JOURNAL OF COMPUTATIONAL PHYSICS 111, 194-203 (1994)

An Improved Acceptance Procedure for the Hybrid Monte Carlo Algorithm

RADFORD M. NEAL

Department of Computer Science, University of Toronto, 10 King's College Road, Toronto, Canada M5S 1A4

Received August 30, 1992; revised June 9, 1993

The probability of accepting a candidate move in the hybrid Monte Carlo algorithm can be increased by considering a transition to be between windows of several states at the beginning and end of the trajectory, with a particular state within the selected window then being chosen according to the Boltzmann probabilities. The detailed balance condition used to justify the algorithm still holds with this *procedure*, provided the start state is randomly positioned within its window. The *new procedure is shown empirically to significantly improve the* acceptance rate for a test system of uncoupled oscillators. It also allows expectations to be estimated using data from all states in the windows, rather than just states that are accepted. © 1994 Academic Press, Inc.

1. INTRODUCTION

The hybrid Monte Carlo algorithm of Duane, Kennedy, Pendleton, and Roweth [4] is a method of sampling from complex distributions, such as those encountered in statistical physics, that combines the advantages of dynamical methods [1] with those of the Metropolis Monte Carlo method [8]. For reviews of these and related methods, see [6, 11].

The problem is to simulate a system parameterized by a vector, $\mathbf{q}$, of dimension N, with a differentiable potential energy function, $E(\mathbf{q})$. This energy function induces a Boltzmann distribution over $\mathbf{q}$, for which the probability density is

$$p(\mathbf{q}) = Z_E^{-1} \exp(-E(\mathbf{q})), \tag{1}$$

where $Z_E = \int_{R^N} \exp(-E(\mathbf{q}))\, d\mathbf{q}$. (A temperature of one is assumed throughout, for simplicity.)

The aim of the simulation is to estimate the expectation of some function, $h(\mathbf{q})$, which is conventionally done using the formula

$$\langle h \rangle = \int_{R^N} h(\mathbf{q})\, p(\mathbf{q})\, d\mathbf{q} \approx \frac{1}{n} \sum_{i=0}^{n-1} h(\mathbf{q}_i), \tag{2}$$

where $\mathbf{q}_0, \mathbf{q}_1, \ldots, \mathbf{q}_{n-1}$ are obtained from the simulation and are each distributed according to the Boltzmann distribution for $\mathbf{q}$ (but are not, in general, independent).

I will first describe the dynamical approach to solving this problem, which suffers from systematic error in the sampling, and then describe the hybrid Monte Carlo method, which eliminates this error by accepting only some of the dynamical moves. Next, I present and justify a generalization of this algorithm in which moves are made between windows of states at the beginning and end of a trajectory, rather than between single states. I show that one advantage of this approach is that it allows all states in the windows to be used when estimating expectations, rather than just the accepted states (as in (2) above). Next, I describe situations in which the generalized algorithm will have a higher acceptance probability than the standard algorithm, and I empirically evaluate the acceptance rate for systems of uncoupled oscillators. Finally, I discuss two variations on the algorithm that may be useful in some circumstances.

2. THE DYNAMICAL METHOD

In the dynamical simulation method, we introduce a momentum vector, $\mathbf{p}$, which, like $\mathbf{q}$, has dimension N, and a Hamiltonian function, $H(\mathbf{q}, \mathbf{p})$, that incorporates both potential and kinetic energy:

$$H(\mathbf{q}, \mathbf{p}) = E(\mathbf{q}) + \tfrac{1}{2} |\mathbf{p}|^2. \tag{3}$$

We then seek to sample from the Boltzmann distribution that H induces on the phase space, $(\mathbf{q}, \mathbf{p})$, for which the density is

$$p(\mathbf{q}, \mathbf{p}) = Z_H^{-1} \exp(-H(\mathbf{q}, \mathbf{p})) = p(\mathbf{q})\, p(\mathbf{p}), \tag{4}$$

where $Z_H = \int_{R^N} \int_{R^N} \exp(-H(\mathbf{q}, \mathbf{p}))\, d\mathbf{q}\, d\mathbf{p}$, and

$$p(\mathbf{p}) = (2\pi)^{-N/2} \exp(-\tfrac{1}{2} |\mathbf{p}|^2). \tag{5}$$

0021-9991/94 $6.00

This sample is generated by simulating an ergodic Markov chain that has the Boltzmann distribution for $(\mathbf{q}, \mathbf{p})$ as its stationary distribution. Values of $\mathbf{q}$ from successive states of this Markov chain, whose marginal distribution is that of Eq. (1), are used to estimate $\langle h \rangle$ using Eq. (2). This Markov chain operates by alternating dynamical transitions with stochastic transitions.

The dynamical transitions consist of simulating the system for some predefined period in a fictitious time, τ, using Hamilton's equations,

$$\frac{d\mathbf{q}}{d\tau} = +\frac{\partial H}{\partial \mathbf{p}} = \mathbf{p} \tag{6}$$

$$\frac{d\mathbf{p}}{d\tau} = -\frac{\partial H}{\partial \mathbf{q}} = -\nabla E(\mathbf{q}). \tag{7}$$

These equations leave H invariant. Furthermore, the volume of a region of phase space remains constant as it evolves according to this dynamics (Liouville's theorem). In consequence, the dynamical transitions sample regions of constant H without bias.

The stochastic transitions allow regions with different values of H to be explored. They consist of replacing $\mathbf{p}$ with a value picked from its Boltzmann distribution (Eq. (5)). Such transitions clearly leave the Boltzmann distribution of $(\mathbf{q}, \mathbf{p})$ with respect to H invariant. The presence of these stochastic transitions will also usually be enough to ensure that the Markov chain is ergodic—i.e., that the system can move to any point in phase space.

It is desirable that the trajectories simulated in the dynamical transitions be long enough that they reach configurations almost independent of their starting configurations. This avoids the slow exploration, at the rate of a random walk, that would result if the direction of motion were frequently randomized by stochastic transitions.

In practice, the dynamics must be simulated with some finite step size. The "leapfrog" method is generally used, with the following steps being iterated some predefined number of times, L, with some specified step size, ε:

$$\mathbf{p}\left(\tau + \frac{\varepsilon}{2}\right) = \mathbf{p}(\tau) - \frac{\varepsilon}{2}\nabla E(\mathbf{q}(\tau)) \tag{8}$$

$$\mathbf{q}(\tau + \varepsilon) = \mathbf{q}(\tau) + \varepsilon \mathbf{p}\left(\tau + \frac{\varepsilon}{2}\right) \tag{9}$$

$$\mathbf{p}(\tau + \varepsilon) = \mathbf{p}\left(\tau + \frac{\varepsilon}{2}\right) - \frac{\varepsilon}{2}\nabla E(\mathbf{q}(\tau + \varepsilon)). \tag{10}$$

This discretized dynamics still preserves phase space volume exactly. However, with a finite ε, it does not leave H exactly constant. This will introduce some systematic error into the sampling.

Discretizations preserving phase-space volume that are accurate to higher-order than the leapfrog method are known [2, 3, 9, 10]; it is also possible to improve efficiency by exploiting special characteristics of the potential energy function [10]. Although I consider only the simple leapfrog method in this paper, the techniques I describe could be used in conjunction with these other methods.

3. THE HYBRID MONTE CARLO ALGORITHM

The systematic error of the dynamical method is eliminated in the hybrid Monte Carlo algorithm [4] by considering the end-point of the trajectory found with the leapfrog method to be merely a candidate for the next state of the Markov chain, to be accepted or rejected as in the Metropolis Monte Carlo algorithm [8].

Acceptance or rejection of the candidate state is based on the amount, ΔH, by which H for the candidate state exceeds H for the current state. The probability of acceptance, $a(\Delta H)$, is given by

$$a(\Delta H) = \min(1, \exp(-\Delta H)). \tag{11}$$

Thus candidate states with lower H are always accepted, while those with higher H are accepted with probability $\exp(-\Delta H)$. If the candidate state is rejected, the new state is the same as the current state (and is counted again in the average of Eq. (2)).

The validity of this procedure for producing a sample from the Boltzmann distribution is more easily seen if we imagine that the trajectory for a dynamical transition is computed using a value for ε whose sign is chosen at random, with positive and negative values being equally likely. The leapfrog method (Eqs. (8) to (10)) is time-reversible, so that a "forward" trajectory, with a positive ε, and a "backward" trajectory, with the corresponding negative ε, are inverses of each other, a fact crucial to the justification of the algorithm. In fact, usual practice is to always use a positive ε, since an effect equivalent to randomly choosing its sign is produced in any case by the randomization of the direction of $\mathbf{p}$ in the stochastic transitions.

To show that the Boltzmann distribution (Eq. (4)) is invariant under such dynamical transitions, it suffices to show that these transitions satisfy the condition known as "detailed balance"—that the probability of a transition from A to B occurring is the same as that for a transition from B to A, given that the start state is Boltzmann distributed.

To see that detailed balance holds, consider a small region of volume δV around the point $A = (\mathbf{q}_A, \mathbf{p}_A)$. Suppose that a forward trajectory from A leads to the point $B = (\mathbf{q}_B, \mathbf{p}_B)$. The other points in the region around A will lead to a region around B, which will also have volume δV,

since the leapfrog method preserves phase space volume. Due to time reversibility, a backward trajectory from B will lead to A. The detailed balance condition with respect to the regions around A and B can now be written as

$$p(\mathbf{q}_A, \mathbf{p}_A)\,\delta V \cdot \tfrac{1}{2} \cdot a(H(\mathbf{q}_B, \mathbf{p}_B) - H(\mathbf{q}_A, \mathbf{p}_A)).$$
$$= p(\mathbf{q}_B, \mathbf{p}_B)\,\delta V \cdot \tfrac{1}{2} \cdot a(H(\mathbf{q}_A, \mathbf{p}_A) - H(\mathbf{q}_B, \mathbf{p}_B)). \quad (12)$$

The left side of the above equation is the probability of moving from the region around A to the region around B. The first factor is the Boltzmann probability for being in the region around A at the start, the second factor ($\frac{1}{2}$) is the probability of selecting a forward direction for the trajectory, and the third factor is the probability that this trajectory will be accepted. The right side expresses the probability of moving from the region around B to the region around A in analogous fashion. The equality of these two probabilities is seen by substituting from Eqs. (4) and (11). The detailed balance condition can be similarly verified for the case where a backward trajectory from A is chosen.

The Boltzmann distribution is also invariant with respect to the stochastic transitions, which simply replace $\mathbf{p}$ with a value chosen from its Boltzmann distribution. Thus, if (as we expect) the Markov chain is ergodic, it will have the Boltzmann distribution as its unique stationary distribution.

4. AN ACCEPTANCE PROCEDURE USING WINDOWS

The standard hybrid Monte Carlo algorithm can be generalized to consider a window of states at the end of the trajectory as candidate destinations for a dynamical transition, rather than just a single end state. In order to maintain detailed balance, a possible move to this window must be considered in relation to an equal-sized window around the current state, with the location of the current state within that window being determined randomly. Due to this later requirement, the trajectory may have to be computed for some number of steps in the reverse of its primary direction. Acceptance or rejection of a move between windows is based on the total probability of the states they contain. Whichever window is selected, a particular state within that window is then picked according to the Boltzmann probabilities.

This procedure can significantly increase the probability of accepting a move and permits use of a more efficient method of estimating expectations, as will be discussed in Sections 6 and 7. First, though, I will define the algorithm in more detail here, and demonstrate its validity in Section 5.

To begin, values for the total number of steps in the trajectory, L, the base step size, ε_0, and the window size, W, are selected from some fixed distribution, with $1 \leqslant W \leqslant L+1$. A forward or backward direction, λ, for the trajectory is then chosen, with equal probabilities for $\lambda = +1$ and $\lambda = -1$, and an offset, K, for the current state within the start window is selected, with $K \in \{0, ..., W-1\}$, each possible value being equally likely.

