Project 3:

FINDING ROOTS METHODS

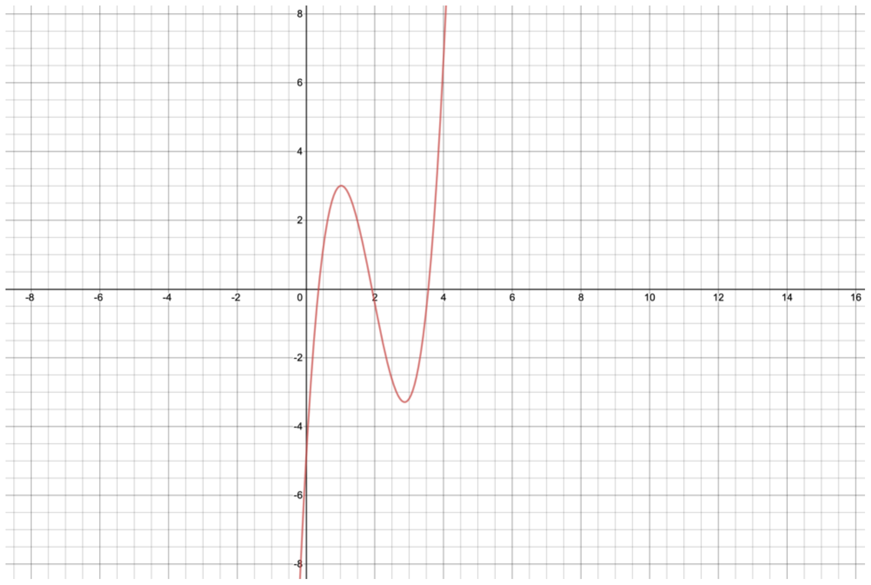
Bisection, Newton-Raphson, Secant, False-Position and Modified Secant

