Project

Snehal Kumar

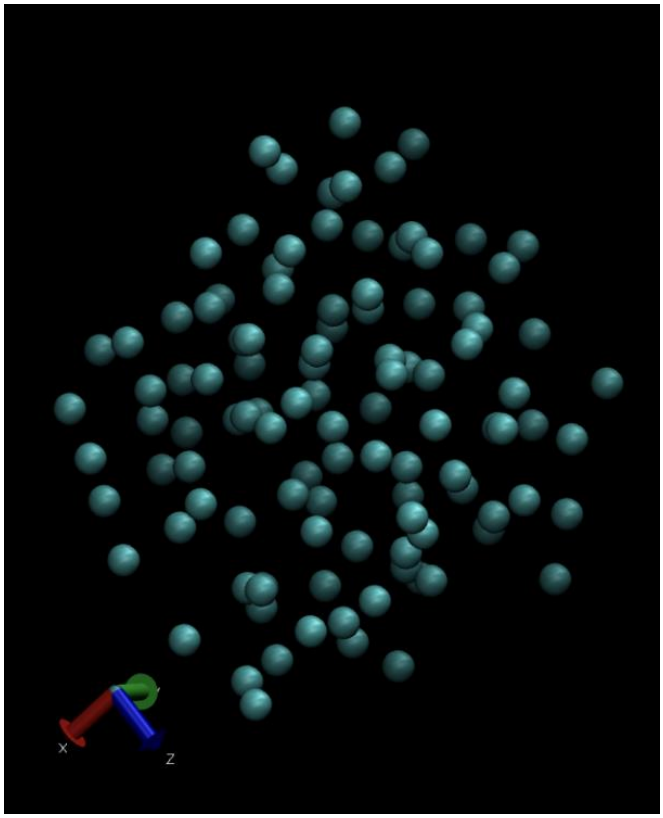
2019101003

Question 1: Generate a random initial configuration given:

- N = 108
- Lx = Ly = Lz = 18.0 A
- Lennard Jones Energy Parameter = 0.238 kcal/mol
- Radius = 3.4 A
- Distance between any pair of atoms at least 3.4 A

Using Periodic Boundary Conditions (PBC), we generate 108 random points (genconf.py) which satisfy the constraints. The configuration is saved in generated.xyz.

Using VMD, we can see the output:



Question 2: Calculate the Lennard Jones Interaction Energy per pair.

The Lennard Jones Potential is an intermolecular pair potential which is defined as:

$$V_{LJ}(r) = 4 \in \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

The pseudo code:

```
def calc_energy(geom):
    eps = 0.238
    sigma = 3.4
    te = 0
    tgeom = np.array(geom)
    pairs = [(a, b) for idx, a in enumerate(tgeom)
             for b in tgeom[idx + 1:]]
    for pair in pairs:
        rij = np.linalg.norm(pbc_sep(pair[0], pair[1]))
        if rij == 0:
            continue
        te += 4*eps*((sigma/rij)**12-(sigma/rij)**6)
```

By taking the sum of all interaction energies, we calculate the energy as: -123.84

Question 3: Minimize total potential energy of system

Using the minimum-image convention (form of PBC), we update the coordinates of the atoms while calculation. To minimize the total energy, we use Steepest Descent Algorithm which takes in parameters for tuning and performs a greedy local search to find the global minima. The implementation can be found in gradient.py.

```
gradient = grad(get_energy)

weight_history = [w]
cost_history = [get_energy(w)]
for k in range(max_its):
    grad_eval = gradient(w)
    w = w - alpha*grad_eval
    print(get_energy(w))
    weight_history.append(w)
    cost_history.append(get_energy(w))
```

The final minimal potential energy: -149.891

Question 4: Calculate the Hessian matrix and get eigenvalues and eigenvectors

The Hessian matrix is defined as:

$$\mathbf{H}_f = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1\,\partial x_n} \\ \frac{\partial^2 f}{\partial x_2\,\partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2\,\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n\,\partial x_1} & \frac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}, \quad (\mathbf{H}_f)_{i,j} = \frac{\partial^2 f}{\partial x_i\,\partial x_j}.$$

It is defined as the Jacobian of the gradient. The calculation to generate the Hessian is in hessian.py. We use the new_molecule.xyz to calculate the matrix. From the matrix, we use the numpy library to calculate the eigenvalues and eigenvectors which is saved in eigen_values.dat, eigen_vectors.dat respectively

Question 5: Get the histogram of vibrational frequencies

From the Hessian matrix and the eigen values and eigenvectors calculated in the previous question, we can use them to calculate the normal mode frequencies. The implementation can be found in frequency.py

Let N be the number of atoms and let m_A and (x_A, y_A, z_A) be the mass and Cartesian coordinates of the A^{th} atom.

$$(\mathbf{H})_{AB} = \frac{\partial^2 E}{\partial X_A \partial X_B} \qquad (X_{3A-2}, X_{3A-1}, X_{3A-0}) = (x_A, y_A, z_A) \qquad \text{for } A \in \{1, \ldots, N\}$$

$$(\tilde{\mathbf{H}})_{AB} = \frac{(\mathbf{H})_{AB}}{\sqrt{M_A M_B}} \qquad (M_{3A-2}, M_{3A-1}, M_{3A-0}) = (m_A, m_A, m_A) \qquad \text{for } A \in \{1, \ldots, N\}$$

