Simulating the motion of Argon atoms based off their initial velocity, and their inter-atomic forces

---

Project Outline

Description

- Project aimed at calculating energy between argon atoms , E_{PE}, using a Lennard-Jones pairwise potential. Inter-atomic force, F, is then calculated using the derivative of the Lennard-Jones potential with respect to distance ($F = -\frac{dE_{pe}}{dr}$).
- Using the force calculated, a new position and velocity of each atom is calculated across a period of time. This cycle is then repeated, resulting in atomic motion.

Applications of Molecular Dynamics Simulations (MDS)

- In the real world, MDSs are used throughout science. In Biochemistry, they are used to simulate protein folding. In Physics, they are used to approximate solutions to n-body problems, such as the *Millenium Simulation*. They are used in material sciences to test how stress affects systems, and to predict things like temperature and pressure in virtual systems.

Features

- This simulation works verifiably for two argon atoms in one dimension, logging metrics such as positions, distance, E_{PE}, F and collisions with walls of container.
- This simulation also can approximate the interatomic energies and forces of many Ar atoms. Many-body behaviour for this simulation is unverifiable (see below), and in all likelihood not accurate enough for commerical use. It should, however, be accurate enough to still be interesting.

- Graphing of results is included in this project. Each simulation results in a file of changes in interatomic distance, energy, and force against time. This is then plotted using `matplotlib`

Non-Features

- This project, like the professional LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator), does not include visualisation. While a basic Python wrapper can be put round the library, the focus of this project was about the data rather than the graphical output.
- Temperature is constant throughout, and, as configured, only Ar atoms can be used. While this is theoretically easily changeable, it was added complexity that, for me, was not necessary in understanding and simulating interatomic energies, and the motion that arises from interatomic forces.

Models

LJ Potential

- The Lennard Jones Potential is used to calculate interatomic potential energy, and can be described as follows, where K represents Kelvin

$$E_{PE} = 4\epsilon\left[(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^{6}\right] \ K \qquad (1)$$

$$E_{PE} = \frac{A}{r^{12}} - \frac{B}{r^{6}} \ K \qquad (2)$$

- In order to calculate E_{PE} and F, I elected to implement the Lennard-Jones Potential (*LJP*). The potential was introduced in 1924 by Sir John Edward Lennard-Jones to describe the interaction of liquid argon.
- The potential is comprised of the sum of 2 distinct interactions - attrative interactions, and repulsive interactions.
    - 3 possible sources for attractive interaction: permanent dipole-dipole, permanent dipole-induced dipole, induced dipole-dipole interactions.

$$\therefore E_{pe(total\ attractive)} = -\frac{1}{(4\pi\epsilon)^2} \cdot (\frac{0.66\mu^4}{k_B T} + \mu^2\alpha + 0.75\alpha^2 I) \cdot (\frac{1}{r^6}) \ K$$

      where ϵ is permittivity of medium, μ is a dipole moment, α is polarisibility of electron cloud, I is first ionisation energy, and r is interatomic distance.

    - At constant temperatures for one species, this simplifies to $-\frac{A}{r^6}\ K$
    - Repulsive interactions (at constant temperature) given by $\frac{B}{r^{12}}\ K$
    - $\implies E_{PE(total)} = \frac{B}{r^{12}} - \frac{A}{r^6} J$

- Since the *LJP* was introduced for liquid Argon, I decided Argon would be the most sensible element to start off with. Gaseous Argon also adheres incredibly well to the LJP, and can be more easily simulated, due to the increased number of simplifications and assumptions that can be made - namely, that gas has no fixed volume.

- I also chose the *LJP* as it is a common choice for commerical MDSs. It is featured in LAMMPS - the *Large-scale Atomic/Molecular Massively Parallel Simulator*, from Sandia National Laboratories.

- LAMMPS is a standard open source library for proper molecular dynamics simulation. However, as an article in the Royal Society of Chemistry pointed out, the *LJP* may not be the best choice for this simulation, as when simulating many atoms, the simulation reduces range instead of considering every possible atomic potential. As below, I did this in a less naive way than it is normally done to attempt to counteract this problem.

