Introduction to Molecular Dynamics simulations

Ali Kerrache

E-mail: ali.kerrache@umanitoba.ca

Who am I?

- **High Performance Computing Analyst**
  - **Grex: support for UofM users**
  - **WestGrid and Compute Canada.**
  - **Software and User Support.**
  - **National teams:**
    - ✓ **BST: Bio-molecular Simulation Team.**
    - ✓ **RSNT: Research Support National Team.**

- **Computational Physicist**
  - **Monte Carlo and Molecular Dynamics codes.**
  - **Study of the properties of materials using MD simulation.**
  - ❖ **Metals, Glasses: Silica, Amorphous silicon, Nuclear Glasses.**
  - ❖ **Mass transport, solid-liquid interfaces, kinetic coefficients, melting, crystallization, mechanical deformations, static and dynamical properties, He diffusion in glasses, …**

Outline

- **Classical Molecular Dynamics simulations:**
  - Introduction
  - Classical MD: basics
  - Algorithms and force fields used in MD
  - Some results:
    - Crystallization of AlNi
    - Shear deformation is a-Si
    - Indentation

- **Setting and running MD simulations: LAMMPS:**
  - Introduction to LAMMPS
  - Building LAMMPS: demonstration
  - Running LAMMPS: demonstration
  - Benchmarks and performance tests

Download the required material

- Use ssh client: PuTTy, MobaXterm, Terminal (Mac or Linux) to connect to cedar and/or graham:

  - ssh –Y username@cedar.computecanada.ca
  - ssh –Y username@graham.computecanada.ca

- Download the files using wget:

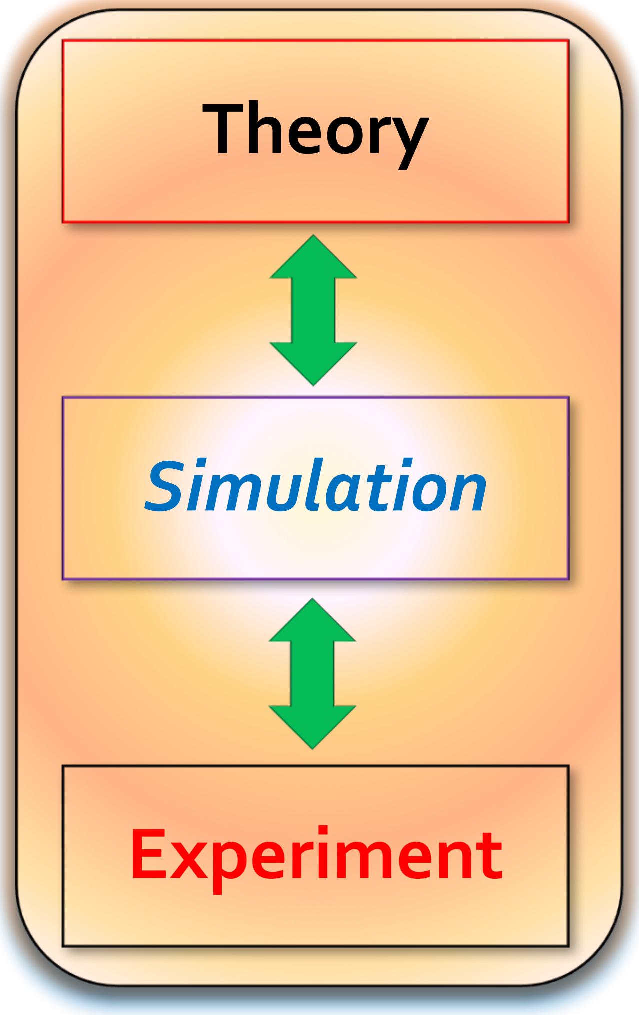
```
wget https://ali-kerrache.000webhostapp.com/uofm/md.tar.gz
wget https://ali-kerrache.000webhostapp.com/uofm/md-slides.pdf
```

- Unpack the archive and change the directory:

```
tar -xvf md.tar.gz
cd UofM-Summer-School-MD
```

Why do we need simulations?

- **Except for simple cases:** no analytical solutions for most of the problems.
- **In most cases, experiments are:**
  - Difficult or impossible to perform.
  - Too dangerous to ...
  - Expensive and time consuming.
  - Blind and too many parameters to control.
- **Simulation is a powerful tool:**
  - can replace some experiments.
  - provoke experiments.
  - explain and understand experiments.
  - complete the theory and experiments.



Atomistic / Molecular simulations

❑ What are atomistic / molecular Simulation?

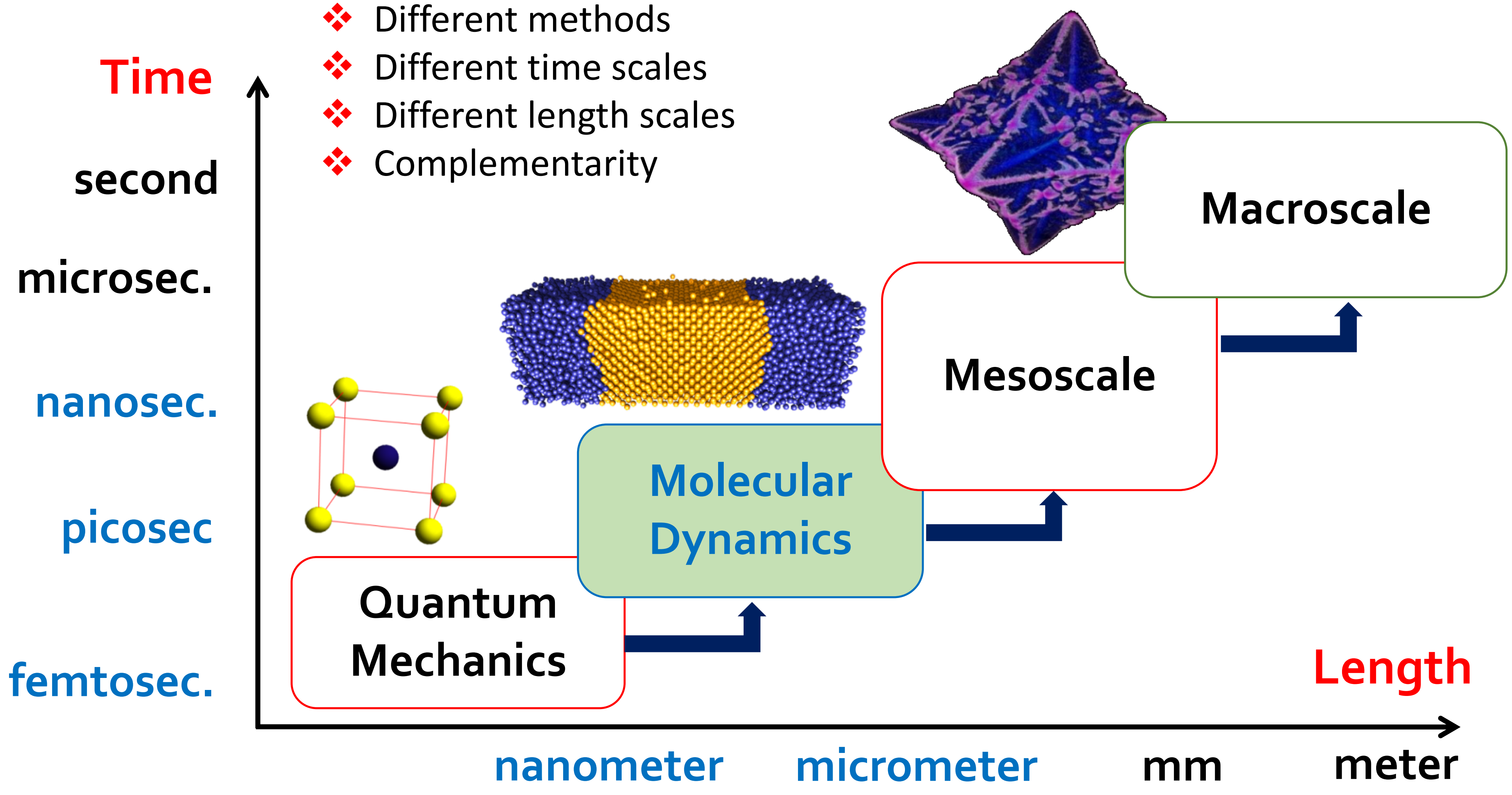
- ➢ a tool to get insights about the properties of materials at atomic or molecular level.
- ➢ used to predict and / or verify experiments.
- ➢ considered as a bridge between theory and experiment.
- ➢ provide a numerical solution when analytical ones are impossible.
- ➢ used to resolve the behavior of nature (the physical world surrounding us) on different time- and length-scales.

❑ Applications, simulations can be applied in, but not limited to:

- ✓ Physics, Applied Physics, Chemistry, …
- ✓ Materials and Engineering, …
- ✓ and more …

Length and time scales

- Different methods
- Different time scales
- Different length scales
- Complementarity



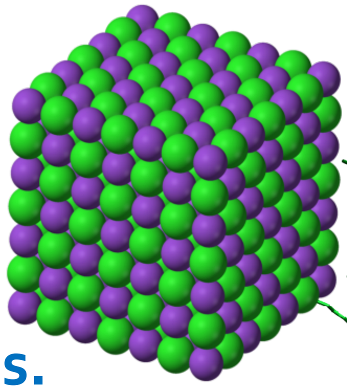
Classical Molecular Dynamics

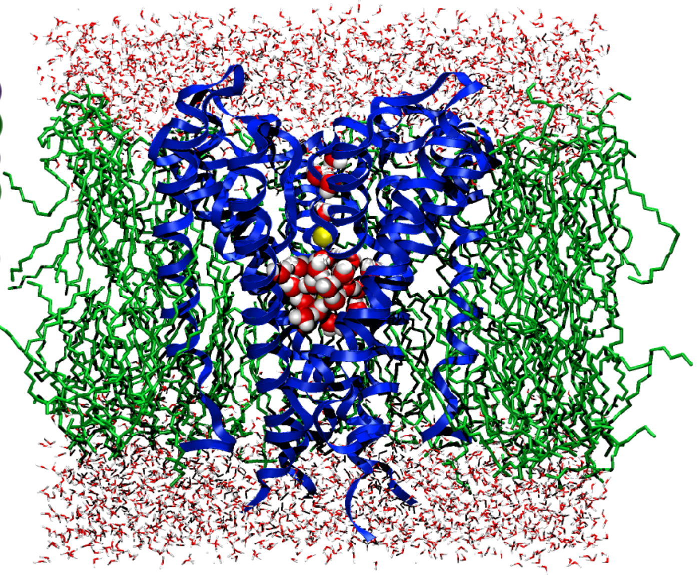
Solution of Newton's equations:

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

$$\vec{F}_i = \sum_{j \neq i}^{N} \vec{f}_{ij}$$

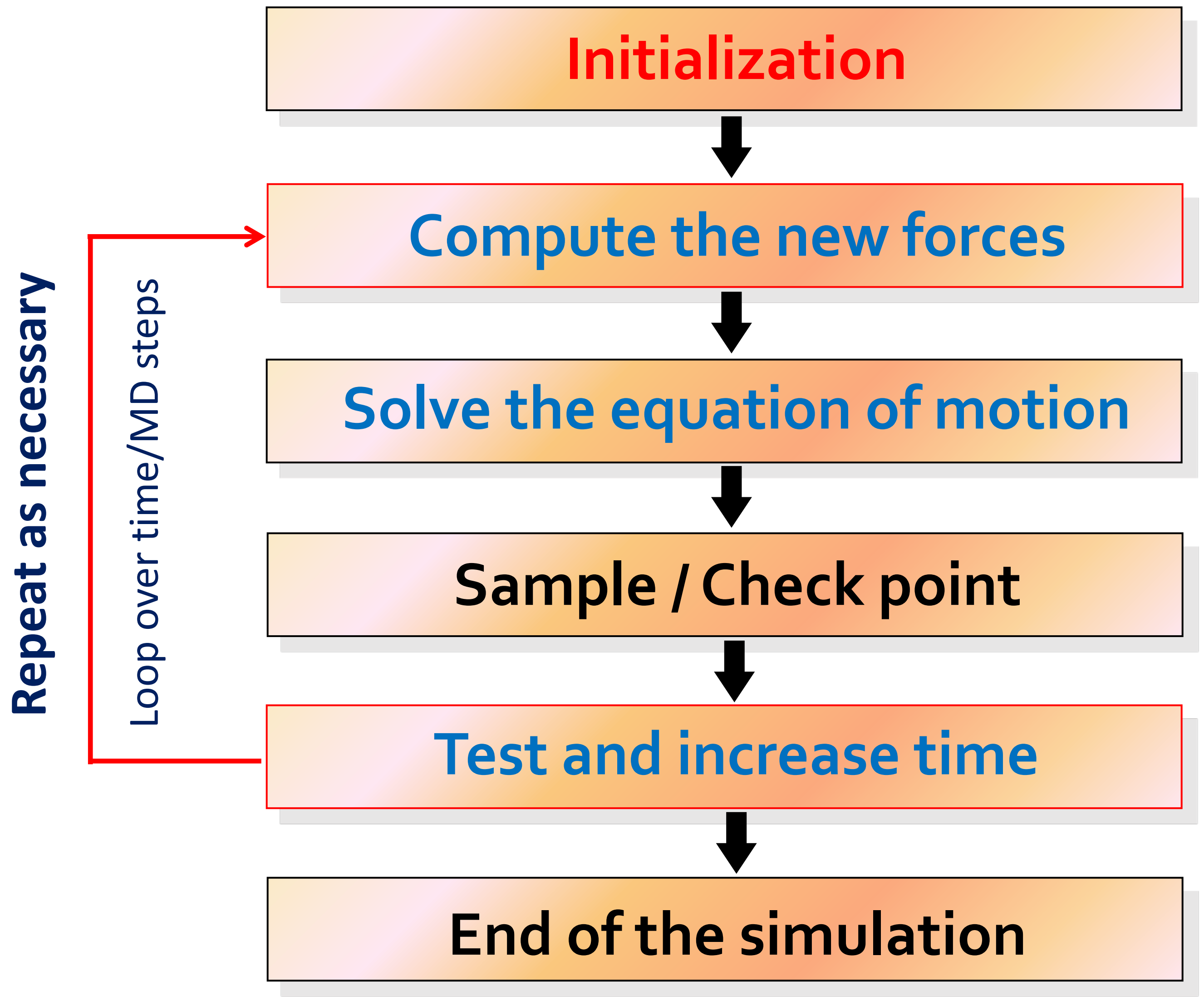
- MD is the solution of the classical equations of motion for a system of N atoms or molecules in order to obtain the time evolution of the system.
- Uses algorithms to integrate the equations of motion.
- Applied to many-particle systems.
- Requires the definition of a force field or potential to compute the forces.
- Potential fitting: first principle calculations and experiments.





$$\vec{f}_{ij} = -\vec{\nabla}_i V(r_{ij})$$

Structure of MD programs



$\mathbf{r}_i(t_0) \quad \mathbf{v}_i(t_0)$

$\mathbf{F}_i(\mathbf{r}_i)$

$\mathbf{r}_i(t) \rightarrow \mathbf{r}_i(t+\Delta t)$

$\mathbf{v}_i(t) \rightarrow \mathbf{v}_i(t+\Delta t)$

- Thermodynamic properties, …
- MD trajectory, …

$t = t + \Delta t$

$t > t_{\max}$

Forces in MD: Newton's equations

- Potential function:

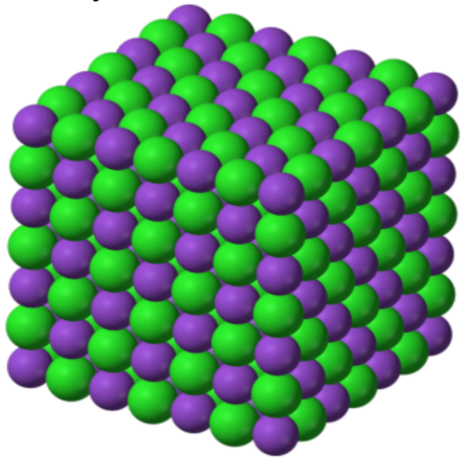
$$U(\mathrm{r}) = U_{bond}(\ldots) + U_{non-bond}(\ldots) + U_{ext}(\ldots)$$

- Evaluate the forces acting on each particle:

- **The force on each atom is determined by:**

$$\mathrm{F}_i = -\nabla U(\mathrm{r})$$

  - $U(\mathrm{r})$: potential function
  - N : number of atoms in the system
  - r_{ij} : vector distance between atoms *i* and *j*

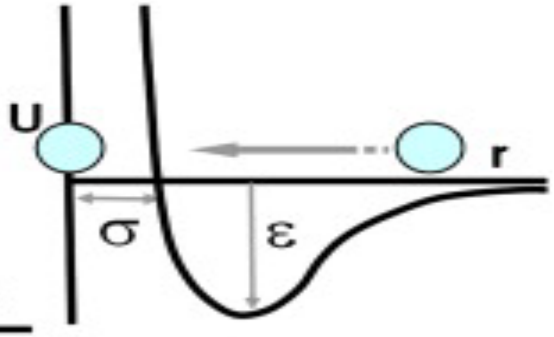


- Newton's equation of motion:

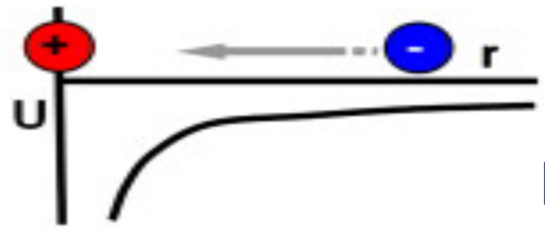
$$m_i \frac{d^2}{dt^2}\vec{x}_i = \vec{F}_i(\vec{x}_1, \ldots, \vec{x}_N) \qquad i = 1 \ldots N$$

Force Fields for MD simulations

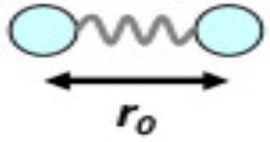
$$U = \sum_{i<j}\sum 4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right]$$



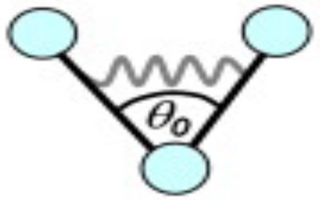
$$+ \sum_{i<j}\sum \frac{q_i q_j}{4\pi\varepsilon_0 r_{ij}}$$



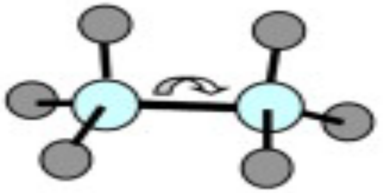
$$+ \sum_{bonds} \frac{1}{2} k_b (r - r_0)^2$$



$$+ \sum_{angles} \frac{1}{2} k_a (\theta - \theta_0)^2$$



$$+ \sum_{torsions} k_\phi [1 + \cos(n\phi - \delta)]$$



Interactions:

- Lenard-Jones
- Electrostatic
- Bonds
- Orientation
- Rotational

Derivation of Verlet algorithm

Taylor's expansions:

position

acceleration

$$r(t+\Delta t) = r(t) + \dot{r}(t)\Delta t + \tfrac{1}{2}\ddot{r}(t)\Delta t^2 + \tfrac{1}{6}\dddot{r}(t)\Delta t^3 + O(\Delta t^4) \qquad (I)$$

$$r(t-\Delta t) = r(t) - \dot{r}(t)\Delta t + \tfrac{1}{2}\ddot{r}(t)\Delta t^2 - \tfrac{1}{6}\dddot{r}(t)\Delta t^3 + O(\Delta t^4) \qquad (II)$$

velocity

Add (I) and (II):

$$r(t+\Delta t) + r(t-\Delta t) = 2r(t) + \ddot{r}(t)\Delta t^2 + O(\Delta t^4)$$

or:

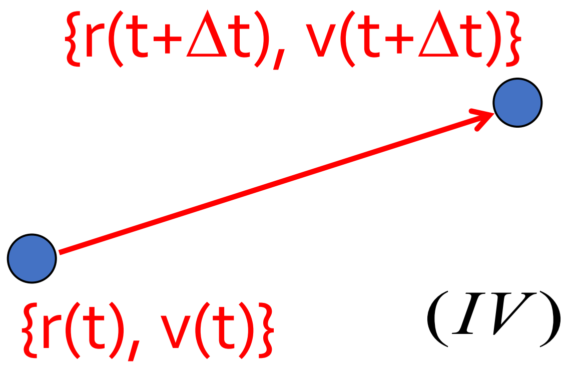
$$r(t+\Delta t) = 2r(t) - r(t-\Delta t) + f(t)\Delta t^2/m + O(\Delta t^4) \qquad (II)$$

Subtract (II) from (I):

$$r(t+\Delta t) - r(t-\Delta t) = 2\dot{r}(t)\Delta t + O(\Delta t^3)$$

or:

$$v(t) = \big(r(t+\Delta t) - r(t-\Delta t)\big)/2\Delta t + O(\Delta t^2) \qquad (IV)$$



Usage of Verlet algorithm

- From the initial positions and velocities: $\mathbf{r}_i(t)$ $\mathbf{v}_i(t)$

$$\mathbf{a}(\mathbf{r}) = \frac{1}{m}\mathbf{F}(\mathbf{r}(t))$$

- Obtain the positions and velocities at: $t + \Delta t$

  - velocity calculated explicitly
  - possible to control the temperature
  - stable in long simulation
  - most used algorithm

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

$$\mathbf{a}(t+\Delta t) = \frac{1}{m}F(\mathbf{r}(t+\Delta t))$$

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

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

- Leap-Frog algorithm

$$\begin{aligned}
\boldsymbol{v}\left(t+\frac{\Delta t}{2}\right) &= \boldsymbol{v}\left(t-\frac{\Delta t}{2}\right) + \frac{\boldsymbol{F}(t)}{m}\Delta t \\
\boldsymbol{r}(t+\Delta t) &= \boldsymbol{r}(t) + \boldsymbol{v}\left(t+\frac{\Delta t}{2}\right)\Delta t
\end{aligned}$$

Predictor – Corrector algorithm

- **Predictor step:**
  - from the initial $\mathbf{r}_i(t)$, $\mathbf{v}_i(t)$ ➔ $\mathbf{a}(\mathbf{r}) = \frac{1}{m}\mathbf{F}(\mathbf{r}(t))$
  - predict $\mathbf{r}_i(t+\Delta t)$, $\mathbf{v}_i(t+\Delta t)$ using Taylor's series

$$\mathbf{r}^P(t+\Delta t) \cong \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \frac{\mathbf{a}(t)}{2}\Delta t^2$$

$$\mathbf{v}^P(t+\Delta t) \cong \mathbf{v}(t) + \mathbf{a}(t)\Delta t$$

$$\mathbf{a}^P(t+\Delta t) \cong \mathbf{a}(t) + \mathbf{r}^{iii}(t)\Delta t$$

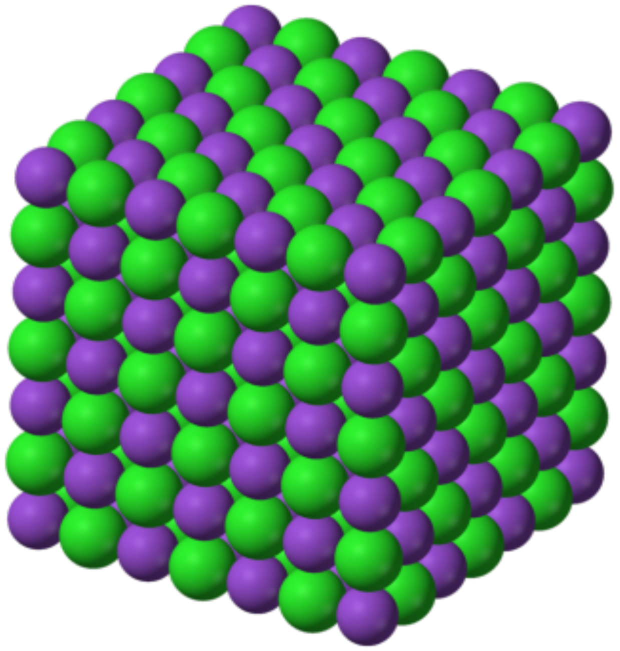
$\mathbf{r}^{iii}$: 3rd order derivatives

