INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 23, 589–606 (1996)

# MULTIGRID ACCELERATION OF SECOND-ORDER ENO SCHEMES FROM LOW SUBSONIC TO HIGH SUPERSONIC FLOWS

B. FAVINI AND R. BROGLIA

*Dipartimento di Meccanica e Aeronautica, Via Eudossiana 18, I-00184 Roma, Italy*

AND

A. DI MASCIO

*Istituto Nazionale per Studi ed Esperienze di Architettura Navale, Via di Vallerano 139, I-00128 Roma, Italy*

## SUMMARY

In the present work the multigrid strategy is applied to second-order ENO schemes for the computation of steady compressible flows. The performances of the algorithm are analysed in many flow situations, ranging from low subsonic to high supersonic flows, for both internal and external problems. Three different Riemann solvers were considered in the study of computational efficiency and solution accuracy.

KEY WORDS: multigrid methods; ENO schemes; compressible flows

## 1. INTRODUCTION

Essentially non-oscillatory (ENO) schemes[1] can be considered among the most advanced methods in numerical simulation of inviscid compressible flows. Their success can be ascribed to several favourable aspects:

1. the conservative form of these schemes ensures that if the numerical solution converges when refining the grid, it converges to some weak solution of the Euler equations (Lax theorem);
2. the theoretical background of these algorithms is sufficiently strong even though it is based mainly on heuristic motivation when the theory departs from the simple case of a scalar problem in one space dimension;
3. they are 'robust' enough to be used without the addition of non-physical extra terms (artificial dissipation) to stabilize the calculation;
4. they are able to describe any shock in the field with extremely sharp profiles, no matter how complex the topology of the flow may be.

All these positive aspects make these algorithms extremely attractive for numerical simulation in unsteady gas dynamics. Conversely, the CPU time requirement is the main shortcoming for steady

CCC 0271–2091/96/060589–18

*Received April 1995*
*Revised January 1996*

flow computations. This disadvantage is particularly felt in design problems, for which one as to handle hundreds of thousands of unknowns and many computations have to be planned. From this point of view, methods based on central differences with artificial dissipation are to be preferred. In fact, besides the low CPU cost per iteration, the use of multigrid algorithms to improve the convergence rate of these schemes is well-established. The most popular algorithm of this kind is Jameson's scheme,[2] based on Runge–Kutta pseudotime integration, which is at present the fastest algorithm in steady compressible flow simulation. Other applications of the multiple-grid strategy to centred schemes can be found e.g. in References 3–5.

More recently the multigrid algorithm has been applied to upwind schemes. Some interesting studies on this topic were carried out e.g. by Koren[6] and Dick,[7] who applied the multigrid algorithm to flux difference splitting methods.

On the basis of the remarkable improvement in performances of centred and upwind schemes with multigrid algorithms, we checked how the multigrid strategy behaves when applied to second-order ENO schemes. To this end we studied a full approximation scheme (FAS) combined with the V-cycle and with the full multiGrid (FMG) cycle. The CPU time requirements were analysed in problems of practical interest for subsonic, transonic and supersonic flows.

Moreover, the analysis was extended to test the performances of the multigrid algorithm when changing the Riemann solver, which is the 'building block' of any Godunov-type scheme. Its importance comes from the fact that it yields the values of the state variables at cell interfaces that are used in the computation of the numerical fluxes; therefore it determines the damping properties of the algorithm and the accuracy of the solution. Three solvers were checked: the exact solver developed by Gottlieb and Groth[8] and two approximate solvers, the well-known Roe solver[9] and another one derived from the general solver developed by Harten *et al*,[10] which is the least expensive of the three.

The paper is organized as follows. In Section 2 a brief review of second-order ENO schemes is reported, together with a short description of the Riemann solvers used in the sequel. In Section 3 the main features of the multigrid algorithm are described. In Section 4 the results for the test cases are discussed in details. Finally in Section 5 the properties of the scheme are summarized.

## 2. NUMERICAL METHOD

A finite volume ENO scheme is used for the numerical integration of the Euler equations. Only a brief review of the practical aspects for the second-order scheme will be given; the reader is referred to Reference 1 for a rigorous and complete discussion on the formulation of the problem and the algorithm.

### *2.1. Integration scheme*

For the finite volume solution the computational domain is divided into disjoint subdomains $D_{i,j}$ $(i = 1, \dots, N_i, j = 1, \dots, N_j)$. Suppose that $\mathbf{V}^n_{i,j}$ is a second-order approximation of the mean value of the state variables $\mathbf{U}(x, y, t^n)$ in the subdomain $D_{i,j}$ at time $t = t^n$. Then the discrete form of the Euler equations used to update the solution at $t^{n+1} = t^n + \tau$ is

$$\mathbf{V}^{n+1}_{i,j} = \mathbf{V}^n_{i,j} - \frac{\tau}{A_{i,j}}(\hat{\mathbf{H}}^n_{i+1/2,j} l_{i+1/2,j} - \hat{\mathbf{H}}^n_{i-1/2,j} l_{i-1/2,j} + \hat{\mathbf{H}}^n_{i,j+1/2} l_{i,j+1/2} - \hat{\mathbf{H}}^n_{i,j=1/2} l_{i,j-1/2}), \tag{1}$$

where, for instance, $\hat{\mathbf{H}}^n_{i+1/2,j}$ is a second-order approximation of the exact flux normal to the cell side $\mathbf{l}_{i+1/2,j}$ (see Figure 1), $A_{i,j}$ is the measure of $D_{i,j}$ and $l_{i+1/2,j}$ is the length of $\mathbf{l}_{i+1/2,j}$.

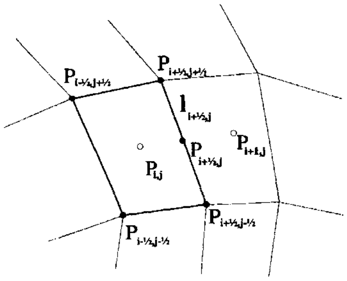


Figure 1. Sketch of discretization

The computation of $\hat{\mathbf{H}}_{i+1/2,j}$ is achieved by the second- order Gauss quadrature formula (or by any other formula at least second-order accurate)

$$\hat{\mathbf{H}}_{i+1/2,j} = \mathbf{H}[\mathbf{U}(x_{i+1/2,j}, y_{i+1/2,j}, t^n + \tau/2)] + O(h^2). \tag{2}$$

The value of $\mathbf{U}(x_{i+1/2,j}, y_{i+1/2,j}, t^n + \tau/2)$ is computed by means of a Taylor series expansion of some auxiliary variables $\mathbf{W}$ (e.g. primitive variables $\rho$, $u$, $v$, $p$) around the centroids of the two nearest cells ($i, j$ and $i + 1, j$ in the example in Figure 1). In order to avoid the onset of unphysical oscillations in the solution, the spatial gradients of $\mathbf{W}$ are computed by means of the so-called minmod function, whereas time derivatives can be related to the spatial gradients by means of the Euler equations written in terms of $\mathbf{W}$.

Because of the two Taylor series expansions, we do not have a unique value of $\mathbf{W}$ at the cell interface for the calculation of the numerical flux. The choice of the state variables is made by means of the solution of a Riemann problem, which is the main distinguishing feature of any Godunov-type algorithm and will therefore be discussed in more detail in the next subsection.

The scheme described above is second-order-accurate in both time and space; when applied to a scalar problem in one space dimension, it is total variation diminishing (TVD). Linear analysis and numerical experiments show that it is stable when

$$\tau \leqslant \min\left(\frac{1}{u + a|\nabla\xi|}, \frac{1}{v + a|\nabla\eta|}\right), \tag{3}$$

where $a$ is the speed of sound and ($\xi$, $\eta$) are curvilinear co-ordinates.

