FYS2160 2023, Lecture 5

Dag Kristian Dysthe

Molecular dynamics

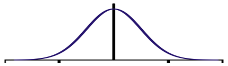
04.09.2023

First lecture:

Thermodynamics

- Macroscopic
- Continuum matter

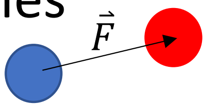
- Differentiable



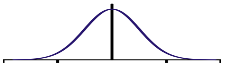
- Necessary relations based on some axioms
  - Always true for all matter
  - Necessary tool for theory
  - Always present in applications (engineering, chemistry, geoscience...)
- All properties of matter (ΔH_m, ΔS_v, c_V, λ, D) must be measured

Statistical physics

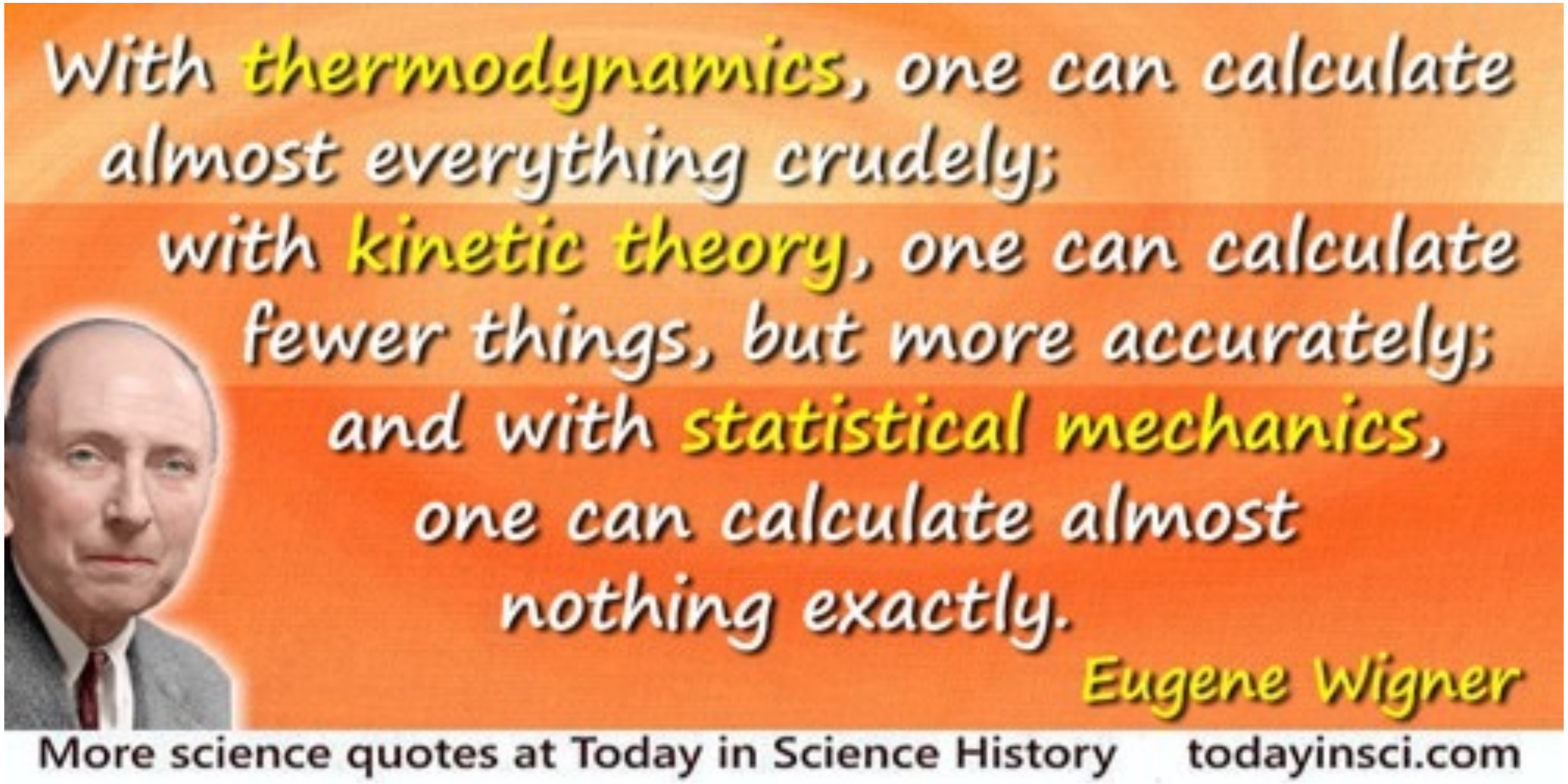
- Microscopic
- Discrete particles
- Mechanics



- Statistical behaviour of simplified models



- Bottom up explanation of thermodynamics
- Properties of model matter (ΔH_m, ΔS_v, c_V, λ, D) can be calculated and measured in simulation



With thermodynamics, one can calculate almost everything crudely; with kinetic theory, one can calculate fewer things, but more accurately; and with statistical mechanics, one can calculate almost nothing exactly.

