

The Newton interpolation method for the data points

$$(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$$

describes the n -degree polynomial interpolant as

$$\begin{aligned}
 p(x) = & c_0 \cdot \underbrace{1}_{n_0(x)} \\
 & + c_1 \cdot \underbrace{(x-x_0)}_{n_1(x)} \\
 & + c_2 \cdot \underbrace{(x-x_0)(x-x_1)}_{n_2(x)} \\
 & \vdots \\
 & + c_n \cdot \underbrace{(x-x_0)(x-x_1)\dots(x-x_{n-1})}_{n_n(x)}
 \end{aligned}$$

The coefficients $\{c_i\}$ are computed using the method of divided differences: For a set of x -values x_i, x_{i+1}, \dots, x_j we define the divided difference as $f[x_i, \dots, x_j]$.

The value of this symbol is recursively defined as:

$$\begin{aligned}
 \rightarrow f[x_i] &= y_i \quad \text{for } i=0, \dots, n \\
 \rightarrow f[x_i, x_{i+1}, \dots, x_j] &= \frac{f[x_{i+1}, \dots, x_j] - f[x_i, x_{i+1}, \dots, x_{j-1}]}{x_j - x_i}
 \end{aligned}$$

Once all relevant divided differences have been computed, the coefficients c_i are given by:

$$c_0 = f[x_0]$$

$$c_1 = f[x_0, x_1]$$

$$c_2 = f[x_0, x_1, x_2]$$

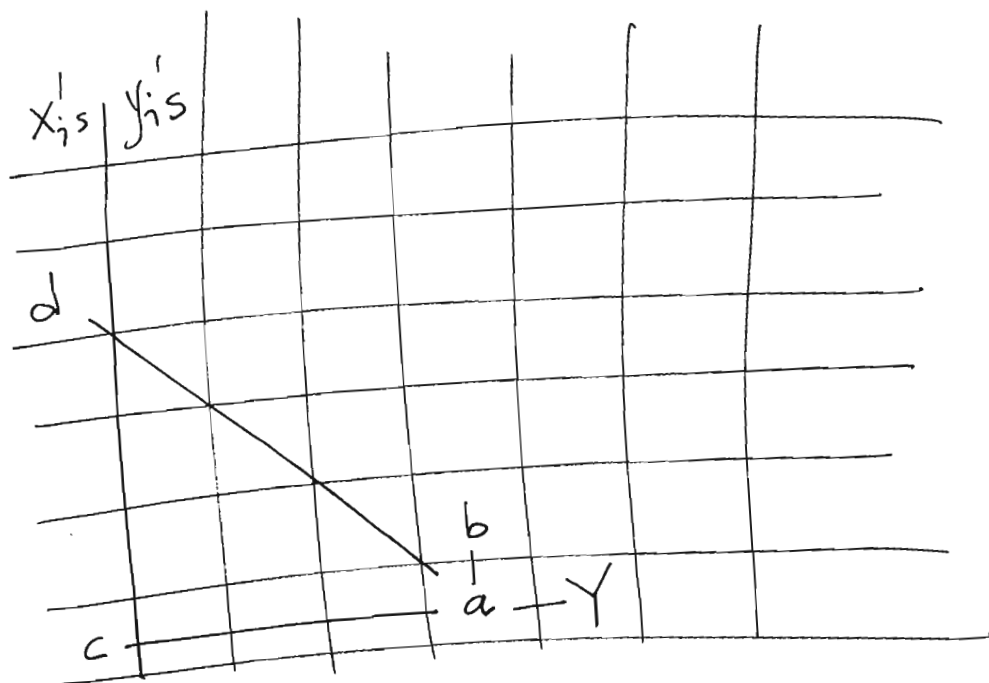
$$\vdots$$

$$c_n = f[x_0, x_1, \dots, x_n]$$

Divided differences are usually tabulated as follows

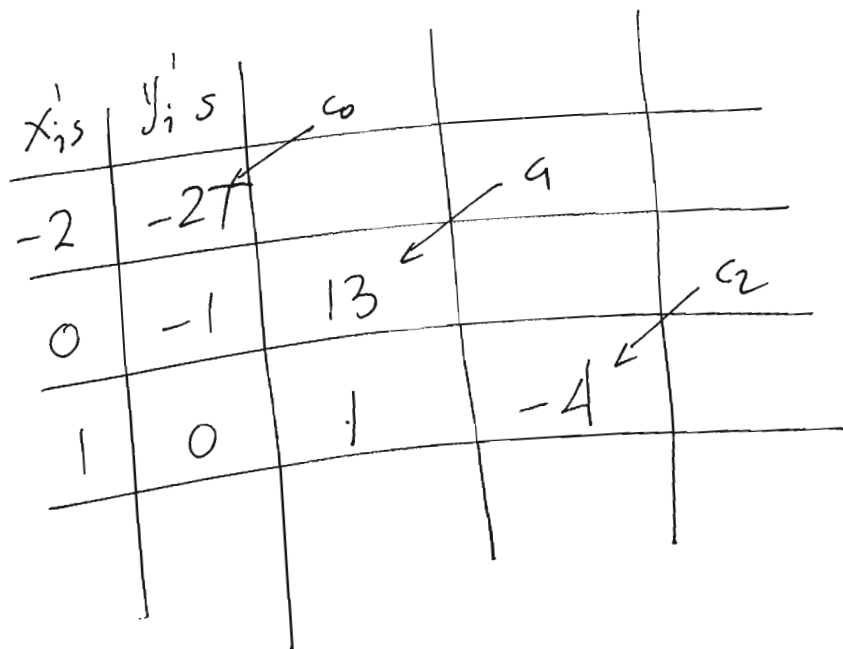
	$f[\cdot]$	$f[\cdot, \cdot]$	$f[\cdot, \cdot, \cdot]$
x_0	$f[x_0]$ $\rightarrow c_0$		
x_1	$f[x_1]$	$f[x_0, x_1]$ $\rightarrow c_1$	
x_2	$f[x_2]$	$f[x_1, x_2]$	$f[x_0, x_1, x_2]$ $\rightarrow c_2$
x_3	$f[x_3]$	$f[x_2, x_3]$	$f[x_1, x_2, x_3]$...
x_4	$f[x_4]$	$f[x_3, x_4]$	$f[x_2, x_3, x_4]$

The recursive definition can be implemented directly on the table as follows 2/15/2011 \triangleleft



$$Y = \frac{a-b}{c-d}$$

e.g. $(x_0, y_0) = (-2, -27)$ $(x_1, y_1) = (0, -1)$ $(x_2, y_2) = (1, 0)$



Easy evaluation

2/15/2011 [3]

e.g.
$$\begin{aligned} p(x) &= c_0 \\ &+ c_1 (x-x_0) \\ &+ c_2 (x-x_0)(x-x_1) \\ &+ c_3 (x-x_0)(x-x_1)(x-x_2) \\ &+ c_4 (x-x_0)(x-x_1)(x-x_2)(x-x_3) = \end{aligned}$$

$$= c_0 + (x-x_0) \left[c_1 + (x-x_1) \left[c_2 + (x-x_2) \left[c_3 + (x-x_3) c_4 \right] \right] \right]$$

The diagram illustrates the nested structure of the polynomial evaluation. It shows four nested brackets, each representing a partial product $Q_k(x)$:

- $Q_4(x)$ is the innermost bracket, enclosing $c_3 + (x-x_3)c_4$.
- $Q_3(x)$ is the next bracket, enclosing $c_2 + (x-x_2)Q_4(x)$.
- $Q_2(x)$ is the next bracket, enclosing $c_1 + (x-x_1)Q_3(x)$.
- $Q_1(x)$ is the next bracket, enclosing $c_0 + (x-x_0)Q_2(x)$.
- $Q_0(x)$ is the outermost and longest bracket, enclosing the entire expression.

$$P(x) = Q_0(x)$$

recursively: $Q_m(x) = c_m$

$$Q_{m-1}(x) = c_{m-1} + (x-x_{m-1})Q_m(x)$$

The value of $P(x) = Q_0(x)$ can be evaluated
(in linear time) by iterating this recurrence
 n times

We also have:

$$Q_{n-1}(x) = c_{n-1} + (x - x_{n-1}) Q_n(x)$$

$$\Rightarrow Q_{n-1}'(x) = Q_n(x) + (x - x_{n-1}) Q_n'(x)$$

Thus, once we have computed all the Q_k 's we can also compute all the derivatives too. Ultimately, $P'(x) = Q_0'(x)$.