MODES
Junior Professorship for
Multi-scale Modeling of
Heterogeneous Catalysis
in Energy Systems

RWTH AACHEN
UNIVERSITY

Thermodynamics,
Modeling & Simulations

Winter Semester
2022/2023

Energy Minimization of Nanoparticle

Changming Chen,
Darren Lam Ming Hui,
Shizhe Xing,
Thanh Mai Vu

supervised by
Abhishek Khetan

Acknowledgements

Before we begin, we would like to acknowledge the work that all our group members have put in. The countless hours and energy spent on this project is paid off tremendously when we finally got the results. We also would like to express our gratitude to other groups for lending a helping hand when we were facing difficulties. Also special thanks to Abhishek Khetan, our supervisor for this project, for his support and inspiration.

Abstract

In this project, we use numerical methods, including line search and conjugate gradient, to study the Minima of Potential Energy using the Lennard-Jones Potential Function and run stochastic simulations with different initial configurations to obtain the putative global potential energy. For reference we take the paper Global Optima of Lennard-Jones Cluster [1] to compare the results. Several methods and algorithms were explored and implemented to achieve high accuracy. At the end we are able to observe the mechanically stable structure of a nanoparticle by visualizing the cluster at its global optima accurately.

Keywords

Keywords: **Numerical Approach for Potential Energy Minimization with Lennard-Jones Clusters**

Contents

1 Introduction

The goal of the project is to find the minimum potential energy so that a nanoparticle conforms to its stable structure. For each simulation, we firstly place the atoms in a three-dimensional cube of size L, which is proportional to the number of atoms N.

For our numerical method, we implement two separate functions, the first one for the total potential energy based on Lennard-Jones equation, which will then be minimized and the second one is its derivative, the force function, what is needed to perform minimization.

The next step is to implement the line search function, which will find the first minimum along a search direction. Once we reach the minimum area, we apply the Golden Section Method to locate the minimum. The reason is to take advantages of smart algorithm, reduce the effort of function calculation and thus computational cost.

Finally, we apply conjugate gradient descent which outperforms steepest descent by taking previous steps, eliminating the oscillatory behaviour and gives us an efficient algorithm. It will calculate different search directions, call line search and return the value, which is a nearby local minimum with a given initial value, a nearest stable conformation.

We then run many simulations to have several local minima with different energy values resulting from different initial configurations. The one that has the lowest energy is selected, a ‘greedy’ step is taken, we get our putative global minimum.

We have opted for 3 different versions of codes for the solution for this assignment which will be introduced later in the solution section. Each with slight changes to the other.

2 Related Work

As we go to the higher number of atoms N, the number of local minima will grow exponentially. Therefore, we can easily end up in one of the many local minima while conducting the conjugate gradient. To validate that our algorithm could reach the lowest possible minima after some simulations, we constantly compare our results with the table shown below.

N	-U	N	-U
2	1.000000	14	47.845157
3	3.000000	15	52.322627
4	6.000000	16	56.815742
5	9.1003852	17	61.317995
6	12.712062	18	66.530949
7	16.505384	19	72.659782
8	19.821489	20	77.177043
9	24.113360	21	81.684571
10	28.422532	22	86.809782
11	32.765970	23	92.844472
12	37.967600	24	97.348815
13	44.326801	25	102.372663

Table 1: Values given by Global Optima of Lennard-Jones Clusters
[1]

During the simulations some significant deviation from the values stated here are observed which will be shown and discussed here Results and Discussions

3 Solution

3.1 Energy Function

The total potential energy

$$U^* = \sum_i \alpha |r_i|^2 + \sum_{i<j} u(r_{ij}) \quad , \quad \alpha = 0.0001 N^{-2/3} \tag{1}$$

with $u(r_{ij})$ given by

$$u(r_{ij}) = \begin{cases} 4\epsilon[(r_{ij}/\sigma)^{-12} - (r_{ij}/\sigma)^{-6}] - 4\epsilon[(r_c/\sigma)^{-12} - (r_c/\sigma)^{-6}] & r_c \geq r_{ij} \\ 0 & r_c > r_{ij} \end{cases} \quad r_c = 2.5\sigma \tag{2}$$

To speed up the code we have

$$term = 4\epsilon((r_c/\sigma)^{-12} - (r_c/\sigma)^{-6}) \tag{3}$$

$$term1 = (\sigma/r_{ij})^6 \tag{4}$$

so that we end up with the simplified energy function as

$$U^* = 4\epsilon term1(term1 - 1) - term \tag{5}$$

3.2 Force Function

The force function is the derivative of the energy function with

$$u_{ij}^{'} = -4\epsilon[12(\sigma^{12}/r_{ij}) - 6(\sigma^6/r_{ij}^7)] \tag{6}$$

For simplification we write

$$term = (\sigma/r_{ij})^6 \tag{7}$$

and the final implemented force equation is given by

$$f_{ij} = -2\alpha r_i - \sum_{i \neq j} 24\epsilon * \hat{r_{ij}}/r_{ij}^2 * (term(2term - 1)) \quad , \hat{r_{ij}} = r_j - r_i \tag{8}$$

3.3 Line Search

[2] **Algorithm:**

Step 1: Start with an initial set of coordinates $r_0^N$and a search direction d

Step 2: Assign $r_1^N = r_0^N + \delta d$ and $r_2^N = r_1^N +, \delta d$

Step 3: Check if $U(r_2^N) > U(r_1^N)$ and $U(r_0^N) > U(r_1^N)$

-> If **No**, we have not reached the minimum region, then

Step 4a: Shift values as $r_0^N = r_1^N, r_1^N = r_2^N, r_2^N = r_1^N + \delta d$, and **go back to step 3**

-> If **Yes** then

Step 4b: Locate the minimum r_{min}^N between r_0^N, r_1^N and r_2^N by Golden Section method.

