

# Root Approximation Methods using GeoGebra

***Bhavik Dodda***

***Department of Mathematics,***

***Sardar Vallabhbhai National Institute of Technology, Surat, Gujarat***

## 1. Aim

Finding the roots of a function  $f(x)$  is straightforward in certain cases, such as polynomials of degree 0 to 4 or a few specific variations. However, for most functions, solving  $f(x) = 0$  exactly is not feasible. In such scenarios, root approximation techniques become the most effective and practical approach to finding solutions.

The four primary root approximation techniques are *Bisection*, *Regula-Falsi*, *Secant*, and *Newton's Method*. These iterative methods provide increasingly accurate approximations to the roots with each new iteration. We can conclude this computationally. Visualizing these methods graphically allows users to gain a clearer understanding of why and how the methods work. The approximations converge to the roots with orders of convergence 1, 1, the golden ratio ( $\phi$ ), and 2, respectively. The order of convergence of a method is said to be  $p$  if every subsequent error of iteration  $e_{n+1} = k \cdot e_n^p$ , where  $e_n = |\text{approximation of root in the } n\text{th iteration} - \text{actual root}|$ . Although these orders ( $p$ ) can be mathematically derived, a visual representation offers a more intuitive grasp of their workings.

## 2. Introduction

The primary goal of this project is to analyze the following root approximation techniques:

1. Bisection Method
2. Regula-Falsi Method
3. Newton-Raphson Method
4. Secant Method

This project utilizes GeoGebra to create an interactive visualization of these techniques, enabling users to explore, observe, and better understand the principles behind the four root approximation methods!

## 3. Theory

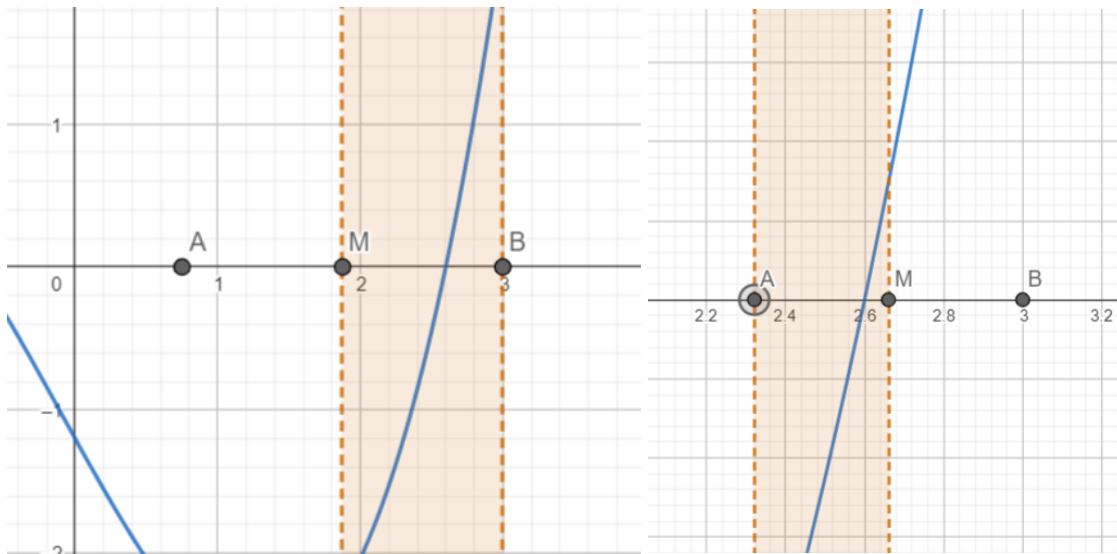
Goal: To find the roots of  $f(x)$ .

These methods are iterative and rely on approximations to converge to the actual solution.

### *Bisection Method*

1. Before we iterate, the initial configuration is an interval  $[A, B]$  such that the root lies between  $A$  and  $B$ .
2. The method divides the interval into two halves, creating two subintervals  $[A, M]$ ,  $[M, B]$  where  $M$  is the midpoint of  $AB$ .

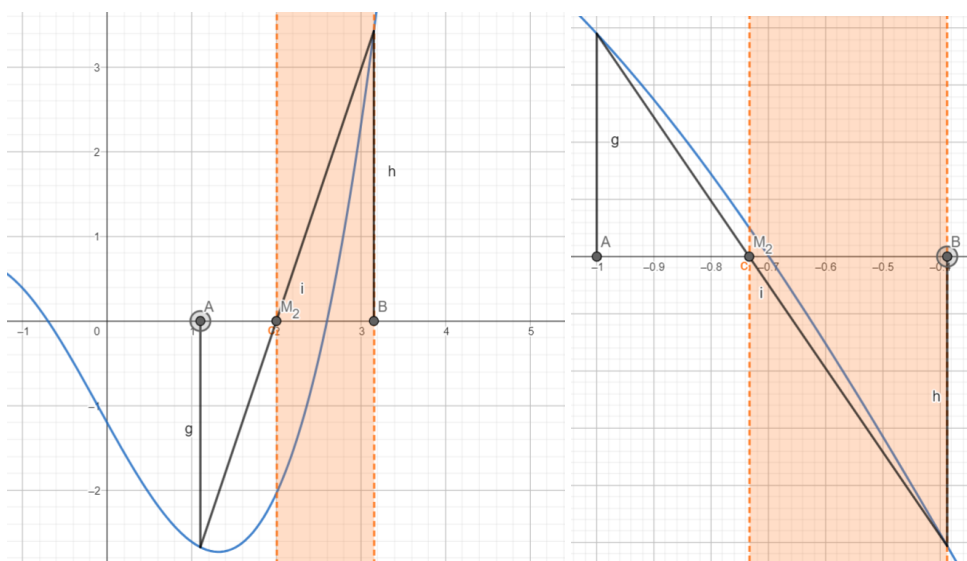
3. In every iteration, we choose the subinterval that contains the root, and only the end-points of the interval are updated based on which subinterval is chosen.
4. The approximate value of root is the midpoint of the interval in each iteration.



The orange interval is the one selected for the next iteration.

### *Regula-Falsi Method*

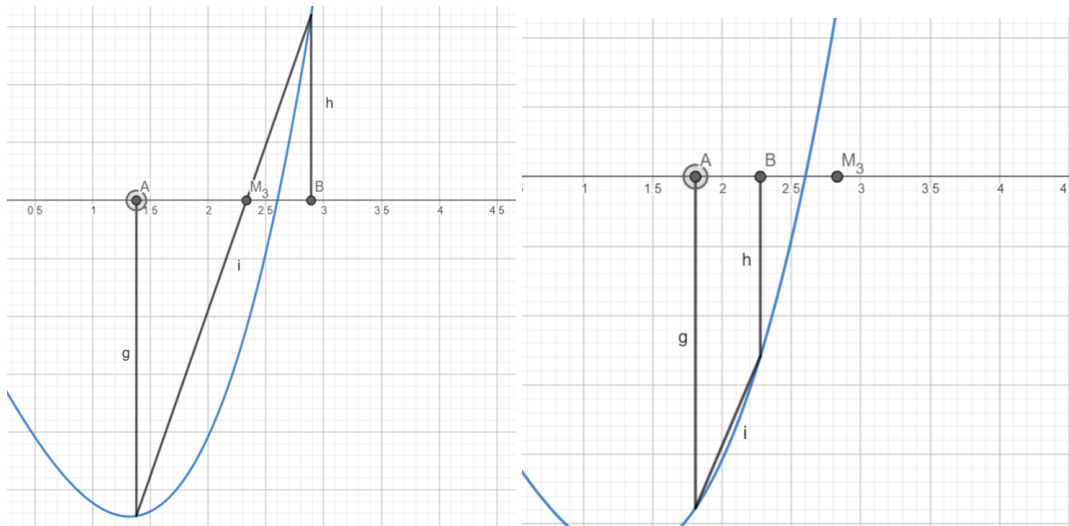
1. Before we iterate, the initial configuration is an interval  $[A, B]$  such that the root lies between A and B.
2. Let  $M_2$  be the point that intersects the secant between A,B on the function and the x-axis.
3. We divide the interval into 2 subintervals  $[A, M_2]$ ,  $[M_2, B]$
4. In every iteration, we choose the subinterval that contains the root, and only the end-points of the interval are updated based on which subinterval is chosen.
5. The approximate value of root is the point intersecting the x-axis and secant between interval's endpoints:  $M_2$



The orange interval is the one selected for the next iteration.

*Secant Method*

1. Before we iterate, the initial configuration could be an interval  $[A, B]$  with any end-points.
2. Let  $M_3$  be the point that intersects the secant between  $A, B$  on the function and the  $x$ -axis.
3.  $A$  is updated to  $B$  and  $B$  is updated to  $M_3$  in each iteration. It is therefore not necessary for  $A < B$  in all iterations.
4. The approximate value of root in each iteration is  $M_3$ .



Here, there are no subintervals to check where the root lies in, we always update  $A, B$  to  $B$  and  $M_3$  respectively.

*Newton's Method*

1. The initial configuration consists of only a single number  $A$ , which could be any initial "guess" of the root.
2. We draw a tangent line at  $(A, f(A))$  on the graph. Let  $M_4$  be the intersection between tangent and  $x$ -axis.
3.  $A$  is updated to  $M_4$  in each iteration.
4. The approximate value of root in each iteration is  $M_4$ .

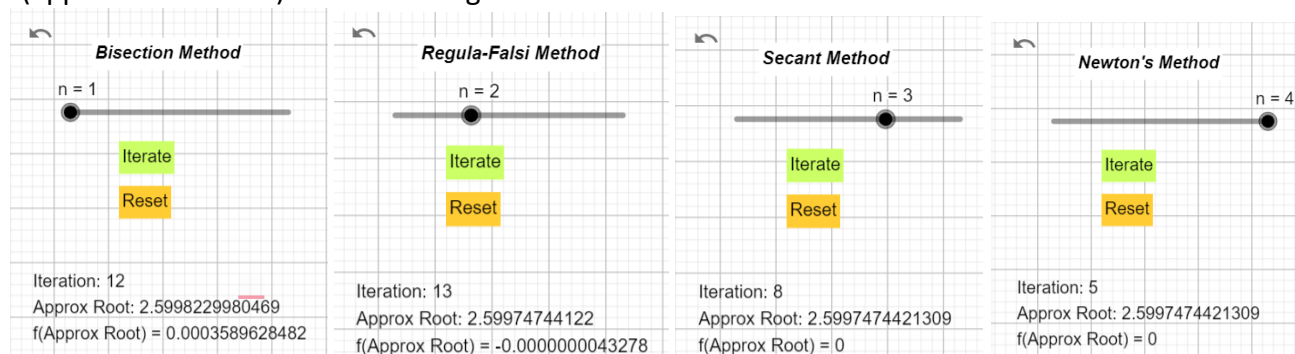


## 4. Usage

An end user of the interactive graph (rootApprox.ggb) would have control over the following parameters:

- Set the function that they want to find the roots of:  $f(x)$
- Set the values of  $A\_0$ , and  $B\_0$  which would serve as the initial configuration of what the interval would be before the iterations.
- Set the method used for iteration using the parameter 'n'.
  - o  $n=1$ : Bisection Method
  - o  $n=2$ : Regula-Falsi Method
  - o  $n=3$ : Secant Method
  - o  $n=4$ : Newton's Method
- There are 2 buttons
  - o Reset: To set the initial configuration
  - o Iterate: To run the iteration based on the method selected.

The graph displays the number of iterations  $N$ , the approximated root at the  $N$ th iteration and the  $f(\text{approximated root})$  which converges to 0 as the number of iterations increase.



