Bisection, Newton-Raphson, Secant, False-Position, and Modified Secant

Athena Ballensky
CS 3010.02
Project 3

All of the variables used to calculate the root were of the data type double. I chose to use the double data type because I needed to select within the floating point category of data types to allow for values after the decimal point, and I also wanted a data type capable of handling very large values. Between the two floating point options, float and double, double was larger.

Function #1

Bisection

n	a	b	c	f(a)	f(b)	f(c)	error
0	0.000	1.000	0.500	-5.000	3.000	1.175	1.000
1	0.000	0.500	0.250	-5.000	1.175	-1.275	1.000
2	0.250	0.500	0.375	-1.275	1.175	0.098	0.333
3	0.250	0.375	0.313	-1.275	0.098	-0.550	0.200
4	0.313	0.375	0.344	-0.550	0.098	-0.217	0.091
5	0.344	0.375	0.359	-0.217	0.098	-0.057	0.043
6	0.359	0.375	0.367	-0.057	0.098	0.021	0.021
7	0.359	0.367	0.363	-0.057	0.021	-0.018	0.011
8	0.363	0.367	0.365	-0.018	0.021	0.001	0.005

Bisection Method: The root 0.365 has been found between 0.000 and 1.000 for function #1 in 9 iterations.

n	a	b	c	f(a)	f(b)	f(c)	error
0	1.000	2.000	1.500	3.000	-0.400	1.975	1.000
1	1.500	2.000	1.750	1.975	-0.400	0.863	0.143
2	1.750	2.000	1.875	0.863	-0.400	0.238	0.067
3	1.875	2.000	1.938	0.238	-0.400	-0.081	0.032
4	1.875	1.938	1.906	0.238	-0.081	0.079	0.016
5	1.906	1.938	1.922	0.079	-0.081	-0.001	0.008

Bisection Method: The root 1.922 has been found between 1.000 and 2.000 for function #1 in 6 iterations.

For the example below, no root was found with the starting guesses x = 2 and x = 3. Because the Bisection method is a bracketing method, the root can only be identified if it is in between the two guesses. Since there is no root in between x = 2 and x = 3 for function #1, no root was found.

n	a	b	c	f(a)	f(b)	f(c)	error
0	2.000	3.000	2.500	-0.400	-3.200	-2.625	1.000
1	2.500	3.000	2.750	-2.625	-3.200	-3.212	0.091
2	2.750	3.000	2.875	-3.212	-3.200	-3.293	0.043
3	2.875	3.000	2.938	-3.293	-3.200	-3.270	0.021
4	2.938	3.000	2.969	-3.270	-3.200	-3.241	0.011
5	2.969	3.000	2.984	-3.241	-3.200	-3.222	0.005
6	2.984	3.000	2.992	-3.222	-3.200	-3.211	0.003
7	2.992	3.000	2.996	-3.211	-3.200	-3.206	0.001
8	2.996	3.000	2.998	-3.206	-3.200	-3.203	0.001

Bisection Method: There are no roots between 2.000 and 3.000 for function #1.

n	a	b	c	f(a)	f(b)	f(c)	error
0	3.000	4.000	3.500	-3.200	6.600	-0.625	1.000
1	3.500	4.000	3.750	-0.625	6.600	2.313	0.067
2	3.500	3.750	3.625	-0.625	2.313	0.687	0.034
3	3.500	3.625	3.563	-0.625	0.687	-0.007	0.018

Bisection Method: The root 3.563 has been found between 3.000 and 4.000 for function #1 in 4 iterations.

False Position

n	a	b	c	f(a)	f(b)	f(c)	error
0	0.000	1.000	0.625	-5.000	3.000	1.980	1.000
1	0.000	0.625	0.448	-5.000	1.980	0.758	0.396
2	0.000	0.448	0.389	-5.000	0.758	0.230	0.152
3	0.000	0.389	0.372	-5.000	0.230	0.065	0.046
4	0.000	0.372	0.367	-5.000	0.065	0.018	0.013
5	0.000	0.367	0.366	-5.000	0.018	0.005	0.004

False Position Method: The root 0.366 has been found between 0.000 and 1.000 for function #1 in 6 iterations.

n	a	b	c	f(a)	f(b)	f(c)	error
0	1.000	2.000	1.882	3.000	-0.400	0.201	1.000
1	1.882	2.000	1.922	0.201	-0.400	0.000	0.020

False Position Method: The root 1.922 has been found between 1.000 and 2.000 for function #1 in 2 iterations.

