Q1.

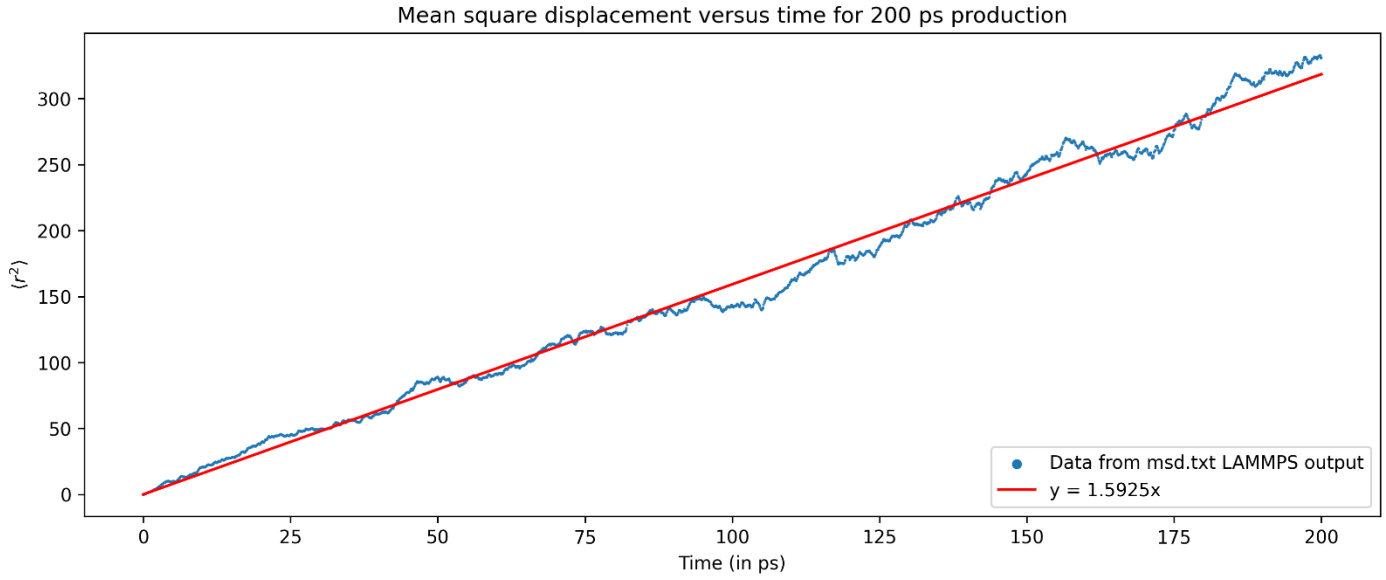
Zeroing the total linear momentum can be done in the following manner after defining the Maxwell velocity distribution for the system in the input script:

```
# Specify initial velocities to all atoms sampled uniformly from a distribution
# corresponding to 298 K temperature
velocity        all create 298.0 12345678 dist uniform
velocity        all zero linear
```

Q2.

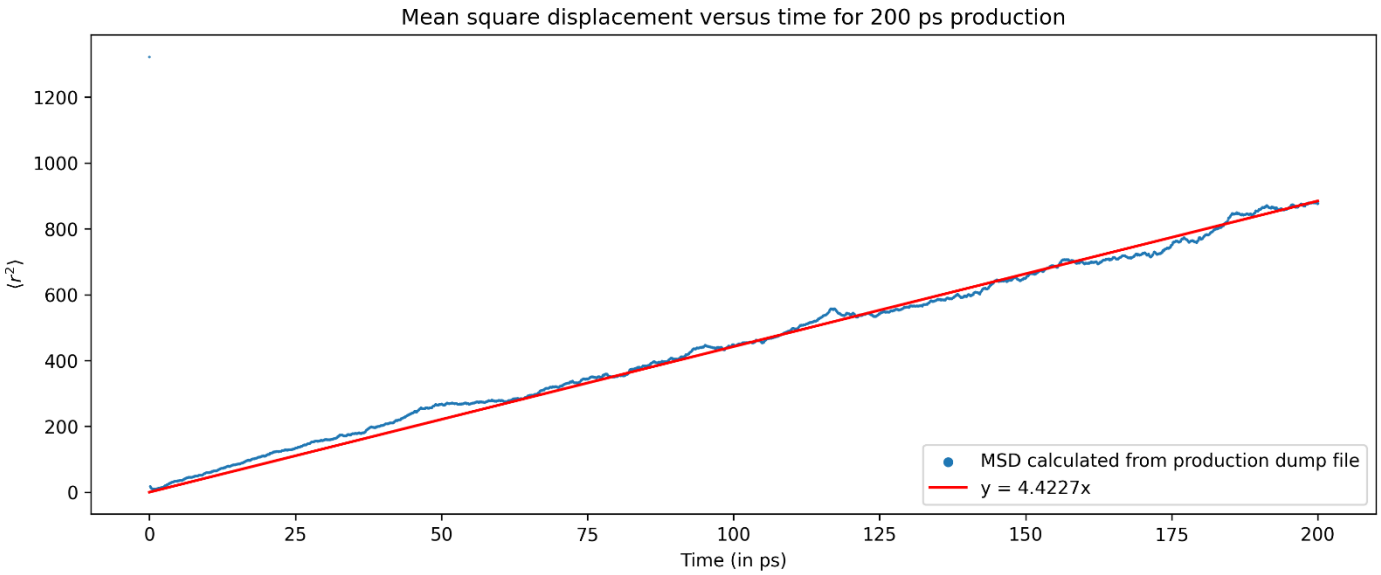
The diffusion constant can be calculated using three methods.

Following is the plot using method 1, where the msd.txt output file from LAMMPS was used directly for MSD values:



The data was fitted to a $y = mx$ plot, and we invoke the equation $\frac{\partial \langle r^2 \rangle}{\partial t} = 6D$ for particles moving in 3-dimensional space with self-diffusivity or diffusion constant D. From this equation and the fitted curve, D is calculated to be 0.2654 $Å^2$/ps i.e 2.65×10^{-9} m^2/s. This is quite close to the experimentally measured diffusion constant value of $2.34 \pm 0.05 \times 10^{-9}$ m^2/s.

For method 2, we use the position data from the dump file of the production step. For a particle defined at (x, y, z), the mean square displacement (MSD) will be = $\big(x(t) - x(0)\big)^2 + \big(y(t) - y(0)\big)^2 + \big(z(t) - z(0)\big)^2$. This will be found for each atom, and the total MSD for a single timepoint will be the mean of these individual sums. Looking up the documentation of how LAMMPS computes msd, I see that it employs two corrections in response to the `com yes average yes` keyword arguments specified in the input script: the first is a centre-of-mass (COM) correction to account for system drift, and the second includes a running average of the atom coordinates (for reasons I am not sure of). The resulting plot is as follows:



Here, we see that the data can be fitted to a linear curve with a slope of 4.4227 units. This results in a diffusion constant value of 0.7371 $Å^2$/ps i.e 7.34×10^{-9} m^2/s. This is quite a bit off from the experimental value. The following is the code block I have used to compute msd keeping these two in mind – discrepancies between the LAMMPS msd compute and my calculations might be due to specific considerations LAMMPS includes that I was unable to glean from the source code. `com[i][j]` list is the spatial coordinate (j = 0, 1, 2 for x, y, z respectively) at the i^{th} iteration.

