Introduction:

In numerical analysis, finding the roots of equations is a fundamental problem. Many real-world problems in science and engineering lead to non-linear equations, which often cannot be solved analytically. Hence, numerical methods like the Bracketing Methods are used.

Linear Equation: An equation in which the highest power (degree) of the variable is 1.Example:($3x - 2 = 0$).

Non-Linear Equation: An equation in which the variable appears with power greater than 1 or in non-linear forms such as exponential, trigonometric, logarithmic.Example:($e^x - 3x = 0$), ($x^2 - 4x - 10 = 0$).

Bracketing Methods:

Bracketing methods are a class of numerical techniques used to locate the root of a nonlinear equation by assuming the root is within a closed interval. The process begins by selecting two initial points, a and b, such that:[$f(a).f(b) < 0$]

This condition indicates that the function values at the two points a and b have opposite signs, which implies that at least one real root must lie within the interval [a, b] provided that the function f(x) is continuous in that region.

Once such an interval is found, bracketing methods use a systematic procedure to gradually shrink the width of the interval, thereby narrowing down the location of the root and ultimately converging to the root.

There are two types of Bracketing Methods:

1.Bisection Method

2.False Position (Regula Falsi) Method

Bisection Method:

The Bisection Method is a simple and dynamic numerical technique used to find roots of continuous functions. It is also known as:

1.Binary chopping method

2.Half-interval method

Algorithm of Bisection Method:

Step 1: Choosing two numbers (x1) and (x2) such that:
[f(x1) f(x2) < 0]

Step 2: Computing the midpoint:
[x0 = (x1 + x2)/2]

Step 3: Evaluating f(x0).

Step 4:
If (f(x0) = 0), then (x0) is the exact root.
If (f(x0) .f(x1) < 0), setting (x2 = x0).
If (f(x0) .f(x2) < 0), setting (x1 = x0).

Step 5: Repeating until the stopping criterion is met:
|x2 - x1| < E

Step 6: Stopping. The approximate root is (x0).

Pseudocode:

```
BisectionMethod(f, a, b, tolerance, max_iter):

   Step 1: Check if: f(a).f(b) < 0
         If not, print "Invalid interval"and stop.
   Step 2: For i = 1 to max_iter:
            x0 = (a + b) / 2
            fx0 = f(x0)
            If |fx0| < tolerance:
                  Return x0 as the root
            Else If f(a). fx0 < 0:
                  b = x0
            Else:
                  a = x0
            End For

   Step 3: Return (a + b) / 2 as the approximate root
```

Convergence of the Bisection Method:

At every iteration of the Bisection Method, the interval containing the root is reduced to half of its previous size.If the initial interval is [x1, x2] then after n iterations the interval width becomes: [(x2 - x1) / 2^n].Therefore, the root is guaranteed to lie within an interval of width [± (Δx / 2^n)],where [Δx = x2 - x1] is the initial interval length.Hence, the error after n iterations is: [En = Δx / 2^n]. The error in the next iteration satisfies: [En+1 = En / 2].

This relationship shows that the error decreases by a factor of 2 at each step, meaning the Bisection Method shows linear convergence.

Examples:

Problem 1: Find root using Bisection Method
Function:[f(x) = 3x - cos x - 1]

Table 1:Iteration Table for Problem 1

Iteration	x1	x2	x0=(x1+x2)/2	f(x0)	New Interval
1	0.0	1.0	0.5	-0.3776	[0.5, 1.0]
2	0.5	1.0	0.75	0.5183	[0.5, 0.75]
3	0.5	0.75	0.625	0.0700	[0.5, 0.625]
4	0.5	0.625	0.5625	-0.1585	[0.5625,0.625]
5	0.5625	0.625	0.59375	-0.0440	[0.59375,0.625]
6	0.59375	0.625	0.609375	0.0131	[0.59375,0.609375]

Approximate root is 0.60.

Problem 2: Compute root using Bisection Method
Function:[f(x) = x^2 - 4x - 10]

Table 2: Iteration Table for Problem 2

Iteration	x1	x2	x0=(x1+x2)/2	f(x0)	New Interval
1	5.0	6.0	5.5	-1.75	[5.5, 6.0]
2	5.5	6.0	5.75	0.0625	[5.5, 5.75]
3	5.5	5.75	5.625	-0.8594	[5.625, 5.75]
4	5.625	5.75	5.6875	-0.4023	[5.6875, 5.75]
5	5.6875	5.75	5.71875	-0.1716	[5.71875, 5.75]
6	5.71875	5.75	5.734375	0.0546	[5.734375, 5.75]

Approximate root is 5.74

Importance:

The bisection method is frequently used in numerical analysis because,It always converges, unlike methods that depend on derivatives.It is simple to implement.It requires no complicated mathematical computations beyond evaluating the function and halving intervals.It works reliably even for complex or irregular functions.The error reduces linearly, giving predictable convergence.

Limitations:

Although reliable, the bisection method has limitations.
It has slow convergence.Requires a sign change.Cannot start without bracketing the root.Not suitable for repeated or very close roots as Interval shrinking becomes inefficient.Fails if discontinuities exist inside the interval.

Practical Applications:

The bisection method is commonly used in:
1.Solving non-linear algebraic equations.
2.Circuit analysis
3.Mechanical engineering problems
4.Computer graphics
5.Control systems

False Position Method:

Algorithm of False Position Method:

Step 1: Choosing two numbers x1 and x2 such that:
f(x1).f(x2)<0
Step 2: Compute the root approximation:
x0=x1-(f(x1).(x2−x1))/(f(x2)-f(x1))
Step 3: Evaluate f(x0).
Step 4:
If f(x0)=0, then x0 is the exact root.
If f(x0).f(x1)<0 then, x2=x0
If f(x0).f(x2)<0 then, x1=x0
Step 5: Repeat until the stopping criterion is met:
|x2−x1|<E
Step 6: Stop. The approximate root is x0

Pseudocode:

```
FalsePositionMethod(f, a, b, tolerance, max_iter):

   Step 1: Check if f(a) . f(b) < 0
         If not, print "Invalid interval"and stop.
   Step 2: For i = 1 to max_iter:
            x0 = a - f(a) * (b - a) / (f(b) - f(a))
            fx0 = f(x0)
            If |fx0| < tolerance:
               Return x0 as the root
            Else If f(a) . fx0 < 0:
               b = x0
            Else:
               a = x0
   Step 3: Return x0 as the approximate root
```

Convergence of the False Position Method:

Unlike the Bisection Method, the interval is not halved, the root is approximated based on linear interpolation between f(x1) and f(x2).Convergence may be faster than Bisection for many functions.

Examples:

Problem 1: Find root using False Position Method

Function: f(x)=3x-cosx-1.

Table 1: Iteration Table for Problem 1

Iteration	x1	x2	x0=x1-(f(x1).(x2-x1))/(f(x2)-f(x1))	f(x0)	New Interval
1	0.0	1.0	0.3333	-0.667	[0.3333,1.0]
2	0.3333	1.0	0.3668	-0.534	[0.3668,1.0]
3	0.3668	1.0	0.3679	0.0001	[0.3668,0.3679]

Approximate root: 0.3679

Problem 2: Find root using False Position Method

Function: f(x)=x2−4x−10

Table 2: Iteration Table for Problem 2

Iteration	x1	x2	x0=x1-f(x1)*(x2-x1)/(f(x2)-f(x1))	f(x0)	New Interval
1	5.0	6.0	5.5263	-0.7184	[5.5263,6.0]
2	5.5263	6.0	5.7340	0.7826	[5.5263,5.7340]
3	5.5263	5.734	5.6923	0.0107	[5.5263,5.6923]

Approximate root: 5.692

Importance:

The False Position method is widely used in numerical analysis because,It often converges faster than Bisection.It does not require derivatives.It is simple to implement and reliable for continuous functions.Works well even for complex or irregular functions.

Limitations:

Convergence may stall if one endpoint remains fixed for many iterations.Requires a sign change, cannot start without bracketing the root.Less predictable error reduction compared to Bisection.Inefficient for highly skewed functions or repeated roots.

Practical Applications:

1. Solving non-linear algebraic equations.
2. Circuit analysis
3. Mechanical engineering problems
4. Fluid mechanics and flow equations
5. Control systems and stability analysis