9. MC simulation of "thermodynamic" ensembles

[Allen-Tildesley; Mandl. See also my lecture notes for "Introduction to atomistic simulations"]

Usually when physicists think about simulation of thermodynamic ensembles, they think of simulation of atomic or molecular systems, as thermodynamics has been developed with these in mind. But it turns out that these methods can often also be used to simulate problems which have nothing to do with atoms at all: hence I put the "thermodynamic" into citation marks in the title.

In this section, I will concentrate on the simulation of atomic systems. But in the next section, dealing with simulated annealing, we will give at least one example of how the very same algorithms used for atom ensembles can be used in entirely different contexts.

9.1. Introduction: why is Monte Carlo useful here?

In the statistical physics formulation of classical thermodynamics, as formulated by Maxwell, Boltzmann, and others, the properties of atomic systems are described by calculating ensemble averages. This involves calculating averages as integrals of the type (this example is for one particular ensemble, the NVT):

$$< \mathcal{A} > = \frac{\int d\mathbf{R} \mathcal{A} e^{-V/kT}}{\int d\mathbf{R} e^{-V/kT}} = \frac{\int d\mathbf{R} \mathcal{A} e^{-V/kT}}{Z}$$

where Z is the (configurational) partition function

$$Z = \int d\mathbf{R} e^{-V/kT} \tag{1}$$

where $V = V(\mathbf{R})$ is the potential energy of the atoms at positions $\mathbf{R}$.

This brings us to the reason why MC simulations can be useful here. Consider the integral (1). It is over $d\mathbf{R}$, where $\mathbf{R}$ contains the positions of all N atoms in the systems. In macroscopic systems $N \sim 10^{23}$, which is of course way too large for any computer. But it turns out that only a few hundred atoms may already be enough to get a decent thermodynamic average.

So, say we would have only $N = 100$. In that case, to evaluate the integral (1) with a computers, we would need to carry out a 300-dimensional integral.

In section 5 on this course, we saw that that MC integration is the best way to numerically determine any integral at dimensions > 6. This is why MC methods are suitable for studying thermodynamic processes.

But before we go on telling how to do that, we recall some basic statistical physics which is useful for the simulations.

9.2. Theory

9.2.1. Ensembles

[Mainly from Allen-Tildesley ch. 2.1]

Statistical physics describes a system of N particles at a given state as one point in $6N$-dimensional phase space, containing the atom positions and momenta. Any instantaneous state of the system can be written as a point

$$\mathbf{\Gamma} = (\mathbf{r}, \mathbf{p})$$

where $\mathbf{r}$ is a vector which contains the positions of the N atoms and $\mathbf{p}$ a vector which contains the momenta of the N atoms.

Any property of the system $\mathcal{A}$ is then a function of the points in phase space. The instantaneous property at a time t is

$$\mathcal{A}(\mathbf{\Gamma}(t))$$

and the macroscopically meaningful observable property $\mathcal{A}_{\rm obs}$ is the time average of this,

$$\mathcal{A}_{\rm obs} = \langle \mathcal{A}(\mathbf{\Gamma}(t)) \rangle_t = \lim_{t_{\rm obs}\to\infty} \frac{1}{t_{\rm obs}} \int_0^{t_{\rm obs}} \mathcal{A}(\mathbf{\Gamma}(t)) dt \qquad (2)$$

In experiments, the time average comes about quite naturally, since almost all experimental methods measure over much longer time scales than the characteristic time scale of atom motion.

To get a quantity $\mathcal{A}$ from simulations which corresponds to the experimental one, Eq. (2), there are two basic possible approaches. One is straightforward; simply use simulation to also determine a time average, taking a discrete sum over M time steps of length Δt:

$$\mathcal{A}_{\mathrm{MD}} = \langle \mathcal{A} \rangle_t = \lim_{M \to \infty} \frac{1}{M \Delta t} \sum_{i=1}^{M} \mathcal{A}(\mathbf{\Gamma}(i \Delta t)) dt \tag{3}$$

As indicated by the subindex, this is the approach taken in MD simulations. There the atom behaviour is followed as a function of time, so it is quite natural and straightforward to obtain the average.

The other approach which can be used in simulations is essentially the same as in conventional statistical mechanics. In the Gibbs formulation of statistical mechanics, the time average is replaced by a so called **ensemble** average. The ensemble is a collection of points $\mathbf{\Gamma}$ in phase space. They are distributed according to some probability density

$$\rho(\mathbf{\Gamma})$$

which gives the probability of being in a given state $\mathbf{\Gamma}$. The distribution ρ is determined by the

boundary conditions on the system studied. Most often, they are denoted by three letters telling what macroscopic quantities are preserved. The most common ensembles are

NVE	microcanonical	isolated
NVT	canonical	system in temperature bath
NPT	isothermal-isobaric	constant T and pressure P
μVT	grand canonical	constant chemical potential and T

We will give the weight functions ρ for the different ensembles in the subsections below, and here only discuss features common to all ensembles, for any

$$\rho = \rho_{\rm ens}$$

In MC simulations the desired thermodynamic quantities are determined as ensemble averages, as in statistical mechanics. A desired quantity $\mathcal{A}$ is obtained as

$$\langle \mathcal{A} \rangle_{\rm ens} = \frac{\sum_{\mathbf{\Gamma}} A(\mathbf{\Gamma}) \rho_{\rm ens}(\mathbf{\Gamma})}{\sum_{\mathbf{\Gamma}} \rho_{\rm ens}(\mathbf{\Gamma})} \tag{4}$$

So the idea is to obtain an average over points in phase space $\mathbf{\Gamma}$ which are "frozen in time". We do not need to simulate any real time dependence of the system, only construct a sequence of states in

phase space in the correct ensemble. It is important to keep in mind that the value of the quantity $\mathcal{A}$ will depend on which ensemble is dealt with.

9.2.2. Ergodicity

For the average in equations 3 or 4 to be a valid approximation of the real average, one has to require that the system will (or would if we would simulate forever) eventually pass through all possible points in phase space, i.e. is **ergodic.** That is, if we think of the phase space as discrete (which it actually always is on computers), one can think of the simulation (MD or MC) forming a long succession of points

$$\cdots (\mathbf{r}_i, \mathbf{p}_i) \longrightarrow (\mathbf{r}_{i+1}, \mathbf{p}_{i+1}) \longrightarrow (\mathbf{r}_{i+2}, \mathbf{p}_{i+2}) \longrightarrow (\mathbf{r}_{i+3}, \mathbf{p}_{i+3}) \cdots$$

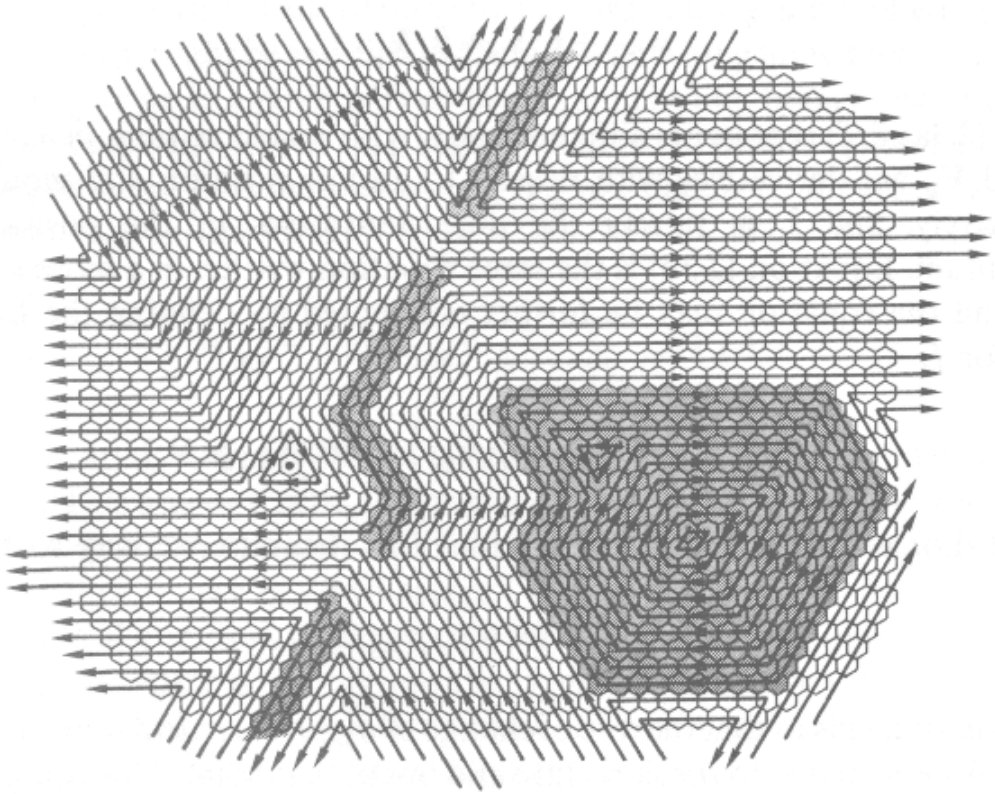
So, to reiterate, if the system is ergodic this succession of points should eventually pass through all possible points $(\mathbf{r}_{i+3}, \mathbf{p}_{i+3})$.

Such a chain of states is known as a **Markov chain** provided:

1° The outcome of each move belongs to a finite set of outcomes $\mathbf{\Gamma}_i$, $i = 1, \ldots, M$, called the **state** space.

2° The outcome n of each move $\mathbf{\Gamma}_m \longrightarrow \mathbf{\Gamma}_n$ only depends on the immediately preceding state m

We now plot a schematic of the points in phase space which illustrates both a fundamental and practical problem with ergodicity:



The **fundamental problem** is the dark area on the lower right. Here the paths are closed and never leave the area. Hence if we would start our simulation in this area, we would only sample a small region of phase space. Conversely, if we would start outside, we would never get into the darker area. So this system is not ergodic.

The **practical problem** is illustrated by the thin dark stripes in the middle. These are not a fundamental problem, since the system does come in and out of them. But they form a practical problem, because the lines to and from them both start and end on the same side of the system, either left or right. This leaves very few paths which would take one from the left to the right side, or vice versa. So this essentially forms a **barrier** which is hard (but not impossible) to cross.

In case the system is fundamentally not ergodic, one can be in real trouble if the isolated areas are big, or have a high weight in the end result. Unfortunately there is no general way to prove that a system is ergodic, so usually it is considered a hypothesis.

