The random walk Metropolis: linking theory and practice through a case study.

Chris Sherlock[1,3], Paul Fearnhead[1], and Gareth O. Roberts[2]

1. Department of Mathematics and Statistics, Lancaster University, Lancaster, LA1 4YF, UK

2. Department of Statistics, University of Warwick, Coventry, CV4 7AL, UK.

3. Correspondence should be addressed to Chris Sherlock.
(e-mail: c.sherlock@lancs.ac.uk).

Summary: The random walk Metropolis (RWM) is one of the most common Markov Chain Monte Carlo algorithms in practical use today. Its theoretical properties have been extensively explored for certain classes of target, and a number of results with important practical implications have been derived. This article draws together a selection of new and existing key results and concepts and describes their implications. The impact of each new idea on algorithm efficiency is demonstrated for the practical example of the Markov modulated Poisson process (MMPP). A reparameterisation of the MMPP which leads to a highly efficient RWM within Gibbs algorithm in certain circumstances is also developed.

Keywords: *random walk Metropolis, Metropolis-Hastings, MCMC, adaptive MCMC, MMPP*

1 Introduction

Markov chain Monte Carlo (MCMC) algorithms provide a framework for sampling from a target random variable with a potentially complicated probability distribution $\pi(\cdot)$ by generating a Markov chain $\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots$ with stationary distribution $\pi(\cdot)$. The single most widely used sub-class of MCMC algorithms is based around the random walk Metropolis (RWM).

Theoretical properties of RWM algorithms for certain special classes of target have been investigated extensively. Reviews of RWM theory have, for example, dealt with optimal scaling and posterior shape (Roberts and Rosenthal, 2001), and convergence (Roberts, 2003). This article does not set out to be a comprehensive review of all theoretical results pertinent to the RWM. Instead the article reviews and develops specific aspects of the theory of RWM efficiency in order to tackle an important and difficult problem: inference for the Markov modulated Poisson process (MMPP). It includes sections on RWM within Gibbs, hybrid algorithms, and adaptive MCMC, as well as optimal scaling, optimal shaping, and convergence. A strong emphasis is placed on developing an intuitive understanding of the processes behind the theoretical results, and then on using these ideas to improve the implementation. All of the RWM algorithms described in this article are tested against data sets arising from MMPPs. Realised changes in efficiency are then compared with theoretical predictions.

Observed event times of an MMPP arise from a Poisson process whose intensity varies with the state of an unobserved continuous time Markov chain. The MMPP has been used to model a wide variety of clustered point processes, for example requests for web pages from users of the World Wide Web (Scott and Smyth, 2003), arrivals of photons from single molecule fluorescence experiments (Burzykowski *et al.*, 2003; Kou *et al.*, 2005), and occurences of a rare DNA motif along a genome (Fearnhead and Sherlock, 2006).

In common with mixture models and other hidden Markov models, inference for the MMPP is greatly complicated by a lack of knowledge of the hidden data. The likelihood function often possesses many minor modes since the data might be approximately described by a hidden process with fewer states. For this same reason the likelihood often does not appoach zero as certain combinations of parameters approach zero and/or infinity and so improper priors lead to improper posteriors (e.g. Sherlock, 2005). Further, as with many hidden data models the likelihood is invariant under permutation of the states, and this "labelling" problem leads to posteriors with several equal modes.

This article focusses on generic concepts and techniques for improving the efficiency of RWM algorithms whatever the statistical model. The MMPP provides a non-trivial testing ground for them. All of the RWM algorithms described in this article are tested against two simulated MMPP data sets with very different characteristics. This allows us to demonstrate the influence on performance of posterior attributes such as shape and orientation near the mode and lightness or heaviness of tails.

Section 2 introduces RWM algorithms and then describes theoretical and practical measures of algorithm efficiency in terms of both convergence and mixing. Next the two main theoretical approaches to determining efficiency are decribed, and the section ends with a brief overview of the MMPP and a description of the data analysed in this article. Section 3 introduces a series of concepts which allow potential improvements in the efficiency of a RWM algorithm. The intuition behind each concept is described, followed by theoretical justification and then details of one or more RWM algorithms motivated by the theory. Actual results are described and compared with theoretical predictions in Section 4, and the article is summarised in Section 5.

2 Background

In this section we introduce the background material on which the remainder of this article draws. We describe the random walk Metropolis algorithm and a variation, the random walk Metropolis-within-Gibbs. Both practical issues and theoretical approaches to algorithm efficiency are then discussed. We conclude with an introduction to the Markov modulated Poisson process and to the data sets used later in the article.

2.1 Random walk Metropolis algorithms

The **random walk Metropolis** (RWM) updating scheme was first applied in Metropolis *et al.* (1953) and proceeds as follows. Given a current value of the d-dimensional Markov chain, $\mathbf{X}$, a new value $\mathbf{X}^*$ is obtained by proposing a jump $\mathbf{Y}^* := \mathbf{X}^* - \mathbf{X}$ from the pre-specified Lebesgue density

$$\tilde{r}\left(\mathbf{y}^*;\lambda\right) := \frac{1}{\lambda^d}\, r\left(\frac{\mathbf{y}^*}{\lambda}\right), \tag{1}$$

with $r(\mathbf{y}) = r(-\mathbf{y})$ for all $\mathbf{y}$. Here $\lambda > 0$ governs the overall size of the proposed jump and (see Section 3.1) plays a crucial role in determining the efficiency of any algorithm. The proposal is then accepted or rejected according to acceptance probability

$$\alpha(\mathbf{x},\mathbf{y}^*) = \min\left(1, \frac{\pi(\mathbf{x}+\mathbf{y}^*)}{\pi(\mathbf{x})}\right). \tag{2}$$

If the proposed value is accepted it becomes the next current value ($\mathbf{X}' \leftarrow \mathbf{X}+\mathbf{Y}^*$), otherwise the current value is left unchanged ($\mathbf{X}' \leftarrow \mathbf{X}$).

The acceptance probability (2) is chosen so that the chain is reversible at equilibrium with stationary distribution $\pi(\cdot)$. In this article the transition kernel, that is the combined process of proposal and acceptance/rejection that leads from one element of the chain ($\mathbf{x}$) to the next, is denoted $P(\mathbf{x},\cdot)$.

An intuitive interpretation of the above formula is that "uphill" proposals (proposals which take the chain closer to a local mode) are always accepted, whereas "downhill" proposals are accepted with probability exactly equal to the relative "heights" of the posterior at the proposed and current values. It is precisely this rejection of some "downhill" proposals which acts to keep the Markov chain in the main posterior mass most of the time.

We now describe a generalisation of the RWM which acts on a target whose components have been split into k sub-blocks. In general we write $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_k)$, where $\mathbf{X}_i$ is the i^{th} sub-block of components of the current element of the chain. Starting from value $\mathbf{X}$, a single iteration of this algorithm cycles through all of the sub-blocks updating each in turn. It will therefore be convenient to define the shorthand

$$\begin{aligned} \mathbf{x}_i^{(B)} &:= \mathbf{x}_1', \ldots, \mathbf{x}_{i-1}', \mathbf{x}_i, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_k \\ \mathbf{x}_i^{(B)*} &:= \mathbf{x}_1', \ldots, \mathbf{x}_{i-1}', \mathbf{x}_i + \mathbf{y}_i^*, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_k \ , \end{aligned}$$

where $\mathbf{x}_j'$ is the *updated value* of the j^{th} sub-block. For the i^{th} sub-block a jump Y_i^* is proposed from symmetric density $\tilde{r}_i(\mathbf{y}; \lambda_i)$ and accepted or rejected according to acceptance probability $\pi\left(\mathbf{x}_i^{(B)*}\right) / \pi\left(\mathbf{x}_i^{(B)}\right)$. Since this algorithm is in fact a generalisation of both the RWM and of the Gibbs sampler (for a description of the Gibbs sampler see for example Gamerman and Lopes, 2006) we follow for example Neal and Roberts (2006) and call this the **random walk Metropolis-within-Gibbs** or RWM-within-Gibbs. The most commonly used random walk Metropolis within Gibbs algorithm, and also the simplest, is that employed in this article: here all blocks have dimension 1 so that each component of the parameter vector is updated in turn.

Even though each stage of the RWM-within-Gibbs is reversible, the algorithm as a whole is not. Reversible variations include the **random scan** RWM-within-Gibbs, wherein at each iteration a single component is chosen at random and updated conditional on all the other components.

Convergence of the Markov chain to its stationary distribution can be guaranteed for all of the above algorithms under quite general circumstances (e.g. Gilks *et al.*, 1996).

2.2 Algorithm efficiency

Adjacent elements of an MCMC Markov chain are correlated and the sequence of marginal distributions converges to $\pi(\cdot)$. Two main (and related) issues arise with regard to the efficiency of MCMC algorithms: convergence and mixing.

2.2.1 Convergence

In this article we will be concerned with practical determination of a point at which a chain has converged. The method we employ is simple heuristic examination of the trace plots for the different components of the chain. Note that since the state space is multi-dimensional it is not sufficient to simply examine a single component. Alternative techniques are discussed in Chapter 7 of Gilks *et al.* (1996).

Theoretical criteria for ensuring convergence (ergodicity) of MCMC Markov chains are examined in detail in Chapters 3 and 4 of Gilks *et al.* (1996) and references therein, and will not be discussed here. We do however wish to highlight the concepts of geometric and polynomial ergodicity. A Markov chain is **geometrically ergodic** with stationary distribution $\pi(\cdot)$ if

$$||P^n(\mathbf{x}, \cdot) - \pi(\cdot)||_1 \leq M(\mathbf{x})\ r^n \tag{3}$$

for some positive $r < 1$ and $M(\cdot)$. Here $||F(\cdot) - G(\cdot)||_1$ denotes the total variational distance between measures $F(\cdot)$ and $G(\cdot)$ (see for example Meyn and Tweedie, 1993). Efficiency of a geometrically ergodic algorithm is measured by the geometric rate of convergence, r, which over a large number of iterations is well approximated by the second largest eigenvalue of the transition kernel (the largest eigenvalue being 1, and corresponding to the stationary distribution $\pi(\cdot)$). Geometric ergodicity is usually a purely qualitative property since in general the constants $M(\mathbf{x})$ and r are not known. Crucially for practical MCMC however any *geometrically ergodic reversible Markov chain satisfies a central limit theorem for all functions with finite second moment with respect to* $\pi(\cdot)$. Thus there is a $\sigma_f^2 < \infty$ such that

$$n^{1/2}\left(\hat{f}_n - \mathbb{E}_\pi\left[f(\mathbf{X})\right]\right) \;\Rightarrow\; N(0,\sigma_f^2) \tag{4}$$

where $\Rightarrow$ denotes convergence in distribution. The central limit theorem (4) guarantees not only convergence of the Monte Carlo estimate (5) but also supplies its standard error, which decreases as $n^{-1/2}$.

When the second largest eigenvalue is also 1 a Markov chain is termed **polynomially ergodic** if

$$||P^n(\mathbf{x},\cdot) - \pi(\cdot)||_1 \leq M(\mathbf{x})\; n^{-r}$$

Clearly polynomial ergodicity is a weaker condition than geometric ergodicity. Central limit theorems for polynomially ergodic MCMC are much more delicate; see Jarner and Roberts (2002) for details.

In this article a chain is referred to as having "reached stationarity" or "converged" when the distribution from which an element is sampled is as close to the stationary distribution as to make no practical difference to any Monte-Carlo estimates.

An estimate of the expectation of a given function $f(X)$, which is more accurate than a naive Monte Carlo average over all the elements of the chain, is likely to be obtained by discarding the portion of the chain $\mathbf{X}_0,\ldots,\mathbf{X}_m$ up until the point at which it was deemed to have reached stationarity; iterations $1,\ldots m$ are commonly termed "burn in". Using only the remaining elements $\mathbf{X}_{m+1},\ldots,\mathbf{X}_{m+n}$ (with $m+n=N$) our Monte Carlo estimator becomes

$$\hat{f}_n := \frac{1}{n}\sum_{m+1}^{m+n} f(\mathbf{X}_i) \tag{5}$$

Convergence and burn in are not discussed any further here, and for the rest of this section the chain is assumed to have started at stationarity and continued for n further iterations.

