Ritz method for the analysis of thin plates

Course of Spacecraft Structures

AY 2017/2018

Riccardo Vescovini

Politecnico di Milano, Department of Aerospace Science and Technology

Trial functions – Overview

- The out-of-plane displacement is expanded by separation of variables as:

$$w = \sum_{ij}^{RS} c_{ij} \phi_i(x) \psi_j(y)$$

- Trial functions (along x direction; y-direction analagous by replacing x with y and a with b)
  - Clamped-Clamped (CC)

$$\phi_i = \left(\frac{x}{a}\right)^{i+3} - 2\left(\frac{x}{a}\right)^{i+2} + \left(\frac{x}{a}\right)^{i+i} \qquad \text{or} \qquad \phi_i = 1 - \cos\frac{2i\pi x}{a}$$

  - Clamped-Free (CF)

$$\phi_i = \left(\frac{x}{a}\right)^{i+1}$$

  - Clamped-Supported (CS)

$$\phi_i = \left(1 - \frac{x}{a}\right)\left(\frac{x}{a}\right)^{i+1}$$

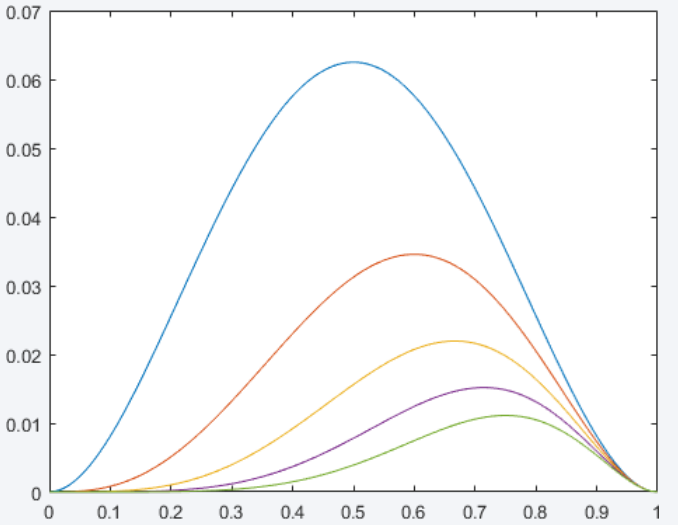
  - Supported-Supported (SS)

$$\phi_i = \left(1 - \frac{x}{a}\right)\left(\frac{x}{a}\right)^{i} \qquad \text{or} \qquad \phi_i = \sin\frac{i\pi x}{a}$$

Trial functions

- Clamped-Clamped (CC)

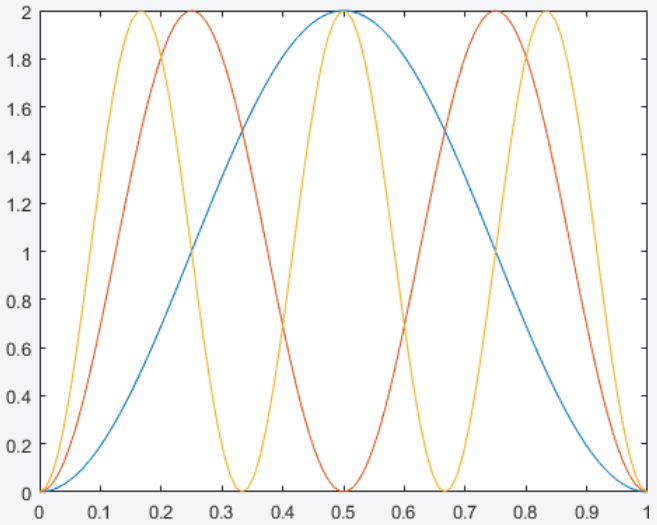
$$\phi_i = \left(\frac{x}{a}\right)^{i+3} - 2\left(\frac{x}{a}\right)^{i+2} + \left(\frac{x}{a}\right)^{i+i} \qquad \text{or} \qquad \phi_i = 1 - \cos\frac{2i\pi x}{a}$$