Step 5:

Calculate a new search direction along r_{min}^N and **go back to step 2**.

Repeat the procedure until the change in energy ΔU is less than *LSstol* 12

Step size (learning rate):

1. Choose a reasonably small step size $\delta = 0.01$

2. Normalize the given search section row-wise with Euclidean norm

We do not normalize the whole matrix by dividing all of its elements by one value, because when some atoms build a cluster, they then have a huge force to each other compared to the one towards those which are far away. Atoms in different clusters do not feel forced by other clusters which makes them only move around themselves and different clusters do not come together. They than build many clusters and thus the energy is not minimized, which is not supposed to be.

The **reason** for row-wise norm is to allow all atoms move with the same peace at the same time and not any is left behind. All atoms come than together to the origin of the coordinate. We can then get the next lowest possible energy and continue searching.

3. Our method is conjugate gradient descent. With some mathematical background descent direction means direction that leads to lower function values. That's why U_1 should always be lower than U_0, if it's not the case, we have to reduce δ by half each time and keep trying until we get the optimal δ.

3.3.1 Golden Ratio Method

The symbol φ represents the Golden Ratio with the equation of

$$\varphi = 1/2(1 - \sqrt{5}) \quad \text{or} \quad \varphi = 1/2(1 + \sqrt{5}) \tag{9}$$

Assuming we have two points x_l and x_u (lower and upper boundaries), which we have obtained from the line search. We can calculate the step taken with the golden ratio.

$$h = (x_u - x_l)\varphi \tag{10}$$

We then calculate

$$x_1 = x_u + h \quad \text{and} \quad x_2 = x_l - h \tag{11}$$

At each iteration we shift the points x_l and x_u close to the minimum energy until a certain tolerance is reached, here it is stated as LStol $= 10^{-8}$. With the condition

$$|U_u - U_l| <= LStol|U_0| \tag{12}$$

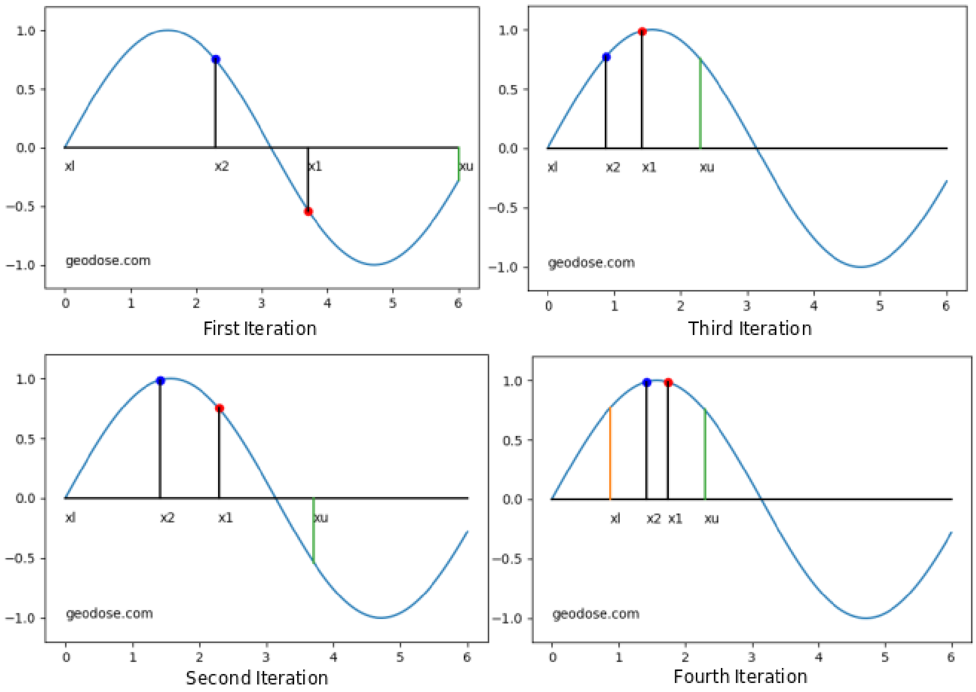

Figure 1: Golden Ratio Method for finding the maximum (an analogue for minimum)

3.4 Conjugate Gradient

We then need a descent direction for our line search, which brings our function fastest and most efficiently to the next local minimum. With steepest descent descent direction, it usually leads to a zig-zag situation, where function value just changes within a very small amount range. To address the mentioned problem, there is conjugate gradient descent which has an ability able to rectify the right angle steps that the line search then takes.

The direction is calculated using the formula

$$d_i^N = f_i^N + \gamma_i d_{i-1}^N \tag{13}$$

which takes into account the previous step.

We calculate the γ using the ratio discovered by Polak–Ribière [3]

$$\gamma_i^N = \frac{(f_i^N - f_{i-1}^N) f_i^N}{f_{i-1}^N f_{i-1}^N} \tag{14}$$

We take only γ, which is greater than 0, into account, because the previous step leading to minimum along a searching line should not be undone or removed.

4 Results and Discussions

Before starting with the results and discussions, we have to mention that there are three varying versions of code. Only the third version is the one that we use as our final solution. And below we will also justify the differences between them.

