School of Engineering

Brown University

FEA Project

Finite Element Analysis of a Timoshenko Beam

Instructor: **Prof. Bower**

Alireza Khorshidi

12/11/13

Introduction [1]: The theory of Timoshenko beam was developed early in the twentieth century by the Ukrainian-born scientist Stephan Timoshenko. Unlike the Euler-Bernoulli beam, the Timoshenko beam model accounts for shear deformation and rotational inertia effects. Therefore, the Timoshenko beam can model thick (short) beams and sandwich composite beams. The stiffness of the Timoshenko beam is lower than the Euler-Bernoulli beam, which results in larger deflections under static loading and buckling. The limiting case of infinite shear modulus will neglect the rotational inertia effects, and therefore will converge to the ordinary Euler-Bernoulli beam.

Dimension Analysis: The variables in the problem are $w, \theta, q, E, I, L, G, A$ and there are two independent variables. Therefore, I expect that the solution finally be a function in the following form:

$$w/L = f_1\left(\frac{I}{L^4}, \frac{A}{L^2}, \frac{E}{G}, \frac{q}{EL}\right), \tag{1}$$

$$\theta = f_2\left(\frac{I}{L^4}, \frac{A}{L^2}, \frac{E}{G}, \frac{q}{EL}\right). \tag{2}$$

Theoretical Formulation [2]: In the Timoshenko theory, the displacement field is assumed to be

$$u_x(x,y) = -y\,\theta(x), \qquad u_y(x,y) = w(x). \tag{3}$$

Therefore, the strain components are

$$\varepsilon_{xx} = \frac{\partial u_x}{\partial x} = -y\,\theta^{'}(x), \qquad \varepsilon_{yy} = 0, \qquad \gamma_{xy} = -\theta(x) + w^{'}(x), \tag{4}$$

and the stress components are (assuming plane stress):

$$\sigma_{xx} = -Ey\theta^{'}(x), \qquad \sigma_{yy} = -E\,\frac{\nu}{1-\nu^2}\,y\,\theta^{'}(x), \quad \sigma_{xy} = \kappa G\,\gamma_{xy} = \kappa G\left(w^{'}(x) - \theta(x)\right). \tag{5}$$

The potential energy of the beam (π) is

$$\pi = \frac{1}{2}\int \sigma_{xx}\varepsilon_{xx}dV + \frac{1}{2}\int \sigma_{xy}\varepsilon_{xy}dV - \int q(x)w(x)dx \tag{6}$$

Substituting the expressions of strains and stresses into Eq. (6) yields

$\pi = \frac{1}{2}\int E{\varepsilon_{xx}}^2 dV + \frac{1}{2}\int \kappa G{\gamma_{xy}}^2 dV - \int q(x)w(x)dx =$

$\frac{1}{2}\int\int Ey^2(\theta'(x))^2 dAdx + \frac{1}{2}\int\int \kappa G(-\theta(x) + w'(x))^2 dAdx - \int q(x)w(x)dx.$

$$\Rightarrow \pi = \frac{1}{2}EI\int_0^L (\theta'(x))^2 dx + \frac{1}{2}\kappa GA\int_0^L (-\theta(x) + w'(x))^2 dx - \int_0^L q(x)w(x)dx \tag{7}$$

.

And therefore, the variation in the potential energy is

$$\delta\pi = EI\int_0^L \frac{d\theta(x)}{dx}\frac{d\delta\theta}{dx}dx + \kappa GA\int_0^L \left(\theta(x) - \frac{dw(x)}{dx}\right)\left(\delta\theta - \frac{d\delta w}{dx}\right)dx - \int_0^L q(x)\delta w dx = 0 \tag{8}$$

Next the FEA interpolation for $w(x)$, $\theta(x)$, δw, and $\delta\theta$ should be introduced [3].

$$w(x) = \sum_a N^a\, w^a, \qquad \theta(x) = \sum_a N^a\, \theta^a,$$
$$\delta w = \sum_b N^b\, \delta w^b, \quad \delta\theta = \sum_b N^b\, \delta\theta^b \tag{9}$$

Plugging the above interpolations into Eq. (8) yields:

$$\delta\pi = EI\theta^a\delta\theta^b \int_0^L \frac{dN^a}{dx}\frac{dN^b}{dx}dx + \kappa GA \int_0^L \left(N^a\theta^a - \frac{dN^a}{dx}w^a\right)\left(N^b\delta\theta^b - \frac{dN^b}{dx}\delta w^b\right)dx$$
$$- \delta w^b \int_0^L q(x)N^b dx = 0 \tag{10}$$

which should hold for any δw^b and $\delta\theta^b$, and therefore

$$\left\{\kappa GA \int_0^L \frac{dN^a}{dx}\frac{dN^b}{dx}dx\right\}w^a + \left\{-\kappa GA \int_0^L N^a \frac{dN^b}{dx}dx\right\}\theta^a = \int_0^L q(x)N^b dx. \tag{11a}$$

$$\left\{-\kappa GA \int_0^L \frac{dN^a}{dx}N^b dx\right\}w^a + \left\{\kappa GA \int_0^L N^a N^b dx + \int_0^L EI\frac{dN^a}{dx}\,\frac{dN^b}{dx}dx\right\}\theta^a = 0. \tag{11b}$$

The above integrals should first be calculated using Gaussian quadrature scheme, and then assembling over all the elements yields

$$\boldsymbol{Ku} = \boldsymbol{F}, \tag{12}$$

where $\boldsymbol{u} = \{w^1 \quad w^2 \quad w^3 \quad \ldots \quad w^N \quad \theta^1 \quad \theta^2 \quad \ldots \quad \theta^N\}^{\mathrm{T}}$.

Finally $\boldsymbol{u}$ is computed as $\boldsymbol{u} = \boldsymbol{K^{-1}F}$ [3].

Eqs. (11) has been implemented in a Matlab code. The code is self-explanatory, and is given in the Appendix. The next section is devoted to the results.

Results:

The Timoshenko beam subjected to uniform load distribution with different boundary conditions has been already solved analytically. The table below summarized the analytical results [4]; in this table ν is the displacement, and the subscripts E and T correspond to Euler-Bernouli beam and Timoshenko beam, respectively.

