JOURNAL OF COMPUTATIONAL PHYSICS 60, 235–252 (1985)

Multigrid Relaxation for the Euler Equations

WIM A. MULDER*

University Observatory, P.O. Box 9513, 2300 RA Leiden, The Netherlands

Received March 7, 1984; revised October 16, 1984

A multigrid method for finding stationary solutions of the Euler equations is described and tested. Spatial discretization is obtained by upwind differencing. Implicit time discretization is applied to construct a switched evolution/relaxation (SER) scheme. The multigrid method, a correction scheme, accelerates the inversion of the large linear system that arises in the SER scheme. In a two-dimensional transonic test problem the correction scheme is used in combination with symmetric block Gauss–Seidel relaxation. Restriction is carried out by the addition of residuals and corresponding blocks; prolongation by the simple distribution of coarse-grid corrections to the finer grid. It turns out that the total amount of work required to obtain a converged solution is roughly proportional to $N^{0.1}$, N being the number of zones, both for a first-order and a second-order accurate solution. The gain in efficiency with respect to a single-grid scheme thereby becomes proportional to about $N^{0.4}$ in both cases.

1. INTRODUCTION

Implicit time discretization, combined with upwind space differencing, yields a fast and robust method for finding stationary solutions of the Euler equations. Particularly successful is the switched evolution/relaxation (SER) scheme, which provides a smooth switching between explicit time integration and Newton's method for finding zero values of a given function. For one-dimensional problems quadratic convergence can be obtained, as shown in an earlier paper [1].

In two dimensions the exact inversion of the linear system arising in the implicit formulation is too costly. Various approximate solvers are described in [2], yielding a convergence speed that is considerably better than the speed of the popular ADI or AF methods.

In this paper an efficient approximate solver based on the multigrid method for the solution of large linear systems, the correction scheme, is introduced. An outline of the basic multigrid concepts can be found in [3]. Here they are formulated in the context of the upwind-differenced Euler equations for problems containing discontinuities (Sect. 2). In Section 3 a test problem is described: transonic flow through a channel with a circular bump at one wall. Numerical results for single grid and multigrid, with first- and second-order spatial accuracy, are given in Section 4. Section 5 contains some conclusions and suggestions.

* Present address: Department of Computer Science, Stanford University, Stanford, CA 94305.

0021-9991/85 $3.00

2. METHOD

For clarity the multigrid method will be explained for a one-dimensional scalar hyperbolic equation. Generalization to a system of equations in one or more dimensions is straightforward and will be presented further on. Let the equation be:

$$\frac{\partial w}{\partial t} = -\frac{\partial f}{\partial x} + q(w) \equiv r(w). \tag{1}$$

Here $f(w)$ is the flux of the state quantity w, $q(w)$ is a source term, and $r(w)$ is the residual, the function that must be made to vanish. The implicit scheme of our choice is the linearized "backward Euler" scheme:

$$L^n \Delta_t w^n = \left[\frac{1}{\Delta t^n} - \left(\frac{dr}{dw}\right)^n\right] \Delta_t w^n = r^n(w^n). \tag{2}$$

The superscript n denotes values at a time t^n, while $\Delta t^n = t^{n+1} - t^n$ and $\Delta_t w^n = w^{n+1} - w^n$. The spatial coordinate x is assumed to be discretized according to $x_i = (i - \frac{1}{2})\,\Delta x$. The discrete values w_i of the state quantity are obtained by volume averaging:

$$w_i = \frac{1}{\Delta x} \int_{x_i - 1/2\Delta x}^{x_i + 1/2\Delta x} dx\, w(x), \qquad i = 1,\ldots, N. \tag{3}$$

The local residual $r_i(w)$ is computed, for the present purpose, by a first-order upwind-difference scheme on a three-point stencil: $r_i(w) = r_i(w_{i-1}, w_i, w_{i+1})$. The timestep Δt^n is determined by

$$\begin{aligned} \Delta t^n &= \varepsilon / \mathrm{RES}^n, \\ \mathrm{RES}^n &= \max_i \left(\frac{|r_i^n(w)|}{|w_i^n| + h_i^n} \right), \end{aligned} \tag{4}$$

where h_i^n is some bias to prevent division by zero. In this way the implicit scheme (2) becomes a SER scheme. In the initial phase of the iteration process, Eq. (4) guarantees that the relative variation of w is at most of the order of ε, the value of which is chosen in advance. If Δt^n is small, the implicit scheme behaves very much like an explicit time-accurate scheme. This is a safeguard against approaching an unphysical solution. Once the solution is getting closer to the steady state, Δt^n becomes larger and the scheme automatically switches to Newton's method. Quadratic convergence can be obtained, but need not be the ultimate goal. Experience teaches that once the residual RES^n has dropped to a level between 10^{-1} and 10^{-3} times its original value, an explicit scheme usually loses its convergence speed. Switching to Newton's method may provide a fast way to reach a level between about 10^{-4} and 10^{-6}, which is sufficient in most cases. Newton's

method requires the solution of the linear system (2), so it pays to put some effort in a reasonably accurate inversion. At this point a multigrid strategy becomes desirable.

The basic ingredients of a multigrid scheme are: (i) relaxation, (ii) restriction, and (iii) prolongation. To simplify the notation we introduce the linear system

$$Lu = f, \tag{5}$$

where $f = r^n$.

The single-grid relaxation scheme provides an approximate solution s^m according to:

$$\begin{aligned} s^m &:= \tilde{L}^{-1} f^m, \\ u^{m+1} &:= u^m + s^m, \\ f^{m+1} &:= f^m - Ls^m. \end{aligned} \tag{6}$$

Here $\tilde{L}$ is an approximation to L. If the initial values are chosen to be $f^0 = r^n$ and $u^0 = 0$, then a proper relaxation scheme will converge to the exact solution $u^* = L^{-1} f$. In practice, one will find an u^M after M sweeps which is a reasonable approximation to u^*, and assign this u^M to $\Delta_t w$. Suitable relaxation schemes are described in [2].

The multigrid correction scheme tries to solve the linear system

$$Le^m = f^m, \tag{7}$$

where the defect $e^m = u^* - u^m$. A simple restriction operation is

$$v_I = R_I \begin{pmatrix} v_i \\ v_{i+1} \end{pmatrix}, \qquad R_I = \tfrac{1}{2}(1 \;\; 1), \; (i = 1, 3, \ldots, N-1; \; I = (i+1)/2), \tag{8}$$

implying that the coarse-grid values v_I are obtained by averaging the quantities v_i and v_{i+1}. This is consistent with (3) if $v = w$ and preserves the conservation form of the right-hand side of Eq. (2) if $v = r$. The linear system (7) can now be restricted as follows:

$$(RL_f R^{-1})(Re_f) = (Rf_f) + \varepsilon_c$$

or

$$L_c \qquad e_c = \quad f_c + \varepsilon_c. \tag{9}$$

The subscript f corresponds to the fine grid and c to the coarse grid; ε_c is an error term to be specified shortly. The matrix R^{-1} may be defined as

$$R^{-1} = 2R^T = \begin{pmatrix} 1 \\ 1 \end{pmatrix}. \tag{10}$$

If both the left-hand and right-hand side of Eq. (9) are multiplied by 2, the computation of f_c and L_c reduces to simple additions, as all the weights in $(2R)$ and R^{-1} are equal to unity. Now RR^{-1} is the identity operator; the combination $R^{-1}R$, however, is not. In fact, the operation $R^{-1}Re_f$, occurring in Eq. (9), replaces two neighboring values of e_f by their average:

$$R^{-1}R\begin{pmatrix} e_i \\ e_{i+1} \end{pmatrix} = \frac{1}{2}\begin{pmatrix} e_i + e_{i+1} \\ e_i + e_{i+1} \end{pmatrix}. \tag{11}$$

The resulting error can be described by an operator $Q = \frac{1}{2}(1 \quad -1)$ and its generalized inverse $Q^{-1} = 2Q^T$. We have

$$Q^{-1}Q\begin{pmatrix} e_i \\ e_{i+1} \end{pmatrix} = \frac{1}{2}\begin{pmatrix} e_i - e_{i+1} \\ e_{i+1} - e_i \end{pmatrix}, \tag{12}$$

so that $R^{-1}R + Q^{-1}Q$ results in the identity operator. It follows that the error term in Eq. (9) equals

$$\varepsilon_c = -RL_fQ^{-1}Qe_f. \tag{13}$$

As seen from Eq. (12), ε_c contains the information about the difference between values in neighboring zones and therefore represents a high-frequency error. As the defect e_f is yet unknown, the error term has to be dropped from the coarse-grid equation (9), which is reasonable if e_f only contains low-frequency components. This can be accomplished by using a fine-grid relaxation scheme with good high-frequency damping before restriction.

An approximate solution of the coarse-grid equation $L_c e_c = f_c$ can be found by applying the relaxation scheme (6), starting with $f_c^0 = f_c$ and $e_c^0 = 0$. After M sweeps one obtains a approximate solution e_c^M, which has to be transferred to the fine grid. For the prolongation of any variable v_I from the coarse grid to the fine grid one only has the conservation condition $v_I = \frac{1}{2}(v_i + v_{i+1})$. In principle this leaves the freedom to choose a gradient in the zone I, yielding a difference between v_i and v_{i+1}. Such a gradient could be computed on the coarse grid by a finite-differencing procedure, similarly as in the second-order accurate upwind-difference scheme described in Appendix II. However, the experiments described later showed that the computation of gradients reduced the number of multigrid cycles by a few, but that the total amount of cpu time increased. The most cost-effective way is to distribute e_c^M uniformly over the fine grid:

$$\begin{aligned} s_f^m &:= e_c^M, \\ u_f^{m+1} &:= u_f^m + s_f^m, \\ f_f^{m+1} &:= f_f^m - Ls_f^m. \end{aligned} \tag{14}$$

Some relaxation sweeps on the fine grid can be carried out subsequently to reduce the interpolation error.

The multigrid strategy used in this paper is a simple V cycle. Of course it could be made as subtle as one would wish. For instance, more effort could be spent on solving the linear system as the solution moves into the regime where Newton's method starts working. At present, however, our only aim is to demonstrate the utility of multigrid relaxation.

A full V-cycle has the following stages:

(i) Compute $r^n(w)$ and $L^n(w)$ on the finest grid and initialize $f_f^0 = r^n(w)$ and $u_f^0 = 0$.

(ii) Perform one or more relaxation sweeps as given by Eq. (6) to reduce the high-frequency error mentioned earlier.

(iii) Restrict the present f_f^m and the matrix L_f and carry out (ii) on the coarser grid. Repeat until the coarsest grid is reached.

(iv) On the coarsest grid one has the choice between an application of the relaxation scheme or an exact inversion.

(v) Prolongate the coarse-grid correction e_c^M to the finer grid according to (14). One or more relaxation sweeps must follow in order to reduce interpolation errors. Repeat until the finest grid is reached.

(iv) Assign the final u_f^M to $\Delta_t w$.

This multigrid scheme may be generalized to a system of equations by replacing the scalar functions $w(x, t)$ and $r(x, t)$ by vector functions of x and t; thus, the local dr/dw becomes a matrix and L gets a block structure. The generalization of restriction (averaging) and prolongation (uniform distribution) to more than one dimension is straightforward. For nonuniform grids it is convenient to multiply Eq. (2) by the local cell volume. This ensures the appropriate weighting during the restriction, which then reduces to a simple addition. Details for the two-dimensional case are given in the following section.

3. Test Problem

The method is tested on the two-dimensional problem of transonic flow through a straight channel. The flow runs along the x direction and is obstructed by a circular arc on the lower wall. The channel has an x coordinate running from -1.5 to 2.5 and a y coordinate running from 0.0 to 2.0. The circular arc between $x = -0.5$ and 0.5 at $y = 0$ has a maximum thickness equal to 4.2% of the chord. Thin airfoil theory is used to transfer the boundary conditions at the arc onto the flow. For simplicity a uniform grid with square zones is adopted. The free-stream Mach number is chosen to be 0.85, resulting in a transonic but unchoked flow.

In this setting the isenthalpic Euler equations in conservation form are solved for an ideal gas with $\gamma = 1.4$:

$$\frac{\partial w}{\partial t} = -\frac{\partial f}{\partial x} - \frac{\partial g}{\partial y}, \tag{15}$$

$$w = \begin{pmatrix} \rho \\ \rho u \\ \rho v \end{pmatrix}, \qquad f = \begin{pmatrix} \rho u \\ \rho(u^2 + c^2/\gamma) \\ \rho uv \end{pmatrix}, \qquad g = \begin{pmatrix} \rho v \\ \rho uv \\ \rho(v^2 + c^2/\gamma) \end{pmatrix}.$$

Here w represents the vector of state quantities and f and g the fluxes in the x and y directions, respectively. The system of equations is closed by specifying the total enthalpy:

$$H = c^2/(\gamma - 1) + \tfrac{1}{2}(u^2 + v^2) = H_\infty. \tag{16}$$

In the test problem the free-stream values are chosen to be: $\rho_\infty = 1$, $c_\infty = 1$, $u_\infty = 0.85$, $v_\infty = 0$. The bias in the timestep given by Eq. (4) can be generalized by letting

$$\mathrm{RES}^n = \max_{ijk}\left(\frac{|r^n_{ijk}|}{|w^n_{ijk}| + h^n_{ijk}}\right), \quad k = 1, 2, 3, \; i = 1,..., N_x, \; j = 1,..., N_y \tag{17}$$

with the bias $h_{ij1} = 0$ and $h_{ij2} = h_{ij3} = \rho c$. The parameter ε is taken to be 1, but this choice is not very critical.

For the unchoked case two boundary conditions at the inlet and one at the outlet should be specified. At the inlet the direction of the flow ($v = 0$) and the total pressure are given; at the outlet the static pressure is specified. For the test problem the inlet and outlet parameters are chosen in accordance with the free-stream values. Boundaries at the lower and upper wall are simulated by the introduction of an extra zone having reflected state quantities. Further details can be found in Appendix I.

The system (2) is discretized in space with the aid of flux-vector splitting to accomplish the upwind differencing. The split fluxes used are those proposed in [4], as these can be easily linearized. They are given below for completeness:

$$f = f^+ + f^-,$$

$$f^+ = \begin{pmatrix} \rho u \\ \rho\gamma_1(u^2 + c_1^2) \\ \rho uv \end{pmatrix} \qquad \text{if} \quad u \geqslant c_1,$$

$$f^+ = \frac{\rho}{4c_1}(u + c_1)^2 \begin{pmatrix} 1 \\ 2\gamma_1 c_1 \\ v \end{pmatrix} \qquad \text{if} \quad |u| < c_1, \tag{18}$$

$$f^+ = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \qquad \text{if} \quad u \leqslant -c_1.$$

Here $c_1^2 = 2\gamma_2 (H - \frac{1}{2}v^2)$, $\gamma_1 = (\gamma+1)/(2\gamma)$ and $\gamma_2 = (\gamma-1)/(\gamma+1)$. The fluxes in the y direction are split similarly.

Flux-vector splitting is used for the interior flow and at the lower and upper boundaries. At the inlet and outlet the full flux is computed from the boundary conditions and extraplolations (see Appendix I). Equation (2) multiplied by $\Delta_i x \Delta_j y$ becomes

$$\begin{aligned} &A_{ij}\Delta_t w_{ij} - B^x_{ij}\Delta_t w_{i-1,j} - C^x_{ij}\Delta_t w_{i+1,j} - B^y_{ij}\Delta_t w_{i,j-1} - C^y_{ij}\Delta_t w_{i,j+1} = r_{ij}, \\ &r_{ij} = -\Delta_j y(f^-_{i+1,j} + f^+_{ij} - f^-_{ij} - f^+_{i-1,j}) \\ &\qquad -\Delta_i x(g^-_{i,j+1} + g^+_{ij} - g^-_{ij} - g^+_{i,j-1}), \end{aligned} \tag{19}$$

$$\begin{aligned} A_{ij} &= \frac{\Delta_i x \Delta_j y}{\Delta t} + \Delta_j y\left[\left(\frac{df^+}{dw}\right)_{ij} - \left(\frac{df^-}{dw}\right)_{ij}\right] \\ &\quad + \Delta_i x\left[\left(\frac{dg^+}{dw}\right)_{ij} - \left(\frac{dg^-}{dw}\right)_{ij}\right], \\ B^x_{ij} &= \Delta_j y\left(\frac{df^+}{dw}\right)_{i-1,j}, \qquad C^x_{ij} = -\Delta_j y\left(\frac{df^-}{dw}\right)_{i+1,j}, \\ B^y_{ij} &= \Delta_i x\left(\frac{dg^+}{dw}\right)_{i,j-1}, \qquad C^y_{ij} = -\Delta_i x\left(\frac{dg^-}{dw}\right)_{i,j+1}. \end{aligned}$$

Restriction simply adds the contributions of zones (i, j), $(i+1, j)$, $(i, j+1)$, and $(i+1, j+1)$ for $i = 1, 3, 5, \ldots, N-1$ and $j = 1, 3, 5, \ldots, M-1$, and assigns the result to zone $(I, J) = ((i+1)/2, (j+1)/2)$ of the coarser grid. For r we obtain

$$r_{I,J} = r_{i,j} + r_{i+1,j} + r_{i,j+1} + r_{i+1,j+1}, \tag{20}$$

and for the linear system

$$\begin{aligned} A_{I,J} &= A_{i,j} + A_{i+1,j} + A_{i,j+1} + A_{i+1,j+1} \\ &\quad -[B^x_{i+1,j} + B^x_{i+1,j+1} + C^x_{i,j} + C^x_{i,j+1}] \\ &\quad -[B^y_{i,j+1} + B^y_{i+1,j+1} + C^y_{i,j} + C^y_{i+1,j}], \\ B^x_{I,J} &= B^x_{i,j} + B^x_{i,j+1} \\ C^x_{I,J} &= C^x_{i+1,j} + C^x_{i+1,j+1} \\ B^y_{I,J} &= B^y_{i,j} + B^y_{i+1,j} \\ C^y_{I,J} &= C^y_{i,j+1} + C^y_{i+1,j+1}. \end{aligned} \tag{21}$$

The right-hand side r_{ij}, if multiplied by the local volume $\Delta_i x \Delta_j y$, can be considered as the integral along the zone boundary of the flux perpendicular to that boundary. This property is preserved by the restriction operator: all the flux contributions from the interior of the coarse zone cancel, leaving only those at the coarse-zone boundary.

The relaxation scheme used in this paper is a linearized version of the symmetric block Gauss–Seidel scheme, with the first sweep in the positive x and y direction and the second sweep in the opposite direction. A forward Gauss–Seidel sweep implies that in computing s_{ij}^m the forward elements $s_{i+1,j}^m$ and $s_{i,j+1}^m$ are assumed to be zero, while for the backward elements $s_{i-1,j}^m$ and $s_{i,j-1}^m$ the values most recently obtained are used. This linearized block Gauss–Seidel involves the inversion of A_{ij}, and a number of matrix multiplications. The combination of two sweeps in opposite directions results in a good short-wave dampening, for both the first-order and the second-order scheme [2].

Second-order spatial accuracy is obtained by assuming the discrete quantities ρ_{ij}, u_{ij}, and v_{ij} to be piecewise linear rather than piecewise constant in a zone. The linear dependence can be represented by the x and y components of the gradient. These are computed from differences across the zone boundaries by an averaging procedure that preserves monotonicity. The gradient is used to find the quantities ρ, u, and v on the zone boundary where the split flux vectors are computed. Details can be found in Appendix II. The linearization of the flux vectors, required for the implicit time discretization, is complicated. Therefore, an incomplete linearization is adopted by using the same Jacobians as in the first-order scheme, but computed from the boundary values (see Appendix II). In this way, the matrix L has the same structure as in the first-order scheme. Obviously, Newton's method is not exactly obtained in the relaxation phase of the SER scheme, which leads to some loss of convergence speed. Nevertheless, it still pays to carry out the multigrid scheme, as will be shown in the following section.

4. Results

The performance of the multigrid scheme was compared with that of the embedded single-grid scheme. The single-grid scheme consists of two block Gauss–Seidel relaxation sweeps in opposite directions, after which $u^{(2)}$ is assigned to $A_t w$. The multigrid scheme includes two relaxation sweeps before every restriction, and two after every prolongation. On the coarsest grid (2×1) an exact inversion was programmed. It turned out that this exact inversion made the multigrid scheme very sensitive to the treatment of the boundary conditions. For instance, a version of the computer program with a different kind of extrapolation at the inlet and outlet—using the quantities ρ, u, and v directly instead of the characteristic variables (see Appendix I)—showed a convergence speed that was considerably worse with the exact inversion than without. Apparently, the exact inversion couples the boundaries so strongly together that any deviation from the proper treatment deteriorates the multigrid convergence.

The efficiencies of the multigrid and single-grid scheme can be measured by the usual quantity "work." Our restriction and prolongation operators are so simple that the relaxation routine is the most time-consuming part of the multigrid scheme. Therefore, the total amount of "work" can be found by counting how many times

the relaxation routine is applied on each grid, weighting each grid total with the relative grid size (number of zones divided by number of zones on the finest grid) and summing over all grids. Defined in this way, the unit of "work" is independent of the number of zones on the finest grid.

A single-grid iteration involves two relaxation sweeps, so the amount of work per iteration is 2. The work involved in a multigrid V-cycle starting, for instance, from a 32×16 grid, equals $4(1 + \frac{1}{4} + \frac{1}{16} + \frac{1}{64}) + \frac{1}{128} = 5.3$. In the latter number the computational effort in restriction and prolongation is neglected. On the other hand, in iterating with the single-grid scheme, there is one non-linear computation of $r^n(w)$ per 2 work units, as opposed to one per 5.3 for the multigrid scheme. Especially for the second-order accurate $r^n(w)$, the non-linear computation is more costly than the linear update of f_l^m in Eqs. (6) and (14). This amply compensates for the additional effort in restriction and prolongation.

Besides the amount of work, the cpu time per iteration was monitored. In comparing the two, one should be aware of the following efforts to keep the total amount of cpu time low:

(i) The inverse of A_{ij} required in the block Gauss–Seidel relaxation scheme is actually computed and stored in memory, so that it can be used in all relaxation sweeps per iteration on the same grid.

(ii) After a non-linear computation of $r^n(w)$, the block matrices required for $L^n(w)$ are not always computed. These matrices are frozen now and then, which avoids a new linearization, all the inversions for relaxation, and all the restrictions of matrices in the multigrid cycle. To control the freezing, a sequence of levels is defined, specifically, 10^{-1}, 3×10^{-2}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-6}, and 0. As soon as $\mathrm{RES}^n/\mathrm{RES}^0$, defined in Eq. (17), drops below the first level, the matrices are frozen until the next lower level is reached. Then all the matrices are computed again and frozen until the following level is reached, etc.

It may happen that RES passes through a certain level from below. Then the freezing is postponed for a few iteration cycles until RES drops below this level again. The choice of levels is somewhat arbitrary, and problem dependent. For our test problem, convergence histories in terms of work were practically identical with or without freezing.

Convergence histories of the first-order accurate solutions are shown in Fig. 1, for a 16×8, a 32×16, and a 64×32 grid. In all cases the multigrid scheme requires less work than the single-grid scheme, and the gain in efficiency is more dramatic as the number of zones increases. Figure 2 shows the convergence of the second-order accurate solutions. The total amount of work has increased, due to the incomplete linearization of the right-hand side $r^n(w)$.

For the quantitative analysis of the convergence results, a quantity τ is introduced,

$$\tau \equiv -\frac{\Delta \text{ work}}{\Delta \log_{10}(\mathrm{RES})}, \tag{22}$$

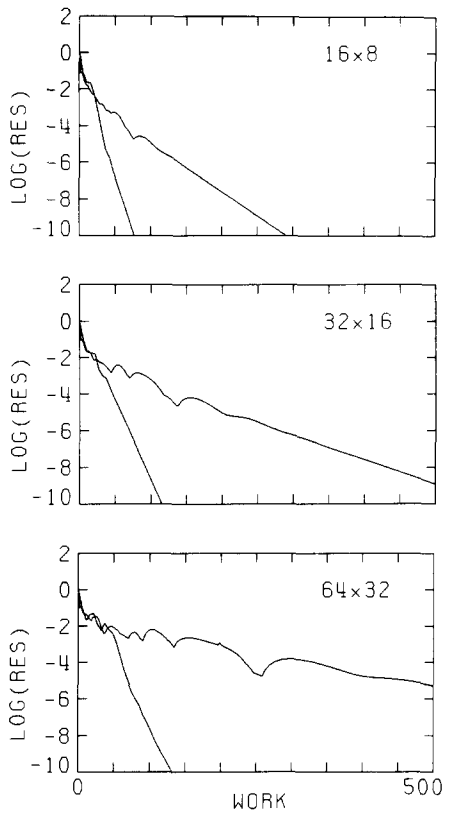

FIG. 1. Convergence histories of the first-order accurate solutions on 3 different grids. In all cases the multigrid scheme is faster than the single-grid scheme. For the multigrid scheme, the total amount of work increases only slowly with the number of zones.

being the amount of work required to bring the residual RES down by a factor 10. Table I shows the convergence data for the 3 grids, for first- and second-order accurate solutions. Three values of τ are given: the first one is computed for the drop of RES from its initial value to 10^{-5}, the second one for the drop from 10^{-5} to 10^{-10}, and the last one for the full drop from the initial value to 10^{-10}. The corresponding number of iterations is given as well. One iteration involves 2 relaxation sweeps in the single-grid case, and a full V cycle in the multigrid case. In both cases, the number of iterations is equal to the number of non-linear evaluations of the right-hand side $r^n(w)$. The cpu time required for these iterations is given in seconds on an Amdahl V7B. WRF is the work-reduction factor being the ratio of τ for the single-grid and for the multigrid scheme; CRF is the corresponding reduction-factor of the cpu time.

The single-grid runs show that the work increases with the number of zones N: roughly as $N^{0.5}$ for both the first-order and second-order accurate solutions. With

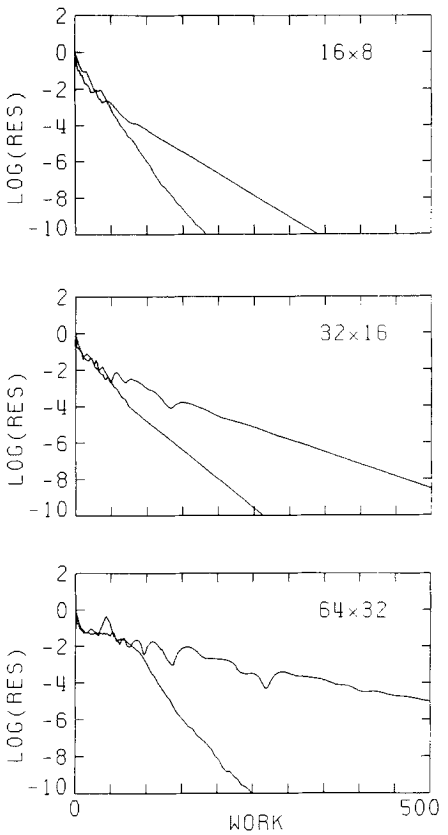

FIG. 2. As Fig. 1, but for the second-order accurate solutions. Due to the approximate linearization, convergence is somewhat slower than in the first-order case.

the multigrid scheme we find a dependence proportional to about $N^{0.1}$ in both cases, making the amount of iterations required to obtain a converged solution almost independent of the number of zones. For elliptic equations it is well known [5] that the number of iterations should be independent of N, if N is large. In the hyperbolic case such a result cannot be expected a priori, certainly not for solutions with discontinuities. Indeed, the weak dependence on N found here is encouraging.

We end this section with Figs. 3a and 3b, showing the distribution of the pressure coefficient on the bottom and top wall, obtained with the first-order and the second-order scheme, respectively. The pressure coefficient, defined as $C_p = (p - p_\infty)/\frac{1}{2}\rho_\infty u_\infty^2$, is computed from the values of ρ, u, and v at the wall obtained by zeroth-order extrapolation from the first-order solution and first-order extrapolation from the second-order solution. Thus, the same values are adopted as used for computing the fluxes at the wall, although formally one could go one order higher in accuracy when extrapolating towards the wall.

TABLE I

Convergence Data

Number of zones	Single-grid			Multigrid					Order of accuracy
	τ	Iter.	Cpu time (s)	τ	Iter.	Cpu time (s)	WRF	CRF	
16×8	20.4	51	7.1	7.2	7	3.1	2.8	2.3	1st
	37.9	95	11.1	8.4	8	2.2	4.5	4.9	
	29.1	146	18.2	7.8	15	5.3	3.7	3.4	
32×16	39.7	99	44	11.6	11	14	3.4	3.2	1st
	76.4	191	78	11.6	11	11	6.6	7.2	
	58.0	290	122	11.6	22	25	5.0	4.9	
64×32	91.8	230	380	13.8	13	65	6.6	5.8	1st
	143.3	358	558	12.6	12	48	11.4	11.6	
	117.6	588	938	13.2	25	113	8.9	8.3	
16×8	26.1	65	10.9	16.7	16	6.0	1.6	1.8	2nd
	41.8	105	15.1	19.8	19	5.3	2.1	2.8	
	34.0	170	26.0	18.3	35	11.3	1.9	2.3	
32×16	47.7	119	68	21.2	20	26	2.2	2.6	2nd
	73.9	185	96	31.4	29	32	2.4	3.0	
	60.8	304	164	26.3	49	58	2.3	2.8	
64×32	99.7	249	520	26.9	25	128	3.3	4.1	2nd
	142.7	357	713	23.0	22	94	6.2	7.6	
	121.2	606	1233	25.0	47	222	4.8	5.6	

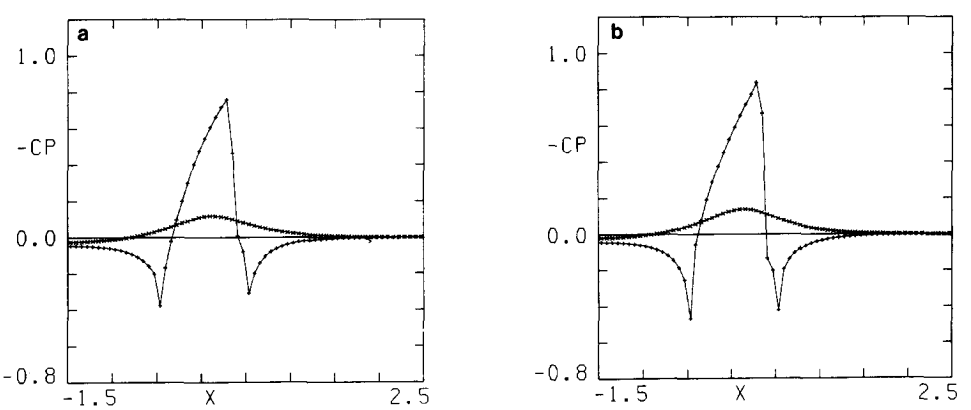

FIG. 3. (a) Pressure coefficient on bottom (+) and top wall (×) for a 4.2% circular arc and a free-stream Mach number 0.85, as computed from the first-order solution on a 64×32 grid. Thin-airfoil theory is applied to transfer the boundary conditions to the flow; (b) As in (a), but computed from the second-order accurate solution.

5. Concluding Remarks

It has been demonstrated that the multigrid technique can be successfully applied to compute a stationary transonic solution of the Euler equations. For the two-dimensional test problem the gain in efficiency with respect to a single-grid scheme is of the order $N^{0.4}$, where N is the total number of zones. Consequently, the number of iterations required to obtain a converged solution, whether first- or second-order accurate, increases only slowly with N: roughly as $N^{0.1}$. This result certainly justifies the additional effort of coding the multigrid scheme, which does not require more programming effort than a line-solver, even in FORTRAN (see [6]).

While these results are encouraging, two important issues must still be addressed.

The first one is: how to go through the initial phase of the iteration process? The searching phase of the SER scheme is almost explicit and the multigrid scheme solves the linear system (2) more accurately than is necessary. A single-grid solution would suffice. For our test problem the searching phase only takes a few iterations, but for a more difficult problem this phase may take much longer. In that case it would be more efficient to use a single-grid scheme until the residual has dropped to, say, a fraction 10^{-2} of its original value. One could also make successive grid refinements to speed up the searching phase.

The second issue is: the choice between saving on storage and saving on cpu time, which are mutually exclusive. The linearized block Gauss–Seidel relaxation scheme is very efficient, but requires the storage of all the necessary matrices. An alternative would be to change from a correction scheme to a full approximation storage (FAS) scheme [3]. The Gauss–Seidel scheme could then be implemented non-linearly and would require only the computation and inversion of the local main-diagonal block A_{ij} (see Eq. (19)).

A FAS scheme for the Euler equations has been successfully applied by Jameson [7]. His relaxation scheme is a four-stage method of the Runge–Kutta type, involving four non-linear updates of the residual per step but no matrix evaluations. In consequence the method requires little storage. The multigrid scheme by Ni [9] (see also [10]) does not exploit the full capacity of the multigrid technique, but merely helps to communicate local corrections to the solution through the entire computational domain. A symmetric non-linear Gauss–Seidel scheme has the same effect and probably performs just as well, without the use of multiple grids. To make such a scheme as simple as Ni's, the matrix A_{ij} in Eq. (18) can be replaced by its spectral radius; this, however, will result in a slow-down of the convergence. The off-diagonal matrices are not needed in a non-linear formulation. For a vectorized version one could use a checkerboard instead of a Gauss–Seidel scheme. Finally, we mention the work by Jespersen [8] that is based on the same ideas as the present method, i.e., upwind differencing through flux-vector splitting and linearized Gauss–Seidel relaxation. His scheme, however, adopts a nodal-point approach, whereas we use a finite-volume discretization with corresponding restriction and prolongation operators. His use of non-differentiable split fluxes might cause problems (see [1]). It is not clear whether his scheme can provide convergence

down to machine-zero or just to a level comparable to the discretization error. In the latter case, the numerical solution may still contain a systematic deviation from the steady state.

It is our hope that the present paper will contribute to the standard use of the multigrid technique in computing stationary solutions of the Euler equations, either in the form of a correction scheme or a FAS scheme. As the computational effort becomes less important, one may concentrate more on accuracy, for instance, by introducing a more sophisticated spatial discretization of the equations or by implementing adaptive grid refinement.

APPENDIX I: BOUNDARY CONDITIONS

The test problem requires a separate treatment of the boundary conditions at the inlet, at the outlet, and at the lower and upper wall. At the inlet the direction of the flow ($v_1 = 0$) and the total pressure p_0 are given; at the outlet the static pressure p_r is specified. The other state quantities at the boundary are obtained by extrapolation from the interior solution.

The natural variables for extrapolation are the characteristic variables. For the isenthalpic Euler equations used in our test problem the eigenvalues of df/dw and the corresponding characteristic variables are

$$\lambda_1 = u - c_- : dJ_1 = \frac{d\rho}{\rho} - \frac{du}{c_+} - (\gamma - 1)\frac{v\,dv}{c^2} = \frac{dp}{p} - \frac{du}{c_-} \tag{AI.1a}$$

$$\lambda_2 = u: \qquad dJ_2 = \frac{dv}{c}, \tag{AI.1b}$$

$$\lambda = u + c_+ : dJ_3 = \frac{d\rho}{\rho} + \frac{du}{c_-} - (\gamma - 1)\frac{v\,dv}{c^2} = \frac{dp}{p} + \frac{du}{c_+}. \tag{AI.1c}$$

Here $p = \rho c^2/\gamma = \rho c_+ c_-$ is the pressure, and the modified velocities of sound are given by

$$c_+ = -\gamma_3 u + (c^2/\gamma + \gamma_3^2 u^2)^{1/2}, \tag{AI.2a}$$

$$c_- = \gamma_3 u + (c^2/\gamma + \gamma_3^2 u^2)^{1/2}, \tag{AI.2b}$$

with $\gamma_3 = \gamma_1\gamma_2 = (\gamma - 1)/(2\gamma)$. From dg/dw similar expressions arise.

