Diffusion Coefficient of the Lennard-Jones Fluid

Molecular Statistic

2013

1 Introduction

In this project we elaborate on the Molecular Dynamics (MD) simulation we have been working on for the past weeks. The main assignment for this project is to calculate the diffusion coefficient D of the Lennard-Jones Fluid.

1.1 Diffusion

Diffusion is a transport property that refers to the phenomenon of a flow of material from one region to another, which arises from migration of matter down a concentration gradient. Since we are dealing with a homogeneous liquid, there is no meaning in defining a concentration gradient. What we are dealing with in this simulation is *self-diffusion*.

As described by Leach[1] if we can calculate the two properties mean-square displacement (MSD) and the velocity auto-correlation function (VACF) we can calculate the self-diffusion coefficient D.

Quick refresh; two equations relate the diffusion coefficient with microscopic properties. The relation between the MSD and VACF and D is as follows;

$$D = \lim_{t\to\infty} \frac{\mathrm{MSD}(t)}{6t} \tag{1.1}$$

$$D = \frac{1}{3}\int_0^\infty \mathrm{VACF}(t) \;\; dt \tag{1.2}$$

MSD is defined as;

$$\mathrm{MSD}(t) = \langle |\mathbf{r}(t_0+t) - \mathbf{r}(t_0)|^2 \rangle = \frac{1}{N}\sum_{i=1}^{N} |\mathbf{r}_i(t_0+t) - \mathbf{r}_i(t_0)|^2 \tag{1.3}$$

where $\langle x \rangle$ indicates the expected, or average, value N is the number of particles and $\mathbf{r}_i(t_0+t)$ is the position vector of particle i at time t away from a time origin.

VACF is defined as;

$$\mathrm{VACF}(t) = \langle \mathbf{V}(t_0+t) \cdot \mathbf{V}(t_0) \rangle = \frac{1}{N}\sum_{i=1}^{N} \mathbf{V}_i(t_0+t) \cdot \mathbf{V}_i(t_0) \tag{1.4}$$

where $\mathbf{V}_i(t_0+t)$ is the velocity vector for particle i at time t away from time origin and N is the number of particles.

[1] Leach section 7.6.3 p. 380

1.2 Usage of Correlation Function

From the book p. 381:

It is also good usual practice to average over time origins when possible.

A short introduction seems appropriate because this is confusing.

When you plot the MSD vs. time it does not look like a smooth linear function but has curves etc, and is therefor hard to do linear regression on. That is why we need some kind of averaging on the results.

When we run the actual simulation, we run it for at least n steps before we do any sampling, because the system needs to be equilibrated first. So when you do your first samples and want to calculate the MSD between step i and 0, this is only relative compared to the time origin, which is *after* equilibration. It is therefor common to use multiple time origins that are separated by several time steps, but in our case we calculate the MSD for each sample as a time origin.

We then calculate a new list `MSD_list` where the index represents how many time steps from a time origin the MSD is calculated. For example the list element

```
MSD_list[5]
```

contains the an average MSD calculated 5 samples away from each time origin. This means choosing t_{max} time steps and calculating `calculate_msd(R`$_5$`, R`$_0$`)` where `R`$_0$ is the current time and `R`$_5$ is the time 5 steps from `R`$_0$. The average is calculated from these MSD-values and `MSD_list[5]` is set to contain this average.

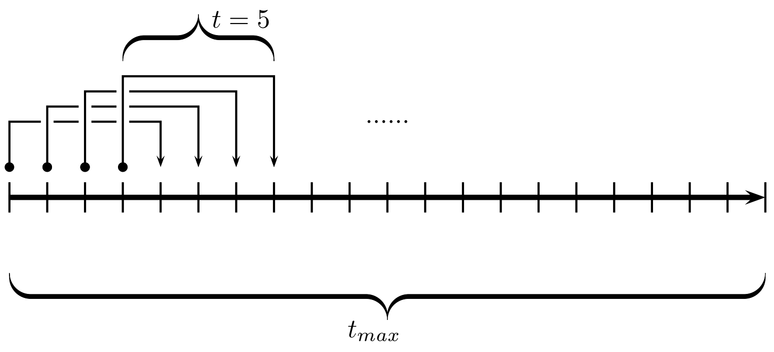


Figure 1.1: Illustrating calculation of `MSD_list[5]`, by calculating the MSD between each time origin (dots) and 5 steps ahead (arrow head).

To help you, we have written the Python code to do this:

```python
t_max = n_samples/10
msd_list = np.zeros(t_max)
vacf_list = np.zeros(t_max)

for t in range(t_max):
    for t0 in range(n_samples - t):
        msd_list[t] += calculate_msd(R_list[t0+t], R_list[t0])
        vacf_list[t] += calculate_vacf(V_list[t0+t], V_list[t0])

    msd_list[t] /= (n_samples - t)
    vacf_list[t] /= (n_samples -t)
```

Note: Remember to have a `time_list` with corresponding dimension as `vacf_list` and `msd_list` and time for each sample. Hint use `np.arange(t_max)`.

1.3 Solving for Diffusion Coeffecient

1.3.1 Limit of MSD

When we want to obtain the diffusion coefficient from MSD(t) we want to find the limit of;

$$\lim_{t \to \infty} \frac{MSD(t)}{t} \tag{1.5}$$

by linear regression. If you plot msd_list as a function of the time t, the slope of that line is $6D$. Luckily there is a Numpy/Scipy package for linear regression. So `import scipy` and use it in the following method, for two lists `X` and `Y`.

```python
import scipy
slope, intercept, r_value, p_value, std_err = scipy.stats.linregress(X, Y)
```

The returned value `slope` is then the limit.

1.3.2 Integrating VACF

When we want to find the diffusion coefficient from VACF we want to solve the integral;

$$\int_0^\infty \mathrm{VACF} \quad dt \tag{1.6}$$

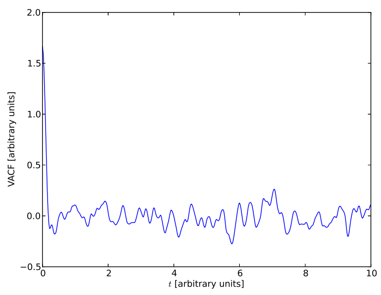
by numerical integration using the trapezoidal rule. We have all the VACF in a list for each time i, then the integral I_i for this time step is defined as;

$$I_i = \frac{1}{2}(\mathrm{VACF}_{i+1} + \mathrm{VACF}_i) \cdot (t_{i+1} - t_i) \tag{1.7}$$

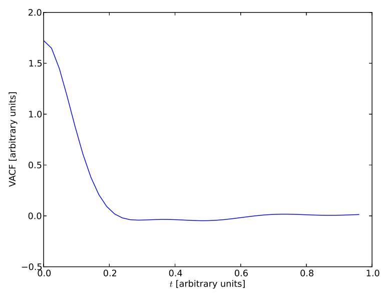
where VACF_{i+1} is the VACF at the next time index, and corresponding time t_{i+1}

The total integral is then the sum of all the individual integrals I_i;

$$\int_0^\infty \mathrm{VACF} \quad dt = \sum_i I_i = \sum_i \frac{1}{2}(\mathrm{VACF}_{i+1} + \mathrm{VACF}_i) \cdot (t_{i+1} - t_i) \tag{1.8}$$



(a)



(b)

Figure 1.2: (a) Uncorrelated VACF vs time. (b) VACF after the correlation function has been done.

2 Assignment

The assignment is to create a python simulation that calculates the diffusion coefficient from MSD and VACF, and simulate the coefficient over different variables.

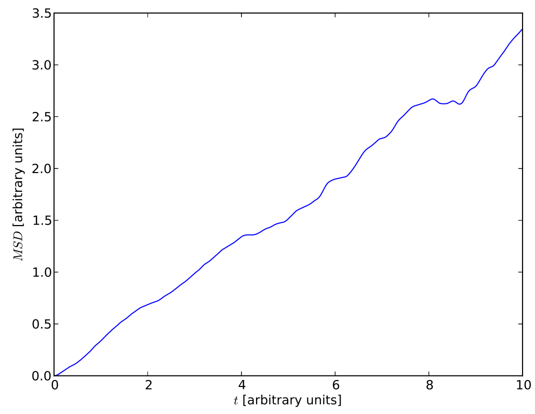
2.1 Simulation Setup

First part is to setup the simulation to calculate D. To make it easy to start we have split the setup into small manageable steps. An example for starting variables is

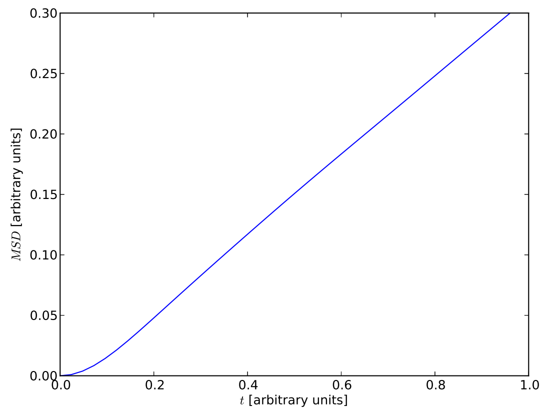
```
n_particles = 108
temperature = 0.5
rho = 0.7
equlib_steps = 10000
n_steps = 10000
sample_freq = 24
dt = 0.001
```

1. Define a sample frequency and save a list of position matrices `R` during the Velo-Verlet simulation, by finishing the `simulation` function.
2. Create a function that calculates the MSD between two coordinate matrices `R`.
3. Calculate MSD for each sampled `R` and plot MSD vs. simulation time. If you use the constants above it should look similar too figure 2.3a.
4. Implement the correlation function method for MSD and plot MSD vs. time. If you use the constants above it should look similar too figure 2.3b.
5. Use the correlated MSD list and the method for calculating the limit of MSD to calculate the diffusion coefficient.
6. Follow the same procedure for calculating the diffusion coefficient with the VACF method.

Hint: If you run your program with the settings suggested above, you will get $D \approx 0.05$.



(a)



(b)

Figure 2.3: (a) Uncorrelated MSD vs time. (b) MSD after the correlation function has been done.

2.2 Simulations

Now that you have a working simulation it is time to do some science. You need to do the following six simulations and conclude on the results (with plots).

- **Simulation 1**
  Check the convergence of the diffusion coefficient with simulation step number.
- **Simulation 2**
  Check the convergence of the diffusion coefficient with system size (particle numbers)
- **Simulation 3**
  Check MSD/VACF over time for given temperature at different densities.
- **Simulation 4**
  Check MSD/VACF over time for a given density ρ at different temperatures.
- **Simulation 5**
  Check the diffusion coefficient for a given temperature for different densities.
- **Simulation 6**
  Check the diffusion coefficient for a given density ρ for different temperatures.