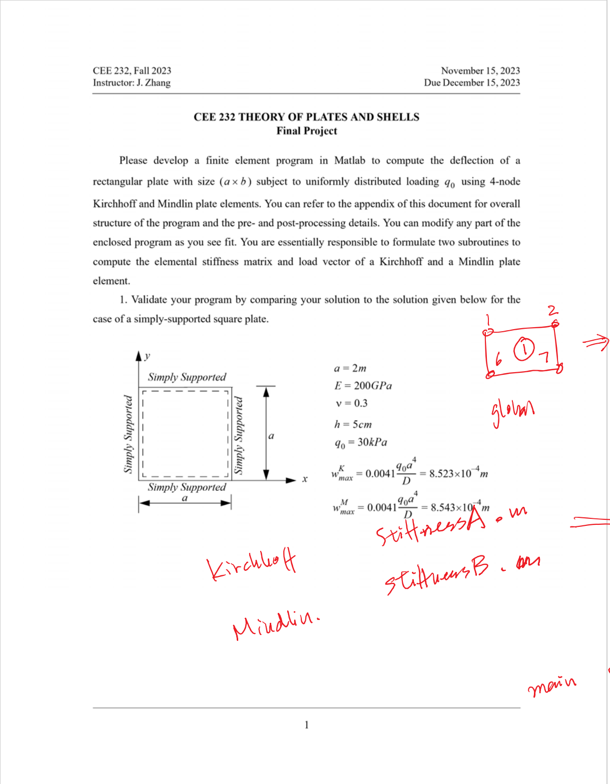
Project

Friday, September 15, 2023 10:15 PM

CEE232_project_description_F23



CEE 232, Fall 2023 November 15, 2023
Instructor: J. Zhang Due December 15, 2023

CEE 232 THEORY OF PLATES AND SHELLS
Final Project

Please develop a finite element program in Matlab to compute the deflection of a rectangular plate with size ($a \times b$) subject to uniformly distributed loading q_0 using 4-node Kirchhoff and Mindlin plate elements. You can refer to the appendix of this document for overall structure of the program and the pre- and post-processing details. You can modify any part of the enclosed program as you see fit. You are essentially responsible to formulate two subroutines to compute the elemental stiffness matrix and load vector of a Kirchhoff and a Mindlin plate element.

1. Validate your program by comparing your solution to the solution given below for the case of a simply-supported square plate.

$a = 2m$
$E = 200GPa$
$\nu = 0.3$
$h = 5cm$
$q_0 = 30kPa$

$w^K_{max} = 0.0041\frac{q_0 a^4}{D} = 8.523\times10^{-4}m$

$w^M_{max} = 0.0041\frac{q_0 a^4}{D} = 8.543\times10^{-4}m$

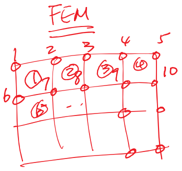
1

1 2 6 7 ① glbal

Kirchhoff stiffnessA.m

Mindlin. stiffnessB.m

main

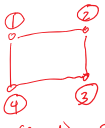


pre-processing

① Discretize

Node # : 1, 2, 3. - - - . . nnode

element # : 1 [1 2 7 6]
2 [2 3 8 7]
⋮
nnel [ ]



(local) $3\times4=12$

At each node

DOFs $\left(w \quad \frac{\partial w}{\partial x} \quad \frac{\partial w}{\partial y}\right)$ Kirchhoff