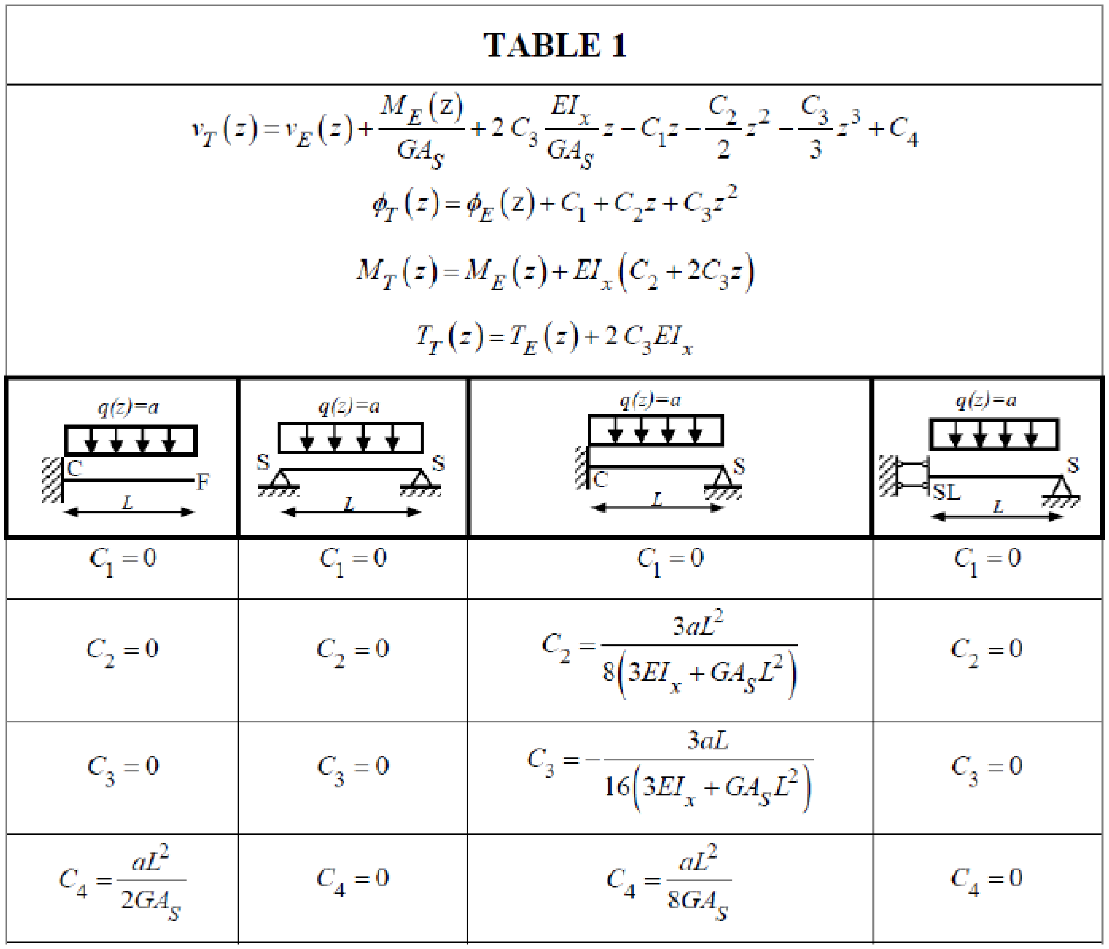


TABLE 1

$$v_T(z)=v_E(z)+\frac{M_E(z)}{GA_S}+2C_3\frac{EI_x}{GA_S}z-C_1z-\frac{C_2}{2}z^2-\frac{C_3}{3}z^3+C_4$$

$$\phi_T(z)=\phi_E(z)+C_1+C_2z+C_3z^2$$

$$M_T(z)=M_E(z)+EI_x(C_2+2C_3z)$$

$$T_T(z)=T_E(z)+2C_3EI_x$$

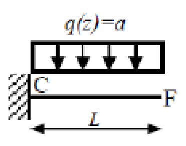
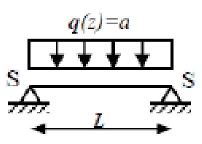
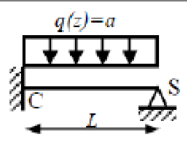
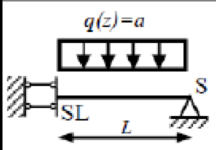
			
$C_1=0$	$C_1=0$	$C_1=0$	$C_1=0$
$C_2=0$	$C_2=0$	$C_2=\frac{3aL^2}{8(3EI_x+GA_SL^2)}$	$C_2=0$
$C_3=0$	$C_3=0$	$C_3=-\frac{3aL}{16(3EI_x+GA_SL^2)}$	$C_3=0$
$C_4=\frac{aL^2}{2GA_S}$	$C_4=0$	$C_4=\frac{aL^2}{8GA_S}$	$C_4=0$

