INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng. 2000; **47**:1381–1400

hp-Clouds in Mindlin's thick plate model

Oscar Garcia[1], Eduardo A. Fancello[1], Clovis S. de Barcellos[1] and C. Armando Duarte[2,*,†]

[1]*Grupo de Análise e Projeto Mecânico (GRANTE), Departamento. de Eng. Mecânica, Universidade Federal de Santa Catarina, 88040-900, Brazil*
[2]*Computational Mechanics Company, 7800 Shoal Creek Boulevard, Suite 290E, Austin, TX 78757, U.S.A.*

SUMMARY

In the last few years a number of numerical procedures called as meshless methods have been proposed. Among them, we can mention the diffuse element method, smooth particle hydrodynamics, element free Galerkin method, reproducing kernel particle method, wavelet Galerkin methods, and the so-called *hp*-cloud method. The main feature of these methods is the construction of a collection of open sets covering the domain which are used as support of the classical Galerkin approximation functions. The *hp*-cloud method is focused here because of its advantage of considering from the beginning the *h* and *p* enrichment of the approximation space. In this work we present, to our knowledge, the first results concerning the behaviour of this technique on the solution of Mindlin's moderately thick plate model. It is demonstrated numerically that the behaviour of the method with respect to shear locking is essentially the same as in the *p*-version of the finite element method, namely, the shear locking can be controlled by using *hp* cloud approximations of sufficiently high polynomial degree. The computational implementation of the method and the issue of numerical integration of the stiffness matrix are also discussed. Copyright © 2000 John Wiley & Sons, Ltd.

KEY WORDS: *hp* clouds; meshless methods; Mindlin's plate; locking

1. INTRODUCTION

In the last two decades, the basis for the next generation of numerical methods have been laid down, where these new procedures preclude the requirements for a mesh as support of approximating functions like in the finite element (FEM), boundary element (BEM) and finite volumes methods, among others. The first attempt was made by Gingold and Monaghan [1] through the smooth particle hydrodynamics method for solving astrophysical problems. They used a kernel

* Correspondence to: C. Armando Duarte, Computational Mechanics Company, 7800 Shoal Creek Boulevard, Suite 290E, Austin, TX 78757, U.S.A.
† E-mail: armando@comco.com

Contract/grant sponsor: CNPq; contract/grant numbers: 520093/96-8, 523564/96-1

CCC 0029-5981/2000/081381–20$17.50

Received 15 October 1998
Revised 21 May 1999

estimation technique and a collocation procedure. Nayroles and colleagues [2] proposed the diffuse element method (DEM) in which moving least-squares (MLS) functions [3] are used for building test and trial functions to be used in the Galerkin method framework. The results are not always satisfactory due to incorrect calculation of derivatives of the MLS function and inaccurate imposition of Dirichlet boundary conditions. Next, Belytschko and colleagues [4] implemented the element free Galerkin method (EFG) in which the derivatives of the MLS functions are computed correctly and Lagrange multipliers are used to impose Dirichlet boundary conditions. Although the EFG results presents high accuracy, it has the inconvenience of requiring the inversion of a matrix at each integration point in order to obtain MLS functions that can reproduce linear- or higher-order polynomials. A closely related approach to the EFG method is the reproducing kernel particle method (RKPM) [5]. Belytschko *et al*. [6] have shown that the MLS functions are, in most cases, identical to the underlying approximation technique used in the RKPM. A good review on the RKPM can be found in Reference [7]. A generalization of the DEM, EFGM and RKPM is the *hp*-cloud method proposed by Duarte and Oden [8, 9]. In this technique, there is no matrix inversion during the computation of the trial/test functions because Shepard functions [10] are used for inexpensively building a partition of unity and the associated lowest-order test/trial functions. In order to improve the quality of the results, *hp*-enrichment schemes were devised, where h now means an increase in nodal density and p the increase of nodal parameters corresponding to additional approximation functions. The great advantage of this scheme is the freedom for defining these additional functions. They are, for example, complete polynomials, Trefftz functions, orthotropic expansions, singular functions, and so on. In addition, this enrichment is much easier to implement than in the conventional *hp*-FEM.

The DEM, EFG and *hp*-clouds share difficulties in applying boundary conditions since the test/trial functions usually lack the Kronecker-delta property. Therefore, a number of procedures have been applied like Lagrange multipliers [4] and modified functionals [11], among others. Babuska and Melenk [12, 13] proposed the partition of unity finite element method (PUFEM) which uses, as in the *hp*-cloud method, partition of unity functions to build the approximation spaces. In their implementation, they use a standard finite element partition of unity.

In this paper the *hp*-cloud meshless method is extended for solving Mindlin's plate problem. The main difficulty when solving this class of problems using the finite element method is the shear locking. The locking happens when the approximation functions are unable to meet the requirements for allowing null transversal shear deformations as the plate becomes thin. One effective approach used in the FEM to overcome this difficulty is the use of finite elements of degree $p=3$ or higher (depending on the relative thickness of the plate). The same approach, however cannot be used in meshless methods based on moving least-square functions, such as the DEM, EFGM and RKPM. This happens because the cost of building a MLS approximation that can reproduce polynomials of degree greater or equal to 3 is prohibitively high. At each integration point it would be necessary to invert a matrix of dimensions 10×10 to obtain MLS functions that can reproduce complete cubic polynomials in a two-dimensional manifold. In addition to this, the support of these functions, i.e. the region where they are non-zero, are considerably larger than finite element shape functions of the same polynomial degree. As a consequence, the bandwidth of the stiffness matrix is also much bigger than in the FEM for the same degree of approximation. In contrast, the *hp* cloud framework allows the construction of high-order p approximations without the inversion of any matrix. Furthermore, the dimension of the support of the *hp* cloud functions does not have to increase with the degree p of these functions (as in the case of MLS functions). These properties of the *hp* cloud functions make them very appealing candidates to solve Mindlin's

plate problem in a meshless framework. The issue of computational performance of the *hp*-cloud method as compared with the EFGM and the FEM is addressed in References [9, 14].

Recently, the work of Donning and Liu on meshless methods for Reissner–Mindlin plate problems has been brought to our attention [15]. They use spline functions and an unmodified displacement-based Galerkin method. A uniform nodal arrangement is used to build the spline functions. In the case of two-dimensional domains, nodal points outside of the domain are used in order to maintain all the required properties of the spline-shape functions in the interior of the domain. They demonstrate that the proposed approximation spaces do not exhibit looking and that the resulting stress fields are continuous.

The paper is organized as follows. In Section 2, the Mindlin's plate model is presented in its variational form. The definition of the *hp* cloud approximation functions is described in Section 3. In Section 4, object oriented implementation and code ACCLOUDS++ are described. Several numerical results concerning convergence and locking are presented in Section 5. Finally, in Section 6, the conclusions and suggestions are outlined.

2. MINDLIN'S PLATE MODEL

A Plate is usually considered as the description of a plane structural component having a small dimension, the thickness, compared to its other two dimensions. In order to take advantage of this, particular kinematic assumptions are introduced in order to simplify the elastic fields description. Consider a plate of uniform thickness, t, homogeneous, referred to a three-dimensional Cartesian co-ordinate system with the x-y reference plane lying on the middle surface of the plate. Its domain, Ω, is defined as

$$\Omega = \left\{ (x, y, z) \in \mathbb{R}^3 \,|\, z \in \left[\frac{-t}{2}, \frac{t}{2} \right], (x, y) \in \Sigma, \ \Sigma \in \mathbb{R}^2 \right\}$$

The basic Mindlin's plate model assumptions are:

(i) The thickness meets: $t \ll L/10$ where L is a characteristic plate dimension like, e.g. the smallest plate width or length.
(ii) $\sigma_{zz} \simeq 0$.
(iii) The normal segments remain straight and unstretched after deformation.

