Computational Physics – Lecture 8: Molecular dynamics I

Kristel Michielsen
Institute for Advanced Simulation
Jülich Supercomputing Centre
Research Centre Jülich
k.michielsen@fz-juelich.de
http://www.fz-juelich.de/ias/jsc/qip

Contents

- Molecular dynamics
  - Definition
  - History
  - Some applications
- Statistical mechanics
  - Macroscopic and microscopic state
  - Ensembles
  - Ensemble average, time average and ergodic hypothesis
- Classical mechanics
  - Newton equation of motion
  - Hamilton equation of motion
  - Integration algorithms
    - Discretization in time
    - Accuracy, efficiency, conservation laws, stability
    - Solving differential equations
      - Runge-Kutta method
      - Predictor-corrector method

Molecular dynamics

The Art of Molecular Dynamics Simulation, D.C. Rapaport, Cambridge University Press (1995)

Simulating the Physical World, H. J. C. Berendsen, Cambridge University Press (2007)

- Previous lecture: Monte Carlo (MC), a stochastic probabilistic simulation method, i.e. "time" progression is random
- Molecular dynamics (MD): a deterministic simulation method, i.e. the future state of the system is completely determined by its present state
- Central idea of MD: Solve Newton's equations of motion (classical !) for a system of particles which interact via a potential energy function (force field)

Molecular dynamics

History

- Introduced by Alder and Wainwright in the late 1950's to study the interaction of hard spheres [B.J.Alder T.E. Wainwright, J. Chem. Phys. 27, 1208 (1957) & B.J. Alder and T.E. Wainwright, J. Chem. Phys. 31, 459 (1959)]
- 1964, Rahman: first simulation using a realistic potential for liquid argon
- 1974, Stillinger and Rahman: simulation of liquid water
- 1977: first protein simulations

Molecular dynamics
Some applications

- (Bio) chemistry: molecular structures, drug design, dynamics of macromolecules such as proteins and DNA, …
- Statistical mechanics and physics: theory of liquids (water), phase transitions, properties of statistical ensembles, …
- Materials science: defects in crystals and their interactions, melting, film growth, microscopic mechanisms of fracture, …

