Exercise 8

A string of length $l = 1$ m is connected at both ends to a wall and is subjected to an external force per unit length of $F(x) = 1(x)\,\frac{\text{MN}}{\text{m}}$ at $0.29 \leq x \leq 0.31$ m and $F(x) = 0\,\frac{\text{MN}}{\text{m}}$ elsewhere. Find the displacement $u(x)$ of the string assumed that the load $F(x)$ is acting down (negative direction) and the displacement $u(x)$ is governed by the differential equation

$$T\frac{d^2u}{dx^2} = -F(x)$$

where $T = 70$ MPa (copper) is the uniform tensile force of the string.

a) What are the boundary conditions for this problem?

b) Derive a finite element scheme for this problem using Galerkin's method.

c) Which simple shape function would be appropriate?

d) Write a 1D computer code to evaluate the displacement $u(x)$.

e) Solve the problem by Comsol in 1D and compare the results.

Solution

a) Since the end points of the string is fixed, the boundary conditions (BCs) are:

$$u(0) = 0 \textbf{ and } u(1) = 0 \tag{1}$$

which are clearly Drichlet type BCs.

b) In order to obtain a numerical scheme using Finite Elements for this problem, we need to first derive the variational form of the problem. To do that, we need to first test the PDE with sufficiently smooth functions (test functions) and integrate over the whole numerical domain, i.e.[1]:

$$T\int_0^1 \frac{d^2u}{dx^2}v\,dx = -\int_0^1 Fv\,dx \tag{2}$$

$$T\int_0^1 \frac{du}{dx}\frac{dv}{dx}\,dx = \int_0^1 Fv\,dx \tag{3}$$

where $v(x)$ are the test functions.

c) Since we are using the Galerkin method, the basis functions and the test functions should be the same (For this example we will use hat functions to make the implementation as easy as possible). Therefore we use hat functions as the basis functions as well. The shape of the hat function can be seen below:

$$b_N^j(x_i) = \delta_{ij} \tag{4}$$

[1] integration by parts is used below i.e. $\int_a^b f(x)g(x)dx = [f(x)g'(x)]_a^b - \int_a^b f'(x)g(x)dx$

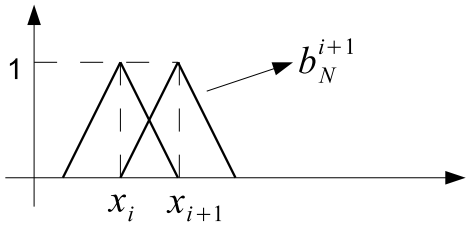

Figure 1: Hat functions used as the basis functions for the problem

By using these basis functions, the solution will be obtained as a superposition of the basis functions:

$$u_{num}(x) = \sum_{i=0}^{N} \mu_i b_N^i(x) \tag{5}$$

where N is the number of nodes used in the discretization and μ_i are the apmlitudes to be determined by the solution of the Finite Element approach.

d)In Eq.3, the integral on the LHS will provide us the mass matrix $\mathbf{A}$ and the integral on the RHS will provide us the load vector $\overrightarrow{\sigma}$. Therefore, we obtain the following matrix system:

$$\mathbf{A}\overrightarrow{\mu} = \overrightarrow{\sigma} \tag{6}$$

By combining Eq.3 with Eq.5, we obtain the entries of the mass matrix $\mathbf{A}$ as follows:

$$a_{ij} = \int_0^1 \frac{db_N^i(x)}{dx}\frac{db_N^j(x)}{dx}dx \tag{7}$$

and the entries of the load vector is given by:

$$\sigma_i = \int_0^1 F(x) b_N^i(x) dx \tag{8}$$

a Matlab code that is responsible for the organization of the matrix entries and the load vector can be seen below (all the code must be saved in a file $onedfem.m$):

```
function [u,sigma]=onedfem(N)
% 1D finite element analysis of string problem. the pde to be solved is:
% d(u)/dx=-F with the zero Drichlet BCs.
% Aytac Alparslan 18/10/2010 Zurich
% N number of nodes
l=1; %length of the computational domain
h = l/(N-1); %mesh size
xx=linspace(0,1,N); %the locations of nodes in x
T=70;
bc(1)=0; % boundary condition at x=0
bc(2)=0; % boundary condition at x=l
A=zeros(N-2,N-2); %initialize mass matrix
```

```
sigma=zeros(N-2,1); %initialize load vector
u=zeros(1,N); %initialize the solution
%note that since the boundary conditions are given, we do not care about
%them, we solve only the part between the boundary points, i.e u(2:N-1)
A(1,1)=2/h;
A(1,2)=-1/h;
sigma(1)=quad(@(x)rhs_int(xx(2),x,h),xx(2)-h,xx(2)+h);
A(N-2,N-2)=2/h;
A(N-2,N-3)=-1/h;
sigma(N-2)=quad(@(x)rhs_int(xx(N-1),x,h),xx(N-1)-h,xx(N-1)+h);
for ii=2:N-3
    %the mass matrix
    A(ii,ii-1)=-1/h;
    A(ii,ii)=2/h;
    A(ii,ii+1)=-1/h;
    %the load vector
    sigma(ii) = quad(@(x)rhs_int(xx(ii+1),x,h),xx(ii+1)-h,xx(ii+1)+h);
end
A=A.*T;
u_num=A\sigma;
u(2:N-1)=u_num;
u(1)=bc(1);
u(N)=bc(2);

%%%%%%
function res=rhs_int(x0,x,h)
% the integrand for the rhs vector
shap=shap_1D(x,x0,h);
ffx=ff(x);
res=shap.*ffx;
return;

%%%%%%
function shap = shap_1D(x,x0,h)
% the shape function of the node with the coordinate x0
% the value of the shape function is 1 at x0 and decreases linearly and
% becomes zero at x0-h and x0+h
shap=zeros(size(x));
for ii=1:length(x)
    if x(ii)>=(x0+h) || x(ii)<=(x0-h)
        shap(ii)=0.0;
    elseif x(ii)>=x0
        shap(ii)=-(x(ii)/h)+(x0/h)+1;
    elseif x(ii)<=x0
        shap(ii)=(x(ii)/h)+1-(x0/h);
    end
end
```

```
return;

%%%%%%
function F_res=ff(x)
%the force function
F_res=zeros(size(x));
for ii=1:length(x)
    if x(ii)>=0.29 && x(ii)<=0.31
        F_res(ii)=-1;
    else
        F_res(ii)=0;
    end
end
return;
```

e) The figure below is a comparison of the results obtained in Matlab code and Comsol.

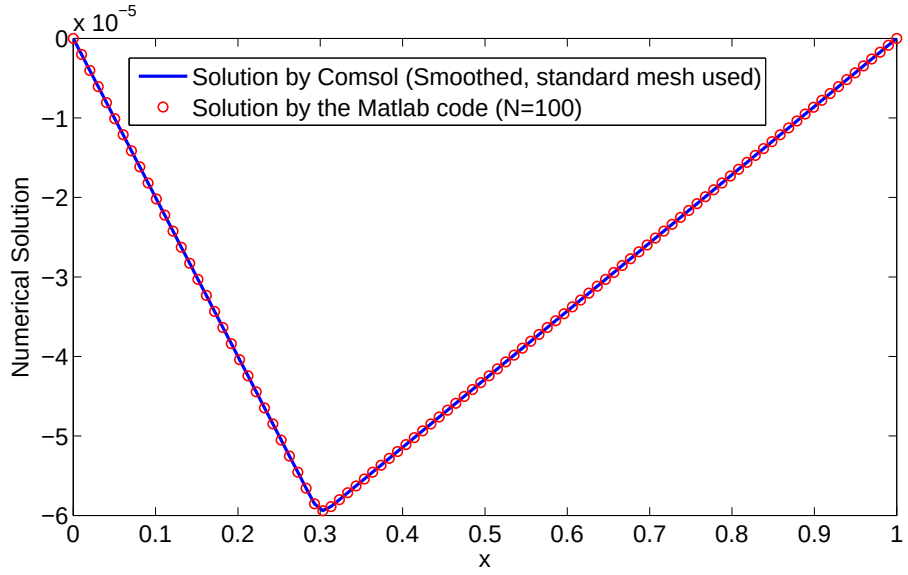

Figure 2: The numerical solutions obtained from the Matlab code and Comsol.