A Finite Point Method for Adaptive Three-Dimensional Compressible Flow Calculation

E. Ortega
E. Oñate
S. Idelsohn

Publication CIMNE Nº-323, October 2008

A Finite Point Method for Adaptive Three-Dimensional Compressible Flow Calculation

E. Ortega
E. Oñate
S. Idelsohn

Publication CIMNE Nº-323, October 2008

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

A FINITE POINT METHOD FOR ADAPTIVE THREE-DIMENSIONAL COMPRESSIBLE FLOW CALCULATIONS

Enrique Ortega, Eugenio Oñate and Sergio Idelsohn[1]

International Center for Numerical Methods in Engineering (CIMNE)
Universidad Politécnica de Cataluña
Edificio C1, Campus Norte, UPC
Gran Capitán, s/n, 08034 Barcelona, España

Abstract. The Finite Point Method (FPM) is a meshless technique which is based on both, a Weighted Least-Squares numerical approximation on local clouds of points and a collocation technique which allows obtaining the discrete system of equations. The research work we present is part of a major investigation into the capabilities of the FPM to deal with three-dimensional applications concerning real compressible fluid flow problems. In the first part of this work, the upwind biased scheme employed for solving the flow equations is described. Secondly, with the aim of exploiting meshless capabilities, an *h*-adaptive methodology for two and three-dimensional compressible flow calculations is developed. This adaptive technique applies a solution-based indicator in order to identify local clouds where new points should be inserted in or existing points could be safely removed from the computational domain. The flow solver and the adaptive procedure have been evaluated and the results are highly encouraging. Several numerical examples are provided throughout the article in order to illustrate their performance.

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. Nowadays, there are two main types of numerical techniques for solving PDEs. On the one hand, there exist mesh-based or conventional discretization methods; among them, Finite Differences (FD), Finite Volumes (FV) and Finite Elements (FE) methods are of singular interest. These techniques are mostly employed in practice due to their robustness, efficiency and high confidence gained through years and years of continuous use and enhancement. On the other hand, there exist meshless methods. Having their pros and cons,

[1] ICREA Research Professor at CIMNE

they offer an alternative to mesh-based techniques. Meshless methods are conceptually attractive; however, their practical implementations are not likely to be so and this is a fact which could explain the comparatively little attention that has been devoted to these techniques. In spite of this, over the last ten years, some difficulties that arose in conventional methods when performing particular applications have brought meshless methods into the focus of attention.

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 typical meshless methods, their implementation issues and practical applications are offered. An interesting work by Fries & Matthies [2] classifies and analyzes the most important meshless methods considering their different origins and viewpoints. The authors highlight the main characteristics and implementation details as well as the advantages and disadvantages of each technique. Some other outstanding reviews on meshless methods can also be found in the literature; see for instance those due to Li & Liu [3], Gu [4] and Duarte [5].

The present work deals with a meshless technique called Finite Point Method (FPM), which was introduced by Oñate *et al.* [6]. In the FPM, the numerical approximation to the problem variables and their 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 and the resulting system of algebraic equations is obtained by means of a collocation technique.

Since the FPM appeared in the literature towards the mid-nineties, it has been successfully applied to solve convective-diffusive problems, incompressible and compressible fluid flow problems [7, 8, 9, 10, 11, 12] and solid mechanics problems [13] among others. As regards fluid flow problems, the first application of the FPM to the solution of the two-dimensional compressible flow equations has been presented by Oñate *et al.* [6, 7] and Fischer [10]. In those works, 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. In addition, the compressible flow equations have been solved using a Taylor-Galerkin scheme. More recently, Sacco [11] 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, such as

a definition of local and normalized approximation bases, a procedure for constructing local clouds of points as well as 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) [14] 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.* [12]. There, two remarkable contributions are well worth mentioning: a reliable procedure for constructing the local clouds (based on a Delaunay technique) and a well-suited upwind biased 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. In this scheme, 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. Other meshless approaches found in the literature share this philosophy, see for instance [15, 16] and the references cited therein.

All the works we have just mentioned, though different, have made remarkable contributions to enhance the performance of the FPM; giving clear evidence of its potential and, in some cases, also revealing important weaknesses. Nowadays, most meshless techniques, and in particular the WLSQ-based methods, are characterized by 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 stability and robustness well-known problems arising from the collocation procedure. Unfortunately, the robustness and the accuracy of the numerical approximation in the cloud of points are absolutely dependent on the previously mentioned features. Moreover, to make matters worse, competitive meshless methods are also in need of a considerable reduction of computational cost, which requires developing more efficient algorithms and data structures. All these considerations become crucial when dealing with real three-dimensional problems of practical application in engineering. Consequently, in our view, improvement in robustness and efficiency seems to be the key to the success of meshless methods.

As regards robustness, some modifications to the FPM have been proposed by Boroomand *et al*. [17] 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 [18] intended to reduce the local approximation dependence on both, the spatial distribution of the cloud of points and the weighting function parameters. 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 in cases where usual approaches fail and avoids modifying the geometrical support where the local approximation is based on.

Regardless of the difficulties meshless methods present in practice (some of them mentioned before), these methods have certain potential advantages over conventional discretization techniques, which explains the scientific interest of many researchers in this area (*cf.* [1, 2, 3]). Most of the advantages are based on the fact that meshless techniques facilitate the treatment of problems involving moving discontinuities, computational domains whose boundaries change with time and *h* and *p*-adaptivity, among others. In our opinion, such topics constitute key opportunities for the development and promotion of meshless methods.

Along the lines of investigation just mentioned, Perazzo *et al.* [19] have recently presented an *h*-adaptive technique for solid mechanics problems which is based on the approximation error obtained at each point by the WLSQ functional. Also, in a previous work [18] we have dealt with high-order FP discretizations in a preliminary manner, exploring FPM capabilities regarding *p*-adaptivity. This time, with the same objective in mind, *i.e.* exploiting the FPM potential, we present an adaptive methodology for two and three-dimensional compressible flow problems.

The rest of the work is organized as follows. In Section 2 the Finite Point approximation is presented. Section 3 is concerned with the domain discretization and the construction of local clouds of points. Next, in Sections 4 and 5, the upwind biased scheme employed for solving the three-dimensional Euler equations is described. Section 6 provides several numerical calculations to show the performance of the flow solver. Then, an *h*-adaptive technique for compressible flow calculations is developed in Section 7 and the performance of this adaptive methodology is evaluated by means of several numerical examples in Section 8. Finally, the conclusions we have reached at are presented in Section 9.

2. NUMERICAL FINITE POINT APPROXIMATIONS ON CLOUDS OF POINTS

In this Section we will present a FP 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 $\boldsymbol{x}_{\mathrm{i}},\ i=1,n$. 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 $\Sigma\Omega_i$ represents a covering for Ω. Each local cloud of points consists of a point $\mathbf{x}_i$ called *star point* and a set of points $\boldsymbol{x}_j$, $j = 2,3,...,np$ surrounding $\boldsymbol{x}_i$, which complete 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 $\Re^d$ (*cf.* [18] for details) and $\boldsymbol{\alpha}$ is an *a-priori* unknown vector. 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 [20]. When the FLS procedure is applied, the approximation methodology is considerably simplified and its computational cost reduced. It should be noticed, though, that FLS approximations lead to multivalued shape functions depending on the cloud in which the approximation is calculated, *i.e.* $\mathbf{N}_{\mathrm{n}}(\boldsymbol{x}_j) \neq \mathbf{N}_{\mathrm{m}}(\boldsymbol{x}_j)$ (subscripts *m* and *n* indicate neighbouring clouds of points). Therefore, the numerical approximation is globally and locally discontinuous and must be considered as valid only at the star point of the cloud where the weighting function is located. Hence, a collocation technique becomes the natural choice in the FPM.

Going back to the minimization procedure, the following discrete functional is defined

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

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

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

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

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

It should be noticed that Eq. (11) must be solved via matrix **A** inversion because the vector $\mathbf{u}^{h}$ is not known in advance. Thus, depending on the spatial distribution of the local cloud of points (especially for the three-dimensional case), matrix **A** can become very ill-conditioned, making it very difficult to invert it with accuracy.

Then, supposing that Eq. (11) is solve accurately enough; and replacing the coefficients α_j in Eq. (1), the approximation to the unknown function at the star point is obtained as follows

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

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 $\boldsymbol{x}_{\mathrm{i}}$ is given by

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

The solution of the equations (8) by direct inversion of matrix **A** is not the most accurate way of solving the LSQ problem. Thus, it must be restricted to cases when the condition number of matrix **A** is moderate. In this work, the procedure adopted to calculate the shape function and its derivatives is the following (*cf.* [18]). 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 (8) are solved by an alternative procedure based on QR factorization. The aim of using a QR factorization technique is to get an acceptable solution in cases where the usual procedure fails without having to modify the geometrical structure of the cloud. The WLSQ problem solution via QR factorization may cost, in terms of CPU-time, up to twice as much as the solution via matrix **A** inversion if $np \gg m$ [21]. However, this extra amount of time is quite unimportant 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. The QR factorization based procedure applied for solving the normal equations system (8) can be summarized as follows.

If matrix **P (**given by Eq. (4)) has rank *m* and $np > m$, it can be uniquely factored as

$$\mathbf{P} = \mathbf{Q}\,\mathbf{R} \tag{15}$$

where matrix $\mathbf{Q} \in \Re^{npx m}$ is orthogonal ($\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$) and matrix $\mathbf{R} \in \Re^{mxm}$ is upper triangular with positive diagonal elements (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 our WLSQ problem, it is necessary to obtain an equivalent unweighted problem. Thus, the next factorization is proposed

$$\tilde{\phi}(\boldsymbol{x}) = \sqrt{\phi(\boldsymbol{x})} \quad \text{such that} \quad \tilde{\boldsymbol{\phi}}^T\tilde{\boldsymbol{\phi}} = \boldsymbol{\phi} \tag{16}$$

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

$$\tilde{\mathbf{P}} = \tilde{\boldsymbol{\phi}}\mathbf{P} \tag{17}$$

After that, it is possible to write an equation system equivalent to the one given by Eq. (8) as

$$\left(\tilde{\mathbf{P}}^T\tilde{\mathbf{P}}\right)\boldsymbol{\alpha} = \left(\tilde{\mathbf{P}}^T\tilde{\boldsymbol{\phi}}\right)\boldsymbol{u}^h \tag{18}$$

Then, the modified matrix (17) is factorized, *i.e.* $\tilde{\mathbf{P}} = \mathbf{QR}$, and replaced in the equivalent unweighted problem (18). This leads to

$$\mathbf{R}\,\boldsymbol{\alpha} = \mathbf{Q}^T\tilde{\boldsymbol{\phi}}\,\boldsymbol{u}^h \tag{19}$$

from which the unknown coefficients α_j can be obtained

$$\boldsymbol{\alpha} = \mathbf{R}^{-1}\left(\mathbf{Q}^T\tilde{\boldsymbol{\phi}}\right)\boldsymbol{u}^h \tag{20}$$

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 quite good quality in cases where they cannot be obtained via 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 approximation methodology.

2.1 The weighting function

In the present work the following normalized Gaussian weighting function is adopted

$$\varphi_i(x_j) = \frac{e^{-\left(d_j/\alpha\right)^k} - e^{-\left(\beta/\alpha\right)^k}}{1 - e^{-\left(\beta/\alpha\right)^k}} \tag{21}$$

where $d_j = \|\boldsymbol{x}_j - \boldsymbol{x}_i\|$, $\alpha = \beta/w$ and $\beta = \gamma\, d_{max}$ ($\gamma > 1.0$). The support of this function is isotropic, circular and spherical in two and three-spatial dimensions respectively. A detailed description of the effects of the free parameters *w, k* and γ on the numerical approximation and some guidelines for their setting have been presented in [18]. However, an important remark about the parameter γ should be done. The parameter γ determines the size of the weighting function's support and, in consequence, an increase of 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 where sudden changes in the distance between neighbouring points happen, *e.g.*, near localized adaptive-refined zones and certain details of three-dimensional geometries. In these cases, which generally lead to highly distorted clouds of points, good results are obtained setting $1 < \gamma < 1.25$.

3. DISCRETIZATION OF THE DOMAIN AND LOCAL CLOUD CONSTRUCTION

An adequate support of points is essential for setting a good local approximation for each cloud. Even though the iterative QR-based technique briefly described above attempts to reduce this dependence, the approximation's spatial support continues playing a major role. At present, there is not a unique criterion to determine the size, shape and structure of the local spatial support and, in consequence, several procedures have been proposed by meshless practitioners. Concerning the FPM, an appropriate methodology for constructing local clouds of points (based on a Delaunay technique) has been suggested by Löhner *et al.* [12]. In the present work we follow the general criteria proposed there.

3.1 Domain discretization

The point discretization of the analysis domain Ω is obtained by means of a modification of the algorithm presented in [22]. It starts from a Delaunay triangulation that bounds the domain and inserts new points in the centre of empty spheres filling Ω. This incremental quality technique, known as *optimization driven point insertion*, allows achieving a fast point discretization of the analysis domain well-suited for Finite Point calculations.

3.2 Local cloud 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 with 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 1(a)), every point $\mathbf{x}_j$ located within the search radius is admissible if it meets the following conditions

$$\cos(\theta) \geq \cos\left(\pi/2 + \delta\right) \quad ; \quad \cos(\theta) = \frac{\boldsymbol{n}_i \cdot \boldsymbol{r}_j}{\|\boldsymbol{n}_i\| \|\boldsymbol{r}_j\|} \tag{22}$$

$$\left\|\boldsymbol{r}_j^t\right\| < \alpha\, r_{search} \tag{23}$$

Condition (22) defines an acceptation zone 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 (23) 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 1(b)). Then, every point within the search radius is admissible if

$$\cos(\theta) \geq \cos\left(\pi/2 + \delta\right) \quad ; \quad \cos(\theta) = \frac{\boldsymbol{n}_{j_{nea}} \cdot \boldsymbol{r}_j}{\|\boldsymbol{n}_{j_{nea}}\| \|\boldsymbol{r}_j\|} \tag{24}$$

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

FIGURE 1

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 np_{max} points nearest to $\mathbf{x}_i$ 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 by Löhner *et al.* [12], 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 7.

4. 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

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}^k}{\partial x_k} = \mathbf{0} \tag{25}$$

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

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u_i \\ \rho e_t \end{bmatrix} \quad , \quad \mathbf{F}^k = \begin{bmatrix} \rho u_k \\ \rho u_i u_k + \delta_{ik} p \\ (\rho e_t + p) u_k \end{bmatrix} \tag{26}$$

where ρ, p and e_t respectively denote 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 subscripts $i,k = 1,d$. The following state relation for a perfect gas closes the system of equations (25)

$$p = \rho(\gamma - 1)\left[e_t - \tfrac{1}{2} u_i u_i \right] \tag{27}$$

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. (25) in a closed domain $\Omega \in \Re^d$ with boundaries $\Gamma = \Gamma_\infty \cup \Gamma_w$ requires appropriate initial and boundary conditions. The initial conditions only start the explicit calculation and they are simple to implement. In general, they could be taken from the far-field state $\mathbf{U}_\infty$. Regarding the boundary conditions, those employed in the present work are of two different kinds. 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. In the case of far-field boundary conditions, the prescribed fluxes at each boundary point are obtained solving an approximate Riemann problem in the outward normal direction to the boundary, between the boundary point state $\mathbf{U}_i$ and the far-field state $\mathbf{U}_\infty$. Over solid boundaries, slip wall conditions are applied forcing the fluxes to remain tangent to the boundaries, *i.e.*, cancelling their components in the boundary normal direction.

5. THE FLOW SOLVER

In this section, the numerical strategy adopted for solving the compressible flow equations is set forth. Despite some modifications to the way in which the divergence of the advective fluxes is discretized in the local cloud of points, the overall scheme follows the general lines proposed by Löhner *et al.* [12].

Recalling the FPM approximation procedure described in Section 2, for each star point $\mathbf{x}_i \in \Omega$ we can state the following numerical approximations

$$\begin{aligned} \hat{\mathbf{U}}\left(\boldsymbol{x}_i\right) &= \hat{\mathbf{U}}_i = \sum_{j\in\Omega_i} N_{ij}\, \mathbf{U}_j^h \\ \hat{\mathbf{F}}^k\left(\boldsymbol{x}_i\right) &= \hat{\mathbf{F}}_i^k = \sum_{j\in\Omega_i} N_{ij} \left(\mathbf{F}_j^k\right)^h \end{aligned} \tag{28}$$

where $N_{ij} = N_i(\boldsymbol{x}_j)$ 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 semi-discrete counterpart of Eq. (25) could be expressed for each star point $\mathbf{x}_i$ by

$$\frac{\partial \hat{\mathbf{U}}_i}{\partial t} = -\frac{\partial \hat{\mathbf{F}}_i}{\partial x} = -\sum_{j\in\Omega_i} \frac{\partial \boldsymbol{N}_{ij}}{\partial x} \mathbf{F}_j^h = -\sum_{j\in\Omega_i} b_{ij}\, \mathbf{F}_j^h \tag{29}$$

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

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 point data. These values are related by Eq.(28), 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 partition of nullities (PNs) property of the shape function derivatives it is possible to infer

$$\sum_{j\in\Omega_i} b_{ij} = b_{ii} + \sum_{j\neq i} b_{ij} = 0 \quad \rightarrow \quad b_{ii} = -\sum_{j\neq i} b_{ij} \tag{30}$$

and replacing Eq. (30) in Eq. (29), the following semi-discrete expression is obtained

$$\frac{\partial \mathbf{U}_i}{\partial t} = -\sum_{j\neq i} b_{ij}\left(\mathbf{F}_j - \mathbf{F}_i\right) \tag{31}$$

Eq. (31) 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 [16] used for its calculation. In this way, we obtain a totally equivalent semi-discrete expression which is given by

$$\frac{\partial \mathbf{U}_i}{\partial t} = -2\sum_{j\neq i} b_{ij}\left(\mathbf{F}_{ij} - \mathbf{F}_i\right) \tag{32}$$

where $\mathbf{F}_{ij}$ is an *a priori* unknown numerical flux vector, evaluated at the midpoint of the line segment connecting the star point $\mathbf{x}_i$ with another point $\mathbf{x}_j \in \Omega_i$. Many possibilities for calculating $\mathbf{F}_{ij}$ can be found in the literature. Following the ideas presented in [12], the Roe's approximate Riemann solver [23] is adopted in this work. Then, the numerical flux results

$$\mathbf{F}_{ij} = \tfrac{1}{2}\left(\mathbf{F}_j + \mathbf{F}_i\right) - \tfrac{1}{2}\left|\mathbf{A}(\mathbf{U}_i, \mathbf{U}_j)\right|\left(\mathbf{U}_j - \mathbf{U}_i\right) \tag{33}$$

where $\mathbf{A}(\mathbf{U}_i,\mathbf{U}_j)$ is the flux Jacobian matrix evaluated at the Roe average-state between the points $\mathbf{x}_i$ and $\mathbf{x}_j$, *i.e.*, $\mathbf{U}_L=\mathbf{U}_i$ and $\mathbf{U}_R=\mathbf{U}_j$. In order to calculate the absolute value of the Roe matrix the procedure suggested by Turkel [24] is applied. This procedure avoids costly matrix-matrix and matrix-vector multiplications in the calculation of the dissipative term $|\mathbf{A}(\mathbf{U}_i,\mathbf{U}_j)|(\mathbf{U}_j-\mathbf{U}_i)$.

FIGURE 2

The multi-dimensional extension of the scheme presented above is straightforward. For each pair of points ($\mathbf{x}_i$,$\mathbf{x}_j$), 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 (32) results

$$\frac{\partial \mathbf{U}_i}{\partial t} = -2\sum_{j\neq i} b_{ij}^k \left[\mathbf{F}_{ij}^k - \mathbf{F}_i^k \right] \tag{34}$$

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

$$\mathbf{F}_{ij}^k = \tfrac{1}{2}\left(\mathbf{F}_j^k + \mathbf{F}_i^k\right) - \tfrac{1}{2}\left|\mathbf{A}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)\right|\left(\mathbf{U}_j - \mathbf{U}_i\right)\cdot \hat{\boldsymbol{n}}^k \tag{35}$$

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 (34) is presented in Figure 3.

FIGURE 3

5.1 Increasing spatial accuracy

The low-order scheme we have developed is useless in practice. In order to make this scheme suitable for capturing all the flow features 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 [25] allows achieving accurate second and third-order schemes using linear and quadratic reconstruction of the variables respectively. Unfortunately, this high-order methodology does not guarantee an oscillation-free solution and monotonicity should be enforced by introducing non-linear *limiters* into the reconstruction process. In brief, these limiters recognize any local extrema of the solution field and automatically switch, at these points, the high-order extrapolation to a zero-order extrapolation, avoiding the appearance of under and overshoots in the numerical solution.

Taking into consideration the high-order approach proposed in [12], in this work we adopt a MUSCL reconstruction of the variables in conjunction with the Van Albada limiter. This results in the following set of reconstructed variables

$$\mathbf{U}_i^+ = \mathbf{U}_i + \frac{\mathbf{s}_i}{4}\Big[(1-\eta)(\mathbf{U}_i - \mathbf{U}_{i-1}) + (1+\eta)(\mathbf{U}_j - \mathbf{U}_i)\Big]$$
$$\mathbf{U}_j^- = \mathbf{U}_j - \frac{\mathbf{s}_j}{4}\Big[(1-\eta)(\mathbf{U}_{j+1} - \mathbf{U}_j) + (1+\eta)(\mathbf{U}_j - \mathbf{U}_i)\Big] \tag{36}$$

where $\mathbf{U}_i^+$ and $\mathbf{U}_j^-$ are, respectively, the leftward and rightward extrapolations to the conservative variables vector at point $\mathbf{x}_{ij}$. In the above expressions the choice of the parameter η=-1 leads to a second-order, leftward-biased scheme for $\mathbf{U}_i$ and a rightward-biased scheme for $\mathbf{U}_j$. For η=1 and η=1/3, a second-order centered scheme and a third-order scheme are obtained respectively. The Van Albada limiters $\mathbf{s}_i$ and $\mathbf{s}_j$ [12] are given by

$$\mathbf{s}_i = \max\left[0, \frac{2(\mathbf{U}_i - \mathbf{U}_{i-1})(\mathbf{U}_j - \mathbf{U}_i) + \varepsilon}{(\mathbf{U}_i - \mathbf{U}_{i-1})^2 + (\mathbf{U}_j - \mathbf{U}_i)^2 + \varepsilon}\right]$$
$$\mathbf{s}_j = \max\left[0, \frac{2(\mathbf{U}_{j+1} - \mathbf{U}_j)(\mathbf{U}_j - \mathbf{U}_i) + \varepsilon}{(\mathbf{U}_{j+1} - \mathbf{U}_j)^2 + (\mathbf{U}_j - \mathbf{U}_i)^2 + \varepsilon}\right] \tag{37}$$

where $\varepsilon \approx$ 1.0E-5 is a small constant included to avoid divisions by zero. 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 i and j

$$\mathbf{U}_i - \mathbf{U}_{i-1} = 2\boldsymbol{l}_{ji}\cdot\nabla\mathbf{U}_i - (\mathbf{U}_j - \mathbf{U}_i)$$
$$\mathbf{U}_{j+1} - \mathbf{U}_j = 2\boldsymbol{l}_{ji}\cdot\nabla\mathbf{U}_j - (\mathbf{U}_j - \mathbf{U}_i) \tag{38}$$

in which $\boldsymbol{l}_{ji} = \boldsymbol{x}_j - \boldsymbol{x}_i$ is the vector linking the points i and j (see Figure 4).

FIGURE 4

Once the high-order extrapolations (36) have been calculated, the midpoint numerical flux (35) is modified according to

$$\mathbf{F}_{ij}^k = \tfrac{1}{2}\left(\mathbf{F}^k(\mathbf{U}_i^+) + \mathbf{F}^k(\mathbf{U}_j^-)\right) - \tfrac{1}{2}\left|\mathbf{A}_{\hat{n}}(\mathbf{U}_i^+, \mathbf{U}_j^-)\right|(\mathbf{U}_j^- - \mathbf{U}_i^+)\cdot\hat{\boldsymbol{n}}^k \tag{39}$$

and then, replacing Eq. (39) in Eq. (34) the high-order semi-discrete scheme is obtained.

5.2 Time discretization

According to [12], the temporal discretization of Eq. (34) 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 $\mathbf{U}^h$ is known at time $t = t^n$, the right hand side of Eq. (34) 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

$$\begin{aligned} \mathbf{U}_i^{(0)} &= \mathbf{U}_i^n \\ &\vdots \\ \mathbf{U}_i^{(s)} &= \mathbf{U}_i^n + \alpha_s \Delta t_i \mathbf{RHS}_i^{(s-1)} \\ &\vdots \\ \mathbf{U}_i^{n+1} &= \mathbf{U}_i^{(s_{\max})} \end{aligned} \tag{40}$$

where Δt_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 $\alpha_4 = 1.0$

The difference between the (^{h}) parameters and the approximated ones (^) has already been pointed out in Section 5. Taking into account that $RHS_i = f(\mathbf{U}_j^h)\ \nabla \mathbf{x}_j \in \Omega_i$, the following linear system has to be solved at the end of each integration stage

$$\mathbf{M}\,\mathbf{U}^h = \hat{\mathbf{U}} \tag{41}$$

where $\mathbf{M} \in \Re^{n \times n}$ is the *mass matrix* of the system, which results from the assembly of the N_{ij} coefficients (see Eq.(28)). Fortunately, as it was said before, this system has excellent properties and can be solved by a few iterations of a Gauss-Seidel method or similar.

It should be noticed that, even though the numerical scheme presented in this section is intended to solve the inviscid compressible flow equations, with minor modifications the same scheme can be applied for solving the viscous flow equations.

6. NUMERICAL EXAMPLES

In this section, some three-dimensional compressible flow calculations are presented with the aim of illustrating the performance of the proposed methodology. The first example concerns a subsonic flow past a sphere. Although this example has barely any 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 to practical aerodynamics problems. With the same

objective in mind, by the end of this section another transonic flow calculation concerning a NACA wing-body configuration is presented.

6.1 Subsonic flow around a sphere

In this example, subsonic inviscid flow past a sphere is solved for a freestream Mach number $M_\infty = 0.2$. The computational domain is discretized by a non-structured distribution of 30,013 points and second-order spatial approximations are obtained in clouds of points with $30 \leq np \leq 40$. Next, Cp and Mach number isolines on the sphere are shown in Figure 5.

FIGURE 5

The calculated Cp distribution around the sphere (in the streamwise direction), is compared with analytical potential flow results in Figure 6.

FIGURE 6

A reasonable agreement between the numerical and potential results can be observed. Note that the separation point on the sphere, obtained by the FP calculation, is almost coincident with the potential rear stagnation point. This fact gives a cue of the low inherent dissipation of the proposed numerical scheme.

FIGURE 7

6.2 Transonic flow over the ONERA M6 wing

This validation test [26] was developed by the ONERA Aerodynamics Department in 1972 with the objective of providing experimental support for studies regarding transonic flows at high Reynolds numbers. Since then, these experimental results, which cover a wide range of subsonic and transonic flows, have turned into a classical reference data for code validation assessments. The ONERA M6 is a semi-span wing with a sweepback $\Lambda_{LE} = 30^o$, an aspect ratio A = 3.8 and a taper ratio $\lambda = 0.562$. The wing-section is an ONERA 'D' symmetrical airfoil constant along the span and the wing has not geometrical twist. In this example we solve the test case # 2308 (*cf.* [26]) which concerns transonic flow over the ONERA M6 wing set at an incidence angle $\alpha = 3.06^o$. 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 also be found in [27].

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 512,141 points and second-order approximation bases are employed for calculating the shape functions and their derivatives in clouds with $30 \leq np \leq 45$. Next, Cp and Mach number numerical results are shown in Figure 8 and Figure 9 respectively.

FIGURE 8

FIGURE 9

A comparison between numerical and experimental Cp distributions along several sections on the wing is shown in Figure 10. In accordance with the available experimental data [26], these sections are located at the following spanwise stations: η = 0.2, 0.44, 0.65, 0.8, 0.9, 0.95 and 0.99 being $\eta = 2y/b$.

FIGURE 10

A good agreement between computed and experimental results can be observed in Figure 10 and, 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. Notice that the experimental data measured at η=0.99 reveals separated flow behind the shock wave on the upper side of the wing. Consequently, experimental and calculated Cp distributions do not match in the separated flow region.

6.3 Transonic flow over a NACA wing-body configuration

This example involves the computation of an inviscid transonic flow over a wing-body configuration [28]. The wing has a sweepback $\Lambda_{1/4}$ = 45º, an aspect ratio A = 4, a taper ratio λ = 0.6 and it has not geometrical twist; moreover, the wing-section is a NACA 65A006 airfoil constant along the wing span. The fuselage has a circular cross-section and its rear part is attached to a sting which supports the model in the wind tunnel test section.

The numerical calculation presented here regards a freestream Mach number $M_\infty = 0.9$ and the model incidence angle is $\alpha = 4^o$. The discretization of the computational domain consists of an unstructured distribution of 512,553 points and second-order approximations are built on clouds with $35 \leq np \leq 45$. Next, Cp and Mach number results computed for the proposed flow conditions are presented in Figure 11 and Figure 12 respectively.

FIGURE 11

FIGURE 12

Then, Cp distributions calculated at two spanwise stations $\eta = 0.4$ and $\eta = 0.8$ on the wing are compared with experimental measurements [28] in Figure 13.

FIGURE 13

Additionally, the longitudinal Cp distribution along the fuselage symmetry plane is contrasted with experimental results in Figure 14.

FIGURE 14

As in the previous case, minor differences (due to the inviscid assumption adopted for the computational flow model) exist between numerical and experimental results. In spite of this, both results match very well and this can be observed in Figures 13 and 14.

7. 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 analysis as regards man-hours, CPU-time and memory requirements significantly. Also, adaptive procedures make the accurate computation of the smaller scales of the flow field easier, especially when we do not have *a priori* information concerning the solution, and become essential for non-stationary problems involving moving discontinuities.
In the introduction to this work we have referred to some topics in numerical computation where meshless approaches seem to have certain advantages over mesh-based approaches and

adaptivity is one of them. The fact that meshless techniques do not need to keep a conforming mesh makes them specially well-suited for implementing adaptive procedures. With the purpose of exploiting this capability, in this Section we develop an adaptive Finite Point procedure for compressible flow problems. The adaptive technique we propose is described in the next.

7.1 The refinement criterion

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 a normalized indicator that evaluates, in an approximate manner, the curvature of the solution at each point

$$\varphi_i = \frac{1}{\varphi_m}\sum_{j=1}^{nn}\left|\boldsymbol{l}_{ji}\cdot\left(\nabla\rho_j - \nabla\rho_i\right)\right| \quad , \quad \varphi_m = \max\left(\varphi_i\right) \quad i=1,n \tag{42}$$

In the expression above *nn* is the number of points in the first layer of nearest neighbours of $\boldsymbol{x}_i$ (already obtained in the local cloud construction stage), $\boldsymbol{l}_{ji} = \boldsymbol{x}_j - \boldsymbol{x}_i$ is the vector linking each pair of points ($\boldsymbol{x}_i$,$\boldsymbol{x}_j$) and ρ is the density of the fluid. Naturally, another flow variable or a combination of flow variables can be adopted for calculating the refinement indicator (42). The last option could be appropriate for the treatment of viscous fluid flows.

The refinement criterion is applied as follows. Based on Eq. (42); new points are inserted around $\boldsymbol{x}_i$ when $\varphi_i > \varphi_{max}$ and, 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 examples presented here $\varphi_{max} \approx 0.1$ and $\varphi_{min} \approx 0.005$ are chosen. It should be notice that 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 taking another local maximum for normalizing the indicator.

7.2 The strategy

Once the refinement criterion has been applied, the remaining of the proposed adaptive procedure 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 *affected* existing

points 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.

7.2.1 Insertion of new points

When a star 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, *i.e.* at the centre of the empty circumcircle/circumsphere calculated for each triangle/tetrahedron (2-D/3-D) composing the Delaunay grid of nearest neighbours. Each candidate point $\boldsymbol{x}_c$ is accepted if it meets the following requirements:

r1. The radius of the empty 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 points.

r2. The radius r_c is smaller than a certain internal measure (d_e) of the triangle/tetrahedron which originates the empty circumcircle/circumsphere. The internal measure d_e is calculated as $d_e = \max(|\boldsymbol{e}_j \cdot \hat{\boldsymbol{i}}|, |\boldsymbol{e}_j \cdot \hat{\boldsymbol{j}}|, |\boldsymbol{e}_j \cdot \hat{\boldsymbol{k}}|)$ where the subscript j stands for each edge of the triangle/tetrahedron and $(\hat{\boldsymbol{i}}, \hat{\boldsymbol{j}}, \hat{\boldsymbol{k}})$ are unitary vectors in each spatial direction.

r3. The distance from the candidate point $\boldsymbol{x}_c$ to another new point previously accepted is greater than the minimum admissible distance between points r_{min}.

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 of 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 (*cf.* [29]). Figure 15 sketches the refinement procedure for a bi-dimensional cloud of points.

7.2.2 Removal of existing points

Point removal capabilities are indispensable for the treatment of non-stationary problems. In this work, the removal of points is restricted only to existing points that have been inserted in prior refinement levels. In other words, the initial set of points (original 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 15

7.2.3 Update

Once the insertion and removal of points is finished, a few steps of a Laplacian smoothing are carried out on the affected area. This is particularly helpful when points have been removed in large quantities. After that, 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 previously existing nearest neighbours.

8. SOME EXAMPLES OF ADAPTIVE FINITE POINT CALCULATIONS

In this Section several numerical examples are presented in order to illustrate the performance of the proposed adaptive procedure. We begin with two computation cases intended to verify the adaptive numerical solution. The first example concerns a bi-dimensional adaptive calculation of a supersonic flow around a double-wedge airfoil and the second one deals with the solution of a shock-tube problem in a bi-dimensional domain. A third example is related to the solution of a transonic flow over a NACA 0012 airfoil and the fourth and last example involves a three-dimensional flow calculation over the ONERA M6 wing. The two final calculation cases give an idea about the possibilities of application of the present adaptive meshless technique to practical engineering problems.

8.1 Supersonic flow past a double wedge airfoil

This example resolves the flow around a double wedge airfoil immersed in a supersonic flow. The airfoil has a unitary chord c = 1 and the wedge angle is $\beta = 20^{o}$; the upstream Mach number is $M_{\infty} = 2$ and the airfoil is set at an incidence angle $\alpha = 0^{o}$. The initial coarse discretization is composed by an unstructured distribution of 1,279 points and second-order spatial approximations are built in clouds where $15 \leq np \leq 20$. The final adapted discretization, achieved after 70 refinement levels, consists of 51,907 points. Next, the initial and the final adapted discretization are shown.

FIGURE 16

Figure 17 presents a comparison between the analytical solution of the problem, calculated along an x-cut in the domain located 0.1c above the airfoil chord-line, and the numerical solution computed at successive refinement levels. It is possible to note in Figure 17 how the numerical solution of successive refined-discretizations converges into the analytical solution of the problem. Finally, the time convergence of the problem is shown in Figure 18 where the complete process of the adaptive numerical computation can be seen.

FIGURE 17

FIGURE 18

When the simulation starts, some 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, the latter is recovered after a few time steps. For a value of the density temporal residual of 1.0E-5, the first refinement level is performed. Then, consecutive refinement levels are carried out every 200 time steps. Note that the peaks of the convergence curve correspond to each refinement level performed during the computation.

8.2 The shock tube problem

The shock tube problem is a one-dimensional non-stationary Riemann problem proposed by Sod in 1978. In this example we adopt a unitary-length bi-dimensional domain and carry out an adaptive shock tube simulation defined by the following initial conditions

$$\mathbf{U}\left(\mathbf{x}, t_0\right) = \begin{cases} \mathbf{U}_L = \left(1, 0, 0, 2.5\right)^T & x \le 0.5 \\ \mathbf{U}_R = \left(0.125, 0, 0, 0.25\right)^T & x > 0.5 \end{cases} \tag{43}$$

which give a pressure ratio across the diaphragm p_L/p_R = 10 (notice that the diaphragm position is x=0.5). According to the given initial conditions, the intensity of the shock is moderate and the flow regime after the expansion is subsonic.

The computational domain is initially discretized by a coarse homogeneous distribution of 217 points and second-order spatial approximations are calculated in clouds with

$12 \leq np \leq 20$. After the rupture of the diaphragm, successive refinement levels are performed at regular periods. The simulation time in this example is t = 0.2 seconds, for which the adapted discretization reaches a total of 1,761 points. Next, Figure 19 presents some snapshots of adapted discretizations taken at different times from the rupture of the diaphragm. There, the coloured points show flow density numerical results.

FIGURE 19

Figure 20 displays several comparisons between the numerical and the analytical solution for the density variable along the tube, corresponding to the simulation times pointed out in Figure 19.

FIGURE 20

In the Figure above, a considerable smoothing of the numerical solution can be observed in the first refinement levels (t = 0.045 and 0.1 secs.) for which the discontinuities are noticeable smeared. This fact can be explained to a great extend by the coarse discretization employed in order to start the simulation. It is necessary to notice that the number of points to be added in a given refinement level depends upon the flow field variables but also on the old point discretization (*cf.* Section 7.2.1). Consequently, certain geometrical restrictions limit the maximum number of new points inserted in a given refinement level, making the discretization unable to adapt instantaneously to the flow variables in a proper manner. Nevertheless, a closer agreement between the numerical and the analytical solution is obtained for the simulation times t = 0.14, 0.19 and 0.20 secs. In these cases, an improved flow resolution but also minor inaccuracies in the discontinuities location can be observed. We suspect that this behaviour could be related, on the one hand, to the straightforward procedure proposed to interpolate the numerical solution between the old and the new refined discretization. On the other hand, to the Laplacian smoothing performed after each addition and/or removal of points; although the solution should not be sensitive to the discretization smoothing if a proper interpolation procedure is employed.

A numerical calculation performed with a fixed homogeneous discretization, having a point density similar to that in the final adapted discretization of Figure 19, is presented at the bottom left corner of Figure 20. Comparing the latter result with its counterpart obtained for the adaptive simulation, it is possible to observe that the numerical dissipation introduced by

the refinement procedure is quite small. It should be noticed that the problem setting employed in both calculations is the same. Finally, it can be observed that the normalization adopted for calculating the refinement indicator may cause some detriment to the contact discontinuity resolution due to the fact that stronger gradients are present at the shock location. In cases like this, it would be useful to adopt a local criterion for calculating the refinement indicator.

8.3 Transonic flow around a NACA 0012 airfoil

This example concerns the computation of a transonic inviscid flow past a NACA 0012 airfoil. In this calculation, the freestream Mach number is $M_\infty = 0.8$ and the incidence angle is $\alpha = 1.25^o$. The initial spatial discretization is composed of an unstructured distribution of 976 points and second-order spatial approximations are calculated in clouds with $15 \leq np \leq 20$. The finest adapted discretization consists of 4,938 points and is achieved after 15 refinement levels. Both, initial and final discretizations are shown in Figures 21 and 22.

FIGURE 21

FIGURE 22

Notice that the adaptive procedure captures all the flow features with precision. The strong shock wave on the upper side of the airfoil, the weaker shock on its lower side and the leading and trailing edge regions are appropriately refined. The next figure shows the Cp field around the airfoil calculated for the final adapted discretization.

FIGURE 23

The computed Cp distribution on the airfoil is compared to numerical reference results [30] in Figure 24, where good agreement can be observed. Finally, the time convergence history of the problem is presented in Figure 25.

FIGURE 24

FIGURE 25

8.4 A three-dimensional example: the ONERA M6 wing

This example solves the flow around the ONERA M6 wing adopting the freestream conditions given in Section 6.2. The initial coarse discretization consists of an unstructured distribution of 66,864 points and second-order approximation bases are employed in clouds with $30 < np < 45$. In this simulation the adapted discretization reaches a total of 102,592 points after 35 refinement levels. Next, Figure 26 shows the original and final discretizations of the wing; coloured points display Cp results.

FIGURE 26

The initial discretization of the wing consists of 14,221 points and 28,314 triangle elements whereas the final adapted discretization is composed of 15,537 points and 30,942 triangle elements. Notice that new points are mainly concentrated around the strong shock wave spanning the wing where large gradients are detected. In order to make the refinement indicator (42) also sensitive to the smaller gradients in the flow field, it is possible to decrease the parameter φ_{max} or change the normalization criterion. However, as the indicator becomes more sensitive, the refinement procedure loses its local character. This would lead to an insertion of large quantities of new points in each refinement level and, in particular cases, the convergence of the problem could be seriously affected. Thus, the adoption of local maximums for normalizing the indicator seems to be a more adequate choice.

Next the Cp distributions along two sections of the wing, calculated with the original and the finest discretization, are compared in Figure 27. In the same figure, a view of the finest adapted point discretization for a cut in the plane x-z of the domain (passing through the same spanwise stations) is presented. Finally, the convergence history of the problem is shown in Figure 28.

FIGURE 27

FIGURE 28

Regarding the computational cost of the proposed adaptive technique, numerical experiments show that the CPU-time required by each refinement level is approximately equal to the time involved in the update stage (see Section 7.2.3) and the cost of inserting and removing points

is almost negligible. In general, the overall CPU-time involved in each refinement level is only a fraction of the time required for advancing the problem solution a single time-step.

9. CONCLUSIONS

An adaptive Finite Point Method for compressible flow calculations has been presented. On the basis of a robust WLSQ procedure and an iteratively-improved local approximation, an upwind semi-discrete scheme is constructed in each cloud of points. This methodology, in conjunction with a multi-stage time integration scheme, allows solving real three-dimensional problems minimizing the dependence of the numerical results on the spatial discretization of the analysis domain, the local cloud topology and the parameters of the local approximation. All these are important achievements which make possible further enhancement and expansion of the FPM capabilities.

In the introduction to this article we made reference to certain topics in numerical simulation which offer good opportunities for the development and promotion of meshless techniques. With the aim of exploiting these opportunities, an adaptive technique for compressible flow calculations has been developed. Several test cases involving stationary and non-stationary flow problems have been presented with the purpose of exemplifying the performance of the proposed technique. All the examples demonstrate that such adaptive technique is capable of properly resolve the essential flow features, achieving a robust and reliable adaptivity with a very low computational cost. Although some numerical tests (of which a few have been reported here) highlight the need for more accurate refinement criteria and an improved treatment of moving discontinuities, the overall performance of the proposed adaptive technique is highly satisfactory and this can be seen as the main achievement of this work.

Real viscous flow involves certain features where meshless techniques, and especially adaptive meshless techniques, could make important contributions, *e.g.* boundary layer discretization and shock-boundary layer interaction problems. In this sense, we have already developed the basic tools for tackling these kinds of problems and solving them constitutes the next short-term goal we expect to achieve.

Regarding computational efficiency, we must say that at present we still lack precise performance comparisons between our Finite Point methodology and conventional discretizations techniques. However, we estimate that the computational cost of a three-dimensional FP computation would exceed a similar FE-based computation, in the best of the cases, by a cost factor of 3 being 5 a typical value. As it can be seen, if a competitive FPM is

sought, an improvement in computational efficiency is essential. In that respect, numerous techniques can be implemented in order to accelerate convergence to the steady state. Combining these techniques with a suitable data-structure and an optimized way to perform the numerical calculations, it is possible to enhance the efficiency of the present Finite Point methodology considerably. Moreover, performance comparisons between the present FPM technique and other meshless techniques accomplishing similar tasks are essential for locating the FPM into the meshless methods actual scenario. In order to conclude, we could say that in general the results obtained are much encouraging though efficiency is still a pending matter, consequently, future efforts should also be aimed at improving this key point.

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. We express our gratitude to Nestor Calvo for helping with the point generation technique employed in this work. Last but not least, 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] Gu Y. T., Meshfree methods and their comparisons. *International Journal of Computational Methods* 2005, **4**:477-515.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

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

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

[11] 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.

[12] 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.

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

[14] 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.

[15] Sridar D., Balakrishnan N. An upwind finite difference scheme for meshless solvers. *Journal of Computational Physics* 2003; **189**:1-29.

[16] 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.

[17] 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.

[18] Ortega E., Oñate E., Idelsohn S. An improved finite point method for three-dimensional potential flows. *Computational Mechanics* 2007; **40**:949-963.

[19] Perazzo F., Löhner R., Perez-Pozo L. Adaptive methodology for meshless finite point method. *Advances in Engineering Software* 2007; **39**:156-166.

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

[21] Demmel J. W. Applied numerical linear algebra. Society for Industrial and Applied

Mathematics, 1997.

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

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

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

[26] 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

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

[28] 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.

[29] Zorin D., Schröder, P., Sweldens, W. Interpolating Subdivision for Meshes with Arbitrary Topology. In Proceedings of SIG-GRAPH, 1996; **96**:189–192.

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

FIGURES

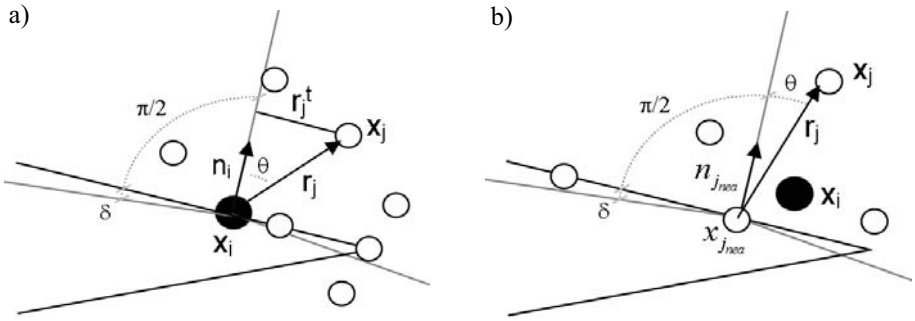

Figure 1: The construction of local clouds near the boundaries: a) The star point located over a solid boundary; b) A cloud of points intercepting a solid boundary.

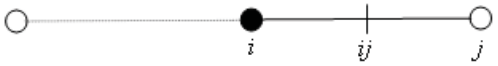

Figure 2: The one-dimensional stencil of points.

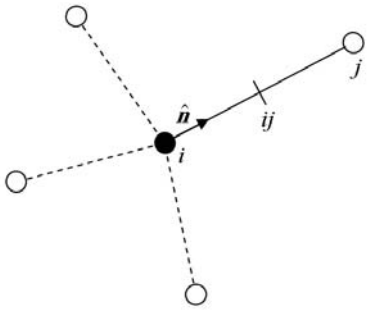

Figure 3: The multi-dimensional stencil of points.

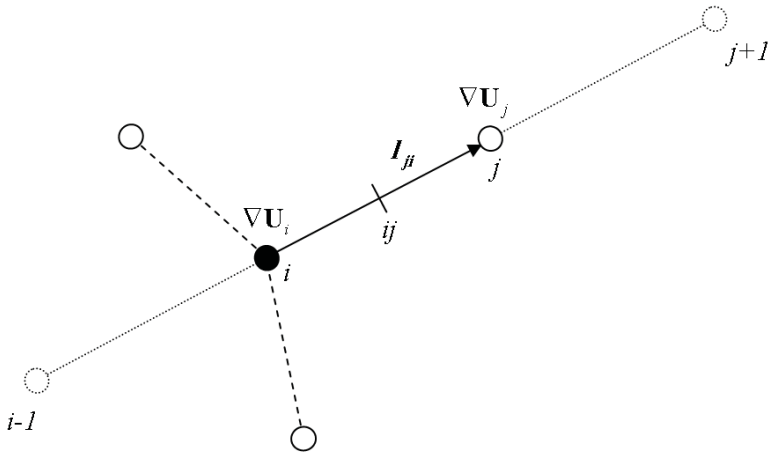

Figure 4: Implementation of the multi-dimensional reconstruction of the variables.

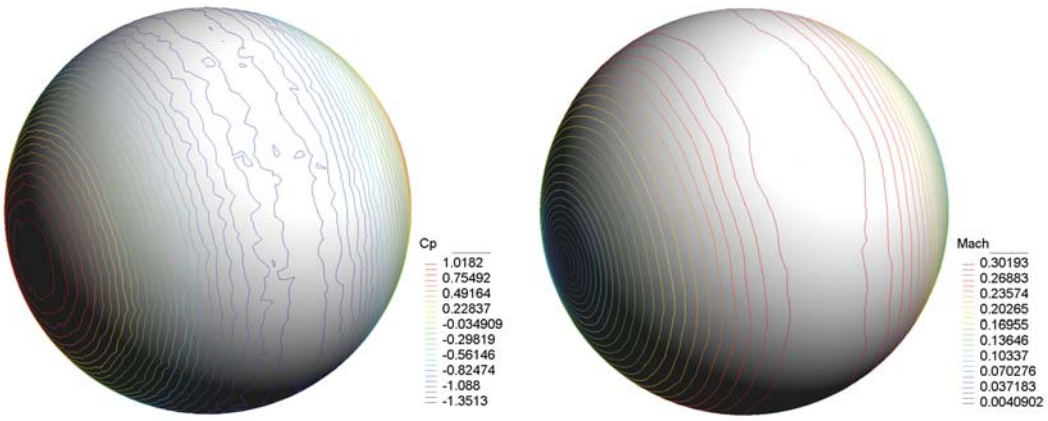

Figure 5: Cp and Mach number isolines on the sphere, $M_\infty = 0.2$.

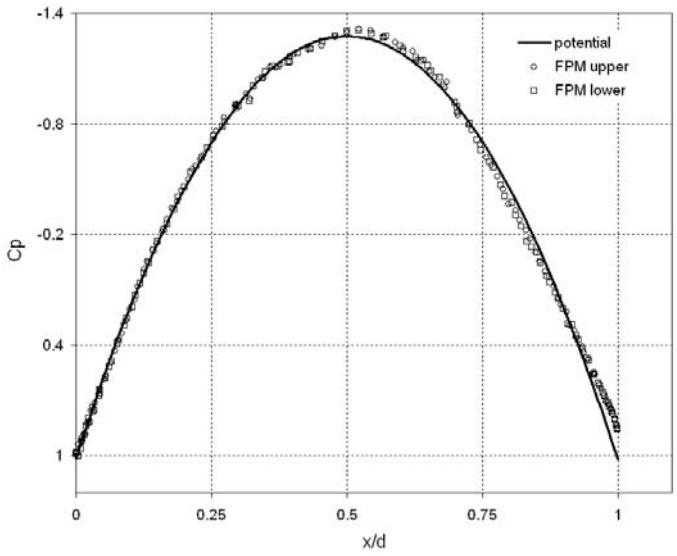

Figure 6: Cp distribution around the sphere; a comparison between the FP calculation and the analytical potential solution. $M_\infty = 0.2$.

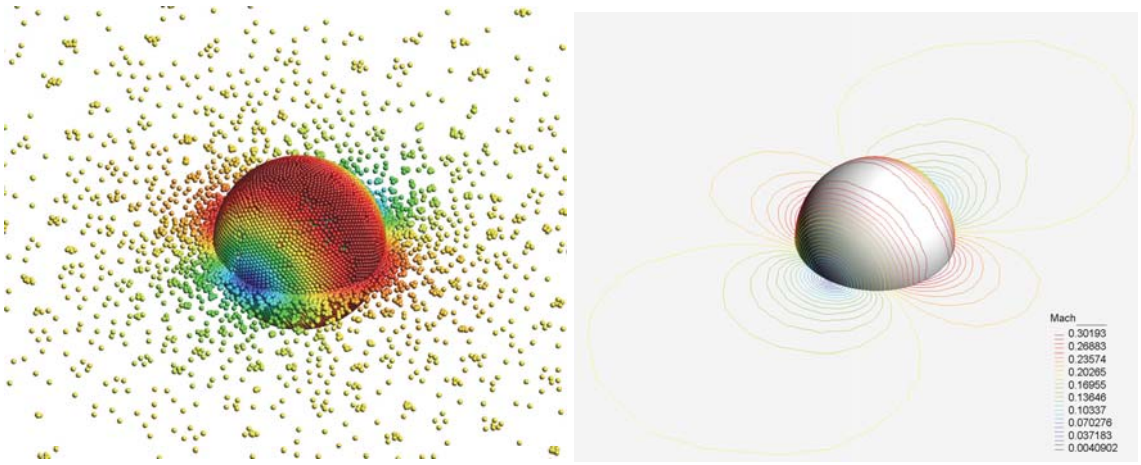

Figure 7: The sphere and the symmetry plane of the problem. Left: points displaying Mach number results; Right: Mach number isolines. $M_\infty = 0.2$.

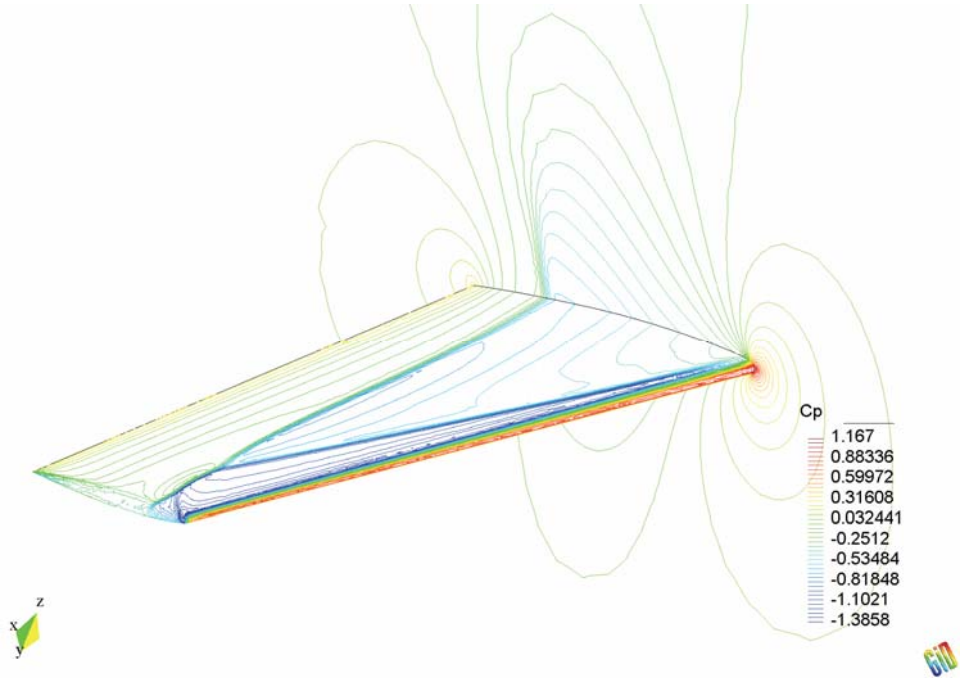

Figure 8: Cp isolines on the upper surface of the ONERA M6 wing and the symmetry plane. $M_\infty = 0.84$ and $\alpha = 3.06º$.

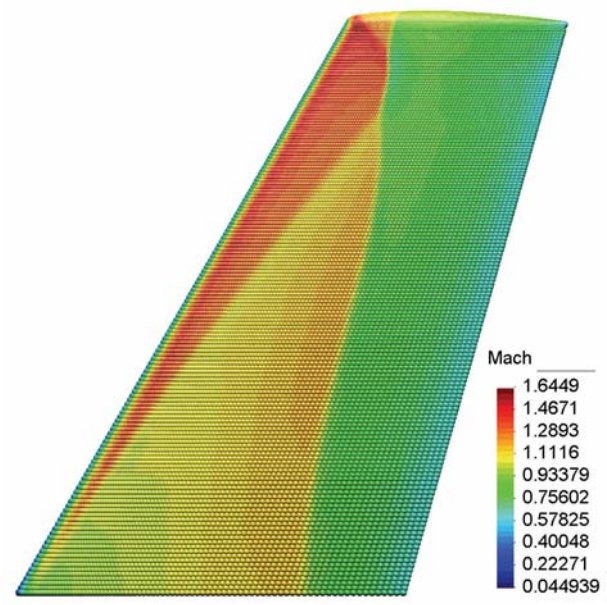

Figure 9: Surface discretization of the ONERA M6 wing (upper surface view); coloured points display Mach number values. $M_\infty=0.84$ and $\alpha=3.06º$.

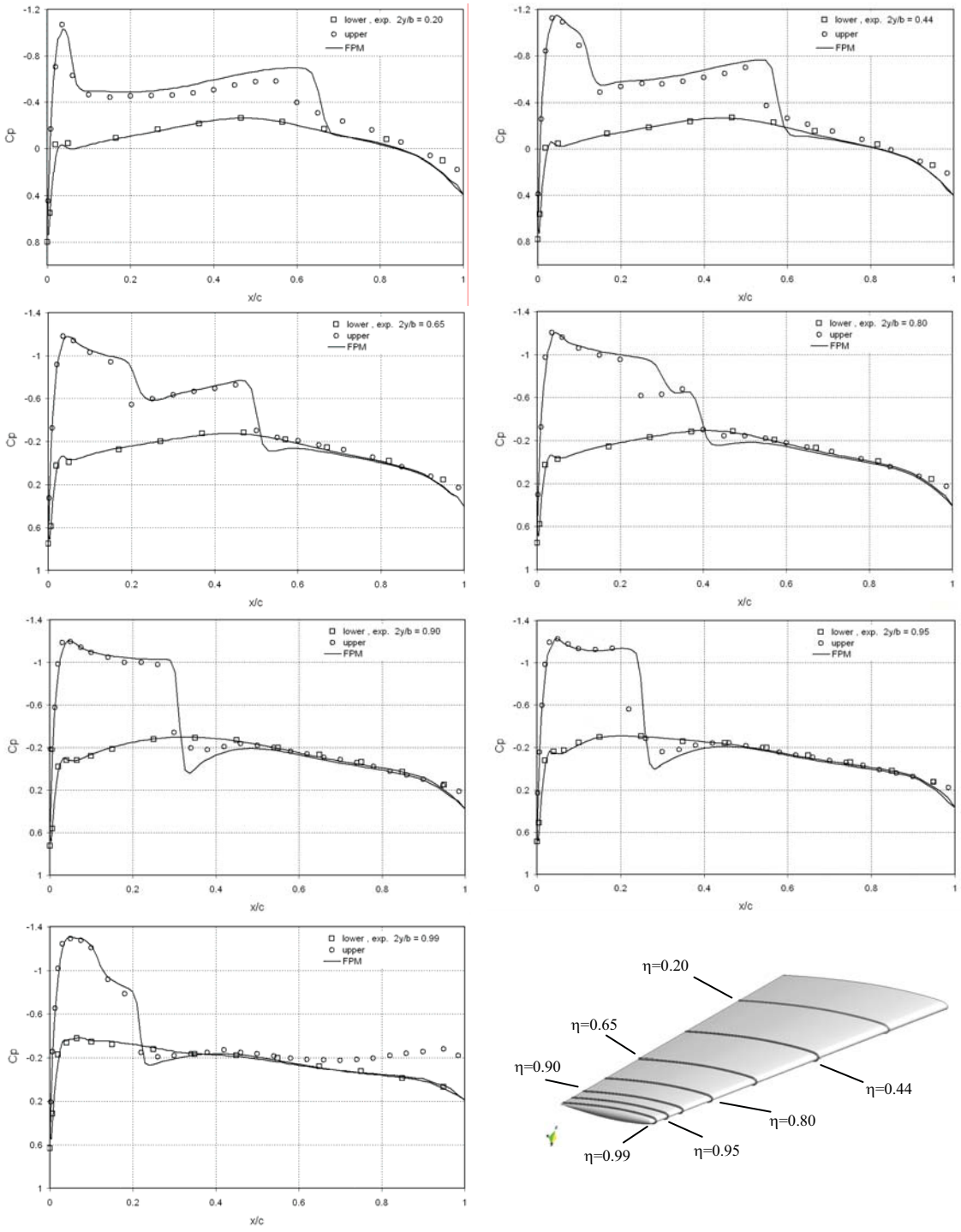

Figure 10: Comparisons between computed and experimental Cp distributions along several sections on the wing. ONERA M6 wing, M_∞=0.84 and α=3.06º.

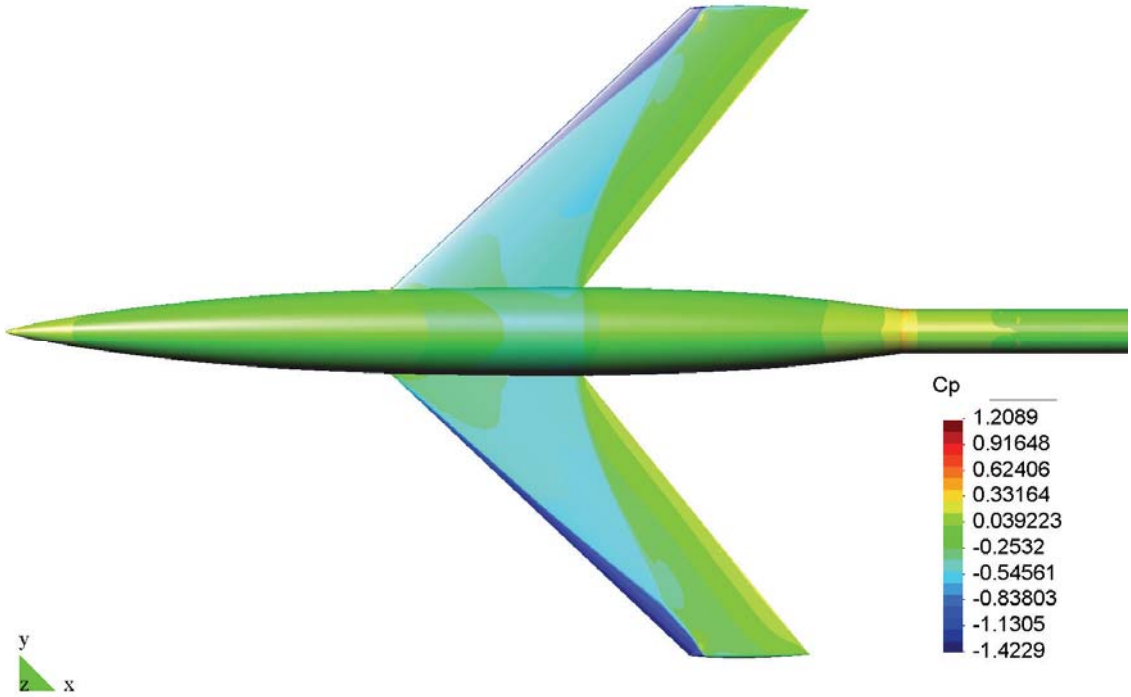

Figure 11: Cp distribution on the NACA wing-body configuration (only half of the model has been calculated, the other part is simply included for visualization purposes). $M_\infty = 0.90$ and $\alpha = 4.0^o$.

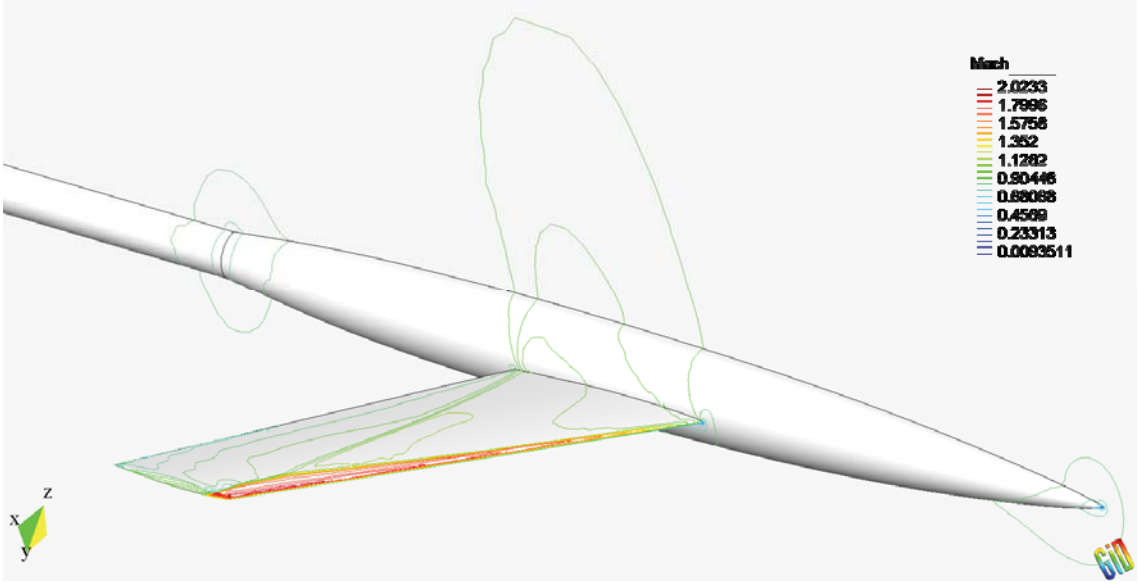

Figure 12: Mach number isolines on the NACA wing-body configuration and the symmetry plane. $M_\infty = 0.90$ and $\alpha = 4.0^o$.

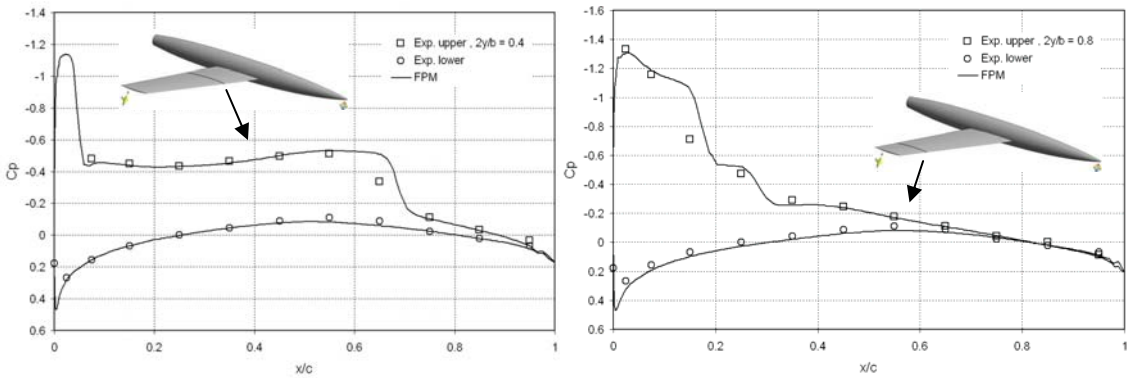

Figure 13: A comparison between computed and experimental Cp distribution along two spanwise wing stations $\eta = 0.4$ and $\eta = 0.8$. NACA wing-body configuration, $M_\infty = 0.90$ and $\alpha = 4.0^o$.

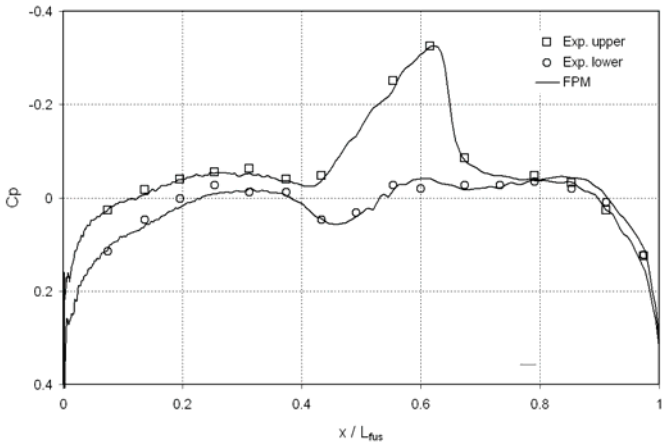

Figure 14: Comparison between computed and experimental Cp distribution along the fuselage symmetry plane. NACA wing-body configuration, $M_\infty = 0.90$ and $\alpha = 4.0^o$.

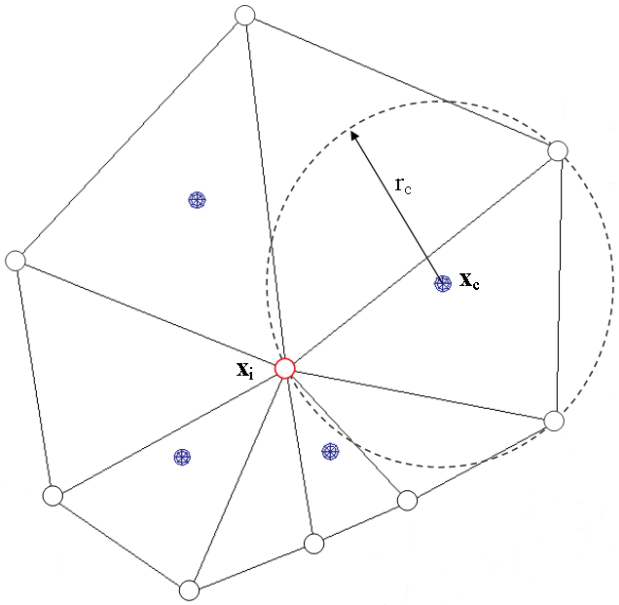

Figure 15: Refinement of a bi-dimensional cloud of points. The filled points $\boldsymbol{x}_c$ meet the requirements r_1-r_3 and, in consequence, are inserted around the star point $\boldsymbol{x}_i$.

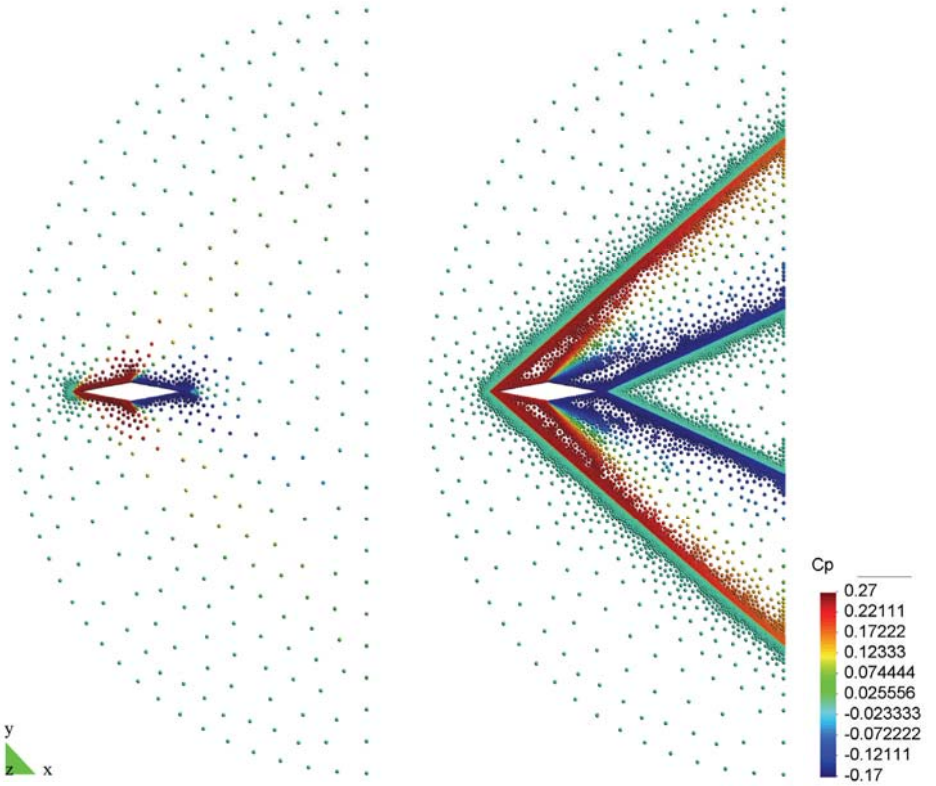

Figure 16: Supersonic flow past a double wedge airfoil. Left: original coarse discretization; Right: final adapted discretization (70 refinement levels). The coloured points show Cp results. M_∞=2.0 and α=0º.

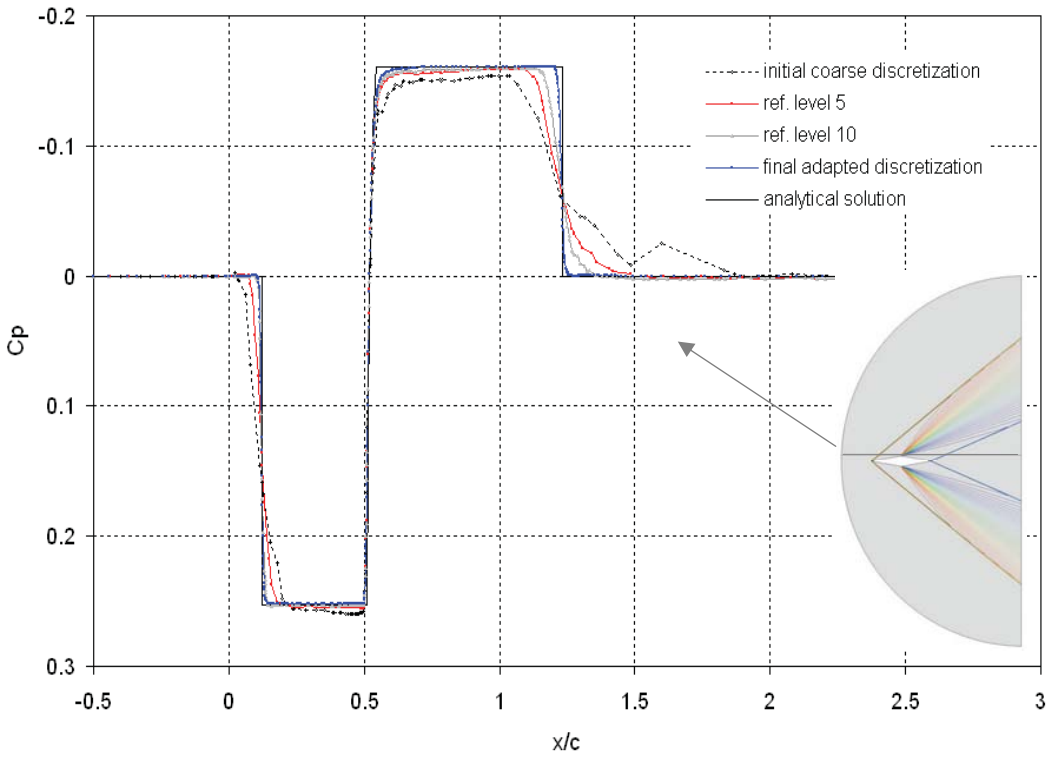

Figure 17: A comparison between the analytical Cp distribution along an x-cut on the domain and computed numerical results obtained at different refinement levels. The cut is located at y/c=0.1 and the airfoil leading edge coincides with the point (x,y) = (0,0). M_∞=2.0 and α=0º.

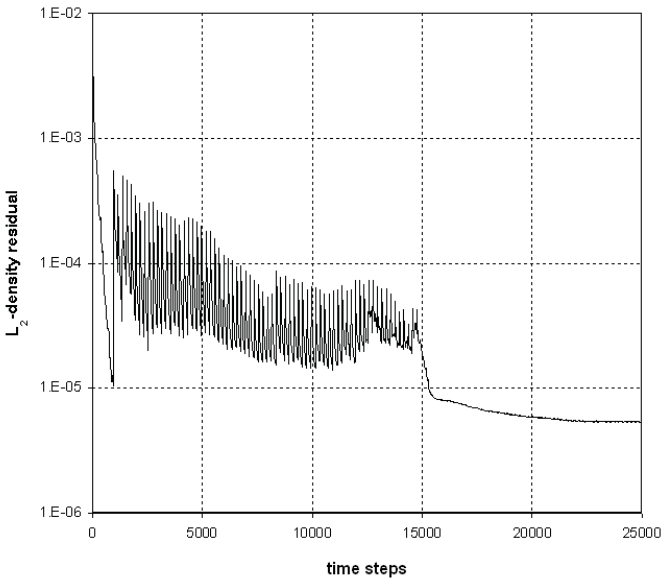

Figure 18: Convergence history of the double wedge airfoil calculation (70 refinement levels). M_∞=2.0 and α=0.0º.

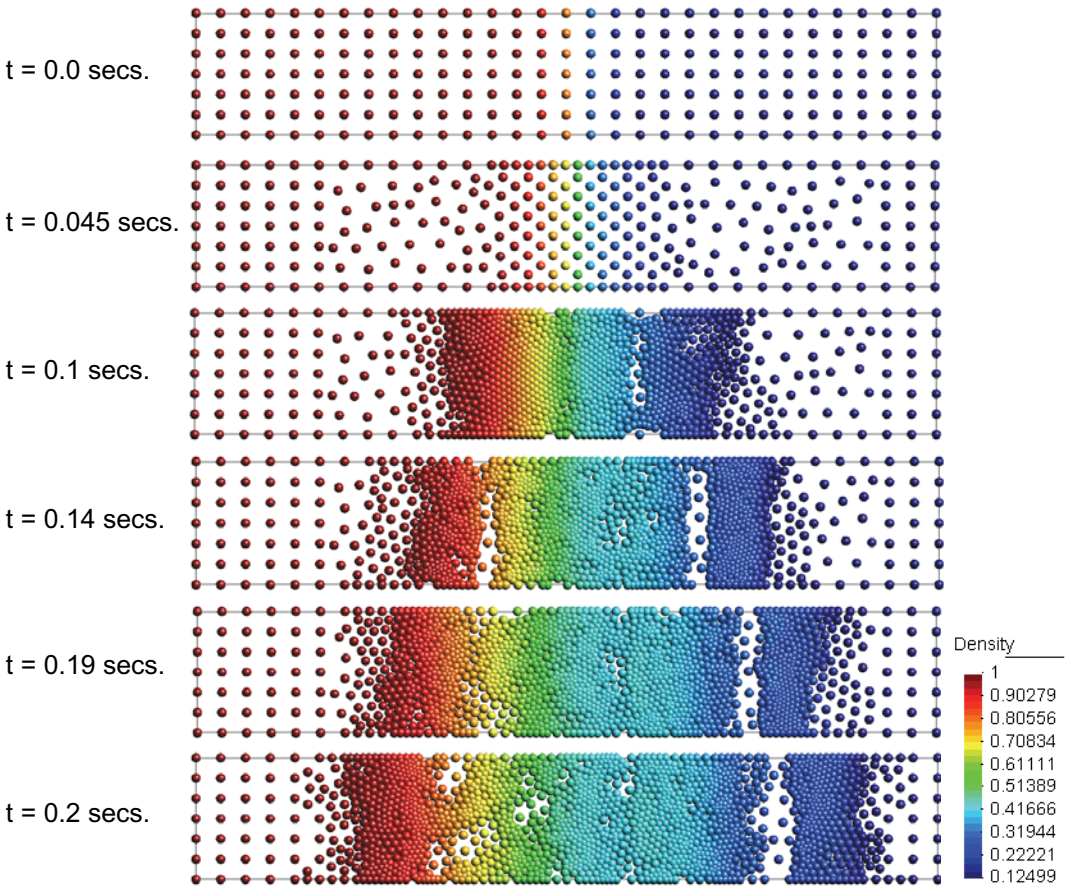

Figure 19: Adapted discretizations obtained for the shock-tube problem (p_L/p_R=10) at different times from the rupture of the diaphragm (the top image shows the initial coarse discretization).

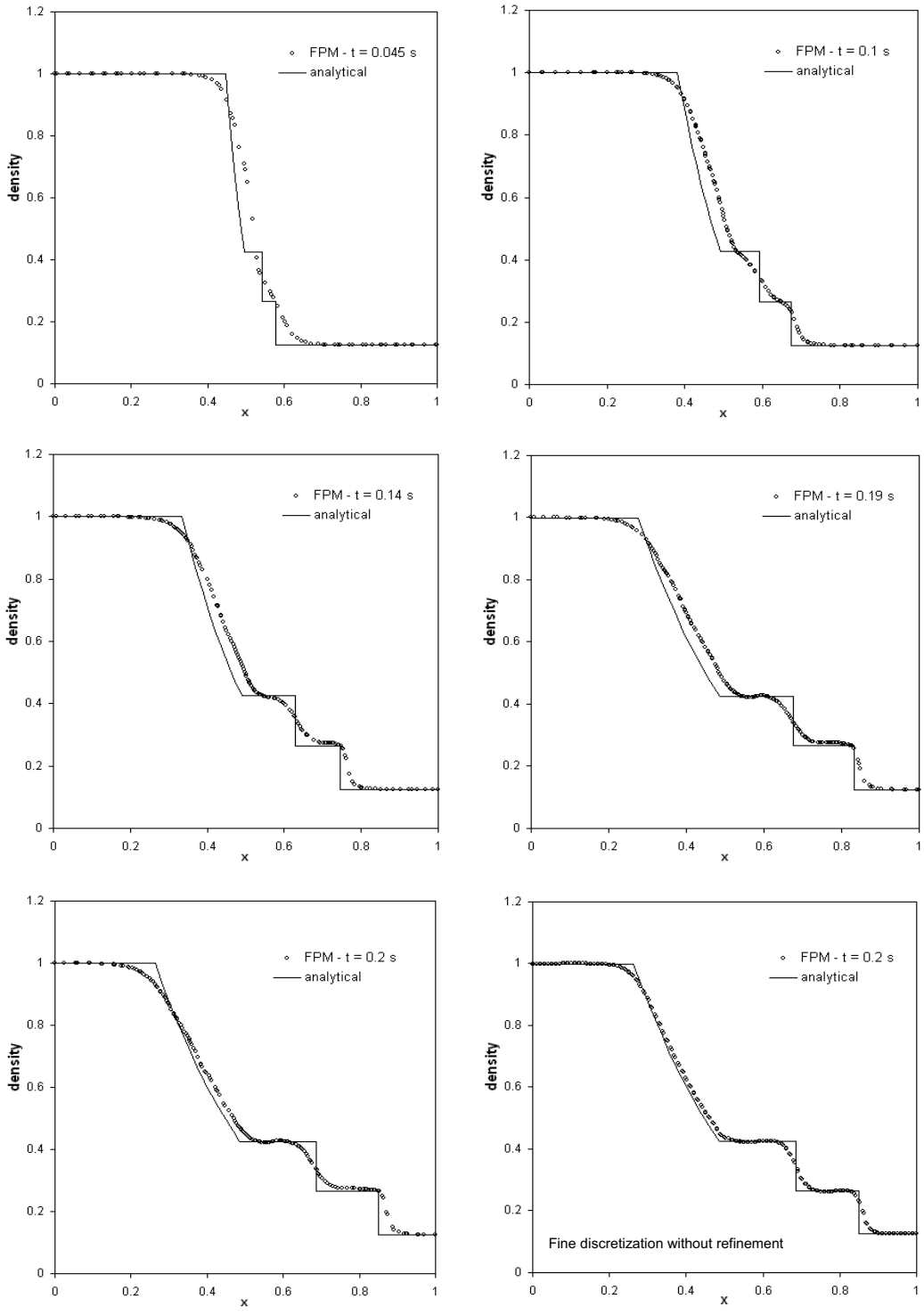

Figure 20: Comparison between numerical and analytical solutions for the density distribution along the centreline of the shock tube at different times from the rupture of the diaphragm (p_L/p_R=10). The numerical solution at the bottom left corner is calculated using a fine discretization without performing any refinement level.

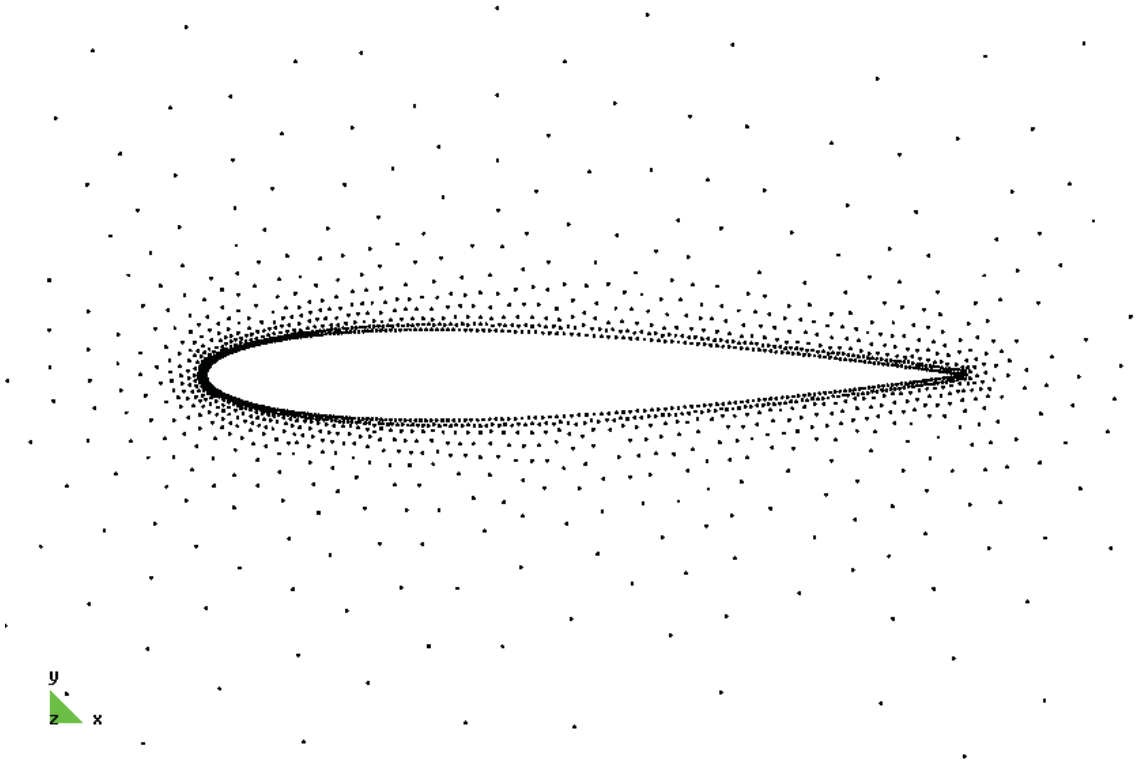

Figure 21: A view of the original coarse discretization in the proximity of the NACA 0012 airfoil.

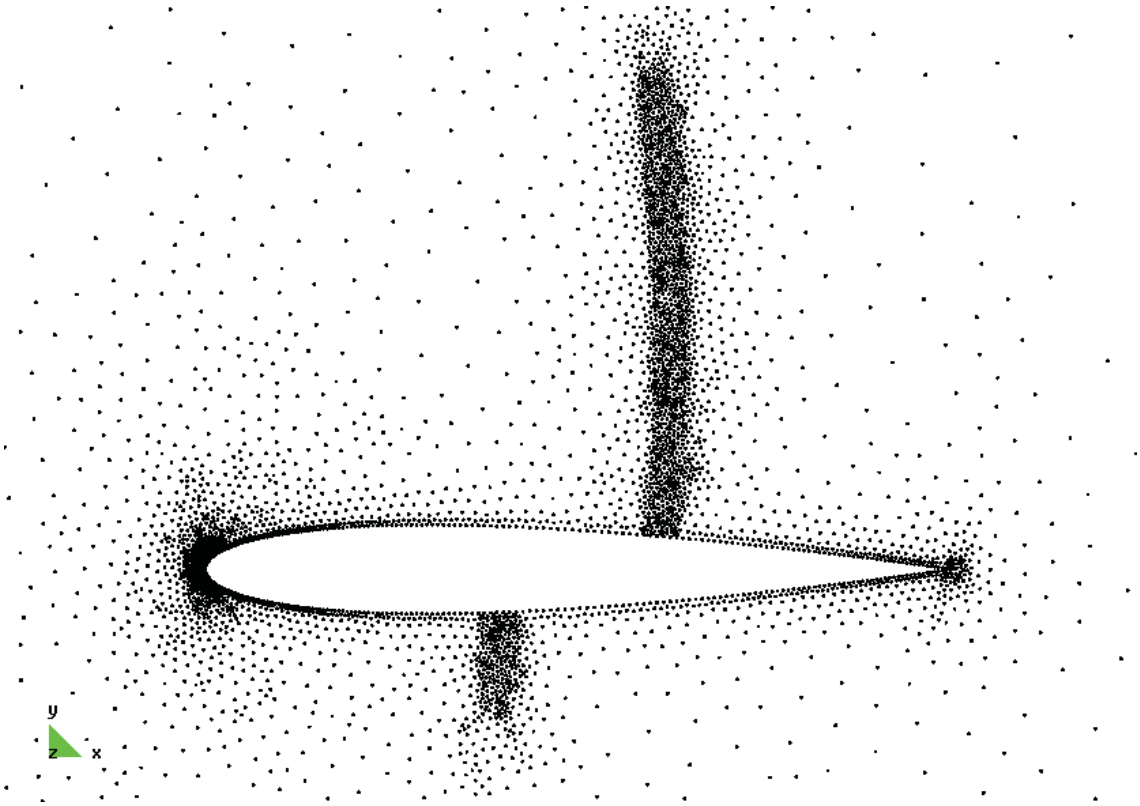

Figure 22: A view of the finest adapted discretization in the proximity of the NACA 0012 airfoil obtained after 15 refinement levels.

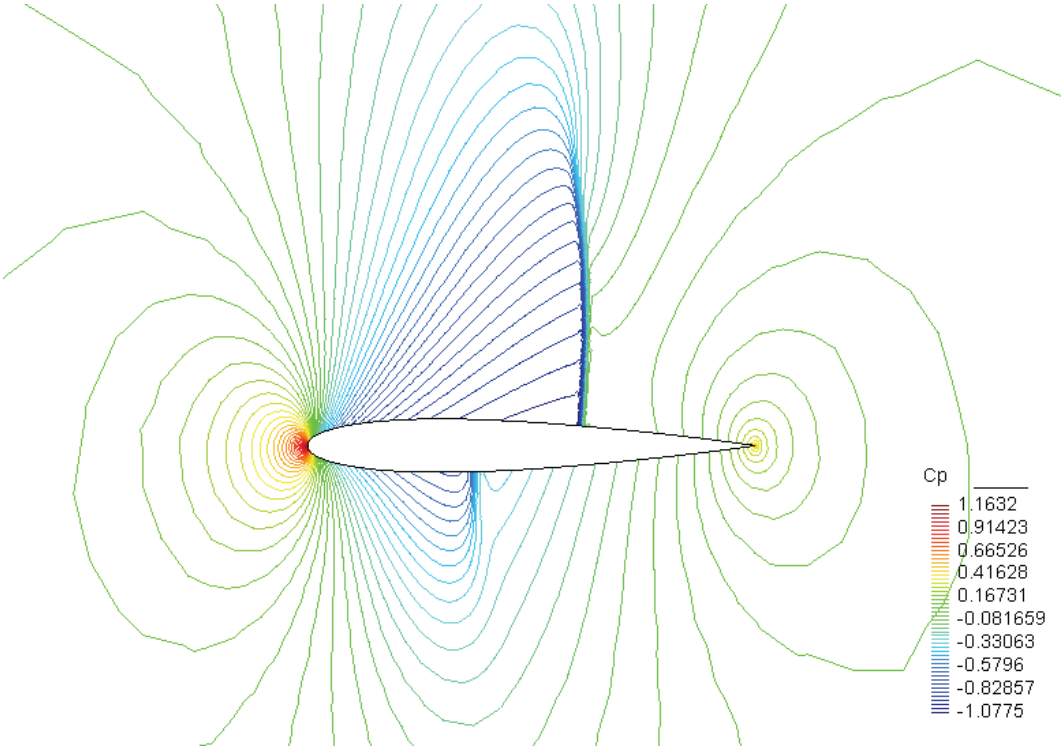

Figure 23: Cp isolines in the near-field of the NACA 0012 airfoil obtained with the finest adapted discretization. M_∞=0.80 and α=1.25º.

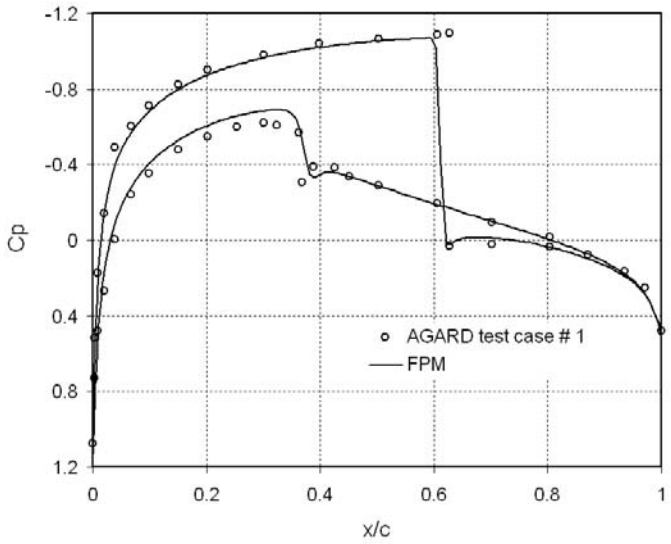

Figure 24: Cp distribution on the NACA 0012 airfoil obtained with the finest adapted discretization. A comparison between computed and numerical reference results [37]. M_∞=0.80 and α=1.25º.

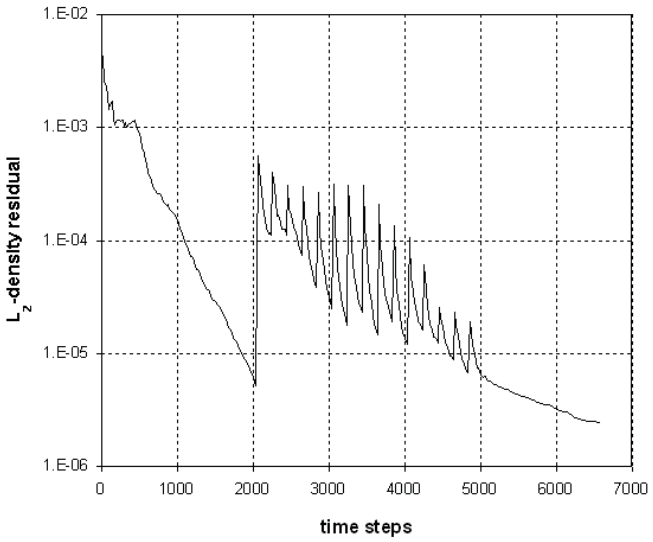

Figure 25: Convergence history of the NACA 0012 airfoil calculation (15 refinement levels). M_∞=0.80 and α=1.25º.

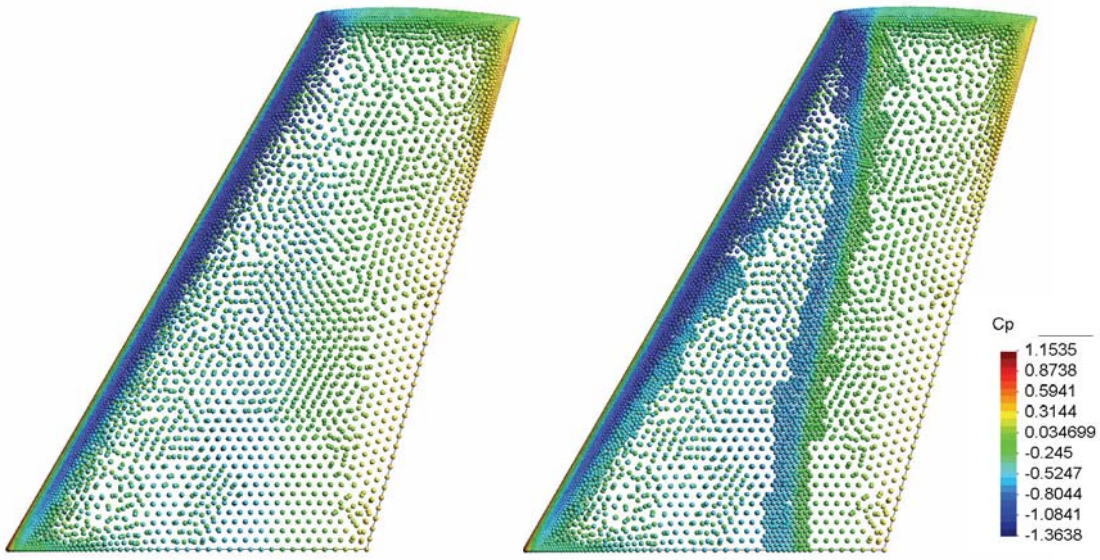

Figure 26: A view of the upper side of the ONERA M6 wing. Left: original coarse discretization; Right: finest adapted discretization (35 refinement levels). M_∞=0.84 and α=3.06º.

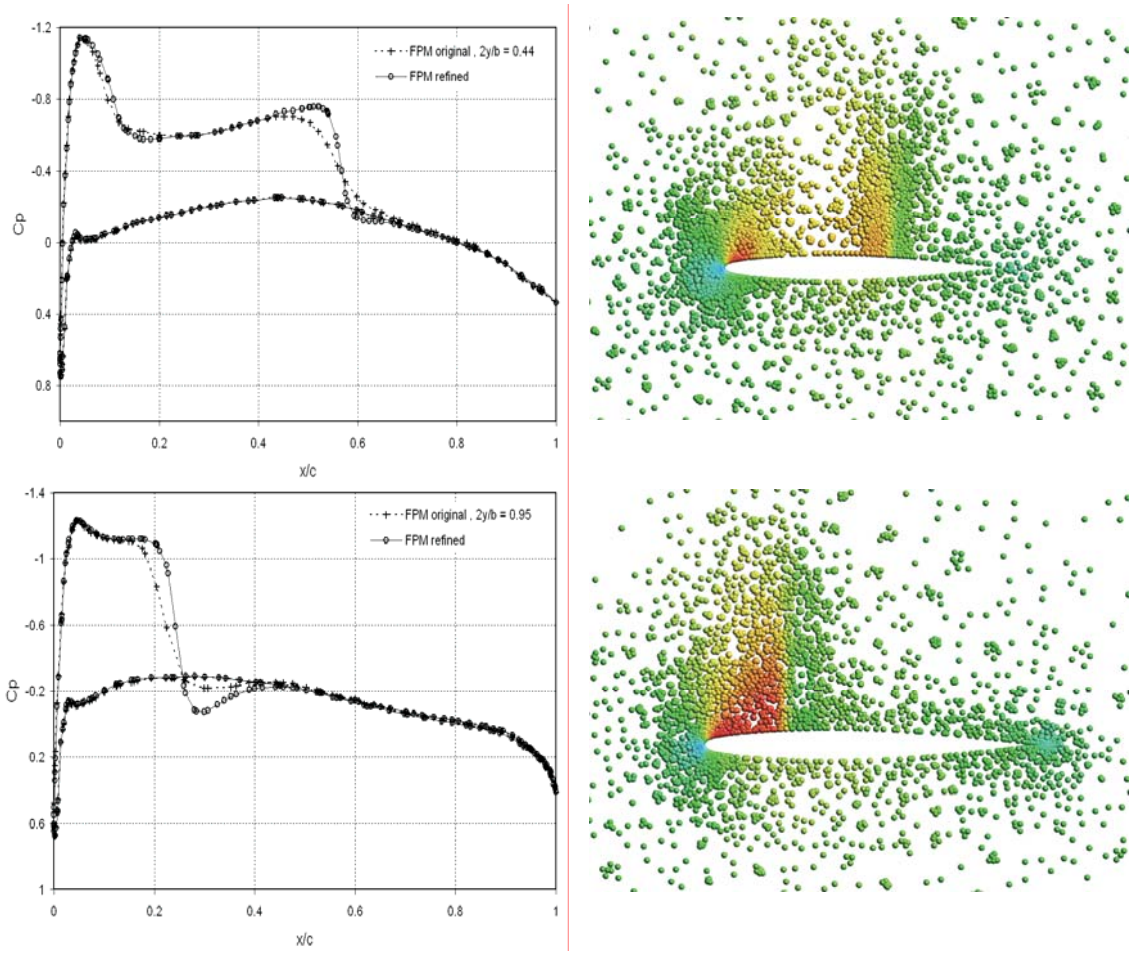

Figure 27: Left: Cp distributions along two wing sections η=0.44 (top) and η=0.95 (bottom) calculated with the original and finest discretizations. Right: cuts x-z of the finest refined domain passing through wing stations η=0.44 (top) and η=0.95 (bottom). ONERA M6 wing M_∞=0.84 and α=3.06º.

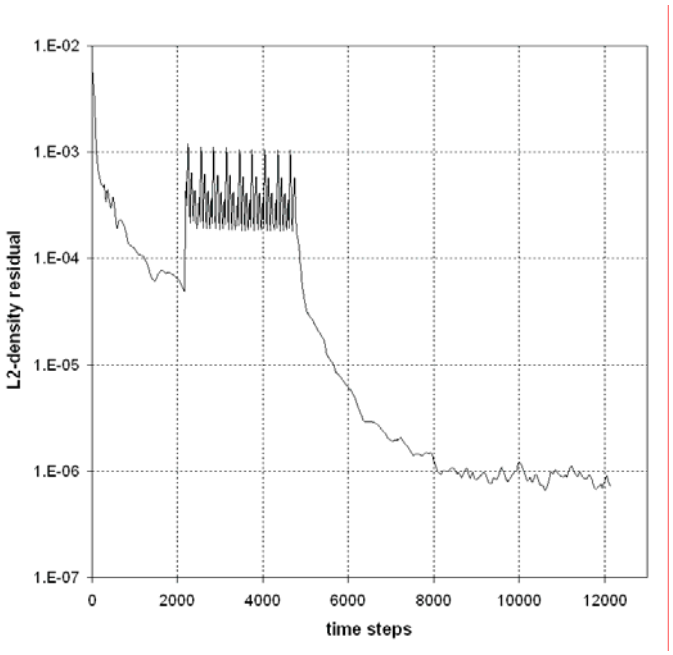

Figure 28: Convergence history of the ONERA M6 wing adaptive calculation (35 refinement levels). M_∞=0.84 and α=3.06º