DAI VUONG

CS3010-01 FALL 2020

Professor: LAIPAT RAI RAHEJA

(a) $f(x) = 2x^3 – 11.7x^2 + 17.7x – 5$



- Root 1: 0.3651

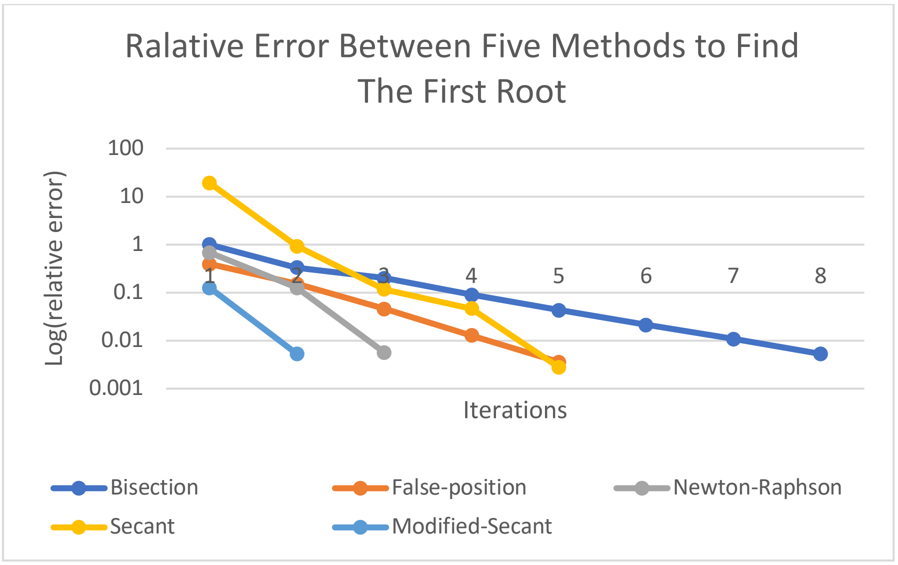
```
BISECTION METHOD:
-----------------
  n       a_n       b_n       c_n     f(a_n)     f(b_n)     f(c_n)     error
  0    0.0000    1.0000    0.5000    -5.0000     3.0000     1.1750    N/A
  1    0.0000    0.5000    0.2500    -5.0000     1.1750    -1.2750    1.0000
  2    0.2500    0.5000    0.3750    -1.2750     1.1750     0.0977    0.3333
  3    0.2500    0.3750    0.3125    -1.2750     0.0977    -0.5503    0.2000
  4    0.3125    0.3750    0.3438    -0.5503     0.0977    -0.2169    0.0909
  5    0.3438    0.3750    0.3594    -0.2169     0.0977    -0.0573    0.0435
  6    0.3594    0.3750    0.3672    -0.0573     0.0977     0.0208    0.0213
  7    0.3594    0.3672    0.3633    -0.0573     0.0208    -0.0181    0.0108
  8    0.3633    0.3672    0.3652    -0.0181     0.0208     0.0014    0.0053
```

```
FALSE POSITION METHOD:
------------------------
  n       a_n       b_n       c_n     f(a_n)     f(b_n)     f(c_n)     error
  0    0.0000    1.0000    0.6250    -5.0000     3.0000     1.9805    N/A
  1    0.0000    0.6250    0.4477    -5.0000     1.9805     0.7585    0.3961
  2    0.0000    0.4477    0.3887    -5.0000     0.7585     0.2298    0.1517
  3    0.0000    0.3887    0.3716    -5.0000     0.2298     0.0646    0.0460
  4    0.0000    0.3716    0.3669    -5.0000     0.0646     0.0178    0.0129
  5    0.0000    0.3669    0.3656    -5.0000     0.0178     0.0049    0.0036
```

```
NEWTON-RAPHSON METHOD:
------------------------
  n       x_n      fx_n   f'(x_n)      x_n+1   f(x_n+1)     error
  1    0.1000   -3.3450   15.4200     0.3169    -0.5019    0.6845
  2    0.3169   -0.5019   10.8866     0.3630    -0.0206    0.1270
  3    0.3630   -0.0206    9.9958     0.3651    -0.0000    0.0057
```

```
SECANT METHOD:
----------------
  n      x_n-1        x_n  f(x_n-1)     f(x_n)      x_n+1   f(x_n+1)     error
  0     0.1000     1.0000   -3.3450     3.0000     0.5745     1.6861    N/A
  1     1.0000     0.5745    3.0000     1.6861     0.0284    -4.5067   19.2286
  2     0.5745     0.0284    1.6861    -4.5067     0.4258     0.5697    0.9333
  3     0.0284     0.4258   -4.5067     0.5697     0.3812     0.1578    0.1170
  4     0.4258     0.3812    0.5697     0.1578     0.3641    -0.0099    0.0469
  5     0.3812     0.3641    0.1578    -0.0099     0.3651     0.0002    0.0028

MODIFIED SECANT METHOD:
-------------------------
  n        x_n      x_n+1    f(x_n)   f(x_n+1)     error
  0     0.1000     0.3171   -3.3450    -0.5002    N/A
  1     0.3171     0.3632   -0.5002    -0.0192    0.1269
  2     0.3632     0.3651   -0.0192     0.0000    0.0053
```



For the first root 0.3651, I choose

- $a_n = 0$ and $b_n = 1$ for bisection and false-position method,
- $x_n = 1$ for Newton-Raphson,
- $x_{n-1} = 0.1$ and $x_n = 1$ for Secant,
- $x_n = 0.1$ for modified-secant method.

The bisection method takes 8 iterations to reach the condition $e_a < 1\%$, it takes the most iteration to compare to all other methods, which means the bisection converges slowest. The false-position method takes 5 iterations to satisfy the error condition. And of course, it converges faster than bisection method. In this particular case, the Newton-Raphson method takes 3 iterations, a lot faster than bisection method. It converges pretty quickly. Secant method takes 5 iterations to satisfy the error, as same as false-position method, But the first error of secant method is a lot higher than all others. However, secant method converges faster than false-position method. Finally, modified-secant method is the fastest method. It takes 2 iterations to satisfy the error, it also converges fastest. Even all methods satisfy the error, but bisection method and false-position method don't get the exact root yet. Three other methods get the root exactly when they stop while the error condition satisfied.