$\phi = 0$ $\quad$ $\phi = 0$

$\phi_{,x} = 0$ $\quad$ $\phi_{,x} = 0$

polynomial



$\phi = 0$ $\quad$ $\phi = 0$

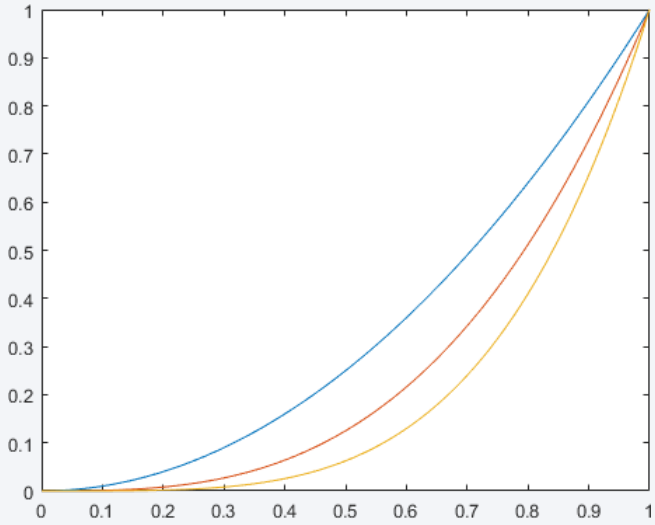
$\phi_{,x} = 0$ $\quad$ $\phi_{,x} = 0$

trigonometric

Trial functions

- Clamped-Free (CF)

$$\phi_i = \left(\frac{x}{a}\right)^{i+1}$$



$\phi = 0$

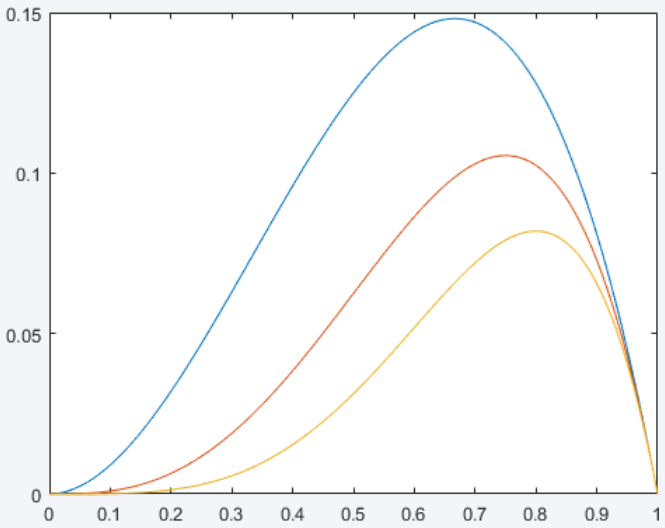
$\phi_{,x} = 0$

polynomial

Trial functions

- Clamped-Supported (CF)

