Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XXXI, págs. 123-135 (artículo completo)
Alberto Cardona, Paul H. Kohan, Ricardo D. Quinteros, Mario A. Storti (Eds.)
Salta, Argentina, 13-16 Noviembre 2012

NUMERICAL SIMULATION OF INTERNAL FLOW TRANSITION IN A ROCKET NOZZLE.

Luciano Garell , Gustavo R. Ríos Rodríguez , Rodrigo R. Paz and Mario A. Storti

Centro Internacional de Métodos Computacionales en Ingeniería (CIMEC), (INTEC-CONICET-UNL)
Güemes 3450, (S3000GLN) Santa Fe, Argentina.

Keywords: Finite element method, flow transition, rocket nozzle, h-adaptive refinement, parallel computing.

Abstract. This work is a first step in the understanding of the interaction process between internal shock waves and the flow transition inside of a rocket nozzle during the start-up process or when it is operated under strongly over-expanded conditions. The interaction process produces a transition in the flow pattern, which in many cases generates side loads in the nozzle due to a change in the pressure distribution on the wall, being harmful for the rocket integrity. In order to understand the process a numerical simulation is carried out by solving the three dimensional Euler equations. With this tridimensional model the computational cost significantly increases, therefore parallel processing is required. Also, an unsteady h-adaptive refinement strategy is used jointly with a SUPG (Streamline Upwind Petrov-Galerkin) and discontinuity capturing scheme, both to keep the mesh size bounded and to sharply resolve the shock wave pattern. The adapted mesh is non-conforming and a smooth size transition among neighbour elements with different levels of refinement is enforced by means of refinement constraints. Computed average wall pressure distributions for various nozzle pressure ratios and at different time instants are compared. The simulations are carried out using the PETSc-FEM software.

1 INTRODUCTION

Nozzles with high area ratio are used in the main space launchers (Space Shuttle Main Engine, Ariane 5). The nozzle contour is often designed according to the theory proposed by Rao (Rao, 1996) that results in a TOP (Thrust Optimized Parabolic or Parabolic Bell Nozzle) nozzle, which has some advantages compared to the traditional conical shapes. These advantages are the smaller length, lower weight, as well as the reduction in energy losses in the expansion of gases (Sutton and Biblarz, 2001; Oates, 1997; Mattingly and Ohain, 2006; Tuner, 2006). These engines must work in conditions ranging from sea level to orbital altitude but an efficient operation is only reached at high altitude, so in many cases they operate under strongly over-expanded condition. The transition from an over-expanded condition to an ideal expansion produces a change in the flow pattern inside the nozzle, which often produces side loads due to a change in the pressure distribution on the wall, being harmful for the rocket integrity.

The flow pattern that develops inside the nozzle is perfectly suited for being adaptively solved because flow features of interest usually develop in very thin regions compared to some characteristic length. The adaptation of the mesh allows to reduce the computational effort required to solve the flow problem since elements are introduced only where and when they are needed. In this work, an unsteady h-refinement adaptation algorithm is used since it is regarded as a good method for transient problems (Löhner and Baum, 1992).

Unsteady problems require to refine and derefine the mesh a great number of times in order to follow discontinuities in their travel throughout the computational domain, so the adaptation of the mesh should demand a small fraction of time compared to the solution time. Besides, the adaptation scheme ought to minimize the geometrical quality degradation of the mesh. To address these issues, an h-refinement strategy for linear tetrahedra and hexaedra based uniquely on the regular $1 : 8$ element subdivision is considered. No transition elements are used to match zones with different levels of refinement so that hanging nodes on edges and faces appear and the refined mesh is non-conforming. Extension to 3-D meshes of the well known 1-irregular vertex refinement constraint is used to ensure that neighbour elements in the mesh have a similar size. This avoids the effort of considering and managing a great number of transition templates or transition elements (Staten, 1996; Löhner and Baum, 1992; Remacle et al., 2002) for eliminating of the hanging nodes and keeps bounded the quality degradation of the mesh (Ríos Rodriguez et al., 2005, 2009). As a consequence, the refinement algorithm used in this work results simple, scaling almost linearly with the number of refined elements (Ríos Rodriguez et al., 2011).

We mention that the solution procedure is partially parallelized. This means that the adaptation of the mesh is sequentially performed while the solution of the flow equations is computed in parallel on a Beowulf cluster (Storti, 2005-2010) using the PETSc-FEM software (Storti et al., 1999-2010; Sonzogni et al., 2002). This latter is a multi physics OOP code which uses both a finite element SUPG formulation to stabilize the advective terms of the equations and shock capturing term for the treatment of non-linear instabilities in the neighbourhood of shocks (Brooks and Hughes, 1980, 1982b; Hughes and Mallet, 1986a,b; Tezduyar and Senga, 2006b). Since continuous finite element functions are considered, constraints to the solution field at irregular vertices are applied. A Mach number-based gradient indicator is used to tag the cells of the mesh that need to be refined or coarsened. The adaptation of the mesh and the solution computation are coupled through an interface which automates the procedure. In this manner,

the boundary conditions for the problem are specified on the starting mesh and are automatically updated later. Also, a projected state on the adapted mesh is given in order to resume the flow computation.

