Algorithm for problems involving flat shells

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

Algorithm for problems involving flat shells

1) Input data:

1.1 Material properties and thickness for each <u>different</u> shell (m):

- Young's modulus: $E^{(m)}$
- Poisson ratio: $\nu^{(m)}$
- Density: $\rho^{(m)}$
- Thickness: $h^{(m)}$

1.2 Mesh (discretization) data:

- Nodal coordinates matrix:

$$[\mathbf{X}] = \left.\begin{bmatrix} & \vdots & \\ \hat{x}^{(n)} & \hat{y}^{(n)} & \hat{z}^{(n)} \\ & \vdots & \end{bmatrix}\right\} \text{Number of nodes } N$$

$\hat{x}^{(n)}$: x-coordinate of node n
$\hat{y}^{(n)}$: y-coordinate of node n
$\hat{z}^{(n)}$: z-coordinate of node n

- Nodal connectivities matrix:

$$[\mathbf{T}_\mathrm{n}] = \left.\begin{bmatrix} & & \vdots & \\ n_1^{(e)} & n_2^{(e)} & n_3^{(e)} & n_4^{(e)} \\ & & \vdots & \end{bmatrix}\right\} \text{Number of elements } N_e$$

$n_i^{(e)}$: global node # assigned to i-th node in element (e)

Algorithm for problems involving flat shells

- Material connectivites matrix:

$$[\mathbf{T}_{\mathrm{m}}] = \begin{bmatrix} \vdots \\ m^{(e)} \\ \vdots \end{bmatrix} \Big\} \text{ Number of elements } N_e$$

$m^{(e)}$: material index # assigned to element (e)

1.3 Boundary conditions:

- Fix nodes and DOFs matrix:

$$[\mathbf{U}_{\mathrm{p}}] = \begin{bmatrix} & \vdots & \\ u^{(p)} & n^{(p)} & j^{(p)} \\ & \vdots & \end{bmatrix} \Big\} \text{ Number of prescribed DOFs}$$

$u^{(p)}$: value of prescribed displ./rot. (p)
$n^{(p)}$: global node # assigned to (p)
$j^{(p)}$: degree of freedom assigned to (p)

1.4 External forces:

- Non-null point forces matrix (N):

$$[\mathbf{F}_{\mathrm{e}}] = \begin{bmatrix} & \vdots & \\ f^{(q)} & n^{(q)} & j^{(q)} \\ & \vdots & \end{bmatrix} \Big\} \text{ Number of point forces}$$

$f^{(q)}$: value of point force/moment (q)
$n^{(q)}$: global node # assigned to (q)
$j^{(q)}$: degree of freedom assigned to (q)

Algorithm for problems involving flat shells

- Distributed loads matrix (N/m²):

$$[\mathbf{P}_\mathrm{e}] = \left[\begin{matrix} & \vdots & \\ p^{(r)} & n^{(r)} & j^{(r)} \\ & \vdots & \end{matrix}\right] \Bigg\} \text{Number of DOFs with distributed loads}$$

$p^{(r)}$: value of distr. force/moment (r)
$n^{(r)}$: global node # assigned to (r)
$j^{(r)}$: degree of freedom assigned to (r)

- Body forces matrix (N/kg):

$$[\mathbf{B}_\mathrm{e}] = \left[\begin{matrix} & \vdots & \\ b^{(s)} & n^{(s)} & j^{(s)} \\ & \vdots & \end{matrix}\right] \Bigg\} \text{Number of DOFs with body forces}$$

$b^{(s)}$: value of body force/moment (s)
$n^{(s)}$: global node # assigned to (s)
$j^{(s)}$: degree of freedom assigned to (s)

Note: Recall that degrees of freedom indices # represent:

$j = 1$: displacement/force in x-direction

$j = 2$: displacement/force in y-direction

$j = 3$: displacement/force in z-direction

$j = 4$: rotation/moment about x-direction

$j = 5$: rotation/moment about y-direction

$j = 6$: rotation/moment about z-direction

Algorithm for problems involving flat shells

2) Assembly of global matrices:

2.1 Initialization:

$N_{\text{dof}} = 6N$ (total number of degrees of freedom)

$[\mathbf{K}] = [\mathbf{0}]_{N_{\text{dof}} \times N_{\text{dof}}}$

$[\mathbf{M}] = [\mathbf{0}]_{N_{\text{dof}} \times N_{\text{dof}}}$

2.2 Assembly process:

For each element e:

a) Compute rotation matrix:

Vector product!!

$$\{\boldsymbol{S}\} = \left(\{\mathbf{X}(\mathbf{T}_{\text{n}}(e,3),:)\}^{\text{T}} - \{\mathbf{X}(\mathbf{T}_{\text{n}}(e,1),:)\}^{\text{T}}\right) \times \left(\{\mathbf{X}(\mathbf{T}_{\text{n}}(e,4),:)\}^{\text{T}} - \{\mathbf{X}(\mathbf{T}_{\text{n}}(e,2),:)\}^{\text{T}}\right)/2$$

