Bayesian Computation via Markov chain Monte Carlo

Radu V. Craiu
Department of Statistics
University of Toronto

Jeffrey S. Rosenthal
Department of Statistics
University of Toronto

(Version of June 6, 2013.)

1 Introduction

A search for Markov chain Monte Carlo (or MCMC) articles on Google Scholar yields over 100,000 hits, and a general web search on Google yields 1.7 million hits. These results stem largely from the ubiquitous use of these algorithms in modern computational statistics, as we shall now describe.

MCMC algorithms are used to solve problems in many scientific fields, including physics (where many MCMC algorithms originated) and chemistry and computer science. However, the widespread popularity of MCMC samplers is largely due to their impact on solving statistical computation problems related to Bayesian inference. Specifically, suppose we are given an independent and identically distributed (henceforth iid) sample $\{x_1, \ldots, x_n\}$ from a *parametric sampling density* $f(x|\theta)$, where $x \in \mathcal{X} \subset \mathbf{R}^k$ and $\theta \in \Theta \subset \mathbf{R}^d$. Suppose we also have some *prior density* $p(\theta)$. Then the Bayesian paradigm prescribes that all aspects of inference should be based on the

posterior density

$$\pi(\theta|\vec{x}) = \frac{p(\theta)f(\vec{x}|\theta)}{\int_\Theta p(\theta)f(\vec{x}|\theta)d\theta} \tag{1}$$

where $\vec{x} = \{x_1, \ldots, x_n\}$. Of greatest interest are the *posterior means* of functionals $g : \mathcal{X} \to \mathbf{R}$, defined by

$$I = \int_\Theta g(\theta)\pi(\theta|\vec{x})d\theta = \frac{\int_\Theta g(\theta)p(\theta)f(\vec{x}|\theta)d\theta}{\int_\Theta p(\theta)f(\vec{x}|\theta)d\theta}. \tag{2}$$

Such expectations are usually impossible to compute directly, because of the integrals that appear in the denominator of (1) and in (2). However, we can still study (2) as long as we can *sample* from π. In the traditional Monte Carlo paradigm, we generate an iid sample $\theta_1, \ldots, \theta_M$ from π, and then estimate (2) using

$$\hat{I}_M = \frac{1}{M}\sum_{i=1}^{M} g(\theta_i). \tag{3}$$

This estimate generally works well in cases where the iid sample $\theta_1, \ldots, \theta_M$ can be generated, and in particular $\hat{I}_M \to I$ with probability 1 as $M \to \infty$.

However, when π is complicated and high-dimensional, classical methods devised to draw independent samples from the distribution of interest are not implementable. In this case, a *Markov chain Monte Carlo* (MCMC) algorithm proceeds by instead constructing an updating algorithm for generating θ_{t+1} once we know θ_t. MCMC updating algorithms are constructed by specifying a set of transition probabilities for an associated Markov chain (see e.g. (44; 43)). It then uses the realizations $\theta_1, \ldots, \theta_M$ obtained from the Markov chain as the Monte Carlo sample in (3), or more commonly with the slight modification

$$\hat{I}_M = \frac{1}{M-B}\sum_{i=B+1}^{M} g(\theta_i). \tag{4}$$

where B is a fixed non-negative integer (e.g. 1,000) indicating the amount of *burn-in*, i.e. the number of initial samples that will be discarded due to being excessively biased

towards the (arbitrary) initial value θ_0. If the Markov chain has π as an invariant distribution, and if it satisfies the mild technical conditions of being aperiodic and irreducible, then the *ergodic theorem* implies that with probability one, $\hat{I}_M \to I$ as $M \to \infty$ (see Section 8.1).

Now, unlike traditional Monte Carlo where the samples are independent, MCMC samplers yield dependent draws. This makes the theoretical study of these algorithms and the assessment of their speed of convergence and Monte Carlo variance much more difficult to assess. A comprehensive evolution of the field can be traced through the articles included in the volumes edited by (1) and (2) as well as the books devoted to Monte Carlo methods in statistics, e.g. (3), (4), (5) and (6). We recognize that for those scientists who are not familiar with MCMC techniques, but need them for their statistical analysis, the wealth of information contained in the literature can be overwhelming. Therefore, in this review we provide, in concise form, the ingredients needed for using MCMC in most applications. Along the way, we will point the user in need of more sophisticated methods to the relevant literature.

1.1 Example: Lupus Data

As a specific example, we present the lupus data which was first analyzed by (7), and subsequently by (8) and (9). This data, presented in Table 1, contains disease status for 55 patients of which 18 have been diagnosed with latent membranous lupus, together with two clinical covariates, IgA and $\Delta IgG = IgG3 - IgG4$, that are computed from their levels of immunoglobulin of type A and of type G, respectively.

In order to model the data generation process we need to formulate the *sampling distribution* of the binary response variable. We can follow (7) and consider a probit regression (PR) model, i.e. for each patient $1 \leq i \leq 55$, we model the disease indicator variables as independent

$$Y_i \sim \text{Bernoulli}(\Phi(x_i^T \beta)), \tag{5}$$

Table 1: *The number of latent membranous lupus nephritis cases (numerator), and the total number of cases (denominator), for each combination of the values of the two covariates.*

	IgA				
ΔIgG	0	0.5	1	1.5	2
-3.0	0/ 1	-	-	-	-
-2.5	0/ 3	-	-	-	-
-2.0	0/ 7	-	-	-	0/ 1
-1.5	0/ 6	0/ 1	-	-	-
-1.0	0/ 6	0/ 1	0/ 1	-	0/ 1
-0.5	0/ 4	-	-	1/ 1	-
0	0/ 3	-	0/ 1	1/ 1	-
0.5	3/ 4	-	1/ 1	1/ 1	1/ 1
1.0	1/ 1	-	1/ 1	1/ 1	4/ 4
1.5	1/ 1	-	-	2/ 2	-

where $\Phi(\cdot)$ is the CDF of $N(0,1)$, $x_i = (1, \Delta IgG_i, IgA_i)$ is the vector of covariates, and β is a 3×1 vector of parameters. We assume a flat prior $p(\beta) \propto 1$ throughout the paper.

For the PR model, the posterior is thus

$$\begin{aligned} \pi_{PR}(\vec{\beta}|\vec{Y}, \vec{I}gA, \vec{\Delta}IgG) &\propto \prod_{i=1}^{55} \Big[\Phi(\beta_0 + \Delta IgG_i\beta_1 + IgA_i\beta_2)^{Y_i} \times \\ &\times \quad (1 - \Phi(\beta_0 + \Delta IgG_i\beta_1 + IgA_i\beta_2))^{(1-Y_i)}\Big]. \end{aligned} \tag{6}$$

We shall return to this example several times below.

1.2 Choice of MCMC Algorithm

Not all MCMC samplers are used equally. Ease of implementation (e.g. pre-existent software), simplicity of formulation, computational efficiency and good theoretical properties are all factors that contribute (not necessarily in this order) to an algorithm's successful and rapid dissemination. In this paper, we shall focus on the most widely used MCMC samplers, including the Metropolis-Hastings algorithm (Section 2), the Gibbs sampler (Section 3), and variable-at-a-time Metropolis (Section 4). We shall also discuss optimising and adapting MCMC algorithms (Section 5), the use of simulated tempering (Section 6), assessing MCMC errors (Section 7), and the theoretical foundations of MCMC (Section 8).

2 The Metropolis-Hastings Algorithm

The Metropolis-Hastings (MH) algorithm was developed by (10) and (11). It updates the Markov chain state as follows. (For simplicity, we shall write the target (posterior) distribution as simply $\pi(\theta)$.) Assume that the state of the chain at time t is θ_t. Then the updating rule to construct θ_{t+1} (i.e., the transition kernel for the MH chain) is defined by the following two steps:

Step 1 A proposal ω_t is drawn from a proposal density $q(\omega|\theta_t)$;

Step 2 Set

$$\theta_{t+1} = \begin{cases} \omega_t & \text{with probability } r \\ \theta_t & \text{with probability } 1-r \end{cases}$$

where

$$r = \min\left\{1, \frac{\pi(\omega_t)q(\theta_t|\omega_t)}{\pi(\theta_t)q(\omega_t|\theta_t)}\right\}. \tag{7}$$

The acceptance probability (7) is independent of the normalizing constant for π (i.e., it does not require the value of the denominator in (1)), and is chosen precisely to ensure that π is an invariant distribution, the key condition to ensure that $\hat{I}_M \to I$ as $M \to \infty$ as discussed above; see Section 8.2.

The most popular variant of the MH algorithm is the *random walk Metropolis* (RWM) in which $\omega_t = \theta_t + \epsilon_t$ where ϵ_t is generated from a spherically symmetric distribution, e.g. the Gaussian case with $\epsilon_t \sim N(0, \Sigma)$. Another common choice is the *independence sampler* (IS) in which $q(\omega|\theta_t) = q(\omega)$, i.e. ω_t does not depend on the current state of the chain, θ_t. Generally, the RWM is used in situations in which we have little idea about the shape of the target distribution and, therefore, we need to meander through the parameter space. The opposite situation is one in which we have a pretty good idea about the target π and we are able to produce a credible approximation q which can be used as the proposal in the IS algorithm.

Modifications of these MH samplers include the delayed-rejection (12), multiple-try Metropolis (13; 14), and reversible-jump algorithms (15; 16), among others.

In practice, one must decide which sampler to use and, maybe more importantly, what simulation parameters values to choose. For instance, in the case of the RWM, the proposal covariance matrix Σ plays a crucial role in the performance of the sampling algorithm (17; 18).

2.1 Application to the Lupus Data

To see the effect of these choices in action, let us consider the lupus data under the PR model formulation. The target distribution has density (6). Since we have little idea of the shape of π_{PR}, it will be hard to come up with a good independence proposal distribution. Instead, we will use the RWM algorithm with a Gaussian proposal. We illustrate using two possible choices for the variance-covariance matrix Σ of the Gaussian proposal distribution: $\Sigma_1 = 0.6\,\mathbf{I}_3$ and $\Sigma_2 = 1.2\,\mathbf{I}_3$, where $\mathbf{I}_d$ is the identity matrix in $\mathbf{R}^{d\times d}$.

In Figure 1(a) we plot 5,000 samples for (β_0, β_1) that are obtained from the RWM with proposal variance Σ_1. The plot is superimposed on the two-dimensional projection of the contour plot for the density π_{PR}. The contour plot used is obtained from a large Monte Carlo sample produced by a state-of-the-art sampler and offers an accurate description of the target. The two red lines mark the coordinates of the initial value of the chain which has been chosen to be the maximum likelihood estimate for β. One can see that the samples do not cover the entire support of the distribution. Moreover, from the autocorrelation plots shown in Figure 1(b), we can see that the chain is very "sticky", i.e. there is strong dependence between the realizations of the chain, despite having an acceptance rate of 39% which, as we will see in Section 5, is usually considered reasonably high. One may be tempted to believe that the strong dependence between the Monte Carlo draws is due to the proposal variance being too small since sampling from a normal distribution with a small variance will result in draws close to the mean.

