Department of
Computer Science and Engineering

University of Dhaka

Title: Interpolation Techniques: Lagrange Interpolation & Newton's Divided Difference Method

CSE 3212: Numerical Analysis Lab

Batch: 28/3rd Year 2nd Semester 2024

1 Objective(s)

- To understand the concept and need of interpolation for the estimation of unknown values.
- Implement Lagrange Interpolation to approximate values of a function using tabulated data.
- Implement Newton's Divided Difference Interpolation and construct the forward interpolation polynomial.
- Compare efficiency, computational complexity, and suitability of both interpolation techniques.

2 Background Theory

Interpolation is a numerical method used to estimate the value of a function between two known values. It constructs a polynomial that passes through a given set of data points. "The following two interpolation approaches are commonly used in scientific computing and data fitting. Each constructs a polynomial passing through all given data points, but differs in formulation and computational cost.

2.1 The Lagrange Polynomial

Lagrange interpolation constructs an interpolating polynomial of degree n for given data points $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$.

$$P_n(x) = \sum_{i=0}^{n} y_i \, L_i(x)$$

where

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

Characteristics

- Simple to understand.
- No need to construct a difference table.
- Requires recomputation if a new data point is added.

2.2 Newton's Divided Difference Interpolation

Newton's interpolation uses a divided difference table to form an interpolation polynomial.

Divided Difference Table Formula

$$f[x_i, x_{i+1}] = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$$

General Divided Difference

$$f[x_i, x_{i+1}, \ldots, x_{i+k}] = \frac{f[x_{i+1}, \ldots, x_{i+k}] - f[x_i, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}$$

Newton Polynomial

$$P_n(x) = f[x_0] + (x - x_0) f[x_0, x_1] + (x - x_0)(x - x_1) f[x_0, x_1, x_2] + \cdots$$

Characteristics

- Efficient for incremental addition of data points.
- Uses a difference table — easier for computation.
- Suitable for numerical algorithms.

2.3 Difference Between Lagrange and Newton Interpolation

Feature	**Lagrange Interpolation**	**Newton's Divided Difference**
Complexity	High for new data addition	Efficient for adding new points
Approach	Direct formula	Table-based recursive computation
Programming Ease	Simple	Slightly complex but scalable
Best Use	Small data sets	Large data sets & incremental updates

3 Sample Data for Demonstration

This section demonstrates manual calculation steps for both methods using a small cubic dataset to ensure conceptual understanding before coding

x	$f(x)$
1	1
2	8
3	27

Estimate $f(2.5)$.

3.1 Lagrange Interpolation: Step-by-Step Calculation

Given points: $(1,1)$, $(2,8)$, $(3,27)$

$$L_0(x) = \frac{(x-2)(x-3)}{(1-2)(1-3)} = \frac{(x-2)(x-3)}{2}$$

$$L_1(x) = \frac{(x-1)(x-3)}{(2-1)(2-3)} = -(x-1)(x-3)$$

$$L_2(x) = \frac{(x-1)(x-2)}{(3-1)(3-2)} = \frac{(x-1)(x-2)}{2}$$

Now form:

$$P(x) = 1 \cdot L_0(x) + 8 \cdot L_1(x) + 27 \cdot L_2(x)$$

Substitute $x = 2.5$ to compute the value.

3.2 Newton's Divided Difference: Step-by-Step Calculation

Divided Difference Table

x	$f[x]$	$f[x_0, x_1]$	$f[x_0, x_1, x_2]$
1	1	$\frac{8-1}{2-1} = 7$	$\frac{27-1}{3-1} = 13$
2	8	$\frac{27-8}{3-2} = 19$	
3	27		

Newton Polynomial

$$P(x) = 1 + (x-1)(7) + (x-1)(x-2)(13)$$

Evaluate at $x = 2.5$.

4 Algorithms

4.1 Algorithm: Lagrange Interpolation

1. Read n and data points $(x[i], y[i])$.
2. Read the value of x for which $f(x)$ is required.
3. Initialize `result = 0`.

4. For $i = 0$ to $n - 1$:
   (a) Compute $L_i = 1$.
   (b) For $j = 0$ to $n - 1$, $j \neq i$:
   $$L_i = L_i \times \frac{x - x[j]}{x[i] - x[j]}$$
   (c) `result = result + L_i * y[i]`.
5. Display result.

4.2 Algorithm: Newton's Divided Difference

1. Read n and input data points $(x[i], y[i])$.
2. Construct divided difference table.
3. Initialize `result = f[x0]`.
4. For $k = 1$ to $n - 1$:
   (a) `term = f[x0, x1, ..., xk]`.
   (b) For $j = 0$ to $k - 1$:
   `term = term * (x - x[j])`
   (c) `result = result + term`.
5. Display result.

5 Lab Tasks (Please implement yourself and show the output to the instructor)

You are given the following measured data as follows,

i	x_i	y_i
0	0	2.00000
1	1	5.43750
2	2.5	7.35160
3	3	7.56250
4	4.5	8.44530
5	5	9.18750
6	6	12.00000

Task 1 — Lagrange interpolating polynomial

1. **Build and evaluate low-order interpolants.**
   (a) Using the four points nearest to $x = 3.5$ construct a *second-order* (quadratic) Lagrange interpolant $P_2(x)$. (Choose three nodes that are closest to 3.5; document your choice.)
   (b) Evaluate $P_2(3.5)$.
2. **Increase polynomial degree and monitor convergence.**
   (a) Construct a cubic Lagrange interpolant $P_3(x)$ using four nodes (again choose nodes to give best local accuracy; document the node order).
   (b) Optionally construct higher degree interpolants $P_4(x), P_5(x), \ldots$ using more points (up to using all seven points).
   (c) For each interpolant P_k report $P_k(3.5)$ and the change compared with the previous degree, i.e.
   $$\Delta_k = \left|P_k(3.5) - P_{k-1}(3.5)\right|.$$
   This helps judge convergence as the degree increases.

3. **Deliverables for Task 1:**
   - The Lagrange basis functions used (symbolic or numeric).
   - The expanded polynomial(s) (coefficients to at least 6 significant digits).
   - Values $P_k(3.5)$ for each degree used and the Δ_k sequence.
   - Provide tables of $P_k(3.5)$, convergence differences Δ_k, and a plot of $P_k(x)$ for all degrees used

Task 2 — Newton's divided-difference polynomial

1. **Node selection and ordering.** For best local accuracy at $x = 3.5$, choose and order the nodes so the Newton form is centered near $x = 3.5$. Explain your ordering (e.g. choose nodes by increasing distance from 3.5).
2. **Divided difference table.** Compute the full divided-difference table for the chosen nodes (show all columns), i.e. compute
$$f[x_i], \quad f[x_i, x_{i+1}], \quad f[x_i, x_{i+1}, x_{i+2}], \ \ldots$$
   to sufficient precision (at least 6–8 significant digits).
3. **Construct Newton polynomial and evaluate.**
   - Form the Newton polynomial $N_k(x)$ for several degrees k (start with $k = 2$ and increase to $k = 3, 4$ etc).
   - Evaluate $N_k(3.5)$ and report results.
4. **Deliverables for Task 2:**
   - Full divided-difference table (formatted).
   - Newton polynomial(s) in Newton form and, optionally, expanded form.
   - Values $N_k(3.5)$ for each k and the convergence differences $\Delta_k = |N_k(3.5) - N_{k-1}(3.5)|$.

6 Lab Report (Please implement yourself and submit as a lab report)

In this task, you will simulate temperature variation along a 100 km highway using sensor data. The goal is to estimate unknown temperatures between sensor locations and analyze interpolation performance. The following table contains the temperature readings (in °C) collected by an array of ground sensors placed along a 100 km highway stretch at various distances (km) from the starting point. Measurements were taken at a specific time (e.g., 12:00 PM).

i	Distance (km)	Temperature (°C)
0	0	25.0
1	10	26.7
2	20	29.4
3	35	33.2
4	50	35.5
5	65	36.1
6	80	37.8
7	90	38.9
8	100	40.0

Tasks for Students

You are to use polynomial interpolation methods to:

1. Predict the temperature at the midpoint, $x = 45\,\text{km}$, using
   - Lagrange interpolation (2nd, 3rd, 4th, and full-degree polynomials).
   - Newton's divided difference method (2nd, 3rd, 4th, and full-degree polynomials).
2. Compare the interpolated values and discuss which method yields more stable results.

3. Generate an interpolation curve $T(x)$ for $x \in [0, 100]$ using each method (with at least 200 points) and plot them on the same graph.

4. Generate a convergence curve with an increasing degree of polynomials.

7 Policy

Copying from the Internet, classmates, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.