$(w \quad \theta_x \quad \theta_y)$ Mindlin

② Element

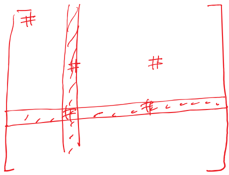
element stiffness $[Ke]_{12\times12}$

element nodal force $\{Pe\}_{12\times1}$

③ Assembly $[Ke] \Rightarrow [K_G]$

$\{Pe\} \Rightarrow \{P_G\}$

DOF constrained



$(3\times nnode) \times (3\times nnode)$

④ Apply boundary conditions

$[K_G] \xrightarrow{B.C.s} [K_{RG}]$

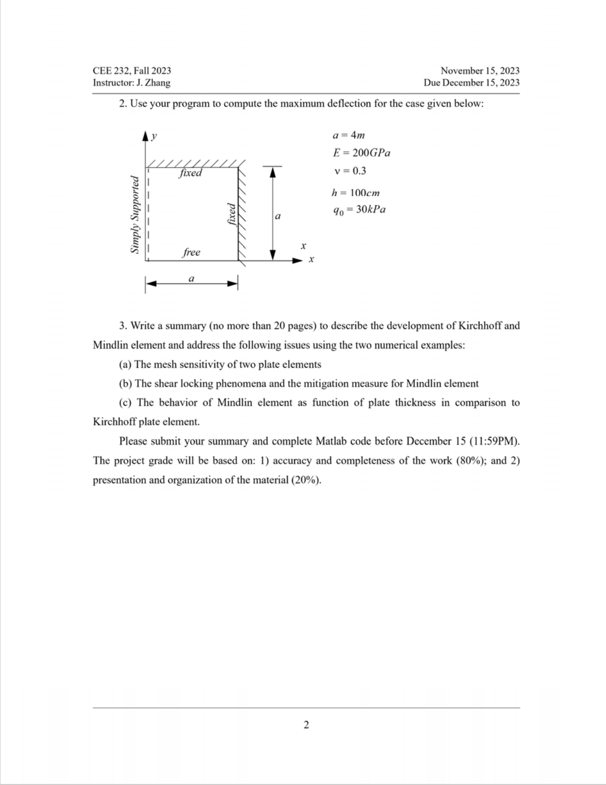
$(nnode\times DOF)\times(nnode\times DOF)$ $(nnode\times DOF - \text{constrained DOFs})\times(nnode\times DOF - \text{constrained DOFs}$

$\{P_G\} \xrightarrow{B.C.s.} \{P_{RG}\}_{(nnode\times DOF - \text{constrained DOFs})\times1}$

⑤ solve $[K_{RG}]\{u\} = \{P_{RG}\} \Rightarrow \{u\} = \text{unconstrained DOFs.}$

⑥ post-processing

$\{u\} \rightarrow$ stresses, strains. plots



CEE 232, Fall 2023 November 15, 2023
Instructor: J. Zhang Due December 15, 2023

2. Use your program to compute the maximum deflection for the case given below:

y
x
x
fixed
fixed
free
Simply Supported
a
a

$a = 4m$
$E = 200GPa$
$\nu = 0.3$
$h = 100cm$
$q_0 = 30kPa$

3. Write a summary (no more than 20 pages) to describe the development of Kirchhoff and Mindlin element and address the following issues using the two numerical examples:

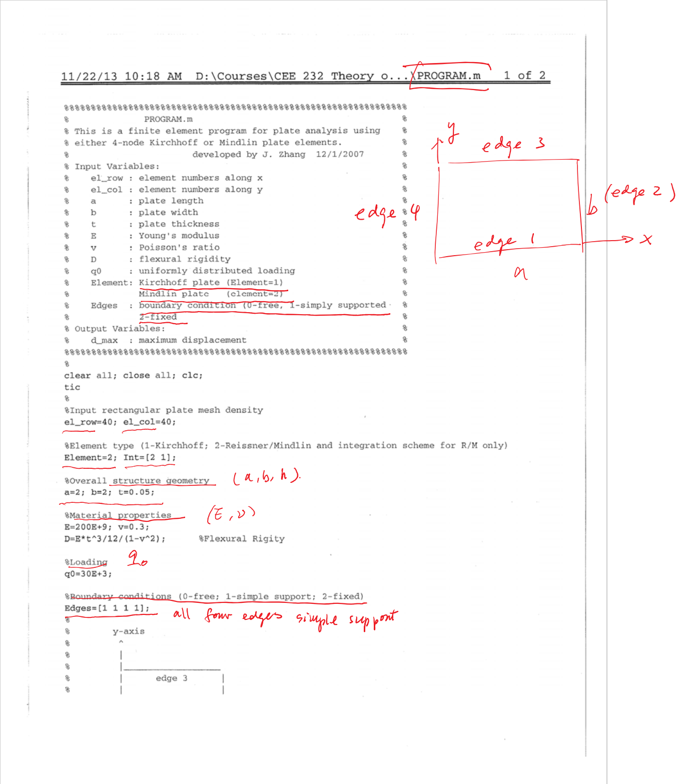
(a) The mesh sensitivity of two plate elements

