

CEE6501 — Lecture 7.2

Introduction to 2D Frame Analysis

Learning Objectives

By the end of this lecture, you will be able to:

-

Agenda

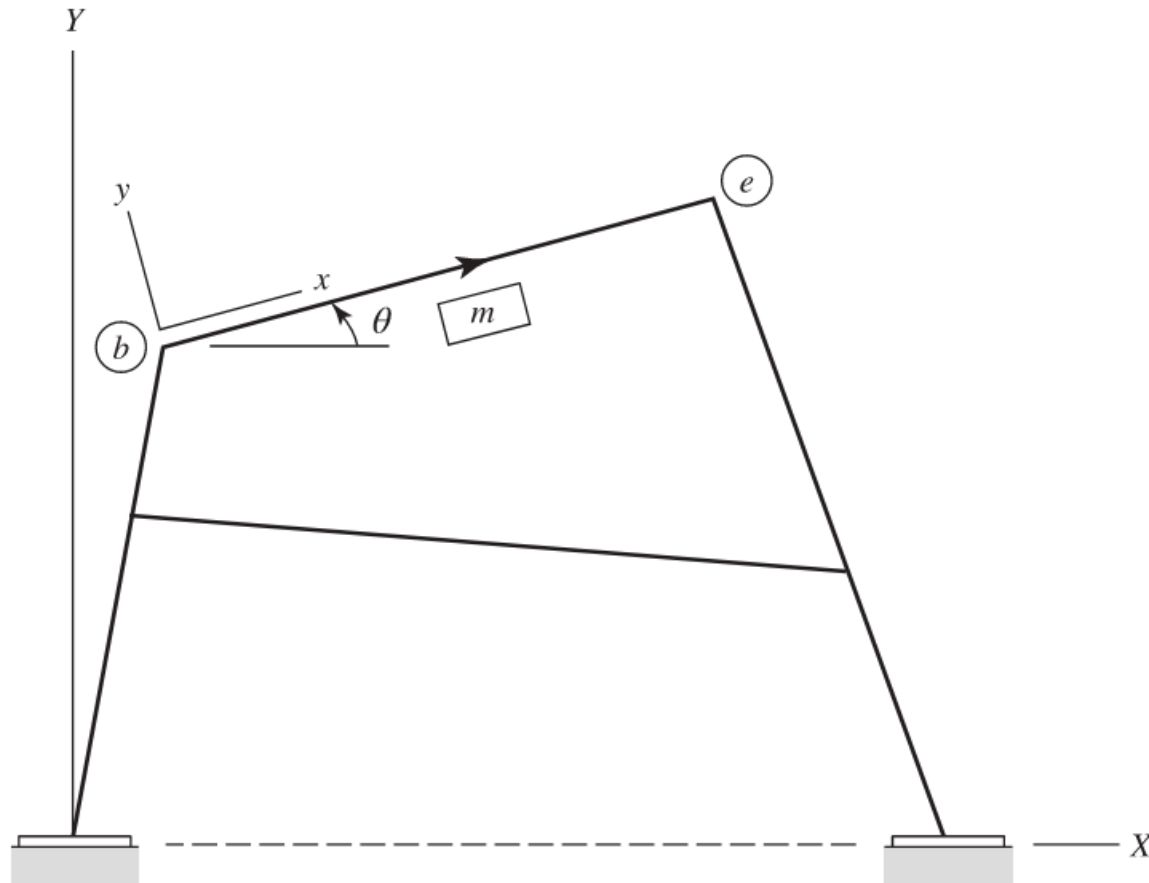
Part 1 — Roadmap: truss → beam → 3D frame

Part 2 —

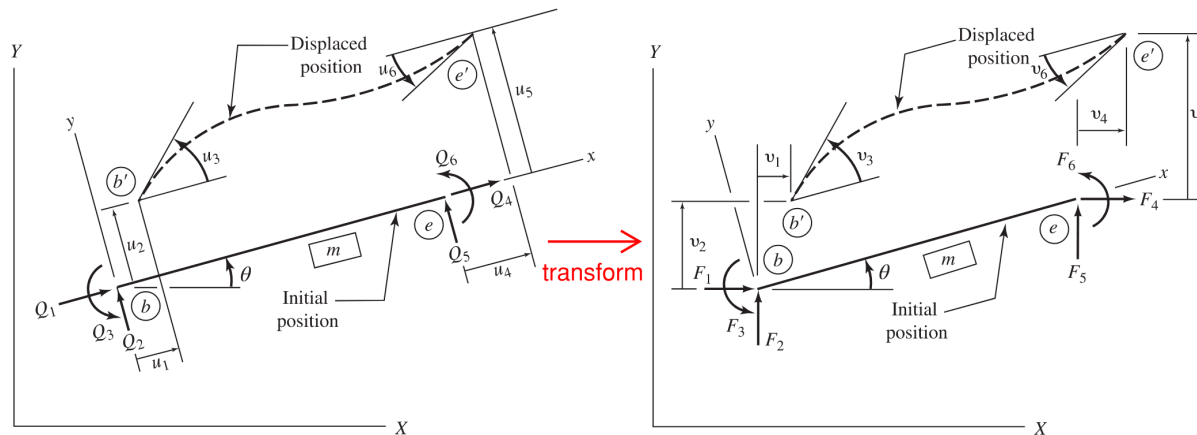
Part 1 - Local to Global Transformation

Note: See Lecture 3.2 slides (Trusses) for detailed information on this.

