Fluid Dynamics Simulations of Liquid Argon (“FluidDyn”)

Joheen Chakraborty, Elyas Obbad
{jc5110, eo2446}@columbia.edu

November 22, 2021

1 Molecular dynamics simulation

We will parallelize a classic problem in computational physics: simulating the molecular dynamics of liquid argon in a closed container. Given a box of side length L containing N liquid argon particles, the system can be well-approximated to behave as a classical fluid interacting under the Lennard-Jones potential. The de-dimensionalized Lennard-Jones potential between two point-mass particles is given by:

$$U(r) = 4\left[\left(\frac{1}{r}\right)^{12} - \frac{1}{2}\left(\frac{1}{r}\right)^{6}\right]$$

Our simulation will be composed of discrete time steps of duration τ. At each time step, we will have 3-dimensional vectors representing the positions $(\vec{r}_1, \vec{r}_2, \ldots, \vec{r}_N)$, velocities $(\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_N)$, and net forces $(\vec{F}_1, \vec{F}_2, \ldots, \vec{F}_N)$ of each particle. The net force on each particle can be calculated from the potential as:

$$\vec{F}_i = -\sum_{j \neq i}^{N} (\vec{r_j} - \vec{r_i}) \frac{dU}{dr_{ij}} = \sum_{j \neq i}^{N} (\vec{r_i} - \vec{r_j}) \left[\left(\frac{1}{r_{ij}}\right)^{14} - \frac{1}{2}\left(\frac{1}{r_{ij}}\right)^{8}\right]$$

Then, we can approximate the updated positions and velocities of each particle in the simulation via the following recurrence relations:

$$\begin{aligned}
\vec{r}_{n+1} &= \vec{r}_n + \vec{v}_n \tau + \left(\tau^2/2\right)\vec{F}_n \\
\vec{v}_{n+1/2} &= \vec{v}_n + \left(\tau/2\right)\vec{F}_n \\
\vec{v}_{n+1} &= \vec{v}_{n+1/2} + \left(\tau/2\right)\vec{F}_{n+1}
\end{aligned}$$

where n indexes the time steps. Any starting configuration can work, but a reasonable baseline is to begin the simulation with equally spaced, static particles. *Note*: This simulation can also be implemented using 2-dimensional vectors for positions, velocities, and net forces.

There are two convenient checks we can make to ensure physical self-consistency of the simulation. In particular, energy should be conserved at all time steps because the container is a closed system. We can check this by summing the total Lennard-Jones potential energies and the total kinetic energy (from the v_i) over all particles; the resulting sum should not vary through the entire simulation. Additionally, after many time steps the system should reach thermal equilibrium regardless of its starting configuration; this can be checked by plotting the distribution of particle velocities, which in thermal equilibrium follows a Maxwell-Boltzmann distribution.

2 Parallelization

Because the system state depends on recurrence relations for the 3-dimensional vectors $\vec{r}$ and $\vec{v}$, we will need to store two $3 \times N$ lists to keep track of this information at successive time steps. We can speed up the computation of total forces on each particle, and the resulting position and velocity updates, by dividing along the N axis and delegating the computations through time on batches of particles to separate threads. For example, for a system with $N = 1000$ particles we could use a parallel 4-thread code in which each thread performs the force, position, and velocity update computations for a batch of 250 particles and moves them through time. This is a good example of data parallelism; because the system will contain a large number of particles, splitting the computations particle-wise should provide significant speedups.

Visualizing the algorithm at work may be possible if we can find an easy-to-use 2D or 3D graphics library. In addition, we'll check the two conditions above (conservation of energy and thermal equilibrium) to verify validity of our solution.