One million atoms walk into 1 Bar at 300 K or: Things I learned about LAMMPS during my PhD

Riley Vickers

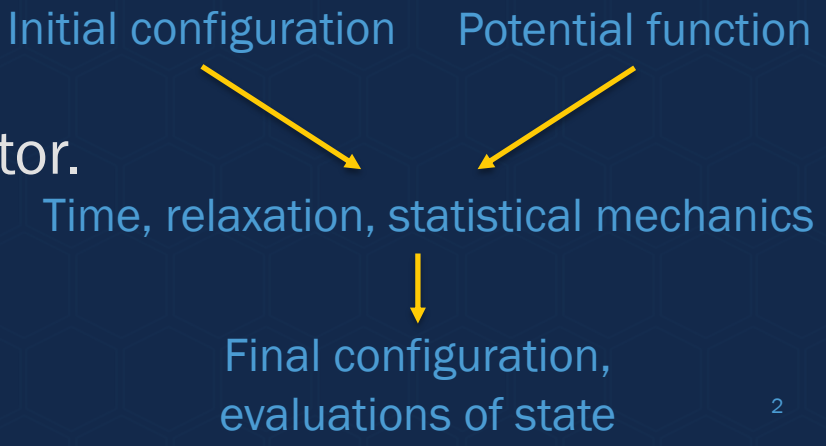
UMich ChE Simulation Center Workshop Series
October 18, 2023

What is LAMMPS?

Large-scale atomic/molecular massively parallel simulator

Not molecular dynamics alone, can be used for Monte Carlo, dissipative particle dynamics, Lattice Boltzmann Method, multiscale computations, etc.

Generally, a particle dynamics simulator.



Why use LAMMPS?

- Open-source and actively developed by individuals from a large variety of backgrounds and for a large suite of use cases.
- Out-of-the-box parallelized, with many options for optimization in both CPU alone and CPU/GPU environments.
- In-line tools for performing specific simulation steps/data analyses.
- Fine grain control of thermodynamical constraints/compensations during molecular dynamics.
- Well maintained/updated documentation.

General advice: Search any documentation for keywords related to analyses you want perform

My use cases for LAMMPS

Analyzing mass transport through and molecular-scale morphological characterization of polymeric materials using molecular dynamics.

In my PhD: crosslinked aromatic polyamide used for reverse osmosis membranes.

At UMich: ion exchange membranes used for water treatment, energy generation, and energy storage.

Molecular dynamics fundamentals

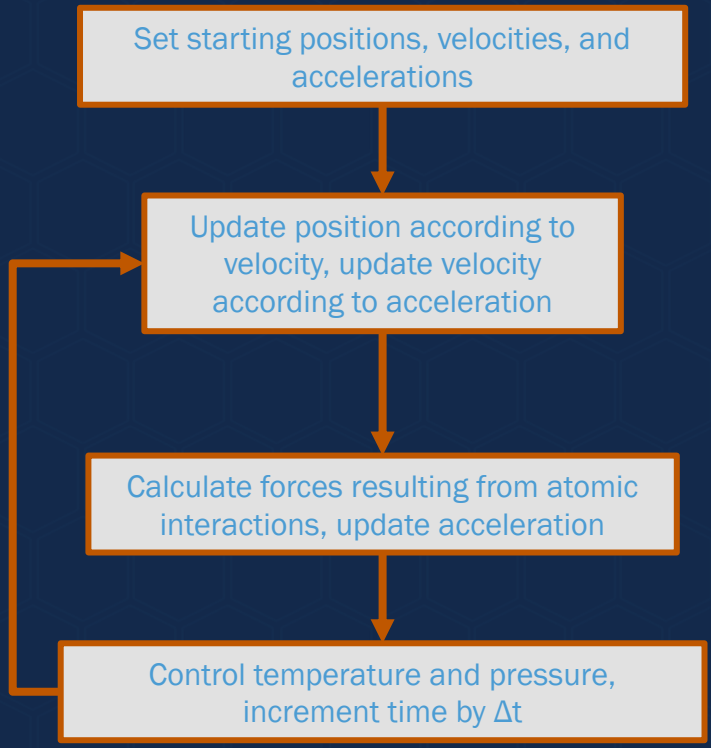
MD approximates energetic interactions between atoms, including the effect of non-bonded and bonded potentials.

Empirical interaction parameters are determined according to experimental characterization and/or ab initio calculation from quantum mechanics (i.e., DFT).

Interactions and the parameters that describe them make up the potential function, also referred to as the force field.

How does one choose which force field to use?

Prior knowledge of the system and relevant scales of interest are necessary.



Considerations of scale for molecular dynamics

Which interactions dominate your system of interest?

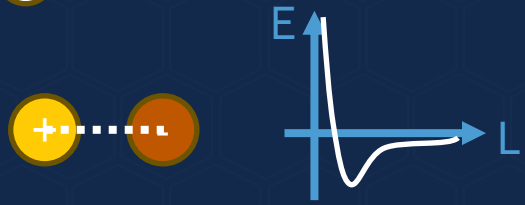
In aqueous systems, resolving the atomistic structure of water is often necessary.