4.1 Version 1

N	-U	Deviation %	N	-U	Deviation %
2	0,983683	1,6316891	14	46,377772	3,0669456
3	2,951049	1,6316893	15	50,643140	3,2098682
4	5,902099	1,6316895	16	54,007955	4,9419171
5	8,940683	1,7548923	17	58,240797	5,0184256
6	12,058173	5,1438459	18	63,396732	4,7109156
7	16,162725	2,0760438	19	67,162387	7,5659390
8	19,364613	2,3049528	20	70,738328	8,3427849
9	23,525946	2,4360505	21	73,055061	10,5644306
10	27,688260	2,5834160	22	80,029805	7,8101528
11	31,868531	2,7389363	23	85,061427	8,3828852
12	35,235303	7,1963900	24	87,303688	10,3186952
13	43,054067	2,8712524	25	88,698210	13,3575241

Table 2: Acquired values from the 1st version of simulation with the deviation from Table 1 5.4.1

4.1.1 Discussion

In the first version, all the implemented algorithm was written as instructed in the exercises. However, the results wasn't as accurate as it is supposed to be, which can be obtained from the deviation in the table above. The deviation grows exponentially with greater number of atoms. Therefore, another method was tried out to with some optimization processes and the second version was produced.

4.2 Version 2

N	-U	Deviation %	N	-U	Deviation %
2	1,000000	0,0000000	14	47,845143	0,0000291
3	3,000000	0,0000000	15	52,322594	0,0000639
4	6,000000	0,0000002	16	55,905160	1,6026927
5	9,103852	0,0380971	17	61,306964	0,0179892
6	12,712058	0,0000335	18	66,284446	0,3705091
7	16,505383	0,0000099	19	72,658218	0,0021520
8	19,821485	0,0000180	20	77,177000	0,0000563
9	24,113353	0,0000287	21	81,650734	0,0414243
10	28,422517	0,0000536	22	86,179764	0,7257449
11	32,765955	0,0000473	23	90,732832	2,2743841
12	37,967588	0,0000316	24	94,500554	2,9258306
13	44,326771	0,0000677	25	101,026652	1,3148149

Table 3: Acquired values from the 2nd version of simulation with the deviation from Table 1 5.4.1

4.2.1 Discussion

The idea for the second version is that the observed deviation was due to the constraints

$$r_{min} = r_{ij} - Lround(r_{ij}/L) \quad and \quad r_{ij} < r_c \tag{15}$$

The r_{min} term allows atoms in different boxes in the lattice to interact with each other. It then often leads to the case when clusters are in the boundaries between boxes. When the density is small enough, at their global minimal potential energy, atoms form a cluster in the middle of the box, the distance between two atoms is always less then L/2, which results in interactions just within one box and thus the equation is no more needed. As the result, we eliminated it.

When there are many atoms beforehand, the distance between them are large. Atoms in the neighborhood immediately start conforming their cluster which is isolated from others. The distance between clusters is sometimes greater than r_c, where atoms do not feel force from others clusters anymore and stay still far away from each other. This phenomenon causes the simulation to reach a higher local minima. As a solution, the constraint is eliminated.

It does show a better result compared to the previous version, but it still deviates at higher N due to the increase of number of local minima. We couldn't reach the lowest of them, many simulation runs are needed to randomly get the better initial guess, which needs a longer runtime.

4.3 Version 3 (Our Final Solution)

N	-U	Deviation %	N	-U	Deviation %
2	1,000000	0,0000000	14	47,843645	0,0031608
3	3,000000	0,0000003	15	52,321065	0,0029854
4	5,999995	0,0000758	16	56,814379	0,0023983
5	9,103834	0,0378977	17	61,315669	0,0037937
6	12,712035	0,0002089	18	66,281077	0,3755720
7	16,505221	0,0009904	19	72,656204	0,0049245
8	19,821288	0,0010156	20	77,170505	0,0084717
9	24,112981	0,0015735	21	81,674564	0,0122508
10	28,422088	0,0015638	22	86,788240	0,0248156
11	32,765364	0,0018505	23	92,835814	0,0093250
12	37,966652	0,0024980	24	97,327194	0,0222096
13	44,326277	0,0011825	25	102,352337	0,0198548

Table 4: Acquired values from the 3nd version of simulation with the deviation from Table 1 5.4.1

4.3.1 Discussion

Instead of randomly get a new initial guess every time we reach one minimum with conjugate gradient, we decided not to stop but just go one more step further to the next local minimum, which can be reasonably larger. From there, we have a good initial guess, what helps us to have a new better set of search directions. We know the path that we have gone though and now we take wiser steps back to the minimum and then to the lower and lower energy value.

From our observation, steepest descent gives a moderate step size and a good decrease in function value, when conjugate gradient take too large steps and often end up somewhere too far away from the actual putative global minimum, so we use force as search direction. Moreover, we do not have tolerance value as stop condition anymore, we allow our function to jump from one local minimum to another and stop the algorithm after some reasonable number of steps.

With that modification we are now able to have a version that is fast (finished within ten hours compared to two days as previous) with small to zero deviation.

