MOLECULAR SIMULATIONS:

1. Monte Carlo simulations
2. Molecular Dynamics Simulations

1

The Monte Carlo Method

Aim: to compute thermal averages of equilibrium systems.

$$\langle A \rangle = \frac{\sum_i \exp(-\epsilon_i / k_B T) A_i}{\sum_i \exp(-\epsilon_i / k_B T)}$$

Where i labels all eigenstates of the system, and

$$A_i = \langle i | A | i \rangle$$

2

Classical limit: replace the SUM over quantum states by an INTEGRAL of phase space

$$\langle A \rangle = \frac{\int \mathrm{d}\mathbf{p}^N \mathrm{d}\mathbf{r}^N \; A(\mathbf{p}^N, \mathbf{r}^N) \exp[-\beta \mathcal{H}(\mathbf{p}^N, \mathbf{r}^N)]}{\int \mathrm{d}\mathbf{p}^N \mathrm{d}\mathbf{r}^N \; \exp[-\beta \mathcal{H}(\mathbf{p}^N, \mathbf{r}^N)]}$$

Where H is the Hamiltonian of the system and β=1/kT

In replacing the sum by an integral, we have attributed a “volume” h^{3N} to every quantum state

3

Problem:

We cannot compute the sum over all quantum states (because there are so many)

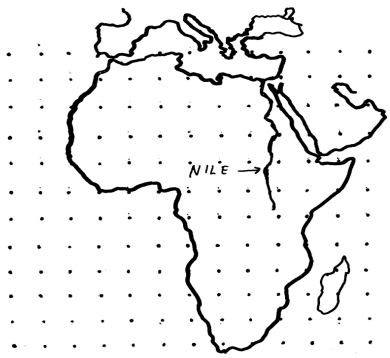
And we cannot compute the classical integral either (except the integration over momenta).

Consider “normal” numerical integration

100 particles, 3 dimensions, 10 points in every direction.

Requires 10^{300} points for a very poor estimate of the integral…

4



Similar problem (but much less serious):

Measure the depth of the Nile by quadrature…

5

We wish to perform a RANDOM WALK in configuration space, such that

The number of times that each point is visited, is proportional to its Boltzmann weight.

$$n(\mathbf{r}^N) = c\exp[-\beta\mathcal{U}(\mathbf{r}^N)]$$

6

Then

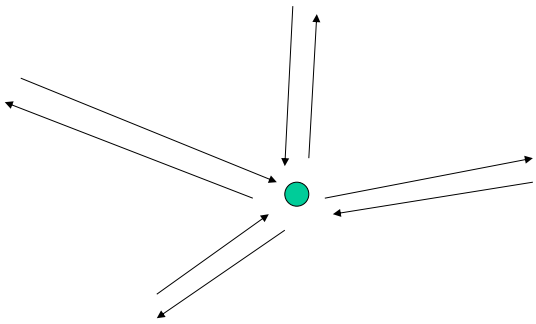
$$\langle A \rangle \approx \frac{1}{L} \sum_{i=1}^{L} n_i \, A(\mathbf{r}_i^N).$$

How do we achieve that?

7

Whatever our rule is for moving from one point to another, it should not destroy the equilibrium distribution.

That is: in equilibrium we must have



8

$$\mathcal{N}(o)\sum_n \pi(o \to n) = \sum_n \mathcal{N}(n)\pi(n \to o)$$

Stronger condition:

$$\mathcal{N}(o)\pi(o \to n) = \mathcal{N}(n)\pi(n \to o).$$

For every pair {n,o}.

Detailed Balance

9

Now we construct the transition probabilities

$$\pi(o \to n) = \alpha(o \to n) \times \mathsf{acc}(o \to n)$$

Then, detailed balance implies that:

$$\mathcal{N}(o)\alpha(o \to n) \times \mathsf{acc}(o \to n)$$

$$=$$

$$\mathcal{N}(n)\alpha(n \to o) \times \mathsf{acc}(n \to o)$$

10

Often, we choose

$$\alpha(o \to n) = \alpha(n \to o)$$

Then it follows that

$$\mathcal{N}(o) \times \text{acc}(o \to n)$$

$$=$$

$$\mathcal{N}(n) \times \text{acc}(n \to o)$$

11

$$\frac{\text{acc}(o \to n)}{\text{acc}(n \to o)} = \frac{\mathcal{N}(n)}{\mathcal{N}(o)} = \exp\{-\beta[\mathcal{U}(n) - \mathcal{U}(o)]\}$$

Metropolis, Rosenbluth,Rosenbluth,

Teller and Teller choice:

$$\text{acc}(o \to n) = \min\left(1, \exp\{-\beta[\mathcal{U}(\mathbf{r}'^N) - \mathcal{U}(\mathbf{r}^N)]\}\right)$$

12

In order to satisfy detailed balance, the attempt frequency for forward and backward moves must be the same.

So: if a trial move is “try to displace a particle by δx”, then the reverse trial move “try to displace a particle by -δx” must be equally likely (or, at least, possible)

Apart from that: how should we choose trial moves? There is much freedom.

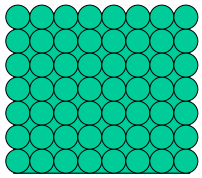
Typical choice :
δx uniform between +Δ and –Δ, with Δ “not too small” and “not too large”

13

Practical issues:

1. Boundary conditions
2. Time-saving devices

14



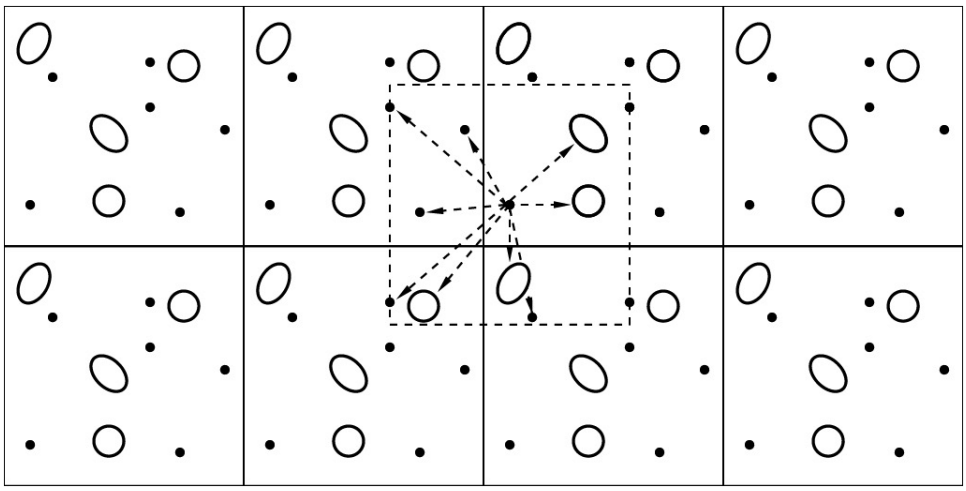
In small systems, boundary effects are always large.

1000 atoms in a simple cubic crystal – 488 boundary atoms.