Non-bonded interactions for atomistic molecular dynamics include Coulombics (electrostatics) and Lennard-Jones.

Bonded interactions for molecules include bonds, angles, dihedrals, and impropers. These control the conformational energy of the molecules.

For polymer systems, sometimes the “bead” scale, or coarse-grained scale is sufficient (think radius of gyration of a polymer chain).

Structure of even the smallest molecules in my systems are known to impact observed behavior



Molecular morphology omitted in favor of parameterization of bead energies

Parameterization of interactions of interest

Even once the structure of the potential function is decided, a specific force field must be chosen.

Force fields are often parameterized only for use in specific contexts, and care must therefore be taken when choosing force fields (e.g., what phase is the material).

This goes for combining force fields used to define for example a solvated molecule and the solvent.

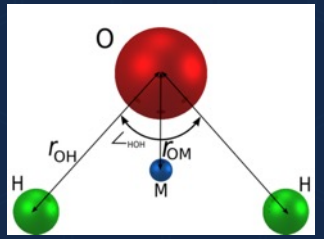
It is important to validate that the force field you choose accurately represents the system via comparison to some in-lab experimental results.

- none - turn off pairwise interactions
- hybrid - multiple styles of pairwise interactions
- hybrid/overlay - multiple styles of superposed pairwise interactions
- hybrid/scaled - multiple styles of scaled superposed pairwise interactions
- zero - neighbor list but no interactions
- adp - angular dependent potential (ADP) of Mishin
- agni - AGNI machine-learning potential
- aip/water/2dm - anisotropic interfacial potential for water in 2d geometries
- airebo - AIREBO potential of Stuart
- airebo/morse - AIREBO with Morse instead of LJ
- amoeba -
- atm - Axilrod-Teller-Muto potential
- awpmd/cut - Antisymmetrized Wave Packet MD potential for atoms and electrons
- beck - Beck potential
- body/nparticle - interactions between body particles
- body/rounded/polygon - granular-style 2d polygon potential
- body/rounded/polyhedron - granular-style 3d polyhedron potential
- bop - BOP potential of Pettifor
- born - Born-Mayer-Huggins potential
- born/coul/dsf - Born with damped-shifted-force model
- born/coul/dsf/cs - Born with damped-shifted-force and core/shell model
- born/coul/long - Born with long-range Coulomb
- born/coul/long/cs - Born with long-range Coulomb and core/shell
- born/coul/msm - Born with long-range MSM Coulomb
- born/coul/wolf - Born with Wolf potential for Coulomb
- born/coul/wolf/cs - Born with Wolf potential for Coulomb and core/shell model
- born/gauss - Born-Mayer / Gaussian potential
- bpm/spring - repulsive harmonic force with damping
- brownian - Brownian potential for Fast Lubrication Dynamics
- brownian/poly - Brownian potential for Fast Lubrication Dynamics with polydispersity

General advice: Search your literature to find which parameterizations are most utilized. Are there limitations in the force fields? Can you improve the fidelity of your results by comparing different force fields?

My current work as an example: parameterization of the system

The transferable intermolecular potential with 4 points (TIP4P) is used to parameterize the water molecules.



Optimized potentials for liquid simulations (OPLS) force field determines the non-bonded and bonded interaction parameters for the polymer molecules.

Alkali and halide ions LJ parameters are determined to accurately replicate solution properties at higher concentrations.

$$E(\phi) = E_{\text{bond}}(\phi) + E_{\text{angle}}(\phi) + E_{\text{n.b.}}(\phi) + E_{\text{torsion}}(\phi)$$

$$E_{\text{ab}} = \sum_{i}^{\text{on a}} \sum_{j}^{\text{on b}} \left[q_i q_j e^2 / r_{ij} + 4\epsilon_{ij} \left(\sigma_{ij}^{12} / r_{ij}^{12} - \sigma_{ij}^{6} / r_{ij}^{6} \right) \right] f_{ij}$$

$$E_{\text{bond}} = \sum_{\text{bonds}} K_r (r - r_{\text{eq}})^2 \qquad E_{\text{angle}} = \sum_{\text{angles}} K_\theta (\theta - \theta_{\text{eq}})^2$$

