Molecular Dynamics Simulation

Prabal K Maiti
Department of Physics
http://www.physics.iisc.ernet.in/~maiti

- Introduction (Basic facts and some history)
- Molecular dynamics, Various schemes for integration,
- Inter and Intra molecular forces, Various ensemble (NVE, NVT, NPT, NPH), atomic charge derivation scheme
- How to make the simulation efficient (Cell List, Neighbor List), Periodic Boundary condition, computing long range interactions
- Advanced free energy calculation methods
- Monte Carlo simulation
- Application: Nanotube, graphene, Liquid Crystal, DNA/protein simulations

Ref:

Computer simulation of Liquids: M. P. Allen and D. J. Tildesley, Oxford (1987)

Understanding Molecular simulation: Daan Frenkel and B Smit (2nd ed)

Molecular Modelling Principles And Applications: Andrew Leach, Prentice Hall (2001)

The art of Molecular dynamics: D. C.Rappaport

Molecular Modeling and Simulation: Tamar Schlick

Introduction to Molecular dynamics: Hemant Kumar and Prabal K Maiti

Web: CCP5 library http://www.ccp5.ac.uk

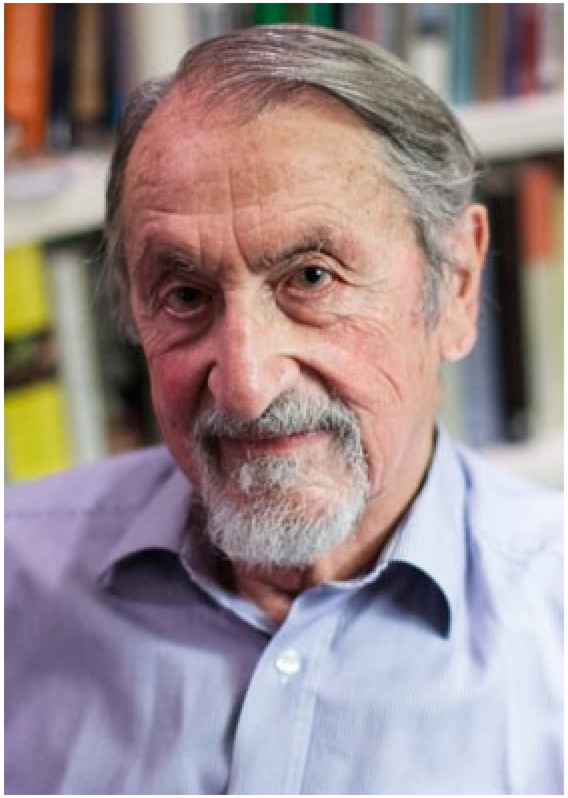
Acknowledgement: **http://www.cheme.buffalo.edu/courses/ce530**

Grading:

1. **5-6 Assignments**
2. **2 individual assignment (need to do some simulation work)**
3. **Term paper (on some related topics covered in the course. Topics to be discussed with the instructor)**
4. **One exam to be decided later**

Course TA: Dr. Ashok Garai email: ashok.garai@gmail.com
Dr. Sudip Chakraborty email: sudip.hammer@gmail.com

Nobel Prize in Chemistry 2013



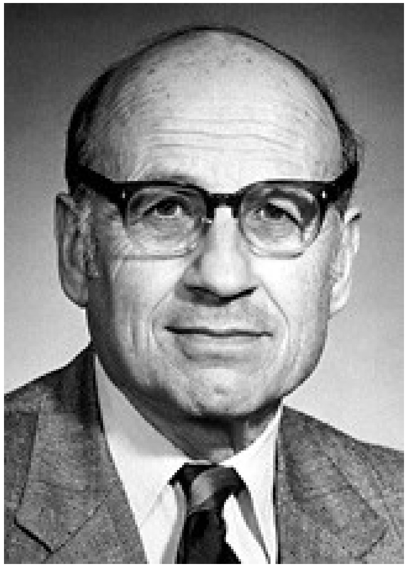
Martin Karplus, Chem.
Harvard

M Levitt, Str Bio
& Computer
Science
Stanford

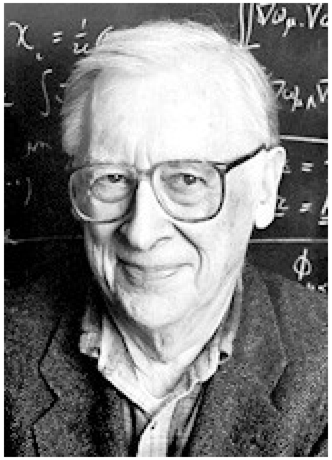
A Warshel,
Chem.
USC

"for the development of multiscale models for complex chemical systems"

Nobel Prize in Chemistry 1998



Walter Kohn
Physics, University of California, Santa Barbara,



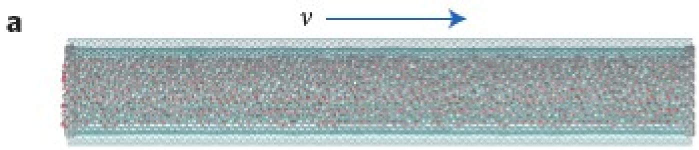
John A. Pople
Chemistry,Northwestern University, Evanston

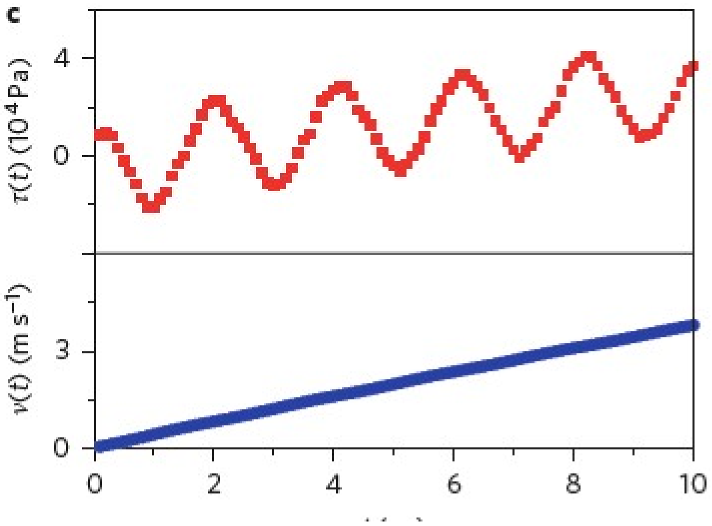
Walter Kohn "for his development of the density-functional theory"**and John A. Pople *"for his development of computational methods in quantum chemistry"*.**

Nanotechnology/Nanofluidics

LETTERS
PUBLISHED ONLINE: 6 JULY 2015 | DOI: 10.1038/NNANO.2015.134
nature nanotechnology

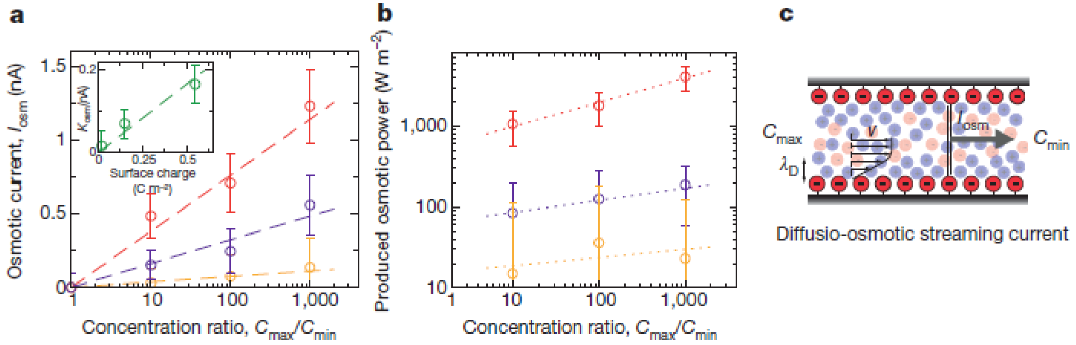
Water transport inside carbon nanotubes mediated by phonon-induced oscillating friction





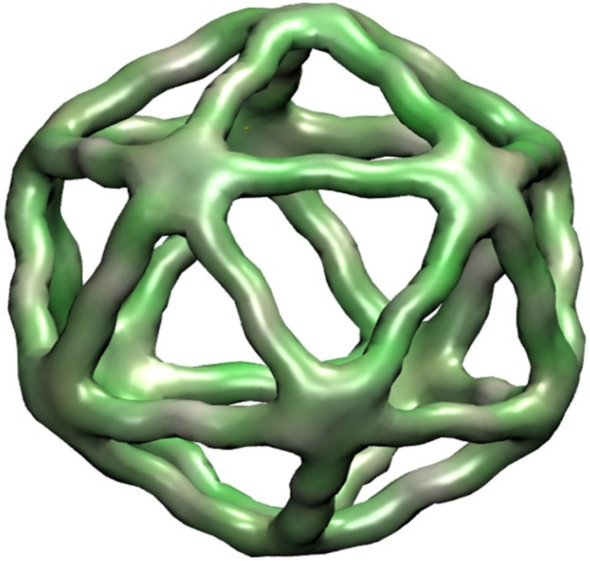
LETTER
doi:10.1038/nature11876

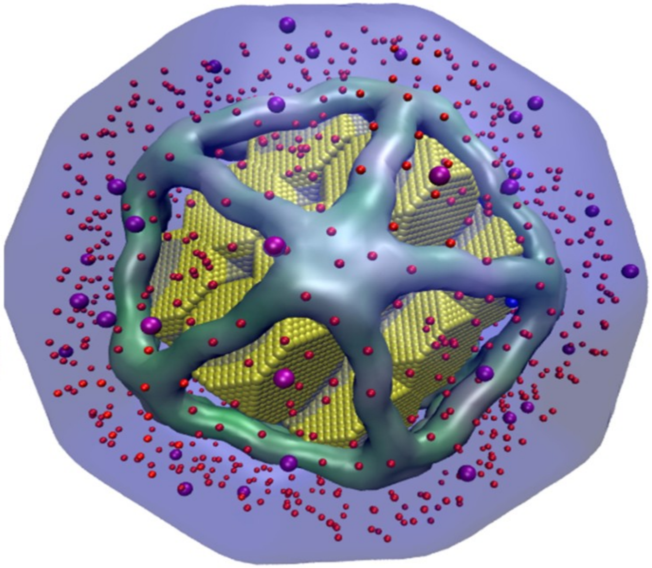
Giant osmotic energy conversion measured in a single transmembrane boron nitride nanotube

