Université Paris Cité

Energy Minimization

Dr. Fuchs

Sebastien BEUTLER

Marc EL HASBANY

M1-ISDD

Contents

Introduction

To perform a Molecular dynamics simulation, a set of rules that describe the behavior of a system with mathematical approximation would be needed. It is commonly known as a force field. A force field includes many types of interaction between particles, namely bonded and non-bonded interactions. The most complex system comprises Bond Stretching, Angle Bending, Dihedral Angles, Van der Waals, and Electrostatic interactions. One crucial step in performing a simulation is making sure that the molecule of interest is in a stable conformation. One method to do so is called energy minimization, a mathematical algorithm that detects the nearest local minimum energy well, potentially closest to the global minimum. There are several methods of energy minimization with varying orders and complexities. A popular method among these is the Steepest descent method. From a starting point which is the initial conformation of the system, and through gradient calculation, it finds the direction in which to move, by a specific step size, closer to the minimum to reach a stable conformation after a number of iterations. It is known for its straightforward implementation and efficiency and thus, was chosen to be implemented in this study to find a stable conformation of a system of 2 water molecules.

Material & Methods

The energy potential function that needs to be minimized can take several forms depending on the model chosen. Semi-empirical force fields are computational models for molecular dynamics based on empirical parameters obtained from experimental data which are combined with theoretical equations that approximate the interactions between atoms. These models are useful to perform molecular dynamics simulations on large scale systems that would not be possible using other computational methods like quantum mechanics which are too computationally expensive to emulate.

In this project we chose to base our approach to minimization on the CHARMM27 force field. It's a widely used model for simulating biomolecules with bonded and non-bonded interactions. This model operates under the Born-Oppenheimer approximation which omits the motion of electrons and considers the atoms by their nuclei and the electronic energy as a function of their positions. In this model, an atom is a data structure composed of 4 properties: the name of the atom (used to obtain specific energy terms), the molecule identifier (to restrain some interactions), the charge and the position of the atom in space. The *System* data structure stores the list of atoms and keeps track of the different covalent bonds and angles information that link the atoms together. For increased performance, the distances between atoms can be calculated once per step and stored for later use to prevent the usage of unnecessary resources.

The energy potential function, derived from the CHARMM27 force field, is composed of 6 energy terms: bounded, angles, dihedrals, improper, bending and unbounded forces. As our system is simply two molecules of water, we can consider only the bounded, angles and unbounded (Van Der Waals and electrostatic) interactions and omit the others as they do not apply on water molecules.

Check Annex for Pseudo-Code.

$$E_p = \sum_{\text{bonds}} k_b \cdot (l - l_0)^2 + \sum_{\text{angles}} k_\theta \cdot (\theta - \theta_0)^2 + \sum_{\text{pairs}} \epsilon \cdot \left(\left(\frac{r_{\min}}{r_{ij}} \right)^{12} - \left(\frac{r_{\min}}{r_{ij}} \right)^{6} \right) + k_e \cdot \frac{q_i \cdot q_j}{r_{ij}}$$

The different energy terms in this equation were obtained from the '*charmm27.ff*' force field parameter files. The one used in this system are listed in the following table:

Bounded		Angles		Lennard-Jones		Coulomb	
k_b O-H	450.0 kcal.mol^{-1}.Å^{-2}	k_θ HOH	0.016764 kcal.mol^{-1}.deg^{-2}	r_{min} H-H	0.449 Å	k_b	332.0716 kcal.Å.mol^{-1}.e^{-2}
				r_{min} O-H	1.9927 Å		
				r_{min} O-O	3.5364 Å		
l_0 O-H	0.9572 Å	θ_0 HOH	104.52°	ϵ H-H	0.046 kcal.mol^{-1}	q_O	-0.834 e
				ϵ O-H	0.0836 kcal.mol^{-1}		
				ϵ O-O	0.1521 kcal.mol^{-1}	q_H	0.417 e

The steepest descent algorithm is an optimization method to minimize the energy of the system based on the gradient of the multi-dimensional function of potential energy E_p. The variables of the functions are the coordinates of the atoms, and the gradient is calculated using a numeric method that computes the derivation for each variable while the others are considered constant, and we obtain a matrix of values which represent the direction of the gradient. By moving the coordinates along the opposite of this direction, we gradually move towards a minimum of potential energy. Each coordinate is translated along the gradient and the λ coefficient indicates the size of the step. A high λ can prevent the system from finding a stable state and a low λ increases the number of steps taken before the system converges.