- **Corrector step:**
  - get corrected acceleration: $\mathbf{a}^C(\mathbf{r}) = \frac{\mathbf{F}(\mathbf{r}^P(t+\Delta t))}{m}$
  - using error in acceleration: $\Delta\mathbf{a}(t+\Delta t) \cong \mathbf{a}^C(t+\Delta t) - \mathbf{a}^P(t+\Delta t)$
  - correct the positions: $\mathbf{r}(t+\Delta t) \cong \mathbf{r}^P(t+\Delta t) + C_0\frac{\Delta t^2}{2}\Delta\mathbf{a}(t+\Delta t)$
  - correct the velocities: $\mathbf{v}(t+\Delta t) \cong \mathbf{v}^P(t+\Delta t) + C_1\Delta t\Delta\mathbf{a}(t+\Delta t)$

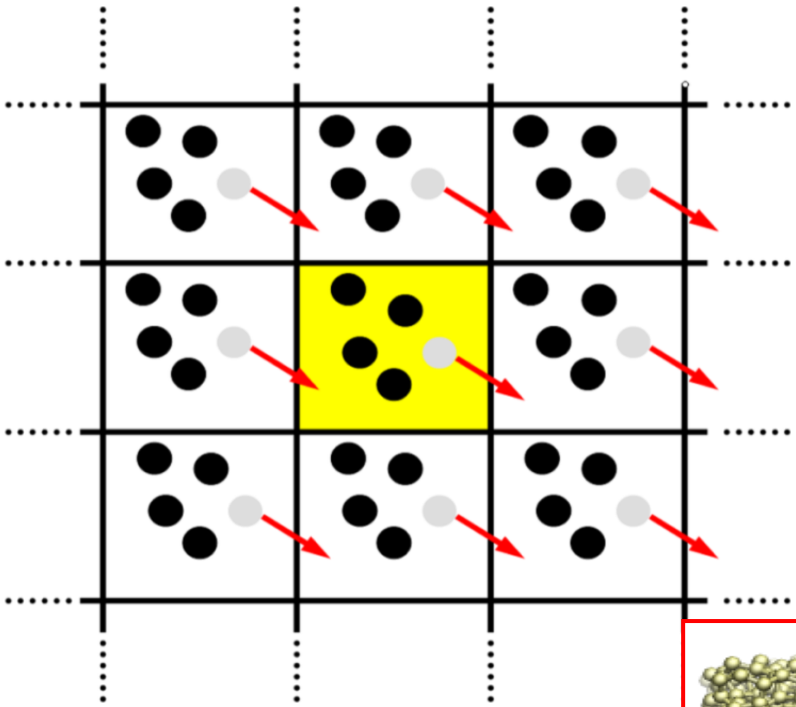
C_n: constants depending accuracy

MD simulation settings

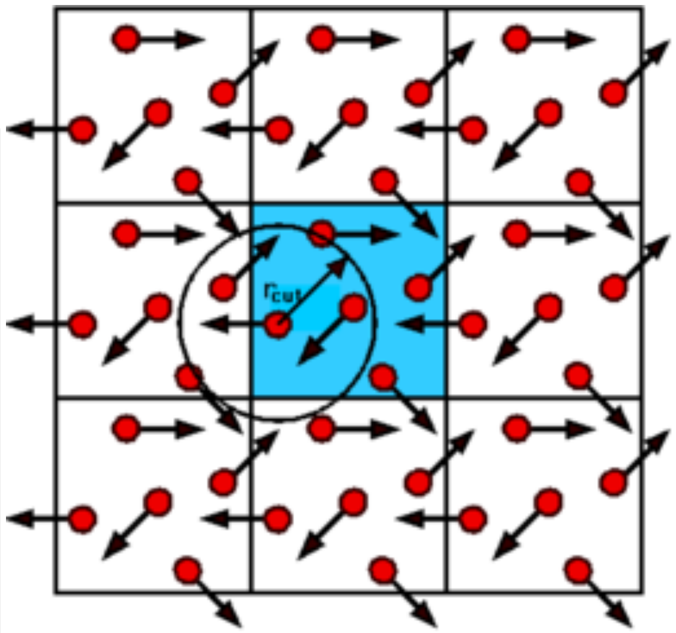
- Starting configuration:
  - Atomic positions (x,y,z)
  - density, mass, charge, ….
- Initial velocities: depend on temperature
- boundary conditions (PBC):
  - PBC: required to simulate bulk properties.
  - or fixed boundary conditions
- set the appropriate potential:
  - available and supported potentials
  - depend on the system to simulate (literature search).
- set the appropriate time step: should be short (order of 1fs).
- set the temperature and pressure control:
  - define the thermodynamic ensemble (NVT, NPT, NVE, …).
- Fix run time and customize the output: depend on the software.



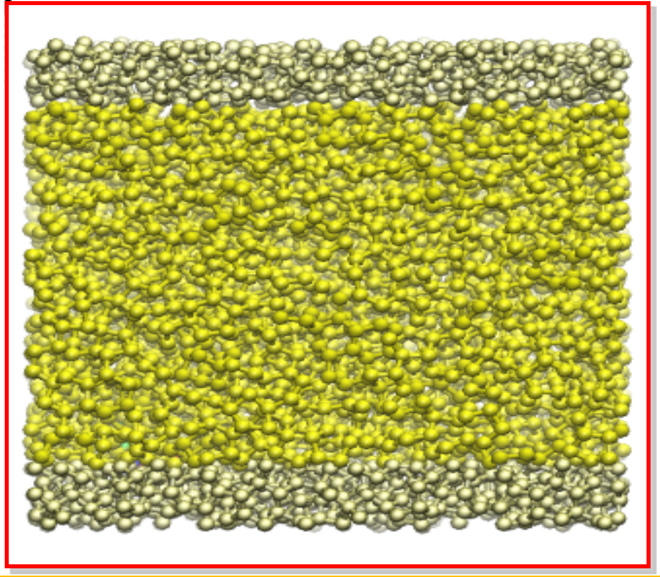
Boundary conditions



- create images of the simulation box: duplication in all directions (x, y and z)
- if an atom is moving out of boundary, it comes from the other side.
- used also in pair interactions evaluation



- **PBC:**
  in x, y directions
- **Walls:**
  fixed boundaries in z direction.

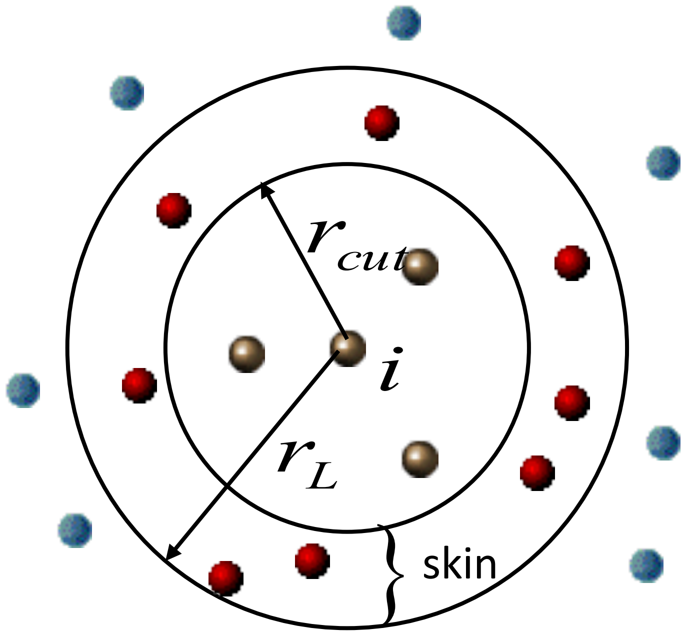


Neighbor list

- **Optimization of MD algorithms:**
- **Evaluating the forces is time consuming:**
  - **Pair potential calculation:** $\propto O(\mathrm{N}^2)$
  - **Atom moves** $< 0.2\,\mathring{\mathrm{A}}$ **per time step**
  - **Cutoff radius: not necessary to include all the possible pairs.**
- **Solution: Verlet neighbor list**
- **Containing all neighbors of each atom within:** r_L
- **Update every** N_L **steps**

For each particle: N-1 pairs.

For N particles: N(N-1) pairs.



$$r_L - r_{cut} > \frac{N_L \bar{\mathbf{v}} \Delta t}{2}$$

Thermodynamic ensembles

- **Ensembles:**
  - **NVE – micro-canonical ensemble**
  - **NVT – canonical ensemble**
  - **NPT – grand-canonical ensemble**
  - **others …**

Each ensemble is used for a specific simulation:

- *Equilibration, …*
- *Production run, …*
- *Diffusion (NVE), …*

- **Temperature control:**
  - **Berendsen thermostat (velocity rescaling)**
  - **Andersen thermostat**
  - **Nose-Hoover chains**

- **Pressure control:**
  - **Berendsen volume rescaling**
  - **Andersen piston**

✓ *Choose the ensemble that best fits your system and the properties you want to simulate*
✓ *start the simulation.*
✓ *Check the thermodynamic properties as a function of time.*

What is MD used for?

- **Goal of MD simulations:**
  - The prime purpose of MD is to sample the *phase space* of the statistical mechanics ensemble.
  - Most physical properties can be related the atomic trajectories and obtained as average as a function of time.

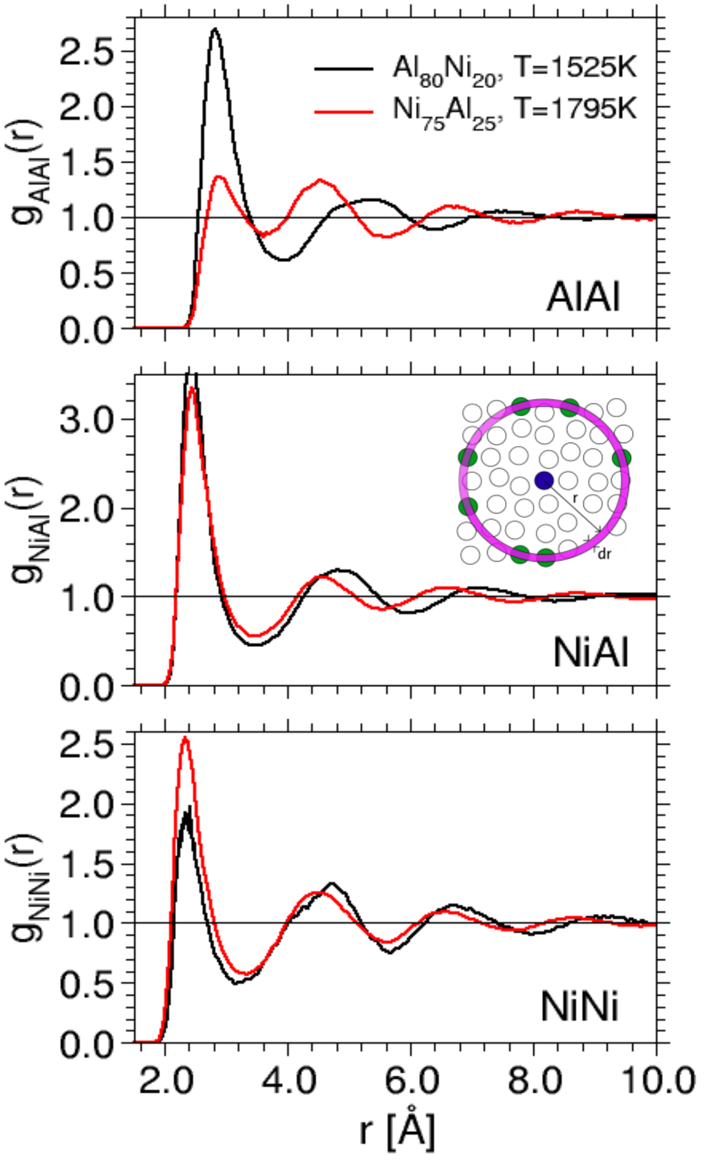
- **Structural properties:**
  - obtained from spatial correlation functions e.g. distribution functions (RDF, S(Q), Van-Hove, ...).

- **Dynamical Properties:**
  - Time dependent properties (MSD, diffusion coefficients) obtained via temporal correlation functions e.g. velocity autocorrelation function, atomic displacements.

Thermodynamic properties

- Kinetic Energy $\xrightarrow{\quad}$ $$\langle K.E.\rangle = \left\langle \frac{1}{2}\sum_{i}^{N} m_i v_i^2 \right\rangle$$
- Temperature $\xrightarrow{\quad}$ $$T = \frac{2}{3Nk_B}\langle K.E\rangle$$
- Configuration Energy $\xrightarrow{\quad}$ $$U_c = \left\langle \sum_{i}\sum_{j>i}^{N} V(r_{ij}) \right\rangle$$
- Pressure $\xrightarrow{\quad}$ $$PV = Nk_BT - \frac{1}{3}\left\langle \sum_{i=1}^{N-1}\sum_{j>i}^{N} \vec{r}_{ij}\cdot\vec{f}_{ij} \right\rangle$$
- Specific Heat $\xrightarrow{\quad}$ $$\left\langle \delta(U_c)^2 \right\rangle_{NVE} = \frac{3}{2}Nk_B^2T^2\left(1-\frac{3Nk_B}{2C_v}\right)$$

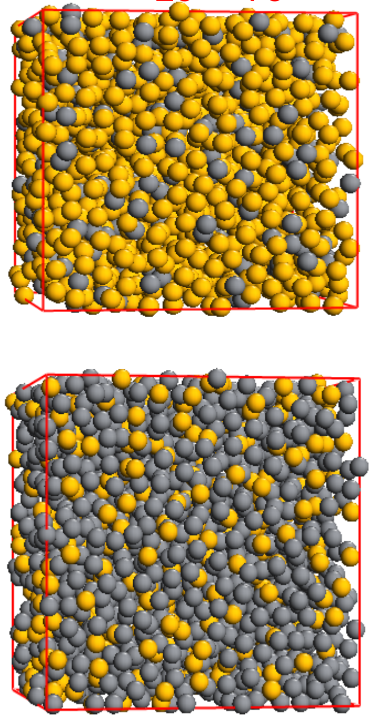
Structural properties



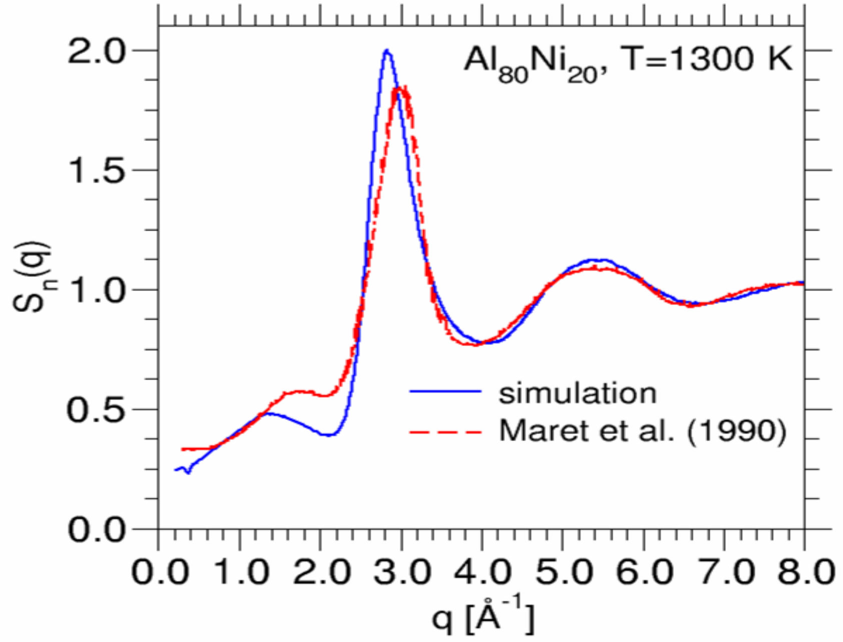
- **Radial Distribution Function (simulation)**

$$g(r) = \frac{\langle n(r) \rangle}{4\pi\rho r^2 \Delta r} = \frac{V}{N^2} \left\langle \sum_i \sum_{j \neq i}^{N} \delta(r - r_{ij}) \right\rangle$$

$Al_{25}Ni_{75}$



$Al_{80}Ni_{20}$



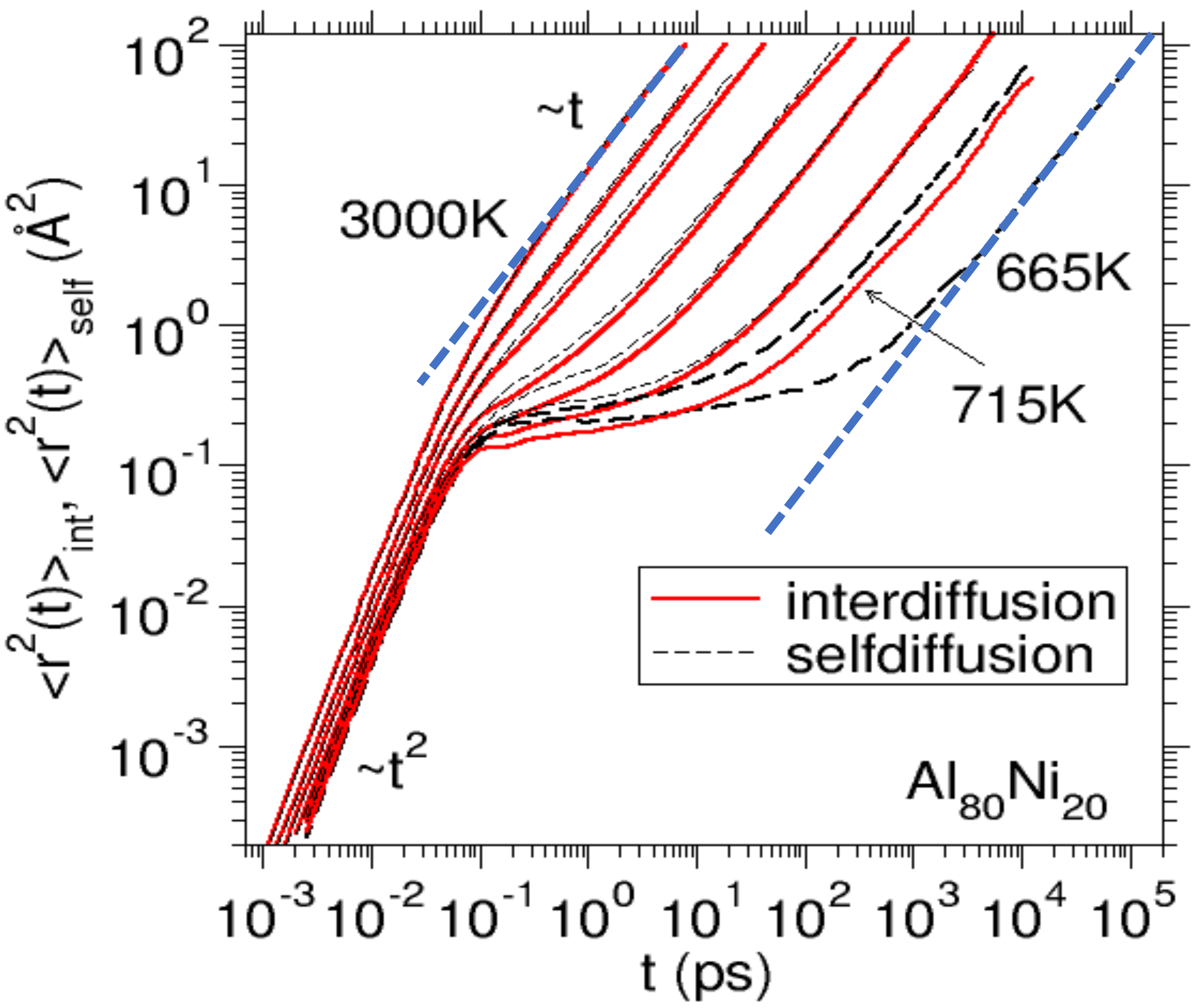
$$S(k) = 1 + 4\pi\rho \int_0^\infty \frac{\sin(kr)}{kr} (g(r) - 1) r^2 dr$$

- **Structure Factor (experiments)**

WESTGRID

Dynamical properties

MSD: Mean Square Displacement (Einstein relation)



$$2Dt = \frac{1}{3}\left\langle |\mathbf{r}_i(t) - \mathbf{r}_i(0)|^2 \right\rangle$$

$$\mathrm{MSD} = c_{\mathrm{Al}}\left\langle (\vec{r}_{\mathrm{s,Ni}}(t) - \vec{r}_{\mathrm{s,Ni}}(0))^2 \right\rangle + c_{\mathrm{Ni}}\left\langle (\vec{r}_{\mathrm{s,Al}}(t) - \vec{r}_{\mathrm{s,Al}}(0))^2 \right\rangle$$

Diffusion constants

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

Recommended textbooks

- **The Art of Molecular Dynamics Simulation**, D.C. Rapaport, Camb. Univ. Press (2004)
- **Understanding Molecular Simulation**, D. Frenkel and B. Smit, Academic Press (2002).
- **Computer Simulation of Liquids**, M.P. Allen and D.J. Tildesley, Oxford (1989).
- **Theory of Simple Liquids**, J.-P. Hansen and I.R. McDonald, Academic Press (1986).
- **Classical Mechanics**, H. Goldstein, Addison Wesley (1980).
- **Glassy Materials and Disordered Solids, An Introduction to their Statistical Mechanics**, 2nd edition, Kob, Walter and Binder, K., 2011

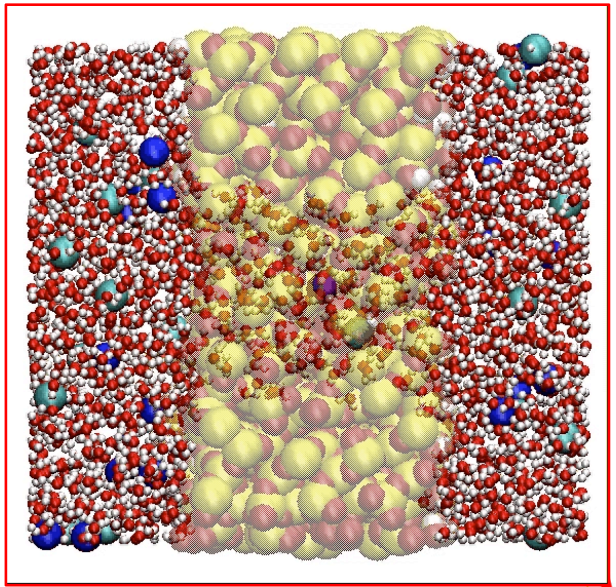
Available MD codes

- ❑ **Open source: free access**
  - ✓ **LAMMPS:** http://lammps.sandia.gov/index.html
  - ✓ **DL_POLY:** http://www.scd.stfc.ac.uk/SCD/44516.aspx
  - ✓ **CP2K:** https://www.cp2k.org/about
  - ✓ **NAMD:** http://www.ks.uiuc.edu/Research/namd/
  - ✓ **GROMACS:** http://www.gromacs.org/
  - ✓ ….

- ❑ **Commercial software:** Amber
  - ✓ **Amber:** http://ambermd.org/

- ❑ **Home made codes:**
  - ✓ C, C++
  - ✓ Fortran, … etc.
  - ✓ Python, … etc.

- ❑ **Visualization:**
  - ➢ VMD
  - ➢ OVITO, …
- ❑ **Analysis?**



Flow of water and ions thru a silica pore

Molecular Dynamics: some results

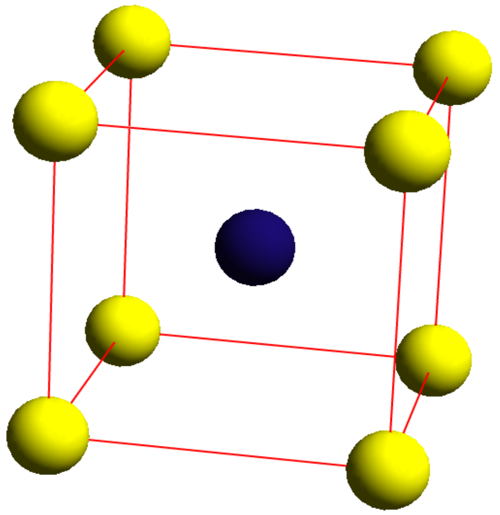
- **Binary Metallic alloys:**
  - Melting and crystallization.
  - Solid-Liquid interfaces.
  - Crystal growth from melt.
  - Crystal growth is diffusion limited process.

- **Glasses:**
  - Shear deformations in amorphous materials
  - How to prepare a glass using MD simulation?
  - Glass Indentation using MD.

Solid-Liquid interface velocities

Why B2-$Al_{50}Ni_{50}$?