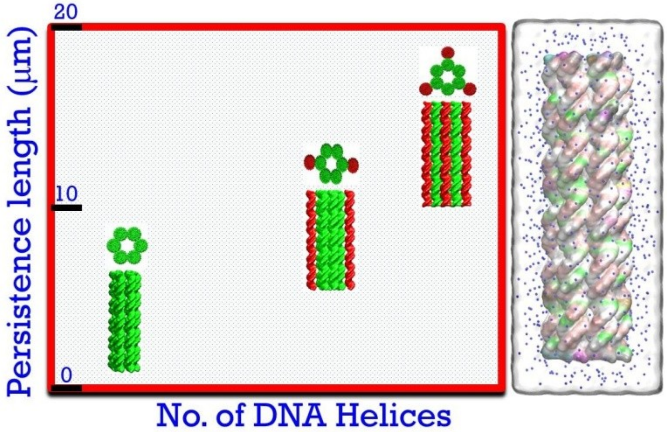


Diffusio-osmotic streaming current

DNA nanotechnology

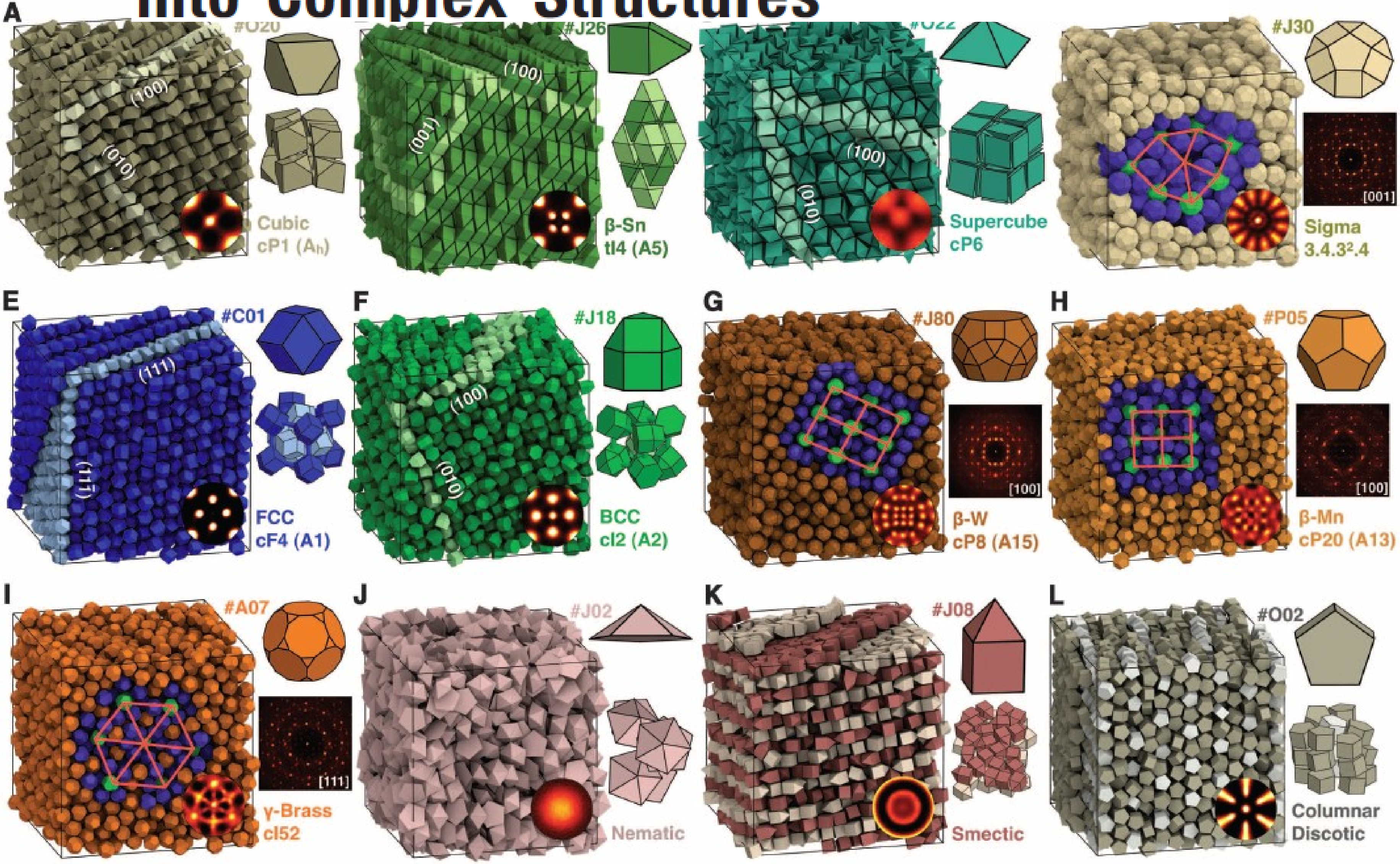




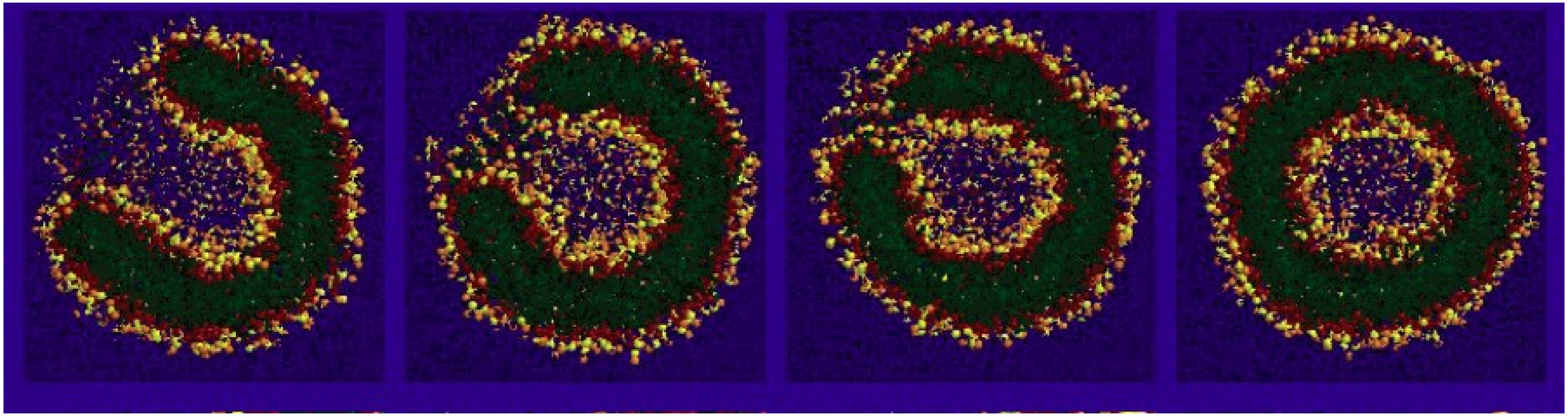


Self-assembly

Predictive Self-Assembly of Polyhedra into Complex Structures



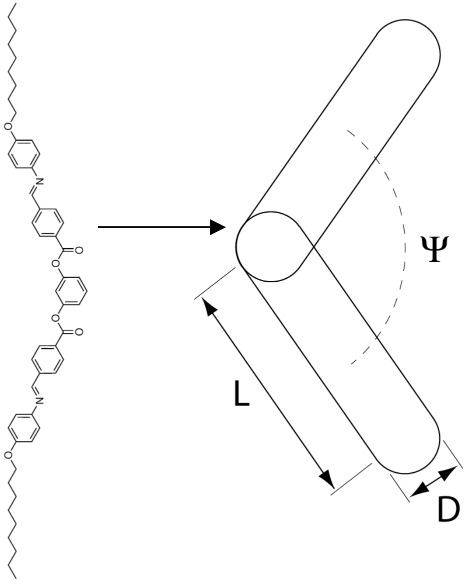
Molecular Dynamics Simulation of the Formation, Structure, and Dynamics of Small Phospholipid Vesicles

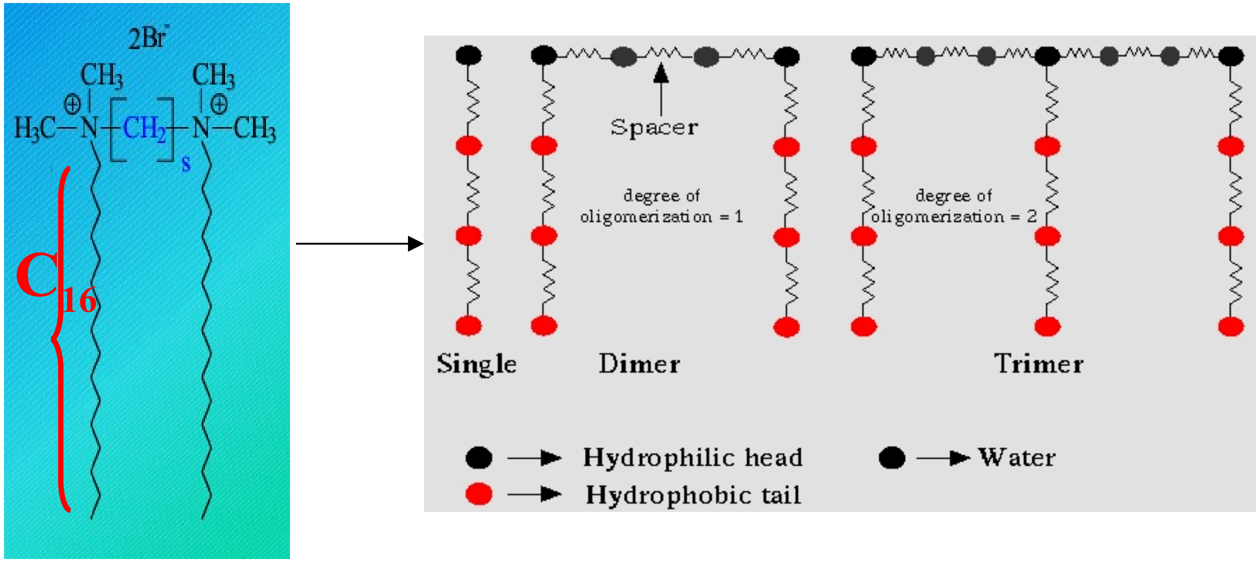


What is molecular simulation?

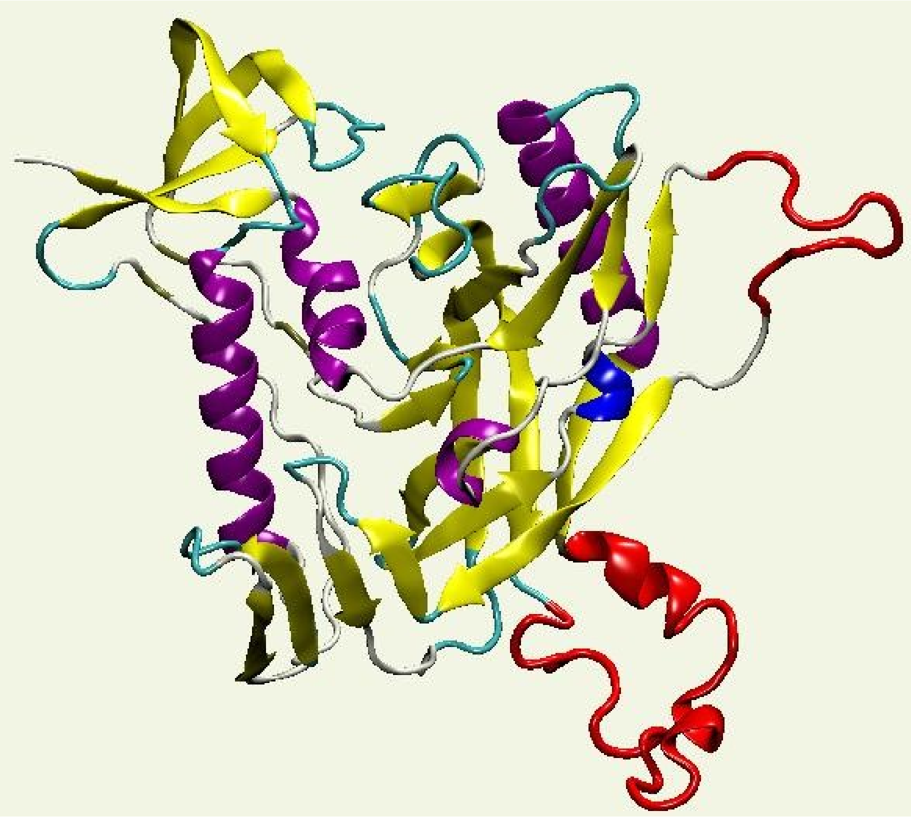
Molecular modeling is the science and art of studying molecular structure and function through model building and computation.

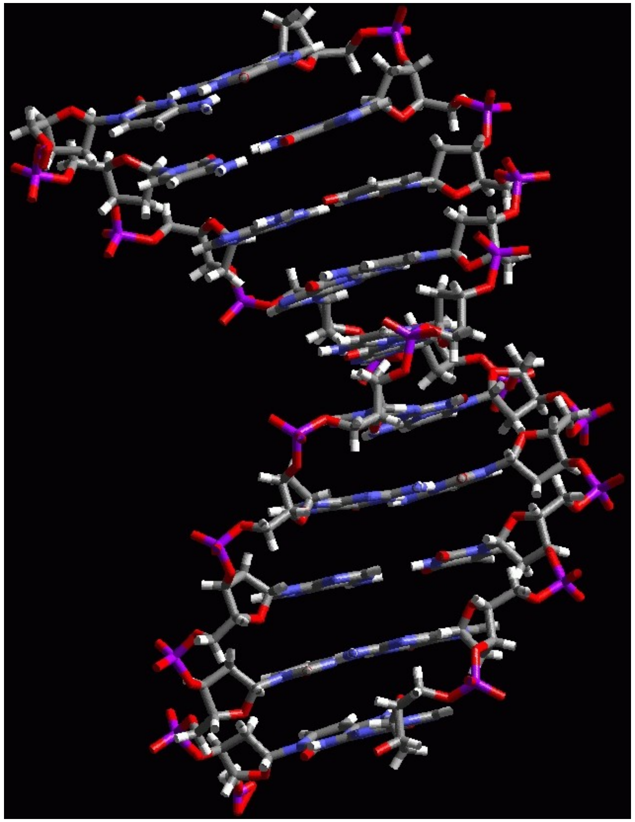
Model building could be as simple as representing molecule by hard/soft sphere (beads) , rigid rods, or other geometrical shape, sphere/beds connected through springs or molecule with full chemical details.





Molecular model of bio-molecules





What is molecular simulation?

Computation can be carried out using following methods:

Molecular mechanics, molecular dynamics, Monte Carlo, Free energy and solvation methods, structure/activity relationship (SAR) and many other established procedure.

Need for Molecular simulation

- There are no general method for a solving complex many-body problems. Hamiltonian is unknown, until we solve the quantum many-body problem! In fact in most cases not possible and requires lots of approximations.
- Molecular simulations are the only possible solution for such complex many body systems.
- In many cases experiments are limited and expensive. Simulations can complement the experiment.
- Simulation can give molecular level understanding even at a single molecule level

Simulation Methodology

Semi-empirical

Give us the phenomena and invent a model to mimic the problem.

ab initio methods

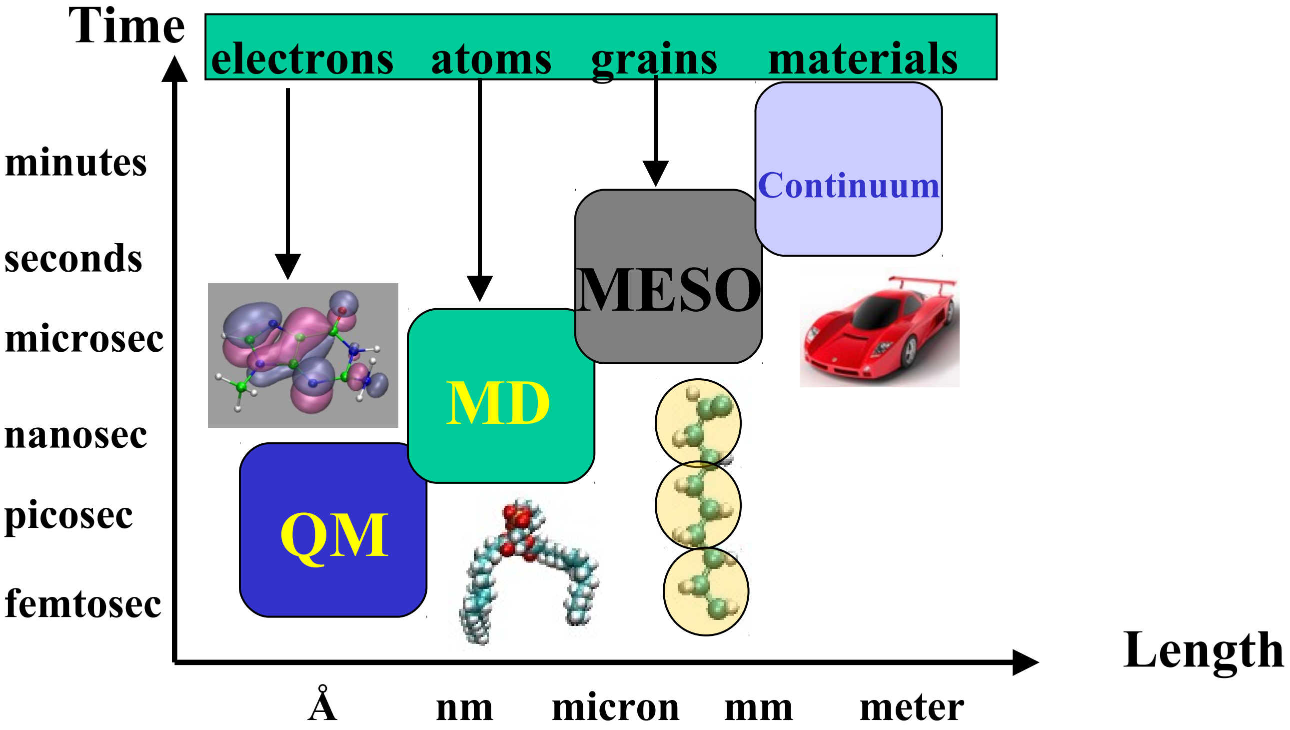
Maxwell, Boltzmann and Schrödinger gave us the model. All we must do is numerically solve the mathematical problem and determine the properties.

These two approaches can be combined to make what is termed as Multi-scale modeling strategies

“The general theory of quantum mechanics is now almost complete. The underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are thus completely known, and the difficulty is only that the exact application of these laws leads to equations much too complicated to be soluble.”

Dirac, 1929

Multi-scale Modeling strategy



High quality multi-scale simulations

- Quantum Mechanical calculations
- First Principles force fields
- Large scale Molecular Dyanmics (MD) simulations
- Mesoscopic modeling (Coarse-grained MD, DPD, BD)
- Macroscopic modeling (finite elements, continuum simulations, Lattice Boltzmann)

What size is too big? What times are too long?

- **QM** (ab initio molecular dynamics)

electrons / N basis sets, speed ~ N^3 or N^4 (may be linear with very high prefactor)

Time steps ~10^{-2} fs.

Example of big and long: 64-256 water molecules during 100 ps.

- **Classical atomistic MD**

Atoms/ N atoms, speed ~N^2 (may be reduced with efficient algorithms, periodic coulomb is most expensive)

Time steps ~0.5-2 fs.

Example of big and long (1 processor): 50,000 atoms for 1 ns.

(parallel simulations, can improve this much. Parallel codes available free or almost free: LAMMPS, AMBER, GROMACS, NAMD)

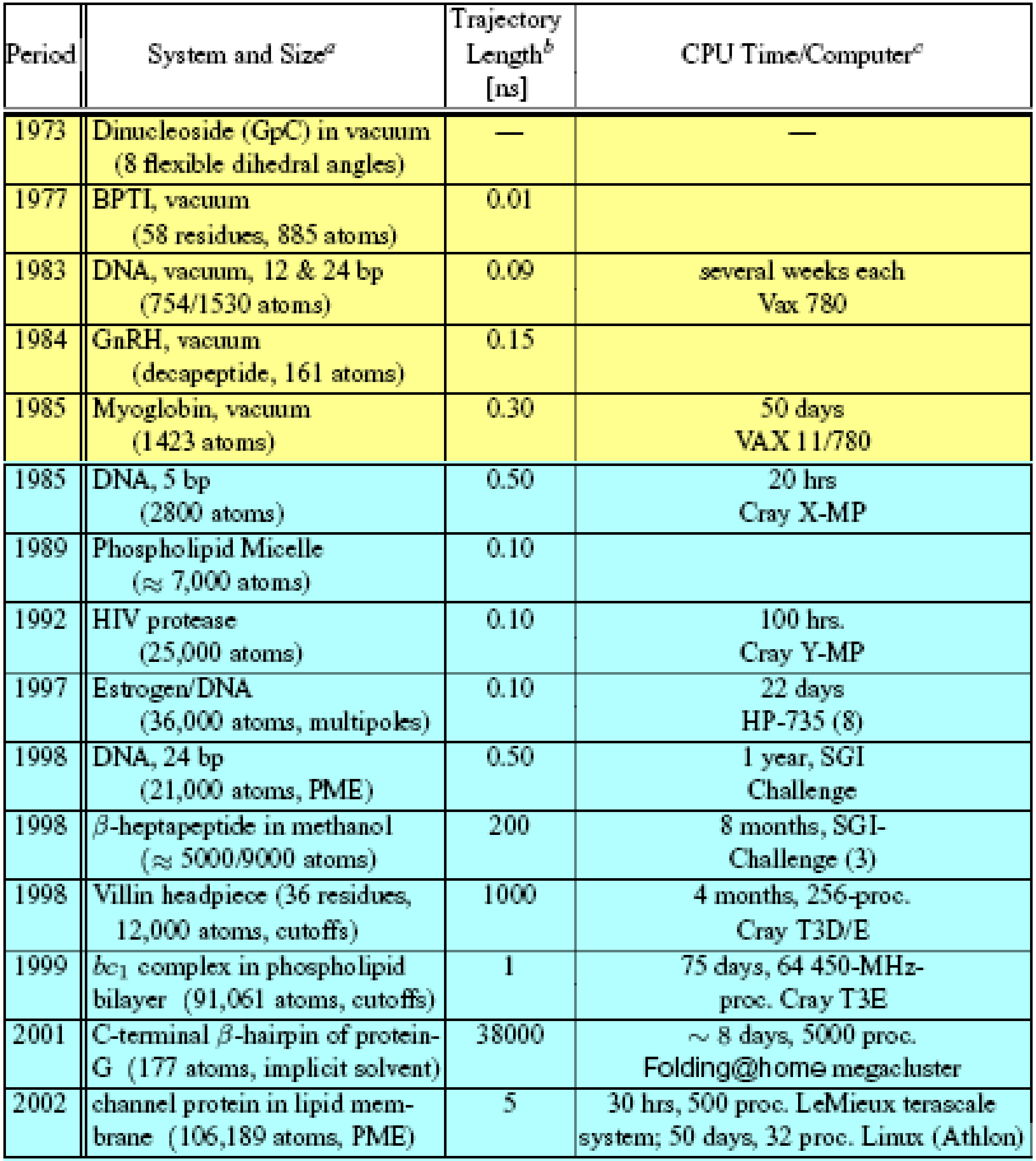
Short history of Molecular Simulations

- Metropolis, Rosenbluth, Teller (1953) Monte Carlo Simulation of hard disks.
- Fermi, Pasta Ulam (1954) experiment on ergodicity
- Alder & Wainwright (1958) liquid-solid transition in hard spheres. “long time tails” (1970)
- Vineyard (1960) Radiation damage using MD
- Rahman (1964) liquid argon, water(1971)
- Verlet (1967) Correlation functions, ...
- Andersen, Rahman, Parrinello (1980) constant pressure MD
- Nose, Hoover, (1983) constant temperature thermostats.
- Car, Parrinello (1985) ab initio MD.

The examples for each period are representative. The first five systems are modeled in vacuum and the others in solvent.

The 38 μs β-hairpin simulation in 2001 represents an ensemble (or aggregate dynamics) simulation, as accumulated over several short runs, rather than a long simulation.

The table is taken from the book by Tamar Schlick



Period	System and Size[a]	Trajectory Length[b] [ns]	CPU Time/Computer[c]
1973	Dinucleoside (GpC) in vacuum (8 flexible dihedral angles)	—	—
1977	BPTI, vacuum (58 residues, 885 atoms)	0.01	
1983	DNA, vacuum, 12 & 24 bp (754/1530 atoms)	0.09	several weeks each Vax 780
1984	GnRH, vacuum (decapeptide, 161 atoms)	0.15	
1985	Myoglobin, vacuum (1423 atoms)	0.30	50 days VAX 11/780
1985	DNA, 5 bp (2800 atoms)	0.50	20 hrs Cray X-MP
1989	Phospholipid Micelle (≈ 7,000 atoms)	0.10	
1992	HIV protease (25,000 atoms)	0.10	100 hrs. Cray Y-MP
1997	Estrogen/DNA (36,000 atoms, multipoles)	0.10	22 days HP-735 (8)
1998	DNA, 24 bp (21,000 atoms, PME)	0.50	1 year, SGI Challenge
1998	β-heptapeptide in methanol (≈ 5000/9000 atoms)	200	8 months, SGI-Challenge (3)
1998	Villin headpiece (36 residues, 12,000 atoms, cutoffs)	1000	4 months, 256-proc. Cray T3D/E
1999	bc_1 complex in phospholipid bilayer (91,061 atoms, cutoffs)	1	75 days, 64 450-MHz-proc. Cray T3E
2001	C-terminal β-hairpin of protein-G (177 atoms, implicit solvent)	38000	~ 8 days, 5000 proc. Folding@home megacluster
2002	channel protein in lipid membrane (106,189 atoms, PME)	5	30 hrs, 500 proc. LeMieux terascale system; 50 days, 32 proc. Linux (Athlon)

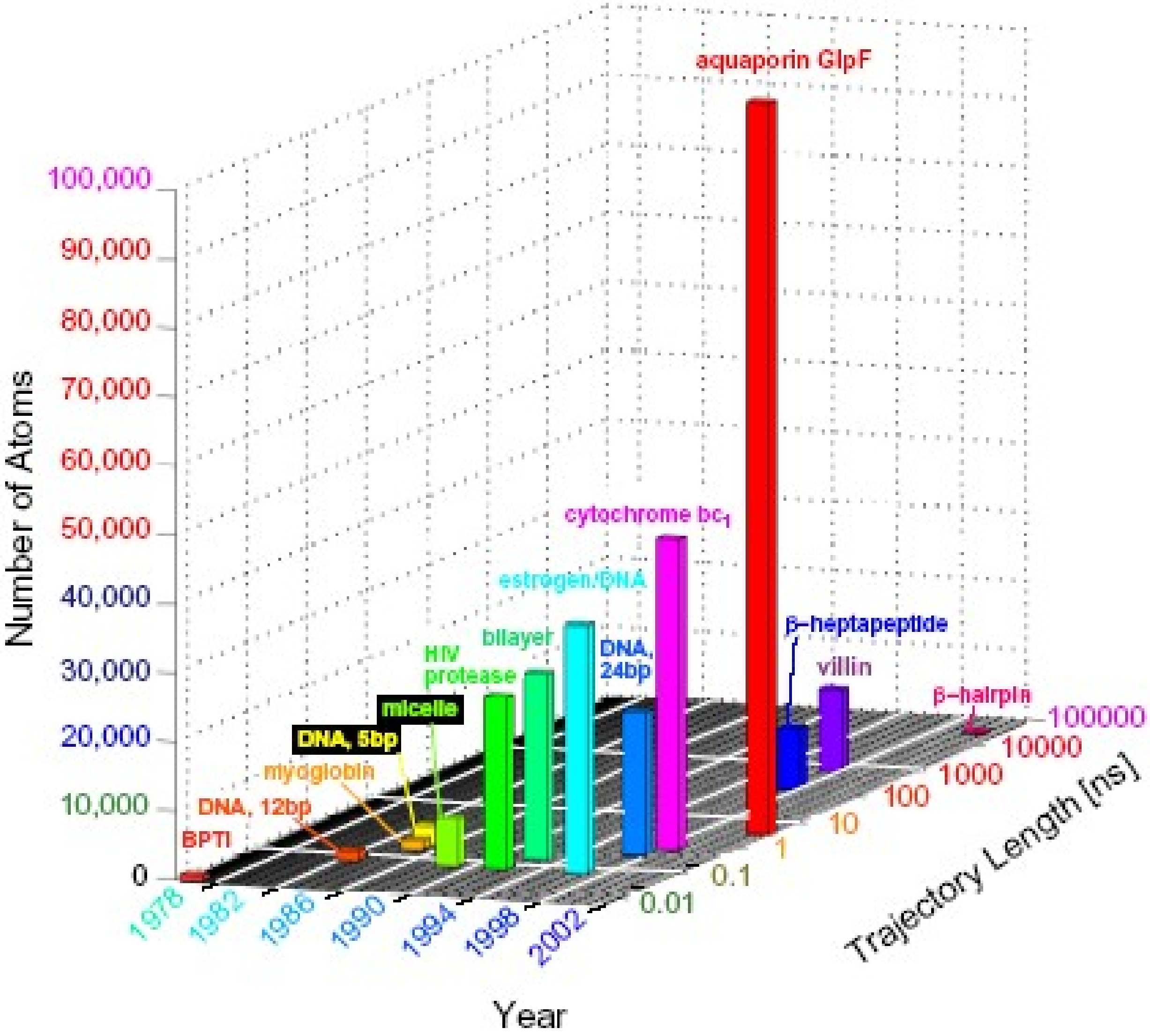


Figure 1.1. The evolution of molecular dynamics simulations with respect to system sizes and simulation lengths (see also Table 1.2).

Flops= Number of Cores/cpus $*$ Average frequency $*$ Operations per cycle
Laptop with 2.5 GHz single processor will have 10 Gflops (4 operation per cycle)

RANK	SITE	SYSTEM	CORES	RMAX (TFLOP/S)	RPEAK (TFLOP/S)	POWER (KW)
1	National Super Computer Center in Guangzhou China	**Tianhe-2**, Intel Xeon Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
2	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray Cray Inc.	560,640	17,590.0	27,112.5	8,209
3	DOE/NNSA/LLNL United States	**Sequoia** - IBM	1,572,864	17,173.2	20,132.7	7,890
4	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 Fujitsu	705,024	10,510.0	11,280.4	12,660
5	DOE/SC/Argonne National Laboratory United States	**Mira** - BlueGene/Q, IBM	786,432	8,586.6	10,066.3	3,945

http://www.top500.org/lists/2014/11/

Vendors	Count
HP	179
IBM	153
Cray Inc.	62
SGI	23
Bull	18
Dell	9
Fujitsu	8
NUDT	5
RSC Group	4
Atipa	3
Others	39

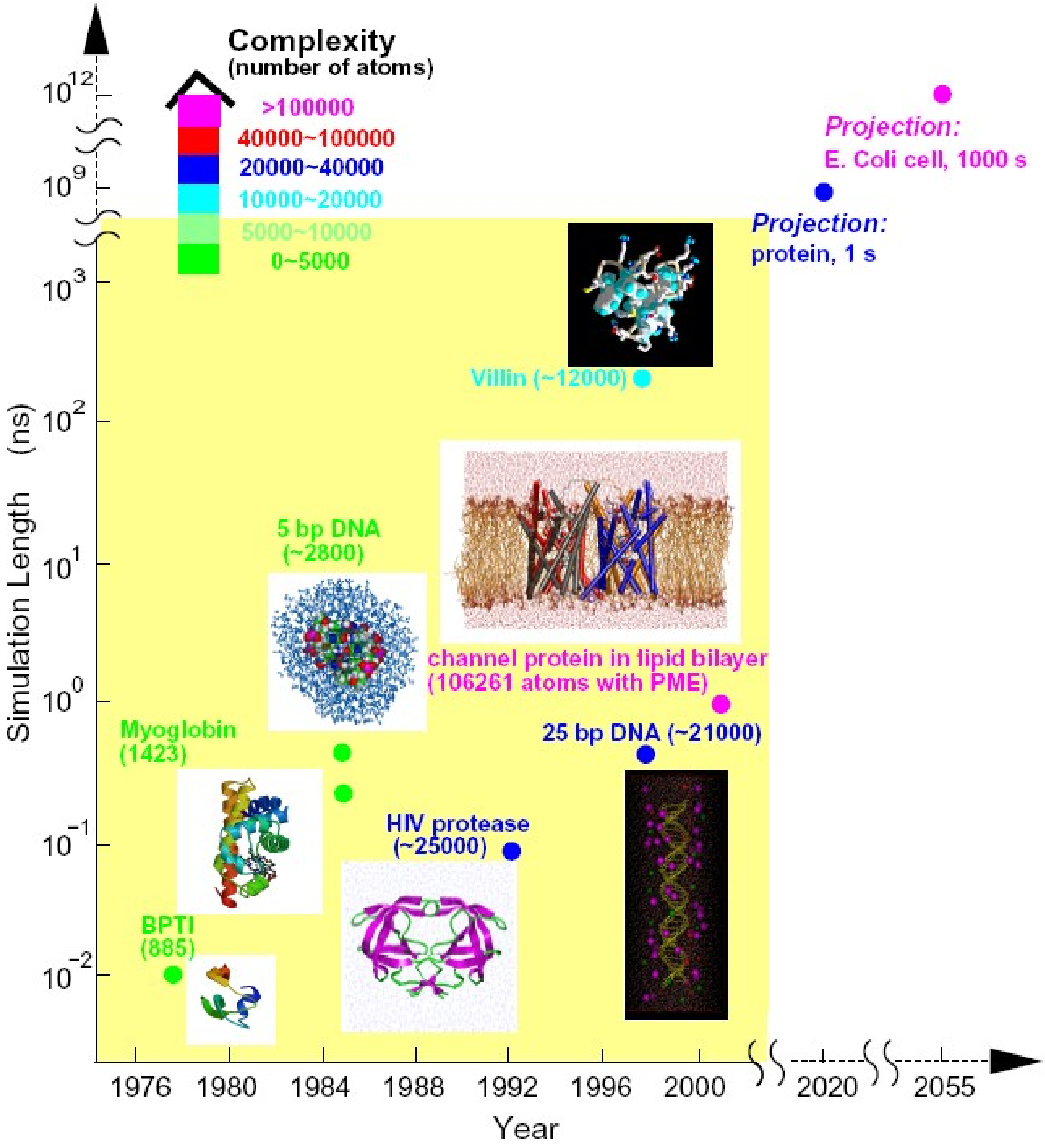


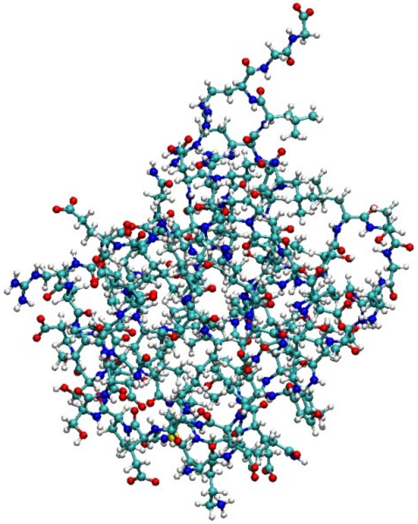
Figure 1.2. The evolution of molecular dynamics simulations with respect to simulation lengths (see also Table 1.2 and Figure 1.1). The data points for 2020 and 2055 represent extrapolations from the 1977 BPTI [136] and 1998 villin [57, 55] simulations, assuming a computational power increase by a factor of 10 every 3–4 years, as reported in [56].

Molecular Dynamics (MD)

- Pick particles, masses and potential.
- Initialize positions and momentum. (boundary conditions in space and time)
- Solve **F** = m **a** to determine **r**(t), **v**(t).

Newton (1667-87)

- Compute properties along the trajectory
- Estimate errors.
- Try to use the simulation to answer physical questions.



What are the forces?

$$\frac{d^2 r}{dt^2} = -\nabla V(\vec{r})$$

- Crucial since V(r) determines the quality of result.
- *Semi-empirical* potentials: potential is constructed on theoretical grounds but using some experimental data.
- Common examples are Lennard-Jones, Coulomb, embedded atom potentials. They are only good for simple materials. The *ab initio* philosophy is that potentials are to be determined directly from quantum mechanics as needed.
- But computer power is not yet adequate in general.
- A powerful approach is to use simulations at one level to determine parameters at the next level.

Unifying QM with molecular dynamics

The Hamiltonian Operator,

$$H = -\sum_I \frac{\hbar^2}{2M_I}\nabla_I^2 - \sum_i \frac{\hbar^2}{2m_e}\nabla_i^2 + \frac{1}{4\pi\varepsilon_0}\sum_{i<j}\frac{e^2}{\left|\vec{r}_i - \vec{r}_j\right|}$$

$$-\frac{1}{4\pi\varepsilon_0}\sum_{i<j}\frac{e^2 Z_I}{\left|\vec{R}_I - \vec{r}_i\right|} + \frac{1}{4\pi\varepsilon_0}\sum_{I<J}\frac{e^2 Z_I Z_J}{\left|\vec{R}_I - \vec{R}_J\right|}$$

$$= -\sum_I \frac{\hbar^2}{2M_I}\nabla_I^2 - \sum_i \frac{\hbar^2}{2m_e}\nabla_i^2 + V_{n-e}\left(\{r_i\},\{R_I\}\right)$$

$$= -\sum_I \frac{\hbar^2}{2M_I}\nabla_I^2 + H_e\left(\{r_i\},\{R_I\}\right)$$

Ref. D. Marx and J. Hutter, Ab Initio Molecular Dynamics, Cambridge Univ Press, 2009.

The evolution of the system given by time dependent Schrödinger equation (TDSE)

$$i\hbar\frac{\partial}{\partial t}\Phi\left(\{r_i\},\{R_I\};t\right)=H\Phi\left(\{r_i\},\{R_I\};t\right)$$

Now our aim is to carry out classical dynamics of nuclei in **QM potential.** Separate nuclear and electronic contributions to the total wave function $\Phi(\{ r_i \},\{ R_I \};t)$

The simplest possible form is a product ansatz

$$\Phi\left(\{r_i\},\{R_I\};t\right)=\Psi\left(\{r_i\},\{R_I\}\right)\chi\left(\{R_I\};t\right)$$

This ansatz was introduced by Born to separate systematically the light electrons from heavy nuclei

Then we have

$$i\hbar\frac{\partial}{\partial t}\Psi(r_i,R_I)\chi(R_I,t)=-[\sum_I\frac{\hbar^2}{2M_I}\nabla_I^2-\sum_i\frac{\hbar^2}{2m_i}\nabla_i^2+V_{n-e}]\Psi(r_i,R_I)\chi(R_I,t)$$

Multiply by $\psi^*(\{r_i\},R_I)$ and integrate over $\{r_i\}$ ⇨ Equation of motion for $\chi(\{R_I\},t)$

Multiply by $\chi^*(\{R_I\},t)$ and integrate over $\{R_I\}$ ⇨ Equation of motion for $\psi(\{r_i\},\{R_I\})$

We can also have following product ansatz which does not invoke solving time-independent electronic Schrödinger equation:

$$\Phi\left(\{r_i\},\{R_I\};t\right)=\Psi\left(\{r_i\},t\right)\chi\left(\{R_I\};t\right)\exp\left[\frac{i}{\hbar}\int_{t_0}^{t}\tilde{E}_e(t')dt'\right]$$

Nuclear and electronic wave functions are separately normalized to unity at every instant of time. $\langle\chi;t\,|\,\chi;t\rangle=1 \quad \langle\psi;t\,|\,\psi;t\rangle=1$

The phase factor was introduced as follows

$$\tilde{E}_e=\int\Psi^*\left(\{r_i\};t\right)\chi^*\left(\{R_I\};t\right)H_e\Psi\left(\{r_i\};t\right)\chi\left(\{R_I\};t\right)drdR$$

Breakdown of the adiabatic Born-Oppenheimer approximation in graphene
Geim et. al. Nat. Mat, 2007 Mar;6(3):198-201
.

Time dependent self-consistent field equations (TDSCF)

$$i\hbar\frac{\partial}{\partial t}\chi=-\sum_I\frac{\hbar^2}{2M_I}\nabla_I^2\chi+\left\{\int dr\Psi^*H_e\Psi\right\}\chi$$

$$H_e=-\sum_i\frac{\hbar^2}{2m_i}\nabla_i^2\quad+V_{n-e}]$$

$$i\hbar\frac{\partial}{\partial t}\Psi(r_i,t)=-\sum_i\frac{\hbar^2}{2m_e}\nabla_i^2\Psi(r_i,t)\quad+\left\{\int dR\,\chi^*(-\sum_I\frac{\hbar^2}{2M_I}+V_{n-e})\chi\right\}\Psi(r_i,t)$$

•Both electrons and nuclei move quantum mechanically in time-dependent effective potentials which are obtained self-consistently.

•The simple product ansatz produces mean field description of the coupled nuclear and electronic dynamics.

Classical dynamics of nuclei and quantum dynamics of electrons

$$\chi = A(R_I,t)\, e^{\frac{i}{\hbar} S(R_I,t)}$$

Where A is the amplitude and S is the phase factor,

Real part govern governs time evolution of phase S and imaginary part time evolution of amplitude A

$$\frac{\partial S}{\partial t} + \sum_I \frac{1}{2M_I}(\nabla_I S)^2 + \int dr \Psi^* H_e \Psi = \hbar^2 \sum_I \frac{1}{2M_I} \frac{\nabla_I^2 A}{A}$$

HW 1

$$\frac{\partial A}{\partial t} + \sum_I \frac{1}{M_I}(\nabla_I A)(\nabla_I S) + \sum_I \frac{1}{2M_I} A\left(\nabla_I^2 S\right) = 0$$

HW 2

Above equation is just the continuity equation if we identify

$$\rho(r) = \chi^* \chi = A^2$$

Probability density

$$J(r) = A^2 \frac{\nabla S}{M_I}$$

Current denstiy

Define velocity field

$$v = \frac{J}{\rho} = \frac{\nabla S}{M_I}$$

Messiah, Vol I , Chapter-6

$$\frac{\partial A}{\partial t}+\sum_I \frac{1}{M_I}(\nabla_I A)(\nabla_I S)+\sum_I \frac{1}{2M_I} A\left(\nabla_I^2 S\right)=0$$

Multiplying the above equation by 2 A and rearranging we get

$$\frac{\partial A^2}{\partial t}+\sum_I \frac{1}{M_I}\nabla_I\left(A^2\nabla_I S\right)=0$$

$$\frac{\partial \rho}{\partial t}+\sum_I \nabla_I J_I=0$$

Note that continuity equation is independent of h

Now if we take the classical limit for the equation of motion for the phase factor $h \rightarrow 0$

$$\frac{\partial S}{\partial t}+\sum_I \frac{1}{2M_I}\left(\nabla_I S\right)^2+\int dr\Psi^* H_e \Psi=0$$

$$\frac{\partial S}{\partial t}+\sum_I \frac{1}{2M_I}\left(\nabla_I S\right)^2+V(r)=0$$

$$\int dr\Psi^* H_e\Psi=V(r)$$

In the classical approximation, χ describes a fluid of non-interacting classical particles of mass M_I and subject to the potential *V(r)*: the density and current density of this fluid at each point of space are at all times respectively equal to the probability density ρ and the probability current density *J* of the quantum particle at that point.

In terms of the velocity field we have

$$\frac{\partial S}{\partial t}+\sum_I \frac{M_I v_I^2}{2}+V(r)=0$$

Take gradient

$$\frac{\partial}{\partial t}\nabla S+\nabla\sum_I \frac{M_I v_I^2}{2}+\nabla V(r)=0$$

$$\left(\frac{\partial}{\partial t} + (v \times \nabla)\right) M_I v + \nabla V = 0$$

$$M_I \frac{dv}{dt} = -\nabla V$$

So we have the equation of motion for the nuclei

$$M_I \ddot{R}_I(t) = -\nabla_I \int dr \Psi^* H_e \Psi$$

Thus the nuclei move according to the classical mechanics in an effective potential which is obtained by solving simultaneously the time dependent electronic Schrodinger equation.

Note that TDSCF equation that describes the time evolution of the electrons still contains the full quantum-mechanical nuclear wave function $\chi(\{R_I\};t)$ instead of just the nuclei position R_I. We can do a classical reduction by the following assumption

$$|\chi(\mathbf{R}_I, t)|^2 = \prod \delta(\mathbf{R} - \mathbf{R}_I(t))$$

$$\int d\mathbf{R}\chi^\star \mathbf{R}_I \chi \rightarrow \mathbf{R}_I(t) \qquad h \longrightarrow 0$$

TDSCF equation for $\psi(\{ r_i \},t)$ then becomes

$$\begin{aligned} i\hbar\frac{\partial\Psi}{\partial t} &= -\sum_i \frac{\hbar^2}{2m_\mathrm{e}}\nabla_i^2\Psi + V_{\mathrm{n-e}}(\{\mathbf{r}_i\}, \{\mathbf{R}_I(t)\})\Psi \\ &= \mathcal{H}_\mathrm{e}(\{\mathbf{r}_i\}, \{\mathbf{R}_I(t)\}) \;\; \Psi(\{\mathbf{r}_i\}, \{\mathbf{R}_I\}; t) \end{aligned}$$

Ehrenfest's theorem

This gives us the equation of motion of the mean values of the coordinates q and the conjugate momenta p of a quantum system

$$i\hbar\frac{d}{dt}\langle q_i\rangle = \langle [q_i, H]\rangle \qquad i\hbar\frac{d}{dt}\langle p_i\rangle = \langle [p_i, H]\rangle$$

We also have $[q_i, H] = i\hbar\frac{\partial H}{\partial q_i}$ $\qquad [p_i, H] = -i\hbar\frac{\partial H}{\partial p_i}$

So we have $\frac{d}{dt}\langle q_i\rangle = \left\langle \frac{\partial H}{\partial p_i}\right\rangle$ $\qquad \frac{d}{dt}\langle p_i\rangle = -\left\langle \frac{\partial H}{\partial q_i}\right\rangle$

In general the mean values do not obey classical laws of motion unless one can replace the mean values of the functions on the right hand side by the function of the mean values

$$\left\langle \frac{\partial H(q_1, ..q_N; p_1,...p_N)}{\partial p_i}\right\rangle = \frac{\partial}{\partial p_i} H\left(<q_1>,..<q_N>;<p_1>,..<p_N>\right)$$

Statistical Ensembles

- Classical phase space is 6N variables ($\mathbf{p}_i$, $\mathbf{q}_i$) and a Hamiltonian function H($\mathbf{q}$,$\mathbf{p}$,t).
- We may know a few constants of motion such as energy, number of particles, volume...
- Ergodic hypothesis: each state consistent with our knowledge is equally “likely”; the *microcanonical* ensemble.
- Implies the average value does not depend on initial conditions.
- A system in contact with a heat bath at temperature T will be distributed according to the canonical ensemble:

$$\exp(-H(q,p)/k_B T)/Z$$

- The momentum integrals can be performed.
- Are systems in nature really ergodic? Not always!

Criteria for an Integrator

- Newton's equation of motion are time reversible and so should be our algorithm.
- Hamiltonian dynamics preserve the magnitude of volume element in phase space and so our algorithm should have this area preserving property
- simplicity (How long does it take to write and debug?)
- efficiency (How fast to advance a given system?)
- stability (what is the long-term energy conservation?)
- reliability (Can it handle a variety of temperatures, densities, potentials?)

The nearly universal choice for an integrator is the Verlet (leapfrog) algorithm

$r(t+\delta t) = r(t) + v(t)\,\delta t + 1/2\, a(t)\,\delta t^{2} + b(t)\,\delta t^{3} + O(\delta t^{4})$ Taylor expand

$r(t-\delta t) = r(t) - v(t)\,\delta t + 1/2\, a(t)\,\delta t^{2} - b(t)\,\delta t^{3} + O(\delta t^{4})$ Reverse time

$r(t+\delta t) = 2\, r(t) - r(t-\delta t) + a(t)\,\delta t^{2} + O(\delta t^{4})$ Add

$v(t) = (r(t+\delta t) - r(t-\delta t))/(2\,\delta t) + O(\delta t^{2})$ estimate velocities

Time reversal invariance is built in □ the energy does not drift.

Velocity is not required to compute the new position.

Once the new position is computed using position at t- δt, discard the old

Position. The current position become the old positions and the new position become the current position.

Note that velocity is used only to compute the kinetic energy and hence the temperature of the system

Time reversible and area preserving

Tuckerman, Berne, Martyna, JCP, **97,** 1990 1992

Review of Hamiltonian Dynamics and Operators in Classical Mechanics

For a classical system, specifying the instantaneous positions and momenta of all the particles constituting the system can specify the microstate at any time t. For N particles there are $3N$ coordinates $q_1, q_2 \ldots q_{3N}$ and $3N$ conjugate momenta $p_1, p_2 \ldots p_{3N}$. The equations of motion are *first order* differential equations

$$\dot{q}_i = \frac{\partial H(q_i, p_i)}{\partial p_i} \qquad \dot{p}_i = -\frac{\partial H(q_i, p_i)}{\partial q_i}$$

Let us consider a simple one-particle system in one dimension with a Hamiltonian

$$H = \frac{p^2}{2m} + U(x)$$

The equations of motion are

$$\dot{q} = \frac{p}{m} \qquad \dot{p} = -\frac{dU}{dx} = F(x)$$

Now Liouville’s theorem says that any phase space function A(x, p) evolves as

$$\frac{dA}{dt}=\{A,H\}$$

where the the {A, H} is the Poisson bracket given by

$$\{A,H\}=\frac{\partial H}{\partial p}\frac{\partial A}{\partial x}-\frac{\partial H}{\partial x}\frac{\partial A}{\partial p}$$

The evolution equation gives back Hamilton’s equation of motion: To see that take A(x,p) =x. Then

$$\frac{dx}{dt}=\dot{x}=\{x,H\}$$

$$\{x,H\}=\frac{p}{m}\frac{\partial x}{\partial x}-\frac{dU}{dx}\frac{\partial x}{\partial p}=\frac{p}{m}$$

So we have $\dot{x}=p/m$ Since $\frac{\partial x}{\partial p}= 0$

Similarly if we take A(x,p) = p

$$\frac{dp}{dt} = \dot{p} = \{p, H\}$$

$$\{p, H\} = \frac{p}{m}\frac{\partial p}{\partial x} - \frac{dU}{dx}\frac{\partial p}{\partial p} = -\frac{dU}{dx} = F(x)$$

So we have $\dot{p} = F(x)$

As expected evolution equation gives back Hamilton's equation of motion

Now define a two-dimensional phase space vector Γ=(x,p). Hamilton's equation of motion for this Γ is

$$\frac{d\Gamma}{dt} = \{\Gamma, H\}$$

Now we define Liouville operator L such that $iL\Gamma = \{\Gamma, H\}$

iL can be expressed as differential operator using Hamilton’s equation

$$
\begin{aligned}
iL &= \frac{\partial H}{\partial p}\frac{\partial}{\partial x} - \frac{\partial H}{\partial x}\frac{\partial}{\partial p} \\
&= \frac{p}{m}\frac{\partial}{\partial x} - \frac{dU}{dx}\frac{\partial}{\partial p} \\
&= \frac{p}{m}\frac{\partial}{\partial x} - F(x)\frac{\partial}{\partial p} \\
&= \dot{x}\frac{\partial}{\partial x} + \dot{p}\frac{\partial}{\partial p}
\end{aligned}
$$

The equation of motion in operator form is given by

$$\frac{d\Gamma}{dt} = iL\Gamma$$

which can be solved to give

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

The operator exp(iLt) is called the classical propagator and the presence of i gives a nice analogy with the QM propagator exp(-iHt/ħ)

Properties of Liouville Operator and propagator

It is Hermitian: $L^{\dagger} = L$ Prove this? (Home work)

The propagator $U(t) \equiv exp(iLt)$ is a unitary operator

$$U^{\dagger}(t)U(t) = I$$

Prove this? (Home work)