$\{\hat{\boldsymbol{k}}'\} = \{\boldsymbol{S}\}/\|\{\boldsymbol{S}\}\|$ (≡ normal vector of the flat shell element)

$$\{\boldsymbol{d}\} = \left(\{\mathbf{X}(\mathbf{T}_{\text{n}}(e,2),:)\}^{\text{T}} + \{\mathbf{X}(\mathbf{T}_{\text{n}}(e,3),:)\}^{\text{T}} - \{\mathbf{X}(\mathbf{T}_{\text{n}}(e,4),:)\}^{\text{T}} - \{\mathbf{X}(\mathbf{T}_{\text{n}}(e,1),:)\}^{\text{T}}\right)/2$$

$$\{\hat{\boldsymbol{\imath}}'\} = \{\boldsymbol{d}\}/\|\{\boldsymbol{d}\}\| \,; \qquad \{\hat{\boldsymbol{\jmath}}'\} = \{\hat{\boldsymbol{k}}'\} \times \{\hat{\boldsymbol{\imath}}'\}$$

$$[\mathbf{R}'] = \begin{bmatrix} \{\hat{\boldsymbol{\imath}}'\} & \{\hat{\boldsymbol{\jmath}}'\} & \{\hat{\boldsymbol{k}}'\} & [\mathbf{0}]_{3\times 2} \\ & [\mathbf{0}]_{3\times 3} & & \{\hat{\boldsymbol{\imath}}'\} \quad \{\hat{\boldsymbol{\jmath}}'\} \end{bmatrix}^{\text{T}}$$

$$\mathbf{R}(:,:,e) = \begin{bmatrix} [\mathbf{R}'] & [\mathbf{0}]_{5\times 6} & [\mathbf{0}]_{5\times 6} & [\mathbf{0}]_{5\times 6} \\ [\mathbf{0}]_{5\times 6} & [\mathbf{R}'] & [\mathbf{0}]_{5\times 6} & [\mathbf{0}]_{5\times 6} \\ [\mathbf{0}]_{5\times 6} & [\mathbf{0}]_{5\times 6} & [\mathbf{R}'] & [\mathbf{0}]_{5\times 6} \\ [\mathbf{0}]_{5\times 6} & [\mathbf{0}]_{5\times 6} & [\mathbf{0}]_{5\times 6} & [\mathbf{R}'] \end{bmatrix}$$

Algorithm for problems involving flat shells

b) Get nodal coefficients for the shape functions:

$$\{\boldsymbol{a}\} = \{-1, 1, 1, -1\}$$

$$\{\boldsymbol{b}\} = \{-1, -1, 1, 1\}$$

c) Compute element matrices:

c1) 1 Gauss point quadrature matrices:

$$\{\boldsymbol{N}_1\} = \{1, 1, 1, 1\}^{\mathrm{T}}/4$$

$$\{\boldsymbol{N}_{1,\xi}\} = \{\boldsymbol{a}\}/4$$

$$\{\boldsymbol{N}_{1,\eta}\} = \{\boldsymbol{b}\}/4$$

$$[\boldsymbol{\mathcal{J}}_1] = [\mathbf{0}]_{2\times 2}$$

For each node i (from 1 to 4) in the element:

$$[\boldsymbol{\mathcal{J}}_1] = [\boldsymbol{\mathcal{J}}_1] + \begin{Bmatrix} \boldsymbol{N}_{1,\xi}(i) \\ \boldsymbol{N}_{1,\eta}(i) \end{Bmatrix} \{\mathbf{X}(\mathbf{T}_\mathrm{n}(e,i),:)\}[\hat{\boldsymbol{\imath}}' \quad \hat{\boldsymbol{\jmath}}']$$

End loop over nodes

$$[\boldsymbol{N}_{1,\boldsymbol{x}'}] = [\boldsymbol{\mathcal{J}}_1]^{-1} \begin{bmatrix} \boldsymbol{N}_{1,\xi} \\ \boldsymbol{N}_{1,\eta} \end{bmatrix}$$

$S_1 = 4 \det[\boldsymbol{\mathcal{J}}_1]$ (≡ area associated to Gauss point)

Algorithm for problems involving flat shells

c1.1) Shear component of stiffness matrix:

For each node i (from 1 to 4) in the element:

$$\mathbf{B}_{\mathrm{s}}'^{(i)}(:,:,i) = \begin{bmatrix} 0 & 0 & \boldsymbol{N}_{1,\boldsymbol{x}'}(1,i) & 0 & \boldsymbol{N}_1(i) \\ 0 & 0 & \boldsymbol{N}_{1,\boldsymbol{x}'}(2,i) & -\boldsymbol{N}_1(i) & 0 \end{bmatrix}$$

End loop over nodes

$$\bar{\mathbf{C}}_{\mathrm{s}}' = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} 5h^{(\mathbf{T}_{\mathrm{m}}(e))} E^{(\mathbf{T}_{\mathrm{m}}(e))} \Big/ \left(12\left(1+\nu^{(\mathbf{T}_{\mathrm{m}}(e))}\right)\right)$$

$$\mathbf{B}_{\mathrm{s}}'(:,:,e) = \left[\mathbf{B}_{\mathrm{s}}'^{(i)}(:,:,1), \mathbf{B}_{\mathrm{s}}'^{(i)}(:,:,2), \mathbf{B}_{\mathrm{s}}'^{(i)}(:,:,3), \mathbf{B}_{\mathrm{s}}'^{(i)}(:,:,4)\right]$$

$$\mathbf{K}_{\mathrm{s}}(:,:,e) = S_1[\mathbf{R}(:,:,e)]^{\mathrm{T}}[\mathbf{B}_{\mathrm{s}}'(:,:,e)]^{\mathrm{T}}[\bar{\mathbf{C}}_{\mathrm{s}}'][\mathbf{B}_{\mathrm{s}}'(:,:,e)][\mathbf{R}(:,:,e)]$$

c1.2) Membrane **transverse** component of stiffness matrix:

For each node i (from 1 to 4) in the element:

$$\mathbf{B}_{\mathrm{m_t}}'^{(i)}(:,:,i) = \begin{bmatrix} \boldsymbol{N}_{1,\boldsymbol{x}'}(2,i) & \boldsymbol{N}_{1,\boldsymbol{x}'}(1,i) & 0 & 0 & 0 \end{bmatrix}$$

End loop over nodes

$$\bar{\mathbf{C}}_{\mathrm{m_t}}' = h^{(\mathbf{T}_{\mathrm{m}}(e))} E^{(\mathbf{T}_{\mathrm{m}}(e))} \Big/ \left(2\left(1+\nu^{(\mathbf{T}_{\mathrm{m}}(e))}\right)\right)$$

$$\mathbf{B}_{\mathrm{m_t}}'(:,:,e) = \left[\mathbf{B}_{\mathrm{m_t}}'^{(i)}(:,:,1), \mathbf{B}_{\mathrm{m_t}}'^{(i)}(:,:,2), \mathbf{B}_{\mathrm{m_t}}'^{(i)}(:,:,3), \mathbf{B}_{\mathrm{m_t}}'^{(i)}(:,:,4)\right]$$

$$\mathbf{K}_{\mathrm{m}}(:,:,e) = S_1[\mathbf{R}(:,:,e)]^{\mathrm{T}}\left[\mathbf{B}_{\mathrm{m_t}}'(:,:,e)\right]^{\mathrm{T}}\left[\bar{\mathbf{C}}_{\mathrm{m_t}}'\right]\left[\mathbf{B}_{\mathrm{m_t}}'(:,:,e)\right][\mathbf{R}(:,:,e)]$$

Algorithm for problems involving flat shells

c2) 4 Gauss points quadrature matrices:

$\mathbf{K}_{\mathrm{b}}(:,:,e) = [\mathbf{0}]_{24\times 24}$

$\mathbf{M}_{\mathrm{e}}(:,:,e) = [\mathbf{0}]_{24\times 24}$

$\{\boldsymbol{\xi}_4\} = \{-1, 1, 1, -1\}/\sqrt{3}$

$\{\boldsymbol{\eta}_4\} = \{-1, -1, 1, 1\}/\sqrt{3}$

$\{\boldsymbol{w}_4\} = \{1, 1, 1, 1\}$

For each Gauss point k (from 1 to 4):

$\boldsymbol{\mathcal{J}}_4 = [\mathbf{0}]_{2\times 2}$

For each node i (from 1 to 4) in the element:

$\boldsymbol{N}_4(i) = \big(1 + \boldsymbol{a}(i)\boldsymbol{\xi}_4(k)\big)\big(1 + \boldsymbol{b}(i)\boldsymbol{\eta}_4(k)\big)/4$

$\boldsymbol{N}_{4,\xi}(1, i) = \boldsymbol{a}(i)\big(1 + \boldsymbol{b}(i)\boldsymbol{\eta}_4(k)\big)/4$

$\boldsymbol{N}_{4,\eta}(1, i) = \boldsymbol{b}(i)\big(1 + \boldsymbol{a}(i)\boldsymbol{\xi}_4(k)\big)/4$

$$\boldsymbol{\mathcal{J}}_4 = \boldsymbol{\mathcal{J}}_4 + \begin{Bmatrix} \boldsymbol{N}_{4,\xi}(i) \\ \boldsymbol{N}_{4,\eta}(i) \end{Bmatrix} \{\mathbf{X}(\mathbf{T}_{\mathrm{n}}(e, i), :)\}[\hat{\boldsymbol{i}}' \quad \hat{\boldsymbol{j}}']$$

End loop over nodes

Algorithm for problems involving flat shells

$$\boldsymbol{N}_{4,\boldsymbol{x}'} = [\boldsymbol{\mathcal{J}}_4]^{-1}\begin{bmatrix}\boldsymbol{N}_{4,\xi}\\ \boldsymbol{N}_{4,\eta}\end{bmatrix}$$

$\boldsymbol{S}_4(e,k) = \boldsymbol{w}_4(k)\cdot\det[\boldsymbol{\mathcal{J}}_4]$ (≡ area associated to Gauss point)

c2.1) Membrane **normal** component of stiffness matrix:

For each node i (from 1 to 4) in the element:

$$\mathbf{B}_{\mathrm{m_n}}^{\prime(i)}(:,:,i) = \begin{bmatrix}\boldsymbol{N}_{4,\boldsymbol{x}'}(1,i) & 0 & 0 & 0 & 0\\ 0 & \boldsymbol{N}_{4,\boldsymbol{x}'}(2,i) & 0 & 0 & 0\end{bmatrix}$$

End loop over nodes

$$\bar{\mathbf{C}}'_{\mathrm{m_n}} = \begin{bmatrix}1 & \nu^{(\mathrm{T_m}(e))}\\ \nu^{(\mathrm{T_m}(e))} & 1\end{bmatrix} h^{(\mathrm{T_m}(e))}E^{(\mathrm{T_m}(e))}\Big/\left(1-\nu^{(\mathrm{T_m}(e))^2}\right)$$

$$\mathbf{B}'_{\mathrm{m_n}}(:,:,e,k) = \left[\mathbf{B}_{\mathrm{m_n}}^{\prime(i)}(:,:,1), \mathbf{B}_{\mathrm{m_n}}^{\prime(i)}(:,:,2), \mathbf{B}_{\mathrm{m_n}}^{\prime(i)}(:,:,3), \mathbf{B}_{\mathrm{m_n}}^{\prime(i)}(:,:,4)\right]$$

$$\mathbf{K}_{\mathrm{m}}(:,:,e) = \mathbf{K}_{\mathrm{m}}(:,:,e) + \boldsymbol{S}_4(e,k)[\mathbf{R}(:,:,e)]^{\mathrm{T}}\left[\mathbf{B}'_{\mathrm{m_n}}(:,:,e,k)\right]^{\mathrm{T}}\left[\bar{\mathbf{C}}'_{\mathrm{m_n}}\right]\left[\mathbf{B}'_{\mathrm{m_n}}(:,:,e,k)\right][\mathbf{R}(:,:,e)]$$