- Root 2: 1.9217

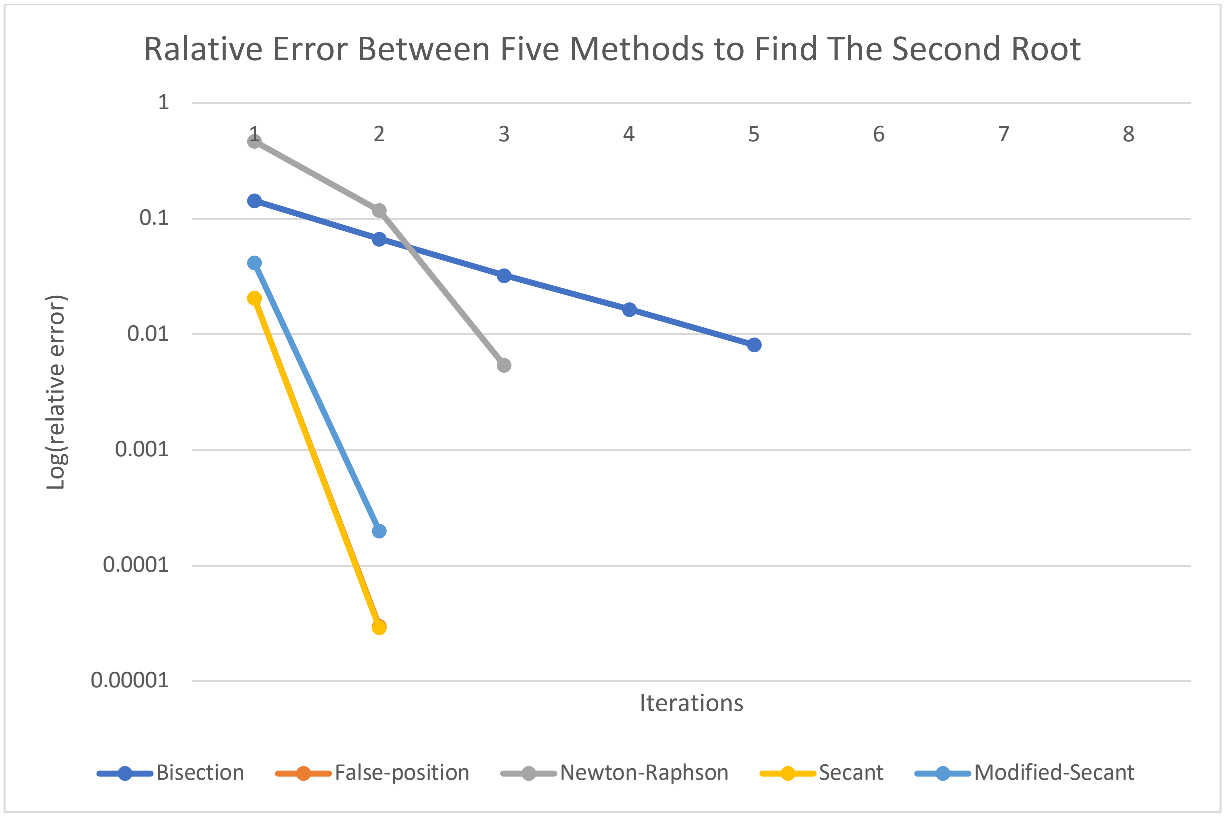
```
BISECTION METHOD:
-----------------
  n       a_n       b_n       c_n     f(a_n)    f(b_n)    f(c_n)     error
  0    1.0000    2.0000    1.5000    3.0000   -0.4000    1.9750    N/A
  1    1.5000    2.0000    1.7500    1.9750   -0.4000    0.8625    0.1429
  2    1.7500    2.0000    1.8750    0.8625   -0.4000    0.2383    0.0667
  3    1.8750    2.0000    1.9375    0.2383   -0.4000   -0.0806    0.0323
  4    1.8750    1.9375    1.9063    0.2383   -0.0806    0.0791    0.0164
  5    1.9063    1.9375    1.9219    0.0791   -0.0806   -0.0007    0.0081

FALSE POSITION METHOD:
------------------------
  n       a_n       b_n       c_n     f(a_n)    f(b_n)    f(c_n)     error
  0    1.0000    2.0000    1.8824    3.0000   -0.4000    0.2009    N/A
  1    1.8824    2.0000    1.9217    0.2009   -0.4000    0.0003    0.0205
  2    1.9217    2.0000    1.9217    0.0003   -0.4000    0.0000    0.0000

NEWTON-RAPHSON METHOD:
------------------------
  n       x_n      fx_n   f'(x_n)     x_n+1  f(x_n+1)     error
  1    2.5000   -2.6250   -3.3000    1.7045    1.0814    0.4667
  2    1.7045    1.0814   -4.7535    1.9320   -0.0527    0.1178
  3    1.9320   -0.0527   -5.1131    1.9217   -0.0000    0.0054

SECANT METHOD:
----------------
  n     x_n-1       x_n  f(x_n-1)    f(x_n)     x_n+1   f(x_n+1)     error
  0    1.0000    2.0000    3.0000   -0.4000    1.8824     0.2009    N/A
  1    2.0000    1.8824   -0.4000    0.2009    1.9217     0.0003    0.0205
  2    1.8824    1.9217    0.2009    0.0003    1.9217    -0.0000    0.0000

MODIFIED SECANT METHOD:
-------------------------
  n       x_n     x_n+1    f(x_n)   f(x_n+1)    error
  0    1.5000    2.0013    1.9750    -0.4064   N/A
  1    2.0013    1.9214   -0.4064     0.0015   0.0415
  2    1.9214    1.9217    0.0015     0.0000   0.0002
```



The second root is 1.9217. I choose

- $a_n = 1$ and $b_n = 2$ for bisection and false-position method,
- $x_n = 2.5$ for Newton-Raphson
- $x_{n-1} = 1$ and $x_n = 2$ for Secant
- $x_n = 1.5$ for modified-secant method.

The bisection method takes 5 iterations to reach the error condition. As previous root, bisection takes the most iteration to compare to all other methods, so bisection still converges slowest.

The false-position method takes 2 iterations to satisfy the error condition. In this case, false-position and secant get the same error for 2 iterations, the line of secant method is over the line of false-position method, so we barely see the line of false-position method.

For this second root, the Newton-Raphson method also takes 3 iterations. Newton-Raphson is faster than false-position and secant method for the first root, but it is slower in this case.

Modified-secant method is the fastest method. It takes 2 iterations to satisfy the error

After all methods satisfy the error, only bisection method doesn't get the exact root yet (`1.9219`). And the rest get exactly the root when they stop while the error condition satisfied.

- Root 3: 3.5630

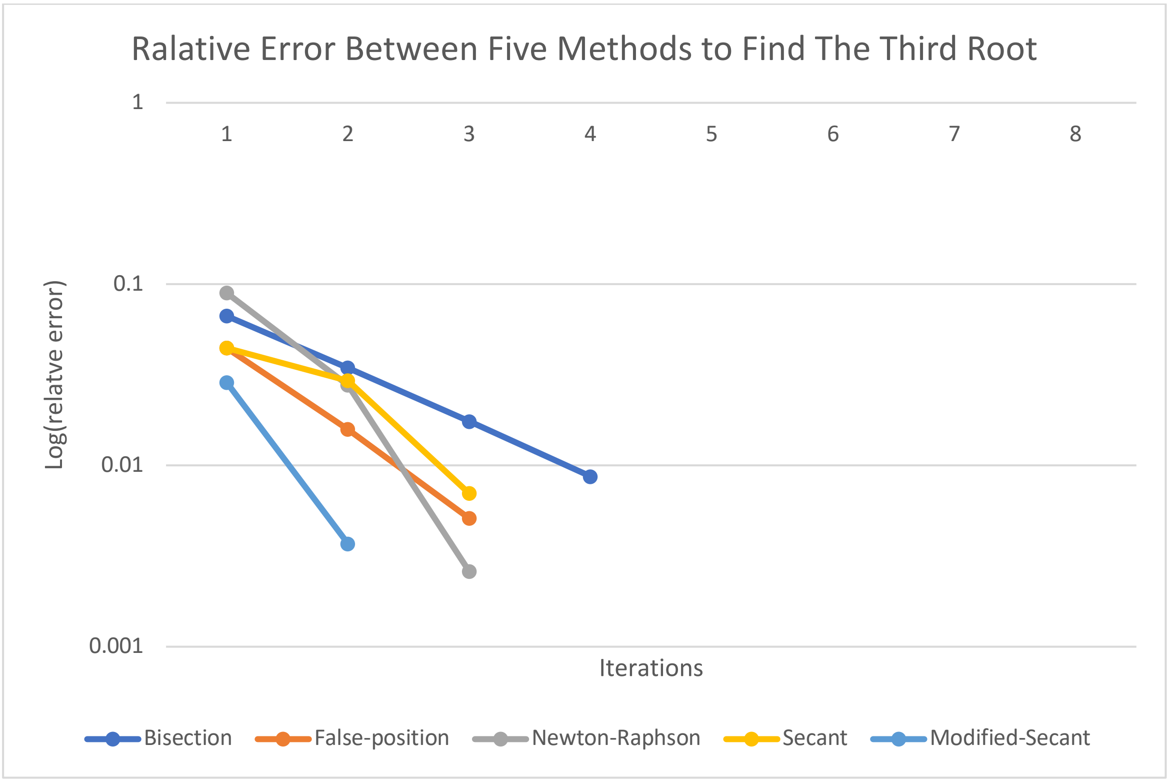
```
BISECTION METHOD:
-----------------
  n       a_n       b_n       c_n    f(a_n)    f(b_n)    f(c_n)     error
  0    3.0000    4.0000    3.5000   -3.2000    6.6000   -0.6250    N/A
  1    3.5000    4.0000    3.7500   -0.6250    6.6000    2.3125    0.0667
  2    3.5000    3.7500    3.6250   -0.6250    2.3125    0.6867    0.0345
  3    3.5000    3.6250    3.5625   -0.6250    0.6867   -0.0069    0.0175
  4    3.5625    3.6250    3.5938   -0.0069    0.6867    0.3303    0.0087
```

```
FALSE POSITION METHOD:
------------------------
  n       a_n       b_n       c_n    f(a_n)    f(b_n)    f(c_n)     error
  0    3.0000    4.0000    3.3265   -3.2000    6.6000   -1.9689    N/A
  1    3.3265    4.0000    3.4813   -1.9689    6.6000   -0.7959    0.0444
  2    3.4813    4.0000    3.5371   -0.7959    6.6000   -0.2671    0.0158
  3    3.5371    4.0000    3.5551   -0.2671    6.6000   -0.0840    0.0051
```

```
NEWTON-RAPHSON METHOD:
------------------------
  n       x_n      fx_n   f'(x_n)     x_n+1  f(x_n+1)     error
  1    4.0000    6.6000   20.1000    3.6716    1.2554    0.0894
  2    3.6716    1.2554   12.6693    3.5726    0.0995    0.0277
  3    3.5726    0.0995   10.6811    3.5632    0.0008    0.0026
```

```
SECANT METHOD:
----------------
  n     x_n-1       x_n  f(x_n-1)    f(x_n)     x_n+1   f(x_n+1)    error
  0    3.0000    4.0000   -3.2000    6.6000    3.3265    -1.9689   N/A
  1    4.0000    3.3265    6.6000   -1.9689    3.4813    -0.7959   0.0444
  2    3.3265    3.4813   -1.9689   -0.7959    3.5863     0.2479   0.0293
  3    3.4813    3.5863   -0.7959    0.2479    3.5613    -0.0191   0.0070
```

```
MODIFIED SECANT METHOD:
-------------------------
  n       x_n     x_n+1    f(x_n)   f(x_n+1)    error
  0    4.0000    3.6795    6.6000     1.3560   N/A
  1    3.6795    3.5769    1.3560     0.1465   0.0287
  2    3.5769    3.5638    0.1465     0.0063   0.0037
```



The third root is 3.563. I choose

- $a_n = 3$ and $b_n = 4$ for bisection and false-position method,
- $x_n = 4$ for Newton-Raphson
- $x_{n-1} = 3$ and $x_n = 4$ for Secant
- $x_n = 4$ for modified-secant method.

The bisection method takes 4 iterations to reach the error condition. Similar to the previous roots, bisection method takes the most iteration, so bisection still converges slowest.

The false-position method takes 3 iterations to satisfy the error condition. In this particular case, secant also takes 3 iterations. It is kind of similar to the two previous cases above. However, false-position method seems converge faster than secant method.

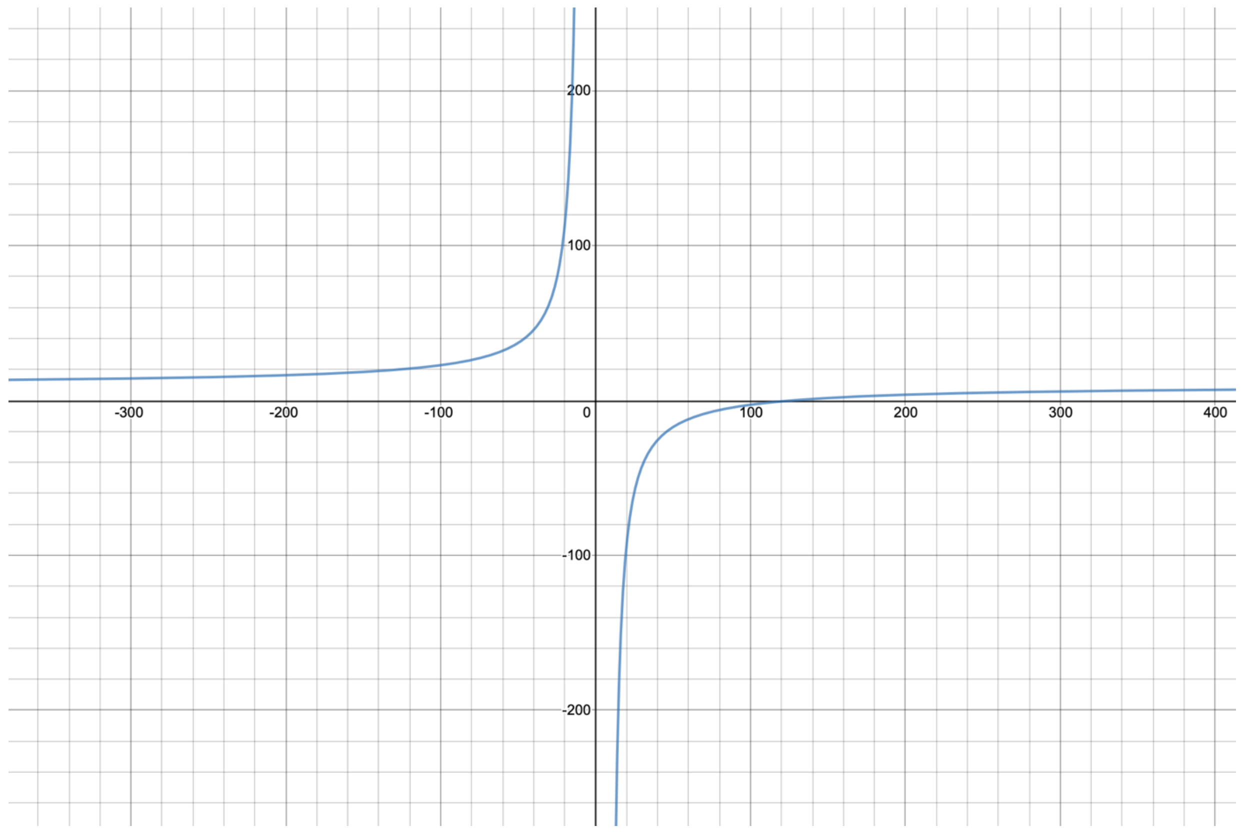
Newton-Raphson method also takes 3 iterations for this third case, it takes 3 iterations as same as false-position and secant method. However, newton-raphson converges faster than false-position method.

Modified-secant method is the fastest method. It takes 2 iterations to satisfy the error. At the first iteration, the error is 0.0287, closes to the error condition.

After all methods stop while satisfy the error condition, only newton-raphson and modified-secant method get close to the root.

From the first root to the third root, the first error of each method get closer to the error condition.

(b) f(x) = x + 10 – x*cosh(50/x)



- Root: 126.632

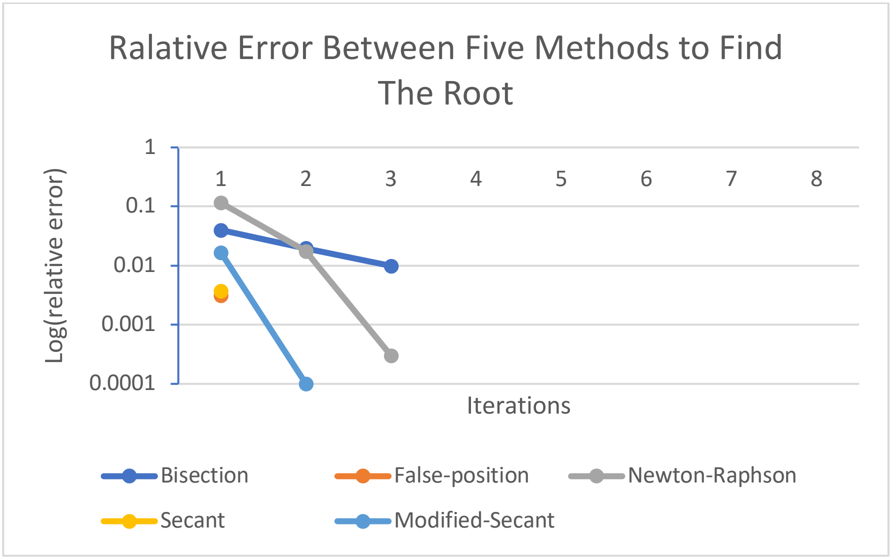
```
BISECTION METHOD:
-----------------
  n       a_n       b_n       c_n     f(a_n)    f(b_n)    f(c_n)     error
  0  110.0000  130.0000  120.0000   -1.5606    0.2655   -0.5682    N/A
  1  120.0000  130.0000  125.0000   -0.5682    0.2655   -0.1340    0.0400
  2  125.0000  130.0000  127.5000   -0.1340    0.2655    0.0698    0.0196
  3  125.0000  127.5000  126.2500   -0.1340    0.0698   -0.0311    0.0099
```

```
FALSE POSITION METHOD:
------------------------
  n       a_n       b_n       c_n     f(a_n)    f(b_n)    f(c_n)     error
  0  110.0000  130.0000  127.0923   -1.5606    0.2655    0.0371    N/A
  1  110.0000  127.0923  126.6952   -1.5606    0.0371    0.0051    0.0031
```

```
NEWTON-RAPHSON METHOD:
------------------------
  n       x_n      fx_n    f'(x_n)     x_n+1  f(x_n+1)     error
  1  110.0000   -1.5606    0.1087  124.3569   -0.1879    0.1154
  2  124.3569   -0.1879    0.0841  126.5900   -0.0034    0.0176
  3  126.5900   -0.0034    0.0811  126.6324   -0.0000    0.0003
```

```
SECANT METHOD:
----------------
  n     x_n-1       x_n  f(x_n-1)     f(x_n)      x_n+1   f(x_n+1)    error
  0  110.0000  130.0000   -1.5606    0.2655   127.0923     0.0371    N/A
  1  130.0000  127.0923    0.2655    0.0371   126.6197    -0.0010    0.0037
```

```
MODIFIED SECANT METHOD:
-------------------------
  n       x_n     x_n+1    f(x_n)   f(x_n+1)    error
  0  110.0000  124.5077   -1.5606    -0.1752   N/A
  1  124.5077  126.6171   -0.1752    -0.0012   0.0167
  2  126.6171  126.6326   -0.0012     0.0000   0.0001
```



The root of this function is 126.632. I choose

- a_n = 110 and b_n = 130 for bisection and false-position method,
- x_n = 110 for Newton-Raphson
- x_{n-1} = 110 and x_n = 130 for Secant
- x_n = 110 for modified-secant method.

Similar to the function a, bisection method in this function converges slowly. But it only takes 3 iterations to satisfy the error < 1%.
For this function, I choose a_n = 110 and b_n = 130 and false-position method only takes 1 iteration to satisfy the error condition. The error is `0.0031 < 0.01` at the first iteration.
Newton-Raphson method takes 3 iterations for this function as same as bisection method. However, we can see that, newton-raphson method converges a lot faster than bisection method. It is not accurate to say that because there is no general convergence criterion for newton-raphson method.
Secant method in this function is similar to false-position method. It stops right at the first iteration.
For the modified-secant method, it takes 2 iterations. The error of the first iteration is pretty close to the error condition. Because it converges rapidly so it satisfies after the second iteration.
Only newton-raphson and modified secant method get the exact root while the error is satisfied.

For the data type, I use double for all methods and starting point variables. I only use int for counter variable.