If the system has many or "high" barriers, one can also be in big trouble since it may be extremely unlikely that one crosses them, leading to long simulation times. Or, even worse, it may be that one never notices the existence of the barriers, and never crosses them at all.

A simple practical way to test for ergodicity problems (both of the fundamental and barrier variety) is to start the simulations in many different regions of phase space, and see if they give the same answers. If they always do, one has some confidence that there is not a problem. But this is by no means watertight: there are many systems where one tends to always start in a limited range of phase space, and may easily miss some important region of it completely, or for a long time (this

is the lecturers personal experience speaking). The best way to avoid this is have a good physical understanding of the system and try to deduce or even prove that there are no regions causing ergodicity trouble.

9.2.3. Importance sampling

We already explained the basic relation between MC simulation of thermodynamic ensembles and MC integration. There is also another central concept in MC integration which we can bring over here: importance sampling.

Recall that the idea of importance sampling was to spend the most effort in the regions of space that matter most for the end result. We can conceptually do the very same thing in MC simulation of thermodynamic ensembles. And we now have a major advantage: for all the common ensembles, we know the distribution of points in space: it is just the weight function of the ensemble

$$\rho_{\rm ens}(\mathbf{\Gamma})$$

The basic approach involves calculating

$$\langle \mathcal{A} \rangle_{\rm ens} = \frac{\sum_{\mathbf{\Gamma}} A(\mathbf{\Gamma}) \rho_{\rm ens}(\mathbf{\Gamma})}{\sum_{\mathbf{\Gamma}} \rho_{\rm ens}(\mathbf{\Gamma})} \tag{5}$$

for any points in phase space $\mathbf{\Gamma}_i$. But often most points have very little weight. As discussed before for importance sampling, we can introduce some weight function $w(\mathbf{\Gamma})$, pick the points to be

sampled according to this function, and calculate the result as

$$\langle \mathcal{A} \rangle_{\rm ens} = \frac{\sum\limits_{\substack{\mathbf{\Gamma} \\ \text{chosen from} \\ \text{weight function } w}} A(\mathbf{\Gamma}) \rho_{\rm ens}(\mathbf{\Gamma}) / w(\mathbf{\Gamma})}{\sum\limits_{\substack{\mathbf{\Gamma} \\ \text{chosen from} \\ \text{weight function } w}} \rho_{\rm ens}(\mathbf{\Gamma}) / w(\mathbf{\Gamma})} \tag{6}$$

In the general case, we did not know how to choose w. But now we know the weight of our points: it is just $\rho_{\rm ens}$! So we can choose

$$w(\mathbf{\Gamma}) = \rho_{\rm ens}(\mathbf{\Gamma})$$

and obtain the simple result:

$$\langle \mathcal{A} \rangle_{\rm ens} = \frac{\sum\limits_{\substack{\mathbf{\Gamma} \\ \text{chosen from} \\ \text{weight function } \rho}} A(\mathbf{\Gamma})}{N_{\rm MCsteps}} \tag{7}$$

Note that we can not, however, choose the points directly with the techniques for generating random numbers in an arbitrary distribution, since now the dimension of $\mathbf{\Gamma}$ is huge. How to obtain points

in the correct distribution depends on the ensemble, and hence is discussed separately for each ensemble.

9.3. Canonical (NVT) ensemble

9.3.1. Basic properties of ensemble

In the canonical (NVT) ensemble the number of particles, system volume and temperature is conserved. This corresponds to a closed system, which, however, can exchange heat with a large surrounding heat bath.

Consider a canonical ensemble with the total energy given by the Hamiltonian

$$H(\mathbf{\Gamma}) = K(\mathbf{\Gamma}) + V(\mathbf{\Gamma})$$

where K is the total kinetic energy and V the total potential energy of the system. Here we assume the particles do not have any internal energy I; if they had, that would be added to the sum, and treated separately (see e.g. Mandl ch. 7.8). In the usual classical and semiclassical formulations atoms do not have internal energy terms, but molecules have, in the form of rotational and vibrational degrees of freedom.

The weight function (density) ρ of the canonical ensemble follows a Boltzmann distribution,

$$\rho \propto e^{-H(\mathbf{\Gamma})/kT}$$

and the partition function is, for an atomic system in quasi-classical form,

$$Q_{NVT} = \frac{1}{N!}\frac{1}{h^{3N}} \int d\mathbf{r}d\mathbf{p}\ e^{-H(\mathbf{r},\mathbf{p})/kT}$$

where N is the number of particles, and h Planck's constant. The reason we mention the partition function is its central role in thermodynamics: most other important quantities can be derived from it.

The crucial property here for simulations is that the total energy of the system is not conserved. Instead, any values of the total energy are allowed, as long as it is possible to retain the desired temperature. This makes dealing with the NVT ensemble slightly difficult in MD simulations (which in the basic formulation conserve energy), but actually turns out to make life easier in MC simulations. We shall now see why.

As long as the Born-Oppenheimer approximation is valid (as it practically always is in equilibrium thermodynamics) the potential energy of the system only depends on the particle coordinates $\mathbf{r}$. And of course the kinetic energy only depends on the particle velocities, i.e. momenta $\mathbf{p}$. Hence we can rewrite the expression for the system Hamiltonian as

$$H(\mathbf{\Gamma}) = K(\mathbf{\Gamma}) + V(\mathbf{\Gamma}) = K(\mathbf{p}) + V(\mathbf{r})$$

So now we see that the partition function factorizes into a product of kinetic (ideal gas) and

potential (excess) parts:

$$Q_{NVT} = \frac{1}{N!}\frac{1}{h^{3N}} \int d\mathbf{p} e^{-K(\mathbf{p})/kT} \int d\mathbf{r} e^{-V(\mathbf{r})/kT}$$

This can be written as a product of the ideal gas contribution and the excess contribution as

$$Q_{NVT} = Q^{\mathrm{id}}_{NVT} Z_{NVT}$$

where

$$Z_{NVT} = \int d\mathbf{r} e^{-V(\mathbf{r})/kT}$$

is the so called configuration integral.

The thermodynamic function of the system is the Helmholtz free energy.

$$A = -kT \ln Q_{NVT}$$

which can now be separated:

$$A = -kT \ln(Q^{\mathrm{id}}_{NVT} Z_{NVT}) = -kT \ln Q^{\mathrm{id}}_{NVT} - kT \ln Z_{NVT}$$

The other quantities of the ensemble can be derived from A. Hence this separation is of crucial

importance: the first part,

$$kT \ln Q^{\mathrm{id}}_{NVT}$$

is just the term for an ideal gas, the properties of which can be found in any statistical physics textbooks.

Hence to treat the NVT ensemble for interacting particles, we can focus our effort on calculating the potential-energy dependent part Z_{NVT}, then add the known ideal gas properties afterwards.

For MC simulations this means that we can use simulation schemes which do not have to treat atom velocities at all, only the potential energy. Next we present a scheme to do this.

9.3.2. The Metropolis method

[Metropolis, Rosenbluth, Rosenbluth, Teller and Teller, J. Chem. Phys. 21 (1953) 1087-; Allen-Tildesley]

The original MC simulation scheme for thermodynamics is the famous Metropolis method. Although presented already in 1953, it still probably is the most widely used MC simulation scheme.

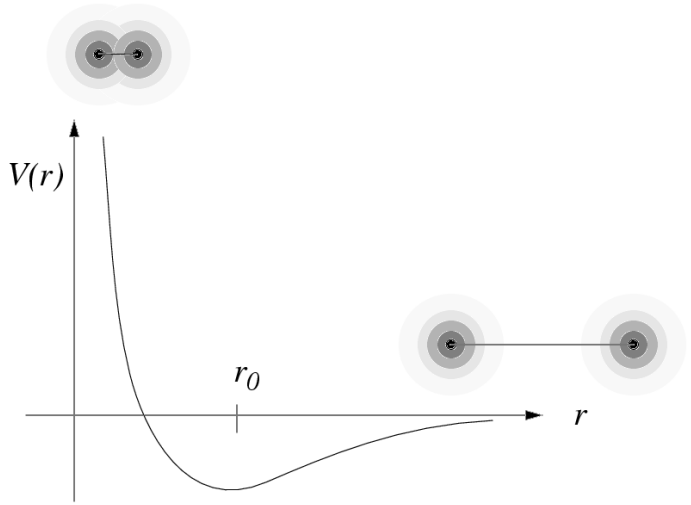
An interesting historical footnote is that essentially the same algorithm was invented independently at about the same time by Bernie Alder, the same person who a few years later developed molecular dynamics together with Wainwright [http://www.nersc.gov/ deboni/Computer.history/Alder.html]

We now present and prove the method almost identically to the original formulation. In the original paper, N hard spheres in a 2D system was considered. We will present it in a slightly more general form, for N particles in 3D interacting by an arbitrary potential energy function $V(\mathbf{r})$ which gives the potential energy for each particle i in the system $V_i(\mathbf{r})$.

9.3.2.1. What is the potential energy function?

Today, there exist a wide range of often quite good classical, analytical potential energy functions for almost all classes of solid and liquid materials. These are discussed at length on my course "Introduction to atomistic simulations" (available on the web), and hence not treated here at any length. The energy used in the algorithm can also be obtained from quantum mechanical calculations as the total electronic energy of a system of atoms and electrons.

But to give a basic flavour of what the potential energy functions involve, we mention the simplest possible case. This is that we just have a pair potential for the atoms, i.e. a potential which only depends on the separation between two atoms $V(r_{ij})$. For a binding system this potential has the qualitative form:



Then for N particles we can write the total potential energy V of the system as

$$V = \frac{1}{2} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} V(r_{ij})$$

where the factor $1/2$ comes in because all pairs are counted twice as the sum is written above.

9.3.2.2. The Metropolis algorithm

The weight function of the NVT ensemble

$$\rho \propto e^{-V/kT}$$

So now we want to do importance sampling of points with this distribution (instead of evenly in V) to obtain averages of the type given in Eq. 7,

$$\langle \mathcal{A} \rangle_{\rm ens} = \frac{\sum\limits_{\substack{\mathbf{\Gamma} \\ \text{chosen from} \\ \text{weight function } \rho}} A(\mathbf{\Gamma})}{N_{\rm MCsteps}}$$

I.e. we want to choose sampled states with a probability proportional to

$$e^{-V/kT} \tag{8}$$

In practice, one wants to move one particle at a time in the system. Then it becomes easier to work with the energy $\mathit{difference}\ \Delta E$ between the new and the previous state, rather than the energy of the whole system. This is done in the Metropolis algorithm in a way which leads to the desired probability of sampling states (8). We first present the algorithm, then prove that this leads to (8).