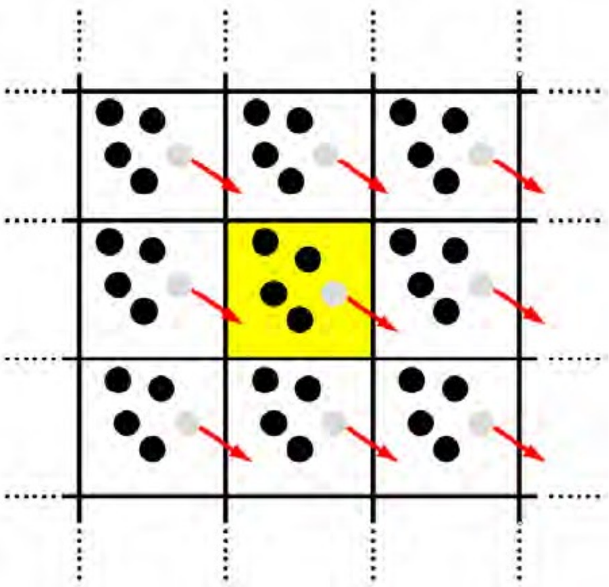
Eugene Wigner

More science quotes at Today in Science History todayinsci.com

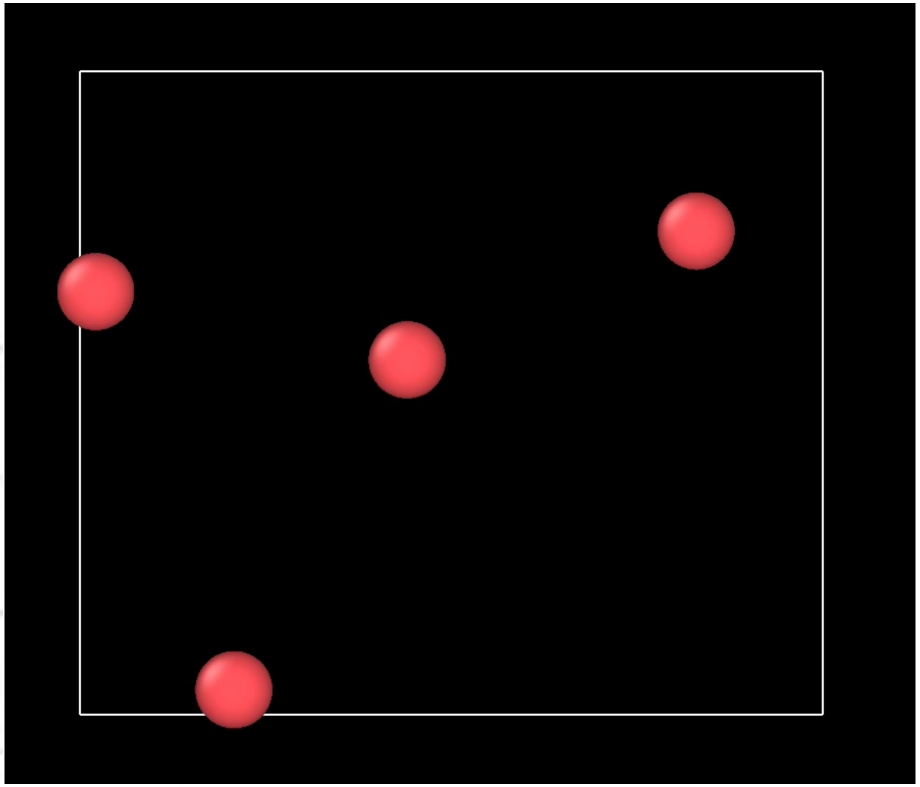
- But statistical mechanics defines how to calculate thermodynamic properties from simulation of model systems.
- This makes stat. mech. even more important in the era of computers!
- This is why computation should be part of your thermydyn. & stat. mech. curriculum!

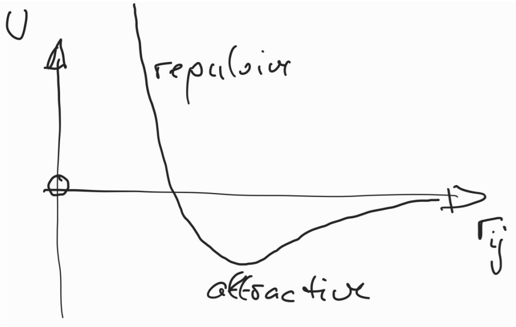
Molecular dynamics simulations

- Method for analyzing the physical movements of atoms and molecules (particles).
  - small systems 10^2-10^9 particles.
  - short duration: ns - µs
  - dynamic motion, not QM effects

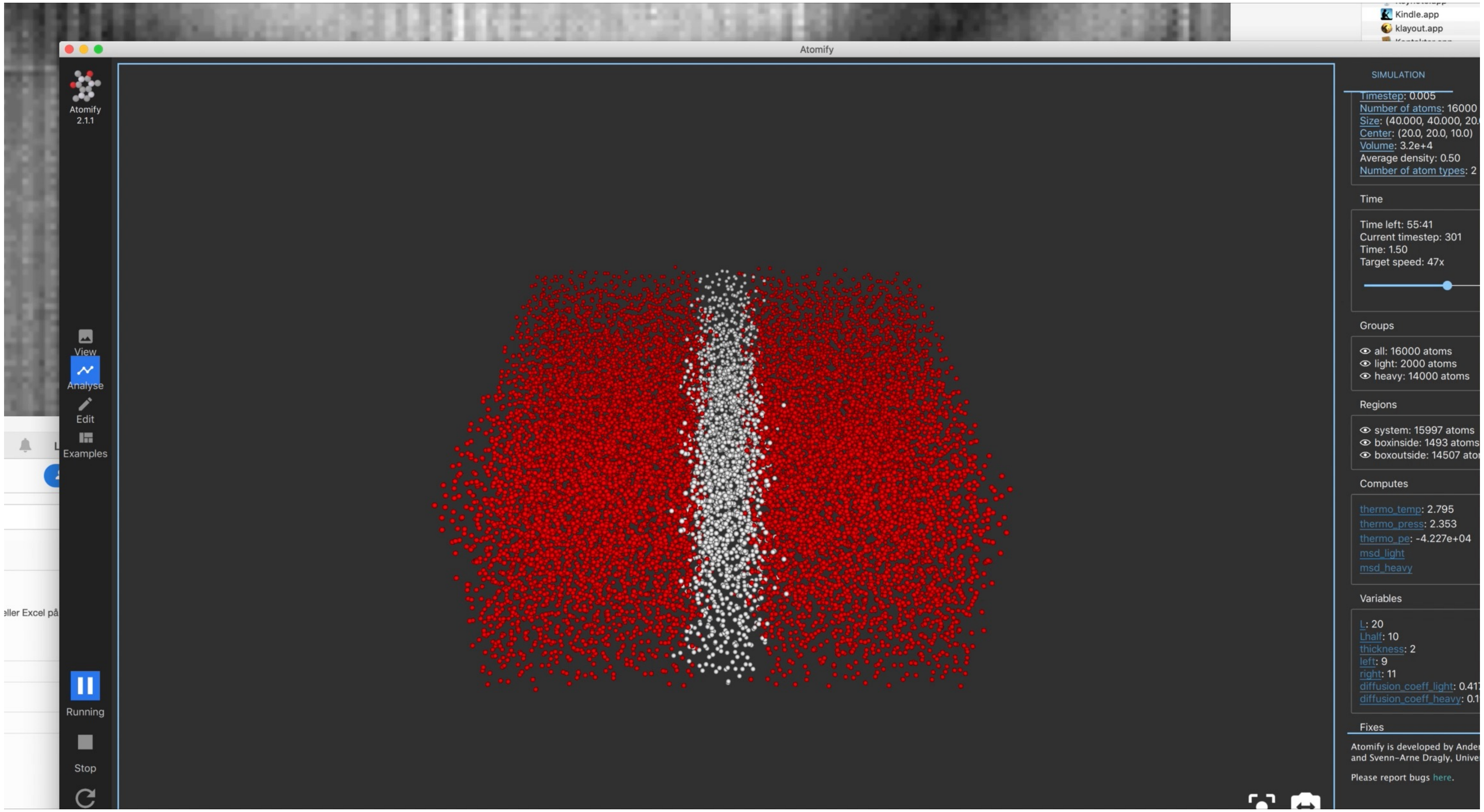


- Atoms interacting by pairwis *(neglect QM, manybody...)*
- Force on particle i: -$\boldsymbol{F}_i = \sum_j$
- Give atoms some initial posi
- Solve Newtons equations of $m\,{d^2\boldsymbol{r}}/_{dt^2}$ discretized
  - $r_i(t+dt) = r_i(t) + v_i(t)\,dt$
  - $v_i(t+dt) = v_i(t) + F_i(t+dt)dt/m_i$
- Record **trajectories,** $\boldsymbol{r}_i(t)$, $\boldsymbol{v}_i(t)$, $\boldsymbol{F}_i(t), U_i(t)$,... to compute statistical averages





Simulation of a model system



SIMULATION

Timestep: 0.005
Number of atoms: 16000
Size: (40.000, 40.000, 20.
Center: (20.0, 20.0, 10.0)
Volume: 3.2e+4
Average density: 0.50
Number of atom types: 2

Time

Time left: 55:41
Current timestep: 301
Time: 1.50
Target speed: 47x

Groups

all: 16000 atoms
light: 2000 atoms
heavy: 14000 atoms

Regions

system: 15997 atoms
boxinside: 1493 atoms
boxoutside: 14507 ato

Computes

thermo_temp: 2.795
thermo_press: 2.353
thermo_pe: -4.227e+04
msd_light
msd_heavy

Variables

L: 20
Lhalf: 10
thickness: 2
left: 9
right: 11
diffusion_coeff_light: 0.41
diffusion_coeff_heavy: 0.1

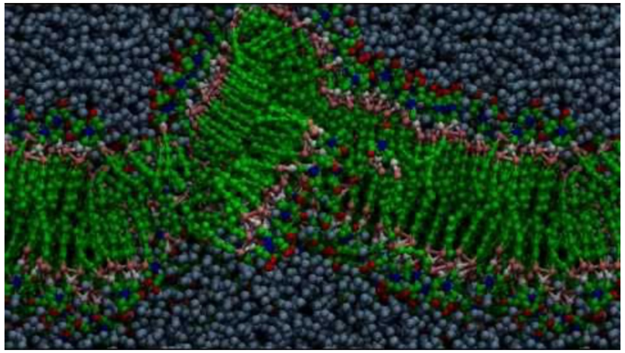
Fixes

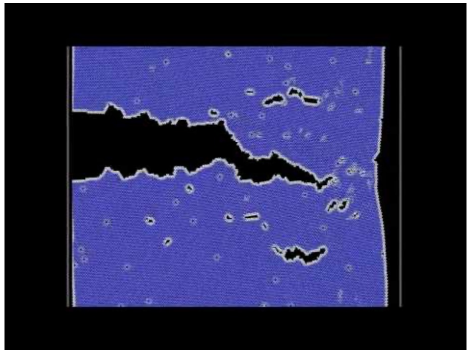
Atomify is developed by Ande
and Svenn-Arne Dragly, Unive

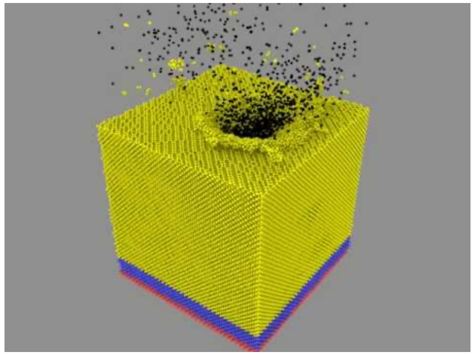
Please report bugs here.

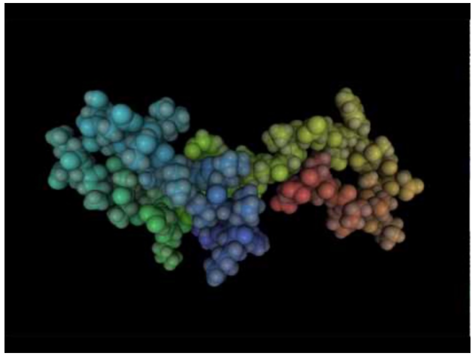
Molecular dynamics simulations

- Method for analyzing the physical movements of atoms and molecules (particles).
  - small systems 10^2-10^9 particles.
  - short duration: ns - µs
  - dynamic motion, not QM effects
- Atoms interacting by pairwise interatomic potentials: $U(\boldsymbol{r}_{ij})$ *(neglect QM, manybody...)*
- Force on particle i: $\boldsymbol{F}_i = \sum_j \frac{d}{dr_{ij}} U(\boldsymbol{r}_{ij})$
- Give atoms some initial positions $\boldsymbol{r}_i$ and velocity $\boldsymbol{v}_i$
- Solve Newtons equations of motion: $\boldsymbol{F} = m\boldsymbol{a} = {}^{d\boldsymbol{p}}\!/_{dt} = m\,{}^{d^2\boldsymbol{r}}\!/_{dt^2}$ discretized
  - $r_i(t+dt) = r_i(t) + v_i(t)\,dt$
  - $v_i(t+dt) = v_i(t) + F_i(t+dt)dt/m_i$
- Record **trajectories,** $\boldsymbol{r}_i(t)$, $\boldsymbol{v}_i(t)$, $\boldsymbol{F}_i(t)$, $U_i(t)$,... to compute statistical averages
- Dynamic behaviour of small systems directly visible and measurable







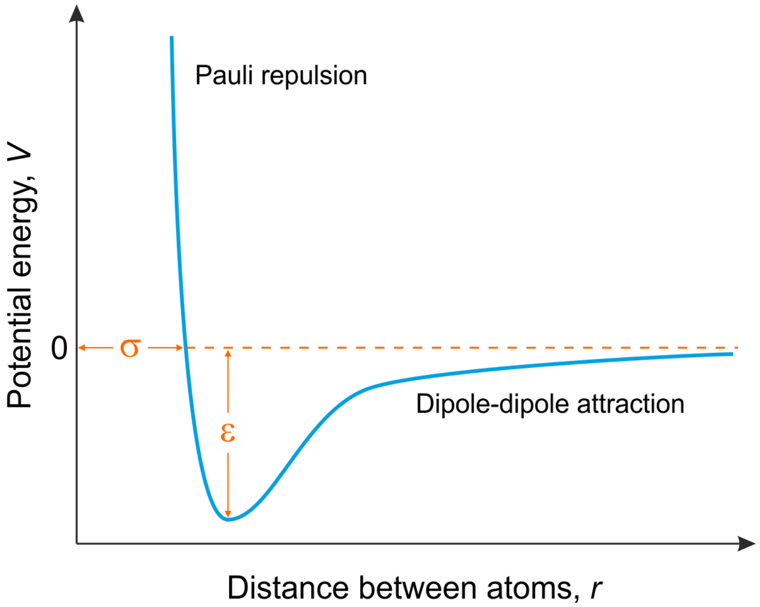


- Some thermodynamical properties *(T, P, E, c_V)* can be calculated from trajectory averages
- Some thermodynamical properties *(S, G, F, c_V,...)* can be calculated from MD «experiments» (perturbation -> response)

Interatomic pair potentials

- Lennard-Jones

$$U(r) = 4\epsilon\left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right)$$

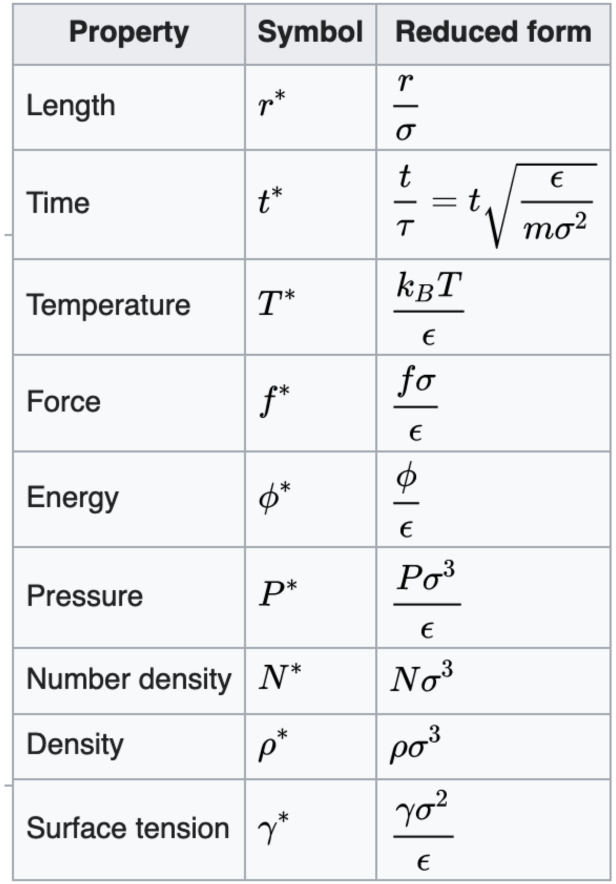


Example parameters:

- Argon:
  - σ = 0.34 nm
  - ε/k_B= 120 K
  - *m* = 40 u = 6.6e-26 kg
- Methane:
  - σ = 0.38 nm
  - ε/k_B= 148 K
  - *m* = 16 u = 6.6e-26 kg

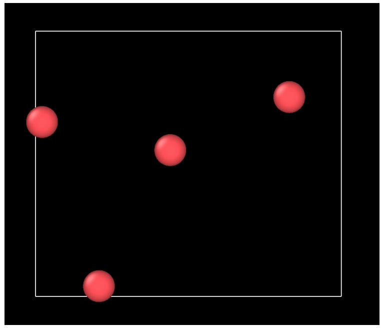
1 u = 1.66e-24 g = 1 g/N_A

Reduced (dimensionless) units

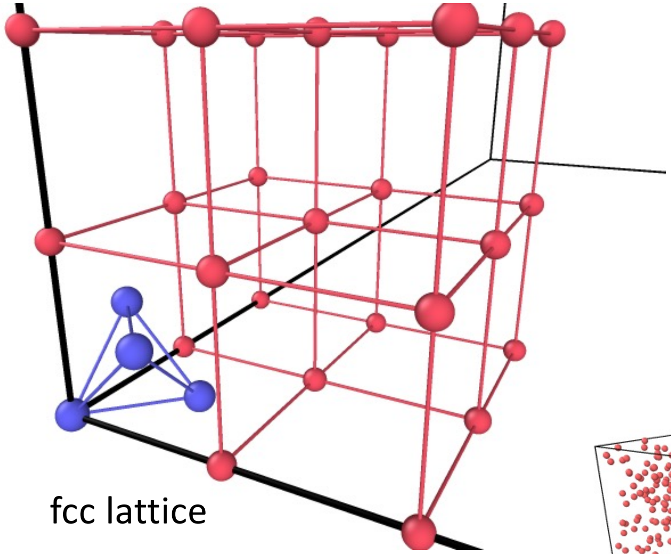


Property	Symbol	Reduced form
Length	r^*	$\frac{r}{\sigma}$
Time	t^*	$\frac{t}{\tau} = t\sqrt{\frac{\epsilon}{m\sigma^2}}$
Temperature	T^*	$\frac{k_B T}{\epsilon}$
Force	f^*	$\frac{f\sigma}{\epsilon}$
Energy	ϕ^*	$\frac{\phi}{\epsilon}$
Pressure	P^*	$\frac{P\sigma^3}{\epsilon}$
Number density	N^*	$N\sigma^3$
Density	ρ^*	$\rho\sigma^3$
Surface tension	γ^*	$\frac{\gamma\sigma^2}{\epsilon}$

