APC 523 Final Project Report: A MD Simulator

Bingjia Yang

Chemistry Department, Princeton University

I. INTRODUCTION

Molecular dynamics is a simulation method for understanding the physical movements and properties of atoms and molecules. It serves as a numerical “thought experiment” using a model that approximates a real physical or chemical system. To test the proposed theory or model, the simulation results may be then compared to the experimental results. We may also carry out simulations when it is difficult or even impossible to do the physical experiments.

There are three aspects needed for a molecular dynamics simulation and any one of them could largely influence the quality of the simulation. First, a model that describes the inter-particle interaction. Second, a calculation method that can be used to get the energies and forces from the model. There could be a trade-off between accuracy and efficiency. Last but not the least, we need an algorithm to integrate the equations of motion.

We want to build a code capable of carrying out molecular dynamics simulation for a given potential model and specified system.

II. MATHEMATICAL MODEL

Consider a classical system composed of N atoms and described by the Hamiltonian

$$H(\boldsymbol{p},\boldsymbol{q}) = \sum_{i=1}^{N} \frac{\boldsymbol{p}_i^2}{2m_i} + U(\boldsymbol{q}_1, ...\boldsymbol{q}_N), \tag{1}$$

where U is the potential energy of the system, and m_i the mass of the ith atom.

We will state the equations of motion and show how the equations of motion can be integrated.

A. Finite difference Method

The velocity Verlet algorithm explicitly uses following equations to evolve positions and velocities of particles simultaneously:

$$\boldsymbol{q}_i(t+\Delta t) = \boldsymbol{q}_i(t) + \frac{\Delta t}{m_i}\boldsymbol{p}_i(t) + \frac{\Delta t^2}{2m_i}\boldsymbol{F}_i(t), \tag{2}$$

$$\boldsymbol{p}_i(t+\Delta t) = \boldsymbol{p}_i(t) + \frac{\Delta t}{2}[\boldsymbol{F}_i(t) + \boldsymbol{F}_i(t+\Delta t)]. \tag{3}$$

It satisfies time-reversibility and symplectic property that are crucial for the long-time stability of numerical solver.

B. Time Evolution Operator and Numerical Integrators

The Liouville operator can be written as a sum of two contributions

$$\begin{aligned} iL = iL_1 + iL_2 &= \sum_{\alpha=1}^{3N} \frac{\partial H}{\partial p_\alpha}\frac{\partial}{\partial q_\alpha} - \sum_{\alpha=1}^{3N} \frac{\partial H}{\partial q_\alpha}\frac{\partial}{\partial p_\alpha} \\ &= \sum_{i=1}^{N} \frac{\boldsymbol{p}_i}{m_i}\frac{\partial}{\partial \boldsymbol{q}_i} + \sum_{i=1}^{N} \boldsymbol{F}_i \frac{\partial}{\partial \boldsymbol{p}_i}. \end{aligned} \tag{4}$$

Since iL_1 and iL_2 generally do not commute, by Trotter theorem, we have

$$e^{iLt} = e^{(iL_1+iL_2)t} = \lim_{P\to\infty,\Delta t\to 0} [e^{iL_2\Delta t/2} e^{iL_1\Delta t} e^{iL_2\Delta t/2}]^P, \tag{5}$$

where $\Delta t = t/P$. For a finite P, we obtain

$$e^{iL\Delta t} = e^{iL_2\Delta t/2} e^{iL_1\Delta t} e^{iL_2\Delta t/2} + O(\Delta t^3) \tag{6}$$

Starting from the initial condition $(\boldsymbol{q}(0), \boldsymbol{p}(0))$, the evolution of the system can be expressed as

$$\begin{pmatrix} \boldsymbol{q}(\Delta t) \\ \boldsymbol{p}(\Delta t) \end{pmatrix} = \exp\left(\frac{\Delta t}{2}\boldsymbol{F}(0)\frac{\partial}{\partial \boldsymbol{p}(0)}\right) \exp\left(\Delta t \frac{\boldsymbol{p}(0)}{m}\frac{\partial}{\partial \boldsymbol{q}(0)}\right) \exp\left(\frac{\Delta t}{2}\boldsymbol{F}(0)\frac{\partial}{\partial \boldsymbol{p}(0)}\right) \begin{pmatrix} \boldsymbol{q}(0) \\ \boldsymbol{p}(0) \end{pmatrix} \tag{7}$$

This yields the position and velocity update of the velocity Verlet algorithm we listed in Section II A. For an example of a single particle moving in one dimension, it can be expressed as a three-step procedure:

$$\begin{aligned} p(\Delta t/2) &= p(0) + \frac{\Delta t}{2}F(x(0)) \\ x(\Delta t) &= x(0) + \frac{\Delta t}{m}p(\Delta t/2) \\ p(\Delta t) &= p(\Delta t/2) + \frac{\Delta t}{2}F(x(\Delta t)). \end{aligned} \tag{8}$$

C. Force Field

In molecular dynamics, force field is used to calculate the forces between atoms. In this work, we take the Lennard-Jones potential as an example.

The Lennard-Jones potential considers both repulsive and attractive interactions, which are the most fundamental interactions in a molecular system. The repulsive term ($1/r^{12}$) describes the Pauli repulsion when r is small. The attractive term ($1/r^6$) describes the attraction and it vanishes when r is infinitely large. The expression is as follows:

$$V(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right], \tag{9}$$

where r is the distance between the two atoms. ϵ and σ are two adjustable coefficients allowing the description of the real systems with acceptable accuracy.

III. SOFTWARE ARCHITECTURE

The code is split into several modules: **atoms** (contains class **system**), **forcefields** (defines function **lj**), **integrator** (defines function **update**) and **simulation** (contains class **dynamics**).

A. Inputs

The user should prepare a json input file determining the initial configuration of the system as well as the simulation parameters. Currently, we support following key words:

- *mode*: defines the simulated ensemble, e.g. *nve*
- *init_temp*: initial temperature used to create random velocities for all atoms, unit: K
- *time_step*: the amount of time by which the integrator steps, unit: fs
- *total_steps*: total simulation time is just *total_steps* multiplied by *time_step*
- *init_config_path*: initial configurations (format: xyz) provided by the user
- *ff_style*: defines which force field is used, e.g. *lj*
- *ff_coeff*: a list of force fields coefficients, e.g. [ϵ, σ, cutoff] for *lj*
- *log_freq*: sets the frequency to print thermodynamic information
- *dump_freq*: sets the frequency to dump a snapshots of the system

The **system** class is extended from the Atoms class in ASE library. It is used to describe simulation box and the collection of atoms within. The arguments of **system** constructor are: atomic symbols, atomic positions, cell size and a boolean array determining where the boundaries are periodic or not. Besides, several getter functions are defined for calculating thermodynamic observables.

B. Simulation

At current stage, only micro-canonical ensemble (NVE) and Lennard-Jones potential are implemented. Due to the modular design of the code, the program is easily extensible to support other simulated ensembles and force fields. After parsing the json input file, the initial configuration and all simulation parameters are stored in an object of **dynamics**. The main driver code - the **run** function in **dynamics** class - handles outputs and calls stepping method. The velocity Verlet algorithm is the integration method we use in this work, which is defined in the **integrator** module. At each simulation step, the **update** function performs time integration to update both the positions and velocities for all atoms. For numerical stability, we wrap the atomic positions by minimal image convention and subtract the velocity of the mass center from the atomic velocities every a few steps.

C. Outputs

There are two kinds of outputs in this program:

- Log file: a list of thermodynamic observables printed every *log_freq* steps to a single text file. The default is time step, potential energy, kinetic energy, total energy and cell volume.
- Dump files: snapshots of atoms which are printed every *dump_freq* steps to separate xyz files. These files could be then easily imported to visualization softwares like OVITO for analysis.

IV. CASE STUDY

We provide a sample problem in the */example/argon_rdf* directory. The system contains 108 Ar atoms interacting with Lennard-Jones potential (coefficients: $\epsilon = 0.01$eV, $\sigma = 3.405\mathring{A}$ and cutoff=$8\mathring{A}$). We use our MD Simulator (MDS) to run the NVE simulation with an initial temperature $T = 94K$ and fixed cell size $[17.35, 17.35, 17.35]\mathring{A}$. The time step and total steps are set as 1fs and 100000, which leads to a trajectory of 100ps.

Radial distribution function (RDF) $g(r)$ is a measure of the local structure in fluids. In our case, it quantifies the probability of finding an Ar atom at a distance of r away from the reference

Ar atom. To obtain the RDF for our system, the distance between atoms in all dumped snapshots are calculated and then grouped into several bins (100 bins in our plot) for the histogram.

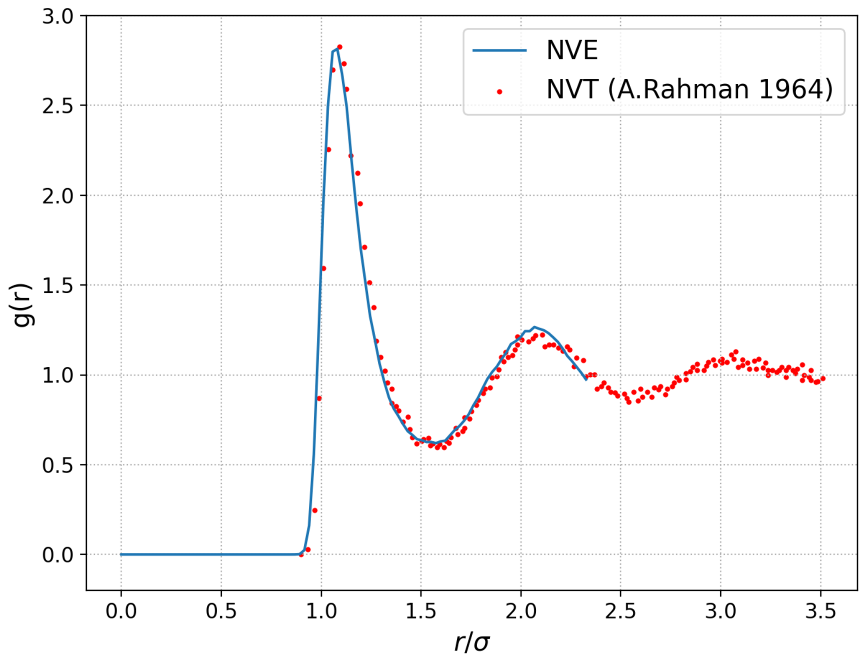

FIG. 1. The Radial distribution function computed by MDS and the reference [2].

We plotted in Fig.1 the resultant RDF and the canonical ensemble (NVT) simulation data by A.Rahman [2] ($T = 94.4$ K, $\rho = 1.374\text{g} \cdot \text{cm}^{-3}$). The computed RDF curve shows a good agreement with the reference. The slight discrepancy between them might be due to the different simulated ensembles and finite size effects.

V. BENCHMARK

We plotted in Fig.2 the wall time of simulating the sample problems defined in Sec.IV against the system size. These simulations used the same total steps (100 steps) and started from the same initial temperature and atomic density. We concluded that the MDS scales linearly in system size for this problem.

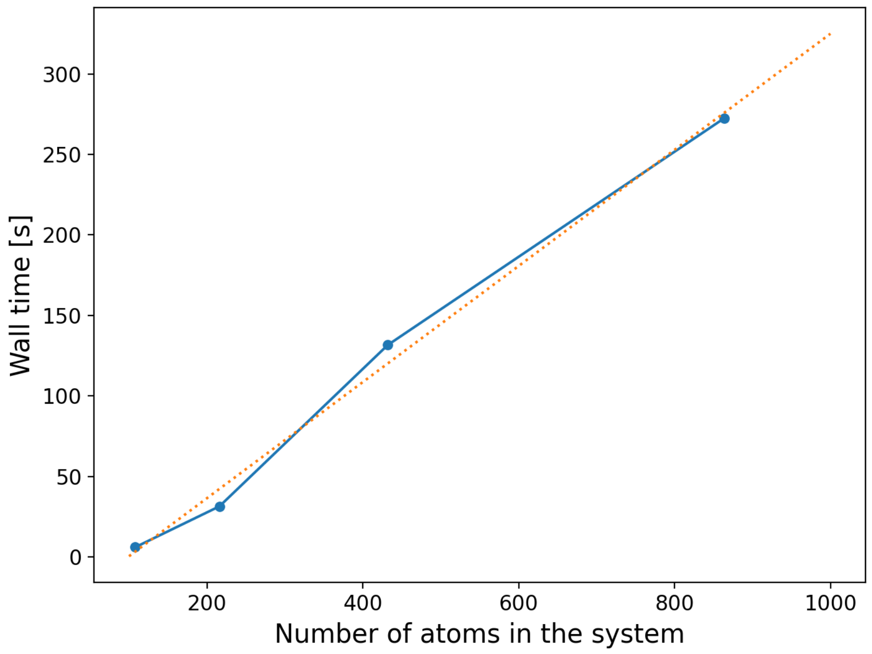

FIG. 2. The timings in seconds for MDS simulations on sample problems for different system size.

VI. SUMMARY AND OUTLOOK

MD Simulator is written in python and the main dependencies are python packages Numpy and ASE. The case study showed good agreement with the previous publication. The program is linear scaling on short-range Lennard-Jones potential. It is possible to parallelize the program in the future.

We implemented a program capable of integrating Newton's equations of motion for a collection of atoms with a given simulated ensemble. But currently the only implemented force field is Lennard-Jones potential. It is designed to be easy-to-extend with new features, such as other boundary conditions, new force fields or new simulated ensembles.

VII. REFERENCES

[1] Tuckerman, Mark., 2010. Statistical mechanics: theory and molecular simulation. Oxford university press.

[2] Rahman, A., 1964. Correlations in the motion of atoms in liquid argon. Physical review, 136(2A), p.A405.