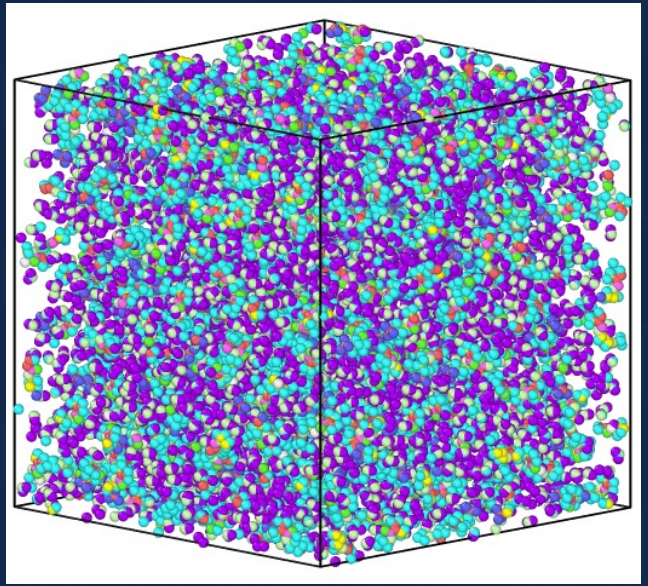
$$E_{\text{torsion}} = \sum_{i} \frac{V_1{}^i}{2} [1 + \cos(\phi_i + f_{i1})] + \frac{V_2{}^i}{2} [1 - \cos(2\phi_i + f_{i2})] + \frac{V_3{}^i}{2} [1 + \cos(3\phi_i + f_{i3})]$$

My current work as an example: validation of the force field

Pre-polymer solution components are randomly distributed through an over-sized domain.

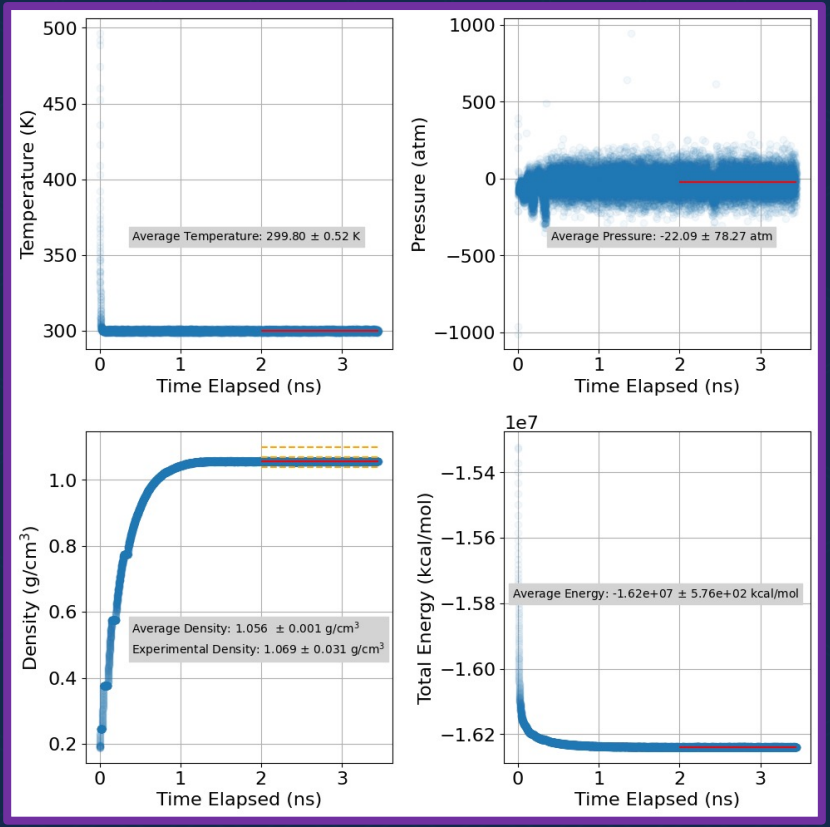
Alternating NVT (constant volume) and NPT (constant pressure) simulations are performed until temperature is stable and experimental pre-polymer solution density is matched.

Density matching makes up one validation point to assert that the selection of force field accurately represents the system.

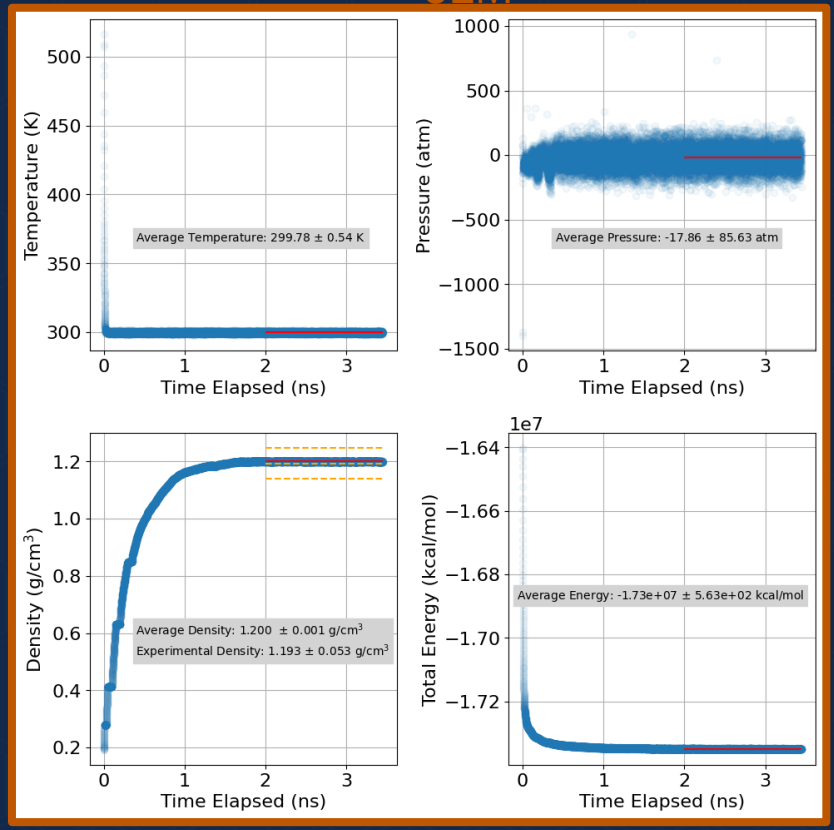


My current work as an example: validation of the force field

AEM

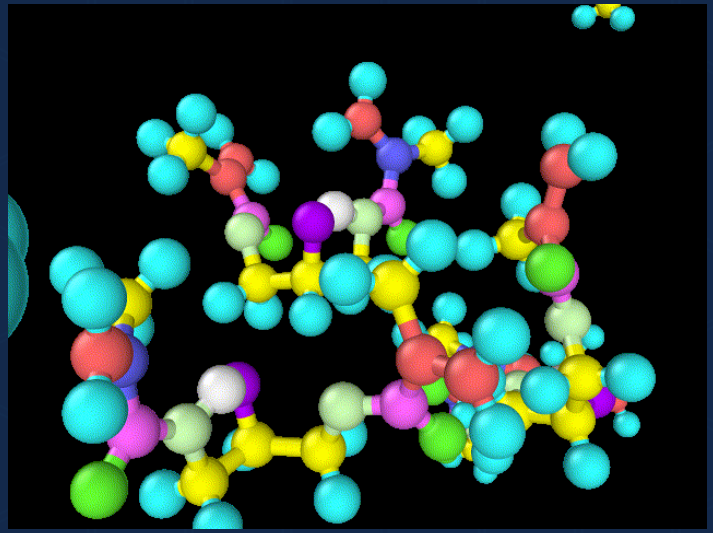


CEM



My current work as an example: useful tools available in LAMMPS

REACTER is utilized to perform the act of searching the simulation domain for candidate reaction sites, and reassigning the force field parameters once a reaction proceeds.



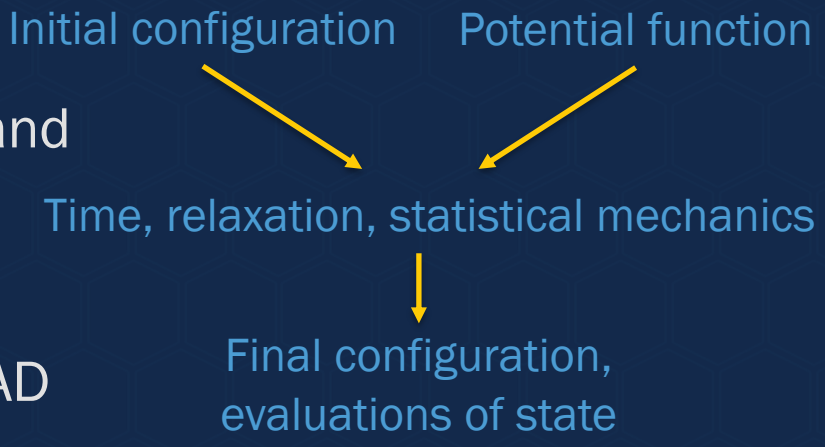
I found this tool, developed by another LAMMPS user, by searching my literature/the LAMMPS manual.

Summary so far

LAMMPS is a particle simulator that has many tools available for a variety of simulation schema.

Whatever you choose to do, see what tools/methods have been used before, and validate these methods for your system.

Your most powerful tool is your ability to search for the information you need. READ THE MANUAL! ASK QUESTIONS! ☺



General MD workflow

1. Force field selection- read the literature
2. Parameterization of particles/molecules- literature/software
3. Initial configuration of system- LAMMPS/software
4. Simulation instructions- LAMMPS
5. Data collection instructions- LAMMPS
6. Data analysis- LAMMPS/software
7. System visualization- software

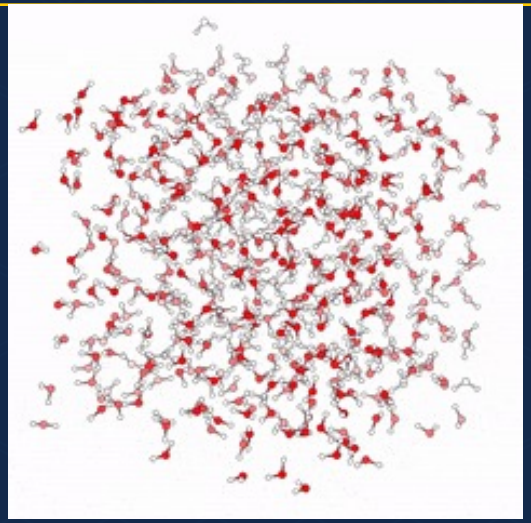
General advice: Where LAMMPS occurs, could replace with your simulator, GROMACS, HOOMD-blue, etc. Some of these simulators may have more of these steps included in their workflows, e.g., 2. and 7.

