si})}{h(\Sigma_{Si})h(\Sigma_{Sj})}$$

where each of the four terms above is obtained as the product of the prior and the likelihood of the appropriate submatrix of $\Sigma$. For instance, for $S$:

$$h(\Sigma_S) = I(W(\Sigma_S; \alpha, \Phi_S) \propto N(x_S; \Sigma_S).$$

Putting together the different terms, as described in Green (1995) the acceptance ratio is equal to:

$$A_{k,k+1} = R_{post} \times q(u)^{-1},$$

where $q(u)$ is the proposal distribution for the extra-free element of $\Sigma_g$.

One difficulty with this approach, is the choice of the spread parameter $\sigma_g^2$ of the proposal distribution. In Giudici and Green (1999) the constant is fixed, on the basis of several pilot runs, to be equal to $0.5 \times n/|V|$, where $n$ is the sample size and $|V|$ the cardinality of the vertex set $V$. Our aim is now to construct efficient reversible jump rules for the varying-dimension move, according to the recipes specified in the previous sections.

First of all, we note that the dimension-matching constraint is specified by a 1-1 function between $\Theta = \{\sigma_{ik} : (i, k) \in E_g\}$ and $\Psi = \{(\sigma_{ik} : (i, k) \in E_g) \cap (\sigma_{ij} : (i, j) \notin E_g \cap (i, j) \in E_{g'})\}$. Therefore, the zeroth order proposal lead us to set the proposal parameter to be given by

$$\sigma_g^{-2} = \sqrt{2 \pi} \times \frac{h(\Sigma_S)h(\Sigma_{Si}, u = 0)}{h(\Sigma_S)h(\Sigma_{Sj})},$$

using the weak non-identifiability centering point for which $\sigma_{ij} = 0$.

The graphical Gaussian model example differs from many others in that the hyper Markov prior distributions lead to a posterior distribution which is structurally very similar. This suggests that the zeroth order method may well provide a more obvious improvement than in the simpler variable
selection context illustrated by the AR example. In addition, the very nature of the model space suggests that, for graphical models, an auxiliary variable scheme, such as those illustrated in Section 4 may improve convergence, especially when highly dimensional datasets are being considered.

The total number of auxiliary variables here is equal to the maximum number of edges possible i.e., the number of edges in the complete graph. Let \( n_{\text{max}} = n(n - 1)/2 \) be this number. In order to implement the dual space approach, we assume that we have a vector of auxiliary variables, \( u_1, \ldots, u_{n_{\text{max}}} \), that we assume distributed as a multivariate Gaussian, with zero mean and variance covariance matrix equal to an intra-class correlation structure

\[
\Phi = \tau (\rho J + (1 - \rho) I),
\]

where \( J \) is the \( p \times p \) matrix of 1’s and \( I \) the identity matrix of order \( p \). We take \( \tau = 1 \).

In the MCMC implementation we sample each \( u_i \) from a proposal distribution corresponding to the full conditionals derived from the previous stationary distribution of the \( u \)'s. It is easy to derive the fact that the full conditionals are Gaussian with mean equal to \( \frac{\rho \sum_{j \neq i} u_j}{1 + |n_{\text{max}} - 2|\rho} \) and variance equal to \( \frac{(1 - \rho)(1 + |n_{\text{max}} - 1|\rho)}{1 + |n_{\text{max}} - 2|\rho} \).

6.1. Results

We first compare the mixing performance of our proposed zeroth algorithm with the vanilla (pilot tuned) reversible jump scheme, as developed in Ghidici and Green (1999). The comparisons will be made using both graphical and more formal model convergence diagnostics. We remark that, for space purposes, all graphical output has been thinned, retaining only one in every 10 observations.

First we briefly consider, for illustrative purposes, one of the simplest, and most analysed, graphical modelling datasets: Fret’s data, described in Whittaker (1990), concerning head measurements on pairs of sons in a sample of 25 families. In this example, since \( k = 4 \), the number of possible graphs is equal to 64, including 3 which are not decomposable.

