William Daniels AMS 326 exam one writeup

February 21, 2023

All of my source files can be found in this github repositiory:
https://github.com/William-J-Daniels/DanielsAms326.git

A detailed README on how the repositiory is used is there. The commit that represents the state of the repository upon submission of the assignment is the one before 6:08 on 2/21/23. (I can't give the hash before I commit).

Description

This problem is to find the root of the given function the intersection of the heart and the circle using three methods:

- Bisection,
- Newton-Raphson, and
- Secant.

The intersection cant be found as-is, we first had to do some manipulation to get a useful form:

$$x^2+(y-\sqrt{(|(x)|)})^2=2=x^2+(y-2)^2$$

We only need to consider one side due to symmetry, so abs(x) → x.

$$y^2-2\sqrt{(x)}y+x=y^2-4y+4$$

$$y(4-2\sqrt{(x)})=4-x$$

$$y=\frac{4-x}{4-2\sqrt{(x)}}$$

We multiply by 4 + 2sqrt(x) over itself to fix the deminiatior and find

$$y=\frac{16-8x}{16+4x}$$

Which is what we find the root of.

Algorithms

Before giving my implimentations for each algorithm, I should first describe how I implimented them. I used an object oriented approach in which each root finding algoritm is implimented in a class which inherits from an abstract class RootFinder. It's implimentation is given in `RootFinding/include/RootFinder.h` and `RootFinding/src/RootFinding.cpp`.

Bisection

Given initiail bounds a and b such that $f(a)f(b)<0$
while (b – a)/2 > desired precision
    $c=(a+b)/2$
    if $f(c)=0$ **break**
    if $f(a)f(b)<0$
        b=c
    else
        a=c
The approximate root is (a+b)/2

Listing 1: Bisection psuedocode

The implimentation is in `RootFinding/include/Bisection.h` and `RootFinding/src/Bisection.cpp`.

Newton-Raphson

Given an itital guess of the root x0
while consecutive iterations adjust the estimate by an amount less than the desired precision

$$x_i = x_{i-1} - \frac{f(x_{i-1})}{f'(x_{i-1})}$$

Listing 2: Newton-Raphson psudo code

The implimentation can be found in `RoodFinding/include/NewtonRaphson.h` and `RoodFinding/src/NewtonRaphson.cpp`. My implimentation also contains a method that uses a second order two point approximation of the derivative, though only the analytic derivative is used for the assignment.

Secant

Given initial guesses a and b that are to the left and right of the root respectively
while consecutive iterations adjust the estimate by an amount less than the desired precision

$$x_i = x_{i-1} - \frac{f(x_{i-1})\, f(x_{i-2})}{f(x_{i-1}) - f(x_{i-2})}$$

Listing 3: Secant Psudocode

This is the a modification of the Newton method using a first order forward approximation of the dirivative.

Given an intial guess x_0
while consecutive iterations adjust the estimate by an amount less than the desired precision

$$x_i = f(x_i)$$

Listing 4: Fixed-point psudocode

Results

All three methods converged to 2.0. As per the parameters in `RootFinding/include/RootFinder.h`, all of these solution are precice to 10^{-12}. The dirver code for these solutions is in `ExamOne/examples/ProblemTwo.cpp`.