

# Molecular Dynamics

## Chapter 4

# Molecular Dynamics Simulation

technique for computing the equilibrium and transport properties of a classical many-body system.

classical means that the motion particles obeys the laws of classical mechanics.

Quantum effects are important for example when we consider the translational or rotational motion of light atoms or molecules (He, H<sub>2</sub>, D<sub>2</sub>) or vibrational motion

MD simulates experiments.

In a real experiment:

- prepare a sample
- connect this sample to a measuring instrument
- measure the property during a certain time interval.
- the longer we average, the more accurate

Molecular Dynamics:

- prepare a sample: select a model system consisting of  $N$  particles
- solve Newton's equations of motion until the properties of the system no longer change with time (we equilibrate the system).
- After equilibration, we perform the actual measurement.

To measure an observable quantity in a Molecular Dynamics simulation, we must first of all be able to express this observable as a function of the positions and momenta of the particles in the system.

Example: definition of the temperature using equipartition of energy.

Average kinetic energy per degree of freedom:

$$\langle \frac{1}{2}mv^2 \rangle = \frac{1}{2}k_B T$$

we use this equation as an operational definition of the temperature.

$$T(t) = \sum \frac{m_i v_i^2}{k N_f}$$

Here  $N_f$  = # degrees of freedom

For  $N$  particles in 3D:  $N_f = 3N$

the total kinetic energy of a system fluctuates

so does the instantaneous temperature

# The program

Initialize: position and velocity of all particles

Calculate the forces over all particles

Integrate the equations of motion (Newton's law)

} MD loop

Sample averages: calculate pressure, density, temperature, energy etc

# Initialization

- To start the simulation we assign initial positions and velocities to all particles in the system.
- The particle positions  $\rightarrow$  compatible with the structure to simulate
- Do NOT put the particles positions with overlap of the atomic or molecular cores.
- Usually in a cubic lattice, as in MC
- Give to each particle a RANDOM velocity
- Shift all velocities, such that the total momentum is zero
- Scale the velocities to adjust the temperature  $T$  to the desired value. Scaling:  $v \rightarrow v (T/T_i)^{1/2}$

MD is NOT done at constant T but constant energy.

The value of the average equilibrium temperature is computed in the simulation, will not be the initial T.

The initial velocities are necessary to integrate the equations of motion the first time.

In the simulation, the distribution of velocities will become Maxwell-Boltzman



# Computation of the Force

The force has to be computed with all near particles, the computing time scales as  $N^2$

As in MC, we use periodic boundaries

Algorithm:

Compute the distance between particles  $i, j$

Choosing the cut off  $r_c = \text{Box}/2 \rightarrow$  compute interaction between nearest periodic image of  $j$ .

With simple cubic periodic boundary conditions, the distance in any direction between  $i$  and the nearest image of  $j$  should always be less (in absolute value) than  $\text{box}/2$ .

Compute all Cartesian components of  $r_{ij}$ , and  $|r_{ij}|^2$

Test if is less than  $rc^2 \rightarrow$  if NOT we immediately skip to the next value of  $j$ .  
 $\rightarrow$  if YES compute the force  $f_{ij}$  (each component)

Do NOT compute  $|r_{ij}|$  itself, because this would be both unnecessary and expensive (as it would involve the evaluation of a square root).

The force is obtained as  $f = -\nabla U$

For a LJ potential, 
$$f = \frac{48\vec{r}}{r^2} \left( \frac{1}{r^{12}} - 0.5 \frac{1}{r^6} \right)$$

## Integrating the equations of motion $ma = f$

Numerical integration is done in small time steps, using Taylor expansions in  $dt$

$$r(t + dt) = r(t) + v(t)dt + \frac{f(t)}{2m}dt^2 + \frac{d^3r}{dt^3} \frac{dt^3}{3!} + O(dt^4)$$

### Euler algorithm

$$r(t + dt) = r(t) + v(t)dt + \frac{f(t)}{2m}dt^2 + O(dt^3)$$

NOT recommended

*Catastrophic* Energy drift

## Integrating the equations of motion $ma = f$

Numerical integration is done in small time steps, using Taylor expansions in  $dt$

$$r(t + dt) = r(t) + v(t)dt + \frac{f(t)}{2m}dt^2 + \frac{d^3r}{dt^3} \frac{dt^3}{3!} + O(dt^4)$$

(VERLET algorithm): trick to get rid of the velocity:

$$r(t - dt) = r(t) - v(t)dt + \frac{f(t)}{2m}dt^2 - \frac{d^3r}{dt^3} \frac{dt^3}{3!} + O(dt^4)$$

$$r(t + dt) + r(t - dt) = 2r(t) + \frac{f(t)}{m}dt^2 + O(dt^4)$$

$$r(t + dt) \approx 2r(t) - r(t - dt) + \frac{f(t)}{m}dt^2 + O(dt^4)$$

The Verlet algorithm DO NOT use the velocities.

However we may want to compute the velocities to calculate for example the kinetic energy and temperature.

To compute the velocities:

$$r(t + dt) - r(t - dt) = 2v(t)dt + O(dt^3)$$

$$v(t) = \frac{r(t + dt) - r(t - dt)}{2dt} + O(dt^2)$$

Leap-frog algorithm  
(derived from Verlet)

$$v(t + \Delta t/2) = v(t - \Delta t/2) + \Delta t \frac{f(t)}{m}$$

$$r(t + \Delta t) = r(t) + \Delta t v(t + \Delta t/2)$$

velocities and positions calculated at different times  
Can be problematic with evaluation of the total energy

What is a good algorithm?  
What criteria should satisfy?

- Speed
- Accuracy
- Energy conservation
- Orbit stability
- Time reversibility
- Theorem of Liouville

- **speed** → not very relevant because the fraction of time spent on integrating the equations of motion (as opposed to computing the interactions) is small

- **accuracy for large time steps** → more important, because the longer the time step that we can use, the fewer evaluations of the forces are needed per unit of simulation time.

some algorithms allow the use of a large time step achieve this by storing information on increasingly higher-order derivatives of the particle's coordinates → require more memory storage. (not a serious problem)



- Energy conservation → important criterion, two kinds:  
short time  
long time

Sophisticated higher-order algorithms have very good SHORT TIME energy conservation but NOT long time: energy drift.

Verlet-style algorithms: moderate short-term energy conservation but little long-term drift.

- Lyapunov instability → two trajectories that are initially very close will diverge exponentially as time progresses.

any integration error, no matter how small, will always cause our simulated trajectory to diverge exponentially from the true trajectory compatible with the same initial conditions. not serious!

The aim of an MD simulation is NOT to predict precisely the trajectory of all molecules, we are always interested in **statistical predictions**.

there is considerable numerical evidence that in MD simulations, statistical predictions are good enough.

- Time reversibility** → equations of motion are time reversible, and so should be the MD algorithms.

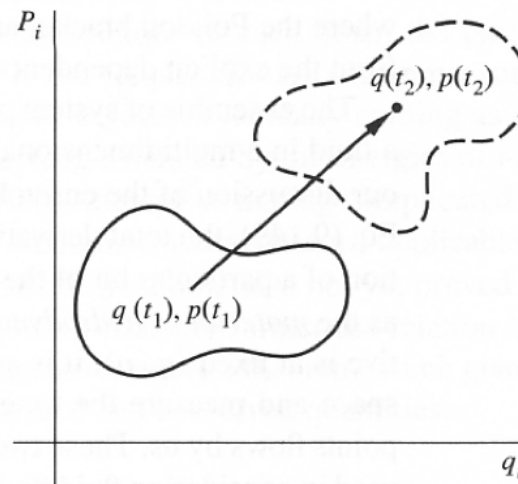
In fact, many algorithms are NOT time reversible. That is, future and past phase space coordinates do not play a symmetric role in such algorithms.

As a consequence, if one were to reverse the momenta of all particles at a given instant, the system would not trace back its trajectory in phase space, even if the simulation would be carried out with infinite numerical precision.

Only in the limit of an infinitely short time step will such algorithms become reversible.

- **Liouville's Theorem: Preservation of volume in phase space**

Hamiltonian dynamics leaves the magnitude of any volume element in phase space unchanged



many numerical schemes (like non-reversible algorithms) are NOT area-preserving: this is not compatible with energy conservation → nonreversible algorithms will have serious long-term energy drift

numerical implementation is NOT time reversible  
(even when we use a time-reversible algorithm)

due to finite machine precision using floating-point  
arithmetic that results in rounding errors

## How about Verlet algorithm?

- is fast.
- not particularly accurate for long time steps.
- requires little memory → This is useful when we simulate very large systems
- short-term energy conservation is fair
- little long-term energy drift
- time reversible
- area preserving
- not very accurate trajectories

## Velocity-corrected Verlet algorithm

$$1) r(t + dt) = r(t) + v(t)dt + \frac{f(t)}{2m} dt^2 + \frac{d^3 r}{dt^3} \frac{dt^3}{3!} + O(dt^4)$$

$$2) r(t - dt) = r(t) - v(t)dt + \frac{f(t)}{2m} dt^2 - \frac{d^3 r}{dt^3} \frac{dt^3}{3!} + O(dt^4)$$

$$3) r(t + 2dt) = r(t) + v(t)2dt + \frac{f(t)}{2m} 4dt^2 + \frac{d^3 r}{dt^3} \frac{8dt^3}{3!} + O(dt^4)$$

$$4) r(t - 2dt) = r(t) - v(t)2dt + \frac{f(t)}{2m} 4dt^2 - \frac{d^3 r}{dt^3} \frac{8dt^3}{3!} + O(dt^4)$$

$$8 \times \{(1)-(2)\} - \{(3)-(4)\} =$$

$$12v(t)dt = 8[r(t + dt) - r(t - dt)] - [r(t + 2dt) - r(t - 2dt)] + O(dt^4)$$

- the error in the velocities is of order  $O(dt^3)$ .
- the velocity can be computed after the next time

## Errors in MD and MC

Errors in MD and MC are of 3 type:

- Systematic: for example related to finite size effects, interaction cutoff. INTRINSIC to the simulation.
- Errors due inadequate sampling, for example a bad RNG, or acceptance rules not satisfying DB: POOR DESIGN, must be corrected.
- Statistical errors due to random fluctuations. These errors determine the degree of confidence in our results.

Only for the statistical errors we can apply statistical analysis.



Suppose the fluctuating property  $A$  is measured  $n$  times in a system in equilibrium. The mean value is:

$$\langle A \rangle = \frac{1}{n} \sum_{i=1}^n A_i$$

and if each measurement is independent with variance

$$\sigma^2(A) = \frac{1}{n} \sum_{i=1}^n (A_i - \langle A \rangle)^2 = \langle A^2 \rangle - \langle A \rangle^2$$

then the variance of the mean is:

$$\sigma^2(\langle A \rangle) = \sigma^2(A)/n$$

In MC we average over many  $n$  MC steps.  
In MD we average over  $n$  time-steps.

The problem is:

there is a correlation between consecutive configurations, then the  $n$  values  $A_i$  in the sample are not all independent

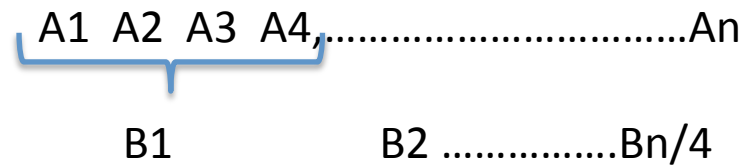
The effective value of independent measurements is less than  $n$ .

A variance calculated with all the  $A_i$  values would underestimate the real  $\sigma$ .

## Estimation of errors in MD

If averages are evaluated over blocks of successive values, as the block size increases the block averages will be decreasingly correlated.

For example, blocks of 4:



{B } are less correlated than {A}

if the block length exceeds the correlation-time, we will have an independent sample.

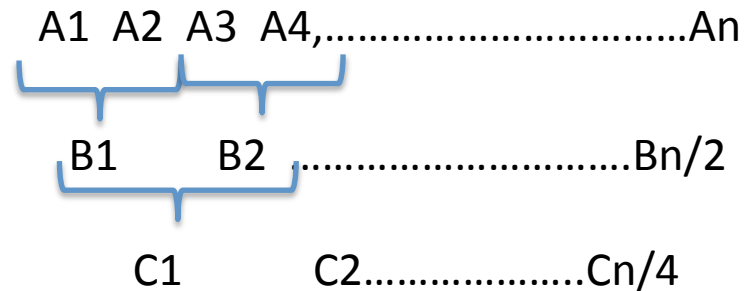
The correlation time is unknown.

How long should the block be to estimate the error?

If the block length is too short, there is little improvement

If it is too long, reduces the number of values in our sample.

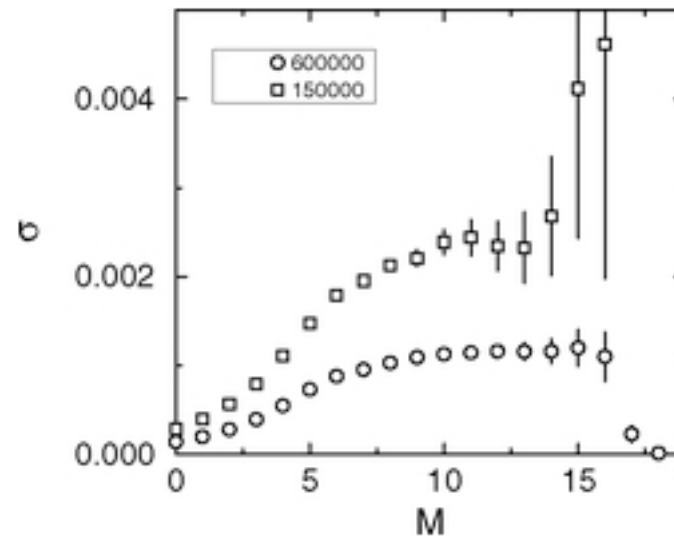
Scheme: Series of successive block sizes  $b=1,2,4,8,\dots$



We calculate the variance of each sample A, B, C....

The successive values of  $\sigma$  will increase until a plateau is reached.

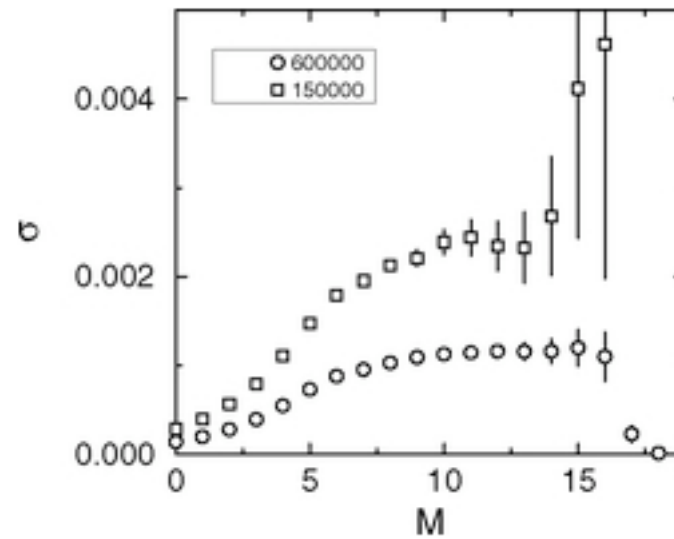
The plateau value is the result for  $\sigma$ .



The standard deviation  $\sigma$  in the potential energy as a function of the number of block operations  $M$  for a simulation of 150,000 and 600,000 time steps.

It also shows if the simulation time is adequate:

If total time is too short, the plateau is not reached or is too narrow



MD at constant T

## MD at constant temperature

In an experiment, constant temperature  $T$  means thermal equilibrium with a reservoir at  $T$ .

In the canonical ensemble the system is also in thermal equilibrium with a bath at  $T$ .

However thermal equilibrium doesn't mean constant temperature!

The instantaneous temperature of a system is related to the kinetic energy, and KE fluctuates



In a canonical ensemble of a finite system, the instantaneous kinetic temperature  $T_k$  fluctuates.

In fact, if we were to keep the average kinetic energy per particle rigorously constant, as is done in the so-called isokinetic MD scheme or velocity-scaling schemes, then we would not simulate the true constant-temperature ensemble.

In practice, the difference between isokinetic and canonical schemes is often negligible.

Problems can be expected if isokinetic simulations are used to measure equilibrium averages that are sensitive to fluctuations .

Any kind of temperature regulation can be used while preparing the system at a desired temperature (i.e., during **equilibration**).

## MD in a Canonical Ensemble: The Andersen Thermostat

The system is coupled to a heat bath that imposes the desired temperature.

The coupling to a heat bath is represented by stochastic impulsive forces that act occasionally on randomly selected particles.

These stochastic collisions with the heat bath can be considered as Monte Carlo moves that transport the system from one constant-energy shell to another.

Between stochastic collisions, the system evolves at constant energy according to the normal Newtonian laws of motion (MD).

The stochastic collisions ensure that all accessible constant-energy shells are visited according to their Boltzmann weight.

Before starting such a constant-temperature simulation, we should first select the strength of the coupling to the heat bath.

This coupling strength is determined by the frequency of stochastic collisions:  $\nu$

A constant-temperature simulation now consists of the following steps:

1. Start with an initial set of positions and momenta  $\{r^N(0), p^N(0)\}$  and integrate the equations of motion for a time  $\Delta t$ .
2. A number of particles are selected to undergo a collision with the heat bath. The probability that a particle is selected in a time step of length  $\Delta t$  is  $v\Delta t$ .
3. If particle  $i$  has been selected to undergo a collision, its new velocity will be drawn from a Maxwell-Boltzmann distribution corresponding to the desired temperature  $T$ . All other particles are unaffected by this collision.

The mixing of Newtonian dynamics with stochastic collisions turns the Molecular Dynamics simulation into a Markov process

The Andersen algorithm does, indeed, generate a canonical distribution.

### Implementation:

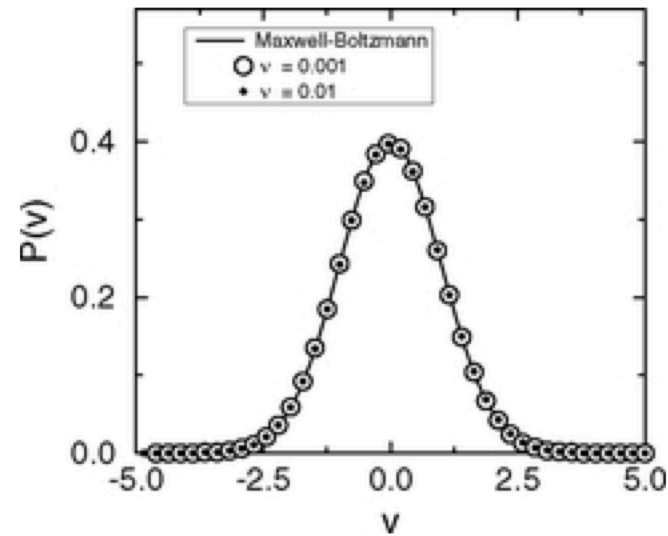
In subroutine SOLVE:

```
sigma=sqrt(temp)
do i=1,npart
  if(ranf.lt.nu*dt) then
    v(i)=gauss(sigma)
  endif
enddo
```

nu is a parameter (the frequency of collisions)

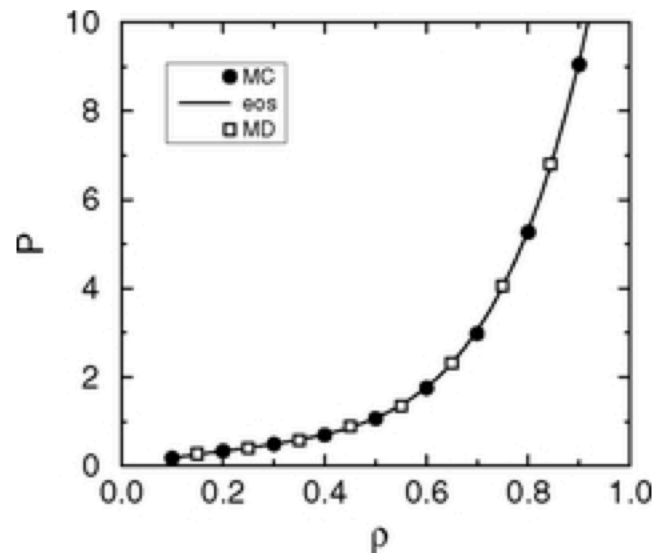
The results should be independent of the value of nu.

Anderson thermostat is NOT deterministic, but stochastic.



**Figure 6.1:** Velocity distribution in a Lennard-Jones fluid ( $T = 2.0$ ,  $p = 0.8442$ , and  $N = 108$ ). The solid line is the Maxwell-Boltzmann distribution (6.1.1), and the symbols are from a simulation using  $v = 0.01$  and  $v = 0.001$  as collision rates.

The results of constant  $N,V,T$  Molecular Dynamics simulations should be identical to those of canonical Monte Carlo



**Figure 6.2:** Equation of state of the Lennard-Jones fluid ( $T = 2.0$  and  $N = 108$ ); comparison of the Molecular Dynamics results using the Andersen thermostat (open symbols) with the results of Monte Carlo simulations (closed symbols) and the equation of state of Johnson *et al.* [62].

Andersen thermostat is fine for equilibrium properties  
Fails for dynamic properties

The stochastic collisions disturb the dynamics in a way that is not realistic

it leads to sudden random decorrelation of particle velocities. This effect will result in an enhanced decay of the velocity autocorrelation function.



## MD at constant T: deterministic approach Nose-Hoover thermostat

The thermal bath is added to the system as **extra coordinates** in the Lagrangian

The MD simulation is carried out for the “extended” system

Energy is allowed to flow dynamically from the reservoir to the system and back

“It is like controlling the volume with a piston in NPT”

Extra degree of freedom:  $s, p_s$

$$\mathcal{L}_{\text{Nose}} = \sum_{i=1}^N \frac{m_i}{2} s^2 \dot{\mathbf{r}}_i^2 - \mathcal{U}(\mathbf{r}^N) + \frac{Q}{2} \dot{s}^2 - \frac{L}{\beta} \ln s,$$

L: parameter

Q: effective “mass” of the extra particle

momenta:

$$p_i = \frac{\partial L}{\partial \dot{r}_i} = m_i s^2 \dot{r}_i$$
$$p_s = \frac{\partial L}{\partial \dot{s}} = Q \dot{s}$$

$$\mathcal{H}_{\text{Nose}} = \sum_{i=1}^N \frac{\mathbf{p}_i^2}{2m_i s^2} + \mathcal{U}(\mathbf{r}^N) + \frac{p_s^2}{2Q} + L \frac{\ln s}{\beta}.$$

$$\begin{aligned} Q_{\text{Nose}} &= \frac{1}{N!} \int d\mathbf{p}_s d\mathbf{p}'^N d\mathbf{r}^N ds \frac{\beta s^{3N+1}}{L} \\ &\quad \times \delta \left\{ s - \exp \left[ -\beta \frac{\mathcal{H}(\mathbf{p}', \mathbf{r}) + p_s^2/(2Q) - E}{L} \right] \right\} \\ &= \frac{1}{N!} \frac{\beta \exp[E(3N+1)/L]}{L} \int d\mathbf{p}_s \exp \left[ -\beta \frac{3N+1}{L} p_s^2/(2Q) \right] \\ &\quad \times \int d\mathbf{p}'^N d\mathbf{r}^N \exp \left[ -\beta \frac{3N+1}{L} \mathcal{H}(\mathbf{p}', \mathbf{r}) \right] \end{aligned}$$

partition function  $\rightarrow$  
$$= C \frac{1}{N!} \int d\mathbf{p}'^N d\mathbf{r}^N \exp \left[ -\beta \frac{3N+1}{L} \mathcal{H}(\mathbf{p}', \mathbf{r}) \right].$$

With the choice  $L = 3N + 1$ , this ensemble reduces to the **canonical ensemble**