Problem 1

Consider a thin-walled circular shaft as shown in Figure 1 with total length $L = 2.0$ m. The shaft has a thickness $h = 2\,\text{mm}$ and (average) radius $r = 30\,\text{mm}$. It is made of steel with shear modulus $G = 70\,\text{GPa}$. The right end of the shaft is clamped and the left end is loaded by a torque with magnitude $T_0 = 300\,\text{Nm}$ as shown in Figure 1.

The strong form for this problem is given as: find φ such that

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}x}\left(G\,I_\mathrm{p}\,\dfrac{\mathrm{d}\varphi(x)}{\mathrm{d}x}\right) = 0 \\ T(0) = -T_0 \\ \varphi(L) = 0 \end{cases}$$

where I_p is the polar moment of area, which for a thin-walled circular cross-section is $I_\mathrm{p} = 2\pi\,r^3\,h$ and $\varphi(x)$ is the rotation angle of the cross-section at coordinate x. In addition, the sectional torque is given through the constitutive relationship $T(x) = G\,I_\mathrm{p}\,\dfrac{\mathrm{d}\varphi(x)}{\mathrm{d}x}$.

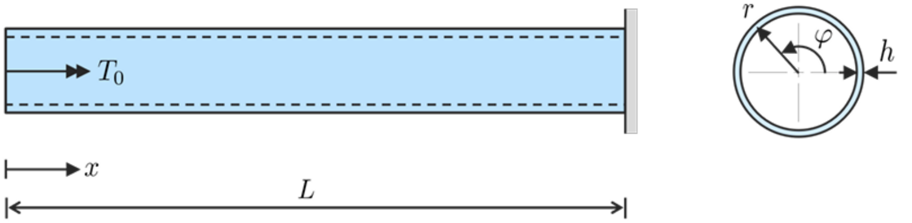


Figure 1: Clamped shaft subjected to a torque T_0 at the free end.

Tasks:

a) From the strong form, derive the weak form. **(1p)**

b) From the weak form, derive the global finite element formulation $\mathbf{K}\,\mathbf{a} = \mathbf{f}$. Then for the case of a linear approximation of the rotation, $\varphi(x)$, derive the element stiffness matrix $\mathbf{K}^e$ for an element with nodes x_i and x_{i+1}. **(1p)**

c) Consider the shaft discretized into two equally long linear elements. Determine the explicit system of equations $\mathbf{K}\,\mathbf{a} = \mathbf{f}$ and determine the rotation angle at $x = 0$. **(2p)**

d) Consider a modified problem where the fixed support is replaced with a rotational spring with stiffness $s = 3 \times 10^4\,\text{Nm}$. This gives a new boundary condition at L (of a mixed/Robin type): $T(L) = -s\,\varphi(L)$. For this modified problem, determine the new system of equations (you do not have to determine the rotation as in task c. **(2p)**

a) Diff. eq. $(GI_p\,\varphi'(x))' = 0$

Multiply with test function $v(x)$ and integrate between 0 and L $\Rightarrow$ $\int_0^L v(GI_p\varphi')'dx = 0 \Rightarrow \{\text{I.B.P.}\} \Rightarrow$

$$-\int_0^L v' GI_p \varphi' dx + \left[v GI_p \varphi'\right]_0^L = 0 \Leftrightarrow$$

$$\int_0^L v' GI_p \varphi' dx = v(L)\underbrace{GI_p\varphi'(L)}_{\substack{=T(L)\\ \text{unknown}}} - v(0)\underbrace{GI_p\varphi'(0)}_{=T(0)=-T_0} = v(L)T(L) + v(0)T_0$$

$\Rightarrow$ weak form:

Find $\varphi(x)$ such that

$$\begin{cases} \int_0^L v' GI_p \varphi' dx = v(L)T(L) + v(0)T_0 \\ \varphi(L) = 0 \end{cases}$$

for arbitrary $v(x)$

b) Insert FE approximation $\varphi(x) \approx \varphi_h(x) = \sum_{i=1}^n N_i a_i = [N_1, \dots, N_n]\begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} = \mathbb{N}\mathbb{a}$

$\Rightarrow \varphi'(x) \approx \varphi'_h(x) = \mathbb{N}'\mathbb{a} = \mathbb{B}\mathbb{a}$ with $\mathbb{B} = \frac{d}{dx}\mathbb{N}$

Approximate test function in the same way (Galerkin method)

$\Rightarrow$ $v(x) \approx v_h(x) = \sum_{i=1}^n N_i c_i = [N_1, \dots, N_n]\begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = \mathbb{N}\mathbb{c}$

where c_i are arbitrary constants

$\Rightarrow$ $v'(x) \approx v'_h(x) = \mathbb{N}'\mathbb{c} = \mathbb{B}\mathbb{c}$

Insert into weak form and using $\varphi' = (\varphi')^T$ etc. $\Rightarrow$

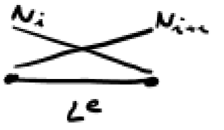
$$\mathbb{c}^T\int_0^L \mathbb{B}^T GI_p \mathbb{B}\, dx\, \mathbb{a} = \mathbb{c}^T \mathbb{N}^T(L) T(L) + \mathbb{c}^T \mathbb{N}^T(0) T_0 \Rightarrow$$

$\mathbb{c}^T\left[\int_0^L \dots dx\,\mathbb{a} - \mathbb{N}^T(L)T(L) - \mathbb{N}^T(0)T_0\right] = 0$ which should hold for arbitrary $\mathbb{c}$ $\Rightarrow$

$$\underbrace{\int_0^L \mathbb{B}^T GI_p \mathbb{B}\, dx}_{\mathbb{K}}\, \mathbb{a} = \underbrace{\mathbb{N}^T(L)T(L) + \mathbb{N}^T(0)T_0}_{\mathbb{f}_b}$$

$\Rightarrow$ FE form $\begin{cases} \mathbb{K}\mathbb{a} = \mathbb{f}_b \\ \varphi(L) = 0 \end{cases}$

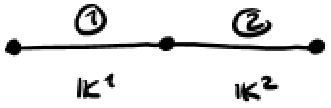
Element stiffness matrix for linear element



$$\Rightarrow \mathbb{B}^e = [N_1^{e\prime} \; N_2^{e\prime}] = \left[-\frac{1}{L^e} \;\; \frac{1}{L^e}\right]$$

$$\mathbb{K}^e = \int_{x_i}^{x_{i+1}} (\mathbb{B}^e)^T GI_p \mathbb{B}^e dx = \int_{x_i}^{x_{i+1}} \begin{bmatrix} -1/L^e \\ 1/L^e \end{bmatrix} GI_p \left[-\frac{1}{L^e} \;\; \frac{1}{L^e}\right] dx = \frac{GI_p}{(L^e)^2}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\int_{x_i}^{x_{i+1}} dx = \frac{GI_p}{L^e}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

c)



same properties in el 1 and 2 $\Rightarrow \mathbb{K}^1 = \mathbb{K}^2 = \frac{GI_p}{L^e}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$

$$\Rightarrow \mathbb{K} = \frac{GI_p}{L^e}\begin{bmatrix} 1 & -1 & 0 \\ -1 & 1+1 & -1 \\ 0 & -1 & 1 \end{bmatrix} = \left\{\begin{array}{c} G = 70\,GPa \\ I_p = 2\pi r^3 h \\ r = 30\,mm \\ h = 2\,m \\ L^e = L/2 \\ L = 2\,m \end{array}\right\} = 23750\begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}$$

$$\mathbb{f} = \mathbb{f}_b = N^T(L)T(L) + N^T(0)T_0 = \begin{bmatrix} 0 \\ 0 \\ T(L) \end{bmatrix} + \begin{bmatrix} T_0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} T_0 \\ 0 \\ T(L) \end{bmatrix} = \begin{bmatrix} 300 \\ 0 \\ T(L) \end{bmatrix}$$

$\Rightarrow$ sys. of. eqns. $23750\begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \\ 0 \end{bmatrix} = \begin{bmatrix} 300 \\ 0 \\ T(L) \end{bmatrix}$ $\Rightarrow$ solve in Python $\Rightarrow$

$\mathbb{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 0.025 \\ 0.013 \\ 0 \end{bmatrix}$ $\Rightarrow$ Rotation angle at $x=0$ is $\varphi(0) = a_1 = 0.025$ radians

d) Brd.c. is changed from $\varphi(L) = 0$ to $T(L) = -s\varphi(L)$ $\Rightarrow$

$$\int_0^L v'(x)\, GI_p\, \varphi'(x)\, dx = v(L)(-s\varphi(L)) + v(0)T_0$$

Insert FE approximations $\Rightarrow$

$$\mathbb{c}^T \int_0^L \mathbb{B}^T GI_p \mathbb{B}\, dx\, \mathbb{a} + \mathbb{c}^T \underbrace{N^T(L)\, s\, N(L)}_{\mathbb{K}_{spring}}\mathbb{a} = \mathbb{c}^T N^T(0) T_0 \Rightarrow$$

$$(\mathbb{K} + \mathbb{K}_{spring})\mathbb{a} = \mathbb{f}_b \quad \text{with} \quad \mathbb{K}_{spring} = N^T(L)\, s\, N(L)$$

2 el. discretization $\Rightarrow \mathbb{K}_{spring} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} s \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & s \end{bmatrix}$ $\Rightarrow$ new sys of. eqns.

$$\begin{bmatrix} 23750 & -23750 & 0 \\ -23750 & 47500 & -23750 \\ 0 & -23750 & 53750 \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 300 \\ 0 \\ 0 \end{bmatrix}$$

since a_3 is now free

Problem 2: Heat equation

In this problem, we shall solve the heat equation,

$$\boldsymbol{\nabla}^{\mathrm{T}}\boldsymbol{q} = h \text{ in } \Omega_{\mathrm{F}} \tag{2.1}$$

on the simple rectangular domain, Ω_{F}, below.

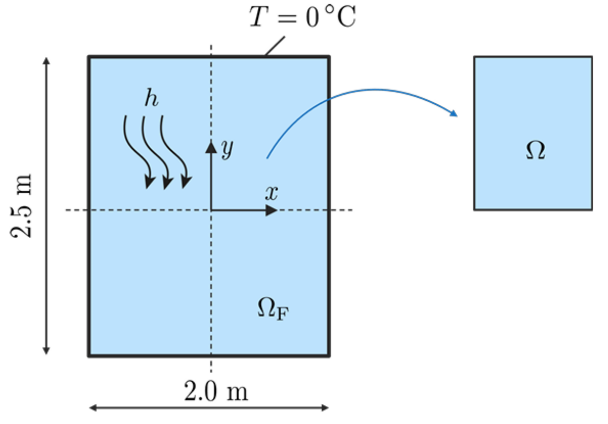


Figure 2: Full domain of Problem 2 with loads and boundary conditions. The right image shows the reduced domain Ω which should be analyzed.

The temperature, $T = 0\,°\mathrm{C}$, is known around the entire boundary. The material is isotropic with heat conductivity, $k = 1\,\mathrm{W}/(\mathrm{m}°\mathrm{C})$, such that the heat flux vector can be written $\boldsymbol{q} = -k\,\boldsymbol{\nabla}T$. The thickness can be set to $1\,\mathrm{m}$ and a constant volumetric heat source, $h = 1\,\mathrm{W}/\mathrm{m}^3$ is acting on the entire domain. Due to symmetry, we can (and will) reduce the computational costs by only analyzing the upper right quarter ($x \geq 0$ and $y \geq 0$). This domain will be denoted Ω.

To solve the problem, you are given a mesh `mesh_data.mat` of Ω with linear triangle elements consisting of the following parts,

- `Coord`: The coordinates of each node `[num_nodes,2]`
- `Dofs`: The degree of freedom for each node `[num_nodes,1]`
- `Edof`: The element degrees of freedom matrix `[num_elements,4]`
- `Ex` and `Ey`: The element x and y coordinates, `[num_elements,3]`
- The degrees of freedom for the nodes located at

- The bottom of Ω: `bottom_dofs`
- The right side of Ω: `right_dofs`
- The top of Ω :`top_dofs`
- The left side of Ω: `left_dofs`

In addition, you also get the solution, `solution.mat`, stored as `solution_vector`. This can be used to validate your solution as well as use for the postprocessing tasks if the FE solution is not found.

Hint: You can use the code provided in the file `Problem_2.py` to load these data structures.

Tasks:

a) Draw the domain Ω and state the complete strong form for the problem on Ω including the boundary conditions. **(0.5p)**

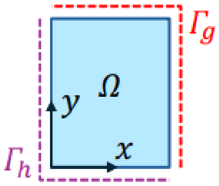
b) Derive the weak form of the heat equation. **(0.5p)**

c) Derive the global FE form of the heat equation for the problem at hand. **(0.5p)**

d) Use CALFEM to solve the FE problem and note that you can use the CALFEM function `flw2te` to calculate the element contributions. **(2.5p)**

e) Calculate the temperature at the point, $\boldsymbol{x} = [0.650\,\mathrm{m}, 0.375\,\mathrm{m}]^{\mathrm{T}}$, located in element number 18. **(1.0p)**

f) Explain how the code in task b should be modified (what must be added and what can be removed) if the Dirichlet boundary conditions are replaced by Robin (convection) boundary conditions. How would the maximum temperature change (assuming the surrounding temperature is the same temperature as previously prescribed by the Dirichlet conditions)? **(1.0p)** a)



$\nabla^{\mathrm{T}} q = h$ in Ω
$T = 0°\mathrm{C}$ on Γ_g
$q_n = 0$ on Γ_h (symmetry condition)

b) We multiply the Partial Differential Equation (PDE) by an arbitrary scalar test function, $v(\mathbf{x})$, and integrate over the domain, Ω, resulting in

$$\int_\Omega v\ \nabla^{\mathrm{T}}\mathbf{q}\ \mathrm{d}\Omega = \int_\Omega v\ h\ \mathrm{d}\Omega \tag{2.1}$$

Applying Green-Gauss theorem to the left hand side, results in

$$\int_\Gamma v\ \mathbf{n}^{\mathrm{T}}\mathbf{q}\ \mathrm{d}\Gamma - \int_\Omega [\nabla v]^{\mathrm{T}}\ \mathbf{q}\ \mathrm{d}\Omega = \int_\Omega v\ h\ \mathrm{d}\Omega \tag{2.2}$$

Finally, we split the boundary terms into the Dirichlet (Γ_g) and Neumann (Γ_h) parts, insert the constitutive law, $\mathbf{q} = -k\nabla T$, and re-arrange to get

$$\int_\Omega [\nabla v]^{\mathrm{T}}\ [k\ \nabla T]\ \mathrm{d}\Omega = \int_\Omega v\ h\ \mathrm{d}\Omega - \underbrace{\int_{\Gamma_h} v\ q_n\ \mathrm{d}\Gamma}_{=0} - \int_{\Gamma_g} v\ \mathbf{n}^{\mathrm{T}}\mathbf{q}\ \mathrm{d}\Gamma \tag{2.3}$$

Still subject to the constraint, $T(\mathbf{x}) = 0\,°\mathrm{C}$ for x on Γ_g.

c)

Insert FE approximation $T(x) \approx T_h(x) = \sum_{i=1}^{n} N_i a_i = [N_1, \dots, N_n]\begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} = \mathbb{N}\mathbb{a}$

$\Rightarrow \nabla T(x) \approx \nabla T_h(x) = \nabla(\mathbb{N}\mathbb{a}) = \mathbb{B}\mathbb{a}$ with $\mathbb{B} = \nabla \mathbb{N}$

Approximate test function in the same way (Galerkin method)

$\Rightarrow \quad v(x) \approx v_h(x) = \sum_{i=1}^{n} N_i c_i = [N_1, \dots, N_n]\begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = \mathbb{N}\mathbb{c}$

where c_i are arbitrary constants

$\Rightarrow \nabla v(x) \approx \nabla v_h(x) = \nabla(\mathbb{N}\mathbb{c}) = \mathbb{B}\mathbb{c}$

Insert FE approximations into (2.3) $\Rightarrow$

$$\mathbb{c}^T \underbrace{\int_\Omega \mathbb{B}^T k \mathbb{B}\, d\Omega}_{\mathbb{K}}\, \mathbb{a} = \mathbb{c}^T \underbrace{\int_\Omega \mathbb{N}^T h\, d\Omega}_{\mathbb{f}_\ell} + \mathbb{c}^T \underbrace{\int_{\Gamma_g} -\mathbb{N}^T q_n\, d\Gamma}_{\mathbb{f}_b} \Leftrightarrow \mathbb{c}^T\left[\mathbb{K}\mathbb{a} - \mathbb{f}_\ell - \mathbb{f}_b\right] = 0$$

which should hold for arbitrary $\mathbb{c}$ $\Rightarrow \begin{cases} \mathbb{K}\mathbb{a} = \mathbb{f}_\ell + \mathbb{f}_b \\ T = 0 \text{ on } \Gamma_g \end{cases}$

d)

```python
k = 1.0 # Heat conduction
eq = 1.0 # Heat supply
ep = [k]
D = np.eye(2)
# Assemble system of equations using the Calfem function flw2te
ndofs = np.max(np.max(Edof[:,1:]))
nel = Edof.shape[0]
K = np.zeros((ndofs, ndofs))
f = np.zeros((ndofs, 1))
for el in range(nel):
    Ke, fe = cfc.flw2te(Ex[el, :], Ey[el, :], ep, D, eq)
    cfc.assem(Edof[el, 1:], K, Ke, f, fe)
# boundary conditions
bc_dofs = np.hstack([right_dofs, top_dofs])[0]
bc_vals = np.zeros_like(bc_dofs)

# solve sys of eqns
a, r = cfc.solveq(K, f, bc_dofs, bc_vals)

# Check difference between my solution and given reference solution
display(np.linalg.norm(a - a_ref)) # should be ≈ 0
```

starting code + the following (see python file for complete solution)

e) Temperature in each element is approximated as

$$T^e = N^e a^e = \begin{bmatrix} N_1^e & N_2^e & N_3^e \end{bmatrix} \begin{bmatrix} a_1^e \\ a_2^e \\ a_3^e \end{bmatrix}$$

$$\Rightarrow T^{18} = N^{18} a^{18} \Rightarrow T(0.650\,m, 0.375\,m) = N^{18}(0.650\,m, 0.375\,m)\, a^{18}$$

```python
# e) Temperature in point x which lies in element 18
el = 18
xe = Ex[el-1, :]
ye = Ey[el-1, :]
x = 0.650 # [m]
y = 0.375 # [m]
Ae = ((xe[1] * ye[2] - xe[2] * ye[1]) - (xe[0] * ye[2] - xe[2] * ye[0]) + (xe[0]
* ye[1] - xe[1] * ye[0]))/2
Ne = np.array([xe[1] * ye[2] - xe[2] * ye[1] + (ye[1] - ye[2]) * x + (xe[2] -
xe[1]) * y,
    xe[2] * ye[0] - xe[0] * ye[2] + (ye[2] - ye[0]) * x + (xe[2] - xe[0]) * y,
    xe[0] * ye[1] - xe[1] * ye[0] + (ye[0] - ye[1]) * x + (xe[1] - xe[0]) *
y])/(2 * Ae)
# display(Ae, Ne)
dofs_e = Edof[el-1, 1:]
ae = a[dofs_e-1]
T = Ne @ ae
display(dofs_e, ae, T)
```

Problem 3

Assume a mechanical FE simulation has been performed of a component with thickness $t = 0.015\,\mathrm{m}$. The material is assumed linear elastic with Young's modulus $E = 210\,\mathrm{GPa}$ and Poisson's ratio $\nu = 0.3$. The simulation used a discretization of linear isoparametric triangular elements as can be seen in Figure 3. In this problem, you will only consider a single element (among those

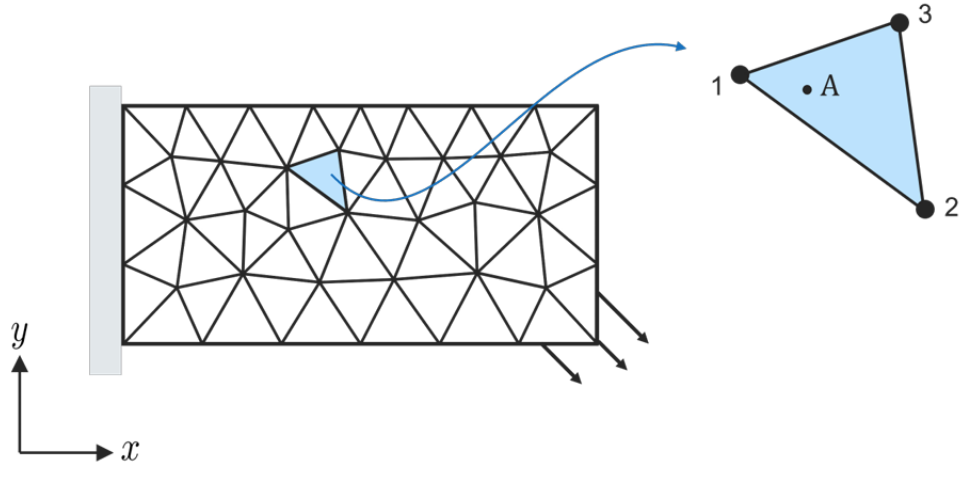


Figure 3: Illustration of component with discretization and the considered element in Problem 3.

used in the FE simulation). The coordinates for the three nodes (on the element level) are

$$\boldsymbol{x}_1^{\mathrm{e}} = [0.010\,\mathrm{m}, 0.014\,\mathrm{m}]^{\mathrm{T}}, \qquad \boldsymbol{x}_2^{\mathrm{e}} = [0.021\,\mathrm{m}, 0.009\,\mathrm{m}]^{\mathrm{T}}, \qquad \boldsymbol{x}_3^{\mathrm{e}} = [0.015\,\mathrm{m}, 0.018\,\mathrm{m}]^{\mathrm{T}}$$

Tasks:

a) Consider a point A within the element with local coordinates $[0.2, 0.1]^{\mathrm{T}}$ in the parent domain. (The point is shown in the right part of Figure 3.) For this point, calculate the corresponding global coordinates $\boldsymbol{x}_{\mathrm{A}} = [x_{\mathrm{A}}, y_{\mathrm{A}}]^{\mathrm{T}}$. **(1p)**

b) Given an FE-solution with the following nodal displacements of the element

$$\boldsymbol{u}_1^{\mathrm{e}} = [4.1, 1.0]^{\mathrm{T}} \times 10^{-5}\,\mathrm{m}, \qquad \boldsymbol{u}_2^{\mathrm{e}} = [3.5, 1.6]^{\mathrm{T}} \times 10^{-5}\,\mathrm{m}, \qquad \boldsymbol{u}_3^{\mathrm{e}} = [3.8, 1.7]^{\mathrm{T}} \times 10^{-5}\,\mathrm{m}$$

calculate for point A: the displacement vector, $\boldsymbol{u}_{\mathrm{A}}$, strains, $\boldsymbol{\epsilon}_{\mathrm{A}}$, and stresses, $\boldsymbol{\sigma}_{\mathrm{A}}$. Assume that the component is in a state of plane stress where the constitutive matrix is given as

$$\boldsymbol{D} = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix}.$$ **(3p)**

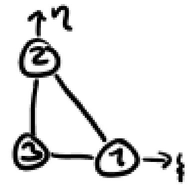
a) Global coordinate $\mathbf{x}_A = \begin{bmatrix} x_A \\ y_A \end{bmatrix} = \begin{bmatrix} \bar{N}^e(\xi_A, \eta_A)\, \hat{x} \\ \bar{N}^e(\xi_A, \eta_A)\, \hat{y} \end{bmatrix}$

See Python code for most computations

with $\xi_A = 0.2$ and $\eta_A = 0.1 \Rightarrow \bar{N}^e = [0.2 \;\; 0.1 \;\; 0.7]$

$$\Rightarrow x_A = [0.2,\, 0.1,\, 0.7] \underbrace{\begin{bmatrix} 10 \\ 21 \\ 15 \end{bmatrix}}_{\hat{x}} = 14.6 \text{ mm}$$

$$y_A = [\qquad\qquad] \underbrace{\begin{bmatrix} 14 \\ 9 \\ 18 \end{bmatrix}}_{\hat{y}} = 16.3 \text{ mm}$$



$N_1 = \xi$

$N_2 = \eta$

$N_3 = 1 - \xi - \eta$

$$\Rightarrow \mathbf{x}_A = \begin{bmatrix} 14.6 \\ 16.3 \end{bmatrix} \text{mm}$$

note: with shape functions from the old formula sheet you will get $\mathbf{x}_A = \begin{bmatrix} 12.7 \\ 13.4 \end{bmatrix}$ mm

b) FE approximation $\mathbf{u}_A = \begin{bmatrix} u_x^e \\ u_y^e \end{bmatrix} = \mathbf{N}^e \mathbf{a}^e = \begin{bmatrix} N_1^e & 0 & N_2^e & 0 & N_3^e & 0 \\ 0 & N_1^e & 0 & N_2^e & 0 & N_3^e \end{bmatrix} \begin{bmatrix} a_{1x} \\ a_{1y} \\ a_{2x} \\ a_{2y} \\ a_{3x} \\ a_{3y} \end{bmatrix}$

$$= \begin{bmatrix} 0.2 & 0 & 0.1 & 0 & 0.7 & 0 \\ 0 & 0.2 & 0 & 0.1 & 0 & 0.7 \end{bmatrix} \underbrace{\begin{bmatrix} 4.1 \\ 1.0 \\ 3.5 \\ 1.6 \\ 3.8 \\ 1.7 \end{bmatrix}}_{\mathbf{a}^e} \cdot 10^{-5} = \begin{bmatrix} 3.8 \\ 1.6 \end{bmatrix} \cdot 10^{-5} \text{ m}$$

note: with shape functions from the old formula sheet you will get $\mathbf{u}_A = \begin{bmatrix} 4.0 \\ 1.2 \end{bmatrix} \cdot 10^{-5}$ m

Strains:

Need the derivatives of the shape functions w.r.t. x and y in point A.

$$\begin{bmatrix} \dfrac{\partial \bar{N}^e}{\partial x} \\ \dfrac{\partial \bar{N}^e}{\partial y} \end{bmatrix} = \mathbf{J}^{-T} \begin{bmatrix} \dfrac{\partial \bar{N}^e}{\partial \xi} \\ \dfrac{\partial \bar{N}^e}{\partial \eta} \end{bmatrix} \qquad \text{with} \qquad \mathbf{J} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial x}{\partial \eta} \\ \dfrac{\partial y}{\partial \xi} & \dfrac{\partial y}{\partial \eta} \end{bmatrix}$$

$$\frac{\partial \bar{N}^e}{\partial \xi} = [1, 0, -1] \; ; \quad \frac{\partial \bar{N}^e}{\partial \eta} = [0, 1, -1]$$

$$\frac{\partial x}{\partial \xi} = \frac{\partial \bar{N}^e}{\partial \xi} \hat{x} \; ; \quad \frac{\partial x}{\partial \eta} = \frac{\partial \bar{N}^e}{\partial \eta} \hat{x} \; ; \quad \frac{\partial y}{\partial \xi} = \frac{\partial \bar{N}^e}{\partial \xi} \hat{y} \; ; \quad \frac{\partial y}{\partial \eta} = \frac{\partial \bar{N}^e}{\partial \eta} \hat{y}$$

with $\dfrac{\partial \bar{N}^e}{\partial x} = \left[\dfrac{\partial N_1}{\partial x}, \dfrac{\partial N_2}{\partial x}, \dfrac{\partial N_3}{\partial x} \right]$ and $\dfrac{\partial \bar{N}^e}{\partial y} = \left[\dfrac{\partial N_1}{\partial y}, \dfrac{\partial N_2}{\partial y}, \dfrac{\partial N_3}{\partial y} \right]$

evaluated in point A, form the $\mathbf{B}$-matrix (3×6)

$$\mathbf{B}_A = \begin{bmatrix} \dfrac{\partial N_1}{\partial x} & 0 & \dfrac{\partial N_2}{\partial x} & 0 & \dfrac{\partial N_3}{\partial x} & 0 \\ 0 & \dfrac{\partial N_1}{\partial y} & 0 & \dfrac{\partial N_2}{\partial y} & 0 & \dfrac{\partial N_3}{\partial y} \\ \dfrac{\partial N_1}{\partial y} & \dfrac{\partial N_1}{\partial x} & \dfrac{\partial N_2}{\partial y} & \dfrac{\partial N_2}{\partial x} & \dfrac{\partial N_3}{\partial y} & \dfrac{\partial N_3}{\partial x} \end{bmatrix}_A = \begin{bmatrix} -130 & 0 & 58 & 0 & 72 & 0 \\ 0 & -87 & 0 & -72 & 0 & 159 \\ -87 & -130 & -72 & 58 & 159 & 72 \end{bmatrix}$$

Strains: $\quad \varepsilon_A = \mathbb{B}^e \mathbb{a}^e = \ldots = \begin{bmatrix} -5.7 \\ 6.8 \\ 8.1 \end{bmatrix} \cdot 10^{-4}$

Stresses: $\quad \sigma_A = \mathbb{D}\,\varepsilon_A = \ldots = \begin{bmatrix} -83 \\ 118 \\ 66 \end{bmatrix}$ MPa

c) $\quad \mathbb{f}_e = \int_{A^e} [\mathbb{N}^e]^T \mathbb{b}\, t\, dA \Rightarrow \left\{ \begin{array}{l} \text{numerical integration} \\ \text{with one point} \Rightarrow \\ \xi_{ip} = \begin{bmatrix} 1/3 \\ 1/3 \end{bmatrix} \end{array} \right\} \Rightarrow$

$$\mathbb{f}_e \approx [\mathbb{N}^e(\xi_{ip})]^T \mathbb{b}(x_{ip})\, t \det \mathbb{J}(\xi_{ip})\, W$$

$$\begin{aligned} x_{ip} &= \bar{N}^e(1/3, 1/3)\,\hat{x} = 0.015\,\text{m} \\ y_{ip} &= \bar{N}^e(1/3, 1/3)\,\hat{y} = 0.014\,\text{m} \end{aligned} \Rightarrow \left\{ \mathbb{b} = \begin{bmatrix} 2x \\ 3xy \end{bmatrix} \right\} \Rightarrow \mathbb{b}(x_{ip}) = \begin{bmatrix} 30.7 \\ 0.6 \end{bmatrix} \text{kN/m}^3$$

- $\mathbb{N}^e(\xi_{ip}) = \frac{1}{3} \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$

- $\det \mathbb{J}(\xi_{ip}) = \ldots = 6.9 \cdot 10^{-5}\,\text{m}^2$

$$\Rightarrow \quad \mathbb{f}_e = \left\{ \begin{array}{l} t = 15 \cdot 10^{-3}\,\text{m} \\ W = 1/2 \end{array} \right\} = \ldots = \begin{bmatrix} 0.0053 \\ 0.0001 \\ 0.0053 \\ 0.0001 \\ 0.0053 \\ 0.0001 \end{bmatrix} \text{N}$$