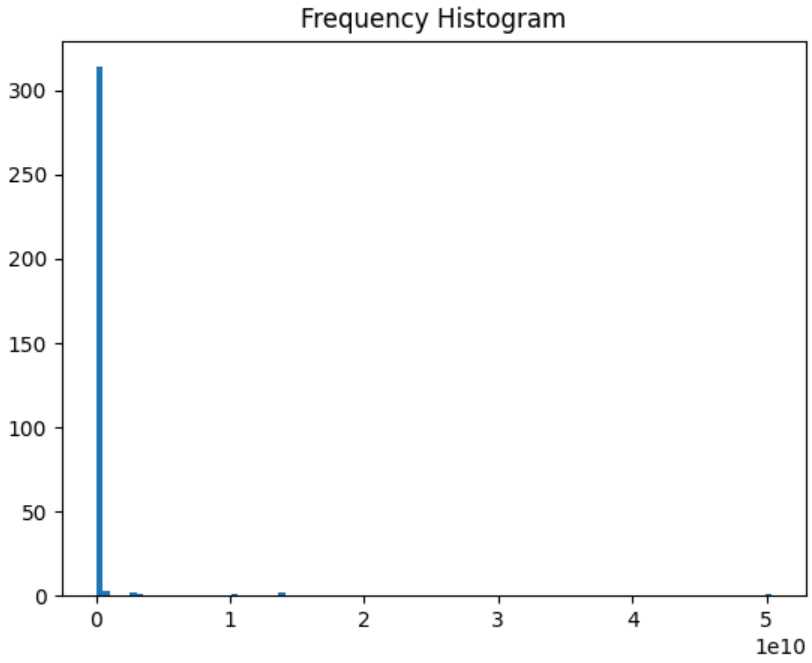
$$\tilde{\mathbf{H}}\,\tilde{\boldsymbol{q}}_A = k_A \tilde{\boldsymbol{q}}_A \qquad \text{for } A \in \{1, \ldots, N\}$$

$$(\boldsymbol{q}_A)_B = \frac{(\tilde{\boldsymbol{q}}_A)_B}{\sqrt{M_B}} \qquad \text{for } A, B \in \{1, \ldots, N\}$$

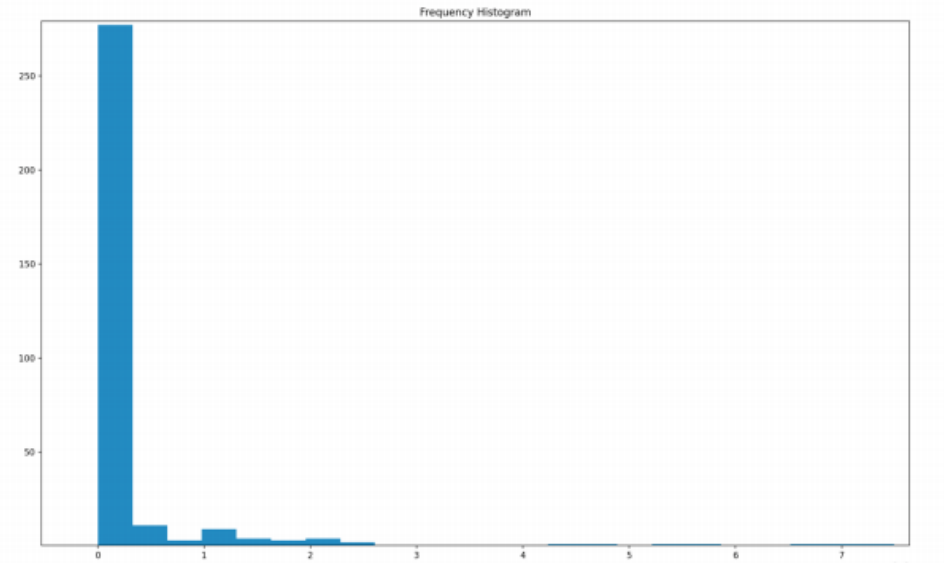
$$k_A \left[\frac{E_h \text{rad}^2}{a_0^2 u}\right] = k_A \left(\frac{E_h[\text{J}]}{a_0[\text{m}]^2 u[kg]}\right) \left[\frac{\text{rad}^2}{\text{s}^2}\right] \qquad \nu_A\,[\text{Hz}] = \frac{1}{2\pi} \left[\frac{\text{cyc}}{\text{rad}}\right] \cdot \sqrt{k_A \left[\frac{\text{rad}^2}{\text{s}^2}\right]} \qquad \tilde{\nu}_A\,[\text{cm}^{-1}] = \frac{\nu_A\,[\text{Hz}]}{c\,[\text{cm/s}]}$$

Using this information, we implement the calculation and plot the histogram to get an idea about the frequency distribution. The normal modes are stored in normalmodes.xyz

Plotting histogram for frequencies:



We can see that some modes are very frequent but others are not


Frequency Histogram
250
200
150
100
50
0
1
2
3
4
5
6
7