

CHEM-E4115

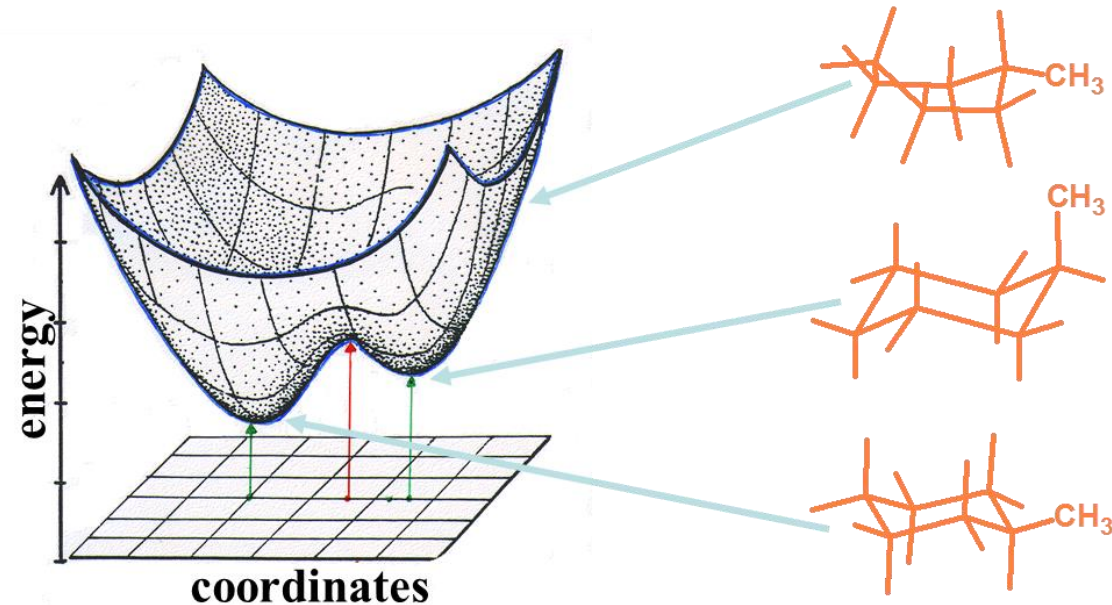
Computational Chemistry I (5op)

2nd part: molecular modelling

Chapters 6.5.-6.9

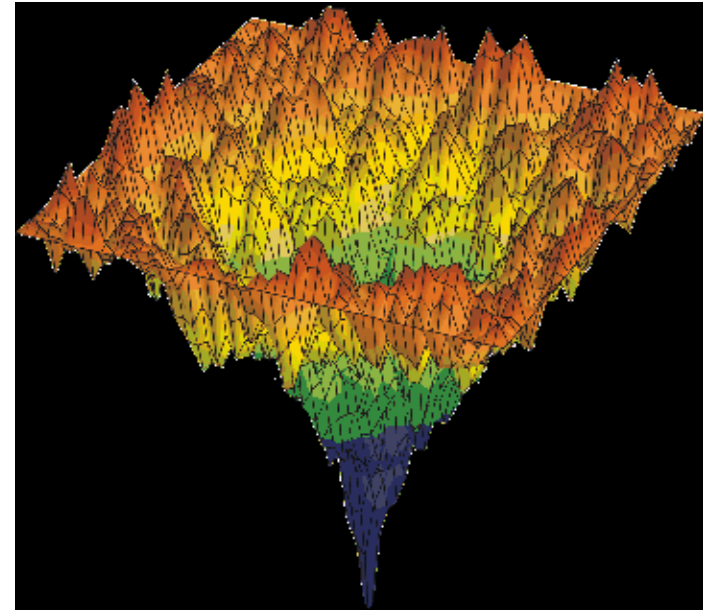
Revision: Potential energy surface

- Defined by force-field for each molecule or molecule system
- Each point represents a molecular conformation



From molecular conformations to measurable averages

- We have: Potential energy surface
- We need: A measurable quantity
- Obtaining the measurable quantity
 - Molecular dynamics: deterministic sampling
 - Monte Carlo: stochastic sampling



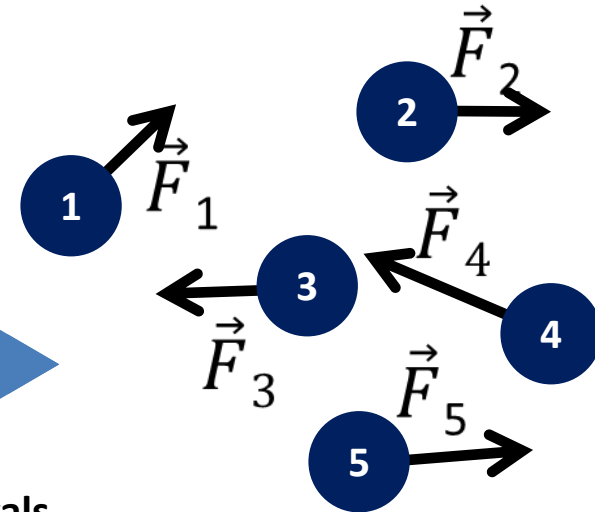
Revision:

Basics of molecular dynamics

- Potential energy functional E (function of nuclei positions) -> Force on each nuclei

$$\vec{F} = -\nabla E$$

$$\vec{F} = m\vec{a} = m\frac{d\vec{v}}{dt} = m\frac{d^2\vec{r}}{dt^2}$$



Force for each particle **calculated** at **discrete time intervals**

Particle **positions updated** assuming particle moves with this force (acceleration) in the direction of force for the entire (short) time interval

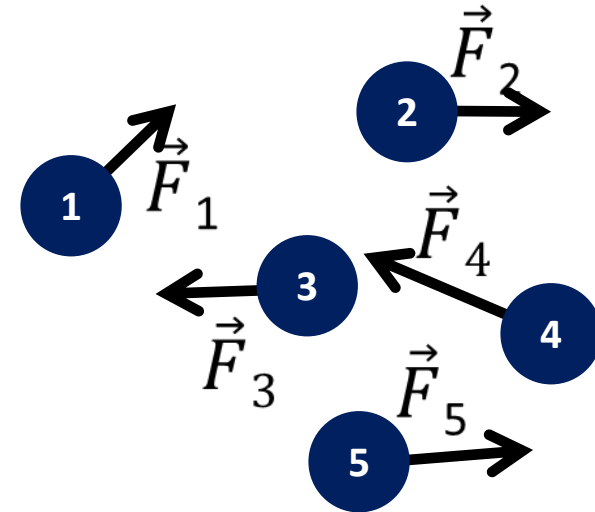
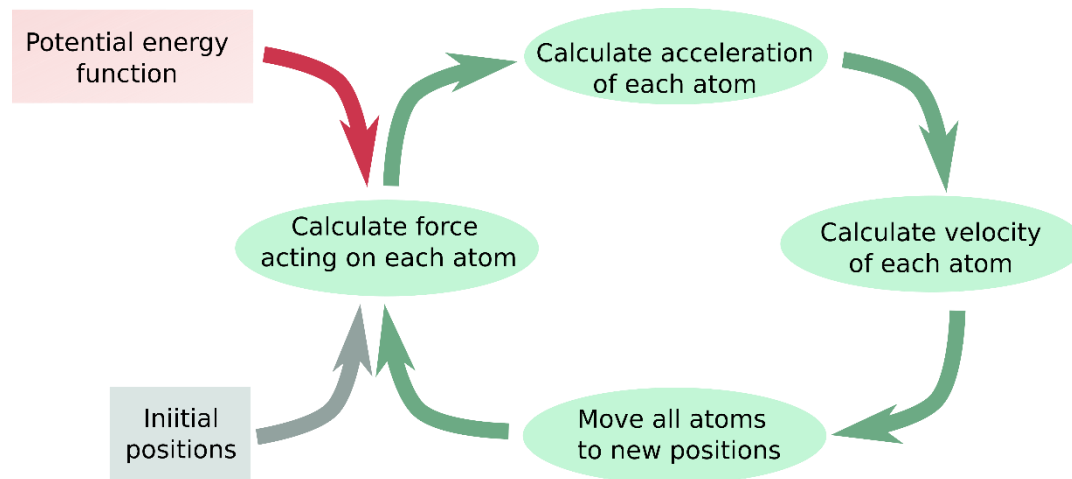
New forces calculated with updated positions

loop-as-long-as-wanted (typically as long as possible)

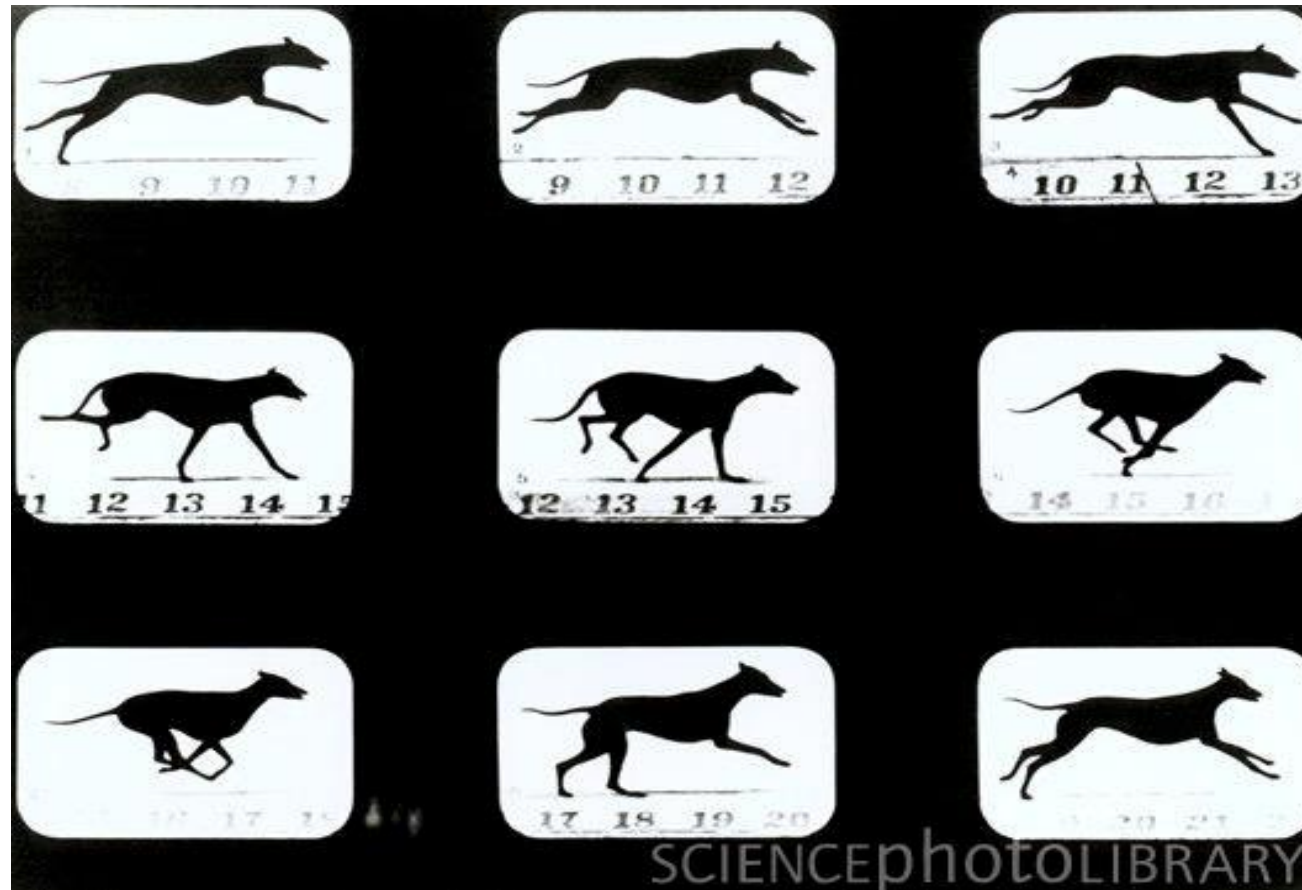
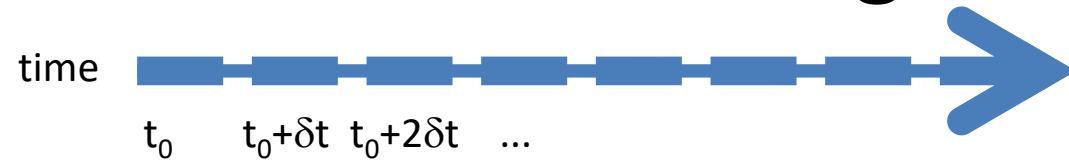
Force-field basics

- Potential energy functional E (function of nuclei positions) -> Force on each nuclei

$$\vec{F} = -\nabla E$$
$$\vec{F} = m\vec{a} = m\frac{d\vec{v}}{dt} = m\frac{d^2\vec{r}}{dt^2}$$



Molecular dynamics in brief: sequence of static images



Molecular dynamics

- Thermodynamic quantities, conformation properties as ensemble average using numerical integration
- M number of time steps

$$A_{ave} = \lim_{\tau \rightarrow \infty} \frac{1}{\tau} \int_{t=0}^{\tau} A(\mathbf{p}^N(t), \mathbf{r}^N(t)) dt$$
$$\langle A \rangle = \frac{1}{M} \sum_{i=1}^M A(\mathbf{r}^N)$$

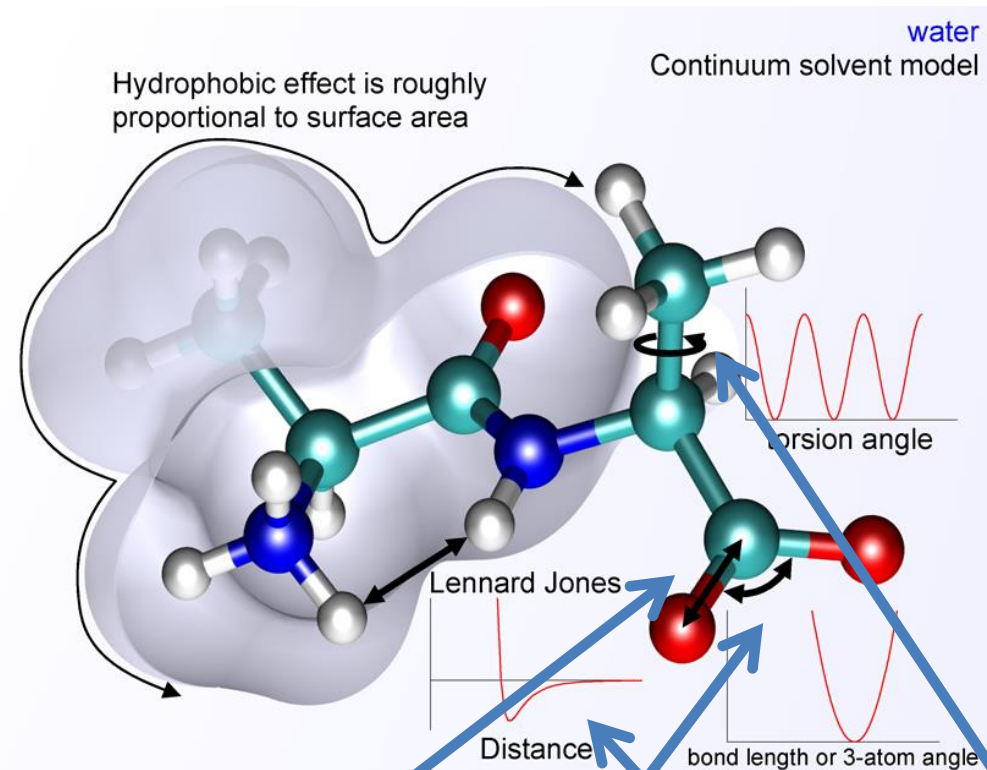
From quantum mechanics to molecular mechanics

- Understanding complex mixtures, assembly structures, macromolecular systems, biological phenomena, ... requires considering systems with a large number of atoms for long time windows.
- Forces acting between atoms and molecules are very complex.
- A very fast method of evaluations molecular interactions is needed to achieve these goals.
- Many molecular systems unfortunately too large to be considered by quantum mechanics
- Force-field methods (molecular mechanics) ignore electronic motion and calculate **the energy of the system as a function of nuclei positions** (molecular subunit positions in coarse-grained force-fields)
 - Enables treating large number of atoms (up to $\sim 10^6$ - 10^7)
 - Loses most electron based characteristics (conductivity, i.e., band-gaps, most often also reaction kinetics*, all chemical reactions* and charge re-distribution*)

* Typically. That is, some specific force-fields are designed to reproduce also reaction barriers and limited reactions (typically bond-order type advanced force fields) and some enable charge re-distribution (polarization) to some extent

