Statistics and computing (1998) **8**, 357–364

A guided walk Metropolis algorithm

PAUL GUSTAFSON

Department of Statistics, University of British Columbia Vancouver, B.C., Canada V6T 1Z2

Submitted July 1997 and accepted March 1998

The random walk Metropolis algorithm is a simple Markov chain Monte Carlo scheme which is frequently used in Bayesian statistical problems. We propose a guided walk Metropolis algorithm which suppresses some of the random walk behavior in the Markov chain. This alternative algorithm is no harder to implement than the random walk Metropolis algorithm, but empirical studies show that it performs better in terms of efficiency and convergence time.

Keywords: Bayesian computation, Markov Chain Monte Carlo, Metropolis–Hastings algorithm

1. Introduction

Markov Chain Monte Carlo (MCMC) algorithms have revolutionized Bayesian statistics. These algorithms allow a user to draw inferences from a complex posterior distribution on a high-dimensional parameter space, by simulating a Markov chain which has the posterior distribution as its stationary distribution. Under weak conditions the chain converges to its stationary distribution, so that posterior quantities can be estimated from the simulation output. A rich literature now surrounds the theory and practice of MCMC methods. Review material can be found in Neal (1993), Smith and Roberts (1994), Tierney (1994), Besag *et al.* (1995), and Kass *et al.* (1998).

Some MCMC algorithms use more information about the posterior distribution than others; generally one expects algorithms which use more information to be more efficient. The Gibbs sampler (Geman and Geman, 1984; Gelfand and Smith, 1990) requires draws from the posterior conditional distribution of each parameter, given the other parameters. Other algorithms, such as the Langevin algorithm (Roberts and Rosenthal, 1998) and the hybrid algorithm (Duane *et al.*, 1987; Neal, 1993) require evaluations of the log unnormalized posterior density along with its first partial derivatives. In contrast, simpler forms of the Metropolis algorithm (Metropolis *et al.*, 1953; Hastings, 1970) require only unnormalized posterior density evaluations. For instance, the random walk Metropolis algorithm operates by proposing that the Markov chain move to a candidate state obtained by adding noise to the current state. This algorithm is commonly used in practice, often to update a few 'tricky' parameter components which are not amenable to Gibbs sampling. In this context, it is sometimes referred to as a 'Metropolis within Gibbs' algorithm. Work in which the random walk Metropolis algorithm is used as part of the computational scheme includes Muller and Rios Insua (1995), Sargent (1997, 1998), Kuo and Yang (1996), Greenhouse and Wasserman (1998), Newton *et al.* (1996), Verdinelli and Wasserman (1998), Muller and Roeder (1997) and Waller *et al.* (1997). The popularity of this approach is probably due to the ease with which the algorithm is implemented, since more efficient but complex algorithms are available.

Whether they are based on random walk proposals or not, many MCMC algorithms exhibit some random walk behaviour in their exploration of the posterior distribution. That is, the direction in which the Markov chain attempts to move is randomized at each transition. This behaviour seems inefficient, since it can take many iterations for the chain to traverse a substantial distance. For this reason, researchers have considered algorithms which attempt to suppress these random walks to some extent, allowing the chain to move more quickly through the posterior distribution. Neal (1995) and others have considered schemes to suppress the random walk behaviour in the Gibbs sampler, while the hybrid algorithm and variants can be viewed as modified Langevin algorithms with random walk suppression. Since many practitioners opt for simplicity over efficiency by using the random walk Metropolis algorithm, the goal of this paper is to investigate a straightforward scheme for suppressing random walks in this setting. Specifically, we modify the random walk Metropolis algorithm so that the chain tends to move in the same direction at successive transitions.

While the proposed 'guided walk' Metropolis algorithm is easy to understand and implement, its characteristics are not amenable to analytic study. Specifically, the Markov chain has a complicated correlation structure, even for simple posterior distributions. This is also true of the random walk Metropolis algorithm, although Gelman *et al.* (1996) and Roberts *et al.* (1997) make some theoretical progress in the limiting case of infinite dimensional state space. The intractable correlation structure limits the present investigation to empirical studies based on simulation. Some general theory on convergence to stationarity for related finite state-space chains is provided by Diaconis *et al.* (1997).