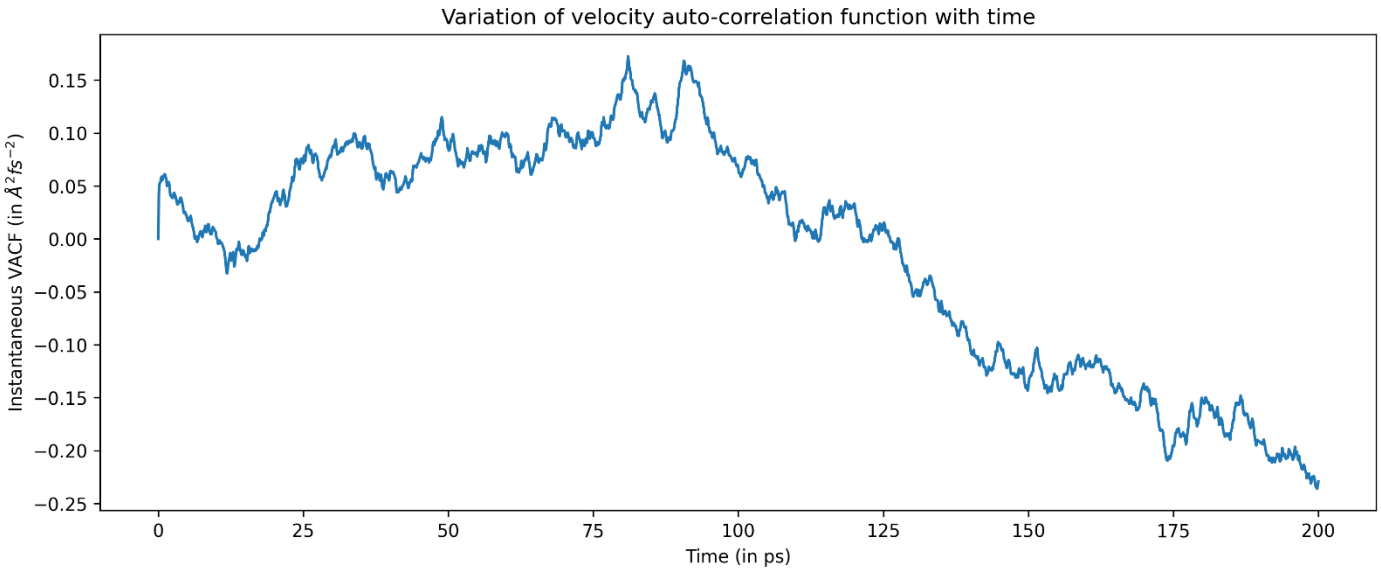
```
n = 0
runavg = np.zeros((2001,3))
for i in range(2001):
    msd_peratom = []
    for k in range(0, 348):
        x_temp = prodtimesteps[i]['data'][k][3] - com[i][0] - runavg[i][0]
        y_temp = prodtimesteps[i]['data'][k][4] - com[i][1] - runavg[i][1]
        z_temp = prodtimesteps[i]['data'][k][5] - com[i][2] - runavg[i][2]
        runavg[i][0] = (runavg[i][0] * n + x_temp)/(n + 1)
        runavg[i][1] = (runavg[i][1] * n + y_temp)/(n + 1)
        runavg[i][2] = (runavg[i][2] * n + z_temp)/(n + 1)
        x_ref = prodtimesteps[0]['data'][k][3] - com[0][0]
        y_ref = prodtimesteps[0]['data'][k][4] - com[0][1]
        z_ref = prodtimesteps[0]['data'][k][5] - com[0][2]
        msd_peratom.append((x_temp - x_ref)**2 + (y_temp - y_ref)**2 + (z_temp - z_ref)**2)
    n += 1
    msdt.append(np.mean(msd_peratom))
```

Another method to find self-diffusivity is to use the velocity auto-correlation function (VACF), which relates to diffusion constant by the following Green-Kubo relation:

$$D = \frac{1}{3}\int_0^\infty d\tau \langle v_x(\tau) v_x(0) \rangle$$

This expression is adapted from the expression in the lecture slides (there the relation was only considering movement along the *x*-direction).

Instantaneous VACF can be calculated by LAMMPS through the `compute vacf` command, and following is a plot of VACF vs time:



From here, using the Green-Kubo relation, we get D = −638.0 $Å^2$/fs i.e. -6.38×10^{-9} m^2/s. This, while being someways off from the magnitude of the experimentally measured value, is also a negative number. Diffusion constant cannot be a negative number.

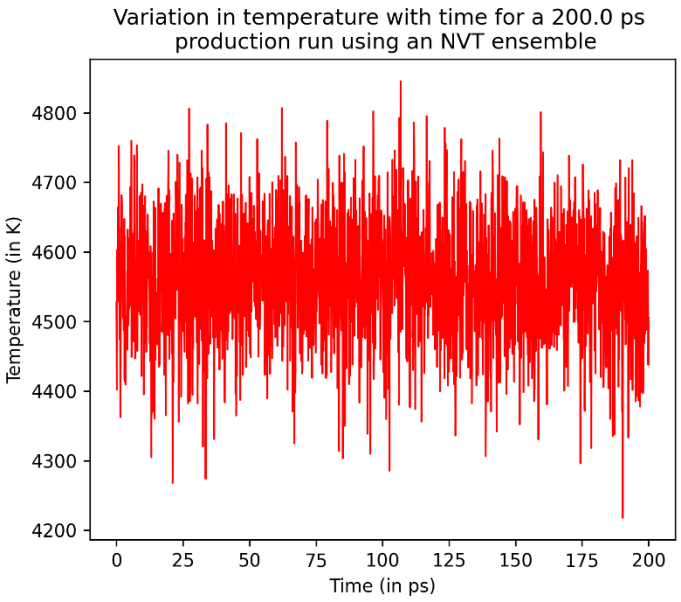
The Materials Science Community Discourse forum addresses this (https://matsci.org/t/negative-diffusion-coef-compute-vacf-command/25203), and I quote: “ Yes, but you have to numerically integrate the vacf out to very long times since vacf is a very slowly decaying function. You got to integrate everything under the very long tail. This makes it not so reliable for estimating transport properties. If you didn’t run your simulations long enough, I wouldn’t be surprised if you got unreasonable diffusion coefficients. Just use mean square displacement. It is much more straightforward, faster, and more accurate.” (answered as a response to a query confirming that D is related to the time integral of VACF).

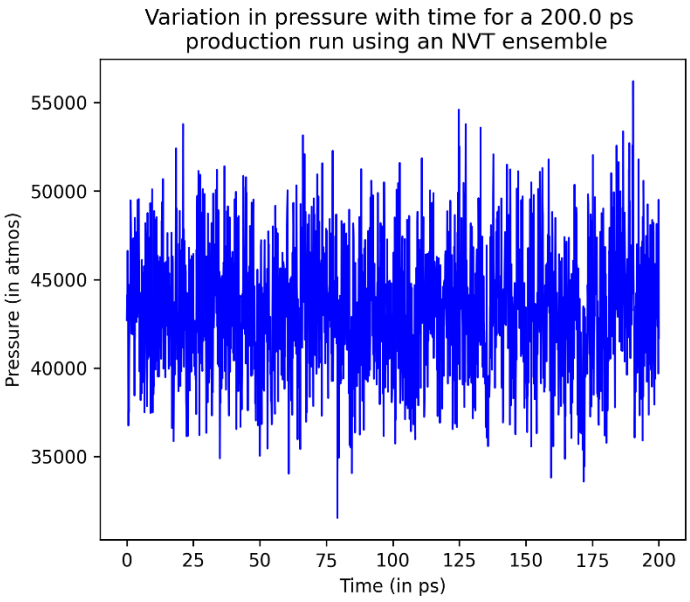
We have run our simulation for 200,000 timesteps, and I am inclined to believe that this is sufficiently long enough to model a simple water-molecules-diffusing system. I hence do not have an explanation for the negative value of D.

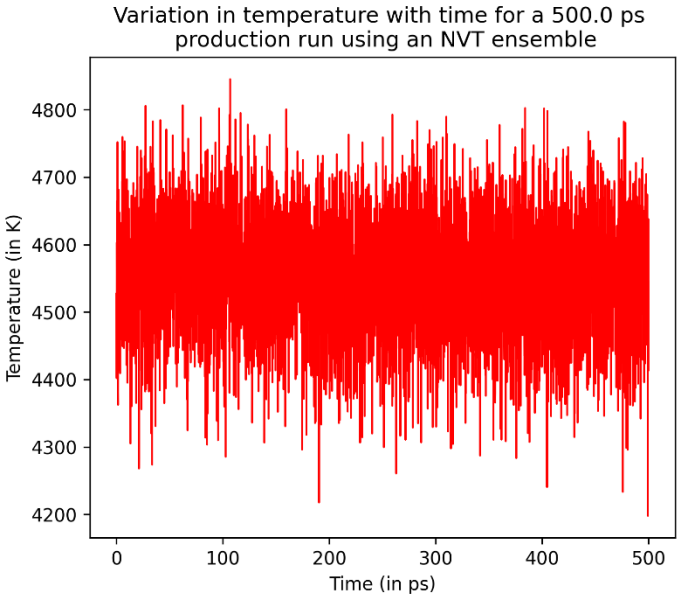
Q3.

