HYPERSONIC FLOW SIMULATION OVER BLUNT BODIES NEAR A PLANE OF SYMMETRY WITH TRIANGULAR UNSTRUCTURED MESH

Marcos Antonio de Souza Lourenço, maslourenco@gmail.com
Emanuel Rocha Woiski, woiski@dem.feis.unesp.br
João Batista Campos Silva, jbcampos@dem.feis.unesp.br
Departamento de Engenharia Mecânica, Faculdade de Engenharia, Unesp - Ilha Solteira
Av. Brasil Centro, 56, 15385-000, Ilha Solteira - SP – Brazil

Abstract. *A numerical simulation of fluid flow over a blunt body in hypersonic regime near a plane of symmetry is taken in this work. The finite element code utilized was based on the Characteristic Based Algorithm in its explicit form with a triangular, unstructured mesh. A method of shock capturing viscosities based on the second gradient of pressure is used on the compressible inviscid flow problem. In order to validate the developed code, some test cases are solved and the effects of the mesh resolution are studied. The results obtained are compared with data available in the literature.*

Keywords*: FEM, CBS algorithm, hypersonic flow, blunt body, shock capture, unstructured mesh, triangular mesh.*

1. INTRODUCTION

The research in hypersonic aerodynamics, like that usually associated with meteorites, satellites or manned space vehicles coming from space and entering in the earth's atmosphere, aims to describe the physical phenomena involved, such as the important changes in the surrounding air and surface temperatures. Mostly due to the high costs of experimental rigs, there are a number of computational methods, schemes and algorithms in the field of hypersonic flows, as cited in Shang, (1991). For the treatment of the strong shocks present in hypersonic flows some scheme of adaptive mesh need to be implemented, in order to guarantee accuracy in the high gradient regions and save grid points in the rest of domain. Moreover, early investigations on the good performance of different shock capturing viscosity schemes in hypersonic flows with the CBS algorithm, done in (Nithiarasu et al. 1998), have shown the improvement in solutions while using the adaptive mesh procedure. Additionally they have found out that, among the four types of artificial viscosity devices analyzed, the one based on the second gradient of pressure represented the best scheme suitable for almost all type of applications. The steps for a basic adaptive mesh refinement are summarized in Zienkiewicz & Taylor (2000) as:

1. Determination of the solution from an initial coarse mesh;
2. Selection of a suitable representative scalar variable and calculation of the maximum and minimum curvatures at all nodes;
3. Calculation of the new element sizes from the values of these curvatures;
4. Calculation of the stretching ratio from the new element sizes; and
5. Remeshing of the whole domain based on the new element size.

In this paper, a finite element computational model of hypersonic flows over blunt bodies has been developed, using triangular unstructured mesh and linear triangular elements. A development of a mesh adaptive less-time consuming scheme is proposed, following the same above cited steps for the mesh refinement and using the second gradient based artificial viscosity.

For these procedures, an efficient unstructured mesh generator is essentially required. For this application, Triangle has been employed. Triangle, as presented in Shewchuck (1996), is a fast, memory-efficient, and robust C program for two-dimensional mesh generation and construction of Delaunay triangulations. It permits the inclusion of user-specified constraints on angles and user-defined triangle areas, holes and concavities, as well as deploys exact Arithmetic's to improve robustness, some desired features to adaptive mesh.

The FEM application has been developed in Python, a very high level, scripting, open-source, object oriented programming language. Since often it will be necessary to deal with a very high number of nodes and to overcome the drawback in performance of scripting languages, modules for some processes have been written in the C programming language, and imported as dynamic libraries in Python. More of the excellent features of the Python language in the scientific data manipulation can be viewed in Pletzer (2002). Thus, thanks to Python and Blender, another open source, free multiplatform program, the same application ended up integrating the mesh generator and the finite element solver as well as a GUI interface for the user to design/draw the domains, include the boundary conditions and save data. Blender can be obtained from http://www.blender.org and Python from http:/www.Python.org.

2. GOVERNING EQUATIONS

The non-viscous, non-heat-conducting Euler equations in conservative form, for two dimensional compressible flow is given by

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}_i}{\partial x_i} = 0 \tag{1}$$

where

$$\mathbf{U} = [\rho, \rho u_1, \rho u_2, \rho E]^{\mathbf{T}} \tag{2}$$

and

$$\mathbf{F}_i = [\rho u_i, \rho u_1 u_i + \delta_{1i} p, \rho u_2 u_i + \delta_{2i} p, u_i(\rho E + p)]^{\mathbf{T}} \tag{3}$$

In the above equations, ρ is the density of the fluid, u_1 and u_2 are the velocities in the directions of the axis x_1 and x_2 respectively, E is the fluid total energy and in the vector convective flux vector $\mathbf{F}$, p is the pressure. The equation of state is given as:

$$p = (\gamma - 1)\rho\left(E - \frac{u_i u_i}{2}\right) \tag{4}$$

and the ideal-gas velocity of sound related to p and ρ as

$$c^2 = \frac{\gamma p}{\rho} \tag{5}$$

3. NUMERICAL PROCEDURES

Here, first the CBS algorithm and the second order pressure gradient based shock capturing viscosity are summarized and then the mesh adaptation are treated.

3.1. The CBS algorithm

In the present paper the compressible flow of the Euler equations is solved using the CBS algorithm. The algorithm in its full form was first published in Zienkiewicz and Codina (1995) and was cited in other several works in Zienkiewicz et al. (1999) and Nithiarasu et al. (2006), and was chosen mainly due its high stability in convective regions, utilizing linear elements Keshtiban & Webster (1995). In order to solve the governing equations using the algorithm in the explicit form, the following steps are taken: in the first step, intermediate values of the conservative variables of the momentum equation are determined neglecting pressure terms. Then in the second step the density changes are determined from a Poisson equation. With the density field, the pressure values are then calculated using the gas law and in the third step, the velocities are updated from pressure. Once the pressure and the momentum fields are determined, the energy equation is solved in the fourth step. Using the characteristic Galerkin method described in Zienkiewicz and Codina (1995), the steps in explicit form can be given as:

Step 1

$$\Delta U_i^* = \Delta t\left[-\frac{\partial}{\partial x_j}\left(u_j U_i\right) + \frac{\Delta t}{2} u_k \frac{\partial}{\partial x_k}\left(\frac{\partial}{\partial x_j}\left(u_j U_i\right)\right)\right]^n \tag{6}$$

where $U_i^* = \rho u_i^*$ is the approximate flux of momentum.

Step 2

$$\Delta\rho = \left(\frac{1}{c^2}\right)^n \Delta p = -\Delta t\left[\frac{\partial U_i^n}{\partial x_i} + \theta_1\left(\frac{\partial \Delta U_i}{\partial x_i}\right)\right] \tag{7}$$

with

$$\Delta p = p^{n+1} - p^n \tag{8}$$

Step 3

$$\Delta U_i = \Delta U_i^* - \Delta t \frac{\partial p^{n+\theta_2}}{\partial x_i} - \frac{\Delta t^2}{2} u_k \frac{\partial Q_i^n}{\partial x_i} \tag{9}$$

where

$$Q_i^{n+\theta_2} = -\frac{\partial p^{n+\theta_2}}{\partial x_i} \qquad \text{and} \qquad \frac{\partial p^{n+\theta_2}}{\partial x_i} = \frac{\partial p^n}{\partial x_i} + \theta_2 \frac{\partial \Delta p}{\partial x_i} \tag{10}$$

Step 4

$$\Delta(\rho E) = -\Delta t \frac{\partial (u_j(\rho E + p))^n}{\partial x_j} + \frac{\Delta t^2}{2} u_k^n \frac{\partial}{\partial x_k}\left(\frac{\partial (u_j(\rho E + p))}{\partial x_j}\right)^n \tag{11}$$

In the Eq. (10) $\theta_2 = 0$ for the explicit case, and $\theta_1 = 0.5$ in the Eq. (7). The above solution procedure is discussed in more detail in Zienkiewicz *et al.* (1999).

3.2. Shock capturing viscosities

In the work of Nithiarasu et al. (1998) they have shown that the second gradient of pressure based methods give the better performance among four different shock capturing schemes. This method is a modified form of shock capturing viscosity developed for finite element computations (Zienkiewicz and Taylor, 2000). It is a suitable form of artificial viscosity for compressible flows and uses a pressure switch calculated from the nodal pressure values (Nithiarasu et al., 2006). Once the variables are calculated from the explicit form, they are modified by the addition of the correction from the pressure switches, which smoothes out the results. For instance, if the values in the time $n+1$ of a scalar component variable ϕ are already determined, then the new values are established as

$$\phi_s^{n+1} = \phi^{n+1} + \mu_a \Delta t \frac{\partial}{\partial x_i}\left(\frac{\partial \phi}{\partial x_i}\right) \tag{12}$$

In the above equation, μ_a is an appropriate artificial viscosity coefficient and in this work is a function of the pressure. With this new approximation the smoothen nodal values $\boldsymbol{\varphi}_s^{n+1}$ of the scalar variable ϕ can be given as

$$\boldsymbol{\varphi}_s^{n+1} = \boldsymbol{\varphi}^{n+1} + \Delta t \mathbf{M}_\mathbf{L}^{-1} \frac{C_e S_e}{\Delta t_e} (\mathbf{M} - \mathbf{M}_\mathbf{L}) \boldsymbol{\varphi}^n \tag{13}$$

where C_e is a non-dimensional coefficient, $\mathbf{M}_\mathbf{L}$ and $\mathbf{M}$ are the lumped and consistent mass matrices respectively and S_e is the pressure switch in the element and is a mean of the nodal switches S_i calculated as in Eq. (14).

$$S_i = \frac{\left|\sum_e (p_i - p_k)\right|}{\sum_e |p_i - p_k|} \tag{14}$$

3.3. Mesh adaptation

Mesh adaptation is a good idea while dealing with large systems of high-speed flow equations. For example, by means of adaptive meshing it has been possible to produce more accurate solutions to Eq. (1), while saving grid points and leading to less CPU-time consuming and storage requirements. Of the number of choices for adaptation, it has been selected the process consisting in evaluating the error associated at nodes and/or elements at any stage of the solution process, and then in refining the mesh with the requirement of an equal error distribution on nodes/elements over the same domain. When non-transients solutions are seek, as in the case of this work, then the problem of derefining certain portions of the mesh is avoided.

The measurement of the error is accomplished in accordance with Löhner et al., (1984). The following criterion needs to be satisfied in the solution process, with the density ρ being the subjected variable,

$$\max_{\Omega} \left| \rho - \hat{\rho} \right| < \varepsilon \tag{15}$$

where $\hat{\rho}$ is the density of the fluid for the approximated solution, ε is a specified tolerance, $|\cdot|$ denotes an appropriate measure and Ω is the solution domain. Several possibilities exist for the measurement of the error in Eq. (15) and are discussed and compared in more detail in the works of Löhner *et al.*, (1984) and Zienkiewicz and Taylor, (2000). Here we will concern only with the second gradient refinement (Zienkiewicz and Taylor, 2000) with interpolation errors and first-order linear triangular elements. The method will be utilized with mesh enrichment and an error measure in accordance with Peraire et al., (1987), given as

$$e = \rho - \hat{\rho} = ch^2 \frac{d^2 \rho}{dx^2} \approx ch^2 \frac{d^2 \hat{\rho}}{dx^2} \tag{16}$$

where c is a user defined constant and is assumed exact nodal values of ρ. The sought-upon equally distributed errors implies that the Eq. (17) must be satisfied over the domain.

$$h^2 \frac{d^2 \rho}{dx^2} \leq e_p \tag{17}$$

where e_p is a permissible value of the error, and can specified by the user. The adaptation proposed here is to proceed with the solution on a coarse mesh, until reach steady-state, and then proceed with the grid adaptation until the criterion of Eq. (17) be satisfied. Although the dependent variable in Eq. (17) is the density, other variables can be utilized obtaining different results. The form of choice of a particular scalar variable is better discussed in Zienkiewicz and Taylor, (2000). In the above expression, it is possible to substitute the second curvature in Eq. (18) by an expression in terms of the pressure switches S_e, of Eq. (13) of the shock capturing viscosities, and a proper scalar variable. In this manner, the second derivative calculation needs to proceed only once, when computing the shock capturing viscosities. Then, from Eq. (18),

$$h_e^2 \left(S_e \phi^n \right) \leq e_p \tag{19}$$

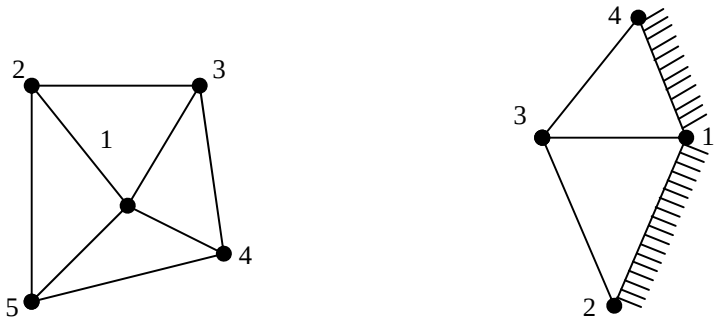
where ϕ is a scalar variable, h_e is a element specific length and the term S_e must be obtained from the nodal pressure values and is given in Zienkiewicz and Taylor, 2000, as a mean of the nodal switches S_i given as

$$S_1 = \frac{\left| 4p_1 - p_2 - p_3 - p_4 - p_5 \right|}{\left| p_1 - p_2 \right| + \left| p_1 - p_3 \right| + \left| p_1 - p_4 \right| + \left| p_1 - p_5 \right|} \tag{20}$$

for an internal node, as depicted in Fig. 1(a) and

$$S_1 = \frac{\left| 5p_1 - 2p_2 - p_3 - 2p_4 \right|}{2\left| p_1 - p_2 \right| + \left| p_1 - p_3 \right| + 2\left| p_1 - p_4 \right|} \tag{21}$$

for a node at the boundary as shown in Fig. 1(b).



(a) (b)

Figure 1. Example elements for calculation of the pressure switches: (a) An inside node; and (b) a boundary node.

As an element error indicator, in this work was chosen the maximum absolute values of the differences of the nodal corrections calculated in the RHS of Eq. (13). Substituting the term in parenthesis in Eq. (19) by this indicator and with a permissible value of the error specified, calculation of the minimum characteristic length h has been possible, as well as its use as an area constraint for the elements. As illustration, some results are presented in the next section.

3. RESULTS

In this work the tests have been developed considering two-dimensional air fluid flow around a cylinder, problem 1, and in a plane of symmetry in an axysimmetric air fluid flow around an sphere, problem 2, both at Mach 6, as depicted in Fig. 1. These two problems have been selected to illustrate the refinement process previously discussed. The results were obtained using the CBS algorithm in its explicit form. In Fig. 2 the initial very coarse mesh is depicted, constructed using the Triangle program as a callable module of Python.

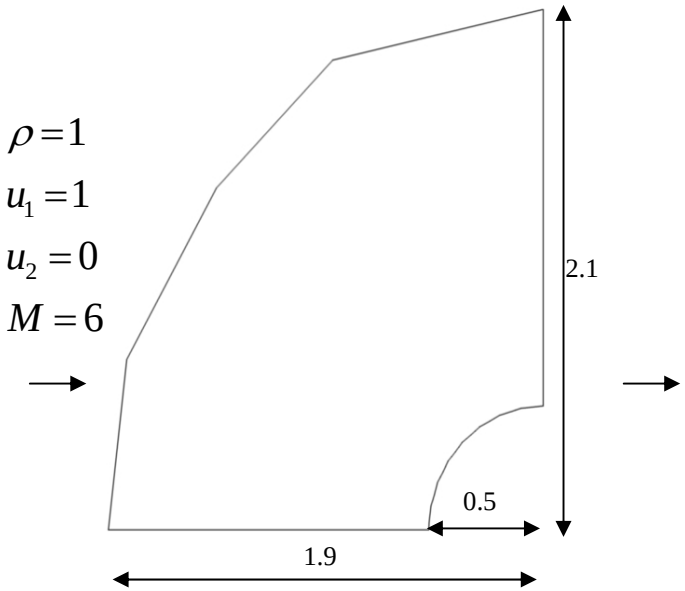


Figure 2. Solution domain and problem specification.

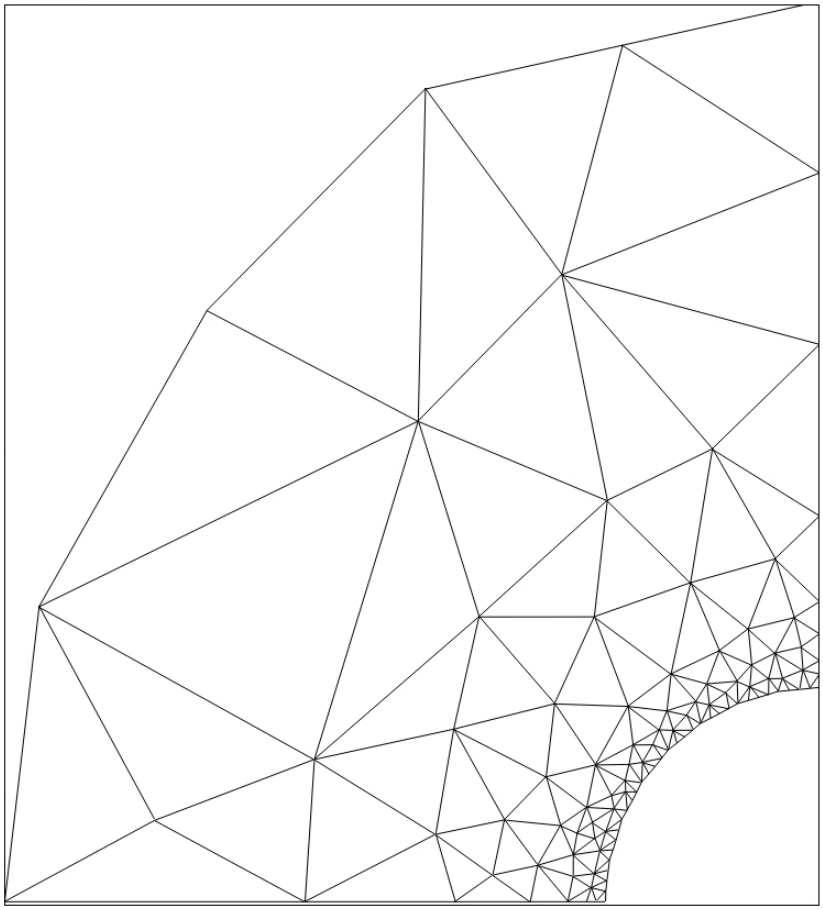
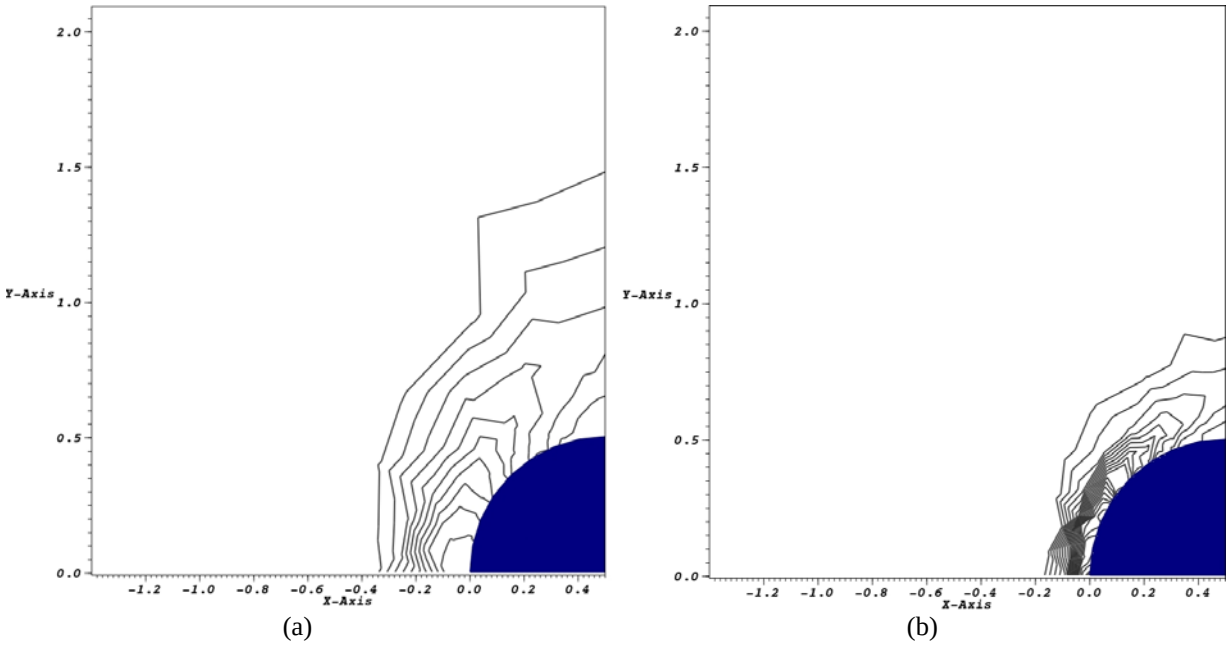
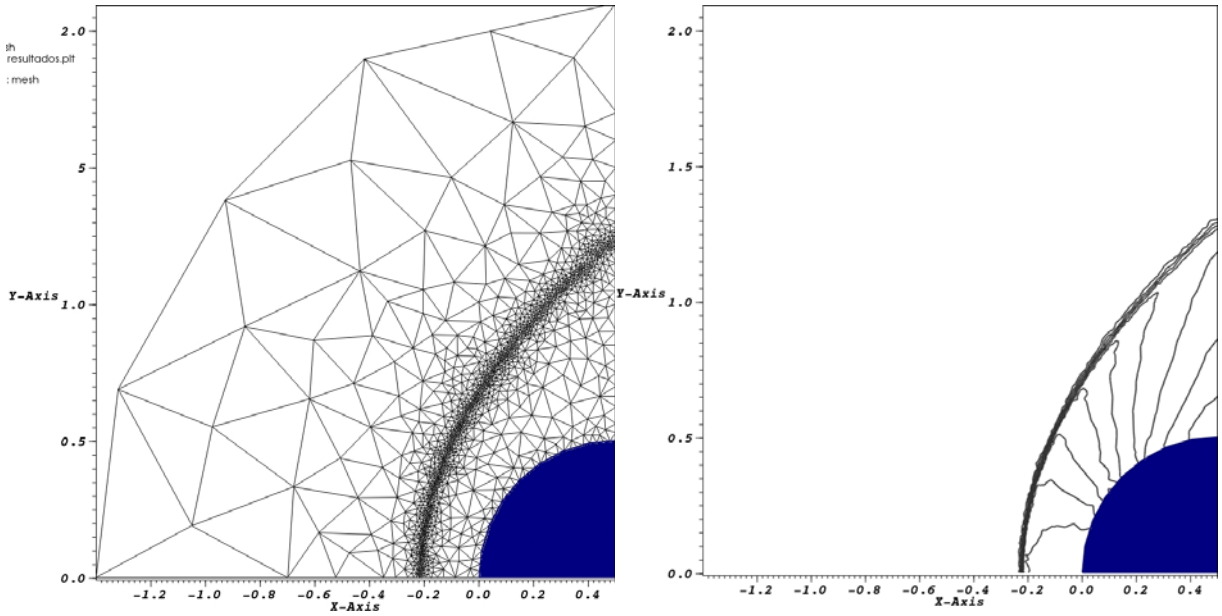


Figure 3. Initial unstructured mesh for problem of Fig. 2.



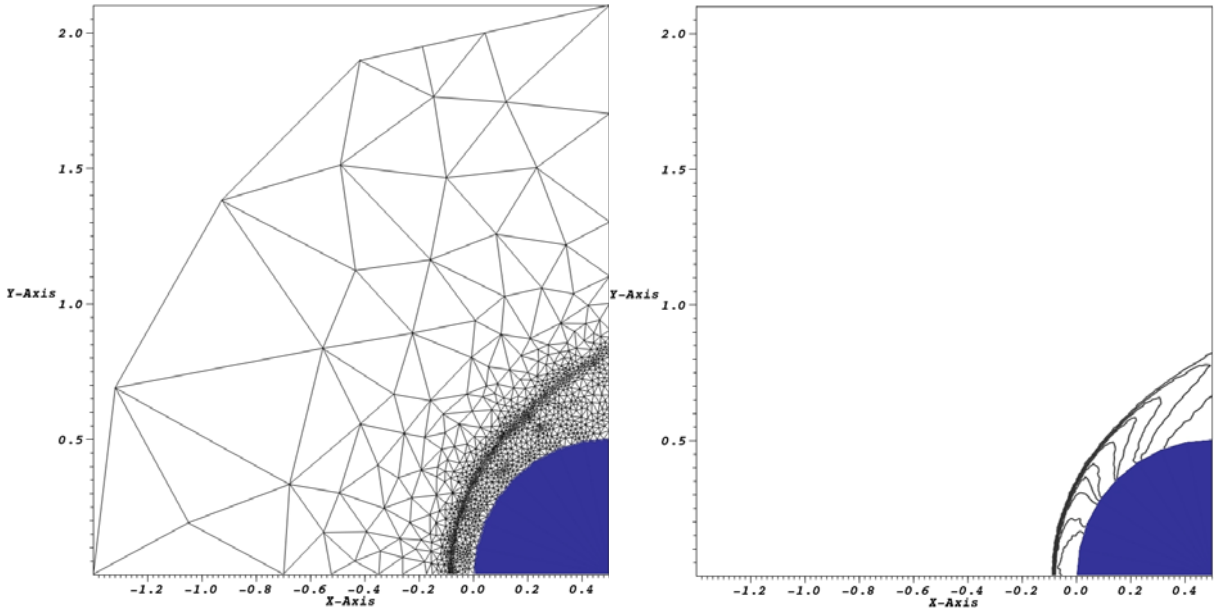
(a) (b)

Figure 4. Pressure contours determined using the mesh of Fig. 3: (a) Results for problem 1; and (b) Results for problem 2.



(a) (b)

Figure 5. Results for problem 1: (a) Adapted mesh after 27 iterations; and (b) pressure contours.



(a) (b)

Figure 6. Results for problem 2: (a) Adapted mesh after 25 iterations; and (b) Pressure contours.

From the above results it is apparently clear the gain in accuracy using the enriched meshes, when confronted with the ones of the original mesh.

4. CONCLUSIONS

An adaptive refinement strategy has been employed here for steady-state solutions for the Euler equations. From the numerical results shown in Figs. (2-6), one can easily notice the enhanced quality of the solution with the use of a mesh enrichment, moreover obtained with a minimal additional computational cost, since the element pressure switches need to be evaluated anyway for the shock capture scheme. In the present work, the multiplatform FEM mesh generator, developed with the sole combination of the open source tools Triangle, Python and Blender, has been proven very useful, flexible, accurate and extensible, with a 3D version already coming forward in the near future.

5. ACKNOWLEDGEMENT

The authors would like to acknowledge the CAPES for the scholarship to first author and FAPESP, process nº. 05/02018-8 by the financial support for the computational resources.

6. REFERENCES

Zienkiewicz, O.C. and Codina R., 1995, 'A general algorithm for compressible and incompressible flow, Part I. The split characteristic based scheme', *Int. J. Numer. Methods Fluids*, Vol. **20**, 869–885.

Zienkiewicz, O. C. e Codina, R., 1995, "Search for a general fluid mechanics algorithm", editors: D.A. Caughey and M.M. Hafez, *Frontiers of Computational Fluid Dynamics*, Wiley, New York, pp. 101–113.

Zienkiewicz, O. C., Nithiarasu, P., Codina, R., Vazquez, M. e Ortiz, P., 1999, "The Characteristic-Based-Split Procedure: An Efficient and Accurate Algorithm for Fluid Problems" *Int. J. Numer. Meth. Fluids*, Vol. 31, p. 359-392.

Nithiarasu, P., Codina, R., Zienkiewicz, O. C., 2006, "The Characteristic-Based Split (CBS) scheme - a unified approach to fluid dynamics", *International Journal for Numerical Methods in Engineering*, Vol. 66, nº 10, p. 1514-1546.

Nithiarasu, P., Zienkiewicz, O. C., Sai, B. V. K. S., Morgan, K., Codina, R., Vazquez, M., 1998, "Shock capturing viscosities for the general fluid mechanics algorithm", *Int. J. Numer. Meth. Fluids*, Vol. 28, p. 1325-1353.

Zienkiewicz, O. C. & Taylor, R. L., 2000, The Finite Element Method 5th ed. vol.3 Fluid Dynamics, Butterworth Heinemann.

Pletzer, A., "Dr. Dobb's Journal", No. 334, p. 36. March 2002, <http://ellipt2d.sourceforge.net>.

Shewchuk, J.R., 1996, "Applied Computational Geometry: Towards Geometric Engineering" (Ming C. Lin and Dinesh Manocha, editors), volume 1148 of Lecture Notes in Computer Science, pages 203-222, Springer-Verlag, Berlin.

Keshtiban I.J., Webster M.F.. "A short note on pressure-correction schemes: TGPC and CBS". <http://www-compsci.swan.ac.uk/reports/yr2005/CSR2-2005.pdf>.

Löhner, R., Morgan, K. and Zienkiewicz, O.C., 1984, "Adaptive grid refinement for the compressible Euler and Navier-Stokes equations", in *Proc. Int. Conf. Accuracy Estimates and Adaptive Refinement in Finite Element Computations*, Lisbon.

Peraire, J., Vahdati, M., Morgan, K. and Zienkiewicz, O.C., 1987, "Adaptive remeshing for compressible flow computations", J. Comp. Phy., 72, pp. 449-66.

Shang, J.S., 1991, "Numerical Simulation of Hypersonic Flows". In T. K. S. Murthy, editor, Computational Methods in Hypersonic Aerodynamics, pages 1-28. Klumer Academic Publishers.

7. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.