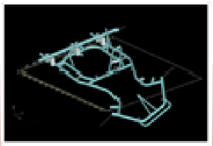
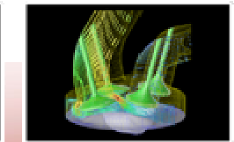
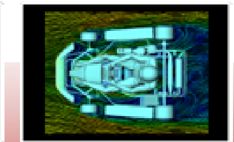
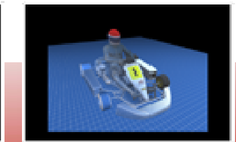
Università degli Studi di Roma Tor Vergata
Corso di Laurea e Dipartimento di Ingegneria Meccanica

tor vergata karting
www.torvergata-karting.it

Dynamic mesh: beyond Fluent

Marco Evangelos Biancolini
Department of Mechanical Engineering
University of Rome "Tor Vergata"



Outline

- Why use smoothing
- Review of methods for mesh smoothing
- Pseudo solid approach, FEM solver
- Radial basis approach
- Example 2D
- Example 3D

Why?

- Smoothing means that mesh topology is preserved: no new mesh required!
- Shape optimisation
- Moving or deforming boundaries (Fluid structure interaction, rigid body movements, morphing)
- Parametric analysis

Review of scientific contributions

- 198 relevant publications in the last 7 years (keyword “moving mesh method”) scanned
- 25 fully examined
- Strategies:
  - Laplacian smoothing
  - Spring models (rotational spring correction)
  - Pseudo solid (variable moduli, fictitious loads correction)
  - Boundary methods
  - Others...

Selected papers…

- **AUTOMATIC MESH MOTION FOR THE UNSTRUCTURED FINITE VOLUME METHOD** *Hrvoje Jasak, ˇZeljko Tukovi´c* ISSN 1333–1124
- **An adaptive mesh rezoning scheme for moving boundary flows and fluid–structure interaction** *Computers and Fluids* Volume: 36, Issue: 1, January, 2007, pp. 77-91 Masud, Arif; Bhanabhagvanwala, Manish; Khurram, Rooh A.
- **A three-dimensional torsional spring analogy method for unstructured dynamic meshes** *Computers and Structures* Volume: 80, Issue: 3-4, February, 2002, pp. 305-316 Degand, Christoph; Farhat, Charbel
- **Mesh deformation using radial basis functions for gradient-based aerodynamic shape optimization** *Computers and Fluids* Volume: 36, Issue: 6, July, 2007, pp. 1119-1136 Jakobsson, S.; Amoignon, O.
- **Mesh deformation based on radial basis function interpolation** *Computers and Structures* Volume: 85, Issue: 11-14, June - July, 2007, pp. 784-795 de Boer, A.; van der Schoot, M.S.; Bijl, H.
- **Finite element mesh update methods for fluid–structure interaction simulations** *Finite Elements in Analysis and Design* Volume: 40, Issue: 9-10, June, 2004, pp. 1259-1269 Xu, Zhenlong; Accorsi, Michael

Pseudo solid approach

- The mesh is supposed to be a deformable structure
- Elastic moduli can be chosen on an element by element basis to minimise distortion
- Corrective loads are added to improve element quality after mesh deformation
- One step method: Masud strategy in which stiffness is proportional to element area (volume)
- Two step methods: stiffness correction and extra loads added to preserve elements shape

Pseudo solid approach

- Element distortion estimator: ratio between inner and outer circle radii (for triangles)
- Advantages
  - Very good results optimising element stiffness
  - Physical pseudo solid guarantees mesh compatibility
  - Large motions can be handled by means of non linear FEM modelling
- Pitfalls:
  - Large problem solution
  - Large memory usage (can be alleviated by means of explicit solver)
  - Non conformal meshes, connectors and polyedra requires special treatment

Implementation details (2D)

- Node positions of the original mesh read from a nastran data file (GRID, CTRIA3, PSHELL, MAT1, SPC and SPCD entries are processed)
- Processing
  - Stiffness matrix is computed for each element
  - DOF are numbered and stored in ID and LM matrixes
  - Augmented stiffness matrix is assembled
  - Linear system is solved
- Checking
  - Residual is calculated
  - Comparison with nastran solution
- Deformed grid is represented by a scatter plot
- Element distortion is calculated

Implementation details (2D)

- Material stiffness matrix
- Element area
- Interpolation matrix
- Stiffness matrix

$$Q(E,\nu) := \frac{E}{1-\nu^2}\cdot\begin{pmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{pmatrix}$$

$$At(x1,y1,x2,y2,x3,y3) := \frac{(x2-x1)\cdot(y3-y1)-(x3-x1)\cdot(y2-y1)}{2}$$

$$B(x1,y1,x2,y2,x3,y3) := \frac{1}{2\cdot At(x1,y1,x2,y2,x3,y3)}\cdot\begin{pmatrix} y2-y3 & 0 & y3-y1 & 0 & y1-y2 & 0 \\ 0 & x3-x2 & 0 & x1-x3 & 0 & x2-x1 \\ x3-x2 & y2-y3 & x1-x3 & y3-y1 & x2-x1 & y1-y2 \end{pmatrix}$$

$$kt(x1,y1,x2,y2,x3,y3,E,\nu,t) := t\cdot At(x1,y1,x2,y2,x3,y3)\cdot B(x1,y1,x2,y2,x3,y3)^T\cdot Q(E,\nu)\cdot B(x1,y1,x2,y2,x3,y3)$$

Implementation details (2D)

- Elemental stiffness matrixes calculation
- Material topology extracted form database read in the nastran file

$$
\begin{aligned}
\text{Kelementi} := \text{for } \ & \text{iel} \in 1..\text{Nelementi} \\
& \text{isez} \leftarrow \text{elementi}_{\text{iel},2} \\
& \text{imat} \leftarrow \text{sezioni}_{\text{isez},2} \\
& E \leftarrow \text{materiali}_{\text{imat},2} \\
& G \leftarrow \text{materiali}_{\text{imat},3} \\
& \nu \leftarrow \text{materiali}_{\text{imat},4} \\
& t \leftarrow \text{sezioni}_{\text{isez},3} \\
& \begin{pmatrix} x1 \\ y1 \end{pmatrix} \leftarrow \text{submatrix}\left(\text{nodi}, \text{elementi}_{\text{iel},3}, \text{elementi}_{\text{iel},3}, 2, 3\right)^T \\
& \begin{pmatrix} x2 \\ y2 \end{pmatrix} \leftarrow \text{submatrix}\left(\text{nodi}, \text{elementi}_{\text{iel},4}, \text{elementi}_{\text{iel},4}, 2, 3\right)^T \\
& \begin{pmatrix} x3 \\ y3 \end{pmatrix} \leftarrow \text{submatrix}\left(\text{nodi}, \text{elementi}_{\text{iel},5}, \text{elementi}_{\text{iel},5}, 2, 3\right)^T \\
& \text{Kelementi}_{\text{iel}} \leftarrow \text{kt}(x1, y1, x2, y2, x3, y3, E, \nu, t)
\end{aligned}
$$

Implementation details (2D)

- Linear system is assembled

$$
Kaug := \left|
\begin{array}{l}
Kaug_{Ndof,\,Ndof} \leftarrow 0 \\
\text{for } iel \in 1..\,Nelementi \\
\quad \text{for } irow \in 1..\,6 \\
\quad\quad \left|
\begin{array}{l}
IDrow \leftarrow \left|LM_{iel,\,irow}\right| \\
\text{for } icol \in 1..\,6 \\
\quad \left|
\begin{array}{l}
IDcol \leftarrow \left|LM_{iel,\,icol}\right| \\
Kaug_{IDrow,\,IDcol} \leftarrow Kaug_{IDrow,\,IDcol} + \left(Kelementi_{iel}\right)_{irow,\,icol}
\end{array}
\right.
\end{array}
\right. \\
out \leftarrow Kaug
\end{array}
\right.
$$

- partitioned

$$Ktt := submatrix(Kaug, 1, NdofA, 1, NdofA)$$

$$Kut := submatrix(Kaug, NdofA + 1, Ndof, 1, NdofA)$$

$$Kuu := submatrix(Kaug, NdofA + 1, Ndof, NdofA + 1, Ndof)$$

$$Ktu := submatrix(Kaug, 1, NdofA, NdofA + 1, Ndof)$$

Implementation details (2D)

