CEE6501 — Lecture 7.1

Introduction to 2D Frame Analysis

Learning Objectives

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

-

Agenda

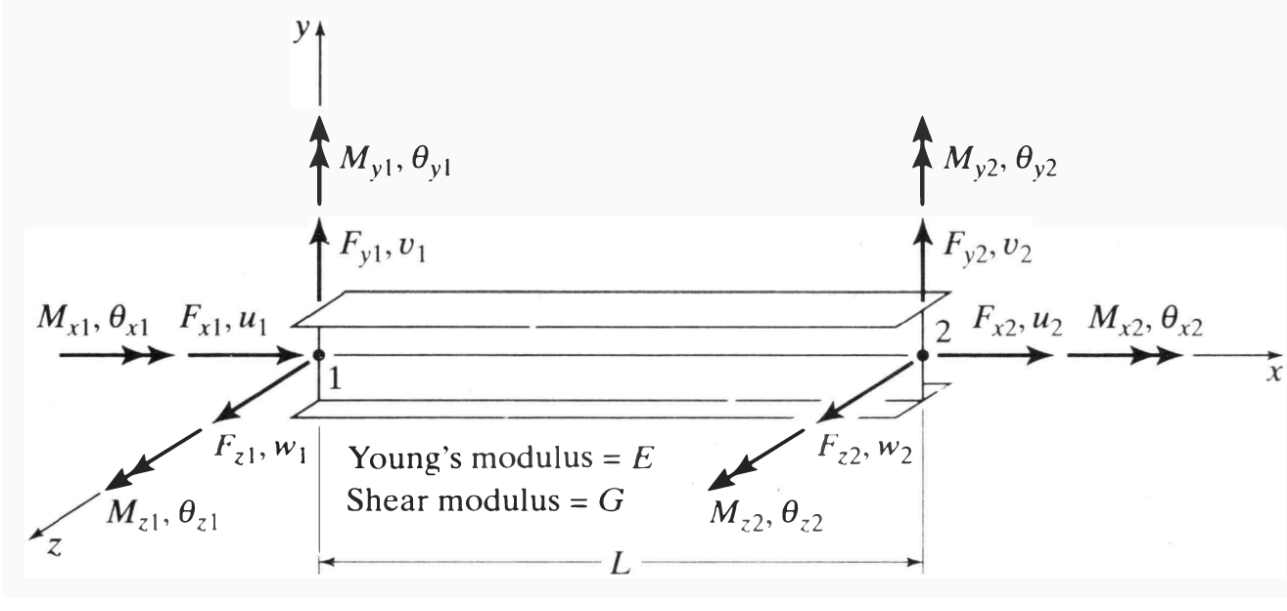
Part 1 — Roadmap: truss → beam → 3D frame
Part 2 —

Part 1 — DSM Element Roadmap

> *From Trusses to 2D Beams to 2D Frames to 3D Frames*

The 3D Frame Element (Where We Are Headed)

- A general 3D frame member has **6 DOFs per node** (12 per element)
  - translations: u_x, u_y, u_z
  - rotations: $\theta_x, \theta_y, \theta_z$
- This single element can represent **axial**, **torsion**, and **bending** behavior



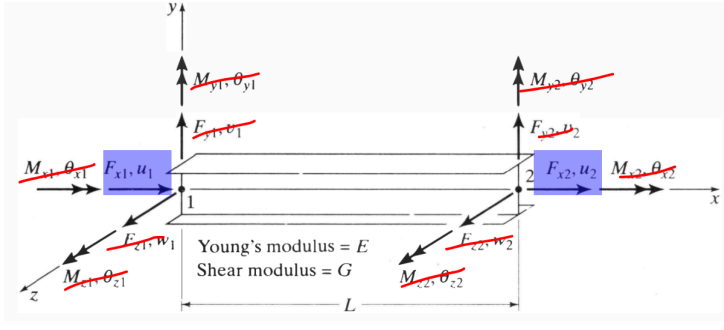
Key idea: Superposition

You can think of the 3D frame element as a **superposition** of four behaviors:

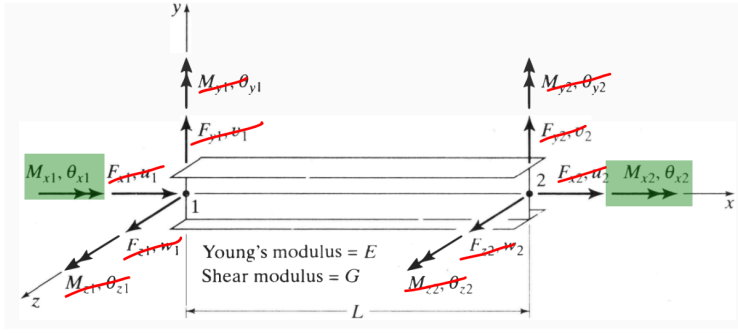
1. **Axial** deformation (truss-like)
2. **Torsion** about the member axis
3. **Bending** about one principal axis (Z - axis)
4. **Bending** about the other principal axis (Y - Axis)

The Four Behaviors

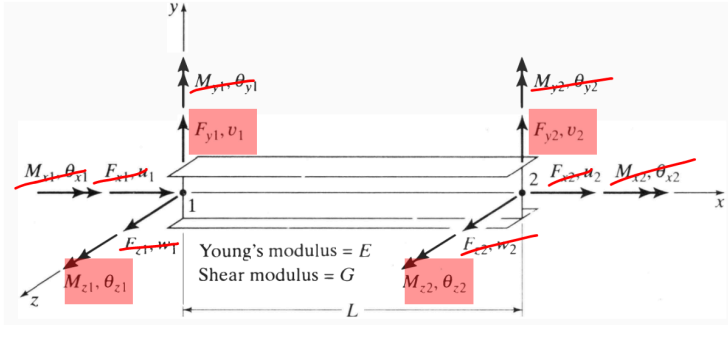
(1) Axial



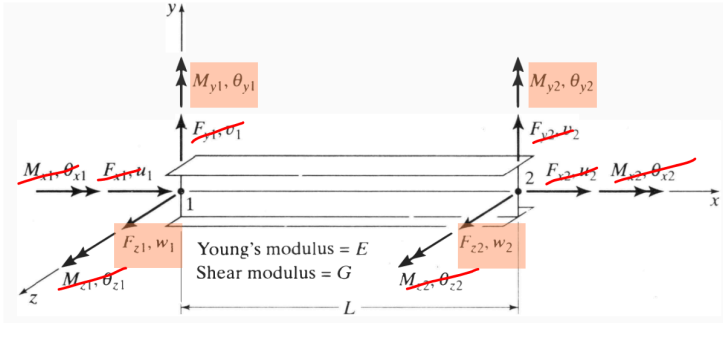
(2) Torsion



(3) Bending about Z-axis



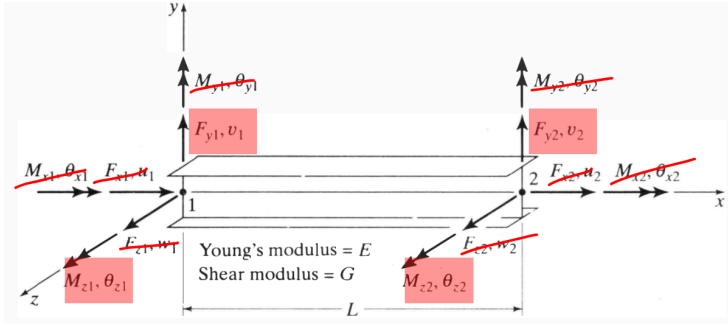
(4) Bending about Y-axis



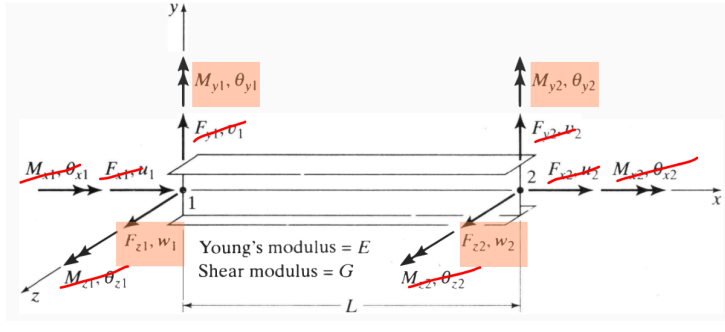
What We Derived Last Lecture

- The **2D beam element** governing planar bending behavior, case (3) and (4)
- Once derived for one principal axis, the formulation extends directly to the other (identical mathematics, different axis)

(3) Bending about Z-axis

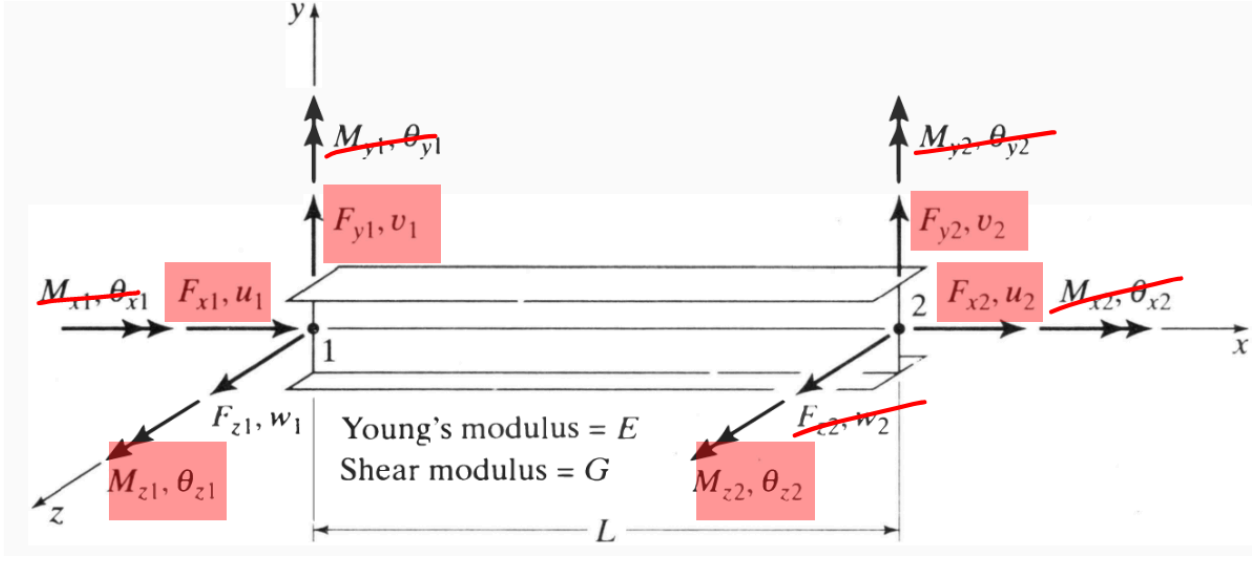


(4) Bending about Y-axis



What We Will Derive Today

- The DSM formulation for the **2D frame element**
- A combination of bending + axial behaviour **bending about the Z-axis**, case (1) + (3)
- Once derived for one principal axis, the formulation extends directly to the other (identical mathematics, different axis)



Part 2 — 2D Frame Analytical Model

> *Representing a frame structure as **members + joints** for stiffness-based analysis.*

Key Conceptual Shift (Beam → Frame)

The major difference from the pure beam formulation:

- We now include **axial deformation** in addition to bending.
- Because axial effects are included, a **frame element may be inclined** at any angle.
- A beam element (in its simplest form) was required to align with the **global X-axis**.
- A frame element therefore requires a **local coordinate system** and transformation to global coordinates.