2.2.2 Practical measures of mixing efficiency

For a stationary chain, $\mathbf{X}_0$ is sampled from $\pi(\cdot)$, and so for all $k > 0$ and $i \geq 0$

$$\text{Cov}\left[f(\mathbf{X}_k), f(\mathbf{X}_{k+i})\right] = \text{Cov}\left[f(\mathbf{X}_0), f(\mathbf{X}_i)\right]$$

This is the *autocorrelation* at lag i. Therefore at stationarity, from the definition in (4),

$$\sigma_f^2 := \lim_{n\to\infty} n\text{Var}\left[\hat{f}_n\right] = \text{Var}\left[f(\mathbf{X}_0)\right] + 2\sum_{i=1}^{\infty} \text{Cov}\left[f(\mathbf{X}_0), f(\mathbf{X}_i)\right]$$

provided the sum exists (e.g. Geyer, 1992). If elements of the stationary chain were independent then σ_f^2 would simply be $\text{Var}\left[f(\mathbf{X}_0)\right]$ and so a measure of the inefficiency of the Monte-Carlo estimate $\hat{f}_n$ relative to the perfect i.i.d. sample is

$$\frac{\sigma_f^2}{\text{Var}\left[f(\mathbf{X}_0)\right]} = 1 + 2\sum_{i=1}^{\infty} \text{Corr}\left[f(\mathbf{X}_0), f(\mathbf{X}_i)\right] \qquad (6)$$

This is the *integrated autocorrelation time* (ACT) and represents the effective number of dependent samples that is equivalent to a single independent sample. Alternatively $n^* = n/ACT$ may be regarded as the effective equivalent sample size if the elements of the chain had been independent.

To estimate the ACT in practice one might examine the chain from the point at which it is deemed to have converged and estimate the lag-i autocorrelation $\text{Corr}\left[f(\mathbf{X}_0), f(\mathbf{X}_i)\right]$ by

$$\hat{\gamma}_i = \frac{1}{n-i}\sum_{j=1}^{n-i}\left(f(\mathbf{X}_j) - \hat{f}_n\right)\left(f(\mathbf{X}_{j+i}) - \hat{f}_n\right) \qquad (7)$$

Naively, substituting these into (6) gives an estimate of the ACT. However contributions from all terms with very low theoretical autocorrelation *in a real run* are effectively random noise, and the sum of such terms can dominate the deterministic effect in which we are interested (e.g. Geyer, 1992). For this article we employ the simple solution suggested in Carlin and Louis (2009): the sum (6) is truncated from the first lag, l, for which the estimated autocorrelation drops below 0.05 . This gives the (slightly biassed) estimator

$$\mathrm{ACT}_{\mathrm{est}} := 1 + 2\sum_{i=1}^{l-1} \hat{\gamma}_i. \tag{8}$$

Given the potential for relatively large variance in estimates of integrated ACT howsoever they might be obtained (e.g. Sokal, 1997), this simple estimator should be adequate for comparing the relative efficiencies of the different algorithms in this article. Geyer (1992) provides a number of more complex window estimators and provides references for regularity conditions under which they are consistent.

A given run will have a different ACT associated with each parameter. An alternative efficiency measure, which is aggregated over all parameters is provided by the Mean Square Euclidean Jump Distance (MSEJD)

$$S^2_{Euc} := \frac{1}{n-1}\sum_{i=1}^{n-1} \left|\left|\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\right|\right|_2^2.$$

The expectation of this quantity at stationarity is referred to as the Expected Square Euclidean Jump Distance (ESEJD). Consider a single component of the target with variance $\sigma_i^2 := \mathrm{Var}\left[X_i\right] = \mathrm{Var}\left[X_i'\right]$, and note that $\mathbb{E}\left[X_i' - X_i\right] = 0$, so

$$\mathbb{E}\left[(X_i' - X_i)^2\right] = \mathrm{Var}\left[X_i' - X_i\right] = 2\sigma_i^2\left(1 - \mathrm{Corr}\left[X_i, X_i'\right]\right)$$

Thus when the chain is stationary and the posterior variance is finite, maximising the ESEJD is equivalent to minimising a weighted sum of the lag-1 autocorrelations.

If the target has finite second moments and is roughly elliptical in shape with (known) covariance matrix $\mathbf{\Sigma}$ then an alternative measure of efficiency is the Mean Square Jump Distance (MSJD)

$$S_d^2 := \frac{1}{n-1}\sum_{i=1}^{n-1}\left(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\right)^t \mathbf{\Sigma}^{-1}\left(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\right),$$

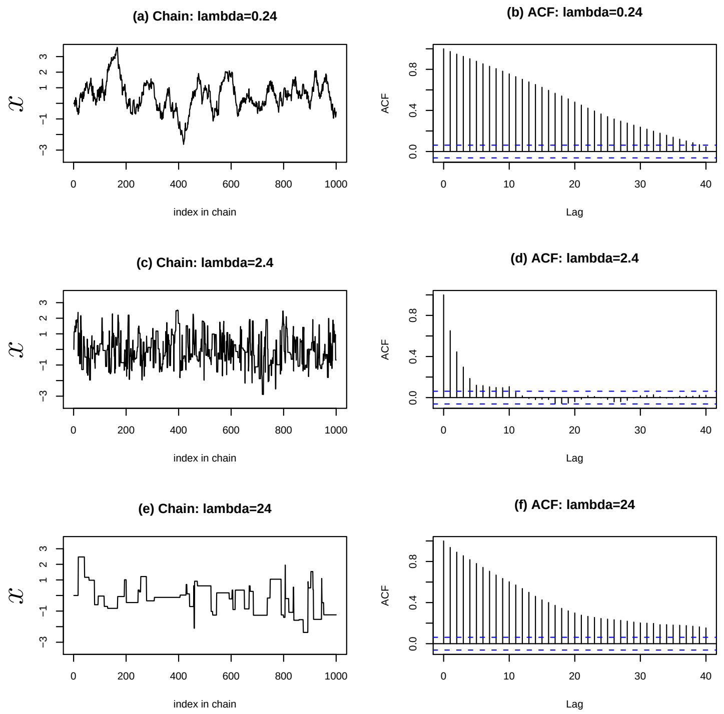

Figure 1: Trace plots ((a), (c), and (e)) and corresponding autocorrelation plots ((b), (d), and (f)), for exploration of a standard Gaussian initialised from $x = 0$ and using the random walk Metropolis algorithm with Gaussian proposal. Proposal scale parameters for the three scenarios are respectively (a) & (b) 0.24, (c) & (d) 2.4, and (e) & (f) 24.

which is proportional to the unweighted sum of the lag-1 autocorrelations over the principal components of the ellipse. The theoretical expectation of the MSJD at stationarity is known as the expected squared jump distance (ESJD).

Figure 1 shows trace plots for three different Markov chains. Estimates of the autocorrelation from lag-0 to lag-40 for each Markov chain appear alongside the corresponding traceplot. The simple window estimator for integrated ACT provides estimates of respectively 39.7, 5.5, and 35.3. The MSEJDs are respectively 0.027, 0.349, and 0.063, and are equal to the MSJDs since the stationary distribution has a variance of 1.

2.2.3 Assessing accuracy

An MCMC algorithm might efficiently explore an unimportant part of the parameter space and never find the main posterior mass. ACT's will be low therefore, but the resulting posterior estimate will be wildly innaccurate. In most practical examples it is not possible to determine the accuracy of the posterior estimate, though consistency between several independent runs or between different portions of the same run can be tested.

For the purposes of this article it was important to have a relatively accurate estimate of the posterior, not determined by a RWM algorithm. Fearnhead and Sherlock (2006) detail a Gibbs sampler for the MMPP; this Gibbs sampler was run for 100 000 iterations on each of the data sets analysed in this article. A "burn-in" of 1000 iterations was allowed for, and a posterior estimate from the last 99 000 iterations was used as a reference for comparison with posterior estimates from RWM runs of 10 000 iterations (after burn in).

2.2.4 Theoretical approaches for algorithm efficiency

To date, theoretical results on the efficiency of RWM algorithms have been obtained through two very different approaches. We wish to quote, explain, and apply theory from both and so we give a heuristic description of each and define associated notation. Both approaches link some measure of efficiency to the expected acceptance rate - the expected proportion of proposals accepted at stationarity.

The first approach was pioneered in Roberts *et al.* (1997) for targets with independent identically distributed components and then generalised in Roberts and Rosenthal (2001) to targets of the form

$$\pi(\mathbf{x}) = \prod_1^d C_i \; f(C_i x_i).$$

The inverse scale parameters, C_i, are assumed to be drawn from some distribution with a given (finite) mean and variance. A single component of the d dimensional chain (without loss of generality the first) is then examined; at iteration i of the algorithm it is denoted $X_{1,i}^{(d)}$. A scaleless, speeded up, continuous time process which mimics the first component of the chain is defined as

$$W_t^{(d)} := C_1 X_{1,[td]}^{(d)},$$

where $[u]$ denotes the nearest integer less than or equal to u. Finally, proposed jumps are assumed to be Gaussian

$$\mathbf{Y}^{(d)} \sim N\left(\mathbf{0}, \lambda_d^2 \mathbf{I}\right).$$

Subject to conditions on the first two deriviatives of $f(\cdot)$, Roberts and Rosenthal (2001) show that if $\mathbb{E}[C_i] = 1$ and $\mathbb{E}[C_i^2] = b$, and provided $\lambda_d = \mu/d^{1/2}$ for some fixed μ (the scale parameter but "rescaled" according to dimension) then as $d \rightarrow \infty$, $W_t^{(d)}$ approaches a Langevin diffusion process with speed

$$h(\mu) = \frac{C_1^2 \mu^2}{b}\,\overline{\alpha}_d \quad \text{where} \quad \overline{\alpha}_d := 2\Phi\left(-\frac{1}{2}\mu I^{1/2}\right). \tag{9}$$

Here $\Phi(x)$ is the cumulative distribution function of a standard Gaussian, $I := \mathbb{E}\left[((\log f)')^2\right]$ is a measure of the roughness of the target, and $\overline{\alpha}_d$ corresponds to the acceptance rate.

Bédard (2007) proves a similar result for a triangular sequence of inverse scale parameters $c_{i,d}$, which are assumed to be known. A necessary and sufficient condition equivalent to (11) below is attached to this result. In effect this requires the scale over which the smallest component varies to be "not too much smaller" than the scales of the other components.

The second technique (e.g. Sherlock and Roberts, 2009) uses expected square jump distance (ESJD) as a measure of efficiency. Exact analytical forms for ESJD (denoted S_d^2) and expected acceptance rate are derived for any unimodal elliptically symmetric target and any proposal density. Many standard sequences of d-dimensional targets ($d = 1, 2, \ldots$), such as the Gaussian, satisfy the condition that as $d \rightarrow \infty$ the probability mass becomes concentrated in a spherical shell which itself becomes infinitesimally thin relative to its radius. Thus the random walk on a rescaling of the target is, in the limit, effectively confined to the surface of this shell. Sherlock and Roberts (2009) show that if the sequence of targets satisfies such a "shell" condition, and a slightly stronger condition is satisfied by the sequence of proposals then as $d \to \infty$

$$\frac{d}{k_x^{(d)^2}} S_d^2(\mu) \to \mu^2\ \overline{\alpha}_d \quad \text{with} \quad \overline{\alpha}_d(\mu) := 2\Phi\left(-\frac{1}{2}\mu\right). \tag{10}$$

Here $\overline{\alpha}_d$ is the limiting expected acceptance rate, $\mu := d^{1/2}\lambda_d k_y^{(d)}/k_x^{(d)}$, and $k_x^{(d)}$ and $k_y^{(d)}$ are the rescalings appropriate for the target and proposal sequences so that the spherical shells to which the mass converges both have radius 1. For target and proposal distributions with independent components, such as are used in the diffusion results, $k_x^{(d)} = k_y^{(d)} = d^{1/2}$, and hence (consistently) $\mu = d^{1/2}\lambda_d$.

A further condition is required on the triangular sequence of inverse scale parameters of the axes of the elliptical target

$$\frac{\max_i c_{i,d}^2}{\sum_{i=1}^d c_{i,d}^2} \to 0 \quad \text{as} \quad d \to \infty \tag{11}$$

Theoretical results from the two techniques are remarkably similar and as will be seen, lead to identical strategies for optimising algorithm efficiency. It is worth noting however that results from the first approach apply only to targets with independent components and results from the second only to targets which are unimodal and elliptically symmetric. That they lead to identical strategies indicates a certain potential robustness of these strategies to the shape of the target. This potential, as we shall see, is born out in practice.

