PHYC30012 Project 1: Semiclassical Quantisation of Molecular Vibrations

Diatomic molecules experience two opposing potentials, due to the Van der Waals force (from dipole attractions[1]) and the Pauli and Coulomb forces (from nuclear repulsion). At small separations, the repulsive force is stronger; and at large separations, the attractive force is stronger. The net result is that the molecules will "vibrate" as the opposing forces interplay, stretching and compressing their molecular bonds. *Here, we aim to calculate the allowed energy levels of vibration for molecular hydrogen, and use these to investigate the Morse potential as a model for the actual potential of molecular hydrogen.*

To simplify our investigation, we will assume that the nuclei are extremely heavy and the electrons react instantaneously; both realistic and valid assumptions on the relative scales we are discussing. We will also assume the molecules do not rotate: a very elementary model, but one that greatly simplifies the calculations to follow.

We will solve as per Bohr's semi-classical approximation, by quantising the action such that:

$$S(E_n) = \frac{\sqrt{2m}}{\hbar} \oint_{r_{in}}^{r_{out}} \sqrt{E_n - V(r)}\, dr = \pi(n + \frac{1}{2}) \tag{1}$$

This is essentially equivalent to requiring the de Broglie wavelength of the molecule to fit in the path as a standing wave.
The terminals r_{in} and r_{out} are the turning points of the molecule: i.e. where momentum $p = 0$, and hence where

$$E_n - V(r) = 0 \tag{2}$$

The turning points can be determined by solving equation 2 for r, to give r_{in} and r_{out} in terms of E_n.

We redefine the problem by changing variables to $\epsilon_n \equiv \frac{E_n}{V_0}$, $v(r) \equiv \frac{V(r)}{V_0}$, and $x \equiv \frac{r}{a}$, where V_0 is the potential at equilibrium, and a is the equilibrium position. Using dimensionless variables reduces the probability of computational addition and rounding errors. Converting to these variables, the dimensionless turning points are now the solutions to:

$$\epsilon_n - v(x) = 0 \tag{3}$$

and the dimensionless energy levels are the solutions to:

$$s(\epsilon_n) = \gamma \int_{x_-}^{x_+} (\epsilon_n - v(x))^{\frac{1}{2}}\, dx - \pi(n + \frac{1}{2}) = 0 \tag{4}$$

Typically, the integral of equation 4 cannot be analytically computed, except for some select few choices of $V(r)$ - this underpins the entire difficulty of calculating energy levels. To evaluate these integrals, we will use numerical integration techniques. These techniques estimate area by dividing up the integration interval into segments, and approximating the contribution to the area under a curve. The most basic technique - a rectangular approximation - is how many students are first introduced to integration. Simply approximate each segment as a rectangle, extending either the left or right hand side to intersect the curve.
We will investigate two more sophisticated approximations: the trapezoidal method

[1] McNaught A.D, Wilkinson A. van der Waals forces [Online]. 1997. http://goldbook.iupac.org/V06597.html [20/08/2016]

approximates the area as a trapezoid, extending both the left and right hand sides of each segment to the curve (see the flow chart in Appendix 1 for more information on how the algorithm runs); and Simpson's method fits a quadratic curve between the intersection points. See Figures 1[2] and 2[3] below showing how each segment is approximated graphically.

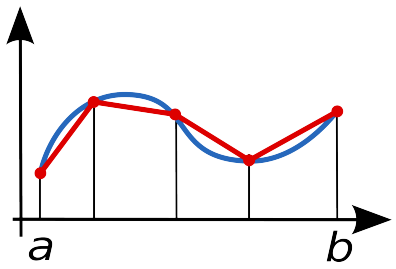


Figure 1: Trapezoidal Integration with four segments[2]

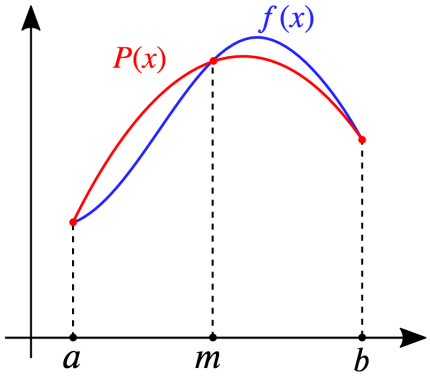


Figure 2: Simpson's Integration with one segment[3]

To determine how accurate each method is for a given number of segments, we evaulated the integral:

$$\int_0^1 \frac{ln(1+x)}{x}\,dx \tag{5}$$

which has a known value of $\frac{\pi^2}{12}$. See Table 1 and Figure 3 below for how accurately each method evaluated the integral for varying numbers of steps.