Notice that to avoid shear locking, we have previously computed the component of the membrane stiffness matrix dealing with transverse strains (with only 1 Gauss point), so now we just need to add the normal component:

$$\left[\mathbf{K}_{\mathrm{m}}^{\prime(e,i)}\right] = \left[\mathbf{B}_{\mathrm{m_n}}^{\prime(e,i)\mathrm{T}} \quad \mathbf{B}_{\mathrm{m_t}}^{\prime(e,i)\mathrm{T}}\right]\begin{bmatrix}\mathbf{C}'_{\mathrm{m_n}} & \mathbf{0}\\ \mathbf{0} & \mathbf{C}'_{\mathrm{m_t}}\end{bmatrix}\begin{bmatrix}\mathbf{B}_{\mathrm{m_n}}^{\prime(e,i)}\\ \mathbf{B}_{\mathrm{m_t}}^{\prime(e,i)}\end{bmatrix} = \left[\mathbf{B}_{\mathrm{m_n}}^{\prime(e,i)}\right]^{\mathrm{T}}\left[\bar{\mathbf{C}}'_{\mathrm{m_t}}\right]\left[\mathbf{B}_{\mathrm{m_t}}^{\prime(e,i)}\right] + \left[\mathbf{B}_{\mathrm{m_n}}^{\prime(e,i)}\right]^{\mathrm{T}}\left[\bar{\mathbf{C}}'_{\mathrm{m_n}}\right]\left[\mathbf{B}_{\mathrm{m_n}}^{\prime(e,i)}\right]$$

Algorithm for problems involving flat shells

c2.2) Bending component of stiffness matrix:

For each node i (from 1 to 4) in the element:

$$\mathbf{B}_{\mathrm{b}}^{\prime(i)}(:,:,i) = \begin{bmatrix} 0 & 0 & 0 & 0 & \boldsymbol{N}_{4,\boldsymbol{x}'}(1,i) \\ 0 & 0 & 0 & \boldsymbol{N}_{4,\boldsymbol{x}'}(2,i) & 0 \\ 0 & 0 & 0 & -\boldsymbol{N}_{4,\boldsymbol{x}'}(1,i) & \boldsymbol{N}_{4,\boldsymbol{x}'}(2,i) \end{bmatrix}$$

End loop over nodes

$$\bar{\mathbf{C}}_{\mathrm{b}}' = \begin{bmatrix} 1 & \nu^{(\mathbf{T}_{\mathrm{m}}(e))} & 0 \\ \nu^{(\mathbf{T}_{\mathrm{m}}(e))} & 1 & 0 \\ 0 & 0 & \left(1-\nu^{(\mathbf{T}_{\mathrm{m}}(e))}\right)/2 \end{bmatrix} h^{(\mathbf{T}_{\mathrm{m}}(e))^3} E^{(\mathbf{T}_{\mathrm{m}}(e))} \Big/ \left(12\left(1-\nu^{(\mathbf{T}_{\mathrm{m}}(e))^2}\right)\right)$$

$$\mathbf{B}_{\mathrm{b}}'(:,:,e,k) = \left[\mathbf{B}_{\mathrm{b}}^{\prime(i)}(:,:,1), \mathbf{B}_{\mathrm{b}}^{\prime(i)}(:,:,2), \mathbf{B}_{\mathrm{b}}^{\prime(i)}(:,:,3), \mathbf{B}_{\mathrm{b}}^{\prime(i)}(:,:,4)\right]$$

$$\mathbf{K}_{\mathrm{b}}(:,:,e) = \mathbf{K}_{\mathrm{b}}(:,:,e) + \boldsymbol{S}_4(e,k)[\mathbf{R}(:,:,e)]^{\mathrm{T}}[\mathbf{B}_{\mathrm{b}}'(:,:,e,k)]^{\mathrm{T}}[\bar{\mathbf{C}}_{\mathrm{b}}'][\mathbf{B}_{\mathrm{b}}'(:,:,e,k)][\mathbf{R}(:,:,e)]$$

c2.3) Mass matrix:

For each node i (from 1 to 4) in the element:

$$\mathbf{N}^{(i)}(:,:,i) = \boldsymbol{N}_4(i)[\mathbf{1}]_{5\times 5} \quad ([\mathbf{1}]_{5\times 5} \equiv \text{Identity matrix of } 5 \times 5)$$

End loop over nodes

Algorithm for problems involving flat shells

$$\overline{\boldsymbol{\rho}}' = \rho^{(\mathbf{T}_{\mathrm{m}}(e))} h^{(\mathbf{T}_{\mathrm{m}}(e))} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & {h^{(\mathbf{T}_{\mathrm{m}}(e))}}^2/12 & 0 \\ 0 & 0 & 0 & 0 & {h^{(\mathbf{T}_{\mathrm{m}}(e))}}^2/12 \end{bmatrix}$$

$$\mathbf{N}(:,:,e,k) = \left[\mathbf{N}^{(i)}(:,:,1), \mathbf{N}^{(i)}(:,:,2), \mathbf{N}^{(i)}(:,:,3), \mathbf{N}^{(i)}(:,:,4)\right]$$

$$\mathbf{M}_{\mathrm{e}}(:,:,e) = \mathbf{M}_{\mathrm{e}}(:,:,e) + \boldsymbol{S}_4(e,k)[\mathbf{R}(:,:,e)]^{\mathrm{T}}[\mathbf{N}(:,:,e,k)]^{\mathrm{T}}[\overline{\boldsymbol{\rho}}'][\mathbf{N}(:,:,e,k)][\mathbf{R}(:,:,e)]$$

End loop over Gauss points

d) Assembly to global matrices:

For each degree of freedom j from 1 to 6

$I_{\mathrm{dof}}(j,1) = 6(\mathbf{T}_{\mathrm{n}}(e,1) - 1) + j$

$I_{\mathrm{dof}}(6+j,1) = 6(\mathbf{T}_{\mathrm{n}}(e,2) - 1) + j$

$I_{\mathrm{dof}}(12+j,1) = 6(\mathbf{T}_{\mathrm{n}}(e,3) - 1) + j$

$I_{\mathrm{dof}}(18+j,1) = 6(\mathbf{T}_{\mathrm{n}}(e,4) - 1) + j$

End loop over DOFs

$\mathbf{K}(I_{\mathrm{dof}}, I_{\mathrm{dof}}) = \mathbf{K}(I_{\mathrm{dof}}, I_{\mathrm{dof}}) + \mathbf{K}_{\mathrm{m}}(:,:,e) + \mathbf{K}_{\mathrm{b}}(:,:,e) + \mathbf{K}_{\mathrm{s}}(:,:,e)$

$\mathbf{M}(I_{\mathrm{dof}}, I_{\mathrm{dof}}) = \mathbf{M}(I_{\mathrm{dof}}, I_{\mathrm{dof}}) + \mathbf{M}_{\mathrm{e}}(:,:,e)$

End loop over elements

Algorithm for problems involving flat shells

3) Compute artificial rotation stiffness matrix:

3.1 Find nodal normal to set criteria for finding coplanar nodes:

$$[\mathbf{n}] = [\mathbf{0}]_{3\times N}$$

For each element e:

a) Compute normal and surface:

$$\{\boldsymbol{S}\} = \left(\{\mathbf{X}(\mathbf{T}_\mathrm{n}(e,3),:)\}^\mathrm{T} - \{\mathbf{X}(\mathbf{T}_\mathrm{n}(e,1),:)\}^\mathrm{T}\right) \times \left(\{\mathbf{X}(\mathbf{T}_\mathrm{n}(e,4),:)\}^\mathrm{T} - \{\mathbf{X}(\mathbf{T}_\mathrm{n}(e,2),:)\}^\mathrm{T}\right)/2$$

$$S(e) = \sqrt{\left(\boldsymbol{S}(1)\right)^2 + \left(\boldsymbol{S}(2)\right)^2 + \left(\boldsymbol{S}(3)\right)^2}$$

$$\hat{\boldsymbol{k}}'(:,e) = \{\boldsymbol{S}\}/S(e)$$

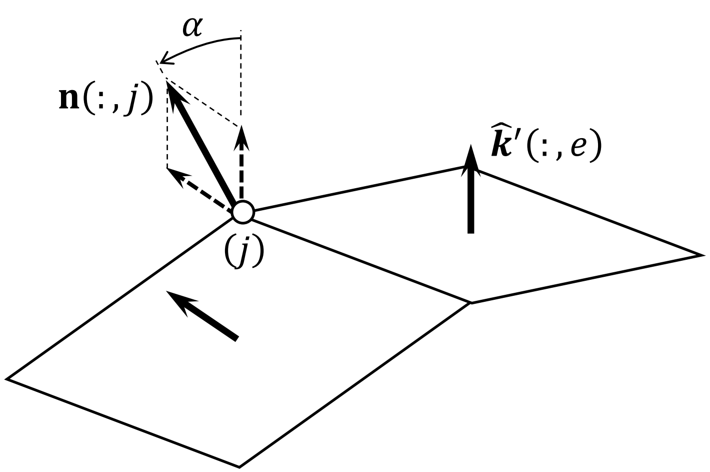
b) Assemble to get nodal normal:

For each element node i

$$\mathbf{n}\left(:,\mathbf{T}_\mathrm{n}(e,i)\right) = \mathbf{n}\left(:,\mathbf{T}_\mathrm{n}(e,i)\right) + \hat{\boldsymbol{k}}'(:,e)$$

End loop over element nodes

End loop over elements



Algorithm for problems involving flat shells

3.2 Compute artificial rotation matrix:

$[\mathbf{K}_\mathrm{r}] = [\mathbf{0}]_{N_\mathrm{dof} \times N_\mathrm{dof}}$

For each element e:

For each element node i:

a) Determine whether it is or not a coplanar node

$\alpha = \cos^{-1}\left(\mathbf{n}\left(:,\mathbf{T}_\mathrm{n}(e,i)\right) \cdot \hat{\boldsymbol{k}}'(:,e) / \left\|\mathbf{n}\left(:,\mathbf{T}_\mathrm{n}(e,i)\right)\right\|\right)$ ← Scalar product!!

If $\alpha < 5°$ (we can consider node coplanar)

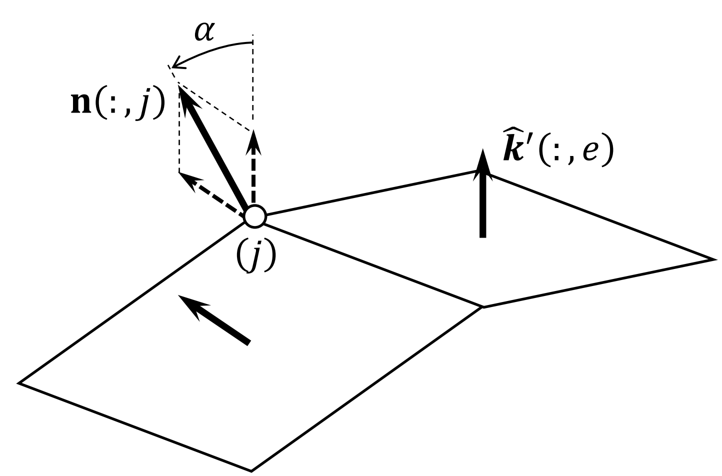
b) Evaluate artificial rotation stiffness component

$I_\mathrm{dof} = 6(\mathbf{T}_\mathrm{n}(e,i) - 1) + \{4,5,6\}^\mathrm{T}$

$\mathbf{K}_\mathrm{r}(I_\mathrm{dof}, I_\mathrm{dof}) = \mathbf{K}_\mathrm{r}(I_\mathrm{dof}, I_\mathrm{dof}) + E^{(\mathbf{T}_\mathrm{m}(e))} h^{(\mathbf{T}_\mathrm{m}(e))} S(e) \left\{\hat{\boldsymbol{k}}'(:,e)\right\}\left\{\hat{\boldsymbol{k}}'(:,e)\right\}^\mathrm{T}$

End loop over element nodes

End loop over elements



3.3 Update stiffness matrix:

$[\overline{\mathbf{K}}] = [\mathbf{K}] + [\mathbf{K}_\mathrm{r}]$

Algorithm for problems involving flat shells

3.2 Compute artificial rotation matrix:

$[\mathbf{K}_{\mathrm{r}}] = [\mathbf{0}]_{N_{\mathrm{dof}} \times N_{\mathrm{dof}}}$

For each element e:

For each element node i:

a) Determine whether it is or not a coplanar node

$\alpha = \cos^{-1}\left(\mathbf{n}\left(:, \mathbf{T}_{\mathrm{n}}(e,i)\right) \cdot \hat{\boldsymbol{k}}'(:,e) / \left\|\mathbf{n}\left(:, \mathbf{T}_{\mathrm{n}}(e,i)\right)\right\|\right)$ ← Scalar product!!

If $\alpha < 5°$ (we can consider node coplanar)

b) Evaluate artificial rotation stiffness component

$I_{\mathrm{dof}} = 6(\mathbf{T}_{\mathrm{n}}(e,i) - 1)$ ...

$\mathbf{K}_{\mathrm{r}}(I_{\mathrm{dof}}, I_{\mathrm{dof}}) = \mathbf{K}_{\mathrm{r}}(I_{\mathrm{d}}$ ...

End loop over element nodes

End loop over elements

3.3 Update stiffness matrix:

$[\overline{\mathbf{K}}] = [\mathbf{K}] + [\mathbf{K}_{\mathrm{r}}]$

In a 3D coupled beams-shells problem, coplanar nodes contained in a beam element do not induce a singularity in the global stiffness matrix (when beam stiffness is also accounted for). In Matlab, for a given shell element node `Tn(e,i)`, we can check whether it is part of a beam element with:

```
ind_beam = ismember(Tn(e,i),Tnb(:))
```

where `Tnb` refers to the nodal connectivities matrix for beam elements. Then, the criterion is:

`if` $\alpha < 5°$ `&& ind_beam == false`

Algorithm for problems involving flat shells

4) Compute global force vector:

4.1 Point loads:

$\{\hat{\boldsymbol{f}}\} = \{\mathbf{0}\}_{N_{\mathrm{dof}}\times 1}$

For row q in $[\mathbf{F}_\mathrm{e}]$

$$\hat{\mathbf{f}}(6(\mathbf{F}_\mathrm{e}(q,2)-1)+\mathbf{F}_\mathrm{e}(q,3),1) = \hat{\mathbf{f}}(6(\mathbf{F}_\mathrm{e}(q,2)-1)+\mathbf{F}_\mathrm{e}(q,3),1) + \mathbf{F}_\mathrm{e}(q,1)$$

End loop over rows in $[\mathbf{F}_\mathrm{e}]$

4.2 Nodal distributed forces:

$[\mathbf{P}] = \{\mathbf{0}\}_{N\times 6}$

For row r in $[\mathbf{P}_\mathrm{e}]$

$$\mathbf{P}\big(\mathbf{P}_\mathrm{e}(r,2),\mathbf{P}_\mathrm{e}(r,3)\big) = \mathbf{P}\big(\mathbf{P}_\mathrm{e}(r,2),\mathbf{P}_\mathrm{e}(r,3)\big) + \mathbf{P}_\mathrm{e}(r,1)$$

End loop over rows in $[\mathbf{P}_\mathrm{e}]$

4.3 Nodal body forces:

$[\mathbf{B}] = \{\mathbf{0}\}_{N\times 6}$

For row s in $[\mathbf{B}_\mathrm{e}]$

$$\mathbf{B}\big(\mathbf{B}_\mathrm{e}(s,2),\mathbf{B}_\mathrm{e}(s,3)\big) = \mathbf{B}\big(\mathbf{B}_\mathrm{e}(s,2),\mathbf{B}_\mathrm{e}(s,3)\big) + \mathbf{B}_\mathrm{e}(s,1)$$

End loop over rows in $[\mathbf{B}_\mathrm{e}]$

Algorithm for problems involving flat shells

4.4 Assembly process:

For each element e:

a) Compute element force vector:

$$\mathbf{b}(:,e) = \{\mathbf{B}(\mathbf{T}_\mathrm{n}(e,1),:), \mathbf{B}(\mathbf{T}_\mathrm{n}(e,2),:), \mathbf{B}(\mathbf{T}_\mathrm{n}(e,3),:), \mathbf{B}(\mathbf{T}_\mathrm{n}(e,4),:)\}^\mathrm{T}$$

$$\mathbf{p}(:,e) = \{\mathbf{P}(\mathbf{T}_\mathrm{n}(e,1),:), \mathbf{P}(\mathbf{T}_\mathrm{n}(e,2),:), \mathbf{P}(\mathbf{T}_\mathrm{n}(e,3),:), \mathbf{P}(\mathbf{T}_\mathrm{n}(e,4),:)\}^\mathrm{T}$$

$$\hat{\mathbf{f}}_\mathrm{e}(:,e) = [\mathbf{M}_\mathrm{e}(:,:,e)]\{\mathbf{b}(:,e)\}$$

For each Gauss point k (from 1 to 4):

$$\hat{\mathbf{f}}_\mathrm{e}(:,e) = \hat{\mathbf{f}}_\mathrm{e}(:,e) + \boldsymbol{S}_4(e,k)[\mathbf{R}(:,:,e)]^\mathrm{T}[\mathbf{N}(:,:,e,k)]^\mathrm{T}[\mathbf{N}(:,:,e,k)][\mathbf{R}(:,:,e)]\{\mathbf{p}(:,e)\}$$

End loop over Gauss points

b) Assembly to global force vector:

For each degree of freedom j from 1 to 6

$$I_\mathrm{dof}(j,1) = 6(\mathbf{T}_\mathrm{n}(e,1)-1)+j$$

$$I_\mathrm{dof}(6+j,1) = 6(\mathbf{T}_\mathrm{n}(e,2)-1)+j$$

$$I_\mathrm{dof}(12+j,1) = 6(\mathbf{T}_\mathrm{n}(e,3)-1)+j$$

$$I_\mathrm{dof}(18+j,1) = 6(\mathbf{T}_\mathrm{n}(e,4)-1)+j$$

End loop over DOFs

$$\hat{\mathbf{f}}(I_\mathrm{dof},1) = \hat{\mathbf{f}}(I_\mathrm{dof},1) + \hat{\mathbf{f}}_\mathrm{e}(:,e)$$

End loop over elements

Algorithm for problems involving flat shells

5) Boundary conditions:

5.1 Initialization:

$\{\hat{\boldsymbol{u}}\} = \{\mathbf{0}\}_{N_{\mathrm{dof}}\times 1}$

5.2 Prescribed and free DOFs:

For row p in $[\mathbf{U}_{\mathrm{p}}]$

$\boldsymbol{I}_{\mathrm{p}}(p) = 6\big(\mathbf{U}_{\mathrm{p}}(p,2) - 1\big) + \mathbf{U}_{\mathrm{p}}(p,3)$ (vector with prescribed degrees of freedom)

$\hat{\boldsymbol{u}}\big(\boldsymbol{I}_{\mathrm{p}}(p),1\big) = \mathbf{U}_{\mathrm{p}}(p,1)$

End loop over rows in $[\mathbf{U}_{\mathrm{p}}]$

$\boldsymbol{I}_{\mathrm{f}} = \{1{:}\,N_{\mathrm{dof}}\} - \{\boldsymbol{I}_{\mathrm{p}}\}$ (**Tip**: in Matlab, this operation can be done with the **`setdiff`** function)

```
If = setdiff(1:Ndof,Ip)
```

6) Solve system of equations (static case):

6.1 Solve system:

$\hat{\boldsymbol{u}}(\boldsymbol{I}_{\mathrm{f}},1) = [\overline{\mathbf{K}}(\boldsymbol{I}_{\mathrm{f}},\boldsymbol{I}_{\mathrm{f}})]^{-1}\big(\hat{\boldsymbol{f}}(\boldsymbol{I}_{\mathrm{f}},1) - [\overline{\mathbf{K}}(\boldsymbol{I}_{\mathrm{f}},\boldsymbol{I}_{\mathrm{p}})]\{\hat{\boldsymbol{u}}(\boldsymbol{I}_{\mathrm{p}},1)\}\big)$ (displacements/rotations at free DOFs)

$\hat{\boldsymbol{f}}_{\mathrm{R}} = [\overline{\mathbf{K}}]\{\hat{\boldsymbol{u}}\} - \{\hat{\boldsymbol{f}}\}$ (reaction forces/moments at prescribed DOFs)

Algorithm for problems involving flat shells

7) Postprocess: Computing local strain and stress in shell elements

7.1 Get stress and strain at each Gauss point:

For each element e:

a) Get each strain component :

For each degree of freedom j from 1 to 6

$$I_{\text{dof}}(j,1) = 6(\mathbf{T}_{\text{n}}(e,1)-1)+j$$
$$I_{\text{dof}}(6+j,1) = 6(\mathbf{T}_{\text{n}}(e,2)-1)+j$$
$$I_{\text{dof}}(12+j,1) = 6(\mathbf{T}_{\text{n}}(e,3)-1)+j$$
$$I_{\text{dof}}(18+j,1) = 6(\mathbf{T}_{\text{n}}(e,4)-1)+j$$

