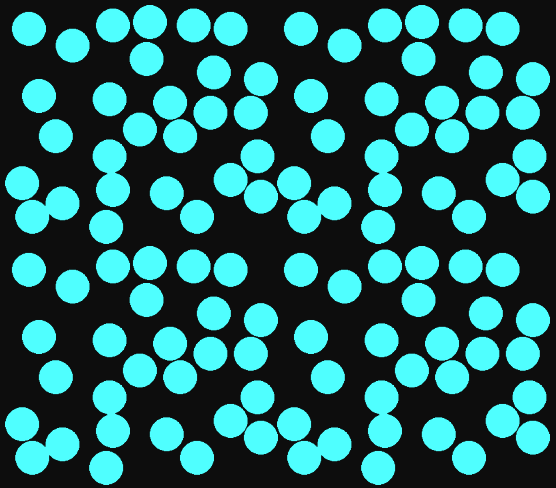
Molecular Dynamics

Ramses van Zon

PHY1610 Winter 2026

Molecular Dynamics Simulations

Used in chemical physics, materials science and the modelling of bio-molecules.



- N interacting microscopic particles (atoms, molecules, nanoparticles).
- Newton's equations of motion

$$m_i \ddot{r}_i = f_i(\{r_j\})$$

- Initial conditions.

Molecular Dynamics Simulations

Numerical solution involves:

- Initializing the system.
- Discretizing time: particles move from one time point to the next.
- Making sure every finite time step solves the differential equation to some approximation.
- Forces need to be computed every time step.
- Many time steps are taken.

Molecular Dynamics Simulations

What makes MD different from other ordinary differential equations?

- Hamiltonian dynamics
- Typical solvers (E.g. Runge Kutta) require too many force evaluations, but it is very expensive to evaluation of $\boldsymbol{f}$ if N is large.
- In a broader sense, N-body gravitational systems fall into this class too.

Specific to Molecular Dynamics

- Often the aim is to study equilibrium properties, or transport properties near equilibrium.
- The number of particles N is very large.
- Different forces acting at the same time, both short range and long range (this matters).
- Usually exchange with an implicit environment, e.g.
  - exchange energy to simulating a system at specific temperature
  - exchange volume if simulating a system at specific pressure
  - exchange particle number if simulating at a specific chemical potential

Hamiltonian dynamics

- Molecular Dynamics aims to compute *equilibrium*, *dynamical* and *transport* properties of *classical many body systems*.
- Many classical systems have Newtonian equations of motion:

$$\dot{r} = \frac{1}{m}p \qquad\qquad \dot{p} = F = -\frac{\partial U}{\partial r},$$

- Energy $H = \frac{|p|^2}{2m} + U(r)$ is conserved under the dynamics.

Hamiltonian dynamics

- Potential energy is typically a sum of pair potentials:

$$U(r) = \sum_{(i,j)} \varphi(r_{ij}) = \sum_{i=1}^{N} \sum_{j=1}^{i-1} \varphi(r_{ij}),$$

  which entails the following expression for the forces $\boldsymbol{F}$:

$$\mathbf{F}_i = -\sum_{j \neq i} \frac{\partial}{\partial \mathbf{r}_i} \varphi(r_{ij}) = \sum_{j \neq i} \varphi'(r_{ij}) \frac{\mathbf{r}_j - \mathbf{r}_i}{r_{ij}}$$

- The double sum makes this (at first) an $O(N^2)$ algorithm.

Equilibrium properties

If the purpose of an MD simulation is some equilibrium property, one want to compute the average of that property in all possible configurations of the atoms.

One can impose different constraints to the possible configurations, such as

- Those with the same number of particles, total energy E, and volume:

  micro-canonical ensemble average

- Those with a prescribed number of particles, temperature T, and volume. This involve specific energy fluctuations: **canonical ensemble average**
- Variations with prescribed pressure, or prescribed chemical potentials.

The microcanonical ensemble is the easiest from a numerical perspective, because total energy, number of particles and volume are intrinsic properties of the system.