- Linear system solution

$$\Delta t := \mathrm{lsolve}\left(Ktt, Pt - Ktu \cdot \Delta u\right)$$

$$Ru := Kut \cdot \Delta t + Kuu \cdot \Delta u$$

$$P := \mathrm{stack}(Pt, Ru)$$

- Complete solution is built

$$\Delta := \mathrm{stack}\left(\Delta t, \Delta u\right)$$

Radial basis approach

- Born as a data interpolation procedure for scattered data
- Meshless field functions are defined
- A set of sources point is defined (may be mapped everywhere in the domain)
- A radial function is defined (compact support, or defined everywhere)
- Polynomial corrector is added to guarantee compatibility for rigid modes
- A linear system (order equal to the number of source point introduced) is solved for coefficients calculation
- The motion of an arbitrary point inside or outside the domain (interpolation/extrapolation) is expressed as the summation of the radial contribution of each source point (if the point falls inside the influence domain)

Radial basis approach

- We look for an interpolation function composed by a radial basis and a polynomial
- Typical radial functions are reported in the table

$$s(\boldsymbol{x}) = \sum_{l=1}^{N} \gamma_l \phi(\|\boldsymbol{x} - \boldsymbol{x}_l\|) + h(\boldsymbol{x}$$

Common radial basis functions

Radial basis function	$\phi(r)$
Spline type (R_n)	$\|r\|^n$, n odd
Thin plate spline (TPS$_n$)	$\|r\|^n \log \|r\|$, n even
Multiquadric (MQ)	$\sqrt{1+r^2}$
Inverse multiquadric (IMQ)	$\frac{1}{\sqrt{1+r^2}}$
Inverse quadratic (IQ)	$\frac{1}{1+r^2}$
Gaussian (GS)	e^{-r^2}

Radial basis approach

- Basis coefficients are obtained imposing the desired function values at source points
- Furthermore the polynomial terms has to give 0 contributions at source points

$$s(\boldsymbol{x}_{k_i}) = g(\boldsymbol{x}_{k_i}), \qquad 1 \leqslant i \leqslant N$$

$$0 = \sum_{i=1}^{N} \gamma_i q(\boldsymbol{x}_{k_i})$$

Radial basis approach

- Defining the following
  - Interpolation matrix
  - Constraint matrix
- We can describe previous condition as a linear system

$$M_{ij} = \phi(\|\boldsymbol{x}_{k_i} - \boldsymbol{x}_{k_j}\|), \quad 1 \leqslant i, \; j \leqslant N$$

$$\mathbf{P} = \begin{pmatrix} 1 & x^0_{k_1} & y^0_{k_1} & z^0_{k_1} \\ 1 & x^0_{k_2} & y^0_{k_2} & z^0_{k_2} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x^0_{k_N} & y^0_{k_N} & z^0_{k_N} \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{M} & \mathbf{P} \\ \mathbf{P}^{\mathrm{T}} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\beta} \end{pmatrix} = \begin{pmatrix} \mathbf{g} \\ 0 \end{pmatrix}$$

Radial basis approach

- Applying the method for each of the three displacements functions we obtain the interpolation field

$$v_x = s_x(\boldsymbol{x}) = \sum_{i=1}^{N} \gamma_i^x \phi(\|\boldsymbol{x} - \boldsymbol{x}_{k_i}\|) + \beta_1^x + \beta_2^x x + \beta_3^x y + \beta_4^x z,$$

$$v_y = s_y(\boldsymbol{x}) = \sum_{i=1}^{N} \gamma_i^y \phi(\|\boldsymbol{x} - \boldsymbol{x}_{k_i}\|) + \beta_1^y + \beta_2^y x + \beta_3^y y + \beta_4^y z,$$

$$v_x = s_z(\boldsymbol{x}) = \sum_{i=1}^{N} \gamma_i^z \phi(\|\boldsymbol{x} - \boldsymbol{x}_{k_i}\|) + \beta_1^z + \beta_2^z x + \beta_3^z y + \beta_4^z z,$$

Implementation details (2D)

- Node positions of the original mesh read from a nastran data file (only GRID, SPC and SPCD entries are processed)
- Processing
  - Constrained nodes are used as source points
  - Imposed displacements are used for mesh deformation
  - Basis generation
  - Calculation of displacements
- Field functions fro x and y displacements can be visualized by a contour or a surface plot
- Deformed grid is represented by a scatter plot

Implementation details (2D)

- Source points are first extracted
- Displacement functions values at source points are extracted
- Radial function is defined

$$
\begin{aligned}
x_{.k} := \text{for } i \in 1..\text{rows}(\text{vincoli}) \\
&\quad \text{INOD} \leftarrow \text{vincoli}_{i,1} \\
&\quad X_i \leftarrow \text{submatrix}(\text{nodi}, \text{INOD}, \text{INOD}, 2, 3)^T
\end{aligned}
$$

$$
\begin{aligned}
g_{.k} := \; & \text{ID} \leftarrow 1 \\
& \text{for } i \in 1..\text{rows}(\text{SPCall}) \\
& \quad \text{if } \text{SPCall}_{i,1} = 1 \lor \text{SPCall}_{i,2} = 1 \\
& \quad\quad G_{\text{ID},1} \leftarrow \text{SPCall}_{i,3} \\
& \quad\quad G_{\text{ID},2} \leftarrow \text{SPCall}_{i,4} \\
& \quad\quad \text{ID} \leftarrow \text{ID} + 1 \\
& \text{OUT} \leftarrow G
\end{aligned}
$$

$$
\phi(r) := \frac{1}{1 + r^2}
$$

Implementation details (2D)

- Interpolation matrix is calculated

$$
M := \text{for } i \in 1..\text{rows}(x_{.k}) \;\; \text{for } j \in 1..\text{rows}(x_{.k}) \;\; M_{i,j} \leftarrow \phi\left(\left|x_{.k_i} - x_{.k_j}\right|\right)
$$

- Constraint matrix is calculated

$$
P := \text{for } i \in 1..\text{rows}(x_{.k}) \;\; \left| \begin{array}{l} P_{i,1} \leftarrow 1 \\ \text{for } j \in 1..\text{rows}\left(x_{.k_1}\right) \\ \quad P_{i,j+1} \leftarrow \left(x_{.k_i}\right)_j \end{array} \right.
$$

Implementation details (2D)

- Linear system is assembled and solved

$$MP := \mathrm{augment}\left(\mathrm{stack}\left(M, P^T\right), \mathrm{stack}(P, zero)\right)$$

$$\gamma\beta_{ic} := \mathrm{lsolve}\left(MP, \mathrm{stack}\left(g_{.k}^{\langle ic \rangle}, zero^{\langle 1 \rangle}\right)\right)$$

- Interpolation coefficients are extracted

$$\gamma_{ic} := \mathrm{submatrix}\left(\gamma\beta_{ic}, 1, \mathrm{rows}\left(x_{.k}\right), 1, 1\right)$$

$$\beta_{ic} := \mathrm{submatrix}\left(\gamma\beta_{ic}, \mathrm{rows}\left(x_{.k}\right) + 1, \mathrm{rows}\left(\gamma\beta_{ic}\right), 1, 1\right)$$

- Interpolation function is defined

$$s_{.xy}(x) := \sum_{i=1}^{\mathrm{rows}\left(x_{.k}\right)} \left[\begin{bmatrix} \left(\gamma_1\right)_i \\ \left(\gamma_2\right)_i \end{bmatrix} \cdot \phi\left(\left| x - x_{.k_i} \right|\right) \right] + \begin{pmatrix} \beta_1 \cdot \mathrm{stack}(1, x) \\ \beta_2 \cdot \mathrm{stack}(1, x) \end{pmatrix}$$

Implementation details (3D)

