INTERNATIONAL CENTER FOR NUMERICAL METHODS IN ENGINEERING

A Finite Point Method for Three-Dimensional Compressible Flow

E. Ortega
E. Oñate
S. Idelsohn

Publication CIMNE Nº-310, September 2007

A Finite Point Method for Three-Dimensional Compressible Flow

E. Ortega
E. Oñate
S. Idelsohn

Publication CIMNE Nº-310, September 2007

International Center for Numerical Methods in Engineering
Gran Capitán s/n, 08034 Barcelona, Spain

Contents

1. INTRODUCTION

Numerical simulation came into the focus of interest of applied sciences and engineering in the last decades. As a result, the development of numerical techniques for solving partial differential equations (PDEs) has been growing continuously, mainly stimulated by increasing computational resources and ever-challenging demands for practical and theoretical applications. Basically, these numerical techniques reduce the original governing differential equation, or a mathematically equivalent form, into an algebraic system of equations, which is easily solved with a computer. A numerical technique for solving PDEs is set apart from another by the way in which the unknown function and its derivatives are approximated, and this is intimately related to the discretization of the physical domain. Regarding domain discretization, most numerical techniques for solving PDEs could be roughly classified into mesh-based methods or meshless methods. The domain discretization in mesh-based methods consists of a list of points or nodes properly ordered by the definition of connectivities between them. These connectivities originate the cells or elements which compose the mesh. Mesh-based methods like Finite Differences (FD), Finite Volumes (FV) and Finite Elements (FE) are usually employed in practice due to their robustness, efficiency and high confidence gained through years and years of continuous use and enhance. In meshless methods, the domain discretization is based on a list of points but an ordered connectivity between them is not required. This fact turns meshless techniques conceptually attractive but their practical implementations are not likely to be so, which explains the relatively little interest that has been devoted to these methods. However, over the last ten years, some difficulties in mesh-based or conventional methods when performing particular applications, have brought meshless methods into focus.

The first meshless methods appeared in the mid-seventies and numerous formulations have been proposed since then. A retrospective view of the evolution of the most relevant meshless methods as well as their connections is presented by Belytschko *et al.* [1]. In their work, the main features of some typical meshless methods, their implementation issues and practical applications are offered. Another interesting work by Fries and Matthies [2] classifies and analyzes the most important meshless methods. For each, the authors highlight the main characteristics and implementation details as well as the advantages and disadvantages. Other helpful reviews on meshless methods, also considered here, can be found in the literature; see for instance Li and Liu [3] and Duarte [4].

The present work deals with a meshless technique called Finite Point Method (FPM) which was introduced by Oñate *et al*. [5]. In this method, the numerical approximation to the unknown function and its derivatives is based on a Weighted Least-Squares (WLSQ) procedure known as Fixed Least Squares (FLS). The strong form of the governing PDEs is sampled at each point by replacing the continuous variables with their approximated counterparts. Finally, the resulting system of algebraic equations is obtained by means of a collocation technique.

Next, a review of the development of the FPM, aimed at providing a framework for the present research work and its motivations, is given. Then, by the end of this section, the objectives of this research and the organization of the contents are presented.

1.1 Preliminary remarks on the Finite Point method.

Since the Finite Point Method appeared in the literature towards the mid-nineties, it has been successfully applied to solve convective-diffusive problems, incompressible and compressible fluid flow problems [6, 7, 8, 9, 10, 11] and solid mechanics problems [12] among others.

As regards fluid flow problems, the first application of the FPM to the solution of the bi-dimensional compressible flow equations has been presented by Oñate *et al.* in [5, 6] and Fischer in [9]. In these works, certain topics such as the construction of local clouds of points and the effects of weighting functions on the numerical approximation have been studied using first and second-order approximation bases. Later, Sacco [10] presented a detailed analysis of the Finite Point (FP) approximation in conjunction with a multi-dimensional application for solving the incompressible flow equations. Outstanding achievements from that work, for instance, a definition of local and normalized approximation bases, a procedure for constructing local clouds of points and a criterion for evaluating their quality have given FPM a more solid base. In relation to the solution of the incompressible flow equations, a fractional step algorithm stabilized through a technique known as Finite Increment Calculus (FIC) [13] has also been successfully employed.

The FP solution of the three-dimensional compressible flow equations has been presented in a pioneer work by Löhner *et al*. [11]. Here, two remarkable contributions are worth of mention: a reliable procedure for constructing the local clouds (based on a Delaunay technique) and a well-suited scheme for solving the flow equations. This scheme is based on a ‘symmetrized’ discrete expression of the advective flux-divergence vector, which is composed of a central difference-like expression plus a corrective term. Then, the central difference-like flux term is

replaced by an upwind numerical flux obtained through an approximate Riemann solver. In the meshless context, this approach is best suited than artificial dissipation methods because it is not necessary to define any kind of geometrical measure in the cloud of points.

All the works listed above have made different but remarkable contributions to enhance the performance of the FPM, not only giving evidence of its potential but also revealing some of its weaknesses. Nowadays, most meshless methods, and in particular the WLSQ-based methods, present a lack of solid theoretical and practical arguments regarding local cloud construction, approximation bases selection and weighting function setting among other important issues. In addition, methods like the FPM, which work with the strong form of the differential governing equations must face some other well-known problems arising from the collocation procedure. Unfortunately, the accuracy of the numerical approximation in the cloud of points is absolutely dependent on the features just mentioned. From another point of view, competitive meshless methods are in need of a considerable reduction of computational cost, which requires developing more efficient algorithms and data structures. All these considerations become crucial when it is necessary to deal with real three-dimensional problems of practical application in engineering. Consequently, robustness and efficiency seem to be the key to the success of any meshless method.

As regards robustness, some modifications to the FPM have been proposed by Boroomand *et al*. [14] with the aim of reducing instabilities in the minimization procedure, especially those arising from non-appropriate local clouds of points. In addition to that, but from another perspective, we have recently presented an alternative approach towards robustness [15]. This *ad hoc* procedure, which is based on a QR factorization in conjunction with an iterative adjustment of the local approximation parameters, allows obtaining a satisfactory minimization problem solution and avoids modifying the local geometrical support where the approximation is based on. This QR-based procedure has demonstrated that it can reduce the approximation dependence on both, the spatial distribution of points and the shape of the weighting function significantly.

The most salient remarks on the FPM have been set forth. Even though this brief review refers only to the FPM, most meshless techniques share this situation and much work is still to be done to make them competitive. This fact it what motivates the present research.

1.2 Objectives

The present work is intended to contribute to a major investigation into the capabilities of the Finite Point method to deal with three-dimensional applications concerning compressible fluid flow problems. Consequently, as a first step towards this ultimate aim, the development of a suitable Finite Point formulation for this kind of analysis is the main objective of the present research. In addition, an adaptive procedure for FP calculations is presented in the last part of this work. Even though this is not one of the main objectives, we think that starting to exploit the full potential of the FPM is also worth of consideration.

1.3 Organization of the present work

The contents of the present research work are organized as follows. In Section 2 the FP approximation procedure is carefully examined emphasizing the effects of the weighting function setting on the numerical approximation. As a result, an alternative procedure aimed at obtaining a satisfactory WLSQ problem solution where the usual approach fails is proposed in Sections 3 and 4. Section 5 is concerned with domain discretization and the construction of local clouds of points. Next, the behaviour of high-order FP approximations is explored by means of some numerical examples in Section 6. Concerning the solution of the three-dimensional compressible flow equations, an upwind biased scheme is developed in Section 7. Then, in Section 8, several numerical calculations are provided to show the performance of the flow solver. Section 9 proposes an *h*-adaptive technique for compressible flow calculations and, by the end of this section, the performance of the proposed adaptive methodology is evaluated by means of some numerical examples. After that, in Section 10, the conclusions we have reached and some investigation lines to be followed are presented.

2. APPROXIMATION IN THE FINITE POINT METHOD

An approximation to an unknown function $u(\boldsymbol{x})$ defined in a closed domain $\Omega \in \Re^{\mathrm{d}}$ (d=1, 2 or 3) which is discretized by a set of points $\boldsymbol{x}_{\mathrm{i}}$, $i = 1, n$ is developed. In order to obtain a local approximation for function $u(\boldsymbol{x})$, the domain Ω is divided into subdomains Ω_{i} (henceforth *clouds of points*) so that $\Sigma\Omega_{\mathrm{i}}$ represents a covering for Ω. Each cloud of points consists of a point $\mathbf{x}_{\mathrm{i}}$ called *star point* and a set of points $\boldsymbol{x}_{\mathrm{j}}$, $j = 2, 3, ..., np$ surrounding $\boldsymbol{x}_{\mathrm{i}}$, which complete the cloud Ω_{i}. Assuming that the function $u(\boldsymbol{x})$ is smooth enough in Ω_{i}, it is possible to state the following approximation

$$u(\boldsymbol{x}) \cong \hat{u}(\boldsymbol{x}) = \sum_{l=1}^{m} p_l(\boldsymbol{x})\,\alpha_l = \boldsymbol{p}^{\mathrm{T}}(\boldsymbol{x})\,\boldsymbol{\alpha} \tag{1}$$

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

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

Next, at each point $\boldsymbol{x}_j \in \Omega_i$ the unknown function is obtained as follows

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

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

$$\mathbf{P} = \begin{bmatrix} \boldsymbol{p}_1^T \\ \vdots \\ \boldsymbol{p}_{np}^T \end{bmatrix} = \begin{bmatrix} p^1(\boldsymbol{x}_1) & p^2(\boldsymbol{x}_1) \dots p^m(\boldsymbol{x}_1) \\ \vdots \\ p^1(\boldsymbol{x}_{np}) & p^2(\boldsymbol{x}_{np}) \dots p^m(\boldsymbol{x}_{np}) \end{bmatrix} \quad (np\times m) \tag{4}$$

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, $np \geq m$ is adopted and the equation system becomes overdetermined. Consequently, an approximate solution is sought by means of a WLSQ technique. This solution minimizes a discrete L_2 error norm in the approximation to $u(\boldsymbol{x})$ 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 on the star point of the cloud, is chosen so that it satisfies the following conditions

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

This kind of approximation, known as Fixed Least-Squares method (FLS), can be considered as a particular case of the Moving Least-Squares Method (MLS) introduced by Lancaster and Salkauskas in the context of interpolation and data fitting [16]. 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 $J(\boldsymbol{x}_i)=J_i$ is defined

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

in which $\varphi_i(\boldsymbol{x}_j)=\varphi(\boldsymbol{x}_j-\boldsymbol{x}_i)$ is a compact support weighting function. Eq. (6) can be rewritten as

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

where $\boldsymbol{\phi}(\boldsymbol{x})$ = diag($\varphi(\boldsymbol{x}_j-\boldsymbol{x}_i)$). The minimization of Eq. (7) with respect to $\boldsymbol{\alpha}$ leads to the following equation system

$$\left(\mathbf{P}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})\,\mathbf{P}\right)\boldsymbol{\alpha} - \left(\mathbf{P}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})\right)\boldsymbol{u}^h = \mathbf{0} \tag{8}$$

known as *normal equations* in the Least-Squares (LSQ) literature. Introducing the matrices

$$\begin{aligned} &\mathbf{A} = \left(\mathbf{P}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})\,\mathbf{P}\right) \quad , \quad A_{kl} = \textstyle\sum_{j=1}^{np} \varphi_i(\boldsymbol{x}_j)\, p_k(\boldsymbol{x}_j)\, p_l(\boldsymbol{x}_j) \quad (m\times m) \\ &\mathbf{B} = \left(\mathbf{P}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})\right) \quad , \quad B_{lj} = p_l(\boldsymbol{x}_j)\,\varphi_i(\boldsymbol{x}_j) \quad (m\times np) \end{aligned} \tag{9}$$

it is possible to express the normal equations (8) as follows

$$\mathbf{A}\,\boldsymbol{\alpha} = \mathbf{B}\,\boldsymbol{u}^h \tag{10}$$

Due to the fact that a fixed weighting function is chosen, the unknown coefficients α_j are constant in Ω_i. These coefficients can be found by

$$\boldsymbol{\alpha} = \mathbf{A}^{-1}\mathbf{B}\,\boldsymbol{u}^h \tag{11}$$

If the columns of matrix **P** are linearly independent, matrix **A** is positive-definite and, consequently, non-singular. Then, the unknown coefficients α_j are uniquely determined by Eq. (11). However, depending on the spatial distribution of the cloud of points (and especially in 2-D and 3-D problems) matrix **A** can become ill-conditioned, making it very difficult to invert the matrix with accuracy. For cases like this, an alternative procedure for solving the LSQ problem is presented later in Section 3. Finally, replacing the unknown coefficients (11) in Eq. (1), the approximation to the unknown function at the star point is obtained as follows

$$\hat{u}(\boldsymbol{x}_i) = \underbrace{\boldsymbol{p}^{\mathrm{T}}(\boldsymbol{x}_i)\,\mathbf{A}^{-1}\,\mathbf{B}}_{\mathbf{N}_i^{\mathrm{T}}(\boldsymbol{x})\quad(1\times np)}\,\boldsymbol{u}^h \tag{12}$$

where $\mathbf{N}_{\mathrm{i}}^{\mathrm{T}}(\boldsymbol{x}) = \left[N_{i,1}, N_{i,2}, ..., N_{i,np} \right]$ is the shape function vector of the point $\boldsymbol{x}_{\mathrm{i}}$ in Ω_{i}. The adoption of an FLS scheme, where matrices **A** and **B** are constant in Ω_{i}, simplifies the calculation of the shape functions derivatives. Consequently,

$$\frac{\partial^l \mathbf{N}_{\mathrm{i}}^{\mathrm{T}}(\boldsymbol{x})}{\partial \boldsymbol{x}_k^l} = \frac{\partial^l \boldsymbol{p}^{\mathrm{T}}(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_k^l} \mathbf{A}^{-1}\mathbf{B} \tag{13}$$

and the approximation to the unknown function derivatives at $\boldsymbol{x}_{\mathrm{i}}$ are given by

$$\frac{\partial^l \hat{u}(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_k^l} = \frac{\partial^l \mathbf{N}_{\mathrm{i}}^{\mathrm{T}}(\boldsymbol{x})}{\partial \boldsymbol{x}_k^l} \boldsymbol{u}^h = \frac{\partial^l \boldsymbol{p}^T(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_k^l} \mathbf{A}^{-1}\mathbf{B}\, \boldsymbol{u}^h \tag{14}$$

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 a numerical method has 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 [1]. Following this approach, it is considered that a set of functions **N**(***x***) has *p*-order consistency if the following conditions are satisfied

$$\begin{aligned} &\mathbf{N}_i^{\mathrm{T}}(\boldsymbol{x})\boldsymbol{p}(\boldsymbol{x}_j) = p(\boldsymbol{x}_i) \\ &\frac{\partial^l \mathbf{N}_i^{\mathrm{T}}(\boldsymbol{x})}{\partial \boldsymbol{x}_k^l} \boldsymbol{p}(\boldsymbol{x}_j) = \frac{\partial^l p(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_k^l} \qquad \nabla \boldsymbol{x}_i \in \Omega \end{aligned} \tag{15}$$

where **p**(***x***) is a complete polynomial base of order *p* [2]. For the MLS approximation it was found that if the base is complete of order *p*, then consistency of order *p* is obtained. It can also be demonstrated that any function in the base can be exactly reproduced [2].

Due to the fact that, in the FLS scheme adopted here the shape function and its derivatives are discontinuous, it is *only* possible to satisfy the consistency requirements (15) in the cloud's star point $\mathbf{x}_{\mathrm{i}}$ where the weighting function is located.

2.2 The approximation bases

In this work the following complete polynomial bases in $\Re^{\mathrm{d}}$ are used:

- 2^{nd} order approximation bases

$$\begin{aligned} &2\text{-D}: \quad \boldsymbol{p}^T(\boldsymbol{x}) = \left[1, x, y, xy, x^2, y^2\right] && m = 6 \\ &3\text{-D}: \quad \boldsymbol{p}^T(\boldsymbol{x}) = \left[1, x, y, z, xy, xz, yz, x^2, y^2, z^2\right] && m = 10 \end{aligned} \tag{16}$$

- 3rd order approximation base, three-dimensional case ($m = 20$)

$$\boldsymbol{p}^T(\boldsymbol{x}) = \left[1, x, y, z, x^2, xy, xz, y^2, yz, z^2, x^3, x^2y, x^2z, xy^2, xyz, xz^2, y^3, y^2z, yz^2, z^3\right] \quad (17)$$

- 4th order approximation base, three-dimensional case ($m = 35$)

$$\begin{aligned}\boldsymbol{p}^T(\boldsymbol{x}) = \Big[&1, x, y, z, x^2, xy, xz, y^2, yz, z^2, x^3, x^2y, x^2z, xy^2, xyz, xz^2, y^3, y^2z, yz^2, z^3, \\ &x^4, x^3y, x^3z, x^2y^2, x^2yz, x^2z^2, xy^3, xy^2z, xyz^2, xz^3, y^4, y^3z, y^2z^2, yz^3, z^4\Big]\end{aligned} \quad (18)$$

With the aim of alleviating ill-conditioning problems arising from the matrix **P**, local and normalized bases are defined [6] by means of

$$x = \frac{x_j - x_i}{d_{\max}} \quad , \quad y = \frac{y_j - y_i}{d_{\max}} \quad , \quad z = \frac{z_j - z_i}{d_{\max}} \quad (19)$$

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

The introduction of the local and normalized approximation bases defined by Eq. (19) simplifies the computation of the shape function vector and their derivatives at the star point $\mathbf{x}_i$ significantly. Note that the terms x, y and z given by Eq. (19) vanish when they are evaluated at the star point of the cloud. Consequently, the approximation bases (16)-(18) and their derivatives become

$$\begin{aligned}\boldsymbol{p}^T(\boldsymbol{x}_i) &= [1, 0, 0, \ldots, 0] \\ \frac{\partial \boldsymbol{p}^T(\boldsymbol{x}_i)}{\partial x} &= \left[0, \frac{1}{d_{\max}}, 0, \ldots, 0\right] \quad , \quad \ldots\end{aligned} \quad (20)$$

and, it is possible to obtain a similar result for the other spatial directions and the higher-order derivatives of the approximation base. This simplification leads to the following expressions

$$\begin{aligned}\mathbf{N}_i^T(\boldsymbol{x}) &= \mathbf{C}_{1,j} \\ \frac{\partial^k \mathbf{N}_i^T(\boldsymbol{x})}{\partial \boldsymbol{x}^k} &= \frac{k!}{\left(d_{\max}\right)^k} \mathbf{C}_{(k+1),j}\end{aligned} \quad (21)$$

where $\mathbf{C} = \mathbf{A}^{-1}\mathbf{B}$ is a $(\mathrm{m \times np})$ matrix and index $j = 1, np$.

2.3 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 and, consequently, 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 the Finite Point method the following normalized Gaussian weighting function is adopted

$$\varphi_i(\boldsymbol{x}_j) = \frac{e^{-\left(d_j/\alpha\right)^k} - e^{-\left(\beta/\alpha\right)^k}}{1 - e^{-\left(\beta/\alpha\right)^k}} \tag{22}$$

where $d_j = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$, $\alpha = \beta/w$ and $\beta = \gamma\, d_{max}$ (γ>1.0). The support of this function is isotropic, circular in 2-D and spherical in 3-D. The parameters w, k and γ govern the functional shape of the weighting function. These are free parameters that should be properly set because the final approximation characteristics 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 properties through a variation of the functional form of the weighting function is a helpful tool to deal with the numerical discretization of complete geometries. Next, a brief analysis of the free parameters involved in the weighting function (22) and their relation with the numerical approximation is outlined with the aim of identifying the capabilities of these parameters to improve the local approximation.

Parameter γ gives more or less weight to the boundary points of the cloud by increasing or decreasing the size of the weighting function's support. Numerical experiments show that the variation of parameter γ has a minor effect on the local approximation when $np \approx m$ and this effect becomes negligible when the number of points in the cloud is increased. The error in the approximation to the unknown function tends to become higher when γ is increased while the error in the approximation to the derivatives of the unknown function becomes smaller. Anyway, in general, these effects are not relevant when appropriate clouds of points are considered and the parameter γ can be set to a constant value, e.g. $\gamma = 1.01$, in the whole domain Ω. Note, however, that parameter γ determines the size of the weighting function's support and, in consequence, an increase in the parameter γ could be interpreted as an enlargement of the overlapping zone between neighbouring clouds of points. This provides a mechanism to improve the approximation quality when sudden changes in the distance between neighbouring points happen in the analysis domain. In these cases, good results are obtained setting $1 < \gamma \leq 1.25$.

Figure 1 shows the effect of the parameter γ on the approximation when appropriate clouds of points are considered. 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 isolated cloud centered on the domain. A complete description of this problem will be presented later in Section 6.

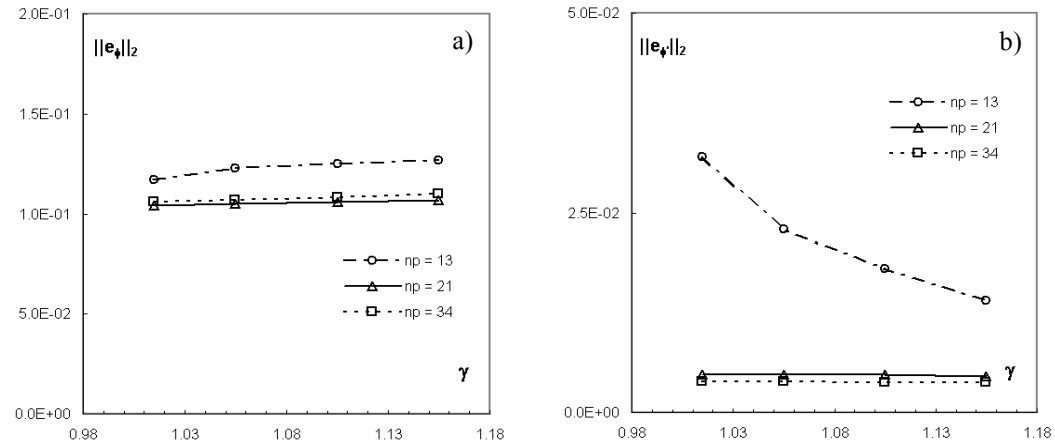

Figure 1: Effect of the parameter γ on the L_2-norm of the approximation error for clouds with a different number of points (np); w=3.5 and 2nd order approximation bases: a) the variable ϕ, b) the first derivatives (average)

The next free parameter in expression (22) 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 that it decreases the weight of the boundary points or viceversa; see Figure 2(b). The effect of the parameter k 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 this purpose. Therefore, the 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 weight 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 Figure 2(a).

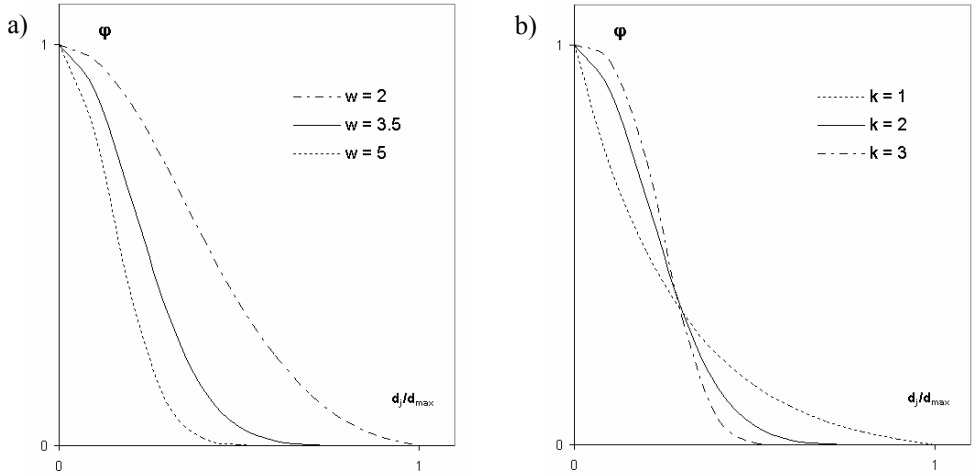

Figure 2: Effects of the parameters *w* and *k* on the weighting function shape: a) Effect of parameter *w*, $\gamma = 1.01$ and $k = 2$; b) Effect of parameter *k*, $\gamma = 1.01$ and $w = 3.5$

For large values of *w*, the shape function tends to the Dirac's delta function (see Figure 2 (a)) and the approximation procedure tends to interpolate nodal data. When *w* is increased, the values of $N_j(\mathbf{x}) \rightarrow 0$, except at the star point where $N_j(\mathbf{x}) \rightarrow 1$, i.e. the shape function also tends to the Dirac's delta function. This causes the error in the approximation to the unknown function to decrease and the condition number of matrix **A** ($\kappa(\mathbf{A})$) to increase. As a result, while *w* increases the problem becomes more and more ill-conditioned. Beyond a given threshold it is not possible to invert matrix **A** with accuracy, the approximation quality deteriorates quickly and numerical instabilities appear. In Figure 3 the effect of *w* on the numerical approximation and the condition number of matrix **A** is shown for the same Poisson's test problem presented in Figure 1. The results displayed belong to an isolated homogeneous cloud centered on the analysis domain.

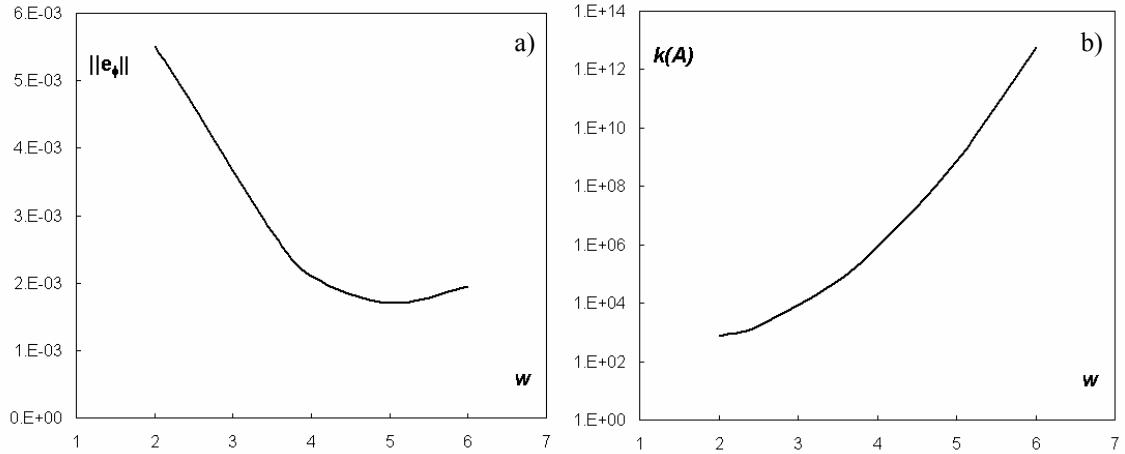

Figure 3: a) effect of parameter *w* on the L_2-norm of the error in the approximation to the variable ϕ, b) effect of *w* on the condition number of matrix **A**; np = 35 and 2nd order approximation bases.

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

Taking into account some numerical experiments, an admissible range for parameter w given by $3.0 \leq w_{max} \leq 4.5$ is found for three-dimensional computation cases. 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 its derivatives. We will go back to this point later in Section 3.

2.4 Discretization of the equations

In the FLS method the weighting function is fixed at the star point of the cloud and this fact leads to multivalued shape functions depending on the cloud in which the approximation is calculated, i.e. $\mathbf{N}_{\mathrm{n}}(\boldsymbol{x}_j) \neq \mathbf{N}_{\mathrm{m}}(\boldsymbol{x}_j)$ (where indices m and n indicate neighbouring clouds of points). Therefore, the numerical approximation is globally and locally discontinuous and must be only considered as valid in the star point of the cloud $\boldsymbol{x}_{\mathrm{i}}$ where the weighting function is located. Hence, a collocation technique becomes the natural choice in the FPM.

Collocation procedures are simple and easy to implement; however, special care must be taken of the resultant global equation systems 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 a biased selection of the cloud's points or through a local manipulation of the weighting function [17]. Robustness of the collocation procedure can be also improved introducing stabilization terms into the governing equations, in particular for Neumann boundary points. See for instance [5, 7] and a similar approach presented in [14].

Following the lines given by [5] and [7], numerical instabilities in the calculations presented in Section 4 are avoided by solving a modified set of governing equations obtained via the so-called Finite Calculus technique (see [13] for details). Concerning compressible flow problems, no special care is taken of the collocation procedure. Due to the fact that the

proposed scheme is fully explicit, the only system of equations to be solved is a well-conditioned system similar to the *mass-consistent* systems arising from FE discretizations.

3. COMPUTATION OF THE SHAPE FUNCTIONS PARAMETERS

According to the FPM approximation methodology presented before, in order to get the unknown coefficients α_j and, consequently, the shape functions and their derivatives for a given cloud of points, the following linear system must be solved

$$\mathbf{A}\boldsymbol{\alpha} = \mathbf{B}\boldsymbol{u}^h \tag{23}$$

It should be noticed that this system must be solved via the inversion of matrix **A** because the vector $\mathbf{u}^h$ is not known in advance. This methodology is not the most accurate for solving LSQ problems, especially when the condition number of matrix **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 **A** are similar to Vandermonde type matrices causing matrix **A** to be *naturally* ill-conditioned. Obtaining an accurate solution of the system (23) is a key task in most meshless methods based on LSQ approximations.

Introducing matrices **A** and **B** defined by (9) in (23), the normal equations are recovered

$$\left(\mathbf{P}^{\mathrm{T}}\phi(\boldsymbol{x})\mathbf{P}\right)\boldsymbol{\alpha} = \left(\mathbf{P}^{\mathrm{T}}\phi(\boldsymbol{x})\right)\boldsymbol{u}^h \tag{24}$$

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

$$\boldsymbol{\alpha} = \left(\mathbf{P}^{\mathrm{T}}\phi(\boldsymbol{x})\mathbf{P}\right)^{-1}\left(\mathbf{P}^{\mathrm{T}}\phi(\boldsymbol{x})\right)\boldsymbol{u}^h \tag{25}$$

It is possible to prove that if matrix **P** has rank *m*, i.e. all their columns are linearly independent, matrix $\mathbf{A} = \mathbf{P}^{\mathrm{T}}\phi(\boldsymbol{x})\mathbf{P}$ is positive-definite and, consequently, non-singular (note that matrix $\phi(\boldsymbol{x})$ is positive-definite by definition (5)). Then, the inverse matrix exists and the unknown coefficients are uniquely determined.

The solution of the equations (25) by direct inversion of matrix **A** must be restricted to cases when the condition number of matrix **A** is moderate. Generally, when the condition number of matrix **A** is large, its inverse is not appropriate to calculate the shape function and its 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 its derivatives is the following. Given a certain cloud of points, first, the direct inversion of matrix **A** is attempted.

If the condition number of **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 (24) 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 the WLSQ problem when matrix **A** is ill-conditioned. The aim of using a QR factorization technique is to get an acceptable solution in cases where the other procedure fails without having to modify the geometrical structure of the cloud. The computational cost of the WLSQ problem solution via QR factorization may cost, in terms of time, up to twice as much as the solution via matrix **A** inversion if $np \gg m$ [18]. However, this extra amount of time is not important in the overall time because the alternative QR-based procedure is only applied to problematic clouds of points, which represent only a small percentage of the whole clouds in the domain.

If matrix **P** has rank m and $np > m$, it can be uniquely factored as

$$\mathbf{P} = \mathbf{Q}\,\mathbf{R} \tag{26}$$

where matrix $\mathbf{Q} \in \Re^{npxm}$ is orthogonal ($\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$) and matrix $\mathbf{R} \in \Re^{mxm}$ 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 **P** is rank deficient or near rank deficient.

In order to apply the QR factorization for solving the WLSQ problem (24), it is necessary to obtain an equivalent unweighted problem for which the next factorization is proposed

$$\tilde{\phi}(\boldsymbol{x}) = \sqrt{\phi(\boldsymbol{x})} \quad \text{such that} \quad \tilde{\boldsymbol{\phi}}^T\tilde{\boldsymbol{\phi}} = \boldsymbol{\phi} \tag{27}$$

and also the following modification of matrix **P**

$$\tilde{\mathbf{P}} = \tilde{\boldsymbol{\phi}}\mathbf{P} \tag{28}$$

After that, it is possible to write an equation system equivalent to the one given by Eq. (24) as

$$\left(\tilde{\mathbf{P}}^T\tilde{\mathbf{P}}\right)\boldsymbol{\alpha} = \left(\tilde{\mathbf{P}}^T\tilde{\boldsymbol{\phi}}\right)\boldsymbol{u}^h \tag{29}$$

and the equivalence between Eq. (29) and (24) is verified by

$$\tilde{\mathbf{P}}^{\mathrm{T}}\tilde{\mathbf{P}}\boldsymbol{\alpha} = \tilde{\mathbf{P}}^{\mathrm{T}}\tilde{\boldsymbol{\phi}}\boldsymbol{u}^{h}$$
$$(\tilde{\boldsymbol{\phi}}\mathbf{P})^{\mathrm{T}}\tilde{\boldsymbol{\phi}}\mathbf{P}\boldsymbol{\alpha} = (\tilde{\boldsymbol{\phi}}\mathbf{P})^{\mathrm{T}}\tilde{\boldsymbol{\phi}}\,\boldsymbol{u}^{h}$$
$$\mathbf{P}^{\mathrm{T}}(\tilde{\boldsymbol{\phi}}^{T}\tilde{\boldsymbol{\phi}})\mathbf{P}\boldsymbol{\alpha} = \mathbf{P}^{\mathrm{T}}(\tilde{\boldsymbol{\phi}}^{\mathrm{T}}\tilde{\boldsymbol{\phi}})\boldsymbol{u}^{h}$$
$$\mathbf{P}^{\mathrm{T}}(\boldsymbol{\phi})\mathbf{P}\boldsymbol{\alpha} = \mathbf{P}^{\mathrm{T}}(\boldsymbol{\phi})\boldsymbol{u}^{h}$$

Then, the modified matrix (28) is factorized, i.e. $\tilde{\mathbf{P}} = \mathbf{QR}$, and replaced in the equivalent unweighted problem (29). This leads to

$$\begin{aligned}(\mathbf{QR})^{\mathrm{T}}\mathbf{QR}\boldsymbol{\alpha} &= (\mathbf{QR})^{\mathrm{T}}\tilde{\boldsymbol{\phi}}\,\boldsymbol{u}^{h}\\ \mathbf{R}^{\mathrm{T}}(\mathbf{Q}^{\mathrm{T}}\mathbf{Q})\mathbf{R}\boldsymbol{\alpha} &= \mathbf{R}^{\mathrm{T}}\mathbf{Q}^{\mathrm{T}}\tilde{\boldsymbol{\phi}}\,\boldsymbol{u}^{h}\end{aligned} \tag{30}$$

where $\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{I}$ due to the orthogonality property. Multiplying both sides by $(\mathbf{R}^{\mathrm{T}})^{-1}$, we get

$$\mathbf{R}\boldsymbol{\alpha} = \mathbf{Q}^{\mathrm{T}}\tilde{\boldsymbol{\phi}}\,\boldsymbol{u}^{h} \tag{31}$$

from which the unknown coefficients α_j can be finally obtained

$$\boldsymbol{\alpha} = \mathbf{R}^{-1}\left(\mathbf{Q}^{\mathrm{T}}\tilde{\boldsymbol{\phi}}\right)\boldsymbol{u}^{h} \tag{32}$$

Here matrix **R** is generally well-conditioned and its inverse is easy to obtain with accuracy, even for the cases when matrix **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 **A**. This fact reduces the dependence of the approximation on the spatial distribution of points and on the functional shape of the weighting function significantly, giving robustness to the Finite Point methodology.

4. AN ITERATIVE PROCEDURE FOR CALCULATING THE SHAPE FUNCTIONS

With the aim of obtaining an appropriate approximation in a given cloud of points, the following iterative procedure is proposed. First, a maximum value of the parameter *w* in the weighting function is set according to $w = w_{\max} \approx 3.5$ and the WLSQ problem is solved via matrix **A** inversion (11). Then, the shape function and their derivatives are obtained by Eq. (21). The resulting approximation is accepted if it satisfies the following requirements:

r₁. $\kappa(\mathbf{A}) \le \kappa_{\max}(\mathbf{A})$

r₂. $\sum_j N_i(x_j) - 1.0 \le \varepsilon$ and $\sum_j \frac{\partial N_i(x_j)}{\partial \boldsymbol{x}} \le \varepsilon$

r₃. Consistency

The first requirement (r_1) imposes a limit to the condition number of matrix **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 fulfilment of the second condition is essential for the implementation of the flow solver that we propose later. The last requirement (r_3) enables the verification of the approximation accuracy by checking the consistency requirements (15) at cloud's star point $\mathbf{x}_i$. To achieve this, it is also possible to replace the approximation base by a known function and assess the approximated values deviation from the exact values [11]. The values adopted for setting $\kappa_{max}(\mathbf{A})$, the parameter ε and the admissible error in the consistency check depend on the problem under consideration. In this work a value $\kappa_{max}(\mathbf{A})$=1.E6 based on the infinite norm of matrix **A** and the parameter ε = 1.E-10 are adopted. In order to check consistency requirements we follow the procedure suggested in [11].

In the case that one of the preceding requirements is not satisfied, the approximation is rejected and the solver changes to the QR factorization-based methodology (32) keeping all the approximation parameters constant. In general, the QR factorization allows obtaining a suitable approximation where the matrix **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 set 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 permit to obtain an appropriate numerical approximation, new points are inserted in the cloud and the described procedure starts again. It could be also possible to decrease the local order of approximation in the cloud of points but this option is not taken into consideration for the numerical calculations presented in this work.

5. DISCRETIZATION OF THE DOMAIN AND LOCAL CLOUDS CONSTRUCTION

An adequate support of points is essential for setting a good local approximation for each cloud. The quality of the local approximation highly 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 presented above 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 spatial 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 [19, 20]. Mixed geometrical-mathematical considerations are also employed. 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 [17, 14, 21]. All these criteria, often aimed at obtaining an *a priori* acceptable local support for the numerical approximation, lead to methods for the construction of the required point's connectivity. Concerning the FPM, a reliable methodology for constructing local clouds of points based on a Delaunay technique has been proposed by Löhner *et al.* [11]. In the present work we follow the general criteria proposed there.

5.1 Domain discretization

The point discretization of the analysis domain is obtained by means of a modification of the algorithm presented in [22]. It starts from a Delaunay triangulation that bounds the computational domain and inserts new points in the centre of empty spheres covering the domain. This extremely fast procedure originates an incremental quality triangulation known as *optimization driven point insertion* avoiding any subsequent smoothing of the discretization.

5.2 Local clouds construction

The local clouds of points are constructed as follows. Given a point discretization of the computational domain and a set of normal vectors belonging to the triangulation that bounds 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 $\mathbf{x}_i$, all neighbours within the search radius (r_s) are found through an octree technique. Any local cloud of points inside the computational domain is constructed taking the closest neighbouring points of the star point. However, if a star point $\mathbf{x}_i$ is located either over or close enough to a solid boundary, the

points included in its cloud (admissible points) must also satisfy the conditions described below.

Case 1: star point located over a solid boundary

In this particular case (sketched in Figure 4(a)), every point $\mathbf{x}_j$ located within the search radius is admissible if it meets the following conditions

$$\cos(\theta) \geq \cos\left(\pi/2 + \delta\right) \quad ; \quad \cos(\theta) = \frac{\boldsymbol{n}_i \cdot \boldsymbol{r}_j}{\|\boldsymbol{n}_i\| \|\boldsymbol{r}_j\|} \tag{33}$$

$$\left\|\boldsymbol{r}_j^t\right\| < \alpha\, r_{search} \tag{34}$$

Condition (33) defines an acceptation area 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 (34) imposes a certain aspect ratio in the cloud, given by the parameter $\alpha \neq 0$.

Case 2: cloud of points intercepting a solid boundary

In this case the point $\mathbf{x}_j$ located over a surface ($\boldsymbol{x}_{j_{nea}}$), nearest to the star point $\mathbf{x}_i$, must be sought (see Figure 4(b)). Then, every point within the search radius is admissible if

$$\cos(\theta) \geq \cos\left(\pi/2 + \delta\right) \quad ; \quad \cos(\theta) = \frac{\boldsymbol{n}_{j_{nea}} \cdot \boldsymbol{r}_j}{\|\boldsymbol{n}_{j_{nea}}\| \|\boldsymbol{r}_j\|} \tag{35}$$

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

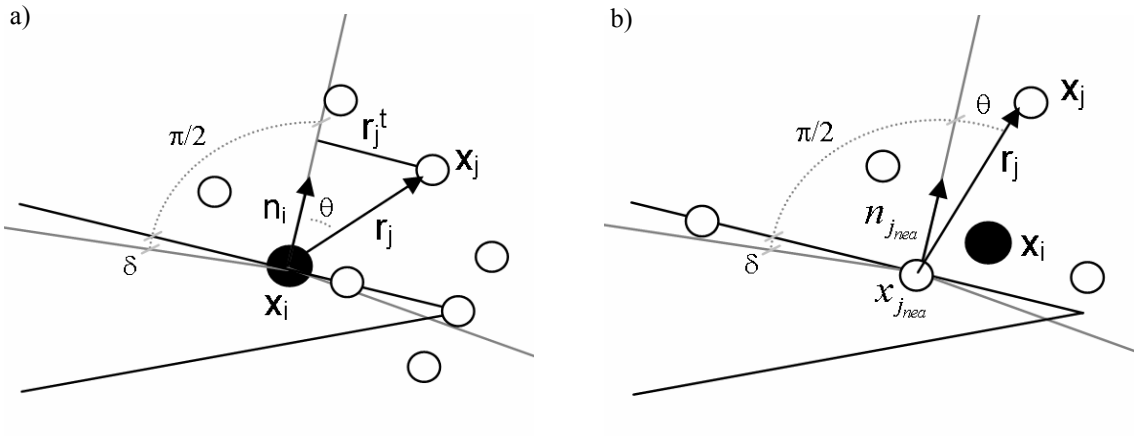

Figure 4: The construction of local clouds near the boundaries: a) The star point located over a solid boundary; b) A cloud of points intercepting a solid boundary.

If the number of admissible points found within the search radius is not enough, the latter is increased until condition $np_{min} \leq np \leq np_{max}$ is satisfied. Otherwise, if the number of admissible points goes beyond np_{max}, only the points nearest to x_i are added to the cloud.

It is very helpful to force the first layer of nearest neighbours of $\mathbf{x}_i$ into the local cloud of points when sudden variations in the distance between neighbouring points occur inside the analysis domain. For each star point this is accomplished by performing a local Delaunay grid with all the points falling within the octree search area. Only the first layer of nearest neighbours is retained and used to initialize the local cloud of points. Finally, admissible nearest points are added until the condition $np_{min} \leq np \leq np_{max}$ is fulfilled. This procedure, which follows the lines proposed in [11], avoids non-overlapping neighbouring clouds of points and improves the quality of the local discretization. Furthermore, the information concerning the first layer of neighbouring points for each star point is very useful to improve several computational procedures. In the present work such information is needed for the adaptive procedure that is presented in Section 9.

6. HIGH-ORDER APPROXIMATIONS. SOME PRELIMINARY RESULTS

In this Section some numerical examples are presented in order to assess the behaviour of high-order FP approximations. 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 proposed methodology is shown in a more realistic calculation case using second-order approximation bases.

6.1 Poisson's problem in a cubic domain

This example concerns a Poisson's problem

$$\begin{aligned} \nabla^2 \phi &= f(x,y,z) \quad \text{in } \Omega \\ \phi &= 0 \qquad\quad\ \ \text{on } \Gamma_D \end{aligned} \tag{36}$$

set in a unit length sides cubic domain Ω with Dirichlet boundary $\Gamma = \Gamma_D$. The source term *f(x,y,z)* is given by

$$\begin{aligned} f(x,y,z) = \{ & -2kyz(1-y)(1-z) + \left[kyz(1-y)(1-z)(1-2x)^2\right]^2 \\ & -2kxz(1-x)(1-z) + \left[kxz(1-x)(1-z)(1-2y)^2\right]^2 \\ & -2kyx(1-x)(1-y) + \left[kxy(1-x)(1-y)(1-2z)^2\right]^2 \ \} \frac{e^{kxyz(1-x)(1-y)(1-z)}}{1-e^{k/64}} \end{aligned} \tag{37}$$

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

$$\phi(x,y,z) = \frac{1 - e^{kxyz(1-x)(1-y)(1-z)}}{1 - e^{k/64}} \tag{38}$$

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

$$\|\varphi\|_2 = \left(\frac{\sum_{i=1}^{n} (\varphi_i^e - \varphi_i^n)^2}{\sum_{i=1}^{n} (\varphi_i^e)^2} \right)^{1/2} \tag{39}$$

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

The domain is discretized by unstructured and homogeneous distributions of 650, 2013, 4468, 8647 and 19850 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 Section 4 are not satisfied by the local approximation. It must be noticed that it is not allowed to both, increase the number of points in a cloud and 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 $\|\mathbf{res}\|_\infty \leq 1.E-12 \|\mathbf{RHS}\|_\infty$. Figure 5 shows the FPM solution for the variable ϕ and the test case n=4468.

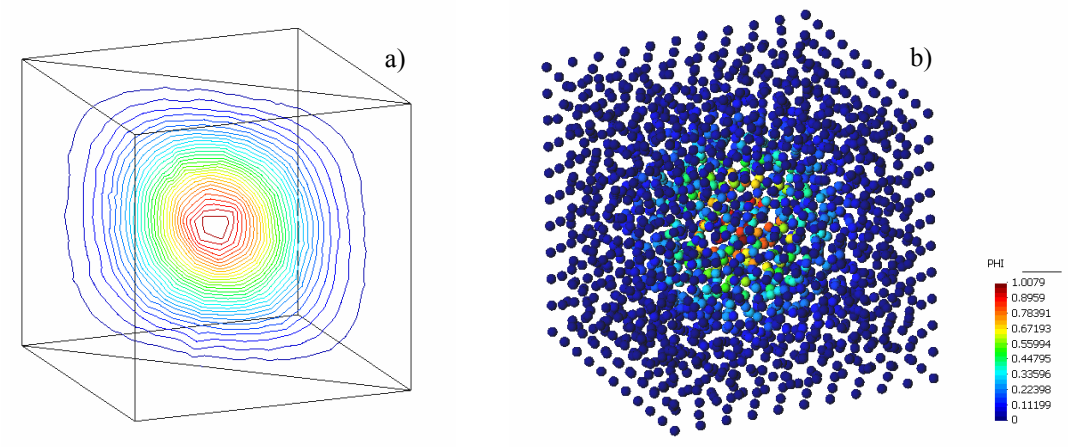

Figure 5: a) Iso-ϕ lines on a surface cut in the domain. b) Problem discretization displaying ϕ-results

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.

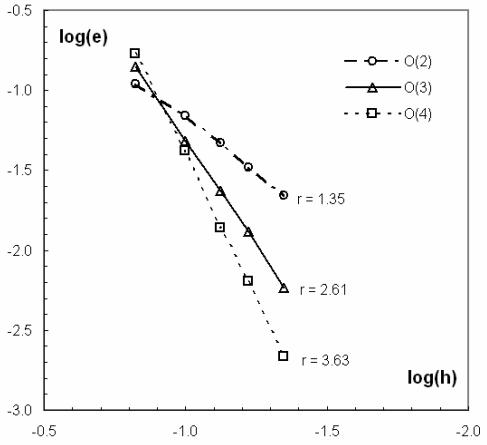

Figure 6: Poisson's problem in a cubic domain: *h*-convergence for $\nabla^2\phi$

Good convergence rates (indicated with *r* in the figure above) can be observed for the present problem. Figure 6 shows that high-order approximations do not improve the 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 its 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 vs. CPU-time is examined in Figure 7. All the cases were computed on a Pentium IV 3.0 GHz processor based machine. For the problem we are dealing with, 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 saving. From the point of view of

the conditioning of the global equation system, Figure 8 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.

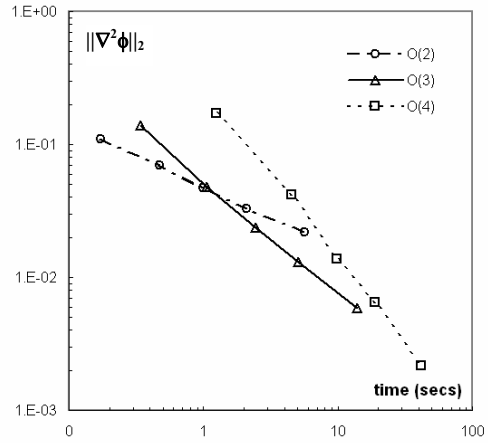

Figure 7: error vs. CPU run time

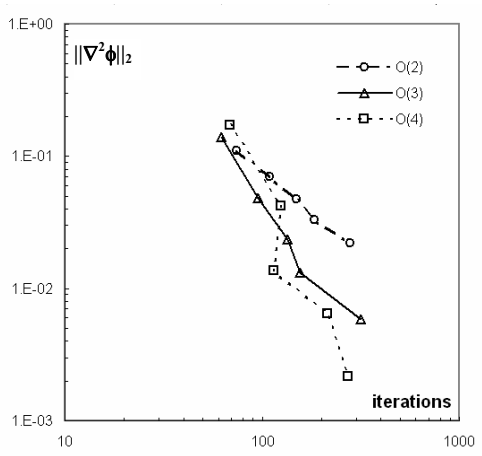

Figure 8: error vs. iterations of BiCG solver

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

$$\begin{aligned} \nabla^2\phi &= 0 \quad \text{in } \Omega \\ \phi - \phi_D &= 0 \quad \text{on } \Gamma_D \\ \hat{\boldsymbol{n}}\cdot\nabla\phi &= 0 \quad \text{on } \Gamma_N \end{aligned} \tag{40}$$

where Γ_D and Γ_N are Dirichlet and Neumann boundaries respectively and $\hat{\boldsymbol{n}}$ is the unitary outward normal vector to Γ_N. Appropriate boundary conditions are set in such a manner that originate an unperturbed velocity field given by $(u,v,w)=(1,0,0)$. Furthermore, in this example a modified form of the Neumann's boundary condition derived through Finite Calculus technique [13] 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 7763 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 approximations respectively. The parameter w = 3.0 is kept fixed in all the test cases and the QR solution of the WLSQ problem is employed when the requirements given in Section 4 are not satisfied by the local approximation. Similar to the numerical example solved in Section 6.1, the order of the local approximation and the number of points in the clouds are 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 Figure 9.

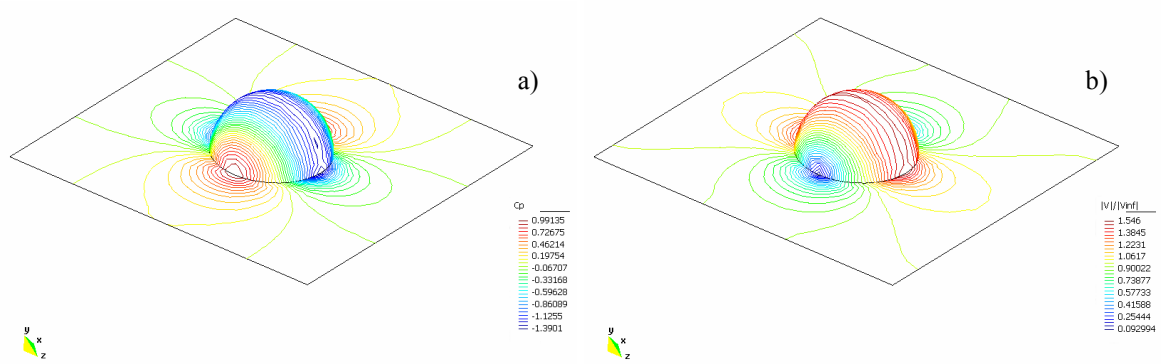

Figure 9: Potential flow around a sphere. Fourth-order calculation case, n=7763 and 80 ≤ np ≤ 90: a) iso-lines of Cp, b) modulus of velocity.

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 Figure 10. The discrete L_2-norms of the error in the numerical approximations obtained by second, third and fourth-order approximation bases are 1.7E-2, 1.3E-2 and 8.8E-3 respectively. These results show that an increase in the order of approximation leads to slightly better numerical results. Note that the spatial discretization is the same in the three calculation cases.

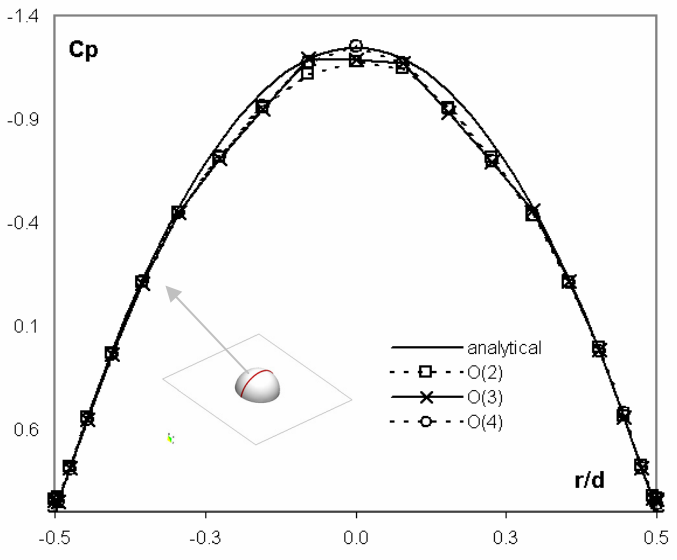

Figure 10: Cp distributions on the sphere using 2nd, 3rd and 4th order approximation bases

Spatial convergence of the solution around the given cross section is examined for three different unstructured and non-homogeneous distributions of points on the sphere using second-order approximation bases. The surface of the half-sphere is discretized by 221, 359 and 1167 points which present higher density around the cross section where the approximation error is computed. Each of these half-sphere surface discretizations belong to an unstructured discretization of the whole domain with 7657, 8151 and 10683 points. The local approximation is built on $25 \leq np \leq 35$ clouds of points with the parameter $w = 3.0$. This setting is kept fixed for the three cases analyzed here. The surface point discretizations on the symmetry plane of the problem are shown in Figure 11.

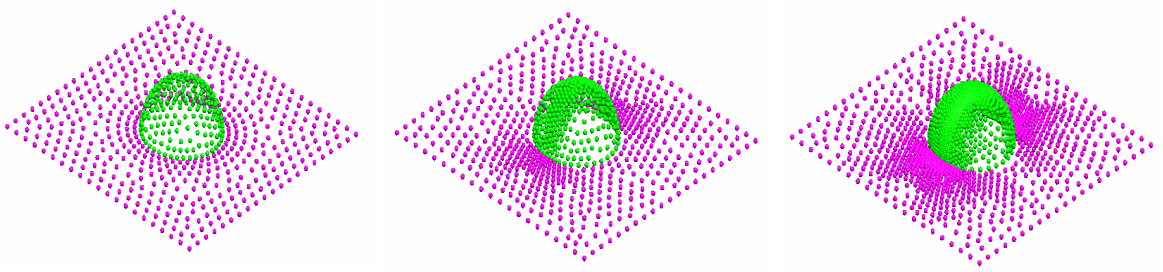

Figure 11: Symmetry plane of the problem. From left to right, half-sphere surface discretization with 221, 359 and 1167 points respectively.

The convergence behaviour with the number of points and the Cp distribution on the sphere for the finest surface discretization ($n_s = 1167$) are shown in Figure 12. 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 higher-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.

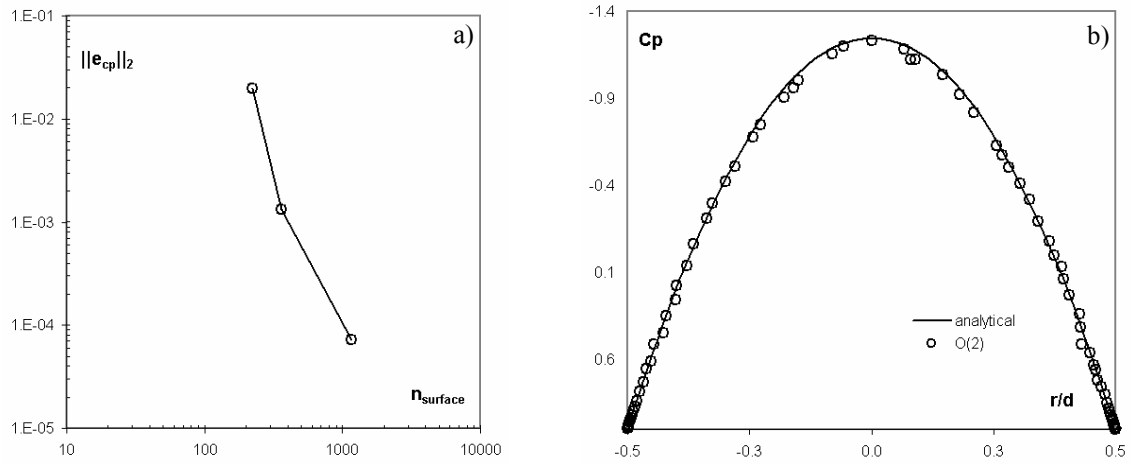

Figure 12: Comparison between calculated and analytical results, n = 10683 and $n_{surface}$ = 1167. a) L_2-norm error vs. number of half-sphere surface points, O(2) with 25 ≤ np ≤ 35. b) Cp distribution on the sphere.

6.3 Potential flow around a semispan wing

The last numerical example is the three-dimensional solution of an ideal irrotational and incompressible fluid past around a semispan wing. The system of equation (40) 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 $(u, v, w) = (1, 0, 0)$. In this example, a modified form of the Neumann's boundary condition derived through the Finite Calculus technique [13] 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 λ = 0.5 and zero sweep-angle with respect to the quarter-chord line. The airfoil section is a NACA-0012 constant along the semispan.

The computational domain is discretized by an unstructured and non-homogeneous distribution of 43335 points. Second-order approximation bases are used with clouds of 50 ≤ np ≤ 70 and the shape functions calculation procedure is allowed to self-adjust according to the iterative procedure presented in Section 4. As in the previous examples, the global equation system is solved by a BiCG method.

Figure 13 shows the surface discretization over the wing (4689 points) and the symmetry plane (837 points); the coloured points display the computed pressure coefficient values. The velocity field around the semispan wing is shown in Figure 14.

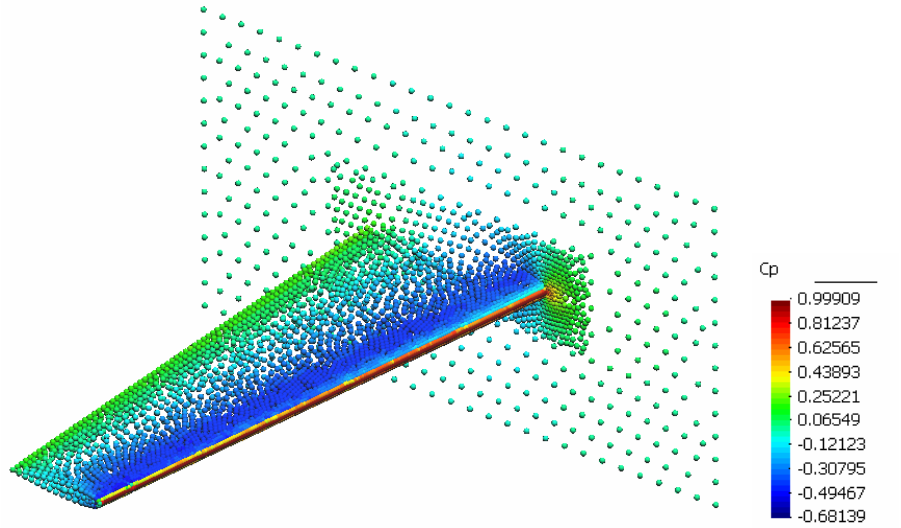

Figure 13: Surface discretization over the semispan wing and the symmetry plane showing surface Cp results.

.

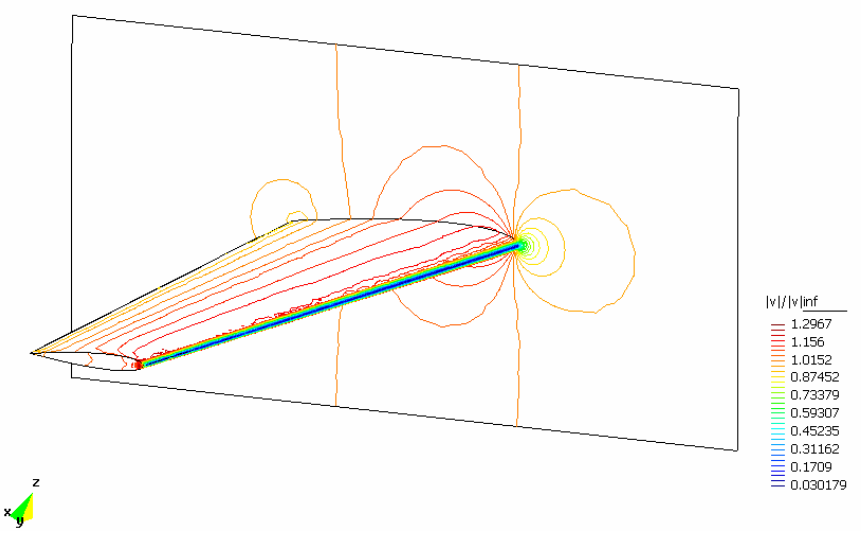

Figure 14: Numerical surface results for non-dimensional modulus of velocity.

The Cp distribution obtained with the present methodology along the root section of the wing is compared with accurate two-dimensional results in Figure 15. A reasonable agreement can be observed.

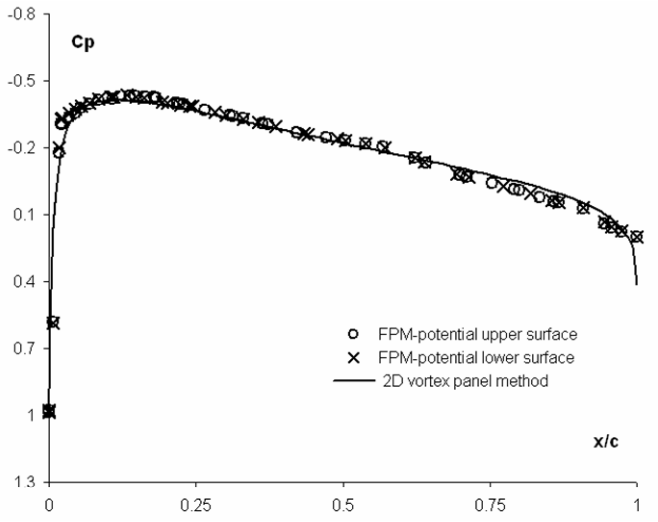

Figure 15: Comparison of Cp distributions along the root section of the semispan wing.

With the aim of demonstrating the performance of the proposed methodology, the CPU-time for each of the stages of the calculation is presented in Table 1. This numerical example was computed on a Pentium IV 3.0 GHz processor based machine.

	CPU-time (secs.)	% Overall time
Local cloud construction	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

Table 1: CPU-times for semispan wing test case (n = 43335, $50 \leq np \leq 70$ and 2^{nd} order approximation bases)

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 seconds. If direct inversion of matrix **A** procedure (11) is used for the shape functions calculation, the time needed is about 5.1 seconds; while if the alternative QR factorization based procedure (32) is used for all the points in the domain, the CPU-time is 11.5 seconds. These facts demonstrate that the iterative procedure needs just 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. It should be noticed that the computational code employed for the calculations presented in Table 1 is not optimized for speed. For this reason, the reported CPU-times must only be considered as a comparison between the different tasks performed but must not be taken as

indicators of the time requirements of the FPM. Evidently, the reported CPU-times can be noticeably improved.

Several numerical experiments, not reported here due to space limits, confirm that the iterative shape function calculation procedure has a noticeably positive impact on the accuracy of the numerical solution, the stability of the collocation procedure and the iterative convergence of the BiCGM. The results obtained are encouraging. However, certain non-expected features, springing from the high-order approximations and related, in general, to particular settings of the approximation parameters, should be studied in more detail.

7. SOLVING THE COMPRESSIBLE FLOW EQUATIONS

In this Section, a Finite Point methodology for solving the three-dimensional Euler equations is presented. The flow solver we propose is based on a modified expression for the calculation of the discrete flux divergence at each point, which allows us to introduce an unknown numerical flux into the formulation. Following the ideas given by the so-called Godunov-type methods, this numerical flux is calculated by means of an approximate Riemann solver. Consequently, this upwind numerical flux provides stabilization to the numerical scheme and also improves their wave capture capabilities. This choice leads to a monotone low-order semi-discrete scheme whose spatial accuracy is improved through a MUSCL reconstruction of the variables. In addition, non-linear limiters are introduced in the formulation in order to preserve the non-oscillatory behaviour of the scheme near discontinuities in the solution field. Finally, the time integration of the semi-discrete scheme is performed by an explicit Runge-Kutta multi-stage scheme.

Next, a detailed description of the methodology above described is presented. Additionally, the procedure employed for applying boundary conditions is described with the aim to round off the description of the flow solver.

7.1 The Euler equations

The first-order hyperbolic system of Euler equations can be written in several equivalent forms. Their conservative differential form is given by

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}^k}{\partial x_k} = \mathbf{0} \tag{41}$$

where k=1,d and d denotes the number of spatial dimensions of the problem. **U** is the conservative variables vector and $\mathbf{F}^k$ is the advective flux vector in the spatial direction x_k. These vectors are given by

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u_i \\ \rho e_t \end{bmatrix} \quad , \quad \mathbf{F}^k = \begin{bmatrix} \rho u_k \\ \rho u_i u_k + \delta_{ik} p \\ (\rho e_t + p) u_k \end{bmatrix} \tag{42}$$

where ρ, p and e_t denote, respectively, the density, pressure and total energy of the fluid; u_i is the i-component of the velocity vector, δ_{ik} is the Kronecker delta and indices i,k = 1,d. The following state relation for a perfect gas closes the system of equations

$$p = \rho(\gamma - 1)\left[e_t - \tfrac{1}{2} u_i u_i \right] \tag{43}$$

in which $\gamma = C_p/C_v$ is the specific heats ratio (in the present work we adopt $\gamma = 1.4$). The solution of Eq. (41) in a closed domain $\Omega \in \Re^d$ with boundaries $\Gamma = \Gamma_\infty \cup \Gamma_w$ requires additional proper initial and boundary conditions, which will be considered later on.

7.1.1 Quasi-linear form of the Euler equations

The system of Euler equations (41) could be written in an equivalent form taking advantage of the fact that the flux vectors $\mathbf{F}^k$ only depends on the conservative variables vector **U**. It allows applying the chain rule to the spatial derivative of the flux vectors and get

$$\frac{\partial \mathbf{F}^k}{\partial x_k} = \frac{\partial \mathbf{F}^k}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial x_k} = \mathbf{A}^k \frac{\partial \mathbf{U}}{\partial x_k} \tag{44}$$

where $\mathbf{A}^k \in \Re^{(d+2)\times(d+2)}$ is the Jacobian matrix of the flux vector $\mathbf{F}^k$. Introducing Eq. (44) in Eq. (41) the following form of the Euler equations is obtained

$$\frac{\partial \mathbf{U}}{\partial t} + \mathbf{A}^k \frac{\partial \mathbf{U}}{\partial x_k} = \mathbf{0} \tag{45}$$

which represents a quasi-linear system of equations if $\mathbf{A}^k = \mathbf{A}(\mathbf{U}, \mathbf{x}_k, t)$ and a linear system if matrix $\mathbf{A}^k$ is constant.

The hyperbolic nature of the Euler equations allows a complete description of the problem in terms of propagating waves. In addition, an important property of this hyperbolic system is that the matrix **A** has real eigenvalues and it is diagonalizable, i.e. it has a complete set of linearly independent eigenvectors. This property allows the following factorization

$$\mathbf{A} = \mathbf{R}\,\boldsymbol{\Lambda}\,\mathbf{R}^{-1} \tag{46}$$

where $\mathbf{R} \in \Re^{(d+2)\times(d+2)}$ is a matrix whose columns are the right eigenvectors of the matrix $\mathbf{A}$, the rows of $\mathbf{R}^{-1}$ are their left eigenvectors and $\boldsymbol{\Lambda} \in \Re^{(d+2)\times(d+2)}$ is a diagonal matrix whose entries are the eigenvalues of $\mathbf{A}$. Even though it is not possible to diagonalize the Jacobian matrices $\mathbf{A}^k$ simultaneously, we can diagonalize any linear combination of these matrices. Therefore, given an arbitrary vector $\mathbf{n} \in \Re^d$ we could state

$$\mathbf{A}_{\boldsymbol{n}} = \mathbf{A}^k n^k = \mathbf{R}_{\boldsymbol{n}}\,\boldsymbol{\Lambda}_{\boldsymbol{n}}\,\mathbf{R}_{\boldsymbol{n}}^{-1} \tag{47}$$

in which the matrices of eigenvectors and eigenvalues are calculated for the Jacobian matrix in the direction of the arbitrary vector $\mathbf{n}$, i.e. $\mathbf{A}_{\boldsymbol{n}}$.

The Jacobian matrices $\mathbf{A}^k$ and their associated eigenvectors and eigenvalues matrices can be calculated analytically if the fluid constitutive relations are specified. Explicit expressions for these matrices can be found in the literature; see for instance [24].

7.2. The low-order scheme

In the present work, the conservative variables vector $\mathbf{U}$ and the advective flux vectors $\mathbf{F}^k$ are approximated by the same set of shape functions. Therefore, for each star point $\mathbf{x}_i \in \Omega$ we have the following numerical approximations

$$\begin{aligned} \hat{\mathbf{U}}\left(\boldsymbol{x}_i\right) &= \hat{\mathbf{U}}_i = \sum_{j\in\Omega_i} N_{ij}\,\mathbf{U}_j^h \\ \hat{\mathbf{F}}^k\left(\boldsymbol{x}_i\right) &= \hat{\mathbf{F}}_i^k = \sum_{j\in\Omega_i} N_{ij}\left(\mathbf{F}_j^k\right)^h \end{aligned} \tag{48}$$

in which $N_{ij} = N_i(\boldsymbol{x}_j)$ is the shape function of the star point $\mathbf{x}_i$ evaluated at the cloud's point $\mathbf{x}_j$ and $(\mathbf{F}_j^k)^h = \mathbf{F}^k(\mathbf{U}_j^h)$. Then, the one-dimensional semidiscrete counterpart of Eq. (41) could be stated for each star point x_i by

$$\frac{\partial \hat{\mathbf{U}}_i}{\partial t} = -\frac{\partial \hat{\mathbf{F}}_i}{\partial x} = -\sum_{j\in\Omega_i} \frac{\partial \boldsymbol{N}_{ij}}{\partial x}\mathbf{F}_j^h = -\sum_{j\in\Omega_i} b_{ij}\mathbf{F}_j^h \tag{49}$$

where $\mathbf{F}_j^h$ is the advective flux vector calculated at a point $x_j \in \Omega_i$ and the coefficient b_{ij} stands for the shape function derivative of x_i evaluated at the same point x_j.

It is important to note that the (^{h}) parameters do not coincide with the approximated ones ($\hat{}$) because in the Finite Point method the shape functions do not interpolate nodal data. These

values are related by Eq. (48), which implies that a linear system must be solved in order to get the (^{h}) parameters. Fortunately, this equation system has excellent properties and can be solved by a few iterations of a Gauss-Seidel method or similar. Henceforth, the markers ($\hat{}$) and (^{h}) will be omitted for the sake of simplicity.

Taking advantage of the PNs property of the shape function derivatives, it is possible to infer

$$\sum_{j\in\Omega_i} b_{ij} = b_{ii} + \sum_{j\neq i} b_{ij} = 0 \quad \rightarrow \quad b_{ii} = -\sum_{j\neq i} b_{ij} \tag{50}$$

and replacing (50) in Eq. (49) the following semidiscrete expression is obtained

$$\frac{\partial \mathbf{U}_i}{\partial t} = -\sum_{j\neq i} b_{ij}\left(\mathbf{F}_j - \mathbf{F}_i\right) \tag{51}$$

Eq. (51) is unstable and needs to be stabilized. For that purpose, a more suitable equivalent form is sought scaling by a factor of 1/2 the stencil of points [23] used for its calculation. In this way, we obtain a totally equivalent semi-discrete expression which is given by

$$\frac{\partial \mathbf{U}_i}{\partial t} = -2\sum_{j\neq i} b_{ij}\left(\mathbf{F}_{ij} - \mathbf{F}_i\right) \tag{52}$$

where $\mathbf{F}_{ij}$ is an *a priori* unknown numerical flux vector at the midpoint of the line segment connecting the star point x_i to another point $x_j \in \Omega_i$. Then, this numerical flux $\mathbf{F}_{ij}$ is calculated by an approximate Riemann solver which naturally provides the required dissipation for the semi-discrete expression. Moreover, the approximate Riemann solver introduces information concerning the exact solution of the problem, giving robustness and excellent shock capturing properties to the numerical scheme. The stencil of points used in the calculation of Eq. (52) is presented in Figure 16.

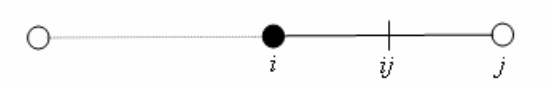

Figure 16: The one-dimensional stencil of points.

7.2.1 Solution of the approximate Riemann problem

The solution of the Riemann problem is one of the few existing analytical solutions for the non-stationary one-dimensional Euler equations. The Riemann problem, centered on the midpoint of the line segment connecting any pair of points x_i and x_j, is set by the following initial conditions

$$\mathbf{U}(x,t_0)=\begin{cases}\mathbf{U}_{\mathrm{L}}=\mathbf{U}_i & x\le x_{ij}\\ \mathbf{U}_{\mathrm{R}}=\mathbf{U}_j & x>x_{ij}\end{cases} \tag{53}$$

where $\mathbf{U}_L$ and $\mathbf{U}_R$ are constant vectors and an infinite length domain is considered. Next, the interaction between the initial states gives rise to the three basic types of flow discontinuities: a shock wave, an expansion fan and a contact discontinuity. For any instant of time $t > t_0$, the ratio of the variables through the discontinuity lines, their exact spatial position and their speed of propagation can be analytically calculated. The initial set up for the Riemann problem is sketched in Figure 17.

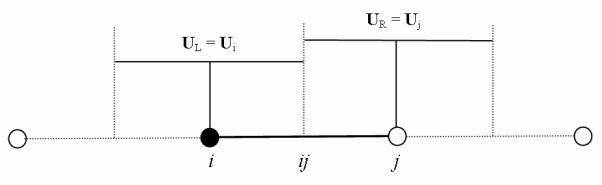

Figure 17: Piecewise constant initial condition for the Riemann problem centered at x_{ij}.

Unfortunately, the exact solution of the Riemann problem involves a non-linear algebraic system of equations whose solution is computationally expensive. Notice that, in order to calculate the semi-discrete expression (52) at each cloud, the Riemann problem must be solved as many times as points there are in the cloud and this penalizes the performance of the numerical technique. Consequently, most techniques based on the Riemann problem do not use its exact solution. These numerical techniques employ the solution of an *approximate* Riemann problem which replaces the real non-linear flux function $\mathbf{F}(\mathbf{U})$ by a locally linearized approximation. In other words, an approximate Riemann solver calculates the exact solution of a linearized problem where the Jacobian matrix $\mathbf{A}$ is considered to be constant. The solution of this linearized problem is trivial and can be written in several equivalent forms, for example, using conservative variables, characteristic variables or in terms of fluxes. The last choice is appropriate for our problem because we need to calculate the unknown flux function at point x_{ij}. Thus, a straightforward calculation [25] for the unknown numerical flux $\mathbf{F}_{ij}=\mathbf{A}\mathbf{U}_{ij}$ leads to

$$\mathbf{F}_{ij}=\tfrac{1}{2}\left(\mathbf{F}_R+\mathbf{F}_L\right)-\tfrac{1}{2}\left|\mathbf{A}\right|\left(\mathbf{U}_R-\mathbf{U}_L\right) \tag{54}$$

where $|\mathbf{A}|=\mathbf{R}|\Lambda|\mathbf{R}^{-1}$ is the absolute value of the Jacobian matrix and $|\Lambda|$ is a diagonal matrix whose entries $|\lambda_i|$ are the absolute value of the eigenvalues of $\mathbf{A}$. Notice that, by self-similarity of the Riemann problem solution, the flux function (54) is constant along a line x_{ij} = const.

The solution of the approximate Riemann problem provides an expression for the unknown numerical flux (54) supposing a locally linearized problem between the points x_i and x_j. Now, it is necessary to calculate an appropriate linear approximation to the real flux function; i.e. we need to define the constant Jacobian matrix **A**.

There are several possibilities for the linearization of the flux function **F**(**U**). In this work we adopt a secant plane approximation [25], i.e. we connect the states $\mathbf{U}_L$ and $\mathbf{U}_R$, which define the linearized Riemann problem, with a plane. The secant plane approximation to the flux function around $\mathbf{U}_L$ could be expressed by

$$\mathbf{F}(\mathbf{U}) \approx \mathbf{F}(\mathbf{U}_L) + \mathbf{A}_{LR}\left(\mathbf{U} - \mathbf{U}_L\right) \tag{55}$$

where the matrix $\mathbf{A}_{LR}$ satisfies

$$\mathbf{F}(\mathbf{U}_R) - \mathbf{F}(\mathbf{U}_L) = \mathbf{A}_{LR}\left(\mathbf{U}_R - \mathbf{U}_L\right) \tag{56}$$

Eq. (56) involves (d+2) equations concerning the components of the flux vector **F**(**U**) and $(d+2)^2$ unknowns which are the components of the matrix $\mathbf{A}_{LR}$. As it was expected, this fact shows that there exist an infinite number of secant planes which contains the line connecting the states $\mathbf{U}_L$ and $\mathbf{U}_R$. With the aim of circumventing this problem, we make use of a result derived from the mean value theorem which states that exists, in the scalar case, an $a_{RL} = a(\xi)$ where ξ is a value between u_L and u_R. In the vector case, it means that exists a matrix

$$\mathbf{A}_{LR} = \mathbf{A}\left(\mathbf{U}_L, \mathbf{U}_R\right) = \mathbf{A}\left(\mathbf{U}_{LR}\right) \tag{57}$$

in which $\mathbf{U}_{RL}$ is an intermediate state between $\mathbf{U}_L$ and $\mathbf{U}_R$. Then, replacing (57) in Eq. (56), the unknowns of the problem are reduced to the (d+2) components of the intermediate state $\mathbf{U}_{RL}$; hence, a well-posed system is obtained. Moreover, statement (57) has another important advantage because any expression based on **A**(**U**) remains true for **A**($\mathbf{U}_{LR}$). This fact allows utilizing, in the linearized problem, the expressions of the Jacobian matrix and their associated eigenvectors and eigenvalues matrices, but evaluated at the new state $\mathbf{U}_{LR}$.

In spite of this progress, the intermediate state vector $\mathbf{U}_{LR}$ has not been obtained yet. Numerous alternatives can be found in the literature; in the present work we adopt the solution developed by Roe [26], which leads to the most popular and less dissipative of the so-called *flux difference splitting* methods.

7.2.2 The Roe average variables

Roe derives an intermediate state vector $\mathbf{U}_{\mathrm{LR}}$ which is defined by the following set of average weighted density variables

$$\begin{aligned}
\tilde{\rho}_{\mathrm{LR}} &= \sqrt{\rho_R\,\rho_L} \\
\tilde{u}_{\mathrm{LR}} &= \frac{\sqrt{\rho_R}u_R + \sqrt{\rho_L}u_L}{\sqrt{\rho_R} + \sqrt{\rho_L}} \\
\tilde{h}_{\mathrm{LR}} &= \frac{\sqrt{\rho_R}h_R + \sqrt{\rho_L}h_L}{\sqrt{\rho_R} + \sqrt{\rho_L}}
\end{aligned} \tag{58}$$

where $h = e_t + p/\rho$ is the specific total enthalpy of the fluid. As a consequence, the average speed of the sound is given by

$$\tilde{c}_{\mathrm{LR}} = \sqrt{(\gamma - 1)\left(\tilde{h}_{RL} - \tfrac{1}{2}\tilde{u}_{RL}^2\right)} \tag{59}$$

Using the average variables given by expressions (58) and (59), it is possible to define the intermediate state $\mathbf{U}_{\mathrm{LR}}$ and, consequently, calculate the constant matrix $\mathbf{A}_{\mathrm{LR}}$ through Eq. (57). Then, replacing the matrix $\mathbf{A}_{\mathrm{LR}}$ (henceforth *Roe matrix*) in Eq. (54), the unknown numerical flux $\mathbf{F}_{\mathrm{ij}}$ is completely defined by

$$\mathbf{F}_{\mathrm{ij}} = \tfrac{1}{2}\left(\mathbf{F}_i + \mathbf{F}_j\right) - \tfrac{1}{2}\left|\mathbf{A}\left(\mathbf{U}_i, \mathbf{U}_j\right)\right|\left(\mathbf{U}_j - \mathbf{U}_i\right) \tag{60}$$

in which $\mathbf{U}_{\mathrm{i}}$ and $\mathbf{U}_{\mathrm{j}}$ have been used instead of $\mathbf{U}_{\mathrm{L}}$ and $\mathbf{U}_{\mathrm{R}}$. With the aim of calculating the absolute value of the Roe matrix, the factorization given by Eq. (46) is employed. We will go back to this point later on.

Due to the fact that a secant plane approximation to the non-linear flux function has been adopted, the Roe solver is able to reproduce the exact solution of the Riemann problem for a single shock or a single contact discontinuity. However, in the particular case of a flow expansion with a sonic transition, the Roe solver allows the appearance of an expansion shock, which violates the second principle of thermodynamics. In order to correct this behaviour, several techniques known as *entropy corrections* have been proposed in the literature. Some of them will be presented later on.

7.2.3 Multi-dimensional extension of the flow solver

The multi-dimensional extension of the scheme presented above is straightforward. For each pair of points ($\mathbf{x}_{\mathrm{i}}$,$\mathbf{x}_{\mathrm{j}}$), a one-dimensional problem is solved in the direction of the vector

$\boldsymbol{l}_{ji} = \boldsymbol{x}_j - \boldsymbol{x}_i$ to obtain the midpoint numerical flux $\mathbf{F}_{ij}$. Then, $\mathbf{F}_{ij}$ is projected onto the Cartesian axis and the semi-discrete scheme (52) results in

$$\frac{\partial \mathbf{U}_i}{\partial t} = -2\sum_{j \neq i} b_{ij}^k \left[\mathbf{F}_{ij}^k - \mathbf{F}_i^k \right] \tag{61}$$

where $k = 1,\mathrm{d}$, being d the number of spatial dimensions of the problem. The Cartesian components of the midpoint numerical flux are obtained by

$$\mathbf{F}_{ij}^k = \tfrac{1}{2}\left(\mathbf{F}_j^k + \mathbf{F}_i^k\right) - \tfrac{1}{2}\left|\mathbf{A}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)\right|\left(\mathbf{U}_j - \mathbf{U}_i\right)\cdot \hat{\boldsymbol{n}}^k \tag{62}$$

where $\hat{\boldsymbol{n}}$ is a versor in the direction of the vector $\boldsymbol{l}_{ji}$ and $|\mathbf{A}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)|$ denotes the absolute value of the Roe matrix calculated in the same direction. The stencil of points employed in the derivation of expression (61) is presented in Figure 18.

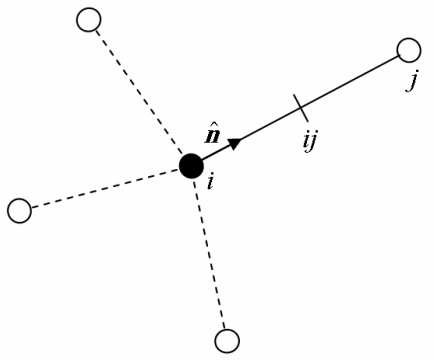

Figure 18: The multi-dimensional stencil of points.

In the multi-dimensional case, the density averaged Roe variables are obtained by means of the following expressions

$$\begin{aligned} \tilde{\rho}_{ij} &= \sqrt{\rho_j \rho_i} \\ \tilde{u}_{ij}^k &= \frac{\sqrt{\rho_j}u_j^k + \sqrt{\rho_i}u_i^k}{\sqrt{\rho_j} + \sqrt{\rho_i}} \\ \tilde{h}_{ij} &= \frac{\sqrt{\rho_j}h_j + \sqrt{\rho_i}h_i}{\sqrt{\rho_j} + \sqrt{\rho_i}} \end{aligned} \tag{63}$$

in which u^k denotes the component of the velocity vector in the cartesian direction x_k. The average Roe variables could be calculated in a more computationally efficient way [24] with the help of a parameter $r = \sqrt{\rho_j / \rho_i}$. Introducing this parameter into Eq. (63) we obtain

$$\tilde{\rho}_{ij} = r\,\rho_i$$
$$\tilde{u}_{ij}^k = \frac{r\,u_j^k + u_i^k}{r+1} \tag{64}$$
$$\tilde{h}_{ij} = \frac{r\,h_j + h_i}{r+1}$$

Then, the average speed of the sound is given by

$$\tilde{c}_{ij} = \sqrt{(\gamma-1)\left[\tilde{h}_{ij} - \tfrac{1}{2}\left(\tilde{u}_{ij}^k \tilde{u}_{ij}^k\right)\right]} \tag{65}$$

As it was mentioned before, the absolute value of the Roe matrix is obtained by factorization (46) in the one-dimensional case and factorization (47) in the multi-dimensional case. Note that the calculation of the dissipation terms, given by Eq. (62) or Eq. (60), requires matrix-matrix and matrix-vector multiplications, which demands a considerable computational effort. In order to carry out these calculations efficiently, Turkel has developed [27] an explicit expression which is presented below.

7.2.4 A practical calculation of the dissipation terms

Section 7.1 states that any linear combination of the Jacobian matrices (in the present case the Roe matrices projected onto the vector linking the points $\mathbf{x}_i$ and $\mathbf{x}_j$) allows the following factorization

$$\mathbf{A}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j) = \mathbf{R}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)\,\boldsymbol{\Lambda}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)\,\mathbf{R}_{\hat{n}}^{-1}(\mathbf{U}_i, \mathbf{U}_j) \tag{66}$$

Thus, the absolute value of this matrix is given by

$$\left|\mathbf{A}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)\right| = \mathbf{R}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)\left|\boldsymbol{\Lambda}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)\right|\mathbf{R}_{\hat{n}}^{-1}(\mathbf{U}_i, \mathbf{U}_j) \tag{67}$$

where, in the three-dimensional case, the diagonal matrix of eigenvalues is $|\boldsymbol{\Lambda}_n(\mathbf{U}_i,\mathbf{U}_j)| = diag\{|\lambda_1|,|\lambda_2|,|\lambda_3|,|\lambda_3|,|\lambda_3|\}$ and

$$\lambda_1 = \hat{u}_{ij} + \tilde{c}_{ij}$$
$$\lambda_2 = \hat{u}_{ij} - \tilde{c}_{ij} \tag{68}$$
$$\lambda_3 = \hat{u}_{ij}$$

In the expressions above, all the variables correspond with the density averaged Roe variables calculated for $(\mathbf{U}_i,\mathbf{U}_j)$ and $\hat{u}_{ij} = \tilde{u}_{ij}^k\,\hat{n}^k$. Notice that the multiplicity of the eigenvalue λ_3 is identical to the number of spatial dimensions of the problem under consideration. Then, the dissipation terms can be expressed by

$$\mathbf{D} = \left|\mathbf{A}_{\hat{n}}(\mathbf{U}_{ij})\right|(\mathbf{U}_j - \mathbf{U}_i) = \mathbf{R}_{\hat{n}}^{-1}(\mathbf{U}_{ij})\left|\mathbf{\Lambda}_{\hat{n}}(\mathbf{U}_{ij})\right|\mathbf{R}_{\hat{n}}(\mathbf{U}_{ij})\cdot\Delta\mathbf{U} \tag{69}$$

in which the difference vector $\Delta\mathbf{U} = (\mathbf{U}_j - \mathbf{U}_i)$. According to the procedure suggested by Turkel [27], the dissipation terms (69) are calculated as follows

$$\mathbf{D} = \left|\lambda_3\right|\Delta\mathbf{U} + \left[\left(\frac{\sigma_1 - \left|\lambda_3\right|}{c^2}\right)\Psi_1 + \frac{\sigma_2}{c}\Psi_2\right]\begin{bmatrix}1\\ \tilde{u}_{ij}^{(1)}\\ \tilde{u}_{ij}^{(2)}\\ \tilde{u}_{ij}^{(3)}\\ \tilde{h}_{ij}\end{bmatrix} + \left[\frac{\sigma_2}{c}\Psi_1 + \left(\sigma_1 - \left|\lambda_3\right|\right)\Psi_2\right]\begin{bmatrix}0\\ \hat{n}^{(1)}\\ \hat{n}^{(2)}\\ \hat{n}^{(3)}\\ \hat{u}_{ij}\end{bmatrix} \tag{70}$$

where

$$\begin{aligned}\Psi_1 &= (\gamma - 1)\left(q\Delta\mathbf{U}^{(1)} - \tilde{u}_{ij}^{(1)}\Delta\mathbf{U}^{(2)} - \tilde{u}_{ij}^{(2)}\Delta\mathbf{U}^{(3)} - \tilde{u}_{ij}^{(3)}\Delta\mathbf{U}^{(4)} + \Delta\mathbf{U}^{(5)}\right)\\ \Psi_2 &= -\hat{u}_{ij}\Delta\mathbf{U}^{(1)} + \hat{n}^{(1)}\Delta\mathbf{U}^{(2)} + \hat{n}^{(2)}\Delta\mathbf{U}^{(3)} + \hat{n}^{(3)}\Delta\mathbf{U}^{(4)}\end{aligned} \tag{71}$$

and

$$\begin{aligned}\sigma_1 &= \tfrac{1}{2}\left(\left|\lambda_1\right| + \left|\lambda_2\right|\right)\\ \sigma_2 &= \tfrac{1}{2}\left(\left|\lambda_1\right| - \left|\lambda_2\right|\right)\\ q &= \tfrac{1}{2}\,\tilde{u}_{ij}^{k}\,\tilde{u}_{ij}^{k}\end{aligned} \tag{72}$$

Eq. (70) gives us an expression for evaluating the dissipation terms in the general three-dimensional calculation case. In the one-dimensional and two-dimensional case, the dissipation terms are obtained cancelling (in Eq. (70) and Eq. (71)) all the variables in the proper Cartesian components.

7.2.5 On the entropy correction

As it was mentioned before, in the particular case that a transition through a sonic point occurs in a flow expansion, the Roe solver will allow the appearance of an expansion shock (pressure and density decrease across the shock). This solution, which is valid in the context of the locally linearized Riemann problem solved by Roe, does not comply with the second principle of thermodynamics because an expansion shock involves entropy reduction. Hence, several techniques have been proposed in order to overcome this unphysical behaviour. In general, all these techniques consist in locating the sonic transition within the computational domain and diffusing the shock expansion into expansion fans. The latter is achieved by fixing a minimum allowable value for the wave speed, i.e. limiting the eigenvalues to a constant value $\lambda_{\min} > 0$. Concerning the dissipation terms behaviour, while the acoustic

eigenvalues λ_1 and λ_2 vanish at sonic points, the eigenvalue λ_3 goes to zero at stagnation points. In addition to the possible appearance of expansion shocks, vanishing eigenvalues could cause misbehaviour of the dissipation terms and, consequently, lead to numerical instabilities for particular flow conditions.

A simple way to overcome all these problems is limiting the minimum value of the wave speeds to a fraction of the Jacobian matrix spectral radius $\rho(\mathbf{A})$. This is achieved [27] re-defining the eigenvalues according to

$$\begin{aligned} |\lambda_1| &= \max\left[\, |\lambda_1|, \alpha_2 \rho(\mathbf{A}) \,\right] \\ |\lambda_2| &= \max\left[\, |\lambda_2|, \alpha_2 \rho(\mathbf{A}) \,\right] \\ |\lambda_3| &= \max\left[\, |\lambda_3|, \alpha_1 \rho(\mathbf{A}) \,\right] \end{aligned} \tag{73}$$

where $\alpha_1 \approx 0.1$, $\alpha_2 \approx 0.2$ and $\rho(\mathbf{A}) = |\hat{u}_{ij}| + \tilde{c}_{ij}$. A more accurate correction, proposed by Harten and Hymann [28], is given by

$$\begin{aligned} &|\lambda_k| = \frac{\lambda_k^2 + \delta^2}{2\delta} \quad \text{if } |\lambda_k| < \delta \\ &\delta = \max\left[0, (\lambda_k - \lambda_k^L), (\lambda_k^R - \lambda_k)\right] \quad k = 1,3 \end{aligned} \tag{74}$$

where λ_k corresponds to the eigenvalues (68) calculated for the Roe average state between $\mathbf{U}_i$ and $\mathbf{U}_j$ and λ_k^L, λ_k^R are the eigenvalues calculated at the points $\mathbf{x}_i$ and $\mathbf{x}_j$ (which define the left and right constant state of the Riemann problem). Additional corrections can be found in the literature, see for instance a comparative study among some of them in [29].

7.3 The high-order scheme

The low-order scheme developed in the preceding section is useless in practice. In order to make this scheme able to capture the features of the flow with precision, it is necessary to increase its spatial order of accuracy. This is accomplished by replacing the zero-order extrapolation of the variables ($\mathbf{U}_L=\mathbf{U}_i$ and $\mathbf{U}_R=\mathbf{U}_j$) at the midpoint $\mathbf{x}_{ij}$ by a higher-order extrapolation. The MUSCL (Monotone Upstream-centered Schemes for Conservation Laws) methodology [30] allows achieving second and third-order accurate schemes using linear and quadratic reconstruction of the variables respectively. Leftward-biased reconstructions of the variables defining the left state of the Riemann problem at the midpoint $\mathbf{x}_{ij}$ are sketched in Figure 19. A similar rightward-biased extrapolation is possible for the right state.

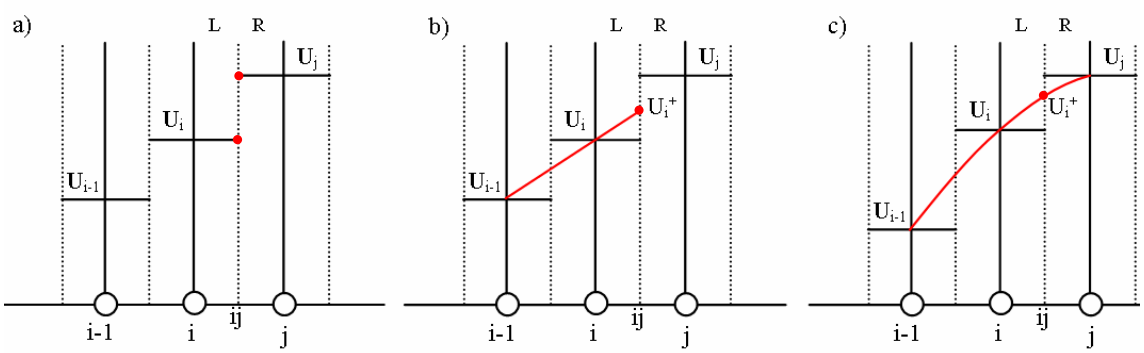

Figure 19: Leftward-biased extrapolations of the variables at the point $\mathbf{x}_{ij}$ ($\mathbf{U}_i^+$): a) zero-order extrapolation (leading to a first-order accurate scheme); b) linear extrapolation (leading to a second-order accurate scheme); c) quadratic extrapolation (leading to a third-order accurate scheme).

The proposed high-order methodology does not guarantee an oscillation free solution. Therefore, in the present work, the monotonicity of the solution is enforced by introducing non-linear *limiters* into the reconstruction process. Basically, these limiters recognize any local extrema of the solution field and automatically switch the high-order extrapolation to a zero-order extrapolation. This mechanism results in a low-order monotone scheme around extrema of the solution field.

Next, the extrapolation procedure is briefly described and some possibilities to calculate the limiters are presented.

7.3.1 Second and third-order accurate schemes

Assuming that $\mathbf{U}(x)$ is a function smooth enough in the neighbourhood of the point x_{ij}, we can state the following leftward-biased approximations [25]

Second-order approximation (linear centered reconstruction)

$$\mathbf{U}(x) \approx \mathbf{U}_i + \frac{\mathbf{U}_j - \mathbf{U}_i}{\Delta x}\left(x - x_i\right) \tag{75}$$

Second-order approximation (linear leftward-biased reconstruction)

$$\mathbf{U}(x) \approx \mathbf{U}_i + \frac{\mathbf{U}_i - \mathbf{U}_{i-1}}{\Delta x}\left(x - x_i\right) \tag{76}$$

Third-order approximation (quadratic leftward-biased reconstruction)

$$\begin{aligned}\mathbf{U}(x) \approx \mathbf{U}_i &- \frac{\mathbf{U}_j - \mathbf{U}_i - (\mathbf{U}_i - \mathbf{U}_{i-1})}{24} + \frac{\mathbf{U}_j - \mathbf{U}_i}{\Delta x}\left(x - x_i\right) \\ &+ \frac{\mathbf{U}_j - \mathbf{U}_i - (\mathbf{U}_i - \mathbf{U}_{i-1})}{2\Delta x^2}\left(x - x_i\right)\left(x - x_j\right)\end{aligned} \tag{77}$$

Then, defining $\Delta x = x_j - x_i$ and taking $x = x_{ij} = x_i + \Delta x/2$, it is possible to combine the approximations (75), (76) and (77) into the following expression

$$\mathbf{U}_i^+ = \mathbf{U}_i + \frac{1}{4}\left[(1-\eta)(\mathbf{U}_i - \mathbf{U}_{i-1}) + (1+\eta)(\mathbf{U}_j - \mathbf{U}_i)\right] \tag{78}$$

where $\mathbf{U}_i^+$ is the leftward approximation to $\mathbf{U}(x_{ij})$. The parameter $\eta = -1$ leads to a second-order leftward-biased approximation for $\mathbf{U}_i^+$ and choosing $\eta = 1$ or $\eta = 1/3$, a second-order centered approximation or a third-order approximation is obtained respectively.

Following the same reasoning line, rightward-biased approximations are obtained by

Second-order approximation (linear centered reconstruction)

$$\mathbf{U}(x) \approx \mathbf{U}_j + \frac{\mathbf{U}_j - \mathbf{U}_i}{\Delta x}(x - x_j) \tag{79}$$

Second-order approximation (linear rightward-biased reconstruction)

$$\mathbf{U}(x) \approx \mathbf{U}_j + \frac{\mathbf{U}_{j+1} - \mathbf{U}_j}{\Delta x}(x - x_j) \tag{80}$$

Third-order approximation (quadratic rightward-biased reconstruction)

$$\begin{aligned}\mathbf{U}(x) \approx \mathbf{U}_j &- \frac{\mathbf{U}_{j+1} - \mathbf{U}_j - (\mathbf{U}_j - \mathbf{U}_i)}{\Delta x} + \frac{\mathbf{U}_j - \mathbf{U}_i}{\Delta x}(x - x_j) \\ &+ \frac{\mathbf{U}_{j+1} - \mathbf{U}_j - (\mathbf{U}_j - \mathbf{U}_i)}{2\Delta x^2}(x - x_i)(x - x_j)\end{aligned} \tag{81}$$

Then, defining $\Delta x = x_j - x_i$ and taking $x = x_{ij} = x_i + \Delta x/2$, it is possible to combine the approximations (79), (80) and (81) into the following expression

$$\mathbf{U}_j^- = \mathbf{U}_j - \frac{1}{4}\left[(1-\eta)(\mathbf{U}_{j+1} - \mathbf{U}_j) + (1+\eta)(\mathbf{U}_j - \mathbf{U}_i)\right] \tag{82}$$

where $\mathbf{U}_j^-$ is the rightward approximation to $\mathbf{U}(x_{ij})$. Similar to the previous case, the parameter $\eta = -1$ leads to a second-order rightward-biased approximation for $\mathbf{U}(x_{ij})$ and choosing $\eta = 1$ or $\eta = 1/3$, a second-order centered approximation or a third-order approximation is obtained respectively.

Notice that the high-order approximations to the variables $\mathbf{U}(x_{ij})$ require an enlargement of the stencil of points. The calculation of the Eq. (78) and Eq. (82) needs the variables vectors evaluated at certain points x_{i-1} and x_{j+1} which, in general, do not coincide with any discrete

points in the discretization. When we are dealing with unstructured discretizations in the multi-dimensional context, obtaining the variables at these fictitious points is not a trivial task and several techniques to deal with this can be found in the literature; see for instance [31, 32]. Following the ideas presented in [11], in the present work the variables $\mathbf{U}_{i-1}$ and $\mathbf{U}_{j+1}$ are obtained by a centered approximation to the $\nabla\mathbf{U}$ at the points $\mathbf{x}_i$ and $\mathbf{x}_j$. This procedure leads to the following expressions

$$\begin{aligned} \mathbf{U}_i - \mathbf{U}_{i-1} &= 2\boldsymbol{l}_{ji} \cdot \nabla\mathbf{U}_i - \left(\mathbf{U}_j - \mathbf{U}_i\right) \\ \mathbf{U}_{j+1} - \mathbf{U}_j &= 2\boldsymbol{l}_{ji} \cdot \nabla\mathbf{U}_j - \left(\mathbf{U}_j - \mathbf{U}_i\right) \end{aligned} \tag{83}$$

in which $\boldsymbol{l}_{ji} = \boldsymbol{x}_j - \boldsymbol{x}_i$ is the vector linking the points $\mathbf{x}_i$ and $\mathbf{x}_j$.

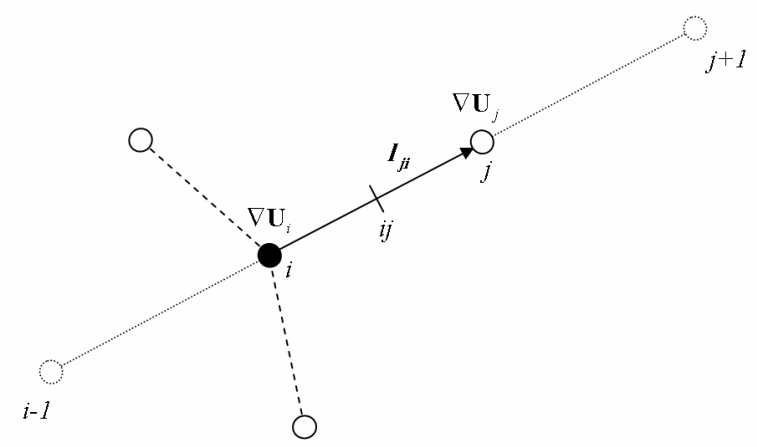

Figure 20: Multi-dimensional reconstruction of the variables

As it was mentioned before, the reconstruction process described above does not guarantee an oscillation free solution near discontinuities. The appearance of spurious oscillations in the solution field depends upon the relation between the difference vectors ($\mathbf{U}_j$-$\mathbf{U}_i$), ($\mathbf{U}_i$-$\mathbf{U}_{i-1}$) and ($\mathbf{U}_{j+1}$-$\mathbf{U}_j$) which give information about the local gradient of the solution. Thus, the non-linear limiters check and correct these quantities following several criteria derived from TVD (Total Variation Diminishing) schemes or similar, which mathematically define the properties of non-oscillatory schemes.

7.3.2 Limiters

Limiting is still an area of active research and numerous approaches can be found in the literature; see for instance [24, 25, 32]. In the following, two approaches for limiting the slopes of the solution in the MUSCL extrapolation technique are proposed. The first approach

adopts the minmod limiter [25] and the second one uses the Van Albada limiter [31], which is less restrictive than the former limiter.

7.3.2.1 The minmod limiter

It is possible to demonstrate [25] that the high-order extrapolations given by Eq. (78) and Eq. (82) do not generate oscillations when $\mathbf{U}_i \le \mathbf{U}_i^+ \le \mathbf{U}_j$, the differences $(\mathbf{U}_i\text{-}\mathbf{U}_{i-1})$ and $(\mathbf{U}_j\text{-}\mathbf{U}_i)$ have the same sign and the following relation holds true

$$\left|\mathbf{U}_i - \mathbf{U}_{i-1}\right| \le \frac{3-\eta}{1-\eta}\left|\mathbf{U}_j - \mathbf{U}_i\right| \tag{84}$$

In addition, it is necessary that $\mathbf{U}_i \le \mathbf{U}_j^- \le \mathbf{U}_j$, the differences $(\mathbf{U}_{j+1}\text{-}\mathbf{U}_j)$ and $(\mathbf{U}_j\text{-}\mathbf{U}_i)$ have the same sign and also

$$\left|\mathbf{U}_{j+1} - \mathbf{U}_j\right| \le \frac{3-\eta}{1-\eta}\left|\mathbf{U}_j - \mathbf{U}_i\right| \tag{85}$$

With the aim of ensuring all these conditions, the reconstruction of the variables is performed in the following manner

$$\begin{aligned}
\mathbf{U}_i^+ = \mathbf{U}_i + \frac{1}{4}\Big\{&(1-\eta)\,minmod\left[\mathbf{U}_i - \mathbf{U}_{i-1}\,,\, b(\mathbf{U}_j - \mathbf{U}_i)\right] + \\
&(1+\eta)\,minmod\left[\mathbf{U}_j - \mathbf{U}_i\,,\, b(\mathbf{U}_i - \mathbf{U}_{i-1})\right]\Big\} \\
\mathbf{U}_j^- = \mathbf{U}_j - \frac{1}{4}\Big\{&(1-\eta)\,minmod\left[\mathbf{U}_{j+1} - \mathbf{U}_j\,,\, b(\mathbf{U}_j - \mathbf{U}_i)\right] + \\
&(1+\eta)\,minmod\left[\mathbf{U}_j - \mathbf{U}_i\,,\, b(\mathbf{U}_{j+1} - \mathbf{U}_j)\right]\Big\}
\end{aligned} \tag{86}$$

where b is called *compression parameter* and is calculated according to $1 \le b \le (3\text{-}\eta)/(1\text{-}\eta)$; the choice of $b = 1$ leads to a more dissipative scheme. The minmod function could be calculated for given arguments x and y by means of the following expression

$$minmod(x, y) = \mathrm{sgn}(x) \cdot \max\left[0\,,\, \min\left(|x|\,,\, \mathrm{sgn}(x) \cdot y\right)\right] \tag{87}$$

which gives as a result the smallest argument in absolute value if the arguments x and y have the same sign and; otherwise, it gives zero.

7.3.2.2 The Van Albada limiter

Following the ideas presented in [31], the Van Albada limiter at points $\mathbf{x}_i$ and $\mathbf{x}_j$ is given by

$$\mathbf{s}_i = \max\left[0\,,\ \frac{2(\mathbf{U}_i - \mathbf{U}_{i-1})(\mathbf{U}_j - \mathbf{U}_i) + \varepsilon}{(\mathbf{U}_i - \mathbf{U}_{i-1})^2 + (\mathbf{U}_j - \mathbf{U}_i)^2 + \varepsilon}\right]$$
$$\mathbf{s}_j = \max\left[0\,,\ \frac{2(\mathbf{U}_{j+1} - \mathbf{U}_j)(\mathbf{U}_j - \mathbf{U}_i) + \varepsilon}{(\mathbf{U}_{j+1} - \mathbf{U}_j)^2 + (\mathbf{U}_j - \mathbf{U}_i)^2 + \varepsilon}\right] \tag{88}$$

where ε is a very small positive constant which avoids possible divisions by zero when the surrounding flow field is smooth. The limiters $\mathbf{s}_i$ and $\mathbf{s}_j$ could be calculated using primitive, conservative or characteristic variables. The latter lead to the best results at the expense of a higher computational cost. In the present work the limiters are calculated using the conservative variables vector.

Once the limiters given by Eq. (88) have been calculated, the reconstruction scheme is modified according to the next

$$\mathbf{U}_i^+ = \mathbf{U}_i + \frac{\mathbf{s}_i}{4}\left[(1-\eta)(\mathbf{U}_i - \mathbf{U}_{i-1}) + (1+\eta)(\mathbf{U}_j - \mathbf{U}_i)\right]$$
$$\mathbf{U}_j^- = \mathbf{U}_j - \frac{\mathbf{s}_j}{4}\left[(1-\eta)(\mathbf{U}_{j+1} - \mathbf{U}_j) + (1+\eta)(\mathbf{U}_j - \mathbf{U}_i)\right] \tag{89}$$

Notice that the limiters (88) take values between $\mathbf{0} \le \mathbf{s} \le \mathbf{1}$. When the limiters are equal to the unity, Eq. (89) leads to a high-order extrapolation to the variables $\mathbf{U}_i^+$ and $\mathbf{U}_j^-$ and, consequently, a high-order accurate scheme is obtained. When the limiters are equal to zero, the zero-order extrapolation is recovered and it leads to the low-order accurate scheme.

The choice of the minmod limiter leads to a more dissipative scheme than the Van Albada limiter choice. In some particular cases in which the Van Albada limiter allows that oscillations appear in the solution field, the adoption of the minmod limiter could be advantageous.

7.3.3 Obtaining the high-order accurate scheme

Eq. (86) and Eq. (89) give a high-order approximation to the left and right states, defining the approximate Riemann problem centered at the point $\mathbf{x}_{ij}$. Then, according to Eq. (62) the high-order approximation to the numerical flux is given by

$$\mathbf{F}_{ij}^k = \tfrac{1}{2}\left(\mathbf{F}^k(\mathbf{U}_i^+) + \mathbf{F}^k(\mathbf{U}_j^-)\right) - \tfrac{1}{2}\left|\mathbf{A}_{\hat{n}}(\mathbf{U}_i^+, \mathbf{U}_j^-)\right|\left(\mathbf{U}_j^- - \mathbf{U}_i^+\right)\cdot\hat{\boldsymbol{n}}^k \tag{90}$$

Finally, replacing Eq. (90) in Eq. (61), the semidiscrete expression of the high-order accurate scheme is obtained.

7.4 Time discretization

The temporal discretization of Eq. (61) is done in a fully explicit manner by means of a multi-stage method that is a subset of the Runge-Kutta family of schemes. Assuming that the vector of conservative variables $\mathbf{U}^{\mathrm{h}}$ is known at time $t = t^n$, the right hand side of Eq. (61) is calculated for each point (RHS_i). Then, it is possible to advance the solution in time from t^n to t^{n+1} by means of the following *s-stage* scheme

$$\begin{aligned} \mathbf{U}_i^{(0)} &= \mathbf{U}_i^n \\ &\vdots \\ \mathbf{U}_i^{(s)} &= \mathbf{U}_i^n + \alpha_s \Delta \mathrm{t}_i \mathbf{RHS}_i^{(s-1)} \\ &\vdots \\ \mathbf{U}_i^{n+1} &= \mathbf{U}_i^{(s_{\max})} \end{aligned} \tag{91}$$

where $\Delta \mathrm{t}_i$ is the time step evaluated at the star point $\mathbf{x}_i$ and α_s are integration coefficients that depend on the number of stages employed (s_{max}). For two, three and four-stages schemes these parameters are set as follows:

2 stages $\rightarrow$ $\alpha_1 = 1/2$ and $\alpha_2 = 1.0$

3 stages $\rightarrow$ $\alpha_1 = 3/5$, $\alpha_2 = 3/5$ and $\alpha_3 = 1.0$

4 stages $\rightarrow$ $\alpha_1 = 1/4$, $\alpha_2 = 1/3$, $\alpha_3 = 1/2$ and $\alpha_4 = 1.0$

Previously, the difference between the ($^{\mathrm{h}}$) parameters and the approximated ones (^) has been pointed out. Taking into account that $\mathrm{RHS}_i = f(\mathbf{U}_j^{\mathrm{h}})\ \forall \mathbf{x}_j \in \Omega_i$, the following linear system has to be solved at the end of each integration stage

$$\mathbf{M}\,\mathbf{U}^h = \hat{\mathbf{U}} \tag{92}$$

where $\mathbf{M} \in \Re^{n \times n}$ is the *mass matrix* of the system, which results from the assembly of the $\mathbf{N}_{ij}$ vectors (see Eq.(48)). Fortunately, as it was said before, this equation system has excellent properties and can be solved by a few iterations of a Gauss-Seidel method or similar.

7.4.1 The time step calculation and stability requirements

It is known that the time step employed in explicit integration schemes must be bounded by some stability criterion. Also, the numerical computation of conservation laws, like the Euler equations, requires satisfying an additional condition called CFL (Courant-Friedrichs-Lewys) condition. In short, the latter states that the numerical domain of dependence (the stencil of points involved in the RHS_i) must contain the physical domain of dependence (bounded by

the waves of the system in the x-t characteristics plane). It is possible to show [25] that this statement simply translates into an inequality restricting the maximum distance that any wave can travel in a single time step. In other words, the CFL condition states that none of the waves can travel faster than the maximum wave speed of the system, i.e. the spectral radius of the Jacobian matrix. The exact inequality which bounds the time step for a given discretization scheme could be obtained performing a linear stability analysis (e.g. von Neumann analysis) and, in many cases, the linear stability condition is the same as the CFL condition.

In conventional discretization techniques, linear stability conditions are usually enforced taking a time step which is equal to a constant $C < 1$, called Courant number, multiplied by a typical distance of the stencil of points and divided by the spectral radius of the Jacobian matrix. Unfortunately, for Finite Point discretizations, stability analysis are very difficult to perform and their results are not valid for arbitrary point discretizations, especially in the multidimensional case. This happens because the distribution of points in the FPM is highly random and the shape functions and their derivatives depend not only on the position of the points but also on other factors like the order of approximation and the weighting function parameters. All these factors, which change from a cloud to another, make it very difficult to devise a general stability criterion.

In the present work, introducing some heuristics, the time step calculation for each star point $\mathbf{x}_i$ is obtained as follows

$$\Delta t_i = C \min_{j \neq i} \left(\frac{\left\| \boldsymbol{l}_{ji} \right\|}{\max\left(\lambda_i^{max}, \lambda_j^{max} \right)} \right) \qquad \nabla \boldsymbol{x}_j \in \Omega_i \tag{93}$$

where $\boldsymbol{l}_{ji} = \boldsymbol{x}_j - \boldsymbol{x}_i$ is the vector linking the points $\mathbf{x}_i$ and $\mathbf{x}_j$ (see Figure 20) and

$$\begin{aligned} \lambda_i^{max} &= \left| u_i^k \, \hat{n}^k \right| + c_i \\ \lambda_j^{max} &= \left| u_j^k \, \hat{n}^k \right| + c_j \end{aligned} \tag{94}$$

are the maximum wave speeds in the direction of $\boldsymbol{l}_{ji}$. In Eq. (94) $\hat{\boldsymbol{n}}$ is a versor in the direction of $\boldsymbol{l}_{ji}$, u^k are the components of the velocity vectors at points $\mathbf{x}_i$ and $\mathbf{x}_j$ and c is the speed of the sound at the same points. Even though the Courant number C could take values above the unity in multi-stages integration schemes, in the present work we restrict this parameter to $C < 1$ in order to reinforce the fulfilment of the stability requirements. The adoption of a local time step Δt_i in Eq. (91) increases the speed of convergence to the steady state for stationary

problems. If the solution of the problem is time dependent, a global time step must be adopted. This global time step is obtained by

$$\Delta t_g = \min_i \left(\Delta t_i \right) \qquad \nabla \boldsymbol{x}_i \in \Omega \tag{95}$$

7.5 Boundary conditions

As it was mentioned in Section 7.1, the system of Euler equations (41) requires the definition of additional initial and boundary conditions in order to complete the description of the problem to be solved. The initial conditions only start the explicit calculation (91) and they are simple to implement. In general, they could be taken from the far-field state $\mathbf{U}_\infty$. On the contrary, the specification of boundary conditions for a given problem is not a trivial matter. For each point, the boundary conditions should be applied observing the mathematical behaviour of the equations. In general terms, variables entering the computational domain must be prescribed according to the far-field state $\mathbf{U}_\infty$ and the variables leaving the computational domain must be able to move freely. The characteristic formulation of the Euler equations can manage boundary conditions in a natural way. Several studies based on characteristics give us some guidelines for applying boundary conditions in agreement with the behaviour of the equations [24, 25]. If the mathematical properties of the equations are not taken into account when applying boundary conditions, the appearance of wave reflections and numerical instabilities at the boundaries is quite common. This misbehaviour could seriously affect the convergence and the accuracy of the numerical solution.

In the present work two kinds of boundary conditions are employed. The first one is concerned with *far-field conditions* applied on outer boundaries (Γ_∞) and the second one is concerned with *slip wall conditions* applied on solid boundaries (Γ_w) (notice that $\Gamma = \Gamma_\infty \cup \Gamma_w$). In the case of far-field boundary conditions, the prescribed fluxes at each boundary point are obtained by solving an approximate Riemann problem in the normal direction to the boundary, between the boundary point state $\mathbf{U}_\mathrm{i}$ and the far-field state $\mathbf{U}_\infty$. Solving this problem automatically gives us the appropriate flux vectors which have to be imposed, in accordance with the propagation of waves across the boundary. Over inner or outer solid boundaries, slip wall conditions are applied. This conditions force the fluxes to remain tangent to the boundaries, cancelling their components in the normal direction. Below, the procedure for applying boundary conditions adopted in the present work is briefly described.

7.5.1 Far-field boundary conditions

The procedure for applying far-field boundary conditions is presented for the multi-dimensional case. For each boundary point $\mathbf{x}_i \in \Gamma_\infty$ we need its normal (outward) versor $\hat{\boldsymbol{n}}_i$ and a set of tangent versors given by $\hat{\boldsymbol{t}}_1$ and $\hat{\boldsymbol{t}}_2$. Then, it is possible to calculate the flux vectors in the normal-tangent system as follows

$$\mathbf{F}_n = \mathbf{F}_i^k \, \hat{n}_i^k \quad ; \quad \mathbf{F}_{t_1} = \mathbf{F}_i^k \, \hat{t}_1^k \quad ; \quad \mathbf{F}_{t_2} = \mathbf{F}_i^k \, \hat{t}_2^k \tag{96}$$

Due to the fact that the exchange of information between the computational domain and the far-field is done in the boundary normal direction, the normal flux $\mathbf{F}_n$ must be modified according to the far-field state. The new normal flux vector is obtained by the solution of the Roe's approximate Riemann problem between the states $\mathbf{U}_i$ and $\mathbf{U}_\infty$

$$\mathbf{F}_n^* = \tfrac{1}{2}\left(\mathbf{F}_n + \mathbf{F}_{\infty,\hat{n}}\right) - \tfrac{1}{2}\left|\mathbf{A}_{\hat{n}}(\mathbf{U}_\infty, \mathbf{U}_i)\right|\left(\mathbf{U}_\infty - \mathbf{U}_i\right) \tag{97}$$

where the Roe matrix $|\mathbf{A}_n(\mathbf{U}_\infty,\mathbf{U}_i)|$ is calculated in the direction of the normal versor $\hat{\boldsymbol{n}}_i$ and the far-field normal flux vector is given by $\mathbf{F}_{\infty,\hat{\boldsymbol{n}}} = \mathbf{F}^k(\mathbf{U}_\infty)\,\hat{n}^k$. Then, the updated flux vectors at point $\mathbf{x}_i$ are obtained, in the Cartesian axes, by

$$\overline{\mathbf{F}}_i^k = \mathbf{F}_n^* \hat{n}^k + \mathbf{F}_{t_1} \hat{t}_1^k + \mathbf{F}_{t_2} \hat{t}_2^k \tag{98}$$

The flux vectors given by Eq. (98) are forced in the solution of the problem in each time step.

7.5.2 Slip wall boundary conditions

The slip wall condition on solid boundaries implies the cancellation of the normal fluxes at each point $\mathbf{x}_i \in \Gamma_w$. This is accomplished by setting

$$\boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_i = 0 \tag{99}$$

where $\hat{\boldsymbol{n}}_i$ is the normal outward versor to the boundary at point $\mathbf{x}_i$. In particular problems, due to the fact that condition (99) is not compatible with the initial conditions, some numerical instability could happen during the initial time steps. It is possible to avoid this numerical instability by means of the relaxation of Eq. (99) [32] according to

$$\boldsymbol{u}_i^{(n)} \cdot \hat{\boldsymbol{n}}_i = \left[\, \boldsymbol{u}_i^{(n-1)} \cdot \hat{\boldsymbol{n}}_i \,\right]\left(1 - \kappa\right) \tag{100}$$

where indices $^{(n)}$ and $^{(n-1)}$ refer to the solution at two contiguous time steps. This weaker slip wall condition allows the fluid to penetrate the solid boundaries during the initial time steps and tends to condition (99) as the solution of the problem advances in time. The parameter $\kappa \leq 1$ controls the fluid penetration during the initial time steps and a value $\kappa=0.8$ is adopted. Notice that, as a consequence of the enforcement of condition (99), all the convective flux components through the solid wall vanish and only the pressure contribution remains at the flux vector in the normal direction to the solid wall, i.e.

$$\mathbf{F}_{\hat{n}} = \mathbf{F}^k \hat{n}^k = \begin{bmatrix} 0 \\ p\,\hat{n}^{(1)} \\ p\,\hat{n}^{(2)} \\ p\,\hat{n}^{(3)} \\ 0 \end{bmatrix} \tag{101}$$

In the present work both slip wall boundary conditions were studied but no important differences between the enforcement of Eq. (99) or Eq. (101) have been observed. Similar to the far-field boundary conditions, the slip wall boundary conditions are applied to the solution of the problem in each time step.

7.5.3 A remark on trailing edge points treatment

According to the previous comments, fluid particles are not allowed to penetrate solid boundaries in inviscid calculations. Hence, slip wall boundary conditions are applied at these points; therefore, the surface normal and tangent vectors are needed. Sharp edges, like the trailing edge of airfoils and wings, present a problematic situation because the normal vectors are geometrically indefinite at these points. Thus, the direction in which the advective flux must be forced to zero is not well-determined. Another problem arises from the clouds construction for trailing edge points, where the procedure presented in Section 5.2, leads to distorted asymmetrical clouds of points. These clouds could violate the CFL condition because it is quite possible that, in such cases, the physical domain of dependence is not totally included in the cloud of points. As a consequence, numerical instabilities may appear in the solution field.

In order to fix the problems described above, in the present work we adopt a heuristic approach. In relation to clouds construction at trailing edge points, we do not apply restrictions to the points in these clouds. As a result, the resulting cloud structure, which in general is symmetrical, only depends upon the distribution of neighbouring points.

Concerning boundary conditions, slip wall conditions are not applied to trailing edge points, i.e. the variables at trailing edge points are allowed to move freely. In short, any point located at the trailing edge is considered as a point in the interior domain. Several numerical experiments show that this treatment allows trailing edge points to automatically adapt their behaviour according to the neighbouring points. Moreover, although this approach does not permit the appearance of rear stagnation points, it allows smooth fluid at the trailing edge avoiding any kind of numerical instabilities to appear in the numerical solution.

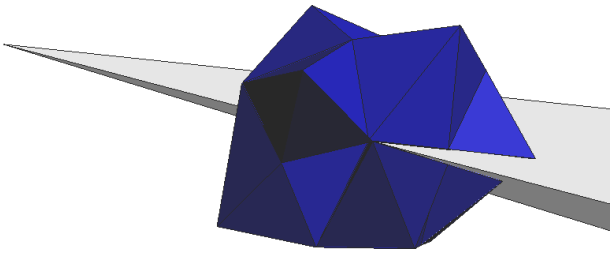

Figure 21: Cloud of points constructed for a star point located at the trailing edge of a wing

8. NUMERICAL EXAMPLES

In this section, some compressible flow calculations are presented with the aim of illustrating the performance of the methodology described in this work. We begin with two examples that solve one-dimensional shock tube problems. Then, some two-dimensional numerical examples are presented. Firstly, a two-dimensional version of the shock tube problem and, secondly, two computations involving subsonic and transonic flow around airfoils. All these examples are aimed at verifying the behaviour of the numerical scheme by checking numerical computations against analytical and numerical sources (verification assessment). Next, some three-dimensional calculations are presented. The first example concerns a subsonic flow over a sphere. Although this example has no practical interest, it allows assessing the low Mach number behaviour of the scheme as well as evaluating its intrinsic dissipation. Then, a transonic flow around the ONERA M6-wing is solved. This example, which is a classic CFD validation test for external flows, allows demonstrating the applicability of the present methodology for practical calculations. With the same objective in mind, by the end of this section another transonic flow calculation concerning a NACA wing-body configuration is presented.

8.1 One-dimensional examples

8.1.1 The sock tube problem: $p_L/p_R = 10$

The shock tube problem is a one-dimensional, transonic and non-stationary Riemann problem proposed by Sod in 1978. The fact that this problem has both analytical and experimental solution, has turned it into one of the most popular benchmarks for numerical schemes. In brief, this problem consists of a closed tube divided by a diaphragm into two compartments (left and right). Each compartment contains a gas at rest with a given pressure and density. The simulation begins when the diaphragm is suddenly removed. After that, a shock wave, a contact discontinuity and an expansion fan are brought about by the interaction between the left and right states. These discontinuities propagate within the tube.

In this example we solve a shock tube problem in a close domain $\Omega = (0,1)$. The problem is defined by the following initial conditions

$$\mathbf{U}\left(\mathrm{x},\mathrm{t}_0\right)=\begin{cases}\mathbf{U}_{\mathrm{L}}=\left(1,0,2.5\right)^{\mathrm{T}} & x\le 0.5\\ \mathbf{U}_{\mathrm{R}}=\left(0.125,0,0.025\right)^{\mathrm{T}} & x>0.5\end{cases} \tag{102}$$

which give a pressure ratio across the diaphragm $p_L/p_R = 10$. According to these initial conditions, the intensity of the shock is moderate and the flow regime after the expansion is subsonic. The computational domain is discretized by a homogeneous distribution of 100 points and second-order spatial approximation is obtained using five-points clouds. A third-order MUSCL extrapolation, in conjunction with the Van Albada limiter, is adopted for performing the high-order computations. Moreover, a four-stage scheme is employed in order to advance the solution in time. Next, the numerical results are computed for a time t = 0.2 seconds after the rupture of the diaphragm. The solution given by the low and high-order scheme is shown in Figure 22 and Figure 23 respectively.

8.1.2 The sock tube problem: $p_L/p_R = 100$

In this case, the initial pressure ratio across the diaphragm is $p_L/p_R = 100$ and a supersonic flow is obtained after the expansion. The initial conditions are given by

$$\mathbf{U}\left(\mathrm{x},\mathrm{t}_0\right)=\begin{cases}\mathbf{U}_{\mathrm{L}}=\left(1,0,2.5\right)^{\mathrm{T}} & x\le 0.5\\ \mathbf{U}_{\mathrm{R}}=\left(0.125,0,0.0025\right)^{\mathrm{T}} & x>0.5\end{cases} \tag{103}$$

The analysis domain is discretized by a homogeneous distribution of 200 points and, similar to the previous test case, second-order spatial approximations in clouds of 5 points are employed. Numerical high-order results are obtained using a third-order MUSCL extrapolation scheme in conjunction with the minmod limiter. The time integration is

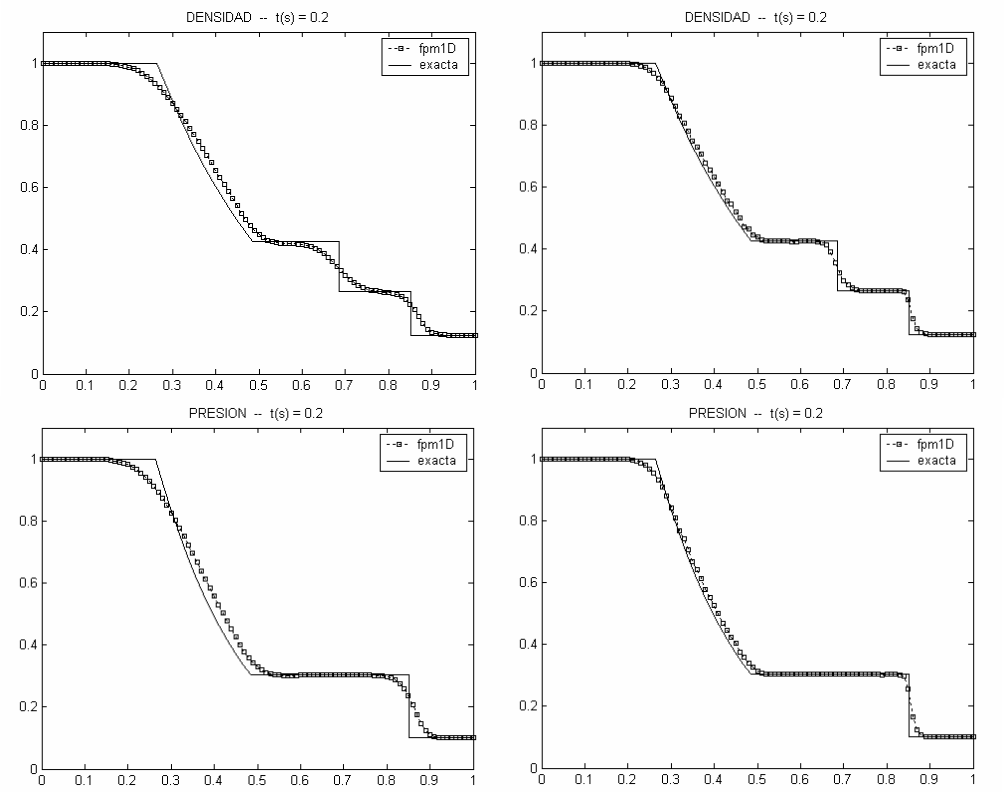

Figure 22: The one-dimensional shock tube problem (p_L/p_R=10). Left: low-order results for the density and pressure variables; Right: high-order results for the same variables.

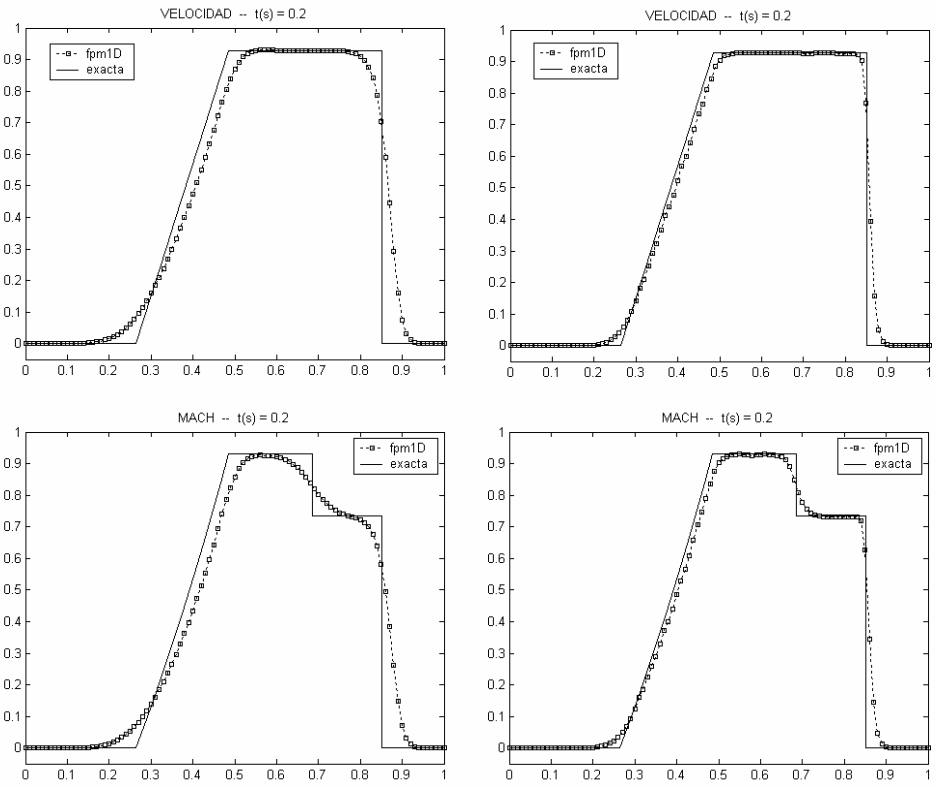

Figure 23: The one-dimensional shock tube problem (p_L/p_R=10). Left: low-order results for the velocity and Mach number variables; Right: high-order results for the same variables.

performed using a four-stage scheme. In Figure 24 below, the numerical results are shown for a time t = 0.15 seconds after the rupture of the diaphragm.

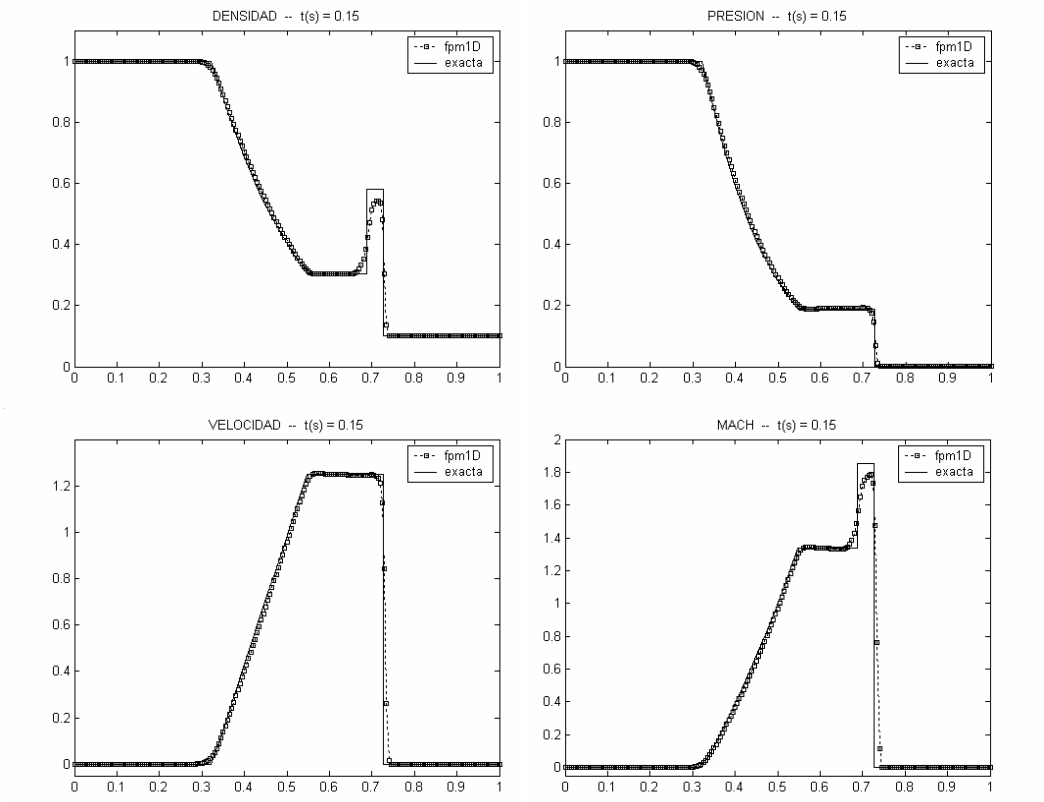

Figure 24: The one-dimensional shock tube problem (p_L/p_R=100). High-order results for the density, pressure, velocity and Mach number.

For this example, numerical computations have shown that the high-order scheme leads to small instabilities in the flow variables when the Van Albada limiter is chosen. Conversely, the minmod limiter gives a smooth solution without affecting the accuracy of the results. A comparison between Van Albada and minmod limiters is presented in Figure 25.

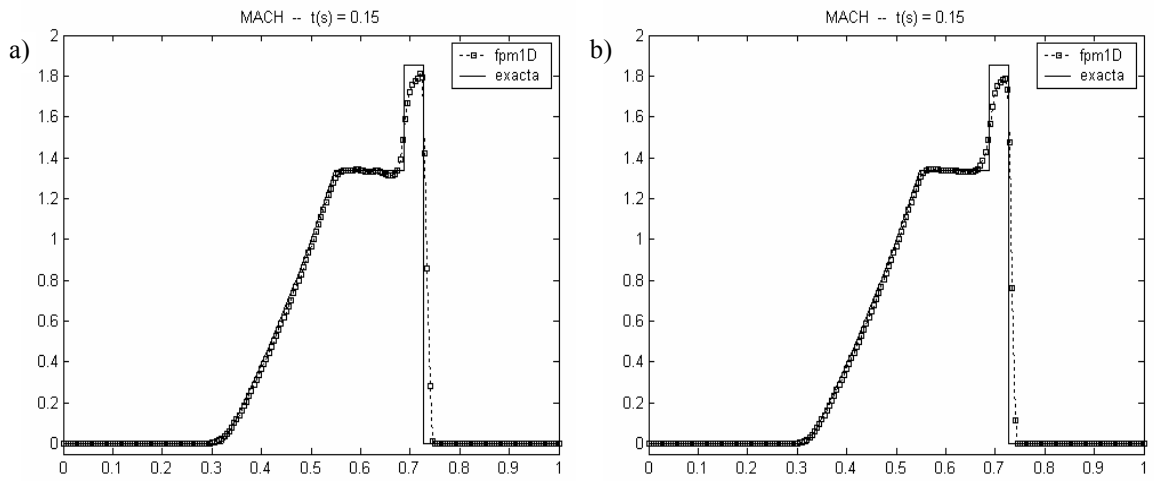

Figure 25: The one-dimensional shock tube problem (p_L/p_R=100): a) Van Albada limiter, b) Minmod limiter

8.2 Two-dimensional examples

8.2.1 The two-dimensional sock tube problem: $p_L/p_R = 10$

In this example, the Riemann problem, set by the initial conditions (102), is solved in a two-dimensional domain $\Omega = (0,1)\times(0,1)$ where slip wall conditions are applied on the boundaries. The domain discretization consists of a structured distribution of 100 points in each spatial direction. A second-order approximation is employed in order to calculate the shape functions and their derivatives and 15 points per cloud are used. The time integration is performed by a four-stage scheme and a third-order MUSCL extrapolation with the Van Albada limiter is chosen. The numerical results are presented below in Figures 26 and 27.

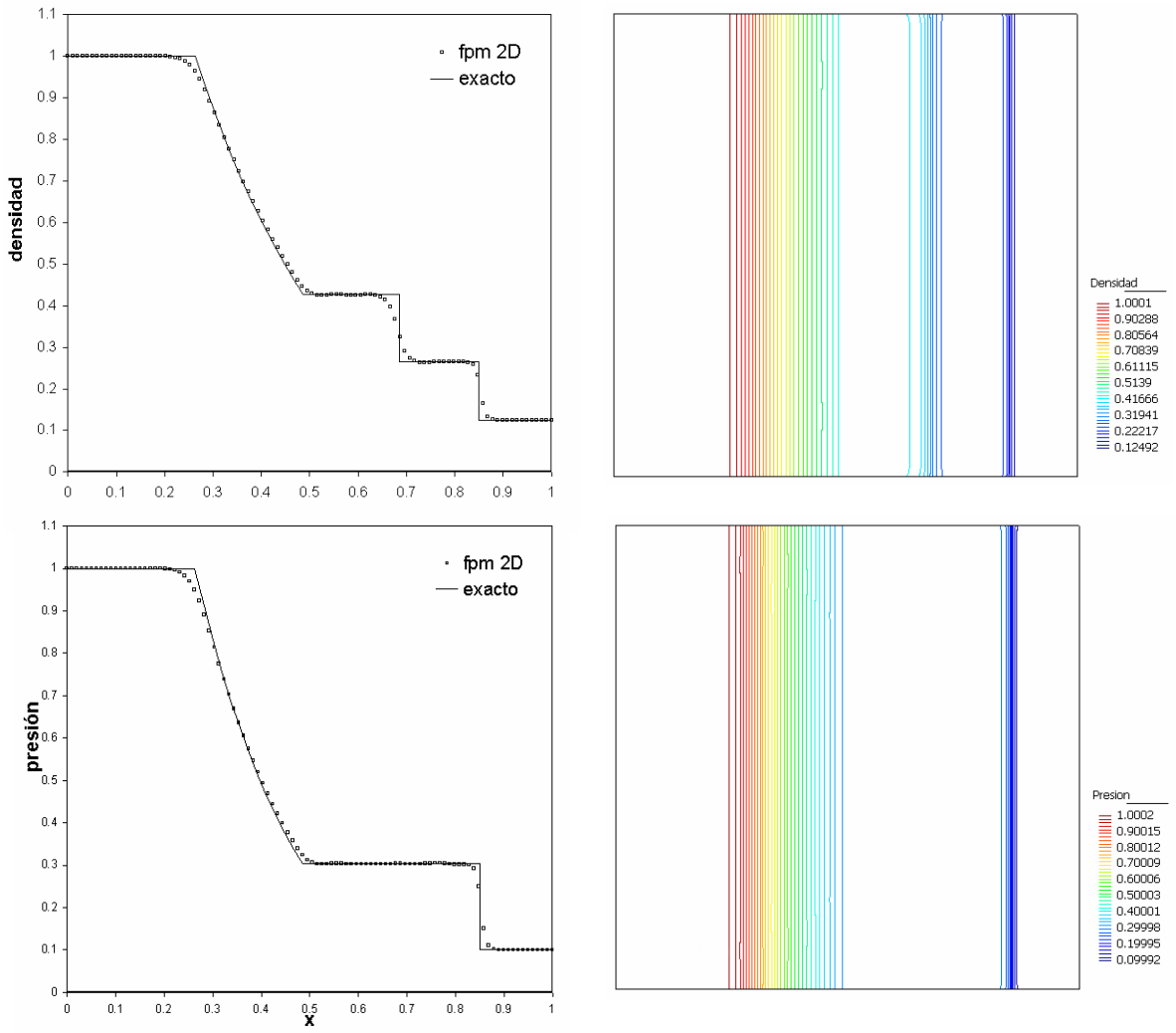

Figure 26: Two-dimensional shock tube problem (p_L/p_R=100). Left: a comparison between calculated and analytical results for density and pressure distribution across the tube; Right: isolines graphs for the same variables.

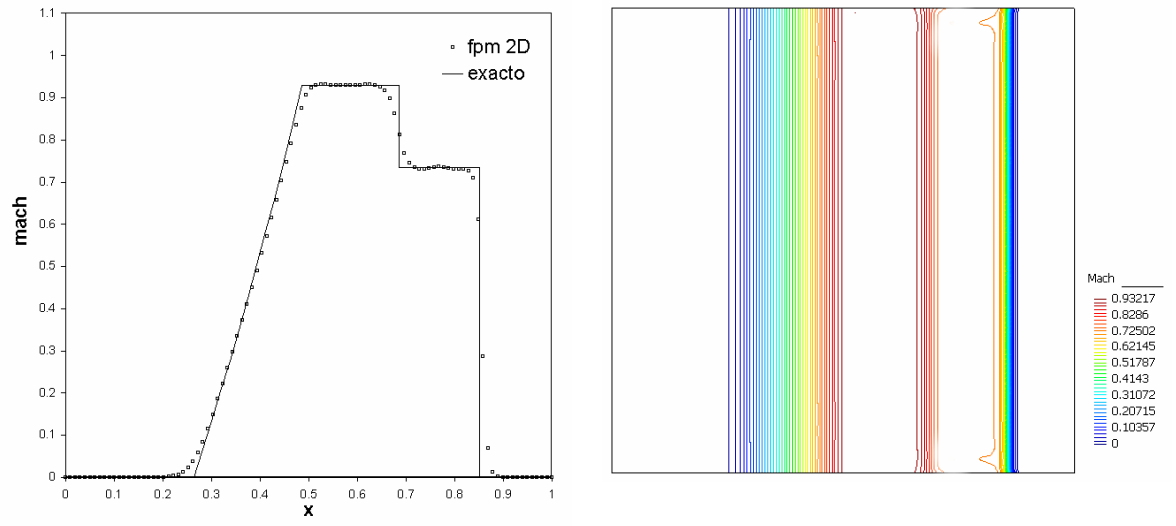

Figure 27: Two-dimensional shock tube problem (p_L/p_R=100). Left: a comparison between calculated and analytical results for the Mach number distribution across the tube; Right: Mach number isolines.

As with one-dimensional examples, a good agreement between analytical and numerical results can be observed in Figure 26 and Figure 27. Note also that the two-dimensional computations preserve the one-dimensional nature of the problem.

8.2.2 Subsonic flow around a NACA 0012 airfoil

The flow around a NACA 0012 airfoil set at zero incident angle (α = 0º) is computed for a freestream Mach number M_∞ = 0.2. The computational domain, whose outer boundary has a radius about twenty times the airfoil chord, is discretized by a non-structured distribution of 3328 points. Second-order approximation bases are used for calculating shape functions and their derivatives in clouds where 13≤ *np* ≤19. Moreover, a third-order MUSCL extrapolation, in conjunction with the Van Albada limiter and a three-stage time integration scheme, is adopted for the flow solver. Next, the spatial discretization of the problem in the proximity of the airfoil is presented in Figure 28. Cp isolines around the airfoil are shown in Figure 29 and a comparison between the calculated Cp distribution on the airfoil and reference potential flow results is presented in Figure 30.

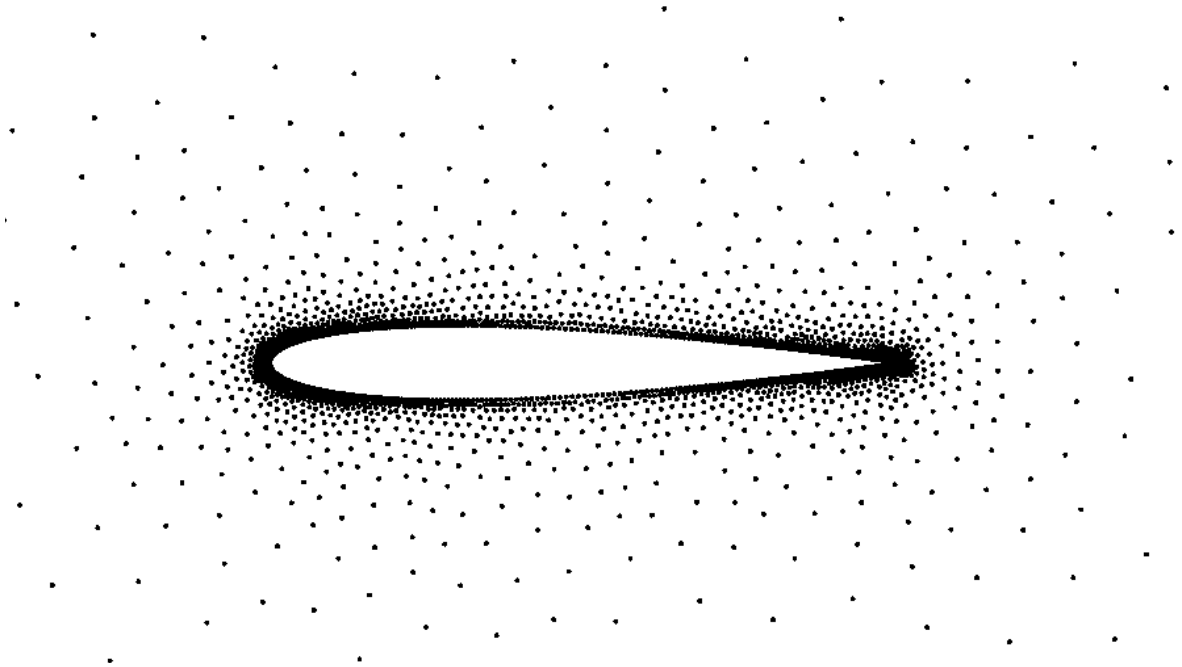

Figure 28: Spatial discretization of the problem in the proximity of the airfoil NACA 0012

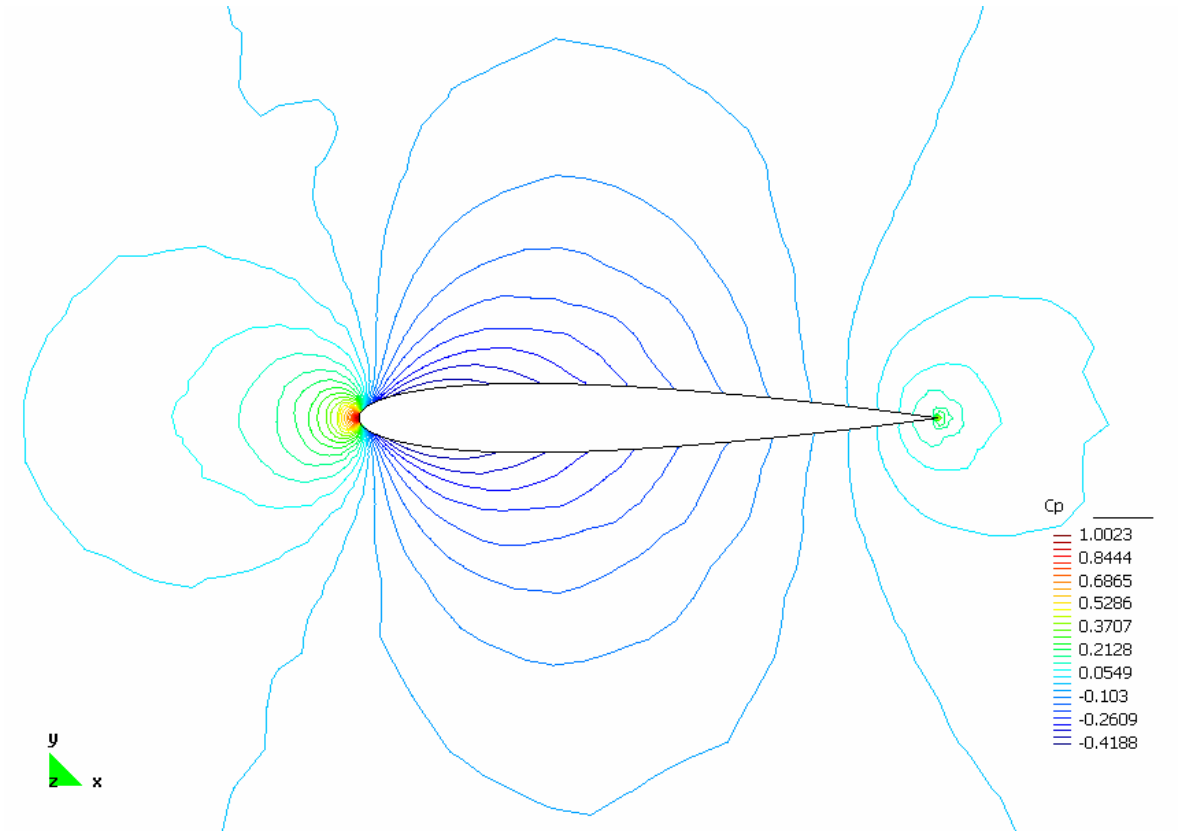

Figure 29: Cp isolines in the near field of the NACA 0012 airfoil, M_∞=0.2 and α=0º.

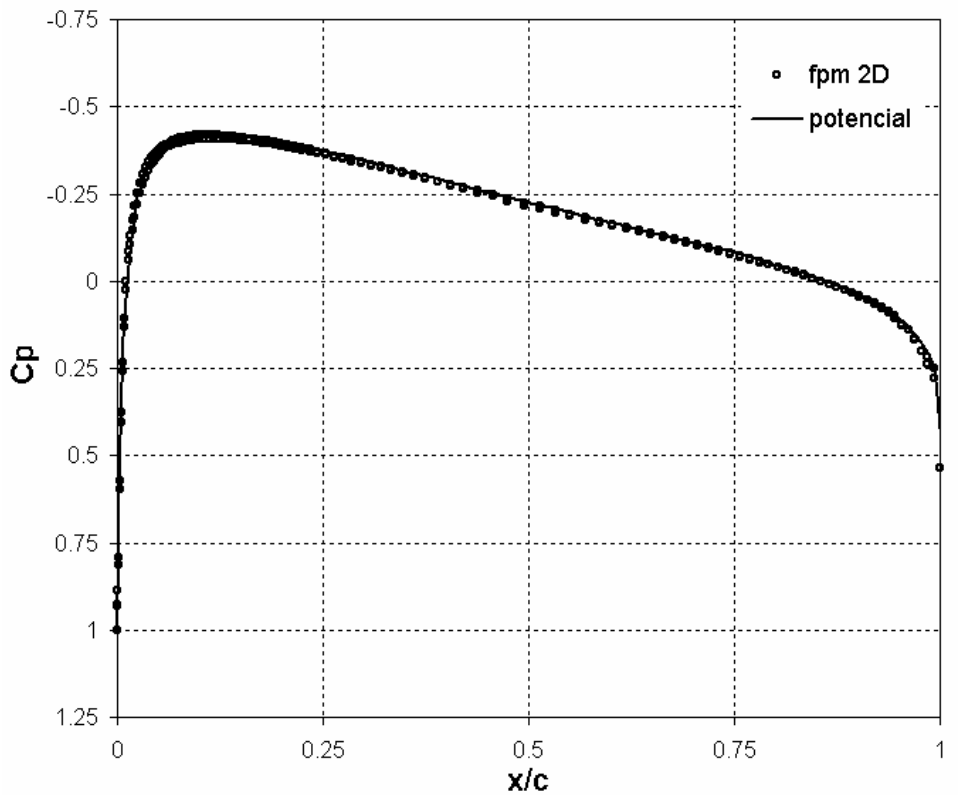

Figure 30: Cp distribution on the airfoil NACA 0012, M_∞=0.2 and α=0º. A comparison between Finite Point and potential flow results.

8.2.3 Transonic flow around a RAE 2822 airfoil

This numerical example involves a calculation of the flow field around a RAE 2822 airfoil. The latter is set at an incidence angle α = 3º and the freestream Mach number is M_∞ = 0.75. The computational domain is discretized by a non-structured distribution of 4207 points and a higher density of points is placed both at the airfoil leading edge area and on the upper side of the airfoil, where the shock wave is expected to be located. Second-order spatial approximations are calculated in clouds where $15 \leq np \leq 20$. Moreover, a third-order MUSCL extrapolation, in conjunction with the Van Albada limiter and a three-stage time integration scheme, is adopted for solving this test case. In the following, the domain discretization in the proximity of the airfoil is presented in Figure 31 and Cp and Mach number isolines around the airfoil are shown in Figure 32 and Figure 33 respectively. Finally, the Cp distribution obtained on the airfoil by the present methodology is compared with a reference numerical computation [33] in Figure 34. This result was obtained using an unstructured mesh with 5071 points and a WENO reconstruction FV method. Despite of some small amplitude

oscillations in the suction side of the airfoil for the Finite Point calculations, a good agreement between the Cp distributions can be observed.

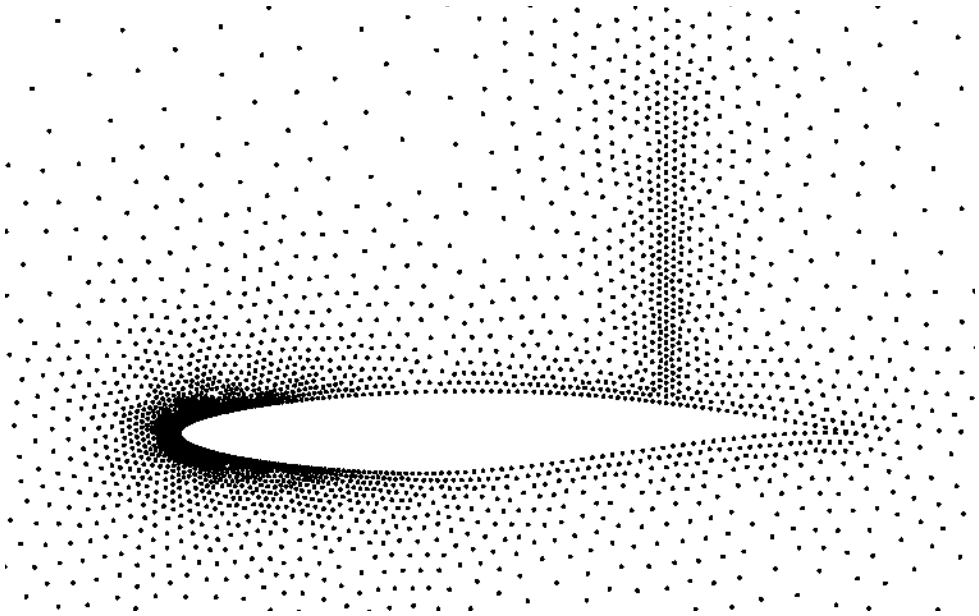

Figure 31: Spatial discretization in the near-field of the RAE 2822 airfoil.

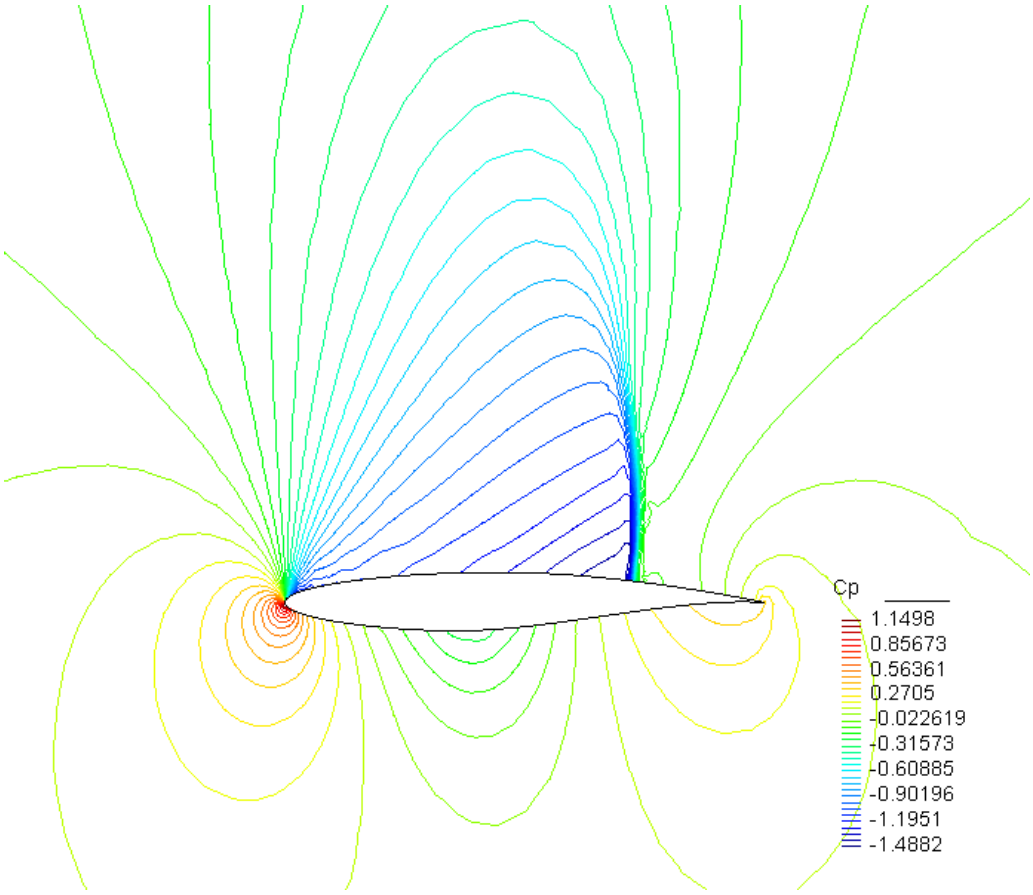

Figure 32: Cp isolines in the proximity of the RAE 2822 airfoil, M_∞=0.75 and α=3º.

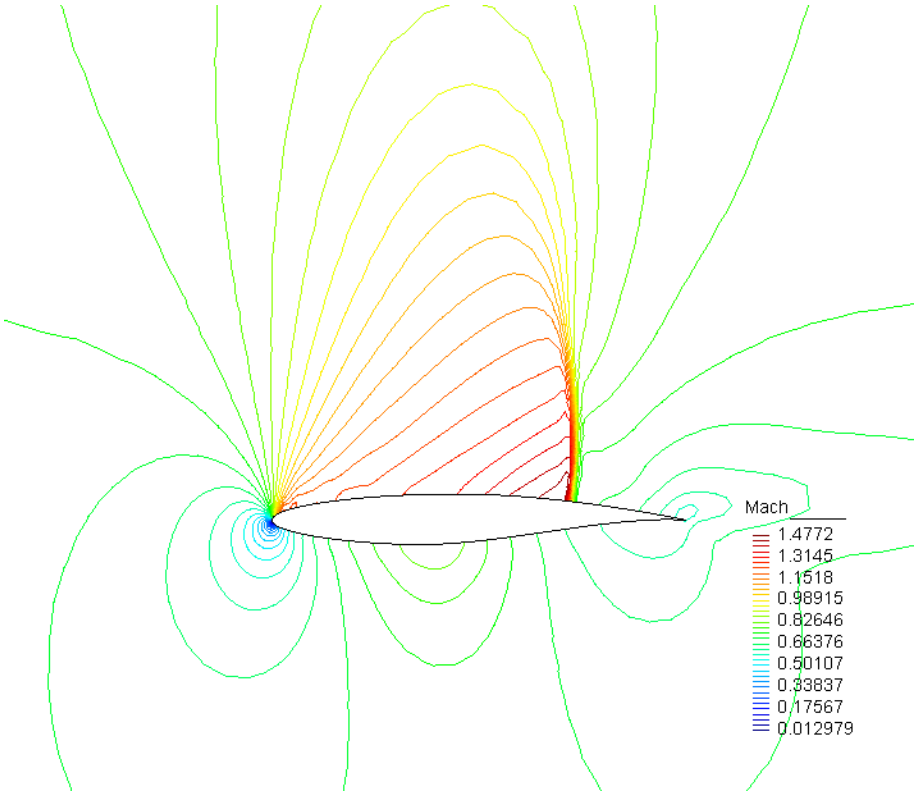

Figure 33: Mach number isolines in the near-field of the RAE 2822 airfoil, M_∞=0.75 and α=3º.

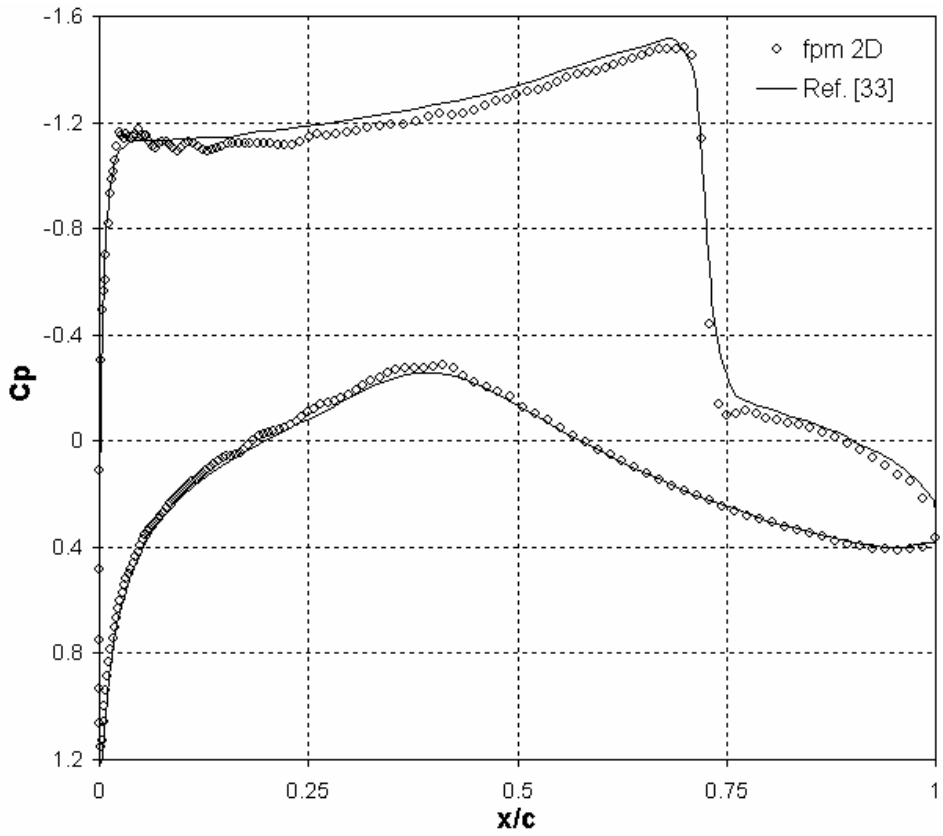

Figure 34: Cp distribution on the airfoil RAE 2822. A comparison between the FP calculation and the numerical results [33]. M_∞=0.75 and α=3º.

8.3 Three-dimensional examples

8.3.1 Subsonic flow around a sphere

In this example, subsonic flow around a sphere is solved for a freestream Mach number M_∞ = 0.2. The computational domain is discretized by a non-structured distribution of 30013 points and second-order spatial approximations are calculated in clouds of points where 30 ≤ np ≤ 40. A third-order MUSCL scheme with the Van Albada limiter and a three-stage time integration scheme are chosen for computing this numerical example. Next, the Cp and Mach number isolines on the sphere are shown in Figure 35.

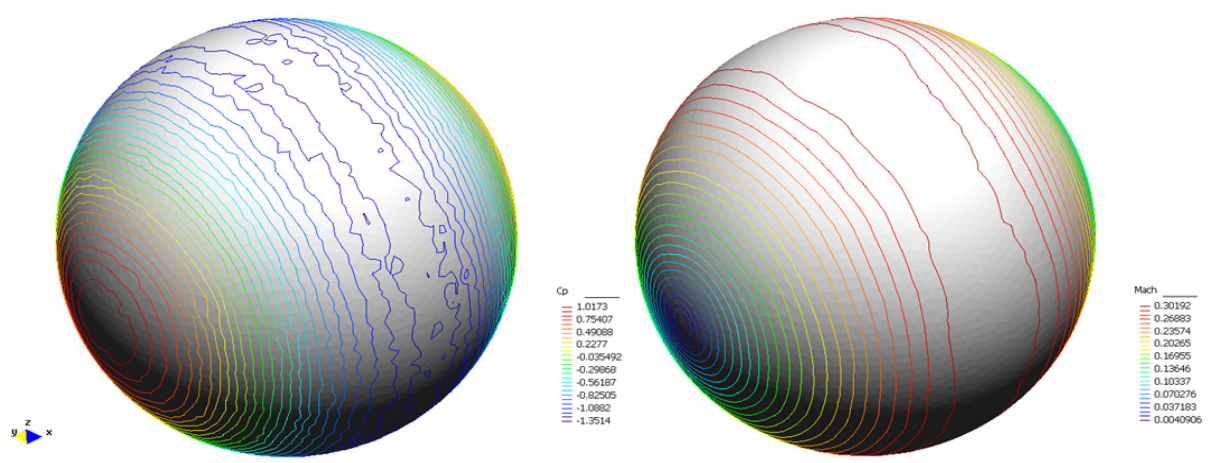

Figure 35: Cp and Mach number isolines on the sphere, M_∞ = 0.2.

The calculated Cp distribution, along a cut in the streamwise direction on the sphere, is compared with analytical results in Figure 36.

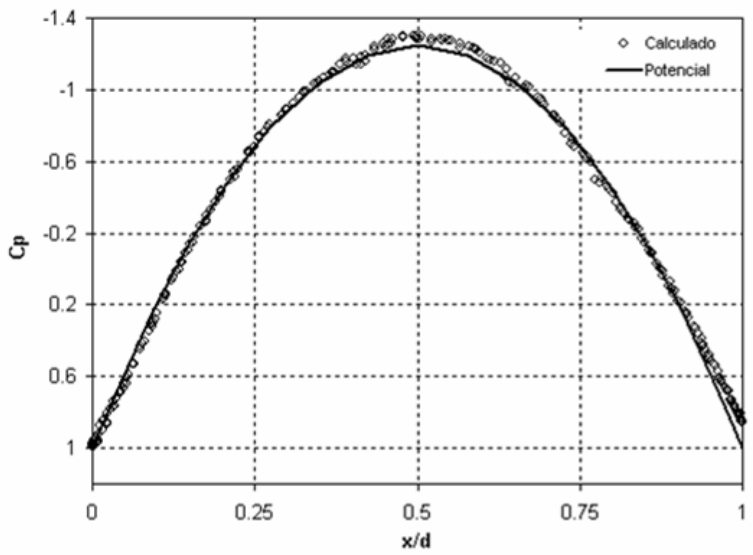

Figure 36: Cp distribution on the sphere along a cut in the streamwise direction. A comparison between the FP results and the analytical solution, M_∞ = 0.2

In Figure 36 a reasonable agreement between the computed and reference results can be observed. Only a small discrepancy is detected at the sphere suction points due to the differences existing between the computational models here compared. Note that the separation point on the sphere, obtained by the FP calculation, is almost coincident with the analytical rear stagnation point. This fact gives a cue of the low inherent dissipation of the proposed numerical scheme.

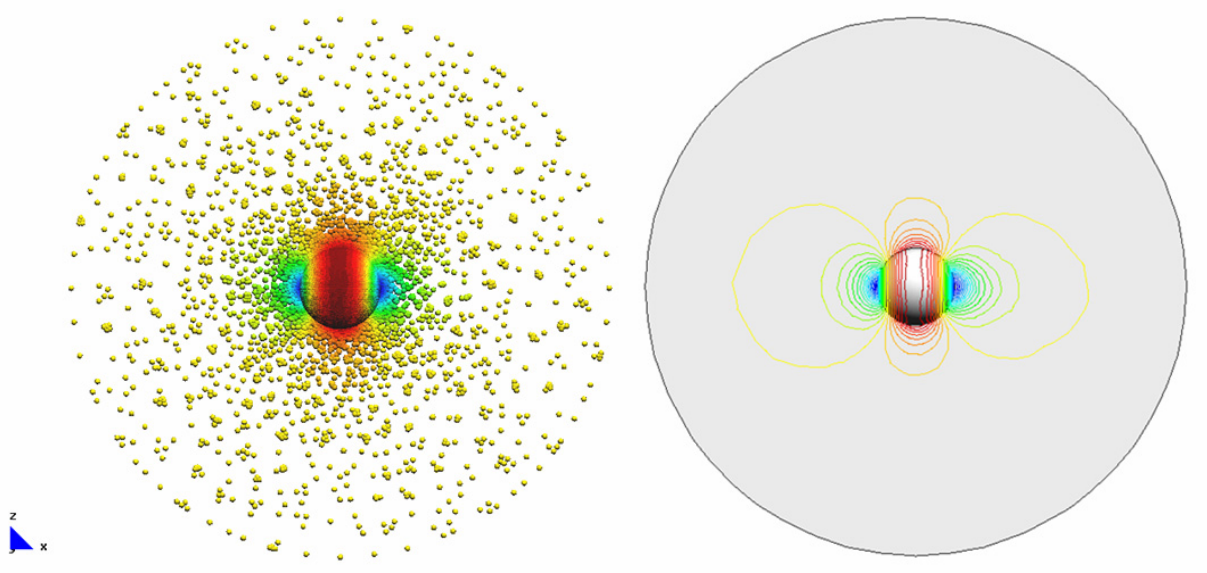

Figure 37: The symmetry plane of the problem. Left: points displaying Mach number results; Right: Mach number isolines. $M_\infty = 0.2$

8.3.2 Transonic flow over the ONERA M6 wing

This validation test [34] was developed by the ONERA Aerodynamics Department in 1972 with the objective of providing experimental support for studies concerning transonic flows at high-Reynolds number. Since then, these experimental results, which cover a wide range of subsonic and transonic flows, have become in classical reference data for code validation assessments. In this example we solve the test case # 2308 presented in [34]. This case concerns transonic flow over the ONERA M6 wing set at an incidence angle $\alpha = 3.06º$. The freestream Mach number is $M_\infty = 0.84$ and the Reynolds number is Re = 11.7E6. The most relevant data about this test case can be found in [35].

Due to the fact that in the present work we are solving the Euler equations, our simulation assumes the fluid to be inviscid. The computational domain is discretized by an unstructured distribution of 512141 points and second-order approximation bases are employed for calculating the shape functions and their derivatives in clouds where $30 < np < 45$. The flow

solver uses a third-order MUSCL extrapolation scheme in conjunction with the Van Albada limiter and a three-stage time integration scheme is employed for advancing the solution in time. Next, the Cp and the Mach number fields on the wing and the symmetry plane are shown in Figure 37 and Figure 38 respectively.

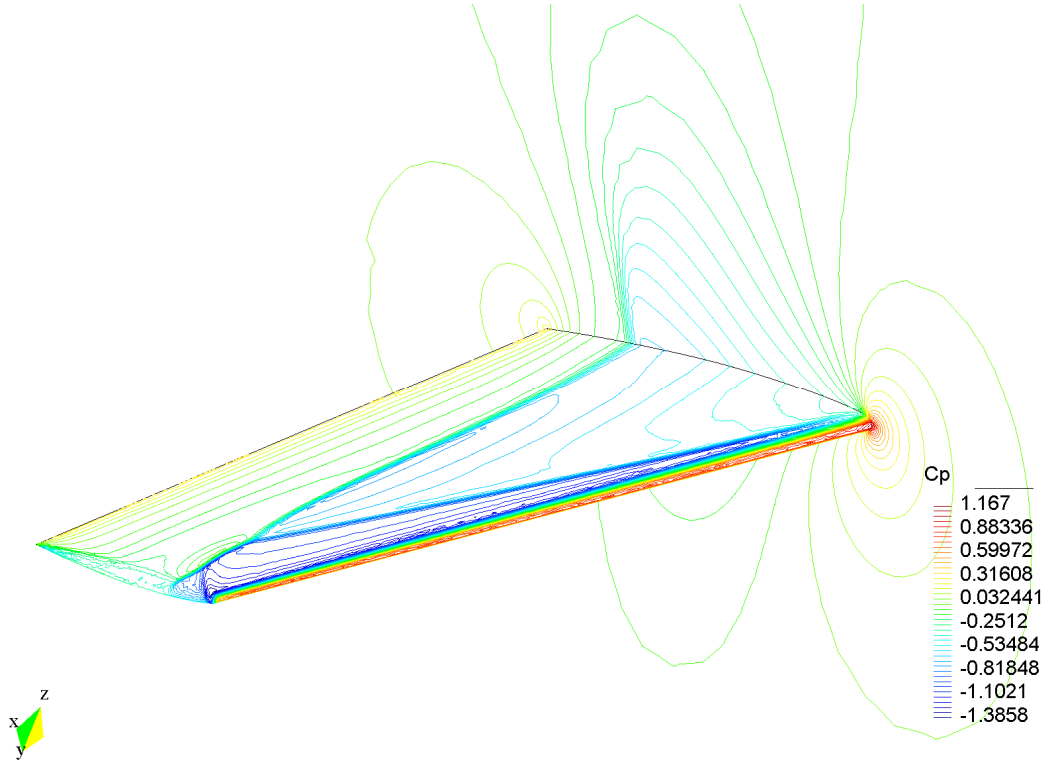

Figure 37: Cp isolines on the upper surface of the ONERA M6 wing and the symmetry plane. $M_\infty = 0.84$ and $\alpha = 3.06º$.

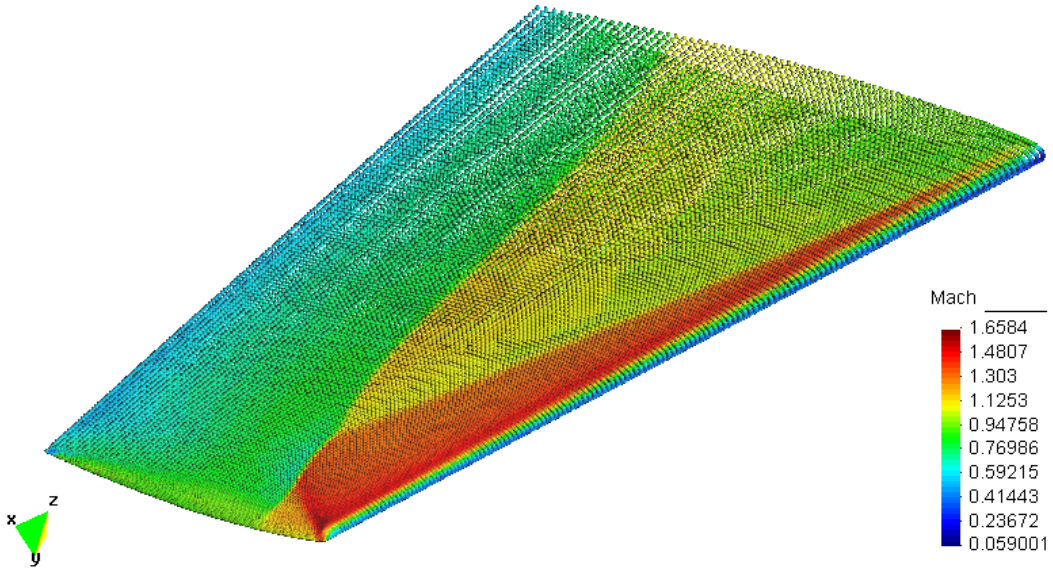

Figure 38: Surface discretization of the ONERA M6 wing. Coloured points display Mach number values. $M_\infty=0.84$ and $\alpha=3.06º$.

The discretization in an x-z plane at the 44 percent of the semispan is presented in Figure 39. Then, a comparison between computed and experimental Cp distributions along a cross-section of the wing, located at the same spanwise station, is shown in Figure 40.

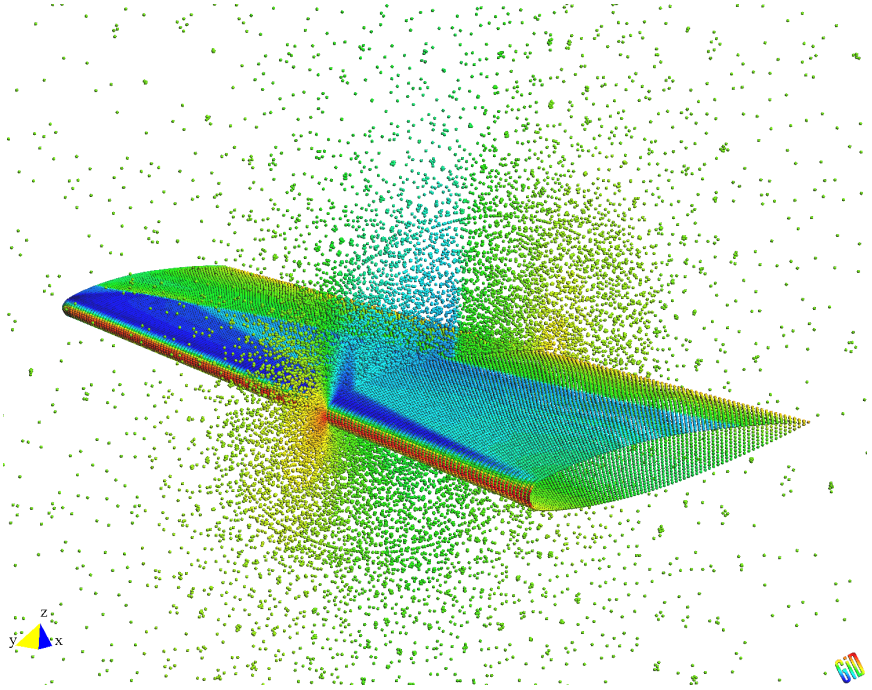

Figure 39: The x-z plane passing through a spanwise station 2 y/b = 0.44 (coloured points display Cp values). ONERA M6 wing, M_∞=0.84 and α=3.06º.

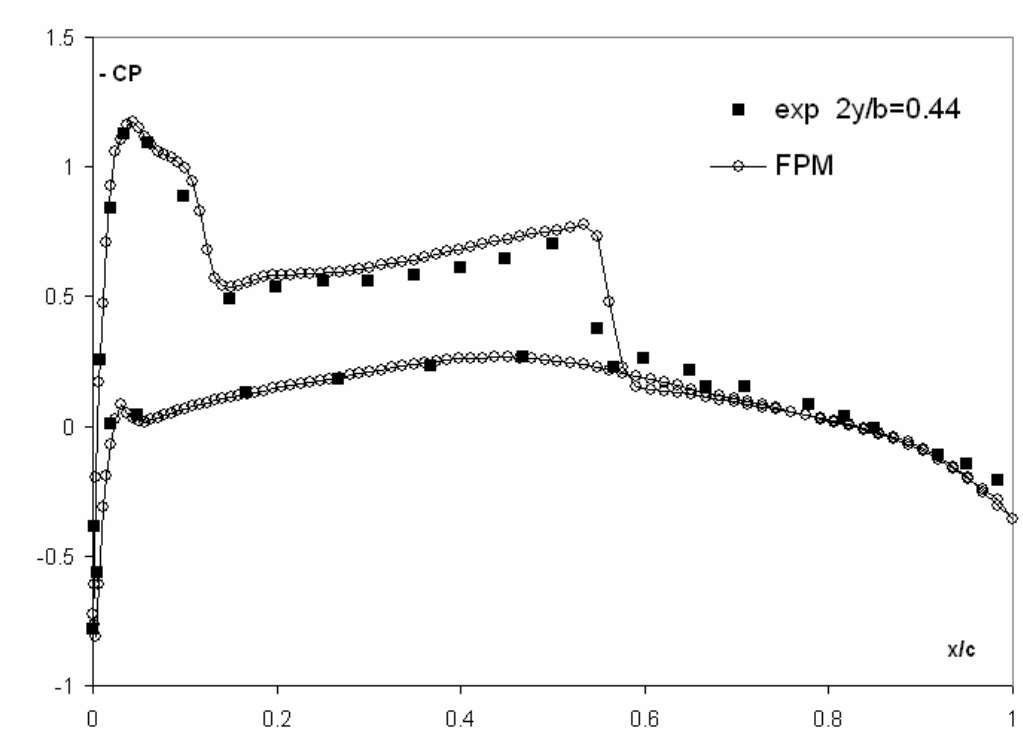

Figure 40: A comparison between computed and experimental Cp distribution along a cross-section located at the spanwise station 2 y/b = 0.44. ONERA M6 wing, M_∞=0.84 and α=3.06º.

Similarly, the discretization in an x-z plane located at the 95 percent of the semispan and the comparison of Cp distributions along a cross-section at the same spanwise station are shown in Figure 41 and Figure 42 respectively.

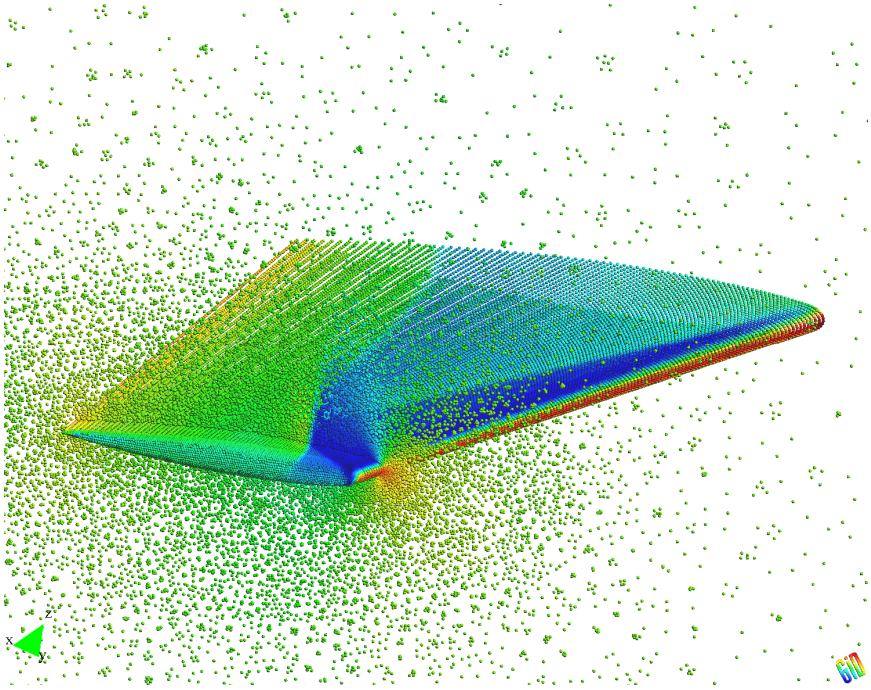

Figure 41: An x-z plane passing through a spanwise station located at 2y/b = 0.95. The coloured points display Cp values. ONERA M6 wing, M_∞=0.84 and α=3.06º.

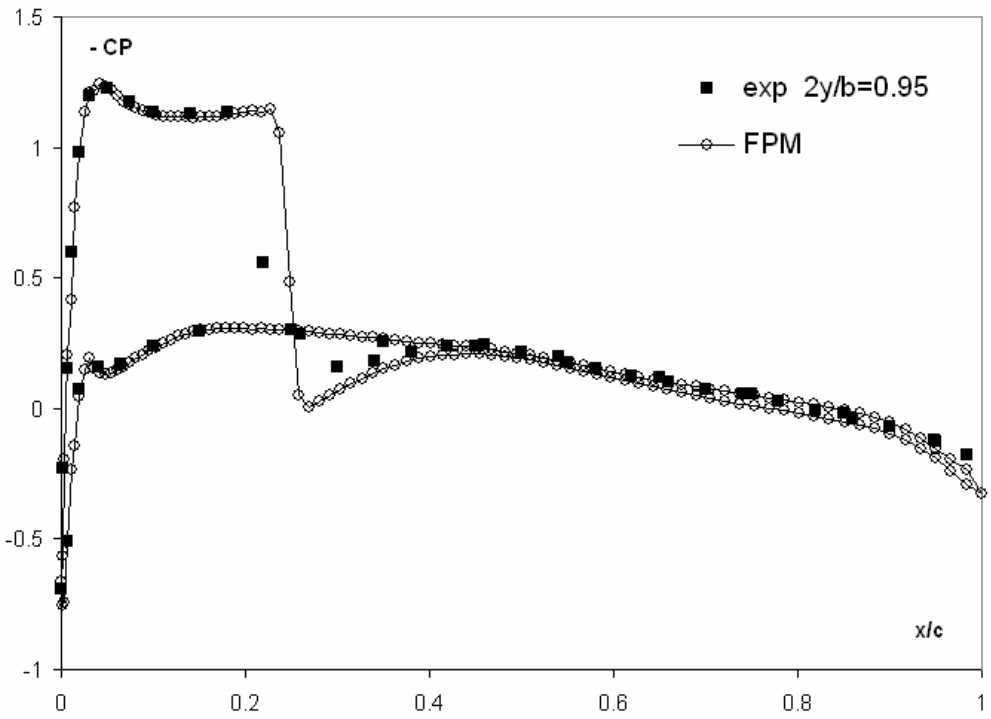

Figure 42: A comparison between computed and experimental Cp distribution along a cross-section located at the spanwise station 2 y/b = 0.95. ONERA M6 wing, M_∞=0.84 and α=3.06º.

A good agreement between computed and experimental results can be observed in Figures 40 and 42. As it was expected, the inviscid computation gives a shock wave which is slightly stronger than the true shock wave and is located close behind the latter.

8.3.3 Transonic flow over a NACA wing-body configuration

This example concerns the computation of a transonic flow over a wing-body configuration [36]. The wing has a sweepback $\Lambda = 45^{\circ}$ measured with respect to the 25-percent chord line, an aspect ratio $A = 4$, a taper ratio $\lambda = 0.6$ and it has not geometric twist. The airfoil section is a NACA 65A006 constant along the wing span. The body has a circular cross section and a fineness ratio of 10. Moreover, its rear part is attached to a sting which supports the model in the wind tunnel test chamber. In the numerical computation presented here, the freestream Mach number is $M_{\infty} = 0.7$ and the model is set at an incidence angle $\alpha = 2^{\circ}$. The discretization of the computational domain consists of an unstructured distribution of 336042 points and second-order approximations are built on clouds with $35 \leq np \leq 45$. The flow solver uses a third-order MUSCL extrapolation, in conjunction with the Van Albada limiter, and a three-stage time integration scheme is employed to advance the solution in time. Next, the surface discretization over the model and the symmetry plane is shown in Figure 43.

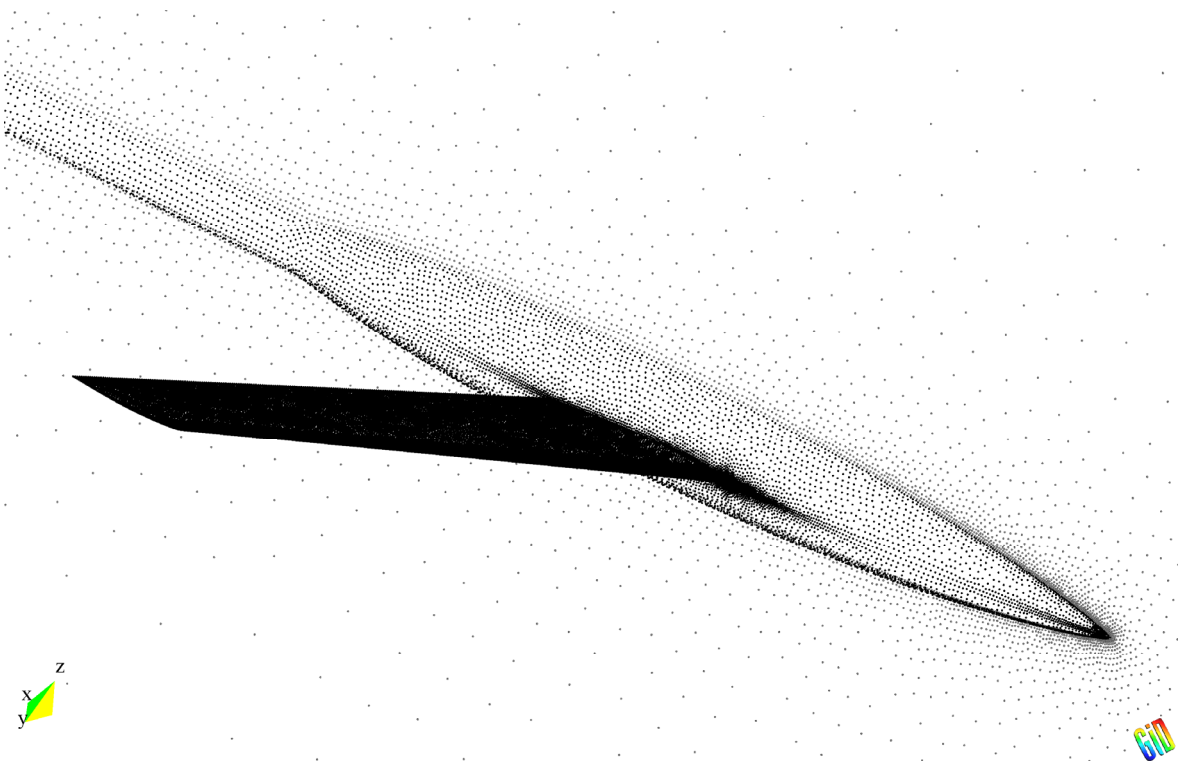

Figure 43: Surface discretization for the NACA wing-body and the symmetry plane.

Next, the Cp and Mach number isolines computed on the model and the symmetry plane are presented in Figure 44 and Figure 45 respectively.

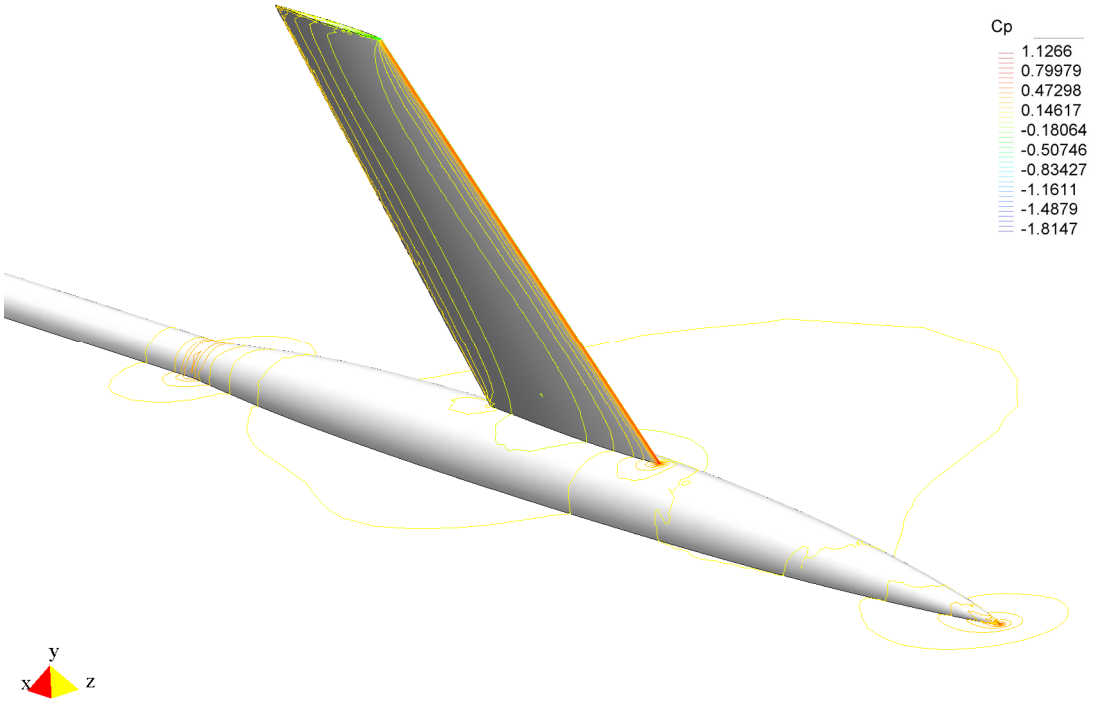

Figure 44: Cp isolines on the NACA wing-body and the symmetry plane. $M_\infty = 0.70$ and $\alpha = 2.0^o$.

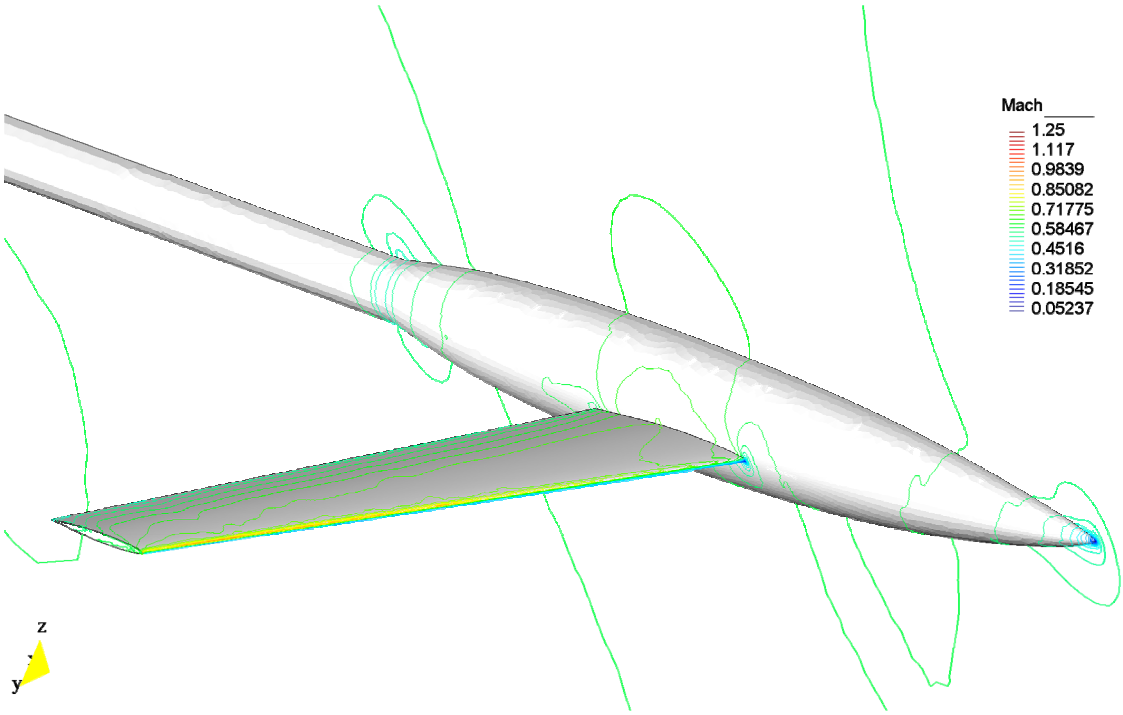

Figure 45: Mach number isolines on the NACA wing-body and the symmetry plane. $M_\infty = 0.70$ and $\alpha = 2.0^o$.

The Cp distribution on the wing, calculated along two streamwise stations located at 2y/b = 0.4 and 2y/b = 0.8, are compared with reference experimental results [37] in Figure 46 and Figure 47. Also here, a good agreement between numerical and experimental results can be observed.

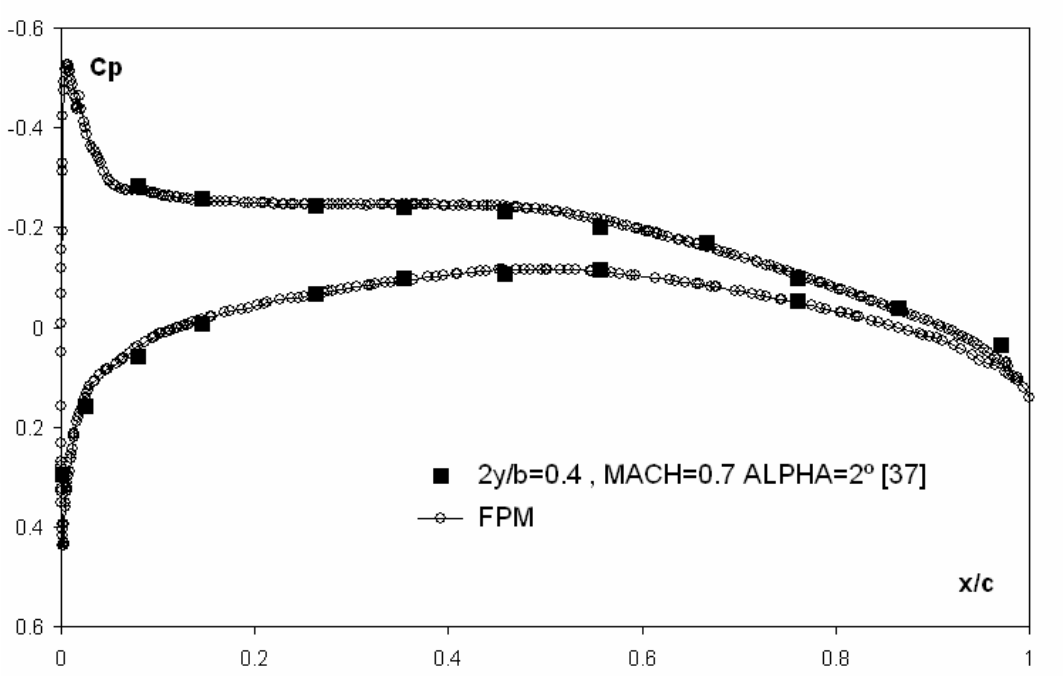

Figure 46: A comparison between computed and experimental Cp distribution along a cross-section located at the spanwise station 2 y/b = 0.40. NACA wing body [37], $M_\infty = 0.70$ and $\alpha = 2.0º$.

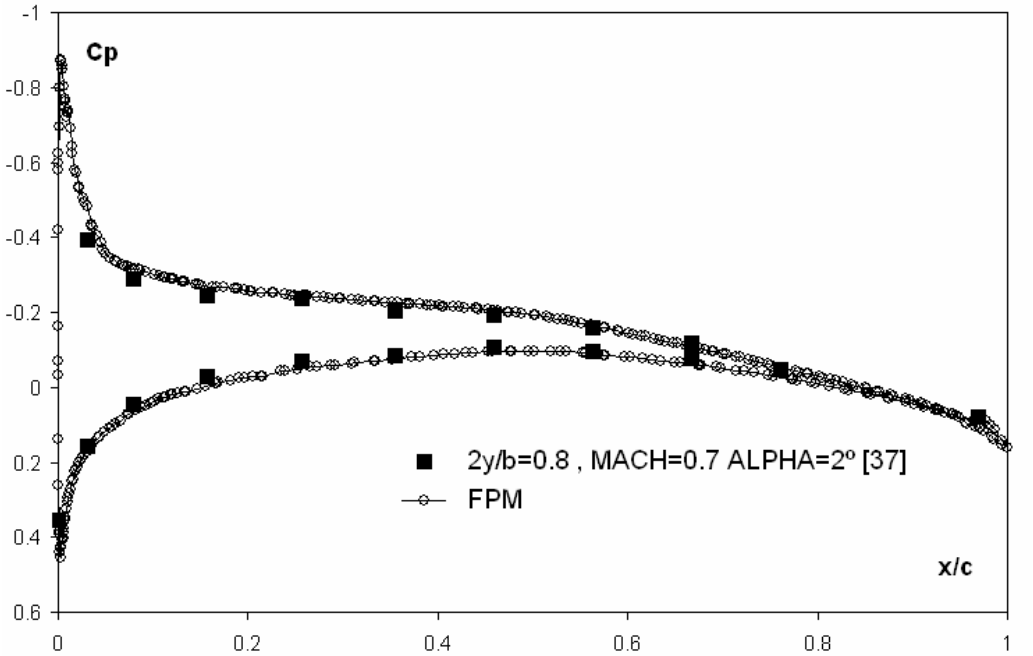

Figure 47: A comparison between computed and experimental Cp distribution along a cross-section located at the spanwise station 2 y/b = 0.80. NACA wing body [37], $M_\infty = 0.70$ and $\alpha = 2.0º$.

9. AN *h*-ADAPTIVE PROCEDURE FOR FINITE POINT CALCULATIONS

There are several reasons that explain the appeal of adaptive strategies in the different fields of numerical simulation. Adaptivity reduces the effort needed to obtain a proper discretization for numerical calculations as regards man-hours, CPU-time and memory requirements significantly. Also, adaptive procedures make the accurate computation of local features of the flow easier, especially when we do not have *a priori* information concerning the solution, and become essential for non-stationary problems involving moving discontinuities. The fact that meshless methods do not need to keep a conforming mesh makes them well-suited for adaptive procedures. Next, an FP adaptive procedure, which works in the same way for two and three-dimensional problems, is presented. Its bases are summarized in the following.

9.1 The indicator

In this work the solution at a previous time-step is employed with the aim of identifying local clouds of points where new points should be inserted or existing points could be removed from the computational domain. This is accomplished by the following normalized indicator that evaluates, in an approximate manner, the curvature of the solution at each point

$$\varphi_i = \frac{1}{\varphi_m}\sum_{j=1}^{ns}\left|\boldsymbol{l}_{ji}\cdot\left(\nabla\rho_j - \nabla\rho_i\right)\right| \quad ; \quad \varphi_m = \max\left(\varphi_i\right) \quad i=1,n \tag{104}$$

In the expression above *ns* is the number of points in the first layer of nearest neighbours (already obtained in the local cloud construction stage) and $\boldsymbol{l}_{ji} = \boldsymbol{x}_j - \boldsymbol{x}_i$ is the vector linking each pair of points ($\boldsymbol{x}_i$,$\boldsymbol{x}_j$). In particular cases, the proposed normalization causes a lack of sensitivity to relative small gradients in the flow field. When this happens, it could be useful to avoid the normalization setting $\varphi_m = 1$ or take another local maximum for normalizing the indicator. Based on the indicator (104); new points are inserted around $\boldsymbol{x}_i$ when $\varphi_i > \varphi_{max}$. Conversely, the point $\boldsymbol{x}_i$ is removed from the computational domain if $\varphi_i < \varphi_{min}$. The limits φ_{max} and φ_{min} depend on the problem under consideration; in the numerical example presented here $\varphi_{max} \approx 0.1$ and $\varphi_{min} \approx 0.005$ are chosen.

9.2 The strategy

The adaptive procedure that we propose can be reduced to three main steps: the insertion of new points, the removal of existing points and an update. The latter makes reference to the construction of the data associated to each new point and the re-construction of the data associated to an affected existing point respectively. We consider that an existing point is

affected when a new point falls inside its cloud or the spatial position of any point in its cloud changes due to smoothing.

9.2.1 Insertion of new points

When a point $\boldsymbol{x}_i$ is marked to refine ($\varphi_i > \varphi_{max}$), its Delaunay grid of nearest neighbours is used to calculate the Voronoi vertices surrounding $\boldsymbol{x}_i$. Next, new candidate points $\boldsymbol{x}_c$ are set at these vertices. Each of them is accepted if it meets the following requirements:

$\mathbf{r_1}$. The candidate point falls inside the triangle/tetrahedron (2-D/3-D), which generates the empty circumcircle/circumsphere centred on $\mathbf{x}_c$.

$\mathbf{r_2}$. The radius of the circumcircle/circumsphere r_c complies with $r_c < r_{min}$ being r_{min} a user-defined parameter which stands for the minimum admissible distance between two points.

$\mathbf{r_3}$. The distance from the candidate point $\boldsymbol{x}_c$ to another new point previously accepted is greater than the $\max(r_{min},\alpha d_e)$, where d_e is the minimum edge of the triangle/tetrahedron which originates the circumcircle/circumsphere and $\alpha \approx 0.75$ is a user-defined parameter.

If any of the edges/triangles of the local Delaunay grid of nearest neighbours lies on the boundaries, a new candidate boundary point is obtained as an average of the position of the points defining this edge/triangle. The candidate boundary point is accepted if the distance to the nearest point is greater than r_{min}. In our algorithm we perform the boundary refinement first and then we refine the discretization into the domain. Note that when the initial boundary discretization is very coarse, the straightforward procedure proposed for boundary refinement could deteriorate the boundaries, resulting in a lack of reliability on the computational model. In such cases, the position of new boundary points could be obtained using a higher-order interpolation of the underlying existing boundary points. Figure 48 sketches the refinement procedure for a bi-dimensional cloud of points.

9.2.2 Removal of existing points

The removal of points is restricted only to existing points that have been inserted in previous refinement levels. In other words, the initial set of points (coarse discretization) is conserved through the calculation, although the spatial position of these points could change due to smoothing. This criterion avoids several time-consuming verifications and guarantees a minimum appropriate geometrical support for the calculation.

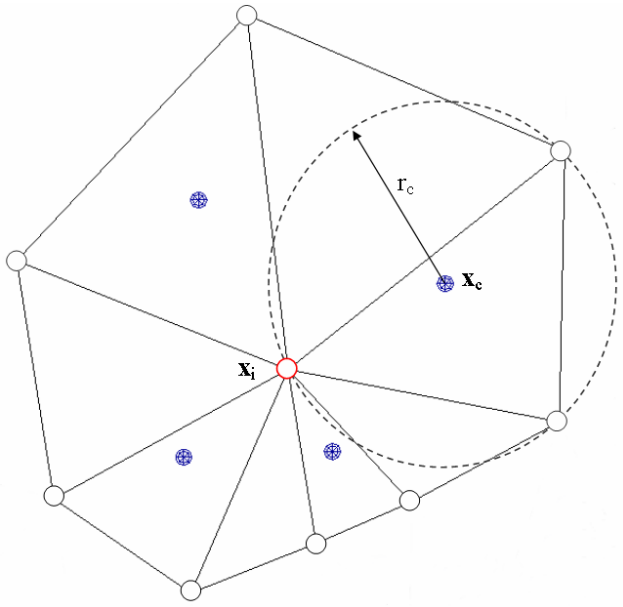

Figure 48: Refinement of a bi-dimensional cloud of points. The filled points $\boldsymbol{x}_c$ satisfy the requirements r_1-r_3 and, in consequence, are inserted around the star point $\boldsymbol{x}_i$.

9.2.3 Update

Once the insertion and removal of points is finished, a few steps of a Laplacian smoothing [31] are carried out on the affected area. This is particularly helpful when points have been removed in large quantities. Next, the clouds of points and shape functions concerning the new points are constructed. In addition, the data concerning existing clouds of points affected by the insertion of new points or smoothing is re-constructed. Finally, the flow variables at new points are calculated as an average of the variables at their existing nearest neighbours.

9.3 A numerical example: adaptive calculation of a transonic flow around an airfoil

This numerical example concerns the computation of a transonic inviscid flow around a NACA 0012 airfoil. The freestream Mach number is $M_\infty = 0.8$ and the incidence angle is $\alpha = 1.25º$. Second-order spatial approximations are calculated in clouds where $15 \leq np \leq 20$ and a third-order MUSCL extrapolation, in conjunction with the Van Albada limiter, is adopted for the flow solver. The time integration is performed by means of a three-stage scheme.

The initial spatial discretization consists of an unstructured distribution of 976 points and a distribution of 4938 points is obtained after 15 refinement levels. The initial and final discretizations are shown in Figures 49 and 50 respectively. Note that the adaptive procedure captures all the flow features with precision. In spite of the normalization adopted for the refinement indicator (104), our adaptive strategy solves the strong shock wave located on the

upper side of the airfoil as well as the weak shock on its lower side and the leading and trailing edge regions appropriately.

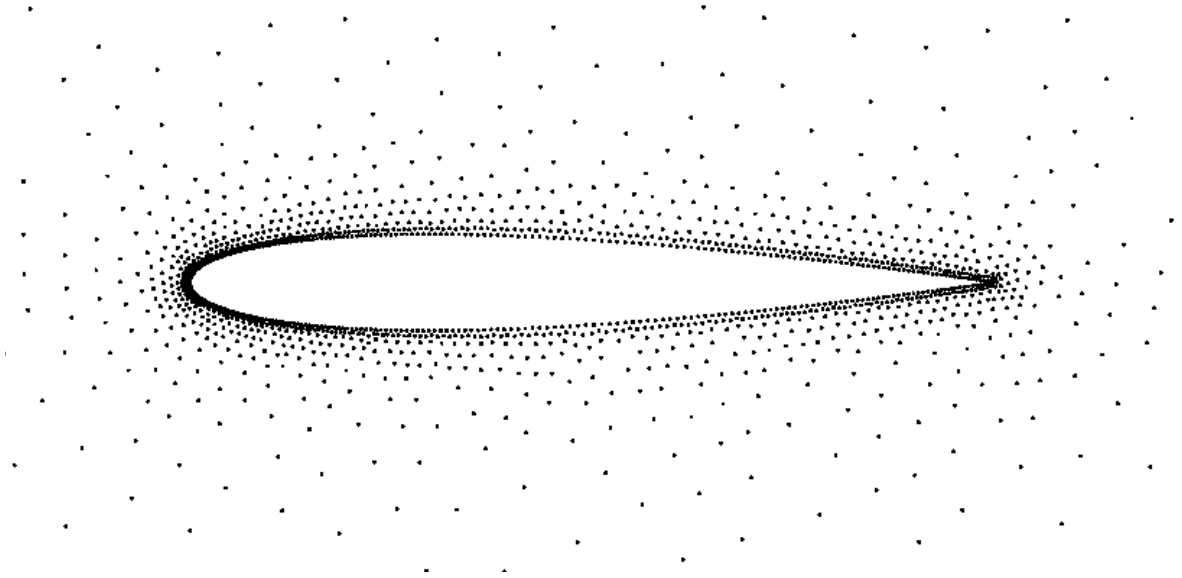

Figure 49: A view of the initial discretization in the proximity of the NACA 0012 airfoil (976 points)

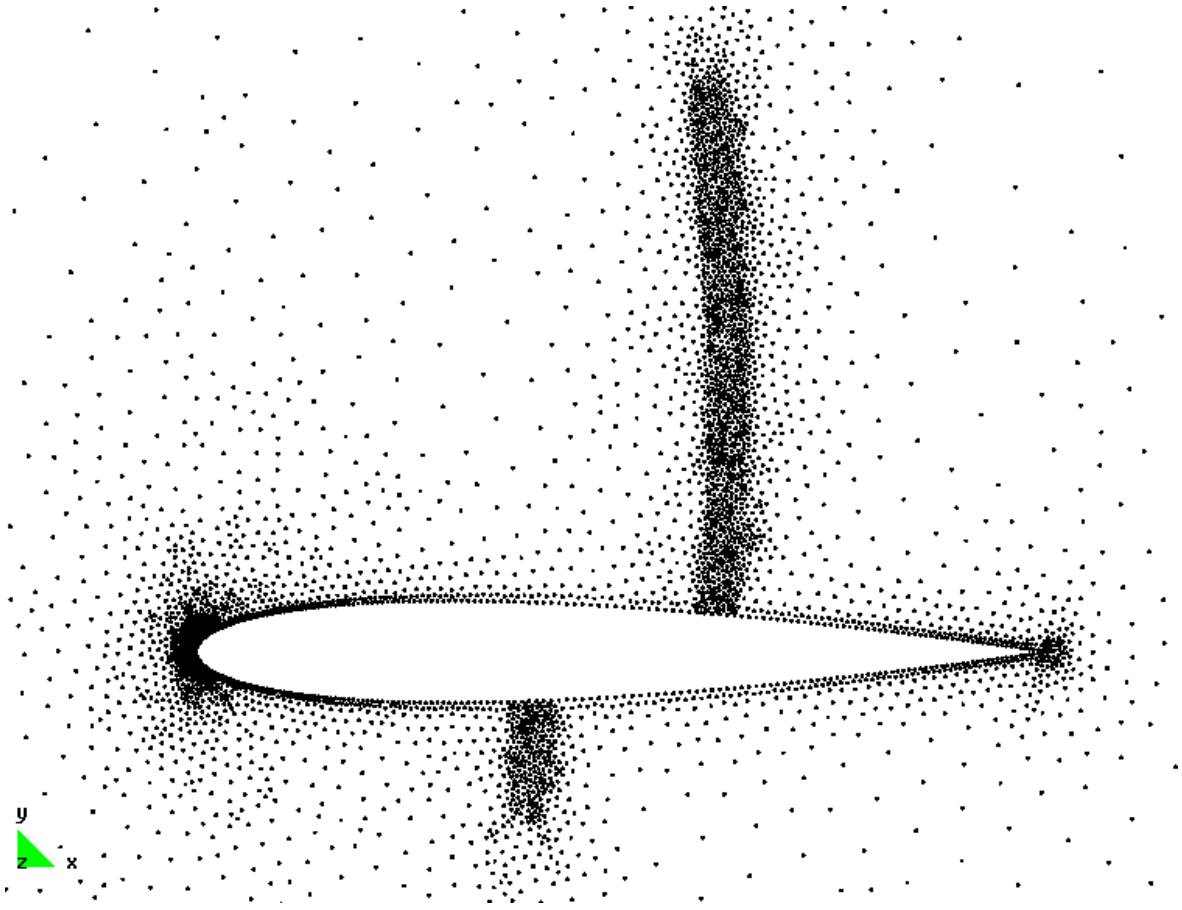

Figure 50: A view of the final adapted discretization in the proximity of the NACA 0012 airfoil obtained after 15 refinement levels (4938 points).

Next, the Cp and Mach number isolines around the airfoil, calculated with the final adapted discretization, are shown in Figures 51 and 52.

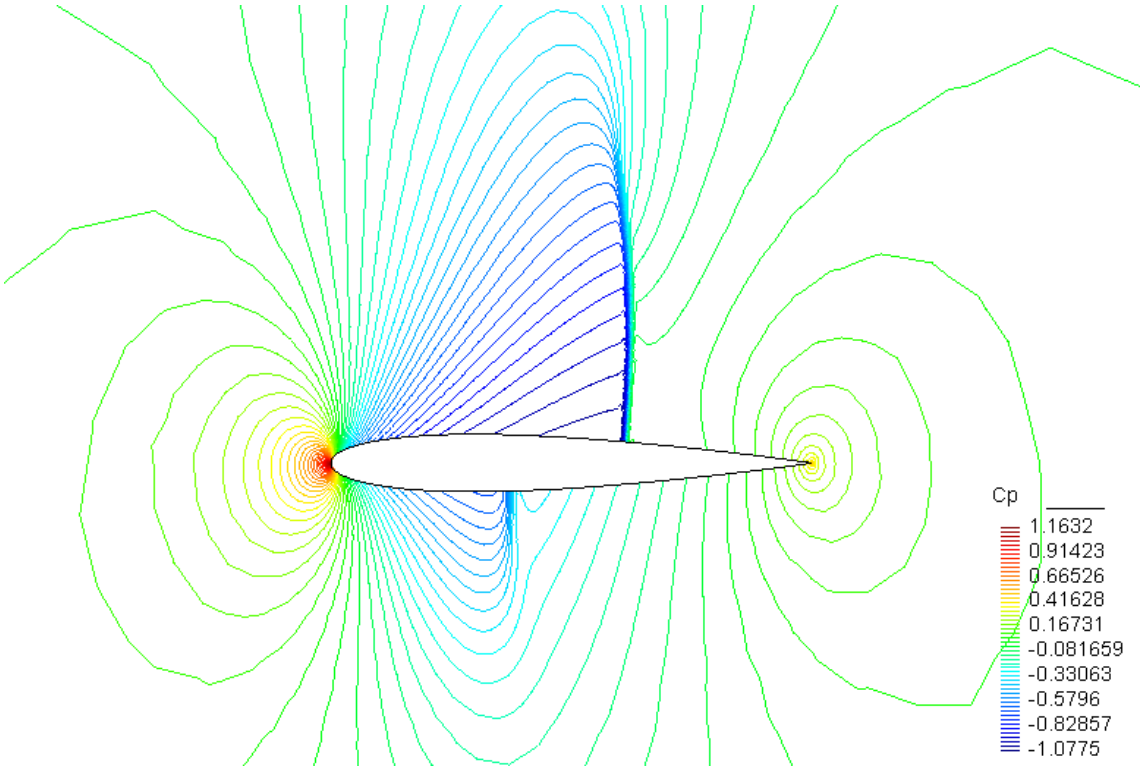

Figure 51: Cp isolines in the near-field of the NACA 0012 airfoil obtained with the final adapted discretization. M_∞=0.80 and α=1.25º

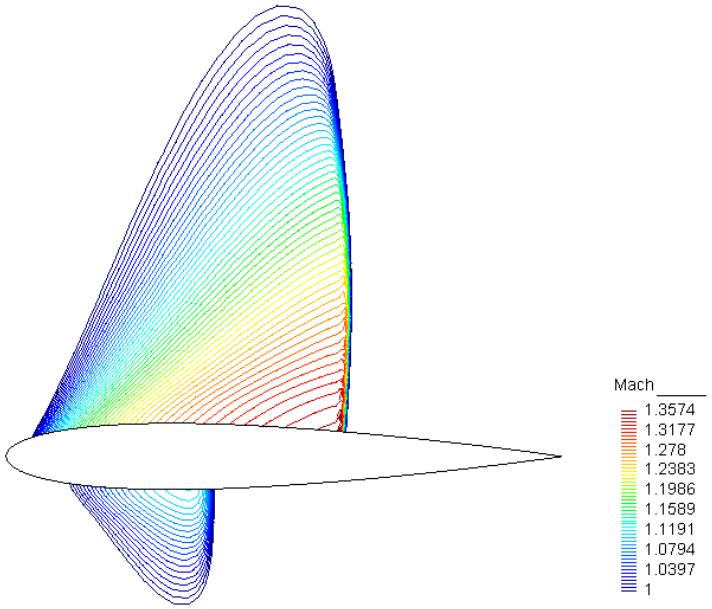

Figure 52: Mach number isolines in the near-field of the NACA 0012 airfoil obtained with the final adapted discretization. Only supersonic flow areas on the lower and upper side of the airfoil are shown. M_∞=0.8 and α=1.25º

Figure 53 below shows the computed Cp distribution on the airfoil compared to the numerical reference results [37], where a good agreement is observed. Finally, the convergence history of the present calculation case is presented in Figure 54.

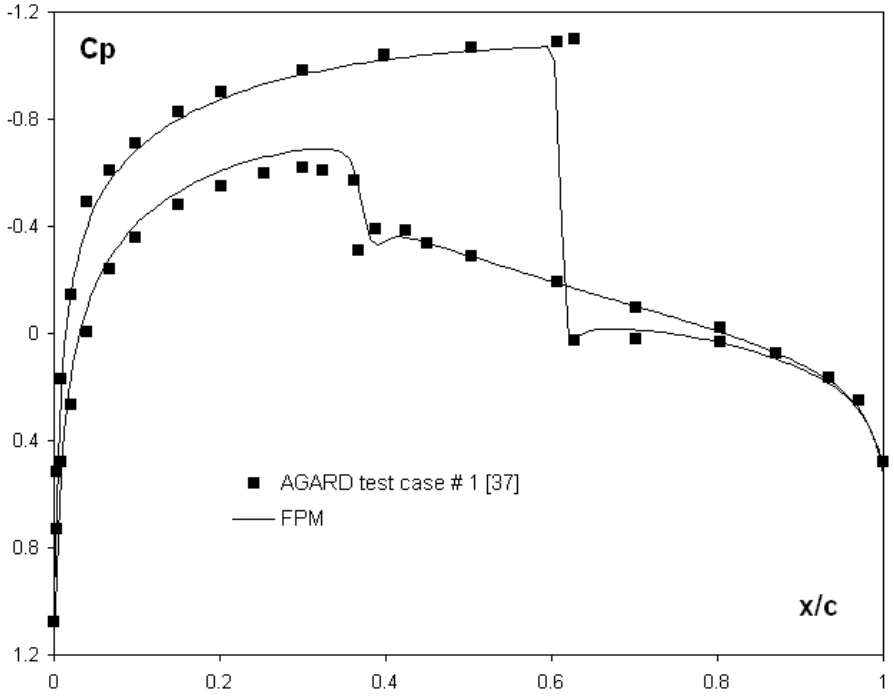

Figure 53: Cp distribution on the NACA 0012 airfoil obtained with the final adapted discretization. A comparison between computed and numerical reference results [37]. M_∞=0.80 and α=1.25º.

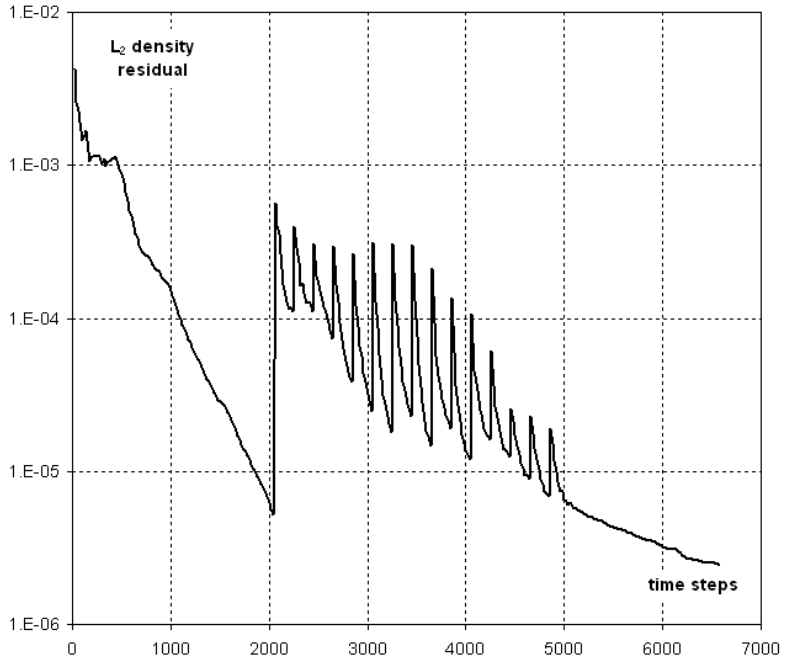

Figure 54: Convergence history of the NACA 0012 airfoil calculation (15 refinement levels). M_∞=0.80 and α=1.25º

Figure 54 above shows the complete process of the adaptive numerical computation. When the simulation starts, 50 time steps are performed using the low-order scheme in order to initialize the flow field around the airfoil. Then, the flow solver switches to the high-order scheme and, even though it affects the convergence, this is recovered after a few time steps. For a value of the density residual of 0.5E-5 the first refinement level is performed. Then, consecutive refinement levels are carried out every 200 time steps. Note that the peaks in the convergence curve correspond to each refinement level performed during the computation.

10. CONCLUSIONS

This work has been intended to contribute to a major investigation into the capabilities of the Finite Point method to deal with three-dimensional applications concerning compressible fluid flow problems. As a first step towards this ultimate aim, we have developed a suitable Finite Point formulation for this kind of analysis, which has been the main goal of the present research work.

In the introduction we have talked about *robustness* and *efficiency*. Later, we made reference to certain topics in numerical simulation which represent good 'opportunities' for the development and promotion of meshless methods. The spatial approximation procedure and the flow solver here presented tend towards *robustness* and several numerical experiments, some of them reported in Section 8, confirm that. Probably, it is very difficult to achieve a meshless technique which offers the robustness and stability of a mesh-based low-order method. However, in some particular cases, the advantages of using high-order approximations with a minimum computational cost could compensate for that.

Regarding high-order Finite Point approximations, some three-dimensional potential flow calculations have been carried out in Section 6. In spite of the fact that the preliminary results are hopeful, certain non-expected features, which are generally related to particular settings of the approximation parameters, should be carefully studied. Moreover, high-order approximations were tested on three-dimensional compressible flow problems. Even though third and fourth-order approximations give good results, they also involve a substantial increase of the computational requirements (mainly CPU-time) because extensive clouds of points are needed. This fact makes third and fourth-order approximations non-appropriate for large explicit computations. Linear approximations were also tested and, although they involve a minimum computational cost, the accuracy of the results is not good enough. Numerical experiments show that second-order approximations give the best accuracy-cost

ratio and, consequently, they were used in all the numerical examples presented in this research work. Notice, however, that the previous comments about high-order approximations are preliminary and further specific tests should be carried out.

In Section 9, and with the aim of exploiting the 'opportunities' that meshless methods have, a reliable adaptive technique which has a very low computational cost has been developed. The three-dimensional extension of this technique is straightforward and we hope to report its success in the near future.

So far, nothing has been said about *efficiency*. At present we still lack precise comparisons between the performance of our FP methodology and conventional discretizations techniques. However, we estimate that the computational cost of a three-dimensional Finite Point computation would exceed a similar FE-based computation, in the best of the cases, by a cost factor of 3. As it can be seen, *efficiency* is another pending matter. However, notice that the comparative estimation we just mentioned concerns only the time needed in order to solve the equations starting from complete discrete data and excludes all the pre-process stages (computational model discretization). Even though the computational requirements of the overall simulation process are difficult to compare, undoubtedly, meshless approaches have some advantages over mesh-based approaches when performing the pre-process stages. Consequently, if all the computational processes involved in the numerical simulation are considered, the previously estimated cost factor would be reduced.

The paragraphs above suggest some important investigation lines which call for our immediate attention. Advantages and disadvantages of using high-order approximations are not clear yet and specific tests will have to be performed in order to extract conclusions about this subject. In addition, the proposed adaptive technique should be coded for three-dimensional problems and its performance should be evaluated. As it was mentioned before, it is possible that high-order approximations and adaptivity could compensate for a higher computational cost but, at any rate, an improvement of the computational efficiency of the present methodology is still essential. Last but not least, we will also to carry out need performance comparisons between our meshless methodology and similar mesh-based formulations (also including geometry pre-process stages).

All in all, we can say that the present research work has allowed us to see that the Finite Point method has a high potential that needs to be exploited and, so far, the results obtained are much encouraging. However, we also need to say that, as it was revealed, the FPM has certain

weaknesses which need to be dealt with in order to turn this method into a suitable tool for the analysis of compressible fluid flows.

ACKNOWLEDGEMENTS

The first author would like to acknowledge the support of Alβan Program, the European Union Programme of High Level Scholarships for Latin America, scholarship Nº E04D027284AR. Roberto Flores's many valuable contributions to this work are also gratefully acknowledged.

REFERENCES

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

[2] Fries T., Matthies H. Classification and overview of meshfree methods. Department of Mathematics and Computer Science, Technical University of Braunschweig. Inf. 2003-3, 2004.

[3] Li S., Liu W. K. Meshfree and particle methods and their applications. *Applied Mechanics Reviews* 2002; **55**:1-34.

[4] Duarte C. A. A review of some meshless methods to solve partial differential equations. Texas Institute for Computational and Applied Mathematics, TICAM Report 95-06

[5] Oñate E., Idelsohn S., Zienkiewicz O. C., Taylor R. L., Sacco C. A Finite Point Method for analysis of fluid mechanics problems. Applications to convective transport and fluid flow. *International Journal for Numerical Methods in Engineering* 1996; **39**: 3839-3866.

[6] Oñate E., Idelsohn S., Zienkiewicz O. C., Fisher T. A Finite Point Method for analysis of fluid flow problems. *Proceedings of the 9th Int. Conference on Finite Elements Methods in Fluids*, Venize, Italy, 1995; 15-21.

[7] Oñate E., Idelsohn S., Zienkiewicz O. C., Taylor R. L., Sacco C. A stabilized Finite Point Method for analysis of fluid mechanics problems. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**: 315-346.

[8] Oñate E., Sacco C., Idelsohn S. A Finite Point Method for incompressible flow problems. *Computing and Visualization in Science* 2000; **3**: 67-75

[9] Fischer T. R. A contribution to adaptive numerical solution of compressible flow problems. Doctoral Thesis, Universitat Politècnica de Catalunya, 1996.

[10] Sacco C. 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, 2001.

[11] Löhner R., Sacco C., Oñate E., Idelsohn S. A Finite Point Method for compressible flow. *International Journal for Numerical Methods in Engineering* 2002; **53**:1765-1779

[12] Oñate E., Perazzo F., Miquel J. A Finite Point Method for elasticity problems', *Computers & Structures* 2001; **79**: 2151-2163.

[13] Oñate E. Derivation of stabilized equations for numerical solution of advective-diffusive transport and fluid flow problems. *Computer Methods in Applied Mechanics and Engineering* 1998; **151**:233-265.

[14] Boroomand B., Tabatabaei A. A., Oñate E. Simple modifications for stabilization of the finite point method. *International Journal for Numerical Methods in Engineering* 2005; **63**:351-379

[15] Ortega E., Oñate E., Idelsohn S. An improved finite point method for three-dimensional potential flows. To appear in *Computational Mechanics*

[16] Lancaster P., Salkauskas K. Surfaces generated by moving least squares methods. *Mathematics of Computation* 1981; **37**:141-158.

[17] Xiaozhong J., Gang L., Aluru N. R. Positivity conditions in meshless collocation methods. *Computer Methods in Applied Mechanics and Engineering* 2004; **193**:1171-1202.

[18] Demmel J. W. Applied numerical linear algebra. Society for Industrial and Applied Mathematics, 1997.

[19] Han W., Meng, X. Error analysis of the reproducing kernel particle method. *Computer Methods in Applied Mechanics and Engineering* 2001; **190**: 6157-6181.

[20] Liu W. K., Li S., Belytschko T. Moving least square reproducing kernel methods. (I) Methodology and convergence. *Computer Methods in Applied Mechanics and Engineering* 1997; **154**: 143-113.

[21] Liszka T. J., Duarte C. A. M., Tworzydlo W. W. hp-Meshless cloud method. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**: 263-288.

[22] Idelsohn S., Calvo N., Oñate E., Polyhedrization of an arbitrary 3D point set. *Computer Methods in Applied Mechanics and Engineering* 2003; **192**:2649-2667.

[23] Praveen C. A positive meshless method for hyperbolic equations. ARDB Centre of Excellence for Aerospace CFD, Department of Aerospace Engineering, Indian Institute of Science, Report 2004-FM-16, 2004

[24] Hirsch C., Numerical computation of internal and external flows. Volume 2. John Wiley

& Sons (1991)

[25] Laney C. B., Computational Gasdynamics. Cambridge University Press (1998)

[26] Roe P. L. Approximate Riemann solvers, parameter vectors and difference schemes. Journal of Computational Physics, 43:357-372 (1981)

[27] Turkel E., Improving the accuracy of central difference schemes. ICASE Report 88-53 (1988)

[28] Harten A., Hyman J. M., Self-adjusting grid methods for one-dimensional hyperbolic conservation laws. *Journal of Computational Physics*, **50**: 235-269 (1983)

[29] Kermani M. J., Plett E. G., Modified entropy correction formula for the Roe scheme. *American Institute for Aeronautics and Astronautics*, AIAA Paper 2001-0083 (2001)

[30] Van Leer B., Towards the ultimate conservative difference scheme. V, A second order sequel to Godunov's method. *Journal of Computational Physics* **32**:101-136 (1979)

[31] Löhner R., Applied CFD Techniques. John Wiley & Sons (2001)

[32] Lyra P. R. M., Morgan K., A review and comparative study of upwind biased schemes for compressible flow computation. Part III: Multidimensional extension on unstructured grids. Arch. Comput. Meth. Engrg 9: 207-256 (2002)

[33] A. Rizzi und H. Viviand, Ed., Numerical Methods for the Computation of Inviscid Transonic Flows with Shock Waves, *Notes on Numerical Fluid Mechanics*, Braunschweig, Wiesbaden, 1981, Vieweg.

[34] Schmitt V., Charpin F. Pressure distributions on the ONERA-M6-wing at transonic Mach numbers. Experimental data base for computer program assessment. Report of the Fluid Dynamics Panel Working Group 04, AGARD AR 138, 1979

[35] NPARC Alliance CFD verification and validation web site. Web page: http://www.grc.nasa.gov/WWW/wind/valid/archive.html

[36] Loving D., Estabrooks B., Transonic-wing investigation in the Langley 8-foot high-speed tunnel at high subsonic Mach numbers and at a Mach number of 1.2. Analysis of pressure distribution of wing-fuselage configuration having a wing of 45º sweepback, aspect ratio 4, taper ratio 0.6, and NACA 65A006 airfoil section. National Advisory Committee for Aeronautics. Research Memorandum NACA RM L51F07 (1951).

[37] Pulliam T. H., Barton J. T., Euler computations of AGARD Working Group 07 Airfoil Test Cases. AIAA 23rd Aerospace Summer Meeting, Reno, Nevada, AIAA Paper 85-0018 (1985).