Comput Mech (2007) 40:949–963
DOI 10.1007/s00466-006-0154-6

ORIGINAL PAPER

An improved finite point method for tridimensional potential flows

Enrique Ortega · Eugenio Oñate · Sergio Idelsohn

Received: 18 July 2006 / Accepted: 6 December 2006 / Published online: 6 March 2007

Abstract At the local level, successful meshless techniques such as the Finite Point Method must have two main characteristics: a suitable geometrical support and a robust numerical approximation built on the former. In this article we develop the second condition and present an alternative procedure to obtain shape functions and their derivatives from a given cloud of points regardless of its geometrical features. This procedure, based on a QR factorization and an iterative adjust of local approximation parameters, allows obtaining a satisfactory minimization problem solution, even in the most difficult cases where usual approaches fail. It is known that high-order meshless constructions need to include a large number of points in the local support zone and this fact turns the approximation more dependent on the size, shape and spatial distribution of the local cloud of points. The proposed procedure also facilitates the construction of high-order approximations on generic geometries reducing their dependence on the geometrical support where they are based. Apart from the alternative solution to the minimization problem, the behaviour of high-order Finite Point approximations and the overall performance of the proposed methodology are shown by means of several numerical tests.

Sergio Idelsohn is ICREA Research Professor at CIMNE.

Enrique Ortega · Eugenio Oñate (✉) · Sergio Idelsohn
International Center for Numerical Methods in Engineering (CIMNE) Universidad Politécnica de Cataluña UPC Campus Nord, edificio C1, Gran Capitán, s/n, 08034 Barcelona, España
e-mail: onate@cimne.upc.edu

1 Introduction

The Finite Point Method (FPM) presented by Oñate et al. [1,2] is a conceptually simple discretization technique which has shown great capacity to solve convective-diffusive problems, incompressible and compressible fluid flow problems [3–7] and solid mechanics problems [8] with good accuracy. In all of these works, the approximation procedure and its behaviour is explored from different points of view. Recurrently, topics such as local distribution of points, weighting function effects and high-order approximations are reviewed and their valuable contributions give the FPM a more solid base.

The main objective of this work is to present an alternative approach to calculate the local approximation in the FPM. This procedure plays an ad-hoc role only when the usual procedure fails and allows getting a satisfactory approximation without the necessity of modifying the local geometrical support. Furthermore, some important features of the Finite Point (FP) approximation methodology are reviewed and the essential issues highlighted. The second objective is to explore practical aspects of high-order approximations with the aim of identifying capabilities of the FP methodology which allow implementing high-order solutions efficiently.

2 Approximation in the finite point method

An approximation to an unknown function $u(x)$ defined in a closed domain $\Omega \epsilon \Re^d$ ($d = 1, 2$ or 3) which is discretized by a set of points x_i, $i = 1, n$ is developed. In order to obtain a "local" approximation for function $u(x)$, the domain Ω is divided into subdomains

Ω_i (henceforth called "clouds of points") so that $\sum \Omega_i$ represents a covering for Ω. Each cloud of points consists of a point x_i called "star point" and a set of points $x_j, j = 1, 2, \ldots, np-1$ surrounding x_i that complete Ω_i. Assuming that the function $u(x)$ in Ω_i is smooth enough, it is possible to state the following approximation

$$u(x) \cong \hat{u}(x) = \sum_{j=1}^{m} p_j(x)\,\alpha_j = \boldsymbol{p}^T(x)\boldsymbol{\alpha} \tag{1}$$

where $\boldsymbol{p}(x)$ is a vector whose m-components are the terms of a complete polynomial basis in $\Re^d$ and $\boldsymbol{\alpha}$ is a vector whose components must be determined. These vectors are given by

$$\begin{aligned} \boldsymbol{p}_j^T &= \begin{bmatrix} p^1(x_j) & p^2(x_j)\cdots p^m(x_j) \end{bmatrix} \quad (1 \times m) \\ \boldsymbol{\alpha} &= \begin{bmatrix} \alpha^1 & \alpha^2 \cdots \alpha^m \end{bmatrix}^T \quad (m \times 1) \end{aligned} \tag{2}$$

Next, the unknown function is obtained at each point $x_j \in \Omega_i$ as follows

$$\boldsymbol{u}^h = \begin{bmatrix} u_1^h \\ u_2^h \\ \vdots \\ u_{np}^h \end{bmatrix} \cong \begin{bmatrix} \hat{u}_1 \\ \hat{u}_2 \\ \vdots \\ \hat{u}_{np} \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}_1^T \\ \boldsymbol{p}_2^T \\ \vdots \\ \boldsymbol{p}_{np}^T \end{bmatrix} \boldsymbol{\alpha} = \boldsymbol{P}\boldsymbol{\alpha} \tag{3}$$

where $u_j^h = u^h(x_j)$ is the value of the unknown function $u(x)$ at $x = x_j$, $\hat{u}_j = \hat{u}(x_j)$ is the approximated value at that point and

$$\begin{aligned} \boldsymbol{P} &= \begin{bmatrix} \boldsymbol{p}_1^T \\ \vdots \\ \boldsymbol{p}_{np}^T \end{bmatrix} \\ &= \begin{bmatrix} p^1(x_1) & p^2(x_1) & \cdots & p^m(x_1) \\ & \vdots & & \\ p^1(x_{np}) & p^2(x_{np}) & \cdots & p^m(x_{np}) \end{bmatrix} (np \times m) \end{aligned} \tag{4}$$

In order to solve the equation system (3) the condition $np = m$ must be fulfilled. This penalizes the approximation flexibility and does not suit a meshless methodology. Thus, $np > m$ is adopted, the problem then becomes overdetermined and an approximate solution of the equation system (3) is sought by means of a Weighted Least Squares (WLSQ) technique. This solution minimizes a discrete L_2 error norm in the approximation to $u(x)$ in Ω_i.

The WLSQ approximation features depend on the shape of the chosen weighting function and the manner in which the latter is applied. In the FPM a fixed weighting function, centred in the star point of the cloud, is chosen so that it satisfies the following conditions

$$\begin{aligned} \varphi_i(x_j) &> 0 \quad \forall x_j \in \Omega_i \\ \varphi_i(x) &= 0 \quad \forall x \notin \Omega_i \\ \varphi_i(x_i) &= 1 \end{aligned} \tag{5}$$

This kind of approximation is called Fixed Least Squares Method (FLS) and can be considered as a particular case of the Moving Least Squares Method (MLS) introduced by Lancaster and Salkauskas [9]. When the FLS procedure is applied, the approximation methodology is considerably simplified and its computational cost reduced. It should be noticed, though, that the approximation functions obtained are discontinuous and this fact imposes certain restrictions on the local approximation.

Going back to the minimization procedure, the following weighted discrete functional $J(x)$ is defined

$$\begin{aligned} J_i &= \sum_{j=1}^{np} \varphi_i(x_j)\left[\hat{u}_j - u_j^h\right]^2 \\ &= \sum_{j=1}^{np} \varphi_i(x_j)\left[\boldsymbol{p}_j^T\boldsymbol{\alpha} - u_j^h\right]^2 \end{aligned} \tag{6}$$

where $\varphi_i(x_j) = \varphi(x_j - x_i)$. Next, it is possible to write the discrete functional (6) in matrix form as follows

$$\boldsymbol{J} = \left(\boldsymbol{P\alpha} - \boldsymbol{u}^h\right)^T \boldsymbol{\phi}(x)\left(\boldsymbol{P\alpha} - \boldsymbol{u}^h\right) \tag{7}$$

where

$$\boldsymbol{\phi}(x) = diag\left(\varphi(x_j - x_i)\right) \quad j = 1, np \quad (np \times np) \tag{8}$$

This discrete functional can be minimized with respect to the unknown coefficients α_j setting $\partial\boldsymbol{J}/\partial\boldsymbol{\alpha} = 0$, which leads to the following equation system

$$\left(\boldsymbol{P}^T\boldsymbol{\phi}(x)\boldsymbol{P}\right)\boldsymbol{\alpha} - \left(\boldsymbol{P}^T\boldsymbol{\phi}(x)\right)\boldsymbol{u}^h = \boldsymbol{0} \tag{9}$$

also known as "*normal equations*" in the least squares literature. Introducing the following matrices

$$\begin{aligned} &\boldsymbol{A} = \left(\boldsymbol{P}^T\boldsymbol{\phi}(x)\boldsymbol{P}\right), \\ &A_{kl} = \sum_{j=1}^{np} \varphi_i(x_j)p_k(x_j)p_l(x_j) \quad (m \times m) \\ &\boldsymbol{B} = \left(\boldsymbol{P}^T\boldsymbol{\phi}(x)\right), \qquad B_{lj} = p_l(x_j)\,\varphi_i(x_j) \quad (m \times np) \end{aligned} \tag{10}$$

it is possible to express the normal equations (9) as

$$\boldsymbol{A}\,\boldsymbol{\alpha} = \boldsymbol{B}\,\boldsymbol{u}^h \tag{11}$$

Next, the vector of unknown coefficients can be found by

$$\boldsymbol{\alpha} = \boldsymbol{A}^{-1}\boldsymbol{B}\,\boldsymbol{u}^h \tag{12}$$

If the columns of matrix $\boldsymbol{P}$ are linearly independent, matrix $\boldsymbol{A}$ is positive-definite and consequently, non-singular. Then, the unknown coefficients α_j are uniquely determined by (12). However, depending on the spatial distribution of the cloud of points (and especially in 2D and 3D problems), matrix $\boldsymbol{A}$ can become ill-conditioned being very difficult to invert it with accuracy. For cases like this one, an alternative procedure is presented later on.

Finally, replacing (12) in (1), the approximation to the unknown function is obtained for $x = x_i$

$$\hat{u}(x_i) = \underbrace{\boldsymbol{p}^T(x_i)\,\boldsymbol{A}^{-1}\boldsymbol{B}}_{N^T(x_i)\quad(1\times np)}\,\boldsymbol{u}^h \tag{13}$$

where $\boldsymbol{N}(x_i)$ is the shape function vector of the point x_i in Ω_i.

The adoption of an FLS scheme, where matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are constant in Ω_i, noticeably simplifies the calculation of shape functions derivatives. Consequently,

$$\frac{\partial^k \boldsymbol{N}^T(x_i)}{\partial x^k} = \frac{\partial^k \boldsymbol{p}^T(x_i)}{\partial x^k}\boldsymbol{A}^{-1}\boldsymbol{B} \tag{14}$$

and the unknown function derivatives are calculated as

$$\frac{\partial^k \hat{u}(x_i)}{\partial x^k} = \frac{\partial^k \boldsymbol{N}^T(x_i)}{\partial x^k}\,\boldsymbol{u}^h = \frac{\partial^k \boldsymbol{p}^T(x_i)}{\partial x^k}\boldsymbol{A}^{-1}\boldsymbol{B}\,\boldsymbol{u}^h \tag{15}$$

2.1 Consistency of the approximation

It is a usual practice in meshless methods to associate (despite its mathematical meaning) the term "consistency" with the ability of a numerical method to reproduce a given polynomial of order p and its derivatives in an exact way. In other words, the ability to reproduce p-order polynomials is equivalent to p-order consistency [10]. Following this approach, it is considered that a set of functions $\boldsymbol{N}(x)$ has p-order consistency if the following conditions are satisfied

$$\sum_{j=1}^{np} N_j(x)\boldsymbol{p}(x_j) = \boldsymbol{p}(x)$$
$$\sum_{j=1}^{np} \frac{\partial^k N_j(x)}{\partial x^k}\boldsymbol{p}(x_j) = \frac{\partial^k \boldsymbol{p}(x)}{\partial x^k} \qquad \nabla x \in \Omega_i \tag{16}$$

where $\boldsymbol{p}(x)$ is a complete polynomial basis of order p [11]. It was found for the MLS approximation that if the basis is complete of order p, then consistency of order p is obtained. It can also be demonstrated that any function in the basis can be exactly reproduced [11].

Due to the fact that the shape function and their derivatives are discontinuous, in the FLS scheme adopted here it is *only* possible to satisfy the consistency requirements (16) in the cloud's star point where the weighting function is located, i.e., $x = x_i$.

2.2 Approximation bases

In this work, complete polynomial bases in $\Re^d$ are used. With the aim of avoiding numerical instabilities due to large entries in matrix $\boldsymbol{P}$, local approximation bases are defined shifting the coordinate origin to the cloud's star point. Additionally, these local bases are scaled in order to improve the conditioning of matrix $\boldsymbol{A}$ [6]. Proof that shifting/scaling approximation bases leads to the same shape functions can be found in the literature (see for instance [12]). Here, for 3D problems, the following second, thid and fourth order approximation bases are employed

$$\begin{aligned}
\boldsymbol{p}^T_{(ii)} &= \left[1, x, y, z, xy, xz, yz, x^2, y^2, z^2\right] \quad m = 10\\
\boldsymbol{p}^T_{(iii)} &= \left[1, x, y, z, x^2, xy, xz, y^2, yz, z^2, x^3, x^2y, x^2z, xy^2,\right.\\
&\quad \left. xyz, xz^2, y^3, y^2z, yz^2, z^3\right] \quad m = 20\\
\boldsymbol{p}^T_{(iv)} &= \left[1, x, y, z, x^2, xy, xz, y^2, yz, z^2, x^3, x^2y, x^2z, xy^2,\right.\\
&\quad xyz, xz^2, y^3, y^2z, yz^2, z^3,\\
&\quad x^4, x^3y, x^3z, x^2y^2, x^2yz, x^2z^2, xy^3, xy^2z, xyz^2, xz^3,\\
&\quad \left. y^4, y^3z, y^2z^2, yz^3, z^4\right] \quad m = 35
\end{aligned} \tag{17}$$

where

$$x = \frac{x_j - x_i}{d_{\max}}, \quad y = \frac{y_j - y_i}{d_{\max}}, \quad z = \frac{z_j - z_i}{d_{\max}} \tag{18}$$

and $d_{\max} = \max(||\boldsymbol{x}_j - \boldsymbol{x}_i||)$ is the distance between the star point and the furthest point in the cloud.

2.3 On the weighting function

The introduction of a compact support weighting function into the minimization problem allows focusing on the information in the close neighbourhood of the star point; thus, enhancing the local character of the approximation. There exist many possibilities for choosing the functional form of a weighting function that satisfies the conditions given in (5). In FPM, a normalized Gaussian function is chosen and defined by

$$\varphi_i(x_j) = \frac{e^{-(d_j/\alpha)^k} - e^{-(\beta/\alpha)^k}}{1 - e^{-(\beta/\alpha)^k}} \tag{19}$$

being $d_j = |\boldsymbol{x}_j - \boldsymbol{x}_i|, \alpha = \beta/w$ and $\beta = \gamma d_{max}$ ($\gamma > 1.0$). The support of this function is isotropic, circular in 2D and spherical in 3D. The parameters w, k and γ govern the functional shape of the weighting function. These are free parameters that should be properly set; moreover, the final approximation features are highly dependent on these parameters.

It is very difficult to define a combination of parameters which allows getting an optimal global approximation for a given problem. Due to this fact, the freedom to locally set and modify the approximation through a variation of the functional form of the weighting function is an important tool for the numerical discretization of complete geometries. Next, a brief analysis of the free parameters involved in the weighting function definition and their relation with the numerical approximation is outlined.

Parameter γ provides more or less weight to the boundary points of the cloud by increasing or decreasing the size of the weighting function's support. It is necessary to point out here that in our case the number of points in each cloud is defined *a priori*. Then, the support size is already determined when the numerical approximation is computed. Hence, only a small effect of the variation of γ on the approximation is observed when $np \approx m$ and this effect becomes negligible when the number of points is increased. The error in the approximation to the function tends to become higher when γ is increased while the error in the approximation to the derivatives of the function becomes smaller. Anyway, these effects are not relevant and the parameter γ is set to a constant value $\gamma = 1.01$ in the whole domain Ω. Figure 1 shows the effect of parameter γ on the approximation. The test problem is a Poisson's problem $\nabla^2\phi = f(x, y, z)$ solved in a cubic domain and the numerical results correspond to an isolate cloud centred in the domain. A complete description of this problem is presented later.

The next free parameter in expression (19) is the exponent k. This parameter changes the shape of the weighting function and increases the weight in the close neighbourhood of the star point at the same time it decreases the weight of the boundary points or viceversa; see Fig. 2b. The effect on the numerical approximation is significant and could be interesting for particular discretization cases. However, since we want to introduce small adjusts in the approximation, through minimum variations of the weighting function, the parameter k is not suitable for that purpose. Therefore, this parameter is set to a constant value $k = 2$ in the whole domain Ω.

Finally, the only parameter taken into account in order to locally adjust the weighting function is the parameter w. It allows changing the weighting of the points in the minimization problem and modifies the local character of the latter. The effect of parameter w on the functional form of the weighting function is presented in Fig. 2a.

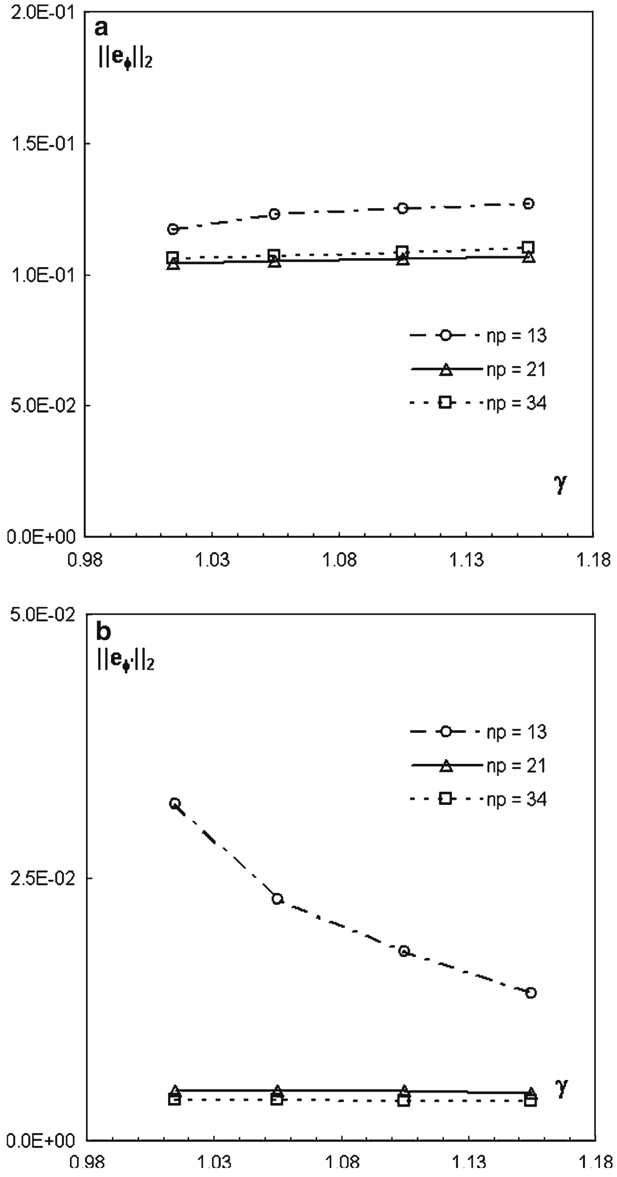


Fig. 1 Effect of the parameter γ on the L$_2$-norm of the error in **a** the variable ϕ, **b** the first derivatives (average) for clouds with different number of points (np); $w = 3.5$ and second-order approximation bases

For large values of w, the shape function tends to the Dirac's delta function (see Fig. 2a) and the approximation procedure tends to interpolate nodal data. When w is increased, the values of $N_j(x) \to 0$ except at the star point where $N_j(x) \to 1$, i.e., the shape function also tends to the Dirac's delta function. This causes the error in the approximation to decrease and the condition number of matrix $\boldsymbol{A}$ ($\kappa(\boldsymbol{A})$) to increase and, as a consequence, the problem becomes more and more

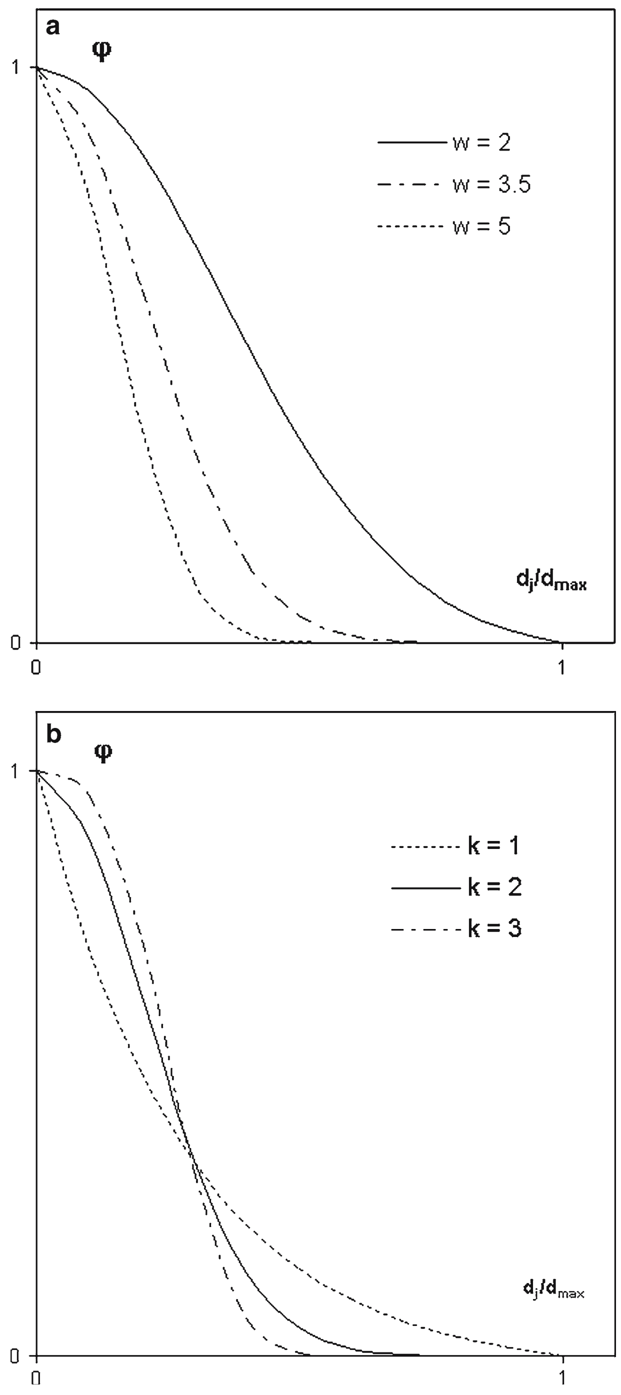


Fig. 2 Effects of the parameters w and k on the weighting function shape **a** effect of parameter w, $\gamma = 1.01$ and $k = 2$; **b** Effect of parameter k, $\gamma = 1.01$ and $w = 3.5$

ill-conditioned. Beyond a given threshold it is not possible to invert matrix $\boldsymbol{A}$ with accuracy, the approximation quality deteriorates quickly and numerical instabilities appear. In Fig. 3 the effect of w on the numerical approximation and the condition number of matrix $\boldsymbol{A}$ is shown for the same Poisson's test problem in Fig. 1. The results displayed correspond to an isolated homogeneous cloud centred in the analysis domain.

Figure 3 above shows that when w increases, the error in the approximation of the unknown function decreases while the approximation function tends to interpolate the nodal values. For a value of parameter $w \approx 5$, ill-conditioning of matrix $\boldsymbol{A}$ becomes relevant and the approximation error rises slowly. Finally, the process diverges because it is not possible to invert matrix $\boldsymbol{A}$ accurately enough.

Taking into account numerical experiments, a maximum admissible range for parameter w given by $3.0 \leq w_{max} \leq 4.5$ for 3D problems is chosen. Note, however, that this range must be defined for each particular problem. In this work, a value $w_{max} = 3.5$ is adopted in the whole domain and then it is reduced in a local manner, i.e., for each cloud of points, whenever necessary, in order to obtain a given accuracy in the approximation to the unknown function and their derivatives. We will go back to this point later on.

2.4 Discretization of equations

In the FLS method the weighting function is fixed at the star point of the cloud. The fact that we have different shape functions for each star point (depending on the cloud in which the shape function was calculated) leads to multivalued shape functions, i.e., $N_j(x_i) \neq N_j(x_k)$. The approximation is globally and locally discontinuous; hence, it must be only considered as valid in the star point of the cloud where the shape function is located. Consequently, a collocation technique becomes the natural choice in FPM.

Collocation procedures are simple and easy to implement; however, special care must be taken of the resultant global equation system since they are likely to suffer numerical instabilities. There is evidence in the literature that the robustness of collocation methods can be enhanced working on the local approximation properties; for example, enforcing certain conditions such as "positivity" by means of the biased selection of the cloud's points or through the local manipulation of the weighting function [13]. In our work the stabilization of the FPM equations has been achieved by solving modified governing equations obtained via the so-called Finite Calculus method. For details see [14]. A similar approach is presented in [15].

3 Computation of the shape functions parameters

According to the FPM approximation methodology presented before, in order to get the unknown coefficients α_j

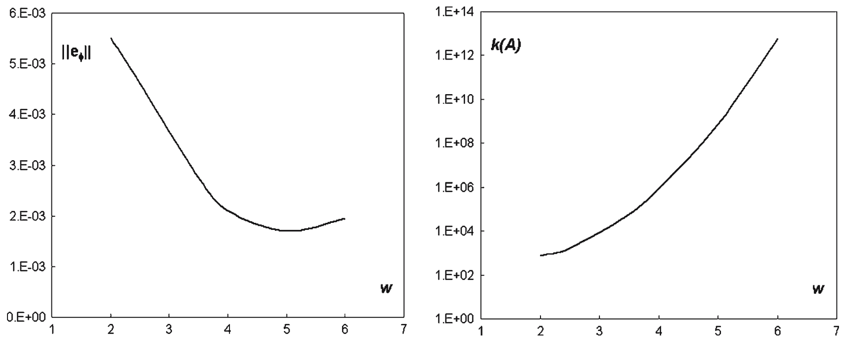


Fig. 3 Effect of parameter w on the L_2-norm of the error in the variable ϕ and in the condition number of matrix $\boldsymbol{A}$; $np = 35$ and second-order approximation bases

and, consequently, the shape functions and their derivatives for a given cloud of points, the following linear system must be solved

$$\boldsymbol{A}\,\boldsymbol{\alpha} = \boldsymbol{B}\,\boldsymbol{u}^h \tag{20}$$

It should be noticed that this system must be solved via the inversion of matrix $\boldsymbol{A}$ because $\boldsymbol{u}^h$ is not known in advance. Although this methodology is fast, it is not the most accurate for solving least squares problems, especially when the condition number of matrix $\boldsymbol{A}$ is large. If we observe the structure of this matrix, we can see that it is composed of a Vandermonde type matrix multiplied by a diagonal matrix, which is, in turn, multiplied by another Vandermonde type matrix. Consequently, the final characteristics of matrix $\boldsymbol{A}$ are similar to Vandermonde type matrices causing matrix $\boldsymbol{A}$ to be "naturally" ill-conditioned. Obtaining an accurate solution of system (20) is a key task in most meshless methods.

Introducing in (20) matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ defined by (10), the normal equations are recovered

$$\left(\boldsymbol{P}^T\phi(x)\,\boldsymbol{P}\right)\boldsymbol{\alpha} = \left(\boldsymbol{P}^T\phi(x)\right)\boldsymbol{u}^h \tag{21}$$

Thus, the vector of the unknown coefficients is obtained as follows

$$\boldsymbol{\alpha} = \left(\boldsymbol{P}^T\phi(x)\,\boldsymbol{P}\right)^{-1}\left(\boldsymbol{P}^T\phi(x)\right)\boldsymbol{u}^h \tag{22}$$

It is possible to prove that if matrix $\boldsymbol{P}$ has rank m, i.e., all their columns are linearly independent, matrix $\boldsymbol{P}^T\phi(x)\,\boldsymbol{P}$ is positive-definite and, consequently, non-singular (note that matrix $\phi(x)$ is positive-definite by definition). Then, the inverse matrix exists and the unknown coefficients are uniquely determined.

The solution of the equations (22) by direct inversion of matrix $\boldsymbol{A}$ must be restricted to cases when the condition number of $\boldsymbol{A}$ is moderate. Generally, when the condition number of matrix $\boldsymbol{A}$ is large, its inverse is not appropriate to calculate the shape function and their derivatives, even for cases when it is still numerically possible to obtain one.

In this work, the procedure adopted to calculate the shape function and their derivatives is the following. Given a certain cloud of points, first, the direct inversion of matrix $\boldsymbol{A}$ is attempted. If the condition number of $\boldsymbol{A}$ is smaller than a given maximum admissible value and if the calculated shape functions satisfy some quality tests, then, the shape functions are accepted. If some of the preceding requirements are not met, the normal equations (21) are solved by an alternative procedure based on QR factorization.

3.1 Solution of the normal equations via QR factorization

QR factorization is a more stable and accurate method for solving least squares problems when matrix $\boldsymbol{A}$ is ill-conditioned. The aim of using a QR factorization technique for the WLSQ problem is to get an acceptable solution in cases where the other procedure fails without having to modify the geometrical structure of the cloud. From the computational cost's point of view, the least squares solution via QR factorization costs up to twice as much as the solution via matrix $\boldsymbol{A}$ inversion if $np \gg m$ [16].

If matrix $\boldsymbol{P}$ has rank m and $np > m$, then it can be uniquely factored as

$$\boldsymbol{P} = \boldsymbol{QR} \tag{23}$$

where $\boldsymbol{Q} \in \Re^{\mathrm{npxm}}$ is orthogonal ($\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$) and $\boldsymbol{R} \in \Re^{\mathrm{mxm}}$ is upper triangular with positive diagonal elements $R_{ii} > 0$. A similar procedure, based on columns pivoting, can be applied for cases in which matrix $\boldsymbol{P}$ is rank deficient or near rank deficient.

In order to apply the QR factorization for solving the WLSQ problem, it is necessary to obtain an equivalent unweighted problem. Thus, the following factorization is proposed

$$\tilde{\phi}(x) = \sqrt{\phi(x)} \quad \text{such that} \quad \tilde{\phi}^{\boldsymbol{T}}\tilde{\phi} = \phi \tag{24}$$

and also the following modification of matrix $\boldsymbol{P}$

$$\tilde{\boldsymbol{P}} = \tilde{\phi}\,\boldsymbol{P} \tag{25}$$

After that, it is possible to write a system equivalent to Eq. (21) as

$$\left(\tilde{\boldsymbol{P}}^T \tilde{\boldsymbol{P}}\right) \alpha = \left(\tilde{\boldsymbol{P}}^T \tilde{\phi}\right) \boldsymbol{u}^h \tag{26}$$

The equivalence between Eqs. (26) and (21) is verified by

$$\begin{aligned}
&\tilde{\boldsymbol{P}}^T \tilde{\boldsymbol{P}} \boldsymbol{\alpha} = \tilde{\boldsymbol{P}}^T \tilde{\phi}\, \boldsymbol{u}^h \\
&(\tilde{\phi}\boldsymbol{P})^T \tilde{\phi} P \boldsymbol{\alpha} = (\tilde{\phi} P)^T \tilde{\phi}\, \boldsymbol{u}^h \\
&\boldsymbol{P}^T (\tilde{\phi}^T \tilde{\phi}) \boldsymbol{P} \boldsymbol{\alpha} = \boldsymbol{P}^T (\tilde{\phi}^T \tilde{\phi})\, \boldsymbol{u}^h \\
&\boldsymbol{P}^T (\phi) \boldsymbol{P} \boldsymbol{\alpha} = \boldsymbol{P}^T (\phi)\, \boldsymbol{u}^h
\end{aligned}$$

Once the equivalent system (26) is obtained, the modified matrix (25) is factorized, i.e., $\tilde{\boldsymbol{P}} = \boldsymbol{QR}$ and replaced in this system leading to

$$\begin{aligned}
&(\boldsymbol{QR})^T \boldsymbol{QR}\, \boldsymbol{\alpha} = (\boldsymbol{QR})^T \tilde{\phi}\, \boldsymbol{u}^h \\
&\boldsymbol{R}^T (\boldsymbol{Q}^T \boldsymbol{Q})\, \boldsymbol{R}\, \boldsymbol{\alpha} = \boldsymbol{R}^T \boldsymbol{Q}^T \tilde{\phi}\, \boldsymbol{u}^h
\end{aligned} \tag{27}$$

where $\boldsymbol{Q}^T \boldsymbol{Q} = \boldsymbol{I}$ due to the orthogonality property and matrix $\mathbf{R}$ is invertible (upper triangular with positive diagonal elements). Multiplying both sides by $(\mathbf{R}^T)^{-1}$, we get

$$\boldsymbol{R}\, \boldsymbol{\alpha} = \boldsymbol{Q}^T \tilde{\phi}\, \boldsymbol{u}^h \tag{28}$$

The unknown coefficients α_j can be finally obtained by

$$\boldsymbol{\alpha} = \boldsymbol{R}^{-1} (\boldsymbol{Q}^T \tilde{\phi})\, \boldsymbol{u}^h \tag{29}$$

Here matrix $\boldsymbol{R}$ is generally well-conditioned and its inverse is easy to obtain with accuracy, even for the cases when matrix $\boldsymbol{P}$ is near rank-deficient.

The described procedure allows getting shape functions of acceptable quality in cases where these can not be obtained via direct inversion of matrix $\boldsymbol{A}$. This fact reduces the approximation's dependence on the spatial distribution of points and on the functional shape of the weighting function significantly, giving robustness to the Finite Point methodology.

4 Shape functions calculation an iterative procedure

With the aim of obtaining a suitable high-order approximation in a given cloud of points despite its geometrical configuration, the following iterative procedure is proposed. First, the maximum admissible value of the parameter w of the weighting function is set ($w_{\text{ini}} = w_{\max}$) and the WLSQ problem is solved via matrix $\boldsymbol{A}$ inversion (12). Then, the shape function and their derivatives are obtained by (13) and (15), respectively. The resulting approximation is accepted if it satisfies the following requirements:

r$_1$. $\kappa(A) \le \kappa_{\max}(A)$
r$_2$. $\sum_j N_j - 1.0 \leqslant tol \quad y \quad \sum_j \frac{\partial N_j}{\partial x} \leqslant tol$
r$_3$. consistency

The first requirement (r$_1$) imposes a limit to the condition number of matrix $\boldsymbol{A}$ in order to guarantee that the latter is correctly inverted. The second requirement (r_2) implies that the shape functions and their derivatives must build a "*partition of unity*" (PU) and a "*partition of nullities*" (PNs), respectively. The last requirement (r$_3$) enables the verification of the approximation accuracy by checking the consistency requirements (16) in the cloud's star point. To achieve this, it is also possible to replace the approximation basis by a known function and check how much the approximated values deviate from the exact values; see for instance [7]. The values adopted for setting $\kappa_{\max}(A)$, the parameter *tol* and the admissible error in the consistency check depend on the problem under consideration. In this work a value $\kappa_{\max}(A) = 1.\text{E}6$ based on the infinite norm of matrix $\boldsymbol{A}$ and the parameter $tol = 1.\text{E-}10$ are adopted. The consistency check (r3) is performed according to the guidelines given in [7].

If any of the preceding requirements is not satisfied, the approximation is rejected and the solver changes to the QR factorization based methodology (29) keeping all approximation parameters constant. In general, the QR factorization allows obtaining a suitable approximation where the matrix $\boldsymbol{A}$ inversion procedure fails. However, in particular cases where highly distorted clouds of points are to be dealt with, it is possible that the local approximation obtained via QR factorization also fails and does not meet the requirements given by r$_1$, r$_2$ and r$_3$. In this case the approximation is improved iteratively. In each iteration the parameter w is decreased setting $w = w^i = \alpha w^{i-1} (\alpha \approx 0.75, w^0 = w_{\max}, i$:iteration counter) and the numerical approximation is calculated again via the QR factorization technique. This procedure continues until all the requirements are satisfied or w reaches a minimum admissible value $w_{\min}$. Numerical experiments have shown that two or three iterations are enough to improve the approximation in highly distorted clouds of points (if $w_{\max}$ is large).

Finally, if a local cloud of points does not allow constructing an appropriate high-order approximation, it is possible to decrease its local order of approximation and start the described shape function calculation procedure again. This last option avoids the necessity of regenerating the cloud of points but its usefulness and effects on the global approximation must be evaluated for each particular problem.

It should be noticed that the first requirement (r_1) loses its value when the QR solver is selected and in that situation $\kappa(\boldsymbol{R})$ is tested.

5 Generating local clouds of points

An adequate support of points is essential for setting a good local approximation for each cloud. The quality of the local approximation depends on the number of points in the cloud and their spatial position in relation to the star point. Even though the shape functions calculation technique already presented attempts to reduce this dependence, the approximation's spatial support continues playing a major role.

At present, there is not a unique reliable criterion that allows determining the size, shape and spatial structure of the local support. Some numerical techniques applied in this order belong to geometrical intuitive considerations such as symmetry, cloud's centre of gravity position, etc. Other techniques introduce mathematical considerations based on the structure of the matrices involved in the shape function's calculation procedure focusing, for example, on conditioning and invertibility features [17,18]. Mixed geometrical-mathematical considerations are also used. Among them, enough overlap within approximation subdomains criteria and other techniques related, for instance, to Point Collocation procedure's stability and the so called "positivity conditions" can be mentioned [13,15,19]. All these criteria, often employed with the aim to obtain *a priori* an acceptable local support for the numerical approximation, lead to methods for generating the required point's connectivity.

A reliable methodology based on a Delaunay technique to generate local clouds of points for Finite Point approximations was proposed in [7]. In the present work the generation of local clouds of points has not a major role. Only some geometrical considerations are applied in order to guarantee that the "physical situation" of all the points in the local cloud is compatible. The procedure here employed is as follows.

Given a point discretization of the whole domain and a set of normal vectors belonging to the surfaces that bound this domain, a maximum ($np_{\max}$) and minimum ($np_{\min}$) allowable number of points in the cloud and an initial search radius are set. Then, for each star point x_i all neighbours within the search radius (r_{search}) are found through an octree technique. If the points found are not enough, the search radius is increased until condition $np_{min} \leq np \leq np_{\max}$ is satisfied. For every star point in the domain which is located either over a surface (solid boundary) or sufficiently close to a surface, the points included in its cloud must satisfy the conditions described bellow.

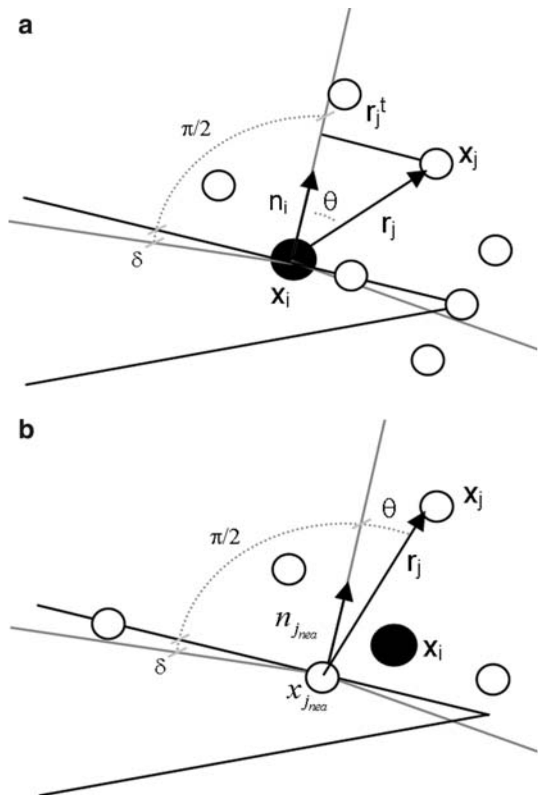


Fig. 4 Local clouds generation cases **a** star point located over a surface; **b** cloud of points intercepting a surface

Case 1: star point located over a surface
In this particular case (sketched in Fig. 4a), every point in the cloud x_j, $j = 1, np - 1$ located within the search radius is accepted if it meets the following conditions

$$\cos(\theta) \geqslant \cos\left(\frac{\pi}{2} + \delta\right); \quad \cos(\theta) = \frac{n_i \cdot r_j}{\|n_i\| \|r_j\|} \tag{30}$$

$$r_j^t < \alpha \, r_{search} \tag{31}$$

Condition (30) determines an acceptable domain around the start point, which is defined in the normal direction to the surface, and δ is a small angle dependent on the surface curvature. The second condition (31) imposes a certain aspect ratio in the cloud given by the parameter $\alpha > 0$.

Case 2: cloud of points intercepting a surface
In this case the point x_j located over a surface ($x_{j_{nea}}$), nearest to the star point x_i, must be sought (see Fig. 4b). Then, every point in the cloud is accepted if

$$\cos(\theta) \geqslant \cos\left(\frac{\pi}{2} + \delta\right); \quad \cos(\theta) = \frac{n_{j_{nea}} \bullet r_j}{\|n_{j_{nea}}\| \|r_j\|} \tag{32}$$

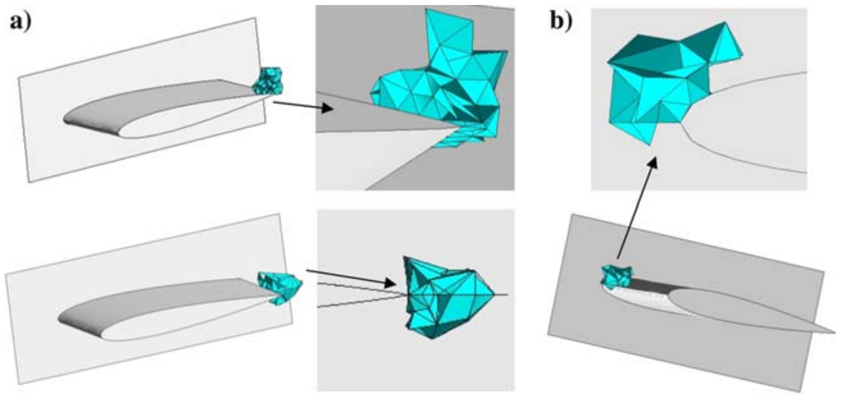


Fig. 5 Clouds of points generated in **a** case 1:; b case 2:

and no restriction is imposed to the aspect ratio of the cloud of points.

Next, some clouds of points generated by the presented techniques are shown in Fig. 5.

6 Numerical examples

Three-dimensional numerical examples are presented next in order to set forth some features of the presented Finite Point methodology. The first and second examples attempt to investigate h and p convergence characteristics of the method using second, third and fourth order approximation bases. In the last example, the performance of the FPM is shown in a more realistic calculation case which solution involves all the preceding techniques.

6.1 Poisson's problem in a cubic domain

The following Poisson's problem is solved

$$\begin{aligned} &\nabla^2\phi = f(x,y,z) && \text{in } \Omega \\ &\phi = 0 && \text{on } \Gamma_\mathrm{D} \end{aligned} \tag{33}$$

where Ω is a unit length sides cubic domain with Dirichlet boundary $\Gamma = \Gamma_\mathrm{D}$. The source term $f(x,y,z)$ is given by

$$\begin{aligned} f(x,y,z) = \big\{ &-2kyz(1-y)(1-z) \\ &+\left[kyz(1-y)(1-z)(1-2x)^2\right]^2 \\ &-2kxz(1-x)(1-z) \\ &+\left[kxz(1-x)(1-z)(1-2y)^2\right]^2 \\ &-2kyx(1-x)(1-y)+\big[kxy(1-x)(1-y) \\ &\times(1-2z)^2\big]^2\big\}\ \frac{e^{kxyz(1-x)(1-y)(1-z)}}{1-e^{k/64}} \end{aligned} \tag{34}$$

where $k = 200$. The above problem has the following analytical solution

$$\phi(x,y,z) = \frac{1-e^{kxyz(1-x)(1-y)(1-z)}}{1-e^{k/64}} \tag{35}$$

which is used to assess the accuracy of the numerical solution. The error in the numerical calculations is evaluated by means of a discrete average quadratic norm given by

$$\|\varphi\|_2 = \left(\frac{\sum\limits_{i=1}^{n}(\varphi_i^e-\varphi_i^n)^2}{\sum\limits_{i=1}^{n}(\varphi_i^e)^2}\right)^{1/2} \tag{36}$$

where ϕ is any variable for which the error is calculated and $(\ )^e$ and $(\ )^n$ refer to the exact solution and the numerical FPM solution, respectively.

The domain is discretized by unstructured and homogeneous distributions of 650, 2,013, 4,468, 8,647 and 19,850 points. Clouds of 21, 40 and 75 points are used with second, third and fourth order approximation, respectively. The initial parameter $w = w_{max} = 3.5$ is set for all cases and it is locally adjusted when the requirements in Sect. 4 are not satisfied by the local approximation. It must be noticed that it is not allowed to decrease the local order of approximation during the shape functions calculation procedure. The equation system is solved iteratively by a Bi-Conjugate Gradient Method (BiCGM) and the stopping criterion employed is $\|\mathbf{res}\|_\infty \leqslant 1.\mathrm{E}-12\,\|\mathbf{RHS}\|_\infty$. Figure 6 shows the FPM solution for the variable ϕ and the test case $n = 4,468$.

Next, the spatial convergence characteristics of the numerical solution for the Laplacian of the unknown function ϕ are investigated. The error norm used is $e = \left\|\nabla^2\phi\right\|_2$ and h is taken as an average point spacing of the spatial discretization.

Good convergence rates (indicated with r in the figure above) can be observed for the present problem. Figure 7 shows that high-order approximations do not

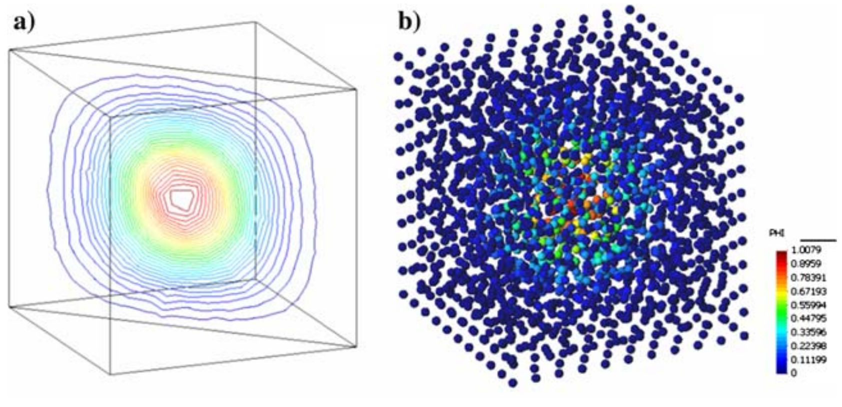


Fig. 6 **a** Iso-ϕ lines on a surface cut in the domain. b Problem discretization displaying ϕ-results

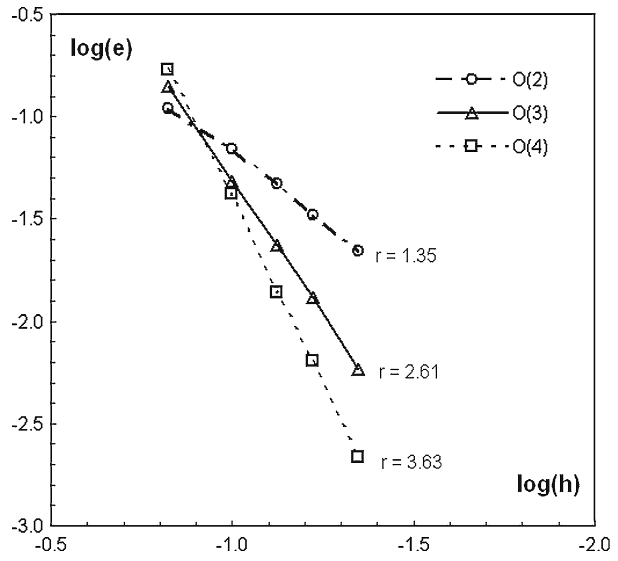


Fig. 7 Poisson's problem in a cubic domain: h-convergence for $\nabla^2\phi$

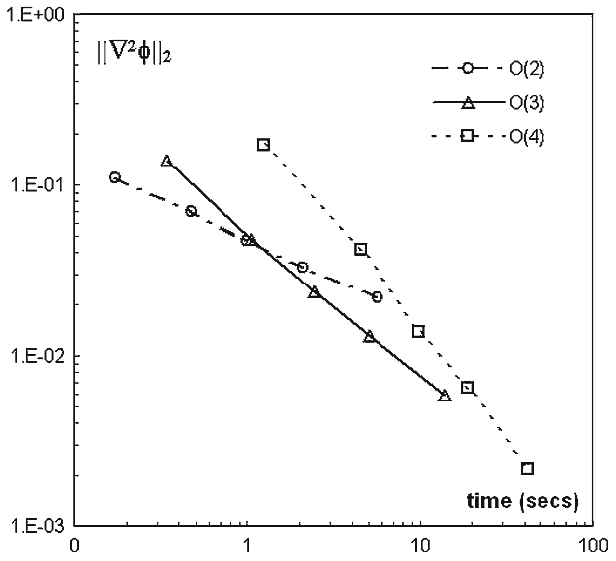


Fig. 8 Error versus CPU run time

improve accuracy when very coarse discretizations are employed. This is an expected behaviour because an increase in the approximation order implies an increase in the support size. This fact brings about extensive clouds of points which cover the problem domain causing the computed unknown function, and specially their derivatives, to be considerably smoothed. This behaviour can be improved using an appropriate domain discretization in such a way that the solution lies within the asymptotic range of convergence.

It should be noticed that the convergence characteristics of FPM solutions are very dependent on local approximation parameters such as np and w and the geometrical distribution of points. Consequently, particular settings could originate a non-expected behaviour of the convergence rates in some variables for which observed and theoretical orders of convergence do not agree.

The convergence of the error norm versus CPU-time is examined in Fig. 8. All cases were computed on a Pentium IV 3.0 GHz processor based machine.

For this problem it is possible to note that high-order approximations allow getting a better accuracy saving CPU-time and storage depending on the spatial discretization employed. As regards the CPU times, the most accurate solution is not always the fastest one but in some cases high-order accurate solutions involve a significant storage savings. From the point of view of the conditioning of the global equation system, Fig. 9 shows that high-order approximations do not necessarily imply ill-conditioning due to the increase in the band-width of the system. The relation between the error and the number of BiCG solver iterations necessary to achieve a given residual seems to be only proportional to the size of the system to be solved. Here the fourth-order approximations present the best rate.

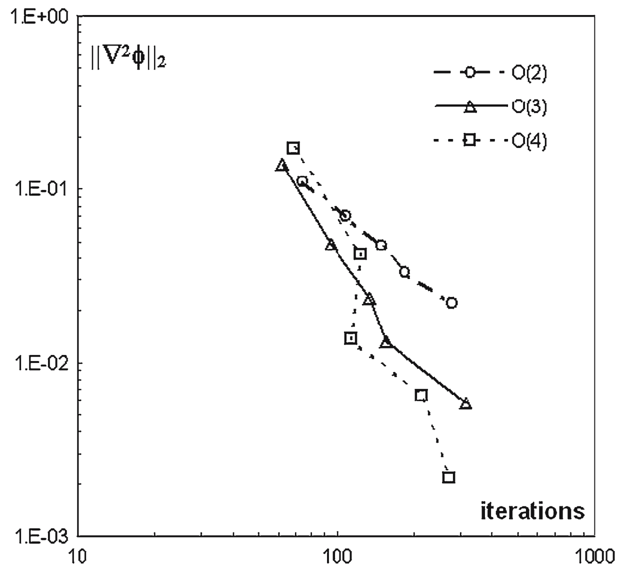


Fig. 9 Error versus iterations of BiCG solver

6.2 Potential flow around a sphere

In this numerical example an ideal, irrotational and incompressible fluid past around a stationary sphere is solved in a closed domain Ω with boundary $\Gamma = \Gamma_D \cup \Gamma_N$. These assumptions lead to the following Laplace's problem

$$\begin{aligned} &\nabla^2\phi = 0 && \text{in } \Omega \\ &\phi - \phi_D = 0 && \text{on } \Gamma_D \\ &\hat{n}\cdot\nabla\phi = 0 && \text{on } \Gamma_N \end{aligned} \tag{37}$$

where Γ_D and Γ_N are Dirichlet and Neumann boundaries, respectively, and $\hat{n}$ is the unitary outward normal vector to Γ_N. Appropriate boundary conditions are set in such a manner that originates an unperturbed velocity field given by $\vec{v}_\infty = (1, 0, 0)$. Furthermore, in this example a modified form of the Neumann's boundary condition derived through Finite Calculus technique [14] is adopted with the aim of overcoming numerical instabilities in the numerical solution.

Due to geometry and flow symmetry, only a half-sphere is computed. The computational domain is discretized by a homogeneous unstructured distribution of 7,763 points and p-convergence is examined. The surface of the half-sphere is covered by a coarse distribution of 253 unstructured points. Clouds of $25 \le np \le 35$, $40 \le np \le 55$ and $80 \le np \le 90$ are used with second, third and fourth order approximation, respectively. The parameter $w = 3.0$ is kept fixed in all test cases and QR solution of the WLSQ problem is employed when requirements given in Sect. 4 are not satisfied by the local approximation. Similar to the numerical example in Sect. 6.1, the order of the local approximation is not allowed to change during the shape functions calculation procedure. The global equation system is solved by an iterative BiCG solver and the stopping criterion is the same as for that example.

Numerical results of the pressure coefficient (Cp) and the velocity field calculated for the fourth-order approximation case are shown in Fig. 10.

A comparison between the analytical Cp distribution along a cross section of the sphere and numerical results obtained using second, third and fourth-order approximations is presented in Fig. 11. The discrete L_2-norms of the error in the numerical approximations are 1.7E-2, 1.3E-2 and 8.8E-3 for the second, third and fourth order approximations, respectively. These results show that the numerical solution converges to the analytical solution when the order of approximation is increased. Note that the spatial discretization is the same in the three cases.

Spatial convergence of the solution around the given cross section is examined for three different unstructured, non-homogeneous distributions of points over the sphere using second-order approximation bases. The surface of the half-sphere is covered by 221, 359 and 1,167 points concentrated around the cross section where the approximation error is computed. Each of these half-sphere surface discretizations belong to an unstructured discretization with 7,657, 8,151 and 10,683 points. Local approximation is built on $25 \le np \le 35$

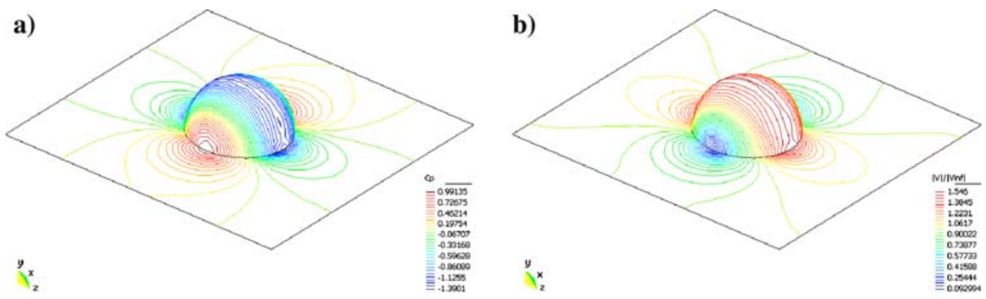


Fig. 10 Potential flow around a sphere **a** iso-lines of Cp; **b** modulus of velocity. Fourth-order calculation case, $n = 7{,}763$ and $80 \le np \le 90$

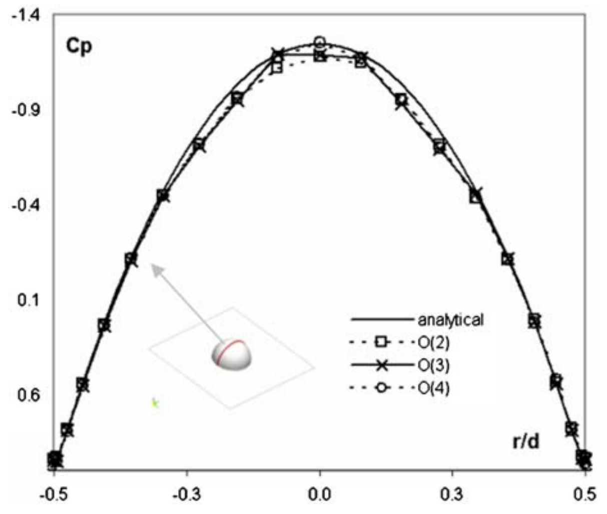


Fig. 11 Cp distributions over the sphere using second, third and fourth order approximation bases

clouds of points with the parameter $w = 3.0$. This setting is kept fixed for the three cases here analyzed. The surface point discretizations over the symmetry plane of the problem are shown in Fig. 12.

The convergence behaviour with the number of points and the Cp distribution on the sphere for the finest surface discretization ($n_s = 1,167$) are shown in Fig. 13. Spatial convergence of the solution in a localized area of the domain is obtained using second-order approximation bases. A similar behaviour is observed using high-order approximations. It should be noticed that in these cases, parameters such as w and the number of points in the cloud must be adjusted according to the local discretization so as to get the best results. This evidences the susceptibility of high-order approximations to the spatial distribution of the points which claims for an individual setting of the approximation's parameters in each cloud.

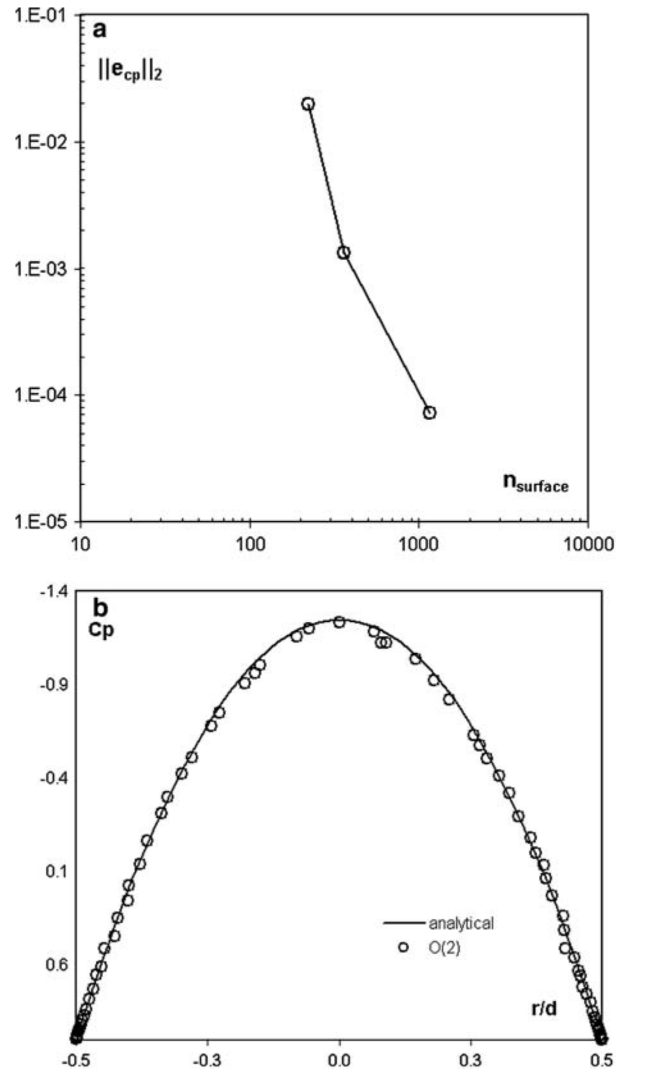


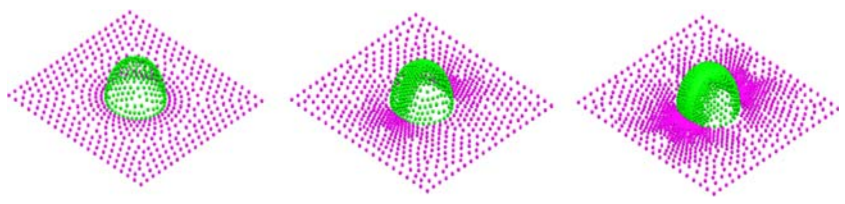
Fig. 13 **a** L_2-norm error versus number of half-sphere surface points, O(2) with $25 \leq np \leq 35$. b Cp distribution over the sphere. Comparison between calculated and analytical results, $n = 10{,}683$ and $n_{\text{surface}} = 1{,}167$

6.3 Potential flow around a semispan wing

The last numerical example is the 3D solution of an ideal irrotational and incompressible fluid past around a semispan wing. The system of equations (37) is solved in a closed domain Ω with boundary $\Gamma = \Gamma_D \cup \Gamma_N$ and proper boundary conditions are imposed in such a manner that originate an unperturbed velocity field given by $\vec{v}_\infty = (1, 0, 0)$. In this example, the modified form of the Neumann's boundary condition derived through the Finite Calculus technique [14] is also adopted in order to avoid numerical instabilities.

The semispan wing is set to zero incidence angle and has an aspect ratio $A = 8$, taper ratio $\lambda = 0.5$ and zero sweep-angle with respect to the quarter-chord line. The

Fig. 12 Symmetry plane of the problem. From left to right, half-sphere surface discretization with 221, 359 and 1,167 points, respectively



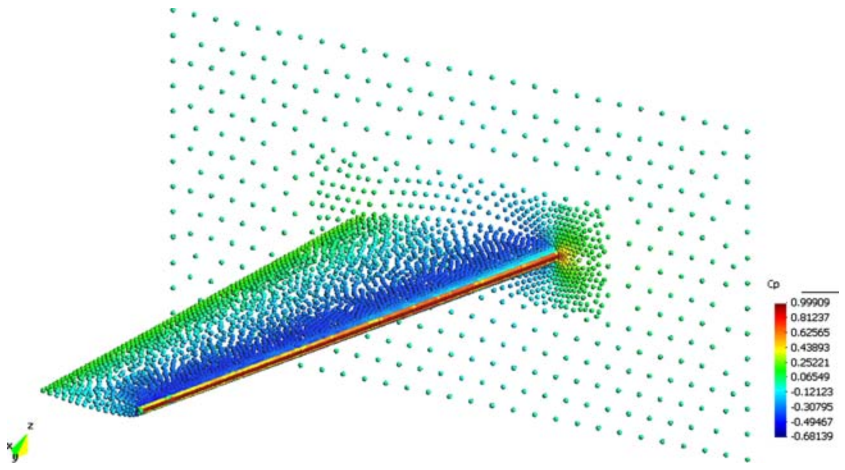


Fig. 14 Surface discretization over the semispan wing and the symmetry plane showing surface Cp results

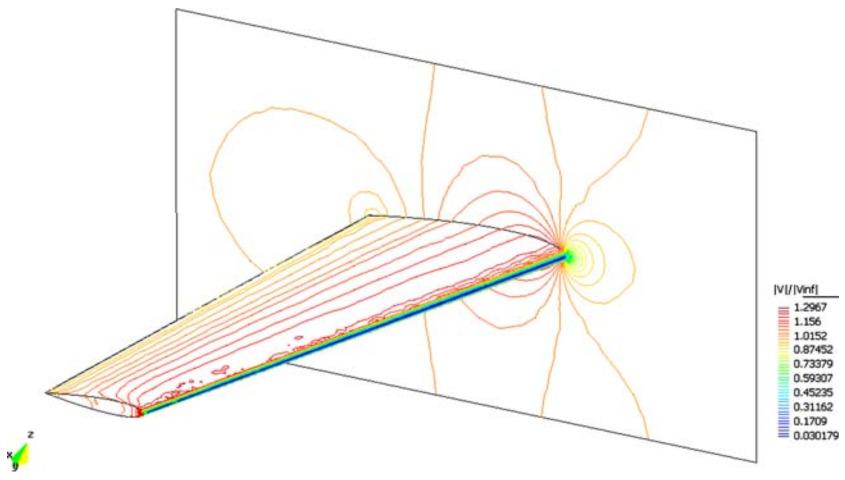


Fig. 15 Numerical surface results for non-dimensional modulus of velocity

airfoil section is a NACA-0012 constant along the semi-span.

The computational domain is discretized by an unstructured and non-homogeneous distribution of 43,335 points. Second-order approximation bases are used with clouds of $50 \leq np \leq 70$ and the shape functions calculation procedure is allowed to self-adjust according to the iterative procedure presented in Sect. 4. In this example it is unnecessary to decrease the order of the local approximation in any cloud of points. As in the previous examples, the global equation system is solved by a BiCG method.

Figure 14 shows the surface discretization over the wing (4,689 points) and the symmetry plane (837 points); the points colour display the computed pressure coefficient values. The velocity field around the semispan wing is shown in Fig. 15.

The Cp distribution obtained with the present methodology along the root section of the wing is compared with accurate 2D results in Fig. 16. A reasonable agreement can be observed.

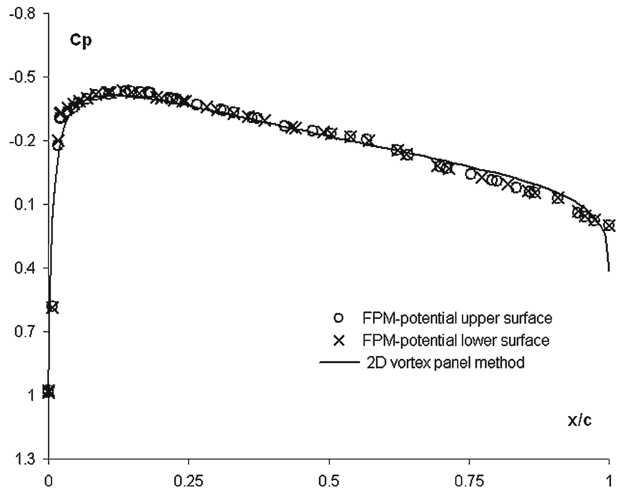


Fig. 16 Comparison of Cp distributions along the root section of the semispan wing

With the aim of demonstrating the performance of the proposed methodology, the CPU-time for each one of the stages of the calculation is presented in Table 1.

Table 1 CPU-times for semispan wing test case ($n = 43,335$, $50 \leq np \leq 70$ and second order approximation bases)

	CPU-time (s)	Overall time (%)
Local cloud generation	4.65	6.59
Shape functions calculation	6.1	8.64
Assembly of the equation system	0.3	0.42
Equation system solution (BiCGM)	50.6	71.67
Others	8.95	12.68
Total:	70.6	100

This numerical example was computed on a Pentium IV 3.0 GHz processor based machine.

As it has been mentioned before, the iterative shape functions calculation procedure has been employed and the time involved in this task is 6.1 s. If direct inversion of matrix $\boldsymbol{A}$ procedure (12) is used for the shape functions calculation, the time needed is about 5.1 s; while if the alternative QR factorization based procedure (29) is used in all the domain the CPU-time is 11.5 s. These facts demonstrate that the iterative procedure only needs a little more time than the usual procedure, and takes around twice more as much in the worst case, when all shape functions in the domain must be recalculated.

Several numerical experiments that are not reported here confirm that the iterative shape function calculation procedure has a noticeably positive impact on the accuracy of the numerical solution, the stability of collocation procedure and the iterative convergence of the BiCGM.

7 Conclusions

An alternative procedure to obtain shape functions and their derivatives for the FPM taking as a starting point a given cloud of points has been presented. Our approach reaches a satisfactory approximation focusing only on the WLSQ problem and thus placing no emphasis on getting an adequate local point distribution. The QR factorization based approximation shows more accuracy and robustness than the usual approximation procedure, based on matrix $\boldsymbol{A}$ inversion, and is more adequate to deal with non-appropriate local distributions of points. In addition, the shape of the weighting function has demonstrated to have very important effects on the approximation characteristics and seems to be a good choice to improve the local approximation quality.

Meshless methods such as the FPM permit to construct high-order approximations with a reasonable computational cost and this fact constitutes one of their main advantages over conventional mesh-based methods. Some three-dimensional numerical tests have been presented in order to point out certain features of the high-order FPM. The results obtained are encouraging. However, certain non-expected features springing from the high-order approximations related, in general, to particular settings of the approximation parameters must be studied in more detail. Finally, the ability of the FPM to undertake realistic calculation problems in competitive CPU times has been satisfactorily demonstrated.

Future work on Finite Point approximations is highly promising. Research areas such as effective local cloud generation techniques, $h - p$ adaptability and efficient high-order approximations should be worked upon in order to exploit the Finite Point Method potential for practical 3D applications.

Acknowledgments The first author would like to acknowledge the support of Alβan Program, the European Union Programme of High Level Scholarships for Latin America, scholarship N° E04D027284AR. The contributions of Prof. Carlos Sacco to this work are also gratefully acknowledged.

References

1. Oñate E, Idelsohn S, Zienkiewicz OC, Taylor RL, Sacco C (1996) A finite point method for analysis of fluid mechanics problems. Applications to convective transport and fluid flow. Int J Numer Methods Eng 39:3839–3866
2. Oñate E, Idelsohn S, Zienkiewicz OC, Fisher T (1995) A finite point method for analysis of fluid flow problems. In: Proceedings of the 9th international conference on finite elements methods in fluids. Venize, Italy, pp 15–21
3. Oñate E, Idelsohn S, Zienkiewicz OC, Taylor RL, Sacco C (1996) A stabilized finite point method for analysis of fluid mechanics problems. Comput Methods Appl Mech Eng 139:315–346
4. Oñate E, Sacco C, Idelsohn S (2000) A finite point method for incompressible flow problems. Comput Vis Sci 3:67–75
5. Fischer TR (1996) A contribution to adaptive numerical solution of compressible flow problems. Doctoral Thesis, Universitat Politècnica de Catalunya
6. Sacco C (2001) Desarrollo del Método de Puntos Finitos en mecánica de fluidos. Doctoral Thesis, Escola Tècnica Superior d'Enginyers de Camins, Canals i Ports de Barcelona, Universitat Politècnica de Catalunya
7. Löhner R, Sacco C, Oñate E, Idelsohn S (2002) A finite point method for compressible flow. Int J Numer Methods Eng 53:1765–1779
8. Oñate E, Perazzo F, Miquel J (2001) A finite point method for elasticity problems. Comput Struct 79:2151–2163
9. Lancaster P, Salkauskas K (1981) Surfaces generated by moving least squares methods. Math Comput 37:141–158
10. Belytschko T, Krongauz Y, Organ D, Fleming M, Krysl P (1996) Meshless methods: an overview and recent developments. Comput Methods Appl Mech Eng 139:3–47
11. Fries T, Matthies H (2004) Classification and overview of meshfree methods. Department of Mathematics and Computer Science, Technical University of Braunschweig. Inf 2003–3

12. Günther FC (1998) A meshfree formulation for the numerical solution of the viscous compressible Navier-Stokes equations. Dissertation, Northwestern University, Evanston
13. Xiaozhong J, Gang L, Aluru NR (2004) Positivity conditions in meshless collocation methods. Comput Methods Appl Mech Eng 193:1171–1202
14. Oñate E (1998) Derivation of stabilized equations for numerical solution of advective-diffusive transport and fluid flow problems. Comput Methods Appl Mech Eng 151:233–265
15. Boroomand B, Tabatabaei AA, Oñate E (2005) Simple modifications for stabilization of the finite point method. Int J Numer Methods Eng 63:351–379
16. Demmel JW (1997) Applied numerical linear algebra. Soc Ind Appl Math
17. Han W, Meng X (2001) Error analysis of the reproducing kernel particle method. Comput Methods Appl Mech Eng 190:6157–6181
18. Liu WK, Li S, Belytschko T (1997) Moving least square reproducing kernel methods. (I) Methodology and convergence. Comput Methods Appl Mech Eng 143:113–154
19. Liszka TJ, Duarte CAM, Tworzydlo WW (1996) hp-Meshless cloud method. Comput Methods Appl Mech Eng 139: 263–288