Molecular Dynamics: basic methods

Pascal Viot

September 20, 2020

Molecular Dynamics : Verlet algorithm

Simple Liquids

- m mass of the particle, σ diameter, and ϵ energy scale of the interaction potential: typical time

$$\tau = \sigma\sqrt{\frac{m}{\epsilon}}$$

Molecular Dynamics : Verlet algorithm

Simple Liquids

- m mass of the particle, σ diameter, and ϵ energy scale of the interaction potential: typical time

$$\tau = \sigma\sqrt{\frac{m}{\epsilon}}$$

- Argon:, $\sigma = 3\text{Å}$, $m = 6.63.10^{-23}kg$ and $\epsilon = 1.64.10^{-20}J$, which gives

$$\tau = 2.8.10^{-14}s$$

.

Molecular Dynamics : Verlet algorithm

Simple Liquids

- m mass of the particle, σ diameter, and ϵ energy scale of the interaction potential: typical time

$$\tau = \sigma\sqrt{\frac{m}{\epsilon}}$$

- Argon:, $\sigma = 3\text{Å}$, $m = 6.63.10^{-23}kg$ and $\epsilon = 1.64.10^{-20}J$, which gives

$$\tau = 2.8.10^{-14}s$$

.

- Typically $\Delta t = 10^{-15}s$, even smaller.

Molecular Dynamics : Verlet algorithm

Simple Liquids

- m mass of the particle, σ diameter, and ϵ energy scale of the interaction potential: typical time

$$\tau = \sigma\sqrt{\frac{m}{\epsilon}}$$

- Argon:, $\sigma = 3\text{Å}$, $m = 6.63.10^{-23}kg$ and $\epsilon = 1.64.10^{-20}J$, which gives

$$\tau = 2.8.10^{-14}s$$

.

- Typically $\Delta t = 10^{-15}s$, even smaller.
- The total number of steps performed in a run is typically of order of magnitude is 10^7; the simulation duration is

$$10^{-8}s$$

Equations of motion

From exact equations to discretization

- $$m\frac{d^2\mathrm{r}_i}{dt^2} = -\sum_{j\neq i} \nabla_{\mathrm{r}_i} u(\mathrm{r}_{ij}).$$

  where m is the particle mass.

Equations of motion

From exact equations to discretization

- $$m\frac{d^2\mathrm{r}_i}{dt^2} = -\sum_{j\neq i} \nabla_{\mathrm{r}_i} u(\mathrm{r}_{ij}).$$

  where m is the particle mass.

- Force calculation acting on particle i involves the computation of de $(N-1)$ elementary forces between each particle and particle i

Equations of motion

From exact equations to discretization

- $$m\frac{d^2\mathrm{r}_i}{dt^2} = -\sum_{j\neq i} \nabla_{\mathrm{r}_i} u(\mathrm{r}_{ij}).$$

  where m is the particle mass.

- Force calculation acting on particle i involves the computation of de $(N-1)$ elementary forces between each particle and particle i

- For the sake of simplicity
$$m\frac{d^2\mathrm{r}}{dt^2} = \mathrm{f}(\mathrm{r}(t))$$

Equations of motion

From exact equations to discretization

- A time series expansion gives

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

Equations of motion

From exact equations to discretization

- A time series expansion gives

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

-

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

Equations of motion

From exact equations to discretization

- A time series expansion gives

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

-

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

- Adding these above equations, one obtains

$$\mathrm{r}(t+\Delta t) + \mathrm{r}(t-\Delta t) = 2\mathrm{r}(t) + \frac{\mathrm{f}(\mathrm{r}(t))}{m}(\Delta t)^2 + \mathcal{O}((\Delta t)^4).$$

Equations of motion

From exact equations to discretization

- The trajectories are calculated with an accuracy of $(\Delta t)^4$.