2.3 The Markov Modulated Poisson Process

Let X_t be a continuous time Markov chain on discrete state space $\{1, \ldots, d\}$ and let $\boldsymbol{\psi} := [\psi_1, \ldots, \psi_d]$ be a d-dimensional vector of (non-negative) intensities. The linked but stochastically independent Poisson process Y_t whose intensity is ψ_{X_t} is a Markov modulated Poisson

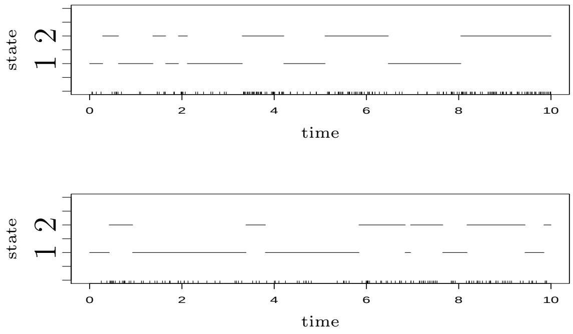

Figure 2: Two 2-state continuous time Markov chains simulated from generator $\mathbf{Q}$ with $q_{12} = q_{21} = 1$; the rug plots show events from an MMPP simulated from these chains, with intensity vectors $\boldsymbol{\psi} = (10, 30)$ (upper graph) and $\boldsymbol{\psi} = (10, 17)$ (lower graph).

process - it is a Poisson process whose intensity is modulated by a continuous time Markov chain.

The idea is best illustrated through two examples, which also serve to introduce the notation and data sets that will be used throughout this article. Consider a two-dimensional Markov chain X_t with generator $\mathbf{Q}$ with $q_{12} = q_{21} = 1$. Figure 2 shows realisations from two such chains over a period of 10 seconds. Now consider a Poisson process Y_t which has intensity 10 when X_t is in state 1 and intensity 30 when X_t is in state 2. This is an MMPP with event intensity vector $\boldsymbol{\psi} = [10, 30]^t$. A realisation (obtained via the realisation of X_t) is shown as a rug plot underneath the chain in the upper graph. The lower graph shows a realisation from an MMPP with event intensities $[10, 17]^t$.

It can be shown (e.g. Fearnhead and Sherlock, 2006) that the likelihood for data from an MMPP which starts from a distribution $\boldsymbol{\nu}$ over its states is

$$L(\mathbf{Q}, \boldsymbol{\Psi}, \mathbf{t}) = \boldsymbol{\nu}' e^{(\mathbf{Q}-\boldsymbol{\Psi})t_1} \boldsymbol{\Psi} \dots e^{(\mathbf{Q}-\boldsymbol{\Psi})t_n} \boldsymbol{\Psi} e^{(\mathbf{Q}-\boldsymbol{\Psi})t_{n+1}} \mathbf{1}. \tag{12}$$

Here $\boldsymbol{\Psi} := \mathrm{diag}(\boldsymbol{\psi})$, $\mathbf{1}$ is a vector of 1's, n is the number of observed events, t_1 is the time from the start of the observation window until the first event, t_{n+1} is the time from the last event until the end of observation window, and t_i $(2 \leq i \leq n)$ is the time between the $i-1^{th}$ and i^{th} events. In the absence of further information, the initial distribution $\boldsymbol{\nu}$ is often taken to be the stationary distribution of the underlying Markov chain.

The likelihood of an MMPP is invariant to a relabelling of the states. Hence if the prior is similarly invariant then so too is the posterior: if the posterior for a two dimensional MMPP has a mode at $(\psi_1, \psi_2, q_{12}, q_{21})$ then it has an identical mode at $(\psi_2, \psi_1, q_{21}, q_{12})$. In this article our overriding interest is in the efficiency of the MCMC algorithms rather than the exact meaning of the parameters and so we choose the simplest solution to this identifiablity problem: the state with the lower Poisson intensity ψ is always referred to as State 1.

2.3.1 MMPP data in this article

The two data sets of event times used in this article arose from two independent MMPP's simulated over an observation window of 100 seconds. Both underlying Markov chains have $q_{12} = q_{21} = 1$; data set D1 has event intensity vector $\boldsymbol{\psi} = [10, 30]$ whereas data set D2 has $\boldsymbol{\psi} = [10, 17]$.

As might be expected the overall intensity of events in D2 is lower than in D1. Moreover because the difference in intensity between the states is so much larger in D1 than in D2 it is also easier with D1 than D2 to distinguish the state of the underlying Markov chain, and thus the values of the Markov and Poisson parameters. Further, in the limit of the underlying chain being known precisely, for example as $\psi_2 \rightarrow \infty$ with ψ_1 finite, and provided the priors are independent, the posteriors for the Poisson intensity parameters ψ_1 and ψ_2 are completely independent of each other and of the Markov parameters q_{12} and q_{21}. Dependence between the Markov parameters is also small, being $O(1/T)$ (e.g. Fearnhead and Sherlock, 2006).

In Section 4, differences between D1 and D2 will be related directly to observed differences in efficiency of the various RWM algorithms between the two data sets.

3 Implementations of the RWM: theory and practice

This section describes several theoretical results for the RWM or for MCMC in general. Intuitive explanation of the principle behind each result is emphasised and the manner in which it informs the RWM implementation is made clear. Each algorithm was run three times on each of the two data sets.

3.1 Optimal scaling of the RWM

Intuition: Consider the behaviour of the RWM as a function of the overall scale parameter of the proposed jump, λ, in (1). If most proposed jumps are small compared with some measure of the scale of variability of the target distribution then, although these jumps will often be accepted, the chain will move slowly and exploration of the target distribution will be relatively inefficient. If the jumps proposed are relatively large compared with the target distribution's scale, then many will not be accepted, the chain will rarely move and will again explore the target distribution inefficiently. This suggests that given a particular target and form for the jump proposal distribution, there may exist a finite scale parameter for the proposal with which the algorithm will explore the target as efficiently as possible. These ideas are clearly demonstrated in Figure 1 which shows traceplots for a one dimensional Gaussian target explored using a Gaussian proposal with scale parameter an order of magnitude smaller (a) and larger (c) than is optimal, and (b) with a close to optimal scale parameter.

Theory: Equation (9) gives algorithm efficiency for a target with independent and identical (up to a scaling) components as a function of the "rescaled" scale parameter $\mu = d^{1/2}\lambda_d$ of a Gaussian proposal. Equation (10) gives algorithm efficiency for a unimodal elliptically symmetric target explored by a spherically symmetric proposal with $\mu = d^{1/2}\lambda_d k_y^{(d)}/k_x^{(d)}$. Efficiencies are therefore optimal at $\mu \approx 2.38/I^{1/2}$ and $\mu \approx 2.38$ respectively. These correspond to actual scale parameters of respectively

$$\lambda_d = \frac{2.38}{I^{1/2}d^{1/2}} \quad \text{and} \quad \lambda_d = \frac{2.38\ k_x^{(d)}}{d^{1/2}k_y^{(d)}}.$$

The equivalence between these two expressions for Gaussian data explored with a Gaussian target is clear from Section 2.2.4. However the equations offer little direct help in choosing a scale parameter for a target is neither elliptical, nor possesses components which are i.i.d. up to a scale parameter. Substitution of each expression into the corresponding acceptance rate equation, however, leads to the same optimal acceptance rate, $\hat{\alpha} \approx 0.234$. This justifies the relatively well known adage that *for random walk algorithms with a large number of parameters, the scale parameter of the proposal should be chosen so that the acceptance rate is approximately* 0.234. On a graph of asymptotic efficiency against acceptance rate (e.g. Roberts and Rosenthal, 2001), the curvature near the mode is slight, especially to its right, so that an acceptance rate of anywhere between 0.2 and 0.3 should lead to an algorithm of close to optimally efficiency.

In practice updates are performed on a finite number of parameters; for example a two dimensional MMPP has four parameters $(\psi_1, \psi_2, q_{12}, q_{21})$. A block update involves all of these, whilst each update of a simple Metropolis within Gibbs step involves just one parameter. In finite dimensions the optimal acceptance rate can in fact take any value between 0 and 1. Sherlock and Roberts (2009) provide analytical formulae for calculating the ESJD and the expected acceptance rate for any proposal and any elliptically symmetric unimodal target. In one dimension, for example, the optimal acceptance rate for a Gaussian target explored by a Gaussian proposal is 0.44, whilst the optimum for a double exponential target ($\pi(x) \propto e^{-|x|}$) explored with a double exponential proposal is exactly $\hat{\alpha} = 1/3$. Sherlock (2006) considers several simple examples of spherically symmetric proposal and target across a range of dimensions and finds that in all cases curvature at the optimal acceptance rate is small, so that a range of acceptance rates is nearly optimal. Further, the optimal acceptance rate is itself between 0.2 and 0.3 for $d \geq 6$ in all the cases considered.

Sherlock and Roberts (2009) also weaken the "shell" condition of Section 2.2.4 and consider sequences of spherically symmetric targets for which the (rescaled) radius converges to some random variable R rather than a point mass at 1. It is shown that, provided the sequence of proposals still satisfies the shell condition, the limiting optimal acceptance rate is strictly less than 0.234. Acceptance rate tuning should thus be seen as only a guide, though a guide which has been found to be robust in practice.

Algorithm 1 (Blk): The first algorithm (Blk) used to explore data sets D1 and D2 is a four dimensional block updating RWM with proposal $\mathbf{Y} \sim N(0, \lambda^2 \mathbf{I})$ and λ tuned so that the acceptance rate is approximately 0.3.

3.2 Optimal scaling of the RWM within Gibbs

Intuition: Consider first a target either spherically symmetric, or with i.i.d. components, and let the overall scale of variability of the target be η. For full block proposals the optimal scale parameter should be $O\left(\eta/d^{1/2}\right)$ so that the square of the magnitude of the total proposal is $O(\eta^2)$. If a Metropolis within Gibbs update is to be used with k sub-blocks and $d_* = d/k$ of the components updated at each stage then the optimal scale parameter should be larger, $O\left(\eta/d_*^{1/2}\right)$. However only one of the k stages of the RWM within Gibbs algorithm updates any given component whereas with k repeats of a block RWM that component is updated k times. Considering the squared jump distances it is easy to see that, given the additivity of square jump distances, the larger size of the RWM within Gibbs updates is exactly canceled by their lower frequency, and so (in the limit) there is no difference in efficiency when compared with a block update. The same intuition applies when comparing a random scan Metropolis within Gibbs scheme with a single block update.

Now consider a target for which different components vary on different scales. If sub-blocks are chosen so as to group together components with similar scales then a Metropolis within Gibbs scheme can apply suitable scale paramaters to each block whereas a single block update must choose one scale parameter that is adequate for all components. In this scenario, Metropolis within Gibbs updates should therefore be more efficient.

Theory: Neal and Roberts (2006) consider a random scan RWM within Gibbs algorithm on a target distribution with i.i.d. components and using i.i.d. Gaussian proposals all having the same scale parameter $\lambda_d = \mu/d^{1/2}$. At each iteration a subset (of size dc_d) of the components is chosen at random and updated as a single block. It is shown (again subject to differentiability conditions on $f(\cdot)$) that the process $W_t^{(d)} := X_{1,[td]}^{(d)}$ approaches a Langevin diffusion with speed

$$h_c(\mu) = 2c\mu^2\Phi\left(-\frac{1}{2}\mu(cI)^{1/2}\right).$$

The optimal scaling is therefore larger than for a standard block update (by a factor of $c^{-1/2}$) but the optimal speed and the optimal acceptance rate (0.234) are identical to those found by Roberts *et al.* (1997).

Sherlock (2006) considers sequential Metropolis within Gibbs updates on a unimodal elliptically symmetric target, using spherical proposal distributions but allowing *different* scale parameters for the proposals in each sub-block. The k sub-blocks are assumed to correspond to disjoint subsets of the principal axes of the ellipse and updates for each are assumed to be optimally tuned. Efficiency is considered in terms of ESEJD and is again found to be optimal (as $d \to \infty$) when the acceptance rate for each sub-block is 0.234. For equal sized sub-blocks, the relative efficiency of the Metropolis within Gibbs scheme compared to

k optimally scaled single block updates is shown to be

$$r = \frac{\frac{1}{k}\sum \overline{c^2}_i}{\left(\frac{1}{k}\sum \frac{1}{\overline{c^2}_i}\right)^{-1}} \,, \tag{13}$$

where $\overline{c^2}_i$ is the mean of the squares of the inverse scale parameters for the i^{th} block. Since r is the ratio of an arithmetic mean to a harmonic mean, it is greater than or equal to one and thus the Metropolis within Gibbs step is always at least as efficient as the block Metropolis. However the more similar the blocks, the less the potential gain in efficiency.

In practice, parameter blocks do not generally correspond to disjoint subsets of the principal axes of the posterior or, in terms of single parameter updates, the parameters are not generally orthogonal. Equation 13 therefore corresponds a limiting maximum efficiency gain, obtainable only when the parameter sub-blocks are orthogonal.

Algorithm 2 (MwG): Our second algorithm (MwG) is a sequential Metropolis within Gibbs algorithm with proposed jumps $Y_i \sim N(0, \lambda_i^2)$. Each scale parameter is tuned seperately to give an acceptance rate of between 0.4 and 0.45 (approximately the optimum for a one-dimensional Gaussian target and proposal).

3.3 Tailoring the shape of a block proposal

Intuition: First consider a general target with roughly elliptical contours and covariance matrix $\mathbf{\Sigma}$, such as that shown in Figure 3. For simplicity we visualise a two parameter posterior but the following argument clearly generalises to any number of dimensions. It seems intuitively sensible that a “tailored” block proposal distribution with the same shape and orientation as the target will tend to produce larger jumps along the target’s major axis and smaller jumps along its minor axis and should therefore allow for more efficient exploration of the target.

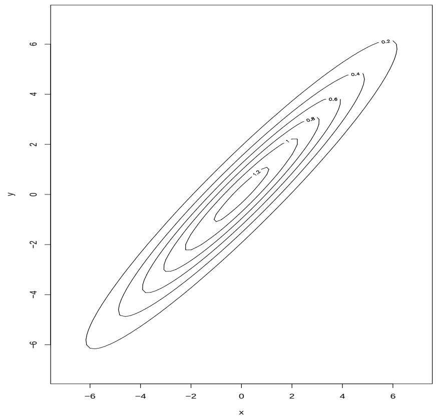

Figure 3: Contour plot for a two dimensional Gaussian density with $\sigma_1^2 = \sigma_2^2 = 1$ and correlation $\rho = 0.95$.

Theory: Sherlock (2006) considers exploration of a unimodal elliptically symmetric target with either a spherically symmetric proposal or a tailored elliptically symmetric proposal in the limit as $d \to \infty$. Subject to condition (11) (and a "shell"-like condition similar to that mentioned in Section 2.2.4), it is shown that with each proposal shape it is in fact possible to achieve the same optimal expected square jump distance. However if a spherically symmetric proposal is used on an elliptical target, some components are explored better than others and in some sense the overall efficiency is reduced. This becomes clear on considering the ratio r, of the expected squared Euclidean jump distance for an optimal spherically symmetric proposal to that of an optimal tailored proposal. Sherlock (2006) shows that for a sequence of targets, where the target with dimension d has elliptical axes with inverse scale parameters $c_{d,1}, \ldots, c_{d,d}$, the limiting ratio is

$$r = \frac{\lim_{d\to\infty} \left(\frac{1}{d}\sum_{i=1}^{d} c_{d,i}^{-2}\right)^{-1}}{\lim_{d\to\infty} \frac{1}{d}\sum_{i=1}^{d} c_{d,i}^{2}}.$$

The numerator is the limiting harmonic mean of the squared inverse scale parameters, which is less than or equal to their arithmetic mean (the denominator), with equality if and only if (for a given d) all the $c_{d,i}$ are equal. Roberts and Rosenthal (2001) examine similar relative efficiencies but for targets and proposals with independent components with inverse scale parameters C sampled from some distribution. In this case the derived measure of relative efficiency is the relative speeds of the diffusion limits for the first component of the target

$$r^* = \frac{\mathbb{E}\left[C\right]^2}{\mathbb{E}\left[C^2\right]}.$$

This is again less than or equal to one, with equality when all the scale parameters are equal. Hence efficiency is indeed directly related to the relative compatibility between target and proposal shapes.

Furthermore Bédard (2008) shows that if a proposal has i.i.d. components yet the target (assumed to have independent components) is wildly asymmetric, as measured by (11), then the limiting optimal acceptance rate can be anywhere between 0 and 1. However even at this optimum, some components will be explored infinitely more slowly than others.

In practice the shape $\mathbf{\Sigma}$ of the posterior is not known and must be estimated, for example by numerically finding the posterior mode and the Hessian matrix $\mathbf{H}$ at the mode, and setting $\mathbf{\Sigma} = \mathbf{H}^{-1}$. We employ a simple alternative which uses an earlier MCMC run.

Algorithm 3 (BlkShp): Our third algorithm first uses an optimally scaled block RWM algorithm (Algorithm 1), which is run for long enough to obtain a "reasonable" estimate of the covariance from the posterior sample. A fresh run is then started and tuned to give an acceptance rate of about 0.3 but using proposals

$$\mathbf{Y} \sim N(\mathbf{0}, \lambda^2 \hat{\mathbf{\Sigma}}).$$

For each data set, so that our implementation would reflect likely statistical practice, each of the three replicates of this algorithm estimated the $\mathbf{\Sigma}$ matrix from iterations 1000-2000 of the corresponding replicate of Algorithm 1 (i.e. using 1000 iterations after "burn in"). In all therefore, six different variance matrices were used.

3.4 Improving tail exploration

Intuition: A posterior with relatively heavy polynomial tails such as the one-dimensional Cauchy distribution has considerable mass some distance from the origin. Proposal scalings which efficiently explore the body of the posterior are thus too small to explore much of the tail mass in a "reasonable" number of iterations. Further, polynomial tails become flatter with distance from the origin so that for unit vector $\mathbf{u}$, $\pi(\mathbf{x} + \lambda\mathbf{u})/\pi(\mathbf{x}) \to 1$ as $||\mathbf{x}||_2 \to \infty$. Hence the acceptance rate for a random walk algorithm approaches 1 in the tails, whatever the direction of the proposed jump. The algorithm therefore loses almost all sense of the direction to the posterior mass.

Theory: Roberts (2003) brings together literature relating the tails of the posterior and the proposal to the ergodicity of the Markov chain and hence its convergence properties. Three important cases are noted

1. If $\pi(\mathbf{x}) \propto e^{-s||\mathbf{x}||_2}$, at least outside some compact set, then the random walk algorithm is geometrically ergodic.
2. If the tails of the proposal are bounded by some multiple of $||x||_2^{-(r+d)}$ and if $\pi(\mathbf{x}) \propto ||\mathbf{x}||_2^{-(r+d)}$, at least outside some compact set, then the algorithm is polynomially ergodic with rate $r/2$.
3. If $\pi(\mathbf{x}) \propto ||\mathbf{x}||_2^{-(r+d)}$, at least for large enough $\mathbf{x}$, and the proposal has tails $q(\mathbf{x}) \propto ||\mathbf{x}||_2^{-(d+\eta)}$ $(0 < \eta < 2)$ then the algorithm is polynomially ergodic with rate r/η.

Thus posterior distributions with exponential or lighter tails lead to a geometrically ergodic Markov chain, whereas polynomially tailed posteriors can lead to polynomially ergodic chains, and even this is only guaranteed if the tails of the proposal are at least as heavy as the tails of the posterior. However by using a proposal with tails so heavy that it has infinite variance, the polynomial convergence rate can be made as large as is desired.

Algorithm 4 (BlkShpCau): Our fourth algorithm is identical to BlkShp but samples the proposed jump from the heavy tailed multivariate Cauchy. Proposals are generated by simulating $\mathbf{V} \sim N(\mathbf{0}, \hat{\mathbf{\Sigma}})$ and $Z \sim N(0,1)$ and setting $\mathbf{Y}^* = \mathbf{V}/Z$. No acceptance rate criteria exist for proposals with infinite variance and so the optimal scaling parameter for this algorithm was found (for each dataset and $\hat{\mathbf{\Sigma}}$) by repeating several small runs with different scale parameters and noting which produced the best ACT's for each data set.

Algorithm 5 (BlkShpMul): The fifth algorithm relies on the fact that taking logarithms of parameters shifts mass from the tails to the centre of the distribution. It uses a random walk on the posterior of $\tilde{\theta} := (\log \psi_1, \log \psi_2, \log q_{12}, \log q_{21})$. Shape matrices $\hat{\mathbf{\Sigma}}$ were estimated as for Algorithm 3, but using the logarithms of the posterior output from Algorithm 1. In the original parameter space this algorithm is equivalent to a proposal with components $X_i^* = X_i \; e^{Y_i^*}$ and so has been called the **multiplicative random walk** (see for example Dellaportas and Roberts, 2003). In the original parameter space the acceptance probability is

$$\alpha(\mathbf{x}, \mathbf{x}^*) = \min\left(1, \frac{\prod_1^d x_i^*}{\prod_1^d x_i} \frac{\pi(\mathbf{x}^*)}{\pi(\mathbf{x})}\right).$$

Since the algorithm is simply an additive random walk on the log parameter space, the usual acceptance rate optimality criteria apply.

A logarithmic transformation is clearly only appropriate for positive parameters and can in fact lead to a heavy *left* hand tail if a parameter (in the original space) has too much mass close to zero. The transformation $\tilde{\theta}_i = \text{sign}(\theta_i) \; \log(1 + |\theta_i|)$ circumvents both of these problems.

3.5 Combining algorithms

Intuition: Different MCMC algorithms may have different strengths and weaknesses. For example algorithm $A^{(1)}$ may efficiently explore the tails of a distribution whereas algorithm

$A^{(2)}$ might efficiently explore the body. In such circumstances a hybrid algorithm which alternates iterations from $A^{(1)}$ and $A^{(2)}$ should combine the strengths of both, with efficiency in a given portion of the posterior no worse than half that of the more efficient algorithm. A similar argument applies when two algorithms are each efficient at exploring a different type of posterior (e.g. relatively heavy tailed and relatively light tailed). In this case alternating iterations from the algorithms produces a hybrid algorithm which is robust to the type of posterior.

Theory: Consider the inner product

$$< \nu_1, \nu_2 >:= \int d\mathbf{x}\ \frac{\nu_1(\mathbf{x})\nu_2(\mathbf{x})}{\pi(\mathbf{x})}, \tag{14}$$

and the associated L^2 norm, $||\nu||_2 :=< \nu, \nu >^{1/2}$. To avoid technical detail, we restrict attention to distributions $\nu(\cdot)$ which are absolutely continuous with respect to $\pi(\cdot)$ and for which the L^2 norm with respect to (14) exists: $\mathbb{E}_\pi\left[|d\nu/d\pi|^2\right] < \infty$. We also assume that each transition kernel ($A, A^{(1)}, A^{(2)}$, and A^*) has a discrete spectrum; a more general theory exists and can be found used in the context of MCMC in Roberts and Rosenthal (1997), for instance.

Within the simplified framework described above, it is shown in Appendix A that from initial distribution ν, for any reversible MCMC kernel A with stationary distribution $\pi(\cdot)$ and second largest eigenvalue β_2,

$$\left|\left|\nu A^k - \pi\right|\right|_2 \leq \beta_2^k \left|\left|\nu - \pi\right|\right|_2 .$$

Since $\left|\left|\nu A^k - \pi\right|\right|_1 \leq \left|\left|\nu A^k - \pi\right|\right|_2$ this demonstrates geometric ergodicity as defined in Section 2.2.1.

Next consider two MCMC algorithms $A^{(1)}$ and $A^{(2)}$ with stationary distribution $\pi(\cdot)$ and second largest eigenvalues $\beta_2^{(1)}$ and $\beta_2^{(2)}$. Let A^* be a combination algorithm which alternates iterations from $A^{(1)}$ and $A^{(2)}$. Of course A^* is not, in general, reversible; nonetheless it can also be shown (see Appendix A) that

$$\left|\left|\nu\left(A^{*}\right)^{k}-\pi\right|\right|_{2}\leq\left(\beta_{2}^{(1)}\beta_{2}^{(2)}\right)^{k}||\nu-\pi||_{2}\,.$$

Thus the bound on geometric convergence rate for A^{*} is at worst the geometric mean of the bounds on the convergence rates of its two component algorithms. The result generalises to the sequential combination of any n algorithms.

Instead of alternating $A^{(1)}$ and $A^{(2)}$, at each iteration one of the two algorithms could be chosen at random with probabilities $1-\delta$ and δ. Combining the triangle inequality with the first result in this section, for this mixture kernel A^{**}

$$||\nu A^{**}-\pi||_{2}=\left|\left|(1-\delta)\left(\nu A^{(1)}\right)-\pi\right)+\delta\left(\nu A^{(1)}\right)-\pi\right)\right|\right|_{2}\leq((1-\delta)\beta_{1}+\delta\beta_{2})\,||\nu-\pi||_{2}\,. \quad (15)$$

The geometric convergence rate for this (reversible) kernel, A^{**} is clearly at most $(1-\delta)\beta_1+\delta\beta_2$. Practical implementation of such a mixture kernel is illustrated in the next section in the context of adaptive MCMC.

3.6 Adaptive MCMC

Intuition: Algorithm 3 used the output from a previous MCMC run to estimate the shape Matrix $\mathbf{\Sigma}$. An overall scaling parameter was then varied to give an acceptance rate of around 0.3. With adaptive MCMC a single chain is run, and this chain gradually alters its own proposal distribution (e.g. changing $\mathbf{\Sigma}$), by learning about the posterior *from its own output*. This simple idea has a major potential pitfall, however.

If the algorithm is started away from the main posterior mass, for example in a tail or a minor mode, then it initially learns about that region. It therefore alters the proposal so that it efficiently explores this region of minor importance. Worse, in so altering the proposal the algorithm may become even less efficient at finding the main posterior mass, remain in an unimportant region for longer and become even more influenced by that unimportant region.

The acceptance rate for each proposal is chosen so that its stationary distribution is $\pi(\cdot)$. However since the transition kernel is continually changing, potentially with the positive feeback mechanism of the previous paragraph, this is not sufficient to guarantee that the overall stationary distribution of the chain is $\pi(\cdot)$. Roberts and Rosenthal (2007) give a very simple adaptive MCMC scheme on a discrete state space for which the resulting stationary distribution is not the intended target.

A simple solution to this stationarity problem is so called *finite adaptation* wherein the algorithm is only allowed to evolve for the first n_0 iterations, after which time the transition kernel is fixed. Such a scheme is equivalent to running a shorter "tuning" chain and then a longer subsequent chain (e.g. Algorithm 3). If the tuning portion of the chain has only explored a minor mode or a tail this still leads to an inefficient algorithm. We would prefer to allow the chain to eventually correct for any errors made at early iterations and yet still lead to the intended stationary distribution. It seems sensible that this might be achieved provided changes to the kernel become smaller and smaller as the algorithm proceeds and provided the above-mentioned positive feedback mechanism can never pervert the entire algorithm.

Theory: At the n^{th} iteration let Γ_n represent the choice of transition kernel; for the RWM it might represent the current shape matrix $\mathbf{\Sigma}$ and the overall scaling λ. Denote the corresponding transition kernel $P_{\Gamma_n}(\mathbf{x},\cdot)$. Roberts and Rosenthal (2007) derive a set of two conditions which together guarantee convergence to the stationary distribution. A key concept is that of *diminishing adaptation*, wherein changes to the kernel must become vanishingly small as $n \rightarrow \infty$

$$\sup_{\mathbf{x}} \left|\left|P_{\Gamma_{n+1}}(\mathbf{x},\cdot) - P_{\Gamma_n}(\mathbf{x},\cdot)\right|\right|_1 \xrightarrow{p} 0 \quad \text{as } n \rightarrow \infty.$$

A second *containment* condition considers the ϵ-convergence time under repeated application of a fixed kernel, γ, and starting point $\mathbf{x}$,

$$M_\epsilon(\mathbf{x},\gamma) := \inf_n \left\{n \geq 1 : \left|\left|P_\gamma^n(\mathbf{x},\cdot) - \pi(\cdot)\right|\right|_1 \leq \epsilon\right\},$$

and requires that for all $\delta > 0$ there is an N such that for all n

$$\mathbb{P}\left(M_\epsilon(\mathbf{X}_n, \Gamma_n) \leq N \mid \mathbf{X}_0 = \mathbf{x}_0, \Gamma_0 = \gamma_0\right) \geq 1 - \delta.$$

The containment condition is, in general, difficult to check in practice; some criteria are provided in Bai *et al.* (2009).

Adaptive MCMC is a highly active research area at present, and so when considering specific schemes, we confine ourselves to adaptations relating to posterior shape and scaling. Roberts and Rosenthal (2009) describe an adaptive RWM algorithm for which the proposal at the n^{th} iteration is sampled from a mixture of adaptive and non-adaptive distributions

$$\mathbf{Y} \sim \begin{cases} N\left(\mathbf{0}, \frac{1}{d}2.38^2\mathbf{\Sigma}_n\right) & w.p. \quad 1-\delta \\ N\left(\mathbf{0}, \frac{1}{100d}\mathbf{I}\right) & w.p. \qquad \delta. \end{cases}$$

Here $\delta = 0.05$ and $\mathbf{\Sigma}_n$ is the variance matrix calculated from the previous $n-1$ iterations of the scheme. Changes to the variance matrix are $O(1/n)$ at the n^{th} iteration and so the algorithm satisfies the diminishing adaptation condition. Haario *et al.* (2001) show that a similar adaptive scheme with $Y \sim N\left(\mathbf{0}, \frac{1}{d}2.38^2\mathbf{\Sigma}_n + \epsilon^2\mathbf{I}\right)$ (for fixed $\epsilon > 0$) is ergodic provided both the target density and its support are bounded.

Choice of the overall scaling factor $2.38^2/d$ follows directly from the optimal scaling limit results reviewed in Section 3.1, with $I = 1$ or $k_x^{(d)} = k_y^{(d)}$. In general therefore a different scaling might be appropriate.

Algorithm 6 (BlkAdpMul): Our adaptive MCMC algorithm is a block multiplicative random walk which samples jump proposals on the log-posterior from the mixture

$$\mathbf{Y} \sim \begin{cases} N\left(\mathbf{0}, m^2\mathbf{\Sigma}_n\right) & w.p. \quad 1-\delta \\ N\left(\mathbf{0}, \frac{1}{d}\lambda_0^2\mathbf{I}\right) & w.p. \qquad \delta. \end{cases}$$

Here $\delta = 0.05$, $d = 4$, and $\mathbf{\Sigma}_n$ is estimated from the logarithms of the posterior sample to date. A few minutes were spent tuning the block multiplicative random walk with proposal variance $\frac{1}{4}\lambda_0^2\mathbf{I}$ to give at least a reasonable value for λ_0 (acceptance rate ≈ 0.3), although this is not stricly necessary.

The overall scaling factor for the adaptive part of the kernel was allowed to vary according to the following scheme.

1. An initial scaling was set to $m_0 = 2.38/d^{1/2}$ and an adaptation quantity $\Delta = m_0/100$ was defined.
2. Proposals from the adaptive part of the mixture were only allowed once there had been at least 10 proposed jumps accepted.
3. If iteration i was from the adaptive part of the kernel then m was altered:
    - If the proposal was rejected then $m \leftarrow m - \Delta/i^{1/2}$.
    - If the proposal was accepted then $m \leftarrow m + 2.3\ \Delta/i^{1/2}$.

Step 2 ensures a sufficient number of different parameter values to calculate a sensible covariance matrix (note that with three or fewer acceptances, rows of the covariance matrix are not even linearly independent). Step 3 leads to an equilibrium acceptance rate of 1/3.3. Changes to m are scaled by $i^{1/2}$ since they must be large enough to adapt to changes in the covariance matrix yet small enough that an equilibrium value is established relatively quickly. As with the variance matrix, such a value would then only change noticeably if there were consistent evidence that it should.

3.7 Utilising problem specific knowledge

Intuition: All of the above techniques apply to RWM algorithms on any posterior. However algorithms are always applied to specific data sets with specific forms for the likelihood and prior. Combining problem specific knowledge with techniques such as optimal scaling and shape adjustmet can often markedly improve efficiency. In the case of the MMPP we define a reparameterisation based on the intuition that for an MMPP with $\psi_1 \approx \psi_2$ the data contain a great deal of information about the average intensity but relatively little information about the difference between the intensities. With this reparameterisation the posterior for data set D2 may then be very efficiently sampled using a Metropolis within Gibbs algorithm.

Theory: For a 2 dimensional MMPP define an overall transition intensity, stationary distribution, mean intensity at stationarity, and a measure of the difference between the two event intensities as follows

$$q := q_{12} + q_{21} \quad , \quad \boldsymbol{\nu} := \frac{1}{q}\left[q_{21}, q_{12}\right]^t \quad , \quad \overline{\psi} := \boldsymbol{\nu}^t \boldsymbol{\psi} \quad \text{and} \quad \delta := \frac{(\psi_2 - \psi_1)}{\overline{\psi}}. \tag{16}$$

Let t_{obs} be the total observation time. If the Poisson event intensities are similar, δ is small, and Taylor expansion of the log-likelihood in δ (see Appendix B) gives

$$l(\overline{\psi}, q, \delta, \nu_1) = n \log \overline{\psi} - \overline{\psi} t_{obs} + 2\delta^2 \nu_1 \nu_2 f(\overline{\psi}\mathbf{t}, q\mathbf{t}) + \delta^3 \nu_1 \nu_2 (\nu_2 - \nu_1) g(\overline{\psi}\mathbf{t}, q\mathbf{t}) + O(\delta^4) \tag{17}$$

for some $f(\cdot,\cdot)$ and $g(\cdot,\cdot)$. Consider a reparameterisation from $(\psi_1, \psi_2, q_{12}, q_{21})$ to $(\overline{\psi}, q, \alpha, \beta)$ with

$$\alpha := 2\delta \left(\nu_1 \nu_2\right)^{1/2} \quad \text{and} \quad \beta := \delta(\nu_2 - \nu_1). \tag{18}$$

Parameters $\overline{\psi}$; q and α; and β (in this order) capture decreasing amounts of variation in the log-likelihood and so, conversely, it might be anticipated that there be corresponding decreasing amounts of information about these parameters contained in the likelihood. Hence very different scalings might be required for each.

Algorithm 7 (MwGRep): A Metropolis within Gibbs update scheme was applied to the reparameterisation $(\overline{\psi}, q, \alpha, \beta)$. A multiplicative random walk was used for each of the first 3 parameters (since they are positive) and an additive update was used for β. Scalings for each of the four parameters were chosen to give acceptance rates of between 0.4 and 0.45.

Algorithm 8 (MwGRepCau): Our final algorithm is identical to MwGRep except that additive updates for β are proposed from a **Cauchy** distribution. The Cauchy scaling was optimised to give the best ACT over the first 1000 iterations.

4 Results

Each RWM variation was tested against data sets D1 and D2 as described in Section 2.3.1. For each data set, each algorithm was started from the known "true" parameter values and was run 3 times with 3 different random seeds (referred to as Replicates 1-3). All algorithms were run for 11000 iterations; a burn in of 1000 iterations was sufficient in all cases.

Priors were independent and exponential with means the known "true" parameter values. The likelihood of an MMPP with maximum and minimum Poisson intensities ψ_{max} and ψ_{min} and with n events observed over a time window of length t_{obs}, is bounded above by $\psi_{max}^n e^{-\psi_{min} t_{obs}}$. In this article only MMPP parameters and their logarithms are considered for estimation. Since exponential priors are employed the parameters and their logarithms therefore have finite variance, and geometric ergodicity is guaranteed.

The accuracy of posterior simulations is assessed via QQ plot comparison with the output from a very long run of a Gibbs sampler (see Section 2.2.3). QQ plots for all replicates were almost entirely within their 95% confidence bounds. Figure 4 shows such plots for Algorithms 1-3 on data set D2 (Replicate 1). In general these three combinations produced the least accurate performance yet even in these cases there is little reason to doubt that ACT's represent each algorithm's exploration of the true posterior rather than a tail or minor mode. The first replicate of Algorithm 4 on D2 also showed an imperfect fit in the tails. This, and the replicate's uncharacteristically high ACT's arise directly from an excursion lasting for about 500 iterations, in which the Markov chain became stuck in a minor mode with $\psi_1 \approx 7, \psi_2 \approx 14, q_{12} \approx 3, q_{21} \approx 0.3$.

The integrated ACT was estimated for each parameter and each replicate using the final 10 000 iterations from that replicate. Calculation of the likelihood is by far the most computationally intensive operation and is performed four times for each Metropolis within Gibbs iteration (once for each parameter) and only once for each block update. To give a truer indication of overall efficiency the ACTs for each Metropolis within Gibbs replicate have therefore been multiplied by four. Table 1 shows the mean adjusted ACT for each algorithm, parameter, and data set. for each set of three replicates most of the ACTs lay within 20% of their mean, and for the exceptions (Blk and BlkShpCau for data sets D1 and D2, and BlkShp and BlkShpMul for data set D2) full sets of ACTs are given in Table 2 in Appendix C.

In general all algorithms performed better on D1 than on D2 because, as discussed in Section 2.3.1 data set D1 contains more information on the parameters than D2; it therefore has lighter tails and is more easily explored by the chain.

As might be expected, the simple block additive algorithm using Gaussian proposals with variance matrix proportional to the identity matrix (Blk) performs relatively poorly on both data sets. In absolute terms there is much less uncertaintly about the transition intensities q_{12} and q_{21} (both are close to 1) than in the Poisson intensities ψ_1 (10) and ψ_2 (17 for D1 and 30 for D2) since the variance of the output from a Poisson process is proportional to its value. The optimal single scale parameter necessarily tunes to the smallest variance and hence explores q_{12} and q_{21} much more efficiently than ψ_1 and ψ_2.

Overall performance improves enormously once block proposals are from a Gaussian with approximately the correct shape (BlkShp). The efficiency of the Metropolis within Gibbs algorithm with additive Gaussian updates (MwG) lies somewhere between the efficiencies of Blk and BlkShp but the improvement over Blk is larger for data set D1 than for data set

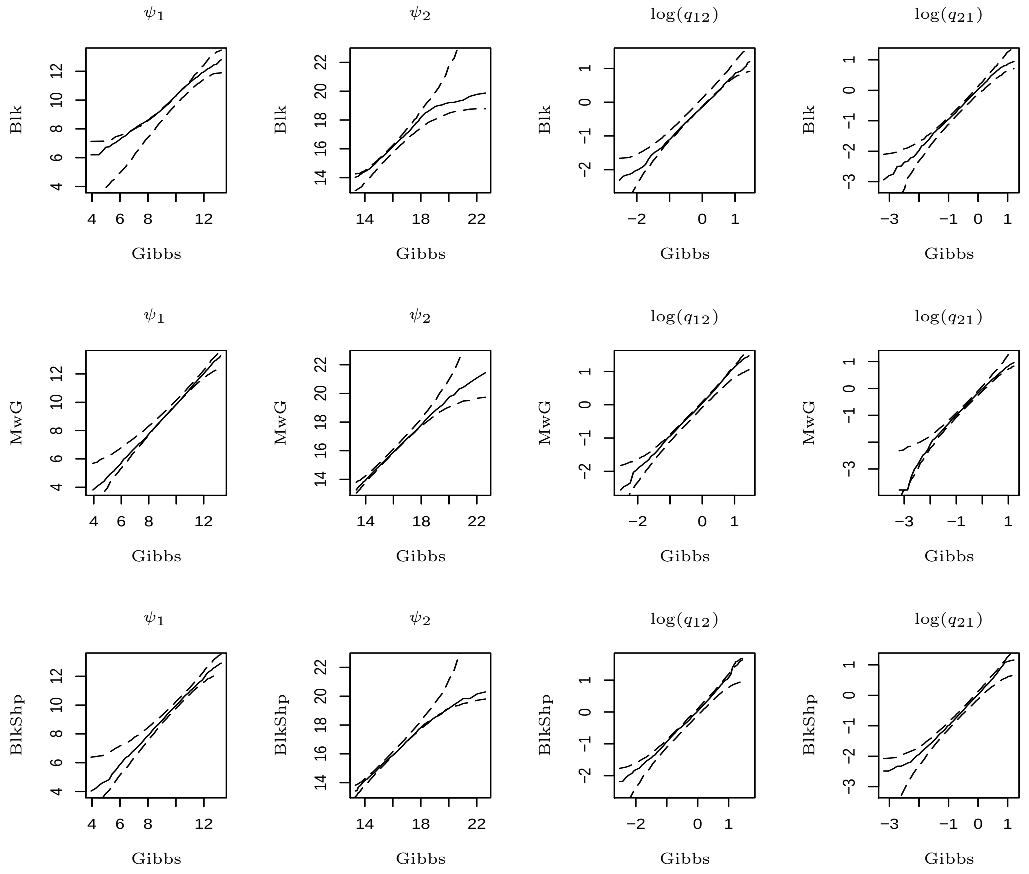

Figure 4: QQ plots for algorithms Blk, MwG, and BlkShp on D2 (Replicate 1). Dashed lines are approximate 95% confidence limits obtained by repeated sampling from iterations 1000 to 100 000 of a Gibbs sampler run; sample sizes were 10 000/ACT, which is the effective sample size of the data being compared to the Gibbs run.

	D1				D2			
Algorithm	ψ_1	ψ_2	$\log(q_{12})$	$\log(q_{21})$	ψ_1	ψ_2	$\log(q_{12})$	$\log(q_{21})$
Blk	66	126	15	19	176	175	80	70
MwG*	22	22	33	33	103	90	114	99
BlkShp	13	18	13	15	46	25	37	36
BlkShpCau	19	32	25	24	63	50	56	38
BlkShpMul	13	17	13	15	33	26	22	16
BlkAdpMul	12	12	14	14	20	20	17	23
MwGRep*	13	14	32	44	20	23	23	21
MwGRepCau*	14	15	37	42	24	233	25	23

Table 1: Mean estimated integrated autocorrelation time for the four parameters over three independent replicates for data sets D1 and D2. *Estimates for MwG replicates have been multiplied by 4 to provide figures comparable with full block updates in terms of CPU time.

D2. As discussed in Section 2.3.1 the parameters in D1 are more nearly independent than the parameters in D2. Thus for data set D1 the principal axes of an elliptical approximation to the posterior are more nearly parallel to the cartesian axes. Metropolis-within-Gibbs updates are (by definition) parallel to each of the cartesian axes and so can make large updates almost directly along the major axis of the ellipse for data set D1.

For the heavy tailed posterior of data set D2 we would expect block updates resulting from a Cauchy proposal (BlkShpCau) to be more efficient than those from a Gaussian proposal. However for both data sets Cauchy proposals are slightly less efficient than Gaussian proposals. It is likely that the heaviness of the Cauchy tails leads to more proposals with at least one negative parameter, such proposals being automatically rejected. Moreover $\hat{\Sigma}$ represents the main posterior mass, yet some large Cauchy jump proposals from this mass will be in the posterior tail. It may be that $\hat{\Sigma}$ does not accurately represent the shape of the posterior tails.

Multiplicative updates (BlkShpMul) make little difference for D1, but for the relatively heavy tailed D2 there is a definite improvement over BlkShpAdd. The adaptive multiplicative algorithm (BlkAdpMul) is slightly more efficient still, since the estimated variance matrix and the overall scaling are refined thoughout the run.

As was noted earlier in this section, due to our choice of exponential priors the quantities estimated in this article have exponential or lighter posterior tails and so all the non-adaptive algorithms in this article are geometrically ergodic. The theory in Section 3.4 suggests ways to improve tail exploration for polynomially ergodic algorithms and so, strictly speaking, need not apply here. However the exponential decay only becomes dominant some distance from the posterior mass, especially for data set D2. Polynomially increasing terms in the likelihood ensure that initial decay is slower than exponential, and that the multiplicative random walk is therefore more efficient than the additive random walk.

The adaptive overall scaling m showed variability of $O(0.1)$ over the first 1000 iterations after which time it quickly settled down to 1.2 for all three replicates on D1 and to 1.1 for all three replicates on D2. Both of these values are very close to the scaling of 1.19 that would be used for a four dimensional update in the scheme of Roberts and Rosenthal (2009). The algorithm similarly learnt very quickly about the variance matrix $\mathbf{\Sigma}$, with individual terms settling down after less than 2000 iterations, and with exploration close to optimal after less than 500 iterations. This can be seen clearly in Figure 5 which shows trace plots for the first 2000 iterations of the first replicate of BlkAdpMul on D2.

The adaptive algorithm uses its own history to learn about $d(d+1)/2$ covariance terms and a best overall scaling. One would therefore expect that the larger the number of parameters, d, the more iterations are required for the scheme to learn about all of the adaptive terms and hence reach a close to optimal efficiency. To test this a data set (D3) was simulated from a three-dimensional MMPP with $\boldsymbol{\psi} = [10, 17, 30]^t$ and $q_{12} = q_{13} = q_{21} = q_{23} = q_{31} = q_{32} = 0.5$. The following adaptive algorithm was then run three times, each for 20 000 iterations.

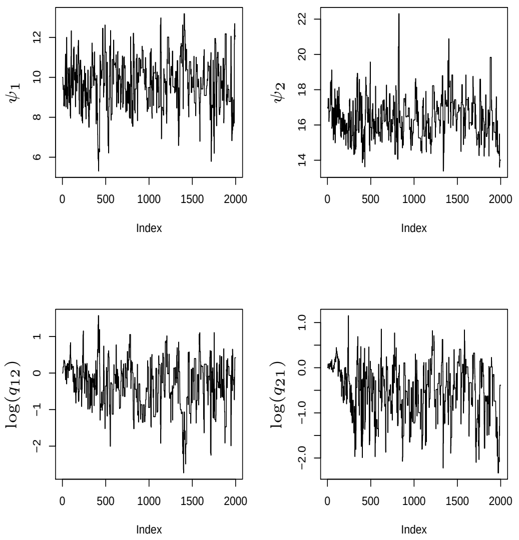

Figure 5: Trace plots for the first 2000 iterations of BlkAdpMul on data set D2 (Replicate 1).

Algorithm 6b (BlkAdpMul(b)): This adaptive algorithm is identical to BlkAdpMul (with $d = 9$) except that no adaptive proposals were used until at least 100 non-adaptive proposals had been accepted, and that if an adaptive proposal was accepted then the overall scaling was updated with $m \leftarrow m + 3\ \Delta/i^{1/2}$ so that the equilibrium acceptance rate was approximately 0.25.

Figure 6 shows the evolution of four of the forty six adaptive parameters (Replicate 1). All parameters seem close to their optimal values after 10 000 iterations, although covariance parameters appear to be still slowly evolving even after 20 000 iterations. In contrast, exploration of the posterior is close to its final optimum after only 1500 iteration as can be seen in trace plots of the first 4000 iterations of the same replicate (Figure 7). This behaviour was repeated across the other two replicates, indicating that, as with the two-dimensional adaptive and non-adaptive runs, even a very rough approximation to the variance matrix improves efficiency considerably. Over the full 20 000 iterations, all three replicates showed a definite multimodality with λ_2 often close to either λ_1 or λ_3, indicating that the data might

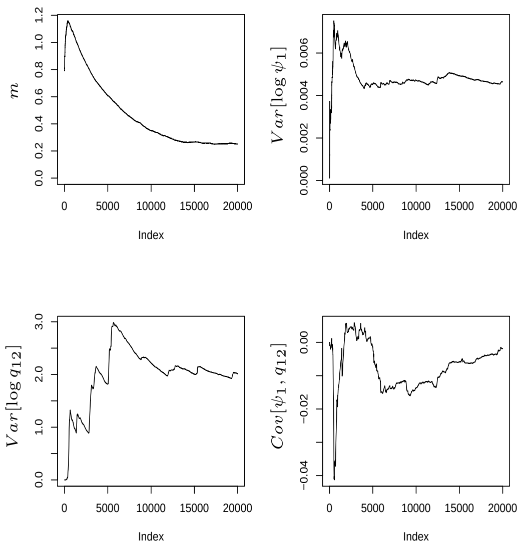

Figure 6: Plots of the adaptive scaling parameter m and three estimated covariance parameters Var $[\psi_1]$, Var $[q_{12}]$, and Cov $[\psi_1, q_{12}]$ for BlkAdpMul(b) on data set D3 (Replicate 1).

reasonably be explained by a two dimensional MMPP. In all three replicates the optimal scaling settled between 0.25 and 0.3, noticeably lower than Roberts and Rosenthal (2009) value of $2.38/\sqrt{9}$. With reference to Section 3.1 this is almost certainly due to the roughness inherent in a multimodal posterior.

The reparameterisation of Section 3.7 was designed for data sets similar to D2, and on this data set the resulting Metropolis within Gibbs algorithm (MwGRep) is at least as efficient as the adaptive multiplicative random walk. On data set D1 however exploration of q_{12} and q_{21} is arguably less efficient than for the Metropolis within Gibbs algorithm with the original parameter set. The lack of improvement when using a Cauchy proposal for β (MwGRepCau) suggests that this inefficiency is not due to poor exploration of the potentially heavy tailed β. Further investigation in the $(\overline{\psi}, q, \alpha, \beta)$ parameter space showed that for data set D1 only q was explored efficiently; the posteriors of $\overline{\psi}$ and β were strongly positively correlated ($\rho \approx 0.8$), and both $\overline{\psi}$ and β were strongly negatively correlated with α ($\rho \approx -0.65$).

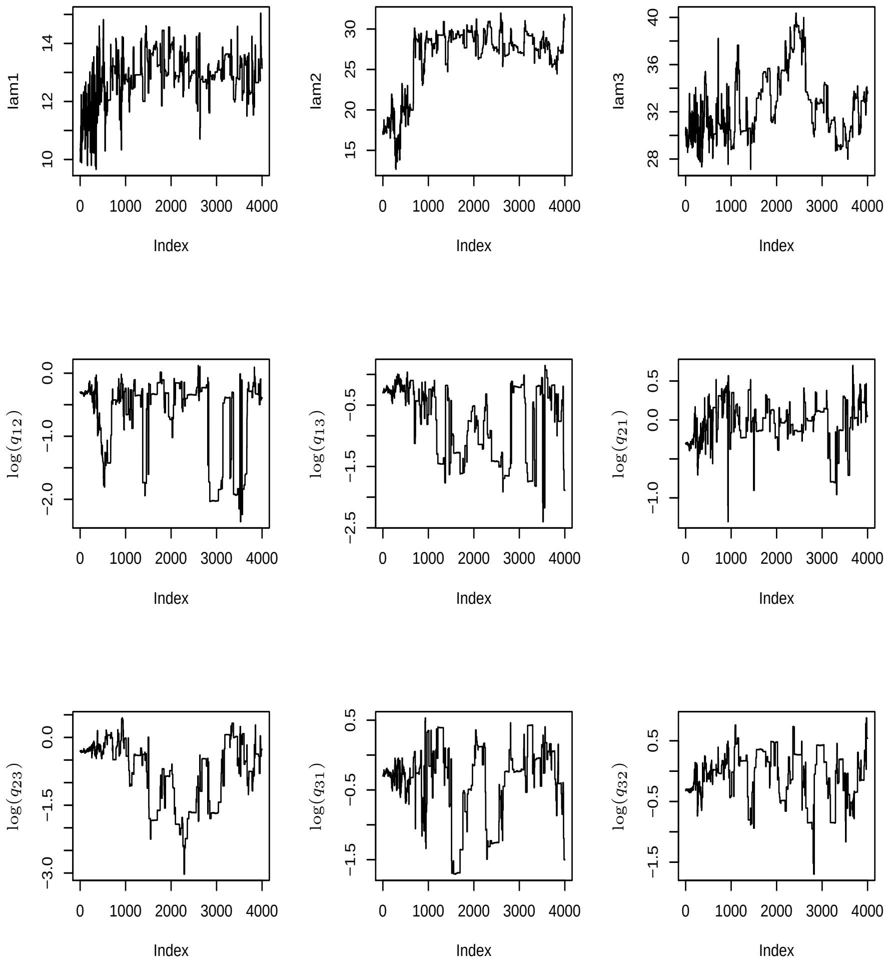

Figure 7: Trace plots for the first 4000 iterations of the first replicate of BlkAdpMul(b) on data set D3.

Posterior correlations were small $|\rho| < 0.3$ for all parameters with data set D2 and for all correlations involving q for data set $D1$.

The optimal scaling for the one-dimensional additive Cauchy proposal in MwGRepCau was approximately two thirds of the optimal scaling for the one-dimensional additive Gaussian proposal in MwGRep. In four dimensions the ratio was approximately one half. These ratios allow the Cauchy proposals to produce similar numbers of small to medium sized jumps to the Gaussian proposals.

5 Discussion

We have described the theory and intuition behind a number of techniques for improving the efficiency of random walk Metropoplis algorithms and tested these on two data sets generated from Markov modulated Poisson processes (MMPPs). Some implementations were uniformly successful at improving efficiency, whilst for other's success depended on the shape and/or tails of the posterior. All of the underlying concepts discussed here are quite general and easily applied to statistical models other than the MMPP.

Simple acceptance rate tuning to obtain the optimal overall variance term for a symmetric Gaussian proposal can increase efficiency by many orders of magnitude. However with our data sets, even after such tuning, the RWM algorithm was very inefficient. The effectiveness of the sampling increased enormously once the shape of the posterior was taken into account by proposing from a Gaussian with variance proportional to an estimate of the posterior variance. For Algorithms 3, 4 and 5 the posterior variance was estimated though a short "training run" - the first 1000 iterations after burn-in of Algorithm 1.

As expected, use of the "multiplicative random walk" (Algorithm 5), a random walk on the posterior of the logarithm of the parameters, improved efficiency most noticeably on the posterior with the heavier tails. However, contrary to expectation, even on the heavier tailed posterior an additive Cauchy proposal (Algorithm 4) was, if anything, less efficient than a Gaussian. Tuning of Cauchy proposals was also more time-consuming since simple acceptance rate criteria could not be used.

Algorithm 6 combined the successesful strategies of optimal scaling, shape tuning, and transforming the data, to create a multiplicative random walk which learned the most efficient shape and scale parameters from its own history as it progressed. This adaptive scheme was easy to implement and was arguably the most efficient algorithm for each of the data sets. A slight variant of this algorithm was used to explore the posterior of a three-dimensional MMPP and showed that in higher dimensions, such algorithms do take longer to discover close to optimal values for the adaptive parameters. These runs also confirmed the finding for the two dimensional MMPP that RWM efficiency improves enormously with knowledge of the posterior variance, even if this knowledge is only approximate. For a multimodal posterior such as that found for the three-dimensional MMPP it might be argued that a different variance matrix should be used for each mode. Such "regionally adaptive" algorithms present additional problems, such as the definition of the different regions, and are discussed further in Roberts and Rosenthal (2009).

Metropolis within Gibbs updates performed better when the parameters were close to orthogonal, at which point they were almost as efficient as an equivalent block updated with tuned shape matrix. The best Metropolis within Gibbs scheme for data set D2 arose from a new reparameterisation devised specifically for the two dimensional MMPP with parameter orthogonality in mind. On D2 this performed nearly as well as the best scheme, the adaptive multiplicative random walk.

The adaptive schemes discussed here provide a significant step towards a goal of completely automated algorithms. However, as already discussed, for d model-parameters, a posterior variance matrix has $O(d^2)$ components. Hence the length of any "training run" or of the adaptive "learning period" increases quickly with dimension. For high dimension it is therefore especially important to utilise to the full any problem specific knowledge that is available so as to provide as efficient a starting algorithm as possible.

References

Bai, Y., Roberts, G. O. and Rosenthal, J. S. (2009). On the containment condition for adaptive Markov chain Monte Carlo algorithms. *Submitted* Preprint.

Bédard, M. (2007). Weak convergence of Metropolis algorithms for non-i.i.d. target distributions. *Ann. Appl. Probab.* **17**(4), 1222–1244.

Bédard, M. (2008). Optimal acceptance rates for Metropolis algorithms: moving beyond 0.234. *Stochastic Process. Appl.* **118**(12), 2198–2222.

Burzykowski, T., Szubiakowski, J. and Ryden, T. (2003). Analysis of photon count data from single-molecule fluorescence experiments. *Chemical Physics* **288**, 291–307.

Carlin, B. P. and Louis, T. A. (2009). *Bayesian methods for data analysis*. Texts in Statistical Science Series, CRC Press, Boca Raton, FL, 3rd edition.

Dellaportas, P. and Roberts, G. O. (2003). An introduction to MCMC. In: *Spatial Statistics and Computational Methods* (ed. J. Moller), number 173 in Lecture Notes in Statistics, Springer, Berlin, 1–41.

Fearnhead, P. and Sherlock, C. (2006). An exact Gibbs sampler for the Markov modulated Poisson processes. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **68**(5), 767–784.

Gamerman, D. and Lopes, H. F. (2006). *Markov chain Monte Carlo*. Texts in Statistical Science Series, Chapman & Hall/CRC, Boca Raton, FL, 2nd edition, stochastic simulation for Bayesian inference.

Geyer, C. J. (1992). Practical Markov chain Monte Carlo. *Statistical Science* **7**, 473–483.

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

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

Jarner, S. F. and Roberts, G. O. (2002). Polynomial convergence rates of Markov chains. *Ann. Appl. Probab.* **12**(1), 224–247.

Kou, S. C., Xie, X. S. and Liu, J. S. (2005). Bayesian analysis of single-molecule experimental data. *Appl. Statist.* **54**, 1–28.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953). Equations of state calculations by fast computing machine. *J. Chem. Phys.* **21**, 1087–1091.

Meyn, S. P. and Tweedie, R. L. (1993). *Markov chains and stochastic stability*. Communications and Control Engineering Series, Springer-Verlag London Ltd., London.

Neal, P. and Roberts, G. (2006). Optimal scaling for partially updating MCMC algorithm. *Ann. Appl. Probab.* **16**, 475–515.

Roberts, G. O. (2003). Linking theory and practice of MCMC. In: *Highly structured stochastic systems*, volume 27 of *Oxford Statist. Sci. Ser.*, Oxford Univ. Press, Oxford, 145–178, with part A by Christian P. Robert and part B by Arnoldo Frigessi.

Roberts, G. O. and Rosenthal, J. S. (1997). Geometric ergodicity and hybrid Markov chains. *Electron. Comm. Probab.* **2**, no. 2, 13–25 (electronic).

Roberts, G. O. and Rosenthal, J. S. (2001). Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science* **16**, 351–367.

Roberts, G. O. and Rosenthal, J. S. (2007). Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. *J. Appl. Probab.* **44**(2), 458–475.

Roberts, G. O. and Rosenthal, J. S. (2009). Examples of adaptive MCMC. *J. Comp. Graph. Stat.* To appear.

Roberts, G. O., Gelman, A. and Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *The Annals of Applied Probability* **7**, 110–120.

Scott, S. L. and Smyth, P. (2003). The Markov modulated Poisson process and Markov Poisson cascade with applications to web traffic modelling. *Bayesian Statistics* **7**, 1–10.

Sherlock, C. (2005). In discussion of 'Bayesian analysis of single-molecule experimental data'. *Journal of the Royal Statistical Society, Series C* **54**, 500.

Sherlock, C. (2006). *Methodology for inference on the Markov modulated Poisson process and theory for optimal scaling of the random walk Metropolis*. Ph.D. thesis, Lancaster University, available from `http://eprints.lancs.ac.uk/850/`.

Sherlock, C. and Roberts, G. O. (2009). Optimal scaling of the random walk Metropolis on elliptically symmetric unimodal targets. *Bernoulli* To appear.

Sokal, A. (1997). Monte Carlo methods in statistical mechanics: foundations and new algorithms. In: *Functional integration (Cargèse, 1996)*, volume 361 of *NATO Adv. Sci. Inst. Ser. B Phys.*, Plenum, New York, 131–192.

A Convergence rates, eigenfunctions, and intuition

To avoid technical details we present theory in a simplified framework where the MCMC kernels have discrete spectra, and consider only distributions for which the L^2 norm resulting from the inner product (14) exists. We first motivate (14).

Proposition 1 *Let $P(\mathbf{x}, d\mathbf{x}')$ be a reversible kernel with stationary distribution $\pi(\cdot)$, eigenfunctions $e_i(\cdot)$, and associated eigenvalues β_i. All of the β_i are real, and with the inner* *product defined in (14),* $< e_i(\cdot), e_j(\cdot) >= \delta_{ij}$.

Proof: *Define*

$$S(\mathbf{x}, d\mathbf{x}') := \left(\frac{\pi(\mathbf{x})}{\pi(\mathbf{x}')}\right)^{1/2} \ P(\mathbf{x}, d\mathbf{x}').$$

Since P *is reversible,*

$$\pi(\mathbf{x})P(\mathbf{x}, d\mathbf{x}') = \pi(\mathbf{x}')P(\mathbf{x}', d\mathbf{x}).$$

Divide both sides by $(\pi(\mathbf{x})\pi(\mathbf{x}'))^{1/2}$ *to see that* $S(\mathbf{x}, d\mathbf{x}') = S(\mathbf{x}', d\mathbf{x})$. *Thus* S *is symmetric and consequently has real eigenvalues* β_i *and associated eigenfunctions* $e_i^*(\cdot)$ *with* $\int d\mathbf{x}\ e_i^*(\mathbf{x})\ e_j^*(\mathbf{x}) = \delta_{ij}$. *Now for any* i,

$$\beta_i\ e_i^*(\mathbf{x}') = \int d\mathbf{x}\ e_i^*(\mathbf{x})\ S(\mathbf{x}, d\mathbf{x}') = \int d\mathbf{x}\ e_i^*(\mathbf{x})\ \frac{\pi(\mathbf{x})^{1/2}}{\pi(\mathbf{x}')^{1/2}}\ dP(\mathbf{x}, \mathbf{x}').$$

Thus $e_i := \pi^{1/2}\ e_i^*$ *is an eigenfunction of* P *with eigenvalue* β_i. *Further*

$$\delta_{ij} = \int d\mathbf{x}\ e_i^*(\mathbf{x})\ e_j^*(\mathbf{x}) = \int d\mathbf{x}\ \frac{e_i(\mathbf{x})\ e_j(\mathbf{x})}{\pi} =< e_i, e_j > .$$

We next motivate the idea of geometric ergodicity and show that a geometric rate of convergence is given by the second largest eigenvalue, provided its value is strictly less than one. We employ the shorthand notation for measure ρ and kernel P, $\rho P := \int d\mathbf{x}\ \rho(\mathbf{x})P(\mathbf{x}, \cdot)$.

