Cu Hauw (Willy) Hung

Numerical Analysis

Project 1 (Spectrum Analysis)

Intro:

This project involves us analyzing different spectral lines from stars and obtaining information about these spectral lines. This methodology falls under the study of spectroscopy. According to NASA, “Spectroscopy is a scientific measurement technique. It measures light that is emitted, absorbed, or scattered by materials and can be used to study, identify and quantify those materials. In other words, knowing the frequencies of the spectral lines allows us to know a specific planet’s chemical composition, which might be crucial information for scientists.

For this project, the astronomer gave me a spreadsheet containing a set of spectral data, measurements of the light intensity from a star at 4000 frequencies and wants to gain more information regarding the spectral lines. Since the data provided is discrete, we are not able to use an adaptive Simpson method to integrate over the given integral. In addition, the continuous line on the overall has a negative slope, which means that we have to keep this in mind and adjust our methods accordingly. Due to these reasons, we have to use different numerical methods in order to perform the necessary tasks for our analysis.

After plotting the data set (x=Frequency, y=Specific intensity), we can clearly see that there six spectral lines, three of which are absorption lines (curves that point downwards) and three of which are emission lines (lines that point upwards). In this project, we are going to label the spectral lines as: A, B, C, D, E, and F. We have to deal with the fourth spectral line differently, because there are two nodes on the fourth curve: there is one with a bigger node and another one with a smaller node. In order to differentiate the difference between these two nodes, we have to name the bigger node on the left as D, and the smaller node on the right as E. Our astronomer friend would like to know the line strength of each of these lines, which is intensity of each line integrated over frequency. This project requires multiple steps and the outline of these steps are presented in the following section.

Outline:

1. Find the intervals in which we should integrate over
2. Use Simpson’s rule to find the whole area between the two points
3. Use the formula of a trapezoid to calculate the area that is not required
4. Find the difference between the Simpson and the calculated area

Finding the optimal interval:

There are numerous ways to find the optimal interval in which we should perform our analysis over. One way to do this is to define a function of a line between two points on the smooth continuous spectrum or in other words, the “straight” parts of the plot. After defining this function, we can then check each point on the data set that has a ‘y’ distance away from the defined line. This will tell us that the data points are deviating away from the defined line, which means that curve is starting to go up or down, marking the beginning/ending of a spectral line; this will tell us the points in which we should perform our analysis over. However, we should use a different method.

The method used instead is based on the assumption that all the spectral lines area symmetrical. This method is used over the one described in the previous paragraph, because this method does not require us to define five new line functions, but instead uses loops to compare the Y values of points. Firstly, we have to create a for loop that runs through an estimated interval of the different spectral lines and calcualte the absolute value of the y-distance (Δy) between point ‘i’ and point ‘i+1.’ Then, Δy gets stored in a temporary variable called minValue. Then this variable minValue gets updated whenever Δy obtained between ‘i’ and ‘i+1’ is smaller than the one that is currently stored; the index in which the minValue gets updated is also stored. We should use the smallest Δy instead of the largest Δy, because we want to obtain the index on the smooth continuous spectrum where the graph is most “stable” and least likely to increases or decrease. In addition, while the for loop was going on, we should also keep track of the highest points on the spectral line and store the index of the extrema in a temporary variable.

Once we have obtained the first point (point where the spectral line opens up), we can then use assumption of symmetry and the index of the extrema, which we have stored, to find the second point (point where the spectral line closes). We can obtain the second point by finding the difference between the index of the first point and the extrema and then add it to the index of extrema; this would give us the index of the second point.

The process described above worked for all the spectral lines pointing upwards. We had to make minor adjustments for the spectral lines going downwards. These adjustments include reversing the extrema that was stored and reversing the index in which the for loop was running. Instead of storing the maxima of a given interval, we instead store the minima. Another adjustment is changing the way the for loop runs. Instead of having the for loop run from ‘a’ to ‘b’ in intervals of ‘1’, we instead should have the for loop run from ‘b’ to ‘a’ in intervals of ‘-1’. In the function we have created, we have an extra input variable called “up” to tell the function if spectral line is opening upwards or downwards. The functions ‘output’ and ‘input’ is as follow:

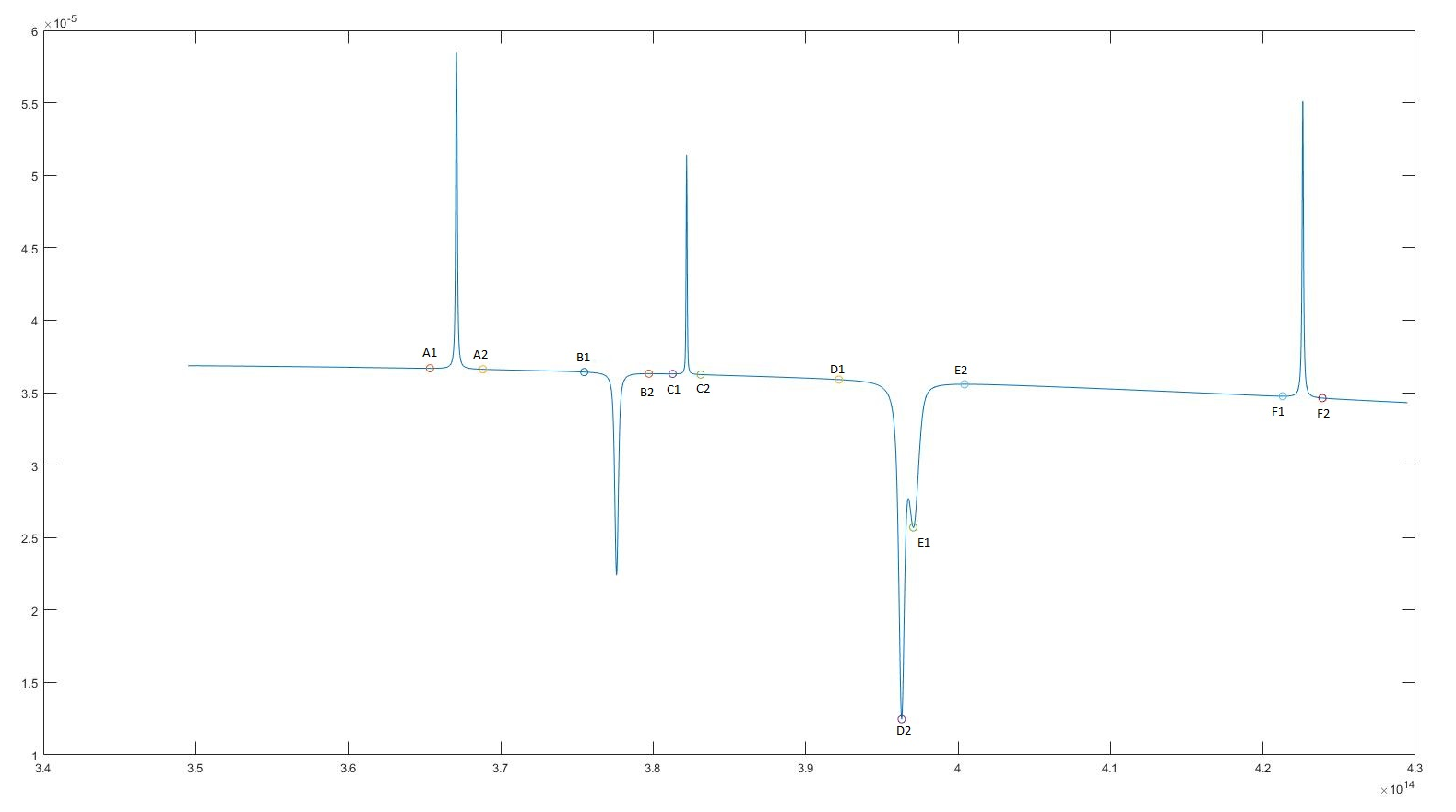
|  |
| --- |
| function [point1, point2] = FindPoints (a, b, x, y, up)   * Point 1 is the point where the spectral line opens up; point 2 is the point where the spectral line closes * ‘a’ and ‘b’ are the estimated intervals in which we are trying to find the optimal intervals we are trying to perform analysis on * X and y are the astronomer’s data points * ‘up’ tells us if the spectral line is pointing upwards or downwards |

This method worked for the spectral lines A, B, C, and F. The left point of the spectral line ‘A’ is labeled as ‘A1,’ the right point of the same spectral line is labeled as ‘A2.’ The following labeling format is done for the other spectral lines; B1, B2, C2, etc.

Adjustments had to be made in order to deal with the node with the double nodes. The SamePoint function was able to find the points in which the fourth spectral line opened up and closed up (D1, E2). However, they couldn’t find the two nodes contained within the spectral line. So, another function titled FindLow was created to help us find the nodes of the two points in the fourth spectral line. The function gave us the points for D2 and E2. Once we have obtained all the intervals, we can simply apply Simpson’s rule and the trapezoid formula, which will be discussed in the following section, to these intervals.

A table of the data points:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Spectral Lines | Left | X-Value | Y-Value | Right | X-Value | Y-Value |
| A | 794 | 365.00e+12 | 36.6799e-06 | 968 | 368.84e+12 | 36.612e-06 |
| B | 1300 | 375.48e+12 | 36.4208e-06 | 1512 | 379.72e+12 | 36.3103e-06 |
| C | 1590 | 381.28e+12 | 36.2943e-06 | 1682 | 383.12e+12 | 36.2451e-06 |
| D | 2135 | 392.18e+12 | 35.8987e-06 | 2341 | 396.3e+12 | 12.4697e-06 |
| E | 2379 | 397.060e+12 | 25.6963e-06 | 2547 | 400.42e+12 | 35.5771e-06 |
| F | 3591 | 421.3e+12 | 34.7524e-06 | 3721 | 423.9e+12 | 34.6241e-06 |

A plot of the data points, along with the intervals is shown in the following figure: 

Both functions used in this section, titled FindPoints and FindLow, can be found in the appendi

x.

Simpson’s rule

General form of the Simpson’s rule (Obtained from Wikipedia):

A simplified version would look like this:

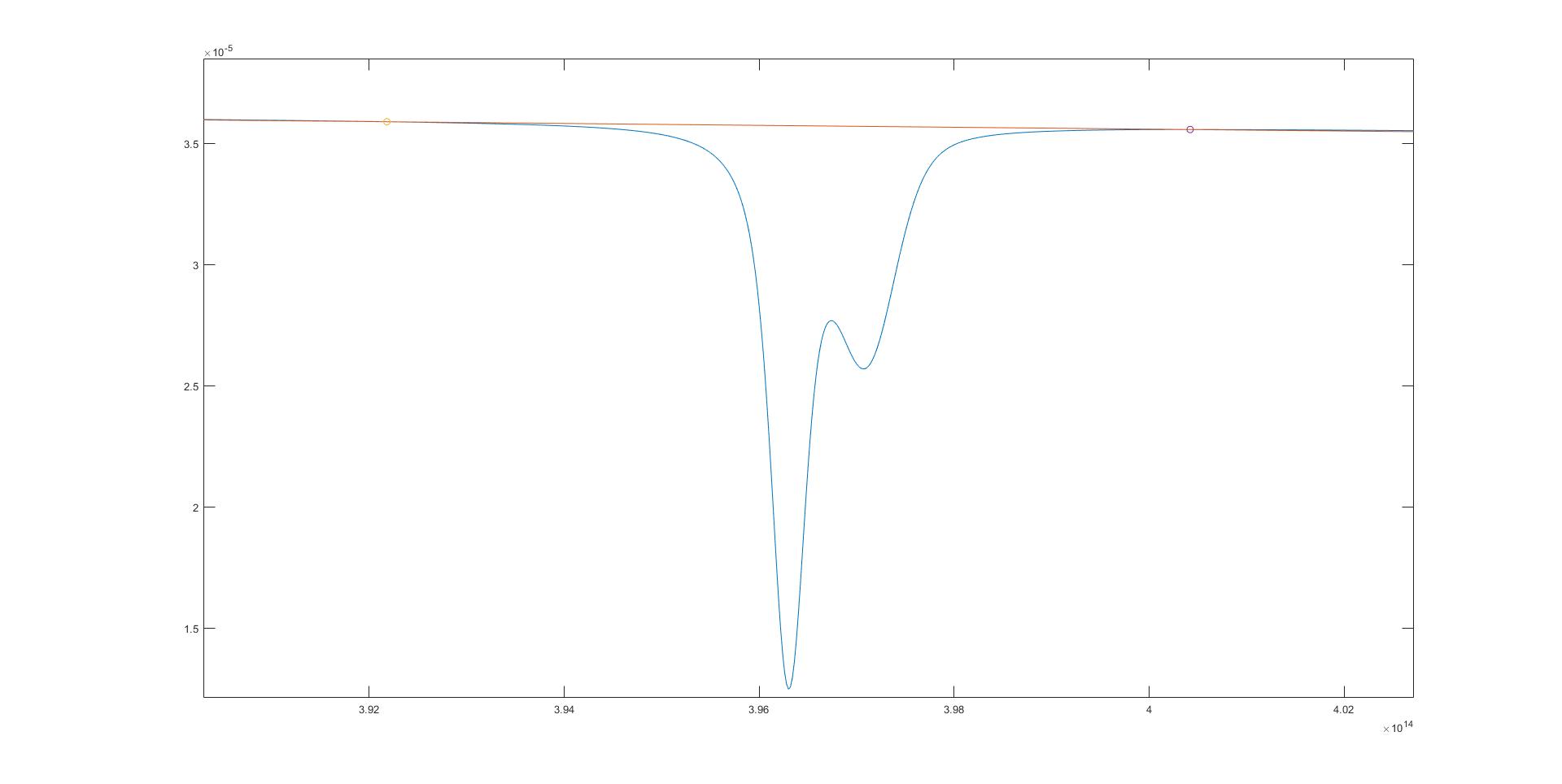
Before the Simpson function was performed on the data points, it was first tested on sample data points, which we already knew the answer of. This was to make sure the Simpson’s rule was fully functional. Then, The Simpson’s formula was performed on the intervals we have obtained in the first section. This would generate the area under the interval for spectrum A, B, C, etc. However, for spectral lines D and E (spectral line with two nodes), we have to multiply the area obtained using the Simpson’s rule by 2. This is because the area obtained using Simpson’s rule in these two cases is from the opening of the spectral line to the local minima, which is only half of the spectral line. Here, we have to use the assumption of symmetry in order “reflect” the area. This matlab function, along with its explanations, can be found in the appendix.

Trapezoid formula/Integrating over a line

In this section, we simply find the area of the trapezoid under the pair of optimal points (A1 and A2; B1 and B2, etc.). A function is created to help us find the area between the optimal points of the interval. This could be done for all the spectral lines. However, in the case of the fourth spectral line with the two nodes, this method would not work and would give us a trapezoid that does not represent the intended area, skewing our data. In order to overcome this problem, we have to define a new function of a line using points D1 and E2, which in Matlab looks like this:

|  |
| --- |
| f = @(z) ((y(E2)-y(D1))/(x(E2)-x(D1)) \*(z-x(D1)) +y(D1)); |

Once we have obtained this function, we can then take the integral of this line from points E1 to E2 and from D1 to D2 and stored it in temporary variables; doing so would give us the chuck of area under the line in the interval. However, we have to multiply both of the obtained areas by 2. This is because the area obtained using the integral in these two cases is from the opening of the spectral line to the local minima, which is half of what we are looking for (similar logic used when finding for Simpson’s rule in these two intervals). The defined line plotted on top of the data plots is shown in the following figure:

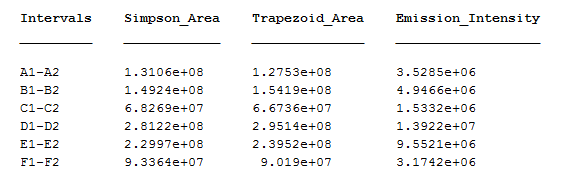


Output

After finding the approximation of the Simpsons over the optimal interval and finding the trapezoidal area (or in the case of the double node, the integral) over the optimal intervals. We just have to subtract the trapezoid area and the Simpson area and take the absolute value of that value; doing so would give us the emission intensity of the spectral line. This is the formula:

|  |
| --- |
| Emission intensity = |Simpson\_Area – Trapezoid\_Area| |

This formula will help us find the emission intensity of the different spectral lines. In the last section of my main, I organized all the data I have obtained into a single table. This helps us visualize the data and gives us a better understanding of the project.



Just from comparing the data found in the emission intensity column with the plot of the data points, we can clearly see that the fourth spectral line, which contains the intervals between D1-D2 and E1-E2 has the biggest emission intensity, which goes along well with our plot. From doing this whole procedure, the emission intensity of the first spectral line, A, is 3.52e+106, the emission line of the second spectral line, B, is 4.9466e+06, the emission intensity of the third spectral line, C, is 1.532e+06, the emission intensity of the fourth one, D, which is the bigger half of the two nodes and calculated based on the symmetry is 1.3922e+07, the emission intensity of the fifth spectral line, E, which is the smaller half of the two nodes is 9.5521e+06, and the emission intensity of the last spectral line, F, is 3.1742e+06.

Conclusion

In conclusion, the FindPoints and FindLow function were used to find the intervals over the spectral lines in which we should perform our analysis. We had to calculate the the points differently for the lines pointing upwards, downwards, and the one with two nodes. Once we have obtained all our points, we then used Simpson to calculate the total area under the different interval. Then we use the trapezoid formula over the same points to find the area of the trapezoid. Finally, we subtracted the trapezoid area from the Simpson’s area to find the area under the spectral lines, but bounded by the smooth continuous spectrum, which is the emission intensity.

Code Appendix

|  |  |
| --- | --- |
| **Simpson Function**  function [Simpson\_output] = Simpson(x, y, a, b)  deltX=((x(b)-x(a))/(b-a));    Simpson\_output=y(a)+y(b);  odd=true;    for z=(a+1):1:(b-1)  if odd==true;  Simpson\_output=Simpson\_output+(4\*y(z));  odd=false;  else  Simpson\_output=Simpson\_output+(2\*y(z));  odd=true;  end    end    Simpson\_output=(deltX/3)\*Simpson\_output;  end | The generic Simpson function to find the area between two given points (a &b)  I added the first and last term on the list  Then I ran a loop to alternate between even and odd. Since, the 2nd term (x1) is predetermined to start as an odd integer, I had the loop add this one first. Then I just switched the if/else statement back and forth to sum up the odd integers\*4 and the even integers\*2. Once I summed up everything, I multiplied the total by delta X over 3, which is the formula for Simpson’s rule. |
| **Trapezoid Function**  function [area] = Trap(x, y, a, b)  area = 0.5\*(x(b)-x(a))\*(y(a)+y(b));    end | Simple trapezoid function to calculate the area between two points |
| **FindPoints Function**  function [point1, point2] = FindPoints (a, b, x, y, up)  in\_a=0; in\_b=0;  yIndex=0;yMax=0; yMin=y(1);  minDiff=1;    % Index the points  for i=1:size(x)  if a==x(i)  in\_a=i;  end    if b==x(i)  in\_b=i;  end  end    % for the sparks going up  if (up==1)  for i=in\_a:1:in\_b  if i<in\_b  diff=abs(y(i+1)-y(i));  end  if diff<minDiff;  minDiff=diff;  point1=i;  end    if y(i)>yMax  yMax=y(i);  yIndex=i;  end  end    point2=((yIndex-point1)+yIndex);    % for the sparks going down  else  for i=in\_b:-1:in\_a    if i>in\_a  diff=abs(y(i+1)-y(i));  end    if diff<minDiff;  minDiff=diff;  point2=i;  end    if y(i)<yMin  yMin=y(i);  yIndex=i;  end  end    point1=yIndex-(point2-yIndex);    end    end | This function finds the optimal interval in which we should perform the Simpson function and the trapezoid function. The a and b values are just approximations of the intervals of the emission line. In other word, we have to first estimate where the intervals are going to be and then the function will help us find a better interval to perform our analysis.  I started this program by indexing the a and b, which was inputted into the function.  Function that finds the optimal interval in which we should integrate over  If the spectral line is pointing down, we use this code. This code has the same logic as the one above, but adjusted for spectral liens pointing downwards. |
| **FindLow Function**  function [yIndex] = FindLow (a, b, x, y)    in\_a=0; in\_b=0;  yIndex=0; yMin=y(1);    % Index the points  for i=1:size(x)  if a==x(i)  in\_a=i;  end    if b==x(i)  in\_b=i;  end  end  for i=in\_a:1:in\_b    if y(i)<yMin  yMin=y(i);  yIndex=i;  end  end  end | Index the inputted values  Function to find the lowest point in a given interval |
| **Main**  % Sparks going up or down  [A1, A2]= FindPoints(x(754), x(1021), x, y, 1);  [C1, C2]= FindPoints(x(1552), x(1752), x, y, 1);  [F1, F2]= FindPoints(x(3500), x(3820), x, y, 1);    % Sparks going down (through eyeballing)  [B1, B2]= FindPoints(x(1250), x(1550), x, y, 0);  [D1, E2]= FindPoints(x(2100), x(2550), x, y, 0);  [D2] = FindLow(x(D1), x(E2), x, y);  E1=2379;    % The intervals we should integrate over along with the points  plot(x, y);  hold on;  scatter(x(A1), y(A1));  scatter(x(A2), y(A2));  scatter(x(C1), y(C1));  scatter(x(C2), y(C2));  scatter(x(F1), y(F1));  scatter(x(F2), y(F2));    scatter(x(B1), y(B1));  scatter(x(B2), y(B2));    scatter(x(D1), y(D1));  scatter(x(D2), y(D2));  scatter(x(E1), y(E1));  scatter(x(E2), y(E2));    % Apply Simpsons over intervals  A\_simp=Simpson(x, y, A1, A2);  B\_simp=Simpson(x, y, B1, B2);  C\_simp=Simpson(x, y, C1, C2);  F\_simp=Simpson(x, y, F1, F2);    D\_simp=2\*Simpson(x, y, D1, D2);  E\_simp=2\*Simpson(x, y, E1, E2);    % Apply Trapezoid over intervals; we have to double dip one differently  A\_T=Trap(x, y, A1, A2);  B\_T=Trap(x, y, B1, B2);  C\_T=Trap(x, y, C1, C2);  F\_T=Trap(x, y, F1, F2)    % We use the point D1 and E2 to find the function of the line that goes  % between these two points, then we just integrate over the line  f = @(z) ((y(E2)-y(D1))/(x(E2)-x(D1))\*(z-x(D1))+y(D1));  D\_T= 2\*integral(f, x(D1), x(D2));  E\_T= 2\*integral(f, x(E1), x(E2));    % Get the differene bewteen the two areas  A\_area=abs(A\_simp-A\_T);  B\_area=abs(B\_simp-B\_T);  C\_area=abs(C\_simp-C\_T);  D\_area=abs(D\_simp-D\_T);  E\_area=abs(E\_simp-E\_T);  F\_area=abs(F\_simp-F\_T);  % Organize the data  Simpson\_Area=[A\_simp, B\_simp, C\_simp, D\_simp, E\_simp, F\_simp];  Trap\_Area=[A\_T, B\_T, C\_T, D\_T, E\_T, F\_T];  Em\_Intervals=[A\_area, B\_area, C\_area, D\_area, E\_area, F\_area];    % Tabelize the data  FinalTable = table(['A1-A2';'B1-B2';'C1-C2';'D1-D2';'E1-E2';'F1-F2'], transpose([Simpson\_Area]), transpose([Trap\_Area]), transpose([Em\_Intervals]));  FinalTable.Properties.VariableNames = {'Intervals' 'Simpson\_Area' 'Trapezoid\_Area' 'Emission\_Intensity'} | In this section, I used the FindPoints function to find the intervals for the difference emission lines. The first |

Works Cited

https://en.wikipedia.org/wiki/Simpson%27s\_rule