Generic Truss Element in a Structure



Local/Global Perspective



- **Local coordinate system (left):** forces Q , displacements u
- **Global coordinate system (right):** forces F , displacements v

Global (\mathbf{F}) \rightarrow Local (\mathbf{Q}) Force Transformation

Use trigonometric relationships to resolve global forces into the local coordinate system.

At node i :

$$\begin{aligned}Q_1 &= F_1 \cos \theta + F_2 \sin \theta \\Q_2 &= -F_1 \sin \theta + F_2 \cos \theta \\Q_3 &= F_3\end{aligned}$$

The local and global z -axes are aligned (out of plane), therefore the bending moment is unchanged:

$$Q_3 = F_3$$

At node, j :

$$\begin{aligned}Q_4 &= F_4 \cos \theta + F_5 \sin \theta \\Q_5 &= -F_4 \sin \theta + F_5 \cos \theta \\Q_6 &= F_6\end{aligned}$$

Global (\mathbf{F}) \rightarrow Local (\mathbf{Q}) Force Transformation

- Local element forces \mathbf{Q} are obtained by **rotating** global nodal forces \mathbf{F} into the member's local coordinate system
- Each node contributes a 3×3 transformation block:
 - Translational DOFs \rightarrow **rotated by θ**
 - Rotational DOF \rightarrow **invariant** (local and global z -axes are aligned)

$$\mathbf{Q} = \mathbf{T} \mathbf{F}$$

$$\begin{Bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \\ Q_6 \end{Bmatrix} = \underbrace{\begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \cos \theta & \sin \theta & 0 \\ 0 & 0 & 0 & -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{T}} \begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \end{Bmatrix}$$

Note: This is identical to the truss transformation for translational DOFs; the only addition is the rotational DOF (unchanged). See truss lecture for details on **direction cosines**.

Global (\mathbf{v}) \rightarrow Local (\mathbf{u}) Displacements

- Nodal displacements are transformed using the **same rotation matrix** as forces
- Displacements and forces transform identically because they are defined along the **same directions**

$$\mathbf{u} = \mathbf{T}\mathbf{v}$$

Local (\mathbf{Q}) \rightarrow Global (\mathbf{F}) Force

- This is the **reverse of the global \rightarrow local process**
- Local member forces or displacements are **rotated back** into the global (X, Y) directions

$$\mathbf{F} = \mathbf{T}^T \mathbf{Q}$$

Local (\mathbf{u}) \rightarrow Global (\mathbf{v}) Displacements

- This is the **reverse of the global \rightarrow local process**
- Local member forces or displacements are **rotated back** into the global (X, Y) directions

$$\mathbf{v} = \mathbf{T}^T \mathbf{u}$$

Part 2 - Global Stiffness Relationship

We start from the known local relation:

$$\mathbf{Q} = \mathbf{k}\mathbf{u} + \mathbf{Q}^F$$

where:

- \mathbf{Q} = local end force vector
- \mathbf{k} = local stiffness matrix
- \mathbf{u} = local displacement vector
- \mathbf{Q}^F = local fixed-end force vector

Step 1 — Transform Forces to Global System

Forces transform as:

$$\mathbf{F} = \mathbf{T}^T \mathbf{Q}$$

Substitute the local stiffness relation:

$$\mathbf{F} = \mathbf{T}^T (\mathbf{k}\mathbf{u} + \mathbf{Q}^F)$$

Distribute:

$$\mathbf{F} = \mathbf{T}^T \mathbf{k}\mathbf{u} + \mathbf{T}^T \mathbf{Q}^F$$

Step 2 — Transform Displacements

Displacements transform as:

$$\mathbf{u} = \mathbf{T}\mathbf{v}$$

Substitute into previous equation:

$$\mathbf{F} = \mathbf{T}^T \mathbf{k} \mathbf{T} \mathbf{v} + \mathbf{T}^T \mathbf{Q}^F$$

Step 3 — Define Global Quantities

Define:

Global Member Stiffness Matrix

$$\mathbf{K} = \mathbf{T}^T \mathbf{k} \mathbf{T}$$

Global Fixed-End Force Vector

$$\mathbf{F}^F = \mathbf{T}^T \mathbf{Q}^F$$

Final Global Member Relation

Substitute definitions:

$$\mathbf{F} = \mathbf{K}\mathbf{v} + \mathbf{F}^F$$

Consistent with our notation, where we use u for global

$$\mathbf{F} = \mathbf{K}\mathbf{u} + \mathbf{F}^F$$

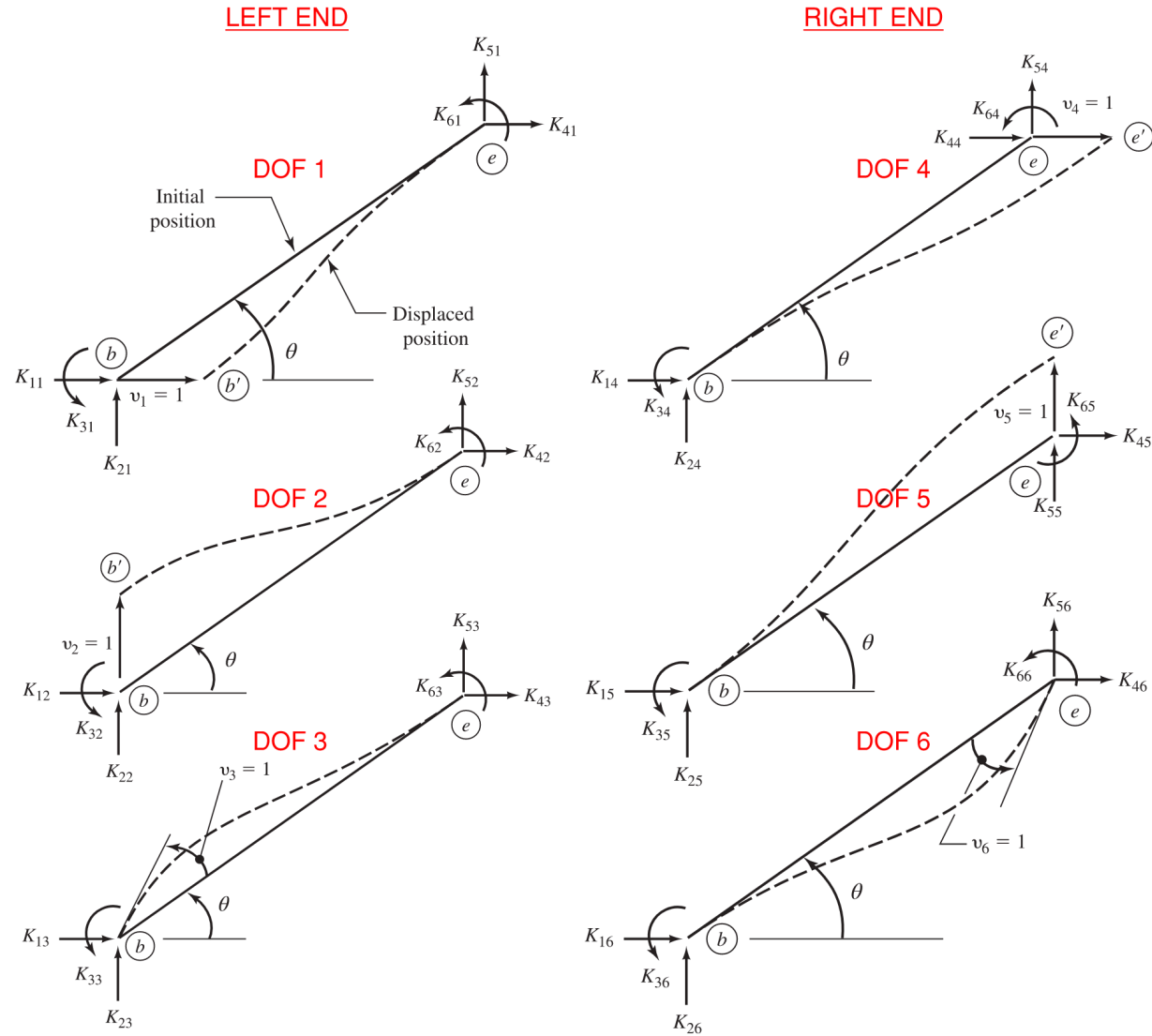
Part 3 - 2D Frame Element Stiffness Matrix

Global Coordinates

Unit Displacement Method

You can derive 2D frame element from scratch using unit displacement method.

k_{ij} = force at DOF i due to a unit displacement at DOF j ,
with all other DOFs fixed.



Closed Form Expression

A closed-form expression for the global member stiffness matrix **does exist** for frame elements (Section 6.4 in Kassimali)

However, unlike the truss case, the resulting expression is considerably more involved and not especially transparent.

$$\mathbf{K} = \frac{EI}{L^3} \begin{bmatrix} \frac{AL^2}{I} \cos^2 \theta + 12 \sin^2 \theta & \left(\frac{AL^2}{I} - 12 \right) \cos \theta \sin \theta & -6L \sin \theta & -\left(\frac{AL^2}{I} \cos^2 \theta + 12 \sin^2 \theta \right) & -\left(\frac{AL^2}{I} - 12 \right) \cos \theta \sin \theta & -6L \sin \theta \\ \left(\frac{AL^2}{I} - 12 \right) \cos \theta \sin \theta & \frac{AL^2}{I} \sin^2 \theta + 12 \cos^2 \theta & 6L \cos \theta & -\left(\frac{AL^2}{I} - 12 \right) \cos \theta \sin \theta & -\left(\frac{AL^2}{I} \sin^2 \theta + 12 \cos^2 \theta \right) & 6L \cos \theta \\ -6L \sin \theta & 6L \cos \theta & 4L^2 & 6L \sin \theta & -6L \cos \theta & 2L^2 \\ -\left(\frac{AL^2}{I} \cos^2 \theta + 12 \sin^2 \theta \right) & -\left(\frac{AL^2}{I} - 12 \right) \cos \theta \sin \theta & 6L \sin \theta & \frac{AL^2}{I} \cos^2 \theta + 12 \sin^2 \theta & \left(\frac{AL^2}{I} - 12 \right) \cos \theta \sin \theta & 6L \sin \theta \\ -\left(\frac{AL^2}{I} - 12 \right) \cos \theta \sin \theta & -\left(\frac{AL^2}{I} \sin^2 \theta + 12 \cos^2 \theta \right) & -6L \cos \theta & \left(\frac{AL^2}{I} - 12 \right) \cos \theta \sin \theta & \frac{AL^2}{I} \sin^2 \theta + 12 \cos^2 \theta & -6L \cos \theta \\ -6L \sin \theta & 6L \cos \theta & 2L^2 & 6L \sin \theta & -6L \cos \theta & 4L^2 \end{bmatrix}$$

