Scientific Computing assignment 3

Odysseas Lazaridis

October 2022

1 Question a

The two plots that show the potential energy of two particles as shown bellow. We see that there is a minimum which a particle can easily sit and by using more of those we can create a stable lattice. In the second graph we see that the potential energy rises when a particle gets too close to an other particle.

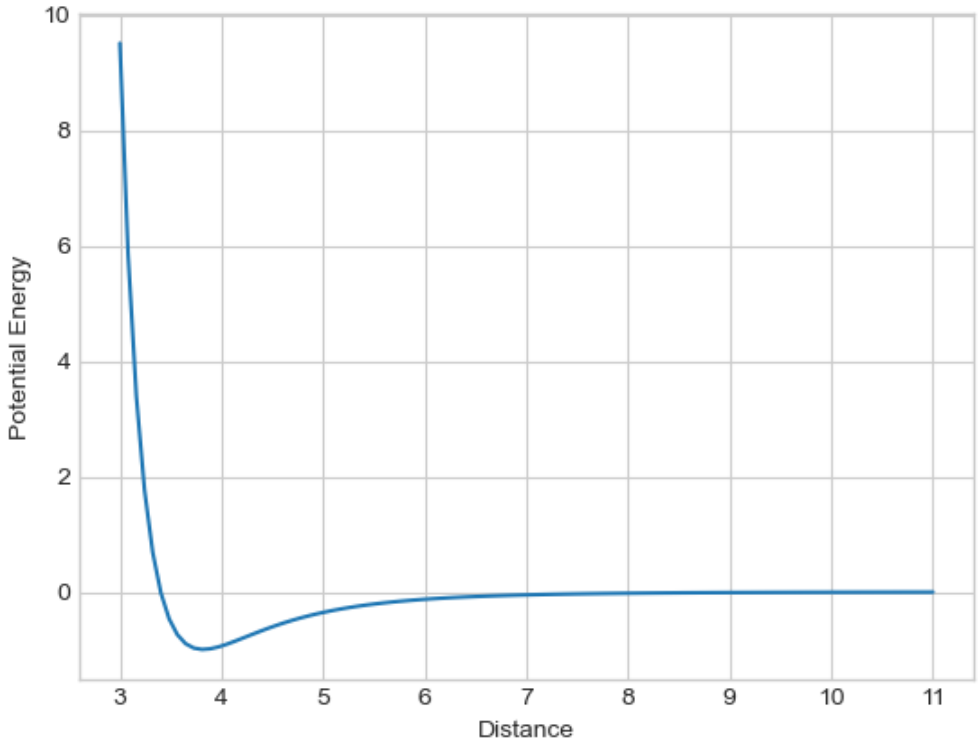

Figure 1: Potential energy as a function of the distance between two particles

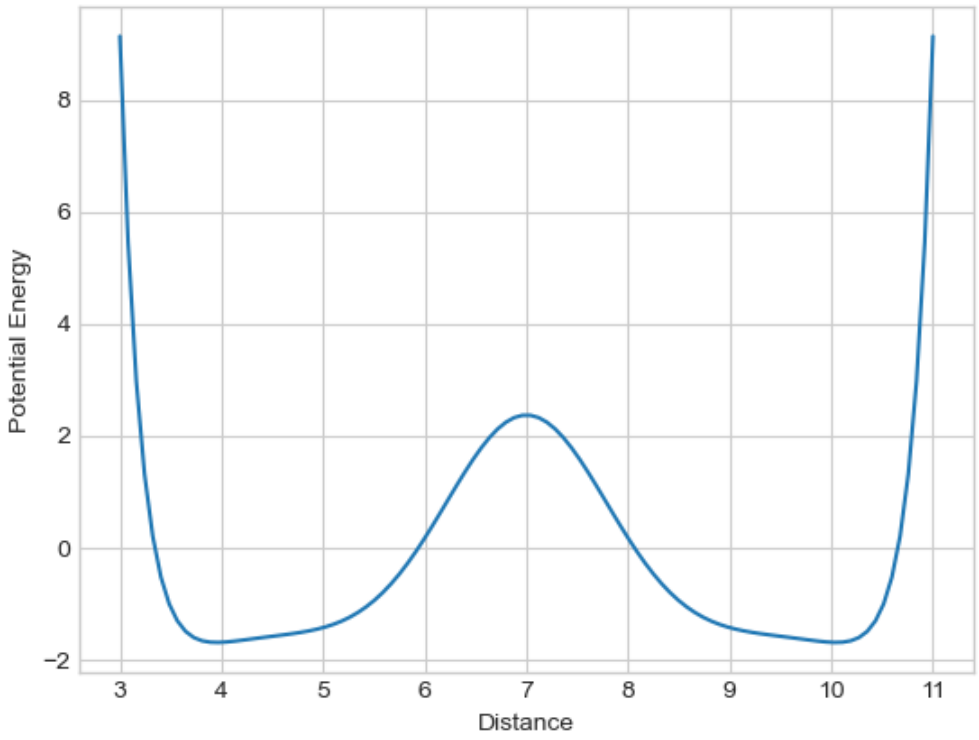

Figure 2: Avalanche size as a function of time

2 Question b

I chose to use recursion in order to calculate the zero of the function so I added some extra inputs to my function.

```
def bisection_root_method(calls,f,f_a,f_b,a,b,tolerance = 1e-13):
```

The reason is that the value that I calculate for $f(middle)$ is the $f(a)$ or $f(b)$ depending on it's sign so I don't need to calculate it again.

The potential energy becomes zero at $x = 3.4010$ angstroms which result agrees with bibliography. In order to calculate it I called the potential energy function 48 times(starting bracket [2,6]).Note that this function will converge always as long as there is at least one root in between [2,6]

3 Question c

With this method it i calculated $x = 3.401$ but with 25 calls of either the V function or gradV which shows that Newton's method converges much faster than bisection method. The only drawback with newtons method is that it will not always converge to the solution

4 Question d

Now I have to combine the two methods above to make a function that will converge no matter the starting point. So i created a function called combined_methods(a,b,x0,tolerance) that applies Newton's method until it outputs a value out of the area that I have confined the root. When that happens I apply bisection method ones and continue with Newrton's method. For a starting value of 2 it converges without the use of bisection method with 27 calls.

