AN H-ADAPTIVE UNSTRUCTURED MESH REFINEMENT STRATEGY FOR UNSTEADY PROBLEMS

G. A. RIOS RODRIGUEZ, N. M. NIGRO and M. A. STORTI

Centro Internacional de Métodos Computacionales en Ingeniería CIMEC
Universidad Nacional del Litoral, CONICET
Güemes 3450, 3000, Santa Fe, Argentina
gusadrr@yahoo.com.ar, mstorti@intec.unl.edu.ar

***Abstract*— An h-adaptive unstructured mesh refinement strategy to solve unsteady problems by the finite element method is presented. The maximum level of refinement for the mesh is prescribed beforehand. The core operation of the strategy, namely the elements refinement, is described in detail. It is shown through numerical tests that one of the advantages of the chosen refinement procedure is to keep bounded the decrease of the mesh's quality. The type of element is not changed and no transition templates are used, therefore hanging nodes appear in the adapted mesh. The 1-irregular nodes refinement constraint is applied and the refinement process driven by this criterion is recursive. Both the strength and weakness of the adaptivity algorithm are mentioned, based on clock time measures and implementation issues. To show the proper working of the strategy, an axisymmetric, compressible non-viscous starting flow in a bell-shaped nozzle is solved over an unstructured mesh of hexaedra.**

***Keywords*— h-adaptivity, unstructured meshes, Quality metrics, hanging nodes.**

I. INTRODUCTION

The benefits of having some kind of mesh adaptivity tool to solve problems by the finite element method are well recognized by the computational fluid dynamics community. Amongst them, h-adaptive or mesh enrichment procedures have the advantage that they do not need to modify the fluid dynamic solver to be used.

The development of an h-adaptive element-based unstructured mesh strategy to solve unsteady problems by the finite element method is described in this work. It can be applied both to two- and three-dimensional unstructured linear finite element meshes. The adaptivity algorithm is designed considering the element refinement stage as the core of the strategy. This allows to extend the algorithm from 2-D meshes to 3-D ones with little extra effort. The refinement algorithm has been succesfully tested, solving steady fluid dynamic problems on meshes made up of triangles, quadrangles and tetrahedra (Ríos Rodriguez et al., 2005). Special care has been taken to keep bounded the quality decrease of the mesh. Edge midpoints or regular 1:4 and 1:8 refinement seems to be the best choice in this sense, for two- and three-dimensional elements correspondingly. It is also shown based on numerical tests that the best choice (from the quality standpoint) to regularly refine a tetrahedron in 8 sub-tetrahedra is to choose the shortest diagonal of the inner octaedron that arises in the partitioning process. Hanging nodes have to be managed since no transition elements are used to match zones of different refinement levels. To ensure a smooth transition in element size between zones with different levels of refinement, the 1-irregular node constraint amongst neighbouring elements is considered. Consequently, the refinement process is recursively designed and the solution must be constrained on these hanging nodes. The selection method of the elements to be refined is shortly described as well. In this work, the strategy is used to solve the unsteady starting flow in a bell-shaped rocket nozzle over an unstructured mesh of hexaedra. While the adaptivity stage runs on a single PC, the fluid dynamic problem solver developed by Storti et al. (1999-2007) runs in parallel on a Beowulf cluster of PC. To show the algorithm efficiency, clock time is measured using a uniformly refined mesh equivalent to the adapted one. Finally, the advantages as well as the disadvantages of the strategy are highlighted.

II. Element refinement

It is considered that the main issue in the design of the refinement algorithm is to minimize the quality drop of the adapted mesh, since high quality meshes are often desired for numerical reasons. Computational cost and programming simplicity are also taken into account in the process design. If a regular 1:8 tetrahedron's refinement is applied, four similar subtetrahedra are obtained at the corners of the parent element and an inside octahedron results. By adding an edge, the octahedron can be splitted into four tetrahedra, as

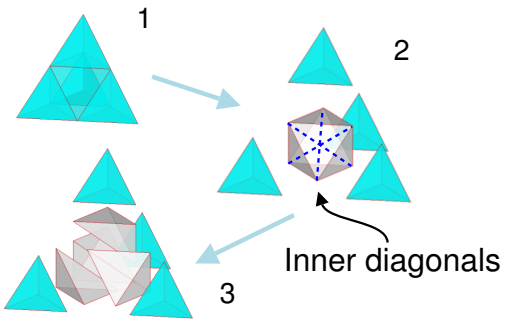


Figure 1: Regular 1:8 tetrahedron's refinement sequence.

Fig.1 shows.

One of the three possible diagonals must be chosen (this diagonals are marked as doted lines in the figure). It is usual to do this randomly but, is there a better choice from the quality standpoint? To answer this question numerical tests are performed.

A. Quality Tests

Several element's shape measures exist in the literature. Some of them are based on geometrical properties of the element while others are related to algebraic properties of the Jacobian matrix and the matrices thereof derived. Knupp (2000) developed the concept of algebraic mesh quality metrics and derives a complete list of them. Here, it is considered that the quality of the mesh is equal to that of the most poorly-shaped element. The tetrahedron's quality is measured using the following algebraic quality metric introduced by Liu and Joe (1994)

$$\eta(T) = 3\sqrt[3]{\lambda_1\lambda_2\lambda_3}/(\lambda_1+\lambda_2+\lambda_3) \qquad (1)$$

where $\lambda_1, \lambda_2, \lambda_3$ are the eigenvalues of the metric tensor which transforms the original tetrahedron T to the regular tetrahedron R which has the same volume as T. They show in the same work that this is equivalent to

$$\eta(T) = \frac{12(3v)^{2/3}}{\sum_{i=1\ldots6} l_i^2} \qquad (2)$$

where v is the volume of the tetrahedron and l_i are the lengths of the edges. Also $\eta(T) = 1$ for the regular tetrahedron and $\eta(T) \to 0$ when the element is poorly-shaped (such as a wedge, a needle or a sliver).

Two quality tests are carried out. The first refinement test starts with tetrahedra of three different qualities, namely a regular, a right and a poorly-shaped. For all refinement iterations, either the longest or the shortest diagonal of the inner octahedron is chosen for its partitioning, and the worst quality element obtained from applying the regular 1:8 subdivision is always refined. No conformity of the mesh is considered.

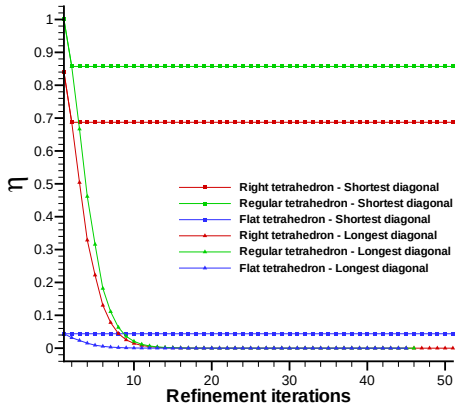


Figure 2: Quality metric vs. refinement iterations for the first quality test

Figure 2 depicts the mesh quality behaviour as a function of refinement iterations.

It is seen that the quality decrease produced by successive refinement is minimized if the shortest diagonal of the inner octahedron is always chosen. In that case, the quality of the mesh diminishes only at the first refinement iteration and then keeps constant. Otherwise, the quality of the mesh diminishes with each refinement iteration, until elements with negative volumen appear.

In the second test, a sensitivity analysis to the initial tetrahedron's shape is performed. Only one refinement iteration is done, always choosing the shortest diagonal for refining. The starting tetrahedra are obtained moving the apex of the regular tetrahedron on a plane parallel to its supporting face, as Fig.3 depicts. A non-linear dependency of the quality's drop on the shape of the initial element is observed in Fig.4. The sensitivity is shown as a contour map of the ratio $\min(\eta(T^1))/\eta(T^0)$, where the superscripts indicate the number of refinement iteration. This figure shows that for some initial tetrahedra the quality does not dimish if the shortest diagonal is always chosen.

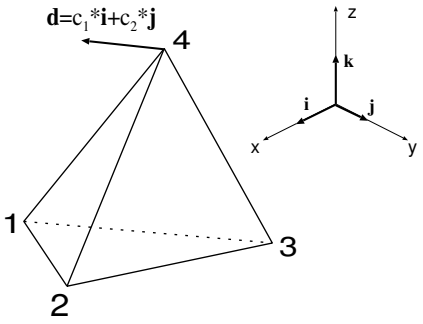


Figure 3: Starting tetrahedron.

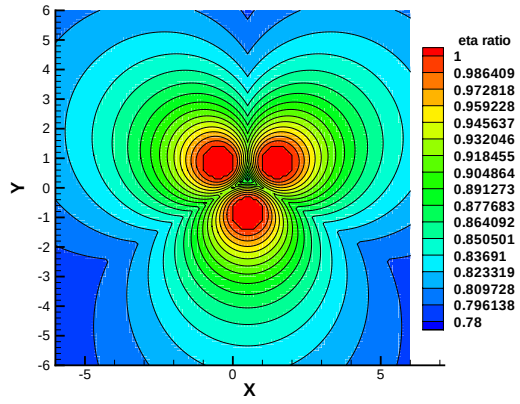


Figure 4: Shape measure ratio $\eta(T^1)/\eta(T^0)$ contours for first refinement iteration - Shortest diagonal.

B. Refinement Algorithm

Adaptivity algorithms for unsteady problems must be computationally inexpensive, since refinement / coarsening operations are applied many times until the final instant. To this end, it is decided both not to use transition elements to match zones with different levels of refinement and to use only one subdivision pattern for each type of element. As a consequence: a) the adapted mesh is non-conformal or has hanging nodes b) the 1-irregular node refinement constraint must be applied to ensure a gradual transition in the elements' size, c) no quality decrease is introduced due to poorly-shaped transition templates usage, d) there is no need to use compatiblity rules for refinement / coarsening, and finally, e) the solution must be constrained on the hanging nodes. Figure 5 shows an example regarding to the 1-irregular node constraint if the shaded element is the one to be refined. Hanging nodes are marked with dots in the same figure. The 1-irregular node constraint says that *no more than 1 hanging node should be shared amongst neighbouring elements through the common edge (2D and 3D) or face (3D) to which the node belongs*. These kind of mesh refinemement was first proposed by Babuska and Rheinboldt (1978) for 2-D elements. In this work it is extended to the 3-D case. Some issues concerning to this kind of refinement are important: a) the problem solver has to be able to manage the constraint of the solution on the hanging nodes, b) the zone to be refined will be "wider" than the zone of interest in the solution field due to this solution constraint and c) some difficulties arise to visualize results over meshes with irregular nodes, since contours are usually drawn element-wise for unstructured finite elements grids (Hannoun and Alexiades, 2007) and discontinuous contours appear.

To describe the refinement algorithm the following

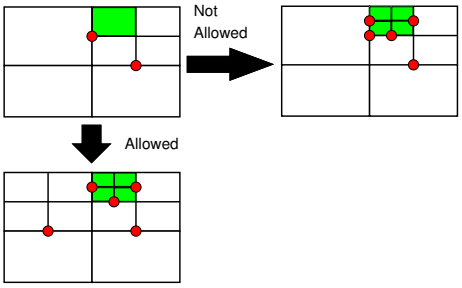


Figure 5: 1-irregular node refinement constraint in 2-D.

concepts about elements, faces and edges are introduced. It is said that an edge is *active* if not *all* the elements that share it are refined, otherwise it is *inactive*. The *parent* of an edge is the edge from which it derives by inserting a midpoint on the *parent* edge. The *parent* of a face is the face from which it derives by joining with edges the midpoints (and centre point in quads) that appear on the *parent* face. All the faces and edges that appear *inside* refined elements have no *parents*. Refined elements are always *deactivated*. Finally, every edge has two *childs*, every face has four *childs* and every element can have either 4 (2-D) or 8 (3-D) *childs*. These geometrical entities are organized in a hierarchical manner, so we say that an entity belongs to a low level of refinement if it has a higher hierarchy, and conversely.

Given a list of elements to be refined *eles2ref*, created either at the error estimate stage or prescribed by the user, the following algorithm is implemented:

1. Find the edges that belong to *eles2ref*. These edges are called *edges2reftmp*.
2. Find the edges in the set *edges2reftmp* that have not been refined yet and call them *edges2ref*.
3. Search the parent edges of the *edges2ref*. If some of them are still *active*, put them in a list *ParentEdges2ref*.
4. Find the elements to which these *ParentEdges2ref* belong and build a list *eles2refNEW*.

The same steps are taken for the faces. As a result, a new list of elements that should be refined at a lower refinement level is built within the actual iteration. Now, the refinement function is called with this list as argument. The recursivity ends when the list *eles2refNEW* in the actual iteration is empty. Then, the elements added to satisfy the 1-irregular criterion are refined from lower to higher levels until the original list of elements to be refined can be treated.

III. Adaptivity strategy

The unsteady strategy was developed based on the following concept: the adapted mesh is updated every n-time steps using a starting or *base mesh*, as the diagram on Fig.6 depicts. For each one of these *adaptivity steps* (except for the first), some elements are marked to be refined, based on the error estimate chosen for the problem and the final solution computed at the previous adaptivity step. The elements to be refined are always selected at the maximum level of refinement. Since no higher level of refinement than the prescribed by the user is desired, a *parents search* algorithm should be used: if elements which do not belong to level 0 of refinement are selected, it is required to search for the parents of these elements in the data structure corresponding to this adaptivity step. The elements selected for refining are replaced by their parents and the process is repeated recursively until none of the elements in the list have parents (level 0). In this way, *preliminary* lists of elements to build each refinement level at the next adaptivity step are obtained. This process is represented by dashed lines in the diagram. Then, the next adaptivity step begins to be built.

Since the elements numbering scheme of the adapted meshes at the same refinement level changes from one adaptivity step to the next, the numbering of the elements in the preliminary list requiered to build that level needs to be updated. Once the list is updated, the corresponding level can be created. The updating and refinement processes are represented on the diagram by continuous and dotted lines, correspondingly. Finally, the problem solver is restarted using the most refined mesh. Only for the first adaptivity step, the problem solver is run a few time steps on the base mesh. Then, elements are marked and refined as many times as it is needed, based on the magnitude of the error estimates. The strategy does not consider coarsening of the *base mesh* and the adaptivity frequency is also constant through the whole run.

An initial state to restart the equation's solver is computed as the linear interpolation of the solution on the last adapted mesh, at the nodes of the base mesh. The boundary conditions are also updated at every adaptivity step. To this end, geometrical entities of the mesh have a property flag attached to it. This flag is associated with a special property, such as a fixation, a periodic or a slip constraint, a geometrical or an element's property or maybe an allowed combination of them. The properties are inherited from parents to childs at the refinement stage by inheritance of the flag, and updated lists of geometrical entities with that properties are built in a post-refinement stage within the adaptivity step.

IV. Error Indication

Since the exact solution of the problem is not known *a priori*, error estimates should be used to find which

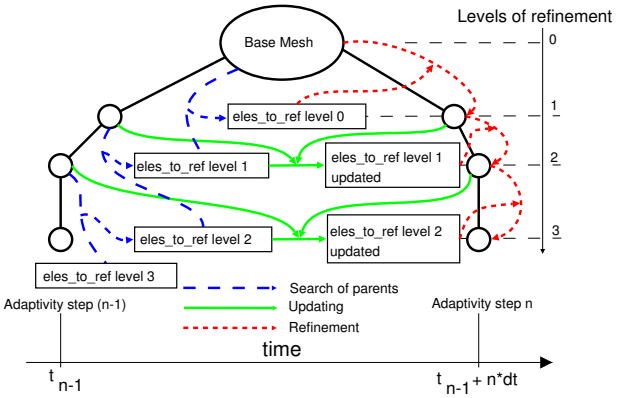


Figure 6: Adaptivity strategy for unsteady problems.

are the zones of the mesh that have to be refined. A great number of error estimate methods have been devised, most of them for elliptic problems. Mavriplis (1995) says that the choice of the right error indicator depends on multiple factors, such as the relative cost to compute the error estimate in regards to that of the rest of the adaptivity process, the affordable error estimate precision and the mathematical character of the equations. Heuristic gradient-based methods are frecuently used in hyperbolic problems because they are computationally chipper, they let the zones to be refined be properly identified and because a more rigurous theory needs to be developed for hyperbolic equations. But success in using these kind of methods depends on the user's experience to choose the right combination of flow variables that better suit. For compressible flows, the gradient of the density and/or pressure fields are often used. An element is marked to be refined when the element-wise computed gradient multiplied by a measure of its size is greater than a prescribed tolerance value ϵ^* for the error, that is

$$\epsilon \geq \epsilon^* \tag{3}$$

where

$$\epsilon \simeq \frac{\partial \bar{u}}{\partial x} \cdot h = \Delta \bar{u} \tag{4}$$

V. Test case

To check the proper working of the unsteady strategy, the transient non-viscous supersonic flow at the starting of a bell-shaped rocket nozzle is solved. Also, to find out if there is a true efficiency gain, the same problem is solved over a fixed mesh finer than the base mesh used for the adaptive process and the clock time required by both runs is compared.

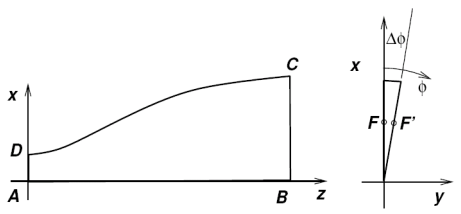


Figure 7: Problem setup for the rocket nozzle.

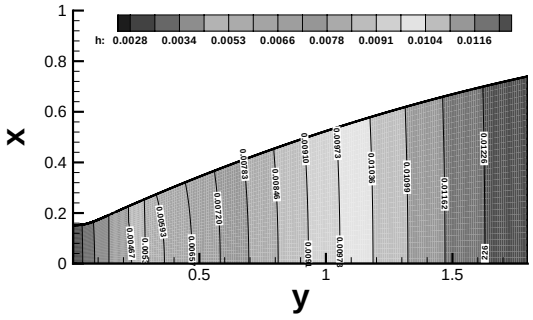


Figure 8: Element size distribution for the base mesh on the $X - Z$ plane.

A. Problem Setup

While the axisymmetric flow equations are solved over an unstructured mesh of hexahedra, the adaptivity strategy is applied over the 2-D grid on the X-Z plane. Once the adapted mesh of quads is obtained, it is projected using the rotation matrix $R(\phi)$. Figure 7 displays the problem setup. Only one layer of hexahedra is used in the circumferential direction ϕ. Hexahedra's nodes along the axis of simmetry are collased so that these elements turn into wedges.

Because of the axial symmetry of the flow, the following boundary conditions are considered

- Slip condition $\vec{V}\cdot\vec{n} = 0$ at the wall $\bar{DC}$ and along the axis of simmetry $\bar{AB}$.
- Dynamic boundary conditions developed by Storti et al. (2008) are imposed at the exhaust section $\bar{BC}$.
- Pressure and density are constants at the throat section $\bar{AD}$. They linearly increase with time from the surrounding up to the stagnation values ($\rho_0 = 0.47541\text{kg/m}^3$, $p_0 = 0.6\text{MPa}$, $T_0 = 4170\text{K}$). Transient time for this condition is assumed to be $t = 5\mu\text{sec}$.
- Axial flow at the throat section ($V_{xAD} = 0$).
- There is no flux in the circumferential direction ($V_\phi = 0$).
- Circumferential periodicity at every point of the domain. This means that scalar fields are the same at points F and F' while vector fields, such as velocity, are constrained to have the values at point F, "rotated" by the same matrix R that transforms the coordinates from point F to point F'.

On the other hand, the initial conditions in the whole domain take the following values ($\rho_{ini} = 0.0018\text{kg/m}^3$, $u_{ini} = 0\text{m/s}$, $p_{ini} = 143\text{Pa}$, $T_{ini} = 262\text{K}$). The gas is considered to be air, and the isentropic coefficient value is $\gamma = 1.17$.

B. Solving the Problem

The 3-D Euler equations are solved using the Advection-Diffusion module of the PETSc-FEM multi-physics FEM solver developed by Storti et al. (1999-2007), using 10 nodes of the Beowulf PC Pentium IV cluster.

The Courant number for both the adaptive and non-adaptive simulations is $Cou = 2$, except for the first time steps corresponding to the transient time at the throat conditions, when $Cou = 0.2$. Selection of the elements to be refined is based on the magnitude of the element-wise computed velocity gradient $\nabla\vec{U}(T)$. Velocity gradient is chosen because its scale range makes easier to apply the following criterion for selecting the elements to be refined

$$c_1 \cdot \max_T(|\nabla\vec{U}|) \leq |\nabla\vec{U}| \leq c_2 \cdot \max_T(|\nabla\vec{U}|) \qquad (5)$$

where constants c_1 and c_2 are adjusted beforehand for the whole run. Besides, the velocity gradient allows to capture the desired flow features with the mesh refinement.
One level of refinement is prescribed for the whole run and the adaptivity frequency is 5 time steps. This means that every 5 time steps the mesh is re-adapted to the last computed solution. It is considered that the problem has reached its stationary state for $t_f = 8.4\text{e-}4\text{sec}$, so this is the final time for the simulation. The base mesh for the adaptive simulation has 9330 elements and 9672 nodes while the fixed mesh has 11430 elements and 11842 nodes. Both of them have smaller size elements at the throat section. Figure 8 shows the size distribution of the elements for the base mesh, which is computed as the radius of the circumscircle of the quadrilateral elements on the $X - Z$ plane.

C. Results

The adaptive simulation requieres a few more time steps to reach the final time than the non-adaptive one, namely 2219 time steps for the latter and 2280 time steps for the former. The time steps number difference is due to the fact that, although the base mesh

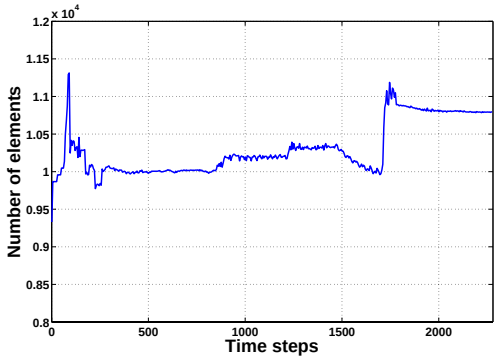


Figure 9: Behaviour of the number of elements for the adaptive run.

has fewer elements than the fixed, both the smaller size of the elements at the adapted zones and the CFL condition makes the time step be reduced. Figure 9 shows that the number of elements for the adaptive simulation was never higher than the corresponding for the non-adaptive run. The strange behaviour in the number of elementes is due to the fact that the value $\max_T(|\nabla\vec{U}(T)|$ changes with time. Although this difference in the number of time steps in favour of the non-adaptive simulation, the adaptive one is faster.

The axial velocity profiles are measured along the axis of simmetry at different time instants. They are shown at Fig.10, where it can be seen that better resolution profiles correspond to the adaptive solution. The velocity increases behind the shock fronts which are travelling to the exit section of the nozzle. The local velocity shoot up immediately behind the secondary shock that appears in the non-adaptive profile at t = 5e-4sec. and t = 6e-4sec. does not almost exists in the adaptive solution. It is worth to mention that all the variables of the flow were scaled due to numerical reasons. Therefore the velocity scale should be multiplied by V_{ref} = 1215.16m/sec to get the actual velocity values in the figures.

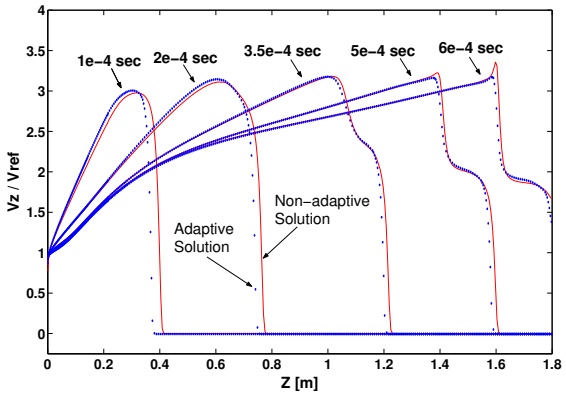
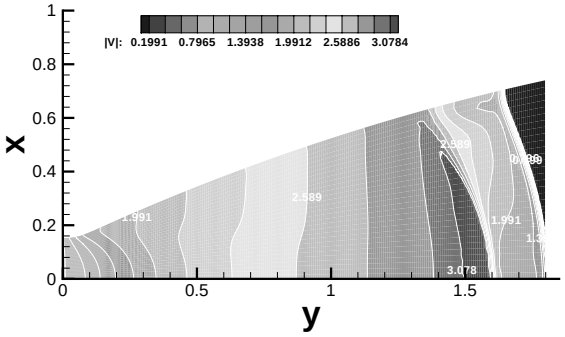
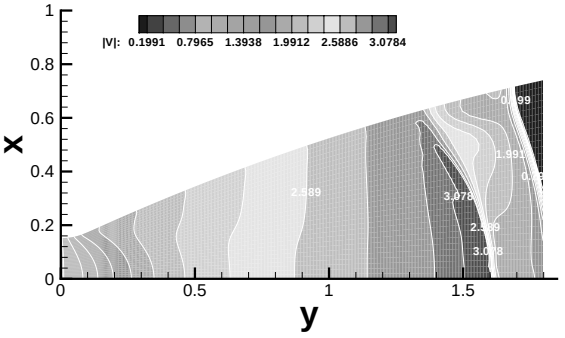


Figure 10: Comparison of axial velocity profiles along the axis of simmetry.

Velocity contours are compared for the same time



(a) Adaptive solution



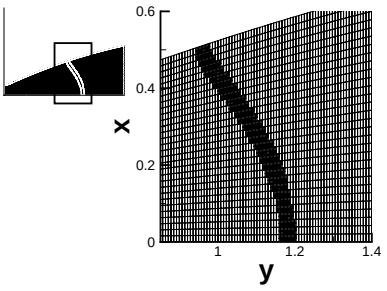
(b) Non-adaptive solution

Figure 11: Velocity contours comparison at t = 6e-4sec.

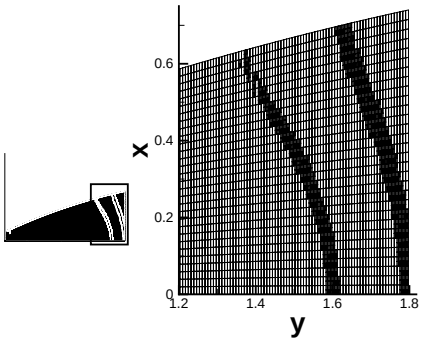
instant at Fig.11. Better resolution of the shock fronts is apparent for the adaptive solution. Although along the axis of simmetry the velocity contours are rather similar, evident differences exists close to the nozzle's wall. Finally, details of the adapted meshes at the shock fronts are presented for two time instants in Fig.12. It can be seen that at t = 3.5e-4sec the velocity gradient was not still big enough as to trigger the refinement of the mesh at the secondary shock location.

VI. CONCLUSIONS

A strategy to adaptively solve unsteady problems over unstructured finite element meshes has been presented. The numerical quality tests show that there exists a partition procedure for tetrahedral elements which keeps bounded the decrease of the mesh's quality. The refinement algorithm to apply this procedure has been described. It is noticed that efficiency and low computational costs are its main advantages while the managment of the hanging nodes may cause some problems for visualising the results. The first test case over an unstructured mesh of hexaedra proves that better use of computational resources can be achieved using the adaptive solver. Both a better resolution and a shorter simulation time give support to this conclusion. The capability of the adaptive code to handle different type of boundary conditions has been also



(a) t = 3.5e-4sec



(b) t = 6e-4sec

Figure 12: Adaptively refined meshes at the shock front locations.

proved. Future work includes testing the unsteady strategy over unstructured meshes of tetrahedra.

References

I. Babuska and W.C. Rheinboldt. Error estimates for adaptive finite element computations. *SIAM J.Numer.Anal*, 15:736–754, 1978.

N. Hannoun and V. Alexiades. Issues in adaptive mesh refinement implementation. *Electronic Journal of Differential Equations*, 15:141–151, 2007.

P.M. Knupp. Algebraic mesh quality metrics. *SIAM Journal of Scientific Computing*, April 2000.

A. Liu and B. Joe. On the shape of tetrahedra from bisection. *Mathematics of Computation*, 63(207):141–154, 1994.

D.J. Mavriplis. Unstructured mesh generation and adaptivity. Report 95-26, ICASE - NASA Langley Research Centre, April 1995.

G.A. Ríos Rodriguez, E.J. López, N.M Nigro, and M. Storti. Refinamiento adaptativo homogéneo de mallas aplicable a problemas bi- y tridimensionales, 2005.

M. Storti, N. Nigro, R. Paz, L. Dalcín, and E. López. *PETSc-FEM, A General Purpose, Parallel, Multi-Physics FEM Program.* CIMEC-CONICET-UNL, 1999-2007.

M.A. Storti, N.M. Nigro, R.R. Paz, and L.D. Dalcín. Dynamic boundary conditions in computational fluid dynamics. *Computer Methods in Applied Mechanics and Engineering*, 197:1219–1232, 2008.