### *2.2. Riemann solvers*

The solution of the Riemann problem at each cell interface deserves further comments. First of all, it is a rather expensive part of the algorithm (at least for second-order ENO schemes) and therefore a particular choice can significantly affects the total CPU time. Moreover, stability and accuracy of the solution depend strongly on this choice and often efficiency is achieved at the expense of accuracy and stability.

In the present work we have analysed the behaviour of the multigrid algorithm with three Riemann solvers. The first one is the solver developed by Gottlieb and Groth,[8] which yields the exact solution of the Riemann problem. This algorithm, based on the Newton method, converges very rapidly to the solution in regions of smooth flow, where two iterations generally suffice, and also near discontinuities, where four or five iterations are required at most.

The second solver is the Roe solver,[9] which yields an approximate solution of the Riemann problem. It is based on a suitable linearization of the Euler equations. This solver requires no iterations and is cheaper than the exact solver. For example, the solution of a standard Riemann problem consisting of a shock, a contact discontinuity and an expansion wave requires about $6 \times 10^{-5}$ s of CPU time (three iterations) on an IBM RS/6000 Model 350 workstation, while the same problem is solved in $1{\cdot}78 \times 10^{-5}$ s by the Roe solver. Moreover, the global numerical solution obtained with this solver is practically identical, in most situations, with the solution achieved by the exact solver.

The third solver is derived from that proposed by Harten *et al.*,[10] who give the general form of the flux at the cell interface in the form

$$\mathbf{H}_{i+1/2,j} = \frac{1}{\lambda^{\mathrm{R}} - \lambda^{\mathrm{L}}}[\lambda^{\mathrm{R}}\mathbf{H}(\mathbf{U}^{\mathrm{L}}) - \lambda^{\mathrm{L}}\mathbf{H}(\mathbf{U}^{\mathrm{R}}) + \lambda^{\mathrm{R}}\lambda^{\mathrm{L}}(\mathbf{U}^{\mathrm{R}} - \mathbf{U}^{\mathrm{L}})], \tag{4}$$

where $\lambda^{\mathrm{R}}$ and $\lambda^{\mathrm{L}}$ are the approximate values of the largest and smallest wave speeds respectively. Einfeldt *et al.*[11] studied the properties of this solver and suggested computing the wave speeds as

$$\lambda^{\mathrm{L}} = \min(\tilde{u}_{\mathrm{n}} - \tilde{a}, u_{\mathrm{n}}^{\mathrm{L}} - a^{\mathrm{L}}), \qquad \lambda^{\mathrm{R}} = \max(\tilde{u}_{\mathrm{n}} + \tilde{a}, u_{\mathrm{n}}^{\mathrm{R}} + a^{\mathrm{R}}), \tag{5}$$

where $\tilde{u}_{\mathrm{n}}$ and $\tilde{a}$ are the normal velocity and the speed of sound computed with the Roe average[9] respectively. In order to make the solver as cheap as possible, i.e. to avoid the calculation of the square roots required by $\tilde{u}$ and $\tilde{a}$, we used a different estimate of $\lambda^{\mathrm{R,L}}$ given by

$$\lambda^{\mathrm{L}} = \min(u_{\mathrm{n}}^{\mathrm{R}} - a^{\mathrm{R}}, u_{\mathrm{n}}^{\mathrm{L}} - a^{\mathrm{L}}), \qquad \lambda^{\mathrm{R}} = \max(u_{\mathrm{n}}^{\mathrm{R}} + a^{\mathrm{R}}, u_{\mathrm{n}}^{\mathrm{L}} + a^{\mathrm{L}}). \tag{6}$$

In this way the solver requires only the evaluation of the largest and smallest eigenvalues of the Jacobian of $\mathbf{F}(\mathbf{U})$ computed with the given left and right states of the Riemann problem. The computing time for the calculation of the interface flux can be as low as one-tenth of the cost of the exact solver, ($0{\cdot}6 \times 10^{-5}$ s of CPU time), making this solver the least expensive of the three. However, it is the most dissipative, in the sense that the numerical solution can be spoiled by a high amount of unphysical entropy production, as will be shown in the sequel.

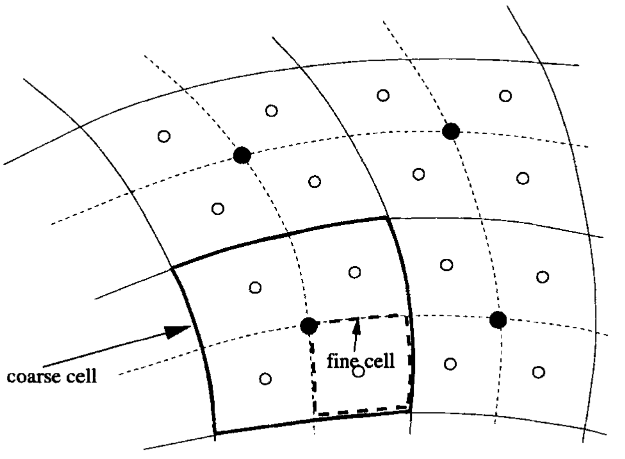
## 3. THE MULTIGRID ALGORITHM

As is well-known, standard relaxation or pseudotime integration techniques efficiently reduce those Fourier components of the error whose wavelengths are comparable with the mesh size, while they degrade in performance with the low-frequency components. The idea underlying the multigrid strategy is to use progressively coarser grids on which the low-frequency error components on the finest grid appear as high-frequency Fourier modes (because of the increased mesh size) for which the relaxation scheme works efficiently.

In the present work, two standard multigrid cycles were studied: the V-cycle and the full multigrid (FMG) cycle. In the following, only some highlights of the multigrid algorithm will be given; the reader can find more information in Reference 12.

The multigrid algorithm can be summarized as follows. Once the fine mesh is generated, the coarser grids are obtained by removing from the current grid every other line in both directions, that is, a coarse grid cell is the sum of four fine grid cells (see Figure 2).

In the V-cycle the iteration starts at the finest mesh. After a fixed number of iterations $\nu_1$, the current approximate solution on the fine mesh is transferred to the next coarser mesh to get the initial



○ : fine grid centroids ● : coarse grid centroids

Figure 2. Sketch of coarse grid generation

guess for the solution at this level, by applying a 'restriction operator'. In order to obtain the same approximation on each level, a forcing term must be added to the discrete Euler equations on the coarse grid. This term represents the truncation error on the coarse grid with respect to the fine grid. After a fixed number of iterations, $v_2$, on the second mesh, the solution is transferred to the third mesh and $v_3$ iterations are made, and so on until the coarsest mesh is reached.

Then the solution on each level is obtained by adding the correction from the next coarser one by means of a 'prolongation operator', which is an interpolation operator on the fine mesh from the four nearest coarse mesh centroids. Two prolongation operators were checked: a bilinear interpolation operator and a cheaper constant-coefficient interpolation (given by the bilinear interpolation for square regular cells). Numerical tests have proved that computational efficiency and solution accuracy are not significantly influenced by this particular choice. After the solution has been corrected, a fixed number of iterations is done on the current mesh and then the solution is transferred back to the next finer mesh until the finest level is reached, where the iterative cycle starts again.

The cycle described above is called a V-cycle with a fixed number of iterations. The so-called full multigrid (FMG) cycle has also been tested. With this cycle the algorithm starts at the coarsest level, where the solution is firstly computed; this solution is interpolated on the next finer mesh, where it is used as initial guess. Then the multigrid strategy is used on the current finest level to compute the solution, which is again interpolated on the next finer grid. The process continues until the finest mesh is reached.

To conclude this brief review, it must be remarked that the coarse mesh operator is different from the operator on the finest mesh, as is often the case in multigrid algorithms. In the present work, the numerical scheme on each coarser mesh (the finest one excluded) is the first-order Godunov scheme, which differs from the second-order ENO scheme because the solution is assumed to be piecewise constant in each cell. Therefore the CPU cost per iteration on the coarser levels is much lower not only because of the smaller number of cells, but also because of the lower cost in the computation of the data for the Riemann problem at each cell interface. This choice, as shown in Reference 12, has no influence on the accuracy of the solution on the finest mesh.

## 4. NUMERICAL TESTS

The performances of the multigrid algorithm described in the previous section have been tested for various flow regimes, in the whole range from low subsonic to high supersonic Mach numbers, for both internal and external flows. In the discussion of the numerical tests, convergence is assumed to be achieved when the $L_2$ norm of the mass convervation residual is reduced by six orders of magnitude. Moreover, the Riemann solvers described in Section 2.2 will be referred to as the 'exact' solver,[8] the 'Roe' solver[9] and the 'HLL' solver.[10]

The global efficiency of a multigrid algorithm will be measured in terms of the work reduction factor (WRF), defined as the ratio of the work needed to reach the steady state in a standard single-grid calculation to the work required by a multigrid calculation, where the work unit is defined as the cost of one iteration on the finest grid. Note that one iteration on the second level requires $1/4\alpha$ work unit, where the factor $\alpha$ takes into account the reduction of the computational work due to the use of a first-order scheme. The total cost of one cycle with $N$ levels is then

$$\text{work} \approx v_1 + \frac{1}{\alpha}\sum_{k=2}^{N}\frac{1}{4^{k-1}}v_k, \tag{7}$$

where the work in interpolating and collecting the solution from one level to the other, being a small fraction of the global work, has been neglected. The value of $\alpha$ depends on the Riemann solver. It was measured by comparing the computing times of a first- and a second-order calculation and it was found that

$$\alpha = 1{\cdot}25 \quad \text{for the exact solver,} \qquad \alpha = 1{\cdot}60 \quad \text{for the HLL solver.}$$

Another measure of the algorithm efficiency is the time reduction factor (TRP), defined as the ratio of the CPU time required by a single-grid scheme to the CPU time in the multigrid computation.

In the test cases in which the V-cycle was used, the number of levels and the number of iterations at each level will be indicated as $V(v_1, v_2, \ldots, v_N)$, which means that $N$ levels were used and the number of iterations was $v_1$ on the first (the finest) grid, $v_2$ on the second, ..., $v_N$ on the last (the coarsest). For each test case, only the best V-cycles will be reported. It should be noticed that in no cases does the V-cycle include smoothing iterations in the ascending phase, i.e. when interpolating the correction from a coarse to a finer grid. Numerical experience showed that this is the best choice from the point of view of global efficiency.

All the computations were performed on an IBM RS/6000 Model 350 workstation. No attempt was made to optimize the Fortran code; therefore the CPU times reported in the tables must be considered just for comparing the performances.

### *4.1. Internal flows*

*4.1.1. Subsonic flow in a channel.* The first test was a subsonic flow in a channel with a circular arc bump on the lower wall. The width of the channel is equal to the length of the bump, while the thickness-to-chord ratio of the bump is 10 per cent. Uniform flow was assumed as initial guess. At the upstream section the values of entropy ($S=0$) and total speed of sound ($a_0=\sqrt{\gamma}$) are imposed, whereas the value of external pressure ($p=0{\cdot}84302$) is enforced on the outflow section in order to have an inflow Mach number equal to 0·5.

Figure 3 shows the pressure contours and Mach number distribution on the lower and upper walls for the converged solution, obtained with the exact Riemann solver on an H-grid with 192 × 64 cells. Because of the discretization error and related spurious entropy production, an asymmetry in the numerical solution with respect to the mid-section can be observed. In contrast, the correct solution should be perfectly symmetric, since the flow is subsonic in the whole field.

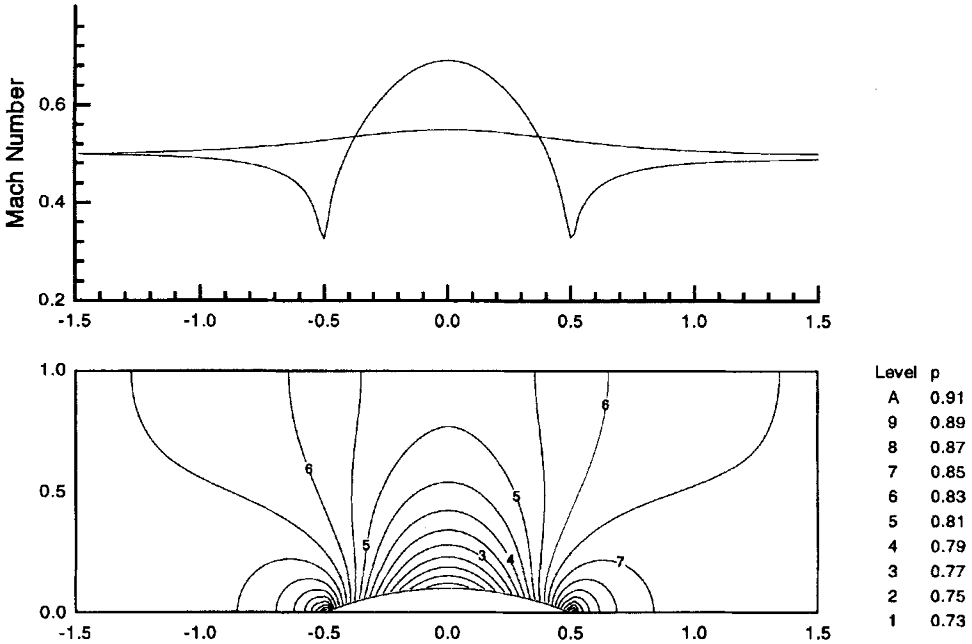


Figure 3. Subsonic flow in channel: Mach number distribution on upper and lower walls (top) and pressure contours (bottom). Exact solver, grid 192 × 64, $M_{\infty} = 0{\cdot}5$

The same flow was also computed with a 96 × 32 grid and both computations were repeated with the HLL solver. The accuracy of the solution is summarized in Table I, where the asymmetry error in terms of the vertical velocity component is reported. As already mentioned, the numerical error in the solution computed with the HLL solver is greater than that obtained with the exact solver on the same grid.

In Table II the CPU time and work required by the calculation on the two grids with both solvers are reported. As can be inferred from the table, the efficiency grows considerably with the grid size when using the HLL solver, while the increase is less significant with the exact solver. In fact, the reduction factors for the 192 × 64 mesh with three grid levels are two times larger than those obtained on the 96 × 32 mesh with the same number of levels for the case of the HLL computation, while the improvement is less than 25 per cent when using the exact solver. Moreover, the performances increase with the number of grid levels; the maximum was found with the four-level computation, in which case a very high time reduction factor is achieved (about 125 with the HLL

Table I. Accuracy of numerical solution for subsonic flow in channel. The 'asymmetry error' is the maximum percentage error on the vertical velocity component at symmetrical points

| Solver | Mesh | Asymmetry error |
|---|---|---|
| Exact | 96 × 32 | 3·38% |
| Exact | 192 × 64 | 1·41% |
| HLL | 96 × 32 | 7·16% |
| HLL | 192 × 64 | 3·42% |

Table II. Efficiency of multigrid calculation for subsonic flow in channel

| Solver | Cycle | Work | CPU time | WRF | TRF |
|---|---|---|---|---|---|
| | | Grid 192 × 64 | | | |
| Exact | SG | 76459·00 | 101962·98 | — | — |
| Exact | $V$(5/25/50) | 2769·50 | 3363·08 | 27·61 | 30·32 |
| Exact | $V$(5/15/25/50) | 2124·13 | 2579·81 | 36·00 | 39·53 |
| HLL | SG | 77335·00 | 80039·33 | — | — |
| HLL | $V$(5/25/50) | 750·30 | 728·22 | 103·07 | 109·91 |
| HLL | $V$(5/15/25/100) | 633·19 | 637·99 | 122·14 | 125·45 |
| | | Grid 96 × 32 | | | |
| Exact | SG | 25672·00 | 7491·42 | — | — |
| Exact | $V$(5/15/25) | 1098·70 | 308·90 | 23·36 | 24·25 |
| HLL | SG | 24733·00 | 4802·67 | — | — |
| HLL | $V$(5/25/50) | 527·48 | 99·34 | 46·89 | 48·34 |

solver and about 40 using the exact solver). The histories of mass residuals for both the single- and multiple-grid solutions are shown in Figure 4 for the exact solver calculation. For the case of the 192 × 64 grid we have also checked the multigrid algorithm with five and six grid levels, but it seems that the performances cannot be improved.

The better behaviour of the multigrid strategy in conjunction with the HLL solver can probably be related to its dissipative properties, which are greater than those of the exact solver, as the lower accuracy in the solution reveals.

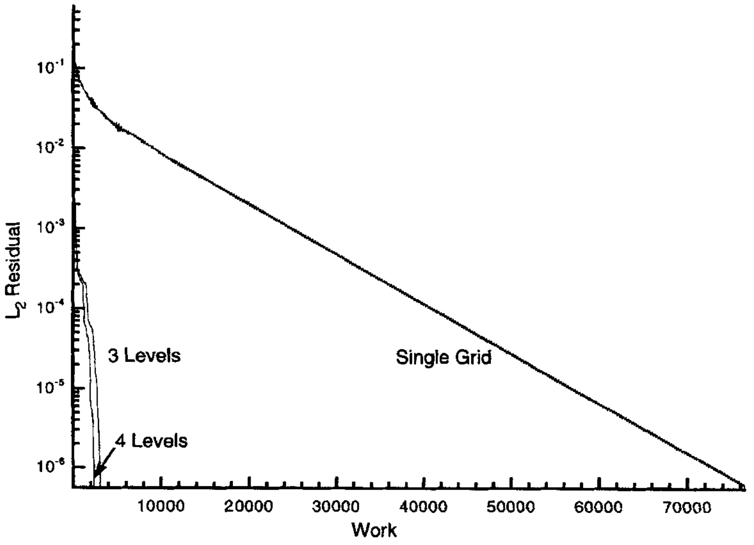


Figure 4. Subsonic flow in channel: $L_2$ mass residual for single- and multiple-grid calculations. Exact solver, grid 192 × 64, $M_\infty = 0 \cdot 5$

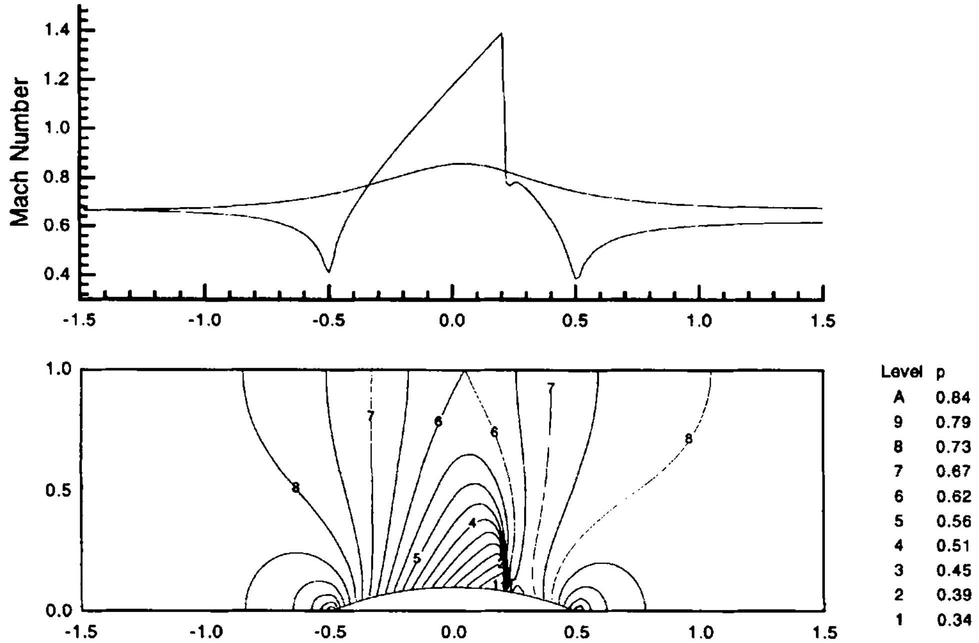


Figure 5. Transonic flow in channel: Mach number distribution on upper and lower walls (top) and pressure contours (bottom). Exact solver, grid 192 × 64, $M_\infty = 0{\cdot}675$

*4.1.2. Transonic flow in a channel.* The same channel geometry was considered for the analysis of a transonic flow computation with inflow Mach number equal to 0·675. Uniform flow was assumed as initial condition. As in the subsonic case, two grids and two Riemann solvers were used to check the performances of the multigrid algorithm.

In Figure 5 the pressure contours and Mach number distribution on the upper and lower walls of the channel are presented. The numerical solution was computed with the same 192 × 64 mesh as in the previous case and with the exact Riemann solver.

To estimate the accuracy of the solution, in Table III the entropy flux through the mid-section (which should be zero, because the upstream flow is shock-free) is reported for the test cases computed. As for the subsonic case, the exact Riemann solver yields a more accurate solution than the HLL solver. As expected, the numerical solution improves upon refining the grid.

Multigrid efficiency was tested with the same grids as in the previous case; Table IV shows the performances that have been obtained. It can be noticed that the reduction factors in the transonic

Table III. Accuracy of numerical solution for transonic flow in channel in terms of entropy flux at mid-section

| Solver | Mesh | Entropy flux |
|---|---|---|
| Exact | 96 × 32 | $0{\cdot}399 \times 10^{-5}$ |
| Exact | 192 × 64 | $0{\cdot}953 \times 10^{-6}$ |
| HLL | 96 × 32 | $0{\cdot}123 \times 10^{-4}$ |
| HLL | 192 × 64 | $0{\cdot}409 \times 10^{-5}$ |

Table IV. Efficiency of multigrid calculation for transonic flow in channel

| Solver | Cycle | Work | CPU time | WRF | TRF |
|---|---|---|---|---|---|
| Grid 192 × 64 | | | | | |
| Exact | SG | 34017·00 | 45413·17 | — | — |
| Exact | *V*(57/15/25) | 2484·00 | 3061·40 | 13·69 | 14·83 |
| Exact | *V*(5/15/25/50) | 2116·61 | 2548·45 | 16·07 | 17·82 |
| HLL | SG | 3388·00 | 34021·95 | — | — |
| HLL | *V*(5/15/25) | 1422·41 | 1396·60 | 23·76 | 24·36 |
| HLL | *V*(5/15/25/50) | 1272·64 | 1265·41 | 26·56 | 26·89 |
| Grid 96 × 32 | | | | | |
| Exact | SG | 16310·00 | 4674·25 | — | — |
| Exact | *V*(5/25/50) | 1958·40 | 539·85 | 8·33 | 8·66 |
| HLL | SG | 15394·00 | 2989·98 | — | — |
| HLL | *V*(57/10/15) | 1182·98 | 233·35 | 13·01 | 12·81 |

case are smaller than in the subsonic case. Anyhow, at least 90 per cent of CPU time is saved with both solvers in all cases. Besides, the improved performances of the multigrid algorithm upon refining the grid and when using the HLL solver are confirmed and also in this case the improvement with the exact solver is much smaller.

Finally, from Tables IV and II one can note a reduction in the number of iterations on the coarse levels in the best V-cycle. We anticipate that this is a general trend observed when the Mach number increases.

*4.1.3. Supersonic flow in a channel.* Figure 6 presents the supersonic flow in a channel with $M_\infty = 1{\cdot}4$ at inflow. In this case the thickness-to-chord ratio is 4 per cent. The calculation reported in the figure refers, as before, to a 192 × 64 grid and to the exact Riemann solver. The outflow being supersonic, boundary conditions are enforced only at the inflow section ($M_\infty = 1{\cdot}4$, $\rho_\infty = 1{\cdot}4$, $p_\infty = 1$).

The structure of the solution is more complex than in the transonic case. At the leading edge a shock appears, which is slightly bent by the interaction with the expansion wave produced by the convex wall of the bump. It is reflected by the upper wall of the channel with a Mach reflection, while the interaction with the expansion continues. Eventually it is reflected again by the lower wall and interacts with the trailing edge shock before leaving the computational field.

As to multigrid efficiency, computing times and work units for both single- and multiple-grid techniques are reported in Table V. In the current case the time reduction factor is much smaller than in the previous tests. However, the multigrid technique is still worth applying. In fact, the multiple-grid calculation allows us to save more than 70 per cent of CPU time in all cases. It must be noticed that, differently from the previous cases, the time reduction factor when using the HLL solver is almost the same as the computation with the exact solver on the 96 × 32 grid, while the HLL calculation is again more efficient on the finer grid (192 × 64).

As noted at the end of the previous subsection, the number of iterations on the coarse meshes is smaller than in the subsonic and transonic cases.

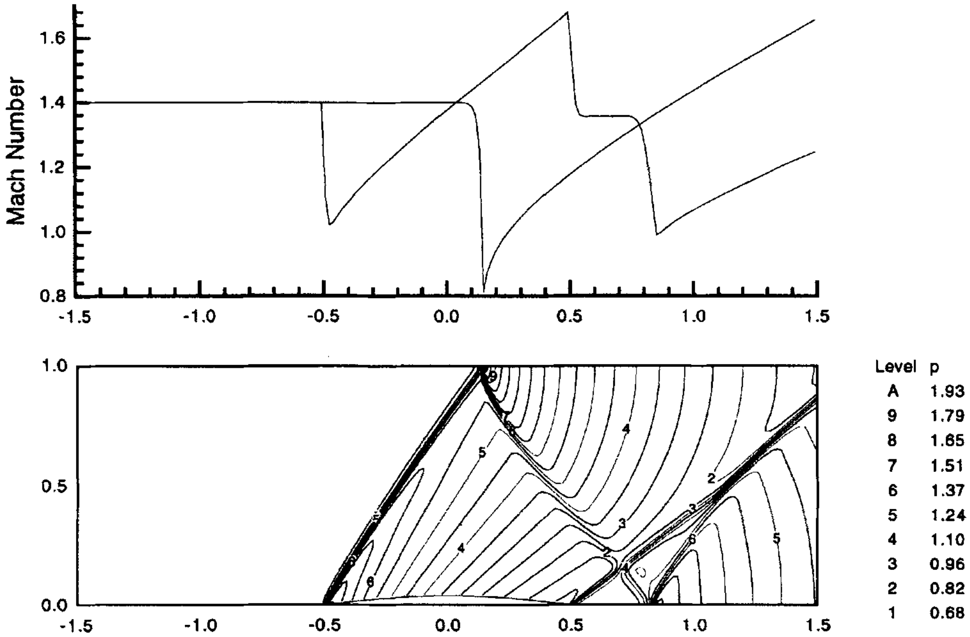


Figure 6. Supersonic flow in channel: Mach number distribution on upper and lower walls (top) and pressure contours (bottom). Exact solver, grid 192 × 64, $M_\infty = 1{\cdot}4$

## *4.2. External flows*

*4.2.1. Transonic flow around an NACA 0012 aerofoil.* The transonic flow past an NACA 0012 aerofoil at $M_\infty = 0{\cdot}85$ and 1° angle of attack is a classical test case.[13] It is a severe test case for inviscid flow solvers, because the numerical solution is extremely sensitive to calculation parameters such as grid size, stretching, far-field condition and outer boundary location. In the test case reported

Table V. Efficiency of multigrid calculation for supersonic flow in channel

| Solver | Cycle | Work | CPU time | WRF | TRF |
|---|---|---|---|---|---|
| | | Grid 192 × 64 | | | |
| Exact | SG | 6856·00 | 8937·20 | — | — |
| Exact | *V*(5/10/15) | 2292·60 | 2896·42 | 2·99 | 3·11 |
| Exact | *V*(5/10/15/20) | 2288·25 | 2814·55 | 3·00 | 3·18 |
| HLL | SG | 6867·00 | 6952·26 | — | — |
| HLL | *V*(5/10/15) | 1399·30 | 1408·45 | 4·91 | 4·94 |
| HLL | *V*(5/10/15/20) | 1273·49 | 1364·38 | 5·39 | 5·10 |
| | | Grid 96 × 32 | | | |
| Exact | SG | 3444·00 | 982·72 | — | — |
| Exact | *V*(5/10/15) | 1216·85 | 310·94 | 2·83 | 3·16 |
| HLL | SG | 3378·00 | 629·33 | — | — |
| HLL | *V*(5/10/15) | 1085·16 | 202·72 | 3·11 | 3·10 |

in this subsection, the physical domain was discretized by means of an O-type mesh whose external boundary is placed 100 chords distant from the aerofoil. Uniform flow is assumed as initial guess.

Figure 7 shows the Mach number contours obtained with the exact solver. As can be observed, the main feature of the flow field is the presence of two supersonic regions, both bounded by a sonic line and a shock wave normal to the aerofoil. In spite of the small angle of attack (only 1°), the upper region is much larger than the lower, while the upper shock is significantly stronger and closer to the trailing edge than the lower one.

In Figure 8 the pressure coefficient ($C_p$) and entropy distributions on the aerofoil surface are shown; one can notice the correct position of both upper and lower shocks when compared with the results of the calculation reported in Reference[13]. Moreover, the plots show the narrow structure of the captured shock, with only one cell in the shock zone. From the entropy distribution one can also observe the unphysical entropy production caused by the strong expansion at the leading edge. Table VI reports the values of the aerodynamic coefficients computed with three different Riemann solvers, which give practically the same results.

The computing times for this test case and their dependence on the Riemann solver and on the type of iteration scheme are shown in Table VII. It is interesting to note that in the single-grid calculation the HLL solver, which is the cheapest of the two for the solution of the single Riemann problem, yields the most expensive global solution. This is due to the reduced stability limit observed when using the HLL solver (the theoretical bound in (3) is reduced by a factor 0·4, while the same factor is 0·65 with the exact and the Roe solver) and to the increased number of iterations required to achieve steady state (compare the work for the two single-grid computations in Table VII).

As in the previous cases, the reduction factors obtained with the HLL solver are greater than those obtained with the exact solver; however, owing to the larger CPU time required in the standard single-grid procedure by the HLL solver, the differences in the final cost with the two solvers are practically negligible.

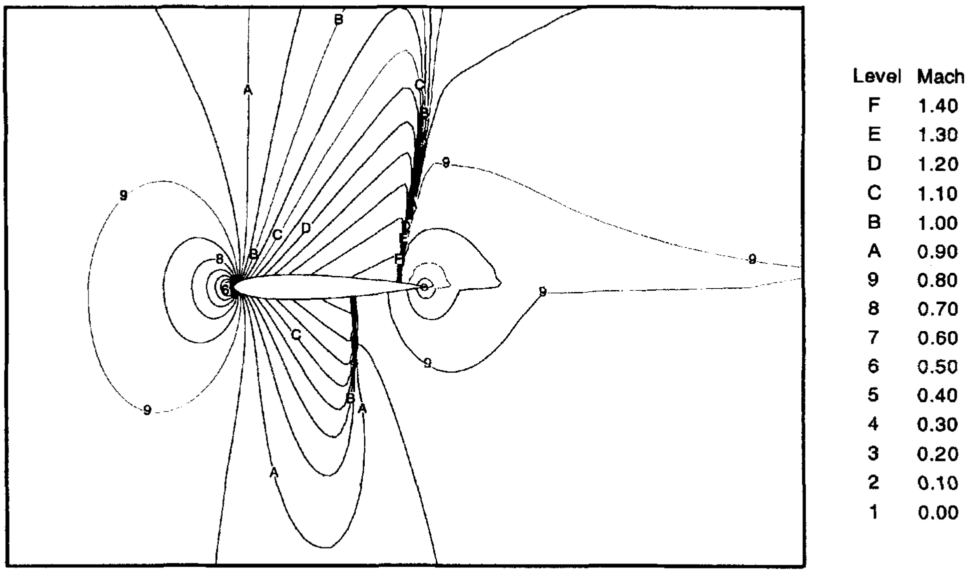


Figure 7. Transonic flow around NACA 0012 aerofoil: Mach number contours. Exact solver, $M_\infty = 0{\cdot}85$, $\alpha = 1°$, O-grid 192 × 64

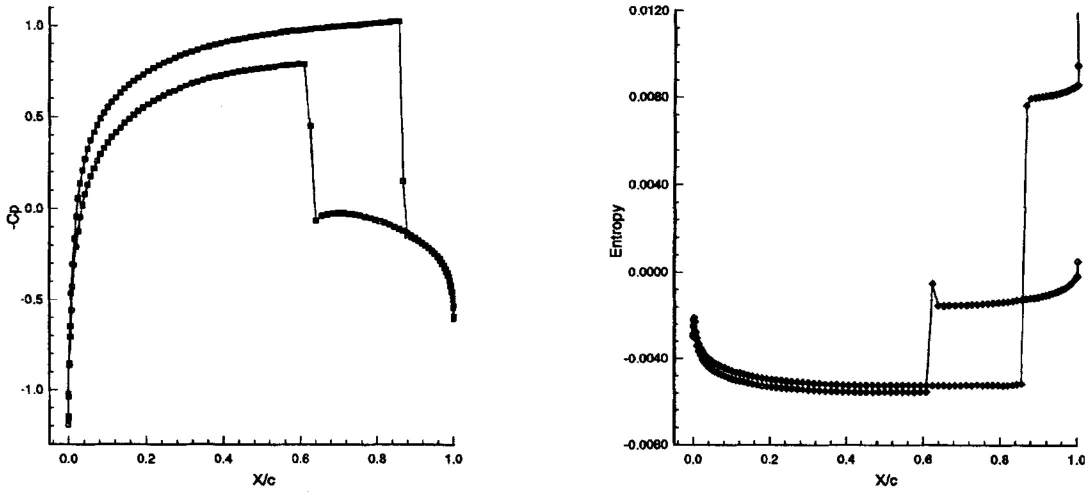


Figure 8. Transonic flow around NACA 0012 aerofoil: $C_p$ and entropy distributions on aerofoil surface. Exact solver, $M_\infty = 0{\cdot}85$, $\alpha = 1°$, O-grid 192 × 64

*4.2.2. Supersonic flow around a cylinder.* Two test cases with upstream Mach numbers $M_\infty = 4$ and 10 are presented. The physical domain is discretized with a C-type grid with 192 × 128 cells. The outer boundary is a parabola (see Figure 9) $2\delta$ distant from the cylinder, where $\delta$ is the distance between the cylinder and the shock wave, evaluated by the approximate relation[14]

$$\delta = 0{\cdot}386\, r\, \mathrm{e}^{4{\cdot}67/M_\infty^2}, \tag{8}$$

$r$ being the radius of the cylinder.

Undisturbed flow is assumed at the parabolic outer boundary, while the variables are extrapolated on the two vertical lines (supersonic outflow conditions); zero normal flow is enforced on the solid wall of the cylinder. As usual, the initial condition is uniform flow.

Table VI. Transonic flow around NACA 0012 aerofoil: aerodynamic coefficients

| Solver | $C_L$ | $C_D$ | $C_M$ |
|---|---|---|---|
| Exact | 0·3620 | 0·0592 | −0·1250 |
| HLL | 0·3624 | 0·0595 | −0·1264 |
| Roe | 0·3617 | 0·0592 | −0·1250 |

Table VII. Transonic flow around NACA 0012 aerofoil: multigrid perfomances

| Solver | Cycle | Work | CPU time | WRF | TRF |
|---|---|---|---|---|---|
| Exact | SG | 11378·00 | 13405·76 | — | — |
| Exact | $V$(5/15/50) | 1695·50 | 2034·19 | 6·71 | 6·59 |
| Exact | $V$(5/15/25/50) | 1397·38 | 1646·41 | 8·14 | 8·00 |
| HLL | SG | 22239·00 | 22507·23 | — | — |
| HLL | $V$(5/15/100) | 1341·75 | 1293·64 | 16·57 | 17·40 |
| HLL | $V$(5/15/25/100) | 1331·95 | 1284·19 | 16·70 | 17·53 |

Only the HLL solver calculation is reported, because we met with difficulties when using both the exact and the Roe solver. In fact, while we always obtained a converged solution with the HLL solver, the computations with the other two solvers often failed to reach the steady state, without any systematic dependence of convergence on the grid size and shape or on the freestream Mach number. However, such troubles are not surprising with this type of flow. Quirk,[15] in his 'catalogue of failings', reports the so-called 'carbuncle phenomenon' as an example of the Roe scheme failing and suggests a way to avoid this drawback. Although we are aware of the possible occurrence of these flaws in Godunov-type schemes, we did not follow the advice of Quirk, simply because it is beyond the scope of the present work.

Figure 9 shows Mach number and entropy contours in the case $M_\infty = 4$. The flow field is characterized by a detached bow shock in front of the cylinder and a subsonic region around the stagnation point. Moreover, the curvature of the bow shock gives rise to vorticity and entropy variation normal to the streamlines in the whole perturbed region.

In Figure 10 the numerical solution in the case $M_\infty = 10$ is presented. The flow has the same structure as in the case $M_\infty = 4$, except that the stand-off distance is smaller and the bow shock stronger.

The accuracy of the numerical solutions can be evaluated by comparing the pressure at the stagnation point with the analytical value. For the case $M_\infty = 4$ the computed pressure is 21·07, which coincides with the exact value up to the second decimal digit. The error is smaller than 0·1 per cent also for $M_\infty = 10$ ($p = 129{\cdot}31$ in the numerical computation, to be compared with the analytical value $p = 129{\cdot}22$).

In Table VIII the CPU cost and work units for the current test are reported. As in the test case of supersonic flow in a channel, the higher the upstream Mach number, the lower is the multigrid efficiency. We can observe that the loss of efficency is dramatic: in fact, the work at $M_\infty = 10$ is three times larger than in the case $M_\infty = 4$. It must also be remarked that in the best cycle at $M_\infty = 10$ the

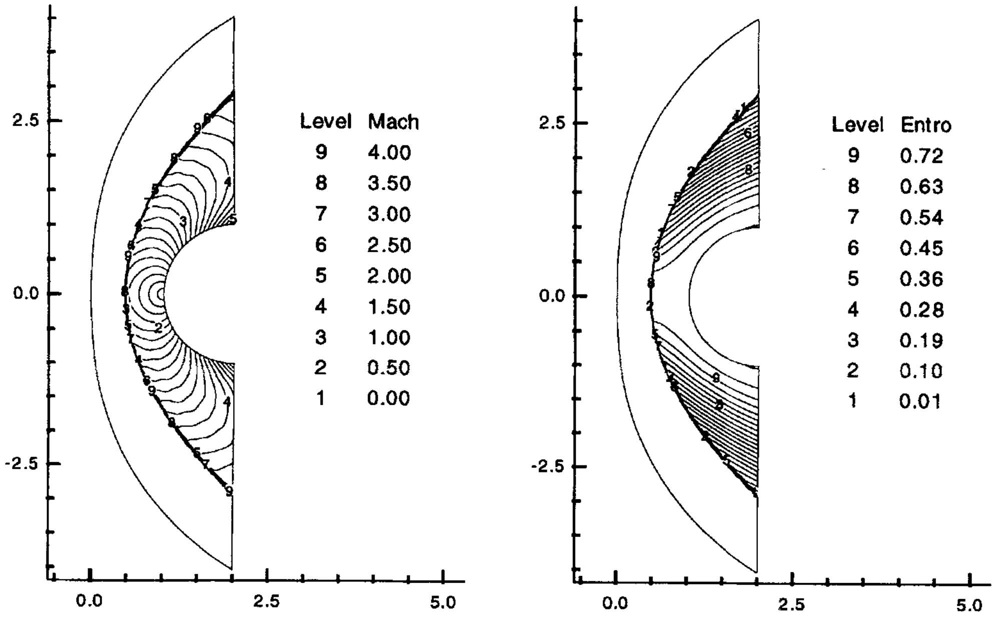


Figure 9. Supersonic flow around cylinder: Mach number (right) and entropy (left) contours. HLL solver, $M_\infty = 4$, C-grid 192 × 128

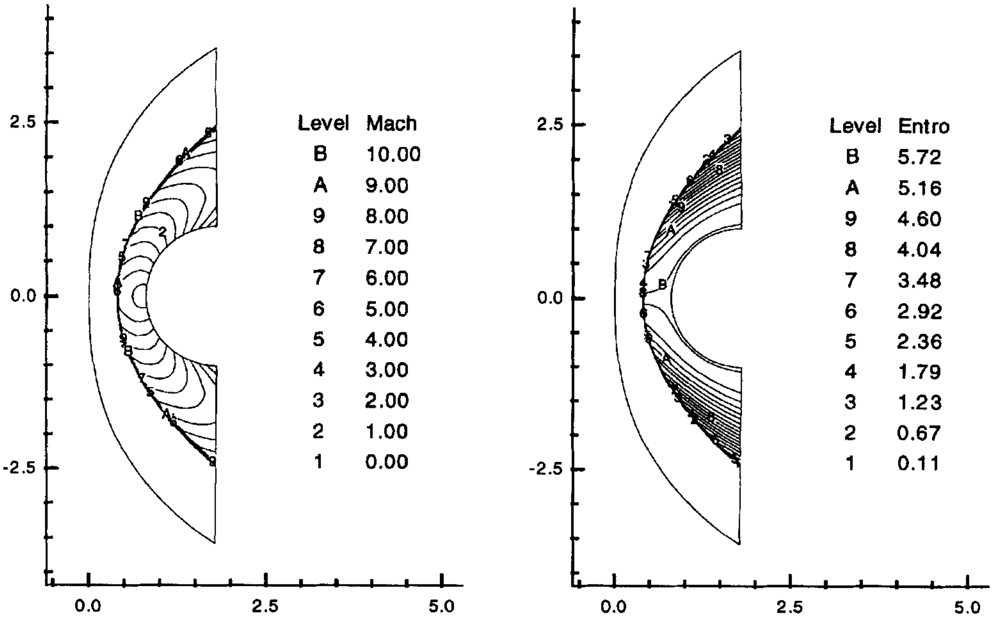


Figure 10. Supersonic flow around cylinder: Mach number (right) and entropy (left) contours. HLL solver, $M_\infty = 10$, C-grid 192 × 128

number of iterations decreases from the finest to the coarsest level (Table VIII), i.e. the structure of the optimum V-cycle is reversed with respect to lower-Mach-number computations.

*4.2.3. Supersonic flow around an ellipse.* The supersonic flow past an ellipse with aspect ratio equal to 4 is the last test case we report in this study. The upstream Mach number is 8·15 and the angle of attack $\alpha$ is set to 30°.

The domain boundaries can be seen in Figure 11, where the numerical solution is shown in terms of Mach number and entropy contours. A C-type grid with 192 × 128 cells was used in the computation. Initial and boundary conditions are the same as in the previous test case. Also in this case the computation with the exact and the Roe solver failed to converge and therefore only the HLL solver could be used in the study of the multigrid performances.

The structure of the flow field looks similar to the flow past a cylinder reported in the previous subsection. The only noticeable difference lies in the asymmetry of the flow with respect to the horizontal axis, which is responsible for the strong expansion on the upper wall of the ellipse downstream of the bow shock.

Table VIII. Supersonic flow around cylinder: multigrid efficiencies for (a) $M_\infty = 4$, (b) $M_\infty = 10$. HLL solver

| Cycle (a) | Work | WRF | Cycle (b) | Work | WRF |
|---|---|---|---|---|---|
| SG | 12644·00 | — | SG | 11524·00 | — |
| *V*(5/15/25) | 1298·15 | 9·74 | *V*(15/10/5) | 3867·11 | 2·98 |
| *V*(5/10/15/25) | 999·52 | 12·65 | *V*(30/15/10/5) | 3255·37 | 3·54 |
| *V*(5/10/15/25/30) | 967·40 | 13·07 | *V*(30/20/15/10/5) | 2527·19 | 4·56 |

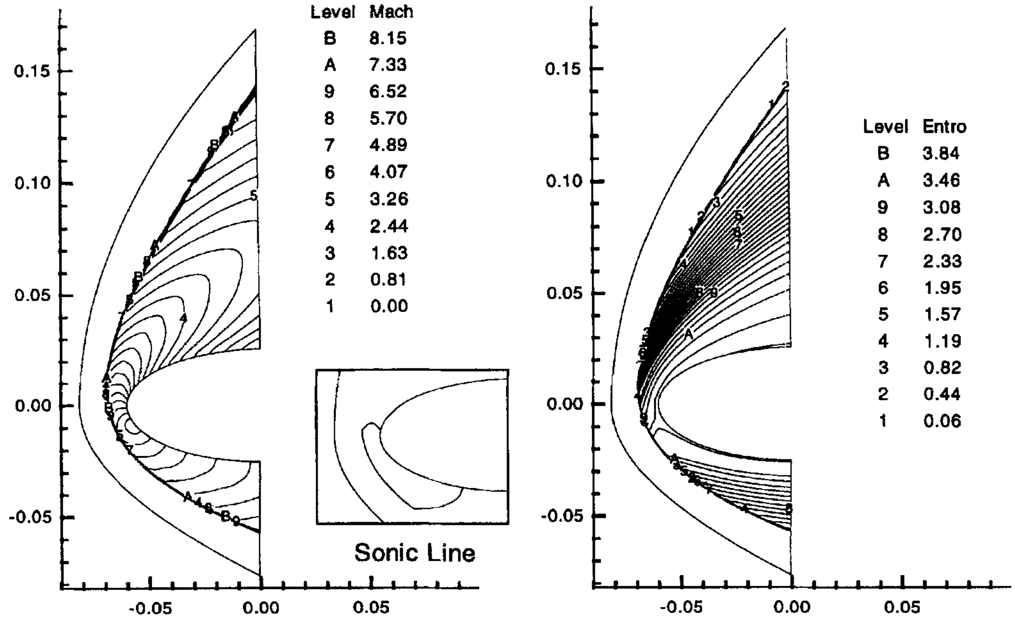


Figure 11. Supersonic flow around ellipse: Mach number (right) and entropy (left) contours. HLL solver, $M_\infty = 8{\cdot}15$, C-grid $192 \times 128$

Table IX. Supersonic flow at $M_\infty = 8{\cdot}15$ around ellipse: multigrid performances

| Cycle | Work | Time | WRF | TRF |
|---|---|---|---|---|
| SG | 18112·00 | 28837·11 | — | — |
| *V*(15/10/5) | 5895·99 | 9282·44 | 3·07 | 3·11 |
| *V*(35/15/10/5) | 4870·25 | 7694·33 | 3·71 | 3·75 |
| *V*(35/20/15/10/5) | 3246·13 | 5265·99 | 5·58 | 5·48 |

As for the supersonic flow past a cylinder, the solution is rather good, the error in terms of pressure at the stagnation point being only 1·5 per cent with respect to the analytical solution ($p = 85{\cdot}08$ and 85·98 are the numerical and the exact value respectively).

In Table IX the results obtained for the single- and the multiple-grid calculations are summarized. It should be noted that, as in the case of the flow past a cylinder at $M_\infty = 10$, the number of iterations for the 'optimal' cycle decreases on the coarser levels. Moreover, the work reduction factor increases with the number of grid levels and reaches a maximum value of about 5·5 with five levels (i.e. the finest grid is 192 × 128, on which 35 iteration are done, while the coarsest is 12 × 8, with only five iterations).

## 5. CONCLUSIONS

In the present work we have analysed the behaviour of the multigrid strategy when applied to second-order ENO-type schemes. It was found that the efficiency of the multigrid algorithm was extremely high when dealing with fully subsonic problems, for which the steady state Euler equations are elliptic everywhere in the field. In the most favourable situation the CPU time required by the multiple-grid algorithm was as low as 1/125 of the CPU time of the single-grid calculation. Although this must be considered as a peak performance, a reduction factor of 35–50 was not unusual in the other subsonic tests.

The algorithm works equally well when dealing with transonic flows for both internal and external problems. In these cases we deal with mixed-type problems, in the sense that the problem changes from elliptic in subsonic regions to hyperbolic in supersonic regions, which are bounded by shock waves and sonic lines along which the problem is parabolic. Numerical experiments reveal that a computation 10 times as cheap as the single-grid computation is to be expected in most cases.

The efficiency of the multigrid algorithm is still good when calculating external supersonic flows, for which a 10-times reduction in CPU time is often obtained if the Mach number is smaller than 3–4. Conversely, it was found that the performances are much worse when calculating external supersonic flows past blunt bodies with Mach number greater than 8–10 or internal supersonic flows past slender obstacles. In these cases the reduction factor to be expected is at most 5. The lowering of multigrid effectiveness can be related to the contraction of the subsonic region in external supersonic flows or to its almost total absence for the case of internal flows past slender obstacles.

With regard to the multigrid effectiveness as a function of the flow speed, we can conclude that we have the best performances in calculating low subsonic flows, while, as expected, the gain in CPU time becomes smaller and smaller as the Mach number grows. Moreover, the structure of the 'optimal' V- cycle regularly changes from the usual cycle with few iterations on the fine grid and many more smoothing steps on the coarse levels to a structure which is completely reversed. It can be inferred from the test cases reported in the previous section that the ratio of the iteration number on the finest grid to the iteration number on the coarsest is a monotonic fuction of the Mach number, ranging from values much smaller than unity in subsonic flows to values larger than unity in high supersonic cases.

The full multigrid algorithm was applied to most of the problems reported in the paper. The results for the transonic flow around an NACA 0012 aerofoil, discussed in Section 4.2.1, are reported in Table X; they can be compared with the performances of the previous calculations in Table VII. It was found that no significant advantages were observed with respect to the simpler V-cycle calculations and therefore it seems not worth doing in view of the increased complexity of the Fortran code.

Regarding the performances of the multigrid algorithm with different Riemann solvers, the best efficiency was always obtained with the HLL solver,[10] which is also the least expensive for the solution of the single Riemann problem. In all the test cases, in fact, the largest time reduction factor was always found when using this solver. However, this favourable aspect is often counterbalanced by a reduced accuracy in the numerical solution. Moreover, this does not mean that the HLL solver is always the best choice from the point of view of CPU time reduction. In fact, in single-grid computation it can happen that the low cost per iteration does not result in a globally cheaper calculation, in that it may require many more iterations for convergence than the exact solver. This behaviour was observed, for instance, in the calculation of the transonic flow past an NACA 0012

Table X. Transonic flow around NACA 0012 aerofoil: FMG performances

| Solver | Cycle | Work | CPU time | WRF | TRF |
|---|---|---|---|---|---|
| Exact | FMG 4 levels | 1239·51 | 1485·96 | 9·18 | 9·02 |
| HLL | FMG 4 levels | 1267·86 | 1155·42 | 17·54 | 19·48 |

Table XI. Transonic flow around NACA 0012 aerofoil: multigrid performances

| Solver | Cycle | Work | CPU time | WRF | TRF |
|---|---|---|---|---|---|
| Roe | SG | 11391·00 | 12693·53 | — | — |
| Roe | *V*(5/15/50) | 1460·37 | 1704·89 | 7·80 | 7·45 |
| Roe | *V*(5/15/25/50) | 1222·59 | 1448·60 | 9·31 | 8·76 |

aerofoil. Anyway, also in this case the HLL solver turned out to yield the least expensive calculation when using the multigrid technique, because of its good smoothing properties.

As to the third solver tested in the paper, the Roe solver,[9] it proved to be only slightly less expensive than the exact one, as shown in Table XI (compare with Table VII) in the transonic flow calculation past the NACA 0012 wing section. In most cases the behaviour of this solver is essentially the same as that of the exact solver both in terms of global numerical efficiency and in terms of accuracy (see Table VI).

In conclusion, we can affirm that the application of the multigrid strategy to ENO-type integration schemes improves the efficiency of steady state calculations in all cases, even though the effectiveness of the multiple-grid strategy decreases when the Mach number increases.

## References

1. A. Harten, B. Engquist, S. Osher and S. R. Chakravarthy, 'Uniformly high order accurate essentially non-oscillatory schemes', *J. Comput. Phys.*, **71**, 231 (1987).
2. A. Jameson, 'Solution of the Euler equation for two dimensional transonic flow by a multigrid method', *MAE Rep. 1631*, Princeton University, 1983.
3. G. M. Johnson, 'Multiple-grid convergence acceleration of viscous and inviscid flow computations', *NASA TM–83361*, 1983.
4. R.-H. NI, 'A multiple-grid scheme for solving the Euler equations', *AIAA J.*, **20**, 1565–1571 (1982).
5. S. Yoon and D. Kwak, 'Multigrid convergence of an implicit symmetric relaxation scheme', *AIAA J.*, **32**, 950–955 (1994).
6. B. Koren, 'Multigrid and defect correction for the steady Navier– Stokes equations', *J. Comput. Phys.*, **87**, 25–46 (1990).
7. E. Dick, 'Multigrid formulation of polynomial flux–difference splitting for the steady Euler equations', *J. Comput. Phys.*, **91**, 161–173 (1990).
8. J. J. Gottlieb and C. P. T. Groth, 'Assessment of Riemann solvers for unsteady one-dimensional, inviscid flows of perfect gases', *J. Comput. Phys.*, **78**, 437 (1988).
9. P. L. Roe, 'Approximate Riemann solvers, parameter vectors, and difference schemes', *J. Comput. Phys.*, **43**, 357 (1981).
10. A. Harten, P. D. Lax and B. Van-Leer, 'On upstream differencing and Godunov-type schemes for hyperbolics conservation laws', *SIAM Rev.*, **25**, 35–61 (1983).
11. B. Einfeldt, C. D. Munz, P. L. Roe and B. Sjögreen, 'On Godunov- type methods near low densities', *J. Comput. Phys.*, **92**, 273– 295 (1991).
12. A. Brandt, *Multigrid Techniques: 1984 Guide with Application to Fluid Dynamics*, Weizmann Institute of Science, Rehovot, 1984.
13. A. Dervieux, B. Van Leer, J. Periaux and A. Rizzi (eds), *Numerical Simulation of Compressible Euler Flows*, NNFM, Vol. 26, Vieweg, Braunschweig, 1989.
14. A. H. Shapiro, *The Dynamics and Thermodynamics of Compressible Fluid Flow*, Ronald, Cambridge, MA, 1954.
15. J. J. Quirk, 'A contribution to the great Riemann solver debate', *ICASE Rep. 92-64*, 1992.