Overview

Class FEMtoolkit contains the following procedures and functions:

- *import_abq(FileName)* – import of ABAQUS inp-file
- *export_abq(FileName)* – export of ABAQUS inp-file
- *import_ndload(FileName,LoadName)* – import values for nodes
- *export_ndload(FileName,LoadName)* – export values for nodes. A created text file contains rows with number of a node and a value divided by comma.
- *import_fcload(FileName,LoadName)* - import values for faces of elements
- *export_fcload(self,FileName,LoadName)* – export values for faces of elements. A created text file contains rows with number of an element, number of a face and a value divided by comma.
- *scale(Scale)* – scale the mesh
- *ExtractCoating(FileName,NsetName,EsetNames)* – create vtu-file with surface elements and values thickness of solid elements simulating coatings in the original mesh
- *CreateCoating(ThickNames,NSet)* – add solid elements to a mesh to simulate coating
- *map_surf(FileName,NodeSet,DistError=0.0001)* – mapping values from vtu-file with surface elements on a node set
- *SymmetryEquations(FileName,NSet1,NSet2,method='Tolerance',tolerance=(0.00001,0.00001))* – create a file with a section of equations simulating cyclic symmetry for ABAQUS inp-file
- *LinearVTUfile(FileName,Eset)* - create vtu-file for element set with values for nodes. The file can be used for mapping
- *mapping(FileName,NodeSet,Tlrnc=0.005)* - mapping values from vtu-file with solid elements on a node set
- *crack(NSet,ESet)* – separate a mesh in the node set
- *NodesCoord(FileName,NSet)* – writer file with coordinates for nodes from node set
- *ProjectNodesToSurf(PrjctNodes,SurfElems,SurfNodes)* – project nodes onto a surface defined by elements an nodes
- *DeleteElements(EsetNames)*- delete elements
- *CreateSubModel(CentralNodes,Radius)* – create submodel
- *ShiftSurf(D0,D1,Axis,ShiftDir,NodeSet)* – shift a surface

Mapping of 3D field onto a mesh

In general, applied fields are a result of calculations. The calculations are performed by means of finite element analysis often. In the case, a field can be stored as mesh of linear tetrahedrons with values in nodes. The field stored as a tetrahedron mesh can be applied by means of simple algorithm in comparison with a field stored as a list of values with coordinates.

Global and local coordinates are connected in compliance with the following equation for a tetrahedron:

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = \begin{bmatrix} x_2 - x_1 & x_3 - x_1 & x_4 - x_1 \\ y_2 - y_1 & y_3 - y_1 & y_4 - y_1 \\ z_2 - z_1 & z_3 - z_1 & z_4 - z_1 \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \\ \zeta \end{Bmatrix} + \begin{Bmatrix} x_1 \\ y_1 \\ z_1 \end{Bmatrix}$$

Where x_i, y_i, z_i – global coordinate of i-th node of the tetrahedron; x, y, z – global coordinates; ξ, η, ζ – local coordinates.

Thus, local coordinates can be found for any point (x, y, z) according to equation:

$$\begin{Bmatrix} \xi \\ \eta \\ \zeta \end{Bmatrix} = M^{-1} \left(\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} - \begin{Bmatrix} x_1 \\ y_1 \\ z_1 \end{Bmatrix} \right)$$

where

$$M = \begin{bmatrix} x_2 - x_1 & x_3 - x_1 & x_4 - x_1 \\ y_2 - y_1 & y_3 - y_1 & y_4 - y_1 \\ z_2 - z_1 & z_3 - z_1 & z_4 - z_1 \end{bmatrix}$$

The following conditions are fulfilled if a point (x, y, z) is placed inside the tetrahedron:

$\xi \geq 0$ $\eta \geq 0$ $\zeta \geq 0$ $\xi + \eta + \zeta \leq 1$

Value for the point inside the tetrahedron is calculated according to the local coordinates and values in tetrahedron nodes by means of the following equation:

$$Value = \begin{Bmatrix} 1 \\ \xi \\ \eta \\ \zeta \end{Bmatrix}^T \begin{Bmatrix} V_1 \\ V_2 - V_1 \\ V_3 - V_1 \\ V_4 - V_1 \end{Bmatrix}$$

Where V_i – value in i-th node of tetrahedron.

If the point is outside of all tetrahedrons, the tool applies a value from the nearest node to the point.