We consider doubling the variance and use $\Sigma_2 = 1.2\,\mathbf{I}_3$ as the proposal's covariance matrix. The larger variance brings the acceptance rate down to 24%. Figure 1(c) show the same plots as Figure 1(a) but for the sampler that uses Σ_2. The chain seems to travel further into the tails of the distribution, but the serial correlation remains extremely high. Such a high autocorrelation implies that the 5K Monte Carlo sample contains the same amount of information that would be contained in a much much

smaller sample of independent realizations. This reduction in *effective sample size* is computationally wasteful, since we spend a lot of time collecting samples that are essentially uninformative. In fact, under certain conditions (19) has shown that the asymptotic variance of $\hat{I}_M$ is σ^2/M, where

$$\sigma^2 = \mathrm{Var}_\pi\{g(\theta)\} + 2\sum_{k=1}^{\infty} \mathrm{cov}\{g(\theta_1), g(\theta_{k+1})\}, \tag{8}$$

which illustrates the importance of having small correlations between the successive draws θ_t.

The high autocorrelation between the successive draws can be explained if we consider the the strong posterior dependence between the parameters, as illustrated by Figure 2 where we have plotted the samples obtained in pairs. The plots provide an intuitive explanation for the poor mixing exhibited by the two RWM samplers, since their proposals have independent components and, therefore, deviate significantly from the target configuration.

We shall use these RWM algorithms for this lupus data to illustrate various MCMC theoretical considerations in Section 8.4.

3 Gibbs Sampler

The Gibbs sampler is an algorithm that was first used by (20) in the context of image restoration and, subsequently, (21) and (22) have recognized the algorithm's power for fitting statistical models. Assume that the vector of parameters $\theta \in \mathbf{R}^d$ is partitioned into s subvectors so that $\theta = (\eta_1, \ldots, \eta_s)$. Assume that the current state of the chain is $\theta^{(t)} = (\eta_1^{(t)}, \ldots, \eta_s^{(t)})$. The transition kernel for the Gibbs chain requires updating, *in turn*, each subvector by sampling it from its conditional distribution, given all the other subvectors. More precisely, step $t+1$ of the sampler involves the following updates:

$$\eta_1^{(t+1)} \quad \sim \quad \pi(\eta_1 | \eta_2^{(t)}, \ldots, \eta_s^{(t)})$$

$$
\begin{aligned}
\eta_2^{(t+1)} &\sim \pi(\eta_2|\eta_1^{(t+1)}, \eta_3^{(t)}, \ldots, \eta_s^{(t)}) \\
\ldots \quad &\ldots \quad \ldots \\
\eta_s^{(t+1)} &\sim \pi(\eta_s|\eta_1^{(t+1)}, \eta_2^{(t+1)}, \ldots, \eta_{s-1}^{(t+1)}).
\end{aligned} \tag{9}
$$

Cycling through the blocks in a fixed order defines the Gibbs sampler with *deterministic scan*; an alternative implementation involves a *random scan* in which the next block to be updated is sampled at random and each η_j has strictly positive probability to be updated. In general, it is not known whether the Gibbs sampler with random scan is more efficient than deterministic scan or not (23; 24; 25).

An obvious choice for the blocks η is obtained when $s = d$ and $\eta_j = \theta_j$ for $1 \leq j \leq d$. However, whenever possible it is recommended to have the blocks η containing as many individual components of θ as possible while being able to sample from the conditional distributions in (9) (see the analysis of 26).

3.1 Application to the Lupus Data

A direct implementation of the Gibbs sampler is not possible because, as can be seen from (6), the conditional distribution of β_j given the data and all the other parameters cannot be sampled directly. However, this difficulty dissolves once we expand the model specification to include auxiliary variables (see also 27). Specifically, for each $i \in \{1, \ldots, 55\}$, consider the latent variables $\psi_i \sim N(x_i^T\beta, 1)$, of which only the sign Y_i is observed i.e. $Y_i = \mathbf{1}(\psi_i > 0)$. Let X be the $n \times p$ matrix whose ith row is x_i and $\psi = (\psi_1, \ldots, \psi_n)$. After introducing ψ, we notice that the conditional distributions of $\beta|\psi, X$ and $\psi|\beta, Y$ can be sampled directly. Alternatively, sampling from the two conditional distributions will yield draws from the conditional distribution $p(\beta, \psi|X, Y)$ whose marginal, in β, is the target $\pi(\beta)$. The Monte Carlo approach makes the marginalization easy since all we need to do is drop the ψ's from the samples $\{(\beta_t, \psi_t); 1 \leq t \leq M\}$ drawn from $p(\beta, \psi|X, Y)$ and retain only the samples $\{\beta_t; 1 \leq t \leq m\}$ as draws from the target $\pi(\beta)$. This computational strategy of

expanding the model so that conditional distributions are available in closed form is known as the Data Augmentation (DA) algorithm (22).

The Gibbs sampler (or DA) for the lupus data alternates between sampling from

$$(I)\ \ \beta|\psi, X \sim N(\tilde{\beta}, (X^T X)^{-1}),$$

with $\tilde{\beta} = (X^T X)^{-1} X^T \psi$ and from

$$(II)\ \ \psi_i|\beta, Y_i \sim TN(x_i^T \beta, 1, Y_i),$$

where $TN(\mu, \sigma^2, Y)$ is $N(\mu, \sigma^2)$ truncated to be positive if $Y = 1$ and negative if $Y = 0$. In this formulation, $\eta_1 = (\beta_0, \beta_1, \beta_2)^T$ and $\eta_{j+1} = \psi_j$ for every $j = 1, \ldots, n$.

The Gibbs sampling does not require tuning and does not reject any of the samples produced. Despite these apparent advantages, the Gibbs sampler is not always preferred over the MH. For instance, in the probit (PR) model considered here, the chain is slowly moving across the parameter space. In Figure 3(a) we plot the trajectory of the chain for the first 300 samples when started at the MLE. The sluggishness suggested by Figure 3(a) is confirmed by the autocorrelation plots which show strong and persistent serial dependence for each parameter (Figure 3(b)). This is not necessarily a characteristic of Gibbs samplers. For the lupus data, the high posterior dependence between parameters makes it a difficult target, given that the Gibbs sampler will always attempt to move the chain in directions that are parallel to the coordinate axes.

The DA algorithms have been studied extensively due to their intensive use in statistical modelling, e.g. linear and non-linear mixed models and mixture models where the auxiliary latent variables are natural extensions of the model specification. (28) and (29) propose modified versions of the vanilla DA that are designed to break the serial dependence between the Monte Carlo samples and have the potential to drastically improve the mixing of the Markov chain. We refer the reader to (7) for a *marginal DA* implementation for the lupus data.

4 Variable-at-a-time Metropolis

It is possible to combine Metropolis-style moves with Gibbs-style variable-at-a-time, to create a *variable-at-a-time Metropolis algorithm*. (This algorithm is also sometimes called *Metropolis-within-Gibbs*, but actually this was the original form of the algorithm used by (10).)

Assume again that the vector of parameters $\theta \in \mathbf{R}^d$ is partitioned into s subvectors so that $\theta = (\eta_1, \ldots, \eta_s)$. Variable-at-a-time Metropolis then proceeds by proposing to move just one coordinate at a time (or a subset of coordinates), leaving all the other coordinates fixed. In its most common form, we might try to move the i^{th} coordinate by proposing a new state ω_{t+1}, where $\omega_{t+1,j} = \eta_{t,j}$ for all $j \neq i$, and where $\eta_{t,i} \sim N(\eta_{t,i}, \sigma^2)$. (Here $\omega_{t+1,j}$ is the j^{th} coordinate of ω_{t+1}, etc.) We then accept the proposal ω_{t+1} according to the Metropolis-Hastings rule (7).

As with the Gibbs sampler, we need to choose which coordinate to update each time, and again can proceed either by choosing coordinates in the sequence $1, 2, \ldots, d$, $1, 2, \ldots$ ("systematic-scan"), or by choosing the coordinate uniformly from $\{1, 2, \ldots, d\}$ each time ("random-scan"). (In this formulation, one systematic-scan iteration is roughly equivalent to d random-scan ones.)

The variable-at-a-time Metropolis algorithm is often a good generic choice, since unlike the full Metropolis algorithm it does not require moving all coordinates at once (which can be very challenging to do efficiently), and unlike Gibbs sampling it does not require being able to sample from the full conditional distributions (which could be infeasible).

4.1 Application to the Lupus Data

We now try using a componentwise RWM for updating each coordinate of β. Specifically, at time $t + 1$, for each coordinate h, the proposal $\omega_{t+1,h}$ is generated from

$N(\beta_{t,h}, \sigma_h^2)$ and is accepted with probability

$$\min\left\{1, \pi(\omega_{t+1,h}|\beta_{t,[-h]})/\pi(\beta_{t,h}|\beta_{t,[-h]})\right\}, \tag{10}$$

where $\beta_{t,[-h]}$ is the vector of the most recent updates for all the components of β except β_h. Note that the ratio involved in (10) is identical to $\pi(\omega_{t+1,h}, \beta_{t,[-h]})/\pi(\beta_{t,h}, \beta_{t,[-h]})$ and it can be computed in closed form since it is independent of any unknown normalizing constants.

We have implemented the algorithm using, $\sigma = (\sqrt{5}, 5, 2\sqrt{2})$. These values were chosen so that the acceptance rates *for each component* are between 20-25%. Figure 3(c) shows the samples obtained and Figure 3(d) presents the autocorrelation functions. Notice that the serial dependence is smaller than in the full MH implementation, but remains high. Also, the samples cover most of the support of the posterior density π. In the one-at-a-time implementation, we are no longer forcing all components of the chain to move together at the same time, and this seems to improve the spread of the resulting sample.

5 Optimising and Adapting the RWM Algorithm

Consider the RWM algorithm, with proposals $\omega_t = \theta_t + \epsilon_t$ where $\epsilon_t \sim N(0, \Sigma)$ (iid). Although this algorithm is very specific, it still allows for great flexibility in the choice of proposal convariance matrix Σ. This raises the question of what choice of Σ leads to the best performance of the algorithm, as we now discuss.

5.1 Optimal Scaling

We first note that if the elements on the diagonal of Σ are very small, then the proposals ω_t will usually be very close to the previous states θ_t. This means that the proposals will usually be accepted, but the chain will still hardly move at all, which is clearly sub-optimal. At the other extreme, if Σ is very large, then the proposals ω_t

will usually be very far from the previous states θ_t. This means (especially in high dimensions) that they are quite likely to be out in the tails of the target density π in at least one coordinate, and thus to have much lower π values. This implies that they will almost always be rejected, which is again clearly sub-optimal.

It follows that the optimal scaling is somewhere *in between* these two extremes. That is, we want our proposal scaling to be not too small, and not too large, but rather "just right" (this is sometimes called the *Goldilocks principle*).

In a pioneering paper, (17) took this a step further, proving that (in a certain idealised high-dimensional limit, at least) the optimal *acceptance rate* (i.e., limiting fraction of proposed moves which are accepted) is equal to the specific fraction 0.234, or about 23%. On the other hand, any acceptance rate between about 15% and 50% is still fairly efficient (see e.g. Figure 3 of (18)).

Later optimal scaling results obtained by (18) and (30) indicate that (again, in a certain idealised high-dimensional limit, at least) the optimal proposal covariance Σ should be chosen to be proportional to the true covariance matrix of the target distribution π (with the constant of proportionality chosen to achieve the 0.234 acceptance rate).