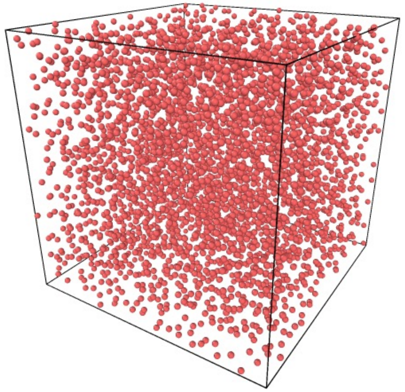
Setting up

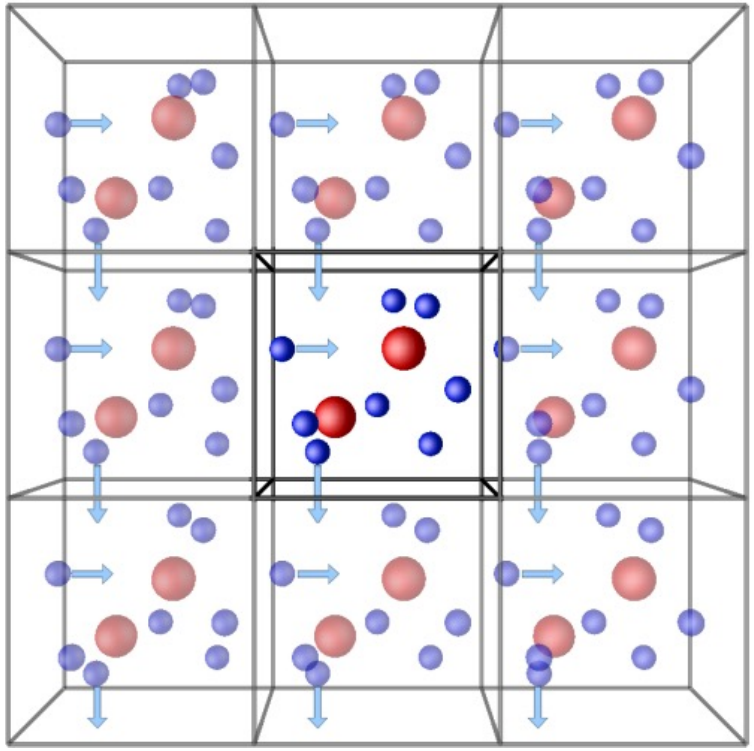


- Initializing: Lattice, melting
- Periodic boundary conditions



fcc lattice

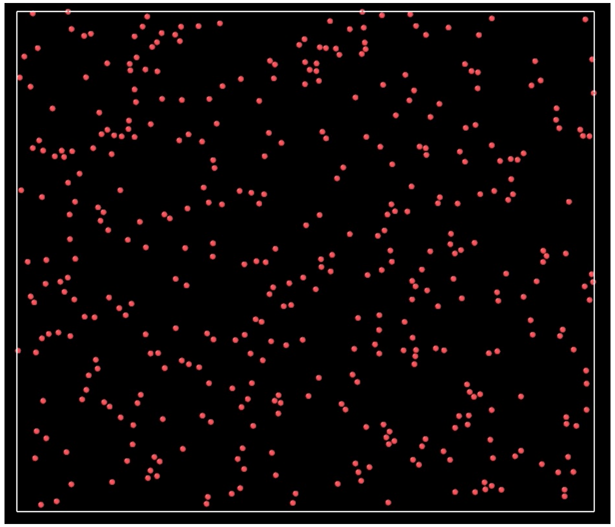


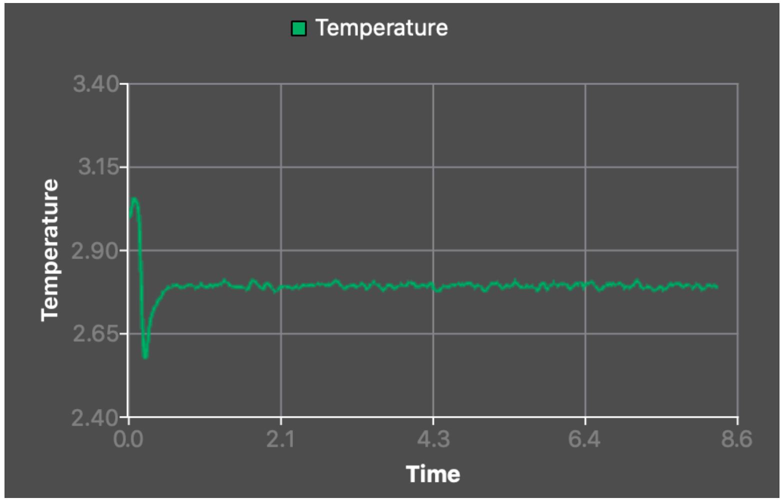


Analyzing trajectories

- Visualizing
  - Ovito
  - VMD
  - …

- Calculating averages
  - Python
  - Matlab
  - …





Or integrated: Atomify

LAMMPS Molecular Dynamics Simulator

Large-scale Atomic/Molecular Massively Parallel Simulator

https://lammps.sandia.gov

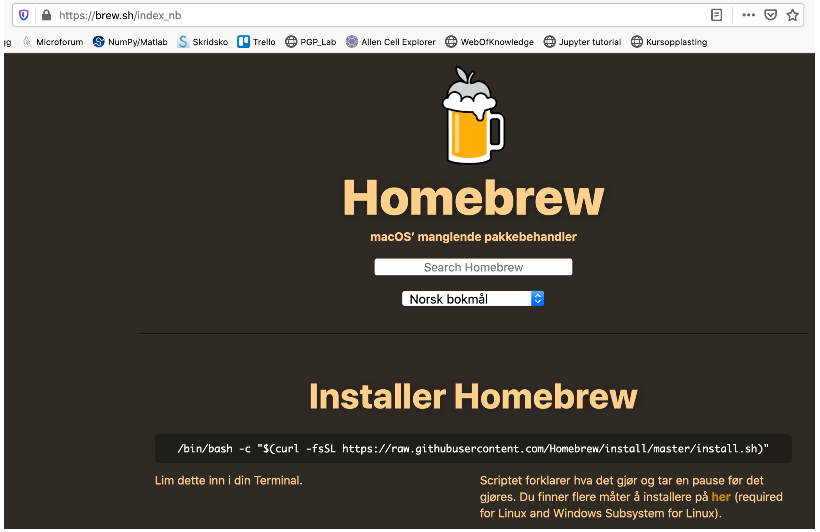
Documentation:

https://docs.lammps.org/

Installing Lammps

- https://docs.lammps.org/Install.html
- Lines starting with dollar sign $ means «in a terminal window write the command after the dollar sign»
- I did it on a Mac:
  - Homebrew installation (7 min)
    - `$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"`
  - `$ brew install lammps` (3 min)
  - `$ brew test lammps -v` (1 min)
  - `$ brew install openkim-models` (1 min)

- Either build or download executable binaries
- Lammps binary is only ~15 MB



Homebrew

macOS' manglende pakkebehandler

Installer Homebrew

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

Lim dette inn i din Terminal.

Scriptet forklarer hva det gjør og tar en pause før det gjøres. Du finner flere måter å installere på her (required for Linux and Windows Subsystem for Linux).

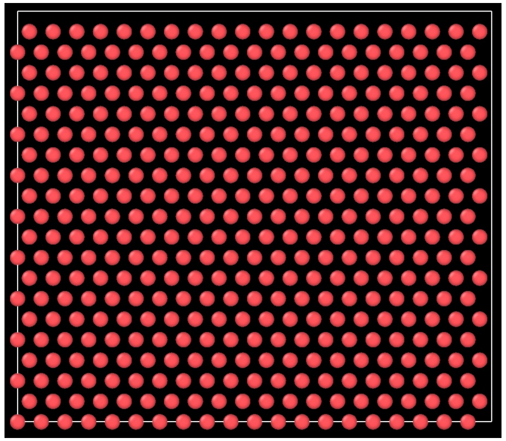
Installing

- Python:
  - Log files: (thermodynamic data): https://github.com/henriasv/lammps-logfile
  - Trajectory files (python not working yet)
- Matlab:
  - Trajectory files: readdump_all.m readdump_one.m
  - Log files (thermodynamic data):
    - readlog.m this does not work for me...
    - strip everything but the numbers and use out=load('log.strip');
- Visualization:
  - download ovito (https://www.ovito.org/)

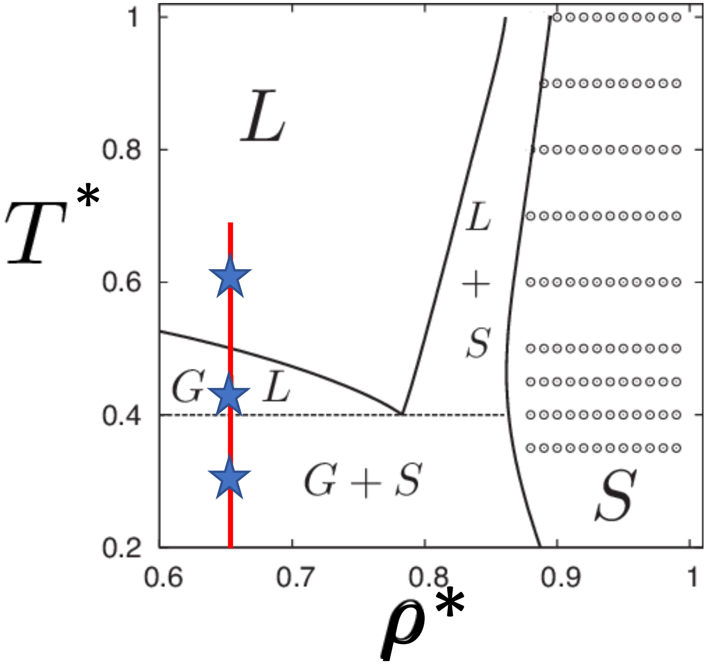
First simulation, run and visualize

- Make a directory for the simulation
- copy **in.myfirstmd** from todays lecture page/
- Run the simulation in a terminal window (Windows: use cmd, not Lammps-shell)
  - **`$ lmp_serial < in.myfirstmd   (Mac)`**
  - **`$ lmp –in in.myfirstmd (Windows)`**
- Vizualisation:
  - download ovito (https://www.ovito.org/)
  - open ovito
  - File>Load File> **dump.lammpstrj**
- Change parameters and rerun
  - open in.myfirstmd in an editor (not Word!)
  - change temperatures in line «**fix 1 all nvt temp 0.3 0.3 0.3**» 0.3 -> 0.4 and 0.6
  - rerun simulation (`$ lmp_serial < in.myfirstmd`)
  - visualize new run (File>Load File> dump.lammpstrj)

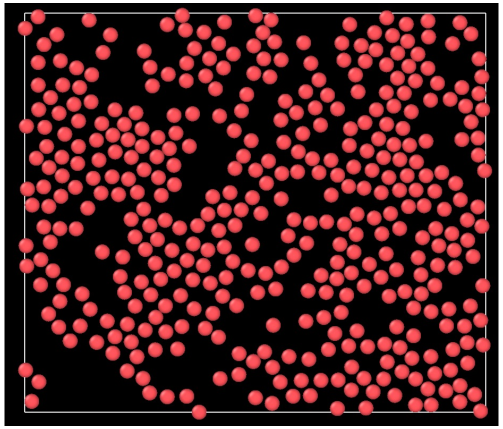
What is the difference between runs?



