Theory

27 January 2022 19:47

Newton's forward Interpolation formula

x	x_0	x_1	x_2	x_3	. . .	x_n
y	y_0	y_1	y_2	y_3	-	y_n

$$f(x) = y_0 + u\Delta y_0 + \frac{u(u-1)}{2!}\Delta^2 y_0 + \frac{u(u-1)(u-2)}{3!}\Delta^3 y_0 + \cdots + \frac{u(u-1)(u-2)\ldots(u-n+1)}{n!}\Delta^n y_0$$

$x = x_0 + uh$

$h = x_1 - x_0$
$= x_3 - x_2$
$= x_2 - x_1$
$\vdots$

Requirement
- x values for u
- h
- x_0
- x & corresponding y

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$
x_0	y_0	Δy_0	$\Delta^2 y_0$	$\Delta^3 y_0$	$\Delta^4 y_0$
x_1	y_1	Δy_1	$\Delta^2 y_1$	$\Delta^3 y_1$	
x_2	y_2	Δy_2	$\Delta^2 y_2$		
x_3	y_3	Δy_3			
x_4	y_4				

Here we can see for 5 values of x there are 4 different order forward difference

x_0	x_1	x_2	x_3	x_4
y_0	y_1	y_2	y_3	y_4
	Δy_0	$\Delta^2 y_0$	$\Delta^3 y_0$	$\Delta^4 y_0$

table

Step 1 - create structure table.

```
typedef struct{
    float x,
    float y;
    float forward_diff,
} table,
```

Step 2 :- Array of table of 'n' values of x

```
table data [5],
```

Step 3 - Take input for x & corresponding y

```
for(i=0, i<5, i++)
{
    cout<<" x[" <<i<<"]",
    cin >> x,
    cout<<"y[" <<i<<"]",
    cin >> y.
}
```

x_0	x_1	x_2	x_3	x_4
y_0	y_1	y_2	y_3	y_4
0	1	2	3	4

Step 4 - Find the u & h,

$\boxed{u = (x - x_0)/h}$ $\boxed{h = x_1 - x_0}$

$x_0 \approx$ table[0].x table[1].x

Step 5 - calculate forward diff.

```
table[0].forwarddiff=1,

for(i=0; i<(n-1) i++)
```

x_0	x_1	x_2	x_3	x_4
y_0	y_1	y_2	y_3	y_4
0	1	2	3	4
	Δy_0	Δy_1	Δy_2	Δy_3

forward

```
table[0].forwarddiff=1,

for(i=0; i<(n-1); i++)
{
    table[i+1].forward = table[i+1].y - table[i].y,
}

for(i=1, i<=n-2, i++)
{
    for(j=n-1, j>i ;j--)
    {
        table[j].forward = table[j].forward - table[j-1].forward
    }
}
```

forward: $\Delta y_0 \quad \Delta y_1 \quad \Delta y_2 \quad \Delta y_3$

$i=1 \rightsquigarrow j=4$

$f[4] = f[4] - f[3]$

0	1	2	3	4
1	Δy_0	Δy_1	Δy_2	$\cancel{\Delta y_3}$ $\Delta^2 y_2$

$j=3$

$f(3) = f(3) - f(2)$

0	1	2	3	4
f	Δy_0	Δy_1	$\cancel{\Delta y_2}$ $\Delta^2 y_1$	$\Delta^2 y_2$

$j=2$

$f(2) = f(2) - f(1)$

0	1	2	3	4
1	Δy_0	$\cancel{\Delta y_1}$ $\Delta^2 y_0$	$\Delta^2 y_1$	$\Delta^2 y_2$

$i=2 \quad j\ 4,3$

1	Δy_0	$\Delta^2 y_0$	$\Delta^3 y_0$	$\Delta^3 y_1$

$i=3 \quad j \to 4$

0	1	2	3	4
	Δy_0	$\Delta^2 y_0$	$\Delta^3 y_0$	$\Delta^4 y_0$

x_0	x_1	x_2	x_3	x_4
y_0	y_1	y_2	y_3	y_4
	Δy_0	$\Delta^2 y_0$	$\Delta^3 y_0$	$\Delta^4 y_0$
0	1	2	3	4

$$y_0 + \frac{u\,\Delta y_0}{1} + \frac{u(u-1)\Delta^2 y_0}{2!} + \frac{u(u-1)(u-2)\Delta^3 y_0}{3!} + \frac{u(u-1)(u-2)(u-3)\Delta^4 y_0}{4!}$$

$$\frac{u(u-1)\ \ (u-n+1)\Delta^n y_0}{n!}$$

$$u = \frac{x - x_0}{h}$$

$$u\ \frac{\text{table}[n].f}{f.}$$

$\frac{1}{f} \qquad \frac{u_1}{u}$

inter [n-1) $\quad n=5$

```
{
  if(n==0)
     return table[n].y;
  static  u = (x-x0)/h
  static  f=1

  val = inter(n-1),
  if( n > 1)
  {  u = u*(u-(n-1))
     f = f*(n)
  }
  return val + u/f * table[n].forward
}
```

$$\text{inter}(4)$$

$$\text{val} = \cancel{\text{inter}}(3) + \frac{u * \Delta^4 y_0}{4!}$$

$$\text{val} = \cancel{\text{inter}}(2) + \frac{u * \Delta^3 y_0}{6}$$

$$\text{val} \quad \cancel{\text{inter}}(1) + \frac{u * \Delta^2 y_0}{2}$$

$$\text{val} = \cancel{\text{inter}}(0)$$

$$y_0 + \frac{u * \Delta y_0}{1}$$

$$u \; \boxed{\cancel{\frac{x - x_0}{h}}} \; u(u-1)(u-2)(u-3)$$

$$f \; \boxed{\cancel{1}, \cancel{2}, \cancel{6}\, 2}\, 4$$