# THE MONTE CARLO METHOD

## I. INTRODUCTION

The Monte Carlo method is often referred to as a 'computer experiment'. One might think of this as a way of conveying the fact that the output of simulations is not an equation, as in conventional theory. Instead, numbers appear on the computer screen in somewhat the same way that numbers appear on a measuring device in the laboratory. Thus there is the implication that somehow simulations are a bit of a 'black box' and that the use of the computer is hiding the underlying physics. The purpose of this note is partly to emphasize some of the mathematical rigor behind Monte Carlo: It is not a happy accident that the computer is generating configurations with the desired probability distribution! Indeed, the fundamental equations underlying simulations are the same as analytic theories, and one can view simulations as a way of solving the mathematics (differential equations) when it becomes too complicated for analytic techniques.

With all that said, it is still useful to pursue the 'Monte Carlo as experiment' point of view. Consider the process of making a measurement in the laboratory. Nature prepares a 'configuration' $i$ of the system, and the experimentalist takes that configuration and records a value for some quantity of interest. To get better statistics (or perhaps inevitably because of finite measuring time) nature actually produces many configurations, and the experimentalist averages over the values obtained. It is useful to emphasize that no matter how long the experimentalist measures, the configurations she sees are an *incredibly* small subset of those that the system is capable of exploring.

Nature uses some very complex rules for time evolving the system from configuration to configuration, for example the many particle Newton or Schroedinger equations. These rules govern the states that the experimentalist sees, and hence the data she takes.

Here's one useful way to think about computer simulations: The goal of a computer simulation is to devise a method where the *computer* plays a similar role to that of *nature* for the experimentalist. That is, the computer generates configurations upon which we make measurements. This leaves us with the problem of devising instructions for the computer that replicate nature's way of generating configurations.

One approach to constructing a simulation would be actually coding up the microscopic equations governing the system's time evolution. Simulations of classical systems going under the name 'molecular dynamics' are actually done precisely this way. One computes the force $F_n$ on each particle $n$, uses the force to compute the acceleration $a_n = F_n/m$, and then moves the velocity and position forward a small time interval $dt$ with, $v_n \to v_n + a_n\,dt$; $x_n \to x_n + v_n\,dt$.

But in the spirit of statistical mechanics, we really don't care about the microscopic time evolution and the paths $x_n(t)$ and $v_n(t)$ the particles take in phase space. All we really need is to replicate the *probability* $P(\{x_n, v_n\})$ that nature uses to generate her configurations. If we can do that, we'll get the same answers as the experimentalist!

If we were doing classical statistical mechanics, the probability distribution that we would be attempting to emulate would be the Boltzmann distribution $P(\{x_n, v_n\}) = Z^{-1}e^{-\beta E(\{x_n,v_n\})}$. However, let's discuss Monte Carlo within the context of a general probability distribution. This will emphasize that Monte Carlo is by no means limited to Boltzmann statistics. To make the notation less unwieldy, we will also label our probabilities by a single index $i$ which will be understood to represent particular values of all the degrees of freedom of the system we are studying (for example $i$ could mean a collection of positions and velocities $\{x_n, v_n\}$). In the remainder of this note I will denote the inverse temperature by $\beta = 1/T$, and set Boltzmann's constant to unity.

As we shall see, to do Monte Carlo, we actually don't need to know $p_i$, but only ratios of $p_j/p_i$ for two configurations. This is certainly an important point for statistical mechanics since $p_j/p_i = e^{-\beta(E_j-E_i)}$ is known, but the individual $p_i$ involve the unknown partition function $Z$.

## II. TRANSITION PROBABILITIES AND DETAILED BALANCE

Our goal is to figure out a rule $T_{ji}$ to evolve the configuration $i$ into the configuration $j$ which will generate configurations with a desired probability. More precisely, our process will not be deterministic, but will involve random changes, so $T_{ji}$ will be the probability to generate $j$ from $i$.

Because probabilities are non-negative, and sum up to unity, the rule $T_{ji}$ satisfies,

$$\begin{aligned} T_{ji} &\geq 0 \\ \sum_j T_{ji} &= 1. \end{aligned} \tag{1}$$

A matrix obeying these two restrictions is called a 'stochastic matrix'. Its eigenvalues obey $|\lambda| \leq 1$. Also, there is an eigenvector with eigenvalue $\lambda = 1$. These facts are simple to prove.

Consider the eigenvector equation,

$$\sum_i T_{ji} v_i = \lambda v_j. \tag{2}$$

Take absolute values of both sides of the equation and use the fact that $T_{ji} \geq 0$ and the triangle inequality.

$$\sum_i T_{ji} \, |v_i| \;\geq\; |\lambda| \, |v_j|. \tag{3}$$

Now sum both sides of the equation over $j$ and use the fact that $\sum_j T_{ji} = 1$,

$$\sum_i |v_i| \;\geq\; |\lambda| \sum_j |v_j|. \tag{4}$$

We now see that $|\lambda| \leq 1$.

To show that a stochastic matrix has an eigenvalue of unity, consider the vector with all components the same, $v_j = 1$. Then, from the second line of Eq. 1,

$$\sum_j v_j T_{ji} = v_i. \tag{5}$$

So the constant vector is a left eigenvector of $T$ of eigenvalue 1. Remember that the eigenvalues of a matrix are the same whether one considers left or right eigenvectors, but the eigenvectors themselves can be quite different unless the matrix is symmetric. This is important to keep in mind, because we will be showing that the right eigenvector of $T$ is not the trivial constant vector. In fact, the right eigenvector of $T$ is $p_j$.

That the eigenvalue $\lambda = 1$ is non-degenerate is a consequence of the Perron-Frobenius theorem.

Because $T$ is our rule for generating one configuration from the next, we can view the generation of a long sequence of configurations in the Monte Carlo process[1] as the repeated application of the matrix $T$. We know that when we repeatedly apply a matrix to a vector, we project out the eigenvector of largest eigenvalue. As we have seen above, because $T$ is stochastic, we will generate the eignevector of eigenvalue 1.

So we can now make our goal a little bit more precise: We want to figure out $T$ obeying Eq. 1 whose eigenvector of eigenvalue one is the vector of desired probabilities $p_i$. Then, repeatedly applying $T$ will give us $p$ and we're done.

It is easy to show that if we construct $T$ to obey 'detailed balance',

$$T_{ji}\, p_i = T_{ij}\, p_j \tag{6}$$

then $p_i$ is an eigenvector of $T$ of eigenvalue 1:

$$\sum_i T_{ji}\, p_i = \sum_i T_{ij}\, p_j = p_j \sum_i T_{ij} = p_j. \tag{7}$$

It is also easy to formulate $T$ to obey detailed balance. The most famous prescription is due to Metropolis. The Metropolis algorithm says that one suggests a move from $i$ to $j$ and then accepts the move with probability one if $j$ is more likely than $i$, that is, if $p_j/p_i > 1$. If $j$ is less likely than $i$, accept $j$ with probability $p_j/p_i$.

Exercise 1: Show that if you generate a random number $r$ uniformly on $[0, 1]$ and accept the move when $r < p_j/p_i$, then the Metropolis algorithm is correctly implemented.

## III. SUMMARY OF WHY MONTE CARLO WORKS

The argument presented above can be summarized as follows:

```
[a] Metropolis  ---> Detailed Balance ---> p is eigenvector of T of eigenvalue 1

[b] T is stochastic ---> max eigenvalue of T is 1

[a+b]  p is eigenvector of T of largest eigenvalue

[c]  repeated application of matrix results in eigenvector of largest eigenvalue

[a+b+c]  repeated application of T gives p, as desired.
```

## IV. SUMMARY OF MONTE CARLO PROCEDURE

(i.) Take an initial configuration $i$ and propose a change to a new configuration $j$.
(ii) Accept the new configuration with probability $\min(1, p_j/p_i)$.
(iii) Measure quantities of interest.
(iv) Repeat the process many times.

There are of course many subtleties: What sort of procedure is followed to "suggest changes"? How many configurations do you need to generate and do you, as in some experiments, need to allow the system time to equilibrate? We will address some of these issues below.

## V. ANALOGIES BETWEEN MONTE CARLO AND EXPERIMENTS

There are a number of useful analogies between Monte Carlo and experiment. Perhaps most importantly, when first encountering Monte Carlo it is natural to wonder how the relatively small sample of configurations that a computer generates could possibly replicate the behavior of physical systems which have incredibly huge numbers of states. A partial answer is provided by noting that the exact same question could be posed to any experimentalist, where, similarly, measurements are based on a very small subset of states which are generated by the time evolution during the course of the experiment.

Second, when one collects data in a Monte Carlo, one measures observables for each configuration generated and then does a simple average over all the configurations. There are no probabilities $p_i$ put in the averaging. The reason, of course, is that the computer is putting the $p_i$ in for you, by generating the configurations with their correct $p_i$. This is just as in an experiment. No experimentalist weights her measurements with the Boltzmann factor!

In addition to averages, error bars are generated in Monte Carlo just as in experiments. (See Sec. XI.)

One often has to wait a little while after starting a simulation to allow the system to equilibrate, much as the experimentalist will not start recording data until some time has passed following perturbing the system in some way.

## VI. A SUBTLETY CONCERNING $T$ AND DETAILED BALANCE

If you think about it, the transition matrix $T$ actually has two pieces in its construction. The first is the suggestion of a new state $j$, and the second is the decision (e.g. Metropolis) whether to accept $j$ or not. It is the *combination* of these two factors which determines $T$ and which must obey detailed balance. Examples of how the method of suggesting states might go awry are given in Exercises 7 and 12.

## VII. SOME EXERCISES

### A. The Heat Bath Algorithm

The Metropolis algorithm is not the only way to go.

Exercise 2: Show that the "heat bath" prescription

$$T_{ji} = p_j/(p_i + p_j) \tag{8}$$

obeys detailed balance.

Thus the starting point 'Metropolis' of [a] in Sec. III can be replaced by 'Heat Bath'.

### B. When Do You Measure?

Exercise 3: Consider a system with two states and associated probabilities $p_1, p_2$. Suppose the observable $A$ has the values $A_1$ and $A_2$ in the two states. What will you get for $\langle A \rangle$ if you measure $A$ only when a move is accepted? What is the correct value for $\langle A \rangle$? When should you measure $A$ to get the right answer? Argue qualitatively that, if $p_1 >> p_2$, the sequence of configurations generated gives a sensible answer when you put your measurement in the correct spot.

### C. Explicitly Constructing the matrix $T$

Exercise 4: Consider a system with three states and associated probabilities $p_1, p_2, p_3$. Construct the 3x3 matrix for $T$ assuming that if you are in a state $i$ you suggest a change to one

of the other two states with probability $\frac{1}{2}$ each, and then accept or reject with Metropolis. Verify that your $T$ obeys Eq. 1 and that the vector $(p_1\ p_2\ p_3)$ is a right eigenvector of $T$ of eigenvalue one. Verify the other two eigenvalues are less than one in absolute value.

Exercises 5 and 6 below emphasize that the rule $T$ for generating states is not unique. The art of Monte Carlo is in finding the $T$ which works best, that is, which moves you around in phase space most rapidly and generates the smallest error bars.

Exercise 5: Do Exercise 4 again, but use the heat bath algorithm.

Exercise 6: Again consider the same three state system as Exercise 4. Construct the 3x3 matrix for $T$ assuming that if you are in a state $i$ you suggest a new state (which could be the same as the one you are in) randomly, that is, with pick the new state to be 1 or 2 or 3, with probabilty $\frac{1}{3}$, and then do Metropolis. Verify that your $T$ obeys Eq. 1 and that the vector $(p_1\ p_2\ p_3)$ is a right eigenvector of $T$ of eigenvalue one. Verify the other two eigenvalues are less than one in absolute value. In which case is the next smallest eigenvalues closer to $|\lambda| = 1$, here, or in Ex. 5?

Exercise 7: Consider a system with an infinite number of states $E_i = \omega\, i$ for $i = 0, 1, 2, 3, \ldots$, and associated probabilities $p_i \propto e^{-\beta E_i}$. Construct the matrix for $T$ assuming that your suggestion of a move from state $i$ is to one of the two immediately adjacent states $j = i \pm 1$, with probability $\frac{1}{2}$. Verify $(p_1\ p_2\ p_3\ \ldots)$ is a right eigenvector of $T$. Be especially careful with the state $i = 0$ which has no state beneath it! Should you suggest state $j = 1$ all the time, since there is no $j = -1$? This is an example of the subtlety mentioned in Sec. VI.

### D. Very Simple Monte Carlo Simulations

Exercise 8: Write a Monte Carlo code which computes $\langle E \rangle$ for the three state system of Exercise 4, using the $T$ described in that exercise.

Exercise 9: Write a Monte Carlo code corresponding to the $T$ of exercise 5.

Exercise 10: Write a Monte Carlo code corresponding to the $T$ of exercise 6.

Exercise 11: Write a Monte Carlo simulation for an energy which has a continuous degree of freedom $x$ with $E(x) = \frac{1}{2}kx^2$. Construct $T$ by suggesting the new state $x' = x + \Delta(r - \frac{1}{2})$

where $\Delta$ is the 'step-size', and $r$ is a random number uniformly distributed on (0,1). Use Metropolis to accept/reject $x'$. What is the correct value for $\langle E \rangle$? Show your code gets it, for any value of $\Delta$. What values of $\Delta$ are best? See also Sec. XI below.

Exercise 12: Do Exercise 11 again, except this time construct $T$ by suggesting the new state $x' = x + \Delta(r - \frac{1}{3})$, and then use Metropolis. Show you code gets the wrong value for $\langle E \rangle$. Show that your $T$ violates detailed balance, even though you use Metropolis after the move suggestion.

### E. Less Simple Monte Carlo Simulations

Many readers going through these notes have a particular (and presumably non-trivial) problem they wish to address with Monte Carlo. If, however, you do not, a standard Monte Carlo simulation to try is the two dimensional Ising model. Many books on simulations contain descriptions.[2–4]

### F. Testing Monte Carlo Simulations

As with all codes, tests are essential. Here I will mention three very useful ones.

First, Monte Carlo simulation codes can often be tested on small lattices. That is, it is often possible to enumerate all the possible states of a small system (e.g. a 4x4 Ising lattice has $2^{16}$ configurations) and have the computer then generate the partition function and expectation values through an explicit enumeration of all the configurations.

Second, limits of the Hamiltonian in question are often soluble. At high and low temperatures, one often can figure out the values of the energy and other observables. Likewise, by turning off one or more terms in the Hamiltonian, it may become analytically soluble. Or, if you write your code so that the interactions are different in different directions, you can sometimes turn off one or more directions and reduce the dimensionality of the system to a point where analytic solutions are known.

Finally, while you will usually write your code so that it averages observable which should be equivalent by some symmetry, looking at the unaveraged values first is often useful: Many bugs in codes are fingered by measurements which should be equal by symmetry but are not.

My experience is that if your code checks out for the entire model on small lattices, and then individual pieces on different lattice sizes, and also the high and low $T$ limits, then it is very likely correct.

## VIII. QUANTUM MONTE CARLO

The above discussion seems to rely very heavily on the fact that the system is classical. However, it turns out to be relatively easy to generalize classical Monte Carlo to Quantum Monte Carlo. All I will say here is that one takes the *operator* $e^{-\beta\hat{H}}$ and writes its trace as a path integral. This sum over paths involves a *classical* action, which then can be attacked with classical Monte Carlo. A nice discussion of this for the quantum harmonic oscillator is given by Creutz.[5] Another particularly interesting example is the mapping of the $d = 1$ Ising model in a transverse magnetic field onto the classical $d = 2$ Ising model.[6]

## IX. RELATION TO MOLECULAR DYNAMICS

At the beginning of these notes, I mentioned that Molecular Dynamics (MD) as one way we might get a computer to emulate nature. Since the equations of MD keep the energy constant (to within time step errors), one way to view MD is as a Monte Carlo where the proposed move does not change the energy, and is therefore accepted with unit probability according to the Metropolis prescription. Thus MD and Monte Carlo should give the same result, under the condition that the MD energy is initialized at the same value $\langle E\rangle$ as the average energy which comes out of the Monte Carlo. This is the usual equivalence of the canonical and microcanonical ensembles in statistical mechanics.

In practice, one might want to do MD at a particular temperature $T$ instead of at fixed $E$. One way to accomplish this is to evolve the positions and velocities $x_n$ and $v_n$ using the MD equations, but then periodically 'refresh' the velocities by replacing their current values with new ones drawn from the Gaussian distribution $e^{-m_n v_n^2/2T}$. Here $m_n$ is the mass of the $n$th particle. We choose to 'refresh' the momenta, because usually the energy takes the form $E = \frac{1}{2}\sum_n m_n v_n^2 + V(\{x_n\})$. That is, the potential energy doesn't depend on the velocities. So the probability distribution for the velocities is Gaussian, and we know how to generate Gaussian distributed random numbers. The positions usually invovle a more complicated $V$ than Gaussian.

This sort of finite $T$ molecular dynamics can be understood as a Monte Carlo with two types of moves. As remarked above, the conventional MD moves keep the energy constant and hence are always accepted. The momentum refreshment moves change the energy and hence are sometimes rejected according to the Metropolis prescription.

Exercise 13: Write a Molecular Dynamics code for $E = \frac{1}{2}kx^2 + \frac{1}{2}mv^2$. (That is, use $F = -kx$ and $a = F/m$ to update $v$.) Verify that you travel along an ellipse in phase space at constant energy, when $dt$ is small. Here '$dt$ small' really means $dt$ is much less than the period $2\pi\sqrt{m/k}$. Compute $\langle x^2\rangle$ and $\langle p^2\rangle$ and show you get the correct answers (which only depend on $x_0$ and $v_0$). Note that the Molecular Dynamics integration of equation of motion introduces time step errors. In the Euler method, the accumulation of these errors will lead to an exponential increase in $E$. The "leap-frog" method is much more robust.

Exercise 14: Include steps which refresh the momentum. Compute $\langle x^2 \rangle$ and $\langle p^2 \rangle$ and show you get the correct answers (which now do not depend on the initial conditions as in Ex. 13 but only on temperature $T$.)

## X. RELATION TO THE LANGEVIN EQUATION

Monte Carlo can also be related to the Langevin Equation. Consider a system with an energy $E(x)$ which depends on the degree of freedom $x$. (For simplicity, let's just consider a single degree of freedom.) Suppose we define a procedure for updating $x$ via

$$x' = x - \epsilon \frac{\partial E}{\partial x} + \sqrt{4\epsilon T}\, r \tag{9}$$

Here $r$ is a Gaussian random number, $p(r) = e^{-r^2}/\sqrt{\pi}$, and $T$ is the temperature. We will now show that this equation satisfies detailed balance with $p(x) \propto e^{-\beta E(x)}$.

The Langevin Equation amounts to a prescription $T$ for getting a new state $x'$ from an original state $x$. Given $x'$ and $x$, the value of $T$ is given by the probability of throwing the appropriate random number which will take you from $x$ to $x'$.

$$T(x', x) = \frac{1}{\sqrt{\pi}} e^{-r^2} \tag{10}$$

$$r = \frac{1}{\sqrt{4\epsilon T}} (x' - x + \epsilon \frac{\partial E}{\partial x}) \tag{11}$$

Therefore, introducing the notation $dx = x' - x$,

$$T(x', x) = \frac{1}{\sqrt{\pi}} e^{-(dx + \epsilon \frac{\partial E}{\partial x})^2/(4\epsilon T)} \tag{12}$$

Likewise,

$$T(x, x') = \frac{1}{\sqrt{\pi}} e^{-(-dx + \epsilon \frac{\partial E}{\partial x})^2/(4\epsilon T)} \tag{13}$$

Note that actually the gradient of the energy n Eq. 13 should now be evaluated at $x'$, but because $\epsilon$ is small, $dx$ will be small and $x'$ close to $x$. Since the gradient is already multiplied by $\epsilon$ the difference is higher order and we drop it.

Putting this together,

$$\frac{T(x', x)}{T(x, x')} = e^{-\frac{\partial E}{\partial x} dx/T} = e^{-dE/T} = e^{-(E(x') - E(x))/T} \tag{14}$$

$$T(x, x') e^{-E(x')/T} = T(x', x) e^{-E(x)/T}, \tag{15}$$

where again we have dropped terms which are higher than linear order in $\epsilon$. Eq. 15 completes the demonstration that the Langevin equation obeys detailed balance. Hence by our general theorems concerning Monte Carlo, it will generate configurations of the system according to the Boltzmann distribution.

A final comment: You will notice that the prefactors of the random force and the force which depends on $E$ in the Langevin equation are not independent. This is an example of the fluctuation-dissipation theorem.

Exercise 15: In proving detailed balance we threw away higher order terms in $\epsilon$. Consider the energy $E(x) = \frac{1}{2}kx^2$ and go through the Langevin analysis retaining terms to all orders in $\epsilon$. Show that detailed balance is obeyed with a $k$ shifted by order $\epsilon$. Will $\langle x^2 \rangle$ be overestimated or underestimated?

Exercise 16: Write a computer code to implement the Langevin equation for $E(x) = \frac{1}{2}kx^2$. Verify your results from Exercise 15.

## XI. ERROR ANALYSIS IN MONTE CARLO

This final section is intended to review error analysis in Monte Carlo, and illustrate it with the very simple example of a simulation of $E(x) = \frac{1}{2}kx^2$.

### A. Introduction and Definitions

We begin with the definitions,

$$\langle x \rangle = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{16}$$

$$\langle x^2 \rangle = \frac{1}{N} \sum_{i=1}^{N} x_i^2 \tag{17}$$

$$\sigma = \sqrt{\frac{\langle x^2 \rangle - \langle x \rangle^2}{N-1}}. \tag{18}$$

Here $x_i$ is the $i$th measured value of $x$, and $N$ is the number of measurements. The definitions of $\langle x \rangle$ and $\langle x^2 \rangle$ are unambiguous. The entire content of this note is to clarify the proper denominator of the definition of $\sigma$. Specifically, the formula for $\sigma$ assumes that the $N$ measurements are all independent. Since successive values of $x$ are generated from each other, this is never true. The $x$ values are more and more related the less time one waits between measurements.

To quantify the correlations among successive $x$ values, we define the autocorrelation function,

$$c(l) = \frac{1}{N-l} \sum_{i=1}^{N-l} y_i y_{i+l} \tag{19}$$

Here $y(i) = x(i) - \langle x \rangle$ measures the fluctuation of $x$ about its average value. $c(l)$ measures whether those fluctuations are related for $x$ values $l$ measurements apart. Saying $x_i$ and $x_{i+l}$ are independent means whether $x_{i+l}$ is above or below $\langle x \rangle$ (the sign of $y_{i+l}$) is unrelated to

the sign of $y_i$. If that is the case, $c(l) = 0$ (to within errors). Clearly $c(0)$ is never zero. In fact, $c(0) = \sigma^2$. It is conventional to redefine $c(0) \to c(0)/\sigma^2$ so that $c(0) = 1$.

### B. Time Histories

Let's begin by looking at actual time histories of $x$. I chose $k = 1$ and $T = 2$ so that $\langle x^2 \rangle = 2$. The step size for suggested changes in $x$ is $\Delta$. I measure every Monte Carlo step and ran for $N = 400000$ sweeps. Here of course since $\langle x \rangle = 0$, $y_i = x_i$. The Monte Carlo time histories are given in Figure 1. That the data are correlated is immediately evident. If a value of $x_i$ is positive, its successors tend to be positive and similarly if it is negative. The dependence on the step size $\Delta$ is easy to interpret.
If $\Delta$ is small you do not suggest much of a change, and successive values of $x$ are highly correlated. Likewise, if $\Delta$ is large, most suggested Monte Carlo moves take you out of the part of phase space of low energy and are rejected. (This results in the long flat regions of the evolution of $x$.)

### C. Correlation Functions

The plots of $c(l)$ resulting from the same data are given in Figure 2. We see that $c(l)$ has a characteristic decaying exponential form. We define the correlation time $\tau$ to be the point when $c(l = \tau) = e^{-1}$ and say that measurements of $x$ are independent when $l$ exceeds $\tau$. (Strictly speaking, we want $c$ to go to zero, but $c(\tau) = e^{-1}$ is an acceptable criterion.) Notice you can almost guess the values of $\tau$ given by Figure 2 directly from the time histories of Figure 1.

As mentioned earlier, in generating the above results I measured $x$ every Monte Carlo step. What happens if one instead measures only every $m$th step? Define $c_m(l)$ to be the correlation function when measurements are only every $m$th Monte Carlo step. It is easy to convince yourself that $c_m(l) = c_1(ml)$, so the correlation function rescales in a trivially fashion. The point is that if one choose $m > \tau$, then the measurements all become independent.

• *So one way to ensure the value for the error bar $\sigma$ is correct is to make sure measurements are separated by a waiting time $m > \tau$.*

This approach has the advantage that one does not waste time making measurements when the measurements are not independent.

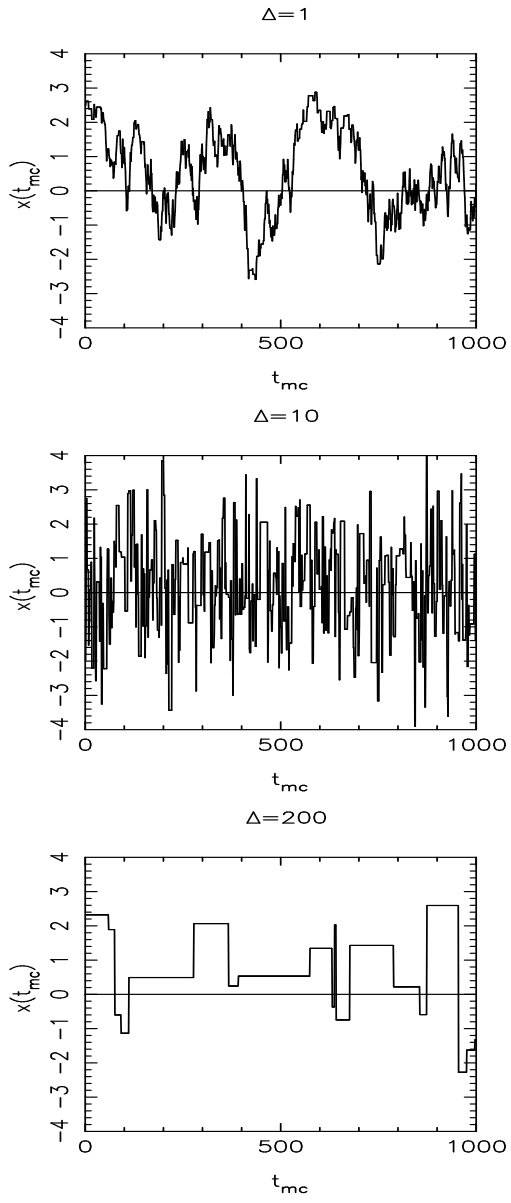


FIG. 1: First 1000 steps in Monte Carlo time history of $x$ for three different step sizes $\Delta = 1, 10, 200$. (Acceptance rates=0.70, 0.35, 0.02).

### D. Rebinning Data

An alternate (and equivalent) approach to getting the correct $\sigma$ is by "rebinning" the data. Take a file containing the complete time history of a measurement, for example the data for $x$ which is partially shown in Figure 1. Choose a "bin size" $M$, and average the data for $x$ over each of the $L = N/M$ bins (remember $N$ = total number of measurements)

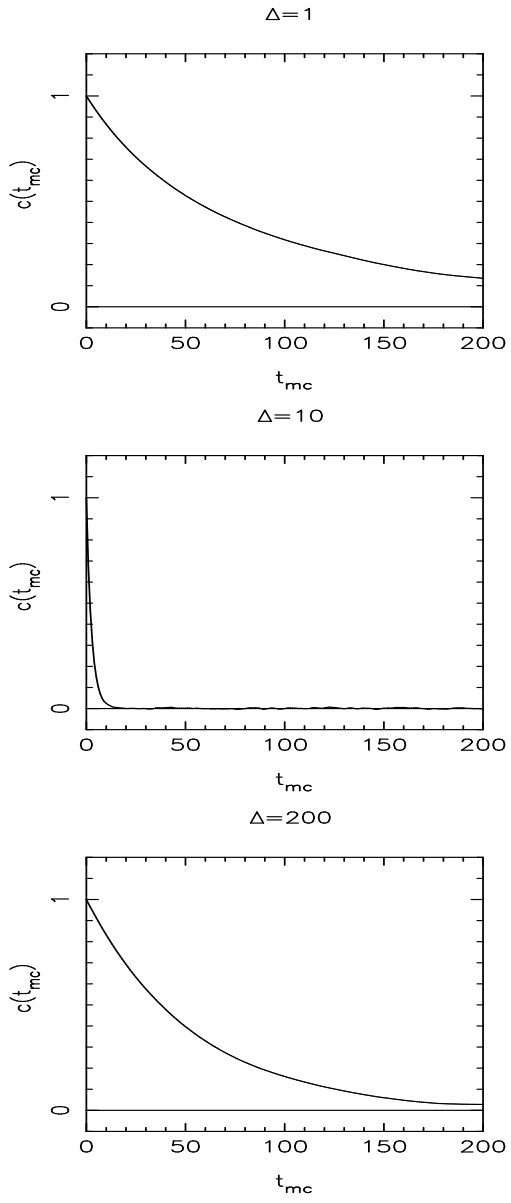


FIG. 2: Autocorrelation functions for the complete data sets (400,000 steps) as in figure 1. (Step sizes $\Delta = 1, 10, 200$, Acceptance rates=0.70, 0.35, 0.02).

to create $L$ "binned measurements" $m_j$.

$$m_j = \frac{1}{M} \sum_{i=M*(j-1)+1}^{M*j} x_i. \tag{20}$$

Treat these $L$ values for $m$ as your independent measurements. As seen in Equation 5, the values for $m$ are already averages over $M$ values of $x$. Define averages and error bars as in Equation 1, with $L$ replacing $N$ in the normalizations $1/N$ and $1/\sqrt{N-1}$. The average $\langle x \rangle$ is independent of $M$ since all one is doing is reordering a linear sum. The average $\langle x^2 \rangle$ is however altered, and hence so is the error bar $\sigma$. Figure 3 shows values for $\sigma$ as a function

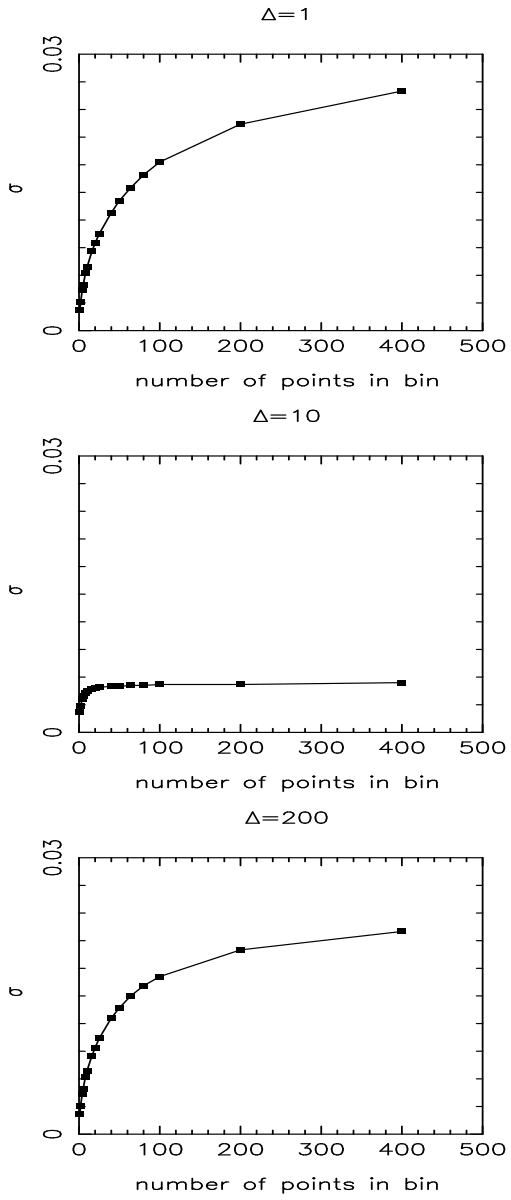


FIG. 3: Error bars for different bin sizes $M$. Data is that of Figures 1,2: step sizes $\Delta = 1, 10, 200$. (Acceptance rates=0.70, 0.35, 0.02).

of the number of $x$ values in each bin, $M$.

What is the interpretation of Figure 3? Consider $M = 1$. In this case only one value of $x$ is put in each bin, so that in calculating $\sigma$ it is assumed all $x$ are independent. The error bar $\sigma$ thus obtained is too small. As $M$ becomes larger, more $x$ values are put in each bin, the number of bins (independent measurements) decreases, and $\sigma$ increases. Eventually $\sigma$ goes to an asymptotic value which gives the correct error bar.

Why does $\sigma$ not increase indefinitely as $M$ increases? You might expect it to, since the denominator $\sqrt{L-1}$ is getting smaller and smaller. The answer is that as more measurements are put in each bin, the different bins fluctuate less and less. The numerator which

measures these fluctuations decreases in exact compensation to the denominator. (However, to reiterate, initially for $M$ small when you put more measurements in the bins the new values are not independent and so the numerator does *not* decrease.)

• *So a second way to ensure the value for the error bar $\sigma$ is correct is to consider different binnings of the data, and use the value obtained asymptotically as each bin contains a lot of data.*

How do we see this result is consistent with the correlation function analysis? There are two checks. The first is to see that the value for $M$ at which $\sigma$ starts to flatten out should be roughly the same as the value of $\tau$ for which $c(l)$ gets small. Second, one can compare the values of $\sigma_1$ at $M = 1$, where one assumes all the $x$ are independent, with the asymptotic value $\sigma_\infty$. The claim is that these should be related by $\sigma_\infty = \sqrt{\tau}\sigma_1$. You can see this is roughly true: For $\Delta = 1$ we get $\sigma_1 = 0.0022$ and $\sigma_\infty = 0.030$ from Figure 3. If you assume all the measurements are independent, you underestimate $\sigma$ by more than an order of magnitude. Meanwhile, from figure 2, $\tau \approx 85$, and hence $\sqrt{\tau}$ similarly reflects this order of magnitude correction factor.

### E. Acceptance Rate

The acceptance rate provides a rough guide to the choice of a good step size. If the acceptance rate is too much greater than 0.5, then one is likely in the limit of Figures (1–3)a where the correlation time is unnecessarily large due to small suggested moves. Likewise, if the acceptance rate is too much less than 0.50, then one is likely in the limit of Figures (1–3)c where the correlation time is unnecessarily large due to multiple rejections.

### F. A "Cheap" Approach

I recommended a "cheapo" approach to error bars, which was to bin the data from your run into 10 bins. ($M = N/10$.) This strategy assumes that you were doing a reasonably long run, so that $N/10 > \tau$. My thinking is that if this is violated, you are in more serious trouble than just getting the wrong error bars: you will have less than 10 independent measurements (and perhaps have not even properly equilibrated) so it is likely your expectation values themselves are wrong. For the particular example we are doing, the reported values from my simplistic approach for $\langle x^2 \rangle$ and $\sigma$ were $1.973 \pm 0.051, 1.993 \pm 0.010, 1.995 \pm 0.039$ for $\Delta = 1, 10, 200$ respectively. These error bars should be the same as the asymptotic values of $\sigma$ in Figure 3.

### G. Final Comments

Does it matter which measurement you look at? I have looked entirely at $x_i$ in doing this analysis. Would it matter if I had examined $x_i^2$ or some other measurement? For this simple problem I don't think so. In more complicated simulations it may be important to calculate correlation times separately for measurements of "local" quantities (observables for degrees of freedom that are close together spatially) and "global" quantities (observables for degrees of freedom that are far apart spatially). The spatial separation of the different degrees of freedom in an observable can affect the autocorrelation time. In particular, observables containing quantities which are widely spaced generally have longer correlation times.

More generally, the size of the system you are simulating (and hence the spatial separations of the degreees of freedom) can greatly affect the correlation time, especially in the vicinity of a phase transition. Just as physical quantities like the specific heat, susceptibility etc can diverge at a critical point, the correlation time can diverge too as the system size increases near a critical point. This of course makes Monte Carlo very difficult. There is a big literature on solving this problem.

## XII. ACKNOWLEDGEMENTS

Preparation of these notes was supported by the National Science Foundation under grant NSF–ITR–0313390.

---

[1] There is one slightly confusing point here. From the description provided, one might think that in a Monte Carlo simulation one stores a vector $p_i$ and applies $T$ to it. This is completely impractical because the number of one configurations $i$ of the system is incredibly huge and we could never allocate enough memory to hold $p_i$. Instead, we generally store only one configuration $i$ at a time, and $T_{ji}$ gives us a probabilistic rule to get the (single) configuration $j$ from $i$. The probability $p_i$ appears through the average of all the single configurations over a long simulation.

[2] "Computational Physics," R.H. Landau and M.J. Páez, Wiley, 1997.

[3] "Computational Physics," N. Giordano, Prentice-Hall, 1997.

[4] "Statistical Mechanics of Phase Transitions," J. Yeomans, Oxford, 1992.

[5] M. Creutz and J. Freedman, Annals of Phys. **132**, 427 (1981).

[6] See, for exmaple, Sec. 3 of the article nato2.pdf available at http://leopard.physics.ucdavis.edu/rts/nato.html