Laboratory 1: Finding minima of functions

Patrick Sinnott

17326757

October 13, 2018

1 Introduction

Aims and overview

The purpose of this lab was to investigate and compare different methods of finding the roots of functions. This lab focused primarily on the bisection method and the Newton-Rhapson method for finding the roots of a function. Both methods were used on an identical function:

$$f(x) = 2x^2 + 6x - 3.$$

The Newton-Raphson method was then studied further by applying it to the potential energy function:

$$V(x) = Ae^{(-x/p)} - e^2/(4\pi\varepsilon ox)$$

Principles of methods used

The bisection method is rather straightforward; it is based on the principle of choosing two values for x which are positive and negative respectably and inserting their mean into the function. Depending on the sign of the resulting value, the x-values are altered to get a better approximation of the root of the function.

The Newton-Raphson method is based on the Taylor series expansion of a function.

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

with *a* being our estimate for the root. However, in order to have a manageable equation to approximate our function, we truncate the series at the second term.

$$\mathrm{f(a+h) = f(a) + hf'(a)}$$

By repeating this approximation, it is possible to calculate an increasingly accurate value for the roots of the function.

2 Methodology

Exercise 1: Python script to find the roots of a parabolic function

1. I chose two values of x (x_1 and x_3), such that the values given for $f(x) = 2x^2 + 6x + 3$ were negative and positive respectably.
2. I took x_2 as the mean of these values of x.
3. I evaluated $f(x_2)$, and either replaced x_3 with x_2 if this result was positive or replaced x_1 with x_2 if the result was negative.
4. This process was repeated until $|f(x_2)|$ was less than a pre-determined tolerance.
5. I then calculated how many steps were required to find the root of the parabola by adding a variable called n-steps which was incremented with every subsequent approximation.
6. I plotted n-steps vs. the logarithm of the tolerance for the approximation.

Exercise 2: The Newton-Raphson method for finding roots of a function

1. I began a new Python script and initialised the variable x_1 to 1.
2. I Added a while loop to update x1 iteratively using the Newton-Raphson rule $\mathrm{x1 = x1 - f(x1)/f'(x1)}$.
3. The process was repeated until $|f(x_1)|$ was less than a pre-determined tolerance.
4. Similar to Exercise 1, I calculated the number of steps required to find the root of the function for several different values of tolerance, then plotted the steps vs. the logarithm of the tolerance.

Exercise 3: Python script to find roots of a potential energy function

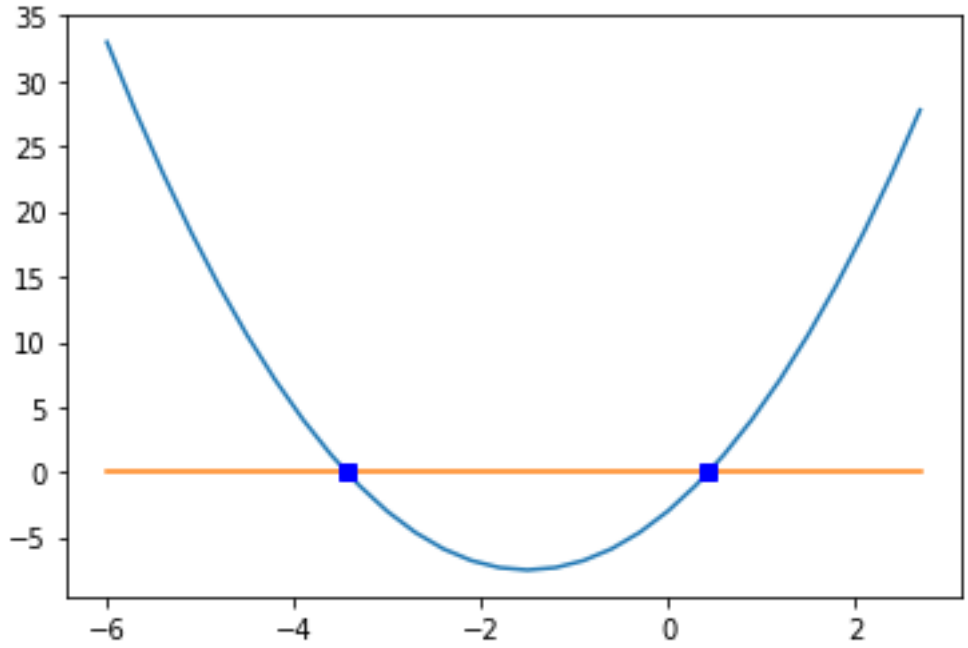
1. I defined the potential energy function: $\mathrm{V(x) = Ae^{(-x/p)} - e^2/(4\pi\varepsilon ox)}$ and wrote a python script to plot it.
2. I obtained a value for V'(x) on paper and defined it in my script.
3. Using these values, The Newton-Raphson code was applied to find the minima of the function.

