Molecular Dynamic Simulations in JavaScript

Daniel Kunin

Abstract

Using a Newtonian mechanical model for potential energy and a force-based layout scheme for graph drawing I investigated the viability and accuracy of a web-based molecular dynamics visualization. I implemented such a simulation in JavaScript for diatomic molecules and suggested improvements moving forward to expand the range and effectiveness of such a visualization.

supervised by

Dr. Matthew Harrison

February 1, 2017

Contents

1 Introduction

For my senior capstone project in Graphs and Networks, I will create a browser-based, dynamic, 3-dimensional simulation of organic molecules. Organic molecule's can be thought of as graph networks where nodes represent atoms in the molecule and edges are covalent bonds between these atoms. In fact, *chemical graph theory* is a well established field that uses algorithms and insights of graph theory and applies it to molecular interactions.

Inspired by energy based algorithms for visualizing networks, my goal is to augment the classic ball and stick representation of organic molecules to reflect the dynamic relationship of atoms. Using a force-based layout scheme for graph drawing, I will create a 3-dimensional simulation that accurately displays the constant harmonic movement of molecules around their numerous local energy minimums. The major changes I will be implementing to the classic force-based layout scheme are:

1. Defining the energy function according to the physics of intramolecular forces
2. Removing the decay or friction term such that the graph (molecule) never reaches a stable equilibrium
3. Visualizing the position vectors of each atom in three dimensions as the simulation runs

I will be designing the project as a front-end web application such that it can be run entirely on a web browser. Web technologies are by far the most effective means of getting software utilized. While there already exists many *molecular dynamics simulation softwares*, very few of these are accessible or completely web-based. Recent advancements in web technologies allow for complex physics simulations, such as this project, to be run entirely on a user's browser while not forfeiting complexity.

2 Model

I will use a Newtonian mechanical model for the potential energy and molecular motion of a molecule. Additionally, I will use the Velocity Verlet algorithm to solve the initial value problem for the motion of atoms in a molecule. Below is a more detailed explanation of this method and necessary definitions.

2.1 Potential Energy

The potential energy of a molecule can be represented as the sum of five different potential energy functions [3].

$$E = E_{bond} + E_{angle} + E_{torsion} + E_{electro} + E_{vdw}$$

Below are the definitions for these five potential energy functions:

$$\begin{aligned}
E_{bond} &= \sum_{(i,j)\in S_{bonds}} k^b_{ij}(r_{ij} - r^0_{ij})^2 \\
E_{angle} &= \sum_{(i,j,k)\in S_{angle}} k^a_{ijk}(\alpha_{ijk} - \alpha^0_{ijk})^2 \\
E_{torsion} &= \sum_{(i,j,k,l)\in S_{torsion}} k^t_{ijkl}[1 + cos(n\alpha_{ijkl} - \alpha^0_{ijkl})] \\
E_{electro} &= \sum_{(i,j)\in S_{electro}} \frac{q_i q_j}{e_{ij} r_{ij}} \\
E_{vdw} &= \sum_{(i,j)\in S_{vdw}} \epsilon_{ij}\big[\big(\frac{\sigma_{ij}}{r_{ij}}\big)^{12} - 2\big(\frac{\sigma_{ij}}{r_{ij}}\big)^{6}\big]
\end{aligned}$$

I will start by focusing on the energy functions above that depend only on pairs of atoms ($E_{bond}, E_{electro}, E_{vdw}$). This will be sufficient for modeling diatomic molecules such as halogen molecules (i.e. $H_2, F_2, Cl_2, Br_2, I_2$) or heteronuclear molecules (i.e. CO, NO). From here I can then add the energy functions that depend on triplets and quadruplets of atoms ($E_{angle}, E_{torsion}$), enabling simulations of more complicated molecules.

2.2 Determining Constants

Determining the correct constants used in the potential energy functions is essential for the accurateness of the simulation. These constants depend on the specific atoms in the molecule and many of the constants are unknown or roughly approximated through experimentation. Because I will be focusing on diatomic molecules, I will start by defining the following constants: $k^b_{ij}, e_{ij}, \epsilon_{ij}, \sigma_{ij}$.

- k^b_{ij} - I will approximate this constant through Badger's Rule. Badger's Rule is an expression for approximating the bond force constant for diatomic molecules [1].

- e_{ij} - I will make the assumption that this constant is Coulombs Constant ($8.98755 * 10^9 Nm^2/C^2$) for all pairs of molecules.
- $\epsilon_{ij}, \sigma_{ij}$ - I will determine these constants (Lennard-Jones parameters) from published experimental results [2].

2.3 Equation for Molecular Motion

Using Newton's Second Law ($F = ma$), we can define the molecular motion of each atom in the molecule as the following.

$$m_i x_i'' = f_i(x_1, x_2, ..., x_n)$$
$$m_i x_i'' = -\frac{\partial E}{\partial x_i}$$

where m_i = mass of atom i, x_i = position vector of atom i

If we define the potential energy function only in terms of $E_{bond}, E_{electro}$, and E_{vdw}, we can then evaluate the potential energy as a function of r_{ij}, the distance between atoms i and j. Under this scheme, the potential energy function is partially separable, and we can define the partial derivates of E as the following:

$$\frac{\partial E}{\partial x_k} = \sum_{j=k+1}^{n} \frac{E'(r_{jk})x_k}{r_{jk}} - \sum_{i=1}^{k} \frac{E'(r_{ik})x_k}{r_{ik}}$$

2.4 Initial Parameters

By defining the initial position(x_i^0) and velocity(v_i^0) of all atoms in the molecule, their will be a unique solution to the equation of molecular motion defined above.

2.4.1 Position

Using Protein Data Bank (PDB) files, I will start all atoms at their experimentally determined position in the molecule's global minimum energy conformer.

$$x_i^0 = \text{PDB coordinate vector}$$

2.4.2 Velocity

I will determine the initial velocity of each atom in the molecule using a Gaussian Distribution such that the expected kinetic energy of the entire molecule reflects

the current temperature of the simulation [3].

$$v_i^0 \overset{i.i.d.}{\sim} N(0, \sqrt{\frac{3k_BT}{m_i}}) \qquad \begin{array}{l} k_B = \text{Boltzmann constant} \\ T = \text{temperature in Kelvin} \\ m_i = \text{mass of atom } i \end{array}$$

2.4.3 Charge, Mass, and Atomic Radius

In addition to the initial position and velocity of each atom in the molecule, I will also need the charge, mass and atomic radius. The first two parameters (charge and mass) will be used when evaluating the energy functions. The latter parameter (atomic radius) will be used when visualizing the molecule. To obtain these parameters I will use the following Wikipedia data sources: *charge*, *mass*, *atomic radius*.

2.5 Velocity Verlet Algorithm

The Velocity Verlet algorithm is an algorithm for efficient numerical integration. This algorithm is commonly used when describing the motion of particles in Newtonian physics given initial parameters. Because atomic movements are very rapid, the algorithm must be run with a very small time step, on the order of a femtosecond ($1.0 * 10^{-15}$ second) [3].

$$\begin{aligned} x_i^{k+1} &= x_i^k + hv_i^k + \frac{h^2 f_i^k}{2m_i} \\ v_i^{k+1} &= v_i^k + \frac{h(f_i^k + f_i^{k+1})}{2m_i} \end{aligned} \qquad \text{where} \qquad \begin{array}{l} h = \text{time-step} \\ m_i = \text{mass of atom } i \\ x_i^k = x_i(t_k) \\ v_i^k = v_i(t_k) \\ f_i^k = \dfrac{\partial E}{\partial x_i^k} \\ t_k = t_0 + kh \end{array}$$

The Velocity Verlet Algorithm has a third-order accuracy for the calculations of position and velocity of particles and exhibits relatively good stability over long time frames [3].

3 Simulation

3.1 Implementation in Matlab

For the initial implementation of the Verlet Algorithm and energy functions I wrote the code in Matlab. This allowed me to easily investigate the functionality

of my implementation by plotting the potential, kinetic and total energy and average bond length of the molecule as a function of time. Upon initial inspection I realized that the constants I was using for the van der Waals potential energy (E_{vdw}) were incorrect and caused the total potential energy to be on a completely different scale than the kinetic energy. Unable to find more accurate parameters I decided to omit van der Waals energy (E_{vdw}) from my simulation. The following simulation results rely only on E_{bond} and $E_{electro}$ and use a time step of $1.0 * 10^{-}16$ seconds.