The 2D frame element is a true generalization of the beam element.

Frame Definition (2D Idealization)

A 2D frame member is modeled as:

- A long, straight prismatic member
- Loaded in a single plane (e.g., the XY-plane)

We adopt the **Euler–Bernoulli beam assumptions**:

- Cross-sections remain **plane**
- Cross-sections remain **perpendicular to the centerline**
- **Shear deformation is neglected**

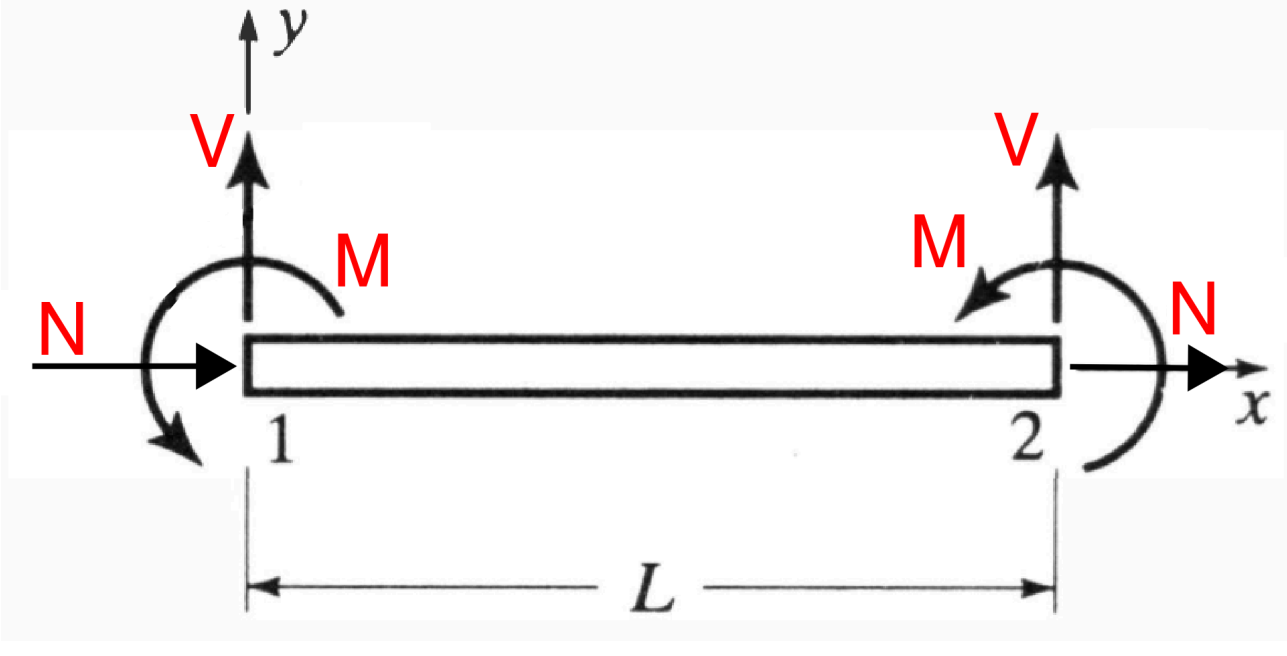
Additionally (for the frame formulation):

- **Axial deformation is included**

Consequences of This Idealization

Each member can develop:

- Axial force N: **Parallel** to the centroidal axis (axial)
- Shear force V: **Perpendicular** to the centroidal axis (shear)
- Bending moment M: In the XY-plane (about the Z-axis, into the page)



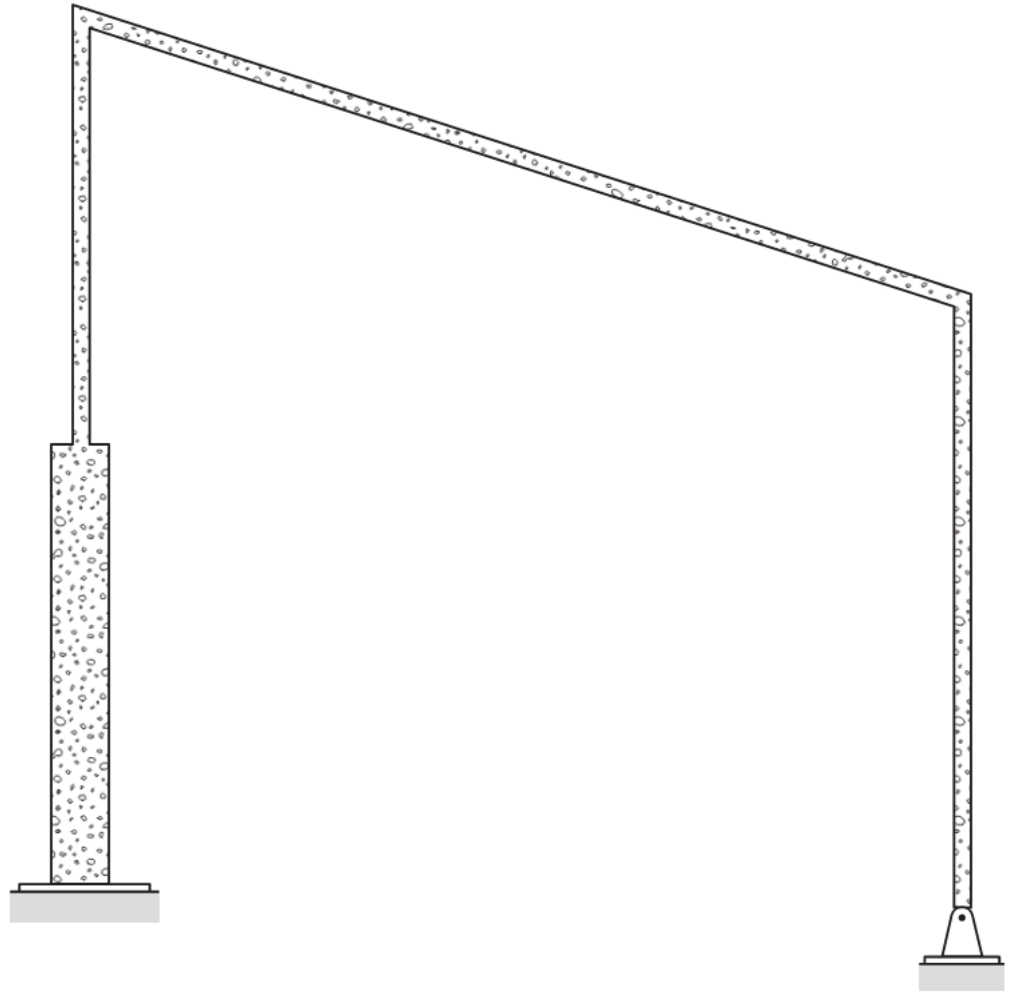
Analytical Model for the DSM

To apply the DSM, we idealize the structure as:

- Straight, prismatic members (constant A, E, I)
- Connected at discrete **joints (nodes)**
- Rigid joint connections (for now)
- Unknown reactions acting **only at joints**
- External loads applied anywhere along members (converted to FEFs)

This converts a continuous structural system into a **finite system of degrees of freedom** suitable for matrix analysis.

Example Frame Structure

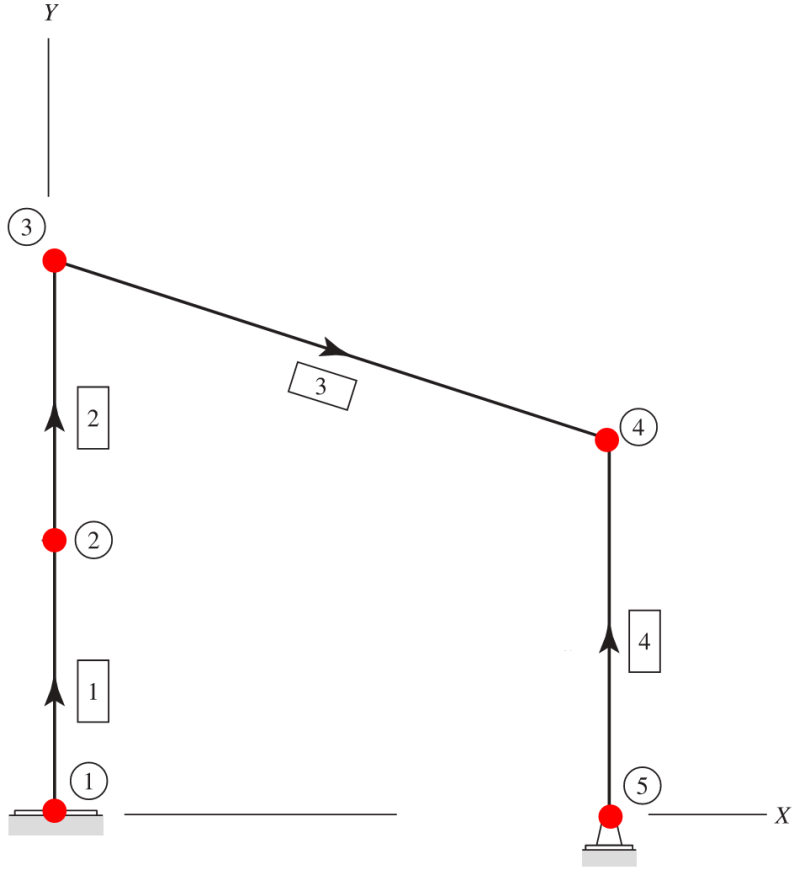


Discretize into Members and Joints

Even if the structure is physically continuous, we often insert joints so that:

- **support reactions** occur at joints (not mid-member)
- each member has **constant properties** (e.g., constant EI)
- **member loads** can be easily turned into Fixed-End Forces (more in Lecture 6.3)

Example Frame Discretization



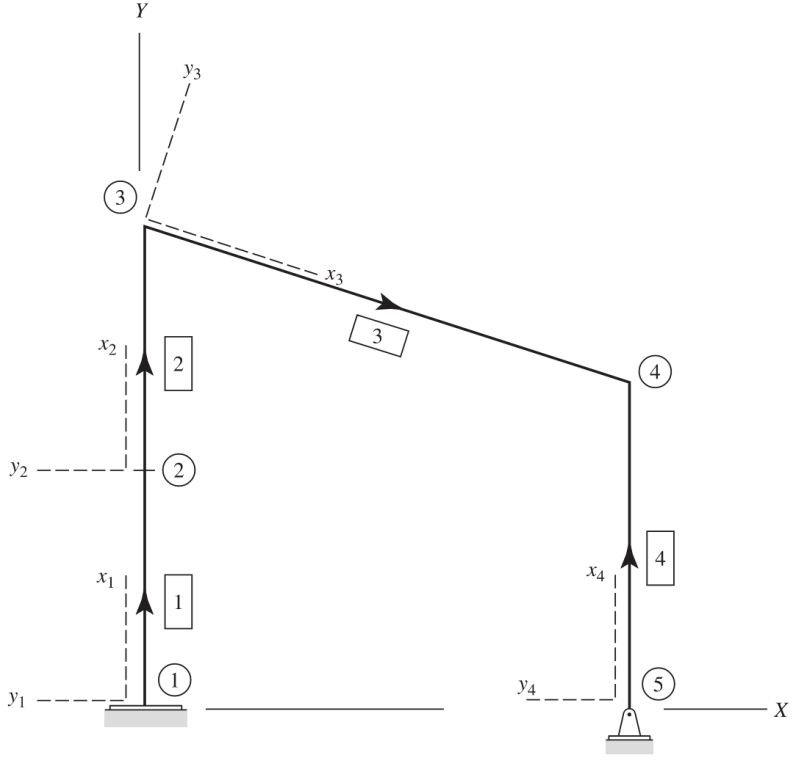
Joints

- **Joint 1:** Fixed support
- **Joint 2:** Free, Δ in EI
- **Joint 3:** Free
- **Joint 4:** Free
- **Joint 5:** Pin

Elements

- **Element 1:** Joint 1 → Joint 2
- **Element 2:** Joint 2 → Joint 3
- **Element 3:** Joint 3 → Joint 4
- **Element 4:** Joint 5 → Joint 4

Global and Local Coordinate Systems



Global

- X axis along the beam (positive to the right)
- Y axis vertical (positive upward)
- loads and reactions lie in the X–Y plane

Local

- origin at the **left end** of the member
- x axis along the member centroidal axis
- y axis vertical (positive upward)

Part 3 — Numbering DOFs, Loads, Reactions

DOFs per Joint (2D Frame Model)

So each joint can have up to **three** types of deformations:

1. horizontal translation: u (along global X)
2. vertical translation: u (along global Y)
3. rotation: θ (about global Z)

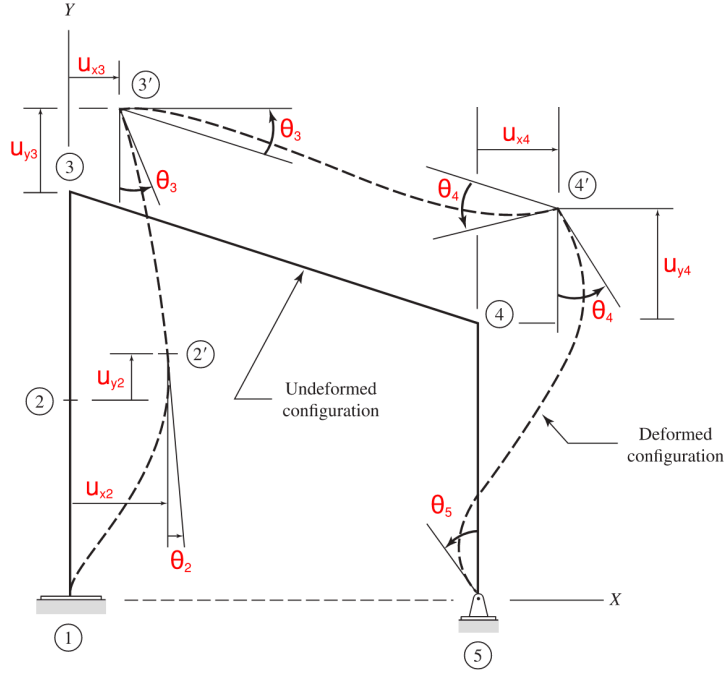
At a joint, i, we collect beam DOFs as:

$$\mathbf{u_i} = \begin{Bmatrix} u_{x,i} \\ u_{y,i} \\ \theta_i \end{Bmatrix} = \begin{Bmatrix} u_{1,i} \\ u_{2,i} \\ u_{3,i} \end{Bmatrix}$$

Sign conventions:

- u_x is positive **right**
- u_y is positive **up**
- θ is positive **counterclockwise**

Example: Deformed Shape, Showing u and θ



From the support types:

- node 1: fixed support
- node 2: free
- node 3: free
- node 4: free
- node 5: pin

Counting Degrees of Freedom

For a frame, each free joint has two translational and one rotational DOFs:

$$N_{\text{CJT}} = 3 \quad (u_x, u_y, \theta)$$

So:

$$\boxed{N_{\text{DOF}} = 3j - r}$$

This is the number of **independent joint displacements** that must be solved for.

DOF Numbering Convention (Recommended)

We adopt a **systematic, equation-based numbering scheme**:

- Number joints sequentially
- At each joint:
    1. number horizontal displacement u_x first
    2. number vertical displacement u_y next
    3. then number rotation θ
- Continue this pattern for all joints (regardless of support type)

With this convention (1-based indexing):

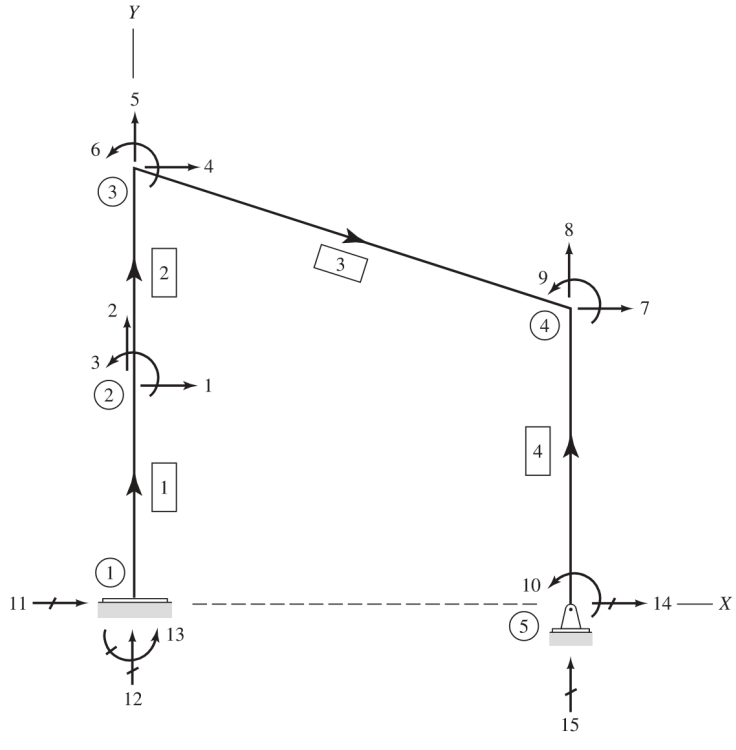
$$\mathrm{DOF}(j, u_x) = 3j - 2$$

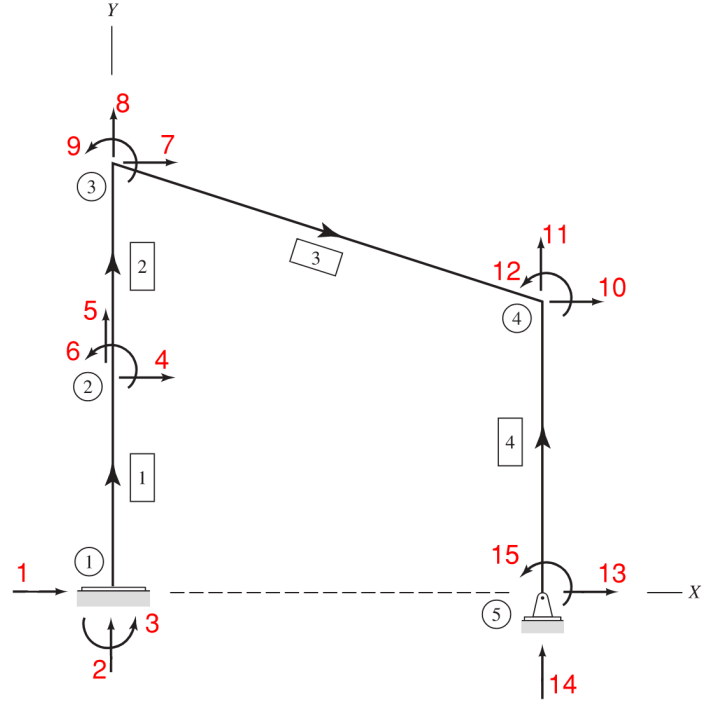
$$\mathrm{DOF}(j, u_y) = 3j - 1$$

$$\mathrm{DOF}(j, \theta) = 3j$$

This produces a fully predictable mapping between joint index and DOF number.

Textbook vs. Our Numbering





Joint Loads vs Member Loads

- **Joint loads**: forces/moments applied at joints
- **Member loads**: loads applied between joints (distributed loads, point loads on a span, a couple, etc.)

Here, the term **load** is used in a broad sense to mean either a **force** or a **moment**, applied in the direction of a DOF.