$$\phi_i = \left(1 - \frac{x}{a}\right)\left(\frac{x}{a}\right)^{i+1}$$



$\phi = 0$

$\phi_{,x} = 0$

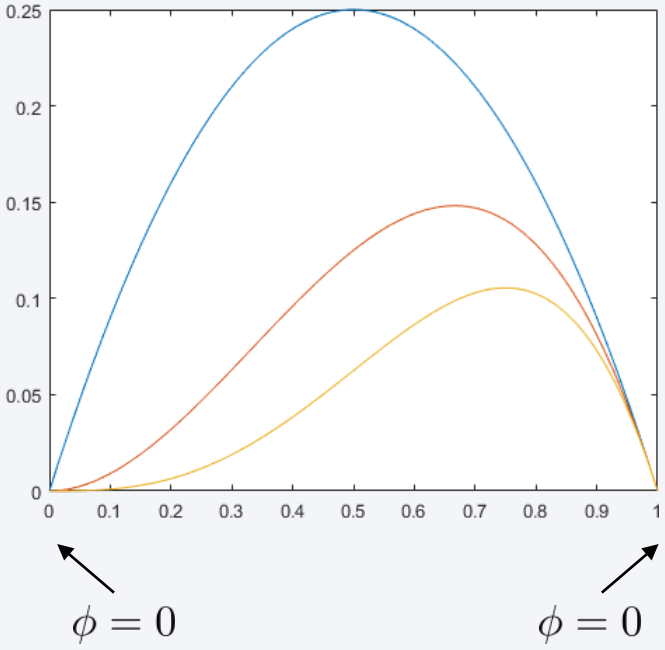
$\phi = 0$

polynomial

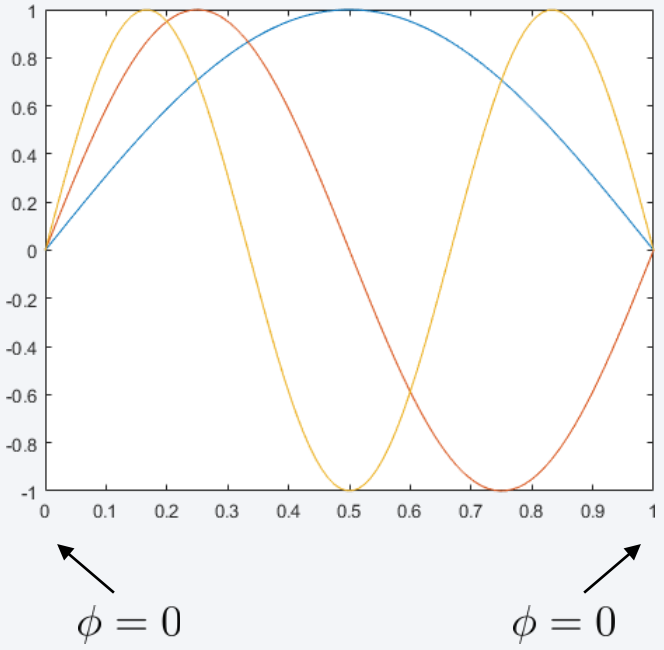
Trial functions

- Supported-Supported (SS)

$$\phi_i = \left(\frac{x}{a} - 1\right)\left(\frac{x}{a}\right)^i \qquad \text{or} \qquad \phi_i = \sin\frac{i\pi x}{a}$$



polynomial



trigonometric

Equation reminder: stiffness matrix

- Stiffness matrix

$$\underset{RS\times RS}{\mathbf{K}} = D_{11}\left(\mathbf{I}_{xx}^{22}\otimes\mathbf{I}_{yy}^{00}\right) + D_{12}\left(\mathbf{I}_{xx}^{20}\otimes\mathbf{I}_{yy}^{02} + \mathbf{I}_{xx}^{02}\otimes\mathbf{I}_{yy}^{20}\right) + \\ + D_{22}\left(\mathbf{I}_{xx}^{00}\otimes\mathbf{I}_{yy}^{22}\right) + 4D_{66}\left(\mathbf{I}_{xx}^{11}\otimes\mathbf{I}_{yy}^{11}\right)$$

- where:

$$\left(\underset{R\times R}{\mathbf{I}_{xx}^{\alpha\beta}}\right)_{ij} = \int_0^a \phi_i^{(\alpha)}\phi_j^{(\beta)}\mathrm{d}x \quad \text{and:} \quad \phi_i^{(\alpha)} = \frac{\mathrm{d}^\alpha\phi_i}{\mathrm{d}x^\alpha} \quad \text{with } i,j = 1\ldots R$$

$$\left(\underset{S\times S}{\mathbf{I}_{yy}^{\alpha\beta}}\right)_{ij} = \int_0^b \psi_i^{(\alpha)}\psi_j^{(\beta)}\mathrm{d}y \quad \text{and:} \quad \psi_i^{(\alpha)} = \frac{\mathrm{d}^\alpha\psi_i}{\mathrm{d}y^\alpha} \quad \text{with } i,j = 1\ldots S$$

Equation reminder: load vector

- Load vector

$$\underset{RS\times 1}{\mathbf{f}} = p \underset{R\times 1}{\mathbf{J}_x} \otimes \underset{S\times 1}{\mathbf{J}_y}$$

- where, for uniform pressure:

$$(\mathbf{J}_x)_i = \int_0^a \phi_i \mathrm{d}x \qquad \text{with} \quad i = 1 \dots R$$

$$(\mathbf{J}_y)_i = \int_0^b \psi_i \mathrm{d}y \qquad \text{with} \quad i = 1 \dots S$$

