

# Adaptive Metropolis-Hastings samplers for the Bayesian analysis of large linear Gaussian systems

Stephen KH Yeung                      Darren J Wilkinson  
stephen.yeung@ncl.ac.uk    d.j.wilkinson@ncl.ac.uk

Department of Statistics, University of Newcastle, UK

## Abstract

This paper considers the implementation of efficient Bayesian computation for large linear Gaussian models containing many latent variables. A common approach is to implement a simple MCMC procedure such as the Gibbs sampler or data augmentation, but these methods are often unsatisfactory when the model is large. This motivates the need to develop other strategies for improving MCMC. This paper considers the combination of adapting algorithms with the Metropolis-Hastings scheme in the construction of efficient MCMC schemes with good mixing properties.

## 1 Introduction

Linear Gaussian Directed Acyclic Graph (DAG) models represent a wide range of statistical models. Consider the DAG model shown in Figure 1, where  $\sigma$  represents a collection of “parameters”. Conditional on  $\sigma$  is the set  $\theta$ , which represents the latent (unobservable) Gaussian variables, and conditional on both  $\sigma$  and  $\theta$  is the collection of observable Gaussian data,  $y$ . DAG models of this form often arise in the context of dynamic linear modelling, where the underlying stochastic process can be represented by a time evolving Gaussian process, and multilevel modelling with random-effects. Therefore, only DAG models of the form in Figure 1 will be considered in this paper.

If interest lies in the full joint distribution  $(\theta, \sigma|y)$ , then a standard Gibbs sampler can be constructed to sample from each element of  $\theta$  and  $\sigma$  in turn. In practice however, the mixing rate for the Gibbs sampler applied to such models is often too slow to be of practical

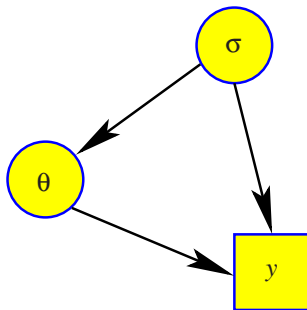


Figure 1: DAG representing the general model structure.

use (Gelfand, Sahu, and Carlin 1995). Roberts and Sahu (1997) suggest reparameterising the model, but an improvement in mixing is not always guaranteed.

It is widely acknowledged that blocking strategies improve mixing of MCMC schemes (Liu, Wong, and Kong 1994; Roberts and Sahu 1997; Hobert and Geyer 1998), but until recently, concern about the computational complexity of implementing such schemes has limited their application to large linear systems. Fortunately, for DAG models, the topology of the DAG can be exploited in a variety of ways for the development of efficient computation and sampling algorithms. Wilkinson and Yeung (2001) describe a range of techniques based on the utilisation of local computation strategies on the DAG to enable block sampling.

The remainder of the paper is organised as follows. In section 2, efficient block MCMC schemes are introduced. In section 3, two adapting strategies for tuning the proposal distribution are described; section 3.1 describes a stochastic search method, and section 3.2 describes a scheme based on fitting a quadratic response surface. An illustrative example application is presented in section 4, before conclusions are drawn in section 5.

## 2 MCMC algorithms

### 2.1 Data augmentation

Data augmentation (Tanner and Wong 1987) is a scheme that enables dependent samples to be drawn from  $(\theta, \sigma|y)$ . This involves alternately sampling from the two conditional distributions,  $(\theta|\sigma, y)$  and  $(\sigma|\theta, y)$ . Sampling from  $(\sigma|\theta, y)$  is trivial, based on semi-conjugate updates as in the Gibbs sampling scheme. Sampling from  $(\theta|\sigma, y)$  requires sampling from the MVN conditioned on the Gaussian observations,  $y$ . This can be non-trivial if the latent variable vector is large or if there are many observations, but is always tractable for linear Gaussian systems. However, large Gaussian DAG models can be constructed and sampled from using the ‘C’ software library, GDAGSIM (<http://www.staff.ncl.ac.uk/d.j.wilkinson/software/gdagsim/>).

The problem with the data augmentation scheme is that poor mixing can still occur if  $\theta$  and  $\sigma$  are highly correlated. One method to improve mixing is to construct a Metropolis-Hastings scheme, and the following sections describe two such methods.

### 2.2 Marginal update scheme

The approach here is to focus on the marginal posterior distribution  $(\sigma|y)$  by “integrating out”  $\theta$  completely from the problem. Thus, the problem becomes one of implementing a Metropolis-Hastings MCMC scheme with  $(\sigma|y)$  as the target distribution.