3 Results

Exercise 1: Python script to find the roots of a parabolic function

This section involves analysis of the function $f(x) = 2x^2 + 6x - 3$ using the bijective method.

The roots were found analytically (using the quadratic formula: $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$) to be 0.4364916731 and -3.436491673

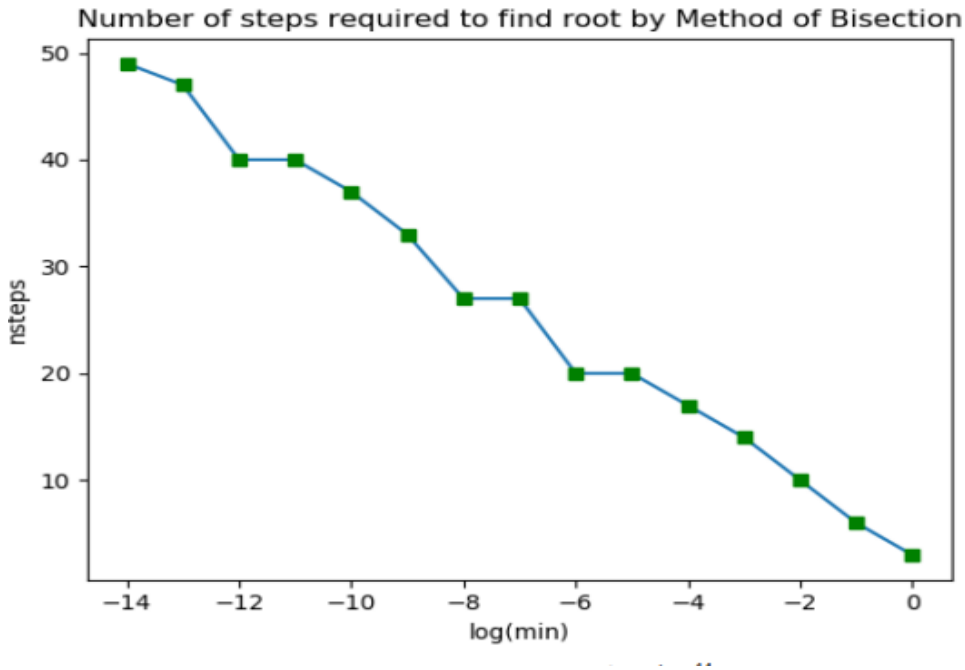


The Bijection method successfully calculated these roots after a number of iterations. I began with x1 = 2 and x3 = -2. The convergence can be seen clearly in the following table.

X1	X2	X3
2.0	1.0	0.0
1.0	0.5	0.0
0.5	0.25	0.0
0.5	0.375	0.25
0.5	0.4375	0.375
0.4375	0.40625	0.375
0.4375	0.421875	0.40625
0.4375	0.4296875	0.421875
0.4375	0.43359375	0.4296875
0.4375	0.435546875	0.43359375
0.4375	0.4365234375	0.435546875
0.4365234375	0.43603515625	0.435546875
0.4365234375	0.436279296875	0.43603515625
0.4365234375	0.4364013671875	0.436279296875
0.4365234375	0.43646240234375	0.4364013671875
0.4365234375	0.436492919921875	0.43646240234375

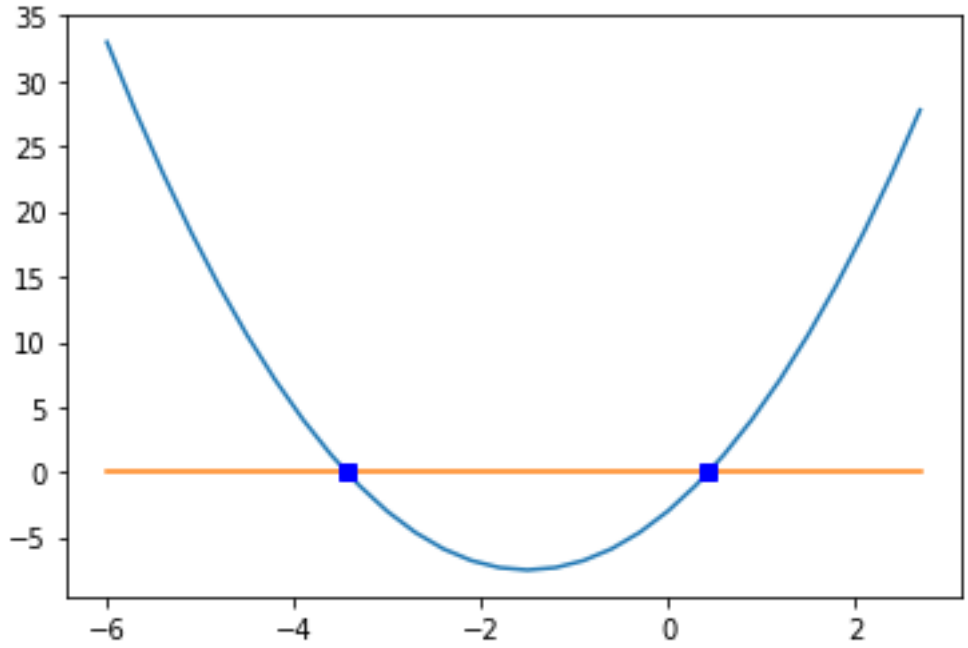
It took 16 iterations to reach the appropriate value. The other root was calculated similarly.

As one would expect, the graph of n-steps vs. log(min) showed a clear negative correlation between the two quantities. This makes sense as when you increase the tolerance level, the number of steps required to reach it decreases as you can see from the graph.



Exercise 2: The Newton-Raphson method for finding roots of a function

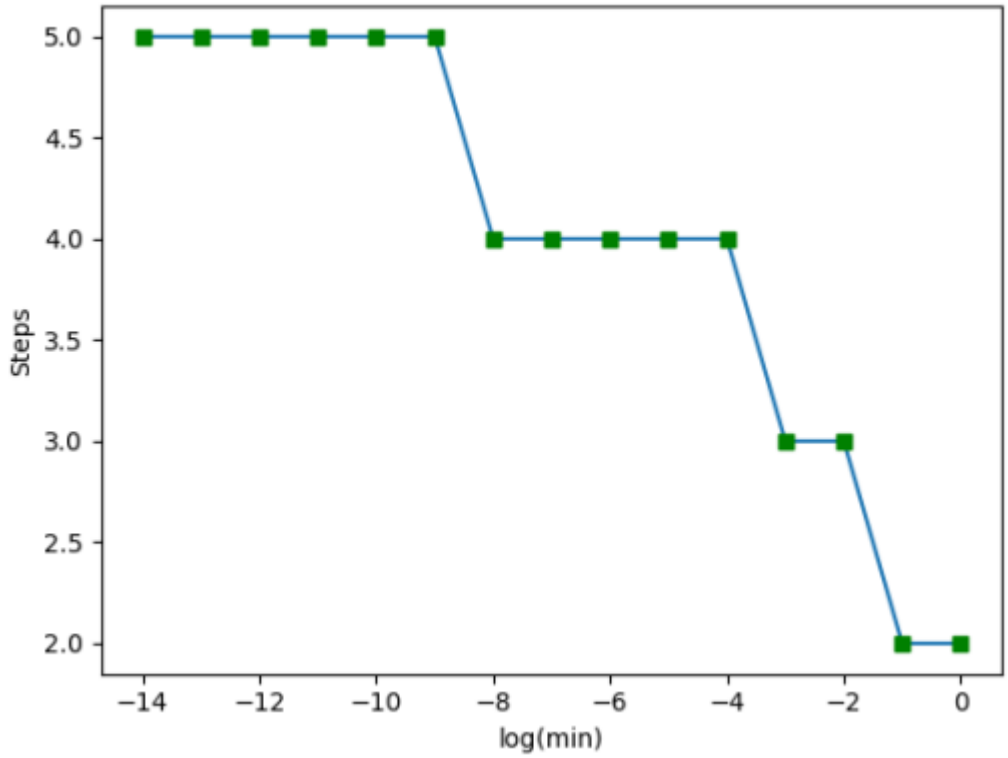
This section involves analysing the function $f(x) = 2x^2 + 6x - 3$, however this time using the Newton – Raphson method.



The Newton-Raphson method was significantly more effective than the bijection method. Whereas the bijection method took 16 iterations to reach the root; as we can see by the table, the Newton-Raphson method reached the root in only 4 steps.

X	F(x)
1.0	5
0.5	0.5
0.4375	0.0078125
0.43649193548387094	2.0323881368966568e-06

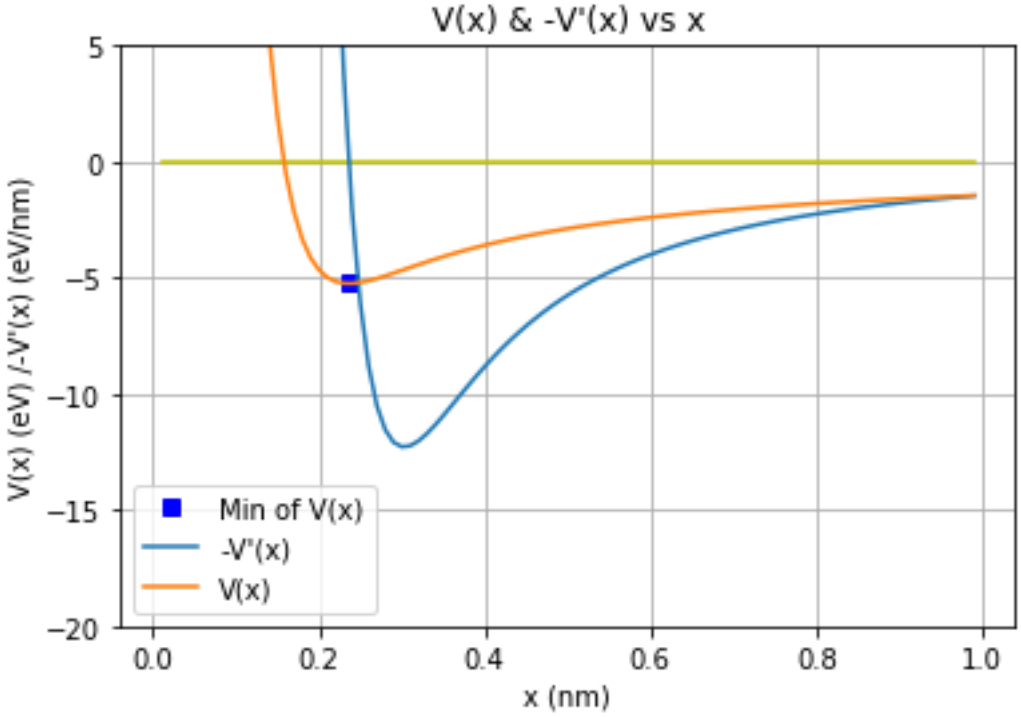
Similar to part 1, the graph of n-steps vs. log(min) showed a clear negative correlation between the two quantities.



However, Unlike the corresponding graph for the Bisection Method, our new relation is not linear but instead the number of steps required remains constant for large ranges of min and then 'steps' down to a lower value.

Exercise 3: Python script to find roots of a potential energy function

The third part of the experiment focused on the potential energy function: $V(x) = Ae^{(-x/p)} - e^2/(4\pi\varepsilon_0 x)$ specifically between ions such as Na+ and Cl-. The Newton-Raphson method was once again very effective. This time it was used to find a minimum value which can be seen in this graph.



I analysed the function using its derivatives.

V'(x) = $-\left(Ae^{(-x/p)}\right)/p - e^2/(4PiEox^2)$

V''(x) = $-\left(Ae^{(-x/p)}\right)/p^2 - e^2/(4PiEox^3)$

As you can see in the graph, the minimum was found to be 0.2360538442nm. This corresponds to r0, the Bond Length of NaCl.

4 Conclusions

In conclusion, the main aims of the experiment were achieved:

- The initial program which used the bijection method was a useful tool for calculating the roots of functions. However, it was tedious to have to select two different x-values to begin the process. Further analysis showed a clear negative relationship between n-steps (The number of steps required to reach the approximation) and min (The degree of accuracy of our approximation.) If I was to repeat this experiment I would choose a function with integer roots it was often hard to check how accurate the calculation was due to the non-rational nature of the roots.

- The Newton-Raphson method was significantly more successful at calculating the roots of our function. It was given the same tolerance but calculated the roots in far less steps. Although each method carries different restrictions ie. Newton-Raphson requires a non-zero derivative, Bijection method requires two x-values at either side of the root. The Newton-Raphson method would be recommended whenever possible.

- In the final part of the lab we applied the skills we had learned to a real-life problem. The bond length of the Na-Cl molecule was calculated accurately using the Newton-Raphson method. Obviously, the computational method of calculating this quantity is far more efficient than attempting these calculations by hand. This demonstrates the utility of the skills we acquired during this lab.