Real example

Keep it simple, smarty!

Start small and uncomplicated.

Validate at every step.



Working with something aqueous?

Let’s do a simulation of pure water!

Walkthrough of the files found on this git.

Forcefield selection- read the literature

Excellent recent review of water models, link in git readme.

For simplicity/simulation efficiency we will use the TIP3P-FB model.

Does a fairly good job at replicating properties of interest for aqueous simulation, but no water model is perfect.

Identify candidate force fields for your own work by reading the literature.

Parameterization of particles/molecules- literature/software

In this case, our system is composed of so few atom/molecule types that parameterization is as simple as reading a table and transcribing the parameters.

In other cases, you will need to feed in molecular topology to specific software (NOT LAMMPS/other MD simulators) to assign the correct interaction parameters.

Examples:

LigParGen is available as a webtool that takes a variety of molecular morphology formats and assigns OPLS parameters

amber/antechamber/am1bcc for GAFF

CHARMM-GUI for CHARMM

Parameterization of particles/molecules- literature/software

tip3p.mol

```
# Water molecule. TIP3P geometry

3 atoms
2 bonds
1 angles

Coords

1    0.00000  -0.06556   0.00000
2    0.75695   0.52032   0.00000
3   -0.75695   0.52032   0.00000

Types

1        1   # O
2        2   # H
3        2   # H

Charges

1       -0.84844
2        0.42422
3        0.42422

Bonds

1   1      1      2
2   1      1      3

Angles

1   1      2      1      3
```

simulation.in

```
log out/simulation.log
units real
atom_style full
region box block -20 20 -20 20 -20 20
create_box 2 box bond/types 1 angle/types 1 &
                extra/bond/per/atom 2 extra/angle/per/atom 1 extra/special/per/atom 2

mass 1 15.9994
mass 2 1.008

pair_style lj/long/coul/long cut long 10
pair_modify mix arithmetic
pair_coeff 1 1 0.155865 3.178
pair_coeff 2 2 0.0    1.0

bond_style zero
bond_coeff 1 1.0118

angle_style zero
angle_coeff 1 108.15

kspace_style pppm 1e-4

molecule water tip3p.mol  # this uses the TIP3P geometry
create_atoms 0 random 2133 34564 NULL mol water 25367 overlap 1.33
fix rigid all shake 0.001 10 0 b 1 a 1
minimize 1.0e-4 1.0e-6 1000 10000

write_data simulation.data
reset_timestep 0
timestep 1.0
velocity all create 300.0 5463576

fix integrate all npt temp 300 300 100.0 iso 1 1 1000.0

thermo 1000
thermo_style custom step temp press density etotal
#dump 1 all custom 100 out/simulation.lammpstrj id mol type x y z vx vy vz

run 100000
```

** This is NOT the only way that parameters can be defined for LAMMPS

Initial configuration of system- LAMMPS/software

If your initial configuration is relatively simple (i.e., well mixed/randomly or simply arranged), LAMMPS and other MD simulators often have significant tools available for creating random arrangements or simple geometries.

Check git for a link to LAMMPS page that contains a helpful list of molecular builder software to aid in the creation of more complicated initial configurations, useful beyond LAMMPS simulations.

Pure water is simple, pack a box on a lattice near the experimental density of water.

Initial configuration of system- LAMMPS/software

tip3p.mol

```
# Water molecule. TIP3P geometry

3 atoms
2 bonds
1 angles

Coords

1    0.00000  -0.06556   0.00000
2    0.75695   0.52032   0.00000
3   -0.75695   0.52032   0.00000

Types

1        1   # O
2        2   # H
3        2   # H

Charges

1       -0.84844
2        0.42422
3        0.42422

Bonds

1   1      1      2
2   1      1      3

Angles

1   1      2      1      3
```

simulation.in

```
log out/simulation.log
units real
atom_style full
region box block -20 20 -20 20 -20 20
create_box 2 box bond/types 1 angle/types 1 &
                extra/bond/per/atom 2 extra/angle/per/atom 1 extra/special/per/atom 2

mass 1 15.9994
mass 2 1.008

pair_style lj/long/coul/long cut long 10
pair_modify mix arithmetic
pair_coeff 1 1 0.155865 3.178
pair_coeff 2 2 0.0    1.0

bond_style zero
bond_coeff 1 1.0118

angle_style zero
angle_coeff 1 108.15

kspace_style pppm 1e-4

molecule water tip3p.mol  # this uses the TIP3P geometry
create_atoms 0 random 2133 34564 NULL mol water 25367 overlap 1.33
fix rigid all shake 0.001 10 0 b 1 a 1
minimize 1.0e-4 1.0e-6 1000 10000

write_data simulation.data
reset_timestep 0
timestep 1.0
velocity all create 300.0 5463576

fix integrate all npt temp 300 300 100.0 iso 1 1 1000.0

thermo 1000
thermo_style custom step temp press density etotal
#dump 1 all custom 100 out/simulation.lammpstrj id mol type x y z vx vy vz

run 100000
```

** This is NOT the only way that parameters can be defined for LAMMPS

Simulation instructions- LAMMPS

Information must be supplied that tells LAMMPS what to do with the parameters assigned to the atoms.

In LAMMPS, this information is provided with “styles” and simulation constraints with fix commands

Computational optimization, physical validity, and experimental conditions are controlled by how LAMMPS runs the simulation.

We want to equilibrate our simulation to its steady state density, then compare state variables to those of in-lab experiment.

Simulation instructions- LAMMPS

tip3p.mol

```
# Water molecule. TIP3P geometry

3 atoms
2 bonds
1 angles

Coords

1    0.00000  -0.06556   0.00000
2    0.75695   0.52032   0.00000
3   -0.75695   0.52032   0.00000

Types

1        1   # O
2        2   # H
3        2   # H

Charges

1       -0.84844
2        0.42422
3        0.42422

Bonds

1   1      1      2
2   1      1      3

Angles

1   1      2      1      3
```

simulation.in

```
units real
atom_style full
region box block -20 20 -20 20 -20 20
create_box 2 box bond/types 1 angle/types 1 &
                extra/bond/per/atom 2 extra/angle/per/atom 1 extra/special/per/atom 2

mass 1 15.9994
mass 2 1.008

pair_style lj/long/coul/long cut long 10
pair_modify mix arithmetic
pair_coeff 2 2 0.0    1.0

bond_style zero

angle_style zero

kspace_style pppm 1e-4

molecule water tip3p.mol  # this uses the TIP3P geometry
create_atoms 0 random 2133 34564 NULL mol water 25367 overlap 1.33
fix rigid all shake 0.001 10 0 b 1 a 1
minimize 1.0e-4 1.0e-6 1000 10000

write_data simulation.data
timestep 1.0
velocity all create 300.0 5463576

fix integrate all npt temp 300 300 100.0 iso 1 1 1000.0

thermo 1000
thermo_style custom step temp press density etotal
#dump 1 all custom 100 out/simulation.lammpstrj id mol type x y z vx vy vz

run 100000
```

** This is NOT the only way that parameters can be defined for LAMMPS

Data collection instructions- LAMMPS

By default LAMMPS does not collect any information for you.

Commands in the input script must be issued that save domain, per-atom, per-molecule, per-whatever information.

In LAMMPS, commands of interest for doing this include thermo, dump, compute, and fix, among others.

In this case, we want to monitor temperature, pressure, density, and total energy, to ensure we adequately match our thermodynamic conditions, while approximating the density of water and conserving energy.

Data collection instructions- LAMMPS

tip3p.mol

```
# Water molecule. TIP3P geometry

3 atoms
2 bonds
1 angles

Coords

1    0.00000  -0.06556   0.00000
2    0.75695   0.52032   0.00000
3   -0.75695   0.52032   0.00000

Types

1        1   # O
2        2   # H
3        2   # H

Charges

1       -0.84844
2        0.42422
3        0.42422

Bonds

1   1      1      2
2   1      1      3

Angles

1   1      2      1      3
```

simulation.in

```
log out/simulation.log

atom_style full
region box block -20 20 -20 20 -20 20
create_box 2 box bond/types 1 angle/types 1 &
                extra/bond/per/atom 2 extra/angle/per/atom 1 extra/special/per/atom 2

mass 1 15.9994
mass 2 1.008

pair_style lj/long/coul/long cut long 10
pair_modify mix arithmetic
pair_coeff 1 1 0.155865 3.178
pair_coeff 2 2 0.0    1.0

bond_style zero
bond_coeff 1 1.0118

angle_style zero
angle_coeff 1 108.15

kspace_style pppm 1e-4

molecule water tip3p.mol  # this uses the TIP3P geometry
create_atoms 0 random 2133 34564 NULL mol water 25367 overlap 1.33
fix rigid all shake 0.001 10 0 b 1 a 1
minimize 1.0e-4 1.0e-6 1000 10000

write_data simulation.data
reset_timestep 0
timestep 1.0
velocity all create 300.0 5463576

fix integrate all npt temp 300 300 100.0 iso 1 1 1000.0

thermo 1000
thermo_style custom step temp press density etotal
#dump 1 all custom 100 out/simulation.lammpstrj id mol type x y z vx vy vz

run 100000
```

** This is NOT the only way that parameters can be defined for LAMMPS

Data analysis- LAMMPS/software

In this case our analysis is as simple as calculating a time average.