Two Body

- LJ Pairwise potential yields most accurate results with two Ar Atoms
- Two atoms are initialised with a position and velocity, and a potential is initialised between them. The potential uses the interatomic distance to yield an energy and a force. The force is applied to each atom, and the atoms are then moved across a time period.
- In testing, the Two Body Simulation yielded accuracy of +0.75% accuracy when a frame was computed at an interval of 1×10^{-7} seconds.
- TODO: Benchmarking of time vs res

Multi-body

- Multibody Simulations generally conducted using the **Embedded Atom Model**(EAM), or more commonly, the **Modified Embedded Atom Model**(MEAM)

- EAM determines E_i, by considering the pairwise potential of atom i to all other atoms, as well as energy needed to embed i in the electron cloud of every other atom present

$$E_i = F_\alpha \left(\sum_{j \neq i} \rho_\beta(r_{ij}) \right) + \frac{1}{2} \sum_{j \neq i} \phi_{\alpha\beta}(r_{ij})$$

- EAM typical use case: inter-ionic energies in metal alloy lattices. EAM does not work where bonds are directional, hence use in metals. Works poorly for covalent substances. For Ar-Ar sim, no directional bonding so still seemingly somewhat appropriate despite unconventional use case.
- EAM vs MEAM: MEAM takes shape of molecule into account when calculating charge density, increasing accuracy for real-world applications. Where molecules are symmetric spheres, $\rho_{EAM} = \rho_{MEAM}$
- This is very much the case with Ar molecules, hence, an EAM approach is equivalent to an MEAM approach.
- EAMs are typically used in metals due to their dependance on no directional bonds for accuracy, making them inappropriate for substances with significant covalent character. Again, Argon being monatomic means that there are no directional bonds present. This, I believe, qualifies the use of the EAM in this simulation despite it being unconventional.

Physical Assumptions

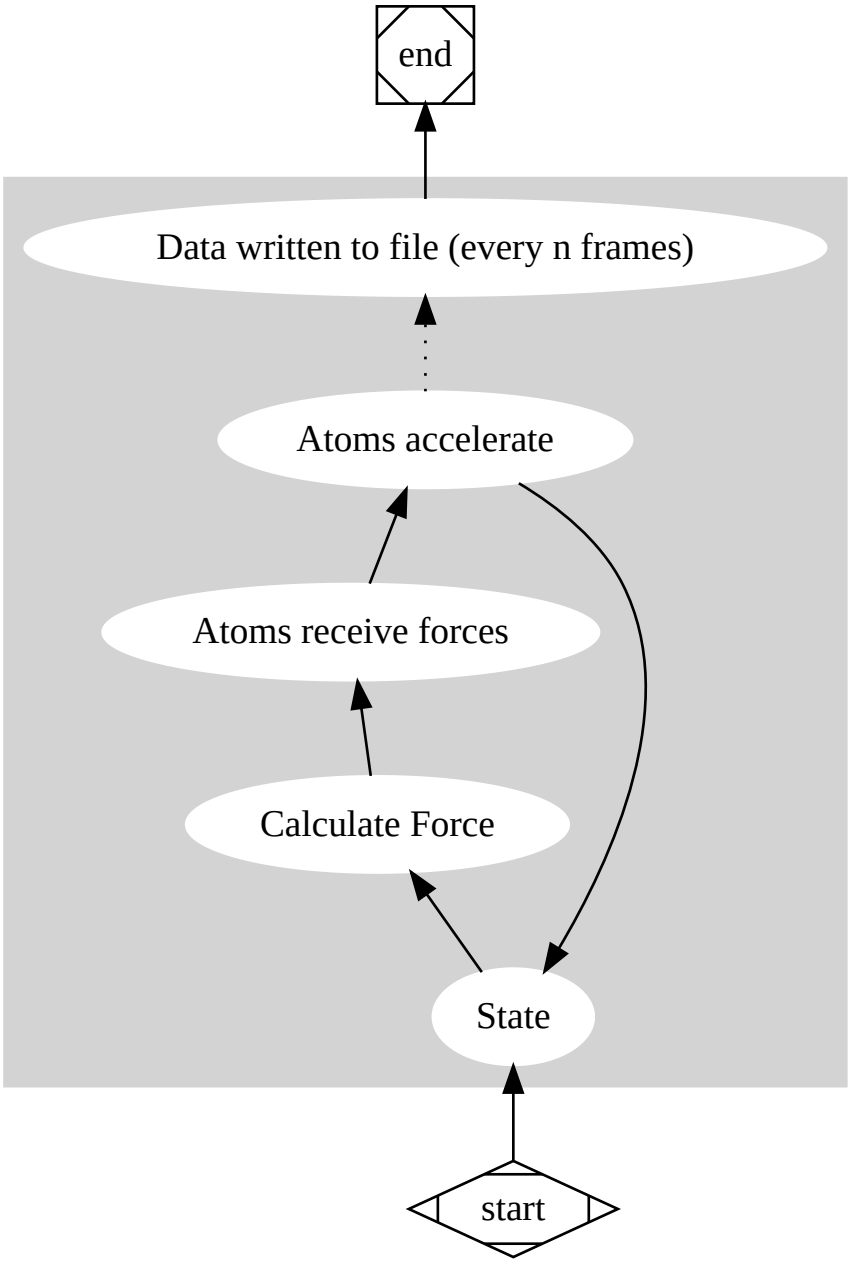
- Collisions with container walls are **inelastic**. This is a modelling assumption that is incorrect, and had to be implemented due to the fact that with elastic collisions, $\sum E$ of the system increased.
- What I did to try to mitigate this problem is calculate a coefficient of restitution, e, using my simulation. Here, e refers to collisions between the walls of the container and the Ar atoms.
- I was able to simulate a value which resulted in a reasonable error margin. This was done by assuming perfectly elastic collisions to begin with, and looking at resultant error. Error was, and is, calculated by looking at average interatomic distance, and seeing how this varies from predicted equilibrium bond distance (see **Verification**)
- This method, however, was scrapped
    - **Too specific**. Calculated values for e required recalibrating when various factors were changed, such as size of container, initial energy of atoms, and initial displacement of atoms.

    - **Circular verification**. Calibrated using deviation in interatomic distance. Tested by making sure that deviation in interatomic distance was small. Arguably therefore not a proof of validity

- In the end, particles stay oscillating together when walls a far enough away from area of action.
- Analytically solving for a general expression for a coefficient of restitution is beyond my ability, as this simulation does not happen under ideal gas laws by definition, hence $pV = nRT$ cannot be used to find e

---

Implementation



- Based on a cycle through states. Each state is calculated solely from the previous state.
- States hold information such as
    - Atoms: Position, velocity, mass
    - *LJP* between atoms: pair of atoms, E_{PE}, F, interatomic distance, $\mathrm{Ar\text{-}Ar}$ values for σ and ϵ
- Each state is seperated by a period of time. It was found that 1×10^{-7} seconds yielded an accuracy of $+0.75$
- Simulated in **one dimension**. Due to design of software, jump to three dimensions should not be too difficult and will likely be doen as I continue to work on this in the future. However here it was not done as it adds complexity, and having one dimension does not detract from atomic motion, when the metric used for verification is interatomic distance.
- Having mentioned ease of implementation, it is not as straight forward as introducting a vector dataclass with a distance method. Stearic factors of the Ar atoms would need to be considered, and this is currently beyond my ability.
- Also assume that all force calculated from potential contributes to the motion of particles (i.e. no loss of energy due to rotation of particles).
- Particles modelled as point masses.

Optimisation

Two Body

- Dynamic Resolution for very small distances relative to eqm distance
    - As force of repulsion increases with r^12, particles experience large forces when edging near each other
    - Hence, as particles get closer, sampling rate should increase to avoid a situation where particles can phase through each other
    - As mentioned, sampling at a frequency of $100\mathrm{ns}$ yields a deviation from equilibrium distance of 0.75% Hence, at equilibrium distance and above, resolution of $100\mathrm{ns}$ is used. At shorter distances, resolution decreases $\propto \sqrt{r}$. This was a fairly arbitrary decision. The square root function was

used due to its exponential decrease in the range $1 \leq x \leq 0$, and the fact that its range is defined as $\sqrt{r} \geq 0$.
- Time complexity of a frame is constant (**only in two body**)
- Time complexity of a full simulation is $\mathcal{O}(n)$, where n represents the number of cycles per frame, ignoring writing to file

Multibody Optimisations

- **Simplification of EAM approach**. EAM was picked over MEAM as Ar atoms are spherical and symmetrical, thus there would be no difference in approach. Approach taken was to ignore proportion of energy that comes from embedding atom i in the electron cloud of other atoms in the system. This was done for two reasons

    - Complexity. While this would not change the time complexity of the multibody systen, it adds a significant proportion more computation.
    - Accuracy. I believe that ommiting this term, whilst probably making the multibody sim too inaccurate for professional use, does not affect the general result too significantly. EAMs and MEAMs are typically used when calculating intermolecular energies in metal alloys. In metal alloys, molecules (in this case ions) are incredibly close, and the delocalised electrons in the metals make the placement of an ion in the electron cloud very energeticlly significant. However, Argon is monatomic, meaning that there is not much of a total electron cloud to account for - certainly much less significant than in metals. Hence, while there is definitely a significant discrepancy in results, results are theoretically close enough to still be interesting.

- **Range cutoff**. Standard cutoff $r_c = 2\sigma$ in lots of LJ sims *(RSC Misuse of LJ)*. Opted for different approach

    - Imagine case where $r_{ij} \geq 2\sigma$. In reality, these atoms experience attractive forces, but if standard r_c used, sim would report no motion.
    - Instead, use relative cut off based on the $\sum |F|$ that the atom experiences.
    - $E_{ref} := \phi_{ij}$, where i is a selected atom, and j is its closest neighbour. ϕ_{ij} is the potential energy as calculated by the Lennard-Jones potential between atoms i and j. $E_n := \phi_{in}$, where n is another atom in the sim. Potentials are only considered where $\frac{E_n}{E_{ref}} > 0.01$
    - Effect on time complexity. Worst case is still $O(n^2)$, as in rare edge cases it could still be the case that every atom pair needs to be

considered. In reality, this approach is similar to a cutoff, vastly limiting the number of pairs that need considering to only a few per atom.

- **Parallelisation** - Once all potentials have been assigned, the computation of forces on the atom can be parallelised, as well as the operations where atoms are moved. **This has not been done in this simulation**, but seemed like the obvious next step. This is included as it was something I had planned to do, but ended up being quite complex.

    - To do this, each atom i would be picked up by a thread I. The thread would then handle the computations for every potential between i and $k : k > i$.
    - The atom's force accumulator would have to be modified to become of the form of a queue, to ensure that different threads are not trying to write to the same resource.
    - Once all of the potential calculations, atom i will be moved by thread I.

N.B. Due to Amdahl's Law, parallelisation may not have been a particularly fruitful route to take. Amdahl's Law is something I learned about very recently, and is used to predict maximum theoretical performance gains when parallelising a process. It is calculated

$$\frac{1}{1-p}$$

where p is the proportion of total (single-threaded) running time of a program that **can be serialised**. The resulting number after the calculation is a theoretical max speed increase. For instance, if the formula yielded 2, then your program would have a maximum $2\times$ speed increase.

Deciding which atoms are interacting is an operation that cannot be parallelised. Neither is calculating the force on atoms once the potentials have been spawned (not trivially anyway), due to the fact that one attribute in an atom (force) will have to be updated by multiple threads.

Verification

Two Body

- Accuracy achieved by comparing average intermolecular distance to expected value

- Finding expected equilibrium distance
  + *LJP* constant ϵ is depth of potential well, in Kelvin
  + Thus, equilibrium distance is distance where energy is minimum

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

$$\implies \frac{dE_{PE}}{dr} = \frac{4\epsilon\sigma}{r^2}\left[6(\frac{\sigma}{r})^{5} - 12(\frac{\sigma}{r})^{11}\right] Knm^{-1}$$

  (where distances are in nm)

$$\text{SP where} \frac{dE_{PE}}{dr} = 0$$

$$\implies r = 2^{\frac{1}{6}} \cdot \sigma \text{ at equilibrium distance}$$

- For argon, this yields an equilibrium distance of 0.3755 nm
- Hence percentage error is calculated as

$$\%\text{Error} = 100 \times \frac{r_{avg} - 2^{\frac{1}{6}} \cdot \sigma}{2^{\frac{1}{6}} \cdot \sigma}$$

- With a constant sample of 100ns, a +0.75% error is calculated

Many Body

- Due to the simplifications made, I do not expect the many body simulation to be entirely accurate.
- I also run into a problem here, in that, while each of the parts of the many body simulation are themselves verifiable, I do not have a good way to verify that the system as a whole works.
- I had considered that it may be possible to compare the distribution of the E_k of the atoms in my simulation to a Maxwell-Boltzmann distribution, and use the average difference as a test for accuracy. However, I don't think that I would have a large enough sample size for the calculated difference to be meaningful enough. Moreover, Maxwell-Boltzmann is a result of kinetic theory of gases. Thus, it is derived using ideal gas assumptions - significantly, they assume no forces between gas particles. This assumption does not hold true in my simulation, and begins to break down in reality at 90K.