Technical Report No. 9722, Department of Statistics, University of Toronto

Markov Chain Monte Carlo Methods Based on 'Slicing' the Density Function

Radford M. Neal
Department of Statistics and Department of Computer Science
University of Toronto, Toronto, Ontario, Canada
http://www.cs.utoronto.ca/~radford/
radford@stat.utoronto.ca

18 November 1997

Abstract. One way to sample from a distribution is to sample uniformly from the region under the plot of its density function. A Markov chain that converges to this uniform distribution can be constructed by alternating uniform sampling in the vertical direction with uniform sampling from the horizontal 'slice' defined by the current vertical position. Variations on such 'slice sampling' methods can easily be implemented for univariate distributions, and can be used to sample from a multivariate distribution by updating each variable in turn. This approach is often easier to implement than Gibbs sampling, and may be more efficient than easily-constructed versions of the Metropolis algorithm. Slice sampling is therefore attractive in routine Markov chain Monte Carlo applications, and for use by software that automatically generates a Markov chain sampler from a model specification. One can also easily devise overrelaxed versions of slice sampling, which sometimes greatly improve sampling efficiency by suppressing random walk behaviour. Random walks can also be avoided in some slice sampling schemes that simultaneously update all variables.

1 Introduction

Markov chain methods such as Gibbs sampling (Gelfand and Smith 1990) and the Metropolis algorithm (Metropolis, *et al* 1953, Hastings 1970) can be used to sample from many of the complex, multivariate distributions encountered in statistics (see, for example, the book edited by Gilks, Richardson, and Spiegelhalter (1996)). However, to implement Gibbs sampling, one may need to devise methods for sampling from non-standard univariate distributions, and to use the Metropolis algorithm, one must find an appropriate 'proposal' distribution that will lead to efficient sampling. The need for such special tailoring limits the routine use of these methods, and inhibits the development of software that automatically constructs Markov chain samplers from model specifications.

In this paper, I describe a class of 'slice sampling' methods that can be applied to a wide variety of distributions, without the special programming that may be required to use

Gibbs sampling, and with less parameter tuning than is needed for other common Markov chain methods. These methods originate with the observation that one can sample from a univariate distribution by sampling points uniformly from the region under the curve of its density function, and then looking only at the horizontal coordinates of the sample points. A Markov chain that converges to this uniform distribution can be constructed by alternately sampling uniformly from the vertical interval defined by the density at the current point, and from the union of intervals that constitutes the horizontal 'slice' though the density that the point chosen from this vertical interval defines. If this last step is still difficult, one may substitute some other update that leaves the uniform distribution over the current slice invariant. To sample from a multivariate distribution, such single-variable slice sampling updates can be applied to each variable in turn, as is done in Gibbs sampling and some Metropolis methods.

Though simple and widely applicable, these single-variable slice sampling methods share with Gibbs sampling and simple forms of the Metropolis algorithm the disadvantage that they explore the distribution by means of an inefficient random walk. Large gains in sampling efficiency can often be produced by using Hybrid Monte Carlo or other dynamical methods (Duane, Kennedy, Pendleton, and Roweth 1987; Horowitz 1991; Neal 1994; and for reviews from a more statistical perspective, Neal 1993, 1996). The benefits of random walk suppression are analysed in some simple contexts by Diaconis, Holmes, and Neal (1997). For some distributions, such benefits can also be obtained by using an overrelaxation method (Adler 1981; Barone and Frigessi 1990; Green and Han 1992; Neal 1995). Dynamical and overrelaxation methods are not always easy to apply, however. Use of Markov chain samplers that avoid random walks would be assisted by the development of methods that require less special programming and parameter tuning.

Two approaches to random walk suppression based on slice sampling are discussed in this paper. First, I show how one can implement an overrelaxed version of the single-variable slice sampling scheme. This may provide the benefits of Adler's (1981) Gaussian overrelaxation method for more general distributions. I also describe slice sampling analogues of dynamical methods, which move around a multi-variable slice using a stepping procedure that proceeds consistently in one direction while reflecting off the slice boundaries. Although these more elaborate slice sampling methods require more tuning than the single-variable slice sampling schemes, they may still be easier to apply than alternative methods that avoid random walks, and hence are also candidates for routine or automated usage.

2 The idea of slice sampling

Suppose we wish to sample from a distribution over some subset of $\Re^n$, with density function proportional to some function $f(x)$. We can do this by sampling uniformly from the $n+1$ dimensional region that lies under the plot of $f(x)$. This idea can be formalized by introducing an auxiliary real variable, y, and defining a joint distribution over x and y that is uniform over the region $U = \{\, (x, y) \;:\; 0 < y < f(x) \,\}$ below the curve defined by $f(x)$.

That is, the joint density for (x, y) is

$$p(x, y) \;=\; \begin{cases} 1/Z & \text{if } 0 < y < f(x) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $Z = \int f(x)\,dx$. The marginal density for x is then

$$p(x) \;=\; \int_0^{f(x)} (1/Z)\,dy \;=\; f(x)\,/\,Z \tag{2}$$

as desired. To sample for x, we can sample jointly for (x, y), and then ignore y.

Generating independent points drawn uniformly from U may not be easy, so we might instead define a Markov chain that will converge to this uniform distribution. Gibbs sampling is one possibility: We sample alternately from the conditional distribution for y given the current x — which is uniform over the interval $(0, f(x))$ — and from the conditional distribution for x given the current y — which is uniform over the region $S = \{\, x : y < f(x) \,\}$, which I call the 'slice' defined by y. Generating an independent point drawn uniformly from S may still be difficult, in which case we can substitute some update for x that leaves the uniform distribution over S invariant.

Similar auxiliary variable methods been used in the past. Higdon (1996) has interpreted the standard Metropolis algorithm in these terms. The highly successful Markov chain algorithm for the Ising model due to Swendsen and Wang (1987) can also be seen as an auxiliary variable method, which has been generalized by Edwards and Sokal (1988). In their scheme, the density (or probability mass) function is proportional to a product of k functions: $p(x) \propto f_1(x) \cdots f_k(x)$. They introduce k auxiliary variables, $y_1, \ldots, y_k$, and define a joint distribution for $(x, y_1, \ldots, y_k)$ which is uniform over the region where $0 < y_i < f_i(x)$ for $i = 1, \ldots, k$. Gibbs sampling, or some related Markov chain procedure, can then be used to sample for $(x, y_1, \ldots, y_k)$, much as described above for the case of a single auxiliary variable. Applications of such methods to image analysis have been discussed by Besag and Green (1993) and by Higdon (1996).

Concurrently with the work reported here, Damien, Wakefield, and Walker (1997) have viewed such methods based on multiple auxiliary variables as a general approach to constructing Markov chain samplers for Bayesian inference problems. They illustrate how a decomposition of $f(x)$ into k factors can often be found for which the intersection of the sets $\{\, x : y_i < f_i(x) \,\}$ is easy to compute. This leads to an easily implemented sampler, but convergence may be slowed by the presence of many auxiliary variables.

Mira and Tierney (1997) have shown that these methods, with one or with many auxiliary variables, are uniformly ergodic under certain conditions. Roberts and Rosenthal (1997) have shown that these methods are geometrically ergodic under weaker conditions, and have also found some quantitative convergence bounds. These results all assume that the sampler generates a new value for x that is uniformly drawn from S, independently of the old value, which may be difficult in practice.

In this paper, I present several practical sampling methods based on slice sampling with a single auxiliary variable, often using updates for x that do not produce a point drawn

independently from S. To motivate these methods, I first discuss the need for Markov chain samplers that can be used routinely, and how well current schemes fulfill this need. Then, in section 4, I discuss a class of slice sampling methods that are intended for routine application to general distributions, and that have some advantages over current alternatives. In sections 5 and 6, I describe more elaborate slice sampling schemes that can improve sampling efficiency by avoiding the random walk movement that is characteristic of simple Markov chain samplers, and which again may have advantages over previous general-purpose schemes for random walk suppression. I conclude by discussing the merits of these and competing schemes in various situations.

3 General-purpose Markov chain sampling methods

Applications of Markov chain sampling in statistics often involve sampling from many distributions. In Bayesian applications, we must sample from the posterior distribution for the parameters of a model given certain data. Different datasets will produce different posterior distributions, which may differ in important characteristics such as diffuseness and multimodality. Furthermore, we will often wish to consider a variety of models. For routine use of Markov chain methods, it is important to minimize the amount of effort that the data analyst must spend in order to sample from all these distributions. Ideally, a Markov chain sampler would be constructed automatically for each model and dataset.

The Markov chain method most commonly used in statistics is Gibbs sampling, popularized by Gelfand and Smith (1990). Suppose again that we wish to sample from a distribution over n state variables (eg, model parameters), written as $x = (x_1, \ldots, x_n)$, with probability density $p(x)$. Gibbs sampling proceeds by sampling in succession from the conditional distributions for each x_i given the current values of the other x_j for $j \neq i$, with conditional densities written as $p(x_i \mid \{x_j\}_{j \neq i})$. Repetition of this procedure defines a Markov chain which leaves the desired distribution invariant, and which in many circumstances is ergodic (eg, when $p(x) > 0$ for all x). Running the Gibbs sampler for a sufficiently long time will then produce a sample of values for x from close to the desired distribution, from which the expectations of quantities of interest (eg, posterior means of parameters) can be estimated.

Gibbs sampling can be done only if we know how to sample from all the required conditional distributions. These sometimes have standard forms for which efficient sampling methods have been developed, but there are many models for which sampling from these conditional distributions requires the development of custom algorithms, or is infeasible in practice (eg, for multilayer perceptron networks (Neal 1996)). Note, however, that once methods for sampling from these conditional distributions have been found, no further tuning parameters need be set in order to produce the final Markov chain sampler.

The routine use of Gibbs sampling has been assisted by the development of Adaptive Rejection Sampling (ARS) (Gilks and Wild 1992; Gilks 1992), which can be used to efficiently sample from any conditional distribution whose density function is log concave, given only the ability to compute some function, $f_i(x_i)$, that is proportional to the conditional density, $p(x_i \mid \{x_j\}_{j \neq i})$ (the ability to also compute the derivative, $f_i'(x_i)$, is helpful, but not essen-

tial). This method is used by the BUGS software (Thomas, Spiegelhalter, and Gilks 1992) to automatically generate Markov chain samplers from model specifications. The first step in applying ARS is to find points on each side of the mode of the conditional distribution (one of which can be the current point). This will in general require a search, which will in turn require the choice of some length scale for an initial step. However, the burden of setting this scale parameter is lessened by the fact that a good value for it can be adaptively chosen based on past iterations of the Markov chain, without invalidating the results (since the setting of this parameter does not affect the distribution sampled from).

The Adaptive Rejection Metropolis Sampling (ARMS) method (Gilks, Best, and Tan 1995) generalizes ARS to conditional distributions whose density functions may not be log-concave. However, when the density is not log-concave, ARMS does not produce a new point drawn independently from the conditional distribution, but merely updates the current point in a fashion that leaves this distribution invariant. Applying ARMS to sample from the conditional distribution of each variable in succession will result in an equilibrium distribution that is exactly correct, but when some conditional distributions are not log-concave, it may take longer to approach this equilibrium than would be the case if true Gibbs sampling were used. Also, when a conditional distribution is not log-concave, the points used to set up the initial approximation to it must not be chosen adaptively, based on past iterations, as this could result in the wrong distribution being sampled (Gilks, Neal, Best, and Tan 1997). The initial approximation must be chosen based only on prior knowledge (including any preliminary Markov chain sampling runs), and on the current values of the other variables. Unlike ARS, neither the current value of the variable being updated, nor any statistics collected from previous updates (eg, a typical scale) can be used. This hinders routine use of the method.

Another general way of constructing a Markov chain sampler is to perform Metropolis updates (Metropolis, *et al* 1953, Hastings 1970), either to all variables simultaneously, or more commonly, to each variable in turn. A Metropolis update starts with the random selection of a 'candidate' state, drawn from a 'proposal' distribution. The candidate state is then accepted or rejected as the new state of the Markov chain, based on the ratio of the probability densities of the candidate state and the current state. If the candidate state is rejected, the new state is the same as the old state.

A simple 'random-walk' Metropolis scheme can be constructed based on a symmetric proposal distribution (eg, Gaussian) that is centred on the current state. All variables could be updated simultaneously in such a scheme, or alternatively, one variable could be updated at a time. In either case, a scale parameter is required for each variable to fix the width of the proposal distribution in that dimension. For the method to be valid, these scale parameters must not be set on the basis of past iterations, but rather only on the basis of prior knowledge (including preliminary runs), and the current values of variables that are not being changed in the present update. Choosing too large a value for the scale of a proposal distribution will result in a high rejection rate, while choosing too small a value will result in inefficient exploration via a random walk with unnecessarily small steps.

Though these methods have been used to do much useful work, there is clearly a need for

better methods, that can be routinely applied in a wider variety of situations. My objective in the next section is to find variations on slice sampling that can be used to sample from any distribution given only the ability to evaluate a 'black-box' function that is proportional to its density. These new methods are not necessarily expected to sample more efficiently than true Gibbs sampling or a well-designed Metropolis scheme. Rather, the hope is that they will be almost as efficient, while requiring less effort to implement and tune.

4 Single-variable slice sampling methods

When Gibbs sampling is difficult, because some of the required conditional distributions are hard to sample from, an alternative is to instead update some or all of the variables by procedures that merely leave their conditional distributions invariant, rather than producing independently-drawn points. This section presents ways of doing this using slice sampling.

For each real-valued variable, x_i, to be updated by such a single-variable slice sampling procedure, we must be able to compute a function, $f_i(x_i)$, that is proportional to $p(x_i \mid \{x_j\}_{j \neq i})$, where $\{x_j\}_{j \neq i}$ are the values of the other variables. Often, the joint distribution for $x_1, \ldots, x_n$ will be defined by some function, $f(x_1, \ldots, x_n)$, that is proportional to the joint density, in which case we can simply take $f_i(x_i) = f(\ldots, x_i, \ldots)$, where the variables other than x_i are fixed to their current values.

To simplify notation, I will here write the single real variable being updated as x, and will write $f(x)$ for the function proportional to its probability density. The single-variable slice sampling methods discussed here replace the current value, x_0, with a new value, x_1, found by a three-step procedure:

a) Draw a real value, y, uniformly from $(0, f(x_0))$, thereby defining a horizontal 'slice': $S = \{\, x \,:\, y < f(x) \,\}$. Note that x_0 is always within S.

b) Find an interval, $I = (L, R)$, around x_0 that contains at least a big part of the slice.

c) Draw the new point, x_1, from the part of the slice within this interval (ie, from $S \cap I$).

Step *(a)* picks a value for the auxiliary variable that is characteristic of slice sampling. Note that there is no need to retain this auxiliary variable from one iteration of the Markov chain to the next, since its old value is forgotten at this point anyway. In practice, it is often safer to compute $g(x) = \log(f(x))$ rather than $f(x)$ itself, in order to avoid possible problems with floating-point underflow. Once can then use the auxiliary variable $z = \log(y) = g(x_0) - e$, where e is exponentially distributed with mean one, and define the slice by $S = \{\, x \,:\, z < g(x) \,\}$.

Steps *(b)* and *(c)* can potentially be implemented in several ways, which must of course be such that the resulting Markov chain leaves the distribution defined by $f(x)$ invariant. Figure 1 illustrates one generally-applicable method, in which the interval is found by 'stepping out', and the new point is drawn with a 'shrinkage' procedure. Figure 2 illustrates an alternative 'doubling' procedure for finding the interval. These and some other variations are described in detail next, followed by a proof that the resulting transitions leave the

correct distribution invariant. Finally, I describe some shortcuts that are possible when the distribution is unimodal.

4.1 Finding an appropriate interval

After a value for the auxiliary variable has been drawn, defining the slice S, the next task is to find an interval $I = (L, R)$, containing the current point, x_0, from which the new point, x_1, will be drawn. We would like this interval to contain as much of the slice as is feasible, so as to allow the new point to differ as much as possible from the old point, but we would also like to avoid intervals that are much larger than the slice, as this will make the subsequent sampling step less efficient.

Several schemes for finding an interval are possible:

1) Ideally, we would set $L = \inf(S)$ and $R = \sup(S)$. That is, we would set I to the smallest interval that contains the whole of S. This may not be feasible, however.

2) If the range of x is bounded, we might simply let I be that range. However, this may not be good if the slice is typically much smaller than the range.

3) Given an estimate, w, for the scale of S, we can randomly pick an initial interval of size w, containing x_0, and then expand it by a 'stepping out' procedure.

4) Similarly, we can randomly pick an initial interval of size w, and then expand it by a 'doubling' procedure.

For each scheme, we must also be able to find the set A of acceptable successor states, defined as follows:

$$A \;=\; \{\, x \,:\, x \in S \cap I \text{ and } P(\text{Select } I \mid \text{At state } x) = P(\text{Select } I \mid \text{At state } x_0) \,\} \quad (3)$$

That is, A is the set of states from which we would be as likely to choose the interval I as we were to choose this I from the current state (for schemes (3) and (4), these probabilities are conditional on the random alignment of the initial interval). When we subsequently sample from within I (see section 4.2), we will ensure that the state chosen is in A, a fact which will be used in the proof of correctness in section 4.3. Clearly, for schemes (1) and (2), $A = S$. For scheme (3), we will arrange that $A = S \cap I$. Things are not so simple for scheme (4), for which a special test of whether a state is in A may be necessary.

Scheme (1), in which I is set to the smallest interval containing S, will be feasible when all solutions of $f(x) = y$ can be found analytically, or by an efficient and robust numerical method, but one cannot expect this in general. Often, even the number of disjoint intervals making up S will hard to determine.

Scheme (2) is certainly easy to implement when the range of x is bounded, and one can of course always arrange this by applying a suitable transformation. However, if the slice is usually much smaller than the full range, the subsequent sampling (see section 4.2) will be inefficient. This scheme has been used by Frey (1997).

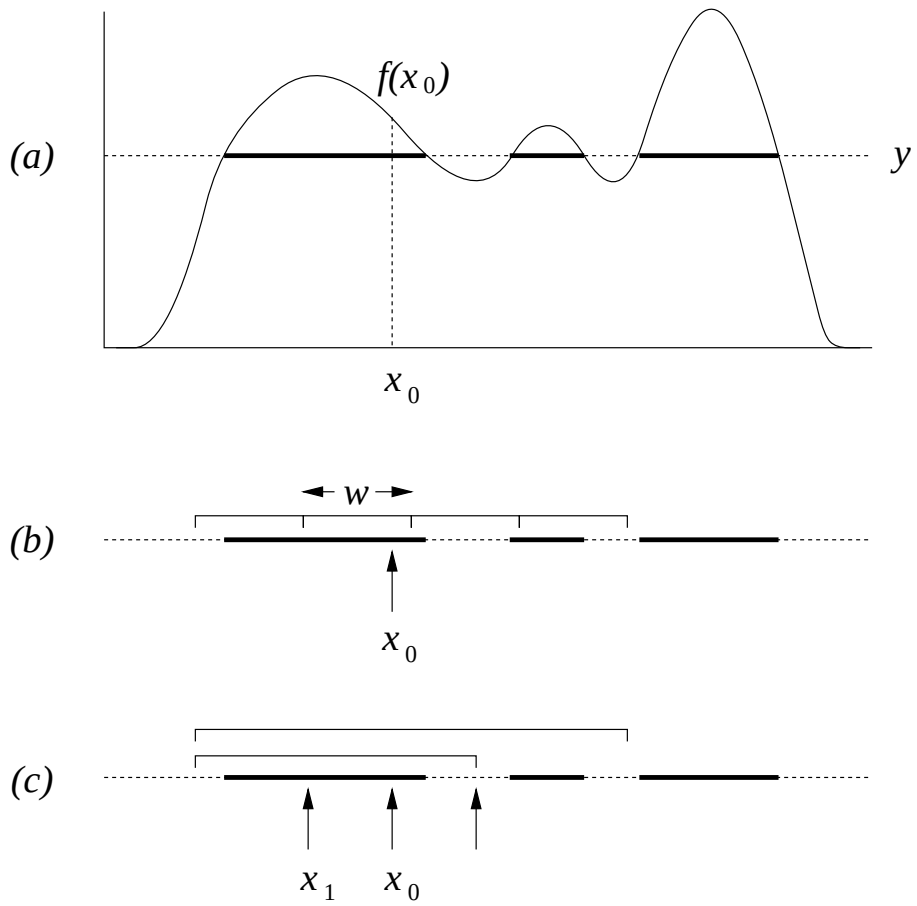

Figure 1: A single-variable slice sampling update using the stepping-out and shrinkage procedures. A new point, x_1, is selected to follow the current point, x_0, in three steps. *(a)* A vertical level, y, is drawn uniformly from $(0, f(x_0))$, and used to define a horizontal 'slice', indicated in bold. *(b)* An interval of width w is randomly positioned around x_0, and then expanded in steps of size w until both ends are outside the slice. *(c)* A new point, x_1, is found by picking uniformly from the interval until a point inside the slice is found. Points picked that are outside the slice are used to shrink the interval.

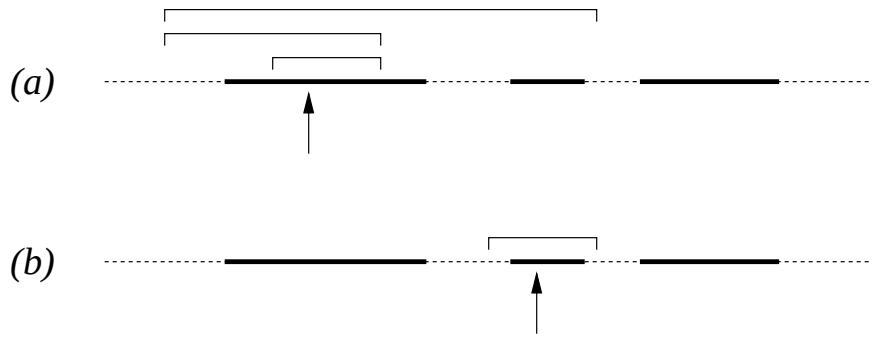

Figure 2: The doubling procedure. In *(a)*, the initial interval is doubled twice, until both ends are outside the slice. In *(b)*, where the start state is different, no doubling is done.

The 'stepping out' procedure (scheme (3) above) is appropriate for any distribution, provided that some rough estimate, w, for the typical width of the slice is available. The manner in which an interval is found by stepping out is illustrated in Figure 1(b) and the procedure is given in detail in Figure 3. The size of the interval found can be limited to mw, for some specified integer m, or the interval can be allowed to grow to any size (ie, m can be set to infinity), in which case the procedure can be simplified in an obvious way (eliminating all references to J and K). Note that the random positioning of the initial interval and the random apportioning of the maximum number of steps m into a limit on going to the left and a limit on going to the right are essential for correctness, as they ensure that the final interval could equally well have been produced from any point within $S \cap I$.

The 'doubling' procedure (scheme (4) above) can expand the interval faster than the stepping out procedure, and hence may be more efficient when the estimated size of the slice (w) turns out to be too small. This procedure is illustrated in Figure 2, and given in detail in Figure 4. Doubling produces a sequence of intervals, each twice the size of the previous one, until an interval is found with both ends outside the slice, or a predetermined limit is reached. Note that when the interval is doubled the two sides are not expanded equally. Instead just one side is expanded, chosen at random (irrespective of whether that side is already outside the slice). This is essential to the correctness of the method, since it produces a final interval that could have been obtained from points other than the current one. The set A of acceptable next states is restricted to those for which the same interval could have been produced, and is in general not all of $S \cap I$. This complicates the subsequent sampling somewhat, as described below.

4.2 Sampling from the part of the slice within the interval

Once an interval, $I = (L, R)$, has been found containing the current point, x_0, the final step of the single-variable slice sampling procedure is to randomly draw a new point, x_1, from within this interval. This point must lie within the set A of points acceptable as the next state of the Markov chain, defined in equation (3).

Two methods could be used to sample from I:

i) Repeatedly sample uniformly from I until a point is drawn that lies within A.

ii) Repeatedly sample uniformly from an interval that is initially equal to I, and which shrinks each time a point is drawn that is not in A, until a point within A is found.

Method *(i)* could potentially be very inefficient, if ever A turns out to be a tiny portion of I. The shrinkage of the interval in method *(ii)* ensures that the expected number of points drawn will not be too large, making this a more appropriate method for general use.

The shrinkage procedure is shown in detail in Figure 5. Note that each rejected point is used to shrink the interval in such a way that the current point remains within it. Since the current point is always within A, the interval used will alway contains acceptable points, ensuring that the procedure will terminate.

Input: f = function proportional to the density
x_0 = the current point
y = the vertical level defining the slice
w = estimate of the typical size of a slice
m = integer limiting the size of a slice to mw

Output: (L, R) = the interval found

$U \sim \text{Uniform}(0,1)$
$L \leftarrow x_0 - w * U$
$R \leftarrow L + w$

$V \sim \text{Uniform}(0,1)$
$J \leftarrow \text{Floor}(m * V)$
$K \leftarrow (m-1) - J$

repeat while $J > 0$ and $y < f(L)$:
    $L \leftarrow L - w$
    $J \leftarrow J - 1$

repeat while $K > 0$ and $y < f(R)$:
    $R \leftarrow R + w$
    $K \leftarrow K - 1$

Figure 3: The 'stepping out' procedure for finding an interval around the current point. The notation $U \sim \text{Uniform}(0,1)$ indicates that U is set to a number randomly drawn from the uniform distribution on $(0,1)$.

Input: f = function proportional to the density
x_0 = the current point
y = the vertical level defining the slice
w = estimate of the typical size of a slice
p = integer limiting the size of a slice to $2^p w$

Output: (L, R) = the interval found

$U \sim \text{Uniform}(0,1)$
$L \leftarrow x_0 - w * U$
$R \leftarrow L + w$
$K \leftarrow p$

repeat while $K > 0$ and $\{\ y < f(L)$ or $y < f(R)\ \}$:
    $V \sim \text{Uniform}(0,1)$
    if $V < 1/2$ then $L \leftarrow L - (R - L)$
        else $R \leftarrow R + (R - L)$
    $K \leftarrow K - 1$

Figure 4: The 'doubling' procedure for finding an interval around the current point. Note that it is possible to save some computation in second and later iterations of the loop, since only one of $f(L)$ and $f(R)$ will have changed from the previous iteration.

Input: f = function proportional to the density
x_0 = the current point
y = the vertical level defining the slice
w = estimate of the typical size of a slice
(L, R) = the interval to sample from

Output: x_1 = the new point

$\bar{L} \leftarrow L, \quad \bar{R} \leftarrow R$

repeat:

$U \sim \text{Uniform}(0, 1)$
$x_1 \leftarrow \bar{L} + U * (\bar{R} - \bar{L})$

if $y < f(x_1)$ and $\text{Accept}(x_1)$ then exit loop

if $x_1 < x_0$ then $\bar{L} \leftarrow x_1$ else $\bar{R} \leftarrow x_1$

Figure 5: The 'shrinkage' procedure for sampling from the interval. The notation $\text{Accept}(x_1)$ represents a test for whether a point, x_1, that is within $S \cap I$ is an acceptable next state. If scheme (1), (2), or (3) was used for constructing the interval, all points within $S \cap I$ are acceptable. If the doubling procedure (scheme (4)) was used, the point must also pass the test of Figure 6, below.

Input: f = function proportional to the density
x_0 = the current point
x_1 = the possible next point
y = the vertical level defining the slice
w = estimate of the typical size of a slice
(L, R) = the interval found by the doubling procedure

Output: whether or not x_1 is an acceptable next state

$\hat{L} \leftarrow L, \quad \hat{R} \leftarrow R$
$D \leftarrow$ false

repeat while $\hat{R} - \hat{L} > 1.1 * w$:

$M \leftarrow (\hat{L} + \hat{R}) / 2$

if $\{ x_0 < M$ and $x_1 \geq M \}$ or $\{ x_0 \geq M$ and $x_1 < M \}$ then $D \leftarrow$ true

if $x_1 < M$ then $\hat{R} \leftarrow M$ else $\hat{L} \leftarrow M$

if D and $y \geq f(\hat{L})$ and $y \geq f(\hat{R})$ then
The new point is not acceptable

The new point is acceptable if not rejected in the loop above

Figure 6: The test for whether a new point, x_1, that is within $S \cap I$ is an acceptable next state, when the interval was found by the 'doubling' procedure. The multiplication by 1.1 in the 'while' condition guards against possible round-off error.

If the interval was found by scheme (1), (2), or (3), the set A is simply $S \cap I$. However, if the doubling procedure (scheme (4)) was used, A may be a smaller subset of $S \cap I$. This is illustrated in Figure 2. In 2(a), an interval is found by doubling an initial interval until both ends are outside the slice. A different starting point is considered in 2(b), one which might have been drawn from the interval found in 2(a). The doubling procedure terminates earlier starting from here, so this point is not in A.

The Accept (x_1) predicate in Figure 5 tests whether a point in $S \cap I$ is in A. If I was found using scheme (1), (2), or (3), this is trivial — all points in $S \cap I$ are in A. But if the doubling procedure (scheme (4)) was used, the acceptance test of Figure 6 is needed. This procedure works backward through the intervals that the doubling procedure would pass through to arrive at I when starting from the new point, checking that none of them have both ends outside the slice, which would lead to earlier termination of the doubling procedure. (Note that one needn't check this explicitly until the intervals differ from those followed from the current point, a condition tracked with the variable D in the procedure.) If the distribution is known to be unimodal, this test can be omitted, as discussed in section 4.4.

4.3 Correctness of single-variable slice sampling

To show that single-variable slice sampling is a correct procedure, we must show that each update leaves the desired distribution invariant. To guarantee convergence to this distribution, the resulting Markov chain must also ergodic. This is not always true, but it is in those situations (such as when $f(x) > 0$ for all x) for which one can easily show that Gibbs sampling is ergodic. I will not discuss the more difficult situations here.

To show invariance, we suppose that the initial state, x_0, is distributed according to $f(x)$. In step *(a)* of single-variable slice sampling, a value for y is drawn uniformly from $(0, f(x))$. The joint distribution for x_0 and y will therefore be as in equation (1). If the subsequent steps update x_0 to x_1 in a manner that leaves this joint distribution invariant, then when we subsequently discard y, the resulting distribution for x_1 will be the marginal of this joint distribution, which is the same as that defined by $f(x)$, as desired.

We therefore need only show that the selection of x_1 to follow x_0 in steps *(b)* and *(c)* of the single-variable slice sampling procedure leaves the joint distribution of x and y invariant. Since these steps do not change y, this is the same as leaving the conditional distribution for x given y invariant, and this conditional distribution is uniform over $S = \{ x : y < f(x) \}$, the slice defined by y. We can show invariance of this distribution by showing that the updates satisfy detailed balance, which for a uniform distribution reduces to showing that the probability density for x_1 to be selected as the next state, given that the current state is x_0, is the same as the probability density for x_0 to be the next state, given that x_1 is the current state, for any states x_0 and x_1 within S.

In the process of picking a new state, various intermediate choices are made randomly. When the interval is found by the stepping out procedure of Figure 3, the alignment of the initial interval is randomly chosen, as is the division of the maximum number of intervals into those used to extend to the left and those used to extend to the right. For the doubling

procedure of Figure 4, the alignment of the initial interval is random and the decisions whether to extend to the right or to the left are also made randomly. When sampling is done using the shrinkage procedure of Figure 5, zero or more rejected points will be chosen before the final point. Let r denote these intermediate random choices. I will prove that detailed balance holds for the entire procedure by showing the following stronger result:

$$\begin{aligned} &P(\text{next state} = x_1,\ \text{intermediate choices} = r \mid \text{current state} = x_0) \\ &\quad = \quad P(\text{next state} = x_0,\ \text{intermediate choices} = \pi(r) \mid \text{current state} = x_1) \end{aligned} \tag{4}$$

where $\pi(r)$ is some one-to-one mapping that has Jacobian one (with regard to the real-valued variables), which may depend on x_0 and x_1. Integrating over all possible values for r then gives the desired result.

In detail, the mapping π used is as follows. First, if the interval I is found by the stepping out or doubling procedure, an intermediate value, U, will be generated by the procedure of Figure 3 or 4, and used to define the initial interval. We define π so that it maps the value U_0 chosen when the state is x_0 to the following U_1 when the state is x_1:

$$U_1 \quad = \quad \text{Frac}\,(U_0 + (x_1 - x_0)/w) \tag{5}$$

where $\text{Frac}\,(x) = x - \text{Floor}\,(x)$ is the fractional part of x. This mapping associates values that produce the same alignment of the initial interval. Note also that it has Jacobian one. If the stepping out procedure is used, a value for J is also generated, uniformly from the set $\{\,0, \ldots, m-1\,\}$. The mapping π associates the J_0 found when the state is x_0 with the following J_1 when the state is x_1:

$$J_1 \quad = \quad J_0 \,+\, (x_1/w - U_1) \,-\, (x_0/w - U_0) \tag{6}$$

Here, $(x_1/w - U_1) - (x_0/w - U_0)$ is an integer giving the number of steps (of size w) from the left end of the interval containing x_0 to the left end of the interval containing x_1. This is the amount by which we must adjust J_0 in order to ensures that if the interval found starting from x_0 grows to its maximum size, the associated interval found starting from x_1 will be identical. Similarly, if the doubling procedure of Figure 4 is used, the sequence of random decisions as to which side of the interval to expand is mapped by π to the sequence of decisions that would cause the interval expanding from x_1 to become identical to the interval expanding from x_0 when the latter first includes x_1, and to remain identical through further expansions. Note in this respect that there is at most one way that an given final interval can be obtained by successive doublings from a given initial interval, and that the alignment of the initial intervals by the association of U_0 with U_1 ensures that doubling starting from x_1 can indeed lead to the same interval as found from x_0. Finally, to complete the definition, π maps the sequence of rejected points used to shrink the interval found from x_0 (see Figure 5) to the same sequence of points when x_1 is the start state.

It remains to show that with this definition of π, equation (4) does indeed hold, for all points x_0 and x_1, and all possible intermediate values r. Note that the equation certainly holds when both sides are zero, so we can eliminate from consideration any situation where

movement between x_0 and x_1 is impossible (in conjunction with the given intermediate values).

Consider first the probability (density) for producing the intermediate values that define the interval I. For the stepping out and doubling procedures, the values U_0 and U_1 (related by π) that are generated from x_0 and x_1 will certainly have the same probability density, since U is drawn from a uniform distribution. Similarly, for the stepping out procedure, the values J_0 and J_1 are drawn from a uniform distribution over $\{0, \ldots, m-1\}$, and hence have the same probability as long as J_0 and J_1 are both in this set, which will be true whenever movement between x_0 and x_1 is possible. For the doubling procedure, a sequence of decisions as to which side to extend is made, with all sequences of a given length having the same probability. Here also, the sequences associated by π will have the same probability, *provided* the same number of doublings are done starting from x_0 as from x_1. This need not be true in general, but if the sequence from x_1 is shorter, the test of Figure 6 will eliminate x_1 as a possible successor to x_0, and if the sequence from x_0 is shorter, x_1 will not be a possible successor because it will be outside the interval I found from x_0. Both sides of equation 4 will therefore be zero in this situation.

Note next that the intervals found by any of the schemes of section 4.1 will be the same for x_0 as for x_1, when the intermediate values chosen are related by π, assuming a transition from x_0 to x_1 is possible. For the stepping out procedure, the maximum extent of the intervals will be the same because of the relationships between U_0 and U_1 and between J_0 and J_1. Furthermore, the actual intervals found by stepping out (limited by the maximum) must also be the same whenever a transition between x_0 and x_1 is possible, since if the interval starting from x_0 has reached x_1, expansion of both intervals will continue in the same direction until the outside of the slice or the maximum is reached, and likewise in the other direction. Similarly, the mapping π is defined to be such that if the interval found by the doubling procedure starting from x_0 includes x_1, the same interval would be found from x_1, provided the process was not terminated earlier (by both ends being outside the slice), in which case x_1 is not a possible successor (as it would be rejected by the procedure of Figure 6). Note also that since the set A is determined by I (for any start state), it too will be the same for x_0 as for x_1.

If we sample from this I by simple rejection (scheme *(i)* in section 4.2), the state chosen will be uniformly distributed over A, so the probability of picking x_0 will be the same as that of picking x_1. If we instead use the shrinkage procedure (scheme *(ii)*, in Figure 5), we need to consider as intermediate values the sequence of rejected points that were used to narrow the interval (recall that under π this sequence is the same for x_0 as for x_1). The probability density for the first of these is clearly the same for both starting points, since I is the same. As the interval shrinks, it remains the same for both x_0 and x_1, since the rejection decisions (based on A) are the same, and since we need consider only the case where the same end of the interval is moved to the rejected point (as otherwise a transition between x_0 and x_1 in conjunction with these intermediate values would be impossible). The probability densities for later rejected points, and for the final accepted state, are therefore also the same.

This completes the proof. Various seemingly reasonable modifications — such as changing the doubling procedure of Figure 4 to not expand the interval on a side that is already outside the slice — would undermine the argument of the proof, and hence cannot be used. However, some shortcuts are allowed when the distribution is unimodal, as discussed next.

4.4 Shortcuts for unimodal distributions

Certain shortcuts can be used when the conditional distribution for the variable being updated is known to be unimodal, or more generally, when the slice, S, is known to consist of a single interval. For some values of the auxiliary variable, S may be a single interval even when the distribution is multimodal, but the effort required to confirm this probably exceeds the gain from using the shortcuts, so I will refer only to the unimodal case here.

Two shortcuts apply when the 'doubling' procedure is used to find the interval. First, for a unimodal distribution, the acceptance test in Figure 6 can be omitted, since it will always indicate that the new point is acceptable. To see this, note that the procedure rejects a point when one of the intervals found by doubling from that starting point has both ends outside the slice, but does not contain the current point. Since both the current point and the new point are inside the slice, this is impossible if the slice consists of only one interval.

Second, the interval found by the doubling procedure can sometimes be shrunk at the outset. The side chosen for extension when the interval doubles will sometimes be outside the slice already. When the distribution is known to be unimodal, it is not possible for such an extension to contain any points within the slice. Accordingly, before sampling is begun, the endpoints of the interval can be set to the first point in each direction that was found to lie outside the slice. This may reduce the number of points generated, while having no effect on the distribution of the point finally chosen.

Finally, if the distribution is known to be unimodal *and* no limit is imposed on the size of the interval found (ie, m and p in Figures 3 and 4 are infinite), the estimate, w, for the typical size of a slice can be set on the basis of past iterations. One could, for example, keep a running average of the distance between the old and new points in past iterations, and use this (or some suitable multiple) as the estimate w. This is valid because the distribution of the new point does not depend on w in this situation, even though w influences how efficiently this new point is found. Indeed, when the distribution is known to be unimodal, one can use any method at all for finding an interval that contains the current point and has both ends outside the slice, as any such interval will lead to the new point finally chosen being drawn uniformly from the slice.

5 Overrelaxed slice sampling

Sampling efficiency can often be improved by 'overrelaxation' methods, which in certain circumstances suppress the random walk behaviour characteristic of simple schemes such as Gibbs sampling. Like Gibbs sampling, overrelaxation methods update each variable in turn, but rather than drawing a new value for a variable from its conditional distribution

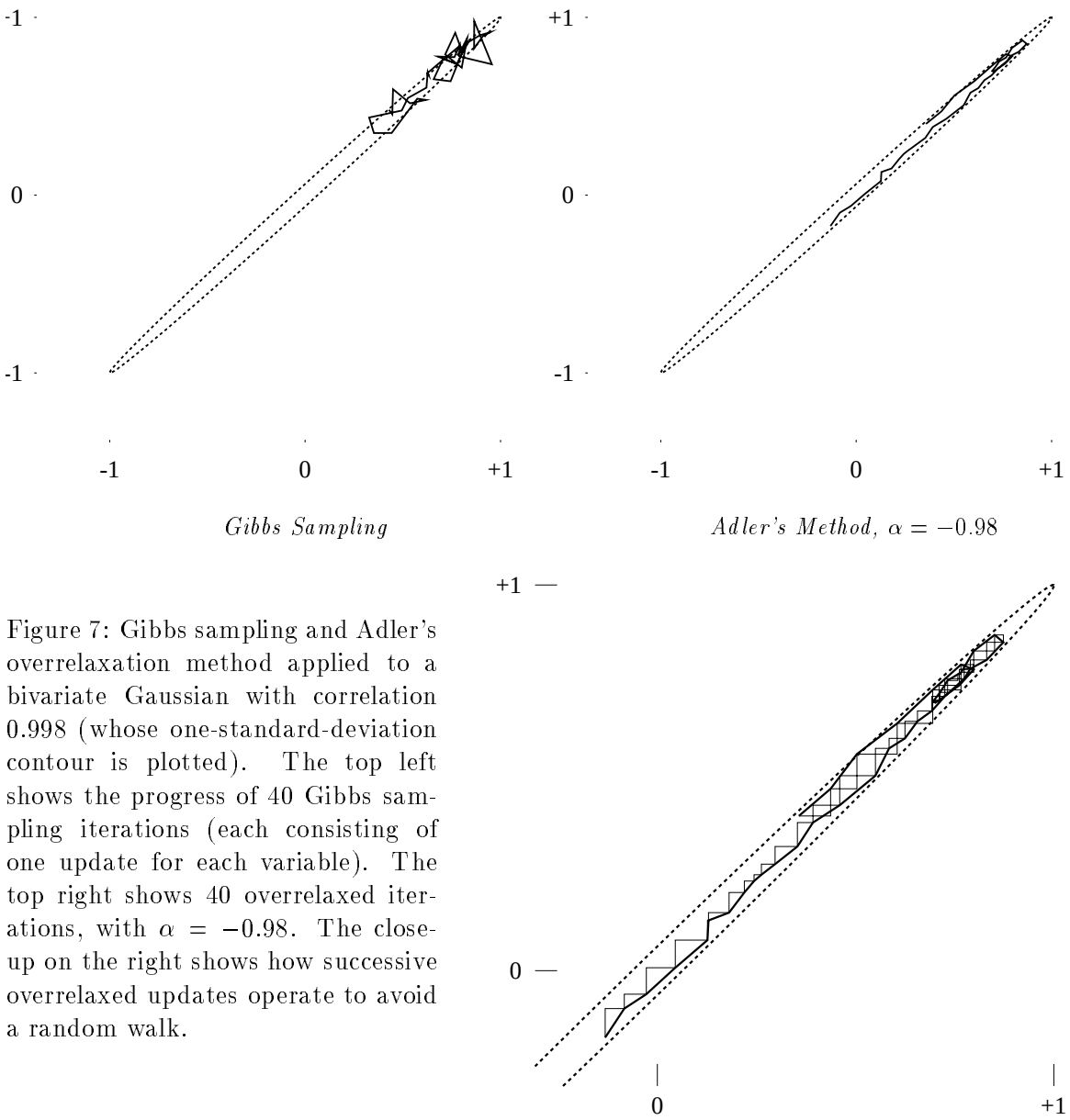

Figure 7: Gibbs sampling and Adler's overrelaxation method applied to a bivariate Gaussian with correlation 0.998 (whose one-standard-deviation contour is plotted). The top left shows the progress of 40 Gibbs sampling iterations (each consisting of one update for each variable). The top right shows 40 overrelaxed iterations, with $\alpha = -0.98$. The close-up on the right shows how successive overrelaxed updates operate to avoid a random walk.

independently of the current value, the new value is instead chosen to be on the opposite side of the mode from the current value. In Adler's (1981) scheme, applicable when the conditional distributions are Gaussian, the new value for variable i is

$$x_i' = \mu_i + \alpha\,(x_i - \mu_i) + \sigma_i\,(1 - \alpha^2)^{1/2}\,n \tag{7}$$

where μ_i and σ_i are the conditional mean and standard deviation of variable i, n is a Gaussian random variate with mean zero and variance one, and α is a parameter slightly greater than -1. This method is analysed and discussed by Barone and Frigessi (1990) and by Green and Han (1992), though these discussions fail in some respects to properly elucidate the true benefits and limitations of overrelaxation (Neal 1995). The crucial ability of overrelaxation to (sometimes) suppress random walks is illustrated for a bivariate Gaussian distribution in Figure 7.

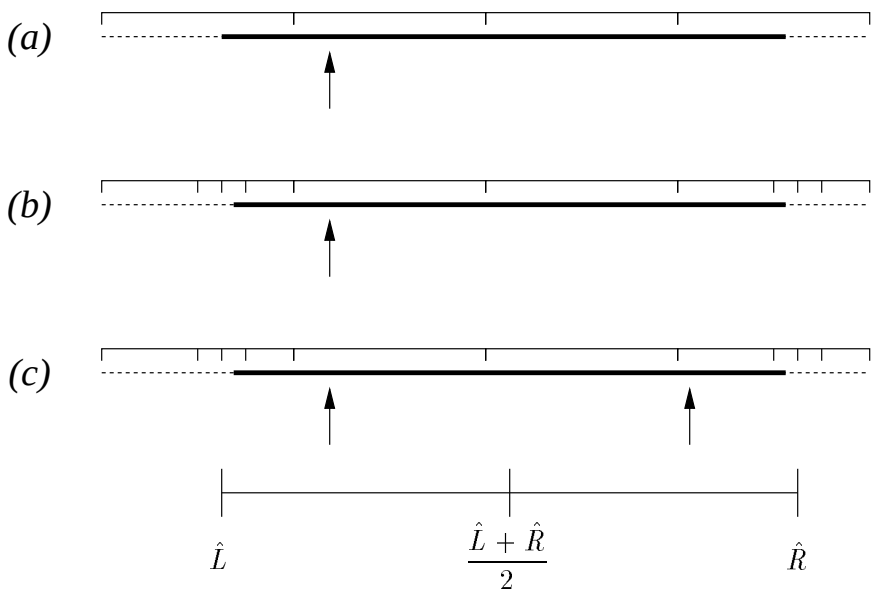

Figure 8: Overrelaxation using the stepping out procedure and bisection. In *(a)*, an interval with both ends outside the slice is found by stepping out from the current point, as was illustrated in Figure 1(b). In *(b)*, the endpoints of the slice are located more accurately using bisection. In *(c)*, a candidate point is found by flipping through the point half-way between the approximations to the endpoints. In this case, the candidate point is accepted, since it is within the slice, and within the orginal interval (prior to bisection).

Various attempts have been made to produce overrelaxation schemes that can be used when the conditional distributions are not Gaussian (Neal (1995) gives one scheme, and reviews others). The concept of overrelaxation seems to apply only when the conditional distributions are unimodal, so we may assume that this is usually the case, though we would like the method to at least remain valid (ie, leave the desired distribution invariant) even if this assumption turns out to be false. As discussed by Neal (1995), to obtain the full benefits of overrelaxation, it is important that almost every update be overrelaxed, with few or no 'rejections' that leave the state unchanged, as such rejections re-introduce an undesirable random walk aspect to the motion through state space.

In this section, I will show how overrelaxation can be done using slice sampling. Many schemes for overrelaxed slice sampling are possible, but I will describe only one in detail, based on the stepping out procedure and on bisection. This scheme is illustrated in Figure 8, and given in detail in Figure 9.

To begin, we apply the stepping out procedure of Figure 3 to find an interval around the current point. Normally, we would apply this procedure with the maximum size of the interval (m) set to infinity, or to some large value, since a proper overrelaxation operation requires that the entire slice be found, but the scheme remains valid for any m.

If the stepping out procedure found an interval around the slice that is bigger than the initial interval, then the two outermost steps will serve to locate the endpoints of the slice to within an interval of size w. (Here, we assume that the slice consists of a single interval, as

Input: f = function proportional to the density
x_0 = the current point
y = the vertical level defining the slice
w = estimate of the typical size of a slice
a = integer limiting endpoint accuracy to $2^{-a}\,w$
(L, R) = interval found by the stepping out procedure

Output: x_1 = the new point

$\bar{L} \leftarrow L, \quad \bar{R} \leftarrow R$
$\bar{w} \leftarrow w, \quad \bar{a} \leftarrow a$

When the interval is only of size w, the following section will narrow it until the mid-point is inside the slice (or the accuracy limit is reached).

if $R - L < 1.1 * w$ then

repeat:

$M \leftarrow (\bar{L} + \bar{R}) / 2$
if $\bar{a} = 0$ or $y < f(M)$ then exit loop
if $x_0 > M$ then $\bar{L} \leftarrow M$ else $\bar{R} \leftarrow M$
$\bar{a} \leftarrow \bar{a} - 1$
$\bar{w} \leftarrow \bar{w} / 2$

Endpoint locations are now refined by bisection, to the specified accuracy.

$\hat{L} \leftarrow \bar{L}, \quad \hat{R} \leftarrow \bar{R}$

repeat while $\bar{a} > 0$:

$\bar{a} \leftarrow \bar{a} - 1$
$\bar{w} \leftarrow \bar{w} / 2$
if $y \geq f(\hat{L} + w)$ then $\hat{L} \leftarrow \hat{L} + w$
if $y \geq f(\hat{R} - w)$ then $\hat{R} \leftarrow \hat{R} - w$

A candidate point is found by flipping from the current point to the opposite side of $(\hat{L}, \hat{R})$. It is then tested for acceptability.

$x_1 \leftarrow \hat{L} + \hat{R} - x_0$
if $x_1 < \bar{L}$ or $x_1 > \bar{R}$ or $y \geq f(x_1)$ then

$x_1 \leftarrow x_0$

Figure 9: The overrelaxation procedure using bisection. It is assumed that the interval (L, R) was found by the stepping out procedure, with a stepsize of w.

it will if the distribution is unimodal.) We then locate the endpoints more precisely using a bisection procedure. For each endpoint, we test whether the mid-point of the interval within which it is located is inside or outside the slice, and shrink this interval appropriately to narrow the location of the endpoint. This is repeated a times, after which each endpoint will be known to lie within an interval of size $2^{-a}\,w$.

If the stepping out procedure found that the initial interval (of size w) already had both ends outside the slice, then before doing any bisection, we narrow this interval, by shrinking it in half repeatedly until its mid-point is within the slice. We then use bisection as above to locate the endpoints to within an interval of size $2^{-a}\,w$.

Once the locations of the endpoints have been narrowed down, we can approximate the entire slice by the interval $(\hat{L}, \hat{R})$, formed from the outer bounds on the endpoint locations. To do an overrelaxed update, we flip from the current point, x_0, to a new point, x_1, that is the same distance as the current point from the middle of this interval, but on the opposite side. That is, we let

$$x_1 \;=\; \frac{\hat{L}+\hat{R}}{2} \;-\; \left(x_0 - \frac{\hat{L}+\hat{R}}{2}\right) \;=\; \hat{L} \;+\; \hat{R} \;-\; x_0 \tag{8}$$

We must sometimes reject this candidate point, in which case the new point is the same as the current point. First of all, we must reject x_1 if it lies outside the interval, $(\bar{L}, \bar{R})$, that had been found prior to bisection, since the interval found from x_1 would then be different, and detailed balance would not hold. However, this situation cannot arise when the distribution is unimodal. Secondly, we must reject x_1 if it lies outside the slice. This can easily happen for a multimodal distribution, and can happen even for a unimodal distribution when the endpoints of the slice have not been located exactly. However, the probability of rejection for a unimodal distribution can be reduced to as low a level as desired, at moderate cost, by locating the endpoints more precisely using more iterations of bisection.

The correctness of this procedure can be seen using arguments similar to those of section 4.3. The interval before bisection can be found by the doubling procedure instead of stepping out, provided the point found is rejected if it fails the acceptance test of Figure 6. However, rejection for this reason will not occur in the presumably typical case of a unimodal distribution.

One could use many methods other than bisection to narrow down the locations of the endpoints before overrelaxing. If the derivative of $f(x)$ can easily be calculated, one could use Newton iteration, whose rapid convergence would often allow the endpoints to be calculated to machine precision in a few iterations. For unimodal distributions, such exact calculations would eliminate the possibility of rejection, and would also make the final result be independent of the way the interval containing the slice was found, thereby allowing use of adaptive methods for finding this interval.

To obtain a full sampling scheme, overrelaxed updates of this sort would be applied to each variable in turn, in a fixed order, for a number of cycles, after which a normal slice sampling update would be done. Alternatively, each update could be done normally with

some small probability. A Markov chain consisting solely of overrelaxed updates might not be ergodic, and might in any case suppress random walks for too long. The frequency of normal updates is a tuning parameter, analogous to the choice of α in Adler's overrelaxation method, and would ideally be set so that the Markov chain moves systematically, rather than in a random walk, for long enough that it traverses a distance comparable to the largest dimension of the multivariate distribution, but for no longer than this. To keep from doing a random walk for around k steps, one would do every k'th update normally, and also arrange for the rejection rate for the overrelaxed updates to be less than $1/k$.

6 Reflective slice sampling

Rather than update variables one-at-time using one of the slice sampling methods described above, we might instead define a multivariate slice with a single draw of an auxiliary variable, and then update all variables simultaneously by some method that leaves the uniform distribution over this slice invariant. Many schemes of this sort are possible. In this section, I will describe schemes based on step-by-step movements that 'reflect' off the boundaries of the slice. Such movement with reflection can be seen as a specialization to uniform distributions of the Hamiltonian dynamics that forms the basis for Hybrid Monte Carlo (Duane, *et al* 1987). Like other dynamical methods, such reflective slice sampling schemes can suppress random walks, and thereby improve sampling efficiency.

As before, suppose we wish to sample from a distribution over $\Re^n$, defined by a function $f(x)$ that is proportional to the probability density, and which we here assume is differentiable. We must be able to calculate both $f(x)$ and its gradient (or equivalently, the value and gradient of $\log f(x)$). In each iteration of the Markov chain, we will draw a value for an auxiliary variable, y, uniformly from $(0, f(x))$, thereby defining an n-dimensional slice $S = \{\, x \,:\, y < f(x) \,\}$. We will also introduce n additional 'momentum' variables, written as a vector p, which serve to indicate the current direction and speed of motion through state space. At the start of each iteration, we pick a value for p, independently of x, from some rotationally symmetric distribution, typically Gaussian with mean zero and identity covariance matrix.

Once y and p have been drawn, we repeatedly update x by stepping in the direction of p. After some predetermined number of steps, we take the final value of x as our new state (provided it is acceptable). In each step, we try to set $x' = x + wp$, for some scale parameter w that determines the average step size. However, if the resulting x' is outside the slice S (ie, $y \geq f(x')$), we must somehow bring try to bring it back inside. The schemes considered here all do this by some form of reflection, but differ in the exact procedure used.

Ideally, we would reflect from the exact point at which movement in the direction of p first takes us outside the slice. This reflection operation modifies p, after which motion continues in the new direction, until we again encounter the boundary of the slice. When we hit the boundary at a point where the gradient of $f(x)$ is g, reflection will change p as follows:

$$p' \;=\; p \;-\; 2\, g\, \frac{p \cdot g}{|g|^2} \tag{9}$$

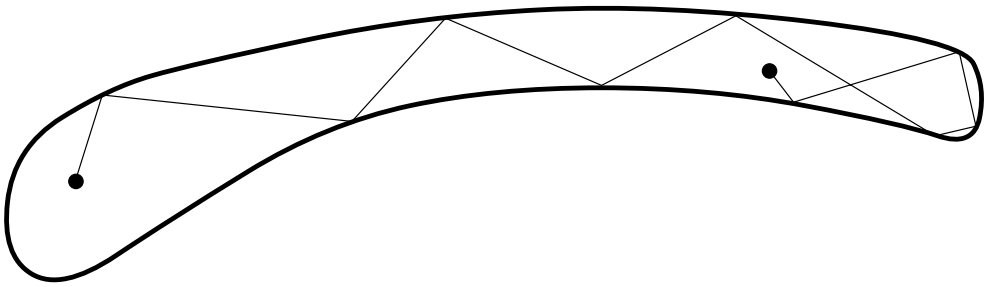

Figure 10: Moving around a two-dimensional slice by reflection from the exact boundaries.

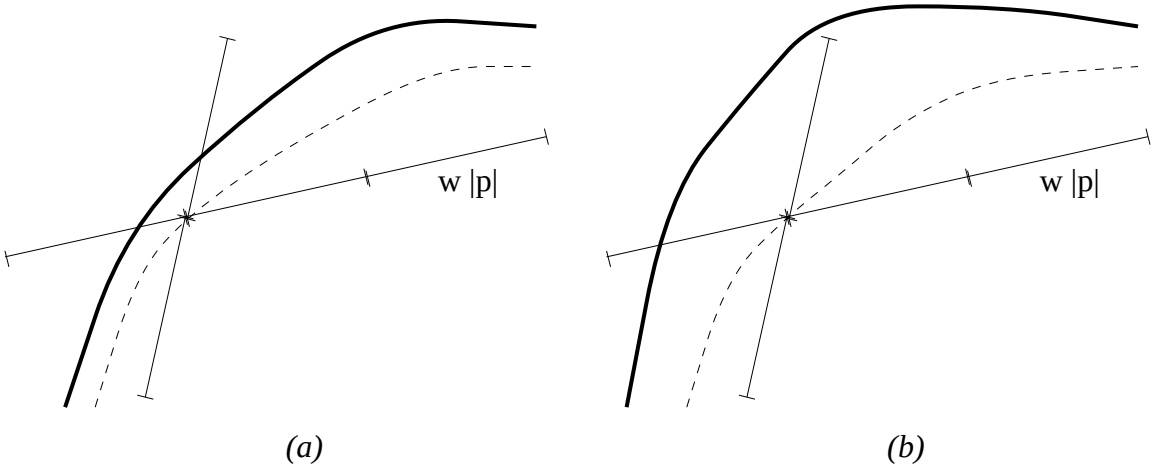

Figure 11: Reflection from an inside point. The trajectories here go in steps of size $w|p|$, starting from the top right, until a point outside the slice is reached, when a reflection is attempted based on the inner contour shown. In *(a)*, the reflection is is sucessful; in *(b)*, it must be rejected, since the reverse trajectory would not reflect at this point.

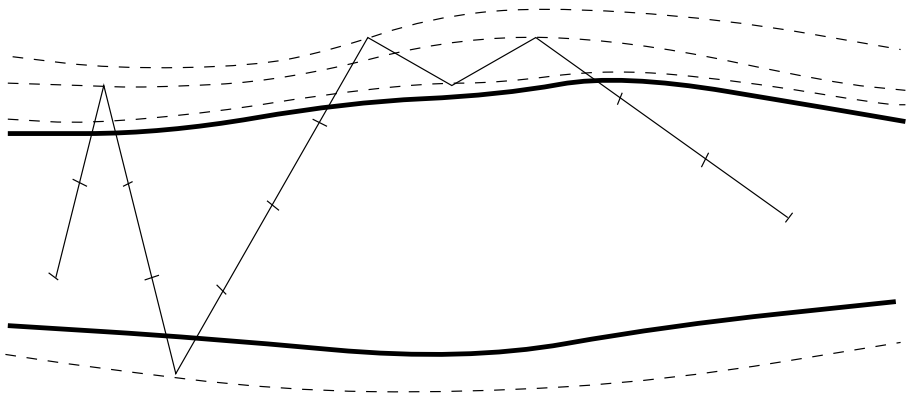

Figure 12: Reflection from outside points. Starting from the left, two reflections based on outside contours lead back inside the slice after the next step. The step after the third reflection is still outside the slice, so further reflections must be done. In this case, the trajectory eventually returns to the slice, and its endpoint would therefore be accepted.

This ideal reflection scheme is illustrated for a two-dimensional slice in Figure 10. Using the fact that the reflection transformation above has Jacobian one and is its own inverse, one can show that movement with reflection for some pre-determined duration leaves invariant the joint distribution of x (uniform within the slice) and p (rotationally symmetric, independent of x), so this way of sampling is valid, with no need for an acceptance test. One can also see from the figure how such motion can proceed consistently in one direction (until the end of the slice is reached), rather than in a random walk.

Ideal reflection is difficult to implement, however, as it requires precise calculation of where the current path intersects the boundary of the slice. Finding this point analytically might sometimes be possible, or we might try to solve for it numerically, but if the slice is not known to be convex, it may be difficult even to determine with certainty that an intersection point that has been found is in fact the first one that would be encountered. Rather than attempt such exact calculations, we can instead employ one of two approximate schemes, based on 'inside' or 'outside' reflection, although the trajectories these schemes produce must sometimes be rejected.

When stepping from x to $x' = x + wp$ takes us outside the slice, we can try to reflect from the last inside point, x, instead of from the exact point where the path intersects the boundary, using the gradient of $f(x)$ at this inside point. The process is illustrated in Figure 11. However, for this method to be valid, we must check that the reverse trajectory would also reflect at this point, by verifying that a step in the direction opposite to our new heading would take us outside the slice. If this is not so, we must either reject the entire trajectory of which this reflection step forms a part, or alternatively, set p and x so that we retrace the path taken to this point (starting at the inside point where the reflection failed).

Alternatively, when we step outside the slice, we can try to reflect from the outside point, x', based on the gradient at that point. A trajectory with several such reflections is shown in Figure 12. After performing a pre-determined number of steps, we accept the trajectory if the final point is inside the slice. Note that for this method to be valid, one must reflect *whenever* the current point is outside the slice, even if this leads one away from the slice rather than toward it. This will sometimes result in the trajectory never returning to the slice, and hence being rejected, but other times, as in the figure, it does return eventually.

Many variations on these procedures are possible. Above, it was assumed that values for y and p are are randomly drawn at the beginning of a trajectory, and then kept the same for many steps (apart from the changes to p when reflections occur). When using inside reflection, we might instead pick a new value for y more often, perhaps before every step, and we might also partially update p, as is done in Horowitz's (1991) variation on Hybrid Monte Carlo. When using outside reflection, the acceptance rate can be increased by terminating the trajectory when either some pre-set maximum number of steps have been taken, *or* some pre-set number of steps have ended inside the slice. When termination occurs for the latter reason, the final point will necessarily be inside, and the trajectory will be accepted. A full exploration of these variations is beyond the scope of this paper.

7 Discussion

As seen in this paper, the idea of slice sampling can be used to produce many Markov chain sampling schemes. In Figure 13, I attempt to summarize the characteristics of these schemes, and of some competing approaches for sampling from general distributions on continuous state spaces.

The first column indicates whether the method requires that derivatives of the (unnormalized) probability density be computable. Derivatives are needed by dynamical methods and reflective slice sampling, which limits their applicability. Adaptive rejection sampling (Gilks and Wild 1992; Gilks 1992) and overrelaxed slice sampling can take advantage of derivatives, but can operate without such information with only a moderate loss of efficiency — eg, when no derivatives are available, overrelaxed slice sampling can use bisection rather than Newton iteration to find the endpoints of the slice.

The second and third columns indicate how critical it is that tuning parameters be set to good values, and under what conditions these parameters can be set adaptively. Adaptive rejection sampling (ARS) for log concave distributions is very good in these respects — a parameter is needed for the size of the first step taken in search of a point on the other size of the mode, but subsequent steps can be made larger (eg, by doubling), so the effect of a poor initial step is not too serious; furthermore, this size parameter can be set adaptively. Parameter tuning is more of a problem when ARMS (Gilks, Best, and Tan 1995) is used for distributions not known to be log concave — a poor choice of parameters may have worse effects, and adaptive tuning is not allowed (Gilks, Neal, Best, and Tan 1997). Tuning is also a problem for simple Metropolis methods — proposing changes that are too small leads to an inefficient random walk, while proposing changes that are too large leads to frequent rejections. Using too small a stepsize with a dynamical method is not quite as

Sampling method	*Derivatives needed?*	*How critical is tuning?*	*Adaptive tuning allowed?*	*Can suppress random walks?*
ARS/ARMS	No (but helpful)	Low/Medium	If log concave	No
Simple Metropolis	No	Medium	No	No
Dynamical methods	Yes	High	No	Yes
Single-variable slice sampling	No	Low	If unimodal	No
Overrelaxed slice sampling	No (but helpful)	Low	If unimodal and endpoints exact	Yes
Reflective slice sampling	Yes	Medium/High	No	Yes

Figure 13: Characteristics of some general-purpose Markov chain sampling methods.

bad, since movement is not in a random walk, but too large a stepsize is disastrous, since the dynamical simulation becomes unstable, and very few changes are accepted. For both these methods, the size parameter must not be set adaptively.

Single variable slice sampling and overrelaxed slice sampling offer advantages over other methods in these respects. Whereas ARS allows adaptive tuning only for log concave distributions, adaptation is allowed when these slice sampling methods are applied to any unimodal distribution (provided the interval is expanded to the whole slice, and endpoints for overrelaxation are computed exactly). Furthermore, the tuning is probably less critical for slice sampling than for the other methods (apart from ARS), as discussed further below. For reflective slice sampling, however, tuning is at least moderately critical, though perhaps less so than for dynamical methods, and adaptive tuning is not allowed.

The final column indicates whether the method can potentially suppress random walk behaviour. This is important when sampling from a distribution with high dependencies between variables, as in such a situation, motion must proceed in small steps, and the difference in efficiency between diffusive and systematic exploration of the distribution can be very large. This is typical with neural network models (Neal 1996), for example.

Another way of exploring the differences between these methods is to see how well they work in various circumstances. The most favourable situation is when our prior knowledge lets us choose good tuning parameters for all the methods (eg, the width of a Metropolis proposal distribution or of the initial interval for slice sampling). The Metropolis algorithm with a simple proposal distribution will then move about the distribution fairly efficiently (although in a random walk), and will have low overhead, since it requires evaluation of $f(x)$ at only a single new point in each iteration. Single variable slice sampling will be comparably efficient, however, provided we stick with the interval chosen initially (ie, we set $m = 1$ in the stepping out procedure of Figure 3). There will then be no need to evaluate $f(x)$ at the boundaries of the interval, and if the first point chosen from this interval is within the slice, only a single evaluation of $f(x)$ will be done. If this point is outside the slice, further evaluations will be required, but this inefficiency corresponds to the possibility of rejection with the Metropolis algorithm. The two methods should therefore perform quite similarly. However, slice sampling will work better if it turns out that we mistakenly chose too large a width for the Metropolis proposal distribution and the initial slice sampling interval. This error will lead to a high rejection rate for the Metropolis algorithm, but the sampling procedure of Figure 5 uses the rejected points to shrink the interval, which is much more efficient if the initial interval was too large.

We might instead know that the conditional distributions are log concave, but not know how wide they are. Adaptive Rejection Sampling (ARS) then works very well, because its width parameter can be set adaptively. Single variable slice sampling will also work well, since in this situation it can also be tuned adaptively (provided no limit is set on the size of the interval). However, ARS does true Gibbs sampling, whereas the slice sampling updates do not produce points that are independent of the previous point. Such dependency is probably a disadvantage (unless deliberately directed to useful ends, as in overrelaxation), so ARS is probably better than single variable slice sampling in this context.

Suppose, however, that we know only that the conditional distributions are unimodal, but not necessarily log concave. We would then need to use ARMS rather than ARS, and would not be able to tune it adaptively, whereas we can still use single variable slice sampling with adaptive tuning. This will likely not be as good as true Gibbs sampling, however, which we should prefer if the conditional distribution happens to be one that can be efficiently sampled from. In particular, if slice sampling is used to sample from a heavy-tailed distribution, it may only infrequently move between the tails and the central region, since this transition can occur only when we move to a point under the curve of $f(x)$ that is as low as the region under the tails, but whose horizontal position is in the central region. However, there appears to be no general purpose scheme that avoids problems in this situation.

Finally, consider a situation where we do not know that the conditional distributions are unimodal, and have only a rough idea of an appropriate width parameter for a proposal distribution or initial slice sampling interval. Single variable slice sampling copes fairly well with this uncertainty. If the initial interval is too small it can be expanded as needed, either by stepping out or by doubling (which is better will depend on whether the faster expansion of doubling is worth the extra overhead from the acceptance test of Figure 6). If instead the initial interval is too big, it will be shrunk efficiently by the procedure of Figure 5. We might try to achieve similar robustness with the Metropolis algorithm by doing several updates for each variable, using proposal distributions with a range of widths. For example, if w is our best guess at an appropriate width, we might do updates with widths of $w/4$, $w/2$, w, $2w$, and $4w$. This may ensure that an appropriate proposal distribution is used some of the time, but it is unattractive for two reasons. First, the limits of the range (eg, from $w/4$ to $4w$) must be set *a priori*. Second, for this approach to be valid, we must continue through the original sequence of widths even after it is clear that we have gone past the appropriate one. These problems are not present with slice sampling.

In any of these situations, we might prefer to use a method that can suppress random walks. Dynamical methods such as Hybrid Monte Carlo (Duane, *et al* 1987) do this well for a wide range of distributions; reflective slice sampling may also work for a wide range of distributions, but preliminary indications are that is less efficient than Hybrid Monte Carlo, when both are tuned optimally. Overrelaxation is sometimes beneficial, but not always (whether it is or not depends on the types of correlation present). For problems where overrelaxation is helpful, overrelaxed slice sampling may often be the best approach to suppressing random walks. If the conditional distributions are unimodal, it offers the possibility of adaptive tuning. It does not require computation of derivatives. For some problems, the fact that overrelaxation updates one variable at a time will permit computational saving, in comparison with the simultaneous updates for dynamical and reflective methods.

Slice sampling methods have so far been used successfully by Frey (1997) for sampling latent variables in a neural network, and I have found that they work well for some Gaussian process models, in which the likelihood is a very complicated function of the parameters of the covariance function. However, experience on isolated examples is not very informative

as to the merits of these methods for routine and automated use. Experience with a wide variety of statistical problems will be needed to see how easily slice sampling and competing methods can be applied in practice.

Acknowledgements

I thank Brendan Frey, Gareth Roberts, Jeffrey Rosenthal, and David MacKay for helpful discussions. This research was supported by the Natural Sciences and Engineering Research Council of Canada.

References

Adler, S. L. (1981) "Over-relaxation method for the Monte Carlo evaluation of the partition function for multiquadratic actions", *Physical Review D*, vol. 23, pp. 2901-2904.

Barone, P. and Frigessi, A. (1990) "Improving stochastic relaxation for Gaussian random fields", *Probability in the Engineering and Informational Sciences*, vol. 4, pp. 369-389.

Besag, J. and Green, P. J. (1993) "Spatial statistics and Bayesian computation" (with discussion), *Journal of the Royal Statistical Society B*, vol. 55, pp. 25-37 (discussion, pp. 53-102).

Damien, P., Wakefield, J., and Walker, S. (1997) "Gibbs sampling for Bayesian nonconjugate and hierarchical models using auxiliary variables", preprint.

Diaconis, P., Holmes, S., and Neal, R. M. (1997) "Analysis of a non-reversible Markov chain sampler", Technical Report BU-1385-M, Biometrics Unit, Cornell University, 26 pages.

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

Edwards, R. G. and Sokal, A. D. (1988) "Generalization of the Fortuin-Kasteleyn-Swendsen-Wang representation and Monte Carlo algorithm", *Physical Review D*, vol. 38, pp. 2009-2012.

Frey, B. J. (1997) "Continuous sigmoidal belief networks trained using slice sampling", in M. C. Mozer, M. I. Jordan, and T. Petsche (editors) *Advances in Neural Information Processing Systems 9*, MIT Press.

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

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

Gilks, W. R. (1992) "Derivative-free adaptive rejection sampling for Gibbs sampling", in J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (editors), *Bayesian Statistics 4*, pp. 641-649, Oxford University Press.

Gilks, W. R., Best, N. G., and Tan, K. K. C. (1995) "Adaptive rejection Metropolis sampling within Gibbs sampling", *Applied Statistics*, vol. 44, pp. 455-472.

Gilks, W. R., Neal, R. M., Best, N. G., and Tan, K. K. C. (1997) "Corrigendum: Adaptive rejection Metropolis sampling", to appear in *Applied Statistics*.

Gilks, W. R. and Wild, P. (1992) "Adaptive rejection sampling for Gibbs sampling", *Applied Statistics*, vol. 41, pp. 337-348.

Green, P. J. and Han, X. (1992) "Metropolis methods, Gaussian proposals and antithetic variables", in P. Barone, *et al.* (editors) *Stochastic Models, Statistical Methods, and Algorithms in Image Analysis*, Lecture Notes in Statistics, Berlin: Springer-Verlag.

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

Higdon, D. M. (1996) "Auxiliary variable methods for Markov chain Monte Carlo with applications", ISDS Discussion Paper 96-17, 25 pages.

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

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

Mira, A. and Tierney, L. (1997) "On the use of auxiliary variables in Markov chain Monte Carlo sampling", preprint.

Neal, R. M. (1993) *Probabilistic Inference Using Markov Chain Monte Carlo Methods*, Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto, 140 pages. Obtainable from `http://www.cs.utoronto.ca/~radford/`.

Neal, R. M. (1994) "An improved acceptance procedure for the hybrid Monte Carlo algorithm", *Journal of Computational Physics*, vol. 111, pp. 194-203.

Neal, R. M. (1995) "Suppressing random walks in Markov chain Monte Carlo using ordered overrelaxation", Technical Report No. 9508, Dept. of Statistics, University of Toronto, 22 pages.

Neal, R. M. (1996) *Bayesian Learning for Neural Networks* (Lecture Notes in Statistics No. 118), New York: Springer-Verlag.

Roberts, G. O. and Rosenthal, J. S. (1997) "Convergence of slice sampler Markov chains", Technical Report No. 9712, Dept. of Statistics, University of Toronto, 21 pages.

Swendsen, R. H. and Wang, J.-S. (1987) "Nonuniversal critical dynamics in Monte Carlo simulations", *Physical Review Letters*, vol. 58, pp. 86-88.

Thomas, A., Spiegelhalter, D. J., and Gilks, W. R. (1992) "BUGS: A program to perform Bayesian inference using Gibbs sampling", in J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (editors), *Bayesian Statistics 4*, pp. 837-842, Oxford University Press.