5.2 Adaptive MCMC

Unfortunately, one generally has little idea about the true covariance of π at the beginning of the simulation. This makes it difficult or impossible to directly apply the optimal scaling results of the previous section. One possible approach is to first perform a number of exploratory MCMC runs to get an idea about the geography of the target's importance regions, and then use this knowledge to tune the proposal to be approximately optimal. However, this approach requires re-starting the chain multiple times, with each run being used to tune different subsets of the simulation parameters. This process can be lengthy and onerous especially in high dimensional spaces, and generally has very limited success in complex models.

Alternatively, one can build upon the recent advances in Adaptive MCMC (AMCMC) where the proposal distribution is changed on the go and continuously at any time t using the information contained in the samples obtained up to that time (see e.g. 31; 32; 33; 34). Such an approach does not require re-starting of the chain, and can be made to be fully automatic. On the other hand, it requires careful theoretical analysis since, by using the past realizations of the chain (and not only the current state), the process loses its Markovian property and asymptotic ergodicity must be proven on a case-by-case basis. However, proving the validity of adaptive samplers has been made easier by the general frameworks developed by e.g. (35) and (36).

5.3 Application to the Lupus Data

We have implemented the adaptive RWM proposed by (31) (see also (32)) in which at each time $t > 1000$, we use for Σ in the Gaussian proposal the approximation

$$\Sigma_t = \frac{(2.4)^2}{3} SamVar_t + \epsilon \mathbf{I}_3, \tag{11}$$

where, $\epsilon = 0.01$ and $SamVar_t$ is the sample variance of all samples drawn up to time $t-1$. This is an attempt to approximately mimic the theoretical optimal scaling results discussed in Section 5.1 above; indeed if $SamVar_t$ happened to actually equal the true covariance matrix of π, and if $\epsilon = 0$, then (11) would indeed be the optimal proposal covariance. In Figure 5 we show the same plots as for Figures 1(a)-1(d). The reduction in serial autocorrelation is apparent. For instance, the mean, median, lower and upper quartiles for the autocorrelations computed up to lag 200 equal $(0.537, 0.513, 0.377, 0.664)$ for the RWM sampler with Σ_2, and equal the much smaller values $(0.065, 0.029, 0.007, 0.059)$ for the adaptive RWM.

6 Simulated Tempering

Particular challenges arise in MCMC when the target density π is multi-modal, i.e. has distinct high-probability regions which are separated by low-probability barriers which are difficult for the Markov chain to traverse. In such cases, it is often easy for a simple MCMC algorithm like RWM to explore well *within* any one of the modal regions, but it may take unfeasibly long for the chain to move *between* modes. This leads to extremely slow convergence, and very poor resulting estimates.

The idea of simulated tempering is to "flatten out" the distribution into related distributions which have less pronounced modes and hence can be more easily sampled. If done carefully, this flattening out can be compensated for, to ultimately yield good estimates for expected values from the original target density π, as we now explain.

Specifically, simulated tempering requires a *sequence* $\pi_1, \pi_2, \ldots, \pi_m$ of target densities, where $\pi_1 = \pi$ is the original density, and π_τ is flatter for the larger-τ distributions. (The parameter τ is usually referred to as the "temperature"; then π_1 is the "cold" density, and π_τ for larger τ are called the "hot" densities.) These different densities can then be combined to define a single joint density $\overline{\pi}$ on $\Theta \times \{1, 2, \ldots, m\}$, defined by $\overline{\pi}(\theta, \tau) = \frac{1}{m}\,\pi_\tau(\theta)$ for $1 \leq \tau \leq m$ and $\theta \in \Theta$. (It is also possible to use other weights besides the uniform choice $\frac{1}{m}$.)

Simulated tempering then uses $\overline{\pi}$ to define a joint Markov chain (θ, τ) on $\Theta \times \{1, 2, \ldots, m\}$, with target density $\overline{\pi}$. In the simplest case, this chain is a version of variable-at-a-time Metropolis which alternates (say) between spatial moves which propose (say) $\theta' \sim N(\theta, \sigma_\theta^2)$ and then accept with the usual Metropolis probability $\min\left(1,\ \frac{\overline{\pi}(\theta',\tau)}{\overline{\pi}(\theta,\tau)}\right) = \min\left(1,\ \frac{\pi_\tau(\theta')}{\pi_\tau(\theta)}\right)$, and temperature moves which propose (say) $\tau' = \tau \pm 1$ (prob $\frac{1}{2}$ each) and then accept with the usual Metropolis probability $\min\left(1,\ \frac{\overline{\pi}(\theta,\tau')}{\overline{\pi}(\theta,\tau)}\right) = \min\left(1,\ \frac{\pi_{\tau'}(\theta)}{\pi_\tau(\theta)}\right)$.

As usual for Metropolis algorithms, this chain should converge in distribution to

the density $\overline{\pi}$. But of course, our interest is in the original density $\pi = \pi_1$, not in $\overline{\pi}$. The genius of simulated tempering is that *in the end, we only "count" those samples corresponding to* $\tau = 1$. That is, once we have a good sample from $\overline{\pi}$, then we simply *discard* all the sample values corresponding to $\tau \neq 1$, and what remains is then a good sample from π, as we now explain.

6.1 A Simple Example

For a specific example, suppose the target density is given by $\pi(\theta) = \frac{1}{2}\,N(0, 1; \theta) + \frac{1}{2}\,N(20, 1; \theta)$, i.e. of a mixture of the standard normal density $N(0, 1; \theta)$ and the normal density $N(20, 1; \theta)$ with mean 20 and variance 1. This chain is highly multimodal (Figure 6(a)), leading to very poor mixing of ordinary RWM (Figure 7(a)).

On the other hand, if $\pi_\tau(\theta) = \frac{1}{2}\,N(0, \tau^2; \theta) + \frac{1}{2}\,N(20, \tau^2; \theta)$, i.e. a mixture of two normal densities with means 0 and 20 but now with variances τ^2 instead of 1, then $\pi_1 = \pi$ is the original target density, but π_τ gets flatter for larger τ (Figures 6(a), 6(b), 6(c)).

This allows us to define a joint simulated tempering chain on $\overline{\pi}$ (with proposal scaling $\sigma_\theta = 1$, say), and indeed that chain mixes much faster due to the flattened higher-temperature distributions (Figure 7(b)).

We can then identify the θ values corresponding to $\tau = 1$ in this faster-mixing joint chain, to get a very good sample from $\pi_1 = \pi$ (Figure 7(c)). Then, like with all Monte Carlo sampling, a good sample from π allows us to compute good estimates for expected values $I = \mathrm{E}(g)$ for functionals g of interest.

6.2 Choosing the Tempered Distributions

Simulated tempering often works quite well, but it raises the question of how to *find* appropriate tempered distributions π_τ. Usually we won't "know" convenient choices like $\pi_\tau = \frac{1}{2}\,N(0, \tau^2) + \frac{1}{2}\,N(20, \tau^2)$ above, so we require more generic choices.

One promising approach is to let the hotter densities $\pi_\tau(\theta)$ correspond to taking smaller and smaller *powers* of the original target density $\pi(\theta)$, i.e. to let $\pi_\tau(\theta) = c_\tau \left(\pi(\theta)\right)^{1/\tau}$ for appropriate normalising constant c_τ. (It is common to write $\beta = 1/\tau$, and refer to β as the *inverse temperature*.) This formula guarantees that $\pi_1 = \pi$, and that π_τ will be flatter for larger τ (since small positive powers move all positive numbers closer to 1), which is precisely what we need.

As a specific example, if it happened that $\pi(\theta)$ is the density of $N(\mu, \sigma^2)$, then $c_\tau(\pi(\theta))^{1/\tau}$ would be the density of $N(\mu, \tau\sigma^2)$. This is indeed a flatter density, similar to the simple example above, which confirms that this is a promising approach.

One problem with this approach is the following. With this formula, if we propose to move τ to τ', then we should accept this proposal with probability

$$\min\left(1,\ \frac{\pi_{\tau'}(\theta)}{\pi_\tau(\theta)}\right) \ = \ \min\left(1,\ \frac{c_{\tau'}}{c_\tau}(\pi(\theta))^{(1/\tau')-(1/\tau)}\right).$$

This formula explicitly depends on the normalising constants c_τ and $c_{\tau'}$, i.e. they do not "cancel" as for ordinary RWM. This is quite problematic since the c_τ are usually unknown and infeasible to calculate. So, what can be done?

6.3 Parallel Tempering

One idea is to use *parallel tempering*, also sometimes called *Metropolis-Coupled MCMC* (MCMCMC). In this algorithm, the state space is Θ^m, corresponding to m different chains each with its own value of θ. So, the state at time t is given by $\theta_t = (\theta_{t1}, \theta_{t2}, \ldots, \theta_{tm})$. The intuition is that each $\theta_{t\tau}$ is "at" its own temperature τ, i.e. converging towards its own target density π_τ. The overall target density is now $\overline{\pi}(\theta) = \pi_1(\theta_1)\ \pi_2(\theta_2) \ldots \pi_m(\theta_m)$, i.e. the density which makes each coordinate of θ independent and following its "own" temperature's density.

The algorithm can then update (for any $1 \leq \tau \leq m$) the chain $\theta_{t-1,\tau}$, at temperature τ, by proposing (say) $\theta'_{t,\tau} \sim N(\theta_{t-1,\tau},\ \sigma^2)$, and then accepting this proposal with the usual Metropolis probability $\min\left(1,\ \frac{\pi_\tau(\theta'_{t,\tau})}{\pi_\tau(\theta_{t-1,\tau})}\right)$.

Crucially, the chain can also choose temperatures τ and τ' (say, each chosen uniformly from $\{1, 2, \ldots, m\}$), and then propose to "swap" the values $\theta_{n,\tau}$ and $\theta_{t,\tau'}$. This proposal will then be accepted with its usual Metropolis probability, $\min\left(1, \frac{\pi_\tau(\theta_{t,\tau'})\,\pi_{\tau'}(\theta_{t,\tau})}{\pi_\tau(\theta_{t,\tau})\,\pi_{\tau'}(\theta_{t,\tau'})}\right)$. The beauty of parallel tempering is that now the normalising constants do indeed cancel. That is, if $\pi_\tau(\theta) = c_\tau\,(\pi(\theta))^{1/\tau}$, then the acceptance probability becomes:

$$\min\left(1,\ \frac{c_\tau \pi(\theta_{t,\tau'})^{1/\tau}\, c_{\tau'} \pi(\theta_{t,\tau})^{1/\tau'}}{c_\tau \pi(\theta_{t,\tau})^{1/\tau}\, c_{\tau'} \pi(\theta_{t,\tau'})^{1/\tau'}}\right) \;=\; \min\left(1,\ \frac{\pi(\theta_{t,\tau'})^{1/\tau}\, \pi(\theta_{t,\tau})^{1/\tau'}}{\pi(\theta_{t,\tau})^{1/\tau}\, \pi(\theta_{t,\tau'})^{1/\tau'}}\right).$$

So, the values of c_τ and $c_{\tau'}$ are not required to run the algorithm.

