# Department of Computing

# MATH 333: Numerical Analysis

# Class: BSCS-8C

# Lab 9: Numerical Differentiation Based on Newton’s Polynomial Interpolation

# Date: April 6th, 2022

# Time: 10:00 am - 1:00 pm

# Lab Engineer: Anum Asif

|  |
| --- |
| **SUBMITTED BY:**  **Fatima Seemab**  **291310**  **Lab 9** |

# Lab 9: Numerical Differentiation based on Newton’s Polynomial Interpolation

**Introduction**

Matlab represents polynomials with a vector of coefficients. The length of the vector will always be one more than the order of the polynomial

y = x2 + 5x -3 is represented as [1 5 -3]

y = -x3/3 + x is represented as [-1/3 0 1 0]

**Objectives**

The purpose of this lab is to get familiar with Numerical Differentiation basics based on Newton’s Polynomial Interpolation and online learning.

**Tools/Software Requirement**

Matlab R2016a, MS 365, Matlab online

**Description**

**Numerical Differentiation Based on Newton’s Polynomial Interpolation**

In [numerical analysis](https://en.wikipedia.org/wiki/Numerical_analysis), numerical differentiation describes [algorithms](https://en.wikipedia.org/wiki/Algorithm) for estimating the [derivative](https://en.wikipedia.org/wiki/Derivative) of a [mathematical function](https://en.wikipedia.org/wiki/Mathematical_function) or function [subroutine](https://en.wikipedia.org/wiki/Subroutine) using values of the function and perhaps other knowledge about the function.

For this lab take any polynomial or other funnction and implement interpolation and take difference and show output for the points.

**Steps:**

1. Make Script for numerical differentiation and interpolation.
2. Functions includes the implementation of differentiation and interpolation you did in Lab 8.
3. Function input parameters include input data. Input data is a function i.e (x^2+4) or any other function and data points which can be nay number.
4. Take screen shots of your outputs and workspace.

**Helping code for Newton’s Interpolation:**

Code for newtint()

**this link for help with interpolation**

**https://www.**

function [yi, p, pval] = newtint(x, y, xi, c)

% NEWTINT Interpolation of equally-spaced points.

% NEWTINT(X,Y,XI,C) interpolates to find YI, the value

% of the underlying function Y at the point XI, using

% either Newton's forward interpolation formula or

% Newton's backward interpolation formula.

%

% C specifies the interpolation method:

% 0 : Newton's forward or backward (default)

% 1 : Newton's forward

% 2 : Newton's backward

%

% [YI,P,PVAL] = NEWTINT() also returns P, the coefficients

% of the calculated interpolating polynomial, and PVAL,

% which specifies the maximum degree of the interpolating

% polynomial.

% Joe Henning - Fall 2011

if (nargin < 3)

c = 0;

end

pval = length(x)\*(0+1) - 1;

n = length(x)-1;

D = zeros(n+1,n+1);

D(:,1) = y(:);

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

for i = 1:n

for j = 1:i

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

end

end

if (c == 0)

if (xi < x(1))

c = 2;

elseif (xi > x(ceil((n+1)/2)) && xi <= x(n+1))

c = 2;

else

c = 1;

end

end

switch c

case 1

% forward difference interpolation

p = diag(D);

q = (xi-x(1))/h;

yi = p(1);

for i = 1:length(p)-1

term = 1;

for k = 0:(i-1)

term = term\*(q-k);

end

term = term/factorial(i);

yi = yi + term\*p(i+1);

end

case 2

% backward difference interpolation

p = D(n+1,:).';

q = (xi-x(n+1))/h;

yi = p(1);

for i = 1:length(p)-1

term = 1;

for k = 0:(i-1)

term = term\*(q+k);

end

term = term/factorial(i);

yi = yi + term\*p(i+1);

end

end

**or you can use codewithc.com/newtons-interpolation-in-matlab/**

**Lab Task:**

|  |
| --- |
| **Q.1** Write a MATLAB program for Numerical Differentiation Based on Newton's Interpolating  Polynomials. Explain with an example.  **Note**: Use different functions and data points and consider at least 7 data points. E.g.  xi = 0.6    x = [0 0.1 0.2 0.3 0.4 0.5 0.6];  y = [0 0.16 0.32 0.56 0.84 0.96 1.23]; |

**CODE:**

|  |
| --- |
| clear  clc  p = input("Please enter the value of x at which u want to differentiate : ");  x = [0 0.1 0.2 0.3 0.4 0.5 0.6];  y = [0 0.16 0.32 0.56 0.84 0.96 1.23];  h = input("Please enter the value of h: ");  % disp(x(4))  if p== x(1)  if h==0.1  % Three point start points end point formula  solution = (1/(2\*h))\*(-3\*y(1)+4\*y(2)-1\*y(3))  end  elseif p== x(7)  % Three point end point formula  if h==0.1  solution = (1/(2\*-h))\*(-3\*y(7)+4\*y(6)-y(5))  end  elseif p== x(4)  if h==0.1  % Three point mid point formula  solution = (1/(2\*h))\*(y(1)-y(7))  end  end |

**OUTPUT**

**Using starting point applying Three Point Formula:**

**Text

Description automatically generated**

**Using starting point applying Three Point End Point Formula:**

**Text

Description automatically generated**

**Using mid point applying Three Point End Point Formula:**

**Text

Description automatically generated**

**Deliverables**

Submit single word file with matlab code and screen shot of Output.