Hamiltonian dynamics as sampling

- If the system is "ergodic" then a time average equals the equilibrium micro-canonical average:

$$\lim_{t_{\text{final}}\to\infty} \frac{1}{t_{\text{final}}} \int_0^{t_{\text{final}}} dt\, A(x(t)) = \frac{\int dx\, A(x)\, \delta(E - H(x))}{\int dx\, \delta(E - H(x))}.$$

- But the simulation can only give finite number of configurations; it is sampling the configuration space.
- Need long times t_{final} for proper sampling.
- It helps to forget initial equilibration time t_{equil}.
- Monte Carlo would work as well, but often, Hamiltonian dynamics moves the system a more appropriate direction than random displacements.

Hamiltonian dynamics as dynamics

- We're often also interested in non-equilibrium dynamics, i.e., non-steady dynamics

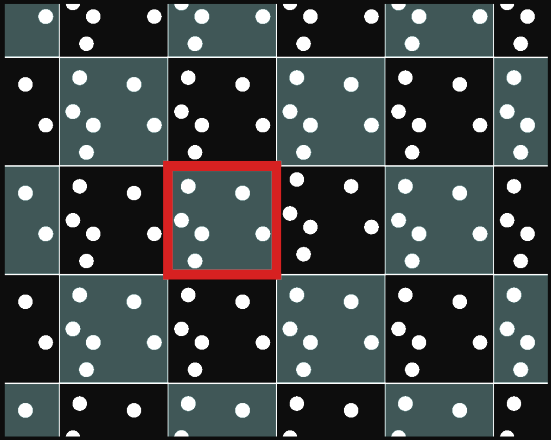
  (protein folding, phase transitions, chemical reactions).
- In non-steady situation, the modeling matters often more, i.e., choice of ensemble, accuracy of the computational method, etc.
- Methods used to improve equilibrium sampling may not be appropriate here.
- Hard to reach large enough time scales to see the macroscopic dynamics.

Boundary conditions

- When simulating a finite system, its container can be modeled with a wall force.
- But one usually has to be satisfied with simulating rather smaller size problems, and a wall force would give finite size effects as well as destroy translation invariance.
- More benign boundary conditions are:

Periodic Boundary Conditions

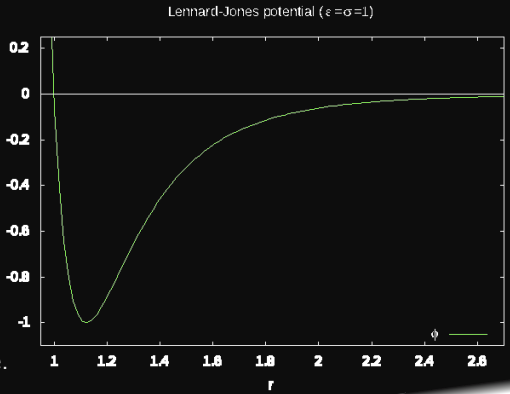
- Infinite repetition of a finite volume.
- Particles coordinates between $-L/2$ and $L/2$.
- A particle exiting simulation box is put back at the other end.
- The box with thick red boundaries is our simulation box.



Force calculations

A common potential for non-bonded interactions (e.g. between neutral, spherical particles), is the Lennard-Jones potential:

$$\varphi(r) = 4\varepsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^{6} \right],$$



- σ is a measure of the range of the potential.
- ε is its strength.
- The potential is positive for small r: repulsion.
- The potential is negative for large r: attraction.
- The potential goes to zero for large r: short-range.
- The potential has a minimum of $-\varepsilon$ at $2^{1/6}\sigma$.

Force computation

- Force computation often the most demanding part of MD:
- Computing all forces in an N-body system requires the computation of $N(N-1)/2$ forces $\mathbf{F}_{ij}$.
- Using periodic boundary conditions makes $N = \infty$.

We will need a few tricks:

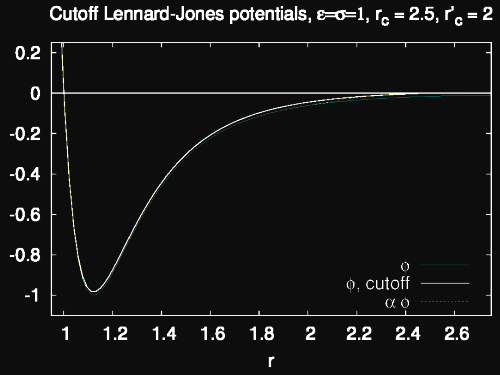
- Cut-off of interaction rage
- Cell division
- Neighbour lists
- Parallelization

Cut-off

Change potential to become zero beyond a *cut-off* distance r_c:

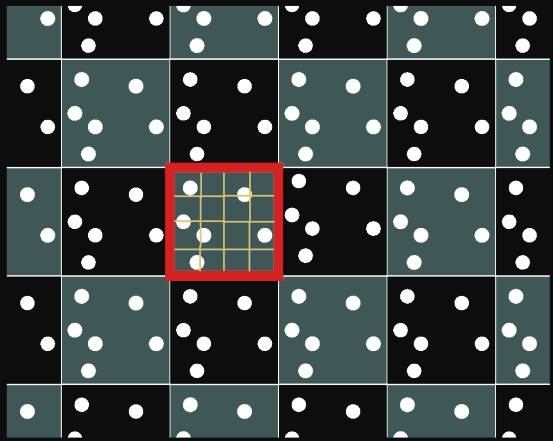
$$\varphi'(r) = \begin{cases} \varphi(r) - \varphi(r_c) & \text{if } r < r_c \\ 0 & \text{if } r \geq r_c \end{cases}$$

This reduces $\mathcal{O}(\infty)$ to $\mathcal{O}(N^2)$.



Cell division

- Divide the simulation box into cells larger than the cutoff r_c.
- Make a list of all particles in each cell.
- In the sum over pairs in the force computation, only sum neighbours i.e., particles in the same cell or in adjacent cells.
- For systems with short-range interactions: $\mathcal{O}(N^2) \rightarrow \mathcal{O}(N)$.
- More refine divisions use *Verlet neighbour lists*.



Long-range interaction

- Gravity and electrostatics are examples of long range interactions that cannot be cut off without seriously altering the physics.
- Electrostatic is a very common non-bonded interaction in MD with charged or polar molecules.
- Without a cut-off, computing the sum over pairs, or "Particle-Particle'', $\mathcal{O}(N^2)$ methods seem unavoidable.

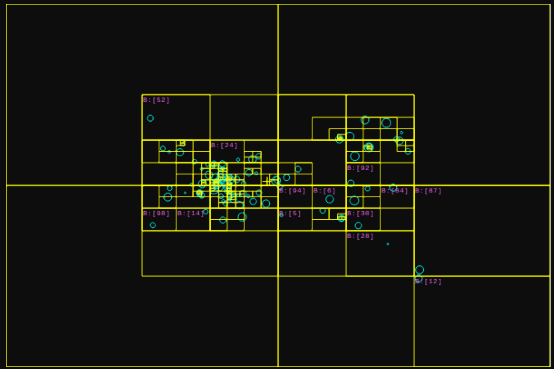
However, not necessarily!

There exists special techniques such as

- Barnes-Hut
- Particle Mesh -Particle-Particle/Particle Mesh (P3M) or Ewald Sums.

Barnes-Hut

- Clump particles together in cells
- Cells with too many particles are subdivided (recursive)
- Leads to a quad (2d) or octal (3d) tree
- Pretend each box is one massive particle at the centre to compute the force.



- Or, more accurately, replace by a multipole expansion.

Particle-Mesh

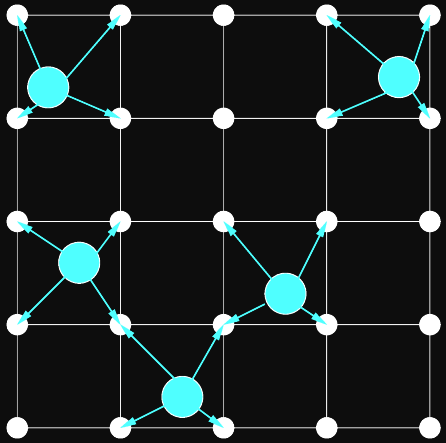
- Choose rectangular mesh
- Distribute masses (blue) to mesh vertices (black circles)
- Determine the gravitational potential using FFT:

$$\nabla^2\Phi = 4\pi G\rho \Rightarrow \hat{\Phi} = -\frac{4\pi G\hat{\rho}}{k^2}$$

- The forces on the lattice are given by the $\nabla\Phi$ in real space, i.e, the fourier inverse of

$$\hat{F} = i\mathbf{k}\hat{\Phi} = -i\mathbf{k}\frac{4\pi G\hat{\rho}}{k^2}$$

- The inverse FFT gives the force on the particles.



- $\mathcal{O}(N \log N)$.

Ewald Sums and P3M

- Particle-Mesh is fast, but not very accurate.
- This is because the short range part of the forces is poorly represented.
- One can do better.
- Idea of P3M or Ewald summation is to do an exact summation of forces with bodies nearby, and perform an approximate calculation for bodies further away.
- Ewald does not assign to grid, but pays for this: $\mathcal{O}(N^{3/2})$.
- P3M still assigns masses to a regular grid, allowing for $\mathcal{O}(N \log N)$ scaling.
- It relies on being able to translate this separation of local and further-away in fourier space.

Bonding interactions

- The molecules in MD models typically have a substructure; they are composed of atoms that are held together by bonding potentials
- For linear molecules (where atoms are linked to one previous and one next atom), one can define potentials for
  1. the distance between subsequent atoms along the chain
  2. the bond angle θ between subsequent links along the chain
  3. the torsion angles ϕ between subsequent link-pairs along the chain.
- (1) is often so strong that the bonds vibrate at high frequency. This reduces the possible time step, so when bond vibrations are not important, one replaces their potentials with a constraint, i.e., $r_{i,i+1} = l$, using Lagrange multipliers.
- Small molecules for which internal vibrations can be neglected, one can treat them as rigid bodies, with orientation and internal angular momenta as additional variable in the algorithm.

Requirements for MD time step algorithms

- **Efficiency**

  (we just took care of most of that with the cells, verlet lists, P3M and Ewald sums)

- **Accuracy**

  Mostly, we don't want unphysical stuff like particle moving through each other.

- **Stability**

- **Respect physical laws:**

  - Time reversal symmetry
  - Conservation of energy
  - Conservation of linear momentum
  - Conservation of angular momentum
  - Conservation of phase space volume

The most efficient algorithm is then the one that allows the largest possible time step for a given level of accuracy, **while maintaining stability and preserving conservation laws**.

Symplectic integrators

Momentum Verlet Scheme (first version)

$$r_{n+1} = r_n + \frac{p_n}{m}h + \frac{F_n}{2m}h^2$$
$$p_{n+1} = p_n + \frac{F_{n+1} + F_n}{2}h$$

The momentum rule appears to pose a problem since F_{n+1} is required. But to compute F_{n+1}, we need only r_{n+1}, which is computed in the integration step as well.

Symplectic integrators

- 'Symplectic' means the approximate dynamics preserves some aspects of the original.
- In particular, for small enough time steps, there's approximate energy conservation, and a limit distribution ($\Rightarrow$ no drift).

Symplectic integrators

Momentum Verlet Scheme (second version)

The extra storage step can be avoided by introducing the half step momenta as intermediates:

$$p_{n+1/2} = p_n + \frac{1}{2}F_n h$$
$$r_{n+1} = r_n + \frac{p_{n+1/2}}{m}h$$
$$p_{n+1} = p_{n+1/2} + \frac{1}{2}F_{n+1}h$$

Also nice and symmetric:

- Half momentum step
- Full position step
- Half momentum step

First step the same as the last (with updated F).

Example code: Serdy

- Serdy is a C code that simulates a Lennard-Jones fluid (think noble gases like Argon).

Serdy Code Modules

- atom
  defines a structure to hold each atom's properties
- system
  defines a structure to hold system parameters.
- lattice
  creates points on a cubic lattice.
- lcg
  random number generator
- cells
  functions related to dividing the system into cells
- forces
  functions to compute the forces.
- inifile
  functions to deal with parameters
- serdy
  glues the components together

```
$ git clone https://github.com/vanzonr/serdy.git
$ cd serdy
$ module load gcc
$ make
$ make test
```

References

Books:

- D.C. Rapaport, *The Art of Molecular Dynamics Simulations*, 2nd ed. (2004)
- D. Frenkel and B. Smit, *Understanding Molecular Simulation*, 2nd ed. (2001)

Lecture notes from CHM1464H 2008 (by Prof. J. Schofield and Dr. R. van Zon):

- https://sites.chem.utoronto.ca/chemistry/jmschofi/simulation/notes.html

Code:

- https://github.com/vanzonr/serdy

Standard MD codes:

- LAMMPS
- GROMACS
- NAMD
- AMBER
- OpenMM
- ...

en.wikipedia.org/wiki/Comparison_of_software_for_molecular_mechanics_modeling