In order to make sure it works no matter the starting point I also used $x_0 = 5.5$ which would diverge if I only applied Newtons method. I found again the same value $x = 3.401$ with 23 calls of either V or gradV function. The number of calls are irrelevant because I started from different initial value

5 Question e

The three components represent the gradient of the Potential energy for each dimention. The non-zero component represents the movement on the axis that goes through the two particles and the other two the movement perpendicular to this axis. BOth of them are zero because an infinite small movement in those two directions doesn't change the potential energy so the gradient is zero. On the other hand, if we move even slightly on the other axis, the potential energy will change so the gradient is non-zero (printed some components of the gradient for representation)

6 Question f

In that question I am called to create a function that finds the zero of N-dimensional Nonlinear function in a restricted area which is a line. This line is given as the direction of the force that a particle has at that point. I found the zero for $a = 0.4517$ by making 43 calls.

7 Question g

In this question we implement the golden search to find the minimum of a unimodal function. The idea of this method is that by calculating the function in to points x_1, x_2 inside [a,b] we can reduce our space to either $[a, x_1]$ or $[x_2, b]$. The minimum of the function F along the line segment was found for a = 0.452 with 17 calls of the functions.

The distance between two Ar atoms was calculated again 3.817 Åand took me 20 calls.

8 Question h

In order to create the BFGS function I used the inverse Hessian method. The calculated value for the two particles was calculated again 3.817 with 628 iteration. This method seems to be unstable when I make the initial position a bit larger.

9 Question i

Now we move on to calculating the position with the lowest potential energy for more particles. For every number of particles it kept converging but in distances way bigger than the distances we found

before. It was only successful for 3 particles as can be seen in the graphs bellow.