velocity all create 0.65 87287

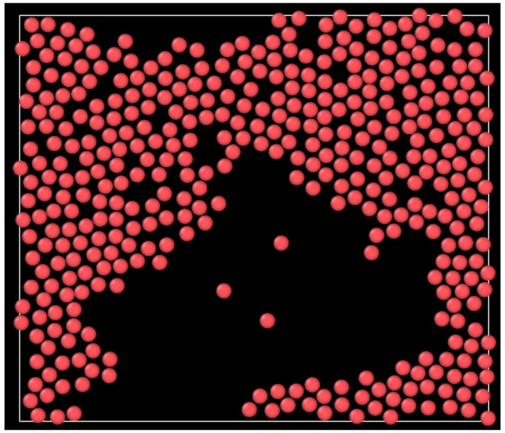


fix 1 all nvt temp 0.6 0.6 0.6



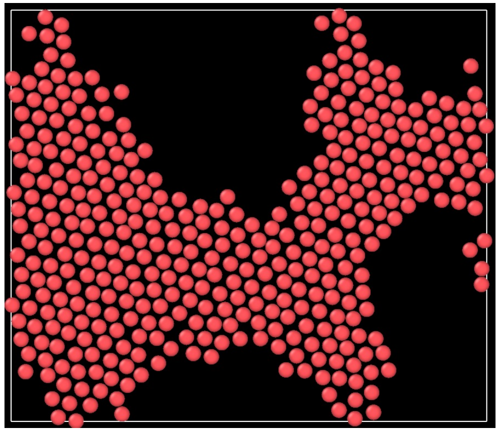
supercritical gas

fix 1 all nvt temp 0.4 0.4 0.4

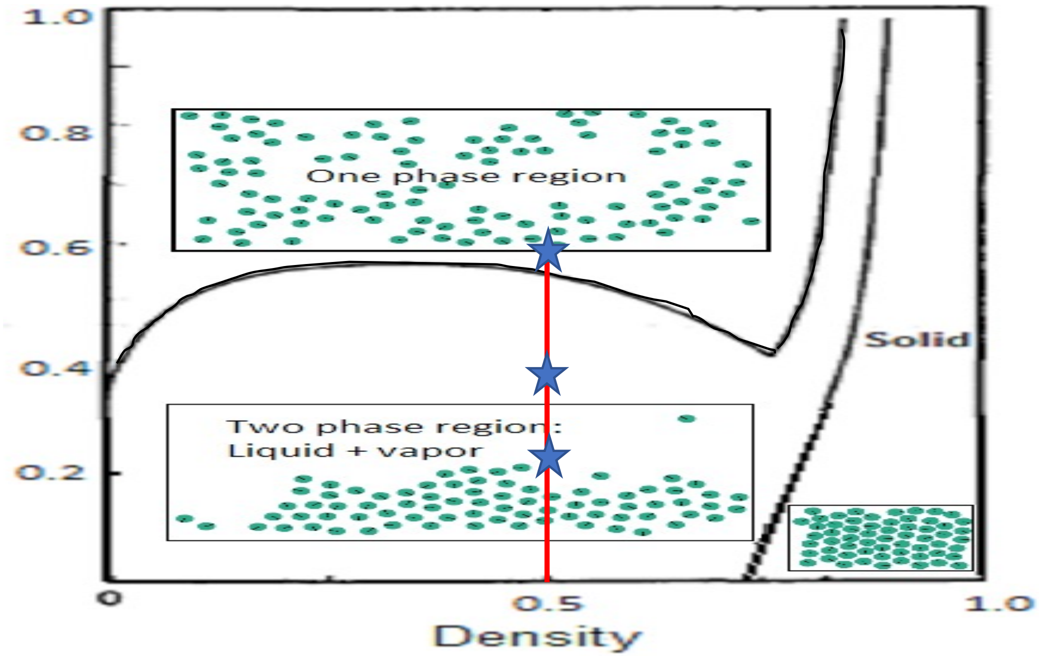


liquid

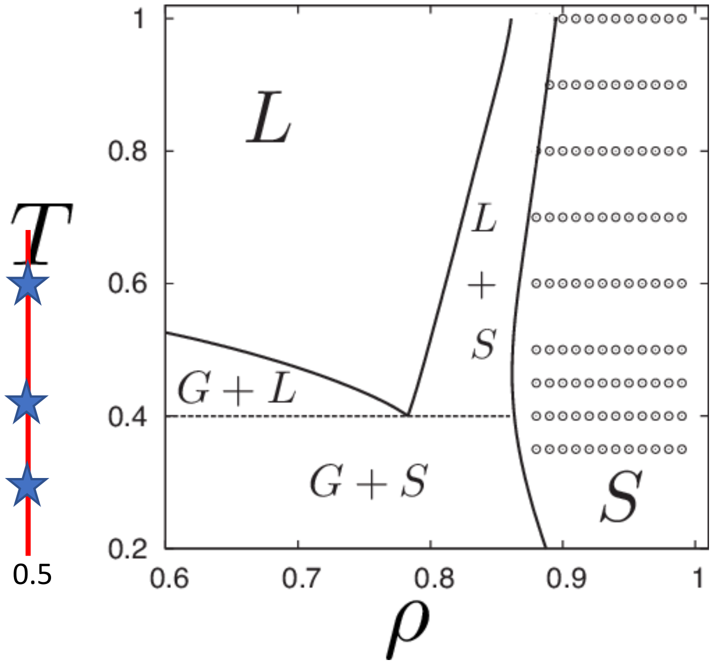
fix 1 all nvt temp 0.3 0.3 0.3



solid



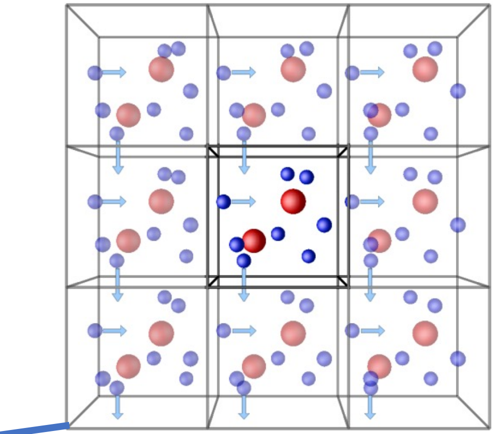
velocity all create 0.5 87287

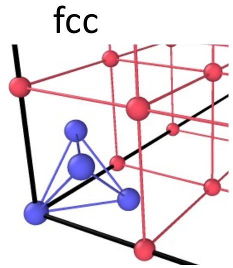


2D Lennard-Jones

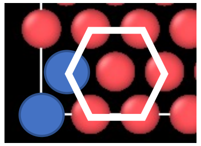
Lammps input file

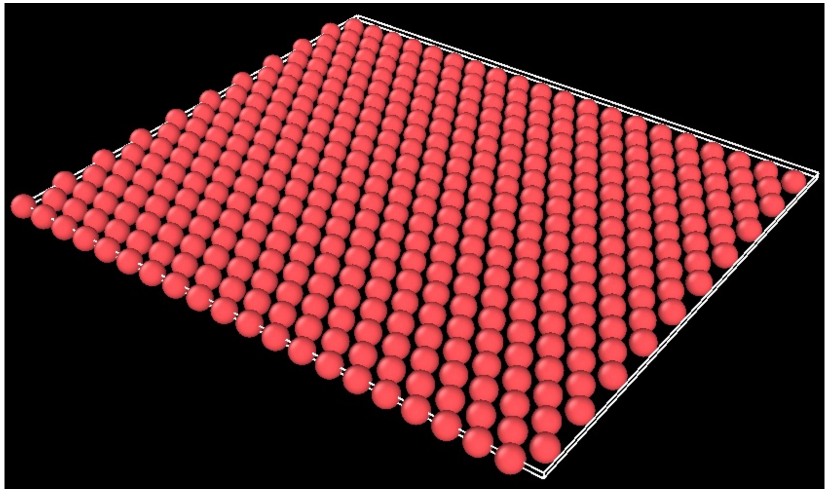
in.myfirstmd





```
# 2d Lennard-Jones gas
units lj
dimension 2
boundary p p p
atom_style atomic
lattice hex 0.75
region simbox block 0 20 0 10 -0.1 0.1
create_box 1 simbox
create_atoms 1 box
```





- **create_atoms** command
  - Syntax: create_atoms type style args keyword values
  - Example: create_atoms 1 region 2
  - This command creates atoms (or molecules) on a lattice, or a single atom (or molecule), or a random collection of atoms (or molecules), as an alternative to reading in their coordinates explicitly via a read_data or read_restart command. A simulation box must already exist, which is typically created via the create_box command. Before using this command, a lattice must also be defined using the lattice command

```
...
lattice hex 0.75
region simbox block 0 20 0 10 -0.1 0.1
create_box 1 simbox
create_atoms 1 box
...
```

- **create_box** command
  - Syntax: create_box N region-ID keyword value
  - N = # of atom types to use in this simulation
  - Example: create_box 2 mybox
  - This command creates a simulation box based on the specified region. Thus a region command must first be used to define a geometric domain.
- **region** command
  - Syntax: region ID style args keyword arg
  - Example: region 2 block 0 10 0 10 -0.5 0.05
  - *block* args = xlo xhi ylo yhi zlo zhi
  - This command defines a geometric region of space.
- **lattice command**
  - Syntax: lattice style scale keyword values
  - Example: lattice hex 0.25
  - style = *none* or *sc* or *bcc* or *fcc* or *hcp* or *diamond* or *sq* or *sq2* or *hex* or *custom*
  - scale = reduced density rho* (for LJ units)

Lammps input

in.myfirstmd

```
...
create_atoms 1 box
mass 1 1.0
                        T
velocity all create 2.5 8
pair_style lj/cut 2.5
pair_coeff 1 1 1.0 1.0
fix 1 all nve
dump 1 all custom 10 dump.lammpstrj id type x y z vx vy vz
thermo_style custom time pe ke temp press
thermo 100
run 5000
```

Dump a snapshot of atom 1 quantities to file dump.lammpstrj every N timesteps in one of several styles (here style=custom)

```
ITEM: TIMESTEP
0
ITEM: NUMBER OF ATOMS
400
ITEM: BOX BOUNDS pp pp pp
0.0000000000000000e+00 2.4816129576055989e+01
0.0000000000000000e+00 2.1491398636470834e+01
-1.2408064788027995e-01 1.2408064788027995e-01
ITEM: ATOMS id type x y z vx vy vz
1 1 0 0 0 1.02727 0.196113 0
2 1 0.620403 1.07457 0 -1.65512 0.826505 0
3 1 1.24081 0 0 -0.616608 -2.40564 0
4 1 1.86121 1.07457 0 0.235659 1.33101 0
5 1 2.48161 0 0 -0.129845 1.05664 0
6 1 3.10202 1.07457 0 2.4403 0.620481 0
7 1 3.72242 0 0 -1.61211 -0.245549 0
8 1 4.34282 1.07457 0 -0.910977 -2.73645 0
9 1 4.96323 0 0 -0.693632 -2.58296 0
10 1 5.58363 1.07457 0 0.705439 -0.91755 0
11 1 6.20403 0 0 -1.76336 2.61689 0
```

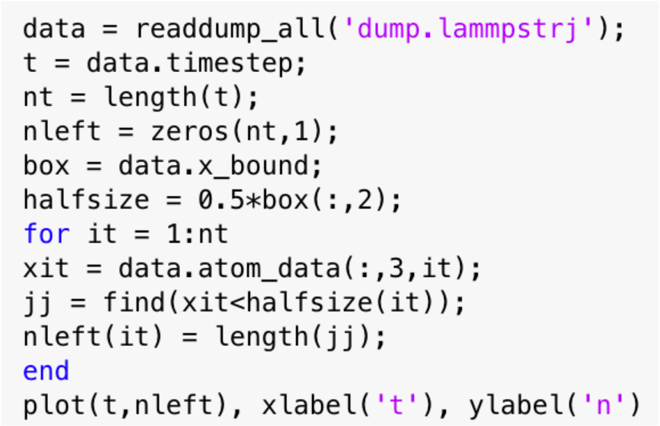
thermo N = output thermodynamics every N timesteps

$${}^{12} - \left(\frac{\sigma}{r}\right)^{6}\Big] \qquad r < r_c$$

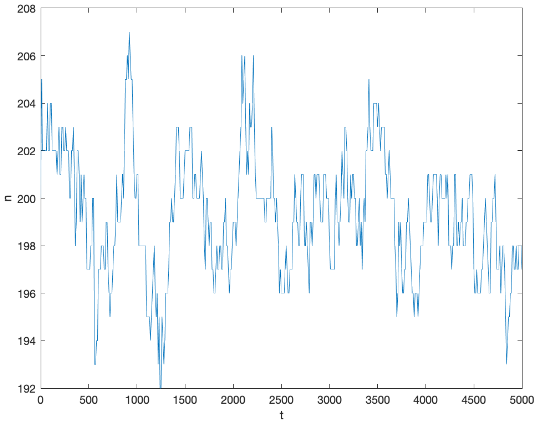
ε	σ	r_c
1.0	1.0	2.5
2.0	2.0	2.5
1.4	1.5	2.5

First run

- `$ lmp_serial < in.myfirstmd`
- download `readdump_all.m` and `readdump_one.m`
- run data analysis program in Matlab
- Vizualisation:
  - download ovito
  - open ovito
  - load dump.lammpstrj



```
data = readdump_all('dump.lammpstrj');
t = data.timestep;
nt = length(t);
nleft = zeros(nt,1);
box = data.x_bound;
halfsize = 0.5*box(:,2);
for it = 1:nt
xit = data.atom_data(:,3,it);
jj = find(xit<halfsize(it));
nleft(it) = length(jj);
end
plot(t,nleft), xlabel('t'), ylabel('n')
```



https://github.com/henriasv/lammps-logfile

Python

```
import lammps_logfile

log = lammps_logfile.File("./log.lammps")

t = log.get("Time")
T = log.get("Temp")
Ek = log.get("KinEng")
Ep = log.get("PotEng")
P = log.get("Press")

import matplotlib.pyplot as plt
plt.plot(t, Ek+Ep)
plt.show()
plt.plot(t, T)
plt.show()
```

Matlab

Logfile:

```
out=load('log.strip');
t=out(:,1);
Ep=out(:,2);
Ek=out(:,3);
T=out(:,4);
P=out(:,5);

figure(2), plot(t,T)
xlabel('t'), ylabel('T')
figure(3), plot(t,Ep+Ek)
xlabel('t'), ylabel('E_{tot}')
```