We run two simulations each of length \( n = 100,000 \), starting from the same point, for the vanilla and the zeroth order methods. In reporting the results from MCMC model selection, we represent a graph by means of a vector of binary variables, indicating whether each edge is present (1) or absent (0), and with edges in a graph being ordered lexicographically. Frets’ data contains at most 6 edges. The two graphs with the highest posterior probability are, with the vanilla method: (110111), with probability 0.13497, and (111011), with probability 0.12804. In other words, the two best graphs differ by the position of the chord that breaks the four-cycle. The zeroth order method gives the same two best graphs: (110111), (0.12704) and (111011), (0.11728). However, the posterior probabilities are slightly lower in the zeroth order case, suggesting that the Markov chain has spent more time in the tails of the distribution. In fact, upon closer inspection, the vanilla algorithm visits only 23 distinct models compared to the zeroth which sees a total of 29. This provides clear evidence of the superiority of the zeroth order method on this example. We next consider a larger example in which this improvement is even more dramatic.

We now consider the analysis of the Fowl bones data-set (Whittaker 1990) concerning measurements on chicken bones. As there are 6 vertices, the number of possible graphs is equal to 32768. The total number of decomposable graphs is equal to about 80% of these and so the resulting RJMCMC simulation runs on a graph space with about 26,300 candidate models. Thus, the model space is considerably larger than that for the Fret’s data. We shall investigate the zeroth order method together with two possible correlated auxiliary variable methods in comparison to the usual (pilot tuned) vanilla method.

An interesting problem here is that for one of the 15 edges, the weak non-identifiability centering turns out to be inappropriate, precisely because the presence of the edge in question within the model is very strongly supported by the data. Thus, improvements on all methods are possible by problem specific centering strategies. Using the zeroth order method for instance, the fact that the centering point is very much in the tail of the proposed model space leads to an extremely large variance for the proposed move. This is inappropriate in this example since what is being proposed is the introduction of a (non null) partial correlation. Therefore, rather than refining the centering point (which would
be a very problem specific fix), we have imposed a truncation of the proposal variance to preclude the proposal of a large majority of impossible values for the partial correlation.

The first part of Table 2 demonstrates the substantial improvement in performance of the zeroth order method over the vanilla with a simulation run length of 1,000,000 thinned to every 10th. The correlated auxiliary variable method was implemented with \( \rho = 0.5 \) and, as we can see from Table 2, performed at least as well as the zeroth order method.

Figure 1 provides a trace plot of the number of edges, which can be taken as a measure of model complexity. The difference in performance between the vanilla and our methods is illustrated by the more rapid transitions between edge counts.

**Figure 1 about here**

Figure 2 compares more closely the behaviour of the posterior probabilities of the models. The plots give the cumulative number of different models visited by the Markov chain for the three algorithms. They demonstrate that the vanilla method has clearly failed to converge since it has visited not much more than half the number of models visited by the zeroth order method or the correlated auxiliary variable method. The correlated auxiliary variable method visits many more models than the zeroth order method also.

**Figure 2 about here**

Looking more closely at the model posterior distributions, we found that all methods find the two highest probability models in the correct order (and appear to visit broadly the same class of popular models if we examine the ordered list of popular models in more detail). However as the second part of Table 2 shows, the auxiliary variable method gives rather lower probabilities to these two popular models (labelled 30504 and 29992 in our notation). This is consistent with the idea that the auxiliary variable method has explored the tails of the space of models more thoroughly, and is also consistent with the plots in Figure 1 where the auxiliary variable method succeeds in visiting 14 edges considerably more often than the other two methods.

Investigation using multiple long runs (not shown here) involving 10 runs of 1,000,000 iterations suggests reasonable stability in estimates from the zeroth and correlated auxiliary variable method, tentatively suggesting that 1,000,000 iterations is sufficient for convergence for both the zeroth and auxiliary methods, though not perhaps for accurate estimation of posterior model probabilities.

### 7. Mixture models

As our final example, we re-examine the classification problem discussed by Richardson and Green (1997) and look at modelling a series of univariate data as a finite mixture of Gaussian distributions. This example presents peculiar difficulties. For instance, the model changing moves involve moving between spaces differing by more than one dimension. Thus, we typically have more degrees of freedom in our proposal distribution for moving from one model to the next. Furthermore, the parameter spaces may be bounded both above and below as a result of the ordering constraints placed upon the component parameters. As a consequence, we are somewhat limited by analytic tractability in terms of the use of the \( k \)th order methods. However, the auxiliary variable methods remain very easy to use.

<table>
<thead>
<tr>
<th></th>
<th>Vanilla</th>
<th>Zeroth</th>
<th>CAV</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \delta )</td>
<td>0.001</td>
<td>0.015</td>
<td>0.026</td>
</tr>
<tr>
<td>ESS</td>
<td>91</td>
<td>874</td>
<td>1403</td>
</tr>
<tr>
<td>Model 30504 probability</td>
<td>0.17</td>
<td>0.18</td>
<td>0.16</td>
</tr>
<tr>
<td>Model 29992 probability</td>
<td>0.10</td>
<td>0.11</td>
<td>0.06</td>
</tr>
</tbody>
</table>

**Table 2.** Fowl bones data: summary statistics (acceptance rate \( \delta \), and effective sample size) and model probability estimates for the two models with greatest posterior mass, for thinned simulation output using different proposal determination methods.
A $k$-component mixture model has density of the form

$$f(x) = \sum_{j=1}^{k} w_j \pi_j(x),$$

where the $\pi_j$, $j = 1, \ldots, k$ are densities which are known at least up to some multiplicative constant, and the proportions $w_j$, $j = 1, \ldots, k$ are such that $\sum_{j=1}^{k} w_j = 1$. We shall concentrate upon mixtures of normal distributions here and therefore we increase the dimension of Markov chain by three for every new component that we add. In practice, models of this sort are easily fitted by introducing a set of auxiliary allocation variables which match each observation to exactly one of the $k$ components. Thus, we may also need to re-assign these variables when we move from one dimension to another. Fortunately, there are already some fairly natural proposals for this aspect of the jump from one model to the next. For example if we wish to replace an existing component by two new ones we can simply re-assign the observations allocated the single component by taking a binomial draw with probability of success proportional to the weight of one of the new components. Thus, we shall concentrate upon the aspects of the move i.e., how to choose a new weight, mean and variance parameters for any new components.

Richardson and Green (1997) propose the use of a RJMCMC sampler which comprises a series of Metropolis-Hastings updates for within model moves and two types of move for jumping between models. The first is a split/combine move in which a single component is split into two or two adjacent components are combined. The second move type is the birth/death move which permits the addition or deletion of an empty component.

There are various ways in which each of these moves may be implemented. However, for illustration, we shall adopt the implementation of Richardson and Green (1997). For the move which splits a single component ($j$) into two ($j_1$ and $j_2$), we adopt a moment-matching strategy so that the weight assigned to that component is split between the two new components. Similarly, we assign means and variances to the new components which preserve the first and second order moments. This can be done by generating three random variables $u_1$, $u_2$ and $u_3$ from any density defined upon $[0,1]$ and setting

$$w_{j_1} = w_j \cdot u_1, \quad w_{j_2} = w_j \cdot (1 - u_1)$$

$$\mu_{j_1} = \mu_j - u_2 \sigma_j \sqrt{\frac{w_{j_1}}{w_j}}$$

$$\mu_{j_2} = \mu_j + u_2 \sigma_j \sqrt{\frac{w_{j_2}}{w_j}}$$

$$\sigma_{j_1}^2 = u_3 (1 - u_2^2) \frac{w_{j_1}}{w_j}$$

$$\sigma_{j_2}^2 = (1 - u_3)(1 - u_2^2) \frac{w_{j_2}}{w_j}$$

Richardson and Green (1997) suggest generating the $u_i$ from independent beta distributions and take

$$u_1 \sim \beta(2,2), \quad u_2 \sim \beta(2,2), \quad u_3 \sim \beta(1,1)$$

7.1. The $k$th order methods

Let us begin by identifying the centering point for the move. Clearly, the weak identifiability centering is obtained when the means and variances of the two new components are identical (and therefore the same as for the original component). The weights can take any value. Thus, the a centering point is given by $(u_1, u_2, u_3) = (0.5, 0, 0.5)$, say. The interesting thing to note here is that the proposals taken by Richardson and Green (1997) for $u_1$ and $u_3$ have modes at the corresponding center points, but this is not so for $u_2$. If we transform the proposal for $u_2$ so that $u_2 = [2v_2 - 1]$, with $v_2 \sim \text{Be}(2,2)$, so that the mode is now at zero, then the acceptance rate for the Split/Combine moves increases from
8% (for the enzyme data set) to 10%, so we immediately appear to observe a small improvement just by thinking more carefully about the proposals. (Note that the acceptance ratio A as defined in Richardson and Green (1997) is undefined if u₂ = 0. However, trivial manipulations allow us to rewrite A with no single u₂ term on the bottom and hence the above results were obtained.)

The obvious kth order approach is to try generating \( u_i \sim \beta(\alpha_i, \beta_i) \) and use perhaps the zeroth to fifth order equations to determine sensible values for these six parameters. A simpler alternative is to set \( \alpha_i = \beta_i = c \) and use just a single constraint to obtain c. Before we begin, we first rewrite \( A \) in terms of a function of the parameters and data which we will denote by \( f(\cdot) \). That is, \( f \) denotes all the terms in the acceptance ratio given by Richardson and Green (1997) except those that include the proposal distributions for the \( u_e \). If we write our proposal for the \( u_i \) in terms of a single parameter \( R \), then we have

\[
A = f(\cdot)/g(u|R).
\]

Thus, in order to implement our method, we simply set \( A = 1 \) at the centering point and invert \( g \) to find the value of \( R \). The problem with this occurs when the inversion of \( g \) is not analytically tractable, so we need to concentrate on proposals which are simple enough so that does not occur.

The simplest example is to sample \( u_1, u_3 \sim U(0.5 - R, 0.5 + R) \) and \( u_2 \sim U(0, 2R) \). In this case, \( g(u|\tau) = (2R)^{-3} \) on an appropriate hyper-cube, and we have \( R = f(\cdot)^{1/3}/8 \). Using the zeroth order method to determine \( R \), we observe that when we propose a split move, \( R \) tends to be small (so we don’t move far), but when we propose a combine move, \( R \) is nearer 0.5, so we had a reasonable chance of proposing the point that we are actually at. Thus, the algorithm appears to be choosing sensible \( R \) values.

Using the zeroth order method to determine the value of \( R \) for the enzyme data set of Richardson and Green (1997), the acceptance rate decreases the split/combine acceptance rate to 2.5% from the 8% acceptance rate observed by the original authors. If we fix \( R \) at 0.5 throughout, we have a rate of 7.6%, which is more comparable. We remark that, although we are not yet improving the results of Richardson and Green (1997), this result has been obtained without any use of pilot-tuning.

Similar approaches may be used for the birth/death move. When we propose a birth, we introduce an empty component which therefore has no likelihood contribution whatever the values of the new component parameters. However, structurally, the model remains unchanged if \( w = 0 \), so we might take \((0, 0.5, 0.5)\) as our centering point. However, the acceptance ratio is undefined at this point and so the implementation of kth order methods would require an alternative centering point here. Conditional maximisation is a possibility, but the maximisation will need to be done numerically and may slow the simulation down considerably, particularly if we return to the six-parameter optimisation problem.

Thus, the kth order methods do not appear to be particularly helpful in the context of normal mixtures. This is mainly due to the severe restrictions that have to be placed upon the proposal distributions in order that the kth order methods become tractable. However, the auxiliary variable methods perform considerably better, demonstrating a significant improvement over the vanilla method as we shall see in the next section.

### 7.2. The auxiliary variable methods

With the auxiliary methods, we need to update \( 3k_{max} \) parameters of which \( 3k \) are associated with the current \( k \)-component model and \( 3(k_{max} - k) \) are the auxiliary variables. Thus,

\[
u_k = (u_{1,k+1}, u_{2,k+1}, u_{3,k+1}, \ldots, u_{1,k_{max}}, u_{2,k_{max}}, u_{3,k_{max}}) .
\]

When we consider the correlated method, we shall consider only correlation between \( u_{i,j} \), \( j = k + 1, \ldots, k_{max} \) and not consider correlation within \( i = 1, 2, 3 \), since we wish to retain independent proposals for the three separate parameters. However, the methods we describe could be extended to the case where dependent proposals were desirable.

It is easiest to begin with the correlated AV method for which we require an updating scheme which induces dependence between successive iterations and across (future) components. We also require that the elements of \( u \) lie in \([0,1]\). These requirements can be satisfied as follows.
Consider a point process \( \{ C^*_t \} \) which follows a random walk on \([0, 1]\) so that
\[
C^*_t = \begin{cases} 
C^{t-1}_t + v^*_t & C^{t-1}_t + v^*_t > 1 \\
C^{t-1}_t + v^*_t & 0 < C^{t-1}_t + v^*_t \leq 1 \\
C^{t-1}_t + v^*_t & C^{t-1}_t + v^*_t \leq 0
\end{cases}
\]
where \( v^*_t \sim U[-\epsilon, \epsilon], \epsilon \leq 0.5 \). This essentially describes a random walk on \([0, 1]\) in which the endpoints are joined. You might think of \( C \) as living on a ring of circumference 1, where the value of \( C \) is determined by the distance along the ring (in a clockwise direction) from a fixed point on the circumference. The stationary distribution for \( C^*_t \) is therefore uniform on \([0, 1]\). We call \( C^*_t \) the "central" point. Given the central point \( C^*_t \), we can generate the \( u^i_{k,j} \) by setting
\[
u^i_{k,j} = \begin{cases} 
C^*_t + w^i_{k,j} - 1 & C^*_t + w^i_{k,j} > 1 \\
C^*_t + w^i_{k,j} & 0 < C^*_t + w^i_{k,j} \leq 1 \\
C^*_t + w^i_{k,j} + 1 & C^*_t + w^i_{k,j} \leq 0
\end{cases}
\]
where \( w^i_{k,j} \sim U[-\delta, \delta], \delta \leq 0.5 \).

The fact that all of the \( u^i_{k,j} \), \( j = k + 1, \ldots, k_{\text{max}} \) are generated from the same distribution induces a dependence between them which increases as \( \delta \rightarrow 0 \). Similarly, the central point moves around the ring inducing a dependence across iterations the strength of which depends upon the value of \( \epsilon \).

Clearly, if we set \( \epsilon = \delta = 0.5 \) we obtain the vanilla RJMCMC algorithm.

For the uncorrelated case, we simply remove the central point and update the \( u^i_{k,j} \) independently of one another, so that
\[
u^i_{k,j} = \begin{cases} 
\frac{u^i_{k,j} - 1}{u^i_{k,j}} & \frac{u^i_{k,j} - 1}{u^i_{k,j}} > 1 \\
\frac{u^i_{k,j}}{u^i_{k,j}} & 0 < \frac{u^i_{k,j}}{u^i_{k,j}} \leq 1 \\
\frac{u^i_{k,j} + 1}{u^i_{k,j}} & \frac{u^i_{k,j} + 1}{u^i_{k,j}} \leq 0
\end{cases}
\]
so that each \( u^i_{k,j} \) follows its own random walk around the ring over time.

In the next section, we discuss the application of these methods and compare their performance with the vanilla RJMCMC algorithm.

### 7.3. Results
Here, we consider the enzyme data set introduced by Richardson and Green (1997) and adopt the priors that they suggest for the model parameters. In order to better explore the tail regions of the model space, we shall take a flat prior over the number of components, limiting the number to lie between 1 and 5. We run each simulation for 200,000 iterations, discarding the initial 100,000 as part of the burn-in and compare the vanilla algorithm with the uncorrelated and correlated auxiliary variable algorithms using the same starting points and pseudo-random seeds. Trace plots of the number of components are provided in Figure 3.

*Figure 3 about here*

From Figure 3 it is clear that the autocorrelated methods are performing significantly better than the vanilla. For example, the 2-component model is visited only rarely by the vanilla algorithm (especially in the second half of the simulation), but much more regularly by the others. The corresponding acceptance rates for between-model moves is 4.3\%, 17.7\% and 5.3\% for the vanilla, uncorrelated and correlated AV methods respectively. The slight “blockiness” apparent in the trace plot for the correlated AV method is due to the introduction of the central point. At certain positions on the ring, moves which increase dimension are preferred, whilst at other positions, moves which decrease dimension are preferred. This leads to the slightly more “blocky” trace plot in Figure 3 and is an extremely useful property to have in the presence of extreme multimodality. The short-term persistence of particular move types in the correlated AV method allows the chain to explore further into the tails and provides greater potential for jumping between modes.