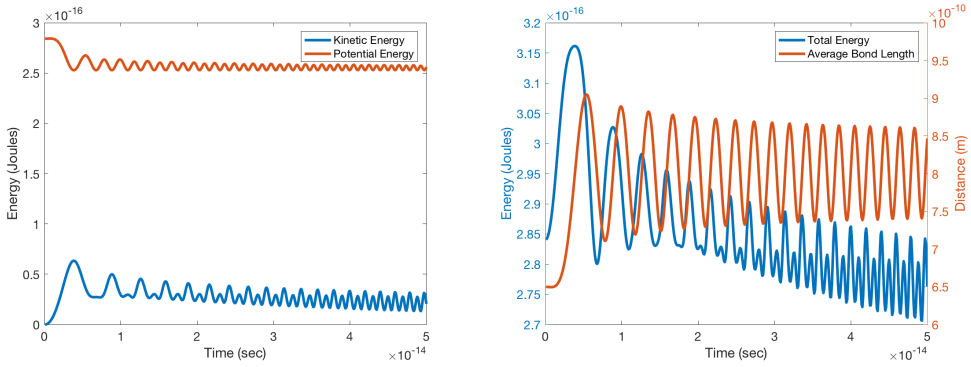

Figure 1: Plot of potential and kinetic energy (left) and total energy and average bond length (right) for a $5 * 10^{-}14$ second simulation of a hydrogen molecule (H_2) at 100 Kelvin.

As seen above, the kinetic energy, potential energy and average bond length are all displaying harmonic motion as would be expected. Unexpectedly, the total energy is also displaying harmonic motion. Ideally, we would see an inverse relationship between potential and kinetic energy such that the total energy is constant. However, in the simulation above the total energy of the molecule is certainly oscillating and decreasing. This most likely due to the fact that the Velocity Verlet algorithm does not perfectly determine position and velocity, the volume and energy of the molecule will change throughout the simulation [3]. Eventually, the molecule will become unstable, causing it to either collapse in on itself or split. By putting a minimum and maximum threshold on the average bond length, then the simulation can run for as long as the molecule stays stable. Below is the result of the same hydrogen gas simulation using this new approach.

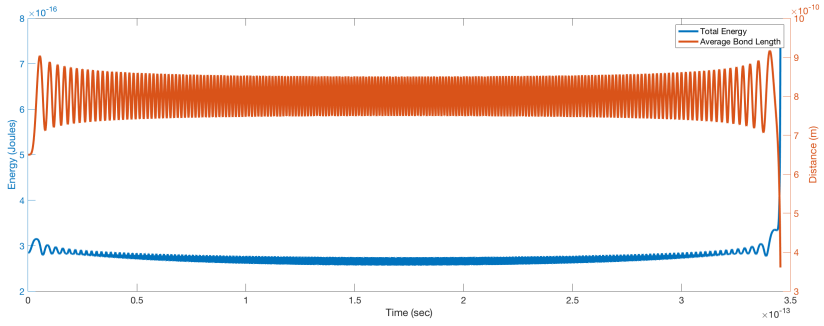

Figure 2: Plot of total energy and average bond length as a function of time for a hydrogen molecule simulation at 100 Kelvin bound by a scalar threshold of 0.5 and 2 of the original bond length.

Using this new approach the above hydrogen simulation ran for nearly $3,500$ iterations before loosing stability. Because each simulation is stochastic the amount of time it will stay stable is also random. The simulation approach using a bond length threshold instead of a time interval is the only way to let simulations run for as long as possible while still providing meaningful results.

3.2 Transitioning to JavaScript

After confirming that the simulation was functioning in Matlab, I transitioned to visualizing the position vectors of each atom using JavaScript, an open-source, front-end language for web development.

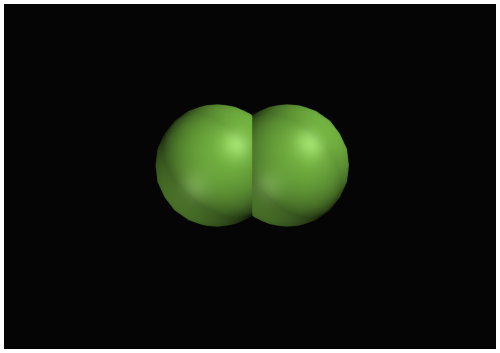

Figure 3: Image of Fluorine molecule from current site.

I adapted an open source template for 3D rendered molecules and uploaded the diatomic molecules I wanted to investigate. I then re-implemented the simulation in JavaScript and added a method for updating the position of each atom. To do this I used three JavaScript libraries extensively:

- *three.js* - A JS library used to render 3D graphics in WebGL.
- *math.js* - A JS library that supplements JavaScript with many mathematical functions and data types.
- *jStat.js* - A JS library that implements advanced statistical operations.

The project is currently hosted on an Amazon server and can be found at the following link: *http://molecular-dynamics-simulation.s3-website-us-east-1.amazonaws.com*

3.3 Frequency and Amplitude

After successfully visualizing the molecular dynamics simulation in JavaScript I started to notice patterns in the motion of the molecules. For example, a simulation of an iodine molecule (I_2) seemed to stay stable for longer than a simulation of a hydrogen molecule (H_2). In order to investigate how different molecules responded to the simulation I wrote an additional function that computed the average frequency and amplitude of the bond length throughout the simulation. There are obvious clusters in this data that corresponds with the molecule being simulated.

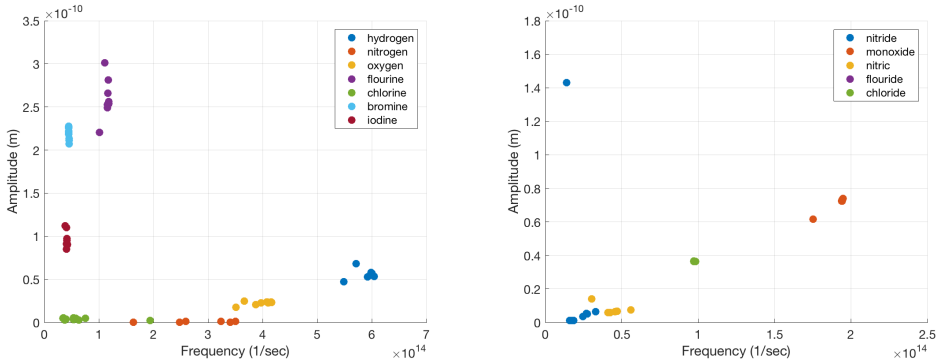

Figure 4: Ten replicates of average amplitude and frequency of bond length for homonuclear (left) and heteronuclear (right) molecules at 100 Kelvin simulation.

In addition to the clustering, there are many trends in the data that appear to be correlated with the molecular makeup of the molecule being simulated. One such trend is that as the mass of the simulated molecule increases ($H_2 \rightarrow I_2$), the average frequency of the bond length decreases, except for nitrogen and oxygen where the trend is switched. Additionally, all of the halogen molecules (F_2, Cl_2, Br_2, I_2) have high amplitudes with the exception of Chlorine (Cl_2). Further, it appears that some clusters have mostly vertical variance in amplitude, while others have mostly horizontal variance in frequency. The patterns and clustering of this data

shows promising results that the simulation is taking into account the specific properties of each molecule.

3.4 Effect of Temperature

I further investigated how the temperature of the simulation affected the dynamics of the molecule. Upon initial inspection there appeared to be no obvious changes in the simulation of a molecule at different temperatures. To investigate this I plotted the average frequency and amplitude of the bond length of a hydrogen molecule as a function of temperature.