In general, the mapping process consists of two steps. At the first step, a tetrahedron linear mesh with values should be created. The tool can create vtu-file with values for a defined set of elements. For instance, there are mesh file *800B_Blade1_Baseline_Mesh.inp* and a file with temperature for nodes (every row contains number of a node and temperature divided by comma) *New_Thg_correction_Ver1_Temperature.txt*. In the case, vtu-file with temperature for element set 'Blade' can be created by the following commands:

model1=FEMtoolkit()

model1.import_abq('800B_Blade1_Baseline_Mesh.inp')

*model1.import_ndload('New_Thg_correction_Ver1_Temperature.txt', **'Temp'**)*

*model1.LinearVTUfile(**'Temperature.vtu'**, 'BLADE')*

At the second stage, a field is mapped onto a mesh from vtu-file and dat-file having values for node of the mesh is recorded. The example with mapping of temperature follows:

model2=FEMtoolkit()

model2.import_abq('GT800B_Blade1_base.inp')

*model2.mapping(**'Temperature.vtu'**, 'NALL')*

*model2.export_ndload('Temp_map.dat', **'Temp'**)*

The function *LinearVTUfile* can be used to check out in Paraview how the field has been applied to the mesh. Enter the following command and open the created file in the Paraview:

*model2.LinearVTUfile(**'Temperature_ch.vtu'**, EALL')*

Submodeling

The tool allows to separate elements, nodes and loads (pressure and temperature) from a global modal around nodes inside spheres defined by a list of nodes and a radius.

The example follows. The global model is *SGT800B_Blade1_ver3.inp*. A submodel is created for nodes *207848,209960,1227246,1228785* with radius 0.003. Temperature and pressure are extracted for the submodel:

model=FEMtoolkit()

model.import_abq('SGT800B_Blade1_ver3.inp')

model.CreateSubModel((207848,209960,1227246,1228785), 0.003)

model.export_abq('FRANC3Dsbml.inp')

model.import_ndload('SGT-800B_Blade1_AM_Tip_Model3_Temperature_blade.txt','Temp')

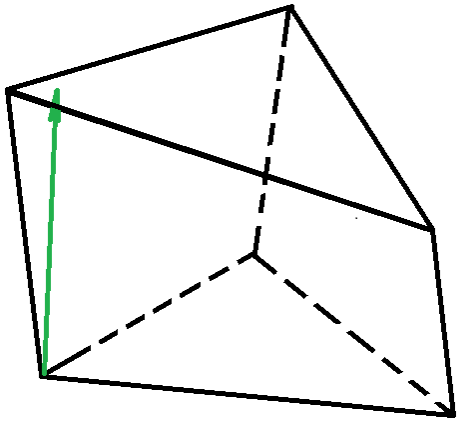
model.export_ndload('Submodel_Temp.inc','Temp')

model.import_fcload('SGT-800B_Blade1_AM_Tip_Model3_Pressure_blade.txt','Press')

model.export_fcload('Submodel_Press.inc','Press')

Coating thickness extraction

If coating is made of linear prism elements for tetrahedral basic mesh the thickness can be extracted as height of prism elements at nodes.



vector normal to the bottom triangle of the prims:

$$\bar{n}_b = \acute{D}\begin{Bmatrix}(y_2-y_1)(z_3-z_1)-(y_3-y_1)(z_2-z_1)\\(x_3-x_1)(z_2-z_1)-(x_2-x_1)(z_3-z_1)\\(x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)\end{Bmatrix}$$

Then coordinates of projection of i-th nodes on the top surface of the prism:

$$\bar{x}_p = \bar{x}_i + \bar{n}_b$$

The top plane of the prism is described by the equation: $\bar{n}_t \cdot \bar{x} = b$

where

$$\bar{n}_t = \begin{Bmatrix}(y_5-y_4)(z_6-z_4)-(y_6-y_4)(z_5-z_4)\\(x_6-x_4)(z_5-z_4)-(x_5-x_4)(z_6-z_4)\\(x_5-x_4)(y_6-y_4)-(x_6-x_4)(y_5-y_4)\end{Bmatrix}$$

$$b = \begin{Bmatrix}(y_5-y_4)(z_6-z_4)-(y_6-y_4)(z_5-z_4)\\(x_6-x_4)(z_5-z_4)-(x_5-x_4)(z_6-z_4)\\(x_5-x_4)(y_6-y_4)-(x_6-x_4)(y_5-y_4)\end{Bmatrix}^T \begin{Bmatrix}x_4\\y_4\\z_4\end{Bmatrix}$$