For this reason, it is generally cleaner and more systematic to just compute:

$$\mathbf{K} = \mathbf{T}^T \mathbf{k} \mathbf{T}$$

Part 4 - Member Forces

Global Coordinates

Why Need to Rotate to Global?

We need the fixed-end forces (FEFs) in **global coordinates** before we assemble the structure, because the global system solution is written and solved in the **global coordinate system**.

Mathematical Expression

The force transformation we use for fixed-end forces is:

$$\mathbf{F}^F = \mathbf{T}^T \mathbf{Q}^F$$

Using the same \mathbf{T} as before:

$$\mathbf{T}^T = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \cos \theta & -\sin \theta & 0 \\ 0 & 0 & 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

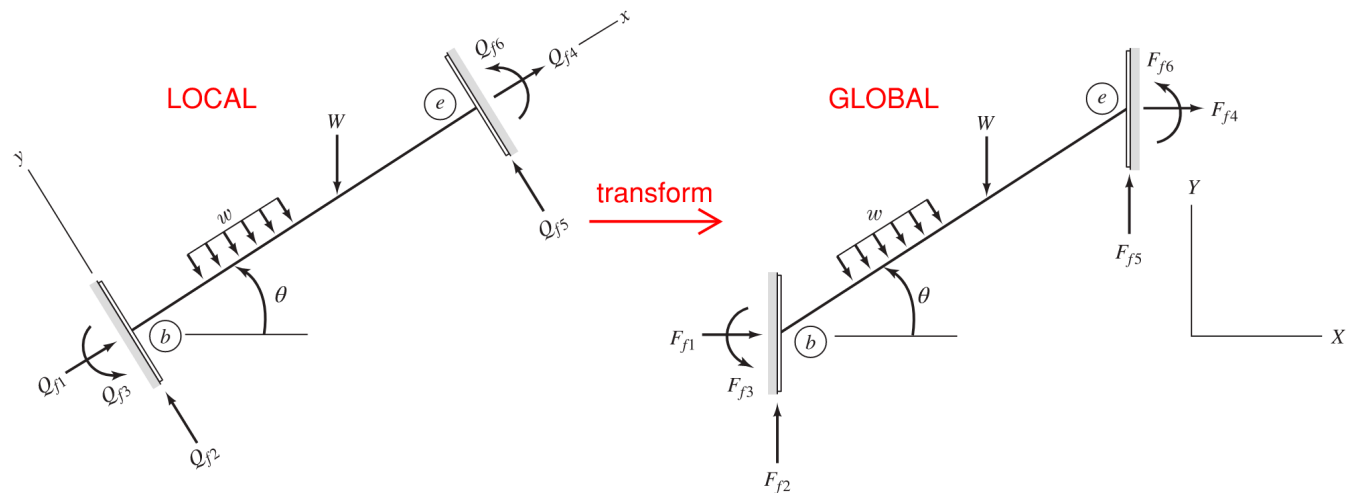
Expression for \mathbf{F}^F

Let $c = \cos \theta$ and $s = \sin \theta$. Then:

$$\begin{Bmatrix} F_1^F \\ F_2^F \\ F_3^F \\ F_4^F \\ F_5^F \\ F_6^F \end{Bmatrix} = \mathbf{T}^T \begin{Bmatrix} Q_1^F \\ Q_2^F \\ Q_3^F \\ Q_4^F \\ Q_5^F \\ Q_6^F \end{Bmatrix} = \begin{Bmatrix} c Q_1^F - s Q_2^F \\ s Q_1^F + c Q_2^F \\ Q_3^F \\ c Q_4^F - s Q_5^F \\ s Q_4^F + c Q_5^F \\ Q_6^F \end{Bmatrix}$$

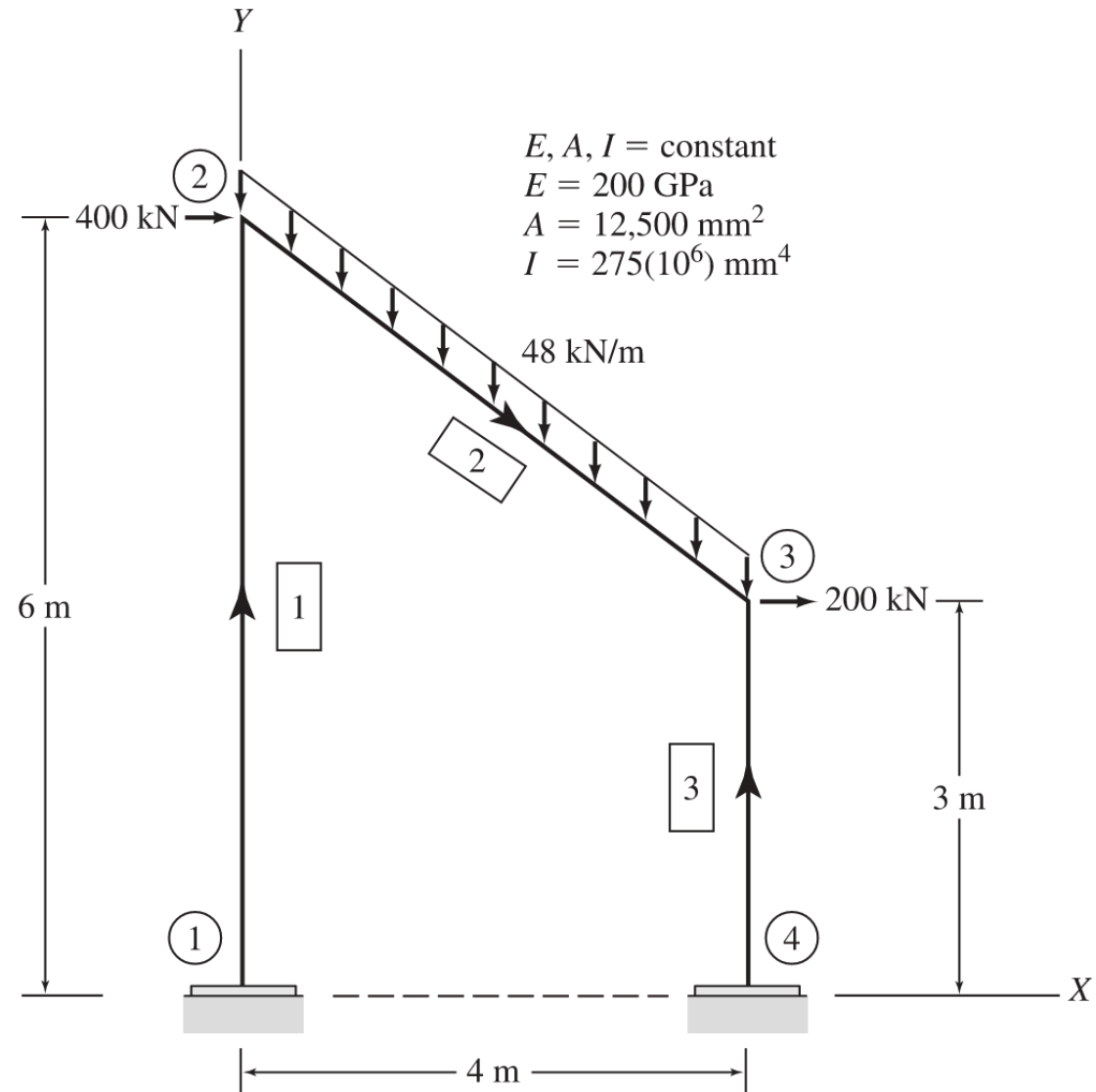
Transformation Process

1. Compute the member fixed-end force vector in **local** coordinates: \mathbf{Q}^F
2. Rotate it into **global** coordinates using: $\mathbf{F}^F = \mathbf{T}^T \mathbf{Q}^F$
3. Assemble \mathbf{F}^F into the global force vector (same coordinate system as the global stiffness matrix)



Part 5 - Example (6.3 and 6.4 in Kassimali)

Continue with same structure.

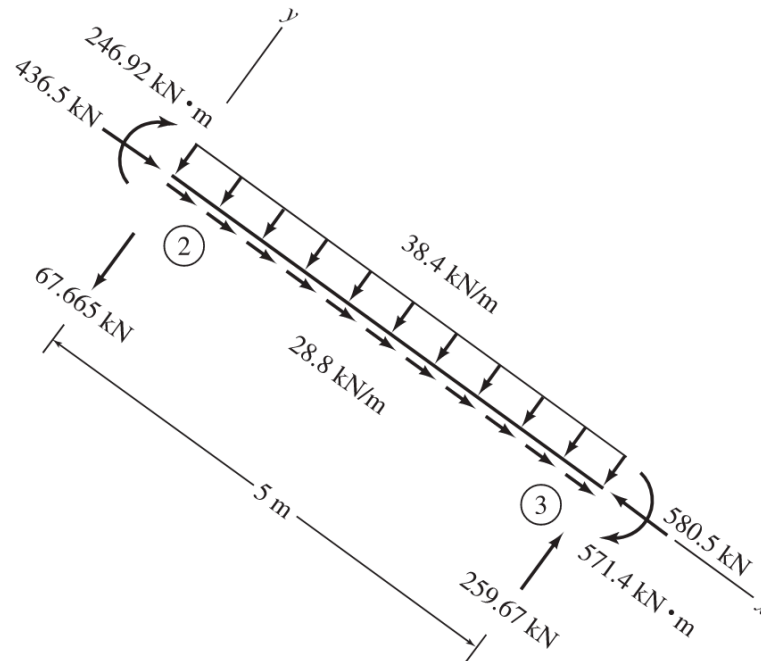


Local -> Global Element Forces

If given local element forces from earlier problem

$$\mathbf{F} = \mathbf{T}^T \mathbf{Q}$$

$$\mathbf{Q} = [436.5 \quad -67.669 \quad -246.929 \quad -580.5 \quad 259.669 \quad -571.418]^T$$



```

In [114]: import numpy as np
np.set_printoptions(precision=3, suppress=True)

def transformation_matrix(theta):
    """
    Returns the 6x6 transformation matrix T for a 2D frame element.
    """
    theta = np.radians(theta) # convert degrees → radians

    c = np.cos(theta)
    s = np.sin(theta)

    T = np.array([
        [ c, s, 0, 0, 0, 0],
        [-s, c, 0, 0, 0, 0],
        [ 0, 0, 1, 0, 0, 0],
        [ 0, 0, 0, c, s, 0],
        [ 0, 0, 0, -s, c, 0],
        [ 0, 0, 0, 0, 0, 1]
    ])

    return T

```

```
In [115]: theta = 270 + np.degrees(np.arctan(4/3))
```

```
T = transformation_matrix(theta)  
print(T)
```

```
[[ 0.8 -0.6  0.  0.  0.  0. ]  
 [ 0.6  0.8  0.  0.  0.  0. ]  
 [ 0.   0.   1.  0.  0.  0. ]  
 [ 0.   0.   0.  0.8 -0.6  0. ]  
 [ 0.   0.   0.  0.6  0.8  0. ]  
 [ 0.   0.   0.  0.   0.   1. ]]
```

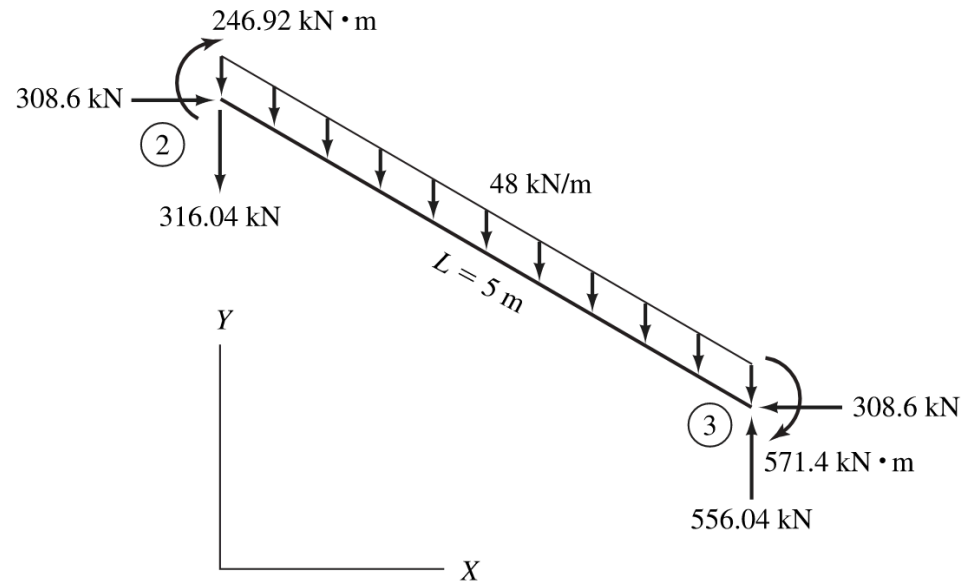


```
In [116]: Q = np.array([436.5, -67.669, -246.929, -580.5, 259.669, -571.418])
```

```
F = T.T @ Q
```

```
print(F)
```

```
[ 308.599 -316.035 -246.929 -308.599  556.035 -571.418]
```