As a first test, we can apply parallel tempering to the above simple example, again with $\pi_\tau(\theta) = \frac{1}{2}\,N(0, \tau^2; \theta) + \frac{1}{2}\,N(20, \tau^2; \theta)$ for $\tau = 1, 2, \ldots, 10$. We see that parallel tempering again works pretty well in this case (Figure 7(d)).

Of course, in this simple example the normalising constants were known, so parallel tempering wasn't really required. However, in many applications the normalising constants are unknown, in which case parallel tempering is often a very useful sampling method.

7 Assessing MCMC Errors

When considering any statistical estimation procedure, an important issue is the amount of *uncertainty* in the estimate, e.g. some measure of its *standard error*. With conventional Monte Carlo as in (3) where the $\{\theta_i\}$ are iid, the standard error is of course given by $\frac{1}{\sqrt{M}}\,\widehat{\mathrm{SD}}(g(\theta))$, where $\widehat{\mathrm{SD}}(g(\theta))$ is the usual estimate of the standard deviation of the distribution of the $g(\theta_i)$. However, with MCMC there is usually extensive serial correlation in the samples θ_i, so the usual iid-based estimate of standard error does not apply (an exception is the class of so-called *perfect sampling algorithms*, see for example, (37) or (38), but they are hard to adapt for Bayesian computation). Indeed, the standard error for MCMC is usually both *larger* than in iid case (due to the correlations), and also harder to quantify.

The simplest way to estimate standard error from an MCMC estimate is to re-run the entire Markov chain over again, a number of times, using the exact same values of run length M and burn-in B as in (4), but started from different initial values θ_0 drawn from the same "overdispersed" (i.e., well spread-out) initial distribution. This leads to a sequence of iid estimates of the target expectation I, and standard errors from this sequence of estimates can then be computed in the usual iid manner. (We shall illustrate this in Section 8.4 below, for the RWM algorithms for the lupus data presented in Section 2.1 above.)

However, such a procedure is often too inefficient, leading to the question of how to estimate standard error from a single run of a single Markov chain. Specifically, we would like to estimate $v \equiv \mathrm{Var}\left(\frac{1}{M-B}\sum_{i=B+1}^{M} g(\theta_i)\right)$.

7.1 Variance Estimate

To estimate the variance of v above, let $\overline{g}(\theta) = g(\theta) - \mathrm{E}(g)$, so $\mathrm{E}(\overline{g}) = 0$. And, assume that B is large enough that $\theta_i \approx \pi$ for $i > B$. Then, writing $\approx$ to mean "equal in the limit as $M \to \infty$", we compute that:

$$
\begin{aligned}
v \;&\approx\; \mathrm{E}\Big[\Big(\Big(\frac{1}{M-B}\sum_{i=B+1}^{M} g(\theta_i)\Big) - \mathrm{E}(g)\Big)^2\Big] \;=\; \mathrm{E}\Big[\Big(\frac{1}{M-B}\sum_{i=B+1}^{M} \overline{g}(\theta_i)\Big)^2\Big] \\
&= \frac{1}{(M-B)^2}\Big[(M-B)\mathrm{E}(\overline{g}(\theta_i)^2) + 2(M-B-1)\mathrm{E}(\overline{g}(\theta_i)\overline{g}(\theta_{i+1})) \\
&\qquad\qquad +2(M-B-2)\mathrm{E}(\overline{g}(\theta_i)\overline{g}(\theta_{i+2})) + \ldots\Big] \\
&\approx \frac{1}{M-B}\Big(\mathrm{E}(\overline{g}(\theta_i)^2) + 2\,\mathrm{E}(\overline{g}(\theta_i)\overline{g}(\theta_{i+1})) + 2\,\mathrm{E}(\overline{g}(\theta_i)\overline{g}(\theta_{i+2})) + \ldots\Big) \\
&= \frac{1}{M-B}\Big(\mathrm{Var}_\pi(g) + 2\,\mathrm{Cov}_\pi(g(\theta_i)g(\theta_{i+1})) + 2\,\mathrm{Cov}_\pi(g(\theta_i)g(\theta_{i+2})) + \ldots\Big) \\
&= \frac{1}{M-B}\,\mathrm{Var}_\pi(g)\Big(1 + 2\,\mathrm{Corr}_\pi(g(\theta_i), g(\theta_{i+1})) + 2\,\mathrm{Corr}_\pi(g(\theta_i), g(\theta_{i+2})) + \ldots\Big) \\
&\equiv \frac{1}{M-B}\,\mathrm{Var}_\pi(g)(\mathrm{ACT}) \;=\; (\text{iid variance})\,(\mathrm{ACT})\,,
\end{aligned}
$$

where "iid variance" is the value for the variance that we *would* obtain if the samples $\{\theta_i\}$ were in fact iid, and

$$\text{ACT} \;=\; 1 + 2\sum_{k=1}^{\infty} \text{Corr}_\pi\Big(g(\theta_0), g(\theta_k)\Big) \;\equiv\; 1 + 2\sum_{k=1}^{\infty} \rho_k \;=\; \sum_{k=-\infty}^{\infty} \rho_k \;=\; 2\left(\sum_{k=0}^{\infty} \rho_k\right) - 1$$

is the factor by which the variance is multiplied due to the serial correlations from the Markov chain (sometimes called the "integrated auto-correlation time"). Here "Corr_π" means the theoretical correlation that would arise from a sequence $\{\theta_i\}_{i=-\infty}^{\infty}$ which was in stationarity (so each θ_i had density π) and which followed the Markov chain transitions; this in turn implies that the correlations are a function of the time lag between the two variables, and in particular $\rho_{-k} = \rho_k$ as above. The standard error is then, of course, given by $se = \sqrt{v} = (iid\text{-}se)\sqrt{\text{ACT}}$.

Now, both the iid variance, and the quantity ACT, can be estimated from the sample run. (For example, R's built-in function "acf" automatically computes the lag correlations ρ_k. Note also that when computing ACT in practice, we don't sum over *all* k, just until, say, $|\rho_k| < 0.05$ or $\rho_k < 0$, since for large k we should have $\rho_k \approx 0$ but the estimates of ρ_k will always contain some sampling error.) This provides a method of estimating the standard error of your sample. It also provides a method of comparing different MCMC algorithms, since usually $\text{ACT} \gg 1$, and "better" chains would have smaller values of ACT. In the most extreme case, one sometimes even tries to design "antithetic" chains for which $\text{ACT} < 1$ (see 8; 9; 39; 40; 41).

7.2 Confidence Intervals

Suppose we estimate $u \equiv \text{E}(g)$ by the quantity $e = \frac{1}{M-B}\sum_{i=B+1}^{M} g(\theta_i)$, and obtain an estimate e and an approximate variance (as above) v. Then what is, say, a 95% confidence interval for u?

Well, *if* a central limit theorem (CLT) applies (as discussed in Section 8 below), then for large $M - B$, we have the normal approximation that $e \approx N(u, v)$. It then

follows as usual that $(e-u)\,v^{-1/2} \approx N(0,1)$, so $\mathbf{P}(-1.96 < (e-u)\,v^{-1/2} < 1.96) \approx 0.95$, so $\mathbf{P}(-1.96\,\sqrt{v} < e-u < 1.96\,\sqrt{v}\,) \approx 0.95$. This gives us our desired confidence interval: with prob 95%, the interval $(e-1.96\,\sqrt{v},\ e+1.96\,\sqrt{v})$ will contain u. (Strictly speaking, we should use the "t" distribution, not normal distribution. But if $M-B$ is at all large, then that doesn't really matter, so we will ignore this issue for now.) Such confidence intervals allow us to more appropriately assess the uncertainty of our MCMC estimates (e.g. 42).

The above analysis raises the question of whether a CLT even holds in the Markov chain setting. This and other questions will be answered when we consider the *theory* of MCMC in the following section.

8 Theoretical Foundations of MCMC

We close with some theoretical considerations about MCMC. Specifically, why does MCMC work? The key is that the distribution of θ_n *converges* in various senses to the target distribution $\pi(\cdot)$. This follows from basic Markov chain theory, as we now discuss.

8.1 Markov Chain Convergence

A basic fact about Markov chains is that if a Markov chain is "irreducible" and "aperiodic", with "stationarity distribution" π, then θ_t converges in distribution to π as $t \to \infty$. More precisely we have the following (see, e.g. 43; 44; 45; 46):

Theorem. If a Markov chain is irreducible, with stationarity probability density π, then for π-a.e. initial value θ_0,
(a) if $g : \Theta \to \mathbf{R}$ with $\mathrm{E}(|g|) < \infty$, then $\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} g(\theta_i) = \mathrm{E}(g) \equiv \int g(\theta)\,\pi(\theta)\,d\theta$; and
(b) if the chain is also aperiodic, then furthermore $\lim_{t\to\infty} \mathbf{P}(\theta_t \in S) = \int_S \pi(\theta)\,d\theta$ for all

measurable $S \subseteq \Theta$.

We now discuss the various conditions of the theorem, one at a time.

Being *irreducible* means, essentially, that the chain has positive probability of *eventually* getting from anywhere to anywhere else. In the discrete case, we can define irreducibility as saying that for all $i, j \in \Theta$ there is $t \in \mathbf{N}$ such that $\mathbf{P}(\theta_t = j \,|\, \theta_0 = i) > 0$. In the general case, this definition won't do, since the probability of hitting any *particular* state is usually zero. Instead, we can define irreducibility (or, ϕ-irreducibility) as saying that there is *some* reference measure ϕ such that for all $\zeta \in \Theta$, and for all $A \subseteq \Theta$ with $\phi(A) > 0$, there is $t \in \mathbf{N}$ such that $\mathbf{P}(\theta_t \in A \,|\, \theta_0 = \zeta) > 0$. This condition is usually satisfied for MCMC (aside from certain rare cases where the state space consists of highly disjoint pieces), and is generally not a concern.

Being *aperiodic* means that there are no forced cycles, i.e. that there do *not* exist disjoint non-empty subsets $\Theta_1, \Theta_2, \ldots, \Theta_d \subseteq \Theta$ for some $d \geq 2$ such that $P(\theta_{t+1} \in \Theta_{i+1} \,|\, \theta_t = \zeta) = 1$ for all $\zeta \in \Theta_i$ $(1 \leq i \leq d-1)$, and $P(\theta_{t+1} \in \Theta_1 \,|\, \theta_t = \zeta) = 1$ for all $\zeta \in \Theta_d$. This virtually always holds for MCMC, e.g. it holds if $P(\theta_{t+1} = \zeta \,|\, \theta_t = \zeta) > 0$ as for the Metropolis algorithm (due to the positive probability of rejection), or if doing two iterations is sometimes equivalent to doing just one iteration as for the Gibbs sampler, or if the transition probabilities have positive densities throughout Θ as is often the case. In short, we have never known aperiodicity to be a problem for MCMC.

The condition that the density π be *stationary* for the chain is the most subtle one, as we discuss next.

8.2 Reversibility and Stationarity of Markov Chains

For ease of notation we focus in this section on discrete Markov chains, though the general case is similar upon replacing probability mass functions by measures and sums by integrals. We thus let π be a probability mass function on Θ, and assume

for simplicity that $\pi(\theta) > 0$ for all $\theta \in \Theta$. We also let $P(i,j) = \mathbf{P}(\theta_1 = j \,|\, \theta_0 = i)$ be the Markov chain's transition probabilities.