2 NUMERICAL MODEL

The nozzle under study corresponds to the subscale S1 parabolic contour, which was designed with the geometrical contour of the Vulcain nozzle in order to study the side load phenomena (Ostlund, 2002). Table(1) shows its principal characteristics

Table 1: Subscale S1 nozzle characteristics:

Area ratio(ε)	20
Nozzle length (L_n)	350 [mm]
Throat diameter (D_t)	67.1 [mm]
Nozzle exit diameter (D_e)	300 [mm]
Design feeding pressure (P_0)	5 [Mpa]

The fluid domain is discretized with tetrahedral elements and a linear interpolation of the variables is used. The gasdynamics Euler equations are solved with a consistent Streamline Upwind Petrov-Galerkin (SUPG) stabilization technique (Brooks and Hughes, 1982a; Franca et al., 1992; Tezduyar and Senga, 2006a) together with the shock-capturing (SC) method (Tezduyar and Senga, 2004).

A time-varying boundary condition is imposed at the inlet of the nozzle

Table 2: Stagnation values used for the combustion chamber

P_0	ρ_0	T_0
$\{0.44; 2.1\}$ [MPa]	$\{3.8; 20.4\}$ [kg/m^3]	300 [K]

The stagnation pressure is linearly increased according to Table(2) from $t_0 = 0$ [sec] to $t_f = 1 \cdot 10^{-2}$ [sec] during 4000 time steps while ρ_0 is computed from the state equation since T_0 is constant. The time step size is set equal to $Dt = 2.5 \cdot 10^{-6}$ [sec] and the Courant Number (Co) ranges between 1.2 and 1.5, being the stability criterion assured for an implicit θ-method time integrator. An absorbent / dynamic boundary condition is set at the outlet with non-linear constraints imposed via Lagrange Multipliers or a Penalty Method (Paz et al., 2010). This boundary condition allows to reduce the extension of the computational domain therefore decreasing the computational cost. Finally, a slip condition is applied at the wall of the nozzle.

Numerical simulations assume the nozzle is operated in over-expanded conditions because the maximum feeding pressure used in the simulations (1.85 [Mpa]) is lower than the design feeding pressure (5 [Mpa]). As a consequence, an internal shock wave develops. The shape and position of the internal shock changes as a function of the feeding pressure.

Figure(1) describes the flow transition as well as the pressure ratio P_{wall}/P_0 in the nozzle , where P_{wall} is the pressure at the wall and P_0 is the total pressure at the combustion chamber. At low pressure ratios (P_0/P_{out}), the flow is detached from the wall and the outlet gases go through the centerline of the nozzle, producing a recirculation zone near the wall.

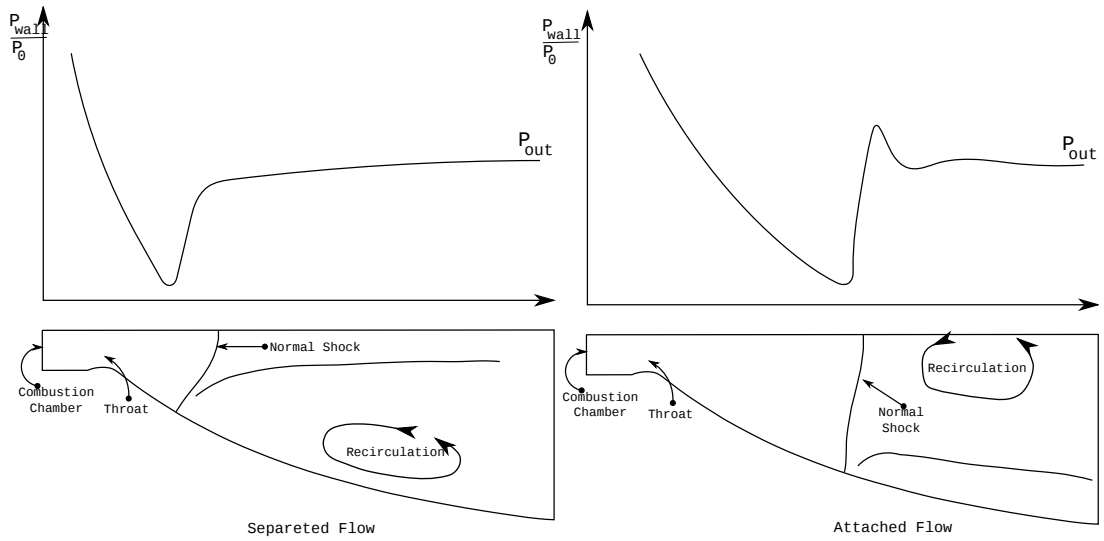


Figure 1: Schematic diagram of the flow transition during startup.

When the pressure ratio increases, the flow pattern changes and the strong shock forces the outlet gases to attach to the nozzle wall. Meanwhile, the recirculation zone places at the centerline of the nozzle and the pressure there is lower than P_{out}. In this work both over-expanded conditions will be analyzed and the pressure distributions on the wall and the axis will be reproduced.

3 MESH ADAPTATION

3.1 Data structure for mesh representation

The data structure chosen to represent the mesh is a key aspect to be considered in the design of the adaptation code, since it has a great influence on both the execution time and the memory consumed. The code used in this work is written in the C++ language and makes extensive use of the containers, iterators and algorithms of the Standard Template Library (Sil, 1993-2012) and the Boost programming libraries (Boo, 1998-2012). These latter are specifically designed to work well with the C++ Standard Library.

A mesh is represented by geometrical entities and its adjacencies. Each entity is uniquely identified by an iD number and can be of the following classes: elements, faces, edges and vertices. So we naturally define the **ele**, **face** and **edge** C++ classes to represent the corresponding mesh entities. Instances of these classes are stored in STL vectors, since this provides random access to each entity in the container, which is a requirement of the refinement / coarsening algorithms. Vertices are stored in a 2-D dynamic-size array from the Boost Multidimensional array library. Boost multi arrays are a more efficient way to represent n-dimensional dynamic-size arrays than an equivalent implementation provided by nested vectors of the STL library.

For the mesh representation we use the following set of *downward* adjacencies: **faces-of-element**, **edges-of-element**, **vertices-of-element**, **edges-of-face** and **vertices-of-edge**. Even when we could have chosen not to use the edges-of-element set, we didn't because of the great number of times that we need to retrieve this information and the implicit cost that requires the second order (indirect) access of retrieving the faces-of-an-element, then the edges-of-a-face and finally computing the union of this final set of edges. Similar reasons lead us to manage the vertices-of-element adjacencies. This conclusion is based on concepts and analysis developed in

(Remacle et al.). Since the number of downward adjacencies for an entity of a given dimension is known beforehand (i.e. a tetrahedra has four faces, six edges and four vertices), fixed size arrays can be used to store them. Boost bidimensional array containers are used to implement the downward adjacencies for all the entities of the same dimensionality. This allows random access and enables to preserve the *local order* given by downward entity templates. This local order would be lost if a second order access were used since the union operation does not preserve it. Templates allow to manage the concept of "orientation" to ensure uniqueness of the entity representation, that is they describe local relationships between downward entities of a same entity type.

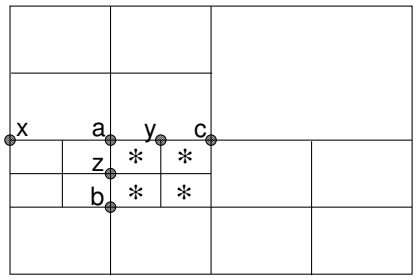
On the other hand, *upward* adjacencies are also requiered in different stages of the algorithm. In this case, the number of entities of a lower dimension that "share" an entity is not known beforehand. This means that it is not efficient to use fixed-size containers. Also, the order of the upward adjacencies has no importance, so we use unordered multimap containers from the Boost Unordered library. These are hashed-type multiple associative containers which allow to access data with constant complexity on average. The upward adjacencies for a given entity are retrieved by iterating through the range in the container whose keys are the same as the entity iD.

For unsteady problems the adaptation of the mesh requires to insert as well as to erase entities in the data structure. The entities that represent the current mesh are defined as "actives" and their iDs are stored in unordered sets. If an entity is refined, it is kept in the data structure, but its iD is replaced by those of its children in the set of active entities. In this case it is said that the refined entity is "deactivated". If later on the child entities are unrefined, their iDs are erased from the active entities set and the parent entity iD is "activated" again. Unordered sets are used because it is required to erase the iDs at any position in the container while the order is not important. Unordered sets allow to insert and erase elements with constant complexity on average.

Since vectors are used to store the entities, it is not efficient to explicitly erase them from the container because this takes linear time on the size of the vector unless this is done at the end (which is not the common situation in the unrefinement of the mesh). We consider that if an entity is unrefined, its iD is inserted in a list of "free indexs" for that kind of entities. This means the iD is available to be reused by a new entity of the same kind. If new entities are created during refinement, their iDs are taken from this free indexs list until it is left empty. If the number of requiered iDs is more than the ones affordable in the free indexs list, the remaining iDs are sequentially created starting from the last (biggest) iD.

A data object of the element class stores the iDs of the elements that appear in the mesh because of its refinement (children), the iD of the element from which derives (parent), the refinement level where it belongs, the iDs of the vertices that defines it (following a local order) and a property label that is used to associate the element to any particular property defined by the user (geometrical, physical or both). This is useful to handle the boundary conditions for the adapted mesh.

Objects of the face and edge classes store similar data than elements, as well as the number of elements that share them. This information is needed when coarsening to decide whether an entity has or not to be unrefined. If the all the elements that share a face (edge) are unrefined, then the parent face (edge) is unrefined (if it already has a parent) by resetting the iDs of its children to a null value and the child entities are removed from the data structure for the current mesh by inserting their iDs in the set of free iDs for that type of entities. If not, the number of elements that share the entity is dimished acordingly. Finally, the parent entity is "activated"

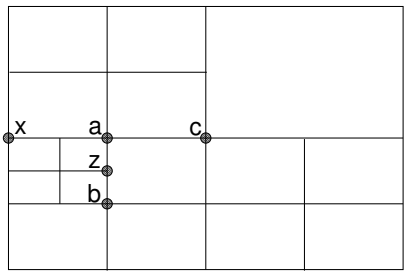


iD nodes mid. node parent n_sh

e1 ----> {a,z} -----> {null} ----> {e3} ----> 2
e2 ----> {a,y} -----> {null} ----> {e4} ----> 1
e3 ----> {a,b} -----> {z} ----> {null} ----> 2
e4 ----> {a,c} -----> {y} ----> {e5} ----> 2
e5 ----> {x,c} -----> {a} ----> {null} ----> 2

active_edges={e1,e2,e4,....}

(a) Elements to be unrefined are marked with an asterisk.



iD nodes mid. node parent n_sh

e1 ----> {a,z} -----> {null} ----> {e3} ----> 1
e2 ----> {null} ----> {null} ----> {null} ----> null
e3 ----> {a,b} -----> {z} ----> {null} ----> 2
e4 ----> {a,c} -----> {null} ----> {e5} ----> 2
e5 ----> {x,c} -----> {a} ----> {null} ----> 2

active_edges={e1,e3,e4,....}
free_iDs={e2}

(b) Edge e2 is *unrefined* because all the elements that share it are unrefined. Edge e3 is *activated*.

Figure 2: Unrefinement sequence.

(see Fig.2). An edge class object also stores the iD of the vertex that appears in its middle, which is required to apply the 1-irregular vertex refinement / coarsening constraints.

3.2 Refinement schemes

The partitioning of the elements considered in this work is the same as used in (Ríos Rodriguez et al., 2009, 2011), (i.e. it only considers regular 1:4 and 1:8 subdivision for 2-D and 3-D elements, correspondingly). Therein it is shown that refinement through the shortest diagonal of the inner octaedron shows a good trade-off between the required computational effort and the geometrical quality of the resulting elements for tetrahedra elements.

3.3 Refinement and unrefinement constraints

A smooth size change among neighbour elements in the mesh is desired because of its influence in the condition number of the stiffness matrix of the finite element method formulation (Shewchuck, 2002). In this sense, we asume the 1-irregular vertex refinement constraint (Babuska and Rheinboldt, 1978; Greaves, 2004; Popinet, 2003; Remacle et al., 2002; Ríos Rodriguez et al., 2009, 2011). The rule states that *no more than one hanging node should be shared among neighbour elements through the common edge to which the hanging node belongs*. This means that if the element marked with an asterisk in Fig.(3.a) is to be refined, it is required that elements marked with a cross be refined first. But this in turns implies that the element marked with a circle be refined formerly. As it is noted, the addition of elements to be refined because of applying this constraint makes recursive the refinement routine.

In 3-D the neighbourhood among elements through edges and faces as well as the refinement of *orphan* edges on triangular faces have to be considered (Ríos Rodriguez et al., 2011). An orphan edge is that which is not obtained by the refinement of another edge.

Elements are selected to be unrefined based on the chosen criteria. Then it is necessary

(a) Refinement case. (b) Unrefinement case.

Figure 3: Description of the 1-irregular vertex constraint.

to guarantee that the unrefinement of these elements circumscribes to the 1-irregular vertex constraint. As can be seen in Fig.(3.b), if any of the edges for the element selected to be unrefined (marked with an asterisk) has an irregular vertex, then that element should be excluded from the list of elements to be unrefined. If not, the adapted mesh would have an edge ($\overline{ab}$ in Fig.3.b) with two irregular vertices. In conclusion, the 1-irregular vertex constraint may exclude elements that were initially marked to be unrefined while may include some elements to be refined. This bias the adaptation algorithm towards refinement rather than derefinement, which is sensible on the grounds of solution accuracy.

Finally, the coarsening algorithm considers that an element can be unrefined if only all its brothers are also marked to be unrefined. In Fig.(3.b) it is seen that because one of the brothers cannot be unrefined, none of them can.

3.4 Adaptation algorithm

The adaptive solution of the problem begins by solving the Euler equations on a conforming mesh, hereafter called the *base* mesh. After a fixed number of time steps ($nsteps$), the regions of the base mesh that need to be refined are selected according to the following criterion, which is based on the magnitude of the Mach number gradient (M) computed in an element-wise fashion. All those elements whose gradients magnitude times theirs size are equal to or greater than a percentage of the maximum corresponding value for all the elements in the mesh are marked to be refined

$$c_1 \leq \frac{\parallel \nabla_i M \parallel \cdot h_i}{\max_i(\parallel \nabla_i M \parallel \cdot h_i)} \tag{1}$$

where c_1 is a constant set beforehand by the user, h_i is a measure of the element size (e.g. the length of the longest edge for the element) and $\parallel \nabla_i M \parallel$ is the magnitude of the Mach gradient computed for the element. The accurate choice of c_1 mostly depends on the user's experience and in this work a value of $c_1 = 0.3$ is chosen.

A succession of nested non-conforming meshes is generated by selecting the elements to be refined and applying the refinement schemes and constraints described in the previous section until a maximum level of refinement is attained. A limit on the number of refinement levels is considered because there is no stopping criterion if discontinuities exists in the solution and Eq.(1) is used to select the elements to be refined.

As the base mesh is refined, the last state computed by the solver is linearly interpolated and the boundary conditions are updated by inheritance of the property flags from parent to child entities. When the maximum level of refinement is attained the interpolated state is used as the

initial condition to resume flow computation. If it is not requiered to refine the current mesh, flow computation is resumed from the last computed state. The strategy does not consider the unrefinement of the base mesh.

After the solution is advanced $nsteps$ time steps, the selection criterion given by Eq.(1) is applied to the last computed solution and elements are marked to be refined again. Also, elements are marked to be unrefined if

$$c_2 \geq \frac{\| \nabla_i M \| \cdot h_i}{\max_i(\| \nabla_i M \| \cdot h_i)} \tag{2}$$

where c_2 must be less than c_1. A value of $c_2 = 0.05$ was used in all simulations. The adaptation strategy assumes that all those elements which satisfy Eq.(2) should be unrefined as much as possible up to the base mesh level, as long as the 1-irregular vertex unrefinement constraint is already satisfied. An unrefinement loop is considered then. If an element can effectively be coarsened, it is replaced by its parents in the current mesh. If it cannot, it is considered to be possibly unrefined in the next iteration of the unrefinement loop. If it is not possible to go further with the coarsening, the loop is interrumpted. After that, the refinement loop begins as described for the first adaptation step. If no elements are refined or coarsened, the flow computation resumes from the last computed state. The value for the mesh adaptation frequency $nstep$ is set equal to 10 time steps. Figure(4) shows the proportional cost of the adaptive refinement as function of the total time consumed to solve 10 time steps. Also is plotted the numbers of D.O.F after each refinement process.

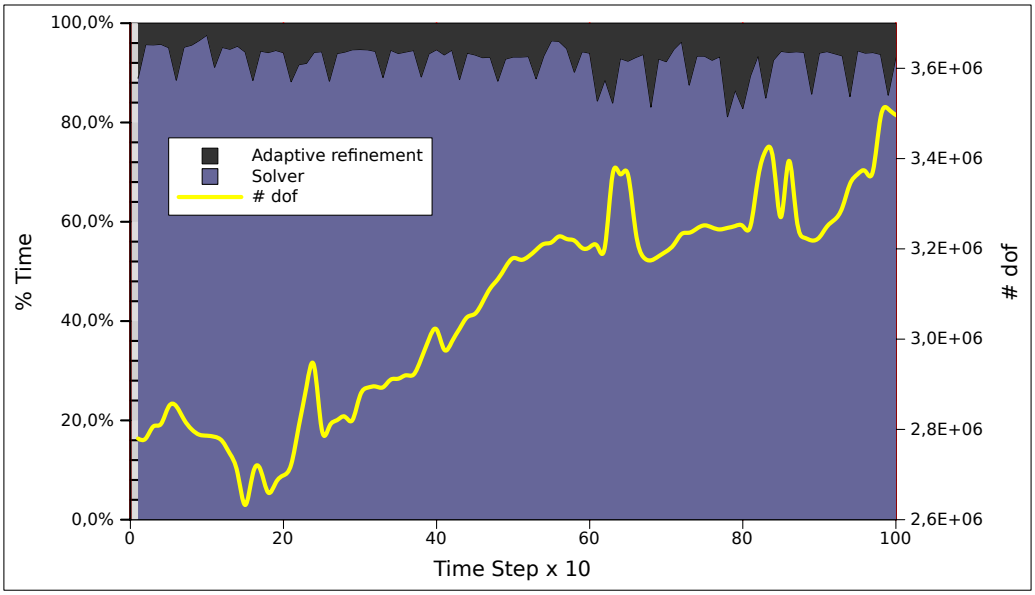


Figure 4: Proportional cost of solver and adaptive refinement.

The proper choice of the adaptation frequency depends on different variables. Several authors (Waltz, 2004; Ripley et al., 2004; Remacle et al., 2002) find in practice that the adaptation of the mesh takes just a small fraction of the overall simulation time (approximately 5 per cent). This result was confirmed with the current strategy in a previous work (Ríos Rodriguez et al., 2011). If the time required by the adaptation of the mesh were found to be a greater percentage of the overall simulation time, then a lower updating frequency should be chosen. However, in this latter case a bigger cost would be transferred to the flow computation stage since the refined

regions of the mesh would need to be "wider" to ensure that discontinuities will be kept inside them until the mesh is updated again. Choosing a higher frequency to adapt the mesh enables to use narrower refined regions around discontinuities and the fluid flow problem is less expensive to solve.

4 NUMERICAL RESULTS

To understand the transition phenomena of the flow inside of the rocket nozzle, the problems is started from a steady condition with $P_0 = 0.44$ [MPa], $\rho_0 = 3.8$ [kg/m^3] and $T_0 = 300$ [K] computed on the base mesh. After the steady state is reached the mesh is refined with a Mach number-based gradient indicator and then it is again converged to a steady state condition.

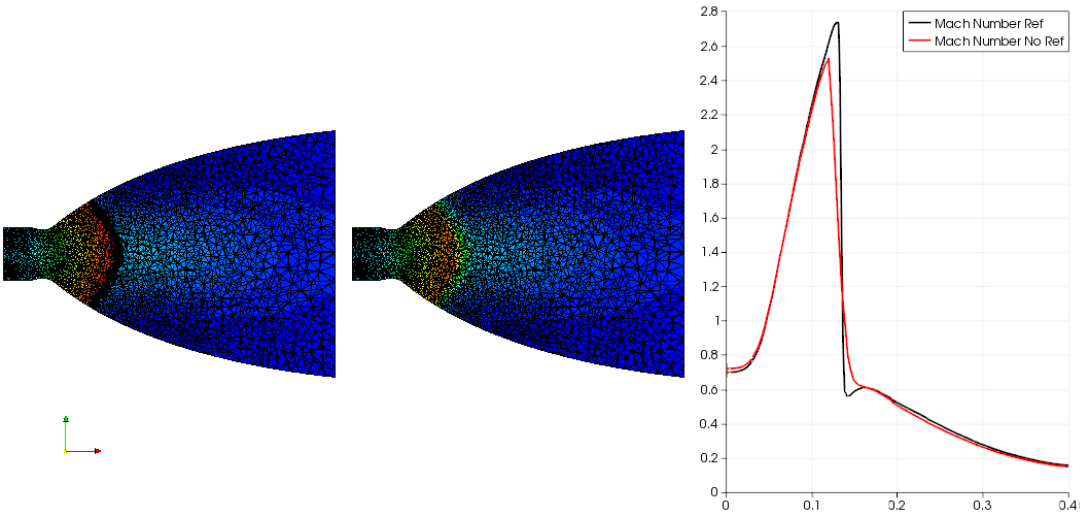


Figure 5: Refinement with a Mach number-based gradient indicator.

Figure (5) shows the improvement in the sharpness of the normal shock due to the refinement of this zone. The area to be refined can be extended or contracted by changing the parameters c_1, c_2 and the numbers of refinement levels. In this problem two refinement levels are used with $c_1 = 0.3$ and $c_2 = 0.05$.

From this condition the pressure is linearly increased in time, from $t_0 = 0$ [sec] to $t_f = 1 \cdot 10^{-2}$ [sec]. We required 4000 time steps and a time step size $\Delta = 2.5 \cdot 10^{-6}$ [sec]. As a consequence, the shock wave starts to move towards the outlet and the pressure distribution on the nozzle wall and along the axis modify, changing the flow pattern.

At low pressure ratios ($P_0 = 0.44$ [MPa]), the flow is detached from the wall after the shock and the outlet gases go through the center of the nozzle, just as explained in Fig.(1). Near the wall, the pressure rapidly reaches the ambient condition and a recirculation zone appears, as it is shown in Fig.(6).

For a pressure value of $P_0 = 1.1$ [MPa] the flow pattern starts to change and now the pressure in the axis, right after the shock is lower than the ambient pressure and a vortex is produced, as shown in Fig.(7). At the nozzle wall there appears a pressure peak after the shock, thus producing the outlet gases to go attached to the wall.

Finally, for $P_0 = 2.1$ [MPa] the flow pattern has fully transitioned. The pressure in the axis after the shock is lower than the ambient pressure producing a large recirculation region, as shown in Fig.(8). On the nozzle wall there is a strong pressure peak after the shock so that the

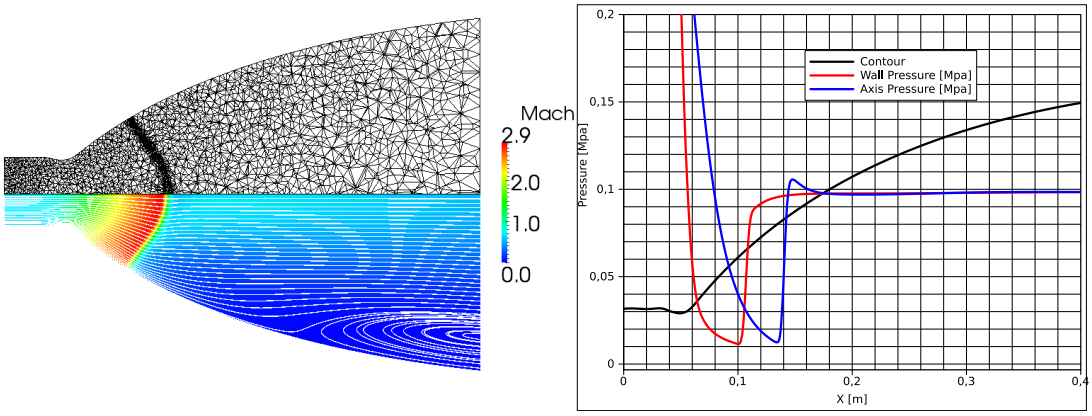


Figure 6: Pressure distributions at the wall and the axis of the nozzle ($P_0 = 0.44$ [MPa]).

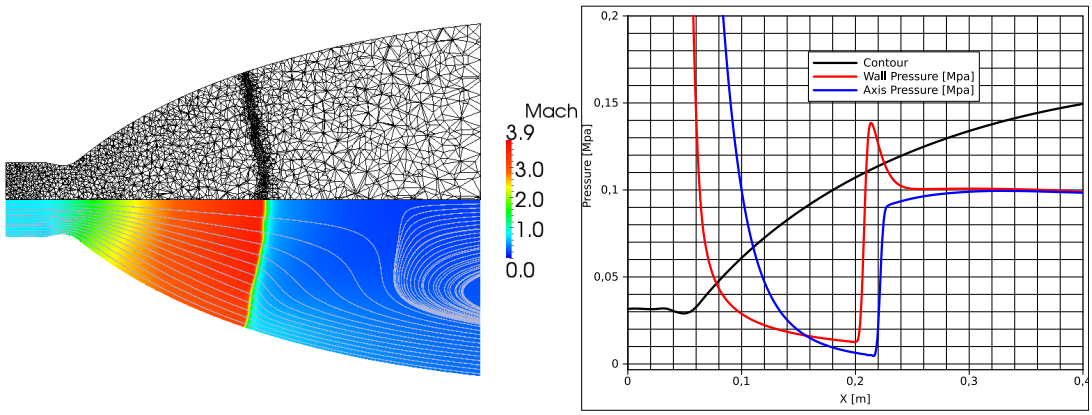


Figure 7: Pressure distributions at the wall and the axis of the nozzle ($P_0 = 1.1$ [MPa]).

outlet gases go attached to the wall.

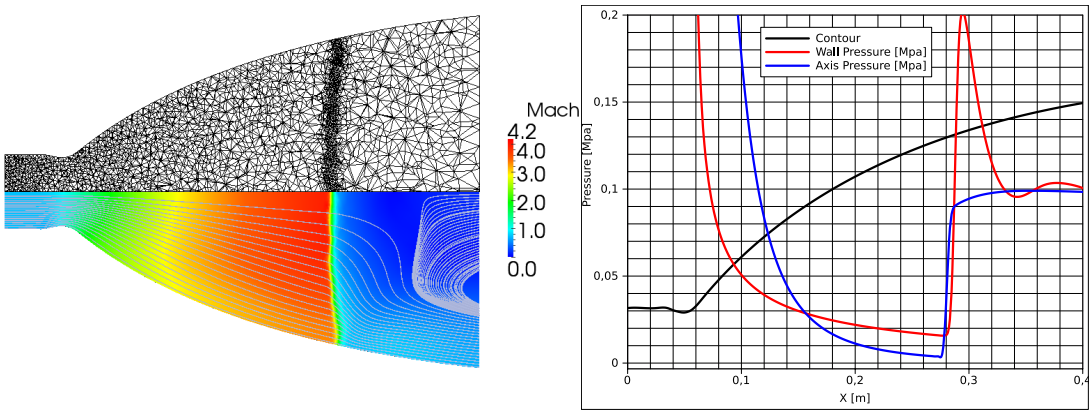


Figure 8: Pressure distributions at the wall and the axis of the nozzle ($P_0 = 2.1$ [MPa]).

The flow transition can also be characterized by the relationship between the ratio $P_{t,i}/P_{out}$, where $P_{t,i}$ is the total pressure ahead of the shock wave for a time instant i, and the nozzle pressure ratio P_0/P_{out}. In the work of Moriñigo and Salvá (2008), the internal flow of the subscale optimized J2-S nozzle is characterized analyzing the evolution of this relationship.

Figure (9) shows the evolution of the total pressure ahead of the shock wave for the subscale S1 nozzle. When pressure $P_{t,i}$ is higher than the outlet pressure, the principal flow is detached from the wall and goes through the center of the nozzle. The onset of the flow transition occurs when $P_0/P_{out} \approx 9.2$ and $P_{t,i} \approx P_{out}$. Thereafter $P_{t,i} \leq P_{out}$ and the recirculation zone appears.

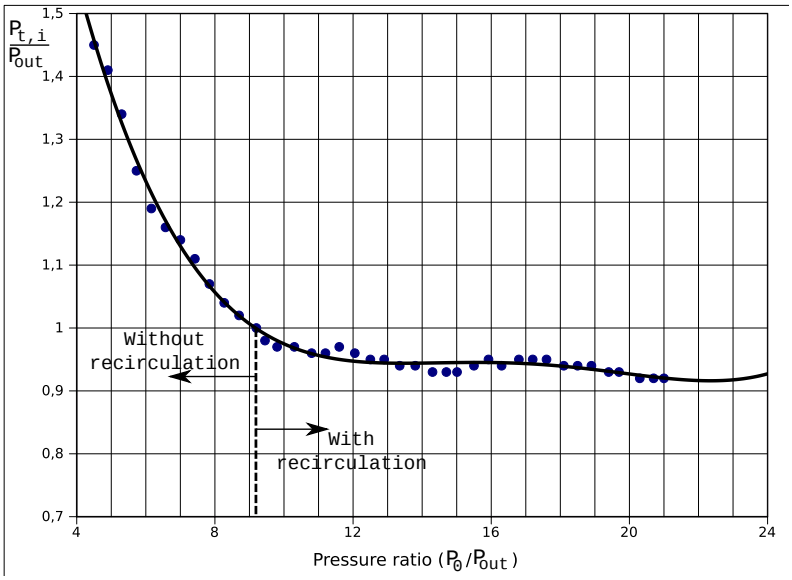


Figure 9: Total to outlet pressure ratio ($P_{t,i}/P_{out}$) at the central axis, just ahead of the shock wave, for a time varying P_0/P_{out}.

5 CONCLUSIONS

As already mentioned, this work is a first step on the understanding of the interaction process between internal shock waves and the flow transition inside of a rocket nozzle during the start-up of the engine or when it is operated under strongly over-expanded conditions. The numerical simulations were carried out using the PETSc-FEM software, which uses both a finite element SUPG formulation to stabilize the advective terms of the equations and shock capturing methods for the treatment of non-linear instabilities in the neighbourhood of shocks. The adaptive refinement strategy helped both to sharply resolve the shock wave pattern and to keep the computational costs as low as possible. An equivalent uniform grid just as fine as the adaptive one would have incurred in very high computational costs, considering that two levels of refinement were used.

The pressure along the wall and the axis of the nozzle were analyzed for different pressure ratios, wherewith the change in the internal flow pattern was clearly identified. Also, the streamlines were plotted in order to characterize the internal flow. This change in the flow pattern during over-expanded conditions has been reported as the origin of side loads.

As future work this study will be extended to a viscous 3-D case in order to analyze the interaction of the boundary layer with the shock waves. Also, lateral loads will be obtained and compared to some reference data.

ACKNOWLEDGEMENTS

This work has received financial support from

- Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET, Argentina, PIP 5271/05),
- Universidad Nacional del Litoral (UNL, Argentina, grants CAI+D 2009-65/334, CAI+D-2009-III-4-2),
- Agencia Nacional de Promoción Científica y Tecnológica (ANPCyT, Argentina, grants PICT-1506/2006, PICT-1141/2007, PICT-0270/2008).

The authors made extensive use of *Free Software* as GNU/Linux OS, GCC/G++ compilers, Octave, Paraview, SGI STL Library, Boost Library. In addition, many ideas from these packages have been inspiring to them.

REFERENCES

I. Babuska and W.C. Rheinboldt. Error estimates for adaptive finite element computations. *SIAM Journal on Numerical Analysis*, 15(4):736–754, 1978.

Boost C++ Libraries. Boost Software, http://www.boost.org, 1998-2012.

A.N. Brooks and T.J.R. Hughes. Streamline Upwind/Petrov Galerkin methods for advection dominated flows. In *Third Internat. Conf. of Finite Element Methods in Fluid Flow*, pages 283–292, Banff, Canada, June 1980.

A.N. Brooks and T.J.R. Hughes. Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 32:199–259, 1982a.

A.N. Brooks and T.J.R. Hughes. Streamline Upwind/Petrov Galerkin formulations for convective dominated flows with particular emphasis on the imcompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Engrg.*, 32:199–259, 1982b.

L.P. Franca, S.L. Frey, and T.J.R. Hughes. Stabilized finite element methods: I. application to the advective-diffusive. *Computer Methods in Applied Mechanics and Engineering*, 95: 253–276, 1992.

Deborah Greaves. A quadtree adaptive method for simulating fluid flows with moving interfaces. *J. Comput. Phys.*, 194(1):35–56, 2004. ISSN 0021-9991.

T.J.R. Hughes and M. Mallet. A new finite element formulation for computational fluid dynamics: Iii. The generalized streamline operator for multidimensional advective-diffusive systems. *Comput. Methods Appl. Mech. Engrg.*, 58(3):305–328, 1986a.

T.J.R. Hughes and M. Mallet. A new finite element formulation for computational fluid dynamics: Iv. A discontinuity-capturing operator for multidimensional advective-diffusive systems. *Comput. Methods Appl. Mech. Engrg.*, 58(3):329–336, 1986b.

R. Löhner and J.D. Baum. Adaptive h-refinement on 3D unstructured grids for transient problems. *Int. Journal for Num. Methods in Fluids*, 14:1407–1419, 1992.

J. Mattingly and Hans Von Ohain. *Elements of propulsion: gas turbines and rockets.* AIAA, 2nd edition, 2006.

J. Moriñigo and J. Salvá. Numerical study of the start-up process in an optimized rocket nozzle. *Aerospace Science and Technology*, 12:6:485–486, 2008.

G. Oates. *Aerothermodynamics of gas turbine and rocket propulsion*. AIAA, 3rd edition, 1997.

J. Ostlund. *Flow processes in rocket engine nozzle whit focus on flow separation and side-loads*. Licentiate thesis, Royal Intitute of Technology, 2002.

R R Paz, M Storti, and L Garelli. Local absorbent boundary condition for nonlinear hyperbolic problems with unknown riemann invariants. *Computers & Fluids,*, 40:52–67, 2010.

S. Popinet. Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. *Journal of Computational Physics*, 190:572–600, 2003.

S.S. Rao. *Engineering optimization*. Wiley and Sons, 1996.

J. Remacle, B.K. Karamete, and M.S. Shepard. Algorithm Oriented Mesh Database. Technical report, Rensselaer Polytechnic Institute, Troy, NY, USA.

J. Remacle, X. Li, N. Chevaugeon, and M.S. Shephard. Transient mesh adaptation using conforming and non conforming mesh modifications. In *Proc. 11th Int. Meshing Roundtable, Sandia National Laboratories*, September 15-18 2002.

G.A. Ríos Rodriguez, E.J. López, N.M Nigro, and M. Storti. Refinamiento adaptativo homogéneo de mallas aplicable a problemas bi- y tridimensionales. In A.E. Larrateguy, editor, *Mecánica Computacional*, volume XXIV, pages 2365–2385, Buenos Aires, Noviembre 2005. AMCA.

G.A. Ríos Rodriguez, M. Storti, and N.M. Nigro. An h-adaptive unstructured mesh refinement strategy for unsteady problems. *Latin American Applied Research*, 39:137–143, 2009.

G.A. Ríos Rodriguez, M.A Storti, E.J López, and S.S Sarraf. An h-adaptive solution of the spherical blast wave problem. *International Journal of Computational Fluid Dynamics*, 25 (1):31–39, 2011. ISSN 1061-8562.

R.C. Ripley, F.S. Lien, and M.M. Yovanovich. Adaptive unstructured mesh refinement of supersonic channel flows. *Int. Journal of Comp. Fluid Dynamics*, 18:189–198, February 2004.

J. R. Shewchuck. What is a good linear finite element? Interpolation, conditioning, anisotropy and quality measures. http://www.cs.berkeley.edu/~jrs/papers/elemj.ps (Link last retrieved 27 December 2010), 2002.

Standard Template Library. Silicon Graphics, Inc., http://www.sgi.com/tech/stl, 1993-2012.

E.V. Sonzogni, A.M. Yommi, N.M. Nigro, and M.A. Storti. A parallel finite element program on a Beowulf cluster. *Adv. Eng. Softw.*, 33(7–10):427–443, July / October 2002.

M.L. Staten. Selective refinement of two and three-dimensional finite element meshes. Master's thesis, Department of Civil Engineering, Brigham Young University, 1996.

M. Storti. Aquiles Cluster at CIMEC, 2005-2010. http://www.cimec.org.ar/aquiles (Link last retrieved 27 December 2010).

M. Storti, N. Nigro, R. Paz, L. Dalcín, L. Battaglia, E. López, and G.A. Ríos Rodriguez. *PETSc-FEM, A General Purpose, Parallel, Multi-Physics FEM Program*. CIMEC-CONICET-UNL, 1999-2010. http://www.cimec.org.ar/petscfem.

G. Sutton and O. Biblarz. *Rocket propulsion elements*. John Wiley and Sons, 7th edition, 2001.

T. Tezduyar and M. Senga. Determination of the shock-capturing parameters in supg formulation of compressible flows. In Tsinghua University Press & Springler-Verlag, editor, *Computational Mechanics WCCM IV, Beijing, China 2004.*, 2004.

T. Tezduyar and M. Senga. Stabilization and shock-capturing parameters in supg formulation of compressible flows. *Computer Methods in Applied Mechanics and Engineering*, 195: 1621–1632, 2006a.

T.E. Tezduyar and M. Senga. Stabilization and shock-capturing parameters in SUPG formulation of compressible flows. *Computer Methods in Applied Mechanics and Engineering*, 195: 1621–1632, 2006b.

M.J.L. Tuner. *Rockets and spacecraft propulsion*. Springer, 2nd edition, 2006.

J. Waltz. Parallel Adaptive Refinement for Unsteady Flow Calculations on 3D Unstructured Grids. *Int. J. Numer. Meth. Fluids*, 46:37–57, 2004.