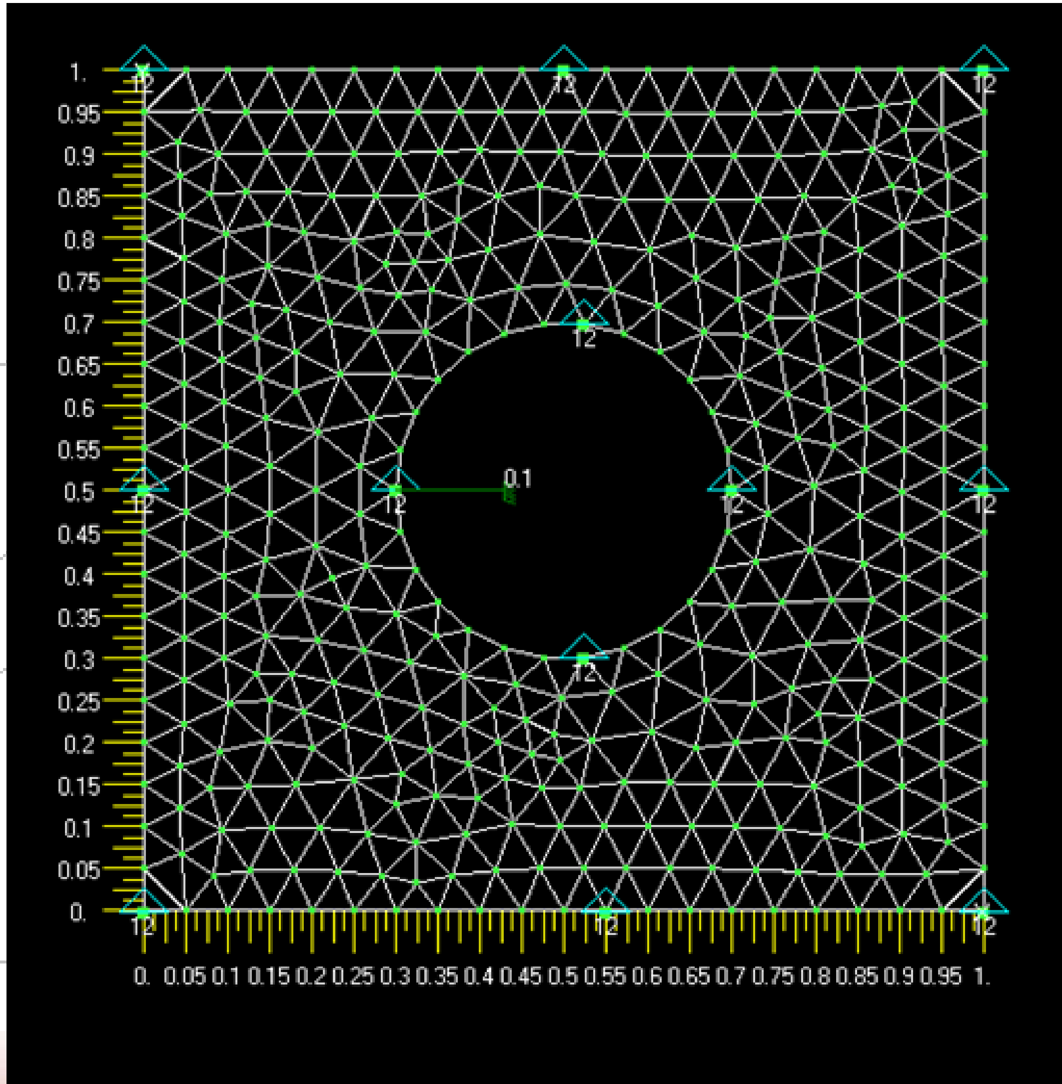
- Node positions of the original mesh read from a nastran data file
- Processing
  - Constrained nodes are used as source points
  - Imposed displacements are used for mesh deformation
  - Basis generation
  - Calculation of displacements
- Nodal displacements are written in standard nastran file *.f06
- Post processing by FEMAP

Radial basis approach

- Advantages
  - Meshless method only grid points are moved regardless of element connected
  - Suitable for parallel implementation
  - Small systems have to be solved (sparse if the compact support is chosen)
  - Fast
- Pitfalls
  - Reversed elements can arise
  - Boundary tracking requires special effort if not all boundary grids are included as source points

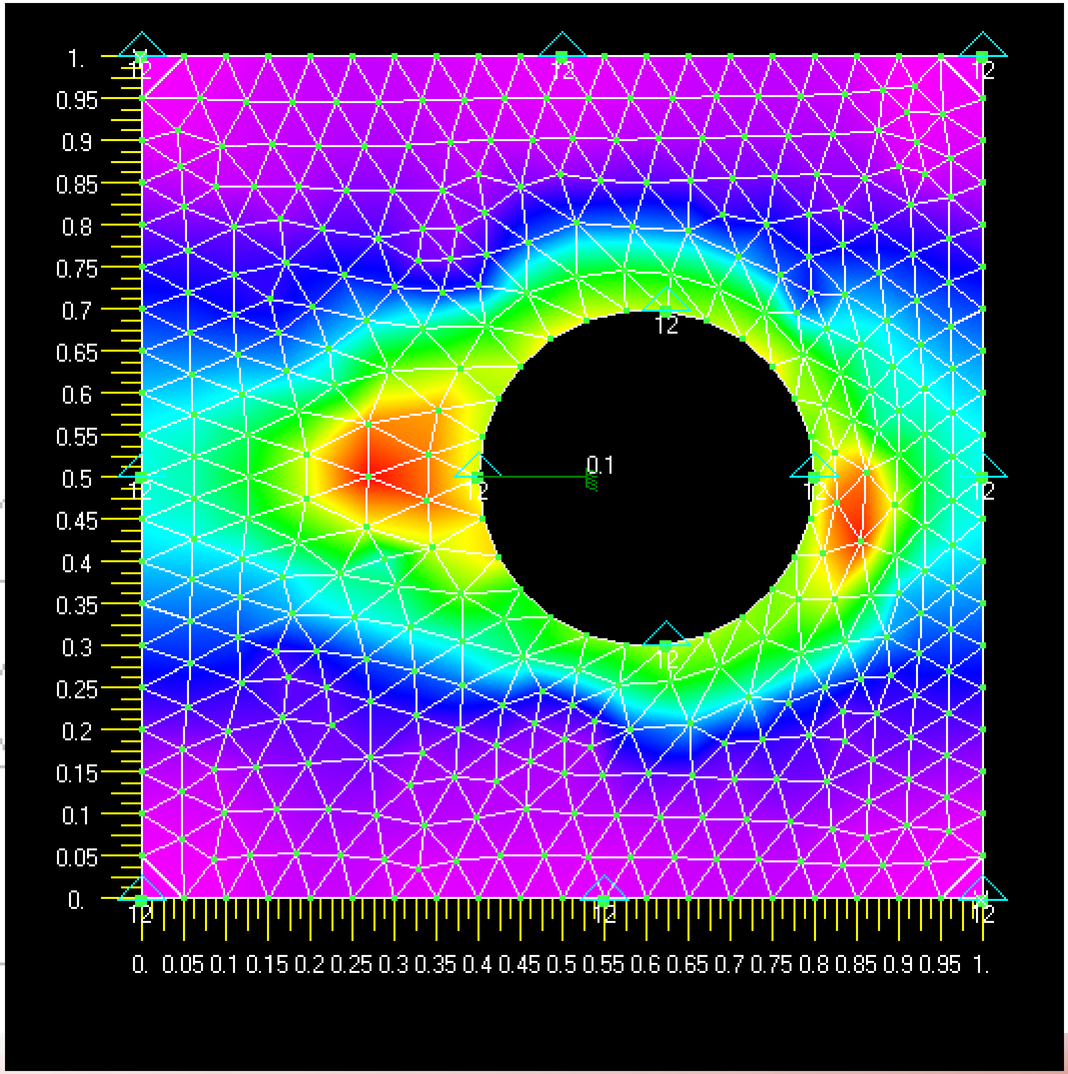
Example 2D: moving hole

- FEM reference solution by Nastran
- Displacement BC on the boundaries
- Constant strain CTRIA elements