5 Graphs

5.1 Task 2

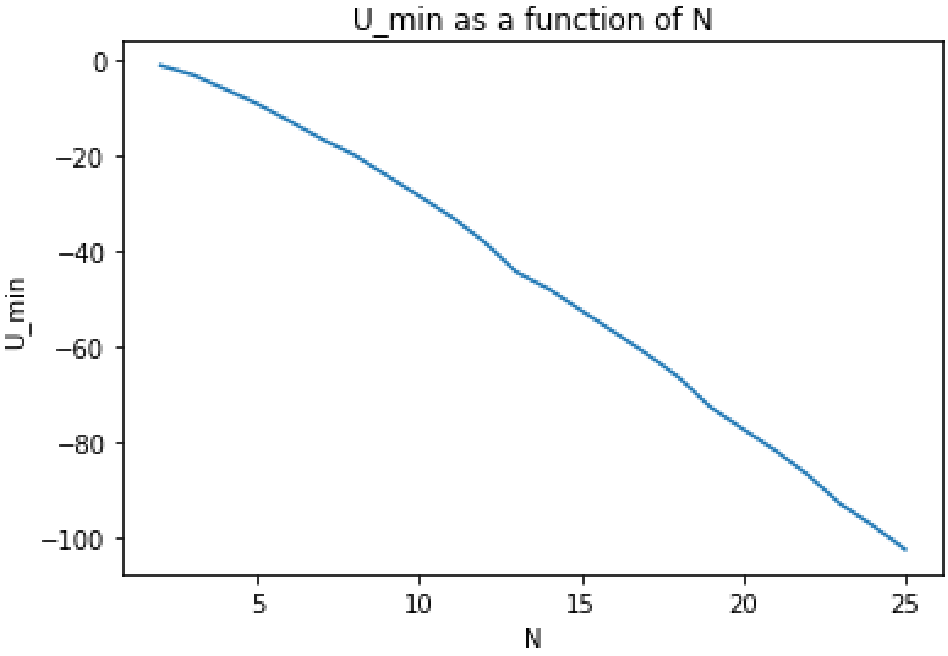

Figure 2: Minimum energy progression at each N

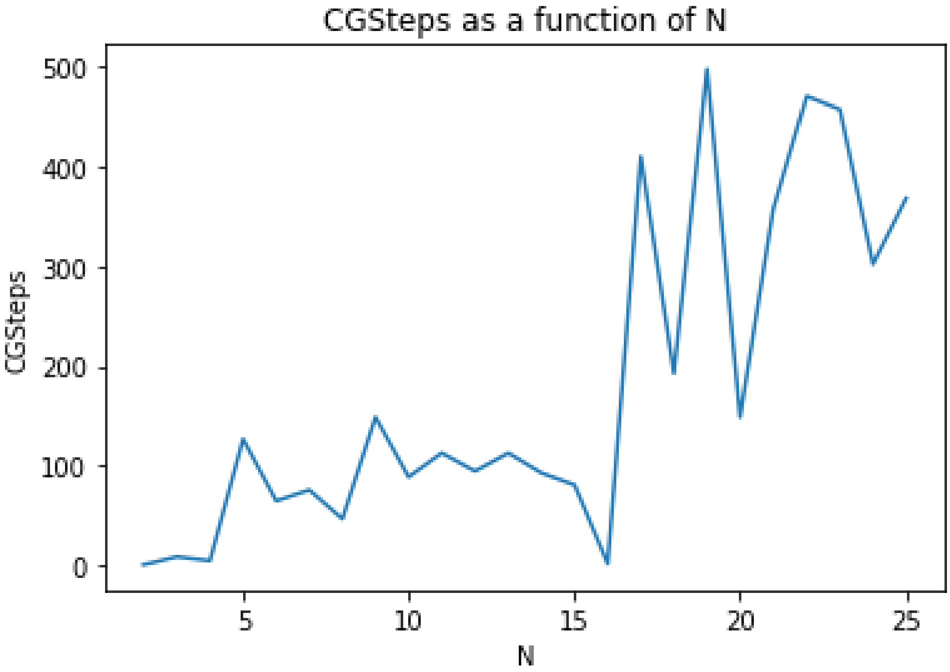

Figure 3: Conjugate gradient step progression at each N

5.2 Task 3

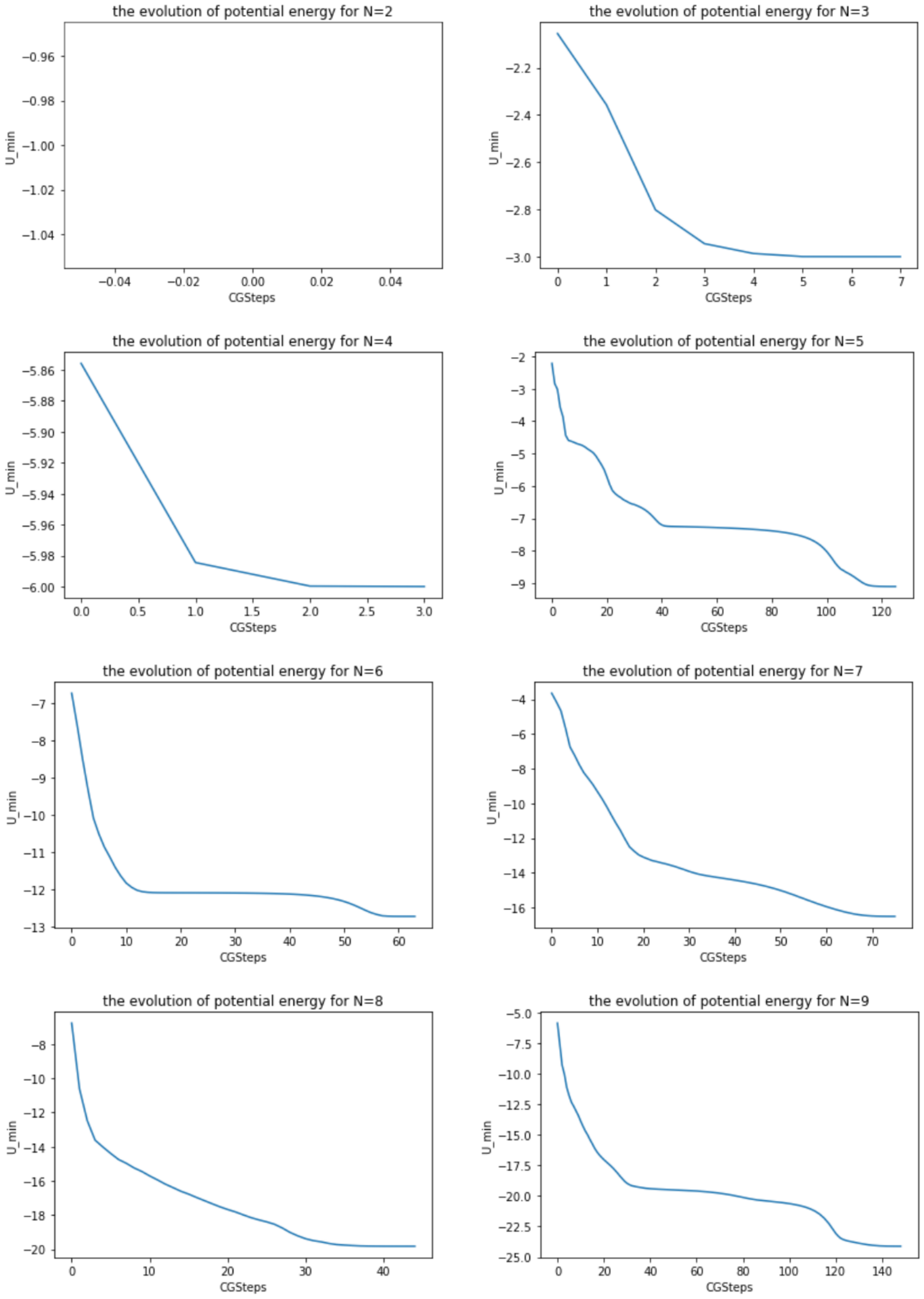

Figure 4: Evolution of the minimum energy at each N (2-9)

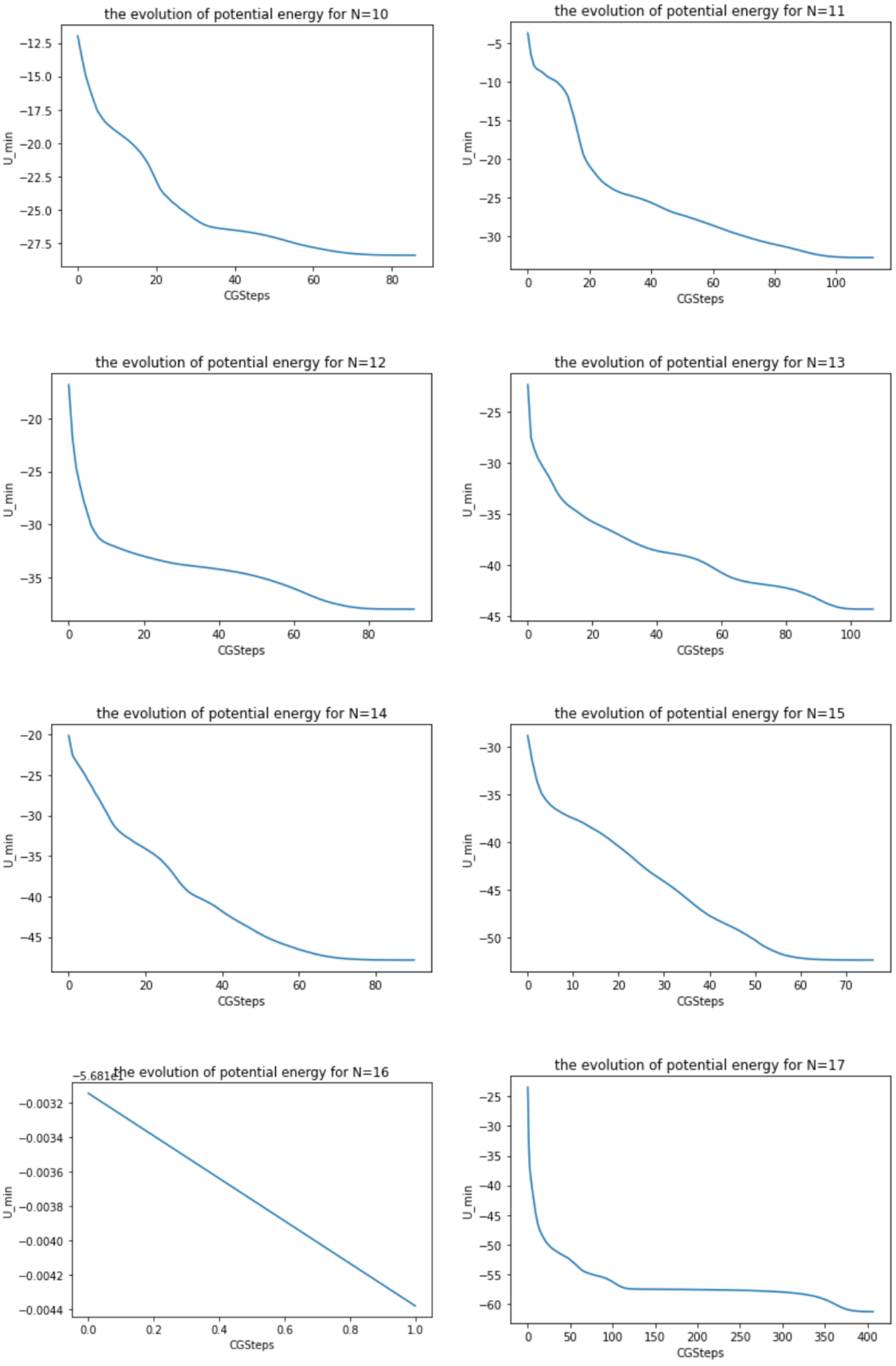

Figure 5: Evolution of the minimum energy at each N (10-17)

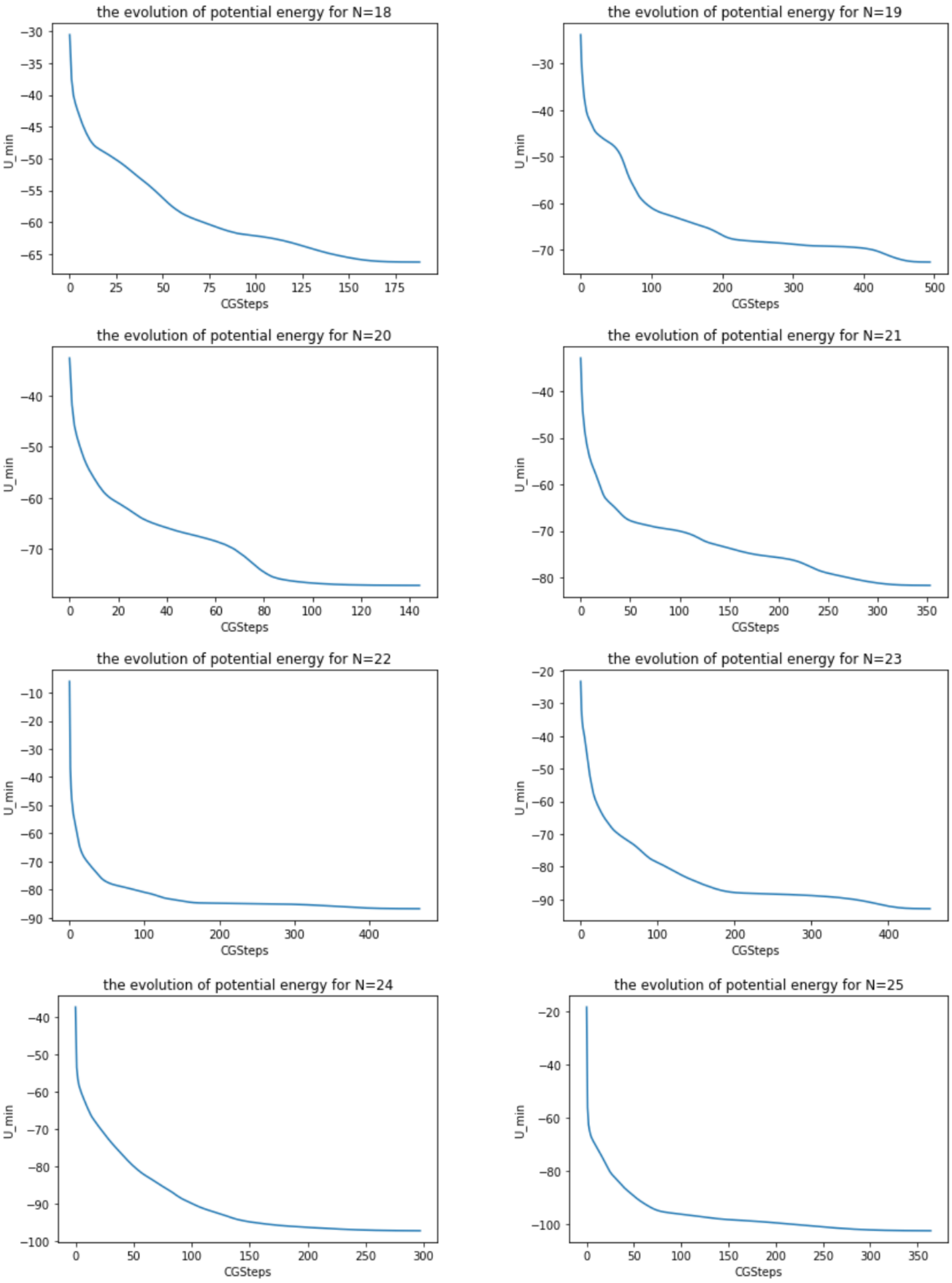

Figure 6: Evolution of the minimum energy at each N (18-25)

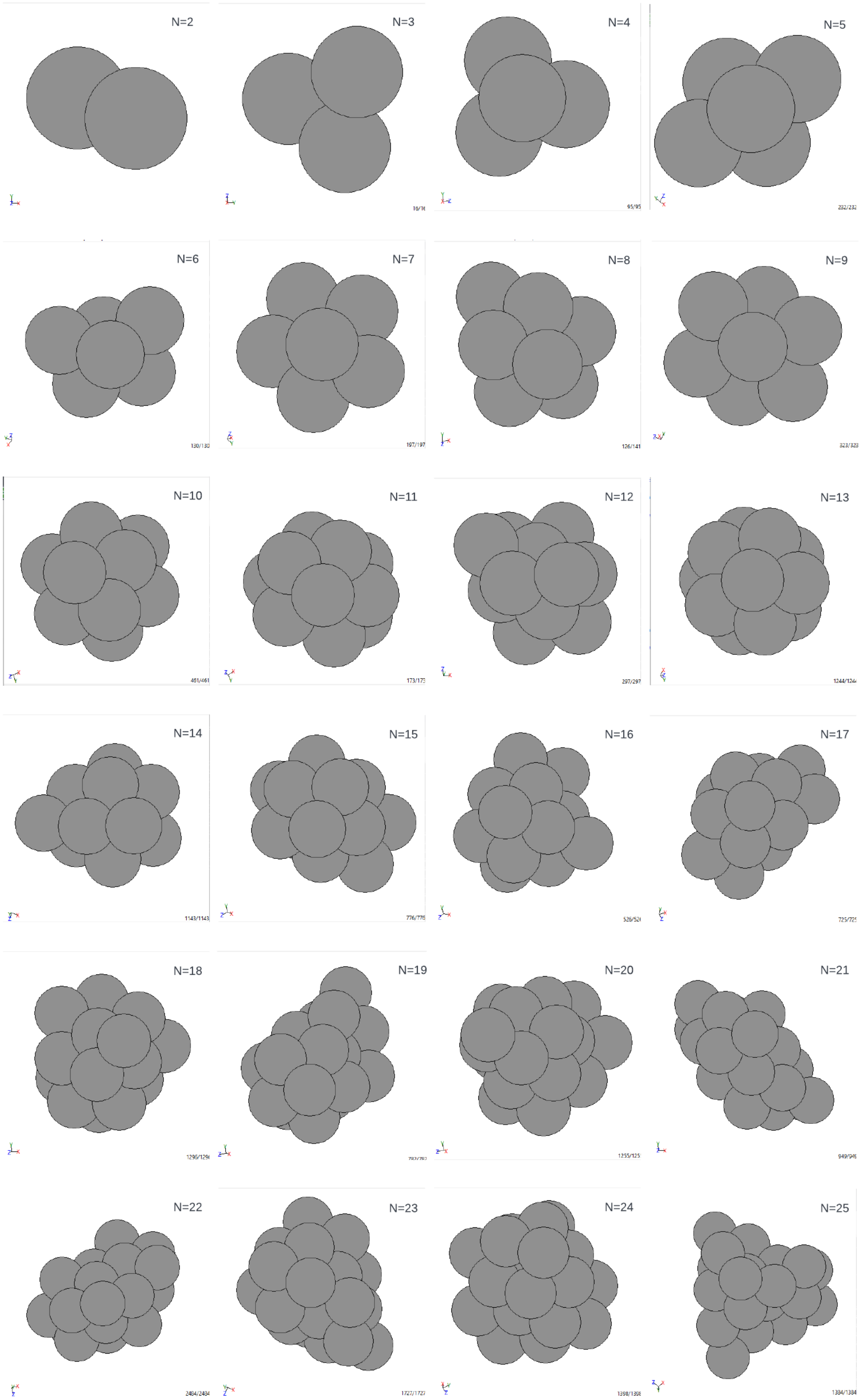

Figure 7: Visualization of each N

5.3 Task 4

The macroscopic scaling estimate for nanoparticle energy

$$U_{macro}(N) = a + bN^{2/3} + cN \tag{16}$$

a	b	c
-3.078349	11.453650	-7.907704

Table 5: The values of a, b and c

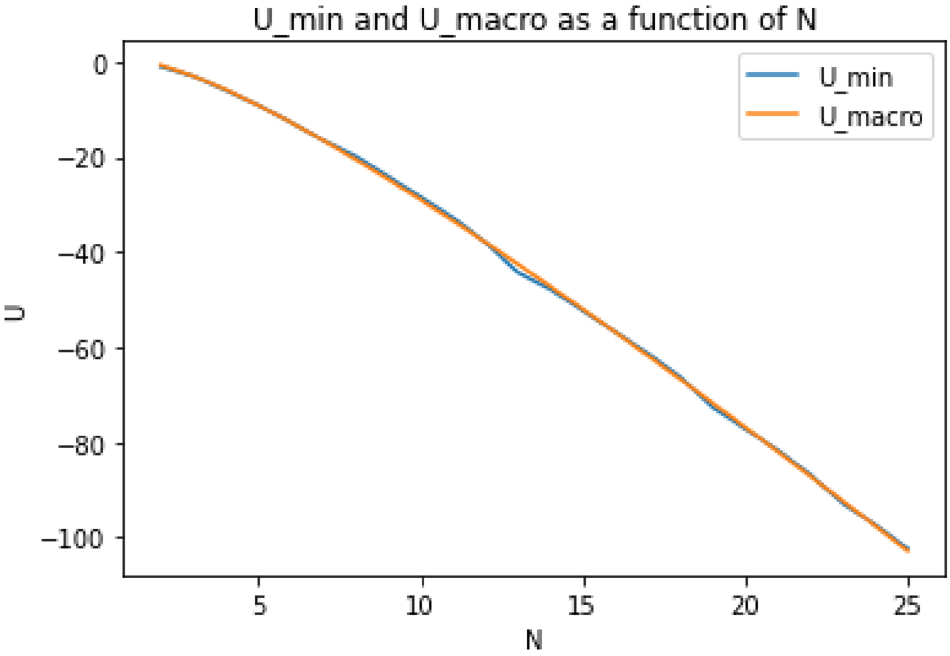

Figure 8: Fitting the minimum potential energy data to the given formula

5.4 Bonus Task

Results for $N/V = 0.01$

5.4.1 Task 2

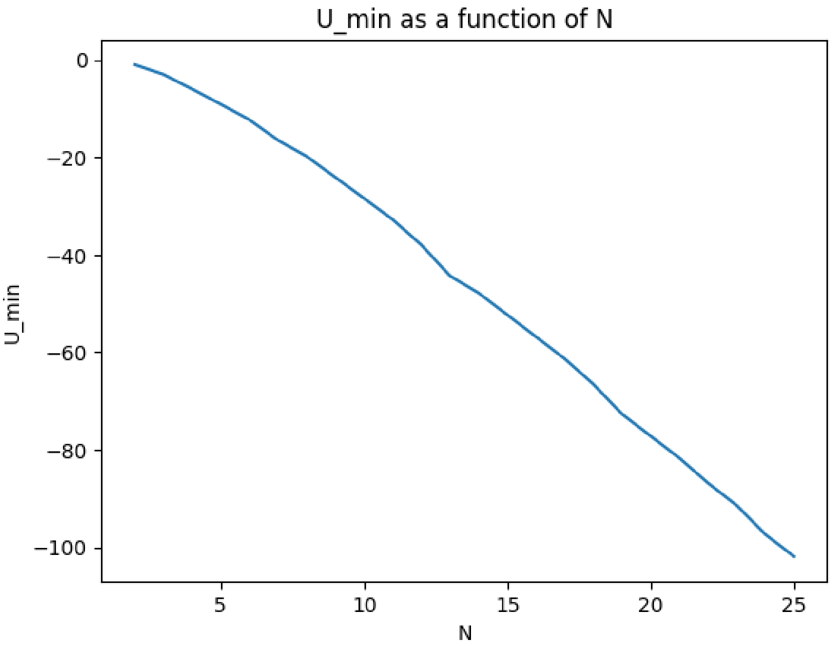

Figure 9: Minimum energy progression at each N

N	-U	N	-U
2	1.000000	14	47.843584
3	3.000000	15	52.320602
4	6.000000	16	56.813694
5	9.103835	17	61.315664
6	12.302552	18	66.526096
7	16.505135	19	72.656046
8	19.821304	20	77.172222
9	24.113126	21	81.677603
10	28.421965	22	86.800246
11	32.765046	23	91.386532
12	37.966161	24	97.335513
13	44.325964	25	101.853382

Table 6: Valeus for the bonus task with $N/V = 0.01$
[1]

6 Discussion

From the graph 2 it is observed that the minimum energy decreases as the number of atoms increases. Due to the increasing number of atoms and their bonds, more energy is to be released for them to come close together, for a perfectly stable conformation of a nanoparticle.

As the number of atoms increases, more conjugate gradient steps are to be performed, because the structure is complicated at first, atoms need to move more to find their most minimum potential energy state. This is observed in graph 3.

The minimum energy decreases with the number of conjugate gradient steps6. However, at N=2 there wasn't a graph line observed. That is because the the minimum state is already achieved after the first conjugate gradient step.

Now let's discuss one interesting conformation in this assignment. In the article [1], there was mention of the 'magic number'. In this assignment the visualization of the conformation at the 'magic number' is done with the python extension "ase"

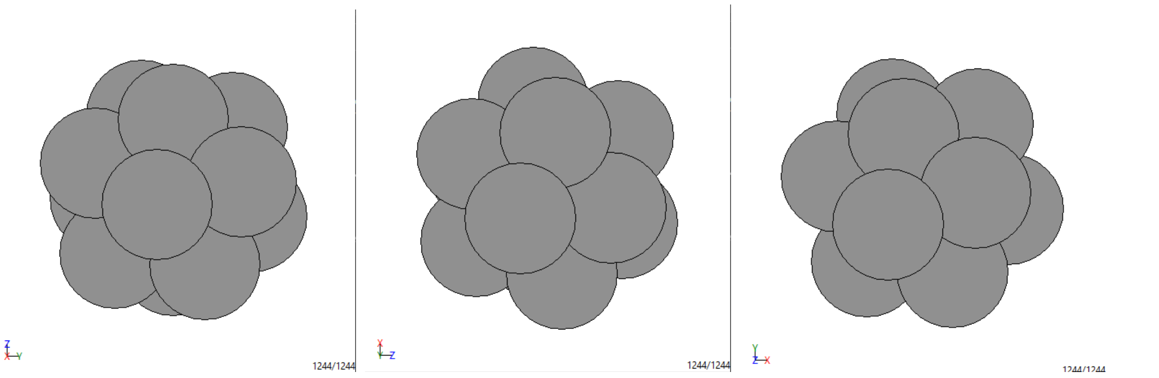

Figure 11: Visualisation of the 'magic number' N=13

Shown here is the single shell of Mackay's icosahedral conformation of N=13. Single shell means there one layer of atoms wrapping around a single core atom. There is also other 'magic number' at N=55 and N=147 with each correspondingly has two and three shell of Mackay's icosahedra.

7 Conclusion

From the graph section 5 the minimum potential energy is reached and the atoms do conform into a nanoparticle at their stable configuration. All the algorithms mentioned was applied successfully. Although, at the initial stage the results were not as promising. That has lead to some more numerical optimization of our own so that the algorithm runs more efficiently.

The scalability of the code is still limited and the algorithm may need further modifications to reach the best accuracy when there are changes in given parameters. For the smaller density like $N/V = 0.01$, α should be increased to $0.01N^{-2/3}$ so that atoms near the boundaries move to the origin. For the larger number of atoms higher than 25 like N=55, there was still a deviation of nine units. The Lennard-Jones cluster is however shown up in their 'magically' shape.

Below is the conformation at N=55, the less uniform structure (which contradicts what is mentioned here [1]) shows that the algorithm is less scalable at higher N, but still good enough to form a magic cluster.

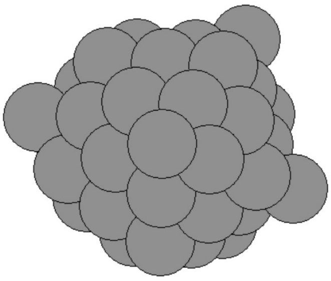

Figure 12: Conformation at N=55

The whole assignment could take a whole new approach if we also added velocities to the atoms, bonded interactions (intramolecular) and also other non-bonded interactions (intermolecular). But we have achieved the first milestone and solved our basic problem - potential energy minimization for microclusters of N neutral atoms interacting pairwise via the Lennard-Jones potential functions.

References

[1] R. H. Leary, “Global optima of lennard-jones clusters,” *Global Optimization*, vol. 11, no. 1, pp. 35–53, 1997. [Online]. Available: https://link.springer.com/article/10.1023/A:1008276425464#citeas

[2] A. Hinchchliffe, *Molecular Modelling for Beginners.* John Wiley & Sons Ltd., 2003, no. 2. [Online]. Available: https://nadjeeb.files.wordpress.com/2009/11/molecular-modelling-for-beginners.pdf

[3] M. E. S. B. R. Shashi Kant Mishra, Suvra Kanti Chakraborty, “A q-polak–ribière–polyak conjugate gradient algorithm for unconstrained optimization problems,” *Journal of Inequalities and Applications*, vol. 25, no. 3, p. 5, 2021. [Online]. Available: https://doi.org/10.1186/s13660-021-02554-6