MODULE 2: MOLECULAR DYNAMICS

LAMMPS Walkthrough

LAMMPS + OVITO

LAMMPS + OVITO

SOFTWARE

- We will be using LAMMPS to perform classical molecular simulations to predict materials properties and behavior

- LAMMPS is freely available from http://lammps.sandia.gov

- We will be visualizing MD trajectories using the OVITO visualization package

- OVITO is freely available from www.ovito.org

LAMMPS + OVITO

EWS PRE-INSTALLATION

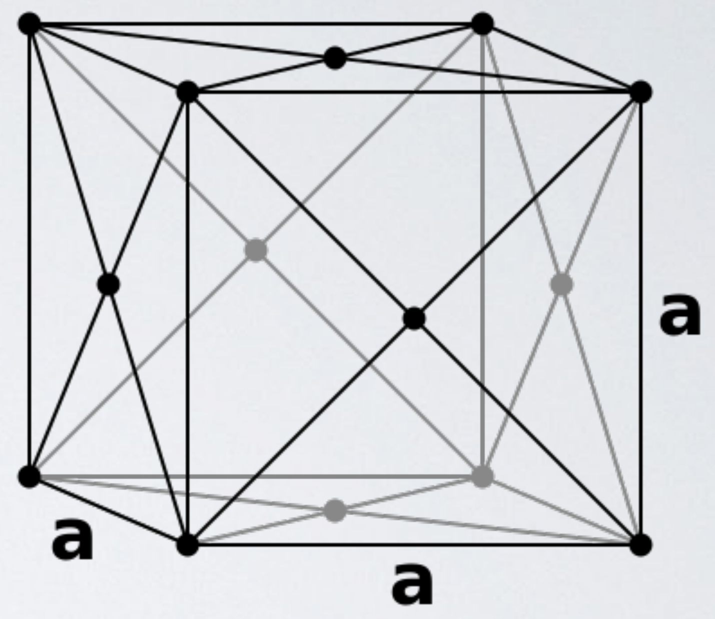
- A full installation of LAMMPS 29Sep2021 release is available on EWS Linux at:
  ```
  /class/mse404pla/lammps-29Sep2021/bin/lmp_serial
  ```
- The LAMMPS documentation is available in online at docs.lammps.org/Manual.html
- An installation of OVITO 2.6.1 is on EWS Linux at:
  ```
  module load ovito
  ```
- This may require you to correct your library path:
  ```
  export LD_LIBRARY_PATH=/lib64/:$LD_LIBRARY_PATH
  ```
- The Ovito documentation is available in online at http://www.ovito.org/manual/

Tutorial 1: Al cohesive energy

Tutorial 1: Al cohesive energy

- We will use LAMMPS to estimate the Al fcc cohesive energy, **E_{cohe}**, and lattice parameter, **a**

$$E_{cohe} = E_{solid} - \cancelto{0}{\sum_{atoms} E_{isolated}}$$



- Experimentally, **E_{cohe}** = -3.39 eV/atom* and **a** = 4.0495 Å*

- **Strategy:** We shall use a modern EAM potential for Al and optimize **E_{cohe}** as a function of **a**

*Charles Kittel. Introduction to Solid State Physics, 8th edition. Hoboken, NJ: John Wiley & Sons, Inc, 2005. *http://periodictable.com/Elements/013/data.html

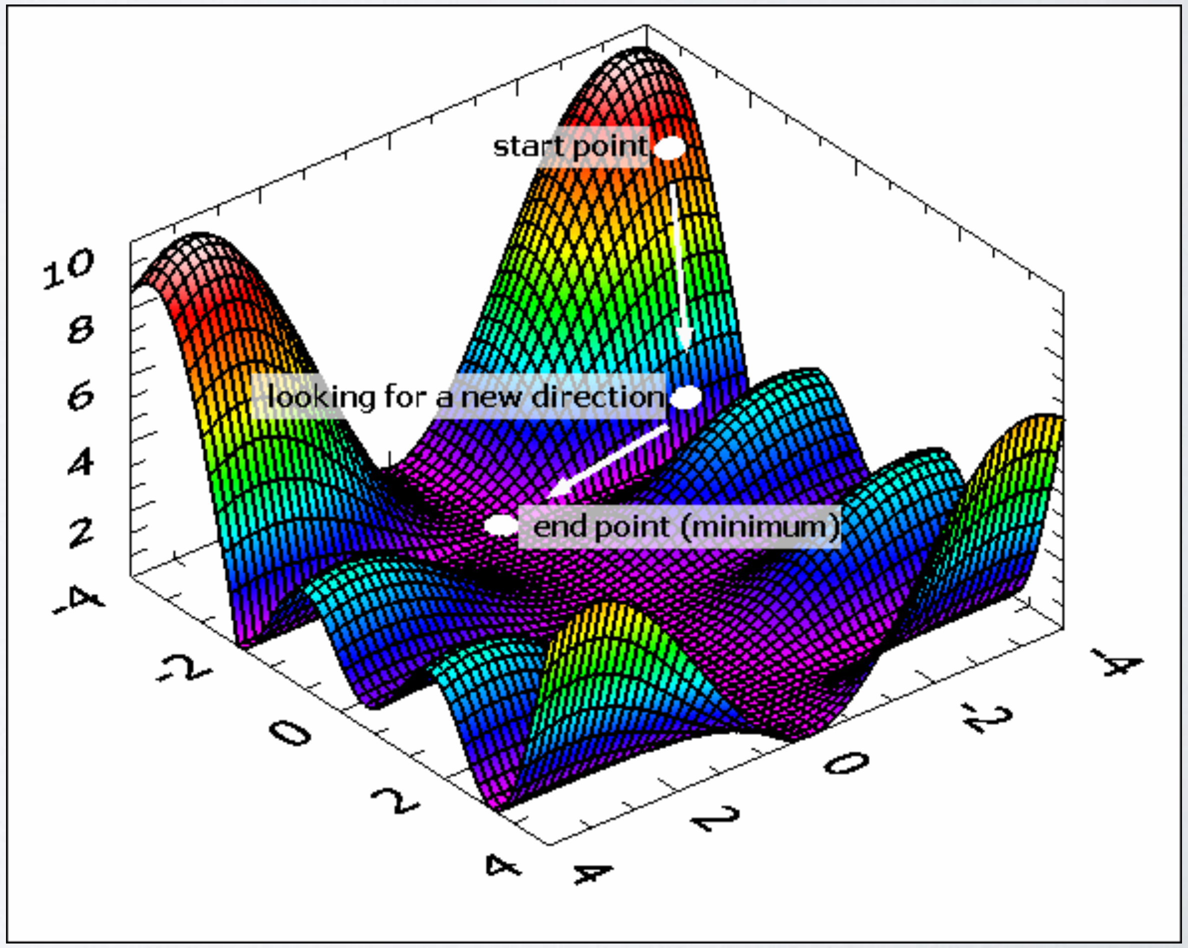
Tutorial 1: Al cohesive energy

- Be careful! In this first tutorial we are **NOT** performing conventional molecular dynamics (i.e., integrating F=ma)

- Rather we are performing a potential energy minimization, to find the lowest potential energy crystal coordinates

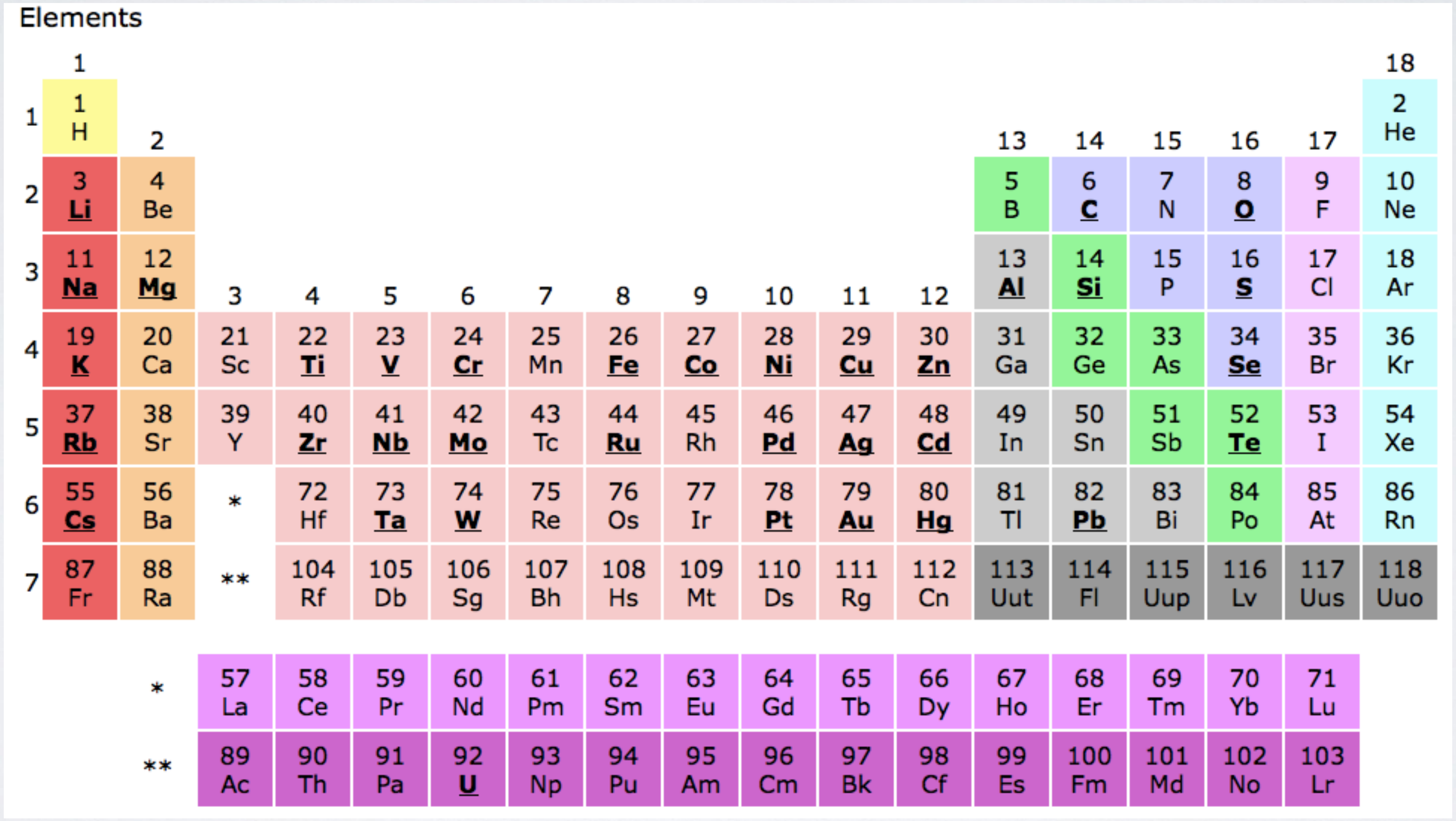
- We achieve this by performing conjugate gradient (or steepest descent) minimization of PE wrt atomic coords

- Accordingly, there are no atom velocities or temperature!



Tutorial 1: Al cohesive energy

1. Download **Al99.eam.alloy** EAM potential from NIST Interatomic Potentials Repository Project (http://www.ctcms.nist.gov/potentials)



Elements

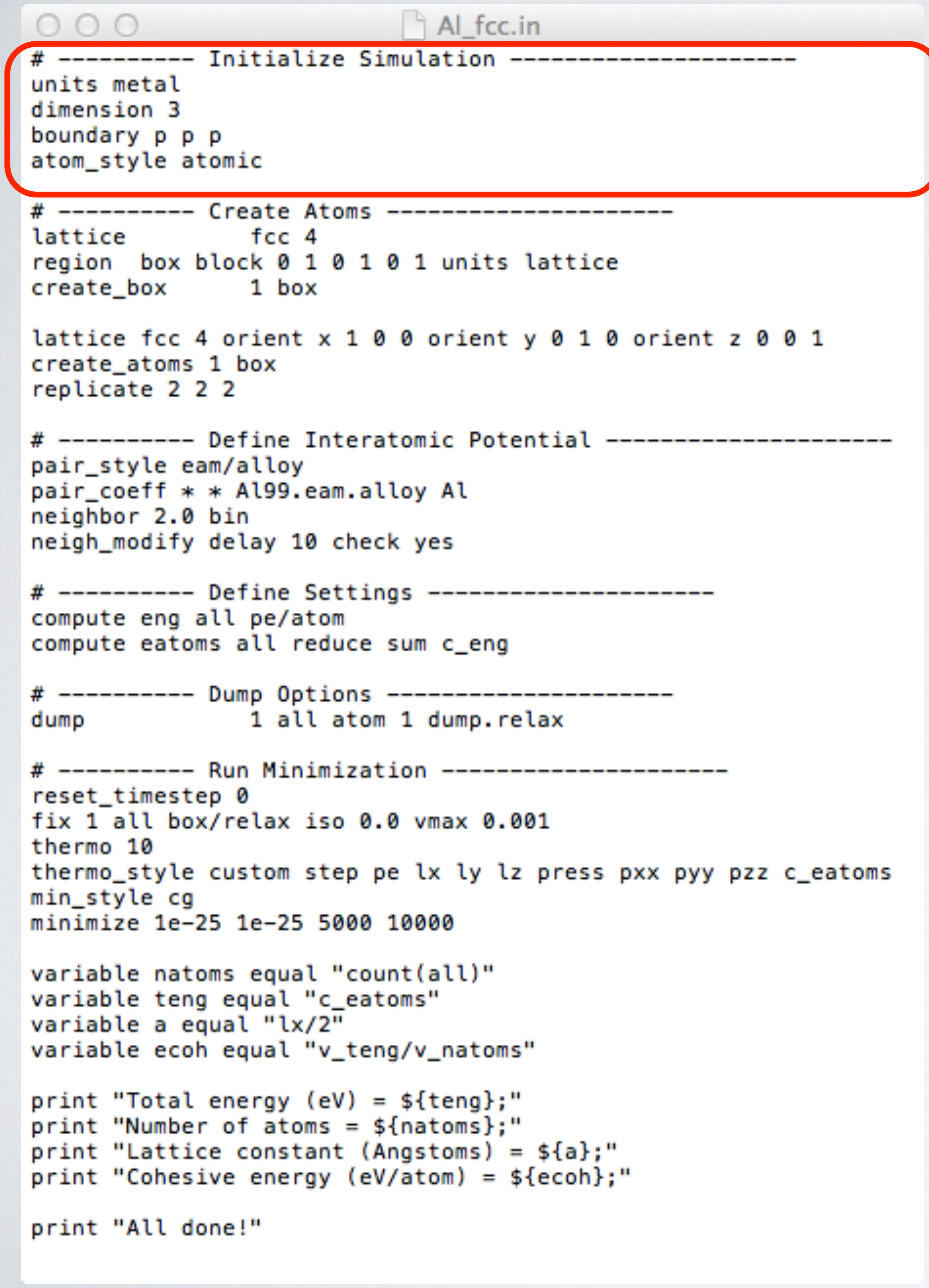
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1 H																	2 He
2	3 **Li**	4 Be											5 B	6 **C**	7 N	8 **O**	9 F	10 Ne
3	11 **Na**	12 **Mg**											13 **Al**	14 **Si**	15 P	16 **S**	17 Cl	18 Ar
4	19 **K**	20 Ca	21 Sc	22 **Ti**	23 **V**	24 **Cr**	25 Mn	26 **Fe**	27 **Co**	28 **Ni**	29 **Cu**	30 **Zn**	31 Ga	32 Ge	33 As	34 **Se**	35 Br	36 Kr
5	37 **Rb**	38 Sr	39 Y	40 **Zr**	41 **Nb**	42 **Mo**	43 Tc	44 **Ru**	45 Rh	46 **Pd**	47 **Ag**	48 **Cd**	49 In	50 Sn	51 Sb	52 **Te**	53 I	54 Xe
6	55 **Cs**	56 Ba	*	72 Hf	73 **Ta**	74 **W**	75 Re	76 Os	77 Ir	78 **Pt**	79 **Au**	80 **Hg**	81 Tl	82 **Pb**	83 Bi	84 Po	85 At	86 Rn
7	87 Fr	88 Ra	**	104 Rf	105 Db	106 Sg	107 Bh	108 Hs	109 Mt	110 Ds	111 Rg	112 Cn	113 Uut	114 Fl	115 Uup	116 Lv	117 Uus	118 Uuo

*	57 La	58 Ce	59 Pr	60 Nd	61 Pm	62 Sm	63 Eu	64 Gd	65 Tb	66 Dy	67 Ho	68 Er	69 Tm	70 Yb	71 Lu
**	89 Ac	90 Th	91 Pa	92 **U**	93 Np	94 Pu	95 Am	96 Cm	97 Bk	98 Cf	99 Es	100 Fm	101 Md	102 No	103 Lr

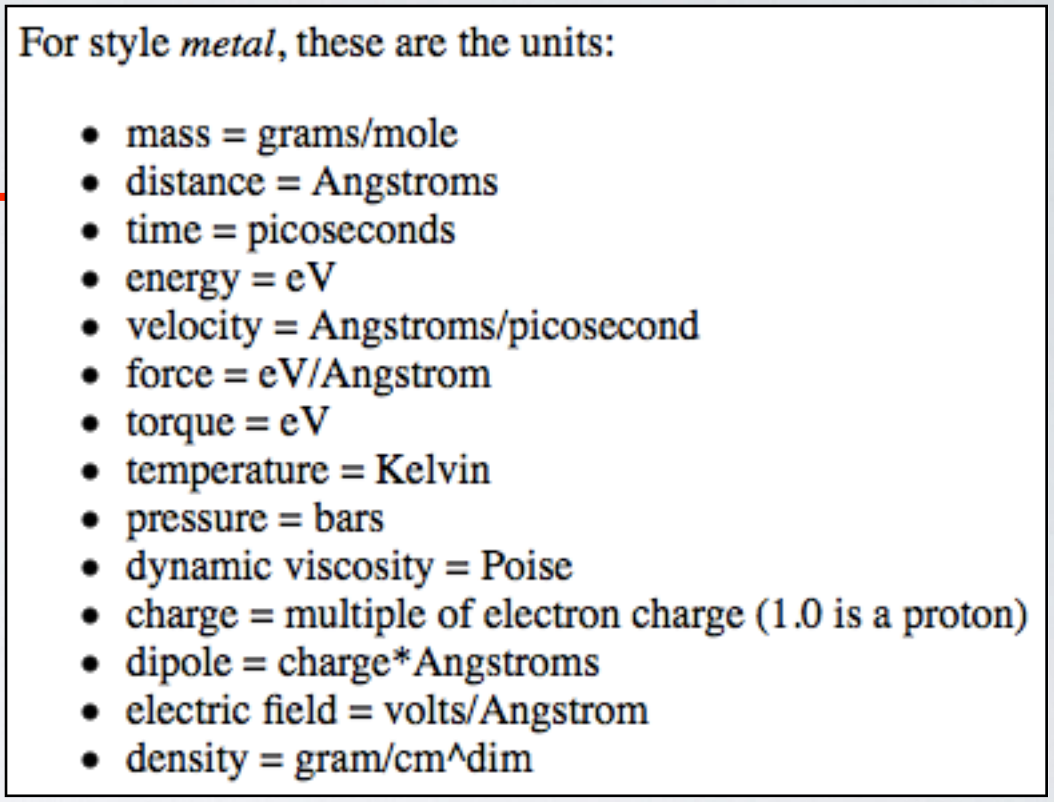
Tutorial 1: Al cohesive energy

2. Copy LAMMPS input file
   `/class/mse404pla/LAMMPS/Al_fcc.in`

Tutorial 1: Al cohesive energy


Al_fcc.in

```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region  box block 0 1 0 1 0 1 units lattice
create_box      1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

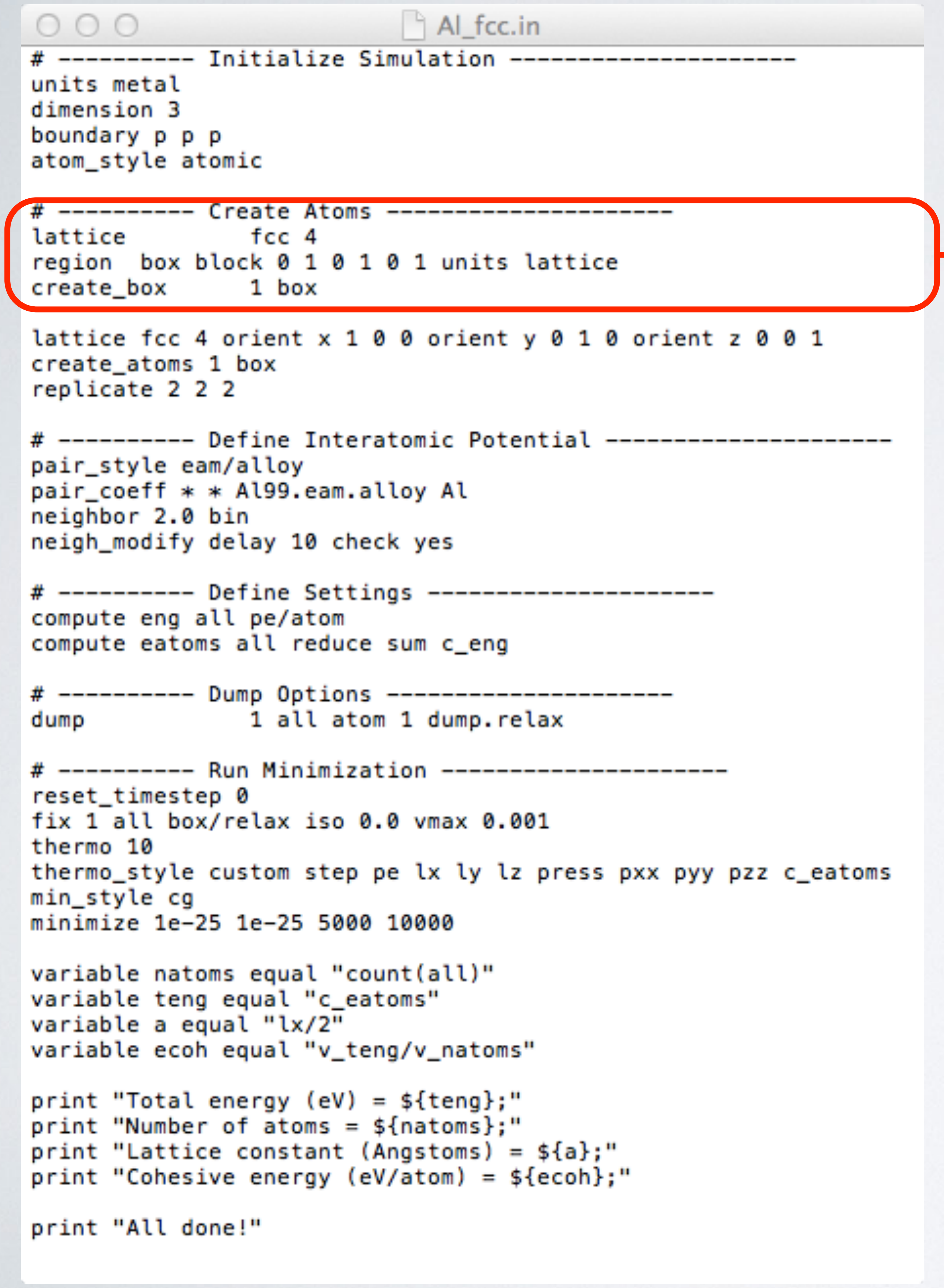

For style *metal*, these are the units:

- mass = grams/mole
- distance = Angstroms
- time = picoseconds
- energy = eV
- velocity = Angstroms/picosecond
- force = eV/Angstrom
- torque = eV
- temperature = Kelvin
- pressure = bars
- dynamic viscosity = Poise
- charge = multiple of electron charge (1.0 is a proton)
- dipole = charge*Angstroms
- electric field = volts/Angstrom
- density = gram/cm^dim

- **#** specifies a comment

- x,y,z **periodic boundaries**

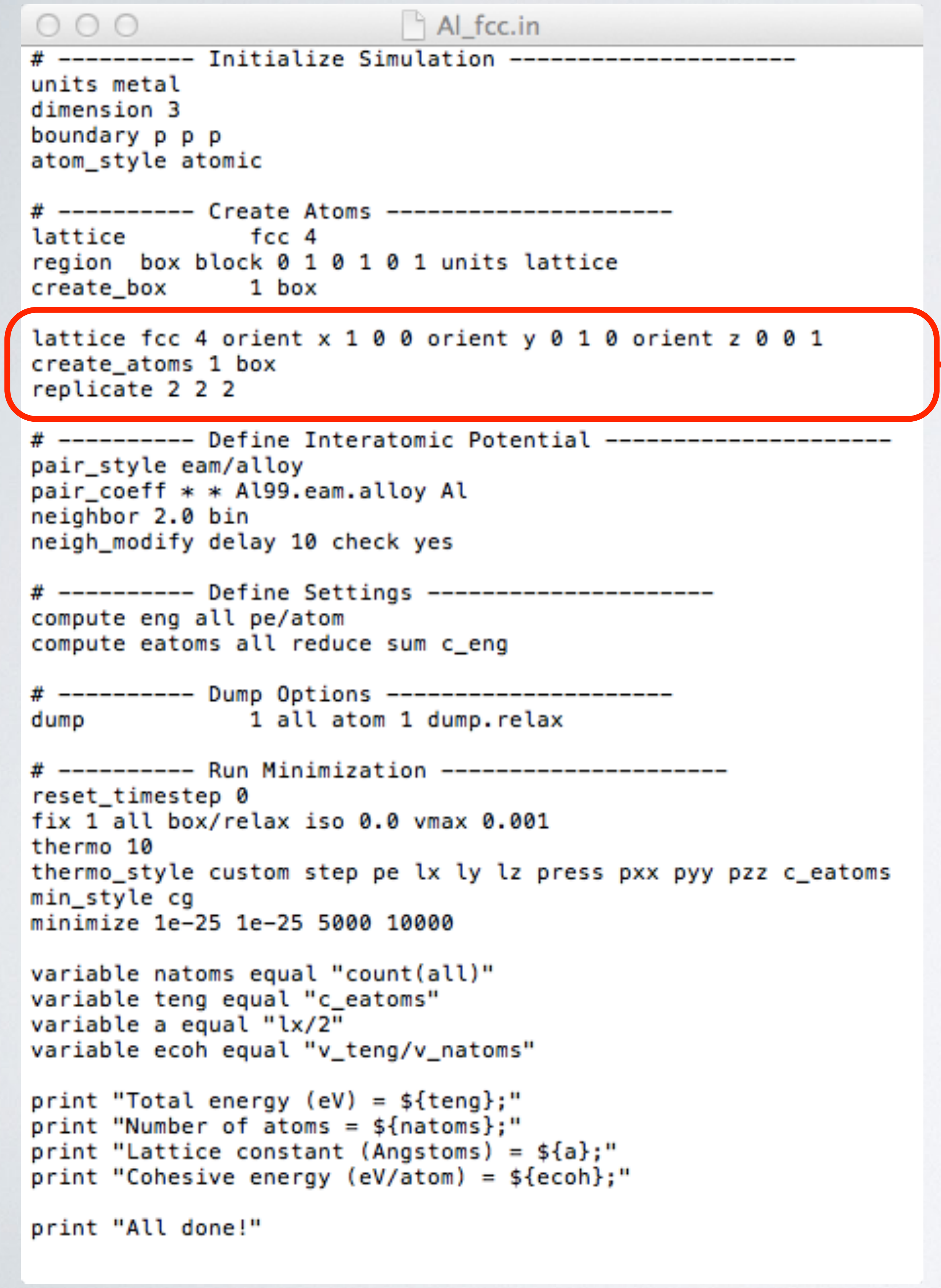
Tutorial 1: Al cohesive energy



```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region  box block 0 1 0 1 0 1 units lattice
create_box      1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

- Specify **fcc lattice** with **a**=4 Å
- Define **cuboidal block** labeled **box** holding **one lattice cell**
- Create **box** with **1** atom type

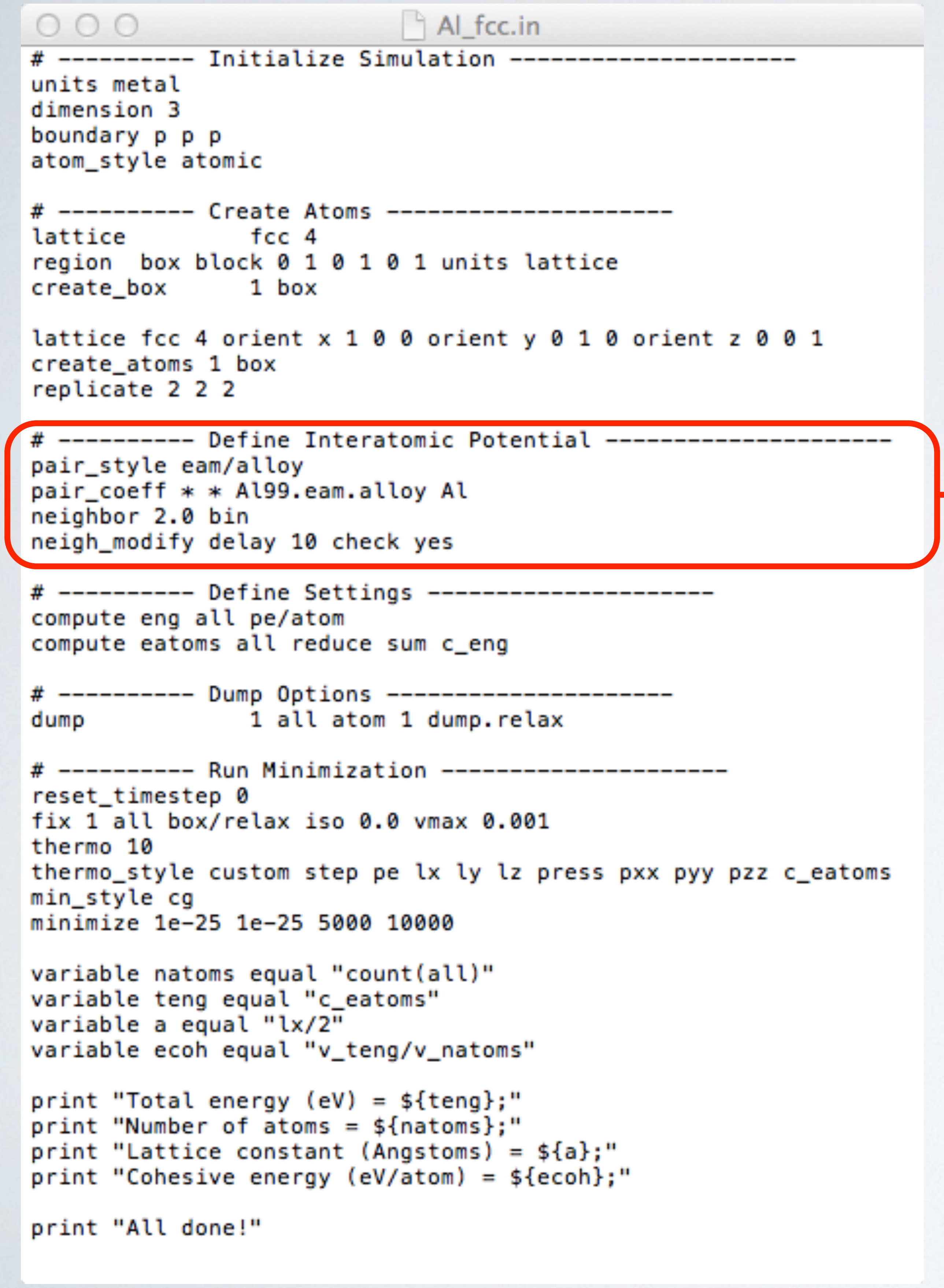
Tutorial 1: Al cohesive energy



```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region  box block 0 1 0 1 0 1 units lattice
create_box      1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

- Specify fcc lattice **orientation**
- Create atoms of type **1** on lattice sites within **box**
- **Replicate domain** by **2x2x2** in x,y,z [`replicate 1 1 1` would be more parsimonious for this trivially periodic system]

Tutorial 1: Al cohesive energy



```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region   box block 0 1 0 1 0 1 units lattice
create_box       1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

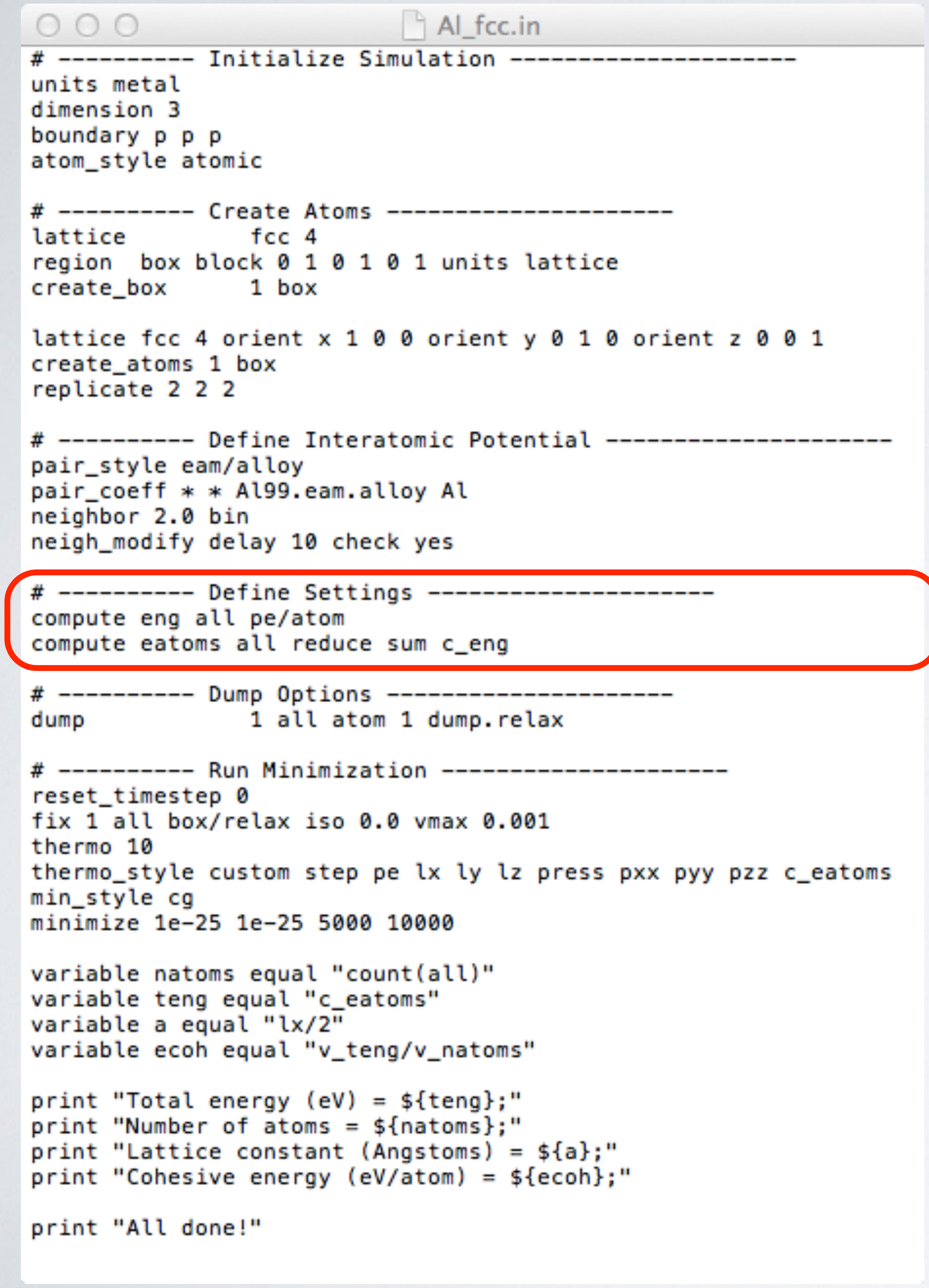
- Define form of pairwise interaction potential as **eam/alloy** [misnomer, EAM is n-body]

- Use **Al** block of **Al99.eam.alloy** - specifies cutoff, F, ρ, and Φ - for all pairs [for one atom type, **1 1** fine]

- **2 Å skin thickness** for **neighbor list binning**

- Build neighbor list every **10 steps**, but **check** atom moved more than half skin thickness

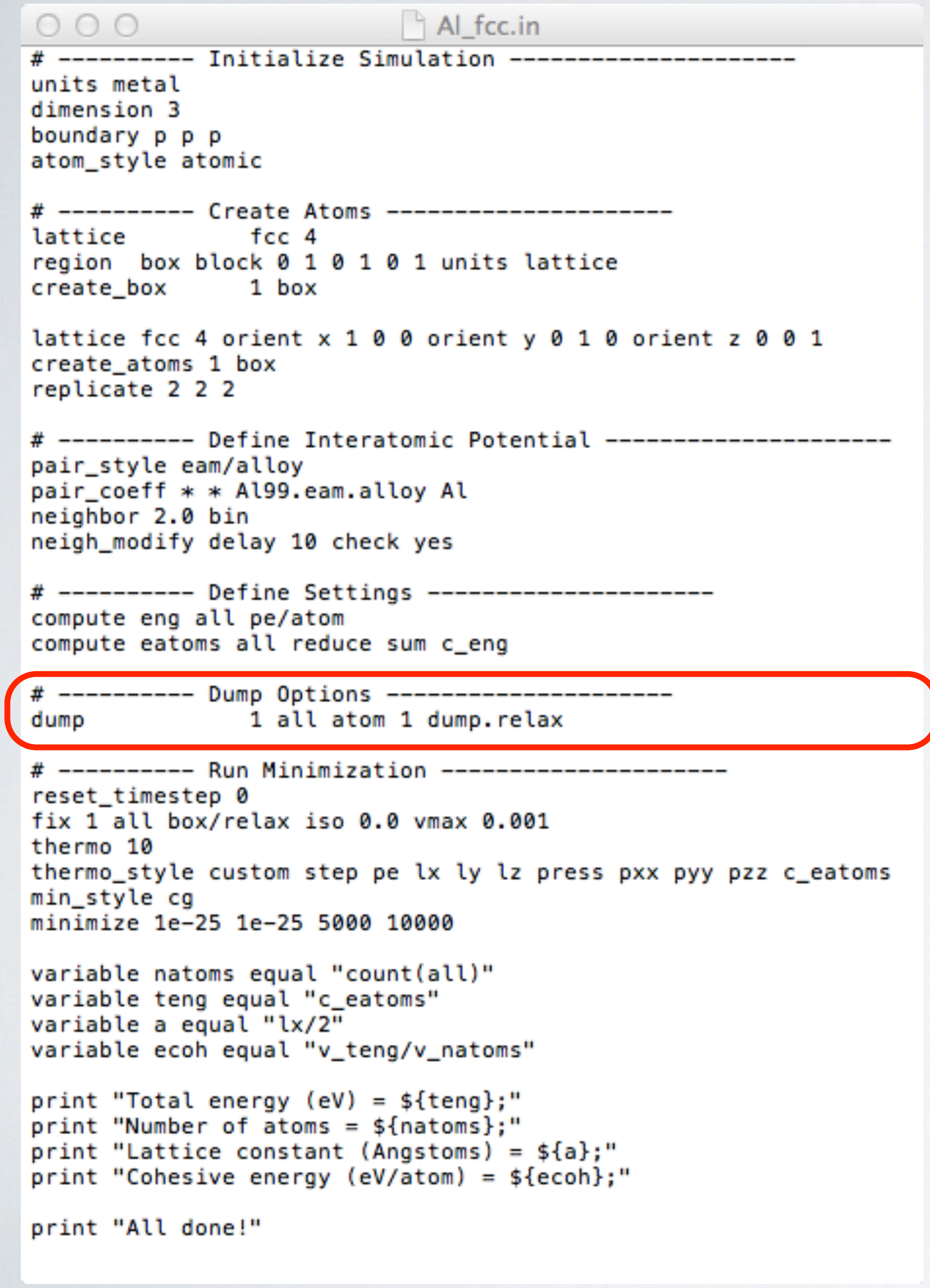
Tutorial 1: Al cohesive energy


Al_fcc.in

```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region  box block 0 1 0 1 0 1 units lattice
create_box      1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

- Define **computes** - quantities recalculated every time step [cf. **variables**, which evaluate a formula when called]
- Reference computes as **c_<name>**
- **c_eng** defined over **all** atoms to compute **potential energy per atom**
- **c_eatoms** performs **sum reduce** of c_eng vector over **all** atoms [alternatively: `compute eatoms all pe`]

Tutorial 1: Al cohesive energy



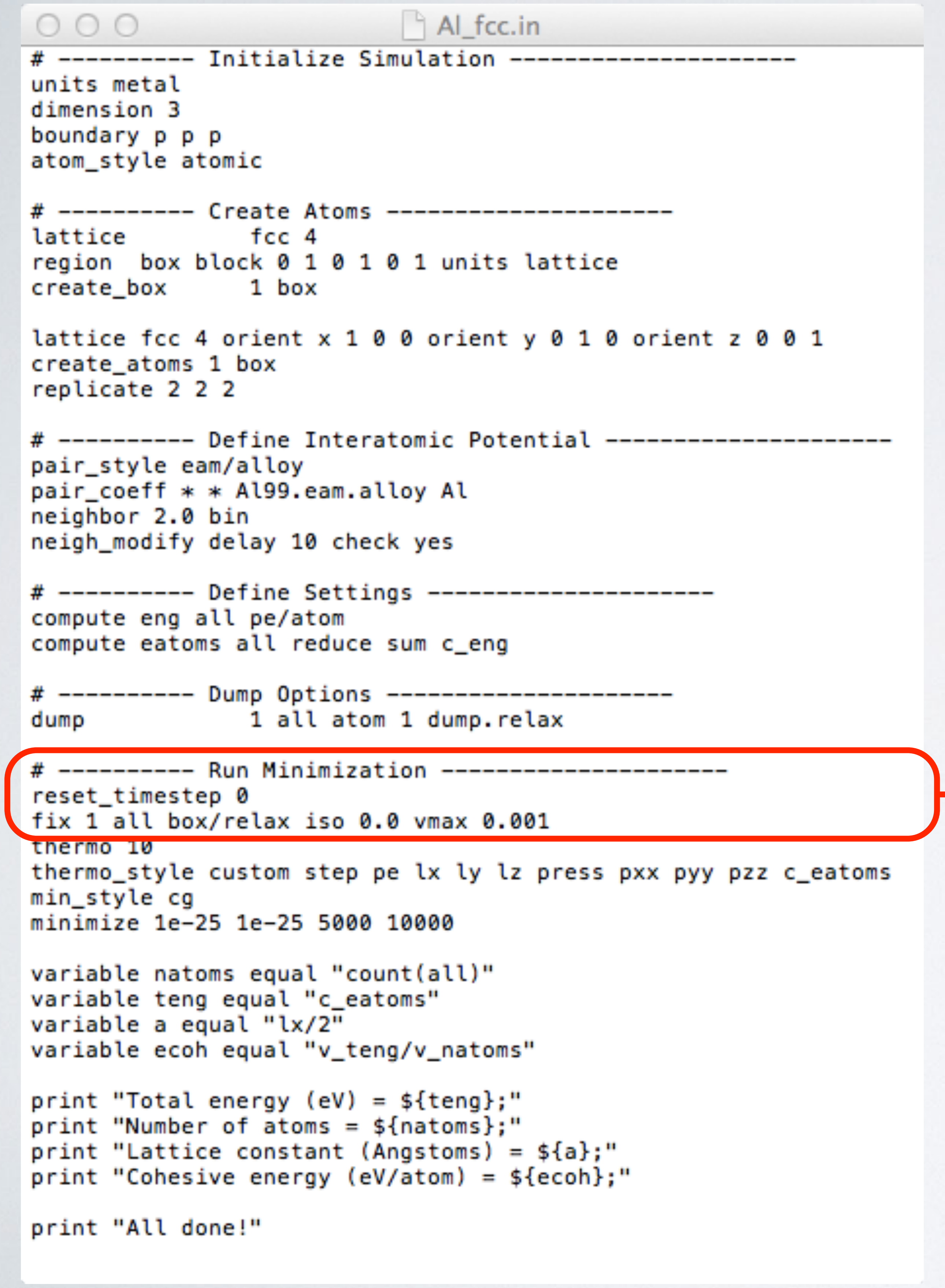
Al_fcc.in

```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region  box block 0 1 0 1 0 1 units lattice
create_box      1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

- A **dump** specifies how to write output data
- Tag dump with id **1** to write to **dump.relax** every **1** steps the coords of **all** of the **atoms**
- Dump format:

```
ITEM: TIMESTEP
0
ITEM: NUMBER OF ATOMS
32
ITEM: BOX BOUNDS pp pp pp
0 8
0 8
0 8
ITEM: ATOMS id type xs ys zs
1 1 0 0 0
2 1 0.25 0.25 0
3 1 0.25 0 0.25
4 1 0 0.25 0.25
```

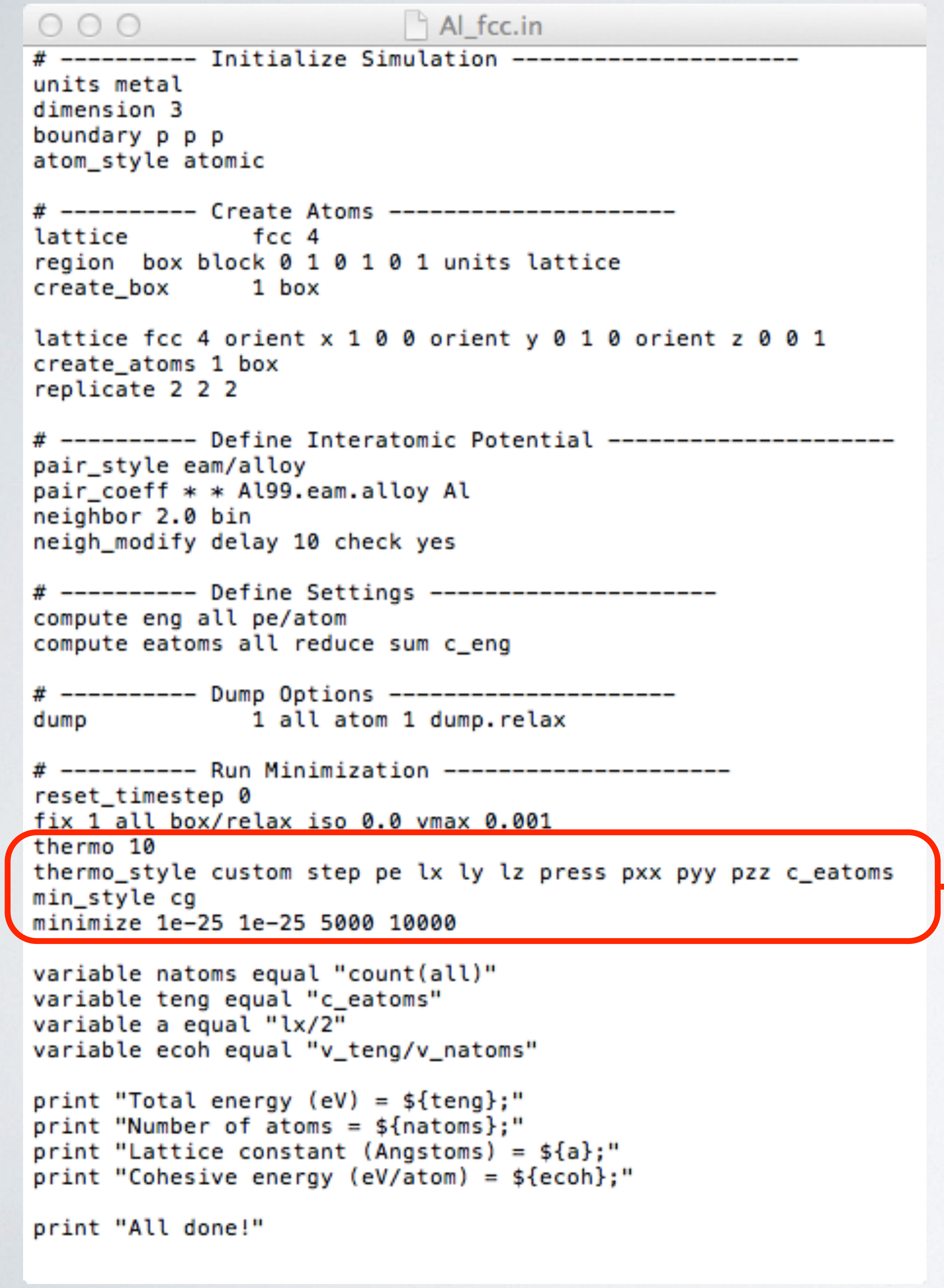
Tutorial 1: Al cohesive energy



```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region  box block 0 1 0 1 0 1 units lattice
create_box      1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

- Reset time steps to **0**
- A **fix** is an operation applied at every time step
- Define fix **1** operating on **all** atoms **relaxes box** to an external **isotropic pressure** of **0.0 bar** with a **0.1% maximum fractional volume change per step**

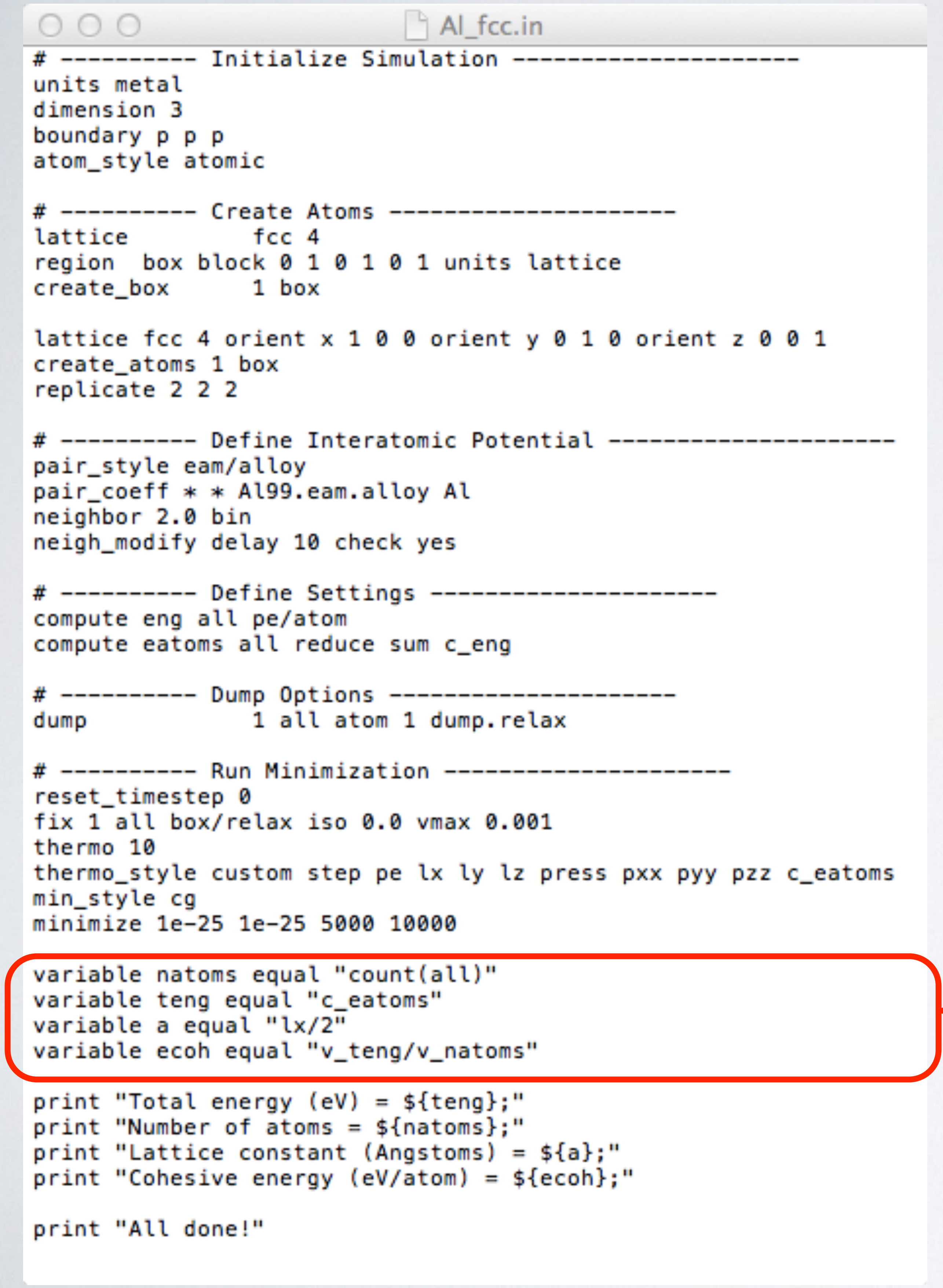
Tutorial 1: Al cohesive energy



