Root Finding Method

Andreas Hegseth

July 9, 2024

1 Introduction

Using four root-finding methods bisection, fixed-point, newton and secant methods. The function f(x) = X**3-x-1, through [1, 1.5] with a tolerance of 10**-6.Bisection

2 Math behind the method

The mathematics being used are numerical methods to finding the root of the given function. Each has a different method of finding it, all have there strengths and weakness but all can find the answer but vary from interactions and degrees of accuracy. Absolute error is also used to see the degree of accuracy in our answers and rate of convergence to see how it approaches zero and to see any anomalies. I believe that all of these methods with converge to the root I believe bisection will be the slowest but accurate and fixed-point with be the slowest to converge and I believe that Newton will be the fastest to a more accurate answer and secant is just a variation to Newton if f'(x)=0.

2.1 bisection

Bisection- This method is to take two point in the function and find the midpoint (a+(b-a))/2) in a given set of parameters [a,b]. Then Once that happens then it repeat this process until it find the root.

2.2 fixed-point

Fixed-point- In fixed-point we set an initial guess at p in the function g the it generates a sequence pn n=1 letting pn = g(pn-1) for n1.

2.3 Newton

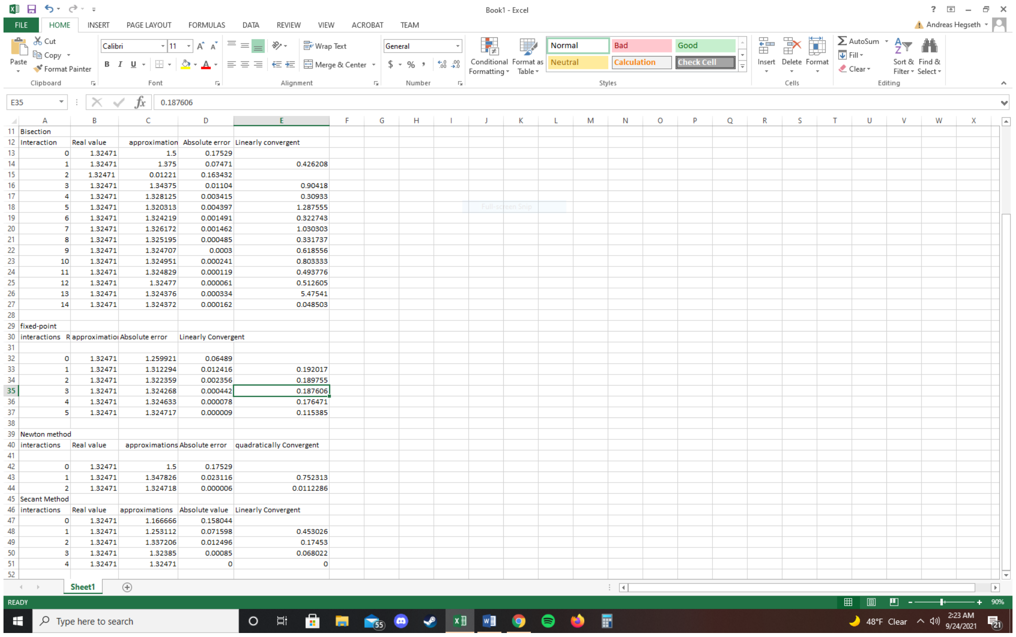
Newton's Method- By using Newton's formula at an initial point we say pn = pn-1 – [f(pn-1)/f'(pn-1)] for n1. We use this formula to find the root of the given function.

2.4 secant

Secant Method- In this method we use the secant formula, with that we choose two points in the function p, q and then apply the formula pn =pn-1-[f(pn-1)*(pn-1-pn-2)]/[f(pn-1)-f(pn-2)] in which it will find the x-interception until it finds p=0.

3 algorithm

With most of these numerical methods I followed the given algorithm in the book Numerical Analysis 10th edition by Richard L. Burden, Douglas J. Faired, and Annette M. Burden. As you will see in my given codes. I used bisection, fixed point, Newton's method and secant method. All have the goal in which to find the root of the given function f(x) = x**3-x-1 which is roughly 1.32471... though the solution is an irrational number. The purpose of these algorithms is to show case if a computer is



	A	B	C	D	E
11	Bisection				
12	Interaction	Real value	approximation	Absolute error	Linearly convergent
13	0	1.32471	1.5	0.17529	
14	1	1.32471	1.375	0.07471	0.426208
15	2	1.32471	0.01221	0.163432	
16	3	1.32471	1.34375	0.01104	0.90418
17	4	1.32471	1.328125	0.003415	0.30933
18	5	1.32471	1.320313	0.004397	1.287555
19	6	1.32471	1.324219	0.001491	0.322743
20	7	1.32471	1.326172	0.001462	1.030303
21	8	1.32471	1.325195	0.000485	0.331737
22	9	1.32471	1.324707	0.0003	0.618556
23	10	1.32471	1.324951	0.000241	0.803333
24	11	1.32471	1.324829	0.000119	0.493776
25	12	1.32471	1.32477	0.000061	0.512605
26	13	1.32471	1.324376	0.000334	5.47541
27	14	1.32471	1.324372	0.000162	0.048503
28					
29	fixed-point				
30	interactions	R approximatioi	Absolute error	Linearly Convergent	
31					
32	0	1.32471	1.259921	0.06489	
33	1	1.32471	1.312294	0.012416	0.192017
34	2	1.32471	1.322359	0.002356	0.189755
35	3	1.32471	1.324268	0.000442	0.187606
36	4	1.32471	1.324633	0.000078	0.176471
37	5	1.32471	1.324717	0.000009	0.115385
38					
39	Newton method				
40	interactions	Real value	approximations	Absolute error	quadratically Convergent
41					
42	0	1.32471	1.5	0.17529	
43	1	1.32471	1.347826	0.023116	0.752313
44	2	1.32471	1.324718	0.000006	0.0112286
45	Secant Method				
46	interactions	Real value	approximations	Absolute value	Linearly Convergent
47	0	1.32471	1.166666	0.158044	
48	1	1.32471	1.253112	0.071598	0.453026
49	2	1.32471	1.337206	0.012496	0.17453
50	3	1.32471	1.32385	0.00085	0.068022
51	4	1.32471	1.32471	0	0

Figure 1: All the results from the code and error and convergent

given the right parameter and the solution is within error of what we want which is 10**-6 then it can run. Each have their own way of solving them.

3.1 Bisection

We see the first interactions with the midpoint between [1, 1.5] in this function is 1.25. It repeat this x-left changes or x-right changes if the parameter are meet there logic statement it then changes the points.(see code for bisection)

3.2 Fixed point

In the code I set an initial guess I find that the closer I am to the approximate answer the less interaction there are but it is 9 interaction when I move father away and when I put it in the negative number it gets the answer has to use complex algebra. This method I find confusing but has more mobility to work with and is accurate.

3.3 Newton's method

I put in Newton's formula I found this one was the quickest to find the approximate answer but it's one weakness is that if f'(x) = 0 then it cannot work. It finds all the x interceptions of the tangent line starting with the initial point, until it finds the root.

3.4 Secant

Secant method start with our two point which I set as our limits and it find the x-interception between the points until meets our tolerance. Though accurate it feel more rigged than the newton's method which a bypass of if f'(x) = 0.

4 data

Look at figure 1

5 conclusion

In the given data I can give my prediction of each method and then draw on the result to see if I was right.

5.1 Bisection conclusion

Bisection I thought Bisection was going to be the fastest because method used logically seem to cut half the function off instantly but after each interaction the rate of accurate in the absolute error begins to slow. I also saw fluctuations in the convergence because of the way the code loops and if one output happens its limits move right then the next move left. As see in figure 1.

5.2 Fixed-point conclusion

In Fixed-point my prediction on fixed point was it was going to be fast but not the fastest in which I was correct when compared to the other methods. The logic of the method seemed to be that if this point exist it will be found now how long could vary depending how we set the limit but since the limit were close it was quick.

5.3 Newton's conclusion

Newton's Method- My prediction this was going to be the fastest as we are using the function derivative to find and if we know anything about calculus this is a tried and true method. I was right it had the shortest interaction but what I didn't think about when I went through the data specifically the convergent rate I found it convergence quadratic ally which given how many interaction there are and how fast we reacted or tolerance of error make sense.

5.4 Secant Conclusion

Secant method I thought that the secant method would be slower than it was because of the use of two point just like the bisection method but was pleasantly surprised. But know looking at the function in our limit in makes sense why it was so fast.