A consistent ale-rezoned mesh adaption algorithm for compressible flow finite-element calculations

Bernadette Palmerio

▶ **To cite this version:**

Bernadette Palmerio. A consistent ale-rezoned mesh adaption algorithm for compressible flow finite-element calculations. [Research Report] RR-0829, INRIA. 1988. inria-00075722

HAL Id: inria-00075722
https://inria.hal.science/inria-00075722

Submitted on 24 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

UNITÉ DE RECHERCHE
INRIA-SOPHIA ANTIPOLIS

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 829

A CONSISTENT ALE-REZONED MESH ADAPTION ALGORITHM FOR COMPRESSIBLE FLOW FINITE-ELEMENT CACULATIONS

Bernadette PALMERIO

AVRIL 1988

RR 829

A CONSISTENT ALE-REZONED MESH ADAPTION ALGORITHM FOR COMPRESSIBLE FLOW FINITE-ELEMENT CALCULATIONS

Bernadette PALMERIO

Laboratoire de Mathématiques, Université de Nice, Parc Valrose, 06034 NICE Cedex (FRANCE) and INRIA, 2004 Route des Lucioles, Sophia-Antipolis 1 et 2, 06565 VALBONNE (FRANCE)

ABSTRACT

In this paper we discuss a method for rezoning in a mesh adaption algorithm applying to two-dimensional unstructured triangular meshes, in order to calculate a compressible inviscid flow. Although the main purpose is to obtain an steady-state solution, the algorithm is organized like a consistent unsteady one.

PAPIER RÉCUPÉRÉ ET RECYCLÉ

UN ALGORITHME DE REMAILLAGE ARBITRAIRE LAGRANGIEN EULERIEN POUR LE CALCUL D'ECOULEMENTS COMPRESSIBLES EN ELEMENTS FINIS

Résumé

On présente une méthode de remaillage s'appliquant à des triangulations non-structurées; la méthode ALE est utilisée dans le cadre d'un algorithme instationnaire consistant permettant le calcul d'écoulements compressibles stationnaires.

Introduction

The adaptation of the mesh to the solution is an important issue in compressible gas flow simulation. Indeed, body-fitted meshes that are first generated for a new geometry are rarely well adapted to flow features; then modifications of the first guess have to be tried. In the hierarchy of modifications of a mesh, we can consider (for increasing complexity) :

(1)Mesh movement : node positions are changed, while the topology is not [1], [2], [3], [4].

(2)Mesh enrichment : nodes are added (the old nodes are not moved, but both integer and real data representing the mesh are changed), the new mesh can be deduced by element division [5], [6], [7], or by regeneration from the set of nodes [8].

(3)Global reconstruction : in [9], a front method is applied to reconstruct the whole mesh, taking into account a criterion deduced from previous computations.

While (2) and (3) are much powerful, because of the many degrees of freedom in changing the mesh, (1) is also interesting because of a somewhat lower cost for each change, which will make easier the capture by thin element strips of discontinuities and thin layers.

This paper deals with the first option (1), applied to steady calculations. While in several previous papers, the problem of the transfer of the solution from the old mesh to the new one was not performed in a consistent way, we propose here an adaption of the ALE method [10], in order to obtain a time-consistent global algorithm which will be more efficient and reliable. In this work, most steps are done explicitely w.r.t. time : For doing this stably, mesh movement is limited; this limitation is also used in order to keep the mesh admissible. The rezoning is fully conservative and relies on a FEM variant of the MUSCL method (referred as MUSCL-FEM [11],[12],[13]).

1. THE MATHEMATICAL AND NUMERICAL EULER MODEL

1.1. *The governing equations*

The conservative form of the stationary Euler system in the two-dimensional space is denoted by :

$$(1) \quad (F(W))_x + (G(W))_y = 0$$

with

$$(2)\quad W = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ e \end{pmatrix} \;;\; F(W) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (e+p)u \end{pmatrix} \;;\; G(W) = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (e+p)v \end{pmatrix}$$

ρ, ρu, ρv, e are the conservative variables ; ρ is the density ; $V = (u, v)$ is the fluid velocity, e is the total energy ; p is the pressure and is obtained through the equation of state :

$$(3)\quad p = (\gamma - 1)(e - \tfrac{1}{2}\rho(u^2 + v^2))$$

where $\gamma = 1.4$.
To solve the system (1),(2) we use (pseudo) unsteady calculations relying on:

$$(4)\quad W_t + (F(W))_x + (G(W))_y = 0 .$$

1.2. *Spatial approximations*

We assume that Ω (see Fig.1) is a bounded domain of $\mathbb{R}^2$. Let $\mathcal{T}_h$ a standard triangulation of Ω and h the maximal length of the edges in $\mathcal{T}_h$. A finite-volume scheme is then constructed : cells are defined via the medians of the triangles ; the discrete system is then obtained by summation of flux integrals over each bi-segment $G_i\, I_{ij}\, G_j$ (see Fig.2) with one evaluation at I_{ij} of the integrand. At the integration point I_{ij} we define two values W_{ij} and W_{ji} by interpolating the dependent variable W in each neighboring cell i and j ; this can be done in several ways which we now detail. The construction of the basic scheme employs some ideas of B. van Leer [11] and was derived by F. Fezoui [12].

1.2.1. *Flux-splitting and original first-order scheme*

In the first-order accurate scheme, a constant-by-cell interpolation is employed

$$(5)\quad W_{ij} = W_i \;\; and \;\; W_{ji} = W_j .$$

Then a flux-splitting is applied ; in the present calculations, we considered the following flux-splittings:
- the Q-splitting ([11]) writes

$$(6)\quad \Phi_{ij} = \tfrac{1}{2}(H(W_{ij}) + H(W_{ji})) + \tfrac{1}{2}|P(\tfrac{W_{ij}+W_{ji}}{2})|(W_{ij} - W_{ji})$$

H is a linear combination of F and G (see [11] and [13], P, the derivative of H is diagonalizable :

$$(7) \quad \begin{cases} P = T \wedge T^{-1}, \ \wedge = (\lambda_k) \\ P^{\pm} = T(\lambda_k^{\pm})T^{-1} \\ |P| = T(|\lambda_k|)T^{-1} \end{cases}$$

- Osher's splitting ([13]) is defined in short by :

$$(8) \quad \Phi_{ij} = \tfrac{1}{2}(H(W_{ij}) + H(W_{ji})) + \tfrac{1}{2}\int_{W_{ji}}^{W_{ij}} |P(W)|dW$$

The Q-splitting and Osher's splitting are employed in the (explicit) spatial approximation. The Q-splitting is less computer time consuming and suitable to transonic calculations, while Osher's splitting is a more robust scheme that is employed in supersonic calculations.

1.2.2. *Second-order accurate extension*

Following the so-called MUSCL-FEM formulation of [11] and [13], an elegant way to reach second-order accuracy is to derive the values W_{ij} and W_{ji} appearing in the equations (5), (6), (8) from linear interpolations. The dependent variable W is linearly interpolated on each cell C_i around a vertex i ; this interpolation is computed from the node value W_i and from an approximate gradient $(\overline{W}_x, \overline{W}_y)$ obtained using the Lagrange interpolant W_L as follows :

$$(9) \quad measure\ (C_i).\ \overline{W}_x(i) \ = \ \int_{C_i} \frac{\partial W_L}{\partial x} dx\ dy$$

C_i is the cell on the control volume around the vertex i (Fig.2). Then the two values W_{ij}, W_{ji} are defined at the mid-point I_{ij} as follows :

$$(10) \quad \begin{cases} W_{ij} \ = \ W_i + \dfrac{1}{2} \begin{Bmatrix} \overline{W}_x(i) \\ \overline{W}_y(i) \end{Bmatrix} . \ \overrightarrow{ij} \\ W_{ji} \ = \ W_j + \dfrac{1}{2} \begin{Bmatrix} \overline{W}_x(j) \\ \overline{W}_y(j) \end{Bmatrix} . \ \overrightarrow{ji} \end{cases}$$

Actually, in the present work, this interpolation has not been applied to the usual conservative variables but instead to the so-called physical/primitive variables (density, velocity components, pressure) following van Leer ([11]). However the flux balance is used to advance in time the conservative variables.

1.3. *Time integration*

An explicit pseudo-unsteady method is used ; considering the unsteady Euler system with local time-stepping, we use a two-step formulation, involving a predictor step

that applies a linearized central differencing scheme, and a corrector step based on the spatial discretization defined previously ; we refer to [14] for details. The use of local time-stepping is essential for a good convergence since strong local mesh shrinking will be produced by adaptation.

We shall combine this scheme with moving the mesh $\mathcal{M}_0$, using the following ingredients:
- constructing a new mesh by the spring method we deduce the displacement of each nodes.
- limiting this displacement we obtain an admissible mesh $\mathcal{M}_{ad}$.
- updating the metrics, and transfering the solution from the mesh $\mathcal{M}_0$ onto the new mesh $\mathcal{M}_{ad}$.

2. THE SPRING METHOD

2.1. *Mesh definition by spring equations*

Recall that the meshes considered are Finite Element-type triangulations. The topology is given (and fixed) ; it is defined by the following informations : are known the indices of vertices for each triangle and the boundary logical pointers for all the vertices. Only deformations are then allowed. To move the mesh, we first assume that each mesh node i is connected to each adjacent mesh point j by a fictitious spring with strength F_{ij} (see Fig.3). For all the node i of the mesh $\mathcal{M}_0$, for each j neigbor of i,

$$(11) \quad \overrightarrow{F_{ij}} = G\ K_{ij}\ \overrightarrow{ij}$$

in which G and K_{ij} are positive numbers; K_{ij} depends on the flow properties (for adaptation). The unknowns are the coordinates $(X(i),\ Y(i))$ of vertices ($i = 1, \cdots, N, N$ is the number of vertices). The resulting mesh is the solution of the equilibrium system, which is written, for each vertex i :

$$(12) \quad \sum_{j\ neighbor\ of\ i} \overrightarrow{F_{ij}} = 0\ .$$

2.2. *Flow criterion*

For choosing the coefficient K_{ij} (see eq. (11)), two kinds of adaptive criteria are generally used :

(I) the mesh can be adapted from an estimation of the local approximation error , the objective being to reduce it.

(II) the mesh can be adapted in order to have a better representation of some sensor S related to the flow.

Since we are interested in directional adaptation, a criterion belonging to the second kind (II) is more straightforward to construct and analyse. We plan to discuss the first option (and other options) in a future paper. Since shock capture is looked for, we may choose S as the Mach number M. It is assumed to be numerically defined on each vertex i. More precisely, in a first study, we put , for two adjacent nodes i, j :

(13) $K_{ij} = |S(i) - S(j)|$

which approximates the directional variation of the sensor S.
Also a smoother value of the spring constant K_{ij} can be introduced to take into account values at each neighbor of the nodes i and j : we refer to [4].

2.3. *Solution algorithm for the spring equations*

The coordinates $X(i)$, $Y(i)$ of the vertices are issued of the equations (11),(12). They are then the solution of a linear system :

(14) $AX = B_1,\ AY = B_2$

with

(15) $$\begin{cases} A_{ij} = -G.K_{ij} \ for\ i \neq j,\ j\ neighbor\ of\ i \\ A_{ii} = \displaystyle\sum_{j\ neighbor\ of\ i} G.K_{ij} \end{cases}$$

2.3.1. *Well posedness and notion of reference mesh*

Let us assume that the flow solution is uniform in some region of the computational domain, then the K_{ij} are identically vanishing in that region, so that the node system (19) is not well posed. Then we have to add some device in order to well define the mesh in such regions. In fact this means that the algorithm must be able to construct an admissible mesh in the particular case of a globally uniform flow ; let us call "reference mesh" the resulting mesh in that case : note in particular, that the present algorithm can be used as a mesh generation method. We consider two options for constructing an admissible reference mesh.

2.3.1.1. *Residual strength*

The K_{ij} (eq (15)) are replaced by

(16) $\overline{K}_{ij} = K_{ij} + C_0,\ C_0 > 0$

C_0 is a constant parameter.
The corresponding reference mesh is defined by the equilibrium of springs with equal strength. Such meshes tend to attain some "optimality" with respect of angles (equilaterality) and (or) repartition (location of nodes).

2.3.1.2. *Connection to initial mesh*

It is possible to enforce diagonal dominance in the matrix A when we wish to adapt the mesh while maintaining it close to the initial mesh. In this purpose, additional terms are easily introduced by modifying the equilibrium Equation (12) as follows : for each node i,

$$(17) \quad (\sum_{j \ neighbor \ of \ i} F_{ij}) + F_i^{in} = 0$$

with

$$(18) \quad F_i^{in} = (F_i^{in,X}, F_i^{in,Y}), \ F_i^{in,Z} = C_1(Z(i) - Z_0(i)), \ Z = X \ or \ Y$$

where $(X_0(i), Y_0(i))$ are the coordinates of the initial mesh ; C_1 is a constant parameter to be adequately chosen.
In [4] we give some examples of reference mesh.

A is a M-Matrix, that is, it satisfies the following properties :

$$(19) \quad A_{ii} \geq 0 \ ; \ A_{ij} \leq 0 \ for \ i \neq j, \ \sum_j A_{ij} \geq 0 \ \forall_i$$

More precisely, for an internal node with all neighbors internal, (A is weakly positive) :

$$(20) \quad \sum_{j \ neighbor \ of \ i} A_{ij} = 0$$

For the nodes i which have boundary nodes neighbors (A is positive) :

$$(21) \quad \sum_{j \ neighbor \ of \ i} A_{ij} > 0 \ if \ G \, K_{ij} \neq 0$$

In that case, provided that the A_{ij} in Equations (20) are not all vanishing, the equation gives a right hand side which corresponds to boundary vertices that we assume to be either fixed or moving along the boundary.

