INT. J. NUM. METH. HEAT FLUID FLOW, VOL. 3, 63–77 (1993)

AN IMPLICIT FLUX-DIFFERENCE SPLITTING METHOD FOR SOLVING THE EULER EQUATIONS ON ADAPTIVE TRIANGULAR GRIDS

J.-Y. TRÉPANIER, M. REGGIO AND D. AIT-ALI-YAHIA

Department of Mechanical Engineering, École Polytechnique de Montréal, CP 6079, Succ A, Montréal, Québec, H3C 3A7, Canada

ABSTRACT

An implicit method for the solution of transonic flows modelled by the time-dependent Euler equations is presented. The method is characterized by a robust linearization for first- and second-order versions of Roe's flux-difference splitting scheme, an implicit treatment of the boundary conditions and the implementation of an adaptive grid strategy for global efficiency. The performance of the method is investigated for the GAMM test circular-arc bump configuration and for the RAE 2822 aerofoil.

KEY WORDS Transonic flows Roe flux-difference scheme RAE 2822 aerofoil

INTRODUCTION

Propelled by the need for advanced elements to be used as primary design tools in aerodynamics, the CFD community has produced several codes based on the Euler equations for the simulation of compressible flows. Among the modern alternatives, there is a proliferation of schemes using unstructured triangular grids[1–3], which have opened up new areas for flow simulation around complex geometries. This trend is mainly driven by the intrinsic ability of triangular grids to deal with general geometries and for the natural setting that they provide for the implementation of adaptive grid-flow coupling procedures. Meshes built-up of simplexes can be locally enriched or coarsened without affecting a large region of the domain. This is an appropriate way to achieve a suitable degree of accuracy with an affordable number of grid nodes.

Despite the fact that much progress has been made in the implementation of the attractive grid-flow coupling, the question of the efficiency of such a coupling needs to be investigated further. Efficiency is not only related to the mesh layout, but also depends heavily on the time integration scheme. It is well known that in terms of the allowable time step implicit schemes are less restrictive than explicit algorithms. Implicit procedures have long been implemented in structured grid systems within the ADI framework[4]. For unstructured grid systems, there are a number of approaches: for example, point implicit algorithms, where a local matrix is solved for each element, have been suggested[5,6], line implicit algorithms have been described[7], which are a kind of generalization of the ADI algorithms; and implicit/explicit procedures have been proposed[8], where some part of the domain is treated implicitly while the rest is integrated explicitly.

In a comparative study of 2-D Euler solvers applied on triangular grids, Whitaker *et al.*[9] have found that a fully implicit approach, coupled with an LU decomposition, seems one of the most

0961–5539/93/010063–15$2.00
© 1993 Pineridge Press Ltd

Received April 1992
Revised September 1992

promising methods for achieving the steady-state solution. One of the alternatives investigated by these authors for solving the underlying equations is Roe's widely adopted flux-difference splitting method, either in its original first-order version[10] or modified second-order versions[2]. However, it has been found that in many cases the implicit second-order extension of Roe's scheme appeared unstable and that a converged solution could not be obtained. There was no full explanation for this behaviour, however. The authors attribute the problems to the non-linearity of the limiter or to the inconsistency between the right and left members of their system of equations.

In light of these findings, this work is aimed at addressing two objectives. The first one is to develop, in a finite-volume context with cell-centred flow properties, robust fully implicit first- and second-order versions of Roe's scheme. Barth's[2] algorithm will be followed for the second-order extension. The second, is the investigation of the efficiency of the coupling of such an implicit procedure with a grid adaptation mechanism.

To achieve the first objective, an implicit discretization and linearization in time is suggested, where all the Jacobian matrices involved are replaced by Roe's average matrix. The resulting equations are assembled through the entire domain, including a consistent implicit treatment of the boundary conditions represented via a system of equations. This global sparse-matrix system has been tackled using a direct LU solver.

To achieve the second objective, a grid adaptation procedure has been implemented where a sensor based on the gradient of the flow variables is used as an *a posteriori* error estimate, and where, the remeshing is performed by means of local enrichment and coarsening of the grid. A simple methodology involving a sequence of solutions on a set of adapted and increasingly finer grids is proposed and investigated.

The performance of the implicit scheme and of the adaptive methodology is evaluated by a comparison with the well-known GAMM benchmark[11]. This case enabled analysis of the implicit *versus* the explicit methodology, comparison of implicit and explicit treatments of the boundary conditions, study of the influence of the CFL number on the rate of convergence and analysis of the benefits of the adaptive grid procedure on the quality of the solution and on the global performance. Finally, the coupled grid adaptation-flow solver method was applied to the computation of the flow field around the RAE 2822 aerofoil.

FLOW EQUATIONS

In 2-D, the integral form of the unsteady Euler system can be written as:

$$\frac{\partial}{\partial t}\iint_{\Omega} Q \, d\Omega + \int_{\partial\Omega} (f\vec{i} + g\vec{j})\cdot\vec{n} \, dl = 0 \tag{1}$$

with:

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix}, \quad f = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{bmatrix}, \quad g = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \end{bmatrix} \tag{2}$$

In these equations ρ stands for the density, p for the pressure, u and v for the two velocity components, E for the total energy per unit volume, and $\vec{n}$ for the outward unit normal vector.

This system is closed with the equation of state for a perfect gas, that is:

$$p = (\gamma - 1)\left[E - \frac{(\rho u)^2 + (\rho v)^2}{2\rho}\right] \tag{3}$$

where γ is the ratio of specific heats.

NUMERICAL SCHEME

Spatial discretization

Since the domain is subdivided into triangular elements, the semidiscrete counterpart of (1) is:

$$\Omega_e \frac{\partial Q_e}{\partial t} = -\sum_{s=1}^{3} F_s \Delta l_s \tag{4}$$

where Ω_e denotes the volume of the cell and F_s a numerical approximation of the normal flux crossing a face s with side length Δl_s.