- ✓ B2-$Al_{50}Ni_{50}$: prototype of binary ordered metals
- ✓ simulations of interfacial growth in binary systems rare
- ✓ growth kinetics of binary metals: diffusion limited?
- ✓ crystal growth slower than in one-component metals
- ✓ understand crystal growth of alloys on microscopic level



Questions:

- ➢ crystal growth & accurate estimation of T_m?
- ➢ solid-liquid interface velocity from interface motion?
- ➢ kinetic coefficients and their anisotropy?
- ➢ solid-liquid interface motion controlled by mass diffusion?
- ➢ solid-liquid coexistence, interface structure?
- ➢ how to distinguish between solid-like & liquid-like particles?

- ➢ Frenkel J., Phys. Z. Sowjetunion, **1** (1932) 498.
- ➢ Wilson H.A., Philos. Mag. , **50** (1900) 238.

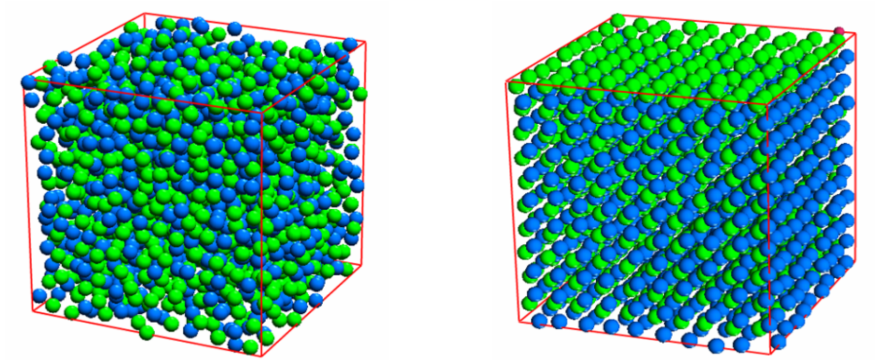
Solid-Liquid interfaces: Al-Ni

❑ solve Newton's equation of motion for system of *N* particles:

- **velocity Verlet algorithm (time step = 1 fs)**
- **NPT ensemble:**
  - ➢ **constant pressure (Anderson algorithm): p = 0**
  - ➢ **constant temperature: stochastic heat bath**
- **periodic boundary conditions in all directions**

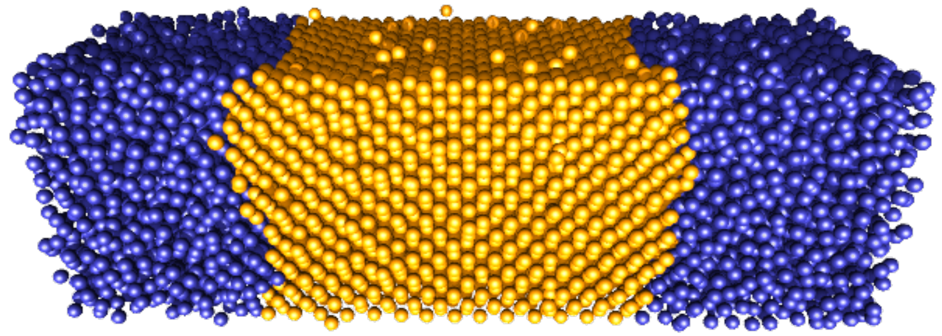
Allen M.P. and Tildeslay D.J., Computer simulation of liquids, 1987
Anderson H.C., JCP **72** (1980) 2384

MD of pure systems



- ➢ lattice properties
- ➢ T dependence of density
- ➢ Structural quantities
- ➢ Self-diffusion constant

MD of inhomogeneous systems



- ➢ accurate melting temperature T_m
- ➢ kinetic coefficients & their anisotropy
- ➢ solid-melt interface structure
- ➢ crystal growth

Simulation parameters

- Binary metallic mixtures - **simple**: Lennard-Jones potential
  - **better**: **EAM**

- **EAM** potential:

$$U_{\text{pot}} = \frac{1}{2}\sum_{k,l} u(r_{kl}) + \sum_{k} F(\bar{\rho}_k)$$

$$\bar{\rho}_k = \sum_{l \neq k} \rho_l(r_{kl})$$

- two body interactions.
- many body interactions (e-density).
- fitting to both experimental and *ab-initio* data.
- reproduces the lattice properties & point defects.
- structure and dynamics of AlNi melts.

Y. Mishin *et al.*, PRB **65**, (2002) 224114.
J. Horbach *et al.*, PRB **75**, (2007) 174304.

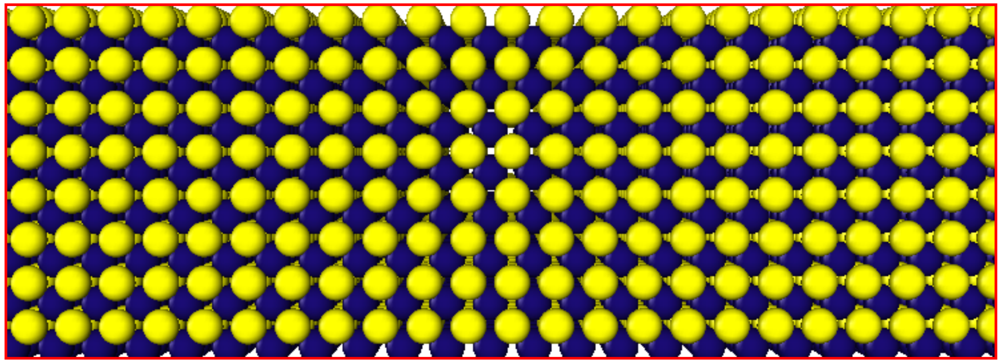
- **Solid** and **liquid** properties:

2000 particles ($L_x = L_y = L_z = 24.6$ Å)

- **Solid-liquid** interfaces (N particles):

$N_{Al} = N_{ni} \Rightarrow D = L_z \simeq 3 \times L_x \simeq 3 \times L_y$

10386 and 12672 particles, … or more



$\longleftarrow$ $D \simeq 3 \times L_x$ $\longrightarrow$

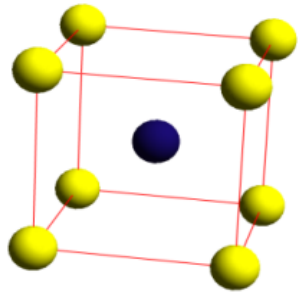
Heating and cooling: pure phases

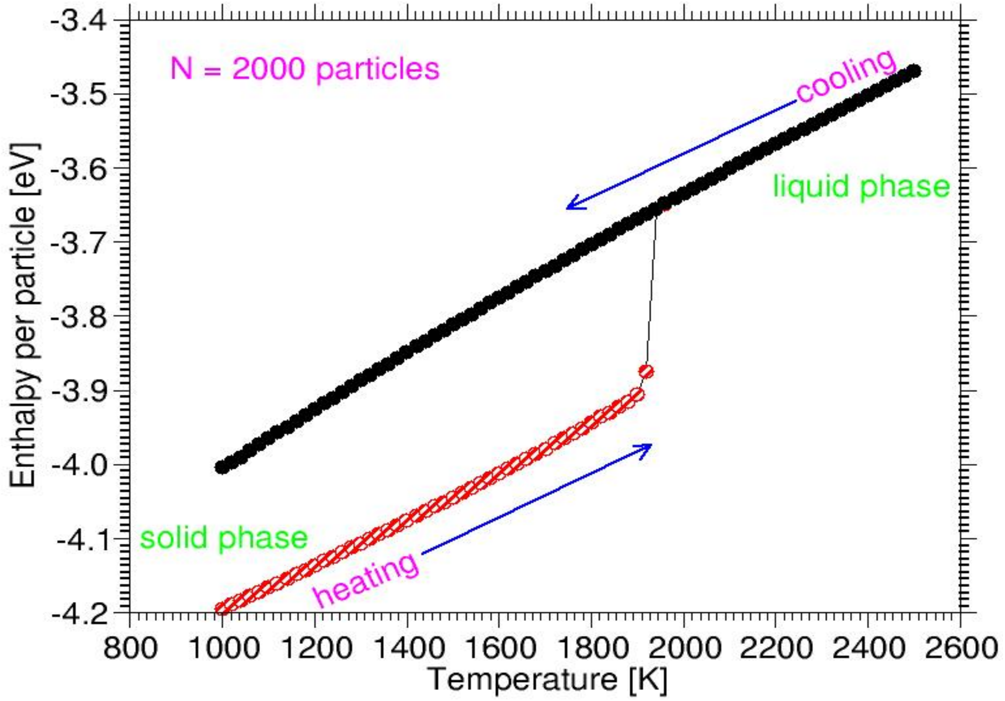
- ❑ How to go from crystal to melt & from melt to crystal?

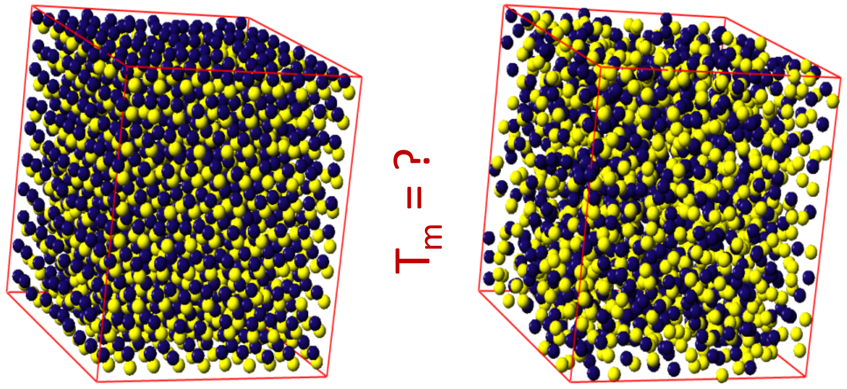
✓ start from B2 phase: equilibration at 1000 K

✓ try to melt the crystal: heating process

✓ cool down the melt: cooling process







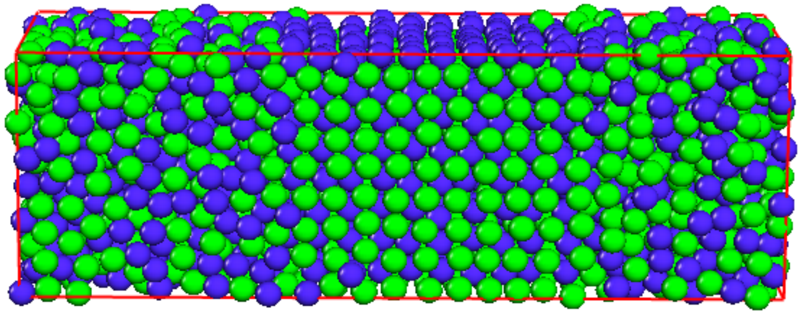
- ➢ binary alloys: glass formers.
- ➢ crystallization: process too slow
- ➢ brute force method:

not appropriate to estimate T_M

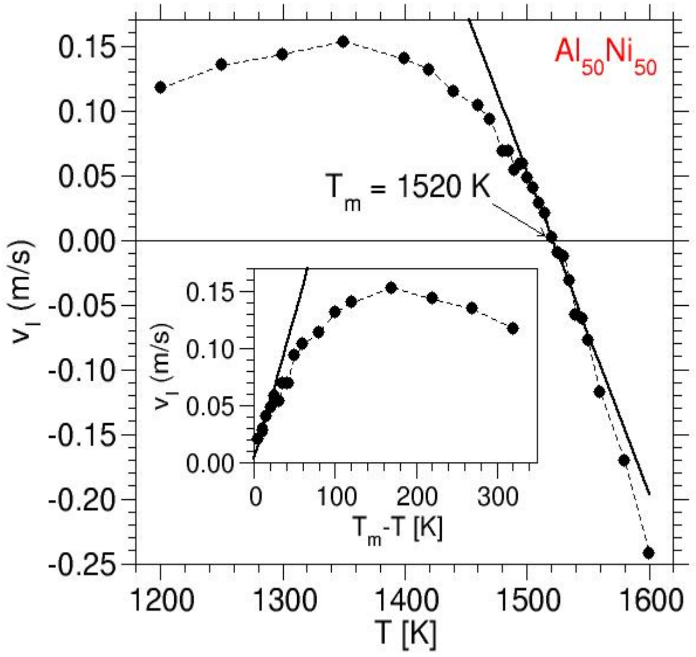
- ❑ How to study crystallization?

Estimation of melting temperature

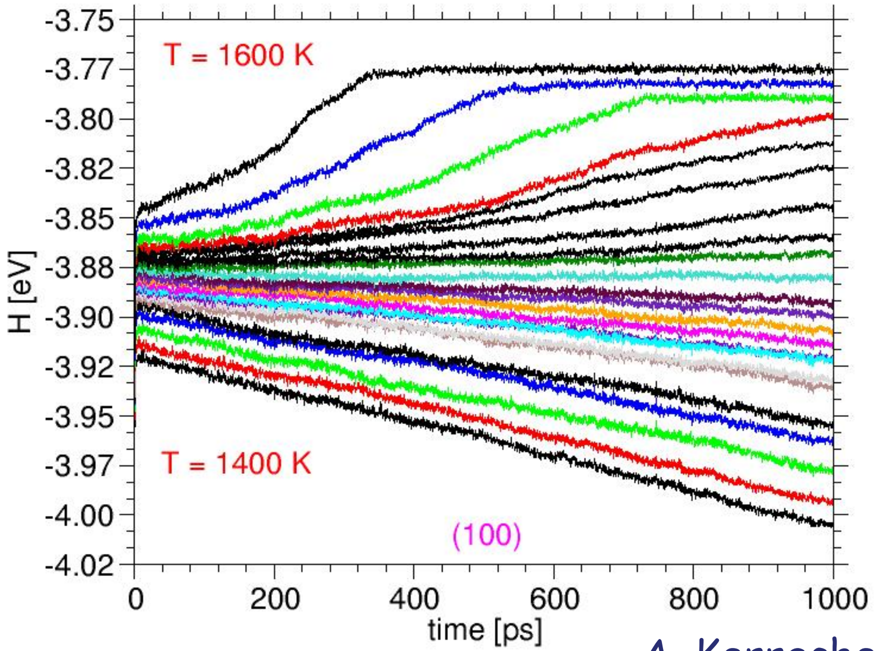
- ❑ How to prepare a system with two pahes?
- ➢ Equilibrate a crystal (NPT, p=0)
- ➢ Fix the particles in the middle of the box
- ➢ Heat away the two other regions
- ➢ Quench at the target temperature



Estimation of the melting temperature T_M from solid-liquid interface motion



Interface velocity



Enthalpy as a function of time

$T > T_M$: *Melting*

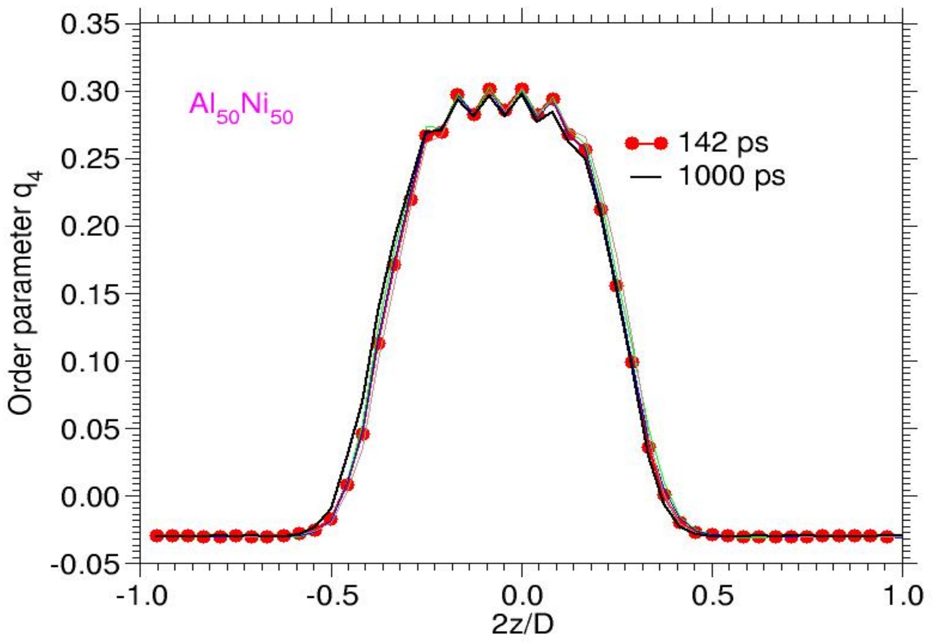
$T = T_M$: *coexistence*

$T < T_M$: *crystallization*

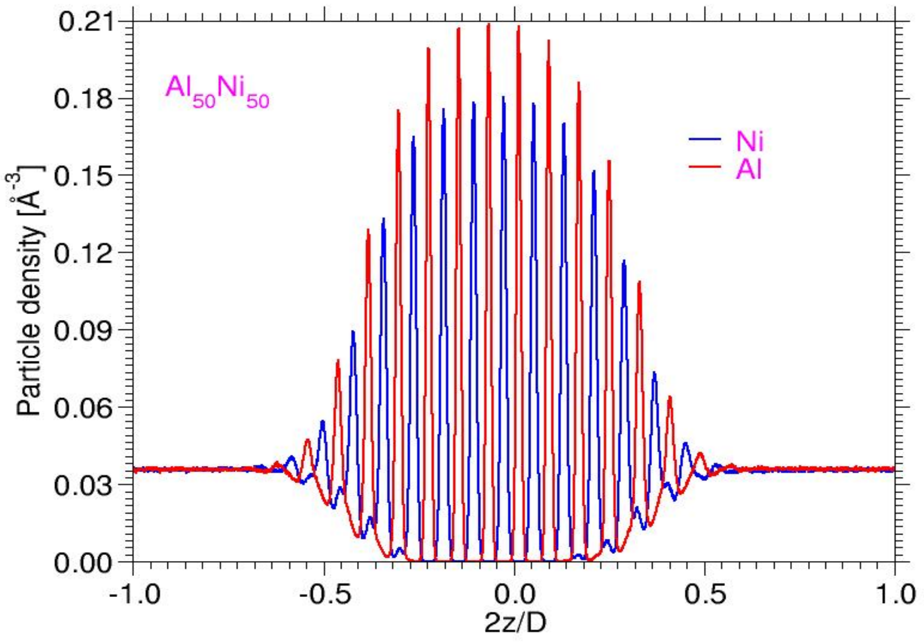
A. Kerrache et al., EPL 2008.

Characterization of SL interfaces

Bond order parameter profile



Partial particle density profile



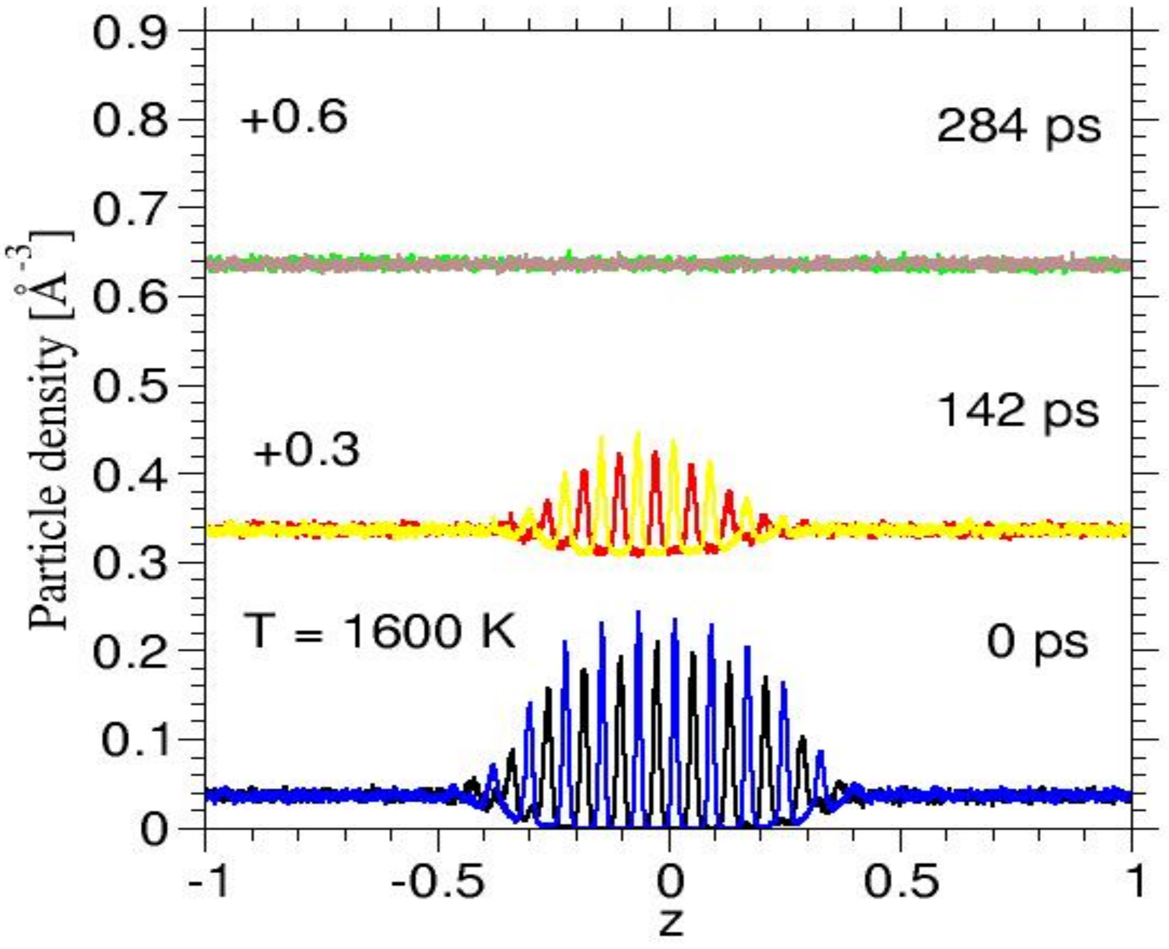
$$q_n = \left\langle \frac{1}{N} \sum_{i,j,k} \cos\left(n\theta_{xy}(i,j,k)\right) \right\rangle$$

$$n = 1,2,\ldots,6$$

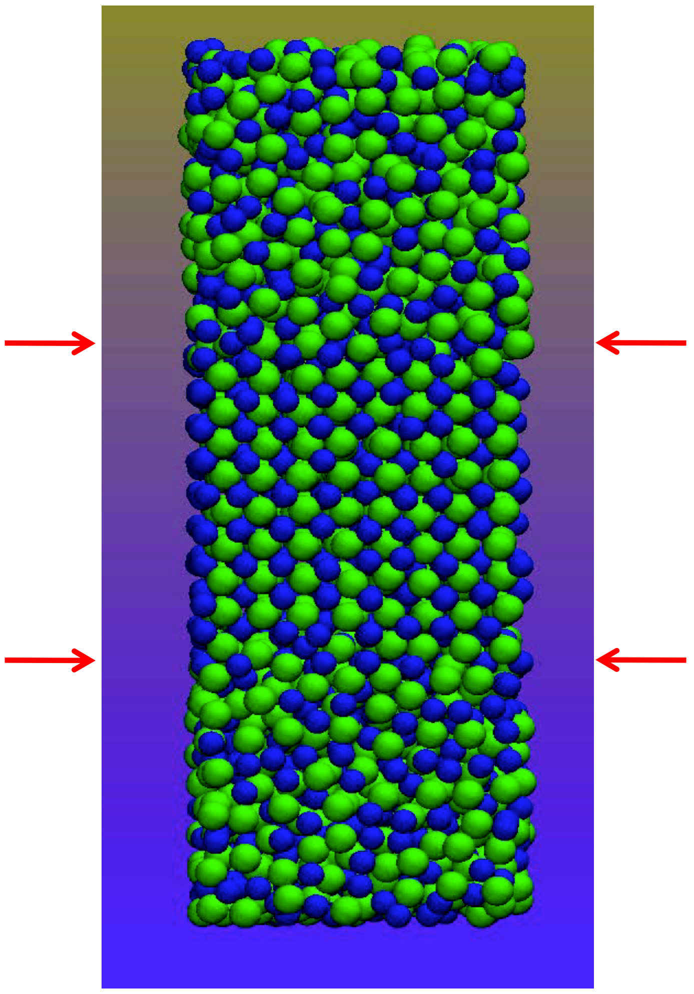
i,j and k: indices for nearest neighbors, θ(i,j,k): bond angle formed by i, j and k atoms.

