CSMD: Assignment 6

Juan Pedro Roldán

March 2018

1 Introduction

In order to adapt the supplied code, some changes were made

- Material parameters were changed to those given in the assignment
- A loop was included inside the main scripts to consider the eight different cases (eight different area/length relationships)
- Timoshenko beam was changed

2 Beam results

Figures 1 to 9 show the maximum displacements, moments and shear force of the different beams.
We can see that if we use the Euler-Bernoulli formulation, the results with respect to forces (moments and shear forces) are practically the same, no matter the relationship between area and length. This is due to the fact that this formulation is valid for slender beams, and then, low values of log10(Area/L). As we do not take into account the effects of the shear forces, cross-section inertia plays a weak role with respect to internal forces in this problem. Euler displacements are relatively similar to the reduced integration scheme, as the full integration methods suffers the so-called shear locking, overestimating the effects of shear forces for thin beams (left half of the plots).
If we now compare the internal forces in both timoshenko formulations, we can see again here the shear-locking effect in the Moment and Shear plots: for thin beams, the Timoshenko full integration scheme shows really high values of Mx and Shear Force.
The Timoshenko reduced integration beam captures the best of both formulations: accounts for the shear effects but has good performance for thin beams

2.1 Euler beam plots

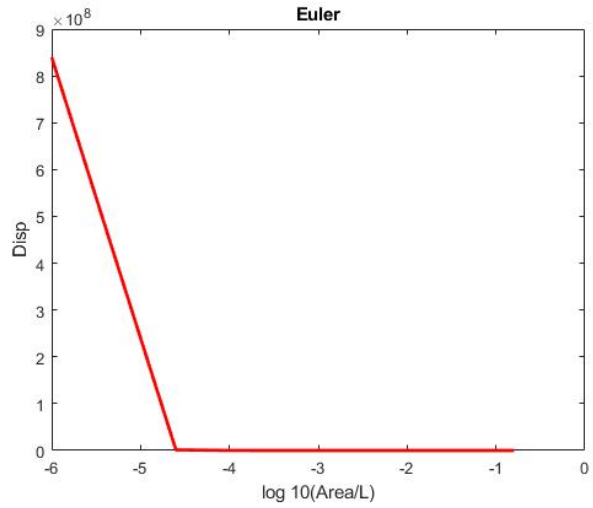

Figure 1: Maximum displacements vs log10 (cross-section/beam length)

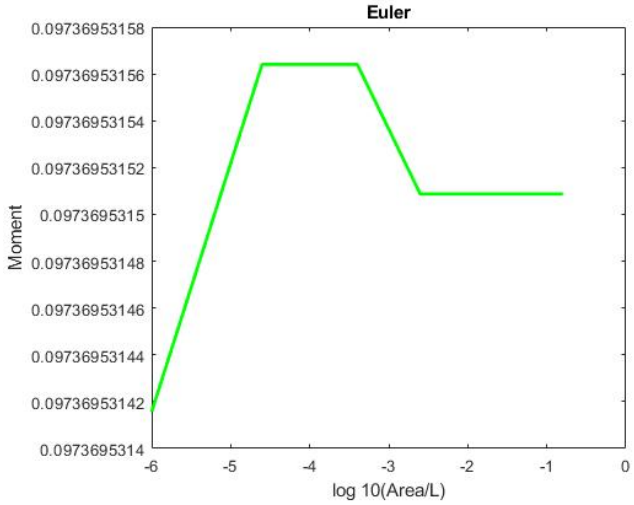

Figure 2: Maximum Mx vs log10 (cross-section/beam length)

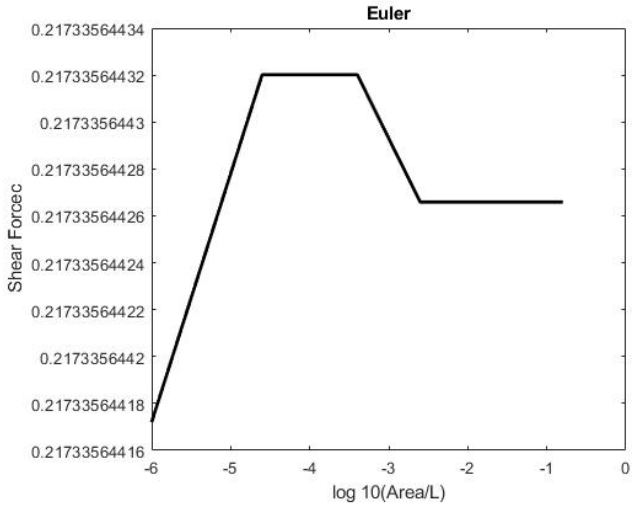

Figure 3: Maximum Shear vs log10 (cross-section/beam length)

