Algorithm for problems involving flat shells

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

UPC

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:

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

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

- Nodal connectivities matrix:

$$[\mathbf{T_n}] = \begin{bmatrix} & & \vdots & & \\ n_1^{(e)} & n_2^{(e)} & n_3^{(e)} & n_4^{(e)} \\ & & \vdots & & \end{bmatrix} \Bigg\} \text{ Number of elements } N_e$$

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

Algorithm for problems involving flat shells

- Material connectivites matrix:

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

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

1.3 Boundary conditions:

- Fix nodes and DOFs matrix:

$$[\mathbf{U}_p] = \begin{bmatrix} & \vdots & \\ u^{(p)} & n^{(p)} & j^{(p)} \\ & \vdots & \end{bmatrix} \Bigg\} \begin{array}{l} \text{Number of} \\ \text{prescribed DOFs} \end{array}$$

$u^{(p)}$: 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):

$$[\mathbf{F}_e] = \begin{bmatrix} & \vdots & \\ f^{(q)} & n^{(q)} & j^{(q)} \\ & \vdots & \end{bmatrix} \Bigg\} \begin{array}{l} \text{Number of point} \\ \text{forces} \end{array}$$

$f^{(q)}$: 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):

$$[\mathbf{P}_e] = \begin{bmatrix} \vdots & & \\ p^{(r)} & n^{(r)} & j^{(r)} \\ & \vdots & \end{bmatrix} \Bigg\}$$ 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):

$$[\mathbf{B}_e] = \begin{bmatrix} \vdots & & \\ b^{(s)} & n^{(s)} & j^{(s)} \\ & \vdots & \end{bmatrix} \Bigg\}$$ 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)

<u>Note</u>: 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:

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

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

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

2.2 Assembly process:

For each element e:

a) Compute rotation matrix:

Vector product!!

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

$$\{\widehat{k}'\} = \{S\}/\|\{S\}\| \quad (\equiv \text{normal vector of the flat shell element})$$

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

$$\{\widehat{i}'\} = \{d\}/\|\{d\}\| \;; \quad \{\widehat{j}'\} = \{\widehat{k}'\} \times \{\widehat{i}'\}$$

$$[\mathbf{R}'] = \begin{bmatrix} \{\widehat{i}'\} & \{\widehat{j}'\} & \{\widehat{k}'\} & [\mathbf{0}]_{3\times 2} \\ & [\mathbf{0}]_{3\times 3} & & \{\widehat{i}'\} \quad \{\widehat{j}'\} \end{bmatrix}^{\mathrm{T}}$$

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

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:

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

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

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

$[\boldsymbol{J}_1] = [\boldsymbol{0}]_{2\times2}$

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

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

End loop over nodes

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

$S_1 = 4 \det[\boldsymbol{J}_1]$ ($\equiv$ 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:

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

End loop over nodes

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

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

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

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

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

$$\mathbf{B}_{\text{m}_\text{t}}'^{(i)}(:,:,i) = [\boldsymbol{N}_{1,\boldsymbol{x}'}(2,i) \quad \boldsymbol{N}_{1,\boldsymbol{x}'}(1,i) \quad 0 \quad 0 \quad 0]$$

End loop over nodes

$$\bar{\mathbf{C}}_{\text{m}_\text{t}}' = h^{(\mathbf{T}_\text{m}(e))} E^{(\mathbf{T}_\text{m}(e))} \Big/ \Big(2\big(1 + \nu^{(\mathbf{T}_\text{m}(e))}\big) \Big)$$

$$\mathbf{B}_{\text{m}_\text{t}}'(:,:,e) = \Big[\mathbf{B}_{\text{m}_\text{t}}'^{(i)}(:,:,1), \mathbf{B}_{\text{m}_\text{t}}'^{(i)}(:,:,2), \mathbf{B}_{\text{m}_\text{t}}'^{(i)}(:,:,3), \mathbf{B}_{\text{m}_\text{t}}'^{(i)}(:,:,4) \Big]$$

$$\mathbf{K}_\text{m}(:,:,e) = S_1 [\mathbf{R}(:,:,e)]^\text{T} \big[\mathbf{B}_{\text{m}_\text{t}}'(:,:,e)\big]^\text{T} [\bar{\mathbf{C}}_{\text{m}_\text{t}}'][\mathbf{B}_{\text{m}_\text{t}}'(:,:,e)][\mathbf{R}(:,:,e)]$$

Algorithm for problems involving flat shells

c2) 4 Gauss points quadrature matrices:

$$\mathbf{K}_b(:,:,e) = [\mathbf{0}]_{24\times24}$$

$$\mathbf{M}_e(:,:,e) = [\mathbf{0}]_{24\times24}$$

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

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

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

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

$$\boldsymbol{J}_4 = [\mathbf{0}]_{2\times2}$$

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

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

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

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

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

End loop over nodes

Algorithm for problems involving flat shells

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

$$S_4(e,k) = w_4(k) \cdot \det[\boldsymbol{J}_4] \quad (\equiv \text{area associated to Gauss point})$$

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

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

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

End loop over nodes

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

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

$$\mathbf{K}_m(:,:,e) = \mathbf{K}_m(:,:,e) + S_4(e,k)[\mathbf{R}(:,:,e)]^{\mathrm{T}}\left[\mathbf{B}_{m_n}'(:,:,e,k)\right]^{\mathrm{T}}\left[\bar{\mathbf{C}}_{m_n}'\right]\left[\mathbf{B}_{m_n}'(:,:,e,k)\right][\mathbf{R}(:,:,e)]$$

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:

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

Algorithm for problems involving flat shells

c2.2) Bending component of stiffness matrix:

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

$$\mathbf{B}_b'^{(i)}(:,:,i) = \begin{bmatrix} 0 & 0 & 0 & 0 & N_{4,x'}(1,i) \\ 0 & 0 & 0 & N_{4,x'}(2,i) & 0 \\ 0 & 0 & 0 & -N_{4,x'}(1,i) & N_{4,x'}(2,i) \end{bmatrix}$$

End loop over nodes

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

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

$$\mathbf{K}_b(:,:,e) = \mathbf{K}_b(:,:,e) + S_4(e,k)[\mathbf{R}(:,:,e)]^{\mathrm{T}}[\mathbf{B}_b'(:,:,e,k)]^{\mathrm{T}}[\bar{\mathbf{C}}_b'][\mathbf{B}_b'(:,:,e,k)][\mathbf{R}(:,:,e)]$$

c2.3) Mass matrix:

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

$$\mathbf{N}^{(i)}(:,:,i) = N_4(i)[\mathbf{1}]_{5\times5} \quad ([\mathbf{1}]_{5\times5} \equiv \text{Identity matrix of } 5 \times 5)$$

End loop over nodes

Algorithm for problems involving flat shells

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

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

$$\mathbf{M}_e(:,:,e) = \mathbf{M}_e(:,:,e) + S_4(e,k)[\mathbf{R}(:,:,e)]^{\mathrm{T}}[\mathbf{N}(:,:,e,k)]^{\mathrm{T}}[\bar{\boldsymbol{\rho}}'][\mathbf{N}(:,:,e,k)][\mathbf{R}(:,:,e)]$$

End loop over Gauss points

d) Assembly to global matrices:

For each degree of freedom j from 1 to 6

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

End loop over DOFs

$$\mathbf{K}(I_{\mathrm{dof}},I_{\mathrm{dof}}) = \mathbf{K}(I_{\mathrm{dof}},I_{\mathrm{dof}}) + \mathbf{K}_m(:,:,e) + \mathbf{K}_b(:,:,e) + \mathbf{K}_s(:,:,e)$$

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

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:

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

$$S(e) = \sqrt{\left(S(1)\right)^2 + \left(S(2)\right)^2 + \left(S(3)\right)^2}$$
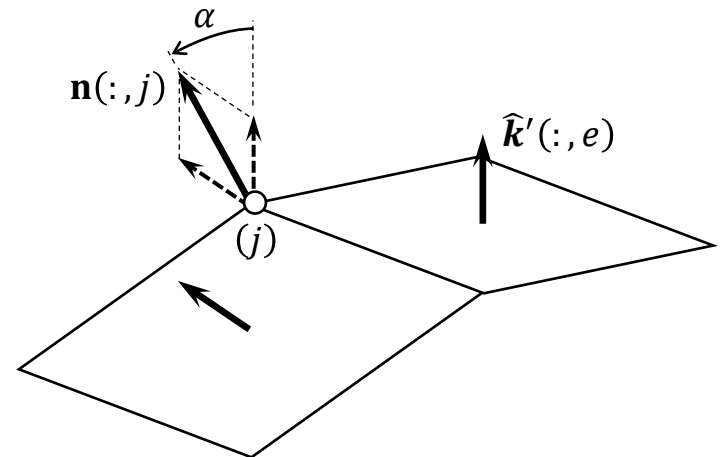
$$\widehat{k}'(:,e) = \{S\}/S(e)$$

b) Assemble to get nodal normal:

For each element node i
$$\mathbf{n}(:,\mathbf{T_n}(e,i)) = \mathbf{n}(:,\mathbf{T_n}(e,i)) + \widehat{k}'(:,e)$$
End loop over element nodes

End loop over elements

Algorithm for problems involving flat shells

3.2 Compute artificial rotation matrix:

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

For each element e:

    For each element node i:

        a) Determine whether it is or not a coplanar node

$$\alpha = \cos^{-1}\big(\mathbf{n}(:, \mathbf{T}_n(e,i)) \cdot \widehat{\boldsymbol{k}}'(:,e) / \|\mathbf{n}(:, \mathbf{T}_n(e,i))\|\big) \quad \longleftarrow \quad \boxed{\text{Scalar product!!}}$$

        If $\alpha < 0.1°$ (we can consider node coplanar)

        b) Evaluate artificial rotation stiffness component

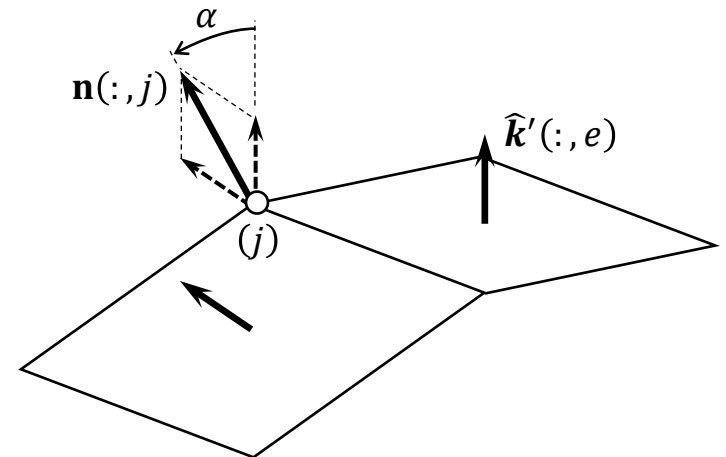$$I_{\text{dof}} = 6(\mathbf{T}_n(e,i) - 1) + \{4, 5, 6\}^{\text{T}}$$

$$\mathbf{K}_r(I_{\text{dof}}, I_{\text{dof}}) = \mathbf{K}_r(I_{\text{dof}}, I_{\text{dof}}) + E^{(\mathbf{T}_m(e))} h^{(\mathbf{T}_m(e))} S(e) \{\widehat{\boldsymbol{k}}'(:,e)\} \{\widehat{\boldsymbol{k}}'(:,e)\}^{\text{T}}$$

    End loop over element nodes

End loop over elements

3.3 Update stiffness matrix:

$$[\overline{\mathbf{K}}] = [\mathbf{K}] + [\mathbf{K}_r]$$

Algorithm for problems involving flat shells

3.2 Compute artificial rotation matrix:

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

For each element e:

For each element node i:

a) Determine whether it is or not a coplanar node

$$\alpha = \cos^{-1}\big(\mathbf{n}(:, \mathbf{T}_n(e, i)) \cdot \widehat{\boldsymbol{k}}'(:, e) / \|\mathbf{n}(:, \mathbf{T}_n(e, i))\|\big) \quad \longleftarrow \boxed{\text{Scalar product!!}}$$

$\boxed{\text{If } \alpha < 0.1° \text{ (we can consider node coplanar)}}$

b) Evaluate artificial rotation stiffness component

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

$$\mathbf{K}_r(I_{\mathrm{dof}}, I_{\mathrm{dof}}) = \mathbf{K}_r(I_{\mathrm{d}}$$

End loop over element nodes

End loop over elements

3.3 Update stiffness matrix:

$$[\overline{\mathbf{K}}] = [\mathbf{K}] + [\mathbf{K}_r]$$

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 α < 5° && ind_beam == false
```

Algorithm for problems involving flat shells

4) Compute global force vector:

4.1 Point loads:

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

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

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

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

4.2 Nodal distributed forces:

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

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

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

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

4.3 Nodal body forces:

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

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

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

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

Algorithm for problems involving flat shells

4.4 Assembly process:

For each element e:

a) Compute element force vector:

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

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

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

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

$$\hat{\mathbf{f}}_e(:,e) = \hat{\mathbf{f}}_e(:,e) + S_4(e,k)[\mathbf{R}(:,:,e)]^T[\mathbf{N}(:,:,e,k)]^T[\mathbf{N}(:,:,e,k)][\mathbf{R}(:,:,e)]\{\mathbf{p}(:,e)\}$$

End loop over Gauss points

b) Assembly to global force vector:

For each degree of freedom j from 1 to 6

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

End loop over DOFs

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

End loop over elements

Algorithm for problems involving flat shells

5) Boundary conditions:

5.1 Initialization:

$\{\hat{u}\} = \{0\}_{N_{\text{dof}} \times 1}$

5.2 Prescribed and free DOFs:

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

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

$\hat{u}\big(I_{\text{p}}(p),1\big) = \mathbf{U}_{\text{p}}(p,1)$

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

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

$\text{If = setdiff(1:Ndof,Ip)}$

6) Solve system of equations (static case):

6.1 Solve system:

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

$\hat{f}_{\text{R}} = [\overline{\mathbf{K}}]\{\hat{u}\} - \{\hat{f}\}$ (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

$I_{\text{dof}}(j, 1) = 6(\mathbf{T}_n(e, 1) - 1) + j$

$I_{\text{dof}}(6 + j, 1) = 6(\mathbf{T}_n(e, 2) - 1) + j$

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

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

End loop over DOFs

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

$\bar{\boldsymbol{\varepsilon}}'_{\text{b}}(:, e, k) = [\mathbf{B}'_{\text{b}}(:,:, e, k)][\mathbf{R}(:,:, e)]\{\hat{\boldsymbol{u}}(I_{\text{dof}}, 1)\}$

$\bar{\boldsymbol{\varepsilon}}'_{\text{m}}(1{:}2, e, k) = \left[\mathbf{B}'_{\text{m}_n}(:,:, e, k)\right][\mathbf{R}(:,:, e)]\{\hat{\boldsymbol{u}}(I_{\text{dof}}, 1)\}$

$\bar{\boldsymbol{\varepsilon}}'_{\text{m}}(3, e, k) = \left[\mathbf{B}'_{\text{m}_t}(:,:, e)\right][\mathbf{R}(:,:, e)]\{\hat{\boldsymbol{u}}(I_{\text{dof}}, 1)\}$

$\bar{\boldsymbol{\varepsilon}}'_{\text{s}}(:, e, k) = [\mathbf{B}'_{\text{s}}(:,:, e)][\mathbf{R}(:,:, e)]\{\hat{\boldsymbol{u}}(I_{\text{dof}}, 1)\}$

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.

$\gamma^{\text{m}}_{x'y'}$
$\gamma^{\text{s}}_{x'z'}$
$\gamma^{\text{s}}_{y'z'}$

$\varepsilon^{\text{b}}_{x'x'}$ $\varepsilon^{\text{m}}_{x'x'}$
$\varepsilon^{\text{b}}_{y'y'}$ $\varepsilon^{\text{m}}_{y'y'}$
$\gamma^{\text{b}}_{x'y'}$
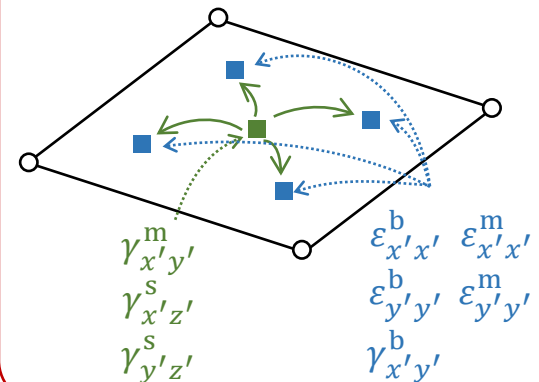
Algorithm for problems involving flat shells

b) Get stress:

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

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

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

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

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

$\boldsymbol{\sigma}'_b(:, e, k) = [\mathbf{C}_p]h^{(\mathbf{T}_m(e))}\{\bar{\boldsymbol{\varepsilon}}'_b(:, e, k)\}/2$ (bending stress on the top surface)

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

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

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

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

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

End loop over Gauss points

End loop over elements