Quote of the Day

“I intend to live forever, or die trying.”

-Groucho Marx
Before we get started…

- On Monday, we learned:
  - Define constraints with ANCF finite elements and rigid bodies
  - ANCF cable element kinematics
  - ANCF cable element strains
  - ANCF cable element inertia forces

- This lecture…
  - Wrap up implementation of ANCF cable element in Chrono
  - Plates and shells
  - Introduce Chrono’s bilinear ANCF shell element
  - Kinematics of ANCF shell element
  - Curvilinear initial configuration
  - Inertia forces
  - Internal forces
ANCF Cable: Virtual Work of Elastic Forces

The virtual of the elastic forces for the gradient-deficient beam element may be defined as

$$\delta W_e = \int \left[ EA \varepsilon_x \delta \varepsilon_x + EI \kappa \delta \kappa \right] dx$$

where

$$\varepsilon_x = \frac{1}{2} (r_x^T r_x - 1)$$

and

$$\kappa = \frac{|r_x \times r_{xx}|}{|r_x|^3}$$

are the axial Green-Lagrange strain and bending curvature

- Calculation of strain variations for virtual work

$$\delta \varepsilon_x = \frac{\partial}{\partial e} \left( \frac{1}{2} (r_x^T r_x - 1) \right) \delta e = r_x^T \frac{\partial r_x}{\partial e} \delta e$$

- For bending curvature:

$$\delta \kappa = \frac{\partial \kappa}{\partial e} \delta e = \frac{1}{g^2} \left( g \frac{\partial f}{\partial e} - f \frac{\partial g}{\partial e} \right) \delta e$$

$$\frac{\partial f}{\partial e} = \frac{\partial}{\partial e} \sqrt{(r_x \times r_{xx})^T (r_x \times r_{xx})} = \frac{(r_x \times r_{xx})^T \left( \frac{\partial}{\partial e} r_x \times r_{xx} + r_x \times \frac{\partial}{\partial e} r_{xx} \right)}{\sqrt{(r_x \times r_{xx})^T (r_x \times r_{xx})}}$$

$$\frac{\partial g}{\partial e} = \frac{\partial}{\partial e} \left( r_x^T r_x \right)^{3/2} = 3 \left( r_x^T r_x \right)^{1/2} \left( r_x^T \frac{\partial r_x}{\partial e} \right)$$
b. Generalized internal forces
Axial

class MyForcesAxial : public ChIntegrable1D<ChMatrixNM<double, 12, 1>> {
    public:
        ChElementCableANCF* element;

        ChMatrixNM<double, 4, 3>* d;   // this is an external matrix, use pointer
        ChMatrixNM<double, 12, 1>* d_dt; // this is an external matrix, use pointer
        ChMatrixNM<double, 3, 12> Sd;
        ChMatrixNM<double, 1, 4> N;
        ChMatrixNM<double, 1, 4> Nd;
        ChMatrixNM<double, 1, 12> strainD;
        ChMatrixNM<double, 1, 1> strain;
        ChMatrixNM<double, 1, 3> Nd_d;
        ChMatrixNM<double, 12, 12> temp;

    // Evaluate (strainD'*strain) at point x
    virtual void Evaluate(ChMatrixNM<double, 12, 1>& result, const double x) {
        element->shapeFunctionsDerivatives(Nd, x);  // Evaluate shape function derivatives at x

        // Sd=[Nd1*eye(3) Nd2*eye(3) Nd3*eye(3) Nd4*eye(3)]
        ChMatrix33<> Sdi;
        Sdi.FillDiag(Nd(0));
        Sd.PasteMatrix(&Sdi, 0, 0);
        Sdi.FillDiag(Nd(1));
        Sd.PasteMatrix(&Sdi, 0, 3);
        Sdi.FillDiag(Nd(2));
        Sd.PasteMatrix(&Sdi, 0, 6);
        Sdi.FillDiag(Nd(3));
        Sd.PasteMatrix(&Sdi, 0, 9);

        Nd_d = Nd * (*d);
        strainD = Nd_d * Sd;

        result = strainD * strain;

        // Declarations needed
}

\[
-\mathbf{r}_x(\xi) \frac{\partial \mathbf{e}_{xx}(\xi)}{\partial e}
\]
b. Generalized internal forces

Axial

Calculation of strain

Optional addition of damping

Result dependent on space parameter

// Add damping forces if selected
if (element->m_use_damping)
    strain(0, 0) += (element->m_alpha) * (strainD * (*d_dt))(0, 0);

Result dependent on space parameter

MyForcesAxial myformulaAx;
myformulaAx.d = &d;
myformulaAx.d_dt = &vel_vector;
myformulaAx.element = this;

ChMatrixNM<double, 12, 1> Faxial;
ChQuadrature::Integrate1D<ChMatrixNM<double, 12, 1>>(Faxial, // result of integration will
go there
    myformulaAx, // formula to integrate
    0,           // start of x
    1,           // end of x
    5            // order of integration
);

Faxial *= -E * Area * length;

Fi = Faxial;
c. Generalized internal forces. Bending strain

// Integrate (k_e'*k_e)

class MyForcesCurv : public ChIntegrable1D<ChMatrixNM<double, 12, 1> > {
public:
    ChElementCableANCF* element;
    ChMatrixNM<double, 4, 3>* d; // this is an external matrix, use pointer
    ChMatrixNM<double, 12, 1>* d_dt; // this is an external matrix, use pointer
    ChMatrixNM<double, 3, 12>* Sd;
    ChMatrixNM<double, 3, 12>* Sdd;
    ChMatrixNM<double, 1, 4> Nd;
    ChMatrixNM<double, 1, 4> Ndd;
    ChMatrixNM<double, 1, 3>* r_x;
    ChMatrixNM<double, 1, 3>* r_xx;
    ChMatrixNM<double, 1, 12>* g_e;
    ChMatrixNM<double, 1, 12>* f_e;
    ChMatrixNM<double, 1, 12>* k_e;
    ChMatrixNM<double, 3, 12>* fe1;

    // Evaluate at point x
    virtual void Evaluate(ChMatrixNM<double, 12, 1>& result, const double x) {
        element->ShapeFunctionsDerivatives(Nd, x);
        element->ShapeFunctionsDerivatives2(Ndd, x);

        ChMatrixNM<double, 1, 3> r_x;
        ChMatrixNM<double, 1, 3> r_xx;
        ChMatrixNM<double, 1, 12> g_e;
        ChMatrixNM<double, 1, 12> f_e;
        ChMatrixNM<double, 1, 12> k_e;
        ChMatrixNM<double, 3, 12> fe1;

        // First and second shape functions needed

        \[ \kappa = \frac{f}{g} = \frac{|\mathbf{r}_x \times \mathbf{r}_{xx}|}{|\mathbf{r}_x|^3}, \quad f = |\mathbf{r}_x \times \mathbf{r}_{xx}|, \quad g = |\mathbf{r}_x|^3 \]

        \[ \delta \kappa = \frac{\partial \kappa}{\partial \mathbf{e}} \delta \mathbf{e} = \frac{1}{g^2} \left( g \frac{\partial f}{\partial \mathbf{e}} - f \frac{\partial g}{\partial \mathbf{e}} \right) \delta \mathbf{e} \]
c. Generalized internal forces.

Bending strain

Shape function matrices: First and second derivative

// Sd=[Nd1*eye(3) Nd2*eye(3) Nd3*eye(3) Nd4*eye(3)]
// Sdd=[Ndd1*eye(3) Ndd2*eye(3)
Ndd3*eye(3) Ndd4*eye(3)]

ChMatrix33<> Sdi;
Sdi.FillDiag(Nd(0));
Sd.PasteMatrix(&Sdi, 0, 0);
Sdi.FillDiag(Nd(1));
Sd.PasteMatrix(&Sdi, 0, 3);
Sdi.FillDiag(Nd(2));
Sd.PasteMatrix(&Sdi, 0, 6);
Sdi.FillDiag(Nd(3));
Sd.PasteMatrix(&Sdi, 0, 9);
Sdi.FillDiag(Ndd(0));
Sdd.PasteMatrix(&Sdi, 0, 0);
Sdi.FillDiag(Ndd(1));
Sdd.PasteMatrix(&Sdi, 0, 3);
Sdi.FillDiag(Ndd(2));
Sdd.PasteMatrix(&Sdi, 0, 6);
Sdi.FillDiag(Ndd(3));
Sdd.PasteMatrix(&Sdi, 0, 9);

r_x.MatrMultiply(Nd, (*d)); // r_x(ξ)

r_x=d'*Nd'; (transposed)

r_xx.MatrMultiply(Ndd, (*d)); // r_xx(ξ)

r_xx=d'*Ndd'; (transposed)
c. Generalized internal forces. Bending strain

Building the bending strain formula and its variation—to obtain generalized bending force

\[
\kappa = \frac{f}{g} = \frac{|\mathbf{r}_x \times \mathbf{r}_{xx}|}{|\mathbf{r}_x|^3}, \quad f = |\mathbf{r}_x \times \mathbf{r}_{xx}|, \quad g = |\mathbf{r}_x|^3
\]

\[
\delta \kappa = \frac{\partial \kappa}{\partial \mathbf{e}} \delta \mathbf{e} = \frac{1}{g^2} \left( g \frac{\partial f}{\partial \mathbf{e}} - f \frac{\partial g}{\partial \mathbf{e}} \right) \delta \mathbf{e}
\]

\[\text{r}_xx.\text{MatrMultiply}(\text{Ndd}, (*d)); \quad \text{r}_xx=d'*\text{Ndd}'; \quad \text{(transposed)}\]

// if (r_xx.Length()==0)
// {r_xx(0)=0; r_xx(1)=1; r_xx(2)=0;}
ChVector<> vr_x(r_x(0), r_x(1), r_x(2));
ChVector<> vr_xx(r_xx(0), r_xx(1), r_xx(2));
ChVector<> vf1 = Vcross(vr_x, vr_xx);
double f = vf1.Length();
double g1 = vr_x.Length();
double g = pow(g1, 3);
double k = f / g;
g_e = (Nd * (*d)) * Sd;
g_e *= (3 * g1);

// do: fe1=cross(Sd,r_xxrep)+cross(r_xrep,Sdd);
for (int col = 0; col < 12; ++col) {
    ChVector<> Sd_i = Sd.ClipVector(0, col);
    fe1.PasteVector(Vcross(Sd_i, vr_xx), 0, col);
    ChVector<> Sdd_i = Sdd.ClipVector(0, col);
    fe1.PasteSumVector(Vcross(vr_x, Sdd_i), 0, col);
}
ChMatrixNM<\text{double}, 3, 1> f1;
f1.PasteVector(vf1, 0, 0);

if (f == 0)
    f_e.MatrTMultiply(f1, fe1);
else {
    f_e.MatrTMultiply(f1, fe1);
    f_e *= (1 / f);
}
### c. Generalized internal forces.

Bending strain

\[
\kappa = \frac{f}{g} = \frac{\mathbf{r}_x \times \mathbf{r}_{xx}}{\left| \mathbf{r}_x \right|^3}, \quad f = \left| \mathbf{r}_x \times \mathbf{r}_{xx} \right|, \quad g = \left| \mathbf{r}_x \right|^3
\]

\[
\delta \kappa = \frac{\partial \kappa}{\partial \mathbf{e}} \delta \mathbf{e} = \frac{1}{g^2} \left( g \frac{\partial f}{\partial \mathbf{e}} - f \frac{\partial g}{\partial \mathbf{e}} \right) \delta \mathbf{e}
\]

\[
k_e = (f_e - g_e * f) * (1 / (\text{pow}(g, 2)));
\]

```cpp
// result: k_e''\*k
result.CopyFromMatrixT(k_e);

// Add damping if selected by user: curvature rate
if (element->m_use_damping)
  k += (element->m_alpha) * (k_e * (*d_dt))(0, 0);

result *= k;
```

```cpp
MyForcesCurv myformulaCurv;
myformulaCurv.d = &d;
myformulaCurv.d_dt = &vel_vector;
myformulaCurv.element = this;
```

```cpp
ChMatrixNM<double, 12, 1> Fcurv;
ChQuadrature::Integrate1D<ChMatrixNM<double, 12, 1> >(Fcurv, // result of integration will
  myformulaCurv, // formula to integrate
  0, // start of x
  1, // end of x
  3 // order of integration
);

Fcurv *= -E * I * length; // note Iy should be the same value (circular section assumption)
```

```cpp
// Subtract contribution of initial configuration
Fi += Fcurv;
```

```cpp
// Integrate formula with order 3. Account for initial configuration. Multiply by material constants.
Fi -= this->m_GenForceVec0;
```
1. Shells and Plates

