Numerical Methods

DS288 and UMC201

Ratikanta Behera

Department of Computational and Data Sciences,
Indian Institute of Science Bangalore

August-December 2025

Divided Difference

Newton's divided difference interpolating

Remark

- In Lagrangian polynomial formulation, if a tabular point is added to the data then all Lagrangian polynomials are to be constructed fresh.
- Therefore, another form of interpolating polynomial is needed to meet this requirement.
- Since, already proved that interpolating polynomial is unique, so only form will be different.

Newton's divided difference interpolating formula

- Nodes: $\longrightarrow x_0, x_1, \cdots, x_n$
- Functional value: $\longrightarrow f(x_0), f(x_1), \cdots, f(x_n)$
- The divided difference of zeroth order for argument x_0 is denoted by $f[x_0]$, defined by

$$f[x_0] = f_0$$

- The divided difference of first order for arguments $x_0,\ x_1$ is denoted by $f[x_0,\ x_1]$, defined by

$$f[x_0,\ x_1] = \frac{f_1 - f_0}{x_1 - x_0} = \frac{f[x_1] - f[x_0]}{x_1 - x_0}$$

- The divided difference of second order for arguments $x_0,\ x_1,\ x_2$ is denoted by $f[x_0,\ x_1, x_2]$, defined by

$$f[x_0,\ x_1,\ x_2] = \frac{f[x_1,\ x_2] - f[x_0,\ x_1]}{x_2 - x_0}$$

Newton's divided difference interpolating formula

- In general the *nth* order divided difference of the function f $n+1$ nodes $x_0, \ x_1, \cdots, \ x_n$

$$f[x_0, x_1, \cdots, x_n] = \frac{f[x_1, x_2, \cdots, x_n] - f[x_0, x_1, \cdots x_{n-1}]}{x_n - x_0}$$

- Interpolating polynomial in the Newton form

$$\begin{aligned} p_n(x) =& f(x_0) + (x - x_0) f[x_0, x_1] \\ & + (x - x_0)(x - x_1) f[x_0, x_1, x_2] \\ & + (x - x_0)(x - x_1)(x - x_2) f[x_0, x_1, x_2, x_3] \\ & + (x - x_0)(x - x_1) \cdots (x - x_{n-1}) f[x_0, x_1, \cdots, x_n] \end{aligned}$$

This is called Newton's divided difference formula for interpolating polynomial.

Newton's divided difference interpolating formula

x	f	1st order	2nd order	3rd order	4th order
x_0	$f[x_0]$				
x_1	$f[x_1]$	$f[x_0, x_1]$			
x_2	$f[x_2]$	$f[x_1, x_2]$	$f[x_0, x_1, x_2]$		
x_3	$f[x_3]$	$f[x_2, x_3]$	$f[x_1, x_2, x_3]$	$f[x_0, x_1, x_2, x_3]$	
x_4	$f[x_4]$	$f[x_3, x_4]$	$f[x_2, x_3, x_4]$	$f[x_1, x_2, x_3, x_4]$	$f[x_0, x_1, x_2, x_3, x_4]$

Example

Construct the Newton's divided difference interpolation polynomial for the following data

$f(10) = 335, \quad f(0) = -5, \quad f(8) = -21, \quad f(1) = -14, \quad f(4) = -125$

Newton's Forward Form for Interpolation

- Newton divided difference formula when $x_0, x_1, x_2, \cdots, x_n$ are equispaced, i.e., $h = x_{i+1} - x_i$ for $i = 0, 1, 2, \cdots n$
- The forward difference operator denoted as Δ
- The forward difference operator defined as $\Delta f(x) = f(x+h) - f(x)$. Now
- Zero order $\longrightarrow$ $\Delta^0 f_i = f_i$
- First order $\longrightarrow$ $\Delta^1 f_i = f_{i+1} - f_i$
- Second order $\longrightarrow$ $\Delta^2 f_i = \Delta f_{i+1} - \Delta f_i = f_{i+2} - 2f_{i+1} + f_i$
- nth order $\longrightarrow$ $\Delta^n f_i = \Delta^{n-1} f_{i+1} - \Delta^{n-1} f_i$

$$\Delta^n f_i = \Delta^n f_{n+i} - \binom{n}{1} f_{n+i-1} + \binom{n}{2} f_{n+i-2} + \cdots + (-1)^n f_i$$

Newton's Forward Form for Interpolation

x	f	Δf	$\Delta^2 f$	$\Delta^3 f$	$\Delta^4 f$
x_0	$f[x_0]$				
x_1	$f[x_1]$	Δf_0			
x_2	$f[x_2]$	Δf_1	$\Delta^2 f_0$		
x_3	$f[x_3]$	Δf_2	$\Delta^2 f_1$	$\Delta^3 f_0$	
x_4	$f[x_4]$	Δf_3	$\Delta^2 f_2$	$\Delta^3 f_1$	$\Delta^4 f_0$

Example

- Prepare the forward difference table for the following data

x	0.2	0.4	0.6	0.8
$f(x)$	3.2	3.6	2.8	3.0

- interpolate $f(0.3)$.

Newton's Forward Divided Difference

When the points $\{x_0, \ldots, x_n\}$ are equally spaced, i.e.

$$h = x_{i+1} - x_i, \quad i = 0, \ldots, n-1,$$

we can write $x = x_0 + sh$, $x - x_k = (s-k)h$ so that

$$P_n(x) = P_n(x_0 + sh) = \sum_{k=0}^{n} s(s-1)\cdots(s-k+1)h^k f[x_0, \ldots, x_k].$$

Using the binomial coefficients, $\binom{s}{k} = \frac{s(s-1)\cdots(s-k+1)}{k!}$

$$\mathbf{P_n(x_0 + sh) = f[x_0] + \sum_{k=1}^{n} \binom{s}{k} k! h^k f[x_0, \ldots, x_k].}$$

This is **Newton's Forward Divided Difference Formula**.

Newton's Forward Divided Difference

Another form, **Newton's Forward Difference Formula** is constructed by using the forward difference operator Δ:

$$\Delta f(x_n) = f(x_{n+1}) - f(x_n)$$

using this notation:

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{1}{h}\Delta f(x_0).$$

$$f[x_0, x_1, x_2] = \frac{1}{2h}\left[\frac{\Delta f(x_1) - \Delta f(x_0)}{h}\right] = \frac{1}{2h^2}\Delta^2 f(x_0).$$

$$f[x_0, \ldots, x_k] = \frac{1}{k!h^k}\Delta^k f(x_0).$$

Thus we can write **Newton's Forward Difference Formula**

$$\mathbf{P_n(x_0 + sh) = f[x_0] + \sum_{k=1}^{n} \binom{s}{k} \Delta^k f(x_0).}$$

Newton's Backward Difference

If we reorder $\{x_0, x_1, \ldots, x_n\} \to \{x_n, \ldots, x_1, x_0\}$, and define the backward difference operator ∇:

$$\nabla f(x_n) = f(x_n) - f(x_{n-1}).$$

we can define the backward divided differences:

$$f[x_n, \ldots, x_{n-k}] = \frac{1}{k!h^k} \nabla^k f(x_n).$$

We write down **Newton's Backward Difference Formula**

$$\mathbf{P_n(x) = f[x_n] + \sum_{k=1}^{n} (-1)^k \binom{-s}{k} \nabla^k f(x_n)}.$$

where

$$\binom{-s}{k} = (-1)^k \frac{s(s+1)\cdots(s+k-1)}{k!}$$

Newton's backward Form for Interpolation

- Newton divided difference formula when $x_0, x_1, x_2, \cdots, x_n$ are equispaced, i.e., $h = x_{i+1} - x_i$ for $i = 0, 1, 2, \cdots n$
- The backward difference operator denoted as ∇
- The forward difference operator defined as $\nabla f(x) = f(x) - f(x-h)$. Now
- Zero order $\longrightarrow \nabla^0 f_i = f_i$
- First order $\longrightarrow \nabla^1 f_i = f_i - f_{i-1}$
- Second order $\longrightarrow \nabla^2 f_i = \nabla f_i - \nabla f_{i-1} = f_i - 2f_{i-1} + f_{i-2}$
- nth order $\longrightarrow \nabla^n f_i = \nabla^{n-1} f_i - \nabla^{n-1} f_{i-1}$

$$\nabla^n f_i = \sum_{i=0}^{n} (-1)^i \binom{n}{i} f_{n-1}$$

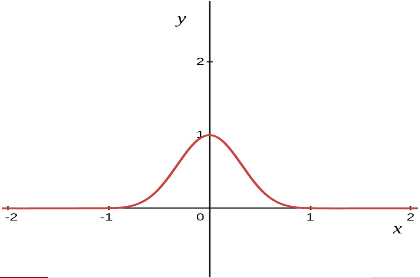
Piecewise Polynomial Interpolation

Piecewise Polynomial Interpolation

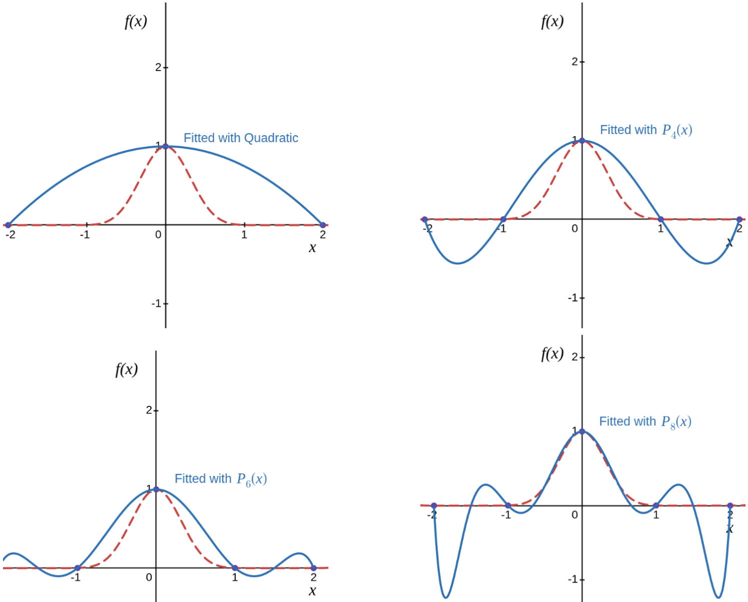
- Large number of data points results the higher degree interpolating polynomial.
- Oscillating nature of high degree polynomials induce more error in interpolation.

Example

let $f(x) = \cos^{10} x$ over the interval $[-2, 2]$. Fit the polynomials of degrees $2, 4, 6,$ and 8.



Piecewise Polynomial Interpolation



Piecewise Polynomial Interpolation

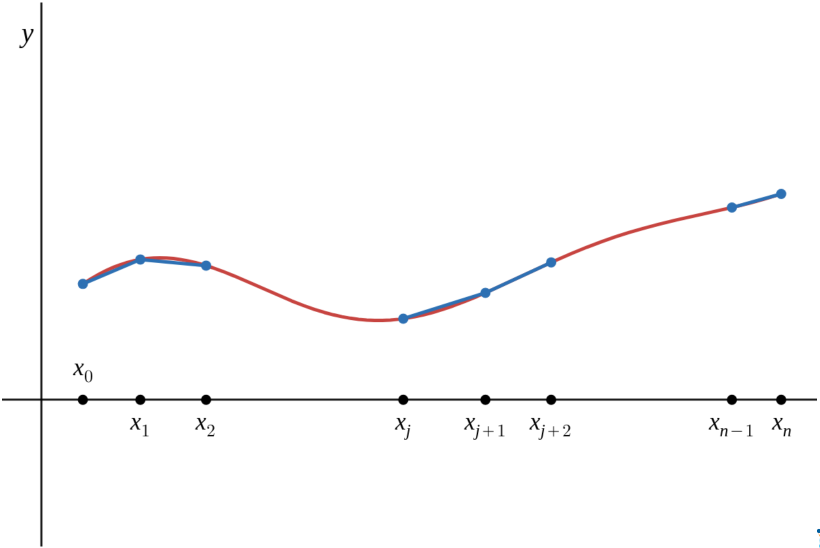
To overcome this inaccuracy:

- Total interval is divided into small sub intervals and on each interval a lower degree interpolating polynomial is constructed.
- Approximation of functions by this type is called piecewise polynomial interpolation.
- So, this is another way to fit a polynomial to a set of data.

➢ Simplest piecewise interpolation is piecewise linear interpolation, which consists of joining a set of data points $(x_0, f_0), (x_1, f_1), (x_2, f_2), \cdots, (x_n, f_n)$ by a series of straight lines.

➢ Tabular points $x_0, x_1, x_2, \cdots x_n$ may not be at necessarily equispaced.

Piecewise Polynomial Interpolation



Piecewise Polynomial Interpolation

- On interval $[x_{i-1}, x_i]$, let us define the linear polynomial

$$p_{1i} = f_{i-1} + (x - x_i) f[x_{i-1}, x_i], \quad i = 1, 2, 3, \cdots, n$$

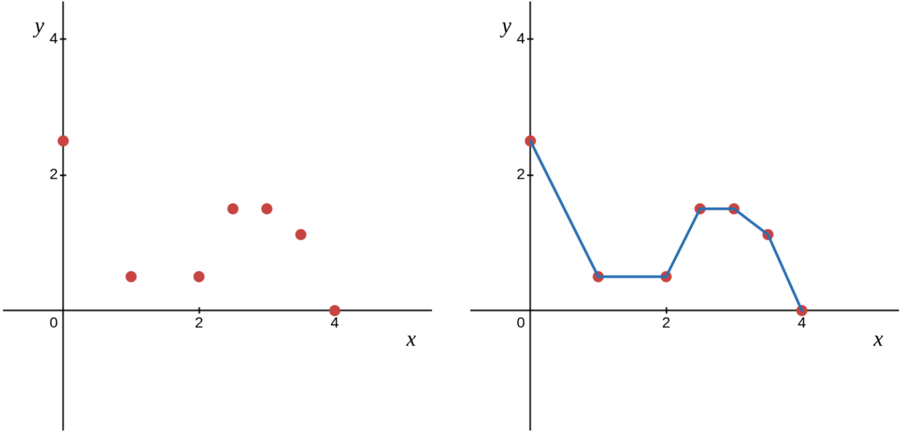
- There are n-such polynomials of degree one for a pair of tabular points. This is known as piecewise linear interpolation.

Example

Consider the data

n	0	1	2	2.5	3	3.5	4
x_n	2.5	0.5	0.5	1.5	1.5	1.12	0

Piecewise Polynomial Interpolation



Upper Bound on Piecewise Linear interpolation

- Let us assume that the tabular points $x_0, x_1, x_2, \cdots, x_n$ are equispaced. Then

  $$x_{i+1} - x_i = h i = 1, 2, \cdots, n-1, \text{ and } x_i - x_0 = ih, i = 1, 2, \cdots, n$$

- Then the upper bound on piecewise linear interpolation is

  $$E(I) = \frac{Mh^2}{8}$$

  where h is width of each sub interval, $I = [x_0, x_n]$ and $M = \max_{x \in I} |f^{''}(x)|$

Disadvantage of linear function approximation

- No differentiability at the end points of the subintervals, i.e. the interpolating function is not smooth.
- From the physical conditions it is required that the approximation function must be continuously differentiable.

Piecewise Quadratic interpolation

- To obtain a somewhat smoother graph, consider using piecewise quadratic interpolation.
- Divided the interval [a, b] into 2n (even number) equal (not necessary) intervals, i.e.

$$\frac{b-a}{2n} = h, \quad \text{such that} \quad a = x_0, x_1, \cdots x_{2n-2}, x_{2n-1}, x_{2n} = b$$

- Begin by constructing the quadratic polynomial that interpolates $\{(x_{2i-1}, f_{2i-2}), (x_{2i-1}, f_{2i-1}), (x_{2i}, f_{2i})\}$
- Then, the quadratic interpolating polynomial is

$$\begin{aligned} P_{2i} =& f_{2i-2} + (x - x_{2i-2)} f[x_{2i-2}, x_{2i-1}] \\ & \quad + (x - x_{2i-2)}(x - x_{2i-1)} f[x_{2i-2}, x_{2i-1}, x_{2i}], \\ & \quad \text{for the interval } [x_{2i-2}, x_{2i}] \quad i = 1, 2, 3, \cdots, n \end{aligned}$$

- These polynomials are n in numbers.

Upper Bound on Piecewise Quadratic interpolation

- with error term

$$E_i = (x - x_{2i-2})(x - x_{2i-1})(x - x_{2i})\frac{f'''(\zeta_i)}{3!}, \quad i = 1, 2, \cdots n$$

for some $\zeta_i \in (x_{2i-2}, x_{2i})$

- Then, the upper bound on the error is

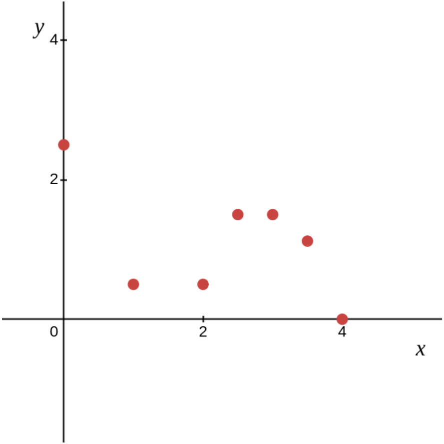
$$|E| = \frac{Mh^3}{9\sqrt{3}}$$

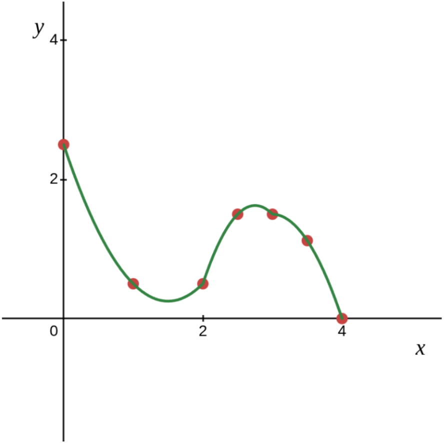
Example

Consider the data

n	0	1	2	2.5	3	3.5	4
x_n	2.5	0.5	0.5	1.5	1.5	1.125	0

Piecewise Quadratic Interpolation






ANY
QUESTIONS?