2.3.2. *Solution of mesh system*

Since, using the above devices, we obtain a diagonally dominant matrix, we can apply a jacobi iteration for solving the mesh system. Since the global process, that is including the flow solution will be also iterative we choose to perform only a few jacobi sweeps at each mesh movement.

3. LIMITATION OF THE DISPLACEMENT OF EACH NODE

An admissible mesh is not surely attained with the spring method : when we try to determine thin layers, the mesh may overlap, particularly in the case where the refined zone becomes very fine for capturing highly curved layers using parallel broken lines. For treating this case, we introduce the following conditions which do not systematically guarantee the mesh admissibility, but are not very costly to enforce : for each node i we construct the following safety area : the sphere with center i and a radius equal to k times the smallest height h_i issued from i in the corresponding cell C_i (see Fig.1), k is a real positive coefficient which has to be smaller that 0.5). Finally, we lower globally the displacement of all the nodes so that each node remains in its safety area as follows:

$$(22) \quad \|\overrightarrow{\Delta M_i'}\| = k' \|\overrightarrow{\Delta M_i}\| \ , \ \textit{with } k' \textit{ given such that, for all } i$$

with

$$(23) \quad k' \|\overrightarrow{\Delta M_j}\| \leq 0.5\, h_j \quad \textit{for all } j \ .$$

4. ALE MESH TO MESH CONSERVATIVE TRANSFER

Our goal is to obtain conservative computation of the flow corresponding to the mesh obtained after the spring-limited step (see Sec. 2 and 3). We realize this by the mean of a discretized equation in which we have to take into account two phenomena :
- the mesh movement, represented by a node velocity.
- the geometrical change of the mesh, appearing in the areas of the cells.

Let us call $\mathcal{M}^{n-1}(W^n)$, the mesh (the flow computation in $\mathcal{M}^{n-1}$) before the spring-limited step and $\mathcal{M}(W^n)$, the final one (the flow computation in $\mathcal{M}^n$), $\overline{W}^n$ is the contribution to the flow, due to the mesh movement :

$$(24) \quad (\mathcal{M}^n, \overline{W}^n) \ = \ P_n \, (\mathcal{M}^{n-1}, \ W^{n-1})$$

in which P_n is the transfer operator from the $(n-1)^{th}$ mesh to the n^{th} mesh. The application of P_n is equivalent to solving the advection equation with a discretization domain defined by $\mathcal{M}_n$:

$$(25) \quad \overline{W}_t^n \ - \ \overrightarrow{V}^{n-1} \, . \nabla \, \overline{W}^n . \ = \ 0$$

$\overrightarrow{V}^{n-1}$ is the velocity of the mesh $\mathcal{M}^{n-1}$. Actually we do not use a discretization of the equation (25) : As we have to preserve the conservation of variables, we are solving an equation of the type :

$$(26) \quad (area\ (C_i)\ \overline{W}_i)_t \ - \ div(\overrightarrow{V}_i\ \overline{W}_i) \ = \ 0$$

C_i the cell of the node i (see Fig.2). $\overrightarrow{V}_i$ is the velocity of the node i. The equation is written for all the nodes i in the mesh $\mathcal{M}^n$. The introduction of the cell area C_i corresponds to the Jacobian of the mapping from $\mathcal{M}^{n-1}$ to $\mathcal{M}^n$. We can explicit the equation (26) by :

$$(27) \quad area(C_i^n)\ \overline{W}_i^n \ - \ area(C_i^{n-1})\ W_i^{n-1} \ = \ \Phi_i^{n-1}$$

with

$$(28) \quad \Phi_i^{n-1} \ = \ \sum_{j\ neighbor\ of\ i} \int_{\partial C_i^{n-1} \cap \partial C_j^{n-1}} W_{\alpha j}^{n-1}.D_i^{n-1}.\overrightarrow{n}\ d\sigma$$

with

$$(29) \quad D_i^{n-1} \ = \ \overrightarrow{V}_i^{n-1}\ \Delta t$$

C_i is the cell issued of the node i (see Fig.2) in the mesh $\mathcal{M}$. $partial C_i^n$ is the boundary of the cell C_i^n. In the equations (24) to (29) ,the index n corresponds to the values at time step n and the index i corresponds to the node i ; D_i^{n-1} is the displacement issued from the velocity $\overrightarrow{V}_i^{n-1}$; $\overrightarrow{n}$ is the outward normal to $\partial C_i^{n-1} \cap \partial C_j^{n-1}$. The value of $W_{\alpha j}^{n-1}$ in the equation (28) is determined by solving(26) with one of the two following upwind schemes.

In the first-order accurate case, we put :

$$(30) \quad W_{\alpha j}^{n-1} \ = \ \begin{cases} W_i^{n-1} \ \ if \ \ \displaystyle\int_{\partial C_i^{n-1} \cap \partial C_j^{n-1}} D_i^{n-1}\ \overrightarrow{n}\ d\sigma \ > \ 0 \\ W_j^{n-1} \ \ if \ \ \displaystyle\int_{\partial C_i^{n-1} \cap \partial c_j^{n-1}} D_i^{n-1}\ \overrightarrow{n}\ d\sigma \ < \ 0 \end{cases}$$

In the second-order accurate case, the MUSCL-FEM is applied : W_{ij}^{n-1} and W_{ji}^{n-1} are defined at the mid point I_{ij}^{n-1} (see Fig.2) of $\partial C_i^{n-1} \cap \partial C_j^{n-1}$ from the equation (10) in which W_{ij}, W_{ji}, W_i, $W_j, \cdots$ are replaced by W_{ij}^{n-1}, W_{ji}^{n-1}, W_i^{n-1}, W_j^{n-1},

$\cdots$. Moreover we use limiters with physical variables see [12],[13]. After computing of $\overline{W}_n$ (equation (25)), we first reactualize the metrics, i.e. the cell areas, the gradient of the basis functions, the coefficients corresponding to the discretization above each side of the triangulation. Secondly we obtain finally the flow solution W_i^n above the new mesh from the formula :

$$(31) \quad W_i^n = (area(C_i^{n-1}).W_i^{n-1} + area(C_i^n)\overline{W}_i^n) / area(C_i^n)$$

The Courant condition for the explicit scheme (31) is automatically satisfied by (23)

5. ORGANIZATION OF THE ALGORITHM

The procedure to couple the flow calculation and the mesh adaption is given by the following algorithm :

0 - initialization of the mesh (the mesh index, imesh, is equal to 0) and of the Eulèr system.

1 - Flow calculation (Section 1).

2 - Mesh index updating : imesh = imesh + 1.

3 - First calculation of the displacement of mesh nodes by springs (Section 2.3).

4 - Limitation of each displacement obtained in Step 3 (Section 3).

5 - Final conservative calculation of the mesh and of the metrics (Section 4).

6 - Go to Step 1.

We have to rely the differents steps of this algorithm by taking into account of two requirements :

- the limiting of the CPU time

- the good-converging of the final pair (mesh, flow).

The first point is attained from the following ways : - several iterations (one to cent iterations) in the flow calculation (Step 1) - a few iterations in solving of the spring system (Equation (15)) by the Jacobi iteration (typically, five iterations (Step 3)).

The second point is obtained by starting from an initial flow well-converged on the initial mesh and by a performing a sufficiently large number N of remeshings. At each new mesh, we compute the the mesh residual R_M^n :

$$(32) \quad R_M^n = \|(X_i^n - X_i^{n-1})\|$$

X_i^n is the limited position of the node i (end of Step 4) ; X_i^{n-1} is the previous position. The L^2 norm is used. The well-convergence of the algorithm is measured by the distance from R_M^n to zero. Near convergence, the mesh is no more much modified by the motion so that the global algorithm works only for the convergence of the flow calculation.

6. NUMERICAL ILLUSTRATIONS

6.1. *The shock tube test*

We give here a classical example of unsteady flow calculation in two cases : in Fig.4, the results obtained adaptionless, in Fig.5 the results obtained with mesh motion, using the algorithm defined in Section 5. The initialization of the flow is uniform from the boundary condition. The flow calculation is realized by using the second-order accurate scheme described in 1.2.2. with Osher's splitting (see 1.2.1). In the adaption case the number of solver iterations is equal to one. The time-stepping is uniform and the C.F.L. is equal to 0.5. In Step 3 (adaption case), the sensor is the Mach number (see Section 2.2) and we choose, as values of parameters :

$$C = 5000 \ (Eq. \ (11)) \ ; \ C_0 = 1 \ (Eq. \ (16)) \ ; \ C_1 = 0 \ (Eq. \ (18))$$

the number of Jacobi iterations is equal to 5. In the adaption case, the time $t = 0.16$ is attained after 629 remeshings. We observe in the Fig.5, the final mesh is well refined in the shock zones. This calculation is too much costly, but demonstates the consistency of the ALE formulation.

6.2. *External supersonic flow around a cylinder (Fig.6)*

The Mach number at infinity is equal to 8. The C.F.l. is equal to 0.5. The values of the parameters are the same than in the above case, excepted for the number of iterations in Step 1, which is equal to 50; we use local time-stepping. In Step 5 the flow projection is performed by the first-order accurate scheme explained in Section 4. The final mesh is obtained after 25 remeshings. Clearly the final mesh and the Isomach contours put in evidence a good capture of the bow shock.

6.3. *External supersonic flow around a NACA0012 (Fig.7, 8, 9, 10, 11)*

The Mach Number at infinity is equal to 2., the angle of attack is equal to 10 degrees; the CFL number is equal to 0.6 ; the scheme used for the flow calculation

is the second-accurate one (see 1.2.2.). Before the mesh adaption by the presented method, we perform two successive enrichments (Fig.9,10). The enrichements are based on the gradient of the entropy (see [5],[6]). The second enriched mesh and the corresponding converged steady-flow calculation are the starting values (Step 0) of the motion algorithm (Section 5). The number of iterations in Step 1 is equal to 100, the flow projection is performed by the first-order accurate scheme. The final mesh is obtained after 30 remeshings. If we compare the same contours, the isomach ones for example, we see the progression in the capturing of the different shocks. If the enrichment steps permit us to locate an adequate number of points in the shock areas, the final moving step give us well captured shocks (Fig.10, 11).

6.4. *Flow past an indented body (Fig.12, 13)*

We give now a further illustration of the method by its application to the calculation of the flow past an idealized spatial aircraft. The Mach number at infinity is 25. and the angle of attack is 40 degrees. Both the high Mach number and the flow at the rear part of the body make necessary the use of a robust algorithm. Several oblique shocks are also present in the flow. Two enrichment phases (based on the gradient of entropy as criterion) are first perform, yielding a mesh with 2377 nodes. The mesh displacement is then applied; results are presented in Fig.12 and 13.

6.5. *Interacting jets (Fig.14)*

A last illustration is a calculation of the interaction of two supersonic jets in a partly open vessel. The Mach number at the entry of the jets is 1.5 . The main new feature of this test is that slip lines are the main detail of the flow. Two enrichment phases (based on the gradient of the Mach number) have been performed before the node displacement. Most part of the slip line are captured with a reduced numerical thickness, also the aspect ratio of the corresponding elements is not much high (Fig.14).

7. CONCLUSION

A new efficient method of moving adaptive grids for solving the multi-dimensional Euler equations is presented. With the present method, adapting grids to high curvature surfaces or shocks is rather easy : it consists in a high number of both mesh solutions and flow projection combined with iterations of the flow system. The adaptive procedures themselves require a small fraction of the computational cost; the additional advantage of the present work with regard to the preceding ones ([5],[3],[4]) is that the shocks are well handled (location, monotonicity) during the time evolution of the flow : this permits us to solve adequately the flow and the shocks in unsteady cases or in steady case by an unsteady approaches. Note that the method for moving the grid requires a priori a sufficiently large number of almost

well distributed points: this leads us to perform enrichment steps before beginning the mesh movement phase. In that case, the use of movement is particularly efficient as a mesh smoother.

8. ACKNOWLEDGEMENTS

The author thanks A. Dervieux and B. Larrouturou for their suggestions and assistance in this work.

REFERENCES

[1] GNOFFO P.A., A finite Volume adaptive grid algorithm applied to planetary entry flowfields, AIAA J., Vol. 21 , no 9 ; 1249-1254 (1983).

[2] NAKAHASHI K., DEIWERT G.S., A practical Adaptive-grid Method for complex fluid-flow problems, Lecture Notes in Physics, Vol. 218, 422-426, Springer (1985).

[3] PALMERIO B., DERVIEUX A., Application of a F.E.M. moving node adaptive method to accurate shock capturing, in Numerical grid generation in Computational Fluid Dynamics, Hauser J., Taylor C., eds. Pineridge Press Swansea 1986.

[4] PALMERIO B., A two dimensional F.E.M. adaptive moving node method for steady Euler flow simulation, to appear in Computer Methods in Applies Mechanics and Engineering.

[5] PALMERIO B., Self-adaptive F.E.M. algorithms for the Euler equations, INRIA Report 338(1984).

[6] PALMERIO B., BILLEY V., DERVIEUX A., Self-adaptive mesh refinements and Finite Element Methods for solving the Euler equations, Proc. of the ICFD Conf. on Numerical Methods for Fluid Dynamics, Reading (1985), (369-388).

[7] ANGRAND F., BILLEY V., DERVIEUX A., PERIAUX J., POULETTY C., STOUFFLET B., 2-D and 3-D Euler flow calculations with a second-order accurate Galerkin F.E.M. AIAA Paper 85-1706.

[8] MAVRIPLIS D., JAMESON A., Multigrid solution of the Euler equations on unstructured and adaptive meshes , AIAA paper 87-0353,1987.

[9] PERAIRE J., PEIRO J., MORGAN K., ZIENKIEWICZ O.C., Finite- element mesh generation and adaptive procedures for CFD, GAMNI-SMAI Conference on automated and adaptive mesh generation, Grenoble, France, oct. 1-2, 1987.

[10] HIRT C.W., AMSDEN A.A., COOK J.L., An arbitrary Lagrangian- Eulerian Computing method for all flow speeds, J. Comp. Phys. 14 (1974), 227-253.

[11] VAN LEER B., Computational methods for ideal compressible flow, von Karman Institute for Fluid Dynamics, Lecture Series 1983-04.

[12] FEZOUI F., Résolution des équations d'Euler par un schéma de van Leer en éléments finis, INRIA-Report 358 (1985).

[13] OSHER S., CHAKRAVARTHY S., Upwind schemes and boundary conditions with applications to Euler equations in general geometries, J. of Comp. Physics, Vol. 50, 3, 447-81 (1983).

[14] STOUFFLET B., PERIAUX J., FEZOUI F., DERVIEUX A., Numerical simulation of 3-D hypersonic Euler flows around space vehicles using adapted finite elements, AIAA Paper 87-0560 (1987).

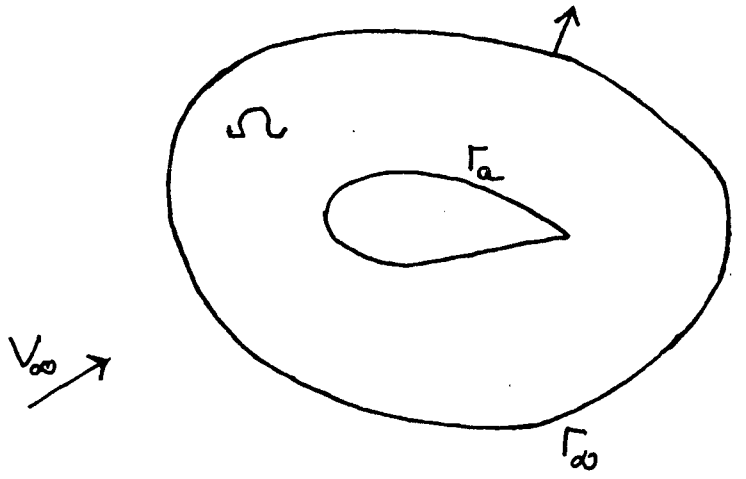
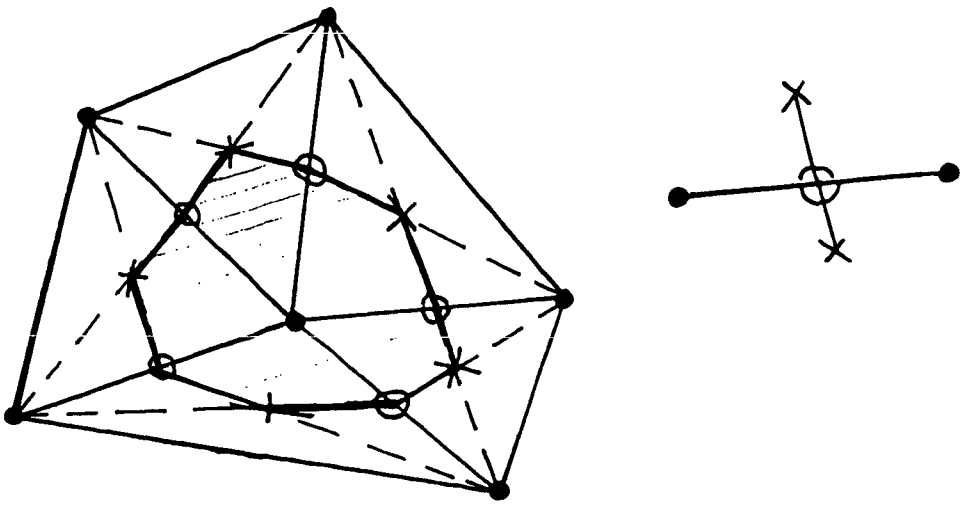


Figure 1 : Computational domain and its boundary



● Vertex (node)

○ numerical integration point

× centroid

▨ Cell C_i

Figure 2 : Cell C_i corresponding to the node i.

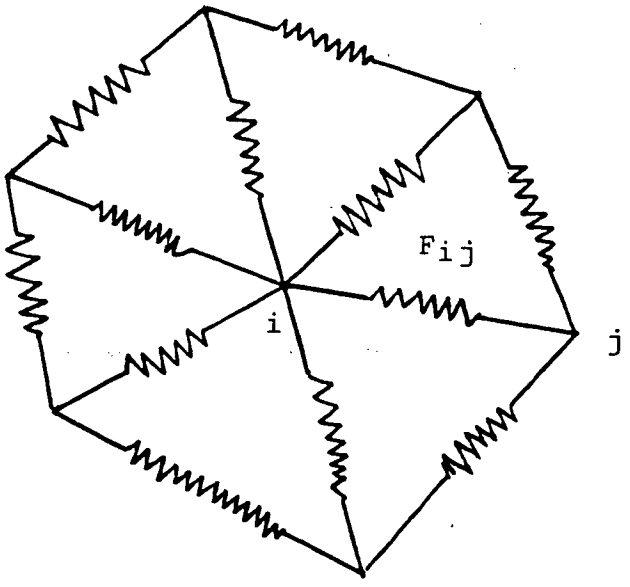
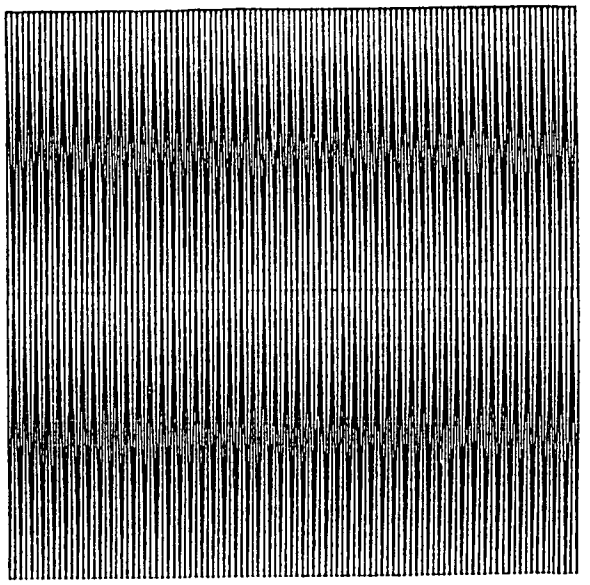
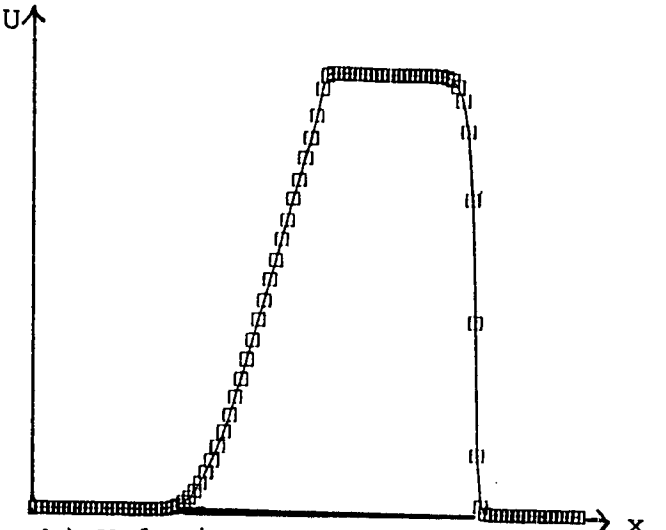


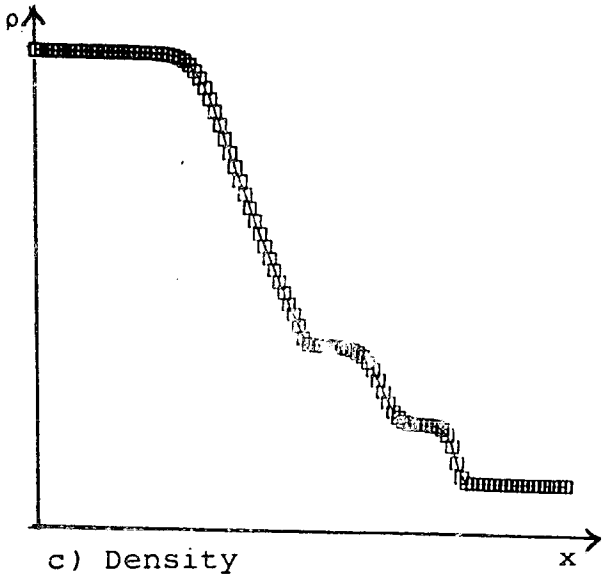
Figure 3 : Representation of the fictitions springs F_{ij} relying each mesh nodes i,j.



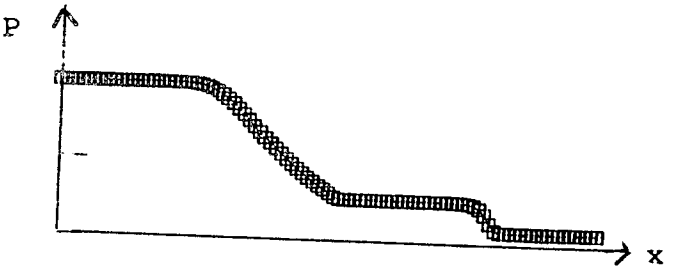
a) Mesh (101x3 points)



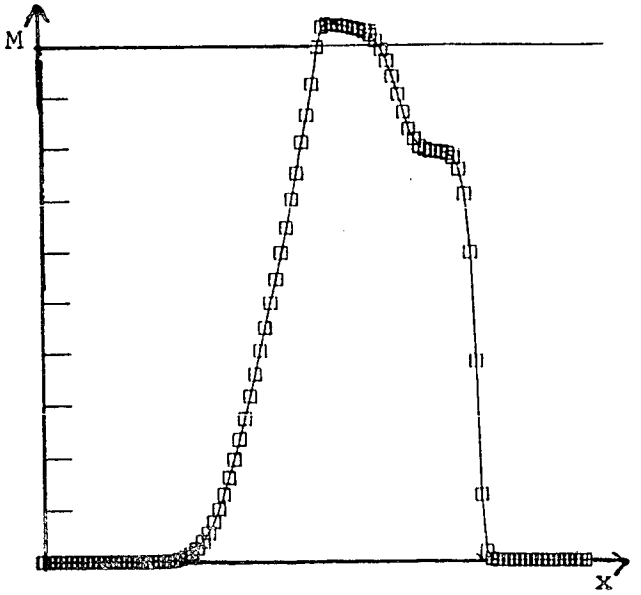
b) Velocity



c) Density

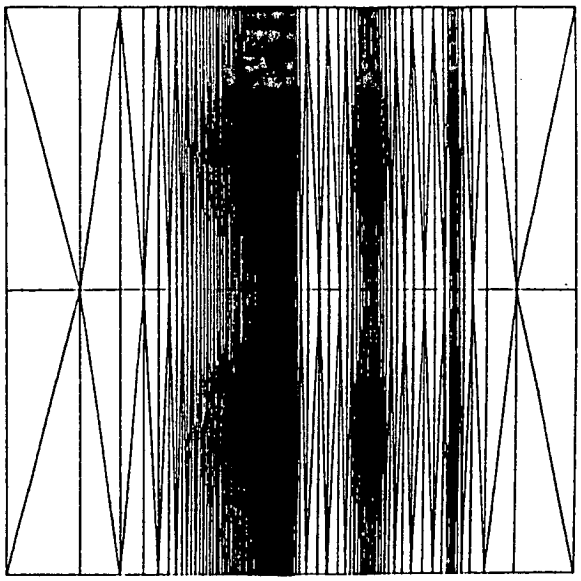


d) Pressure

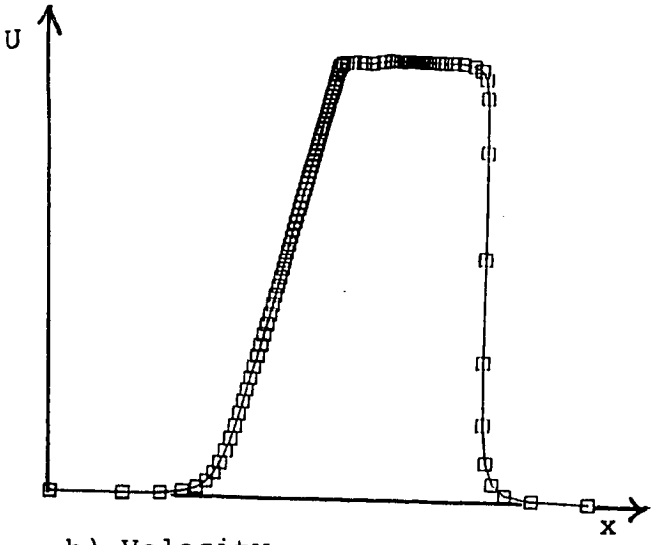


e) Mach

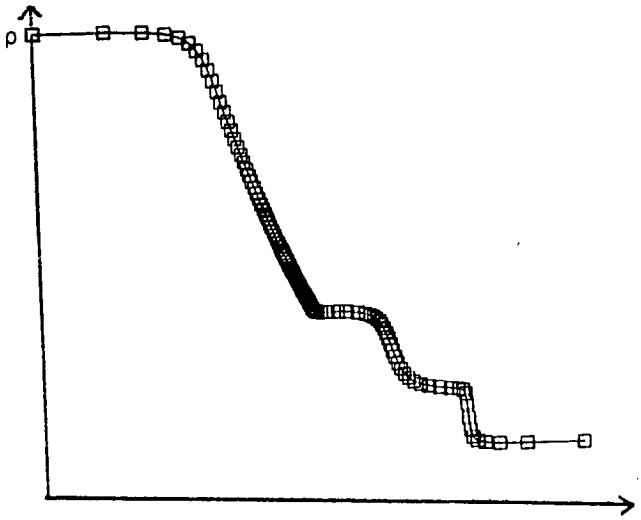
Figure 4 : the shock tube test (Osher second-order accurate scheme, t = 0.16)



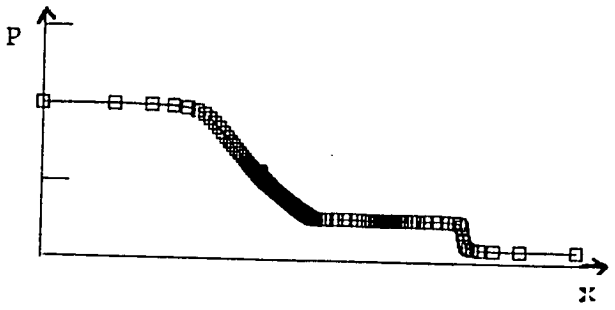
a) Final mesh (101x3 points)



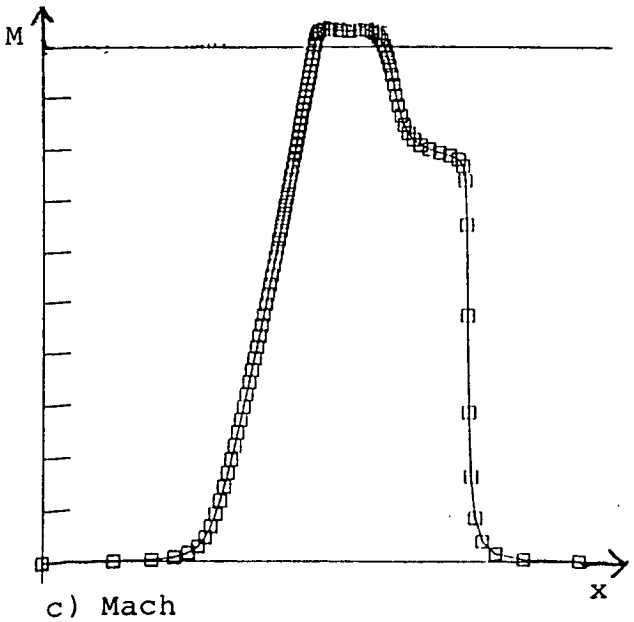
b) Velocity



c) Density

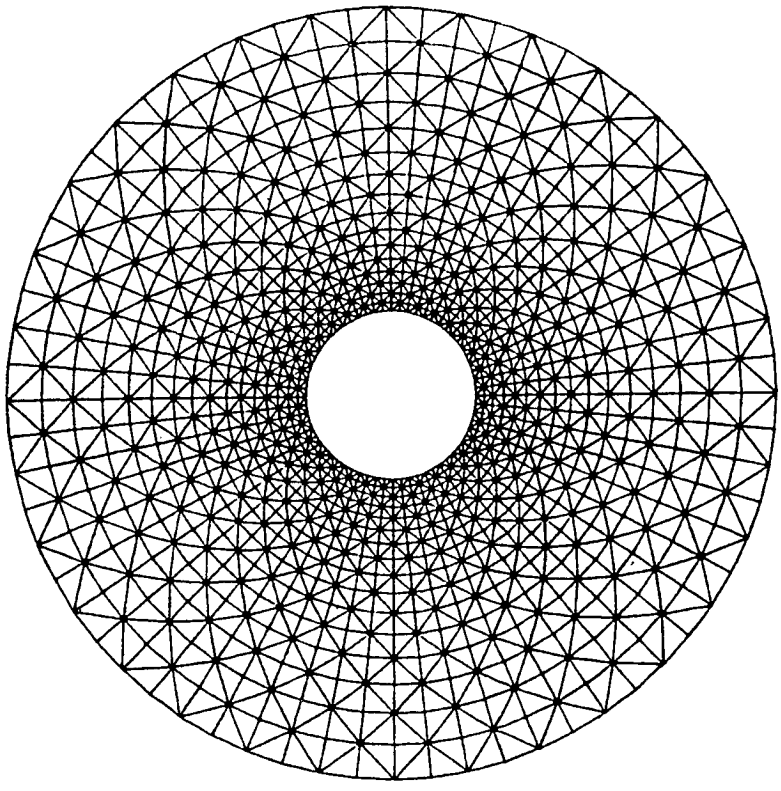


d) Pressure

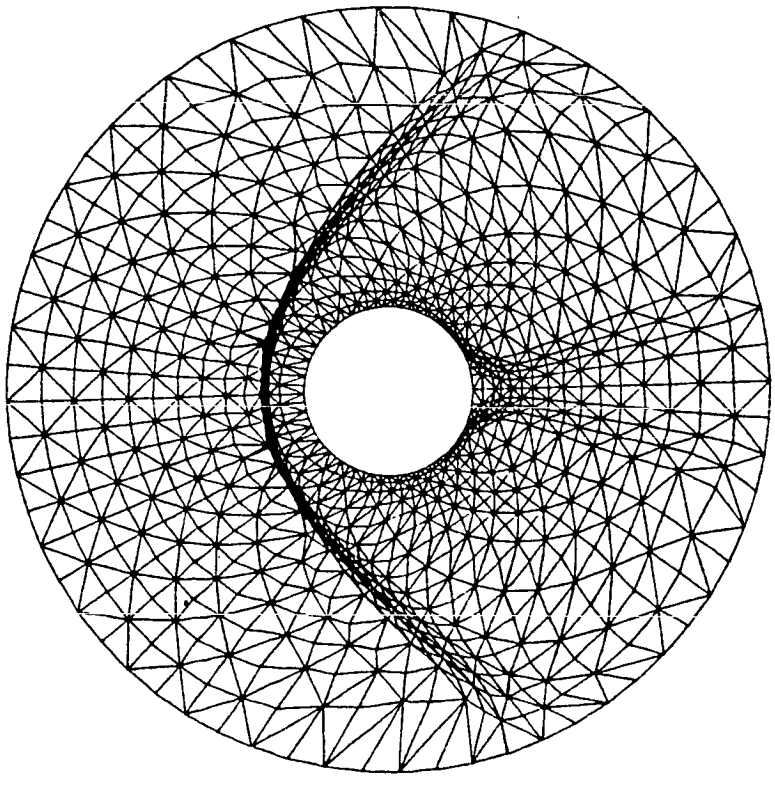


c) Mach

Figure 5 : The shock tube test (Osher second-order accurate scheme, t = 0.16).

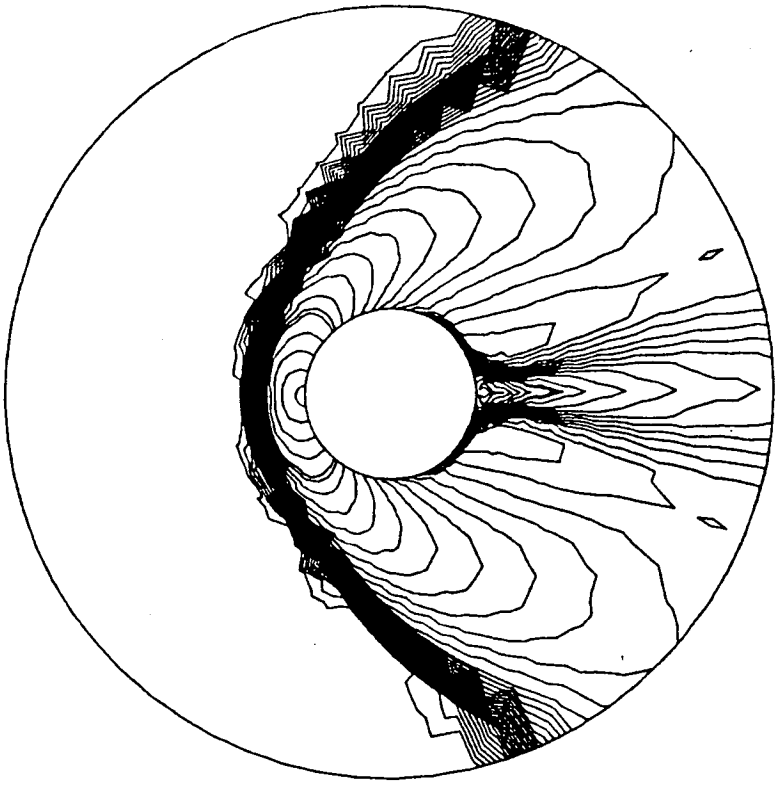


a) Initial mesh (1088 points)

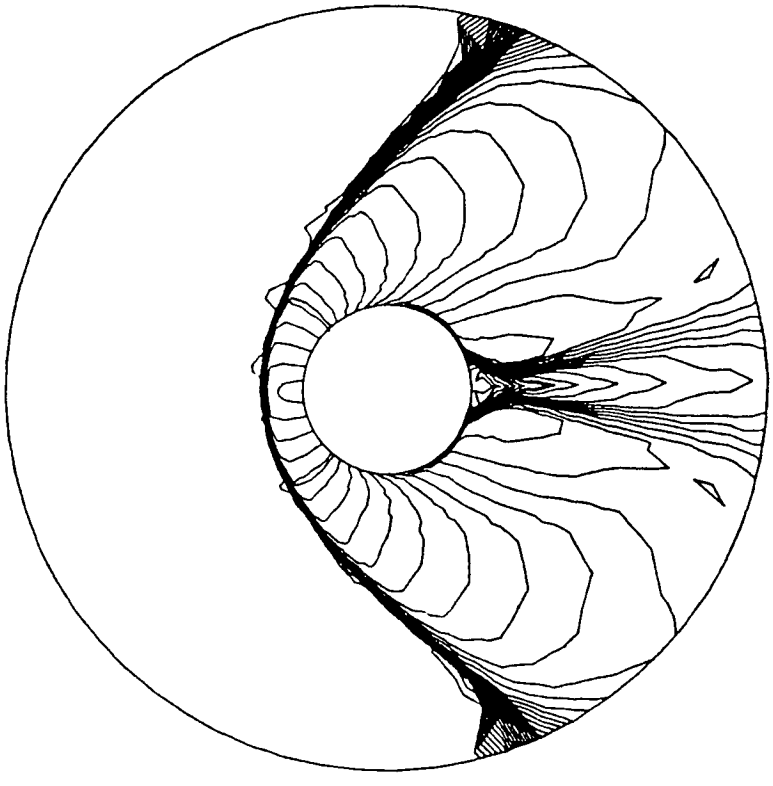


b) Final mesh

Figure 6.1 : external flow around a cylinder with a Mach at infinity equal to 8., mesh movement with ALE method.

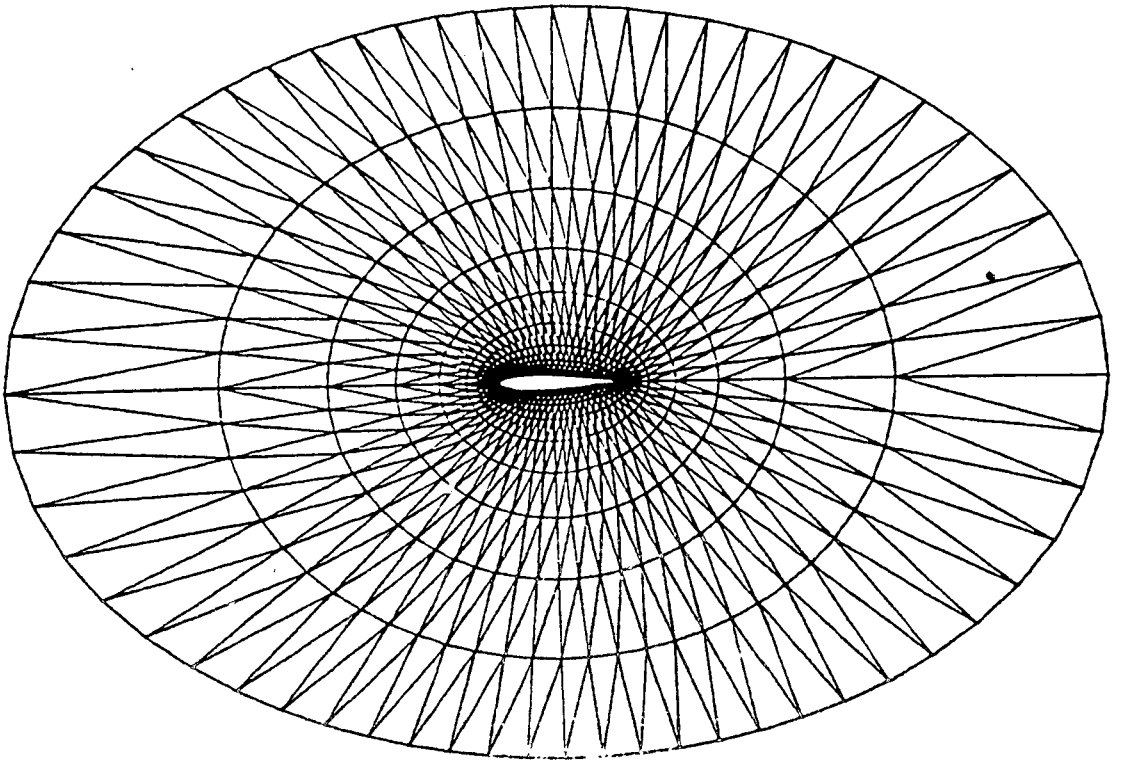


a) Without adaption

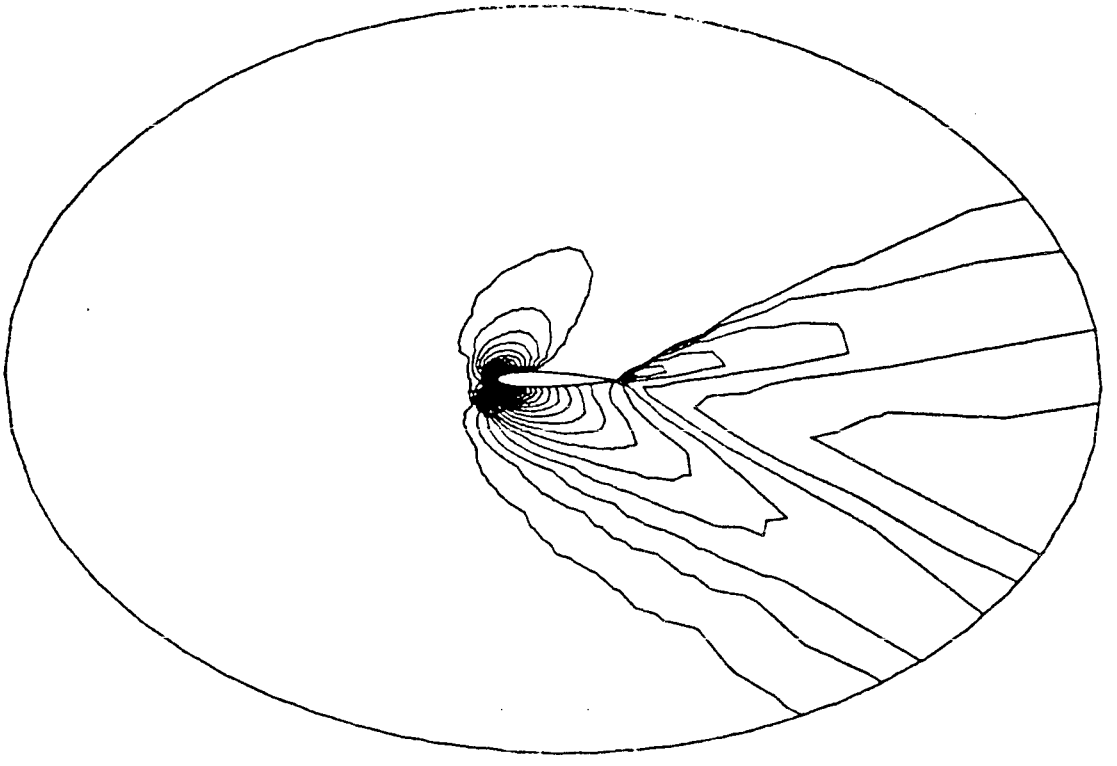


b) with mesh moving

Figure 6.2 : Isomach contours corresponding to Figure 9.1.

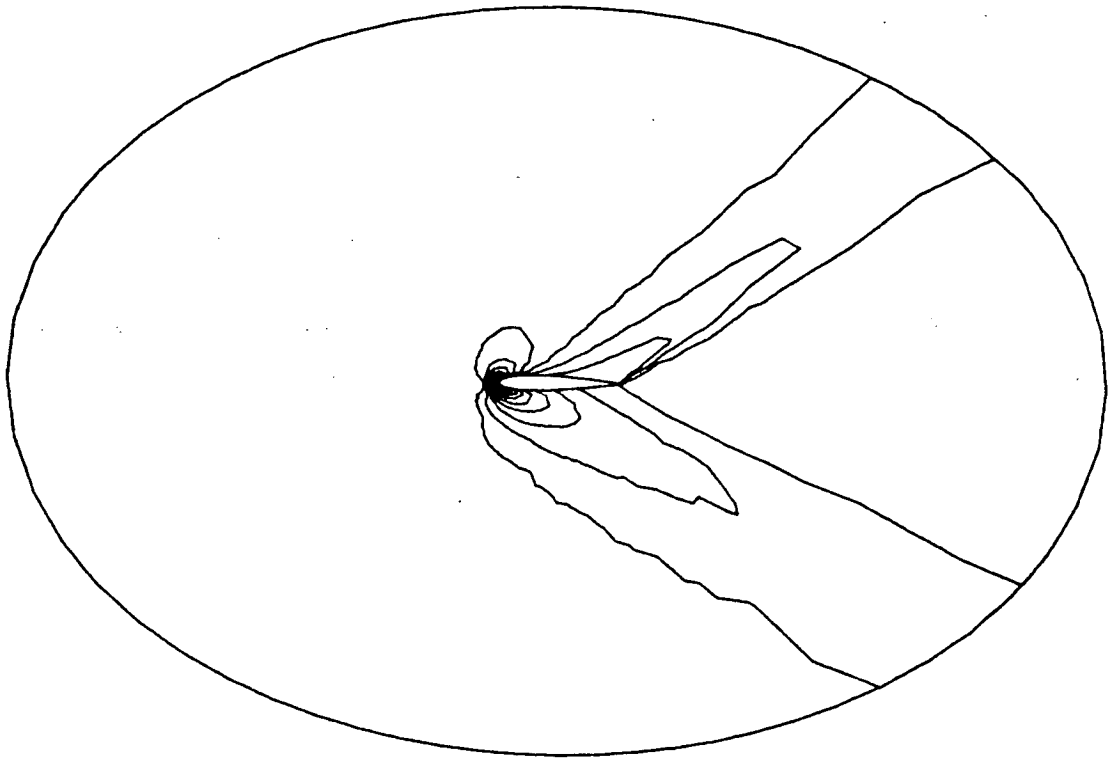


a) Initial mesh (600 points).



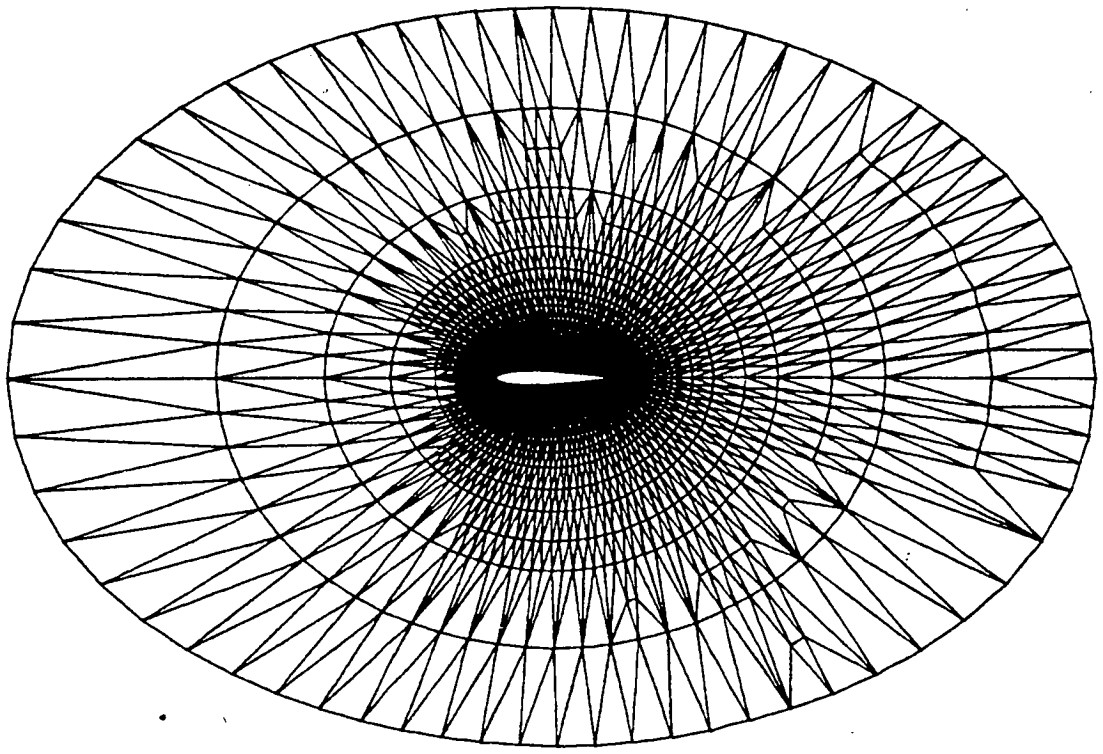
b) Isomach contours

Figure 7 : External flow around a NACA 0012 with q Mach at infinity equal to 2. and an angle of attack equal to 10 degrees.

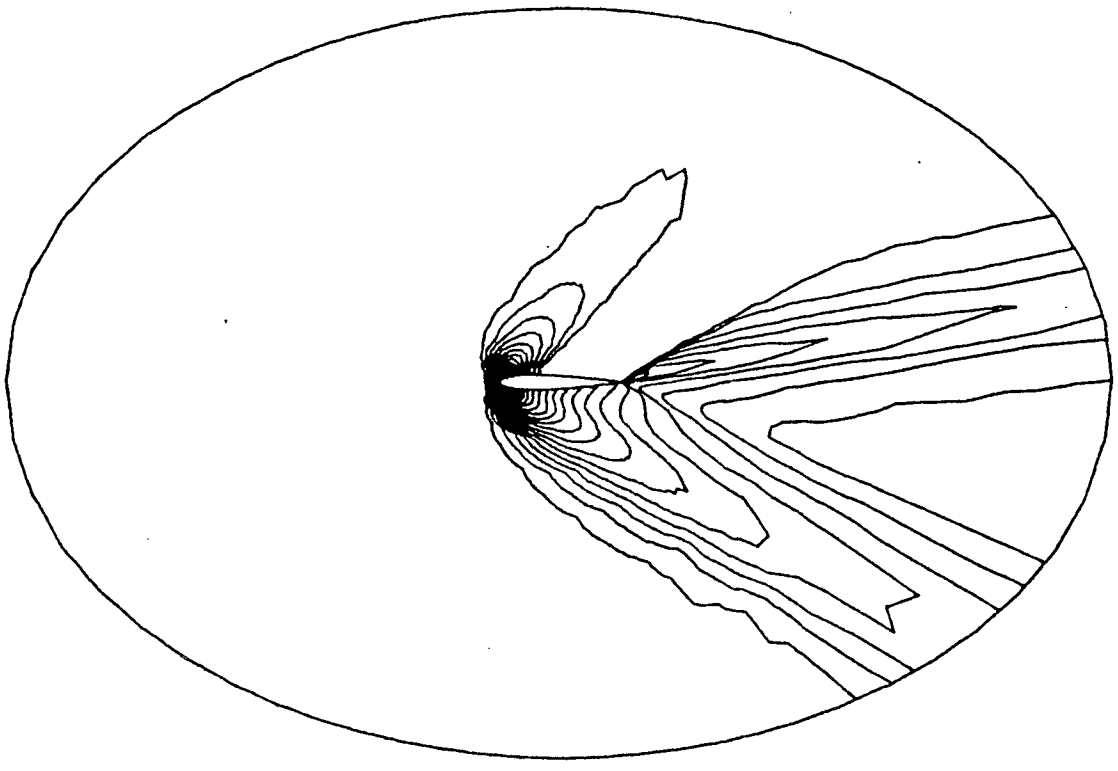


c) Isobar contours

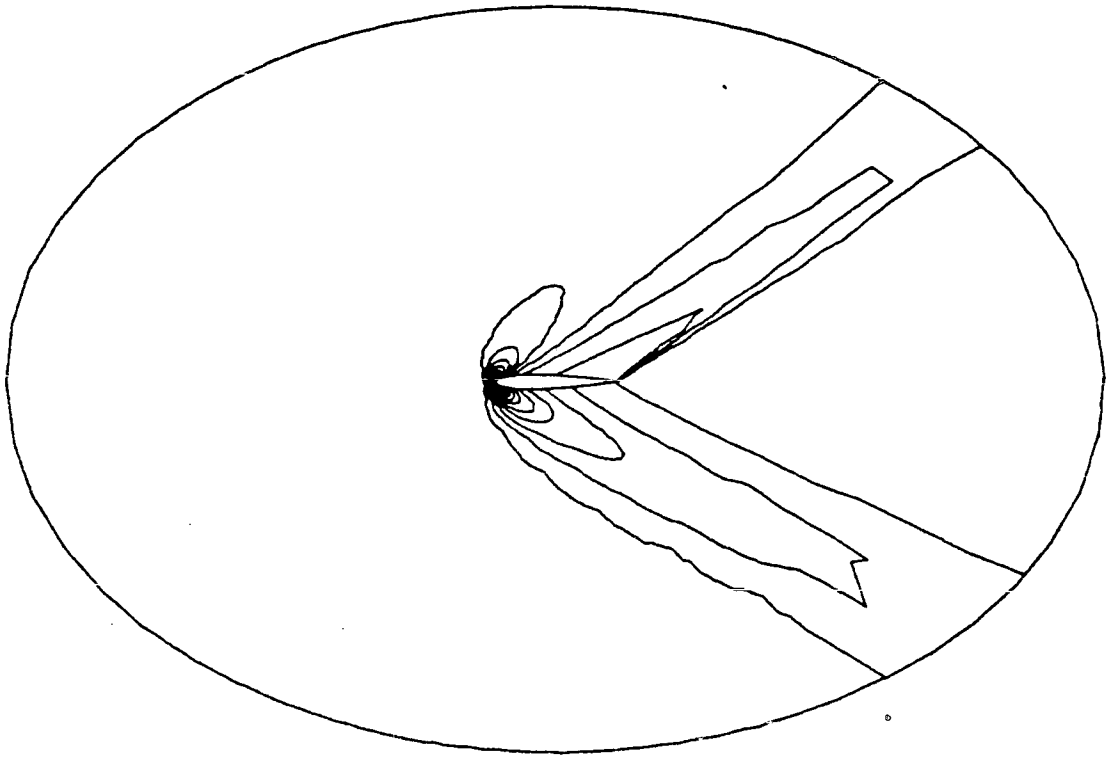
Figure 8 : External flow defined in Figure 8. The results obtained after one enrichment



a) Mesh (1890 points)

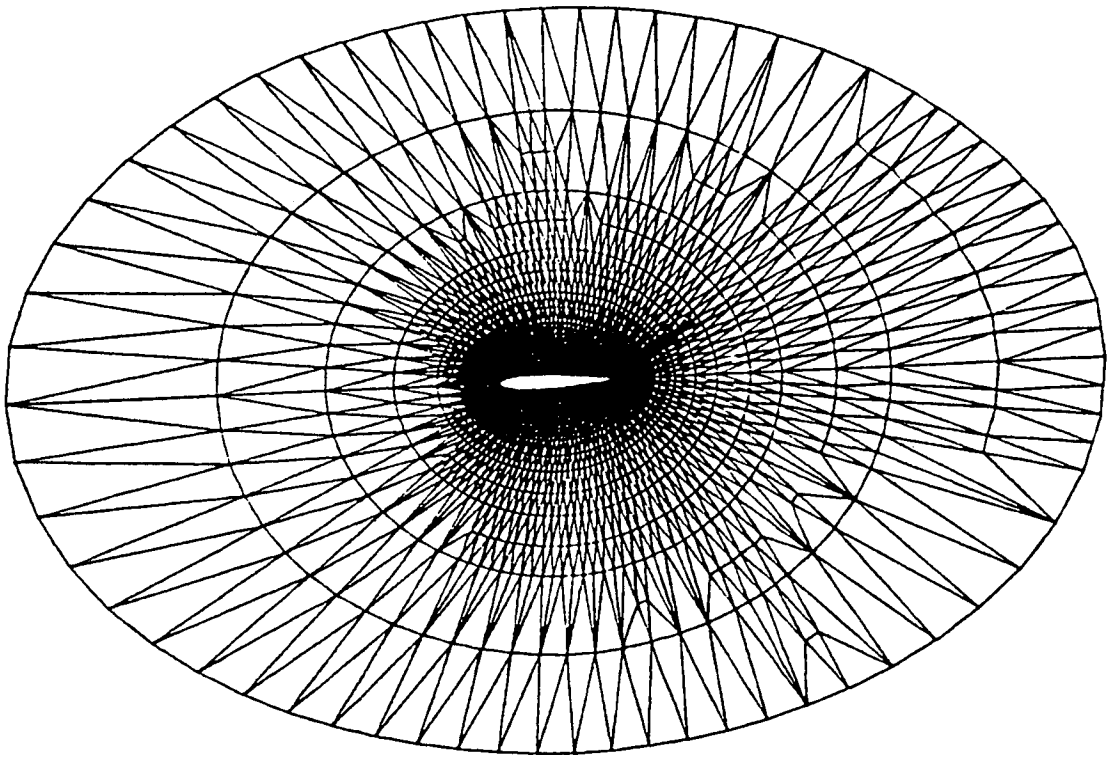


b) Isomach contours

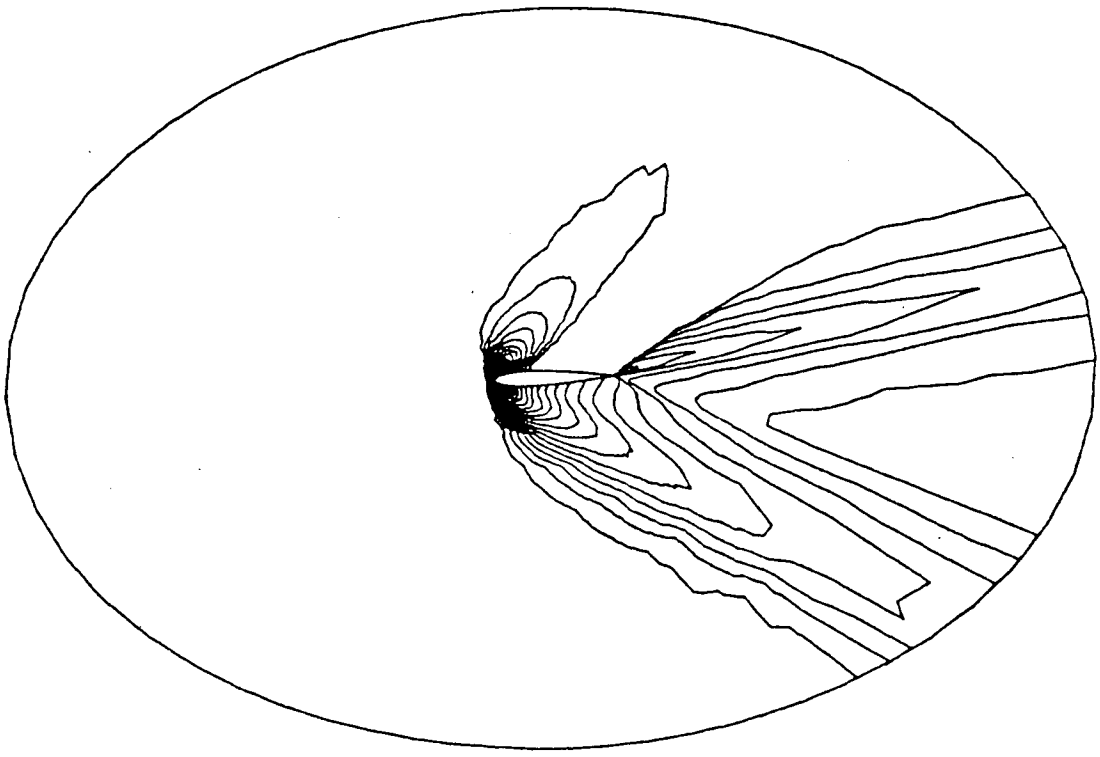


c) Isobar contours

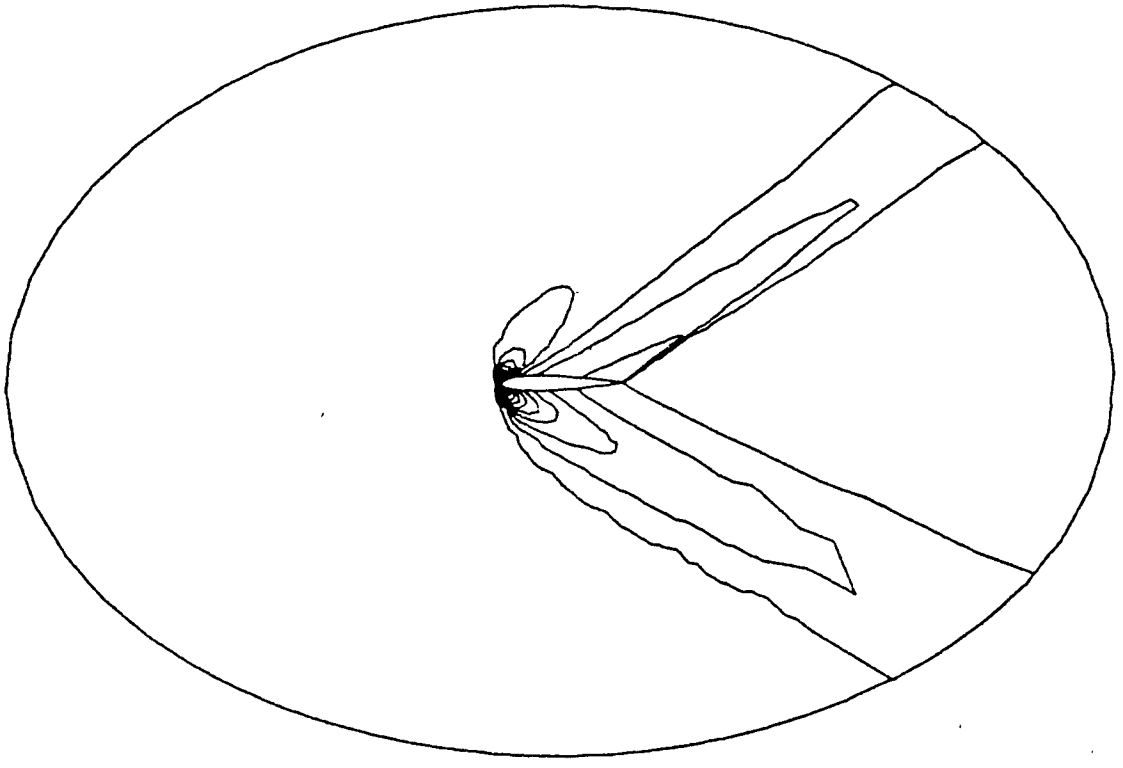
Figure 9 : Second enrichment of test case of Figure 9



a) Mesh (2489 points)

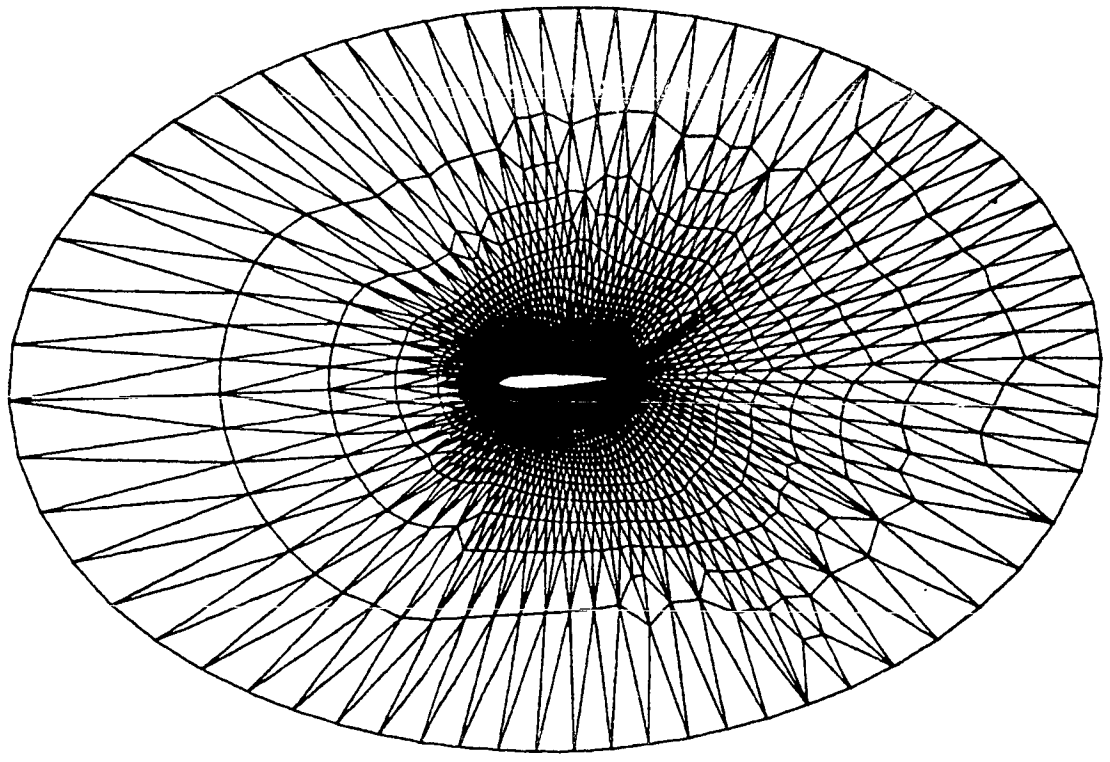


b) Isomach contours

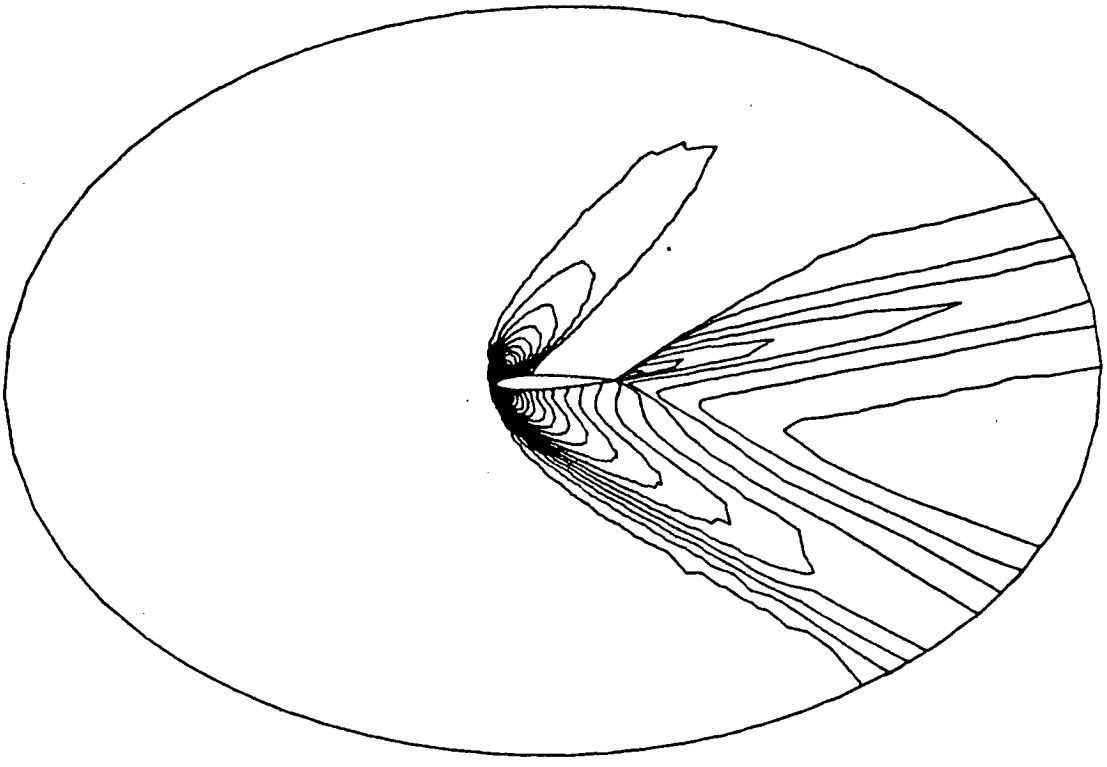


c)Isobar contours

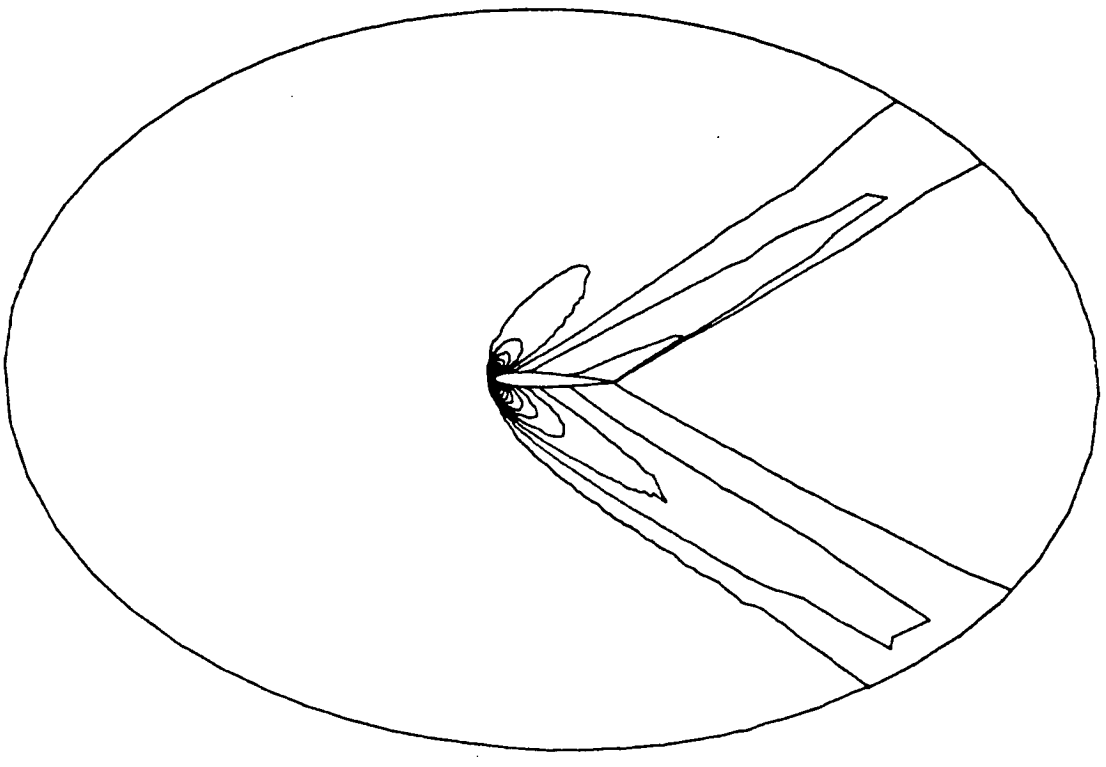
Figure 10 : External flow defined in Figure 8. Mesh motion, with initialization defined by Figure 10.



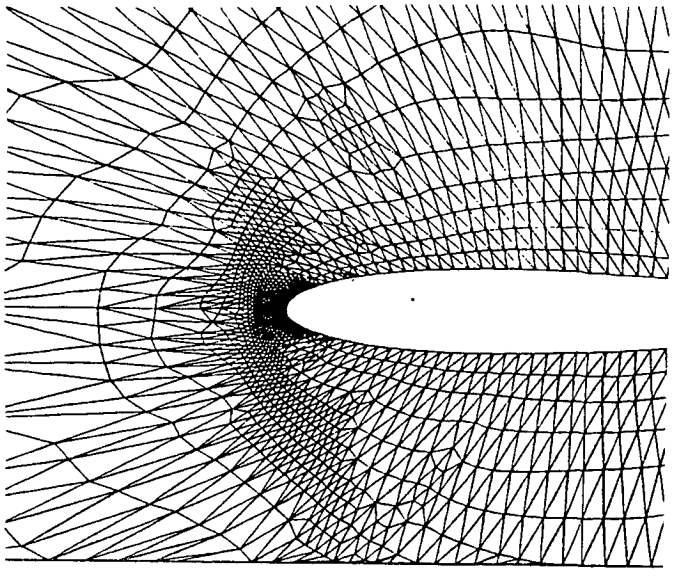
a) Mesh (2489 points).



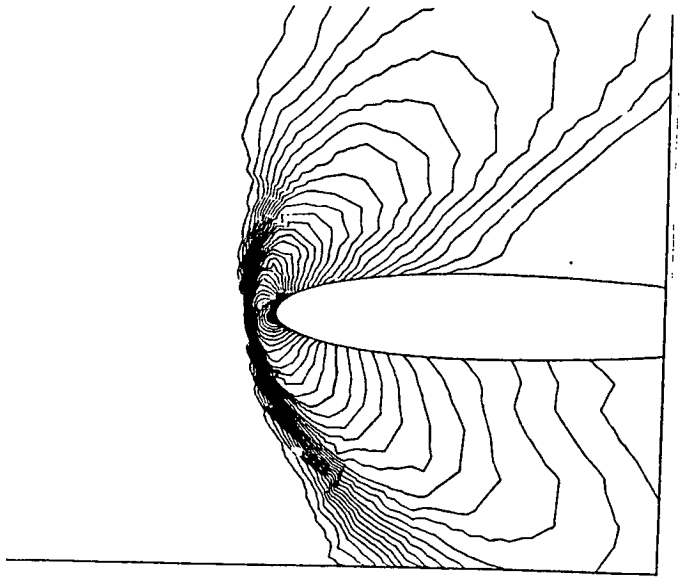
b) Isomach contours



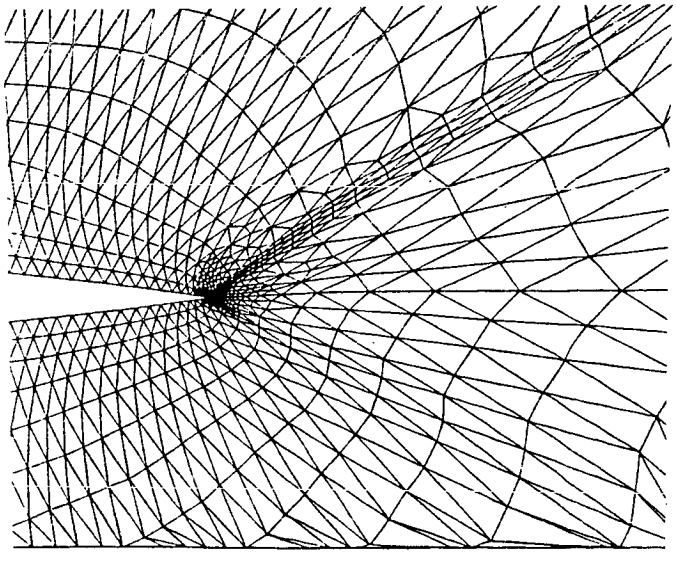
c) Isobar contours



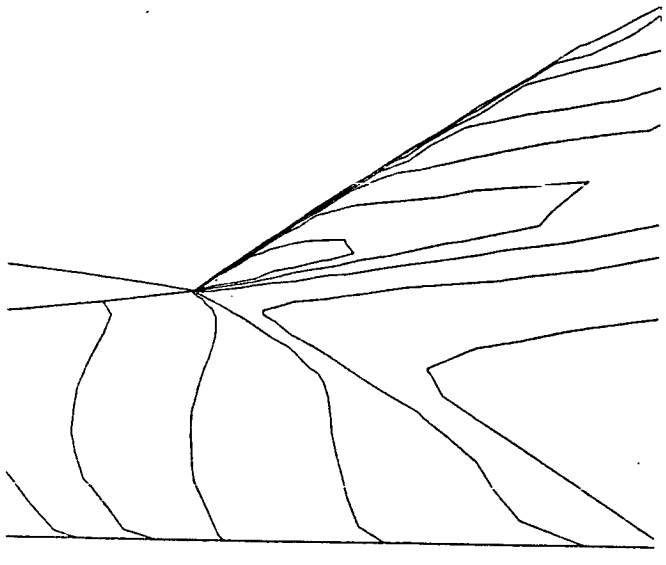
a) Final mesh, front partial view



b) Isomach lines corresponding to a)

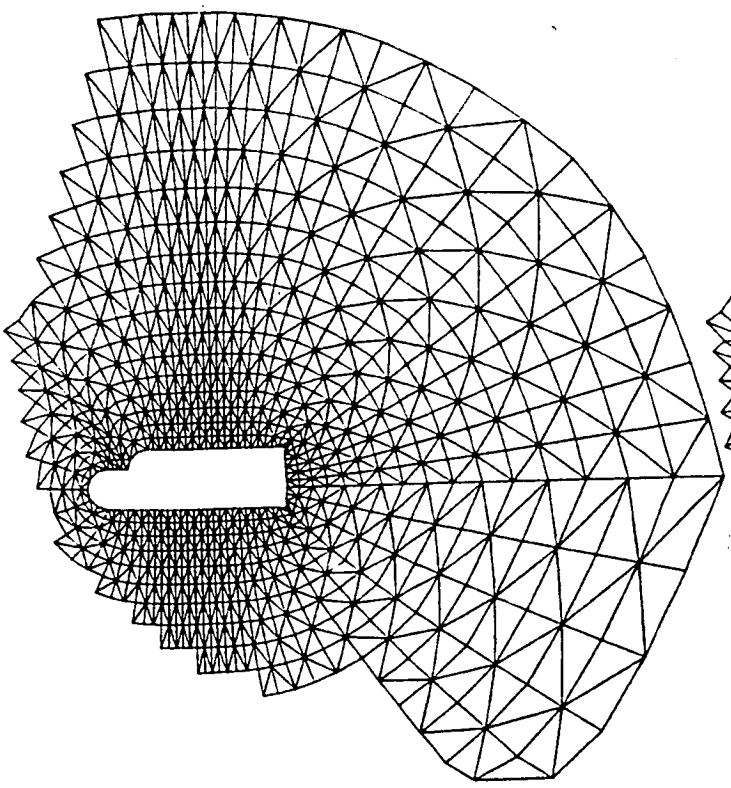


c) Final mesh, rear partial view

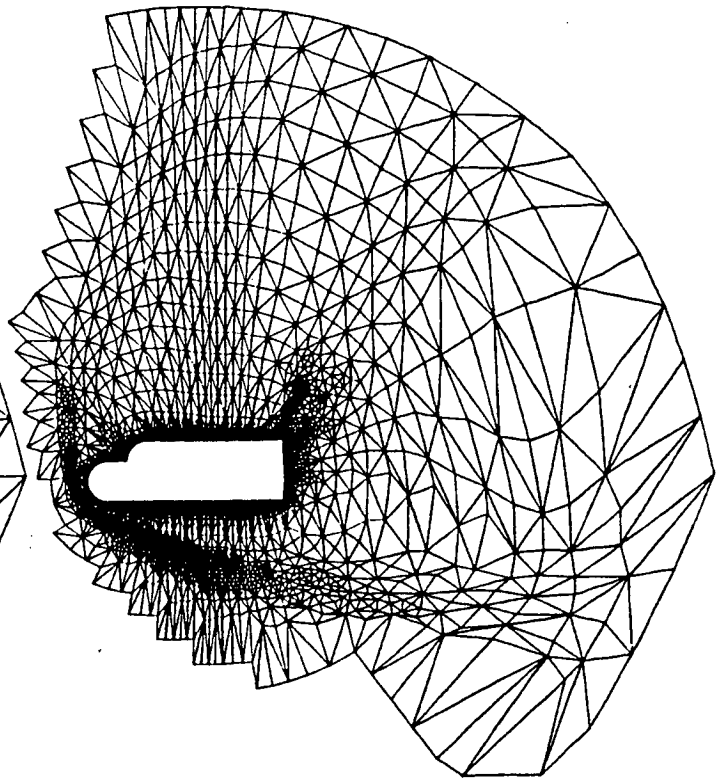


d) Isomach lines corresponding to c)

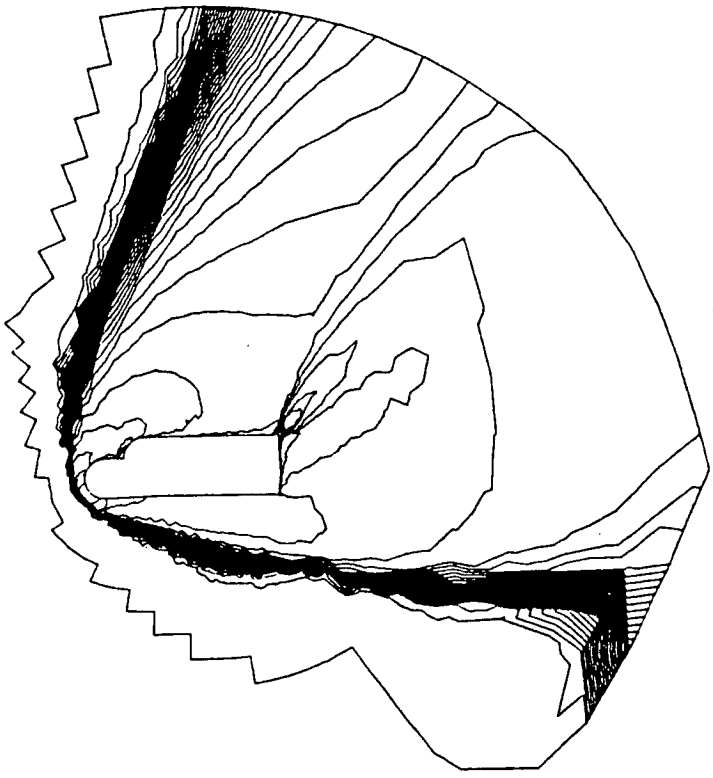
Figure 11 : Blow-up of the final results of Figure 11 (2489 points)



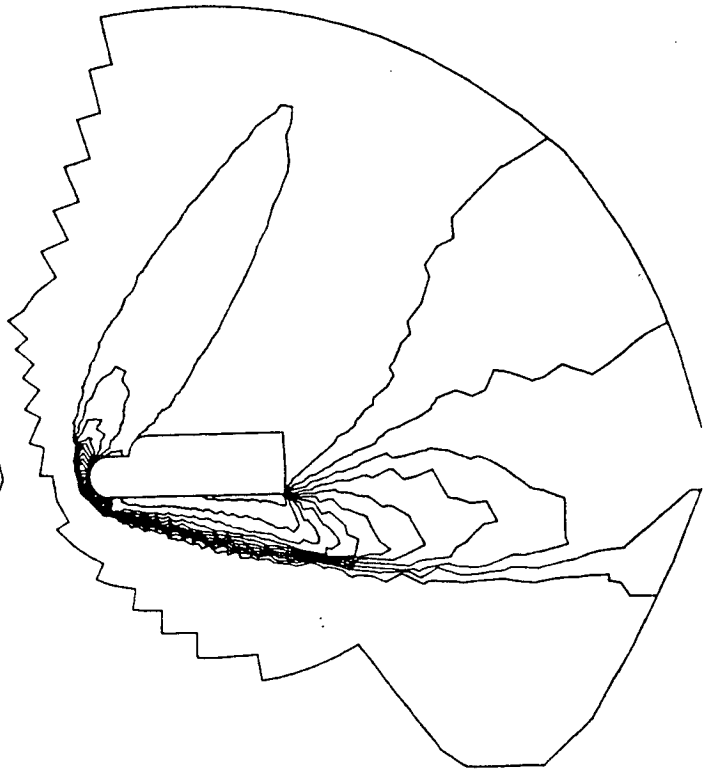
a) Initial mesh (874 points)



b) Final mesh (2377 points)

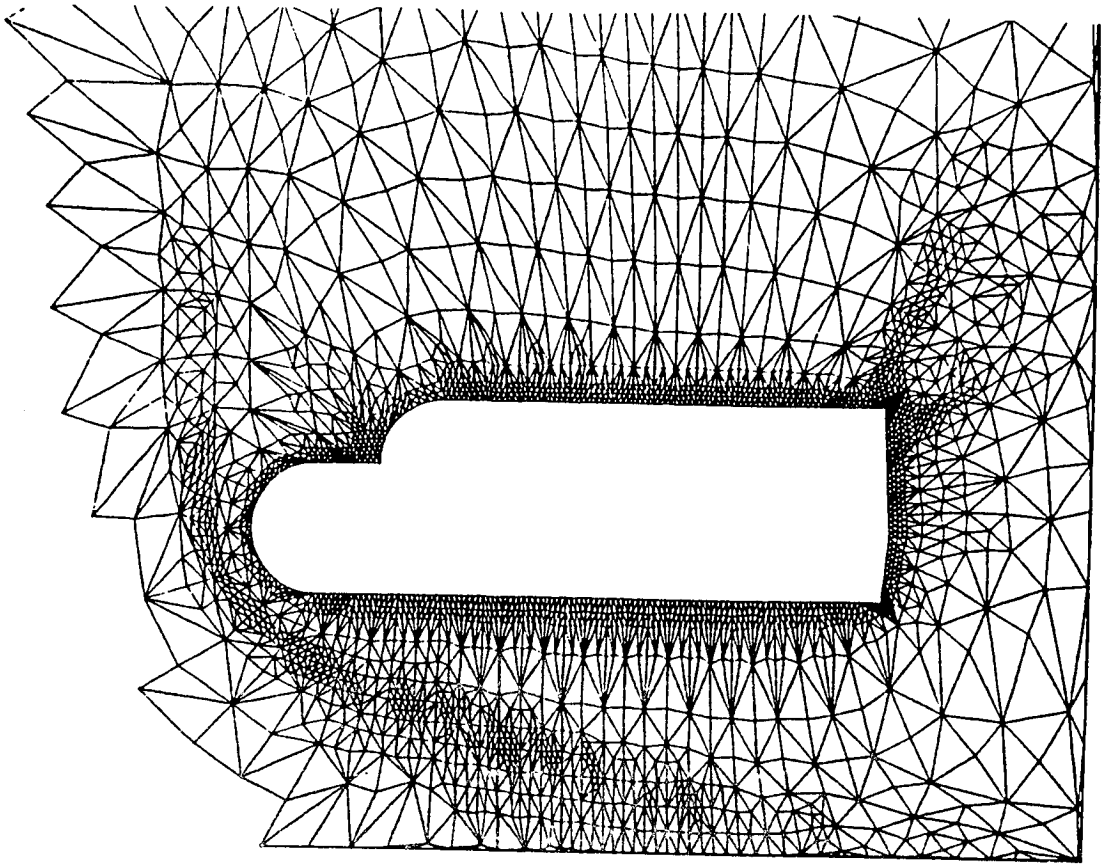


c) Isomach contours

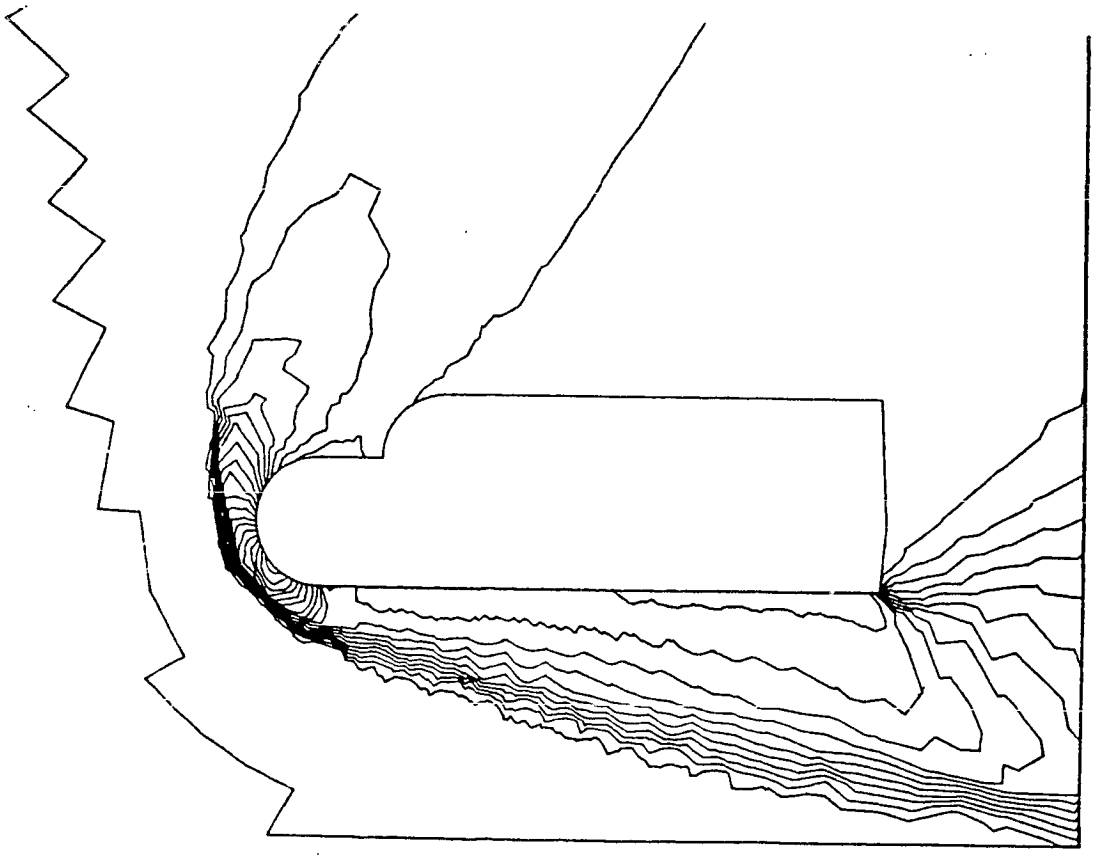


d) Isobar contours

Figure 12 : Flow past a space aircraft (M = 25, $\alpha = 40°$).

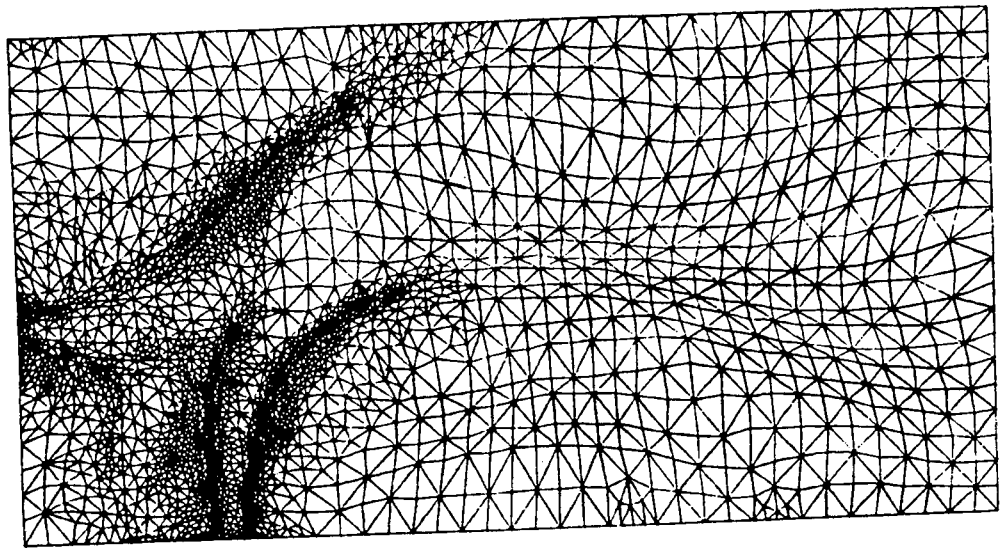


a) Blow-up of the final mesh

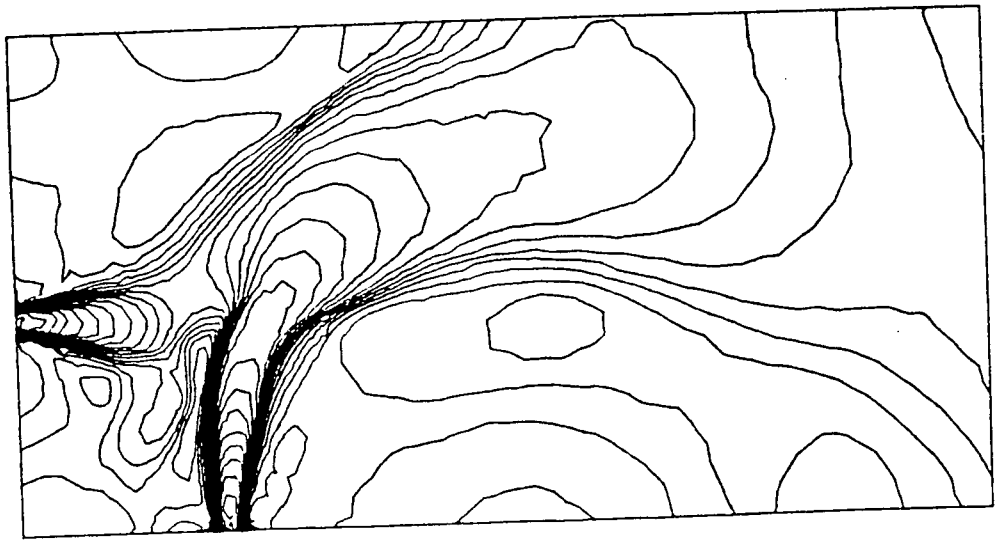


b) Blow-up of the isobar contours

Figure 13 : Flow past a space aircraft M = 25, $\alpha = 40°$)



a) Final mesh (2438 points)



a) Isomach contours

Figure 14 : Interacting jets

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique