Lab session: Molecular Dynamics and Glasses D. Coslovich & W. Kob

DPG School on Computational Physics of Glassy Systems 22/09/2015

The goal of this lab session is to get insight into the microscopic nature of liquids, glasses and crystals by means of visualization and post-processing tools. In the first part, you will perform molecular dynamics simulations of a pure Lennard-Jones system at different (but unknown!) state points. By inspecting the particles' trajectories and by evaluating static and dynamic correlation functions, you will be able to characterize these configurations as liquid, glassy or crystalline. Finally, you will analyze the dynamics of the Kob-Andersen binary Lennard-Jones mixture [Phys. Rev. E **51**, 4626 (1995)], possibly the most popular model for simulations of glass-forming liquids.

The programs and files for this lab can be downloaded from the school website

https://www.uni-oldenburg.de/index.php?id=43594

Download and uncompress the package. In the folder you will find the following files:

- `md.f90` : simple molecular dynamics code for Lennard-Jones particles
- `params.txt` : parameters file for the MD code
- `input_*.xyz` : starting configurations for the MD code
- `vis.py` : visualization script in Python
- `pp.*` : templates for a post-processing code in Fortran, C or Python

(A) Molecular dynamics simulations

The code `md.f90` performs a molecular dynamics simulation for particles interacting via the Lennard-Jones potential

$$u(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right]$$

Throughout the code, energy and distances are measured in units of ϵ and σ, respectively. The time unit is given by $\sqrt{m\sigma^2/\epsilon}$, where m is the mass of a particle, taken here as unity. The equations of motion are integrated using the velocity-Verlet algorithm and therefore the system samples the microcanonical ensemble.

You are provided with three configurations in xyz format that can be used to as initial configuration for molecular dynamics code. A configuration in xyz format is a block of $N + 2$ lines, where N is the number of particles. The first line contains the number of particles itself. The second line is a comment, which we use to store the lengths of simulation cell. The remaining lines contain the chemical species, the positions and the velocities of the particles, one particle per line.
We'll start off by making some short simulations for each of these configurations and then analyze their structure and dynamics. Your task is to determine which initial configuration corresponds to a liquid, which to a glass and which to a crystal.

1. Compile and execute the program by typing the following commands from a terminal

```
gfortran -O3 md.f90
./a.out params.txt
```

where `params.txt` is an "input file", which contains the simulation parameters in the form

`<parameter_name>  <value>`

By default, the code produces two output files:

- `output.log` contains some thermodynamic properties evaluated during the simulation
- `output.xyz` contains the particles' coordinates stored at regular time intervals in xyz format

The names of these two files, as well as of other simulation parameters, can be changed by modifying the input file. In particular, remember to change the name of the starting configuration file (parameter `file_input`) and to increase the number of steps (parameter `nsteps`) to about 10000.

2. The last column of `output.log` contains the total energy of the system. Inspect the degree of energy conservation to make sure your simulation is running correctly. Experiment with different values of the time step Δt while keeping the total simulation time $t_{sim} = \Delta t \times n$ constant, where n is the number of steps. Which is the largest value of Δt that allows to maintain a "reasonable" degree of energy conservation?

(B) Visualization and post-processing

3. The python script `vis.py` reads a trajectory file in xyz format and then displays a movie of the particles' motion using the Visual Python package (http://www.vpython.org). To visualize the trajectory generated during your simulations, type the following command

   `./vis.py <file>`

   where `<file>` is the trajectory file. What qualitative differences do you observe between the three trajectories?

4. Using one of the template codes, `pp.f90`, `pp.c` or `pp.py`, develop your own post-processing tools to calculate (a) the radial distribution function $g(r)$ and (b) the mean squared displacement $\delta r^2(t)$ Note that the particles' coordinates stored in the trajectory files are not folded back in the central simulation cell. Remember to take care of this when evaluating the distance between two particles!

5. Analyze the structure and the dynamics of the system at the three state points you simulated. Plot the radial distribution functions obtained from the three trajectories in a single plot: which differences do you observe? Produce a similar plot for the mean squared displacement. Use this information to identify liquid, glassy and crystalline samples. If needed, perform longer simulations to improve the statistics.

(C) Glassy dynamics

6. You can download the trajectory of a Kob-Andersen binary Lennard-Jones mixture close to the mode-coupling critical temperature from this link. Configurations are stored using the same xyz format as for the trajectories generated by the MD code. Thus, you can use the post-processing tools you just developed to analyze the sample. Note, however, that the interval between configurations is 8096 steps and the timestep 0.004. Add the corresponding mean squared displacement to the plot generated in the previous exercise and compare. What are your conclusions?

7. Modify your post-processing tools to identify mobile and immobile particles using a threshold value of 0.3σ on the particles' displacement after a time t. Visualize some configurations using the `vis.py` script, coloring mobile and immobile particles differently. Is there any spatial heterogeneity in the particles' displacements? Explore the role of the time lag t in the mobility calculation and compare qualitatively the results.