For the example below, no root was found with the starting guesses x = 2 and x = 3. Because the False Position method is a bracketing method, the root can only be identified if it is in between the two guesses. Since there is no root in between x = 2 and x = 3 for function #1, no root was found.

n	a	b	c	f(a)	f(b)	f(c)	error

False Position Method: f(a) = -0.400 and f(b) = -3.200 have the same sign. No root is bracketed between 2.000 and 3.000 for function #1.

n	a	b	c	f(a)	f(b)	f(c)	error
0	3.000	4.000	3.327	-3.200	6.600	-1.969	1.000
1	3.327	4.000	3.481	-1.969	6.600	-0.796	0.044
2	3.481	4.000	3.537	-0.796	6.600	-0.267	0.016
3	3.537	4.000	3.555	-0.267	6.600	-0.084	0.005
4	3.555	4.000	3.561	-0.084	6.600	-0.026	0.002
5	3.561	4.000	3.562	-0.026	6.600	-0.008	0.000

False Position Method: The root 3.562 has been found between 3.000 and 4.000 for function #1 in 6 iterations.

Newton Raphson

n	x_n	x_n+1	f(x_n)	f(x_n+1)	f'(x_n)	error
0	0.000	0.282	-5.000	-0.889	17.700	1.000
1	0.282	0.359	-0.889	-0.058	11.569	1.000
2	0.359	0.365	-0.058	-0.000	10.067	0.226

Newton Raphson Method: The root 0.365 has been found with a starting guess of x = 0.000 for the function #1 in 3 iterations.

The example below demonstrates the methodology I used to determine divergence. I had conditions checking if the current x value was equal to NaN, if the current value was equal to infinity, if the error was greater than the maximum diverging error, or if f'(x) was zero. In this case, the large relative approximate error triggered the divergence message and returned the method.

n	x_n	x_n+1	f(x_n)	f(x_n+1)	f'(x_n)	error
0	1.000	-9.000	3.000	-2570.000	0.300	1.000
1	-9.000	-5.402	-2570.000	-757.341	714.300	1.185
2	-5.402	-3.029	-757.341	-221.608	319.203	1.971
3	-3.029	-1.487	-221.608	-63.756	143.656	2.633
4	-1.487	-0.517	-63.756	-17.563	65.756	4.857
5	-0.517	0.042	-17.563	-4.279	31.409	36.486

Newton Raphson Method: Error. This equation is diverging.

n	x_n	x_n+1	f(x_n)	f(x_n+1)	f'(x_n)	error
0	2.000	1.922	-0.400	0.001	-5.100	1.000

Newton Raphson Method: The root 1.922 has been found with a starting guess of x = 2.000 for the function #1 in 1 iterations.

n	x_n	x_n+1	f(x_n)	f(x_n+1)	f'(x_n)	error
0	3.000	5.133	-3.200	48.090	1.500	1.000
1	5.133	4.270	48.090	12.956	55.687	0.297
2	4.270	3.793	12.956	2.948	27.172	0.353
3	3.793	3.600	2.948	0.398	15.263	0.186
4	3.600	3.564	0.398	0.012	11.216	0.064
5	3.564	3.563	0.012	0.000	10.522	0.010

Newton Raphson Method: The root 3.563 has been found with a starting guess of x = 3.000 for the function #1 in 6 iterations.

Secant

n	x_i-1	x_i	x_i+1	f(x_i-1)	f(x_i)	f(x_i+1)	error
1	0.000	1.000	0.625	-5.000	3.000	1.980	0.600
2	1.000	0.625	-0.103	3.000	1.980	-6.958	7.042
3	0.625	-0.103	0.464	1.980	-6.958	0.890	1.223
4	-0.103	0.464	0.399	-6.958	0.890	0.329	0.161
5	0.464	0.399	0.362	0.890	0.329	-0.036	0.104
6	0.399	0.362	0.365	0.329	-0.036	0.001	0.010

Secant Method: The root 0.365 has been found with the starting guesses x = 0.000 and x = 1.000 for function #1 in 6 iterations.

Looking at the two examples below, note that with the Secant method, unlike the two bracketing methods of Bisection and False Position, several variations of x values will yield a correct result for the root. This is one of the benefits of the Secant method.

```
n          x_i-1           x_i             x_i+1           f(x_i-1)        f(x_i)          f(x_i+1)        error
------------------------------------------------------------------------------------------------------------------
1          1.000           2.000           1.882           3.000           -0.400          0.201           0.062
2          2.000           1.882           1.922           -0.400          0.201           0.000           0.020
Secant Method: The root 1.922 has been found with the starting guesses x = 1.000 and x = 2.000 for function #1 in 2 iterations.
```

```
n          x_i-1           x_i             x_i+1           f(x_i-1)        f(x_i)          f(x_i+1)        error
------------------------------------------------------------------------------------------------------------------
1          2.000           3.000           1.857           -0.400          -3.200          0.329           0.615
2          3.000           1.857           1.964           -3.200          0.329           -0.214          0.054
3          1.857           1.964           1.922           0.329           -0.214          0.001           0.022
Secant Method: The root 1.922 has been found with the starting guesses x = 2.000 and x = 3.000 for function #1 in 3 iterations.
```

```
n          x_i-1           x_i             x_i+1           f(x_i-1)        f(x_i)          f(x_i+1)        error
------------------------------------------------------------------------------------------------------------------
1          3.000           4.000           3.327           -3.200          6.600           -1.969          0.202
2          4.000           3.327           3.481           6.600           -1.969          -0.796          0.044
3          3.327           3.481           3.586           -1.969          -0.796          0.248           0.029
4          3.481           3.586           3.561           -0.796          0.248           -0.019          0.007
5          3.586           3.561           3.563           0.248           -0.019          -0.000          0.001
Secant Method: The root 3.563 has been found with the starting guesses x = 3.000 and x = 4.000 for function #1 in 5 iterations.
```

Modified Secant

The NaN value for f(x_i+1) triggered the divergence message and caused the return of the program.

```
n          x_i             x_i+1           f(x_i)          f(x_i+1)        f'(x_i)         error
--------------------------------------------------------------------------------------------------
0          0.000           NaN             -5.000          NaN             17.700          NaN
Mofified Secant Method: Error. This equation is diverging.
```

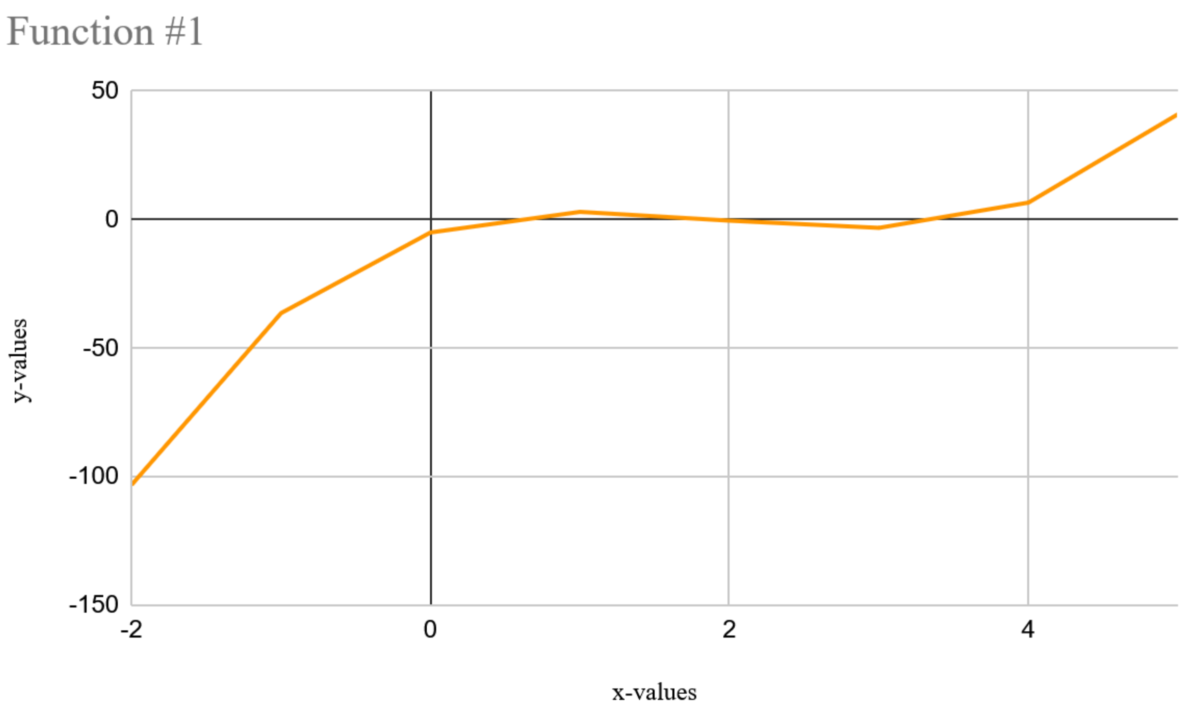
```
n          x_i             x_i+1           f(x_i)          f(x_i+1)        f'(x_i)         error
--------------------------------------------------------------------------------------------------
0          1.000           -11.336         3.000           -4622.118       0.300           1.088
1          -11.336         -6.947          -4622.118       -1362.937       1053.916        0.632
2          -6.947          -4.042          -1362.937       -399.770        469.776         0.719
3          -4.042          -2.138          -399.770        -115.859        210.310         0.891
4          -2.138          -0.917          -115.859        -32.616         95.150          1.331
5          -0.917          -0.176          -32.616         -8.498          44.206          4.199
6          -0.176          0.212           -8.498          -1.757          22.015          1.833
7          0.212           0.348           -1.757          -0.174          13.012          0.391
8          0.348           0.365           -0.174          -0.001          10.285          0.047
Modified Secant Method: The root 0.365 has been found with a starting guess of x = 1.000 for the function #1 in 9 iterations.
```

```
n          x_i             x_i+1           f(x_i)          f(x_i+1)        f'(x_i)         error
--------------------------------------------------------------------------------------------------
0          2.000           1.921           -0.400          0.001           -5.100          0.041
Modified Secant Method: The root 1.921 has been found with a starting guess of x = 2.000 for the function #1 in 1 iterations.
```

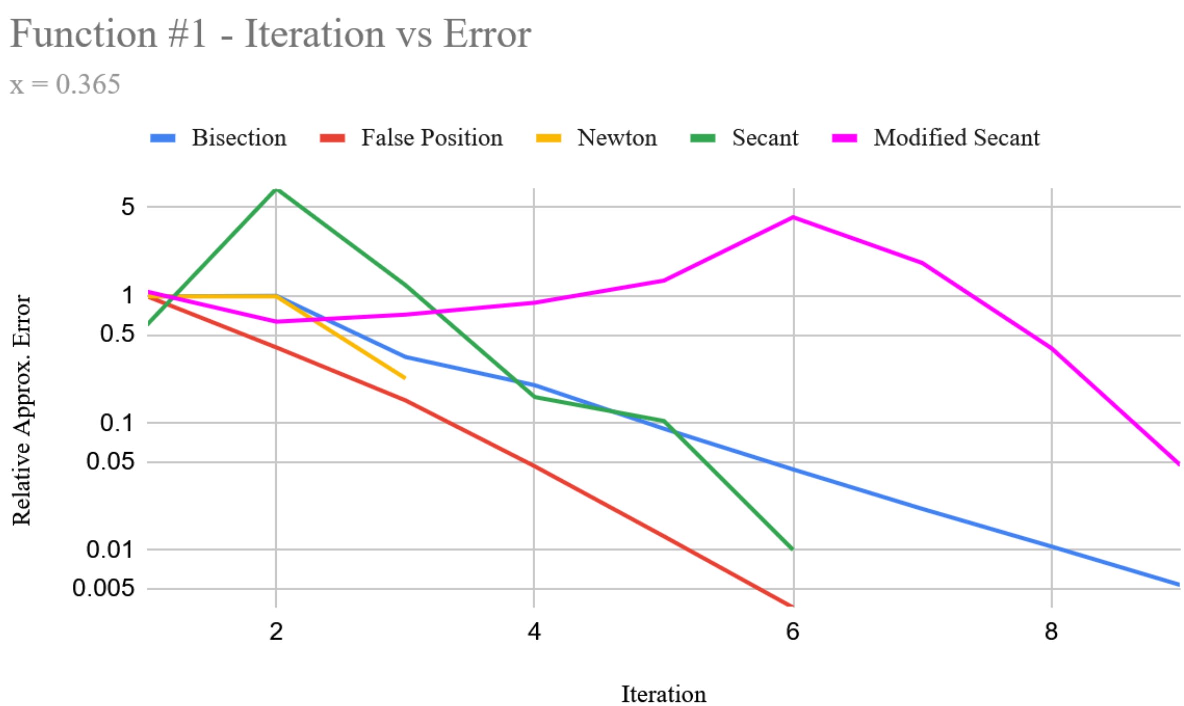
```
n          x_i             x_i+1           f(x_i)          f(x_i+1)        f'(x_i)         error
--------------------------------------------------------------------------------------------------
0          3.000           4.893           -3.200          35.763          1.500           0.387
1          4.893           4.138           35.763          9.604           46.838          0.182
2          4.138           3.737           9.604           2.133           23.599          0.107
3          3.737           3.589           2.133           0.278           14.053          0.041
4          3.589           3.564           0.278           0.013           11.003          0.007
5          3.564           3.563           0.013           0.000           10.523          0.000
Modified Secant Method: The root 3.563 has been found with a starting guess of x = 3.000 for the function #1 in 6 iterations.
```

Graphs

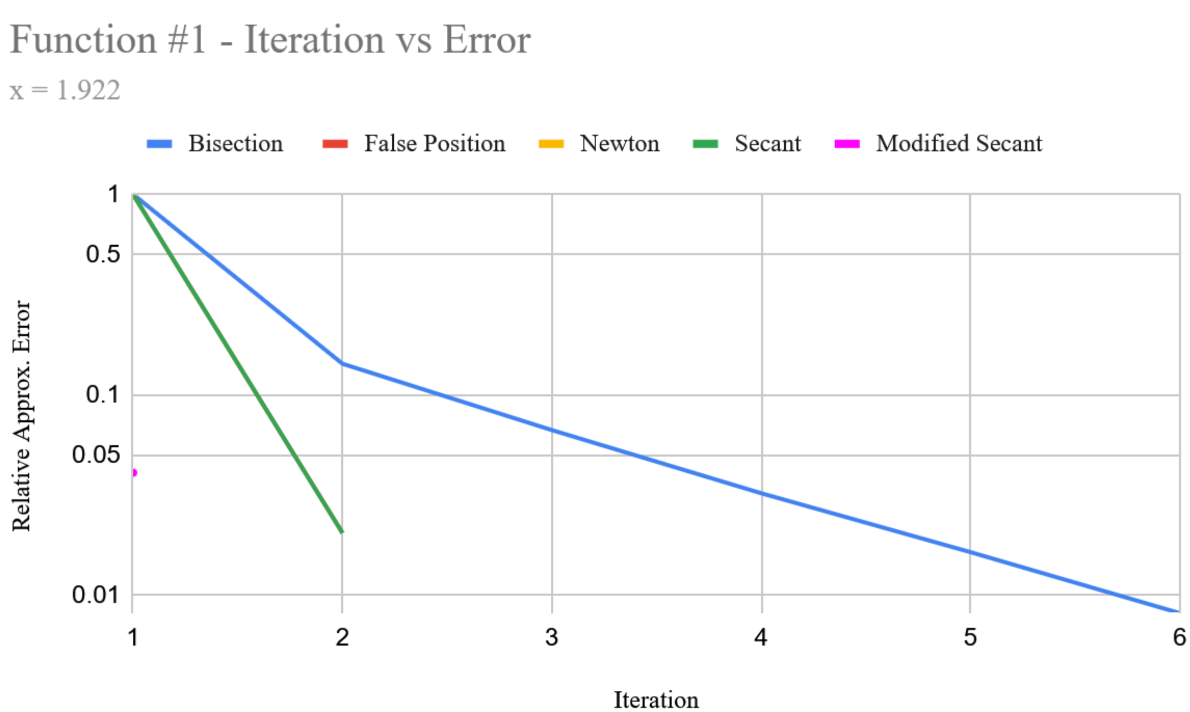
The graph of function #1 crosses the x-axis 3 times from x = -2 to x = 5, signifying the existence of the three roots x = 0.365, x = 1.922, and x = 3.563.



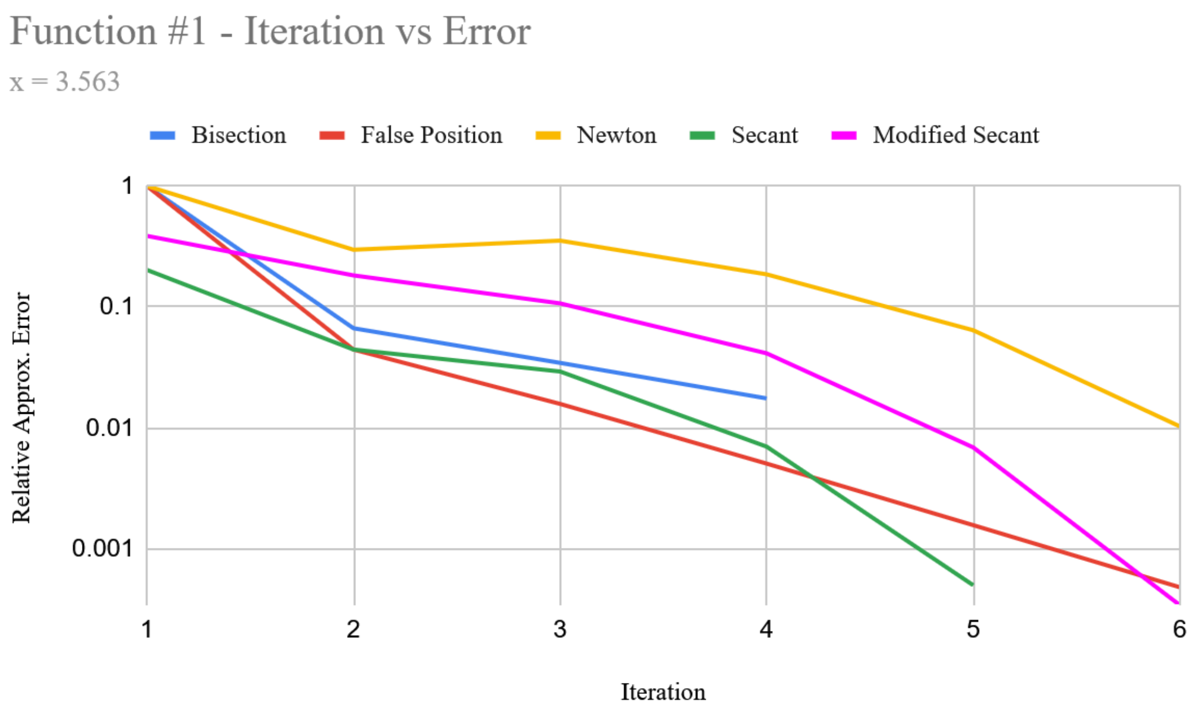
In the graph below, some lines come to a halt faster than others because the associated methods were able to determine the root in fewer iterations.



For the root x = 1.922, many of the lines are missing from the graph below because the actual root was identified so rapidly that only one point was available for the error–below the minimum of two points to plot a line.



Compared to the other roots, the process of finding x = 3.563 was the most similar amongst the different methods. One thing I noticed while programming is that modifying the desired error from .1% to 1% resulted in almost none of the methods finding x = 3.563, but still finding the other two roots. This suggests to me that x = 3.563 is a comparably tricky root for these methods to identify, and requires extra precision.

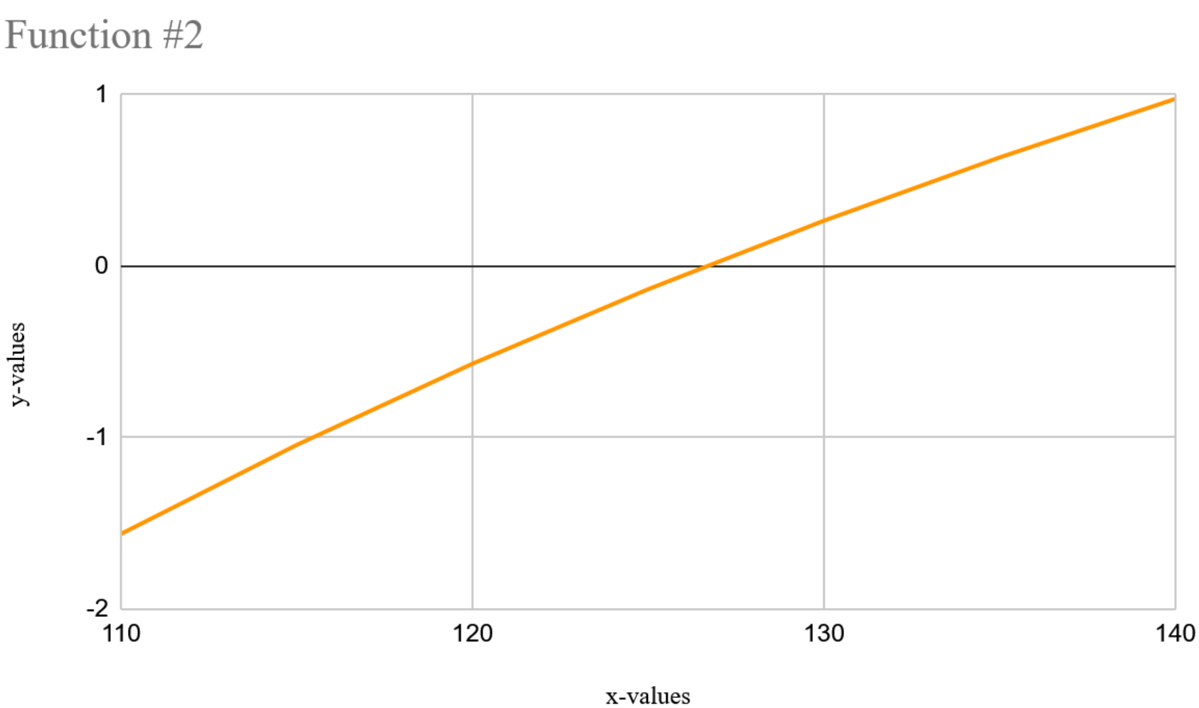


Function #2

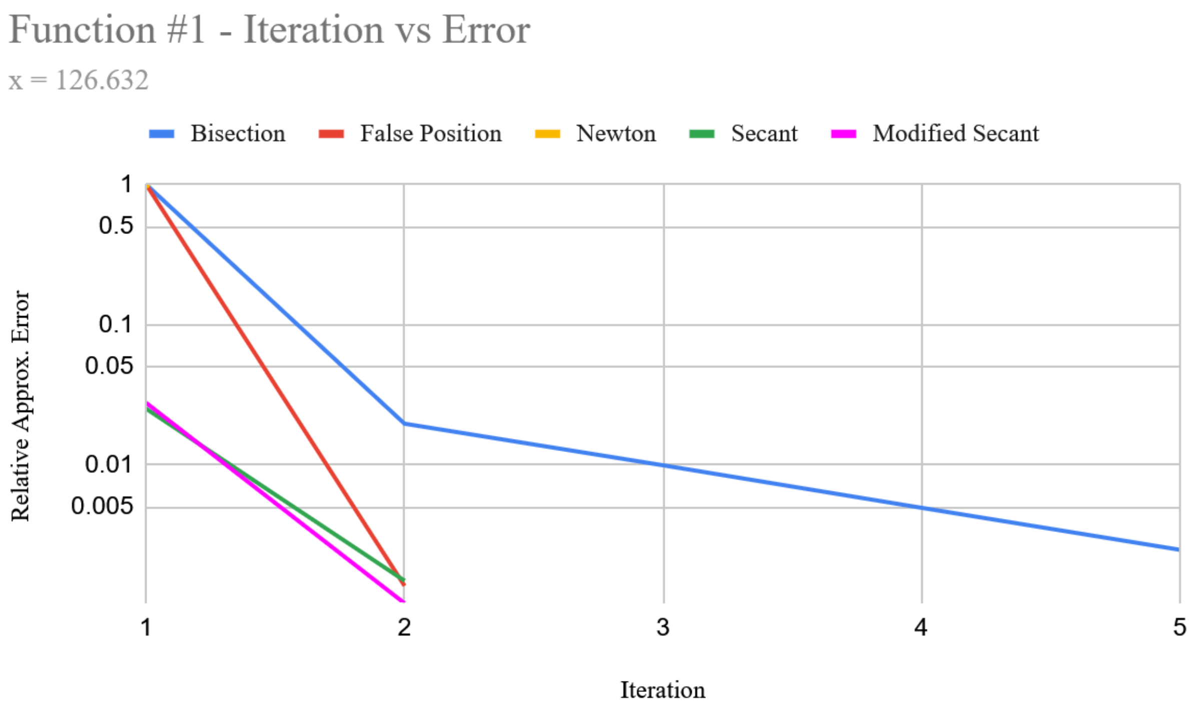
Because of the simpler nature of function #2 in the range of x = 120 to x = 130, with a continuously increasing curve and only one root, none of my divergence tests or other error messages were triggered.

```
n          a              b              c              f(a)           f(b)           f(c)           error
------------------------------------------------------------------------------------------------------------
0          120.000        130.000        125.000        -0.568         0.265          -0.134         1.000
1          125.000        130.000        127.500        -0.134         0.265          0.070          0.020
2          125.000        127.500        126.250        -0.134         0.070          -0.031         0.010
3          126.250        127.500        126.875        -0.031         0.070          0.020          0.005
4          126.250        126.875        126.563        -0.031         0.020          -0.006         0.002
Bisection Method: The root 126.563 has been found between 120.000 and 130.000 for function #2 in 5 iterations.

n          a              b              c              f(a)           f(b)           f(c)           error
------------------------------------------------------------------------------------------------------------
0          120.000        130.000        126.816        -0.568         0.265          0.015          1.000
1          120.000        126.816        126.642        -0.568         0.015          0.001          0.001
False Position Method: The root 126.642 has been found between 120.000 and 130.000 for function #2 in 2 iterations.

n          x_n            x_n+1          f(x_n)         f(x_n+1)       f'(x_n)        error
-------------------------------------------------------------------------------------------
0          130.000        126.540        0.265          -0.008         0.077          1.000
Newton Raphson Method: The root 126.540 has been found with a starting guess of x = 130.000 for the function #2 in 1 iterations.

n          x_i-1          x_i            x_i+1          f(x_i-1)       f(x_i)         f(x_i+1)       error
------------------------------------------------------------------------------------------------------------
Secant Method: The root 126.627 has been found with the starting guesses x = 120.000 and x = 130.000 for function #2 in 2 iterations.

n           x_i           x_i+1          f(x_i)         f(x_i+1)       f'(x_i)        error
-------------------------------------------------------------------------------------------
0          130.000        126.504        0.265          -0.010         0.077          0.028
1          126.504        126.634        -0.010         0.000          0.081          0.001
Modified Secant Method: The root 126.634 has been found with a starting guess of x = 130.000 for the function #2 in 2 iterations.
```

Graphs



Note that for the graph below, the line for the Newton method is not shown because the Newton method identified the root x = 126.632 in just one iteration. The Bisection method required the most iterations, which is to be expected given that my two starting values were x = 120 and x = 130, which is a larger range of numbers than I would normally supply.



Function #1 - Iteration vs Error
x = 126.632
Bisection
False Position
Newton
Secant
Modified Secant
1
0.5
0.1
0.05
0.01
0.005
Relative Approx. Error
1
2
3
4
5
Iteration