Exercise

① Write a python program to implement nth order Newton's interpolation formula.

Ans

Algorithm :

(i) Create numpy arrays to enter the given data points

(ii) Then compute the divided difference coefficients. $\boxed{\text{How??}}$

Suppose we have 5 data points. Then we can do

	0	1	2	3	4
0	$f(x_0)$	$[x_1, x_0]$	$[x_2\ x_1\ x_0]$	$[x_3\ x_2\ x_1\ x_0]$	$[x_4\ x_3\ x_2\ x_1\ x_0]$
1	$f(x_1)$	$[x_2, x_1]$	$[x_3\ x_2\ x_1]$	$[x_4\ x_3\ x_2\ x_1]$	
2	$f(x_2)$	$[x_3, x_2]$	$[x_4\ x_3\ x_2]$		
3	$f(x_3)$	$[x_4, x_3]$			
4	$f(x_4)$				

4 th order polynomial interpolation

(iii) Insert the coefficients to Newton's formula & get the desired value of $f(x)$.

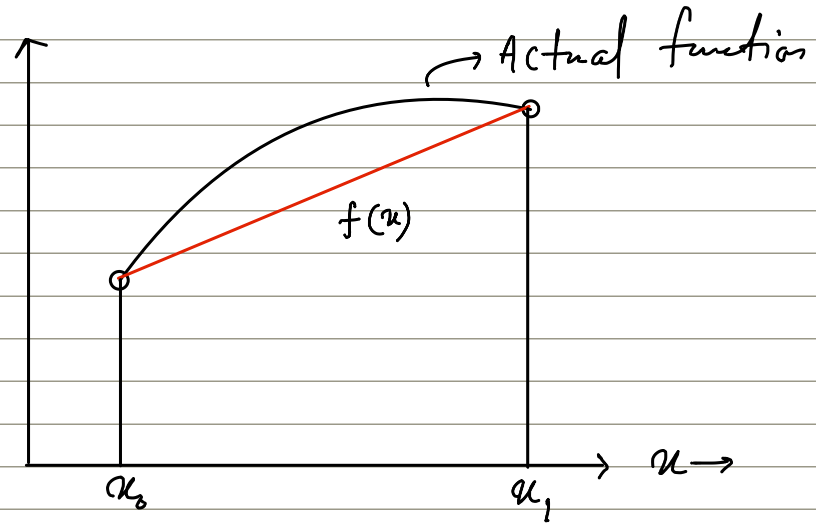
Let's implement this algorithm in python.

Lagrange Interpolating Polynomials

1. Linear Interpolation :-

Similar approach as Newton's divided difference method.

Assume a straight line is passing through the two data points, as shown in fig. below :



The straight line is represented by a linear function in the form,

$$f(x) = ax + b \qquad -(1)$$

where a & b are two constants. They can be determined as :

$$\text{at } x = x_0, \quad f(x_0) = ax_0 + b \qquad -(2)$$

$$\text{at } x = x_1, \quad f(x_1) = ax_1 + b \qquad -(3)$$

Substracting (2) from (3)

$$f(x_1) - f(x_0) = a(x_1 - x_0) \qquad -(4)$$

$$\Rightarrow \boxed{a = \frac{f(x_1) - f(x_0)}{x_1 - x_0}} \qquad -(5)$$

Now, substitute the value of 'a' into eqn (2)

$$\left[\frac{f(x_1) - f(x_0)}{x_1 - x_0}\right] x_0 + b = f(x_0)$$

$$\Rightarrow \boxed{b = f(x_0) - \left[\frac{f(x_1) - f(x_0)}{x_1 - x_0}\right] x_0} \quad \text{—(6)}$$

Thus, eqn (1) becomes,

$$f(x) = \left[\frac{f(x_1) - f(x_0)}{x_1 - x_0}\right] x + f(x_0) - \left[\frac{f(x_1) - f(x_0)}{x_1 - x_0}\right] x_0$$

$$= \frac{[(x_1 - x_0) - x + x_0]}{x_1 - x_0} + \left[\frac{x - x_0}{x_1 - x_0}\right] f(x_1)$$

$$= \left(\frac{x_1 - x}{x_1 - x_0}\right) f(x_0) + \left(\frac{x_0 - x}{x_0 - x_1}\right) f(x_1)$$

$$\Rightarrow \boxed{f(x) = L_0(x) f(x_0) + L_1(x) f(x_1)} \quad \text{—(7)}$$

where, $L_0(x) = \dfrac{x_1 - x}{x_1 - x_0}$

& $L_1(x) = \dfrac{x_0 - x}{x_0 - x_1}$

The functions $L_0(x)$ & $L_1(x)$ are called the Lagrange interpolation function.

They satisfy the following condition,

$$L_i(x) = \begin{cases} 1 & \text{for } x = x_i \\ 0 & \text{for } x \neq x_i \end{cases} \quad \text{—(8)}$$

Check

2. Quadratic Interpolation :-

We have 3 data points, (x_0, y_0), (x_1, y_1) & (x_2, y_2).

Choose, $f(x) = ax^2 + bx + c$ —(9)

a, b & c are 3 constants. Can be determined from,

at $x = x_0$, $f(x_0) = ax_0^2 + bx_0 + c$

at $x = x_1$, $f(x_1) = ax_1^2 + bx_1 + c$

at $x = x_2$, $f(x_2) = ax_2^2 + bx_2 + c$

Try to solve these 3 equations for a, b & c.

Then $f(x) = L_0(x) f(x_0) + L_1(x) f(x_1) + L_2(x) f(x_2)$ —(10)

where, $$L_0(x) = \frac{(x_2 - x)(x_1 - x)}{(x_2 - x_0)(x_1 - x_0)}$$

$$L_1(x) = \frac{(x_2 - x)(x_0 - x)}{(x_2 - x_1)(x_0 - x_1)}$$

$$L_2(x) = \frac{(x_1 - x)(x_0 - x)}{(x_1 - x_2)(x_0 - x_2)}$$

Choose arbitrary values of x_0, x_1 & x_2 and check the validity of condition given in eqⁿ (8). (By plotting L_0, L_1 & L_2)

3. Polynomial Interpolation :-

Generalize the concept.

Choose nth order polynomial passing through $n+1$ points.

$$f(x) = \sum_{i=0}^{n} a_i x^i \quad — (11)$$

Then we can get (by using $f(x_i)$ at different x_i's)

$$\boxed{f(x) = \sum_{i=0}^{n} L_i(x) f(x_i) \quad — (12) \qquad \text{where, } L_i(x) = \prod_{\substack{j=0 \\ (j \neq i)}}^{n} \frac{x - x_j}{x_i - x_j} \quad — (13)}$$

$\Pi \rightarrow$ product

$\Sigma \rightarrow$ sum

example

$$L_2(x) = \frac{x - x_0}{x_2 - x_0} \quad \frac{x - x_1}{x_2 - x_1}$$

(for $n = 2$)

Let's implement this formula in Python.

~~Imp~~

Please read what is extrapolation from P. Dechaumphai book (shared in whp)