- constant density in the liquid region.
- solid-liquid interface over several layers.
- pronounced chemical ordering in the solid region.
- mass transport required for crystal growth.

Melting of AlNI at 1600 K



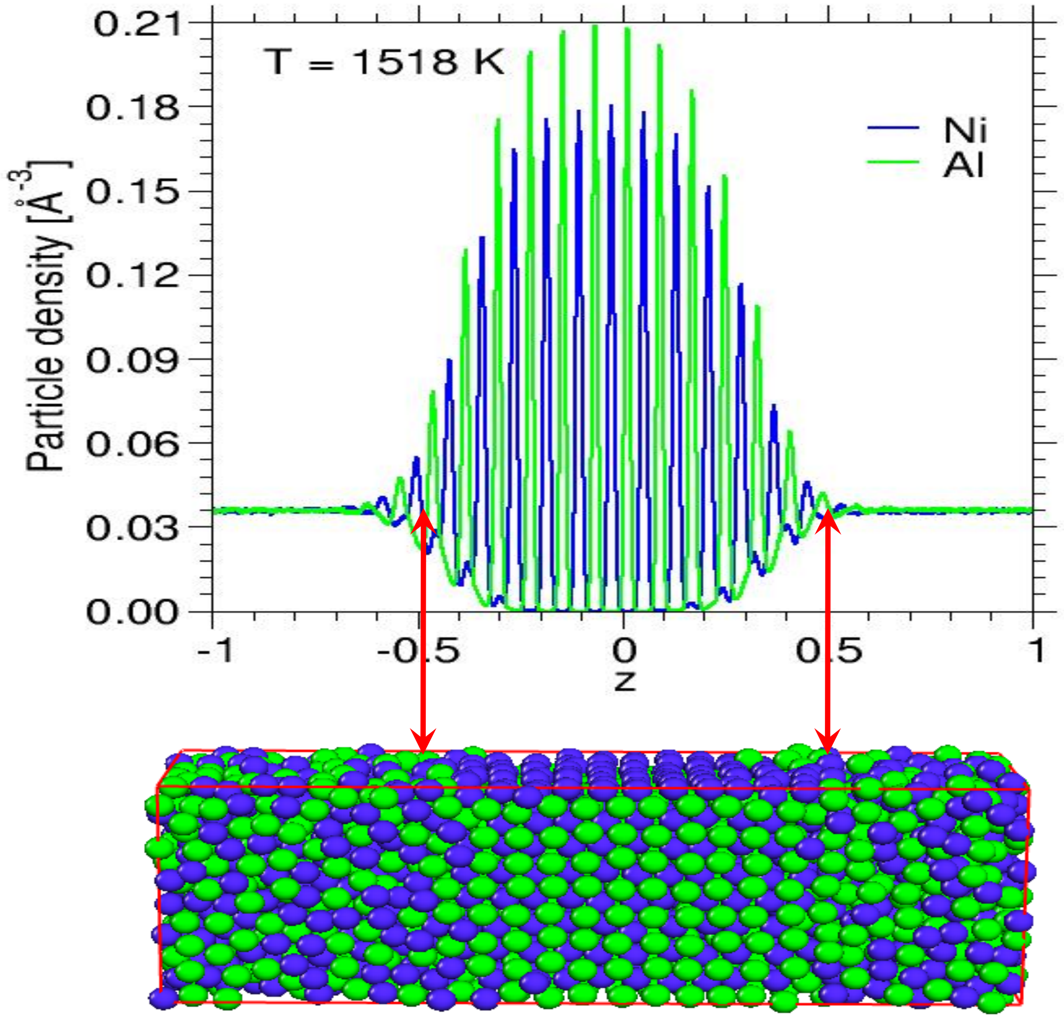
Particle density along the solid-liquid interface

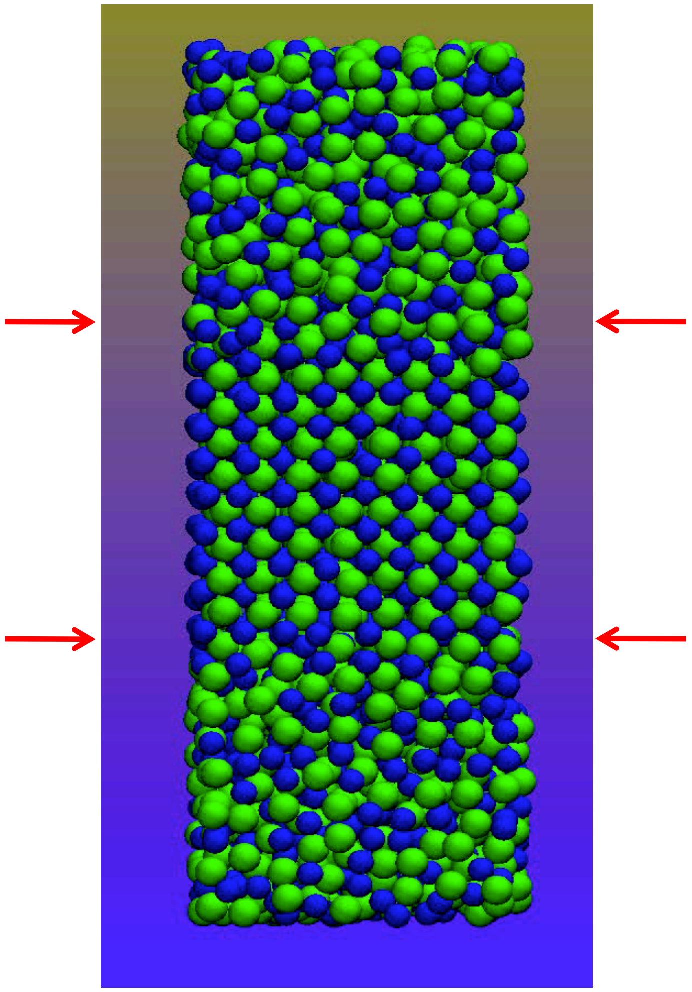


- Melting

Solid-Liquid interface coexistence at 1518 K

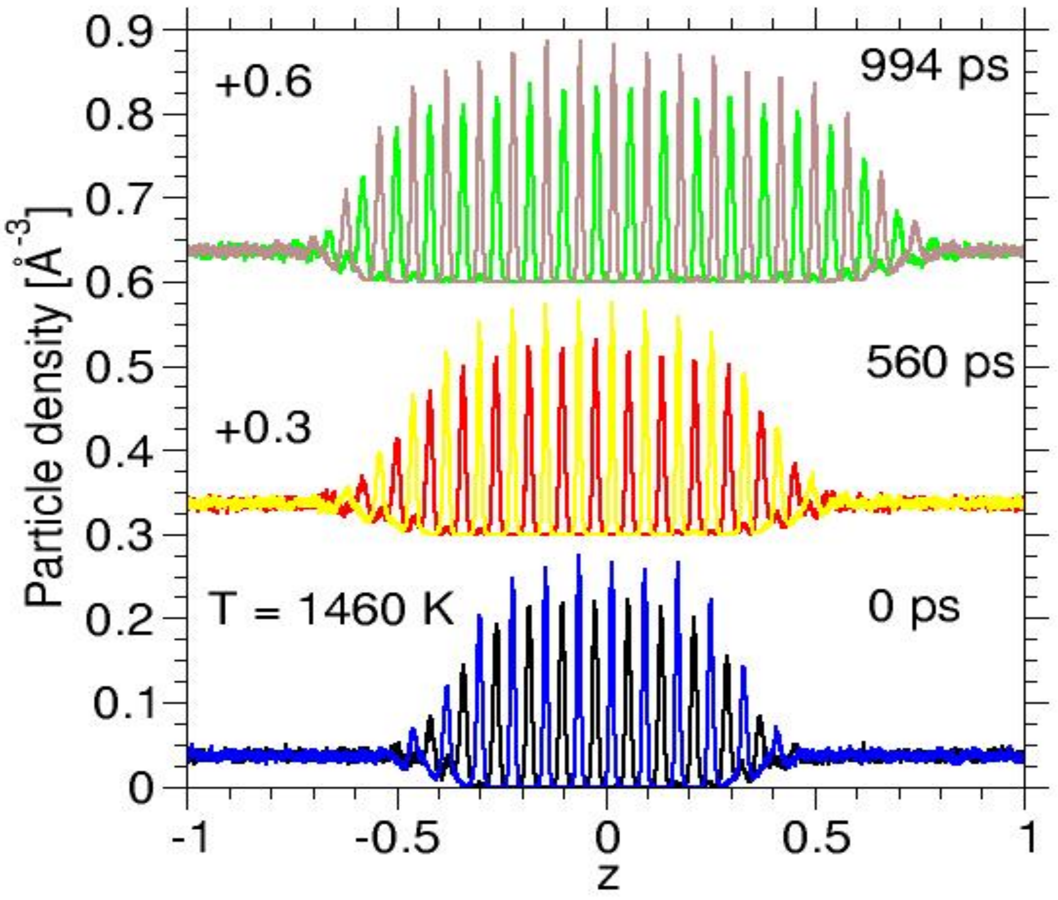
Particle density along the solid-liquid interface



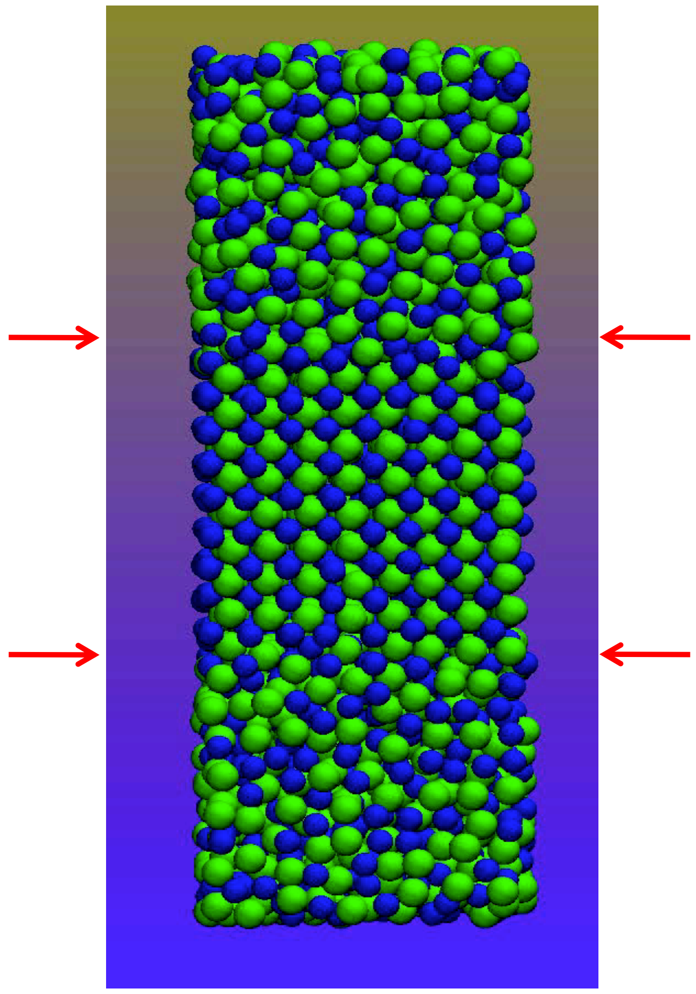


❑ Coexistence

Crystallization of AlNi at 1400 K

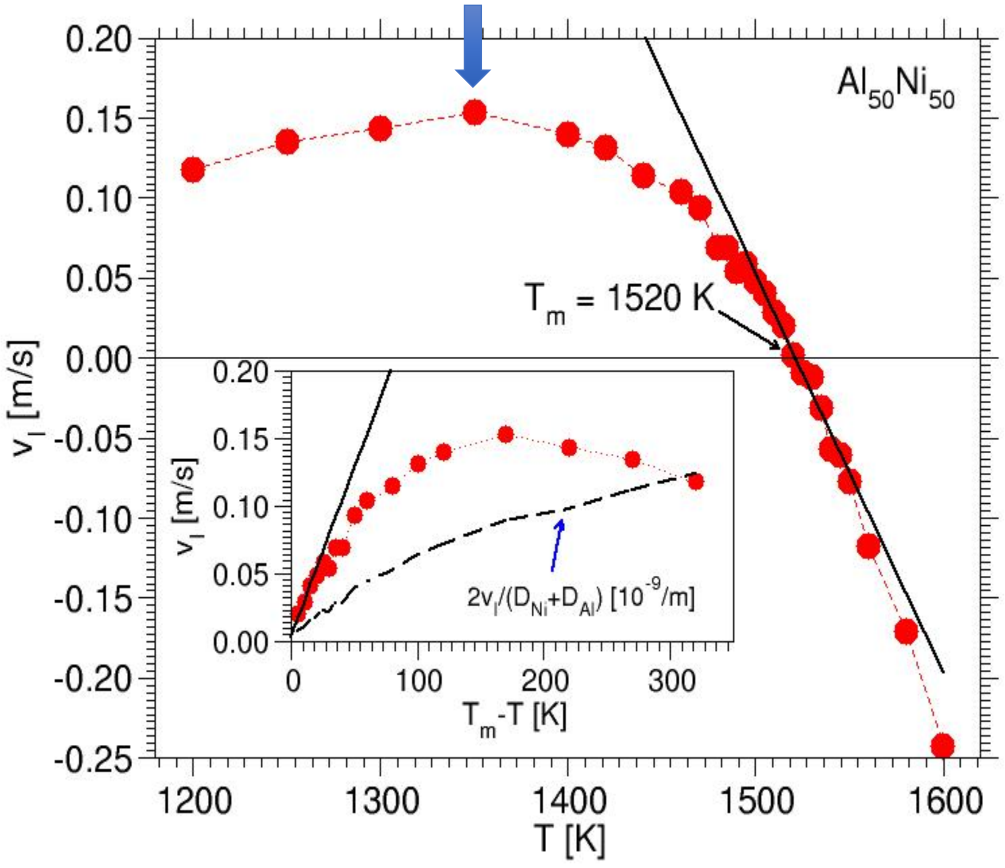


Particle density along the solid-liquid interface



❑ Crystallization

Crystal growth: diffusion limited



Solid-liquid interface velocity as a function of temperature

Inset: as a function of under-cooling

❑ Why the solid-liquid interface velocity presents a maximum?

✓ Maximum of 0.15 m/s at 180 K

Interface velocity divided by the average self diffusion constant.

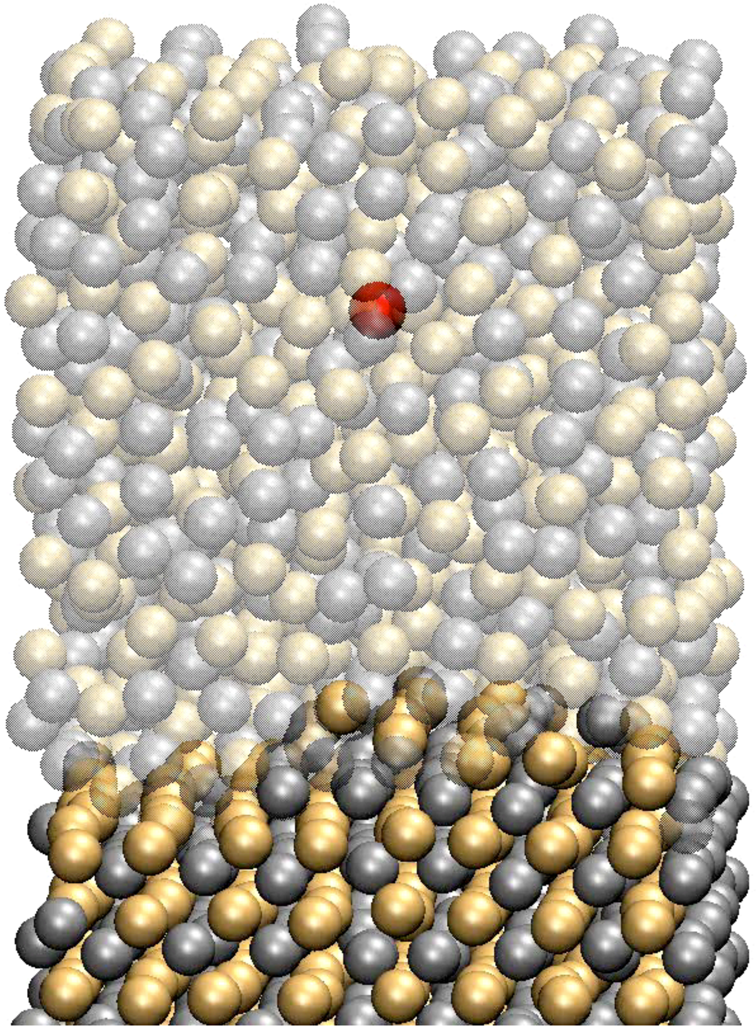
✓ Maximum due to decreasing of diffusion constant.

✓ Linear regime only up to 30 K of under-cooling.

What about the mass transport across the solid-liquid interface?

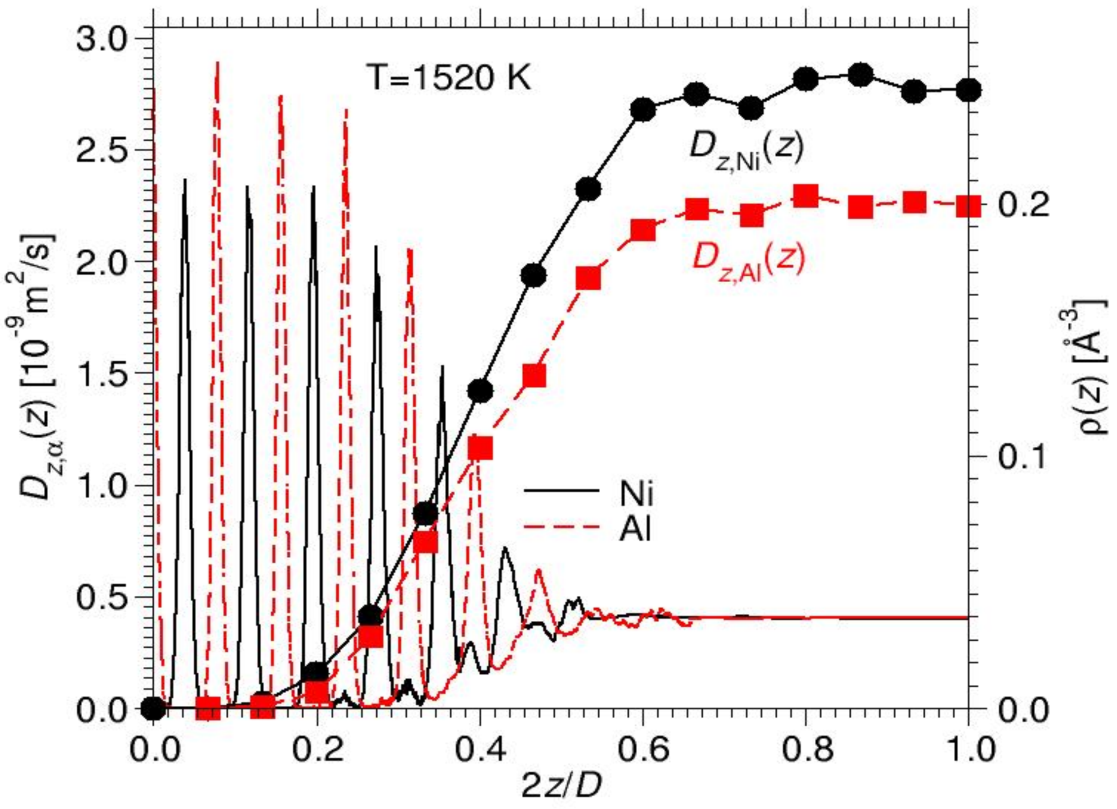
Mass transport across the interface

- Role of the mass transport on the crystal growth:
  - order parameter to distinguish solid and liquid particles locally.
  - compute the particle density and mass density profiles.
  - order parameter profile.
  - number of solid-like particles.
  - solid-liquid interface velocities from the number of solid-like particles.
  - diffusion across the interface.



Mass transport on the liquid phase and across the interface

Mass transport across the interface



Mass transport and particle density across the solid-liquid interface

$$D_{z_s,\alpha}(z_s) = \lim_{t\to\infty} \frac{1}{N_s} \sum_{i_s=1}^{N_s} \frac{\left\langle (z_{i_s}(t) - z_{i_s}(0))^2 \right\rangle}{2t}$$

The diffusion constants decrease when we cross the solid-liquid interface.

Wilson-Frenkel theory:
activated process controlled by mass diffusion in the liquid phase

*Wilson H.A. Philos. Mag. , **50** (1900) 238.*
*Frenkel J., Phys. Z. Sowjetunion, **1** (1932) 498.*
A. Kerrache et al. EPL, 2008.

Crystal growth: controlled by mass transport in the liquid phase and **solid-liquid** interface

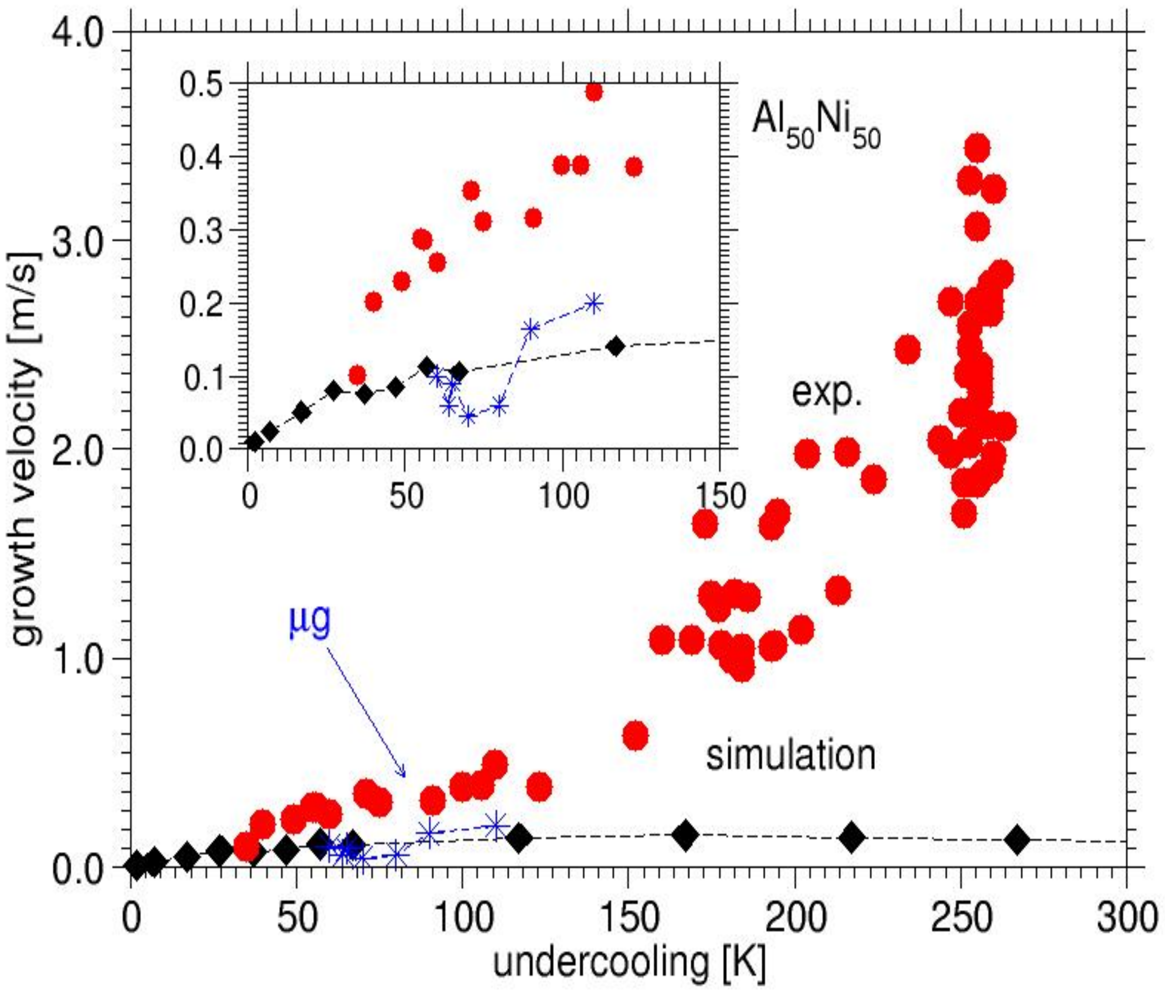
Experimental data?

Comparison to experimental data

Interface velocities: simulation, lab. exp., micro-gravity experiments

- ✓ terrestrial data (Assadi *et al.*)
- ✓ μg data (parabolic flight) , H. Hartmann (PhD thesis)

H. Assadi, *et al.*, Acta Mat. 54, 2793 (2006).



*A. Kerrache et al., EPL **81** (2008) 58001.*

Molecular Dynamics: some results

- **Binary Metallic alloys:**
  - Melting and crystallization.
  - Solid-Liquid interfaces.
  - Crystal growth from melt.
  - Crystal growth is diffusion limited process.

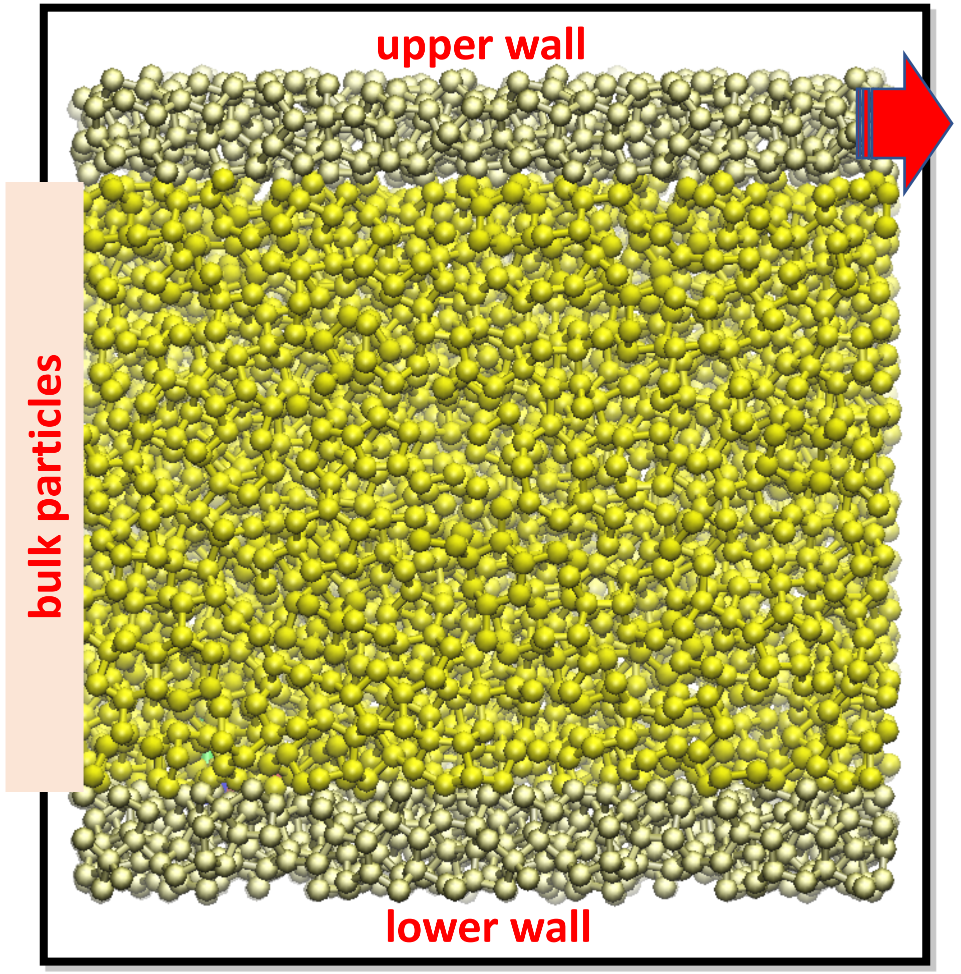
- **Glasses:**
  - **Shear deformations in amorphous materials**
  - **How to prepare a glass using MD simulation?**
  - **Glass Indentation using MD.**

Shear deformations in amorphous silicon

- equilibration of the sample at the desired temperature (Bulk simulation).
- define the lower & upper walls.
- move the particles of upper wall with a fixed shear velocity v_s
- integrate the equation of motion of the mobile particles.
- equilibration for 5 ns: fixed walls
- periodic boundary conditions fixed in y direction.

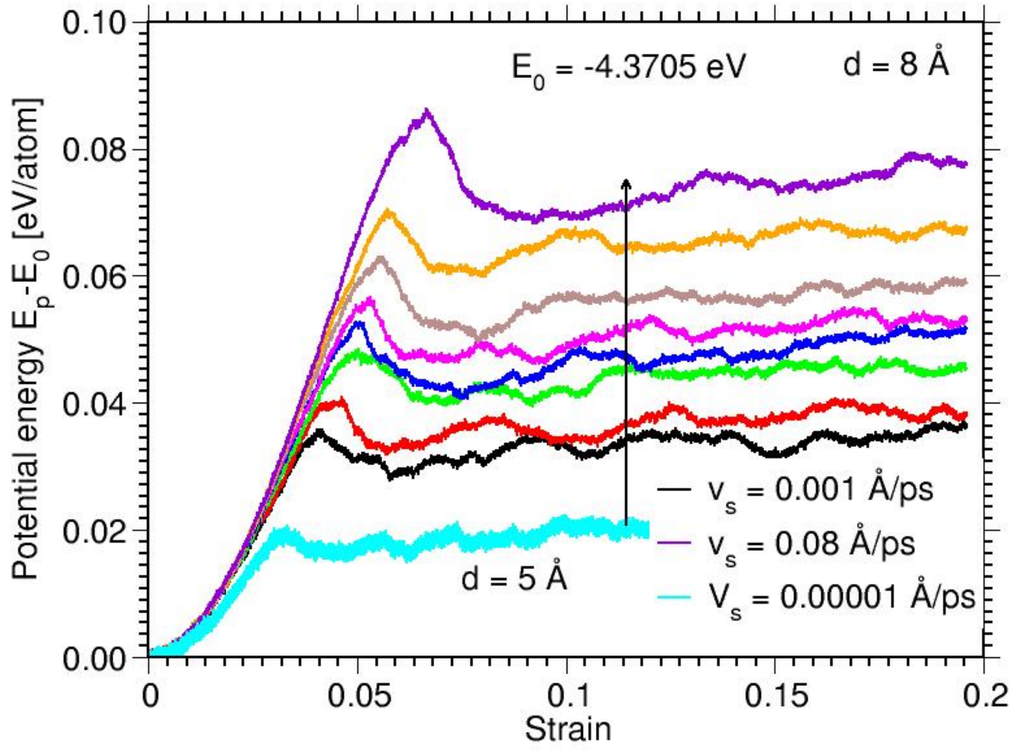
❑ Temperature:

rescaling the velocities using the y and z components

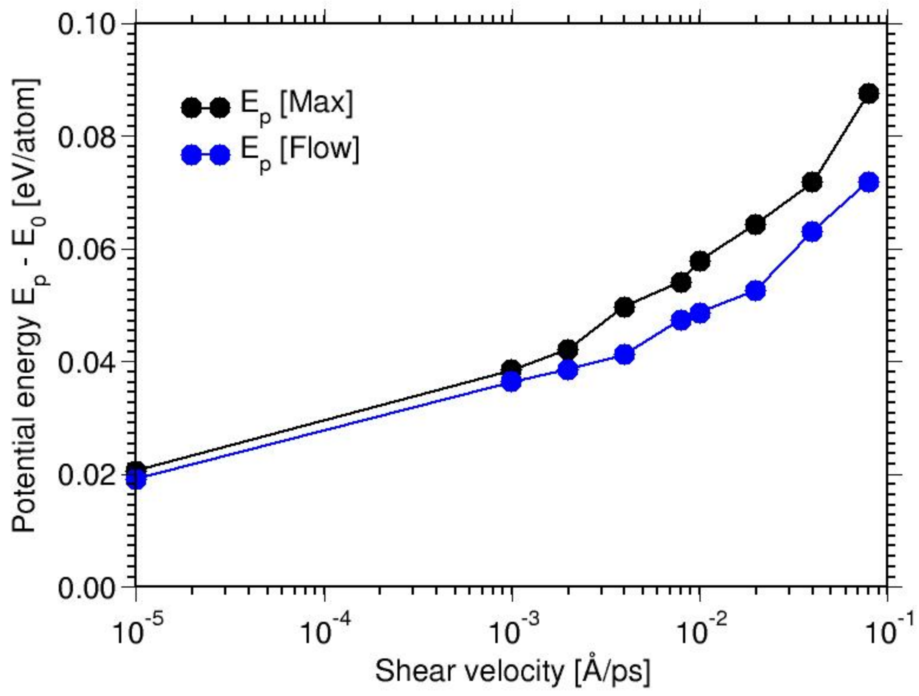


Typical starting configuration of s-Si

Shear deformations at 300 K



Potential energy difference ΔE as a function of imposed strain at 300 K



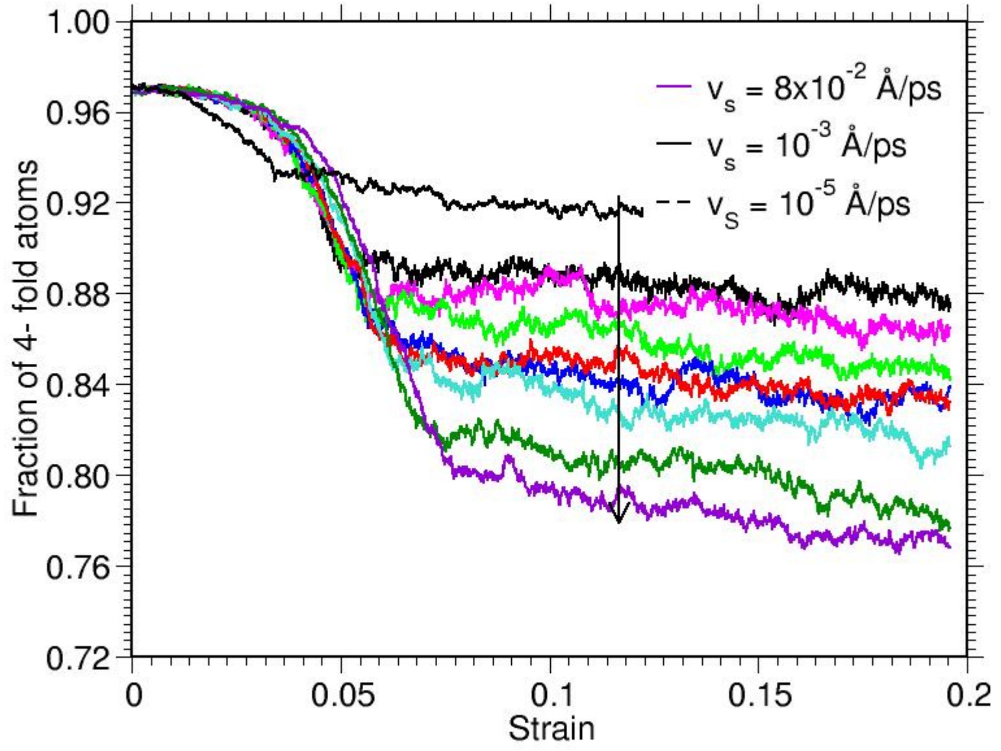
- Shear velocity: 10^{-5} to 8×10^{-2} Å/ps
- Quadratic increase of E_p and strain
- independent at small strain.
- E_{max} increases with shear velocity.

10^{-3} Å/ps ➔ 8 ns
10^{-5} Å/ps ➔ 800 ns

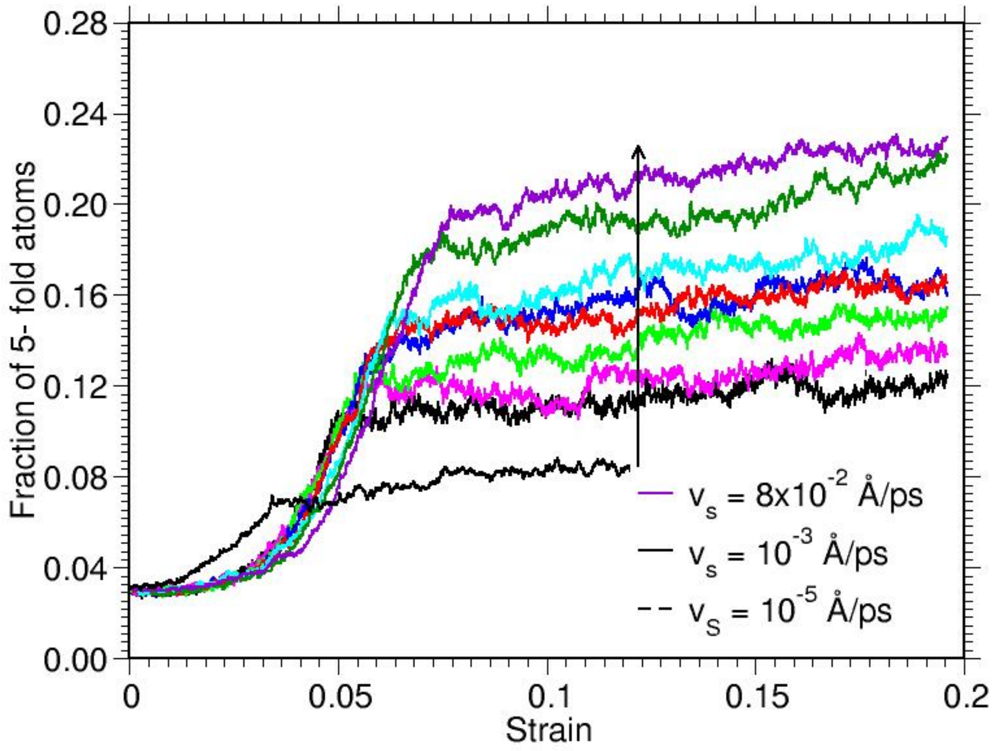
A. Kerrache et al. *PRB* ***83*** *(2011) 134122.*

Shear deformations at 300 K

Fraction of perfect 4- fold atoms



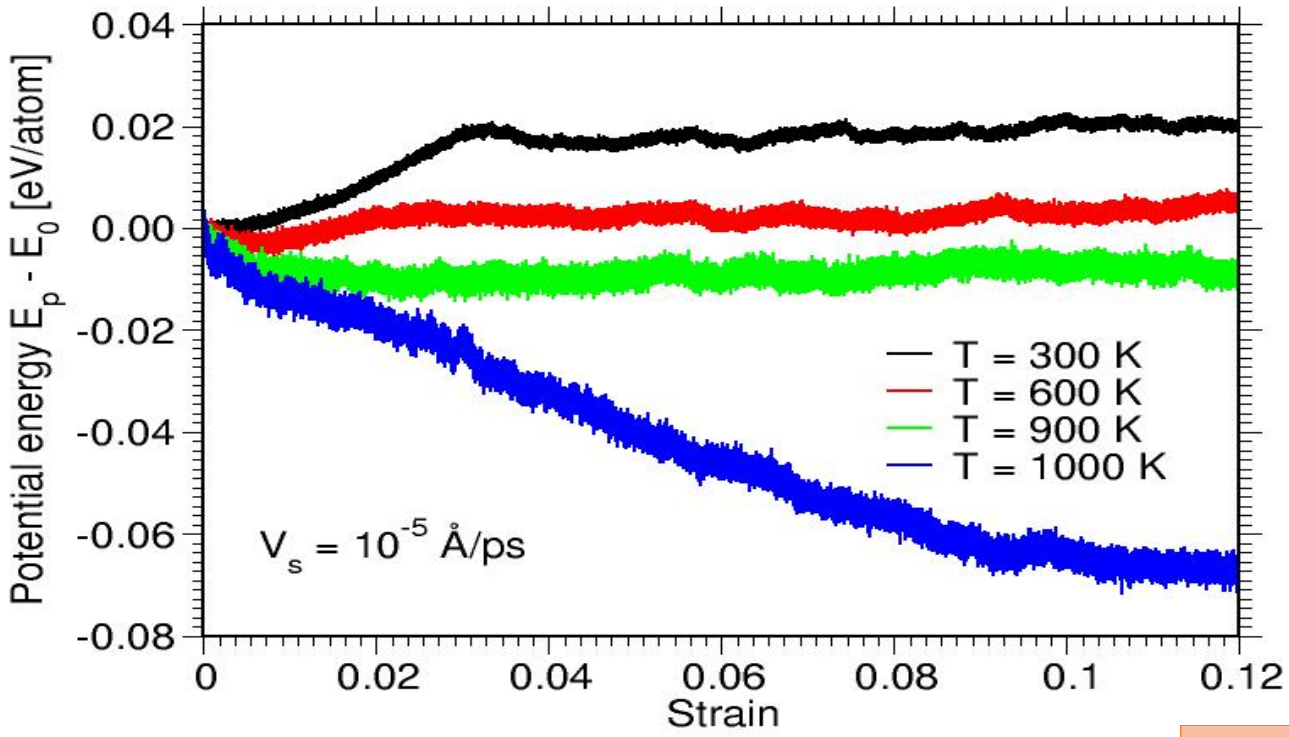
Fraction of 5- fold atoms



- ❑ Increasing shear velocity leads:
  - ➢ an increase of the fraction of 5-fold atoms
  - ➢ a decrease of the fraction of 4-fold atoms
- ❖ Increases the disorder and the defects

❑ What will happened if the shear deformations were applied at high temperature?

Crystallization induced by shear



- Temperatures: 300, 600, 900 and 1000 K
- Shear velocity: 10^{-5} Å/ps
- Total displacement: δ = 5 Å

Potential energy ΔE_P as a function of imposed strain

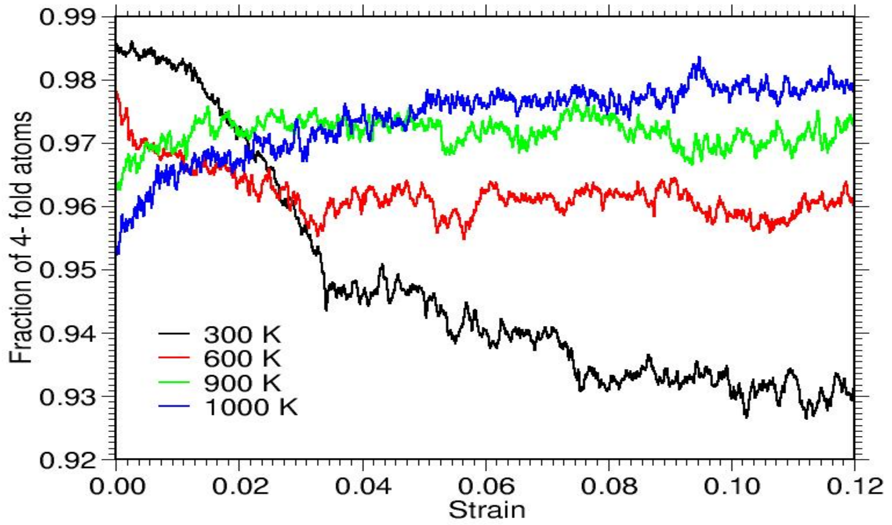
T	300	600	900	1000
ΔE_P	+0.02	+0.005	-0.01	-0.065

- Increase of the disorder at 300 and 600 K.
- At 900 K: increase of the order without crystallization.
- At 1000 K: shear induce the crystallization of *a-Si*.

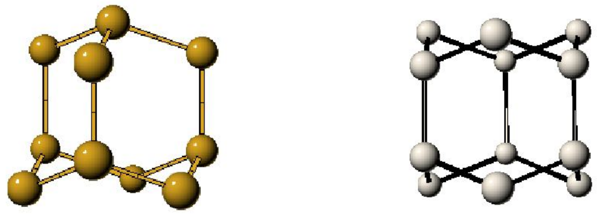
A. Kerrache et al. PRB 84 (2011) 041110.

Crystallization induced by shear

P. Beaucage et al. PRB (2005).

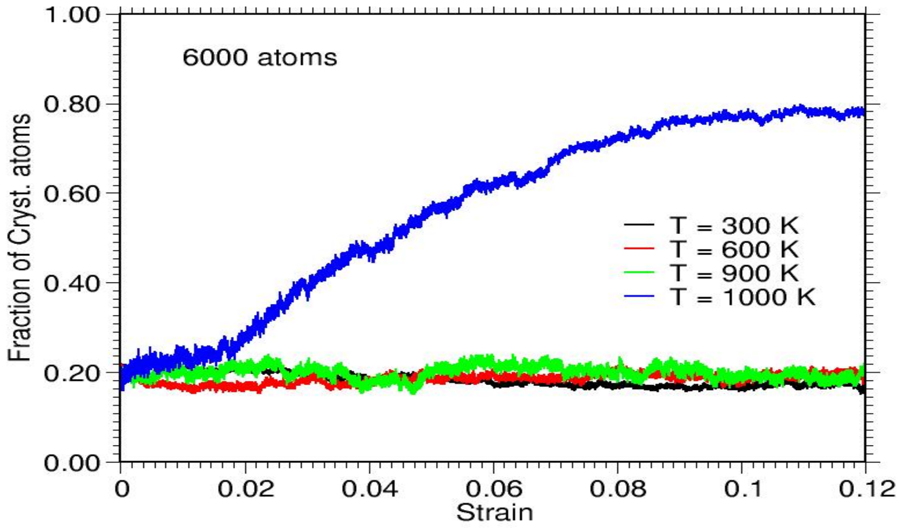


Coordination Number

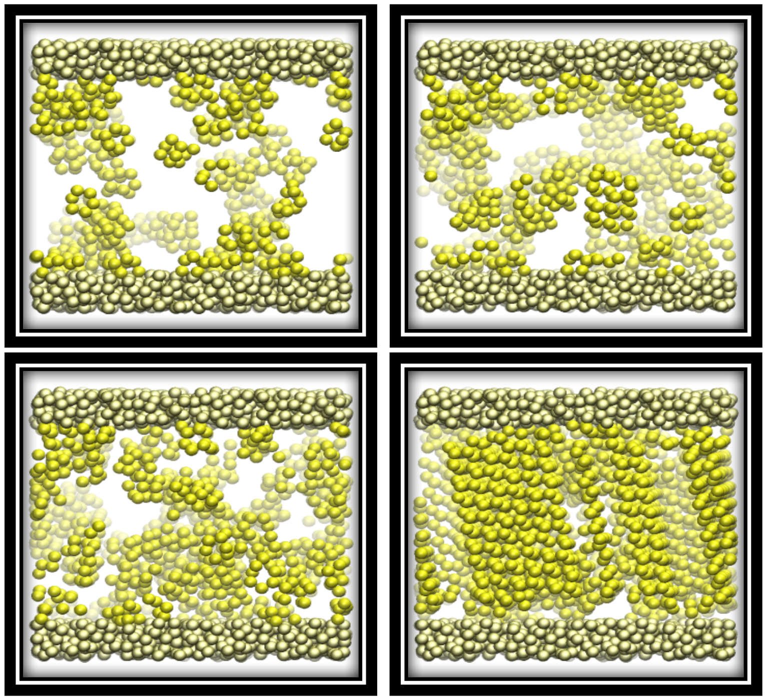


300 K

600 K



Number of crystalline particles



900 K

1000 K

Molecular Dynamics: some results

Binary Metallic alloys:

- Melting and crystallization.
- Solid-Liquid interfaces.
- Crystal growth from melt.
- Crystal growth is diffusion limited process.

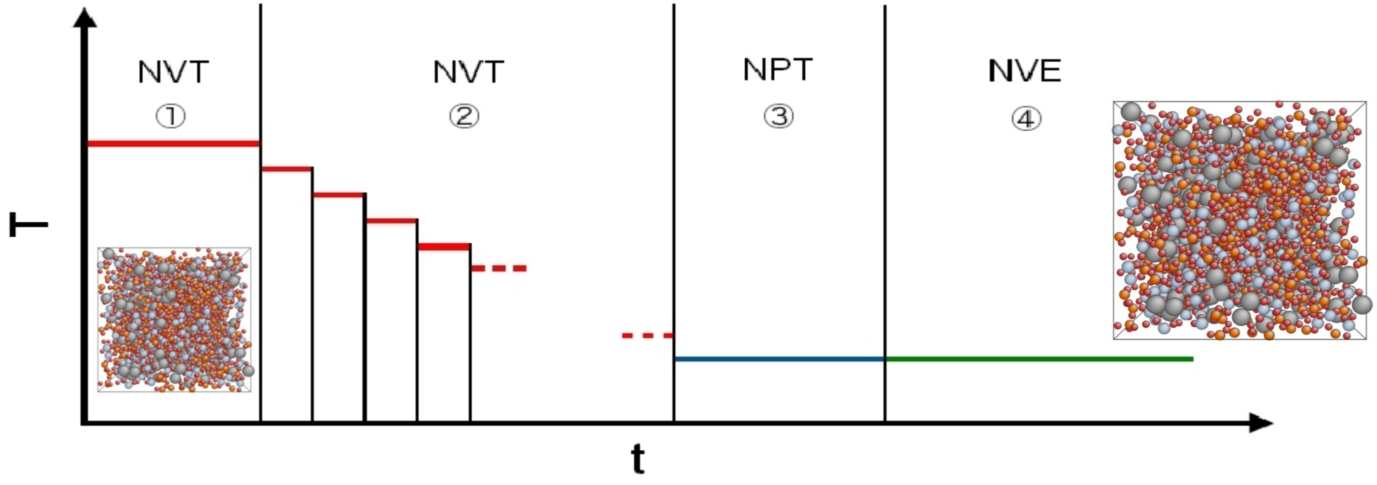
Glasses:

- Shear deformations in amorphous materials
- How to prepare a glass using MD simulation?
- Glass Indentation using MD.

How to prepare a glass? MD/experiment

Glass preparation diagram:

Cooling rates: 10^{12} to 10^{13} K/s



Glass preparation procedure:

- ✓ Random configuration (N atoms).
- ✓ Liquid equilibration du at 5000 K (NVT).
- ✓ Cooling per steps of 100 K– (NVT).
- ✓ Glass equilibration at 300 K (NPT).
- ✓ Trajectory simulation at 300 K (NVE).

Model

- MD Simulations (**DL-POLY**).
- System of N particules.
- Time step: 1 fs

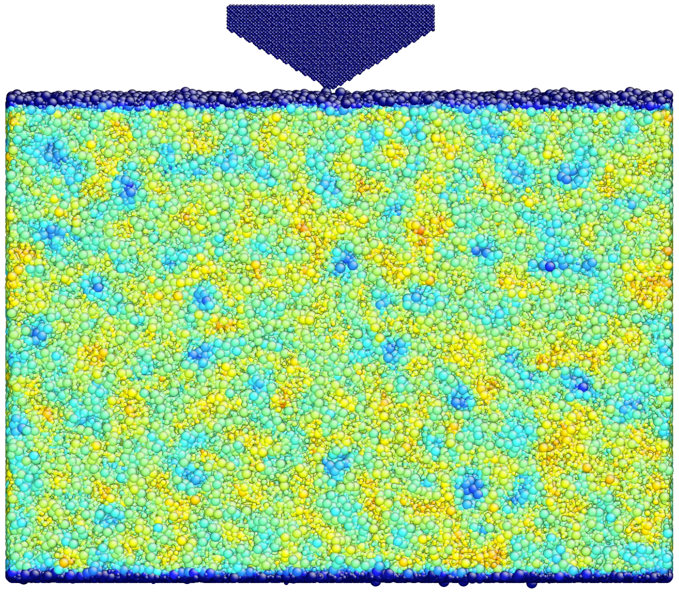
SBN glasses

- SiO_2- B_2O_3- Na_2O
  - ✓ R = $[Na_2O] / [B_2O_3]$
  - ✓ K = $[SiO_2] / [B_2O_3]$

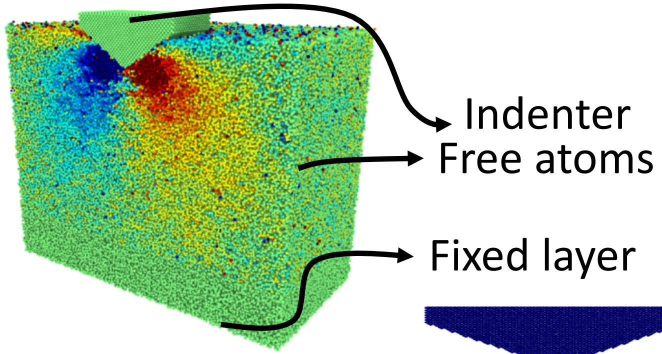
Glass indentation

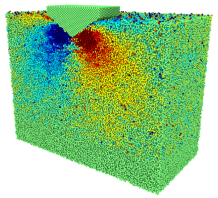
Movie provided by: Dimitrios Kilymis

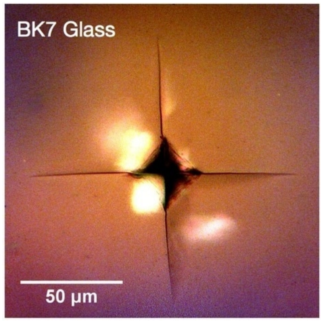
Laboratoire Charles Coulomb (L2C), UMR 5221 CNRS-Univ. Montpellier, France.

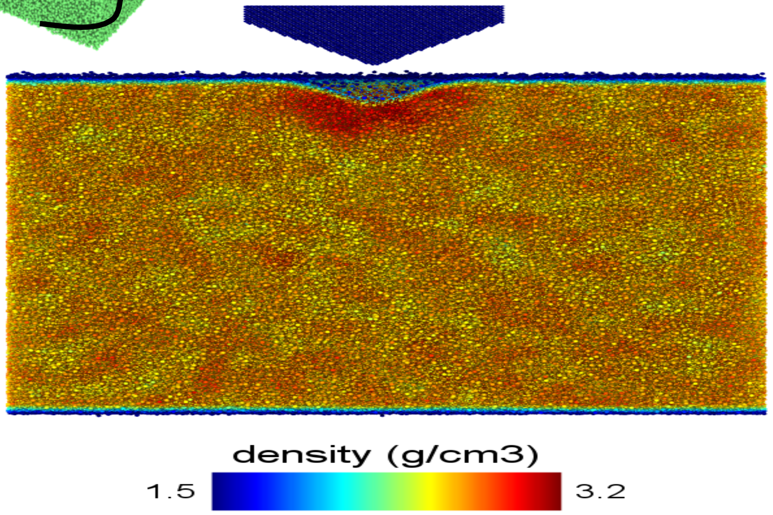


Glass indentation









- N = 2.1 x 10^6 atoms
- Temperature : 300 K
- Speed: 10 m/s
- Depth: ~3.0 nm

Acknowledgments

Prof. Dr. Jürgen Horbach, Dusseldorf, Germany.
Prof. Dr. Kurt Binder, Mainz, Germany.
Prof. A. Meyer and Prof. D. Herlach (DLR), Koln.

Prof. Normand Mousseau, Qc, Canada.
Prof. Laurent J. Lewis, Qc, Canada.

Dr. Dimitrios Kilymis, Montpellier, France.
Prof. Jean-Marc Delaye, CEA, France.

Dr. Victor Teboul, Angers, France.
Prof. Hamid Bouzar, UMMTO, Tizi-Ouzou, Algeria.

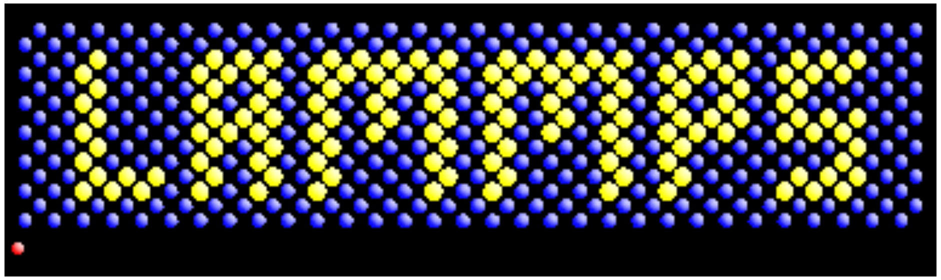
Setting and Running MD simulations

LAMMPS

- LAMMPS: Molecular Dynamics Simulator (introduction).
- Building LAMMPS step by step.
- Running LAMMPS (input, output, …).
- Benchmark and performance tests.

LAMMPS

Large-scale Atomic / Molecular Massively Parallel Simulator



Source: *some material and images were adapted from LAMMPS home page*

Where to start to learn LAMMPS?

Large-scale **A**tomic / **M**olecular **M**assively **P**arallel **S**imulator

S. Plimpton, A. Thompson, R. Shan, S. Moore, A. Kohlmeyer, …

Sandia National Labs: *http://www.sandia.gov/index.html*

- **Home Page: http://lammps.sandia.gov/**

Results:

- **Papers: http://lammps.sandia.gov/papers.html**
- **Pictures: http://lammps.sandia.gov/pictures.html**
- **Movies: http://lammps.sandia.gov/movies.html**

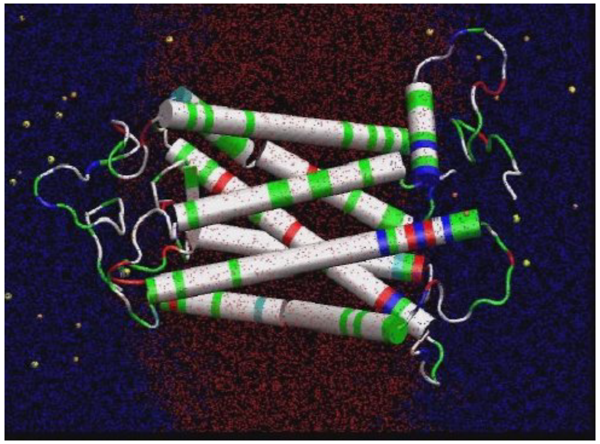
Resources:

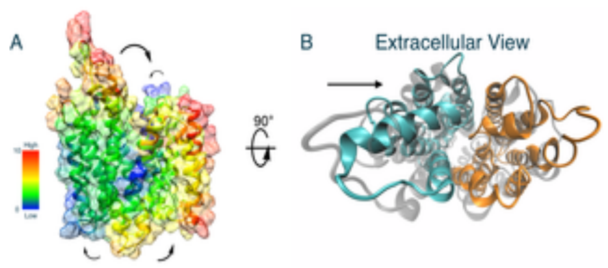
- **Online manual: http://lammps.sandia.gov/doc/Manual.html**
- **Search the mailing list: http://lammps.sandia.gov/mail.html**
- **Subscribe to the Mailing List:**

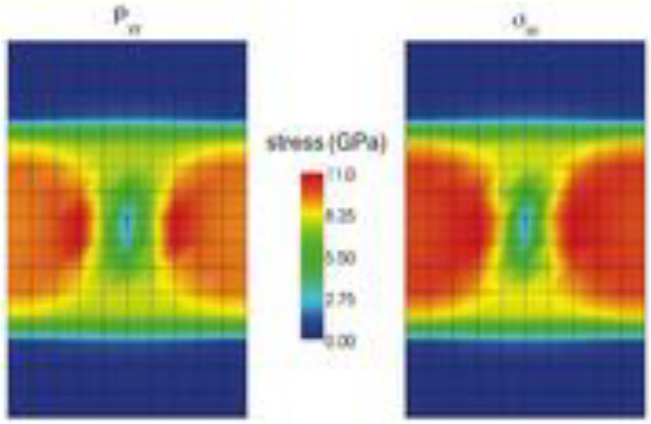
https://sourceforge.net/p/lammps/mailman/lammps-users/

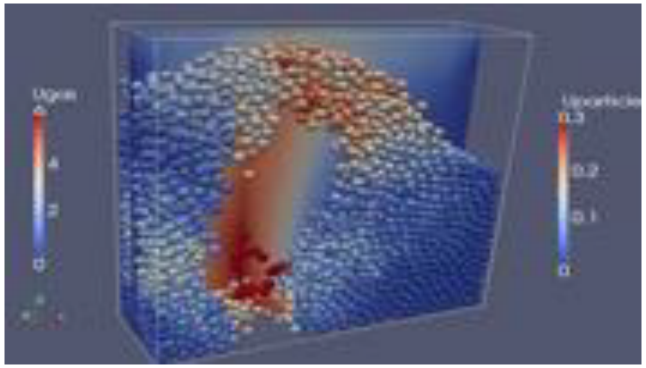
Where LAMMPS has been used?

- Biophysics



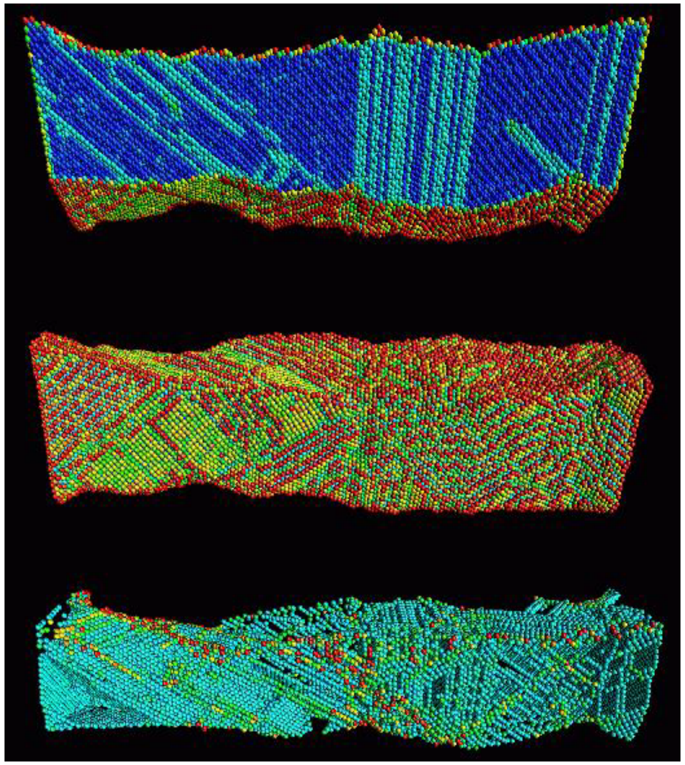






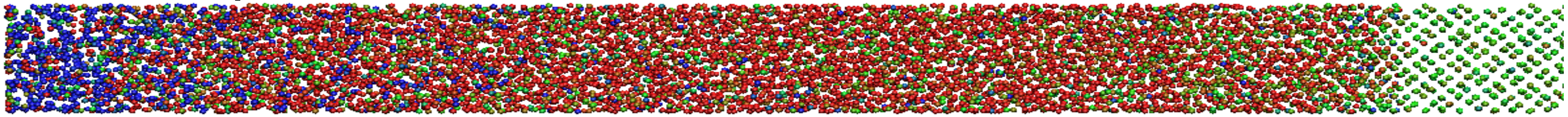
- Granular Flow

- Solid Mechanics



- Material Science

- Chemistry



LAMMPS project page

Big Picture	Code	Documentation	Results	Related Tools	Context	User Support
Features	Download	Manual	Publications	Pre/Post processing	Authors	Mail list
Non-features	SourceForge	Developer guide	Pictures	Pizza.py Toolkit	History	Workshops
FAQ	Latest features & bug fixes	Tutorials	Movies	Offsite LAMMPS packages & tools	Funding	User scripts and HowTos
Wish list	Unfixed bugs	MD to LAMMPS glossary	Benchmarks	Visualization	Open source	Contribute to LAMMPS

Recent LAMMPS News

- (5/18) New fix bond/react command to enable simulation of one or more complex heuristic reactions that rearrange molecular topology. See details here.
- (3/18) New stable release, 16Mar18 version.
- (9/17) Wrapper on the LATTE DFTB (density-functional tight-binding) quantum code via the fix latte command. See details here.
- (9/17) USER-MESO package from the Karniadakis group at Brown University, with various dissipative particle dynamics (DPD) models, including eDPD, mDPD, tDPD. See details here.
- (8/17) New stable release, 11Aug17 version.

How LAMMPS is designed?

❖ **License**

- LAMMPS is provided through **GNU Public License**
  https://www.gnu.org/licenses/licenses.en.html#GPL
- Free to Use, **Modify**, and Distribute.
- Contribute to LAMMPS: http://lammps.sandia.gov/contribute.html

❑ **Code Layout**

- C++ and Object-Oriented approach
- Parallelization via **MPI** and **OpenMP**; runs on **GPU**.
- is invoked by **commands** through **input scripts**.
- **p**ossibility to **customized output**.
- could be interfaced with other codes (python, ...): **library**.
- possibility to **contribute to LAMMPS**: potential, fixes, ...

How to obtain LAMMPS?

- **Download Page:**

  http://lammps.sandia.gov/download.html

- **Distributions:**
  - ✓ Download a tarball ← **Build LAMMPS from source**
  - ✓ Git checkout and update
  - ✓ SVN checkout and update
  - ✓ Pre-built Ubuntu executables
  - ✓ Pre-built binary RPMs for Fedora/RedHat/CentOS/openSUSE
  - ✓ Pre-built Gentoo executable
  - ✓ OS X with Homebrew
  - ✓ Windows installer package
  - ✓ Applying patches

Building LAMMPS

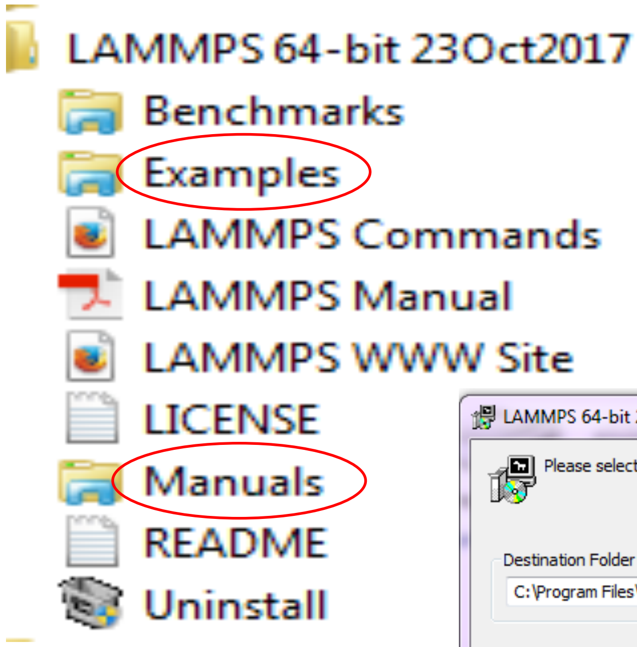
- Build from RPMs
  - ✓ Pre-built Ubuntu executables
  - ✓ Pre-built binary RPMs for Fedora/RedHat/CentOS/openSUSE
  - ✓ Pre-built Gentoo executable
- Mac
  - ✓ OS X with Homebrew
- Install under windows
  - ✓ Windows installer package
- Build from source code
  - ✓ Download a tarball
  - ✓ Git checkout and update
  - ✓ SVN checkout and update
  - ✓ Applying patches

does not include all packages
super-user access

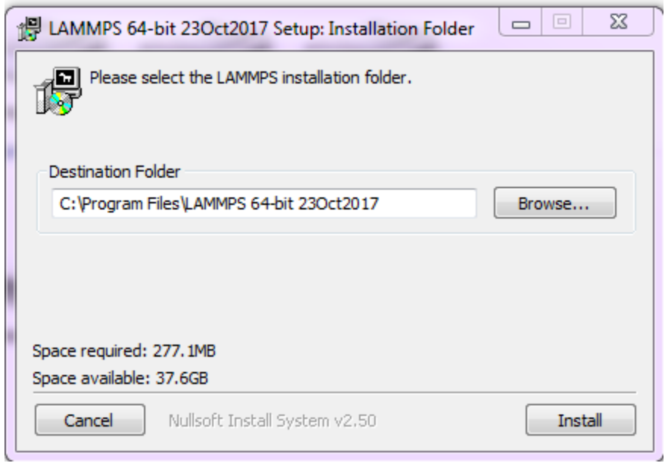
For a customized installation: build from source files
Different versions

Windows installer for LAMMPS

- **Download Page:** http://rpm.lammps.org/windows.html
- **Installer: lammps-64bit-latest.exe**


LAMMPS 64-bit 23Oct2017
- Benchmarks
- Examples
- LAMMPS Commands
- LAMMPS Manual
- LAMMPS WWW Site
- LICENSE
- Manuals
- README
- Uninstall

Directory:
Program Files\LAMMPS 64-bit 20171023
Binaries under bin: abf_integrate.exe ffmpeg.exe
lmp_mpi.exe restart2data.exe binary2txt.exe chain.exe
lmp_serial.exe msi2lmp.exe createatoms.exe

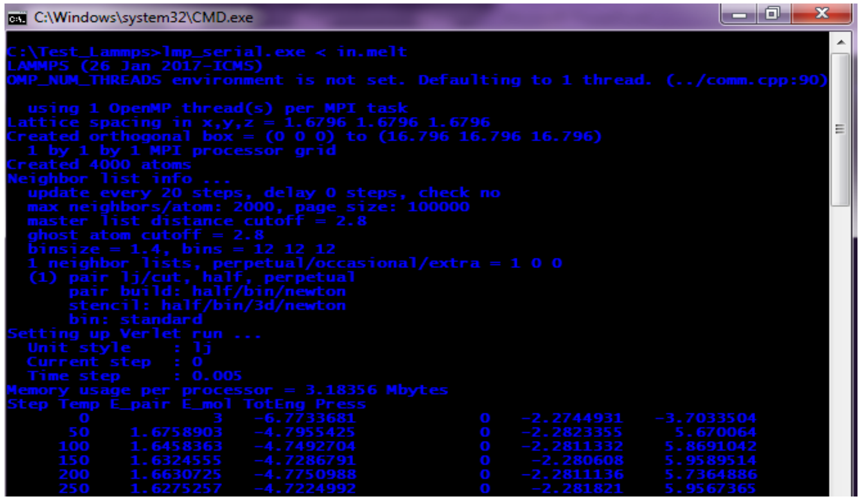

LAMMPS 64-bit 23Oct2017 Setup: Installation Folder

Please select the LAMMPS installation folder.

Destination Folder: C:\Program Files\LAMMPS 64-bit 23Oct2017

Space required: 277.1MB
Space available: 37.6GB

Nullsoft Install System v2.50


C:\Windows\system32\CMD.exe

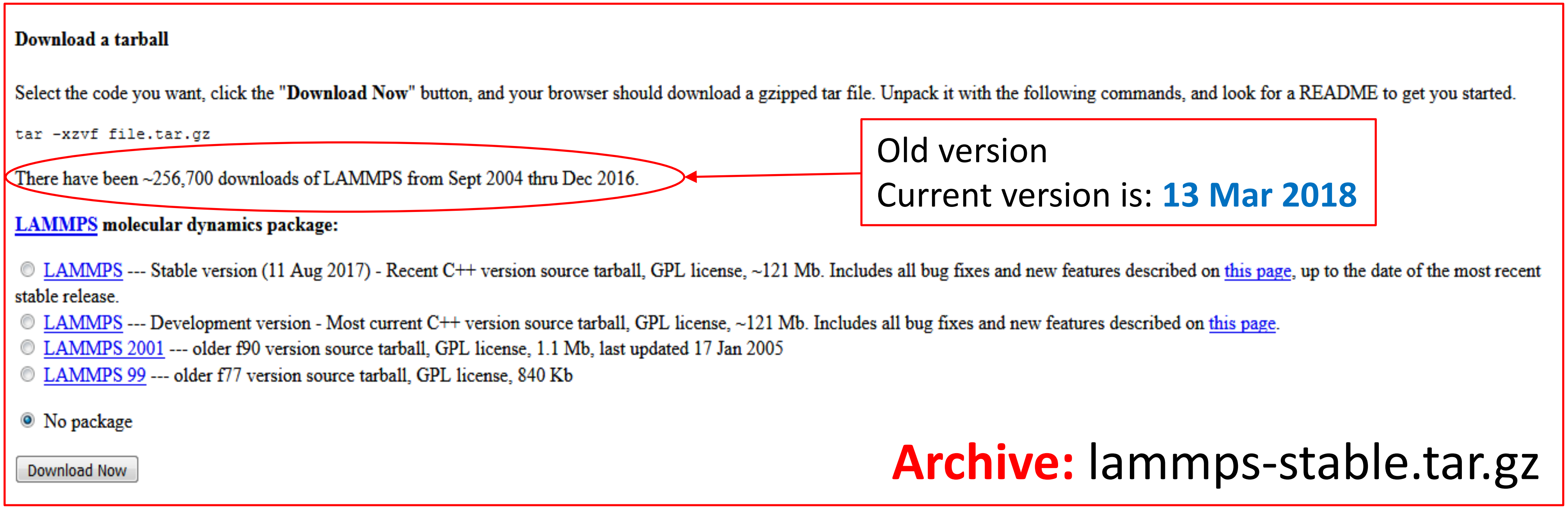
```
C:\Test_Lammps>lmp_serial.exe < in.melt
LAMMPS (26 Jan 2017-ICMS)
OMP_NUM_THREADS environment is not set. Defaulting to 1 thread. (../comm.cpp:90)

  using 1 OpenMP thread(s) per MPI task
Lattice spacing in x,y,z = 1.6796 1.6796 1.6796
Created orthogonal box = (0 0 0) to (16.796 16.796 16.796)
  1 by 1 by 1 MPI processor grid
Created 4000 atoms
Neighbor list info ...
  update every 20 steps, delay 0 steps, check no
  max neighbors/atom: 2000, page size: 100000
  master list distance cutoff = 2.8
  ghost atom cutoff = 2.8
  binsize = 1.4, bins = 12 12 12
  1 neighbor lists, perpetual/occasional/extra = 1 0 0
  (1) pair lj/cut, half, perpetual
      pair build: half/bin/newton
      stencil: half/bin/3d/newton
      bin: standard
Setting up Verlet run ...
  Unit style    : lj
  Current step  : 0
  Time step     : 0.005
Memory usage per processor = 3.18356 Mbytes
Step Temp E_pair E_mol TotEng Press
       0            3   -6.7733681            0   -2.2744931   -3.7033504
      50    1.6758903   -4.7955425            0   -2.2823355     5.670064
     100    1.6458363   -4.7492704            0   -2.2811332    5.8691042
     150    1.6324555   -4.7286791            0    -2.280608    5.9589514
     200    1.6630725   -4.7750988            0   -2.2811136    5.7364886
     250    1.6275257   -4.7224992            0    -2.281821    5.9567365
```

- **Execute: lmp_serial.exe < in.lammps**

Building LAMMPS from source

Download the source files:

http://lammps.sandia.gov/download.html#tar



Download a tarball

Select the code you want, click the "**Download Now**" button, and your browser should download a gzipped tar file. Unpack it with the following commands, and look for a README to get you started.

```
tar -xzvf file.tar.gz
```

There have been ~256,700 downloads of LAMMPS from Sept 2004 thru Dec 2016.

Old version
Current version is: **13 Mar 2018**

LAMMPS molecular dynamics package:

- ○ LAMMPS --- Stable version (11 Aug 2017) - Recent C++ version source tarball, GPL license, ~121 Mb. Includes all bug fixes and new features described on this page, up to the date of the most recent stable release.
- ○ LAMMPS --- Development version - Most current C++ version source tarball, GPL license, ~121 Mb. Includes all bug fixes and new features described on this page.
- ○ LAMMPS 2001 --- older f90 version source tarball, GPL license, 1.1 Mb, last updated 17 Jan 2005
- ○ LAMMPS 99 --- older f77 version source tarball, GPL license, 840 Kb

● No package

Archive: lammps-stable.tar.gz

Alternatively, use wget from your terminal:

- **wget http://lammps.sandia.gov/tars/lammps-stable.tar.gz**
- **wget http://lammps.sandia.gov/tars/lammps-11May18.tar.gz**

LAMMPS source overview

- **Download** and **unpack** the source code: lammps-stable.tar.gz
- LAMMPS directory: lammps-11May18 (lammps-version)
  - ✓ **bench:** benchmark tests (potential, input and output files).
  - ✓ **doc:** documentation (PDF and HTML)
  - ✓ **examples:** input and output files for some simulations
  - ✓ **lib:** libraries to build before building LAMMPS
  - ✓ **LICENSE** and **README** files.
  - ✓ **potentials:** some of the force fields supported by LAMMPS
  - ✓ **python:** to invoke LAMMPS library from Python
  - ✓ **src:** source files (*.cpp, PACKAGES, USER-PACKAGES, ...)
  - ✓ **tools:** some tools like xmovie (similar to VMD but only 2D).

Building programs

- **Common problems:**
  - command not found, ….
  - undefined reference to fftw, boost, petsc, …
  - permission denied, …

- **Configuration:**
  - ✓ ./configure --prefix=/home/$USER/software … options
  - ✓ cmake .. --DCMAKE_INSTALL_PREFIX=/home/$USER/software
  - ✓ ./setup or other provided scripts

- **Compile or build the program:**
  - make or make {options}

- **Installation:**
  - ✓ make install

Building LAMMPS using GNU make

- ❑ First: Build libraries if required (atc, meam, reax, …).
- ❑ Choose a Makefile: compatible with your system (Compiler, …)
- ❑ Choose and install the packages you need.
  - ✓ make package # list available packages
  - ✓ make package-status (ps) # status of all packages
  - ✓ make yes-package # install a single package in src
  - ✓ make no-package # remove a single package from src
  - ✓ make yes-all # install all packages in src
  - ✓ make no-all # remove all packages from src
  - ✓ make yes-standard (yes-std) # install all standard packages
  - ✓ make no-standard (no-std) # remove all standard packages
  - ✓ make yes-user # install all user packages
  - ✓ make no-user # remove all user packages
- ❑ Build LAMMPS:
  - ➢ make machine # build LAMMPS for machine

Choose an appropriate Makefile

- machine is one of these from src/MAKE:
  - # mpi = MPI with its default compiler
  - # serial = GNU g++ compiler, no MPI
- … or one of these from src/MAKE/OPTIONS:
  - # icc_openmpi = OpenMPI with compiler set to Intel icc
  - # icc_openmpi_link = Intel icc compiler, link to OpenMPI
  - # icc_serial = Intel icc compiler, no MPI
- … or one of these from src/MAKE/MACHINES:
  - # cygwin = Windows Cygwin, mpicxx, MPICH, FFTW
  - # mac = Apple PowerBook G4 laptop, c++, no MPI, FFTW 2.1.5
  - # mac_mpi = Apple laptop, MacPorts Open MPI 1.4.3, …
  - # ubuntu = Ubuntu Linux box, g++, openmpi, FFTW3
- … or one of these from src/MAKE/MINE: (write your own Makefile)

Building LAMMPS: demonstration

- Download the latest stable version from LAMMPS home page.
- Untar the archive: tar -xvf lammps-stable.tar.gz
- Change the directory and list the files: cd lammps-11May2018
  - **bench doc examples lib LICENSE potentials python README src tools**
- Choose a Makefile (for example: machine=icc_openmpi)
  - src/MAKE/OPTIONS/Makefile.icc_openmpi
- Load the required modules (Intel, OpenMPI, …)
- Check the packages: package, package-status, yes-package, …
- To build LAMMPS, run: **make icc_openmpi**
- Add or remove a package (if necessary), then recompile.
- If necessary, edit Makefile and fix the path to libraries.

- **Exercise: use an interactive job asking for 4 cores**
  - Compile LAMMPS without any package included.
  - Add a package and recompile.

Building LAMMPS: demonstration

- **Steps to follow:**

1. Download and unpack the source files:
   wget http://lammps.sandia.gov/tars/lammps-stable.tar.gz
   tar –xvf lammps-stable.tar.gz
2. Submit an interactive job: 4 cores, 1 hour, mem-per-cpu=3500M
3. Load the required modules (eigen, voro++, hdf5, …)
4. Choose and edit the appropriate Makefile: **Makefile.icc_openmpi**
5. Remove all the packages: **make no-all**
6. Compile LAMMPS: **make icc_openmpi**
7. Add one or two packages: make **yes-asphere**; **make yes-voro++**
8. Make the necessary changes: **paths**, **libraries**, … etc.
9. Clean and recompile the code: **make clean-all && make icc_openmpi**
10. To add more packages: repeat 7 to 9 (**with different packages**).

Building LAMMPS: demonstration

1. wget http://lammps.sandia.gov/tars/lammps-stable.tar.gz

   tar –xvf lammps-stable.tar.gz
2. cp -r lammps-16Mar18 build-lammps-16Mar18
3. cd build-lammps-16Mar18/src && make clean-all
4. source ../../get_lammps_dependencies.sh
5. make icc_openmpi
6. make clean-all
7. make yes-voro++
8. Change the Makefile: Makefile.icc_openmpi
9. Recompile: make icc_openmpi

Command line for running LAMMPS

- Executable: lmp_machine
- Files:
  - Input File: in.lmp_file
  - Potential: see examples and last slides for more details.
  - Initial configuration: can be generated by LAMMPS, or another program or home made program.
- Interactive Execution:

```
$ ./lmp_machine < in.lmp_file
$ ./lmp_machine –in in.lmp_file
```

- Redirect output to a file:

```
$ ./lmp_machine < in.lmp_file > output_file
$ ./lmp_machine –in in.lmp_file –l output_file
```

LAMMPS: modules

- To know more about the modules installed, use "module spider" .
- Search for modules with the name "lammps"

```
module spider lammps
```

- Search for all modules that have the character "lammps" in their names:

```
module -r spider '.*lammps.*'
```

- Search of a particular module of interest: lammps-omp/20170811

```
module spider lammps-omp/20170811
```

- Load the module of interest: lammps-omp/20170811

```
module load nixpkgs/16.09  intel/2016.4  openmpi/2.1.1 lammps-omp/20170811
```

- Check if the module is correctly loaded: `module list`
- For more information:

```
module show lammps-omp/20170811
```

Command line options

- **Command-line options:**

At run time, LAMMPS recognizes several optional command-line switches which may be used in any order.

-e or -echo, -h or –help, -i or –in, -k or –kokkos, -l or –log, -nc or –nocite, -pk or –package, -p or –partition, -pl or –plog, -ps or –pscreen, -r or –restart, -ro or –reorder, -sc or –screen, -sf or –suffix, -v or –var

- **As an interactive job:**
  - **mpirun -np 16 lmp_machine -in in.alloy**
  - **mpiexec -n 4 lmp_machine < in.alloy**
- **As a submitted job (Torque, SLURM, …):**
  - **mpiexec lmp_machine < in.alloy > my.log**
  - **mpirun lmp_machine < in.alloy > my.log**
  - **srun lmp_machine < in.alloy > my.log**

Simulation run overview

INPUT	RUNNING	OUTPUT
• Initial positions • Initial velocities • Time step • Mass • Charges • PBC (or fixed) • Units • Potential • Ensemble • .... etc.	• Molecular Dynamics Simulation (NPT, NVT, NVE) • Minimization • Monte Carlo • Atomic to Continuum	• Trajectories • Velocities • Forces • Energy • Temperature • Pressure • Density • Snapshots • Movies • ... etc.

Overview of a simulation settings

- **Command Line:**
  - Every simulation is executed by supplying an input text script to the LAMMPS executable: lmp < lammps.in > log_lammps.txt

- **Parts of an input script:**
  - **Initialize:** units, dimensions, PBC, etc.
  - Atomic positions (built in or read from a file) and velocities.
  - **Settings:**
    - ✓ Inter-atomic potential (pair_style, pair_coeff)
    - ✓ Run time simulation parameters (e.g. time step)
    - ✓ Fixes: operations during dynamics (e.g. thermostat)
    - ✓ Computes: calculation of properties during dynamics
    - ✓ Rendering: snapshots of MD trajectory, movie.

- **Run the simulation for N steps (time step = δt).**

LAMMPS input file example: LJ melt

```
# 3d Lennard-Jones melt

units            lj
atom_style     atomic

lattice          fcc 0.8442
region           box  block  0  10  0 10  0 10
create_box         1 box
create_atoms     1 box
mass           1 1.0

velocity       all  create  3.0   87287

# Potential

pair_style     lj/cut  2.5
pair_coeff     1 1  1.0  1.0  2.5
```

Comment

Define units

Create the simulation box

Or read data from a file

Initialize the velocities

Define the potential

LAMMPS input file example: LJ melt

```
# Neighbour list:
neighbor           0.3 bin
neigh_modify    every 20 delay 0 check no

#  set the thermodynamic ensemble:
fix                1 all nve

dump          id all atom 50 dump.melt
#dump_modify ….

log                log.melt
thermo_style  custom  step temp etotal ….
thermo           50

run                250

# End of the simulation.
```

Monitor the neighbour list

Thermodynamic Ensemble

Store the trajectory

Log file: customize output

Run the simulation for N steps

LAMMPS: input commands

❑ **Initialization**

➢ **Parameters:** set parameters that need to be defined before atoms are created: ***units***, *dimension*, *newton*, *processors*, ***boundary***, ***atom_style***, *atom_modify*.

➢ If force-field parameters appear in the files that will be read: ***pair_style***, *bond_style*, *angle_style*, *dihedral_style*, *improper_style*.

➢ **Atom definition:** there are 3 ways to define atoms in LAMMPS.

✓ Read them in from a data or restart file via the ***read_data*** or ***read_restart*** commands.

✓ Or create atoms on a lattice (with no molecular topology), using these commands: *lattice*, *region*, *create_box*, *create_atoms*.

✓ Duplicate the box to make a larger one the ***replicate*** command.

LAMMPS: settings

- Once atoms are defined, a variety of settings need to be specified:
  force field coefficients, simulation parameters, output options ...

  - **Force field coefficients:**
    pair_coeff, ***bond_coeff***, ***angle_coeff***, ***dihedral_coeff***, ***improper_coeff***, ***kspace_style***, ***dielectric***, ***special_bonds***.

  - **Various simulation parameters:**
    neighbor, ***neigh_modify***, ***group***, ***timestep***, ***reset_timestep***, ***run_style***, ***min_style***, ***min_modify***.

  - **Fixes:** ***nvt***, ***npt***, ***nve***, ...

  - **Computations during a simulation:**
    compute, ***compute_modify***, and ***variable*** commands.

  - **Output options:** ***thermo***, ***dump***, and ***restart*** commands.

Customize output

thermo **freq_steps**
thermo_style **style args**

Thermodynamic properties

- **style =** *one* or *multi* or *custom*
- **args =** list of arguments for a particular style

  one args = none
  multi args = none *custom*
  args = list of keywords possible

❑ **keywords = step, elapsed, elaplong, dt, time, cpu, tpcpu, spcpu, cpuremain, part, timeremain, atoms, temp, press, pe, ke, etotal, enthalpy, evdwl, ecoul, epair, ebond, eangle, edihed, eimp, emol, elong, etail, vol, density, lx, ly, lz, xlo, xhi, ylo, yhi, zlo, zhi, xy, xz, yz, xlat, ylat, zlat, bonds, angles, dihedrals, impropers, pxx, pyy, pzz, pxy, pxz, pyz ..... Etc**

❑ **Example:**

thermo_style custom step temp press pe ke etotal density lx ly lz

Customize output

- **dump command:**
  Options: vtk, h5md, molfile, netcdf, image, movie
- **Syntax:**
  dump ID group-ID style N file args

Trajectories
Snapshots Movie

- D = user-assigned name for the dump
- group-ID = ID of the group of atoms to be dumped
- style = *atom* or *atom/gz* or *atom/mpiio* or *cfg* or *cfg/gz* or *cfg/mpiio* or *custom* or *custom/gz* or *custom/mpiio* or *dcd* or *h5md* or *image* or or *local* or *molfile* or *movie* or *netcdf* or *netcdf/mpiio* or *vtk* or *xtc* or *xyz* or *xyz/gz* or *xyz/mpiio*
- N = dump every this many time steps
- file = name of file to write dump info to
- args = list of arguments for a particular style

- Example:
- dump myDump all atom 100 dump.atom
- dump 2 subgroup atom 50 dump.run.bin

Running LAMMPS: demonstration

- After compiling LAMMPS, run some examples:

- Where to start to learn LAMMPS?
  - Make a copy of the directory examples to your working directory.
  - Choose and example to run.
  - Indicate the right path to the executable or use modules available (if any).
  - Edit the input file and check all the parameters.
  - Check the documentation for the commands and their arguments.
  - Run the test case: lmp_icc_openmpi < in.melt
  - Check the output files (log files), plot the thermodynamic properties, ...
  - Use VMD (or any other software) for visualization.

Running LAMMPS: demonstration

- Connect to cedar and/or graham

  ssh –Y user@cedar.computecanada.ca

  ssh –Y user@graham.computecanada.ca

- Go to the directory where you copied or download the exercises.
- To run LAMMPS interactively, submit an inteactive job using salloc

  salloc –ntasks=1 –mem-per-cpu=2500M –time=00-00:30

- Submit some jobs using sbatch: sbatch your_script.sh
- Edit the input files
- Run the jobs
- Check the output files.

Output example: LJ melt

```
LAMMPS (30 Jul 2016)
using 1 2 OpenMP thread(s) per MPI task
# 3d Lennard-Jones melt

units           ljatom_style      atomic
lattice         fcc 0.8442Lattice spacing in x,y,z = 1.6796 1.6796 1.6796
region          box block 0 10 0 10 0 10
create_box    1  box

Created orthogonal box = (0 0 0) to (16.796 16.796 16.796)  2 by 2 by 3 MPI
processor grid

create_atoms   1 box
Created 4000 atoms

mass              1 1.0
```

Output example: LJ melt

```
thermo          100
run             25000
Neighbor list info ...
    1 neighbor list requests
    update every 20 steps, delay 0 steps, check no
    max neighbors/atom: 2000, page size: 100000
    master list distance cutoff = 2.8
    ghost atom cutoff = 2.8
    binsize = 1.4 -> bins = 12 12 12
    Memory usage per processor = 2.05293 Mbytes
Step Temp E_pair E_mol TotEng Press
 0     3           -6.7733681    0    -2.2744931   -3.7033504
 100   1.6510577   -4.7567887    0    -2.2808214    5.8208747
 200   1.6393075   -4.7404901    0    -2.2821436    5.9139187
 300   1.6626896   -4.7751761    0    -2.2817652    5.756386
```

Output example: LJ melt

```
25000    1.552843    -4.7611011         0    -2.432419      5.7187477

Loop time of 15.4965 on 12 procs for 25000 steps with 4000 atoms
Performance: 696931.853 tau/day, 1613.268 timesteps/s
90.2% CPU use with 12 MPI tasks x 1 OpenMP threads

MPI task timing breakdown:
Section  | min time | avg time | max time |%varavg| %total
---------------------------------------------------------------
Pair     |6.6964    |7.1974    |7.9599    | 14.8  | 46.45
Neigh    | 0.94857  |1.0047    |1.0788    |  4.3  |  6.48
Comm     |6.0595    |6.8957    |7.4611    | 17.1  | 44.50
Output   |0.01517   |0.01589   |0.019863  |  1.0  |  0.10
Modify   |0.14023   |0.14968   |0.16127   |  1.7  |  0.97
Other    |          |0.2332    |          |       |  1.50

Total wall time: 0:00:15
```

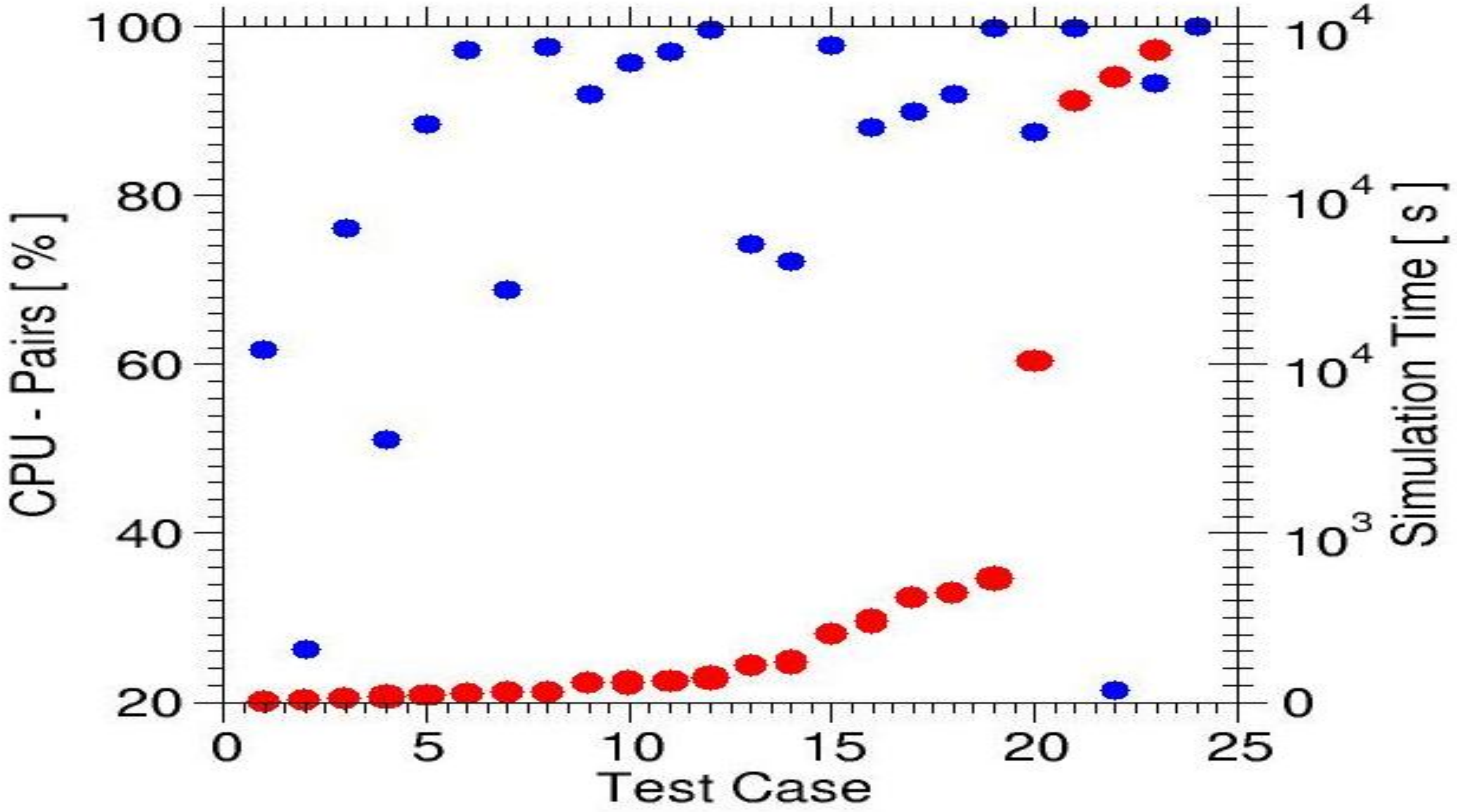
Benchmarking: potential

1. granular
2. fene
3. lj
4. dpd
5. eam
6. sw
7. rebo
8. tersoff
9. eim
10. adp
11. meam
12. peri
13. spce
14. protein
15. gb
16. reax_AB
17. airebo
18. reaxc_rdx
19. smtbq_Al
20. vashishta_table_sio2
21. eff
22. comb
23. vashishta_sio2
24. smtbq_Al2O3

Parameters:

- 24 different cases.
- Number of particles: about 32000
- CPUs = 1
- MD steps = 1000
- Record the simulation time and the time used in computing the interactions between particles.

Benchmarking: potential



Performance test: LJ

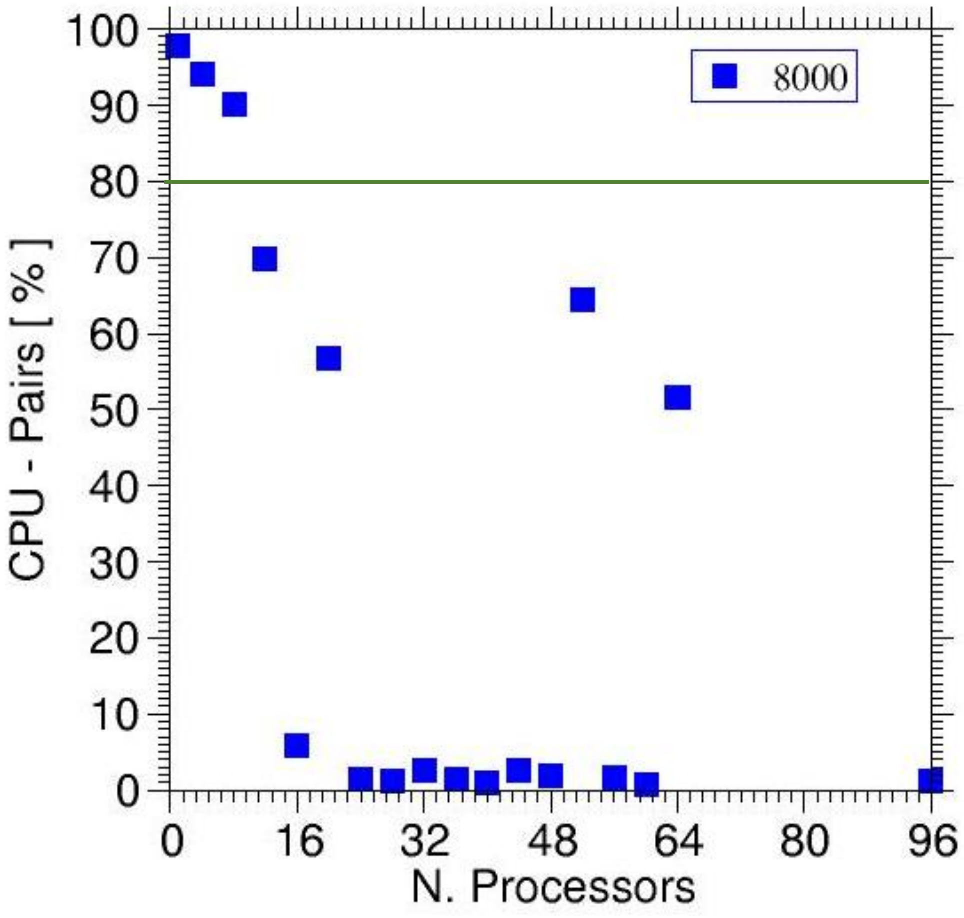
- ❑ Directory: Benchmark
- ❑ Simulation: LJ Melt
- ❑ Number of atoms: 2048, 4000, 6912, 13500
- ❑ Exercise:
  - ➢ Submit the jobs using different number of cores: 1,2, 4, 8, 16
  - ➢ For each system: collect the data:
  - ➢ time used for pair interactions as a function of number of cores.
  - ➢ time used for communications as a function of number of cores.