Definition of shell/plate: structural element whose thickness is significantly small than its other two dimensions. Remember, for beams, two dimensions were smaller than the other one.

**Plate:** Action of an external load is counteracted by bending and twisting moments and (maybe) shear

**Shell:** Action of an external load is counteracted by membrane stresses; that is, stresses are parallel to the tangential plane and distributed uniformly over the thickness:

For both cases:

- Thin/thick shell/plates may be considered: Strain definitions differ
- They have known differential equations. Some cases, analytical solution
- Shell thickness much smaller than its radius of curvature
1. Shells

**Shell:** Membrane stresses

**Plate:** Bending, twisting, shear

1. Shells

If assuming shell conditions, bending and shearing forces vanish:

\[ N_x = \int_{-h/2}^{h/2} \sigma_x \left(1 - \frac{z}{r_y}\right) dz = \frac{Eh}{1-\nu^2} (\varepsilon_x + \nu \varepsilon_y) \]

\[ N_y = \int_{-h/2}^{h/2} \sigma_y \left(1 - \frac{z}{r_x}\right) dz = \frac{Eh}{1-\nu^2} (\varepsilon_y + \nu \varepsilon_x) \]

\[ N_{yx} = N_{xy} = \int_{-h/2}^{h/2} \tau_{yx} \left(1 - \frac{z}{r_x}\right) dz = \frac{\gamma h E}{2(1+\nu)} \]

Shell: Membrane stresses
1. Shells

“In many problems of deformation of shells the bending stresses can be neglected, and only the stresses due to strain in the middle surface of the shell need be considered. Take, as an example, a thin spherical container submitted to the action of a uniformly distributed internal pressure normal to the surface of the shell. Under this action the middle surface of the shell undergoes a uniform strain; and since the thickness of the shell is small, the tensile stresses can be assumed as uniformly distributed across the thickness.”

Examples

- Balloon
- Pressurized tank
- Tire (in many scenarios)
2. ANCF Shell Elements

In terms of formulations of finite elements:

- Plate elements look for capturing \textit{bending curvature}
- Shell elements are based on \textit{in-plane strains}

ANCF shell/plate elements may be categorized according to

- Whether tranverse shear is considered (\textit{thick} or \textit{thin})
- The set of position gradient vectors used

ANCF shell/plate elements may also suffer from a variety of locking phe- nomena

- Shear
- Volumetric/Poisson
2. ANCF Shell Elements. Types

- Fully parameterized
- Gradients are continuous
- Continuum-based
- Locking
- Gradients $r_x$ and $r_y$, thin plate
- Gradients are discontinuous
- Structural based – plate formulation

$$\kappa_{\text{thin}} = \begin{bmatrix} n^T \frac{\partial^2 r_{\text{mid}}}{\partial x \partial x} & n^T \frac{\partial^2 r_{\text{mid}}}{\partial y \partial y} & n^T \frac{\partial^2 r_{\text{mid}}}{\partial x \partial y} \end{bmatrix}^T$$

$$n = \frac{r_x \times r_y}{|r_x \times r_y|}$$
2. ANCF Shell Elements. Types

- Gradients $r_x$ and $r_y$, and $\frac{\partial^2 r}{\partial x \partial y}$, thin plate
- Gradients are continuous
- Structural based – plate formulation

$$\kappa_{\text{thin}} = \left[ n^T \frac{\partial^2 r_{\text{mid}}}{\partial x \partial x} \quad n^T \frac{\partial^2 r_{\text{mid}}}{\partial y \partial y} \quad n^T \frac{\partial^2 r_{\text{mid}}}{\partial x \partial y} \right]^T$$

