Manual for running the BD Codes

January 18, 2026

Instructions for Running the Codes Stored in Folder: liquid_vapor_coexistence

In this section, you will calculate the coexistence density between the gas and liquid phases of a Lennard-Jones fluid. The aim is first to create a highly dense liquid phase without any bubbles or nuclei of the gas phase, then, merge the liquid phase with a gas phase. And finally, calculate the density profile of the combined system, after equillibrating the system.

To achieve this, you will use multiple LAMMPS input files, each configured as described below.

System configuration:

To make a high density liquid regime, you will use the file named `liquid_box_creator.in`. The description of a few key commands are given as below:

1. **Creating a large rectangular box:**

   ```
   region my_box block 0 15 0 15 0 90
   create_box 1 my_box
   ```

   All the details have already been explained.

2. **Applying NpT ensemble to compress the fluid and make it a liquid phase:**

   ```
   fix my_pressure all npt temp 0.4 0.4 1 z 10 10 1.0 couple none
   ```

   After generating the box throughout the whole box, you use a barostat along the z-axis with some pressure. We use a guessed pressure where the system can exist in the liquid phase and apply that pressure. For example, in the given script, you are applying the temperature `0.4 0.4`, and the pressure along the z-axis as `10 10`. Applying pressure along only the z-axis does not mean that in other directions the pressure will be different. It is just for the shape of the box, where the box will be compressed along the z-axis to bring the system to the specified pressure.

3. **Printing the last configuration after the equilibration:**

   ```
   write_data compressed_coordinate.data
   ```

We use the command to print the last configuration. The NpT thermostat and barostat have compressed the box along the z-axis such that the box will now be located in the middle of the initial box. For example, if the box initially extended between $0 - 90$, then after compression, it might be located between $38 - 52$.

After generating the liquid phase, the next step is to merge it with the gas phase. To achieve this, we take the previously generated and equilibrated box with the extension from 38 to 52 along the z-axis and place it into a larger box that spans from 20 to 70. The empty regions, from 20 to 38 and from 52 to 70, are then filled with a low-density gas phase. This is done by randomly placing an appropriate number of gas particles.

If the number of gas particles is too low, it will result in a lower chemical potential, causing some of the liquid-phase particles to evaporate and transition into the gas phase. Conversely, if there are too many gas particles, the chemical potential of the gas will be higher, prompting some of the gas particles to condense and deposit into the liquid phase. This process will continue until the vapor pressure and chemical potential reach their equilibrium values at the specified temperature.

To merge the two phases, we place the liquid phase into a larger box that extends further along the z-axis. The box is enlarged by editing the particle data, stored in the file named `compressed_coordinate.data`, which was output by the script `liquid_box_creator.in`. However, it is important to remove the periodic neighborhood along the z-axis, as the particles that were previously counted within the interaction range are no longer periodically connected. This is due to the wall shifting away from the particles at both ends (e.g., at lengths 38 and 52).

To remove the periodicity along the z-axis, we use a separate script called `intermediate.in`, which loads the particle's data and rewrites with the same file name, after running for 0 time steps. After removing periodicity, you can change the box size by altering the last generated particle data file `compressed_coordinate.data`.

1. **Removing periodicity along the z-axis:**

```
boundary p p s
```

   In this script, the periodic conditions are modified only for the z-axis. The `s` indicates that the particles are no longer connected through the periodic boundary in the nearest-neighbor list along the z-axis. This ensures that the wall applied to the system fully encloses all the particles along the z-axis, leaving no particles excluded.

   After removing the periodicity, the updated data is written to a new file, which is then loaded by the script named `coexistence_simulator.in`. In this script, the system data is loaded, and regions are defined on either side of the liquid phase, where particles will be added.

2. **Defining the regions and adding particles:**

```
region gas_region_1 block 0 15 0 15 20 35
region gas_region_2 block 0 15 0 15 55 70
create_atoms 1 random 100 87910 gas_region_1 overlap 1. maxtry 50
create_atoms 1 random 100 87910 gas_region_2 overlap 1. maxtry 50
```

   In this step, two regions are defined, but both are within the overall simulation box. These regions are then filled with gas particles.

3. **Binning the number of particles along the z-axis:**

   ```
   compute chunk_1 all chunk/atom bin/1d z lower 0.02 units reduced
   compute myChunk1 all property/chunk chunk_1 count
   fix 1 all ave/time 100 1 100 c_myChunk1 ...
   ...file bin_particles.lammpstrj mode vector
   ```

   This command is used to print the number of particles in each spatial segment along the z-axis. The z-axis is divided into bins of size 0.02 (in reduced units), and the number of particles in each bin is counted. For further details, consult the LAMMPS documentation or feel free to ask for additional clarification.

   Note: The rest of the description of the code remains unchanged.

Steps to follow:

To solve the relevant question, students are advised to follow these steps:

1. Run the LAMMPS script `liquid_box_creator.in` after setting the desired value of pressure, which ensures the high-density liquid phase. If you do not create a high-density liquid phase, during the simulation with the gas phase, the gas bubbles (which act as nuclei of the gas phase) will grow, leading to a fragmented region of liquid and gas.

2. Run the script `intermediate.in`, which loads the file `compressed_coordinate.data`.

3. Now you have the file `compressed_coordinate.data`, with periodic boundary conditions switched off along the z-axis. Open the data file using a text editor and change the box length, extending it symmetrically on both sides of the z-axis. For example, if the box length along the z-axis is `38 52 zlo zhi`, extend it to `20 70 zlo zhi`.

4. Save and close the file `compressed_coordinate.data`.

5. Run the script `coexistence_simulator.in`, which will equilibrate the system and generate the number of particles in each bin along the z-axis at different time steps. The data will be stored in the file named `bin_particles.lammpstrj`. This data will be used later to calculate the density distribution.

6. Run the Python script named `data_rectifier.py`, which will load the binned data for the number of particles and average it over time. The script will then print the density distribution ρ versus z in a text file named `averaged_density_profile.txt`.

7. Open the DESMOS online graph plotter in your browser and type the function $tanh(x)$. Observe the shape of the curve. Near the gas-liquid interface, you will notice a similar density distribution.

8. Run the Python script `density_profile_fitter.py`, which will fit the ρ versus z data to a *tanh* function. The terminal will display the gas and liquid coexistence densities. A plot of the simulated data and the curve fit will also be exported as a PNG file named `density_profile_fit.png`.

9. Now, write down your coexistence densities in front of the temperature in the first column. The second column must represent the vapor density and the third the liquid density. No need to say that the liquid density is always higher than its corresponding vapor density. Save the file named `liquid_vapor_densities.txt`

10. Now, run the plotter python file named `plotter.py`, which will export the Binodal plot, showing the coexistence curve along with the critical point.