The minimization steps are repeated until a convergence criterion is verified. There are multiple ways to implement it, and most of the time it is when a property of the system reaches a custom threshold, which can be the difference of energies or the RMSD between the last two steps, the maximum component of the gradient or the GRMS (Gradient Root Mean Square). In this model, the GRMS was chosen as the convergence criterion as it is the most commonly used. It measures the magnitude of the gradient and converges towards 0 as the potential energy is minimized.

The pseudo-code present here was implemented in Python 3.10 using the NumPy library for matrix, arrays and vectors calculation and the MatplotLib package for the plotting.

Results & Discussion

A 2-step minimization was first done on the system to minimize only bounded terms with a convergence condition of GRMS hitting a value of less than 0.01, a δ of 0.01, and a λ of 0.001. The results can be seen in the graphs below. First, By looking at the total Energy plot, it is noticeable that it abruptly dropped from an initial state of 51.314 Kcal/mol to around 0.5 Kcal and oscillated around 0 before gradually reaching a value of -0.5 Kcal/mol after 3000 steps and remained constant. The energy difference, RMSD, max gradients, and GRMS between the initial conformation and the different conformations reached during minimization was shown to be drastic at the early steps but quite low afterwards suggesting that the system adopts a stable conformation early on. Furthermore, this sharp drop in fields shows that the system moves from a high energy to a low energy state. Lastly, the interactions plot showed that the bonded term interactions almost immediately stabilize while it takes a significant amount of steps for the non-bonded interactions to converge to a stable state.

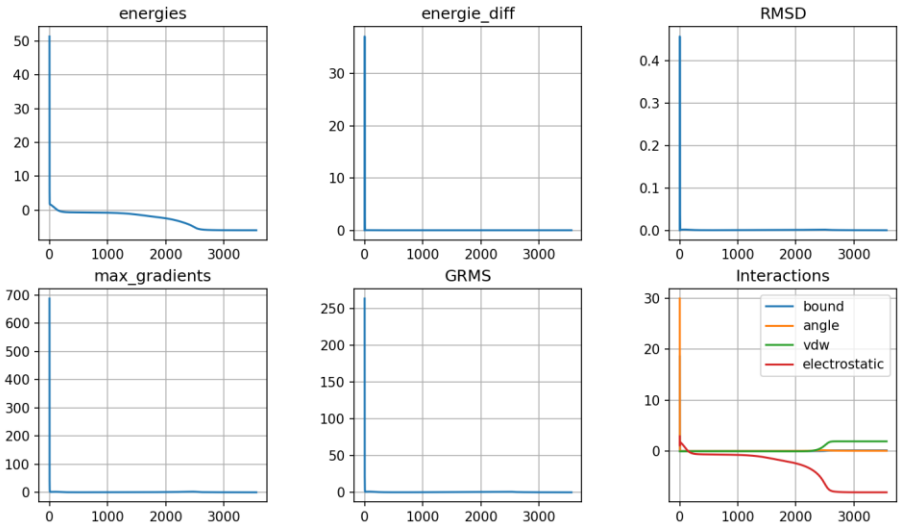

Figure n°1 : Results of the minimization.