1000000 atoms in a simple cubic crystal – still 6% boundary atoms…

15

“Solution” : Periodic boundary conditions



16

The most time-consuming part of any simulation is the evaulation of all the interactions between the molecules.

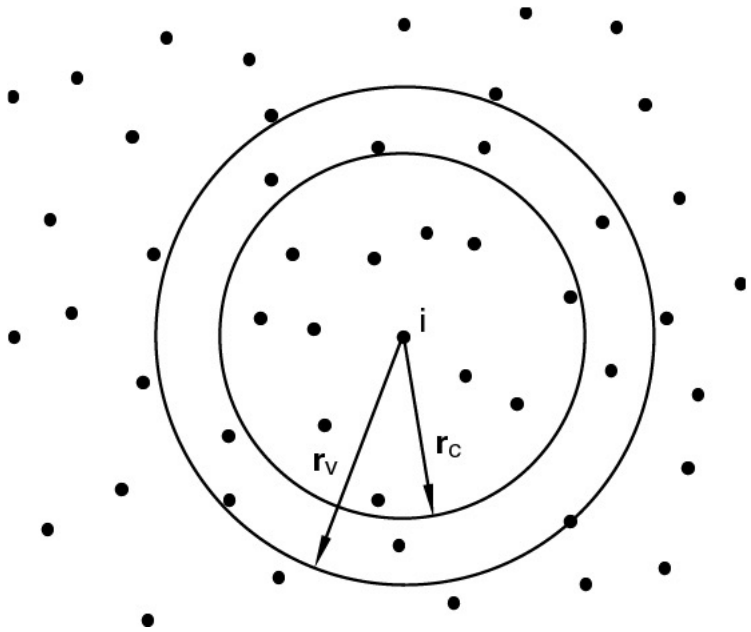
In general: $N(N-1)/2 = O(N^2)$

But often, intermolecular forces have a short range:

Therefore, we do not have to consider interactions with far-away atoms…

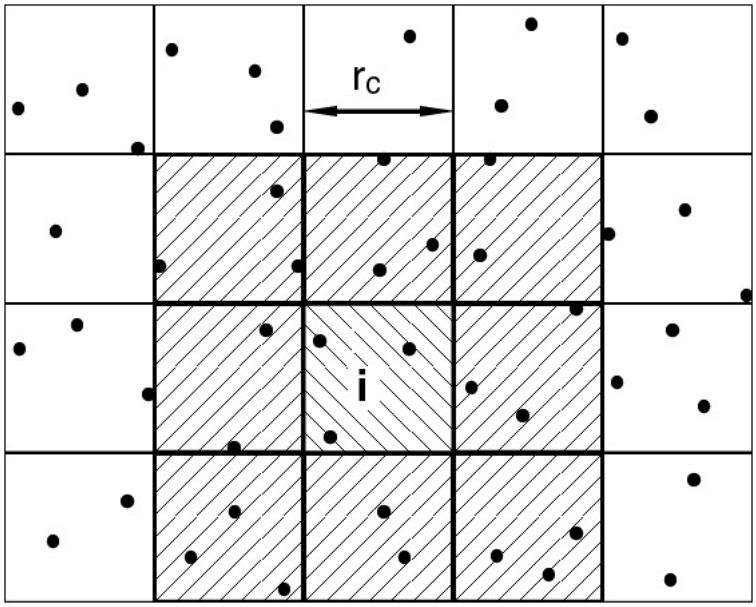
17

Verlet list



18

Link list



19

NOTE:

Long-ranged forces require special techniques.

1. Coulomb interaction (1/r in 3D)
2. Dipolar interaction ($1/r^3$ in 3D)

…and, in a different context:

1. Interactions through elastic stresses (1/r in 3D)
2. Hydrodynamic interactions (1/r in 3D)
3. …

20

Monte Carlo myth:
Optimal acceptance rate of trial moves is 50%...

Quotes from the literature:

100 blocks for error analysis. We have optimized the Monte Carlo move maximum step sizes so that 50% of the trials are accepted.

(i) A trial value of V is selected as $\ln V_{new} = \ln V_{old} + q(\delta \ln V)_{max}$, where q is a random value selected uniformly on $(-1,1)$ and $(\delta \ln V)_{max}$ is tuned to achieve a target acceptance rate, typically 50%.

if d_{max} is too large then most of the trail moves is rejected and there is little movement through the phase space.

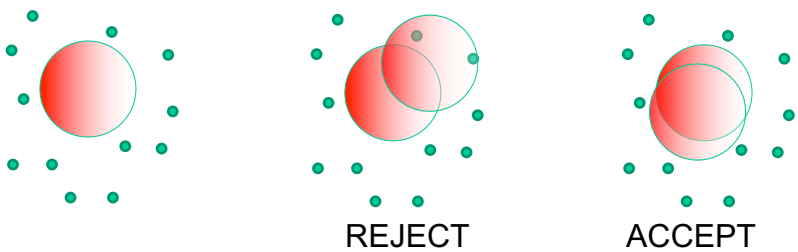
Usually d_{max} is chosen so that ~ 50% of the trail moves is accepted.

21

Optimal acceptance should yield `best sampling' per CPU second…

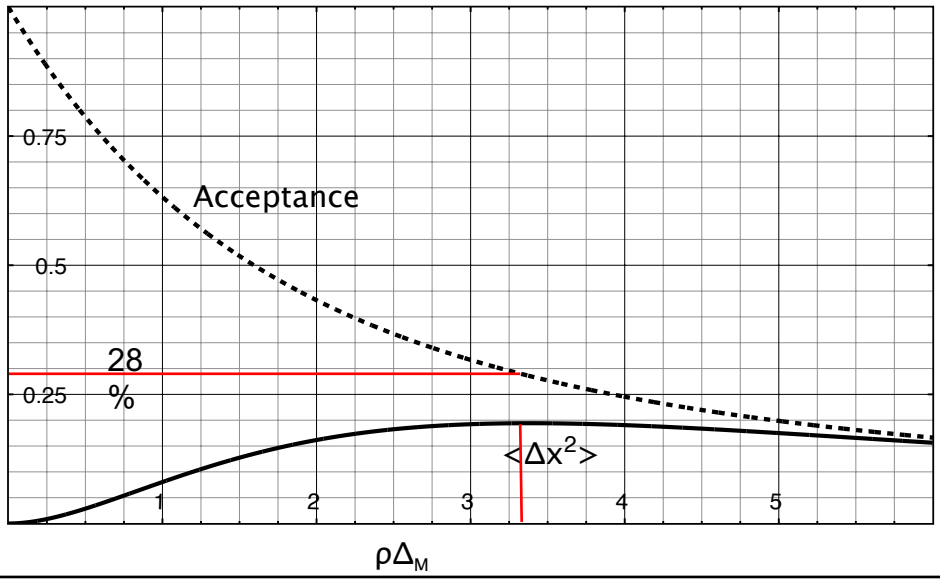
Problem: there is no unique definition of best sampling.

Simple example: Brownian particle in ideal gas.



22

In this case `optimal' means: fastest diffusion of Brownian particle.



23

Beyond constant-NVT Monte Carlo

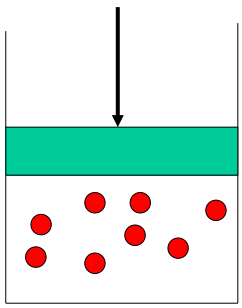
24

Simulations at constant PRESSURE (constant NPT-MC)

or at constant CHEMICAL POTENTIAL (constant μVT-MC)

25

Constant NPT Monte Carlo



We fix the

» Temperature (T)

» Pressure (P)

» Number of particles (N)

26

Intermezzo: Scaled coordinates

We can rewrite the constant NVT Partition function

$$Q_{NVT} = \frac{1}{\Lambda^{3N} N!} \int \mathrm{d}\mathrm{r}^N \exp\left[-\beta U\left(\mathrm{r}^N\right)\right].$$

By introducing "scaled" coordinates

$$\mathbf{s}_i = \mathbf{r}_i / L$$

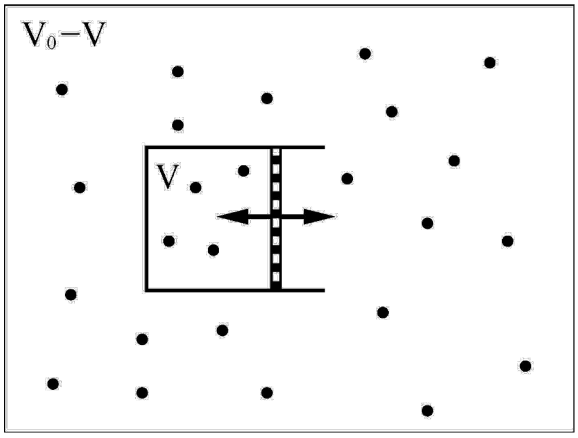
This gives for the partition function

$$Q_{NVT} = \frac{L^{3N}}{\Lambda^{3N} N!} \int \mathrm{d}\mathrm{s}^N \exp\left[-\beta U\left(\mathrm{s}^N; L\right)\right]$$

$$= \frac{V^N}{\Lambda^{3N} N!} \int \mathrm{d}\mathrm{s}^N \exp\left[-\beta U\left(\mathrm{s}^N; L\right)\right]$$

27

CONSTANT NPT:



N in volume V

M in volume V_0-V

28

Consider a system with volume V in a (large) reservoir with volume V-V_r. The partition function of the combined system is:

$$Q(N, M_r, V, V_r - V) = \exp\left(-\beta[F_r(V_r - V) + F_s(V)]\right)$$
$$= \exp[-\beta F_r(V_r)] \exp[-\beta PV] \exp[-\beta F_s(V)]$$
$$\sim \exp[-\beta PV] Q(N, V, T)$$

Next, we allow the volume of the small system V to take all possible values less than V_r

The probability that the small system has a volume V is then:

$$\mathcal{P}(V) = \frac{\exp[-\beta PV] Q(N,V,T)}{\int dV' \ \exp[-\beta PV'] Q(N,V',T)}$$

$$\equiv \frac{\exp[-\beta PV] Q(N,V,T)}{Q(N,P,T)}$$

29

Now, ignoring unimportant constants, we have:

$$Q(N, V, T) \sim V^N \int d\mathbf{s}^n \exp[-\beta U(\mathbf{s}^N)]$$

The probability that the small system has a volume V and scaled coordinates $\mathbf{s}^N$ is then

$$\mathcal{P}(V, \mathbf{s}^N) \sim \exp[-\beta PV] V^N \exp[-\beta \mathcal{U}(\mathbf{s}^N; L)]$$
$$= \exp[-\beta \left(PV - \beta^{-1} N \ln V + \mathcal{U}(\mathbf{s}^N; L)\right)]$$

This looks like a Boltzmann weight!

We can now do normal Monte Carlo sampling on all coordinates …**and** V

30

Constant-NPT MC

$$\mathcal{P}(V, \mathbf{s}^N) \sim \exp[-\beta\left(PV - \beta^{-1} N \ln V + \mathcal{U}(\mathbf{s}^N; L)\right)]$$

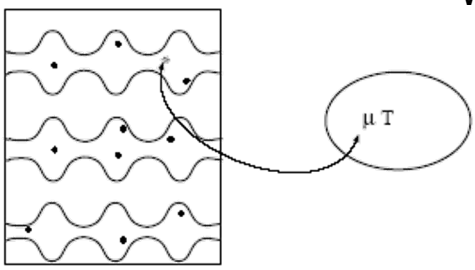
Volume "moves" should satisfy detailed balance

$$\frac{\mathrm{acc}(V_o \to V_n)}{\mathrm{acc}(V_n \to V_o)} =$$

$$\exp[-\beta\left(P(V_n - V_o) - \beta^{-1} N \ln(V_n/V_o) + \mathcal{U}(\mathbf{s}^N; L_n) - \mathcal{U}(\mathbf{s}^N; L_o)\right)]$$

31

Similarly: **constant μVT** (Grand-canonical ensemble)



We impose:

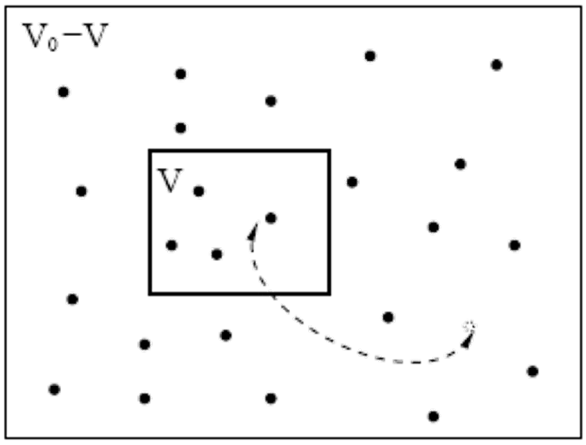
- Temperature (T)
- Chemical potential (μ)
- Volume (V)

- But **NOT** pressure

32

32

System in large reservoir with chemical potential μ



33

33

Similar to NPT, we can work out the acceptance rule for the addition/removal of a particle

$$\frac{\mathrm{acc}(N \to N+1)}{\mathrm{acc}(N+1 \to N)} = \frac{\exp(\beta\mu)V}{\Lambda^3(N+1)} \exp\left[-\beta\Delta\mathcal{U}(\mathbf{s}^N; \mathbf{s}_N + 1)\right]$$

$\Delta\mathcal{U}(\mathbf{s}^N; \mathbf{s}_N + 1)$ denotes the change in the potential energy when **one** particle is added at a (random) position $\mathbf{s}_{N+1}$ to a system of **N** particles at (scaled) coordinates $\mathbf{s}^N$

34

The shaky foundations of…

Molecular Dynamics

35

Molecular Dynamics

The Basis:

$F_i = m_i a_i$

$i = 1, 2, \ldots, N$

36

N-body problem. Can only be solved numerically (except in very special cases)

How?

$$X(t+\Delta t) = X(t) + \dot{X}(t)\Delta t + \frac{1}{2!}\ddot{X}(t)\Delta t^2 + \frac{1}{3!}\dddot{X}(t)\Delta t^3 + \ldots$$

...at least, in principle.

37

Naive approach: truncate Taylor expansion.

$$\cancel{X(t+\Delta t) \approx X(t) + \dot{X}(t)\Delta t + \frac{1}{2!}\ddot{X}(t)\Delta t^2}$$

ABSOLUTELY FORBIDDEN!

38

The naive "forward Euler" algorithm

- is not time reversible
- does not conserve volume in phase space
- suffers from energy drift

Better approach: "Verlet" algorithm

39

$$X(t+\Delta t) = X(t) + \dot{X}(t)\Delta t + \frac{1}{2!}\ddot{X}(t)\Delta t^2 + \frac{1}{3!}\dddot{X}(t)\Delta t^3 + \frac{1}{4!}\ddddot{X}(t)\Delta t^4 + \ldots$$

$$X(t-\Delta t) = X(t) - \dot{X}(t)\Delta t + \frac{1}{2!}\ddot{X}(t)\Delta t^2 - \frac{1}{3!}\dddot{X}(t)\Delta t^3 + \frac{1}{4!}\ddddot{X}(t)\Delta t^4 + \ldots$$

+

$$X(t+\Delta t) + X(t-\Delta t) = 2X(t) + \ddot{X}(t)\Delta t^2 + O(\Delta t^4)$$

or

$$\boxed{X(t+\Delta t) \approx 2X(t) - X(t-\Delta t) + \ddot{X}(t)\Delta t^2}$$

Verlet algorithm

40

Verlet algorithm

- is time reversible
- does conserve volume in phase space (see later)
- (is “symplectic”)
- does not suffer from energy drift

41

...but is it a good algorithm?

i.e. does it predict the time evolution of the system correctly???

42

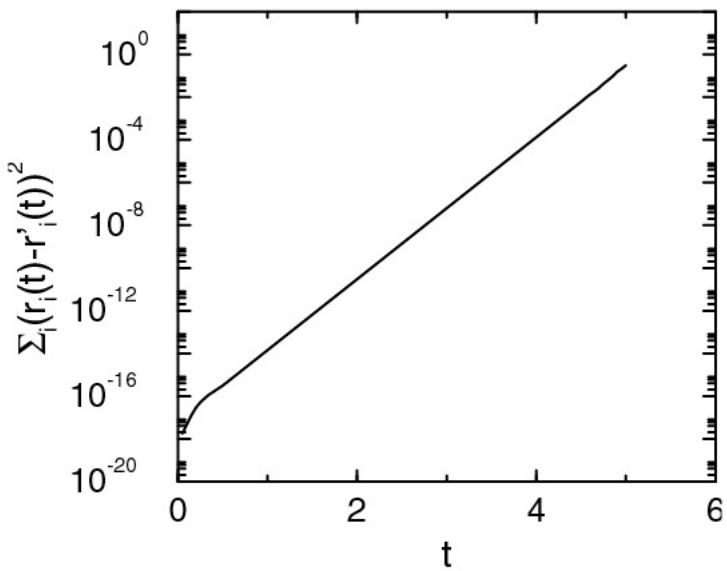
Dynamics of “well-behaved” classical many-body system is chaotic.

Consequence:

Trajectories that differ very slightly in their initial conditions DIVERGE EXPONENTIALLY (“Lyapunov instability”)

43

The Lyapunov disaster in action...



44

Any small error in the numerical integration of the equations of motion, will blow up exponentially....

always...

...and for any algorithm!!

45

SO:

Why should anyone believe Molecular Dynamics simulations ???

46

Answers:

1. In fact, one should not...

exit Molecular Dynamics...

47

Answers:

1. In fact, one should not...
2. Good MD algorithms (e.g. Verlet) can also be considered as good Monte Carlo algorithms – they therefore yield reliable STATIC properties (“Hybrid Monte Carlo”)

What is the point of simulating dynamics, if we cannot trust the resulting time-evolution???

48

Answers:

1. In fact, one should not...
2. Good MD algorithms (e.g. Verlet) can also be considered as good Monte Carlo algorithms – they therefore yield reliable STATIC properties ("Hybrid Monte Carlo")
3. All is well (probably), because of...

The Shadow Theorem....

49

For any realistic many-body system, the shadow theorem is merely a hypothesis.

It states that (my words):

Good algorithms generate numerical trajectories that are "close to" a REAL trajectory of the many-body system.

50

Question:

Does the Verlet algorithm indeed generate “shadow” trajectories?

51

Take a different look at the problem.

Do not discretize NEWTON’s equation of motion...

...but discretize the ACTION

52

Intermezzo:

Classical mechanics – the Lagrangian approach.

Newton: $\mathbf{F_i} = m_i \ddot{\mathbf{r}}_i$

Lagrange: Consider a system that is at a point r_0 at time t=0 and at point r_t at time t=t, then:

The system follows a trajectory r(t) such that:

53

"Action"

$$S \equiv \int_0^t dt' \mathcal{L}(\mathbf{r}(t))$$ is an extremum.

"Lagrangian"

Where the Lagrangian is defined as:

$$\mathcal{L}(\mathbf{r}(t)) = T_{kinetic} - U_{pot}(\mathbf{r})$$

For example, if we use cartesian coordinates:

$$\mathcal{L}(\mathbf{r}(t)) = \sum_{i=1}^{N} \frac{1}{2} m_i \dot{\mathbf{r}}_i^2 - U(\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_N)$$

54

What does this mean?

Consider the "true" path R(t), with $R(0)=r_0$ and$R(t)=r_t$.

Now, consider a path close to the true path:

$$\mathbf{r}(t') = \mathbf{R}(t') + \delta\mathbf{r}(t')$$

Then the action S is an extremum if

$$\frac{\delta S}{\delta \mathbf{r}(t')} = 0 \quad , (\forall t')$$

(what does this equation mean??)

55

$$S_{\text{continuous}} = \int_{t_0}^{t_1} dt\, L(t)$$

Discretized version

$$S_{\text{discrete}} = \Delta t \sum_{i=0}^{i_{max}} L(t_i)$$

56

$$L(t_i) = T(t_i) - U(t_i)$$

e.g. for one coordinate in one dimension

$$L(t_i)\Delta t = \frac{1}{2} m\Delta t \frac{(X_{i+1} - X_i)^2}{\Delta t^2} - U(X_i)\Delta t$$

and hence the discretized action is

57

$$S_{discrete} = \sum_{i=1}^{i_{max}} \left(\frac{m(X_{i+1} - X_i)^2}{2\Delta t} - U(X_i)\Delta t \right)$$

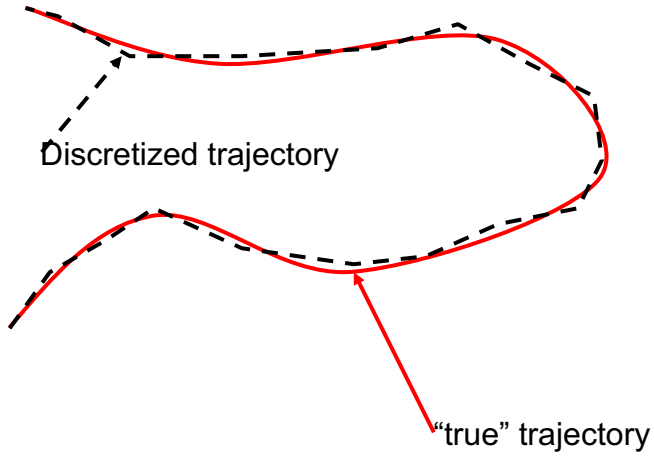
Now do the standard thing:

Find the extremum for small variations in the path, i.e. for small variations in all X_i.

$$\frac{\partial S_{discrete}}{\partial X_i} = 0 \qquad (\forall i)$$

58

This will generate a discretized trajectory that starts at time t_0 at X_0 , and ends at time t_1 at X_1.



59

$$\frac{\partial S_{discrete}}{\partial X_i} = \frac{\partial}{\partial X_i} \sum_{i=1}^{i_{max}} \left(\frac{m(X_{i+1} - X_i)^2}{2\Delta t} - U(X_i)\Delta t \right)$$

$$\frac{\partial S_{discrete}}{\partial X_i} = \frac{-m(X_{i+1} - X_i) + m(X_i - X_{i-1})}{\Delta t} - \Delta t \frac{\partial U(X_i)}{\partial X_i}$$

And hence:

$$0 = \frac{m}{\Delta t} \left(2X_i - X_{i+1} - X_{i-1} - \frac{\Delta t^2}{m} \frac{\partial U(X_i)}{\partial X_i} \right)$$

60

$$0 = \left(2X_i - X_{i+1} - X_{i-1} - \frac{\Delta t^2}{m} \frac{\partial U(X_i)}{\partial X_i} \right)$$

REWRITE AS:

$$X_{i+1} = 2X_i - X_{i-1} + \frac{\Delta t^2}{m} F(X_i)$$

VERLET!!!

61

The Verlet algorithm generates trajectory that satisfies the boundary conditions of a REAL trajectory – both at the beginning and at the endpoint.

62

If we are interested in statistical information about the dynamics (e.g. time-correlation functions, transport coefficients, power spectra...)

...then a “good” MD algorithm (e.g. Verlet) is fine.

But not if we want to predict with high accuracy the future on the basis of the initial positions/momenta (e.g. a satellite orbit)

63

VELOCITY VERLET ALGORITHM

$$\mathbf{r}(t+\Delta t) \approx \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \frac{\Delta t^2}{2m}\mathbf{f}(t)$$

$$\mathbf{v}(t+\Delta t) \approx \mathbf{v}(t) + \frac{\Delta t}{2m}\left[\mathbf{f}(t+\Delta t) + \mathbf{f}(t)\right]$$

...IS EQUIVALENT TO POSITION VERLET ALGORITHM

$$\mathbf{r}(t+\Delta t) = 2\mathbf{r}(t) - \mathbf{r}(t-\Delta t) + \frac{\mathbf{F}(t)}{m}\Delta t^2$$

64

First, use: $\mathbf{r}(t+\Delta t) = \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \frac{\Delta t^2}{2m}\mathbf{f}(t)$

and $\mathbf{r}(t) = \mathbf{r}(t-\Delta t) + \mathbf{v}(t-\Delta t)\Delta t + \frac{\Delta t^2}{2m}\mathbf{f}(t-\Delta t)$

The last line can be rearranged to yield:

$$r(t-\Delta t) = r(t) - v(t-\Delta t)\Delta t - \frac{f(t-\Delta t)}{2m}\Delta t^2.$$

65

Now add:

$$r(t+\Delta t) = r(t) + v(t)\Delta t + \frac{f(t)}{2m}\Delta t^2$$

$$r(t-\Delta t) = r(t) - v(t-\Delta t)\Delta t - \frac{f(t-\Delta t)}{2m}\Delta t^2.$$

+

$$r(t+\Delta t)+r(t-\Delta) = 2r(t)+[v(t)-v(t-\Delta)]\Delta t+\frac{f(t)-f(t-\Delta)}{2m}\Delta t^2.$$

But: $\mathbf{v}(t+\Delta t) \approx \mathbf{v}(t) + \frac{\Delta t}{2m}\left[\mathbf{f}(t+\Delta t) + \mathbf{f}(t)\right]$

Hence $\mathbf{v}(t) - \mathbf{v}(t-\Delta t) = \frac{\Delta t}{2m}\left[\mathbf{f}(t) + \mathbf{f}(t-\Delta t)\right]$

66

Thus:

$$r(t+\Delta t)+r(t-\Delta t) = 2r(t)+\frac{f(t)+f(t-\Delta t)}{2m}\Delta t^2+\frac{f(t)-f(t-\Delta t)}{2m}\Delta t^2.$$

Which yields:

$$r(t+\Delta t)+r(t-\Delta t) = 2r(t)+\frac{f(t)}{m}\Delta t^2$$

Position Verlet !

67

We can interpret the Velocity Verlet algorithm:

$$\mathbf{r}(t+\Delta t) \approx \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \frac{\Delta t^2}{2m}\mathbf{f}(t)$$

$$\mathbf{v}(t+\Delta t) \approx \mathbf{v}(t) + \frac{\Delta t}{2m}\left[\mathbf{f}(t+\Delta t) + \mathbf{f}(t)\right]$$

as follows:

$$\mathbf{v}(t+\Delta t/2) \approx \mathbf{v}(t) + \frac{\Delta t}{2m}\left[\mathbf{f}(t)\right]$$ Change ONLY v

$$\mathbf{r}(t+\Delta t) \approx \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \frac{\Delta t^2}{2m}\mathbf{f}(t)$$ Change ONLY r

$$\mathbf{v}(t+\Delta t) \approx \mathbf{v}(t+\Delta t/2) + \frac{\Delta t}{2m}\left[\mathbf{f}(t+\Delta t)\right]$$ Change ONLY v

68

Important property of the Velocity Verlet algorithm:
it is "area preserving"

because the successive coordinate transformations in either **r** or **p** all have a Jacobian determinant equal to one:

$$\mathbf{p}(t+\Delta t/2)=\mathbf{p}(t)+\frac{\Delta t}{2}\mathbf{F}(\mathbf{r})$$
$$\mathbf{r}(t)=\mathbf{r}(t)$$
$$J_1=\det\begin{vmatrix}1 & \frac{\Delta t}{2}\frac{\partial\mathbf{F}(\mathbf{r})}{\partial\mathbf{r}}\\ 0 & 1\end{vmatrix}=1$$

$$\mathbf{p}(t+\Delta t/2)=\mathbf{p}(t+\Delta t/2)$$
$$\mathbf{r}(t+\Delta t)=\mathbf{r}(t)+\frac{\Delta t}{m}\mathbf{p}(t+\Delta t/2)$$
$$J_2=\det\begin{vmatrix}1 & 0\\ \Delta t/m & 1\end{vmatrix}=1$$

$$\mathbf{p}(t+\Delta t)=\mathbf{p}(t+\Delta t/2)+\frac{\Delta t}{2}\mathbf{F}(\mathbf{r}(t))$$
$$\mathbf{r}(t)=\mathbf{r}(t)$$
$$J_3=\det\begin{vmatrix}1 & \frac{\Delta t}{2}\frac{\partial\mathbf{F}(\mathbf{r})}{\partial\mathbf{r}}\\ 0 & 1\end{vmatrix}=1$$

69

Liouville formulation

Consider an arbitrary function $f\left(\mathbf{r}^N,\mathbf{p}^N\right)\equiv f\left(\Gamma\right)$ of the coordinates and momenta of the N particles (together denoted by $\mathbf{\Gamma}$).
As the system evolves in time (due to the motion of the particles), then f evolves too:

$$\dot{f}\left(\Gamma\right)=\dot{\Gamma}\left(\frac{\partial f(\Gamma)}{\partial\Gamma}\right)\equiv iLf(\Gamma)$$

iL is called the "Liouville operator". It is shorthand for

$$iL\equiv\dot{\Gamma}\frac{\partial}{\partial\Gamma}$$

70

The we can perform a Taylor expansions of f(**Γ**(t))

$$f(t) = f(0) + iLtf(0) + \frac{(iLt)^2}{2!}f(0) + \frac{(iLt)^3}{3!}f(0) + \cdots$$

We write this as symbolically as:

$$f(t) = e^{iLt}f(0)$$

However, that does not help much, as we cannot evaluate this

Things become simpler if we look at the two parts of the Liouville operator separately:

$$iL \equiv iL_r + iL_p = \dot{r}\frac{\partial}{\partial r} + \dot{p}\frac{\partial}{\partial p}$$

71

Now, consider the effect of iL_r and iL_p separately:

First iL_r:

$$f(t) = \exp(iL_r t)f(0)$$

$$f(t) = \exp(iL_r t)f(0) = \sum_{n=0}^{\infty} \frac{(\dot{\mathbf{r}}(0))^n}{n!}\frac{\partial^n}{\partial \mathbf{r}^{\mathbf{n}}}f(0)$$

$$= f\left(\mathbf{p}(0), \mathbf{r}(0) + \dot{\mathbf{r}}(0)t\right)$$

This case is easy to compute: the effect of the operator is to shift the positions, but not the momenta.

72

Similarly, if we consider $iL_p \equiv \dot{\mathbf{p}}\frac{\partial}{\partial \mathbf{p}}$

Then we would have:

$$f(t) = \exp(iL_p t) f(0)$$

$$= f\left(\mathbf{p}(0) + \dot{\mathbf{p}}(0)t, \mathbf{r}(0)\right)$$

This case is also easy to compute: the effect of the operator iL_p is to shift the momenta, but not the positions.

$$\mathbf{p}(\mathbf{0}) \rightarrow \mathbf{p}(\mathbf{0}) + \dot{\mathbf{p}}(\mathbf{0})\mathbf{t}$$

73

But what we actually must compute is:

$$e^{(iL_r + iL_p)\Delta t} f\left(\mathbf{p}^N(0), \mathbf{r}^N(0)\right)$$

BUT THIS IS NOT **EQUAL TO:**

$$e^{iL_r \Delta t} e^{iL_p \Delta t} f\left(\mathbf{p}^N(0), \mathbf{r}^N(0)\right)$$

because $(\mathrm{i}L_r)(\mathrm{i}L_p) \neq (\mathrm{i}L_p)(\mathrm{i}L_r)$

74

Solution: Trotter identity for non-commuting operators

$$e^{A+B} \neq e^A e^B$$

but

$$e^{A+B} = \lim_{P\to\infty} (e^{A/2P} e^{B/P} e^{A/2P})^P$$

We will assume that for large but finite P, we have

$$e^{A+B} \approx (e^{A/2P} e^{B/P} e^{A/2P})^P$$

75

Now we define:

$$\frac{A}{P} = \frac{\mathrm{i}L_p t}{P} \qquad \frac{B}{P} = \frac{\mathrm{i}L_r t}{P} \qquad \Delta t = \frac{t}{P}$$

then

$$f\left(\mathbf{p}^N(\Delta t), \mathbf{r}^N(\Delta t)\right) \approx e^{iL_p\Delta t/2} e^{iL_r\Delta t} e^{iL_p\Delta t/2} f\left(\mathbf{p}^N(0), \mathbf{r}^N(0)\right)$$

What is the effect of applying the operator

$$e^{iL_p\Delta t/2} e^{iL_r\Delta t} e^{iL_p\Delta t/2}$$

76

Step 1:

$$e^{iL_p\Delta t/2} f\left(\mathbf{p}^N(0), \mathbf{r}^N(0)\right) = f\left(\mathbf{p}^N(0) + F(0)\Delta t/2, \mathbf{r}^N(0)\right) = f\left(\mathbf{p}^N(\Delta t/2), \mathbf{r}^N(0)\right)$$

Step 2:

$$e^{iL_r\Delta t} f\left(\mathbf{p}^N(\Delta t/2), \mathbf{r}^N(0)\right) = f\left(\mathbf{p}^N(\Delta t/2), \mathbf{r}^N(0) + \mathbf{p}^N(\Delta t/2)/m\Delta t\right)$$
$$= f\left(\mathbf{p}^N(\Delta t/2), \mathbf{r}^N(\Delta t)\right)$$

Step 3:

$$e^{iL_p\Delta t/2} f\left(\mathbf{p}^N(\Delta t/2), \mathbf{r}^N(\Delta t)\right) = f\left(\mathbf{p}^N(\Delta t/2) + \mathbf{F}(\Delta t)\Delta t/2, \mathbf{r}^N(\Delta t)\right)$$
$$= f\left(\mathbf{p}^N(\Delta t), \mathbf{r}^N(\Delta t)\right)$$

77

Net result:

$$\mathbf{r}(0) \to \mathbf{r}(0) + \Delta t \frac{\mathbf{p}(\frac{\Delta t}{2})}{m} = \mathbf{r}(0) + \Delta t \frac{\mathbf{p}(0)}{m} + \frac{(\Delta t)^2}{2m}\mathbf{F}(0)$$

$$\mathbf{p}(0) \to \mathbf{p}(0) + \Delta t/2\left(\mathbf{F}(0) + \mathbf{F}(\Delta t)\right)$$

But wait a minute…. **This is the Velocity Verlet algorithm !**

Comment:
Actually, the algorithm should be called the **momentum-Verlet algorithm.**
I will come back to this point later.

78

In summary: the Verlet algorithm is:

1. Time reversible (✓)
2. Area preserving (✓)
3. Generates trajectories that are (probably) close to a true trajectory (✓)

You may use other algorithms, but only if they share at least these properties with Verlet.

79

Stigler's law of Epynomy:

No scientific discovery is named after its original discoverer

As Stigler noted, `Stigler's law' was discovered by Robert Merton.

80

Who invented the Verlet algorithm?

Wikipedia:

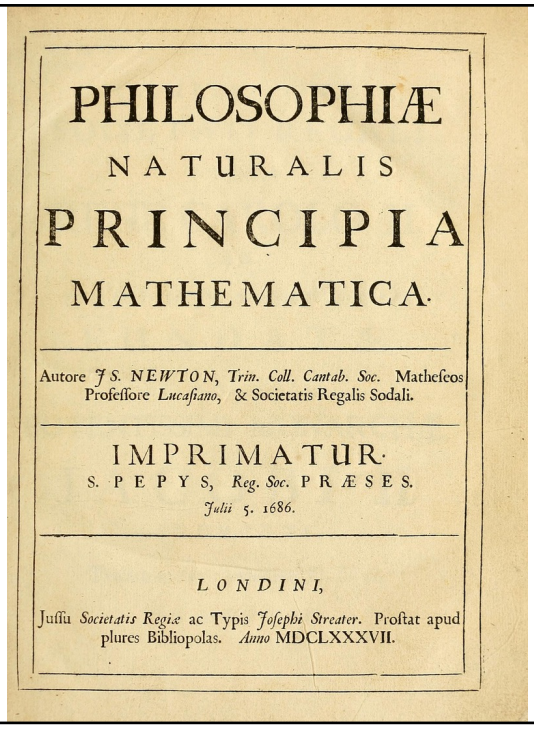
The algorithm was first used in 1791 by Delambre and has been rediscovered many times since then, most recently by Loup Verlet in the 1960s for use in molecular dynamics. It was also used by Cowell and Crommelin in 1909 to compute the orbit of Halley's Comet, and by Carl Størmer in 1907 to study the trajectories of electrical particles in a magnetic field (hence it is also called Störmer's method)…

81

So, should Verlet's algorithm be called Delambre's algorithm?

Not quite…

…there is an earlier publication



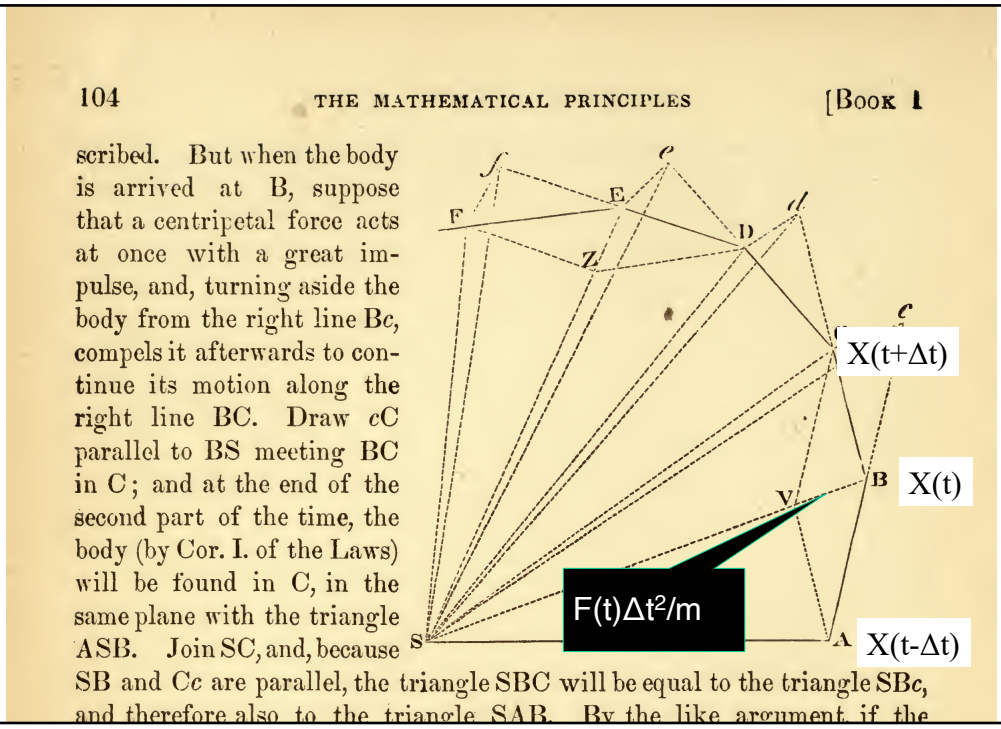
PHILOSOPHIÆ
NATURALIS
PRINCIPIA
MATHEMATICA.

Autore J S. NEWTON, Trin. Coll. Cantab. Soc. Matheſeos Profeſſore Lucaſiano, & Societatis Regalis Sodali.

IMPRIMATUR.
S. PEPYS, Reg. Soc. PRÆSES.
Julii 5. 1686.

LONDINI,
Juſſu Societatis Regiæ ac Typis Joſephi Streater. Proſtat apud plures Bibliopolas. Anno MDCLXXXVII.

82



104 THE MATHEMATICAL PRINCIPLES [Book I

scribed. But when the body is arrived at B, suppose that a centripetal force acts at once with a great impulse, and, turning aside the body from the right line B*c*, compels it afterwards to continue its motion along the right line BC. Draw *c*C parallel to BS meeting BC in C; and at the end of the second part of the time, the body (by Cor. I. of the Laws) will be found in C, in the same plane with the triangle ASB. Join SC, and, because SB and C*c* are parallel, the triangle SBC will be equal to the triangle SB*c*, and therefore also to the triangle SAB. By the like argument, if the

83

Beyond Newtonian dynamics **(Jürgen Horbach)** :

1. Langevin dynamics
2. Brownian dynamics
3. Stokesian dynamics
4. Dissipative particle dynamics
5. MPCD
6. Etc etc

84

Changing the trial moves to increase acceptance probability

Metropolis Monte Carlo:

1. generate trial moves
2. Move if accepted
3. Otherwise, stay where you are

Metropolis, Rosenbluth,Rosenbluth, Teller and Teller:

$$\mathrm{acc}(o \to n) = \min\left(1, \exp\{-\beta[\mathcal{U}(\mathbf{r}'^N) - \mathcal{U}(\mathbf{r}^N)]\}\right)$$

85

Unsatisfactory?

86

Solution of conflict: do not impose

$$\alpha(o \to n) = \alpha(n \to o)$$ then

$$\frac{\mathrm{acc}(o \to n)}{\mathrm{acc}(n \to o)} = \frac{\alpha(n \to o)}{\alpha(o \to n)} \exp\{-\beta[\mathcal{U}(n) - \mathcal{U}(o)]\}.$$

In particular, if we manage to change the trial moves such that

$$\frac{\alpha(n \to o)}{\alpha(o \to n)} = \exp\{-\beta[\mathcal{U}(o) - \mathcal{U}(n)]\}.$$

Then

$$\frac{\mathrm{acc}(o \to n)}{\mathrm{acc}(n \to o)} = 1$$ (100% acceptance)

87

100% acceptance can be achieved in special cases: e.g. Swendsen-Wang/Wolff algorithm (and generalizations)

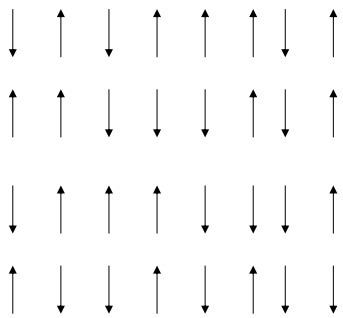
Discrete spin models (Potts, Ising).

Illustration: 2D Ising model:

Parallel nearest neighbor spins:	energy –J
Anti-parallel nearest neighbor spins:	energy +J

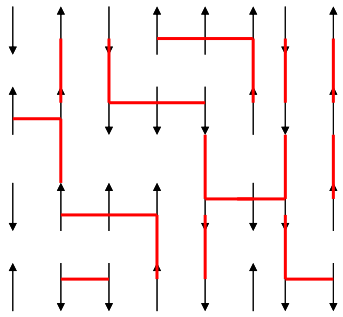
$$U = -J \sum_{i,j} s_i s_j$$

88



Snapshot: some neighbors are parallel, others anti-parallel

89

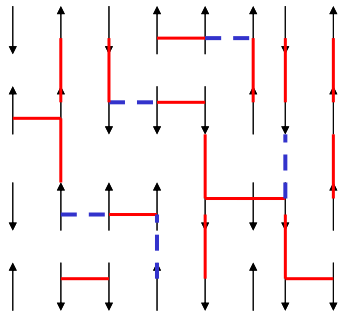


Count number of bonds between parallel neighbors: N_p

Number of bonds between anti-parallel neigbors is: N_a

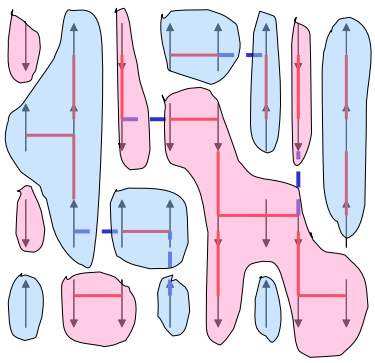
Total energy: $U = (N_a - N_p)\ J$

90



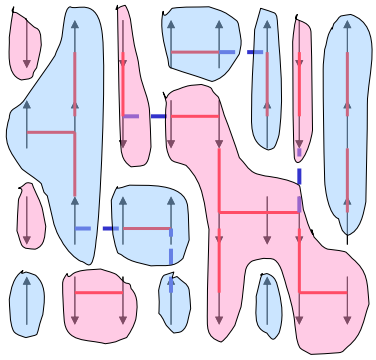
Now, make “bonds”. Bonds only form between parallel neighbors. The probability to have a bond (red line) between parallel neighbors is p (as yet undetermined). With a probability 1-p, parallel neighbors are not connected (blue dashed line).

91



Form clusters of all spins that are connected by bonds. Some clusters are all “spin up” others are all “spin down”. Let us denote the number of clusters by M.

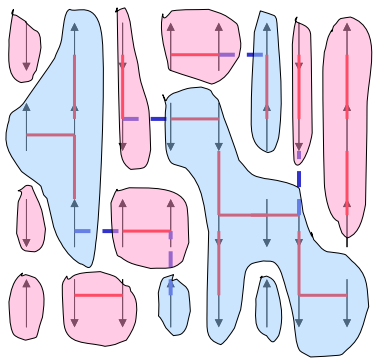
92



The probability to generate a particular cluster structure where there are $\mathbf{n_c}$ bonds between $\mathbf{N_p}$ pairs of parallel neighbors is:

$$P_{gen} = p^{n_c}(1-p)^{N_p - n_c}$$

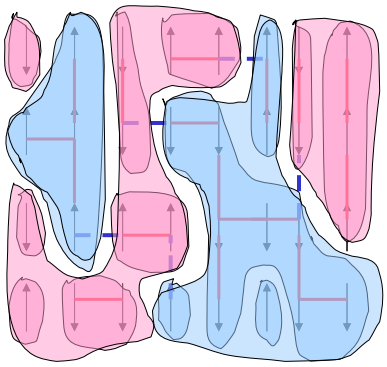
93



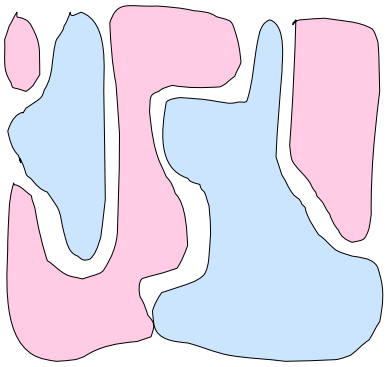
Now randomly flip clusters. This yields a new cluster configuration with probability $P_{(flip)} = (1/2)^M$.

Then reconnect parallel spins

94



95



New cluster structure!

Now make it into a Monte Carlo algorithm:

96

$$P_o P_{clus}(o) P_{flip}(M) P_{acc}(o \to n)$$

$$=$$

$$P_n P_{clus}(n) P_{flip}(M) P_{acc}(n \to o)$$

$$\exp(-\beta U_o) p^{n_c} (1-p)^{N_p(o)-n_c} (1/2)^M P_{acc}(o \to n)$$

$$=$$

$$\exp(-\beta U_n) p^{n_c} (1-p)^{N_p(n)-n_c} (1/2)^M P_{acc}(n \to o)$$

97

$$P_o P_{clus}(o) P_{flip}(M) P_{acc}(o \to n)$$

$$=$$

$$P_n P_{clus}(n) P_{flip}(M) P_{acc}(n \to o)$$

$$\exp(-\beta U_o) p^{n_c} (1-p)^{N_p(o)-n_c} (1/2)^M P_{acc}(o \to n)$$

$$=$$

$$\exp(-\beta U_n) p^{n_c} (1-p)^{N_p(n)-n_c} (1/2)^M P_{acc}(n \to o)$$

98

Moreover, we want 100% acceptance, i.e.:

$P_{acc}(o \to n) = P_{acc}(n \to o) = 1$

$$\exp(-\beta U_o)\cancel{p^{n_c}}(1-p)^{N_p(o)-\cancel{n_c}}(1\cancel{/2})^M P_{acc}(o \to n)$$

$$=$$

$$\exp(-\beta U_n)\cancel{p^{n_c}}(1-p)^{N_p(n)-\cancel{n_c}}(1\cancel{/2})^M P_{acc}(n \to o)$$

99

Hence:

$$\exp(-\beta U_o)(1-p)^{N_p(o)} = \exp(-\beta U_n)(1-p)^{N_p(n)}$$

$$\exp(\beta(U_n - U_o)) = (1-p)^{N_p(n)-N_p(o)}$$

But remember:

$$U_n - U_o = J(N_a(n) - N_p(n)) - J(N_a(o) - N_p(o))$$

or

$$\Delta U = J(\Delta N_a - \Delta N_p)$$

100

But: $\Delta N_a = -\Delta N_p$

and therefore

$$\Delta U = -2J\Delta N_p$$

$$\exp(\beta(U_n - U_o)) = \exp(-2\beta J(N_p(n) - N_p(o)))$$

Combining this with:

$$\exp(\beta(U_n - U_o)) = (1-p)^{N_p(n)-N_p(o)}$$

we obtain:

$$p = 1 - \exp(-2\beta J)$$

101

100% acceptance!!!

102