2.1. Kinematic assumptions and plate stresses

Here, the membrane deformations are not accounted for since they are uncoupled from the bending and shear deformations. Hence, the displacement field is described by

$$\mathbf{u}(x, y, z) = \begin{Bmatrix} u_1(x, y, z) \\ u_2(x, y, z) \\ u_3(x, y, z) \end{Bmatrix} = \begin{Bmatrix} -z\theta_x(x, y) \\ -z\theta_y(x, y) \\ w(x, y) \end{Bmatrix} \tag{1}$$

where $w(x, y)$ is the transversal displacement of a point initially lying on the reference plane, Σ, and θ_x and θ_y are the normal segments rotations around their midpoints with respect to the $-y$

and x directions, respectively. The associated linear strain deformation tensor is

$$\boldsymbol{\varepsilon}(\mathbf{u}) = \tfrac{1}{2}(\nabla\mathbf{u} + \nabla^{\mathrm{T}}\mathbf{u}) = \begin{bmatrix} -z\theta_{x,x} & -\frac{z}{2}(\theta_{x,y} + \theta_{y,x}) & \frac{1}{2}(w_{,x} - \theta_x) \\ & -z\theta_{y,y} & \frac{1}{2}(w_{,y} - \theta_y) \\ symm & & 0 \end{bmatrix} \tag{2}$$

The strain deformation components can be rearranged in a vector form according to their nature–bending or shear deformation. This is accomplished by defining

$$\boldsymbol{\varepsilon}_{\mathrm{b}}(\mathbf{u}) = \begin{bmatrix} -\theta_{x,x} \\ -\theta_{y,y} \\ -(\theta_{x,y} + \theta_{y,x}) \end{bmatrix} \tag{3}$$

$$\boldsymbol{\varepsilon}_{\mathrm{s}}(\mathbf{u}) = \begin{bmatrix} w_{,x} - \theta_x \\ w_{,y} - \theta_y \end{bmatrix} \tag{4}$$

Since $t \ll L$, it is assumed that $\sigma_{zz} \simeq 0$. Therefore, for a linear elastic isotropic material, the Hooke's Law implies that

$$\boldsymbol{\sigma}_{\mathrm{b}}(\mathbf{u}) = \begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \tau_{xy} \end{Bmatrix} = z\frac{E}{(1-\nu^2)} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \dfrac{(1-\nu)}{2} \end{bmatrix} \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \gamma_{xy} \end{Bmatrix} = z\mathbf{C}_1\boldsymbol{\varepsilon}_{\mathrm{b}}(\mathbf{u}) \tag{5}$$

$$\boldsymbol{\sigma}_{\mathrm{s}}(\mathbf{u}) = \begin{Bmatrix} \tau_{xz} \\ \tau_{yz} \end{Bmatrix} = \frac{E}{(1-\nu^2)} \begin{bmatrix} \dfrac{(1-\nu)}{2} & 0 \\ 0 & \dfrac{(1-\nu)}{2} \end{bmatrix} \begin{Bmatrix} \gamma_{xz} \\ \gamma_{yz} \end{Bmatrix} = \mathbf{C}_2^c\boldsymbol{\varepsilon}_{\mathrm{s}}(\mathbf{u}) \tag{6}$$

where $\boldsymbol{\sigma}_{\mathrm{b}}(\mathbf{u})$ and $\boldsymbol{\sigma}_{\mathrm{s}}(\mathbf{u})$ are the bending and transverse shear stresses at an arbitrary point, respectively. The plate stress resultants can be written as

$$\begin{aligned} \mathbf{M}(\mathbf{u}) = \begin{bmatrix} M_{xx} \\ M_{yy} \\ M_{xy} \end{bmatrix} &= \int_{-t/2}^{t/2} z\,\boldsymbol{\sigma}_{\mathrm{b}}(\mathbf{u})\,\mathrm{d}z = \int_{-t/2}^{t/2} z^2\,\mathbf{C}_1\boldsymbol{\varepsilon}_{\mathrm{b}}(\mathbf{u})\,\mathrm{d}z \\ &= \frac{t^3}{12}\,\mathbf{C}_1\boldsymbol{\varepsilon}_{\mathrm{b}}(\mathbf{u}) = \mathbf{C}_{\mathrm{b}}\boldsymbol{\varepsilon}_{\mathrm{b}}(\mathbf{u}) \end{aligned} \tag{7}$$

$$\begin{aligned} \mathbf{Q}(\mathbf{u}) = \begin{bmatrix} Q_x \\ Q_y \end{bmatrix} &= \int_{-t/2}^{t/2} \boldsymbol{\sigma}_{\mathrm{s}}(\mathbf{u})\,\mathrm{d}z = \int_{-t/2}^{t/2} k_c\,\mathbf{C}_2\boldsymbol{\varepsilon}_{\mathrm{s}}(\mathbf{u})\,\mathrm{d}z \\ &= k_c t\,\mathbf{C}_2\boldsymbol{\varepsilon}_{\mathrm{s}}(\mathbf{u}) = \mathbf{C}_{\mathrm{s}}\boldsymbol{\varepsilon}_{\mathrm{s}}(\mathbf{u}) \end{aligned} \tag{8}$$

where $\mathbf{C}_{\mathrm{b}} = (t^3/12)\mathbf{C}_1$, $\mathbf{C}_{\mathrm{s}} = k_c t\mathbf{C}_2$ and $k_c = \frac{5}{6}$ is the shear factor, [16]. The implied convention of these stress resultants are shown in Figure 1.

2.2. Equilibrium equations

Consider a plate under the domain distributed moments $\mathbf{m}$, transversal loads q, Neumann boundary conditions $\bar{\mathbf{m}}$ and $\bar{q}$ on $\partial\Sigma_{\mathrm{N}}$ and Dirichlet boundary conditions $\boldsymbol{\theta} = \bar{\boldsymbol{\theta}}$, $w = \bar{w}$ on $\partial\Sigma_{\mathrm{D}}$. For each dual pair $(\bar{\mathbf{m}}, \bar{\boldsymbol{\theta}})$ and $(\bar{q}, \bar{w})$, the boundary is accordingly divided into $\partial\Sigma_{\mathrm{D}}$ and $\partial\Sigma_{\mathrm{N}}$ such that

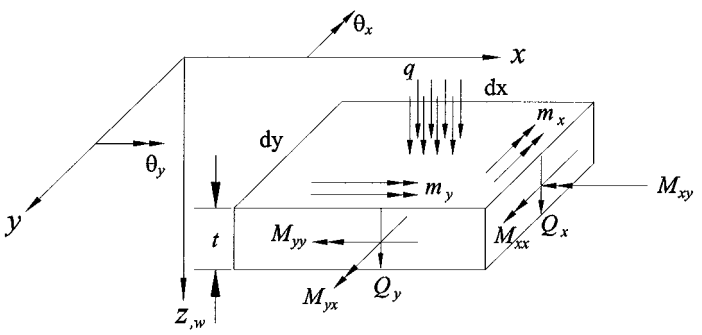

Figure 1. Plane stress resultants.

$\partial\Sigma = \partial\Sigma_D \cup \partial\Sigma_N$ and $\partial\Sigma_D \cap \partial\Sigma_N = \emptyset$. The classical equilibrium equations and boundary conditions are:

$$\operatorname{div}\mathbf{M} - \mathbf{Q} = \mathbf{m} \tag{9}$$

$$\operatorname{div}\mathbf{Q} + q = 0, \quad \forall \mathbf{x} \in \Sigma$$

$$\mathbf{Mn} = -\bar{\mathbf{m}} \tag{10}$$

$$\mathbf{Qn} = \bar{q} \quad \forall \mathbf{x} \in \partial\Sigma_N$$

$$\boldsymbol{\theta} = \bar{\boldsymbol{\theta}} \tag{11}$$

$$w = \bar{w} \quad \forall \mathbf{x} \in \partial\Sigma_D$$

2.3. *Variational principles*

The principle of minimum potential energy implies that the weak formulation for equilibrium can be stated as finding $\mathbf{u}(\boldsymbol{\theta}, w)$, $(\boldsymbol{\theta}, w) \in \mathscr{K}in$, such that, Washizu [17]:

$$\int_\Sigma [\mathbf{M}(\mathbf{u})\cdot\boldsymbol{\varepsilon}_b(\mathbf{u}^*) + \mathbf{Q}(\mathbf{u})\cdot\boldsymbol{\varepsilon}_s(\mathbf{u}^*)]\,d\Sigma = \int_\Sigma q w^*\,d\Sigma + \int_\Sigma \mathbf{m}\cdot\boldsymbol{\theta}^*\,d\Sigma + \int_{\partial\Sigma_N} \bar{q} w^*\,d\partial\Sigma + \int_{\partial\Sigma_N} \bar{\mathbf{m}}\cdot\boldsymbol{\theta}^*\,d\partial\Sigma \quad \forall w^*, \boldsymbol{\theta}^* \in \mathscr{V}ar \tag{12}$$

where $\mathscr{K}in$ and $\mathscr{V}ar$ are defined as follows:

$$\mathscr{K}in = \{\boldsymbol{\theta}, w: \boldsymbol{\theta}, w \in H^1(\Sigma), \boldsymbol{\theta} = \bar{\boldsymbol{\theta}},\ w = \overline{w} \quad \forall\, \mathbf{x} \in \partial\Sigma_D\} \tag{13}$$

$$\mathscr{V}ar = \{\boldsymbol{\theta}^*, w^*: \boldsymbol{\theta}^*, w^* \in H^1(\Sigma)\boldsymbol{\theta}^* = \mathbf{0},\ w = 0 \quad \forall\, \mathbf{x} \in \partial\Sigma_D\}. \tag{14}$$

Now, we have to note that the Dirichlet boundary conditions are strongly imposed in this variational statement through the $\mathscr{K}in$ and $\mathscr{V}ar$ definitions. Alternatively, we can impose such boundary conditions through Lagrange multipliers by modifying the variational statement. Here, these Lagrange multipliers are identified with the boundary reactions. Taking this into account, we can modify the variational statement by explicitly considering such reactions in terms of the displacement field. An advantage of this procedure is to avoid an increase of the size of the algebraic equation to be solved and the presence of a null diagonal submatrix, but, on the other hand, the convergence rate will be that of the resultant stresses, which is lower than that for the displacement ones. This approach was also used by Lu *et al.* [11] for the EFGM scheme and, in

the present study, it consists in determining the displacement field $\mathbf{u}(\boldsymbol{\theta}, w)$, $(\boldsymbol{\theta}, w) \in H^1(\Sigma)$ such that

$$\begin{aligned}
&\int_\Sigma [\mathbf{M}(\mathbf{u})\cdot\boldsymbol{\varepsilon}_\mathrm{b}(\mathbf{u}^*) + \mathbf{Q}(\mathbf{u})\cdot\boldsymbol{\varepsilon}_\mathrm{s}(\mathbf{u}^*)]\,\mathrm{d}\Sigma - \int_\Sigma q w^*\,\mathrm{d}\Sigma - \int_\Sigma \mathbf{m}\cdot\boldsymbol{\theta}^*\,\mathrm{d}\Sigma \\
&\quad - \int_{\partial\Sigma_\mathrm{N}} \bar{q} w^*\,\mathrm{d}\partial\Sigma_\mathrm{N} - \int_{\partial\Sigma_\mathrm{N}} \bar{\mathbf{m}}\cdot\boldsymbol{\theta}^*\,\mathrm{d}\partial\Sigma_\mathrm{N} \\
&\quad + \int_{\partial\Sigma_\mathrm{D}} \mathbf{M}_n(\mathbf{u})\cdot\boldsymbol{\theta}^*\,\mathrm{d}\partial\Sigma_\mathrm{D} - \int_{\partial\Sigma_\mathrm{D}} Q_n(\mathbf{u}) w^*\,\mathrm{d}\partial\Sigma_\mathrm{D} + \int_{\partial\Sigma_\mathrm{D}} \mathbf{M}_n(\mathbf{u}^*)\cdot(\boldsymbol{\theta} - \bar{\boldsymbol{\theta}})\,\mathrm{d}\partial\Sigma_\mathrm{D} \\
&\quad - \int_{\partial\Sigma_\mathrm{D}} Q_n(\mathbf{u}^*)(w - \bar{w})\,\mathrm{d}\partial\Sigma_\mathrm{D} = 0 \quad \forall w^*, \boldsymbol{\theta}^* \in H^1(\Sigma)
\end{aligned} \tag{15}$$

where the boundary bending moments $\mathbf{M}_n$ and shear forces Q_n are given by

$$\mathbf{M}_n(\mathbf{u}) = \mathbf{n}_\mathrm{b}\mathbf{M}(\mathbf{u}) = \begin{bmatrix} n_x & 0 & n_y \\ 0 & n_x & n_y \end{bmatrix} \begin{Bmatrix} M_{xx} \\ M_{yy} \\ M_{yx} \end{Bmatrix} = \begin{Bmatrix} M_{xx}n_x + M_{yx}n_y \\ M_{yx}n_x + M_{yy}n_y \end{Bmatrix} \tag{16}$$

$$Q_n(\mathbf{u}) = \mathbf{n}_\mathrm{s}\mathbf{Q}(\mathbf{u}) = \begin{bmatrix} n_x & n_y \end{bmatrix} \begin{Bmatrix} Q_x \\ Q_y \end{Bmatrix} = Q_x n_x + Q_y n_y \tag{17}$$

Therefore, we are considering the cases of hard simply supported and clamped boundary conditions, since we are restricting the normal segments rotations around the normals to the boundary $\partial\Sigma$.

2.4. Discretized equations

Using the standard finite element procedure on the variational form (15), one writes the following expansions in terms of the approximation functions, φ_i, and its derivatives (which can be, e.g. a finite-element-shape function or a cloud-shape function as defined in Section 3):

$$\begin{aligned}
&\boldsymbol{\theta}^* \cong \mathbf{N}_\theta\mathbf{U} \qquad \boldsymbol{\varepsilon}_\mathrm{b}(\mathbf{u}) \cong \mathbf{B}_\mathrm{b}\mathbf{U} \qquad \mathbf{M}(\mathbf{u}) \cong \mathbf{C}_\mathrm{b}\mathbf{B}_\mathrm{b}\mathbf{U} \\
&w \cong \mathbf{N}_w\mathbf{U} \qquad \boldsymbol{\varepsilon}_\mathrm{s}(\mathbf{u}) \cong \mathbf{B}_\mathrm{s}\mathbf{U} \qquad \mathbf{Q}(\mathbf{u}) \cong \mathbf{C}_\mathrm{s}\mathbf{B}_\mathrm{s}\mathbf{U}
\end{aligned} \tag{18}$$

where, N is the dimension of the approximation space,

$$\begin{aligned}
\mathbf{N}_\theta &= \begin{bmatrix} 0 & \varphi^1 & 0 & \cdots & \cdots & 0 & \varphi^N & 0 \\ 0 & 0 & \varphi^1 & \cdots & \cdots & 0 & 0 & \varphi^N \end{bmatrix} \\
\mathbf{N}_w &= \begin{bmatrix} \varphi^1 & 0 & 0 & \cdots & \cdots & \varphi^N & 0 & 0 \end{bmatrix} \\
\mathbf{U}^\mathrm{T} &= \begin{bmatrix} w^1 & \theta_x^1 & \theta_y^1 & \cdots & \cdots & w^n & \theta_x^N & \theta_y^N \end{bmatrix} \\
\mathbf{B}_\mathrm{b} &= \begin{bmatrix} 0 & -\varphi^1_{,x} & 0 & \cdots & \cdots & 0 & -\varphi^N_{,x} & 0 \\ 0 & 0 & -\varphi^1_{,y} & \cdots & \cdots & 0 & 0 & -\varphi^N_{,y} \\ 0 & -\varphi^1_{,x} & -\varphi^1_{,y} & \cdots & \cdots & 0 & -\varphi^N_{,x} & -\varphi^N_{,y} \end{bmatrix} \\
\mathbf{B}_\mathrm{s} &= \begin{bmatrix} \varphi^1_{,x} & -\varphi^1 & 0 & \cdots & \cdots & \varphi^N_{,x} & -\varphi^N & 0 \\ \varphi^1_{,y} & 0 & -\varphi^1 & \cdots & \cdots & \varphi^N_{,y} & 0 & -\varphi^N \end{bmatrix}
\end{aligned} \tag{19}$$

Using relations (18) and (19), the discretized stress resultants $\mathbf{M}_n$ and Q_n can be written in terms of the displacement parameters, $\mathbf{U}$, as

$$\begin{aligned} \mathbf{M}_n &\cong \mathbf{n}_\mathrm{b}(\mathbf{C}_\mathrm{b}\mathbf{B}_\mathrm{b}\mathbf{U}) \\ Q_n &\cong \mathbf{n}_\mathrm{s}(\mathbf{C}_\mathrm{s}\mathbf{B}_\mathrm{s}\mathbf{U}) \end{aligned} \tag{20}$$

In order to explicitly state the prescribed boundary displacement degrees of freedom, the approximation functions matrix is modified by including S_{dof} (dof $= w, \theta_x$ or θ_y). Therefore, $\mathbf{N}_\theta$ and $\mathbf{N}_w$ are replaced by $\tilde{\mathbf{N}}_\theta$ and $\tilde{\mathbf{N}}_w$ which are defined as

$$\tilde{\mathbf{N}}_\theta = \begin{bmatrix} 0 & S_{\theta x}\,\varphi_i & 0 \\ 0 & 0 & S_{\theta y}\,\varphi_i \end{bmatrix}$$

$$\tilde{\mathbf{N}}_w = [\, S_w\,\varphi_i \quad 0 \quad 0\,], \quad i = 1, \ldots, n$$

where

$$S_{\mathrm{dof}} = \begin{cases} 1 & \text{if dof is prescribed in } \partial\Sigma_\mathrm{D} \\ 0 & \text{if dof is not prescribed in } \partial\Sigma_\mathrm{D} \end{cases}$$

Therefore, the modified variational principle implies that

$$\begin{aligned} &\Bigg\{ \int_\Sigma (\mathbf{B}_\mathrm{b}^\mathrm{T}\mathbf{C}_\mathrm{b}\mathbf{B}_\mathrm{b} + \mathbf{B}_\mathrm{s}^\mathrm{T}\mathbf{C}_\mathrm{s}\mathbf{B}_\mathrm{s})\mathbf{U}\,\mathrm{d}\Sigma \\ &\quad + \int_{\partial\Sigma_\mathrm{D}} (\mathbf{B}_\mathrm{b}^\mathrm{T}\mathbf{C}_\mathrm{b}\mathbf{n}_\mathrm{b}^\mathrm{T}\tilde{\mathbf{N}}_\theta + \tilde{\mathbf{N}}_\theta^\mathrm{T}\mathbf{n}_\mathrm{b}\mathbf{C}_\mathrm{b}\mathbf{B}_\mathrm{b})\mathbf{U}\,\mathrm{d}\partial\Sigma \\ &\quad - \int_{\partial\Sigma_\mathrm{D}} (\mathbf{B}_\mathrm{s}^\mathrm{T}\mathbf{C}_\mathrm{s}\mathbf{n}_\mathrm{s}^\mathrm{T}\tilde{\mathbf{N}}_w + \tilde{\mathbf{N}}_w^\mathrm{T}\mathbf{n}_\mathrm{s}\mathbf{C}_\mathrm{s}\mathbf{B}_\mathrm{s})\mathbf{U}\,\mathrm{d}\partial\Sigma \\ &\quad - \int_\Sigma \mathbf{N}_w^\mathrm{T} q\,\mathrm{d}\Sigma - \int_\Sigma \mathbf{N}_\theta^\mathrm{T}\mathbf{m}\,\mathrm{d}\Sigma - \int_{\partial\Sigma_\mathrm{N}} \mathbf{N}_w^\mathrm{T}\bar{q}\,\mathrm{d}\partial\Sigma - \int_{\partial\Sigma_\mathrm{N}} \mathbf{N}_\theta^\mathrm{T}\bar{\mathbf{m}}\,\mathrm{d}\partial\Sigma \\ &\quad + \int_{\partial\Sigma_\mathrm{D}} (-\mathbf{B}_\mathrm{b}^\mathrm{T}\mathbf{C}_\mathrm{b}\mathbf{n}_\mathrm{b}^\mathrm{T}\tilde{\mathbf{N}}_\theta + \mathbf{B}_\mathrm{s}^\mathrm{T}\mathbf{C}_\mathrm{s}\mathbf{n}_\mathrm{s}^\mathrm{T}\tilde{\mathbf{N}}_w)\bar{\mathbf{U}}\,\mathrm{d}\partial\Sigma \Bigg\} \cdot \mathbf{U}^* = 0 \quad \forall \mathbf{U}^* \in \mathbb{R}^n \end{aligned} \tag{21}$$

The arbitrariness of the vector $\mathbf{U}^*$, leads to the following stiffness matrix and force vector, respectively:

$$\begin{aligned} \mathbf{K} = &\int_\Sigma (\mathbf{B}_\mathrm{b}^\mathrm{T}\mathbf{C}_\mathrm{b}\,\mathbf{B}_\mathrm{b} + \mathbf{B}_\mathrm{s}^\mathrm{T}\mathbf{C}_\mathrm{s}\,\mathbf{B}_\mathrm{s})\,\mathrm{d}\Sigma \\ &+ \int_{\partial\Sigma_\mathrm{D}} (\mathbf{B}_\mathrm{b}^\mathrm{T}\mathbf{C}_\mathrm{b}\,\mathbf{n}_\mathrm{b}^\mathrm{T}\,\tilde{\mathbf{N}}_\theta + \tilde{\mathbf{N}}_\theta^\mathrm{T}\,\mathbf{n}_\mathrm{b}\mathbf{C}_\mathrm{b}\,\mathbf{B}_\mathrm{b})\,\mathrm{d}\partial\Sigma \\ &- \int_{\partial\Sigma_\mathrm{D}} (\mathbf{B}_\mathrm{s}^\mathrm{T}\,\mathbf{C}_\mathrm{s}\mathbf{n}_\mathrm{s}^\mathrm{T}\tilde{\mathbf{N}}_w + \tilde{\mathbf{N}}_w^\mathrm{T}\,\mathbf{n}_\mathrm{s}\mathbf{C}_\mathrm{s}\mathbf{B}_\mathrm{s})\,\mathrm{d}\partial\Sigma \end{aligned} \tag{22}$$

$$\begin{aligned} \mathbf{F} = &\int_\Sigma \mathbf{N}_w^\mathrm{T} q\,\mathrm{d}\Sigma + \int_\Sigma \mathbf{N}_\theta^\mathrm{T}\mathbf{m}\,\mathrm{d}\Sigma + \int_{\partial\Sigma_N} \mathbf{N}_w^\mathrm{T}\,\bar{q}\,\mathrm{d}\partial\Sigma \\ &+ \int_{\partial\Sigma_\mathrm{N}} \mathbf{N}_\theta^\mathrm{T}\bar{\mathbf{m}}\,\mathrm{d}\partial\Sigma + \int_{\partial\Sigma_\mathrm{D}} (\mathbf{B}_\mathrm{b}^\mathrm{T}\mathbf{C}_\mathrm{b}\mathbf{n}_\mathrm{b}^\mathrm{T}\tilde{\mathbf{N}}_\theta - \mathbf{B}_\mathrm{s}^\mathrm{T}\mathbf{C}_\mathrm{s}\,\mathbf{n}_\mathrm{s}^\mathrm{T}\tilde{\mathbf{N}}_w)\bar{\mathbf{U}}\,\mathrm{d}\partial\Sigma \end{aligned} \tag{23}$$

and to the algebraic system of linear equations

$$\mathbf{KU} = \mathbf{F} \tag{24}$$

3. *hp*-CLOUD APPROXIMATION FUNCTIONS

The essential feature of the *hp*-Cloud method lies in the way the approximation functions are built in order to trivially implement the *p*-enrichment. In order to accomplish this task, one has to define an open covering of the domain Σ and an associated partition of unity. In this section we review the main concepts which are detailed in, e.g. References [9, 18].

Let Σ be an open bounded domain in $\mathbb{R}^n$, $n = 1, 2, 3$ and let Q_N be an arbitrary set of N points (*nodes*) $\mathbf{x}_\alpha \in \bar{\Sigma}$, that is

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

We then associate with each node $\mathbf{x}_\alpha$ an open set ω_α, called a *cloud*. Clouds ω_α, $\alpha = 1, \ldots, N$, are chosen such that they form a finite open covering, $\Im_N = \{\omega_\alpha\}_{\alpha=1}^N$ of Σ, i.e.

$$\bar{\Sigma} \subset \bigcup_{\alpha=1}^{N} \bar{w}_\alpha \tag{25}$$

The basic building block of an *hp* cloud approximation is a partition of unity subordinated to the open covering $\Im_N$. A set of functions $\mathscr{L}_N = \{\psi_\alpha\}_{\alpha=1}^N$ is called partition of unity (POU) subordinated to the open covering $\Im_N$ if the following holds:

$$\begin{aligned} &\sum_{\alpha=1}^{N} \psi_\alpha(\mathbf{x}) = 1 \quad \forall \mathbf{x} \in \Sigma \\ &\psi_\alpha \in C_0^s(\omega_\alpha), \quad s \geqslant 0, \ \ \alpha = 1, \ \ldots, N \end{aligned} \tag{26}$$

The last property implies that the functions ψ_α, $\alpha = 1, \ldots, N$ are non-zero only over the open sets ω_α (the clouds). Examples of partitions of unity are Lagrangian finite-element-shape functions, Shepard functions [10] and MLS functions [3]. In the case of finite element partition of unity, a node $\mathbf{x}_\alpha$ is a nodal point of the finite element mesh and a cloud ω_α is the union of all finite elements connected to the node $\mathbf{x}_\alpha$ [19]. In this study, we use Shepard POU because they can be built without partitioning the domain, i.e. they constitute a meshless partition of unity. In addition, they can be build quite inexpensively as compared, for example, with MLS functions—another example of a meshless POU. The Shepard functions can also be build with any degree of regularity. Although for the case of Mindlin's equations the Galerkin method requires only C^0 functions, in the case of, e.g. the biharmonic equation, C^1 continuity is required. The construction of Shepard functions is summarized in Section 3.2.

3.1. The family of hp cloud functions $\mathscr{F}_N^p$

Let $\{L_i\}_{i \in \mathscr{I}}$ denote a set of functions which can approximate well, in an appropriate norm, the solution u of a boundary value problem, i.e. there exists u_i, $i \in \mathscr{I}$ such that

$$\|u_{hp} - u\| < \varepsilon$$

where $u_{hp} = \sum_{i \in \mathscr{I}} u_i L_i$ and $\mathscr{I}$ denotes an index set.

Now consider the following set of *cloud-shape functions*, defined as

$$\phi_i^\alpha := \psi_\alpha L_i, \quad \alpha = 1, \ldots, N, \ \ i \in \mathscr{I}$$

where ψ_α is a partition of unity function. Then it is not difficult to show that linear combinations of these cloud-shape functions can also approximate well the function u

$$\begin{aligned} \sum_\alpha \sum_i u_i \phi_i^\alpha &= \sum_\alpha \sum_i u_i \psi_\alpha L_i = \sum_\alpha \psi_\alpha \sum_i u_i L_i \\ &= \sum_\alpha \psi_\alpha u_{hp} = u_{hp} \sum_\alpha \psi_\alpha = u_{hp} \end{aligned} \tag{27}$$

Note that:

(i) The cloud-shape functions ϕ_i^α, $\alpha = 1, \ldots, N$, $i \in \mathscr{I}$, are non-zero only over the cloud ω_α.
(ii) The functions L_i, $i \in \mathscr{I}$, can be chosen with great freedom. The most straighforward choice, and the one we use in this work, is polynomial functions since they can approximate smooth functions well. However, for many classes of problems, including the case of Mindlin's plate problem, there are better choices. Examples are harmonic polynomials for the solution of Poisson's equation [20], Muskhelishvili functions [21] for the solution of plane elasticity problems, etc.
(iii) The h refinement of an hp cloud discretization consists of adding more clouds of smaller size to the covering of the domain while keeping the degree of the cloud-shape functions fixed. In the case of p enrichment, the number of clouds is kept fixed while the polynomial degree of the functions L_i used in the construction of the cloud-shape functions is increased.
(iv) The cloud-shape functions do *not* have the Kronecker-delta property

$$\phi_i^\alpha(\mathbf{x}_\beta) = \delta_\alpha^\beta$$

Hence, the Dirichlet boundary conditions cannot be imposed by directly specifying the cloud parameters as it is generally done in the FEM. One of the alternatives is to apply the Dirichlet boundary conditions in a weak form through Lagrange multipliers. In this case, these multipliers can be identified with the reactions and therefore can be expressed in terms of the displacements, resulting in the modification of the variational form (17). Although these variational forms are similar, the latter leads to a lower convergence rate because of the lower rate of convergence of the stresses as compared to the displacements.

In this work we choose the minimal set of complete polynomials of degree less or equal to p in a two-dimensional manifold for the functions L_i, i.e.

$$L_i = x^l y^m, \quad 0 \leqslant l, m \leqslant p, \quad l + m \leqslant p \tag{28}$$

We then define the family $\mathscr{F}_N^p$ of cloud-shape functions of degree p by

$$\mathscr{F}_N^p = \{\phi_i^\alpha = \psi_\alpha L_i, \ \alpha = 1, \ldots, N, \ i \in \mathscr{I}\} \tag{29}$$

where the functions ψ_α, $\alpha = 1, \ldots, N$, form a partition of unity. Note that so far we have not chosen a particular partition of unity. Any set of functions fulfilling properties (26) is valid. Also, note that in the above definition we use the same set of functions $\{L_i\}_{i \in \mathscr{I}}$ for all nodes $\alpha = 1, \ldots, N$. This constrain is not necessary, since each node can have a different polynomial order, regardless of the polynomial order of neighbouring nodes.

From (28) and the definition of $\mathscr{F}_N^p$ we have that (the proof follows as in (27))

$$\mathscr{P}_p \subset \text{span}\{\mathscr{F}_N^p\}$$

where $\mathscr{P}_p$ denotes the set of complete polynomials of degree less or equal to p.

3.2. Construction of Shepard partitions of unity

In this section we describe the construction of the partition of unity we use in our computations, i.e. Shepard functions.

Let $\mathscr{W}_\alpha : \mathbb{R}^n \to \mathbb{R}$ denote a *weighting function* with compact support ω_α that belongs to the space $C_0^s(\omega_\alpha)$, $s \geqslant 0$ and suppose that

$$\mathscr{W}_\alpha(\mathbf{x}) \geqslant 0 \quad \forall \mathbf{x} \in \Omega$$

We use clouds $\omega_\alpha,\ \alpha = 1, \ldots, N$ defined by

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

which are circles with radius h_α in two dimensions. In this case, the weighting functions $\mathscr{W}_\alpha$ can be implemented with any degree of regularity using 'ridge' functions. More specifically, the weighting functions $\mathscr{W}_\alpha$ can be implemented through the composition

$$\mathscr{W}_\alpha(\mathbf{x}) := g(r_\alpha)$$

where g is, e.g. a B-spline with compact support $[-1, 1]$ and r_α is the functional

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

In the computations presented in Section 5, g is the quartic $C^3([-1, 1])$ B-spline shown in Figure 2. Details on the construction of the B-splines can be found in, e.g. [22].

The partition of unity functions ψ_α can then be defined by

$$\psi_\alpha(\mathbf{x}) = \frac{\mathscr{W}_\alpha(\mathbf{x})}{\sum_\beta \mathscr{W}_\beta(\mathbf{x})} \quad \beta \in \{\gamma : \mathscr{W}_\gamma(\mathbf{x}) \neq 0\} \tag{30}$$

which are known as Shepard functions [10]. The main advantages of this particular partition of unity are

(i) low computational cost and simplicity of computation,
(ii) it is meshless—there is no need to partition the domain to build this partition of unity,
(iii) it can easily be implemented in any dimension,
(iv) it can be constructed with any degree of regularity
(v) it allows easy implementation of h adaptivity.

Figure 3 shows the Shepard function ψ_α associated with a cloud ω_α centred at the origin of the domain $(-1, 1) \times (-1, 1)$. It is a $C^3(\omega_\alpha)$ function built using the B-spline depicted in Figure 2.

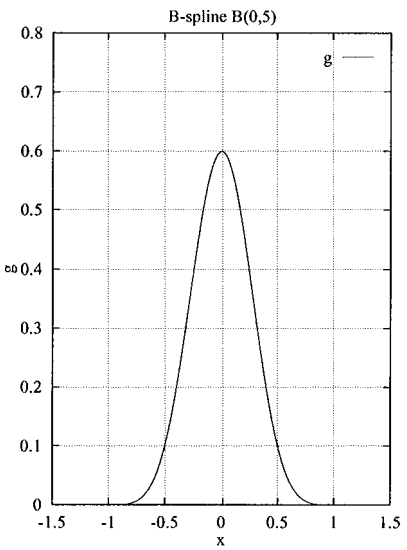

Figure 2. B-spline used in the construction of weighting functions.

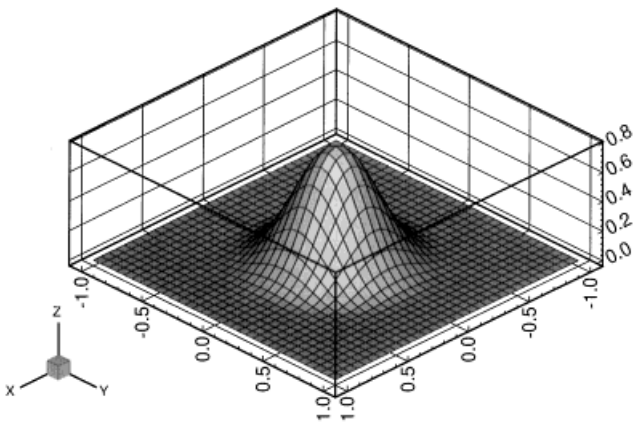

Figure 3. Shepard partition of unity function.

4. OBJECT-ORIENTED IMPLEMENTATION

4.1. Object-oriented programming

The purpose of implementing a code following an object-oriented programming has the objective of reaching the theoretical advantages that this paradigm provides

(i) modularized code,
(ii) possibility of organizing the data in analogous structures as the ones reached by mathematical reasoning,

(iii) data protection,
(iv) portability and code sharing with other programmers.

The edge is that OOP forces the programmer to organize his code in such a way the former principles must be met. In other words, a good OOP needs a reasonable time dedicated to the design of the code before its implementation. This is a healthy attitude, mainly when the intention is to construct a software structure dynamic enough to support future development and improvements.

This paradigm (OOP) is based on the definition of work cells, called Classes, linking data and functions completely. These functions are operators that can work with inner variables of the classes and/or external ones coming by parameters when the function is called.

Many applications have been developed following these guidelines, based on a group of class libraries called ACDPOOP [23, 24] (Computational Environment for Code Development based on OOP). A description of the code ACCLOUDS++ is shown next, focusing on the main aspects of the data structure.

4.2. Code ACCLOUDS++

As pointed, the OOP paradigm is based on the definition of classes. The code under development is oriented towards the simulation of modified variational principles for plane elasticity and plates by making use of the new approximation functions shown in the previous section. However, much care is taken in order for not restricting the development only to these type of problems. Co-ordinates, materials, geometrical properties, degrees of freedom, main procedures of integration and flux control are coded as generally as possible in order to gain flexibility for updating and implementing new procedures.

The main classes are named as follows:

Classes for geometric definition and discretization information:

(i) acKeyPoint,
(ii) acBoundaryGrp,
(iii) acBoundary,
(iv) acClouds.

Classes dealing with constitutive relations and geometric constants:

(i) acMaterial,
(ii) acGeomProperties.

Classes dealing with degrees of freedom definitions and boundary conditions:

(i) acBoundCondGrp.

Classes dealing with domain properties:

(i) acDomain.

Classes dealing with integration point by specific problems:

(i) acIP,
(ii) acIPPlate, acIPPlStrs, acIPPlStrn, etc.

The class **acKeyPoint** (derived of a class named **acCoord2D**) deals with reading, writing and storing information of nodes used to define the domain geometry. The class **acBoundaryGrp** keeps

the information related to boundary definitions and discretization as well as the evaluation of the boundary condition quantities. It has, as a main variable, a vector of objects of class **acBoundary**, where each of its components is related to a specific contour with its particular properties: acLine, acArch, acBSpline, and so on. All of them are inherited from the generic class acBoundary.

Using the so-called *virtual functions*, it is possible to build the main procedures of the software with almost total independence from the particular geometric characteristics of the boundary. Procedures for reading, discretization, etc., are determined by main code using the same function name. In replying to this order, each contour will take a different attitude depending on the specific computations it needs to perform in order to meet the request.

The class **acClouds** has functions which define domain covering, as well as the search procedures for identifying which clouds cover an arbitrary point. In order to perform this search, a quad-tree management system, Fancello [23], is introduced for allowing a fast identification of which clouds cover each integration point. In addition, acClouds also computes all the associated approximation functions and its derivatives.

The class **acBoundCondGrp** takes care of the boundary conditions where the essential boundary conditions and their values are prescribed.

The class **acIP** is a generic class of integration points from which the classes acIPPlate, acIPPlsStrs, and others are inherited. These classes are responsible for the evaluation of the stiffness matrix and loads terms associated with an integration point of a determined mathematical model like: plate, elastic plane strain, and so on.

The class **acDomain** contains the functions which deal with the domain information, superpositions on the global stiffness matrix and on the domain load vector.

5. NUMERICAL RESULTS

Next, we discuss the first numerical results regarding the following aspects:

(i) locking;
(ii) convergence analysis in L_2 norm;
(iii) maximum displacement convergence.

5.1. Locking behaviour

The locking occurs when the approximation functions are unable to meet the requirement for allowing null transversal shear deformations as the plate becomes thin. Then, the shear deformation energy is overestimated and so is the stiffness matrix. For this investigation, we look at two problems where the square plates are uniformly loaded on its domain. The first one consists of a simply supported plate and the second of a clamped one under a uniform distributed load. In both examples, just one-fourth of the plate is modelled by using symmetry conditions and is covered by 25 clouds as shown in Figure 4, and complete polynomials of degrees 3 and 4 are used for *p*-enrichment.

5.1.1. Simply supported square plate. The transversal displacement at the centre of the plate is normalized with respect to the Kirchhoff–Love thin plate solution, which in this case is given by

$$w_{\text{central}} = 0.004062 \frac{qL^4}{D} \tag{31}$$

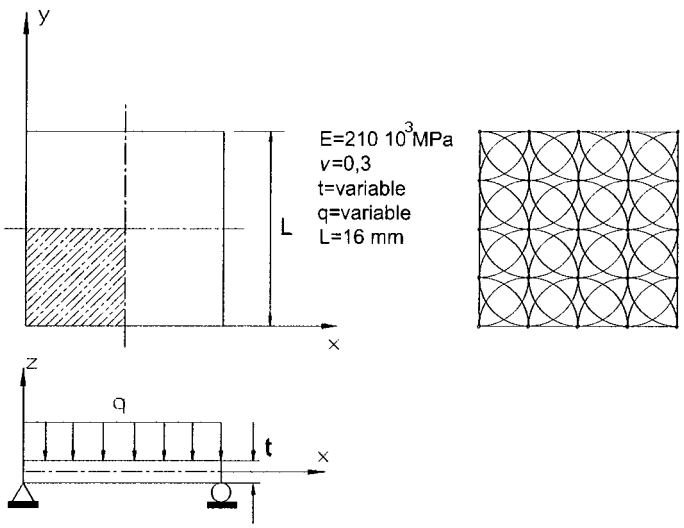

Figure 4. Plate model and covering with 25 clouds.

where D is the bending stiffness

$$D = \frac{Et^3}{12(1-v^2)} \tag{32}$$

The results show that for a relation $L/t \geqslant 10^2$ the solution converges to the thin plate one and the locking is present for cubic polynomial enrichment and for $L/t > 10^4$. When the approximation functions are further enriched the locking is unnoticeable as verified in Figure 5(a).

5.1.2. Clamped square plate. Again, the results for cubic and quartic polynomial enrichment are displayed against the thin plate model solution which now reads

$$w_{\text{central}} = 0.00126\frac{qL^4}{D} \tag{33}$$

One may verify from Figure 5(b) that the results for $L/t \geqslant 10^2$ approach the thin plate solution and that the locking appears when $L/t > 10^3$ and a cubic enrichment is used. If the enrichment is of fourth order the locking is not significant. The oscillations, which occur for very thin plates, are due to poor stiffness matrices conditioning.

In all the above computations, the same cloud-shape functions are used to approximate the transversal displacement w and rotations θ_x and θ_y. The shape functions are built from the product of a Shepard partition of unity and monomials L_i of degree three or four as defined in (29). The above experiments show that this choice of cloud functions behaves like finite-element-shape functions of degree $p = 3$ or 4 with respect to shear locking. Namely, the locking can be overcome by using sufficiently high p approximations. Other choices for the functions L_i could also be used. Donning and Liu [15] have recently shown that cardinal splines are very effective to solve Mindlin's plate problem.

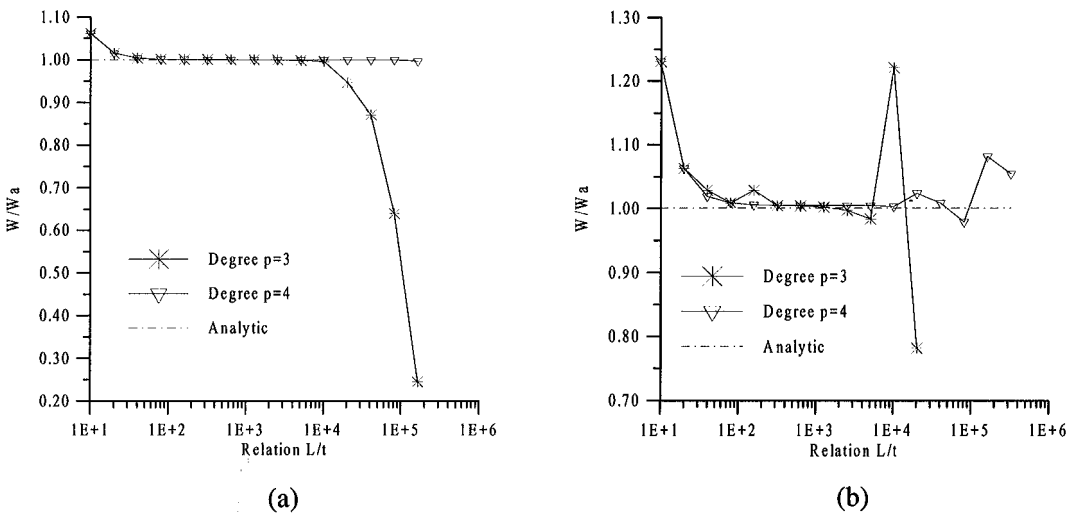

Figure 5. Normalized central displacement: (a) simply supported plate; and (b) clamped plate.

5.2. *Convergence results*

5.2.1. Relative transversal displacement error in L_2 norm. In the following, we present results for the relative error measured in the L_2 norm when compared to analytic Mindlin plate model solutions. The uniform transversal load is $q_0 = 0.1\,\mathrm{N/mm^2}$, the plate has all its edges simply supported, and it is $t = 0.1$ mm thick and 16 mm wide. All these solutions were obtained for homogeneous p-enrichment and the coverings were as follows:

(i) 5 clouds and $0 \leqslant p \leqslant 4$;
(ii) 25 clouds and $0 \leqslant p \leqslant 4$;
(iii) 87 clouds and $0 \leqslant p \leqslant 2$.

The analytic solution is obtained by using series expansions according to Marguerre and Woernie [16]:

$$w(x, y) = \sum_{n=1}^{10} \bar{w}(x) \sin(y) \tag{34}$$

where

$$\bar{w}(x) = \frac{q_n}{D}\left(\frac{b}{n\pi}\right)\left\{1 + \hat{h}^2 - \frac{1}{\cosh\,\hat{a} + 1}\left[\begin{array}{c}(1 + \hat{h}^2)(\cosh\,\hat{x} + \cosh(\hat{a} - \hat{x})) \\ +\tfrac{1}{2}(\hat{x}\,\sinh(\hat{a} - \hat{x}) + (\hat{a} - \hat{x})\sinh(\hat{x}))\end{array}\right]\right\}$$
$$\text{for } n = 1, \ldots, 10 \tag{35}$$

In the above expression, a and b are the plate dimensions along the x and y directions, respectively, and the following notations are adopted for brevity:

$$\hat{a} = \frac{n\pi}{2}a, \quad \hat{x} = \frac{n\pi}{b}x, \quad \hat{h}^2 = \left(\frac{n\pi}{2}\right)^2 h^2$$
$$h = \sqrt{\frac{D}{Gt_s}}, \quad G = \frac{E}{2(1+\nu)}, \quad t_s = \frac{t}{1.2}, \quad q_n = \frac{4q_0}{n\pi}$$

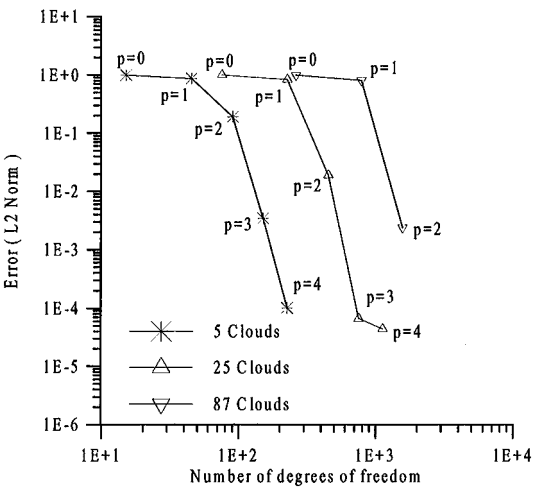

Figure 6. Relative displacement in the L_2 norm.

The relative error, E, measured in the L_2 norm is defined as

$$E = \frac{\|w - w_h\|_{L_2}}{\|w\|_{L_2}} = \frac{\sqrt{\sum_{n=1}^{N_{\text{ip}}} \int_{\Sigma} (w_n - w_n^h)(w_n - w_n^h)\,\mathrm{d}\Sigma}}{\sqrt{\sum_{n=1}^{N_{\text{ip}}} \int_{\Sigma} (w_n)(w_n)\,\mathrm{d}\Sigma}} \tag{36}$$

In this expression, w_n and w_n^h stand for the analytical and numerical solutions for the transversal displacement at each one of the N_{ip} integration points.

The results depicted in Figure 6 show that the errors decay faster with p-enrichment than with the increase in the number of clouds. This is expected since the solution is smooth and the Shepard functions always lead to higher solution errors. On the other hand, one can notice the fast error decay when the p-enrichment is applied. Note in the second example that the relative error decays from 0.019075 to 0.000067 when the polynomial order is increased from 2 to 3.

5.2.2. Transversal displacement convergence at the plate centre. For this convergence analysis we consider the plate sketched in Figure 4 under a uniformly distributed load $q_0 = 0.1\,\text{N/mm}^2$ and it is also 0.1 mm thick. The centre point transversal displacement is normalized with respect to the Mindlin's plate solution and the results against the number of degrees of freedom are shown in Figure 7.

Once again, one can note that the p-enrichment convergence rate is much higher than that for h-refinement. In addition, good results can be obtained by using few clouds as in the first example and excellent accuracy can be reached as in the second example. The oscillation verified in Figure 7(b) is believed to be due to the use of the modified variational principle in which the Dirichlet boundary conditions are not strongly imposed and, therefore, one loses the monotonic convergence.

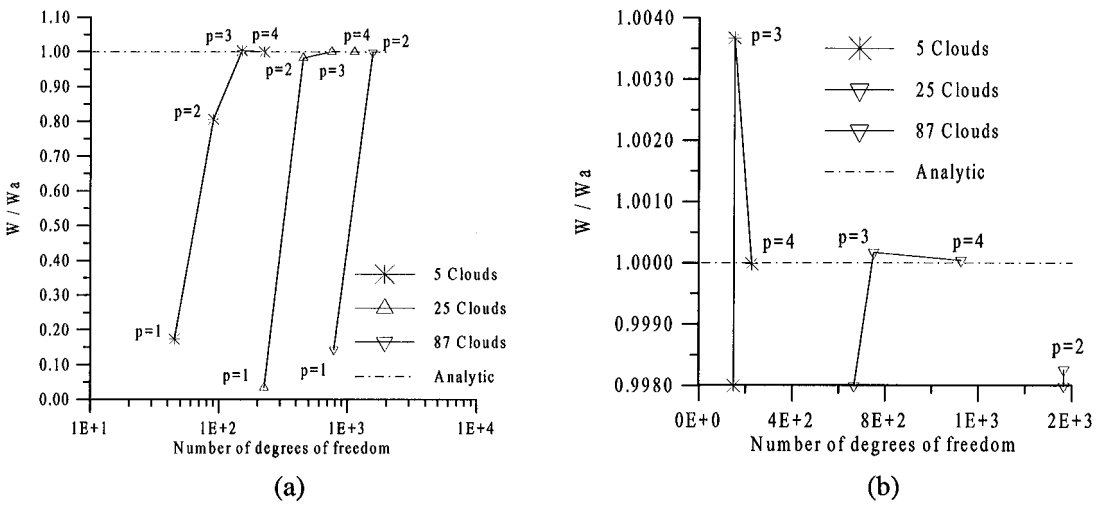

Figure 7. Normalized central displacement for a simply supported plate.

5.2.3. Bending moment relative error in L_2 norm. For looking at the bending moment relative error we consider the same plate and loading as before and with the same discretization. Again, the reference is the Mindlin's plate analytical solution found by series expansion Marguerre and Woernie [16].

The moments are denoted by M_{xx}, M_{yy} and M_{xy} and related to the displacement field according to

$$M_{xx} = -D\{w'' + \nu w^{\cdot\cdot} - h^2[\Phi'' + \nu\Phi^{\cdot\cdot}]\} \tag{37}$$

$$M_{yy} = -D\{w^{\cdot\cdot} + \nu w'' - h^2[\Phi^{\cdot\cdot} + \nu\Phi'']\} \tag{38}$$

$$M_{xy} = -D(1-\nu)(w'^{\cdot} - h^2\Phi'^{\cdot}) \tag{39}$$

$$\tag{40}$$

where: $(\cdot)'' = \partial^2(\cdot)/\partial x^2$, $(\cdot)^{\cdot\cdot} = \partial^2(\cdot)/\partial y^2$, $(\cdot)'^{\cdot} = \partial^2(\cdot)/\partial x y$ and

$$\Phi(x,y) = \frac{4q_0 b^2}{D\pi^3} \sum_{n=1,3,5\ldots}^{10} \frac{1}{n^3} \left[\left(\frac{\cosh\,\hat{a} - 1}{\sinh\,\hat{a}} \right) \sinh(\hat{x}) - \cosh(\hat{x}) \right] \sin\left(\frac{n\pi y}{b}\right) \tag{41}$$

The relative moment error measured in L_2 norm is defined as

$$E_{\mathbf{m}} = \frac{\|\mathbf{m} - \mathbf{m}_h\|_{L_2}}{\|\mathbf{m}\|_{L_2}} = \frac{\sqrt{\sum_{n=1}^{N_{\text{ip}}} \int_\Sigma (\mathbf{m}_n - \mathbf{m}_n^h)(\mathbf{m}_n - \mathbf{m}_n^h)\,\mathrm{d}\Sigma}}{\sqrt{\sum_{n=1}^{N_{\text{ip}}} \int_\Sigma (\mathbf{m}_n)(\mathbf{m}_n)\,\mathrm{d}\Sigma}} \tag{42}$$

The symbols $\mathbf{m}_n$ and $\mathbf{m}_n^h$ denote the moments obtained through the analytical solution and by the *hp*-clouds, respectively, and they are evaluated at each integration point on the domain. In Figure 8(a) $E_{\mathbf{m}}$ is shown in terms of the number of degrees of freedom.

One may notice, in Figure 8(a), the influence of the *h*-refinement on the pre-asymptotic convergence rates of the relative moment error throughout the plate. Moreover, one may also visualize the

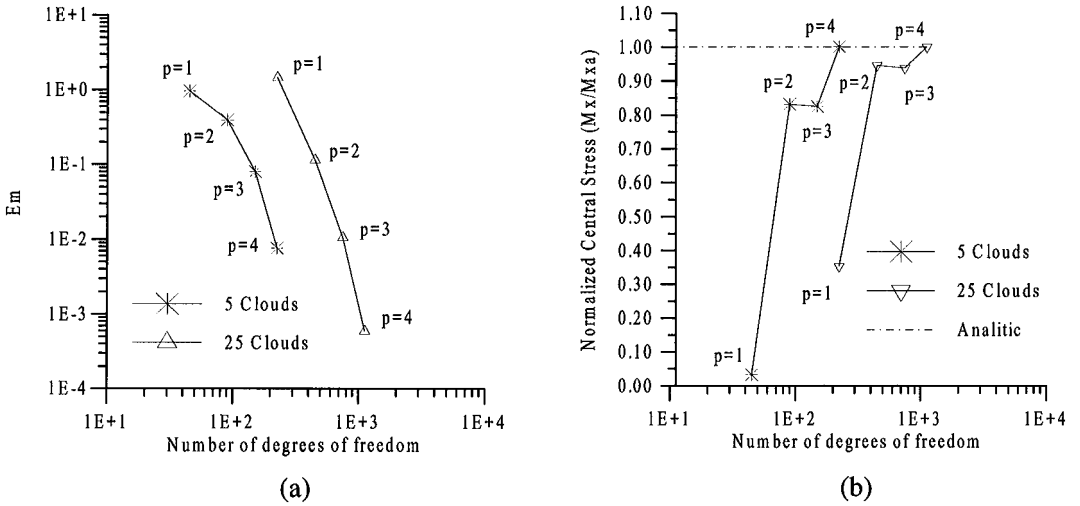

Figure 8. Convergence of moment measured in the L_2 norm and of normalized bending moment at centre of the plate: (a) relative moment error—$E_{\mathbf{m}}$; and (b) normalized bending moment at the centre point.

h-convergence for the same *p*-enrichment. This rate of convergence increases with the polynomial order as expected, but when using Shepard functions the results seems to diverge. In Figure 8(b) one may verify the rapid convergence of the bending moment at the plate centre. In addition, since the solution at the centre point associated cloud is an even function there is no improvement when odd functions are used in the enrichment and it even slightly deteriorates due to the increase in the condition number.

5.3. Integration

Although it has been claimed that the meshless methods, as the *hp*-cloud, can lead to higher convergence rates than the usual FEM (when, e.g. customized approximations are used), several aspects are still unclear. One of them is the best choice of integration scheme. To this end, one must once more consider the plate sketched in Figure 4 under the same loading as before and covered by 25 clouds. Since both the solution and the approximation functions are regular, this study tries to correlate the polynomial order with the number of integration points in order to obtain better convergence, or avoid under-integration. An underlying triangular mesh was used and, for each triangle, sets of $12, 16, 25, 79$ integration points were implemented. For integrating the boundary terms, each edge was subdivided in $20, 40, 60$ segments and three integration points are placed over each segment. The results for homogeneous polynomial enrichments of orders 3 and 4 are shown in Figure 9 which shows the relation of the number of integration points with respect to the normalized plate centre transversal displacement. From the results for the plate centre transversal displacement we can verify that the 12-point rule leads to poor results due to under-integration. Better results are obtained with 16 and 25 integration points, but to obtain the full quality the 79-point scheme is needed with this underlying mesh together with 180 points on the boundary. Therefore, in order to increase the efficiency we still need to look for more efficient integration schemes, probably some adaptive ones. Another, simpler alternative would be to use the partition

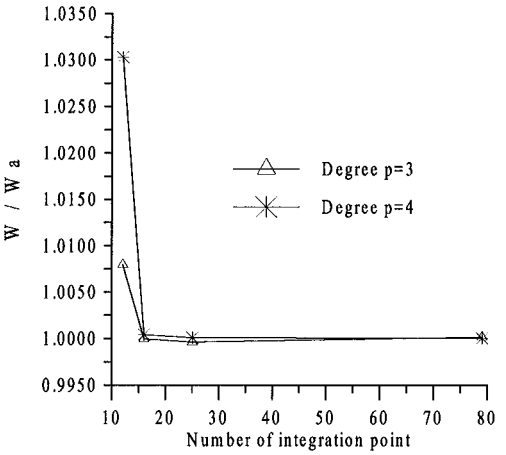

Figure 9. Convergence of the central displacement with the number of integration points.

of unity furnished by the conventional FEM and to *p*-enrich along the *hp*-cloud guidelines [19]. This would simplify the integration and eliminate the quad-tree procedure for identifying each integration point coverage.

6. CONCLUSIONS

An extension of the *hp*-cloud method for solving Mindlin's plate model problems is presented together with a brief review of this procedure. The implementation of the code ACCLOUDS++ based on the whole concepts of the *hp*-cloud method has been sketched, although most of the technical details which deserve attention have been omitted for brevity, but this has allowed an ordered and progressive code implementation. The first results illustrate the high convergence rates as usual in the *hp*-cloud method. The easy implementation of approximation functions with higher continuity together with a simple *p*-enrichment makes this method a candidate for investigating higher-order plate models, boundary layers and hierarchic plate models. However, the increase in the condition number with *p*-refinement is still of concern and adequate pre-conditioners ought to be devised.

ACKNOWLEDGEMENTS

Authors Garcia, Fancello and Barcellos gratefully acknowledge the CNPq (Brazil) for the support of this work through the grants 520093/96-8 and 523564/96-1

REFERENCES

1. Gingold RA, Monaghan JJ. Kernel estimates as a basis for general particle methods in hydrodynamics. *Journal of Computational Physics* 1982; **46**:429–453.
2. Nayroles B, Touzot G, Villon P. Generalizing the finite element method: diffuse approximation and diffuse elements. *Computational Mechanics* 1992; **10**:307–318.

3. Lancaster P, Salkauskas K. Surfaces generated by moving least squares methods. *Mathematics of Computation* 1981; **37**(155):141–158.
4. Belytschko T, Lu YY, Gu L. Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering* 1994; **37**:229–256.
5. Liu WK, Jun S, Zhang YF. Reproducing kernel particle methods. *International Journal for Numerical Methods in Engineering* 1995; **20**:1081–1106.
6. Belytschko T, Krongauz Y, Organ D, Fleming M. Meshless methods: an overview and recent developments. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**:3–47.
7. Liu WK, Chen Y, Jun S, Chen JS, Belytschko T, Pan C, Uras RA, Chang CT. Overview and applications of the reproducing kernel particle methods. *Archives of Computational Methods in Engineering. State of Art Reviews* 1996; **3**:3–80.
8. Duarte CAM, Oden JT. Hp clouds-a meshless method to solve boundary-value problems. *Technical Report 95-05*, TICAM, The University of Texas at Austin, May 1995.
9. Duarte CAM, Oden JT. *Hp* clouds—an *hp* meshless method. *Numerical Methods for Partial Differential Equations* 1996; **12**:673–705.
10. Shepard D, A two-dimensional function for irregularly spaced data. *ACM National Conference*, 1968; 517–524.
11. Lu YY, Belytschko T, Gu L. A new implementation of the element free Galerkin method. *Computer Methods in Applied Mechanics and Engineering* 1994; **113**:397–414.
12. Babuska I, Melenk JM. The partition of unity finite element method. *Technical Report BN-1185*, Institute for Physics Science and Technology, University of Maryland, June 1995.
13. Babuška I, Melenk JM. The partition of unity finite element method. *International Journal for Numerical Methods in Engineering*1997; **40**:727–758.
14. Armando Duarte C. The hp Cloud Method. *Ph.D. Thesis*, The University of Texas at Austin, Austin, TX, USA, December 1996.
15. Liu WK, Donning B. Meshless methods for shear-deformable beams and plates. *Computer Methods in Applied Mechanics and Engineering* 1998; **152**:47–72.
16. Marguerre K, Woernie H. *Elastic Plates*. Blaisdell: New York, 1969.
17. Washizu K, *Variational Methods in Elasticity and Plasticity*. Pergamon Press: Oxford, 1974.
18. Duarte CAM, Oden JT. An *hp* adaptive method using clouds. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**:237–262.
19. Oden JT, Duarte CA, Zienkiewicz OC. A new cloud-based *hp* finite element method. *Computer Methods in Applied Mechanics and Engineering* 1998; **153**:117–126.
20. Melenk JM. Finite element methods with harmonic shape functions for solving laplace's equation. *Master's Thesis*, The University of Maryland, 1992.
21. Muskhelishvili NI. *Some Basic Problems of the Mathematical Theory of Elasticity*. P. Noordhoff, Groningen, 1963.
22. Carl deBoor. *A Practical Guide to splines*. Springer: New York, 1978.
23. Fancello EA, Guimarães AS, Feijoó RA, Venere M. Geração automática de malhas em programação orientada a objetos. In *XI Congresso Brasileiro de Engenharia Mecânica*, São Paulo, Brazil, 1991.
24. Theoretical and computational solid mechanics group: http://www.lncc.br/tacsom.html.