Moreover, a 2-step energy minimization was performed on the same system with a bounded-only minimization followed by a full minimization. The resulting plots below show the drop in energy in 2 steps, from an initial state of around 1.825 Kcal/mol to approximately 0.5 Kcal/mol within a few steps and stays constant for about 1500 steps before drastically dropping to a state of -5.939 Kcal/mol after 3000 steps and stays constant. The energy difference plot shows 2 spikes of approximately the same amplitude (0.0150-0.0175 kcal/mol) in the early steps of minimization and one around the 3000th step of around 0.0110 of amplitude. This is consistent with the first energy plot and suggests that the system was unstable and underwent significant changes. Moreover, the same allure was observed in the RMSD, max gradient and GRMS plots showing 2 amplitudes during the early steps and a third around the 3000th step, further suggesting that the system moved from an unstable to a more stable state. Lastly, the interactions plot, which reveals the breakdown of every energy terms' contribution, shows that the bond stretch and angle bending terms stabilize quickly with almost no variations, while the non-bonded interaction terms show some variations before stabilizing around the 3000th step. This indicates that the non-bonded interactions significantly contributed to the energy minimization of the system by first achieving a lower energy initial state from when only the bonded interactions were considered. We can conclude that the minimization goes through 3 states, a first one (step 0-200) where there is a significant drop in energy and electrostatic

potential, probably dues to the fact that the two oxygens in the system are close together initially, a second step (step 200-2500) where the electrostatic forces are the major factor in the gradient and a third steps (2500-3500) where the molecules are closer to each other and find an energy equilibrium between the electrostatic forces and the Van Der Waals interactions.

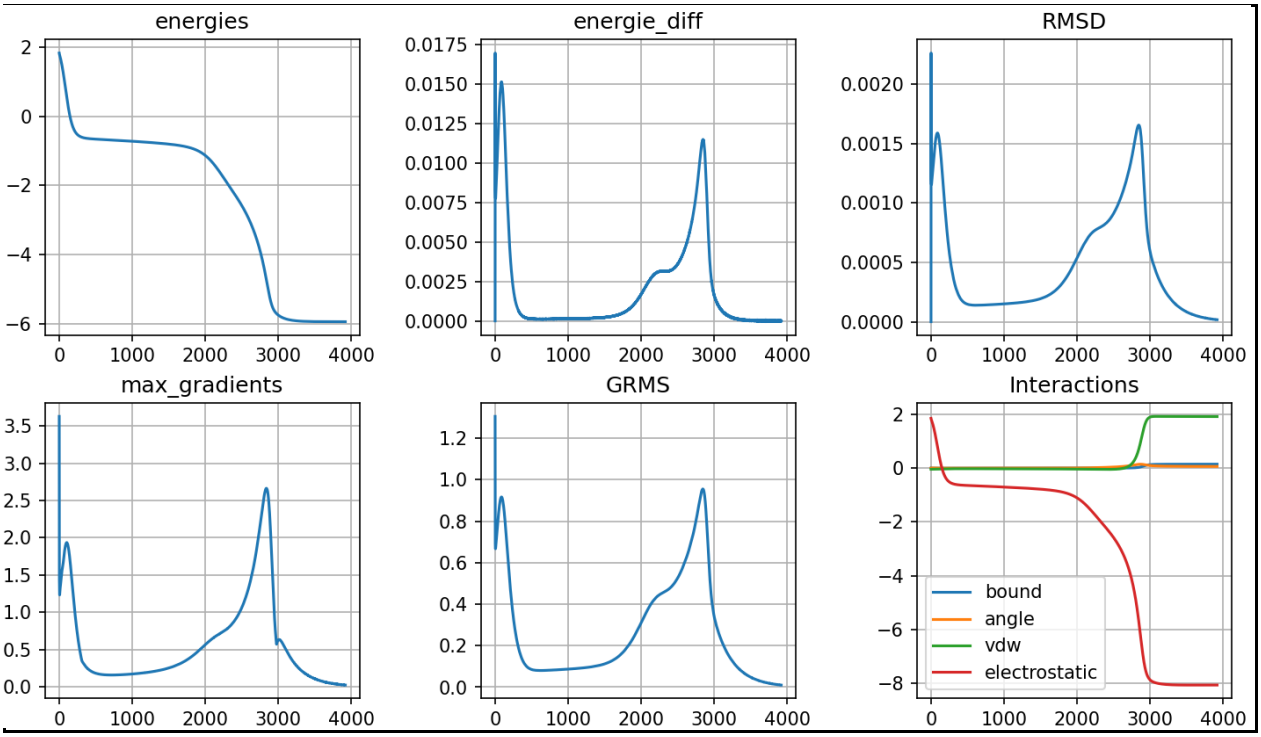

Figure n°2 : 2-step minimization. (Lower initial energy)
Results after a bounded-only minimization then a normal minimization.
Energy: from 51.314 to 1.825 to -5.939 kcal/mol

