Math 342: Project 1

Connor Emmons

Documentation: I used ChatGPT solely for looking up Latex commands. The main Project 1 MatLab script and all required dependencies are located in the Project 1 folder found here: <https://github.com/Connor-Lemons/Emmons-Math-342>. No other resources used.

Project 1:

Part 1: Aitken’s delta-squared process

Problem 1: Consider the problem of approximating the root of using Fixed-Point iteration.

Part A: Show that the root of is also a fixed point of the function .

Begin with the function . Finding the root is equivalent to finding the value of which satisfies the equation:

(1)

Rewriting equation (1) gives:

(2)

(3)

Defining gives:

(4)

Finding the value of which satisfies is equivalent to finding the fixed point of . Because this is the same value of that gives , the root of is the same as the fixed point of .

Part B: Use the Fixed-Point Theorem to show that Fixed-Point Iteration with will converge to for any in the interval .

Theorem 2.4 states that if (i.e., for all ), exists on , and there exists some constant such that , then for any the sequence converges to a unique fixed point .

Because is a decreasing monotone function (i.e., ), if and , then .

(1)

(2)

(3)

(4)

Thus, both and are in the interval .

Consider and its derivative . Both of these functions are defined for all , and are continuous functions for real number inputs. Notably, this means that exists on the interval because . Because is a monotone increasing function, will have the largest value on the interval at . This gives:

(1)

Thus, satisfies the condition

(2)

because

(3)

Therefore, because has been shown to satisfy all the conditions of Theorem 2.4 on the interval , where and , will converge to a unique fixed point for any .

Part C: Write a MatLab script to implement Fixed-Point Iteration with and . Use as the stopping criterion.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 1 | 0.67032 | 0.27032 |
| 2 | 0.51154 | 0.15878 |
| 3 | 0.59957 | 0.088024 |
| 4 | 0.54905 | 0.05052 |
| 5 | 0.5775 | 0.028451 |
| 6 | 0.5613 | 0.016199 |
| 7 | 0.57047 | 0.0091664 |
| 8 | 0.56526 | 0.0052052 |
| 9 | 0.56821 | 0.00295 |
| 10 | 0.56654 | 0.0016737 |
| 11 | 0.56749 | 0.00094903 |
| 12 | 0.56695 | 0.00053831 |
| 13 | 0.56725 | 0.00030528 |
| 14 | 0.56708 | 0.00017314 |
| 15 | 0.56718 | 9.8194e-05 |
| 16 | 0.56712 | 5.5691e-05 |
| 17 | 0.56715 | 3.1584e-05 |
| 18 | 0.56714 | 1.7913e-05 |
| 19 | 0.56715 | 1.0159e-05 |
| 20 | 0.56714 | 5.7617e-06 |
| 21 | 0.56714 | 3.2677e-06 |
| 22 | 0.56714 | 1.8533e-06 |
| 23 | 0.56714 | 1.0511e-06 |
| 24 | 0.56714 | 5.9611e-07 |

Code can be found in Appendix A.

Part D: Show that the solution obtained using Fixed-Point Iteration converges linearly to by showing that

(0)

for the 4 largest values of .

For the iteration above, use .

For :

(1)

For :

(2)

For

(3)

Note that this quotient equals zero for because the value of is taken to be the value of , which was the first that satisfied the error requirement. It is difficult to determine from these points alone whether holds. Calculating this quotient for all gives a more accurate picture. When accounting for all , it is clearer that the quotient is approximately constant, and thus this iteration converges linearly.

|  |  |
| --- | --- |
|  |  |
| 1 | 0.53886 |
| 2 | 0.58321 |
| 3 | 0.55804 |
| 4 | 0.57232 |
| 5 | 0.56418 |
| 6 | 0.56886 |
| 7 | 0.5661 |
| 8 | 0.56786 |
| 9 | 0.56652 |
| 10 | 0.56787 |
| 11 | 0.56606 |
| 12 | 0.56894 |
| 13 | 0.56406 |
| 14 | 0.57258 |
| 15 | 0.55768 |
| 16 | 0.58411 |
| 17 | 0.53811 |
| 18 | 0.62109 |
| 19 | 0.48028 |
| 20 | 0.74799 |
| 21 | 0.32537 |
| 22 | 1.3102 |
| 23 | 0 |

Code can be found in Appendix A.

Problem 2: Consider the problem of approximating the root of using Steffensen’s Method.

Part A: Write a MatLab script to implement Steffensen’s Method with and . Use as the stopping criterion.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 1 | 0.5703 | 0.1703 |
| 2 | 0.56714 | 0.003151 |
| 3 | 0.56714 | 1.0178e-06 |
| 4 | 0.56714 | 1.0633e-13 |

Code can be found in Appendix A.

Part B: Show that the solution obtained using Fixed-Point Iteration converges linearly to by showing that

(0)

for the 4 largest values of .

For the iteration above, use .

For :

(1)

For :

(2)

For :

(3)

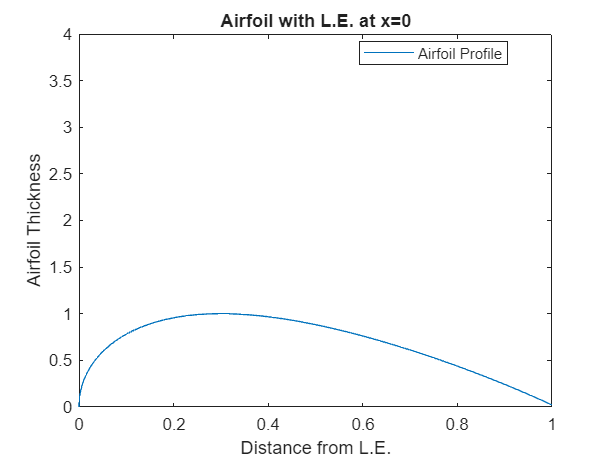
Note that this quotient equals zero for because the value of is taken to be the value of , which was the first that satisfied the error requirement. This quotient is relatively constant across all iterations of Steffensen’s Method, and thus this iteration converges quadratically.

|  |  |
| --- | --- |
|  |  |
| 1 |  |
| 2 |  |
| 3 | 0 |

Code can be found in Appendix A.

Part 2: Newton’s Method

Problem 1: Plot the airfoil described by from to where is the thickness of the airfoil and is the distance from the leading edge of the airfoil ().



Problem 2: Consider the problem of finding the thickest point of the airfoil using Newton’s Method.

Part A: Derive a function such that the root of corresponds to the location of the thickest point of the airfoil.

Note that the thickest point of the airfoil described by will simply be the maximum value of . For some function of , , this will occur when the function’s derivative is equal to zero. Thus, the function is maximized (i.e., the airfoil achieves its maximum thickness) at some from the leading edge such that . Taking this derivative gives

(1)

where the root of corresponds to the location of the thickest point of the airfoil.

Part B: Write a MatLab script to implement Newton’s Method with and . Use as the stopping criterion.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 1 | 0.1972486 | 0.09724864 |
| 2 | 0.2762193 | 0.0789707 |
| 3 | 0.2986261 | 0.02240674 |
| 4 | 0.2998248 | 0.001198702 |
| 5 | 0.2998279 | 3.103562e-06 |
| 6 | 0.2998279 | 2.069539e-11 |

Code can be found in Appendix A.

Problem 3: Now consider the problem of finding the thickest point of the airfoil using Steffensen’s Method.

Part A: Derive Steffensen’s Method from Newton’s Method by replacing with the forward difference and applying the substitution . Explain why is a valid choice.

Begin with Newton’s Method:

(1)

The forward difference approximation for the derivative of the function is:

(2)

This comes from the definition of the derivative, which is:

(3)

Note that this means that the approximation in (2) gets more accurate the closer to zero gets. Because of this, letting is a valid choice because Newton’s method iterates to find the such that , and subsequent values will drive to zero. Applying this gives the function :

(4)

Substituting into equation (1) gives

(5)

which is Steffensen’s Method.

Part B: Write a MatLab script to implement Steffensen’s Method with and . Use as the stopping criterion.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 1 | 0.25318 | 0.15318 |
| 2 | 0.33455 | 0.081362 |
| 3 | 0.31674 | 0.017805 |
| 4 | 0.30374 | 0.012996 |
| 5 | 0.30004 | 0.0037032 |
| 6 | 0.29983 | 0.00021297 |
| 7 | 0.29983 | 6.4019e-07 |
| 8 | 0.29983 | 5.7534e-12 |

Code can be found in Appendix A.

Note that for Steffensen’s Method, the equation was in the iteration used to account for the fact that Steffensen’s Method finds .

Part C: Show that the solution obtained in Part B converges quadratically to by calculating for your largest value of .

For :

(1)

The rest of the derivative values are as follows:

|  |  |
| --- | --- |
|  |  |
| 1 | -1.563 |
| 2 | 1.043 |
| 3 | 0.4613 |
| 4 | 0.1087 |
| 5 | 0.005993 |
| 6 | 1.797e-5 |
| 7 | 0.03098 |
| 8 | 0.08628 |

Code can be found in Appendix A.