Table 1: Beam under a uniformly distributed load

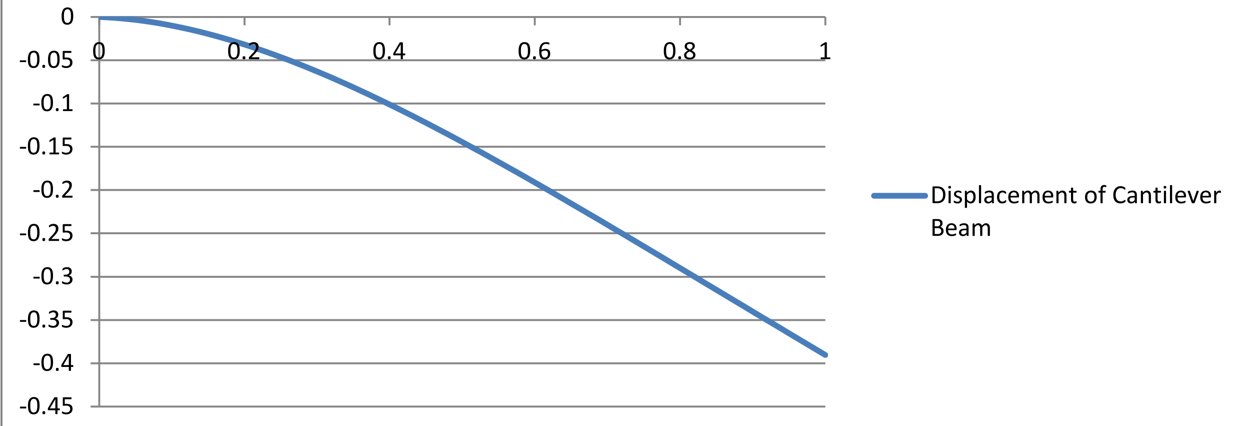
For $q=-1000, L=10, b=1, h=2, E=5*10^6, \nu=0.3, \kappa=5/6$, the simply-supported Timoshenko beam is solved with FEA and analytically. The below table compare the results:

Table 2: Displacements of the simply supported Timoshenko beam

Node No	Position	Exact Results	FEA Results
1	0	0.0000	0
2	1	-0.0137	-0.0134
3	2	-0.0257	-0.0253
4	3	-0.0350	-0.0345
5	4	-0.0409	-0.0403
6	5	-0.0430	-0.0423
7	6	-0.0409	-0.0403
8	7	-0.0350	-0.0345
9	8	-0.0257	-0.0253
10	9	-0.0137	-0.0134
11	10	0.0000	0

The next example is the cantilever beam with the same loading, material properties, and geometries as the previous example. One hundred elements have been used for this example. The deformed shape of the beam is shown below:

Fig 1: Displacements of the Cantilever Beam



The maximum displacement at the end of the beam is 0.3906. For the limiting case of $\kappa \to \infty$, the special case of Euler-Bernoulli beam is obtained. In this example, the maximum displacement of the Euler-Bernoulli beam is calculated as 0.3750. And so, the shear deformation is around 4% of the total displacement in the Timoshenko beam.

In the FEA code, first two Gaussian points were used for all integrations. However, the convergence was relatively slow, and therefore, I finally ended up with two integration points for the bending part of the stiffness matrix and one integration point for the shear part of the stiffness matrix.

References:

[1] http://en.wikipedia.org/wiki/Timoshenko_beam_theory

[2] Theory of Elasticity, Stephen Timoshenko

[3] Class lecture notes

[4] S. Cutrona, S. Di Lorenzo, A. Pirrotta, Timoshenko vs Euler-Bernoulli beam: fractional visco-elastic behavior.

Appendix: The Matlab code used for the above calculations is given below:

```
function Bending_of_Timoshenko_Beam_V3
%..............................................................

% MATLAB Code for FE Analysis of Timoshenko Beam under Bending

clc;
clear all

% Material properties:  E (modulus of elasticity), nu (Possion's ratio), G
% (shear modulus), kappa
E = 5000000; nu = 0.3; G = E/(2*(1+nu)); kappa = 1000000;

% Geometric properties:  L (length of beam), h (thickness of beam)
% I: second moments of area
L = 10; h = 2;
I = h^3/12;
EI = E*I;

%Distributed load
q = -1000;

% Mesh
nelm = 100;
ncoords=linspace(0,L,nelm+1); % node coordinates

% Connectivity is the list of nodes for each element
for i=1:size(ncoords,2)-1
   connect(i,1)=i;
   connect(i,2)=i+1;
end
nnodes = nelm + 1; % nnodes is the total number of nodes
TotalDofs=2*nnodes; % TotalDofs: global number of degrees of freedom

% Computation of the system stiffness and residual matrices
[stiffness,resid]=...
   GlobalStiffnessAndResidual(TotalDofs,nelm,connect,nnodes,q,EI,kappa,h,G,L);

% boundary conditions (simply-supported at both bords)
%fixedNodeW =[1 ; nnodes];
%fixedNodeTX=[];
% boundary conditions (clamped at both bords)
%fixedNodeW =[1 ; nnodes];
%fixedNodeTX=[1 ; nnodes];
% boundary conditions (cantilever)
fixedNodeW =[1];
fixedNodeTX=[1];
prescribedDof=[fixedNodeW; fixedNodeTX+nnodes];
```

```
% solution
%displacements=solution(TotalDofs,prescribedDof,stiffness,resid);

activeDof=setdiff((1:TotalDofs)', (prescribedDof));

U=stiffness(activeDof,activeDof)\resid(activeDof);
displacements=zeros(TotalDofs,1);
displacements(activeDof)=U;

% displacements
disp('Displacements and Rotations:')
%displacements=displacements1;
jj=1:TotalDofs; format
[jj' displacements]

end
%
%================= STIFFNESS MATRIX AND RESIDUAL VECTOR
================================
%
% This function computes the stiffness matrix and residual vector for
% Timoshenko beam
%
function [stiffness,resid]=...
   GlobalStiffnessAndResidual(TotalDofs,nelm,connect,nnodes,q,EI,kappa,h,G,L)

stiffness=zeros(TotalDofs);
resid=zeros(TotalDofs,1);

% Computing bending contribution for the stiffness matrix

% Integration points and wieghts for bending
integrationpoints=[-0.577350269189626,0.577350269189626];
w=ones(1,2);
%
%  Loop over elements
%
for e=1:nelm
   indice=connect(e,:);
   elmdof=[ indice indice+nnodes];
   ndof=length(indice);
   dett=L/(2*nelm);dxidx=1/dett;
   %
   %  Loop over integration points
   %
   for intpt=1:size(w,2);
      %
      %  Compute shape functions && derivatives wrt local coords
      %
      xi=integrationpoints(intpt);
```

```
        N = shapefunctions(xi);
        dNdxi = shapefunctionderivs(xi);
        %
        %  Convert shape function derivatives:derivatives wrt global coords
        %
        dNdx=dNdxi*dxidx;
        %
        % Compute B matrix
        %
        B=zeros(2,2*ndof);
        B(1,ndof+1:2*ndof)  = dNdx(:);
        %
        % Compute K matrix and residual vector
        %
        stiffness(elmdof,elmdof) = stiffness(elmdof,elmdof) + transpose(B)*B*w(intpt)*dett*EI;
        resid(indice) = resid(indice) + N*q*dett*w(intpt);
    end
end
%
%  Shear contribution for stiffness matrix
%
%  Integration points and wieghts for shear
integrationpoints=[0];
w=[2];
%
%  Loop over elements
%
for e=1:nelm
    indice=connect(e,:);
    elmdof=[ indice indice+nnodes];
    ndof=length(indice);
    %
    %  Loop over the integration points
    %
    for intpt=1:size(w,2) ;
        %
        %     Compute shape functions && derivatives wrt local coords
        %
        xi=integrationpoints(intpt);
        N = shapefunctions(xi);
        dNdxi = shapefunctionderivs(xi);
        %
        %     Convert shape function derivatives:derivatives wrt global coords
        %
        dNdx=dNdxi*dxidx;
        %
        % Compute B matrix
        %
        B=zeros(2,2*ndof);
        B(2,1:ndof) = dNdx(:);
        B(2,ndof+1:2*ndof)  =- N;
        %
        % Compute K matrix
        %
        stiffness(elmdof,elmdof) = stiffness(elmdof,elmdof) + kappa*h*G*transpose(B)*B*w(intpt)*dett;
    end

end

end
%
%================= SHAPE FUNCTIONS ==================================
%
%       Calculates shape functions for various element types
%
function N = shapefunctions(xi)

N = zeros(2,1);
%
%  1D elements
%
N(1) = 0.5*(1.-xi(1));
N(2) = 0.5*(1.+xi(1));
end
%
%================= SHAPE FUNCTION DERIVATIVES ======================
%
function dNdxi = shapefunctionderivs(xi)

dNdxi = zeros(1,2);
%
% 1D elements
%
dNdxi(1) = -0.5;
dNdxi(2) = +0.5;
end
%
%
```