The expression for the flux F_s at a given interface s using Roe's scheme is:

$$F_s(Q_L, Q_R) = \tfrac{1}{2}[F(Q_L) + F(Q_R) - |\tilde{A}|(Q_L - Q_R)] \tag{5}$$

where $\tilde{A}$ represents Roe's linearized Jacobian matrix[10] and the subscripts L and R indicate the left and right states which share a face s. For a first-order scheme, the left and right states are simply the piecewise constant values in the adjacent cells of each elementary face. For the second-order extension, the piecewise constant solution is replaced by a linear one, according to the relation:

$$Q(x, y)_A = Q(x_0, y_0)_A + \Phi_A \nabla Q_A \cdot \Delta r \quad \text{with} \quad \Phi_A \in [0, 1] \tag{6}$$

where ∇Q_A represents the constant gradient of the solution on each cell and Δr the local coordinates of a point within the element. In this expression, the parameter Φ_A is a slope limiter used to avoid new maxima or minima[2]. For the current implementation, the gradient is evaluated by considering the triangle B–C–D illustrated in *Figure 1*. Additional details concerning this step can be found in References 2 and 14.

Temporal discretization

The steady-state solution of the hyperbolic system is obtained by an implicit formulation, with the discretization of the time-dependent term based on a backward first-order difference. Accordingly, (4) is expressed as:

$$\Omega_e \frac{\Delta Q_e}{\Delta t} = R_e^{m+1}(Q_e, Q_{nb}) \tag{7}$$

where $\Delta Q_e = Q_e^{m+1} - Q_e^m$ denotes the increment in time of the cell-centred values, Δt the time step, and R_e^{m+1} the residual given by the summation of the fluxes over the faces of the triangular cell. Along each face, the flux is given by (5), with the e and nb subscripts corresponding to the L and R subscripts.

To solve the non-linear algebraic equation system resulting from (7), a linearization in time about the time level m is applied, yielding:

$$\Omega_e \frac{\Delta Q_e}{\Delta t} = \left[R_e^m(Q_e, Q_{nb}) + \frac{\partial R}{\partial Q}\Delta Q\right] \tag{8}$$

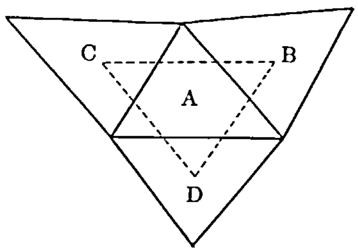

Figure 1 Integration path for the calculation of the gradients

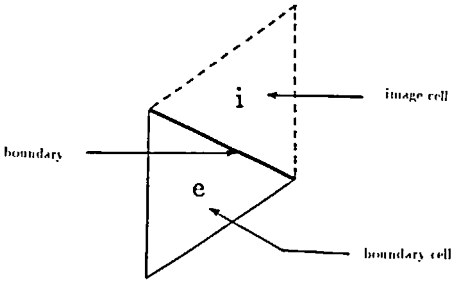

Figure 2 Definition of image cells at the boundaries

It can be appreciated that for a very large time step, the left-hand side term becomes negligible and the procedure reduces to Newton's method. Introducing the flux expression given by (5), written for the time level m, (8) becomes:

$$\Omega_e \frac{\Delta Q_e}{\Delta t} = \left\{ R_e^m - \frac{1}{2}\sum_{s=1}^{3} [A_e^m \Delta Q_e + A_{nb}^m \Delta Q_{nb} - |\tilde{A}_s^m|(\Delta Q_{nb} - \Delta Q_e)]\Delta l_s \right\} \tag{9}$$

with:

$$A_e^m = \frac{\partial F(Q_e)}{\partial Q_e} \qquad A_{nb}^m = \frac{\partial F(Q_{nb})}{\partial Q_{nb}} \tag{10}$$

At this point, and consistent with Roe's scheme, a further simplification is introduced, and A_e^m and A_{nb}^m are approximated by $\tilde{A}_s^m$. In the second-order case, linearly reconstructed values are used in the computation of $\tilde{A}_s^m$. In so doing, the final implicit expression for the computation of the temporal increment ΔQ is:

$$\left\{ \frac{2\Omega_e}{\Delta t} I + \sum_{s=1}^{3} [\tilde{A}_s^m + |\tilde{A}_s^m|]\Delta l_s \right\} \Delta Q_e + \sum_{s=1}^{3} [\tilde{A}_s^m - |\tilde{A}_s^m|]\Delta l_s \Delta Q_{nb} = 2R_e^m \tag{11}$$

This equation relates four spatial variables (ΔQ_e and its three neighbours ΔQ_{nb}) per element, each of which includes the flow variables, $\Delta\rho$, $\Delta(\rho u)$, $\Delta(\rho v)$ and ΔE.

Boundary conditions

A simple method for imposing boundary conditions for cell-centred schemes is to create image cells at the boundaries where the flow properties are initialized according to the type of boundary, and then, to apply the Riemann solver to compute the flux across these boundaries. This treatment makes it possible to take into account the incoming and outgoing characteristics. This approach is very easily implemented in an explicit scheme where the flow properties in the image cells are set at the beginning of each time step. For implicit schemes, this (explicit) treatment can also be used to set the flow properties in the image cells at the beginning of each implicit iteration. However, if large time steps are used, there will be a considerable time lag between the solutions and the boundary conditions. This may be a source of instabilities which can be avoided by applying a different treatment.

In this study, a fully implicit version of image cell boundary conditions has been developed. Essentially, for each type of boundary, a system of equations is used to represent the boundary conditions. This, written for each boundary side, gives rise to additional equations that are included in and solved with the global system. In the second-order version, linearly reconstructed values are used to ensure a proper imposition of the boundary conditions.

Since the scheme is written in delta form, the boundary system sets the relations between the increments of the variables in the image cell with those in the boundary cells. This system can be written as:

$$M_e \Delta Q_e + M_i \Delta Q_i = 0 \tag{12}$$

The subscripts i and e denote the image and element cells adjacent to a given boundary, as illustrated in *Figure 2*. The matrices M_e and M_i depend on the type of boundaries and will be selected such that the correct number of characteristics is imposed at each boundary.

Because some boundary conditions are applied in terms of the non-conservative variable increments $\Delta W = (\rho, \rho u, \rho v, p)$, the relation between these and the conservative variable increments, $\Delta Q = (\rho, \rho u, \rho v, E)$, must be first established. It can easily be found that the relation between these two sets of variables can be represented by:

$$\Delta Q = T \Delta W \tag{13}$$

where T is given by:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\frac{1}{2}(u^2 + v^2) & u & v & \frac{1}{\gamma - 1} \end{bmatrix} \tag{14}$$

and in which the pressure increment,

$$\Delta p = (\gamma - 1)[\Delta E + \tfrac{1}{2}(u^2 + v^2)\Delta\rho - u\Delta(\rho u) - v\Delta(\rho v)]$$

obtained from the equation of state, has been used.

Solid wall. For a solid wall, the required boundary condition is to impose a zero flux through the wall. This can easily be done by setting density, energy and tangential velocity increments to the image cell equal to those of the adjacent boundary cell, while the normal velocity increments to be imposed are of equal modulus but of opposite sign. The coupling matrices can thus be written as:

$$M_e = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & k_x & k_y & 0 \\ 0 & -k_y & k_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad M_i = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & k_x & k_y & 0 \\ 0 & k_y & -k_x & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \tag{15}$$

where k_x and k_y indicate the components of the outward normal unit vector at the boundary.

Subsonic inflow. At a subsonic inflow, three characteristics come from the outside of the domain and one comes from the interior[12]. To respect this characteristic's behaviour, the value of the pressure at the image cell assigned is that of the adjacent interior cell, while the remaining properties are computed by assuming an isentropic expansion from a hypothetical reservoir to the inlet. The flow angle must also be prescribed to ensure the completeness of the system of equations.

After some tedious manipulation, the coupling matrices can be obtained as follows:

$$M_e = \begin{bmatrix} \frac{q^2}{2} & -u & v & 1 \\ (\gamma-1)\frac{q^2}{2}\cos\alpha & -(\gamma-1)u\cos\alpha & -(\gamma-1)v\cos\alpha & (\gamma-1)\cos\alpha \\ (\gamma-1)\frac{q^2}{2}\sin\alpha & -(\gamma-1)u\sin\alpha & -(\gamma-1)v\sin\alpha & (\gamma-1)\sin\alpha \\ \frac{q^2}{2} & -u & v & 1 \end{bmatrix}_e \tag{16}$$

$$M_i = \begin{bmatrix} -\frac{\gamma}{\gamma-1}p^{\frac{\gamma-1}{\gamma}} & 0 & 0 & 0 \\ -qup^{\frac{1}{\gamma}} & qp^{\frac{1}{\gamma}} & 0 & 0 \\ -qvp^{\frac{1}{\gamma}} & 0 & qp^{\frac{1}{\gamma}} & 0 \\ -\frac{q^2}{2} & u & v & -1 \end{bmatrix}_i \tag{17}$$

where q denotes the velocity modulus and α is the inflow angle.

Subsonic outflow. In the case of a subsonic outlet, only the static pressure is imposed, while the remaining variables are extrapolated, that is:

$$\begin{bmatrix} \rho \\ \rho u \\ \rho v \end{bmatrix}_i = \begin{bmatrix} \rho \\ \rho u \\ \rho v \end{bmatrix}_e \tag{18}$$

This system of equations, including the pressure (a primitive variable), can be written for the variable increments in time, $\Delta W = [\Delta\rho, \Delta\rho u, \Delta\rho v, \Delta p]^{\mathrm{T}}$, as:

$$\Delta W_i = D\Delta W_e \tag{19}$$

with:

$$D = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{20}$$

Finally, using (13), the boundary-condition system for the conservative variables becomes:

$$M_i\Delta Q_i + M_e\Delta Q_e = 0 \tag{21}$$

with $M_i = (T_i D)^{-1}$, or explicitly:

$$M_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\frac{1}{2}(u^2+v^2) & u & v & 0 \end{bmatrix}_i^{-1} \tag{22}$$

Note that it is not necessary to inverse the above matrix, and that the matrix M_e in (21) is the identity matrix negated.

Supersonic inflow–outflow. For a supersonic inflow, the properties in the image cell are all imposed. Because these are known values, they are not included in the global matrix system. For a supersonic outflow, all the flow properties in the mirror cell are extrapolated from the adjacent interior cell. The increments are set equal to the increments of its neighbour cell and the coupling matrices are thus simply given by:

$$M_e = -M_i = I \tag{23}$$

Assembly and solution

As mentioned earlier, this cell-centred triangular discretization implies four elementary (4 × 4) matrices per element. Therefore, the assembly of all the N elements, including the exterior image cells, generates a (4 × N) matrix system, and the global system, which is sparse because of the intrinsic character of the unstructured grid, can be written as:

$$C\Delta \boldsymbol{Q} = \boldsymbol{R} \tag{24}$$

where C denotes the global (4 × N) matrix, $\boldsymbol{R}$ the residual and $\Delta \boldsymbol{Q}$ the vector correction of the flow variables, i.e., $[\Delta\rho, \Delta(\rho u), \Delta(\rho v), \Delta E]^{\mathrm{T}}$.

The method adopted to solve the system is based on a direct LU decomposition[13]. As a preprocessing step, element renumbering is applied to minimize the matrix profile.

Due to the non-linear nature of the problem, the system represented by (24) has to be solved iteratively to converge to the steady-state solution. The main cost in the LU method is given by the decomposition step, particularly when the grid is fine and many equations are involved. To improve the global performance of the iterative process, the residual evolution was monitored, and after reaching a preestablished level[14] (appropriate values were found after numerical experimentation), the global matrix C of (24) was frozen. This means that thereafter the solution simplifies to backward triangular substitutions plus residual updating.

The convergence of the method was analysed by means of the $\mathscr{L}^2$ norm of the residual $\boldsymbol{R}$, weighted by the total number of equations:

$$\|R\| = \frac{\left(\sum_{i=1}^{4N} r_i^2\right)^{\frac{1}{2}}}{4N} \tag{25}$$

where r_i denotes the residual of the ith equation.

GRID ADAPTATION

The underlying idea of grid adaptation is to use the minimum amount of elements for the discretization for a given precision. This goal can be achieved by applying some convenient type of error estimation. In the following, we will give a general description of the mechanism used in this work.

Adaptive feedback loop

The proposed adaptive feedback loop is very simple. A solution is first computed on an initial coarse grid. Then, *a posteriori* treatment of the solution is performed by estimating the error as being proportional to the difference between reconstructed piecewise linear and initial constant solutions. After this, a new grid size distribution can be obtained. In an attempt to equidistribute the error, the number of triangles is allowed to increase by a user-specified factor (usually 1.5) and a new solution is computed on the adaptively regenerated grid. This remeshing process is stopped either when the required accuracy, or when a maximum number of allowed triangles, is reached.

It will be noted that due to the intrinsic nature of compressible flow solutions, very high ratios of minimum and maximum required areas are obtained. This results in extremely small triangles in regions of high gradients of the solution. To overcome this problem, maximum and minimum triangles size limiters have been imposed as part of a global strategy. The limits on the smallest and largest triangle areas are specified in tems of a fraction of the initial grid, and are thus locally defined.

The remeshing technique

The algorithm which uses local grid refinement[15,16], coarsening and cure techniques, has been followed for the remeshing process. Such a procedure is very efficient in cases where very few triangles need to be refined or coarsened at each remeshing step. The refinement is obtained through triangle subdivision, where a triangle is branched into two triangles by cutting it on its longest side. This process is performed on all the triangles requiring this option, and the reconnection of unmatched sides is performed last. In the case of a side located on a curved boundary, the new node inserted on that side is relocated on the boundary, so as to remain consistent with the geometric representation. The coarsening is performed through node removal. A node is selected to be removed if all its neighbouring triangles are to be coarsened. This node removal leaves an open polygon, which is then remeshed. A further advantage of such a remeshing technique is that the solution can be transferred simultaneously from the old to the new grid as the grid is adapted. There is no need for a subsequent interpolation step and the design of a conservative transfer operator is relatively easy.

RESULTS

GAMM-test case

A study of the convergence rate was carried out for the well-known GAMM-test case[11] for transonic flows. This consists of a parallel-walled channel with a bump along the lower wall. The free stream flows from left to right with an inlet Mach number of 0.85. The convergence tests are conducted on meshes comprising 1067 and 1955 triangles, hereafter called A-grid and B-grid, with a total of 571 and 1030 nodes and 11 and 19 nodes on the bump, respectively. These are shown in *Figures 3* and *5* together with the computed solutions, illustrated in

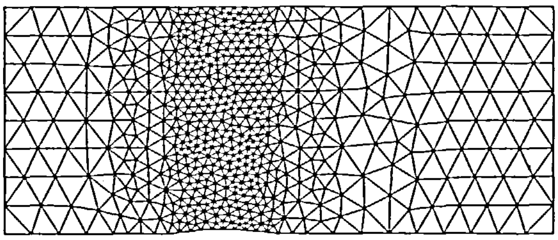
Figure 3 GAMM-test A-grid (1050 elements and 571 nodes)

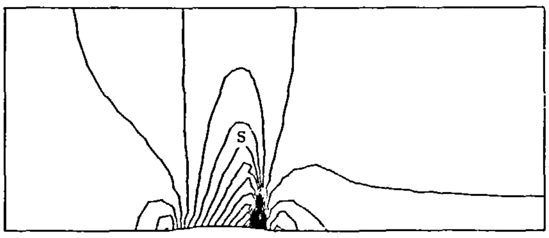

Figure 4 Mach number contours for A-grid (S: Sonic line, $\Delta M = 0.04$)

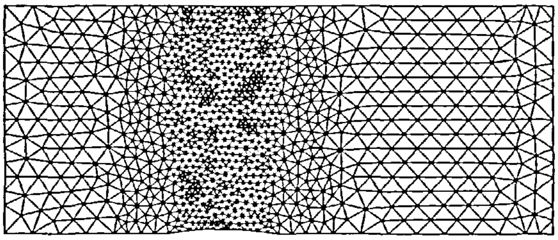
Figure 5 GAMM-test B-grid (1955 elements and 1027 nodes)

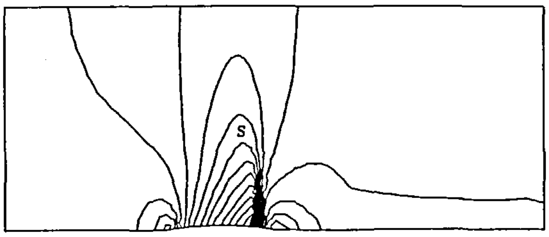

Figure 6 Mach number contours for B-grid (S: Sonic line, $\Delta M = 0.04$)

Figures 4 and *6* by means of iso-Mach contours, using the second-order version of the scheme. In both simulations, the shock is well captured near the bump, but spreads out away from it.

The effect of the CFL number. The properties of the C matrix appearing in (24) depend on the choice of Δt (or the CFL number). The influence of this parameter was therefore studied for the implicit method with Jacobian updating, CFL numbers of 1, 100, 500, 1000, 10,000 and 100,000 were used for both the first- and second-order versions. In each case, the CFL coefficient multiplies the minimum time step given by the Courant–Friedrichs–Lewy criterion, to obtain the actual time step.

In *Figures 7a* and *7b*, the rates of convergence for the first-order scheme, in terms of the number of iterations for the different CFL numbers, are compared for the A- and B-grids respectively. These plots indicate that for a CFL number between 1 and 500 the convergence is slow, but, as the CFL number is increased, convergence is obtained in fewer and fewer iteration steps. In fact, for CFL numbers of 1000 and 10,000 for the A-grid and 10,000 and 100,000 for B-grid, convergence is reached ($\log\|R\| = -8.5$) after approximately, 38 and 31 iterations respectively. Note that the convergence rate does not behave linearly.

The second-order convergence plots are shown in *Figures 7c* and *7d*. The behaviour is now different, since the rate of convergence increases up to a certain limit in the CFL number and then decreases if the CFL number is raised further. For the A-grid, the maximum rate of convergence was found to be around CFL = 1000. However, even if the rate of convergence is lower for CFL = 10,000, a further increment of this parameter (for example, CFL = 100,000 was tried) does not modify the convergence behaviour any more, and divergence is never observed. Fot the B-grid, a similar pattern is found. The best rate of convergence is found to be around CFL = 10,000, after which this rate weakens with a limit represented by the curve for CFL = 100,000.

Although not fully equivalent, a qualitative comparison can be attempted between these convergence results and those presented by Whitaker *et al.*[9] for the fully implicity LU method. In this case, grids with 1005 and 1999 elements, comparable to the A- and B-grids presented here, were used. For the A-type grid and for the first-order scheme, convergence was reached around 80 iterations steps, while for the second-order extension, about 1400 iterations were needed to achieve (approximately) $\log\|R\| = -8.5$. For the B-type grid and for the first-order scheme, the number of iteration steps required to approach the steady-state solution remained almost constant. However, for the second-order scheme, the results do not show convergence.

On the contrary, for the first- and second-order versions of the scheme presented here, convergence was always found with the iteration count not exceeding the value of 60. The robustness of this procedure is attributable to the type of linearization and to the implicitness with which the boundary conditions are imposed.

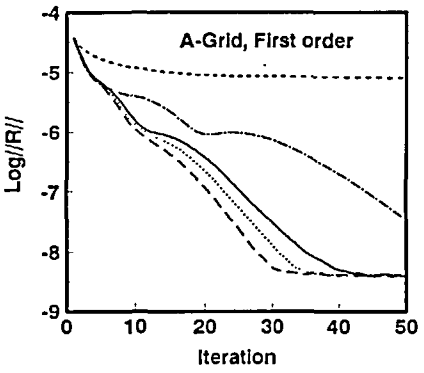

Fig. 6a

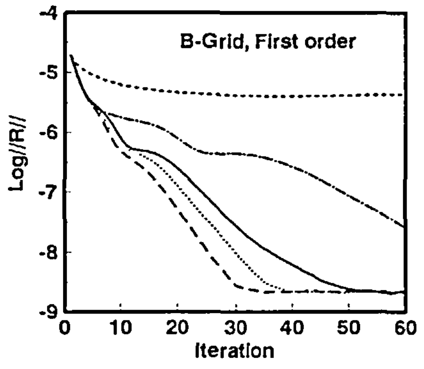

Fig. 6b

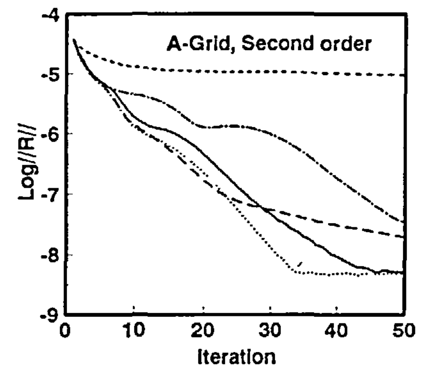

Fig. 6c

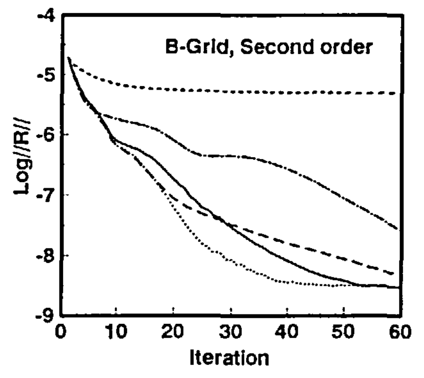

Fig. 6d

	A-Grid	B-Grid
-----	CFL=1	CFL=1
—·—	CFL=100	CFL=100
——	CFL=500	CFL=1000
·········	CFL=1000	CFL=10000
— — -	CFL=10000	CFL=100000

Figure 7 Effect of CFL number on the rate of convergence

Implicit vs. explicit CPU time. Besides the accuracy of the solution, a fundamental question concerning the performance of a method is the CPU time requirement. In this respect, convergence rates between explicit and implicit computations, with and without Jacobian reuse, as a function of CPU time have been studied.

A first series of plots in *Figures 8a* and *8b* show the rates of convergence for all three time integration methods used with the first-order version of the scheme. From these Figures, it can be clearly seen that the best scheme for reaching the steady-state solution is the implicit one with Jacobian reuse. Also, in both Figures, the influence of the standard explicit treatment for the boundary conditions, that is, using time-lagged variables at the boundaries, *versus* the implicit treatment described earlier is presented. These alternatives were evaluated using Jacobian

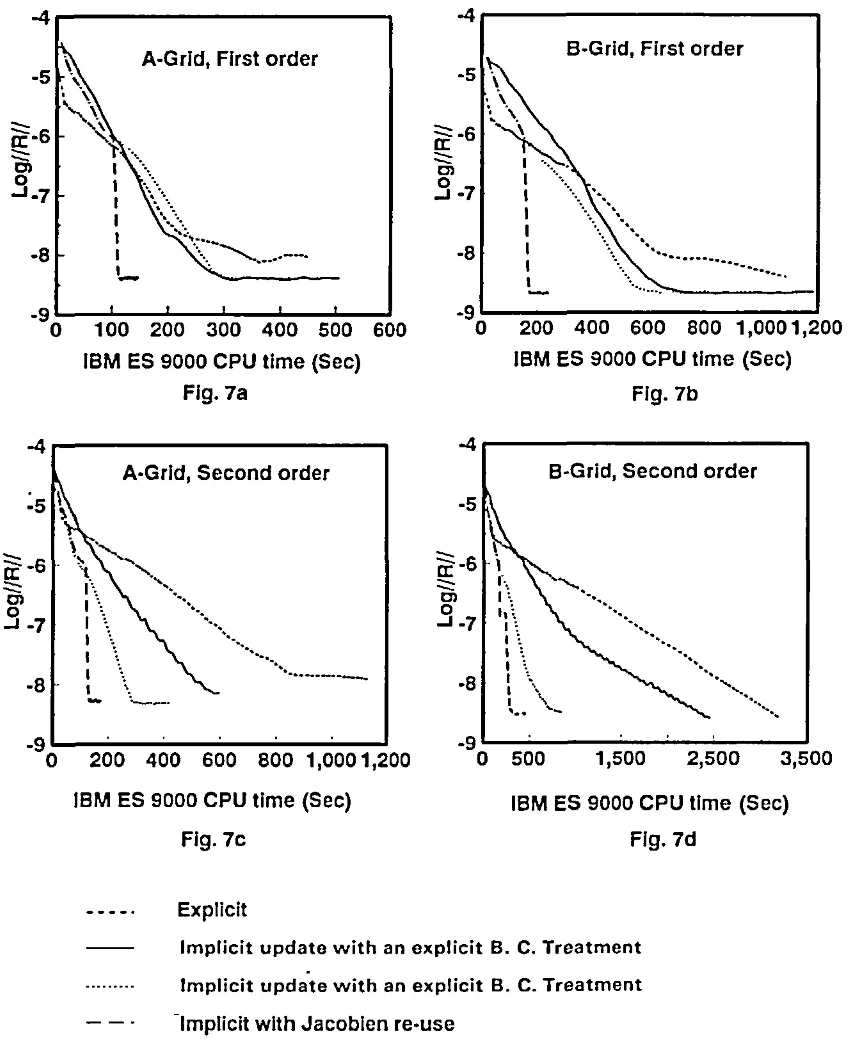

Figure 8 Effect of the temporal discretization strategy on the rate of convergence

updating during calculation. For the A-grid, the difference is negligible, while for the B-grid, a small improvement is obtained when an implicit boundary condition procedure is used.

This kind of investigation was repeated for the second-order scheme. The results depicted in *Figures 8c* and *8d* indicate once again that the implicit method with Jacobian reuse yields the best performance. In particular, for the B-grid, the implicit approach is about seven times faster than its explicit counterpart. For this second-order version of the scheme, the study of implicit *versus* explicit boundary condition treatment reveals that the benefit of implicit handling is more pronounced as the number of elements increases.

Grid adaptation. The quality of the solution computed after the second-order scheme was then studied in terms of the grid adaptation procedure. A flow simulation was performed using grid adaptation and the sequence of grids generated is depicted in *Figure 9.* This sequence of grids indicates that coarse elements arise in regions of weak gradients, while small triangles occur

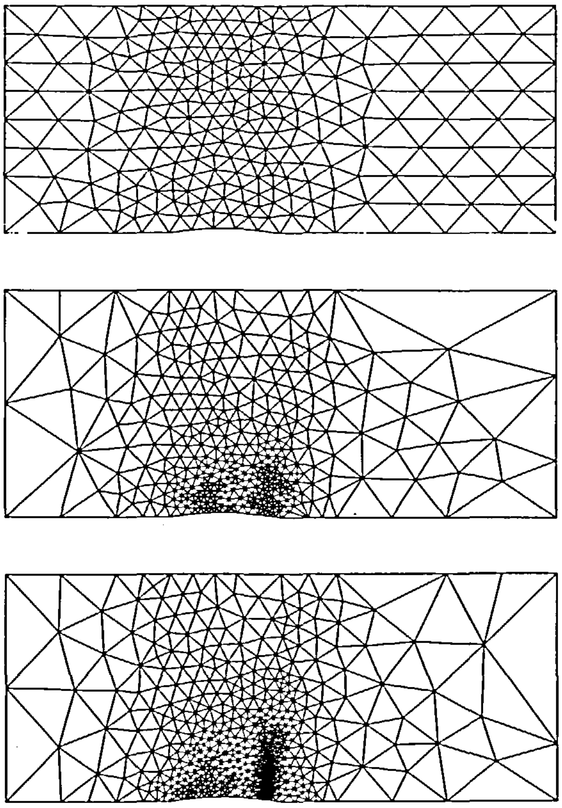
Figure 9 Sequence of meshes using grid adaptation

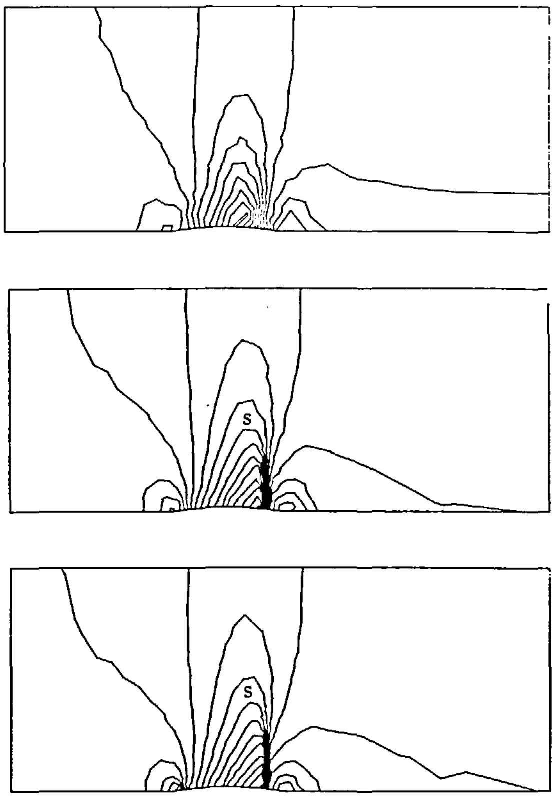

Figure 10 Mach number contours for the meshes of *Figure 8* (S: Sonic line, $\Delta M = 0.04$)

near the shock. The corresponding solutions are illustrated by means of Mach number contours in *Figure 10*. From this it can be seen that the shock narrows as the grid is adapted and that it remains narrow away from the bump. This behaviour reveals that the enrichment-coarsening strategy is well founded. *Table 1* gives details of the adaptation steps.

In *Figure 11*, the pressure coefficient C_p on the lower wall of the channel, calculated with and without grid adaptation, is compared with the results from Reference 11 which were obtained using a 72×21 structured mesh with 42 nodes on the bump. Both of these computations agree well with the workshop results, the one using the adaptive grid strategy being closer, particularly in the vicinity of the shock.

From the point of view of the efficiency, the method using adaptive grids produces a speed-up in the calculation time. In fact, the B-grid (*Figure 5*) required about 18.4 sec/iteration, while in

Table 1 Details of the sequence of meshes

Mesh	Elements	Nodes	Nodes on bump	CPU/iteration
1	445	246	6	2.8 sec
2	655	349	17	4.2 sec
3	966	509	21	9.1 sec

applying the adaptive grid strategy, 16.2 sec/iteration were needed. Although the gain in CPU time per iteration is not dramatic, fewer iterations are needed in the adaptive strategy since the solutions are transferred between grids. Also, the quality of the solution is considerably improved, and much less memory is required than with the fixed grid.

RAE 2822 aerofoil

A final test was undertaken to compute the flow over the RAE 2822 configuration using the second-order scheme combined with the grid adaptation mechanism. This calculation was performed with a free-stream Mach number of 0.75 and with an angle of attack $\alpha = 2.8°$.

Figure 12 depicts the initial mesh and the final mesh obtained after 6 cycles of adaptation, which is composed of 3967 elements and 2026 nodes with 65 over the aerofoil. The grid density of the final grid is high in regions of strong gradients, which the leading edge and the shock neighbourhood are. The solutions computed using both of these meshes are illustrated in *Figure 13* by means of Mach number contour plots. The benefit of using grid adaptation is obvious.

The pressure coefficient distribution is compared with the experimental data in *Figure 14.* The simulation results agree well with the experimental data on the pressure side, but there are some discrepancies on the suction side. In particular, the peak at the leading edge is not fully predicted and the shock intensity and position differ from those in the data. To be more specific, from *Figure 14* it can be appreciated that this solution predicts a shock position around 70% of the cord instead of the 55% as given by experimental data. This phenomenon has been blamed on

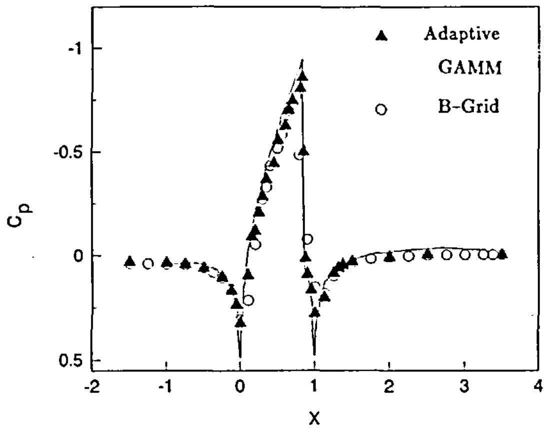

Figure 11 Comparison of the C_p on the lower surface of the bump

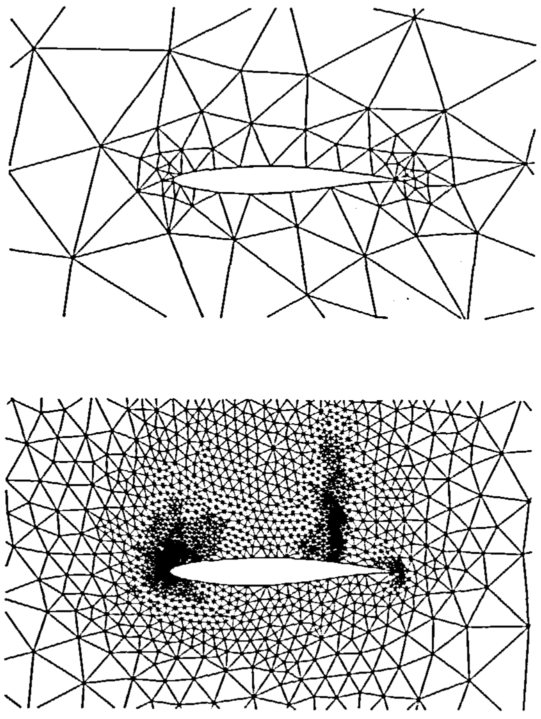

Figure 12 RAE 2822 aerofoil: initial and final grids

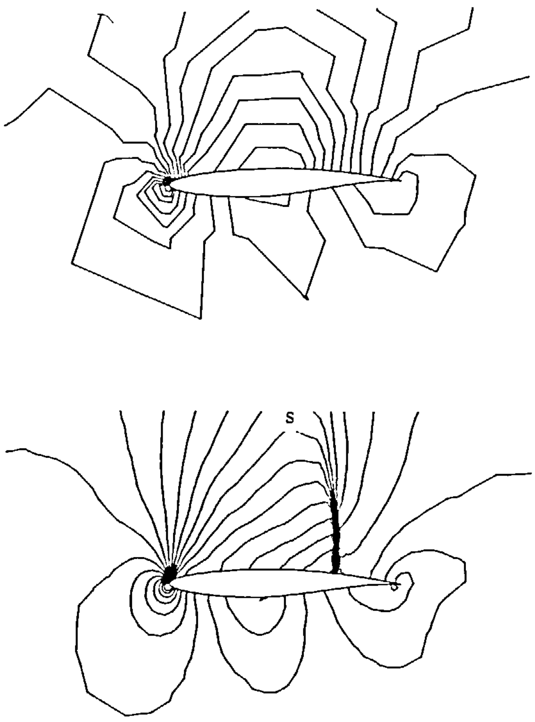

Figure 13 Mach number contours for the meshes of *Figure 11* (S: Sonic line, $\Delta M = 0.05$)

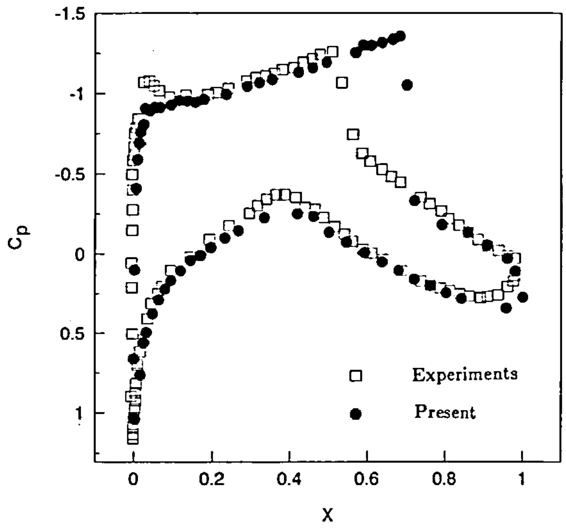

Figure 14 Pressure distribution comparison for RAE 2822 aerofoil

the model, which cannot take into account the viscous effects and the shock-boundary layer interaction taking place. In the GAMM workshop for transonic flows[11], this supercritical flow was simulated by several authors using Euler solvers. They all obtained a similar pattern, predicting a shock location in the range of 75–85% of the cord.

CONCLUSION

A fully implicit method for the steady-state solution of the Euler equations has been implemented using Roe's approximate Riemann solver in a finite-volume framework. A particular linearization, tailored for Roe's scheme, has been developed and a new implicit treatment of the boundary conditions has been designed. This has led to robust first- and second-order schemes allowing the use of high CFL numbers to obtain a converged solution in a minimum of time. The system of algebraic equations was solved directly at each time level by the LU factorization method.

A comparison of the explicit method with two implicit approaches was carried out for the two versions of the flow solver. The implicit procedures differentiate between them, and depend on Jacobian updating or Jacobian reuse. In all cases, the implicit method with Jacobian reuse was found to be the most efficient. Common time-lagged boundary conditions and implicit ones implemented via a matrix formulation were analysed. The results indicate that an implicit handling of the boundary conditions consistent with the implicitness of the scheme produces better results. A study of the rate of convergence in terms of the CFL number revealed that optimal values of the time step depend on the order of the scheme. However, the choice of very high CFL numbers always leads to a converged solution much more efficiently than using an explicit scheme.

Unstructured triangular grids have been used, allowing efficient grid adaptation as the solution develops. The mesh is refined to resolve fine structures, and coarsened in regions where the solution is smooth. The solution-adaptive grid strategy was first studied in the parallel-walled bump test case and then on the RAE 2822 aerofoil. In both cases, the result is a non-uniform grid density providing a good level of accuracy at a lower cost than a fixed grid.

ACKNOWLEDGEMENTS

The authors would like to thank the NSERC of Canada, the CRIM (Centre de Recherche Informatique de Montréal) and the MES (Ministère d'Ensignement Supérieur d'Algérie) for their financial support of this work.

REFERENCES

1 Jameson, A., Baker, T. B. and Weatherill, N. P. Calculation of inviscid transonic flow over a complete aircraft, *AIAA Paper 86-0102* (1986)
2 Barth, T. J. and Jespersen, D. C. The design and application of upwind schemes on unstructured meshes, *AIAA Paper 89-0336* (1989)
3 Peraire, J., Peiro, J., Formaggia, L., Morgan, K. and Zienkiewicz, O. C. Finite element Euler equation computations in three dimensions, *AIAA Paper 88-0032* (1988)
4 Nakahashi, K. FDM-FEM zonal approach for computations of compressible viscous flow, *Lect. Notes Phys.* (264), pp. 494–498 (1986)
5 Gnoffo, P. A. Application of program LAURA to three-dimensional AOTV flow-fields, *AIAA Paper 86-0565* (1986)
6 Thareja, R. R., Prabhu, R. K., Morgan, K., Peraire, J., Peiro, J. and Soltani, S. Application of an adaptive unstructured solution algorithm to the analysis of high-speed flows, *AIAA Paper 90-0395* (1990)
7 Hassan, O., Morgan, K. and Peraire, J. An implicit/explicit scheme for compressible viscous high-speed flows, *Comp. Math. Appl. Mech. Eng.* (1989)
8 Hassan, O., Morgan, K. and Peraire, J. An implicit finite element method for high-speed flows, *AIAA Paper 89-0363* (1989)
9 Whitaker, D. L., Slack, D.C. and Walters, R. W. Solution algorithms for the two-dimensional Euler equations on unstructured meshes, *AIAA Paper 90-0697* (1990)
10 Roe, P. L. and Pike, J. Efficient construction and utilization of approximate Riemann solutions, *Comp. Meth. Appl. Sci. Eng.* VI, 499–518 (1984)
11 Rizzi, A. and Viviand, H. Numerical methods for the computation of inviscid transonic flows with shock waves, *Proc. Symp. Notes Num. Fluid Mech.* 3 (1981)
12 Arts, A. Cascade flow calculations using a finite volume method, in *Numerical Methods for Flow in Turbomachinery Bladings*, VKI Lectures Series 5 (1982)
13 Page, M., Garon, A. and Camarero, R. Sous-programmes pour las résolution d'un système d'équations algébriques stockées en ligne de ciel, *Rapport Technique EMP-RT-89113*, École Polytechnique de Montréal, September (1989)
14 Ait Ali Yahia, D. Développement d'une méthode implicit pour la résolution des équations d'Euler sur un maillage triangulaire adaptatif, *Master's Thesis*, École Polytechnique de Montréal (1991)
15 Lauzé, Y., Camarero, R. and Yang, H. Interactive generation of structured/unstructured surface meshes with adaptivity, in *Third Int. Conf. Num. Grid Generation in CFD and Related Fields*, (Eds A. S. Arcilla *et al.*), North-Holland, Amsterdam (1991)
16 Yang, H. *ADX Version 2.2, User's Guide*. Technical Report, École Polytechnique de Montréal (1992)