**BME 313L: Introduction to Numerical Methods in Biomedical Engineering**

**Lab Report**

**Lab #11: Numerical Differentiation**

**Last name, First name: Swindall, Nathan**

**EID: nts279**

**Lab Section: Friday**

**Problem 1. From textbook problem 21.16**

Employ the MATLAB function *diff* to

(1) Plot the second derivative vs. for x=-2: 0.1: 2.

(2) Approximate where .

**Things to discuss**: (100 word minimum for each question, 50 word minimum for discussing what you learned, what was reinforced)

(1) What is the difference between diff and gradient?

(2) What is the relationship between the length of increment and truncation error.

**MATLAB code**

%prepare the workstation

clear all

close all

clc

%initial values given

x=-2.1:.1:2.1; %range of x values for function

y=(1/sqrt(2\*pi))\*exp(-x.^2/2); %function given

xm=-2.05:0.1:2.05; %midpoint of xvalues

xmm=-2:.1:2; %midpoint of the midpoint values

% The first and second derivative

dydx=diff(y)./diff(x); %first derivative

d2ydx2=diff(dydx)./diff(xm); %seconde derivative

%plot of the data

plot(xmm,d2ydx2); %plots d2yd2x vs xmm

xlabel('x') %label for x

ylabel('d2ydx2') %label for y

title('d2yx2 vs x') %title

**MATLAB function**

%prepare the workstation

clear all

close all

clc

%initial values given

x=-2.1:.1:2.1; %range of x values for function

y=(1/sqrt(2\*pi))\*exp(-x.^2/2); %function given

xm=-2.05:0.1:2.05; %midpoint of xvalues

xmm=-2:.1:2; %midpoint of the midpoint values

% The first and second derivative

dydx=diff(y)./diff(x); %first derivative

d2ydx2=diff(dydx)./diff(xm); %seconde derivative

%plot of the data

plot(xmm,d2ydx2); %plots d2yd2x vs xmm

xlabel('x') %label for x

ylabel('d2ydx2') %label for y

title('d2yx2 vs x') %title

**Results**

****

**Discussion**

**Problem 2. From textbook problem 21.19**

Calculate derivative of at x=2 with four different formulas:

1. Improved forward finite difference approximation
2. Improved backward finite difference approximation
3. Centered finite difference approximation
4. Improved centered finite difference approximation

Change the increment ‘*h*’ from 0.5 to 0.01 with -0.01 intervals (dx=0.5:-0.01:0.01). Generate 4 plots in one graph *(improved forward, improved backward, improved centered, centered finite difference approximation* vs *dx*).

**Things to discuss**: (100 word minimum for each question, 50 word minimum for discussing what you learned, what was reinforced)

(1) How could we improve these finite difference approximations?

(2) Which finite difference approximation could give you the most accurate estimate?

**MATLAB code**

%Prepare workstation

clear all

close all

clc

%initialization

f=@(x) exp(-2\*x)-x; %equation given in problem

xo=2; %value we are trying to find the derivative of

h=.5:-.01:.01; %range of different increment values

dx=h; %relabeling the h vector.

for i=1:length(dx)

improvedforward(i)= (-f(xo+2\*h(i))+4\*f(xo+h(i))-3\*f(xo))/(2\*h(i)); %Improved forward finite difference approximation

improvedbackward(i)=(3\*f(xo)-4\*f(xo-h(i))+f(xo-2\*h(i)))/(2\*h(i)); %Improved backward finite difference approximation

centeredfinite(i)= (f(xo+h(i))-f(xo-h(i)))/(2\*h(i)); %Centered finite difference approximation

improvedcentered(i)=(-f(xo+2\*h(i))+8\*f(xo+h(i))-8\*f(xo-h(i))+f(xo-2\*h(i)))/(12\*h(i)); %Improved centered finite difference approximation

end

%plotting the data

plot(dx,improvedforward,dx, improvedbackward,dx,centeredfinite,dx,improvedcentered)

%Above is the plot of all the different methods

%Below are all the labels for the graph

xlabel('increment size')

ylabel('Derivative values')

title('Derivative values vs increment size')

legend('Improvedforward','Improvedbackward', 'Centeredfinite', 'Improvedcentered')

**MATLAB function**

%Prepare workstation

clear all

close all

clc

%initialization

f=@(x) exp(-2\*x)-x; %equation given in problem

xo=2; %value we are trying to find the derivative of

h=.5:-.01:.01; %range of different increment values

dx=h; %relabeling the h vector.

for i=1:length(dx)

improvedforward(i)= (-f(xo+2\*h(i))+4\*f(xo+h(i))-3\*f(xo))/(2\*h(i)); %Improved forward finite difference approximation

improvedbackward(i)=(3\*f(xo)-4\*f(xo-h(i))+f(xo-2\*h(i)))/(2\*h(i)); %Improved backward finite difference approximation

centeredfinite(i)= (f(xo+h(i))-f(xo-h(i)))/(2\*h(i)); %Centered finite difference approximation

improvedcentered(i)=(-f(xo+2\*h(i))+8\*f(xo+h(i))-8\*f(xo-h(i))+f(xo-2\*h(i)))/(12\*h(i)); %Improved centered finite difference approximation

end

%plotting the data

plot(dx,improvedforward,dx, improvedbackward,dx,centeredfinite,dx,improvedcentered)

%Above is the plot of all the different methods

%Below are all the labels for the graph

xlabel('increment size')

ylabel('Derivative values')

title('Derivative values vs increment size')

legend('Improvedforward','Improvedbackward', 'Centeredfinite', 'Improvedcentered')

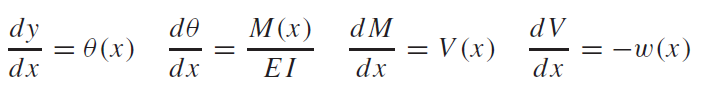
**Results**

**[‘**

**Discussion**

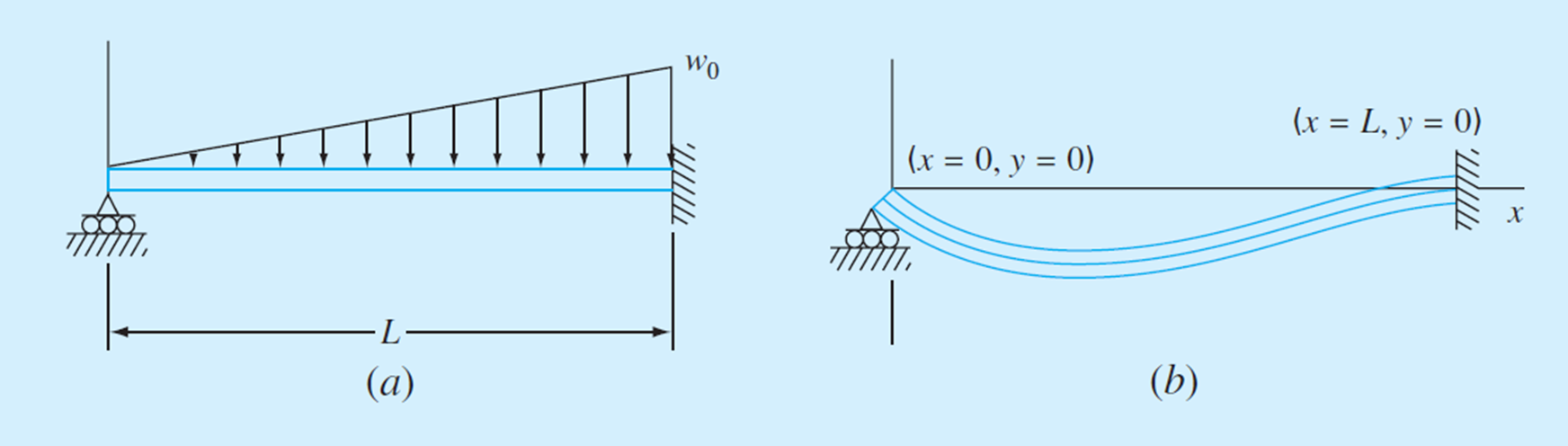
**Problem 3 From textbook problem 21.37**

The following relationships can be used to analyze uniform beams subject to distributed loads:

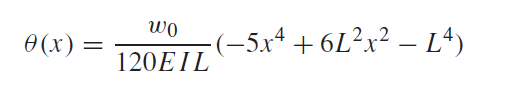


where *x* = distance along beam (m), *y* = deflection (m), (*x*) = slope (m/m), *E* = modulus of elasticity (Pa = N/m2), *I* = moment of inertia (m4), *M*(*x*) = moment (N m), *V*(x) = shear (N), and *w*(*x*) = distributed load (N/m).

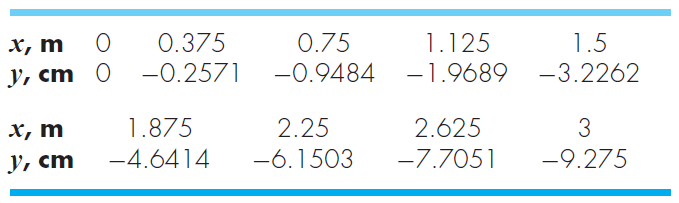
For the case of a linearly increasing load (Fig. 1).



The slope can be computed analytically as



You measure the following deflections along the length of a simply supported uniform beam



Employ numerical differentiation to compute the slope, the moment (in N m), the shear (in N) and the distributed load (in N/m). Use the following parameter values in your computation: E = 200 GPa, and I = 0.0003 m4.

**Things to discuss**: (100 word minimum for each question, 50 word minimum for discussing what you learned, what was reinforced)

(1) What kind of finite difference approximation would you use for the data points on boundaries? Why?

**MATLAB code**

**MATLAB function**

**Results**

**Discussion**