Here, a *proposal*,  $\sigma^*$ , is sampled from some proposal density  $f(\sigma^*|\sigma)$ , where  $\sigma$  is the current value, and accepted with probability  $\min\{1, A\}$  where

$$\begin{aligned} A &= \frac{[\sigma^*|y]f(\sigma|\sigma^*)}{[\sigma|y]f(\sigma^*|\sigma)} \\ &= \frac{[\sigma^*][y|\sigma^*]f(\sigma|\sigma^*)}{[\sigma][y|\sigma]f(\sigma^*|\sigma)}. \end{aligned} \tag{1}$$

Direct calculation of the marginal likelihood  $[y|\sigma]$  is tractable for linear Gaussian systems (Wilkinson and Yeung 2001). If the proposal density  $f(\cdot|\cdot)$  is symmetric, then (1) reduces to

$$A = \frac{[\sigma^*][y|\sigma^*]}{[\sigma][y|\sigma]}. \tag{2}$$

See section 4.3 for an example of such an approach in practice.

If there is interest in the distribution of  $\theta$ , then a sample may be drawn from  $(\theta|\sigma, y)$ , as in the data augmentation scheme, at each iteration, since these will be averaged over the posterior distribution of  $\sigma$  in order to give (in the limit) draws from  $(\theta|y)$ . This approach motivates the single block scheme discussed in the next section.

### 2.3 Single block Metropolis-Hastings scheme

Another way to sample from  $(\theta, \sigma|y)$  is to implement a single block Metropolis-Hastings scheme. A proposal,  $(\theta^*, \sigma^*)$ , is obtained by sampling  $\sigma^*$  from some proposal density  $f(\sigma^*|\sigma, \theta)$ , and  $\theta^*$  from  $(\theta^*|\sigma^*, y)$ , and jointly accepted with probability  $\min\{1, A\}$  where

$$\begin{aligned} A &= \frac{[\sigma^*, \theta^*|y]f(\sigma|\sigma^*, \theta^*)[\theta|\sigma, y]}{[\sigma, \theta|y]f(\sigma^*|\sigma, \theta)[\theta^*|\sigma^*, y]} \\ &= \frac{[\sigma^*][\theta^*|\sigma^*][y|\sigma^*, \theta^*]f(\sigma|\sigma^*, \theta^*)[\theta|\sigma, y]}{[\sigma][\theta|\sigma][y|\sigma, \theta]f(\sigma^*|\sigma, \theta)[\theta^*|\sigma^*, y]} \\ &= \frac{[\sigma^*][y|\sigma^*]f(\sigma|\sigma^*, \theta^*)}{[\sigma][y|\sigma]f(\sigma^*|\sigma, \theta)}. \end{aligned} \quad (3)$$

The difference between this and the marginal update scheme is that  $\sigma^*$  can be sampled from more sophisticated proposal distributions, including “heated” versions of the full conditionals,  $(\sigma|\theta, y)$ . See section 4.4 for an example of such an approach in practice.

## 3 Adaptive Metropolis-Hastings

It is widely accepted that a well chosen proposal distribution is crucial to the performance of a Metropolis-Hastings sampler that mixes well and has low autocorrelations. A naive choice may lead to a sampler that does not mix particularly well. In practice however, making a good choice *a priori* is difficult.

The problem of choosing the “best” proposal distribution has motivated the development of Metropolis-Hastings schemes incorporating adaptive algorithms. These do not simply rely on a homogeneous proposal, but instead allow the proposal to change during the run. Gelfand and Sahu (1994) described an approach based on regarding transition kernels as stochastic matrices and comparing their eigenvalues to choose the better proposal. Haario, Saksman, and Tamminen (1999) adapt the proposal distribution according to the covariance calculated from a fixed number of past sampled values. Browne and Draper (2000) adapt the proposal during a preliminary run in order to maintain the acceptance rate of  $x\%$  within what they refer to as a *tolerance interval* for all parameters of interest. Once this has been achieved, adaptation stops and the main monitoring run proceeds with the current proposals. Gilks, Roberts, and Sahu (1998) described a similar scheme based on acceptance rates and tolerance intervals, though by allowing adaptation to occur at *regeneration times*, their scheme allows the proposal to be changed on-line indefinitely, without disturbing the stationary distribution.