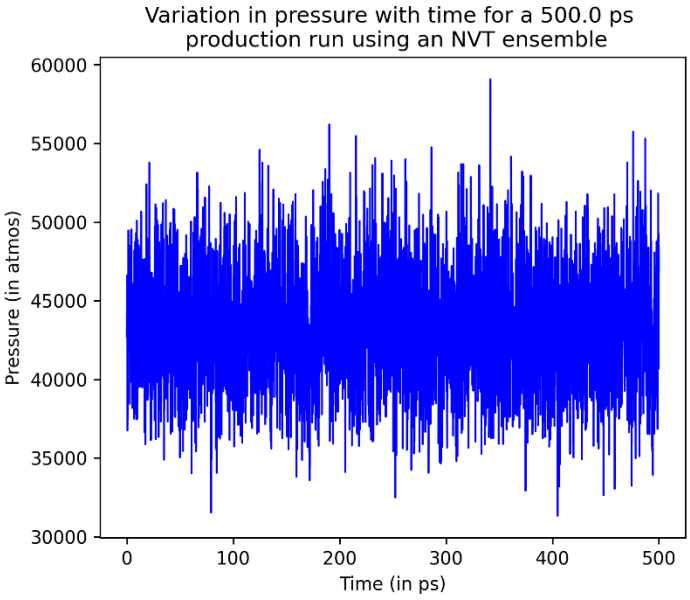
The system now is an unrestrained NVE ensemble i.e. the thermostat and barostat have been removed. This was achieved by removing the `lang_thermo` and `prbe` fixes in the input script.

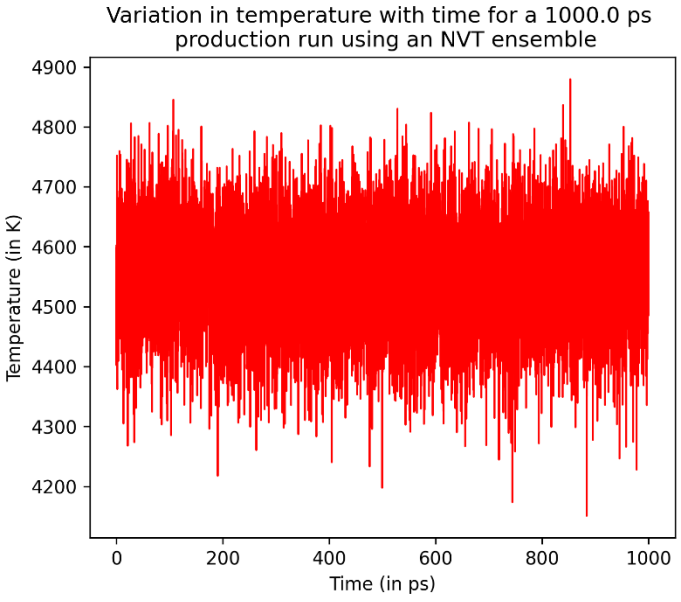
I have varied the production time to be 200,000, 500,000 and 1,000,000 timesteps (200ps, 500ps, 1ns respectively) to check if the VACF compute improves. In order to see how temperature and pressure are then affected, I generated plots of each of these variables versus time.