$$n = \frac{r_x \times r_y}{|r_x \times r_y|}$$

- Gradients $r_x$ and $r_y$, and $\frac{\partial^2 r}{\partial x \partial y}$, thin plate
- 8 nodes
- Continuum-based: St. Venant-Kirchhoff material
- Extends membrane theory to out of plane strains

$$\varepsilon = 0.5 \cdot (F^T F - I)$$
3. Chrono’s ANCF Shell Element

- Gradient $r_z$, thick plate – integration over volume
- 4 Nodes
- Bilinear: Product of linear shape functions
  \[ s_1 = \frac{1}{4} (1 - \xi) (1 - \eta) \quad s_2 = \frac{1}{4} (1 + \xi) (1 - \eta) \]
  \[ s_3 = \frac{1}{4} (1 + \xi) (1 + \eta) \quad s_4 = \frac{1}{4} (1 - \xi) (1 + \eta) \]
- Suffers from locking if not alleviated
- Continuum-based approach
- Includes out-of-plane strains

Kinematics of the element:

\[ \mathbf{r}(\xi, \eta, t) = \underbrace{\mathbf{r}_m(\xi, \eta, t)}_{\text{Position of mid-plane}} + z \underbrace{\frac{\partial \mathbf{r}}{\partial z}(\xi, \eta, t)}_{\text{Position on shell thickness}} \]

3. Chrono’s ANCF Shell Element

Note that shape functions, position vector gradients, angles, transformation matrices, intermediate operations between frames of reference, and strains are adimensional. The position of an arbitrary point in the shell may be described as

\[ r^i(x^i, y^i, z^i) = S^i(x^i, y^i, z^i)e^i, \]

where the combined shape function matrix is given by \( S^i = [S^i_m \ z^iS^i_m] \). Similarly, the coordinates of the element may be grouped together as \( e^i = [(e^i_p)^T \ (e^i_g)^T]^T \), where \((e^i_p)\) \((e^i_g)\) are the element position and gradient coordinates.
3. Chrono’s ANCF Shell Element

Relying on this kinematic description of the shell element, the Green-Lagrange strain tensor may be calculated as

\[
\varepsilon^{i} = \frac{1}{2} \left( (F^{i})^T F^{i} - I \right),
\]

where \( F^{i} \) is the deformation gradient matrix defined as the current configuration over the reference configuration. Using the current absolute nodal coordinates, this matrix may be defined as (chain rule)

\[
F^{i} = \frac{\partial r^{i}}{\partial X^{i}} = \frac{\partial r^{i}}{\partial x^{i}} \left( \frac{\partial X^{i}}{\partial x^{i}} \right)^{-1}
\]

The strain tensor can then expressed in vector form in the following manner

\[
\varepsilon^{i} = \begin{bmatrix} \varepsilon_{xx}^{i} & \varepsilon_{yy}^{i} & \gamma_{xy}^{i} \\ \varepsilon_{zz}^{i} & \gamma_{xz}^{i} & \gamma_{yz}^{i} \end{bmatrix}^T
\]

Membrane strains
3. Chrono’s ANCF Shell Element

The elastic internal forces are spatially integrated over the element volume using Gaussian quadrature:

\[ Q_k^i = - \int_{V_0} \left( \frac{\partial \epsilon^c}{\partial \epsilon^i} \right) \frac{\partial W^i(\epsilon^c + \epsilon^{EAS})}{\partial \epsilon^c} dV_0 \]  \hspace{1cm} (1)

where \( \epsilon^c \) is the compatible strain, obtained from the displacement field using “Assumed Natural Strain” interpolation to avoid transverse/in-plane shear. Further, the term \( W^i(\epsilon^c + \epsilon^{EAS}) \) denotes the strain energy density function, which must be obtained by adding an enhanced strain contribution, \( \epsilon^{EAS} \). The second Piola–Kirchhoff stress tensor is obtained from the relation \( \sigma^i = \frac{\partial W^i(\epsilon^c + \epsilon^{EAS})}{\partial \epsilon^c} \).

The addition of assumed natural strains and enhanced strains finds justifications of the mixed variational principle by Hu–Washizu.

More on this later!
4. Shell strains: Orthotropic and curvilinear reference

Due to manufacturing processes, the initial configuration isn’t always “straight”; initial curved geometries of the flexible bodies need to be considered

$$\mathbf{F}^i = \frac{\partial \mathbf{r}^i}{\partial \mathbf{X}^i} = \frac{\partial \mathbf{r}^i}{\partial \mathbf{x}^i} \left( \frac{\partial \mathbf{X}^i}{\partial \mathbf{x}^i} \right)^{-1}$$

$\mathbf{r}^i$: Current configuration; $\mathbf{X}^i$: Initial configuration; $\mathbf{x}^i$: Element reference

The tensor $\mathbf{J}^i = \left( \frac{\partial \mathbf{X}^i}{\partial \mathbf{x}^i} \right) = \frac{\partial (\mathbf{S}^i_{0})}{\partial \mathbf{x}^i}$ is constant and can be inverted. The gradient tensor $\mathbf{F}^i$ defines strains in the global frame from a straight configuration as $\mathbf{\varepsilon}^i = 0.5 \cdot (\mathbf{F}^i \mathbf{T} \mathbf{F}^i - \mathbf{I})$
4. Shell strains: Orthotropic and curvilinear reference

In the curved initial configuration, the position of a material point in the shell is given by

$$ r_0(x, y, z) = r_{m0}(x, y) + zr_{z0}(x, y) $$

The element local coordinate system is a Lagrangian coordinate system in which strains are to be measured. This frame of reference defines covariant base vectors along the three curvilinear coordinate lines

$$(g_0)_1 = \frac{\partial r_0}{\partial x} = \frac{\partial r_{m0}}{\partial x}(x, y) + z \frac{\partial r_{0z}}{\partial x}(x, y)$$
$$(g_0)_2 = \frac{\partial r_0}{\partial y} = \frac{\partial r_{m0}}{\partial y}(x, y) + z \frac{\partial r_{0z}}{\partial y}(x, y)$$
$$(g_0)_3 = \frac{\partial r_0}{\partial z} = r_0(z)(x, y)$$
4. Shell strains: Orthotropic and curvilinear reference

The covariant base vector along the coordinate lines in the current configuration is given by

\[
\begin{align*}
(g)_1 &= \frac{\partial r}{\partial x} = \frac{\partial r_m}{\partial x}(x, y) + z \frac{\partial r_z}{\partial x}(x, y) \\
(g)_2 &= \frac{\partial r}{\partial y} = \frac{\partial r_m}{\partial y}(x, y) + z \frac{\partial r_z}{\partial y}(x, y) \\
(g)_3 &= \frac{\partial r}{\partial z} = r_z(x, y)
\end{align*}
\]

Current deformed reference

Each component of the covariant Green strain tensor in the \textit{curvilinear} system is defined as

\[
E_{IJ} = \frac{1}{2} \left( C_{IJ} - C_{IJ}^0 \right),
\]

where \( C_{IJ} = (g)_I \cdot (g)_J \) and \( C_{IJ}^0 = (g_0)_I \cdot (g_0)_J \)
4. Shell strains: Orthotropic and curvilinear reference

To define the Green strain tensor regularized to its orthonormal frame, we first orthonormalize the covariant base at the initial configuration

\[
(e_0)_1 = \frac{(g_0)_1}{|(g_0)_1|} \\
(e_0)_2 = (e_0)_3 \times (e_0)_1 \\
(e_0)_3 = |r_z|
\]

In which, theta is the angle \( \theta \) defines a principal direction in the material (used in tires: steel belts).

This frame defines the actual frame in which the material properties are defined. If the material is orthotropic, directions in the material can be considered. For example,

\[
(e_0)_1^{Or} = (e_0)_1 \cos \theta + (e_0)_2 \sin \theta \\
(e_0)_3^{Or} = (e_0)_3 \\
(e_0)_2^{Or} = -(e_0)_1 \sin \theta + (e_0)_2 \cos \theta
\]
4. Shell strains: Orthotropic and curvilinear reference