In this paper we focus our attention on minimizing the autocorrelations to improve mixing in a preliminary period, at which point adaptation stops and the main monitoring run proceeds. Consider the proposal distribution,  $f(\sigma^*|\cdot)$  from section 2. In this paper we consider adaptive Metropolis-Hastings to tune  $f(\sigma^*|\cdot)$  in order to minimize the autocorrelations. For some arbitrary  $f(\sigma^*|\cdot)$ , let this depend on a “tuning parameter”  $\kappa = (\kappa_1, \dots, \kappa_d)' \in \mathbb{R}^d$ . Now  $f(\sigma^*|\cdot)$  can be modified by varying  $\kappa$ , and thus focus now shifts towards finding an estimate of the optimal value,  $\kappa_{Opt}$ , that minimizes the autocorrelations. We denote this estimate by  $\kappa_{\widehat{Opt}}$ .

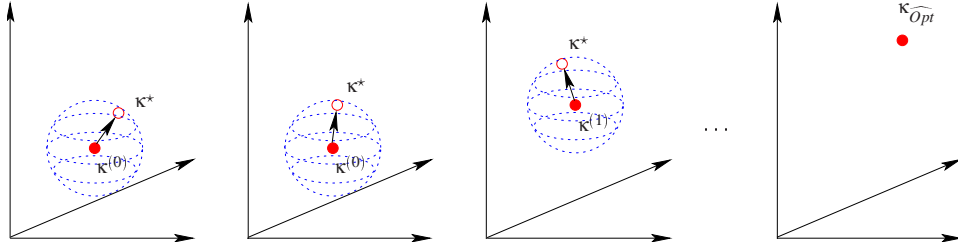


Figure 2: The stages of the stochastic search method.

The exact role of  $\kappa$  depends on the proposal used in the Metropolis-Hastings step. For example, in a random walk Metropolis algorithm, the proposal distribution is typically of the form  $q(\phi^*|\phi) = N(\phi, \sigma^2)$ . In this case,  $\kappa = \sigma^2 \in (0, \infty)$ . See section 4 for two alternative roles of  $\kappa$ .

The remainder of this section describes two adaptive algorithms. Both of the algorithms are based on an iterative process that involves a series of *experiments* and *evaluation*. The term experiment refers to running an MCMC sampler for some given  $\kappa$  to obtain a batch of size  $N$ , from which the *maximum* lag  $p$  autocorrelation of the output is evaluated. This maximum is computed over the components of the chain, and we denote this value by  $lag_p^{(max)}(\kappa)$ . It represents an easily computable lower bound for the *maximal* lag  $p$  autocorrelation, which is known to determine the overall convergence rate of the Markov chain; see Liu, Wong, and Kong (1994) for further details. Clearly, a better lower bound could be obtained by using the maximum eigenvalue of the lag  $p$  autocorrelation matrix, but the improvement in performance of the adaptation method gained by adopting such an approach has yet to be investigated.

### 3.1 Stochastic search (SS) algorithm

This algorithm proceeds by carrying out the process of experiments and evaluation as described in section 3. This process is repeated until  $\kappa_{Opt}$  is obtained, or until a prespecified number of  $n$  phases, according to the following algorithm:

1. Initialise counter to  $j = 1$  and initialise  $\kappa^{(0)}$ .
2. Generate a proposal  $\kappa^*|\kappa^{(j-1)}$ .
3. If  $lag_p^{(max)}(\kappa) > lag_p^{(max)}(\kappa^*)$ , put  $\kappa = \kappa^*$ .  
Otherwise put  $\kappa = \kappa$ .
4. Change counter from  $j$  to  $j + 1$  and return to step 2.

Figure 2 helps to convey the idea.

An obvious question is how is  $\kappa^*$  proposed? A sensible proposal distribution can be formulated by considering the space within which  $\kappa_{Opt}$  is presumed to lie, but a Normal random walk with some variance seems like a good choice. See section 4 for an example of this approach in practice.

### 3.2 Quadratic response surface (QRS) algorithm

Consider an experiment where  $\kappa^{(i)}$  is held at a different level for  $i = 1, \dots, n$ . A good choice is to sample  $\kappa^{(i)}$  uniformly over all permissible values of  $\mathbb{R}^d$ . However, if a smaller space

within which  $\kappa_{Opt}$  lies is known beforehand, i.e. if  $\kappa_{Opt} \in \mathbb{S} \subseteq \mathbb{R}^d$ , then a search within the subspace  $\mathbb{S}$  is sufficient. Hence the algorithm is initialised by running the sampler for some given  $\kappa^{(i)} \in \mathbb{S}$  to obtain a batch of size  $N$ , and the response  $y_i \left( = \text{lag}_p^{(max)} \left( \kappa^{(i)} \right) \right)$  is calculated for  $i = 1, \dots, n$ .

