Assignment #7:
Root finding methods

Timmy Nguyen

March 21, 2017

1 Overview

Given a function: $y = x2$ where y = 7. Solve via Newton Method.

$x_{n+1} = x_1 - \frac{f(x_1)}{f'(x_1)}$

2 Code Analysis

First, I declared my variables where 'e' represents the tolerance.

```
1 double x1,x2;
2 double e = 0.00001;
```

Then, create a c function which initiate f(x) as well as another function to return a derivative.

```
1
2 double f(double x)
3 {
4   return x * x - 7;
5 }
6
7
8 double fderiv(double x)
9 {
10   double h = 0.0001;
11
12   // Difference Quotient
13   double diff_q =  ( f(x+h) - f(x-h) ) / (2 * h);
14   return diff_q;
15 }
```

After settings the value for x1, the code is ready to follow Newton Method. Apply the method in a loop until the tolerance(+- 0.00001) is met.

```
1
2   for (int i = 0; i < steps; ++i)
3   {
4     x2 = x1 - f(x1) / fderiv(x1);
5
6     if (fabs(x2-x1) < e)
7     {
8       break;
9     }
10
11    printf("Results are %f\n", x2);
12
13    x1 = x2;
14
15
16  }
```

3 Program Output

```
1 Results are 2.666667
2 Results are 2.645833
3 Results are 2.645751
```

The root obtain for function $x^2 - 7$ via Newton is represented on line 3, "Results are 2.645751"

4 C Code

```c
/**
 * Name: Timmy Nguyen
 * Lab Exercise 7: Newton Method
 * BIO 480 Spring 2017
 * Date: 3-21-17
 */

#include <stdio.h>
#include <math.h>

// Global Constants
double x1,x2;
double e = 0.00001;

// Create function f(x)
/**
 *
 * Y = x^2 ; y = 7
 * x^2 - 7 = 0
 *
 */
double f(double x)
{
  return x * x - 7;
}

// Finds the derivative of f(x)
double fderiv(double x)
{
  double h = 0.0001;

  // Difference Quotient
  double diff_q =  ( f(x+h) - f(x-h) ) / (2 * h);
  return diff_q;
}


int main(int argc, char const *argv[])
{
  // Prompt User for initial x1 value
  printf("Set initial guess: \n");
  scanf("%lf", &x1);

```

```
44    /**
45     * Iteratives until tolerance += 0.00001 is met
46     * - maximum 100 steps
47     */
48    int steps = 100;
49
50    for (int i = 0; i < steps; ++i)
51    {
52      x2 = x1 - f(x1) / fderiv(x1);
53
54      // When tolerance condition is met, end the loop
55      if (fabs(x2-x1) < e)
56      {
57        break;
58      }
59
60      printf("Results are %f\n", x2);
61
62      x1 = x2;
63
64
65    }
66
67    return 0;
68  }
```