MAT362 - Project0 Report

Christopher D. Whitney

August 29, 2017

1 Introduction

In the following report we shall outline and describe the algorithms used, their relative theory and the techniques used to implement them in Matlab. The problems this project deals with is the numerical approximate of integrals, solving initial value problems, first and second derivatives and finding zeros values.

2 Midpoint Rule

3 Euler's Method

4 Newtons Method

Newtons method uses equation of the tangent line to find when a given equation equals zero. To implement this method we used two different approach, the first is a sequential approach where a while loop is used to recalculate X until the stop criteria is met, the second approach uses a recursive function to calculate and re-calculate x until a base-case (stopping criteria) is met. Though the recursive solution is more allegiant does have some serious draw backs such as reaching maximum recursive depth.

To execute this analysis run the file $run_nm.m$, this will call the two different approach $np_seq.m$ (sequential approach) and $nm_rec.m$ (recursive approach). The following are the results after running the script.

```
--- Project0 Netwon Methods ---
 Where,
 f(x) = x^2 ,  g(x) = x^4
 f - g = x^2 - x^4
 fp = 2x - 4x^3
 Tollerance thershold = 1e-10
-- Guess 1 Method 1 (Sequential) --
 x0 = 2
chi=0.42857 xn = 1.5714
chi=0.29312 xn = 1.2783
```

```
chi=0.17868 xn = 1.0996
chi=0.081089 xn = 1.0185
chi=0.017733 xn = 1.0008
chi=0.00080692 xn = 1
chi=1.6299e-06 xn = 1
chi=6.6414e-12 xn = 1
 Result = 1
-- Guess 2 Method 1 (Sequential) --
 x0 = 1/4
chi=0.13393 xn = 0.11607
chi=0.058839 xn = 0.057232
chi=0.02871 xn = 0.028522
chi=0.014272 xn = 0.014249
chi=0.0071261 xn = 0.0071232
chi=0.0035618 xn = 0.0035614
chi=0.0017807 xn = 0.0017807
chi=0.00089034 xn = 0.00089034
chi=0.00044517 xn = 0.00044517
chi=0.00022258 xn = 0.00022258
chi=0.00011129 xn = 0.00011129
chi=5.5646e-05 xn = 5.5646e-05
chi=2.7823e-05 xn = 2.7823e-05
chi=1.3912e-05 xn = 1.3912e-05
chi=6.9558e-06 xn = 6.9558e-06
chi=3.4779e-06 xn = 3.4779e-06
chi=1.7389e-06 xn = 1.7389e-06
chi=8.6947e-07 xn = 8.6947e-07
chi=4.3473e-07 xn = 4.3473e-07
chi=2.1737e-07 xn = 2.1737e-07
chi=1.0868e-07 xn = 1.0868e-07
chi=5.4342e-08 xn = 5.4342e-08
chi=2.7171e-08 xn = 2.7171e-08
chi=1.3585e-08 xn = 1.3585e-08
chi=6.7927e-09 xn = 6.7927e-09
chi=3.3964e-09 xn = 3.3964e-09
chi=1.6982e-09 xn = 1.6982e-09
chi=8.4909e-10 xn = 8.4909e-10
chi=4.2455e-10 xn = 4.2455e-10
chi=2.1227e-10 xn = 2.1227e-10
chi=1.0614e-10 xn = 1.0614e-10
chi=5.3068e-11 xn = 5.3068e-11
 Result = 5.3068e-11
-- Guess 1 Method 2 (Recurive) --
 x0 = 2
chi=0.42857 xn = 1.5714
chi=0.29312 xn = 1.2783
```

```
chi=0.17868  xn = 1.0996
chi=0.081089  xn = 1.0185
chi=0.017733  xn = 1.0008
chi=0.00080692  xn = 1
chi=1.6299e-06  xn = 1
chi=6.6414e-12  xn = 1
 Result = 1.5714
-- Guess 2 Method 1 (Recurive) --
 x0 = 1/4
chi=0.13393  xn = 0.11607
chi=0.058839  xn = 0.057232
chi=0.02871  xn = 0.028522
chi=0.014272  xn = 0.014249
chi=0.0071261  xn = 0.0071232
chi=0.0035618  xn = 0.0035614
chi=0.0017807  xn = 0.0017807
chi=0.00089034  xn = 0.00089034
chi=0.00044517  xn = 0.00044517
chi=0.00022258  xn = 0.00022258
chi=0.00011129  xn = 0.00011129
chi=5.5646e-05  xn = 5.5646e-05
chi=2.7823e-05  xn = 2.7823e-05
chi=1.3912e-05  xn = 1.3912e-05
chi=6.9558e-06  xn = 6.9558e-06
chi=3.4779e-06  xn = 3.4779e-06
chi=1.7389e-06  xn = 1.7389e-06
chi=8.6947e-07  xn = 8.6947e-07
chi=4.3473e-07  xn = 4.3473e-07
chi=2.1737e-07  xn = 2.1737e-07
chi=1.0868e-07  xn = 1.0868e-07
chi=5.4342e-08  xn = 5.4342e-08
chi=2.7171e-08  xn = 2.7171e-08
chi=1.3585e-08  xn = 1.3585e-08
chi=6.7927e-09  xn = 6.7927e-09
chi=3.3964e-09  xn = 3.3964e-09
chi=1.6982e-09  xn = 1.6982e-09
chi=8.4909e-10  xn = 8.4909e-10
chi=4.2455e-10  xn = 4.2455e-10
chi=2.1227e-10  xn = 2.1227e-10
chi=1.0614e-10  xn = 1.0614e-10
chi=5.3068e-11  xn = 5.3068e-11
 Result = 0.11607
```

It is easy to see that in both approaches that $x_0 = 2$ coverages quicker.

5 First and Second Derivative