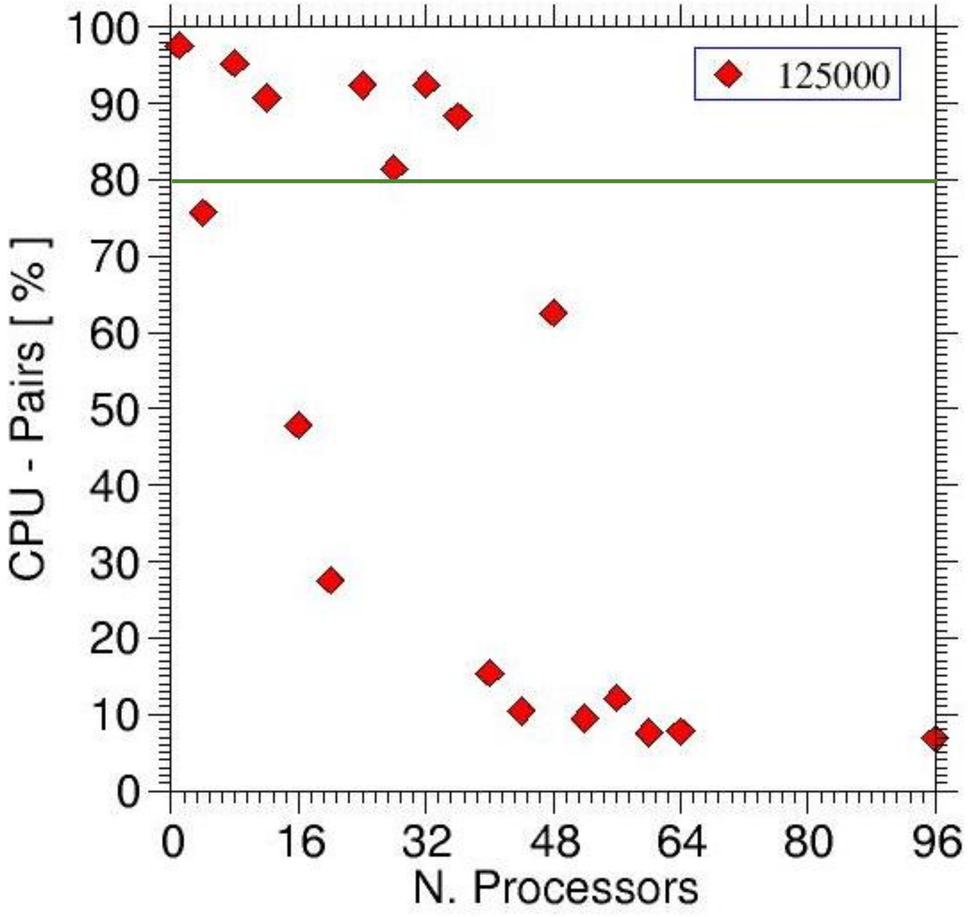
Performance test: LJ

Time spend for pair interactions computing and communications as a function of number of cores for different systems

Cores	Pairs	Comm	Pairs	Comm	Pairs	Comm	Pairs	Comm
1	73.68	1.36	73.70	1.28	73.66	1.27	73.72	1.29
2	70.35	5.19	70.77	4.68	70.51	5.11	67.80	8.77
4	62.77	13.98	64.93	12.19	67.52	8.99	67.74	8.71
8	58.36	20.14	61.78	15.58	64.10	12.86	62.06	8.71
16	56.69	20.18	56.70	20.18	56.97	19.80	56.41	20.38
	2048		4000		6912		13500	

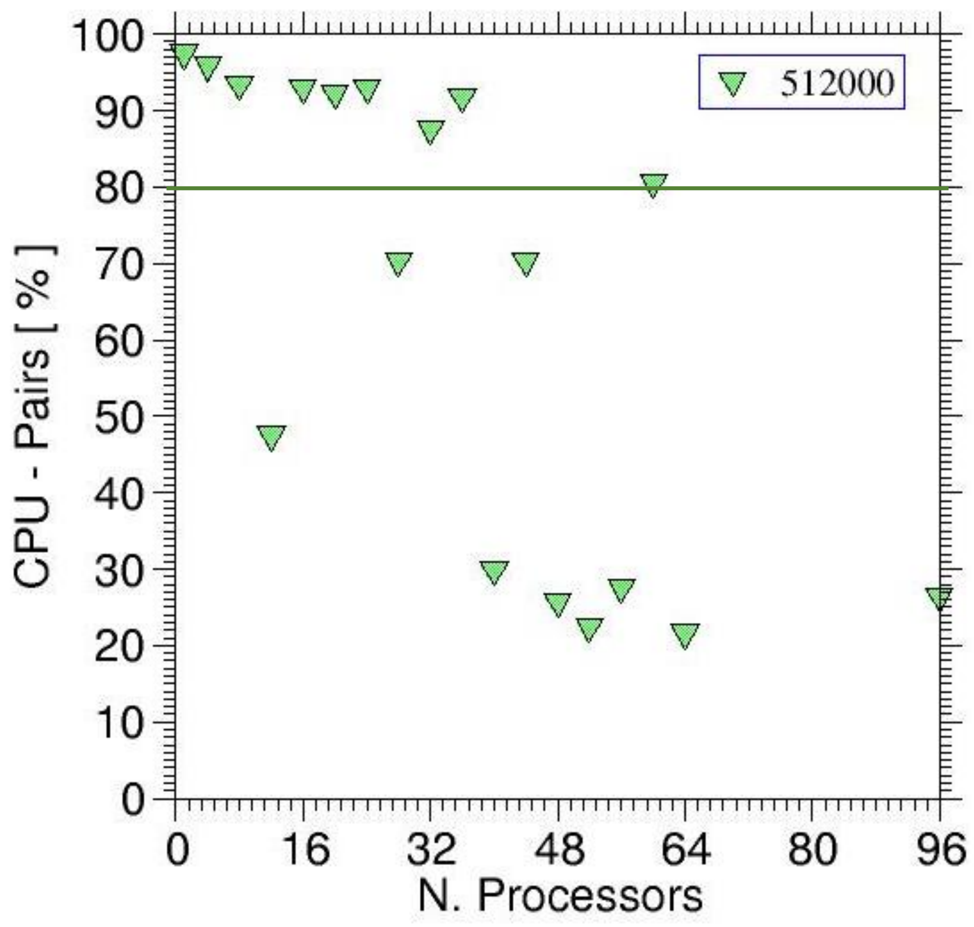
Performance test: Tersoff potential

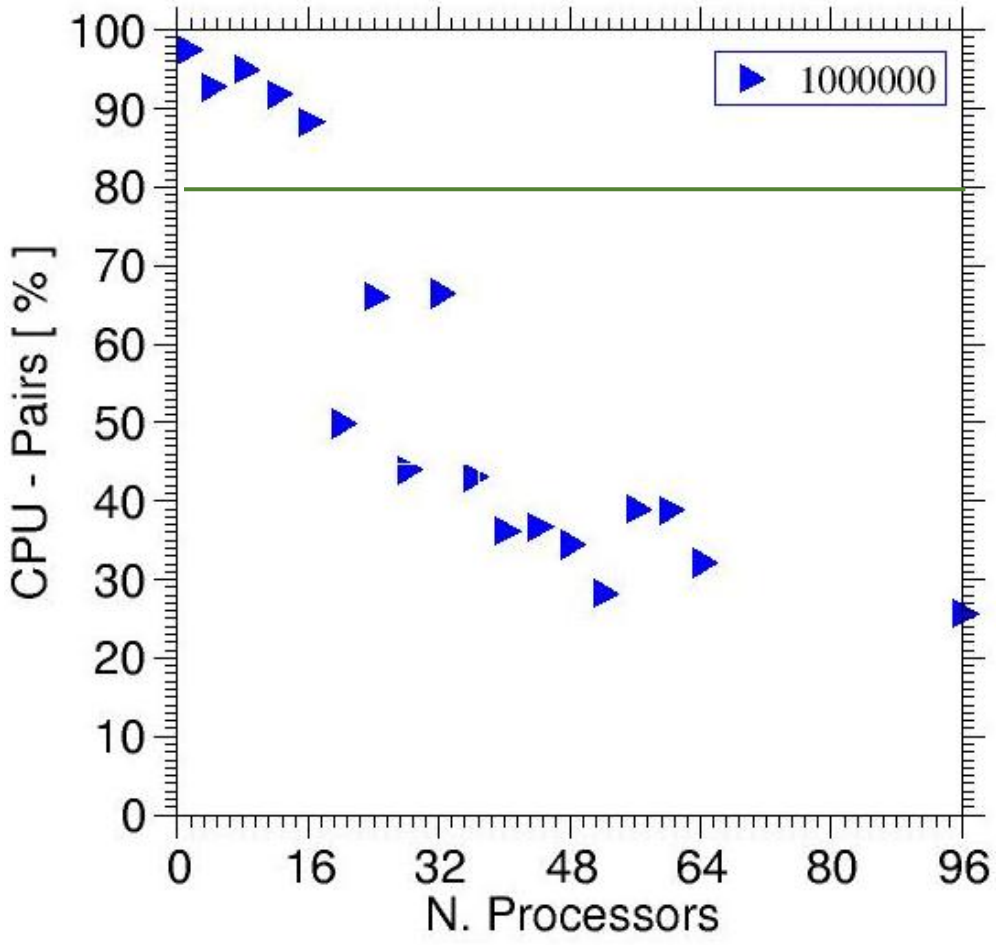




CPU time used for computing the interactions between particles as a function the number of processors for different system size.

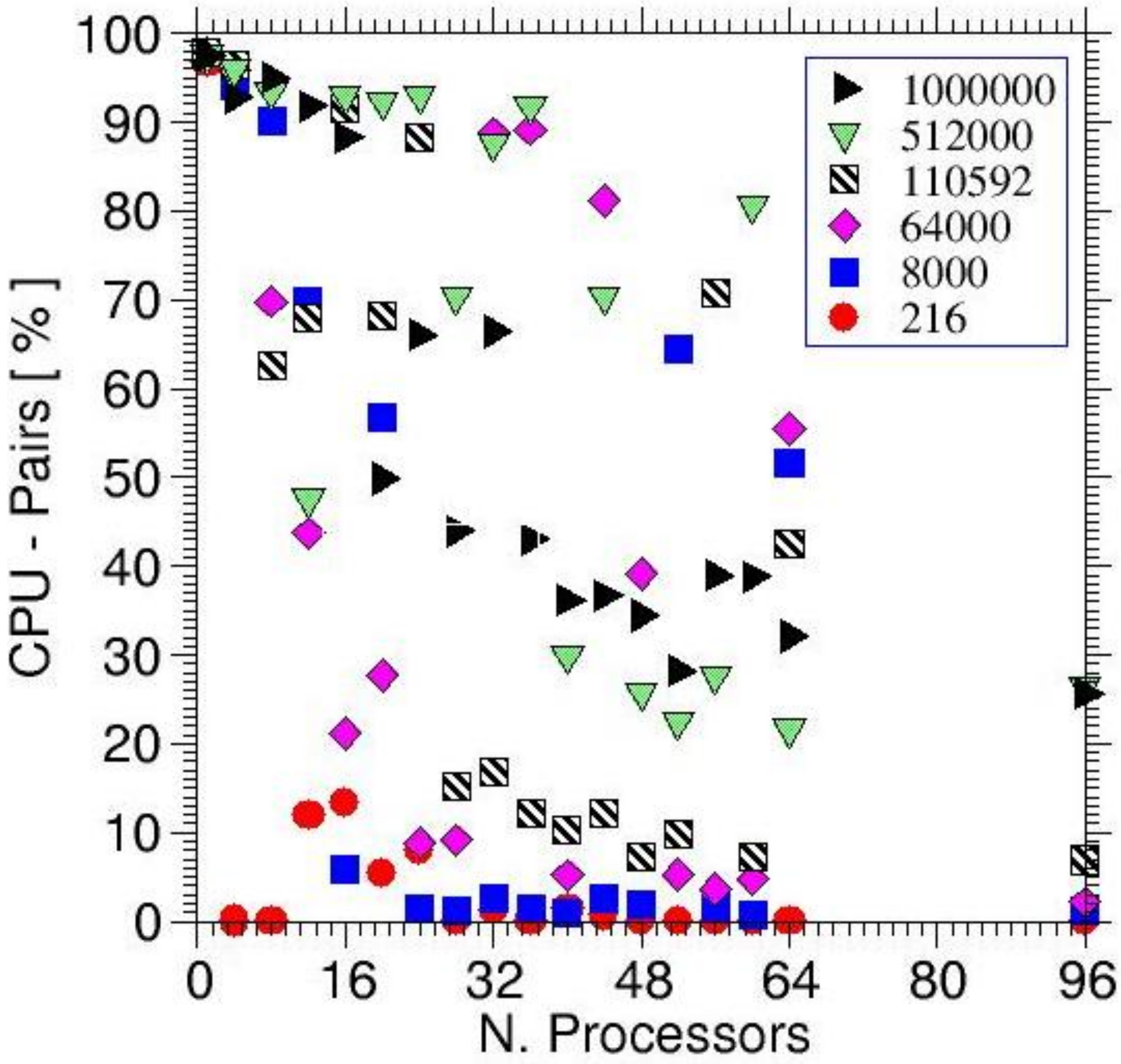
Performance test: Tersoff potential



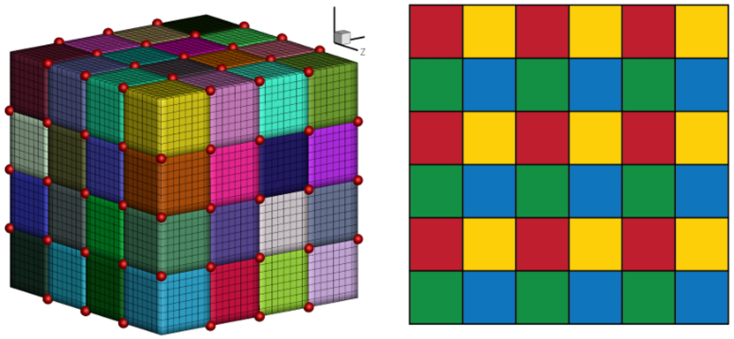


CPU time used for computing the interactions between particles as a function the number of processors for different system size.

Performance test: Tersoff potential



❑ Domain decomposition



- Size, shape of the system.
- Number of processors.
- size of the small units.
- correlation between the communications and the number of small units.
- Reduce the number of cells to reduce communications.

Learn more about LAMMPS

- **Home Page:** http://lammps.sandia.gov/
- **Examples:** deposit, friction, micelle, obstacle, qeq, streitz, MC, body, dipole, hugoniostat, min, peptide, reax, tad, DIFFUSE, colloid, indent, msst, peri, rigid , vashishta, ELASTIC, USER, comb, eim, nb3b, pour, shear, voronoi, ELASTIC_T, VISCOSITY, coreshell, ellipse, meam, neb, prd, snap, HEAT, accelerate, crack, flow, melt, nemd
- **Results:**
  - Papers: http://lammps.sandia.gov/papers.html
  - Pictures: http://lammps.sandia.gov/pictures.html
  - Movies: http://lammps.sandia.gov/movies.html
- **Resources:**
  - **Online Manual:** http://lammps.sandia.gov/doc/Manual.html
  - **Search the mailing list:** http://lammps.sandia.gov/mail.html
  - **Mailing List:**

    https://sourceforge.net/p/lammps/mailman/lammps-users/

Potentials: classified by materials

- **Bio-molecules:** CHARMM, AMBER, OPLS, COMPASS (class 2), long-range Coulombic via PPPM, point dipoles, ...
- **Polymers:** all-atom, united-atom, coarse-grain (bead-spring FENE), bond-breaking, ...
- **Materials:** EAM and MEAM for metals, Buckingham, Morse, Yukawa, Stillinger-Weber, Tersoff, EDIP, COMB, SNAP, ...
- **Chemistry:** AI-REBO, REBO, ReaxFF, eFF
- **Meso-scale:** granular, DPD, Gay-Berne, colloidal, peri-dynamics, DSMC...
- **Hybrid:** combine potentials for hybrid systems: water on metal, polymers/semiconductor interface, colloids in solution, ...

Potentials: classified by functional form

- **Pair-wise potentials:** Lennard-Jones, Buckingham, …
- **Charged Pair-wise Potentials:** Coulombic, point-dipole
- **Many-body Potentials:** EAM, Finnis/Sinclair, modified EAM (MEAM), embedded ion (EIM), Stillinger-Weber, Tersoff, AI-REBO, ReaxFF, COMB
- **Coarse-Grained Potentials:** DPD, GayBerne, …
- **Meso-scopic Potentials:** granular, peri-dynamics
- **Long-Range Electrostatics:** Ewald, PPPM, MSM
- **Implicit Solvent Potentials:** hydrodynamic lubrication, Debye
- **Force-Field Compatibility with common:** CHARMM, AMBER, OPLS, GROMACS options

Force fields: examples

pair_style none - turn off pairwise interactions
pair_style hybrid - multiple styles of pairwise interactions
pair_style hybrid/overlay - multiple styles of superposed pairwise interactions
pair_style zero - neighbor list but no interactions
pair_style adp - angular dependent potential (ADP) of Mishin
pair_style airebo - AIREBO potential of Stuart
pair_style airebo/morse - AIREBO with Morse instead of LJ
pair_style beck - Beck potential
pair_style body - interactions between body particles
pair_style bop - BOP potential of Pettifor
pair_style born - Born-Mayer-Huggins potential
pair_style born/coul/long - Born-Mayer-Huggins with long-range Coulombics
pair_style born/coul/long/cs - Born-Mayer-Huggins with long-range Coulombics and core/shell
pair_style born/coul/msm - Born-Mayer-Huggins with long-range MSM Coulombics

Force fields: examples

pair_style born/coul/wolf - Born-Mayer-Huggins with Coulombics via Wolf potential

pair_style brownian - Brownian potential for Fast Lubrication Dynamics

pair_style brownian/poly - Brownian potential for Fast Lubrication Dynamics with polydispersity

pair_style buck - Buckingham potential

pair_style buck/coul/cut - Buckingham with cutoff Coulomb

pair_style buck/coul/long - Buckingham with long-range Coulombics

pair_style buck/coul/long/cs - Buckingham with long-range Coulombics and core/shell

pair_style buck/coul/msm - Buckingham long-range MSM Coulombics

pair_style buck/long/coul/long - long-range Buckingham with long-range Coulombics

pair_style colloid - integrated colloidal potential

pair_style comb - charge-optimized many-body (COMB) potential

pair_style comb3 - charge-optimized many-body (COMB3) potential

Force fields: examples

pair_style coul/cut - cutoff Coulombic potential
pair_style coul/debye - cutoff Coulombic potential with Debye screening
pair_style coul/dsf - Coulombics via damped shifted forces
pair_style coul/long - long-range Coulombic potential
pair_style coul/long/cs - long-range Coulombic potential and core/shell
pair_style coul/msm - long-range MSM Coulombics
pair_style coul/streitz - Coulombics via Streitz/Mintmire Slater orbitals
pair_style coul/wolf - Coulombics via Wolf potential
pair_style dpd - dissipative particle dynamics (DPD)
pair_style dpd/tstat - DPD thermostatting
pair_style dsmc - Direct Simulation Monte Carlo (DSMC)
pair_style eam - embedded atom method (EAM)
pair_style eam/alloy - alloy EAM
pair_style eam/fs - Finnis-Sinclair EAM
pair_style eim - embedded ion method (EIM)
pair_style gauss - Gaussian potential

Force fields: examples

pair_style gayberne - Gay-Berne ellipsoidal potential
pair_style gran/hertz/history - granular potential with Hertzian interactions
pair_style gran/hooke - granular potential with history effects
pair_style gran/hooke/history - granular potential without history effects
pair_style hbond/dreiding/lj - DREIDING hydrogen bonding LJ potential
pair_style hbond/dreiding/morse - DREIDING hydrogen bonding Morse potential
pair_style kim - interface to potentials provided by KIM project
pair_style lcbop - long-range bond-order potential (LCBOP)
pair_style line/lj - LJ potential between line segments
pair_style lj/charmm/coul/charmm - CHARMM potential with cutoff Coulomb
pair_style lj/charmm/coul/charmm/implicit - CHARMM for implicit solvent
pair_style lj/charmm/coul/long - CHARMM with long-range Coulomb
pair_style lj/charmm/coul/msm - CHARMM with long-range MSM Coulombics
pair_style lj/class2 - COMPASS (class 2) force field with no Coulomb
pair_style lj/class2/coul/cut - COMPASS with cutoff Coulomb

Force fields: examples

pair_style lj/gromacs/coul/gromacs - GROMACS-style LJ and Coulombic potential
pair_style lj/long/coul/long - long-range LJ and long-range Coulombics
pair_style lj/long/dipole/long - long-range LJ and long-range point dipoles
pair_style lj/long/tip4p/long - long-range LJ and long-range Coulomb for TIP4P water
pair_style lj/smooth - smoothed Lennard-Jones potential
pair_style lj/smooth/linear - linear smoothed Lennard-Jones potential
pair_style lj96/cut - Lennard-Jones 9/6 potential
pair_style lubricate - hydrodynamic lubrication forces
pair_style lubricate/poly - hydrodynamic lubrication forces with polydispersity
pair_style lubricateU - hydrodynamic lubrication forces for Fast Lubrication Dynamics
pair_style lubricateU/poly - hydrodynamic lubrication forces for Fast Lubrication with polydispersity
pair_style meam - modified embedded atom method (MEAM)

Force fields: examples

pair_style mie/cut - Mie potential
pair_style morse - Morse potential
pair_style nb3b/harmonic - nonbonded 3-body harmonic potential
pair_style nm/cut - N-M potential
pair_style nm/cut/coul/cut - N-M potential with cutoff Coulomb
pair_style nm/cut/coul/long - N-M potential with long-range Coulombics
pair_style peri/eps - peridynamic EPS potential
pair_style peri/lps - peridynamic LPS potential
pair_style peri/pmb - peridynamic PMB potential
pair_style peri/ves - peridynamic VES potential
pair_style polymorphic - polymorphic 3-body potential
pair_style reax - ReaxFF potential
pair_style rebo - 2nd generation REBO potential of Brenner
pair_style resquared - Everaers RE-Squared ellipsoidal potential
pair_style snap - SNAP quantum-accurate potential
pair_style soft - Soft (cosine) potential

Force fields: examples

pair_style sw - Stillinger-Weber 3-body potential
pair_style table - tabulated pair potential
pair_style tersoff - Tersoff 3-body potential
pair_style tersoff/mod - modified Tersoff 3-body potential
pair_style tersoff/zbl - Tersoff/ZBL 3-body potential
pair_style tip4p/cut - Coulomb for TIP4P water w/out LJ
pair_style tip4p/long - long-range Coulombics for TIP4P water w/out LJ
pair_style tri/lj - LJ potential between triangles
pair_style vashishta - Vashishta 2-body and 3-body potential
pair_style yukawa - Yukawa potential
pair_style yukawa/colloid - screened Yukawa potential for finite-size particles
pair_style zbl - Ziegler-Biersack-Littmark potential