Trajectory file:

```
data = readdump_all('dump.lammpstrj');
t = data.timestep;
vx=squeeze(data.atom_data(:,5,:));
vy=squeeze(data.atom_data(:,6,:));
vxsq=mean(vx.^2);
vysq=mean(vy.^2);
T=(vxsq+vysq)/2;
figure(1), plot(t,vxsq,'-r',t,vysq,'-b',t,T,'-k')
xlabel('t'), ylabel('T')
```

Computing macroscopic properties

- **compute** command
  - Syntax: compute ID group-ID style args
  - Define a computation that will be performed on a group of atoms. Quantities calculated by a compute are instantaneous values

- **compute temp** command
  - Syntax: compute ID group-ID temp
  - Example: compute 1 all temp
  - Define a computation that calculates the temperature of a group of atoms.
  - The temperature is calculated by the formula KE = 3/2 N k T, where KE = total kinetic energy of the group of atoms (sum of 1/2 m v^2)
  - This compute subtracts out degrees-of-freedom due to fixes that constrain molecular motion

Macroscopic properties

- Pressure = Force / Area
  - *[P]= [F]/[A]=N/m^2*
- Newtonian mechanics
  - $\vec{F} = m\vec{a} = \frac{d\vec{p}}{dt}$
  - Used this to calculate pressure of ideal gas:

$$P_x = \frac{1}{A}\sum_i \frac{\Delta p_{x,i}}{\Delta t} = \frac{1}{A} N \frac{m\bar{v}_x^2}{2L} = \frac{N k_B T}{V} = \rho k_B T$$

- When forces at distance:
  - $P = \rho k_B T + \frac{1}{3V}\sum_{i<j} \vec{f}\left(\vec{r}_{ij}\right) \cdot \vec{r}_{ij}$
  - second term: virial

- **compute pressure** command
  - Syntax: compute ID group-ID pressure temp-ID keyword
  - Example: compute 1 all pressure thermo_temp
  - Define a computation that calculates the pressure of the entire system of atoms.

MD heat capacity experiment

- fix heat command
  - Syntax: fix ID group-ID heat N eflux
  - Add non-translational kinetic energy (heat) to a group of atoms in a manner that conserves their aggregate momentum.

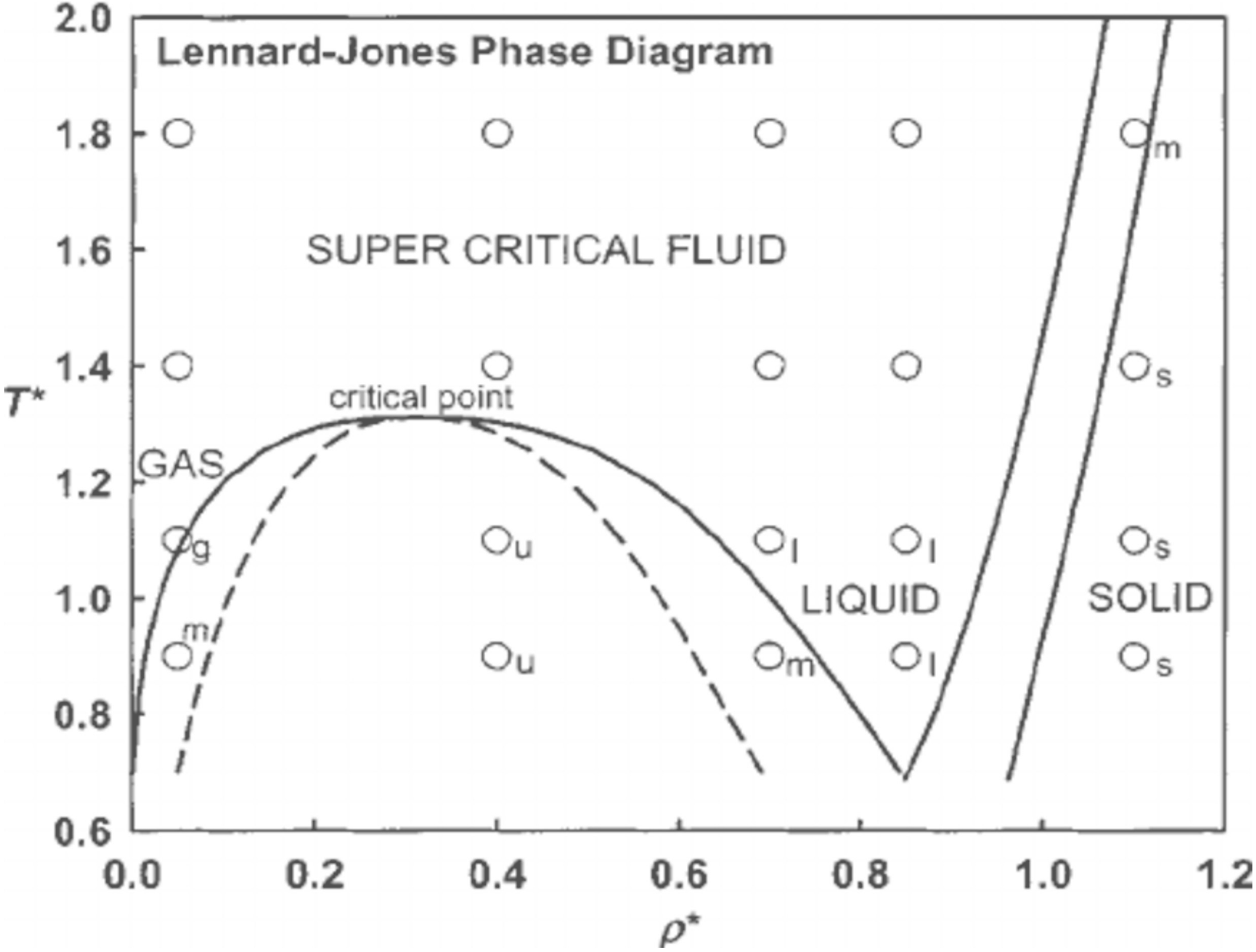
```
variable eFlux equal 0.01

fix heat all heat 1 ${eFlux} region box
```

Summary

- Molecular dynamics: experiments on model systems
- Statistical mechanics is used to find how to measure thermodynamic properties
- Lammps: Efficient implementation can run on telephone (Atomify)
- Visualization: Ovito, VMD, Atomify

3D



Lennard-Jones Phase Diagram
SUPER CRITICAL FLUID
critical point
GAS
LIQUID
SOLID
g
u
u
l
l
l
m
m
m
s
s
s
T^*
ρ^*
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.0
0.2
0.4
0.6
0.8
1.0
1.2

Integrating equations of motion

- Velocity Verlet

$$\begin{aligned}
\mathbf{v}_i(t + \varDelta t/2) &= \mathbf{v}_(t) + \mathbf{F}_i(t)/m_i \varDelta t/2 \\
\mathbf{r}_i(t + \varDelta t) &= \mathbf{r}_(t) + \mathbf{v}_i(t + \varDelta t/2) \\
\mathbf{F}_i(t + \varDelta t) &= -\nabla V(\mathbf{r}_i(t + \varDelta t)) \\
\mathbf{v}_i(t + \varDelta t) &= \mathbf{v}_(t + \varDelta t/2) + \mathbf{F}_i(t + \varDelta t)/m_i \varDelta t/2 \ , ,
\end{aligned}$$