We consider a system with N particles at positions $\mathbf{r}_i$.

0 a° Place the particles in some initial configuration $\mathbf{r}_i$.

0 b° Choose a maximum displacement vector (d_x, d_y, d_z)

0 c° Set the number of Monte Carlo steps $n_{MCS} = 0$ and $A_{\rm sum} = 0$

1° Choose a particle i at random among the N particles

2° Calculate the energy of the system before the transition $E_b = V(\mathbf{r})$

3° Generate three uniform random numbers u_1, u_2, u_3 **between -1 and 1**

4° Displace atom i by $\Delta\mathbf{r} = (u_1 d_x, u_2 d_y, u_3 d_z)$: $\mathbf{r}_i = \mathbf{r}_i + \Delta\mathbf{r}$

5° Calculate the energy of the system after the transition $E_a = V(\mathbf{r})$

6° Calculate $\Delta E = E_a - E_b$, then :

- **7°** If $\Delta E \leq 0$ accept the state
- **8°** If $\Delta E > 0$:
  - **8 a°** Generate a random number u between 0 and 1
  - **8 b°** Accept the state only if $u < e^{-\Delta E/kT}$

9° If the state is rejected, return to the previous state: $\mathbf{r}_i = \mathbf{r}_i - \Delta\mathbf{r}$

10° Sum up the desired physical property A for positions $\mathbf{r}$: $A_{\rm sum} = A_{\rm sum} + A(\mathbf{r})$

11° Set $n_{\rm MCS} = n_{\rm MCS} + 1$

12° If $n_{\rm MCS} < n_{\rm max}$ return to step 1

13° Calculate and print out the desired average $\dfrac{A_{\rm sum}}{n_{\rm max}}$

We can now note several things directly related to the algorithm:

1. The physical properties should be calculated at each MC step regardless of whether the jump is accepted or rejected.

2. The atom velocities or forces between atoms are not needed anywhere. Indeed, it is not even clear how velocities could be defined in this scheme.

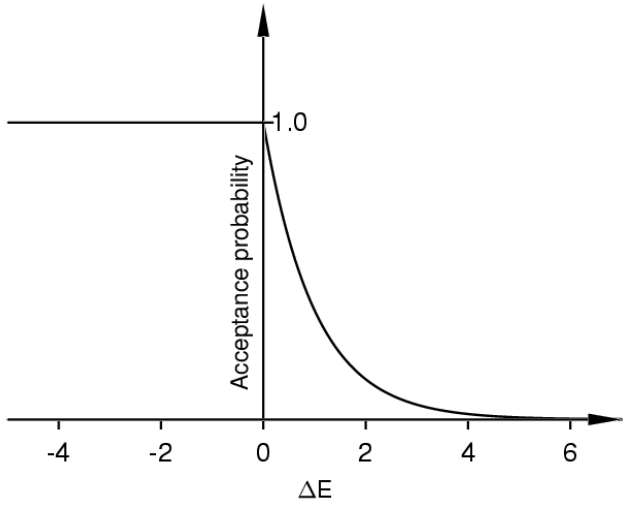
3. There is no time scale involved here. We do count the number of Monte Carlo steps n_{MCS} but it is not in any way clear whether these correspond to any real time scale.

4. The only steps really specific to an atomic system are 3° , 4° and 9° . The energy could actually be any variable measuring the state of a system, steps 3° and 4° could be replaced by any kind of a trial, and step 9° by any kind of a return-back operation. This gives a quite general algorithm useful in many contexts. In the next full section we will see how this can be used for minimization using "simulated annealing".

5. The vector $\mathbf{d}$ is clearly a free parameter. Its choice will affect how many of the trials are accepted or rejected. A frequently cited rule of thumb is that about half of the jumps should be accepted for optimally efficient probing of phase space.

6. For any given system one can find a good choice of $\mathbf{d}$ more or less by trial and error. Or one can easily devise an adaptive scheme for this: if the number of accepted jumps over the last, say, 1000 steps is much below 0.5, decrease $\mathbf{d}$, and vice versa.

The basic method of acceptance can be illustrated by the following graph, giving the acceptance rate as a function of ΔE:



9.3.2.3. Proof that this produces the right distribution of states

Although the algorithm clearly seems related to the desired distribution of states

$$e^{-V/kT}$$

it is not immediately obvious that it really produces states in this distribution. Hence we will now prove it; the proof is directly from the original paper.

We assume the ergodic hypothesis to be true, i.e. that all states can be reached. If the trial distance $\mathbf{d}$ is large enough, any atom can jump anywhere in the system. Since in this case the list of possible points in phase space only involved the atom positions, it thus seems quite likely the ergodic hypothesis really holds now.

Consider now a large number of NVT systems. We consider again discrete systems with a finite number of possible states. Let ν_r be the number of systems which are in a specific state r, which has an energy E_r. What we must prove is that after many moves in each system, the distribution of energies in the different systems tends towards the Boltzmann distribution, i.e. that

$$\nu_r \propto e^{-Er/kT} \tag{9}$$

Now let's make a move in all the systems. The probability that the transition would go from state r to state s *before* we account for the $\exp(-\Delta E/kT)$ factor is called P_{rs}. Since all states and

transitions of the type described above are equal,

$$P_{rs} = P_{sr}$$

Now assume $E_r > E_s$. Then the number of systems moving from state r to state s will simply be

$$\nu_r P_{rs}$$

since all transitions to lower energies are allowed in the Metropolis scheme. But the number of systems doing the opposite transition $s \longrightarrow r$ will be

$$\nu_s P_{sr} e^{-(E_r - E_s)/kT}$$

Now the net number of systems $M_{s \to r}$ moving from s to r is

$$M_{s \to r} = \nu_s P_{sr} e^{-(E_r - E_s)/kT} - \nu_r P_{rs} = P_{rs}(\nu_s e^{-(E_r - E_s)/kT} - \nu_r) = P_{rs}(\nu_s \frac{e^{-E_r/kT}}{e^{-E_s/kT}} - \nu_r)$$

since $P_{sr} = P_{rs}$.

From this we see that if

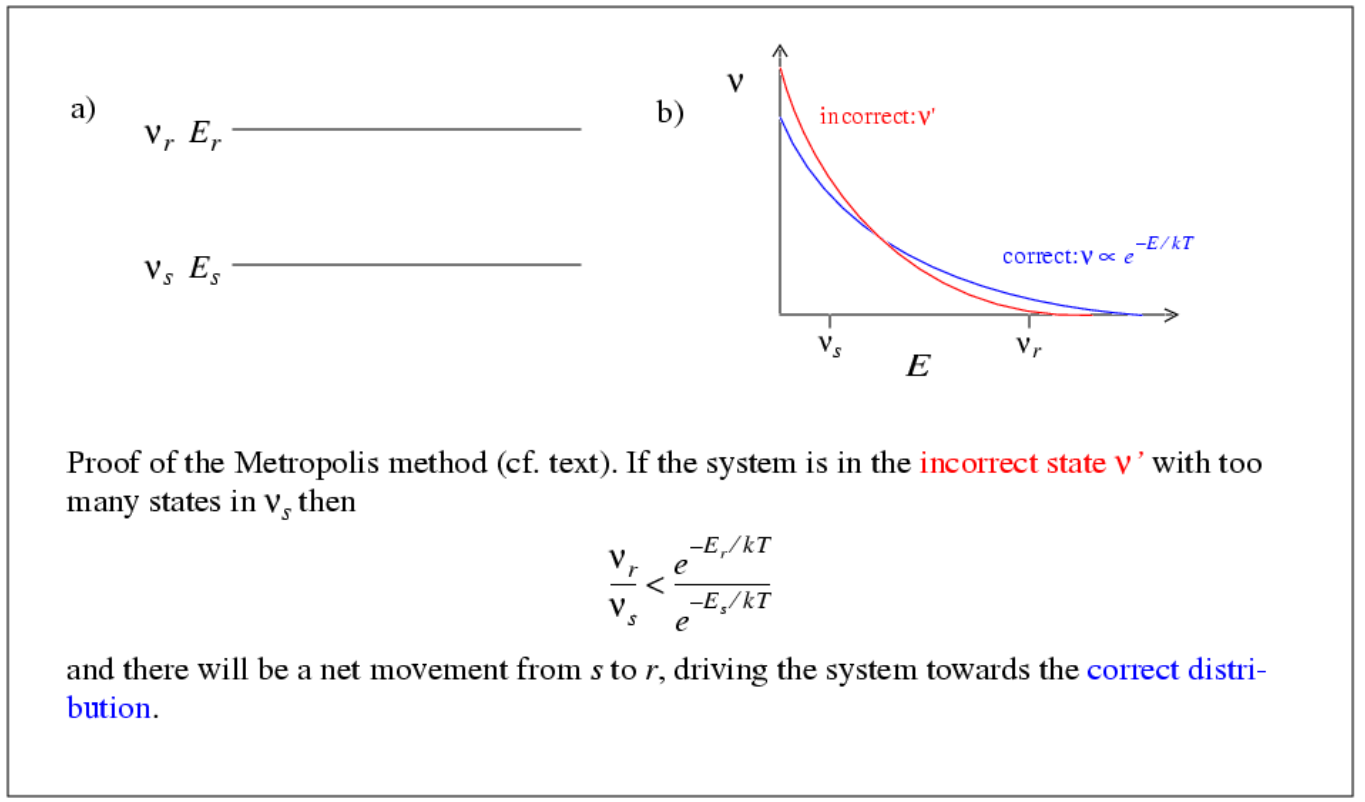
$$\nu_s \frac{e^{-E_r/kT}}{e^{-E_s/kT}} < \nu_r$$

then the net movement $M_{s \to r}$ is negative, i.e. more systems move from r to s. This clause can be rewritten by dividing with ν_s.