Thus

$$\bar{n}_t \cdot \bar{n}_b = b - \bar{n}_t \cdot \bar{x}_i$$

$$\acute{D}\begin{Bmatrix}(y_5-y_4)(z_6-z_4)-(y_6-y_4)(z_5-z_4)\\(x_6-x_4)(z_5-z_4)-(x_5-x_4)(z_6-z_4)\\(x_5-x_4)(y_6-y_4)-(x_6-x_4)(y_5-y_4)\end{Bmatrix}^T\begin{Bmatrix}(y_2-y_1)(z_3-z_1)-(y_3-y_1)(z_2-z_1)\\(x_3-x_1)(z_2-z_1)-(x_2-x_1)(z_3-z_1)\\(x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)\end{Bmatrix}=$$

$$=\begin{Bmatrix}(y_5-y_4)(z_6-z_4)-(y_6-y_4)(z_5-z_4)\\(x_6-x_4)(z_5-z_4)-(x_5-x_4)(z_6-z_4)\\(x_5-x_4)(y_6-y_4)-(x_6-x_4)(y_5-y_4)\end{Bmatrix}^T\begin{Bmatrix}x_4-x_i\\y_4-y_i\\z_4-z_i\end{Bmatrix}$$

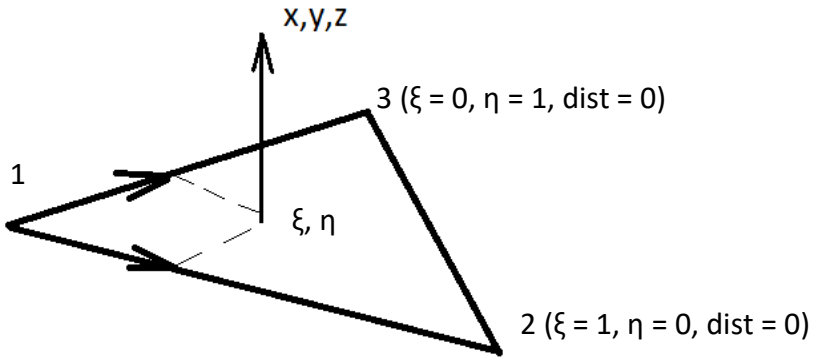
Algorithm for surface mapping

Plane for a triangle

Equation for a plane: $\bar{n}\cdot\bar{x}=b$

$$\bar{n}=\begin{Bmatrix}(y_2-y_1)(z_3-z_1)-(y_3-y_1)(z_2-z_1)\\(x_3-x_1)(z_2-z_1)-(x_2-x_1)(z_3-z_1)\\(x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)\end{Bmatrix}$$

$$b=\begin{Bmatrix}(y_2-y_1)(z_3-z_1)-(y_3-y_1)(z_2-z_1)\\(x_3-x_1)(z_2-z_1)-(x_2-x_1)(z_3-z_1)\\(x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)\end{Bmatrix}^T\begin{Bmatrix}x_1\\y_1\\z_1\end{Bmatrix}$$



$$\begin{Bmatrix}x\\y\\z\end{Bmatrix}=\begin{bmatrix}x_2-x_1 & y_3-y_1 & \frac{1}{|\bar{n}|}\left((y_2-y_1)(z_3-z_1)-(y_3-y_1)(z_2-z_1)\right)\\y_2-y_1 & y_3-y_1 & \frac{1}{|\bar{n}|}\left((x_3-x_1)(z_2-z_1)-(x_2-x_1)(z_3-z_1)\right)\\z_2-z_1 & z_3-z_1 & \frac{1}{|\bar{n}|}\left((x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)\right)\end{bmatrix}\begin{Bmatrix}\xi\\\eta\\dist\end{Bmatrix}+\begin{Bmatrix}x_1\\y_1\\z_1\end{Bmatrix}$$

$$\begin{Bmatrix} \xi \\ \eta \\ dist \end{Bmatrix} = \begin{bmatrix} x_2 - x_1 & y_3 - y_1 & \frac{1}{|\bar{n}|}\big((y_2 - y_1)(z_3 - z_1) - (y_3 - y_1)(z_2 - z_1)\big) \\ y_2 - y_1 & y_3 - y & \frac{1}{|\bar{n}|}\big((x_3 - x_1)(z_2 - z_1) - (x_2 - x_1)(z_3 - z_1)\big) \\ z_2 - z_1 & z_3 - z_1 & \frac{1}{|\bar{n}|}\big((x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)\big) \end{bmatrix}^{-1} \left(\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} - \begin{Bmatrix} x_1 \\ y_1 \\ z_1 \end{Bmatrix} \right)$$

$$V(\xi, \eta) = [V_2 - V_1 \quad V_3 - V_1] \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} + V_1$$