At the inlet the boundary values ρ_1, u_1, v_1 are computed from four conditions:

(i) the total enthalpy: $H = c_1^2/(\gamma - 1) + \frac{1}{2}(u_1^2 + v_1^2)$;

(ii) the total pressure: $p_0 = p_1\{(\gamma - 1)H/c_1^2\}^{\gamma/(\gamma-1)}$;

(iii) the direction of the flow: $v_1 = 0$;

(iv) one outgoing characteristic: $\Delta_1 J_1 = \Delta_1 p/p - \Delta_1 u/c_-$.

The boundary values are denoted by a subscript 1, the average values in the zone

next to the boundary have no subscript. The difference $\Delta_1 a = a - a_1$ for any variable a. In a first-order accurate scheme we set $\Delta_1 J_1 = 0$; for a second-order scheme $\Delta_1 J_1$ is taken equal to half the difference of ΔJ_1 over the other side of the zone: $\Delta_1 J_1 = \frac{1}{2}(\Delta J_1)_{2-1/2,j}$ for any j (see Appendix II). If the conditions (i) and (ii) are linearized, the following linear system of equations is obtained:

$$u\Delta_1 u + v\Delta_1 v + \frac{2c}{\gamma - 1}\Delta_1 c = 0, \tag{AI.3a}$$

$$\gamma_3 \frac{\Delta_1 p}{p} - \frac{\Delta_1 c}{c} = 0, \tag{AI.3b}$$

$$\Delta_1 v = v, \tag{AI.3c}$$

$$\frac{\Delta_1 p}{p} - \frac{\Delta_1 u}{c_-} = \Delta_1 J_1. \tag{AI.3d}$$

From this it follows that

$$p_1 = p + \rho \frac{(v^2 - uc_-\Delta_1 J_1)}{(1 + \gamma(uc_-/c^2))}. \tag{AI.4}$$

Condition (ii) now can be used to compute c_1^2 and by (i) and (iii), u_1 is found. Finally, the density $\rho_1 = \gamma p_1 / c_1^2$.

The implicit time discretization requires the linearization of the flux f_1 with respect to the conserved state quantities w in the interior zone. The only dependence of f_1 on w is through Eq. (AI.4), so we take

$$\frac{df_1}{dw} = \frac{df_1}{dp_1} \cdot \frac{dp_1}{dw} \approx \frac{df_1}{dp_1} \cdot \frac{dp}{dw}. \tag{AI.5}$$

At the outlet the static pressure is specified and there are two outgoing characteristics. We now have the conditions.

(i) total enthalpy: $H = c_r^2/(\gamma - 1) + \frac{1}{2}(u_r^2 + v_r^2)$;

(ii) static pressure: $p_r = p_\infty = 1/\gamma$;

(iii) $\Delta_r J_2 = \Delta_r v / c$;

(iv) $\Delta_r J_3 = \Delta_r p / p + \Delta_r u / c_+$.

The values at the boundary are denoted by the subscript r, the values in the zone next to the boundary have no subscript, and the difference $\Delta_r a = a_r - a$ for any a. Again we set $\Delta_r J_2 = \Delta_r J_3 = 0$ for the first-order scheme and extrapolate for the second-order scheme. We readily obtain

$$\begin{aligned} u_r &= u + c_+ \left(\frac{p - p_r}{p_r} + \Delta_r J_3 \right), \\ v_r &= v + c\Delta_r J_2. \end{aligned} \tag{AI.6}$$

From these values c_r^2 follows by condition (i) and $\rho_r = \gamma p_r / c_r^2$. For the derivative of the flux f_r with respect to the internal state quantities w we neglect the derivatives of c_+, $\Delta_r J_2$, and $\Delta_r J_3$.

At the lower and upper wall the boundaries are simulated by the introduction of an extra zone having reflected state quantities,

$$\begin{aligned} \rho' &= \rho, \\ u' &= u, \\ v' &= -v + 2u \tan \alpha. \end{aligned} \tag{AI.7}$$

Here ρ, u, and v are the quantities in the interior zone next to the boundary, whereas the primed quantities are the reflected ones. The angle α denotes the local angle of the wall with respect to the horizontal axis, which is zero except at the arc. From the primed quantities the split flux is computed and its linearization is found by using (AI.6).

APPENDIX II: SECOND-ORDER ACCURACY

Second-order accuracy in space can be obtained by assuming a set of basic quantities to be piecewise linear rather than piecewise constant [11]. The distribution of a quantity q in zone (i, j) can be described by

$$\begin{gathered} q(\xi, \eta) = q_{ij} + \xi(\overline{\Delta_x q})_{ij} + \eta(\overline{\Delta_y q})_{ij}, \\ \xi = \frac{x - x_{ij}}{\Delta_i x}, \qquad \eta = \frac{y - y_{ij}}{\Delta_j y}, \qquad |\xi| < \tfrac{1}{2}, \qquad |\eta| < \tfrac{1}{2}. \end{gathered} \tag{AII.1}$$

The x and y components $\overline{\Delta_x q}$ and $\overline{\Delta_y q}$ of the gradient within a zone can be found from neighboring values by finite differencing and averaging with a procedure that preserves monotonicity. It has been found in an earlier paper [1], that the differences of the characteristic variables (given by Eqs. (AI.1) for our test problem) are best suited for the determination of gradients. The computation of the x component of the gradient is described below. For the y component similar results are obtained.

In a given zone (i, j) the x component of the gradient can be found by first computing the differences ΔJ_k $(k = 1, 2, 3)$ across zone boundaries, for instance,

$$(\Delta_x J_1)_{i-1/2,j} = \frac{(p_{ij} - p_{i-1,j})}{(1/2)(p_{ij} + p_{i-1,j})} - \frac{(u_{ij} - u_{i-1,j})}{(1/2)(c_{-ij} + c_{-i-1,j})}. \tag{AII.2}$$

Here we have assumed an equidistant grid. The obvious way of determining $\bar{\Delta} \equiv (\overline{\Delta_x J_1})_{ij}$ from $\Delta_- \equiv (\Delta_x J_1)_{i-1/2,j}$ and $\Delta_+ \equiv (\Delta_x J_1)_{i+1/2,j}$ would be $\bar{\Delta} = \frac{1}{2}(\Delta_+ + \Delta_+)$, but this leads to oscillations at the foot or head of a discontinuity. To preserve monotonicity near discontinuities, the value of $\bar{\Delta}$ has to be limited to $O(\Delta x)$ [11].

A smooth switch due to Van Albada [12], is given by

$$\bar{\Delta} = \frac{\Delta_+(\Delta_-^2 + \varepsilon_a^2) + \Delta_-(\Delta_+^2 + \varepsilon_a^2)}{(\Delta_-^2 + \varepsilon_a^2) + (\Delta_+^2 + \varepsilon_a^2)}. \tag{AII.3}$$

It adopts the arithmetic mean of Δ_- and Δ_+ in smooth regions, but tends to the smaller of the two in the neighborhood of discontinuities. The bias ε_a prevents division by zero and may be tuned to prevent the clipping of smooth extrema. It is recommended to use such a smooth switch rather than a non-smooth one. The latter kind might cause problems during convergence.

In our test problem we used the L_1-norm of $\Delta_x J_k$ and $\Delta_y J_k$ ($k = 1, 2, 3$) to determine an optimum value for ε_a, resulting in $\varepsilon_a = 0.08\ \Delta x$ for all our second-order runs (note that $\Delta x = \Delta y$). This value was determined a posteriori. For practical purposes the optimum value of ε_a could be determined at several stages of the iteration process, for instance, just before the freezing of the linearization as described in Section 4.

At the lower and upper wall the differences across the boundary are computed with the aid of the reflected quantities. At the inlet and outlet the procedure described in Appendix I is followed.

The differences $(\Delta_x J_k)_{i\pm 1/2,j}$ and $(\Delta_y J_k)_{i,j\pm 1/2}$ are averaged by (AII.3) to find $\overline{(\Delta_x J_k)}_{i,j}$ and $\overline{(\Delta_y J_k)}_{i,j}$, which are then transformed into gradients for ρ, u, and v through Eqs. (AI.1a)–(AI.1c). We have thus obtained all the numerical values required to evaluate (AII.1), for $q = \rho$, u, or v.

For the computation of the split fluxes at the zone boundaries we now can use (AII.1) instead of zeroth-order extrapolation. For the x direction we obtain

$$\begin{aligned} \rho_{i\pm 1/2,j} &= \rho_{ij} \pm \tfrac{1}{2}\overline{(\Delta_x \rho)}_{ij}, \\ u_{i\pm 1/2,j} &= u_{ij} \pm \tfrac{1}{2}\overline{(\Delta_x u)}_{ij}. \\ v_{i\pm 1/2,j} &= v_{ij} \pm \tfrac{1}{2}\overline{(\Delta_x v)}_{ij}. \end{aligned} \tag{AII.4}$$

These values are used to compute the split fluxes at the zone boundaries: $f^{\pm}_{i\pm 1/2,j} = f^{\pm}(\rho_{i\pm 1/2,j}, u_{i\pm 1/2,j}, v_{i\pm 1/2,j})$.

The quantities $(df^{\pm}/dw)_{i\pm 1/2,j}$, needed in scheme (2), are computed only approximately. Their dependence on w_i, w_{i-1}, and w_{i+1} is simplified by assuming that

$$\left(\frac{\partial f^{\pm}}{\partial w}\right)_{i\pm 1/2,j} := \left(\frac{\partial f^{\pm}}{\partial w}\right)_{w_{i\pm 1/2,j}}, \tag{AII.5}$$

as in [1]. This incomplete linearization yields a linear system of the same form as for first-order space differencing, i.e., the block-pentadiagonal form (19). For large Δt, however, the second-order scheme does not turn into Newton's method, implying some loss of convergence speed.

ACKNOWLEDGMENT

It is a pleasure to thank Bram van Leer for his encouragement and detailed comments on the manuscript.

REFERENCES

1. W. A. MULDER AND B. VAN LEER, *J. Comput. Phys.* **59** (1985), 232–246.
2. B. VAN LEER AND W. A. MULDER, "Relaxation Methods for Hyperbolic Equations," *in* "Proceedings of the INRIA workshop on Numerical Methods for the Euler Equations for Compressible Fluids," Le Chesnay, France, Dec. 1983; Soc. Indus. Appl. Math., in press.
3. A. BRANDT, Lecture Notes in Mathematics Vol. 960, pp. 220–312, Springer-Verlag, Berlin, 1981.
4. B. VAN LEER, Lecture Notes in Physics Vol. 170, pp. 507–512, Springer-Verlag, Berlin, 1982.
5. W. HACKBUSCH, Lecture Notes in Mathematics Vol. 960, pp. 177–219, Springer-Verlag, Berlin, 1981.
6. F. C. THAMES, "Multigrid Applications to Three-Dimensional Elliptic Coordinate Generation," presented at the International Multigrid Conference, Copper Mountain, Colo., April 1983.
7. A. JAMESON, *Appl. Math. Comput.* **13** (1983), 327.
8. D. C. JESPERSEN, *Appl. Math. Comput.* **13** (1983), 357.
9. R. H. NI, "A Multiple Grid Scheme for Solving the Euler Equations," Amer. Inst. Aeronautics and Astronautics, paper 81–1025, June 1981.
10. R. V. CHIMA AND G. M. JOHNSON, "Efficient Solution of the Euler and Navier–Stokes Equations with a Vectorized Multiple-Grid Algorithm," Amer. Inst. Aeronautics and Astronautics paper 83–1893, July 1983.
11. B. VAN LEER, *J. Comput. Phys.* **23** (1977), 276.
12. G. D. VAN ALBADA, B. VAN LEER AND W. W. ROBERTS, *Astron. Astrophys.* **108** (1982), 76.