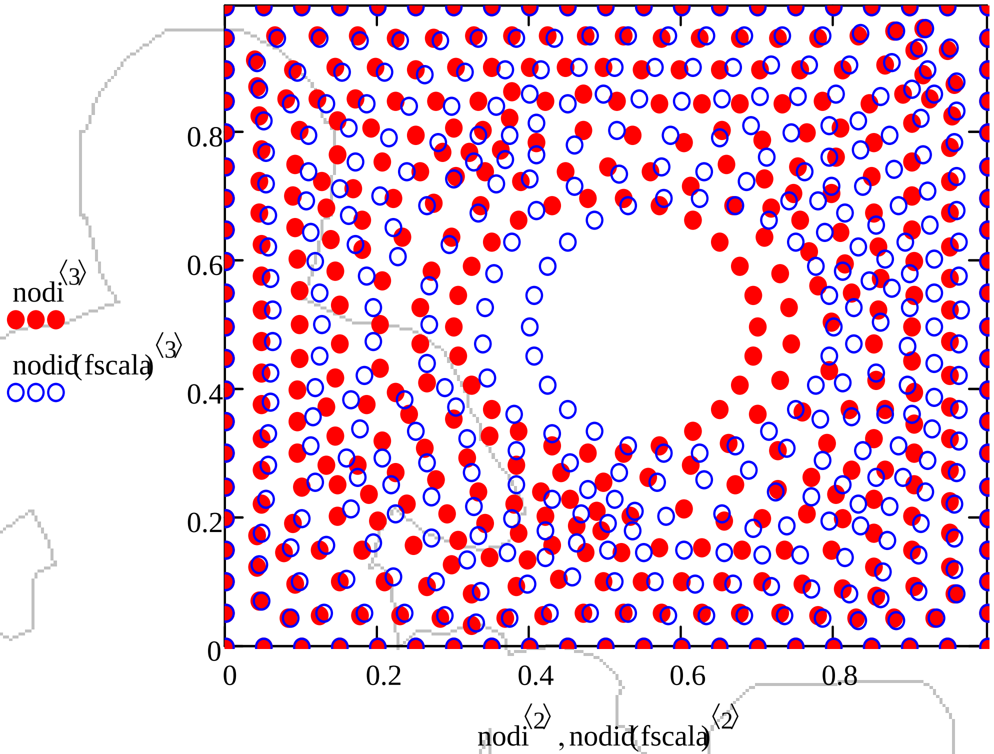
Example 2D: moving hole

- Deformed mesh
- Strain energy density



Example 2D: moving hole

- Mathcad solution
- FEM

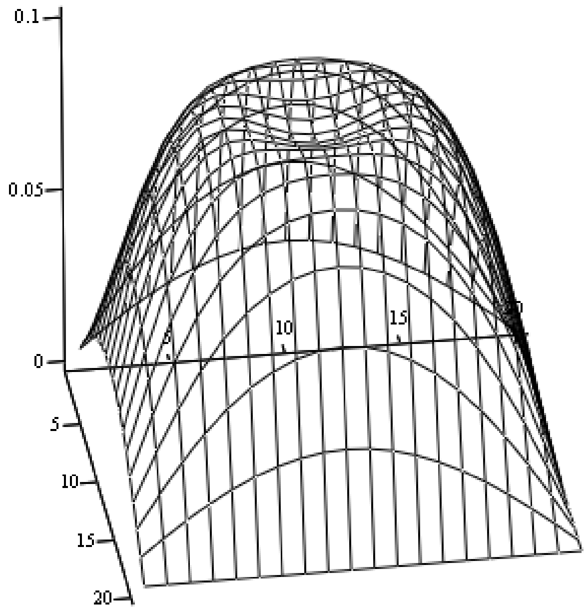


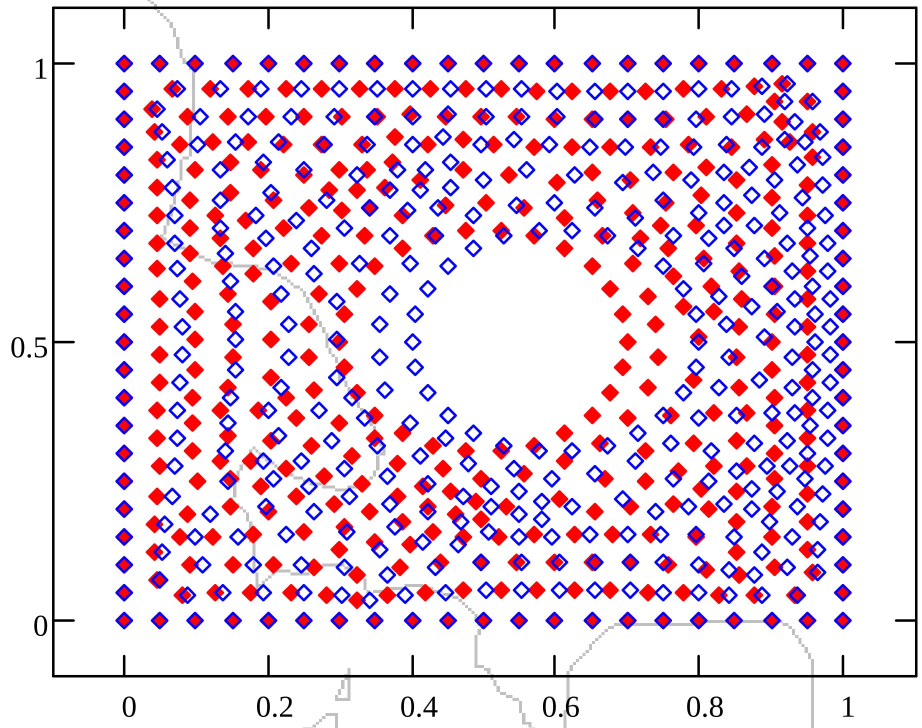
Max difference (Nastran – proposed FEM)