2.2 Timoshenko Full integrated beam plots

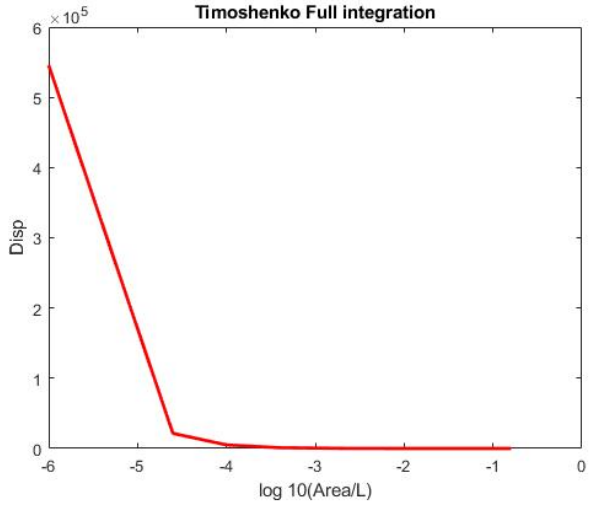

Figure 4: Maximum displacements vs log10 (cross-section/beam length)

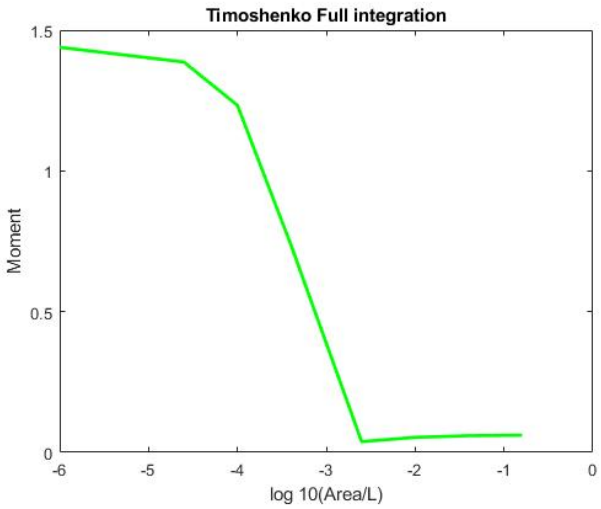

Figure 5: Maximum Mx vs log10 (cross-section/beam length)

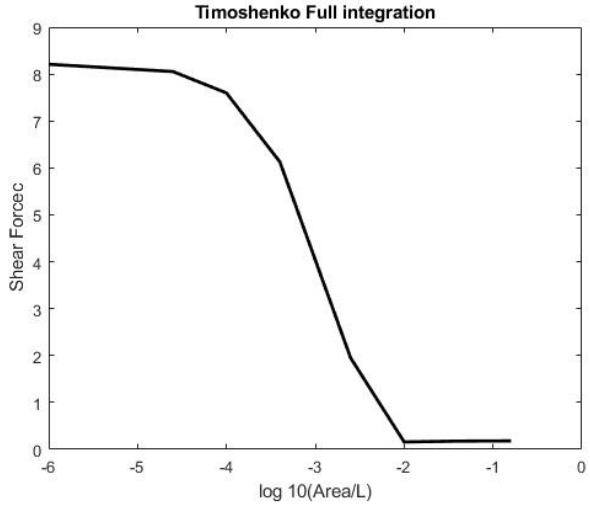

Figure 6: Maximum Shear vs log10 (cross-section/beam length)

2.3 Timoshenko Reduced integrated beam plots

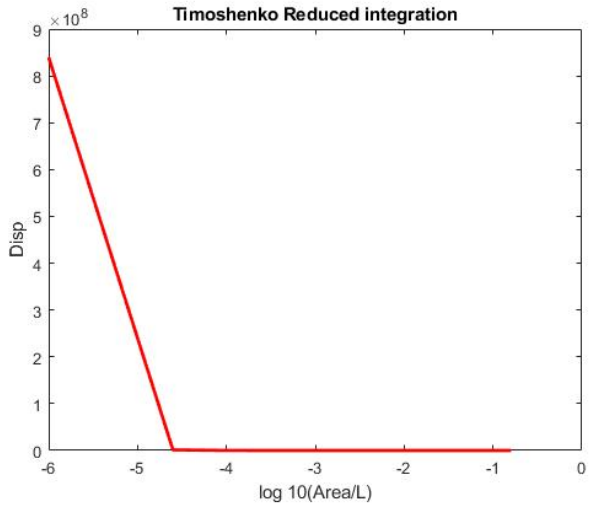

Figure 7: Maximum displacements vs log10 (cross-section/beam length)

