# Lab 9: Derivatives using Newton’s Forward, Backward, Sterling’s Central Difference Interpolation Formula

**Task 1:**

## 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=7;

D = zeros(4);

h = X(2)-X(1);

syms x

s = (x - X(4))/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(4,1) + (s\*(D(3,2)+D(4,2)))/2 + s^2\*D(3,3)/2 + s\*(s^2-1)\*((D(2,4)+D(3,4)))/(factorial(3)\*2) + (s^2\*(s^2 -1)\*D(2,5))/factorial(4) + (s\*(s^2 -1)\*(s^2-4)\*(D(1,6)+D(2,6)))/(factorial(5)\*2) + D(1,7)

P3 = simplify(P3)

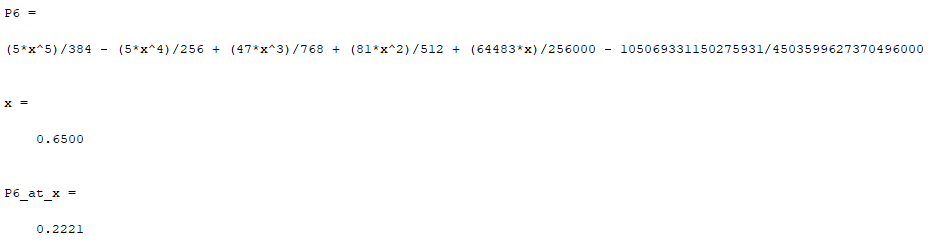
x= 0.65

P3\_at\_x = eval(P3)

## Output:

P6 =

(111\*x)/200 + (7\*(5\*x - 7/2)\*((5\*x - 7/2)^2 - 1))/12000 + (11\*(5\*x - 7/2)^2)/1000 + ((5\*x - 7/2)^2\*((5\*x - 7/2)^2 - 1))/24000 + ((5\*x - 7/2)\*((5\*x - 7/2)^2 - 1)\*((5\*x - 7/2)^2 - 4))/240000 - 628252148018185563/4503599627370496000



**Task 2:**

Considering a uniform beam of one meter long simply supported at both ends, the bending moment is given by the following relation:

Where is the deflection, is the bending moment and is the flexural rigidity.

Calculate the bending moment at

(i): using Newton’s Forward Difference Formula

(ii): using Newton’s Backward Difference Formula,

assuming that the deflection distribution is among the following:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
| , | 0.0000000 | 7.7800000 | 10.6800000 | 8.3700000 | 3.9700000 | 0.000000 |

**Part 1:**

## Code:

X = [0.0;0.2;0.4;0.8;1.0];

Y = [0.0;7.78;10.68;8.37;3.97;0.0];

n=6;

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

y1 = D(1,1)

y2 = s\*D(1,2)

y3 = s\*(s-1)\*D(1,3)/2

y4 = s\*(s-1)\*(s-2)\*D(1,4)/factorial(3)

y5 = s\*(s-1)\*(s-2)\*(s-3)\*D(1,5)/factorial(4)

y6 = s\*(s-1)\*(s-2)\*(s-3)\*(s-4)\*D(1,6)/factorial(5)

P5 = y1 + y2 + y3 + y4 + y5 + y6

P5 = simplify(P5)

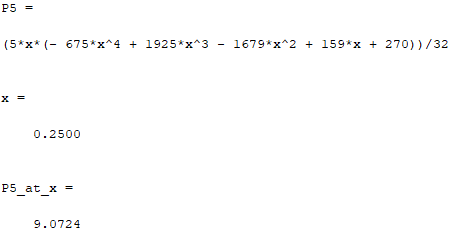
x= 0.25

P5\_at\_x = eval(P5)

## Output:

P5 =

(389\*x)/10 - (61\*x\*(5\*x - 1))/5 - (11\*x\*(5\*x - 1)\*(5\*x - 2))/40 + (23\*x\*(5\*x - 1)\*(5\*x - 2)\*(5\*x - 3))/32 - (27\*x\*(5\*x - 1)\*(5\*x - 2)\*(5\*x - 3)\*(5\*x - 4))/16



**Part 2:**

## Code:

X = [0.0;0.2;0.4;0.8;1.0];

Y = [0.0;7.78;10.68;8.37;3.97;0.0];

n=6;

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

y1 = D(6,1)

y2 = s\*D(5,2)

y3 = s\*(s+1)\*D(4,3)/2

y4 = s\*(s+1)\*(s+2)\*D(3,4)/factorial(3)

y5 = s\*(s+1)\*(s+2)\*(s+3)\*D(2,5)/factorial(4)

y6 = s\*(s+1)\*(s+2)\*(s+3)\*(s+4)\*D(1,6)/factorial(5)

P5 = y1 + y2 + y3 + y4 + y5 + y6

P5 = simplify(P5)

x= 0.9

P5\_at\_x = eval(P5)

## Output:

P5 =

(43\*x\*(5\*x + 1))/40 - (397\*x)/20 + (21\*x\*(5\*x + 1)\*(5\*x + 2))/10 - (x\*(5\*x + 1)\*(5\*x + 2)\*(5\*x + 3))/8 - (27\*x\*(5\*x + 1)\*(5\*x + 2)\*(5\*x + 3)\*(5\*x + 4))/160