(b) The shear locking phenomena and the mitigation measure for Mindlin element

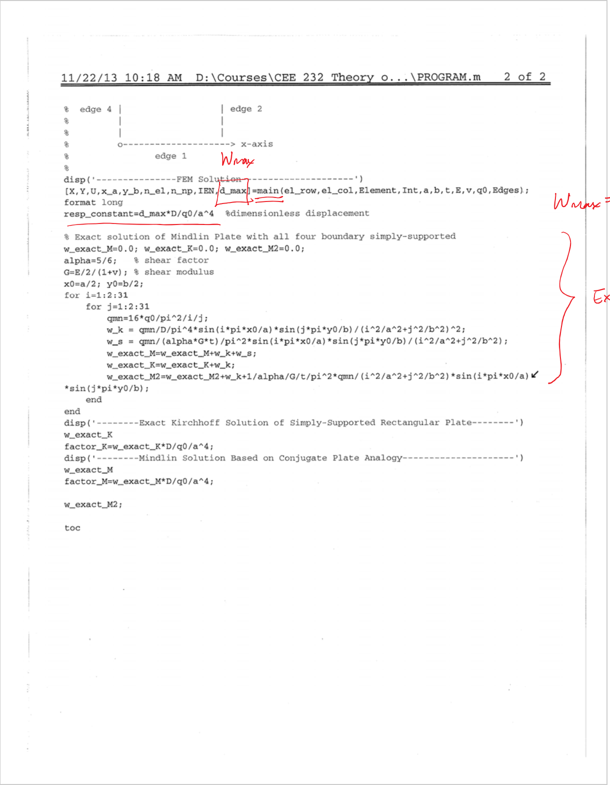
(c) The behavior of Mindlin element as function of plate thickness in comparison to Kirchhoff plate element.

Please submit your summary and complete Matlab code before December 15 (11:59PM). The project grade will be based on: 1) accuracy and completeness of the work (80%); and 2) presentation and organization of the material (20%).

2


y
edge 3
edge 4
b (edge 2)
edge 1
x
a

```
11/22/13 10:18 AM  D:\Courses\CEE 232 Theory o...\PROGRAM.m   1 of 2

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%               PROGRAM.m                                         %
% This is a finite element program for plate analysis using      %
% either 4-node Kirchhoff or Mindlin plate elements.              %
%                         developed by J. Zhang  12/1/2007        %
% Input Variables:                                                %
%    el_row : element numbers along x                             %
%    el_col : element numbers along y                             %
%    a      : plate length                                        %
%    b      : plate width                                         %
%    t      : plate thickness                                     %
%    E      : Young's modulus                                     %
%    v      : Poisson's ratio                                     %
%    D      : flexural rigidity                                   %
%    q0     : uniformly distributed loading                       %
%    Element: Kirchhoff plate (Element=1)                         %
%             Mindlin plate   (element=2)                         %
%    Edges  : boundary condition (0-free, 1-simply supported      %
%             2-fixed                                             %
% Output Variables:                                               %
%    d_max  : maximum displacement                                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
clear all; close all; clc;
tic
%
%Input rectangular plate mesh density
el_row=40; el_col=40;

%Element type (1-Kirchhoff; 2-Reissner/Mindlin and integration scheme for R/M only)
Element=2; Int=[2 1];

%Overall structure geometry    (a, b, h).
a=2; b=2; t=0.05;

%Material properties     (E, ν)
E=200E+9; v=0.3;
D=E*t^3/12/(1-v^2);      %Flexural Rigity

%Loading   q0
q0=30E+3;

%Boundary conditions (0-free; 1-simple support; 2-fixed)
Edges=[1 1 1 1];   all four edges simple support
%
%        y-axis
%         ^
%         |
%         |_________________
%         |      edge 3     |
%         |                 |
```

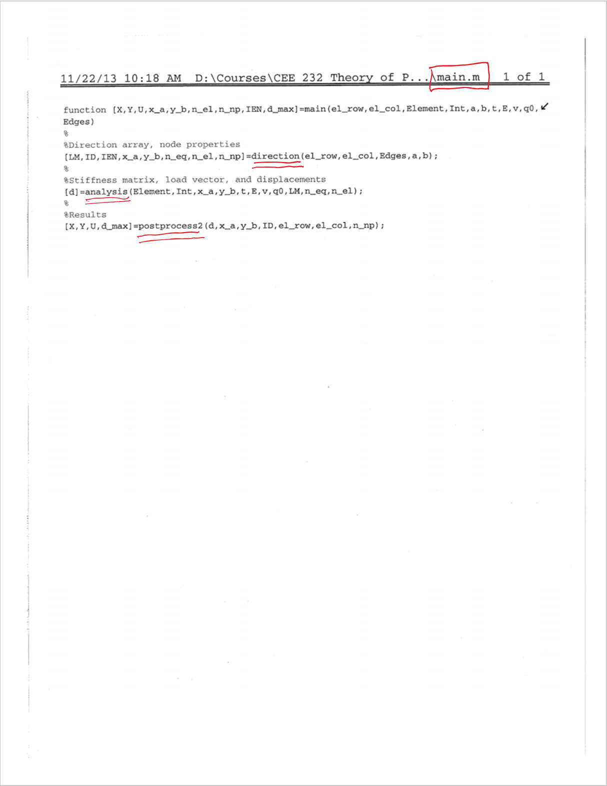


```
11/22/13 10:18 AM  D:\Courses\CEE 232 Theory o...\PROGRAM.m   2 of 2

%  edge 4 |                 | edge 2
%         |                 |
%         |                 |
%         o-------------------> x-axis
%              edge 1        Wmax
%
disp('---------------FEM Solution--------------------')
[X,Y,U,x_a,y_b,n_el,n_np,IEN,d_max]=main(el_row,el_col,Element,Int,a,b,t,E,v,q0,Edges);
format long
resp_constant=d_max*D/q0/a^4  %dimensionless displacement

% Exact solution of Mindlin Plate with all four boundary simply-supported
w_exact_M=0.0; w_exact_K=0.0; w_exact_M2=0.0;
alpha=5/6;   % shear factor
G=E/2/(1+v); % shear modulus
x0=a/2; y0=b/2;
for i=1:2:31
    for j=1:2:31
        qmn=16*q0/pi^2/i/j;
        w_k = qmn/D/pi^4*sin(i*pi*x0/a)*sin(j*pi*y0/b)/(i^2/a^2+j^2/b^2)^2;
        w_s = qmn/(alpha*G*t)/pi^2*sin(i*pi*x0/a)*sin(j*pi*y0/b)/(i^2/a^2+j^2/b^2);
        w_exact_M=w_exact_M+w_k+w_s;
        w_exact_K=w_exact_K+w_k;
        w_exact_M2=w_exact_M2+w_k+1/alpha/G/t/pi^2*qmn/(i^2/a^2+j^2/b^2)*sin(i*pi*x0/a)↙
*sin(j*pi*y0/b);
    end
end
disp('--------Exact Kirchhoff Solution of Simply-Supported Rectangular Plate--------')
w_exact_K
factor_K=w_exact_K*D/q0/a^4;
disp('--------Mindlin Solution Based on Conjugate Plate Analogy---------------------')
w_exact_M
factor_M=w_exact_M*D/q0/a^4;

w_exact_M2;

toc
```

response constant

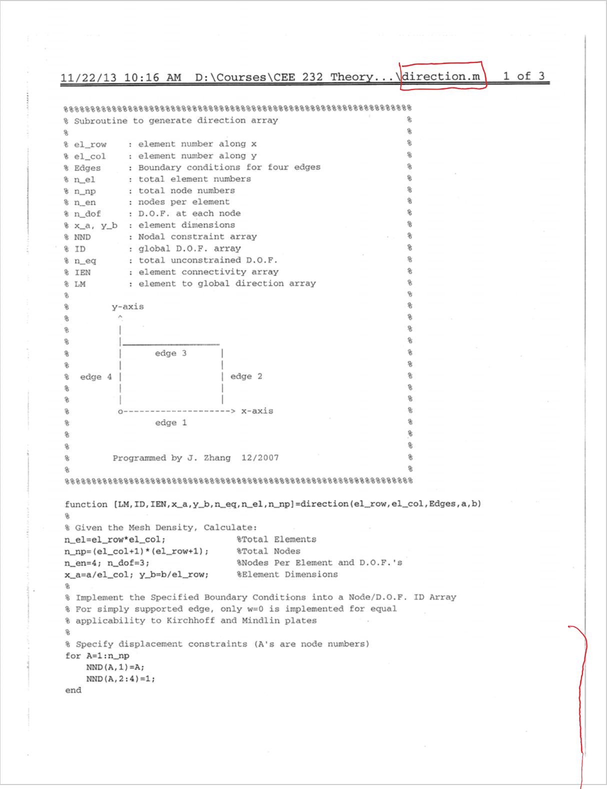
$$W_{max} = 8\frac{q_0 a^4}{D}$$

Exact solution for Mindlin plate. (all four edge simple supported).



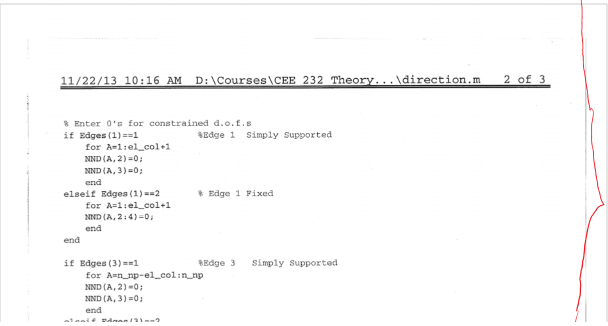
11/22/13 10:18 AM D:\Courses\CEE 232 Theory of P...\main.m 1 of 1

```
function [X,Y,U,x_a,y_b,n_el,n_np,IEN,d_max]=main(el_row,el_col,Element,Int,a,b,t,E,v,q0,↙
Edges)
%
%Direction array, node properties
[LM,ID,IEN,x_a,y_b,n_eq,n_el,n_np]=direction(el_row,el_col,Edges,a,b);
%
%Stiffness matrix, load vector, and displacements
[d]=analysis(Element,Int,x_a,y_b,t,E,v,q0,LM,n_eq,n_el);
%
%Results
[X,Y,U,d_max]=postprocess2(d,x_a,y_b,ID,el_row,el_col,n_np);
```



11/22/13 10:16 AM D:\Courses\CEE 232 Theory...\direction.m 1 of 3

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Subroutine to generate direction array                         %
%                                                                %
% el_row     : element number along x                            %
% el_col     : element number along y                            %
% Edges      : Boundary conditions for four edges                %
% n_el       : total element numbers                             %
% n_np       : total node numbers                                %
% n_en       : nodes per element                                 %
% n_dof      : D.O.F. at each node                               %
% x_a, y_b   : element dimensions                                %
% NND        : Nodal constraint array                            %
% ID         : global D.O.F. array                               %
% n_eq       : total unconstrained D.O.F.                        %
% IEN        : element connectivity array                        %
% LM         : element to global direction array                 %
%                                                                %
%        y-axis                                                  %
%          ^                                                     %
%          |                                                     %
%          |__________________                                   %
%          |      edge 3      |                                  %
%          |                  |                                  %
%  edge 4  |                  | edge 2                           %
%          |                  |                                  %
%          |                  |                                  %
%          o-------------------> x-axis                          %
%                 edge 1                                         %
%                                                                %
%                                                                %
%        Programmed by J. Zhang  12/2007                         %
%                                                                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [LM,ID,IEN,x_a,y_b,n_eq,n_el,n_np]=direction(el_row,el_col,Edges,a,b)
%
% Given the Mesh Density, Calculate:
n_el=el_row*el_col;              %Total Elements
n_np=(el_col+1)*(el_row+1);      %Total Nodes
n_en=4; n_dof=3;                 %Nodes Per Element and D.O.F.'s
x_a=a/el_col; y_b=b/el_row;      %Element Dimensions
%
% Implement the Specified Boundary Conditions into a Node/D.O.F. ID Array
% For simply supported edge, only w=0 is implemented for equal
% applicability to Kirchhoff and Mindlin plates
%
% Specify displacement constraints (A's are node numbers)
for A=1:n_np
    NND(A,1)=A;
    NND(A,2:4)=1;
end
```

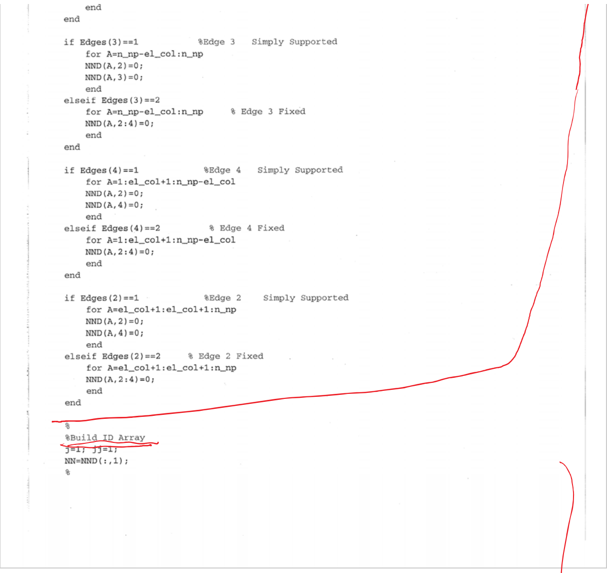


11/22/13 10:16 AM D:\Courses\CEE 232 Theory...\direction.m 2 of 3

```
% Enter 0's for constrained d.o.f.s
if Edges(1)==1           %Edge 1  Simply Supported
    for A=1:el_col+1
    NND(A,2)=0;
    NND(A,3)=0;
    end
elseif Edges(1)==2       % Edge 1 Fixed
    for A=1:el_col+1
    NND(A,2:4)=0;
    end
end

if Edges(3)==1           %Edge 3   Simply Supported
    for A=n_np-el_col:n_np
    NND(A,2)=0;
    NND(A,3)=0;
    end
elseif Edges(3)==2
```

$$\begin{array}{cccc} node & w & \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} \\ 1 & 1 & 0 & 0 \\ & 0 & 0 & 0 \end{array}$$

fixed.



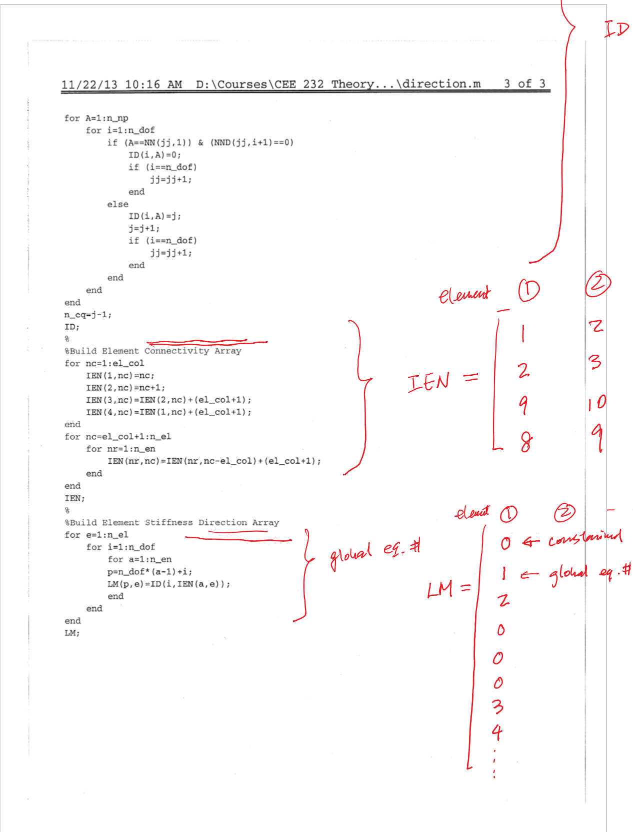
```
    end
end

if Edges(3)==1           %Edge 3   Simply Supported
    for A=n_np-el_col:n_np
    NND(A,2)=0;
    NND(A,3)=0;
    end
elseif Edges(3)==2
    for A=n_np-el_col:n_np     % Edge 3 Fixed
    NND(A,2:4)=0;
    end
end

if Edges(4)==1            %Edge 4   Simply Supported
    for A=1:el_col+1:n_np-el_col
    NND(A,2)=0;
    NND(A,4)=0;
    end
elseif Edges(4)==2          % Edge 4 Fixed
    for A=1:el_col+1:n_np-el_col
    NND(A,2:4)=0;
    end
end

if Edges(2)==1            %Edge 2    Simply Supported
    for A=el_col+1:el_col+1:n_np
    NND(A,2)=0;
    NND(A,4)=0;
    end
elseif Edges(2)==2     % Edge 2 Fixed
    for A=el_col+1:el_col+1:n_np
    NND(A,2:4)=0;
    end
end

%
%Build ID Array
j=1; jj=1;
NN=NND(:,1);
%
```

$$NND=\begin{array}{c|ccc}\text{node} & w & \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y}\\ \hline 1 & 1 & 0 & 0\\ 2 & 0 & 0 & 0\\ \vdots & & & \\ \text{nnode} & & & \end{array}\quad \text{fixed.}$$

* all constrained DOFs are assigned "0" to it

$$ID=\begin{array}{cccccc|c}\text{node} & 1 & 2 & 3 & \cdots & n\_np & \text{DOF}\\ \hline & 0 & & & & & 1\\ & \text{eq.\#} & & & & & 2\\ & \text{eq.\#} & & & & & 3\end{array}$$

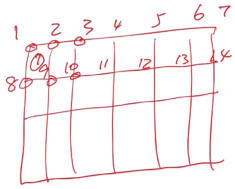
* give eq. #s for all unconstrained DOFs.



11/22/13 10:16 AM D:\Courses\CEE 232 Theory...\direction.m 3 of 3

```
for A=1:n_np
    for i=1:n_dof
        if (A==NN(jj,1)) & (NND(jj,i+1)==0)
            ID(i,A)=0;
            if (i==n_dof)
                jj=jj+1;
            end
        else
            ID(i,A)=j;
            j=j+1;
            if (i==n_dof)
                jj=jj+1;
            end
        end
    end
end
n_eq=j-1;
ID;
%
%Build Element Connectivity Array
for nc=1:el_col
    IEN(1,nc)=nc;
    IEN(2,nc)=nc+1;
    IEN(3,nc)=IEN(2,nc)+(el_col+1);
    IEN(4,nc)=IEN(1,nc)+(el_col+1);
end
for nc=el_col+1:n_el
    for nr=1:n_en
        IEN(nr,nc)=IEN(nr,nc-el_col)+(el_col+1);
    end
end
IEN;
%
%Build Element Stiffness Direction Array
for e=1:n_el
    for i=1:n_dof
        for a=1:n_en
            p=n_dof*(a-1)+i;
            LM(p,e)=ID(i,IEN(a,e));
        end
    end
end
LM;
```

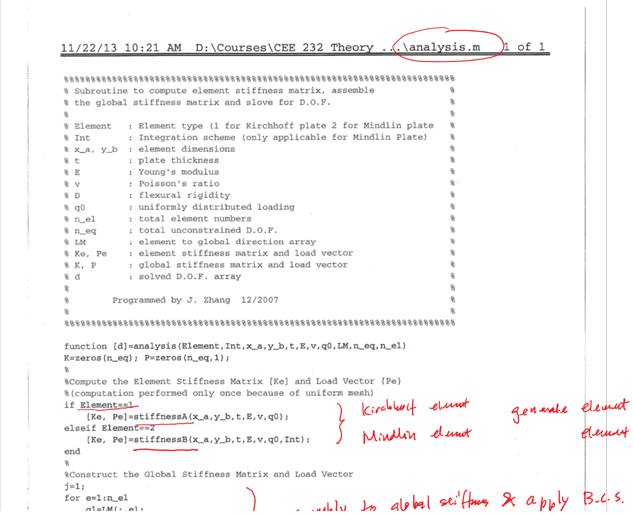
$$IEN=\begin{array}{cccc|c}\text{element} & ① & ② & \cdots\ \text{nel} & \\ \hline & 1 & 2 & & \text{node } 1\\ & 2 & 3 & & \text{node } 2\\ & 9 & 10 & & \text{node } 3\\ & 8 & 9 & & \text{node } 4\end{array}$$



global eq. #

$$LM=\begin{array}{cccc|c}\text{element} & ① & ② & \cdots\ \text{nel} & \text{D.O.F.}\\ \hline & 0\ \leftarrow \text{constrained} & & & 1\\ & 1\ \leftarrow \text{global eq.\#} & & & 2\\ & 2 & & & 3\\ & 0 & & & 4\\ & 0 & & & 5\\ & 0 & & & 6\\ & 3 & & & 7\\ & 4 & & & 8\\ & \vdots & & & 9\\ & & & & 10\\ & & & & 11\\ & & & & 12\end{array}$$

1, 2, 3 } node 1
4, 5, 6 } node 2
7, 8, 9 } node 3
10, 11, 12 } node 4



11/22/13 10:21 AM D:\Courses\CEE 232 Theory ...\analysis.m 1 of 1

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Subroutine to compute element stiffness matrix, assemble              %
% the global stiffness matrix and slove for D.O.F.                      %
%                                                                       %
% Element   : Element type (1 for Kirchhoff plate 2 for Mindlin plate   %
% Int       : Integration scheme (only applicable for Mindlin Plate)    %
% x_a, y_b  : element dimensions                                        %
% t         : plate thickness                                           %
% E         : Young's modulus                                           %
% v         : Poisson's ratio                                           %
% D         : flexural rigidity                                         %
% q0        : uniformly distributed loading                             %
% n_el      : total element numbers                                     %
% n_eq      : total unconstrained D.O.F.                                %
% LM        : element to global direction array                         %
% Ke, Pe    : element stiffness matrix and load vector                  %
% K, P      : global stiffness matrix and load vector                   %
% d         : solved D.O.F. array                                       %
%                                                                       %
%        Programmed by J. Zhang  12/2007                                %
%                                                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [d]=analysis(Element,Int,x_a,y_b,t,E,v,q0,LM,n_eq,n_el)
K=zeros(n_eq); P=zeros(n_eq,1);
%
%Compute the Element Stiffness Matrix [Ke] and Load Vector {Pe}
%(computation performed only once because of uniform mesh)
if Element==1
    [Ke, Pe]=stiffnessA(x_a,y_b,t,E,v,q0);
elseif Element==2
    [Ke, Pe]=stiffnessB(x_a,y_b,t,E,v,q0,Int);
end
%
%Construct the Global Stiffness Matrix and Load Vector
j=1;
for e=1:n_el
    g1=LM(:,e);
```

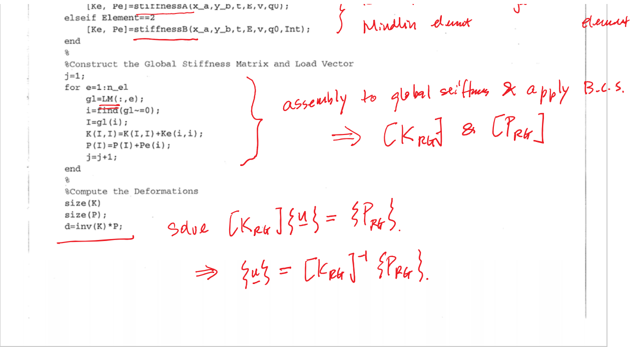
Kirchhoff element
Mindlin element

generate element stiffness matrix [Ke]
element load vector {Pe}.

mbly to global stiffness & apply B.C.s.

$$\begin{Bmatrix} w_1\\ \frac{\partial w_1}{\partial x}\\ \frac{\partial w_1}{\partial y}\\ \vdots \end{Bmatrix}\qquad \begin{Bmatrix} \#\\ 0\\ 0\\ \# \end{Bmatrix}$$

node free in w



```
    [Ke, Pe]=stiffnessA(x_a,y_b,t,E,v,q0);
elseif Element==2
    [Ke, Pe]=stiffnessB(x_a,y_b,t,E,v,q0,Int);
end
%
%Construct the Global Stiffness Matrix and Load Vector
j=1;
for e=1:n_el
    gl=LM(:,e);
    i=find(gl~=0);
    I=gl(i);
    K(I,I)=K(I,I)+Ke(i,i);
    P(I)=P(I)+Pe(i);
    j=j+1;
end
%
%Compute the Deformations
size(K)
size(P);
d=inv(K)*P;
```

} Mindlin element

element load vector $\{P_e\}$.

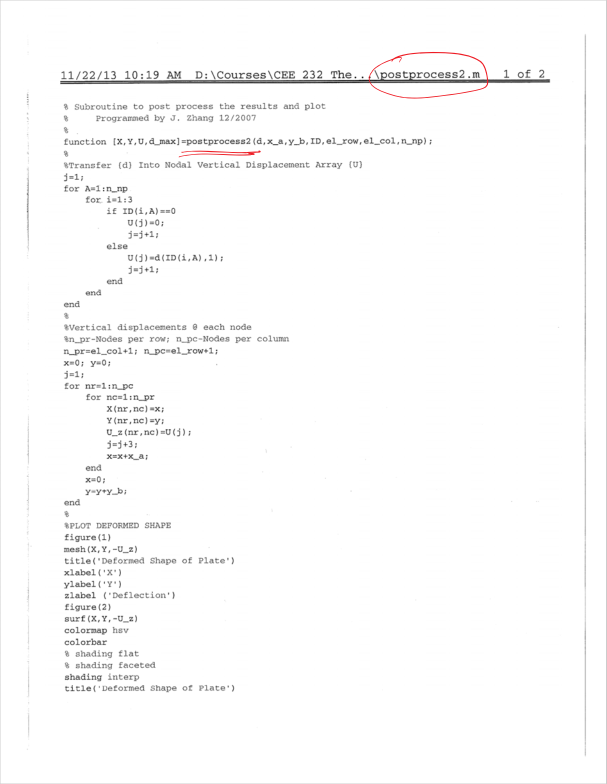
assembly to global stiffness & apply B.C.s.

$\Rightarrow [K_{RG}]$ & $[P_{RG}]$

solve $[K_{RG}]\{\underline{u}\} = \{P_{RG}\}$.

$\Rightarrow \{\underline{u}\} = [K_{RG}]^{-1}\{P_{RG}\}$.

$$\begin{Bmatrix} w_1 \\ \frac{\partial w_1}{\partial x} \\ \frac{\partial w_1}{\partial y} \\ \vdots \\ w_4 \\ \frac{\partial w_4}{\partial x} \\ \frac{\partial w_4}{\partial y} \end{Bmatrix} \qquad \begin{Bmatrix} 0 \\ 0 \\ \# \\ 0 \\ 0 \\ \# \\ 0 \\ 0 \\ \# \\ 0 \\ 0 \end{Bmatrix} \; 12\times 1$$



11/22/13 10:19 AM D:\Courses\CEE 232 The...\postprocess2.m   1 of 2

```
% Subroutine to post process the results and plot
%     Programmed by J. Zhang 12/2007
%
function [X,Y,U,d_max]=postprocess2(d,x_a,y_b,ID,el_row,el_col,n_np);
%
%Transfer {d} Into Nodal Vertical Displacement Array {U}
j=1;
for A=1:n_np
    for i=1:3
        if ID(i,A)==0
            U(j)=0;
            j=j+1;
        else
            U(j)=d(ID(i,A),1);
            j=j+1;
        end
    end
end
%
%Vertical displacements @ each node
%n_pr-Nodes per row; n_pc-Nodes per column
n_pr=el_col+1; n_pc=el_row+1;
x=0; y=0;
j=1;
for nr=1:n_pc
    for nc=1:n_pr
        X(nr,nc)=x;
        Y(nr,nc)=y;
        U_z(nr,nc)=U(j);
        j=j+3;
        x=x+x_a;
    end
    x=0;
    y=y+y_b;
end
%
%PLOT DEFORMED SHAPE
figure(1)
mesh(X,Y,-U_z)
title('Deformed Shape of Plate')
xlabel('X')
ylabel('Y')
zlabel ('Deflection')
figure(2)
surf(X,Y,-U_z)
colormap hsv
colorbar
% shading flat
% shading faceted
shading interp
title('Deformed Shape of Plate')
```

```
xlabel('X')
ylabel('Y')
zlabel ('Deflection')
%Find Maximum Deflection
format long
d_max=max(max(U_z(:,:)))
```