## horizontal line



Question 5:

Finite Difference Method

11.11.2020 (Date of Presentation)

**─**

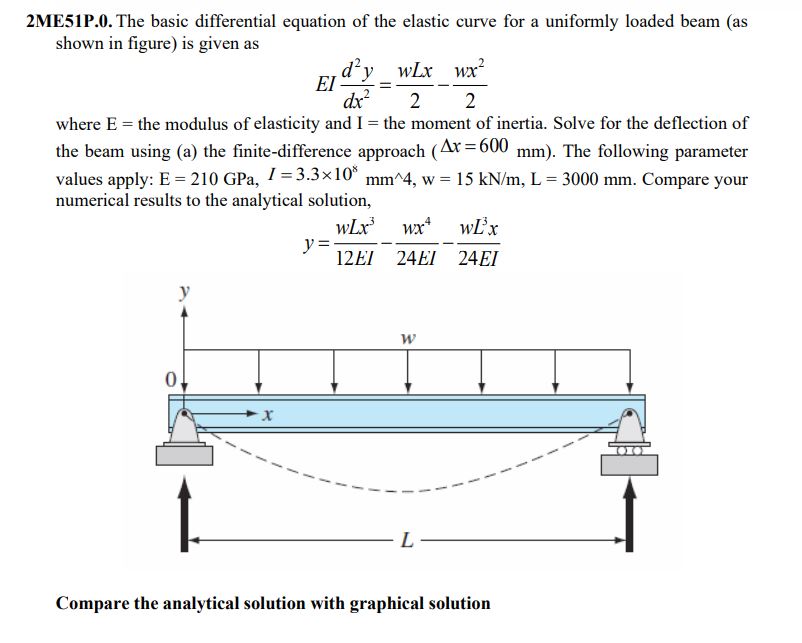
19ME439 Keivalya Pandya

19ME441 Varshil Patel

**Group Number: 72**

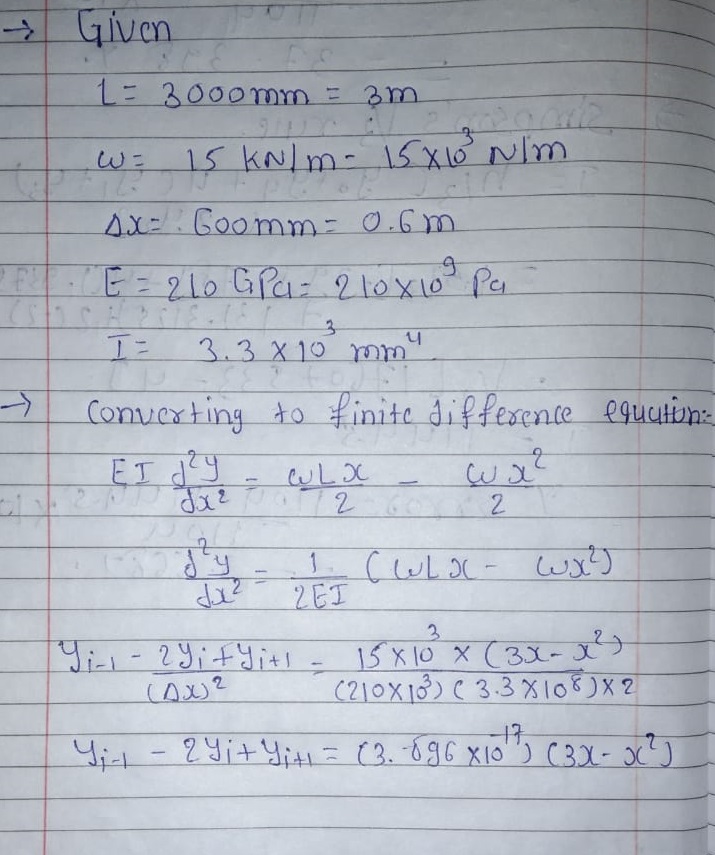
# Problem Statement

***Solve the deflection of the beam using the FINITE DIFFERENCE approach. Compare numerical results to the analytical solutions, also using a graph.***



# 

# Numerical Solution



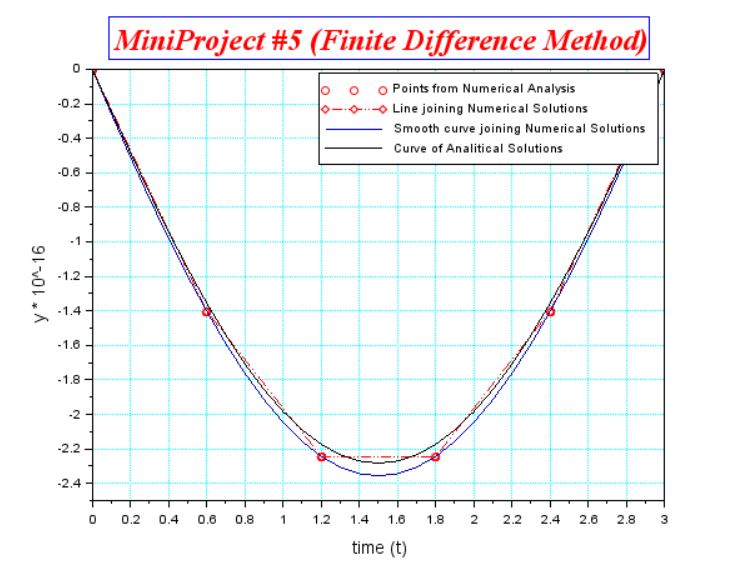
# SCILAB Code

|  |
| --- |
| clc;  *// Initialization - declaring variables*  L=3; *// in meters*  h=0.6; *// in meters*  N=(L/h) - 1; *// N = 4;*  CN = [0 0]; *// +*  A=zeros(N); *// +*  w=15\*10^3; *//converted kN/m to N/m*  E=210\*10^9; *// Converted GPa to Pa*  I=3.3\*10^8; *// in mm^4*  *// NUMERICAL*  *// Constant from: y'' = constant\*(Lx - x^2) by rearranging given formula*  constant = w\*(h^2)/(2\*E\*I);  *//constant = 3.896D-17*  *// For loop to create matrices*  for i=1:N  x=i\*h;  *// for Matrix B*  B(i) = constant\*(3\*x-x^2);  *// for Matrix A*  for j=1:N  if i==j  A(i,j)=-2;  else  if j==i+1  A(i,j)=1;  elseif j==i-1  A(i,j) = 1;  end  end  end  end  M=[A, B]; *// Augmented Matrix*  printf("\n \tAugmented Matrix from finite-diff Method which is to be solved");  disp(M);  Mx=rref(M); *//converts matrix M into RREF(reduced row ec. form)*  printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*", 's');  printf("\n\n \tReduced Row-Echelon Form of Matrix");  disp(Mx);  Mx(:,$); *// ($) selects last element - Hence this is last element of every row(:)*  YN = [CN(1) Mx(:,$)' CN(2) ]; *// NUMERICAL SOLUTIONS*  printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*", 's');  printf("\n\n",'s');  printf("\t OUR SOLUTIONS are as follows\n",'s');  for i=1:length(YN)  printf("\tThe %d solution is \t %e\n",i,YN(i));  end  *// Plotting for Numerical Solutions*  xdata = 0:0.6:3; *// X-axis points for Numerical*  ydata = YN\*10^16; *// Storing Numerical Solutions (magnified)*  xlabel("time (t)","fontsize", 3);  ylabel("y \* 10^-16","fontsize", 3);  title("MiniProject #5",'color','red','edgecolor','blue','fontsize',5, 'fontname', 'times bold italic');  xgrid(4);  *// NUMERICAL ENDS*  *// ANALYTICAL*  function y\_anal=g(x)  y\_anal = (w\*L\*(x^3) - (w\*(x^4)/2) - (w\*(L^3)\*x/2))/(E\*I\*12);  endfunction  funcprot(0); *// to avoid warning shown by scilab: redefining g(x)*  for i=0:0.1:3  y\_analytical(10\*i + 1)=g(i); *// Analytical Solutions*  end  y\_analytical = y\_analytical\*10^16; *// Storing Analytical Solutions, (magnified)*  printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*", 's');  printf("\n\n\t Numerical Analytical Error\n",'s');  for i=0:0.6:3  n = i\*(10/6)+1;  *//err(1) = 0;*  *//err(6) = 0;*  err(n) = abs((YN(n) - g(i))/g(i))\*100; *// ERROR Calculation*  if n == 1 || n == 6 *// at first and last node, error = 0*  err(n) = 0; *// NUM = ANA = 0 at {0, 3}*  end  printf("\t %e \t %e \t %f%%\n",YN(n),g(i),err(n));  end  *// Plotting of Analytical Graph*  x\_analytical=[0:0.1:3] *// X-axis for Analytical*  plot(xdata',ydata',"ro"); *// Points for Numerical Solution*  plot(xdata',ydata','diamondred:'); *// Dot-Dashed line joining Numerical Solution*  yi=smooth([xdata;ydata],0.1); *// obtaining smooth curve joining Numerical Solutions*  plot2d(yi(1,:)',yi(2,:)',10); *// Plotting entire Numerical portion*  plot2d(x\_analytical',y\_analytical', 1); *// Plotting entire Analytical Portion*  a = gca(); *// get current axes*  a.data\_bounds = [0 -2.5; 3 0]; *// Bounding X-axis and Y-axis to magnify our graph*  *// ANALYTICAL ENDS*  legend("Points from Numerical Analysis", "Line joining Numerical Solutions", "Smooth curve joining Numerical Solutions", "Curve of Analytical Solutions"); |

**Link:** [GitHub Folder of my Project](https://github.com/keivalya/2ME01/tree/master/Mini%20Project)

**Link to SciLab Code:** [Mini Project CODE](https://github.com/keivalya/2ME01/blob/master/Mini%20Project/MiniProject.sce)

# Output Graph



# Conclusion

1. *Finite difference method* is **fairly accurate** (only ~3% error in each value)
2. The graph is nearly **quadratic** in nature within the range [0, 3]