$$\max\left(NASTRAN^{T} - TT\right) = 2.7217 \times 10^{-6}$$

Example 2D: moving hole

- Mathcad solution
- Radial basis

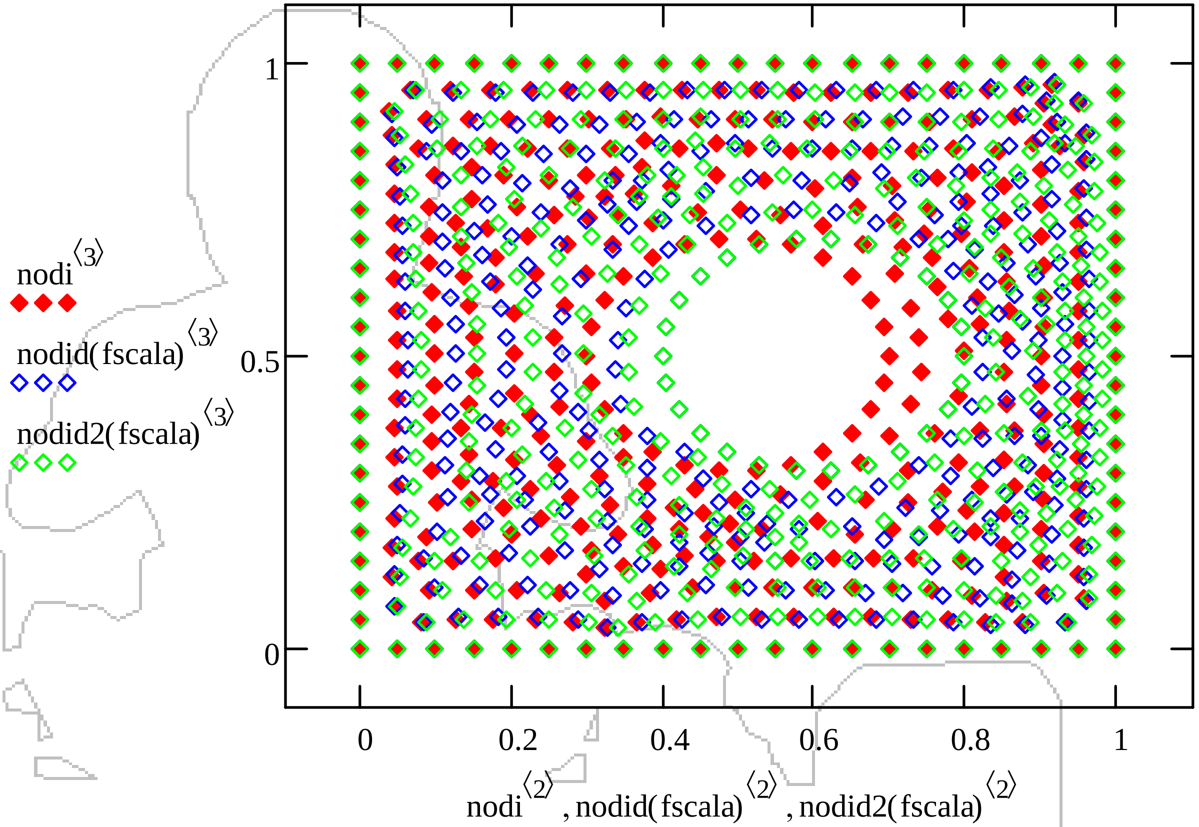




mplot

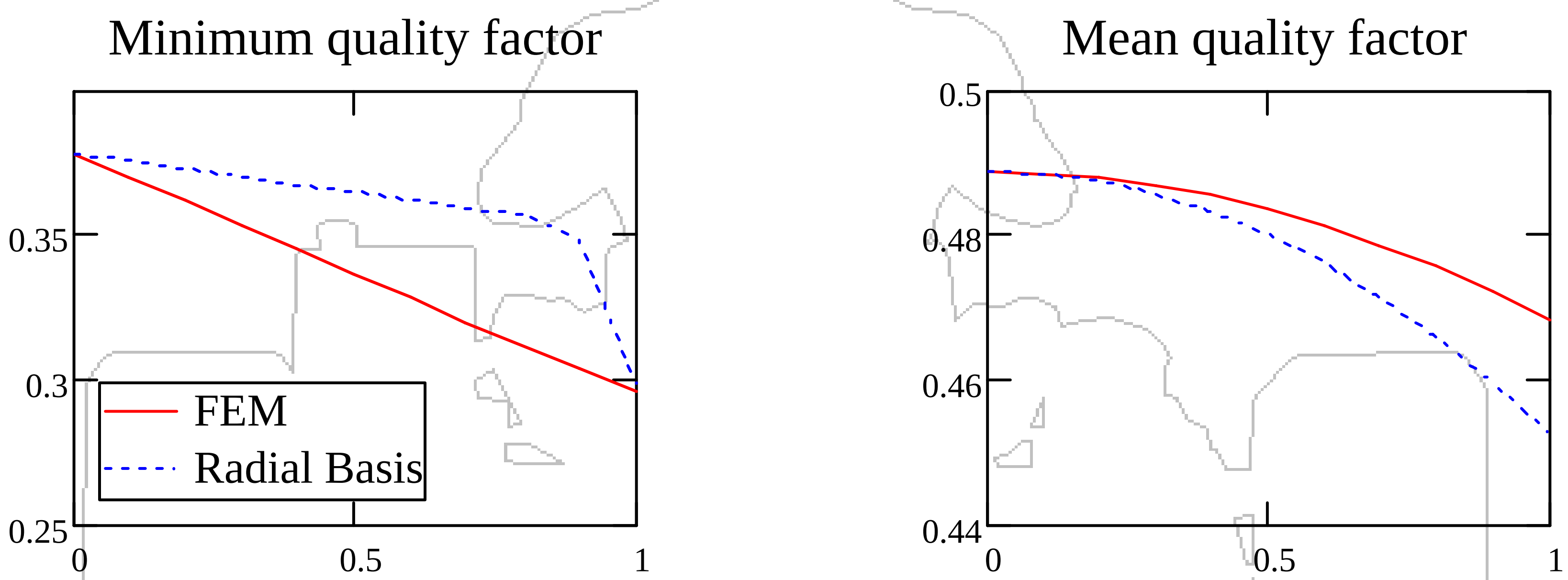
Example 2D: moving hole

- Mathcad solution
- FEM (blue)
- Radial basis (green)



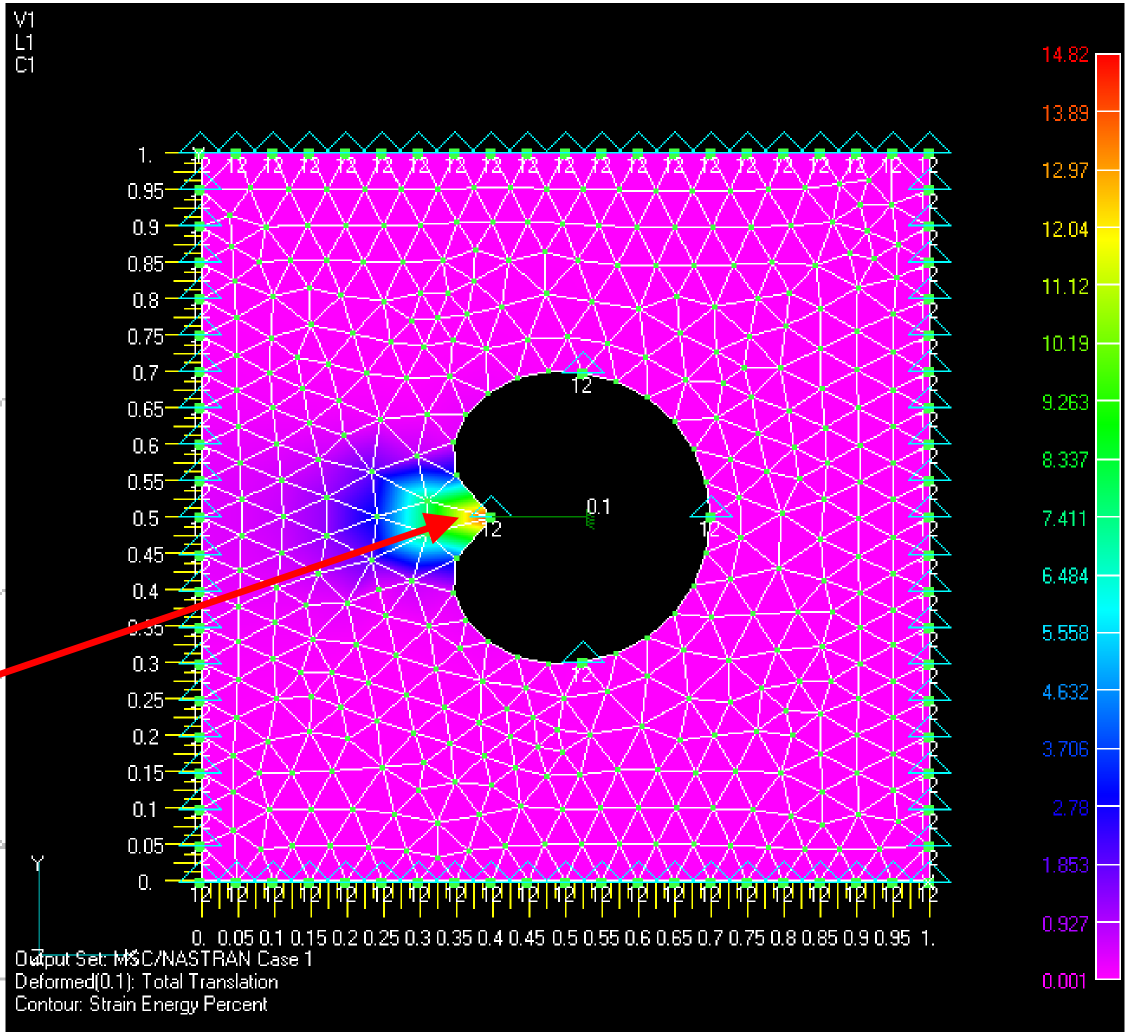
Example 2D: moving hole

- Mathcad solution
- Mesh quality comparison changing deformation extent



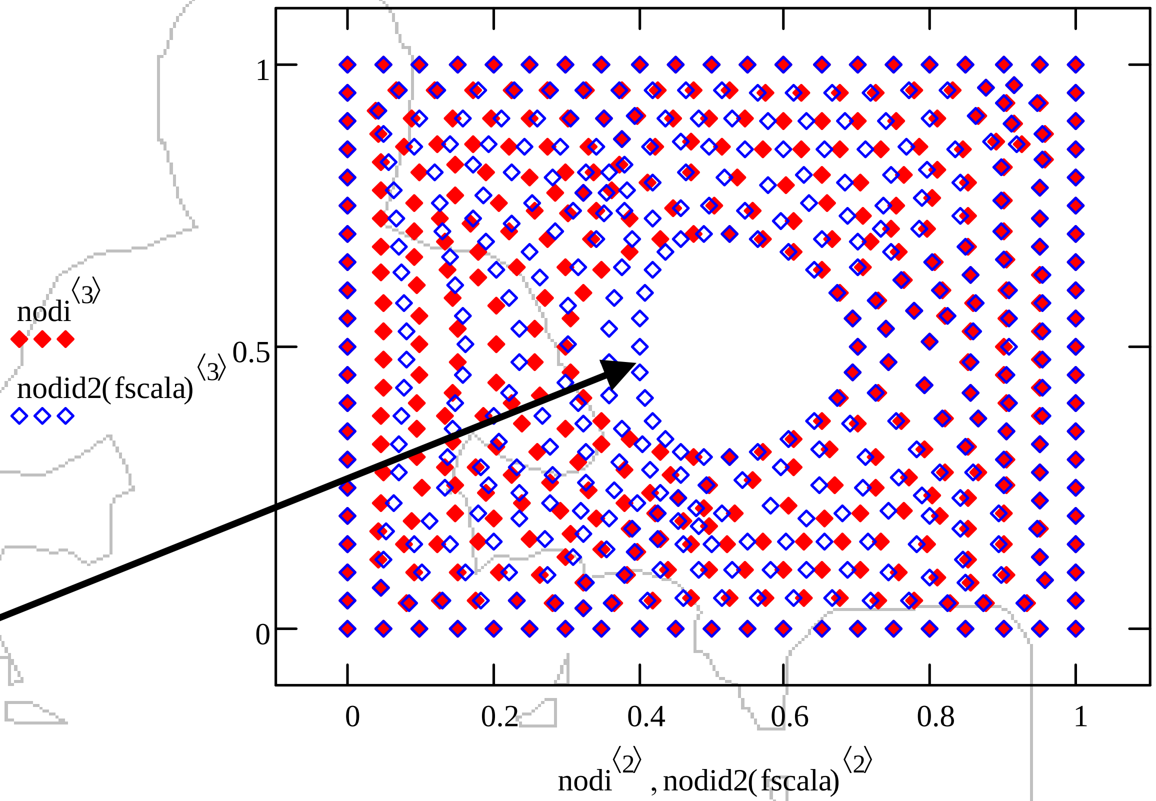
Example 2D: moving hole, boundary points

- Nastran solution
- 4 boundary points on the hole
- External boundary
- Note the sharp deformation of the boundary



