Molecular Dynamics
Part 2: MD algorithms

Tadeusz Andruniów
Molecular dynamics allows for simulating the motion of each particle as a function of time by using equations of motion and functional form of potential energy (force field).

Equations of motion are integrated using discreet time steps (total time length is divided into small time steps).

To use the finite time-step method the potential must be a continuous function of particle positions (particles must change positions smoothly with time).

If the potential varies sharply there will be a discontinuity in velocities.
Characteristics of MD

- MD is (formally) deterministic - given a state of the system at the present time $t_0$ it is possible to determine the state of the system in the future.

- What about chaos?
  - Lyapunov instability - any integration error, will cause the generated trajectory to diverge from the 'correct' trajectory.

- MD integrators assume that velocities and accelerations are constant over a given time step.
Characteristics of MD

- All particles move in time according to the laws of Newtonian mechanics
  - Classical mechanical treatment of nuclei movement
  - Quantum effects are neglected
- Random initial distribution of particles
- Every new distribution is derived from an old one
- A sequence of distributions of particles formed as time advances = trajectory
Steps in MD simulations

1. Initial conditions (positions and velocities)
2. Compute forces on each atom from energy expression
3. Integrate Newton’s equations of motions - update $r$ and $v$
4. Collect data
5. Compute properties: structural, dynamical, thermodynamic and statistical
Newton’s formalism

System of N particles can be described by 3N independent generalized coordinates $q_i$ and 3N velocities $\dot{q}_i$

$$F_i = m_i a_i = m_i \ddot{r}_i$$

$$F_i = -\frac{\partial}{\partial r_i} U = \nabla U(r_i)$$

$$\begin{cases} 
\dot{r}_i = \frac{p_i}{m_i} \\
\dot{p}_i = F_i 
\end{cases}$$
Lagrange formulation

- Lagrange function is the difference between kinetic and potential energies:

\[ \mathcal{L}(r, \dot{r}) = T(r, \dot{r}) - U(r) \]

\[ \mathcal{L} = \frac{1}{2} \sum_{i=1}^{3N} m_i \dot{r}_i^2 - U(r_i) \]

- Lagrange function is independent of the coordinate system
Equations of motion of a system in the Lagrangian form are:

\[
\frac{d}{dt} \frac{\partial L}{\partial \dot{r}} - \frac{dL}{dr} = 0
\]

Equation of motion has the form of a single-order differential equation.
Hamilton function is the sum of kinetic and potential energies:

\[ \mathcal{H} = T + U \]

Equations of motion can be derived from Hamiltonian:

\[
\begin{cases}
-\frac{\partial \mathcal{H}}{\partial \mathbf{r}_i} = \mathbf{p}_i \\
\frac{\partial \mathcal{H}}{\partial \mathbf{p}_i} = \mathbf{r}_i
\end{cases}
\]
Comparison of different formalisms

\[ \mathcal{H} = \sum_i \dot{r}_i \frac{\partial \mathcal{L}}{\partial \dot{r}_i} - \mathcal{L} \]

- Total energy, linear (and angular) momentum are conserved.

- In Hamilton formulation a coupled set of first-order differential equations is solved whereas in Lagrange formalism it is a single second-order differential equation.

- In both Hamilton and Lagrange formulations any set of nonredundant variables can be used (in Newtonian only Cartesian spatial coordinates and corresponding velocities).
Integration schemes

In all integration schemes (if we assume that a classical trajectory is continuous) positions, velocities, accelerations and higher order derivatives are expanded in Taylor series (for any continuous and differentiable function):

\[ r_i(t + \delta t) = r_i(t) + v_i(t)\delta t + \frac{1}{2!} a_i(t)\delta t^2 + \frac{1}{3!} b_i(t)\delta t^3 + \frac{1}{4!} c_i(t)\delta t^4 + \ldots \]

\[ v_i(t + \delta t) = v_i(t) + a_i(t)\delta t + \frac{1}{2!} b_i(t)\delta t^2 + \frac{1}{3!} c_i(t)\delta t^4 + \ldots \]

\[ a_i(t + \delta t) = a_i(t) + b_i(t)\delta t + \frac{1}{3!} c_i(t)\delta t^2 + \ldots \]

\[ b_i(t + \delta t) = b_i(t) + c_i(t)\delta t + \ldots \]

\( v = \frac{dr}{dt} \) \quad velocity \quad \( a = \frac{d^2r}{dt^2} \) \quad acceleration \quad \( b = \frac{d^3r}{dt^3} \) \quad jerk \quad \( c = \frac{d^4r}{dt^4} \) \quad snap (jounce)
Integration schemes

- From Taylor-based equations it is possible to estimate new values of positions, velocities and accelerations from one time step to another.