Loads and Reactions Correspond to Global Beam DOF Numbering

Each beam joint has up to three DOFs (u_x, u_y, θ).

To every DOF, there corresponds a **generalized load**.

- If the DOF is **free** → an external load may be applied
- If the DOF is **restrained** → an unknown reaction will develop

For a 2D frame element:

- $u_x \longleftrightarrow$ Axial force F_x (horizontal force)
- $u_y \longleftrightarrow$ Shear force F_y (vertical force)
- $\theta_z \longleftrightarrow$ Bending moment M_z

Part 4 — Setting up the Global System of Equations

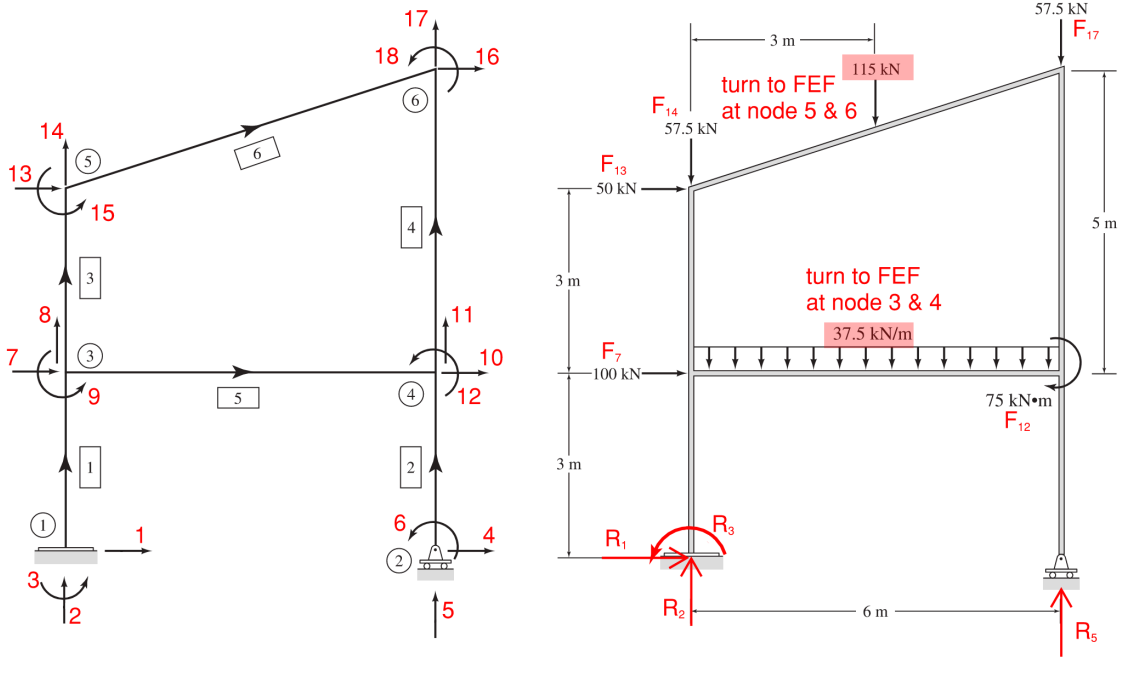
Once the beam is discretized and DOFs are numbered:

$$\boxed{\mathbf{K}\mathbf{u} = \mathbf{f} - \mathbf{f}^{\mathbf{F}}}$$

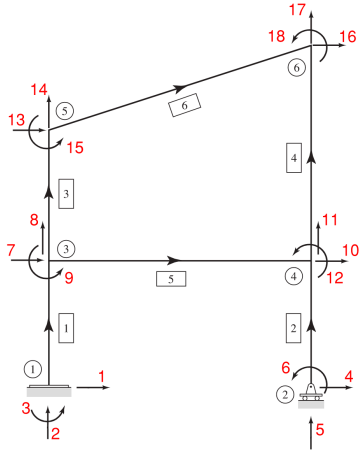
- $\mathbf{u}$ collects joint translations and rotations at the selected coordinates
- $\mathbf{f}$ collects the corresponding joint forces and moments
- $\mathbf{f}^{\mathbf{F}}$ collects the corresponding fixed end joint forces and moments
- $\mathbf{K}$ comes from **assembling beam element stiffness matrices**

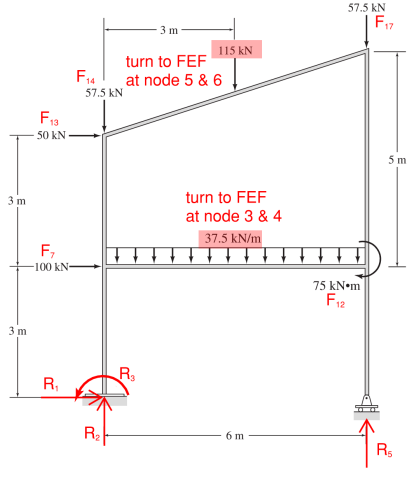
Example Structure

A frame with 6 joints (3 DOFs per joint)



Deriving FEF forces for frames will be covered later in the lecture, when talking about local member forces.





$$\mathbf{K}_{18\times 18}\begin{Bmatrix} 0 \\ 0 \\ 0 \\ u_4 \\ 0 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \\ u_{10} \\ u_{11} \\ u_{12} \\ u_{13} \\ u_{14} \\ u_{15} \\ u_{16} \\ u_{17} \\ u_{18} \end{Bmatrix} = \begin{Bmatrix} R_1 \\ R_2 \\ R_3 \\ 0 \\ R_5 \\ 0 \\ 100 \\ 0 \\ 0 \\ 0 \\ 0 \\ -75 \\ 50 \\ -57.5 \\ 0 \\ 0 \\ -57.5 \\ 0 \end{Bmatrix} - \{ F^F \}$$

Part 5 — 2D Frame Element Local Stiffness Relations

> *Local Element Mechanics*

Frame Element Response

The **member stiffness relations** express the end forces of a beam element as functions of the **end displacements**

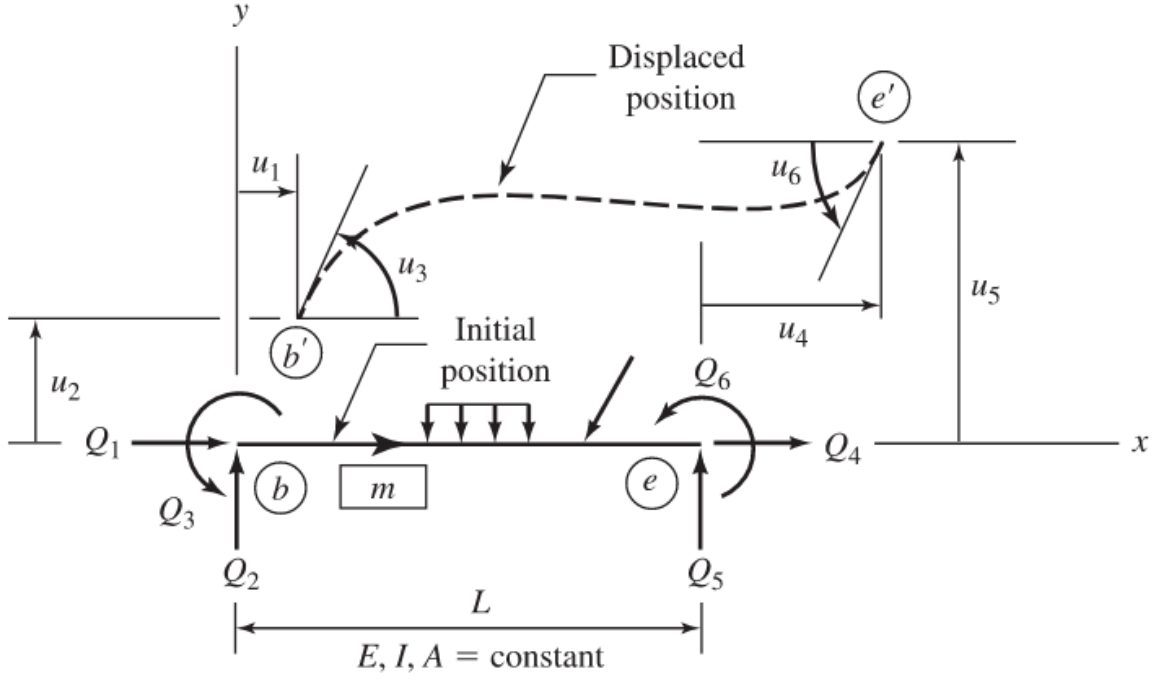
When a beam element is subjected to external loading:

- **Axial forces** develop at the ends
- **Shear forces** develop at the ends
- **Bending moments** are induced at the ends

These loads are fully determined by the **displacements and rotations at the element ends**.

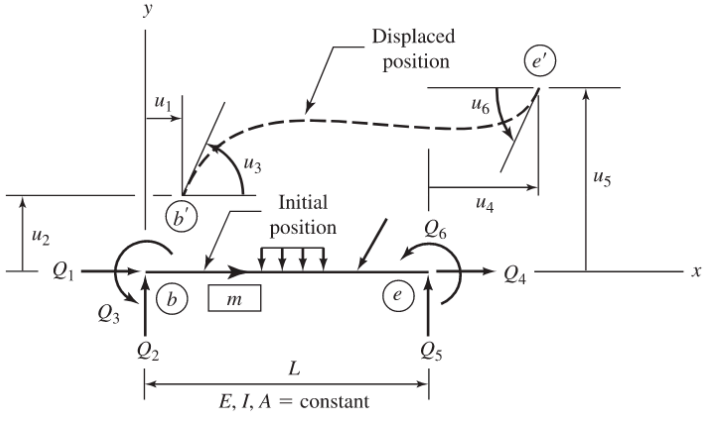
Generic Displacement for a 2D Frame Element

We define all quantities in the **local coordinate system**, with origin at the left end b, and ending at node e.



2D Beam Element DOF Numbering

Degrees of freedom are ordered from **left** → **right** node



- **DOF 1:** u_1 — node b, local x
- **DOF 2:** u_2 — node b, local y
- **DOF 3:** u_3 — node b, θ
- **DOF 4:** u_4 — node e, local x
- **DOF 5:** u_5 — node e, local y
- **DOF 6:** u_6 — node e, θ

Sign conventions:

- $+u_x$ → right (local x direction)
- $+u_y$ → up (local y direction)
- $+\theta$ → counterclockwise
- Forces follow the same order and convention

Relating Local Displacement and Force

$$\boldsymbol{Q} = \boldsymbol{k}\,\boldsymbol{u} + \boldsymbol{Q_f}$$

- $\mathbf{Q}$ denotes the 6×1 member end-force vector
- $\mathbf{u}$ denote the 6×1 member end-displacement vectors
- $\mathbf{k}$ represents the 6×6 member local stiffness matrix
- $\mathbf{Q}_f$ is the 6×1 member fixed-end force vector in the local coordinate system.