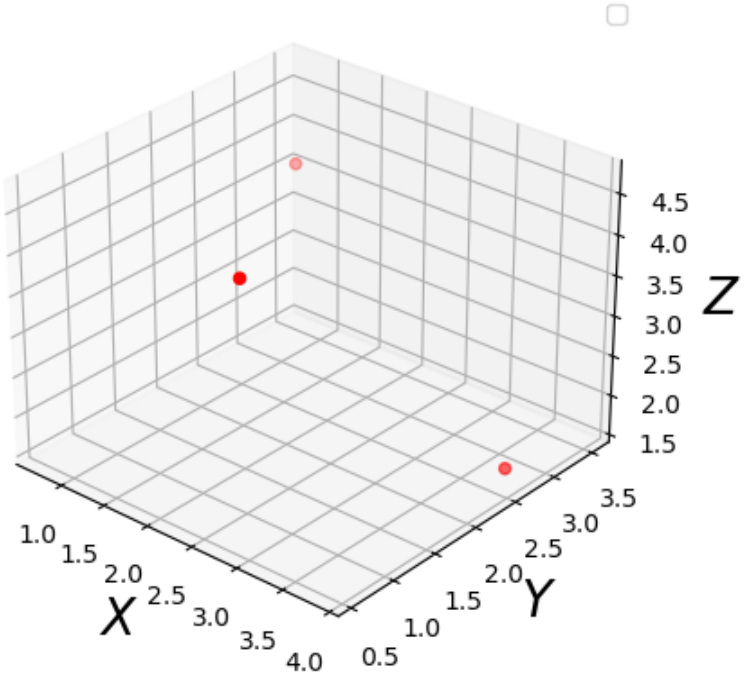

Figure 3: Position for 3 particles system

It can be seen clearly that the distances are way more than 3.81Åthat was calculated all the previous times so that this method does not converge for these initial states. Next I show the number of particles that closer together(within 1% of the 2-particles optimum distance).

Number of particles	2	3	4	5	6	7	8	9	20
Total number distances	1	3	6	10	15	21	28	36	190
Number of distances within 1% of the two-particle optimum	1	3	1	0	3	1	0	0	1

It seems that this method pushes apart the particles that happened to be close together in the initial state and bring closer the ones that happened to be closer. What happens is that how much the x will change has to do with the gradient potential energy. If the particles are close to each other, on the next step they get pushed very far away, and when they are that far away the steps are so small that they never get to converge.

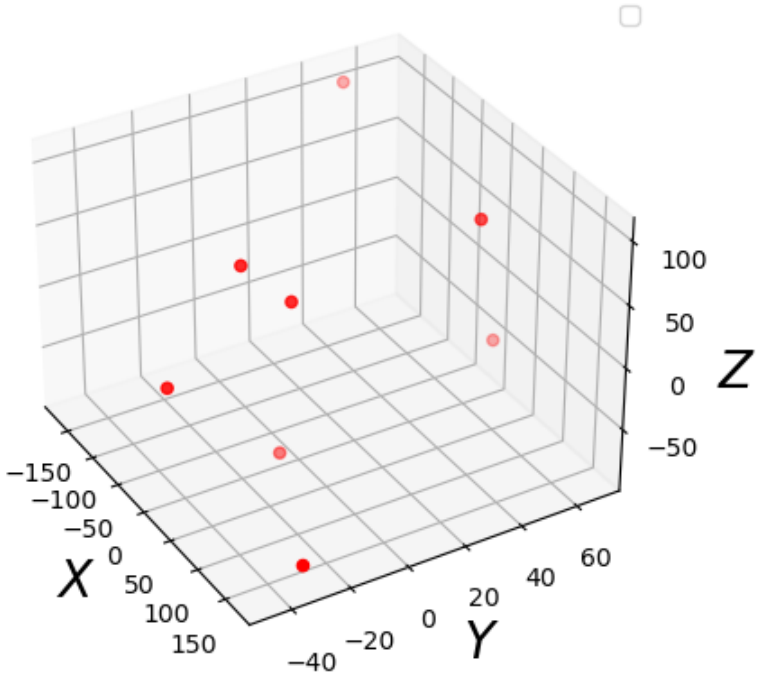

Figure 4: Position for 8 particles system

10 Question j

Now it's time to implement the linesearch method. The difference now is that with linesearch we can control the scaling of x so that it doesn't push the very far apart or close together. We can see from the number of distances that are comparable with the r_0 that this method works much better and we can see that the from the graphs that they really look like lattices.

Number of particles	2	3	4	5	6	7	8	9	20
Total number distances	1	3	6	10	15	21	28	36	190
Number of distances within 1% of the two-particle optimum	1	3	6	6	12	15	18	21	34

These results are actually very good even for the 20 particle system because if we assume that the 190 connections are equally distributed to all the particles (which of course is wrong, because the ones on the borders have less connections) we have 5.85 connections for each particle which could resemble to an actuall lattice. BY executing the cod, you can move around the 20 particles system and see that they indeed create a pretty good lattice

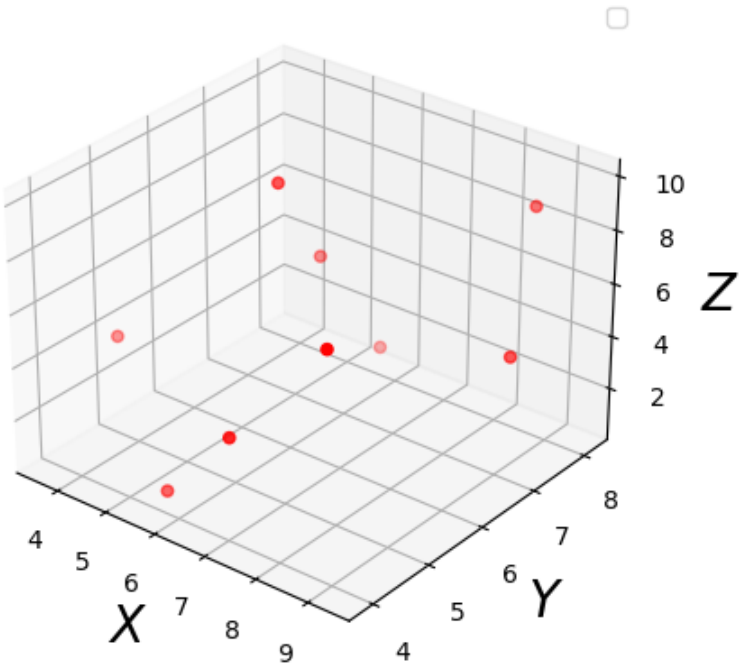

Figure 5: Position for 9 particles system with lineasearch

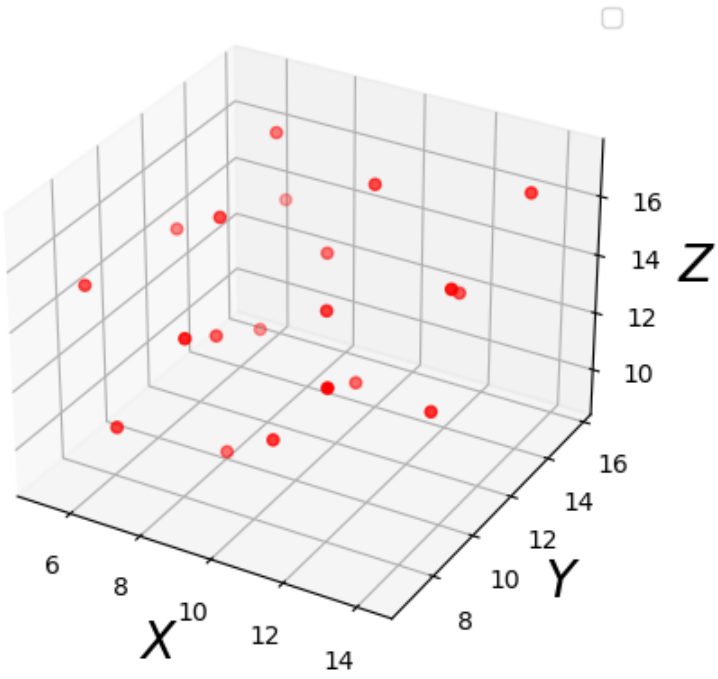

Figure 6: Position for 20 particles system with lineasearch