Experiment-2
Root Finding

(Duration: 90 mins)

Purpose: The aim of this lab is to apply different numerical root finding methods to equations of two or more orders.

Introduction

The root-finding problem is an important computational problem. It arises in a wide variety of practical applications in physics, chemistry, biosciences, engineering,etc. Determination of any unknown appearing implicitly in scientific or engineering formulas gives rise to a root-finding problem. Most of the numerical root finding methods require an interval that contain a root as their argument. Finding an appropriate interval is necessary for the application of these methods.

In this lab, you are asked to write a program that finds intervals that contain roots of three degree polynomial.

Problem Statement

This week we will look at and compare two different root finding methods, interval and bisection. The coefficients of the function, the interval, step size and maximum number of iterations will be taken from the user. The interval method scans by moving up the step size until it finds the root, while the bisection method divides the interval in half and scans for N iterations until it finds the root. After N iterations, when the interval is small enough to find the root, it terminates the search.

To find such an interval containing the roots of a function, consider an interval of width [a,b] and divide it into subintervals. Then discuss which method is more efficient in finding roots.

Lab Procedure

- Enter Polynomial coefficients, interval to search for root, step size for interval method, max iteration number of Bisection method. (**10 pt**)

$$f(x) = x^3 - 3x^2 - 2x - 1 \tag{1}$$

- Write a function that computes the 3d degree polynomial function for a given value. (**20 pt**)

```
// Function Prototype for Polynomial Function
float polysum(int order, //Polynomial order\\
float array[MAXSIZE],// Polynomial coefficients\\
float x); // Evaluation value
```

- Write a program that

  a) finds the root using the interval method which checks each small step for a root and stops when it finds the root, printing the estimated root value and the number of iterations (**30 pt**)

b) finds the root using the Bisection Method which divides the full interval into two equal parts and checks if one of them has a root, and keeps dividing the interval containing root until convergence tolerance or max iteration number is reached, finally printing the estimated root and the number of iterations (number of times interval is divided to two parts) (**40 pt**)

Program should look like this:

```
Enter the order of the polynomial
3
Enter 4 coefficients
1 -3 -2 -1
Given polynomial is:
+1.00*x^3-3.00*x^2-2.00*x^1-1.00 = 0
Input interval values (x1,x2), step_size and maximum iterations:
0
5
0.001
100


f(3.599999)=-0.424021, f(3.699999)=1.182976, f(3.649999)=0.359603
f(3.599999)=-0.424021, f(3.649999)=0.359603, f(3.624999)=-0.037132
f(3.624999)=-0.037132, f(3.649999)=0.359603, f(3.637499)=0.159999
f(3.624999)=-0.037132, f(3.637499)=0.159999, f(3.631248)=0.061122
f(3.624999)=-0.037132, f(3.631248)=0.061122, f(3.628124)=0.011919
f(3.624999)=-0.037132, f(3.628124)=0.011919, f(3.626561)=-0.012625
f(3.626561)=-0.012625, f(3.628124)=0.011919, f(3.627342)=-0.000358
f(3.627342)=-0.000358, f(3.628124)=0.011919, f(3.627733)=0.005776
f(3.627342)=-0.000358, f(3.627733)=0.005776, f(3.627537)=0.002707
f(3.627342)=-0.000358, f(3.627537)=0.002707, f(3.627440)=0.001176
f(3.627342)=-0.000358, f(3.627440)=0.001176, f(3.627391)=0.000407
f(3.627342)=-0.000358, f(3.627391)=0.000407, f(3.627367)=0.000022
Iteration of bisection method = 11
Iteration of interval method = 3627
Root = 3.627367
```