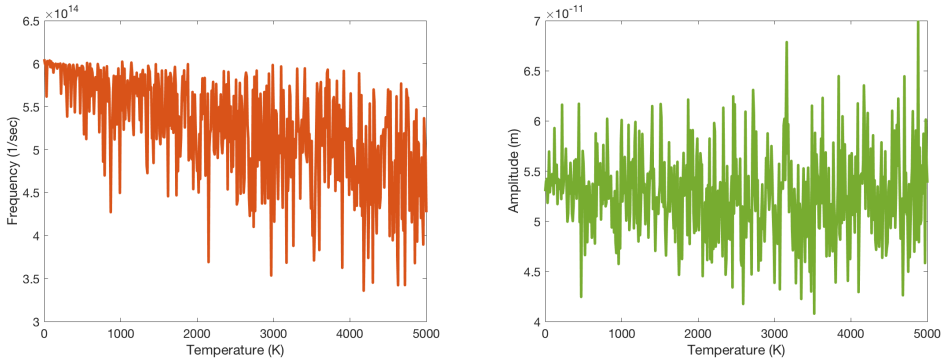

Figure 5: Plots of average frequency (left) and amplitude (right) of a hydrogen molecule bond as a function of simulation temperature.

Each data point in the time series above is a the average of three identical simulations. I did this to try and reduce variance in the data, but even so it is hard to discern any noticeable changes in the properties of the system. From visual inspection, it appears that the frequency of the bond length decreases as the temperature of the simulation increases, while the amplitude of the bond length stays generally the same. I then ran the same experiment but plotted the average time of the simulation as a function of temperature as shown below.

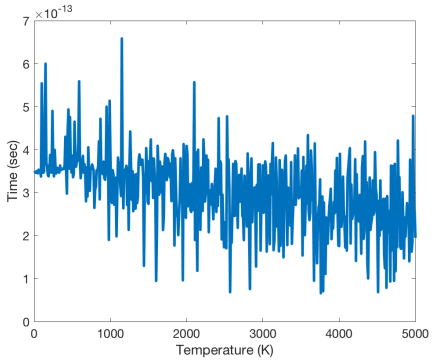

Figure 6: Plot of time of simulation for a hydrogen molecule as a function of temperature.

Similar to the frequency, there appears to be an inverse relationship between the time of a simulation and the temperature of a simulation. This relationship is expected as the molecule will have more energy leading to a larger deviation between the real dynamics of the molecule and the estimates from the Velocity Verlet Algorithm. Which, will in turn lead to the simulation becoming unstable sooner.

4 Improvements

There are a lot of improvements to be made to both the implementation of the energy functions and the visualization of the position vectors. The first major improvement would be to model the remaining energy functions ($E_{angle}, E_{torsion}$) to account for more complicated molecular structures. However, as seen with the van der Waals potential energy (E_{vdw}), implementing the energy function is only half the challenge. Finding accurate constants is necessary for the accuracy of the simulation. The next major improvement would be to find a way of speeding up the JavaScript simulation. Currently, the JavaScript simulation must use a step size nearly 100 times larger than the one used in the Matlab simulation so that the visualization even appears to be dynamic. This larger step size leads to a less accurate and shorter simulation. Finding a way of decreasing the step size while keeping the current speed of the simulation would be a major improvement. Finally, I think a lot more could be explored of the relationship between the molecule being simulated and the descriptive statistics of the simulation. I would be interested in how this relationship could be explained or confirmed by theoretical and experimental findings.

5 Conclusion

Using a Newtonian mechanical model for potential energy, experimentally determined energy constants, and a force-based layout scheme for graph drawing, I successfully implemented a browser-based molecular dynamics simulation of diatomic molecules in three dimensions. I further investigated descriptive statistics of my simulation and showed that the simulation is taking into account the specific properties of each molecule. Finally, I suggested improvements to both the simulation and the visualization moving forward. As web-technologies continue to improve, complex physics simulations can be adapted to be completely web-based in order to increase accessibility and exposure.

References

[1] Badger, Richard M. "The Relation Between Internuclear Distances and the Force Constants of Diatomic Molecules." Physical Review 48.3 (1935): 284-85. Web.

[2] Jasper, Ahren W., and James A. Miller. "Lennard-Jones Parameters for Combustion and Chemical Kinetics Modeling from Full-dimensional Intermolecular Potentials." Combustion and Flame 161.1 (2014): 101-10. Web.

[3] Wu Zhijun. Lecture Notes on Computational Structural Biology. Singapore: n.p., 2008. Print.