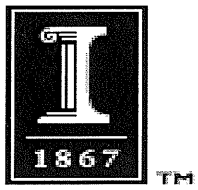
CIVIL ENGINEERING STUDIES
Structural Research Series No. 638

UILU-ENG-2005-2004



ISSN-0197-9191

A Technique to Combine Meshfree-and Finite Element-Based Partition of Unity Approximations

by

C.A. Duarte, D.Q. Migliano, E.B. Baker

Department of Civil and Environmental Engineering
University of Illinois at Urbana-Champaign

May 2005

A Technique to Combine Meshfree- and Finite Element-Based Partition of Unity Approximations

C. A. Duarte[a,*], D. Q. Migliano[b] and E. B. Becker[b]

[a] Department of Civil and Environmental Eng.

University of Illinois at Urbana-Champaign

Newmark Laboratory, 205 North Mathews Avenue

Urbana, Illinois 61801, USA

*Corresponding author: caduarte@uiuc.edu

[b] ICES - Institute for Computational Engineering and Science

The University of Texas at Austin, Austin, TX, 78712, USA

Abstract

A technique to couple finite element discretizations with any partition of unity based approximation is presented. Emphasis is given to the combination of finite element and meshfree shape functions like those from the hp cloud method. H and p type approximations of any polynomial degree can be built. The procedure is essentially the same in any dimension and can be used with any Lagrangian finite element discretization. Another contribution of this paper is a procedure to built generalized finite element shape functions with any degree of regularity using the so-called R-functions. The technique can also be used in any dimension and for any type of element. Numerical experiments demonstrating the coupling technique and the use of the proposed generalized finite element shape functions are presented.

Keywords: Meshfree methods; Generalized finite element method; Partition of unity method; Hp-cloud method; Adaptivity; P-method; P-enrichment;

1 Introduction

One of the major difficulties encountered in the finite element analysis of tires, elastomeric bearings, seals, gaskets, vibration isolators and a variety of other of products made of rubbery materials, is the excessive element distortion. Distortion of elements is inherent to Lagrangian formulations used to analyze this class of problems. Rubber components often have geometric features

that give rise to very steep or even singular gradients. Elements in the neighborhood of these features inevitably become highly distorted, often to the point of material eversion (negative determinant of deformation gradient). This event effectively terminates the solution process and renders the model useless for further simulation. For finite element models with coarse elements, it is often possible to achieve the desired degree of loading before element failure occurs. These model, however, will not provide accurate solutions. When fine meshes are used this "element collapse" can occur under very small applied loads. Too much is demanded of an element adjacent to, for example, a singular point. The analysis must then trade load level for accuracy, since accurate solutions for the necessary load levels can not be obtained.

Meshfree methods such as the hp-cloud method [15, 16], the method of finite spheres [9], reproducing kernel particle methods [26–28], the element-free Galerkin method [3–6, 22, 37], the finite point method [34–36], the generalized finite difference method [24, 25, 45, 46], the diffuse element method [32], the modified local Petrov-Galerkin method [1], smooth particle hydrodynamics [19, 43, 44], among others, offer an attractive alternative for the solution of many classes of problems that are difficult or even not feasible to solve using the finite element method. In particular, meshfree methods have shown to be very effective for the solution of problems involving large deformations like those described above [7, 8]. Excellent overviews of meshfree methods and their applications can be found in, for example, [3, 27].

In meshfree methods, the approximation of field variables is constructed in terms of nodes without the aid of a mesh. The actual implementation of some meshfree methods, however, requires the partition of the domain through the use of a "background grid" for domain integration. Nevertheless, due to the flexibility in constructing conforming shape functions to meet specific needs for different applications, it has been reported [7, 8, 12, 13, 15, 16, 30] that meshfree methods are particularly suitable for the simulation of crack propagation, hp-adaptivity, and modeling of large deformation problems. The use of smooth shape functions appears to be particularly effective in dealing with large deformation problems.

One of the main drawbacks of meshfree methods has been the fact that the computational cost is too high in some applications due to the fact that one has to use a great number of integration points in order to integrate the meshfree functions and their products over a computational domain. One approach to ameliorate the computational cost is to use these methods only in parts of the domain where they are strictly needed and use a finite element discretization elsewhere. Besides reducing the overall computational cost, this approach has several other appealing features. It facilitates, for example, the implementation of Dirichlet boundary conditions [22] and the coupling with classical structural finite elements, like rods, shell, rigid bars, etc.

Several techniques to couple meshfree and finite element methods have been proposed. Belytschko et al. [6] proposed a coupling technique in which some finite element nodes are replaced by meshfree nodes and a ramp function is used to build the transition between the finite element and meshfree discretizations. Linear consistency is attained with this approach. A generalization of this idea was proposed by Hegen [20] based on the use of Lagrange multipliers.

Huerta and Fernandez-Mendez [21] proposed a technique to couple finite element and

meshfree discretizations based on consistency or reproducibility conditions of the resulting shape functions and moving least squares techniques. The support size of the meshfree functions and their distribution must obey some rules in order to be admissible [21]. A related technique was proposed by Liu et al. [29] with the goal of improving a finite element discretization with meshfree shape functions.

Many of the meshfree shape functions like moving least squares and Shepard functions, constitute a partition of unity. In other words, these functions add to the unity at any point in the domain. Lagrangian finite element shape functions also possess this property. In this paper, we present a technique to couple meshfree and finite element discretizations that explores the partition of unity property of these functions. The procedure, while simple, is quite flexible and generic. The only requirement on the finite element and meshfree shape functions is that the union of their supports completely covers the computational domain. H and p type approximations can be built in the finite element and meshfree parts of the domain. Exponential convergence of the resulting coupled approximation is demonstrated though a numerical example.

Another approach to reduce the cost of numerical integration of meshfree shape functions is to use a finite element mesh to build them and enforce that the support of all functions coincide with the support of corresponding global Lagrangian shape functions defined on the same mesh. In this case, the method is no longer strictly meshfree but some of the attractive features of meshfree approximations, like high regularity of the approximation, can still be retained. Edwards [18] has proposed such approach and have shown that it can be used in any dimension and for any kind of finite element (triangular, quadrilateral, tetrahedral, hexahedral, etc) while rendering C^∞ finite element shape functions. Edwards' approach however, has a serious practical limitation– It requires that the support of the functions be convex. It is not possible, in general, to build finite element meshes such that the support of the corresponding finite element shape functions be convex. In this paper, we generalize Edwards' approach to handle non-convex supports with the aid of the so-called R-functions [38, 39].

In the following sections the formulation of the proposed coupling technique and the generalization of Edwards' approach [18] to build finite element shape functions with arbitrary regularity on non-convex supports is introduced. Illustrative numerical experiments are also presented.

2 Partition of Unity Shape Functions

In this section, the construction of partition of unity shape functions is briefly reviewed. Examples of this kind of shape functions are hp-cloud [15,16] and generalized finite element shape functions [11, 12, 41, 42].

Let the functions $\varphi_\alpha,\ \alpha = 1, \ldots, N$, denote a partition of unity (PoU) subordinate to the open covering $\mathcal{T}_N = \{\omega_\alpha\}_{\alpha=1}^N$ of a domain $\Omega \subset \mathbb{R}^n,\ n = 1, 2, 3$. Here, ω_α is the support of the partition of unity function φ_α and N the number of functions. We call ω_α a *cloud* and associate with each one of them a node, denoted by $\boldsymbol{x}_\alpha$.

From the above we have that

$$\varphi_\alpha \in C_0^s(\omega_\alpha),\ s \geq 0, \qquad 1 \leq \alpha \leq N$$
$$\sum_\alpha \varphi_\alpha(\boldsymbol{x}) = 1 \qquad\qquad \forall\, \boldsymbol{x} \in \Omega$$

Let $\chi_\alpha(\omega_\alpha) = \text{span}\{L_{i\alpha}\}_{i\in\mathcal{I}(\alpha)}$ denote local spaces defined on $\omega_\alpha,\ \alpha = 1, \ldots, N$, where $\mathcal{I}(\alpha), \alpha = 1, \ldots, N$, are index sets and $\{L_{i\alpha}\}_{i\in\mathcal{I}(\alpha)}$ a basis for the space $\chi_\alpha(\omega_\alpha)$. Functions $L_{i\alpha}$ are also denoted by enrichment or local approximation functions.

The partition of unity shape functions associated with a node $\boldsymbol{x}_\alpha$ are then defined by

$$\phi_i^\alpha := \varphi_\alpha L_{i\alpha},\ i \in \mathcal{I}(\alpha) \qquad \text{(no sum on } \alpha) \tag{1}$$

Different choices for the partition of unity functions are possible. Each one of then will lead to a different class of shape functions. Some of the possible choices are discussed below.

2.1 Shepard Partition of Unity

Shepard's formula [23, 40] is a very simple approach to build partition of unity functions and it is often used in meshfree methods [9, 15, 16]. Let $\mathcal{W}_\alpha : \mathbb{R}^n \to \mathbb{R}$ denote a *weighting function* with compact support ω_α and that belongs to the space $C_0^s(\omega_\alpha),\ s \geq 0$. Suppose that such weighting function is built at every cloud $\omega_\alpha,\ \alpha = 1, \ldots, N$. Then, the partition of unity functions φ_α associated with the clouds ω_α, $\alpha = 1, \ldots, N$, are defined by

$$\varphi_\alpha(\boldsymbol{x}) = \frac{\mathcal{W}_\alpha(\boldsymbol{x})}{\sum_{\beta(\boldsymbol{x})} \mathcal{W}_\beta(\boldsymbol{x})} \qquad \beta(\boldsymbol{x}) \in \{\gamma \mid \mathcal{W}_\gamma(\boldsymbol{x}) \neq 0\} \qquad \alpha = 1, \ldots, N \tag{2}$$

which are known as Shepard functions [23, 40].

The choice of the weighting functions $\mathcal{W}_\alpha$ is quite arbitrary. If, for example, the clouds are spheres with radius h_α and centered at $\boldsymbol{x}_\alpha$, i.e.,

$$\omega_\alpha = \{\boldsymbol{x} \in \mathbb{R}^n : \|\boldsymbol{x}_\alpha - \boldsymbol{x}\|_{\mathbb{R}^n} < h_\alpha\}$$

the weighting functions can be built through the following composition

$$\mathcal{W}_\alpha(\boldsymbol{x}) := g(r_\alpha) \tag{3}$$

where $g : \mathbb{R} \to \mathbb{R}$ is, e.g., a B-spline with compact support $[-1, 1]$ and r_α is the functional

$$r_\alpha := \frac{\|\boldsymbol{x} - \boldsymbol{x}_\alpha\|_{\mathbb{R}^n}}{h_\alpha}$$

Here, h_α is the radius of the support ω_α of the radial weighting function $\mathcal{W}_\alpha$.

In this case, the Shepard partition of unity is said to be meshfree since they do not require a finite element mesh for their definition. Note also that the regularity of the partition of unity depends only on the regularity of the weighting functions. Therefore, Shepard partition of unity functions with arbitrary regularity can easily be built.

2.2 Finite Element Partition of Unity

Lagrangian finite element shape functions constitute a partition of unity. In this case, the cloud ω_α is simply the union of the finite elements sharing a vertex node $\boldsymbol{x}_\alpha$ in the mesh. Each node is associated with its own cloud comprised by the elements surrounding that node. Figure 1 show examples of such clouds. The cloud of node 1 includes elements *c*,*d*,*i*,*h* and *g* and is a *convex* cloud, cloud 2 comprises elements *a*, *b*, *e*, *d* and *c* and is a *non-convex* cloud.

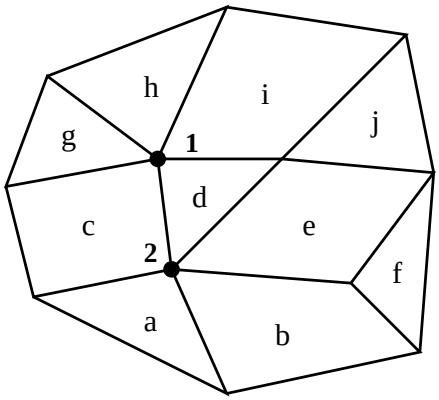


Figure 1: Examples of finite element clouds.

The partition of unity function φ_α is equal to the usual global finite element shape function. Finite element shape functions are inexpensive to compute and to numerically integrate since they are (mapped) polynomial functions while Shepard functions are, in general, rational polynomials. However, they are, for most practical matters, limited to C^0 regularity in two or higher dimensional spaces.

2.3 A C^∞ Finite Element Based PoU for Convex Clouds

A technique to build C^∞ partition of unity shape functions over convex finite element clouds was proposed by Edwards [18]. The resulting shape functions can be seem as C^∞ finite element shape functions and used in any standard finite element implementation. One important practical limitation of the technique, however, is that it is limited to convex finite element clouds. As discussed in the previous section, a finite element cloud (the elements sharing a finite element vertex node)

can be non-convex. In this section, we review Edwards' approach to built C^∞ finite element based PoU for convex clouds. In Section 2.4, we present an extension of Edwards' approach that can handle the case of non-convex finite element clouds while rendering partition of unity functions with arbitrary smoothness.

2.3.1 C^∞ Finite Element Based Weighting Function for Convex Clouds

In this section, a technique to build C^∞ weighting functions over convex finite element clouds is discussed [18]. These weighting functions are said to be finite element based since they have the same support (cloud) as the classical global finite element shape functions. Therefore, the intersection of the support of these weighting functions coincide with the finite elements of the mesh. As a consequence, the numerical integration of these functions and their products can be efficiently done using the finite element mesh.

A C^∞ finite element based weighting function with a convex support can be built from the product of the so-called *cloud boundary functions*. Lets consider first the case of a cloud associated with a node not at the boundary of the domain. This cloud is denoted by *interior cloud*. One example is shown in Figure 2. The boundary of a cloud in two dimensions is the polygonal built from the edges of the elements in the cloud that are not connected to its node. This is indicated as side j, $j = 1, \ldots, 7$, in the example of Figure 2.

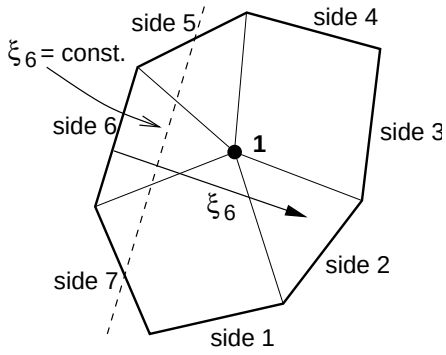


Figure 2: Setup for the construction of cloud boundary functions.

Associated with each side j at the boundary of a cloud, there is a parametric coordinate ξ_j measured in the direction perpendicular to the edge and set to zero at the edge (Cf. Figure 2). A function that vanishes smoothly as the edge is approached and that is greater than zero for points in the cloud is called a *cloud boundary function*. It can be defined, for example, as

$$\mathcal{E}_{\alpha,j}[\boldsymbol{x}(\xi_j)] = \widehat{\mathcal{E}}_{\alpha,j}(\xi_j) := \begin{cases} e^{-\xi_j^{-\gamma}} & , \quad 0 < \xi_j \\ 0 & , \quad \text{otherwise} \end{cases} \tag{4}$$

where γ is a positive constant.

A cloud boundary function and all of its derivatives are zero on the corresponding edge and on the "negative" side of the edge. Figure 3 illustrates a cloud boundary function.

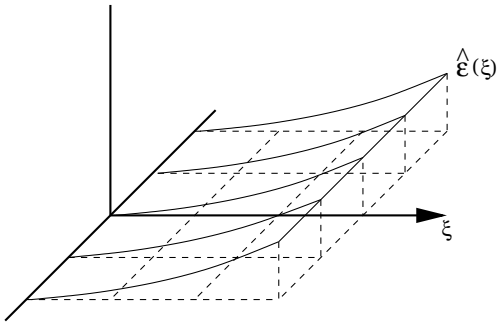


Figure 3: Example of a two-dimensional cloud boundary function.

The cloud boundary function defined above can be used to built a C^∞ weighting function that is zero at the boundary of the cloud and greater than zero inside the cloud as follows

$$\mathcal{W}_\alpha(\boldsymbol{x}) := \prod_{j=1}^{M_\alpha} \mathcal{E}_{\alpha,j}(\boldsymbol{x}) \tag{5}$$

where M_α is the number of cloud boundary functions for the cloud α.

Further consideration of the weighting functions defined above shows that it is applicable only to convex clouds. If the extension of any edge intersects the cloud, i.e., if the cloud is non-convex, the weighting function will also vanish in the interior of the cloud which is not desirable. This issue is dealt with in Section 2.4.

For clouds with nodes located at the boundary of the domain, the procedure is basically the same as above. The cloud is still the union of the elements sharing the node. However, the node will not be completely surrounded by elements. Consider, for example, cloud α in Figure 4. The weighting function for cloud α is given by

$$\mathcal{W}_\alpha(\boldsymbol{x}) = \prod_{j=1}^{3} \mathcal{E}_{\alpha,j}(\boldsymbol{x})$$

Therefore, we use cloud boundary functions only for the edges inside of the domain.

Having the finite element based weighting functions, we can now use Shepard's formula (2) to build a partition of unity subordinate to the finite element clouds. The resulting partition of unity functions have the same regularity as the weighing functions. In addition, the numerical integration of these functions and their products can be efficiently done using the underlying finite element mesh since their support coincide with the finite elements. This partition of unity can then be used to build shape functions of any polynomial degree using the technique described in Section 2. The resulting shape functions are also C^∞ functions if the enrichment functions, $L_{i\alpha}$, have this property. The high regularity of these shape functions can be advantageous to solve, for

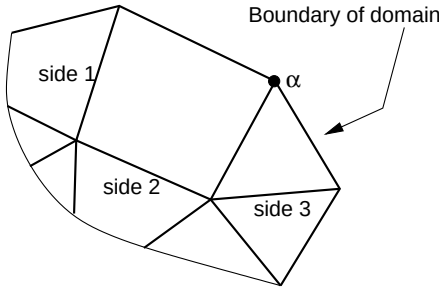


Figure 4: Example of a two-dimensional cloud associated with a node at the boundary of the domain.

example, plate and shell problems that require C^1 continuity of the shape functions. Least square finite element methods can also benefit from the high regularity of these functions.

2.4 C^k Finite Element Based PoU for Non-Convex Clouds

Consider the finite element cloud α depicted in Figure 5 and comprising four elements. The re-entrant corner at the intersection of sides 5 and 6 makes the cloud non-convex and the procedure outlined in the previous section can not be used because the cloud boundary functions $\mathcal{E}_{\alpha,5}(\boldsymbol{x})$ and $\mathcal{E}_{\alpha,6}(\boldsymbol{x})$ vanish at points inside the cloud. Consequently, the weighting function $\mathcal{W}_\alpha(\boldsymbol{x})$ for cloud α will also be zero in the interior of the cloud if they are built using (5). We seek a modification to the approach of Section 2.3.1 that can handle non-convex finite element clouds.

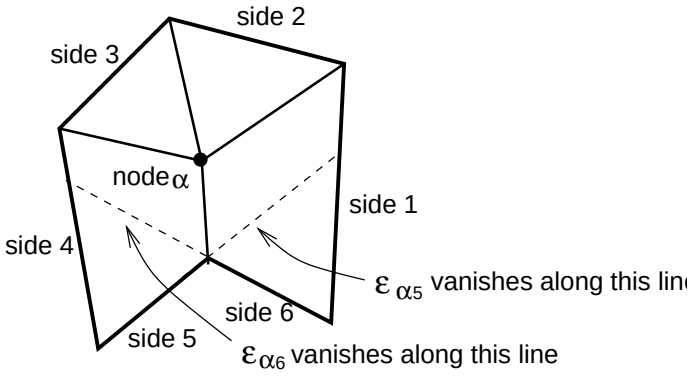


Figure 5: Non-convex finite element cloud.

The proposed procedure is similar to the original one—a cloud boundary function $\mathcal{E}_{\alpha,j}(\boldsymbol{x})$ is constructed for each side of a finite element cloud and the weighting function $\mathcal{W}_\alpha(\boldsymbol{x})$ is again defined as the product of these functions.

Consider again the example of Figure 5. Cloud boundary functions $\mathcal{E}_{\alpha,j}(\boldsymbol{x})$, $j = 1, \ldots, 4$, are built using (4). Next, we combine functions $\mathcal{E}_{\alpha,5}(\boldsymbol{x})$ and $\mathcal{E}_{\alpha,6}(\boldsymbol{x})$ associated with a re-entrant

corner into a single cloud boundary function using the notion of R-functions [38, 39]. An R-function is a real-valued function whose sign is completely determined by the signs of its arguments. As an example, the R-function $f(x, y, z) = xyz$ can be negative only if the number of its negative arguments is odd. Such functions "encode" Boolean logic functions and are called R-functions.

Consider now the R-function $(f_1 \vee_0^k f_2)$ with two arguments, f_1 and f_2, defined by

$$(f_1 \vee_0^k f_2) := \left(f_1 + f_2 + \sqrt{f_1^2 + f_2^2}\right)\left(f_1^2 + f_2^2\right)^{\frac{k}{2}} \tag{6}$$

where k is a positive integer. This function is analytic everywhere except at the origin $(f_1 = f_2 = 0)$, where it is at least k times differentiable, i.e., it belongs to $C^k(\Omega)$ [39].

It $f_1 \geq 0$ and $f_2 \geq 0$ define two regions in $\mathbb{R}^n$ then

- $(f_1 \vee_0^k f_2) \geq 0$ and,
- $(f_1 \vee_0^k f_2) > 0$ if $f_1 > 0$ or $f_2 > 0$.

Note that R-functions can be defined in any dimension and the arguments, f_1 and f_2, can also define regions with curved boundaries.

Suppose now that sides m and n are identified as non-convex sides for a finite element cloud α (e.g., sides 5 and 6 for the cloud of Figure 5). A new cloud boundary function combining $\mathcal{E}_{\alpha,m}$ and $\mathcal{E}_{\alpha,n}$ is then defined as

$$\mathcal{E}_{\alpha,mn}^{nc}(\boldsymbol{x}) := \mathcal{E}_{\alpha,m}(\boldsymbol{x}) \vee_0^k \mathcal{E}_{\alpha,n}(\boldsymbol{x}) \tag{7}$$

where the parameter k is chosen according to the degree of smoothness desired. This cloud boundary function and all other boundary functions for the cloud α are then used to build the node weighting function $\mathcal{W}_\alpha(\boldsymbol{x})$ using (5). The procedure to build cloud boundary functions like, $\mathcal{E}_{\alpha,mn}^{nc}(\boldsymbol{x})$, must be used for all re-entrant corners of a finite element cloud.

Shepard's formula (2) is again used build a partition of unity using the weighting functions defined above and generalized finite element shape functions are built using (1). The resulting shape functions are at least k-times continuously differentiable. In fact, they are C^∞ functions except at the re-entrant corners of the clouds where they are C^k, with k arbitrarily large.

3 Construction of a Meshfree-Finite Element Partition of Unity

A technique to combine finite element approximations with any other partition of unity based approximation is presented in this section. Emphasis is given to the combination of finite element and meshfree shape functions. The basic idea is to treat finite element shape functions of any kind

as a weighting function and use Shepard's formula (2) to build a partition of unity. We call the resulting PoU a meshfree-finite element partition of unity. Details of the formulation are presented next.

3.1 Finite Element and Meshfree Weighting Functions

Let Ω be an open domain in $\mathbb{R}^n,\ n = 1, 2, 3$, covered by a finite element mesh consisting of any type of linear Lagrangian element. Let $\boldsymbol{x}_\alpha$ denote a finite element vertex node in the mesh. Associated with each node $\boldsymbol{x}_\alpha$ there is a linear finite element shape function $N_\alpha(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ with support $\omega_\alpha = \{\boldsymbol{x} \in \Omega : N_\alpha(\boldsymbol{x}) \neq 0\}$. We also refer to the function N_α as a *finite element weighting function*.

Suppose that some of the finite element nodes and associated shape functions are removed from the discretization. Let $\mathcal{I}_{fe}$ denote the index set of all remaining finite element nodes and M_{fe} the dimension of this set, i.e., $M_{fe} = \text{card}\{\mathcal{I}_{fe}\}$. In addition, suppose that M_{mf} meshfree nodes $\boldsymbol{y}_\beta,\ \beta = 1, \ldots, M_{mf}$, are arbitrarily added to Ω. Let $\mathcal{I}_{mf}$ denote the index set of all meshfree nodes. Associated with each meshfree node $\boldsymbol{y}_\beta$ there is a so-called *meshfree weighting function* $\mathcal{W}_\beta(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ with support $\omega_\beta = \{\boldsymbol{x} \in \Omega : \mathcal{W}_\beta(\boldsymbol{x}) \neq 0\}$. These weighting functions are said to be meshfree if they do not require a mesh for their definition. An example is the radial weighting functions given in Section 2.1.

Hypothesis 1 *The supports $\{\omega_\beta\}_{\beta\in\mathcal{I}_{mf}}$ and $\{\omega_\alpha\}_{\alpha\in\mathcal{I}_{fe}}$, of the meshfree and finite element weighting functions are such that*

$$\mathcal{T}_{mf,fe} := \left\{\{\omega_\beta\}_{\beta\in\mathcal{I}_{mf}} \cup \{\omega_\alpha\}_{\alpha\in\mathcal{I}_{fe}}\right\}$$

constitutes an open covering for Ω. i.e.,

$$\bar{\Omega} \subset \mathcal{T}_{mf,fe}$$

Remark 1 *The deletion of finite element nodes and addition of meshfree nodes is completely arbitrary. It is valid, for example, not to delete any finite element node. Also, the number and/or location of added meshfree nodes do not have to coincide with the number and/or location of deleted finite element nodes. Figure 6 shows one example of a meshfree and finite element nodal distribution in a two-dimensional domain. Typically, meshfree weighting functions are used when a finite element discretization is not appropriate or robust.*

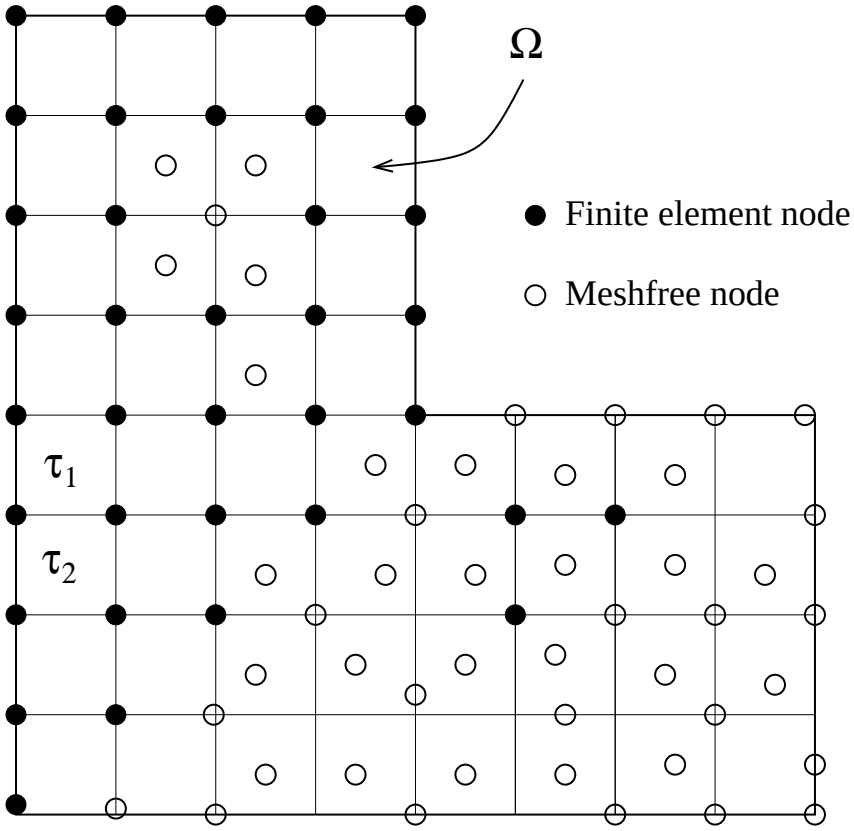


Figure 6: Meshfree and finite element nodes in a domain Ω.

3.2 A Meshfree-Finite Element Partition of Unity

For each finite element node $\boldsymbol{x}_\alpha,\ \alpha \in \mathcal{I}_{fe}$, and each meshfree node $\boldsymbol{y}_\alpha,\ \alpha \in \mathcal{I}_{mf}$, we define a partition of unity function φ_α using Shepard's formula (2)

$$\varphi_\alpha(\boldsymbol{x}) := \begin{cases} \dfrac{N_\alpha(\boldsymbol{x})}{\sum_{\beta\in\mathcal{I}_{fe}(\boldsymbol{x})} N_\beta(\boldsymbol{x}) + \sum_{\gamma\in\mathcal{I}_{mf}(\boldsymbol{x})} \mathcal{W}_\gamma(\boldsymbol{x})} & \text{if } \alpha \in \mathcal{I}_{fe} \\ \dfrac{\mathcal{W}_\alpha(\boldsymbol{x})}{\sum_{\beta\in\mathcal{I}_{fe}(\boldsymbol{x})} N_\beta(\boldsymbol{x}) + \sum_{\gamma\in\mathcal{I}_{mf}(\boldsymbol{x})} \mathcal{W}_\gamma(\boldsymbol{x})} & \text{if } \alpha \in \mathcal{I}_{mf} \end{cases} \tag{8}$$

where

$$\mathcal{I}_{fe}(\boldsymbol{x}) = \{\beta \in \mathcal{I}_{fe} : N_\beta(\boldsymbol{x}) \neq 0\}$$
$$\mathcal{I}_{mf}(\boldsymbol{x}) = \{\beta \in \mathcal{I}_{mf} : \mathcal{W}_\beta(\boldsymbol{x}) \neq 0\}$$

Let $\mathcal{I}_{mf,fe} = \mathcal{I}_{fe} \cup \mathcal{I}_{mf}$ denote the index set of all nodes in the domain Ω. Then, it is straightforward to show that the set

$$\{\varphi_\alpha\}_{\alpha\in\mathcal{I}_{mf,fe}}$$

constitutes a partition of unity subordinate to the open covering $\mathcal{T}_{mf,fe}$, i.e.,

$$\sum_{\alpha\in\mathcal{I}_{mf,fe}} \varphi_\alpha(\boldsymbol{x}) = 1 \qquad \forall\, \boldsymbol{x} \in \Omega$$

The denominator in (8) is equal to the sum of weighting functions (meshfree and finite element) that are non zero at $\boldsymbol{x}$. It scales the numerator such that the resulting functions, φ_α, constitutes a partition of unity. Consider now elements τ_1 or τ_2 indicated in Figure 6. Suppose that all meshfree weighting functions are zero inside those elements. Then, there is no need to use (8) to build the partition of unity since the finite element shape functions of the elements already constitutes a partition of unity. Therefore, elements like τ_1 or τ_2 are standard Lagrangian finite elements. Equation (8) is also not needed if the meshfree weighting functions already constitutes a partition of unity and no finite element shape functions are used in this part of the domain. Examples of meshfree functions that form a partition of unity are the moving least square functions [23] which are used in several meshfree methods.

The procedure above makes the transition between a finite element PoU and any other type of PoU quite natural and transparent. It can be used in any dimension and for any type of Lagrangian finite element shape functions.

Figure 7 shows an example of meshfree and finite element weighting functions in a one-dimensional domain. The finite element weights are just the standard linear hat functions and the meshfree weights were built from cubic B-splines using the composition defined in (3). Figure 8 shows the resulting partition of unity functions built using the weighting functions of Figure 7 and Equation (8).

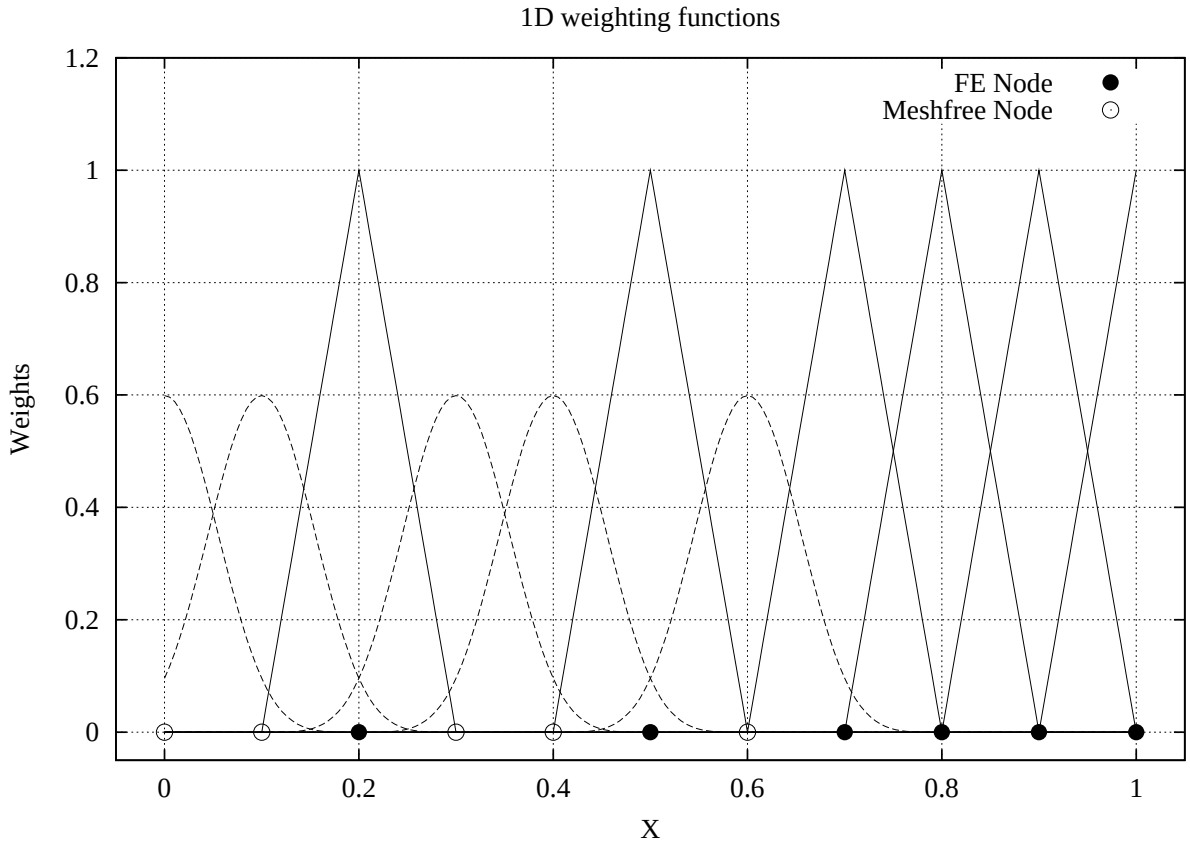


Figure 7: Meshfree and finite element weighting functions in a one-dimensional domain.

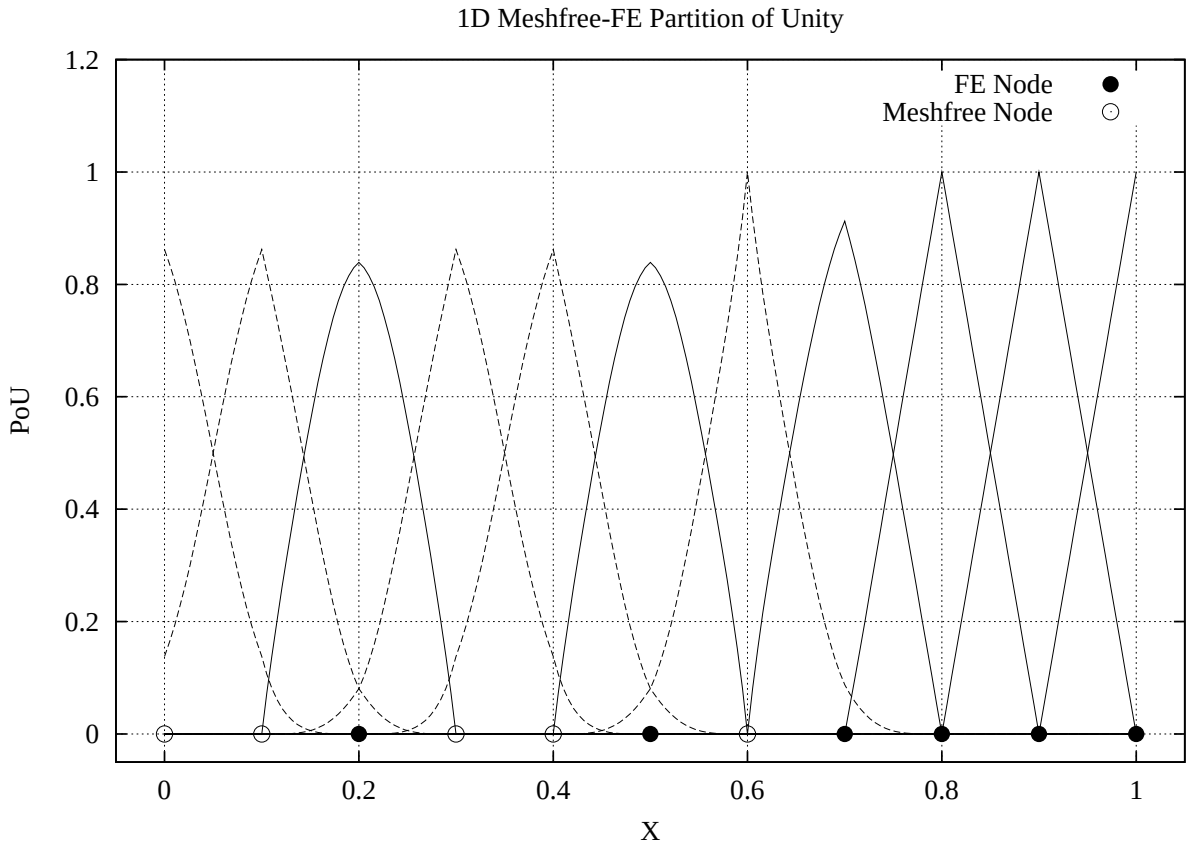


Figure 8: Meshfree-finite element partition of unity built using the weighting functions of Figure 7 and Equation (8).

4 Hp-Cloud-Generalized Finite Element Shape Functions

The meshfree-finite element partition of unity defined in the previous section can be used to build partition of unity shape functions as described in Section 2. The construction of the shape functions follows the same procedure used in partition of unity methods like the hp cloud [14–17, 33] or generalized finite methods [2, 10, 12, 13, 31]. The only difference is in the definition of the partition of unity. Here, we use the meshfree-finite element partition of unity defined in the previous section.

Let $\chi_\alpha(\omega_\alpha) = \text{span}\{L_{i\alpha}\}_{i\in\mathcal{I}(\alpha)}$ denote local spaces defined on $\omega_\alpha,\ \alpha \in \mathcal{I}_{mf,fe}$, where $\mathcal{I}(\alpha),\ \alpha \in \mathcal{I}_{mf,fe}$, are index sets and $L_{i\alpha}$ denotes local approximation or enrichment functions defined over the cloud ω_α.

The hp-cloud-generalized finite element shape functions associated with a vertex node $\boldsymbol{x}_\alpha$ are defined by

$$\phi_i^\alpha = \varphi_\alpha L_{i\alpha},\ i \in \mathcal{I}(\alpha) \qquad \text{(no sum on } \alpha) \tag{9}$$

where φ_α is the partition of unity function defined in (8).

The approximation properties of the functions defined in (9) follows directly from the theory of partition of unity methods presented in [16, 17, 30, 31].

The resulting shape functions built using (9) are like those in the hp-cloud method over parts of the domain covered with meshfree weighting functions and like generalized finite element shape functions in regions covered only by finite element shape functions. The transition between the meshfree and finite element approximations is handled quite naturally using the partition of unity defined in (8). The procedure is essentially the same in any dimension, only the construction of the weighting functions change.

5 Numerical Experiments

5.1 P convergence using a meshfree-finite element PoU

As a first experiment, we compute L^2 projections of the function $u = \sin(4\pi x)$ on spaces spanned by meshfree-finite element shape functions defined by (9). The meshfree-finite element partition of unity is shown on Figure 8 and the enrichment functions, $L_{i\alpha}$, are monomials of degree less or equal to p for all nodes $\boldsymbol{x}_\alpha,\ \alpha = 1, \ldots, 11$. The polynomial degree is uniformly increased from $p = 0$ to $p = 6$. Figure 9 shows the error measured in the L^2 norm of the computed projections versus the number of degrees of freedom, N. It can be observed that the rate of convergence increases with p. This behavior is typical of spectral methods and indicates that the hp-cloud-GFE shape functions defined in Section 4 are able to deliver exponential convergence.

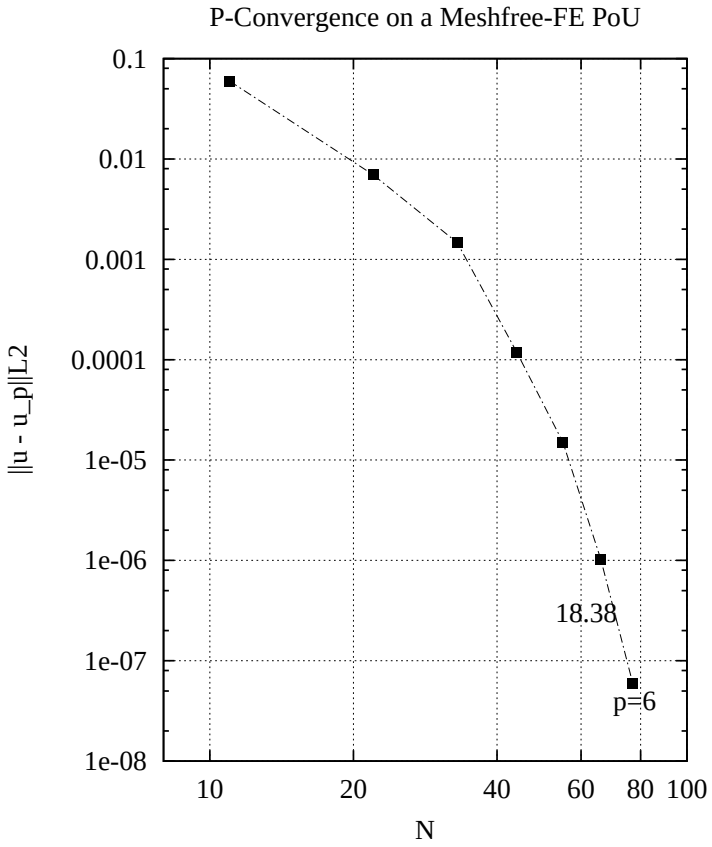


Figure 9: P-convergence of hp-cloud-generalized finite element approximation of function $\sin(4\pi x)$. The partition of unity shown in Figure 8 was used to build the shape functions.

5.2 Two-Dimensional Bracket

The bracket model illustrated in Figure 10 is analyzed in this section. It is fixed at the left end and pressure is applied as indicated. The material properties are also indicated in Figure 10.

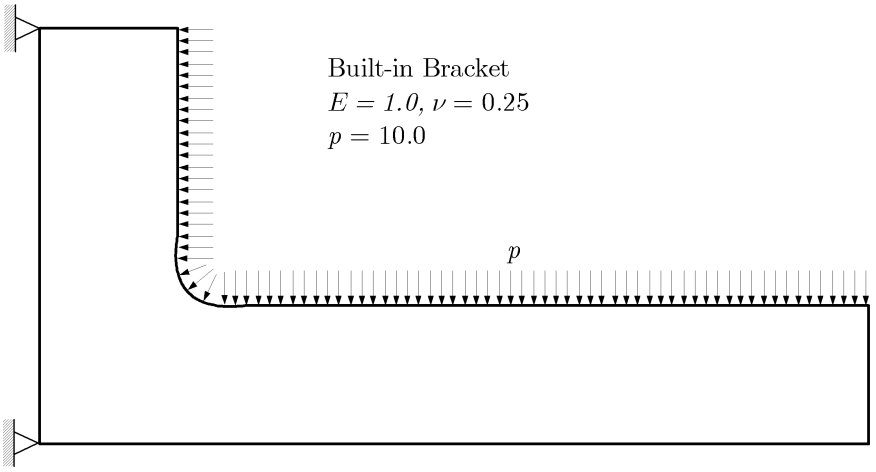


Figure 10: Bracket model with distributed pressure.

This problem was solved using the classical finite element method with the mesh shown on Figure 11. The mesh has 72 nine node Lagrangian quadratic elements. The computed von Mises stress distribution is also shown in Figure 11.

The problem of Figure 10 was also solved using the hp-cloud-GFE shape functions defined in Section 4. The meshfree weighting functions are the C^∞ or C^k finite element weighting functions described in Sections 2.3 and 2.4, respectively. It can be observed in Figure 12 that some of the clouds are non-convex. Therefore, the technique presented in Section 2.4 is required to built the weighting functions. The finite element weighting functions are provided by standard bilinear finite element shape functions. Since the domain has curved boundaries, the meshfree and the linear finite element weighting functions are mapped to the physical domain using quadratic shape functions. In particular, the nine node Lagrangian quadratic shape functions are used.

Figure 12 illustrates the location of linear and C^∞ or C^k finite element weighting functions. All the nodes of elements $37, 41, 45, 49, 53, 57, 61, 65, 69$ have linear finite element weighting functions. The use of standard finite element weighting functions along the left end of the bracket facilitates the imposition of the Dirichlet boundary condition prescribed there. Elements $33, 34, 38, 42, 46, 50, 54, 58, 62, 66, 70$ have some linear and some C^∞ or C^k finite element weighting functions. All other elements have only C^∞ or C^k finite element weighting functions.

The meshfree and the finite element weighting functions are then used to build a meshfree-finite element partition of unity as described in Section 3. Next, hp-cloud-GFE shape functions are built by multiplying this partition of unity by monomials of degree less or equal to two. The

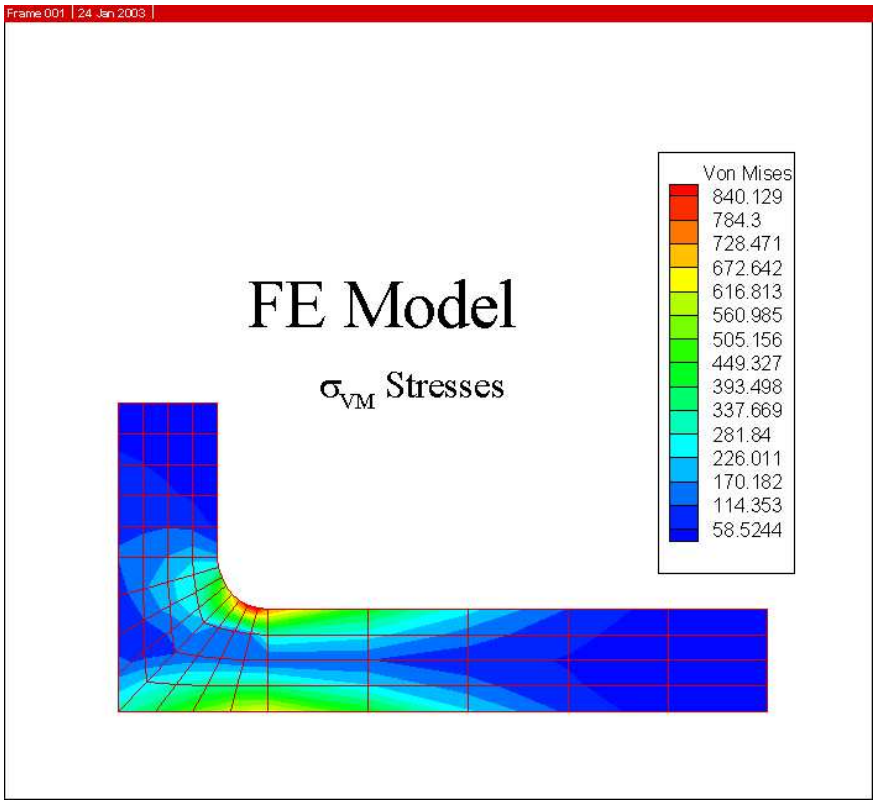


Figure 11: Mesh and von Mises stress distribution for the FE model.

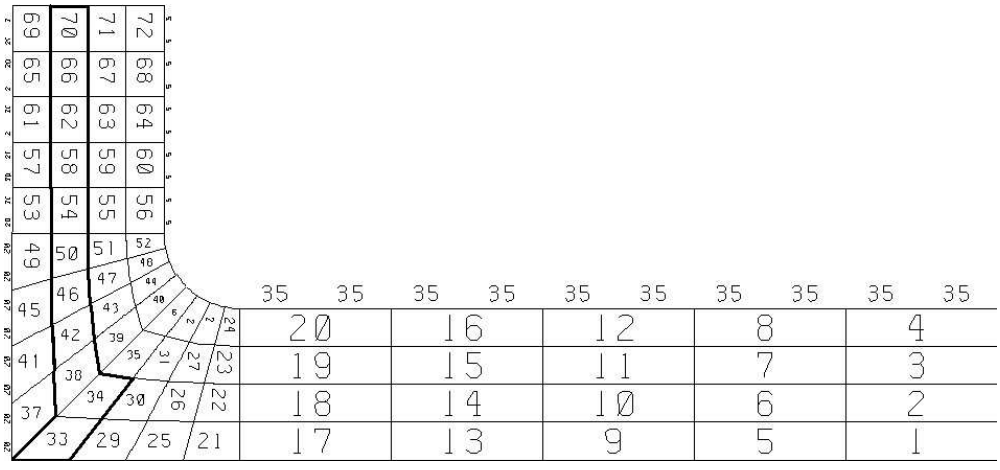


Figure 12: All the nodes of elements $37, 41, 45, 49, 53, 57, 61, 65, 69$ have linear finite element weighting functions.

hp-cloud-GFE shape functions, ϕ_i^α, for a vertex node $\boldsymbol{x}_\alpha$ are given by

$$\varphi_\alpha \times \left\{1, \frac{x-x_\alpha}{h_\alpha}, \frac{y-y_\alpha}{h_\alpha}, \left(\frac{x-x_\alpha}{h_\alpha}\right)\left(\frac{y-y_\alpha}{h_\alpha}\right), \left(\frac{x-x_\alpha}{h_\alpha}\right)^2, \left(\frac{y-y_\alpha}{h_\alpha}\right)^2\right\}$$

as described in Section 4. The monomials in the expression above correspond to the enrichment functions $L_{i\alpha}$ that appear in (9), φ_α is the partition of unity function for node $\boldsymbol{x}_\alpha = (x_\alpha, y_\alpha)$ and h_α is a scaling factors taken as the diameter of the largest finite element sharing node $\boldsymbol{x}_\alpha$.

Figure 13 shows the von Mises stress distribution computed with the hp-cloud-GFE discretization. The scale used is the same as for Figure 11. The stress distribution is almost identical to that provided by the finite element model. No perturbation in the results is observed along the interface between the regions that use meshfree and finite element weighting functions.

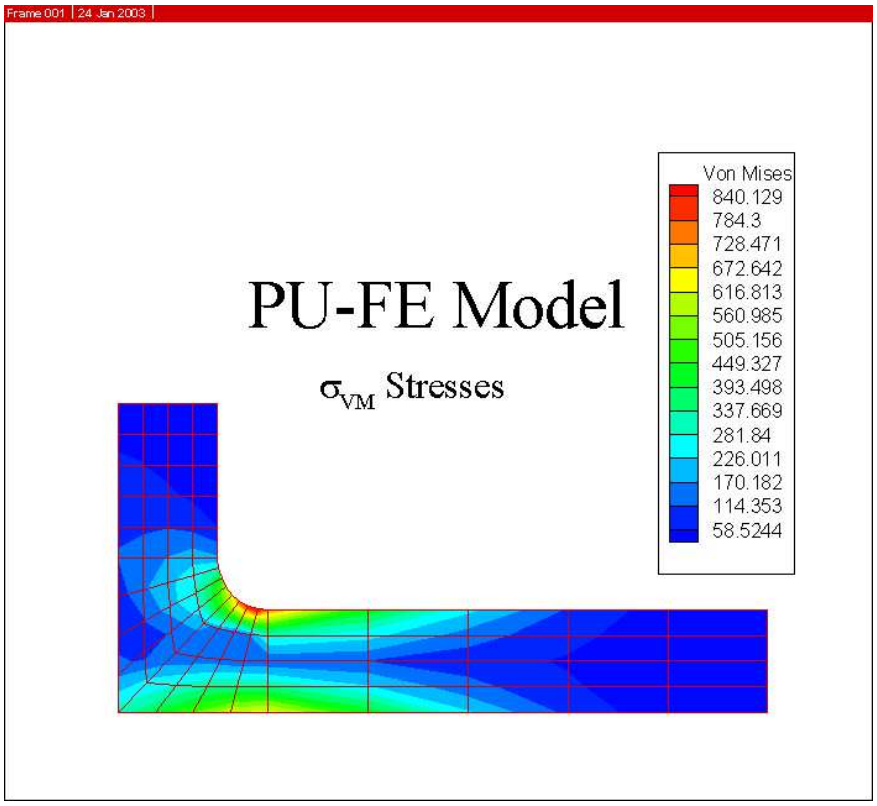


Figure 13: Distribution of von Mises stress computed with the hp-cloud-GFE discretization.

6 Conclusions

A technique for combining finite element discretizations with any meshfree shape function that constitute a partition of unity is presented. Examples of meshfree shape functions with this property are Shepard and moving least squares functions which are used in several popular meshfree

methods. This coupling technique allows the use of meshfree approximations only in parts of the domain where they are strictly needed and finite element discretizations elsewhere. The coupled discretization enjoys the robustness and flexibility of meshfree methods and, to some extent, the computational efficiency of the finite element method. In addition, the proposed approach can be used to facilitate the implementation of Dirichlet boundary conditions and the modeling of complex structures through the use classical elements like rods, shells, rigid bars, etc.

The approximation properties of the combined shape functions are similar to any other partition of unity method like the hp cloud or the generalized finite element method. The procedure is essentially the same in any dimension and can be used with any Lagrangian finite element discretization. A numerical example demonstrating exponential convergence of the proposed shape functions is presented.

Another technique to ameliorate the computation cost of meshfree approximations while retaining some of their attractive features is presented. In this approach, a finite element mesh is used to build meshfree shape functions with supports corresponding to global Lagrangian finite element shape functions defined on the same mesh. As a consequence, the numerical integration of these functions and their products can be efficiently done using the finite element mesh.

The resulting shape functions can also be seem as generalized finite element shape functions with arbitrary degree of regularity. The procedure to build these C^k GFE shape functions is an extension to the one proposed by Edwards [18] and it uses the so-called R-functions. It allows the construction of shape functions with non-convex supports in any spatial dimension and with any degree of regularity.

The proposed C^k GFE shape functions can also be combined with classical finite element shape functions using the proposed coupling technique. A numerical example demonstrating the performance of the proposed C^k GFE shape functions and their coupling with classical finite elements is presented.

References

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

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

[3] 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.

[4] 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.

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

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

[7] 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.

[8] 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.

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

[10] C. A. Duarte and I. Babuška. Mesh-independent directional *p*-enrichment using the generalized finite element method. In K. J. Bathe, editor, *Computational Fluid and Solid Mechanics*, volume 2, pages 1555–1558. Elsevier, June 2001. Proceedings of First MIT Conference on Computational Fluid and Solid Mechanics, June 12-15, 2001.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

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

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

[18] H. C. Edwards. C^∞ finite element basis functions. Technical report, TICAM Report 96-45, The University of Texas at Austin, 1996.

[19] 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.

[20] D. Hegen. Element-free Galerkin methods in combination with finite element approaches. *Computer Methods in Applied Mechanics and Engineering*, 135(1–2):143–166, 1996.

[21] A. Huerta and S. Fernandez-Mendez. Enrichment and coupling of the finite element and meshless methods. *International Journal for Numerical Methods in Engineering*, 48:1615–1636, 2000.

[22] Y. Krongauz and T. Belytschko. Enforcement of essential boundary conditions in meshless approximations using finite elements. *Computer Methods in Applied Mechanics and Engineering*, 131:133–145, 1996.

[23] P. Lancaster and K. Salkauskas. *Curve and Surface Fitting, an Introduction*. Academic Press, San Diego, 1986.

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

[25] Tadeusz 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.

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

[27] W. K. Liu, Y. Chen, S. Jun, T. Belytschko, C. Pan, R. A. Uras, and C. T. Chang. *Overview and Applications of the Reproducing Kernel Particle Methods*. CIMNE, 1996.

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

[29] W. K. Liu, R. A. Uras, and Y. Chen. Enrichment of the finite element method with reproducing kernel particle method. *Journal of Applied Mechanics*, 64:861–870, 1997.

[30] J. M. Melenk. *On Generalized Finite Element Methods*. PhD thesis, The University of Maryland, 1995.

[31] 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.

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

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] D. Organ, M. Fleming, T. Terry, and T. Belytschko. Continuous meshless approximations for nonconvex bodies by diffraction and transparency. *Computational Mechanics*, 18:225–235, 1996.

[38] V. L. Rvachev and T. I. Sheiko. R-functions in boundary value problems in mechanics. *Applied Mechanics Review*, 48(4):151–188, 1995.

[39] V. Shapiro. Theory of R-functions and applications: a primer. Technical Report TR91-1219, Computer Science Department, Cornell University, Ithaca, NY, 1991.

[40] D. Shepard. A two-dimensional function for irregularly spaced data. In *ACM National Conference*, pages 517–524, 1968.

[41] 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.

[42] T. Strouboulis, K. Copps, and Babuška I. The generalized finite element method. *Computer Methods in Applied Mechanics and Engineering*, 190:4081–4193, 2001.

[43] 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.

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

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

[46] 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.