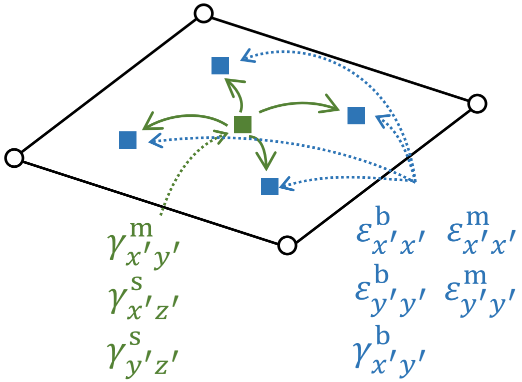
End loop over DOFs

For each Gauss point k (from 1 to 4):

$$\bar{\boldsymbol{\varepsilon}}'_{\text{b}}(:,e,k) = [\mathbf{B}'_{\text{b}}(:,:,e,k)][\mathbf{R}(:,:,e)]\{\hat{\boldsymbol{u}}(I_{\text{dof}},1)\}$$
$$\bar{\boldsymbol{\varepsilon}}'_{\text{m}}(1:2,e,k) = \left[\mathbf{B}'_{\text{m}_{\text{n}}}(:,:,e,k)\right][\mathbf{R}(:,:,e)]\{\hat{\boldsymbol{u}}(I_{\text{dof}},1)\}$$
$$\bar{\boldsymbol{\varepsilon}}'_{\text{m}}(3,e,k) = \left[\mathbf{B}'_{\text{m}_{\text{t}}}(:,:,e)\right][\mathbf{R}(:,:,e)]\{\hat{\boldsymbol{u}}(I_{\text{dof}},1)\}$$
$$\bar{\boldsymbol{\varepsilon}}'_{\text{s}}(:,e,k) = [\mathbf{B}'_{\text{s}}(:,:,e)][\mathbf{R}(:,:,e)]\{\hat{\boldsymbol{u}}(I_{\text{dof}},1)\}$$

End loop over Gauss points

Since $\gamma^{\text{m}}_{x'y'}$, $\gamma^{\text{s}}_{x'z'}$ and $\gamma^{\text{s}}_{y'z'}$ are evaluated at just one Gauss point, we assume that the value is the same for all the element, so we assign them at the 4-Gauss points positions where the other strain components are evaluated.



Algorithm for problems involving flat shells

b) Get stress:

$$[\mathbf{C}_\mathrm{p}] = \begin{bmatrix} 1 & \nu^{(\mathbf{T}_\mathrm{m}(e))} & 0 \\ \nu^{(\mathbf{T}_\mathrm{m}(e))} & 1 & 0 \\ 0 & 0 & \left(1-\nu^{(\mathbf{T}_\mathrm{m}(e))}\right)/2 \end{bmatrix} E^{(\mathbf{T}_\mathrm{m}(e))} \Big/ \left(1-\nu^{(\mathbf{T}_\mathrm{m}(e))^2}\right)$$

$$[\mathbf{C}_\mathrm{s}] = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} E^{(\mathbf{T}_\mathrm{m}(e))} \Big/ \left(2\left(1+\nu^{(\mathbf{T}_\mathrm{m}(e))}\right)\right)$$

For each Gauss point k (from 1 to 4):

$\boldsymbol{\sigma}'_\mathrm{m}(:,e,k) = [\mathbf{C}_\mathrm{p}]\{\bar{\boldsymbol{\varepsilon}}'_\mathrm{m}(:,e,k)\}$ (constant membrane stress over the thickness)

$\boldsymbol{\sigma}'_\mathrm{s}(:,e,k) = [\mathbf{C}_\mathrm{s}]\{\bar{\boldsymbol{\varepsilon}}'_\mathrm{s}(:,e,k)\}$ (constant shear stress over the thickness **assumed**)

$\boldsymbol{\sigma}'_\mathrm{b}(:,e,k) = [\mathbf{C}_\mathrm{p}]h^{(\mathbf{T}_\mathrm{m}(e))}\{\bar{\boldsymbol{\varepsilon}}'_\mathrm{b}(:,e,k)\}/2$ (bending stress on the top surface)

$\boldsymbol{\sigma}'_+ = \{\boldsymbol{\sigma}'_\mathrm{m}(:,e,k) + \boldsymbol{\sigma}'_\mathrm{b}(:,e,k); \boldsymbol{\sigma}'_\mathrm{s}(:,e,k)\}^\mathrm{T}$ (stress on the top surface)

$$\sigma_\mathrm{VM}^+ = \left(\sigma'_+(1)^2 + \sigma'_+(2)^2 - \sigma'_+(1)\sigma'_+(2) + 3(\sigma'_+(3)^2 + \sigma'_+(4)^2 + \sigma'_+(5)^2)\right)^{1/2}$$

$\boldsymbol{\sigma}'_- = \{\boldsymbol{\sigma}'_\mathrm{m}(:,e,k) - \boldsymbol{\sigma}'_\mathrm{b}(:,e,k); \boldsymbol{\sigma}'_\mathrm{s}(:,e,k)\}^\mathrm{T}$ (stress on the bottom surface)

$$\sigma_\mathrm{VM}^- = \left(\sigma'_-(1)^2 + \sigma'_-(2)^2 - \sigma'_-(1)\sigma'_-(2) + 3(\sigma'_-(3)^2 + \sigma'_-(4)^2 + \sigma'_-(5)^2)\right)^{1/2}$$

$$\boldsymbol{\sigma}_\mathrm{VM}(e,k) = \max\{\sigma_\mathrm{VM}^+, \sigma_\mathrm{VM}^-\}$$

End loop over Gauss points

End loop over elements