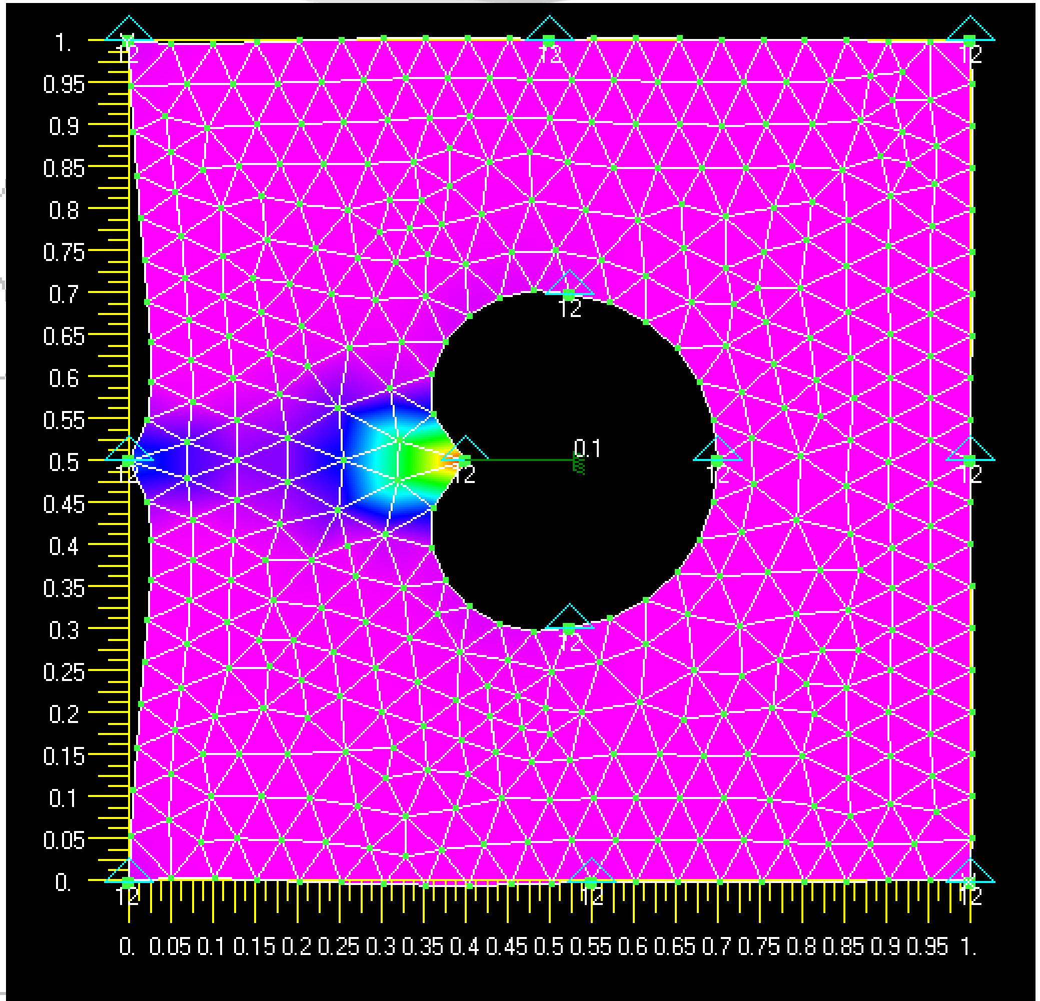
Example 2D: moving hole, boundary points

- Radial basis solution
- 4 source points on the hole
- All the external boundary
- Note how the boundary remains smooth



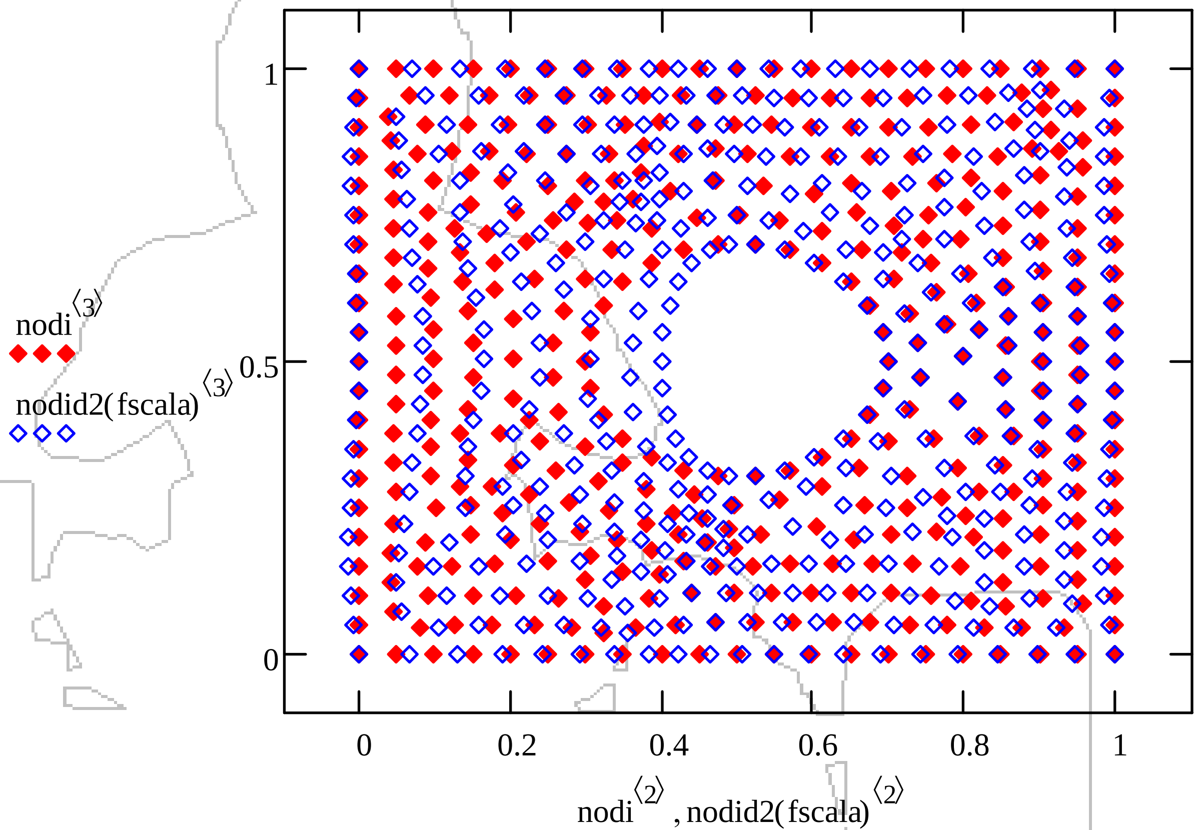
Example 2D: moving hole, boundary points

- Nastran solution
- 12 boundary points



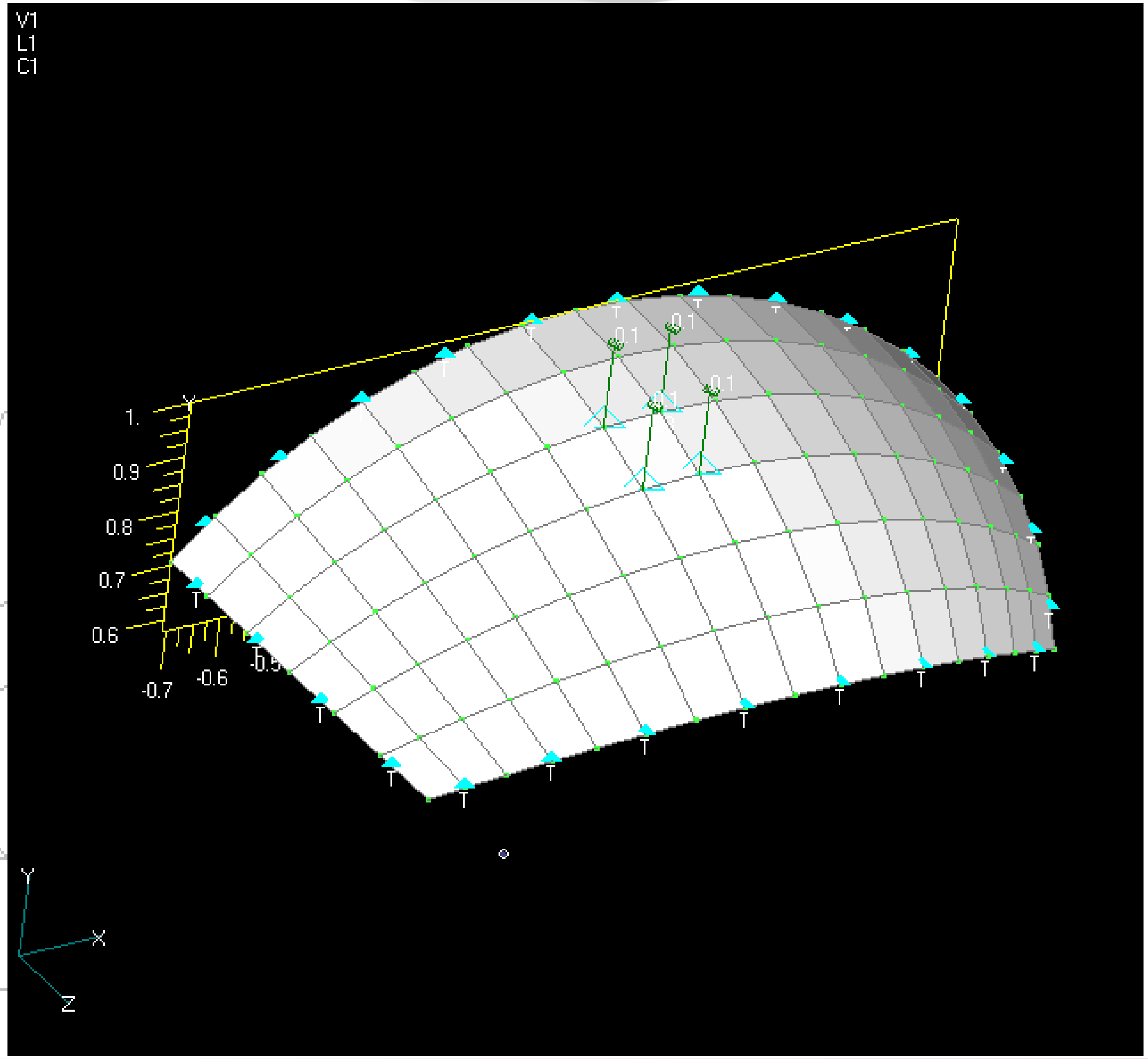
Example 2D: moving hole, boundary points