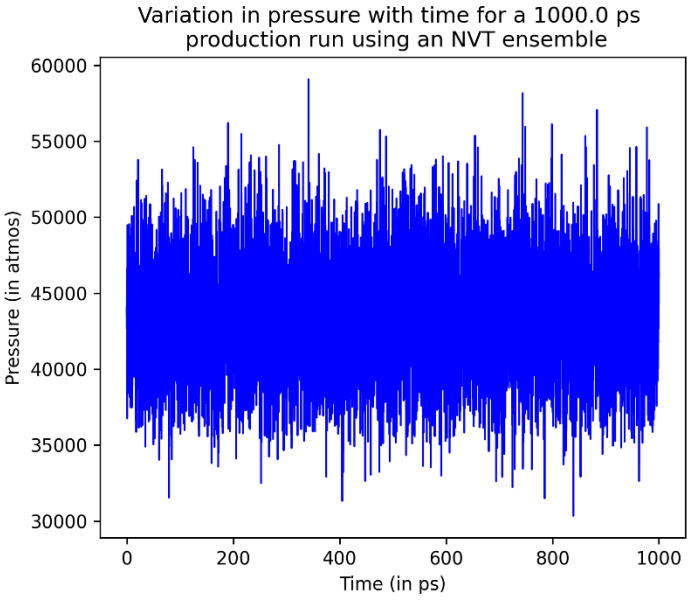






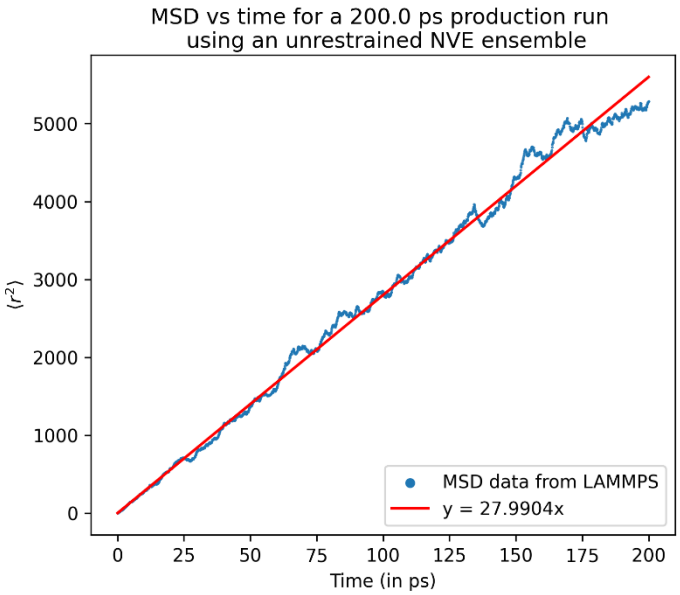


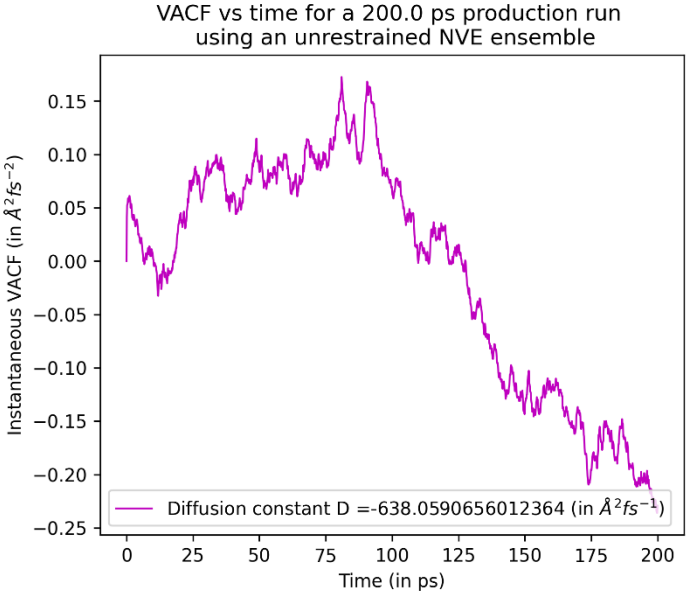


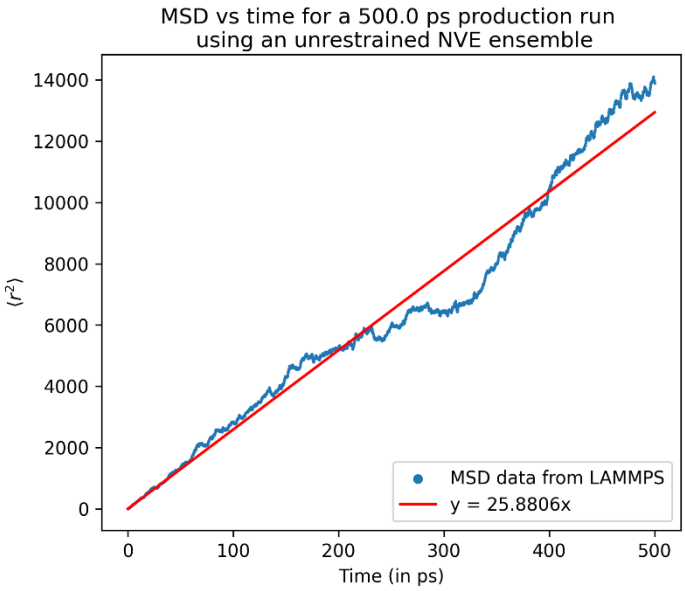


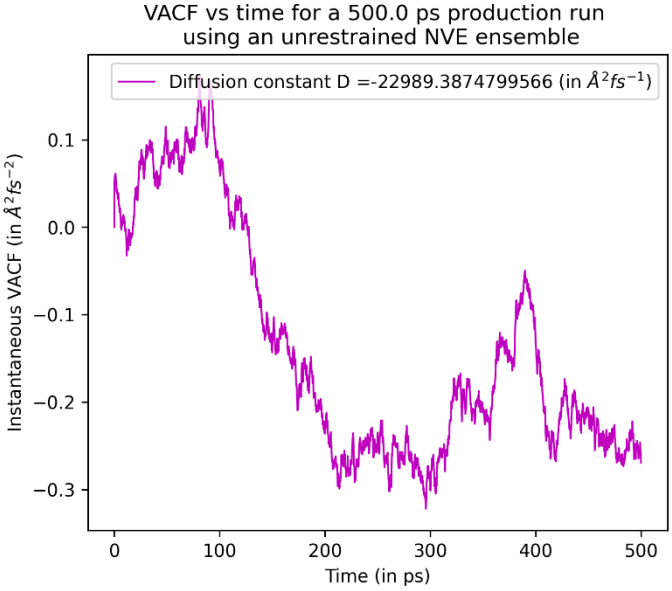
We see that temperature is fluctuating wildly between ~4400 to 4700 K regardless of simulation time – a ridiculous situation! Even though the initial velocity space has been defined with respect to 298 K, we see that the system is unable to maintain this temperature in any capacity.

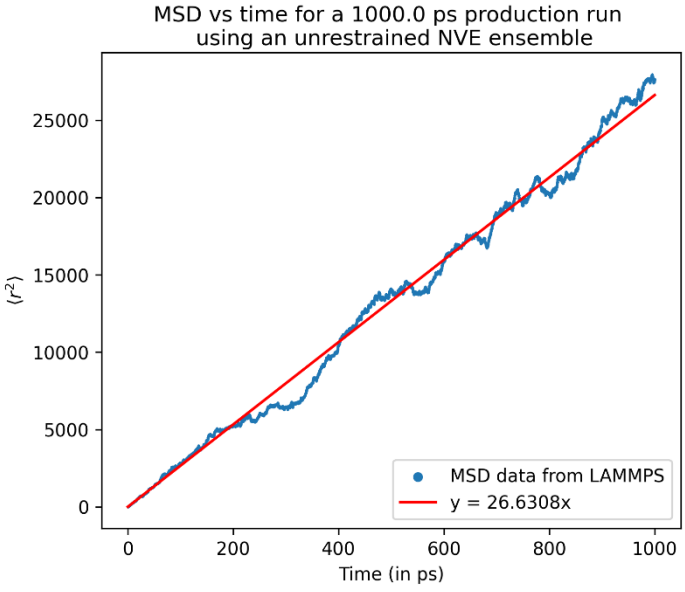
To calculate the diffusion constants, I have employed the previously mentioned three methods. Following are the plots using the MSD compute from LAMMPS, and the VACF time integral. As we have seen before, VACF integral can give wildly incorrect results, so the results here are only presented to contrast the methodologies.