In matrix form, the coefficients of contravariance transformation may be obtained from the Jacobian of the position vectors at the reference configuration and the local Cartesian frame including anisotropy in the following form:

\[
\beta = \begin{bmatrix} Y^{-1}_{C1T} & \frac{\partial x}{\partial x} \\ Y^{-1}_{C2T} & (e_0)^{Or}_1 \\ Y^{-1}_{C3T} & (e_0)^{Or}_2 & (e_0)^{Or}_3 \end{bmatrix}
\]

where \( Y^{-1}_{Ci} \) is the \( i \) column of the inverse of \( Y = \frac{\partial x}{\partial x} = \begin{bmatrix} (g_0)^1_1 & (g_0)^1_2 & n_0 \end{bmatrix} \). The components of the 3-by-3 matrix \( \beta \) are used to set up a transformation matrix necessary for the calculation of strains:

\[
\beta = \begin{bmatrix} \beta_{11} & \beta_{12} & \beta_{13} \\ \beta_{21} & \beta_{22} & \beta_{23} \\ \beta_{31} & \beta_{32} & \beta_{33} \end{bmatrix}
\]

where \( \beta_{ij} = \beta(i, j) \).
4. Shell strains: Orthotropic and curvilinear reference

Finally the compatible strains are calculated as:

$$\varepsilon = \frac{1}{2} \beta^T \left( \begin{bmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{bmatrix} - \begin{bmatrix} (g_0)_{11} & (g_0)_{12} & (g_0)_{13} \\ (g_0)_{21} & (g_0)_{22} & (g_0)_{23} \\ (g_0)_{31} & (g_0)_{32} & (g_0)_{33} \end{bmatrix} \right) \beta =$$

where $g_{ij} = (g)_i \cdot (g)_j$ and $(g_0)_{ij} = (g_0)_i \cdot (g_0)_j$. Final expression! These strains are related to constitutive equations
4. Shell strains: Orthotropic and curvilinear reference

A few words about the derivations in previous slides

- It is general: No membrane-strain assumption, i.e. fully 3D
- Initial configuration must be accounted for in the numerical implementation
- Initial configuration must be considered when obtaining the generalized elastic/material forces
5. Generalized forces

The energy for a linear elastic material:

\[ U_e = \int_V \varepsilon^T \sigma \varepsilon \, dV_0 \Rightarrow Q_e = \frac{\partial U_e}{\partial \varepsilon} \]

Generalized internal forces...

\[ Q_e^i = \int_{V^i} \frac{\partial \varepsilon^T}{\partial \varepsilon} E^i \varepsilon^i \, dV_0^i = \int_{V^i} \frac{\partial \varepsilon^T}{\partial \varepsilon} \left( e_0^i, e^i, \beta^i \right) E^i \varepsilon^i \left( e_0^i, e^i, \beta^i \right) \, dV_0^i = \]

\[ \int_{V^i} \frac{\partial}{\partial \varepsilon} \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} & \varepsilon_{xy} & \varepsilon_{xz} & \varepsilon_{yz} \end{bmatrix} E^i \varepsilon^i \left( e_0^i, e^i, \beta^i \right) \, dV_0^i \]

where \( \sigma \) is the second Piola-Kirchhoff stress tensor, and \( dV_0^i \) is the (infinitesimal) element volume at the initial configuration.

- Integrals are integrated numerically using **Gauss points**
- **Order of integration** depends on order of integrand (in turn, depends on shape functions’)

Full and reduced
5. Generalized forces

\[
\frac{\partial \varepsilon}{\partial e} = \frac{1}{2} \beta^T \frac{\partial}{\partial e} \begin{bmatrix}
g_{11} & g_{12} & g_{13} 
g_{21} & g_{22} & g_{23} 
g_{31} & g_{32} & g_{33}
\end{bmatrix} \beta =
\]

\[
\frac{\partial \varepsilon}{\partial e} = \frac{1}{2} \beta^T \frac{\partial}{\partial e} \begin{bmatrix}
S_x^T S_x & S_y^T S_y & S_z^T S_z 
S_y^T S_x & S_y^T S_y & S_z^T S_z 
S_z^T S_x & S_z^T S_y & S_z^T S_z
\end{bmatrix} \beta =
\]

\[
\varepsilon_d = \frac{\partial \varepsilon}{\partial e} = \frac{1}{2} \beta^T \begin{bmatrix}
S_x^T S_x & S_y^T S_y & S_z^T S_z 
S_y^T S_x & S_y^T S_y & S_z^T S_z 
S_z^T S_x & S_z^T S_y & S_z^T S_z
\end{bmatrix} \beta'
\]

Beta matrices need manipulations
to accommodate strain vector
6. Jacobian of internal forces

Jacobian of internal forces is needed when implicit numerical integration is needed

- Computationally more demanding than internal forces
- Accurate enough approximation needed for convergence

\[ \varepsilon^T E \varepsilon \rightarrow \frac{\partial Q_\varepsilon}{\partial e} = \frac{\partial^2}{\partial e^2} \left( \varepsilon^T E \varepsilon \right) = \left( \varepsilon^T E \frac{\partial^2 \varepsilon^T}{\partial e^2} + \frac{\partial \varepsilon^T}{\partial e} E \frac{\partial \varepsilon}{\partial e} \right) \]

Integrand of elastic energy

\[ \text{Algebraic manipulations needed for this term} \]

- More details on this, in Chrono implementation

- \( E \) is a matrix of elastic coefficients that contain moduli of elasticity, rigidity, and Poisson ratios in the 3 directions: \( E_x, E_y, E_z, G_x, G_y, G_z, \nu_x, \nu_y, \nu_z \)
7. Mass matrix

The mass matrix of the element is given by

\[ M^i = \int_{V^i_o} \rho_0^i (S^i)^T S^i dV^i_o, \]

which remains constant throughout the simulation. The equations of motion may be written as

\[ M^i \ddot{e}^i = Q_k^i(e^i, \dot{e}^i, \alpha^i) + Q_e^i(e^i, \dot{e}^i, t), \]

where \( Q_k \) is the element elastic force vector and \( Q_e \) is the external force vector.
8. Generalized external forces: Point load

Point load:

- concentrated load
- acts on one finite element at any point
- does not require numerical integration

\[
\delta W_{cl} = F^T \delta r^P = Q_{cl}^T \delta e \\
Q_{cl} = S^T (\xi_P, \eta_P) F^P
\]
8. Generalized external forces: Pressure

Pressure:

- distributed load
- acts normal to the surface
- use Principle of Virtual Work to obtain generalized counterpart

\[ \mathbf{Q}_{\text{pres}} = - \int_A \mathbf{S}_e^T (\xi, \eta) p \mathbf{n} \det[\mathbf{J}] \, dA = \sum_{j=0}^{n_j} \sum_{i=0}^{n_i} w_i w_j \mathbf{S}_e^T (\xi_i, \eta_j) p \mathbf{n} (\xi_i, \eta_j) \det[\mathbf{J}] \]
8. Generalized external forces: Gravity load

Gravity load:
- volumetric, distributed load
- acts along a global direction
- do not depend on the finite element’s coordinates

\[ Q_{grav} = - \int_V \underbrace{S^T(\xi, \eta, \zeta)}_{\text{shape function}} \underbrace{\rho \ g}_{\text{density}} \underbrace{\text{det} \ [J]}_{\text{Det. of Jacobian of the transformation}} \ dV = \sum_{k=0}^{n_k} \sum_{j=0}^{n_j} \sum_{i=0}^{n_i} w_i w_j w_k S^T(\xi_i, \eta_j, \zeta_k) \rho g \text{det} \ [J] \]

Numerically solve the integral: Gauss Quadrature
9. Locking issues: Convergence

Figure 4. Numerical convergence with large deformation (initially flat)

<table>
<thead>
<tr>
<th>Dimensions</th>
<th>E (MPa)</th>
<th>G (MPa)</th>
<th>Density</th>
<th>Vertical Force</th>
<th>Simulation type</th>
<th>Ansys element</th>
<th>Converged disp</th>
</tr>
</thead>
<tbody>
<tr>
<td>1mx1mx0.01m</td>
<td>210</td>
<td>80.8</td>
<td>500 kg/m³</td>
<td>-50N</td>
<td>Dynamic</td>
<td>Shell181 (EAS)</td>
<td>-0.649m</td>
</tr>
</tbody>
</table>