This says then that if

$$1^\circ \qquad \frac{\nu_r}{\nu_s} > \frac{e^{-E_r/kT}}{e^{-E_s/kT}} : \qquad \text{Net movement is from } r \text{ to } s \tag{10}$$

$$2^\circ \qquad \frac{\nu_r}{\nu_s} < \frac{e^{-E_r/kT}}{e^{-E_s/kT}} : \qquad \text{Net movement is from } s \text{ to } r \tag{11}$$



Proof of the Metropolis method (cf. text). If the system is in the incorrect state ν' with too many states in ν_s then

$$\frac{\nu_r}{\nu_s} < \frac{e^{-E_r/kT}}{e^{-E_s/kT}}$$

and there will be a net movement from s to r, driving the system towards the correct distribution.

What does this mean? Compare this with the desired result, Eq. (9). We see that if this condition is already fulfilled for both r and s, there is no net movement in any direction, i.e. the steady state is indeed given by Eq. (9), as it should.

On the other hand, let us imagine Eq. (9) is not fulfilled. Then if condition 1° is fulfilled, there are too many states in r, and the algorithm drives the system from r towards s. Conversely, if condition 2° is fulfilled, there are too many states in s, and the system is driven from s towards r. In either case, the result is that the system is driven towards the correct distribution, Eq. (9). Since we assume ergodicity, it is clear the final state will eventually be reached Hence this completes the proof.

Note, however, that neither the algorithm nor the proof gives us an idea how many steps it would actually take to reach the equilibrium distribution.

9.3.2.4. Comments on proof

We can restate the central step in the Metropolis algorithm by saying that the transition probability $r \longrightarrow s$ is

$$W_{r\to s} = \min(1, e^{-\Delta E/kT})$$

because if $\Delta E < 0$, the exponential becomes larger than 1. It turns out that the proof above is valid even for other forms of $W_{r\to s}$; any kind of transition which fulfills the so called **detailed**

balance condition

$$W_{r\to s}e^{-E_r/kT} = W_{s\to r}e^{-E_s/kT}$$

will produce the Boltzmann distribution. This is quite easy to see by comparison with the proof given above.

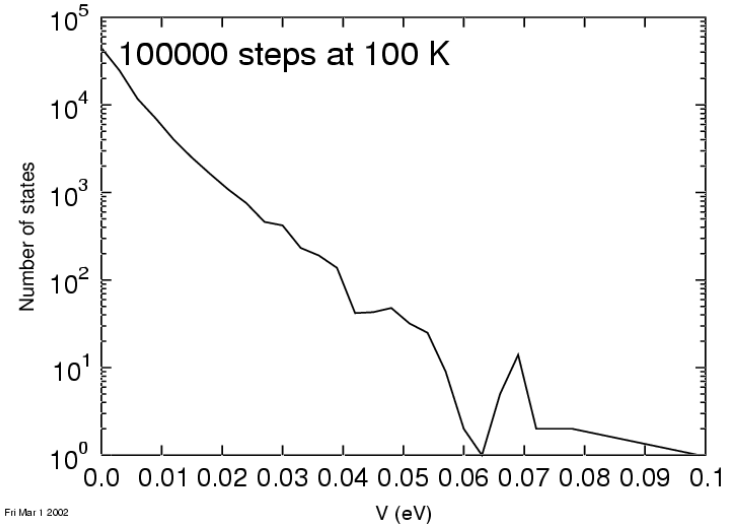
This now also explains the "detailed balance" criterion mentioned in the previous full section dealing with KMC. In case we do KMC using a transition probability which satisfies detailed balance, we can also examine the NVT ensemble.

9.3.3. Example: single particle

As a first, very basic test I implemented the Metropolis algorithm on a single particle, interacting with a simple harmonic potential:

$$V = 5x^2$$

and simulated the movement of this particle with the Metropolis algorithm (this way we do not have to worry about the density of states). The resulting distribution is



which is a nice exponential distribution to a pretty good precision.

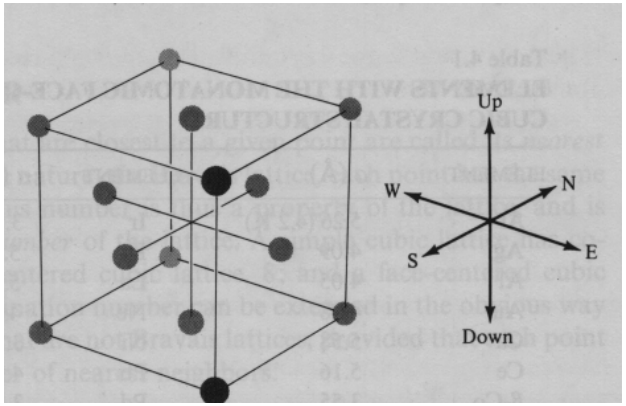
9.3.4. Example: Cu with Morse potential

To obtain a really useful example of the Metropolis MC method, I simulated Cu with the Morse potential. This interatomic potential [Morse, Phys. Rev. 34 (1930) 57] has the functional form

$$V(r) = D_0(e^{-2\alpha(r-r_0)} - 2e^{-\alpha(r-r_0)}$$

It has been parametrized for most metallic elements by Girifalco and Weizer [Phys. Rev. 114 (1959) 687]; what they essentially did is fit the three parameters to reproduce the lattice constant, cohesive energy and bulk modulus correctly. The parameters they give for Cu are $D_0 = 0.3429eV$, $\alpha = 1.3588$ 1/Å and $R_0 = 2.866$ Å. This gives a typical pair potential with a shape similar to that shown in the figure above.

The only practical problem is that the potential has infinite range. Hence I simply introduced a cutoff of 7.0 Å. From the code written for the "atomistic simulations course", I know that this potential has at 0 K the equilibrium lattice constant of $a = 3.62538$ and a cohesive (potential) energy of -3.49903 eV/atom. The crystal structure is face-centered cubic (fcc); this means that the atoms are arranged in an array of cubes, where there is an atom at every corner of a cube, and an atom at the center of each cube side face. This is a close-packed atom structure, and it is known that the Morse potential gives this as the ground-state structure for Cu.



In coding this, I did one trick for speedup. The energy of the system is calculated with a subroutine, which can be called so that it either calculates the energy of a single atom i, or the total energy of the system. In the Metropolis algorithm when applied for pair potentials, it is enough to calculate the energy on atom i to get the change in energy. In my code, this is implemented such that the energy of the atom i is calculated both before and after the trial change in a position.

An even more efficient implementation would be to keep track of the potential energy of each atom, correcting it after every accepted trial. This way one would have to calculate the energy on atom i only once every trial. But this would make the code less clear, and hence I chose not to implement this.

Also, many of the tricks presented on the "atomistic simulations" course (neighbour list and link-cell method) could be used to speed up the present implementation if the number of atoms is large.

Here is the central part of the code; the whole thing can be downloaded from the course web page.

```
do
   ! Trial jump
   ! ---------- Select atom ----------------------------
   i=int(grnd()*N)+1; if (i>N) i=N;
   ! ---------- Get old energy for atom i --------------
   call energy(atom,N,size,Eiold,i)
   ! ---------- Move it randomly -----------------------
   dx=(1.0-2.0*grnd())*djump;
   dy=(1.0-2.0*grnd())*djump;
   dz=(1.0-2.0*grnd())*djump;
   atom(i)%x=atom(i)%x+dx
   atom(i)%y=atom(i)%y+dy
   atom(i)%z=atom(i)%z+dz
   if (atom(i)%x >= size/2) atom(i)%x=atom(i)%x-size
   if (atom(i)%x < -size/2) atom(i)%x=atom(i)%x+size
   if (atom(i)%y >= size/2) atom(i)%y=atom(i)%y-size
   if (atom(i)%y < -size/2) atom(i)%y=atom(i)%y+size
   if (atom(i)%z >= size/2) atom(i)%z=atom(i)%z-size
   if (atom(i)%z < -size/2) atom(i)%z=atom(i)%z+size

   ! ---------- Get new energy for atom i --------------
   call energy(atom,N,size,Einew,i)
   accept=.false.
   if (Einew <= Eiold) then
      accept=.true.
```

```
else
   if (grnd() < exp(-(Einew-Eiold)*e/(kB*T))) accept=.true.
endif
if (accept) then
   isteplastsuccess=istep
   nsuccess=nsuccess+1
   ! Follow evolution of total energy
   Etot=Etot-Eiold+Einew
else
   atom(i)%x=atom(i)%x-dx
   atom(i)%y=atom(i)%y-dy
   atom(i)%z=atom(i)%z-dz
   if (atom(i)%x >= size/2) atom(i)%x=atom(i)%x-size
   if (atom(i)%x < -size/2) atom(i)%x=atom(i)%x+size
   if (atom(i)%y >= size/2) atom(i)%y=atom(i)%y-size
   if (atom(i)%y < -size/2) atom(i)%y=atom(i)%y+size
   if (atom(i)%z >= size/2) atom(i)%z=atom(i)%z-size
   if (atom(i)%z < -size/2) atom(i)%z=atom(i)%z+size
   nfail=nfail+1
endif
istep=istep+1

! ---------- Collect physical results ---------------
if (istep>istepcollect) then
   Esum=Esum+Etot
   Eave=Esum/(N*(istep-istepcollect))
endif

! ---------- Print out results every now and then ---
if (mod(istep,10000)==1) then
   ! Get total energy from scratch to prevent numerical
```

```
      ! drift
      call energy(atom,N,size,Etot,0)

      print *,"E",istep,1.0*nsuccess/istep,Etot/N,Eave
    endif

    if (istep>= isteplastsuccess+Nfailmax) exit
    if (istep>= istepmax) exit
  enddo
```