```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region   box block 0 1 0 1 0 1 units lattice
create_box       1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

- Output **thermodynamic info** to screen every **10** steps [use **fix** / **dump** for file write]
- Customize thermo output
- Perform energy minimization by **conjugate gradient**
- **Minimize** $E = E_{FF} + E_{fix}$ with $\Delta E=10^{-25}$ (i.e., 1 part in 10^{25}) and $\Delta f=10^{-25}$, and a maximum of 5000 iterations and 10000 energy evaluations

Tutorial 1: Al cohesive energy


Al_fcc.in

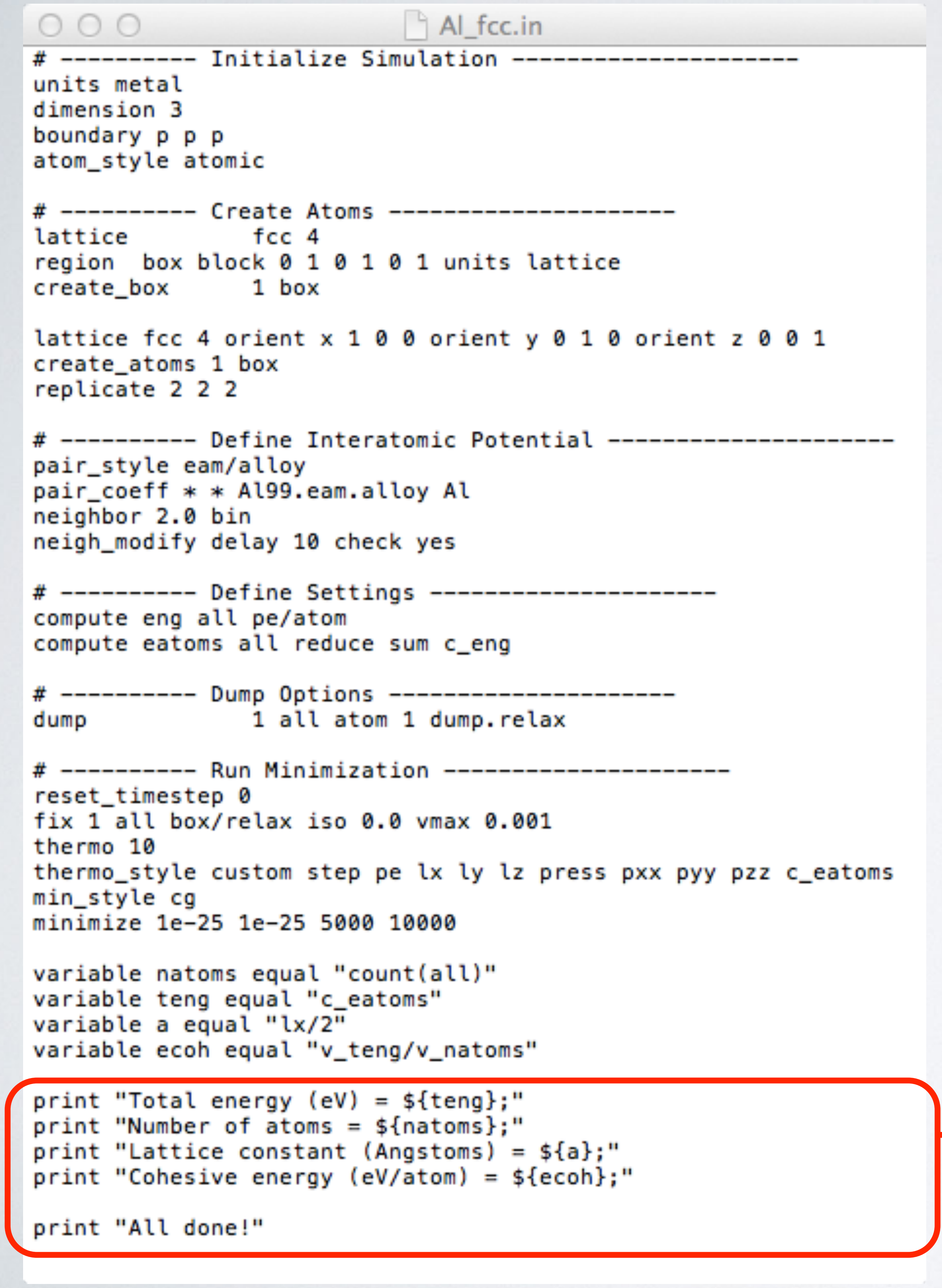
```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region  box block 0 1 0 1 0 1 units lattice
create_box      1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

- Define **variables** as formulas evaluated when called [cf. **computes**, simulation values recomputed each step]

- Reference variables as **v_<name>**

- natoms = # atoms
  teng = total PE (c_eatoms)
  a = lattice parameter
  (box side in x divided by
  # x replicas = 2)
  ecoh = cohesive energy /atom

Tutorial 1: Al cohesive energy

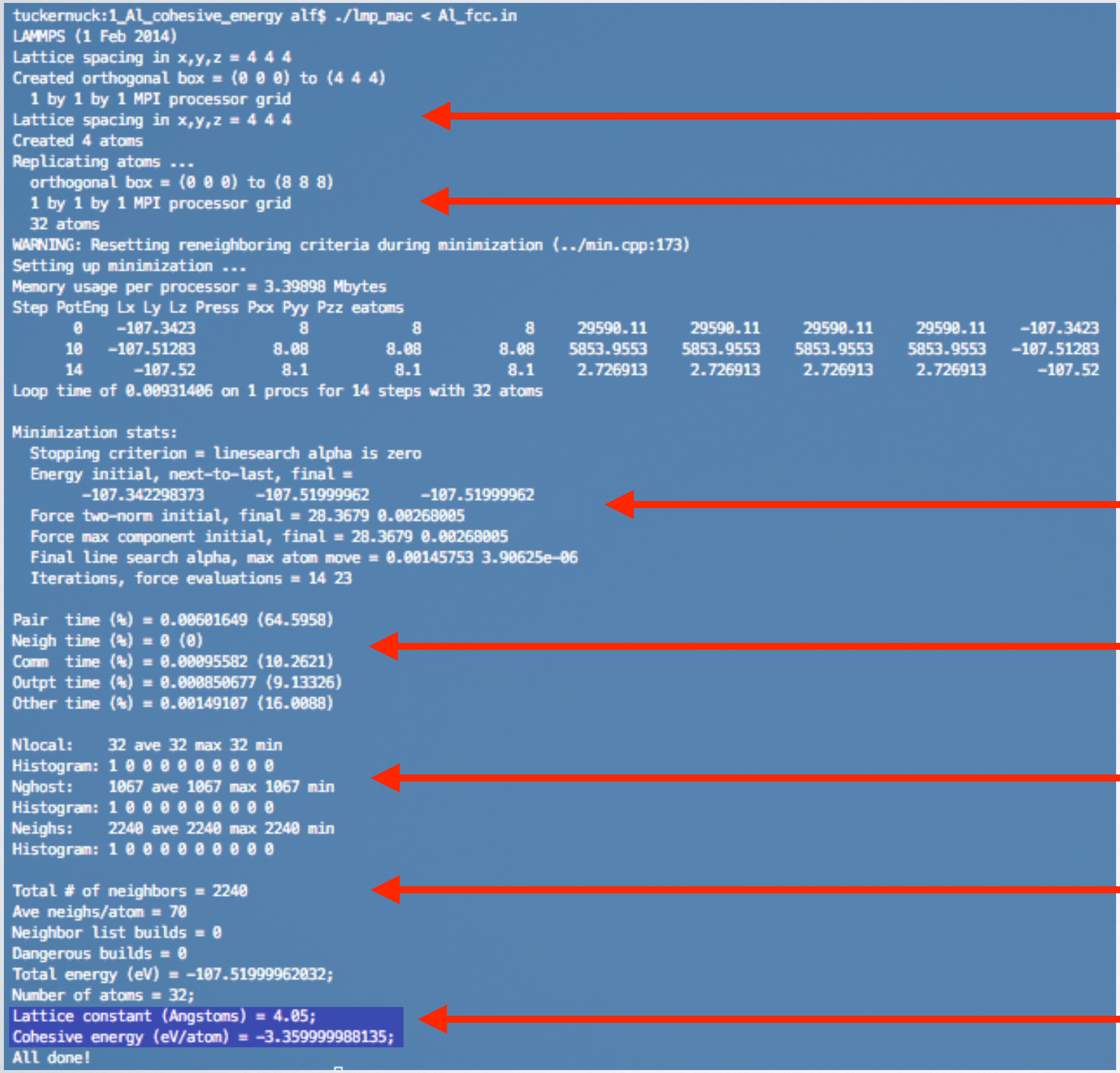


```
# ---------- Initialize Simulation ---------------------
units metal
dimension 3
boundary p p p
atom_style atomic

# ---------- Create Atoms ---------------------
lattice         fcc 4
region  box block 0 1 0 1 0 1 units lattice
create_box      1 box

lattice fcc 4 orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms 1 box
replicate 2 2 2

# ---------- Define Interatomic Potential ---------------------
pair_style eam/alloy
pair_coeff * * Al99.eam.alloy Al
neighbor 2.0 bin
neigh_modify delay 10 check yes

# ---------- Define Settings ---------------------
compute eng all pe/atom
compute eatoms all reduce sum c_eng

# ---------- Dump Options ---------------------
dump            1 all atom 1 dump.relax

# ---------- Run Minimization ---------------------
reset_timestep 0
fix 1 all box/relax iso 0.0 vmax 0.001
thermo 10
thermo_style custom step pe lx ly lz press pxx pyy pzz c_eatoms
min_style cg
minimize 1e-25 1e-25 5000 10000

variable natoms equal "count(all)"
variable teng equal "c_eatoms"
variable a equal "lx/2"
variable ecoh equal "v_teng/v_natoms"

print "Total energy (eV) = ${teng};"
print "Number of atoms = ${natoms};"
print "Lattice constant (Angstoms) = ${a};"
print "Cohesive energy (eV/atom) = ${ecoh};"

print "All done!"
```

- Print terminal output to screen

Tutorial 1: Al cohesive energy

3. Let's run!   `lmp_serial < Al_fcc.in`



```
tuckernuck:1_Al_cohesive_energy alf$ ./lmp_mac < Al_fcc.in
LAMMPS (1 Feb 2014)
Lattice spacing in x,y,z = 4 4 4
Created orthogonal box = (0 0 0) to (4 4 4)
  1 by 1 by 1 MPI processor grid
Lattice spacing in x,y,z = 4 4 4
Created 4 atoms
Replicating atoms ...
  orthogonal box = (0 0 0) to (8 8 8)
  1 by 1 by 1 MPI processor grid
  32 atoms
WARNING: Resetting reneighboring criteria during minimization (../min.cpp:173)
Setting up minimization ...
Memory usage per processor = 3.39898 Mbytes
Step PotEng Lx Ly Lz Press Pxx Pyy Pzz eatoms
       0    -107.3423            8            8            8     29590.11     29590.11     29590.11     29590.11    -107.3423
      10   -107.51283         8.08         8.08         8.08    5853.9553    5853.9553    5853.9553    5853.9553   -107.51283
      14      -107.52          8.1          8.1          8.1     2.726913     2.726913     2.726913     2.726913      -107.52
Loop time of 0.00931406 on 1 procs for 14 steps with 32 atoms

Minimization stats:
  Stopping criterion = linesearch alpha is zero
  Energy initial, next-to-last, final =
        -107.342298373     -107.51999962     -107.51999962
  Force two-norm initial, final = 28.3679 0.00268005
  Force max component initial, final = 28.3679 0.00268005
  Final line search alpha, max atom move = 0.00145753 3.90625e-06
  Iterations, force evaluations = 14 23

Pair  time (%) = 0.00601649 (64.5958)
Neigh time (%) = 0 (0)
Comm  time (%) = 0.00095582 (10.2621)
Outpt time (%) = 0.000850677 (9.13326)
Other time (%) = 0.00149107 (16.0088)

Nlocal:    32 ave 32 max 32 min
Histogram: 1 0 0 0 0 0 0 0 0 0
Nghost:    1067 ave 1067 max 1067 min
Histogram: 1 0 0 0 0 0 0 0 0 0
Neighs:    2240 ave 2240 max 2240 min
Histogram: 1 0 0 0 0 0 0 0 0 0

Total # of neighbors = 2240
Ave neighs/atom = 70
Neighbor list builds = 0
Dangerous builds = 0
Total energy (eV) = -107.51999962032;
Number of atoms = 32;
Lattice constant (Angstoms) = 4.05;
Cohesive energy (eV/atom) = -3.359999988135;
All done!
```

building system

serial run

thermo

minimization stopping criteria

CPU accounting

atom accounting

neighbor accounting (dangerous builds)

terminal print

Tutorial 1: Al cohesive energy

4. Analysis

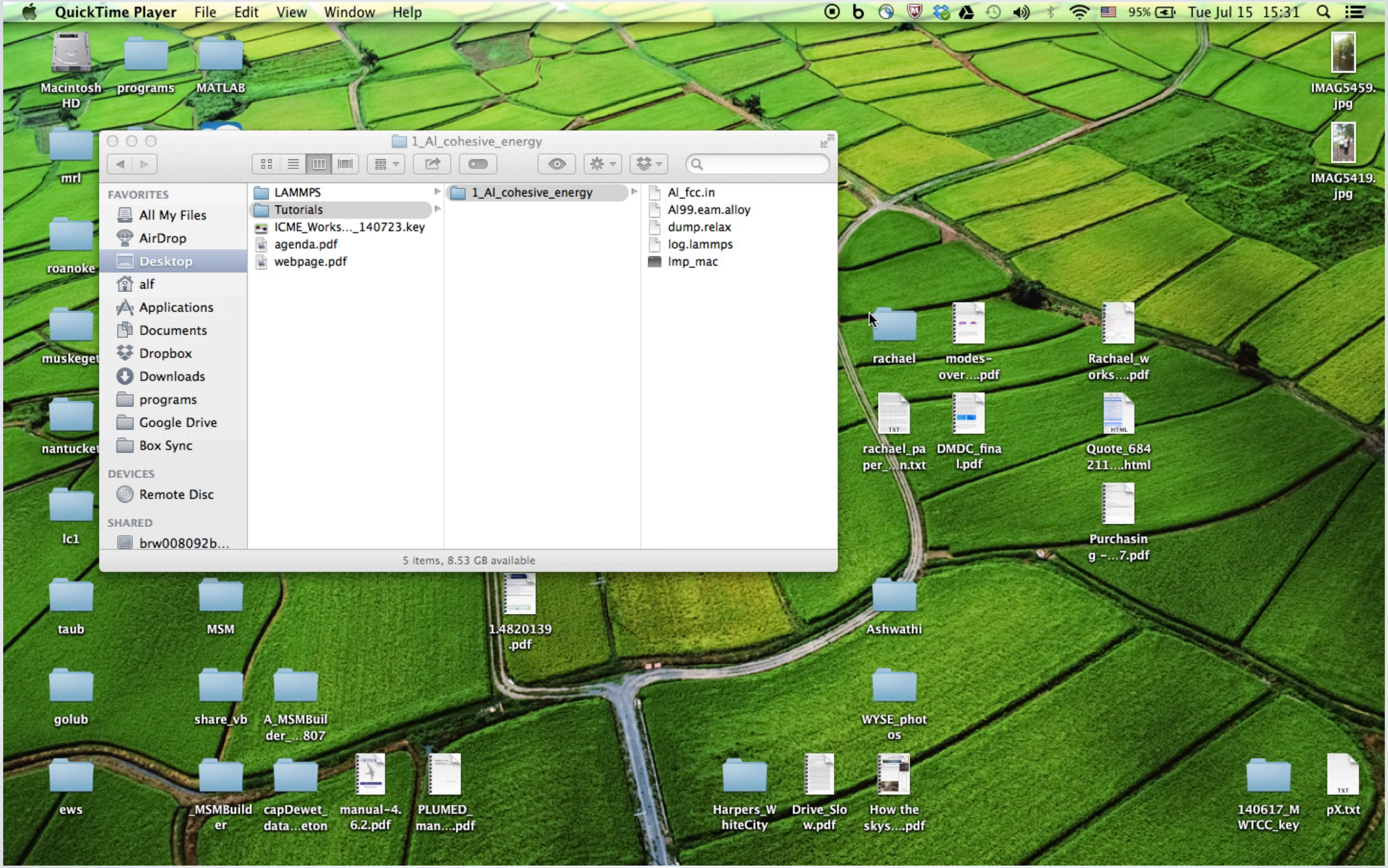
	LAMMPS	Expt.	
Lattice constant / Å	4.05	4.0495	*
Cohesive energy / eV/atom	-3.36	-3.39	*

- We should be shocked if these quantities did **not** agree — EAM FF parametrized wrt experimental data
- **Q.** What about if we were studying a new material with experimentally unknown $\mathbf{E_{cohe}}$ and $\mathbf{a}$?

*Charles Kittel. Introduction to Solid State Physics, 8th edition. Hoboken, NJ: John Wiley & Sons, Inc, 2005. *http://periodictable.com/Elements/013/data.html

Tutorial 1: Al cohesive energy

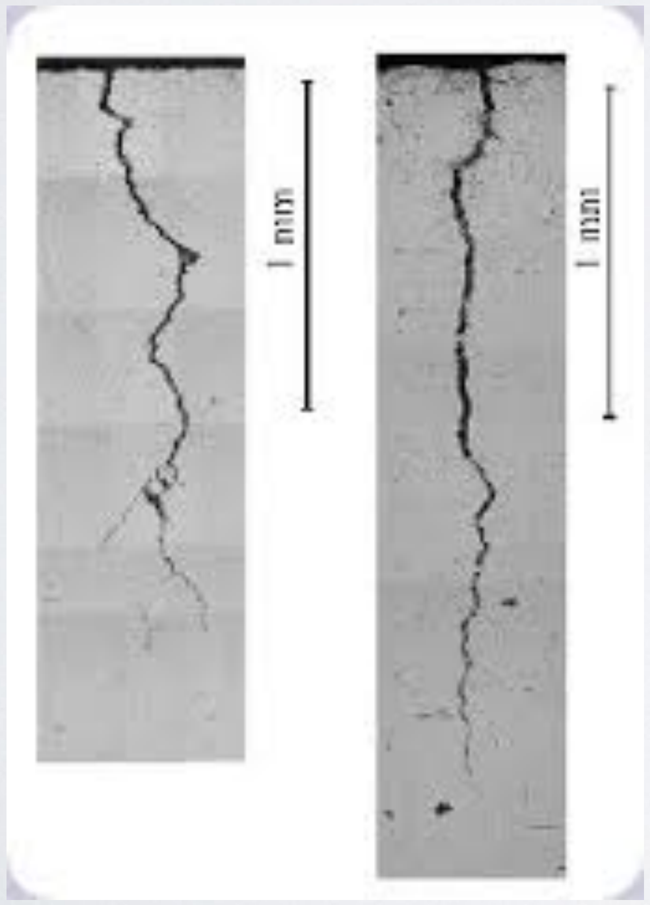
5. Visualization in OVITO

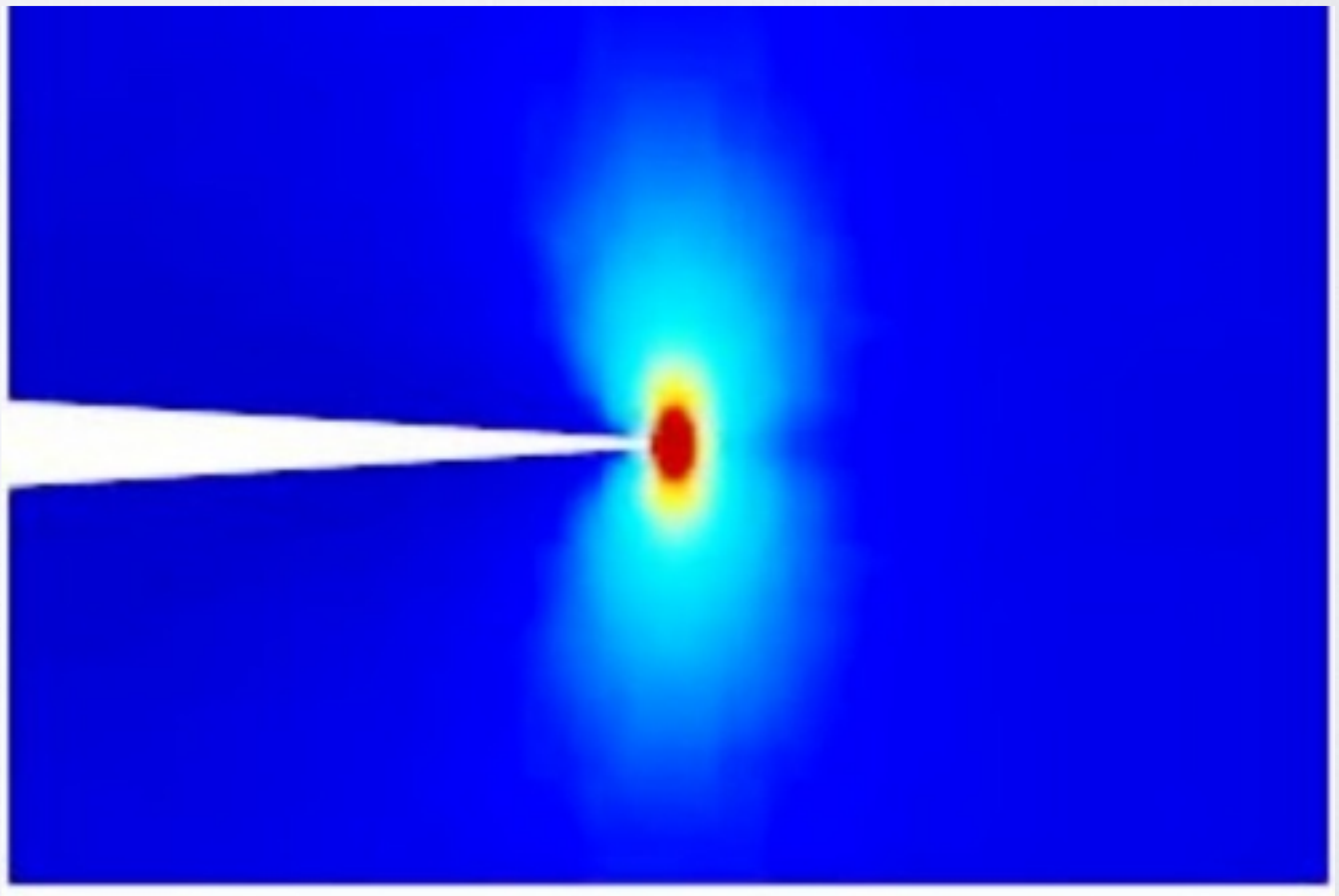


Tutorial II: Al crack propagation

Tutorial II: Al crack propagation

- OK, now to the real MD simulations!
- An important mode of materials failure is propagation of exterior cracks
- The stress field at the crack tip in an amorphous material can be modeled by continuum equations or FEM

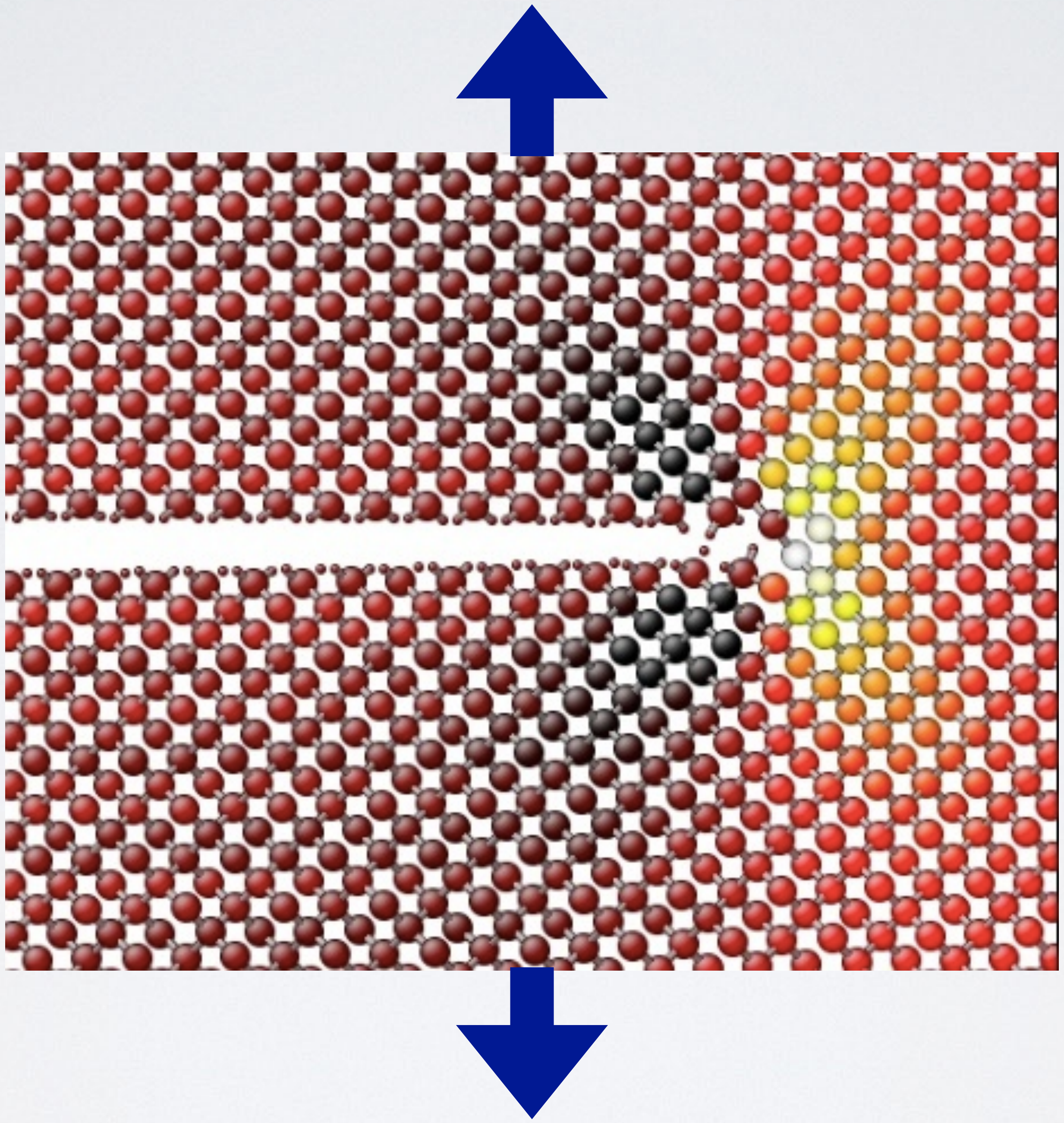




https://advanceseng.com/mechanical-engineering/dynamic-steady-state-analysis-of-crack-propagation-in-rubber-like-solids-using-an-extended-finite-element-method/

Tutorial II: Al crack propagation

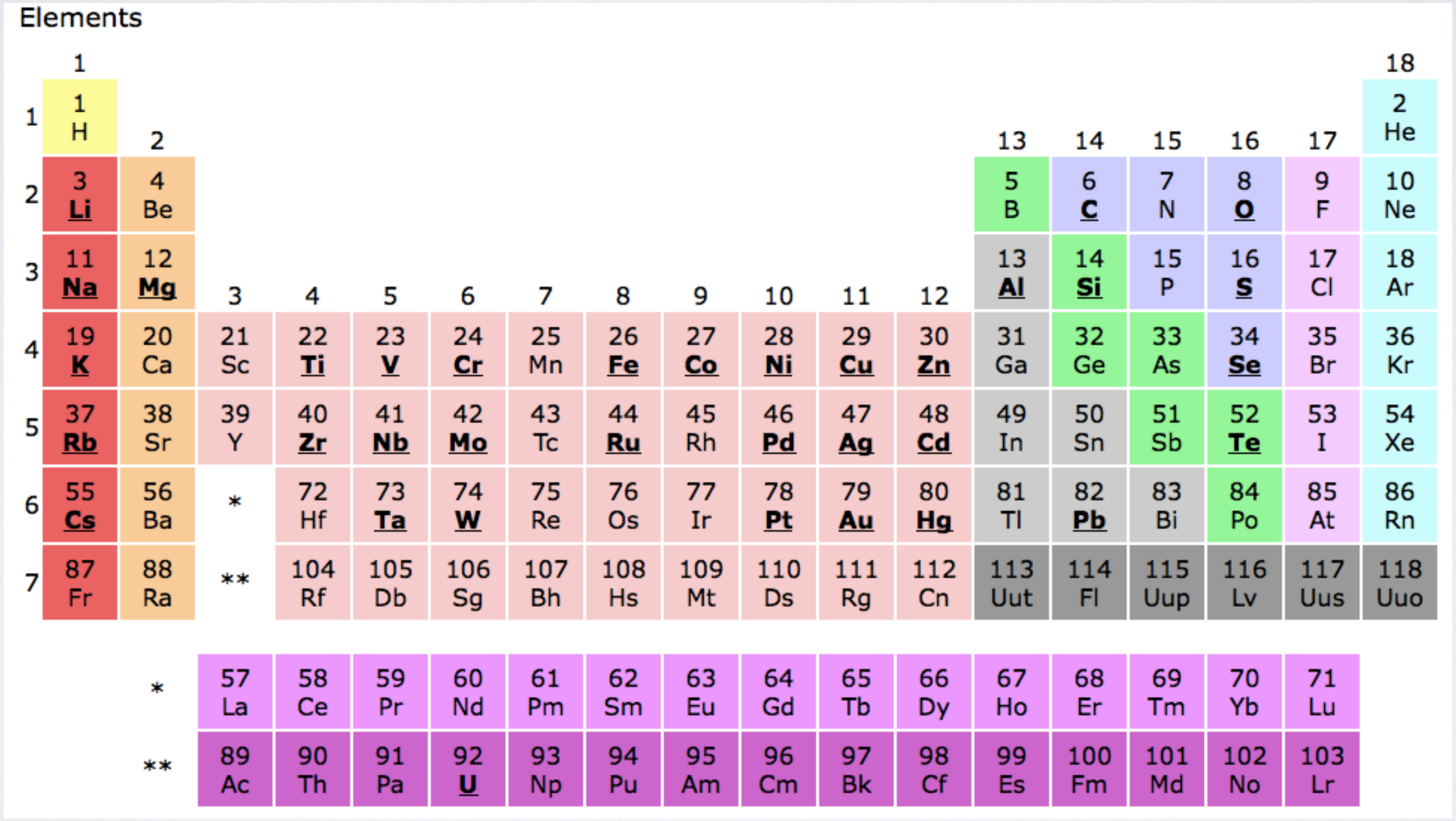
- Using MD simulation, we can visualize the stress field near the crack tip with **atomistic resolution**



- **Strategy:** Construct an exterior crack in a semi-periodic fcc Al xtal and measure atomic stress upon deformation

Tutorial II: Al crack propagation

1. Download **Al99.eam.alloy** EAM potential from NIST Interatomic Potentials Repository Project (http://www.ctcms.nist.gov/potentials)



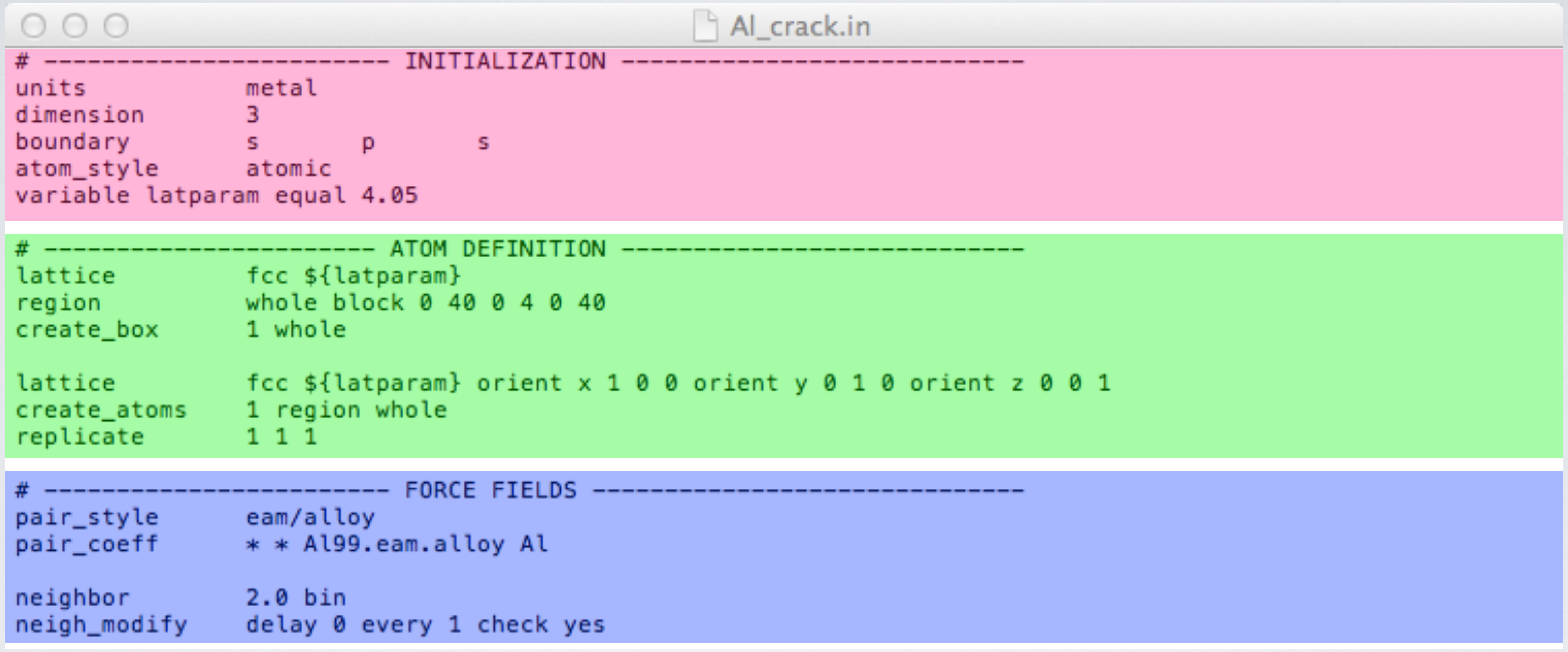
Elements

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1 H																	2 He
2	3 **Li**	4 Be											5 B	6 **C**	7 N	8 **O**	9 F	10 Ne
3	11 **Na**	12 **Mg**											13 **Al**	14 **Si**	15 P	16 **S**	17 Cl	18 Ar
4	19 **K**	20 Ca	21 Sc	22 **Ti**	23 **V**	24 **Cr**	25 Mn	26 **Fe**	27 **Co**	28 **Ni**	29 **Cu**	30 **Zn**	31 Ga	32 Ge	33 As	34 **Se**	35 Br	36 Kr
5	37 **Rb**	38 Sr	39 Y	40 **Zr**	41 **Nb**	42 **Mo**	43 Tc	44 **Ru**	45 Rh	46 **Pd**	47 **Ag**	48 **Cd**	49 In	50 Sn	51 Sb	52 **Te**	53 I	54 Xe
6	55 **Cs**	56 Ba	*	72 Hf	73 **Ta**	74 **W**	75 Re	76 Os	77 Ir	78 **Pt**	79 **Au**	80 **Hg**	81 Tl	82 **Pb**	83 Bi	84 Po	85 At	86 Rn
7	87 Fr	88 Ra	**	104 Rf	105 Db	106 Sg	107 Bh	108 Hs	109 Mt	110 Ds	111 Rg	112 Cn	113 Uut	114 Fl	115 Uup	116 Lv	117 Uus	118 Uuo
		*	57 La	58 Ce	59 Pr	60 Nd	61 Pm	62 Sm	63 Eu	64 Gd	65 Tb	66 Dy	67 Ho	68 Er	69 Tm	70 Yb	71 Lu	
		**	89 Ac	90 Th	91 Pa	92 **U**	93 Np	94 Pu	95 Am	96 Cm	97 Bk	98 Cf	99 Es	100 Fm	101 Md	102 No	103 Lr	

Tutorial II: Al crack propagation

2. Copy `Al_crack.in`, `Al_eq.m`, and `Al_crack.m` from `/class/mse404pla/LAMMPS/`

Tutorial II: Al crack propagation



```
# ------------------------ INITIALIZATION ----------------------------
units           metal
dimension       3
boundary        s       p       s
atom_style      atomic
variable latparam equal 4.05

# ----------------------- ATOM DEFINITION ----------------------------
lattice         fcc ${latparam}
region          whole block 0 40 0 4 0 40
create_box      1 whole

lattice         fcc ${latparam} orient x 1 0 0 orient y 0 1 0 orient z 0 0 1
create_atoms    1 region whole
replicate       1 1 1

# ------------------------ FORCE FIELDS ------------------------------
pair_style      eam/alloy
pair_coeff      * * Al99.eam.alloy Al

neighbor        2.0 bin
neigh_modify    delay 0 every 1 check yes
```

- Initializing system
- Building geometry
- Defining force field

Tutorial II: Al crack propagation

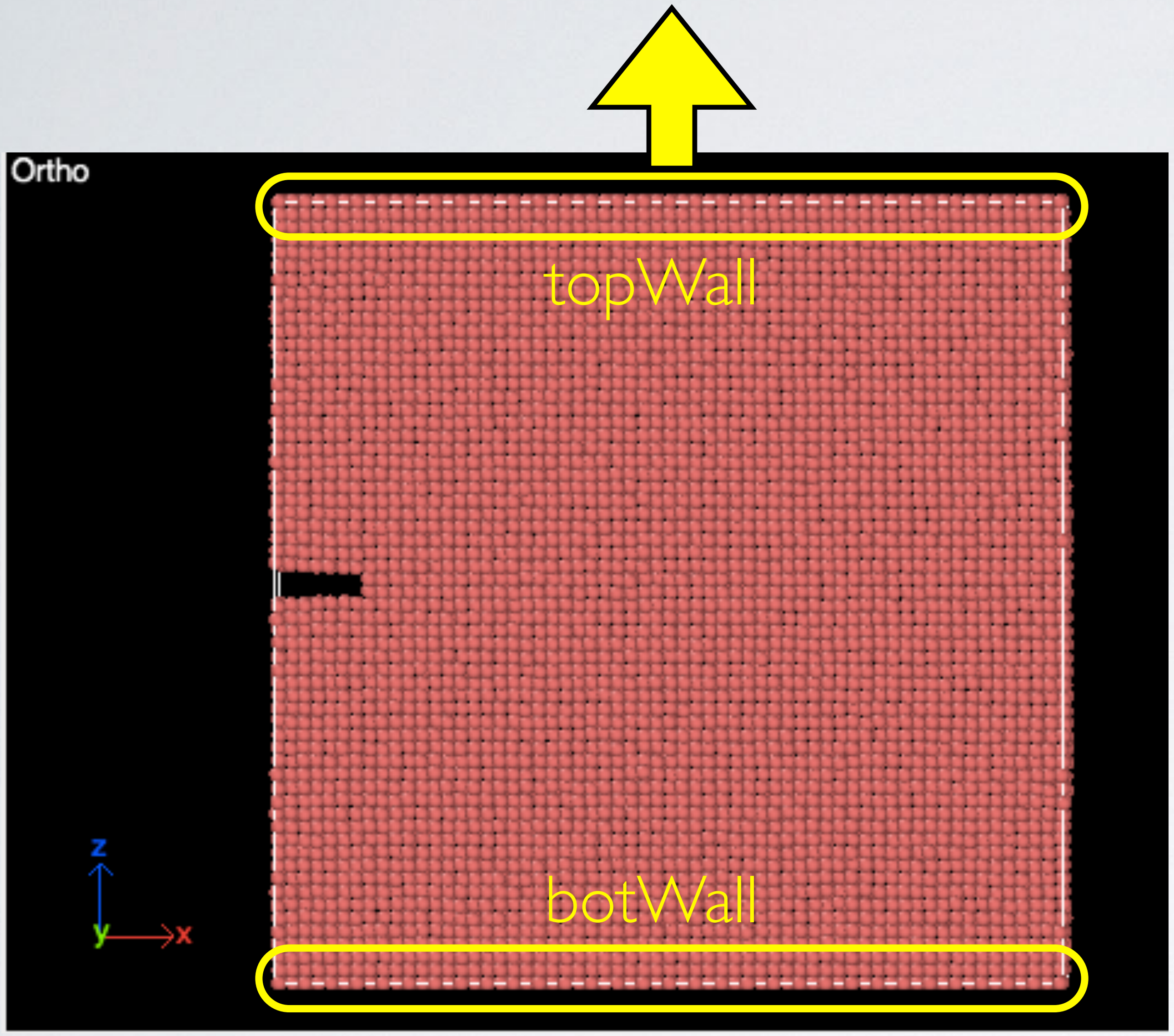
```
# ------------------------- GROUPS --------------------------------
variable DX equal ${latparam}*(1/2+1/24)
variable DY equal ${latparam}*(1/2+1/24)
variable DZ equal ${latparam}*(1/2+1/24)

variable tmp equal "xlo"
variable XLO equal ${tmp}
variable tmp equal "xhi"
variable XHI equal ${tmp}
variable tmp equal "ylo"
variable YLO equal ${tmp}
variable tmp equal "yhi"
variable YHI equal ${tmp}
variable tmp equal "zlo"
variable ZLO equal ${tmp}
variable tmp equal "zhi"
variable ZHI equal ${tmp}

variable maxX equal "v_XLO + v_DX"
variable minX equal "v_XHI - v_DX"
variable maxY equal "v_YLO + v_DY"
variable minY equal "v_YHI - v_DY"
variable maxZ equal "v_ZLO + v_DZ"
variable minZ equal "v_ZHI - v_DZ"

region          topWall block INF INF INF INF ${minZ} INF units box
region          botWall block INF INF INF INF INF ${maxZ} units box
group           topWall region topWall
group           botWall region botWall

group           boundary union topWall botWall
group           mobile subtract all boundary
```

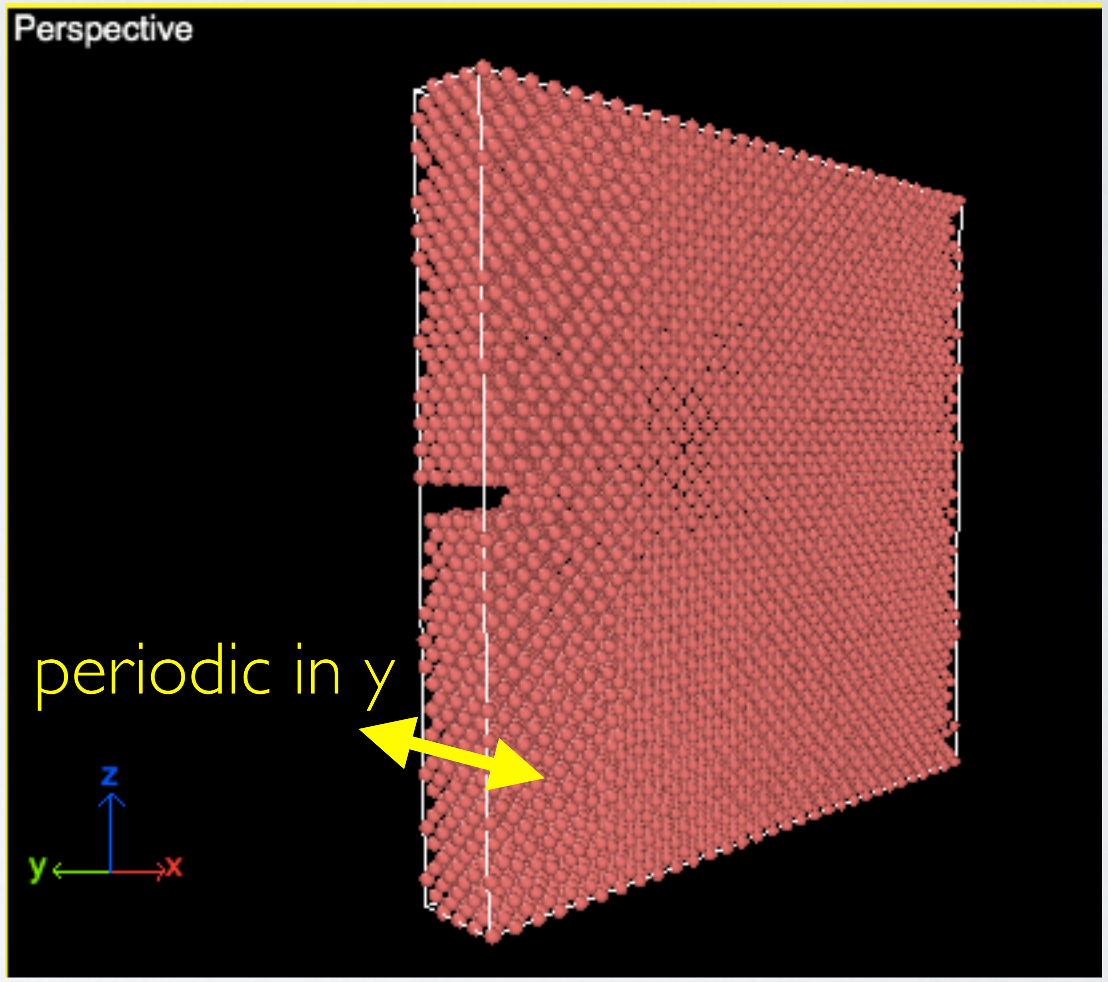
```
# ------------------------- CRACK -------------------------------
variable ZHI_crack equal "0.5*(v_ZHI-v_ZLO) + v_ZLO + 0.75*v_latparam"
variable ZLO_crack equal "0.5*(v_ZHI-v_ZLO) + v_ZLO - 0.25*v_latparam"
#variable XHI_crack equal "(1/16)*(v_XHI-v_XLO)"
variable XHI_crack equal "3*v_latparam"
region          void block INF ${XHI_crack} INF INF ${ZLO_crack} ${ZHI_crack} units box
delete_atoms    region void
```

```
# ------------------------- SETTINGS --------------------------------
compute         csym all centro/atom fcc
compute         eng all pe/atom
compute         atomStress all stress/atom virial
```

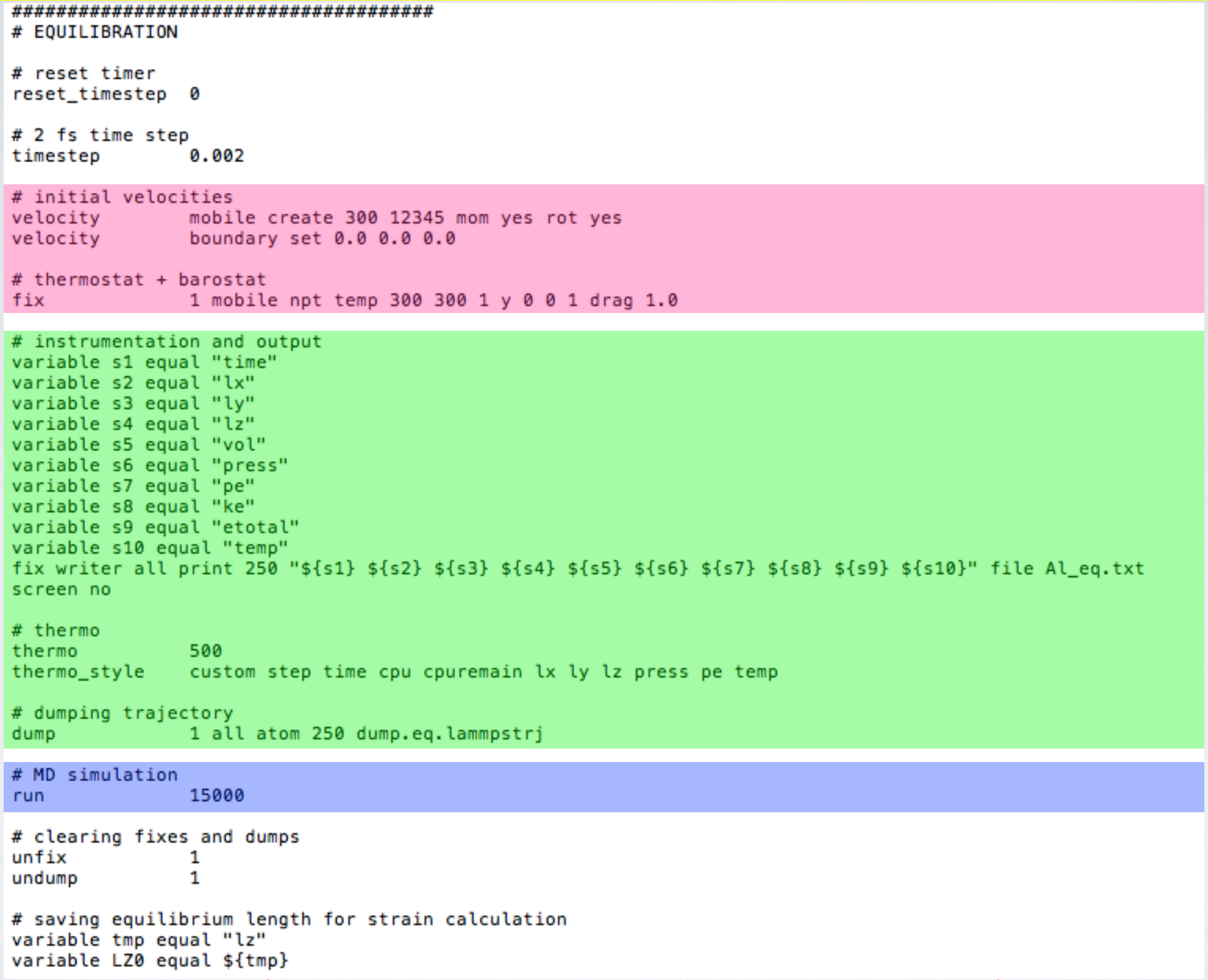
- Defining groups: topWall, bottomWall, and mobile
- Eliminating a notch of atoms to form the crack
- Specifying computes (incl. per atom stress tensor)

Tutorial II: Al crack propagation





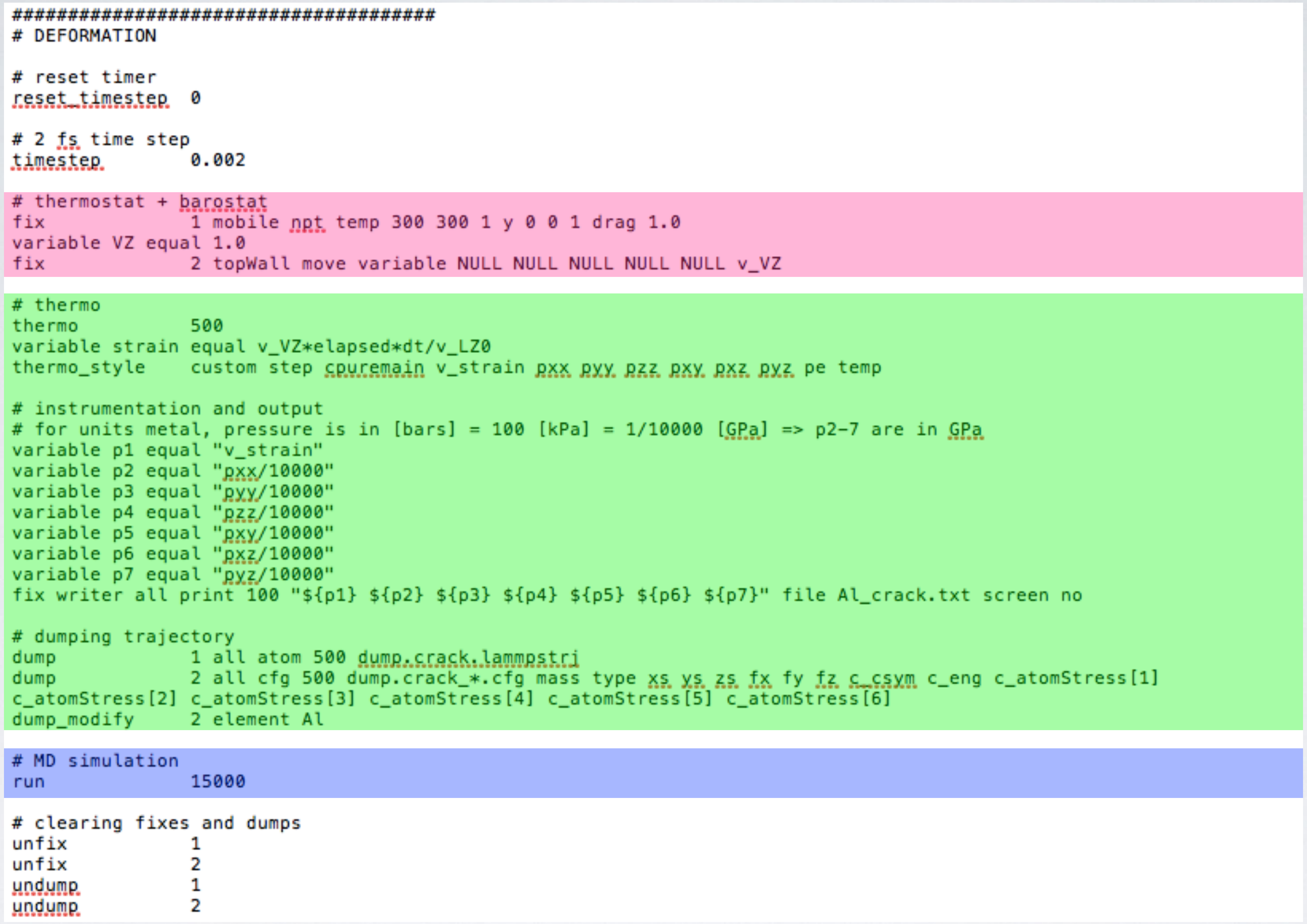
Tutorial II: Al crack propagation



```
##########################################
# EQUILIBRATION

# reset timer
reset_timestep  0

# 2 fs time step
timestep        0.002

# initial velocities
velocity        mobile create 300 12345 mom yes rot yes
velocity        boundary set 0.0 0.0 0.0

# thermostat + barostat
fix             1 mobile npt temp 300 300 1 y 0 0 1 drag 1.0

# instrumentation and output
variable s1 equal "time"
variable s2 equal "lx"
variable s3 equal "ly"
variable s4 equal "lz"
variable s5 equal "vol"
variable s6 equal "press"
variable s7 equal "pe"
variable s8 equal "ke"
variable s9 equal "etotal"
variable s10 equal "temp"
fix writer all print 250 "${s1} ${s2} ${s3} ${s4} ${s5} ${s6} ${s7} ${s8} ${s9} ${s10}" file Al_eq.txt
screen no

# thermo
thermo          500
thermo_style    custom step time cpu cpuremain lx ly lz press pe temp

# dumping trajectory
dump            1 all atom 250 dump.eq.lammpstrj

# MD simulation
run             15000

# clearing fixes and dumps
unfix           1
undump          1

# saving equilibrium length for strain calculation
variable tmp equal "lz"
variable LZ0 equal ${tmp}
```

- NVT equilibration of mobile atoms, topWall & botWall frozen
- Instrumentation, thermo, and output
- MD simulation

Tutorial II: Al crack propagation



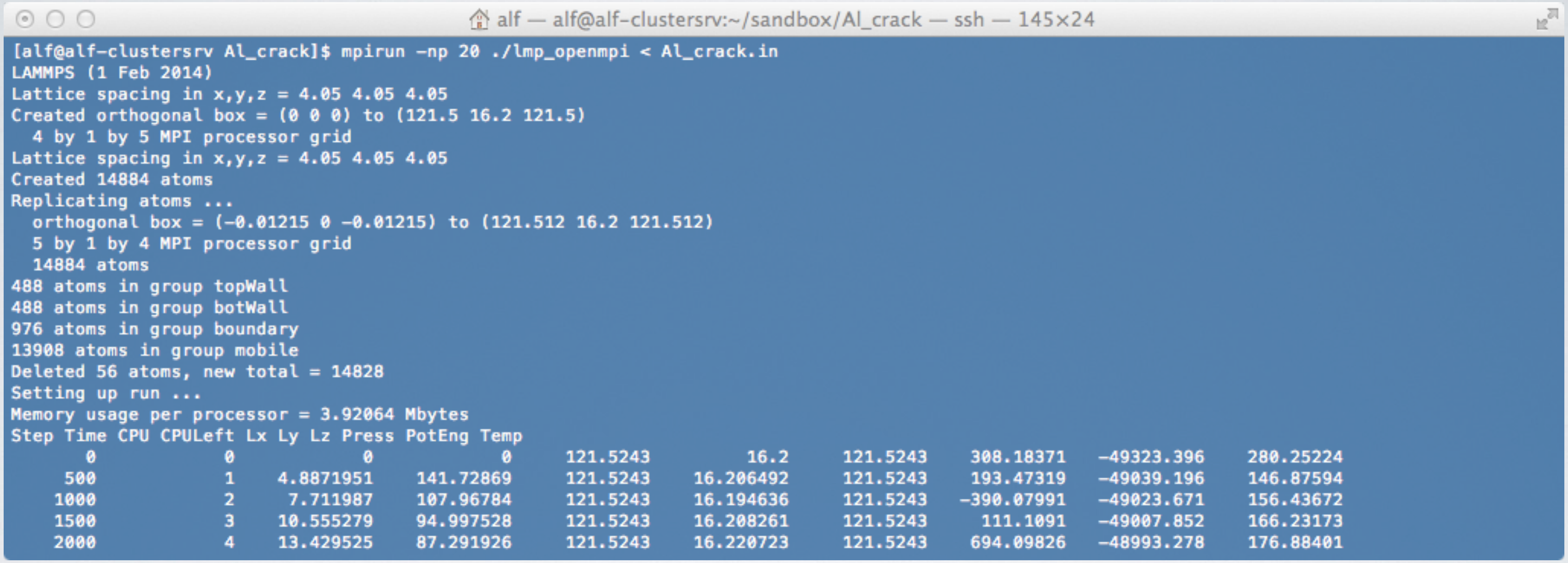
```
########################################
# DEFORMATION

# reset timer
reset_timestep  0

# 2 fs time step
timestep        0.002

# thermostat + barostat
fix             1 mobile npt temp 300 300 1 y 0 0 1 drag 1.0
variable VZ equal 1.0
fix             2 topWall move variable NULL NULL NULL NULL NULL v_VZ

# thermo
thermo          500
variable strain equal v_VZ*elapsed*dt/v_LZ0
thermo_style    custom step cpuremain v_strain pxx pyy pzz pxy pxz pyz pe temp

# instrumentation and output
# for units metal, pressure is in [bars] = 100 [kPa] = 1/10000 [GPa] => p2-7 are in GPa
variable p1 equal "v_strain"
variable p2 equal "pxx/10000"
variable p3 equal "pyy/10000"
variable p4 equal "pzz/10000"
variable p5 equal "pxy/10000"
variable p6 equal "pxz/10000"
variable p7 equal "pyz/10000"
fix writer all print 100 "${p1} ${p2} ${p3} ${p4} ${p5} ${p6} ${p7}" file Al_crack.txt screen no

# dumping trajectory
dump            1 all atom 500 dump.crack.lammpstrj
dump            2 all cfg 500 dump.crack_*.cfg mass type xs ys zs fx fy fz c_csym c_eng c_atomStress[1]
c_atomStress[2] c_atomStress[3] c_atomStress[4] c_atomStress[5] c_atomStress[6]
dump_modify     2 element Al

# MD simulation
run             15000

# clearing fixes and dumps
unfix           1
unfix           2
undump          1
undump          2
```

- NVT integration of mobile atoms, topWall pulled up, botWall frozen in place
- Instrumentation, thermo, and output (incl. custom strain calculation and atom stress cfg dump)

- MD simulation

Tutorial II: Al crack propagation

3. Let's run! `lmp_serial < Al_crack.in`



alf — alf@alf-clustersrv:~/sandbox/Al_crack — ssh — 145×24

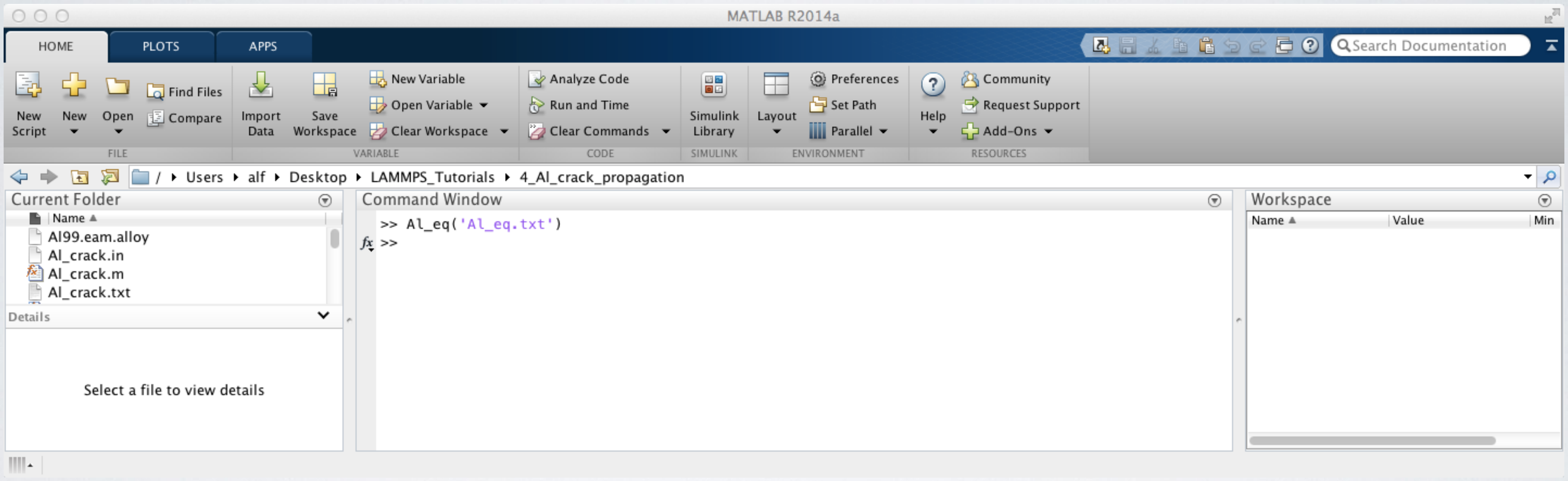
```
[alf@alf-clustersrv Al_crack]$ mpirun -np 20 ./lmp_openmpi < Al_crack.in
LAMMPS (1 Feb 2014)
Lattice spacing in x,y,z = 4.05 4.05 4.05
Created orthogonal box = (0 0 0) to (121.5 16.2 121.5)
  4 by 1 by 5 MPI processor grid
Lattice spacing in x,y,z = 4.05 4.05 4.05
Created 14884 atoms
Replicating atoms ...
  orthogonal box = (-0.01215 0 -0.01215) to (121.512 16.2 121.512)
  5 by 1 by 4 MPI processor grid
  14884 atoms
488 atoms in group topWall
488 atoms in group botWall
976 atoms in group boundary
13908 atoms in group mobile
Deleted 56 atoms, new total = 14828
Setting up run ...
Memory usage per processor = 3.92064 Mbytes
Step Time CPU CPULeft Lx Ly Lz Press PotEng Temp
       0            0            0            0     121.5243         16.2     121.5243    308.18371   -49323.396    280.25224
     500            1    4.8871951    141.72869     121.5243    16.206492     121.5243    193.47319   -49039.196    146.87594
    1000            2     7.711987    107.96784     121.5243    16.194636     121.5243   -390.07991   -49023.671    156.43672
    1500            3    10.555279    94.997528     121.5243    16.208261     121.5243     111.1091   -49007.852    166.23173
    2000            4    13.429525    87.291926     121.5243    16.220723     121.5243    694.09826   -48993.278    176.88401
```

If execution is very slow on a serial machine, reduce system size:
`region whole block 0 15 0 2 0 15`

Tutorial II: Al crack propagation

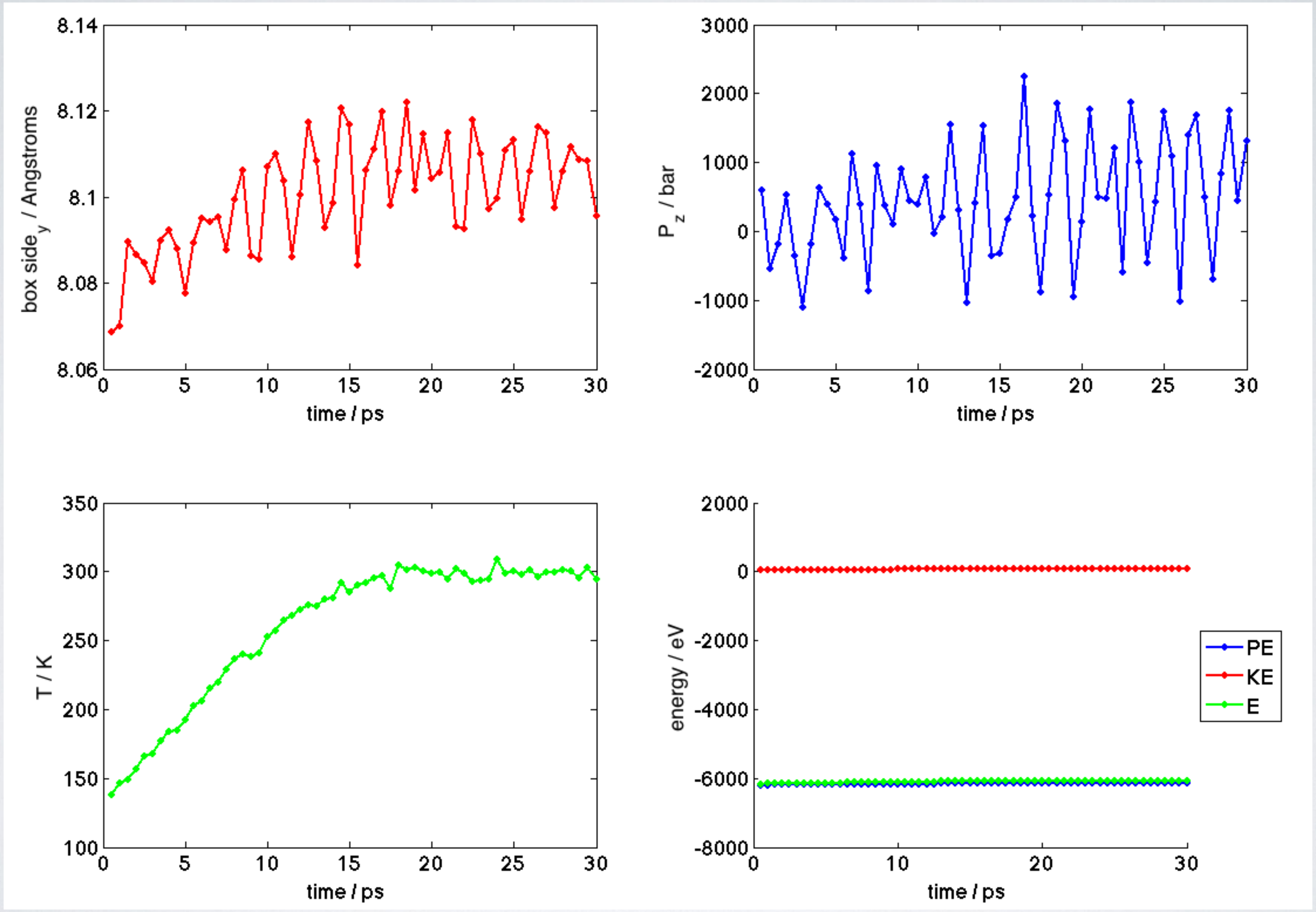
4. Analyze approach to equilibrium using **Al_eq.m**

```
>> Al_eq('Al_eq.txt')
```



```
>> Al_eq('Al_eq.txt')
>>
```

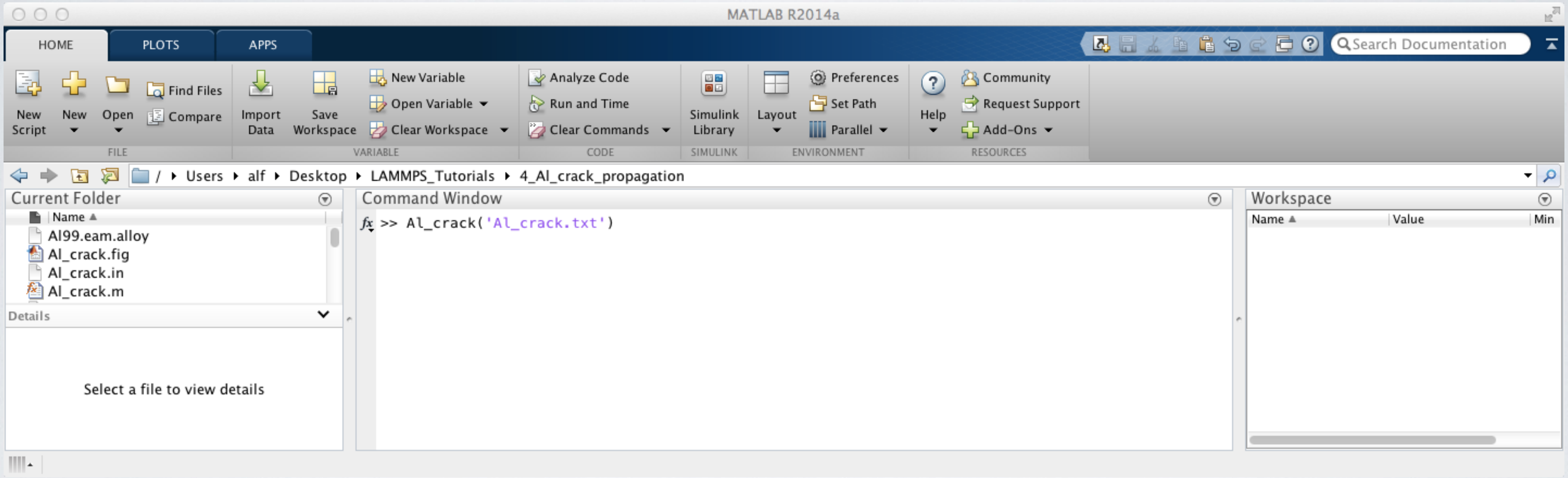
Tutorial II: Al crack propagation



Tutorial II: Al crack propagation

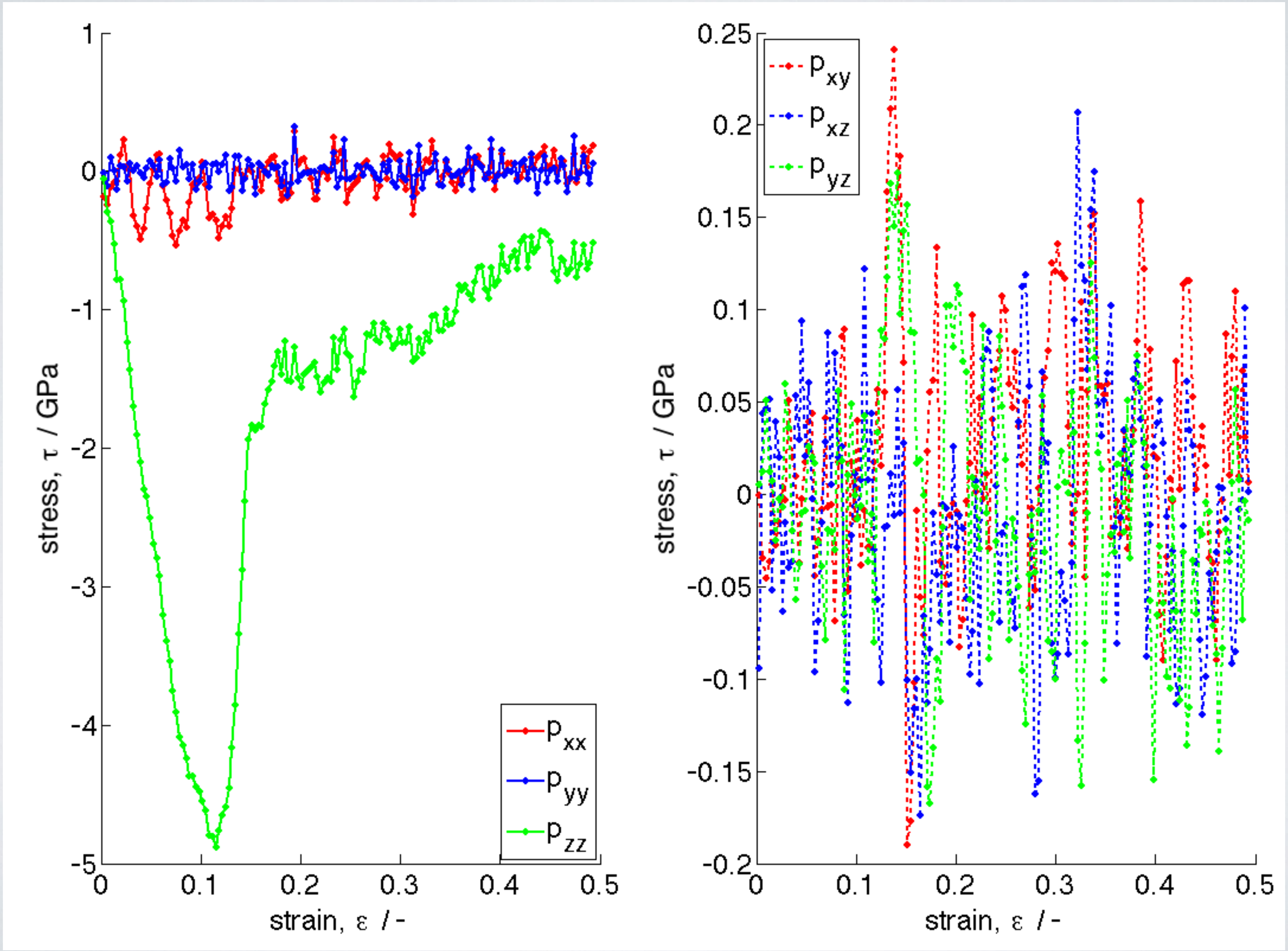
5. Analyze crack formation using **Al_crack.m**

```
>> Al_crack(‘Al_crack.txt’)
```



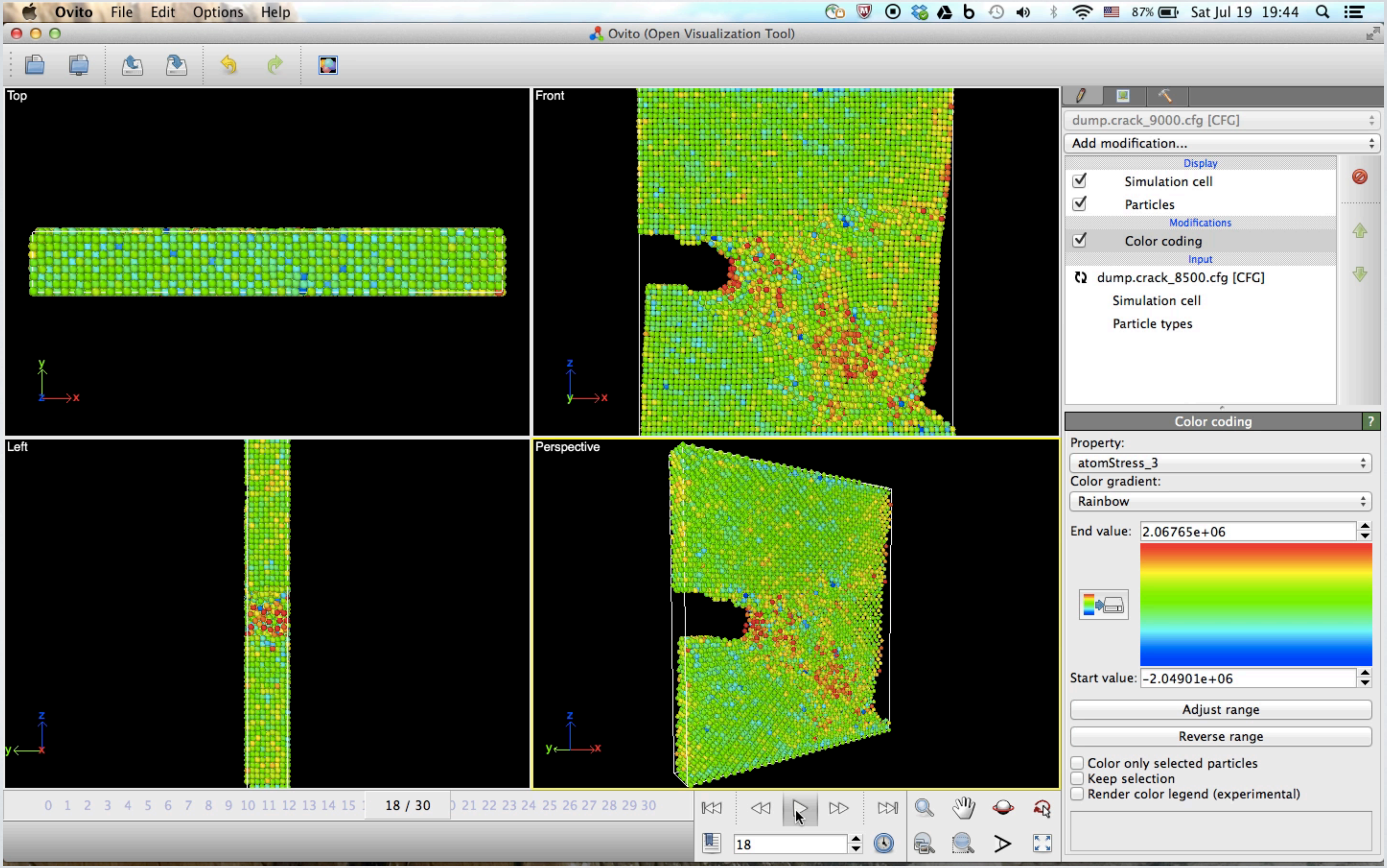
```
>> Al_crack('Al_crack.txt')
```

Tutorial II: Al crack propagation



Tutorial II: Al crack propagation

6. Visualization in OVITO



dump.crack_9000.cfg [CFG]

Add modification...

Display
- ✓ Simulation cell
- ✓ Particles

Modifications
- ✓ Color coding

Input
- dump.crack_8500.cfg [CFG]
  - Simulation cell
  - Particle types

Color coding

Property: atomStress_3

Color gradient: Rainbow

End value: 2.06765e+06

Start value: -2.04901e+06

Adjust range

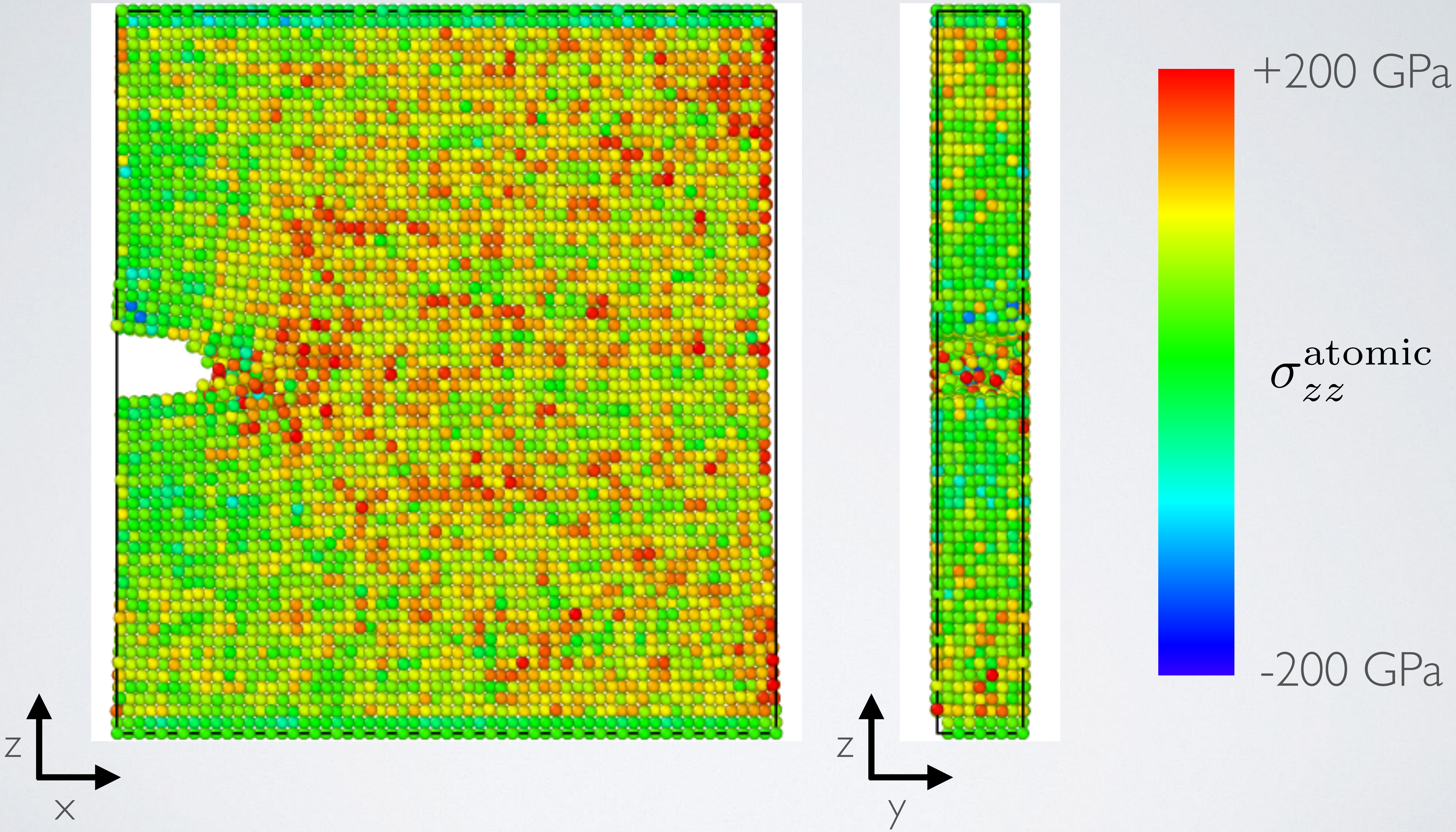
Reverse range

- ☐ Color only selected particles
- ☐ Keep selection
- ☐ Render color legend (experimental)

18 / 30

18

Tutorial II: Al crack propagation



Tutorial II: Al crack propagation

Extension I

- ✦ Change crystal orientation to explore the effect of cracks in difference crystallographic faces

Extension II

- ✦ {Reduce / remove / enlarge} initial crack to explore the impact of the initial imperfection size

Extension III

- ✦ Explore the effect of system size in x,y,z on atomic stresses