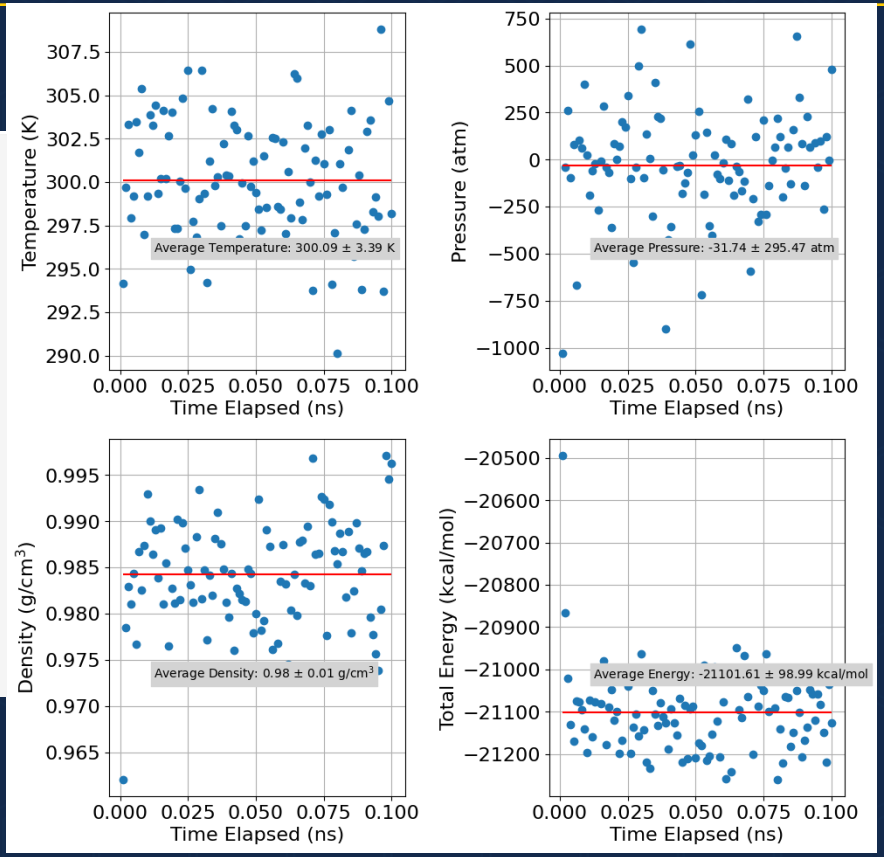
I advocate for writing your own analysis code (at least a simplified version so that you can understand what it is that more developed code is doing).

LAMMPS can perform data analysis in-line, in some cases using specific fix or compute calls.

For this step I have provided a simple python notebook to analyze the output generated by the LAMMPS simulation.

Data analysis- LAMMPS/software

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams.update({'font.size': 16})

file = './out/simulation.log'
data=[]
runFlag=False

fig,axes=plt.subplots(2,2,figsize=(10,10))
axes=axes.flatten()

with open(file) as f:
    while True:
        line = f.readline()
        if line.split(' ')[0] == 'run':
            runFlag=True
        if (line.split(' ')[0] == 'Per') and runFlag:
            header = f.readline().split()
            while True:
                line2 = f.readline()
                try:
                    float(line2.split()[0])
                except:
                    break
                else:
                    data.append([float(entry) for entry in line2.split()])

        if not line:
                break

df = pd.DataFrame(data, columns=header).iloc[1:]
df['Time'] = df['Step'] / 1e6
df.plot('Time','Temp',ax=axes[0],legend=False,ylabel='Temperature (K)', xlabel='Time Elapsed (ns)',ls='None',marker='o')
df.plot('Time','Press',ax=axes[1],legend=False,ylabel='Pressure (atm)', xlabel='Time Elapsed (ns)',ls='None',marker='o')
df.plot('Time','Density',ax=axes[2],legend=False,ylabel=r'Density (g/cm$^3$)', xlabel='Time Elapsed (ns)',ls='None',marker='o')
df.plot('Time','TotEng',ax=axes[3],legend=False,ylabel=r'Total Energy (kcal/mol)', xlabel='Time Elapsed (ns)',ls='None',marker='o')

fig.tight_layout()
mean_df = df.mean()
std_df = df.std()

for k,l,m,n in zip([0,1,2,3],['Temp','Press','Density','TotEng'],['K', 'atm', r'g/cm$^3$', 'kcal/mol'],['Temperature','Pressure', 'Density', 'Energy']):

    axes[k].annotate(r'Average {3}: {0:.2f} $\pm$ {1:.2f} {2}'.format(mean_df[l],std_df[l],m,n),xy=(0.15,0.33),xycoords='axes fraction',backgroundcolor="lightgray",fontsize=10)
    axes[k].hlines(y=mean_df[l],xmin=df['Time'].iloc[0],xmax=df['Time'].iloc[-1], color='r')
    axes[k].grid()
```

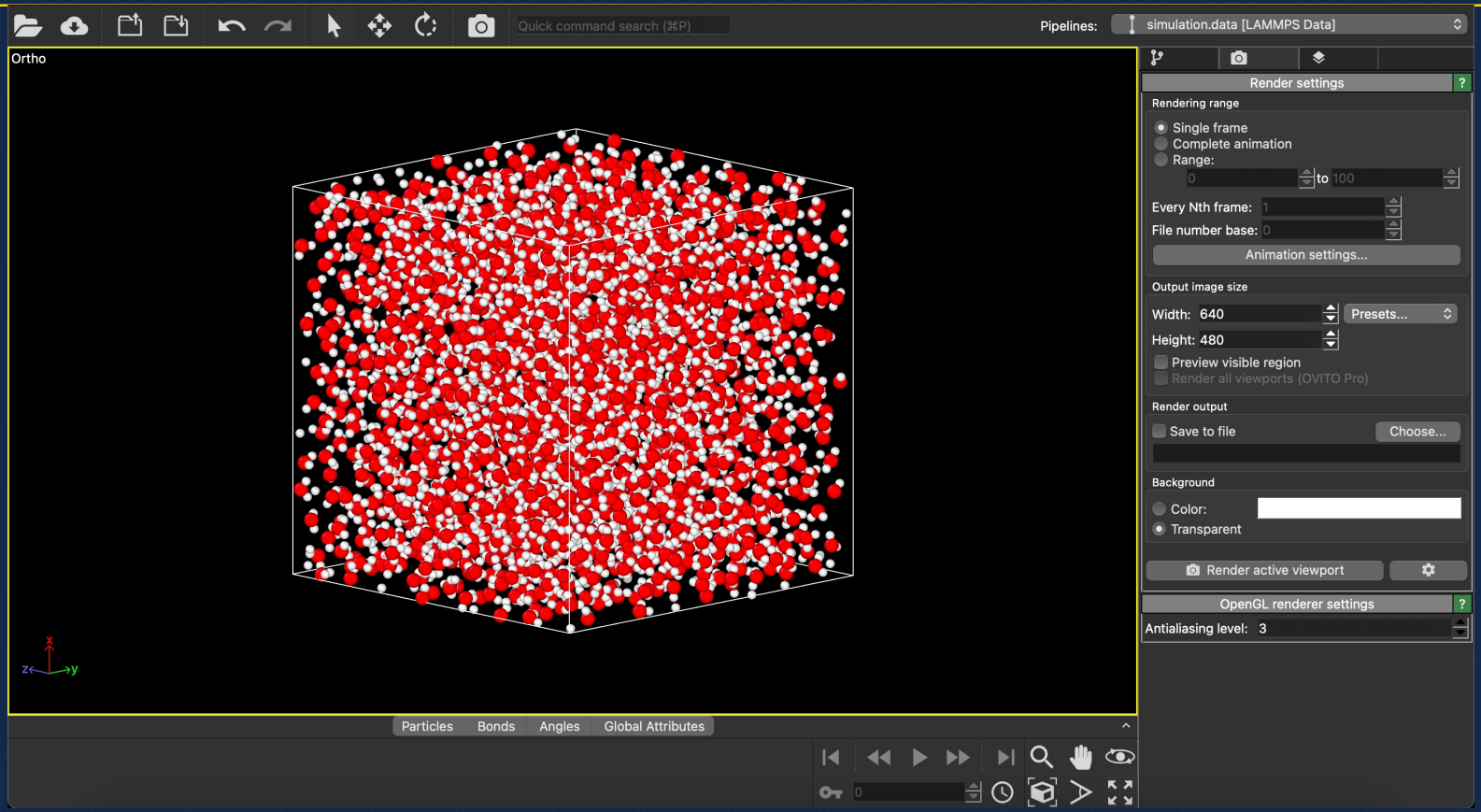


System visualization- software

While quantification can help in making sure simulations are behaving correctly, visualization is often necessary to diagnose problems and/or for publication of results.

Again, there is a helpful list of software available through a link in the git. I recommend OVITO for simple analyses as the workflow is relatively simple out-of-the-box.

System visualization- software



General tips and tricks

- Avoid paralysis by analysis. Think before you do, but don’t let thinking get in the way of acting.
- Visualize often. The storage overhead is usually worth it to have a visual of your system that you can check.
- Numpy, scipy, pandas, and matplotlib are your friends. These modules, if used correctly, can be sufficient in analyzing data for surprisingly large systems.
- Write your own code, but don’t feel like you must write your own toolkit. The importance is understanding, not software development.
- Don’t feel tied to a particle representation only, there are many voxel-based image processing tools that can aid in answering even molecular-scale questions.