INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2006; **00**:1–28 *Prepared using* nmeauth.cls *[Version: 2002/09/18 v2.02]*

Clustered Generalized Finite Element Methods for Mesh Unrefinement, Non-Matching and Invalid Meshes

C.A. Duarte[a,*], T.J. Liszka[b] and W.W. Tworzydlo[b]

[a] *Department of Civil and Environmental Engineering*
University of Illinois at Urbana-Champaign
Newmark Laboratory, 205 North Mathews Avenue
Urbana, Illinois 61801, USA
[b] *Altair Engineering, Inc.,*
7800 Shoal Creek Blvd. Suite 290E
Austin, Texas, 78757, USA

SUMMARY

In spite of significant advancements in automatic mesh generation during the past decade, the construction of quality finite element discretizations on complex three-dimensional domains is still a difficult and time demanding task.

In this paper, the partition of unity framework used in the generalized finite element method (GFEM) is exploited to create a very robust and flexible method capable of using meshes that are unacceptable for the finite element method, while retaining its accuracy and computational efficiency. This is accomplished not by changing the mesh but instead by clustering groups of nodes and elements. The clusters define a modified finite element partition of unity that is constant over part of the clusters. This so-called clustered partition of unity is then enriched to the desired order using the framework of the GFEM.

The proposed generalized finite element method can correctly and efficiently deal with: (i) Elements with negative Jacobian; (ii) Excessively fine meshes created by automatic mesh generators; (iii) Meshes consisting of several sub-domains with non-matching interfaces. Under such relaxed requirements for an acceptable mesh, and for correctly defined geometries, today's automated tetrahedral mesh generators can practically guarantee successful volume meshing that can be entirely hidden from the user. A detailed technical discussion of the proposed generalized finite element method with clustering along with numerical experiments and some implementation details are presented. Copyright © 2006 John Wiley & Sons, Ltd.

KEY WORDS: Generalized finite element methods; Partition of unity methods; Meshless methods; Mesh unrefinement; Non-matching meshes; Mesh generation.

*Correspondence to: C. A. Duarte, Department of Civil and Environmental Eng., University of Illinois at Urbana-Champaign, Newmark Laboratory, 205 North Mathews Avenue, Urbana, Illinois 61801, USA. Tel.: +1-217-244-2830; Fax: +1-217-333-3821. e-mail: caduarte@uiuc.edu.

Received 10 March 2006
Revised 26 April 2006

1. INTRODUCTION

Computational simulation on complex three-dimensional (3-D) domains has become a common task in recent years in many research laboratories and industries. The construction of quality finite element discretizations on complex 3-D domains is, however, a difficult and time demanding task. Acceptable finite element meshes must satisfy several criteria that are not easily matched when the geometry of the domain is complex. All elements in a mesh must, for example, be properly connected to their neighbors and accurately represent the geometry of the domain. In addition, the aspect ratio of the elements must be within acceptable bounds and elements with small or negative Jacobians are not acceptable.

In spite of significant advancements in automatic mesh generation during the past decade, several difficulties still exist. Automatic mesh generators for hexahedral elements are still a subject of intense research and those currently available require considerable user intervention and tuning of parameters in order to produce acceptable meshes in complex geometries. This limitation has lead to increasing use of automatic tetrahedral mesh generators, although the solutions from these elements are in general of lesser quality than comparable hexahedral models. Tetrahedral mesh generators are much more robust then their hexahedral counterparts, but they also suffer from several limitations. They tend, for example, to produce elements of poor quality near curved boundaries that need to be manually fixed by the user, leading to an overall very time consuming process. These difficulties are compounded when quadratic or higher order elements are used (Cf. Section 7). Automatic mesh generators also often create an excessive number of elements in order to keep the aspect ratio of the elements within reasonable bounds. This is especially pronounced when the domain has transition zones between bulky and slender parts. Another drawback of automatic mesh generators, especially when tetrahedral elements are used, is that they inhibit the optimal use of p-anisotropic approximations, that is, approximations that have different polynomial orders associated with each direction. Problems where boundary layers occur, such as in the analysis of orthotropic materials or high speed flow or where one of the dimensions of the structural part is much smaller than the others, are examples in which p-orthotropic approximations may lead to considerable savings in the number of degrees of freedom needed to achieve acceptable accuracy [34].

It is common practice, especially in the well established finite element (FE) analysis centers in the industry, that the solution process be split into two (very) separate stages: meshing and analysis. As the main effort measurable in human time expenditure is used on creating acceptable FE meshes, the meshing itself became the goal, although actually it has absolutely no value to the engineer, who is interested in the numerical solution (e.g. stress distribution and its implications for the design process).

The difficulties associated with the generation of *quality meshes* for the finite element method has lead to the investigation of alternative methods for solving boundary value problems. In particular, this has lead to the development of the so-called meshless or meshfree methods [4, 15, 16, 18, 22–24, 28, 41, 42, 46, 47, 58, 59, 61, 63, 66, 71–73, 82–85, 88, 89]. Excellent overviews of meshfree methods and their applications can be found in, for example, [6, 14, 48, 51, 57, 60, 62]. Variants of the finite element method were also proposed in the last decade with the aim of removing some limitations of the classical finite element method. Among these alternatives is the generalized finite element method (GFEM) [9, 35–37, 64, 68, 69, 78, 79] and the extended finite element method [27, 32, 65, 80, 81]. These methods are instances of the so-called partition of unity method [8, 9, 33, 40, 42, 64] and are closely related to the classical

finite element method while providing a much higher level of flexibility than the later.

This paper presents a generalized finite element method that addresses the problem of meshing complex 3-D domains by reducing the quality requirements for the initial mesh and thus allowing the analyst to concentrate on the solution process as a whole, in particular, on the quality of the computed solution, as opposed to the meshing process. Our goal is to create a fully automatic analysis package, where automatic mesh generators are a hidden part of the whole process, and the output solution is of guaranteed good accuracy, even if the automatically generated mesh is not acceptable for a standard finite element method. This is accomplished not by changing the mesh but instead by "clustering" sets of nodes and elements. The clusters define a modified finite element partition of unity that is constant over part of the clusters. This so-called *clustered partition of unity* is then enriched to the desired order using the framework of the GFEM. We refer to the GFEM presented here as a *GFEM with clustering*.

The proposed generalized finite element method has the following unique features:

1. It can perform effective unrefinement of finite element meshes composed of any type of elements in two- or three-dimensional spaces. Numerical examples demonstrating the application of the proposed technique to large, three-dimensional, meshes are presented in Section 5.
2. It accepts non-matching finite element meshes. That is, meshes composed of subparts that were meshed independently of each other. Here, the partition of unity framework is used to "glue" the subparts together in such way that the solution is continuous along the entire interface between the subparts. This technique along with numerical examples are presented in Section 6. A brief review of existing techniques to handle non-matching finite element meshes is also presented in Section 6.
3. The proposed GFEM with clustering can also accept meshes containing elements with large aspect ratios and even elements with negative Jacobians, while retaining its accuracy and computational efficiency (Cf. Section 7).

The proposed GFEM with clustering combines and extends the best features of the finite element method while allowing for easier and automatic model preparation. It may virtually guarantee that an acceptable computational model can be created even for the most complex domains with little or no user intervention, and that such a model can be solved using the computer facilities currently available to the analyst. Importantly, presently there are very few techniques that are able to handle non-matching meshes, mesh unrefinement and, especially, elements of poor quality. Some existing methods, while used in practice, have difficulties in retaining theoretical correctness and consistency. A detailed technical discussion of the proposed GFEM with clustering along with numerical experiments are presented in the next sections. Brief reviews of related techniques are presented as well.

2. PARTITION OF UNITY AND GENERALIZED FINITE ELEMENT METHODS

The clustered generalized finite element method presented in this paper is one instance of the so-called partition of unity approximations [9, 35, 36, 41, 42, 64, 69, 78, 79]. A partition of unity-based approximation of a scalar field $u(\boldsymbol{x})$ defined on a domain $\Omega \subset \mathbb{R}^n, n = 1, 2, 3$, can

be written as

$$u_h(\boldsymbol{x}) = \sum_{\alpha=1}^{N} \varphi_\alpha(\boldsymbol{x}) u_{h\alpha}(\boldsymbol{x}) \tag{1}$$

where

(i) $\mathsf{POU}_N = \{\varphi_\alpha\}_{\alpha=1}^N$ constitute a *partition of unity* (POU) with N functions φ_α defined on Ω and with properties

$$\varphi_\alpha \in C_0^s(\omega_\alpha),\ s \geq 0, \qquad 1 \leq \alpha \leq N \tag{2}$$

$$\sum_{\alpha}^{N} \varphi_\alpha(\boldsymbol{x}) = 1 \qquad \forall \boldsymbol{x} \in \Omega \tag{3}$$

The support of φ_α, $\{\boldsymbol{x} : \varphi_\alpha(\boldsymbol{x}) \neq 0\}$, is denoted by ω_α (often called *cloud*) and $\boldsymbol{x}_\alpha$ denotes a node associated with function φ_α and its support. Examples of partition of unities are standard finite element shape functions, functions generated by moving least squares methods and Shepard functions [42, 56].

(ii) $u_{h\alpha}(\boldsymbol{x})$ denotes a local approximation of the field $u(\boldsymbol{x})$ defined on ω_α and belonging to the local space

$$\chi_\alpha(\omega_\alpha) = \text{span}\{L_{i\alpha}(\boldsymbol{x})\}_{i\in\mathcal{I}(\alpha)} \tag{4}$$

where the basis functions $L_{i\alpha}, i \in \mathcal{I}(\alpha)$, are also denoted by *enrichment functions* and $\mathcal{I}(\alpha)$ is an index set such that

$$L_{1\alpha} = 1 \tag{5}$$

Examples of enrichment functions for a node $\boldsymbol{x}_\alpha = (x_\alpha, y_\alpha)$ in two-dimensions are

$$\begin{aligned} &\left\{1, \frac{(x-x_\alpha)}{h_\alpha}, \frac{(y-y_\alpha)}{h_\alpha}\right\} && \text{linear} \\ &\left\{1, \frac{(x-x_\alpha)}{h_\alpha}, \frac{(y-y_\alpha)}{h_\alpha}, \frac{(x-x_\alpha)^2}{h_\alpha^2}, \frac{(x-x_\alpha)}{h_\alpha}\frac{(y-y_\alpha)}{h_\alpha}, \frac{(y-y_\alpha)^2}{h_\alpha^2}\right\} && \text{quadratic} \end{aligned} \tag{6}$$

where h_α is a scaling factor [36].

Using the definitions above, we can write $u_{h\alpha}(\boldsymbol{x})$ as

$$u_{h\alpha}(\boldsymbol{x}) = \sum_{i\in\mathcal{I}(\alpha)} a_{i\alpha} L_{i\alpha}(\boldsymbol{x}), \qquad a_{i\alpha} \in \mathbb{R} \tag{7}$$

The approximations used in *all* partition of unity methods like the *hp*-cloud method [41, 42], the generalized finite element method [9, 35, 36, 64, 69, 78, 79], the particle-partition of unity method [47, 49, 50], the extended finite element method [17, 65, 81], among others, are special cases of (1), the basic difference being the choice of the partition of unity functions, $\varphi_\alpha,\ \alpha = 1, \ldots, N$, and/or the enrichment functions, $L_{i\alpha}, i \in \mathcal{I}(\alpha)$.

The partition of unity approximation $u_h(\boldsymbol{x})$ can be written as

$$u_h(\boldsymbol{x}) = \sum_{\alpha=1}^{N} \varphi_\alpha(\boldsymbol{x}) \sum_{i\in\mathcal{I}(\alpha)} a_{i\alpha} L_{i\alpha}(\boldsymbol{x}) = \sum_{\alpha=1}^{N} \sum_{i\in\mathcal{I}(\alpha)} a_{i\alpha} \phi_{i\alpha}(\boldsymbol{x}) \tag{8}$$

where

$$\phi_{i\alpha}(\boldsymbol{x}) := \varphi_\alpha(\boldsymbol{x}) L_{i\alpha}(\boldsymbol{x}) \qquad \text{(no sum on } \alpha) \tag{9}$$

are denoted partition of unity, cloud or generalized finite element shape functions.

In the generalized finite element method the partition of unity is in general provided by linear Lagrangian finite element shape functions. The support ω_α of φ_α is then given by the union of the finite elements sharing a vertex node $\boldsymbol{x}_\alpha$. Figure 1 shows a one-dimensional finite element discretization. The partition of unity functions φ_α are the usual global finite element shape functions, the classical "hat-functions", associated with node $\boldsymbol{x}_\alpha$. The support ω_α is thus the union of the elements $\tau_{\alpha-1}$ and τ_α. The resulting shape functions are called generalized

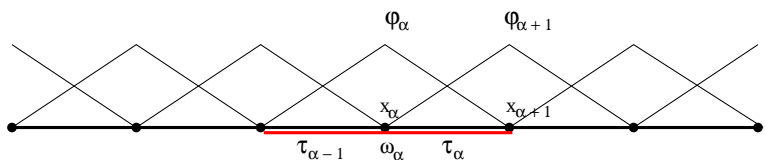


Figure 1. One-dimensional finite element partition of unity.

finite element shape functions. Figure 2 illustrates the construction of these functions in two dimensions. The number of partition of unity functions, N, is given by the number of vertex nodes, $\boldsymbol{x}_\alpha$, in the finite element mesh. The enrichment, $L_{i\alpha}$, and corresponding shape functions, $\phi_{i\alpha}$, are defined on a node-by-node or cloud-by-cloud basis. Each node may have a different set of enrichment functions [36, 69].

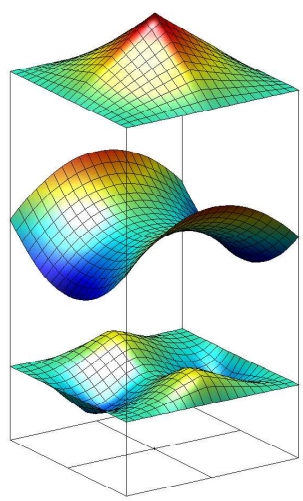


Figure 2. Construction of a generalized FE shape function using polynomial enrichment: In this example, φ_α is the function on the top, the enrichment function, $L_{i\alpha}$, is the function in the middle, and the generalized FE shape function, $\phi_{i\alpha}$, is the resulting bottom function.

Using property (5), we can write a partition of unity and, in particular, a generalized finite element approximation as

$$\begin{aligned} u_h(\boldsymbol{x}) &= \sum_{\alpha=1}^{N} \sum_{i\in\mathcal{I}(\alpha)} a_{i\alpha}\varphi_\alpha(\boldsymbol{x})L_{i\alpha}(\boldsymbol{x}) \\ &= \sum_{\alpha=1}^{N} \varphi_\alpha(\boldsymbol{x}) \left[a_{1\alpha} + \sum_{i\in\mathcal{I}(\alpha), i\neq 1} a_{i\alpha}L_{i\alpha}(\boldsymbol{x}) \right] \\ &= \underbrace{\sum_{\alpha=1}^{N} a_{1\alpha}\varphi_\alpha(\boldsymbol{x})}_{\text{regular interpolation}} + \underbrace{\sum_{\alpha=1}^{N} \sum_{i\in\mathcal{I}(\alpha), i\neq 1} a_{i\alpha}\varphi_\alpha(\boldsymbol{x})L_{i\alpha}(\boldsymbol{x})}_{\text{POU-based enrichment}} \end{aligned} \tag{10}$$

The above decomposition of $u_h(\boldsymbol{x})$ is used in the extended finite element method [17, 65, 81].

An a-priori error estimate for partition of unity approximations and, in particular, for the generalized finite element method, was proved by Babuška and Melenk [64]. The estimate says that if the partition of unity $\mathsf{POU}_N = \{\varphi_\alpha\}_{\alpha=1}^N$ satisfies some mild requirements and the error of the local approximations, $u_{h\alpha} \in \chi_\alpha(\omega_\alpha),\ \alpha = 1, \ldots, N$, are bounded by

$$\|u - u_{h\alpha}\|_{E(\omega_\alpha)} < \epsilon(\alpha, u), \qquad \alpha = 1, \ldots, N, \tag{11}$$

then the error of a partition of unity approximation, u_h, given by (1) is bounded by

$$\|u - u_h\|_{E(\Omega)} < C \left(\sum_{\alpha=1}^{N} \epsilon^2(\alpha, u) \right)^{1/2} \tag{12}$$

where $\|.\|_E$ denotes the energy norm and C is a constant. Details and proofs can be found in [33, 42, 64].

3. A GENERALIZED FINITE ELEMENT METHOD WITH CLUSTERING

The definition of a partition of unity given in the previous section (Cf. (2) and (3)) imposes only mild requirements on the partition of unity functions $\varphi_\alpha,\ \alpha = 1, \ldots, N$. In particular, given any partition of unity $\mathsf{POU}_N = \{\varphi_\alpha\}_{\alpha=1}^N$, another partition of unity, $\mathsf{POU}_{N'}$ with $N' < N$, can be created by adding elements $\varphi_\beta,\ \beta \in \mathcal{J}$, belonging to the original partition of unity POU_N, where $\mathcal{J}$ is an index set. The resulting set of functions has less elements then the original one if $\mathcal{J} \neq \emptyset$. This idea of creating new partitions of unity from existing ones is the cornerstone of the generalized finite element clustering technique presented here.

Unrefined or Clustered Partition of Unity:

Let $\mathsf{POU}_N = \{\varphi_\alpha\}_{\alpha=1}^N$ be a partition of unity composed of N functions. A *clustered or unrefined partition of unity* is then defined by

$$\mathsf{POU}_{N'} = \{\varphi_\alpha = \sum_{\beta\in\mathcal{J}(\alpha)} \varphi_\beta \ :\ \alpha = 1, \ldots, N,\ \alpha \notin \mathcal{J}_{clustered}\} \tag{13}$$

Here, $\mathcal{J}(\alpha)$ are the indices of the functions that are added to build the *clustered function* φ_α and $\mathcal{J}_{clustered}$ are the indices of the functions in the original set that are *not* in the *clustered*

or unrefined set $\mathsf{POU}_{N'}$. The dimension of the set $\mathcal{J}_{clustered}$ is an indication of the degree of clustering performed.

Let us consider some examples of clustered partitions of unity in the case of a finite element partition of unity in one- and two-dimensional spaces. From now on, we will refer to "partition of unity function associated with a node" simply by "node". In the figures, solid circles indicate nodes with a standard FE POU function; A solid square indicates a node with a clustered POU function, $\varphi_\alpha = \sum_{\beta\in\mathcal{J}(\alpha)} \varphi_\beta$, $\dim\{\mathcal{J}(\alpha)\} > 1$, as defined in (13); A circle indicates a node whose function $\varphi_\alpha \in \mathcal{J}_{clustered}$.

Figure 3 shows a one-dimensional finite element partition of unity composed of seven functions ($N = 7$). Figure 4 shows examples of clusterings performed on the original partition of unity of Fig. 3. In the figure, shaded boxes indicate regions where a partition of unity function is identically equal to one. In the case of the unrefined partition of unity shown in Fig. 4 we have, using the notation introduced in (13),

$$N = 7, \qquad N' = 5, \qquad \mathcal{J}_{clustered} = \{3, 6\}$$

$$\mathcal{J}(2) = \{2, 3\}, \qquad \mathcal{J}(5) = \{5, 6\}, \qquad \mathcal{J}(\alpha) = \alpha : \ \alpha = 1, 4, 7$$

If $\mathcal{J}(\alpha) = \alpha$, the partition of unity function φ_α is not modified.

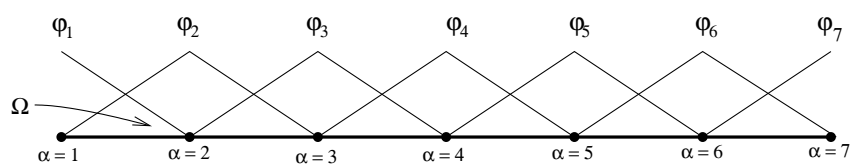


Figure 3. One-dimensional finite element partition of unity composed of seven functions ($N = 7$).

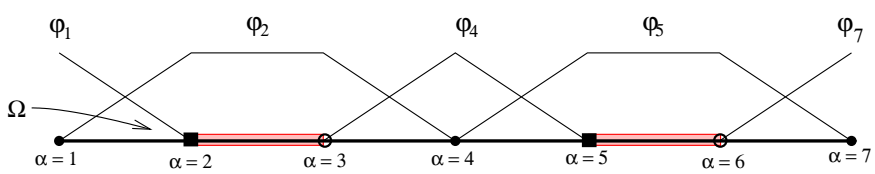


Figure 4. Unrefinement of the partition of unity shown in Fig. 3.

Figure 5 shows examples of clusterings or unrefinements performed on a uniform 3×3 mesh of quadrilateral finite elements. In the figure, shaded boxes indicate regions where a partition of unity function is identically equal to one. In this example, all partition of unity functions of the upper-right element were clustered into a single function while three functions of the lower-left element were clustered . For this example we have

$$N = 16, \qquad N' = 11, \qquad \mathcal{J}_{clustered} = \{2, 5, 12, 15, 16\}$$

$$\mathcal{J}(1) = \{1, 2, 5\}, \quad \mathcal{J}(11) = \{11, 12, 15, 16\}, \quad \mathcal{J}(\alpha) = \alpha : \ \alpha \notin \mathcal{J}_{clustered} \text{ and } \alpha \neq 1, 11$$

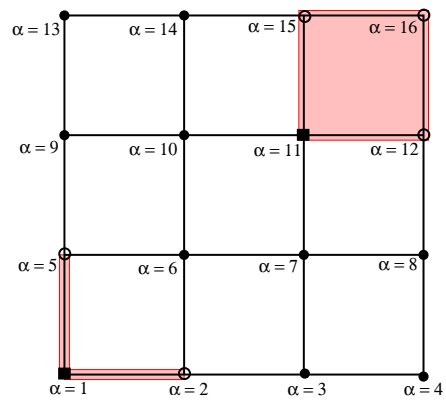


Figure 5. Clustering or unrefinement of a two dimensional finite element mesh. The clustering can be done quite arbitrarily. In this example, all nodes (i.e., partition of unity functions) of the right-upper element were clustered into a single node. As a result, the partition of unity function $\varphi_{11}(\boldsymbol{x})$ is identically equal to one over the element.

3.1. GFEM Shape Functions with Clustered POU

A clustered or unrefined partition of unity can be used in the construction of generalized finite element shape functions exactly as described in Section 2, Equation (9). Consider, for example, the case of the clustered POU shown in Fig. 4. The GFEM shape functions are given by

$$\phi_{i\alpha}(\boldsymbol{x}) = \varphi_\alpha(\boldsymbol{x}) L_{i\alpha}(\boldsymbol{x}), \qquad \alpha = \{1, 2, 4, 5, 7\}, \qquad i \in \mathcal{I}(\alpha)$$

where $\mathcal{I}(\alpha)$ is an index set for the enrichment functions used at node $\boldsymbol{x}_\alpha$. A *single* set of GFEM shape functions is used with each clustered POU functions φ_2 and φ_5.

The convergence properties of the GFEM shape functions with clustered POU follows from the a-priori error estimate discussed in Section 2.

The proposed unrefinement or clustering technique is conceptually simple, very generic, *can be used with any type of finite element and in any spatial dimension*. One important property of the technique is that there is no loss of information about the geometry of the domain as the finite element mesh is clustered since all the elements of the original (not clustered) mesh are used in the clustered mesh. The reduction in the number of degrees of freedom comes from the clustering of partition of unity functions associated with finite element nodes in the mesh.

Besides being used to reduce the size of a computational model (Cf. Section 5), the clustering technique described above can also be used, in a local sense, to handle non-matching finite element meshes (Section 6), meshes with very distorted elements, and even elements with negative Jacobians (Section 7). A clustered POU function is identically equal to one over part of its support as illustrated in Figs. 4 and 5. This property can be used to eliminate the well known linear dependence of generalized finite element shape functions [36, 87]. The multigrid method is not directly applicable to the linear system when the stiffness matrix is positive semi-definite [6]. Superconvergent points for the GFEM solution can be created when the POU functions are identically equal to one over part of their support [7]. These last two applications are not investigated in this paper. Additional discussion on the application of the proposed clustered GFEM can be found in [38, 39].

3.2. Polynomial Enrichment of Clustered Generalized FE Approximations

Generalized finite element shape functions, as defined in (9), are built from the product of a partition of unity function, φ_α, and a local enrichment function, $L_{i\alpha}$. If the partition of unity functions are linear finite element shape functions and the enrichment functions are of degree $p-1$, the resulting shape functions are of degree p [36].

Let us now consider the case of a clustered partition of unity as defined in (13). In this case, a POU function φ_α may be constant over an edge or face of an element, an entire element, or a combination of these entities. Therefore, the corresponding generalized FE shape functions do not have a well defined polynomial order. If the enrichment functions, $L_{i\alpha}$, are of degree $p-1$, the generalized FE shape functions are of degree p over edges, faces, and elements with partition of unity functions that are not clustered and of degree $p-1$ elsewhere. This must be taken into account when enriching a clustered discretization.

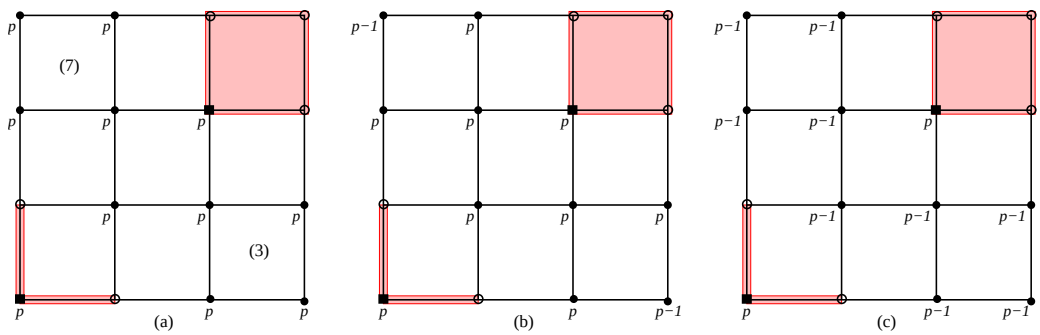


Figure 6. Nodal enrichment of the clustering shown in Fig. 5 using the proposed approaches.

We list three approaches to enrich a clustered generalized FE approximation to degree p. The clustering shown in Fig. 5 is used to illustrate and contrast the approaches. They are described with the aid of Fig. 6 which corresponds to the clustering illustrated in Fig. 5.

1. The first approach uses local enrichment functions, $L_{i\alpha}$, of degree (up to) p at every node. This is the most conservative approach and leads to the largest number of degrees of freedom among all strategies discussed here. Figure 6(a) illustrates the application of this approach to the clustering shown in Fig. 5. This approach leads to an approximation of degree $p+1$ over elements that do not have any clustered node (elements (3) and (7) of Fig. 6(a), for example).
2. In this approach, local enrichment functions of degree (up to) p are used at all the nodes of elements that have at least one partition of unity function that has been clustered. This guarantees that linear combinations of the shape functions of any element can reproduce polynomials of degree (up to) p, as is the case of standard finite elements of degree p. For this reason, we adopt this approach in all numerical experiments presented in this paper. Figure 6(b) illustrates the application of this approach to the clustering shown in Fig. 5.
3. The third approach uses local enrichment functions of degree (up to) p only at nodes associated with partition of unity functions that has been clustered. This will lead to the

smallest number of degrees of freedom among all enrichment strategies proposed here. Figure 6(c) illustrates this approach.

Each enrichment strategy leads, in general, to a different number of degrees of freedom and to a different approximation space. Clustered nodes have more degrees of freedom than unclustered ones in all strategies described above. However, the overall number of degrees of freedom is still reduced depending on the size of the clusters.

4. IMPLEMENTATION OF THE CLUSTERED GFEM

In this section, we briefly discuss some implementation aspects of a clustered generalized finite element method. We start by introducing some terminology. Hereafter, *node* means the partition of unity function associated with a node. *Cluster* denotes a set of partition of unity functions (nodes) that have been clustered (unrefined) into a single function (node) and a *clustered element* is an element with all nodes clustered into a single node.

4.1. Stiffness and Mass Matrix Calculation

The proposed mesh unrefinement technique preserves the representation of the geometry of the domain since all elements from the original (not clustered) mesh are used in the clustered mesh. This however implies that the computational work to compute the global stiffness and mass matrices are the same in the original and unrefined mesh. In fact, the cost to compute the matrices in the unrefined mesh can be larger than in the original mesh if the same global polynomial order is used. This is due to the fact that, as described in the previous section, the polynomial order of the transition elements between clustered and non-clustered elements is one degree higher than the polynomial order in the clustered elements. Therefore, these transition elements have more degrees of freedom and also require more integration points than the elements in the original mesh.

The cost of computing the matrices relative to the cost of solving the system of equations decreases as the number of degrees of freedom increases. In addition, the matrix computations can be fully parallelized almost trivially as in most finite element methods. The cost of matrix computations can also be reduced if we take advantage of special features of GFEM shape functions over clustered elements. Two approaches are discussed below.

4.1.1. Local Assembly of Clustered Elements All clustered elements belonging to a given cluster have the same set of shape functions and, therefore, the same set of degrees of freedom. This suggests that it is more efficient to locally assemble the stiffness/mass matrices of these elements into a local matrix and only then assemble this single matrix into the global matrix. This minimizes the number of direct accesses to the global matrices, which involves searching operations in the case of, for example, sparse matrix storages.

4.1.2. Enrichment Functions that Satisfy the Underlying PDE The local enrichment functions $L_{i\alpha}$ used in the construction of the GFEM shape functions can be chosen with great freedom. They can, for example, be functions that satisfy the partial differential equations being solved. Let us consider the case of a GFEM shape function of an element τ in which all partition of unity functions (nodes) have been clustered into a single function (node) φ_β as in

the case of the right-upper element shown in Fig. 5. The resulting partition of unity function over this clustered element τ is identically equal to one and a GFEM shape function for this element is given by

$$\phi_{i\beta}(\boldsymbol{x}) = \varphi_\beta(\boldsymbol{x}) L_{i\beta}(\boldsymbol{x}) = L_{i\beta}(\boldsymbol{x}), \qquad \boldsymbol{x} \in \text{ clustered element}, \tag{14}$$

according to (9). Therefore, *if $L_{i\alpha}$ satisfies the partial differential equations being solved, $\phi_{i\beta}$ also does.* This property can be used during the matrix calculations to transform domain integrals into boundary integrals. That is, instead of integrating over all elements in a given cluster, the integration can be done over the boundary of the cluster. For clusters composed of a large number of elements this can lead to substantial savings in the computation of global matrices. This technique however, requires the knowledge of local enrichment functions $L_{i\alpha}$ that satisfy the partial differential equations being solved. Similar techniques are used in the implementation of hybrid finite elements like the Voronoi finite element method proposed by Ghosh et al. [44, 45].

4.2. Clustering Techniques

The existing technology for finite element mesh unrefinement is limited to two-dimensional meshes [29, 53]. In the case of three-dimensional meshes, the unrefinement can, in general, only be done on meshes obtained by successive refinements [76]. That is, the unrefinement algorithms cannot be used to obtain a mesh that is coarser than the initial mesh.

The unrefinement procedure described in the previous section, in contrast, can be used with any type of finite element and in any spatial dimension. The clustering technique can be applied to as many nodes in a finite element mesh as needed. In addition, any set of nodes can be clustered into a single node.

The approximation properties of clustered GFE shape functions depend on the size and shape of their support which, in turn, depend on the set of nodes used in the construction of the clusters. Therefore, care must be taken in the selection of nodes used in each cluster. The optimal selection of these nodes for complex unstructured finite element meshes is not trivial.

We denote by a *clustering technique* an algorithm for the selection of groups of nodes used in clusters. We have investigated two techniques based on mesh partitioning algorithms for distributed memory parallel finite element processing. The first one uses Hilbert-Peano space filing curves [77] and the second one is based on the Metis algorithm [55] which is based on the theory of graphs. In both cases, either nodes or elements can be send to the mesh partitioner to create cluster data. The Metis-based technique produced usually more compact clusters, and node-based partitioning produced somewhat better clusters, and it was also algorithmically simpler, so this was selected as the default method in our code.

The performance of a clustering technique can be measured by (i) Convexity of the clusters. Non-convex clusters tend to produce stiff solutions and therefore should be avoided; (ii) Continuity of the clusters. The clustering algorithm should guarantee that each cluster is contiguous and possibly simply connected.

Clusters created using either the Metis- or the space filing curve-based technique can be further improved using the concept of Voronoi cells. Additional details on the implementation of clustering techniques used in our code can be found in [39].

4.2.1. Special Cases Our current implementation of the clustered GFEM is quite conservative and prohibits, with a few exceptions, clustering of nodes with boundary conditions or connected

to a non-GFEM element (typically, a structural element like a shell or beam). This restriction prevents unrefinement below a certain level even if only a few (and therefore large) clusters are requested. This is mostly a limitation of our current implementation and not of the clustering technique itself. We are currently investigating how to more efficiently handle clustering of nodes with boundary conditions.

5. NUMERICAL EXAMPLES: GENERALIZED MESH UNREFINEMENT

In this section, examples are solved using the GFEM with clustering presented in previous sections. The CPU times are reported in seconds for an Athlon 2400 Linux computer. Most of examples presented are small and as such the time spend on the computation and assembly of global matrices is large compared with the linear solver time. None of the techniques discussed on Section 4.1 were used to optimize the matrix computation and assembly. Thus, CPU times are relative only, and in particular should not be used to judge the effectiveness of the method in terms of CPU time.

Increasing the size of a cluster of nodes or elements in a GFEM discretization reduces the total number of degrees of freedom and has the same effect as using a coarser (unrefined) standard finite element mesh. Therefore, the discretization error increases with the size of the clusters. On the other hand, the CPU time and memory required for the solution of the problem is reduced.

Given a finite element mesh, it is also possible to select the size of the clusters such that a quick solution that provides a rough idea of the exact solution can be computed. This kind of analysis is difficult to be done using the standard finite element method since it would entail the construction of coarser meshes, a task that in most cases is not feasible to automate or is very time consuming.

5.1. Patch Test

The patch test illustrated in Fig. 7 is solved in this section. This particular type of the patch test corresponds to form C described in [92]. The domain is 10 units long in both x and y directions with a constant thickness of 0.1 units. The mesh is also illustrated in Fig. 7. A single layer of elements is used in the z direction. The elements have an aspect ratio of about 10:1. Unity tractions are applied as illustrated in Fig. 7 and point Dirichlet boundary conditions are prescribed in order to prevent rigid body motion. The material is isotropic and linearly elastic with Young's modulus $E = 1.0$ and Poisson's ratio $\nu = 0.3$.

The proposed clustered GFEM is used with polynomial order $p = 1$ and 20 clusters. Figure 8 illustrates the clusters built using a Metis-based algorithm as described in Section 4.2. Stress components are computed at element nodes and are exact to within round-off error (double precision was used which produced round-off error of less than 10^{-12} in the quantities computed).

5.2. Cantilever Beam

The cantilever beam shown in Fig. 9 is solved in this section using the GFEM with and without clustering. The mesh has elements with 1:20 aspect ratio. This example tests the robustness of the GFEM to this kind of mesh and also the performance of the proposed GFEM with

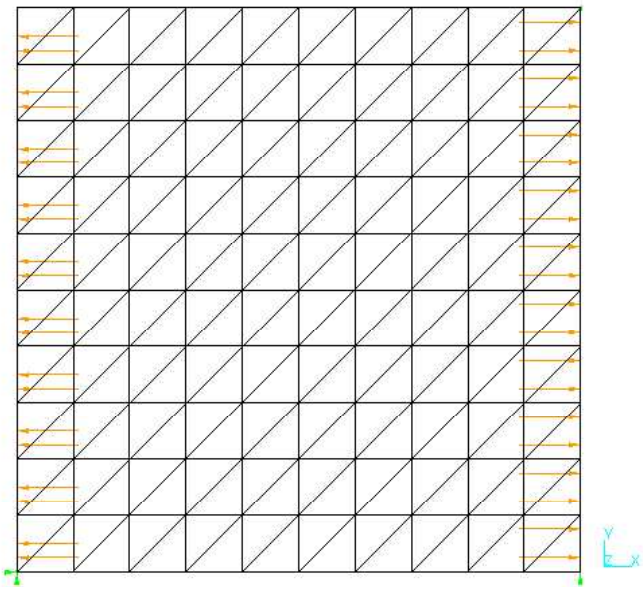


Figure 7. Top view of tetrahedral mesh used for patch test. Unity tractions are applied at the left and right hand side faces of the domain.

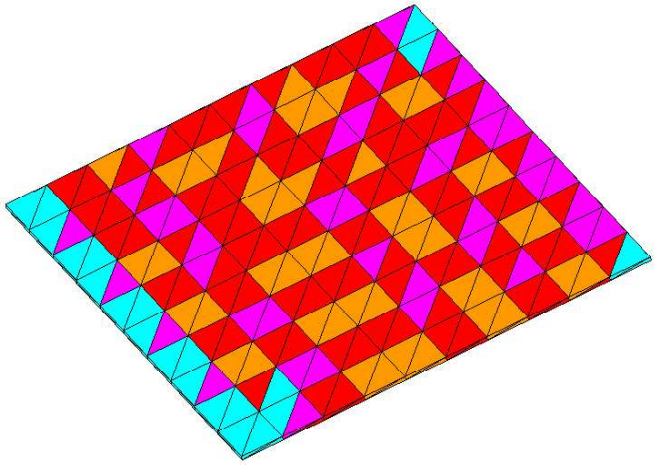


Figure 8. Clustered GFEM discretization. The color of the elements indicate the number of nodes clustered: orange = all nodes, red = three nodes, purple = two nodes, blue = none (standard GFEM). The mesh has 600 elements and 20 clusters were requested.

clustering.

The model without clustering has 24 nodes that are enriched with polynomial functions of degree (up to) $p = 2$. The enrichment functions are from the same family illustrated in (6). The resulting GFEM shape functions are of degree $p = 3$. The number of degrees of freedom for this model is

$$\text{Number of dofs} = 24 * 10 * 3 = 720$$

since there are 10 polynomials of degree less or equal $p = 2$ in three-dimensions.

A single cluster is used for the GFEM model with clustering. The nodes with boundary conditions (four at $x = 0.0$ and four at $x = L$), are not clustered, as discussed in Section 4.2. These nodes are enriched with polynomial functions of degree (up to) $p = 2$. All other nodes are clustered into a single node. This node is enriched with polynomial functions of degree (up to) $p = 4$. The number of degrees of freedom for this model is

$$\text{Number of dofs} = 8 * 10 * 3 + 1 * 35 * 3 = 345$$

since there are 35 polynomials of degree less or equal $p = 4$ in three-dimensions. This is less than half the number of degrees of freedom of the GFEM model without clustering.

The material parameters, mesh and domain dimensions are indicated in Fig. 9. Table I summarizes the results. The stress component σ_{xx} is reported at the center of top surface with coordinates $\boldsymbol{x} = (12.5, 0.5, 1)$. The results are compared with those of the Euler-Bernoulli beam theory. It can be observed that the GFEM method, with or without clustering, provides very accurate results.

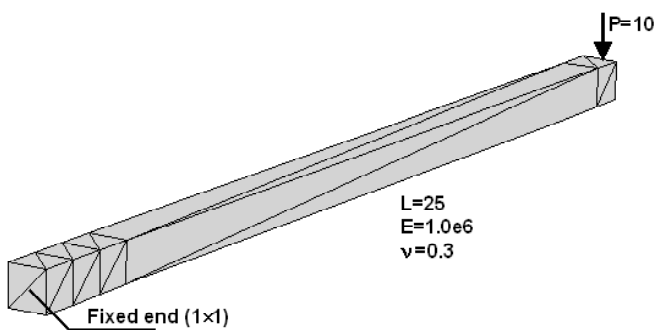


Figure 9. Simple cantilever beam of length $L = 25$ and square cross-section of dimensions 1×1. The tetrahedral mesh has elements with 1:20 aspect ratio.

Table I. Results for the analysis of the cantilever beam shown in Fig. 9.

Method:	No clustering	One cluster	Exact
# eqns.	720	345	(beam theory)
Max. displ.	0.624	0.618	0.625
Strain energy	3.120	3.090	3.125
σ_{xx} at (12.5,0.5,1)	745	735	750

5.3. Bulky Three-Dimensional Model

The three-dimensional model shown in Fig. 10 is used in this section to investigate the effects of mesh clustering. The material is assumed to be isotropic and linear elastic with Young's modulus $E = 210,000$ and Poisson's ratio $\nu = 0.3$. Tetrahedral meshes with up to $820,142$ elements are used. The component is fixed at both vertical openings and a unity pressure is applied on faces of elements at the upper opening, as illustrated in Fig. 10.

Figure 11 shows a clustered GFEM discretization. In this example, 120 clusters were requested and a Metis-based algorithm, as discussed in Section 4.2, was used on a mesh with $15,527$ elements. The color of the elements indicate the number of nodes clustered in the element: orange = all nodes, red = three nodes, purple = two nodes, blue = none. Our current clustering algorithm, as discussed in Section 4.2.1, does not cluster nodes with boundary conditions. This explains the large number of blue elements around the vertical openings in Fig. 11.

The bracket has singularities along the non-convex edges. Clustering nodes in the neighborhood of these edges substantially increases the discretization error and should therefore be avoided. This is not taken into account by our clustering algorithm. Figure 11 shows cluster of nodes across non-convex edges of the bracket.

Large models could not be solved on a 32-bit computer without clustering. Also, a large number of clusters, especially for $p = 2$, required more memory than available.

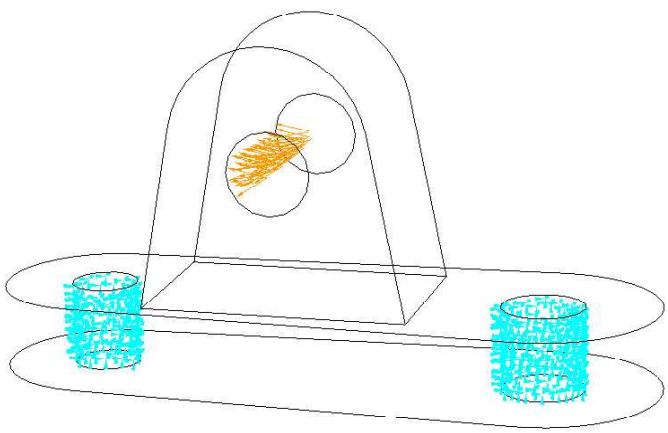


Figure 10. Bulky three-dimensional model used to investigate the proposed unrefinement algorithm. The component is fixed at both vertical openings and a unity pressure is applied on faces of elements at the upper opening as indicated in the figure.

Tables II and III summarize the results for meshes with $92,149$ and $820,142$ elements, respectively. The number of equations and corresponding CPU time, memory usage and computed strain energy are listed for different levels of mesh clustering and polynomial order equal to $p = 1$ and $p = 2$. Reducing the number of clusters leads to a larger number of nodes in each cluster and therefore to fewer equations in the model and larger discretization errors. The computed strain energy decreases quite substantially with clustering of nodes. The clustering of nodes in the neighborhood of the non-convex edges contributes to this behavior, as discussed above.

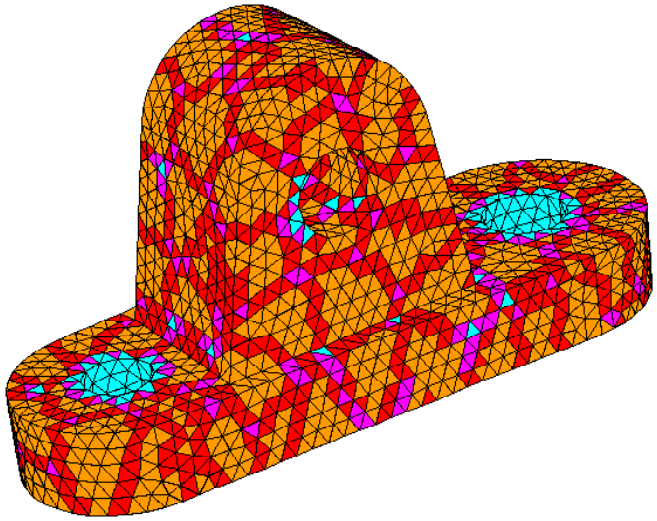


Figure 11. Clustered GFEM discretization built using a Metis-based algorithm. The color of the elements indicate the number of nodes clustered: orange = all nodes, red = three nodes, purple = two nodes, blue = none (standard GFEM). The mesh has $15,527$ elements and 120 clusters were requested.

The results clearly show that it is possible to reduce the CPU time and memory requirements using the proposed clustering technique. The tables show that clustering can be used to solve problems that would otherwise be beyond the available computational resources. On the other hand, the solution computed on models with large clusters is of lesser quality, as expected.

It can be observed from Table III, that models with 2000 clusters have substantially fewer degrees of freedom than those without clustering. Requesting fewer clusters, however, do not further reduce the number of equations substantially. This happens because, in our current implementation, nodes with applied boundary conditions are never clustered (Cf. Section 4.2.1). Therefore, this creates a lower bound on the number of degrees of freedom in a model.

Table II. Results for the analysis of the three-dimensional model shown in Fig. 10. Mesh with $92,149$ elements.

# of clusters	# Eqns	CPU time	RAM(MB)	Strain Energy
$p = 1$				
No clustering	61,041	60.05	70	1.289970
200	21,072	28.76	116	0.881289
20	18,888	23.91	99	0.466014
$p = 2$				
No clustering	244,164	2,018.01	549	1.351846
200	39,738	109.32	169	1.202554
20	34,296	82.78	137	0.910704

Table III. Analysis on a mesh with 820, 142 elements. Empty entries correspond to models that could not be solved on a 32-bit computer.

# of clusters	# Eqns	CPU time	RAM(MB)	Strain Energy
$p = 1$				
No clustering	505,209	–	–	–
2000	129,936	468.53	387	1.219049
200	108,300	350.08	334	0.806180
20	106,116	332.57	328	0.397022
$p = 2$				
No clustering	2,020,836	–	–	–
2000	250,608	–	–	–
200	196,518	927.34	843	1.258240

6. NON-MATCHING MESHES

In this section, we describe how the clustering technique presented in Section 3 can be used to handle finite element meshes that have localized mesh inconsistencies. This kind of mesh can arise in several applications. It is not uncommon, for example, to have complex mechanical parts divided in subparts that are geometrically simpler and meshed independently of each other. The problem then is how to "tie" the subparts together such that the behavior of the assembly can be accurately represented.

Several methods to handle non-conforming or non-matching finite element meshes have been proposed in the literature. Among the widely used ones are the mortar element or Lagrange Multiplier method [19, 43, 74, 90] and rigid elements or point collocation in solid mechanics [12, 91]. Lagrange Multiplier methods [5] lead to a saddle point problem and the Babuška-Brezzi condition [20, 67] must be satisfied for stability. This is not trivial to accomplish especially when the interfaces between the subparts have corners or other types of singularities or the interfaces are between different materials. In addition, this approach leads to non-positive definite matrices.

In the rigid-element or point-collocation approach, the continuity of the solution between subparts is imposed only at a discrete set of points. Therefore, the behavior of the whole assembly depends on the number and configuration of rigid elements/collocation points used. Moreover, the solution in the neighborhood of the interfaces exhibits artificial stress singularities that do not disappear with mesh refinement.

Another approach to connected dissimilar finite element meshes is to modify the formulation of the elements at the non-matching interface such that first-order patch tests are passed [30, 31]. This method is recommended only for the case of linear elements since it leads to sub-optimal convergence rates in the case of higher order elements. Mixed finite elements [2, 3], discontinuous Galerkin finite element methods [21], Nitsche's method [13, 52], interface elements based on moving least-square approximations [25, 26], among others, have also been used to handle the problem of non-matching meshes.

Huang and Xu [54] and Bacuda and Xu [11] have presented the theory of a finite element method for non-matching overlapping grids based on the concept of a partition of unity. The non-overlapping case of their method is related to the approach described below.

6.1. Local Clustering for Non-Matching Interfaces

The technique presented in this section is basically a local application of the shape function clustering technique presented in Section 3. As such, it can be used with standard variational principles while rendering a solution that is continuous along the entire interface between inconsistent meshes.

The technique is better described with the aid of a few examples. Let us consider as a first example the mesh shown in Fig. 12(a). This mesh can not be used, as it is, in the classical finite element method since the global shape functions associated with nodes 3, 4 and 5 are discontinuous along the boundary between elements τ_3 and τ_1, τ_2. These global shape functions are given by

$$\varphi_3 = N_3^{\tau_1} \cup N_3^{\tau_3}, \quad \varphi_4 = N_4^{\tau_1} \cup N_4^{\tau_2}, \quad \varphi_5 = N_5^{\tau_2} \cup N_5^{\tau_3} \tag{15}$$

where $N_\alpha^{\tau_i}$ represents the bilinear shape function of element τ_i associated with global node α.

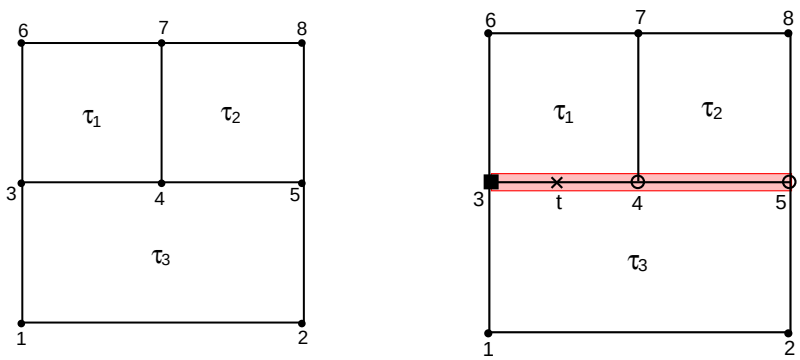


Figure 12. (a) Non-matching mesh between element τ_3 and elements τ_1 and τ_2. (b) Representation of the clustered partition of unity used to handle the non-matching mesh of Fig. 12(a). The partition of unity function $\varphi_3(\boldsymbol{x})$ is identically equal to one along the non-matching interface and it is therefore continuous.

These discontinuous global functions can be clustered into a single function using the definition of a clustered POU given in (13) with

$$N' = 6, \qquad \mathcal{J}_{clustered} = \{4, 5\}$$

$$\mathcal{J}(3) = \{3, 4, 5\}, \qquad \mathcal{J}(\alpha) = \alpha : \ \alpha = 1, 2, 6, 7, 8$$

This gives

$$\varphi_3 \doteq \varphi_3 + \varphi_4 + \varphi_5 \tag{16}$$

A representation of the clustered partition of unity is shown in Fig. 12(b). Let us now show that this partition of unity is continuous. Let $\boldsymbol{t} \in \bar{\tau}_1 \cap \bar{\tau}_3$, as shown in Fig. 12(b). If this function is computed from element τ_1 we have

$$\varphi_3|_{\tau_1}(\boldsymbol{t}) = N_3^{\tau_1}(\boldsymbol{t}) + N_4^{\tau_1}(\boldsymbol{t}) = 1 \tag{17}$$

since the only non-zero shape functions of element τ_1 along the edge $\bar{\tau}_1 \cap \bar{\tau}_3$ are $N_3^{\tau_1}$ and $N_4^{\tau_1}$ and the shape functions of a Lagrangian finite element constitute a partition of unity. Similarly, if function φ_3 is computed from element τ_3 we have

$$\varphi_3|_{\tau_3}(\boldsymbol{t}) = N_3^{\tau_3}(\boldsymbol{t}) + N_5^{\tau_3}(\boldsymbol{t}) = 1 \tag{18}$$

Therefore $\varphi_3|_{\tau_1}(\boldsymbol{t}) = \varphi_3|_{\tau_3}(\boldsymbol{t})$ and the function φ_3 is continuous at $\boldsymbol{t}$. The same argument can be used at any other point along the interface between element τ_3 and τ_1, τ_2.

As a second example, let us consider the finite element mesh shown in Fig. 13(a). The global shape functions associated with nodes $1-7$ are discontinuous. These functions can be clustered into a single function using again the definition of a clustered POU given in (13) with

$$\mathcal{J}_{clustered} = \{2, \ldots, 7\}$$

$$\mathcal{J}(1) = \{1, \ldots, 7\}, \qquad \mathcal{J}(\alpha) = \alpha : \ \alpha \notin \mathcal{J}_{clustered} \text{ and } \alpha \neq 1$$

This gives

$$\varphi_1 \doteq \sum_{\beta=1}^{7} \varphi_\beta \tag{19}$$

which can be shown to be continuous using the same arguments as in the previous example. A representation of the resulting clustered partition of unity is shown in Fig. 13(b). This continuous partition of unity can then be used to build GFEM shape functions as described in Section 3.1.

Based on the examples above, an algorithm to handle a non-matching mesh is proposed below.

Let $\Gamma_m \subset \Omega$ be a non-matching interface. This is a curve in two-dimensional domains and a surface in the case of three-dimensional domains. We assume that there are no gaps between the non-matching meshes along Γ_m. A method to handle non-matching grids with gaps between meshes can be found in [43]. Let

$$\mathcal{J}_m = \{\alpha \ : \ \varphi_\alpha(\boldsymbol{x}) \neq 0, \ \boldsymbol{x} \in \Gamma_m\}$$

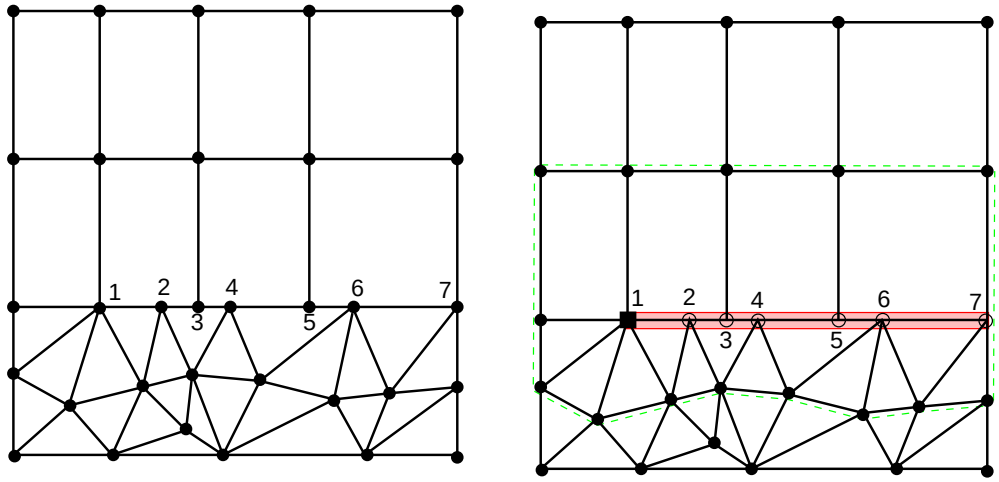
denote the index set of partition of unity functions that are discontinuous due to the non-matching interface Γ_m. Let β be an arbitrary node with index in $\mathcal{J}_m$ ($\beta = 1$ in the example of Fig. 13). A continuous, clustered, partition of unity can then be built using (13) with

$$\begin{aligned} \mathcal{J}_{clustered} &= \mathcal{J}_m - \beta \\ \mathcal{J}(\beta) &= \mathcal{J}_m \\ \mathcal{J}(\alpha) &= \alpha : \ \alpha \notin \mathcal{J}_{clustered} \text{ and } \alpha \neq \beta \end{aligned}$$

6.2. Techniques to Handle Large Non-Matching Interfaces

The technique described above is quite general and can be used with any type of element that can be clustered using the method presented in Section 3. One limitation, however, is the size of the support of the GFEM shape functions along a non-matching interface Γ_m. The support of a GFEM shape function is given by the intersection of the supports of a POU and an enrichment function (Cf. (9)).

The support of a clustered partition of unity function at a non-matching interface is equal to the union of the support of all originally discontinuous functions along the interface. In



(a) Non-matching mesh.

(b) Representation of the clustered partition of unity used to handle the non-matching mesh.

Figure 13. Non-matching mesh composed of quadrilateral and triangular elements and corresponding clustered partition of unity. The dashed lines in Fig. (b) indicate the boundary of the support of the clustered function φ_1.

addition, only one set of GFEM shape functions is used with each clustered POU function, as discussed in Section 3.1. In case of the clustered POU represented in Fig. 13(b), for example, the only POU function along the non-matching interface is φ_1. The support of φ_1 is equal to the union of all elements sharing nodes $1-7$. This support is represented by the dashed lines in the figure. If polynomial enrichments are used at node $\boldsymbol{x}_1$, the support of the GFEM shape functions will be equal to that of φ_1 since polynomials do not have compact support. The approximation at a non-matching interface is analogous to a single finite element with size equal to the union of all elements with nodes on the interface. This may sometimes lead to problems.

The size of the support of a clustered partition of unity function can be minimized by clustering only nodes that are strictly necessary to make the approximation continuous across a non-matching interface. These are the nodes with index in the set $\mathcal{I}_m$ defined above. However, this does not guarantee that the supports of the shape functions at the interface are small enough to give acceptable results. Reducing the size of the non-matching interface will, of course, reduce the supports of the corresponding shape functions at the interface but this is not always feasible. If, however, the solution is smooth along a non-matching interface then the quality of the GFEM approximation can be controlled through p-enrichments. This approach is used in the example presented in Section 6.3.

Alternatively, the size of the support of GFEM shape functions along a non-matching interface, and their approximation properties, can be controlled by using enrichment functions

with compact support. These enrichment functions can be, for example, *hp* cloud shape functions [41, 42] which are, themselves, partition of unity shape functions with compact support and arbitrary degree of smoothness.

6.3. Numerical Examples: Non-Matching Meshes

6.3.1. Plate with a Hole A three-dimensional model of a plate with a hole is solved in this section. The finite element model includes both hexahedral and very long tetrahedral elements and has a large non-matching interface. Figure 14 shows a top view of the mesh. The technique described in Section 6.1 is used to handle the non-matching mesh. The material is assumed to be isotropic and linearly elastic with Young's modulus $E = 3000.0$ and Poisson's ratio $\nu = 0.3$. Symmetry boundary conditions and a unity traction are applied on the right and left faces of the domain, respectively.

The non-matching interface is quite large but since the solution of this problem is smooth along the non-matching interface, p enrichment can approximate the solution well [86]. The hexahedral elements are enriched to $p = 2$ in the in-plane directions and to $p = 1$ in the out-of-plane direction while the tetrahedrals are enriched to $p = 4$ in the in-plane directions and to $p = 1$ in out-of-plane direction. Orthotropic enrichment of GFEM discretizations is described in [34]. The number of equations for this discretization is equal to $49,800$.

The computed displacements and stresses are shown in Figs. 15 and 16, respectively. They exhibit very good symmetry with respect to the vertical and horizontal planes of symmetry.

The analytic solution for an infinite two-dimensional strip in plane stress gives $\sigma_{xx} = 3.25$ at the top and bottom of the hole. The GFEM results compare quite favorably with this solution: $\sigma_{xx} = 3.38$ at the top of the hole and $\sigma_{xx} = 3.14$ at the bottom of the hole. These are raw stress values. No averaging or smoothing was done.

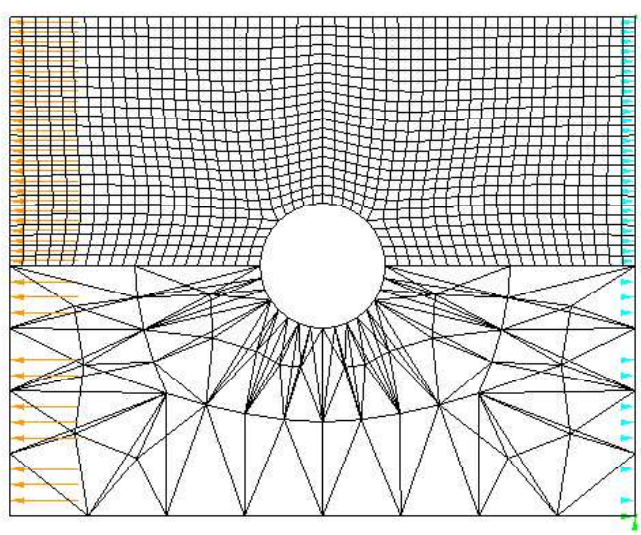


Figure 14. Top view of non-matching mesh and boundary conditions for a plate with a hole. A unity traction and symmetry boundary conditions are applied on the left and right faces of the domain, respectively.

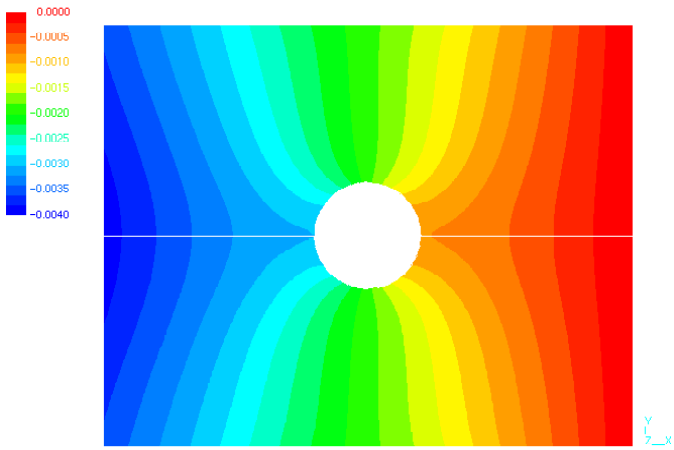


Figure 15. Contour plot of the horizontal component of the displacement vector.

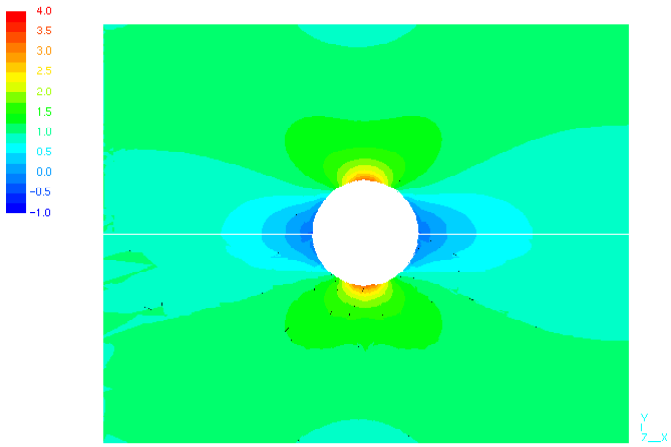


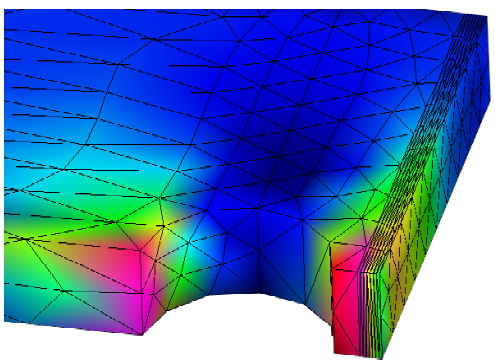
Figure 16. Computed stress component σ_{xx}.

7. ELEMENTS OF UNACCEPTABLE QUALITY

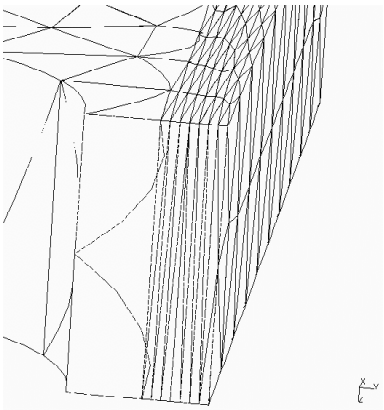
It is well known by finite element practitioners and from a-priori error estimates that the quality of a finite element approximation to the solution of boundary or initial value problems depends on the aspect ratio of the elements in the mesh. The aspect ratio of an element is defined as the ratio between the diameter of the smallest circle (or sphere) that circumscribes

the element and the diameter of the largest circle (or sphere) that can be inscribed in the element [67, 70].

Elements of unacceptable aspect ratio can appear in a finite element mesh for several reasons. During the mesh generation, for example, the transition zones between small and large elements, in general, contain elements with large aspect ratios. This can only be avoided, in most cases, by substantially increasing the number of elements in the mesh. Even meshes initially containing only elements of good quality can become unacceptable in the course of a simulation involving nodal movement. This happens, for example, during the shape optimization of mechanical parts (cf. Figs. 17(a) and 17(b)) or in the analysis of problems involving finite deformations using a Lagrangian formulation. A common approach to handle this situation is to remesh the entire domain whenever elements of unacceptable quality appear. This cannot always be done without the intervention of the user and in general leads to a loss of accuracy caused by mapping the solution between meshes.



(a) The mesh seems correct because element edges are usually drawn as straight lines.



(b) Zoom on the area with distorted elements. Some edges of the elements near the hole cross over their neighbors.

Figure 17. Elements distorted due to translation of a hole during the shape optimization of a mechanical part.

In this section, we describe how elements of unacceptable quality for the finite element method can be handled in the GFEM using a local version of the element clustering technique proposed in Section 3. We concentrate on the case of elements with a Jacobian equal or close to zero at some point(s) in the element. This causes the transformation from the reference or master coordinate system to the physical or global coordinate system to be non-invertible. This inverse mapping is used in most finite element implementations to compute derivatives of the shape functions of the element in the physical directions. Elements with this type of problem arise quite often during automatic mesh generation around curved boundaries as illustrated in Fig. 18. Most mesh generators for elements with quadratic geometry, first generate a mesh with straight edges or faces and then move the center nodes on the edges or faces to the actual

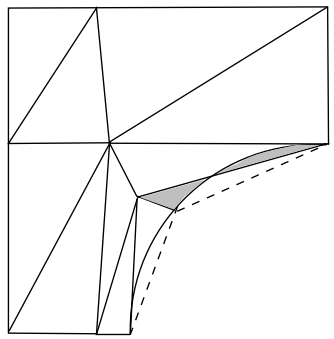


Figure 18. Element of unacceptable quality near a curved boundary. The mapping from the reference element to the shaded element cannot be inverted. If the element is clustered, however, the inverse mapping is not needed.

geometry. This last step is very prone to deteriorating the quality of elements. An example of a mesh generated using this approach is shown on Fig. 18. The edge of the shaded element that is on the curved boundary was moved from the dashed line to the position shown. The resulting element has a negative or zero Jacobian at several points and is therefore unacceptable.

Let us consider more closely how the shape functions and their derivatives are computed in an element τ, in which all partition of unity functions (nodes) have been clustered into a single function (node) φ_β as in the case of the shaded elements shown in Figs. 18 and 5. Such an element is said to be clustered. The resulting partition of unity function over element τ is identically equal to one and a GFEM shape function $\phi_{i\beta}$ for this element is given by

$$\phi_{i\beta}(\boldsymbol{x}) = \varphi_\beta(\boldsymbol{x}) L_{i\beta}(\boldsymbol{x}) = L_{i\beta}(\boldsymbol{x}) \qquad \boldsymbol{x} \in \text{clustered element} \tag{20}$$

according to (9). The derivatives of these functions with respect to the global directions (x_1, x_2, x_3) are therefore given by

$$\frac{\partial \phi_{i\beta}}{\partial x_a} = \frac{\partial L_{i\beta}}{\partial x_a} \qquad a = 1, 2, 3 \tag{21}$$

The computation of the derivatives of enrichment functions $L_{i\beta}$ does not use the inverse of the Jacobian matrix since these functions are independent of the partition of unity. Therefore, the calculation of the GFEM shape functions and their derivatives over clustered elements is not affected by the shape of the elements. This property of the GFEM shape functions over clustered elements suggests the following strategy to handle elements of unacceptable quality:

- Check the element Jacobians at sampling points and cluster those with near zero or negative values. Here, the sampling points can be, for example, the vertices of elements, integration points, etc.
- Check the quality of elements in a given mesh by computing the aspect ratio and/or any other measure of element quality and cluster elements of unacceptable quality.
- Enrich the approximation to the desired order.

The approach to handle elements of unacceptable quality can be used with any type of finite element in two- and three-dimensional meshes. In addition, it does not require any modification of the finite element mesh.

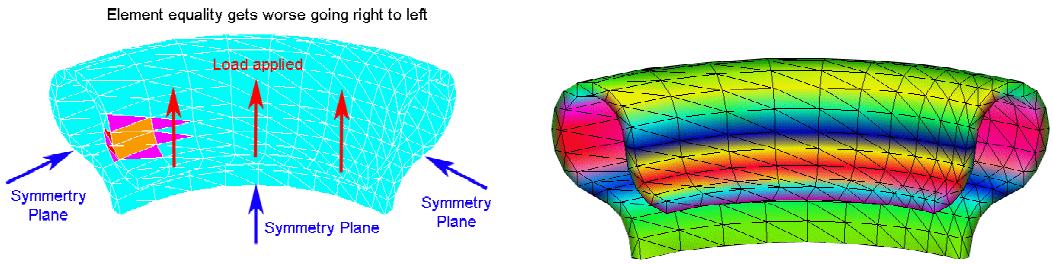
The numerical integration over an element with negative volume ("flipped-out") or over portions of an element which has a negative Jacobian, is equivalent to the subtraction of the element contribution. When there are two overlapping elements, the overlapped area is integrated three times–once with a negative sign and twice with positive sign. Therefore the final result is correct. This also works correctly for elements which are "flipped-out" and extend outside of the actual domain, which often happens on curved boundaries as illustrated in Fig. 18. In this case, the volume outside of the actual domain is covered by two elements and their contributions cancel out. In the case of standard finite elements this approach works for volume computation (which is equivalent to the integration of a constant function). However, it does not work in general for the integration of the stiffness or mass matrices, for example. In standard finite elements, each element integrates its own shape functions and therefore the contributions do not cancel out. In the case of a clustered GFEM subdomain, the same shape functions are defined over the entire cluster and the finite elements inside of the cluster serve only as numerical integration domains.

7.1. Numerical Examples: Elements of Unacceptable Quality

The examples in this section illustrate the proposed technique to handle elements of unacceptable quality. The meshing errors in the examples below are not detected by some commercial finite element codes while some others do and either refuse to use the mesh or require the use of commands that force the computation of solution, while warning that results may be incorrect. Our GFEM code does not require such warning, because, with proper clustering, as described in the previous section, the formulation is mathematically correct, even when elements with negative Jacobian are present.

7.1.1. Hoop with Negative Jacobian Elements Figure 19(a) shows the geometry and boundary conditions for the model of a quarter of a hoop. The reference solution for this problem was obtained using the *hp*-adaptive finite element code PHLEXsolid [75] and a hexahedral mesh. The GFEM results presented below are normalized with respect to this reference solution. The problem was solved with the GFEM using the mesh of curved tetrahedral elements shown in Fig. 19(a). The elements on the left side of the figure are longer than those on the right side, and as a result of the curved edges, four elements (shown in orange) have a negative Jacobian at some vertices. This tetrahedral model was solved using the GFEM with clustering of the elements with negative Jacobian as described in Section 7. With second order approximation, the normalized maximum von Mises stress along the left and right planes of symmetry were 0.87 and 0.96, respectively. By using third order approximation, the normalized maximum von Mises stress was 0.98 and 0.99, respectively. Figure 19(b) shows the Von Mises stress distribution on the clustered mesh and $p = 3$.

7.1.2. Elements with Zero Jacobian As a second example, we consider the wheel rim model shown in Fig. 20. The mesh contains elements on the surface that has exactly zero volume. Such elements are often generated by automatic mesh generators on highly curved domains. In this example, five such elements are present. Most finite element codes require manual "pruning" of these elements. The problem was solved with the GFEM using the mesh of Fig. 20. The computed maximum von Mises stress was within 6% of the maximum value computed on a mesh with all zero volume elements removed. The computed von Mises stress distribution



(a) (b)

Figure 19. (a) Tetrahedral element model with boundary conditions. The curved elements with negative Jacobian, showed in orange, are clustered. (b) Von Mises stress distribution on clustered mesh and $p = 3$.

in the region with maximum stress is shown in Fig. 21. The mesh with zero-volume elements shown in Fig. 20 was used in the computations.

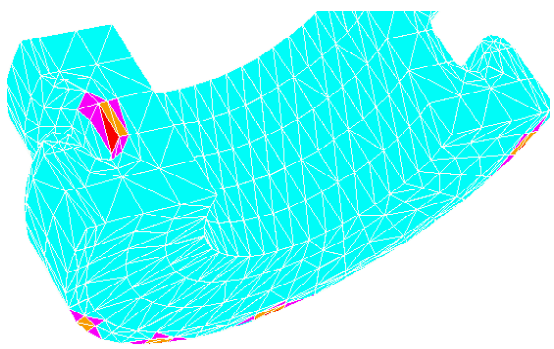


Figure 20. Wheel rim model: Zero Jacobian elements. Five clusters with faulty elements are shown. The color of the elements indicate the number of nodes clustered (orange = all nodes, red = three nodes, purple = two nodes)

8. SUMMARY AND CONCLUSIONS

The GFEM presented in this paper addresses difficulties with meshing complex three-dimensional geometries by reducing the quality requirements for the initial mesh. The combination of the proposed GFEM with a-posteriori error estimation and *hp* adaptivity [1, 10, 86] may allow the creation of fully automatic analysis packages, where automatic mesh generators are a hidden part of the process, and the computed solution is of guaranteed accuracy even if the automatically generated mesh is not ideal.

The proposed technique may have a great impact in computer engineering and scientific

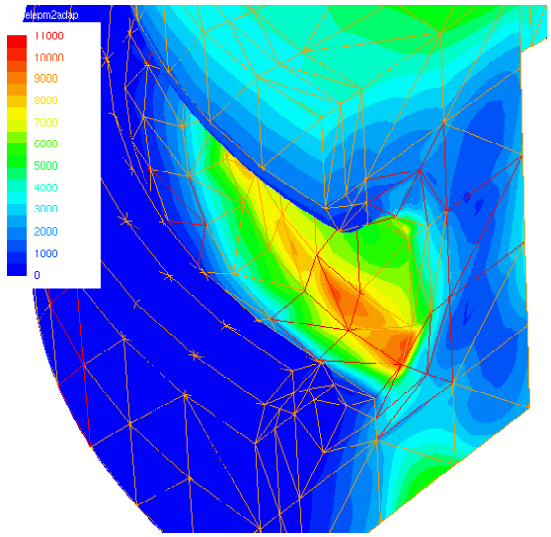


Figure 21. Von Mises stress distribution computed on the mesh with zero-volume elements shown in Fig. 20.

simulations by providing the following benefits:

- *A virtually* 100 *percent success rate in automatic discretization of complex CAD geometries without user intervention.* Given the relaxed requirements for an acceptable GFEM mesh (high element distortions, negative Jacobians, collapsed elements, element mis-matches), today's automated tetrahedral mesh generators can practically guarantee successful volume meshing of geometries that are correctly defined, i.e., without gaps, overlaps, etc. The automation of the meshing generation process will make it feasible to investigate a much broader range of alternative designs, thus leading to better designs in a shorter period of time;
- Reduction of mesh size through "clustering". *This guarantees, within reasonable bounds, the solution of large models using only the computational resources available to the analyst.* Our current implementation, however, prevents unrefinement of a model below a certain level, as discussed in Section 4.2.1. The CPU times and memory requirements presented in Section 5, show that for large meshes it is possible to reduce problem size and CPU usage in exchange for the accuracy of the solution. Thus eventually it should be possible to implement a user controllable "slider", to select how much accuracy or how much computer resources should be used to solve a given problem. The automatic model reduction capability may also be used to perform convergence analysis using several levels of mesh resolution with little or no user intervention;
- The capability of mesh unrefinement considerably expands the scope of mesh adaptivity. With today's technology, a mesh can not be unrefined (coarsened) past the initial mesh even if it is too fine for the requested accuracy. In addition, with the proposed mesh reduction, the representation of the geometry of the domain is preserved, i.e., there is *no* loss in the approximation of the geometry of the domain. This is in contrast with traditional mesh coarsening and it is unlikely that such a feature will be matched in the

near feature using existing meshing technology.

In addition, the method presented herein retains all of the attractive features of classical finite element methods. In particular:

1. The shape functions are polynomials and the integration of the matrices is done with the aid of the so-called master element exactly as in the classical finite element. This is in contrast with most meshless methods in which the numerical integration of the matrices is a major issue;
2. The computational performance is essentially the same as a finite element method when the same mesh is used. This is mainly due to the previous property;
3. It can be applied to solve the same classes of problems solvable by the finite element method (linear and non-linear, static, time-dependent, eigenvalue problems, etc.) In addition, a GFEM discretization can be mixed with classical finite elements if such a need arises.

To our knowledge, there is no technique available in the literature that can deliver all the features above.

As topics for future work we can mention:

- *Improvement of clustering algorithms:* Clustering techniques based on mesh partitioning algorithms for distributed memory parallel finite element processing are quite generic and computationally efficient. However, further improvements are needed. In particular, we plan to further investigate the following aspects:
  - Control of convexity and continuity of the clusters (Section 4.2);
  - Improved handling of nodes with prescribed boundary conditions. Our currently implementation prohibits, with a few exceptions, clustering of nodes with boundary conditions. This constraint may greatly limit mesh clustering capabilities and the performance of the method.
- *Optimization of stiffness and mass matrices computation:* In our current implementation, the computational work to compute the global stiffness and mass matrices does not reduce with mesh unrefinement and may even increase in some cases. Possible solutions to this limitation are discussed in Section 4.1.
- *Large non-matching interfaces:* The presented algorithm used to handle non-matching meshes is not effective, in general, for large non-matching interfaces. Section 6.2 presents some possible solutions.

ACKNOWLEDGMENTS

The partial support of this work by Altair Engineering is gratefully acknowledged. We would like to thank the anonymous reviewers for their comments which have helped us improving the paper.

references

[1] M. Ainsworth and J. T. Oden. A posteriori error estimation in finite element analysis. *Computational Mechanics Advances. Special Issue of Computer Methods in Applied Mechanics and Engineering*, 142:1–88, 1997.

[2] T. Arbogast. Non-mortar mixed finite element method for elliptic problems on non-matching multiblock grids. *Computer Methods in Applied Mechanics and Engineering*, 149(1–4):255–265, 1997.

[3] T. Arbogast, L. C. Cowsar, M. F. Wheeler, and I. Yotov. Mixed finite element methods on non-matching multiblock grids. *SIAM Journal on Numerical Analysis*, 37(4):1295–1315, 2000.

[4] S. N. Atluri and T. Zhu. A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics. *Computational Mechanics*, 22:117–127, 1998.

[5] I. Babuska. The finite element method with lagrange multipliers. *Numer. Math.*, 20: 179–192, 1973.

[6] I. Babuška, U. Banerjee, and J. E. Osborn. Survey of meshless and generalized finite element methods: A unified approach. *Acta Numerica*, 12:1–125, May 2003.

[7] I. Babuška, U. Banerjee, and J. E. Osborn. Superconvergence in the generalized finite element method. Technical Report ICES Report 05-45, The University of Texas at Austin, Austin, TX, November 2005.

[8] I. Babuška, G. Caloz, and J. E. Osborn. Special finite element methods for a class of second order elliptic problems with rough coefficients. *SIAM J. Numerical Analysis*, 31 (4):745–981, 1994.

[9] I. Babuška and J. M. Melenk. The partition of unity finite element method. *International Journal for Numerical Methods in Engineering*, 40:727–758, 1997.

[10] I. Babuška and T. Strouboulis. *The Finite Element Method and its Reliability*. Numerical Mathematics and Scientific Computation. Oxford Science Publications, New York, USA, 2001.

[11] C. Bacuta and J. Xu. Partition of unity for the Stokes problem on nonmatching grids. In *Copper Mountain Conference on Multigrid Methods*, 2003.

[12] K. J. Bathe. *Finite Element Procedures in Engineering Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[13] R. Becker and P. Hansbo. A finite element method for domain decomposition with non-matching grids. Technical Report RR-3613, INRIA, 1999.

[14] T. Belytschko, Y. Krongauz, D. Organ, and M. Fleming. Meshless methods: An overview and recent developments. *Computer Methods in Applied Mechanics and Engineering*, 139: 3–47, 1996.

[15] T. Belytschko, Y. Y. Lu, and L. Gu. Crack propagation by element free Galerkin methods. In *Advanced Computational Methods for Material Modeling*, pages 191–205, 1993. AMD-Vol. 180/PVP-Vol. 268, ASME 1993.

[16] T. Belytschko, Y. Y. Lu, and L. Gu. Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering*, 37:229–256, 1994.

[17] T. Belytschko, N. Moes, S. Usui, and C. Parimi. Arbitrary discontinuities in finite elements. *International Journal for Numerical Methods in Engineering*, 50:993–1013, 2001.

[18] T. Belytschko, D. Organ, and Y. Krongauz. A coupled finite element–element-free Galerkin method. *Computational Mechanics*, 17:186–195, 1995.

[19] C. Bernadi, Y. Maday, and A. Patera. A new non-conforming approach to domain decomposition: the mortar element method. In H. Brezis and J. L. Lions, editors, *Nonlinear partial differential equations and their applications*, pages 13–51. Pitman, 1994.

[20] G. F. Carey and J. T. Oden. *Texas Finite Element Series Volume II–A Second Course*. Prentice-Hall, New Jersey, 1983.

[21] E. G. D. Carmo and A. V. C. Duarte. A discontinuous finite element-base domain decomposition method. *Computer Methods in Applied Mechanics and Engineering*, 190: 825–843, 2000.

[22] J. S. Chen, C. Pan, and C. T. Wu. Large deformation analysis of rubber based on a reproducing kernel particle method. *Computational Mechanics*, 19:211–227, 1997.

[23] J. S. Chen, C. Pan, C. T. Wu, and W. K. Liu. Reproducing kernel particle methods for large deformation analysis of nonlinear structures. *Computer Methods in Applied Mechanics and Engineering*, 139:195–227, 1996.

[24] J. S. Chen, C.-T. Wu, S. Yoon, and Y. You. A stabilized conforming nodal integration for Galerkin mesh-free methods. *International Journal for Numerical Methods in Engineering*, 50:435–466, 2001.

[25] Y.-S. Cho and S. Im. MLS-based variable-node elements compatible with quadratic interpolation. Part I: Formulation and application for non-matching meshes. *International Journal for Numerical Methods in Engineering*, 65:494–516, 2006.

[26] Y.-S. Cho, S. Jun, S. Im, and H.-G. Kim. An improved interface element with variable nodes for non-matching finite element meshes. *Computer Methods in Applied Mechanics and Engineering*, 194:3022–3046, 2005.

[27] C. Daux, N. Moës, J. Dolbow, N. Sukumar, and T. Belytschko. Arbitrary branched and intersecting cracks with the extended finite element method. *International Journal for Numerical Methods in Engineering*, 48:1741–1760, 2000.

[28] S. De and K. J. Bathe. The method of finite spheres. *Computational Mechanics*, 25: 329–345, 2000.

[29] A. P. de la Hoz. The fractal behaviour of triangular refined/derefined meshes. *Communications in Numerical Methods in Engineering*, 12(5):295–302, 1996.

[30] C. R. Dohrmann, S. W. Key, and M. W. Heinstein. A method for connecting dissimilar finite element meshes in two dimensions. *International Journal for Numerical Methods in Engineering*, 48:655–678, 2000.

[31] C. R. Dohrmann, S. W. Key, and M. W. Heinstein. Methods for connecting dissimilar three-dimensional finite element meshes. *International Journal for Numerical Methods in Engineering*, 47:1057–1080, 2000.

[32] J. Dolbow, N. Moes, and T. Belytschko. Discontinuous enrichment in finite elements with a partition of unity method. *Finite Elements in Analysis and Design*, 36:235–260, 2000.

[33] C. A. Duarte. *The hp Cloud Method*. PhD dissertation, The University of Texas at Austin, December 1996. Austin, TX, USA.

[34] C. A. Duarte and I. Babuška. Mesh-independent directional p-enrichment using the generalized finite element method. *International Journal for Numerical Methods in Engineering*, 55(12):1477–1492, 2002.

[35] C. A. Duarte, I. Babuška, and J. T. Oden. Generalized finite element methods for three dimensional structural mechanics problems. In S. N. Atluri and P. E. O'Donoghue, editors, *Modeling and Simulation Based Engineering*, volume I, pages 53–58. Tech Science Press, October 1998. Proceedings of the International Conference on Computational Engineering Science, Atlanta, GA, October 5-9, 1998.

[36] C. A. Duarte, I. Babuška, and J. T. Oden. Generalized finite element methods for three dimensional structural mechanics problems. *Computers and Structures*, 77:215–232, 2000.

[37] C. A. Duarte, O. N. Hamzeh, T. J. Liszka, and W. W. Tworzydlo. A generalized finite element method for the simulation of three-dimensional dynamic crack propagation. *Computer Methods in Applied Mechanics and Engineering*, 190:2227–2262, 2001.

[38] C. A. Duarte, T. J. Liszka, and W. W. Tworzydlo. Improving the robustness of the finite element method using a clustered partition of unity. In P. R. M. Lyra, S. M. B. A. da Silva, F. S. Magnani, L. J. do N. Guimaraes, L. M. da Costa, and E. Parente Junior, editors, *XXV Iberian Latin-American Congress on Computational Methods in Engineering*, Recife, PE, Brazil, November 2004. 16 pages. ISBN Proceedings CD: 857 409 869-8.

[39] C. A. Duarte, T. J. Liszka, W. W. Tworzydlo, and T. A. Westermann. The generalized finite element method – improving finite elements through meshless technology. Structural Research Series 639, Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champaign, May 2005. ISSN-0197-9191.

[40] C. A. M. Duarte and J. T. Oden. Hp clouds–a meshless method to solve boundary-value problems. Technical Report 95-05, TICAM, The University of Texas at Austin, May 1995.

[41] C. A. M. Duarte and J. T. Oden. An *hp* adaptive method using clouds. *Computer Methods in Applied Mechanics and Engineering*, 139:237–262, 1996.

[42] C. A. M. Duarte and J. T. Oden. *Hp* clouds—an *hp* meshless method. *Numerical Methods for Partial Differential Equations*, 12:673–705, 1996.

[43] B. Flemisch, M. A. Puso, and B. I. Wohlmuth. A new dual mortar method for curved interfaces: 2d elasticity. *International Journal for Numerical Methods in Engineering*, 63: 813–832, 2005.

[44] S. Ghosh, K. Lee, and S. Moorthy. Multiple scale analysis of heterogeneous elastic structures using homogenization theory and Voronoi cell finite element method. *International Journal of Solids and Structures*, 32(1):27–62, 1995.

[45] S. Ghosh and S. Moorthy. Three dimensional Voronoi cell finite element model for microstructures with ellipsoidal heterogeneities. *Computational Mechanics*, 34:510–531, 2004.

[46] R. A. Gingold and J. J. Monaghan. Kernel estimates as a basis for general particle methods in hydrodynamics. *Journal of Computational Physics*, 46:429–453, 1982.

[47] M. Griebel and M. A. Schweitzer. A particle-partition of unity method for the solution of elliptic, parabolic and hyperbolic pdes. *SIAM Journal Scientific Computing*, 22(3): 853–890, 2000.

[48] M. Griebel and M. A. Schweitzer, editors. *Meshfree Methods for Partial Differential Equations*, volume 26 of *Lecture Notes in Computational Science and Engineering*, 2002. Springer.

[49] M. Griebel and M. A. Schweitzer. A particle-partition of unity method—Part II: Efficient cover construction and reliable integration. *SIAM Journal Scientific Computing*, 23(5): 1655–1682, 2002.

[50] M. Griebel and M. A. Schweitzer. A particle-partition of unity method—Part III: A multilevel solver. *SIAM Journal Scientific Computing*, 24(2):377–409, 2002.

[51] M. Griebel and M. A. Schweitzer, editors. *Meshfree Methods for Partial Differential Equations II*, volume 43 of *Lecture Notes in Computational Science and Engineering*, 2005. Springer.

[52] P. Hansbo and J. Hermansson. Nitsche's method for coupling non-matching meshes in fluid-structure vibration problems. *Computational Mechanics*, 32:134–139, 2003.

[53] N. V. Hattangady. Coarsening of mesh models for representation of rigid objects in finite element analysis. *International Journal for Numerical Methods in Engineering*, 44:313–326, 1999.

[54] Y. Huang and J. Xu. A conforming finite element method for overlapping and nonmatching grids. *Mathematics of Computation*, 72(243):1057–1066, 2003.

[55] G. Karypis. `http://www-users.cs.umn.edu/~karypis/metis/index.html`.

[56] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, 1981.

[57] S. Li and W. K. Liu. *Meshfree Particle Methods*. Springer, November 2004. ISBN: 3540222561.

[58] T. Liszka and J. Orkisz. Finite difference method for arbitrary irregular meshes in nonlinear problems of applied mechanics. In *IV SMiRt*, San Francisco, 1977.

[59] T. Liszka and J. Orkisz. The finite difference method at arbitrary irregular grids and its application in applied mechanics. *Computers and Structures*, 11:83–95, 1980.

[60] G. R. Liu. *Mesh Free Methods: Moving Beyond the Finite Element Method*. CRC Press, July 2002.

[61] W. K. Liu and Y. Chen. Wavelet and multiple scale reproducing kernel methods. *International Journal for Numerical Methods in Fluids*, 21:901–931, 1995.

[62] W. K. Liu, Y. Chen, S. Jun, J. S. Chen, T. Belytschko, C. Pan, R. A. Uras, and C. T. Chang. *Overview and Applications of the Reproducing Kernel Particle Methods*, volume 3 of *Archives in Computational Methods in Engineering: State of Art Reviews*. CIMNE, Bearcelona, Spain, 1996.

[63] W. K. Liu, S. Jun, and Y. F. Zhang. Reproducing kernel particle methods. *International Journal for Numerical Methods in Engineering*, 20:1081–1106, 1995.

[64] J. M. Melenk and I. Babuška. The partition of unity finite element method: Basic theory and applications. *Computer Methods in Applied Mechanics and Engineering*, 139:289–314, 1996.

[65] N. Moes, J. Dolbow, and T. Belytschko. A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering*, 46:131–150, 1999.

[66] B. Nayroles, G. Touzot, and P. Villon. Generalizing the finite element method: Diffuse approximation and diffuse elements. *Computational Mechanics*, 10:307–318, 1992.

[67] J. T. Oden and G. F. Carey. *Texas Finite Element Series Volume IV–Mathematical Aspects*. Prentice-Hall, New Jersey, 1983.

[68] J. T. Oden, C. A. Duarte, and O. C. Zienkiewicz. A new cloud-based *hp* finite element method. Technical Report 96-55, TICAM, The University of Texas at Austin, Austin, Texas, USA, December 1996.

[69] J. T. Oden, C. A. Duarte, and O. C. Zienkiewicz. A new cloud-based *hp* finite element method. *Computer Methods in Applied Mechanics and Engineering*, 153:117–126, 1998.

[70] J. T. Oden and J. N. Reddy. *An Introduction to the Mathematical Theory of Finite Elements*. John Wiley and Sons, New York, 1976.

[71] E. Onate and S. Idelsohn. A mesh-free finite point method for advective-diffusive transport and fluid flow problems. *Computational Mechanics*, 21:283–292, 1998.

[72] E. Onate, S. Idelsohn, and O. C. Zienkiewicz. Finite point methods in computational mechanics. Technical report, International Center for Numerical Methods in Engineering, Barcelona, Spain, July 1995.

[73] E. Onate, S. Idelsohn, O. C. Zienkiewicz, R. L. Taylor, and C. Sacco. A stabilized finite point method for analysis of fluid mechanics problems. *Computer Methods in Applied Mechanics and Engineering*, pages 315–346, 1996.

[74] C. Park, K, C. A. Felippa, and G. Rebel. A simple algorithm for localized construction of non-matching structural interfaces. *International Journal for Numerical Methods in Engineering*, 53:2117–2142, 2002.

[75] PhlexSolids. PHLEXSolids, 2001. Altair Engineering Inc., http://www.altair.com.

[76] A. Plaza, M. A. Padron, and G. F. Carey. A 3D derefinement algorithm for tetrahedral grids. In *Trends in Unstructured Mesh Generation*, AMD-Vol. 220, pages 17–23. ASME, 1997.

[77] H. Sagan. *Space Filing Curves*. Springer-Verlag, Berlin, Heidelberg, New York, 1994.

[78] T. Strouboulis, I. Babuška, and K. Copps. The design and analysis of the generalized finite element mehtod. *Computer Methods in Applied Mechanics and Engineering*, 81 (1–3):43–69, 2000.

[79] T. Strouboulis, K. Copps, and I. Babuška. The generalized finite element method: An example of its implementation and illustration of its performance. *International Journal for Numerical Methods in Engineering*, 47(8):1401–1417, 2000.

[80] N. Sukumar, D. Chopp, N. Moes, and T. Belytschko. Modeling holes and inclusions by level sets in the extended finite element method. *Computer Methods in Applied Mechanics and Engineering*, 190:6183–6200, 2001.

[81] N. Sukumar, N. Moes, B. Moran, and T. Belytschko. Extended finite element method for three-dimensional crack modelling. *International Journal for Numerical Methods in Engineering*, 48(11):1549–1570, 2000.

[82] N. Sukumar, B. Moran, and T. Belytschko. The natural element method in solid mechanics. *International Journal for Numerical Methods in Engineering*, 43(5):839–887, 1998.

[83] D. Sulsky, Z. Chen, and H. L. Schreyer. A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118:179–196, 1994.

[84] J. W. Swegle, S. W. Attaway, M. W. Heinstein, F. J. Mello, and D. L. Hicks. An analysis of the smoothed particle hydrodynamics. Technical Report SAND93-2513 UC-705, Sandia, 1994.

[85] J. W. Swegle, D. L. Hicks, and S. W. Attaway. Smoothed particle hydrodynamics stability analysis. *Journal of Computational Physics*, 116:123–134, 1995.

[86] B. Szabo and I. Babuška. *Finite Element Analysis*. John Wiley and Sons, New York, 1991.

[87] R. Tian and G. Yagawa. Generalized nodes and high-performance elements. *International Journal for Numerical Methods in Engineering*, 64:2039–2071, 2005.

[88] W. Tworzydlo. Analysis of large deformations of membrane shells by the generalized finite difference method. *Computers and Structures*, 27(1):39–59, 1987.

[89] W. Tworzydlo. The FDM in arbitrary curvilinear co-ordinates - formulation, numerical approach and applications. *International Journal for Numerical Methods in Engineering*, 28:261–277, 1989.

[90] B. I. Wohlmuth. A comparison of dual lagrange multiplier spaces for mortar finite element discretizations. *Mathematical Modeling and Numerical Analysis*, 36(6):995–1012, 2002.

[91] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method,* 4th *Edition*, volume I. McGraw-Hill, 1981.

[92] O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu. *The Finite Element Method: Its Basis and Fundamentals*. Elsevier Butterworth-Heinemann, 2005.