ELSEVIER

Comput. Methods Appl. Mech. Engrg. 139 (1996) 237-262

Computer methods in applied mechanics and engineering

An *h-p* adaptive method using clouds

C. Armando Duarte, J. Tinsley Oden*

TICAM—Texas Institute for Computational and Applied Mathematics, The University of Texas at Austin, Taylor Hall 2.400, Austin, TX 78712, USA

Received 24 April 1996

Abstract

Several computational and mathematical features of the *h-p* cloud method are demonstrated in this paper. We show how *h*, *p* and *h-p* adaptivity can be implemented in the *h-p* cloud method without traditional grid concepts typical of finite element methods. The mathematical derivation of an a posteriori error estimate for the *h-p* cloud method is also presented. Several numerical examples illustrate the main ideas of the method.

1. Introduction

In recent months, there has been growing interest in several new families of methods for the computer simulation of complex problems in science and engineering. They have been categorized under diverse headings such as meshless methods, particle methods, wavelet-type methods, element-free methods, finite point methods, etc. All of these techniques have one common feature: they do not rely on traditional grid concepts typical of finite element, finite difference, or finite volume methods. While many of these methods are not truly 'meshless' in that they may involve somewhat structured collections of quadrature points, they possess a number of interesting computational and mathematical properties which suggest they could be used advantageously for a broad class of scientific and engineering problems. Also, there is a strong connection between these methods and the so-called particle methods which have been in use for a number of decades to study highly non-linear problems in physics and mechanics.

Duarte and Oden [10,11] have shown that the moving least squares functions (MLSF) [12] constitutes a partition of unity and have developed a new meshless method called *h-p* clouds. The basic idea of the method is to multiply a partition of unity (i.e. MLSF) by polynomials or other appropriate class of functions. The resulting functions, called *h-p* clouds, retain good properties of the MLSF, such as high regularity and compactness, and linear combinations of these functions can represent polynomials of any degree. This property allows the implementation of *p* and *h-p* adaptivity in the *h-p* clouds context with the same remarkable features of *h-p* finite element methods but without the burden of a mesh.

Following this introduction, the construction of the *h-p* cloud spaces is discussed in Section 2. Practical aspects of the *h-p* cloud method like the implementation of essential boundary conditions and the handling of domain geometry are also discussed in Section 2. An a posteriori error estimate for the *h-p* cloud method is presented in Section 3. In Section 4 a number of interesting mathematical and

* Corresponding author. Professor, Director, TICAM. e-mail-: oden@ticam.utexas.edu

PII S0045-7825(96)01085-7

computational properties of the *h-p* cloud method are demonstrated along with the solution of three boundary-value problems. The first problem demonstrates how *h*, *p* and *h-p* adaptivity can be implemented in the *h-p* cloud method. The algorithms used are also discussed. The second problem investigates the sensitivity of the method to Poisson locking and compares the performance of the method to the *h* and *p* version of the finite element method. The third problem in Section 4 focuses on the use of *p*-orthotropic approximations using the *h-p* cloud method. Finally, in Section 5 the conclusions are outlined.

2. Construction of *h-p* cloud spaces

In this section, the construction of the *h-p* cloud basis functions is described and some properties of these functions are reviewed. One key idea used in the construction of *h-p* cloud spaces is that of a partition of unity. These class of functions can be used to construct linearly independent functions that have many properties in common with the global basis functions used in the finite element method like local compactness and polynomial reproducing properties. But, unlike the finite element basis functions, the functions used in *h-p* cloud method can be as smooth as desired, even $C^{\infty}(\Omega)$ functions. And, most remarkably, there is no need to partition the domain into smaller subdomains, e.g. finite elements, to construct the *h-p* cloud functions. All that is needed is an *arbitrarily* placed set of nodes which serve as origins for local spectral-type approximations.

2.1. The Partition of unity

Let Ω be an open bounded domain in $\mathbb{R}^n$, $n = 1, 2$ or 3 and Q_N denote an arbitrarily chosen set of N points $\boldsymbol{x}_\alpha \in \Omega$ referred to as *nodes*

$$Q_N = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}, \quad \boldsymbol{x}_\alpha \in \bar{\Omega}$$

We associate with the set Q_N a *finite open covering* of Ω (Fig. 1) in the following way: let ω_α, $\alpha = 1, \ldots, N$, denote a set of segments/balls/spheres (referenced simply as *clouds* from now on) centered at $\boldsymbol{x}_\alpha$ and with radii h_α chosen in such a way that $\mathcal{T}_N := \{\omega_\alpha\}_{\alpha=1}^{N}$ constitutes an open covering of Ω

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

$$\bar{\Omega} \subset \bigcup_{\alpha=1}^{N} \omega_\alpha$$

A class of functions $\mathcal{S}_N := \{\varphi_\alpha\}_{\alpha=1}^{N}$ is called a *partition of unity* subordinated to the open covering $\mathcal{T}_N$ if it possesses the following properties:

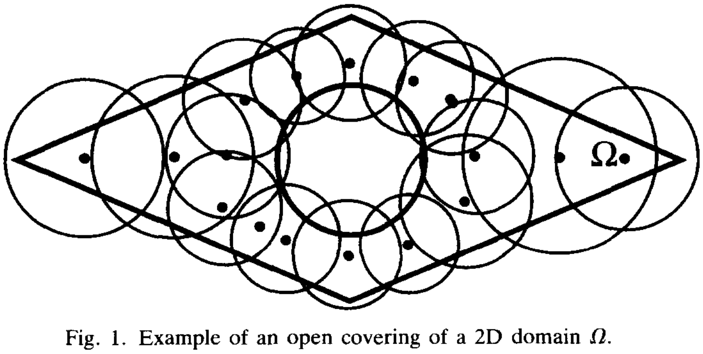


Fig. 1. Example of an open covering of a 2D domain Ω.

(1) $\varphi_\alpha \in C_0^\infty(\omega_\alpha)\,, \quad 1 \leq \alpha \leq N$

(2) $\sum_{\alpha=1}^{N} \varphi_\alpha(\boldsymbol{x}) = 1\,, \quad \forall\, \boldsymbol{x} \in \Omega$

Note that $\varphi_\alpha(\boldsymbol{x})$ may be negative.

The following approach is used in the *h-p* cloud method to build a partition of unity $\mathcal{S}_N$:

Let $\mathcal{W}_\alpha : \mathbb{R}^n \rightarrow \mathbb{R}$ denote a *weighting function* that belongs to the space $C_0^s(\omega_\alpha)$, $s \geq 0$ with the following properties:

. $\mathcal{W}_\alpha(\boldsymbol{y}) \geq 0 \quad \forall\, \boldsymbol{y} \in \Omega$

. $\mathcal{W}_\alpha(\boldsymbol{y}) := W_\alpha(\boldsymbol{y} - \boldsymbol{x}_\alpha)$

where the functions W_α belong to the space $C_0^s(\mathcal{B}_{h_\alpha})$, $s \geq 0$ and $\mathcal{B}_{h_\alpha}$ is a cloud of radius h_α centered at the origin

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

Next, we introduce a family of inner products defined on Ω by

$$(f, g)_y := \sum_{\alpha=1}^{N} \mathcal{W}_\alpha(\boldsymbol{y}) f(\boldsymbol{x}_\alpha) g(\boldsymbol{x}_\alpha)\,, \qquad f, g : \Omega \rightarrow \mathbb{R}, \quad f, g \in C^l(\Omega) l \geq 0 \tag{2.1}$$

ASSUMPTION 1. Given a set of m functions $\mathbb{P} = \{P_1, P_2, \ldots, P_m\}$, $P_i : \Omega \rightarrow \mathbb{R}$, $P_i \in C^l(\Omega)$, $l \geq 0$ for $i = 1, \ldots, m$, the weighting functions $\mathcal{W}_\alpha$ defined above and the functions P_i are such that $\forall\, \boldsymbol{x} \in \bar{\Omega}$ there holds

$$\sum_{k=1}^{m} a_k (P_k, P_l)_x \equiv 0 \quad \text{for } l = 1, \ldots, m \text{ if and only if } a_k \equiv 0 \quad \text{for } k = 1, \ldots, m\,.$$

Necessary and sufficient conditions for the satisfaction of Assumption 1 have been presented in [10,11].

The partition of unity function φ_α associated with the cloud ω_α is defined by

$$\boxed{\varphi_\alpha(\boldsymbol{x}) := \mathbb{P}^t(\boldsymbol{x}) \boldsymbol{A}^{-1}(\boldsymbol{x}) \boldsymbol{B}_\alpha(\boldsymbol{x})} \tag{2.2}$$

where

- $\mathbb{P}(\boldsymbol{x}) := \{P_1(\boldsymbol{x}), P_2(\boldsymbol{x}), \ldots, P_m(\boldsymbol{x})\}^t$, $\exists\, P_j$ s.t. $P_j(\boldsymbol{x}) \equiv 1$,
- $A_{ij}(\boldsymbol{x}) := (P_i, P_j)_x\,, \quad P_i, P_j \in \mathbb{P}$
- $\boldsymbol{B}_\alpha(\boldsymbol{x}) := \mathcal{W}_\alpha(\boldsymbol{x}) \mathbb{P}(\boldsymbol{x}_\alpha)$.

The set of functions $\mathbb{P}$ is, in general, a set of complete polynomials in $\mathbb{R}^n$ and, therefore, they are $C^\infty(\Omega)$ functions. The weighting functions $\mathcal{W}_\alpha$ can be constructed in such a way that they are also $C^\infty(\Omega)$ functions [9]. The following theorem is proved in [10,11]:

THEOREM 1. Let P_i, $i = 1, \ldots, m$, $\exists\, P_j$ s.t. $P_j(\boldsymbol{x}) \equiv 1$ and $\mathcal{W}_\alpha$, $\alpha = 1, \ldots, N$ be the basis functions and the weighting functions used to construct the functions φ_α defined in (2.2). Suppose that P_i, $i = 1, \ldots, m \in C^l(\Omega)$ and $\mathcal{W}_\alpha$, $\alpha = 1, \ldots, N \in C_0^q(\omega_\alpha)$. Then the functions φ_α defined in (2.2) belong to the space $C_0^{\min(l,q)}(\omega_\alpha)$ and $\Sigma_\alpha\, \varphi(\boldsymbol{x}) = 1\, \forall\, \boldsymbol{x} \in \Omega$.

Therefore, if the functions P_i and the weighting functions $\mathcal{W}_\alpha$ are sufficiently smooth, the definition of φ_α given in (2.2) satisfies the definition of a partition of unity. It can also be shown that [10,11]

$$P_j(\boldsymbol{x}) = \sum_{\alpha=1}^{N} P_j(\boldsymbol{x}_\alpha)\varphi_\alpha(\boldsymbol{x}) , \quad \boldsymbol{x} \in \Omega \tag{2.3}$$

REMARK 1. If $\mathbb{P} = \{1\}$, then

$$\varphi_\alpha(\boldsymbol{x}) = \frac{\mathcal{W}_\alpha(\boldsymbol{x})}{\Sigma_\beta \mathcal{W}_\beta(\boldsymbol{x})} \tag{2.4}$$

which are known as Shepard functions [18]. In all numerical examples of Section 4, this partition of unity is used in the construction of the family of functions $\mathcal{F}_N^{k,p}$ defined in Section 2.2. The main appeal of this particular partition of unity is its low computational cost and the simplicity of computation.

2.1.1. *Choice of the weighting functions*

The approach described above is quite general and can be used in any dimension. Also, the choice of the weighting function $\mathcal{W}_\alpha$ is very flexible. These functions can, for example, be constructed in such a way that the decay of the functions φ_α towards the boundary of ω_α is fast. Melenk [16] has shown recently that if a function u_{hp} can approximate u much better on the interior of a ball ω_α than on the whole of ω_α, the decay of the functions φ_α towards the boundary of ω_α can be used to get better local estimates of the error on ω_α.

It should be noted that the support ω_α of the weighting functions does not have to be a circle or a sphere. It can, for example, be an ellipse or a rectangle in two dimensions.

One important situation where the judicious choice of the weighting functions can be beneficial is in the imposition of essential boundary conditions. Lancaster and Salkauskas [12] have shown that if the weighting functions are of the form

$$\overline{\mathcal{W}}_\alpha(\boldsymbol{x}) = \|\boldsymbol{x} - \boldsymbol{x}_\alpha\|_{\mathbb{R}^n}^{-\beta} \mathcal{W}_\alpha \quad \mathcal{W}_\alpha \in C_0^\infty(\omega_\alpha) \tag{2.5}$$

with β an even positive integer, $\boldsymbol{x} \notin \boldsymbol{Q}_N$, then the functions φ_α corresponding to $\mathbb{P} = \{1\}$ constitute a partition of unity and have the property

$$\varphi_\alpha(\boldsymbol{x}_\beta) = \delta_{\alpha\beta} \quad \alpha, \beta = 1, \dots, N \tag{2.6}$$

This property, as demonstrated in Section 2.4, is very useful to impose Dirichlet boundary conditions.

2.2. *The Families* $\mathcal{F}_N^{k,p}$

The fundamental idea in the *h-p* cloud method is the construction of the family of functions $\mathcal{F}_N^{k,p}$ using the partition of unity $\mathcal{S}_N$ defined in the previous section. These class of functions can be constructed at a cost comparable to the computation of finite element shape functions and has the property that, for a proper choice of the set $\mathbb{P}$, we can ensure that $\mathcal{P}_p \subset span\{\mathcal{F}_N^{k,p}\}$ where $\mathcal{P}_p$ denotes the space of polynomials of degree less or equal to p. In this section the construction of the families $\mathcal{F}_N^{k,p}$ is described and some theorems concerning fundamental properties of these functions are stated.

Let $\mathcal{L}_p$ denote a set of tensor-product complete polynomials L_{ijm} in $\mathbb{R}^3$,

$$L_{ijm}(\boldsymbol{x}) = L_i(x_1)L_j(x_2)L_m(x_3) , \quad 0 \leq i, j, m \leq p$$

where L_i is a polynomial of degree i in $\mathbb{R}$. Other sets of complete polynomials can be used as well; e.g. the smallest set of complete polynomials Π_p. In the following, $\mathcal{S}_N^k := \{\varphi_\alpha^k\}_{\alpha=1}^N$ denotes a partition of unity that is $\mathcal{L}_k$*-reducible* for the set $\boldsymbol{Q}_N$; that is, given any element $L_{ijm} \in \mathcal{L}_k$ the following holds $\forall \boldsymbol{x} \in \bar{\Omega}$:

$$L_{ijm}(\boldsymbol{x}) = \sum_{\alpha=1}^{N} L_{ijm}(\boldsymbol{x}_\alpha)\varphi_\alpha^k(\boldsymbol{x}) \tag{2.7}$$

Therefore, k represents the polynomial degree that the partition of unity $\mathcal{S}_N^k$ can represent through linear combinations.

The family of functions $\mathcal{F}_N^{k,p}$ is defined by

$$\boxed{\begin{aligned}\mathscr{F}_N^{k,p} = \{\{\varphi_\alpha^k(\boldsymbol{x})\} \cup \{\varphi_\alpha^k L_{ijm}(\boldsymbol{x})\} : 1 \leqslant \alpha \leqslant N;\quad 0 \leqslant i, j, m \leqslant p,\\ i \text{ or } j \text{ or } m > k;\ p \geqslant k\}\end{aligned}} \tag{2.8}$$

If, instead of tensor-product polynomials, the set Π_p is used, then

$$\boxed{\begin{aligned}\mathscr{F}_N^{k,p} = \{\{\varphi_\alpha^k(\boldsymbol{x})\} \cup \{\varphi_\alpha^k L_{ijm}(\boldsymbol{x})\} : 1 \leqslant \alpha \leqslant N;\quad 0 \leqslant i, j, m \leqslant p,\\ k < i + j + m \leqslant p;\ p \geqslant k\}\end{aligned}} \tag{2.9}$$

The idea behind the definition in (2.8) or (2.9) is to add, hierarchically, appropriate elements to the set $\mathscr{S}_N^k$ such that the resulting set can reproduce, as linear combinations, polynomials of degree $p \geqslant k$. It can be shown [10,11] that those elements are precisely the product of the functions φ_α^k with the elements from the set $\mathscr{L}_p$ that are missing from the set $\mathscr{L}_k$.

For consistent results, regardless of the scale of the problem, the *h-p* cloud functions introduced in (2.8) are implemented using the mappings given by

$$\begin{aligned}&\boldsymbol{F}_\alpha : \hat{\omega} \rightarrow \omega_\alpha\\ &\boldsymbol{F}_\alpha(\boldsymbol{\xi}) = h_\alpha \boldsymbol{\xi} + \boldsymbol{x}_\alpha\,, \quad \boldsymbol{\xi} \in \hat{\omega}\end{aligned} \tag{2.10}$$

where

$$\hat{\omega} := \{\boldsymbol{\xi} \in \mathbb{R}^n : \|\boldsymbol{\xi}\|_{\mathbb{R}^n} < 1\}$$

is a cloud of radius one and

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

is the (interior of the) support of the function φ_α.

Note that a different mapping is used for each cloud ω_α. Each of the mappings (2.10) represents a translation and a dilation. Nonetheless, there are situations where more general mappings involving also rotations are useful. One example is discussed in Section 4.3.

The *h-p* cloud function $\varphi_\alpha L_{ijm}(\boldsymbol{x})$ is implemented in $\mathbb{R}^3$ by

$$\varphi_\alpha L_{ijm}(\boldsymbol{x}) := \varphi_\alpha(\boldsymbol{x})(\hat{L}_{ijm} \circ \boldsymbol{F}_\alpha^{-1}(\boldsymbol{x})) \tag{2.11}$$

where $\hat{L}_{ijm}(\boldsymbol{\xi})$ is a polynomial defined on $[-1,1]^3$. In the numerical examples of Section 4, the polynomials

$$\hat{L}_{ij} = \xi^i \eta^j \quad 0 < i + j \leqslant p$$

are used to build the families $\mathscr{F}_N^{k=0,p}$ in $\mathbb{R}^2$.

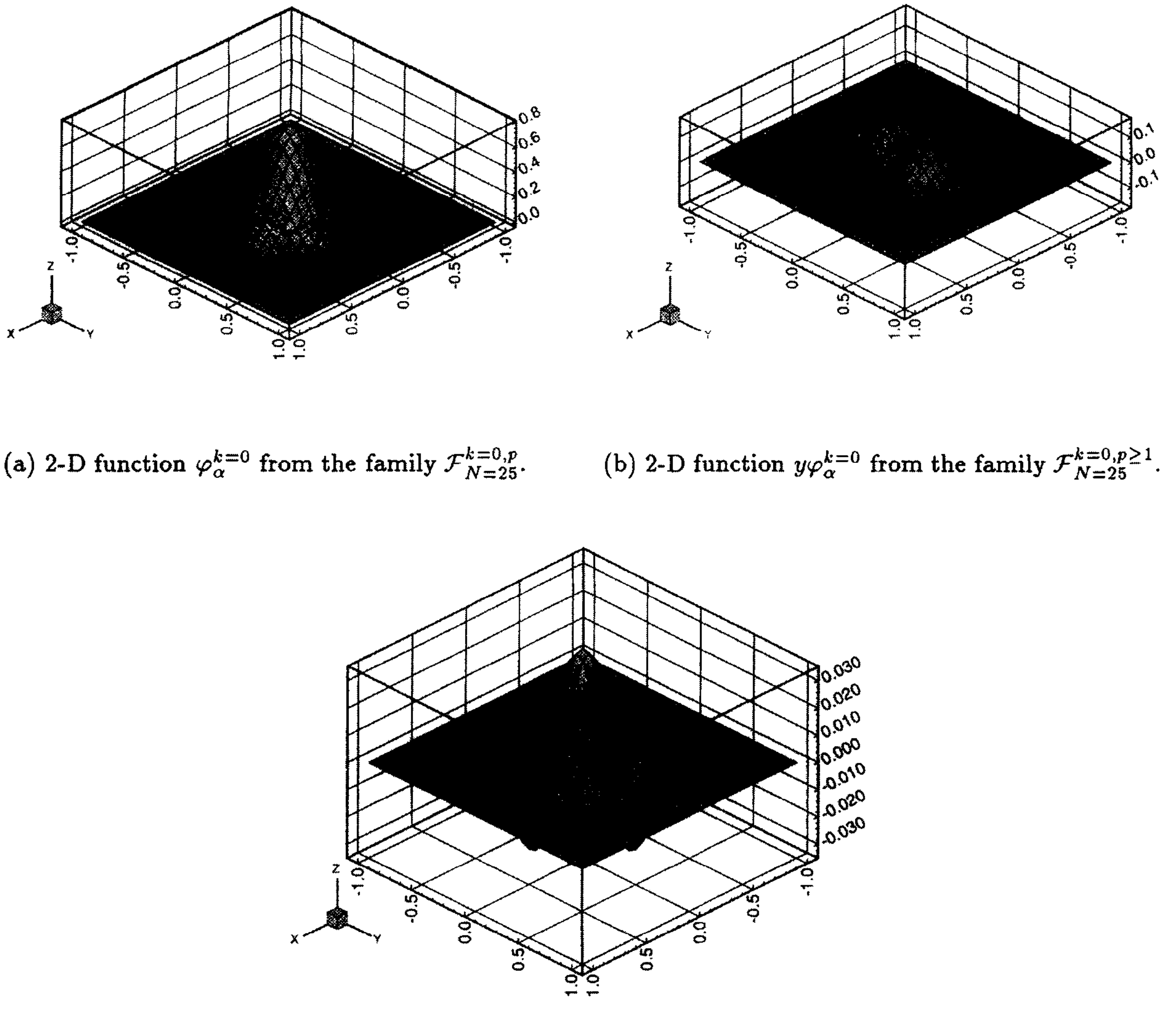
The following theorem is proved in [10].

THEOREM 2. $\mathscr{L}_p \subset span\{\mathscr{F}_N^{k,p}\}$.

Fig. 2(a) shows the function $\varphi_\alpha^{k=0}$ from the family $\mathscr{F}_{N=25}^{k=0}$ associated with a node $\boldsymbol{x}_\alpha$ at the origin. Figs. 2(b) and 2(c) show the functions $y\varphi_\alpha^{k=0}$ and $xy\varphi_\alpha^{k=0}$ from the families $\mathscr{F}_{N=25}^{k=0,p\geqslant 1}$ and $\mathscr{F}_{N=25}^{k=0,p\geqslant 2}$, respectively. A uniform 5 × 5 node arrangement is used to build the partition of unity.

2.2.1. *Beyond polynomials*

In the definition of the family $\mathscr{F}_N^{k,p}$ given in (2.8) and (2.9), the elements from the partition of unity $\mathscr{S}_N$ are multiplied by polynomials. Nonetheless, in cases where there is some knowledge of the function



(a) 2-D function $\varphi_\alpha^{k=0}$ from the family $\mathcal{F}_{N=25}^{k=0,p}$.

(b) 2-D function $y\varphi_\alpha^{k=0}$ from the family $\mathcal{F}_{N=25}^{k=0,p\geq 1}$.

(c) 2-D function $xy\varphi_\alpha^{k=0}$ from the family $\mathcal{F}_{N=25}^{k=0,p\geq 2}$.

Fig. 2. Examples of 2-D *h-p* cloud basis functions.

being approximated, the dimension of the space $\mathcal{F}_N^{k,p}$ can be made smaller without deteriorating its approximating properties. Melenk [16] has shown that harmonic polynomials can be used to locally approximate the solution of the Laplace's equation and generalized harmonic polynomials can be used in the case of the two-dimensional elasticity equations. Note that the dimension of the set of harmonic polynomials grows linearly with p in two dimensions while for the full set of polynomials the dimension grows quadratically.

Another important class of problems where the use of special functions can be advantageous is in the analysis of crack problems. It is well known that (in 2D) near the crack tip the singular part of the solution $\boldsymbol{u}$ can be expressed as [20]

$$\boldsymbol{u} = K_{\mathrm{I}}\sqrt{r}\boldsymbol{F}_1(\theta) + K_{\mathrm{II}}\sqrt{r}\boldsymbol{F}_2(\theta)$$

where (r, θ) is a polar coordinate system at the crack tip, $\boldsymbol{F}_i$, $i = 1, 2$, are smooth functions of θ and K_{I}, K_{II} are stress intensity factors.

Functions with this same kind of singularity can easily be incorporated in the construction of the *h-p* cloud spaces.

2.3. Handling the boundaries

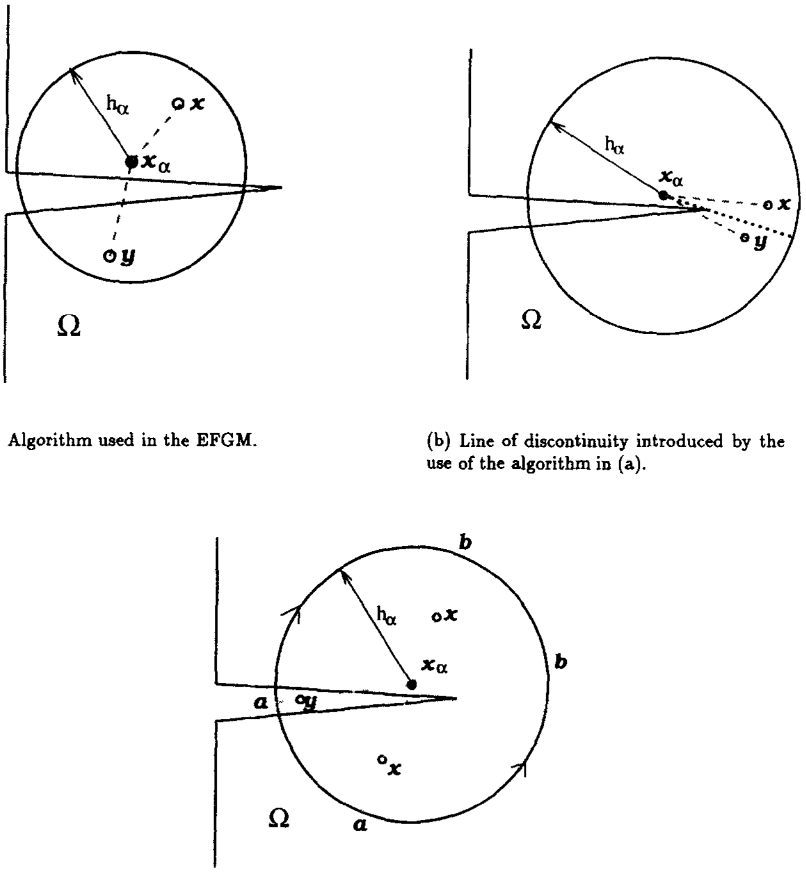
The use of a covering like $\mathcal{T}_N = \{\omega_\alpha\}_{\alpha=1}^N$ may lead to implementation difficulties when the domain Ω is not strictly convex; that is, when the segment connecting two points $x, y \in \Omega$ intersects the boundary $\partial\Omega$. The covering $\mathcal{T}_N$ is also used in other meshless methods, like the element free Galerkin method [4] and the reproducing kernel method [13,14]. Given an arbitrary point $x \in \Omega$, one needs an effective algorithm to find the set of functions φ_α that are non-zero at x. Let us denote this set by *Connex*(x) (for 'connectivity' of clouds relative to x).

To find if a function φ_α belongs to *Connec*(x) it is *not* sufficient to check if $\|x - x_\alpha\|_{\mathbb{R}^n} < h_\alpha$ if $n > 1$, because the domain Ω may not be strictly convex. Fig. 3(a) illustrates the situation when a domain $\Omega \subset \mathbb{R}^2$ has a re-entrant corner and a cloud is a circle. Let φ_α denote the partition of unity function associated with node x_α shown in Fig. 3(a). We would like, for obvious reasons, that $\varphi_\alpha \in$ *Connec*(x) but $\varphi_\alpha \not\in$ *Connec*(y).

In the element-free Galerkin method, a function φ_α belongs to *Connec*(x) if [3,4]:

- $\|x - x_\alpha\|_{\mathbb{R}^n} < h_\alpha$ and
- The segment connecting the points x and x_α does not intersect the boundary $\partial\Omega$.

Although in the situation illustrated in Fig. 3(a) the above algorithm works well, there are situations where the use of this algorithm has the effect of using functions that are *discontinuous*. The situation is illustrated in Fig. 3(b). According to the above algorithm, the function φ_α belongs to *Connec*(x) but does *not* belong to *Connec*(y), which is equivalent to impose that $\varphi_\alpha(y) = 0$. Consequently, the function φ_α will be discontinuous along the dotted line showed in Fig. 3(b).



(a) Algorithm used in the EFGM.

(b) Line of discontinuity introduced by the use of the algorithm in (a).

(c) Algorithm used in the *h-p* cloud method.

Fig. 3. Algorithms used to find the connectivities of a point.

In the *h-p* cloud method, this problem is circumvented by using the following algorithm to check if a function φ_α belongs to *Connec*($\boldsymbol{x}$) (in two dimensions and when a cloud is a circle):

- check if $\|\boldsymbol{x}-\boldsymbol{x}_\alpha\|_{\mathbb{R}^n} < h_\alpha$ and if true,
- let $\boldsymbol{a}$ and $\boldsymbol{b}$ be the intersections of the line defined by the points $\boldsymbol{x}$ and $\boldsymbol{x}_\alpha$ with the circumference $\partial\omega_\alpha$ of radius h_α centered at $\boldsymbol{x}_\alpha$ (see Fig. 3(c)). Let $\boldsymbol{ab}$ and $\boldsymbol{ab}'$ be the two arcs defined by the points $\boldsymbol{a}$ and $\boldsymbol{b}$ on the circumference $\partial\omega_\alpha$. If both $\boldsymbol{ab}$ and $\boldsymbol{ab}'$ intersect the boundary $\partial\Omega$, then $\varphi_\alpha \not\subseteq$ *Connec*($\boldsymbol{x}$).

This algorithm is illustrated in Fig. 3(c). Note that the function $\varphi_\alpha \in$ *Connec*($\boldsymbol{x}$) but $\varphi_\alpha \not\subseteq$ *Connec*($\boldsymbol{y}$). Belytschko and colleagues have recently proposed modifications to their original approach that overcome the discontinuity problem mentioned above [2].

2.4. Imposition of essential boundary conditions

One major difficulty with all meshless methods is the imposition of essential boundary conditions. This is because, in general, the approximating functions do not satisfy the Kronecker-delta condition (2.6). In addition, approximating functions associated with nodes not at the boundary may be non-zero at the boundary. Nonetheless, there are many ways to overcome this problem. Most meshless methods use Lagrange multipliers or penalty methods to impose essential boundary conditions [3,4]. In the *h-p* cloud method, the following approach can also be used:

Suppose that weighting functions of the type given by (2.5) are used to build the partition of unity functions φ_α. If $\mathbb{P}=\{1\}$ (this corresponds to the case $k=0$ in the definitions (2.8) and (2.9)) then

$$\varphi_\alpha(\boldsymbol{x}) = \frac{\overline{\mathcal{W}}_\alpha(\boldsymbol{x})}{\Sigma_{\gamma=1}^{N}\ \overline{\mathcal{W}}_\gamma(\boldsymbol{x})}$$

From the above and the definition of $\overline{\mathcal{W}}_\alpha$, it is immediately shown that

$$\varphi_\alpha(\boldsymbol{x}_\beta) = \delta_{\alpha\beta} \quad \alpha, \beta = 1, \dots, N \tag{2.12}$$

Suppose that the functions $\hat{L}_{ijm}(\boldsymbol{\xi})$ used to build the family of cloud functions $\mathcal{F}_N^{k,p}$ are polynomials given by ($\hat{L}_{ijm}$ are defined in (2.11))

$$\hat{L}_{ijm}(\boldsymbol{\xi}) = \xi^i\eta^j\zeta^m \quad 0 \le i, j, m \le p \quad k = 0 < i+j+m \le p$$

Then

$$\hat{L}_{ijm}(\boldsymbol{0}) = 0 \tag{2.13}$$

In the general case the functions $\hat{L}_{ijm}$ can always be translated such that they are zero at $\boldsymbol{\xi}=\boldsymbol{0}$. From the definition of the mappings $\boldsymbol{F}_\alpha$ and (2.13) we have that

$$\hat{L}_{ijm} \circ \boldsymbol{F}_\alpha^{-1}(\boldsymbol{x}_\beta) = 0 \quad \text{if } \boldsymbol{x}_\beta = \boldsymbol{x}_\alpha\ .$$

Therefore, the *h-p* cloud functions $\varphi_\alpha L_{ijm}(\boldsymbol{x})$ satisfies

$$\varphi_\alpha L_{ijm}(\boldsymbol{x}_\beta) = \varphi_\alpha(\boldsymbol{x}_\beta)(\hat{L}_{ijm} \circ \boldsymbol{F}_\alpha^{-1}(\boldsymbol{x}_\beta)) = 0 \quad \alpha, \beta = 1, \dots, N$$

Consequently, the only non-zero cloud function at a node $\boldsymbol{x}_\alpha$ is the partition of unity function φ_α; moreover, $\varphi_\alpha(\boldsymbol{x}_\alpha)=1$.

Let $\phi_{\alpha j}$ denote an *h-p* cloud function from the space $\mathcal{F}_N^{k,p}$ with $\phi_{\alpha 1} \equiv \varphi_\alpha$. From the above we have that

$$\phi_{\alpha j}(\boldsymbol{x}_\beta) = \delta_{j1}\delta_{\alpha\beta} \quad \alpha, \beta = 1, \dots, N \quad j = 1, \dots, M(\alpha) \tag{2.14}$$

where N is the number of nodes in the discretization and $M(\alpha)$ is the number of cloud functions associated with node $\boldsymbol{x}_\alpha$.

Let the function

$$u_{hp}(\boldsymbol{x}) = \sum_{\alpha=1}^{N} \sum_{j=1}^{M(\alpha)} u_{\alpha j}\phi_{\alpha j}(\boldsymbol{x})$$

be an *h-p* cloud approximation to a function u defined on Ω. Eq. (2.14) implies that

$$u_{hp}(\boldsymbol{x}_\beta) = \sum_{\alpha=1}^{N} \sum_{j=1}^{M(\alpha)} u_{\alpha j}\phi_{\alpha j}(\boldsymbol{x}_\beta) = \sum_{\alpha=1}^{N} \sum_{j=1}^{M(\alpha)} u_{\alpha j}\delta_{j1}\delta_{\alpha\beta} = u_{\beta 1} \quad \forall\, \boldsymbol{x}_\beta \in Q_N$$

where $u_{\beta 1}$ is the coefficient associated with the *h-p* cloud function φ_β. An *h-p* finite element approximation has a similar property: all high-order shape functions are zero at any finite element vertex node and the only non-zero shape function at a vertex node is equal to one there.

If we set $u_{\beta 1} = u(\boldsymbol{x}_\beta)\ \forall\, \boldsymbol{x}_\beta \in Q_N$, then

$$u_{hp}(\boldsymbol{x}_\beta) = u(\boldsymbol{x}_\beta) \quad \forall\, \boldsymbol{x}_\beta \in Q_N \tag{2.15}$$

Now suppose that we want to impose the following boundary condition

$$u_{hp} = \bar{u} \quad \text{on } \Gamma_{\text{D}} \tag{2.16}$$

If (2.15) is true for at least one node $\boldsymbol{x}_\beta \in \Gamma_{\text{D}}$, the boundary condition (2.16) can be imposed following the approach used in the p and *h-p* version of the finite element method or in the spectral method. In the p version of the FEM, for example, the coefficients of the non-zero shape functions at Γ_{D} are evaluated by computing the $H^1(\Gamma_{\text{D}})$ semi-norm projection of $\bar{u}$ over the space spanned by these shape functions [20]. These coefficients are then imposed on the global system of equations. A more detailed account of this approach is the subject of a forthcoming paper.

3. An explicit a posteriori error estimator for *h-p* clouds

In this section, an explicit a posteriori error estimator for the *h-p* cloud method is derived. We follow the proofs of Ainsworth and Oden [1] in the context of the finite element method. We make use of the high regularity of the *h-p* cloud approximations to avoid the calculation of flux jumps in the domain, as is done in the error estimate of Ainsworth and Oden [1]. The error estimator is restricted to the case where the families $\mathcal{F}_N^{k,p}$ are used in a Galerkin method, as is the case of the numerical examples of Section 4.

3.1. *Model problem*

Let $\Omega \subset \mathbb{R}^2$ be a bounded domain with Lipschitz boundary $\partial\Omega$. Consider the model elliptic boundary value problem of finding the solution u of

$$-\Delta u + cu = f \quad \text{in } \Omega \tag{3.1}$$

subject to the boundary conditions

$$\frac{\partial u}{\partial n} = g \quad \text{on } \Gamma_{\text{N}}$$
$$u = 0 \quad \text{on } \Gamma_{\text{D}}$$

where $\partial\Omega = \overline{\Gamma_{\text{D}} \cup \Gamma_{\text{N}}}$, $\Gamma_{\text{D}} \cap \Gamma_{\text{N}} = \emptyset$.

The variational form of this problem is to find $u \in V_{\text{D}}$ such that

$$B(u, v) = L(v) \quad \forall\, v \in V_{\text{D}}$$

where V_{D} is the space

$$V_{\text{D}} = \{v \in H^1(\Omega) : v = 0 \text{ on } \Gamma_{\text{D}}\}$$

and where

$$B(u,v)=\int_\Omega (\nabla u\cdot\nabla v + cuv)\,\mathrm{d}x$$

$$L(v)=\int_\Omega fv\,\mathrm{d}x + \int_{\Gamma_\mathrm{N}} gv\,\mathrm{d}x$$

with $\mathrm{d}x=\mathrm{d}x_1\,\mathrm{d}x_2$.

Suppose that $X^{hp}\subset V_\mathrm{D}$ is a subspace built using the *h-p* cloud family $\mathcal{F}_N^{k,p}$, that is, $X^{hp}=span\{\mathcal{F}_N^{k,p}\}$. Then, the *h-p* cloud approximation of this problem is to find $u_{hp}\in X^{hp}$ such that

$$B(u_{hp},v_{hp})=L(v_{hp})\quad \forall\, v_{hp}\in X^{hp}$$

The error $e=u-u_{hp}$ belongs to the space V_D and satisfies, $\forall\, v\in V_\mathrm{D}$,

$$B(e,v)=B(u,v)-B(u_{hp},v)=L(v)-B(u_{hp},v) \tag{3.2}$$

Moreover, the standard orthogonality condition for the error in the Galerkin projection holds:

$$B(e,v_{hp})=0\quad \forall\, v_{hp}\in X^{hp} \tag{3.3}$$

3.2. The error estimate

Suppose that $X^{hp}\subset(C^2(\Omega)\cap V_\mathrm{D})$. Then, integration by parts of (3.2) gives

$$\begin{aligned}B(e,v)&=\int_\Omega fv\,\mathrm{d}x+\int_{\Gamma_\mathrm{N}} gv\,\mathrm{d}x-\int_{\partial\Omega}\frac{\partial u_{hp}}{\partial n}v\,\mathrm{d}x+\int_\Omega(\Delta u_{hp}-cu_{hp})v\,\mathrm{d}x\\ &=\int_\Omega(f+\Delta u_{hp}-cu_{hp})v\,\mathrm{d}x+\int_{\Gamma_\mathrm{N}}\left(g-\frac{\partial u_{hp}}{\partial n}\right)v\,\mathrm{d}x\quad\forall\, v\in V_\mathrm{D}\end{aligned}$$

or

$$B(e,v)=\int_\Omega rv\,\mathrm{d}x+\int_{\Gamma_\mathrm{N}} Rv\,\mathrm{d}x\quad \forall\, v\in V_\mathrm{D} \tag{3.4}$$

where r is the interior residual

$$r=f+\Delta u_{hp}-cu_{hp}\quad \text{in } \Omega$$

and R is the boundary residual

$$R=g-\frac{\partial u_{hp}}{\partial n}\quad \text{on } \Gamma_\mathrm{N}$$

Each of these quantities is well defined thanks to the smoothness of the data and the regularity of the approximation u_{hp} on the *whole domain*. If the approximation is less regular, the integration by parts may introduce additional terms related to the jumps of derivatives.

The orthogonality property (3.3) may now be used as follows. For a given $v\in V_\mathrm{D}$, let v_{hp} be an approximation to v from the space X^{hp}. Eqs. (3.3) and (3.4) imply that

$$0=B(e,v_{hp})=\int_\Omega rv_{hp}\,\mathrm{d}x+\int_{\Gamma_\mathrm{N}} Rv_{hp}\,\mathrm{d}x \tag{3.5}$$

Combining (3.4) and (3.5) gives

$$B(e,v)=\int_\Omega r(v-v_{hp})\,\mathrm{d}x+\int_{\Gamma_\mathrm{N}} R(v-v_{hp})\,\mathrm{d}x\quad \forall\, v\in V_\mathrm{D} \tag{3.6}$$

Using the fact that $\Sigma_\alpha\,\varphi_\alpha\equiv 1$ on Ω, we can write $\forall\, v\in V_\mathrm{D}$

$$
\begin{aligned}
B(e, v) &= \int_{\Omega} \sum_{\alpha} \varphi_{\alpha} r(v - v_{hp})\, \mathrm{d}\boldsymbol{x} + \int_{\Gamma_{\mathrm{N}}} \sum_{\alpha} \varphi_{\alpha} R(v - v_{hp})\, \mathrm{d}\boldsymbol{x} \\
&= \sum_{\alpha} \int_{\omega_{\alpha} \cap \Omega} \varphi_{\alpha} r(v - v_{hp})\, \mathrm{d}\boldsymbol{x} + \sum_{\alpha} \int_{\partial(\omega_{\alpha} \cap \Omega) \cap \Gamma_{\mathrm{N}}} \varphi_{\alpha} R(v - v_{hp})\, \mathrm{d}\boldsymbol{x}
\end{aligned}
$$

Using the assumption that $\|\varphi_{\alpha}\|_{L^{\infty}(\Omega)} \leqslant C_{\infty}$ and the Cauchy Schwarz inequality, we get

$$
B(e, v) \leqslant C_{\infty} \sum_{\alpha} \{\|r\|_{L^2(\omega_{\alpha} \cap \Omega)} \|v - v_{hp}\|_{L^2(\omega_{\alpha} \cap \Omega)} + \|R\|_{L^2(\partial(\omega_{\alpha} \cap \Omega) \cap \Gamma_{\mathrm{N}})} \|v - v_{hp}\|_{L^2(\partial(\omega_{\alpha} \cap \Omega)\Gamma_{\mathrm{N}})}\} \tag{3.7}
$$

We shall restrict ourselves to the case of families $\mathscr{F}_N^{k=0,p}$. Having this in mind and the fact that $v_{hp} \in X^{hp}$, we have [10]

$$
v_{hp} = \sum_{\alpha} \varphi_{\alpha} v_{\alpha}^{hp} : v_{\alpha}^{hp} \in X_{\alpha}^{hp} ,
$$

where X_{α}^{hp} is the restriction to $(\Omega \cap \omega_{\alpha})$ of the elements from the space

$$
\tilde{X}_{\alpha}^{hp}(\omega_{\alpha}) := span\{\hat{L}_{ijm} \circ \boldsymbol{F}_{\alpha}^{-1}\}
$$

where the mapping $\boldsymbol{F}_{\alpha}$ is defined in (2.10) and the polynomials $\hat{L}_{ijm}$ are those in (2.11).

Also, since $\Sigma_{\alpha} \varphi_{\alpha} \equiv 1$ on Ω, the following can be written:

$$
\|v - v_{hp}\|_{L^2(\omega_{\alpha} \cap \Omega)} = \left\| \sum_{\alpha} \varphi_{\alpha}(v - v_{\alpha}^{hp}) \right\|_{L^2(\omega_{\alpha} \cap \Omega)} = \|v - v_{\alpha}^{hp}\|_{L^2(\omega_{\alpha} \cap \Omega)} \tag{3.8}
$$

Duarte and Oden [10,11] have shown that if the partition of unity φ_{α} is built using only the unity constant ($k = 0$), then $\exists\, v_{\alpha}^{hp} \in X_{\alpha}^{hp}$ such that

$$
\|v - v_{\alpha}^{hp}\|_{L^2(\omega_{\alpha} \cap \Omega)} \leqslant \tilde{C}(p) h_{\alpha} |v|_{H^1(\omega_{\alpha} \cap \Omega)} \tag{3.9}
$$

For generality, lets assume that we can also prove that $\exists\, v_{\alpha}^{hp} \in X_{\alpha}^{hp}$ such that

$$
\|v - v_{\alpha}^{hp}\|_{L^2(\omega_{\alpha} \cap \Omega)} \leqslant C \frac{h_{\alpha}}{p_{\alpha}} |v|_{H^1(\omega_{\alpha} \cap \Omega)} \tag{3.10}
$$

$$
\|v - v_{\alpha}^{hp}\|_{L^2(\partial(\omega_{\alpha} \cap \Omega))} \leqslant C \frac{h_{\alpha}^{1/2}}{p_{\alpha}^{1/2}} |v|_{H^1(\omega_{\alpha} \cap \Omega)} \tag{3.11}
$$

where C is independent of p.

Inserting (3.10) and (3.11) in inequality (3.7), we get

$$
\begin{aligned}
B(e, v) \leqslant C_{\infty} C \Bigg\{ & \sum_{\alpha} \left[\|r\|_{L^2(\omega_{\alpha} \cap \Omega)} \frac{h_{\alpha}}{p_{\alpha}} |v|_{H^1(\omega_{\alpha} \cap \Omega)} \right] \\
& + \sum_{\alpha} \left[\|R\|_{L^2(\partial(\omega_{\alpha} \cap \Omega) \cap \Gamma_{\mathrm{N}})} \frac{h_{\alpha}^{1/2}}{p_{\alpha}^{1/2}} |v|_{H^1(\omega_{\alpha} \cap \Omega)} \right] \Bigg\} \quad \forall v \in V_{\mathrm{D}}
\end{aligned}
$$

Applying the Cauchy Schwarz inequality for the two terms in the sum gives

$$
\begin{aligned}
B(e, v) \leqslant C_{\infty} C \sqrt{2} \Bigg\{ & \left[\sum_{\alpha} \left(\|r\|_{L^2(\omega_{\alpha} \cap \Omega)} \frac{h_{\alpha}}{p_{\alpha}} |v|_{H^1(\omega_{\alpha} \cap \Omega)} \right) \right]^2 \\
& + \left[\sum_{\alpha} \left(\|R\|_{L^2(\partial(\omega_{\alpha} \cap \Omega) \cap \Gamma_{\mathrm{N}})} \frac{h_{\alpha}^{1/2}}{p_{\alpha}^{1/2}} |v|_{H^1(\omega_{\alpha} \cap \Omega)} \right) \right]^2 \Bigg\}^{1/2}
\end{aligned}
$$

Applying the Cauchy Schwarz inequality for each of the sums on α,

$$B(e,v) \leq C_\infty C\sqrt{2}\left\{\left(\sum_\alpha \|r\|^2_{L^2(\omega_\alpha\cap\Omega)}\frac{h_\alpha^2}{p_\alpha^2}\right)\left(\sum_\alpha |v|^2_{H^1(\omega_\alpha\cap\Omega)}\right)\right.$$
$$\left.+\left(\sum_\alpha \|R\|^2_{L^2(\partial(\omega_\alpha\cap\Omega)\cap\Gamma_\mathrm{N})}\frac{h_\alpha}{p_\alpha}\right)\left(\sum_\alpha |v|^2_{H^1(\omega_\alpha\cap\Omega)}\right)\right\}^{1/2}$$
$$= C_\infty C\sqrt{2}\left(\sum_\alpha |v|^2_{H^1(\omega_\alpha\cap\Omega)}\right)^{1/2}\left\{\sum_\alpha\left(\|r\|^2_{L^2(\omega_\alpha\cap\Omega)}\frac{h_\alpha^2}{p_\alpha^2}+\|R\|^2_{L^2(\partial(\omega_\alpha\cap\Omega)\cap\Gamma_\mathrm{N})}\frac{h_\alpha}{p_\alpha}\right)\right\}^{1/2}$$

Using the overlapping condition of the covering $\mathcal{T}_\mathrm{N}=\{\omega_\alpha\}_{\alpha=1}^N$, that is, $\exists\, \rho\in\mathbb{N}$ such that

$$card\{\alpha : \boldsymbol{x}\in\omega_\alpha\}\leq\rho \quad \forall \boldsymbol{x}\in\Omega$$

we can write $\Sigma_\alpha |v|^2_{H^1(\omega_\alpha\cap\Omega)}$ in terms of $|v|^2_{H^1(\Omega)}$:

$$\sum_\alpha |v|^2_{H^1(\omega_\alpha\cap\Omega)} = \sum_\alpha\int_{\omega_\alpha\cap\Omega}|\nabla v|^2\,\mathrm{d}\boldsymbol{x}$$

For each $\boldsymbol{x}\in\Omega$ the integrand will be computed no more than ρ times since each $\boldsymbol{x}\in\Omega$ belongs to no more than ρ clouds ω_α. Thus, we can write

$$\sum_\alpha\int_{\omega_\alpha\cap\Omega}|\nabla v|^2\,\mathrm{d}\boldsymbol{x}\leq\rho\int_\Omega|\nabla v|^2\,\mathrm{d}\boldsymbol{x}=\rho|v|^2_{H^1(\Omega)}$$

Therefore,

$$B(e,v)\leq\rho^{1/2}C_\infty C\sqrt{2}|v|_{H^1(\Omega)}\left\{\sum_\alpha\left(\|r\|^2_{L^2(\omega_\alpha\cap\Omega)}\frac{h_\alpha^2}{p_\alpha^2}+\|R\|^2_{L^2(\partial(\omega_\alpha\cap\Omega)\cap\Gamma_\mathrm{N})}\frac{h_\alpha}{p_\alpha}\right)\right\}^{1/2}$$

or

$$B(e,v)\leq\rho^{1/2}C_\infty\bar{C}\|v\|_{E,\Omega}\left(\sum_\alpha\eta_\alpha^2\right)^{1/2}\quad\forall v\in V_\mathrm{D} \tag{3.12}$$

where

$$\boxed{\eta_\alpha^2=\frac{h_\alpha^2}{p_\alpha^2}\|r\|^2_{L^2(\omega_\alpha\cap\Omega)}+\frac{h_\alpha}{p_\alpha}\|R\|^2_{L^2(\partial(\omega_\alpha\cap\Omega)\cap\Gamma_\mathrm{N})}} \tag{3.13}$$

and we used the fact that $|v|_{H^1(\Omega)}\leq\|v\|_{E,\Omega}$.

Since the fact (3.12) is valid for all $v\in V_\mathrm{D}$, it is valid in particular for e. Thus, we get the following a posteriori error estimate

$$B(e,e)=\|e\|^2_{E,\Omega}\leq\rho^{1/2}C_\infty\bar{C}\|e\|_{E,\Omega}\left(\sum_\alpha\eta_\alpha^2\right)^{1/2}$$

or

$$\boxed{\|e\|_{E,\Omega}\leq\rho^{1/2}C_\infty\bar{C}\left(\sum_\alpha\eta_\alpha^2\right)^{1/2}} \tag{3.14}$$

The contributions η_α from the balls ω_α are denoted *error indicators* and they are used in Section 4 as a basis for guiding local node refinements/enrichments.

4. Numerical examples

In this section three boundary-value problems are solved using the *h-p* cloud method. The first and the third are Poisson's problems in the plane and the second is a three-dimensional elasticity problem with axisymmetry. Each of the problems focus on different aspects of the *h-p* cloud method. The first problem demonstrates the use of *h*, *p* and *h-p* adaptivity. The second problem investigates the sensitivity of the method to Poisson locking and compares the performance of the method to the *h* and *p* versions of the finite element method. Finally, the third problem focuses on the use of *p*-orthotropic approximations in the *h-p* cloud method. In all three problems, the following is adopted:

(i) The Galerkin method is used to generate the set of equations governing the discrete model.

(ii) The essential boundary conditions are imposed using the method of Lagrange multipliers.

(iii) The domain integrations are performed using a background cell structure that exactly covers the domains, i.e. there is no integration point outside of the domain. Nonetheless, there is no relationship between the background cell structures and the nodes x_α used in the discretizations with the exception that denser node arrangements generally require finer background cell structures to guarantee the precise integration of the functions.

(iv) The discrete approximations are built using only the family of functions $\mathcal{F}_N^{k=0,p}$. That is, the partition of unity is composed of Shepard functions as defined in (2.4). Mathematical and numerical analysis performed by Duarte and Oden [10,11] have shown that the family of functions $\mathcal{F}_N^{k=0,p}$ are the best choice for the *h-p* cloud method.

(v) In all problems analyzed, the size of the supports of the *h-p* cloud functions are set by first imposing the condition that every quadrature point belongs to the support of at least one cloud. Then the radius h_α of the clouds are multiplied by a factor $\beta = 1.5$. the algorithm used is described in detail in [10,11]. The use of clouds with larger supports (larger βs) increases considerably the computation costs since it increases the bandwidth of the global matrices. This causes the global matrices to be more expensive to generate and to factor. For this reason, and because the optimal value of β is problem dependent (and therefore not known a priori), the use of small cloud sizes is preferred.

(vi) The weighting functions used to build the partition of unity $\mathcal{S}_N$ are implemented using 'Ridge' functions. More specifically, the weighting functions $\mathcal{W}_\alpha$ are implemented through the composition

$$\mathcal{W}_\alpha(x) := g(r)$$

where $g(r)$ is a quartic $C^3(\Omega)$ B-spline with compact support $[-1, 1]$ and r is the functional

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

Details on the construction of the B-splines can be found in [6].

4.1. *Solution of a poisson problem using h, p and h-p adaptivity*

In this section, the use of the *h*, *p* and *h-p* versions of the *h-p* cloud method to solve boundary-value problems is discussed. The three approaches are described along with the analysis of the following problem:

$$\begin{aligned} -\Delta u &= f && \text{in } \Omega = (0,1)\times(0,1) \\ -\frac{\partial u}{\partial n} &= \hat{g}(x, y) && \text{on } \Gamma_{\mathrm{N}} \\ u &= 0 && \text{on } \Gamma_{\mathrm{D}} \end{aligned} \tag{4.1}$$

where f and $\hat{g}(x, y)$ are chosen to correspond to the exact solution

$$u = \arctan[\alpha(\xi - \xi_0)](x - x^2)(y - y^2)$$

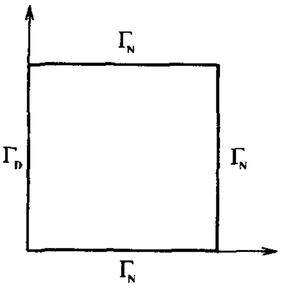


Fig. 4.

and

$$\xi = \frac{x+y}{\sqrt{2}}, \qquad \xi_0 = 0.8, \qquad \alpha = 20$$

The domain Ω and the boundary segments Γ_D and Γ_N are shown in Fig. 4.

4.1.1. *h Adaptivity*

The h adaptive version of the FEM can be implemented in several ways. One of the most successful approaches is based on the use of constrained nodes [7]. This technique guarantees that the h refinement at some region of the domain will not propagate throughout the entire domain while guaranting the continuity of the solution [7]. In the h-p cloud method, the use of constrained nodes is completely unnecessary. The implementation of the h refinement is achieved simply by inserting nodes in the regions of interest. There is no need to add extra nodes or to constraint some of them only to make the solution continuous. Fig. 5 shows the h-p cloud discretization used in the first step of the h adapted solution of problem (4.1). The discretization consists of 25 nodes uniformly distributed and the polynomial order associated with each node is equal to $p = 1$ ($k = 0$). The error in the L^2 norm for this discretization is equal to

$$\frac{\|u - u_h\|_{L^2(\Omega)}}{\|u\|_{L^2(\Omega)}} = 0.2366$$

and the total number of degrees of freedom is equal to 84, including the 9 degrees of freedom corresponding to the Lagrange multipliers used to impose the Dirichlet boundary conditions.

The following algorithm implements h adaptivity in the h-p cloud method (two-dimensional version):

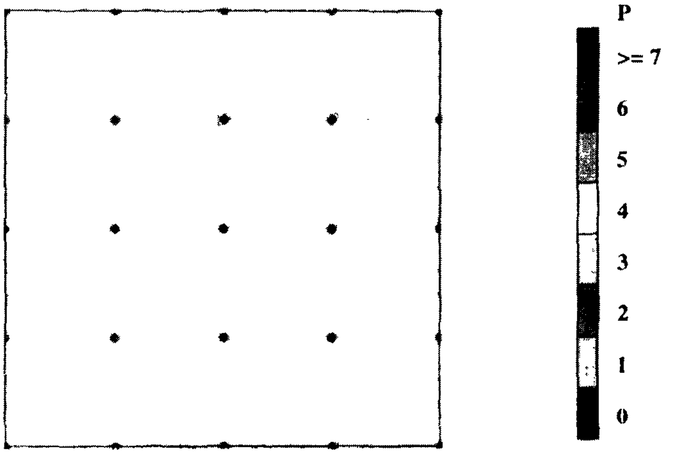


Fig. 5. Discretization used in the step 0 of h, p and h-p adaptation. For h adaptation $p = 1$ and $k = 0$, as shown. For p and h-p adaptation $p = k = 0$ in the step 0.

(i) Compute the error indicators given by (3.13) for each of the clouds $\omega_\alpha \in \mathcal{T}_N$. If the estimated global error is smaller than a predefined value stop.

(ii) Build a list, *refine*, of nodes to be refined in the x- and y-directions using the error indicators computed in step (i).

(iii) For each node x_α with coordinates (x, y) in the list *refine* do:

(iii.i) Compute the distance d_m from this node to the closest node in the *previous* discretization.

(iii.ii) Create new nodes at

$$\left(x, y + \frac{d_m}{2}\right), \left(x, y - \frac{d_m}{2}\right), \left(x + \frac{d_m}{2}, y\right), \left(x - \frac{d_m}{2}, y\right)$$

if $d_m/2 > \mathit{MinNodeDist}$ and nodes at

$$\left(x + \frac{d_m}{2}, y + \frac{d_m}{2}\right), \left(x - \frac{d_m}{2}, y + \frac{d_m}{2}\right), \left(x + \frac{d_m}{2}, y - \frac{d_m}{2}\right), \left(x - \frac{d_m}{2}, y - \frac{d_m}{2}\right)$$

if $d_m * 0.29 > \mathit{MinNodeDist}$. MinNodeDist is the minimum allowed distance between two nodes in the discretization.

In addition, a new node is created *only* if:

- The segment connecting the node x_α to the new node does not intersect the boundary $\partial\Omega$.
- The new node is not too close to other new nodes *created at this step* of h adaptation. Note that it is not necessary to check if a new node is not too close to nodes in the previous discretization since the use of d_m as described above takes care of that.

(iv) Build the open covering $\mathcal{T}_N$ using the new nodes added at this step.

(v) Compute another h-p cloud approximation and go to step (i).

It is worthwhile to mention that:

- The algorithm can easily be modified to implement h refinements in directions other than the x and y directions. Also, the refinements can be anisotropic.
- It is straigthforward to modify the algorithm above in order to perform more than one level of refinement before a new solution is computed.
- The extension of the algorithm to the three-dimensional case is immediate.
- The algorithm guarantees that the resulting node distribution will not have two nodes too close to each other and that new nodes will not be created outside of the domain.
- Step (iii.i) is efficiently implemented using the searching algorithm described in [19].

Fig. 7 illustrate the refinement of a node in two dimensions.

Fig. 6 shows the covering obtained after four steps of h refinement of the initial covering shown in

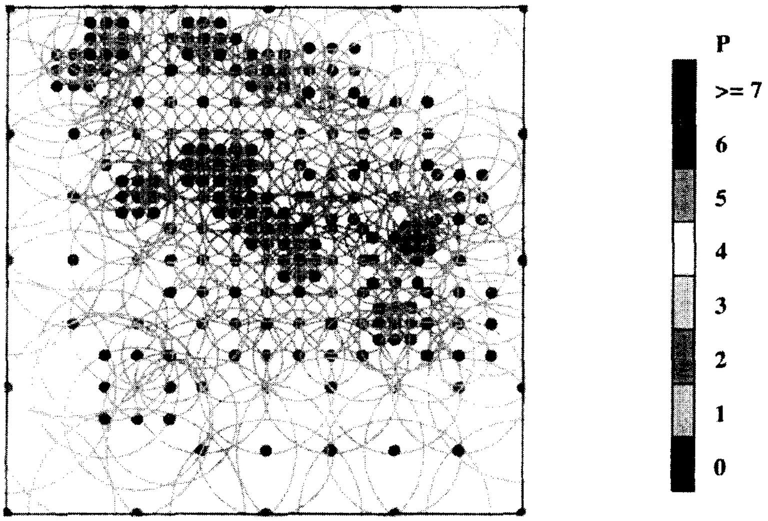


Fig. 6. h adaptation using clouds.

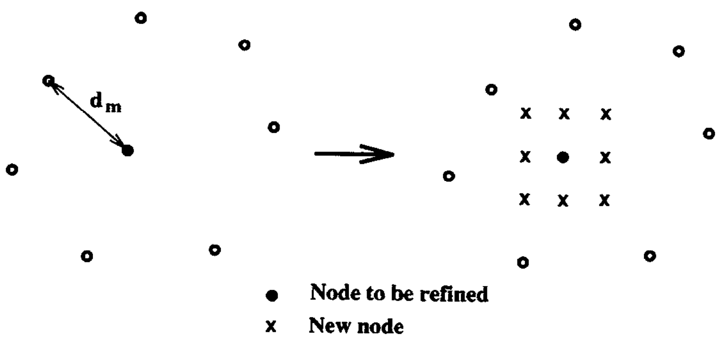


Fig. 7. Isotropic h refinement of a node in two dimensions.

Fig. 5. Note that the polynomial order associated with each cloud ω_α is kept fixed ($k = 0$, $p = 1$) and new nodes are added. The error in the L^2 norm for this discretization is equal to

$$\frac{\|u - u_h\|_{L^2(\Omega)}}{\|u\|_{L^2(\Omega)}} = 0.03262$$

and the total number of degrees of freedom is equal to 786. Fig. 8 shows a contour plot of the error.

4.1.2. *p Adaptivity*

As in the finite element method, the implementation of p adaptivity in the h-p cloud method is easier than the implementation of h or h-p adaptivity. The p version of the h-p cloud method can be implemented in at least two forms. One, for example, can fix the size h_α of the balls and increase the parameter p keeping k fixed. Another possibility is to increase simultaneously k and p. Nonetheless, mathematical analysis and numerical experiments performed by Duarte and Oden [10,11] have shown that the first variant is preferable.

Two open coverings are used to solve Problem (4.1) by the p version of the h-p cloud method. They are represented in Figs. 9 and 5 (but for the p version, $k = p = 0$ in the first step, instead of $k = 0$, $p = 1$ as shown in the figure). The covering in Fig. 9 was generated using a random number generator to assign the coordinates of the nodes and then the radius h_α of each cloud ω_α was automatically set. At each step of the p adaptation, the error indicators (3.13) are computed for each cloud ω_α and then used to select which clouds should be enriched. For this problem, the clouds are enriched by simultaneously

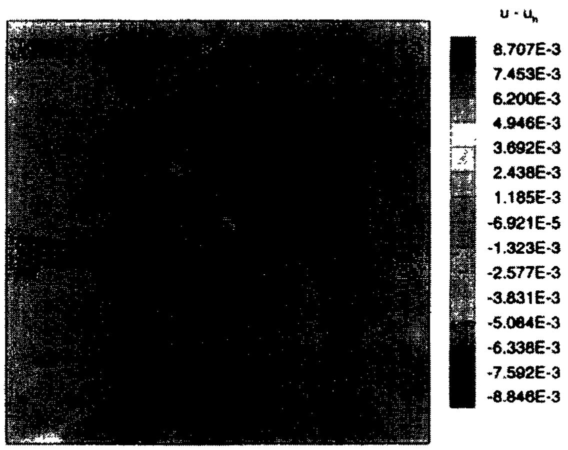


Fig. 8. Pointwise error after h adaptation.

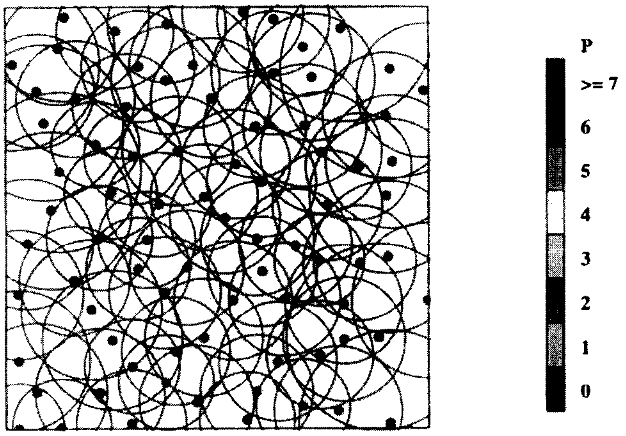


Fig. 9. Randomly generated clouds.

increasing the polynomial orders in the x and y directions. The use of p-orthotropic enrichments, i.e. different polynomial order associated with each direction, is discussed in Section 4.3.

Figs. 12 and 10 show the polynomial orders associated with each cloud after eight steps of p adaptation. The colors represent the polynomial orders of the clouds. It can be observed that each cloud ω_α can have a different polynomial order associated with it, independently of the polynomial orders associated with neighboring clouds. The errors associated with various disretizations used are listed in Table 1.

Fig. 11 shows the three-dimensional plot of the solution obtained using the discretization shown in

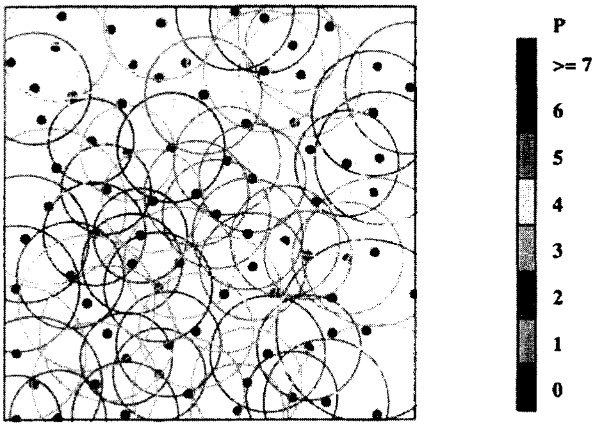


Fig. 10. p Adaptation using clouds.

Table 1
Errors associated with various p discretizations

Discretization	$\frac{\|u-u_p\|_{L_\infty}}{\|u\|_{L_\infty}}$	$\frac{\|u_{,x}-u_{p,x}\|_{L_\infty}}{\|u_{,x}\|_{L_\infty}}$	$\frac{\|u_{,y}-u_{p,y}\|_{L_\infty}}{\|u_{,y}\|_{L_\infty}}$	$\frac{\|u-u_p\|_{L^2}}{\|u\|_{L^2}}$	$\frac{\|u-u_p\|_{H^1}}{\|u\|_{H^1}}$	ndof
Fig. 5	0.9745	0.9923	0.9930	0.8631	0.8509	34
Fig. 12	0.00765	0.04133	0.04071	0.00282	0.02756	527
Fig. 9	0.6924	0.8529	1.1299	0.6898	0.7669	89
Fig. 10	0.00234	0.01878	0.01784	0.00069	0.01009	944

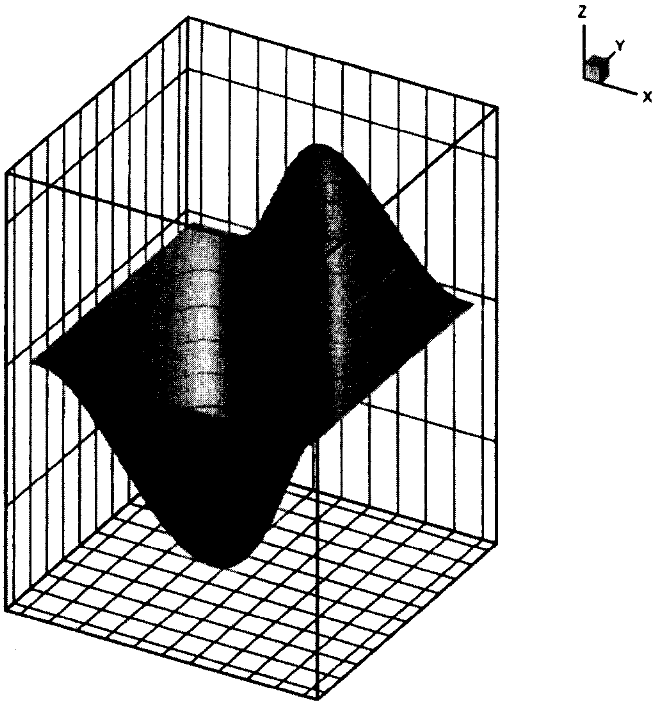


Fig. 11. Temperature distribution after p adaptation.

Fig. 10. It can be observed in Fig. 10 that the clouds with higher-order approximation are those in the part of the domain where the solution has a strong gradient.

4.1.3. *h-p Adaptivity*

The implementation of h-p adaptivity is straightforward after h and p adaptivity have been implemented. The simplest approach is to perform a few h steps to resolve any singularity followed by a number of p steps until the discretization error is below a preset value. This approach is implemented in our h-p cloud code. Other more sophisticated algorithms that use information about the asymptotic behavior of the discretization error, in the same spirit of the Texas-Three-Step algorithm [17], are currently under investigation.

The covering in Fig. 5 (but for the h-p version, $k = p = 0$ in the first step, instead of $k = 0$, $p = 1$ as

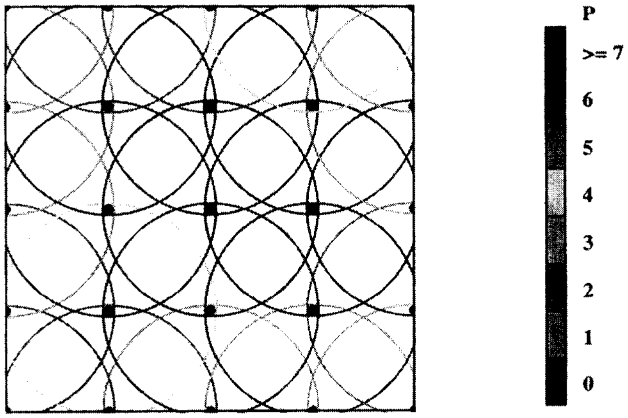


Fig. 12. p Adaptation using clouds.

shown in the figure) is again used as a starting point to solve problem (4.1). The *h-p* adaptive algorithm consists of one step of *h* adaptation where 31 new clouds are added to the discretization followed by *p* adaptation steps. After the *h* step, the radii h_α of all clouds are automatically reset and kept fixed during the *p* adaption steps. The final covering of the domain is represented in Fig. 13. The color of each circle represents the polynomial order associated with each cloud ω_α after the last *p* step.

The errors in the first and in the last step of the *h-p* adaptation are listed in Table 2.

Fig. 14 shows a three-dimensional plot of the x-derivative of the approximate solution obtained using the discretization shown in Fig. 13.

4.2. Three-dimensional elasticity with axisymmetry

In this section, the problem of a thick-walled cylinder under internal pressure and constrained at both ends is analyzed. This problem was proposed by MacNeal and Harder [15] as a standard problem to test finite element accuracy. The domain and the boundary conditions for the problem are represented in Fig. 15.

The set of differential equations governing the solutions of this problem are those from three-dimensional elasticity specialized for the axisymmetric case [5,21]. The following parameters are used in the problem analyzed:

- Young modulus $E = 1000$ MPa.
- Poisson ratio $\nu = 0.4999$.
- Internal pressure $P = 1$ MPa.
- Internal radius $R_i = 3$ mm.
- External radius $R_e = 9$ mm.

The boundary conditions imposed at both ends of the cylinder along with the radial symmetry of the problem confines the material in all but the radial direction. This intensifies the numerical difficulty

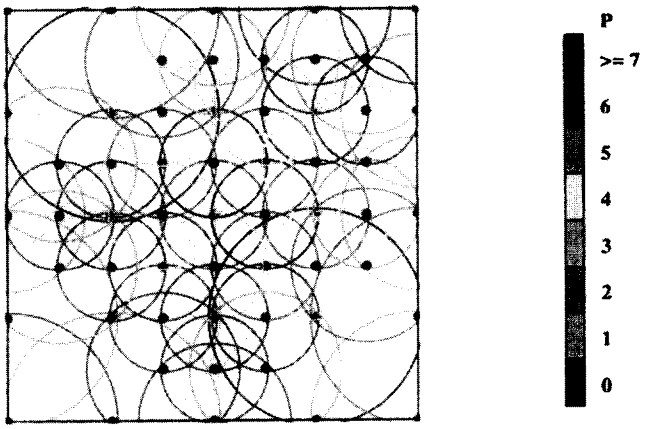


Fig. 13. *h-p* Adaptation using clouds.

Table 2
Errors before and after *h-p* adaptation

Discretization	$\frac{\Vert u - u_p \Vert_{L_\infty}}{\Vert u \Vert_{L_\infty}}$	$\frac{\Vert u_{,x} - u_{p,x} \Vert_{L_\infty}}{\Vert u_{,x} \Vert_{L_\infty}}$	$\frac{\Vert u_{,y} - u_{p,y} \Vert_{L_\infty}}{\Vert u_{,y} \Vert_{L_\infty}}$	$\frac{\Vert u - u_p \Vert_{L^2}}{\Vert u \Vert_{L^2}}$	$\frac{\vert u - u_p \vert_{H^1}}{\vert u \vert_{H^1}}$	ndof
Fig. 5	0.9745	0.9923	0.9930	0.8631	0.8509	34
Fig. 13	0.00376	0.02289	0.02291	0.00109	0.01471	771

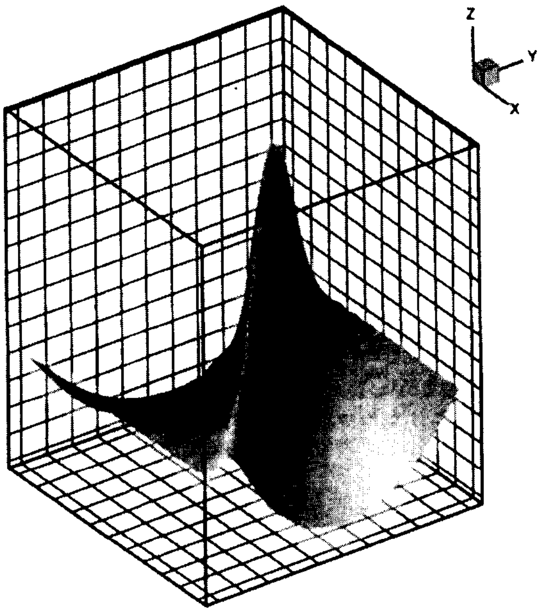


Fig. 14. $\partial u_{hp}/\partial x$ after *h-p* adaptation.

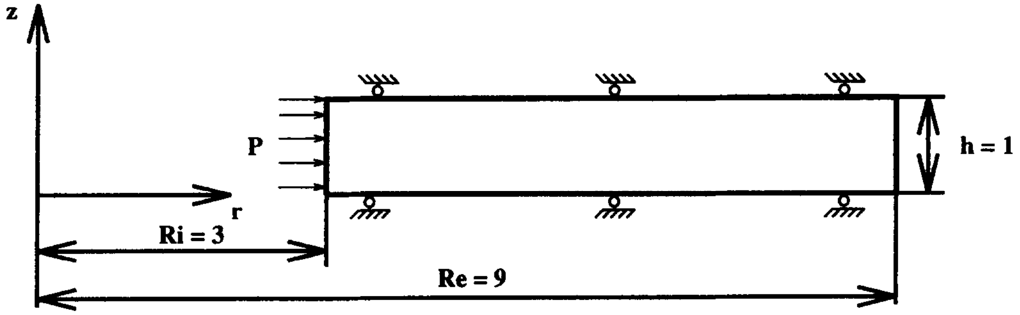


Fig. 15. Domain and boundary conditions for MacNeal–Harder problem.

caused by the near-incompressibility of the material [15]. The analytical solution of this problem is given by [5]:

$$
\begin{aligned}
\sigma_r &= \frac{R_i^2 P}{R_e^2 - R_i^2}\left(1 - \frac{R_e^2}{r^2}\right) \\
\sigma_\theta &= \frac{R_i^2 P}{R_e^2 - R_i^2}\left(1 + \frac{R_e^2}{r^2}\right) \\
\sigma_z &= \frac{2\nu P R_i^2}{R_e^2 - R_i^2} \\
u &= \frac{R_i^2 P(1+\nu) r}{E(R_e^2 - R_i^2)}\left[(1-2\nu) + \frac{R_e^2}{r^2}\right] \\
v &= w = \sigma_{r\theta} = \sigma_{rz} = \sigma_{\theta z} = 0
\end{aligned}
\tag{4.2}
$$

This problem is solved using the *p* version of the *h-p* cloud method. The *h-p* cloud functions used are

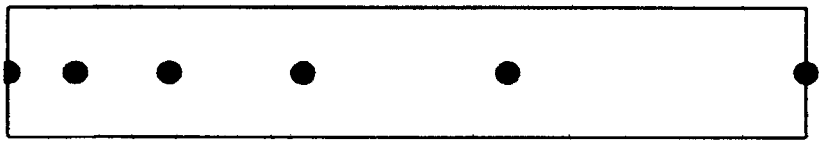


Fig. 16. *h-p* Cloud discretization.

those from the family $\mathscr{F}_{N=6}^{k=0,p}$ with p ranging from zero to four. The nodal arrangement and the supports of the *h-p* cloud functions are depicted in Fig. 16.

The *h-p* cloud results are compared with those presented by MacNeal and Harder [15] for the *h* version of the finite element method and with those presented by Duarte and Barcellos [8] for the *p* version of the finite element method (FEM). The finite element mesh used by NacNeal and Harder and by Duarte and Barcellos is shown in Fig. 17. It should be noted that there is a correspondence between the node arrangements used in the *h-p* cloud discretization and the common boundary of two finite elements in the mesh of Fig. 17.

The displacement of a point at the internal surface of the cylinder is shown in Table 3 for many formulations analyzed by MacNeal and Harder, for the *p* version of the FEM and for the *h-p* cloud method. Some of the finite elements used by MacNeal and Harder use reduced integration to ameliorate the locking effects (this is indicated by (R) in Table 3). The values in Table 3 are normalized with respect to the exact solution given by (4.2). The *h-p* cloud results are the most accurate.

Fig. 18 shows the *h-p* cloud and the *p* FE convergence in the energy norm. The very poor results obtained by both methods when using low-order approximations ($p \leq 1$) is an indication of Poisson locking. Nonetheless, the increase in the polynomial orders of the approximations leads to very high rates of convergence for both methods. The performance of the *h-p* cloud method is almost identical to that of the FEM. However, the *h-p* cloud curve flatten out for $p \geq 4$. The reason for this is that, for $p = 4$, the energy error, $B(\boldsymbol{u} - \boldsymbol{u}_p, \boldsymbol{u} - \boldsymbol{u}_p)$, where $B(.,.)$ is the bilinear form associated with this problem, is already of order 10^{-7} and integration and round-off errors dominate. Fig. 19 shows the error in the radial component of the displacement vector for the *h-p* cloud solution corresponding to $p = 4$.

Fig. 20 shows the convergence of the radial and hoop stress at $r = 3.0$ for the *h-p* cloud method and for the *p* version of the FEM. The horizontal solid lines represent the analytical solution computed from (4.2). The figure shows that the stresses computed using the *h-p* cloud method are more accurate than

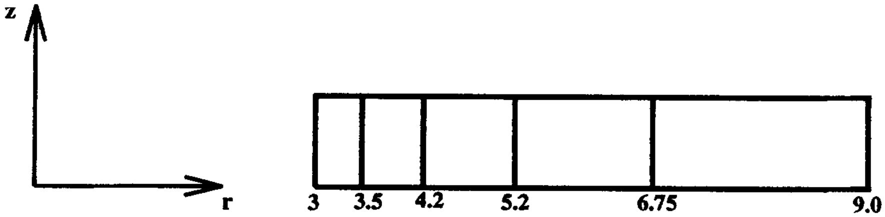


Fig. 17. Finite element mesh [8,15].

Table 3
Computed displacement at $r = R_i$

h FEM [15]	u_h/u	p FEM [8]	u_p/u	h-p Clouds	u_p/u
QUAD2	0.018	$p = 1$	0.0530	$p = 1$	0.0841
QUAD4(R)	0.053	$p = 2$	0.8789	$p = 2$	0.9944
QUAD8(R)	0.967	$p = 3$	0.9991	$p = 3$	1.0000
HEXA8(R)	0.986	$p = 4$	1.0000	$p = 4$	1.0000
HEX20	0.879	$p = 5$	1.0000	$p = 5$	1.0000
HEX20(R)	1.000				

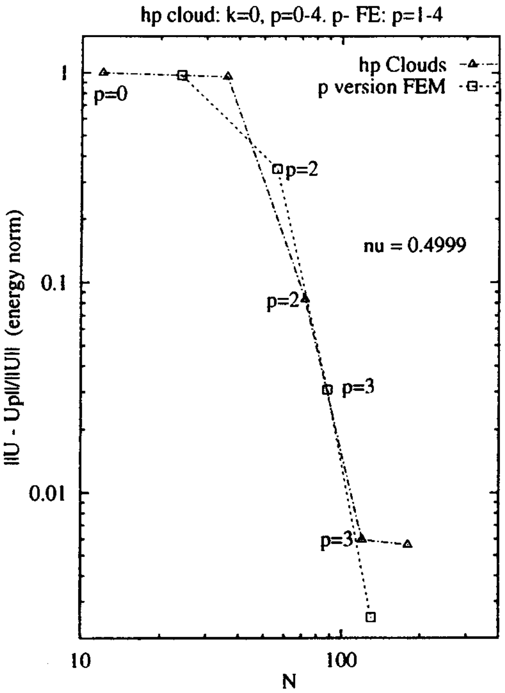


Fig. 18. Convergence in the energy norm.

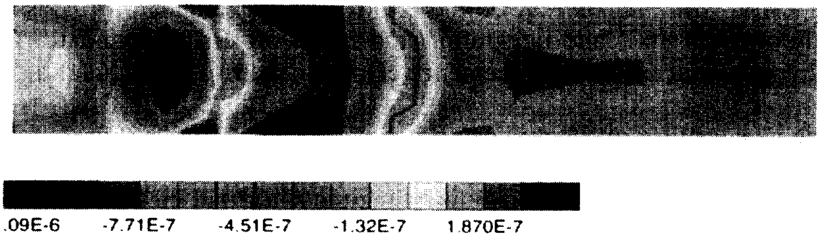


Fig. 19. Pointwise error in the radial component of the displacement vector for the *h-p* cloud solution corresponding to $p = 4$.

the finite element counterparts and also that the convergence is monotonic in the case of the *h-p* cloud method, in contrast with the FEM.

4.3. *p-Orthotropic approximations using clouds*

There are many important practical situations where the solution of a boundary-value problem has a very strong gradient in one direction but is relatively flat in other directions. This is the case, for example, in problems where boundary layers occur or in the analysis of orthotropic materials. One very efficient approach to solve this class of problem is to use *p*-orthotropic approximations, that is, approximations that have different polynomial orders associated with each direction. This technique is well known in the finite element community, but is infrequently used, mainly for practical reasons. In the finite element method the analyst must know a priori the preferential directions of the solution and build the finite element mesh accordingly. Nonetheless, in most cases, such directions are not known a priori and, even when they are, the geometry of the domain may preclude the construction of a finite element mesh along these directions. This constraint is an inherent part of the FEM and forces the use of isotropic approximations to solve this class of problem.

In the *h-p* cloud method, there is much more flexibility in dealing with these types of problems. As mentioned in Section 2.2, associated with each cloud ω_α there is a mapping $F_\alpha : \hat{\omega} \rightarrow \omega_\alpha$ between the

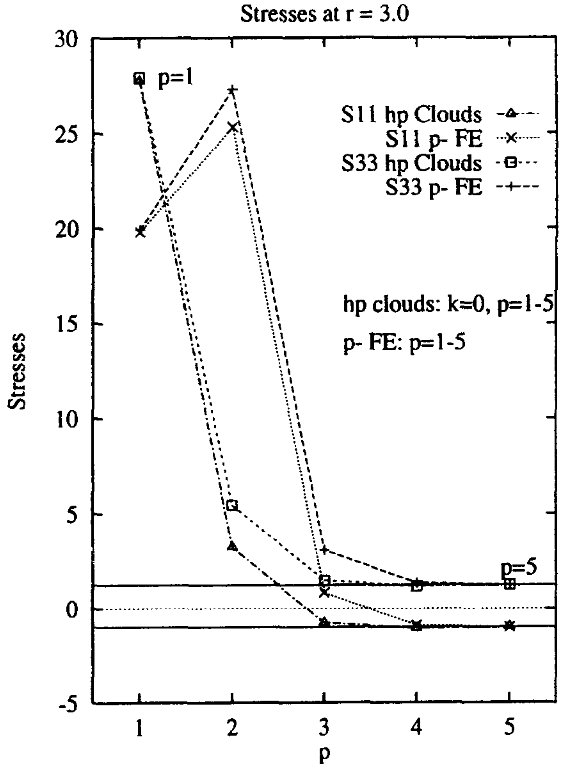


Fig. 20. Convergence of Radical and Hoop Stresses.

master cloud $\hat{\omega}$ and the cloud ω_α. The mappings F_α may be arbitrarily chosen regardless of the mappings used in neighboring clouds. This flexibility may be exploited by the analyst or by an a posteriori error estimator to align the local coordinate system associated with the clouds with the preferential directions of the solution. This approach can always be used, even if the geometry of the domain is complex.

The technique described above is demonstrated in this section through the solution of the following boundary-value problem:

$$\begin{aligned} -\Delta u &= -y\,e^x && \text{in } \Omega \\ -\frac{\partial u}{\partial n} &= -e^x(yn_x + n_y) && \text{on } \partial\Omega \end{aligned}$$

where (n_x, n_y) is the unit normal vector to $\partial\Omega$ and the solution u is set to zero at $(0, 0)$. The domain Ω is represented in Fig. 21.

The solution of this problem,

$$u(x, y) = y\,e^x ,$$

has a strong gradient along the x direction but changes only linearly along the y direction. One might suspect that significant computational effort could be saved if a p-orthotropic approximation could be used to solve this problem. The geometry of the domain, although very simple, makes the use of p-orthotropic finite element approximations somewhat difficult. Triangular finite elements could be used and at least one finite element (near the corner **A** shown in Fig. 21) should be p-isotropic. For the h-p cloud method, the use of the mapping F_α given by (2.10) for all clouds ω_α suffices.

This problem is solved using the p version of the h-p cloud method with p-isotropic and p-orthotropic approximations. Fig. 22 shows the 15 nodes arrangement used in the discretization and the associated open covering. In the p-isotropic case, the families $\mathcal{F}_{N=15}^{k=0,p}$, $0 \leqslant p \leqslant 4$ are used and in the

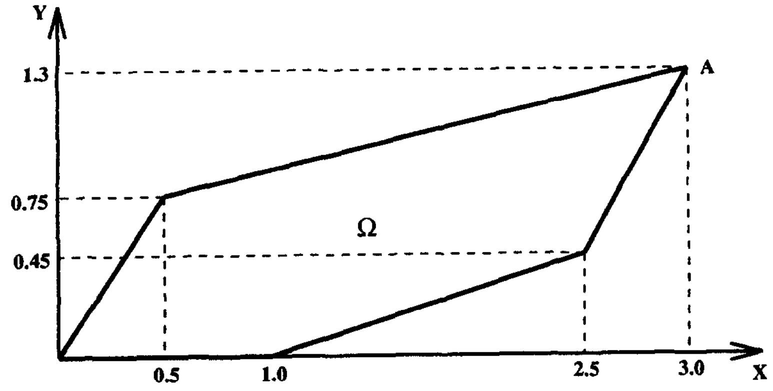


Fig. 21. Domain Ω.

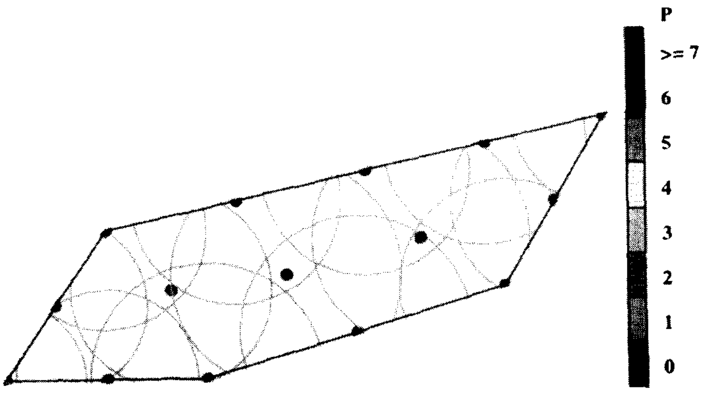


Fig. 22. *h-p* Cloud discretization.

p-orthotropic case the families $\mathcal{F}_{N=15}^{k=0;p_x,p_y}$, $0 \leqslant p_x \leqslant 4$, $0 \leqslant p_y \leqslant 1$ where p_x and p_y denote the polynomial orders in the x and y directions, are used. Fig. 4.22 represents the case $p_x = p_y = 1$.

Fig. 23 shows the convergence in the energy norm for the p-isotropic and for the p-orthotropic cloud approximations. It can clearly be observed that the use of p-orthotropic approximations is advantageous and may lead to very high rates of convergence (up to 15 in the case of p-orthotropic clouds).

Fig. 24 shows the three-dimensional plot of the *h-p* cloud solution corresponding to $p_x = 4$ and $p_y = 1$. The contour plot represents the *h-p* cloud flux in the x direction. The L_∞ error of the solution and fluxes for this discretizations are

$$\frac{\|u - u_p\|_{L_\infty}}{\|u\|_{L_\infty}} = 1.557 \times 10^{-6}$$

$$\frac{\|u,_x - u_{p,x}\|_{L_\infty}}{\|u,_x\|_{L_\infty}} = 2.907 \times 10^{-5}$$

$$\frac{\|u,_y - u_{p,y}\|_{L_\infty}}{\|u,_y\|_{L_\infty}} = 4.875 \times 10^{-5}$$

and the total number of degrees of freedom is 136.

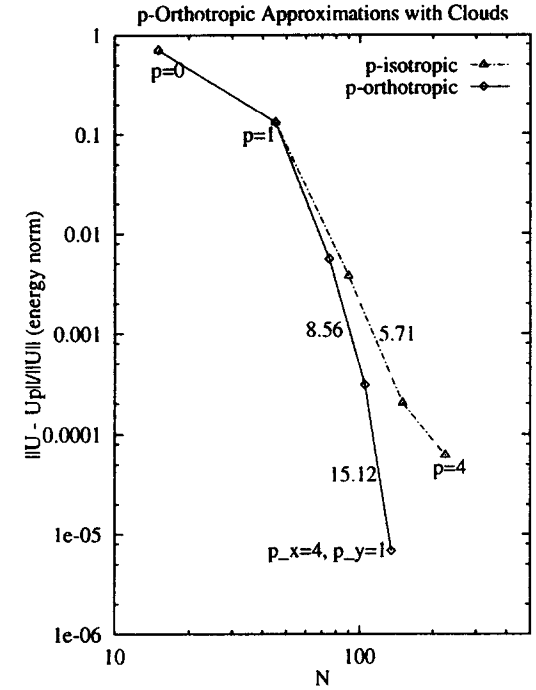


Fig. 23. Convergence of p-isotropic and p-orthotropic clouds.

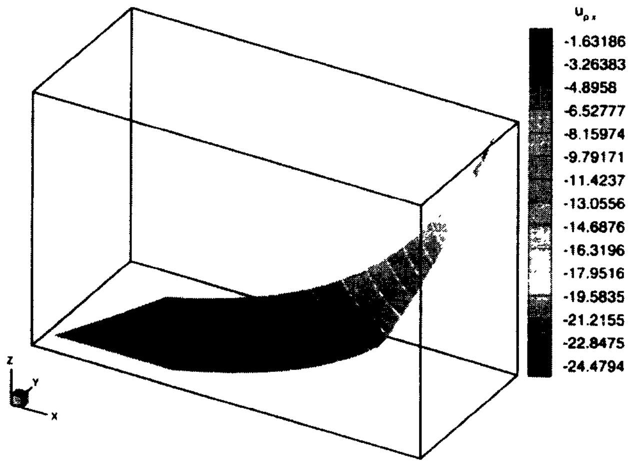


Fig. 24. h-p Cloud solution and flux in the x direction using a p-orthotropic approximation.

5. Conclusions

Several computational and mathematical features of the h-p cloud method are demonstrated in this paper. The fundamental idea behind the method is to use a partition of unity to construct the family $\mathcal{F}_N^{k,p}$ of h-p cloud functions. We demonstrate how this three-parameter family of functions allows the efficient implementation of h, p and h-p adaptivity. The parameter N allows the efficient implementation of h adaptivity by controlling the number of clouds in the discretization and consequently the size of the support of the approximating functions. The parameters k and p represent the polynomial degrees that the partition of unity and the functions $\mathcal{F}_N^{k,p}$ can represent (reproduce) through linear combinations. These parameters are used to implement *p* and *h-p* adaptivity in the *h-p* cloud method leading in many situations to spectral convergence.

Another contribution of this paper is the mathematical derivation of an a posteriori error estimate for the *h-p* cloud method. The high regularity of the functions $\mathscr{F}_N^{k,p}$ allows the derivation of an error estimate that involves only the computation of interior residuals and the residuals where Neumann boundary conditions are prescribed. Although simple, the error estimate is able to detect regions of rapid solution changes. This is demonstrated by numerical examples.

The high flexibility and accuracy of the *h-p* cloud method also represents some challenging problems. One issue needing further study is the development of precise and computationally efficient numerical integration schemes of the *h-p* cloud functions. Two main factors contribute to this cost: the number of integration points used to compute the matrices and the number of non-zero entries in these matrices. The last factor can be controlled through the use of small clouds. The first factor can be controlled by optimizing the number and location of the integration points. This approach is currently under investigation.

Acknowledgments

The support of this work by the Army Research Office (J.T. Oden) under contract DAAL03-92-G-0253, of the National Science Foundation, U.S.–Brazil program under grant INT 9402416, and of the CNPq (Brazil) Graduate Fellowship Program (C.A. Duarte) grant # 200498/92-4 is gratefully acknowledged.

References

[1] M. Ainsworth and J.T. Oden, A posteriori error estimation in finite element analysis, Comput. Mech. Advances, Elsevier, Amsterdam (to appear).

[2] T. Belytschko, Personal communication, 1995.

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

[4] T. Belytschko, Y.Y. Lu and L. Gu, Element-free Galerkin methods, Int. J. Numer. Methods Engrg. 37 (1994) 229–256.

[5] A.P. Boresi and P.P. Lynn, Elasticity in Engineering Mechanics (Prentice-Hall, Englewood Cliffs, NJ, 1974).

[6] C. deBoor, A Practical Guide to Splines, (Springer-Verlag, New York, 1978).

[7] L. Demkowicz, J.T. Oden, W. Rachowicz and O. Hardy, Toward a universal *h-p* adaptive finite element strategy, Part 1. Constrained approximation and data structure, Comput. Methods Appl. Mech. Engrg. 77 (1989) 79–112.

[8] C.A. Duarte and C.S. de Barcellos, Solution of axisymmetric problems using the *p* version of the finite element method, in: 7th Brazilian Symposium on Piping and Pressure Vessels, ABCM, Florianopolis, SC, Brazil, October 1992. In Portuguese.

[9] C.A.M. Duarte, A review of some meshless methods to solve partial differential equations, Technical Report 95-06, TICAM, The University of Texas at Austin, 1995.

[10] C.A.M. Duarte and J.T. Oden. *Hp* clouds—an *hp* meshless method, Numer. Methods Partial Diff. Eqs., to appear.

[11] 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, 1995.

[12] P. Lancaster and K. Salkauskas, Surfaces generated by moving least squares methods, Math. Comput. 37(155) (1981) 141–158.

[13] W.K. Liu and Y. Chen, Wavelet and multiple scale reproducing kernel methods, Int. J. Numer. Methods Fluids 21 (1995) 901–933.

[14] W.K. Liu, S. Jun, S. Li, J. Adee and T. Belytschko, Reproducing kernel particle methods for structural dynamics, Int. J. Numer. Methods Engrg. 38 (1995) 1655–1679.

[15] R.H. MacNeal and R.L. Harder, A proposed set of problems to test finite element accuracy, Finite Elem. Anal. Des. 1 (1985) 3–20.

[16] J.M. Melenk, On generalized finite element methods. Ph.D. Thesis, The University of Maryland, 1995.

[17] J.T. Oden, A. Patra and Y.S. Feng, An *hp* adaptive strategy, in: A.K. Noor, ed., Adaptive, Multilevel and Hierarquical Computational Strategies, AMD-Vol. 157 (ASME, 1992) 23–46.

[18] D. Shepard, A two-dimensional function for irregularly spaced data, in: ACM National Conference, (1968) 517–524.

[19] 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.

[20] B. Szabo and I. Babuska, Finite Element Analysis (John Wiley and Sons, New York, 1991).

[21] S.P. Timoshenko and J.N. Goodier, Theory of Elasticity (McGraw-Hill, London, 1982).