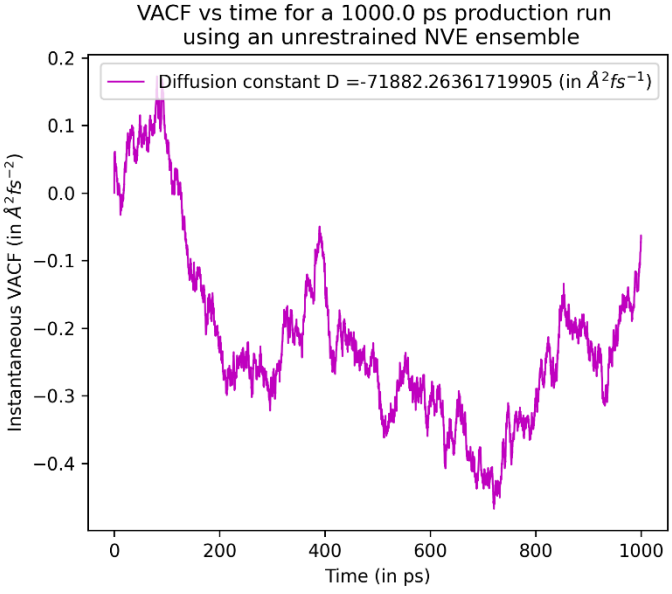




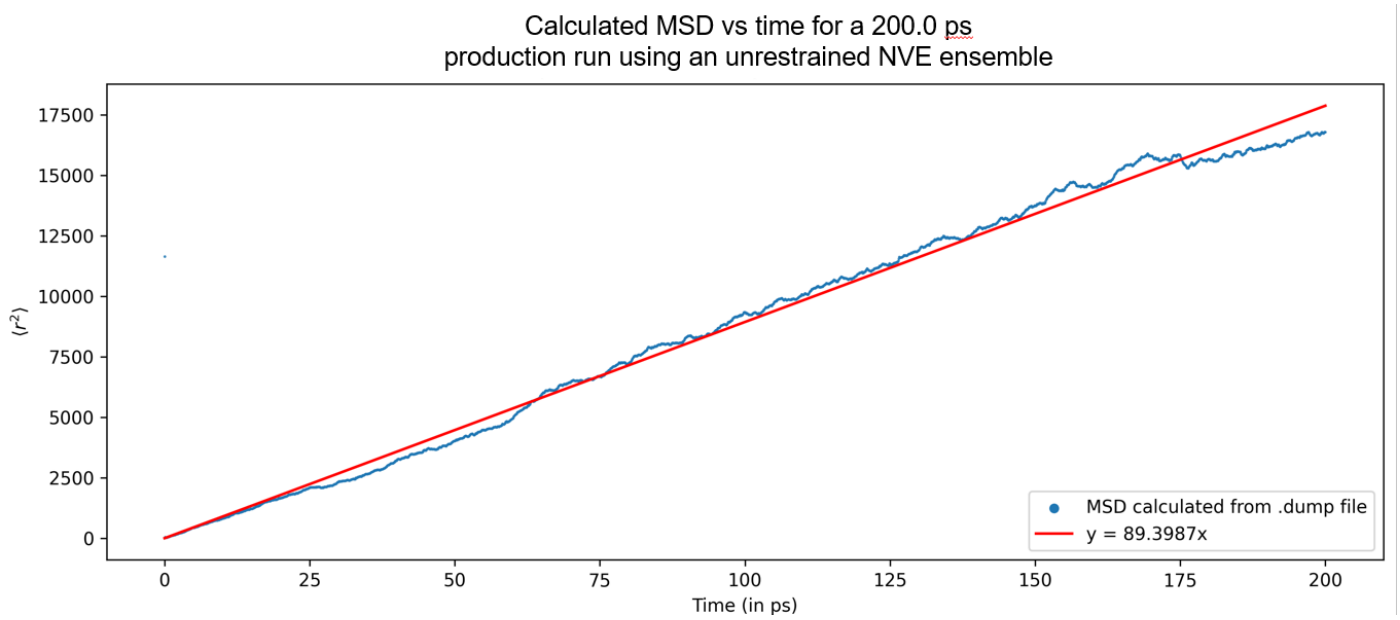


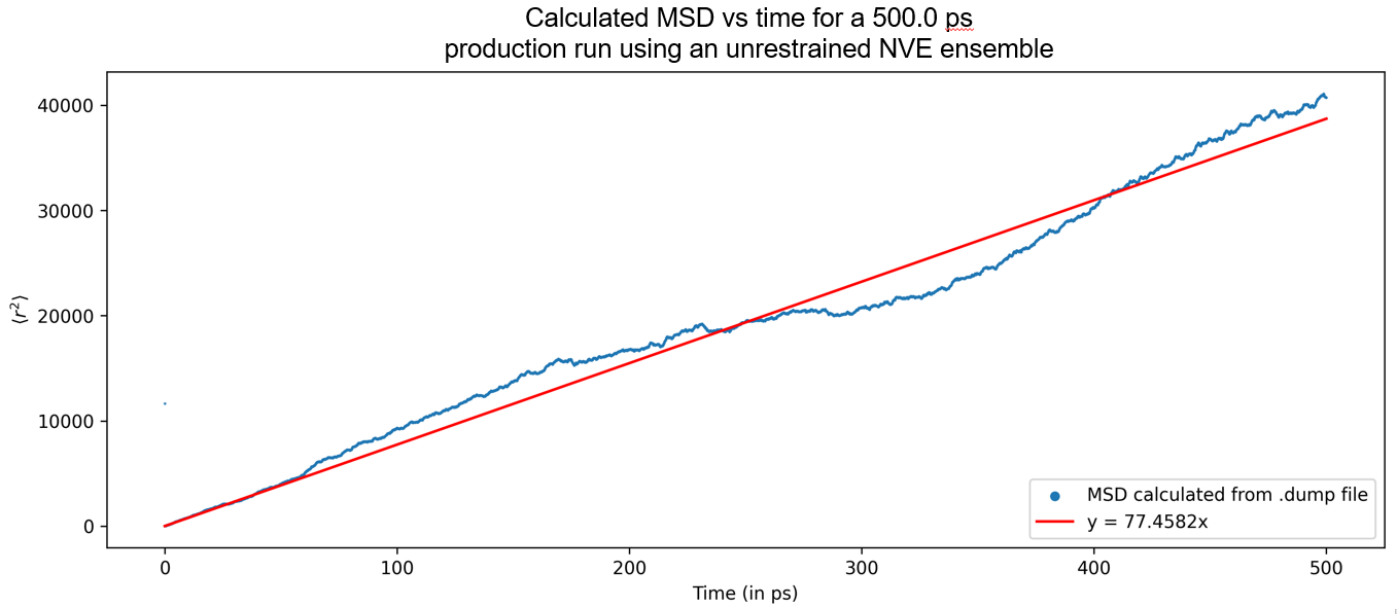


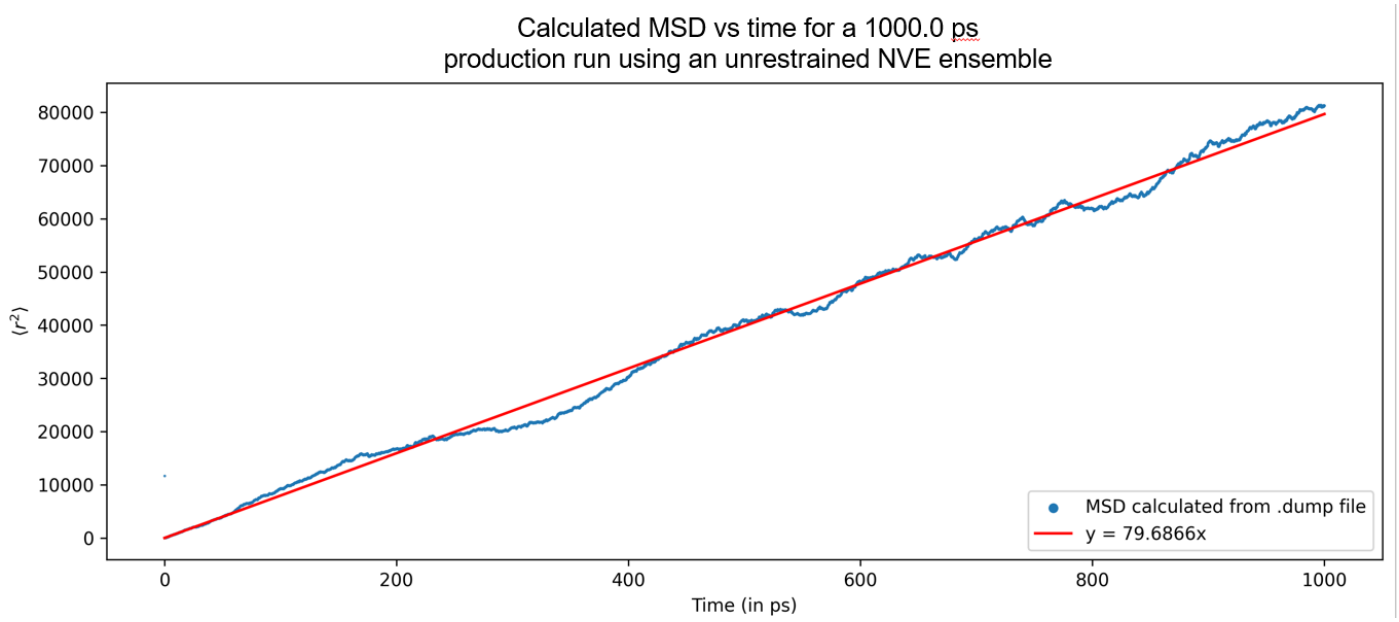




Below are the plots for MSD calculated from position data as shown previously vs time:







The following table tabulates the calculated diffusion constants (in units of 1×10^{-9} m^2 s^{-1}):

	200 ps	500 ps	1000 ps

MSD (LAMMPS)	46.65	43.13	44.38
MSD (calculated)	149.00	129.10	132.81
VACF	−6.38	−22.99	−71.88

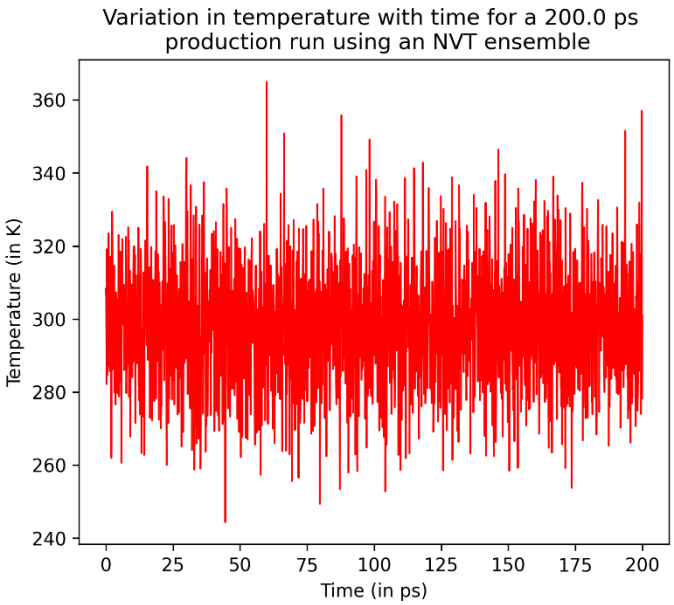
Q4.