Equations of motion

From exact equations to discretization

- The trajectories are calculated with an accuracy of $(\Delta t)^4$.
- $$\mathrm{v}(t) = \frac{\mathrm{r}(t+\Delta t) - \mathrm{r}(t-\Delta t)}{2\Delta t} + \mathcal{O}((\Delta t)^2)$$

Equations of motion

From exact equations to discretization

- The trajectories are calculated with an accuracy of $(\Delta t)^4$.
- $$\mathrm{v}(t) = \frac{\mathrm{r}(t+\Delta t) - \mathrm{r}(t-\Delta t)}{2\Delta t} + \mathcal{O}((\Delta t)^2)$$
- The Verlet's algorithm preserves the time symmetry. This is, if we change $\Delta t \to -\Delta t$, discrete equation of motion are the same

Equations of motion

From exact equations to discretization

- The trajectories are calculated with an accuracy of $(\Delta t)^4$.
- $$\mathrm{v}(t) = \frac{\mathrm{r}(t+\Delta t) - \mathrm{r}(t-\Delta t)}{2\Delta t} + \mathcal{O}((\Delta t)^2)$$
- The Verlet's algorithm preserves the time symmetry. This is, if we change $\Delta t \to -\Delta t$, discrete equation of motion are the same
- Leapfrog algorithm

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

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

Hard sphere model

Event-driven algorithm

- Hamiltonian

$$\mathcal{H}_N = \sum_i^N \frac{1}{2} m v_i^2 + \frac{1}{2} \sum_{i \neq j} u(\mathrm{r}_i - \mathrm{r}_j),$$

where the interaction potential is

$$u(\mathrm{r}_i - \mathrm{r}_j) = \begin{cases} +\infty & |\mathrm{r}_i - \mathrm{r}_j| \leq \sigma \\ 0 & |\mathrm{r}_i - \mathrm{r}_j| > \sigma \end{cases}$$

Hard sphere model

Event-driven algorithm

- Hamiltonian

$$\mathcal{H}_N = \sum_i^N \frac{1}{2} m v_i^2 + \frac{1}{2} \sum_{i \neq j} u(\mathrm{r}_i - \mathrm{r}_j),$$

where the interaction potential is

$$u(\mathrm{r}_i - \mathrm{r}_j) = \begin{cases} +\infty & |\mathrm{r}_i - \mathrm{r}_j| \leq \sigma \\ 0 & |\mathrm{r}_i - \mathrm{r}_j| > \sigma \end{cases}$$

- $\exp(-\beta u(\mathrm{r}_i - \mathrm{r}_j))$ is a binary variable: 0 (when two spheres overlap) or 1 otherwise.

Hard sphere model

Event-driven algorithm

- Hamiltonian

$$\mathcal{H}_N = \sum_i^N \frac{1}{2} m v_i^2 + \frac{1}{2} \sum_{i \neq j} u(\mathrm{r}_i - \mathrm{r}_j),$$

where the interaction potential is

$$u(\mathrm{r}_i - \mathrm{r}_j) = \begin{cases} +\infty & |\mathrm{r}_i - \mathrm{r}_j| \leq \sigma \\ 0 & |\mathrm{r}_i - \mathrm{r}_j| > \sigma \end{cases}$$

- $\exp(-\beta u(\mathrm{r}_i - \mathrm{r}_j))$ is a binary variable: 0 (when two spheres overlap) or 1 otherwise.
- The excess thermodynamic quantities do not depend on the temperature.

Hard sphere model

Event-driven algorithm

- Galilean motion between binary collisions

Hard sphere model

Event-driven algorithm

- Galilean motion between binary collisions
- Exact molecular dynamics: event-driven algorithm

Hard sphere model

Event-driven algorithm

- Galilean motion between binary collisions
- Exact molecular dynamics: event-driven algorithm
- Consider two spheres of identical mass: during the collision, the total momentum is conserved

$$v_1 + v_2 = v_1' + v_2'$$

Hard sphere model

Event-driven algorithm

- Galilean motion between binary collisions
- Exact molecular dynamics: event-driven algorithm
- Consider two spheres of identical mass: during the collision, the total momentum is conserved

$$v_1 + v_2 = v_1' + v_2'$$

- Elastic collision

$$(v_1' - v_2').n = -(v_1 - v_2).n$$

$$(v_1' - v_2').t = (v_1 - v_2).t$$

Hard sphere model

Event-driven algorithm

- $$v_1' = v_1 + \frac{(v_2 - v_1).(r_1 - r_2)}{(r_1 - r_2)^2}(r_1 - r_2)$$
$$v_2' = v_2 + \frac{(v_1 - v_2).(r_1 - r_2)}{(r_1 - r_2)^2}(r_1 - r_2)$$

Hard sphere model

Event-driven algorithm

- $$v_1' = v_1 + \frac{(v_2 - v_1).(r_1 - r_2)}{(r_1 - r_2)^2}(r_1 - r_2)$$
$$v_2' = v_2 + \frac{(v_1 - v_2).(r_1 - r_2)}{(r_1 - r_2)^2}(r_1 - r_2)$$

- Between collisions, the positions of i and j are given by

$$r_i = r_i^0 + v_i t$$

$$r_j = r_j^0 + v_j t$$

Hard sphere model

Event-driven algorithm

- $$v_1' = v_1 + \frac{(v_2 - v_1).(r_1 - r_2)}{(r_1 - r_2)^2}(r_1 - r_2)$$
$$v_2' = v_2 + \frac{(v_1 - v_2).(r_1 - r_2)}{(r_1 - r_2)^2}(r_1 - r_2)$$

- Between collisions, the positions of i and j are given by
$$r_i = r_i^0 + v_i t$$
$$r_j = r_j^0 + v_j t$$

- The contact condition between two particles is given by relation
$$\sigma^2 = (r_i - r_i)^2 = (r_i^0 - r_j^0)^2 + (v_i - v_j)^2 t^2 + 2(r_i^0 - r_j^0)(v_i - v_j)t$$

Hard sphere model

Event-driven algorithm

- The collision time is a solution of a quadratic equation,

Hard sphere model

Event-driven algorithm

- The collision time is a solution of a quadratic equation,
- 1. Complex roots: the collision time is set to a very large value in simulation

Hard sphere model

Event-driven algorithm

- The collision time is a solution of a quadratic equation,
- 1. Complex roots: the collision time is set to a very large value in simulation
  2. Real negative roots: : the collision time is set to a very large value in simulation

Hard sphere model

Event-driven algorithm

- The collision time is a solution of a quadratic equation,
- 1. Complex roots: the collision time is set to a very large value in simulation
  2. Real negative roots: : the collision time is set to a very large value in simulation
  3. Positive and negative roots: the collision time corresponds to the positive value

Hard sphere model

Event-driven algorithm

- The collision time is a solution of a quadratic equation,
- 1. Complex roots: the collision time is set to a very large value in simulation
  2. Real negative roots: : the collision time is set to a very large value in simulation
  3. Positive and negative roots: the collision time corresponds to the positive value
  4. Two positive roots: the smallest one is selected as the collision time.

Hard sphere model

Event-driven algorithm

- The collision time is a solution of a quadratic equation,
- 1. Complex roots: the collision time is set to a very large value in simulation
  2. Real negative roots: : the collision time is set to a very large value in simulation
  3. Positive and negative roots: the collision time corresponds to the positive value
  4. Two positive roots: the smallest one is selected as the collision time.
- A necessary condition for having a positive solution for t (collision in the future) is that

$$(\mathrm{r}_i^0 - \mathrm{r}_j^0).(\mathrm{v}_i - \mathrm{v}_j) < 0$$

Hard sphere model

Event-driven algorithm

- Once all pair of particles have been examined, the shortest time is selected, namely the first collision.

Hard sphere model

Event-driven algorithm

- Once all pair of particles have been examined, the shortest time is selected, namely the first collision.
- The trajectories of particles evolve rectilinearly until the new collision occurs (calculation of trajectories is exact)
- At the collision time, velocities of the two particles are updated, and one calculates the next collision again.