- Radial basis solution
- 12 source points



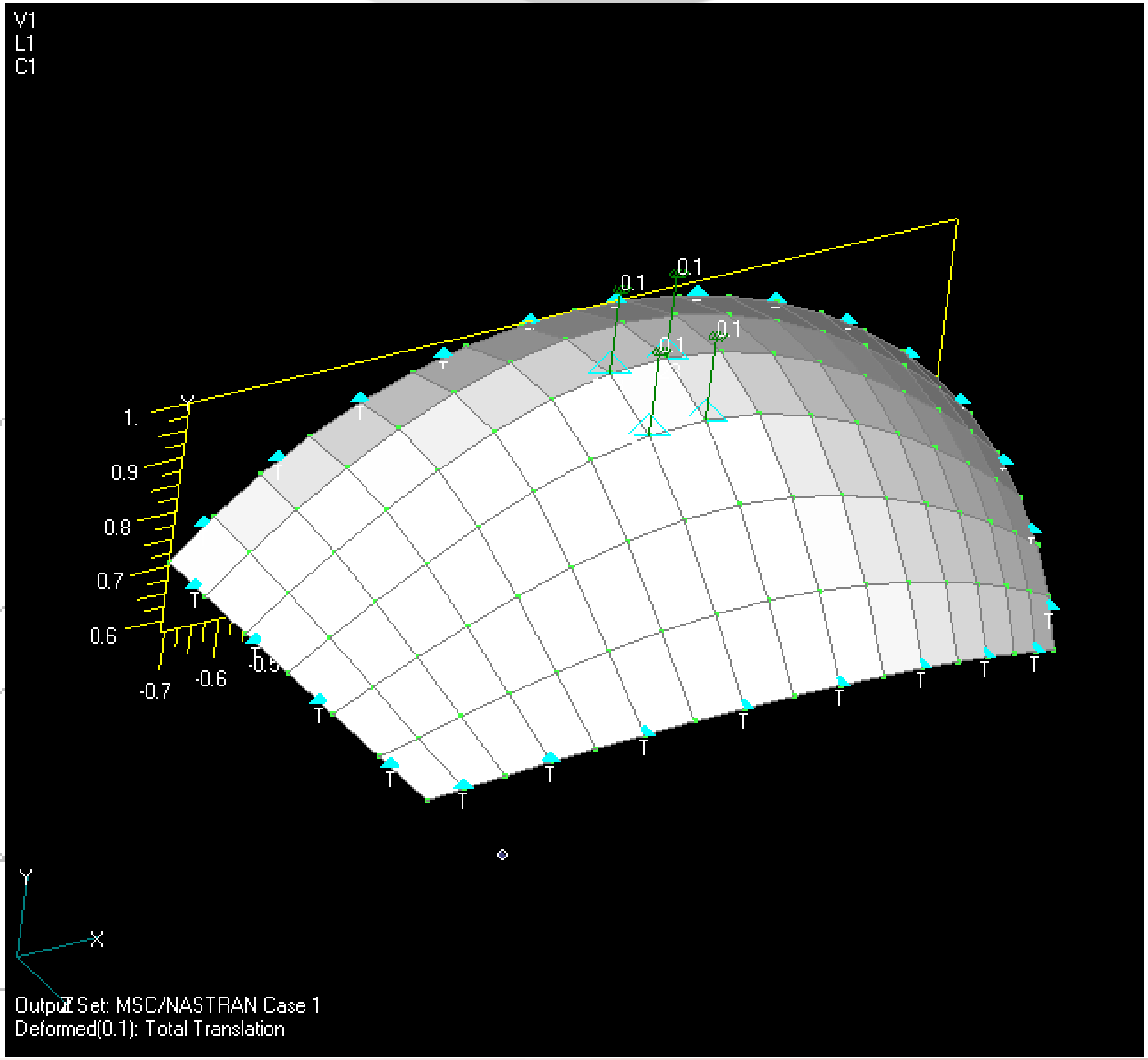
Example 3D: spherical shell

- Undeformed shape



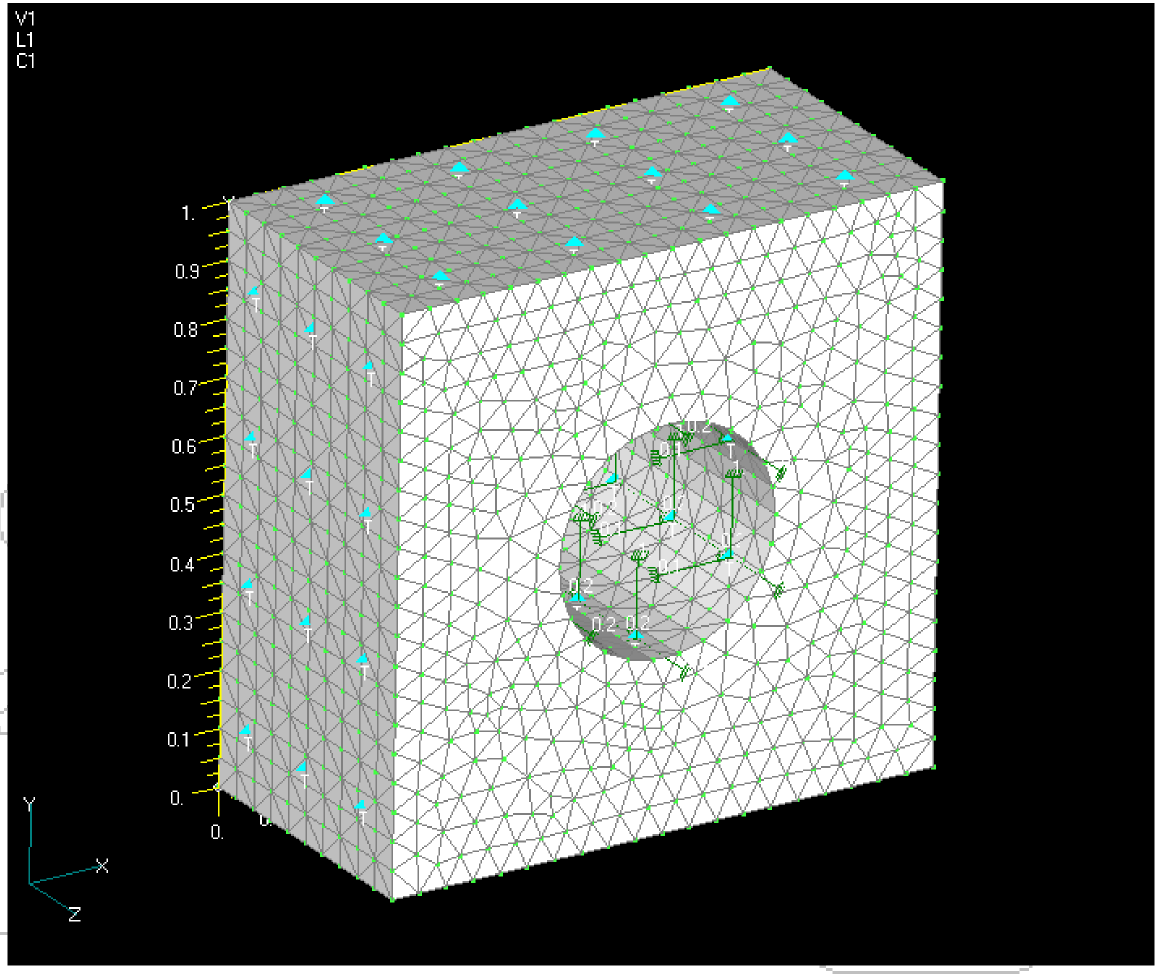
Example 3D: spherical shell

- Deformed shape
- Radial basis 3D
- Post processing of deformation using nastran format
- 4 internal source points
- Boundary source points
- Smooth shape even with 4 internal control points!



