Understanding the Metropolis–Hastings Algorithm

Siddhartha CHIB and Edward GREENBERG

We provide a detailed, introductory exposition of the Metropolis–Hastings algorithm, a powerful Markov chain method to simulate multivariate distributions. A simple, intuitive derivation of this method is given along with guidance on implementation. Also discussed are two applications of the algorithm, one for implementing acceptance–rejection sampling when a blanketing function is not available and the other for implementing the algorithm with block-at-a-time scans. In the latter situation, many different algorithms, including the Gibbs sampler, are shown to be special cases of the Metropolis–Hastings algorithm. The methods are illustrated with examples.

KEY WORDS: Gibbs sampling; Markov chain Monte Carlo; Multivariate density simulation; Reversible Markov chains.

1. INTRODUCTION

In recent years statisticians have been increasingly drawn to Markov chain Monte Carlo (MCMC) methods to simulate complex, nonstandard multivariate distributions. The Gibbs sampling algorithm is one of the best known of these methods, and its impact on Bayesian statistics, following the work of Tanner and Wong (1987) and Gelfand and Smith (1990), has been immense as detailed in many articles, for example, Smith and Roberts (1993), Tanner (1993), and Chib and Greenberg (1993). A considerable amount of attention is now being devoted to the Metropolis–Hastings (M–H) algorithm, which was developed by Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller (1953) and subsequently generalized by Hastings (1970). This algorithm is extremely versatile and gives rise to the Gibbs sampler as a special case, as pointed out by Gelman (1992). The M–H algorithm has been used extensively in physics, yet despite the paper by Hastings, it was little known to statisticians until recently. Papers by Müller (1993) and Tierney (1994) were instrumental in exposing the value of this algorithm and stimulating interest among statisticians in its use.

Because of the usefulness of the M–H alogrithm, applications are appearing steadily in the current literature (see Müller (1993), Chib and Greenberg (1994), and Phillips and Smith (1994) for recent examples). Despite its obvious importance, however, no simple or intuitive exposition of the M–H algorithm, comparable to that of Casella and George (1992) for the Gibbs sampler, is available. This article is an attempt to fill this gap. We provide a tutorial introduction to the algorithm, deriving the algorithm from first principles. The article is self-contained since it includes the relevant Markov chain theory, issues related to implementation and tuning, and empirical illustrations. We also discuss applications of the method, one for implementing acceptance–rejection sampling when a blanketing function is not available, developed by Tierney (1994), and the other for applying the algorithm one "block at a time." For the latter situation, we present an important principle that we call the product of kernels principle and explain how it is the basis of many other algorithms, including the Gibbs sampler. In each case we emphasize the intuition for the method and present proofs of the main results. For mathematical convenience, our entire discussion is phrased in the context of simulating an absolutely continuous target density, but the same ideas apply to discrete and mixed continuous-discrete distributions.

The rest of the article is organized as follows. In Section 2 we briefly review the acceptance–rejection (A–R) method of simulation. Although not an MCMC method, it uses some concepts that also appear in the Metropolis–Hastings algorithm and is a useful introduction to the topic. Section 3 introduces the relevant Markov chain theory for continuous state spaces, along with the general philosophy behind MCMC methods. In Section 4 we derive the M–H algorithm by exploiting the notion of reversibility defined in Section 3, and discuss some important features of the algorithm and the mild regularity conditions that justify its use. Section 5 contains issues related to the choice of the candidate-generating density and guidance on implementation. Section 6 discusses how the algorithm can be used in an acceptance–rejection scheme when a dominating density is not available. This section also explains how the algorithm can be applied when the variables to be simulated are divided into blocks. The final section contains two numerical examples, the first involving the simulation of a bivariate normal distribution, and the second the Bayesian analysis of an autoregressive model.

2. ACCEPTANCE–REJECTION SAMPLING

In contrast to the MCMC methods described below, classical simulation techniques generate non-Markov (usually independent) samples, that is, the successive observations are statistically independent unless correlation is artificially introduced as a variance reduction device. An important method in this class is the A–R method, which can be described as follows:

The objective is to generate samples from the absolutely continuous *target density* $\pi(x) = f(x)/K$, where $x \in \mathcal{R}^d$, $f(x)$ is the unnormalized density, and K is the (possibly unknown) normalizing constant. Let $h(x)$ be a density that can be simulated by some known method, and suppose there is a known constant c such that $f(x) \leq ch(x)$ for all x. Then, to obtain a random variate from $\pi(\cdot)$,

- (*) Generate a candidate Z from $h(\cdot)$ and a value u from $\mathcal{U}(0, 1)$, the uniform distribution on $(0, 1)$.

Siddhartha Chib is at the John M. Olin School of Business, Washington University, St. Louis, MO 63130. Edward Greenberg is at the Department of Economics, Washington University, St. Louis, MO 63130. The authors express their thanks to the editor of the journal, the associate editor, and four referees for many useful comments on the paper, and to Michael Ogilvie and Pin-Huang Chou for helpful discussions.

- If $u \le f(Z)/ch(Z)$
  —return $Z = y$.
- Else
  —goto (*).

It is easily shown that the accepted value y is a random variate from $\pi(\cdot)$. For this method to be efficient, c must be carefully selected. Because the expected number of iterations of steps 1 and 2 to obtain a draw is given by c^{-1}, the rejection method is optimized by setting

$$c = \sup_x \frac{f(x)}{h(x)}.$$

Even this choice, however, may result in an undesirably large number of rejections.

The notion of a generating density also appears in the M–H algorithm, but before considering the differences and similarities, we turn to the rationale behind MCMC methods.

3. MARKOV CHAIN MONTE CARLO SIMULATION

The usual approach to Markov chain theory on a continuous state space is to start with a transition kernel $P(x, A)$ for $x \in \mathcal{R}^d$ and $A \in \mathcal{B}$, where $\mathcal{B}$ is the Borel σ-field on $\mathcal{R}^d$. The transition kernel is a conditional distribution function that represents the probability of moving from x to a point in the set A. By virtue of its being a distribution function, $P(x, \mathcal{R}^d) = 1$, where it is permitted that the chain can make a transition from the point x to x, that is, $P(x, \{x\})$ is not necessarily zero.

A major concern of Markov chain theory [see Nummelin (1984), Billingsley (1986), Bhattacharya and Waymire (1990), and, especially, Meyn and Tweedie (1993)] is to determine conditions under which there exists an invariant distribution π^* and conditions under which iterations of the transition kernel converge to the invariant distribution. The invariant distribution satisfies

$$\pi^*(dy) = \int_{\mathcal{R}^d} P(x, dy)\pi(x)\,dx \tag{1}$$

where π is the density with respect to Lebesgue measure of π^* (thus $\pi^*(dy) = \pi(y)\,dy$). The nth iterate is given by $P^{(n)}(x, A) = \int_{\mathcal{R}^d} P^{(n-1)}(x, dy)P(y, A)$, where $P^{(1)}(x, dy) = P(x, dy)$. Under conditions discussed in the following, it can be shown that the nth iterate converges to the invariant distribution as $n \to \infty$.