6 Linear Equations (one per DOF)

For a linear elastic element, the force at any DOF is a **linear combination** of the force induced by all DOF displacements:

- displacing one DOF can induce forces at *all* DOFs
- the proportionality constants are the stiffness coefficients k_{ij}

Each equation expresses **force equilibrium at a single local degree of freedom**.

For example DOF 1:

$$Q_1 = k_{11}u_1 + k_{12}u_2 + k_{13}u_3 + k_{14}u_4 + Q_{f,1}$$

Rather than writing out all 6 equations:

$$Q_i = \sum_{j=1}^{6} \left(k_{ij}\, u_j\right) + Q_{fi}, \qquad i = 1, 2, \ldots, 6$$

Same Equations in Matrix Form

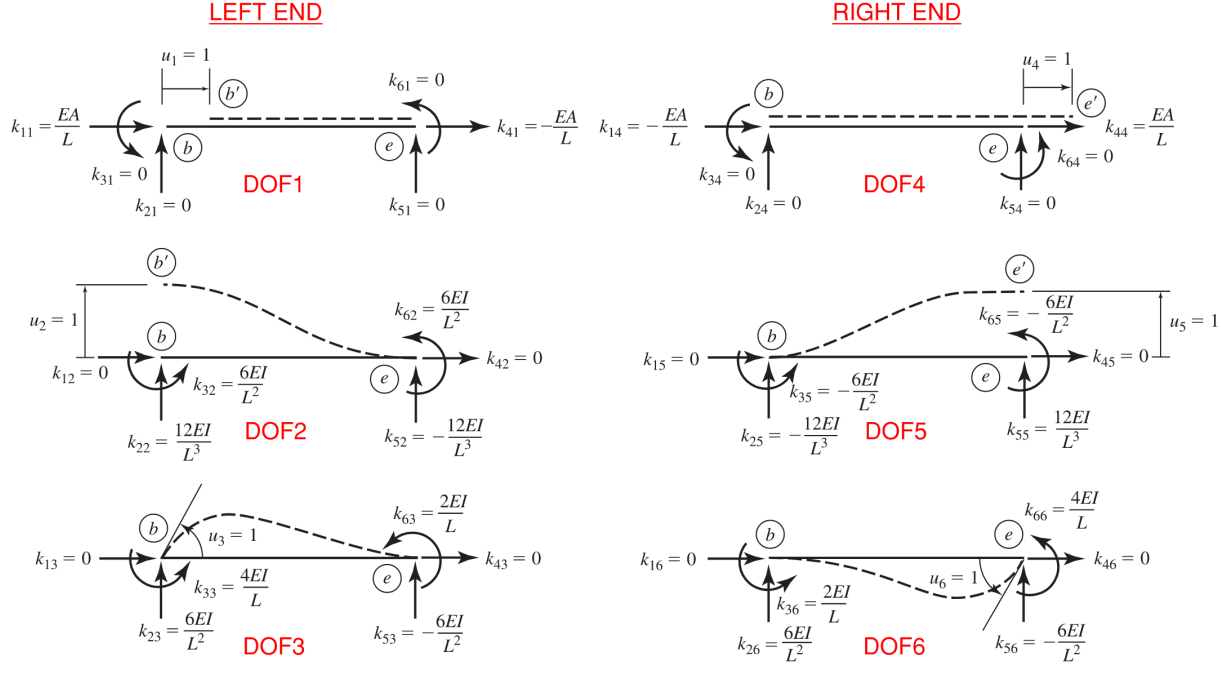
$$\begin{Bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \\ Q_6 \end{Bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} & k_{26} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} & k_{36} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} & k_{46} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & k_{56} \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{Bmatrix} + \begin{Bmatrix} Q_{f1} \\ Q_{f2} \\ Q_{f3} \\ Q_{f4} \\ Q_{f5} \\ Q_{f6} \end{Bmatrix}$$

Part 6 - 2D Frame Element Local Stiffness Matrix

Unit Displacement Method

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

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



Combination of Truss and Beam Stiffnesses

We have already derived the **axial (truss)** and **bending (beam)** stiffness matrices.

For a 2D frame element (local DOF ordering $(u_1,\, v_1,\, \theta_1,\, u_2,\, v_2,\, \theta_2)$):

- Axial effects act in **DOFs 1 and 4**
- Bending effects act in **DOFs 2, 3, 5, 6**
- In local coordinates, these effects are **decoupled**

Therefore, the frame stiffness matrix is obtained by **superposing** the two contributions:

Axial + Bending Stiffness

$$\mathbf{k}_{\text{frame}} = \underbrace{\frac{EA}{L}\begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\text{axial}} + \underbrace{\frac{EI}{L^3}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 12 & 6L & 0 & -12 & 6L \\ 0 & 6L & 4L^2 & 0 & -6L & 2L^2 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -12 & -6L & 0 & 12 & -6L \\ 0 & 6L & 2L^2 & 0 & -6L & 4L^2 \end{bmatrix}}_{\text{bending}}$$

2D Frame Stiffness Matrix

$$\mathbf{k}_{\text{frame}} = \begin{bmatrix} \frac{EA}{L} & 0 & 0 & -\frac{EA}{L} & 0 & 0 \\ 0 & \frac{12EI}{L^3} & \frac{6EI}{L^2} & 0 & -\frac{12EI}{L^3} & \frac{6EI}{L^2} \\ 0 & \frac{6EI}{L^2} & \frac{4EI}{L} & 0 & -\frac{6EI}{L^2} & \frac{2EI}{L} \\ -\frac{EA}{L} & 0 & 0 & \frac{EA}{L} & 0 & 0 \\ 0 & -\frac{12EI}{L^3} & -\frac{6EI}{L^2} & 0 & \frac{12EI}{L^3} & -\frac{6EI}{L^2} \\ 0 & \frac{6EI}{L^2} & \frac{2EI}{L} & 0 & -\frac{6EI}{L^2} & \frac{4EI}{L} \end{bmatrix}$$

2D Frame Stiffness Matrix (Factored)

$$\mathbf{k}_{\text{frame}} = \frac{EI}{L^3}\begin{bmatrix} \frac{AL^2}{I} & 0 & 0 & -\frac{AL^2}{I} & 0 & 0 \\ 0 & 12 & 6L & 0 & -12 & 6L \\ 0 & 6L & 4L^2 & 0 & -6L & 2L^2 \\ -\frac{AL^2}{I} & 0 & 0 & \frac{AL^2}{I} & 0 & 0 \\ 0 & -12 & -6L & 0 & 12 & -6L \\ 0 & 6L & 2L^2 & 0 & -6L & 4L^2 \end{bmatrix}$$

Part 7 - Local Member Forces

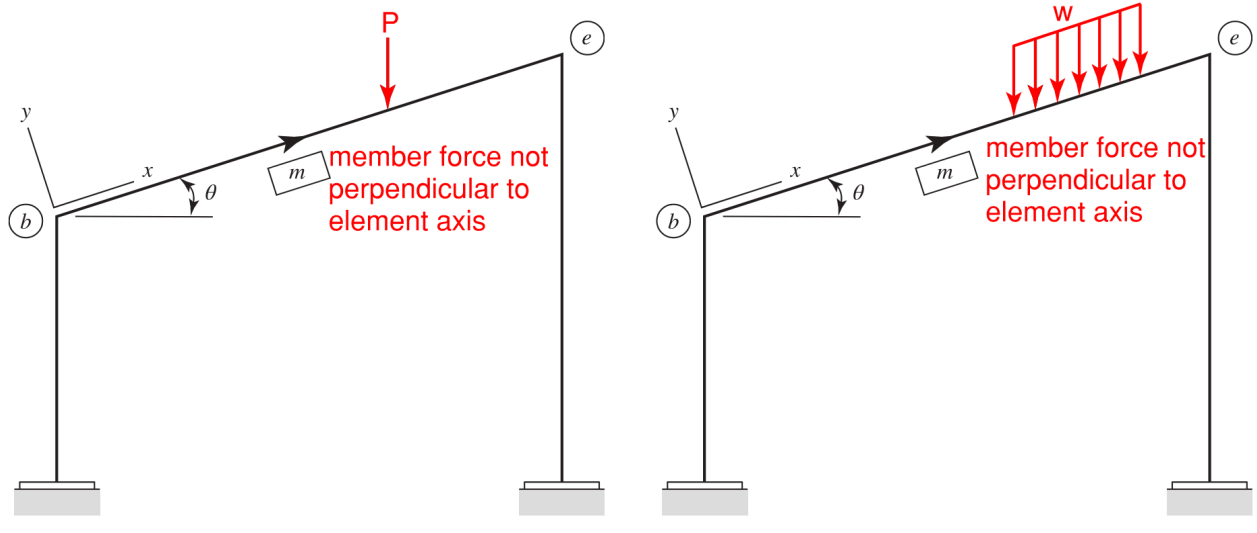
Fixed-End Forces in Frames

For frame members, two loading cases:

1. **Load perpendicular to member axis**
    - Produces **shear (V)** and **moment (M)**
    - No axial force
    - Covered in previous lecture
2. **Load not perpendicular (inclined member or load)**
    - Must resolve into components
    - Produces:
        - Axial force (N)
        - Shear (V)
        - Moment (M)

Inclined Member Forces

- In **beams**:
  - Loads are applied **perpendicular** to the member axis
  - No need for resolution
- In **frames**:
  - Members and loads may be **inclined**
  - Forces must be **resolved into local components**



