University of Stuttgart
Germany

INSTITUTE FOR COMPUTATIONAL PHYSICS

Simulating Soft Matter with ESPResSo, ESPResSo++ and VOTCA

Christian Holm

Institut für Computerphysik, Universität Stuttgart
Stuttgart, Germany

Intro to Soft Matter Simulations

- What is Soft Matter?
- What can simulations do for you?
- What is needed to perform **good** simulations?
- Bits and pieces of necessary background information for understanding molecular simulations
- ESPResSo: history, aim, background

What is Soft Matter?

What is Soft Matter?

….and why is it interesting?

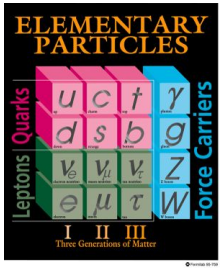
What is Soft Matter?

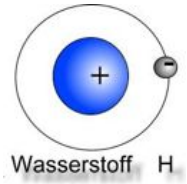
What is Soft Matter?

- Gummy bears, gels, networks: Rubber, low fat food,
- Fibers (z.B. Goretex, Nylon)…
- Colloidal systems: milk, mayonnaise, paints, cosmetics…
- "Simple" plastics: joghurt cups, many car parts, CDs, …
- Membranes: cell walls, artificial tissue, vesicles…
- Many parts of the cell, cytoskeleton, nucleus
- Most biomolecules (RNA, DNA, proteins, amino-acids)
- Liquid crystals
- Many applications: smart materials (actuators, sensors, photonic crystals), biotechnology, biomedicin (hyperthermia, drug targeting, cell separation techniques), model systems for statistical physics

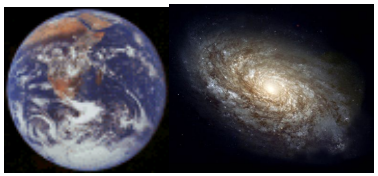
Length Scales of Matter

1 fm 1pm 1 Å 1nm 10 µm 1mm 1m 1km 10^3km 10^6parsec





Soft Matter



10^{-15}m 10^{-12}m 10^{-9}m 10^{-6}m 10^{-3} m 10^{0}m 10^{3}m 10^{6}m

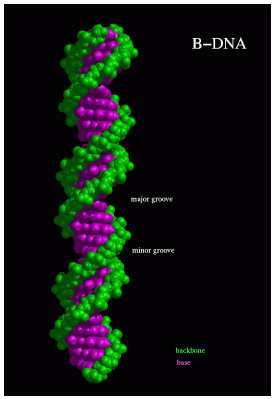
Length Scales of Soft Matter

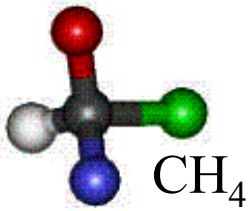
1 Å 1nm 10 µm 1mm

Soft Matter

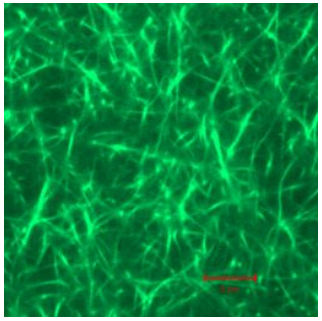
10^{-15}m 10^{-12}m 10^{-9}m 10^{-6}m 10^{-3} m 10^{0}m 10^{3}m 10^{6}m

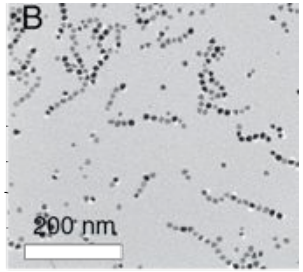
1 nm 10 nm 100 nm 1 µm 10 µm 100 µm

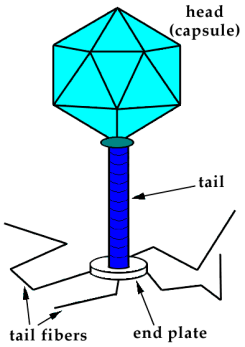


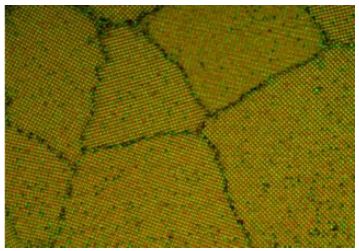


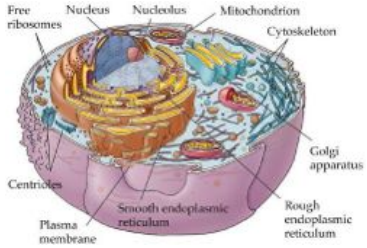
CH_4

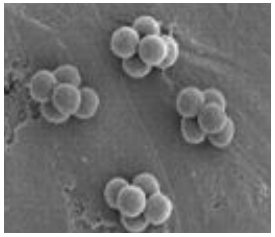


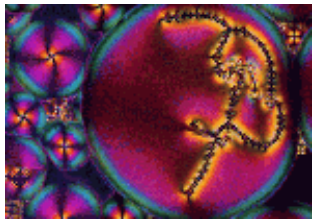


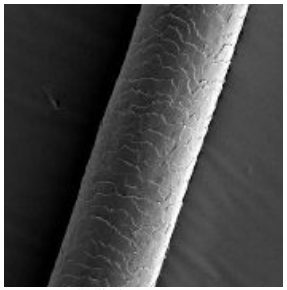






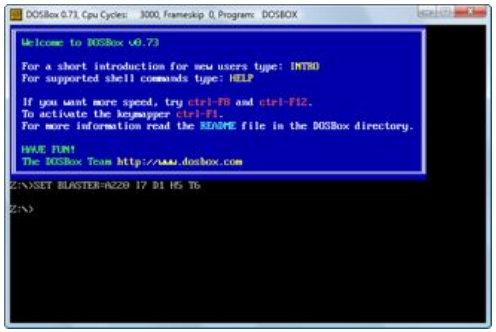


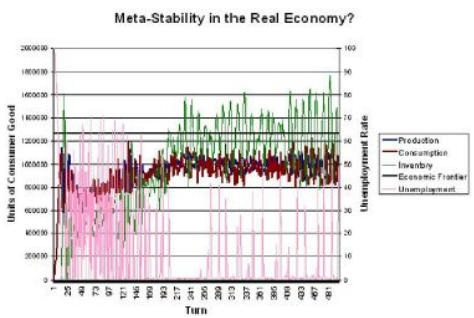


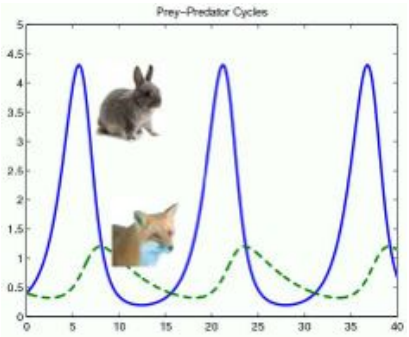


Who needs Simulations?

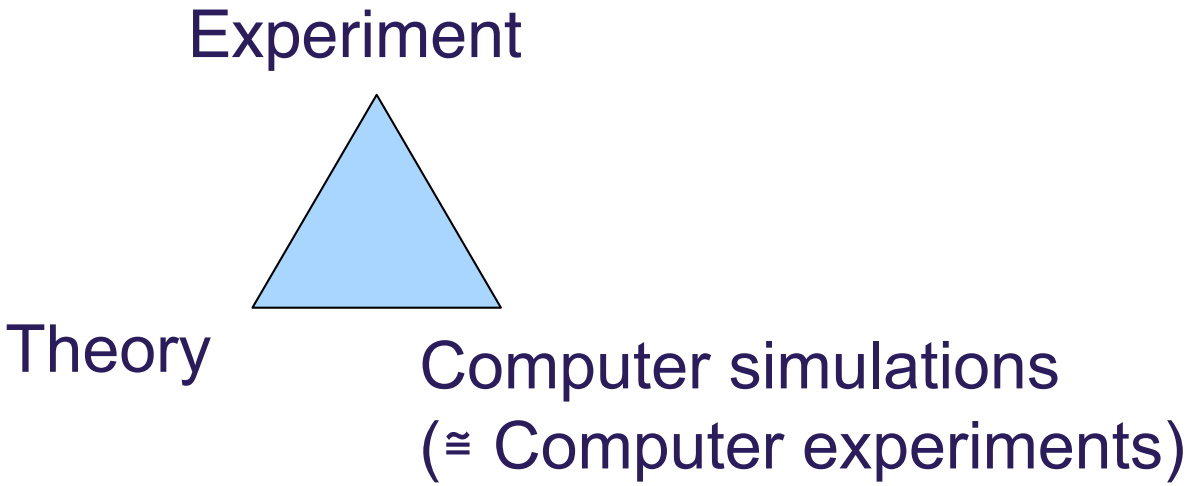
- <u>Goal: Understanding and prediction of interesting systems</u>
- Computer science: Network simulations, “emulations” of not-yet-existing CPUs, …
- Economy: Simulations of economical cycles
- Biology: Simulations of metabolic networks, ecological simulations (e.g. Predator-prey-systems, population dynamics)
- Physics: Simulations of quantum systems, simulations of mechanical systems, astronomical simulations, weather prediction
- Here: Physics/Chemistry/Boplogy: Simulation of Soft Matter and Bio Systems (Polymers, Fluids, Proteins, …)





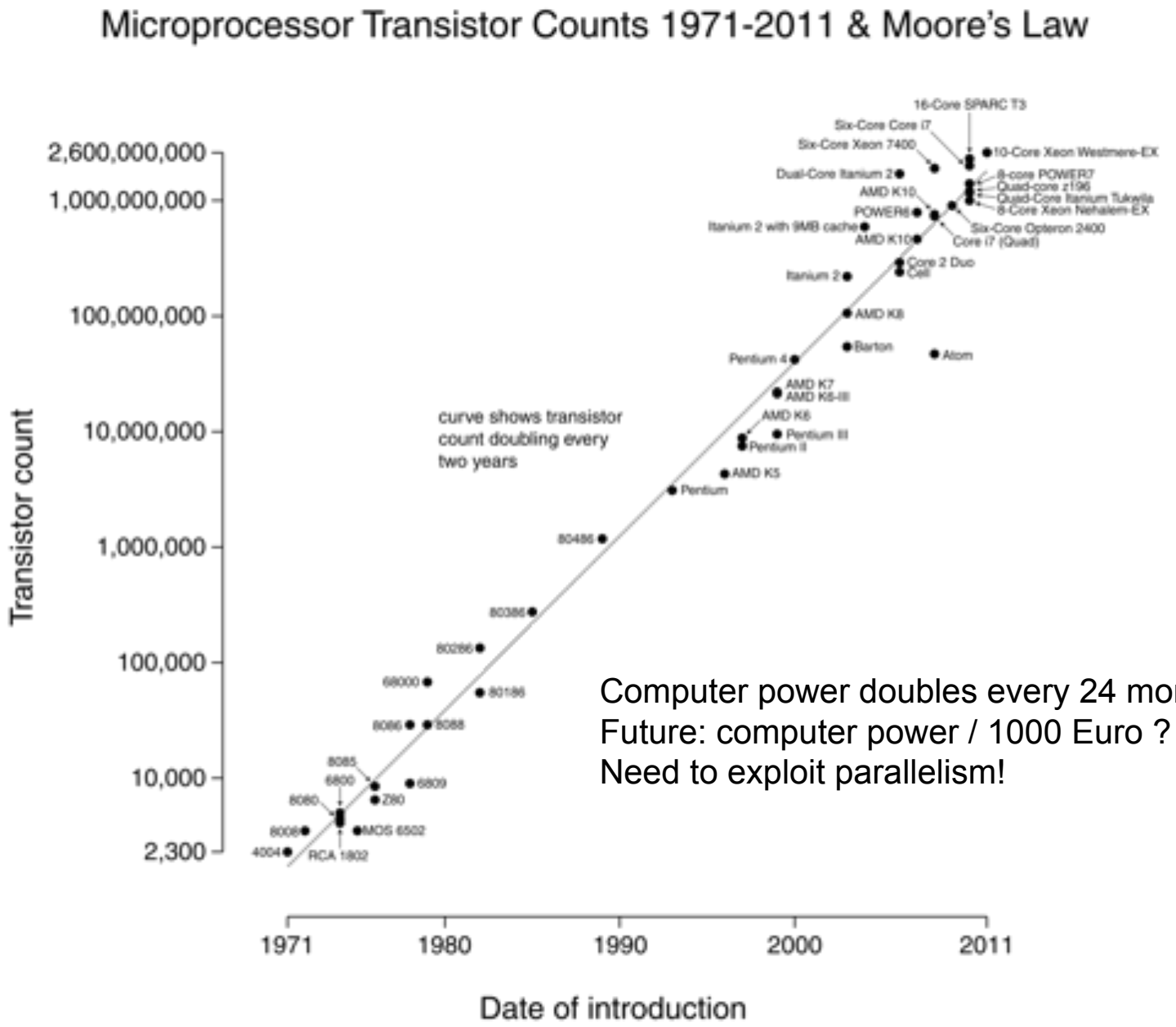


The New Trinity of Physics



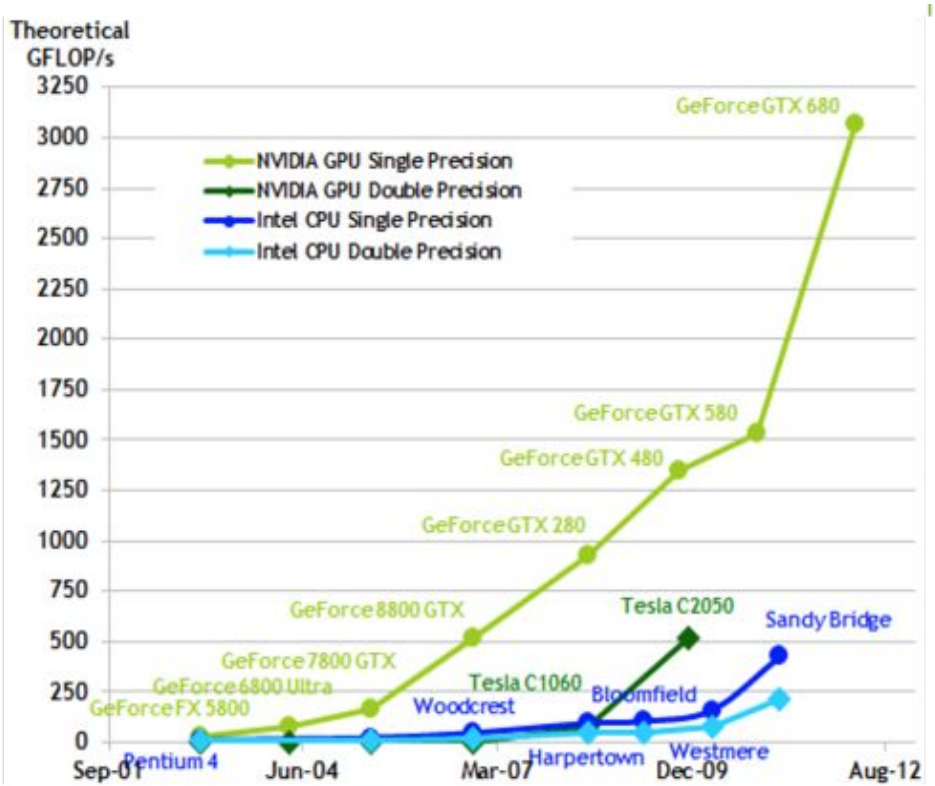
- Why using simulations in physics?
- All laws of nature can be expressed as mathematical formulas
- However, only few physical systems can be solved analytically
- Simulations can be used to numerically solve the most complex formulas and to compare them to experimental results
- System properties can be estimated without actually creating the system (cheaper, simpler, faster and/or less dangerous, well controlled)

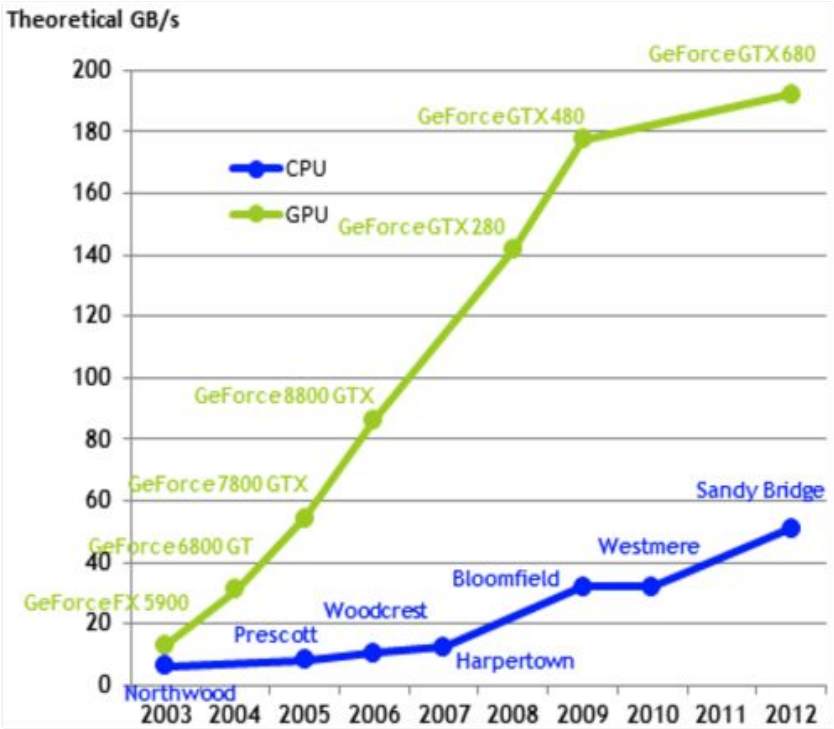
Natural Speed-ups and ....



Computer power doubles every 24 months ends
Future: computer power / 1000 Euro ?
Need to exploit parallelism!

other architectures (GPUs) and...





http://developer.nvidia.com/cuda/nvidia-gpu-computing-documentation

... more clever Algorithms can help!

The more powerful modern computers are, the greater is the advantage of optimal algorithms.

Example Consider following two algorithms (N is the input size):

$$\text{ALG1} \qquad T_{CPU} \sim N^2$$

$$\text{ALG2} \qquad T_{CPU} \sim N$$

In a certain time (neglecting the proportionality factors) ALG1 can run a problem 100 times larger compared to a problem 10,000 times larger by ALG2 or in other words: ALG1 takes 100 times longer!

To illustrate the power of optimal algorithms we will analyze the numerical solution of Poisson's equation on a cube of size $N = n^3$ (table 1.1):

$$\nabla^2 u = f \tag{1.2}$$

Year	Method	Storage	Flops
1947	Gauss elimination	n^5	n^7
1950	Optimal Successive Overrelaxation (SOR)	n^3	$n^3 \lg n$
1971	Conjungated Gradient (CG)	n^3	$n^{3.5} \lg n$
1984	Full Multigrid	n^3	n^3

Table 1.1: Algorithms for solving Poisson's equation on a cube $N = n^3$

This means that for a cube of size $n = 64$ the optimization of the algorithm will give us a reduction of flops in the order of $n^4 \sim 17 \cdot 10^6$!

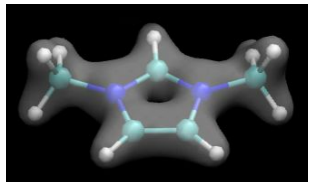
Smart Algorithms can (and do) outperforme Moore‘s law !!

Molecular Simulations

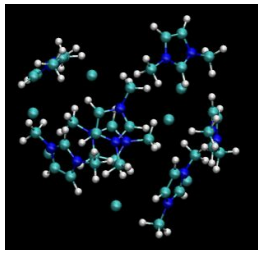
- In a molecular simulation, the evolution of the states of a molecular system needs to be simulated
- In principle, only a pure quantum mechanical description of such a system is *exact* (careful, even here are pitfalls! How many exact solutions are known?)
  - Only very small systems can be simulated on that level
  - ➔ The system has to be simplified ("coarse-grained")
  - ➔ First step: Classical Atoms and Interactions
- Real systems have ~10^{23} atoms
  - ➔ A statistical description is needed
  - ➔ Only a part of a molecular system can be simulated
  - ➔ The simulated system has significant boundaries

Coarse-graining

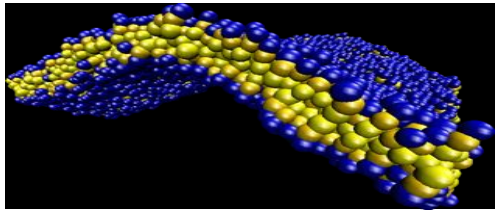
- A model consists of a number of *Degrees of Freedom* (e.g. the atom positions) and the *Interactions* between them
- *Coarse-graining:*
  - reduce the number of degrees of freedom by keeping only the "important" degrees of freedom
  - Use "effective" interactions
- Classical first step: Atoms and Interactions (*all-atom* or *atomistic*)
- Further coarse-graining is often needed and useful
- For Soft Matter we are often on the molecular and mesoscopic level



Quantum

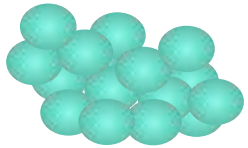


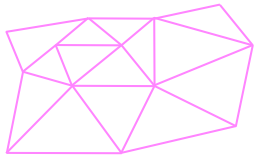
All-atom



Molecular

Mesoscopic
Fluid Methods





Continuum

Computational Approaches

- Quantum: ab-initio QM or first principles high-level QM, PHF, MP2,Car-Parrinello MD, Born-Oppenheimer MD, TBDFT, hybrid embedded QM/MM, ...
- Atomistic: Classical Force Field AA MD,MC
- Coarse-grained: Classical DFT, Molecular Dynamics, Monte Carlo, Field theoretic methods (SCFT)
- Mesoscopic Fluid: Lattice-Boltzmann, MPC, DPD
- Continuum Solvers: Computational Fluid Dynamics codes (Navier-Stokes), Poisson-Boltzmann, Lattice-Boltzmann, FEM

Available Programs:

•First principles Quantum: TURBOMOLE, Molpro (Stuttgart), Gaussian,...

•DFT: CP2K, Car-Parrinello MD, Quantum Espresso, Wien2K,... Look on www.psi-k.org

•All-Atom: GROMOS, GROMACS, NAMD, AMBER, CHARM, DL_POLY, LAMMPS...

•Coarse-grained: DL_POLY, LAMMPS, ESPResSo, OCTA,...

Continuum

•For PB: Delphi, APBS,UHBD

•For FEM: DUNE, more on http://www.cfd-online.com/Wiki/Codes

•Lattice-Boltzmann: openLB

• .....much more than I can list

Making Molecular Simulations

- How to make a molecular simulation?
- Choose the system to be simulated
- Choose the model and coarse-graining level of the simulation
- Determine the initial state of the model
- Simulate the model (using an appropriate algorithm and appropriate tools)
- Analyze and interpret the results

➔ Executing the simulation is only a small part of the work!

Possible errors

- Simulations have plenty of sources for errors!
- Errors of the boundary of the system
- Errors of the initial state
- Errors of the model / level of coarse-graining
- Numerical errors (Errors of the simulation)
- Errors of the interpretation / analysis

Theory and experimental verifications are still needed

Remember Murphy‘s Law!

What do I need to know...

...before I start simulating?

- Statistical Mechanics
- Theory behind my system (i.e. Soft Matter theory)
- The program I am using (best way is to write it yourself!)
- Background of the algorithm (strength, weakness, limitations)
- Clever ways of analyzing the data

Aim of this week long tutorial?

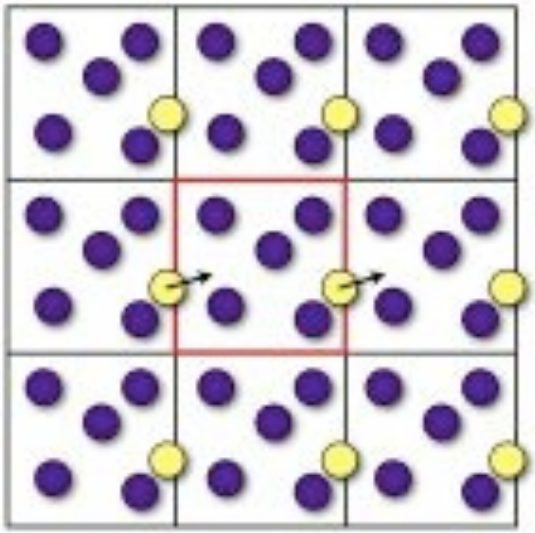
- Describe some Algorithms:
- Long range interactions
- CG Hydrodynamics
- Membrane simulations (Mbtools)
- VOTCA, AdResS, ESPResSo++
- Some sample applications
- ...there is much more you need to know....

→

- Bits and pieces
- Meet developers for specific questions

Periodic Boundary Conditions

- Simulated systems are much smaller than “real” systems
  - ➔ Boundaries make up a significant part of the system!
  - ➔ Surface/Volume not small (i.e. for N=1000 the boundary makes up 49%)
- Trick: *Periodic boundary conditions*
- The simulated system has infinitely many copies of itself in all directions
- A particle at the right boundary interacts with the particle at the left boundary in the image
- *Minimum image convention*: Each particle only interacts with the closest image of another particle (i.e. interaction range L/2)
- Pseudo-infinite system without boundaries
- Significantly reduces boundary-related errors
- More tricky for long range interactions…

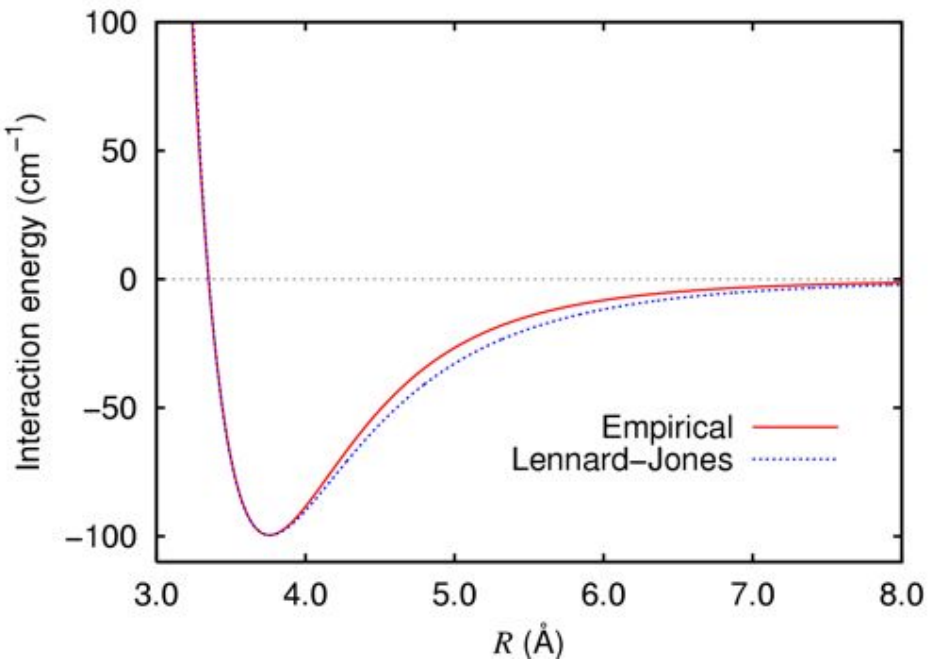


Example: Modeling Liquid Argon

- Very simple system:
  - Noble gas: no bonds between atoms
  - Closed shell: almost spherical
- Contributions to the interaction (from QM):
  - Pauli exclusion principle: strongly repulsive core (exact functional form does not matter)
  - Van-der-Waals interaction: attractive interaction for larger distances ~-1/r^6

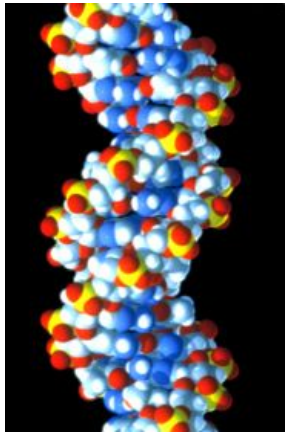
- Semi-empirical *Lennard-Jones-Potential*:
- $$V^{LJ}(r) = 4\varepsilon\left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right)$$

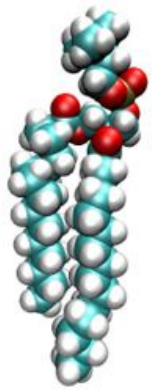
  Liquid argon: σ = 3.4 Å, ε = 100 cm^{-1}



All-Atom Models

- Most commonly used model
- Each atom is represented by one spherical particle
- A *force field (FF)* describes the interactions between the atoms and consists of
  - a set of equations
  - a long table of parameters for all atom type pairs
- For different applications, various different force fields exist (e.g. GROMOS, AMBER, OPLS, Charm…)
- The interactions can be split into two groups:
  - Non-bonded potentials: e.g. Lennard-Jones, Coulomb
  - Bonded potentials for bonded atoms





Non-bonded Potentials

- Non-bonded potentials model the interaction between atoms that do not have bonds
- Lennard-Jones potential accounts for Pauli exclusion and van-der-Waals interaction:

$$V_{LJ}(\boldsymbol{r}_{ij}) = 4\epsilon_{ij}\left(\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right)$$

- Coulomb interaction for charged atoms:

$$V_c(r_{ij}) = f\frac{q_i q_j}{\varepsilon_r r_{ij}}$$

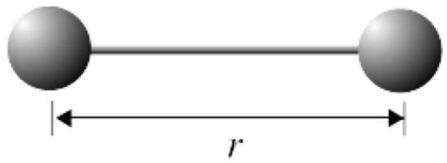
  - Beware: The Coloumb interaction is long-ranged. This may require special measures to compute it!
  - In some force fields, usually uncharged atoms can carry partial charges to account for polarization effects in certain compounds (for example water)

Bonded Potentials

- Bonded potentials model the bonds between atoms
- Bond-stretching: harmonic 2-body potential models bond length:

$$V_b\left(r_{ij}\right) = \frac{1}{2}k_{ij}^{b}(r_{ij} - b_{ij})^2$$

  - Classical spring potential!

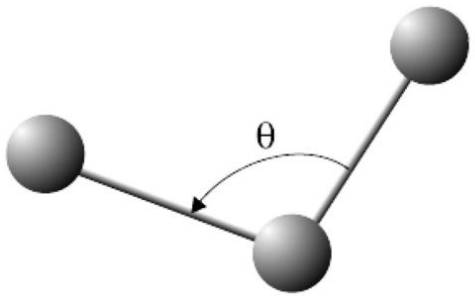


- Bond-angle potential (3-body) models bond angle:

$$V_a(\theta_{ijk}) = \frac{1}{2}k_{ijk}^{\theta}(\theta_{ijk} - \theta_{ijk}^{0})^2$$

or

$$V_a(\theta_{ijk}) = \frac{1}{2}k_{ijk}^{\theta}\left(\cos(\theta_{ijk}) - \cos(\theta_{ijk}^{0})\right)^2$$

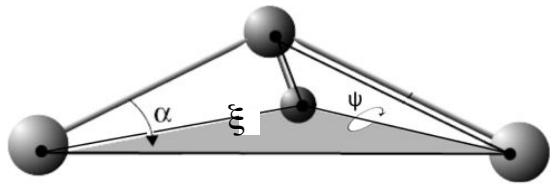


Dihedral Potentials (4-body)

- The dihedral angle is the angle between the planes of 4 bonded atoms
- *Improper dihedrals* keep planar groups planar (e.g. aromatic rings):

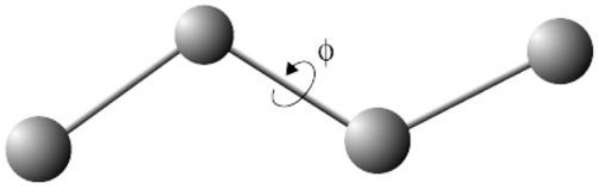
$$V_{id}(\xi_{ijkl}) = k_\xi(\xi_{ijkl} - \xi_0)^2$$

  - again: harmonic potential



- *Proper dihedrals* model cis/trans conformations:

$$V_d(\phi_{ijkl}) = k_\phi(1 + \cos(n\phi - \phi_0))$$



How to find the FF parameters?

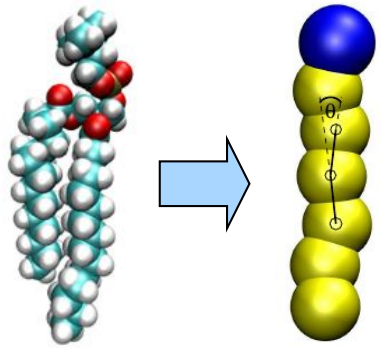
- Fit experimental data (density, g(r), diffusion, heat of vaporization, ...)
- use QM calculations to calculate some interaction parameters
- FF work for the situation where they were parametrized, hope carries us along... (transferability)
- Combining FF parameters is non-trivial, often needs reparametrization

Pros and Cons of FF

- PRO:
- Fast and easy to use (practically linear scaling)
- Visualization of microscopic behavior
- Mechanistic insight
- CON:
- Quality difficult to asses
- Chemical reactions difficult to model
- Orbital interactions (polarizability) often not included

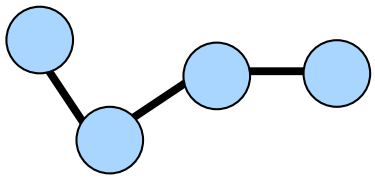
Coarse-grained Models

- Large and complex molecules (e.g. long polymers) can not be simulated on the all-atom level
- Requires coarse-graining of the model
- Coarse-grained models are usually also particles (beads) and interactions (springs, …)
- A bead represents a group of atoms
- Coarse-graining a molecule is highly non-trivial, see systematic coarse-graining, VOTCA, AdResS



Gaussian Polymer in a Θ-solvent

- Conformational properties of a Gaussian polymer in a Θ-solvent are that of a random walk



- ➔ Basis for bead-spring model of a polymer!
- Use a harmonic potential for the bonds:

$$V_{\mathrm{h}}(r) = \frac{k}{2}\left(r - r_0\right)^2$$

- We can compute the partition function exactly

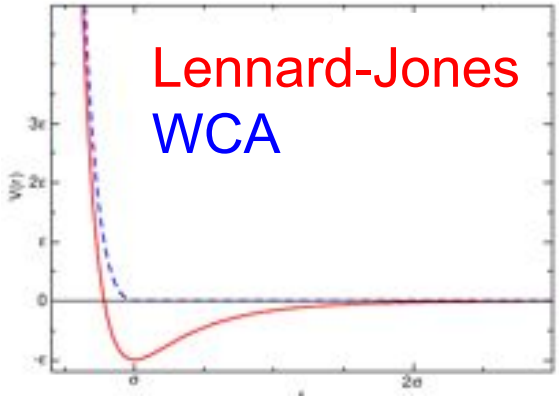
$$H_0 = \frac{1}{2}\underbrace{\frac{3k_BT}{b^2}}_{k}\sum_{i=0}^{N-1}|\vec{r_i} - \vec{r_{i+1}}|^2$$

- Random walk and bead-spring model generate the same partition function!

Gaussian Chains in Good Solvent

- Θ-solvent is a special case!
- Solvents are good or poor w.r. to the polymer
- Good solvent can be modeled via a repulsive potential
  - Use the repulsive part of Lennard-Jones (aka Weeks-Chandler-Anderson)

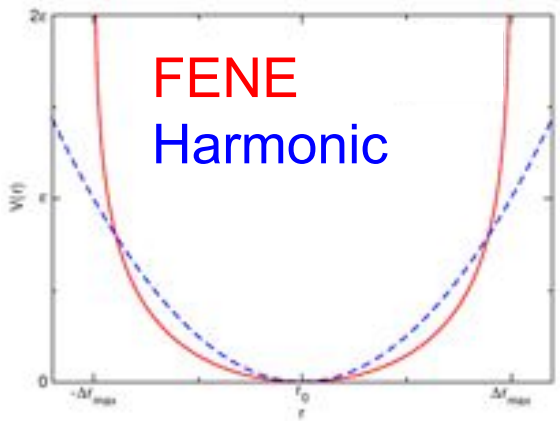
$$V_{\mathrm{WCA}}(r) = \begin{cases} V_{\mathrm{LJ}}(r) + \varepsilon & \text{if } r < 2^{1/6}\sigma \\ 0 & \text{, otherwise} \end{cases}$$



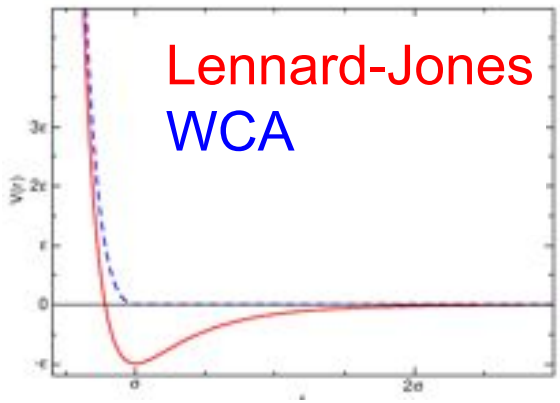
- FENE (Finite Extensible Nonlinear Elastic) bond

$$V_{\mathrm{FENE}}(r) = -\frac{1}{2}\epsilon(\Delta r_{\max})^2 \log\left(1 - \frac{r - r_0}{\Delta r_{\max}}^2\right)$$

  - Has a maximal extension/compression
  - Very similar to harmonic potential at r_0



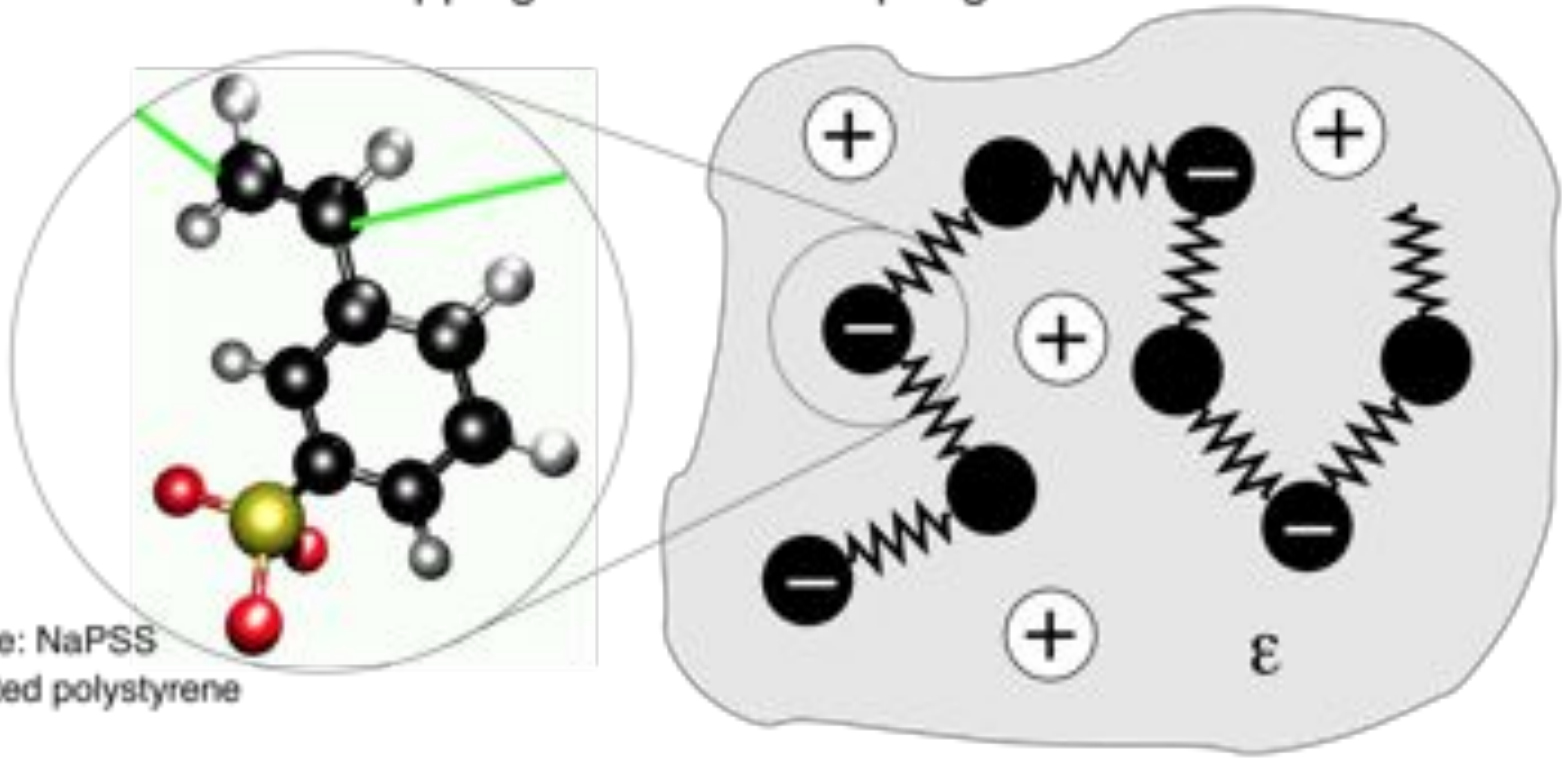
Gaussian Chains in Poor Solvent



- Poor solvent can be modeled via a full Lennar-Jones potential
- Polymer monomers experience an attraction, since they want to minimize contact with solvent
- the quality of the solvent can be changed by
- varying the attraction via the interaction parameter ε and the cut-off
- Scaling laws $R \propto N^{\nu}$ with Flory exponent ν
- RW $\nu = 0.5$
- SAW $\nu = 0.588$ (3/5)
- Globule $\nu = 1/3$
- Rod $\nu = 1$

Charged Polymers

Mapping onto a bead–spring model



Example: NaPSS
sulfonated polystyrene

monomers, ions	→	beads with charge fraction f
bond potential	→	nonlinear springs
solvent	→	dielectric background ε
solvent quality τ	→	effective bead–bead interaction

Molecular Dynamics

Newton's Equations of Motion

- Basic idea of Molecular Dynamics (MD):
- The system consists of point particles and interactions (e.g. atoms and their interactions)
- Solve the classical equations of motion for the particles on the computer:

$$\vec{F}_i = m\vec{a}_i$$

$$\text{or} \qquad \ddot{\vec{x}}_i = \frac{f(\vec{x}_i)}{m}$$

- Can be applied to a wide range of problems:
  - Molecular systems (gases, fluids, polymers, proteins, liquid crystals, …)
  - Granular materials (sand, sugar, salt, …)
  - Planetary motion
  - Nuclear missiles
  - …

Euler's Method of Integration

- Numerical integration: discretize in time, time-step Δt
- Use finite differences:

$$\frac{\partial f(x)}{\partial x} = \frac{f(x+\varepsilon) - f(x)}{\varepsilon}$$

- Taylor expand $x(t+\Delta t)$

$$x(t+\Delta t) = x(t) + v(t)\cdot\Delta t + \frac{f(x(t))}{2m}\cdot\Delta t^2 + \mathcal{O}(\Delta t^3)$$

Euler's Method of Integration

- Numerical integration: discretize in time, time-step Δt
- Use finite differences:

$$\frac{\partial f(x)}{\partial x} = \frac{f(x+\varepsilon) - f(x)}{\varepsilon}$$

- Taylor expand $x(t+\Delta t)$

$$x(t+\Delta t) = x(t) + v(t)\cdot\Delta t + \cancel{\frac{f(x(t))}{2m}}\cdot\Delta t^2 + \mathcal{O}(\Delta t^3)$$

- Truncate at higher order terms!

➔ Positions in the next time-step can be computed!

➔ Simplest integration method, least accurate

Estimating the Time-step

- How large should the time-step Δt be?
- It should not cause numerical instabilities of the integration algorithm
- It should allow to observe the collision of two particles
- ➔ Rule of thumb: Particles should move maximally ~1/10 of the particle diameter *d* per time-step
- ➔ Time-step depends on the maximal velocity v_{max}

$$\Delta t \approx \frac{d}{10} \cdot \frac{1}{v_{max}}$$

Required Iterations

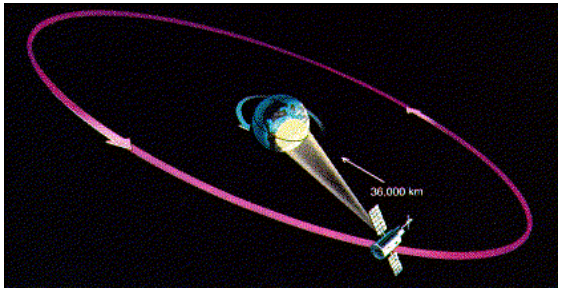
	Atoms / Molecules	Granular matter	Astro-physics
Diameter d	10^{-10} m	10^{-3} m	10^{7} m
Maximal velocity v_{max}	10 m/s	1 m/s	10^{8} m/s
Time-step Δt	10^{-12} s	10^{-4} s	10^{-2} s
Wanted simulation time	1s	10^{2} s	10^{13} s
Required Iterations	10^{12}	10^{6}	10^{15}

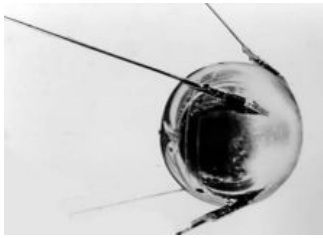
Good Integration Algorithms

- What is a good integration algorithm?
- Easy to implement, fast to compute
- Numerically stable for large time-steps to allow for long simulations
- Trajectory should be reproducible
- Should conserve energy, linear and angular momentum

Short-time Stability

- Depending on the problem at hand, different properties of the integration algorithm are important
- For some systems, it is important that the algorithm has a minimal error in the trajectory ("short-time stable") (e.g. satellite orbits)
- Note that the error in the trajectory always grows exponentially over time due to positive Lyapunov exponents





Long-time Stability

- In molecular simulations, we want to compute statistical averages (i.e. *ensemble averages*) of observables
- MD uses the *Ergodic hypothesis:*
  $\langle A \rangle_{trajectory} = \langle A \rangle_{ensemble}$
- Accurate trajectories are not important
- Instead, the correct physical ensemble should be described throughout the simulation:
  - Conservation of energy, linear and angular momentum
  - Time-reversibility
  - (In fact: conservation of phase space)
- Integrators that do this are “long-time stable” (or “symplectic”)

Verlet Algorithm

- *Verlet Integrator* (1967) is more accurate than Euler's method, and it is long-time stable(!)
- To derive it, Taylor expand *x(t)* forward and backward in time

$$
\begin{aligned}
x(t+\Delta t) &= x(t) + v(t)\Delta t + \frac{f(t)}{2m}\Delta t^2 + \frac{\partial^3 x}{3!\partial t^3}\Delta t^3 + O(\Delta t^4) \\
x(t-\Delta t) &= x(t) - v(t)\Delta t + \frac{f(t)}{2m}\Delta t^2 - \frac{\partial^3 x}{3!\partial t^3}\Delta t^3 + O(\Delta t^4)
\end{aligned}
$$

- This results in

$$
x(t+\Delta t) = 2x(t) - x(t-\Delta t) + \frac{f(t)}{2m}\Delta t^2 + O(\Delta t^4)
$$

- Straightforward algorithm, long-time stable
- Bootstrapping problem: Requires $x(t-\Delta t)$ for the initial step

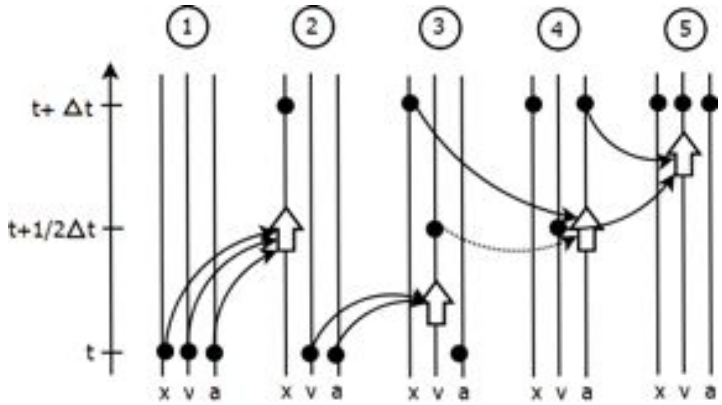
Velocity Verlet Algorithm

$$
\begin{aligned}
x(t+\Delta t) &= x(t) + v(t)\Delta t + \frac{1}{2}a(t)\Delta t^2 \\
v(t+\Delta t) &= v(t) + \frac{a(t)+a(t+\Delta t)}{2}\Delta t
\end{aligned}
$$

- Mathematically equivalent to Verlet algorithm
- Same accuracy $\mathcal{O}(\Delta t^4)$
- No bootstraping problem
- Requires initial velocities instead
- Symplectic- preserves shadow Hamiltonian

➔ Standard algorithm for MD simulations of atomistic and molecular systems

Velocity Verlet in Practice



- Start with x(t),v(t),a(t)
- Calculate new positions: $x(t+\Delta t) = x(t) + v(t)\Delta t + \frac{1}{2}a(t)\Delta t^2$
- Calculate intermediate velocities: $v(t+\frac{1}{2}\Delta t) = v(t) + \frac{1}{2}a(t)\Delta t$
- Compute the new acceleration $a(t+\Delta t)$
- Compute the new velocities:

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

Higher order algorithms

- For problems that require short-time stable behavior for example higher order Runge-Kutta methods can be utilized
- Example (4th order Runge-Kutta):

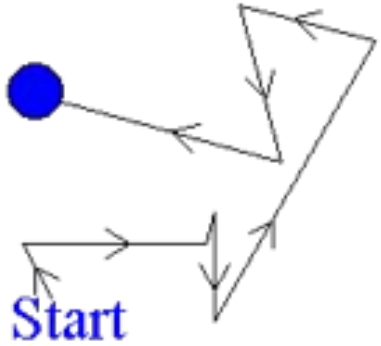
$$
\begin{aligned}
v_1 &= v(t) + \tfrac{1}{2}\Delta t \frac{f(x(t))}{m} \\
x_1 &= \Delta t v(t) \\
v_2 &= v(t) + \Delta t \frac{f(x(t)+\frac{x_1}{2})}{m} \\
x_2 &= \Delta t v_1 \\
v_3 &= v(t) + \Delta t \frac{f(x(t)+\frac{x_2}{2})}{m} \\
x_3 &= \Delta t v_2 \\
x_4 &= \Delta t v_3
\end{aligned}
$$

$$x(t+\Delta t) = x(t) + \frac{x_1}{6} + \frac{x_2}{3} + \frac{x_3}{3} + \frac{x_4}{6} + \mathcal{O}(\Delta t^5)$$

MD in various Ensembles

- Equations of motion are energy conserving
- ➔ NVE (microcanonical) ensemble
- Dynamics can be modified to yield other ensembles:
  - NVT: canonical ensemble
  - NPT: isothermal – isobaric
  - μPT: Gibbs ensemble
- Often achieved via changing the equations of motions (i.e. barostats, thermostats,…)
- Methods that go beyond standard MD are often needed

Langevin Dynamics

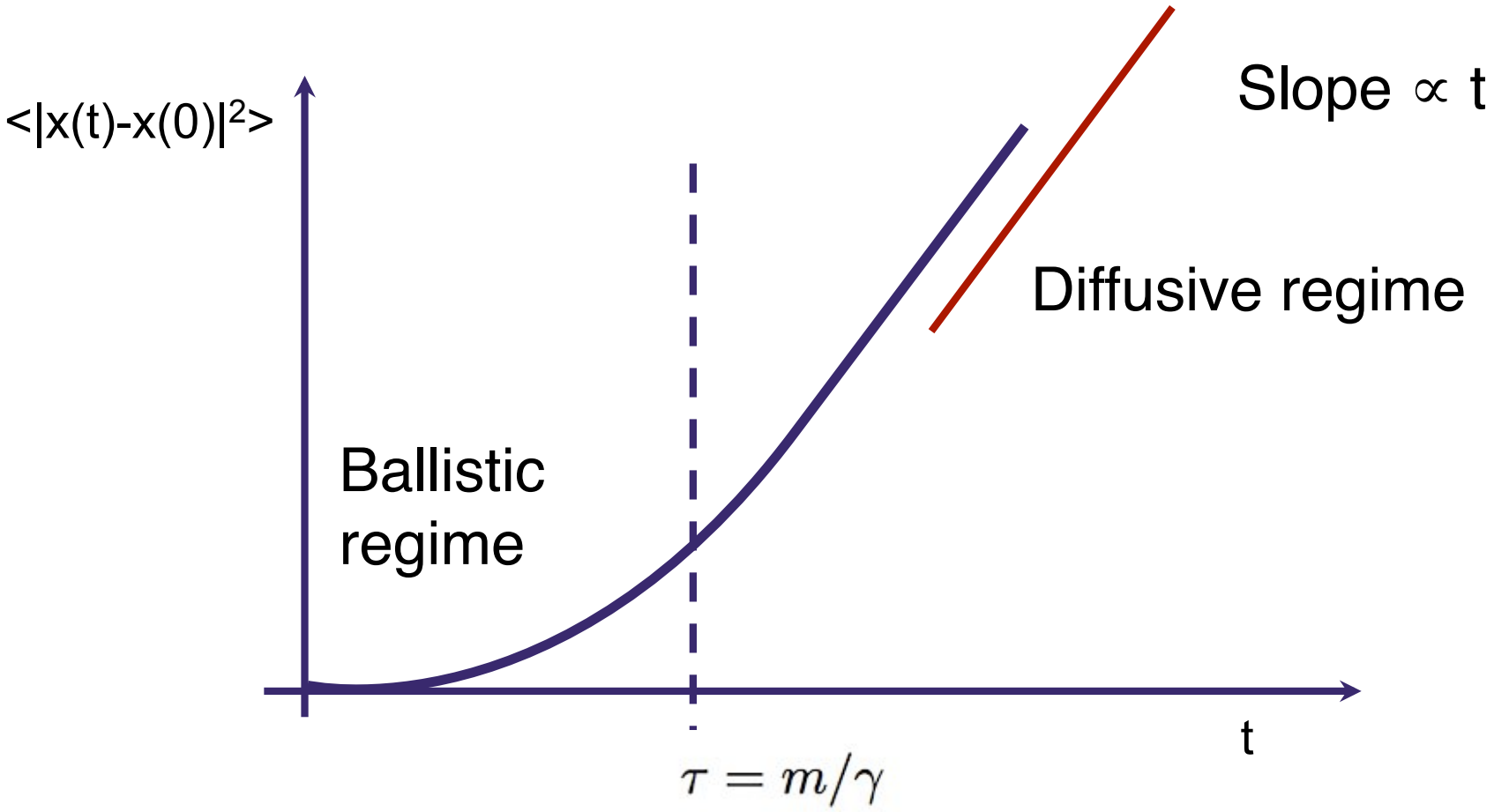


- Simulated molecules are usually not in vacuum. Air or solvent molecules collide constantly with the molecules, leading to Brownian motion
- Simulating all solvent particles would be tedious and time consuming
- *Langevin Dynamics (LD)* models solvent kicks via a random force and a friction:

$$m_i \ddot{\vec{r}}_i = -\nabla V(\{r_i\}) - \underbrace{\Gamma \dot{\vec{r}}_i}_{\text{Drag force}} + \underbrace{\xi_i(t)}_{\text{random force}}$$

- Nice side-effect: LD thermalizes the system (simulates constant temperature, NVT ensemble)

Mean-squared deviation (MSD) in a Langevin Simulation



Advanced MD Techniques

- Parallel tempering
- Metadynamics, Wang-Landau sampling
- Widom insertion
- Flux forward sampling / Transition path sampling / other rare event techniques
- Expanded ensemble techniques
- Umbrella sampling
- Steered MD
- MC/MD hybrids.... and many more...

MD versus Monte-Carlo (MC)

- Properties of Monte-Carlo as compared to Molecular Dynamics:
- Does not (easily) allow to observe dynamics
- Easier to implement
- Harder to parallelize
- No time-step required
- Good random number generator required
- Faster for some systems (special moves!)
- Often need physical insight to select good MC moves

Some historical remarks

ESPResSo at MPI-P in Mainz

- Pre 1: 1998 U. Micka (Fortran Basis, PME)
- Pre 2: 1998 M. Deserno (polymd, P3M)
- Pre 3: 1999 M. Puetz (fast, parallel, no electrostatics, P++)
- 2000 T. Soddemann (extensions on P++)
- 1999 -2001 H.J. Limbach (P++ plus P3M)
- 1998 – 2002 A. Arnold (VMD, more electrostatics routines)

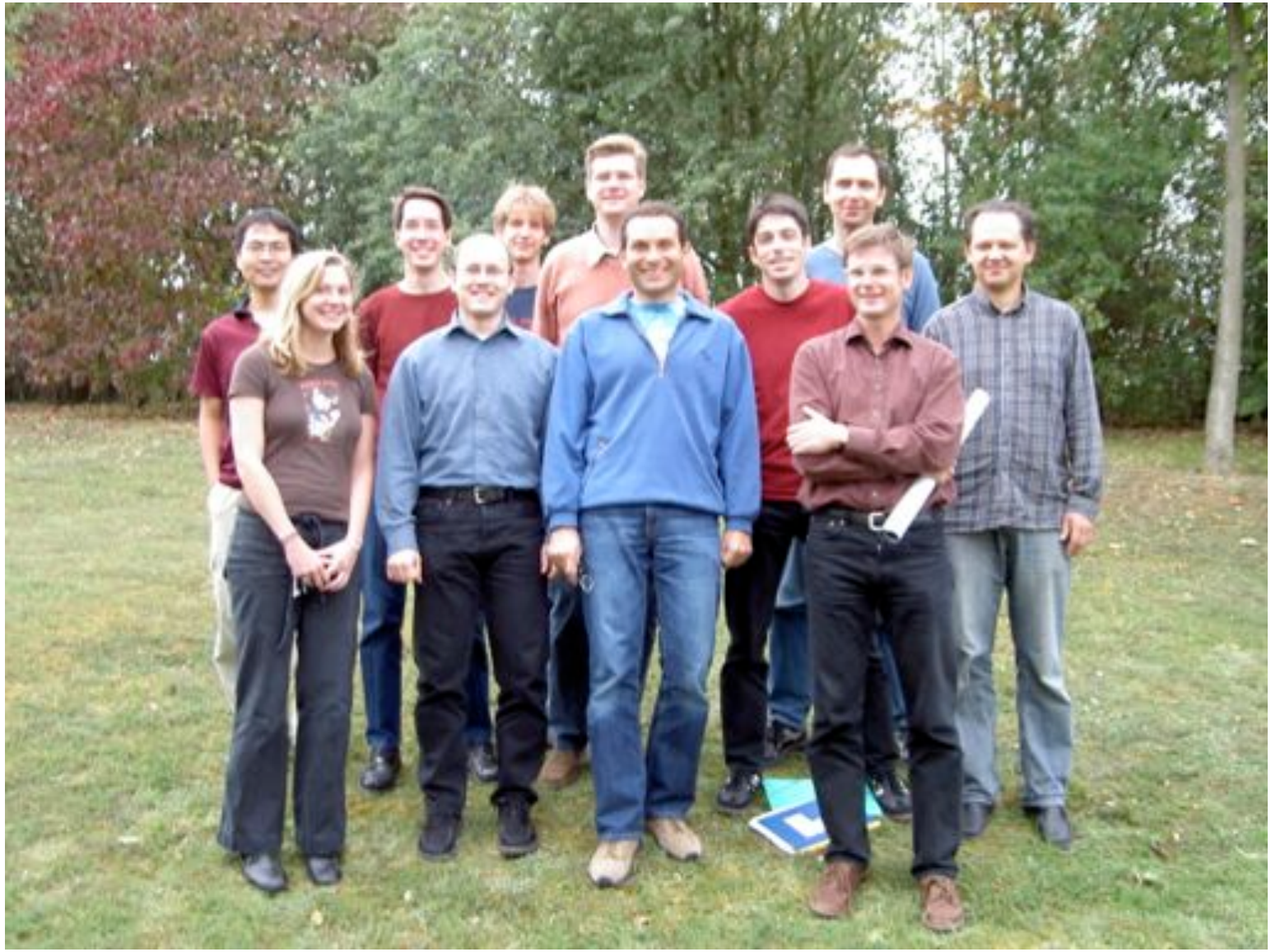
History of ESPResSo

- Start in 2001 Codename „TCL_MD“
  Effort to create an efficiently parallized MD code with P3M (Coulomb interactions), extensible-flexible research tool

- Heinz-Billing Prize 2003 => ESPResSo (release party 25.4.2003)

- Early 2005 => paper ready
  H. J. Limbach and A. Arnold and B. A. Mann and C. Holm *ESPResSo - An Extensible Simulation Package for Research on Soft Matter Systems*, Comp. Phys. Comm. **174** 704-727, 2006.

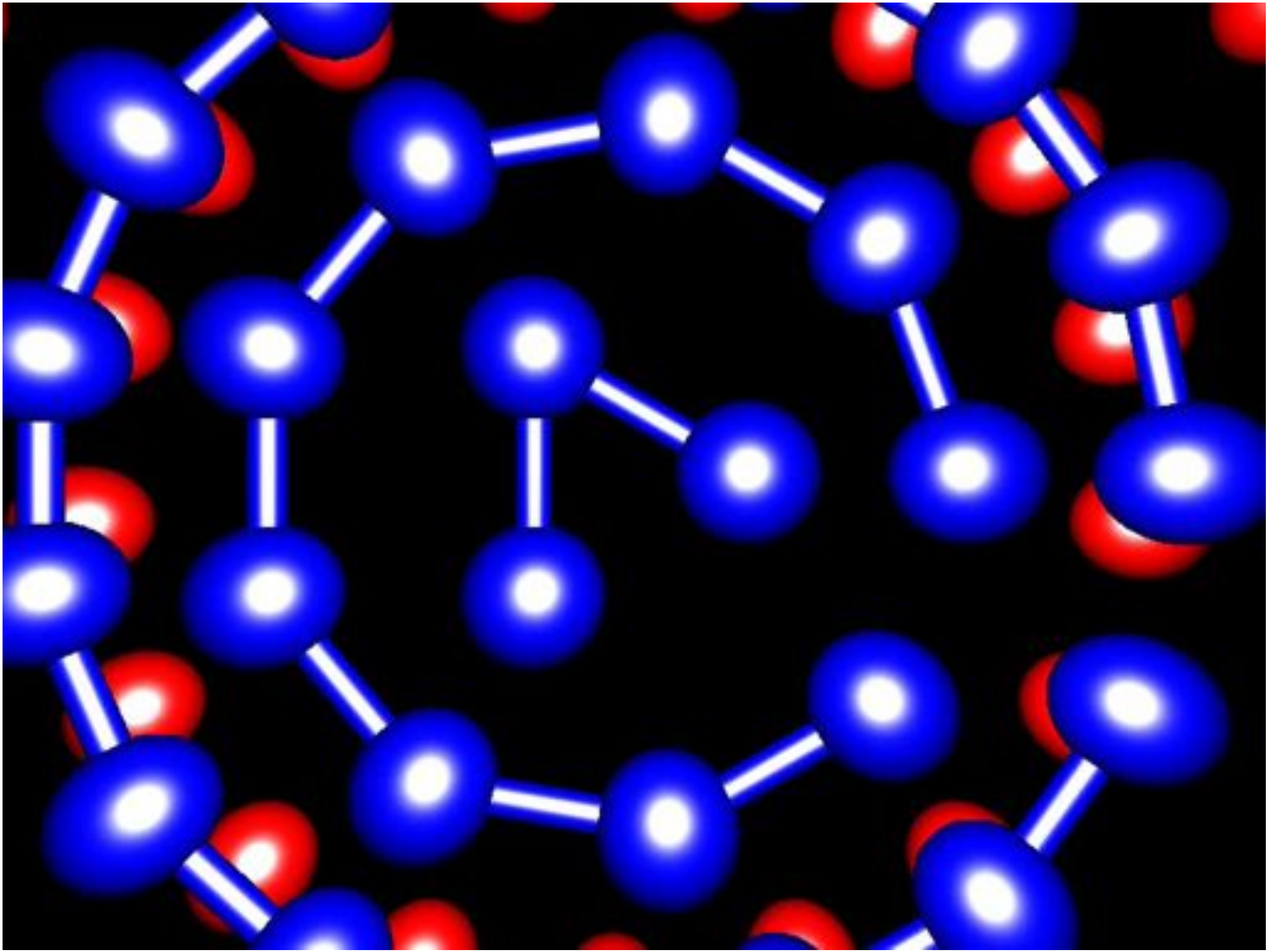
Initiall a pep effort (2004)



Initiall a pep effort (2004)

It soon aquired more people for a lift-off

It soon aquired more people for a lift-off



Ready for Star Wars??

The END

Thank you for your attention