INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING, VOL. 40, 2459–2474 (1997)

COMBINATION OF ADAPTIVITY AND MESH SMOOTHING FOR THE FINITE ELEMENT ANALYSIS OF SHELLS WITH INTERSECTIONS

J. RICCIUS, K. SCHWEIZERHOF AND M. BAUMANN

Institut für Mechanik, Universität Karlsruhe (TH), Kaiserstr. 12, 76128 Karlsruhe, Germany

SUMMARY

A compatible hierarchical adaptive scheme is proposed which allows to control both density and geometrical properties of meshes with four-node linear finite shell elements. The algorithm produces a sequence of meshes with two aims, nearly equal distribution of the local error in each element and a mesh with regular elements, thus internal element angles near 90° and length ratios of adjacent element sides near unity. This goal is achieved in an efficient manner imposing a combination of a local smoothing algorithm with the adaptive mesh generation.

New created nodes are positioned on the real shell surface and shell boundaries which may be given e.g. by CAD data. Also the shell directors are determined from the normals on the real geometry. Shell intersections are detected automatically as common curves of two adjacent shell parts. As a shell continuum cannot be assumed for these intersections and thus simple standard adaptive schemes fail, shell intersections have to be treated in a way similar to shell boundaries. For some numerical examples the developed algorithms are demonstrated and the resulting meshes are shown. © 1997 by John Wiley & Sons, Ltd.

KEY WORDS: adaptivity; error computation; shell structures; linear finite element analysis; mesh smoothing

1. INTRODUCTION

The mesh-dependent efficiency of the Finite Element Method is influenced both from the mesh density distribution and, in particular, when linear trial functions are used, from the shape of the elements. An optimal mesh density distribution can be achieved using an adaptive finite element method. However, the resulting shapes are often rather irregular, thus a mesh smoothing seems to be advisable.

In this contribution only *h*-adaptivity is considered, which means that the polynomial degree of the trial functions is fixed and a sequence of successively refined meshes is created until a prescribed accuracy is achieved, see also Reference 1.

Essential ingredients for the realization of this method in combination with mesh smoothing are

1. refinement indicator
2. mesh refinement strategy
3. stopping criterion
4. mesh smoothing.

For two parts, refinement indication and stopping criterion *a posteriori* error estimators have become the standard tool. For the following analysis the error estimator based on the post-processing approach of Zienkiewicz and Zhu[2] is used which is one of the most robust error

CCC 0029–5981/97/132459–16$17.50

Received 7 November 1995
Revised 9 September 1996

estimators as shown by Babuska *et al.*[3] For this error estimator improved nodal stresses are gained by interpolation from stresses at the so-called superconvergent stress points.

For a reasonable interpolation the stresses have to be related to a common global co-ordinate system. In the case of curved, continuous and smooth shell structures, local Cartesian systems (triads) are the best bases for stress description as e.g. the continuous triads proposed by Vu-Quoc.[4] There the third component of the triad is assumed to be identical to the shell director.

In the presence of shell intersections the shell director and as a result the co-ordinate basis changes rapidly and is not continuous when moving from one part to the other part of the structure. In such cases stress interpolation between these parts is no longer admissible. A reasonable procedure is to treat shell intersections like boundaries. Then separate directors for each shell part have to be taken at the intersection nodes—perpendicular to each adjacent shell part, and at the intersection nodes an interpolation/extrapolation scheme for the stresses has to be used for each shell part separately.

In order to improve the shapes of the elements mesh, smoothing algorithms may be incorporated which have been applied with success for problems with plane elements.[5] Within the method proposed in this paper the nodal positions are modified while the number of nodes remains fixed. For reasons of efficiency, single nodes are successively relocated while keeping all other nodes at their position. Here the optimal nodal position in each local mesh smoothing step is determined by solving a non-linear least-squares-fit problem approximately. Extensions of the algorithm allow to move nodes along shell boundaries and shell intersections during this process of mesh optimization.

The contents of this contribution are arranged in four sections: Algorithms for adaptive mesh refinement are presented in Sections 2 (error estimation) and 3 (mesh refinement), the mesh smoothing method is explained in Section 4. Various numerical results in Section 5 show the efficiency of the proposed algorithms.

2. ERROR ESTIMATION

The error estimation for boundaries/intersections and for the shell continuum has to be performed in a different fashion. In particular, the procedures how improved nodal stresses can be found, differ considerably as is shown in this section.

2.1. Shell continuum

A standard smoothing-based error estimator is considered for the elasticity problem in the interior, for which in the case of a linear problem the equilibrium equations can be written in matrix notation as

$$\mathbf{B}^{\mathrm{T}}\mathbf{C}\mathbf{B}\mathbf{u} = \mathbf{f} \quad \text{in } \Omega$$

The energy norm of the error of a finite element E_i,

$$\|\mathbf{e}\|_{a,E_i} = \left(\int_{E_i} (\boldsymbol{\sigma} - \boldsymbol{\sigma}_h)^{\mathrm{T}} \mathbf{C}^{-1} (\boldsymbol{\sigma} - \boldsymbol{\sigma}_h) \, \mathrm{d}\Omega \right)^{1/2}$$

is approximated by the value η_i,

$$\eta_i = \left(\int_{E_i} (\boldsymbol{\sigma}^* - \boldsymbol{\sigma}_h)^{\mathrm{T}} \mathbf{C}^{-1} (\boldsymbol{\sigma}^* - \boldsymbol{\sigma}_h) \, \mathrm{d}\Omega \right)^{1/2} \tag{1}$$

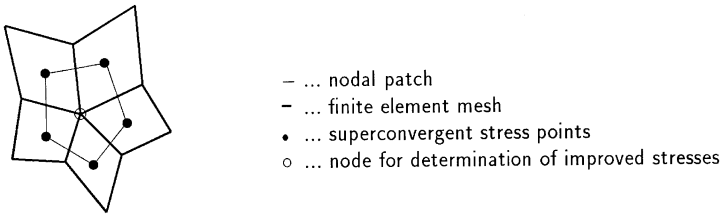


Figure 1. Nodal patch for least-squares stress interpolation

where $\mathbf{B}$ is the strain–displacement operator, $\mathbf{C}$ the material tensor, $\mathbf{u}$ the exact solution, $\mathbf{e}$ $(=\mathbf{u}-\mathbf{u}_h)$ the error of finite element solution, $\mathbf{u}_h$ the finite element solution, $\boldsymbol{\sigma}$ $(=\mathbf{CBu})$ the exact stresses, $\boldsymbol{\sigma}_h$ $(=\mathbf{CBu}_h)$ the finite element stresses and $\boldsymbol{\sigma}^*$ the improved finite element stresses.

As exact stresses are not known, improved finite element stresses $\boldsymbol{\sigma}^*$ must be defined, a task which can be performed in many ways; see References 6–8. A natural way is to use improved nodal stresses $\boldsymbol{\sigma}^*_{\text{node}_i}$ and finite element trial functions $N_l, l = 1,4$:

$$\boldsymbol{\sigma}^* = \sum_{l=1}^{4} N_l \boldsymbol{\sigma}^*_{\text{node}_l} \tag{2}$$

From the various methods tested by the authors for computing improved nodal stresses $\boldsymbol{\sigma}^*_{\text{node}_i}$, see, e.g., References 1 and 8, the least-squares-fit approach of Zienkiewicz and Zhu[2] based on element patches is chosen for the following analyses; see also Figure 1.

Then the jth component of stress $\sigma_j^{\text{P}}(x,y,z)$ in a patch is interpolated as

$$\sigma_j^{\text{P}}(x,y,z) = \mathbf{P}(x,y,z)\mathbf{a}_j, \quad j = 1,m \tag{3}$$

where m is the number of stress components, $\mathbf{P}(x,y,z) = (1,x,y,z)\ldots$ are the linear interpolation functions and $\mathbf{a}_j = (a_{j1}, a_{j2}, a_{j3}, a_{j3})^{\text{T}}\ldots$ coefficients (stresses).

The coefficients a_{jl} are determined by solving the following minimization problem:

$$\sum_{k=1}^{n} (\sigma_j^h(x_k,y_k,z_k) - \underbrace{\mathbf{P}(x_k,y_k,z_k)\mathbf{a}_j}_{\sigma_j^{\text{P}}})^2 \rightarrow \min$$

where n is the number of superconvergent stress points in the patch, $\mathbf{x}_k = (x_k,y_k,z_k)\ldots$ are the co-ordinates of the superconvergent stress points (SSP) and σ_j^h the jth component of stress at SSP in the FE mesh.

This linear least-squares problem is solved using the Euler equations

$$\sum_{k=1}^{n} \mathbf{P}^{\text{T}}(\mathbf{x}_k)\mathbf{P}(\mathbf{x}_k)\mathbf{a}_j = \sum_{k=1}^{n} \mathbf{P}^{\text{T}}(\mathbf{x}_k)\sigma_j^h(\mathbf{x}_k) \tag{4}$$

After determining the vectors $\mathbf{a}_j$ by equation (4), the co-ordinates of the searched stress point are combined with equation (3) in order to get the improved stress values at the nodes:

$$\sigma^*_{j,\text{node}} = \mathbf{P}(x_{\text{node}}, y_{\text{node}}, z_{\text{node}})\mathbf{a}_j$$

Now an error estimator η_i for the element E_i can be computed using equations (2) and (1).

2.2. *Treatment of boundaries and shell intersections*

Transforming the stresses of two adjacent elements, whose common side forms a shell intersection, into a common co-ordinate system yields often non-vanishing normal stresses in normal

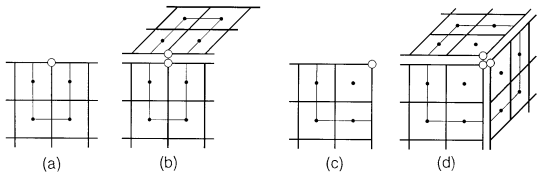


Figure 2. Extrapolation scheme for boundaries, vertices and shell intersections. (a) boundary; (b) simple intersection; (c) vertices; (d) multiple shell intersection

direction of the shell. This is inconsistent with basic assumptions of shell theory. In order to avoid this discrepancy, stress smoothing is not performed at shell intersections and each element adjacent to a shell intersection is treated like a boundary element.

At the boundaries only two and at vertices only one superconvergent stress point are available for a patch. Thus, the interpolation procedure for each stress component σ_i has to be modified to take into account also the superconvergent stress points of the neighbouring elements which results in an extrapolation of the standard bilinear patch as shown in Figure 2.

In the case of shell intersections each shell part adjacent to the shell intersection is treated like a separate boundary or vertices. Also the errors which could be computed from the equilibrium at the boundaries are not considered.

3. MESH REFINEMENT ALGORITHMS

3.1. Standard refinement technique

A hierarchical mesh refinement strategy is used. This means that all initially given and already generated nodes will be included in the subsequent meshes. So the mesh refinement can be performed very efficiently.

The elements with a local error estimator beyond a certain tolerance are refined into 4 new elements by connecting the middle of the sides with each other. In order to get compatible finite element meshes, refined and unrefined areas are connected by transition mesh parts consisting of three quadrilaterals; see Figure 3. For further details see References 9 and 1. The shape of the generated transition mesh parts is often not very favourable for further analyses, thus the irregularly shaped transition mesh parts are replaced by regular shape refinements in any further refinement steps, if indicated.

3.2. Approximation of shell geometry and shell intersections

The exact shell geometry is often defined independently of any finite element mesh by implicit functions f_{shell_i} for known geometric entities as e.g. cylinders and ellipsoids or by patches given from CAD systems, where each can be also described by implicit functions for each patch:

$$f_{\text{shell}_1}(x, y, z) = 0 \tag{5}$$

All vertices nodes of an element are usually placed on the shell surface, thus satisfying equation (5). Then all 'child' nodes of this element created during refinement should also be positioned to fulfill equation (5).

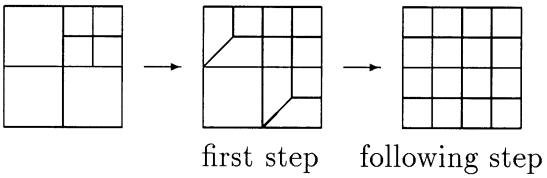
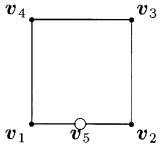


Figure 3. Refinement strategy

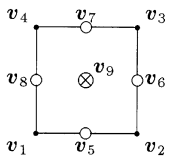


$$|\boldsymbol{v}_5 - \boldsymbol{v}_1| = |\boldsymbol{v}_2 - \boldsymbol{v}_5| \tag{6}$$

Figure 4. 'Equal-distance' condition for new edge nodes, e.g. node 5

Table I. Case-dependent additional conditions for new edge nodes e.g. node 5

Cases Basis: Two adjacent vertices nodes $\mathbf{v}_1$ and $\mathbf{v}_2$ of the parent element with $\mathbf{v}_i = (v_{x,i}, v_{y,i}, v_{z,i})^{\mathrm{T}}$	Conditions For new generated vertices node $\mathbf{v}_5$
Standard case (not boundary, not shell intersection)	$\mathbf{v}_5 = \mathbf{v}_1 + \eta_1^*(\mathbf{v}_2 - \mathbf{v}_1) + \eta_2^* \frac{\nabla f(\mathbf{v}_1) + \nabla f(\mathbf{v}_2)}{2}$
Boundary nodes $f_{\text{boun}}(\mathbf{v}_1) = 0; f_{\text{boun}}(\mathbf{v}_2) = 0$	$f_{\text{boun}}(\mathbf{v}_5) = 0$
Shell intersection nodes $f_{\text{shell}_2}(\mathbf{v}_1) = 0; f_{\text{shell}_2}(\mathbf{v}_2) = 0$	$f_{\text{shell}_2}(\mathbf{v}_5) = 0$



$$\begin{aligned} |\boldsymbol{v}_6 - \boldsymbol{v}_9| &= |\boldsymbol{v}_9 - \boldsymbol{v}_8| \\ |\boldsymbol{v}_9 - \boldsymbol{v}_5| &= |\boldsymbol{v}_7 - \boldsymbol{v}_9| \end{aligned}$$

Figure 5. Condition for a new interior node, e.g. node 9

To fix a node in space uniquely three conditions have to be satisfied. Thus, in addition to equation (5) the following conditions arise for the various cases of new nodes.

(a) *New edge nodes*: For all edge nodes which are not vertices nodes of the physical domain of computation an 'equal-distance' condition as defined in Figure 4 is applied. In addition, one case-dependent condition as listed in Table I for the three possible cases has to be defined. The parameters used in this table are: η_1^*, η_2^*, parameters obtained from equations (5) and (6); f_{boun}, boundary curve; and f_{shell_2}, second shell function.

(b) *New interior nodes*, e.g. $\mathbf{v}_9$: For all interior nodes two 'equal-distance' conditions as defined in Figure 5 are applied.

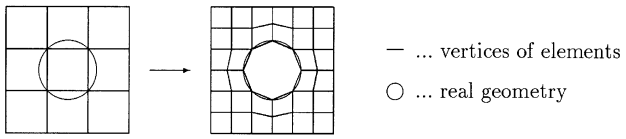


Figure 6. Increasing distortion of elements during uniform mesh refinement due to increasing better approximation of curved boundary

4. MESH SMOOTHING

The generation of meshes for domains with complicated boundaries often leads to distorted elements. In such elements a loss of accuracy e.g. for stresses must be expected. This loss is continued in the adaptive refinement process, when a hierarchical refinement technique is used. When the real geometry of the domain is approximated with increasing quality in the adaptive refinement process the distortion of the elements may be increased too, see e.g. Figure 6.

Known methods for the global generation of smooth meshes are based on the solution of the equations of Laplace or Poisson, see e.g. Reference 10, and require the solution of large systems of equations. It is possible to increase the efficiency of this method by cubic interpolation of coarse grid points, see e.g. Reference 11; however, this method is not applicable for adaptively created meshes. Also local mesh smoothing algorithms based on an evolution strategy are known, see e.g. Reference 12, which have the disadvantage that boundary nodes remain at their original position and distorted elements may occur at the boundaries.

Thus, a new and efficient local mesh smoothing algorithm is proposed, which allows to overcome the mentioned disadvantages.

4.1. Objective and design of the proposed mesh smoothing algorithm

In the proposed mesh smoothing algorithm the geometrical properties of the elements of the mesh are improved by relocating the nodes. For reasons of efficiency the algorithm is designed as a sequence of local mesh smoothing steps. In each step only one node is moved while the others remain fixed; see Figure 7 for an example.

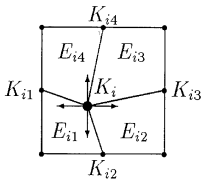
When node K_i is relocated only the m elements adjacent to K_i are changed. Considering one element E_{ij} and for reasons of simplicity suppressing the index i which specifies the node number, we can note for this element E_j with side vectors $\mathbf{a}_j, \mathbf{b}_j, \mathbf{c}_j$ and $\mathbf{d}_j$, as shown in Figure 8, the following geometrical properties:

The optimal choice of a perfect quadrilateral, flat element with equal sides is described by the following three conditions:

$$|\mathbf{a}_j| = |\mathbf{b}_j| \tag{7}$$

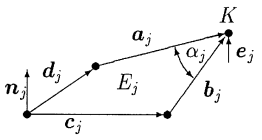
$$\alpha_j = 90^\circ \tag{8}$$

$$|\mathbf{e}_j| = 0 \tag{9}$$



— ... FE–mesh
• ... node K_i to be relocated
• ... fixed nodes K_{ij}; $j = 1, m$
E_{ij} ... elements surrounding node K_i ; $j = 1, m$
m ... number of elements surrounding node K_i

Figure 7. Nodal patch for mesh smoothing



$$\begin{aligned} \boldsymbol{n}_j &= \frac{\boldsymbol{c}_j \times \boldsymbol{d}_j}{|\boldsymbol{c}_j \times \boldsymbol{d}_j|} \\ \boldsymbol{e}_j &= (\boldsymbol{a}_j \cdot \boldsymbol{n}_j)\boldsymbol{n}_j \\ |\boldsymbol{e}_j| &= |\boldsymbol{a}_j \cdot \boldsymbol{n}_j| \end{aligned}$$

Figure 8. One element of nodal patch

which will be achieved only in very special cases. In order to meet conditions (7)–(9) in the best possible fashion the optimal nodal position is determined as a solution of the following combined local minimization problem:

$$f(\mathbf{x}) \to \min \tag{10}$$

$$f(\mathbf{x}) = \sum_{j=1}^{m} \left(\left(w_1 \frac{|\mathbf{a}_j| - |\mathbf{b}_j|}{|\mathbf{a}_j| + |\mathbf{b}_j|} \right)^2 + \left(w_2 \left(\arccos \left(\frac{\mathbf{a}_j \cdot \mathbf{b}_j}{|\mathbf{a}_j||\mathbf{b}_j|} \right) - \frac{\pi}{2} \right) \right)^2 + (w_3(\mathbf{a}_j \cdot \mathbf{n}_j))^2 \right)$$

with $\mathbf{x} = (x, y, z)^{\mathrm{T}}$ being the co-ordinates of node K and w_k $(k = 1, 3)$ the weights for conditions (7)–(9).

This involves the restriction that the node K has to be on the shell surface also after the mesh-smoothing step; thus the co-ordinates of K have to fulfill the shell function (5). The vectors $\mathbf{a}_j$ and $\mathbf{b}_j$ are dependent implicitly on $\mathbf{x}$. The weight coefficients w_k which allow to favour one condition over another are usually set to one. In order to solve the minimization problem given by equation (10), a two-step iterative procedure is suggested. The index l is then the current mesh-smoothing iteration step for the considered node K. The initial co-ordinates of node K are used as starting vector $\mathbf{x}_0 = \mathbf{x}$.

The two steps are:

1. *Minimization of $f(\mathbf{x})$ in the tangent plane of the shell in node K*:

First an intermediate location

$$\mathbf{x}_{l+1/2} = \mathbf{x}_l - \beta_1^* \mathbf{t}_{l1} - \beta_2^* \mathbf{t}_{l2} \tag{11}$$

is determined. $\mathbf{t}_{l1} = (t_{xk1}, t_{yk1}, t_{zk1})$ and $\mathbf{t}_{l2} = (t_{xk2}, t_{yk2}, t_{zk2})$ are orthogonal unit tangent vectors of the shell at node location $\mathbf{x}_l$. The factors β_1^* and β_2^* are obtained by minimizing $f(\mathbf{x}_{l+1/2})$:

$$f(\mathbf{x}_{l+1/2}(\beta_1^*, \beta_2^*)) = \min_{\substack{\beta_1 \in R^1 \\ \beta_2 \in R^1}} f(\mathbf{x}_l - \beta_1 \mathbf{t}_{l1} - \beta_2 \mathbf{t}_{l2}) \tag{12}$$

This non-linear least-squares problem is solved using the Gauss–Newton method; see e.g. Reference 13. For this purpose the minimization problem from equation (12) is written in the following manner—index l is suppressed for simplicity reasons:

$$f(\mathbf{x}(\beta_1, \beta_2)) = \|\mathbf{F}(\mathbf{x}(\beta_1, \beta_2))\|_2^2 \tag{13}$$

$$= \mathbf{F}(\mathbf{x}(\beta_1, \beta_2))^{\mathrm{T}} \mathbf{F}(\mathbf{x}(\beta_1, \beta_2)) \tag{14}$$

$$= \sum_{i=1}^{n} f_i^2(\mathbf{x}(\beta_1, \beta_2)) \xrightarrow{\mathbf{x}} \min \tag{15}$$

with

$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{x}) \end{bmatrix}, \quad n = 3m \tag{16}$$

$$f_i(\mathbf{x}) = w_1 \frac{|\mathbf{a}_i| - |\mathbf{b}_i|}{|\mathbf{a}_i| - |\mathbf{b}_i|}, \quad i = 1, \ldots, m \tag{17}$$

$$f_{i+m}(\mathbf{x}) = w_2 \left(\arccos \frac{\mathbf{a}_i \mathbf{b}_i}{|\mathbf{a}_i||\mathbf{b}_i|} - \frac{\pi}{2} \right), \quad i = 1, \ldots, m \tag{18}$$

$$f_{i+2m}(\mathbf{x}) = w_3(\mathbf{a}_j \mathbf{n}_j), \quad i = 1, \ldots, m \tag{19}$$

$$m = \text{number of elements surrounding node } K_i \tag{20}$$

The Gauss–Newton method is identical to a linearization of equation (15):

$$||\mathbf{F}(\mathbf{x}_l) - \mathbf{F}'(\mathbf{x}_l)[\beta_1 \mathbf{t}_{l1} + \beta_2 \mathbf{t}_{l2}]||_2^2 \xrightarrow{\beta_1, \beta_2} \min \tag{21}$$

with

$$\mathbf{F}'(\mathbf{x}_k) = \begin{bmatrix} \dfrac{\partial f_1}{\partial \beta_1} & \dfrac{\partial f_1}{\partial \beta_2} \\ \vdots & \vdots \\ \dfrac{\partial f_n}{\partial \beta_1} & \dfrac{\partial f_n}{\partial \beta_2} \end{bmatrix}$$

$$\frac{\partial f_i}{\partial \beta_j} = \frac{\partial f_i}{\partial x} \cdot t_{xlj} + \frac{\partial f_i}{\partial y} \cdot t_{ylj} + \frac{\partial f_i}{\partial z} \cdot t_{zlj}$$

This linear least-squares problem is solved using the Euler equations as shown in Section 2.1.

2. *Back projection to shell surface*:

As in general the intermediate location $\mathbf{x}_{l+1/2}$ will no longer fulfill the shell equation $f_{\text{shell}_1}(\mathbf{x}_{l+1/2}) = 0$, a projection back to the shell surface—a second step is mostly necessary.

As search direction for the final location $\mathbf{x}_{l+1}$ the gradient of the shell description function $f_{\text{shell}_1}(\mathbf{x})$ is used:

$$\mathbf{x}_{l+1} = \mathbf{x}_{l+1/2} + \lambda \nabla f_{\text{shell}_1}(\mathbf{x}_{l+1}) \tag{22}$$

Thus, the projection is performed in perpendicular direction to the shell surface and the parameter λ is determined iteratively such that $\mathbf{x}_{l+1}$ finally satisfies the shell equation (5).

4.2. Treatment of shell intersections and boundaries

In order to remain in the physical domain of computation the real vertices nodes of the shell are fixed. Edge nodes and vertices nodes of elements arising from the piecewise linear approximation of curved shell intersections or boundaries may be moved along the intersections or boundaries in the smoothing process.

Like in Section 4.1, a two-step iterative procedure may be used to move edge nodes:

1. Minimization of $f(\mathbf{x})$ along a line tangential to two adjacent shell parts f_{shell_1} and f_{shell_2} which form a shell intersection.

   A tangent vector to f_{shell_1} and f_{shell_2} at any point $\mathbf{x}_l$ may be determined by

$$\mathbf{t}_l = \nabla f_{\text{shell}_1}(\mathbf{x}_l) \times \nabla f_{\text{shell}_2}(\mathbf{x}_l) \tag{23}$$

   Now the intermediate location is computed as

$$\mathbf{x}_{l+1/2} = \mathbf{x}_l - \gamma^* \mathbf{t}_l \tag{24}$$

   The scalar parameter γ^* is determined such that $f(\mathbf{x}_{l+1/2})$ will be a minimum:

$$f(\mathbf{x}_{l+1/2}(\gamma^*)) = \min_{\gamma \in R^1} f(\mathbf{x}_l - \gamma \mathbf{t}_l)$$

   Combining equation (24) with the Gauss–Newton-iteration formula, see e.g. Reference 13, leads to a scalar quadratic minimization problem for γ^* which may be solved directly:

$$\gamma^* = \frac{\sum_{i=1}^{n} f_i(\mathbf{x}_l) \left(\frac{\partial f_i(\mathbf{x}_l)}{\partial x} t_x + \frac{\partial f_i(\mathbf{x}_l)}{\partial y} t_y + \frac{\partial f_i(\mathbf{x}_l)}{\partial z} t_z \right)}{-\sum_{i=1}^{n} \left(\frac{\partial f_i(\mathbf{x}_l)}{\partial x} t_x + \frac{\partial f_i(\mathbf{x}_l)}{\partial y} t_y + \frac{\partial f_i(\mathbf{x}_l)}{\partial z} t_z \right)^2} \tag{25}$$

   Then the intermediate location $\mathbf{x}_{l+1/2}$ is determined, which is often not on the shell surface.
2. Two shell equations have to be fulfilled by the final location $\mathbf{x}_{l+1}$ on the shell surface; thus $\mathbf{x}_{l+1}$ depends on two parameters λ_1 and λ_2:

$$\mathbf{x}_{l+1} = \mathbf{x}_{k+1/2} + \sum_{i=1}^{2} \lambda_i \nabla f_{\text{shell}_i}(\mathbf{x}_{l+1}) \tag{26}$$

   Then λ_1 and λ_2 are determined by solving the following non-linear system of two equations

$$\begin{aligned} f_{\text{shell}_1}(\mathbf{x}_{l+1}(\lambda_1, \lambda_2)) &= 0 \\ f_{\text{shell}_2}(\mathbf{x}_{l+1}(\lambda_1, \lambda_2)) &= 0 \end{aligned} \tag{27}$$

   by the Newton method.

4.3. Treatment of transition mesh parts

Transition mesh parts resulting from the adaptive refinement process introduced in Section 3.1 contain generally more distorted elements compared to the elements of the initial mesh; see Figure 3.

However, the overall convergence of the adaptive algorithm is not improved, when they are incorporated into the smoothing process during the adaptive refinement, despite smoothing would reduce the distortion at the current refinement level. This is due to the fact that the transition mesh parts are modified in the next refinement step and the resulting elements would be more distorted; see Figure 9, than simply excluding the transition mesh parts from the smoothing process.

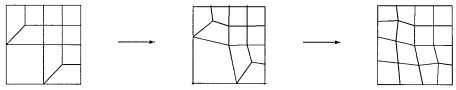


Figure 9. Smoothing of transition elements and following refinement step

Similarly distorted elements would occur if the smoothed transition mesh parts would be further subdivided into the following adaptive steps. Thus, vertices nodes of the elements in transition mesh parts are excluded from smoothing and only in the final mesh of an adaptive process smoothing is applied also to the transition mesh part.

5. NUMERICAL EXAMPLES

For all numerical examples a bilinear isoparametric finite shell element based on the Reissner–Mindlin theory with assumed natural strains as proposed by Dvorkin and Bathe[14] and developed further by Gebhard[15] is used.

5.1. Cylindrical roof with hole

This example, see figure 7, is chosen for a demonstration of the combination of adaptive refinement and mesh smoothing.

The cylindrical roof shown in Figure 10, initially proposed by Scordelis and Lo[16] without a hole is modified by cutting out a circular hole. The problem is symmetric, thus only one quarter has to be modelled, see Figure 11, with symmetry boundary conditions along sides *a–b* and *d–e*. Side *b–c* is supported in radial and in circumferential direction. Dead load is acting in *z*-direction. The initial mesh consists of two elements. In Figure 16 the remarkable reduction of the estimated relative error combining adaptivity and mesh smoothing as well as uniform refinement and mesh smoothing in every step is compared to simple adaptive and uniform refinement. In Figures 12–16 the meshes for the various refinement strategies are shown. The benefit of smoothing becomes obvious in Figures 13 and 15, as the artificial mesh refinement along the diagonal in Figure 13 is mainly a result of badly shaped elements.

5.2. Cantilever beam with T-cross-section

This example, see Figure 17, is chosen to show adaptive refinement for a structure with a shell intersection.

The geometrical parameters are: length $l = 9$, flange width $w = 1$ each, uniform thickness $t = 0{\cdot}1$. The material data are: $E = 6 \cdot 825 \times 10^7$; $v = 0$. The beam is clamped at the left side; thus the translational and rotational degrees of freedom of all boundary nodes at this side are fixed. A uniform load of 10 per unit square is acting in *z*-direction on one flange only. In Figures 18 and 19 the refined final meshes are shown. The effect of the adaptive refinement strategy at the intersection and at the boundaries is clearly visible. A good improvement for the adaptive scheme

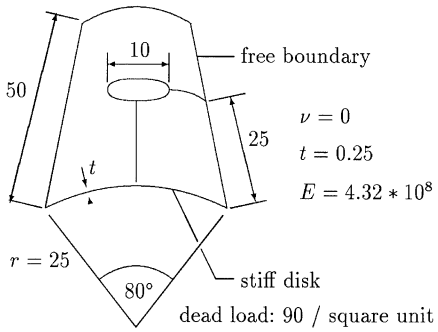


Figure 10. Cylindrical roof, geometry

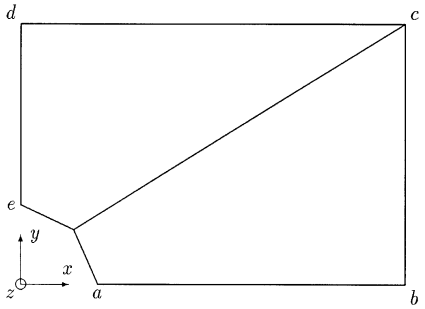


Figure 11. Initial mesh of cylindrical roof: plan view

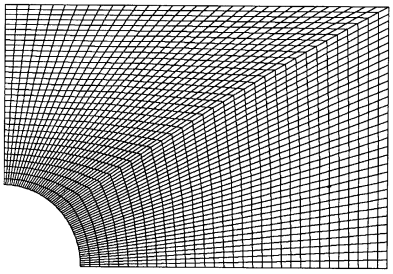


Figure 12. Cylindrical roof. Uniform refinement without smoothing; final mesh: plan view

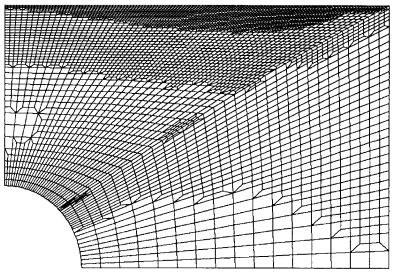


Figure 13. Cylindrical roof. Adaptively refined final mesh without smoothing: plan view

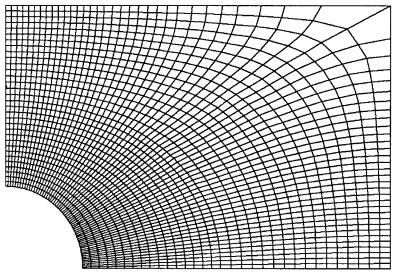


Figure 14. Cylindrical roof. Uniform refinement with smoothing; final mesh: plan view

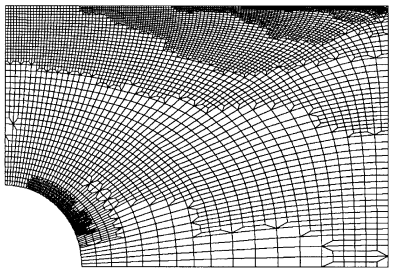


Figure 15. Cylindrical roof. Adaptive refinement with smoothing; final mesh: plan view

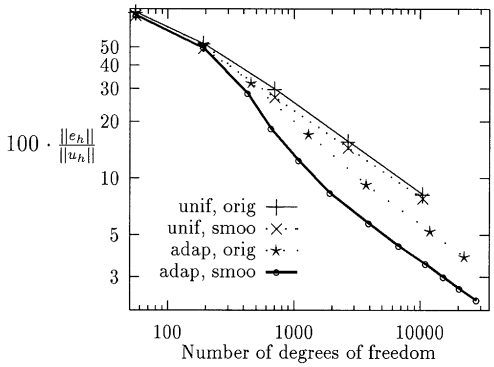


Figure 16. Cylindrical roof. Evolution of relative error using various mesh refinement strategies

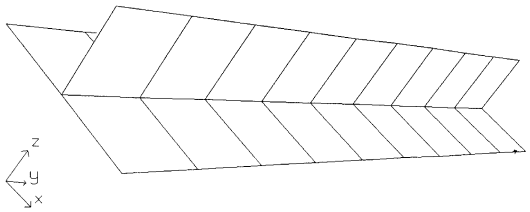


Figure 17. Initial mesh of T-beam

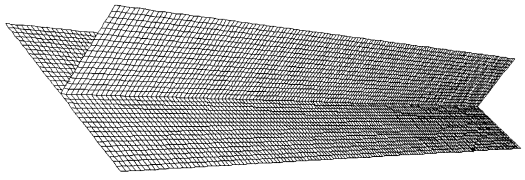


Figure 18. Uniformly refined mesh of T-beam

concerning convergence is shown in Figure 20. Mesh smoothing is not applied, as the initial mesh contains only regular elements and during the adaptive process distorted meshes occur only in the transition mesh parts.

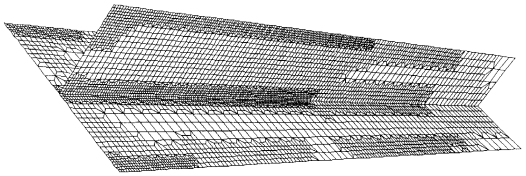


Figure 19. Adaptively refined mesh of T-beam

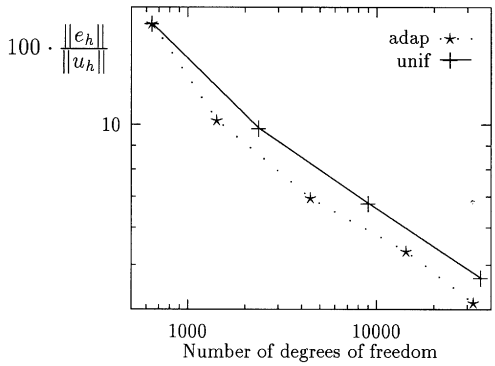


Figure 20. Evolution of relative error using uniform and adaptive mesh refinement for T-beam

5.3. *Cylinder–plate intersection*

This example includes an intersection of a cylinder and a plate with an angle of 45°. The effect of adaptive refinement in combination with mesh smoothing is studied. The length of the plate in direction of the axis of symmetry of the example is $l = 15\sqrt{2}$, its width is $w = 15$. The cylinder has a radius of $r = 6$ and an average length of $l = 10$. Both, cylinder and plate have a wall thickness of $t = 1$. The modulus of elasticity is $E = 1{\cdot}0 \times 10^6$, the Poisson ratio v is set to zero. The plate is simply supported along all edges. The pressure load has a value of 1000 acting in the direction of the longitudinal axis of the cylinder and is uniformly distributed along the free end of the cylinder. The initial mesh, see Figure 21, consists of 8 elements. Figures 22 and 23 show the final mesh using uniform, respectively, adaptive mesh refinement without and with smoothing. In Figure 23 the effect of smoothing becomes visible and in Figure 24 the evolution of the relative error using the various mesh refinement strategies is depicted.

The abbreviations used are: UNIF for uniform refinement, ADAP for adaptive refinement, ORIG if mesh smoothing is not used and SMOO if mesh smoothing is used. For the smoothed meshes with less than 500 elements larger errors are obtained than with non-smoothed meshes with the same number of elements. This is mainly caused by the relatively large elements at the vertices of the plate in the smoothed mesh, see Figure 25 right, compared to the size of the vertices elements in the non-smoothed mesh, see Figure 25 left. The estimated relative errors are above 30 per cent for the whole domain at this rather low discretization level. It seems to be obvious that at this state any error estimation may be not exact enough and the error at the vertices of the plate is underestimated. At high refinement levels the combination of adaptive mesh refinement and mesh smoothing produces lower errors compared to the other refinement strategies considered here. The differences between the various methods seem to be small at a first look, however e.g. the final error of 7·40 per cent is reached using mesh smoothing with 17 586 degrees of freedom; without

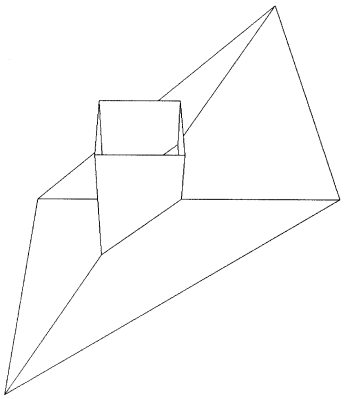


Figure 21. Initial mesh of cylinder–plate intersection

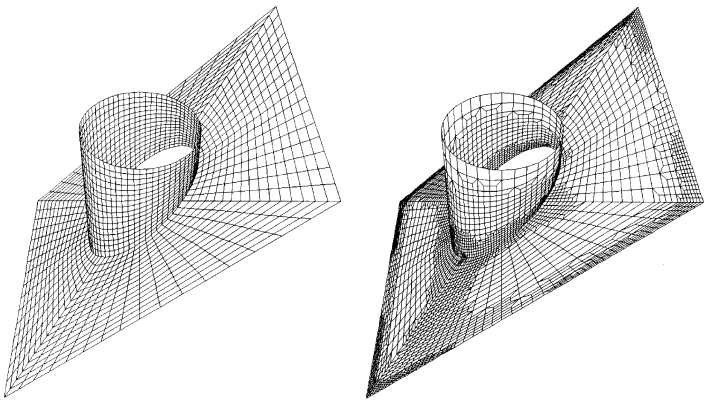


Figure 22. Final meshes using uniform (left side) and adaptive (right side) refinement without smoothing (unif, orig) resp. (adap, orig)

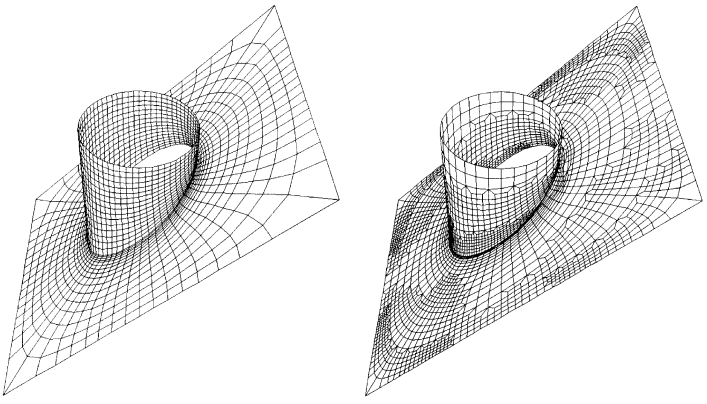


Figure 23. Final meshes using uniform (left side) and adaptive (right side) refinement and smoothing (unif, smoo) resp. (adap, smoo)

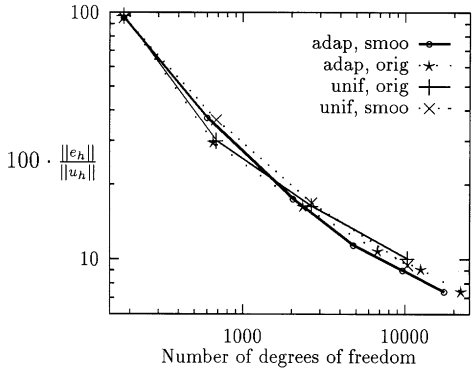


Figure 24. Evolution of estimated relative error using various mesh refinement strategies for the cylinder–plate intersection example

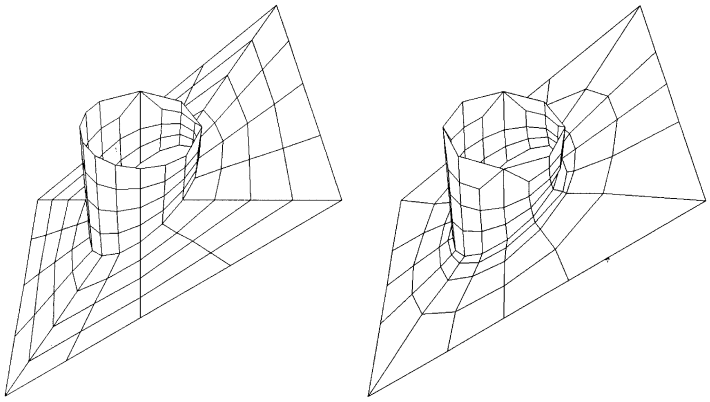


Figure 25. Mesh after second adaptive refinement step without (left side) and with (right side) smoothing

smoothing an error of 7·44 per cent is reached with 22 278 degrees of freedom. This advantage is achieved with marginal effort as the additional number of numerical operations for the smoothing are small—smoothing is performed at each refinement level, but only one Gauss–Newton iteration step per node is necessary.

6. CONCLUSIONS

The proposed adaptive mesh refinement procedure for shell problems with intersections has been shown to work well, if the intersections are treated similar to boundaries. Some remarkable improvements towards error reduction can be achieved adding the suggested smoothing procedure. However, the success of the smoothing is dependent on the element distortions already present in the initial mesh or on the occurrence of some badly shaped elements created during the refinement process. As the smoothing is very efficient, it is suggested to combine mesh refinement and smoothing in all numerical applications.

REFERENCES

1. M. Baumann, 'Lineare Finite Element Konzepte für Schalentragwerke unter Berücksichtigung von Adaptiven Methoden', *Doktorarbeit*, Universität Karlsruhe, 1994.
2. O. C. Zienkiewicz and J. Z. Zhu, 'The superconvergent patch recovery (spr) and adaptive finite element refinement', *Comput. Methods Appl. Mech. Eng.*, **101,** 207–224 (1992).
3. I. Babuska, T. Stroboulis, J. T. Upadhyay, S. K. Gangaraj and K. Copps, 'An objective criterion for assessing the reliability of *a-posteriori* error estimators in finite element computations', *IACM Bull.*, **10,** 27–37 (1995).
4. L. Vu-Quoc and J. A. Mora, 'A class of simple and efficient degenerated shell elements—analysis of global spurious-mode filtering', *Comput. Methods Appl. Mech. Eng.*, **74,** 117–175 (1989).
5. J. Riccius, K. Schweizerhof and M. Baumann, 'Ein schnelles Verfahren zur Verbesserung der Elementgeometrie bei FE-Berechnungen am Beispiel von Flächentragwerken', *ZAMM*, **75,** 275–276 (1995).
6. O. C. Zienkiewicz and J. Z. Zhu, 'A simple error estimator and adaptive procedure for practical engineering analysis', *Int. j. numer. methods eng.*, **24,** 337–357 (1987).
7. O. C. Zienkiewicz and J. Z. Zhu, 'The superconvergent patch recovery and a posteriori error estimates. Part 1: the recovery technique', *Int. j. numer. methods eng.*, **33,** 1131–1364 (1992).
8. K. Schweizerhof, J. Riccius and M. Baumann, 'Verbesserung von Finite Element Berechnungen durch Adaptivität und Netzglättung am Beispiel ebener und gekrümmter Flächentragwerke', *Technical Report*, Universität Karlsruhe, Institut für Wissenschaftliches Rechnen und Mathematische Modellbildung, 1993.
9. L. Plank, 'Netzadaption und Mehrgitterverfahren für die numerische Behandlung von Faltwerken', Forschungs- und Seminarberichte aus dem Bereich der Mechanik der Universität Hannover, Bericht Nr. F 90/3, 1990., Appelstr.9a, 3000 Hannover 1, 1990.
10. J. F. Thompson, Z. U. A. Warsi and C. W. Mastin, Numerical Grid Generation—Foundations and Applications, North-Holland, Amsterdam, 1985.
11. J. Zhu, W. Rodi and B. Schönung, 'A fast method for generating smooth grids', *Technical Report*, Institut für Hydromechanik, Universität Karlsruhe, 1988.
12. B. Jüdt, 'Geometrische und topologische Modellierung für Finite-Elemente-Analysen', *Doktorarbeit*, Universität Gesamthochschule Essen, 1992.
13. P. Spellucci, *Numerische Verfahren der nichtlinearen Optimierung*, Birkhäuser, Basel, 1993.
14. E. Dvorkin and K. J. Bathe, 'A continuum mechanics based four-node shell element for general nonlinear analysis', *Eng. Comput*., **1,** 77–88 (1984).
15. H. Gebhardt, 'Finite Element Konzepte für schubelastische Schalen mit endlichen Drehungen', *Doktorarbeit*, Universität Karlsruhe, 1990.
16. A. C. Scordelis and K. S. Lo, 'Computer analysis of cylindrical shells', *J. Am. Concrete Inst.*, **61,** 539–561 (1969).