MCMC methods turn the theory around: the invariant density is known (perhaps up to a constant multiple)—it is $\pi(\cdot)$, the *target* density from which samples are desired—but the transition kernel is unknown. To generate samples from $\pi(\cdot)$, the methods find and utilize a transition kernel $P(x, dy)$ whose nth iterate converges to $\pi(\cdot)$ for large n. The process is started at an arbitrary x and iterated a large number of times. After this large number, the distribution of the observations generated from the simulation is approximately the target distribution.

The problem then is to find an appropriate $P(x, dy)$. What might appear to be a search for the proverbial needle in a haystack is somewhat simplified by the following considerations. Suppose that the transition kernel, for some function $p(x, y)$, is expressed as

$$P(x, dy) = p(x, y)\,dy + r(x)\delta_x(dy), \tag{2}$$

where $p(x, x) = 0$, $\delta_x(dy) = 1$ if $x \in dy$ and 0 otherwise, and $r(x) = 1 - \int_{\mathcal{R}^d} p(x, y)\,dy$ is the probability that the chain remains at x. From the possibility that $r(x) \neq 0$, it should be clear that the integral of $p(x, y)$ over y is not necessarily 1.

Now, if the function $p(x, y)$ in (2) satisfies the reversibility condition (also called "detailed balance," "microscopic reversibility," and "time reversibility")

$$\pi(x)p(x, y) = \pi(y)p(y, x), \tag{3}$$

then $\pi(\cdot)$ is the invariant density of $P(x, \cdot)$ (Tierney 1994). To verify this we evaluate the right-hand side of (1):

$$\begin{aligned}
\int P(x, A)\pi(x)\,dx &= \int \left[\int_A p(x, y)\,dy\right]\pi(x)\,dx \\
&\quad + \int r(x)\delta_x(A)\pi(x)\,dx \\
&= \int_A \left[\int p(x, y)\pi(x)\,dx\right]dy \\
&\quad + \int_A r(x)\pi(x)\,dx \\
&= \int_A \left[\int p(y, x)\pi(y)\,dx\right]dy \\
&\quad + \int_A r(x)\pi(x)\,dx \\
&= \int_A (1 - r(y))\pi(y)\,dy + \int_A r(x)\pi(x)\,dx \\
&= \int_A \pi(y)\,dy. \qquad (4)
\end{aligned}$$

Intuitively, the left-hand side of the reversibility condition is the unconditional probability of moving from x to y, where x is generated from $\pi(\cdot)$, and the right-hand side is the unconditional probability of moving from y to x, where y is also generated from $\pi(\cdot)$. The reversibility condition says that the two sides are equal, and the above result shows that $\pi^*(\cdot)$ is the invariant distribution for $P(\cdot, \cdot)$.

This result gives us a sufficient condition (reversibility) that must be satisfied by $p(x, y)$. We now show how the Metropolis–Hastings algorithm finds a $p(x, y)$ with this property.

4. THE METROPOLIS–HASTINGS ALGORITHM

As in the A–R method, suppose we have a density that can generate candidates. Since we are dealing with Markov chains, however, we permit that density to depend on the current state of the process. Accordingly, the *candidate-generating density* is denoted $q(x, y)$, where $\int q(x, y)\,dy = 1$. This density is to be interpreted as saying that when a process is at the point x, the density generates a value y from $q(x, y)$. If it happens that $q(x, y)$ itself satisfies the reversibility condition (3) for all x, y, our search is over. But most likely it will not. We might find, for example, that for some x, y,

$$\pi(x)q(x, y) > \pi(y)q(y, x). \tag{5}$$

In this case, speaking somewhat loosely, the process moves from x to y too often and from y to x too rarely. A convenient way to correct this condition is to reduce the number of moves from x to y by introducing a probability

$\alpha(x, y) < 1$ that the move is made. We refer to $\alpha(x, y)$ as the *probability of move*. If the move is not made, the process again returns x as a value from the target distribution. (Note the contrast with the A–R method in which, when a y is rejected, a new pair (y, u) is drawn independently of the previous value of y.) Thus transitions from x to y $(y \neq x)$ are made according to

$$p_{MH}(x, y) \equiv q(x, y)\alpha(x, y), \qquad x \neq y,$$

where $\alpha(x, y)$ is yet to be determined.

Consider again inequality (5). It tells us that the movement from y to x is not made often enough. We should therefore define $\alpha(y, x)$ to be as large as possible, and since it is a probability, its upper limit is 1. But now the probability of move $\alpha(x, y)$ is determined by requiring that $p_{MH}(x, y)$ satisfies the reversibility condition, because then

$$\begin{aligned}\pi(x)q(x, y)\alpha(x, y) &= \pi(y)q(y, x)\alpha(y, x) \\ &= \pi(y)q(y, x). \qquad (6)\end{aligned}$$

We now see that $\alpha(x, y) = \pi(y)q(y, x)/\pi(x)q(x, y)$. Of course, if the inequality in (5) is reversed, we set $\alpha(x, y) = 1$ and derive $\alpha(y, x)$ as above. The probabilities $\alpha(x, y)$ and $\alpha(y, x)$ are thus introduced to ensure that the two sides of (5) are in balance or, in other words, that $p_{MH}(x, y)$ satisfies reversibility. Thus we have shown that in order for $p_{MH}(x, y)$ to be reversible, the probability of move must be set to

$$\begin{aligned}\alpha(x, y) &= \min\left[\frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}, 1\right], \qquad \text{if } \pi(x)q(x, y) > 0 \\ &= 1, \qquad \text{otherwise.}\end{aligned}$$

To complete the definition of the transition kernel for the Metropolis–Hastings chain, we must consider the possibly nonzero probability that the process remains at x. As defined above, this probability is

$$r(x) = 1 - \int_{\mathcal{R}^d} q(x, y)\alpha(x, y)\, dy.$$

Consequently, the transition kernel of the M–H chain, denoted by $P_{MH}(x, dy)$, is given by

$$\begin{aligned}P_{MH}(x, dy) &= q(x, y)\alpha(x, y)\, dy \\ &\quad + \left[1 - \int_{\mathcal{R}^d} q(x, y)\alpha(x, y)\, dy\right] \delta_x(dy),\end{aligned}$$

a particular case of (2). Because $p_{MH}(x, y)$ is reversible by construction, it follows from the argument in (4) that the M–H kernel has $\pi(x)$ as its invariant density.