- and, for concentrated load:

$$(\mathbf{J}_x)_i = \phi_i (x_0) \qquad \text{with} \quad i = 1 \dots R$$

$$(\mathbf{J}_y)_i = \psi_i (y_0) \qquad \text{with} \quad i = 1 \dots S$$

Overview of the program – Main

- The structure of the program consists of following steps:

```
% --- Input data
INPUT = input_model;
[ PLATE, RITZ, LOAD ] = set_model( INPUT );
TRIAL = set_trial_fcs( PLATE.bcs );

% --- Evaluate integrals
IRITZ = evaluate_integrals( TRIAL, PLATE, RITZ, LOAD );

% --- Assembly stiffness matrices
K = assembly_stiffness( PLATE, IRITZ );
F = assembly_load_vector( LOAD, IRITZ );

% --- Solve the linear static problem
RITZ.C = K \ F;

% --- Post-process
... whatever you like
```

Overview of the program – Structures

- The functions used in in the program are PLATE, RITZ, LOAD, TRIAL, IRITZ whose fields are:

```
PLATE =
      a: 400
      b: 200
      E: 72000
     nu: 0.3000
      t: 1
     I0: 2.7000e-09
    bcs: 'CCCC'
      D: [3×3 double]
```

```
a: plate dimension along x
b: plate dimension along y
E: Young's modulus
nu: Poisson's ratio
t: thickness
I0: plate inertia
bcs: boundary conditions
D: bending stiffness
```

$$\mathbf{D} = \frac{Et^3}{12\left(1-\nu^2\right)} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix}$$

Overview of the program – Structures

- The functions used in in the program are PLATE, RITZ, LOAD, TRIAL, IRITZ whose fields are:

```
RITZ =
 struct with fields:
    R: 3                          R: number of trial functions along x
    S: 3                          S: number of trial functions along y
    C: [9×1 double]               C: Ritz amplitudes
```

```
LOAD =
  struct with fields:
            p0: 1                 p0: pressure load intensity
     load_type: 'uniform'         load_type: ‘uniform’ or ‘concentrated’
    load_pos_x0: []               load_pos_x0: x-position of concentrated load (if any)
    load_pos_y0: []               load_pos_y0: y-position of concentrated load (if any)
```

Overview of the program – Structures

- The trial functions can be defined as function handles. For instance, in the case of a simply-supported plate, using trigonometric functions, they are defined as:

```
TRIAL =

  struct with fields:

      phi: @(x,a,i)sin(i*pi*x/a)
     dphi: @(x,a,i)(i*pi/a)*cos(i*pi*x/a)
    ddphi: @(x,a,i)-(i*pi/a)^2*sin(i*pi*x/a)
      psi: @(x,a,i)sin(i*pi*x/a)
     dpsi: @(x,a,i)(i*pi/a)*cos(i*pi*x/a)
    ddpsi: @(x,a,i)-(i*pi/a)^2*sin(i*pi*x/a)
```

```
phi: Ritz functions along x
dphi: x-derivative of phi
ddphi: x-derivative of dphi
psi: Ritz functions along y
dpsi: x-derivative of psi
ddpsi: x-derivative of dpsi
```

Overview of the program – Structures

- The Ritz integrals are collected into the structure IRITZ whose fields are:

```
IRITZ =

  struct with fields:

    Ixx_00: [3×3 double]
    Ixx_02: [3×3 double]
    Ixx_11: [3×3 double]
    Ixx_20: [3×3 double]
    Ixx_22: [3×3 double]
    Iyy_00: [3×3 double]
    Iyy_02: [3×3 double]
    Iyy_11: [3×3 double]
    Iyy_20: [3×3 double]
    Iyy_22: [3×3 double]
        Jx: [3×1 double]
        Jy: [3×1 double]
```

```
Ixx_00: int_0^a phi*phi;
Ixx_02: int_0^a phi*ddphi;
... and so on

Jx: Ritz integral along x of load vector
Jy: Ritz integral along y of load vector
```

Definition of the trial functions

```
% --- Input data
INPUT = input_model;
[ PLATE, RITZ, LOAD ] = set_model( INPUT );
TRIAL = set_trial_fcs( PLATE.bcs );

% --- Evaluate integrals
IRITZ = evaluate_integrals( TRIAL, PLATE, RITZ, LOAD );

% --- Assembly stiffness matrices
K = assembly_stiffness( PLATE, IRITZ );
F = assembly_load_vector( LOAD, IRITZ );

% --- Solve the linear static problem
RITZ.C = K \ F;

% --- Post-process
... whatever you like
```

Definition of the trial functions

- In this function, the trial functions are defined as function handles. The choice of the functons along x and y depends on the boundary conditions to be accounted for

```
function TRIAL = set_trial_fcs( bcs )

% ... select the trial functions depending on the boundary conditions
TRIAL.phi   = @(x,a,i)              sin(i*pi*x/a);
TRIAL.dphi  = @(x,a,i)  (i*pi/a)  * cos(i*pi*x/a);
TRIAL.ddphi = @(x,a,i) -(i*pi/a)^2* sin(i*pi*x/a);

TRIAL.psi   = @(x,a,i)              sin(i*pi*x/a);
TRIAL.dpsi  = @(x,a,i)  (i*pi/a)  * cos(i*pi*x/a);
TRIAL.ddpsi = @(x,a,i) -(i*pi/a)^2* sin(i*pi*x/a);
```

- Example relative to trigonometric functions and SSSS boundary conditions

Evaluation of the Ritz integrals

```
% --- Input data
INPUT = input_model;
[ PLATE, RITZ, LOAD ] = set_model( INPUT );
TRIAL = set_trial_fcs( PLATE.bcs );

% --- Evaluate integrals
IRITZ = evaluate_integrals( TRIAL, PLATE, RITZ, LOAD );

% --- Assembly stiffness matrices
K = assembly_stiffness( PLATE, IRITZ );
F = assembly_load_vector( LOAD, IRITZ );

% --- Solve the linear static problem
RITZ.C = K \ F;

% --- Post-process
... whatever you like
```

Evaluation of the Ritz integrals

- The Ritz integrals can be performed numerically using the integral.m function

```
function IRITZ = evaluate_integrals( TRIAL, PLATE, RITZ, LOAD )

a = PLATE.a;
b = PLATE.b;

R = RITZ.R;
S = RITZ.S;

phi   = TRIAL.phi;
dphi  = TRIAL.dphi;
ddphi = TRIAL.ddphi;

for i = 1 : R
     for j = 1 : S
         IRITZ.Ixx_00(i,j) = integral( @(x) phi(x,a,i) .* phi(x,a,j), 0, a );
         % ... and so on
      end
end
```

Assembly of the stiffness matrix

```
% --- Input data
INPUT = input_model;
[ PLATE, RITZ, LOAD ] = set_model( INPUT );
TRIAL = set_trial_fcs(PLATE.bcs);

% --- Evaluate integrals
IRITZ = evaluate_integrals( TRIAL, PLATE, RITZ, LOAD );

% --- Assembly stiffness matrices
K = assembly_stiffness( PLATE, IRITZ );
F = assembly_load_vector( LOAD, IRITZ );

% --- Solve the linear static problem
RITZ.C = K \ F;

% --- Post-process
... whatever you like
```

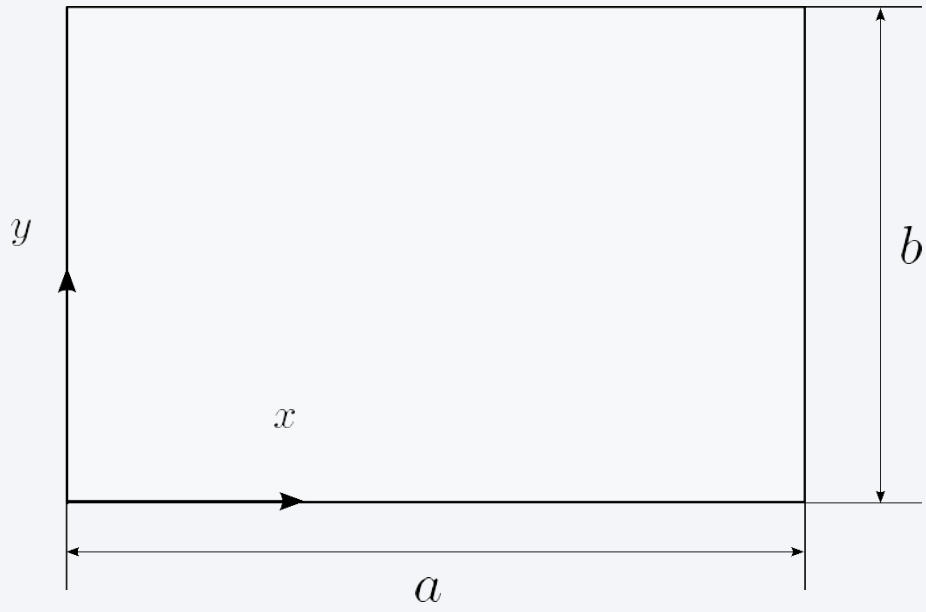
Assembly of the stiffness matrix

- The assembly of the stiffness matrix is straightforward thanks to the use of the Kronecker product

```
function K = assembly_stiffness( PLATE, IRITZ )

D11 = PLATE.D(1,1);
D12 = PLATE.D(1,2);
D22 = PLATE.D(2,2);
D66 = PLATE.D(3,3);

Ixx_22 = IRITZ.Ixx_22;
Ixx_20 = IRITZ.Ixx_20;
Ixx_02 = IRITZ.Ixx_02;
Ixx_00 = IRITZ.Ixx_00;
Ixx_11 = IRITZ.Ixx_11;

Iyy_22 = IRITZ.Iyy_22;
Iyy_20 = IRITZ.Iyy_20;
Iyy_02 = IRITZ.Iyy_02;
Iyy_00 = IRITZ.Iyy_00;
Iyy_11 = IRITZ.Iyy_11;

% --- Build stiffness matrix
K = D11 * kron( Ixx_22, Iyy_00 ) + ...
    D12 * ( kron( Ixx_20, Iyy_02 ) + kron( Ixx_02, Iyy_20 ) ) + ...
    D22 * kron( Ixx_00, Iyy_22 ) + ...
    4 * D66 * kron( Ixx_11, Iyy_11 );
```

Example

- Rectangular plate of aluminum



Input data

a = {100, 200, 400} mm

b = 200 mm

t = 1 mm

E = 72 GPa

v = 0.3

I_0 = 2.7e-9 N/mm^2

p = 1e-3 N/mm^2
(distributed load)

P = 10 N
(concentrated load applied in the middle)

Exercise

- Complete the program to allow the static analysis of plates with SSSS and CCCC conditions

- Extend the program to the free-vibration analysis (exemplary results are reported next for comparison purposes)

- Once the program is validated:
  - study the convergence of the mid-displacement in the case of uniform and concentrated loads
  - study the convergence of the first natural frequency for different values of R and S
  - analyse the effect of the boundary conditions over the first natural frequency

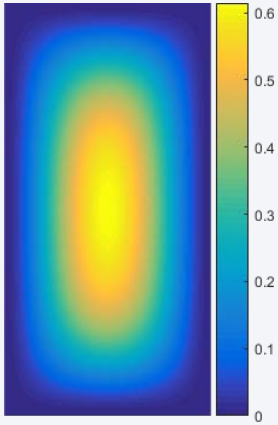
Example: bending response

- Displacement in the middle of the plate using:
  - 5×5 functions
  - trigonometric functions for SSSS boundary conditions
  - polynomial functions for CCCC boundary conditions

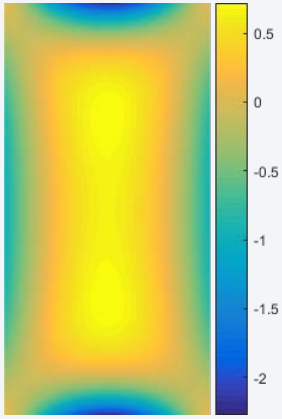
w_{mid}		a/b		
Bcs	Load	0.5	1.0	2.0
CCCC	uniform	0.0384	0.3070	0.6148
CCCC	concent	0.0989	0.3237	0.3954
SSSS	uniform	0.1538	0.9861	2.4603
SSSS	concent	0.2434	0.6930	0.9735

Example: bending response

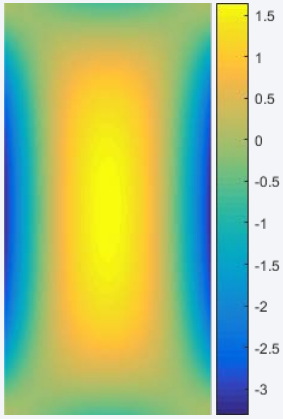
- CCCC plate, 5×5 functions, a = 400 mm
  - uniformly distributed load



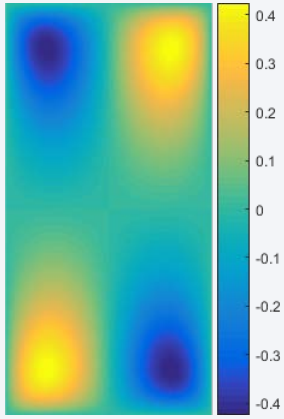
w



M_{xx}

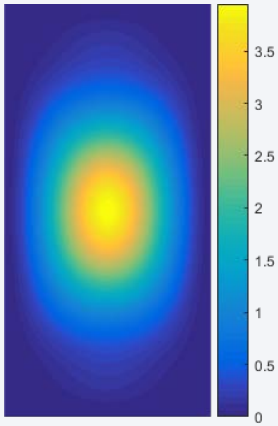


M_{yy}

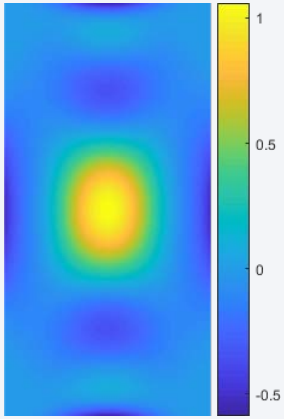


M_{xy}

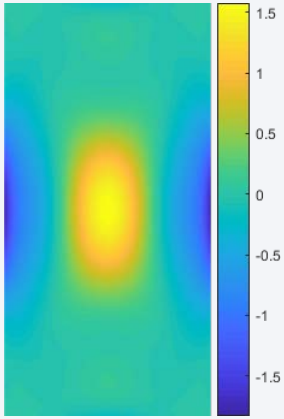
  - concentrated load



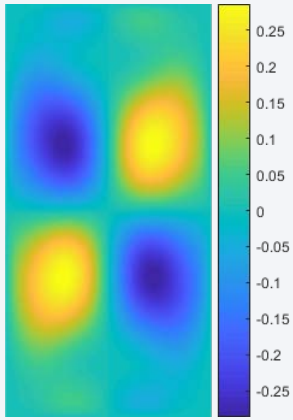
w



M_{xx}



M_{yy}



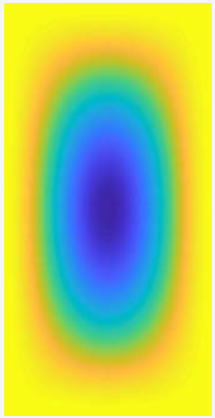
M_{xy}

Example: free vibrations

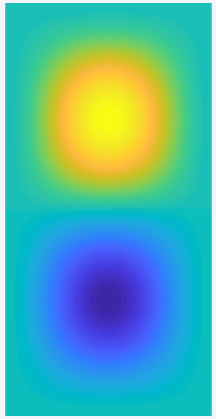
- Free vibrations of CCCC plate (using 5×5 functions)

a/b	0.5	1.0	2.0
Mode		$\overline{\omega}_i$	
1	24.5782	35.9855	98.3127
2	31.8329	73.4121	127.3315
3	44.8082	73.4121	179.2330
4	64.0039	108.2574	256.0154

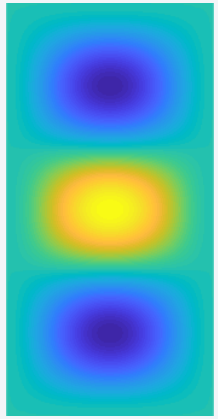
$$\overline{\omega}_i = \omega_i a^2 \sqrt{\frac{I_0}{D}}$$



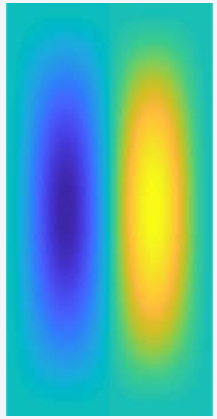
Mode 1



Mode 2



Mode 3



Mode 4