- Taylor-based equations will not generate correct trajectories as time advances!

- To do that one has to correct predicted quantities.

- Equations of motion have to be introduced, e.g. through a predictor-corrector algorithm.

- From the predicted new positions $r(t+\delta t)$, having the equation of motion, the forces and then correct accelerations are calculated at time $t+\delta t$. 
Desired features of good algorithms

⇒ fast
⇒ small memory and storage requirements
⇒ easy to implement
⇒ time-reversible
⇒ area-preserving (symplectic)
⇒ reveal small divergence from exact classical trajectory
⇒ conserve energy and momentum
⇒ good stability when using large time steps $\delta t$
Euler algorithm uses positions, velocities and accelerations at time $t$ to compute new positions at time $t + \delta t$

$$\mathbf{r}_i(t + \delta t) = \mathbf{r}_i(t) + \mathbf{v}_i(t) \delta t + \frac{1}{2} \mathbf{\ddot{v}}_i \delta t^2$$

$$\mathbf{v}_i(t + \delta t) = \mathbf{v}_i(t) + \mathbf{\ddot{v}}_i(t) + \mathbf{\dddot{v}}_i(t) \delta t^2$$

- Recently Euler is NEVER used due to energy drifts
- It is no area-preserving and not time reversible
Verlet algorithm

- Verlet uses positions and acceleration at time step $t$ and previous step $t-\delta t$ to update positions at $t+\delta t$

- Taylor expansion around $r_i(t)$ can be written as:

$$r_i(t + \delta t) = r_i(t) + v_i(t)\delta t + \frac{1}{2} \ddot{v}_i(t)\delta t^2 + \frac{1}{6} \dddot{v}_i(t)\delta t^3 + O(\delta t^4)$$

$$r_i(t - \delta t) = r_i(t) - v_i(t)\delta t - \frac{1}{2} \ddot{v}_i(t)\delta t^2 + \frac{1}{6} \dddot{v}_i(t)\delta t^3 + O(\delta t^4)$$

- Add equations for $r_i(t+\delta t)$ and $r_i(t-\delta t)$ and rearrange:

$$r_i(t + \delta t) = 2r_i(t) - r_i(t - \delta t) + \ddot{r}_i(t)\delta t^2 + O(\delta t^4)$$
⇒ Velocity is not used for propagation but can be calculated as:

\[
\mathbf{v}_i(t + \frac{1}{2}\delta t) = \frac{[\mathbf{r}_i(t + \delta t) - \mathbf{r}_i(t)]}{\delta t} \quad \text{or} \\
\mathbf{v}_i(t) = \frac{[\mathbf{r}_i(t + \delta t) - \mathbf{r}_i(t - \delta t)]}{2\delta t}
\]

⇒ Velocities are not needed to generate trajectories but are useful to calculate kinetic energy
Features of Verlet integrator

⇒ The truncation error is of order $O(\delta t^4)$ for positions update while velocities are subject to errors $O(\delta t^2)$

⇒ time-reversible

⇒ good energy conserving properties even for large time steps

⇒ numerically imprecise as small term $O(\delta t^2)$ is added to much larger term $O(\delta t^0)$
Leap-frog

First calculate the velocities at time $t+1/2\delta t$:

$$v_i(t + \frac{1}{2}\delta t) = v_i(t - \frac{1}{2}\delta t)\delta t + \dot{v}_i(t)\delta t$$

use velocities at time $t+1/2\delta t$ to calculate positions at time $t+\delta t$:

$$r_i(t + \delta t) = r_i(t) + v_i(t + \frac{1}{2}\delta t)$$
Leap-frog scheme

1. Velocities leap over the positions to calculate their values at time $t+1/2\delta t$.

2. Positions leap over the velocities to calculate their values at time $t+\delta t$.

3. Velocities at $t+3/2\delta t$ leap over the positions to calculate their values.

4. And so on until positions and velocities for subsequent time steps are found.

Number of time steps = time length of simulations divided by time step $\delta t$. 
Velocities are not calculated at the same time as positions (velocities are known accurately at $t+1/2\delta t$ while positions at $t+\delta t$)

However, velocities at time $t$ can be estimated from:

$$v_i(t) = \frac{1}{2} \left[ v_i(t - \frac{1}{2}\delta t) + v_i(t + \frac{1}{2}\delta t) \right]$$

Improvements over standard Verlet:

- more accurate (no calculation of differences between large and small numbers)
- the same memory requirements as Verlet
- velocities are explicitly calculated in leap-frog scheme
Velocity Verlet yields positions, velocities and accelerations at the same time $t$, thus minimizing round-off errors. Calculate positions at $t+\delta t$:

$$
\mathbf{r}_i(t + \delta t) = \mathbf{r}_i(t) + \mathbf{v}_i(t)\delta t + \frac{1}{2} \dot{\mathbf{v}}_i(t)\delta t^2
$$

Calculate velocities at $t+1/2\delta t$:

$$
\mathbf{v}_i(t + \frac{1}{2}\delta t) = \mathbf{v}_i(t) + \frac{1}{2} \dot{\mathbf{v}}_i\delta t
$$

Calculate accelerations at $t+1/2\delta t$ using previously calculated values of $\mathbf{r}_i(t+\delta t)$ and $\mathbf{v}_i(t+1/2\delta t)$

Calculate velocities at $t+\delta t$:

$$
\mathbf{v}_i(t + \delta t) = \mathbf{v}_i(t) + \frac{1}{2} \left[ \dot{\mathbf{v}}_i(t) + \dot{\mathbf{v}}_i(t + \delta t) \right] \delta t
$$
⇒ Truncation errors are the same as in standard Verlet

⇒ More accurate expressions for velocities are used by Beeman algorithm ($O(\delta t^3)$) and Velocity-corrected Verlet algorithm ($O(\delta t^4)$)
1. Predict $r_i$, $v_i$ and $a_i$ at a time $t+\delta t$ using the current values of these quantities at a time $t$ (from Taylor expressions)

2. Calculate forces, and via equations of motion accelerations, from new positions $r(t+\delta t)$

3. Compare values of $a(t+\delta t)$ obtained from both steps:

$$\Delta a(t + \delta t) = a^c(t + \delta t) - a^{Taylor}(t + \delta t)$$

4. Correct positions, velocities, accelerations and higher order terms using new accelerations $a(t+\delta t)$

5. return to point 1 for the next step
\[ r^c(t + \delta t) = r^{Taylor}(t + \delta t) + c_0 \Delta a(t + \delta t) \]
\[ v^c(t + \delta t) = v^{Taylor}(t + \delta t) + c_1 \Delta a(t + \delta t) \]
\[ a^c(t + \delta t) = a^{Taylor}(t + \delta t) + c_2 \Delta a(t + \delta t) \]
\[ b^c(t + \delta t) = b^{Taylor}(t + \delta t) + c_3 \Delta a(t + \delta t) \]

coefficients determined to maximize the stability of algorithm
\( c_0 = 1/6, c_1 = 5/6, c_2 = 1 \) and \( c_3 = 1/3 \) (Gear)
Predictor-corrector

- both positions and velocities corrected to $\delta t^4$
- accurate for small $\delta t$ (Verlet is more accurate for large $\delta t$)
- not time reversible and not area preserving
- time consuming (two force evaluations per step vs. single force evaluation per step for Verlet)
- high memory requirements (15N vs. 9N for Verlet)
To simulate physical and/or biological phenomena occurring on very long time scales one needs to use time step as large as possible.

**Dilemma?**

- a too small time step increases the cpu time of simulations and prohibits investigations of many long-time processes
- a too large time step results in less accurate (large forces) and less stable (large forces=small interparticles distances) integration procedures
There is always a compromise between the accuracy and the value of the time step.

The largest value of $\delta t$ is determined by the fastest process taking place in the system.

- The fastest motion in most molecular systems is due to vibrational stretching of C-H bonds which has a period of 10 fs (ca. 3000 cm$^{-1}$)

For molecular systems $\delta t$ should be at least 10-fold smaller than the time of the fastest motion $\delta t < 1$ fs.

Time for sampling should be ca. 10 times longer than the timescale of the investigated process.
reduce truncation errors by using higher order integration schemes resulting in the possibility of choosing a larger time step to achieve the same accuracy

freeze high-frequency modes (for example C-H stretching modes) which allows to increase time step (SHAKE and RATTLE algorithms)

use different time steps for different parts of the system oscillating on different time scales (one can also split forces into short and long-range to which different time step may be applied)
The state of the system at a time $t$ is given by:

$$\Gamma(t) = e^{iLt} \Gamma(0)$$

- $\Gamma(r, p, t)$ is a classical mechanical distribution function
- $e^{iLt}$ is a classical time evolution operator

Liouville operator for a molecular system consisted of $N$ atoms in Cartesian coordinates ($3N$ coordinates) is defined as:

$$iL = \sum_{i=1}^{3N} \left[ \dot{r}_i \frac{\partial}{\partial r_i} + F_j \frac{\partial}{\partial p_i} \right]$$
F(x) can be defined as:

$$F(r) = F_{\text{stretch}}(r) + F_{\text{bend}}(r) + F_{\text{dihedral}}(r) + F_{\text{vdw}}(r) + F_{\text{electrostatic}}(r)$$

nonbonded $F(r)$ ($F_{nb}$) can be divided into (e.g. through switching function):

$$F_{nb}(r) = F_{nb}^{\text{short}} + F_{nb}^{\text{long}}$$
Liouville operator can be decomposed into parts:

\[ L = L_1 + L_2 + \ldots \]

and then each Liouville term is associated with specific force, e.g. \( F_{nb}^{short} \) with \( L_1 \) and \( F_{nb}^{long} \) with \( L_2 \)

let \( \delta t_1 \) be the time step employed for calculation of short-range non-bonded forces, then \( \delta t_2 \) which is the time step for long-range non-bonded forces can be written as:

\[ \delta t_2 = n_1 \delta t_1 \]
Constrained dynamics

If the highest frequency motions are removed with holonomic constraints (SHAKE, RATTLE) the time step can be increased by a factor of 2.

Constraints are introduced through Lagrangian (or Hamiltonian):

\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{r}_i} \right) - \frac{\partial L}{\partial r_i} = g_i
\]

where the constraint force is given by:

\[
g_i = \sum_{\alpha=1}^{M} \Lambda_\alpha \left( \frac{\partial \chi_\alpha}{\partial \dot{r}_i} \right)
\]

\(\Lambda_\alpha\) - undetermined Lagrange multiplier associated with \(\alpha\) constraint.
Constrained dynamics

Constraints need to satisfy the following condition:

\[
\begin{align*}
\chi_1(r) &= 0 \\
\vdots \\
\chi_M(r) &= 0
\end{align*}
\]

in case a bond has a fixed length \( \chi \) will be defined as:

\[
\chi(r_1, r_2) = (r_1 - r_2)^2 - b^2 = 0
\]

where \( b \) is the fixed bond length between atoms 1 and 2
SHAKE uses the Verlet (leap-frog) algorithm for unconstrained particles:

\[ \mathbf{r}_i'(t + \delta t) = 2\mathbf{r}_i(t) - \mathbf{r}_i(t - \delta t) + \frac{\delta t^2}{m_i} \mathbf{f}_i \]

after imposing constraints:

\[ \mathbf{r}_i(t + \delta t) = 2\mathbf{r}_i(t) - \mathbf{r}_i(t - \delta t) + \frac{\delta t^2}{m_i} (\mathbf{f}_i - \mathbf{g}_i) \]

nonlinear equation for \( \Lambda \) multiplier has to be solved:

\[ \mathbf{r}_i(t + \delta t) = \mathbf{r}_i'(t + \delta t) - \frac{\delta t^2}{m_i} \sum_{\alpha=1}^{M} \Lambda_\alpha \left( \frac{\partial \chi_\alpha(\mathbf{r}(t))}{\partial \mathbf{r}_i} \right) \]

choosing \( \Lambda \) such that \( \chi(\mathbf{r}(t+\delta t)) = 0 \)
SHAKE cycles through constraints adjusting positions iteratively until all constrained equations are satisfied (to within a specified tolerance)

RATTLE uses the Velocity Verlet algorithm and two sets of Lagrange multipliers

In SHAKE only positions are restrained, in RATTLE both positions and velocities
Periodic boundary conditions

- In small systems surface effects can dominate
- The number of particles close to the surface is ca. $6N^{2/3}$ and the ratio of surface particles to bulk particles is $6N^{-1/3}$
  - for a system consisting of 1000 water molecules ca. 60% of them will be at the surface (even for $10^6$ water molecules surface effects are still non-negligible with a surface fraction of 6%)
- This will introduce artifacts
Periodic boundary conditions

- It is not possible to simulate the number of atoms contained in macroscopic samples of substances (of the order of $10^{23}$ particles).

- In simulations the ratio between the boundary atoms and the total number of atoms will always be much larger than in reality resulting in much more pronounced surface effects.

- A solution to the boundary effects caused by finite-size is to use periodic boundary conditions.
Issues with periodic boundary conditions

- Upon introducing periodic boundary conditions particles are placed in a box.
- The box is replicated throughout space by translations in all directions.
- The central box is surrounded by 26 copies of the original box.
- When a particle leaves the central box its image (from the neighboring box) will simultaneously enter the central box from the other side.
- The number of particles in the central box and thus in the entire system is conserved.
Issues with periodic boundary conditions

- Arbitrary shift of the cell boundaries has no effect on the system
- PBC eliminates surface effects - no boundaries=no surface problem!
- A particle from central box will interact not only with other particles in this box but also with their images in neighboring boxes
- As a result the number of pair-wise interactions vastly increases
Issues with periodic boundary conditions

- PBC does not eliminate artifacts caused by a small number of particles
  - the simulation box should be large enough not to generate additional artifacts

- for some physical processes, e.g. adsorption onto a surface (motion perpendicular to the surface), standard PBC cannot be used in all directions

- PBC will not work properly if the cell size is smaller than the distance over which the interactions act (e.g. liquid-gas critical point)
What should be the shape of the box?

- Any space-filling shape:
  - cube
  - hexagonal prism
  - truncated octahedron
  - rhombic dodecahedron
  - elongasted dodecahedron

It is suggested to choose the periodic cell that reflects most closely the shape of the system.
Minimum image convention

- only the closest image (each atom interacts with at most one image of every atom in the system) is considered in calculations of energy and forces

- interactions between all pairs of atoms (including images) that are further apart than the cut-off distance are set to zero

- the largest value of the cut-off may not exceed half the length of the box (when using a cubic box; for a rectangular box it should not be greater than half the length of the shortest side)
  
  - the particle cannot see its own image otherwise duplicate forces are calculated and simulations are erroneous
Problems with minimum image convention

- the nearest image is not always energetically favourable
- splitting charged parts of molecules
- suppress long-range correlations
- new artificial correlations
Look-up tables

- precompute the values of non-bonded potential for different distances and store them in the form of a table
- these values are then used during MD simulations to evaluate the potential through interpolation schemes
Finding all interacting pairs is a time-consuming process for biomolecules $O(N^2)$.

Not all of these interactions are relevant and cut-off techniques and switching/shifting potentials are utilized.

However, it still may be a daunting task to calculate the list of neighbors.

Efficient algorithms, such as cell list and Verlet neighbor list, are utilized.
define neighbor cut-off radius as potential cut-off radius and 'skin' radius \((r_L = r_c + r_{skin})\)

update list whenever atoms move more than \(r_{skin}\)

the list update is \(O(N^2)\) so where is the computational-cost gain?

particles are assumed to move slowly so the list will have to be updated only occasionally

for solids the list may not need to be updated ever while for liquids very seldom

large memory requirements (size of the list is \((4\pi(r_c+r_{skin})\cdot\rho\cdot N/3)\)
Cell list

- divide the volume into a set of cells of equal size with an edge length of at least $r_c$

- prepare the list of atoms in each cell

- interaction is computed between particles in the same and adjacent cells

- as the interaction between particle $i$ and $j$ is the same as between particle $j$ and $i$ only half of the neighboring cells needs to be considered
computational cost is reduced from $O(N^2)$ to $O(N)$

with cells having an edge length of $r_c$ the algorithm is highly inefficient (84% of all calculated pair-wise interactions are unnecessary)

choosing the cell edge-length of $r_c/2$ improves the efficiency (63% of spurious distance evaluations)
Ensembles in MD

In MD one is interested to study properties of systems as a function of temperature and pressure rather than volume and energy.

- microcanonical (NVE)
- grand-canonical (\(\mu VT\))
- canonical (NVT)
- isothermal-isobaric (NPT)
Equipartition energy theorem relates the temperature to the average kinetic energy of the system:

\[ < K > = \frac{3}{2} N k_B T \]

Instantaneous temperature is expressed by:

\[ K(t) = \frac{3}{2} N k_B T(t) \]

In some cases the number of degrees of freedom for N particles will be 3N-6 as center of mass velocity and angular momentum are constant.
For pair-wise potentials the pressure is related to the sum of the products of the particles positions and forces acting on them. The calculation of the pressure in MD is based on virial theorem of Clausius.

\[
PV = \left( N k_B T \right) - \frac{1}{3} \left\langle \sum_{i}^{N} \sum_{j>i}^{N} \mathbf{r}_{ij} \mathbf{f}_{ij} \right\rangle
\]

\( \mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j \)

and \( \mathbf{f}_{ij} \) is the force on particle \( i \) due to particle \( j \).
The velocity is simply scaled at each time step:

\[ v_i^{\text{new}} = c_t v_i^{\text{old}} \]

where

\[ c_t = \sqrt{\frac{T_0}{T}} \]

This procedure leads to rapid energy transfer between different degrees of freedom of the particle. Resetting of velocities may be required from time to time.
To maintain the temperature the system is coupled weakly to an external heat bath with a fixed temperature ($T_0$).

The equation of motion is modified to obtain the desired temperature

$$\dot{r}_i(t) = \dot{v}_i(t)$$

$$m_i \ddot{v}_i(t) = \dot{F}_i(t) - \gamma(t) m_i \dot{v}_i(t)$$

$$\gamma(t) = \frac{1}{2\tau} \left(1 - \frac{T_0}{T(t)}\right)$$
Berendsen thermostat

The velocities are scaled at each step by the scaling factor $c_t$:

$$c_t = \sqrt{1 - \frac{\delta t}{\tau} \left(1 - \frac{T_0}{T(t)}\right)}$$

where
- $\gamma$ - heat flow variable
- $T$ - instantaneous kinetic temperature
- $T_0$ - thermostat temperature
- $\tau$ - time constant of the coupling to the heat bath
The value of $\tau$ reveals the strength of coupling between the system and the bath. To achieve good temperature control this value has to be chosen with care.

- when $\tau \to \infty$ Berendsen thermostat does not work (sampling microcanonical ensemble)
- when $\tau = \delta t$ - Berendsen thermostat turns into a simple scaling scheme
- when $\tau$ is very small the coupling to the heat bath is large and there is a significant energy exchange between the system and the thermal bath
- efficient
- does not lead to the canonical distribution
To obtain constant pressure (under assumptions of PBC, finite volume, isotropic system) Berendsen proposed:

- scaling of Cartesian coordinates:

\[ r_{i}^{\text{new}} = c_p r_{i}^{\text{old}} \]

- scaling of volume
The scaling factor is expressed as:

\[ c_P = \left[ 1 - \frac{\beta \Delta t}{\tau} (P_0 - P) \right]^{1/3} \]

where
- \( \beta \) - isothermal compressibility
- \( P \) - instantaneous pressure
- \( P_0 \) - target pressure
- \( \tau \) - pressure coupling time
The equation of motion can be written as:

$$\dot{r}_i(t) = v_i(t) - \left(\frac{\beta(P_0 - P(t))}{3\tau}\right)r_i(t)$$

Pressure is kept constant while the volume of the system is allowed to fluctuate by changing the cell size uniformly but not its shape (for isotropic system) does not lead to canonical distribution.

For this reason the Langevin piston method is mostly used in MD simulations!
The Andersen approach couples velocity of a particle to a heat bath via stochastic collisions.

Procedure of Andersen algorithm:

1. Starting from initial conditions, integrate the equations of motion for $\delta t$.
2. Particles that undergo collisions with the heat bath are selected. The probability of selecting a particle during the integration time step $\delta t$ is $v\delta t$, where $v$ is the probability of collision.
3. New velocities for selected particles are chosen at random from Maxwell-Boltzmann distribution at the desired temperature; all other particles remain unaffected.