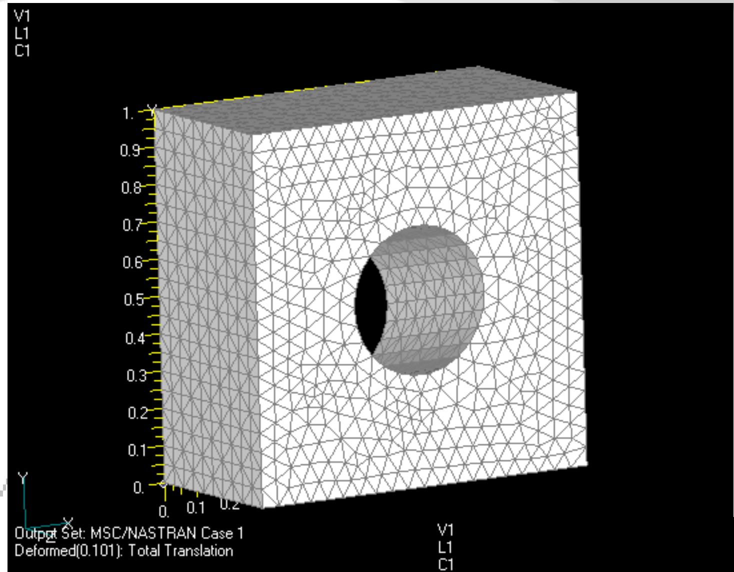
Example 3D: moving hole

- Solid mesh
- External walls fixed
- Various movement of hole surface



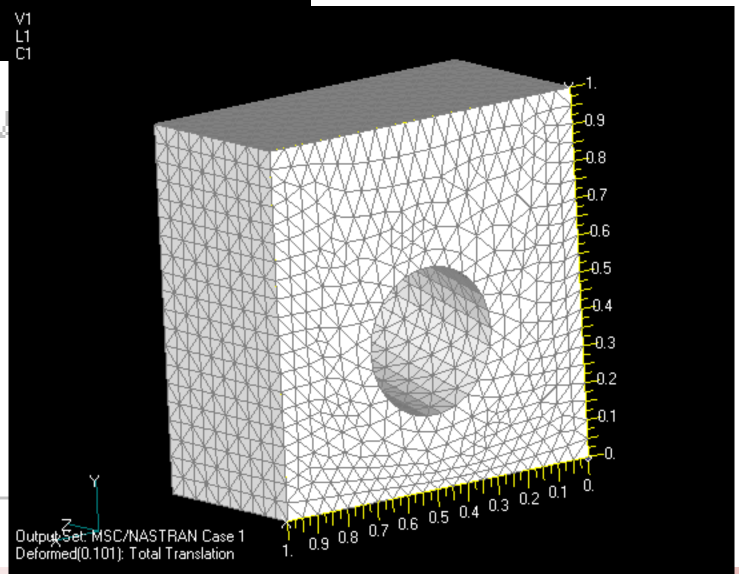
Example 3D: moving hole

- Undeformed shape
- Radial basis 3D
- Post processing of deformation using nastran format
- Rigid rotation of hole surface



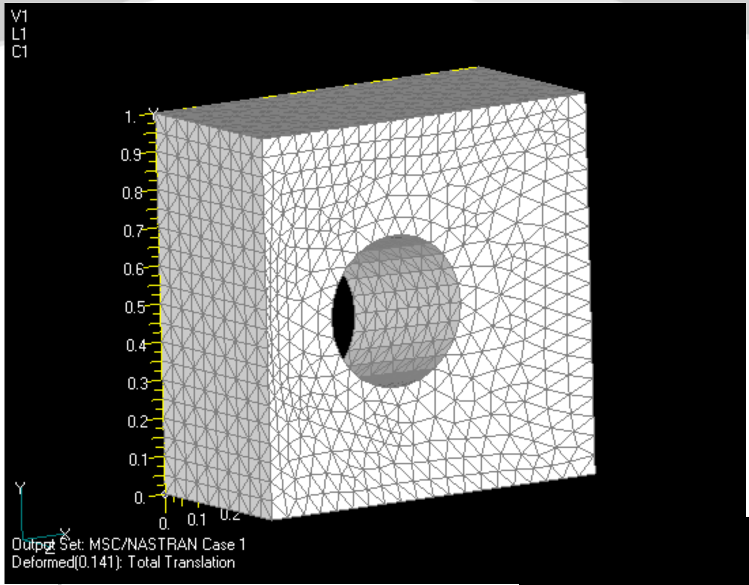
Front

Rear

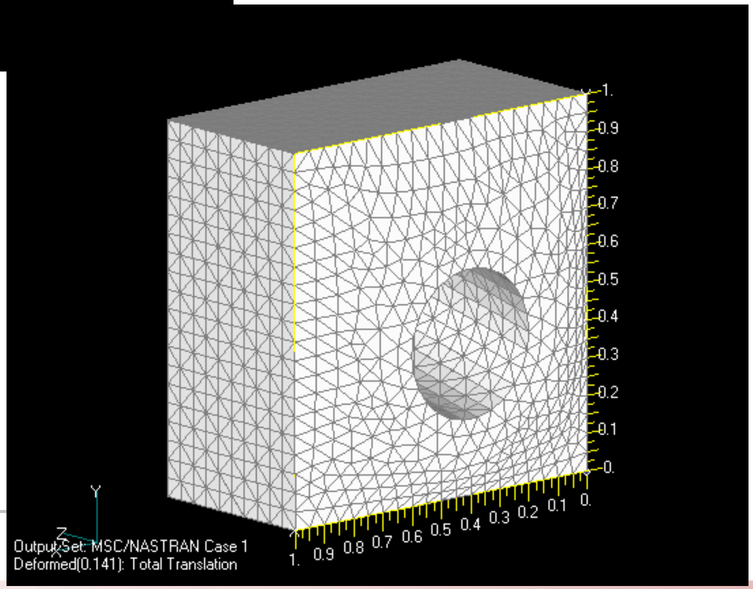


Example 3D: moving hole

- Deformed shape
- Radial basis 3D
- Post processing of deformation using nastran format
- Rigid rotation and translation of hole surface



Front



Rear

Fluent implementation

- Pseudosolid
  - External solver that interact with *.cas file (see Sculptor)
  - External solver (Nastran) linked via udf (variable stiffness not easy to implement)
  - External FEM library linked via UDF
  - Internal FEM solver in UDF (faster)
- Key points
  - Element decomposition (polyedron handling)
  - Non conformal interfaces

Fluent implementation

- Radial basis
  - External tool that interact with *.cas file
  - External tool that interact via UDF
  - Internal tool via UDF (faster, GUI difficult to implement)
- Key points
  - Choosing of the basis functions (compact support)
  - Location of source points
  - Extra Laplacian smoothing or iterative stage in the method to guarantee shape preservation

Computational effort

- Simple test case with MathCAD
- Description
  - 3D case previously presented (“moving hole”)
  - About 20000 nodes
  - About 500 source points
- Results
  - Total computation time (including I/O) 90s
  - Input and basis generation 10s
  - Mesh motion and output 80s

Constrained boundaries strategy

- There are many practical application in which part of the boundary nodes have to move onto a prescribed surface
- The method have to preserve a good quality of the mesh on the deformable boundary
- The method have to propagate all boundaries motion (prescribed or constrained) in the interior nodes

Constraint definition

- Prescribed surface can be defined in several ways:
  - Simple geometry (i.e. plane cylinder sphere)
  - Complex geometry (custom NURBS or external solid modeller evaluator)
  - Mesh based definition (extrapolation is required when the nodes move out of the original meshed surface)
- Projection method capable to move nodes onto the constraint surface

Overall strategy

- Boundary nodes are first partitioned:
  - Set A Prescribed motion nodes
  - Set B Constrained nodes (linked onto surfaces)
- A first radial basis step is performed generating the field using set A to move set B
- The correction is applied projecting transformed set B onto the constraint, nodes of set B can now be handled ad prescribed motion nodes
- A second radial basis step is performed generating the field using set A and B to move interior nodes

Example: A cylinder with a hole

- Two step strategy is applied to move a hole in a cylinder, preserving the shape of the cylindrical surface