Exercise 7 – Computational Materials Engineering

Prof. M. Moseler – WS 2020/2021

Goals:

- Learn how to perform a molecular dynamics simulation at constant temperature
- Learn how to perform a local structural optimization
- Understand what finding a local minimum in the potential energy surface means
- Understand that there are many different local minima in the potential energy surface
- Understand that energy is not conserved in an NVT simulation

In this exercise sheet we will melt and quench a copper cluster using a Langevin thermostat and the FIRE algorithm discussed in the lecture. The interatomic interactions are modeled using the Morse potential with the parameters for copper from Phys. Rev. 114, 687, 1959.

It is recommended to create a new directory for this exercise. Make sure to copy *MorseCalculator.py* to this directory.

Tasks

Task 1: We start with a small finite fcc copper crystal. Download the script *A_initial_md.py* from ILIAS, run it and try to understand what it does. Note that in contrast to the previous exercises including copper, we do not impose periodic boundary conditions. Why is the system exploding when running the simulation? Check the temperature in the md.log file.

Task 2: Create a new script called *B_relax_system.py* based on *A_initial_md.py* that relaxes the structure into its closest minimum using the FIRE algorithm prior to running the Verlocity-Verlet propagation.

You can import the FIRE algorithm from ASE by adding *from ase.optimize import FIRE* to your script. Creating the FIRE object works in an analogous way as creating the VelocityVerlet object, namely *dyn = FIRE(atoms, trajectory='relax.traj')*. Use the *run* method of the FIRE object to optimize your atomic configuration into the closest minimum in the potential energy landscape. When calling *run*, specify the maximal force threshold *fmax=0.01* rather than the number of steps. After the relaxations the magnitude of all forces on all individual atoms should be less than 0.01 eV/ Å. Check that this is true by printing the forces after the relaxation. Investigate the trajectory of the relaxation with OVITO.

Check the trajectory of the subsequent Velocity-Verlet propagation. Why are the changes in the system much smaller than in Task 1?

Task 3: Create a new script called C_*heat_system.py* based on *B_relax_system.py*. In this script we want to apply a Langevin thermostat to the atoms and perform an NVT simulation. In order to do so, replace the VelocityVerlet object by a Langevin object

*dyn = Langevin(atoms, timestep=5*units.fs, temperature=2500*units.kB, friction=0.005, trajectory='langevin.traj', logfile='langevin.log')*

The timestep is increased to 5 fs and the target temperature is 2500 K. Run the simulation for 4000 steps. Do not forget to add *from ase.md.langevin import Langevin*. Note that the simulations might take several minutes.

Inspect the trajectory using OVITO. Plot the total energy as a function of time. Why is it not conserved? What does the convergence mean? Plot the potential energy and the temperature as a function of time. What is the state of matter of the system?

Task 4: Redo task 3, but reduce the friction from 0.005 to 0.002. What changes?

Task 5: Create a new script called *D_relax_system.py* to relax the final configuration of the trajectory obtained in Task 3. You can read in the final configuration of a trajectory with the command *atoms = read('langevin.xyz', index=-1)*. Inspect the trajectory of the relaxation using OVITO. Compare the energy of the relaxed structure with the energy of the relaxed system of task 2. Why is the energy different?