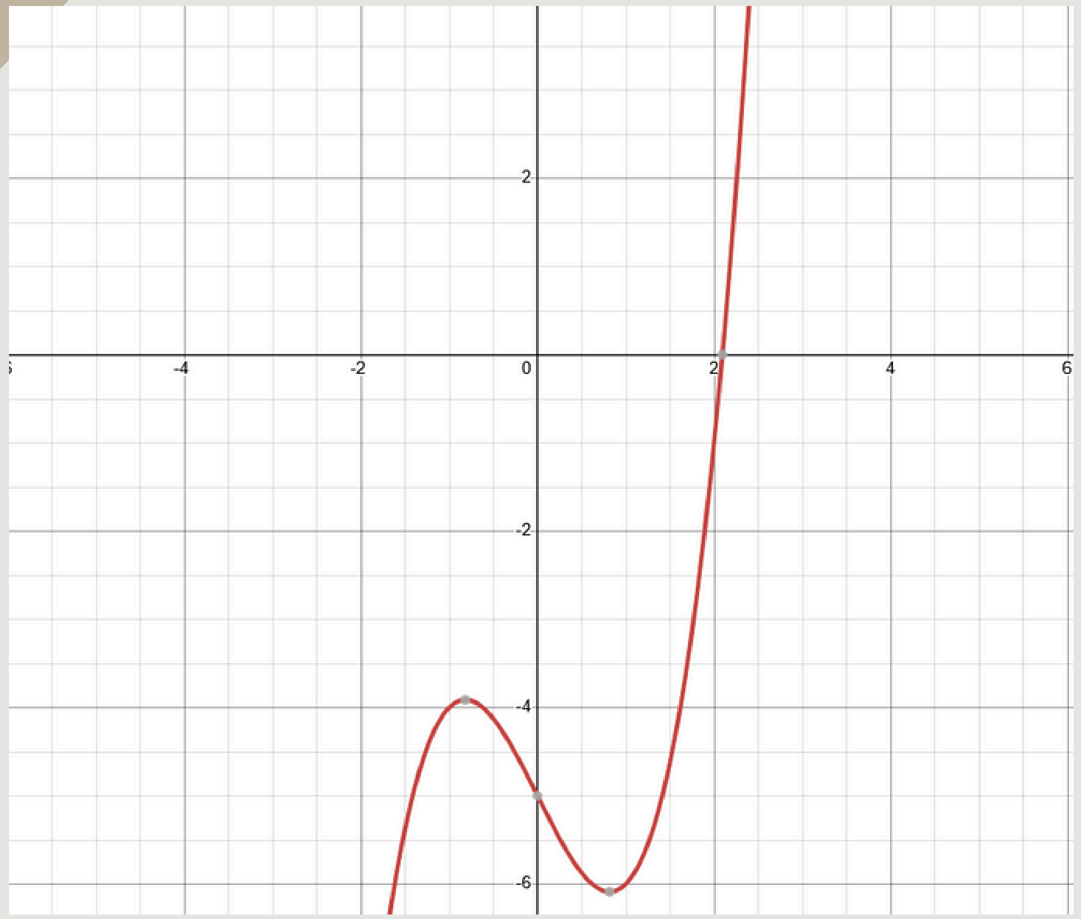
BISECTION METHOD & LAGRANGE'S INTERPOLATION METHOD

Presented by
Group CII
Mohammad Mostafizur Rahman
Sabbir Ahmed Talukdar
Abdullah Mohammed
MD.Ziad Hasan

BISECTION METHOD

MOHAMMAD MOSTAFIZUR RAHMAN
ABDULLAH MOHAMMED

BISECTION METHOD



- A numerical technique used to find the root of a continuous function within a given interval [a,b].
- If $f(x)$ is continuous on [a,b] & $f(a)$ & $f(b)$ have opposite signs, then there exists at least one root c in (a,b) such that $f(c)=0$.
- The method works by repeatedly dividing the interval in half and selecting the subinterval where the root lies.

BISECTION METHOD ALGORITHM

- Step 1: Choose an interval [a,b] such that $f(a) \cdot f(b) < 0$.
- Step 2: Compute the midpoint $c=(a+b) / 2$.
- Step 3: Check for convergence:
  - If $f(c)=0$ or the interval [a,b] is smaller than a predefined tolerance, c is the root.
- Step 4: Update the interval:
  - If $f(a) \cdot f(c) < 0$, the root lies in [a,c]. Set $b=c$.
  - If $f(b) \cdot f(c) < 0$, the root lies in [c,b]. Set $a=c$.
- Step 5: Repeat until the stopping criterion is met.

CODE SNNIPET

MAIN FUNCTION

```cpp
int main() {
    double a = 2.0;  // Left endpoint of the interval
    double b = 3.0;  // Right endpoint of the interval
    double tol = 1e-6;  // Tolerance for convergence

    double root = bisection(a, b, tol);

    if (!isnan(root)) {
        cout << "Root: " << root << endl;
    }

    return 0;
}
```

EQUATION FUNCTION

```cpp
double f(double x) {
    // Example function: f(x) = x^3 - 2x - 5
    return x * x * x - 2 * x - 5;
}
```

BISECTION FUNCTION

```cpp
double bisection(double a, double b, double tol) {
    if (f(a) * f(b) >= 0) {
        cout << "Error: The function must have opposite signs at a and b." << endl;
        return NAN;  // Return NaN if the interval is invalid
    }

    double c;
    while ((b - a) / 2 > tol) {  // Stopping condition
        c = (a + b) / 2;  // Compute midpoint

        if (f(c) == 0) {
            break;  // Check if c is the root
        }
        else if (f(a) * f(c) < 0) {  // Root lies in [a, c]
            b = c;
        }
        else {  // Root lies in [c, b]
            a = c;
        }
    }

    return (a + b) / 2;  // Return the approximate root
}
```

RESULT

- **Root Found**: Approximately **2.0945512**.
- **Number of Iterations**: **20 iterations** were required to achieve the desired accuracy.
- **Convergence**: The method converged reliably to the root, as expected for a continuous function with a sign change over the interval.

CONCLUSION

The Bisection Method is a simple and reliable numerical technique for finding roots of continuous functions.

It guarantees convergence as long as the function is continuous & the interval [a,b] contains a root.

The implementation demonstrates its effectiveness in approximating roots with high accuracy.

Langrange´s interpolation Method

SABBIR AHMED TALUKDAR
MD.ZIAD HASAN

THEORETICAL EXPLANATION

1
What is Lagrange's Interpolation Method?

A numerical method to estimate the value of a function given some known data points.

2
Key Idea:

Constructs a polynomial that passes through all the given points.

3
Importance:

Frequently used in numerical analysis for function approximation.

THEORETICAL BACKGROUND

Given a set of n+1 distinct data $(x_0, y_0), (x_1, y_1) \ldots (x_n, y_n)$ points where x_i are the independent variables and y_i are the corresponding dependent variables, the Lagrange interpolating polynomial P(x) is defined as:

$$P(x) = \sum_{i=0}^{n} y_i \cdot L_i(x)$$

where Li(x) are the Lagrange basis polynomials, given by:

$$L_i(x) = \prod_{j=0,\ j\neq i}^{n} \frac{(x - x_i)}{(x_i - x_j)}$$

The key property of the Lagrange basis polynomials is that

- $L_i(x_j) = 1 \text{ if } i=j$
- $L_i(x_j) = 0 \text{ if } i\neq j$

This ensures that the interpolating polynomial p(x) passes through all the given data points, i.e., $p(x_i)=y_i$ for all $i=0,1,\ldots,n$.

LANGRANGE INTERPOLATION FORMULA

Lagrange Interpolation Formula for nth Order

$$f(x) = \frac{(x-x_1)(x-x_2)\ldots(x-x_n)}{(x_0-x_1)(x_0-x_2)\ldots(x_0-x_n)} \times y_0 + \frac{(x-x_0)(x-x_2)\ldots(x-x_n)}{(x_1-x_0)(x_1-x_2)\ldots(x_1-x_n)} \times y_1 + \ldots + \frac{(x-x_0)(x-x_1)\ldots(x-x_n-1)}{(x_n-x_0)(x_n-x_1)\ldots(x_n-x_n-1)} \times y_n$$

Lagrange Interpolation Formula for First Order

$$f(x) = \frac{(x-x_1)}{(x_0-x_1)} \times y_0 + \frac{(x-x_0)}{(x_1-x_0)} \times y_1$$

Lagrange Interpolation Formula for Second Order

$$f(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} \times y_0 + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \times y_1 + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} \times y_2$$

ALGORITHM FOR LAGRANGE INTERPOLATION

```python
def lagrange_interpolation(x_values, y_values, x):
    result = 0

    n = len(x_values)

    for i in range(n): #i = 0 to n
        term = y_values[i]
        for j in range(n):
            if i != j:
                term *= (x-x_values[j])/(x_values[i]-x_values[j])
        result += term
    return result


x_values = [2, 5, 10, 12, 15]
y_values = [5, 15, 30, 25, 14]
x = 19

print(f"x is {x} and y is {lagrange_interpolation(x_values, y_values, x)}")
```

INPUT:

- x_values[]: List of known x-coordinates.
- y_values[]: Corresponding y-coordinates.
- x: The point at which interpolation is to be performed.

OUTPUT:

The interpolated value y at x.

STEP-BY-STEP EXPLANATION

1. Initialize result = 0 to store the final interpolated value.
2. Determine the number of points, n (length of x_values).
3. Loop over each i from 0 to n-1 to calculate Lagrange basis polynomial terms:
    - Set term = y_values[i] (start with y value of the current term).
    - Loop over each j from 0 to n-1 (except i itself) to compute the Lagrange basis polynomial:
        - Multiply term by ((x - x_values[j]) / (x_values[i] - x_values[j])).
    - Add the computed term to result.
4. Return result, which is the interpolated y value for the given x

CONCLUSION

01 Lagrange’s Interpolation is a powerful tool for estimating values.

02 Used in various applications like computer graphics, data fitting, and engineering problems.

It's versatile and widely used in various fields.

Thank You

For your attention