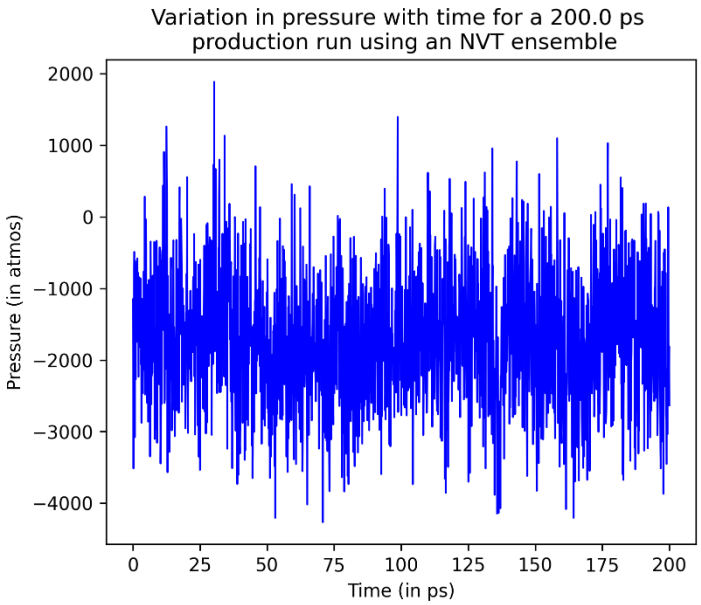
For an NVT ensemble, the following fix was implemented in the input script:

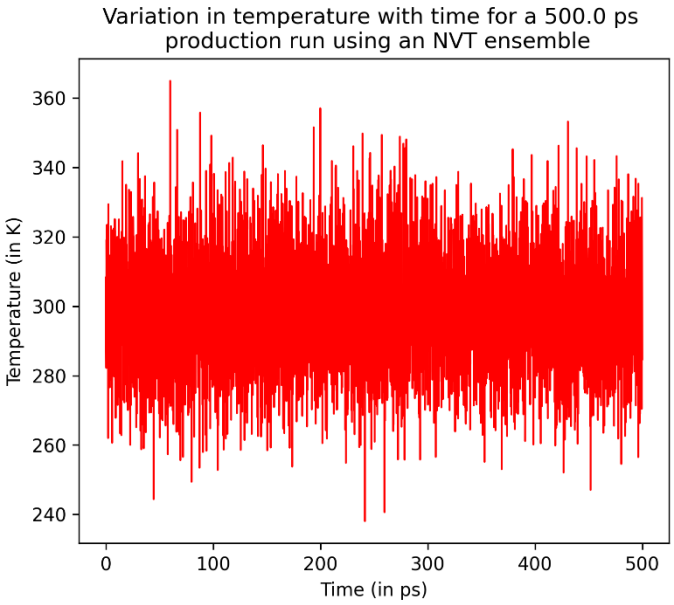
```
# Set-up NVT simulations with a Nose-Hoover thermostat
fix             1 all nvt temp 298.0 298.0 100.0
```

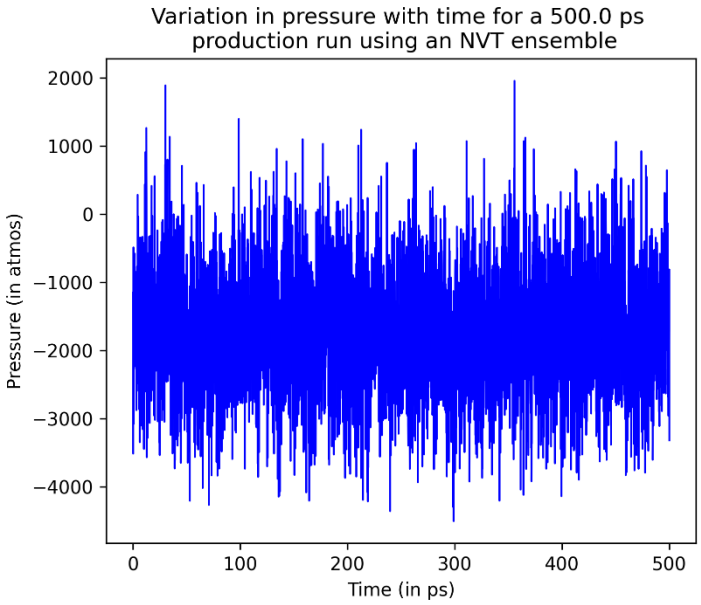
The damping constant (third argument after temp) relates to the frequency at which the thermostat corrects the temperature. A value of 100.0 means that after every 100 timesteps, the thermostat acts. This is the value recommended by the LAMMPS documentation, and it makes sense for our simulation as well: we are outputting thermodynamic and position data every 100 timesteps.

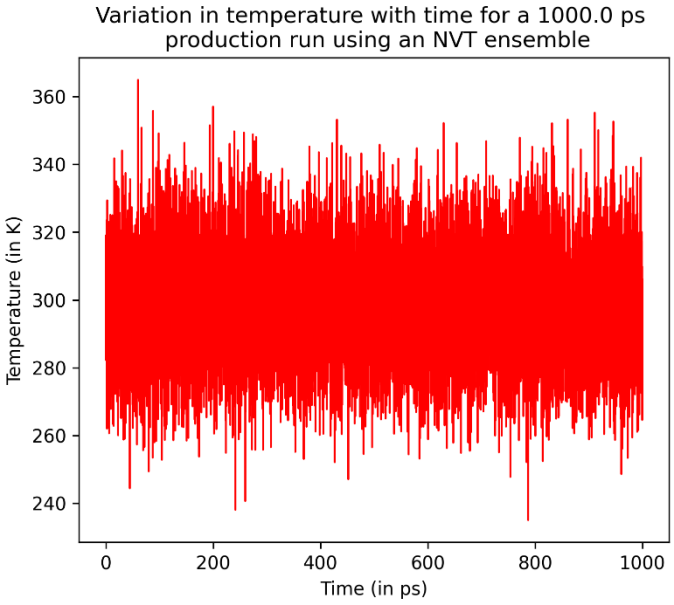
Following a similar logic as in Q3, I have generated plots of T & P vs time as follows:

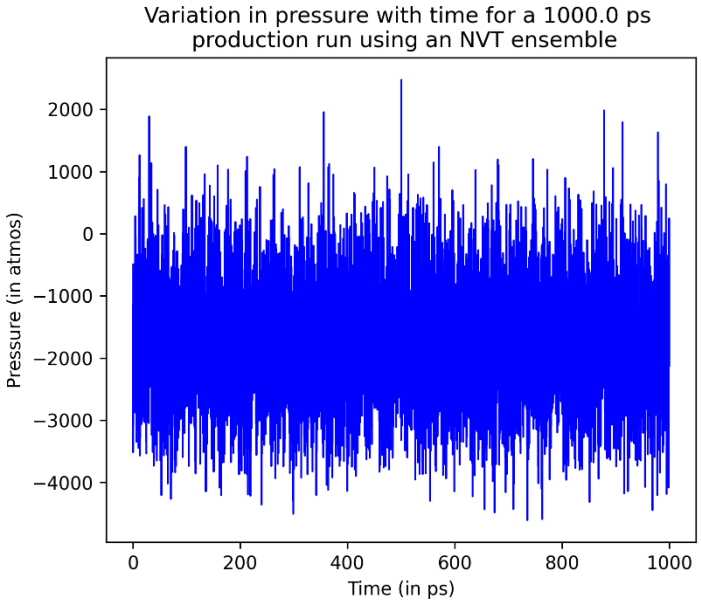






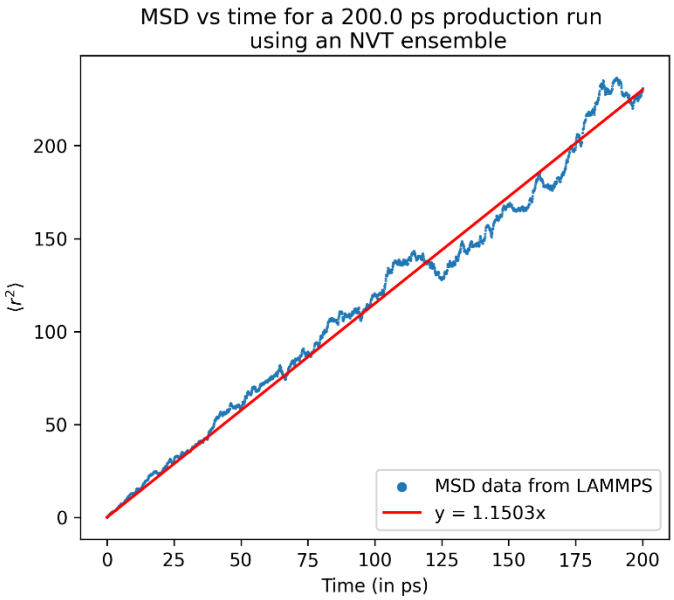


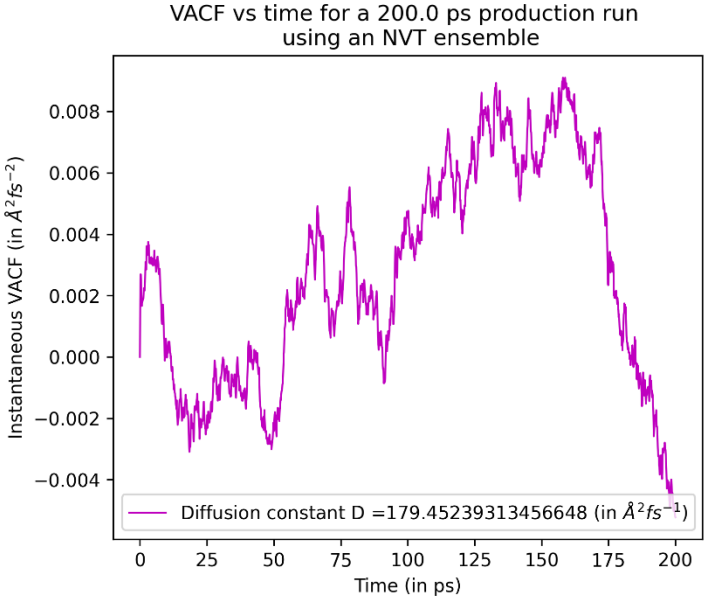


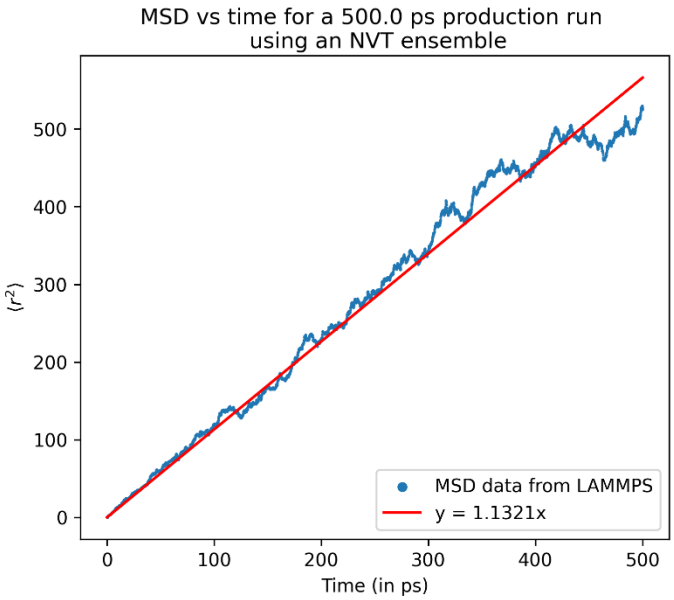


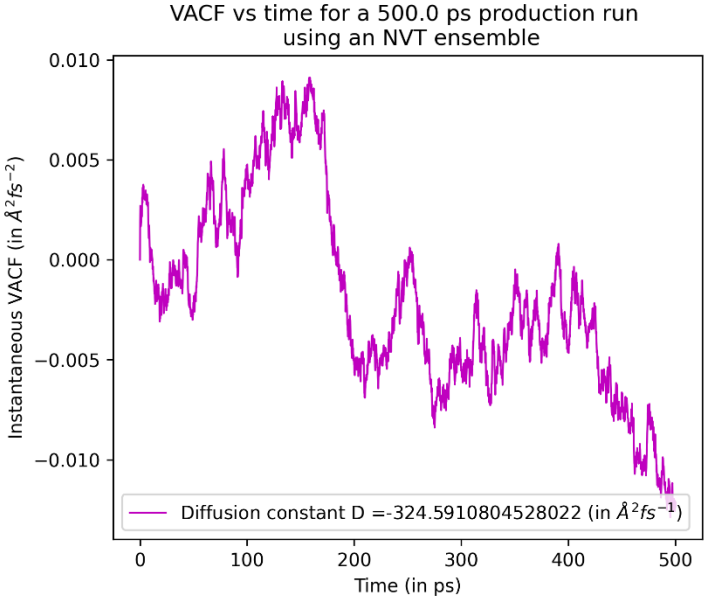
Now, we see that temperature fluctuates around our desired temperature of ~300 K. The fluctuations are significant; however, I believe lowering the dampening constant value will then interfere with the natural evolution of the system under the force field. It is interesting to note that pressure is often negative.

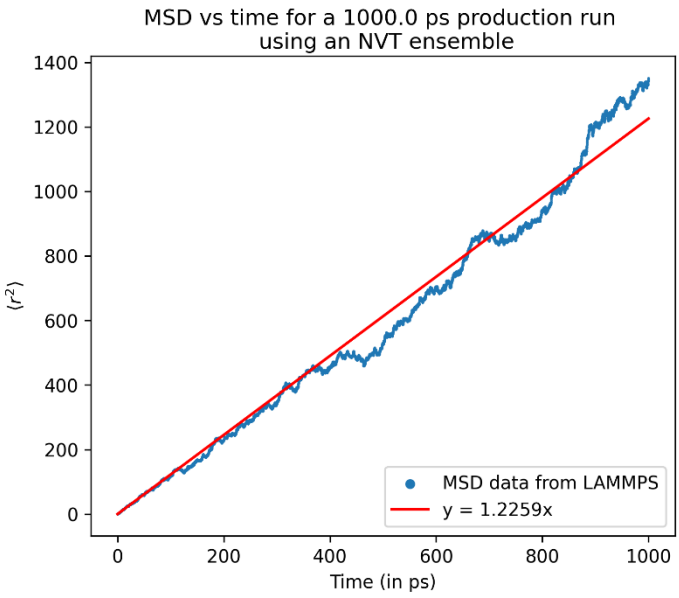
Following are the plots of MSD and VACF vs time to estimate the diffusion constants:

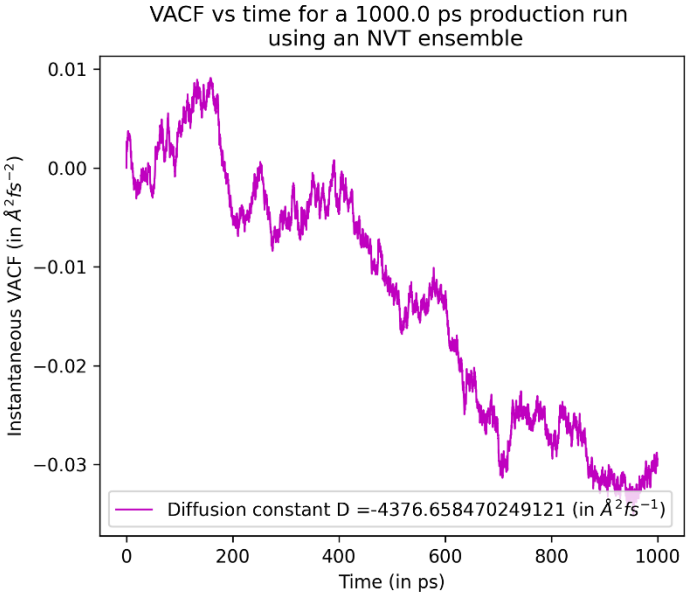




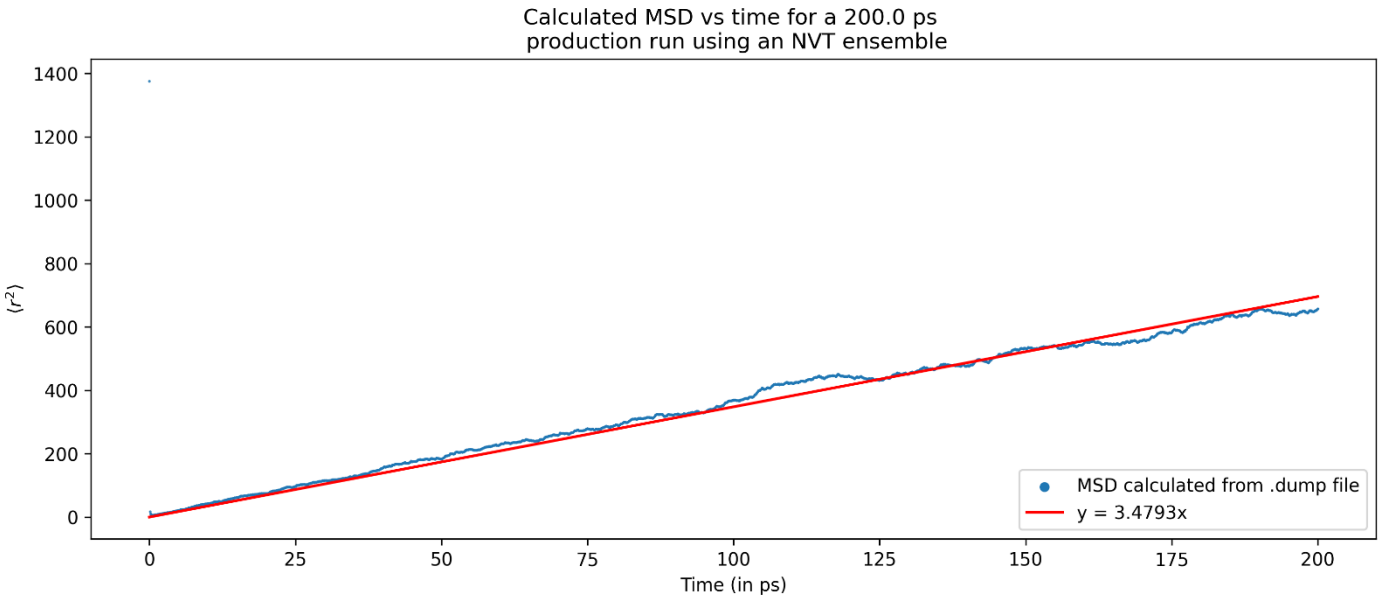


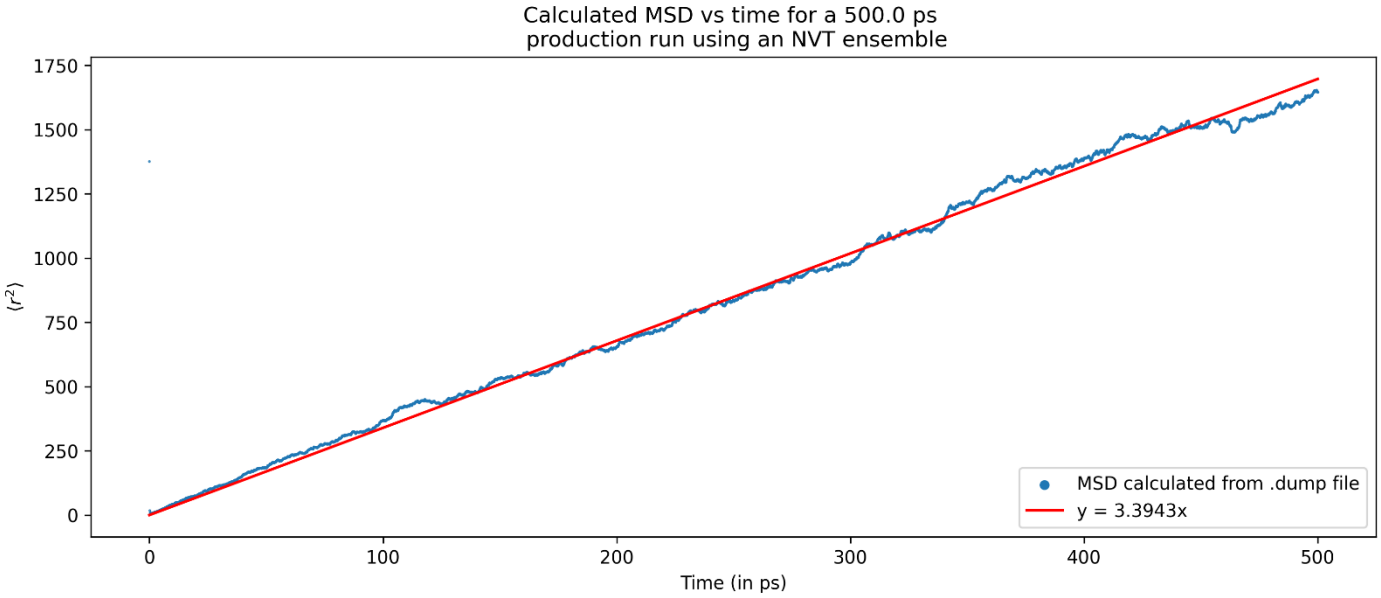


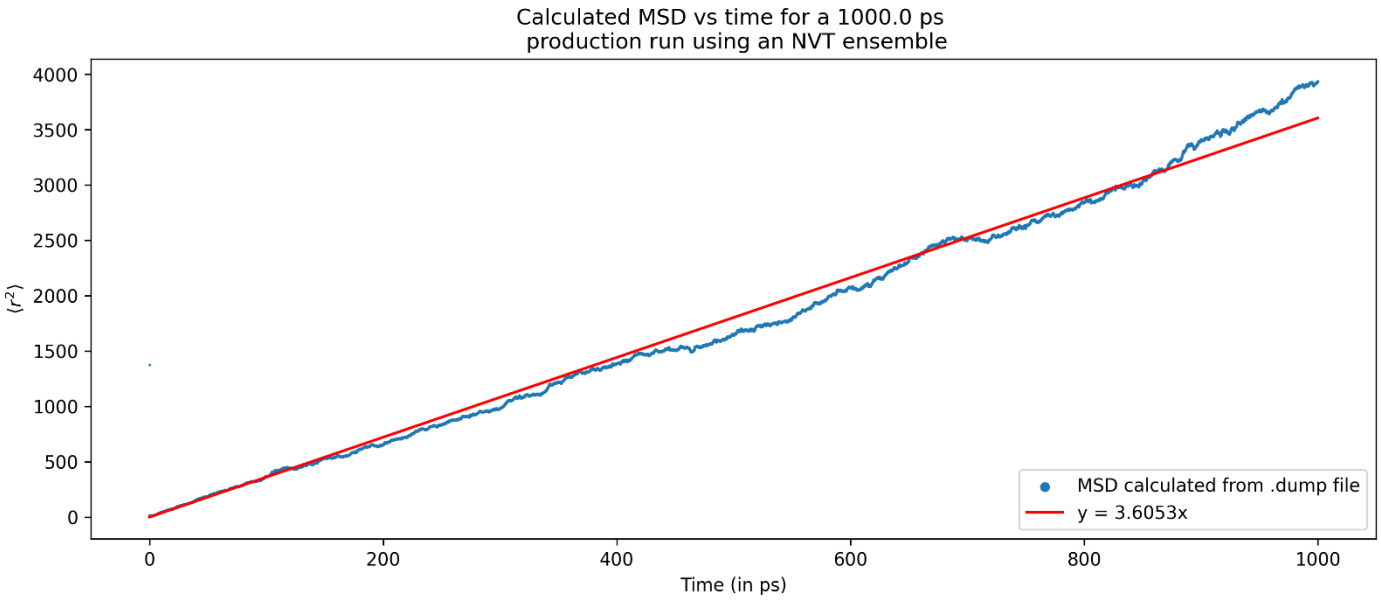




Following are the plots for $MSD_{calculated}$ vs time:







The diffusion constants calculated can be tabulated as follows (in units of 1×10^{-9} m^2 s^{-1}):

	200 ps	500 ps	1000 ps
MSD (LAMMPS)	1.91	1.89	2.04
MSD (calculated)	5.80	5.66	6.01

VACF	1.79	–3.24	–43.76

To summarize this assignment, I will tabulate all the diffusion constant values calculated over various ensembles and time steps. Values are again in units of 1×10^{-9} m^2 s^{-1}

	NVE	**NVE (no thermo- or barostat)**			**NVT**		
	200 ps	**200 ps**	**500 ps**	**1000 ps**	**200 ps**	**500 ps**	**1000 ps**
MSD (LAMMPS)	2.65	46.65	43.13	44.38	1.91	1.89	2.04
MSD (calculated)	7.34	149.00	129.10	132.81	5.80	5.66	6.01
VACF	–6.38	–6.38	–22.99	–71.88	1.79	–3.24	–43.76

We can make the following conclusions:

1. VACF is a terrible way to estimate diffusion constant; MSD as outputted from LAMMPS gives better results
2. The simulation accuracy follows the trend: NVE > NVT > unrestrained NVE