Resolving Applied Forces

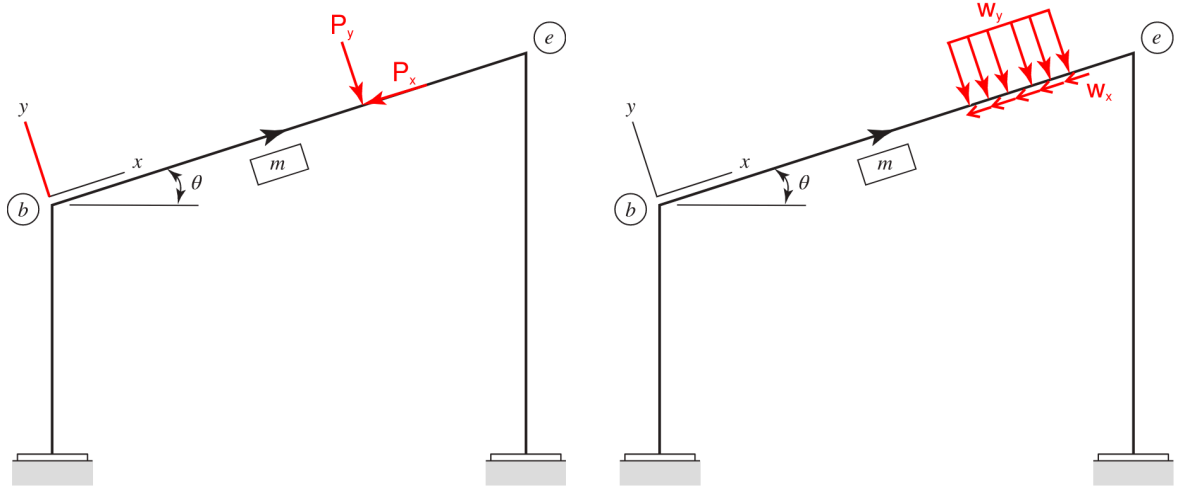
Use trigonometry to decompose loads:

Point Load:

$$P_x = P\sin\theta, \qquad P_y = P\cos\theta$$

Distributed Load:

$$w_x = w\sin\theta, \qquad w_y = w\cos\theta$$



Sign converntion for applied load: positive is equal and opposite to local element coordinate frame

For the case shown: +y = down, +x = left

Two Components ⟶ Two Effects

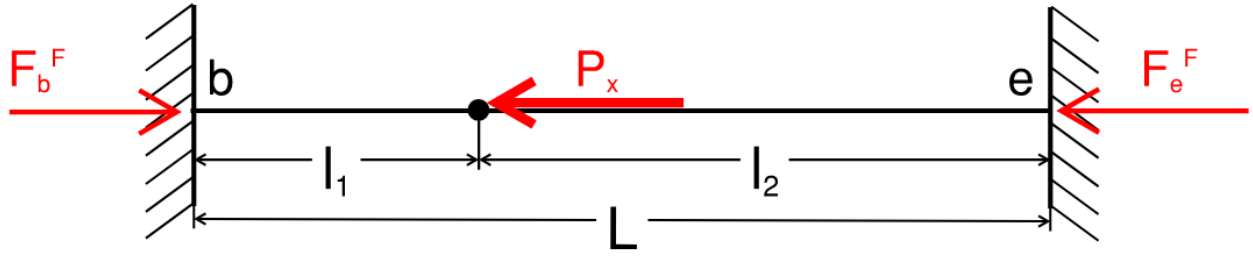
- **Perpendicular component (P_y, w_y):**
  - Use standard beam FEF formulas
  - Produces **shear and bending moment**
- **Parallel component (P_x, w_x):**
  - Produces **axial force**
  - Acts along the member axis

Axial Fixed-End Forces

- Correspond to **axial DOFs**:
    - Q_1 and Q_4
- Question:
    - What are the **fixed-end axial forces** at each node due to loads **parallel to the member axis**?

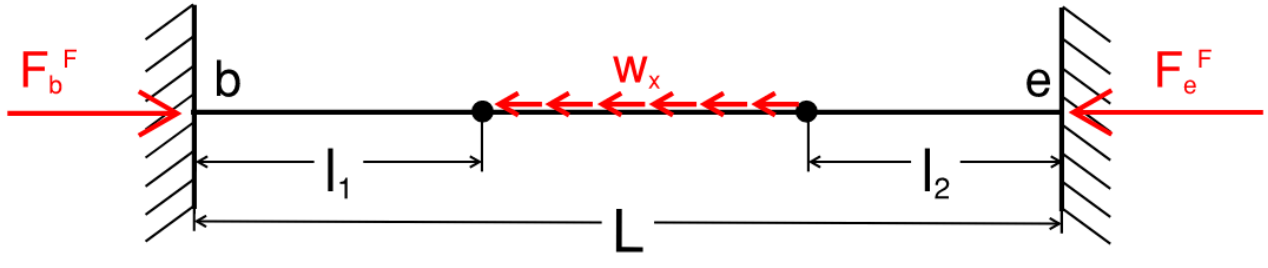
These following expressions come from equilibrium of an **axially loaded member**. See Kassimali — Section 6.2, for full derivation.

Axial FEF — Point Load



$$F_b^F = P_x \frac{l_2}{L}, \qquad F_e^F = P_x \frac{l_1}{L}$$

Axial FEF — Distributed Load



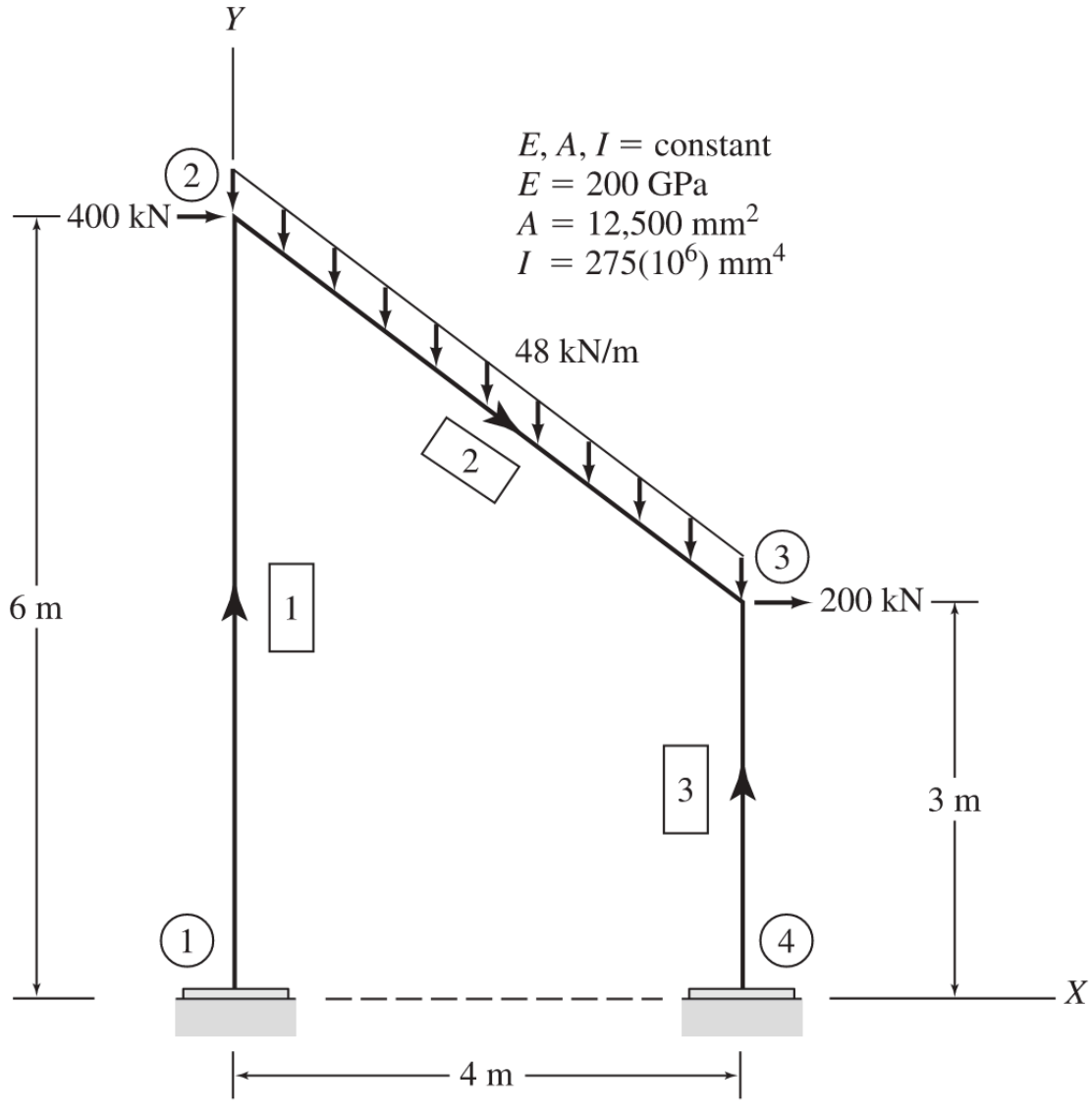
$$F_b^F = \frac{w_x}{2L}(L - l_1 - l_2)(L - l_1 + l_2)$$

$$F_e^F = \frac{w_x}{2L}(L - l_1 - l_2)(L + l_1 - l_2)$$

Key takeaway:

- **Inclined member loads → axial + shear + moment**
- Must account for all three in frame analysis as FEFs

Part 8 - Example


Y
E, A, I = constant
E = 200 GPa
A = 12,500 mm²
I = 275(10⁶) mm⁴
2
400 kN
48 kN/m
2
3
200 kN
6 m
1
3
3 m
1
4
X
4 m

1. **Compute the local stiffness matrix.**

For member 2, use:

$$E = 200\ \text{GPa}, \quad A = 12{,}500 \times 10^{-6}\ \text{m}^2, \quad I = 275 \times 10^{-6}\ \text{m}^4,$$
$$L = \sqrt{4^2 + 3^2} = 5\ \text{m}.$$

In [155]:

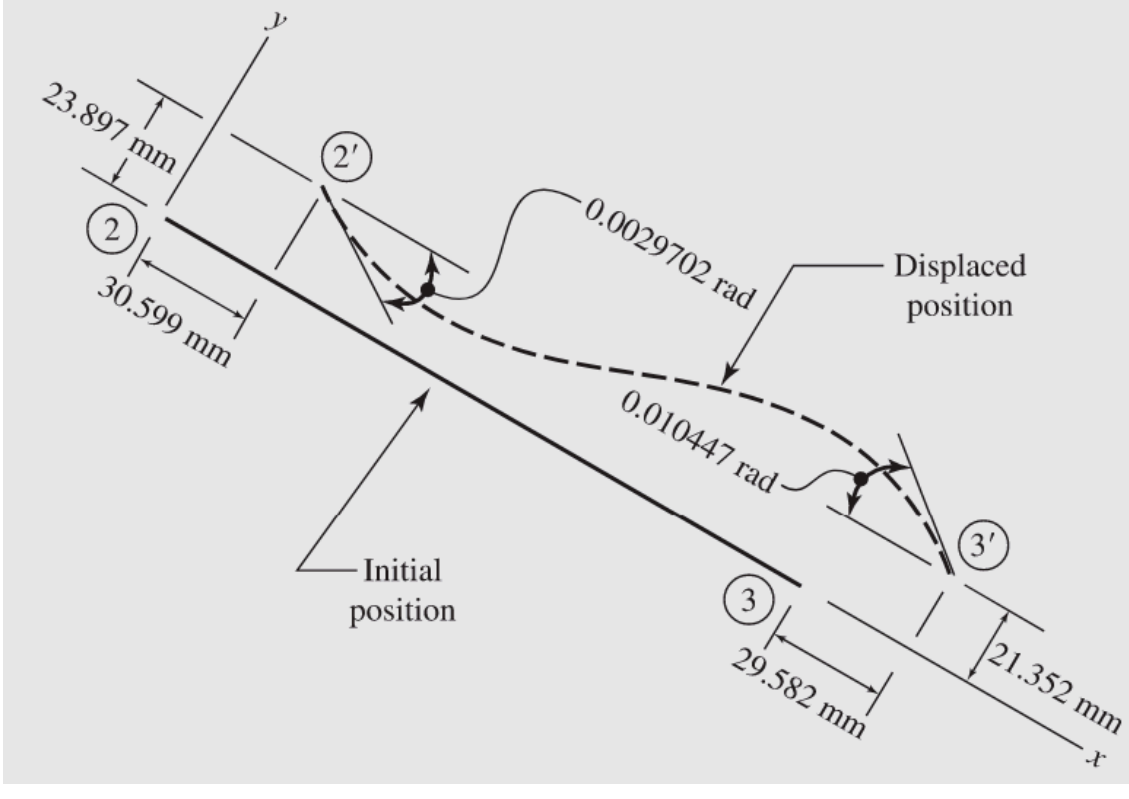
```python
import numpy as np
np.set_printoptions(precision=1, suppress=True)

def frame_element_kl(E, A, I, L):
    """
    Return 6x6 *local* stiffness matrix for a 2D frame element
    """

    factor = (E * I) / (L**3)

    kl = factor * np.array(
        [
            [ (A*L**2)/I,  0.0,      0.0,     -(A*L**2)/I,  0.0,      0.0],
            [ 0.0,         12.0,     6.0*L,    0.0,        -12.0,     6.0*L],
            [ 0.0,         6.0*L,    4.0*L**2, 0.0,        -6.0*L,    2.0*L**2
            [-(A*L**2)/I,  0.0,      0.0,      (A*L**2)/I,  0.0,      0.0],
            [ 0.0,        -12.0,    -6.0*L,    0.0,         12.0,    -6.0*L],
            [ 0.0,         6.0*L,    2.0*L**2, 0.0,        -6.0*L,    4.0*L**2
        ],
        dtype=float,
    )

    return kl
```

In [156]:
```
# --- properties in kN and m ---
E = 200e6           # kN/m^2  (converted from 200e9 N/m^2)
A = 12500e-6        # m^2
I = 275e-6          # m^4
L = 5.0             # m

k_local = frame_element_kl(E, A, I, L)

np.set_printoptions(precision=3, suppress=True)
print(k_local)

k_local = frame_element_kl(E, A, I, L)
print(k_local)
```

```
[[ 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.]]
[[ 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.]]
```

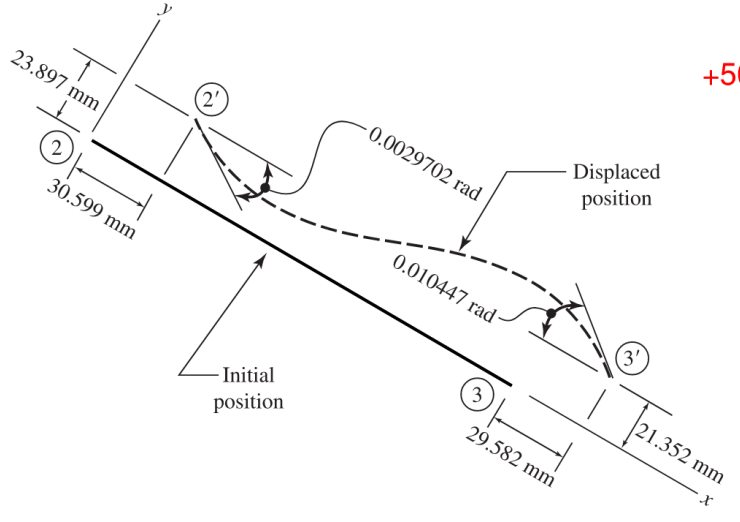
2. Given the following local displacement pattern, calculate the local member forces.

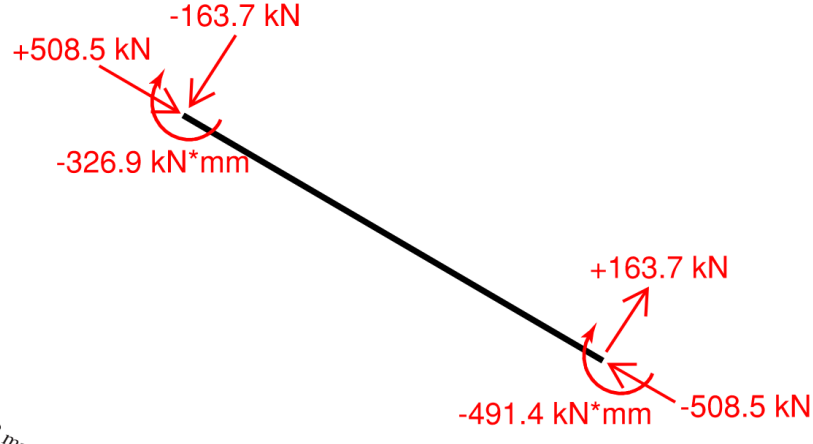
$$\boldsymbol{Q} = \boldsymbol{k}_{local}\boldsymbol{u}$$



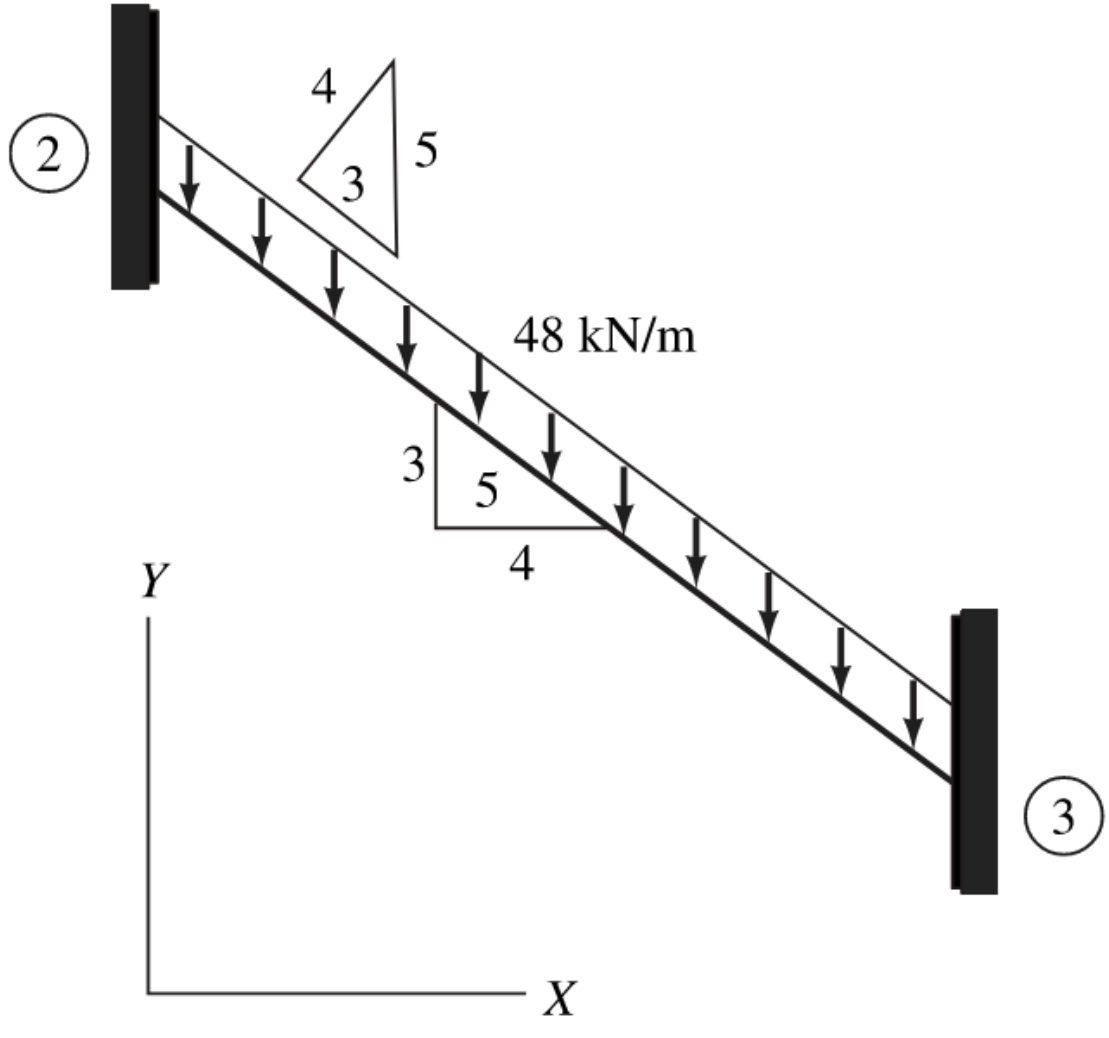
In [157]:
```python
# displacement vector (units: m, rad)
u = np.array([
    0.030599,
    0.023897,
   -0.0029702, # Clockwise rotation is negative
    0.029582,
    0.021352,
   -0.010447 # Clockwise rotation is negative
], dtype=float)

Q = k_local @ u
print(Q)
```

```
[ 508.5   -163.669 -326.929 -508.5    163.669 -491.418]
```


y
23.897 mm
2′
2
30.599 mm
0.0029702 rad
Displaced
position
0.010447 rad
3′
Initial
position
3
29.582 mm
21.352 mm
x


-163.7 kN
+508.5 kN
-326.9 kN*mm
+163.7 kN
-491.4 kN*mm
-508.5 kN

3. Given the applied member load, turn this into equivalent FEFs



In [158]:
```python
import numpy as np

def resolve_uniform_load_to_local(w_global_down, theta):
    """
    Resolve a *vertical downward* uniform load (kN/m)
    into local (x, y) components using angle theta (degrees).

    Convention:
        w_x = w * sin(theta)
        w_y =  w * cos(theta)

    Returns:
        (w_x, w_y) in kN/m
    """
    theta = np.radians(theta)  # Convert degrees to radians

    w_x = w_global_down * np.sin(theta)
    w_y =  w_global_down * np.cos(theta)
    return w_x, w_y
```

In [159]:
```
theta = np.arctan(-3/4)  # radians
theta = np.degrees(theta)
print(theta)

# Same as
theta = 270 + np.degrees(np.arctan(4/3))
print (theta)

w = 48.0

w_x, w_y = resolve_uniform_load_to_local(w, theta)
print(f"w_x (kN): {w_x:.2f}")
print(f"w_y (kN): {w_y:.2f}")
```

```
-36.86989764584402
323.13010235415595
w_x (kN): -28.80
w_y (kN): 38.40
```

Perpendicular Component (w_y) → Fixed-End Forces

For a **uniform load perpendicular to the member axis**, use standard beam fixed-end force relationships.

$$\{\boldsymbol{F}^F\}_y = \begin{Bmatrix} 0 \\ \frac{w_y L}{2} \\ \frac{w_y L^2}{12} \\ 0 \\ \frac{w_y L}{2} \\ -\frac{w_y L^2}{12} \end{Bmatrix}$$

- Produces **shear** and **bending moment**
- Acts along DOFs 2,3,5,6

In [160]:
```python
def fef_uniform_perpendicular(wy, L):
    """
    Fixed-end forces for a fixed-fixed member under a uniform transverse load w
    acting in the local y-direction (signed).

    DOF order: [F1, F2, F3, F4, F5, F6] = [u_i, v_i, th_i, u_j, v_j, th_j]
    """
    F1 = 0.0
    F2 = wy * L / 2
    F3 = wy * L**2 / 12
    F4 = 0.0
    F5 = wy * L / 2
    F6 = -wy * L**2 / 12
    return np.array([F1, F2, F3, F4, F5, F6], dtype=float)
```

Parallel Component (w_x) → Fixed-End Forces

For a **uniform load parallel to the member axis**, the equivalent nodal forces are purely axial:

$$\{\boldsymbol{F}^F\}_x = \begin{Bmatrix} \frac{w_x L}{2} \\ 0 \\ 0 \\ \frac{w_x L}{2} \\ 0 \\ 0 \end{Bmatrix}$$

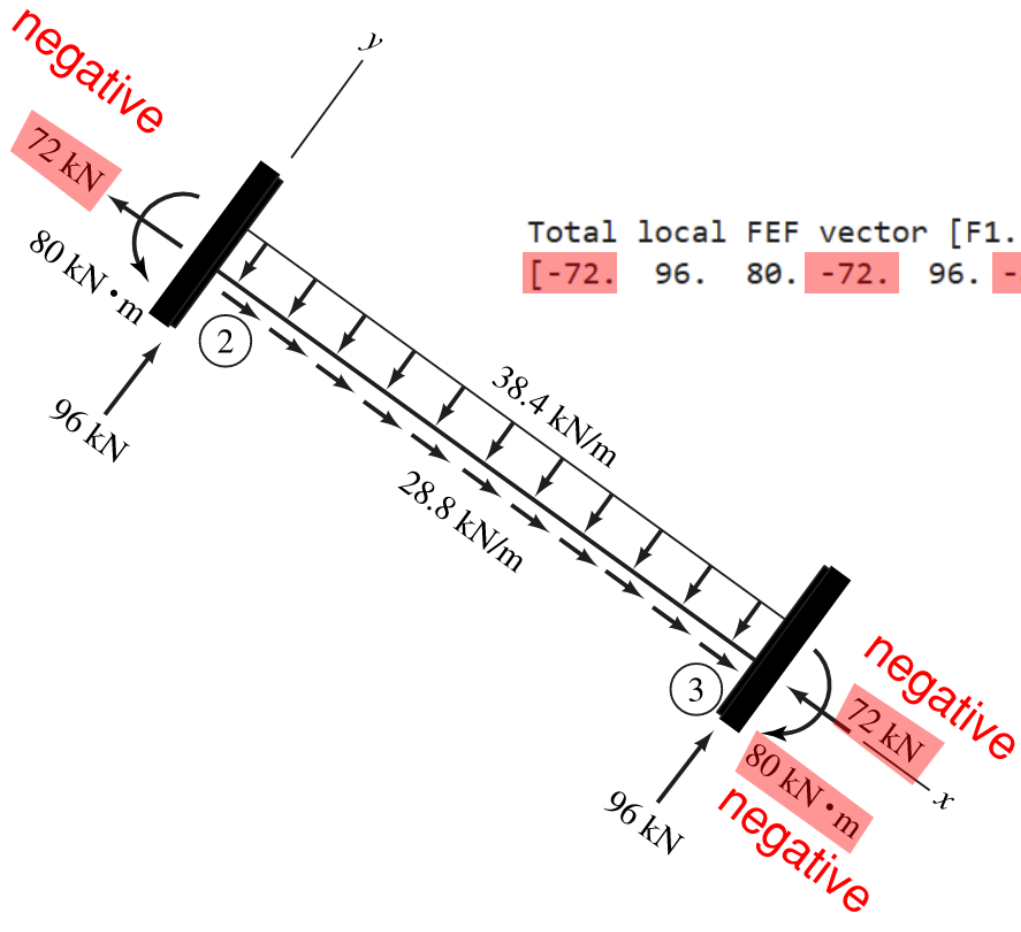
- Produces **axial force only**
- Acts along DOFs 1 and 4

In [161]:

```python
def fef_uniform_parallel(wx, L, l1, l2):
    """
    Fixed-end forces for a distributed axial load wx (kN/m)

    Parameters
    ----------
    wx : Axial distributed load (kN/m)
    L : Member length (m)
    l1 : Distance from node i to start of load (m)
    l2 : Distance from node j to end of load (m)
    """

    Fb = (wx / (2 * L)) * (L - l1 - l2) * (L - l1 + l2)
    Fe = (wx / (2 * L)) * (L - l1 - l2) * (L + l1 - l2)

    F1 = Fb
    F2 = 0.0
    F3 = 0.0
    F4 = Fe
    F5 = 0.0
    F6 = 0.0

    return np.array([F1, F2, F3, F4, F5, F6], dtype=float)
```

In [162]:
```
# --- perpendicular FEF (beam part) ---
Qf_y = fef_uniform_perpendicular(wy=w_y, L=L)

# --- parallel FEF (axial part) ---
Qf_x = fef_uniform_parallel(wx=w_x, L=L, l1=0.0, l2=0.0)

# --- total FEF vector ---
Qf_total = Qf_x + Qf_y

print("FEF from perpendicular component:")
print(Qf_y, "\n")

print("FEF from parallel component:")
print(Qf_x, "\n")

print("Total local FEF vector [F1..F6]:")
print(Qf_total)
```

```
FEF from perpendicular component:
[  0.  96.  80.   0.  96. -80.]

FEF from parallel component:
[-72.   0.   0. -72.   0.   0.]

Total local FEF vector [F1..F6]:
[-72.  96.  80. -72.  96. -80.]
```



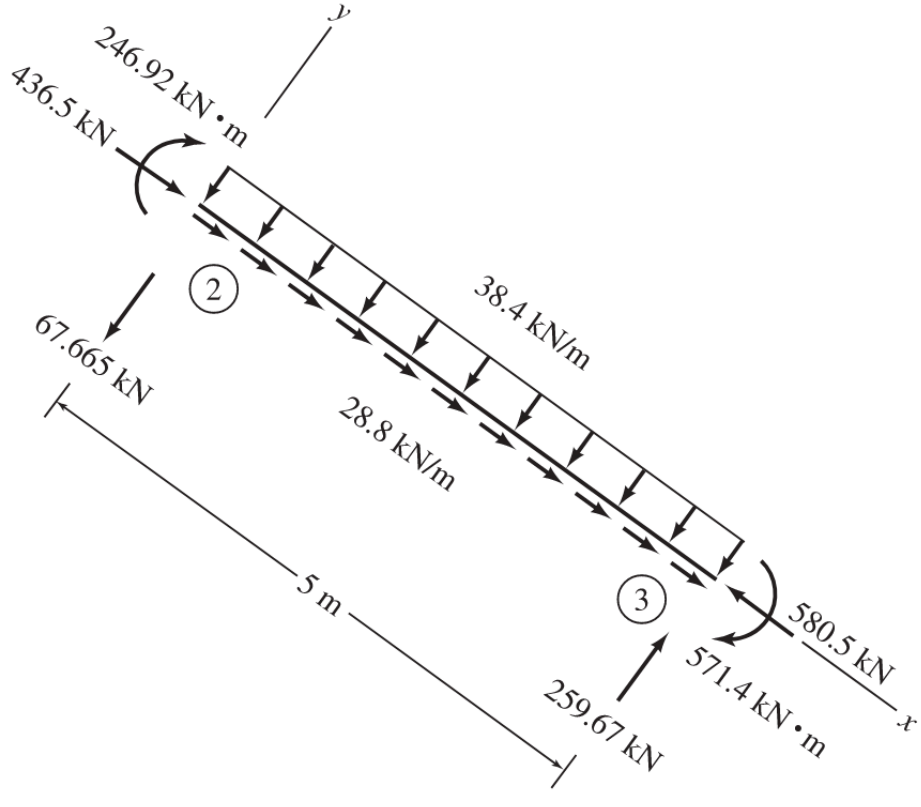
Total local FEF vector [F1..F6]:
[-72. 96. 80. -72. 96. -80.]

4. Calculate the complete local element end forces

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

In [163]:
```
Q2 = Q+Qf_total
print(Q2)
```

```
[ 436.5    -67.669 -246.929 -580.5    259.669 -571.418]
```



Part 9 - In-Class Excercise

Wrap-Up

In this lecture, we:

Next lecture: