BROOKHAVEN NATIONAL LABORATORY

May 1987 BNL-39747

CONF-8705129--1

MONTE CARLO ALGORITHMS FOR LATTICE GAUGE THEORY

Michael Creutz
Physics Department
Brookhaven National Laboratory,
Upton, NY 11973

BNL--39747
DE87 010045

ABSTRACT

In these lectures I review various techniques which have been used in numerical simulations of lattice gauge theories. After formulating the problem, I discuss the Metropolis *et al.* algorithm and some interesting variations. I then summarize the numerous proposed schemes for including fermionic fields in the simulations. Here I also treat Langevin, microcanonical, and hybrid approaches to simulating field theories via differential evolution in a fictitious time coordinate. I conclude with some speculations on new approaches to fermionic simulations.

Lectures presented at the Symposium/Workshop on Lattice Gauge Theory Using Parallel Processors, Beijing, China, 21 May-2 June, 1987.

This manuscript has been authored under contract number DE-AC02-76CH00016 with the U.S. Department of Energy. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

MONTE CARLO ALGORITHMS FOR LATTICE GAUGE THEORY

MICHAEL CREUTZ
Brookhaven National Laboratory, Upton, NY, U.S.A.

Abstract In these lectures I review various techniques which have been used in numerical simulations of lattice gauge theories. After formulating the problem, I discuss the Metropolis *et al.* algorithm and some interesting variations. I then summarize the numerous proposed schemes for including fermionic fields in the simulations. Here I also treat Langevin, microcanonical, and hybrid approaches to simulating field theories via differential evolution in a fictitious time coordinate. I conclude with some speculations on new approaches to fermionic simulations.

I. INTRODUCTION

Lattice gauge theory has become the primary theoretical tool for obtaining non-perturbative information about the gauge theory of quarks and gluons.[1] John Kogut's lectures at this meeting treat the basic motivations for this approach. To obtain quantitative numerical information on the solution of this theory, Monte Carlo simulation is now the dominant technique. My lectures review the various algorithms used in these simulations.

While Monte Carlo methods are quite old, their use by particle theorists to study quantum field theories is rather recent. The techniques are borrowed directly from the solid state physicists. This is possible because Feynman path integral formulation of quantum field theory exhibits a mathematical equivalence to classical statistical mechanics. Indeed, the path integral is formally a partition function.

The generic partition function we wish to simulate takes the form

$$Z = \int (dU)\,(d\psi)\,(d\psi^*)\,e^{-S(U,\psi,\psi^*)}. \tag{1.1}$$

Here the gauge fields are denoted U and the quark fields by ψ and ψ^*. Usually the action is quadratic in the quark fields, and we write

$$S(U,\psi,\psi^*) = S_G(U) + \psi^* M(U)\psi \tag{1.2}$$

where $M(U)$ contains both kinetic and mass terms for the fermionic fields and S_G represents the pure gauge part of the action and depends on the gauge fields U only. The quark fields are anticommuting objects, a fact that considerably complicates computer simulations. I will return to this point later, and for now consider only the pure gauge theory, for which eqn. 1.1 reduces to

$$Z = \int (dU)\, e^{-S_G}. \tag{1.3}$$

I will frame my discussion in terms of standard lattice gauge theory. Thus the variables U_{ij} are elements of the gauge group associated with the bonds of a hypercubic lattice. The integral over (dU) represents integration over all of these variables using the invariant group measure.[2] Although there are numerous interesting variations, I will restrict myself in these lectures to the standard Wilson action

$$S_G = \sum_p \frac{\beta}{N} \text{Re Tr} U_p \tag{1.4}$$

where the gauge group is $SU(N)$ and the sum runs over all elementary squares or plaquettes p on the lattice. The quantity U_p is an ordered product of the group elements U_{ij} around the square in question.

The Feynman path integral reduces quantum field theory to quadrature.[3] Indeed, the goal of the Monte Carlo approach is to numerically study integrals of the type appearing in eqn. (1.3). These integrals are, however, of rather large dimensionality. A 10^4 site lattice with the 8 parameter gauge group $SU(3)$ makes eqn. (1.3) a 32000 dimensional integral. The appearance of large numbers immediately suggests statistical techniques. Indeed, consider a familiar statistical system, say a glass of beer. The corresponding partition function would be an integral over the positions of all the contained molecules. However, to know the basic properties of beer, one need not consider all these possible configurations. All you need is a few dozen glasses of beer to know its important characteristics.

This, then, is the basic idea of the Monte Carlo approach. One replaces integrals over all possible fields by sums over a few typical configurations. These states are sampled with weighting proportional to the "Boltzmann weight" e^{-S_G}. Green's functions then follow from expectation values in an ensemble of such configurations. As the size of the ensemble goes to infinity, this average becomes exact. On a finite ensemble, fluctuations of an observable give a measure of the statistical error.

A Monte Carlo program for a lattice gauge theory creates a Markov chain of configurations. An initial state of the lattice variables is stored in the computer memory. Pseudo-random changes are then made in such a manner that the ultimate probability of encountering any configuration C with corresponding action $S_G(C)$ is proportional to e^{-S_G}. Each step in the algorithm is defined in terms of the probability $P(C',C)$ for taking configuration C into configuration C'. As this is a probability, it has the properties

$$P(C',C) \geq 0, \tag{1.5a}$$

$$\sum_{C'} P(C',C) = 1. \tag{1.5b}$$

In this and the following equations the sum is over all possible configurations; this is a shorthand notation for the integral over all U in eqn. (1.3).

The probability function $P(C',C)$ determining an algorithm is by no means unique; indeed, there are many different algorithms in common use for bringing lattices into equilibrium. One obvious necessary condition is that an equilibrium ensemble be left in equilibrium. Thus, considering P as a matrix, the Boltzmann weights should form an eigenvector with unit eigenvalue

$$e^{-S_G(C)} = \sum_{C'} P(C,C')\, e^{-S_G(C')}. \tag{1.6}$$

Remarkably, if the algorithm also has eventual access to all configurations, this condition is sufficient to take any ensemble closer to equilibrium.

To make this more precise, I introduce the concept of a distance between ensembles. Consider two ensembles, E and E', each containing a large number of configurations. Denote the probability density for configuration C in E or E' to be $p(C)$ or $p'(C)$ respectively. Then I define the distance between E and E' to be

$$||E-E'|| = \sum_C |p(C) - p'(C)|. \tag{1.7}$$

This non-negative quantity vanishes if and only if the ensembles have equal probability distributions.

Now suppose that E' resulted from the application of an algorithm satisfying eqn. (1.6) to E

$$p'(C) = \sum_{C'} P(C,C')\, p(C'). \tag{1.8}$$

We are interested in the relative distances of E and E' from the equilibrium ensemble E_{eq}, defined to have a probability distribution

$$p_{eq}(C) \propto e^{-S_G(C)}. \tag{1.9}$$

Combining eqns. (1.5-1.9) gives the desired result that the algorithm reduces the distance from equilibrium

$$\begin{aligned} &||E' - E_{eq}|| \\ &= \sum_C |\sum_{C'} P(C,C')\,(p(C') - p_{eq}(C'))\,| \\ &\leq \sum_{C,C'} P(C,C')\,|(p(C') - p_{eq}(C'))\,| \\ &= ||E - E_{eq}||. \end{aligned} \tag{1.10}$$

Note that if $P(C,C')$ is ergodic, that is if there is non-vanishing probability to reach any configuration, then this inequality is strict whenever E is not in equilibrium. This also implies uniqueness for the stationary distribution.

To insure that an algorithm has the equilibrium distribution as an eigenvector, most schemes used in practice are based on a sequence of steps, each satisfying a condition of detailed balance

$$P(C',C)\, e^{-S_G(C)} = P(C,C')\, e^{-S_G(C')}. \tag{1.11}$$

Summing over C' and using eqn. (1.5b) immediately gives eqn. (1.6). In words, when the algorithm is applied to an equilibrium ensemble, eqn. (1.11) implies that the rate of taking configurations C to C' will equal the rate of taking C' to C. Thus there will be a steady state and equilibrium is maintained. Detailed balance is not, however, a necessary condition. Indeed, equilibrium could still be a steady state with an algorithm generating a net circulation amongst the configurations.

Note that in general the product of two different steps each satisfying detailed balance does not. Usual Monte Carlo treatments update the lattice variables in a prescribed sequence. Thus a full sweep of a lattice will not in general satisfy detailed balance. Nevertheless, this is irrelevant to establishing equilibrium, which only requires the eigenvector condition of eqn. (1.6).

In the following section I present some results of simple simulations using various algorithms. To simplify programming, these all used skew periodic boundary conditions on a $7 \times 7 \times 7 \times 6$ lattice. The links in any given direction were updated in a checkerboard style, with all those eminating in a positive direction from odd sites being updated before those from even sites. Where error bars are shown, they were obtained by repeating the respective experiments several times.

Skew periodic or helical boundary conditions are convenient for rapidly finding neighbors. Consider working on a $N_x \times N_y \times N_z \times N_t$ lattice. A site at integer coodinates (x, y, z, t) can be uniquely associated with the integer

$$j = x + y\,(N_x - 1) + z\,(N_x - 1)\,(N_y - 1) + t\,(N_x - 1)\,(N_y - 1)\,(N_z - 1)\,. \tag{1.12}$$

The neighbors in any particular direction are found by merely shifting j by an appropriate amount. Note, however, that then a shift of x by N_x is equivalent to shifting y by one, and so forth. The lattice is truely periodic only in the time direction. Note that to divide sites into two classes according to their parity, as on a checkerboard, requires N_x, N_y, and N_z to be odd and N_t even.

Most present supercomputers are vector machines, which means that they are particularly efficient in doing a repeated computation on long strings of numbers (vectors) occupying consecutive memory locations in the machine. This feature is easily utilized in lattice gauge theory where the various steps involved in a Monte Carlo updating can be performed on sequences of independent gauge variables. This also makes it convenient to have subroutines do often needed tasks such as multiplying strings of matrices together. Structuring a program in this way, with long inner loops, also helps performance on serial machines because of reduced overhead.

For the following, it is useful to have a simple measure of the correlation between two lattices A and B with corresponding links $(g_l)_A$ and $(g_l)_B$. For this purpose I define

$$C\left(A,B\right)=\frac{1}{n_l N}\sum_l \mathrm{ReTr}\left(\left(g_l^{-1}\right)_A\ \left(g_l\right)_B\right). \qquad (1.13)$$

where the sum is over all links l and n_l is the total number of links. This quantity is unity when A and B are the same, and vanishes for uncorrelated lattices. I will be considering lattice B obtained from A through a few applications of various Monte Carlo algorithms. The speed with which this correlation drops to zero is then one indicator of the efficiency of an algorithm.

II. METROPOLIS, HEAT BATH, AND OVERRELAXATION ALGORITHMS

Early in the history of Monte Carlo simulation, Metropolis, *et al.*[4] presented a particularly simple way to enforce detailed balance by an acceptance criterion on trial changes. Consider updating a single gauge variable U. The Metropolis *et al.* procedure begins by considering a trial new value U' to replace U. This is selected with an arbitrary probability distribution $P_{T,U}\left(U'\right)$. Here the subscript U on P is a reminder that the trial change can in general depend on the old element. Finally, the change to U' is accepted with a conditional probability

$$P_A=\min\left[1\,,\frac{P_{T,U'}\left(U\right)}{P_{T,U}\left(U'\right)}\times\frac{\exp\left(-S_G\left(U'\right)\right)}{\exp\left(-S_G\left(U\right)\right)}\right]. \qquad (2.1)$$

To implement this, a random number uniformly distributed between 0 and 1 is generated. If this number is less than P_A, then the change is made. If this conditional probability is not met, then the change is rejected and the old value of U is kept.

This construction automatically satisfies the detailed balance condition. When $U' \neq U$, the overall probability $P(U', U)$ for taking U to U' is $P_{T,U}(U')$ times P_A. Multiplying by $e^{-S_G(U)}$ gives

$$P(U', U)\, e^{-S_G(U)} = \min\left[P_{T,U}(U')\ e^{-S_G(U)},\ P_{T,U'}(U)\ e^{-S_G(U')}\right]. \qquad (2.2)$$

Detailed balance is manifested in the symmetry of this expression under exchange of primed and nonprimed variables.

As usually implemented, U' is found by multiplying U by a group element h which is chosen with a probability distribution peaked around the identity and with equal probability for h and h^{-1}. For example, this can be done by choosing h from a table which contains the inverse of each of its elements. This particular method of choosing the trial change has the symmetry property

$$P_{T,U}(U') = P_{T,U'}(U). \qquad (2.3)$$

This is convenient because the ratio of P_T factors in eqn. (2.1) is then unity and can be ignored.

This approach contains an essential dependence on two parameters. The first represents the average distance the element h lies from the identity. If this distance is too large, the trial energies will likely be large and the changes will rarely be accepted. On the other hand, if h always lies too close to unity, the changes will usually be accepted, but their small size will make the exploration of phase space rather slow. Lore is for a compromise with an acceptance rate of order 50%.

The second parameter of this standard approach is the number of trial changes attempted on any link before proceeding to the next. In most statistical mechanics problems this is taken to be one; however, for gauge theories the interaction is rather complicated, requiring considerable arithmetic to calculate the environment of the link being updated. In terms of real computer time, it is often of value to test several trial elements, during which time the multiplication of neighboring links need not be repeated. Although the number of such "hits" is a useful parameter in this standard application of the Metropolis *et al.* algorithm, I will shortly discuss a variation of the algorithm where but a single hit is best.

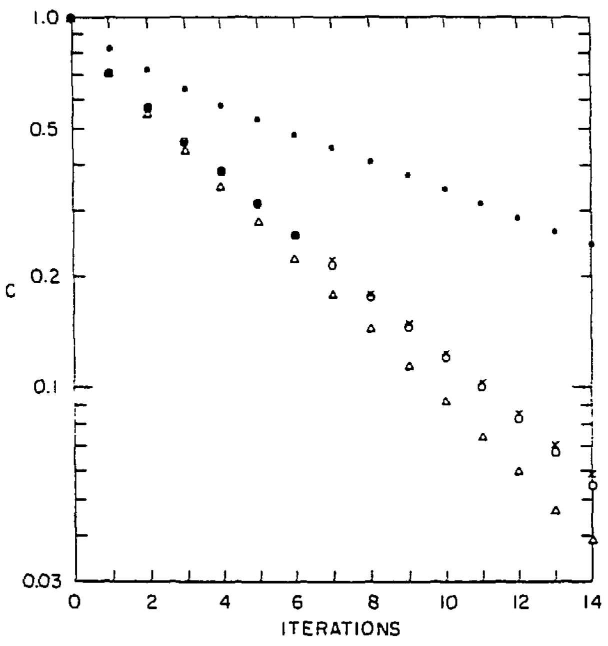

Fig. 2.1: The correlation between two lattices as a function of the number of simulation iterations separating them. The gauge group is $SU(3)$ at $\beta = 2.3$. The solid points, the crosses, and the open circles represent the standard Metropolis *et al.* algorithm with 10, 64, and 128 hits per link, respectively. The open triangles represent the overrelaxation algorithm discussed in the text.

Figure 2.1 shows the fall off of the interlattice correlation from eqn. (1.13) as a function of the number of Monte Carlo iterations separating the lattices A and B. One interation represents a sweep updating all the lattice variables. Here the gauge group is $SU(3)$ and $\beta = 6.0$. For these experiments the initial lattice had been equilibrated with multiple sweeps with a 10 hit algorithm. Here I show runs with 10, 64, and 128 "hits" or trial changes for each element before moving to update the neighbors. In addition this figure shows the performance of the overrelaxation algorithm which I will discuss shortly. These runs used trial elements U' selected by multiplying U with a matrix h chosen with probability

$$P(h) \propto \exp(k\ \mathrm{ReTr}h)\,. \tag{2.4}$$

I chose $k = 2\beta$ because, as I will exhibit momentarily, this empirically optimizes the correlation decrease, at least for this value of β. Although the runs in this figure are consistent with approaching exponentials, it might be dangerous to assume that this continues. There could be hidden long time correlations which emerge upon further running.

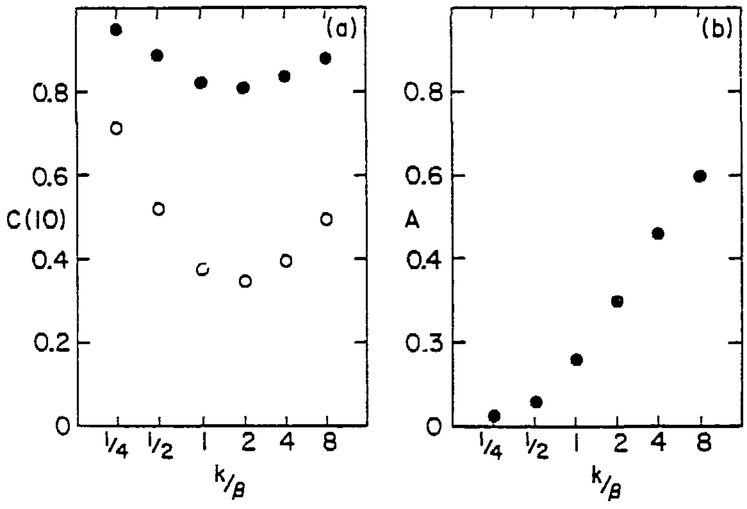

Fig. 2.2a: The correlation between two lattices separated by 10 iterations as a function of the bias parameter k. The solid and open circles are for one and ten hits per link, respectively. b: The acceptance per hit as a function of the bias parameter.

Figure 2.2a shows the correlation between two lattices separated by 10 lattice sweeps as a function of the parameter k appearing in eqn. 2.4 to bias the size of the trial changes. This figure shows results with both one and ten hits per link. Note that the optimum bias parameter is approximately 2β independent of the number of hits. Figure 2.2b shows the acceptance per hit as a function of the bias parameter. The optimum acceptance is approximately 30% for this value of β and for this prescription for choosing the trial changes.

The "heat bath" algorithm[5] is a special case of the generalized Metropolis *et al.* approach, wherein the trial element is taken randomly from the entire group manifold but with a weighting proportional to the Boltzmann factor. In this case the factors appearing in the acceptance criterion of equation (2.1) cancel and the trial element is always accepted. The result is equivalent to the limit of taking a large number of repetitions, or "hits," of any ergodic algorithm to a single variable. This algorithm is in some sense the most intuitive, being equivalent to placing the lattice variables successively into contact with a thermal bath at the corresponding temperature. It is readily implemented for simple group manifolds such as $SU(2)$.

I will now discuss a third method[6] for choosing the trial element U'. Suppose I have some simple way to find a group element U_0 which approximately minimizes the action $S_G(U)$. Suppose further that U_0 is obtained with no direct use of the element U; that is, U_0 only has a dependence on the remaining lattice variables. An interesting trial element for a Metropolis *et al.* updating lies on the "opposite side" of this element U_0 from the old value U. In particular, consider taking

$$U' = U_0\, U^{-1}\, U_0. \tag{2.5}$$

Note that this construction also satisfies the symmetry relation of eqn. (2.3). Thus, just as in the usual application of the Metropolis *et al.* algorithm, the acceptance or rejection of this element follows solely from the comparison of a random number with the exponential of the resulting action change.

The selection in eqn. (2.5) is motivated by overrelaxation ideas as discussed in the context of Monte Carlo simulation by Adler and Whitmer.[7] There are two intuitive arguments for this idea. The simplest is that the trial element is placed rather far from the old value without exacting a large energy penalty. Thus one might expect a rather rapid flow through phase space. A second argument is based on the overrelaxation idea used in minimization schemes such as used in solving linear equations. The position of minimum action for a given variable is indirectly influenced by the variable itself. When the neighbors were updated, they assumed values which tended to accomodate the position of the current variable. If however, that variable were allowed to float, they would in general move away and one might expect that the best value for the variable being updated might lie somewhat further away than the position of lowest energy with the neighbors held fixed.

The only rigorous requirement on finding U_0 is that it be independent of the old link U. There is clearly an advantage in keeping the selection procedure simple. For $SU(3)$ fields, I determine g_0^{-1} from a Gram-Schmidt orthogonalization process on the matrix interacting with the element being updated. In figure 2.1 I show the performance of this algorithm in comparison with the previously discussed standard approach. For all runs in this figure the correlation appears to be monotonically decreasing, with the overrelaxation algorithm decreasing the fastest. Indeed, noting the small change in going from 64 to 128 hits, the figure suggests that the new algorithm outperforms a heat bath. This is true even with the naive method for projecting onto the group. I find the acceptance rate for the trial changes is 57%.

In some cases, most notably with the groups $SU(2)$ and $U(1)$, the matrix interacting with a link is always proportional to a group element. Projecting onto this particular element results in U' of eqn. (2.5) having exactly the same energy as U. Then the Metropolis *et al.* prescription will always accept the change, and the algorithm is deterministic and microcanonical. This causes two minor complications. First, the total energy of the system is fixed and thus will not relax to any value other than where it is initially set. Second, the algorithm is actually independent of the temperature β^{-1}. Indeed, as with other microcanonical algorithms, the temperature should be measured during the simulation with some sort of thermometer, such as an average kinetic energy,[8] using an auxiliary variable with simple dynamics,[9] or from a dynamical equation involving both the temperature and measurable correlation functions.[10]

This issue can be avoided if desired by putting a small amount of randomness into U_0. For example, U_0 could be the product of a deterministic estimate of the element minimizing the action with a random element h chosen near the identity. If h has a small probability of lying anywhere in the group, this would also eliminate possible worries about ergodicity. Nevertheless, I have done limited studies which suggest that correlation times tend to increase with additional noise in U_0.

Although not microcanonical for $SU(3)$, the algorithm is approximately so when the system is fairly ordered. Indeed, when all group elements are unity, eq. 2.5 does not suggest any change. Thus, while the algorithm appears to move quite rapidly through phase space, it does not rapidly cross energy shells. Indeed, the plaquette correlation was studied in Ref. 6, and was found to decorrelate at comparable rates with this algorithm and a ten hit standard algorithm.

Perhaps the greatest advantage of the overrelaxation approach is its computational speed due to simplicity. For both $SU(2)$ and $SU(3)$ Ref. 6 obtained a link decorrelation rate per iteration comparable to a heat bath. Although the heat-bath algorithm is rather easily implemented for $SU(2)$, this is not the case for $SU(3)$ and thus most lattice gauge simulations have used a standard Metropolis *et al.* procedure with of order 10 trial changes on any link before proceeding to the next. Using eqn. (2.5) for the trial changes performs best with but a single hit; indeed, further attempts to change a given variable will just return to earlier trials. Furthermore, the construction of the trial element takes only minimally more computation than a single hit in a standard Metropolis *et al.* application. This advantage may be even greater for spin systems where there is substantially less overhead involved in calculating the interacting neighborhood of a variable being updated.

III. INTRODUCING FERMIONS

Fermionic fields provide several challenging obstacles to the field of lattice gauge theory. One appears already at the level of formulating an appropriate action. Here we have the notorious doubling of species appearing in the simplest schemes incorporating chiral symmetry. In these lectures on Monte Carlo methods I will ignore this issue and assume that one has an acceptable lattice transcription of the Dirac equation as manifested in the matrix M appearing in eqn. (1.2).

The problem I will address in some detail is the difficulties in simulating the partition function of the coupled fermion-gauge system

$$Z = \int (dA)\,(d\psi)\,(d\psi^*)\ \exp\left(-S_G(A) - \psi^* M(A)\,\psi\right). \tag{3.1}$$

As I will be concentrating on fermionic details, I will ignore the technicality that the gauge fields are group elements and simply write them as A.

The essence of the problem lies in the fact that the fermionic fields are not numbers, and that therefore the exponentiated action cannot be regarded as a probability. Instead, the fields ψ and ψ^* are anticommuting Grassmann variables, integration over which I now briefly review.

I begin by considering a set $\{\psi_i\}$ of anticommuting Grassmann variables

$$[\psi_i, \psi_j]_+ \equiv \psi_i\psi_j + \psi_j\psi_i = 0. \tag{3.2}$$

Generalizing complex conjugation to include these variables, I adopt the convention that corresponding to each ψ_i, I have another independent Grassmann variable ψ_i^*. Furthermore, I postulate

$$\begin{aligned}(\psi_i^*)^* &= \psi_i \\ (\psi_1 \ldots \psi_n)^* &= \psi_n^* \ldots \psi_1^*.\end{aligned} \tag{3.3}$$

If I consider just a single variable ψ, a general function $f(\psi)$ can be expanded with just two terms

$$f(\psi) = f_0 + \psi f_1. \tag{3.4}$$

To define integration over an anticommuting variable, I demand the properties of linearity and invariance under a translation of variables. These are summarized in the axioms

$$\int d\psi\,(f(\psi)\,\alpha + g(\psi)\,\beta) = \left(\int d\psi f(\psi)\right)\alpha + \left(\int d\psi g(\psi)\right)\beta \tag{3.5a}$$

$$\int d\psi f(\psi) = \int d\psi f(\psi + \psi'). \tag{3.5b}$$

This is sufficient to imply that for the function in eqn. (3.4)

$$\int d\psi f(\psi) = K f_1 \tag{3.6}$$

where the normalization K is undetermined. I adopt the convention $K = i$ so that

$$\begin{aligned}
&\int d\psi\ \psi = i \\
&\int d\psi\ 1 = 0 \\
&\int d\psi^* d\psi\ \psi^* \psi = 1.
\end{aligned} \tag{3.7}$$

Note that under multiplicative rescaling a Grassmann integral behaves as

$$\int d\psi f(\psi a) = \left(\int d\psi f(\psi)\right) a. \tag{3.8}$$

This can be written in the heuristic form $d(\psi a) = (d\psi)/a$.

For integration over several anticommuting variables, we have

$$\int d\psi_1 \ldots d\psi_n \psi_1 \ldots \psi_n = i^n (-1)^{n(n-1)/2}. \tag{3.9}$$

The analog of eqn. (3.8) in this case is

$$\int (d\psi) f(M\psi) = |M| \int d\psi f(\psi) \tag{3.10}$$

where M is an arbitrary invertable matrix, $|M|$ is its determinant, and $(d\psi)$ denotes $d\psi_1 \ldots d\psi_n$. Note that eq. (3.1) immediately implies the Matthews-Salam[11] formula for a fermionic gaussian integral

$$\int (d\psi^* d\psi)\ e^{\psi^* M \psi} = |M| \tag{3.11}$$

where $(d\psi^* d\psi) = d\psi_1^* d\psi_1 \ldots d\psi_n^* d\psi_n$.

Eqn. (3.11) provides an easy way out of the difficulty that our partition function is not an ordinary integral. Indeed, we explicitly integrate out the fermions to convert eqn. (3.1) to

$$Z = \int (dA)\ |M|\ e^{S_G}. \tag{3.12}$$

This is now an integral over numbers and therefore in principle amenable to Monte Carlo attack. For the remainder of these lectures

I will assume that the fermions have been formulated such that M is a positive matrix and thus the integrand in eqn. (3.12) can be regarded as proportional to a probability measure. If this is not so, one can always double the number of fermionic species, using $M^\dagger$ for the extra ones, thus replacing M by $MM^\dagger$.

Direct Monte Carlo attack of the partition function in eqn. (3.12) is still not practical because of the large size of the matrix M. In our compact notation, this is a square matrix of dimension equal to the number of lattice sites times the number of Dirac components times the number of internal symmetry degrees of freedom. Thus, it is typically a tens of thousands by tens of thousands matrix, precluding any direct attempt to calculate its determinant. It is, however, generally an extremely sparse matrix because most actions in practice do not directly couple distant sites. All the Monte Carlo algorithms used in practice for simulation of this problem make essential use of this fact.

IV. PSEUDOFERMIONS

Fucito, Marinari, Parisi, and Rebbi[12] proposed a simple approximate method for calculating changes in the determinant of the matrix M. They begin by rewriting eqn. (3.12) in the form

$$Z = \int (dA)\ e^{-S_{PF}}. \tag{4.1}$$

where

$$S_{PF} = S_G(A) - \mathrm{Tr}\ \log M(A) \tag{4.2}$$

For a Metropolis *et al.* updating scheme one needs to know the change in the action upon a trial change of A. As a first approximation, consider taking only small changes in the gauge field so that the change in the action can be determined from its first derivative with respect to A

$$\frac{dS_{PF}}{dA} = \frac{dS_G}{dA} + \mathrm{Tr}\left(M^{-1}\frac{dM}{dA}\right). \tag{4.3}$$

The quantity $\frac{dM}{dA}$ is easily calculated for a local M. The inverse of the matrix M is estimated using

$$\left(M^{-1}\right)_{ij} = \langle \xi_j^*\ \xi_i\rangle \tag{4.4}$$

where the expectation value is over fields ξ, called pseudofermions, and distributed with weighting

$$P(\xi) \propto \exp(-\xi^* M \xi) . \tag{4.5}$$

A standard Monte Carlo simulation is used to give a set of N_c configurations of the ξ fields to estimate this expectation value. This simulation is normally done only once per full sweep of the lattice variables. This is not a major new assumption because a small step size approximation is already being made in using only the first derivative of the action to calculate the changes in the action.

This algorithm, and indeed most of the fermionic algorithms used in practice, is not exact because of the approximation of small changes in A. The step size is a parameter in the standard applications of the Metropolis *et al.* algorithm. For example, it is determined by the parameter k appearing in eqn. (2.4). That the step size is sufficiently small can in principle be determined by comparing results for several values and doing an extrapolation to zero. Unfortunately, the amount of computer time necessary for a pseudofermionic simulation is sufficiently large that this check is rarely made.

Another possible source of error with this approach appears if the pseudofermionic fields are not calculated with the appropriate distribution. This would happen with insufficient equilibration time during the Monte Carlo simulation from which they are obtained. This error can in principle be eliminated by a trick if M is the square of a simple operator, say $M = DD^\dagger$. In this case consider first generating a random vector χ with a gaussian distribution

$$P(\chi) \propto e^{-\chi^\dagger \chi} \tag{4.6}$$

As all components of χ are uncorrelated, it can be rather quickly generated. Then a simple change of variables gives a properly distributed pseudofermionic field

$$\chi = D^\dagger \xi \tag{4.7}$$

This equation can in principle be solved by some interative algorithm such as the conjugate gradient method. This trick replaces the convergence of a Monte Carlo updating of the pseudofermionic fields with a potentially tedious inversion. I mention it here because, as I will discuss later, this use of gaussion random numbers is potentially quite useful with other fermionic algorithms as well.

The finite number N_c of configurations of pseudofermionic fields used to estimate the expectation value in eqn. 4.4 introduces a random error into the estimate of the inverse of M. These errors, however, average out in the final extrapolation of observables to zero step size. This is a point I will return to later when I discuss an algorithm which interpolates between pseudofermions and the Langevin approach.

V. SOME EXACT ALGORITHMS

The pseudofermionic approach involves an approximation of finite step size. A similar approximation is inherent in the Langevin and microcanonical approaches to be discussed later. A simple but time consuming algorithm with the usual statistical but no systematic errors was presented by Weingarten and Petcher.[13] They observe that by introducing another set of complex scalar fields ϕ one can rewrite eqn. 3.12 in the form

$$Z = \int (dA)\,(d\phi^*\;d\phi)\exp\left(-S_G - \phi^* M^{-1}\phi\right). \tag{5.1}$$

Thus a successful fermionic simulation would be possible if one could obtain configurations of fields ϕ and A with probability distribution

$$P\,(A,\phi) \propto \exp\left(-S_G - \phi^* M^{-1}\phi\right). \tag{5.2}$$

Ref. 13 notes that while M^{-1} is the inverse of an enormous matrix, one really only needs $\phi^* M^{-1}\phi$, which is just one matrix element of this inverse. Indeed, there exist reasonably efficient iterative schemes for finding the inverse of a large matrix applied to a single vector. Thus it was proposed to directly simulate the partition function in eqn. (5.1) using a Gauss-Seidel algorithm to calculate $M^{-1}\phi$. It now appears that the conjugate gradient algorithm may be somewhat preferable for this inversion.

The conjugate gradient algorithm to solve the equation $\xi = M^{-1}\phi$ works by finding the minimum over ξ of the function $|M\xi - \phi|^2$.

The solution is iterative; starting with some ξ_0, a sequence of vectors ξ_i is obtained by successively moving in directions d_i which maximize the decrease of this function subject to the orthogonality condition $d_i M^\dagger M d_j = 0$ whenever $i \neq j$. This last condition helps to eliminate useless oscillations in undesirable directions. The algorithm is guaranteed to converge to the correct minimum in a number of steps equal to the dimension of the matrix.

In practice, at least when the correlation length is not large, the conjugate gradient inversion adequately converges in a number of iterations which does not grow with the lattice size. As each step involves a sum over the vector, which has length proportional to the lattice volume, this means that the conjugate gradient step takes a time which grows with the volume of the system. Thus the algorithm of Ref. 13 is expected to require computer time which grows as the square of the volume of the lattice. Such a severe growth has precluded use of this algorithm on any but the smallest lattices. Nevertheless, it does show the existance of an exact algorithm with considerably less computational complexity than would be required for a repeated direct evaluation of the determinant of the fermionic matrix.

Two other exact, but also volume squared, algorithms were presented in Ref. 14. To use a Metropolis scheme to find a configuration of A fields with distribution

$$P_{eq}(A) \propto |M(A)| e^{-S_G(A)}, \tag{5.3}$$

requires knowledge of how the determinant $|M|$ changes when A is replaced by a trial value A'. Actually one needs only the ratio of the old and the new determinants, and this can be calculated as a pseudofermionic expectation value in two ways. First, if we construct an ensemble of fields ξ with distribution

$$P(\xi) \propto e^{-\xi^* M(A) \xi} \tag{5.4}$$

then we have

$$\frac{|M(A')|}{|M(A)|} = \frac{1}{\langle \exp\left(-\xi^* \left(M(A') - M(A)\right) \xi\right) \rangle}. \tag{5.5}$$

Alternatively, if we construct the fields ξ using the trial A'

$$P(\xi) \propto e^{-\xi^* M(A') \xi}, \tag{5.6}$$

then we have

$$\frac{|M(A')|}{|M(A)|} = \langle \exp\left(-\xi^* \left(M(A) - M(A')\right)\xi\right)\rangle. \qquad (5.7)$$

Both approaches involve a Monte Carlo to find the ensemble of ξ fields inside the Monte Carlo determination of the A fields. Thus they are also volume squared algorithms.

Grady[15] found a particularly intriguing variation on the second of these two approaches. In particular, he showed that an ensemble average was unnecessary in this "look ahead" scheme where the probability for ξ is determined from the trial field A'. Consider a trial change A' chosen with a probability distribution $P_{T,A}(A')$. Solely to simplify the following equations, assume that this trial probability is symmetric under interchange of A and A', in analogy with eqn. (2.3). Then generate a single ξ field with probability distribution as given in eqn. (5.6). The prescription is to accept the change with probability

$$P_{acc} = \min\left[1,\ \exp\left(S_G - S'_G + \xi^*\left(M' - M\right)\xi\right)\right]. \qquad (5.8)$$

Here I use the shorthand notation S_G, S'_G, M, and M' for $S_G(A)$, $S_G(A')$, $M(A)$, and $M(A')$, respectively.

To justify this procedure, consider the overall probability for taking A to A'

$$P(A \to A') = P_{T,A}(A') \frac{1}{Z_\xi} \int (d\xi^* d\xi)\, e^{-\xi^* M' \xi} P_{acc}. \qquad (5.9)$$

Here I have defined the normalization factor for the ξ integral

$$Z_\xi = \int (d\xi^* d\xi)\, e^{-\xi^* M' \xi} \ \propto\ |M'|^{-1}. \qquad (5.10)$$

Multiplying eqn. (5.9) by the equilibrium distribution in eqn. (5.3) and combining things gives

$$P_{eq}(A)\ P(A \to A') \propto |M||M'|P_{T,A}(A')$$
$$\min\left[e^{-S_G - \xi^* M' \xi},\ e^{-S'_G - \xi^* M \xi}\right]. \qquad (5.11)$$

Remembering the symmetry of P_T, we see that this expression is symmetric under interchange of primed and non-primed variables.

This is precisely the statement of detailed balance for the equilibrium distribution from eqn. (5.3).

The fact that a large ensemble of ξ fields is not required is a definite advantage of this approach. Nevertheless, it still contains a Monte Carlo inside a Monte Carlo to obtain the equilibrated ξ field. Thus as an exact algorithm this still requires computer time growing as the system volume squared.

It appears that all known exact algorithms for simulating fermionic fields require volume squared times. To avoid this, most fermionic schemes used in practice involve some sort of approximation. As noted above for the pseudofermionic method, this usually involves an expansion in the size of the variable changes. To eliminate systematic errors in principle requires an extrapolation to the limit of vanishing step size.

The advantage of making small steps lies in the fact that a time consuming step such as a conjugate gradient inversion or a Monte Carlo generation of pseudofermionic fields need only be done once per sweep of all the gauge variables. In essence, this is an attempt to eliminate a Monte Carlo inside a Monte Carlo. Once one is making small steps anyway, there is no particular loss in using algorithms formulated in terms of a differential evolution. This is the basis of both the Langevin [16,17] and the microcanonical[18] methods for including fermionic fields in lattice gauge simulations.

VI. LANGEVIN, MICROCANONICAL, AND HYBRID SCHEMES

Both the Langevin and microcanonical algorithms for lattice gauge theory are formulated as differential equations for evolution in a fictitious "time" τ. While these approaches are applicable for the pure gauge theory, their main interest appears with fermionic simulations because a differential evolution permits time consuming conjugate gradient inversions to be done only once per sweep of the lattice variables. Although there is considerable overlap of the material in this section with the lectures of Kogut and Parisi, I include this discussion for completeness and because repetition from another point of view is sometimes helpful.

Rather than in terms of field theory, I frame this discussion in the context of a single degree of freedom, the coordinate x of a particle of mass m moving in one dimension. This is mainly for fun, because the basic ideas of these algorithms are nothing more than generalizations of Newton's equation. I will also treat the Langevin and microcanonical approaches together as limits of a more general hybrid formalism. There exists an extensive literature on stochastic differential equations; for a recent review emphasizing nonequilibrium effects see Ref. 19.

I begin by considering a particle moving in a potential $V(x)$. Newton's equation for the particle motion is

$$m\frac{d^2x}{d\tau^2} = -\frac{\partial V}{\partial x}. \tag{6.1}$$

I now doctor this motion by adding two terms. First I add a drag slowing the particle down with a force proportional to its velocity. This will tend to damp out any motion until the particle lies at a minimum of the potential. To keep things moving, I then add a random noise to the system. Thus consider the equation

$$m\frac{d^2x}{d\tau^2} = -\frac{\partial V}{\partial x} - \alpha\frac{dx}{d\tau} + \left(\frac{2\alpha}{\beta}\right)^{1/2} \eta(\tau). \tag{6.2}$$

where α and β are parameters. Here the noise $\eta(\tau)$ formally satisfies

$$\langle \eta(\tau)\,\eta(\tau')\rangle = \delta(\tau - \tau'). \tag{6.3}$$

How the $\eta(\tau)$ is actually defined will become clearer momentarily when I make the evolution in τ discrete. I have written the coefficient of the noise as $(2\alpha/\beta)^{1/2}$ with a certain amount of hindsight. It is convenient to introduce the momentum p of the particle and rewrite this second order equation as two first order equations

$$\begin{aligned} \frac{dp}{d\tau} &= -\frac{\partial V}{\partial x} - \frac{\alpha p}{m} + \left(\frac{2\alpha}{\beta}\right)^{1/2} \eta(\tau), \\ \frac{dx}{d\tau} &= \frac{p}{m}. \end{aligned} \tag{6.4}$$

For simulation purposes the fictitious time is discreetly made discrete. Thus consider taking steps of size ϵ in τ. In one such step, p and x

at time τ will become p' and x' at time $\tau+\epsilon$. Eqn. 6.4 would thus transcribe to read

$$\begin{aligned} p' &= p + \epsilon\left(-\frac{\partial V}{\partial x} - \frac{\alpha p}{m} + \left(\frac{2\alpha}{\beta}\right)^{1/2}\eta\right), \\ x' &= x + \frac{\epsilon p'}{m}. \end{aligned} \tag{6.5}$$

Note that I have written the updating of x such as to use the new value p' of the momentum. This amounts to alternately updating the coordinates and the momenta of the system. Such a "leap frog" procedure effectively treats these variables at interleaved times. In the deterministic limit this advantageous technique serves to eliminate $O\left(\epsilon^2\right)$ errors in the evolution.

The quantity η is to be obtained from a random number generator with probability distribution $\rho(\eta)$. The properties required of $\rho(\eta)$ are simply specified in terms of its moments

$$\begin{aligned} \int d\eta\ \rho(\eta)\ \eta^j &= 1\ , \qquad j=0 \\ &= 0\ , \qquad j=1 \\ &= 1/\epsilon\ , \quad\ j=2 \\ &\le O\left(\epsilon^{-j/2}\right)\ , \ j>2. \end{aligned} \tag{6.6}$$

In this equation the first part indicates that the probability distribution is normalized, the second balances positive and negative noise, the third normalizes the delta function in eqn. (6.3), and the fourth eliminates any nasty tails from the distribution. In many discussions the noise is considered as gaussian, but this is not generally necessary.

To proceed, I frame the discussion in terms of ensembles of particle coordinates and momenta. As discussed in section I, a necessary condition for any simulation algorithm is that it leave the equilibrium ensemble unchanged. Thus I am interested in finding invariant ensembles under the evolution of eqn. (6.5).

Consider an ensemble with a probability density $P(x,p)$ of finding a state with given coordinate x and momentum p. Updating the states gives a new ensemble with probability distribution

$$\begin{aligned} P'(x',p') &= \int dx\, dp\, P(x,p)\, P(x,p \to x',p') \\ &= \int dx\, dp\, d\eta\, \rho(\eta)\;\; P(x,p) \\ &\times \delta\left(p' - p - \epsilon\left(-\frac{\partial V}{\partial x} - \frac{\alpha p}{m} + \left(\frac{2\alpha}{\beta}\right)^{1/2} \eta\right)\right) \\ &\times \delta\left(x' - x - \frac{\epsilon p'}{m}\right) \end{aligned} \tag{6.7}$$

A little algebra gives the result

$$P'(x,p) = P(x,p) + \epsilon\, \times$$
$$\left[\left(\frac{\partial H}{\partial x}\frac{\partial P}{\partial p} - \frac{\partial H}{\partial p}\frac{\partial P}{\partial x}\right) + \alpha\left(\frac{1}{\beta}\frac{\partial^2 P}{\partial P^2} + \frac{p}{m}\frac{\partial P}{\partial p} + \frac{1}{m}P\right)\right] + O\left(\epsilon^2\right) \tag{6.8}$$

Here I have defined the Hamiltonian corresponding to the original Newton's equation of eqn. 6.1

$$H = \frac{p^2}{2m} + V(x) \tag{6.9}$$

In deriving eqn. (6.8) it is necessary to keep terms of order η^2 because of the $1/\epsilon$ in the third part of eqn. (6.6).

Eqn. (6.8) is equivalent to a Fokker-Planck equation for the evolution of the probability density $P(x,p)$. It is now easily verified that to order ϵ a stationary distribution for this evolution is the simple Boltzmann weight

$$P'(x,p) = P(x,p) = \exp\{-\beta\left(\frac{p^2}{2m} + V(x)\right)\}. \tag{6.10}$$

When α is non zero so that the algorithm is ergodic, the arguments of section I imply the uniqueness of this solution. Note that this distribution factors into a function of p alone times a function of x. Thus the equilibrium distributions of p and x are independent.

We see that repeated updating of an ensemble with the stochastic differential equation of eqn. (6.2) will eventually give thermal equilibrium at inverse temperature β. To apply the technique to our gauge theory problem and obtain a distribution of fields as in eqn. 5.2, we merely generalize, replacing the variable x with the fields A and ϕ and replacing the potential $\beta V(x)$ with the action $S_G + \phi^* M^{-1}\phi$. Note that calculating the term involving $\frac{\partial V}{\partial x}$ will require, among other things, the evaluation of

$$\frac{\partial}{\partial A}\phi^* M^{-1}\phi = -\phi^* M^{-1}\frac{\partial M}{\partial A}M^{-1}\phi. \qquad (6.11)$$

Thus we will need to know $M^{-1}\phi$, which requires a conjugate gradient or equivalent inversion every time step.

It is interesting to consider various limits of this stochastic evolution. First, suppose that in eqn. (6.2) I had not included the drag term. This situation follows from taking α to zero with β varying proportionally to keep the stochastic term. Thus with noise but no drag we go to infinite temperature; that is, the random force will, on the average, increase the system energy without bound. Alternatively, if I include the drag but not the noise, the system will drop into a minimum energy state at effectively zero temperature. A finite temperature simulation requires the presence of both the drag and noise terms.

Note that the distribution in eqn. 6.10 is independent of the parameter α. Thus there really is a class of many possible algorithms. One of these corresponds to taking parameter m to zero. This can be effected by simultaneously adjusting α and rescaling the units of time. In this case eqn. (6.2) becomes first order and represents the usual Langevin equation as used in Refs. [16] and [17].

$$\frac{dx}{d\tau} = -\frac{\partial V}{\partial x} + \left(\frac{2}{\beta}\right)^{1/2} \eta(\tau). \qquad (6.12)$$

For later reference, I write this in the discrete form for evolving x to x' in one time step of length ϵ

$$x' = x - \epsilon \times \left(\frac{\partial V}{\partial x} + \left(\frac{2}{\beta}\right)^{1/2} \eta\right). \qquad (6.13)$$

Another interesting limit corresponds to taking α to zero while holding β constant. This case removes both the drag and noise terms, and returns simply to Newton's equation. This is the microcanonical approach, first advocated for pure gauge theories by Callaway and Rahman [8] and proposed for fermionic simulations in Ref. 18. In this case the algorithm has no explicit dependence on β. Indeed, as mentioned in section II, microcanonical algorithms require the temperature to be determined after the fact by some sort of thermometer, such as the average kinetic energy $\frac{1}{2}kT = \langle\frac{p^2}{2m}\rangle$. To change the temperature, one should start with a different total initial energy, which remains constant during the evolution.

Intermediate values of α represent hybrid algorithms which interpolate between the Langevin and microcanonical approaches. An alternative hybrid approach was recently proposed by Duane and Kogut,[20] and is discussed in Kogut's lectures at this meeting. They advocate updating with a microcanonical scheme for some number of iterations and then doing a step where all the momenta touch a heat bath. Eqn. (6.10) shows that the momenta in equilibrium are gaussianly distributed; thus, the latter step consists of replacing them all with new gaussian random numbers.

When the microcanonical updating time is small, this approach is easily related to the Langevin equation. To see this, consider first replacing p with a gaussianly distributed random number and then update the system microcanonically for a total time δ. This will take the coordinate x into

$$x' = x + \frac{p\delta}{m} - \frac{\delta^2}{2m}\frac{\partial V}{\partial x} + O\left(\delta^3\right). \tag{6.14}$$

If the microcanonical updating time δ is small enough that the $O\left(\delta^3\right)$ effects are negligible, then this evolution is identical to that in eqn. (6.13) where $\frac{\delta^2}{2m}$ plays the role of ϵ and $\left(\frac{\beta}{m\epsilon}\right)^{\frac{1}{2}} p$ represents the noise η.

With these hybrid approaches, one could adjust together both the step size ϵ and either the parameter α or the refreshing frequency in such a manner as to hold the finite step errors in observables at an acceptable size. Then an optimum algorithm would minimize the number of steps required to decorrelate lattices under a measure such as defined in eqn. (1.13).

VII. ANOTHER ALGORITHM

In this section I discuss another fermion algorithm recently presented by Gavai and myself.[21] The approach has similarities with the Langevin evolution but is based on a small step-size limit of the Metropolis *et al.* scheme. A simple modification permits an interpolation to the pseudofermionic algorithm.

The goal is to generate an ensemble of configurations of fields A and ϕ distributed as in eqn. (5.2). To explicitly insure the positivity of the fermionic matrix M, assume that it is a square

$$M = DD^\dagger \tag{7.1}$$

This effectively doubles the number of fermionic species, one interacting with A via $D(A)$ and the other via $D^\dagger(A)$. I will later mention a possible way to remove this doubling. With this form for M, the desired probability distribution for A and ϕ is

$$P(A,\phi) \propto \exp\left(-S_G - \phi^* \left(D^\dagger\right)^{-1} D^{-1}\phi\right). \tag{7.2}$$

The algorithm consists of alternate sweeps through the ϕ and A fields. The ϕ updating is particularly simple and makes use of the gaussian random number trick mentioned at the end of section IV. First generate a random vector χ with gaussian weight

$$P(\chi) \propto e^{-\chi^\dagger \chi} \tag{4.6}$$

I now change variables and construct

$$\phi = D\chi. \tag{7.3}$$

This will be distributed with the desired probability

$$P_\phi \propto \exp\left(-\phi^* \left(D^\dagger\right)^{-1} D^{-1}\phi\right). \tag{7.4}$$

The Jacobian factor associated with the change of variables in eqn. (7.3) is irrelevant as the fields A are being held fixed during this step.

This construction is computationally fast because the individual components of χ are independent and because the matrix D is assumed to be local. Thus we rapidly obtain a new ϕ field independent of its old value. This trick for updating ϕ is also used in the implementation of the Langevin algorithm in Ref. 16. Actually, the remainder of the algorithm does not explicitly need ϕ. Although I could eliminate this field and consider only χ, the discussion is simpler in terms of the coupled probability in eqn. (7.2).

In addition to the field χ, the updating of the gauge fields will require another quantity

$$\xi = \left(D^\dagger\right)^{-1} \chi = M^{-1}\phi. \tag{7.5}$$

This, unfortunately, is not so trivial to obtain, requiring a conjugate gradient inversion. Such a step is in common with the Langevin and microcanonical approaches.

I now come to the updating of the gauge field. What would be most desirable would be something like a Metropolis *et al.* procedure where the acceptance of trial changes is governed by changes in the action

$$S(A,\phi) = S_G + \phi^* \left(D^\dagger\right)^{-1} D^{-1}\phi. \tag{7.6}$$

However, this is impractical because every time A is changed, D changes and its inverse on ϕ would have to be recalculated. To avoid this slow procedure, consider making only small changes in A. The changes in the action are then related to the first derivative with respect to A

$$\begin{aligned}\frac{\partial S}{\partial A}|_\phi &= \frac{\partial S_G}{\partial A} - 2\ \mathrm{Re}\left(\phi^* \left(D^\dagger\right)^{-1} D^{-1}\frac{\partial D}{\partial A}D^{-1}\phi\right)\\ &= \frac{\partial S_G}{\partial A} - \xi^*\frac{\partial D}{\partial A}\chi - \chi^*\frac{\partial D^\dagger}{\partial A}\xi.\end{aligned} \tag{7.7}$$

Now consider the quantity

$$S_T(A,\chi,\xi) = S_G - \xi^* D\chi - \chi^* D^\dagger \xi. \tag{7.8}$$

Eqn. (7.7) implies

$$\frac{\partial S}{\partial A}|_\phi = \frac{\partial S_T}{\partial A}|_{\chi,\xi}. \tag{7.9}$$

If we consider small changes in A, first order changes of the action S at constant ϕ equal the changes in S_T calculated at constant χ and ξ. As only changes in the action enter into the Metropolis *et al.* algorithm, updating the A fields using S_T is equivalent to lowest ord৳: to using the exact action S. This is the proposal of Ref. 21, and is easily implemented because S_T is local.

As with the pseudofermion, Langevin, and microcanonical methods, this algorithm makes a small step size approximation. To have confidence that the errors induced by a finite step are small, one should study a desired measurable for a few values of this step size and extrapolate to the infinitesimal limit. Ref. 16 argued that for the Langevin algorithm, a finite step represents a simulation with an effective action which differs from the initial one by terms vanishing with the step size. If this new action has the same continuum limit, then even these finite step simulations should give the same numerical results for physical observables. Nevertheless, an extrapolation to vanishing step is still necessary to compare results of different algorithms with a given set of parameters at a finite lattice spacing.

The solid points in figure 7.1 show the average plaquette $P = \langle \frac{1}{3}\mathrm{Re\ Tr}\ U_p \rangle$ measured with this algorithm for the $SU(3)$ theory at $\beta = 4.5$. This is plotted versus the acceptance per hit, a simple measure of the step size. This simulation was done using the action from Ref. 18 with eight flavors and a fermion mass of 0.1 in lattice units. The zero step limit, which follows from extrapolation to 100% acceptance, represents the correct plaquette value with the inclusion of the dynamical fermions. The crosses in this figure were obtained in a standard pseudofermionic run.

The Metropolis *et al.* algorithm in the limit of small step size is quite close to the Langevin approach. Both cases involve small random changes in the field variables. A standard Metropolis *et al.* program first tries unbiased changes about the old field, and then, to maintain the desired peaking of the distribution towards lower action, rejects a fraction of those changes which go toward larger action. In contrast, the Langevin approach always accepts the changes, but makes them in a direction biased towards lower action. This bias is determined by the same first derivative of the action with respect to A used above to construct S_T.

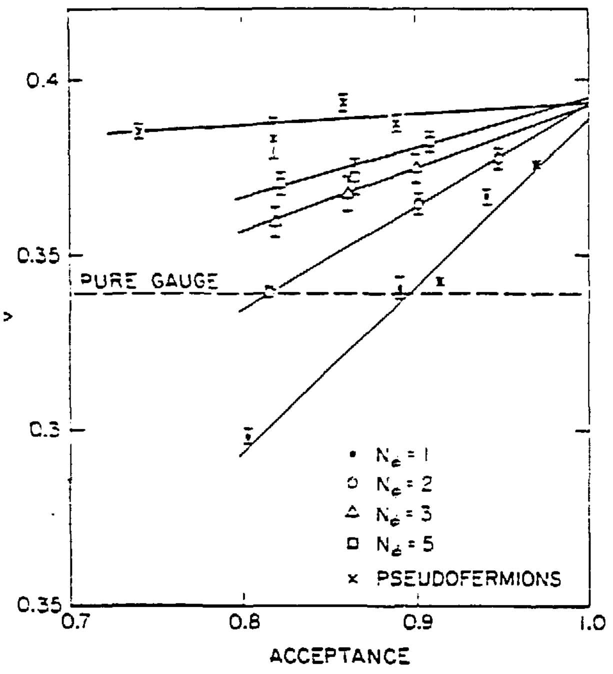

Fig. 7.1: The average plaquette as a function of the acceptance probability per Metropolis *et al.* hit. The parameter N_ϕ is discussed in the text. Note the interpolation between the simple algorithm of this section as shown by the solid points and the pseudofermionic simulation shown by the crosses.

The similarity of the approaches suggests that the finite step errors should be comparable. To directly make such a comparison, one should use a common definition of step size. One such measure would be to use the number of iterations needed to decorrelate lattices. I conjecture that the behavior of the solid points in figure 7.1 will mimic that of a Langevin simulation when plotted versus the decorrelation time.

This algorithm also has close connections with the pseudofermion method. To see this note that our field ξ has a probability distribution precisely the same as the pseudofermionic one in eqn. (4.5). Indeed, the present algorithm is equivalent to using but a single pseudofermionic field for the expectation value used in eqn. (4.4) to estimate M^{-1}. As mentioned in the last paragraph of section IV, the systematic errors from using a finite number of pseudofermionic configurations average out after the extrapolation to zero step size.

Clearly the present algorithm represents an extreme case. One could interpolate between this and the pseudofermionic algorithm by averaging over some fixed number N_ϕ of ξ fields. This may also be thought of as considering N_ϕ species of fermions, each with its own ξ field, but then letting each species contribute only $1/N_\phi$ in the updating of the A field. As N_ϕ increases, we approach the pseudofermion algorithm. The remaining points in figure 7.1 exhibit this interpolation.

The allowing of each species to contribute only fractionally to the updating of the A field may provide a scheme to reduce the effective number of fermion species overall. Naively, this can remove the extra doubling introduced in eqn. (7.1) as well as any inherent doubling in the basic formulation of the fermions. Such a possibility has been mentioned in the context of both Langevin and pseudofermionic algorithms. There may, however, be some danger in this procedure because chiral symmetry breaking and anomalies suggest nonanalytic behavior as the number of fermionic species varies.

VIII. NEW APPROACHES AND SPECULATIONS

All the commonly used fermionic algorithms, including pseudofermions, Langevin, microcanonical, and that of the previous section, involve an extrapolation in a step size parameter. This is unfortunate in that lattice gauge calculations already involve tenuous extrapolations to zero lattice spacing and infinite volume. On the other hand, as discussed in section V, all known exact algorithms require computer time growing as the system volume squared. In this section I return to such algorithms with the hope of minimizing the coefficient of this bad growth.

The difficulty with the approach of Ref. 13 is that a time consuming inversion must be done to test every trial change in the gauge field. The approximate schemes all work to reduce the frequency of such inversions to once per sweep. One could imagine making a trial changes of all lattice variables simultaneously, and then accepting or rejecting the entire new configuration using the exact action. The problem with this approach is that a global random change in the gauge fields will generally increase the action by an amount proportional to the lattice volume, and thus the final acceptance rate will fall exponentially with the volume. The acceptance rate could in principle be increased by decreasing the step size of the trial changes, but then the step size would have to decrease with the volume. Exploration of a reasonable region of phase space would thus require a number of steps growing as the lattice volume. The net result is again an exact algorithm which requires computer time growing as volume squared.

So far this discussion has assumed that the trial changes are made in a random manner. If, however, one can properly direct these variations, it might be possible to reduce the coefficient of the volume squared behavior. An algorithm of this type was proposed in Ref. 22. For example, one could do either a Langevin or Metropolis *et al.* sweep using the action S_T of the last section. By keeping track of all the probabilities for accepting changes along the way, one could in principle calculate the inverse probability for taking the new lattice back to the original in a similar sweep. Then one can construct the analog of eqn. (2.1) for a generalized acceptance for the entire lattice which will exactly restore detailed balance. If the changes in S_T are a good approximation to the changes in the true action, the factors in the acceptance criterion should tend to cancel, giving a reasonably large final acceptance rate. Attempts to use this approach in Ref. 23 were moderately successful, although those authors felt that standard hybrid Langevin techniques were superior.

There are some interesting variations on this approach. Because the final acceptance makes the algorithm exact, there is no restriction on the initial scheme used to find the trial changes. Recent results [17] suggest that a large part of the effect of fermionic loops is simply a renormalization of the gauge coupling. For heavy quarks this may be understood in terms of the hopping parameter expansion presented in Ref. 24. This suggests obtaining a trial configuration by a Monte Carlo sweep of either the entire lattice or some large fraction using only the pure gauge action with a different effective value of the coupling. A final acceptance criterion for this new set of fields can then be used to restore detailed balance with the full action including fermionic effects.

More precisely, suppose that a trial configuration C' is obtained in a manner satisfying detailed balance with an effective action S_E

$$P\left(C',C\right)e^{-S_E(C)} = P\left(C,C'\right)e^{-S_E(C')}. \qquad (8.1)$$

Detailed balance with respect to the full action S is restored if we now accept this trial configuration with probability

$$P_A = \min\left[1\,, \frac{\exp\left(-S_E\left(U\right)\right)}{\exp\left(-S_E\left(U'\right)\right)} \times \frac{\exp\left(-S\left(U'\right)\right)}{\exp\left(-S\left(U\right)\right)}\right]. \qquad (8.2)$$

The essence of this approach is to use for S_E an easily calculated quantity depending on a small number of coupling parameters. The latter should to be adjusted in such a manner to maximize the final acceptance. In this way it may be possible to update many links simultaniously for a single calculation of the full action $S\left(U\right)$, which in general will involve a slow inversion step. The procedure should work particularly well when the quarks are heavy so that the first terms in the hopping parameter expansion adequately describe the effects of fermions. It remains to be demonstrated how effective such an approach will be when the quarks become light and the effective action requires terms of increasing non-locality.

ACKNOWLEDGEMENTS

This work supported under Contract DE-AC02-76CH00016 with the U. S. Department of Energy. The simulations presented in section II were run on the CYBER 205 at Florida State University using time granted under the Energy Sciences Advanced Computation program of the DOE.

REFERENCES

1. M. Creutz, *Quarks, Gluons and Lattices*, (Cambridge Univ. Press, 1985).
2. An elementary discussion of invariant group integration is contained in chapter 8 of Ref. 1.
3. E. Seiler, private communication.
4. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, J. Chem. Phys.21, 1087 (1953).

5. Yang, C.-P., *Proc. of Symposia in Applied Mathematics*, Vol. XV, 351 (Amer. Math. Soc., Providence, R.I., 1963); M. Creutz, Phys. Rev. D21, 2308 (1980).
6. M. Creutz, preprint BNL-39445 (1987); F. R. Brown and T. J. Woch, Columbia Univ. preprint CU-TP-365 (1987).
7. S. L. Adler, Phys. Rev. D23, 2901 (1981); C. Whitmer, Phys. Rev. D29, 306 (1984); Princeton University PhD thesis (1984).
8. D. Callaway and A. Rahman, Phys. Rev. D28, 1506 (1983).
9. M. Creutz, Phys. Rev. Lett. 50, 1411 (1983).
10. R. Swendsen, Phys. Rev. Lett. 52, 1165 (1984); D. Callaway and R. Petronzio, Phys. Lett. 139B, 189 (1984).
11. P. T. Matthews and A. Salam, Nuovo Cim. 12, 563 (1954).
12. F. Fucito, E. Marinari, G. Parisi and C. Rebbi, Nucl. Phys. B180[FS2], 369 (1981).
13. D. Weingarten and D. Petcher, Phys. Lett. 99B, 333 (1981).
14. G. Bhanot, U.M. Heller and I.O. Stamatescu, Phys. Lett. 129B, 440 (1983).
15. M. Grady, Phys. Rev. D32, 1496 (1985).
16. G.G. Batrouni, G.R. Katz, A.S. Kronfeld, G.P. Lepage, B. Svetitsky and K.G. Wilson, Phys. Rev. D32, 2736 (1985).
17. A. Ukawa and M. Fukugita, Phys. Rev. Lett. 55, 1854 (1985).
18. J. Polonyi and H.W. Wyld, Phys. Rev. Lett. 51, 2257 (1983); J. Kogut, J. Polonyi, H.W. Wyld and D.K. Sinclair, Phys. Rev. Lett. 54, 1475 (1985).
19. B.J. West and K. Lindenberg, to be published (1987).
20. S. Duane and J. Kogut, Phys. Rev. Lett. 55, 2774 (1985); Nucl. Phys. B275, 398 (1986).
21. M. Creutz and R. Gavai, Nucl. Phys. B280, 181 (1987).
22. R. T. Scalettar, D.J. Scalapino and R.L. Sugar, Phys. Rev. B34, 7911 (1986).
23. S. Gottlieb, W. Liu, D. Toussaint and R.L. Sugar, Phys. Rev. D35, 2611 (1986).
24. A. Hasenfratz and P. Hasenfratz, Phys. Lett. 104B, 489 (1981).