Problem set 3 - solutions

Problem 1

Assuming that the approximate longitudinal displacement $u^h(x,t)$ can be expressed as

$$u^h(x,t) = h_1(x)q_1(t) + h_2(x)q_2(t) + h_3(x)q_3(t),$$

where $h_i(x)$ represent the quadratic shape functions and $q_i(t)$ denote the unknown nodal displacements, the semi-discrete weak formulation is expressed as follows:

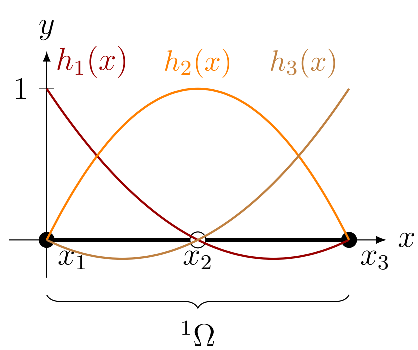
$$\mathbf{M}\ddot{\mathbf{q}}(t) + \mathbf{K}\mathbf{q}(t) = 0, \quad t \in]0, T[\tag{1}$$

where $\mathbf{M}$ and $\mathbf{K}$ are the global mass and stiffness matrices and $\mathbf{q}(t) = \{q_1(t), q_2(t), q_3(t)\}^T$ is the unknown nodal displacement vector. The components of $\mathbf{M}$ and $\mathbf{K}$ can be specified as

$$m_{ij} = \int_0^\ell \rho A \; h_i \; h_j \, dx,$$
$$k_{ij} = \int_0^\ell EA \, \left(\frac{\partial h_i}{\partial x}\right)\left(\frac{\partial h_j}{\partial x}\right) dx,$$

Note that the system of equations (3) will still need to be modified to include the homogenoeus boundary condition $q_1 = 0$.
Since the finite element $\Omega =]0, \ell[$ is quadratic, the three global parabolic shape functions, which are linearly independent and satisfy the convergence criteria, are expressed as

$$h_1(x) = l_1[x_1, x_2, x_3](x) = \prod_{\substack{m=1\\m\neq 1}}^{3} \frac{x - x_m}{x_1 - x_m} = 1 - \frac{3}{\ell}x + \frac{2}{\ell^2}x^2$$
$$h_2(x) = l_2[x_1, x_2, x_3](x) = \prod_{\substack{m=1\\m\neq 2}}^{3} \frac{x - x_m}{x_2 - x_m} = \frac{4}{\ell}x - \frac{4}{\ell^2}x^2$$
$$h_3(x) = l_3[x_1, x_2, x_3](x) = \prod_{\substack{m=1\\m\neq 3}}^{3} \frac{x - x_m}{x_3 - x_m} = -\frac{1}{\ell}x + \frac{2}{\ell^2}x^2$$



Inserting these expressions into the coefficients m_{ij} and k_{ij} of the mass and stiffness matrices leads, after integration, to the system of equations:

$$\frac{\rho A\ell}{30}\begin{bmatrix}4 & 2 & -1\\ 2 & 16 & 2\\ -1 & 2 & 4\end{bmatrix}\ddot{\mathbf{q}}(t)+\frac{EA}{3\ell}\begin{bmatrix}7 & -8 & 1\\ -8 & 16 & -8\\ 1 & -8 & 7\end{bmatrix}\mathbf{q}(t)=0 \tag{2}$$

in which the homogeneous boundary condition $q_1 = 0$ and its virtual equivalence must now be incorporated.
Due to the nature of this boundary condition (clamping), the virtual displacement δq_1 is zero since the virtual displacement field $\boldsymbol{\delta}\mathbf{q}$ is kinematically admissible. It follows that the first equation of (2) must be eliminated, so that the problem reduces to solving for the displacements $q_2(t)$ and $q_3(t)$, $t \in [0, T]$.

$$\frac{\rho A\ell}{30}\begin{bmatrix}16 & 2\\ 2 & 4\end{bmatrix}\begin{Bmatrix}\ddot{q}_2\\ \ddot{q}_3\end{Bmatrix}+\frac{EA}{3\ell}\begin{bmatrix}16 & -8\\ -8 & 7\end{bmatrix}\begin{Bmatrix}q_2\\ q_3\end{Bmatrix}=\begin{Bmatrix}0\\ 0\end{Bmatrix}$$

By assuming an harmonic solution of pulsation ω, the above expression becomes:

$$\left(\frac{\omega^2\rho A\ell}{30}\begin{bmatrix}16 & 2\\ 2 & 4\end{bmatrix}-\frac{EA}{3\ell}\begin{bmatrix}16 & -8\\ -8 & 7\end{bmatrix}\right)\begin{Bmatrix}p_2\\ p_3\end{Bmatrix}=\begin{Bmatrix}0\\ 0\end{Bmatrix}$$

where the coefficients p_i $(i = 2, 3)$ are the amplitudes of the harmonic function at nodal points 2 and 3. The characteristic equation [1] associated with

$$\omega^4-\omega^2\left(\frac{104E}{3\rho\ell^2}\right)+\frac{80E^2}{\rho^2\ell^4}=0$$

provides the following two solutions, which correspond to the approximate natural frequencies of the bar,

$$\omega_1 = 1.577\sqrt{\frac{E}{\rho\ell^2}} \qquad \text{and} \qquad \omega_2 = 5.673\sqrt{\frac{E}{\rho\ell^2}}$$

Let us note the excellent accuracy of the fundamental frequency, which deviates by only 0.4% from the exact value. However, the relative error in the second calculated frequency is significantly higher, reaching 20% compared to the exact result. This significant difference arises because the bar is discretized using a single finite element. The basis functions, which have a parabolic shape, are poorly suited for accurately representing the three-quarters sine wave corresponding to the second mode, while they are sufficiently capable of approximating the quarter sine wave reflecting the first mode.
Nevertheless, it can be shown that higher-order natural frequencies quickly converge to the exact values as the mesh refinement increases.

Problem 2

Discretizing the weak form using the finite element procedure leads to the following system of equations:

$$\mathbf{M}\ddot{\mathbf{q}}(t)+\mathbf{K}\mathbf{q}(t)=\mathbf{r}(t) \tag{3}$$

[1] Refer to the MATLAB code for a detailed calculation.

where the mass and stiffness matrices have compenents given by

$$m_{ij} = \rho \int_0^2 \int_0^2 h_i(x,y) h_j(x,y)\, dx\, dy \quad (i,j = 1,\ldots,9)$$
$$k_{ij} = S \int_0^2 \int_0^2 \partial_x h_i(x,y) \partial_x h_j(x,y) + \partial_y h_i(x,y) \partial_y h_j(x,y)\, dx\, dy \quad (i,j = 1,\ldots,9)$$

Moreover the components of the vector of applied forces are

$$r_i(t) = \int_0^2 \int_0^2 h_i(x,y) p(x,y,t)\, dx\, dy \quad (i,j = 1,\ldots,9).$$

Notice that, due to the vanishing boundary conditions on Γ, the vector of unknown nodal displacements $\mathbf{q} = \{q_1,\ldots,q_9\}^T$ has only one non zero component q_5, representing the transveral displacement of the center of the membrane. Therefore the system (3) reduces to one single equation:

$$m_{55}\ddot{q}_5(t) + k_{55} q_5(t) = r_5(t)$$

coupled with the initial conditions $q_5(0) = \dot{q}_5(0) = 0$. Finally, the bilinear quadrilateral shape function h_5 is defined as follows:

$$h_5(x,y) = \begin{cases} xy & 0 \le x, y \le 1 \\ (2-x)y & 1 \le x \le 2,\ 0 \le y \le 1 \\ x(2-y) & 0 \le x \le 1, \quad 1 \le y \le 2 \\ (2-x)(2-y) & \pi \le x, y \le 2, \end{cases}$$

The following code, available on the MATLAB drive, is used to calculate and to plot the time response of the geometric center of the membrane $q_5(t)$. We start by definig the physical parameters that describe the membrane's properties and the symolic variables.

```
clc; clear; close all;
%% Problem Parameters
% Membrane length in x and y directions (m)
l = 2;
% Tension per unit length (N/m)
S = 1000;
% Material density (kg/m^2)
rho = 1200;
% Load amplitude (N/m^2)
A = 100;
% Forcing frequency (rad/s)
omega_bar = 10*pi;

% Define symbolic variables
syms x y t q(t)
```

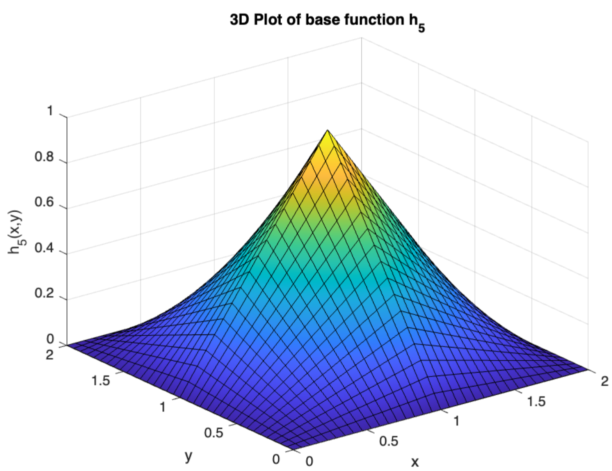
The forcing function (p) represents the time-dependent external force (pressure) applied to the membrane. This force will vary with time and cause the membrane to oscillate.

```
% Define forcing function
p = A*sin(omega_bar*t);
```

The shape function (h_5) is a piecewise function that defines the displacement shape of the membrane as a function of spatial coordinates x and y. The function is broken into four regions of the membrane, each with a different mathematical expression.

```
% Define shape function and plot it
h_5=piecewise((x >= 0) & (x <= 1/2) & (y >= 0) & (y <= 1/2), x*y, ...
                    (x >= 1/2) & (x <= 1) & (y >= 0) & (y <= 1/2), (2-x)*y, ...
                    (x >= 0) & (x <= 1/2) & (y >=1/2) & (y <= 1), x*(2-y),...
                    (x >= 1/2) & (x <= 1) & (y >=1/2) & (y <= 1),(2-x)*(2-y));

figure;
fsurf(h_5, [0, 2, 0, 2])
xlabel('x')
ylabel('y')
zlabel('h_5(x,y)')
title('3D Plot of base function h_5')
grid on
```




```
% Compute the components of the stiffness and mass matrices and forces vector

grad_h_5 = gradient(h_5, [x, y]);

k_55 = vpa(S * int(int(transpose(grad_h_5)*grad_h_5,x,[0,1]),y,[0,1]),4)
m_55 = vpa(rho * int(int(h_5^2,x,[0,1]),y,[0,1]),4)
r_5(t) = int(int(p*h_5,x,[0,1]),y,[0,1])
```

Equation of Motion (ODE): this is the core of the problem: the second-order ordinary differential equation (ODE) that describes the motion of the membrane over time. The displacement and the velocity at time $t = 0$ is assumed to be zero. This represents the initial rest state of the membrane. Solving the ODE: dsolve: This command solves the ODE symbolically to find the displacement of the membrane as a function of time.

```
% Define the ODE resulting form the discretization and solve it to find the
% time response at the node x_5
ode = m_55 * diff(q, t, 2) + k_55 * q == r_5(t);
initial_cond = [q(0) == 0, subs(diff(q, t), t, 0) == 0];
q_5 = dsolve(ode, initial_cond);

fplot(q_5,[0,10]);
xlabel('Time (seconds)')
ylabel('q_5(t) (meters)')
grid on
```