110770716

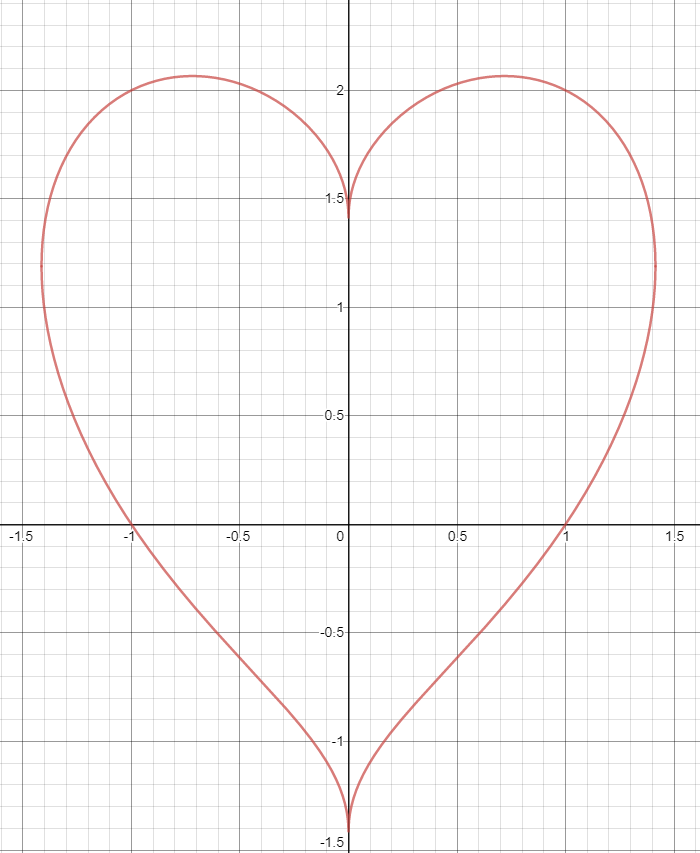
Darren Kong

AMS326

Homework 1

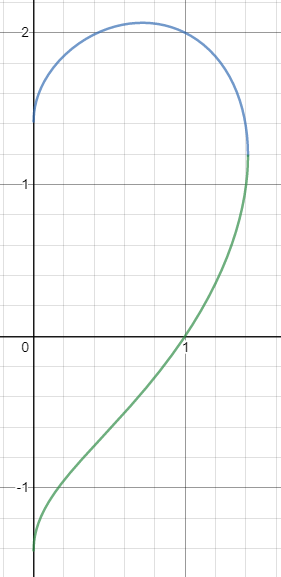
**Problem 1.1**

The heart equation, is graphed below.



The problem states to find the area of the remaining heart after a max area circle has been cut from it. First, I had to find the upper and lower sections of the heart equations first. I was able to derive the upper heart equation

and the lower heart equation



The blue and green lines represent the two equation respectively.

To find the approximation of the area of the heart numerically, I utilized the trapezoidal rule. The code is linked here “<https://github.com/kong0716/AMS326HW/blob/master/hw1pt1.py>”. One can download the file as a python file and run it by using “python ./hw1pt1.py” in their terminal.

The f\_(x) and h\_(x) function represented the equations above. The heartArea(n) function is split into 3 parts. The first two parts were the area under the curve of the functions with the restriction that the function stayed above the x-axis. The third part was the area of the curve underneath the x-axis. The parameter “n” was the “height” of each trapezoid and thus the smaller the width, the more accurate the result. By summing up the small trapezoids together, I am able to get an approximation of the area under each line curve.

By utilizing simple mathematical manipulations, I was able to get the area of the half heart above and multiply it by 2 to get the total area of the heart to be 6.2831…, basically converging to 2π.

To find the max area of the circle, I had to alter the equation of the circle to fit my needs.

Became

Since we know that the value of the trough of the heart is , the max area circle must intersect it. We simply rewrote the center of the circle in terms of the radius. Furthermore, after some algebra, we got

Since we needed the circle to intersect the lower portion of the heart.

The bottomCircle(x,r) was simply the equation above. The intersectionTest(x1, x2, r, increments, epsilon) in O(n) time found the intersection of the circle and the heart given the range of the values, the radius of the circle, the increments to increase by, and the epsilon margin of error. It returned the amount of times, there were intersections.

The findmaxCircleRadius(minr, maxr, increments, epsilon) was simply a binary search for the optimal radius of the circle. The minr and maxr represent the minimum and maximum range of values for the radius. The remaining parameters are the same as explained above. I received the answers below with increment and epsilon set to .

I had taken a cheap way of calculating the number of floating point operation by simply iterating a global variable whenever a calculation was performed. I got the number of operations to be 36,143,397. This did not count the calling of math.sqrt or math.pow.

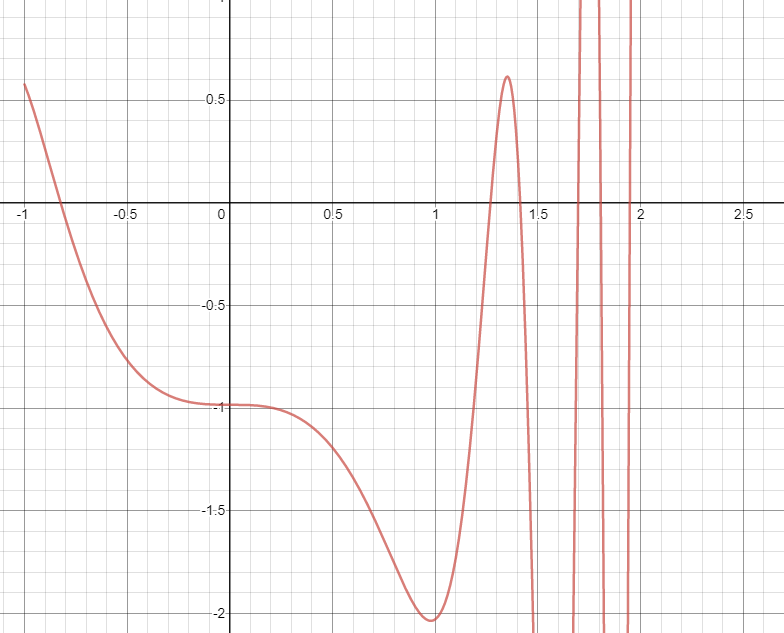
Area of the heart is 6.283185262968954

Area of the circle is 3.3508201643959663

Area of the remaining heart is 2.9323650985729874

**Problem 1.2**

The equation is graphed below.



The code is linked here “<https://github.com/kong0716/AMS326HW/blob/master/hw1pt1.py>”. One can download the file as a python file and run it by using “python ./hw1pt2.py” in their terminal.

The function f(x) is simply the representation of the above equation. The bisection\_method(a,b, epsilon) takes in the bounding range “a” and “b” of where the root may be and calculates it within the error epsilon. The algorithm works due to the fact that for the roots above, f(a) and f(b) are of opposite signs and will remain opposite signs while slowly converging to f(x). The replacement values of “a” and “b” is always “x” which will be half of the previous sum of “a” and “b”. The ranges were found by “eyeballing the graphical representation of the equation. The results are below with epsilon .

Root 1 is -0.8242922973632814

Root 2 is 1.2691925048828123

Root 3 is 1.4142974853515624

Root 4 is 1.6955993652343753

Root 5 is 1.806716918945312

Root 6 is 1.9482727050781246