We say that π is *stationary* for the Markov chain if it is preserved under the chain's dynamics, i.e. if it has the property that whenever $\theta_0 \sim \pi$ (meaning that $\mathbf{P}(\theta_0 = i) = \pi(i)$ for all $i \in \Theta$), then also $\theta_1 \sim \pi$ (i.e., $\mathbf{P}(\theta_1 = i) = \pi(i)$ for all $i \in \Theta$). Equivalently, $\sum_{i \in \Theta} \pi(i)\, P(i,j) = \pi(j)$ for all $j \in \Theta$. Intuitively this means that the probabilities π are left invariant by the chain, which explains why the chain might perhaps converge to those probabilities in the limit.

We now show that reversibility is automatically satisfied by Metropolis-Hastings algorithms; indeed this explains why the Metropolis acceptance probabilities are defined as they are. Indeed, let $q(i,j) = \mathbf{P}(\omega_t = j \,|\, \theta_{t-1} = i)$ be the proposal distribution, which is then accepted with probability $\min\left(1,\ \frac{\pi(j)\, q(j,i)}{\pi(i)\, q(i,j)}\right)$. Then, for $i, j \in \Theta$ with $i \neq j$, we have that

$$P(i,j) \ = \ q(i,j) \ \min\left(1,\ \frac{\pi(j)\, q(j,i)}{\pi(i)\, q(i,j)}\right).$$

It follows that

$$\pi(i)\, P(i,j) \ = \ \pi(i)\, q(i,j) \ \min\left(1,\ \frac{\pi(j)\, q(j,i)}{\pi(i)\, q(i,j)}\right) \ = \ \min\left(\pi(i)\, q(i,j),\ \pi(j)\, q(j,i)\right).$$

By inspection, this last expression is symmetric in i and j. It follows that $\pi(i)\, P(i,j) = \pi(j)\, P(j,i)$ for all $i, j \in \Theta$ (at least for $i \neq j$, but also case $i = j$ is trivial). This property is described as π being *reversible* for the chain. (Intuitively, it implies that if $\theta_0 \sim \pi$, then $\mathbf{P}(\theta_0 = i,\ \theta_1 = j) = \mathbf{P}(\theta_0 = j,\ \theta_1 = i)$, i.e. we have the same probability of starting at i and then moving to j or vice-versa, which is also called being "time reversible".)

The importance of reversibility is that it in turn implies stationarity of π. Indeed, using reversibility, we compute that if $\theta_0 \sim \pi$, then:

$$\mathbf{P}(\theta_1 = j) \ = \ \sum_{i \in \Theta} \mathbf{P}(\theta_0 = i)\, P(i,j) \ = \ \sum_{i \in \Theta} \pi(i)\, P(i,j) \ = \ \sum_{i \in \Theta} \pi(j)\, P(j,i)$$

$$= \pi(j) \sum_{i \in \Theta} P(j,i) = \pi(j),$$

i.e. $\theta_1 \sim \pi$ too, so π is stationary.

We conclude that the stationarity condition holds automatically for any Metropolis-Hastings algorithm. Hence, assuming irreducibility and aperiodicity (which, as noted above, are virtually always satisfied for MCMC), the above Theorem applies and establishes the asymptotic validity of MCMC.

8.3 MCMC Convergence Rates

Write $P^t(\zeta, S) = \mathbf{P}[\theta_t \in S \,|\, \theta_0 = \zeta]$ for the t-step transition probabilities for the chain, and let $D(\zeta, t) = \|P^t(\zeta, \cdot) - \Pi(\cdot)\| \equiv \sup_{S \subseteq \Theta} |P^t(\zeta, S) - \Pi(S)|$ be a measure (specifically, the *total variation distance*) of the chain's distance from stationarity after t steps, where $\Pi(S) = \int_S \pi(\zeta)\, d\zeta$ is the target probability distribution. Then the chain is called *ergodic* if $\lim_{t \to \infty} D(\zeta, t) = 0$ for π-a.e. $\zeta \in \Theta$, i.e. if the chain transition probabilities $P^t(\zeta, S)$ converge (uniformly) to Π as $t \to \infty$, and indeed the above theorem indicates that this is usually the case for MCMC. However, ergodicity alone says nothing about how *quickly* this convergence occurs, i.e. what is the convergence *rate*.

By contrast, a *quantitative bound* on convergence is an actual number t^* such that $D(\zeta, t^*) < 0.01$, i.e. such that the chain's probabilities are within 0.01 of stationary after t^* iterations. (The cut-off value 0.01 is arbitrary, but has become fairly standard.) We then sometimes say that the chain "converges in t^* iterations". Quantitative bounds, when available, are the most useful, since they provide precise instructions about how long an MCMC algorithm must be run. Unfortunately they are often difficult to establish for complicated statistical models, though some progress has been made (e.g. 47; 48; 49; 50).

Halfway between these two extremes is *geometric ergodicity*, which is more useful than plain ergodicity, but often easier to compute than quantitative bounds. A chain

is geometrically ergodic if there is $\rho < 1$, and $M : \Theta \to [0, \infty]$ which is Π-a.e. finite, such that $D(\zeta, t) \leq M(\zeta)\, \rho^t$ for all $\zeta \in \Theta$ and $t \in \mathbf{N}$, i.e. such that the convergence to Π happens exponentially quickly.

One important fact is that if a Markov chain is geometrically ergodic, and if $g : \Theta \to \mathbf{R}$ such that $\mathrm{E}(|g|^{2+a}) < \infty$ for some $a > 0$, then a Central Limit Theorem (CLT) holds for quantities like $e = \frac{1}{M-B} \sum_{i=B+1}^{M} g(\theta_i)$ (44; 19), we have the normal approximation that $e \approx N(u, v)$. (In fact, if the Markov chain is *reversible* as above, then it suffices to take $a = 0$ (51).) As explained in Section 7.2 above, this is then key to obtaining confidence intervals and thus more reliable estimates.

Now, if the state space Θ is *finite*, then assuming irreducibility and aperiodicity, any Markov chain on Θ is *always* geometrically ergodic. However, on infinite state spaces this is not the case. The random-walk Metropolis (RWM) algorithm is known to be geometrically ergodic essentially (i.e., under a few mild technical conditions) if and only if π has exponential tails, i.e. there are $a, b, c > 0$ such that $\pi(\theta) \leq a e^{-b|\theta|}$ whenever $|\theta| > c$. (52; 53) And the Gibbs sampler is known to be geometrically ergodic for certain models (e.g. 54). But in some cases, geometric ergodicity can be difficult to ascertain.

In the absence of theoretical convergence bounds, it is difficult to ascertain whether the chain has reached stationarity or not. One option is to independently run some large number K of chains that have each been started from the same overdispersed starting distribution. If M and B are large enough, then we expect that the estimators provided by each chain to be in agreement. For mathematical formalization of this general principle see e.g. (55) and (56).

8.4 Convergence of RWM for the Lupus Data

We now illustrate some of the above ideas using the RWM algorithm for the lupus data, as presented in Section 2.1 above.

We consider running RWM for $M = 6000$ iterations, using a burn-in of $B =$

1000 iterations. We initialize the chain using draws from an overdispersed starting distribution centred at the MLE, by setting $\beta_{init} = \hat{\beta}_{MLE} + W$ where W is a vector of 3 iid random variables each generated from a Student distribution with 2 degrees of freedom.

We repeated this entire experiment a total of $K = 350$ times with proposal variance-covariance matrix $\Sigma_1 = 0.6\,\mathbf{I}_3$ (Figure 8), and another $K = 350$ times with proposal variance-covariance matrix $\Sigma_2 = 1.2\,\mathbf{I}_3$ (Figure 9). Inspection of the corresponding lists of estimates of the three β_i values illustrated in these figures shows that despite the wide overdispersed starting distributions, the resulting estimates are fairly closely concentrated around particular values (boxplots, top rows; histograms, bottom rows) showing fairly good convergence, and are approximately normally distributed (normal Q-Q plots, middle rows; histograms, bottom rows) showing an approximate CLT. Choosing larger values of M and B would be expected to result in even more concentrated values and more normal-looking distributions of the various estimates.

This brief experiment illustrates that even in the absence of theoretical convergence bounds, one can use multiple independent runs from overdispersed starting distributions to assess the convergence and accuracy and normality of MCMC estimates.

8.5 The Case of the Independence Sampler

When it comes to MCMC convergence rates, one case is particularly tractable, namely the independence sampler. Now, it is not surprising that, as long as an independence sampler's proposal satisfies that $q(\theta) > 0$ whenever $\pi(\theta) > 0$, irreducibility and aperiodicity and stationarity all follow easily, so the above Theorem immediately establishes ergodicity. But what is remarkable is that the independence sampler is geometrically ergodic *if and only if* there is $\delta > 0$ such that $q(\theta) \geq \delta\pi(\theta)$ for π-a.e. $\theta \in \Theta$, and furthermore in this case $D(\zeta, n) \leq (1-\delta)^n$ for π-a.e. $\zeta \in \Theta$ (52; 53). That is, for the independence sampler, we have an easy test for geometric ergodicity, and

a free quantitative bound thrown in.

For a simple specific example, consider an independence sampler on $\Theta = [0, \infty)$ with target density $\pi(\theta) = e^{-\theta}$. If the proposal density is, say, $q(\theta) = 0.01\, e^{-0.01\theta}$, then $q(\theta) \geq 0.01\, \pi(\theta)$ for all $\theta \in \Theta$, i.e. the above condition holds with $\delta = 0.01$, so the chain is geometrically ergodic with $D(\zeta, t) \leq (1-\delta)^t = (0.99)^t$ and hence converges in $t^* = 459$ iterations (since $(0.99)^{459} < 0.01$). By contrast, if $q(\theta) = 5e^{-5\theta}$, then the above condition does not hold for any value $\delta > 0$, so the chain is *not* geometrically ergodic, and in fact it has been shown (57) that in this case $4,000,000 \leq t^* \leq 14,000,000$, i.e. it takes at least four million iterations to converge. This illustrates how geometric ergodicity can sometimes make a tremendous difference between MCMC algorithms which converge efficiently and those which converge very poorly.

Acknowledgements. We thank Nancy Reid for encouraging us to write this review paper, and thank the anonymous referee for a very careful reading which led to many improvements.

References

[1] Spiegelhalter DJ, Best NG, Carlin BP, van der Linde A. 2002. Bayesian measures of model complexity and fit (with discussion). *Journal of the Royal Statistical Society, Series B* 64:583–639(57)

[2] Brooks S, Gelman A, Jones GL, Meng XL, eds. 2011. Handbook of Markov chain Monte Carlo. Chapman & Hall/CRC, Boca Raton, FL

[3] Chen MH, Shao QM, Ibrahim J. 2000. Monte Carlo methods in Bayesian computation. Springer Verlag

[4] Liu JS. 2001. Monte Carlo strategies in scientific computing. Springer

[5] Robert CP, Casella G. 2004. Monte Carlo statistical methods. Springer-Verlag New York Inc.