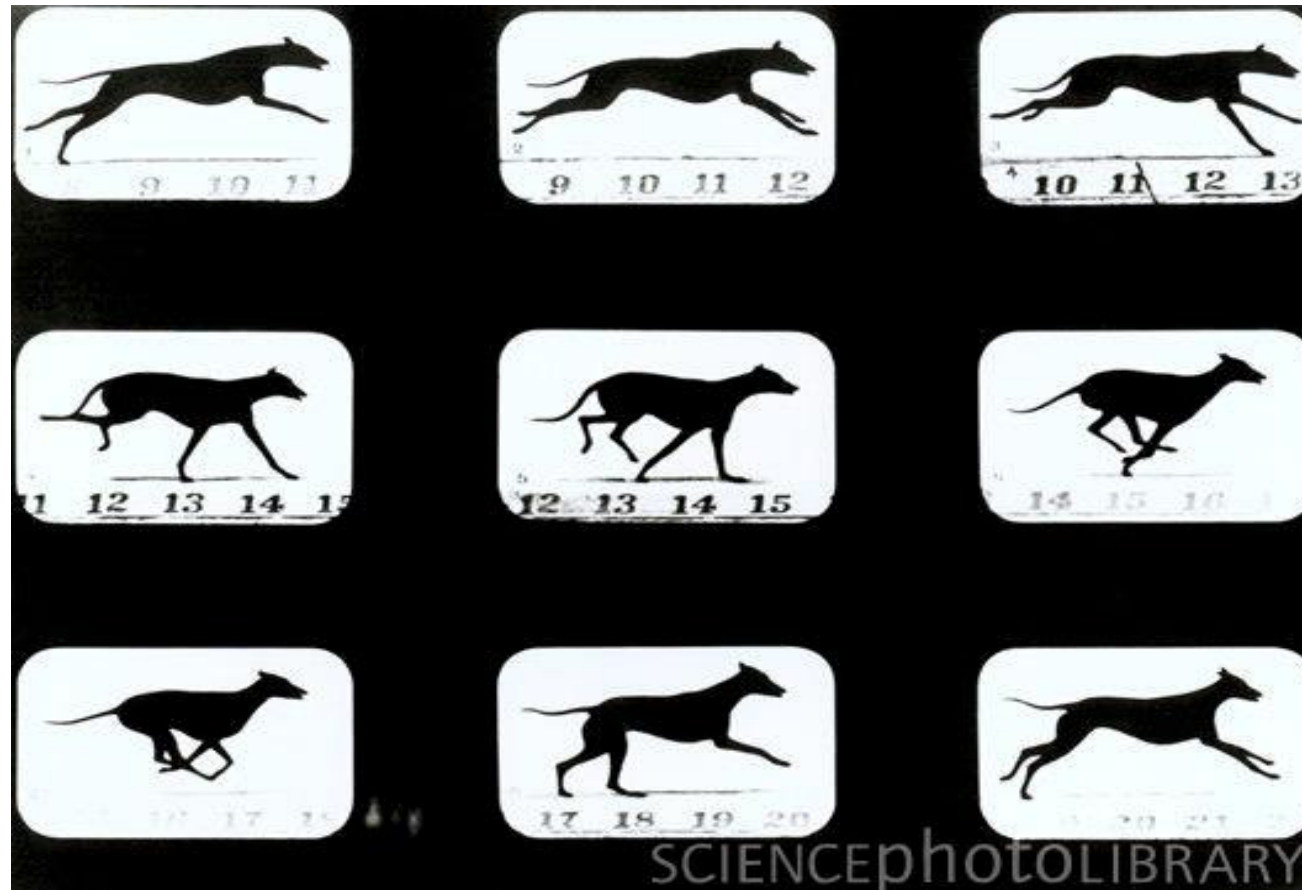
Brief glimpse on where F comes from: Typical representation of a force-field (Potential energy surface)

Dialanine peptide in implicit (continuum) solvent

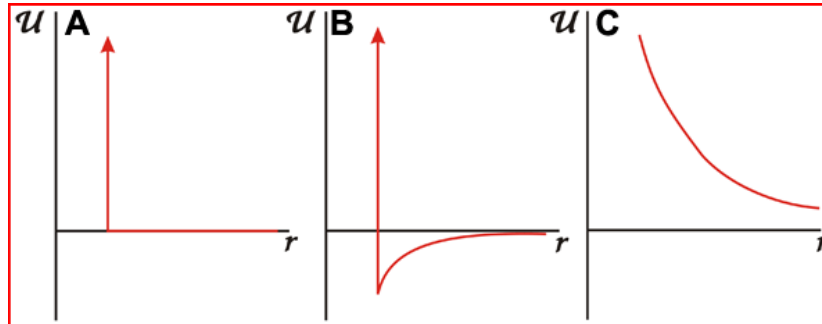


$$E_{\text{bonded}} = E_{\text{bond}} + E_{\text{angle}} + E_{\text{dihedral}}$$
$$E_{\text{nonbonded}} = E_{\text{electrostatic}} + E_{\text{van der Waals}}$$

Next: How to get to the sequence of static images from a potential energy (force-field)



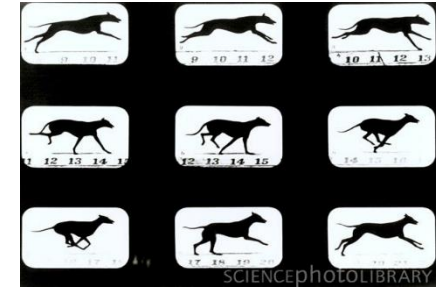
Molecular dynamics



- $\vec{F} = m\vec{a}$
- $\frac{d^2x_i}{dt^2} = \frac{F_{xi}}{m_i}$
- Discrete potentials, constant force models
 - Analytical calculation until next collision, relatively simple
 - Identify next collision
 - Calculate positions at next collision
 - Determine new velocities after collision (conservation of momentum)
 - Loop
- Continuous potentials
 - Discrete stepwise integration (finite difference)
 - **Attention here!**



Finite difference methods for time propagation: Molecular dynamics of continuous potentials



- Basis of all algorithms: Taylor's series

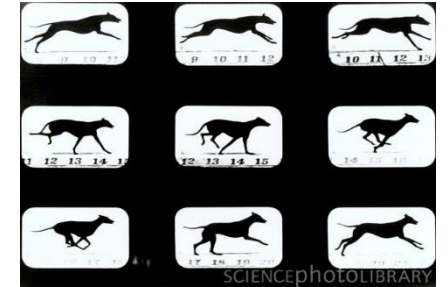
$$r(t + \delta t) = r(t) + v(t)\delta t + \frac{1}{2} a(t)\delta t^2 + \frac{1}{6} b(t)\delta t^3 + \frac{1}{24} c(t)\delta t^4 + \dots$$

$$v(t + \delta t) = v(t) + a(t)\delta t + \frac{1}{2} b(t)\delta t^2 + \frac{1}{6} c(t)\delta t^3 + \dots$$

$$a(t + \delta t) = a(t) + b(t)\delta t + \frac{1}{2} c(t)\delta t^2 + \dots$$

$$b(t + \delta t) = b(t) + c(t)\delta t + \dots$$

Finite difference methods for time propagation: Molecular dynamics of continuous potentials

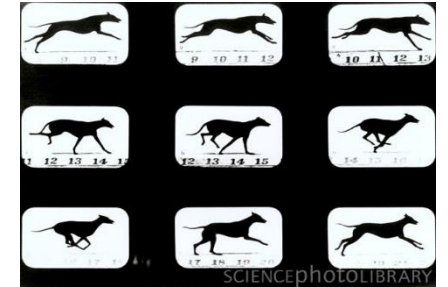
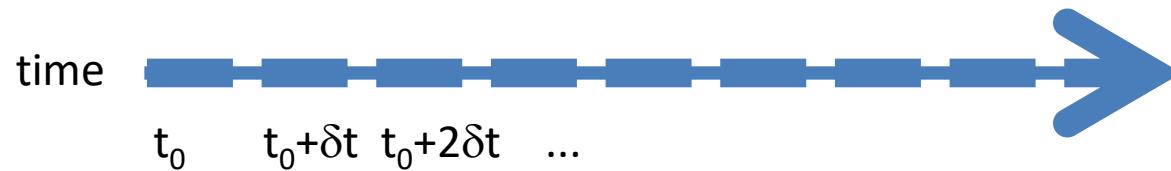


- Basis of all algorithms: Taylor's series

Most simple:
Euler algorithm

$$r(t + \delta t) = r(t) + v(t)\delta t + \frac{1}{2} a(t)\delta t^2 + \frac{1}{6} b(t)\delta t^3 + \frac{1}{24} c(t)\delta t^4 + \dots$$
$$v(t + \delta t) = v(t) + a(t)\delta t + \frac{1}{2} b(t)\delta t^2 + \frac{1}{6} c(t)\delta t^3 + \dots$$
$$a(t + \delta t) = a(t) + b(t)\delta t + \frac{1}{2} c(t)\delta t^2 + \dots$$
$$b(t + \delta t) = b(t) + c(t)\delta t + \dots$$

Finite difference methods for time propagation: Molecular dynamics of continuous potentials



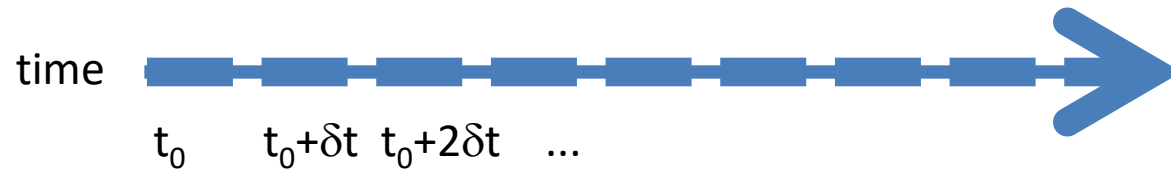
- Basis of all algorithms: Taylor's series

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

$$v(t + \delta t) = v(t) + a(t)\delta t$$

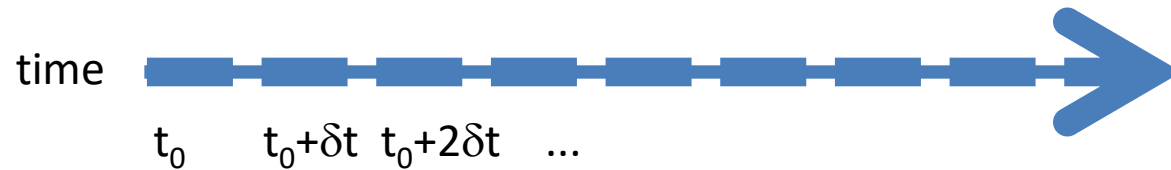
Most simple:
Euler algorithm

Molecular dynamics integration algorithm should be



- Fast
- Use little memory
- **Allow a long time step δt**
- Reproduce the correct path (note: never possible)
- **Conserve energy (&reversible in time)**
- Be easy to implement
- Contain only one force evaluation/time step

Molecular dynamics integration algorithm should be



- Fast
- Use little memory
- **Allow a long time step δt**
- Reproduce the correct path (note: never possible)
- **Conserve energy (&reversible in time)**
- Be easy to implement
- Contain only one force evaluation/time step

Euler algorithm:

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

$$v(t + \delta t) = v(t) + a(t)\delta t$$

Note that Euler is

- 1) Not energy conserving
- 2) Not reversible in time

-> Not recommended for molecular dynamics simulations, other simulations can employ (for example, common choice in Brownian dynamics / Langevin dynamics simulations)

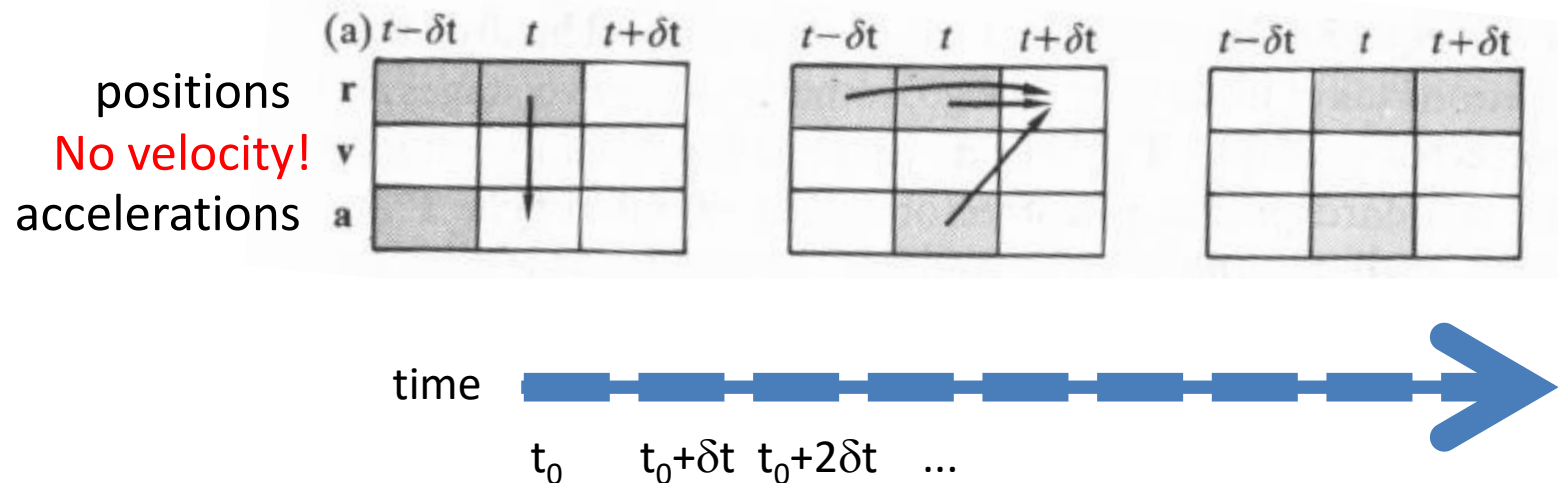
Actual options in molecular dynamics simulations:

1. Verlet algorithm: Taylor series developed at two different times

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

$$r(t - \delta t) = r(t) - v(t)\delta t + \frac{1}{2}a(t)\delta t^2$$

SUM $r(t + \delta t) = 2r(t) - r(t - \delta t) + a(t)\delta t^2$



Verlet L. Computer "Experiments" on Classical Fluids. I. *Thermodynamical Properties of Lennard-Jones Molecules*. Phys Rev. 1967;159: 98–103. [doi:10.1103/PhysRev.159.98](https://doi.org/10.1103/PhysRev.159.98)

Actual options in molecular dynamics simulations:

1. Verlet algorithm: Taylor series developed at two different times

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

$$r(t - \delta t) = r(t) - v(t)\delta t + \frac{1}{2}a(t)\delta t^2$$

SUM $r(t + \delta t) = 2r(t) - r(t - \delta t) + a(t)\delta t^2$

Disadvantages:

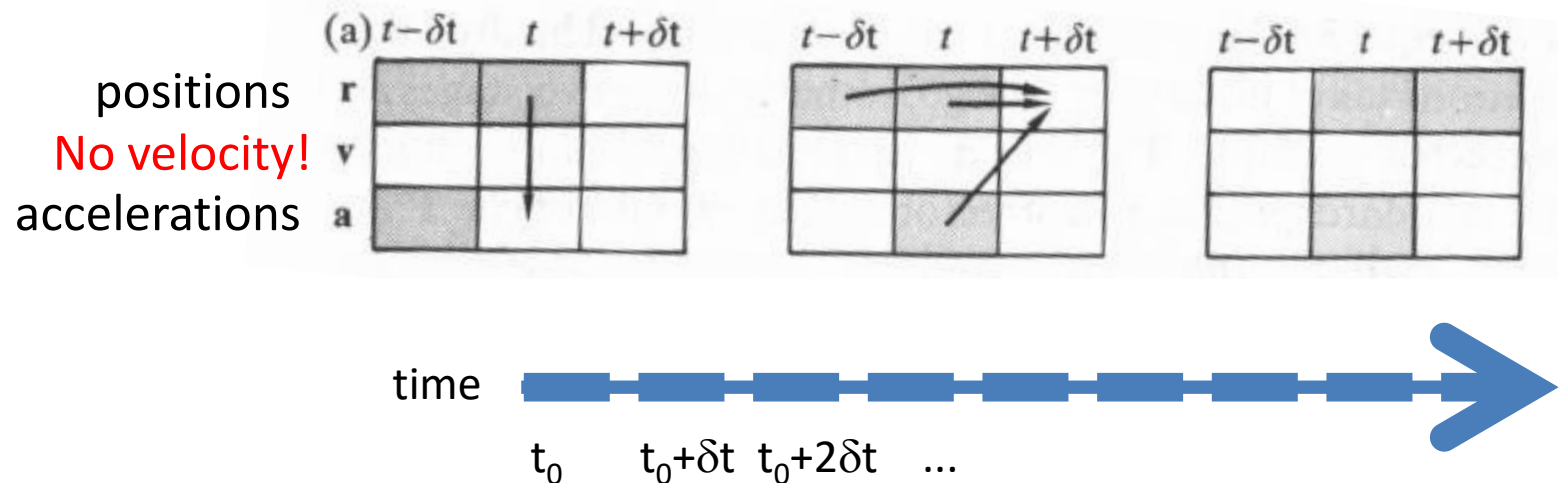
needs 2 sets of positions (also at t=0),
acceleration term much smaller than
position terms (loss of precision)

Velocities

$$v(t) = [r(t + \delta t) - r(t - \delta t)]/2\delta t$$

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

disadvantage: velocities ½ step off from
positions



Verlet L. Computer "Experiments" on
Classical Fluids. I. *Thermodynamical
Properties of Lennard-Jones Molecules.*
Phys Rev. 1967;159: 98–103.
[doi:10.1103/PhysRev.159.98](https://doi.org/10.1103/PhysRev.159.98)

Actual options in molecular dynamics simulations:

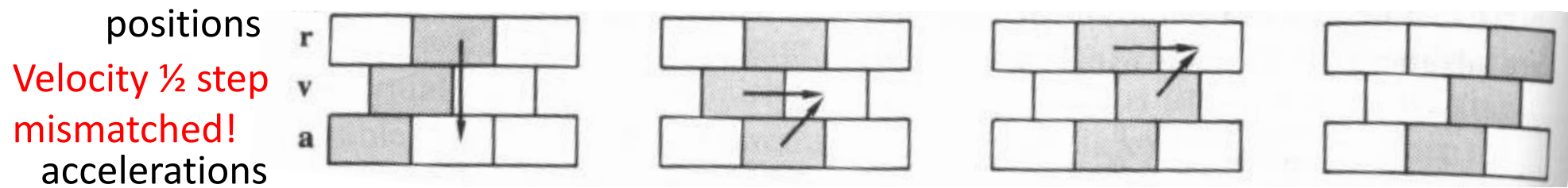
2. Verlet leap-frog algorithm: Positions and velocities leap by $\frac{1}{2}$ step over each other (=a better choice)

$$r(t + \delta t) = r(t) + v\left(t + \frac{1}{2}\delta t\right)\delta t$$

$$v\left(t + \frac{1}{2}\delta t\right) = v\left(t - \frac{1}{2}\delta t\right) + a(t)\delta t$$

Now, velocities explicitly present but $\frac{1}{2}$ time step off from positions! Kinetic energy / temperature off by $\frac{1}{2}$ time step.

No summation of small δt^2 terms ☺



Actual options in molecular dynamics simulations:

3. Velocity Verlet: positions, velocities and accelerations at the same moment (=the most used choice)

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

Calculate $a(t + \delta t)$ using new positions $r(t + \delta t)$

Use both $a(t)$ and $a(t + \delta t)$ to calculate

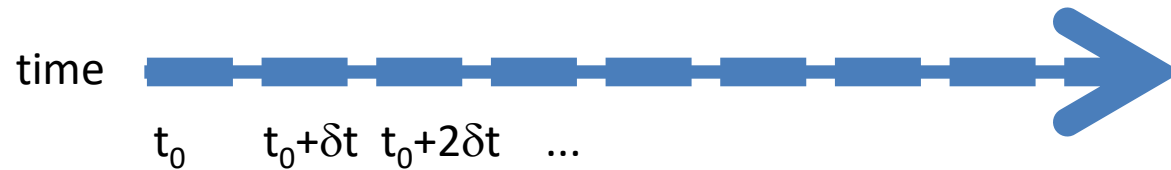
$$v(t + \delta t) = v(t) + \frac{1}{2}(a(t) + a(t + \delta t))\delta t$$

time-reversible and energy conserving

Mathematically equivalent to original Verlet algorithm but much more stable (longer time step ok)

Due to its simplicity and stability the Velocity Verlet is the most widely used algorithm in molecular dynamics simulations.

Revision: Molecular dynamics integration algorithm should be



- Fast
- Use little memory
- **Allow a long time step δt**
- Reproduce the correct path (note: never possible)
- **Conserve energy (&reversible in time)**
- Be easy to implement
- Contain only one force evaluation/time step

In Gromacs simulations software (=the exercises), you choose integration algorithm in the **mdp file**:

integrator = md

; A leap frog algorithm

integrator = md-vv

; A velocity Verlet algorithm

integrator = md-vv-avek

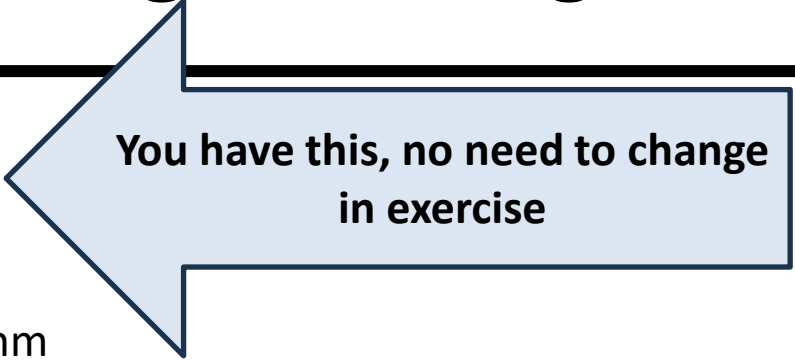
; A velocity Verlet algorithm same as md-vv except the kinetic energy is calculated as the average of the two half step kinetic energies. More accurate than the md-vv.

integrator = sd

; An accurate leap frog stochastic dynamics integrator.

integrator = bd

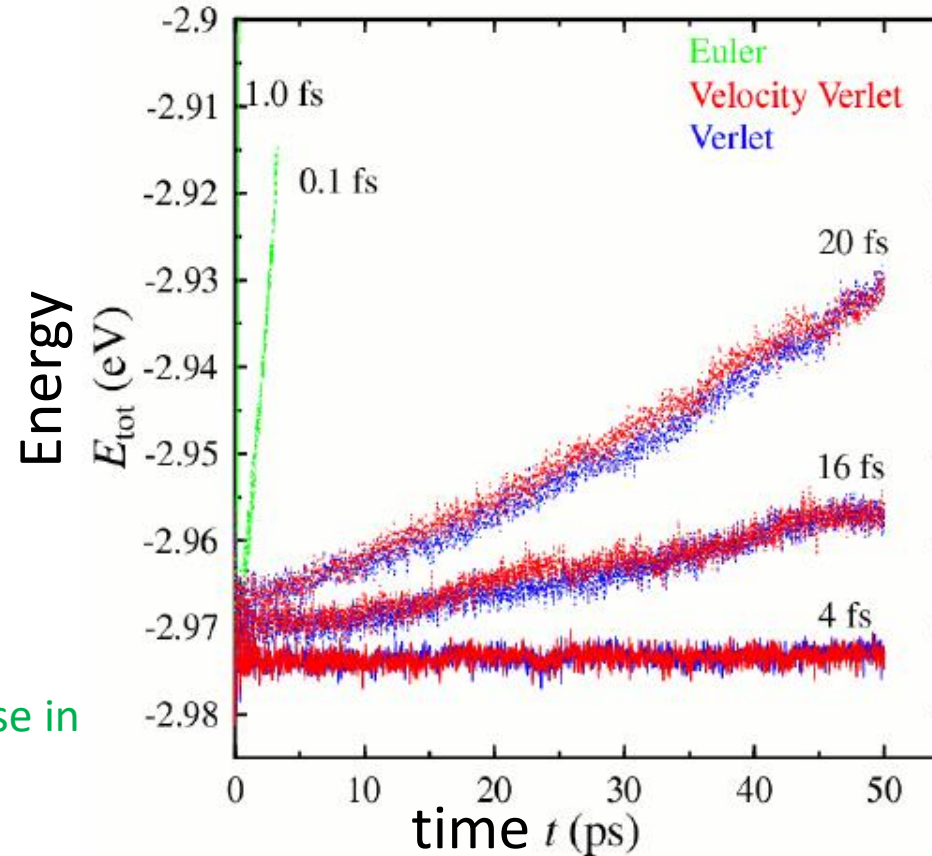
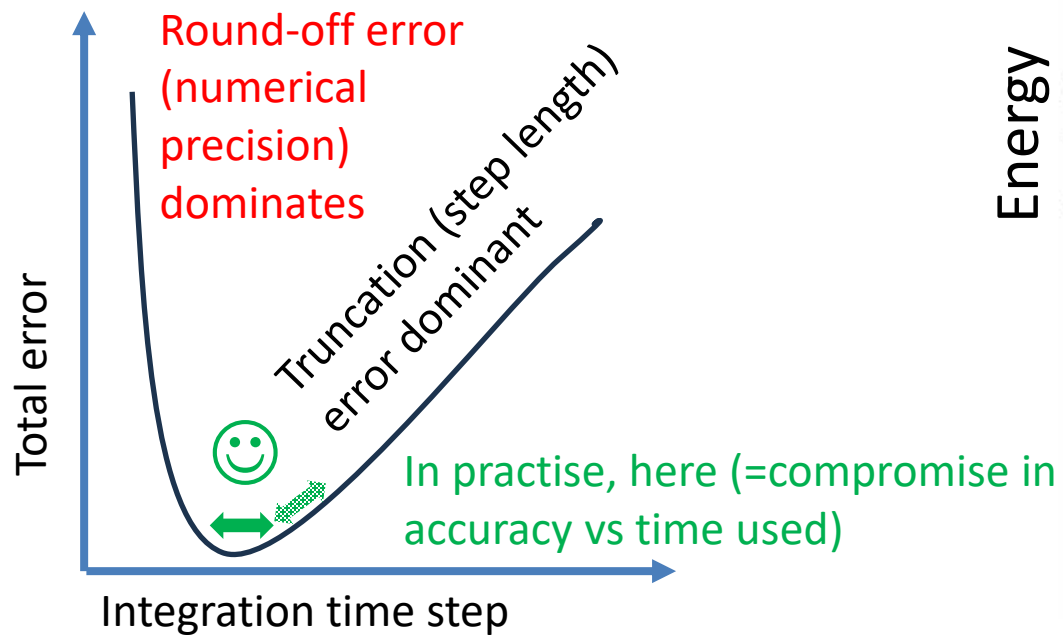
; A Euler integrator for Brownian or position Langevin dynamics.



You have this, no need to change
in exercise

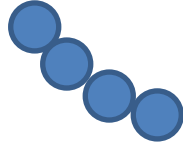
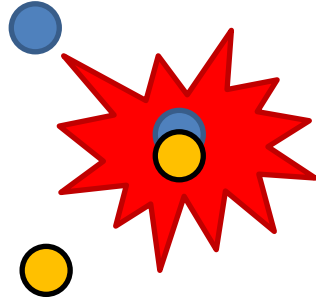
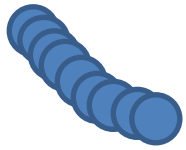
Error in molecular dynamics simulation time propagation: energy conservation and time steps

Origins of finite time integration error



300 Cu atoms at 300K

Choosing a time step



Too short!

Too long!

Outcome: Typical time steps in atomistic simulations

Atoms	10fs
Rigid molecules	5fs
Flexible molecules (bonds with H const.)	2fs
Flexible molecules, flexible bonds	0.5-1fs

Larger time step allows to run simulation faster, but accuracy decreases.

- Verlet family integrators are stable for time steps $\delta t \leq \frac{1}{\pi f}$ where f is oscillation frequency
- Vibrations hydrogen bonds have period of 10 fs
- Bond vibrations involving heavy atoms and angles involving hydrogen atoms have period of 20 fs
- Stretching of bonds with the lightest atom H is the fastest motion.
- As period of oscillation of a C-H bond is about 10 fs, Verlet integration is stable for time steps < 3.2 fs (In practice, 1 fs time step is recommended for this).
- Time step can be doubled by constraining bonds with hydrogens. **Your exercises do this!**
- Further increase of time step requires constraining bonds between all atoms and angles involving hydrogen atoms

In Gromacs simulations software (=the exercises),
you choose simulation time step in the **mdp file**:

dt = 0.002

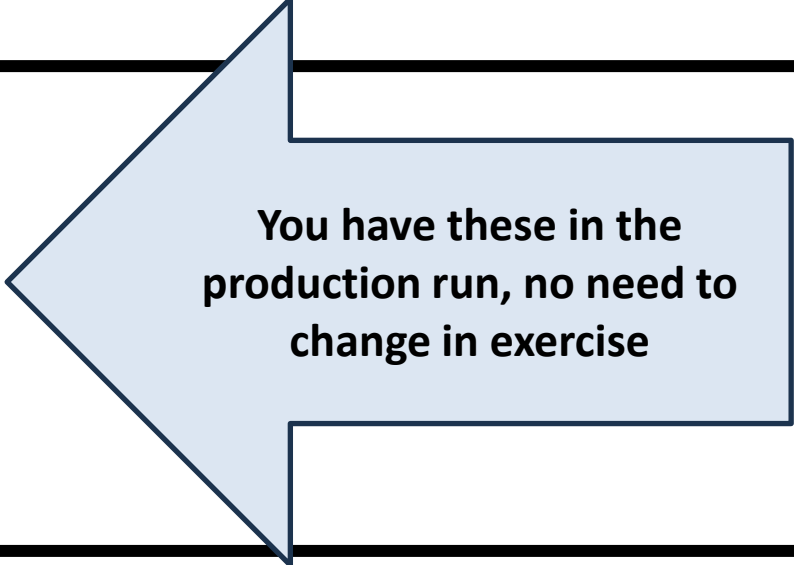
; Time step, ps, (2 ns)

nsteps = 500000

; Number of steps to simulate, this is now 1 ns

tinit = 0

; Time of the first step



You have these in the
production run, no need to
change in exercise

Constraint algorithms (to extend time step)

- To constrain bond length in a simulation the equations of motion must be modified.
- The goal is to constrain some bonds without affecting dynamics and energetics of a system.
- One way to constrain bonds is to apply constraint force acting along a bond in opposite direction.
- In constrained simulation first the unconstrained step is done, then corrections are applied to satisfy constraints.
 - As bonds in molecules are coupled satisfying all constraints in a molecule becomes increasingly complex for larger molecules.
 - Several algorithms have been developed for use specifically with small or large molecules.
- SETTLE
 - Very fast analytical solution for small molecules.
 - Widely used to constrain bonds in water molecules.
- SHAKE
 - Iterative algorithm that resets all bonds to the constrained values sequentially until the desired tolerance is achieved.
 - Simple and stable, it can be applied for large molecules.
 - Works with both bond and angle constraints.
 - Slower than SETTLE and hard to parallelize.
 - SHAKE may fail to find the constrained positions when displacements are large.
 - Extensions of the original SHAKE algorithm: RATTLE, QSHAKE, WIGGLE, MSHAKE, P-SHAKE.
- LINCS
 - Linear constraint solver
 - 3-4 times faster than SHAKE and easy to parallelize.
 - The parallel LINCS (P-LINCS) allows to constrain all bonds in large molecules.
 - Not suitable for constraining both bonds and angles.

Source: <https://computecanada.github.io/molmodsim-md-theory-lesson-novice/aio/index.html>

In Gromacs simulations software (=the exercises), you choose constraint algorithm in the **mdp file** (also in topology file for some constraints, not shown here):

constraints = h-bonds

; Constrain bonds with hydrogen atoms

constraints = all-bonds

; Constrain all bonds

constraints = h-angles

; Constrain all bonds and additionally the angles that involve hydrogen atoms

constraints = all-angles

; Constrain all bonds and angles

constraint-algorithm = LINCS

; Use LINCS

constraint-algorithm = SHAKE

; Use SHAKE

shake-tol = 0.0001

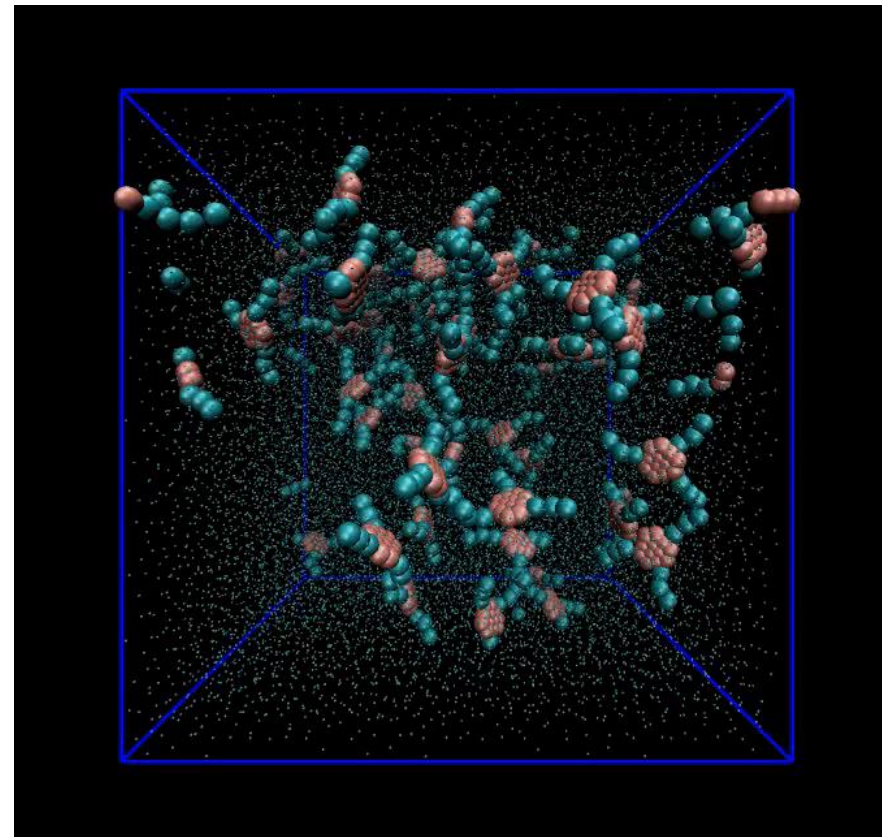
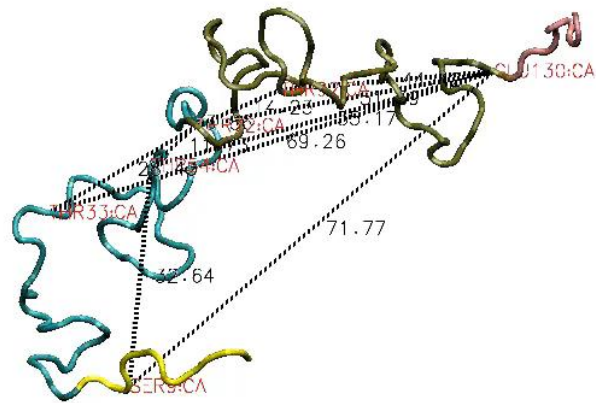
; Relative tolerance for SHAKE, default value is 0.0001.

You have this, no need to change in exercise

You have this, no need to change in exercise

Molecular dynamics in practise

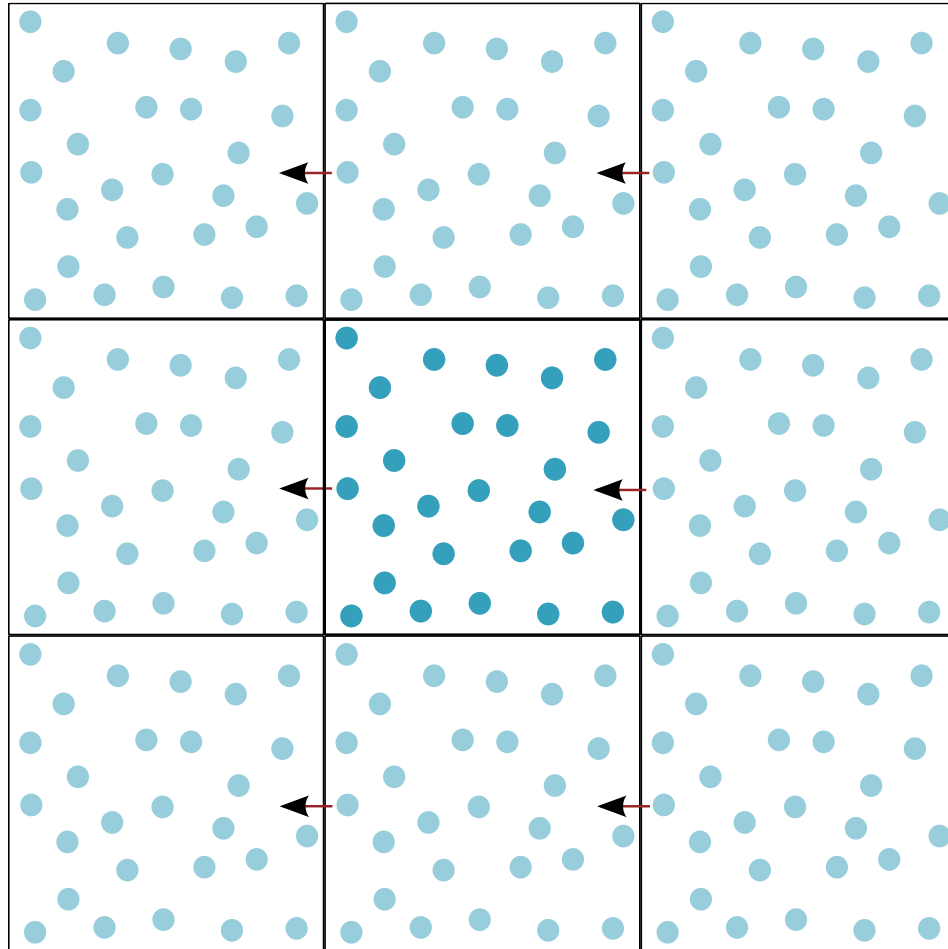
- Next: things to consider



Molecular dynamics in practice: Computational efficiency

- System size: How large?
- Simulation box
 - Boundary conditions, simulation box shape
- Cut-off schemes
 - Do we need to calculate every single particle interaction with all the other particles?
 - If not, how to define which?
 - Cut-off errors

Why periodic boundary conditions?

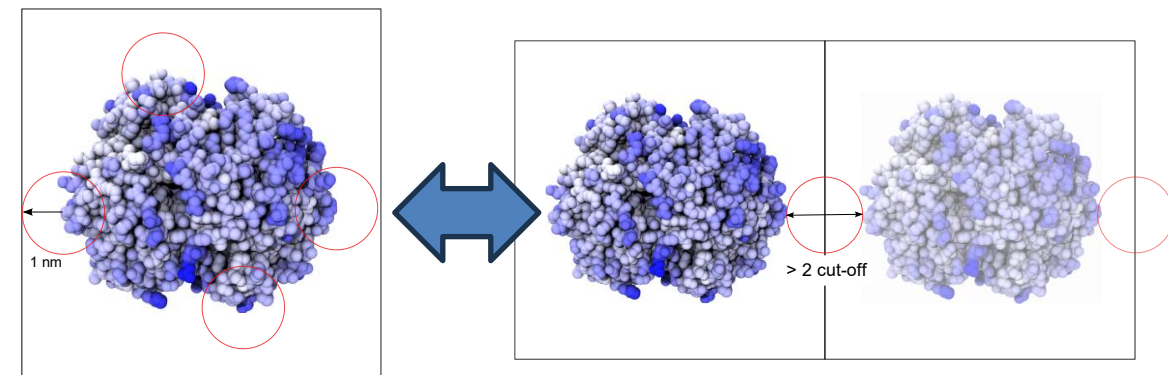
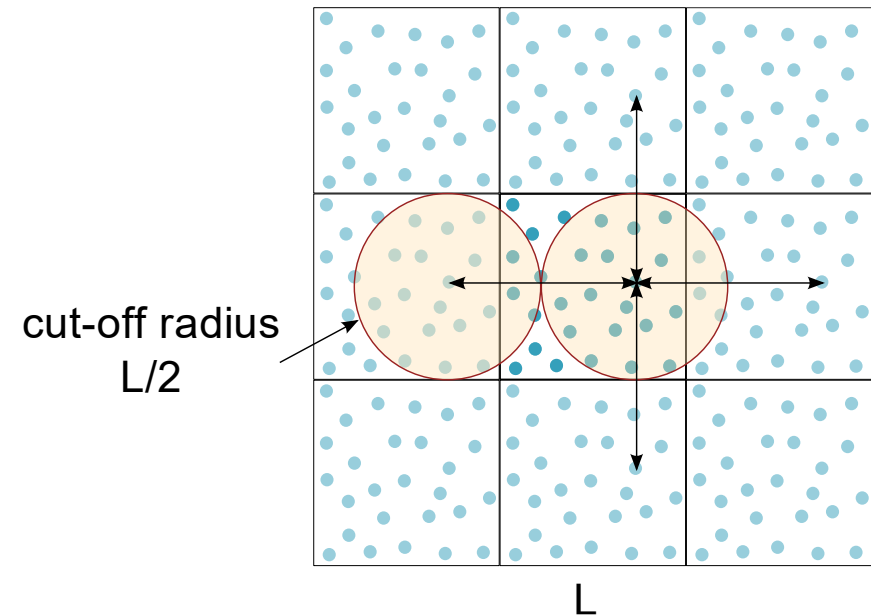


- Simulating a system in realistic environment, such as solution.
- How would you simulate a droplet of water? A boundary to contain water, control temperature, pressure, and density needed.
- Periodic boundary conditions allow approximating an infinite system by using a small part (unit cell).

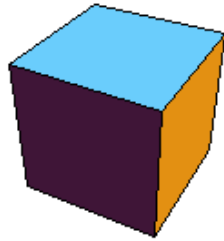
- Unit cell is surrounded by an infinite number of translated copies in all directions (images).
- A particle moving across the boundary reappears on the opposite side.
- Each molecule always interacts with its neighbors even though they may be on opposite sides of the simulation box.
- Artifacts caused by a vacuum are replaced with the PBC artifacts which are in general much less severe.

Computational efficiency: simulation box and simulation box size

- Typically: periodic boundary conditions
- Small is good for computational efficiency
- BUT: Box must be large enough that the system properties are not affected by size
 - Finite size effects
 - Not always achievable!!!
- Rule of thumb: Molecule cannot see its own influence as image over periodic boundary, box must be over $L/2$ in shortest dimension
- For solutes in solvent (water) the minimum box size in atomistic detail modelling should be 1 nm from the solute
- Distance comes from force-field cut-offs (practical choice, not absolute rule)
- Distance different in oil solvents (charge screening difference)

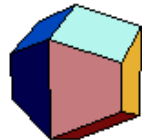


Computational efficiency: simulation box size



Cube vs.

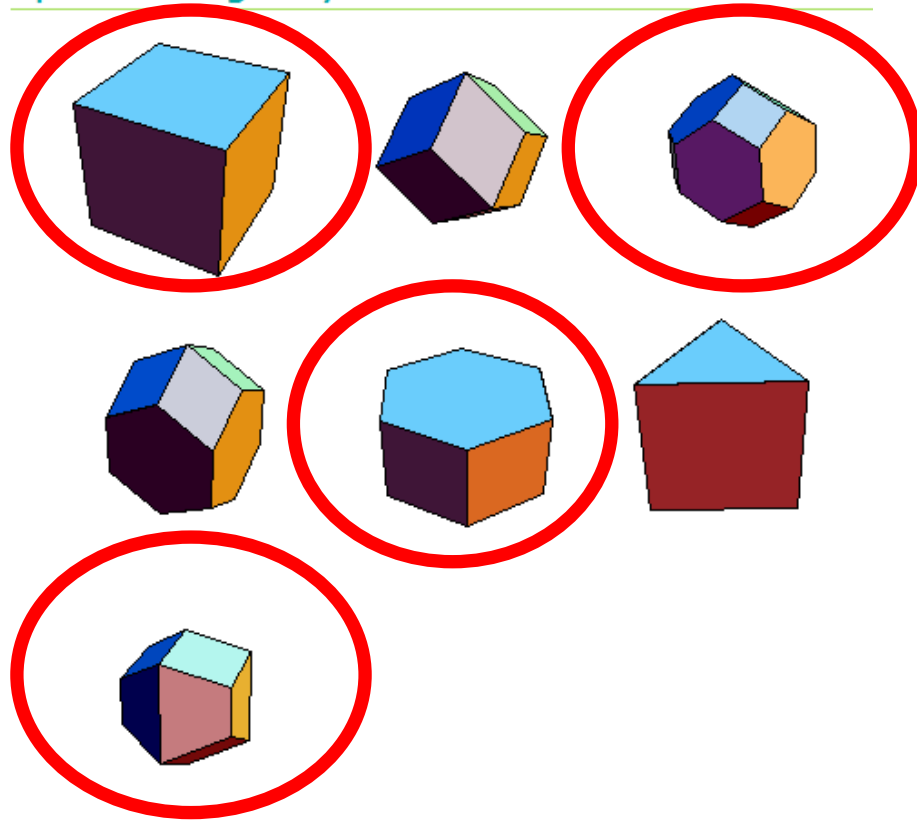
Rhombic dodecahedron:



Consider a solvation simulation of a spherical molecule which cannot see (effectively) its own image through the periodic box images: Cube contains approx 30% more water than rhombic dodecahedron for same solute molecule minimum image distance!

Boundary conditions in simulations (simulation box size)

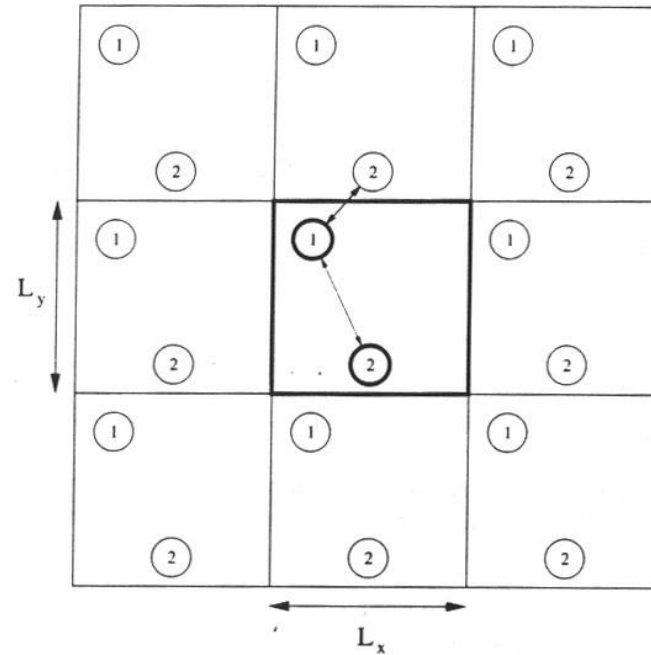
Space-Filling Polyhedron



Commonly used cells:
Cube
Truncated octahedron
Hexagonal prism
Rhombic dodecahedron

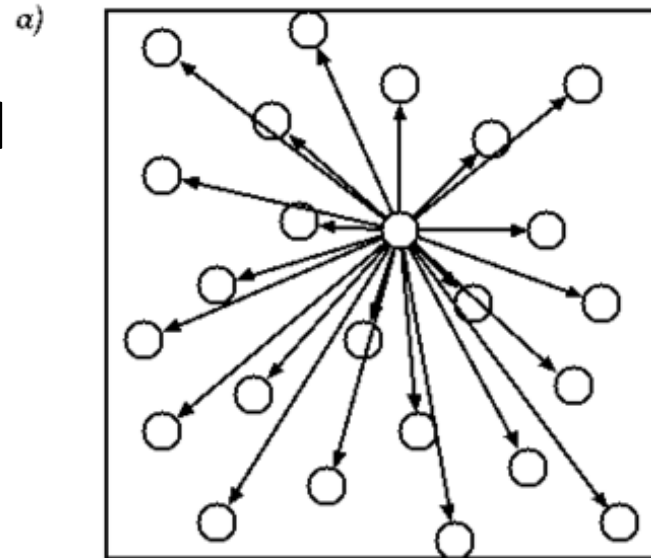
Minimum image convention of periodic boundary conditions

- Particle sees at most just one image every atom in the system (does not see itself)
- Energy or force calculated to closest image
- Typically interaction cut-off radius involved

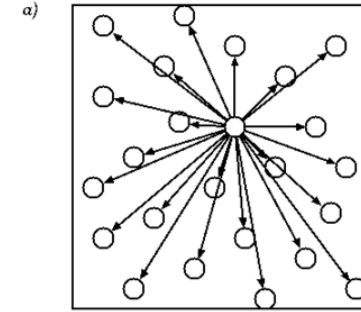


Cut-off schemes in calculating interactions

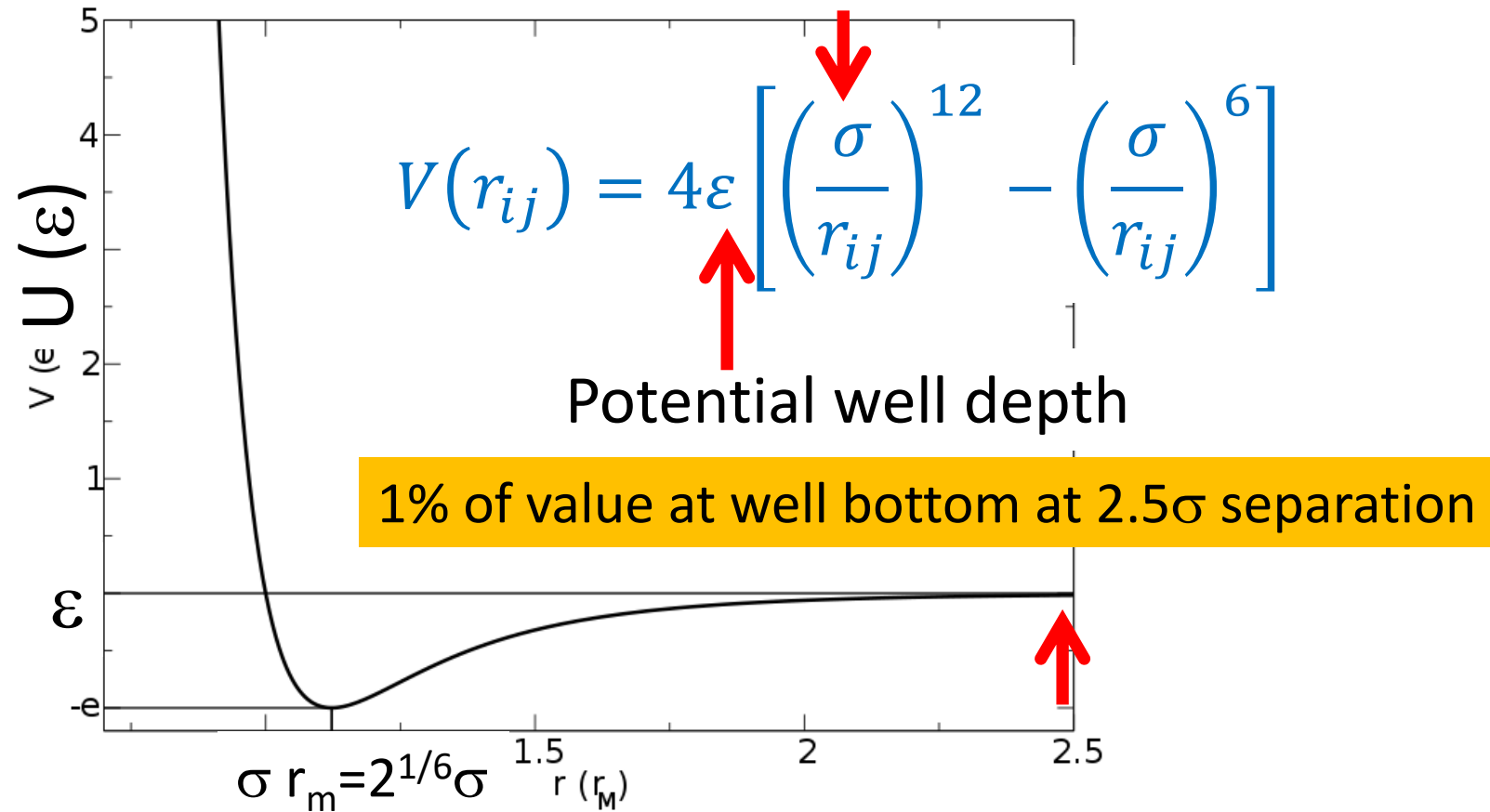
- Truncating the potential and neighbor lists
 - Bonded interactions have limited number of particles involved and scale as $O(N)$ (N number of particles)
 - Non-bonded (in principle) interactions involve all combinations of N particles in an N particle system. Scales as $O(N^2)$. PROBLEM!



Let's take a look at Lennard-Jones potential: Decays as r^{-6}

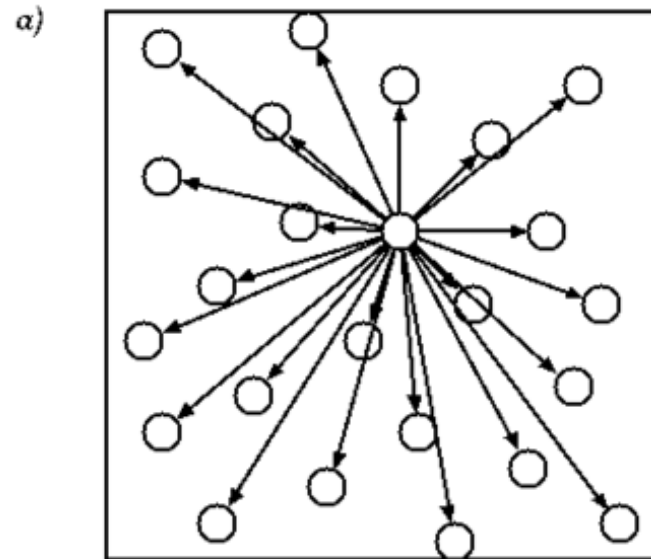


zero-separation



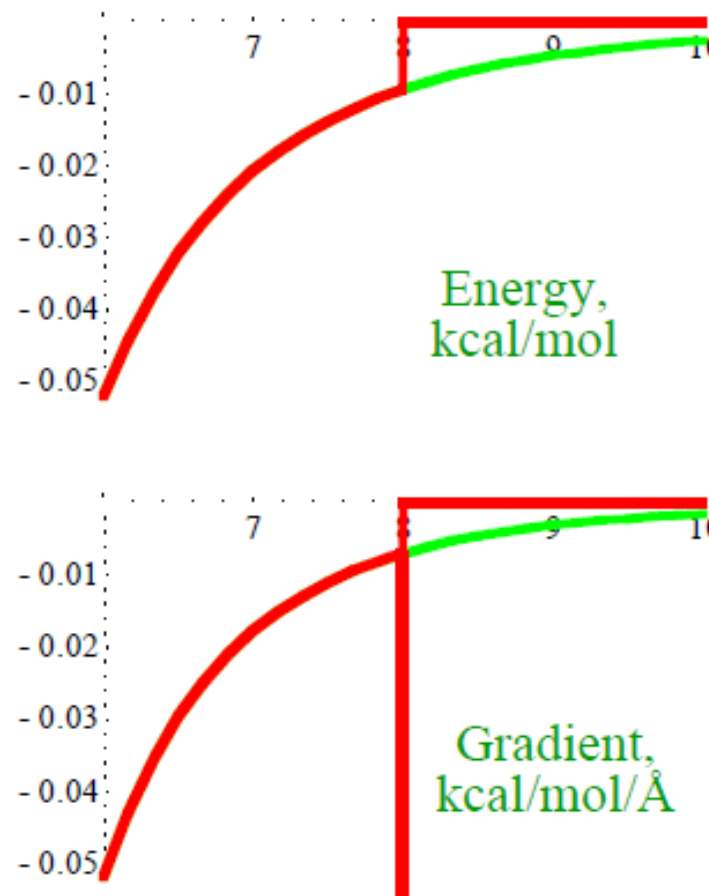
Cut-off schemes in calculating interactions

- Minimum image convention: at max $\frac{1}{2}$ of smallest box side
- Lennard-Jones: 2.5σ corresponds to 1% error
- Coulombic interactions: any kind of cut-off has been shown to cause artifacts: Long range electrostatics such as PME or multipole expansions preferred, reaction field type methods use $\sim 1\text{nm}$ switch cut-off
- More about long-range electrostatics later



Fine-tuning the truncation scheme

- Typically just cut-off, but discontinuity in energy / force may be problem
- To remove discontinuity energy function may be
 - shifted to zero at cut-off
 - switched to zero at cut-off (switching function)
- To remove force discontinuity, the derivative values may be modified at cut-off region

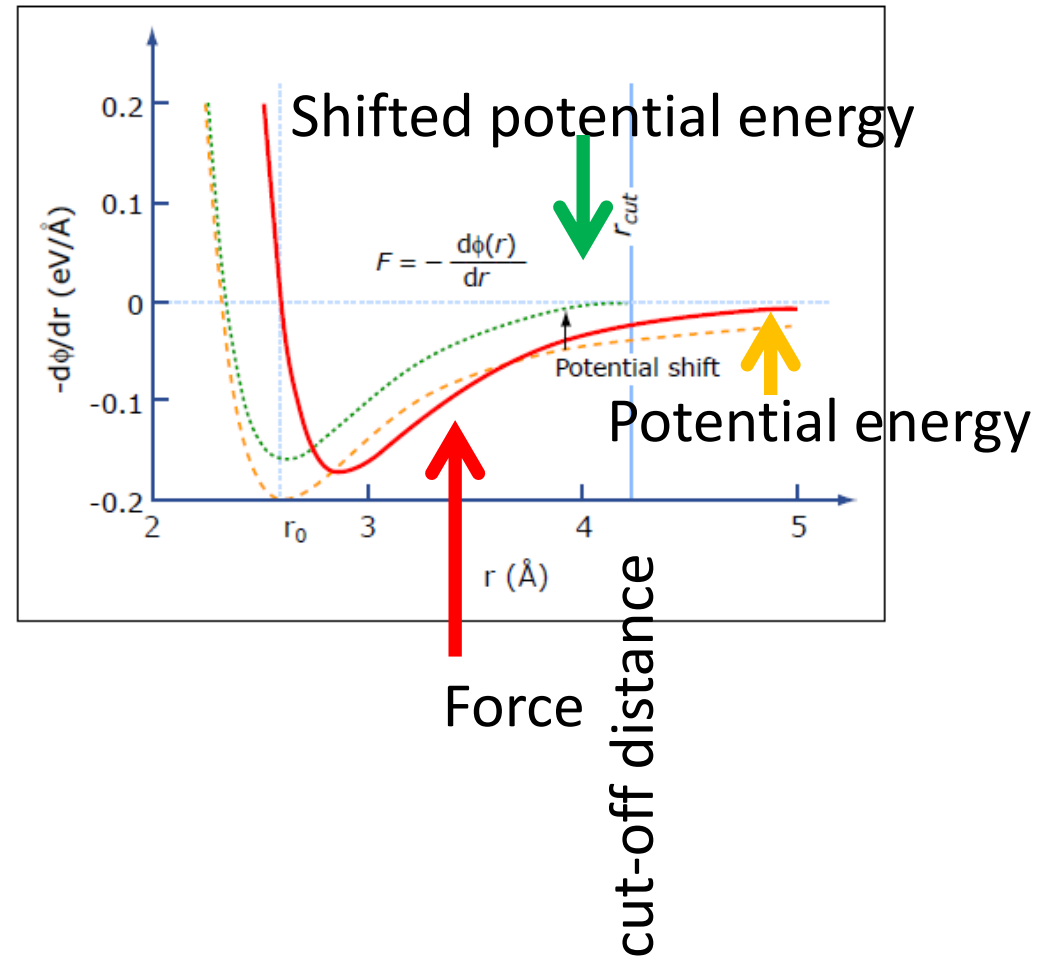


Fine-tuning the truncation scheme

- Potential shifted to zero at cut-off
 - $v'(r)=v(r)-v(r_{cut}), r < r_{cut}$
 - $v'(r)=0, r > r_{cut}$
 - Does not affect force
- Force has discontinuity at cut-off: drop from finite value to zero
 - Force discontinuity can be avoided by setting derivative zero at cut-off
 - $v'(r)=v(r)-v(r_{cut})-\left(\frac{dv(r)}{dr}\right)_{r=r_{cut}}(r - r_{cut}), r < r_{cut}$
 - $v'(r)=0, r > r_{cut}$
 - May be complicated to implement in many body potentials

Fine-tuning the truncation scheme

Lennard-Jones as example of shifted potential

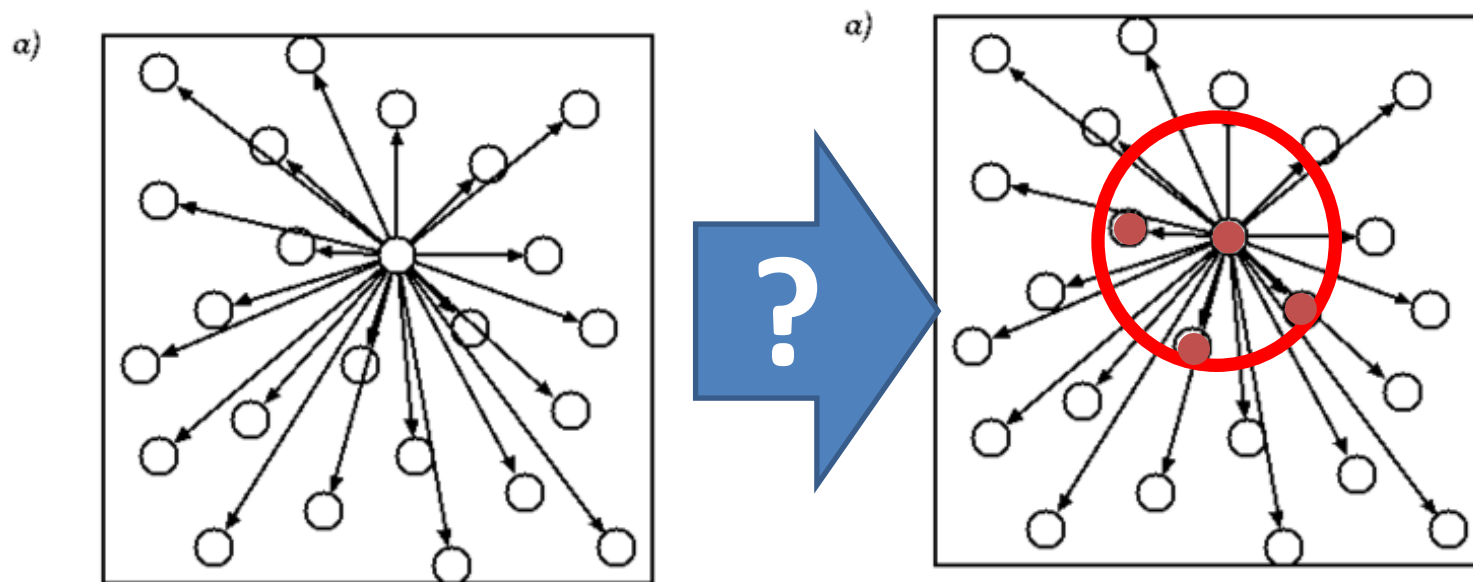


Fine-tuning the truncation scheme

- Potential switched to zero either over $r < r_{cut}$ or over a short region before r_{cut} (switching function).
 - Switching function $v'(r) = v(r)S(r)$, $r < r_{cut}$
 - $v'(r) = 0$, $r > r_{cut}$
 - $S(r=0) = 1$ $S(r_{cut}) = 0$
 - Affects force
- Preferentially 1st and 2nd derivative values at onset of switching and at r_{cut} zero!! (No “jumps” in force)
- Correcting for switching function “jumps” critical in reactive force-fields

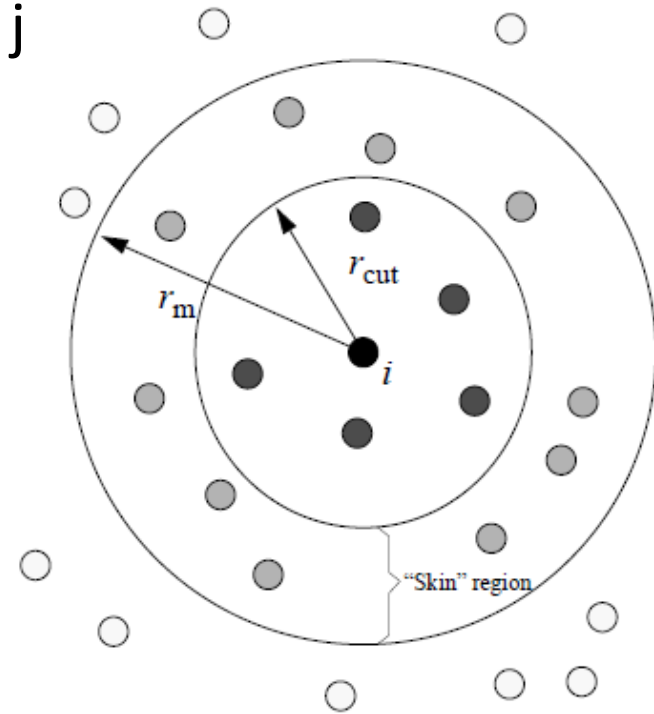
Computational efficiency: How to define which particles are interacting if there is a cut-off?

- If we need to calculate distances to all the particles (minimum image convention), the computational effort is almost as large as calculating all the energies without cut-off
- Most neighbors stay same on consequent steps
- How does one define, which particles are within cut-off distance of each particle?



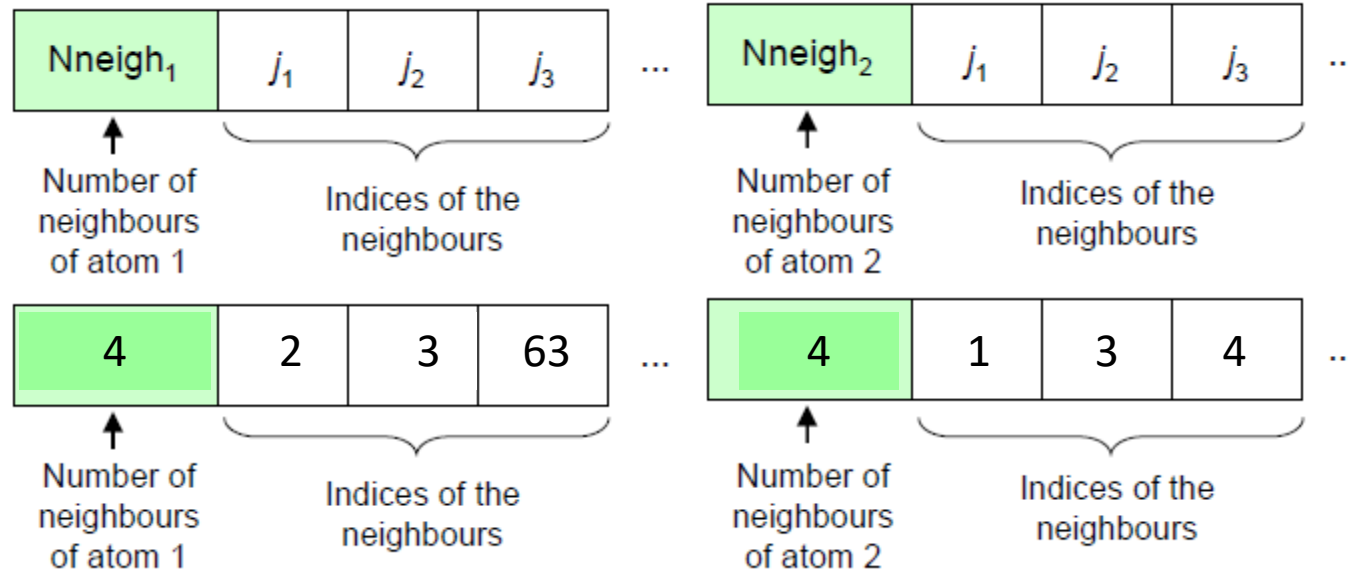
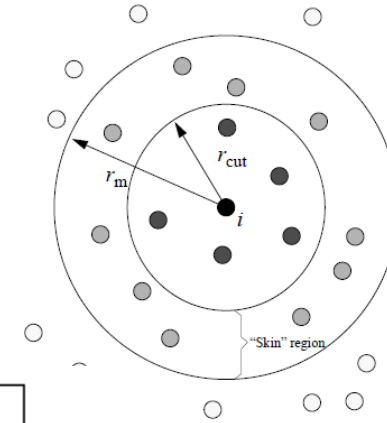
Common solution: Verlet neighbor list

- For each particle i , a list of all particles j within cut-off distance r_{cut} + neighbor list skin thickness distance r_m
 - The list is updated only every M time steps
 - M and $r_m - r_{\text{cut}}$ are chosen such that
- $r_m - r_{\text{cut}} > Mv\delta t$, where v is a typical atom velocity and δt the time step
- Update interval M can be 1) constant interval (simplest), 2) coupled to average v (better) or 3) coupled to maximum displacement of particles kept track with (best)

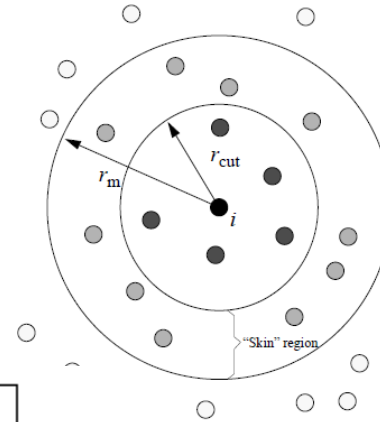


Verlet neighbor list: version 1

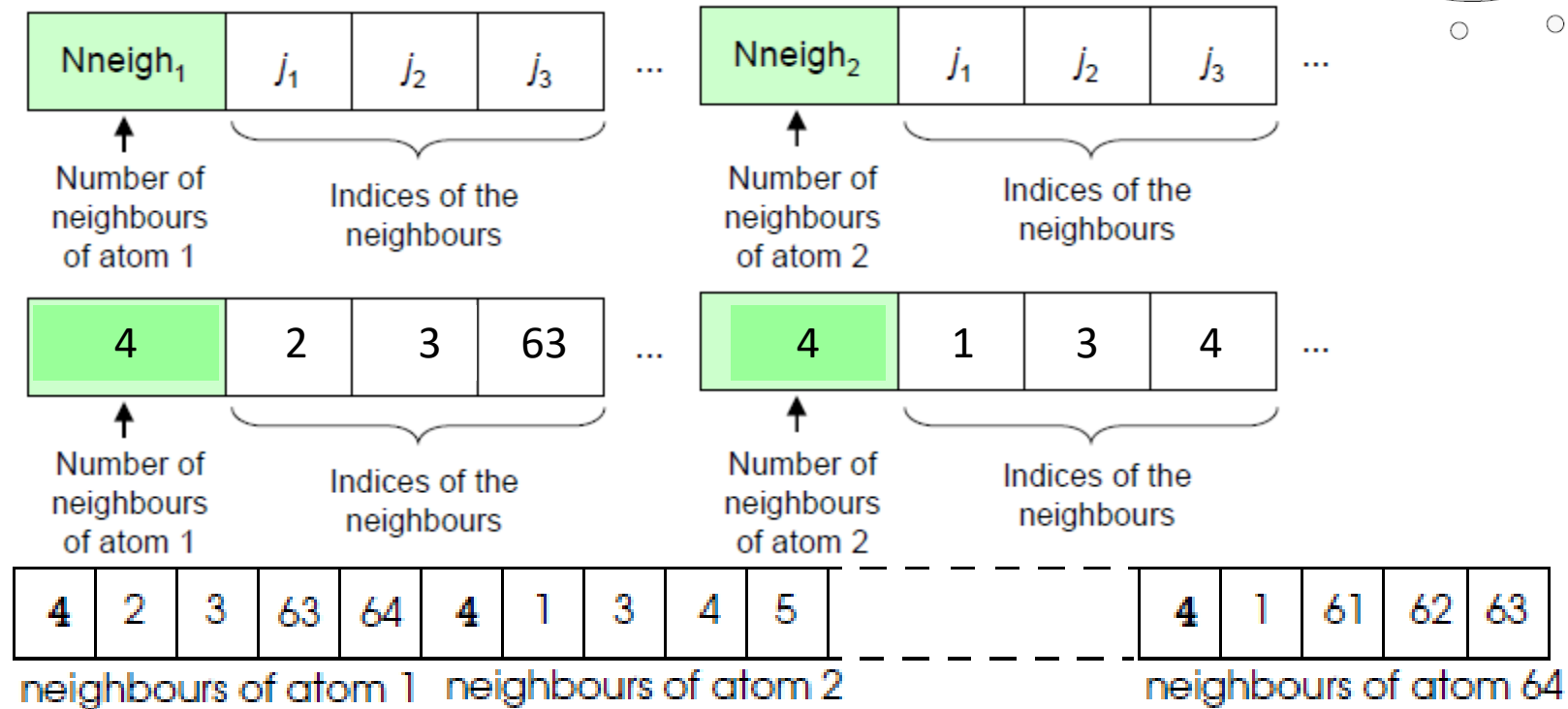
- Construction with 1 array



Verlet neighbor list: version 1



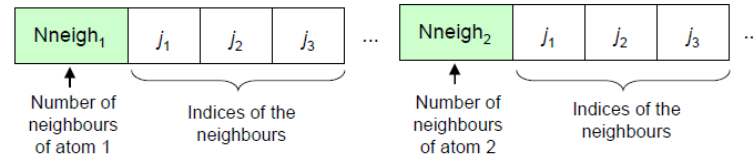
- Construction with 1 array



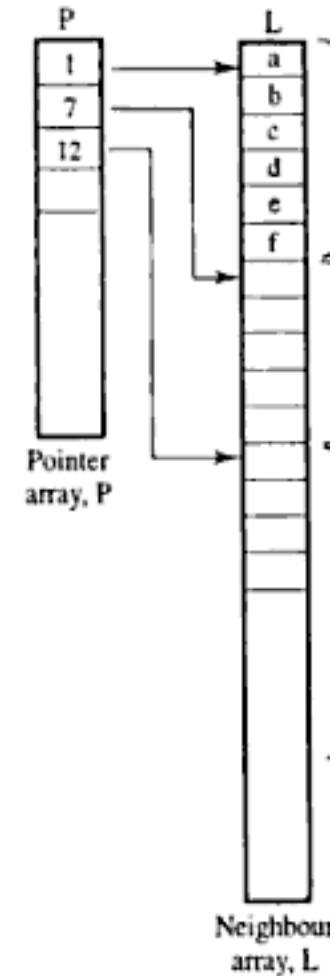
Example: 64 atom system, each atom has 4 neighbors

Verlet neighbor list: version 1

- Construction with 1 array

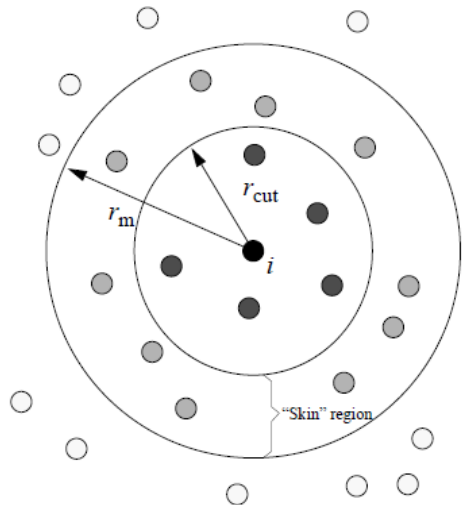


- Note: Course book uses 2 arrays for the same algorithm



Verlet neighbor list: version 2

- Construction with 1 pointer list (head) and 1 array (list)



Particle 1 has particles 8, 7, and 5 as neighbors.
Which particles are neighbors of particle 2?

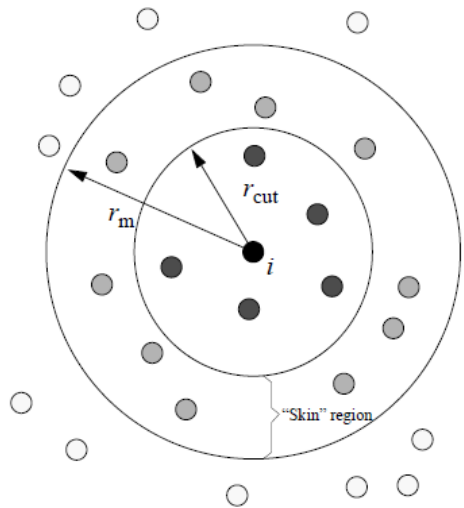


	P(head)	L (List)
1	8	0
2	10	0
3		0
4		3
5		0
6		4
7		5
8		7
9		6
10		9

A red arrow points from the value 8 in the P(head) column of row 1 to the value 7 in the L(List) column of row 8. Three orange curved arrows point from the L(List) column to the right, starting from rows 5, 6, and 7.

Verlet neighbor list: version 2

- Construction with 1 pointer list (head) and 1 array (list)

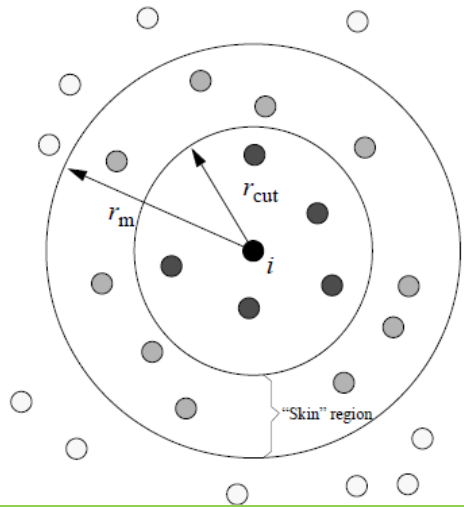


	P(head)	L (List)
1	8	0
2	10	0
3		0
4		3
5		0
6		4
7		5
8		7
9		6
10		9

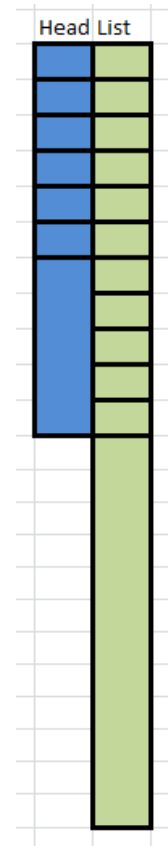
A red arrow points from the P(head) value of particle 2 (10) to the start of the list for particle 10. Blue arrows on the right indicate the sequence of particles in the list: 3, 4, 5, 6, 7, 9.

Verlet neighbor list: version 2

- Construction with 1 pointer list (head) and 1 array (list)



Particle 1 has particles 8, 7, 5, and 1 as neighbors.
Particle 2 has particles 10, 9, 6, 4, and 3 as neighbors

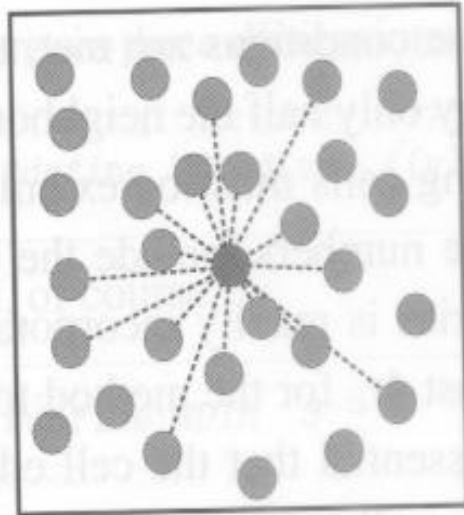


	P(head)	L (List)
1	8	0
2	10	0
3		0
4		3
5		0
6		4
7		5
8		7
9		6
10		9

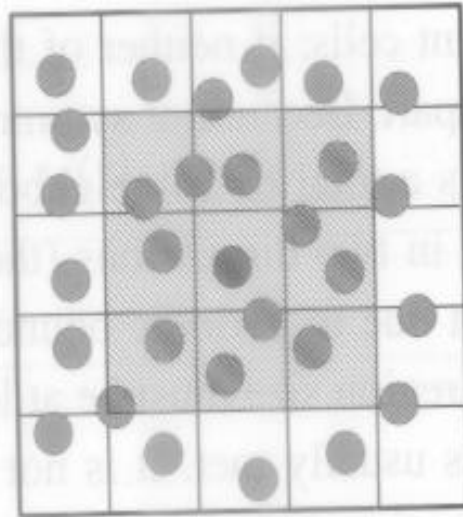
A red arrow points from the 'P(head)' column to the 'L (List)' column, starting at row 2 and ending at row 10. Blue curved arrows on the right side of the table point from the 'L (List)' column back to the 'P(head)' column, starting at row 3 and ending at row 10.

Computational efficiency: all pairs versus neighbor lists

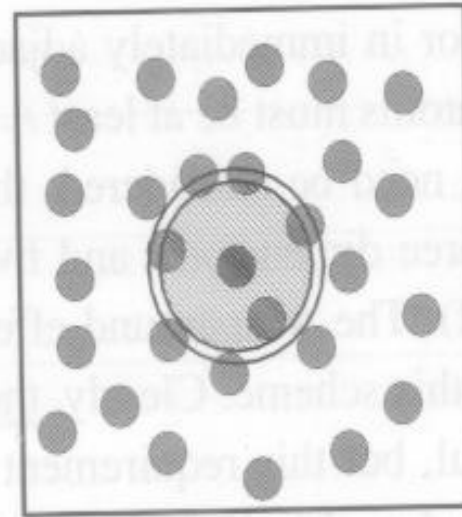
All pair method



Cell subdivision



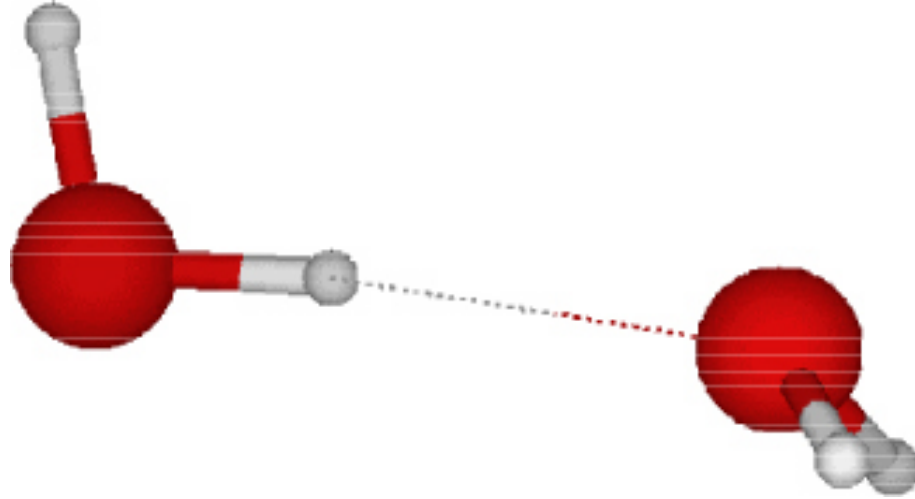
Neighbor lists



Limiting numbers of neighbors can be done either with neighbor lists or cell subdivision
Cell subdivision follows similar Verlet construction

Cut-offs

- Same neighbor list commonly used for different interactions (van der Waals cut-off, electrostatics real space cut-off,...)
- Water dimer

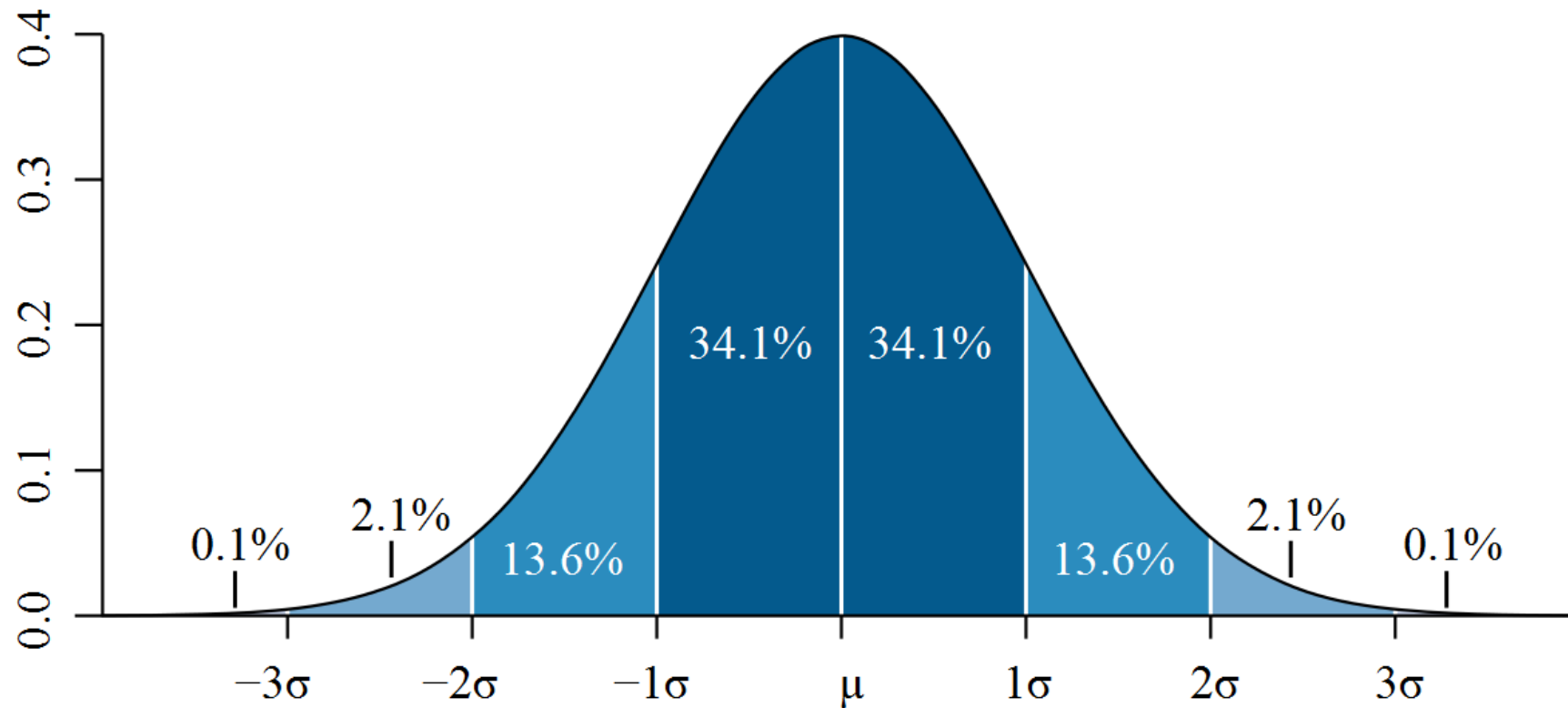


Estimating errors in simulations

- Same set of initial conditions -> same results
- But there is both systematic & statistical error
 - Simulation model
 - Algorithm
 - Time steps, truncation, shifts & algorithm modifications
 - Rounding error (numerical error)
- Even if all systematic error is eliminated, statistical error remains!!!

Estimating errors in simulations

- Standard deviation and method of blocks



Electrostatics

- Computationally, electrostatics poses a major challenge
 - long-ranged and decays as $1/r$
 - In general, we define a long-range interaction as one for which $V(r) \sim 1/r^a$, where $a < d$, and d is the dimension of space
- Cut-off, reaction-field, Ewald-type methods, multipole expansions, ...

Effect of truncating electrostatic interactions in lipid bilayer: radial distribution function

Bare truncation of Coulomb interactions
is likely to cause major error

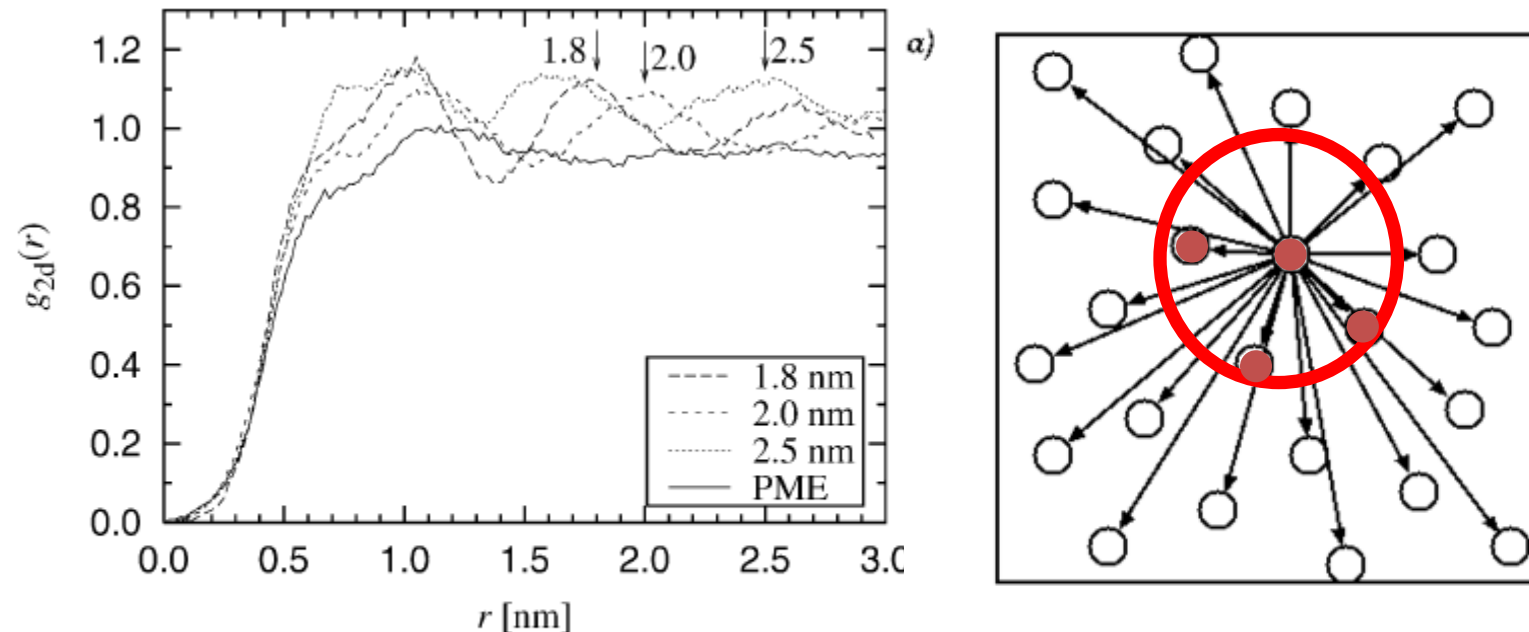


FIGURE 2 Radial distribution function $g_{2d}(r)$ for the center of mass positions of the DPPC molecules (Patra *et al.*, 2003).

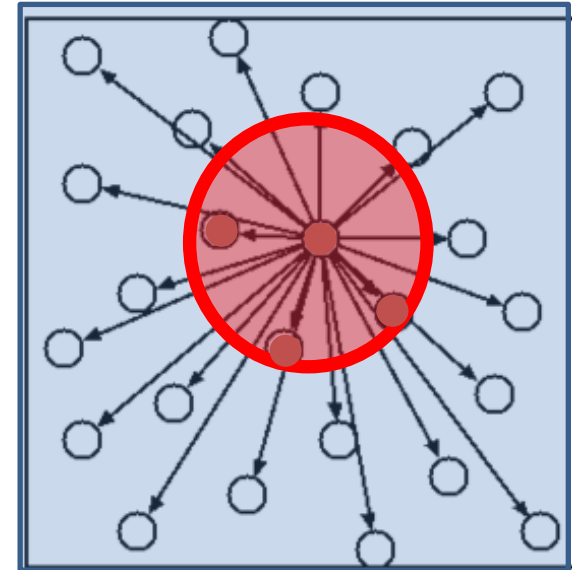
M. Patra et al., Biophys. J., 84:3636-3645, 2003

Reaction field electrostatics

- Explicit electrostatics with $r < r_{\text{cut}}$.
- For $r > r_{\text{cut}}$ the system is treated on a mean-field level and is thus completely described by its dielectric constant ϵ .

$$\mathcal{V}(r) = \frac{q_i q_j}{4\pi\epsilon_0 r} \left[1 + \frac{\epsilon - 1}{2\epsilon + 1} \left(\frac{r}{r_{\text{cut}}} \right)^3 \right] - \frac{q_i q_j}{4\pi\epsilon_0 r_{\text{cut}}} \frac{3\epsilon}{2\epsilon + 1},$$

for $r \leq r_{\text{cut}}$.



Ewald summation

- Ewald converted 1927 the slowly, conditionally convergent sum for the Coulomb potential in infinite lattice into two sums that converge rapidly and absolutely, one in real space another in reciprocal space

$$\frac{1}{r} = \frac{f(r)}{r} + \frac{1 - f(r)}{r}$$

Ewald sum: periodicity

A.Y. Toukmaji, J.A. Board Jr. / Computer Physics Communications 95 (1996) 73–92

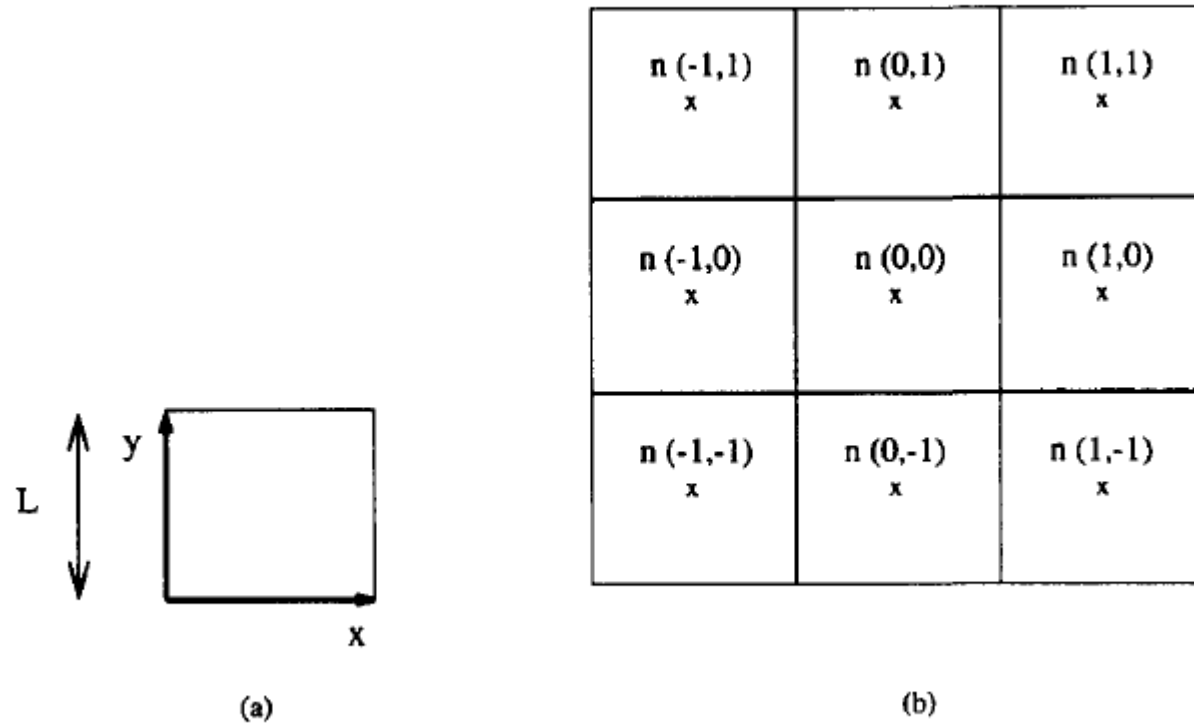


Fig. 1. In a 2D system (a) the unit cell coordinates and (b) a 3×3 periodic lattice built from unit cells.

Ewald sum

$\frac{n(-1,1)}{x}$	$\frac{n(0,1)}{x}$	$\frac{n(1,1)}{x}$
$\frac{n(-1,0)}{x}$	$\frac{n(0,0)}{x}$	$\frac{n(1,0)}{x}$
$\frac{n(-1,-1)}{x}$	$\frac{n(0,-1)}{x}$	$\frac{n(1,-1)}{x}$

$$U = \frac{1}{2} \sum_n \sum_{i=1}^N \sum_{j=1}^N \frac{q_i q_j}{r_{ij,n}},$$

- $U_{\text{Ewald}} = U^r + U^m + U^0$
 - U^r Real space sum
 - U^m Reciprocal space sum
 - U^0 Constant term

$$U^r = \frac{1}{2} \sum_{i,j} \sum_n q_i q_j \frac{\text{erfc}(\alpha r_{ij,n})}{r_{ij,n}},$$

$$U^m = \frac{1}{2\pi V} \sum_{i,j} q_i q_j \sum_{\mathbf{m} \neq \mathbf{0}} \frac{\exp(-(\pi \mathbf{m} / \alpha)^2 + 2\pi i \mathbf{m} \cdot (\mathbf{r}_i - \mathbf{r}_j))}{m^2},$$

$$U^0 = \frac{-\alpha}{\sqrt{\pi}} \sum_{i=1}^N q_i^2.$$

V is the volume of the simulation box, $\mathbf{m} = (l, j, k)$ is a reciprocal-space vector, and \mathbf{n} was defined earlier. The self-term U^0 is a correction term that cancels out the interaction of each of the introduced artificial counter-charges with itself as will be explained in Section 2.2. The complimentary error function decreases monotonically as x increases and is defined by $\text{erfc}(x) = 1 - \text{erf}(x) = 1 - (2/\sqrt{\pi}) \int_0^x e^{-u^2} du$. The theory of Ewald summation is described in more detail by Kittel [33] and Tosi [51].

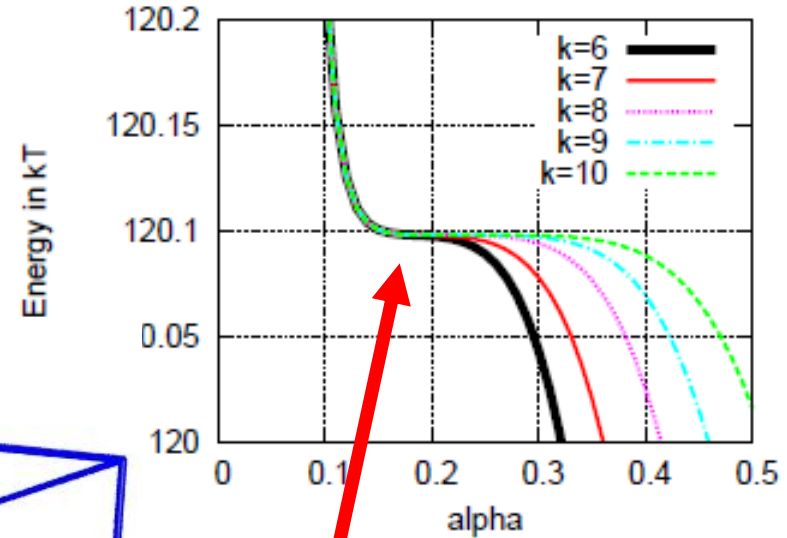
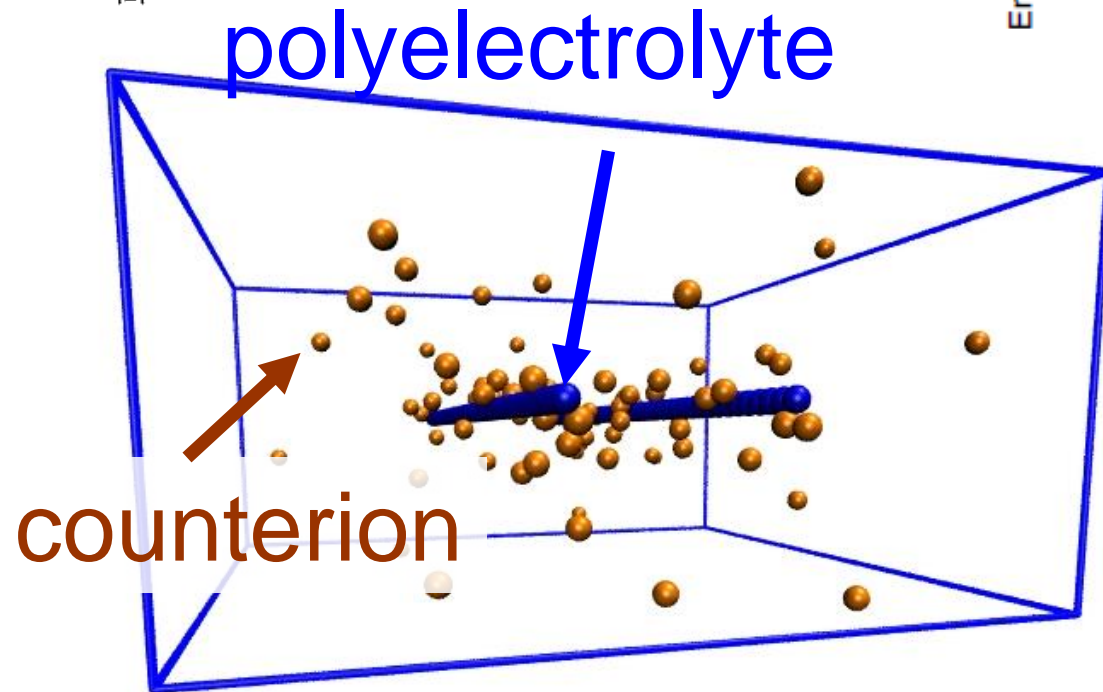
A.Y. Toukmaji, J.A. Board Jr./Computer Physics Communications 95 (1996) 73-92

Ewald summation convergence: Example

$$U^r = \frac{1}{2} \sum_{i,j}^{N'} \sum_{\mathbf{n}} q_i q_j \frac{\operatorname{erfc}(\alpha r_{ij,n})}{r_{ij,n}},$$

$$U^m = \frac{1}{2\pi V} \sum_{i,j}^N q_i q_j \sum_{\mathbf{m} \neq \mathbf{0}} \frac{\exp(-(\pi \mathbf{m} / \alpha)^2 + 2\pi i \mathbf{m} \cdot (\mathbf{r}_i - \mathbf{r}_j))}{m^2},$$

$$U^o = \frac{-\alpha}{\sqrt{\pi}} \sum_{i=1}^N q_i^2.$$



Convergence
Region (plateau)

Electrostatics

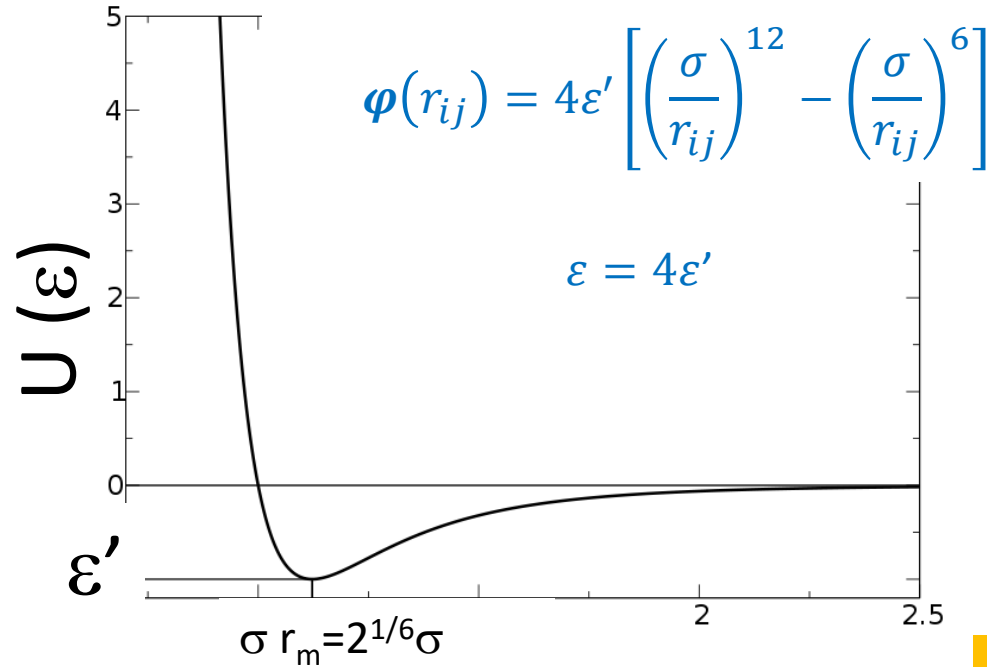
- Computationally, electrostatics poses a major challenge
 - long-ranged and decays as $1/r$
 - In general, we define a long-range interaction as one for which $V(r) \sim 1/r^a$, where $a < d$, and d is the dimension of space
- Cut-off, reaction-field, Ewald-type methods, multipole expansions, ...

Dimensionless units

- Advantages
 - numerical values ~ 1 , instead of typically very small values associated with atomic scale
 - simplification of equations of motion (absorption of parameters defining the model into units)
 - possibility of scaling results of single simulation for a whole class of systems described by same model

Dimensionless units

Example: Lennard-Jones 12-6



Property	Reduced Form
Length	$r^* = r/\sigma$
Time	$t^* = t/\tau = t(\epsilon/m\sigma^2)^{1/2}$
Temperature	$T^* = k_B T/\epsilon$
Force	$f^* = f\sigma/\epsilon$
Energy	$\phi^* = \phi/\epsilon$
Pressure	$P^* = P\sigma^3/\epsilon$
Number density	$N^* = N\sigma^3$
Density	$\rho^* = \sigma^3 \rho/m$
Surface tension	$\gamma^* = \gamma\sigma^2/\epsilon$

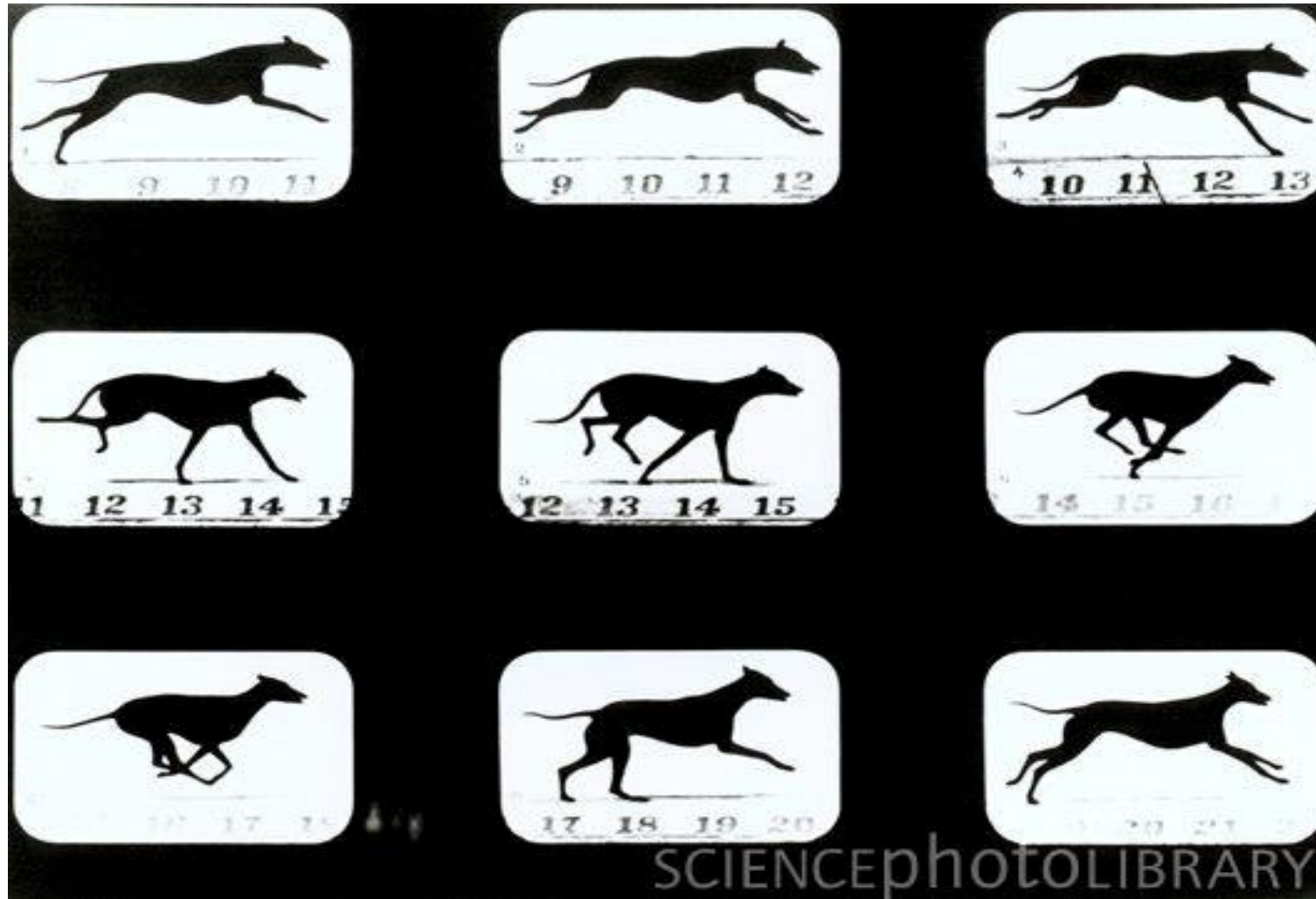
$$\phi^*(r^*) = \left[\left(\frac{1}{r^*} \right)^{12} - \left(\frac{1}{r^*} \right)^6 \right]$$

Initial velocities for MD simulations

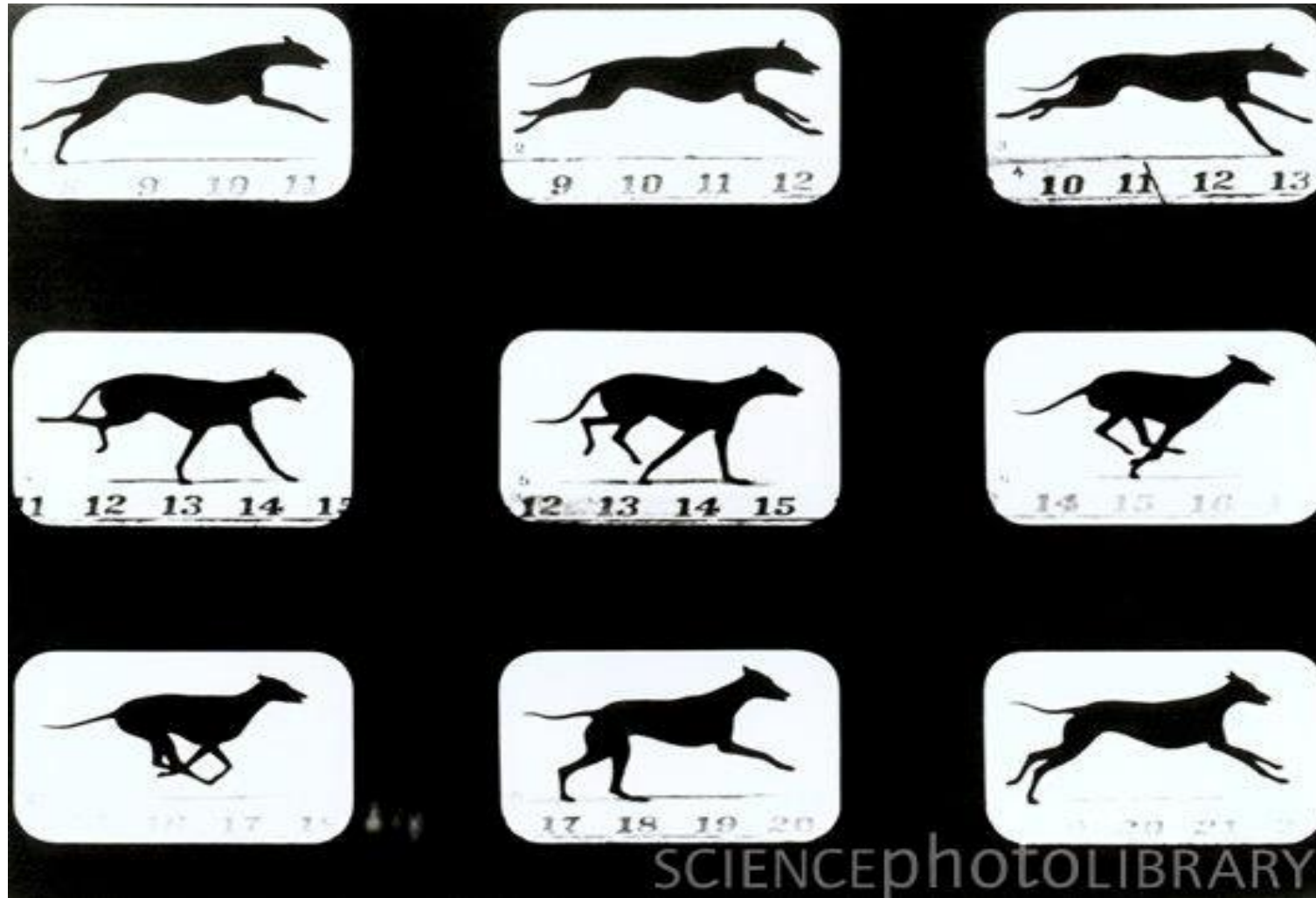
- Random initial velocities such that total momentum vanishes
average conforms to desired temperature
- Initial distribution of velocity components may be
 - Uniform between $-v_{\min}$ and $+v_{\max}$
 - Gaussian:

$$\rho(v_{i,x}) = \sqrt{\frac{m_i}{2\pi k_B T}} \exp\left\{-\frac{1}{2} m_i v_{i,x}^2 / (k_B T)\right\}$$

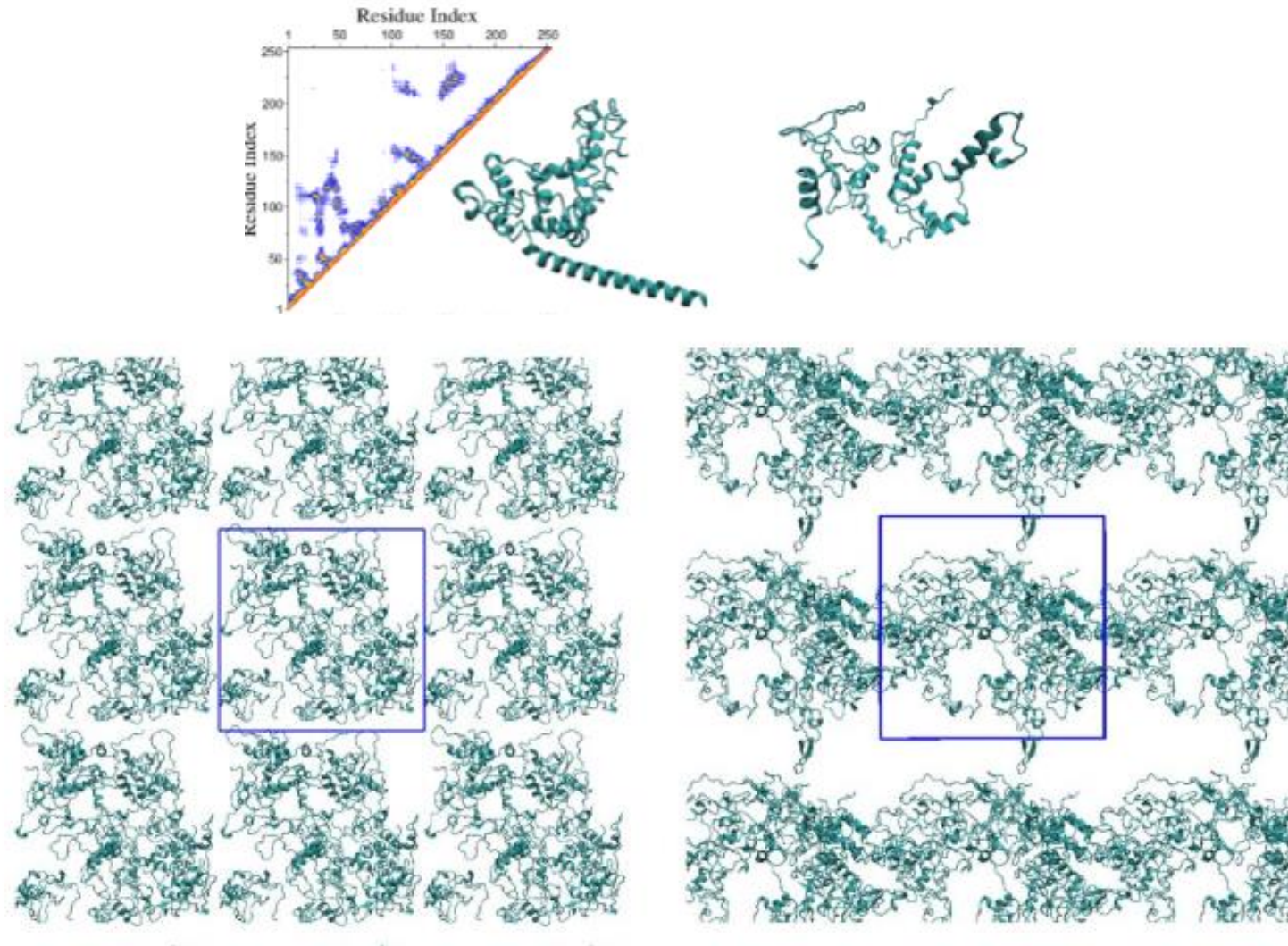
Molecular dynamics in brief: sequence of static images



Note: Average may not be representative



Example: Sup35 protein configurations and periodicity



Movies for visualizing molecular modelling

- Materials simulations (metals, surfaces, shear flows, liquids, some molecular materials...)
 - <http://lammps.sandia.gov/movies.html>
- Biomolecules
 - <http://www.ks.uiuc.edu/Gallery/Movies/>

Example: Sodium dodecyl sulfate