2. The Metropolis–Hastings algorithm

The Metropolis–Hastings algorithm is the fundamental building block of most MCMC algorithms. Given a target distribution Π, which would be the posterior distribution in Bayesian applications, we wish to construct a Markov chain $\{X_i\}_{i=0}^{\infty}$ with Π as its stationary distribution. If $X_n = x_n$ is the current state of the chain, then the Metropolis–Hastings algorithm proceeds by simulating a candidate or proposal value y from a transition density, $q(x,\cdot)$. The next state, X_{n+1}, is then randomly assigned to be either y with probability $\alpha(x_n, y)$, or x_n with probability $1-\alpha(x_n,y)$, where

$$\alpha(x,y) = \min\left\{\frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}, 1\right\}$$

is the acceptance probability. It is straightforward to verify that the transition from X_n to X_{n+1} does indeed leave Π invariant. By choosing different proposal transition densities $q(\cdot,\cdot)$ we obtain different MCMC algorithms, including the Gibbs sampler, the Langevin algorithm, and the random walk Metropolis algorithm amongst others.

In the case that Π is a continuous univariate distribution on $\mathbb{R}$, the random walk Metropolis algorithm works as follows. The candidate state is obtained by adding noise to the current state; specifically, $q(x,y) = f(y-x)$, for some density f which is symmetric about zero. Commonly f is taken to be normal (mean zero and variance σ^2), in which case the algorithm for updating from $X_n = x_n$ to $X_{n+1} = x_{n+1}$ can be expressed as:

$$\begin{aligned} y &\leftarrow x_n + z, \text{ where } z \sim \mathrm{N}(0,\sigma^2) \\ \alpha &\leftarrow \min\left(\frac{\pi(y)}{\pi(x_n)}, 1\right) \\ x_{n+1} &\leftarrow \begin{cases} y & \text{with probability } \alpha, \\ x_n & \text{with probability } 1-\alpha. \end{cases} \end{aligned}$$

Note that the symmetry property of the proposal transition, $q(x,y) = q(y,x)$, leads to a simple form for the acceptance probability. For brevity we refer to this algorithm simply as the RW algorithm.

Real problems of interest involve high-dimensional target distributions, but algorithms for univariate distributions are still useful. As with the Gibbs sampler, univariate updates can be applied in turn to the full conditional distributions associated with the target. That is, x_1 is updated with a transition based on the target distribution $\Pi(x_1|x_2,\ldots,x_p)$, then x_2 is updated using $\Pi(x_2|x_1,x_3,\ldots,x_p)$, and so on. Since each individual transition leaves Π invariant, the composite transition based on cycling once through all the components also leaves Π invariant. We refer to this use of univariate updates as 'component by component' updating. Most practical Bayesian problems are tackled this way.

3. A guided walk algorithm

At each transition of the RW algorithm, the candidate state is generated from a symmetric distribution centered at the current state. A simple way to suppress this random walk behaviour is as follows. Extend the state from X to (X,P) where $P \in \{-1,+1\}$ is an auxiliary variable. Correspondingly, extend the target distribution from Π on X to the independence product of Π on X and the discrete distribution satisfying $pr\{P=+1\} = pr\{P=-1\} = 0.5$. Clearly we can use a simulated chain with this extended stationary distribution to learn about Π, simply by discarding the sampled P states. A transition of the *Guided Walk* (GW) algorithm from $(X_n,P_n) = (x_n,p_n)$ to $(X_{n+1},P_{n+1}) = (x_{n+1},p_{n+1})$ proceeds as

$$\begin{aligned} y &\leftarrow x_n + p_n|z|, \text{ where } z \sim \mathrm{N}(0,\sigma^2) \\ \alpha &\leftarrow \min\left(\frac{\pi(y)}{\pi(x_n)}, 1\right) \\ (x_{n+1},p_{n+1}) &\leftarrow \begin{cases} (y,p_n) & \text{with probability } \alpha, \\ (x_n,-p_n) & \text{with probability } 1-\alpha. \end{cases} \end{aligned}$$

In contrast to the RW algorithm, here the direction of the candidate y relative to the current state x_n is not randomized at each transition. Rather, the chain moves consistently in the same direction until a candidate state is rejected. Upon rejection, the auxiliary variable P is negated, and the chain begins a series of moves in the opposite direction. If a small value of σ is chosen, then rejections, and consequently reversals, should be infrequent. As a result, the chain should sweep back and forth across the distribution. Thus some random walk suppression is achieved, at the cost of making smaller jumps than might otherwise be used. The guided algorithm is an adaptation of an idea of Horowitz (1991) from the more sophisticated hybrid MCMC setting. In the hybrid setting P, which has a continuous distribution, is interpreted as momentum vector, with a physical interpretation in statistical mechanics problems.

To see that the GW transition leaves the extended target distribution invariant, note that it can be expressed as the composition of two transitions, each of which individually leaves the extended target invariant. Starting from (x, p), the first transition is a Metropolis–Hastings update with candidate state $(x + p|z|, -p)$. Under this transition, a move from the current state to the candidate state is equally as likely as the time-reversed move from the candidate state to the current state. This yields a simple form for the acceptance probability. The second transition is simply an unconditional negation of p, which clearly leaves the extended target distribution invariant.

Both the RW and GW algorithms require specification of σ, the standard deviation of the proposal distribution. As alluded to previously, the choice of σ will affect the equilibrium acceptance rate, which is the expected value of the acceptance probability α under the target distribution. It is straightforward to verify that if the RW and GW algorithms are operated with the same σ, the they will have the same equilibrium acceptance rate.

Example 1 Let the target distribution be standard normal (mean zero and variance one). As noted by Gelman *et al.* (1996), the equilibrium acceptance probability in this situation is $(2/\pi)$ arctan $(2/\sigma)$. Thus we can choose σ to obtain a desired acceptance rate.

Figure 1 displays sample paths for both the RW and GW algorithms. In each case the initial state is $x_0 = -6$, so the chain must move toward the target distribution. The equilibrium acceptance rate for the RW algorithm is set at 0.45, which Gelman *et al.* (1996) found to be a good value according to several criteria. The acceptance rate for the GW algorithm is set at 0.9, to promote sweeping trajectories with relatively few reversals of direction. The plot verifies that the GW algorithm does have a tendency to sweep back and forth across the distribution, suppressing some of the random walk behaviour evident in the sample path from the RW algorithm. □

To compare the two algorithms, we use a χ^2 goodness of fit statistic which measures how well a simulated chain represents the target distribution. In particular, partition the state space into r disjoint subsets $A_1, \ldots, A_r$, with $\Pi(A_i) = 1/r$ for $i = 1, \ldots, r$. Let $E = n/r$ where n is the number of iterations comprising the chain, and let O_i be the count of how many of the n sampled states lie in A_i. As a somewhat general measure of how well the simulated values fit the target distribution, consider the root mean square relative error,

$$RMSRE = \left\{ \frac{1}{r} \sum_{i=1}^{r} \left(\frac{O_i - \mathrm{E}}{\mathrm{E}} \right)^2 \right\}^{1/2},$$

where the mean is taken over the r sets in the a partition. We interpret the *RMSRE* as the typical relative error incurred when using a sample proportion to estimate the target probability of a partition set. Note that $RMSRE = n^{-1/2} FIT$, where

$$FIT = \sqrt{\left\{ \sum_{i=i}^{r} \frac{(O_i - \mathrm{E})^2}{\mathrm{E}} \right\}}$$

is the square root of the χ^2 goodness-of-fit statistic. That is, FIT^2 would approximately have a χ^2 distribution with $r - 1$ degrees of freedom (mean $(r - 1)$ and variance $2(r - 1)$), if the sampled values were independent draws from the target distribution. It is convenient to report *FIT*, rather than *RMSRE*, as the basic measure of the fit of the simulation output to the target distribution. Another possibility would be to assess a chain's performance on the basis of its estimated autocorrelation structure. We prefer the χ^2 criterion because of its direct interpretation in terms of relative error of probability estimates.

Example 1, Continued A set of $m = 1000$ starting values are simulated from the standard normal target distribition. For each starting point, and each in a series of acceptance rates ranging from 25% to 95% in 5% increments, GW and RW chains of length $n = 500$ are simulated. The *FIT* statistic is computed for each chain, using a partition of $r = 10$ adjacent intervals which are equiprobable under the target distribution. The two algorithms perform similarly at low acceptance rates, when the GW algorithm is making frequent reversals, but the GW algorithm outperforms the TW algorithm at high acceptance rates, when the jumps are small but the GW algorithm is heavily 'guided'. On the coarse grid of acceptance rates considered, the median FIT statistic is smallest when the acceptance rate is 70% for the RW algorithm, and 80% for the GW algorithm (though in both cases nearby acceptance rates give virtually identical performance). Using these respective acceptance rates, we examine the ratio of *FIT* statistics (GW to RW). Over the set of $m = 1000$ starting values, the three-number summary (lower quartile, median, upper quartile) of this ratio is (0.62, 0.79, 1.03). Given the interpretation of the *FIT* statistic, the median ratio of 0.79 suggests that relative error can be reduced by 20% as a result of using the GW algorithm instead of the RW algorithm. □

A more practical scenario for comparing the two algorithms involves a multivariate target distribution with component by component updating.