[6] Robert CP, Casella G. 2010. Introducing Monte Carlo methods with R. Use R! New York: Springer

[7] van Dyk D, Meng XL. 2001. The art of data augmentation (with discussion). *J. Comput. Graph. Statist.* **10**:1–111

[8] Craiu RV, Meng XL. 2005. Multi-process parallel antithetic coupling for forward and backward MCMC. *Ann. Statist.* 33:661–697

[9] Craiu RV, Lemieux C. 2007. Acceleration of the multiple-try Metropolis algorithm using antithetic and stratified sampling. *Statistics and Computing* 17:109–120

[10] Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller E. 1953. Equations of state calculations by fast computing machines. *J. Chem. Ph.* 21:1087–1092

[11] Hastings WK. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57:97–109

[12] Green P, Mira A. 2001. Delayed rejection in reversible jump Metropolis-Hastings. *Biometrika* 88:1035–1053

[13] Liu J, Liang F, Wong W. 2000. The multiple-try method and local optimization in Metropolis sampling. *Journal of the American Statistical Association* 95:121–134

[14] Casarin R, Craiu RV, Leisen F. 2013. Interacting multiple try algorithms with different proposal distributions. *Statistics and Computing* :to appear

[15] Green PJ. 1995. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika* 82:711–732

[16] Richardson S, Green PJ. 1997. On Bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society: Series B* 59:731–792

[17] Roberts GO, Gelman A, Wilks W. 1997. Weak convergence and optimal scaling of random walk Metropolis algorithms. *Ann. Appl. Probab.* 7:110–120

[18] Roberts GO, Rosenthal JS. 2001. Optimal scaling for various Metropolis-Hastings algorithms. *Statist. Sci.* 16:351–367

[19] Geyer CJ. 1992. Practical Markov chain Monte Carlo (with discussion). *Statistical Science* 7:473–483

[20] Geman S, Geman D. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6:721–741

[21] Gelfand AE, Smith AFM. 1992. Sampling-based approaches to calculating marginal densities. *J. Amer. Statist. Assoc.* 87:523–532

[22] Tanner MA, Wong WH. 1987. The calculation of posterior distributions by data augmentation. *J. Amer. Statist. Assoc.* 82:528–540

[23] Amit Y. 1991. On rates of convergence of stochastic relaxation for Gaussian and non-Gaussian distributions. *J. Multivariate Anal.* 38:82–100

[24] Amit Y. 1996. Convergence properties of the Gibbs sampler for perturbations of Gaussians. *Ann. Statist.* 24:122–140

[25] Liu JS, Wong WH, Kong A. 1995. Covariance structure and convergence rate of the Gibbs sampler with various scans. *JRSS-B* 57:157–169

[26] Liu JS, Wong WH, Kong A. 1994. Covariance structure of the Gibbs sampler with applications to the comparisons of estimators and augmentation schemes. *Biometrika* 81:27–40

[27] Albert J, Chib S. 1993. Bayesian analysis of binary and polychotomous response data. *JASA* 88:669–679

[28] Liu JS, Wu YN. 1999. Parameter expansion for data augmentation. *Journal of the American Statistical Association* 94:1264–1274

[29] Meng XL, van Dyk D. 1999. Seeking efficient data augmentation schemes via conditional and marginal augmentation. *Biometrika* 86:301–320

[30] Bedard M. 2006. On the robustness of optimal scaling for random walk Metropolis algorithms. Ph.D. thesis, Department of Statistics, University of Toronto

[31] Haario H, Saksman E, Tamminen J. 2001. An adaptive Metropolis algorithm. *Bernoulli* 7:223–242

[32] Roberts GO, Rosenthal JS. 2009. Examples of adaptive MCMC. *J. Comput. Graph. Statist.* 18:349–367

[33] Craiu RV, Rosenthal JS, Yang C. 2009. Learn from thy neighbor: Parallel-chain adaptive and regional MCMC. *Journal of the American Statistical Association* 104:1454–1466

[34] Bai Y, Craiu RV, Di Narzo A. 2011. Divide and conquer: A mixture-based approach to regional adaptation for MCMC. *J. Comput. Graph. Statist.* 20:63–79

[35] Andrieu C, Moulines E, Priouret P. 2005. Stability of stochastic approximation under verifiable conditions. *Siam Journal On Control and Optimization* 44:283–312

[36] Roberts GO, Rosenthal JS. 2007. Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. *J. Appl. Probab.* 44:458–475

[37] Propp JG, Wilson DB. 1996. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms* **9**:223–252

[38] Craiu RV, Meng XL. 2011. In *Handbook of Markov Chain Monte Carlo*, eds. S Brooks, A Gelman, G Jones, XL Meng. Chapman & Hall/CRC, Boca Raton, FL, 199–226

[39] Adler SL. 1981. Over-relaxation methods for the Monte Carlo evaluation of the partition function for multiquadratic actions. *Phys. Rev. D* 23:2901–2904

[40] Barone P, Frigessi A. 1990. Improving stochastic relaxation for Gaussian random fields. *Probab. Engrg. Inform. Sci.* 4:369–389

[41] Neal RM. 1995. Suppressing random walks in Markov chain Monte Carlo using ordered overrelaxation. Tech. Rep. 9508, University of Toronto

[42] Flegal J, Haran M, Jones G. 2008. Markov chain Monte Marlo: Can we trust the third significant figure? *Statist. Sci.* **23**:250–260

[43] Meyn SP, Tweedie RL. 1993. Markov Chains and Stochastic Stability. Communications and Control Engineering Series. London: Springer-Verlag

[44] Tierney L. 1994. Markov chains for exploring posterior distributions. *Ann. Statist.* 22:1701–1728

[45] Rosenthal JS. 2001. A review of asymptotic convergence for general state space Markov chains. *Far East J. Theor. Stat.* 5:37–50

[46] Roberts GO, Rosenthal JS. 2004. General state space Markov chains and MCMC algorithms. *Probab. Surv.* 1:20–71 (electronic)

[47] Rosenthal JS. 1995. Minorization conditions and convergence rates for Markov chain Monte Carlo. *J. Amer. Statist. Assoc.* 90:558–566

[48] Jones G, Hobert J. 2001. Honest exploration of intractable probability distributions via Markov chain Monte Carlo. *Statistical Science* 16:312–334

[49] Rosenthal JS. 2002. Quantitative convergence rates of Markov chains: a simple account. *Electron. Comm. Probab.* 7:123–128 (electronic)

[50] Douc R, Moulines E, Rosenthal J. 2004. Quantitative bounds on convergence of time-inhomogeneous Markov chains. *Annals of Applied Probability* 14:1643–1665

[51] Roberts GO, Rosenthal JS. 1997. Geometric ergodicity and hybrid Markov chains. *Electron. Comm. Probab.* 2:no. 2, 13–25 (electronic)

[52] Mengersen KL, Tweedie RL. 1996. Rates of convergence of the Hastings and Metropolis algorithms. *The Annals of Statistics* 24:101–121

[53] Roberts GO, Tweedie RL. 1996. Geometric convergence and central limit theorems for multidimensional Hastings and Metropolis algorithms. *Biometrika* 83:95–110

[54] Papaspiliopoulos O, Roberts GO. 2008. Stability of the Gibbs sampler for Bayesian hierarchical models. *Ann. Statist.* 36:95–117

[55] Gelman A, Rubin DB. 1992. Inference from iterative simulation using multiple sequences. *Statistical Science* Vol. 7:457–472

[56] Brooks SP, Gelman A. 1998. General methods for monitoring convergence of iterative simulations. *J. Comput. Graph. Statist.* 7:434–455

[57] Rosenthal JS, Roberts GO. 2011. Quantitative non-geometric convergence bounds for independence samplers. *Methodology and Computing In Applied Probability* 13:391–403

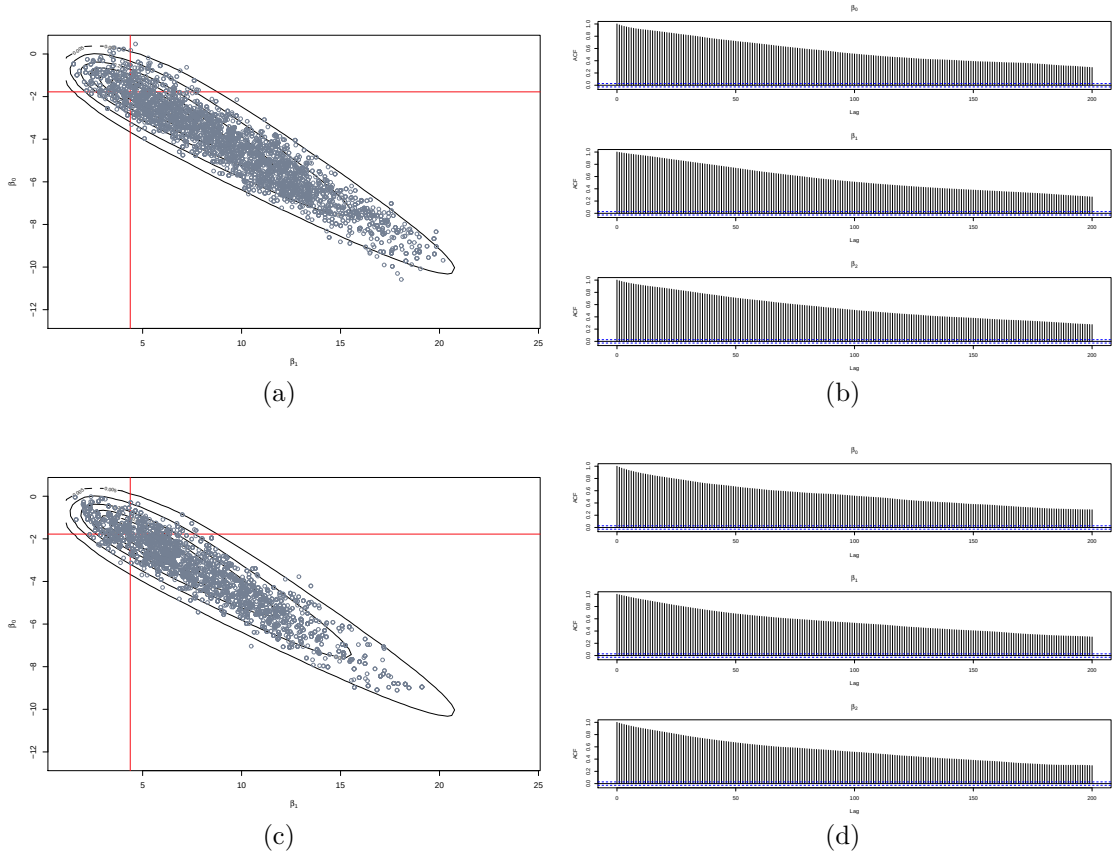


Figure 1: *Left panels: Scatterplots of 5,000 samples for* (β_0, β_1) *obtained using the RWM with proposal variance (a)* Σ_1 *and (c)* Σ_2*. The points are superimposed on the 2-dimensional projection of the contour plot for the target* π_{PR}*. Right panels: Autocorrelation plots for the three components of the chain for the RWM with proposal variance (b)* Σ_1 *and (d)* Σ_2*.*