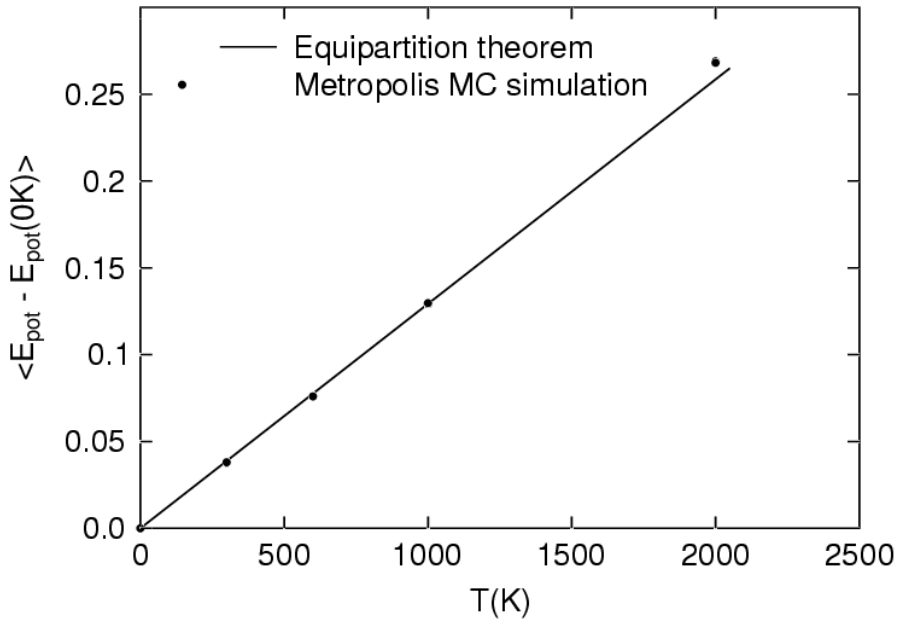
As a first test of the code (after trivial debugging), I simulated Cu starting from the perfect FCC structure at different temperatures, and calculated the average potential energy at each temperature. This tests the simulation model, since the energy equipartition theorem says that in a classical system at constant temperature, the average potential energy should be $\frac{1}{2}kT$ per degree of freedom. Now we have exactly 3 degrees of freedom per atom (x, y and z), so the average potential energy per atom, compared to the result at 0 K, should just be $\frac{3}{2}kT$

The simulations gave the following result (`djump` is the maximum jump distance in the Metropolis model)

T	djump	Eave	Eave-Eave(0K)	Equipartition prediction
0	0.2	-3.4990306	0.0	0.0
300	0.2	-3.4612282	0.03780	0.038778
300	0.2	-3.4609813	0.0380493	0.038778

600	0.2	-3.4229604	0.07607	0.07755
600	0.2	-3.4230638	0.075967	0.07755
1000	0.2	-3.3699146	0.129116	0.12926
1000	0.2	-3.3693646	0.129666	0.12926
2000	0.2	-3.2305089	0.268521	0.25852
2000	0.2	-3.2301234	0.268907	0.25852
2000	0.4	-3.2310927	0.267937	0.25852
2000	0.4	-3.2309152	0.268115	0.25852

The comparison is better illustrated by a graph:



which clearly shows that up to 1000 K, the theoretical and simulation prediction are in excellent agreement. This supports the theoretical prediction that the Metropolis method indeed produces the NVT ensemble.

At 2000 K there is some deviation. I did not analyze what could be the reason to this, but there are several possibilities. One is that the artificial cutoff at 7 Å causes distortions in the data. Another is that at this temperature defects starts forming (2000 K is actually more than the experimental melting point of Cu, but not more than the melting point of this crude Morse model of Cu). A third is that because the potential is not harmonic, anharmonic effects start to play in at high temperatures.

After this encouraging result, I did a much more ambitious test. I generated 256 random atom positions for a cell with a size which corresponds to $4 \times 4 \times 4$ unit cells in Cu. So the equilibrium structure of this system is the perfect FCC at all temperatures below melting. So the Metropolis MC method should in principle find the fcc structure, and then return the same values for `Eave` as in the table above, for each T.

But in the simulations I never managed to get to the correct ground-state structure from the random positions, despite trying fairly hard. Here are some results:

T	djump	Nsteps	fraction of failures	Eave
2000	0.2	1e6	0.412087	-3.061507
1000	0.2	1e6	0.268201	-3.210508
1000	0.4	1e6	6.099085E-02	-3.202355
600	0.2	1e6	0.172835	-3.269553
600	0.2	3e6	0.171711	-3.284650
300	0.2	3e6	7.716486E-02	-3.304991
300	0.1	3e6	0.311042	-3.316886
100	0.1	3e6	0.110436	-3.334477
100	0.03	10e6	0.577190	-3.347841

Comparison with the Eave results in the table above shows that we always are above the desired result, by some ~ 0.2 eV/atom. Experimentally speaking, this is huge; the stored potential energy would, if suddenly converted to kinetic energy, heat the sample by some 1500 K! In real Cu, this would melt the whole sample.

This shows that although the system almost certainly is ergodic, there are large barriers which may prevent sampling all of phase space, and hence leading to completely wrong results.

And it is especially important to realize that it was only our good physical understanding of this system which allowed us to know that we are getting the wrong results: the latter group of simulations, starting from the random positions, do not directly reveal that we do not obtain the right structure. Although the fact that different choices of `Nsteps` and `djump` lead to different results, should have given us a clue that something is fishy.

In the next full section, simulated annealing, we will describe a method which helps exceed the barriers in the system.

9.3.5. Other examples of the use of Metropolis MC

[Gould-Tobochnik ch. 17.3-]

Here we will list a few other applications of the Metropolis MC method, to give a flavour of what it can do. This list is anywhere from complete; there must be today so many applications of the Metropolis MC method that it is impossible for anyone to find and enumerate them all. An INSPEC search (which covers all scientific publications of physics, electrical engineering and computer science in 1969-2001) gave 1056 hits on "Metropolis", but most papers using the method probably do not mention the name "Metropolis".

9.3.5.1. Ideal gas

Above we showed essentially that the Metropolis method produces the NVT ensemble $because$ it produces an exponential distribution

$$e^{-E/kT}.$$

Above E was the potential energy of the system. But it can also be some other energy. As a simple example, consider an ideal gas. In that, the particles do per definition not interact, so there is no potential energy V at all. But instead we do have a kinetic energy $K = 1/2mv^2$, and this is also

Boltzmann-distributed. The partition function is of the form

$$Z \propto \int d\mathbf{v} e^{-(1/2mv^2)/kT} \tag{12}$$

where $\mathbf{v}$ is the velocity vector and v the velocity. So we can use the Metropolis method just as above to examine the probability distribution of velocities $P(v)$, simply by replacing the position coordinates $\mathbf{r}_i$ with the velocities $\mathbf{p}_i$, and the potential energy V with the kinetic one K. The trial velocity change can be as above just displacing each velocity with some

$$\Delta\mathbf{v} = (u_1 d_x, u_2 d_y, u_3 d_z) : \mathbf{v}_i = \mathbf{v}_i + \Delta\mathbf{v}$$

where u_i are uniform random numbers between -1 and 1.

9.3.5.2. Planar spin in an external magnetic field

Consider a classical magnet with a magnetic moment μ_0. The magnet can be oriented in any direction $\mathbf{d}$ in the x-y plane, and has (from basic electrodynamics) the interaction energy

$$E = -\mu_0 B \cos\phi$$

with an external magnetic field $\mathbf{B}$. Here ϕ is the angle between the field $\mathbf{B}$ and the direction $\mathbf{d}$.

We can use the Metropolis method to simulate this system at a temperature T. The energy is given directly by the simple equation above, and the trial would naturally doing a small change $\Delta\phi$ in

the angle ϕ. This would allow us to examine questions like what the mean energy of the system is at different temperatures, and the probability density of different angles.

Of course this simple problem can be solved analytically as well, but more complex variants of it may not.

9.3.5.3. The Ising model

[Gould-Tobochnik 17.4; http://www.nyu.edu/classes/tuckerman/stat.mech/lectures/lecture_26/node2.html (stored as tuckerman_ising.html)]

The Ising model is probably the single most studied model in computational physics. At its surface, it is the simplest possible model of ferromagnetism.

The source of ferromagnetism in materials is the quantum mechanical exchange interaction between particles (outer electrons of atoms) with a spin. That is, it originates from the requirement that the wave function of a pair of electrons is antisymmetric. If the wave functions of the outermost electrons of nearby atoms overlap strongly, this leads to a preference of the spins to be aligned in parallel. The simplest possible way to describe this is to

a) assume that only nearest-neighbour spins interact,

b) that the lattice is a square lattice with one spin per lattice point,

c) that spins can only point in 2 directions ("up" and "down")

These assumptions lead directly to the Ising model. Although all of the approximations are of questionable validity in a real ferromagnetic material, it turns out that the Ising model still does describe many central features of ferromagnetism qualitatively correctly. In particular, it can predict the main feature characteristic of ferromagnetism, that the material has a spontaneous magnetization M at low temperatures, which falls to zero at some critical temperature T_c. In the Ising model $kT_c \approx 2.269$.

However, the Ising model is not really good enough to describe true experimental ferromagnetism in detail. For instance, any solid state physics course will explain to you that at ferromagnetic domain boundaries the transition in spin directions is gradual, which is not possible due to approximation (c) in the Ising model.

But it turns out that the Ising model can be applied to an amazingly wide range of problems in science, some well beyond physics. For instance, the one-dimensional Ising model has been applied to study cell membranes, immune-system modelling, and social behavior [Ising Centennial Colloquium, Brazilian Journal of Physics 30(4) 2000].

The wide applicability of the Ising model is for instance illustrated in that an INSPEC search (Mar 14, 2004) yielded 16151 hits on "Ising model", but only 3107 hits on "Ising model and (magnetism or ferromagnetism)". That is, about 80 % of the papers on the Ising model apparently do not deal with magnetism.

The third reason of the popularity of the Ising model is that in 1 and 2 dimensions it's main features can be solved analytically. In 1D this is not hard, in 2D it was done by Onsager in 1944 [L.?Onsager, Crystal Statistics. I. A Two-Dimensional Model with a Order-Disorder Transition.Phys. Rev. 65, 117-149, 1944.]. If you want the Nobel prize, just solve the 3D Ising model analytically...

The Ising model handles the energy of a spin system as a sum of nearest-neighbour spins,

$$E = -J \sum_{i,j=nn(i)}^{N} s_i s_j - \mu_0 B \sum_{i=1}^{N} s_i \tag{13}$$

Here the summation takes every spin pair into account once, i.e. double summation should be avoided. Here s is a spin state, which has always the same magnitude but can be $+1$ or -1 (up or down). J gives the energy scale. B is an external magnetic field, and can be zero. If the model is 2D, there are 4 nearest-neighbour spins.

The physical quantities of interest include the mean energy

$$< E > \tag{14}$$

and the heat capacity

$$C = \frac{\partial < E >}{\partial T} = \frac{1}{kT^2}\left(< E^2 > - < E >^2 \right) \tag{15}$$

where the former part is the standard definition, and the latter part is a well-known thermodynamic equality which allows for calculating C from the average fluctuation in the system.

Also of interest is the magnetization

$$M = \sum_{i=1}^{N} s_i \tag{16}$$

and the magnetic susceptibility

$$\chi = \lim_{H \to 0} \frac{\partial < M >}{\partial H} \tag{17}$$

where H is the magnetic field

$$H = \frac{1}{\mu_0} B - M \tag{18}$$

Similarly to the heat capacity, χ can also be calculated from the fluctuations, as

$$\chi = \frac{1}{kT}\left(< M^2 > - < M >^2 \right) \tag{19}$$

where the M are evaluated at $H = 0$.

The Ising model can be readily simulated with the Metropolis method. There are many possible ways to do this; we will only deal with the most obvious one. That is simply making the Metropolis trial the operation of flipping a single spin:

$$s_i \longrightarrow -s_i,$$

then calculating the associated energy change ΔE and using the Metropolis weight function to determine whether the trial is accepted.

The energy change due to the flipping of spin i is

$$\Delta E = -2J \sum_{j=nn(i)} s_i s_j - \mu_0 B s_i \tag{20}$$

Hence the Metropolis trial term is

$$e^{-\Delta E/kT} = e^{\frac{2J}{kT}\sum_{j=nn(i)} s_i s_j} e^{\frac{-\mu_0 B}{kT} s_i} \tag{21}$$

from where we see that a natural temperature unit is the ratio J/k.

The boundaries of the Ising model are conventionally set to be periodic, i.e. a spin at the right edge has as its right-hand neighbour the spin on the opposite side on the left edge.

I will not present a code to do this, since it is pretty obvious how that should be done, and besides the web is full of Metropolis MC codes for the Ising model (a Google search on "ising.c" gave 480 hits).

To get an intuitive feeling for how the Ising model works, the reader is recommended to find one of the numerous "Ising model java applets" in the web and test it out interactively.

The main qualitative features are obvious: at low temperatures all spins will be aligned in the same direction, since this gives the minimum energy state. At very high temperatures, when $kT >> J$ almost all trials will be accepted, i.e. the temperature fluctuations will remove all order from the system.

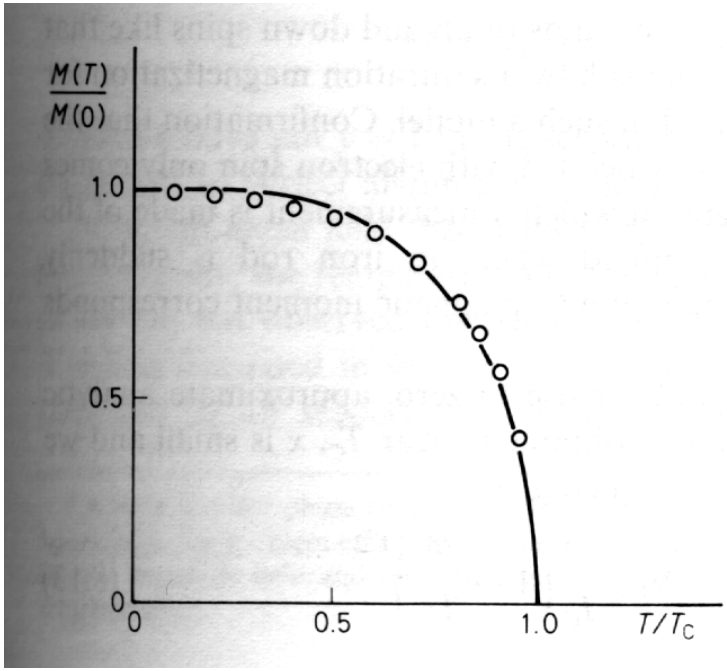
One noteworthy slightly non-obvious feature of the Ising model is that it takes a while for the system to reach an equilibrium state for a given temperature. Hence before one starts calculating the thermodynamic quantities of interest, one should equilibrate the system for a while. To determine when equilibrium is reached, one can at the simplest just follow the "time" evolution of the quantities of interest until they fluctuate around an average. A more precise way of examining the equilibration is to use the time displaced autocorrelation function

$$C_A(t) = \frac{< A(t+t_0)A(t_0) > - < A >^2}{< A^2 > - < A >^2} \tag{22}$$

where A is a physical quantity of interest, e.g. the energy E. The averages are over all possible times of origin t_0, which means that in the program one needs extra storage arrays which collect data over some time interval. $C_A(t)$ measures the correlation between the system state at time t and time 0, and will go towards 0 as $t \longrightarrow \infty$. Hence one can set some small threshold value for C_A to determine when equilibrium has been reached.

9.3.5.4. The Ising phase transition

One of the most popular applications of the Ising model is the study of the Ising phase transition. The transitions between the nonmagnetic and magnetic state is abrupt in the infinite Ising model, just as in real ferromagnetic materials:



In a computer model with a finite number of spins the transition is gradual, but still it is possible to get fairly close to the abrupt behaviour.

The behaviour of the system near the critical temperature T_c is often studied by looking at what power laws the system properties follow. For instance, the magnetization can be described as

$$M \propto (T - T_c)^{\beta} \tag{23}$$

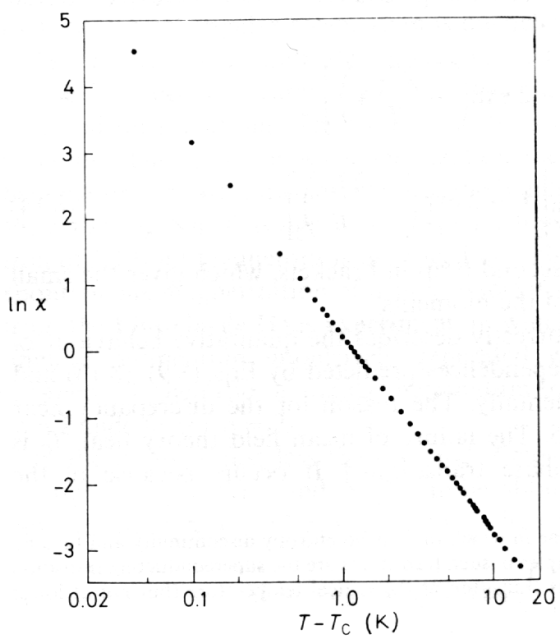
where β is the so called **critical** exponent. Similarly one can write

$$\chi \propto |T - T_c|^{-\gamma} \tag{24}$$

and

$$C \propto |T - T_c|^{-\alpha} \tag{25}$$

Below the behaviour of χ near T_c is illustrated. This is actually a plot of experimental data for iron, giving $\gamma = 1.33$:



Onsagers exact solution of the Ising model gives $\alpha = 0$, $\beta = 1/8$ and $\gamma = 7/4$. Comparison of

the experimental γ value for iron 1.33 with 7/4=1.75 already shows that the Ising model does not describe true ferromagnetism exactly right.

Practical simulation of the Ising model is inefficient near the critical point, since the fluctuations are large There are more advanced Metropolis schemes than the one presented above (e.g. the cluster algorithms) which can be used to study the Ising model more efficiently near the critical temperature. However, on this course we will not delve deeper on these. If interested, read e.g. chapter 17.11 of Gould-Tobochnik 2nd ed. for an introduction.

9.4. Microcanonical (NVE) ensemble

9.4.1. Basic properties of the ensemble

[Allen-Tildesley ch 2.2, Mandl]

In the microcanonical (NVE) ensemble the number of particles, system volume and total energy is conserved. This corresponds to a completely closed system which does not interact in any way with the environment, and is in a container of fixed volume V.

Again we will for simplicity neglect internal degrees of freedom. Then the system energy will again be a sum of kinetic and potential energies. Now since the total energy E must be conserved, the criterion will simply be that the total Hamiltonian

$$H(\mathbf{\Gamma}) = K(\mathbf{\Gamma}) + V(\mathbf{\Gamma}) = \text{constant} = E$$

which means that not all, but only those states in phase space $\mathbf{\Gamma}$ that conserve the total energy are allowed. This can also be stated so that the probability density of the ensemble is

$$\delta(H(\mathbf{\Gamma}) - E)$$

where the δ is the Kronecker delta for a discrete system, and the Dirac delta function for a continuous one.

The partition function can be written

$$Q_{NVE} = \frac{1}{N!}\frac{1}{h^{3N}} \int d\mathbf{r} d\mathbf{p}\ \delta(H(\mathbf{\Gamma}) - E)$$

where N is the number of particles, and h Planck's constant.

Simulating this ensemble with MD is straightforward, since numerical solution of Newton's equations of motion conserve energy (as they of course should). For MC we need to find some algorithm which will produce the NVE ensemble. In the next subsection I present such a method.

9.4.2. The demon algorithm

[G+T ch. 16-3; Creutz, Phys. Rev. Lett. 50 (1983) 1411]

We rewrite the microcanonical partition function as

$$Q_{NVE} = \frac{1}{N!}\frac{1}{h^{3N}} \int d\mathbf{r}d\mathbf{p}\ \delta(K(\mathbf{p}) + V(\mathbf{r}) - E)$$

which just emphasizes that the total energy has potential V and kinetic energy K parts. For MC simulations similar to the Metropolis type, we have the problem that we do not have any treatment of the atom kinetic energies, and hence can not calculate $K(\mathbf{p})$ explicitly.

In the demon method devised by Creutz in 1993, the main idea is to in essence replace the kinetic energy of all N atoms,

$$K(\mathbf{p}) = \sum_{i=1}^{N} \frac{p^2}{2m}$$

with a single "demon energy" E_D which simulates the true sum, i.e.

$$Q_{NVE} = \frac{1}{N!}\frac{1}{h^{3N}} \int d\mathbf{r}d\mathbf{p}\ \delta(E_D + V(\mathbf{r}) - E)$$

Note, however, that E_D and $K(\mathbf{p})$ are not directly comparable. The demon runs around the system, exchanging energy with the particles in it either in sequential or random order. But there has to be some restriction on how it can do it, to prevent it from sucking up all the energy in the system. The restriction is simply that the demon energy should always be positive (which is quite natural, as the system kinetic energy of course always stays constant).

The algorithm can be written as follows. The parts which differ from the Metropolis algorithm are **written in bold**.

0 a° Place the particles in some initial configuration $\mathbf{r}_i$.

0 b° Choose a maximum displacement vector (d_x, d_y, d_z)

0 c° Set the number of Monte Carlo steps $n_{MCS} = 0$ and $A_{\rm sum} = 0$

0 d° Set the demon energy $E_D = 0$.

