Molecular Dynamics Simulation

Consider classical nuclear motion of N atoms on a given potential energy surface $E_{\mathbf{R}}$.

Newton's equations of motion:

$$\mathbf{f}_I = M_I \ddot{\mathbf{R}}_I \quad \forall \quad I \tag{1}$$

where $\mathbf{f}_I = \mathrm{d}\mathbf{p}_I/\mathrm{d}t$ is the force on atom I, given by $\mathbf{f}_I = -\frac{\partial E_{\mathbf{R}}}{\partial \mathbf{R}_I}$.

Notice:

- Eq. 1 is a system of $3N$ partial differential equations of order 2
- The PES $E_{\mathbf{R}}$ couples all $3N$ nuclear degrees of freedom

$\rightarrow$ No analytic solution!

$\rightarrow$ Eq. 1 must be solved numerically.

Solving Eqs. 1 for molecular systems is generally referred to as Molecular Dynamics (MD) simulation.

Types of MD Simulations

Classical molecular dynamics (CMD): $E_{\mathbf{R}}$ given by an empirical function of nuclear coordinates.

Density functional-based molecular dynamics (DFT-MD): Electronic energy $E_{\mathbf{R}}$ and forces $\mathbf{f}_I$ obtained from density functional theory (DFT) calculations.

Ab-initio molecular dynamics (AIMD): Electronic energy $E_{\mathbf{R}}$ and forces $\mathbf{f}_I$ obtained from ab-initio calculations.

You may come across other advanced MD techniques, which we will not discuss here, e.g.,

Non-adiabatic molecular dynamics (NAMD): Goes beyond the BO approximation, more than one PES, for photo-chemistry and photo-physics

Path-integral molecular dynamics (PIMD): a classical MD technique in an extended phase space to account for nuclear quantum effects

There are two ways of carrying out AIMD or DFT-MD simulations with regard to the calculation of $E_{\mathbf{R}}$ and $\mathbf{f}_I$.

"On-the-fly": Electronic structure calculation of $E_{\mathbf{R}}$ and $\mathbf{f}_I$ is carried out after each update of the nuclear positions according to Eq. 1.

Analytic PES: Electronic structure calculations of $E_{\mathbf{R}}$ and $\mathbf{f}_I$ are carried out for many different nuclear configurations of the system, followed by fit to analytic functions, prior to MD simulation.

On-the-fly AIMD or DFT-MD is nowadays the standard.

Time Stepping algorithms

Newton's equation for interacting many-particle systems (Eq. 1) cannot be solved analytically due to the complicated dependence of the potential energy, $E_{\mathbf{R}}$, on the nuclear degrees of freedom in real systems.
$\rightarrow$ Iterative numerical schemes have to be used.

First step: discretization of time, define time step δt.

Successive equidistant points on the time axis: $t_m = m\delta t$ with $t_0 = 0$,

Evolution of the system described by a time series of coordinate values, termed trajectory:

$$\mathbf{R}(t_0) = \mathbf{R}(0), \ldots, \mathbf{R}(t_{m-1}) = \mathbf{R}(t_m - \delta t), \mathbf{R}(t_m), \ldots, \mathbf{R}(t_{m+1}) = \mathbf{R}(t_m + \delta t) \qquad (2)$$

plus a similar series for the velocities $\dot{\mathbf{R}}$:

$$\dot{\mathbf{R}}(t_0) = \dot{\mathbf{R}}(0), \ldots, \dot{\mathbf{R}}(t_{m-1}) = \dot{\mathbf{R}}(t_m - \delta t), \dot{\mathbf{R}}(t_m), \ldots, \dot{\mathbf{R}}(t_{m+1}) = \dot{\mathbf{R}}(t_m + \delta t) \qquad (3)$$

Three popular algorithms to calculate positions and velocities: Verlet, Velocity Verlet and leap-frog algorithms.

Verlet algorithm

This algorithm is based on a **Taylor series expansion of the coordinates** around *t* forward and backward in time:

$$\mathbf{R}_I(t+\delta t) = \mathbf{R}_I(t) + \dot{\mathbf{R}}_I(t)\delta t + \frac{\mathbf{f}_I(t)}{2M_I}\delta t^2 + \frac{\mathbf{b}_I(t)}{6}\delta t^3 + O(\delta t^4)$$

$$\mathbf{R}_I(t-\delta t) = \mathbf{R}_I(t) - \dot{\mathbf{R}}_I(t)\delta t + \frac{\mathbf{f}_I(t)}{2M_I}\delta t^2 - \frac{\mathbf{b}_I(t)}{6}\delta t^3 + O(\delta t^4)$$

Where we have expressed the second derivative of coordinates in terms of force and mass. Adding the two equations, we obtain:

$$\mathbf{R}_I(t+\delta t) = 2\mathbf{R}_I(t) - \mathbf{R}_I(t-\delta t) + \frac{\mathbf{f}_I(t)}{M_I}\delta t^2 + O(\delta t^4)$$

Note that the accuracy is fourth order in time, one order of magnitude better than the original expansions. Subtracting the two equations, we obtain an expression for the velocities:

$$\dot{\mathbf{R}}_I(t) = \frac{1}{2\delta t}[\mathbf{R}_I(t+\delta t) - \mathbf{R}_I(t-\delta t)] + O(\delta t^3)$$

In the Verlet algorithm one needs to know the positions at time *t-δt* and *t+δt* in order to obtain the velocities At t. That is, the **velocity update is one step behind the position**. This can be inconvenient if when would like to calculate velocity dependent quantities such as kinetic energy.

Velocity Verlet algorithm

This algorithm is a modification of the Verlet agorithm designed so that **positions and velocities are available at the same time step.**

The positions are given by the forward expansion in the previous slide disregarding terms higher than second order:

$$\mathbf{R}_I(t+\delta t) = \mathbf{R}_I(t) + \dot{\mathbf{R}}_I(t)\delta t + \frac{\mathbf{f}_I(t)}{2M_I}\delta t^2 + O(t^3)$$

To obtain the velocities at time *t* we first calculate the force at the new position:

$$\mathbf{f}_I(t+\delta t) = \mathbf{f}_I(\mathbf{R}_I(t+\delta t))$$

Substituting this expression in the Taylor expansion for the positions back in time from $t + \delta t$ to *t*, we find

$$\mathbf{R}_I(t) = \mathbf{R}_I(t+\delta t) - \dot{\mathbf{R}}(t+\delta t)\delta t + \frac{\mathbf{f}_I(t+\delta t)}{2M_I}\delta t^2 + O(t^3)$$

Adding this expression to the forward expansion *t* to $t + \delta t$ given above, we obtain the updated velocities at $t + \delta t$,

$$\dot{\mathbf{R}}_I(t+\delta t) = \dot{\mathbf{R}}_I(t) + \frac{1}{2M_I}[\mathbf{f}_I(t) + \mathbf{f}_I(t+\delta t)]\delta t + O(t^3)$$

Although the appear dissimilar, the **Verlet and Velocity Verlet algorithms are equivalent** and produce exactly the same trajectory.