Cochran and Cox (1992) showed how  $y_i$  can be related to  $\kappa^{(i)}$  by using a *response surface* function. The general form of a quadratic response surface is given by

$$f(\kappa) = \kappa' \mathbf{A} \kappa + \underline{b}' \kappa + c$$

where

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1d} \\ a_{21} & a_{22} & \dots & a_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ a_{dd} & a_{d2} & \dots & a_{dd} \end{pmatrix}, \underline{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_d \end{pmatrix}, c = \text{constant},$$

and  $\mathbf{A}$  is symmetric and positive-definite.

For every  $y_i$  there is an associated experimental error,  $\varepsilon_i$ , where  $y_i = f(\kappa^{(i)}) + \varepsilon_i$ . The algorithm proceeds by obtaining estimates for  $\mathbf{A}$ ,  $\underline{b}$ , and  $c$  by minimising the sum of the squared error terms using a constrained optimisation approach to ensure  $\mathbf{A}$  is positive definite. Denote these estimates by  $\hat{\mathbf{A}}$ ,  $\hat{\underline{b}}$  and  $\hat{c}$  respectively. Given these estimates,  $\kappa_{\widehat{Opt}}$  is found by finding the stationary value of  $f(\kappa)$ .

The algorithm can be summarised as follows:

1. Observe the pairs  $(\kappa^{(i)}, y_i)$  for  $i = 1, \dots, n$ , where

$$y_i = f(\kappa^{(i)}) + \varepsilon_i \quad \text{for } i = 1, \dots, n,$$

where  $\varepsilon_i$  is the experimental error associated with the observation  $y_i$ .

2. Minimize  $G = \sum_{i=1}^n \varepsilon_i^2$  using constrained optimisation to obtain  $\hat{\mathbf{A}}$ ,  $\hat{\underline{b}}$  and  $\hat{c}$ .
3. Find the stationary value of  $f(\kappa)$  by solving

$$\frac{d(f(\kappa))}{d\kappa} = 2\hat{\mathbf{A}}\kappa + \hat{\underline{b}} = 0.$$

4.  $\kappa_{\widehat{Opt}}$  is given by  $-\frac{1}{2}\hat{\mathbf{A}}^{-1}\hat{\underline{b}}$ .

See section 4 for an example of a suitable choice for  $n$  and  $N$  and an illustration of this algorithm.

### 3.3 Hybrid quadratic response surface with stochastic search algorithm

Another feasible method would be to combine the two techniques of section 3.1 and section 3.2. This method requires two stages to obtain  $\kappa_{\widehat{Opt}}$ . First, the QRS algorithm is implemented to obtain a ‘‘best’’ guess for  $\kappa_{\widehat{Opt}}$ , which we assume lies in the optimal neighbourhood of  $\kappa_{Opt}$ . Next,  $\kappa_{\widehat{Opt}}$  from the first stage is used as the initial starting value,  $\kappa^{(0)}$ , in the SS algorithm.

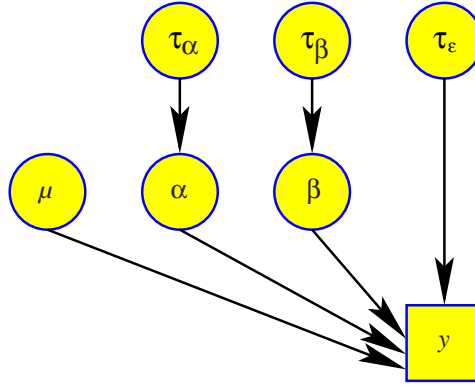


Figure 3: Reduced DAG for the dynamic 3 level model with random effects

## 4 Application: Dynamic 3 level model with random effects

Consider data arising from a sampling inspection procedure, where at time  $i$ ,  $q_i$  batches are sampled. The  $j$ th batch sampled at time  $i$  contains  $n_{ij}$  items. The dynamic 3 level model with random effects is appropriate for modelling highly structured data of this form. In the next section we introduce the model.

### 4.1 Model structure

The dynamic 3 level model with random effects has the form

$$y_{ijk} = \mu + \alpha_i + \beta_{ij} + \varepsilon_{ijk}, \quad i = 1, \dots, p, \quad j = 1, \dots, q_i, \quad k = 1, \dots, r_{ij}, \quad (4)$$

where  $\mu \sim N(a, 1/b)$ ,  $\beta_{ij} \sim N(0, 1/\tau_\beta)$ ,  $\varepsilon_{ijk} \sim N(0, 1/\tau_\varepsilon)$  are all independent. Further, suppose the  $\alpha_i$  follow a normal random walk, where  $\alpha_1 \sim N(0, 1/\tau_\alpha)$  and  $\alpha_i | \alpha_{i-1} \sim N(\alpha_{i-1}, 1/\tau_\alpha)$  for  $i = 2, \dots, p$ .

In the case where the precision components are unknown, the model is completed with prior specifications for these quantities, typically of the independent semi-conjugate form

$$\tau_\alpha \sim \Gamma(a_\alpha, b_\alpha), \quad \tau_\beta \sim \Gamma(a_\beta, b_\beta), \quad \tau_\varepsilon \sim \Gamma(a_\varepsilon, b_\varepsilon).$$

The DAG for this model is as given in Figure 3, where  $\alpha = (\alpha_1, \dots, \alpha_p)'$ , and  $\beta = (\beta_1, \dots, \beta_q)'$ .

The target density of this model is  $[\mu, \alpha, \beta, \tau_\alpha, \tau_\beta, \tau_\varepsilon | y]$ . A standard univariate Gibbs sampler can be constructed to sample from this density using the BUGS package (Spiegelhalter, Thomas, Best, and Gilks 1996). However, the mixing of the Gibbs sampler applied to this model is often too slow due to high positive correlations between model parameters (Roberts and Sahu 1997). Gelfand, Sahu, and Carlin (1995) suggest that in multilevel mixed models, forming a hierarchically centred parameterisation of the model will often lead to a Gibbs sampler with better mixing properties, but provide no general guidance as to the optimal parameterisation. Although a full hierarchically centred parameterisation is not convenient for a model of this form, partially centred parameterisations are possible, but do not improve mixing in general. No convenient parameterisation of model (4) produces a sampler that mixes as well as the block schemes analysed in the following sections.

## 4.2 Data augmentation scheme

If for (4) we use  $\theta$  to represent the collection of latent Gaussian variables and  $\sigma$  to represent the collection of precision components, so that  $\theta = (\mu, \alpha, \beta)$ , and  $\sigma = (\tau_\alpha, \tau_\beta, \tau_\epsilon)$ , then clearly the DAG of Figure 3 reduces to the form of the DAG in Figure 1. It is possible to obtain a sample from  $(\theta, \sigma | y)$  by simulating from  $(\theta | \sigma, y)$  and  $(\sigma | \theta, y)$ , as described in section 2.1, using the following algorithm:

1. Initialise the iteration counter to  $m = 1$ . Initialise the state of the chain to some initial values  $(\sigma^{(0)}, \theta^{(0)})$ .
2. Obtain a new set of values  $(\sigma^{(m)}, \theta^{(m)})$  from  $(\sigma^{(m-1)}, \theta^{(m-1)})$  by successive generation of values

$$\begin{aligned}\sigma^{(m)} &\sim (\sigma | \theta^{(m-1)}, y) \\ \theta^{(m)} &\sim (\theta | \sigma^{(m)}, y).\end{aligned}\tag{5}$$

3. Change counter  $m$  to  $m + 1$  and return to step 2.

Simulation of  $\sigma$  from  $(\sigma | \theta, y)$  requires no more than sampling  $\tau_\alpha, \tau_\beta, \tau_\epsilon$  using Gibbs sampling. The vector of latent variables  $\theta$ , is drawn from  $(\theta | \sigma, y)$  using the `GDAGsim` software.

## 4.3 Marginal update scheme

In this example there are three precision variables,  $\sigma = (\tau_\alpha, \tau_\beta, \tau_\epsilon)$ . We update  $\sigma$  using the MCMC scheme from [Knorr-Held and Rue \(2000\)](#). For each precision variable  $\tau_i$ , let  $\kappa_i \in (0, 1)$ , and sample  $\tau_i^* \sim U[\tau_i \kappa_i, \tau_i / \kappa_i]$ ,  $i \in \{\tau_\alpha, \tau_\beta, \tau_\epsilon\}$ . Having obtained the proposed parameter values  $\sigma^* = (\tau_\alpha^*, \tau_\beta^*, \tau_\epsilon^*)$ , accept these with probability  $\min\{1, A\}$ , where

$$A = \frac{[\tau_\alpha^*][\tau_\beta^*][\tau_\epsilon^*][y|\sigma^*]\tau_\alpha\tau_\beta\tau_\epsilon}{[\tau_\alpha][\tau_\beta][\tau_\epsilon][y|\sigma]\tau_\alpha^*\tau_\beta^*\tau_\epsilon^*}.$$