It should be noted here that the poor performance of the vanilla algorithm in this case is due to the restriction of the model space to those models with 5 or fewer components. Though these models
contain over 80% of the posterior mass, the existence of higher order models improves mixing by allowing the chain to visit larger models thereby splitting observations into smaller groups which can subsequently be recombined in different ways as the number of models decreases again. This allows the algorithm to move between modes corresponding to different arrangements of the observations within components. Thus, the vanilla algorithm is able to mix well in this case. When we restrict the model space, the vanilla algorithm struggles to move between these modes. However, the auxiliary variable methods perform equally well in both the restricted and unrestricted model order cases.

In practice, it is difficult to determine how the range of models to be considered will affect the mixing properties of the corresponding algorithm. Indeed, in some cases, the range of models may be constrained to a small set by factors relating to the problem at hand. Thus, it is impossible to predict whether or not the range of models allowed are sufficient to enable the vanilla algorithm to mix. This example illustrates that the AV methods will be unaffected and should perform well regardless. In fact, in the presence of extreme multimodality, the AV methods may perform significantly better than the vanilla method whatever the range of models considered.

8. Concluding remarks

We have introduced a collection of techniques for reversible jump proposal choice, firstly in the traditional setting as introduced by Green (1995) and secondly in the more flexible dual space setting. It is shown that a number of techniques used to construct proposals in Euclidean spaces (for example Langevin diffusion motivated methods, and Hamiltonian auxiliary variable techniques) can in fact be extended to our setting in this paper.

The results have been applied to varied Bayesian examples, AR model choice, finite mixtures, and Gaussian graphical model choice. The results show that the new techniques can produce considerable improvements over (even heavily pilot tuned) standard methods in many cases. However the results are far from being uniformly positive towards the use of our techniques, and an important question raised by our investigation asks in what classes of problems are our methods most successful. The relative performance of our methods in comparison to vanilla techniques seems to be best in more complex problems with large numbers of models and model spaces which are highly non-linear. Thus, the zeroth order method performs considerably better than the vanilla method in the graphical model examples, whereas the AR model choice example mixes perfectly adequately using vanilla methodology, so that only a marginal improvement in performance is observed with some of our methods.

As with Langevin algorithms, the methods proposed here could suffer from problems where the proposed variance values are totally inappropriate (as for example in the fowl bones example of Sectiongraph). This could be caused by unrepresentative centering points, or indeed by target densities within models not being sufficiently smooth. For this reason, it will be sensible in many applications to adopt a truncation on the algorithm scaling parameters.

It is clear that further work on the choice of centering points is necessary. This issue is only briefly touched on in our paper, where the non-identifiability and conditional maximisation methods are described. Apart from the promising idea introduced in Green (2000), this is a highly undeveloped and important area.

One subsidiary point that comes out of the examples studied is the issue of convergence diagnostics for reversible jump algorithms in general. It seems that the problems encountered with such diagnostics in Euclidean state spaces are exacerbated in the context of reversible jump algorithms on more complex spaces. These issues are investigated further in Brooks et al (2000), for example.

9. Acknowledgments

We are extremely grateful to Peter Green for very helpful discussions on the subject of this paper, and for allowing us access to the computer code which we modified in order to run the algorithms in the mixtures and graphical models examples. This work has been supported by EU TMR network ERB-FMRX-CT96-0095 on ‘Computational and Statistical methods for the analysis of spatial data.
References


Giudici, P. and P. J. Green (1999), Decomposable Graphical Gaussian Model Determination. *Biometrika* 86, 785–801


Green, P. J. and A. Mira (2000), Delayed rejection in Reversible Jump Metropolis-Hastings. Technical report, University of Bristol


Fig. 1. Fowl bones data: diagnostic plots on the number of edges present for the vanilla zeroth order and auxiliary variable methods.
Fig. 2. Cumulative number of models visited during simulation for vanilla (solid), zeroth order (dotted) and auxiliary variable (dashed) methods. Total numbers of models visited by vanilla, zeroth and auxiliary variable methods during simulation are 116, 209 and 245 respectively.
Fig. 3. Trace plot of number of components for the vanilla, uncorrelated and correlated auxiliary variable methods for the enzyme data set, considering only models with five components or less.