Yang Yu

August 06, 2018

Cs361_Final_pt1

ID: 23141168

X1=0.2314

X2=0.1168

Algorithm:

1. Calculate the number of intersection points of the two ellipses and Classification of the relative positions

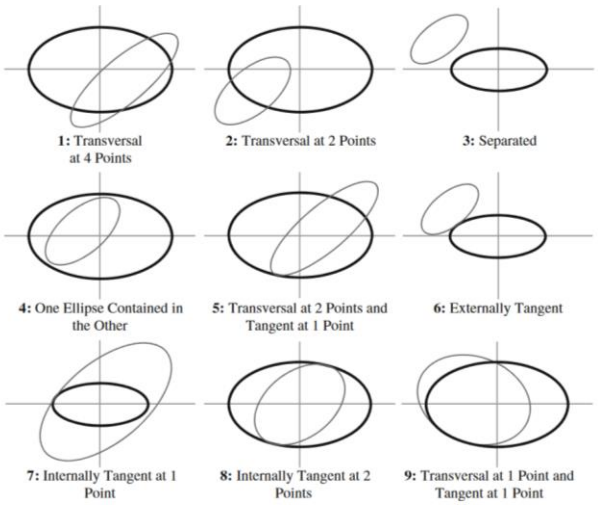
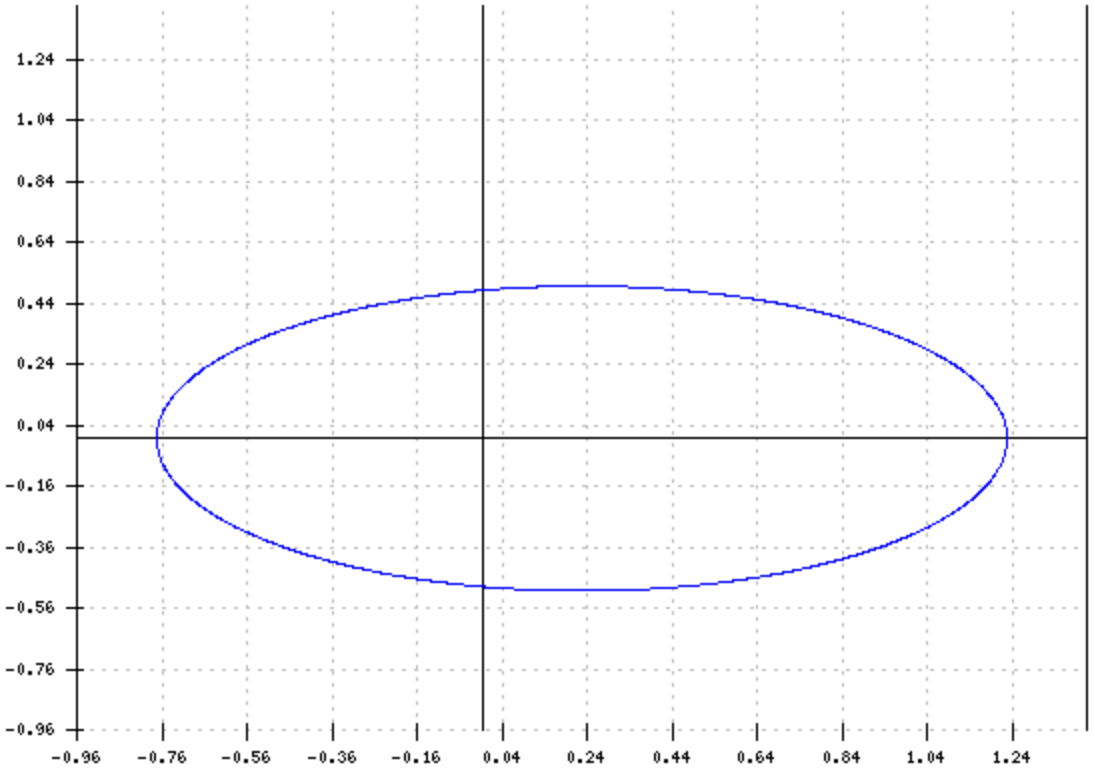



Image from: Hughes, Gary B., and Mohcine Chraibi. “Calculating Ellipse Overlap Areas.”

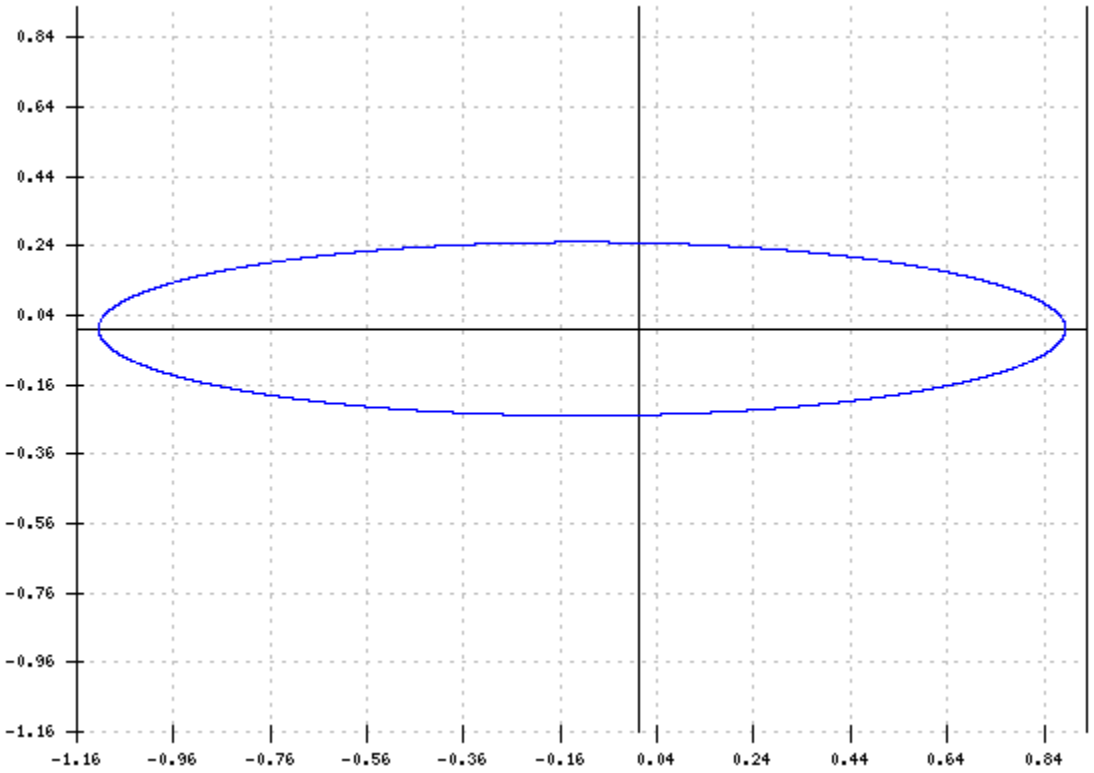
$(x-x1)^2+2y^2=1$ --a
$(x+x2)^2+4y^2=1$ --b

2*a-b
->$2(x-x1)^2-(x+x2)^2=1$
->>$2(x-0.2314)^2-(x+0.1168)^2=1$
→x^2 - 1.1592 x - 0.90655 = 0
→x≈-0.535069

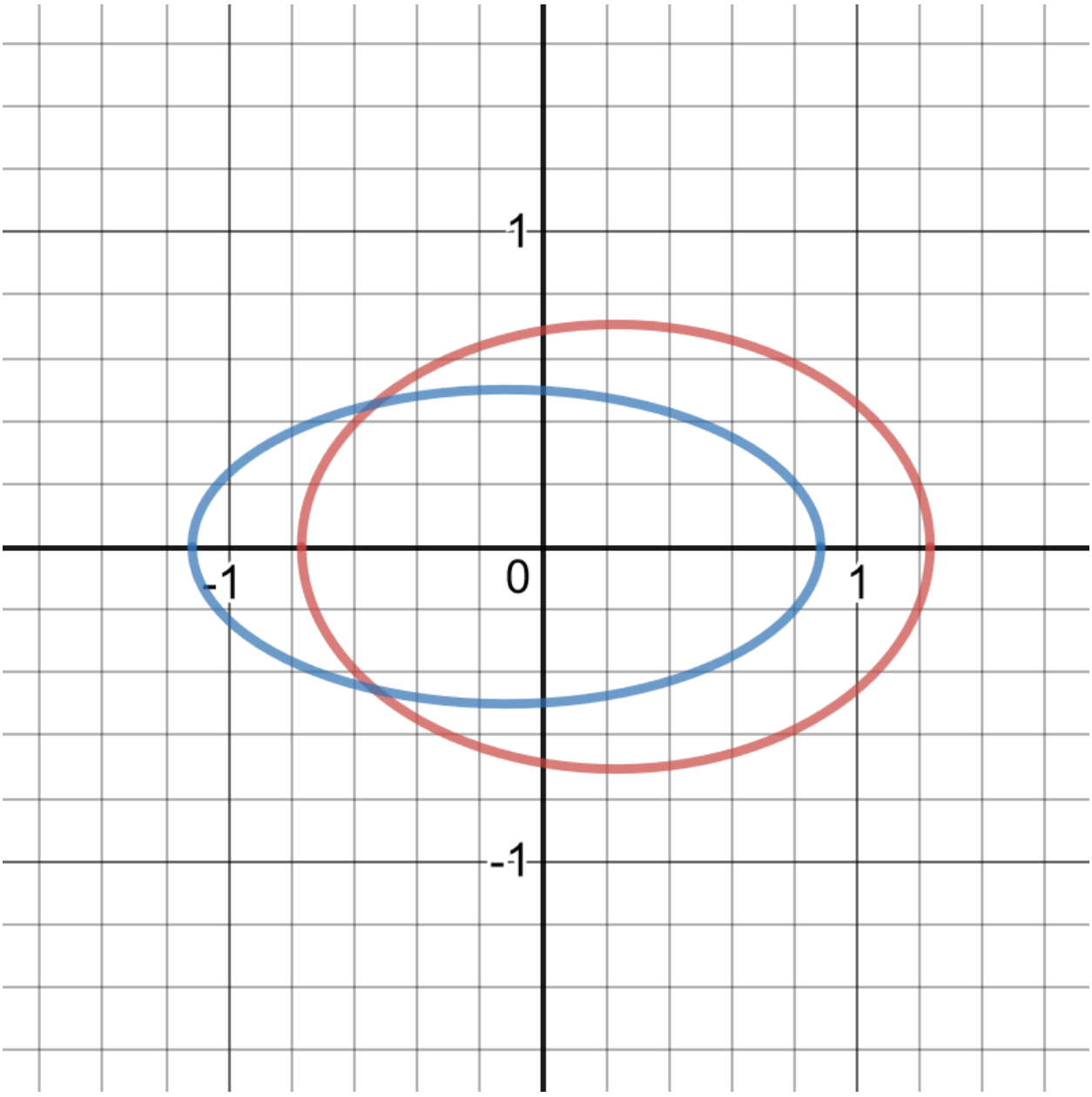
I use online plot to check the two ellipses’ position




$(x - x1)^2 + 2y^2 = 1$ --a




$(x + x2)^2 + 4y^2 = 1$ --b



Transversal at 2 Points

Step2:

Use trapezoidal rule to calculate

$(x - 0.2314)^2 + 2y^2 = 1$ y=0→x=-0.7686 left bond of ellipse(red)

$(x + 0.1688)^2 + 4y^2 = 1$ y=0→x=0.8312 right bond of ellipse(blue)

x≈-0.535069(intersection point)

$y=\sqrt{\frac{1 \ -(x-0.2314)^2}{2}}$ [a,b]=[-0.7686,- -0.535069] –I1

y=$\sqrt{\frac{1\ -(x+0.1168)^2}{4}}$ [a,b]=[ -0.535069,0.8312]—I2

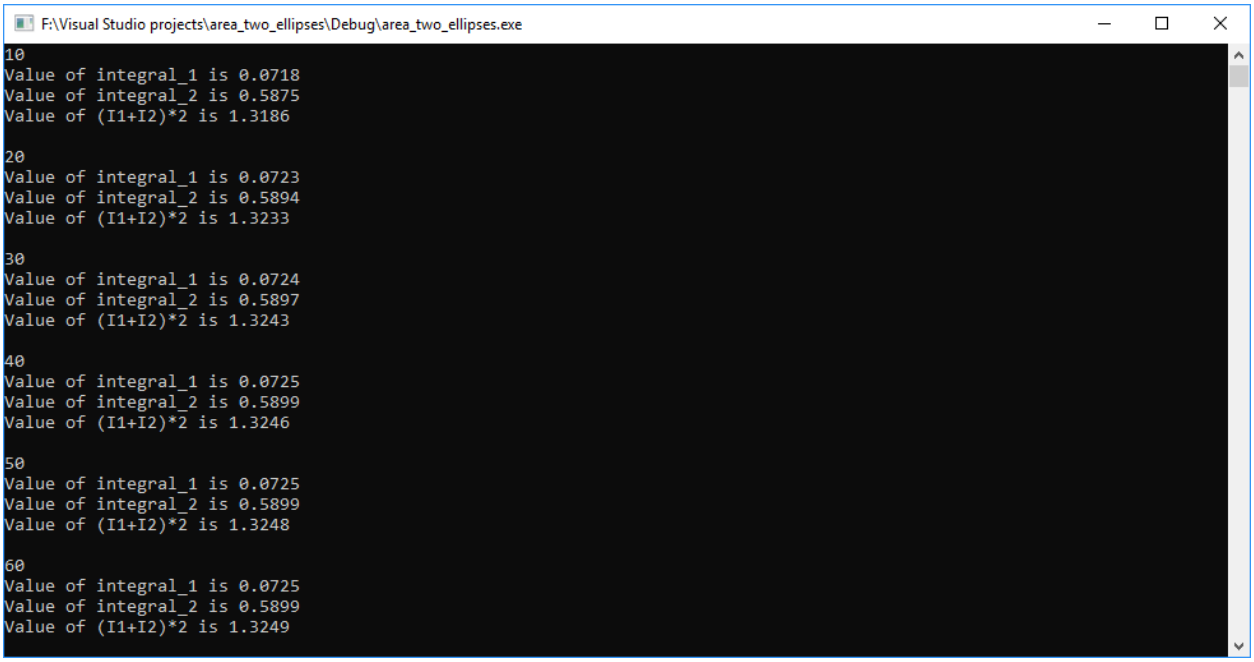
x-axis symmetry

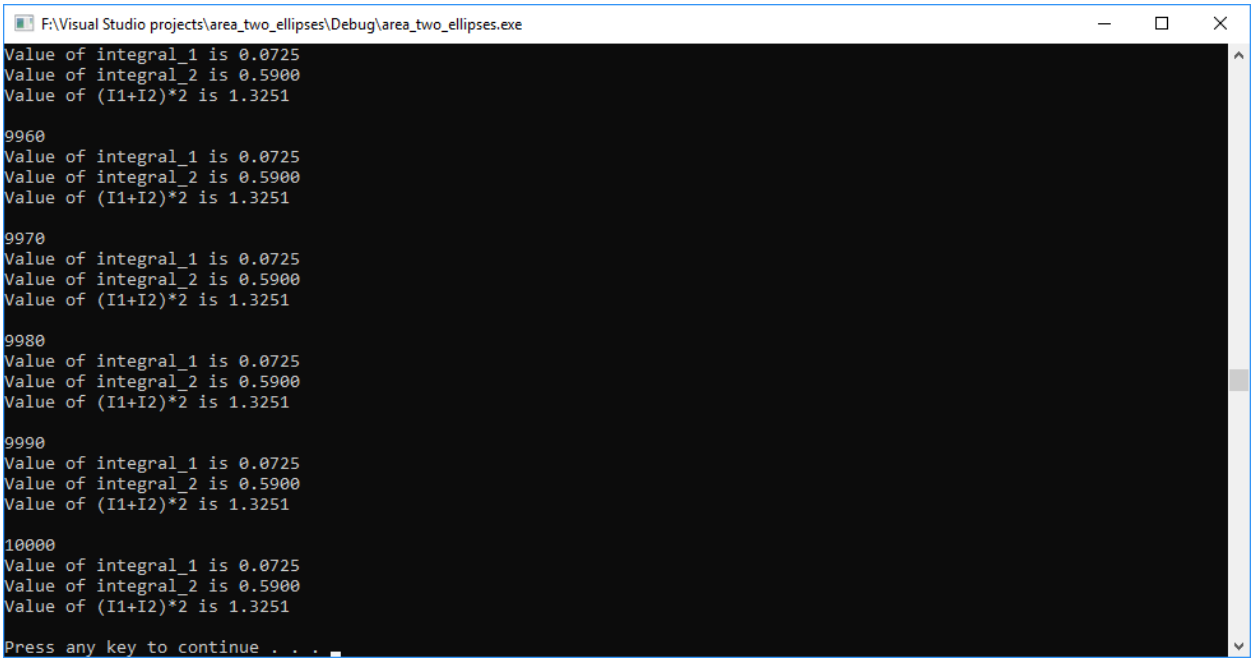
result= (I1+I2)*2

output:

| n | I1 | I2 | (I1+I2)*2 |
|---|---|---|---|
| 10 | 0.0718 | 0.5875 | 1.3186 |
| 20 | 0.0723 | 0.5894 | 1.3233 |
| 30 | 0.0724 | 0.5879 | 1.3243 |
| 40 | 0.0725 | 0.5899 | 1.3246 |
| 80 | 0.0725 | 0.5900 | 1.3250 |
| 160 | 0.0725 | 0.5900 | 1.3251 |
| 10000 | 0.0725 | 0.5900 | 1.3251 |

According to the form we can see when n >=160 the result will not change.






```
1.  // area_two_ellipses.cpp : Defines the entry point for the console application. // //Calculate
     the area enclosed between the two ellipses, to 4 decimal places // C++ program to implement T
    rapezoidal rule
2.  #include "stdafx.h"
3.  #include < stdio.h > #include < cmath > #include < iostream > #include < string > using namesp
    ace std; // A function whose definite integral's // approximate value is computed using Trapez
    oidal rule
4.  float x1 = 0.2314; //first four digits of id
5.  float x2 = 0.1168; //last four digits of id //
6.  float y1(float x) { // Declaring the function f(x) = y=√((1 -(x-0.2314)^2)/2)
7.      return sqrt((1 - pow((x - x1), 2)) / 2);
8.  }
9.  float y2(float x) { // Declaring the function f(x) = y=√((1 -(x+0.1168)^2)/2)
10.         return sqrt((1 - pow((x + x2), 2))  / 4);
11.     } // Function to evalute the value of integral
12. float trapezoidal_1(float a, float b, float n) { // Grid spacing
13.     float h = (b - a) / n; // Computing sum of first and last terms // in above formula
14.     float s = y1(a) + y1(b); // Adding middle terms in above formula
15.     for (int i = 1; i < n; i++) s += 2 * y1(a + i * h); // h/2 indicates (b-
    a)/2n. Multiplying h/2 // with s.
16.     return (h / 2) * s;
17. }
18. float trapezoidal_2(float a, float b, float n) { // Grid spacing
19.         float h = (b - a) / n; // Computing sum of first and last terms // in above formula
20.         float s = y2(a) + y2(b); // Adding middle terms in above formula
21.         for (int i = 1; i < n; i++) s += 2 * y2(a + i * h); // h/2 indicates (b-
    a)/2n. Multiplying h/2 // with s.
22.         return (h / 2) * s;
23.     } // Driver program to test above function
24. int main() {
25.     float x1 = 0.2314; //first four digits of id
26.     float x2 = 0.1168; //last four digits of id // Range of definite integral
27.     float a1 = -0.7686; //[-0.7686, -0.535069]
```

```
    float b1 = -0.535069;
    float a2 = -0.535069; //[ -0.535069,0.8312]
    float b2 = 0.8312; // Number of grids. Higher value means // more accuracy
    int n = 6;
    for (int i = 10; i <= 1e4; i += 10) {
        cout << i << endl;
        printf("Value of integral_1 is %6.4f\n", trapezoidal_1(a1, b1, i));
        printf("Value of integral_2 is %6.4f\n", trapezoidal_2(a2, b2, i));
        printf("Value of (I1+I2)*2 is %6.4f\n", (trapezoidal_1(a1, b1, i) + trapezoidal_2(a2,
b2, i)) * 2);
        printf("\n");
    }
    system("pause");
    return 0;
}
```