Subsequently, a film was generated of the changes the system undergoes during energy minimization. First, the water molecules, located parallel to each other, rotate so that one Oxygen would face the other molecule's Hydrogens which would optimize the electrostatic interactions between the electronegative Oxygen and the Hydrogens. Then, the two molecules move towards each other, enhancing the Van der Waals interactions as seen in the slight increase in the interactions plot until one displays a 90 degrees rotation to maximize the alignment of a Hydrogen bond, allowing the system to reach its stable low energy state. This explains the 3 peak allures that we see in the plots above, as the minimization occurs in 3 steps, first the rotation of the water molecules, second their movement towards one another, and third, the rotation of one water molecule to compensate for the close proximity and optimize Hydrogen bonding.

As a final step, an energy minimization using the steepest descent method of the system was performed using MOE, in an effort to assess the quality of the developed algorithm. Minimization with MOE reached a final state of -6.778 kcal/mol close to the final state reached with the algorithm developed in this study.

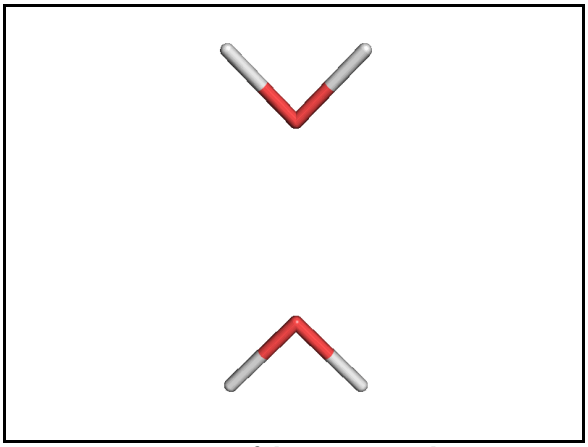

Figure n°3 : Animation of the system. (File: animation.gif)

Conclusion

In conclusion, the successful application of the steepest descent method for energy minimization in a system of two water molecules using the CHARMM27 force field was demonstrated. The method reduced the energy from an initial 51.314 kcal/mol to -5.939 kcal/mol, showcasing the potential of the algorithm in predicting stable states. Furthermore, visualization of the energy minimization confirmed the critical role of both bonded and non-bonded interactions in reaching equilibrium. Lastly, validation of the algorithm using MOE software's results, which were in close agreement with the final energy state obtained in the study, shows that the algorithm performs with great accuracy and quality. Furthermore, this model could be improved in a lot of different ways. It is possible to increase the performances of the calculations to include more atoms in the system by using matrices to store the positions of the atoms and other terms that would speed up the energy calculation by relying on the fast matrix operations and calculating the distances between atoms in parallel. The amount of steps before converging could also be improved by replacing the constants λ and δ by the Line Search method or the Brent's method. Finally, adding more energy terms constants from the CHARMM27 force field parameter file would allow to perform a minimization with a more various types of atoms and interactions.

Bibliography

1. Meza, J. C. (2010). Steepest descent. *WIREs Computational Statistics*, *2*(6), 719–722. https://doi.org/10.1002/wics.117
2. Vilar, S., Cozza, G., & Moro, S. (2008). Medicinal Chemistry and the Molecular Operating Environment (MOE): Application of QSAR and molecular docking to drug discovery. *Current Topics in Medicinal Chemistry*, *8*(18), 1555–1572. https://doi.org/10.2174/156802608786786624
3. Vanommeslaeghe, K., Hatcher, E., Acharya, C., Kundu, S. C., Zhong, S., Shim, J., Darian, E., Guvench, O., Lopes, P. E. M., Vorobyov, I., & MacKerell, A. D. (2009). CHARMM general force field: A force field for drug-like molecules compatible with the CHARMM all-atom additive biological force fields. *Journal of Computational Chemistry*, *31*(4), 671–690. https://doi.org/10.1002/jcc.21367
4. Brooks, B. R., Brooks, C. L., MacKerell, A. D., Nilsson, L., Petrella, R. J., Roux, B., Won, Y., Archontis, G., Bartels, C., Boresch, S., Caflisch, A., Caves, L. S. D., Cui, Q., Dinner, A. R., Feig, M., Fischer, S., Gao, J., Hodošček, M., Im, W., . . . Karplus, M. (2009). CHARMM: The biomolecular simulation program. *Journal of Computational Chemistry*, *30*(10), 1545–1614. https://doi.org/10.1002/jcc.21287