Example 2 Let the target distribution Π be k-variate normal, with zero mean vector, and covariance matrix $\Sigma = (1 - \rho)I_k + \rho J_k$, where I_k is the $k \times k$ identity matrix, and J_k is the $k \times k$ matrix where every entry is one. Thus ρ

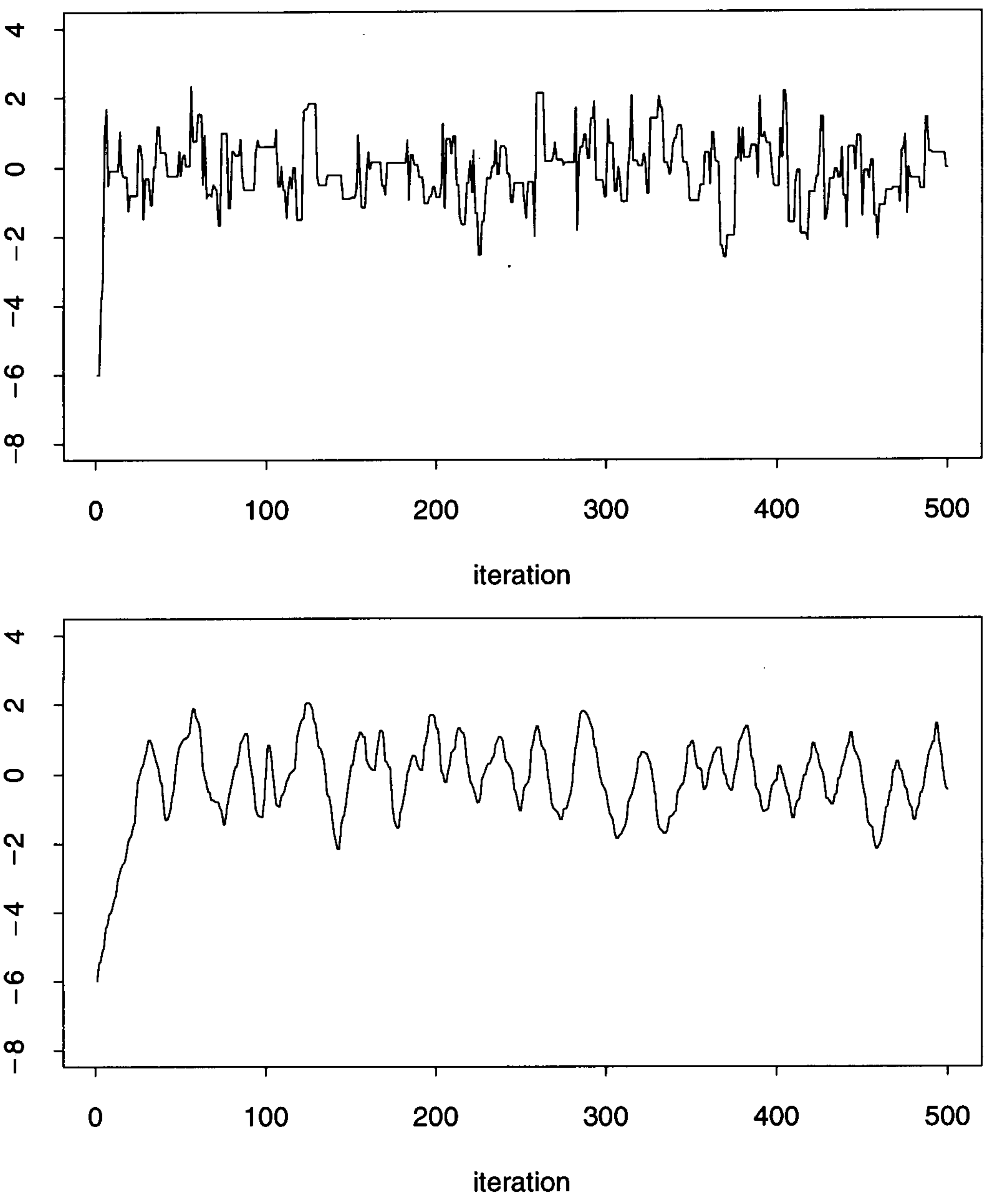

Fig. 1. *Sample paths of the RW and GW algorithms for a standard normal target distribution. The acceptance rate is 0.45 for the RW algorithm (first panel), and 0.9 for the GW algorithm (second panel). The initial state is* $x_0 = -6$ *in each case*

is the correlation between any two distinct components of X. When ρ is close to one, this target distribution will be challenging for MCMC schemes using component by component updating, since the conditional distribution of X_i given the other components will be very narrow compared to the marginal distribution of X_i. Specifically, let $X_{(i)} = (X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_k)$, and let $\bar{X}_{(i)}$ be the mean of $X_{(i)}$. Then the conditional distribution of $X_i|X_{(i)}$ is normal, with mean

$$\mathrm{E}(X_i|X_{(i)}) = \frac{(k-1)\rho}{1+(k-2)\rho}\bar{X}_{(i)}$$

and variance $\mathrm{Var}(X_i|X_{(i)}) = \tau^2$, where

$$\tau^2 = 1 - \rho^2\left[\frac{k-1}{1+(k-2)\rho}\right].$$

We apply component by component updating for both the RW and GW algorithms, using the same proposal standard deviation σ for each component. Upon adapting the previously noted calculation of Gelman *et al.* (1996) to the current example, we find that the equilibrium acceptance rate is $(2/\pi)$ arctan $(2\tau/\sigma)$.

The partition of the state space used to assess fit is based on transforming the target distribution to the standard multivariate normal distribution (mean vector 0 and covariance matrix I_k). Specifically, if $(X_1, \ldots, X_k)$ are distributed according to the target distribution, then $(X_1^*, \ldots, X_k^*)$ are standard multivariate normal, where

$$X_i^* = \frac{1}{\sqrt{1-\rho}} \left\{ X_i - \left[1 - \sqrt{\frac{1-\rho}{1+(k-1)\rho}} \right] \bar{X} \right\}.$$

After transformation, a 'dart board' partition is applied to X^*. This partition comprises the $r = L \times 2^k$ sets obtained by first partitioning the sampled X^* vectors into L equiprobable sets on the basis of the magnitude of $\|X^*\|^2 = (X_1^*)^2 + \cdots + (X_k^*)^2$ compared to appropriate quantiles of the χ_k^2 distribition. Each of the L sets is then further partitioned into 2^k sets on the basis of the pattern of signs of the components of X^*.

The reported simulations are based on a $k = 5$ dimensional target distribution, with correlations of $\rho = 0.95$. A collection of $m = 1000$ starting vectors is simulated from the target distribution. To assess fit a partition of $r = 160$ sets is constructed, using $L = 5$. Each simulated chain is of length $n = 8000$, which gives reasonably large expected counts of $E = 50$ for the *FIT* statistic. The simulation results, given in Figure 2, are qualitatively similar to the results from Example 1. With respect to the coarse grid of acceptance rates used, the median *FIT* statistic is smallest for the GW and RW algorithms at acceptance rates of 45% and 60% respectively. For these rates, the three-number summary of the ratio of *FIT* statistics (GW to RW) is (0.67, 0.83, 1.01). Again, using the GW algorithm instead of the RW algorithm yields a decrease in relative error on the order of 20%. □

The third panel of Figure 2, and the analogous plot for Example 1 (not shown) suggest that the GW algorithm yields near-optimal performance over a wider range of acceptance rates (or equivalently proposal variances) than does the RW algorithm. This suggests that the advantage of the GW algorithm over the RW algorithm might be more pronounced in problems where the variance of each conditional target distribition varies with the conditioning variables. We investigate this with an example.

Example 3 Consider a bivariate target distribution for (X_1, X_2) under which conditionally $X_1|X_2$ is normal with mean zero and variance $\exp(cX_2 - (c^2/2))$ for some $c > 0$, while marginally $X_2 \sim$ N(0, 1). Marginally both X_1 and X_2 have unit variance, so it seems reasonable to use the same proposal variance for updating both components. The conditional variance of $X_1|X_2$ depends more strongly on X_2 when c is large; thus we conjecture that the GW algorithm will have a more pronounced advantage over the RW algorithm as c increases. Specifically, note that the ratio $\{\text{Var}(X_1|X_2 = 0.674)\}^{1/2}\{\text{Var}(X_1|X_2 = -0.674)\}^{1/2} = \exp(0.674c)$ reflects the variability in the conditional standard deviation when X_2 takes on its marginal quartile values. We try values of c which make this ratio 5, 20, and 40 respectively.

The simulation study is based on $m = 1000$ chains of length $n = 1000$, each started in equilibrium. An *ad hoc* heuristic is used to select proposal standard deviations, and empirical acceptance rates are reported. The fit is measured by transforming (X_1, X_2) to a standard bivariate normal distribution and then applying the 'dart board' partition with $L = 5$ to the transformed variates. Specifically, we transform according to $X_1^* = \exp(-(c/2)X_2 + (c^2/4))X_1$ and $X_2^* = X_2$. Using an empirically optimal σ for each algorithm, the three-number summary of the GW *FIT* to RW *FIT* ratio is (0.62, 0.79, 0.98) when the conditional standard deviation ratio is 5. When this ratio is 20 and 40 we obtain three-number summaries of (0.51, 0.75, 1.03) and (0.53, 0.76, 1.01) respectively. In all three cases, the best acceptance rate for the RW algorithm is in the vicinity of 50% , while the best rate for the GW algorithm is near 60%. While the advantage of the GW algorithm might be slightly enhanced when the conditional standard deviation ratio is 20 or 40 rather than 5, the effect is not nearly as striking as intuition suggests. As with the two previous examples, the GW algorithm reduces relative error by roughly 20% compared to the RW algorithm. □

4. Speed of convergence

Thus far we have compared the RW and GW algorithms in terms of how well their output represents the target distribution, given that the chain is started in equilibrium. In practical problems it is not possible to start the chain in equilibrium, so the speed at which the chain reaches equilibrium is an important issue. Since the GW algorithm moves in a consistent direction until a rejection occurs, and moves corresponding to an increase in target density are always accepted, we conjecture that the GW algorithm might be more effective than the RW algorithm at finding the target distribution. We provide the following example comparing how many iterations each algorithm requires to reach the stationary distribution.

Example 4 Consider the exchangeable multivariate normal target distribution $(k = 5, \rho = 0.95)$ of Example 2. We generate $m = 1000$ initial state vectors from the uniform distribution on the 'hyper-square' (0, 30) $\times \ldots \times$ (0, 30). (Since the target distribution is centered at the origin, we purposefully sample initial vectors from a distribution not centered at the origin, in order to make it more challenging for the algorithms to converge.) A GW chain and a RW chain are simulated for each starting value, using acceptance rates that were optimal in Example 2 (45% for the RW algorithm, and 60% for the

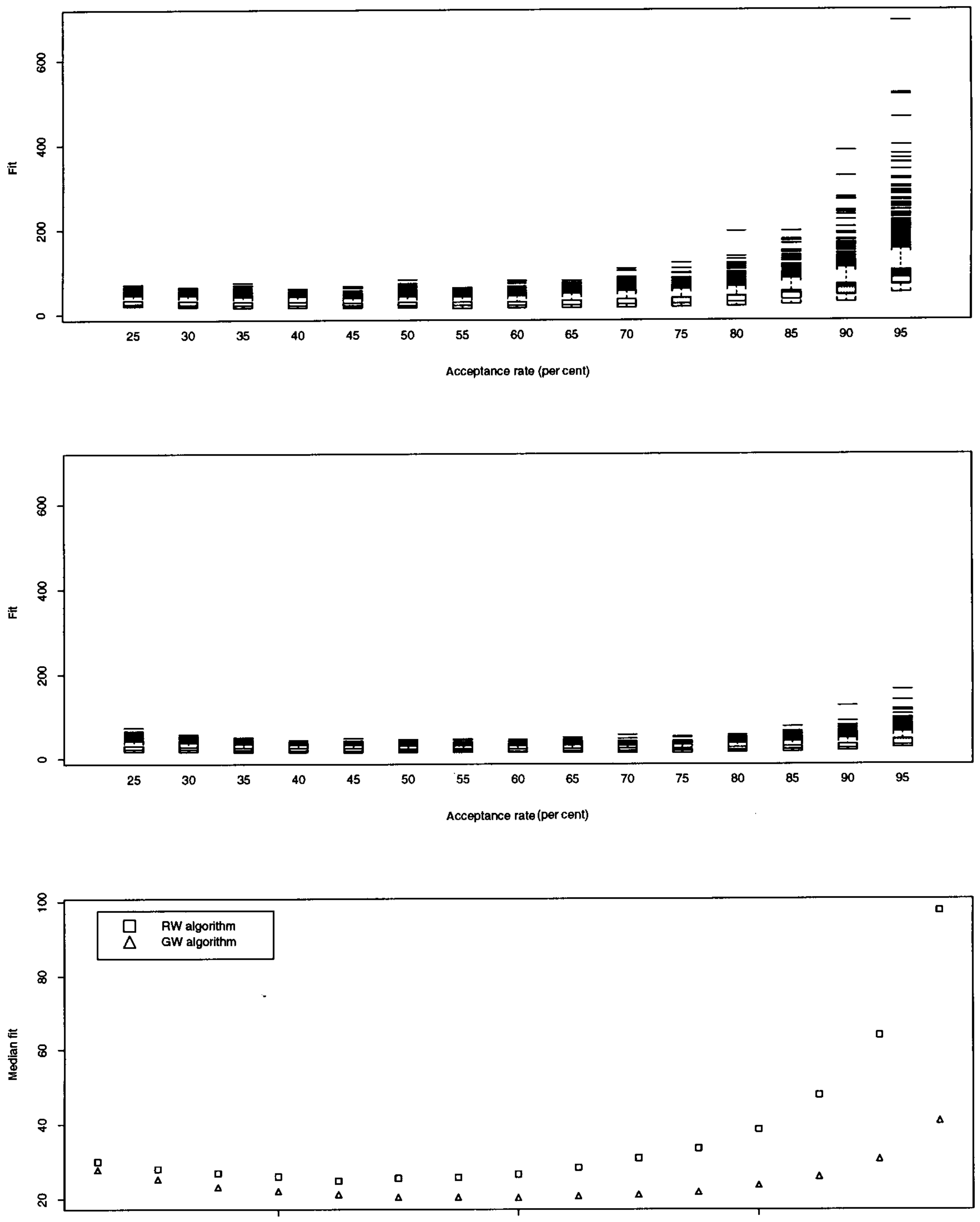

Fig. 2. *Comparison of RW and GW algorithms for an exchangeable multivariate normal target distribution* ($k = 5$, $\rho = 0.95$). *The first panel contains boxplots for the* $m = 1000$ *FIT statistics obtained using the RW algorithm, as a function of acceptance rate. The second panel contains the analogous boxplots for the GW algorithm. The third panel compares the median FIT statistics for the two algorithms*

GW algorithm). Thus in terms of seeking the target distribition, we want to know if the smaller jumps of the GW algorithm are more than compensated for by the 'guided' progress toward the target. Since $X^{\mathrm{T}}\Sigma^{-1}X$ is χ^2_k distributed under the target distribution, each chain is labelled as having converged once $x_n^T\Sigma^{-1}x_n$ is smaller than the 0.95 quantile of the χ^2_k distribition. A scatterplot (not shown) indicates that the sample sizes required for convergence of the two algorithms observe a roughly linear relationship, with no outlying points. The three-number summary of the ratio of required sample sizes (GW to RW) is (0.64, 0.67, 0.70), confirming that the GW algorithm does converge more quickly than the RW algorithm, for this example at least. □

5. Discussion

The guided walk Metropolis algorithm appears to offer consistent but not spectacular gains over the random walk Metropolis algorithm. For the three 'synthetic' target distributions considered here – a univariate normal distribution, an exchangeable multivariate normal distribution with high correlations, and a non-standard bivariate distribution with non-constant conditional variances – the guided walk algorithm offers a median reduction in relative error on the order of 20% . Since relative error is $O(n^{-1/2})$, a 56% increase in chain length would be required to obtain the same reduction simply via extra computation with the random walk Metropolis algorithm. In addition to this gain in efficiency, the guided algorithm tends to converge in one-third fewer iterations in the exchangeable multivariate normal target distribution example.

A multivariate GW algorithm in which the whole state vector is updated simultaneously was also tested. This involved alternately implementing a univariate GW step in a particular direction and then slightly perturbing the direction vector in a manner which leaves the extended target distribution invariant. For the exchangeable multivariate normal target distribution, the performance of this algorithm was virtually identical to that of the multivariate RW algorithm. Consequently detailed results are not reported, and the multivariate GW algorithm was not pursued further.

While the efficiency and convergence time gains achieved by random walk suppression with the GW algorithm might be viewed as moderate in an era of inexpensive computation, it should be noted that they come essentially for free. The extra human effort required to understand and program the guided algorithm rather than the random walk algorithm, and the extra machine time required to complete a guided walk transition rather than a random walk transition are both negligible. Issues regarding choice of proposal variance affect both algorithms, but in fact the guided algorithm appears to be somewhat less sensitive to this specification. Thus the guided walk Metropolis algorithm compares favourably with the popular random walk Metropolis algorithm, at least in the synthetic contexts studied here. A more relevant assessment of the guided algorithm will be made if practitioners try using it in applications.

Acknowledgements

This research was supported by a grant from the Natural Sciences and Engineering Research Council of Canada.

References

Besag, J., Green, P., Higdon, D. and Mengersen, K. (1995) Bayesian computation and stochastic systems (with discussion). *Statistical Science*, **10**, 3–36.

Diaconis, P., Holmes, S. and Neal, R. M. (1997) Analysis of a non-reversible Markov chain sampler. Technical Report BU-1385-M, Biometrics Unit, Cornell University.

Duane, S., Kennedy, A. D., Pendleton, B. J. and Roweth, D. (1987) Hybrid Monte Carlo. *Physics Letters B*, **195**, 216–222.

Gelfand, A. E. and Smith, A. F. M. (1990) Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, **85**, 398–409.

Gelman, A., Roberts, G. and Gilks W. (1996) Efficient Metropolis jumping rules, in *Bayesian Statistics 5*, Berger, J. O., Bernardo, J. M., Dawid, A. P. and Smith, A. F. M. (eds), Oxford University Press.

Geman, S. and Geman, D. (1984) Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE transactions on Pattern Analysis and Machine Intelligence*, **6**, 721–741.

Greenhouse, J. and Wasserman, L. (1996) A practical robust method for Bayesian model selection: a case study in the analysis of clinical trials, in *Bayesian Robustness* Berger, J. O., Betro, B., Moreno, E., Pericchi, L. R., Ruggeri, F., Salinetti, G. and Wasserman, L. (eds), Institute of Mathematical Statistics Lecture Notes – Monograph Series, pp. 41–58.

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

Horowitz, A. M. (1991) A generalized guided Monte Carlo algorithm. *Physics Letters B*, **268**, 247–252.

Kass, R. E., Carlin, B. P., Gelman, A. and Neal, R. M. (1998) Markov chain Monte Carlo in practice: a roundtable discussion. *The American Statistician*, **52**, 93–100.

Kuo, L. and Yang, T. Y. (1996) Bayesian computation for non-homogeneous Poisson processes in software reliability. *Journal of the American Statistical Association*, **91**, 763–773.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953) Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, **21**, 1087–1092.

Muller, P. and Rios Insua, D. (1995) Issues in Bayesian analysis of neural network models. Working Paper 95–31, Institute of Statistics and Decision Sciences, Duke University.

Muller, P. and Roeder, K. (1997) A Bayesian semiparametric model for case-control studies with errors in variables. *Biometrika*, **84**, 523–537.

Neal, R. M. (1993) Probabilistic inference using Markov Chain Monte Carlo methods. Technical Report CRG–TR–93–1, Department of Computer Science, University of Toronto.

Neal, R. M. (1995) Suppressing random walks in Markov Chain Monte Carlo using ordered overrelaxation. Technical Report 9508, Department of Statistics, University of Toronto.

Newton, M. A., Czado, C. and Chappell, R. (1996) Bayesian inference for semiparametric binary regression. *Journal of the American Statistical Association*, **91**, 142–153.

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

Roberts, G. O. and Rosenthal, J. S. (1998) Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society B*, to appear.

Sargent, D. J. (1997) A flexible approach to time-varying coefficients in the Cox regression setting. Lifetime Data Analysis, **3**, 13–25.

Sargent, D. J. (1998) A general framework for random effects survival analysis in the Cox proportional hazards setting. Biometrics, **54**, to appear.

Smith, A. F. M. and Roberts, G. O. (1994) Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods (with discussion). *Journal of the Royal Statistical Society B*, **55**, 3–23.

Tierney, L. (1994) Markov chains for exploring posterior distributions (with discussion). *Annals of Statistics*, **22**, 1701–1762.

Verdinelli and Wasserman (1998) Bayesian goodness of fit testing using infinite dimensional exponential families. Annals of Statistics, **26**, 1215–1241.

Waller, L. A., Carlin, B. P., Xia, H. and Gelfand, A. E. (1997) Hierarchical spatio-temporal mapping of disease rates. *Journal of the American Statistical Association*, **92**, 607–617.