UNIVERSITY OF CALIFORNIA BERKELEY
Department of Civil Engineering
Spring 2010

Structural Engineering,
Mechanics and Materials
Professor: S. Govindjee

Mechanics of Structures (CE130N) Lab 3

1 Objective

The objective of this lab is for you to set up the necessary data structures for defining a truss and to use them to generate the compatibility matrix for a truss and the resulting stiffness matrix too. The results of this lab will also be used for the next lab where we will program the boundary conditions and loads into the file and solve the resultant equilibrium equations.

2 Truss equations synopsis

- Equilibrium

$$\boldsymbol{A}^T\boldsymbol{R} = \boldsymbol{F}\,,$$

- Compatibility/Strain-displacement relation

$$\boldsymbol{\varepsilon} = \left\lceil \frac{1}{L} \right\rfloor \boldsymbol{A}\boldsymbol{u}\,,$$

- Constitutive relation

$$\boldsymbol{\sigma} = \lceil E \rfloor \boldsymbol{\varepsilon}\,,$$

- Resultant definition

$$\boldsymbol{R} = \lceil A \rfloor \boldsymbol{\sigma}\,.$$

The combined equilibrium equation solely in terms of the nodal displacements is

$$\boxed{\boldsymbol{K}\boldsymbol{u} = \boldsymbol{F}\,,}$$

where $\boldsymbol{K} = \boldsymbol{A}^T\lceil AE/L \rfloor\boldsymbol{A}$. Recall further that the dimensions of $\boldsymbol{A}$ are $b \times 2n$, where b is the number of truss bars and n is the number of nodes in the truss. Each row of $\boldsymbol{A}$ has two non-zero (vectorial) entries. For a generic row r in $\boldsymbol{A}$, the non-zeros are located in columns associated with the two nodes associated with bar r. These entries are $\boldsymbol{e}_{n_1n_2}^T$ and $\boldsymbol{e}_{n_2n_1}^T$, where the first goes into the column(s) associated with node n_2 and the second into the column(s) associated with node n_1. The vectors themselves are the unit vectors connecting nodes n_1 and n_2.

3 Download

Download the two lab files from bspace. `plotmesh_truss.m` is a plotting routine which you will use in your program. There is no need to edit it. `truss_student.m` is the file you will need to edit. In this file there are several locations identified with the text `COMPLETE AS APPROPRIATE`. At these points you will need to provide the requisite code.

4 Walk Through

4.1 Location of truss nodes

You will first need to define the locations of the nodes. This will be stored in a $2 \times n$ matrix `truss.node`. The first row of this matrix will contain the x-coordinates of the nodes and the second row the y-coordinates. Each column will thus represent the x, y coordinates of the node and there will be one column per node. Set up this matrix for the two bar example from lecture - one node at the origin [node 1], one at $(1, 1)$ [node 2]and one at $(1, 0)$ [node 3].

4.2 Definition of the bars

You will next need to define which nodes are connected to each other by bars. This information will be stored in a $2 \times b$ matrix `truss.conn`. Each column of `truss.conn` will correspond to a bar and the entry in the first row will be the node number for the first node for the bar and in the second row will be the node number for the second node for the bar. Set up this matrix for our two bar example from lecture where bar one is the diagonal bar and bar two is the vertical bar.

4.3 Checkpoint

At this point comment out the rest of the file below the call to `plotmesh_truss` and run it. You should get a plot of the truss. Go back and add a third bar which is horizontal and re-run to make sure things are working ok. Once satisfied, remove the third bar.

4.4 Initialize $\boldsymbol{A}$

Determine the size that $\boldsymbol{A}$ should be from the information in the structure `truss` and initialize.

4.5 Compute and insert entries into $\boldsymbol{A}$

We will now loop over the bars and construct the compatibility matrix using the information we have set up in `truss.node` and `truss.conn`. In the loop, r refers to the generic bar r.

4.5.1 End to end vector

Using the information in `truss`, compute the end to end vector from the first node of bar r to the second node of bar r. Do not normalize yet. The result will go in `en1n2` as a 2×1 vector.

4.5.2 Compute bar length

Compute the length of the bar and store in `L`.

4.5.3 Compute AE/L

Set an array of AE/L values for the bars. For simplicity, assume $AE = 10$ for every bar. Store in `AEoL`.

4.5.4 Normalize

Normalize the length of `en1n2` and store the result back into `en1n2`.

4.5.5 Compute the end to end vector in the other direction

Compute the normalized end to end vector from the second node to the first node and store in `en2n1`.

4.5.6 Insert into $\boldsymbol{A}$

For the bar r set up row r in $\boldsymbol{A}$ by inserting the transpose of `en1n2` and `en2n1`. This is the tricky part. For example $\boldsymbol{e}_{n_1 n_2}^T$ is a 1×2 matrix. It will need to be inserted into the two columns associated with node n_2. As a concrete hint, suppose the first node of bar 4 was node 1 and the second node was node 3. Then one needs to insert the first component of $\boldsymbol{e}_{31}^T$ into row 4 column 1 and the second component into row 4 column 2. Further one needs to insert the first component of $\boldsymbol{e}_{13}^T$ into row 4 column 5 and the second component into row 4 column 6.

4.6 Compute K

Compute $\boldsymbol{K}$. Hint: To turn the vector of AE/L values into a diagonal matrix use the command `diag`.

5 Verify

5.1 Two bar example

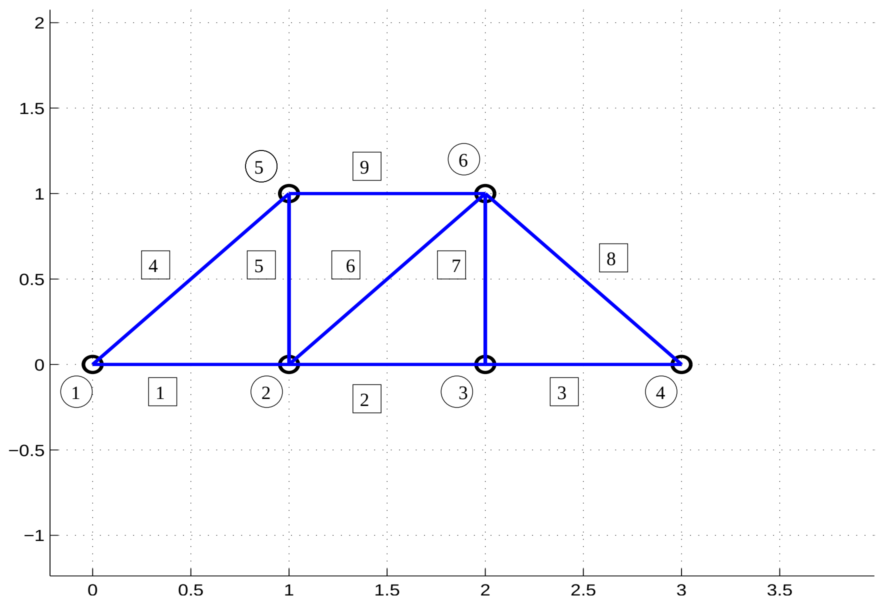
Verify that your code is producing the correct expressions for $\boldsymbol{A}$ and for $\boldsymbol{K}$ for the two bar truss from lecture with $AE = 10$. The answers are (assuming the same numbering as in lecture)

$$\boldsymbol{A} = \begin{bmatrix} -7.0711e-01 & -7.0711e-01 & 7.0711e-01 & 7.0711e-01 & 0 & 0 \\ 0 & 0 & 0 & 1.0000e+00 & 0 & -1.0000e+00 \end{bmatrix}$$

$$\boldsymbol{K} = \begin{bmatrix} 3.5355e+00 & 3.5355e+00 & -3.5355e+00 & -3.5355e+00 & 0 & 0 \\ 3.5355e+00 & 3.5355e+00 & -3.5355e+00 & -3.5355e+00 & 0 & 0 \\ -3.5355e+00 & -3.5355e+00 & 3.5355e+00 & 3.5355e+00 & 0 & 0 \\ -3.5355e+00 & -3.5355e+00 & 3.5355e+00 & 1.3536e+01 & 0 & -1.0000e+01 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1.0000e+01 & 0 & 1.0000e+01 \end{bmatrix}$$

5.2 Test

Use your code to input the following truss; assume $AE = 10$ and compute $\boldsymbol{A}$ and $\boldsymbol{K}$. Make sure you use the given number scheme.



1
2
3
4
5
6
1
2
3
4
5
6
7
8
9