The unitarity of the propagator implies time reversal symmetry in the equations of motion. If the system is propagated forward in time up to a time *t* and then the clock is allowed to run backwards for a time –*t*, the system will evolve according to the same equations of motion but the direction of the velocities will be reversed, so that the system will simply return to its initial condition.

$$U(-t) = exp(-iLt)$$

Now apply U(t) on Γ(0) to get Γ(t) followed by U(-t) : $\Gamma(t) = U(t)\,\Gamma(0)$

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

So $U(-t)\,U(t) = I \Rightarrow U(-t) = U^{\dagger}(t)$ since $U(t)$ is unitary

Since $U^{\dagger}(t)$ is equivalent to backward propagation in time, it implies time reversibility since $U^{\dagger}(t)U(t) = I$

Another important property of the unitary operator $U(t)$ is that its determinant is 1 (Homework)

Unitarity of the propagator is consistent with the fact the volume in phase space is preserved under Hamilton's equation (Homework)

Trotter Theorem

We have the evolution of the phase space vector

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

In general it is difficult to evaluate *exp(iLt)* the reason for which will be clear from the following discussion

Remember iL can be written as

$$iL=\frac{p}{m}\frac{\partial}{\partial x}+F(x)\frac{\partial}{\partial p}=iL_1+iL_2$$

$$iL_1=\frac{p}{m}\frac{\partial}{\partial x} \qquad iL_2=F(x)\frac{\partial}{\partial p}$$

The difficulty in the any computation arises from the fact that iL_1 and iL_2 do not commute : $[iL_1,iL_2]\neq 0$

Show that iL_1 and iL_2 do not commute : $[iL_1,iL_2]\neq 0$ (Homework)

Since they don't commute

$$\exp(iL_1t+iL_2t) \neq \exp(iL_1t)\exp(iL_2t)$$

We can see this easily if we expand both side by Taylor expansion

$$e^{iLt} = 1+(iL_1+iL_2)t+\frac{1}{2}(iL_1+iL_2)^2t^2+......$$

$$= 1+iLt+\frac{1}{2}\Big[(iL_1)^2+(iL_2)^2+(iL_1)(iL_2)+(iL_2)(iL_1)\Big]t^2+....$$

Now $e^{iL_1t} = 1+(iL_1)t+\frac{1}{2}(iL_1)^2t^2+......$ $e^{iL_2t} = 1+(iL_2)t+\frac{1}{2}(iL_2)^2t^2+......$

$$e^{iL_1t}e^{iL_2t} = [1+(iL_1)t+\frac{1}{2}(iL_1)^2t^2+...][1+(iL_2)t+\frac{1}{2}(iL_2)^2t^2+...]$$

$$= 1+iL_1t+iL_2t+[(iL_1)(iL_2)+\frac{1}{2}(iL_1)^2+\frac{1}{2}(iL_2)^2]t^2+...$$

$$\neq e^{(iL_1+iL_2)t}$$

Trotter theorem comes to our rescue

$$e^{(iL_1+iL_2)t} = \lim_{M\to\infty}\left[e^{iL_2 t/2M}\, e^{iL_1 t/M}\, e^{iL_2 t/2M}\right]^M$$

For a proof see Techniques and Applications of Path Integral by L. S. Schulman

For large but finite M above equation can be approximated as

$$e^{(iL_1+iL_2)t} = \left[e^{iL_2 t/2M}\, e^{iL_1 t/M}\, e^{iL_2 t/2M}\right]^M$$

$$e^{(iL_1+iL_2)t/M} \approx e^{iL_2 t/2M}\, e^{iL_1 t/M}\, e^{iL_2 t/2M}$$

The expression on the left looks like approximate propagation of the system up to time t by M application of the operator in the bracket. If we interpret t/M as single time step, δt, then we have

$$e^{iL\delta t} = e^{(iL_1 + iL_2)\delta t} \approx e^{iL_2 \delta t/2} e^{iL_1 \delta t} e^{iL_2 \delta t/2}$$

This is the propagator U(δt) for the time step δt. Like U(t), U(δt) is unitary and preserve the time reversibility of the dynamics

Show U†(δt)= U(-δt) = U-1(δt)
and U(-δt) U(δt) = I (Homework)

Now let us see what is the effect of the propagator U(δt) on the coordinates and momenta of the particles

Some useful identity $$e^{c\frac{\partial}{\partial x}} x = x + c \qquad [1 + c\frac{\partial}{\partial x} + \frac{1}{2}c^2\frac{\partial^2}{\partial x^2} + \ldots]x = x + c$$

$$e^{c\frac{\partial}{\partial x}} g(x) = \sum_{k=0}^{\infty} \frac{1}{k!}\left[c\frac{\partial}{\partial x}\right]^k g(x)$$

$$= \sum_{k=0}^{\infty} \frac{1}{k!} c^k g^{(k)} x$$

This is just the Taylor series of g(x+c) so $e^{c\frac{\partial}{\partial x}} g(x) = g(x+c)$

Note that the action of exp(a∂/∂p) on x or g(x) has no effect: it acts like identity operator

$$U(\delta t)x = e^{\frac{\delta t}{2}F(x)\frac{\partial}{\partial p}} e^{\delta t\frac{p}{m}\frac{\partial}{\partial x}} e^{\frac{\delta t}{2}F(x)\frac{\partial}{\partial p}} x$$

$$= e^{\frac{\delta t}{2}F(x)\frac{\partial}{\partial p}} e^{\delta t\frac{p}{m}\frac{\partial}{\partial x}} x$$

First operator acting on x has no effect, it involves only momentum derivative

$$= e^{\frac{\delta t}{2}F(x)\frac{\partial}{\partial p}} (x+\frac{\delta t}{m}p)$$

Second operator changes x to x+δt p/m

$$= x+\frac{\delta t}{m}(p+\frac{\delta t}{2}F(x))$$

Last operator acting on x has no effect. It acts on p and changes p to p+δt F(x)/2

$$x(\delta t) = x+\frac{\delta t}{m}p+\frac{\delta t^2}{2m}F(x)$$

Similarly we can apply U(δt) on p to get

$$U(\delta t)p = e^{\frac{\delta t}{2}F(x)\frac{\partial}{\partial p}} e^{\delta t\frac{p}{m}\frac{\partial}{\partial x}} e^{\frac{\delta t}{2}F(x)\frac{\partial}{\partial p}} p$$

$$= e^{\frac{\delta t}{2}F(x)\frac{\partial}{\partial p}} e^{\delta t\frac{p}{m}\frac{\partial}{\partial x}} (p+\frac{\delta t}{2}F(x))$$

First operator acting on p changes p to p+δt F(x)/2

$$= e^{\frac{\delta t}{2}F(x)\frac{\partial}{\partial p}} (p+\frac{\delta t}{2}F(x+\frac{\delta t}{m}p)$$

Next operator acting on F(x) changes it to F(x+δt p/m)

$$= p+\frac{\delta t}{2}F(x)+\frac{\delta t}{2}F(x+\frac{\delta t}{m}(p+\frac{\delta t}{2}F(x)))$$

Last operator acting on p changes p to p+δt F(x)/2

$$p(\delta t) = p+\frac{\delta t}{2}[F(x)+F(x+\frac{\delta t}{m}p+\frac{\delta t^2}{2m}F(x))]$$

$$= p(0)+\frac{\delta t}{2}[F(x(0))+F(x(\delta t))]$$

Argument of the second force in the above expression is just x(δt)

Velocity Verlet algorithm

$r(\delta t) = r(0) + v(0)\,\delta t + 1/2\; a(r(0))\,\delta t^{2} + b(t)\,\delta t^{3} + O(\delta t^{4})$ good to 2nd order in δt

Do the Taylor expand for velocities

$v(\delta t) = v(0) + \delta t\; a(r(0)) + O(\delta t^{2})$ good to 1st order in δt

To get v also accurate to the 2nd order in δt we consider starting from δt and applying the rule backward in time (i.e. for a time $-\delta t$) so that we end up back at

$r(0) = r(\delta t) - v(\delta t)\,\delta t + 1/2\; a(r(\delta t))\,\delta t^{2}$

$v(\delta t) = (r(\delta t) - r(0))/(\delta t) + \delta t/2\; a(r(\delta t))$

Recall the previous Position Verlet algorithm

$r(t+\delta t) = 2\,r(t) - r(t-\delta t) + a(t)\,\delta t^{2} + O(\delta t^{4})$

$v(t) = (r(t+\delta t) - r(t-\delta t))/(2\,\delta t) + O(\delta t^{2})$

Now using the position equation we have

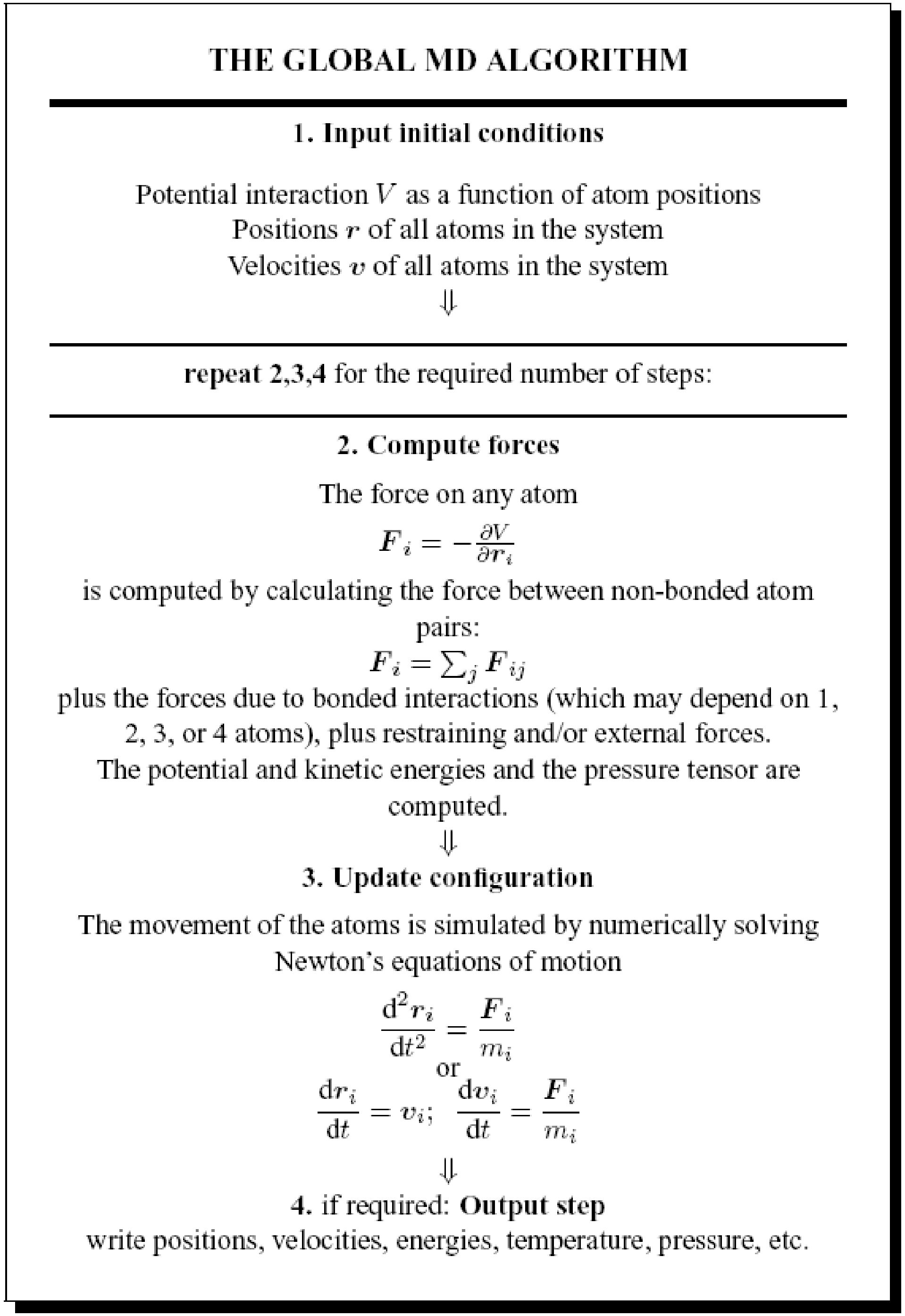
$r(\delta t) - r(0) = v(0)\,\delta t + 1/2\; a(r(0))\,\delta t^{2}$

So we have $v(\delta t) = v(0) + \delta t/2\; [a(r(0)) + a(r(\delta t))]$

Combining we have

$r(\delta t) = r(0) + v(0)\,\delta t + 1/2\; a(r(0))\,\delta t^{2}$

$v(\delta t) = v(0) + \delta t/2\; [a(r(0)) + a(r(\delta t))]$ Velocity Verlet



THE GLOBAL MD ALGORITHM

---

1. Input initial conditions

Potential interaction V as a function of atom positions
Positions $\boldsymbol{r}$ of all atoms in the system
Velocities $\boldsymbol{v}$ of all atoms in the system
$\Downarrow$

---

repeat 2,3,4 for the required number of steps:

---

2. Compute forces

The force on any atom

$$\boldsymbol{F}_i = -\frac{\partial V}{\partial \boldsymbol{r}_i}$$

is computed by calculating the force between non-bonded atom pairs:

$$\boldsymbol{F}_i = \sum_j \boldsymbol{F}_{ij}$$

plus the forces due to bonded interactions (which may depend on 1, 2, 3, or 4 atoms), plus restraining and/or external forces.
The potential and kinetic energies and the pressure tensor are computed.
$\Downarrow$

3. Update configuration

The movement of the atoms is simulated by numerically solving Newton's equations of motion

$$\frac{\mathrm{d}^2\boldsymbol{r}_i}{\mathrm{d}t^2} = \frac{\boldsymbol{F}_i}{m_i}$$

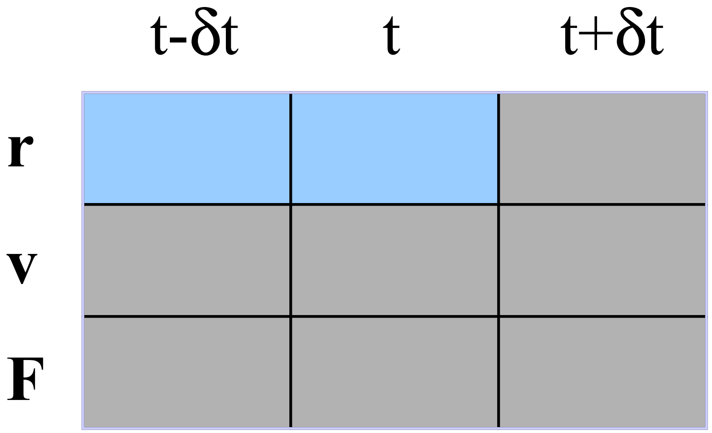
or

$$\frac{\mathrm{d}\boldsymbol{r}_i}{\mathrm{d}t} = \boldsymbol{v}_i; \quad \frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t} = \frac{\boldsymbol{F}_i}{m_i}$$

$\Downarrow$

4. if required: **Output step**
write positions, velocities, energies, temperature, pressure, etc.

Verlet Algorithm: Flow Diagram

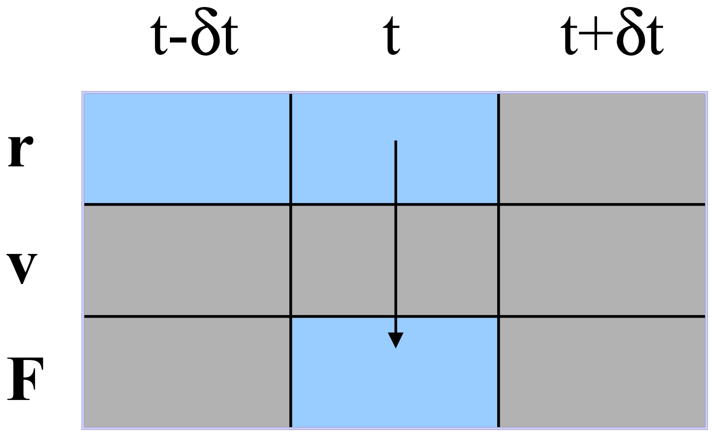


Given current position and position at end of previous time step

Schematic from Allen & Tildesley, Computer Simulation of Liquids

Slide from Kofke Lecture

Verlet Algorithm: Flow Diagram

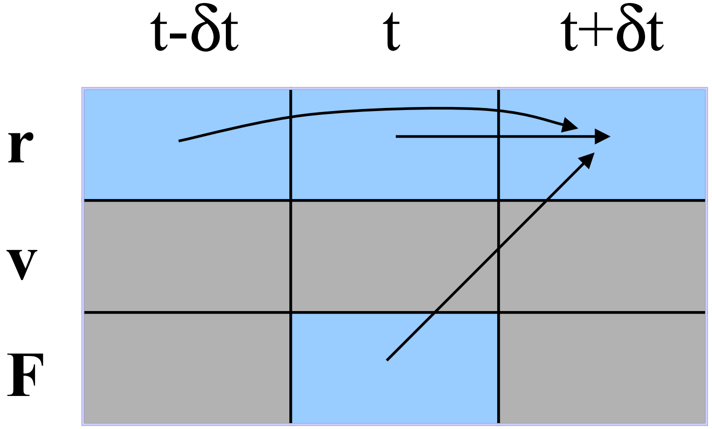


Compute the force at the current position

Schematic from Allen & Tildesley, Computer Simulation of Liquids

Slide from Kofke Lecture

Verlet Algorithm: Flow Diagram

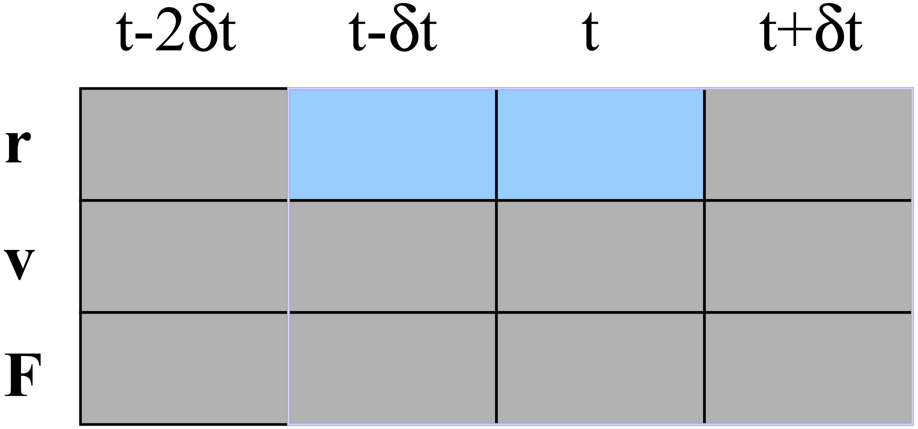


Compute new position
from present and previous
positions, and present force

Schematic from Allen & Tildesley, Computer Simulation of Liquids

Slide from Kofke Lecture

Verlet Algorithm : Flow Diagram



t-2δt t-δt t t+δt

r

v

F

Advance to next time step, repeat

Schematic from Allen & Tildesley, Computer Simulation of Liquids

Slide from Kofke Lecture

Verlet Algorithm: Loose Ends

- Initialization
  - how to get position at "previous time step" when starting out?
  - simple approximation

$$\mathbf{r}(t_0 - \delta t) = \mathbf{r}(t_0) - \mathbf{v}(t_0)\delta t$$

- Obtaining the velocities
  - not evaluated during normal course of algorithm
  - needed to compute some properties, e.g.
    - temperature
    - diffusion constant
  - finite difference

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

Verlet Algorithm Performance Issues

- Time reversible
  - forward time step

$$\mathbf{r}(t+\delta t) = 2\mathbf{r}(t) - \mathbf{r}(t-\delta t) + \tfrac{1}{m}\mathbf{F}(t)\delta t^2$$

  - replace δt with –δt

$$\mathbf{r}(t+(-\delta t)) = 2\mathbf{r}(t) - \mathbf{r}(t-(-\delta t)) + \tfrac{1}{m}\mathbf{F}(t)(-\delta t)^2$$

$$\mathbf{r}(t-\delta t) = 2\mathbf{r}(t) - \mathbf{r}(t+\delta t) + \tfrac{1}{m}\mathbf{F}(t)\delta t^2$$

  - same algorithm, with same positions and forces, moves system backward in time
- Numerical imprecision of adding large/small numbers

$$\overbrace{\boxed{\mathbf{r}(t+\delta t)-\mathbf{r}(t)}}^{O(\delta t^1)} = \overbrace{\boxed{\underbrace{\boxed{\mathbf{r}(t)}}_{O(\delta t^0)} - \underbrace{\boxed{\mathbf{r}(t-\delta t)}}_{O(\delta t^1)}}}^{O(\delta t^1)} + \underbrace{\boxed{\tfrac{1}{m}\mathbf{F}(t)\delta t^2}}_{O(\delta t^2)}$$

Initial Velocity

- Random direction
  - randomize each component independently
  - randomize direction by choosing point on spherical surface
- Magnitude consistent with desired temperature. Choices:
  - Maxwell-Boltzmann: $prob(v_x)\ \mu\ \exp\left(-\frac{1}{2}mv_x^2/kT\right)$
  - Same for y, z components
- Be sure to shift so center-of-mass momentum is zero

$$P_x \equiv \frac{1}{N}\sum p_{i,x}$$

$$p_{i,x} \rightarrow p_{i,x} - P_x$$

Leapfrog Algorithm

- Eliminates addition of small numbers O(δt^2) to differences in large ones O(δt^0)
- Algorithm

$$\mathbf{r}(t+\delta t) = \mathbf{r}(t) + \mathbf{v}(t+\tfrac{1}{2}\delta t)\delta t$$

$$\mathbf{v}(t+\tfrac{1}{2}\delta t) = \mathbf{v}(t-\tfrac{1}{2}\delta t) + \tfrac{1}{m}\mathbf{F}(t)\delta t$$

Leapfrog Algorithm

- Eliminates addition of small numbers $O(\delta t^2)$ to differences in large ones $O(\delta t^0)$
- Algorithm

$$\mathbf{r}(t+\delta t)=\mathbf{r}(t)+\mathbf{v}(t+\tfrac{1}{2}\delta t)\delta t$$

$$\mathbf{v}(t+\tfrac{1}{2}\delta t)=\boxed{\mathbf{v}(t-\tfrac{1}{2}\delta t)+\tfrac{1}{m}\mathbf{F}(t)\delta t}$$

- Mathematically equivalent to Verlet algorithm

$$\mathbf{r}(t+\delta t)=\mathbf{r}(t)+\left[\mathbf{v}(t-\tfrac{1}{2}\delta t)+\tfrac{1}{m}\mathbf{F}(t)\delta t\right]\delta t$$

Leapfrog Algorithm

- Eliminates addition of small numbers $O(\delta t^2)$ to differences in large ones $O(\delta t^0)$
- Algorithm

$$\mathbf{r}(t+\delta t) = \mathbf{r}(t) + \mathbf{v}(t+\tfrac{1}{2}\delta t)\delta t$$

$$\mathbf{v}(t+\tfrac{1}{2}\delta t) = \boxed{\mathbf{v}(t-\tfrac{1}{2}\delta t) + \tfrac{1}{m}\mathbf{F}(t)\delta t}$$

- Mathematically equivalent to Verlet algorithm

$$\mathbf{r}(t+\delta t) = \mathbf{r}(t) + \left[\mathbf{v}(t-\tfrac{1}{2}\delta t) + \tfrac{1}{m}\mathbf{F}(t)\delta t\right]\delta t$$

r(t) as evaluated from previous time step

$$\mathbf{r}(t) = \overbrace{\mathbf{r}(t-\delta t) + \mathbf{v}(t-\tfrac{1}{2}\delta t)\delta t}$$

Leapfrog Algorithm

- Eliminates addition of small numbers $O(\delta t^2)$ to differences in large ones $O(\delta t^0)$
- Algorithm

$$\mathbf{r}(t+\delta t)=\mathbf{r}(t)+\mathbf{v}(t+\tfrac{1}{2}\delta t)\delta t$$

$$\mathbf{v}(t+\tfrac{1}{2}\delta t)=\boxed{\mathbf{v}(t-\tfrac{1}{2}\delta t)+\tfrac{1}{m}\mathbf{F}(t)\delta t}$$

- Mathematically equivalent to Verlet algorithm

$$\mathbf{r}(t+\delta t)=\mathbf{r}(t)+\left[\mathbf{v}(t-\tfrac{1}{2}\delta t)+\tfrac{1}{m}\mathbf{F}(t)\delta t\right]\delta t$$

r(t) as evaluated from previous time step

$$\mathbf{r}(t)=\mathbf{r}(t-\delta t)+\mathbf{v}(t-\tfrac{1}{2}\delta t)\delta t$$

$$\mathbf{r}(t+\delta t)=\mathbf{r}(t)+\left[\left(\mathbf{r}(t)-\mathbf{r}(t-\delta t)\right)+\tfrac{1}{m}\mathbf{F}(t)\delta t^2\right]$$

Leapfrog Algorithm

- Eliminates addition of small numbers $O(\delta t^2)$ to differences in large ones $O(\delta t^0)$
- Algorithm

$$\mathbf{r}(t+\delta t)=\mathbf{r}(t)+\mathbf{v}(t+\tfrac{1}{2}\delta t)\delta t$$

$$\mathbf{v}(t+\tfrac{1}{2}\delta t)=\boxed{\mathbf{v}(t-\tfrac{1}{2}\delta t)+\tfrac{1}{m}\mathbf{F}(t)\delta t}$$

- Mathematically equivalent to Verlet algorithm

$$\mathbf{r}(t+\delta t)=\mathbf{r}(t)+\left[\mathbf{v}(t-\tfrac{1}{2}\delta t)+\tfrac{1}{m}\mathbf{F}(t)\delta t\right]\delta t$$

r(t) as evaluated from previous time step

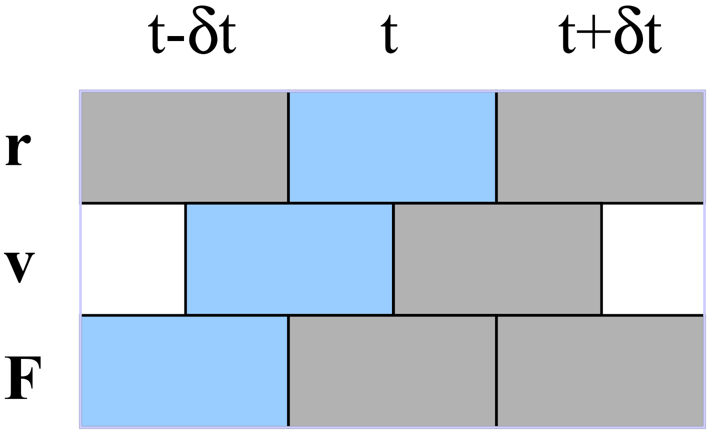
$$\mathbf{r}(t)=\mathbf{r}(t-\delta t)+\mathbf{v}(t-\tfrac{1}{2}\delta t)\delta t$$

$$\mathbf{r}(t+\delta t)=\mathbf{r}(t)+\left[\left(\mathbf{r}(t)-\mathbf{r}(t-\delta t)\right)+\tfrac{1}{m}\mathbf{F}(t)\delta t^2\right]$$

$$\mathbf{r}(t+\delta t)=2\mathbf{r}(t)-\mathbf{r}(t-\delta t)+\tfrac{1}{m}\mathbf{F}(t)\delta t^2$$

original Verlet algorithm

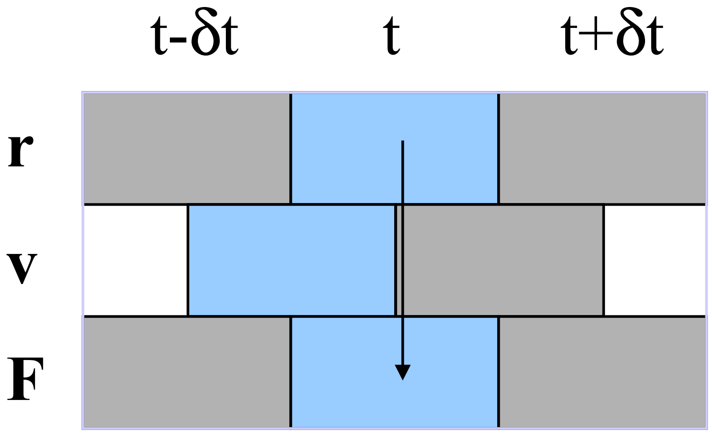
Leapfrog Algorithm: Flow Diagram



Given current position, and velocity at last half-step

Schematic from Allen & Tildesley, Computer Simulation of Liquids

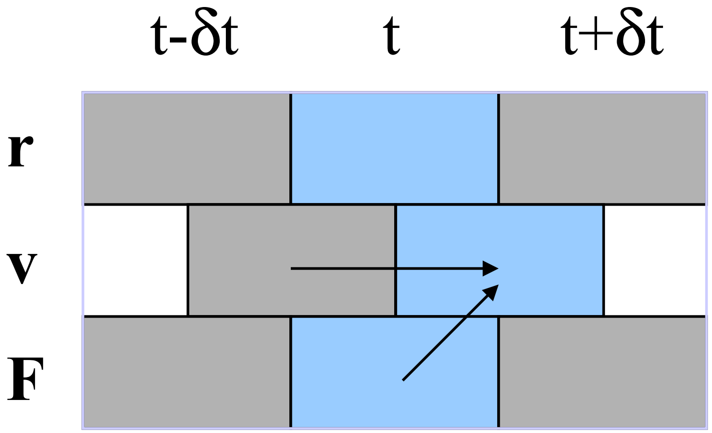
Leapfrog Algorithm: Flow Diagram



Compute current force

Schematic from Allen & Tildesley, Computer Simulation of Liquids

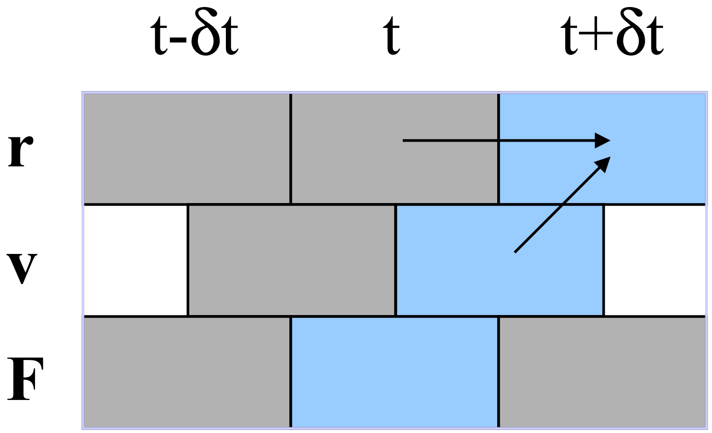
Leapfrog Algorithm: Flow Diagram



Compute velocity at next half-step

Schematic from Allen & Tildesley, Computer Simulation of Liquids

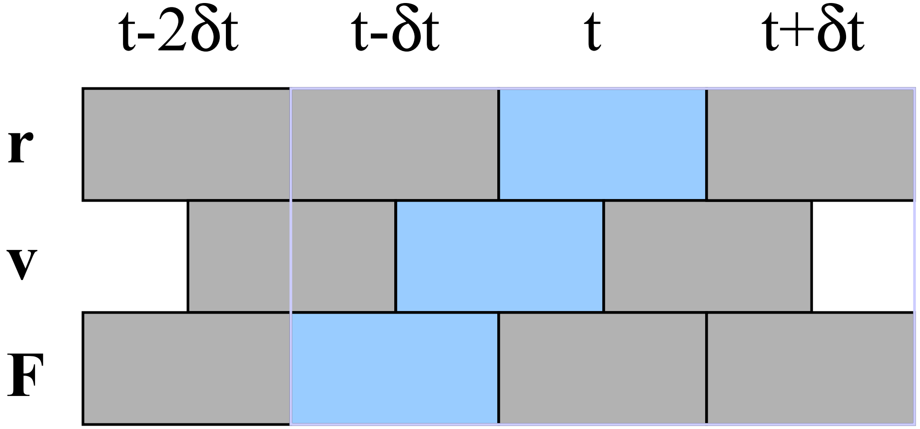
Leapfrog Algorithm: Flow Diagram



Compute next position

Schematic from Allen & Tildesley, Computer Simulation of Liquids

Leapfrog Algorithm: Flow Diagram



Advance to next time step, repeat

Schematic from Allen & Tildesley, Computer Simulation of Liquids

Leapfrog Algorithm Loose Ends

- Initialization
  - how to get velocity at “previous time step” when starting out?
  - simple approximation

$$\mathbf{v}(t_0 - \delta t) = \mathbf{v}(t_0) - \tfrac{1}{m}\mathbf{F}(t_0)\tfrac{1}{2}\delta t$$

- Obtaining the velocities
  - interpolate

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

Velocity Verlet algorithm

$r(\delta t) = r(0) + v(0)\,\delta t + 1/2\; a(r(0))\,\delta t^{2} + b(t)\,\delta t^{3} + O(\delta t^{4})$ good to 2nd order in δt

Do the Taylor expand for velocities

$v(\delta t) = v(0) + \delta t\; a(r(0)) + + O(\delta t^{2})$ good to 1st order in δt

To get V also accurate to the 2nd order in δt we consider starting from δt and applying the rule backward in time (I.e. for a time $-\delta t$) so that we end up back at

$r(0) = r(\delta t) - v(\delta t)\,\delta t + 1/2\; a(r(\delta t))\,\delta t^{2}$

$v(\delta t) = (r(\delta t) - r(0))/(\delta t) + \delta t/2\; a(r(\delta t))$

Now using the position equation we have

$r(\delta t) - r(0) = v(0)\,\delta t + 1/2\; a(r(0))\,\delta t^{2}$

So we have $v(\delta t) = v(0) + \delta t/2\; [a(r(0)) + a(r(\delta t))]$

Combining we have

$r(\delta t) = r(0) + v(0)\,\delta t + 1/2\; a(r(0))\,\delta t^{2}$

$v(\delta t) = v(0) + \delta t/2\; [a(r(0)) + a(r(\delta t))]$ Velocity Verlet

Velocity Verlet Algorithm

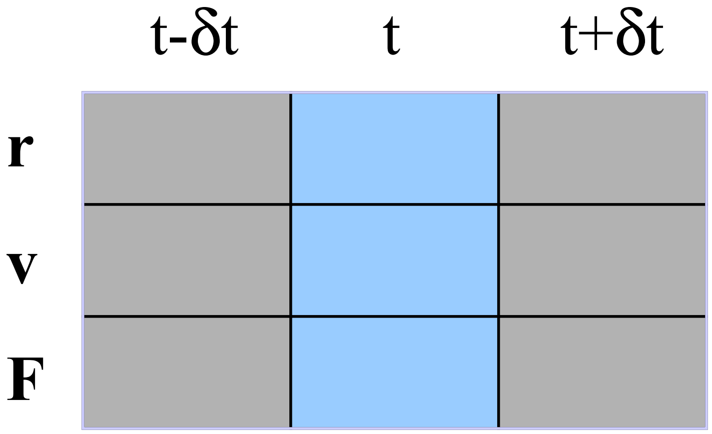
- Round off advantage of leapfrog, but better treatment of velocities
- Algorithm

$$\mathbf{r}(t+\delta t)=\mathbf{r}(t)+\mathbf{v}(t)\delta t+\tfrac{1}{2m}\mathbf{F}(t)\delta t^2$$

$$\mathbf{v}(t+\delta t)=\mathbf{v}(t)+\tfrac{1}{2m}\left[\mathbf{F}(t)+\mathbf{F}(t+\delta t)\right]\delta t$$

- Implemented in stages
  - evaluate current force
  - compute ***r*** at new time
  - add current-force term to velocity (gives ***v*** at half-time step)
  - compute new force
  - add new-force term to velocity
- Also mathematically equivalent to Verlet algorithm (in giving values of **r**)

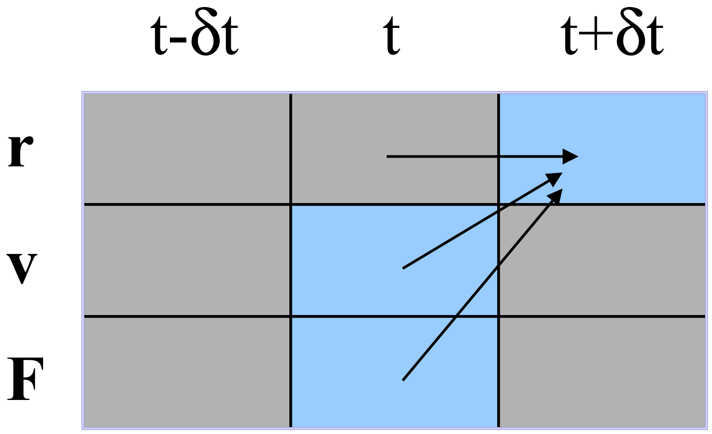
Velocity Verlet Algorithm Flow Diagram



Given current position,
velocity, and force

Schematic from Allen & Tildesley, Computer Simulation of Liquids

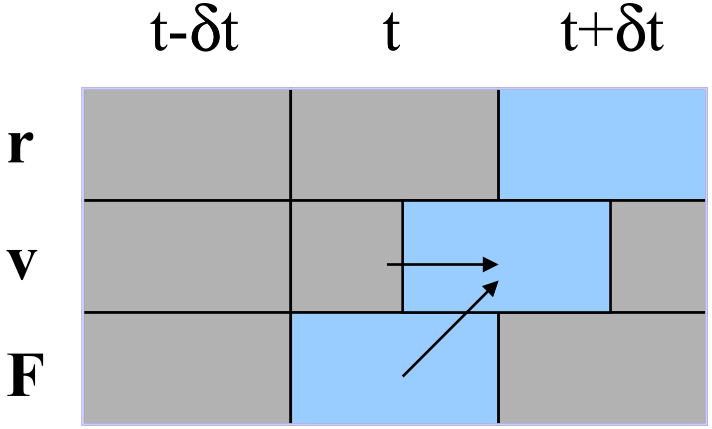
Velocity Verlet Algorithm Flow Diagram



Compute new position

Schematic from Allen & Tildesley, <u>Computer Simulation of Liquids</u>

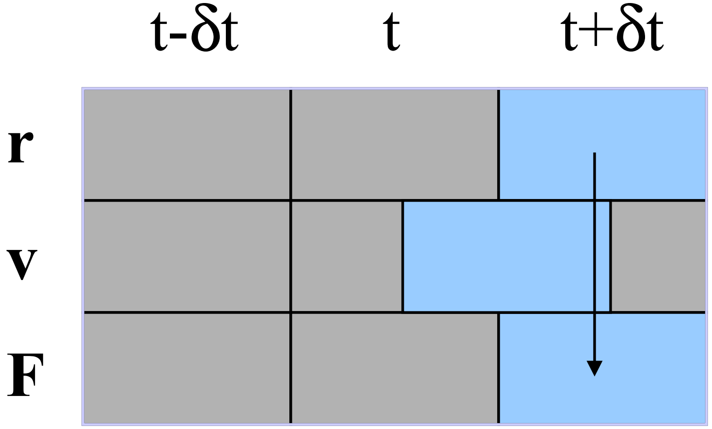
Velocity Verlet Algorithm Flow Diagram



Compute velocity at half step

Schematic from Allen & Tildesley, Computer Simulation of Liquids

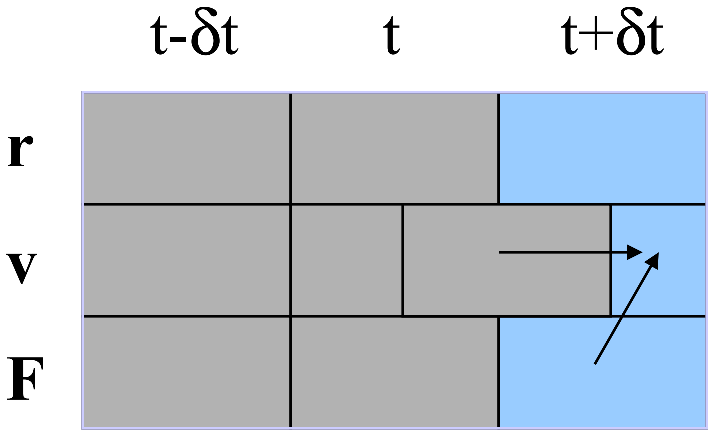
Velocity Verlet Algorithm Flow Diagram



Compute force at new position

Schematic from Allen & Tildesley, *<u>Computer Simulation of Liquids</u>*

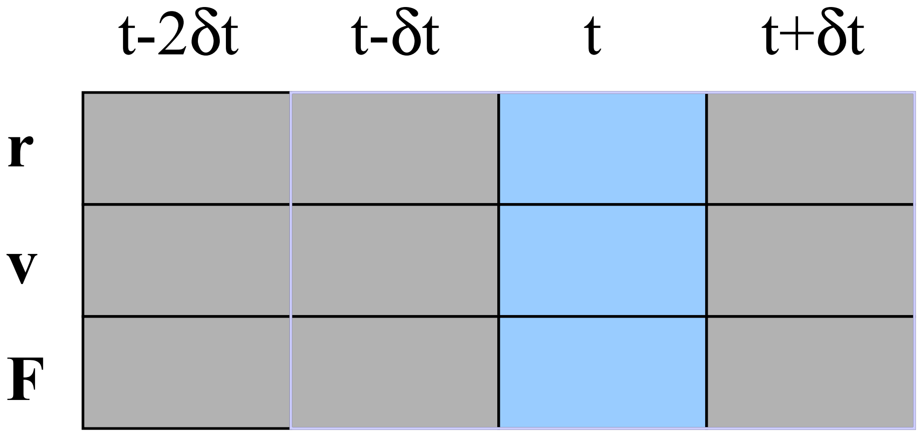
Velocity Verlet Algorithm Flow Diagram



Compute velocity at full step

Schematic from Allen & Tildesley, Computer Simulation of Liquids

Velocity Verlet Algorithm Flow Diagram



Advance to next time step, repeat

Schematic from Allen & Tildesley, Computer Simulation of Liquids

Lines of code for Leap-frog Verlet algorithm

```
/* Carry out half-timestep update of atomic velocities using old forces. */
  velocity_step(n_atoms, atom_vels, atom_mass, delta, comp_forces);


  /* Carry out full-timestep update of atomic positions using half-timestep velocities. */


  position_step(n_atoms, atom_coords, atom_vels, atom_move,
           scaled_atom_coords, h_inv, h, delta, neigh_switch);


 /* Add short-range nonbonded forces to force accumulators. */
    for (i = 0; i < n_atoms; ++i) {
      for ( k = 0; k < NDIM; ++k)
      comp_forces[i][k] += f_vdw_s[i][k] + f_coul_s[i][k];
    }
/* Carry out half-timestep update of atomic velocities using new forces. */
  velocity_step(n_atoms, atom_vels, atom_mass, delta, comp_forces);
```

```
void velocity_step(int n_atoms, double **atom_vels,
                    double *atom_mass, double delta, double **comp_forces)
{
   int i, k;
   double delta_over_2m;

   /* Update velocities. */
   for (i = 0; i < n_atoms; ++i) {
      delta_over_2m = delta /(2*atom_mass[i]) ;
      for ( k =  0; k < NDIM; ++k)
      atom_vels[i][k] += delta_over_2m * comp_forces[i][k];
   }
}
```

```
/* Carry out full-timestep update of atomic positions using half-timestep velocities. */
void position_step(int n_atoms, double **atom_coords,
                   double **atom_vels,
                   double **atom_move, double **scaled_atom_coords,
                   double **h_inv, double **h, double delta, int neigh_switch)
{
  int i, k;
  double dr[NDIM];
 /* Update positions. */
  for (i = 0; i < n_atoms; ++i) {
      for (k = 0; k < NDIM;  ++k) {
     dr[k] = delta * atom_vels[i][k];
     atom_coords[i][k] += dr[k];
} }
 /* If we are using periodic boundary conditions, calculate scaled atomic coordinates. */
     scaled_atomic_coords(n_atoms, h_inv, atom_coords, scaled_atom_coords);
     periodic_boundary_conditions(n_atoms, h, scaled_atom_coords, atom_coords);
}
```

Velocity sampling routine

```
/* Sample atomic velocities from a Maxwellian distribution. */
void sample_velocities(long *idum, double **atom_vels, int n_atoms,
                 double *sqrt_kT_over_m)
{
  int i, k;
  double sqrt_kT_by_m;

  /* Choose Gaussian-distributed cartesian velocity components for each
    atom, with zero mean and standard deviation equal to sqrt(kT/m). */
  for (i = 0; i < n_atoms; ++i) {
    sqrt_kT_by_m = sqrt_kT_over_m[i];
     for (k = 0; k < NDIM; ++k)
    atom_vels[i][k] = sqrt_kT_by_m * gasdev(idum);
  }
}
```

Other Algorithms

Leap Frog: This can be derived from Verlet algorithm. Velocities at half integer time step can be written as

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

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

From the above two equation we get the expression for the new position

$$r(t+\delta t) = r(t) + \delta t\, v(t+\delta t/2)$$

To update the velocity we use the expression from Verlet algorithm

$$v(t+\delta t/2) = v(t-\delta t/2) + \delta t\, a(t)$$

Note that velocities are not defined at the same time as the positions, so KE and PE are also not defined at the same time, and hence we can not directly compute the total energy in the Leap-Frog scheme

Higher Order algorithm: Predictor-corrector

Predictor: use the position and its first *n* derivatives at time *t* (velocity, acceleration etc.) to find the position and its first n derivatives (velocity, acceleration etc.) at time $t+\delta t$

Force evaluation: Use the predicted position to compute the force and acceleration at the predicted positions. The resulting acceleration will be in general different from the "predicted acceleration" in previous step.

Corrector: use the new acceleration to correct the predicted position, velocities and acceleration.