Several remarks about this algorithm are in order. First, the M–H algorithm is specified by its candidate-generating density $q(x, y)$ whose selection we take up in the next section. Second, if a candidate value is rejected, the current value is taken as the next item in the sequence. Third, the calculation of $\alpha(x, y)$ *does not* require knowledge of the normalizing constant of $\pi(\cdot)$ because it appears both in the numerator and denominator. Fourth, if the candidate-generating density is symmetric, an important special case, $q(x, y) = q(y, x)$ and the probability of move reduces to $\pi(y)/\pi(x)$; hence, if $\pi(y) \geq \pi(x)$, the chain moves to y; otherwise, it moves with probability given by $\pi(y)/\pi(x)$. In other words, if the jump goes "uphill," it is always accepted; if "downhill," it is accepted with a nonzero probability. [See Fig. 1 where, from the current point x, a move to candidate y_1 is made with certainty, while a move to candidate y_2 is made with probability $\pi(y_2)/\pi(x)$.] This is the algorithm proposed by Metropolis et al. (1953). Interestingly, it also forms the basis for several optimization algorithms, notably the method of simulated annealing.

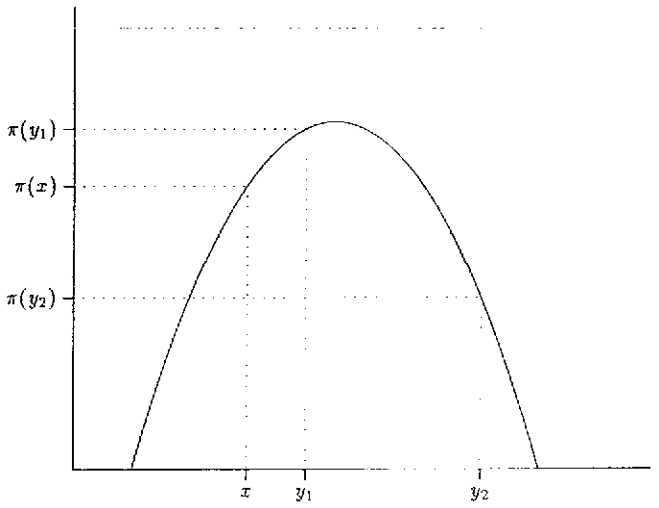

Figure 1. Calculating Probabilities of Move With Symmetric Candidate-Generating Function (see text).

We now summarize the M–H algorithm in algorithmic form initialized with the (arbitrary) value $x^{(0)}$:

- Repeat for $j = 1, 2, \ldots, N$.
- Generate y from $q(x^{(j)}, \cdot)$ and u from $\mathcal{U}(0, 1)$.
- If $u \leq \alpha(x^{(j)}, y)$
  —set $x^{(j+1)} = y$.
- Else
  —set $x^{(j+1)} = x^{(j)}$.
- Return the values $\{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\}$.

As in any MCMC method, the draws are regarded as a sample from the target density $\pi(x)$ only after the chain has passed the transient stage and the effect of the fixed starting value has become so small that it can be ignored. In fact, this convergence to the invariant distribution occurs under mild regularity conditions. The regularity conditions required are irreducibility and aperiodicity [see Smith and Roberts (1993)]. What these mean is that, if x and y are in the domain of $\pi(\cdot)$, it must be possible to move from x to dy in a finite number of iterations with nonzero probability, and the number of moves required to move from x to dy is not required to be a multiple of some integer. These conditions are usually satisfied if $q(x, y)$ has a positive density on the same support as that of $\pi(\cdot)$. It is usually also satisfied by a $q(x, y)$ with a restricted support (e.g., a uniform distribution around the current point with finite width).

These conditions, however, do not determine the rate of convergence [see Roberts and Tweedie (1994)], so there is an empirical question of how large an initial sample of size n_0 (say) should be discarded and how long the sampling should be run. One possibility, due to Gelman and Rubin (1992), is to start multiple chains from dispersed initial values and compare the within and between variation of the sampled draws. A simple heuristic that works in some situations is to make n_0 and N increasing functions of the first-order serial correlation in the output.

This entire area, however, is quite unsettled and is being actively researched. For more details the reader should consult Gelman and Rubin (1992) and the accompanying discussion.

5. IMPLEMENTATION ISSUES: CHOICE OF $q(x,y)$

To implement the M–H algorithm, it is necessary that a suitable candidate-generating density be specified. Typically, this density is selected from a family of distributions that requires the specification of such tuning parameters as the location and scale. Considerable recent work is being devoted to the question of how these choices should be made and, although the theory is far from complete, enough is known to conduct most practical simulation studies.

One family of candidate-generating densities, that appears in the work of Metropolis et al. (1953), is given by $q(x,y) = q_1(y - x)$, where $q_1(\cdot)$ is a multivariate density [see Müller (1993)]. The candidate y is thus drawn according to the process $y = x + z$, where z is called the increment random variable and follows the distribution q_1. Because the candidate is equal to the current value plus noise, this case is called a *random walk* chain. Possible choices for q_1 include the multivariate normal density and the multivariate-t with the parameters specified according to the principles described below. Note that when q_1 is symmetric, the usual circumstance, $q_1(z) = q_1(-z)$; the probability of move then reduces to

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

As mentioned earlier, the same reduction occurs if $q(x,y) = q(y,x)$.

A second family of candidate-generating densities is given by the form $q(x,y) = q_2(y)$ [see Hastings (1970)]. In contrast to the random walk chain, the candidates are drawn independently of the current location x—an *independence chain* in Tierney's (1994) terminology. As in the first case, we can let q_2 be a multivariate normal or multivariate-t density, but now it is necessary to specify the location of the generating density as well as the spread.

A third choice, which seems to be an efficient solution when available, is to exploit the known form of $\pi(\cdot)$ to specify a candidate-generating density [see Chib and Greenberg (1994)]. For example, if $\pi(t)$ can be written as $\pi(t) \propto \psi(t)h(t)$, where $h(t)$ is a density that can be sampled and $\psi(t)$ is uniformly bounded, then set $q(x,y) = h(y)$ (as in the independence chain) to draw candidates. In this case, the probability of move requires only the computation of the ψ function (not π or h) and is given by

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

A fourth method of drawing candidates is to use the A–R method with a *pseudo*dominating density. This method was developed in Tierney (1994), and because it is of independent interest as an M–H acceptance–rejection method, we explain it in Section 6.1.

A fifth family, also suggested by Tierney (1994), is represented by a vector autoregressive process of order 1. These *autoregressive chains* are produced by letting $y = a + B(x - a) + z$, where a is a vector and B is a matrix (both conformable with x) and z has q as its density. Then, $q(x,y) = q(y - a - B(x - a))$. Setting $B = -I$ produces chains that are reflected about the point a and is a simple way to induce negative correlation between successive elements of the chain.

We now return to the critical question of choosing the spread, or scale, of the candidate-generating density. This is an important matter that has implications for the efficiency of the algorithm. The spread of the candidate-generating density affects the behavior of the chain in at least two dimensions: one is the "acceptance rate" (the percentage of times a move to a new point is made), and the other is the region of the sample space that is covered by the chain. To see why, consider the situation in which the chain has converged and the density is being sampled around the mode. Then, if the spread is extremely large, some of the generated candidates will be far from the current value, and will therefore have a low probability of being accepted (because the ordinate of the candidate is small relative to the ordinate near the mode). Reducing the spread will correct this problem, but if the spread is chosen too small, the chain will take longer to traverse the support of the density, and low probability regions will be undersampled. Both of these situations are likely to be reflected in high autocorrelations across sample values.

Recent work by Roberts, Gelman, and Gilks (1994) discussed this issue in the context of q_1 (the random walk proposal density). They show that if the target and proposal densities are normal, then the scale of the latter should be tuned so that the acceptance rate is approximately .45 in one-dimensional problems and approximately .23 as the number of dimensions approaches infinity, with the optimal acceptance rate being around .25 in as low as six dimensions. This is similar to the recommendation of Müller (1993), who argues that the acceptance rate should be around .5 for the random walk chain.

The choice of spread of the proposal density in the case of q_2 (the independence proposal density) has also come under recent scrutiny. Chib and Geweke [work in progress] show that it is important to ensure that the tails of the proposal density dominate those of the target density, which is similar to a requirement on the importance sampling function in Monte Carlo integration with importance sampling [see Geweke (1989)]. It is important to mention the caveat that a chain with the "optimal" acceptance rate may still display high autocorrelations. In such circumstances it is usually necessary to try a different family of candidate-generating densities.

6. APPLICATIONS OF THE M–H ALGORITHM

We hope that the reader is now convinced that the M–H algorithm is a useful and straightforward device with which to sample an arbitrary multivariate distribution. In this section we explain two uses of the algorithm, one involving the A–R method, and the other for implementing the algorithm with block-at-a-time scans. In the latter situation many different algorithms, including the Gibbs sampler, are shown to arise as special cases of the M–H algorithm.

6.1 An M–H Acceptance–Rejection Algorithm

Recall that in the A–R method described earlier, a constant c and a density $h(x)$ are needed such that $ch(x)$ dominates or blankets the (possibly) unnormalized target density $f(x)$. Finding a c that does the trick may be difficult in some applications; moreover, if $f(x)$ depends on parameters that are revised during an iterative cycle, finding a new value of c for each new set of the parameters may significantly slow the computations. For these reasons it is worthwhile to have an A–R method that does not require a blanketing function. Tierney's (1994) remarkable algorithm does this by using an A–R step to generate candidates for an M–H algorithm. This algorithm, which seems complicated at first, can be derived rather easily using the intuition we have developed for the M–H algorithm.

To fix the context again: we are interested in sampling the target density $\pi(x)$, $\pi(x) = f(x)/K$, where K may be unknown, and a pdf $h(\cdot)$ is available for sampling. Suppose $c > 0$ is a known constant, but that $f(x)$ is not necessarily less than $ch(x)$ for all x; that is, $ch(x)$ does not necessarily dominate $f(x)$. It is convenient to define the set C where domination occurs:

$$C = \{x: f(x) < ch(x)\}.$$

In this algorithm, given $x^{(n)} = x$, the next value $x^{(n+1)}$ is obtained as follows: First, a candidate value z is obtained, *independent of the current value* x, by applying the A–R algorithm with $ch(\cdot)$ as the "dominating" density. The A–R step is implemented through steps 1 and 2 in Section 2.

What is the density of the rv y that comes through this step? Following Rubinstein (1981, pp. 45–46), we have

$$\begin{aligned} q(y) &= P(y \mid U \le f(Z)/ch(Z)) \\ &= \frac{P(U \le f(Z)/ch(Z) \mid Z = y) \times h(y)}{\Pr(U \le f(Z)/ch(Z))}. \end{aligned}$$

But because $P(U \le f(Z)/ch(Z) \mid Z = y) = \min\{f(y)/ch(y), 1\}$, it follows that

$$q(y) = \frac{\min\{f(y)/ch(y), 1\} \times h(y)}{d},$$

where $d \equiv \Pr(U \le f(Z)/ch(Z))$. By simplifying the numerator of this density we obtain a more useful representation for the candidate-generating density:

$$\begin{aligned} q(y) &= f(y)/cd, \qquad \text{if } y \in C \\ &= h(y)/d, \qquad \text{if } y \notin C. \end{aligned} \tag{7}$$

(Note that there is no need to write $q(x, y)$ for this density because the candidate y is drawn independently of x.)

Because $ch(y)$ does not dominate the target density in C^c (by definition), it follows that the target density is not adequately sampled there. See Figure 2 for an illustration of a nondominating density and the C region. This can be corrected with an M–H step applied to the y values that come through the A–R step. Since x and y can each be in C or in C^c, there are four possible cases: (a) $x \in C, y \in C$; (b) and (c) $x \notin C, y \in C$ or $x \in C, y \notin C$; and (d) $x \notin C$, $y \notin C$.

The objective now is to find the M–H moving probability $\alpha(x, y)$ such that $q(y)\alpha(x, y)$ satisfies reversibility.

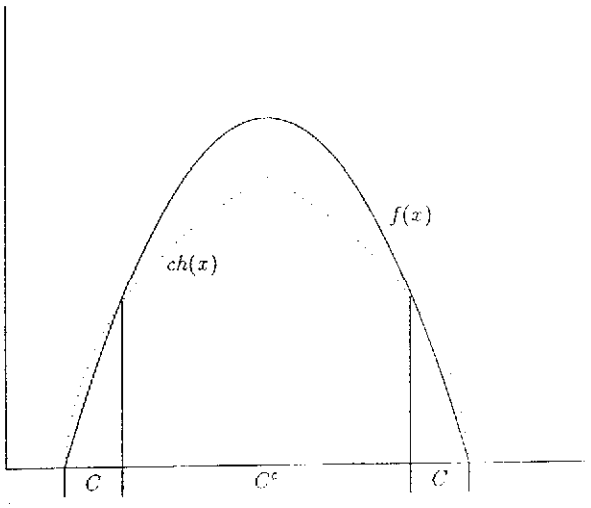

Figure 2. Acceptance–Rejection Sampling With Pseudodominating Density ch(x).

To proceed, we derive $\alpha(x, y)$ in each of the four possible cases given above. As in (2), we consider $\pi(x)q(y)$ and $\pi(y)q(x)$ [or, equivalently, $f(x)q(y)$ and $f(y)q(x)$] to see how the probability of moves should be defined to ensure reversibility. That is, we need to find $\alpha(x, y)$ and $\alpha(y, x)$ such that

$$f(x)q(y)\alpha(x, y) = f(y)q(x)\alpha(y, x)$$

in each of the cases (a)–(d), where $q(y)$ is chosen from (7).

Case (a): $x \in C$, $y \in C$. In this case it is easy to verify that $f(x)q(y) \equiv f(x)f(y)/cd$ is equal to $f(y)q(x)$. Accordingly, setting $\alpha(x, y) = \alpha(y, x) = 1$ satisfies reversibility.

Cases (b) and (c): $x \notin C$, $y \in C$ *or* $x \in C$, $y \notin C$. In the first case $f(x) > ch(x)$, or $h(x) < f(x)/c$, which implies (on multiplying both sides by $f(y)/d$) that

$$\frac{f(y)h(x)}{d} < \frac{f(y)f(x)}{cd},$$

or, from (7), $f(y)q(x) < f(x)q(y)$. We now see that there are relatively too few transitions from y to x and too many in the opposite direction. By setting $\alpha(y, x) = 1$ the first problem is alleviated, and then $\alpha(x, y)$ is determined from

$$\frac{f(y)h(x)}{d} = \alpha(x, y)\frac{f(x)f(y)}{cd}$$

which gives $\alpha(x, y) = ch(x)/f(x)$. If $x \in C$, $y \notin C$, reverse the roles of x and y above to find that $\alpha(x, y) = 1$ and $\alpha(y, x) = ch(y)/f(y)$.

Case (d): $x \notin C$, $y \notin C$. In this case we have $f(x)q(y) = f(x)h(y)/d$ and $f(y)q(x) = f(y)h(x)/d$, and there are two possibilities. There are too few transitions from y to x to satisfy reversibility if

$$\frac{f(x)h(y)}{d} > f(y)q(x).$$

In that case set $\alpha(y, x) = 1$ and determine $\alpha(x, y)$ from

$$\alpha(x, y)\frac{f(x)h(y)}{d} = \frac{f(y)h(x)}{d},$$

which implies

$$\alpha(x, y) = \min\left\{\frac{f(y)h(x)}{f(x)h(y)}, 1\right\}.$$

If there are too few transitions from x to y, just interchange x and y in the above discussion.

We thus see that in two of the cases, those where $x \in C$, the probability of move to y is 1, regardless of where y lies. To summarize, we have derived the following probability of move to the candidates y that are produced from the A–R step:

- Let $C1 = \{f(x) < ch(X)\}$; and $C2 = \{f(y) < ch(y)\}$.
- Generate u from $\mathcal{U}(0, 1)$ and
  - —if $C1 = 1$, then let $\alpha = 1$;
  - —if $C1 = 0$ and $C2 = 1$, then let $\alpha = (ch(x)/f(x))$;
  - —if $C1 = 0$ and $C2 = 0$, then let $\alpha = \min\{(f(y)h(x)/f(x)h(y),)1\}$.
- If $u \leq \alpha$
  - —return y.
- Else
  - —return x.

6.2 Block-at-a-Time Algorithms

Another interesting situation arises when the M–H algorithm is applied in turn to subblocks of the vector x, rather than simultaneously to all elements of the vector. This "block-at-a-time" or "variable-at-a-time" possibility, which is discussed in Hastings (1970, sec. 2.4), often simplifies the search for a suitable candidate-generating density and gives rise to several interesting hybrid algorithms obtained by combining M–H updates.

The central idea behind these algorithms may be illustrated with two blocks, $x = (x_1, x_2)$, where $x_i \in \mathbb{R}^{d_i}$. Suppose that there exists a conditional transition kernel $P_1(x_1, dy_1 \mid x_2)$ with the property that, for a fixed value of x_2, $\pi^*_{1|2}(\cdot \mid x_2)$ is its invariant distribution (with density $\pi_{12}(\cdot \mid x_2)$), that is,

$$\pi^*_{1|2}(dy_1 \mid x_2) = \int P_1(x_1, dy_1 \mid x_2)\pi_{1|2}(x_1 \mid x_2)\, dx_1. \quad (8)$$

Also, suppose the existence of a conditional transition kernel $P_2(x_2, dy_2 \mid x_1)$ with the property that, for a given x_1, $\pi^*_{2|1}(\cdot \mid x_1)$ is its invariant distribution, analogous to (8). For example, P_1 could be the transition kernel generated by a Metropolis–Hastings chain applied to the block x_1 with x_2 fixed for all iterations. Now, somewhat surprisingly, it turns out that the product of the transition kernels has $\pi(x_1, x_2)$ as its invariant density. The practical significance of this principle (which we call the *product of kernels principle*) is enormous because it allows us to take draws in succession from each of the kernels, instead of having to run each of the kernels to convergence for every value of the conditioning variable. In addition, as suggested above, this principle is extremely useful because it is usually far easier to find several conditional kernels that converge to their respective conditional densities than to find one kernel that converges to the joint.

To establish the product of kernels principle it is necessary to specify the nature of the "scan" through the elements of x (Hastings mentions several possibilities). Suppose the transition kernel $P_1(\cdot, \cdot \mid x_2)$ produces y_1 given x_1 and x_2, and the transition kernel $P_2(\cdot, \cdot \mid y_1)$ generates y_2 given x_2 and y_1. Then the kernel formed by multiplying the conditional kernels has $\pi^*(\cdot, \cdot)$ as its invariant distribution:

$$\begin{aligned}
&\iint P_1(x_1, dy_1 \mid x_2)P_2(x_2, dy_2 \mid y_1)\pi(x_1, x_2)\, dx_1\, dx_2 \\
&= \int P_2(x_2, dy_2 \mid y_1)\left[\int P_1(x_1, dy_1 \mid x_2)\pi_{1|2}(x_1 \mid x_2)\, dx_1\right] \\
&\quad \times \pi_2(x_2)\, dx_2 \\
&= \int P_2(x_2, dy_2 \mid y_1)\pi^*_{1|2}(dy_1 \mid x_2)\pi_2(x_2)\, dx_2 \\
&= \int P_2(x_2, dy_2 \mid y_1)\frac{\pi_{2|1}(x_2 \mid y_1)\pi^*_1(dy_1)}{\pi_2(x_2)}\pi_2(x_2)\, dx_2 \\
&= \pi^*_1(dy_1)\int P_2(x_2, dy_2 \mid y_1)\pi_{2|1}(x_2 \mid y_1)\, dx_2 \\
&= \pi^*_1(dy_1)\pi^*_{2|1}(dy_2 \mid y_1) \\
&= \pi^*(dy_1, dy_2),
\end{aligned}$$

where the third line follows from (8), the fourth from Bayes theorem, the sixth from assumed invariance of P_2, and the last from the law of total probability.

With this result in hand, several important special cases of the M–H algorithm can be mentioned. The first special case is the so-called "Gibbs sampler." This algorithm is obtained by letting the transition kernel $P_1(x_1, dy_1 \mid x_2) = \pi^*_{1|2}(dy_1 \mid x_2)$, and $P_2(x_2, dy_2 \mid y_1) = \pi^*_{2|1}(dy_2 \mid y_1)$, that is, the samples are generated directly from the "full conditional distributions." Note that this method requires that it be possible to generate independent samples from each of the full conditional densities. The calculations above demonstrate that this algorithm is a special case of the M–H algorithm. Alternatively, it may be checked that the M–H acceptance probability $\alpha(x, y) = 1$ for all x, y.

Another special case of the M–H algorithm is the so-called "M–H within Gibbs" algorithm (but see our comments on terminology below), in which an intractable full conditional density [say $\pi_{1|2}(y_1 \mid x_2)$] is sampled with the general form of the M–H algorithm described in Section 4 and the others are sampled directly from their full conditional distributions. Many other algorithms can be similarly developed that arise from multiplying conditional kernels.

We conclude this section with a brief digression on terminology. It should be clear from the discussion in this subsection that the M–H algorithm can take many different forms, one of which is the Gibbs sampler. Because much of the literature has overlooked Hastings's discussion of M–H algorithms that scan one block at a time, some unfortunate usage ("M–H within Gibbs," for example) has arisen that should be abandoned. In addition, it may be desirable to define the Gibbs sampler rather narrowly, as we have done above, as the case in which all full conditional kernels are sampled by independent algorithms in a fixed order. Although a special case of the M–H algorithm, it is an extremely important special case.

7. EXAMPLES

We next present two examples of the use of the M–H algorithm. In the first we simulate the bivariate normal to illustrate the effects of various choices of $q(x, y)$; the

second example illustrates the value of setting up blocks of variables in the Bayesian posterior analysis of a second-order autoregressive time series model.

7.1 Simulating a Bivariate Normal

To illustrate the M–H algorithm we consider the simulation of the bivariate normal distribution $\mathcal{N}_2(\mu, \Sigma)$, where $\mu = (1, 2)'$ is the mean vector and $\Sigma = (\sigma_{ij})$: 2×2 is the covariance matrix given by

$$\Sigma = \begin{pmatrix} 1 & .9 \\ .9 & 1 \end{pmatrix}.$$

Because of the high correlation the contours of this distribution are "cigar-shaped," that is, thin and positively inclined. Although this distribution can be simulated directly in the Choleski approach by letting $y = \mu + P'u$, where $u \sim \mathcal{N}_2(0, I_2)$ and P satisfies $P'P = \Sigma$, this well-known problem is useful for illustrating the M–H algorithm.

From the expression for the multivariate normal density, the probability of move (for a symmetric candidate-generating density) is

$$\alpha(x, y) = \min\left\{\frac{\exp\left[-\frac{1}{2}(y-\mu)'\Sigma^{-1}(y-\mu)\right]}{\exp\left[-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)\right]}, 1\right\}, \quad x, y \in \mathcal{R}^2. \quad (9)$$

We use the following candidate-generating densities, for which the parameters are adjusted by experimentation to achieve an acceptance rate of 40% to 50%:

1. Random walk generating density ($y = x + z$), where the increment random variable z is distributed as bivariate uniform, that is, the ith component of z is uniform on the interval $(-\delta_i, \delta_i)$. Note that δ_1 controls the spread along the first coordinate axis and δ_2 the spread along the second. To avoid excessive moves we let $\delta_1 = .75$ and $\delta_2 = 1$.
2. Random walk generating density ($y = x + z$) with z distributed as independent normal $\mathcal{N}_2(0, D)$, where $D =$ diagonal(.6, .4).
3. Pseudorejection sampling generating density with "dominating function" $ch(x) = c(2\pi)^{-1}|D|^{-1/2} \exp[-\frac{1}{2}(x-\mu)'D(x-\mu)]$, where $D =$ diagonal(2, 2) and $c = .9$. The trial draws, which are passed through the A–R step, are thus obtained from a bivariate, independent normal distribution.
4. The autoregressive generating density $y = \mu - (x - \mu) + z$, where z is independent uniform with $\delta_1 = 1 = \delta_2$. Thus values of y are obtained by reflecting the current point around μ and then adding the increment.

Note that the probability of move in cases 1, 2, and 4 is given by (9). In addition, the first two generating densities do not make use of the known value of μ, although the values of the δ_i are related to Σ. In the third generating density we have set the value of the constant c to be smaller than that which leads to true domination. For domination it is necessary to let all diagonal entries of D be equal to 1.9 (the largest eigenvalue of Σ) and to set $c = \sqrt{|D|/|\Sigma|}$ [see Dagpunar (1988, p. 159)].

Each of these four candidate-generating densities reproduces the shape of the bivariate normal distribution being simulated, although overall the best result is obtained from the fourth generating density. To illustrate the characteristics of the output, the top panel of Figure 3 contains the scatter plot of $N = 4{,}000$ simulated values from the Choleski approach and the bottom panel the scatter plot of $N = 6{,}000$ simulated values using the fourth candidate-generating density. More observations are taken from the M–H algorithm to make the two plots comparable. The plots of the output with the other candidate-generating densities are similar to this and are therefore omitted. At the suggestion of a referee, points that repeat in the M–H chain are "jittered" to improve clarity. The figure clearly reveals that the sampler does a striking job of visiting the entire support of the distribution. This is confirmed by the estimated tail probabilities computed from the M–H output for which the estimates are extremely close to the true values. Details are not reported to save space.

For the third generating density we found that reductions in the elements of D led to an erosion in the number of times the sampler visited the tails of the distribution. In addition, we found that the first-order serial correlation of the sampled values with the first and second candidate-generating densities is of the order .9, and with the other two it is .30 and .16, respectively. The high serial correlation with the random walk generating densities is not unexpected and stems from the long memory in the candidate draws. Finally, by reflecting the candidates we see that it is possible to obtain a beneficial reduction in the serial correlation of the output with little cost.

7.2 Simulating a Bayesian Posterior

We now illustrate the use of the M–H algorithm to sample an intractable distribution that arises in a stationary second-order autoregressive [AR(2)] time series model. Our presentation is based on Chib and Greenberg (1994), which contains a more detailed discussion and results for the general ARMA(p, q) model.

For our illustration, we simulated 100 observations from the model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \epsilon_t, \qquad t = 1, 2, \ldots, 100, \quad (10)$$

where $\phi_1 = 1$, $\phi_2 = -.5$, and $\epsilon_t \sim N(0, 1)$. The values of $\phi = (\phi_1, \phi_2)$ lie in the region $S \subset \mathcal{R}^2$ that satisfies the stationarity restrictions

$$\phi_1 + \phi_2 < 1; \qquad -\phi_1 + \phi_2 < 1; \qquad \phi_2 > -1.$$

Following Box and Jenkins (1976), we express the (*exact* or *unconditional*) likelihood function for this model given the $n = 100$ data values $Y_n = (y_1, y_2, \ldots, y_n)'$ as

$$l(\phi, \sigma^2) = \Psi(\phi, \sigma^2) \times (\sigma^2)^{-(n-2)/2} \times \exp\left[-\frac{1}{2\sigma^2}\sum_{t=3}^{n}(y_t - w_t'\phi)^2\right], \quad (11)$$

where $w_t = (y_{t-1}, y_{t-2})'$,

$$\Psi(\phi, \sigma^2) = (\sigma^2)^{-1}|V^{-1}|^{1/2} \exp\left[-\frac{1}{2\sigma^2}Y_2'V^{-1}Y_2\right] \quad (12)$$

is the density of $Y_2 = (y_1, y_2)'$,

$$V^{-1} = \begin{pmatrix} 1-\phi_2^2 & -\phi_1(1+\phi_2) \\ -\phi_1(1+\phi_2) & 1-\phi_2^2 \end{pmatrix},$$

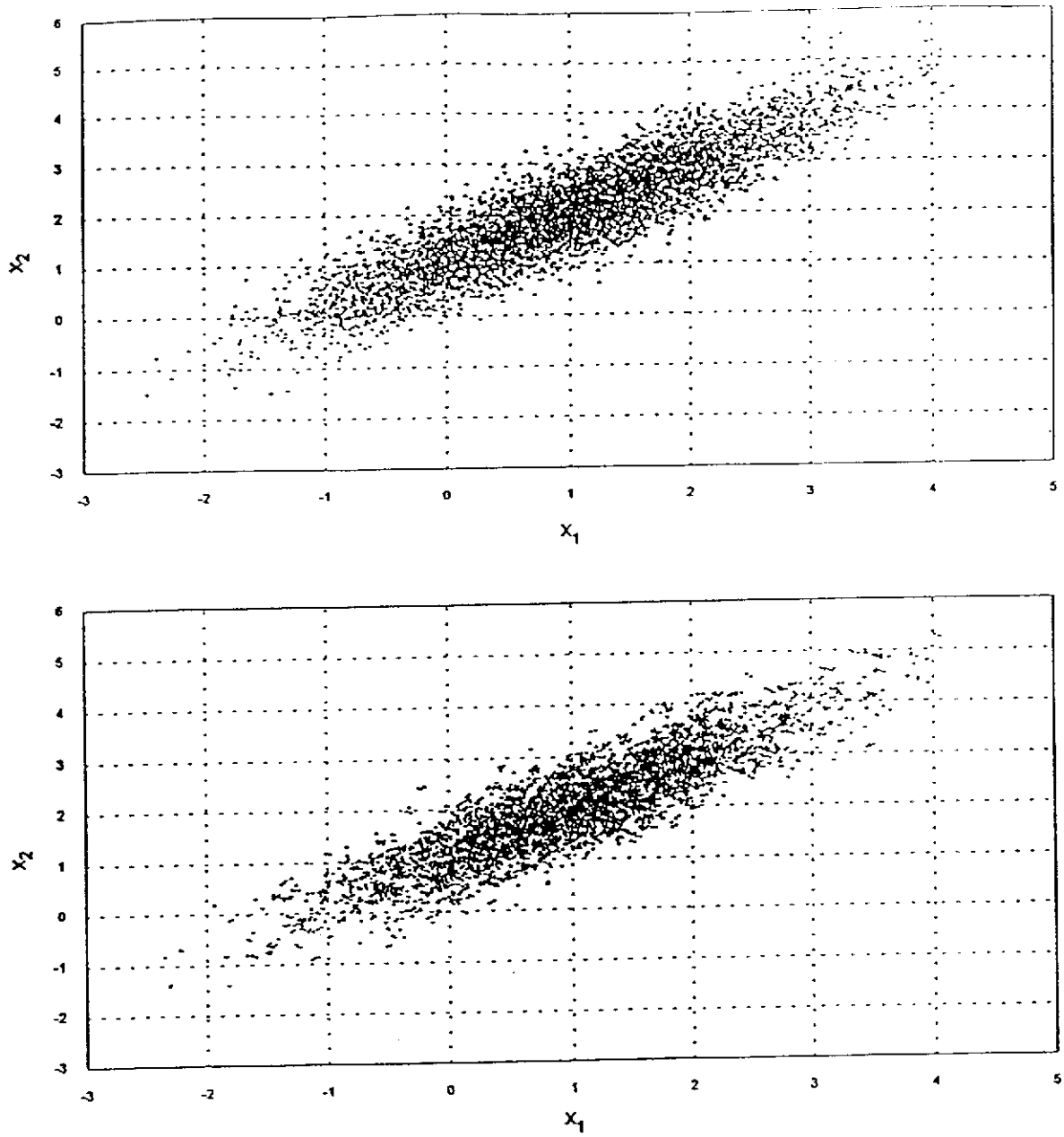

Figure 3. Scatter Plots of Simulated Draws. Top panel: Generated by Choleski approach. Bottom panel: Generated by M–H with reflection candidate-generating density.

and the third term in (11) is proportional to the density of the observations $(y_3, \ldots, y_n)$ given Y_2.

If the only prior information available is that the process is stationary, then the posterior distribution of the parameters is

$$\pi(\phi, \sigma^2 \mid Y_n) \propto l(\phi, \sigma^2) I[\phi \in S],$$

where $I[\phi \in S]$ is 1 if $\phi \in S$ and 0 otherwise.

How can this posterior density be simulated? The answer lies in recognizing two facts. First, the blocking strategy is useful for this problem by taking ϕ and σ^2 as blocks. Second, from the regression ANOVA decomposition, the exponential term of (11) is proportional to

$$\exp\left[-\frac{1}{2\sigma^2}(\phi - \widehat{\phi})' G(\phi - \widehat{\phi})\right],$$

where $\widehat{\phi} = G^{-1} \sum_{t=3}^{n} (w_t y_t)$ and $G = \sum_{t=3}^{n} (w_t w_t')$. This is the kernel of the normal density with mean $\widehat{\phi}$ and covariance matrix $\sigma^2 G^{-1}$. These observations immediately lead to the following full conditional densities for σ^2 and ϕ:

1. The density of σ^2 given ϕ and Y_n is inverted gamma with parameters $n/2$ and $Y_2 V^{-1} Y_2 + \sum_{t=3}^{n} (y_t - w_t'\phi)^2$.
2. The density of ϕ given σ^2 and Y_n is

$$\pi(\phi \mid Y_n, \sigma^2) \propto \Psi(\phi, \sigma^2) \times \{f_{\text{nor}}(\phi \mid \widehat{\phi}, \sigma^2 G^{-1}) I[\phi \in S]\}, \tag{13}$$

where f_{nor} is the normal density function.

A sample of draws from $\pi(\sigma^2, \phi \mid Y_n)$ can now be obtained by successively sampling ϕ from $\pi(\phi \mid Y_n, \sigma^2)$, and given this value of ϕ, simulating σ^2 from $\pi(\sigma^2 \mid Y_n, \phi)$. The latter simulation is straightforward. For the former,

Table 1. Summaries of the Posterior Distribution for Simulated AR(2) Model

	Posterior						
Param.	Mean	Num. SE	SD	Median	Lower	Upper	Corr.
ϕ_1	1.044	.002	.082	1.045	.882	1.203	.133
ϕ_2	−.608	.001	.082	−.610	−.763	−.445	.109
σ^2	1.160	.003	.170	1.143	.877	1.544	.020

because it can be shown that $|V^{-1}|^{1/2}$ is bounded for all values of ϕ in the stationary region, we generate candidates from the density in curly braces of (13), following the idea described in Section 5. Then, the value of ϕ is simulated as: At the jth iteration (given the current value $\sigma^{2(j)}$), draw a candidate $\phi^{(j+1)}$ from a normal density with mean $\hat{\phi}$ and covariance $\sigma^{2(j)}G^{-1}$; if it satisfies stationarity, move to this point with probability

$$\min\left\{\frac{\Psi(\phi^{(j+1)}, \sigma^{2(j)})}{\Psi(\phi^{(j)}, \sigma^{2(j)})}, 1\right\}$$

and otherwise set $\phi^{(j+1)} = \phi^{(j)}$, where $\Psi(\cdot, \cdot)$ is defined in (12). The A–R method of Section 2 can also be applied to this problem by drawing candidates $\phi^{(j+1)}$ from the normal density in (13) until $U \leq \Psi(\phi^{(j+1)}, \sigma^{2(j)})$. Many draws of ϕ may be necessary, however, before one is accepted because $\Psi(\phi, \sigma^2)$ can become extremely small. Thus the direct A–R method, although available, is not a competitive substitute for the M–H scheme described above.

In the sampling process we ignore the first $n_0 = 500$ draws and collect the next $N = 5{,}000$. These are used to approximate the posterior distributions of ϕ and σ^2. It is worth mentioning that the entire sampling process took just 2 minutes on a 50 MHz PC. For comparison we obtained samples from the A–R method, which took about 4 times as long as the M–H algorithm.

The posterior distributions are summarized in Table 1, where we report the posterior mean (the average of the simulated values), the numerical standard error of the posterior mean (computed by the batch means method), the posterior standard deviations (the standard deviation of the simulated values), the posterior median, the lower 2.5 and upper 97.5 percentiles of the simulated values, and the sample first-order serial correlation in the simulated values (which is low and not of concern). From these results it is clear that the M–H algorithm has quickly and accurately produced a posterior distribution concentrated on the values that generated the data.

8. CONCLUDING REMARKS

Our goal in this article is to provide a tutorial exposition of the Metropolis–Hastings algorithm, a versatile, efficient, and powerful simulation technique. It borrows from the well-known A–R method the idea of generating candidates that are either accepted or rejected, but then retains the current value when rejection takes place. The Markov chain thus generated can be shown to have the target distribution as its limiting distribution. Simulating from the target distribution is then accomplished by running the chain a large number of times. We provide a simple, intuitive justification for the form taken by the probability of move in the M–H algorithm by showing its relation to reversibility. We also discuss implementation issues and two applications, the M–H acceptance rejection algorithm and the use of the algorithm in block-at-a-time setting. Finally, the procedures are illustrated with two examples.

[Received April 1994. Revised January 1995.]

REFERENCES

Bhattacharya, R. N., and Waymire, E. C. (1900), *Stochastic Processes with Applications*, New York: John Wiley.

Billingsley, P. (1986), *Probability and Measure* (2nd ed.), New York: John Wiley.

Box, G. E. P., and Jenkins, G. M. (1976), *Time Series Analysis: Forecasting and Control* (rev. ed.), San Francisco: Holden-Day.

Casella, G., and George, E. (1992), "Explaining the Gibbs Sampler," *The American Statistician*, 46, 167–174.

Chib, S., and Greenberg, E. (1993), "Markov Chain Monte Carlo Simulation Methods in Econometrics," *Econometric Theory*, in press.

——— (1994), "Bayes Inference for Regression Models with ARMA(p, q) Errors," *Journal of Econometrics*, 64, 183–206.

Dagpunar, J. (1988), *Principles of Random Variate Generation*, New York: Oxford University Press.

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. (1992), "Iterative and Non-Iterative Simulation Algorithms," in *Computing Science and Statistics (Interface Proceedings)*, 24, 433–438.

Gelman, A., and Rubin, D. B. (1992), "Inference from Iterative Simulation Using Multiple Sequences" (with discussion), *Statistical Science*, 7, 457–511.

Geweke, J. (1989), "Bayesian inference in econometric models using Monte Carlo integration," *Econometrica*, 57, 1317–1340.

Hastings, W. K. (1970), "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," *Biometrika*, 57, 97–109.

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.

Meyn, S. P., and Tweedie, R. L. (1993), *Markov Chains and Stochastic Stability*, London: Springer-Verlag.

Müller, P. (1993), "A Generic Approach to Posterior Integration and Gibbs Sampling," manuscript.

Nummelin, E. (1984), *General Irreducible Markov Chains and Non-Negative Operators*, Cambridge: Cambridge University Press.

Phillips, D. B., and Smith, A. F. M. (1994), "Bayesian Faces via Hierarchical Template Modeling," *Journal of the American Statistical Association*, 89, 1151–1163.

Roberts, G. O., Gelman, A., and Gilks, W. R. (1994), "Weak Convergence and Optimal Scaling of Random Walk Metropolis Algorithms," Technical Report, University of Cambridge.

Roberts, G. O., and Tweedie, R. L. (1994), "Geometric Convergence and Central Limit Theorems for Multidimensional Hastings and Metropolis Algorithms," Technical Report, University of Cambridge.

Rubinstein, R. Y. (1981), *Simulation and the Monte Carlo Method*, New York: John Wiley.

Smith, A. F. M., and Roberts, G. O. (1993), "Bayesian Computation via the Gibbs Sampler and Related Markov Chain Monte Carlo Methods," *Journal of the Royal Statistical Society*, Ser. B, 55, 3–24.

Tanner, M. A. (1993), *Tools for Statistical Inference* (2nd ed.), New York: Springer-Verlag.

Tanner, M. A., and Wong, W. H. (1987), "The Calculation of Posterior Distributions by Data Augmentation," *Journal of the American Statistical Association*, 82, 528–549.

Tierney, L. (1994), "Markov Chains for Exploring Posterior Distributions" (with discussion), *Annals of Statistics*, 22, 1701–1762.