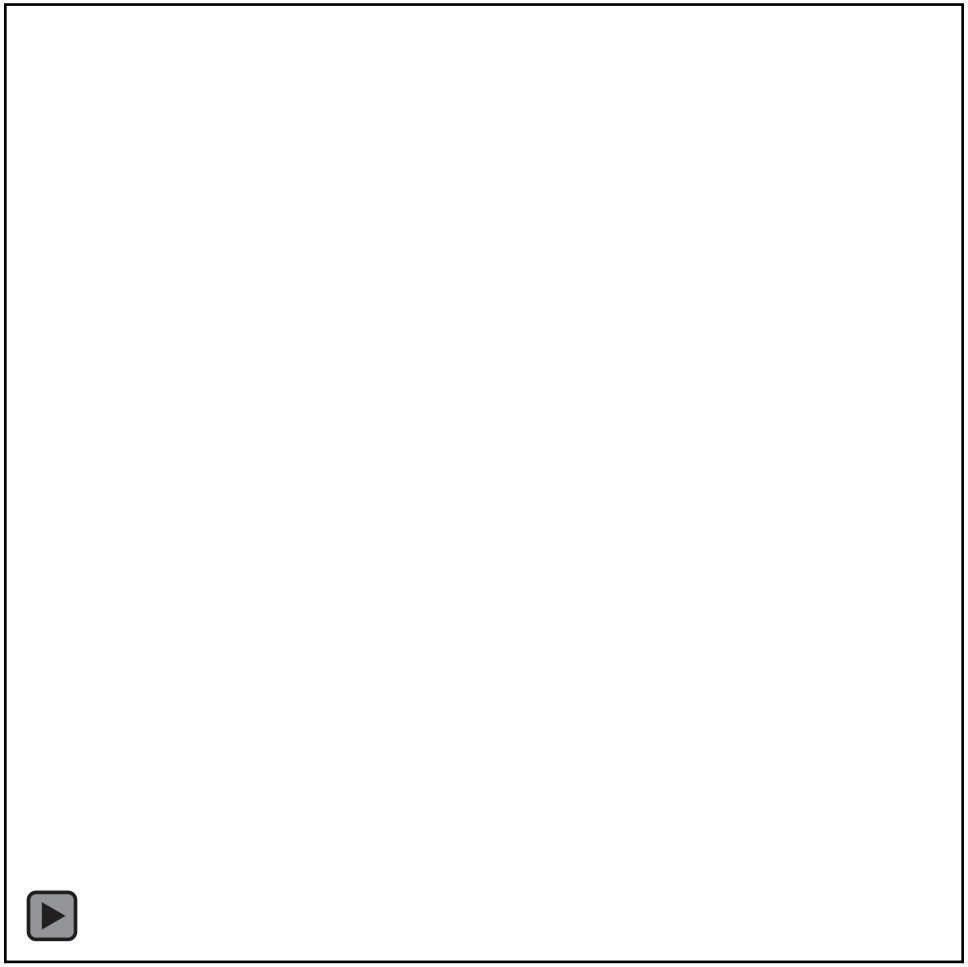
Molecular dynamics
Some applications

- Example materials science: Study of the nucleation and dynamics of dislocations during nanoindentation of a (111) FCC plane

Molecular dynamics

Some applications

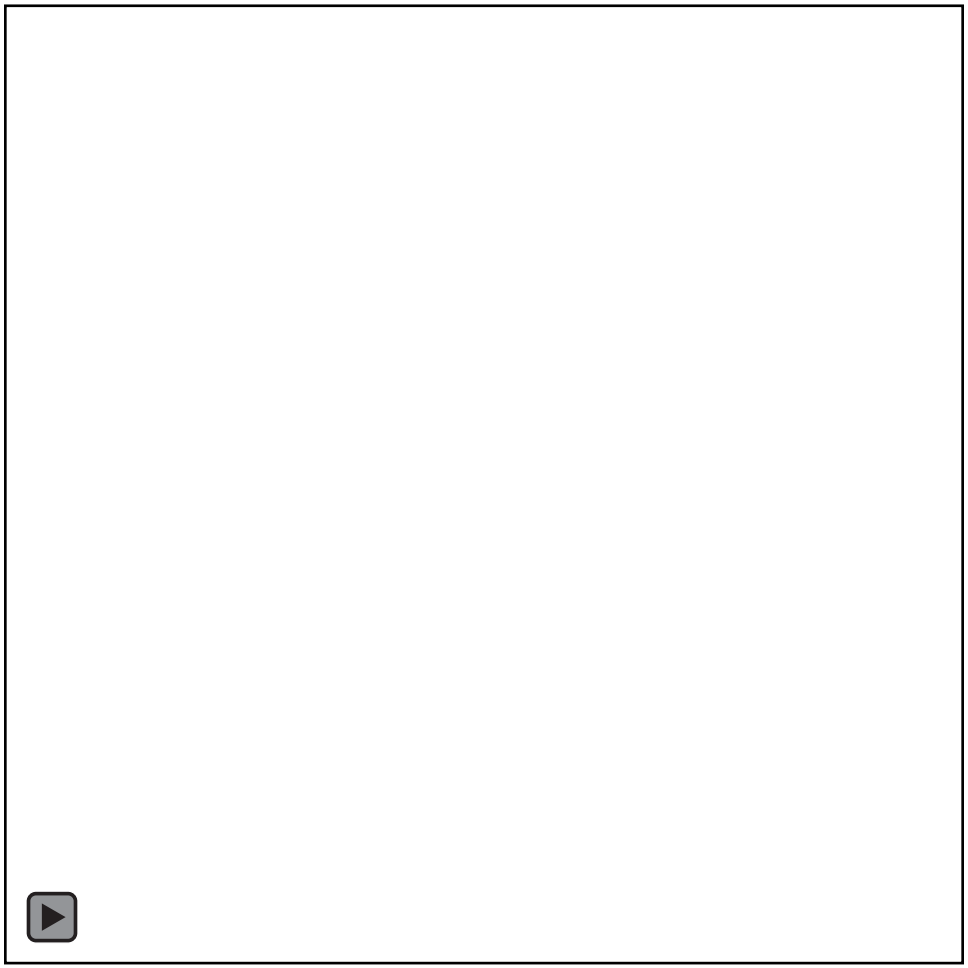
Pure Al

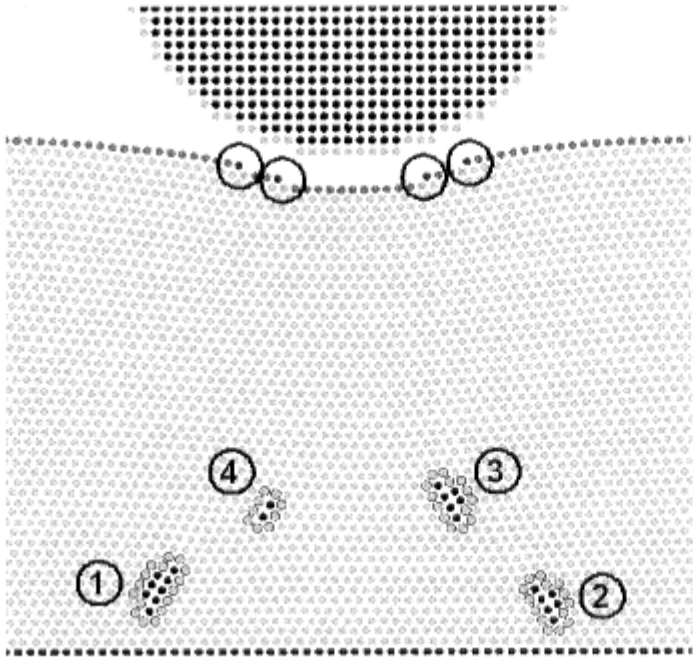


Molecular dynamics

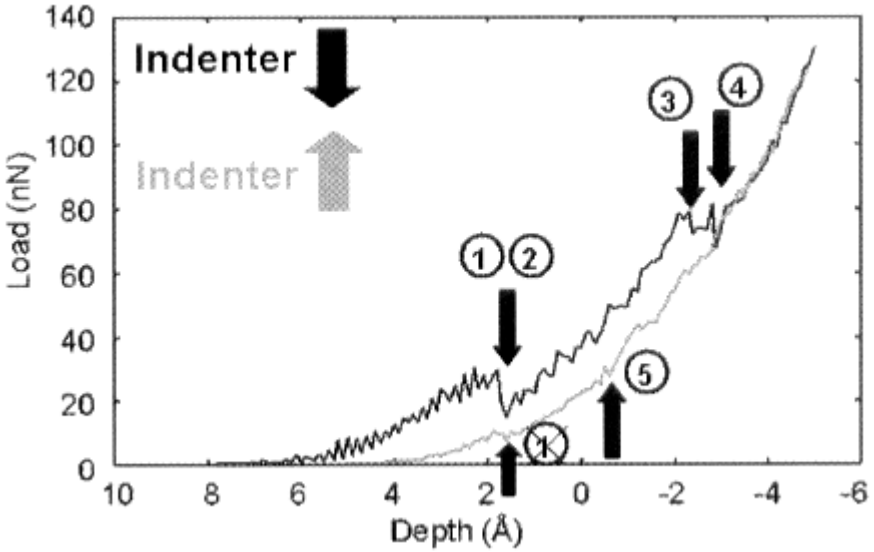
Some applications

Al + Mg





Model parameters
$m = 9.298810^{-26}\text{kg}$
$\epsilon = 1\text{eV}$
$\sigma = 0.15\text{nm}$
$T = 50$ K
$R_c = 1.5$ nm
$\Delta t = 2.5$ fs
$d_t = 6.7$ nm
$\epsilon' = 10$ eV
$\sigma' = 0.4$ nm
$y_t = 0.01$ nm
$N_t = 800$
$v = 5$ m/s
$h_b = 1$ nm
$h_e = -0.5$ nm



Molecular dynamics
Statistical mechanics

- MD simulations generate information at the microscopic level: particle positions and velocities
- Use statistical mechanics to convert this microscopic information to macroscopic observables such as energy, specific heat, …

Molecular dynamics
Statistical mechanics

- Thermodynamic (macroscopic) state: defined by small set of parameters, e.g. pressure P, temperature T and number of particles N
- Microscopic state: defined by particle positions $\mathbf{r}^N$ and momenta $\mathbf{p}^N$, also to be considered as coordinates of points $(\mathbf{r}^N, \mathbf{p}^N)$ in $6N$ -dimensional phase space
  - a single point in phase space describes the state of the system

Molecular dynamics
Statistical mechanics

- Ensemble: collection of points in phase space satisfying the conditions of a particular thermodynamic state
  - Microcanonical ensemble (N, V, E): isolated system
  - Canonical ensemble (N, V, T): system in thermal equilibrium with a heat bath
  - Grand canonical ensemble (μ, V, T): open system in the sense that it can exchange energy and particles with a reservoir

Molecular dynamics
Statistical mechanics

- Average values of an observable A: defined as ensemble averages, averages taken over a large number of replicas of the system considered simultaneously

$$\langle A \rangle_{\text{ensemble}} = \iint d\mathbf{r}^N d\mathbf{p}^N A(\mathbf{r}^N, \mathbf{p}^N) \rho(\mathbf{r}^N, \mathbf{p}^N)$$

The probability density of the ensemble is given by (in units of $k_B = 1$)

$$\rho(\mathbf{r}^N, \mathbf{p}^N) = \exp\left[-H(\mathbf{r}^N, \mathbf{p}^N)/T\right]/Z$$

Molecular dynamics
Statistical mechanics

where H is the Hamiltonian and Z is the partition function

$$Z = \iint d\mathbf{r}^N d\mathbf{p}^N \exp\left[-H(\mathbf{r}^N, \mathbf{p}^N)/T\right]$$

- extremely difficult to calculate since all states of the system need to be calculated

- MD simulation: generates a sequence of points in phase space as a function of time (phase trajectories) → to calculate an ensemble average the MD simulation must pass through all possible states corresponding to the particular thermodynamic constraints

Molecular dynamics
Statistical mechanics

- Time average:

$$\langle A \rangle_{\text{time}} = \lim_{T\to\infty} \frac{1}{T} \int_0^T A(\mathbf{r}^N(t), \mathbf{p}^N(t)) dt$$

$$\approx \frac{1}{K} \sum_{k=1}^{K} A(\mathbf{r}^N(t_k), \mathbf{p}^N(t_k))$$

where K is the number of time steps Δt in the simulation ($T = K\Delta t$) and $t \to t_k = k\Delta t$

Molecular dynamics
Statistical mechanics

- Dilemma:
  - MD simulations → time averaging
  - Experimental observable → ensemble averaging (assumption)

  → Ergodic hypothesis: $\langle A \rangle_{\text{ensemble}} = \langle A \rangle_{\text{time}}$
  - Basic idea: If the system evolves in time infinitely, then the system will eventually pass through all possible states
    However, simulations have a fixed duration → phase space must be sampled properly

Molecular dynamics

Classical mechanics

Simulating the Physical World, H. J. C. Berendsen, Cambridge University Press (2007)

- Classical, Newtonian dynamics of interacting particles in an external field
- Equation of motion $m_n \mathbf{a}_n = \mathbf{F}_n$
  - Mass of particle n : m_n
  - Acceleration of particle n : $\mathbf{a}_n = \dfrac{d^2 \mathbf{r}_n}{dt^2}$
  - Force acting on particle n : $\mathbf{F}_n$
- Notation: $\mathbf{X} = (\mathbf{X}_1, \ldots \mathbf{X}_N)$ with $\mathbf{X}_n = (X_{n,x}, X_{n,y}, X_{n,z})$ for $n = 1, \ldots, N$

Molecular dynamics
Classical mechanics

- If the force acting on each particle is known, then the acceleration of each particle can be determined
- Integration of the equations of motion yields a trajectory that describes the positions, velocities and accelerations of the particles as they vary with time.
- From this trajectory, the average values of properties can be determined.

Molecular dynamics
Classical mechanics

- Equation of motion: Alternative, equivalent formulation (Hamilton):

$$\frac{dr_{n,\alpha}}{dt} = \frac{\partial H}{\partial p_{n,\alpha}} \quad , \quad \frac{dp_{n,\alpha}}{dt} = -\frac{\partial H}{\partial r_{n,\alpha}} \quad , \quad H(\mathbf{r},\mathbf{p}) = \sum_{n=1}^{N} \frac{\mathbf{p}_n^2}{2m_n} + V(\mathbf{r}) \quad , \quad \mathbf{p}_n = m_n \mathbf{v}_n$$

$$\frac{dr_{n,\alpha}}{dt} = \frac{p_{n,\alpha}}{m_n} \quad , \quad \frac{dp_{n,\alpha}}{dt} = -\frac{\partial V(\mathbf{r})}{\partial r_{n,\alpha}} = F_{n,\alpha}(\mathbf{r}) \qquad \alpha = x, y, z$$

  - A set of N second-order differential equations can be rewritten as a set of $2N$ first-order differential equations

Molecular dynamics
Classical mechanics

- Solving Newton's equations of motion
  - Potential (function of all particle positions) has a complicated structure → no analytical solution to the equations of motion; they must be solved numerically (numerical integration)
  - Only simple algorithms really work
  - But the algorithms should not be too simple…

Molecular dynamics
Integration algorithms

- Discretization (in time)

$$\frac{d\mathbf{r}_n}{dt} \approx \frac{\mathbf{r}_n((k+1)\Delta t) - \mathbf{r}_n(k\Delta t)}{\Delta t} = \mathbf{v}_n(k\Delta t) \quad \Rightarrow \quad \mathbf{r}_n(\overbrace{(k+1)\Delta t}^{t+\Delta t}) = \mathbf{r}_n(\overbrace{k\Delta t}^{t}) + \Delta t \mathbf{v}_n(k\Delta t)$$

$$\frac{d\mathbf{p}_n}{dt} \approx \frac{\mathbf{p}_n((k+1)\Delta t) - \mathbf{p}_n(k\Delta t)}{\Delta t} = \mathbf{F}_n(\mathbf{r}(k\Delta t)) \Rightarrow \quad \mathbf{v}_n((k+1)\Delta t) = \mathbf{v}_n(k\Delta t) + \frac{\Delta t}{m_n}\mathbf{F}_n(\mathbf{r}(k\Delta t))$$

  - Given $\mathbf{r}_n(0)$ and $\mathbf{v}_n(0)$, we can compute $\mathbf{r}_n(\Delta t)$ and $\mathbf{v}_n(\Delta t)$ and so on
- Approximation introduces deviations from the real trajectory at each time step → finally the system can be driven toward wrong directions in the phase space

Molecular dynamics
Integration algorithms

- A good algorithm for MD simulations must at least have the following properties:
  - accuracy: the trajectory should obey the equations of motion to good approximation
    - decreasing the time step improves the accuracy
      - because of the exponential rate of separation of near-by trajectories in phase space (Lyapunov instability) this is of limited help
  - efficiency: simple, fast, and little memory usage
    - number of force evaluations
    - maximum time step size
  - conservation laws
    - time reversal symmetry, inherent in Newton's equations of motion
    - energy and momenta conservation
    - conservation of phase space volume (symplectic)
  - stability: the algorithm may not show an energy drift

Lyapunov instability

- The stability of solutions of differential equations describing dynamical systems near to a point of equilibrium may be discussed by the theory of Lyapunov
- In simple words: if trajectories that start out near an equilibrium point stay near this point forever, then this equilibrium point is Lyapunov stable

Symplectic algorithm

- The exact solution of Hamilton equations of motion

$$\frac{dr_{n,\alpha}}{dt} = \frac{\partial H}{\partial p_{n,\alpha}} \quad , \quad \frac{dp_{n,\alpha}}{dt} = -\frac{\partial H}{\partial r_{n,\alpha}}$$

  has the following properties

  - The mapping from $(\mathbf{r},\mathbf{p})$ at $t=0$ to $(\mathbf{r}',\mathbf{p}')$ at $t=T$ along the solution is
    - Exactly symplectic, i.e. $\sum d\mathbf{r}d\mathbf{p} = \sum d\mathbf{r}'d\mathbf{p}'$ or in practice, the mapping

$$\begin{pmatrix}\mathbf{r}'\\ \mathbf{p}'\end{pmatrix} = M\begin{pmatrix}\mathbf{r}\\ \mathbf{p}\end{pmatrix}$$ is symplectic if $\det M = 1$

    - Conserves the energy $H(\mathbf{r},\mathbf{p}) = H(\mathbf{r}',\mathbf{p}')$
- Symplectic algorithms preserve global stability

Solving differential equations: General

- Runge-Kutta method (for details see e.g. https://lpsa.swarthmore.edu/NumInt/NumIntIntro.html)
  General: solve the differential equation

$$\frac{dx}{dt} = f(x,t); \quad x(t_0) = x_0$$

→ solve for the trajectory $x(t)$ numerically, given the initial point $x(t_0)$ at time $t = t_0$

– replace the derivative by finite differences: $dt \to \Delta t, dx \to x(t+\Delta t) - x(t)$ which results in

$$x(t+\Delta t) = x(t) + f(x(t),t)\Delta t + O((\Delta t)^2)$$

Local error

First order Runge-Kutta method = Euler method

Solving differential equations: General

- local error = local truncation error after one time step
  - meaning of $O((\Delta t)^k)$: if $A = O((\Delta t)^k)$ then $\lim_{\Delta t \to 0} A/(\Delta t)^k$ is finite. However, no indication for the size of Δt
- global error that results after a time T, requiring $K = T/\Delta t$ time steps

$$x(t_0 + \Delta t) = x(t_0) + f(x(t_0), t_0)\Delta t + O((\Delta t)^2)$$

$$x(t_0 + 2\Delta t) = \left[x(t_0) + f(x(t_0), t_0)\Delta t + O((\Delta t)^2) \right] + f(x(t_0 + \Delta t), t_0 + \Delta t)\Delta t + O((\Delta t)^2)$$

$$= x(t_0) + \left(f(x(t_0), t_0) + f(x(t_0 + \Delta t), t_0 + \Delta t) \right) \Delta t + 2O((\Delta t)^2)$$

$$\vdots$$

$$x(t_0 + K\Delta t) = x(t_0) + \sum_{k=0}^{K-1} f(x(t_0 + k\Delta t), t_0 + k\Delta t)\Delta t + KO((\Delta t)^2)$$

$$\downarrow$$

$$O(T\Delta t)$$

First order in Δt
Error grows with T
→ drift

Solving differential equations: General

- Runge-Kutta method
  - improve accuracy: evaluate the function f at more points

$$k_1 = f(x(t), t)\Delta t$$

$$k_2 = f\left(x(t) + \frac{k_1}{2}, t + \frac{\Delta t}{2}\right)\Delta t$$

$$x(t + \Delta t) = x(t) + k_2 + O((\Delta t)^3)$$

Second order Runge-Kutta method = midpoint method

Solving differential equations: General

- Runge-Kutta method
  - improve accuracy: evaluate the function f at more points

$$k_1 = f(x(t), t)\Delta t$$

$$k_2 = f\left(x(t) + \frac{k_1}{2}, t + \frac{\Delta t}{2}\right)\Delta t$$

$$k_3 = f\left(x(t) + \frac{k_2}{2}, t + \frac{\Delta t}{2}\right)\Delta t$$

$$k_4 = f\left(x(t) + k_3, t + \Delta t\right)\Delta t$$

Fourth order Runge-Kutta method

$$x(t + \Delta t) = x(t) + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(\Delta t^5)$$

Solving differential equations: General

- Runge-Kutta method
  - suitable for many applications for systems involving few degrees of freedom
  - almost never used for molecular dynamics involving many degrees of freedom
    - fourth order method: four evaluations of the derivative in every step → computational demanding
    - no time-reversal symmetry
    - not symplectic
    - erroneous behavior on longer time scales

Solving differential equations: General

- Gear algorithm: predictor-corrector method
  - Predictor stage: use Taylor series to estimate the changes in positions and their time derivatives (e.g. up to third order) over a small time step Δt

$$\mathbf{r}^p(t+\Delta t) = \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \frac{1}{2}\mathbf{a}(t)(\Delta t)^2 + \frac{1}{6}\mathbf{a}'(t)(\Delta t)^3$$

$$\mathbf{v}^p(t+\Delta t) = \mathbf{v}(t) + \mathbf{a}(t)\Delta t + \frac{1}{2}\mathbf{a}'(t)(\Delta t)^2$$

$$\mathbf{a}^p(t+\Delta t) = \mathbf{a}(t) + \mathbf{a}'(t)\Delta t$$

$$\mathbf{a}'^p(t+\Delta t) = \mathbf{a}'(t)$$

Solving differential equations: General

– Corrector stage: evaluate the accelerations from the forces at the predicted position

$$\mathbf{a}^c(t+\Delta t) = -\frac{1}{m}\frac{\partial V}{\partial \mathbf{r}}\bigg|_{\mathbf{r}=\mathbf{r}^p(t+\Delta t)}$$

and correct the positions and their derivatives on the basis of the deviation between predicted and evaluated accelerations

$$\Delta\mathbf{a}(t+\Delta t) = \mathbf{a}^c(t+\Delta t) - \mathbf{a}^p(t+\Delta t)$$

$$\mathbf{r}^c(t+\Delta t) = \mathbf{r}^p(t+\Delta t) + c_0\Delta\mathbf{a}(t+\Delta t)$$

$$\mathbf{v}^c(t+\Delta t) = \mathbf{v}^p(t+\Delta t) + c_1\Delta\mathbf{a}(t+\Delta t)$$

$$\mathbf{a}^c(t+\Delta t) = \text{computed from the forces (see above)}$$

$$\mathbf{a}'^c(t+\Delta t) = \mathbf{a}'^p(t+\Delta t) + c_2\Delta\mathbf{a}(t+\Delta t)$$

Solving differential equations: General

- Gear algorithm: predictor-corrector method
  - has been used much in MD simulations in the past, but not anymore
  - only one calculation of forces per time step
  - accurate for small time steps, but not very stable for larger time steps
  - when the forces are not very precise it does not help to use higher orders
  - no time-reversible symmetry
  - not symplectic