A reverse portion of the trajectory is then computed by applying the leapfrog method with a step size of $\varepsilon = -\lambda\varepsilon_0$, beginning with the start state, $X(0)$. This is done for K iterations, producing states labeled $X(-1)$, ..., $X(-K)$. The start state is then restored, and the leapfrog method is applied with a step size of $= +\lambda\varepsilon_0$ for $L-K$ iterations, producing states $X(1)$, ..., $X(L-K)$.

The "reject" window at the front of the trajectory, around the current state, is defined as the set $\mathscr{R} = \{X(-K), ..., X(-K+W-1)\}$. The "accept" window at the far end of the trajectory is defined as $\mathscr{A} = \{X(L-K-W+1), ..., X(L-K)\}$. These windows may overlap.

The "free energy" for a window is defined as

$$F(\mathscr{W}) = -\log\left(\sum_{X \in \mathscr{W}} \exp(-H(X))\right). \quad (13)$$

The free energies are used to decide whether the next state will come from the accept window or the reject window. Using the acceptance function of Eq. (11), the accept window is selected with probability $a(\Delta F)$, where $\Delta F = F(\mathscr{A}) - F(\mathscr{R})$; otherwise we remain in the reject window (which contains the current state). Having decided on the window $\mathscr{W}$, a particular state within that window is then selected according to the probabilities

$$P(X) = \exp(-H(X) + F(\mathscr{W})). \quad (14)$$

The state selected becomes the next state in the Markov chain.

Implementatons of both the standard and generalized algorithms must somehow restore previously computed states whenever the next state selected for the Markov chain is not the end state of the trajectory, and, in the case of the generalized algorithm, after the reversed part of the trajectory has been computed. This may be done by retracing the trajectory, exploiting the reversibility of the leapfrog method (although one should note that reversibility is not absolutely perfect if the increments of Eqs. (8) to (10) are done using floating-point arithmetic). Alternatively, computation time will usually be reduced, at a cost in memory, if states that may be needed later are saved when they are first computed. For the generalized algorithm, one might think that all states in both the accept and reject windows would have to be saved with this approach, but, in fact, one need only save the start state, so that it can be restored after the reversed portion of the trajectory has been calculated, along with a single state from the accept window and a

single state from the reject window, one or the other of which will become the next state of the Markov chain.

In detail, let $F^i_{\mathscr{A}}$ be the free energy for the first i states that have been visited in the accept window, and let $C^i_{\mathscr{A}}$ be a state chosen according to the Boltzmann probabilities from among these first i states. These variables can be calculated incrementally as new states in the accept window are visited. To start, $F^0_{\mathscr{A}} = \infty$ and $C^0_{\mathscr{A}}$ is undefined. When we visit the ith state in the accept window, $(\mathbf{q}_i, \mathbf{p}_i)$, we can calculate

$$F^i_{\mathscr{A}} = -\log(\exp(-H(\mathbf{q}_i, \mathbf{p}_i)) + \exp(-F^{i-1}_{\mathscr{A}})) \tag{15}$$

$$C^i_{\mathscr{A}} = \begin{cases} C^{i-1}_{\mathscr{A}} & \text{with probability } \exp(-F^{i-1}_{\mathscr{A}} + F^i_{\mathscr{A}}) \\ (\mathbf{q}_i, \mathbf{p}_i) & \text{with probability } \exp(-H(\mathbf{q}_i, \mathbf{p}_i) + F^i_{\mathscr{A}}). \end{cases} \tag{16}$$

Once all states have been visited, we will have $F(\mathscr{A}) = F^W_{\mathscr{A}}$, and $C^W_{\mathscr{A}}$ will be a state picked from $\mathscr{A}$ according to the Boltzmann probabilities. Analogous variables, $F^i_{\mathscr{R}}$ and $C^i_{\mathscr{R}}$ are maintained for the reject window. Once all states have been seen, a decision as to whether to use the accept window or the reject window can easily be made, and, in either case, a state selected from the chosen window is available.

When $W = 1$, the generalized algorithm is equivalent to the standard hybrid Monte Carlo algorithm. When $W = L + 1$ the procedure reduces to simply picking a state from those anywhere along the trajectory in accordance with their Boltzmann probabilities. Some simplification in the implementation is then possible.

5. VALIDITY OF THE GENERALIZED ALGORITHM

To demonstrate the validity of the generalized algorithm, we need to show that the detailed balance condition holds for the dynamical transitions. As L, W, and ε_0 are chosen from a fixed distribution, independently of the current state, we may choose to regard them as fixed, since if detailed balance holds for transitions with any values of these parameters, it will hold for a mixture of such transitions. It will prove necessary to average over the values selected for λ and K, however.

Detailed balance will be proved separately for transitions to a state in the accept window and for those to a state in the reject window. If the two windows overlap, as they will if $W > (L+2)/2$, for some pairs of states there will be the possibility of transitions of either type. If detailed balance holds for the two types of transitions individually, however, it will also hold for the combination.

To prove detailed balance for transitions within the reject window, note first that the set of possible trajectories (for the various values of λ and K) that start at state A and that include state B in the reject window is the same as the set of trajectories that start at B and include A in the reject window. Here, a "trajectory" is defined as the set of states visited, along with the subsets of states that make up the accept and reject windows. In detail, if for the trajectory starting at $A = X_A(0)$, with direction λ_A and window offset K_A, we have $B = X_A(J)$ within the reject window, then the identical trajectory will be produced by starting at $B = X_B(0)$, with direction $\lambda_B = \lambda_A$ and window offset $K_B = K_A + J$, and $A = X_B(-J)$ will be in the reject window.

We can thus prove detailed balance separately for each such trajectory. The probability of being in a small region of volume δV around A, of then picking values for λ and K that generate a particular trajectory for which a state in the region of B is in the reject window, of then choosing to pick a state from the reject window, and of finally choosing the state in the region of B as the next state, is

$$p(\mathbf{q}_A, \mathbf{p}_A)\,\delta V \cdot \frac{1}{2} \cdot \frac{1}{W} \cdot (1 - a(F(\mathscr{A}) - F(\mathscr{R}))) \cdot \exp(-H(\mathbf{q}_B, \mathbf{p}_B) + F(\mathscr{R})). \tag{17}$$

The probability of generating the same trajectory starting from the region of B, and of then ending up in the region of A, is

$$p(\mathbf{q}_B, \mathbf{p}_B)\,\delta V \cdot \frac{1}{2} \cdot \frac{1}{W} \cdot (1 - a(F(\mathscr{A}) - F(\mathscr{R}))) \cdot \exp(-H(\mathbf{q}_A, \mathbf{p}_A) + F(\mathscr{R})). \tag{18}$$

These are readily seen to be equal upon substituting from Eqs. (4) and (11).

Detailed balance for transitions to a state in the accept window follows similarly, using the fact that the set of possible trajectories that start at state A and that include state B in the accept window is the same as the set of trajectories that start at B and include A in the accept window, except that in the later trajectories the accept and reject windows are exchanged. In detail, if for the trajectory starting at $A = X_A(0)$, with window offset K_A and direction λ_A, we have $B = X_A(J)$ within the accept window, $\mathscr{A}_A$, while A is in the reject window, $\mathscr{R}_A$, then the trajectory produced by starting at $B = X_B(0)$, with window offset $K_B = L - K_A - J$ and $\lambda_B = -\lambda_A$, will lead to $A = X_B(J)$ being in the accept window, $\mathscr{A}_B$, while B is in the reject window, $\mathscr{R}_B$. Furthermore, we will have $\mathscr{A}_A = \mathscr{R}_B$ and $\mathscr{R}_A = \mathscr{A}_B$.

The probability of being in a small region of volume δV around A, of then picking values for λ and K that generate a particular trajectory for which a state in the region of B is in the accept window, of then chosing to pick a state from the accept window, and of finally choosing the state in the region of B as the next state, is

$$p(\mathbf{q}_A, \mathbf{p}_A)\,\delta V \cdot \frac{1}{2} \cdot \frac{1}{W} \cdot a(F(\mathscr{A}_A) - F(\mathscr{R}_A)) \cdot \exp(-H(\mathbf{q}_B, \mathbf{p}_B) + F(\mathscr{A}_A)). \tag{19}$$

For the probability of generating the same trajectory, but with accept and reject windows exchanged, starting from the region of B, and of then picking a state in the region of A, we have

$$p(\mathbf{q}_B, \mathbf{p}_B)\,\delta V \cdot \frac{1}{2} \cdot \frac{1}{W} \cdot a(F(\mathscr{A}_B) - F(\mathscr{R}_B)) \cdot \exp(-H(\mathbf{q}_A, \mathbf{p}_A) + F(\mathscr{A}_B)). \tag{20}$$

Again, these are seen to be equal upon subsituting from Eqs. (4) and (11), remembering that $\mathscr{A}_A = \mathscr{R}_B$ and $\mathscr{R}_A = \mathscr{A}_B$.

6. IMPROVED ESTIMATION WITH THE GENERALIZED ALGORITHM

When using the generalized algorithm, it is possible to utilize all states in both the accept and the reject windows when estimating the expectation of a function, rather than just the states actually accepted. This can be done using an extension of a technique due to Kalos and Whitlock [5].

Rather than estimate the expectation of h using Eq. (2), one may instead obtain an unbiased estimate as follows:

$$\langle h \rangle \approx \frac{1}{n} \sum_{i=0}^{n-1} E[h(\mathbf{q}_i') \mid \mathbf{q}_i] \tag{21}$$

Here, $E[\cdot \mid \mathbf{q}_i]$ denotes expectation with respect to the hypothetical operation of a Monte Carlo procedure that generates a new state, $\mathbf{q}_i'$, from the current state $\mathbf{q}_i$, in such a fashion that $\mathbf{q}_i'$ has the desired Boltzmann distribution if $\mathbf{q}_i$ does. (This procedure might, but need not, be the same as the procedure used to generate $\mathbf{q}_{i+1}$ from $\mathbf{q}_i$.) In most cases of practical interest, it is likely that an estimate using Eq. (21) will have lower variance than that of Eq. (2). (However, this is not true in general when the $\mathbf{q}_i$ are dependent, due to the possibility that negative correlations among the $\mathbf{q}_i$ could lower the variance of the estimate of Eq. (2) below that which would be obtained with independent sampling, an advantage that might disappear when using Eq. (21).)

To apply this idea to the generalized hybrid Monte Carlo algorithm, we must first view the dynamical transitions of the Markov chain slightly differently, as applying to a state that includes not only the position, $\mathbf{q}$, and momentum, $\mathbf{p}$, and but also the trajectory parameters, K and λ, the latter two having no effect on the energy, and hence having a uniform Boltzmann distribution. The previously described procedure for generating a new state via a dynamical transition can now be described as follows. First, compute the trajectory from $\mathbf{q}_i$ and $\mathbf{p}_i$ using parameters K_i and λ_i. Second, select a state from the accept or reject windows of this trajectory, as previously described. As was shown in Section 5, if $\mathbf{q}_i$, $\mathbf{p}_i$, K_i, and λ_i are Boltzmann distributed before this operation, the selected $\mathbf{q}_i'$ and $\mathbf{p}_i'$ will also be Boltzmann distributed. Finally, to satisfy the requirement of leaving the entire state Boltzmann distributed, replace K_i and λ_i with new values chosen uniformly at random.

This is clearly equivalent to the previous description, in which new values for K and λ were selected *before* the trajectory was computed, since all that is relevant is that these two operations alternate. The advantage of the present description is that the distribution of $\mathbf{q}_i'$ with respect to the operation of this procedure is confined to states along a single trajectory, which would not be the case if K and λ were randomly chosen at the start. We can therefore easily estimate $\langle h \rangle$ using Eq. (21)—we just average the values of $E[h(\mathbf{q}_i') \mid \mathbf{q}_i]$ for all iterations, i, with the expectation simply being the weighted average of $h(\mathbf{q})$ for all states in the accept and reject windows of the trajectory computed from $\mathbf{q}_i$, the weight for each state being proportional to the probability it had of being accepted as the new state.

It appears preferable, however, to take advantage of the fact that the Monte Carlo operation with respect to which the expectation of Eq. (21) is taken need not be the same as that actally used to select the new state. For these expectations, we can therefore use the acceptance function

$$a(\Delta H) = 1/(1 + \exp(\Delta H)) \tag{22}$$

in place of that of Eq. (11). (One can readily verify that this acceptance function also leads to detailed balance being satisfied.) It is then easy to see that the probability of a state in the accept or reject window of the trajectory being chosen as the new state is simply proportional to its Boltzmann probability (except that if the windows overlap, states that are in both windows have two chances). Each term in the estimation formula of Eq. (21) then reduces to the average of $h(\mathbf{q})$ over all states in the accept and reject windows, with each state being given a weight proportional to its Boltzmann probability (except that states in both windows are given double weight). Note, however, that the *total* weight for all such states must be the same for all iterations, regardless of how the total Boltzmann probability for states in the trajectories may vary.

When the window size, W, is either $L + 1$ or $(L + 1)/2$, all states along the trajectory are used in estimating $\langle h \rangle$, with weights given by their Boltzmann probabilities. This eliminates one disadvantage of the hybrid Monte Carlo method with respect to uncorrected dynamical methods, in which one may also utilize all the states along the trajectories.

7. ACCEPTANCE PROBABILITY FOR THE GENERALIZED ALGORITHM

Two situations where the use of windows will increase the acceptance probability of the hybrid Monte Carlo algorithm are illustrated in Fig. 1.

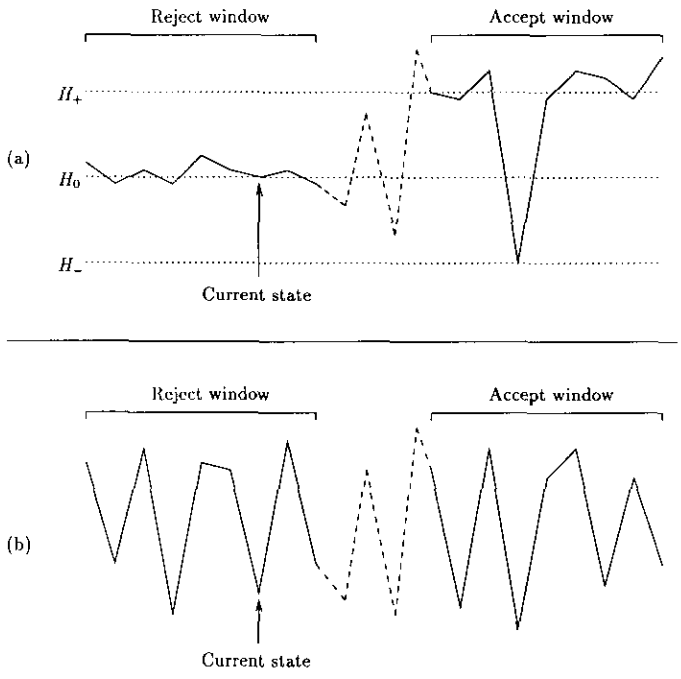

FIG. 1. Two situations where the use of windows increases the acceptance probability. The graphs show the change in total energy along two hypothetical trajectories.

In the trajectory of Fig. 1a, the energies of the states in the reject window, at the start of the trajectory, are all approximately equal to the energy of the current state, H_0. Most of the states in the accept window, at the end of the trajectory, have energies in the vicinity of H_+, much greater than that of the current state. However, one state has a much lower energy, H_-.

If the standard hybrid Monte Carlo algorithm is applied in this situation, with the end point of the trajectory randomly picked from the last W states shown, the probability of acceptance will be approximately $1/W$, since moves to states with energy H_+ would almost certainly be rejected, while a move to the one state with energy H_- would be accepted.

In contrast, if the generalized algorithm is applied with a window of size W, the move will always be accepted. The free energy of the reject window will be

$$F(\mathscr{R}) \approx -\log(W \exp(-H_0)) \approx H_0 - \log(W). \quad (23)$$

Assuming $H_+ \gg H_-$, the free energy of the accept window will be

$$F(\mathscr{A}) \approx -\log((W-1)\exp(-H_+) + \exp(-H_-)) \approx H_-. \quad (24)$$

Thus, if $H_0 - H_- > \log(W)$, the move will be accepted, and the particular state chosen within the accept window will almost certainly be the one with energy H_-. Note that it is essential to this example that a state in the accept window have lower energy than those in the reject window. If this is not the case, the acceptance probability is the same as for the standard algorithm.

The trajectory of Fig. 1b illustrates a different, perhaps more typical, situation where the use of windows also increases the acceptance probability. Here, both windows contain approximately equal numbers of low and high energy states. Note that the current state is likely to be one of low energy, since it is Boltzmann distributed.

With the standard algorithm, the end state might equally well be one of high energy or one of low energy. In the former case, the move would likely be rejected, while in the latter it would have a good probability of being accepted. The total acceptance probability will thus be around $\frac{1}{2}$.

If the generalized algorithm is used with a window size that is large compared to the time scale of the energy fluctuations, however, the free energy of both the accept and reject windows will be approximately equal, and the acceptance probability will be near one. The particular state chosen within the accept window will likely be one with low energy.

These examples are suggestive only—they show that the use of windows can be beneficial, but they do not indicate the magnitude of the benefit, nor whether there is any improvement in the asymptotic form of the time requirements as system size increases. Indeed, for fixed values of L, ε_0, and W, the acceptance probability declines exponentially with increasing system size, as for the standard algorithm. This scaling behaviour is due to the free energies of the windows, and hence their difference, being extensive quantities that increase in proportion to system size.

8. PERFORMANCE FOR A SYSTEM OF UNCOUPLED OSCILLATORS

To gain insight into the acceptance probability using the generalized algorithm and its scaling behaviour with system size, I have tested it empirically on systems of uncoupled oscillators. These simple systems are meant to model more complex systems in which the components are not completely independent, but which interact only weakly. The behaviour of the standard hybrid Monte Carlo algorithm for systems of uncoupled oscillators has been analysed in detail by Kennedy and Pendleton [7].

The potential energy function for such a system is

$$E(\mathbf{q}) = \frac{1}{2}\sum_{i=1}^{N} \omega_i^2 q_i^2. \quad (25)$$

In the Boltzmann distribution with respect to E, each q_i is independent and distributed as a Gaussian with zero mean

and standard deviation $1/\omega_i$. Since the operation of the hybrid Monte Carlo algorithm is unaffected by translation and rotation of the coordinates, these systems in fact model the behaviour of the algorithm as applied to any multivariate Gaussian distribution.

For the leapfrog method to be stable, with the error in H remaining bounded even for long trajectories, it is necessary for the step size to be less than $2/\omega_{\max}$, where $\omega_{\max}$ is the largest of the ω_i. For efficient exploration of the other coordinates, the average length of a trajectory in fictitious time, T_t, should be in the vicinity of $1/\omega_{\min}$, where $\omega_{\min}$ is the smallest of the ω_i. This choice of trajectory length results in the endpoint of the trajectory being approximately independent of the start point. If trajectories much shorter than this are used, the reversals of direction when the momenta are replaced in the stochastic transitions result in an inefficient random walk, while computing trajectories longer than this is at best a waste of time (at least if the estimation formula of Eq. (2) is used). With this choice of trajectory length, any dependencies among the q_i generated by the Markov chain will be largely the result of some trajectories being rejected, and the autocorrelation time for typical functions of the state will be around $1/(1-\rho)$, where ρ is the rejection probability.

I assume here that the approximate magnitudes of $\omega_{\min}$ and $\omega_{\max}$ and the general distribution of the ω_i are known and may be used to select appropriate values for the step size and trajectory length as described above. However, in order to avoid results that would not generalize to more complex systems, I assume that the *exact* values of these quantities are not known—in a realistic system, they may not even be precisely defined, since the components of the system may not be completely uncoupled. Accordingly, the step size (and hence the trajectory length as well) was varied slightly at random, in order to avoid results that depend on precise tuning.

I also assume that we are interested in systems for which the ratio $\omega_{\max}/\omega_{\min}$ is large. This ratio is a measure of the inherent difficulty of the problem, being (roughly) the number of leapfrog steps required to generate an independent configuration. In the experiments, the length in fictitious time of a trajectory was chosen to be large enough for the acceptance probability to have reached equilibrium. In a real system, however, the approriate trajectory length ($T_t \approx 1/\omega_{\min}$) might well be much longer than this.

Since chosing any particular value for T_t would be arbitrary, I have for the most part evaluated the algorithms on the assumption that T_t is very large. The "cost" of a particular combination of window size and average step size was accordingly taken to be

$$C = 1/(\bar{\varepsilon}(1-\rho)), \tag{26}$$

where ρ is the probability of a move being rejected and $\bar{\varepsilon}$ is the average step size (recall that the actual step sizes used will be slight perturbations around this average). This cost measure is proportional to the number of energy gradient evaluations needed to generate a given number of accepted moves of average length T_t, provided T_t is much greater than T_w, the length in fictitious time of the windows. Note that this cost measure places no value on transitions within the reject window, although these presumably improve sampling at least somewhat.

If T_w is comparable to T_t, then the above measure would not be appropriate, since it ignores the effort expended in computing those portions of the trajectory that lie before the current state and after the new state. An appropriate cost in this case would be $(1 + T_w/T_t)/(\bar{\varepsilon}(1-\rho))$.

I assume that for a given window length, T_w, and a given system size, N, the value of $\bar{\varepsilon}$ that minimizes C can feasibly be found and that this minimal value of C is thus the appropriate measure of the cost of the algorithm for a particular window length. If the rejection probability has the functional form

$$\rho = F(g(N)\,\bar{\varepsilon}^p) \tag{27}$$

then one can show by straightforward means that this minimum is reached at a value of ρ that is independent of N. The scaling properties of the algorithm can then be found by determining how $\bar{\varepsilon}$ must decrease as N increases in order to keep the rejection probability constant. Note that we are measuring cost as the number of evaluations of the energy gradient, not the time required for these evaluations. Computation time per evaluation will vary with system size in a manner which depends on the application, but which will generally add an additional factor of at least N to the total computation time as a function of system size.

I have tested the standard and the generalized algorithms on systems for which $N = 100$, 200, 400, 800, 1600, and 3200. In each case, the ω_i were selected randomly from the range 500 to 1000, with a uniform distribution for $\log(\omega)$. Although I assume that components with much smaller ω would also be present in a real system, such were not actually included in the simulation, in order to save time. Their inclusion would have had a negligible effect on the acceptance probability, since the accuracy of the leapfrog method is very high when $1/\omega$ is much greater than the step size.

An average trajectory length of $T_t = 1 \approx 1000 \times (1/\omega_{\max})$ was used. Several of the simulations were done with $T_t = 2$ as well, and the results confirmed that the acceptance probability had indeed reached equilibrium at $T_t = 1$.

For a given average step size, $\bar{\varepsilon}$, and for a given window size, W, the number of steps in the entire trajectory, L, was set so that $T_t = \bar{\varepsilon}(L - W + 1)$. (This calculation accounts for the average number of steps before and after the current and new states.) The trajectory was then computed with these

values of L and W and with a value of ε_0 randomly selected from the region within 1% of the given $\bar{\varepsilon}$.

Runs were done of the standard algorithm, for which $W=1$, and of the generalized algorithm with a window length of $T_w=0.20$, for which the number of states in the window was $W=T_w/\bar{\varepsilon}$. Some runs with $T_w=0.05$ and $T_w=0.10$ were done as well, with results that were similar to but not quite as good as those for $T_w=0.20$. Values for $\bar{\varepsilon}$ of ..., 0.000707, 0.000841, 0.0001000, 0.001189, ..., in geometric steps of $2^{1/4}$ were used. In each run, 1000 trajectories were generated starting from position and momentum coordinates picked from the Boltzmann distribution, independently for each trajectory. The proportion of rejected moves, $\hat{\rho}$, was taken to be an estimate of the rejection probability, ρ. The standard error for this estimate is ± 0.016 at $\rho=0.5$, declining to ± 0.009 at $\rho=0.1$ or $\rho=0.9$. An estimate, $\hat{C}$, for the cost follows from Eq. (26).

The results are shown in Fig. 2, which plots the estimated rejection probability and consequent cost for each value of N and for various values of $\bar{\varepsilon}$, for both the standard algorithm and the generalized algorithm with $T_w=0.20$. When the best value of $\bar{\varepsilon}$ is used in each case, the cost of the generalized algorithm is roughly half that of the standard algorithm, with some indication that the advantage of the generalized algorithm may be greater for the larger N. Interestingly, the rejection probability with the optimal $\bar{\varepsilon}$ is significantly lower for the generalized algorithm than for the standard algorithm.

In analysing this data further, we can first compare with the analytic results of [7]. They derive the following expression for the rejection probability of the standard algorithm applied to a system of uncoupled oscillators (adapted from their equation (2.10)):

$$\rho \approx \operatorname{erf}(\sqrt{N\bar{\varepsilon}^4\sigma/32}), \tag{28}$$

where $\sigma = N^{-1}\sum_i \omega_i^4 \sin^2(\omega_i T_t)/4$. Assuming that the small random variation in ε, and hence T_t, is enough to randomize the phases in this sum, and using the fact that $(1/2\pi)\int_0^{2\pi}\sin^2(x)\,dx = \frac{1}{2}$, we obtain that, for the experiment described here, the rejection probability should be

$$\rho \approx \operatorname{erf}(\sqrt{N\bar{\varepsilon}^4\bar{\sigma}/256}), \tag{29}$$

where $\bar{\sigma} = N^{-1}\sum_i \omega_i^4 \approx 3.38\times 10^{11}$. The rejection probability thus has (approximately) the form of Eq. (27). As system size increases, $\bar{\varepsilon}$ should be scaled as $N^{-1/4}$ in order to keep the rejection probability constant, and the cost will grow as $N^{1/4}$.

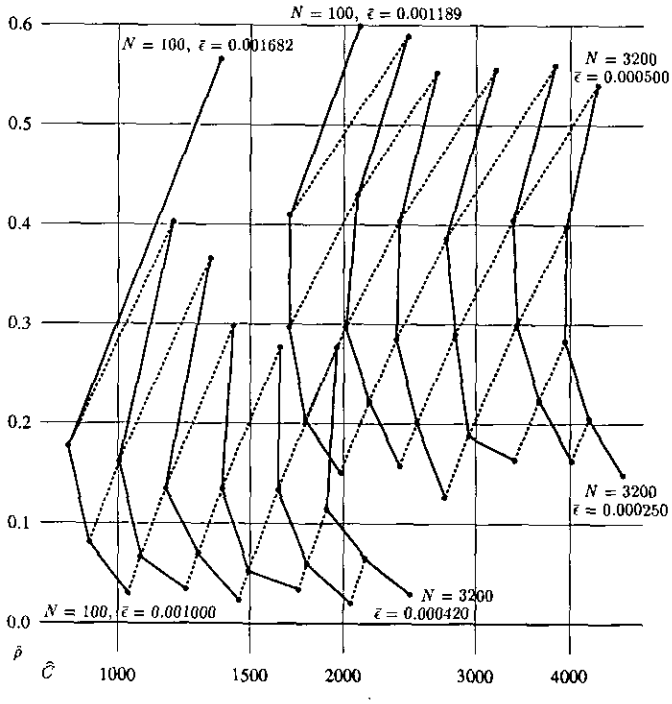

FIG. 2. Results with the standard and generalized algorithms. Each dot shows one run, with the estimated rejection probability, $\hat{\rho}$, plotted on the vertical axis, and the estimated cost, $\hat{C}$, plotted on the horizontal axis (on a logarithmic scale). Note that these are not independent measurements, since $\hat{C}=1/(\bar{\varepsilon}(1-\hat{\rho}))$. Runs of the same algorithm on systems of the same size (N), but using different step sizes ($\bar{\varepsilon}$) are connected by solid lines. Runs with the same $\bar{\varepsilon}$ but different N are connected by dotted lines. Runs of the standard algorithm (with $W=1$) are in the upper right; those of the generalized algorithm (with $T_w=0.20$) are in the lower left (there is a small degree of overlap).

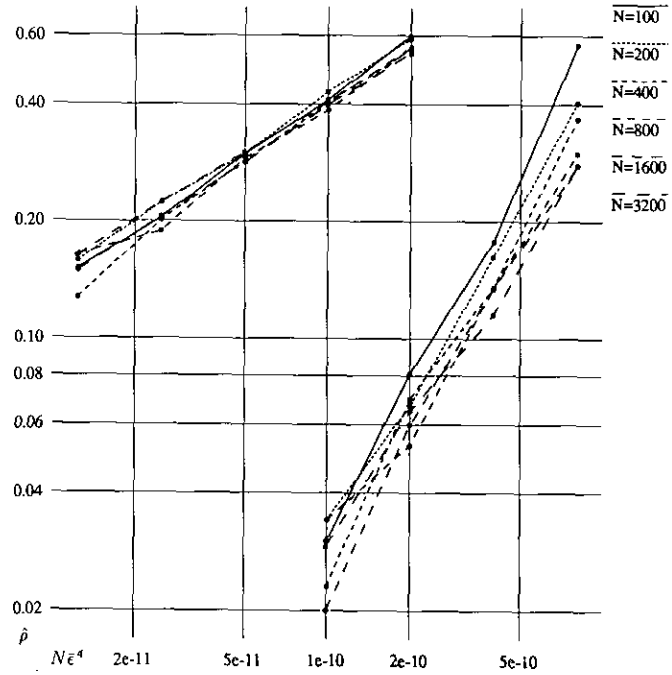

FIG. 3. Rejection probabilities for the two algorithms plotted under the assumption that they are functions of $N\bar{\varepsilon}^4$. Results for the standard algorithm are in the upper left; those for the generalized algorithm ($T_w=0.20$) are in the lower right. Both scale are logarithmic.

In Fig. 3, $\hat{\rho}$ is plotted against $N\bar{\varepsilon}^4$ for all runs of the standard algorithm. As expected, ρ appears to be a function of $N\bar{\varepsilon}^4$, as the spread in the data points is comparable to the standard error. (Note that the standard error is apparently larger for smaller values of ρ, due to the logarithmic scale.) Comparison with the predictions of Eq. (29) (not shown in the figure) shows a good fit, except for a slight departure for large values of the rejection probability.

We can hypothesize that the rejection probability of the generalized algorithm for a given window length will also be a function of $N\bar{\varepsilon}^4$. If this is the case, the cost for the generalized algorithm will also scale as $N^{1/4}$, although there could, as seen above, be an improvement in the constant factor over the standard algorithm. To test this hypothesis, Fig. 3 also shows $\hat{\rho}$ plotted against $N\bar{\varepsilon}^4$ for all runs of the generalized algorithm with $T_w = 0.20$. A tendency of the curves for the larger N to lie below those for the smaller N is apparent, leading one to reject this hypothesis.

We can consider the more general hypothesis that the rejection probability is a function of $N\bar{\varepsilon}^p$, for some p. This hypothesis can only be tested in conjunction with some hypothesis for the form of the function. From Fig. 3, it seems reasonable to consider the form $\log(\rho) = a + b\log(N\bar{\varepsilon}^p)$, for some constants a and b, though this model cannot be literally true, as for sufficiently large values of $N\bar{\varepsilon}^p$ it gives a rejection probability greater than one. In fact, the fit to the actual data is also not very good, but this is due entirely to the single data point at $N = 100$, $\bar{\varepsilon} = 0.001682$, for which the observed rejection rate has the anomalously high value of 0.566. It is plausible that with this small system size and large step size the particular ω_i that were chosen might heavily influence the observed rejection rate. This data point was therefore omitted for further analysis.

A maximum likelihood fit to the remaining data produced the estimates $\hat{p} = 4.47 \pm 0.06$, $\hat{a} = 24.8 \pm 0.6$, and $\hat{b} = 1.07 \pm 0.03$. The standard errors were obtained through a Monte Carlo investigation of the distributions of these estimators for data sets generated from the model, with the parameters set to their estimated values.

If the rejection probability does indeed scale in this way, with p as estimated above, the step size for the generalized algorithm should be scaled as $N^{-1/p} = N^{-0.224}$ to maintain a constant rejection probability, and the cost will consequently grow with system size as $N^{1/p} = N^{0.224}$, an improvement over the $N^{1/4}$ scaling of the standard algorithm. This result must be regarded as tentative, however. It seems possible that for very large N the window length, T_w, might have to increase at some rate in order to maintain good scaling behaviour. This would ultimately affect the cost, once T_w became comparable to T_t. Behaviour could also conceivably depend on the exact distribution of the ω_i. A theoretical analysis is thus needed to gain a better understanding of the performance of the generalized algorithm. It is clear, however, that at the very least, it can improve performance by a significant constant factor.

9. VARIATIONS ON THE ALGORITHM

Here, I will describe two variations on the generalized algorithm that may occasionally be useful.

First, when a move to the accept window is rejected, it is valid to simply remain at the current state, rather than selecting a new state from the reject window according to the Boltzmann probabilities. This follows from the fact that detailed balance was shown above to hold independently for the accept and reject transitions. Detailed balance thus continues to hold if all reject transitions are eliminated.

With this variation, the overhead of saving states is somewhat reduced. Typically, however, this overhead is small compared to the cost of evaluating the gradient of the energy, and its elimination may not be worth giving up the additional exploration provided by the reject transitions. Note that it is necessary in any case to visit all the states in the reject window, in order to compute its free energy.

A second variation may be useful when the ideal step size is not known a priori, or when the ideal step size varies from region to region. In such cases, it may be desirable to select a step size at random from a fairly broad distribution. Trajectories computed with a step size that is too large will result in large changes in H and are unlikely to be accepted.

To save computation, we can terminate such trajectories early, stopping whenever a single leapfrog step changes H by an amount, positive or negative, whose magnitude is greater than some threshold. The state reached after this large change in H is not included. Such truncated trajectories will have smaller than normal accept and/or reject windows, but examination of the proof of validity in Section 5 shows that treating them the same as normal trajectories still preserves detailed balance, although it is possible that with truncation the method might cease to be ergodic. (Note that the accept window for a truncated trajectory may be null, in which case the move is rejected. The reject window will always contain at least the current state.)

This variation is applicable to the standard hybrid Monte Carlo algorithm, where the window size is one. In this case, all truncated trajectories are rejected, with the current state remaining unchanged.

ACKNOWLEDGMENTS

I thank David MacKay and the anonymous referees for helpful comments. This work was supported by the Natural Sciences and Engineering Research Council of Canada, and by the Ontario Information Technology Research Centre.

REFERENCES

1. H. C. Andersen, *J. Chem. Phys.* **72**, 2384 (1980).
2. M. Campostrini and P. Rossi, *Nucl. Phys. B* **329**, 753 (1990).
3. M. Creutz and A. Gocksch, *Phys. Rev. Lett.* **63**, 9 (1989).
4. S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, *Phys. Lett. B* **195**, 216 (1987).
5. M. H. Kalos and P. A. Whitlock, *Monte Carlo Methods, Volume I: Basics* (Wiley, New York, 1986), Section 4.2.
6. A. D. Kennedy, "The Theory of Hybrid Stochastic Algorithms," in *Probabilistic Methods in Quantum Field Theory and Quantum Gravity*, edited by P. H. Damgaard *et al.* (Plenum, New York, 1990), p. 209.
7. A. D. Kennedy and B. Pendleton, *Nucl. Phys. B (Proc. Suppl.)* **20**, 118 (1991).
8. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, *J. Chem. Phys.* **21**, 1087 (1953).
9. R. D. Ruth, *IEEE Trans. Nucl. Sci.* **30**, 2669 (1983).
10. J. C. Sexton and D. H. Weingarten, *Nucl. Phys. B* **380**, 665 (1992).
11. D. Toussaint, *Comput. Phys. Commun.* **56**, 69 (1989).