Annex

```
Class Atom:
    name: string
    molecule: string
    charge: float
    position: array float[3]
```

```
Class System:
    atoms: array Atom[]
    bounds: array (Atom, Atom)[]
    angles: array (Atom, Atom, Atom)[]
    distances: dictionary { (Atom, Atom): float } // Optional
```

```
Function energy_bound(bound: (Atom, Atom), k: float, l0: float) → float:
    atom1, atom2: Atom ← bound    // Deconstruct the tuple
    Return k * (distance(atom1.position, atom2.position) - l0)²
```

```
Function energy_angle(atoms: (Atom, Atom, Atom), k: float, θ0: float) → float:
    atom1, atom2, atom3: Atom ← atoms    // Deconstruct the tuple
    Return k * (angle(atom1.position, atom2.position, atom3.position) - θ0)²

Function angle(p1: array float[3], p2: array float[3], p3: array float[3]) → float:
    vec1: array float[3] ← p1 - p2
    vec2: array float[3] ← p3 - p2
    cos_angle: float ← dot(vec1, vec2) / ( norm(vec1) * norm(vec2) )
    Return arccos(cos_angle)
```

```
Function energy_lennard_jones(atom1: Atom, atom2: Atom, ε: float, rmin: float) → float:
    r: float ← distance(atom1, atom2)
    rfinal ← rmin / r
    Return ε * ( rfinal^12 - rfinal^6 )
```

```
Function energy_coulomb(atom1: Atom, atom2: Atom) → float:
    Ke: ← float ← 332.0716
    Return Ke * (atom1.charge * atom2.charge) / distance(atom1, atom2)
```

```
Function gradient(atom: Atom, system: System) → array float[3]:
    gradients: array float[3]
    energy_plus, energy_minus: float

    δ: float ← 0.01
    i: int ← 0

    ForEach coord: float in atom.position:
        atom.position[i] ← coord + δ
        energy_plus: ← energy_total(system)
        atom.position[i] ← coord - δ
        energy_minus: ← energy_total(system)
        atom.position[i] ← coord  // Reset to original position
        gradients[i] ← -(energy_plus - energy_minus) / (2 * δ)

    Return gradients
```

```
Function minimize(system: System, max_steps: int, threshold: float):
    GRMS: float, gradients: array[ array float[3] ]
    λ: float ← 0.001
    step: int ← 0

    While step <= max_steps:
        step ← step + 1
        gradients ← []

        // Perform a minimization step
        ForEach atom: Atom in system.atoms:
            gradients.append(gradient(atom, system)) // Store gradient
            atom.position ← atom.position + λ * gradients[-1]

        // Check the stop criterion
        GRMS ← sqrt( sum(gradients²) / length(gradients) )
        If GRMS < threshold:
            Break
```

```
Function energy_total(system: System) → float:
    // Store all pairs of atoms which are not part of the same molecule
    atom_pairs: array (Atom, Atom)[] ← pairs(system.atoms, bonded=FALSE)

    energies_bounds: float ← sum(map(energy_bound, system.bounds))
    energies_angles: float ← sum(map(energy_angle, system.angles))
    energies_vdw: float ← sum(map(energy_lennard_jones, atom_pairs))
    energies_elec: float ← sum(map(energy_coulomb, atom_pairs))

    Return energies_bounds + energies_angles + energies_vdw + energies_elec
```

Repartition:

Introduction: Marc

Method: Sebastien

Results & Discussion: Marc et Sebastien

Conclusion: Marc et Sebastien

Code:

Bonded Interactions: Sebastien

Non-Bonded Interactions: Marc