# **Bisection Method**

**Bisection method** is a numerical technique used to find the roots of a continuous function. It is based on the intermediate value theorem, which states that if a function f(x) changes sign over an interval [a, b], then there exists at least one root c within [a, b] such that f(c)=0. The method iteratively halves the interval by evaluating f(x) at the midpoint, narrowing down the subinterval where the root lies, until a desired level of accuracy is achieved. It is simple, reliable, and works well if the function is continuous and the initial interval is chosen The correctly.

C =

Algorithm:

1.Calculate the midpoint:

C =

2. Evaluate f(c):

* If f(c) =0, c is the root, and the process stops.
* Otherwise, proceed to the next step.

3. Determine the new interval:

* If f(c)⋅f(a)<0, the root lies in [a, c], so set b=c.
* If f(c)⋅f(b)<0, the root lies in [c, b], so set a=c.

4.Repeat steps 1–3 until ∣b−a∣<ϵ .

Procedure, Advantages, Disadvantages & Bisection Method ...

Bisection Method Procedure To solve bisection method problems, given below is the step-by-step explanation of the working of the b...

**Error! Filename not specified.**

Vedantu

Procedure, Advantages, Disadvantages & Bisection Method ... For equation : f(x)=x3-x-2

Sample input: a=1, b=2 , tolerance =.001

Sample output: Iteration 1: c = 1.5, f(c) = -0.125

Iteration 2: c = 1.75, f(c) = 1.60938

Iteration 3: c = 1.625, f(c) = 0.666016

The approximate root is: 1.52148

Bisection Method Procedure To solve bisection method problems, given below is the step-by-step explanation of the working of the b...

**Error! Filename not specified.**

Vedantu

Bisection Method Procedure To solve bisection method problems, given below is the step-by-step explanation of the working of the b...

**Error! Filename not specified.**

Vedantu

**False Position Method**

The **false position method** (or **regula falsi method**) is a root-finding technique that improves upon the bisection method by interpolating the root more efficiently. It is used to find a root of a continuous function f(x) over an interval [a, b] where f(a)⋅f(b)<0. Unlike the bisection method, which uses the midpoint of the interval, the false position method uses a weighted interpolation formula to estimate the root.

C = b -

Algoritms:

1. Begin with an interval [a, b] such that f(a)⋅f(b)<0.

2. Compute c using the formula above.

3. Evaluate f(c):

If f(c)=0, c is the root.

4.Otherwise, update the interval:

* If f(a)⋅f(c)<0, set b=c.
* If f(b)⋅f(c)<0, set a=c.

5. Repeat the process until the desired tolerance is achieved.

Sample input : a=1, b=2 , tolerance =.001

Sample output: Iteration 1: c = 1.33333, f(c) = -0.962963

Iteration 2: c = 1.46269, f(c) = -0.333338

Iteration 3: c = 1.50402, f(c) = -0.046892

Iteration 4: c = 1.50804, f(c) = -0.000511

The approximate root is: 1.50804

**Cramer’s Rule**

Cramer’s Rule is a method for solving a system of linear equations using determinants. It is applicable when the system has n equations and n unknowns, and the coefficient matrix has a non-zero determinant.

### **Formula:**

Given a system of linear equations:

a11x1+a12x2+a13x3=b1

a21x1+a22x2+a23x3=b2

a31x1+a3x2+a33x3=b3

Represent the system in matrix form:

Ax=b

Cramer’s Rule states :

Xi = where i = 1,2,3

det(A): Determinant of the coefficient matrix A.

Ai​: Matrix obtained by replacing the i-th column of A with the constants column b.

### **Algorithm**

* Coefficient matrix A (size n×n ).
* Constants column b.
* Compute det(A).
* For each variable xi ​:
  + Form Ai by replacing the i-th column of A with b.
  + Compute det(Ai) .
* If det(A)=0, the system has no unique solution (either inconsistent or infinitely many solutions).**:**
* For each i: Xi =

Sample input :

2 1 -1 8

-3 -1 2 -11

-2 1 2 -3

Sample output: x1=1

x2=-2

x3 =-3

### **Secant Method**

The Secant Method is an iterative numerical technique used to find the root of a real-valued function. Unlike the Newton-Raphson method, it does not require the derivative of the function but uses two initial guesses to approximate the root. The iterative formula is:

Xn+1=Xn+

This method is especially useful when the derivative of the function is difficult or expensive to compute.

Algorithm :

1.Input :

* + - Function is f(x).
    - Derivative is f’(x).
    - Initial guess x0=0.
    - Tolarence is e.
    - Iteration number n.

2.Iteration:

* + - While n<Nmax
    - Calculate Xn+1
    - If absolute value of (Xn+1-Xn) < e
    - Then stop and return Xn+1 as root.
    - Else update Xn=Xn+1, Xn-1=Xn , n by 1.

3.Output :

* + - If convergence criteria are met, return xn+1.
    - Otherwise, after Nmax ​ iterations, report failure to converge.

Sample Input :

* Initial guess X0=1,X1=2.
* Tolerance (e): 0.0001
* Maximum iterations (Nmax): 100

Sample Output :

Root: 1.4142

Number of iterations: 6

**Gauss-Seidel Method**

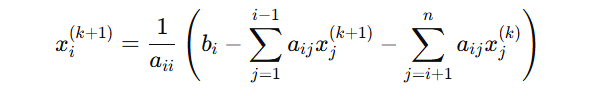
The Gauss-Seidel method is an iterative technique for solving a system of linear equations of the form:

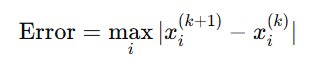
Ax=b

Here, A is a coefficient matrix, x is the vector of unknowns, and b is the constant vector. It refines the solution by solving each equation for one variable at a time and updating the solution immediately.

Algorithms:

1.Input:

* + - * Coefficient matrix A.
      * Constant vector .
      * Initial guess vector x0 = 0.
      * Tolarence is e.
      * Maximum iteration number N.
* 2.Iteration:
  + - * Set n= 0.
      * While n < N max
      * Calculate

* + - * If value of Error < e 
      * Then stop and return Xk+1as solution.
      * Else update Xk=Xk+1, n by 1.

3.Output :

* + - * If convergence criteria are met, return xK+1.
      * Otherwise, after Nmax ​ iterations, report failure to converge.

Sample Input :

### **System of Equations**

2x – 2y + 3z = 20  
x + 3y – z = 12  
3x – y + 4z = 22

e = .001

N = 100

Sample Output :

x1​=21

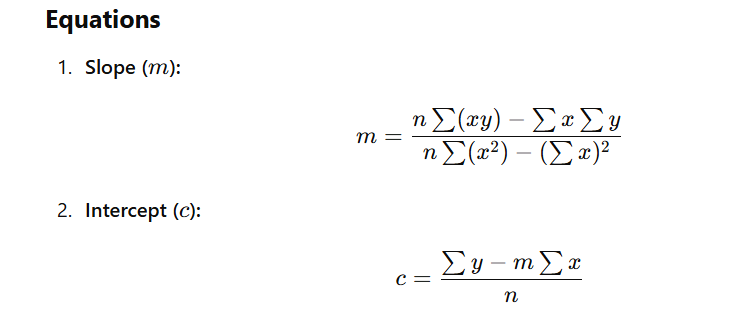
x2=-7

x3​=12 Number of iterations: 7

**Least Squares Regression**

The Least Squares Regression method is a statistical technique used to find the best-fitting line (or curve) through a set of data points by minimizing the sum of the squared differences (errors) between the observed values and the values predicted by the line.

For a linear relationship y=mx+c, the least squares method determines the slope mmm and intercept ccc of the line.



Where n is the number of data points .

x,y are the data points.

**Algorithm :**

1.Input:

* Enter the range n of data points.
* Input x and y values for n points.

2.Calculate Sums:

* Compute , , , ^2 using a loop.

3.Find Coefficients:

* Compute **b =**
* Compute **A =**

**a =**

4.Output Results:

* Print the values of a and b.

**Sample Input:**

Enter the Range of x and y:  
5  
Enter the input of X And Y:  
1 0.5  
2 2  
3 4.5  
4 8  
5 12.5

**Sample Output:**

The value of a :0.5

The value of b :2

**Fitting a Transcendental Function**

Fitting transcendental equations involves approximating data using equations with functions such as **exponential**, **logarithmic**, or **trigonometric forms**. The method often transforms the equation into a linear form, applies the **least squares method** to determine the unknown constants, and then converts the results back to the original form of the equation.

y=aebx

ln(y) = ln(a) + bx

y = a + bx

**b =**

**A =**

**Algorithm :**

1.Input:

* Enter the range n of data points.
* Input x and y values for n points.

2.Calculate Sums:

* Compute , , , ^2 using a loop.

3.Find Coefficients:

* Compute **b =**
* Compute **A =**

**a =**

4.Output Results:

* Print the values of a and b.

**Sample Input:**

Enter the Range of x and y:  
5  
Enter the input of X And Y:  
1 0.5  
2 2  
3 4.5  
4 8  
5 12.5

**Sample Output:**

The value of a :0.5

The value of b :1.99

**Euler's Method**

Euler's method is a straightforward numerical technique used to approximate the solution of an ordinary differential equation (ODE) of the form:

dx/dy​ = f(x,y)

The method starts with an initial value y(x0)= y0 ​ and iteratively calculates the values of y at subsequent points. It uses a simple formula to find the next value of y based on the current value and the slope of the function.

The iterative formula is:

yn+1=yn+h⋅f(xn,yn)

Where:

* h is the step size (the distance between the current and the next point).
* Yn​ is the current value of y.
* f(xn,yn)is the value of the function at (xn,yn).
* yn+1​ is the next value of y.

**Algorithm :**

1.Input:

Read , , step size h, and ​.

2.Initialize:

Set x = ​, y = ​, and compute n = .

3.Iterate:

For i=0 to n−1:

Calculate yn+1=yn+h⋅f(xn,yn)

Update yn = yn+1

Update xn+1 = xn + h.

Print x, y.

4.End:

Show the value of x,y.

**Sample Input:**

Enter the initial value of x and y:  
1 2  
Enter the step size:  
0.5  
Enter the end value of x:  
2

**Sample Output:**

X: 1 Y: 2

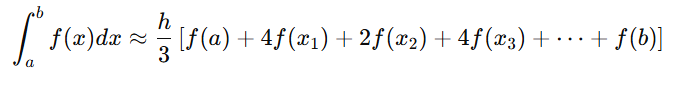
X: 1.5 Y: 4

X: 2 Y: 7.875

**Simpson’s 1/3 Rule**

Simpson’s 1/3 Rule is a numerical method used for approximating the definite integral of a function. It is particularly effective when the function can be approximated by a second-degree polynomial over small subintervals. The rule is derived from fitting a parabola through three consecutive points and integrating it.

The formula for the rule is:



Where:

* a and b are the lower and upper limits of integration.
* h=(b−a)/n ​ is the step size.
* n is the number of subintervals (must be even).
* x1,x2​… are the intermediate points in the range [a, b].

### **Algorithm for Simpson's 1/3 Rule**

#### **1. Input:**

* The function f(x).
* The limits of integration a and b.
* The number of subintervals n (must be even).

#### **2. Calculate Step Size:**

* Compute h=(b−a)/h.

#### **3. Compute the Integral:**

* Initialize the result: Result=f(a)+f(b)
* Add the contributions of the intermediate points:
  + For odd indices i: Result+=4⋅f(xi)
  + For even indices ii: Result+=2⋅f(xi).
* Multiply the entire result by h\3

#### **4. Output:**

* The approximate value of the definite integral.

**Sample Input:**

Enter the value of h:  
0.5  
Enter the range:(lower and higher value)  
1 1.5

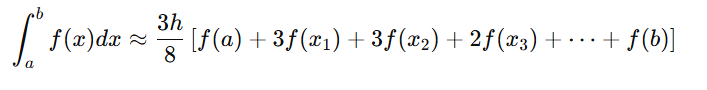
**Sample Output:**

Result of Simpsons 1/3:  
1.0625

**Simpson’s 3/8 Rule**

Simpson’s 3/8 Rule is another numerical method used to approximate the definite integral of a function. It is based on fitting a cubic polynomial to the function over three subintervals (four points). This rule is particularly useful when the number of subintervals nnn is a multiple of 3.

The formula for Simpson’s 3/8 Rule is:



Where:

* a and b are the lower and upper limits of integration.
* h=(b−a)/n ​ is the step size.
* n is the number of subintervals (must be multiple of 3).
* x1,x2​… are the intermediate points in the range [a, b].

### **Algorithm for Simpson's 3/8 Rule**

#### **1. Input:**

* The function f(x).
* The limits of integration a and b.
* The number of subintervals n (must be multiple of 3).

#### **2. Calculate Step Size:**

* Compute h=(b−a)/h.

#### **3. Compute the Integral:**

* Initialize the result: Result=f(a)+f(b)
* Add the contributions of the intermediate points:
  + For indices I divisible by 3 : Result+=2⋅f(xi)
  + For all other indices ii: Result+=3⋅f(xi).
* Multiply the entire result by 3h\8.

#### **4. Output:**

The approximate value of the definite integral.

**Sample Input:**

Enter the value of h:  
0.5  
Enter the range:(lower and higher value)  
1 1.5

**Sample Output:**

Result of Simpsons 3/8 :  
1.19531