Proposition 2 *Let* P *be a reversible kernel with stationary distribution* π, *eigenvalues* β_i *with* $1 = \beta_1 \geq \beta_2 \geq \beta_3, \ldots$. *For initial density* ρ,

$$||\rho P - \pi||_2 \leq \beta_2\, ||\rho - \pi||_2\,.$$

Proof: *Write*

$$\rho(\cdot) = \sum_{i=1}^{\infty} a_i\ e_i(\cdot)$$

and note that, since $e_1 = \pi$, $a_1 =< \rho, e_1 >= 1$. *Thus*

$$||\rho - \pi||_2 = \left|\left|\sum_2^{\infty} a_i\ e_i\right|\right|_2 = \left(\sum_2^{\infty} a_i^2\right)^{1/2}.$$

But $\rho\ P = \sum_1^\infty a_i\ \beta_i e_i$ *and so*

$$||\rho P - \pi||_2 = \left|\left|\sum_2^\infty a_i\ \beta_i\ e_i\right|\right|_2 \leq \beta_2 \left|\left|\sum_2^\infty a_i\ e_i\right|\right|_2 = \beta_2 \left(\sum_2^\infty a_i^2\right)^{1/2}.$$

Note that $\left|\left|\rho P^k - \pi\right|\right|_2 = \left|\left|\left(\rho P^{k-1}\right) P - \pi\right|\right|_2 \leq \beta_2 \left|\left|\rho P^{k-1} - \pi\right|\right|_2$. Iterating this procedure, we find that $\left|\left|\rho P^k - \pi\right|\right|_2 \leq \beta_2^k \left|\left|\rho - \pi\right|\right|_2$.

We finally consider two reversible kernels with the same stationary distribution, and apply them sequentially.

Proposition 3 *Let reversible kernels* $A^{(1)}$ *and* $A^{(2)}$ *both have stationary distribution* $\pi(\cdot)$*, and denote their second largest eigenvalues as* $\beta_2^{(1)}$ *and* $\beta_2^{(2)}$ *respectively. Let* A^* *be a combination algorithm which alternates iterations from* $A^{(1)}$ *and* $A^{(2)}$*. Then*

$$||\nu A^* - \pi||_2 \leq \beta_2^{(1)} \beta_2^{(2)} \, ||\nu - \pi||_2 \, .$$

Proof: *First decompose the eigenfunctions of* $A^{(1)}$ *in terms of the eigenfunctions of* $A^{(2)}$*:*

$$e_i^{(1)} = \sum_{j=1}^\infty c_{ij} e_j^{(2)},$$

where $c_{ij} =< e_i^{(1)}, e_j^{(2)} >$*. Denote the remaining eigenvalues of* $A^{(1)}$ *and* $A^{(2)}$ *by* $\beta_i^{(1)}$ *and* $\beta_i^{(2)}$ *and expand* ρ *in terms of the eigenfunctions of* $A^{(1)}$ *to obtain*

$$\rho A^* = \sum_{i=1}^\infty a_i \beta_i^{(1)} e_i^{(1)}\ A^{(2)} = \pi + \sum_{i=2}^\infty a_i \beta_i^{(1)} e_i^{(1)}\ A^{(2)} = \pi + \sum_{i=2}^\infty a_i \beta_i^{(1)} \sum_{j=2}^\infty c_{ij} \beta_j^{(2)} e_j^{(2)}.$$

Therefore

$$||\rho A^* - \pi||_2 = \left|\left|\sum_{i=2}^\infty a_i \beta_i^{(1)} \sum_{j=2}^\infty c_{ij} \beta_j^{(2)} e_j^{(2)}\right|\right|_2 \leq \beta_2^{(1)} \beta_2^{(2)} \left|\left|\sum_{i=2}^\infty a_i \sum_{j=2}^\infty c_{ij} e_j^{(2)}\right|\right|_2 = \beta_2^{(1)} \beta_2^{(2)} \, ||\nu - \pi||_2 \, .$$

Repeated application of this result leads to: $\left|\left|\nu A^{*k} - \pi\right|\right|_2 \leq \left(\beta_2^{(1)} \beta_2^{(2)}\right)^k ||\nu - \pi||_2$.

B Reparameterisation for the 2 dimensional MMPP

A Taylor expansion of the log-likelihood of a two-dimensional MMPP with $\psi_1 \approx \psi_2$ was given in Section 3.7. The derivation is sketched in this appendix and further details of the $(\overline{\psi}, q, \alpha, \beta)$ reparameterisation are provided. For a fuller derivation the reader is referred to Sherlock (2006).

For a two dimensional MMPP with stationary distribution $[\nu_1, \nu_2]^t$, first reparameterise to $(\overline{\psi}, \mathbf{\Psi}^*, q, \mathbf{Q}^*)$ with

$$\overline{\psi} = \boldsymbol{\nu}^t \boldsymbol{\psi} \ , \quad \mathbf{\Psi} = \overline{\psi}(\mathbf{I} + \mathbf{\Psi}^*) \ , \quad q = q_{12} + q_{21} \ , \quad \mathbf{Q}^* = -\frac{1}{q}\mathbf{Q} = \left[\begin{array}{cc} \nu_2 & -\nu_2 \\ -\nu_1 & \nu_1 \end{array} \right] .$$

With this reparameterisation

$$e^{(\mathbf{Q}-\mathbf{\Psi})t_i} = e^{-\overline{\psi}t_i} e^{-(\mathbf{Q}^* q t_i + \mathbf{\Psi}^* \overline{\psi} t_i)}$$

and therefore

$$L(\mathbf{Q}, \mathbf{\Psi}, \mathbf{t}) = \overline{\psi}^n e^{-\overline{\psi} t_{obs}} \boldsymbol{\nu}^t e^{-(\mathbf{Q}^* q t_1 + \mathbf{\Psi}^* \overline{\psi} t_1)} (\mathbf{I} + \mathbf{\Psi}^*) \dots$$
$$\dots e^{-(\mathbf{Q}^* q t_n + \mathbf{\Psi}^* \overline{\psi} t_n)} (\mathbf{I} + \mathbf{\Psi}^*) e^{-(\mathbf{Q}^* q t_{n+1} + \mathbf{\Psi}^* \overline{\psi} t_{n+1})} \mathbf{1}.$$

But

$$e^{-(\mathbf{Q}^* q t_i + \mathbf{\Psi}^* \overline{\psi} t_i)} = \mathbf{I} - (\mathbf{Q}^* q t_i + \mathbf{\Psi}^* \overline{\psi} t_i) + \frac{1}{2}(\mathbf{Q}^* q t_i + \mathbf{\Psi}^* \overline{\psi} t_i)^2 + \dots \ .$$

Expand the likelihood in terms of $\mathbf{\Psi}^*$ and for notational simplicity, temporarily ignore the factor $\overline{\psi}^n e^{-\overline{\psi} t_{obs}}$ and products of powers of $\overline{\psi} t_i$ and $q t_i$. Since $\mathbf{Q}^{*n} = \mathbf{Q}^*$, terms in $\mathbf{\Psi}^*$, $(\mathbf{\Psi}^*)^2$, and $(\mathbf{\Psi}^*)^3$ are then multiples respectively of

$$\boldsymbol{\nu}^t \mathbf{Q}^{*a_1} \mathbf{\Lambda}^* \mathbf{Q}^{*a_2} \mathbf{1} \ , \quad \boldsymbol{\nu}^t \mathbf{Q}^{*b_1} \mathbf{\Lambda}^* \mathbf{Q}^{*b_2} \mathbf{\Lambda}^* \mathbf{Q}^{*b_3} \mathbf{1} \ , \ \text{and} \ \ \boldsymbol{\nu}^t \mathbf{Q}^{*c_1} \mathbf{\Lambda}^* \mathbf{Q}^{*c_2} \mathbf{\Lambda}^* \mathbf{Q}^{*c_2} \mathbf{\Lambda}^* \mathbf{Q}^{*c_4} \mathbf{1}$$

with $a_1, a_2, b_1, b_2, b_3, c_1, c_2, c_3, c_4$ all either 0 or 1. From their definitions

$$\boldsymbol{\nu}^t \mathbf{Q} = \mathbf{Q}\mathbf{1} = \boldsymbol{\nu}^t \mathbf{\Lambda}^* \mathbf{1} = 0$$

and so to third order the only non vanishing terms are quadratic terms with $b_1 = b_3 = 0$ and cubic terms with $c_1 = c_4 = 0$. Further $\mathbf{\Lambda}^*\mathbf{1} = \delta[-\nu_2, \nu_1]^t$ is a right eigenvector of $\mathbf{Q}^*$ and $\boldsymbol{\nu}^t\mathbf{\Lambda}^* = \delta[\nu_1, \nu_2]$ is a left eigenvector of $\mathbf{Q}^*$, both with eigenvalues 1. Hence in the above products the remaining powers of $\mathbf{Q}^*$ have no effect: both quadratic terms evaluate to $\delta^2\nu_1\nu_2$, and all cubic terms evaluate to $\delta^3\nu_1\nu_2(\nu_2 - \nu_1)$. To cubic terms in δ, the likelihood is therefore

$$L(\overline{\psi}, q, \delta, \nu_1) \approx \overline{\psi}^n e^{-\overline{\psi}t_{obs}} \left(1 + 2\delta^2\nu_1\nu_2 f(\overline{\psi}\mathbf{t}, q\mathbf{t}) + \delta^3\nu_1\nu_2(\nu_2 - \nu_1)g(\overline{\psi}\mathbf{t}, q\mathbf{t})\right)$$

where $f(\cdot, \cdot)$ and $g(\cdot, \cdot)$ are the sums of the many product terms in the expansion of the likelihood involving respectively two and three occurences of $\mathbf{\Lambda}^*$. Equation (17) follows directly after a final Taylor expansion.

Viewed in terms of the original parameters, the transformation given in Section 3.7 is

$$\overline{\psi} := \frac{q_{21}\lambda_1 + q_{12}\lambda_2}{q_{12} + q_{21}} \ , \ q := q_{12} + q_{21} \ , \ \alpha := 2\frac{(\lambda_2 - \lambda_1)(q_{12}q_{21})^{1/2}}{q_{21}\lambda_1 + q_{12}\lambda_2} \text{ and } \beta := \frac{(\lambda_2 - \lambda_1)(q_{12} - q_{21})}{q_{21}\lambda_1 + q_{12}\lambda_2}.$$

Its Jacobian is

$$\frac{\partial(\overline{\psi}, q, \alpha, \beta)}{\partial(\lambda_1, \lambda_2, q_{12}, q_{21})} = \frac{|\lambda_2 - \lambda_1|(q_{12} + q_{21})^2}{(q_{21}\lambda_1 + q_{12}\lambda_2)^2(q_{12}q_{21})^{1/2}}.$$

C Runs with highly variable ACTs

Three replicates were performed for each data set and algorithm, and ACTs are summarised by their mean in Table 1. However for certain algorithms and data sets the ACTs varied considerably; full sets of ACTs for these replicates are given in Table 2.

Algorithm	ψ_1	ψ_2	$\log(q_{12})$	$\log(q_{21})$
Blk (D1)	59,64,75	120,155,104	12,15,17	19,21,17
BlkShpCau (D1)	28,16,12	36,29,31	20,20,35	26,23,24
Blk (D2)	121,259,146	107,262,157	41,139,61	51,110,48
BlkShp (D2)	54,51,34	23,24,29	40,45,27	50,35,23
BlkShpCau (D2)	92,51,46	48,57,46	94,42,31	35,41,39
BlkShpMul (D2)	53,24,23	22,33,25	20,23,24	17,18,13

Table 2: Estimated integrated autocorrelation time for the four parameters, on three independent replicates for Blk and BlkShpCau on data set D1 and Blk, BlkShp, BlkShpCau and BlkShpMul on data set D2.