## 4.4 Single block Metropolis-Hastings scheme

For each precision variable  $\tau_i$ , the conditional density  $[\tau_i | \theta, y]$  is known. So suppose a ‘‘heated’’ version of  $[\tau_i | \theta, y]$  is chosen for  $f(\tau_i^* | \tau_i, \theta)$ , which also depends on a ‘‘temperature’’  $\kappa_i \in (0, 1)$ , that determines how diffuse the proposal is relative to the full-conditional. If  $(\tau_i | \theta, y)$  has a  $\Gamma(a_i(\theta, y), b_i(\theta, y))$  distribution, then choosing a proposal of the form  $f(\tau_i^* | \tau_i, \theta, \kappa_i) = \Gamma(\tau_i^*; a_i(\theta, y)\kappa_i, b_i(\theta, y)\kappa_i)$  inflates the variance of the full-conditional, proportionally to  $\kappa_i$ , whilst leaving the mean unchanged. The algorithm thus proceeds by obtaining proposed parameter values  $\sigma^* = (\tau_\alpha^*, \tau_\beta^*, \tau_\epsilon^*)'$ , then sampling a corresponding  $\theta^*$  from  $[\theta^* | \sigma^*, y]$ , and jointly accepting  $(\theta^*, \sigma^*)$  with probability  $\min\{1, A\}$ , where

$$A = \frac{[\tau_\alpha^*][\tau_\beta^*][\tau_\epsilon^*][y|\sigma^*]f(\tau_\alpha|\tau_\alpha^*, \theta^*, \kappa_\alpha)f(\tau_\beta|\tau_\beta^*, \theta^*, \kappa_\beta)f(\tau_\epsilon|\tau_\epsilon^*, \theta^*, \kappa_\epsilon)}{[\tau_\alpha][\tau_\beta][\tau_\epsilon][y|\sigma]f(\tau_\alpha^*|\tau_\alpha, \theta, \kappa_\alpha)f(\tau_\beta^*|\tau_\beta, \theta, \kappa_\beta)f(\tau_\epsilon^*|\tau_\epsilon, \theta, \kappa_\epsilon)}.$$

## 4.5 Results

In this section, the Gibbs sampler is applied to (4) using simulated data. We compare the three block MCMC schemes from section 2 against the Gibbs sampler. In particular, we illustrate the adaptive algorithms from section 3 applied to the Metropolis-Hastings schemes of section 4.3 and section 4.4. In this application, we set  $(N, n, p) = (1000, 100, 10)$ .

For all schemes, the MCMC sampler was run for 60,000 iterations, with the first 10,000 discarded as burn-in, and the remaining 50,000 used for the main monitoring run. The data set was simulated with  $p = 15$ ,  $q_i = 15$ ,  $r_{ij} = 5$ ,  $\forall i, j$ , and true values for  $(\mu, \tau_\alpha, \tau_\beta, \tau_\epsilon) = (10, 1, 100, 1)$ . For the purpose of this example, inferences were made for the Gaussian variables  $\mu, \alpha_1, \beta_{11}$  and the precision variables  $\tau_\alpha, \tau_\beta, \tau_\gamma$ . The hyperparameters used were  $a = 0, b = 0.0001, a_i = 0.001, b_i = 0.001, i \in \{\alpha, \beta, \epsilon\}$ .

For the adaptive algorithm of section 3.1,  $\kappa^{(0)}$  was initialised to  $(0.5, 0.5, 0.5)$  for the marginal update scheme of section 4.3, and  $(1, 1, 1)$  for the single block scheme of section 4.4. For both schemes, the proposal  $\kappa^*$  was generated according to

$$\kappa^* | \kappa^{(j-1)} \sim N \left( \kappa^{(j-1)}, \left( \text{lag}_p(\kappa^{(j-1)}) \right)^2 \right).$$

This proposal uses the property that if the autocorrelations for a component  $\tau_i \in \{\alpha, \beta, \epsilon\}$  is large, then it can be assumed that  $\kappa_i$  is not within the optimal neighbourhood of  $\kappa_{\widehat{Opt}}$ , and thus  $\kappa_i^*$  can make a large jump within  $\mathbb{R}^i$ . The converse is also true. After the adaptive phase,  $\kappa_{\widehat{Opt}} = (0.725, 0.453, 0.967)$  for the marginal update scheme of section 4.3, and  $\kappa_{\widehat{Opt}} = (0.370, 0.009, 0.580)$  for the single block Metropolis-Hastings scheme of section 4.4.

For the adaptive algorithm of section 3.2, a minimisation routine was applied in order to obtain  $\hat{\mathbf{A}}, \hat{\mathbf{L}}$  and  $\hat{c}$ , from which  $\kappa_{\widehat{Opt}}$  could be determined. After the adaptive phase,  $\kappa_{\widehat{Opt}} = (0.556, 0.434, 0.904)$  for the marginal update scheme of section 4.3, and  $\kappa_{\widehat{Opt}} = (0.987, 0.003, 0.599)$  for the single block Metropolis-Hastings scheme of section 4.4.

The autocorrelations for the four MCMC schemes are shown in Figure 4. Figure 4 (c) and (e) shows the autocorrelations for the SS algorithm applied to the MCMC schemes of section 4.3 and section 4.4, respectively. Figure 4 (d) and (f) shows the autocorrelations for the QRS algorithm applied to the MCMC schemes of section 4.3 and section 4.4, respectively.

Figure 4 (b)-(f) clearly shows the benefits of the MCMC schemes of section 2 over the standard Gibbs sampling scheme (a), where the sampled values for many of the Gaussians in  $\theta$  and  $\log(\tau_\beta)$  are highly autocorrelated. The data augmentation scheme of section 4.2 improves on the Gibbs sampler by reducing the autocorrelations for the Gaussians in  $\theta$ . However, the autocorrelations for  $\log(\tau_\beta)$  are not much improved. The tuning parameters for both the marginal update scheme ((c) and (e)) and the single block Metropolis-Hastings scheme ((d) and (f)) were computed using the adaptive algorithms of section 3 to reduce the *maximum* autocorrelation. It is therefore not surprising to observe the significant reduction in the autocorrelations for the problem variable,  $\log(\tau_\beta)$ .

This example illustrates that when the standard Gibbs sampler and data augmentation schemes fail to produce samples with satisfactory mixing properties, the marginal update scheme and single block Metropolis-Hastings scheme may provide a significant improvement. In addition, adaptive algorithms can be used to find the optimal set of tuning parameters for a given proposal distribution, in order to reduce the autocorrelations.

## 5 Conclusions

Many statistical models have DAGs that can be reduced to the form of the general GDAG as introduced in this paper. The usual approach to making inferences about these models is to implement a Gibbs sampler or data augmentation scheme, but this paper has shown how inadequate these approaches are when the model is large. Other block MCMC schemes such as the marginal update and single block Metropolis-Hasting schemes offer a significant

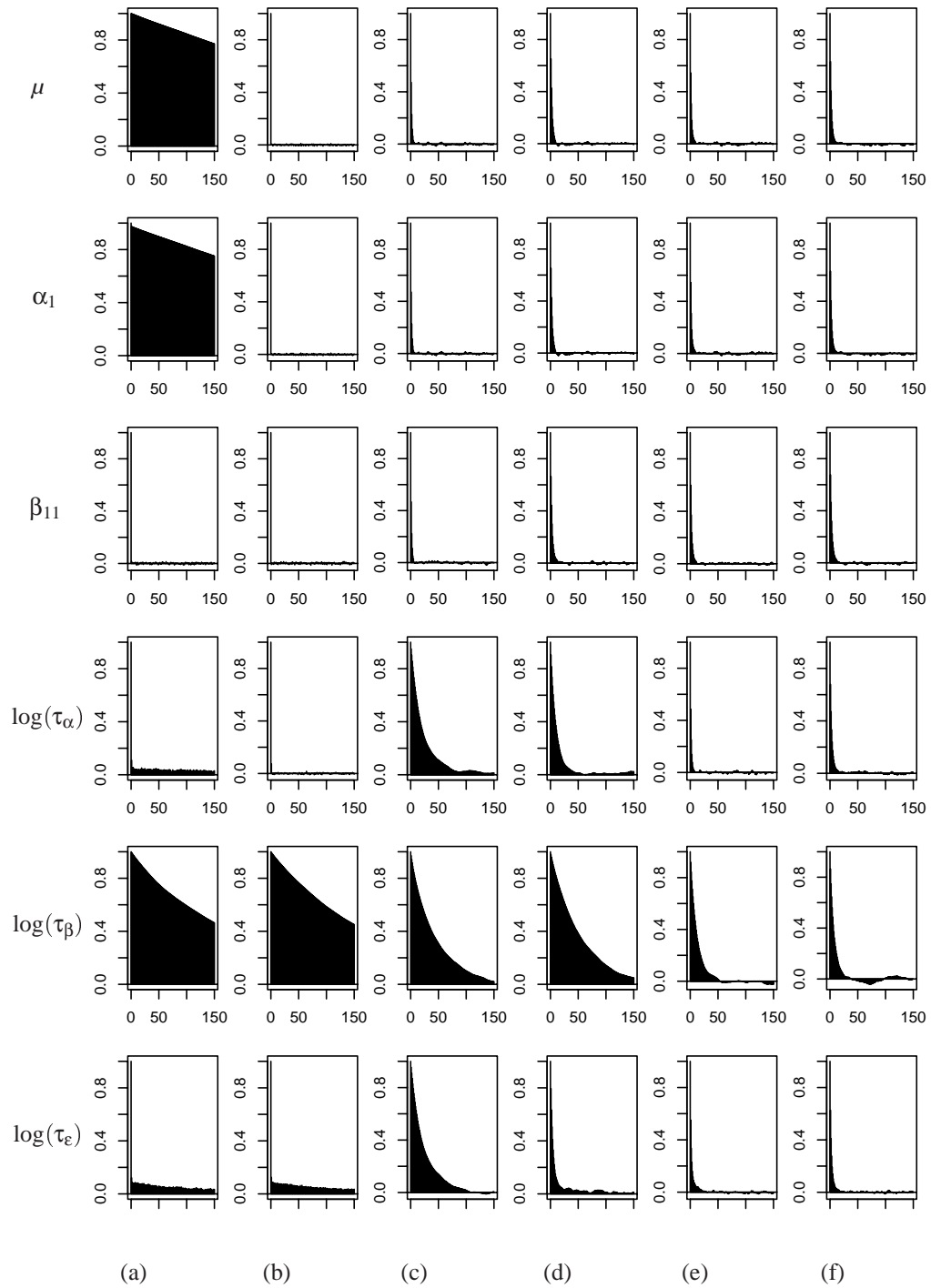


Figure 4: Autocorrelation plots for a selection of the variables from the output of the 6 MCMC schemes; (a) Gibbs sampler, (b) Data augmentation, (c) SS marginal update, (d) QRS marginal update, (e) SS single block sampler, (f) QRS single block sampler.

improvement, but the problem becomes one of choosing an optimal proposal distribution. By allowing any arbitrary proposal to depend on a set of tuning parameters, this paper has described two adaptive algorithms that obviate the need to use ad-hoc tuning methods to find an optimal set.

## References

- Browne, W. J. and D. Draper (2000). Implementation and performance issues in the Bayesian and likelihood fitting of multilevel models. *Computational Statistics* 15(3), 391–420.
- Cochran, W. G. and G. M. Cox (1992). *Experimental Designs* (second ed.). New York: Wiley.
- Gelfand, A. E. and S. K. Sahu (1994). On Markov chain Monte Carlo acceleration. *Journal of the American Statistical Association* 3(3), 261–276.
- Gelfand, A. E., S. K. Sahu, and B. P. Carlin (1995). Efficient parameterisations for normal linear mixed models. *Biometrika* 82(3), 479–488.
- Gilks, W. R., G. O. Roberts, and S. K. Sahu (1998). Adaptive Markov chain Monte Carlo through regeneration. *Journal of the American Statistical Association* 93(443), 1045–1054.
- Haario, H., E. Saksman, and J. Tamminen (1999). Adaptive proposal distribution for random walk Metropolis algorithm. *Computational Statistics* 14(3), 375–395.
- Hobert, J. P. and C. J. Geyer (1998). Geometric ergodicity of Gibbs and block Gibbs samplers for a hierarchical random effects model. *Journal of Multivariate Analysis* 67, 414–430.
- Knorr-Held, L. and H. Rue (2000). On block updating in Markov random field models for disease mapping. Technical Report Discussion paper No. 210, University Munich, Institute of Statistics. Available at <ftp://ftp.stat.uni-muenchen.de/pub/leo/block.ps>.
- Liu, J. S., W. H. Wong, and A. Kong (1994). Covariance structure of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika* 81, 27–40.
- Roberts, G. O. and S. K. Sahu (1997). Updating schemes, correlation structure, blocking and parameterisation for the Gibbs sampler. *Journal of the Royal Statistical Society B*:59(2), 291–317.
- Spiegelhalter, D., A. Thomas, N. G. Best, and W. Gilks (1996). *BUGS: Bayesian inference using Gibbs sampling, Version 0.5, (version ii)*. MRC Biostatistics Unit, Cambridge.
- Tanner, M. A. and W. H. Wong (1987). The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association* 82(398), 528–540.
- Wilkinson, D. J. and S. K. H. Yeung (2001). A sparse matrix approach to Bayesian computation in large linear models. Statistics Preprint STA01,2, University of Newcastle.