# Lab 8: Newton’s Divided & Forward Difference interpolation

**Introduction**

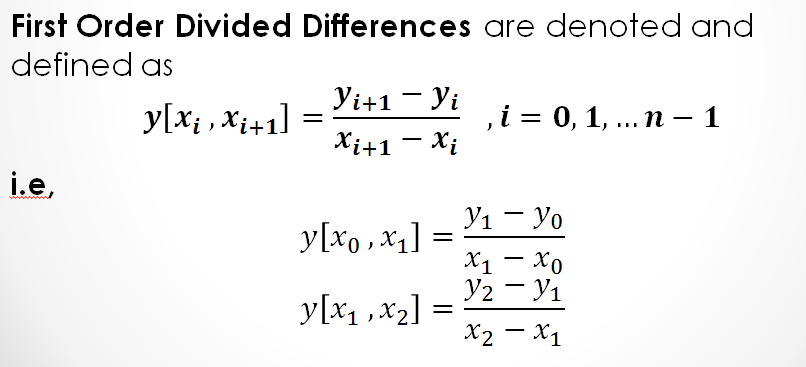
Newton's divided difference interpolation formula is a interpolation technique used when the interval difference is not same for all sequence of values. Divided differences are symmetric with respect to the arguments. i.e independent of the order of arguments.

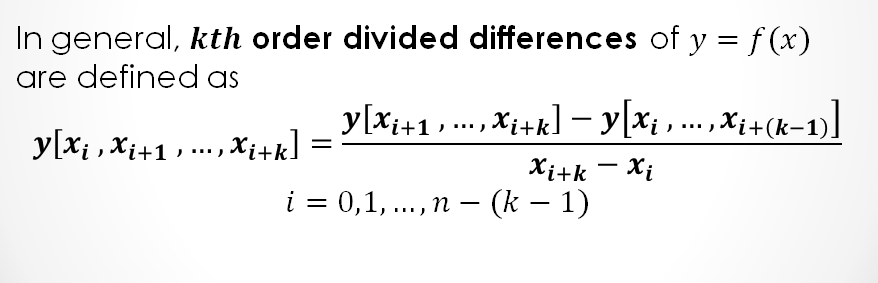
**Tools/Software Requirement**

Matlab R2016a

**Description**

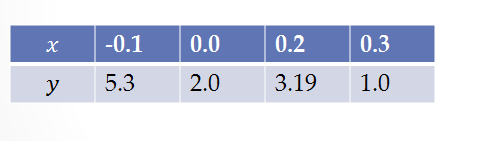
The application of the Lagrange formula becomes quite lengthy and difficult if number of values in the data is more than four. It can be seen that inclusion of a point in the data list leads to add one more term in the formula and the numerator and denominator of each term get changed. Further, during calculations there is always a chance of committing some error due to a number of positive and negative signs in the coefficients of each term. Except this, there is no way to guess the degree of the resulted polynomial before the final result is obtained after lengthy calculations. e.g. suppose data corresponds to a polynomial of degree two and data points are five. So we have to simplify five terms with each coefficients as a polynomial of degree four. But, only after simplifications of these five terms we can obtain a polynomial of degree two. Hence, it is not advisable to use Lagrange formula for more than four points in the data set. To overcome this problem, we prefer to use the Newton’s divided difference formula derived as below.

****

****

**Lab Task 1**

1. Construct a divided difference table for the following data using matlab.
2. Using Newton’s Divided Difference formula, construct polynomials of degree **(i)** one **(ii)** two and **(iii)** three for the following data. Also interpolate value of for in each case.



## Code:

X = [-0.1;0.0;0.2;0.3];

Y = [5.3;2.0;3.19;1.0];

n=4;

D = zeros(n);

for i = 1:n

D(i,1) = Y(i);

end

j = 1;

for m = 1:n-1

for i = 1:n-m

D(i,j+1) = (D(i+1,j) - D(i,j))/(X(i+j) - X(i));

end

j=j+1

end

D

syms x

P1 = D(1,1) + (x - X(1))\*D(1,2)

P2 = P1 + (x-X(1))\*(x-X(2))\*D(1,3)

P3 = P2 + (x-X(1))\*(x-X(2))\*(x-X(3))\*D(1,4)

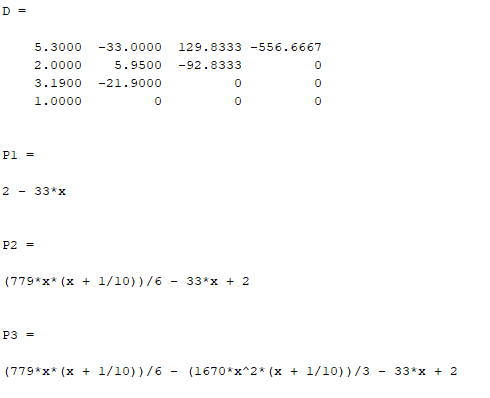
x = 0.1

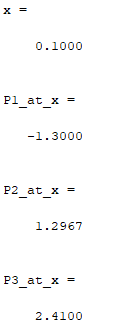
P1\_at\_x = eval(P1)

P2\_at\_x = eval(P2)

P3\_at\_x = eval(P3)

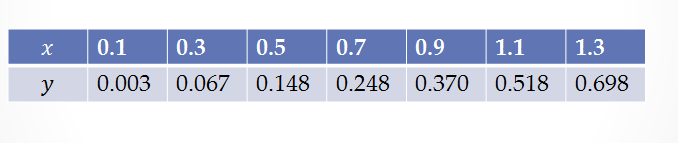
## Output:





**Lab Task 2**

1. Construct a forward difference table for the following data using matlab.



**b**. Interpolate value of for .

**c.** Construct Third degree interpolation polynomial using Newton’s Forward Difference Formula.

## Code:

X = [0.1;0.3;0.5;0.7;0.9;1.1;1.3];

Y = [0.003;0.067;0.148;0.248;0.370;0.518;0.698];

n=4;

D = zeros(4);

h = X(2)-X(1)

syms x

s = (x - X(1))/h

for i = 1:n

D(i,1) = Y(i);

end

j = 1;

for m = 1:n-1

for i = 1:n-m

D(i,j+1) = D(i+1,j) - D(i,j);

end

j=j+1;

end

D

P3 = D(1,1) +s\*D(1,2) +(s\*(s-1)/factorial(2))\*D(1,3)+(s\*(s-1)\*(s-2)/factorial(3))\*D(1,4)

x= 0.213

P3\_at\_x = eval(P3)

## Output:

