Final Project (part 1)

Got a Ni cif file from materials project. Went on Vesta set it to cubic and kept the lattice parameters the same at 100 010 001. I then exported a POSCAR file as one of my input files to run DFT. I then made the POTCAR from the given POT\_GGA\_PAW\_PBE\_54 from the shared rcc classroom for nickel then I copied the INCAR, KPOINTS ( set to auto) from project 2.

Ran the DFT and used the output files : CONTCAR and OSZICAR in my jupyter notebook to find the x1 and x2 as well and the coefficients A and B for the Lennard Jones potential curve that will be used later for the Tensile test.

Jupyter script and output values:

#Find the x1 and x2 values for the Lennard Jones linear equation : Ax1 + Bx2 from all 10 output files of contcar

from pymatgen.io.vasp.inputs import Poscar

for cont\_name in ['CONTCAR','CONTCAR\_3.49','CONTCAR\_3.48','CONTCAR\_3.47','CONTCAR\_3.46', 'CONTCAR\_3.45', 'CONTCAR\_3.44','CONTCAR\_3.43', 'CONTCAR\_3.42', 'CONTCAR\_3.41']:

# Note that you want to replace POSCAR to be the full name of your CONTCAR loop

str0 = Poscar.from\_file(cont\_name).structure

# This is the cutoff of your atomic distance, we assume atom pair that has distance beyond this value

# will not contribute to the total energy, here I use 5.0, you can change this to whatever you want

rcut = 20

n\_atom = len(str0)

x1, x2 = 0.0, 0.0

for i in range(n\_atom):

for j in range(n\_atom):

if i == j:

continue

r\_ij = str0.distance\_matrix[i,j]

# If distance is beyond cutoff distance, it does not count

if r\_ij > rcut:

continue

x1 += 1/r\_ij\*\*12

x2 += 1/r\_ij\*\*6

print(f'x1 = {x1}, x2 = {x2}')

OUTPUT:

x1 = 0.0003106766919017552, x2 = 0.06105833524442885

x1 = 0.00023521017038911774, x2 = 0.05312741330677989

x1 = 0.00024345028801287514, x2 = 0.0540500088450918

x1 = 0.00025200405120858155, x2 = 0.05499135036078838

x1 = 0.00026088435433101344, x2 = 0.055951874427691534

x1 = 0.0002701046613704239, x2 = 0.056932029091233745

x1 = 0.0002796790328741314, x2 = 0.057932274204363636

x1 = 0.0002896221542239614, x2 = 0.058953081774302

x1 = 0.00029994936534206946, x2 = 0.05999493632053318

x1 = 0.0003106766919017552, x2 = 0.06105833524442885

#Get the total energies from the OSZICAR output files from DFT calculation

import numpy as np

from sklearn.linear\_model import LinearRegression

E\_total = [-21.618996, -21.618996, -21.666048, -21.707778,-21.744356,-21.775927,-21.802639,-21.824717,-21.842270,-21.855472]

x1 = [0.0003106766919017552,0.00023521017038911774,0.00024345028801287514, 0.00025200405120858155, 0.00026088435433101344, 0.0002701046613704239, 0.0002796790328741314, 0.0002896221542239614, 0.00029994936534206946, 0.0003106766919017552]

x2 = [0.06105833524442885, 0.05312741330677989, 0.0540500088450918, 0.05499135036078838, 0.055951874427691534, 0.056932029091233745, 0.057932274204363636, 0.058953081774302, 0.05999493632053318, 0.06105833524442885]

X = np.array([[item1, item2] for item1, item2 in zip(x1,x2)])

model = LinearRegression(fit\_intercept=False)

model.fit(X,E\_total)

# The two parameters below are the two LJ potential you want to use for lammps calculation ( A and B coefficients for the LJ equation)

print(model.coef\_)

OUTPUT:

[76151.19557037 -743.91647072]

I then went to my lammpos input files NPT and NVT and edited the original pair\_style and pair\_coeff to :

mass 1 1.0

pair\_style lj/cut 2.5 ## set interatomic potential style to be EAM

pair\_coeff 1 1 a1 a2 2.5

(a1 = first coefficient found in jupyter script and a2 = second coefficient from jupyter notebook)

Then I changed the cna / atom and timestamp value to be smaller to have the NPT and NVT file run to completion without any errors. It took 2 seconds to run completely for both NPT and NVT(deform).

To graph the convergence from the Lennard Jones potential of Nickel I used the plot\_convergence.py file from the last lab using the data from the output file log.lammps from the NVT calculations and got back this figure.