[2]Alexandrov O, Wikimedia. Composite trapezoidal rule illustration small [Online]. 1997. https://upload.wikimedia.org/wikipedia/commons/4/4b/Composite_trapezoidal_rule_illustration_small.png [19/08/2016]

[3]Popletibus, Wikimedia. Simpsons method illustration [Online]. 2016. https://upload.wikimedia.org/wikipedia/commons/c/ca/Simpsons_method_illustration.svg [19/08/2016]

Number of steps	Trapezoidal Accuracy (%)	Simpson Accuracy (%)
1	97.15245583	99.95815475
5	99.87614832	99.99933181
10	99.96894153	99.99991093
50	99.99875642	99.99999986
100	99.99968910	99.99999999
500	99.99998756	100
1,000	99.99999689	100
5,000	99.99999988	100
10,000	99.99999997	100
50,000	100	100
100,000	100	100

Table 1: Accuracy of Trapezoidal and Simpson's methods for varying numbers of steps

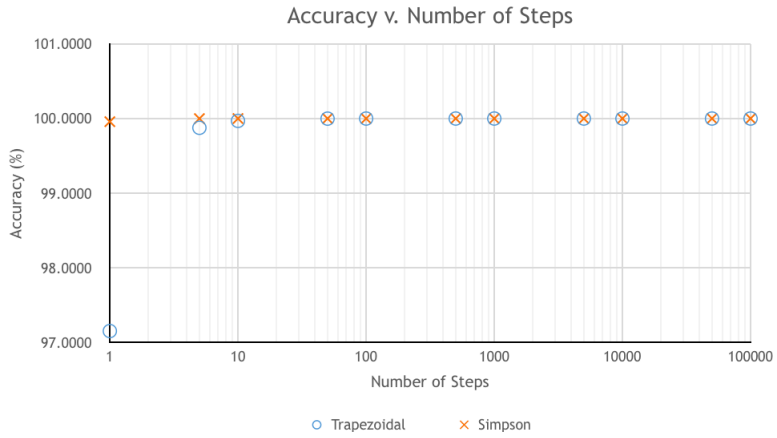


Figure 3: Accuracy of Numerical Integration Methods v. Number of Steps

Both methods converge rapidly, although Simpson's method is much faster than the trapezoidal method, converging at just 500 segments, rather than the trapezoidal's 50,000 (here working to eight decimal places for 'perfect' convergence). The difference in step number requirements may appear significant, but the computing power required to complete the calculation was insignificant: the integral was evaluated by both methods consequtively using 100,000 segments in just 0.012s; even 100,000,000 steps took only 6.633s (run on a MacBook Air with a 1.3GHz processor).
All the same, as Simpson's method does converge faster than the trapezoidal method, we will continue to use it in our following calculations.

We will also need to calculate the root of an equation (namely, solving for ϵ_n in equation 4). To do so, we will implement the false position algorithm. This requires two points, x_{i-1} and x_i that lie on either side of the root; i.e.

$$f(x_{i-1})f(x_i) < 0 \tag{6}$$

The algorithm then calculates the next point from:

$$x_{i+1} = x_i - \frac{f(x_i)(x_i - x_{i-1})}{f(x_i) - f(x_{i-1})} \tag{7}$$

And then uses either x_{i-1} or x_i along with x_{i+1} (depending on which still satisfies equation 6) to calculate the next point, until $x_{i+1} = 0$. See Appendix 2 for a flow chart of this process.

To test our implementation of this algorithm, we will solve the equation:

$$\int_0^x t^2 \, dt = x \tag{8}$$

for its non-zero root, which a moment's thought shows is $\sqrt{3}$. To do so obviously requires evaluation of an integral; as previously mentioned, we will be using Simpson's rule to do so. Rather than requiring two straddling points as inputs to our program, we will instead just require one 'guess' point. From here we can search for another point that will straddle the root, using an input step size (and reversing the direction of our search if we stray too far).

There are now four parameters that will effect the accuracy of our answers: the number of segments (used in Simpson's method), the starting position to look for a root, the step size to use when searching for a straddling point, and the tolerance level used to determine equivalency.

See Table 2 below for how varying these parameters changed the accuracy of the solution. The decimal value of $\sqrt{3}$ is 1.732051 (to six decimal places). The cells are shaded such that darker greens are less accurate.

	Number of Divisions	1,000			10,000			100,000		
	Step Size	0.1	0.5	1	0.1	0.5	1	0.1	0.5	1
Tolerance	Starting Position									
10^{-3}	0.1	1.731558	1.731829	1.731599	1.731558	1.731829	1.731599	1.731558	1.731829	1.731599
	1	1.731947	1.731644	1.731644	1.731947	1.731644	1.731644	1.731947	1.731644	1.731644
	10	1.732043	1.731986	1.731644	1.732043	1.731986	1.731644	1.732043	1.731986	1.731644
10^{-5}	0.1	1.732046	1.732047	1.732046	1.732046	1.732047	1.732046	1.732046	1.732047	1.732046
	1	1.732050	1.732048	1.732048	1.732050	1.732048	1.732048	1.732050	1.732048	1.732048
	10	1.732051	1.732050	1.732048	1.732051	1.732050	1.732048	1.732051	1.732050	1.732048
10^{-7}	0.1	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051
	1	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051
	10	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051	1.732051

Table 2: Accuracy of False Position algorithm for varying accuracy parameters

Changing the tolerance level had the most significant effect, although there were also some small variations due to step size and starting position. Overall, a small tolerance tended to lead to an accurate result.

We are now in a position to test our program to determine vibrational energy levels. In order to test its logical flow (see Appendix 3) and accuracy, we worked with

a simple potential that can be analytically evaluated. Taking the quadratic potential,

$$V(r) = 4V_0(\frac{r}{a} - 1)(\frac{r}{a} - 2) \tag{9}$$

if we divide through by V_0, we obtain the scaled potential:

$$v(x) = 4(x-1)(x-2) \tag{10}$$

Now, the turning points are the solutions to equation 3. Substituting in our expression for $v(x)$, we obtain the quadratic:

$$-4x^2 + 12x - 8 + \epsilon_n = 0 \tag{11}$$

This has the solutions:

$$x = \frac{1}{2}(3 \pm \sqrt{1+\epsilon_n}) \tag{12}$$

Hence our two turning points are $x_- = \frac{1}{2}(3 - \sqrt{1+\epsilon_n})$ and $x_+ = \frac{1}{2}(3 + \sqrt{1+\epsilon_n})$.

Now we find the solution of the equation 4. Taking $\gamma = 1$, and substituting our quadratic expression for $\epsilon_n - v(x)$, we obtain:

$$\int_{x_-}^{x_+} \sqrt{-4x^2 + 12x - 8 + \epsilon_n}\, dx - (n + \frac{1}{2})\pi = 0 \tag{13}$$

Using the provided equation:

$$\int_{x_-}^{x_+} \sqrt{ax^2 + bx + c}\, dx = \frac{(4ac - b^2)\pi}{8a\sqrt{-a}}, \quad a < 0 \tag{14}$$

We obtain:

$$\frac{1}{4}(1 + \epsilon_n)\pi - (n + \frac{1}{2})\pi = 0 \tag{15}$$

And hence:

$$\epsilon_n = 4n + 1 \tag{16}$$

Taking $V_0 = 1$, the first few energy levels are:
$E_0 = 1$ $E_1 = 5\text{eV}$ $E_2 = 9\text{eV}$ $E_3 = 13\text{eV}$ $E_4 = 17\text{eV}$
and so on, increasing in steps of 4.

See Tables 3 and 4 over page for the program output for varying accuracy parameters for the n=0 and n=3 cases.

Evidently, again the step size and starting position make no noticeable impact on the result. For these tolerances there is no change in accuracy, as opposed to the marked improvement between 10^{-5} and 10^{-7} that we observed in our prior analysis of the false position function. The improvements appear to be solely due to the increase in number of divisions (at least to this level of accuracy - six decimal places). Looking at the difference between the n=0 and n=3 cases, we observe that for 10,000 divisions, even though the absolute size of the error grows as n increases, in fact the error as a percentage of the result decreases: from $10^{-8}\%$ to $4.7 \times 10^{-9}\%$. Regardless, it's an incredibly small error and overall an extremely stable result.

	Number of Divisions	10,000		100,000	
	Step Size	0.1	1	0.1	1
Tolerance	Starting Position				
10^(-5)	0.1	1.000001	1.000001	1.000000	1.000000
	1	1.000001	1.000001	1.000000	1.000000
10^(-7)	0.1	1.000001	1.000001	1.000000	1.000000
	1	1.000001	1.000001	1.000000	1.000000

Table 3: ϵ_0 for varying accuracy parameters

	Number of Divisions	10,000		100,000	
	Step Size	0.1	1	0.1	1
Tolerance	Starting Position				
10^(-5)	0.1	13.000006	13.000006	13.000000	13.000000
	1	13.000006	13.000006	13.000000	13.000000
10^(-7)	0.1	13.000006	13.000006	13.000000	13.000000
	1	13.000006	13.000006	13.000000	13.000000

Table 4: ϵ_3 for varying accuracy parameters

Now, we consider a realistic potential: the Morse potential.

$$V_{Morse}(r) = V_0[(1 - e^{-(r-r_{min})/a})^2 - 1] \tag{17}$$

A moment's inspection shows that this cannot be easy to integrate analytically. Instead, we will locate the turning points analytically, and use our numeric methods to determine the energy levels.
First, the normalised potential is given by:

$$v_{Morse}(x) = (1 - e^{-x+\frac{r_{min}}{a}})^2 - 1 \tag{18}$$

So we need to solve the equation:

$$\epsilon_n - v_{Morse}(x) = 0$$

$$(1 - e^{-x+\frac{r_{min}}{a}})^2 = \epsilon_n + 1$$

$$1 - e^{-x+\frac{r_{min}}{a}} = \pm\sqrt{\epsilon_n + 1}$$

$$e^{-x+\frac{r_{min}}{a}} = 1 \mp \sqrt{\epsilon_n + 1}$$

$$-x + \frac{r_{min}}{a} = ln(1 \mp \sqrt{\epsilon_n + 1})$$

$$x = \frac{r_{min}}{a} - ln(1 \mp \sqrt{\epsilon_n + 1})$$

As $\epsilon_n = \frac{E_n}{V_0}$, where $V_0 > 0$ and $E_n < 0$, ϵ_n must be negative; and as $\|V_0\| > \|E_n\|$, for all n, $\|\epsilon_n\| < 1$. Hence the argument of the natural logarithm lies between 0 and 2, exclusive (a valid range of inputs).
As $-1 < \epsilon_n < 0$, we know that $ln(1 - \sqrt{\epsilon_n + 1}) < 0$, and $ln(1 + \sqrt{\epsilon_n + 1}) > 0$. Hence:

$$x_- = \frac{r_{min}}{a} - ln(1 + \sqrt{\epsilon_n + 1})$$

$$x_+ = \frac{r_{min}}{a} - ln(1 - \sqrt{\epsilon_n + 1})$$

As ϵ_n has a very small range of valid values, our program is brittle when it comes to drastically varying the accuracy parameters. As such, the starting position and step size for this stage of the project have been hard-coded into the program as -0.5 and 0.01, to preserve output validity. The value of γ for molecular hydrogen has been provided as $\gamma = 33.6567a$, where a is undetermined. As such, we will vary the value of a to best fit several energy levels.
For the $n = 0$ first energy level, the optimal value of a was calculated to be $a = 0.514840$. The calculated energy levels and the known values are shown below in Table 5.

Energy Level	Calculated Energy (eV)	Known Energy (eV)	Error (%)
0	-4.477	-4.477	0
1	-3.961	-3.962	0
2	-3.347	-3.475	0
3	-3.023	-3.017	0.2
4	-2.602	-2.587	0.6
5	-2.212	-2.185	1.2
6	-1.854	-1.811	2.4
7	-1.527	-1.466	4.2
8	-1.232	-1.151	7.0
9	-0.969	-0.867	11.7
10	-0.737	-0.615	19.8
11	-0.537	-0.400	34.2
12	-0.368	-0.225	63.8
13	-0.232	-0.095	146.4
14	-0.126	-0.017	643.7

Table 5: Energy levels using the Morse potential with a = 0.514840

For the $n = 1$ second energy level, the optimal value of a was calculated to be $a = 0.515723$. The calculated energy levels and the known values are shown over page in Table 6.

The accuracy of these energy levels is very good for levels close to those for the optimised a. They drift considerably for the largest values of n. In order to investigate whether this drift is due to the large distance between the $n = 1$ and $n = 14$ energy levels, or if it is inherent from the Morse potential, we perform the same calculation for the $n = 10$ energy level, which has an optimal value $a = 0.487412$, as shown below in Table 7.

Energy Level	Calculated Energy (eV)	Known Energy (eV)	Error (%)
0	-4.477	-4.477	0
1	-3.962	-3.962	0
2	-3.478	-3.475	0.1
3	-3.026	-3.017	0.3
4	-2.605	-2.587	0.7
5	-2.215	-2.185	1.4
6	-1.857	-1.811	2.6
7	-1.531	-1.466	4.4
8	-1.236	-1.151	7.4
9	-0.973	-0.867	12.2
10	-0.741	-0.615	20.5
11	-0.541	-0.400	35.1
12	-0.372	-0.225	65.2
13	-0.234	-0.095	149.4
14	-0.129	-0.017	656.8

Table 6: Energy levels using the Morse potential with a = 0.515723

Energy Level	Calculated Energy (eV)	Known Energy (eV)	Error (%)
0	-4.462	-4.477	0.3
1	-3.919	-3.962	1.1
2	-3.410	-3.475	1.9
3	-2.938	-3.017	2.6
4	-2.500	-2.587	3.4
5	-2.010	-2.185	4.0
6	-1.730	-1.811	4.4
7	-1.399	-1.466	4.6
8	-1.102	-1.151	4.2
9	-0.841	-0.867	3.0
10	-0.615	-0.615	0
11	-0.424	-0.400	6.1
12	-0.269	-0.225	19.5
13	-0.149	-0.095	58.3
14	-0.064	-0.017	276.4

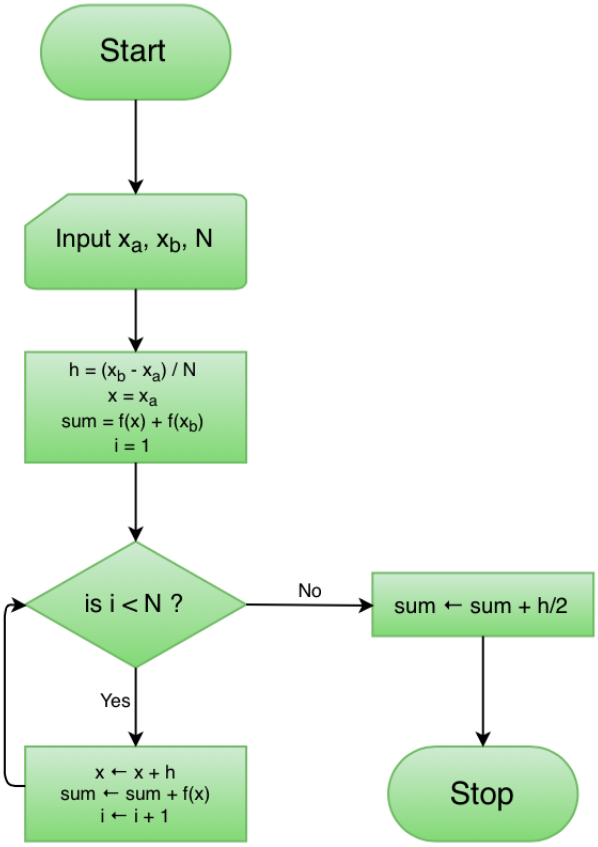
Table 7: Energy levels using the Morse potential with a = 0.487412

We still observe a large error for the highest energy levels even when we optimise a for a close energy level.

Conclusions By creating programs that are able to numerically evaluate integrals and locate roots with high precision, we were able to combine these into a powerful program that is able to compute the root of an integral equation.
We tested the validity and accuracy of this program by inputting an equation that can by analytically solved. The output showed both that the program's logical flow worked as intended, and that the results retained the high levels of accuracy shown by its component functions.

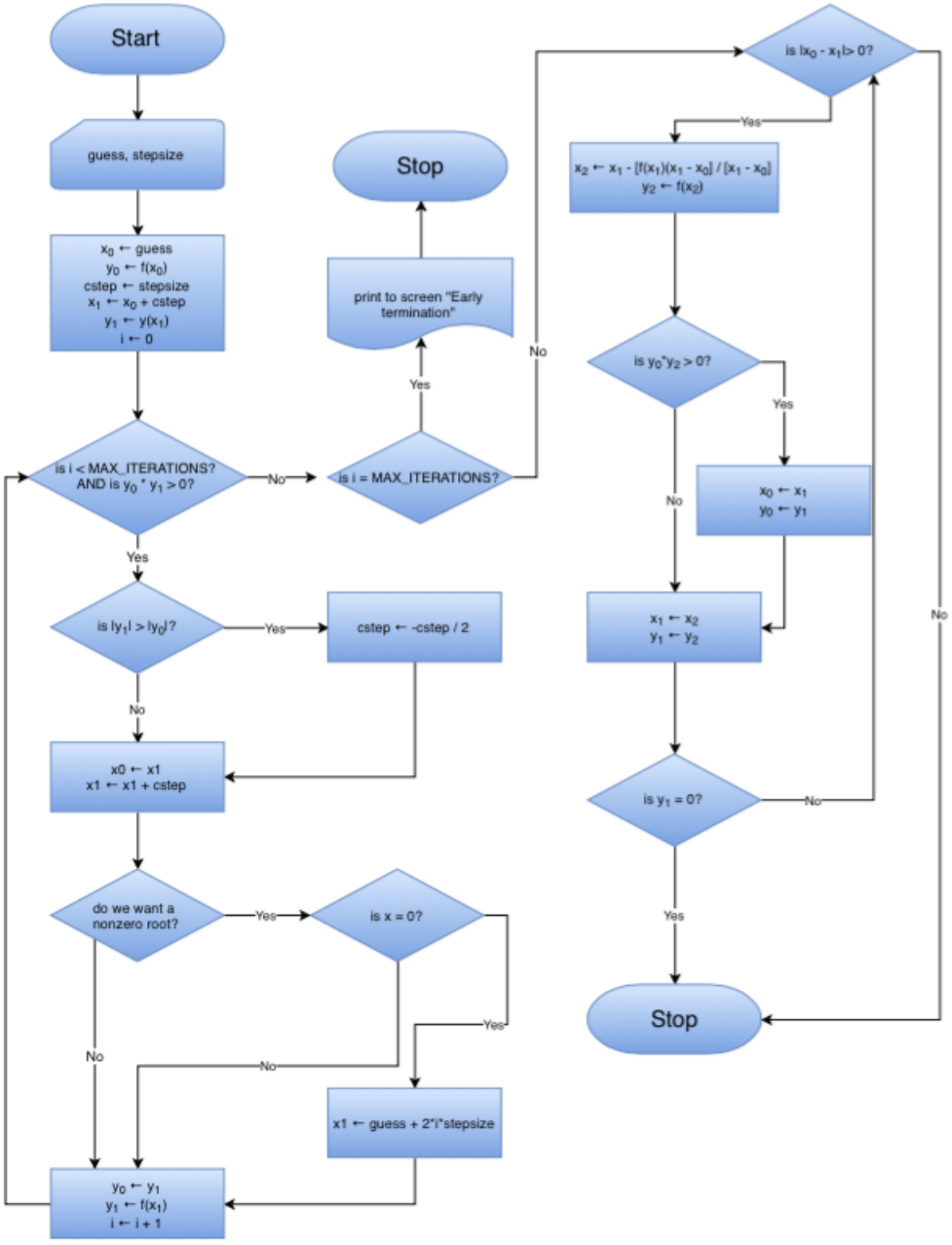
To evaluate the vibrational energy levels of molecular hydrogen, we used the Morse potential as the potential function. By adjusting the value of a, the equilibrium position, for several energy levels, we were able to fit the calculated energy levels to their known value, which generally provided a reasonable fit for other nearby energy levels. This reasonable fit degraded sharply for the higher energy levels, $n = 12, 13, 14$, where the validity of the Morse potential seems to decrease. By adjusting a to fit a high energy level and observing the outcome, we were able to conclude that the Morse potential loses accuracy for high energy levels.

Appendix 1



Appendix 1: Flow chart for the trapezoidal method

Appendix 2



Appendix 2: Flow chart for the false position algorithm

Appendix 3

Appendix 3: Flow chart for determining vibrational energy levels