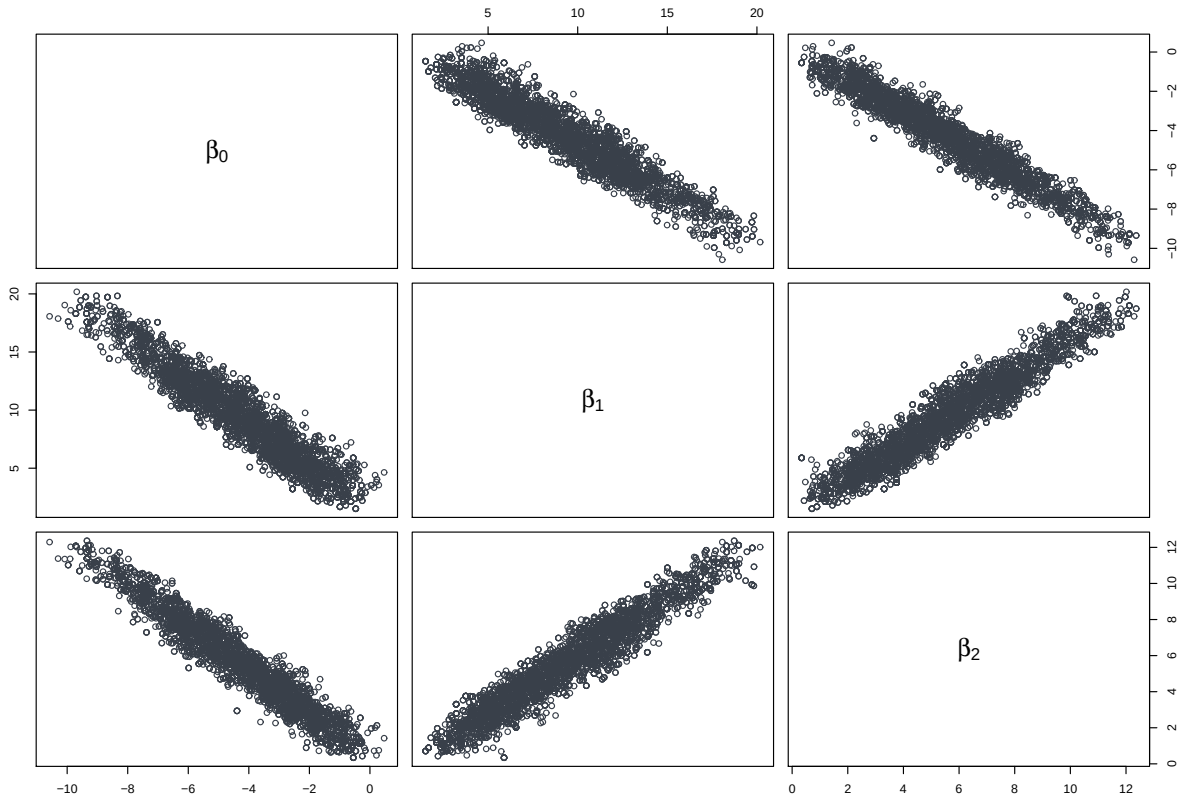


Figure 2: *Pair plots for the samples obtained using the RWM with proposal variance* Σ_2.

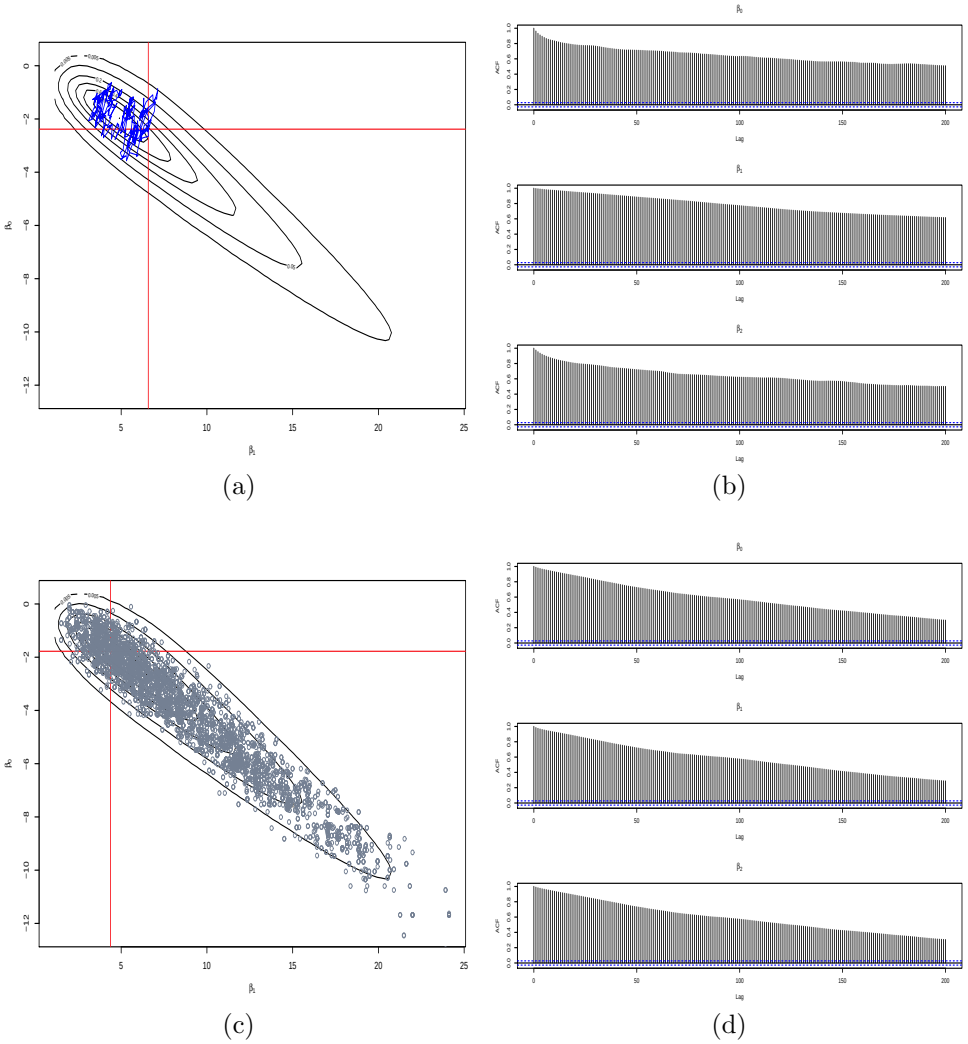
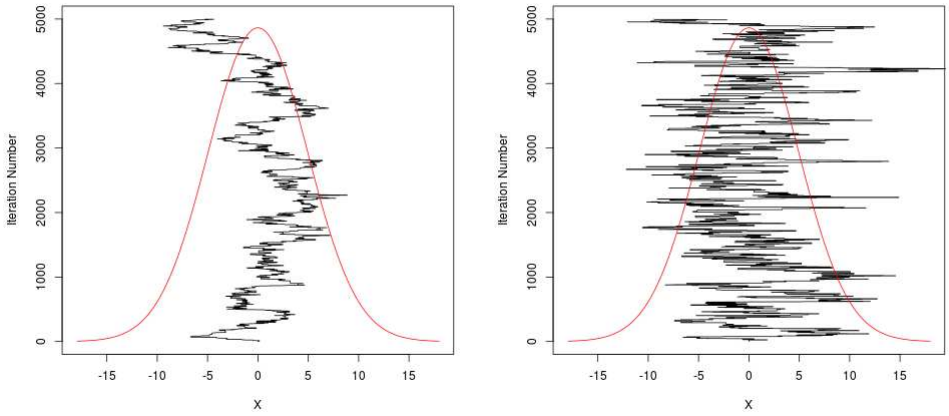


Figure 3: *Left panels: (a) Trajectory of the Gibbs chain for 300 updates for* (β_0, β_1) *(c) Scatterplots of 5,000 samples for* (β_0, β_1) *obtained using the variable-at-a-time MH. The points are superimposed on the 2-dimensional projection of the contour plot for the target* π_{PR}*. Right panels: Autocorrelation plots for the three components of the chain for (b) Gibbs sampler and (d) variable-at-a-time MH.*



(a) (b)

Figure 4: *Trace plots of the first coordinate of RWM, on the same 20-dimensional target, with acceptance rates both approximately 0.234, where the proposal covariance matrix* Σ *is proportional to either: (a) the identity* $\mathbf{I}_{20}$ *or (b) to the target covariance matrix. The run in(b) clearly mixes much faster.*

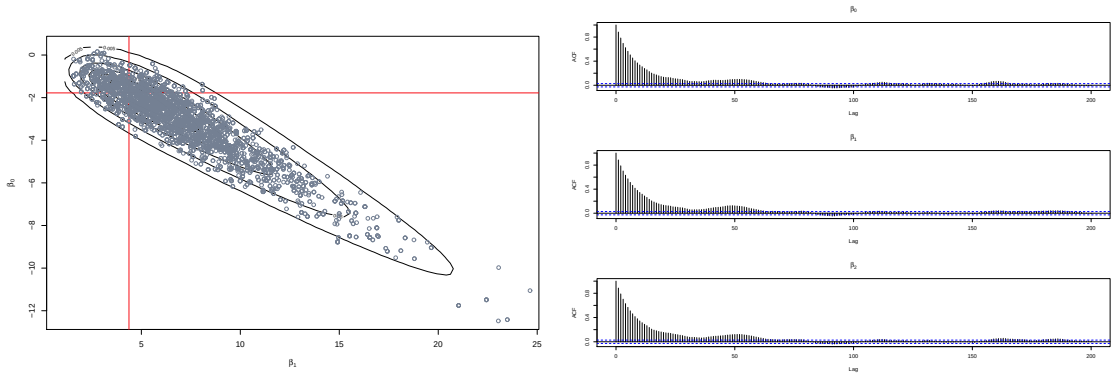
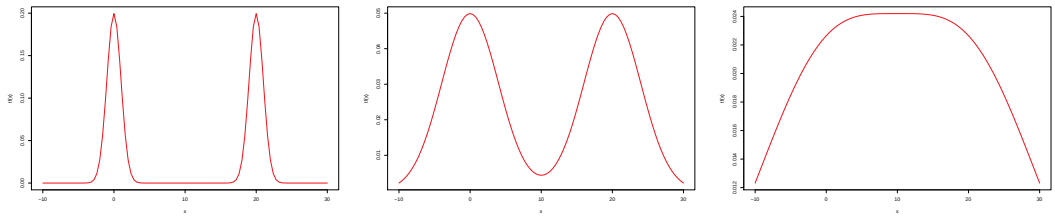


Figure 5: *Top left panel: Scatterplots of 30,000 samples for* (β_0, β_1) *obtained using the RWM with adaptive variance. The points are superimposed on the 2-dimensional projection of the contour plot for the target* π_{PR}*. Top, right panel: Autocorrelation plots for the three components of the chain show much lower serial dependence when compared with the non-adaptive RWM samplers.*



(a) (b) (c)

Figure 6: *(a) The highly multimodal target density* $\pi(\theta) = \frac{1}{2}\,N(0, 1; \theta) + \frac{1}{2}\,N(20, 1; \theta)$. *(b) A somewhat flatter density* $\pi_4 = \frac{1}{2}\,N(0, 4^2; \theta) + \frac{1}{2}\,N(20, 4^2; \theta)$. *(c) An even flatter density* $\pi_{10} = \frac{1}{2}\,N(0, 10^2; \theta) + \frac{1}{2}\,N(20, 10^2; \theta)$.

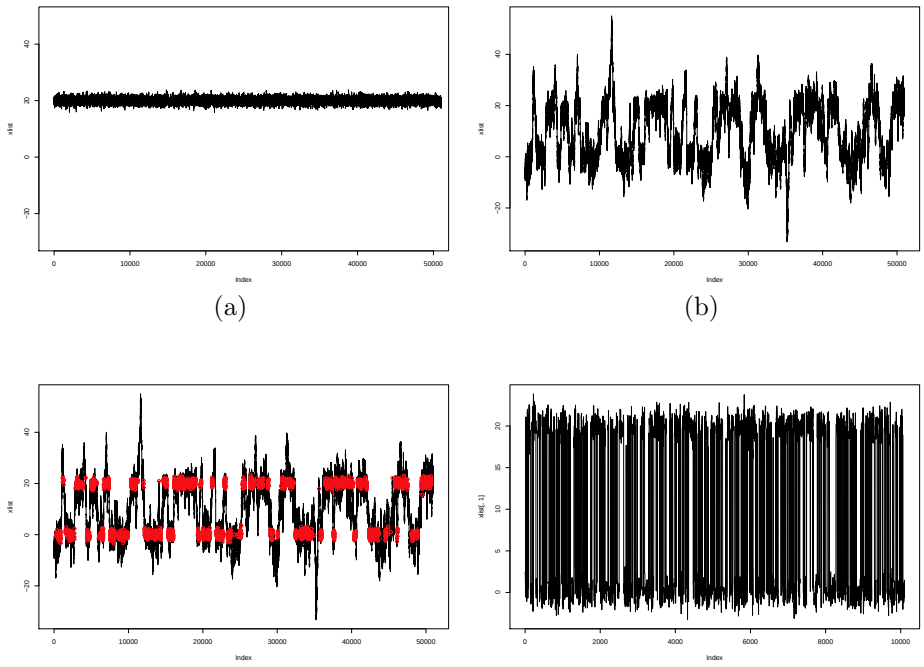


Figure 7: *Trace plots for the* $\pi(\theta) = \frac{1}{2}\,N(0,1;\theta) + \frac{1}{2}\,N(20,1;\theta)$ *example. (a) Ordinary RWM gets stuck in* π*'s modal region near 20, and cannot find the second modal region near 0. (b) The* θ *coordinate of simulated tempering for* $\overline{\pi}$*. (c) Identifying (red circles) the* θ *values of the simulated tempering corresponding to* $\tau = 1$ *(and hence to* π*). (d) The coordinate* θ_1 *for the corresponding parallel tempering algorithm, showing excellent mixing.*

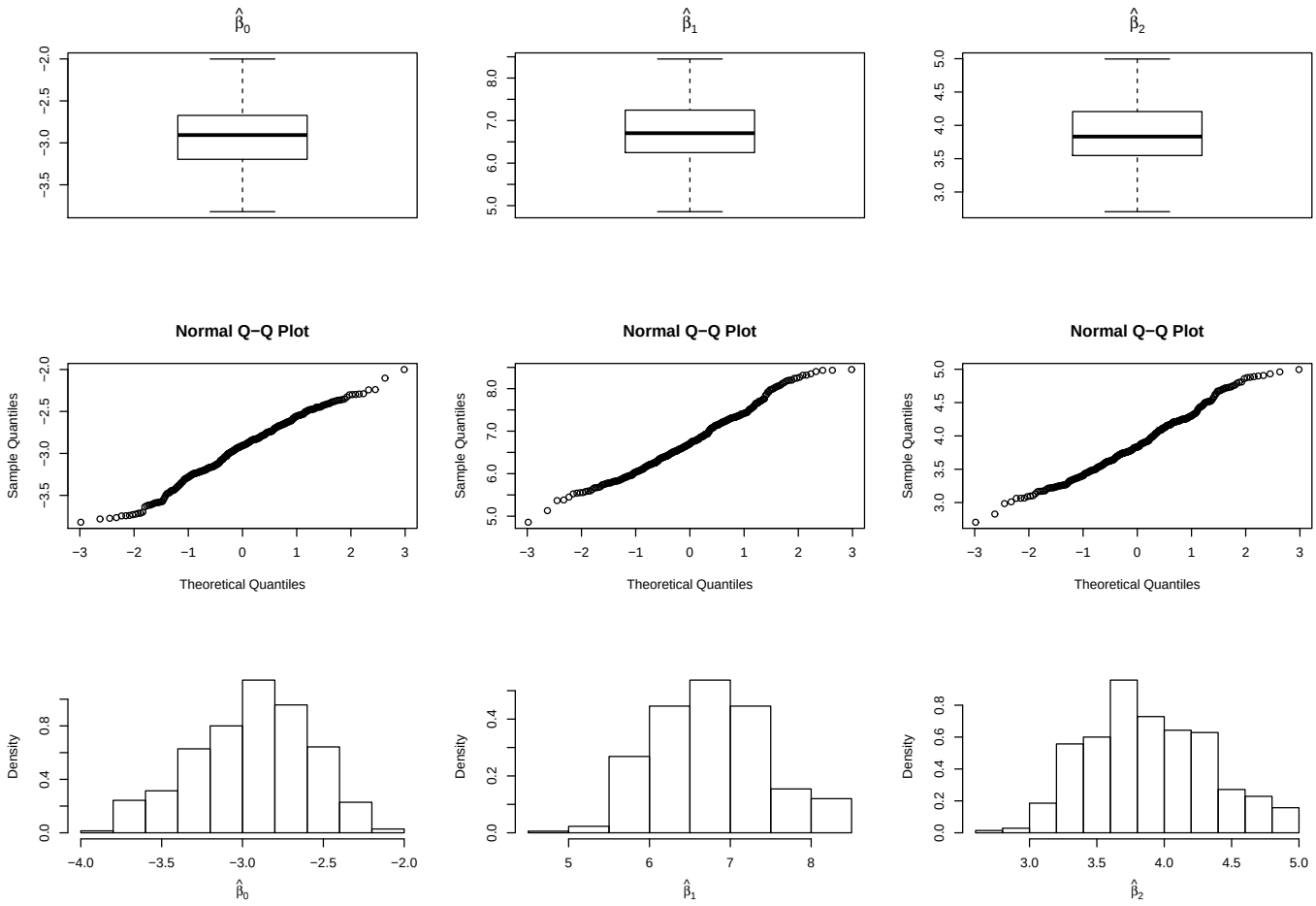


Figure 8: *Results of $K = 350$ independent replications of a RWM algorithm for the lupus data as in Section 2.1, with proposal variance-covariance matrix $\Sigma_1 = 0.6\,\mathbf{I}_3$, for estimates of the quantities β_0 (left column) and β_1 (middle column) and β_2 (right column), showing the resulting boxplot (top row) and normal Q-Q plot (middle row) and histogram (bottom row).*

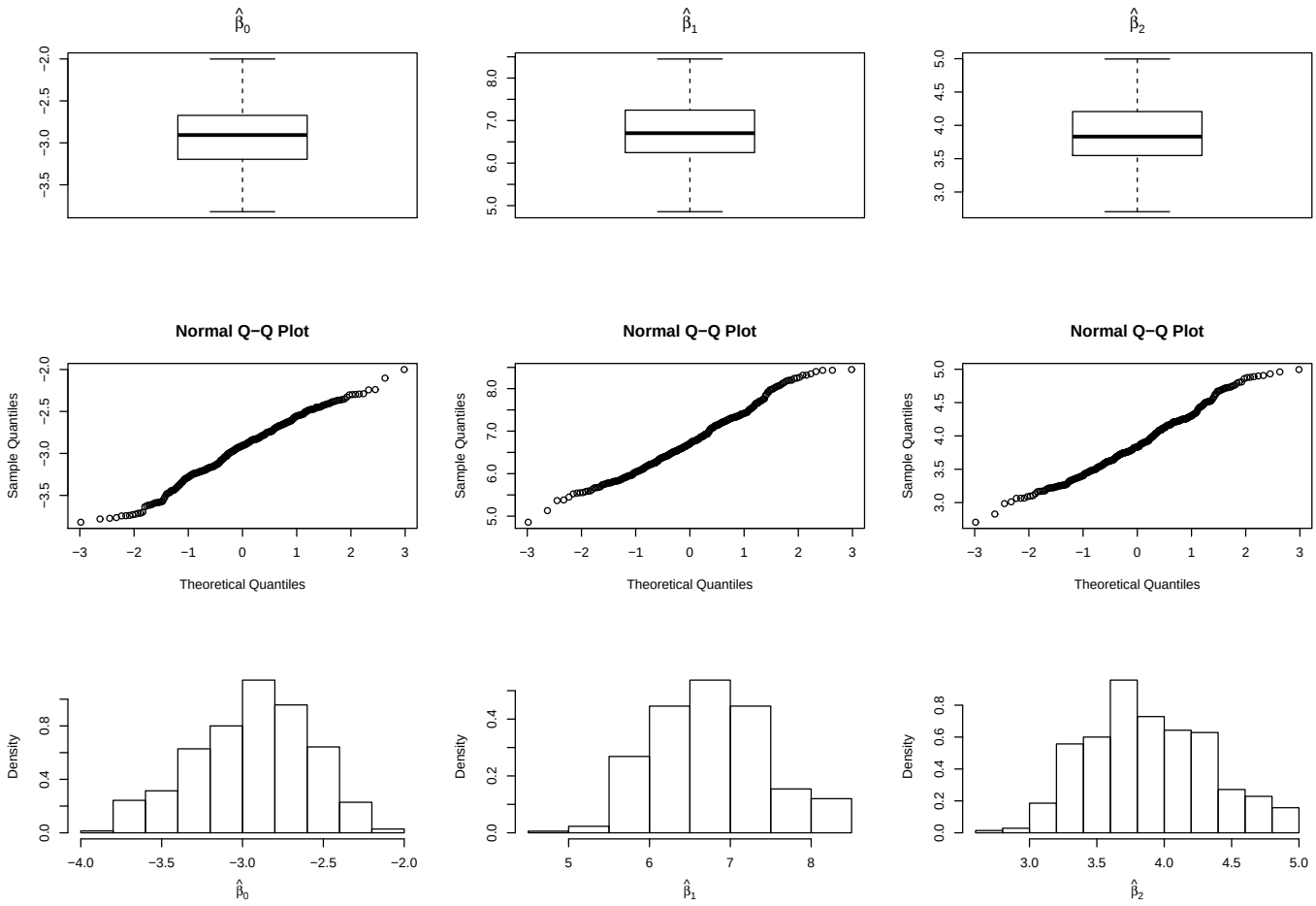


Figure 9: *Results of $K = 350$ independent replications of a RWM algorithm for the lupus data as in Section 2.1, with proposal variance-covariance matrix $\Sigma_2 = 1.2\,\mathbf{I}_3$, for estimates of the quantities β_0 (left column) and β_1 (middle column) and β_2 (right column), showing the resulting boxplot (top row) and normal Q-Q plot (middle row) and histogram (bottom row).*