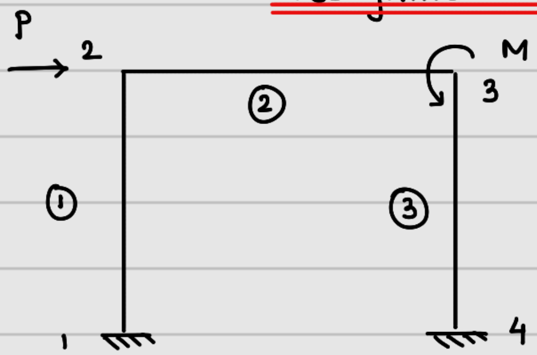
CH20B012

Anish Anand Pophale

Assignment 6

Q1)



At 1 and 4, essential boundary conditions are given as

$u_1 = v_1 = \theta_1 = u_4 = v_4 = \theta_4 = 0$

The element equations in LCS are

$k_l \, d_l = F_l$ where,

$$k_l = \begin{bmatrix} EA/L & 0 & 0 & -EA/L & 0 & 0 \\ 0 & 12EI/L^3 & 6EI/L^2 & 0 & -12EI/L^3 & 6EI/L^2 \\ 0 & 6EI/L^2 & 4EI/L & 0 & -6EI/L^2 & 2EI/L \\ -EA/L & 0 & 0 & EA/L & 0 & 0 \\ 0 & -12EI/L^3 & -6EI/L^2 & 0 & 12EI/L^3 & -6EI/L^2 \\ 0 & 6EI/L^2 & 2EI/L & 0 & -6EI/L^2 & 4EI/L \end{bmatrix}$$

The transformation matrix T is given as

$$T = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \cos\theta & \sin\theta & 0 \\ 0 & 0 & 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

In the GCS, $kd = F$ where,

$K = T^T k_l T$ and $d = T^T d_l$

The global force vector can be written as a 12×1 vector

$$F^T = [\,0 \quad 0 \quad 0 \quad P \quad 0 \quad 0 \quad 0 \quad 0 \quad M \quad 0 \quad 0 \quad 0\,]$$

Each node has 3 DOF and there are 4 nodes

$\therefore$ K is 12×12 and d, F are 12×1

To find force in element 1, we first convert d to d_ℓ for element 1.

$$d_\ell = T d$$

$$\text{Axial force} = \frac{EA}{L}\left[d_\ell(4) - d_\ell(1)\right] = \frac{EA}{L}\left[u_2 - u_1\right]$$

Bending moment =

$$EI\left[-6 + 12s \quad L(-4 + 16s) \quad 6 - 12s \quad L(-2 + 6s)\right]\begin{bmatrix} v_1 \\ \theta_1 \\ v_2 \\ \theta_2 \end{bmatrix}$$

Here, s is used for mapping as $0 \leq s \leq 1$. So we get bending moment in element 1 as a function of s.

This is solved using the MATLAB code given below along with the results.

Q1)

MATLAB Code:

```
clc; clear all; close all; format compact; format shortg;

%Constants
E = 200*10^9;
A = 6400*10^-6;
I = 1.2*10^5*10^-12;
P = 10*10^3;
L = 10;
M = 5*10^3;
EA = E*A; EI = E*I;

nodes = [0,0;0,L;L,L;L,0];
conn =[1,2;2,3;3,4];
n = size(nodes,1); %Number of nodes
N = size(conn,1); %Number of elements

%Assembling the global stiffness matrix
k_global = zeros(3*n,3*n);
for i = 1:N
    node1 = conn(i,1); node2 = conn(i,2);
    x1 = nodes(node1,1); x2 = nodes(node2,1);
    y1 = nodes(node1,2); y2 = nodes(node2,2);

    k = PlaneFrameElement(L,EA,EI,x1,y1,x2,y2);
    k_global(3*node1-2:3*node1, 3*node1-2:3*node1) = k_global(3*node1-2:3*node1,
3*node1-2:3*node1) + k(1:3,1:3);
    k_global(3*node1-2:3*node1, 3*node2-2:3*node2) = k_global(3*node1-2:3*node1,
3*node2-2:3*node2) + k(1:3,4:6);
    k_global(3*node2-2:3*node2, 3*node1-2:3*node1) = k_global(3*node2-2:3*node2,
3*node1-2:3*node1) + k(4:6,1:3);
    k_global(3*node2-2:3*node2, 3*node2-2:3*node2) = k_global(3*node2-2:3*node2,
3*node2-2:3*node2) + k(4:6,4:6);
end

%External force vector
F =  zeros(3*n,1);
F(4) = P;
F(9) = M;

%Removing rows and columns for EBCs
k_global_r = k_global;
k_global_r([1:3,end-2:end],:) = [];
k_global_r(:,[1:3,end-2:end]) = [];

F_r = F;
F_r([1:3,end-2:end]) = [];

%Solving for the nodal displacement and slopes
d = zeros(3*n,1);
d_r = k_global_r\F_r;
d(4:end-3) = d_r;

fprintf("Displacement and rotation at nodes 2 and 3:\n")
fprintf("Displacement of node 2 in X direction is %f m\n",d(4));
fprintf("Displacement of node 2 in Y direction is %f m\n",d(5));
```

```
fprintf("Rotation at node 2 is is %f\n",d(6));
fprintf("Displacement of node 3 in X direction is %f m\n",d(7));
fprintf("Displacement of node 3 in Y direction is %f m\n",d(8));
fprintf("Rotation of node 3 is %f\n\n",d(9));

%Forces and moments in element 1
node1 = conn(1,1); node2 = conn(1,2);
x1 = nodes(node1,1); x2 = nodes(node2,1);
y1 = nodes(node1,2); y2 = nodes(node2,2);
d1 = d(1:6);
[F1,M1] = ForceMoment(L,EA,EI,x1,y1,x2,y2,d1);
M1 = vpa(M1,5);

fprintf("Force and moment in element 1:\n")
fprintf("Axial force in element 1 is %f N\n",F1);
fprintf("Bending moment in element 1 in terms of s is %s Nm\n",char(M1));

function k = PlaneFrameElement(L,EA,EI,x1,y1,x2,y2)
    %Direction cosines for the element
    ls = (x2-x1)/L; ms=(y2-y1)/L;

    %Local stiffness matrix
    kl = [EA/L, 0, 0, -EA/L, 0, 0;
          0, 12*EI/L^3, 6*EI/L^2, 0, -12*EI/L^3, 6*EI/L^2;
          0, 6*EI/L^2, 4*EI/L, 0, -6*EI/L^2, 2*EI/L;
          -EA/L, 0, 0, EA/L, 0, 0;
          0, -12*EI/L^3, -6*EI/L^2, 0, 12*EI/L^3, -6*EI/L^2;
          0, 6*EI/L^2, 2*EI/L, 0, -6*EI/L^2, 4*EI/L];

    %Transformation matrix
    T = eye(6);
    T(1:2,1:2) = [ls,ms;-ms,ls];
    T(4:5,4:5) = [ls,ms;-ms,ls];

    %Global stiffness matrix
    k = T'*kl*T;
end

function [F,M] = ForceMoment(L,EA,EI,x1,y1,x2,y2,d)
    %Direction cosines for the element
    ls = (x2-x1)/L; ms=(y2-y1)/L;

    %Transformation matrix
    T = eye(6);
    T(1:2,1:2) = [ls,ms;-ms,ls];
    T(4:5,4:5) = [ls,ms;-ms,ls];

    %Local d vector
    dl = T*d;
    %Force
    F = EA/L*(dl(4)-dl(1));
    %Moment
    syms s;
    M = EI*[-6+12*s, L*(-4+16*s), 6-12*s, L*(-2 + 6*s)]*[dl(2);dl(3);dl(5);dl(6)];
end
```

Code Output:

Command Window

```
Displacement and rotation at nodes 2 and 3:
Displacement of node 2 in X direction is 24.057586 m
Displacement of node 2 in Y direction is 0.000030 m
Rotation at node 2 is is -1.512904
Displacement of node 3 in X direction is 24.057545 m
Displacement of node 3 in Y direction is -0.000030 m
Rotation of node 3 is -1.165678

Force and moment in element 1:
Axial force in element 1 is 3857.140378 N
Bending moment in element 1 in terms of s is 4.75e+6*s - 2.7381e+6 Nm
fx >>
```

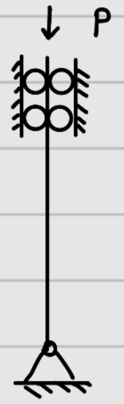
- The displacement at nodes 2 and 3 in the X direction is very high which is because of the large value of L and small value of I. Due to this, the associated stiffness which is directly proportional to EI and inversely proportional to L decreases and the displacement is high.
- Instead, if we reduce the length of each element to 1m or increase the moment I by a few orders of magnitudes, we get the following results. Using I = 1.2*10^8 mm^2,

Command Window

```
Displacement and rotation at nodes 2 and 3:
Displacement of node 2 in X direction is 0.024104 m
Displacement of node 2 in Y direction is 0.000030 m
Rotation at node 2 is is -0.001520
Displacement of node 3 in X direction is 0.024063 m
Displacement of node 3 in Y direction is -0.000030 m
Rotation of node 3 is -0.001169

Force and moment in element 1:
Axial force in element 1 is 3854.664858 N
Bending moment in element 1 in terms of s is 4.753e+6*s - 2.7413e+6 Nm
fx >>
```

Q2)



$A = 300\text{ mm} \times 500\text{ m}$

$I = \dfrac{bh^3}{12}$

As we need to take I_{min}, we use

$I_{min} = \dfrac{500 \times 300^3}{12}\text{ mm}^4$

For this problem, the element equation is

$$\left\{ \frac{1}{l}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \frac{Pl}{EI}\begin{bmatrix} 1/3 & 1/6 \\ 1/6 & 1/3 \end{bmatrix} \right\} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = 0$$

Assembling the global equations, we will get

$(K + P K_P)\, d = 0$

we solve $K + P K_P = 0$ as an eigen value problem and use the smallest eigen values as the critical buckling load P.

From Euler theory of beams, we know that

$$P_{cr} = \frac{\pi^2 EI}{L^2}$$

we increase the number of elements till we get the following criteria

$\dfrac{|P - P_{cr}|}{P_{cr}} < 10^{-3}$ (For which N = 29 elements are needed)

this is done using MATLAB, the code and the results are given below.

Q2)

MATLAB Code:

```
clc; clear all; close all; format compact; format shortg;

%Constants
L = 12;
E = 200*10^9;
A = 300*500*10^-6;
I =  (500*300^3*10^-12)/12; %I_min = bh^3/12, h = 300 mm
syms P;

%Analytical solution for critical load
P_actual = pi^2*E*I/L^2;
err = 1;
N = 1;
P_vec = [];

%Increasing number of elements till error in P critial is below 10^-3
while err > 10^-3
    N = N+1; %Number of elements
    n = N+1; %Number of nodes
    l = L/N; %Length of each element

    %Constructing the global matrices
    ke_global = zeros(n,n);
    kp_global = sym(zeros(n,n));
    for i = 1:N
        [ke,kp] = buckling(P,l,E,I);
        ke_global(i:i+1,i:i+1) = ke_global(i:i+1,i:i+1) + ke;
        kp_global(i:i+1,i:i+1) = kp_global(i:i+1,i:i+1) + kp;
    end
    k_global = sym(ke_global) - kp_global;

    %Removing rows and columns for EBC
    k_global([1,end],:) = [];
    k_global(:,[1,end]) = [];

    %Solving the eigen value problem to find P_cr
    P_cr = vpa(min(solve(det(k_global) == 0)),5);
    P_vec(N-1) = P_cr;

    %Calculating the absolute error
    err = abs(P_cr-P_actual)/P_actual;
end

fprintf("The converged value for the critical buckling load is %.3f kN
\n",P_cr/1000)
fprintf("The analytical value for the critical buckling load is %.3f kN
\n",P_actual/1000)
fprintf("The relative error is %f\n",err)

figure;
plot(2:1:N,P_vec,'-or',LineWidth=1.3)
xlabel("Number of elements")
ylabel("Critical Load")
title("Critical load vs Number of elements")
```

```
function [ke, kp] = buckling(P,L,E,I)
    ke = 1/L*[1, -1; -1, 1];
    kp = (P*L)/(E*I)*[1/3, 1/6; 1/6, 1/3];
end
```

Code Output:

```
Command Window
The converged value for the critical buckling load is 15436.344 kN
The analytical value for the critical buckling load is 15421.257 kN
The relative error is 0.000978
fx >>
```

Hence, the critical buckling load is **15436.344 kN**.

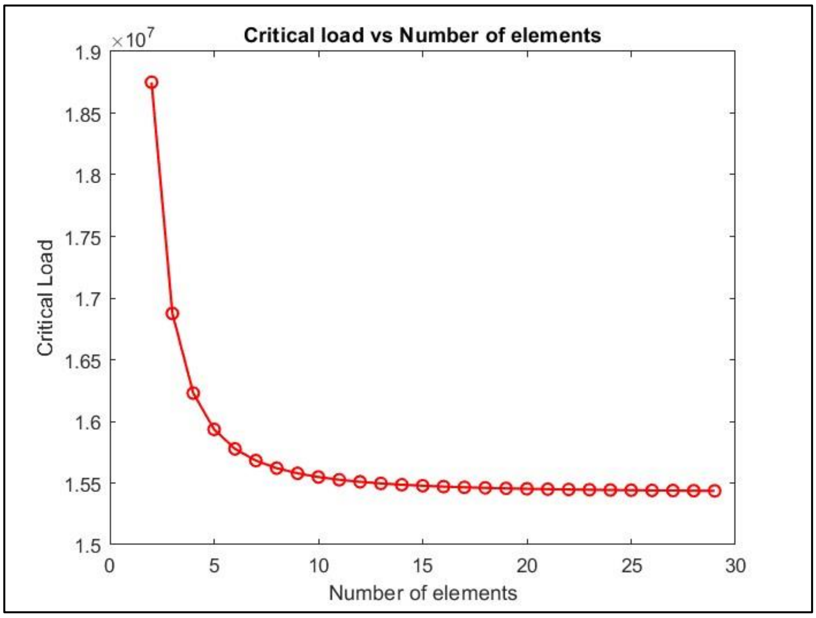


Fig 1: Convergence of the critical load with increasing number of elements