Generate canonical ensemble:

Less realistic dynamics and thus not used for computing dynamic quantities (e.g. diffusion coefficients, lifetime of water hydrogen bonds).
Nose-Hoover proposed that the bath is an integral part of the system by adding a fictitious degree of freedom to the physical system. As a result, the system does possess $6N+1$ degrees of freedom in $\Gamma$ space.

\[
\dot{r}_i(t) = \dot{v}_i(t) \\
m_i\dot{v}_i(t) = F_i(t) - \zeta(t)m_i\dot{v}_i(t)
\]
The heat flow variable has its own equation of motion enabling fluctuations of temperature:

\[ m_t \dot{\zeta}_t(t) = 2 v^T m v - g k_B T_0 \]

where

- \( \zeta_t \) - heat flow variable (thermodynamic friction coefficient)
- \( m_t \) - fictitious mass
- \( g \) - number of degrees of freedom in the system
System Hamiltonian is conserved:

\[ H(\mathbf{r}_i, \mathbf{p}_i, \zeta) = K(\mathbf{p}_i) + U(\mathbf{r}_i) + \frac{1}{2}(m_t\zeta_t^2) + \frac{3}{2}Nk_b T_0 x_t \]

**Nose-Hoover:**
- allows the temperature to fluctuate about the average value and controls the oscillations of temperature
- may oscillate for a system far from equilibrium thus it is not recommended for the initial stages of simulations
- leads to canonical distribution
Nose-Hoover barostat

- volume of the system is not constant but instead is allowed to fluctuate under constant pressure

- an approach analogous to Nose-Hoover thermostat for NVT ensemble is used for describing Nose-Hoover barostat (NPT ensemble)

- two sets of variables are used: \((x_p, m_p, \zeta_p)\) associated with thermostat and \((x_t, m_t, \zeta_t)\) associated with barostat

- the algorithm drives the system to the state where the average internal pressure is equal to the applied external pressure.
In experiments an extremely large number of particles are sampling conformational space very effectively (the value of a measured property is an average value over time of the measurement).

\[
\overline{A} = \lim_{T \to \infty} \frac{1}{T} \int_{t=0}^{T} A(p^N(t), r^N(t)) \, dt
\]

In molecular simulations the value of A is obtained as time average on the trajectory:

\[
\overline{A} = \lim_{n_t \to \infty} \frac{1}{n_t} \sum_{t=0}^{n_t} A(p^N(t), r^N(t))
\]

where \( n_t \) is the number of time-steps.
Can MD predict thermodynamic quantities?

- Ergodic hypothesis links conformations of the particle sampled by MD over simulation time with the conformations of the particle in a thermodynamic ensemble.

- In other words, the ergodic hypothesis assumes that temporal average value of $A$ ($\bar{A}$) of an equilibrium system is equal to the ensemble average value of $A$ ($\langle A \rangle$).

- Therefore, in MD simulations:

$$\bar{A} = \lim_{n_t \to \infty} \frac{1}{n_t} \sum_{t=0}^{n_t} A(p^N(t), r^N(t)) = \lim_{M \to \infty} \frac{1}{M} \sum_{i=1}^{M} A_i(p^N, r^N) = \langle A \rangle$$

$M$ is the number of configurations
Is the ergodic hypothesis true?

► For most cases no, as the sampling time does not approach infinity.

► However, it is true for special cases where barriers between states are on the order of RT.
Examples of time averages in MD

Average potential energy:

\[ \overline{U} = \frac{1}{M} \sum_{i=1}^{M} U_i \]

Average kinetic energy:

\[ \overline{K} = \frac{1}{M} \sum_{j=1}^{M} \left[ \sum_{i=1}^{N} \frac{m_i}{2} v_i v_j \right] \]

where M is the # of configurations and N is the # of atoms
Autocorrelation functions

Time-dependent autocorrelation functions in MD measure the correlation between particle quantity at time $t$ and time zero. A general expression for autocorrelation function:

$$C_{AA}(t) = \frac{1}{N} \sum_{i=1}^{N} \frac{\langle A_i(t)A_i(0) \rangle - \langle A_i(0) \rangle^2}{\langle A_i(0)A_i(0) \rangle - \langle A_i(0) \rangle^2}$$

Autocorrelation functions (ACFs) can be defined and calculated for any particle quantity (e.g. $v_i$) or any system quantity (e.g. $U$, $T$, $P$, $\rho$).

The initial value of ACF at $t=0$ is 1 and it approaches 0 as $t \to \infty$ (usually the decay of ACFs is exponential and the relaxation time is the time required for the function to approach zero).
To calculate an autocorrelation function of any property the relaxation time of this property must be much shorter than the simulation time.

In most cases this condition is fulfilled and many sets of data can be extracted from trajectory (# of independent data sets to be used in computation of ACF is equal to the # of time steps comprising the total simulation time minus the # of time steps needed to reach the relaxation time) and average over to calculate the correlation function.

The more data sets we use in calculations of ACF the more statistically precise value of ACF can be determined.
Velocity autocorrelation function

\[ C_{vv}(t) = \frac{1}{NM} \sum_{j=1}^{M} \sum_{i=1}^{N} \frac{\langle v_i(t_j)v_i(t_j + t) \rangle}{\langle v_i(t_j)v_i(t_j) \rangle} \]

sum over M goes through different time origins \((t_j)\)
Dipole autocorrelation function

Fourier transform of the dipole autocorrelation function is related to the far-IR spectrum of the solvent.

\[ I(\omega) = \int_{-\infty}^{+\infty} \langle \mu(t)\mu(0) \rangle e^{i\omega t} dt \]
Dynamical properties

Diffusion coefficient:

\[
D = \lim_{t \to \infty} \frac{1}{6Nt} \left\langle \sum_{i=1}^{N} \left[ \mathbf{r}_i(t) - \mathbf{r}_i(0) \right]^2 \right\rangle
\]

Formula based on velocity autocorrelation function:

\[
D = \frac{1}{3N} \int_0^{\infty} \left\langle \sum_{i=0}^{N} \mathbf{v}_i(t) \mathbf{v}_i(0) \right\rangle dt
\]
Dynamical properties

Thermal conductivity

\[ \lambda = \lim_{t \to \infty} \left( \frac{1}{6kT^2 Vt} \right) \langle \sum_{\alpha} \left[ \sum_{i} r_{\alpha i}(t)e_i(t) - \sum_{i} r_{\alpha i}(0)e_i(0) \right]^2 \rangle \]

\( e_i \) is the instantaneous excess energy of particle \( i \)

\( \alpha \) stands for spatial coordinates
Dynamical properties

Formula based on autocorrelation function:

$$\lambda = \frac{V}{3kT^2} \int_{0}^{\infty} \langle S(t)S(0) \rangle dt$$

where

$$S = \frac{1}{V} \left[ \sum_{i} e_{i}v_{i} + \frac{1}{2} \sum_{ij} r_{ij}(f_{ij}v_{ij}) \right]$$
Radial distribution function (radial pair correlation function) describes the probability of finding a particle within a distance range of \( r \) and \( r+dr \) from another particle relative to the ideal gas distribution. Radial distribution function \( g(r) \) can be written as:

\[
g(r) = \frac{1}{N} \frac{dn(r)}{4\pi r^2 dr \rho}
\]

\( 4\pi r^2 dr \) is the volume of a spherical shell of thickness \( dr \) at a distance \( r \) from a chosen atom

\( \rho \) is an ideal gas density
Radial distribution function

\( dn(r) \) is the number of particles between \( r \) and \( r + \text{d}r \) from another particle:

\[
    dn(r) = \sum_{i}^{N} \sum_{i \neq j}^{N} \delta(r - r_{ij})
\]

\( g(r) \) is larger/smaller than 1 when the probability of finding a particle within a distance range of \( r \) and \( r + \text{d}r \) from another particle is larger/smaller than that expected from a completely uniform distribution (e.g. an ideal gas).

In the ideal gas \( g(r) = 1 \).
g(r) is normalized in the following way:

\[
\frac{1}{N} \int_{0}^{\infty} \rho g(r) 4\pi r^2 dr = 1
\]

It is important to choose the right bin size otherwise the results will be meaningless:

- too large \(dr\) leads to low resolution
- too small \(dr\) leads to large statistical errors

g(r) can be obtained from experimental studies:

- in a crystal directly from X-ray
- in solution from neutron diffraction by Fourier transform of structure factor