Lab 7 Report

Bisection method

**Assumptions:**

f(x) is a continuous function in interval [a, b]

1. f(a) * f(b) < 0

**Steps:**

1. Find middle point **c**= (a + b)/2 .
2. **If** f(c) == 0, then c is the root of the solution.
3. **Else** f(c) != 0
    1. **If** value f(a)*f(c) < 0 then root lies between a and c. So we recur for a and c
    2. **Else If** f(b)*f(c) < 0 then root lies between b and c. So we recur b and c.
    3. **Else** given function doesn't follow one of assumptions.

Newton method

**Algorithm:**

Input: initial x, func(x), derivFunc(x)

Output: Root of Func()

1. Compute values of func(x) and derivFunc(x) for given initial x
2. Compute h: h = func(x) / derivFunc(x)
3. While h is greater than allowed error ε
    1. h = func(x) / derivFunc(x)
    2. x = x – h

Secant method

The secant method is derived from the newton formula by replacing $f'(x_k)$with

$X_{k+1}= x_k-f(x_k)*( x_k- x_{k-1})/(f(x_k)-f(x_{k-1}))$.

In the lab seven we proved :

Advantage

Bisection method: it works good with smaller intervals and when f(b)*f(a)<0.

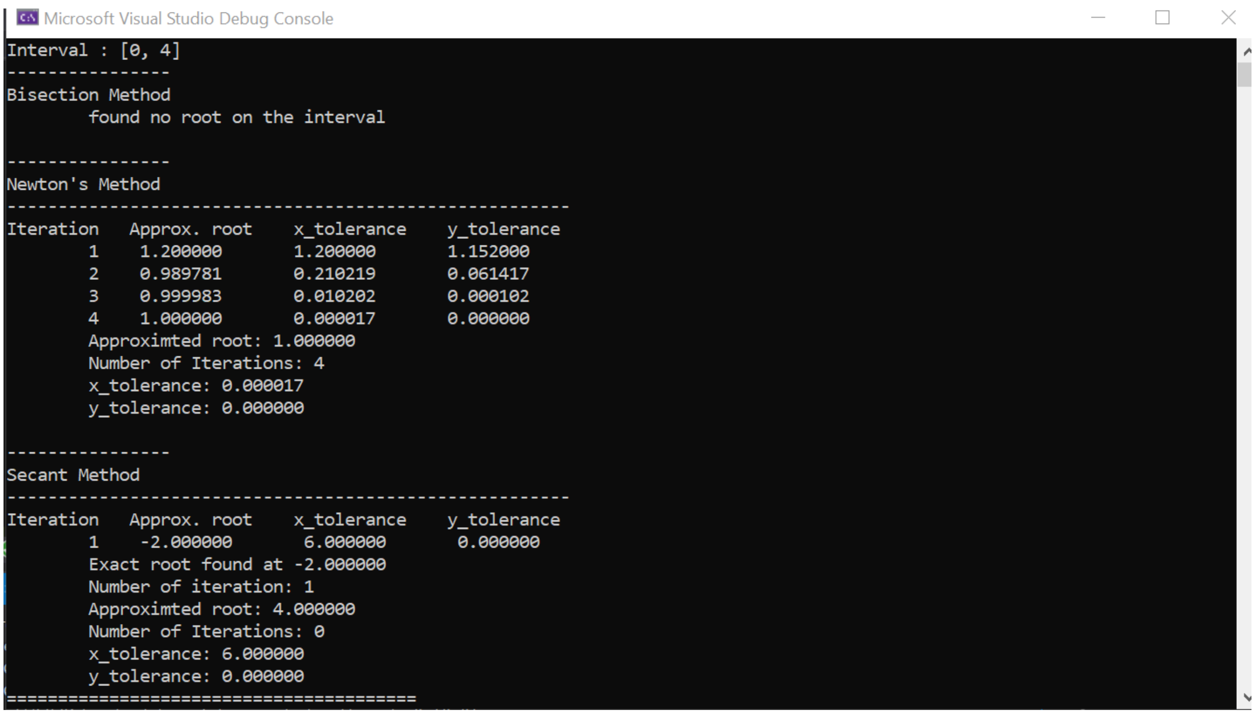
Newton's method: it works faster than the bisection method.

Secant method: faster than the above two and do not need to calculate first derivative.

Disadvantage

1. Bisection method is slower than the two other methods. It also does not work when f(b)*f(a)>0.
2. Newton’s method is complicated when we try to calculate the first derivative and it fails if the derivative is evaluated at 0.
3. Secant method is less complicated than Bisection and Newton methods.

What I prefer: secant method because it is faster and does not need the calculation of first derivative.



```
Microsoft Visual Studio Debug Console
Interval : [0, 4]
---------------
Bisection Method
        found no root on the interval

---------------
Newton's Method
------------------------------------------------------
Iteration   Approx. root    x_tolerance    y_tolerance
        1    1.200000       1.200000       1.152000
        2    0.989781       0.210219       0.061417
        3    0.999983       0.010202       0.000102
        4    1.000000       0.000017       0.000000
        Approximted root: 1.000000
        Number of Iterations: 4
        x_tolerance: 0.000017
        y_tolerance: 0.000000

---------------
Secant Method
------------------------------------------------------
Iteration   Approx. root    x_tolerance    y_tolerance
        1    -2.000000       6.000000       0.000000
        Exact root found at -2.000000
        Number of iteration: 1
        Approximted root: 4.000000
        Number of Iterations: 0
        x_tolerance: 6.000000
        y_tolerance: 0.000000
========================================
```