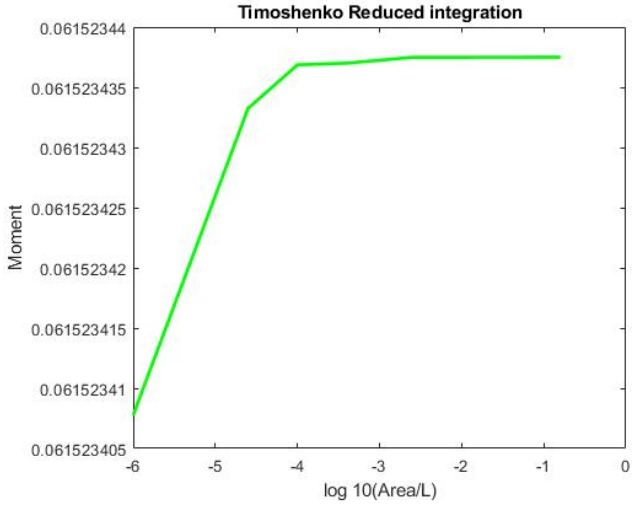

Figure 8: Maximum Mx vs log10 (cross-section/beam length)

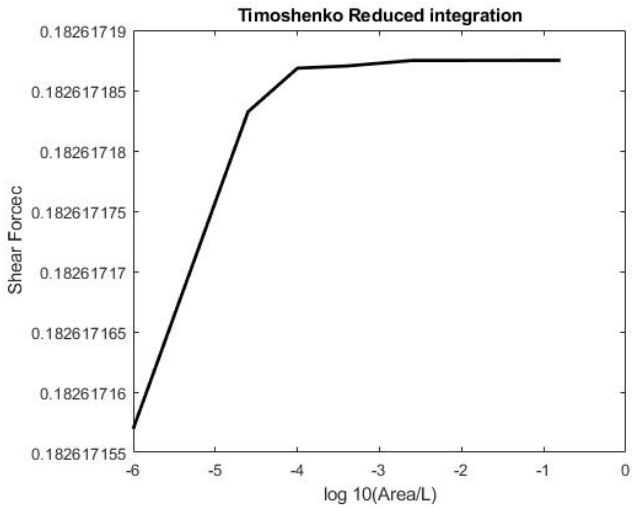

Figure 9: Maximum Shear vs log10 (cross-section/beam length)

7

3 Annex: Modified code

The Timoshenko Beam script was modified so it could include the reduced integration scheme:

1. Figure 10: fullint variable describes the method: 1 for Full Integration, any other value for Reduced Integration
2. Figure 10 : (continuation) we include the different values of areas that we are going to include, and their cross-section inertias
3. Figure 11 The shear matrix for the reduced integration is included
4. Figure 12 The integration points for the reduced integration scheme are also included

```
%               nodal loads.

  file name = input('Enter the file name :','s');
    fullint = 1; % 0 for reduced intregration scheme


  tic;                    % Start clock
  ttim = 0;               % Initialize time counter
  eval (file_name);       % Read input file
  areas = [0.001 0.005 0.010 0.020 0.050 0.1 0.2 0.4];
  areas = areas.^2;
  inertias = (areas.^2)/12;

  Results = zeros(4,8);
  Results(1,:) = areas/4; % First row: relationship areas/length
for o = 1:8
    area = areas(o);
    inercia = inertias(o);

% Finds basics dimentions
  npnod  = size(coordinates,1);        % Number of nodes
  nelem  = size(elements,1);           % Number of elements
  nnode  = size(elements,2);           % Number of nodes per element
  nndof  = npnod*2;                   % Number of total DOF
```

Figure 10: First modification

```
if fullint
    K_shear = [   1   ,   len/2 ,   -1   ,   len/2 ;
            len/2 , len^2/3 , -len/2 , len^2/6 ;
             -1   ,  -len/2 ,    1   ,  -len/2 ;
            len/2 , len^2/6 , -len/2 ,  len^2/3 ];
else
    K_shear = [   1   ,   len/2 ,   -1   ,   len/2 ;
                len/2 , (len^2)/4 , -len/2 , (len^2)/4 ;
                 -1   ,  -len/2 ,    1   ,  -len/2 ;
                len/2 , (len^2)/4 , -len/2 ,  (len^2)/4 ];
end
```

Figure 11: Second modification

```
x_j = coordinates(inods_j); % Elem. coordinates
len = x_j - x_i;

if fullint
gaus1 =-1/sqrt(3);
gaus2 = 1/sqrt(3);
else
    gaus1 = 0;
    gaus2 = 0;
end

bmat_f=[ 0, -1/len, 0, 1/len];

bmat_s1=[-1/len,-(1-gaus1)/2, 1/len,-(1+gaus1)/2];
bmat_s2=[-1/len,-(1-gaus2)/2, 1/len,-(1+gaus2)/2];

Str(ielem,1) = dmatf*(bmat_f *transpose(u_elem));
Str(ielem,2) = dmats*(bmat_s1*transpose(u_elem));
Str(ielem,3) = dmats*(bmat_s2*transpose(u_elem));
```

Figure 12: Third modification