MOLECULAR DYNAMICS

202303012308

1. MOLECULAR DYNAMICS (MD)

is where you model the trajectory of *each particle* in a many-particle system.

you model the particles’ interaction with distance-dependent interatomic potentials. then numerically solve newton’s equations of motion for the system of particles.

is useful for modelling dense gases, liquids, solids.

¶ assumptions (about the particles)

- classical
- spherical
- chemically inert
- interactions between them depend only on distance.

2. THE INTERMOLECULAR POTENTIAL

¶ general form of U: should be a sum of two-particle interaction potentials between all the particles.

$$U_{\text{total}} = \sum_i \sum_j U\left(r_{ij}\right) \qquad \text{for } i = 1 : (N-1)\,,\ j = (i+1) : N \tag{1}$$

¶ two properties that U(r) must satisfy

1. strong repulsion for small r.

   - [because the pauli exclusion principle… there’s a core repulsion as the wave functions of the two particles distort to avoid overlapping. causes some of the electrons to “be” in different quantum states. ∴ an effective repulsion].

2. weak attraction for big r.

- [pq the vdw potential. mutual polarisation of each molecule].

3. LENNARD JONES POTENTIAL

is

$$U(r) = 4\varepsilon\left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right) \tag{2}$$

two parameters: ε and σ.

- σ = “length”. this is the point where U(r) = 0 (at r = σ). U(r) is close to zero for r > 2.5σ.
- ε = the depth of the potential at the minimum of U(r). the minimum occurs at separation 2$^($1/6$^)$σ.

note if you look at a plot of the lennard jones potential U(r) it’s easy to see why σ is a length and ε is an energy.

4. REDUCED UNITS

¶ **you can express all units in terms of σ,ε,m.**

- since σ is a distance, define it to be the distance unit for the problem.
- and since ε is an energy, define it to be the energy unit.
- also define the mass of one atom/molecule/particle/whatever to be the mass unit.

in terms of these base units:

¶ **velocity:** $(\varepsilon/m)^{1/2}$

- where the fuck did this come from? «

¶ **time:** $\sigma(\varepsilon/m)^{-1/2} = \sigma(m/\varepsilon)^{1/2}$

- ok I get this one, d = vt ∴ t = d/v «<

¶ **force:** ε/σ

- f units are m·v/t = $(\varepsilon/m)^{1/2} m \frac{1}{\sigma} (\varepsilon/m)^{1/2} = \varepsilon/\sigma$

¶ **pressure:** ε/σ^2

¶ **temperature:** ε/k_B

5. INTEGRATOR

need a symplectic integrator because:

- preserves phase space volume
- preserves conservation laws
- time reversible
- accurate for large time steps, to reduce cpu time needed for the total time of the simulation.

¶ **velocity verlet**

$$x_{n+1} = x_n + v_n \Delta t + \frac{1}{2} a_n (\Delta t)^2$$

$$v_{n+1} = v_n + \frac{1}{2} (a_{n+1} + a_n) \Delta t$$

- this is only for one component of a particle’s motion.
- you use the new position, $x_{(}n+1_{)}$, to find the new acceleration, $a_{(}n+1_{)}$.
- then you use both the new acceleration $a_{(}n+1_{)}$ and the current acceleration $a_{(}n_{)}$ to find the new velocity $v_{(}n+1_{)}$.

6. PERIODIC BOUNDARY CONDITIONS

if a particle leaves the box it should “reenter” from the opposite edge/wall.

note, irl, the effects of collisions with walls is negligible. the fraction of particles interacting with walls is small compared to the total number of particles. in the simulation, we have a finite size problem.

¶ **minimum image approximation**

7. MD PROGRAM

use one “class” to represent all N particles.

store the x,y comps of pos and vel in the state array. store accelerations of pcls in separate array.

you need to figure out a scheme for initialising the positions and velocities.

- for velocities, use the equipartition theorem
- for positions, there are a few ways. the easiest way to get a config with a desired density is to place them on a regular lattice.

the most time consuming part of the program is computing the accelerations of the particles.

8. INITIAL VELOCITIES

equipartition theorem: the average KE of a particle per degree of freedom, $K_{pp,pdof}$ is:

$$K_{pp,pdof} = \frac{1}{2} k_B T \tag{3}$$

also, mind that:

$$\begin{aligned} K_{pp,pdof} &= K_{total} / \text{ numDoFs} \\ &= K_{pp}/d \\ &= K_{total}/Nd \end{aligned} \tag{4}$$

- N is the number of particles, d is number of spatial dimensions.
- and K_{total} is the average *total* KE of the system, i.e. $\sum_i \frac{1}{2} m_{[i]} {v_{[i]}}^2$ for all the particles in the system

∴ define temperature at time t:

$$k_B T(t) = \frac{2K_{total}(t)}{Nd} = \frac{1}{Nd} \sum_{i=1:N} m_{[i]} \left(\vec{v}_{[i]}(t) \bullet \vec{v}_{[i]}(t) \right) \tag{5}$$