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

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

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

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

The difference between the predicted (step 1) and calculated (step 2) acceleration is given by

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

and is used to correct the positions and velocities in the correction step as follows

$$r^c(t+\delta t) = r(t+\delta t) + c_0 \Delta a(t+\delta t)$$

$$v^c(t+\delta t) = v(t+\delta t) + c_1 \Delta a(t+\delta t)$$

$$a^c(t+\delta t) = a(t+\delta t) + c_2 \Delta a(t+\delta t)$$

$$b^c(t+\delta t) = b(t+\delta t) + c_3 \Delta a(t+\delta t)$$

Coefficients are tabulated for q-th order predictors (Gear): $C_0=1/6$, $C_1 = 5/6$, $C_2=1$, $C_3 = 1/3$

How to set the time step

- Adjust to get energy conservation to 1% of kinetic energy.
- Even if errors are large, you are close to the exact trajectory of a nearby physical system with a different potential.
- Since we don't really know the potential surface that accurately, this is satisfactory.
- Leapfrog algorithm has a problem with round-off error.
- Use the equivalent velocity Verlet instead:

$$\boxed{\begin{aligned}&\mathbf{r}(t+\delta t) = \mathbf{r}(t) + \delta t\,[\,\mathbf{v}(t) + (\delta t/2)\,\mathbf{a}(t)]\\&\mathbf{v}(t+\delta t/2) = \mathbf{v}(t) + (\delta t/2)\,\mathbf{a}(t)\\&\mathbf{v}(t+\delta t) = \mathbf{v}(t+\delta t/2) + (\delta t/2)\,\mathbf{a}(t+\delta t)\end{aligned}}$$

Linear Stability analysis for Harmonic oscillator

$$F(x) = -\omega^2 x$$

Position Verlet scheme can be written as

$$\begin{pmatrix} \omega x(t+\delta t) \\ v(t+\delta t) \end{pmatrix} = S \begin{pmatrix} \omega x(t) \\ v(t) \end{pmatrix}$$

(Home work)

where $S = \begin{bmatrix} 1-\varepsilon^2/2 & \varepsilon(1-\varepsilon^2/4) \\ -\varepsilon & 1-\varepsilon^2/2 \end{bmatrix}$ and $\varepsilon = \omega\delta t$

Powers of S is bounded if $\varepsilon^2 < 4$

$$\delta t < 2/\omega$$
$$< T_p/2\pi$$

$$T_p = 2\pi/\omega$$

How to increase time step?

Limiting factors: intra-molecular motions

Vibrational mode	Wave number $(1/\lambda)$ cm^{-1}	Period T_p (λ/c) fs	$T_p/2\pi$ (fs)
O-H, N-H stretch	3200-3600	9.8	3.1
C-H stretch	3000	11.1	3.5
C≡C, C≡N stretch	2100	15.9	5.1
C=C stretch	1700	19.6	6.2
H-O-H bend	1600	20.8	6.4
O-C-O bend	700	47.6	15

Freeze or constrain the fast motion: make all bond, angle involving H rigid

How to increase time step?

Multiple time steps algorithm

$$F = F_{short} + F_{long}$$

or

$$F = F_{short} + F_{med} + F_{long}$$

Short-range interactions governs the intra-molecular motion, time scale of which are very fast. In that time scale "long range" part of the interaction hardly changes and need not be computed at same frequency of the short range interactions.

Use Multiple time steps

- a time step to compute short-range interactions
- another time stop to compute medium range interactions
- another time step to compute long-range interactions

We can write the Liouville operator L as a sum of three operators that characterize the scales of motions associated with different potential components

$$L = L_{fast} + L_{med} + L_{slow}$$

$$e^{(iL)\delta t} = \exp\left[i\delta t \left(L_{fast} + L_{med} + L_{slow} \right) \right]$$

Tuckerman, Berne, Martyna, JCP, **97,** 1990 (1992)

Physical Quantities in Molecular Simulation

- State variables
  - each variable has an associated “conjugate” variable
    - temperature ⇔ energy (kT,E)
    - pressure ⇔ volume (P,V)
    - chemical potential ⇔ number of molecules (μ,N)
  - specification of state requires fixing one of each pair
  - the dependent variable can be measured by the simulation
- Configuration variables
  - position, orientation, momentum of each atom or molecule
  - energy, forces and torques
  - time
- Properties
  - transport coefficients, free energy, structural quantities, etc.
- Molecular model parameters
  - characteristic energy, size, charge

Separation of the Energy

- Total energy is sum of kinetic and potential parts
  - $E(p^N, r^N) = K(p^N) + U(r^N)$
- Kinetic energy is quadratic in momenta

$$K(p^N) = \sum_i p_i^2 / 2m_i$$

- Kinetic contribution can be treated analytically in partition function

$$Q = \frac{1}{h^{3N} N!} \int dp^N e^{-\beta \sum p_i^2 / 2m} \int dr^N e^{-\beta U(r^N)}$$

$$= \frac{1}{\Lambda^{3N}} \boxed{\frac{1}{N!} \int dr^N e^{-\beta U(r^N)}}$$

$$= \frac{1}{\Lambda^{3N}} Z_N$$

configuration integral

thermal de Broglie wavelength

$$\Lambda = \frac{h}{\sqrt{2\pi mkT}}$$

- And it drops out of position averages

$$\langle A \rangle = \frac{1}{Z_N} \frac{1}{N!} \int dr^N A(r^N) e^{-\beta U(r^N)}$$

Simple Averages 1. Energy

- Average energy

$$\langle E\rangle = \tfrac{1}{Q}\frac{1}{h^{3N}N!}\int dp^N \int dr^N E(p^N,r^N)e^{-\beta E(p^N,r^N)}$$

- Note thermodynamic connection

$$\langle E\rangle = -\frac{\partial \ln Q}{\partial \beta} = \frac{\partial (A/kT)}{\partial (1/kT)} = E_{\text{internal}}$$

- Average kinetic energy

$$\langle K\rangle = \frac{1}{h^{3N}}\int dp^N \sum \frac{p_i^2}{2m}e^{-\beta\sum p_i^2/2m}$$

$$= \tfrac{3}{2}NkT$$

Equipartition of energy: kT/2 for each degree of freedom

- Average potential energy

$$\langle U\rangle = \tfrac{1}{Z_N}\tfrac{1}{N!}\int dr^N U(r^N)e^{-\beta U(r^N)}$$

Simple Averages 2. Temperature

- Need to measure temperature in microcanonical ensemble (NVE) simulations
- Define instantaneous kinetic temperature

$$T = \frac{1}{3Nk} \sum p_i^2 / m$$

More generally, divide by number of molecular degrees of freedom instead of 3N

- Thermodynamic temperature is then given as ensemble average

$$T = \langle T \rangle$$

- Relies on equipartition as developed in canonical ensemble
- A better formulation has been developed recently (Thermostating)

Simple Averages 3a. Pressure

- From thermodynamics and bridge equation

$$P = -\left(\frac{\partial A}{\partial V}\right)_{T,N} = kT\frac{\partial}{\partial V}\ln\left[\tfrac{1}{N!}\int dr^N e^{-\beta U(r^N)}\right]$$

$$P = \frac{NkT}{V} + \frac{1}{3V}\left\langle \sum_{\text{pairs i,j}} \vec{r}_{ij} \times \vec{f}_{ij} \right\rangle$$

Simple Averages 4. Heat Capacity

- Example of a “2nd derivative” property

$$C_v = \left(\frac{\partial E}{\partial T}\right)_{V,N} = -k\beta^2\left(\frac{\partial^2(\beta A)}{\partial\beta^2}\right)_{V,N}$$

$$= -k\beta^2 \frac{\partial}{\partial\beta}\frac{1}{Q(\beta)}\int dr^N dp^N E e^{-\beta E}$$

- Expressible in terms of fluctuations of the energy

$$C_v = k\beta^2\left[\left\langle E^2\right\rangle - \left\langle E\right\rangle^2\right]$$

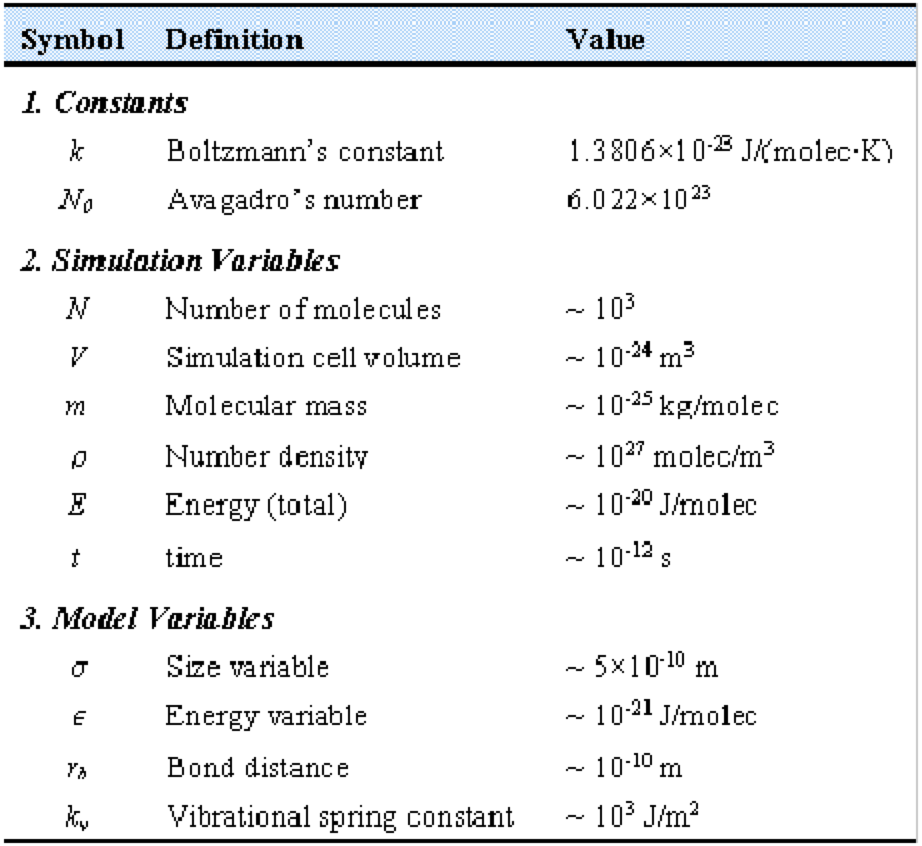
Note: difference between two $O(N^2)$ quantities to give a quantity of $O(N)$

- Other 2nd-derivative or “fluctuation” properties
  - isothermal compressibility

$$\kappa_T = -\frac{1}{V}\left(\frac{\partial V}{\partial P}\right)_{T,N}$$

Dimensions and Units 1. Magnitudes

- Important extensive quantities small in magnitude
  - when expressed in macroscopic units
- Small numbers are inconvenient
- Two ways to magnify them
  - work with atomic-scale units
    - ps, amu, nm or Å
  - make dimensionless with characteristic values
    - model values of size, energy, mass

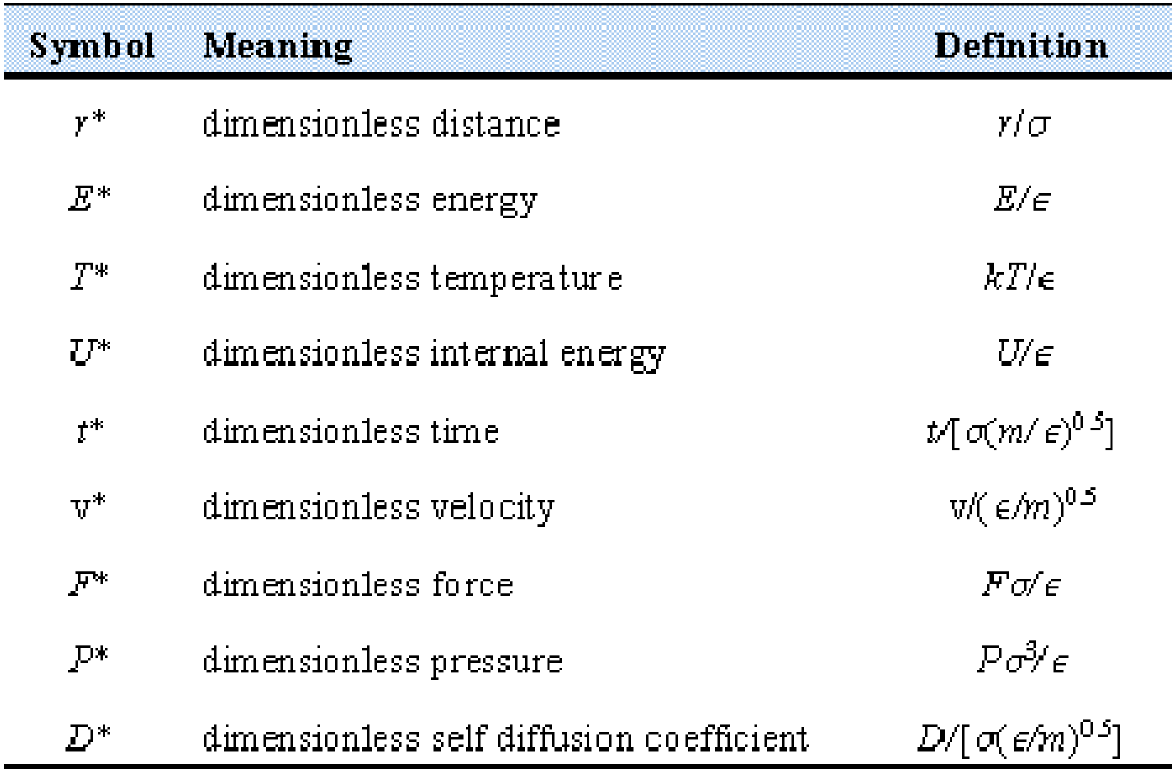


Symbol	Definition	Value
1. Constants		
k	Boltzmann's constant	1.3806×10^{-23} J/(molec·K)
N_0	Avagadro's number	6.022×10^{23}
2. Simulation Variables		
N	Number of molecules	$\sim 10^{3}$
V	Simulation cell volume	$\sim 10^{-24}$ m^3
m	Molecular mass	$\sim 10^{-25}$ kg/molec
ρ	Number density	$\sim 10^{27}$ molec/m^3
E	Energy (total)	$\sim 10^{-20}$ J/molec
t	time	$\sim 10^{-12}$ s
3. Model Variables		
σ	Size variable	$\sim 5\times10^{-10}$ m
ϵ	Energy variable	$\sim 10^{-21}$ J/molec
r_b	Bond distance	$\sim 10^{-10}$ m
k_v	Vibrational spring constant	$\sim 10^{3}$ J/m^2

Dimensions and Units 2. Scaling

- In simulations it is often convenient to express quantities such as temperature, density, pressure and the like in reduced units. This means that we choose a convenient unit of energy, length and mass and then express all the other quantities in terms of these basic units. A natural choice of our basic units is the following
  - size σ
  - energy ε
  - mass m

In terms of these basic units, all other units follow



Symbol	Meaning	Definition
r^*	dimensionless distance	r/σ
E^*	dimensionless energy	E/ϵ
T^*	dimensionless temperature	kT/ϵ
U^*	dimensionless internal energy	U/ϵ
t^*	dimensionless time	$t/[\sigma(m/\epsilon)^{0.5}]$
v^*	dimensionless velocity	$v/(\epsilon/m)^{0.5}$
F^*	dimensionless force	$F\sigma/\epsilon$
P^*	dimensionless pressure	$P\sigma^3/\epsilon$
D^*	dimensionless self diffusion coefficient	$D/[\sigma(\epsilon/m)^{0.5}]$

Why reduced Units?

- Many combinations of ρ, T, ε and σ all correspond to the same state in reduced units. This is the law of corresponding states: the same simulation can of a LJ model can be used to study the Argon at 60 K and density 840 kg/m3 and Xe at 112 K and a density at 1617 kg/m3. In reduced unit both simulations corresponds to the state point ρ = 0.5 and T = 0.5. Scaling by model parameters
- In reduced units almost all quantities of interest are of order 1 (say between 10-3 and 103). Hence if we suddenly find very large (or very small) number in our simulations, suspect some error somewhere.
- Simulation results obtained in reduced units can be translated back into real units.

See the following table

Conversion of reduced Units to real Units for LJ argon system: $\varepsilon/k_B = 119.8$ K, $\sigma=3.405\times10^{-10}$ m, m = 0.03994 kg/mol

Quantity	Reduced Units	Real Units
Temperature	$T^* = 1$	$T = \varepsilon/k_B = 119.8$ K
Density	$\rho^* = 1$	$\rho = 1680$ kg/m^3
Time	$\delta t^* = 0.005$	$\delta t=1.09 \times 10^{-14}$s
Pressure	$P^*=1$	P = 41.9 MPa