## 5. Methodology

Please refer to the attached video rootApprox.mp4

- i. Define the function  $f(x)$  such that it has at least 1 root.
- ii. Set  $A\_0$  and  $B\_0$  appropriately, to cover the root. These are the initial end-points of the interval.
- iii. Define  $C, D$  for the bounds of interval. These variables would be updated with every iteration.
- iv. Define  $N$  to be the number of iterations.
- v. Define a slider for variable 'n' which can take on values 1,2,3,4 only and make sure to pin it to the screen. Throughout the graph, we create objects which are displayed according to a specific 'n' value. This is done using: **If( $n==...$ , object)**
- vi. Add 2 text elements, Reset and Iterate and make them as "buttons" by attaching a javascript program in the "Scripting" tab, which will run when clicking them.
  - a. Define basic functions setVal, getVal, setList, getList to fetch and update value of variables.

- b. On clicking the Reset button, the following events are executed-
  - i.  $C \rightarrow A\_0$
  - ii.  $D \rightarrow B\_0$
  - iii.  $N \rightarrow 0$
- c. On clicking the Iteration button, the following events are executed
  - i.  $C \rightarrow C\_next$
  - ii.  $D \rightarrow D\_next$
  - iii.  $N \rightarrow N+1$
  - iv.  $C\_next$  and  $D\_next$  are what the bounds will be updated to. This will depend on the method of root approximation chosen using variable  $n$ . In the following section, we'll derive what they should be for  $n=1,2,3,4$ .
- d.  $C\_next$  and  $D\_next$  for Bisection Method ( $n=1$ )
  - i. Midpoint of interval  $[C,D]$  is defined with  $M=(C+D)/2$
  - ii. Figuring out which subinterval  $[C,M]$  or  $[M,D]$  to choose for the next interval:
    - 1.  $f(C)f(M)<0$  implies that  $f(C),f(M)$  have opposite sign and therefore the root lies in subinterval  $[C,M]$ . In this case,  $C\_next=C$  and  $D\_next=M$
    - 2.  $f(D)f(M)<0$  implies that  $f(D),f(M)$  have opposite sign and therefore the root lies in subinterval  $[M,D]$ . In this case,  $C\_next=M$  and  $D\_next=D$
  - iii. Additionally display what interval will be chosen in the next iteration for clarity using: If  $(f(C) f(M)<0, C<x<M, M<x<D)$
- e.  $C\_next$  and  $D\_next$  for Regula-Falsi Method ( $n=2$ )
  - i. Finding the intersection  $M\_2$  between x-axis and secant between  $(C,f(C))$  and  $(D,f(D))$ :
    - 1. Slope of secant = Slope between  $(C,f(C))$  and  $(M\_2,0)$
    - 2.  $(f(D)-f(C))/(D-C) = -f(C)/(M\_2-C)$
    - 3. Solving for  $M\_2$ , we get  $M\_2 = -f(C)[(D-C)/(f(D)-f(C))] + C$
  - ii. Next, we choose the subinterval  $[C,M\_2]$  or  $[M,D\_2]$  based on where the root lies in, using the same method as the Bisection Method.
    - 1. If  $f(C)f(M)<0$ ,  $C\_next=C$  and  $D\_next=M\_2$
    - 2. If  $f(D)f(M)<0$ ,  $C\_next=M\_2$  and  $D\_next=D$
- f.  $C\_next$  and  $D\_next$  for Secant Method ( $n=3$ )
  - i. The intersection  $M\_3$  between x-axis and secant between  $(C,f(C))$  and  $(D,f(D))$  is  $-f(C)[(D-C)/(f(D)-f(C))] + C$  derived in point e.
  - ii. In this method, we don't selectively choose a subinterval but do:  $C\_next = D$ ,  $D\_next = M\_3$
- g. Newton's Method ( $n=4$ )
  - i. In this method we need only 1 variable to update during iterations. In the graph, I've chosen to update  $D$  with  $D\_next$ , but we can use either variable.
  - ii. Let  $M\_4$  be the intersection between the x-axis and tangent line at  $(D,f(D))$ .
    - 1. Slope of tangent = Slope between  $(D,f(D))$  and  $(M\_4,0)$
    - 2.  $f'(D) = -f(D)/(M\_4-D)$
    - 3.  $M\_4 = D - f(D)/f'(D)$
  - iii.  $D\_next = M\_4$
- vii. Since  $C\_next$  and  $D\_next$  are different for each of  $n=1,2,3,4$ , we define them separately and then ultimately define them using the If function:

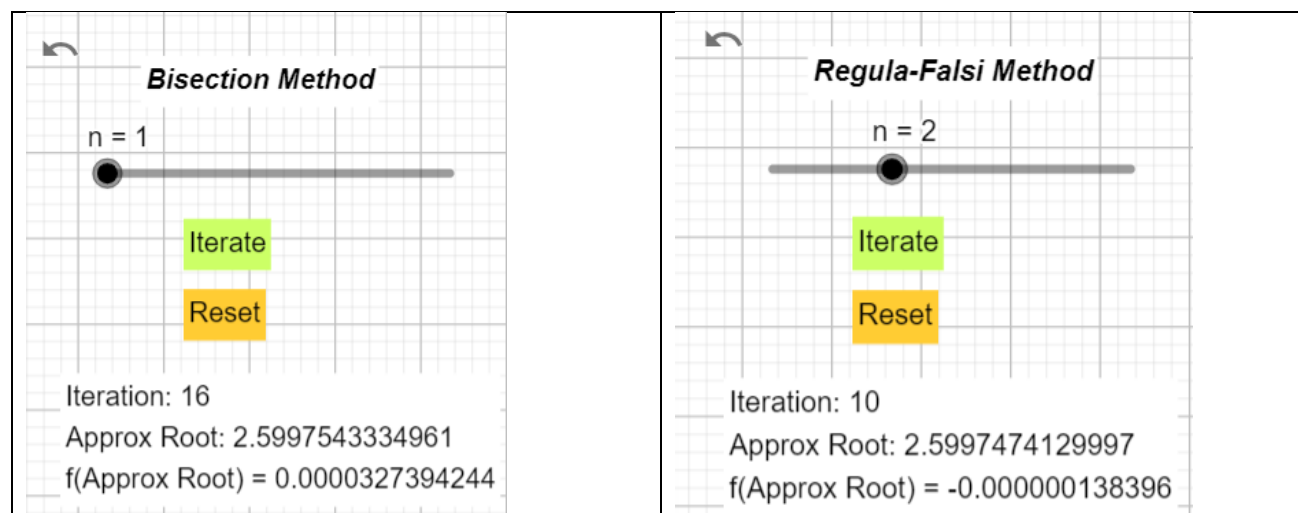
- a.  $C_{next} = \text{If}(n \geq 1, C_{\{bisection\}}, \text{If}(n \geq 2, C_{\{falsi\}}, \text{If}(n \geq 3, C_{\{secant\}}, C)))$
- b.  $D_{next} = \text{If}(n \geq 1, D_{\{bisection\}}, \text{If}(n \geq 2, D_{\{falsi\}}, \text{If}(n \geq 3, D_{\{secant\}}, \text{If}(n \geq 4, M_{\{4\}}, D))))$
- viii. Analyzing rate of convergence of the iteration methods
  - a. Since we don't know what the exact root is prior to finding the convergence value, we don't have a measure of how close the approximations are to the actual root.
  - b. We can instead observe how close  $d=f(\text{approximated root})$  is from 0.
  - c. As the number of iterations  $N$  increases,  $d$  approaches 0.
  - d. Defining a variable 'Accuracy' which calculates the integer 'e' such that  $|d|$  is less than  $10^e$ . This is found to be  $e=\text{floor}(\log(10, \text{abs}(d)))+1$
  - e. For each method, we can now keep a record of how many iterations are required for the value of  $d$  to be less than  $10^0, 10^{-1}, 10^{-2}, 10^{-3}, \dots$  etc. That is, we need to find the threshold minimum number of iterations to make  $e=0, -1, -2, -3$  etc. This will help analyze the rate at which the approximated root approaches the real root.
  - f. This record is stored within the **iterations** list variable.
    - i. It is updated using javascript included within the Scripts tab and executed when the Iterate text is clicked.
    - ii. It calculates the value of 'e' and appends to  $N$  to the iterations list whenever 'e' updates.
  - g. Interpretation: If the iterations list contains smaller integers, it means that few number of iterations were required to make  $d < 1, 10^{-1}, 10^{-2}$ , etc. This would therefore imply that the rate of convergence is faster.

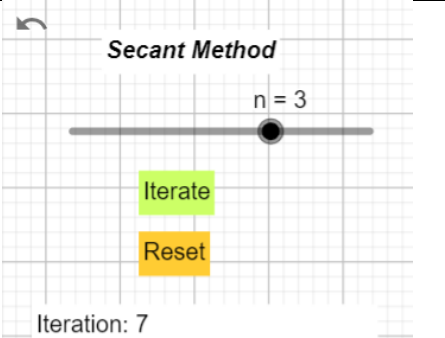
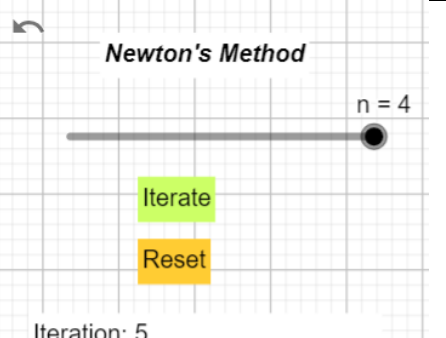
## 6. Observations

These might vary depending on the function and the initial condition, but the conclusions still hold valid for other functions and initial conditions.

Example: for function  $f(x) = 0.3(x^3 + 0.3x^2 - 6x - 4)$  and initial condition  $A_0 = 0.75, B_0 = 3$ ,

Bisection Method	Regula-Falsi Method
iterations = {1, 5, 8, 11, 14}	iterations = {1, 3, 4, 5, 7, 8, 9}
Accuracy = -4	Accuracy = -6
After N=1 iteration, $d < 10^0$	After N=1 iteration, $d < 10^0$
After N=5 iterations, $d < 10^{-1}$	After N=3 iterations, $d < 10^{-1}$
After N=8 iterations, $d < 10^{-2}$	After N=4 iterations, $d < 10^{-2}$
After N=11 iterations, $d < 10^{-3}$	After N=5 iterations, $d < 10^{-3}$
After N=14 iterations, $d < 10^{-4}$	After N=7 iterations, $d < 10^{-4}$



Secant Method	Newton's Method
iterations = {1, 3, 4, 0, 0, 5, 0, 0, 0, 6, 0, 0, 0, 0, 7}	iterations = {0, 0, 1, 0, 0, 2, 0, 0, 0, 0, 0, 3, 0, 0, 0, 4}
Accuracy = -14	Accuracy = -15
After N=1 iteration, $d < 10^0$	After N=1 iteration, $d < 10^{-2}$
After N=3 iterations, $d < 10^{-1}$	After N=2 iterations, $d < 10^{-5}$
After N=4 iterations, $d < 10^{-2}$	After N=3 iterations, $d < 10^{-11}$
After N=5 iterations, $d < 10^{-5}$	After N=4 iterations, $d < 10^{-15}$
After N=6 iterations, $d < 10^{-9}$	..
After N=7 iterations, $d < 10^{-14}$	...
 <p>Iteration: 7 Approx Root: 2.5997474421309 f(Approx Root) = 0</p>	 <p>Iteration: 5 Approx Root: 2.5997474421309 f(Approx Root) = 0</p>

## 7. Conclusion

Testing using the iteration list with various initial configurations and functions, we can conclude that the 4 iteration techniques can be ranked in increasing order of convergence as follows:

Bisection Method < Regula-Falsi Method < Secant Method < Newton's Method

The interactive tool also allows the user to better understand visually how the iteration algorithms for the 4 methods work in solving for the roots of a function!