1° Choose a particle i at random among the N particles

2° Calculate the energy of the system before the transition $E_b = V(\mathbf{r})$

3° Generate three uniform random numbers u_1, u_2, u_3 between -1 and 1

4° Displace atom i by $\Delta\mathbf{r} = (u_1 d_x, u_2 d_y, u_3 d_z)$: $\mathbf{r}_i = \mathbf{r}_i + \Delta\mathbf{r}$

5° Calculate the energy of the system after the transition $E_a = V(\mathbf{r})$

6° Calculate $\Delta E = E_a - E_b$, then :

- **7°** If $\Delta E \leq 0$ accept the state and **give the energy ΔE to the demon,** $E_D = E_D - \Delta E$
- **8°** If $\Delta E > 0$:
  - **8 a° If $E_D \geq \Delta E$ let the demon give the necessary energy to the system, $E_D = E_D - \Delta E$, and accept the configuration**
  - **8 b° Otherwise the configuration is rejected.**

9° If the state is rejected, return to the previous state: $\mathbf{r}_i = \mathbf{r}_i - \Delta\mathbf{r}$

10° Sum up the desired physical property A for positions $\mathbf{r}$: $A_{\rm sum} = A_{\rm sum} + A(\mathbf{r})$

11° Set $n_{\rm MCS} = n_{\rm MCS} + 1$

12° If $n_{\rm MCS} < n_{\rm max}$ return to step 1

13° Calculate and print out the desired average $\dfrac{A_{\rm sum}}{n_{\rm max}}$

Note that it indeed is only 3 steps which differ from the Metropolis approach. So if we have a Metropolis MC (NVT) code for a given system, it is very easy to turn it into a demon (NVE) code. Similarly, many of the basic features of the Metropolis method, such as how the physical quantities are calculated, and that there are no atom velocities, are directly valid here as well.

We can now note a few other things directly related to this particular algorithm:

1. The total energy (demon E_D + atom potential energy E_{pot}) obviously always stays constant. But because the demon is a single degree of freedom, compared to the many atom degrees of freedom, the fluctuations in the atom potential energy are only of order $1/N$.

2. The energy of the demon will have a Boltzmann distribution,

$$P(E_D) \propto e^{-E_D/kT}$$

This observation allows us to define a temperature in the system as

$$kT = \langle E_D \rangle$$

where $\langle E_D \rangle$ is the long-term average of the demon energy.

3. The energy of the demon will be very small compared to the energy of the whole system, for large numbers of atoms.

This algorithm can be converted to the Metropolis algorithm at least in two ways. If the demon is connected to a heat bath at each time step just before step **8**$^\circ$, i.e. given a new energy with a Boltzmann weight $\exp(-E/kT)$, then the algorithm will identically become the Metropolis one.

Also, if one releases a large number of demons rather than only one, their total energy will become appreciable, and with increasing numbers of demons one will eventually reach the Metropolis (NVT) distribution. This means one can smoothly move between the NVE and NVT ensembles by modifying the number of demons!

9.4.2.1. Example: Cu again

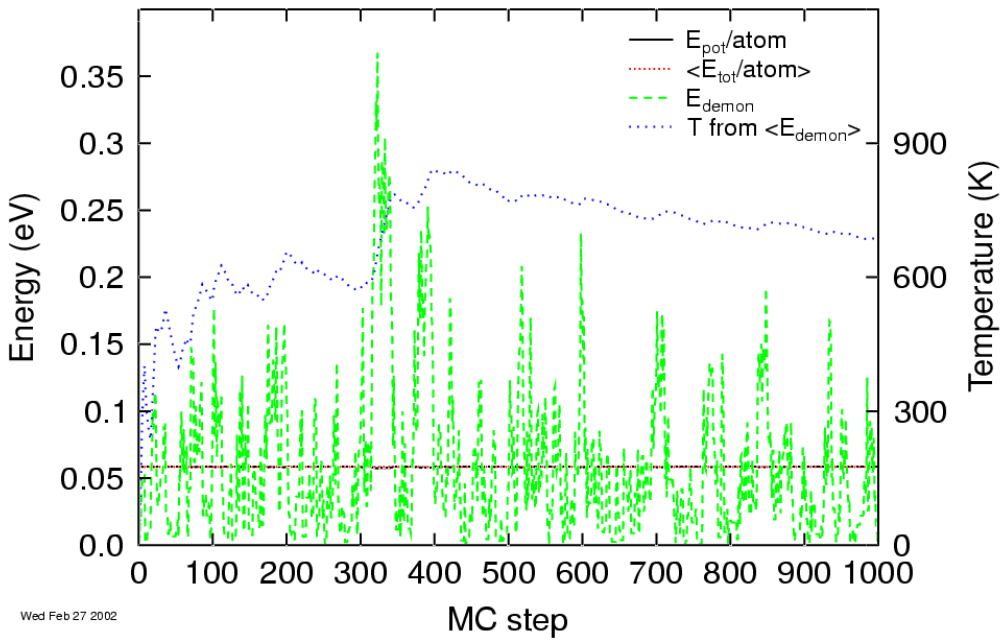
As an example, I used the demon algorithm to simulate the same 256 atom FCC Morse Cu system used to test the Metropolis approach. Rewriting the Metropolis code to a "demon" code only took a few minutes; the main routine is identical except for the way in which the trials are tested for. The central section of the code is

```
do
   ! ------------------ Trial jump ----------------------
   (OMITTED, SEE METROPOLIS MC CODE)
```

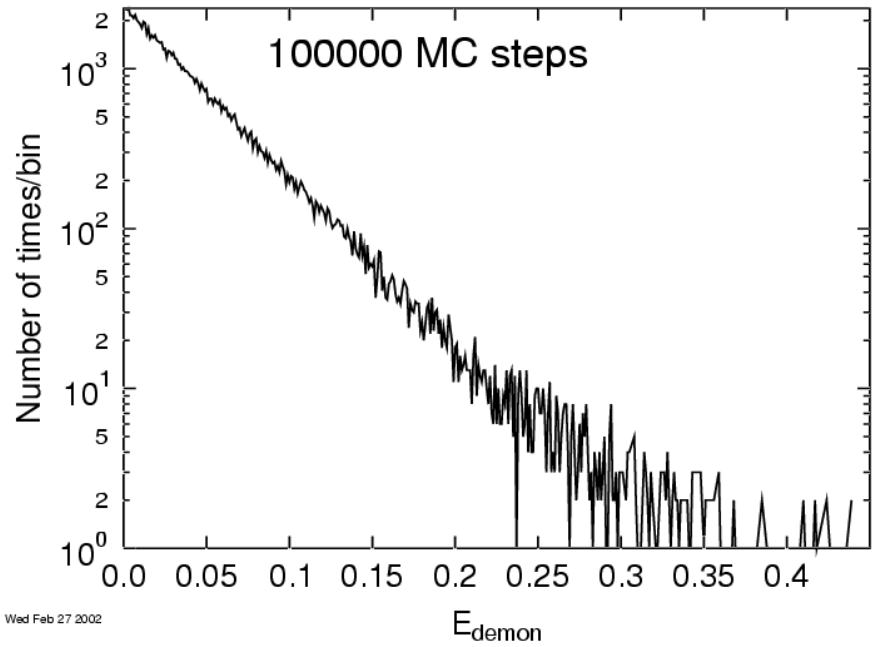
```
! ---------- Get new energy for trial atom i ---------
call energy(atom,N,size,Einew,i)

! -------------- Demon algorithm ---------------------
accept=.false.
DeltaE=Einew-Eiold
if (DeltaE <= 0.0) then
   accept=.true.
   Edemon=Edemon-DeltaE
else
   if (Edemon > DeltaE) then
      accept=.true.
      Edemon=Edemon-DeltaE
   else
      accept=.false.
   endif
endif
if (accept) then
   isteplastsuccess=istep
   nsuccess=nsuccess+1
   ! Follow evolution of total energy
   Etot=Etot-Eiold+Einew
else
   (OMITTED, SEE METROPOLIS MC CODE)
endif
istep=istep+1
```

```
   ! -----------  Collect physical results --------------
   if (istep>istepcollect) then
      Esum=Esum+(Etot-Eorig)
      Eave=Esum/(N*(istep-istepcollect))
      Edemonsum=Edemonsum+Edemon
      Edemonave=Edemonsum/(istep-istepcollect)
   endif

   ! Estimate temperature assuming Ed is Boltzmann distributed
   T=Edemonave*e/kB
   print *,"E",istep,1.0*nsuccess/istep,(Etot-Eorig)/N,Eave,Edemon,T

   ! --------- Print out results every now and then ----------
   (OMITTED, SEE METROPOLIS MC CODE)
enddo
```

I ran this for a while, and printed out $E_{\rm pot}$ (calculated above the ground state), $\langle E_{\rm pot}\rangle$ $E_{\rm demon}$, and the temperature calculated using $k_BT = \langle E_{\rm demon}\rangle$. This gave the following result:



We see that the demon energy fluctuates heavily, as it should. The potential energy per atom, on the other hand, is quite constant, as it should be by the prediction that the energy fluctuations in the atomic system are only of the order of $1/N$. If we make a statistics out of the demon energy, we see that it indeed has a Boltzmann distribution:



Note that we should not interpret

$$k_B T_{\text{instantaneous}} = E_{\text{demon,instantaneous}} \qquad \text{(not good)}$$

as the instantaneous temperature of the system, since in a real system the temperature does not fluctuate nearly as heavily as the demon energy. It is only the interpretation of the average demon energy $\langle E_{\text{demon}} \rangle$ as corresponding to temperature that makes sense!

9.5. Isobaric-isothermal (NPT) ensemble

[Allen-Tildesley 4.5, Gould-Tobochnik 17.9]

The way to simulate the NPT ensemble is in principle quite similar to that in the NVT ensemble, the main difference being that the volume is not held constant, and we need a 'volume' trial as well.

To avoid confusion in the notation, we now use the following symbols:

$$E \quad \text{for the Hamiltonian (previously } H\text{)}$$

$$U \quad \text{for the potential energy (previously } V\text{)}$$

The probability density for this ensemble is

$$e^{-(E+PV)/kT}$$

i.e. we also have the pressure term determining what pressure we are working on, and the volume V has been added to the set of state points $\mathbf{\Gamma}$. The characteristic thermodynamic quantity is the enthalpy H

$$H = E + PV$$

The partition function is

$$Q_{NPT} = \frac{1}{N!}\frac{1}{h^{3N}}\frac{1}{V_0}\int dV \int d\mathbf{r}d\mathbf{p}\; e^{-(E+PV)/kT}$$

and by leaving out the kinetic energy terms we get the configuration integral

$$Z_{NPT} = \int dV e^{-PV/kT} \int d\mathbf{r} e^{-U/KT}$$

The average of a quantity A is obtained using

$$\langle A \rangle_{NPT} = \frac{\int_0^\infty dV e^{-PV/kT} V^N \int d\mathbf{s} A(\mathbf{s}) e^{-U(\mathbf{s})/kT}}{Z_{NPT}}$$

Here an important change to the usual setup has been made: instead of the normal coordinates $\mathbf{r}$ ranging from 0 to L (or $-L/2$ to $L/2$) in each dimension (L is the system size in each dimension), we use unitless position coordinates

$$\mathbf{s} = \frac{\mathbf{r}}{L}$$

which range from 0 to 1, or -1/2 to 1/2. This is because when the system volume keeps changing, the coordinates $\mathbf{r}$ would need to be scaled to the new size every time the size is changed. But the coordinates $\mathbf{s}$ stay unchanged!

Now the distribution to be approached is also different. Since the equation to calculate an average for the quantity A has a factor of

$$e^{-PV/kT}V^N = e^{-PV/kT}e^{\ln V^N} = e^{-PV/kT+N\ln V}$$

the distribution to be approached is

$$e^{-U(\mathbf{s})/kT-PV/kT+N\ln V} \tag{26}$$

The way to simulate the NPT ensemble is to use a Metropolis scheme, but instead of just the trial on moving an atom, there is an additional trial on changing the volume. This trial is conceptually identical to the moving of an atom:

3 V° Generate random number u_V between -1 and 1

4 V° Change system size V by $\Delta V = u_V d_V$: $V_t = V + \Delta V$

where d_V is some predetermined maximum volume change.

Then we evaluate the change in the exponent of Eq. 26,

$$\Delta W = \Delta U(\mathbf{s}) + P(V_t - V) - NkT\ln\frac{V_t}{V}$$

and accept the trial as follows:

7 V° If $\Delta W \leq 0$ accept the state

8 V° If $\Delta W > 0$:

8 a V° Generate a random number u between 0 and 1

8 b V° Accept the state only if $u < e^{-\Delta W/kT}$

A few things can be noted here:

1. Calculating the new $U(\mathbf{s})$ requires calculating all interactions from scratch. Hence the volume trial is much slower than the atom trial, and it is not worth attempting it at every step.

2. P is the desired pressure. Note that we do not necessarily need to be able to calculate the pressure; we can in principle just trust that the algorithm should lead to the right pressure.

3. But in practice it is almost certainly healthy to be able to calculate the pressure to check that we are coming to the right answer. This can be either done exactly by calculating the virial of the

system, or approximately by numerically evaluating $\frac{\partial U}{\partial V}$. Calculating the virial requires information on the forces between atoms, which are not otherwise needed in MC, so doing the numerical estimate is much easier to implement, and in my experience often accurate enough. How the virial is calculated is explained below.

9.6. Grand canonical (μVT) ensemble

[Allen-Tildesley 4.6, Gould-Tobochnik 17.9]

In the grand canonical ensemble, a system is in contact with a heat bath with which it also can exchange particles. The probability density for this ensemble is

$$e^{-(E-\mu N)/kT}$$

where μ is the chemical potential, i.e. the derivative of the free energy with respect to the the number of particles in the system,

$$\mu = \frac{\partial G}{\partial n}$$

If there is only a single component in the system this reduced to the simple form

$$\mu = \frac{G}{N}$$

The partition function is

$$Q_{\mu VT} = \sum_N \frac{1}{N!}\frac{1}{h^{3N}} e^{\mu N/kT} \int d\mathbf{r} d\mathbf{p}\; e^{-E/kT}$$

and the average of a quantity A is obtained using

$$\langle A \rangle_{\mu VT} = \frac{\sum_{N=0}^{\infty} \frac{1}{N!} V^N \left(\frac{e^{\mu/kT}}{\Lambda} \right)^N \int d\mathbf{s} A(\mathbf{s}) e^{-U(\mathbf{s})/kT}}{Q_{\mu VT}}$$

where

$$\Lambda = \sqrt{\frac{h^2}{2\pi m k T}}$$

is the thermal de Broigle wavelength

This is a bit complicated, and on this course we do not have the time to go into the meaning of all the terms. But the important thing now is that the distribution we want to reach is

$$e^{-U(\mathbf{s})/kT + \mu N/kT - N \ln \Lambda^3 - \ln N! + N \ln V} \tag{27}$$

Now there is no single clear way in which we can achieve this. The ordinary particle trial is still as in the Metropolis scheme. But we also need to modify the number of particles in the simulation, to see which average number is the equilibrium one. One way to do this is create another reservoir system of "ghost" particles which move around freely, and sometimes allow particles to switch between the real and ghost system, with a Metropolis scheme utilizing Eq. 27.

Another, and according to Allen-Tildesley the most used scheme is the simpler one, in which there are three different kinds of trials:

a$^\circ$ Displace a particle, as in original Metropolis scheme

b$^\circ$ Destroy a particle

c$^\circ$ Create a new particle in a random position in the system

The criteria for accepting trials b$^\circ$ and c$^\circ$ are as follows. Denote

$$\mu^* = \mu - kT \ln \frac{\Lambda^3}{V}$$

which allows the exponent in 27 to be written as

$$W = \mu^* N/kT + \ln N! + U/kT$$

If W is increased both trials b$^\circ$ and c$^\circ$ are accepted. If W decreases, the change is accepted with probability

b, destruction: $N e^{-(\mu^* + (U_{N+1} - U_N))/kT}$

c, creation: $\frac{1}{N+1} e^{(\mu^* - (U_{N+1} - U_N))/kT}$

Since we want to simulate constant T, V and μ, μ^* is a constant input parameter for the simulation.

In practical implementation of this, one would have to consider that when a particle is created or destroyed, empty spaces may be left in the arrays. One could either move particles down to fill these in, but this is rather inefficient since the particle information may be in several different arrays. Another option is to use a separate array `LOCATE` which points to the particles which are alive.

9.7. Calculating thermodynamic quantities

So now we know how to simulate the most important thermodynamic ensembles with MC. We have not touched almost at all at the question of how to actually obtain physical information out of the simulation, except saying how to calculate the total potential energy (which corresponds to the measurable cohesive energy).

Here we describe briefly how to obtain a few of the more common quantities of interest. We only mention quantities which are not explicitly dependent on the velocities in the system, since velocities were not present in any of the schemes presented above.

First we recall that the basic way to obtain an average in MC is to calculate the average over a large number of trials,

$$\mathcal{A}_{\rm MC} = \langle \mathcal{A} \rangle_{\rm ens} \tag{28}$$

where $\langle\rangle$ stands for the average over trials, using the averaging scheme for each ensemble.

Since in all of the Metropolis-like schemes the idea is to produce states with a probability given by the correct distribution, obtaining the average in the code becomes quite simple, just summing up a quantity and dividing by the number of steps in the end.

Obtaining the potential energy is easy, and since the temperature is known in all the schemes

presented above, the total energy E can be obtained using

$$E = \frac{3}{2}NkT + \langle U \rangle$$

The pressure can be obtained using

$$PV = NkT + \langle W \rangle$$

where the virial W is obtained from the forces

$$W = -\frac{1}{3}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ j\neq i}}^{N} \mathbf{r}_{ij}\mathbf{f}_{ij}$$

where $\mathbf{r}_{ij}$ is the distance between atoms i and j and $\mathbf{f}_{ij}$ the force acting between them. For more details, see the atomistic simulations course. As mentioned above, it is also possible to estimate the virial numerically as

$$\frac{\Delta U}{\Delta V}$$

by doing a change ΔV in the system size and calculating the potential energy difference due to this. With a 3-point parabolic approximation this can be quite accurate.

Utilizing the energies, temperatures and pressures several quantities can be calculated with the following basic thermodynamic equalities:

$$\text{constant volume heat capacity} \quad C_V = \left(\frac{\partial E}{\partial T}\right)_V \tag{29}$$

$$\text{constant pressure heat capacity} \quad C_P = \left(\frac{\partial H}{\partial T}\right)_P \tag{30}$$

$$\text{Thermal expansion coefficient} \quad \alpha_P = \frac{1}{V}\left(\frac{\partial V}{\partial T}\right)_P \tag{31}$$

$$\text{Isothermal compressibility} \quad \beta_T = -\frac{1}{V}\left(\frac{\partial V}{\partial P}\right)_T \tag{32}$$

$$\text{Thermal pressure coefficient} \quad \gamma_V = \left(\frac{\partial P}{\partial T}\right)_V \tag{33}$$

$$\text{Bulk modulus} \quad B = -V\left(\frac{\partial P}{\partial V}\right)_T \tag{34}$$

The subindex says which quantity should remain constant in the simulations.

Moreover, since $\alpha_P = \beta_T \gamma_V$ it is enough to get one of the three quantities $\alpha_P, \beta_T, \gamma_V$. Similarly B and β_T are related by $B = 1/\beta_T$.

So how does one obtain the partial derivatives in practice? The idea is to assume the curve is reasonably smooth in some interval of the derived variable, then simulate a few points on the curve, fit a function to it and thus obtain the derivative.

E.g. if we would to determine C_V, we would determine $E(T)$ by simulating at different temperatures, keeping the volume constant, then fit something to the $E(T)$ data (cubic splines may well be good enough), and obtain $C_V(T)$ from derivating the $E(T)$ curve.

It is also possible to obtain some of these by looking at the fluctuations from the average (as already mentioned in the Ising model chapter). For instance C_V can be derived from a single simulation by looking at the fluctuations of the momentaneous enthalpy in the NVT ensemble:

$$\langle \Delta H^2 \rangle_{NVT} = kT^2 C_V$$

Finally, we note that it is possible to derive the absolute value of the entropy S from MC and MD simulations, using so called thermodynamic integration. See [Frenkel, Ladd, J. Chem. Phys. 81 (1984) 3188]. Once the entropy is known, one can determine the free energy

$$F = E - TS$$

explicitly. This is useful e.g. in the study of melting: the melting temperature is the temperature at which $F(T)$ for the solid and liquid phase cross each other.