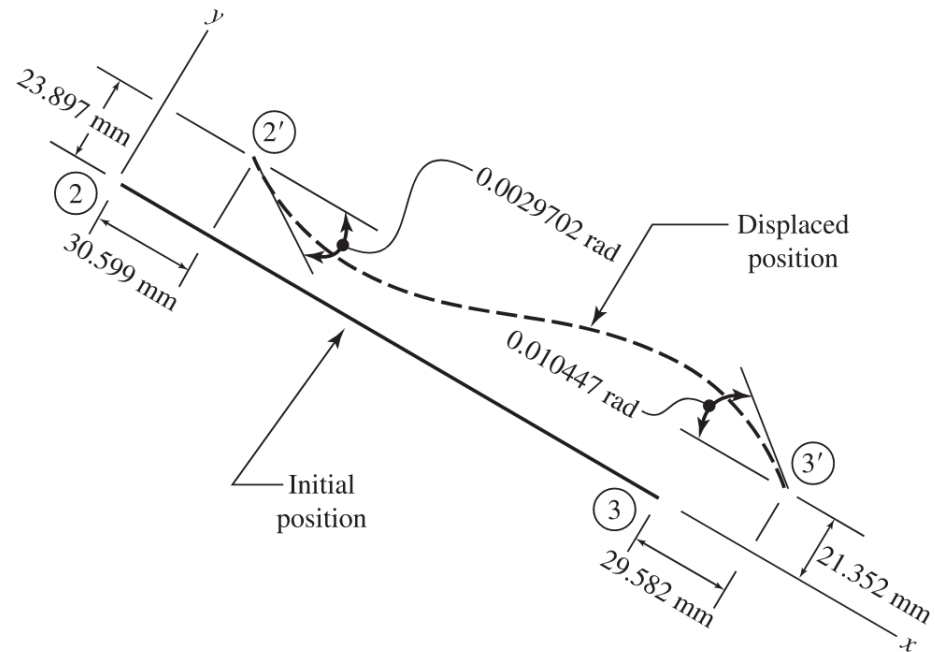


Local -> Global Element Displacements

If given local element forces from earlier problem

$$\mathbf{v} = \mathbf{T}^T \mathbf{u}$$

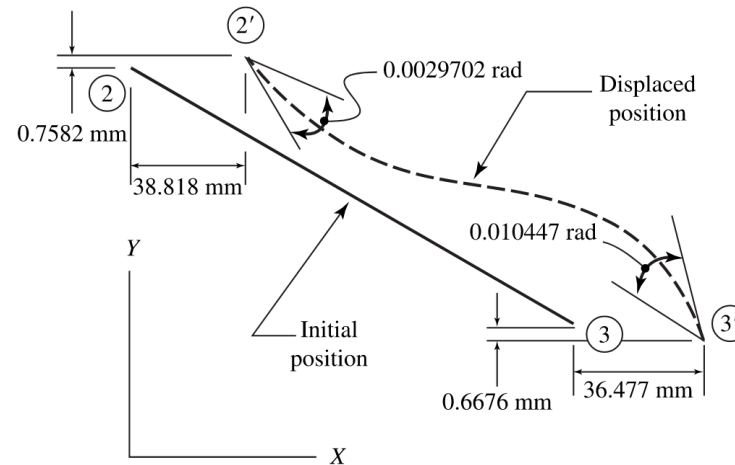
$$\mathbf{u} = [0.030599 \quad 0.023897 \quad -0.0029702 \quad 0.029582 \quad 0.021352 \quad -0.010447]^T$$



```
In [117]: # displacement vector (units: m, rad)
u = np.array([
    0.030599,
    0.023897,
    -0.0029702, # Clockwise rotation is negative
    0.029582,
    0.021352,
    -0.010447 # Clockwise rotation is negative
], dtype=float)

v = T.T @ u
print(v)
```

```
[ 0.039  0.001 -0.003  0.036 -0.001 -0.01 ]
```



Element Forces Using Global Stiffness Relationship

$$\mathbf{F} = \mathbf{T}^T \mathbf{k} \mathbf{T} \mathbf{v} + \mathbf{T}^T \mathbf{Q}^F$$

Given:

$$\mathbf{k} = \begin{bmatrix} 500000 & 0 & 0 & -500000 & 0 & 0 \\ 0 & 5280 & 13200 & 0 & -5280 & 13200 \\ 0 & 13200 & 44000 & 0 & -13200 & 22000 \\ -500000 & 0 & 0 & 500000 & 0 & 0 \\ 0 & -5280 & -13200 & 0 & 5280 & -13200 \\ 0 & 13200 & 22000 & 0 & -13200 & 44000 \end{bmatrix}$$

$$\mathbf{Q}^F = [-72 \quad 96 \quad 80 \quad -72 \quad 96 \quad -80]^T$$

$$\mathbf{v} = [0.0388174 \quad 0.0007582 \quad -0.0029702 \quad 0.0364768 \quad -0.0006676 \quad -0.010447]^T$$

```
In [118]: k = np.array([
    [ 500000,      0,      0, -500000,      0,      0],
    [      0,  5280, 13200,      0, -5280, 13200],
    [      0, 13200, 44000,      0, -13200, 22000],
    [-500000,      0,      0,  500000,      0,      0],
    [      0, -5280, -13200,      0,  5280, -13200],
    [      0, 13200, 22000,      0, -13200, 44000]
], dtype=float)

K = T.T @ k @ T

print(K)
```

```
[[ 321900.8 -237465.6   7920. -321900.8  237465.6   7920. ]
 [-237465.6  183379.2  10560.   237465.6 -183379.2  10560. ]
 [   7920.    10560.  44000.   -7920.   -10560.  22000. ]
 [-321900.8  237465.6  -7920.   321900.8 -237465.6  -7920. ]
 [  237465.6 -183379.2 -10560.  -237465.6  183379.2 -10560. ]
 [   7920.    10560.  22000.   -7920.   -10560.  44000. ]]
```

```
In [119]: Qf = np.array([-72, 96, 80, -72, 96, -80], dtype=float)

Ff = T.T @ Qf
print(Ff)

[ 0. 120.  80.   0. 120. -80.]
```

```
In [120]: # Should give same result as T.T @ Qf

def transform_Qf_to_global(Qf, theta):
    """Compute  $F^F = T^T Q^F$  using the closed-form expressions."""
    theta = np.radians(theta)
    c = np.cos(theta)
    s = np.sin(theta)

    Q1, Q2, Q3, Q4, Q5, Q6 = Qf

    Ff = np.array([
        c*Q1 - s*Q2,
        s*Q1 + c*Q2,
        Q3,
        c*Q4 - s*Q5,
        s*Q4 + c*Q5,
        Q6
    ], dtype=float)

    return Ff

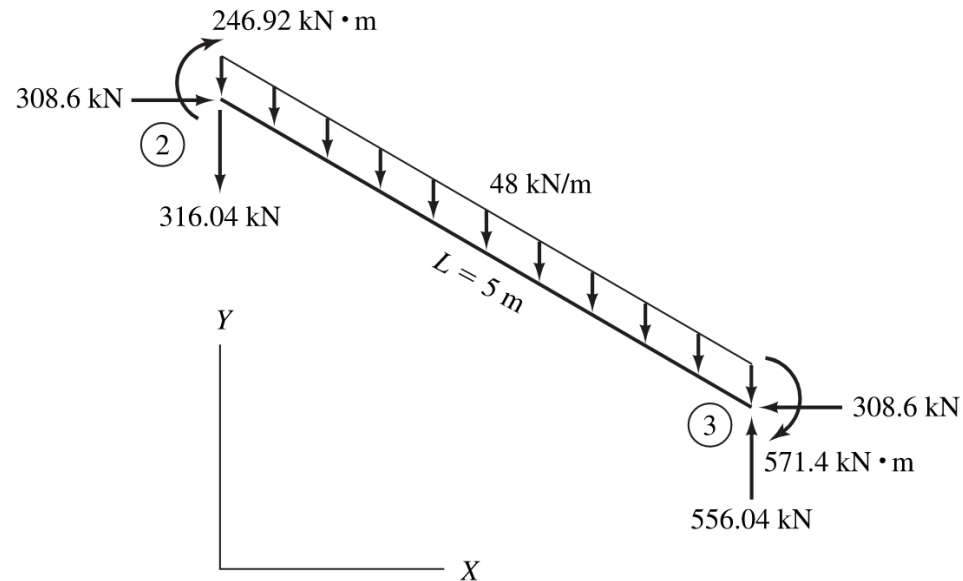
Ff = transform_Qf_to_global(Qf, theta)
print(Ff)
```

```
[ 0. 120. 80.  0. 120. -80.]
```

```
In [121]: v = np.array([0.0388174, 0.0007582, -0.0029702,
                        0.0364768, -0.0006676, -0.010447], dtype=float)
```

```
F = K @ v + Ff
print(F)
```

```
[ 308.598 -316.036 -246.929 -308.598  556.036 -571.418]
```



Part 6 — DSM Full Procedure for 2D Frames

Forward Pass — Global Structural Analysis

1. Define the structure

- Node numbering and coordinates
- Global DOF numbering
- Element connectivity
- Support restraints and applied loads

2. Compute element stiffness matrices

- For each element, compute geometry: L, θ
- Build the transformation matrix \mathbf{T}
- Compute global element stiffness:

$$\mathbf{k} = \mathbf{T}^T \mathbf{k}' \mathbf{T}$$

3. Compute element fixed-end force (FEF) vectors (if needed)

- Identify the load case
- Compute local closed-form FEFs, \mathbf{Q}^F
- Transform to global:

$$\mathbf{F}^F = \mathbf{T}^\top \mathbf{Q}^F$$

4. Assemble global stiffness matrix

- Scatter-add element stiffness contributions into \mathbf{K}

5. Assemble global load vectors

- Assemble element FEF contributions into \mathbf{F}^F
- Assemble applied joint loads into \mathbf{F}

6. Apply boundary conditions

- Partition DOFs into free (f) and restrained (r)

7. Solve for unknown displacements

$$\mathbf{u}_f = \mathbf{K}_{ff}^{-1} (\mathbf{F}_f - \mathbf{F}_f^F - \mathbf{K}_{fr} \mathbf{u}_r)$$

8. Recover support reactions

$$\mathbf{F}_r = \mathbf{K}_{rf} \mathbf{u}_f + \mathbf{K}_{rr} \mathbf{u}_r + \mathbf{F}_r^F$$

Backward Pass — Element Recovery and Design

7. Extract element global displacement vectors

- For each member, collect the relevant entries from \mathbf{u} to form $\mathbf{u}_{(e)}$

8. Transform displacements to local coordinates

$$\mathbf{u}' = \mathbf{T} \mathbf{u}_{(e)}$$

9. Extract element FEF vectors

- For each member, collect the relevant entries from \mathbf{F}^F to form $\mathbf{F}_{(e)}^F$
- You already have these from Step (3)

10. Transform FEFs to local coordinates

$$\mathbf{Q}_{(e)}^F = \mathbf{T} \mathbf{F}_{(e)}^F$$

10. **Compute local element end forces and moments**

$$\mathbf{f}' = \mathbf{k}' \mathbf{u}' + \mathbf{Q}^F$$

11. **Compute axial stress and bending stress (design quantities)** - With the beam end forces, you can draw shear and bending moment diagrams for each member

Wrap-Up

In this lecture, we:

Next lecture: