AIAA JOURNAL
Vol. 38, No. 10, October 2000

Factorized Implicit Upwind Methods Applied to Inviscid Flows at High Mach Number

Lorenzo Mottura,* Luigi Vigevano,† and Marco Zaccanti‡
Politecnico di Milano, 20158 Milan, Italy

The performance of several approximate factorization methods, coupled with a finite volume spatial discretization using Roe's approximate Riemann solver, are compared by means of numerical tests on a two-dimensional steady inviscid flow past a blunt body at Mach numbers ranging from 5 to 20. The comparisons are carried out evaluating, by numerical experiments, the optimal Courant number of each method. The alternating direction implicit and the lower–upper symmetric Gauss–Seidel methods result in the most efficient factorizations, in terms of CPU time. The former behaves smoothly with increasing Mach number, and its performance is not affected by the grid size. The latter may achieve higher efficiency but is strongly dependent on the number of relaxation steps performed, requiring optimization in terms of Mach number and grid size.

I. Introduction

AN IMPLICIT time integration is usually required to improve convergence toward steady state in time-marching, shock-capturing methods. This requirement is particularly true for high Mach number real gas flows where the practical stability limit of explicit methods is particularly severe. The large block banded linearized operators, resulting from the application of multistep implicit methods to multidimensional structured grids, are solved most efficiently by resorting to approximate factorization (AF). The implicit operator may be split along the grid directions, as in the early alternating direction implicit (ADI) schemes of Briley and McDonald[1] and Beam and Warming,[2] or into two lower–upper (LU) factors, as in the original LU schemes of Steger and Warming[3] and Jameson and Turkel.[4] The ADI scheme, without the addition of an appropriate amount of numerical damping, may, however, become unstable in three dimensions when formulated in delta form. Therefore, the LU factorization has become increasingly popular, and many improvements to this technique have been proposed in recent years.[5] Yoon and Jameson,[6,7] combined the LU factorization with a symmetric Gauss–Seidel (SGS) relaxation method to devise the more efficient LU-SGS scheme and applied the scheme to transonic flows. For incompressible flows, Whitfield[8] and Briley et al.[9] recently proposed a general LU-AF formulation that includes both the LU and LU-SGS schemes as particular cases, coupled with a Newton–Raphson procedure to reduce the factorization and linearization error.

The application of factorized implicit methods to high speed flows has generally been carried out in conjunction with upwind spatial discretisations. Yee et al.[10] used the ADI scheme. Others[11–13] employed the LU or LU-SGS schemes with simplified, diagonally dominant factors based on Jacobian spectral radii. Shuen[14] successfully applied a classical LU scheme with consistent upwind split of the flux Jacobians. However, no attempt has been made to verify the behavior of the implicit factorized methods when coupled with the same upwind discretisation scheme. Therefore, the purpose of this paper is to compare the numerical performance of different factorization approaches, coupled with a second-order Roe's Approximate Riemann Solver (ARS),[15] when applied to a well defined test case, the steady flow of perfect and equilibrium real gases past a blunt body configuration in the Mach number range $5 \le M \le 20$.

II. Approximate Factorizations

When a two-step implicit time integration, with time linearization of the numerical fluxes, is applied to the two-dimensional Euler equations discretized with a cell-centered finite volume scheme on a structured grid, the resulting algebraic system may be written

$$a^{n,1}_{i,j}\Delta U^n_{i-1,j} + a^{n,2}_{i,j}\Delta U^n_{i,j-1} + \big(I + a^{n,3}_{i,j} + a^{n,4}_{i,j} + a^{n,5}_{i,j} + a^{n,6}_{i,j}\big)\Delta U^n_{i,j} + a^{n,7}_{i,j}\Delta U^n_{i+1,j} + a^{n,8}_{i,j}\Delta U^n_{i,j+1} = b^n_{i,j} \qquad (1)$$

where $a^{n,k}_{i,j}$ is the k-th coefficient matrix at the time step n, multiplying the correspondent vector of the conservative variable variation $\Delta U^n_{i,j}$ relative to the volume (i, j), whereas $b^n_{i,j}$ is the summation of the incoming fluxes into the same volume. In the following discussion, the subscript i, j and the superscript n will not be indicated for the coefficient matrices, i.e., $a^{n,k}_{i,j}$ will read a^k. In compact form, Eq. (1) may be written

$$(I + A)\Delta \boldsymbol{U}^n = \boldsymbol{b} \qquad (2)$$

A. ADI/LU Factorization

The coefficient matrix of Eq. (2) may be approximately factorized as

$$(I + A) = (I + A_1)(I + A_2) + o(\Delta t^2) \qquad (3)$$

introducing an error of order Δt^2 that does not alter the formal accuracy of the time integration scheme.

Equation (2) is then replaced by the two systems:

$$(I + A_1)\boldsymbol{Y} = \boldsymbol{b}, \qquad (I + A_2)\Delta\boldsymbol{U}^n = \boldsymbol{Y} \qquad (4)$$

which may be easily solved by opportunely choosing how to split matrix A.

The ADI factorization is obtained carrying out the split along the coordinates of the computational grid:

$$\big[a^1\Delta U^n_{i-1,j} + (I + a^3 + a^5)\Delta U^n_{i,j} + a^7\Delta U^n_{i+1,j}\big] \times \big[a^2\Delta U^n_{i,j-1} + (I + a^4 + a^6)\Delta U^n_{i,j} + a^8\Delta U^n_{i,j+1}\big] = b^n_{i,j} \qquad (5)$$

This system is solved by means of a simple reorganization of the matrix entry ordering and inverting two block tridiagonal systems. However, the LU factorization is recovered by splitting A into a lower and an upper matrix:

$$\big[a^1\Delta U^n_{i-1,j} + a^2\Delta U^n_{i,j-1} + (I + a^5 + a^6)\Delta U^n_{i,j}\big] \times \big[(I + a^3 + a^4)\Delta U^n_{i,j} + a^7\Delta U^n_{i+1,j} + a^8\Delta U^n_{i,j+1}\big] = b^n_{i,j} \qquad (6)$$

Received 13 October 1998; revision received 13 March 2000; accepted for publication 20 March 2000. Copyright © 2000 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

*Aeronautical Engineer, Via La Masa 34.

†Assistant Professor, Dipartimento di Ingegneria Aerospaziale, Via La Masa 34. Member AIAA.

‡Postgraduate Fellow, Dipartimento di Ingegneria Aerospaziale; currently Ph.D. Candidate, MSU/NSF ERC for Computational Field Simulation, P.O. Box 9627, Mississippi State, MS 39762.

Because the sparseness of the nonfactorized coefficient matrix, this system is efficiently solved by simply performing two block back-substitutions.

B. LU-SGS Factorization

The coefficient matrix A may also be written[8] as the sum of a strictly lower, a diagonal, and a strictly upper matrix:

$$\left[a^1 \Delta U^n_{i-1,j} + a^2 \Delta U^n_{i,j-1}\right] + \left[(I + a^3 + a^4 + a^5 + a^6)\Delta U^n_{i,j}\right]$$
$$+ \left[a^7 \Delta U^n_{i+1,j} + a^8 \Delta U^n_{i,j+1}\right] = b^n_{i,j} \tag{7}$$

or, in compact notation:

$$(L + D + U)\Delta \boldsymbol{U}^n = \boldsymbol{b} \tag{8}$$

The system (8) can be factorized as

$$(D + L)D^{-1}(D + U)\Delta \boldsymbol{U}^n = \boldsymbol{b} \tag{9}$$

and solved in the two following steps:

$$(D + L)\boldsymbol{Y} = \boldsymbol{b}, \qquad (D + U)\Delta \boldsymbol{U}^n = D\boldsymbol{Y} \tag{10}$$

While the solution still needs two block back-substitutions, it requires the inversion of the diagonal blocks only once, allowing a reduction in operation counts when compared to the original LU factorization. Relations (10) can also be considered as the first step of the iterative SGS relaxation, which, in the present notation, may be written in the form

$$(D + L)\chi^{p+\frac{1}{2}} + U\chi^p = \boldsymbol{b}, \qquad L\chi^{p+\frac{1}{2}} + (D + U)\chi^{p+1} = \boldsymbol{b} \tag{11}$$

where p is the iteration index and $\chi^{p+1} = \Delta \boldsymbol{U}^n$. Using an iterative method, like the multiple step LU-SGS just described, allows the reduction of the factorization error.

C. LU-AF Factorization

Introducing a parameter α when partitioning the coefficient matrix, as proposed by Whitfield,[9] it is possible to seek to improve the convergence rate by finding a value of α that minimizes the error introduced by the AF. This general formulation is indicated as the LU-AF scheme.

The coefficient matrix may again be written as the sum of three matrices: a lower, a diagonal, and an upper one, as in the LU-SGS scheme. However, diagonal blocks are now present in the lower and upper matrices, and the system takes the form

$$\left[a^1 \Delta U^n_{i-1,j} + a^2 \Delta U^n_{i,j-1} + (1 - \alpha)(a^5 + a^6)\Delta U^n_{i,j}\right]$$
$$+ \left[(I + \alpha a^3 + \alpha a^4 + \alpha a^5 + \alpha a^6)\Delta U^n_{i,j}\right]$$
$$+ \left[(1 - \alpha)(a^3 + a^4)\Delta U^n_{i,j} + a^7 \Delta U^n_{i+1,j} + a^8 \Delta U^n_{i,j+1}\right] = b^n_{i,j} \tag{12}$$

which can still be solved with the relations (10). If α is taken to be equal to 0, the LU-AF reduces to the LU factorization, whereas for $\alpha = 1$, the scheme reduces to the one-step LU-SGS.

As will be shown later, for high Mach number flows, the introduction of the α parameter does not always accelerate the convergence of the implicit scheme.

D. Diagonally Dominant ADI Factorization

The ADI factorization may easily be modified to increase the diagonal dominance of the two factors[16] by decomposing the matrix A as

$$\left[a^1 \Delta U^n_{i-1,j} + a^7 \Delta U^n_{i+1,j}\right] + \left[(I + a^3 + a^4 + a^5 + a^6)\Delta U^n_{i,j}\right]$$
$$+ \left[a^2 \Delta U^n_{i,j-1} + a^8 \Delta U^n_{i,j+1}\right] = b^n_{i,j} \tag{13}$$

or, in compact notation

$$(N_\xi + D + N_\eta)\Delta \boldsymbol{U}^n = \boldsymbol{b} \tag{14}$$

The above system is then factorized in the same way as system (8); the resulting factorization, similar to Eq. (9) and hereinafter denoted diagonally dominant ADI (DDADI), still requires the inversion of two block tridiagonal systems.

Other ADI factorizations based on a diagonalization technique[17,18] are not considered here.

E. Newton–Raphson Subiterations

Newton–Raphson (NR) subiterations may be applied together with any of the above factorizations to reduce the time–linearization error. This method, enforcing the satisfaction of the original discrete equation, allows, when converged, eliminating any factorization error as well. The following discretized form of the Euler equations may be written

$$[I + A(\boldsymbol{U}^{n+1,m})](\boldsymbol{U}^{n+1,m+1} - \boldsymbol{U}^{n+1,m})$$
$$= -(\boldsymbol{U}^{n+1,m} - \boldsymbol{U}^n) + \theta \boldsymbol{b}(\boldsymbol{U}^{n+1,m}) + (1 - \theta)\boldsymbol{b}(\boldsymbol{U}^n) \tag{15}$$

where m is the NR iteration index and $A(\boldsymbol{U}^{n+1,m})$ and $\boldsymbol{b}(\boldsymbol{U}^{n+1,m})$ are, respectively, the coefficient matrix of the implicit system and the residual vector, evaluated at the m-th iteration. Choosing $\boldsymbol{U}^{n+1,1} = \boldsymbol{U}^n$, the first iteration reduces exactly to the solution of the implicit method written with the flux linearization.

III. Implicit Formulation Comparisons

A. Euler Equation Solver

The numerical finite volume method used to solve the Euler equations is based on the ARS proposed by Roe,[15] generalized for equilibrium real gas flow. Among all the possible generalizations, Vinokur's approach[19] has been used in this work because in previous studies[20] it proved to be the most consistent and to give the best results. A 5 species (O_2, N_2, NO, O, N) air model is employed for the equilibrium calculations.

Second-order space accuracy is achieved with Yee's symmetric scheme[21] and a Minmod 2 limiter. A certain amount of numerical dissipation is introduced, when required to reach convergence to steady state or to improve the convergence rate, applying, to all simple waves, the Harten–Hyman's entropy fix[22] as formulated by Yee.[10]

The steady state is reached starting from asymptotic uniform conditions in the flow field and letting the solution evolve up to convergence. Any attempt to use as a starting solution an analytically computed shock wave was found to be too dependent on the initial location of the shock itself.

B. Computed Results

The effectiveness of the approximate factorizations described in the previous paragraphs was evaluated considering a two-dimensional flow past a rounded cone (vertex angle 18 deg) at 10 km altitude in standard atmosphere ($\rho = 0.414$ kg/m^3, $P = 26{,}500$ Pa) at Mach numbers from 5 to 20 (Table 1). Air is considered both as a perfect gas and as an equilibrium real gas.

A first-order implicit integration was applied in all tests because that integration allows using higher values of the Courant number[14] than the trapezoidal scheme. However, the steady-state results are independent of both the time integration scheme and the selected time-step value. In order to assess the influence of the grid size on the results, two finite volume grids were employed: a 70 × 20 coarse grid and a 140 × 40 fine grid, obtained by halving the cells of the coarse one.

In Fig. 1, the quality of the steady-state results is presented by means of density isolines for a $M = 20$ solution computed on the 70 × 20 grid. Stagnation temperature values T_0 for the considered grids and air models are reported in Table 2.

For perfect gas, tests were carried out with all the factorization methods described, namely, ADI, DDADI, LU, LU-SGS with one

Table 1 Asymptotic test case conditions

Mach	Velocity, m/s
5	1497
10	2994
15	4490
20	5987

Table 2 Effect of grid size on stagnation temperature

		Temperature, K		
Air model	M	Grid size 70 × 20	Grid size 140 × 40	Δ, %
Perfect gas	5	1,339.9	1,337.5	0.2
(δ = 0)	10	4,699.1	4,673.4	0.5
	15	10,286.2	10,231.4	0.5
	20	18,041.9	18,034.9	0.0
Perfect gas	5	1,340.2	1,337.0	0.2
(δ = 0.2)	10	4,697.4	4,676.4	0.4
	15	10,347.8	10,263.3	0.8
	20	18,119.3	18,037.5	0.5
Equilibrium	5	1,255.8	1,253.9	0.2
	10	3,517.6	3,514.5	0.1
	15	5,793.4	nc	
	20	8,147.5	nc	

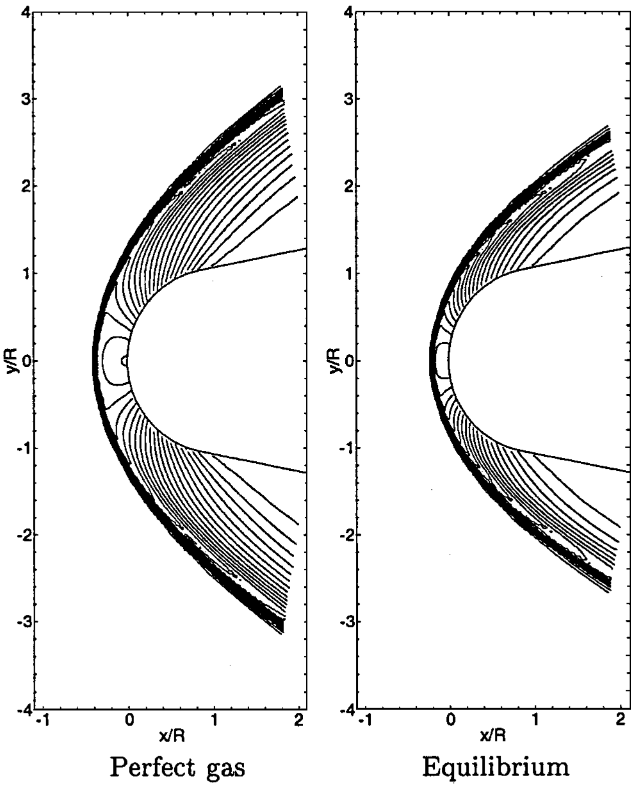

Fig. 1 Density isolines spaced at 0.1 kg/m^3 at M = 20.

relaxation step (LUSGS1) and with five relaxation steps (LUSGS5), LU-AF with $\alpha = 0.5$, and combining the one-step LU-SGS with five NR subiterations (NR5-LUSGS1).

For equilibrium real gas, tests were carried out only with the implicit methods that provided better performances in the perfect gas tests, namely, ADI, DDADI, LUSGS1, and LUSGS5.

Tables 3–5 summarize the results in terms of the Mach number (M), the optimal Courant number (CFL), the CPU time (HP C240 Series), and the iteration number to reach convergence (err = $\max(b^n_{i,j}/U^n_{i,j}) = 10^{-13}$). In addition, these tables report the accuracy ΔC with which the optimal CFL number is determined, as specified later. The value of the numerical dissipation parameter δ is also given. As already mentioned, a value $\delta > 0$ was used whenever doing so was necessary to reach convergence to machine accuracy or was useful to improve the convergence rate. However, the effect of the parameter on the computed stagnation temperature was found negligible (within 0.6% accuracy), as shown in Table 2.

C. Unit Computational Cost

Table 3 Comparison of eight factorization methods with convergence parameters of a coarse grid and perfect gas

Method	M	CFL	ΔC, %	CPU, s	Iteration	δ
ADI	5	11.4	0.9	34.9	540	0.20
	10	6.6	0.8	55.2	855	0.20
	15	3.8	2.3	122.1	1890	0.20
	20	1.9	2.6	221.0	3430	0.20
DDADI	5	7.6	0.7	63.9	945	0.20
	10	6.6	0.8	66.9	990	0.20
	15	4.6	0.4	110.9	1640	0.20
	20	2.1	1.2	224.6	3320	0.20
LU	5	2.0	2.6	154.1	2730	0.20
	10	1.5	1.4	141.1	2500	0.00
	15	1.7	2.1	242.7	4300	0.20
	20	1.0	4.8	340.4	6030	0.20
LUSGS1	5	15.6	1.3	30.4	560	0.20
	10	13.2	0.8	35.3	650	0.20
	15	2.5	2.8	163.9	3020	0.20
	20	1.3	1.9	275.7	5080	0.20
LUSGS5	5	15.4	1.0	29.4	395	0.20
	10	4.1	0.3	67.1	900	0.00
	15	6.7	2.6	72.3	970	0.20
	20	2.1	1.2	231.2	3100	0.20
LU-AF	5	13.0	0.8	33.8	535	0.20
	10	8.6	1.3	49.0	775	0.20
	15	2.3	2.3	199.4	3150	0.20
	20	1.2	2.1	350.6	5540	0.20
NR5-	5	19.6	0.5	76.4	305	0.00
LUSGS1	10	8.2	0.2	119.0	475	0.00
	15	3.6	2.5	518.5	2070	0.20
	20	2.8	2.7	568.6	2270	0.20
NR5-	5	7.9	1.3	128.6	665	0.20
LUSGS1,	10	6.1	0.9	169.2	875	0.20
Frozen	15	3.6	1.9	400.3	2070	0.20
Jacobian	20	2.8	2.7	442.9	2290	0.20

Table 4 Comparison of seven factorization methods with convergence parameters of a fine grid and a perfect gas

Method	M	CFL	ΔC, %	CPU, s	Iteration	δ
ADI	5	9.8	2.1	443	1,290	0.20
	10	4.6	2.4	933	2,720	0.20
	15	3.0	2.3	1,568	4,570	0.20
	20	1.3	3.8	2,882	8,400	0.20
DDADI	5	5.2	1.0	950	2,730	0.20
	10	4.5	1.0	1,096	3,150	0.20
	15	3.0	1.2	1,690	4,860	0.20
	20	1.3	3.8	2,849	8,190	0.20
LU	5	1.8	1.1	1,888	6,400	0.20
	10	1.7	2.6	2,094	7,100	0.20
	15	1.4	2.5	2,811	9,530	0.20
	20	0.7	6.7	5,055	17,140	0.20
LUSGS1	5	17.6	1.2	343	1,240	0.20
	10	3.8	1.2	979	3,540	0.20
	15	1.5	2.3	2,431	8,790	0.20
	20	1.2	4.2	2,522	9,120	0.20
LUSGS5	5	8.8	0.6	553	1,440	0.20
	10	5.5	2.0	887	2,310	0.20
	15	3.2	1.1	1,660	4,320	0.20
	20	1.2	4.2	3,519	9,160	0.20
LU-AF	5	6.4	0.8	599	1,870	0.20
	10	2.8	1.6	1,434	4,480	0.20
	15	1.4	2.4	2,987	9,330	0.20
	20	0.7	6.7	6,484	20,250	0.20
NR5-	5	11.3	1.8	1,767	1,350	0.20
LUSGS1	10	4.9	0.9	3,377	2,580	0.20
	15	1.8	2.0	10,053	7,780	0.20

A comparison of the unit computational cost associated with the different methods is given in Fig. 2. The average CPU times required for one implicit iteration on the 70 × 20 coarse grid are shown for each of the tested methods. The values are normalized with respect to the CPU time required by the LUSGS1 factorization for perfect gas, which features the lowest computational cost/iteration ratio.

The LU scheme requires a smaller CPU time with respect to the ADI and DDADI factorizations. The reasons for the higher efficiency of the LU method are twofold: it does not need to reorganize the matrix element ordering, and it employs block back-substitutions that are computationally more efficient than the solution of block tridiagonal systems.

Table 5 Comparison of various factorization methods and grids using equilibrium real gas

Grid/ method	M	CFL	ΔC, %	CPU, s	Iteration	δ
			Coarse grid			
ADI	5	10.4	0.5	65	630	0.20
	10	5.0	1.1	112	1,090	0.20
	15	0.9	3.8	605	5,870	0.25
	20	0.4	1.4	1,369	13,270	0.25
DDADI	5	5.9	0.9	132	1,255	0.20
	10	1.5	0.7	360	3,410	0.20
	15	0.9	0.8	637	6,040	0.25
	20	0.4	1.4	1,402	13,280	0.25
LUSGS1	5	16.4	3.0	57	610	0.20
	10	3.1	1.0	180	1,920	0.00
	15	0.9	2.0	565	6,020	0.25
	20	0.4	1.4	1,230	13,110	0.25
LUSGS5	5	11.2	0.4	65	590	0.20
	10	2.6	0.9	249	2,260	0.00
	15	0.9	2.0	671	6,100	0.25
	20	0.4	1.4	1,459	13,270	0.25
			Fine grid			
ADI	5	5.9	3.4	851	1,910	0.20
	10	1.0	1.8	3,427	7,690	0.00
LUSGS1	5	5.9	3.4	794	2,090	0.20
	10	0.9	2.0	3,018	7,940	0.00
LUSGS5	5	6.2	3.3	872	1,750	0.20
	10	0.9	2.0	3,942	7,910	0.00

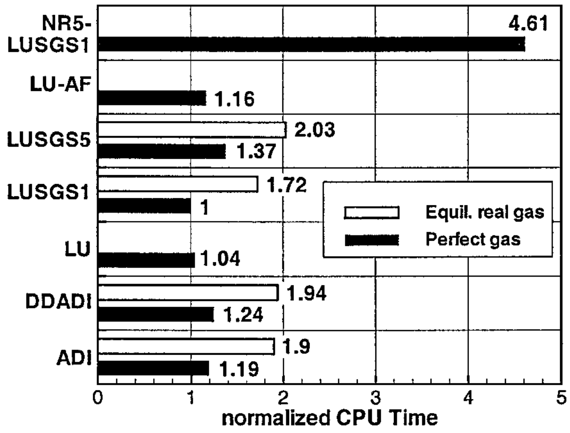

Fig. 2 Computational cost per iteration.

The LUSGS1 allows a further CPU time reduction because it needs to invert the diagonal blocks only once. The general LU-AF method results are computationally more demanding because the method loses the advantage of a single inversion of the diagonal blocks and introduces additional matrix–vector multiplications. Successive SGS iterations do not highly affect the unit computational costs because they do not require recomputing the coefficients of the Jacobian matrix and the residues at each subiteration. Finally, carrying out 5 NR iterations is almost equivalent to performing 5 LUSGS1 iterations, with a corresponding large increase in the unit computational cost.

D. Optimal CFL Selection

The results presented in Tables 3–5 were achieved considering the optimal CFL for each of the applied methods. The following procedure was adopted to determine empirically the optimal CFL: for all of the examined configurations (Mach number, grid, and factorization method), several calculations were run increasing the CFL number until it was no longer possible to reach convergence to machine accuracy. Among the converged solutions, the one featuring the highest convergence rate was selected, and the corresponding CFL number was considered to be the optimal one. The accuracy of this value was computed according to the sampling intervals.

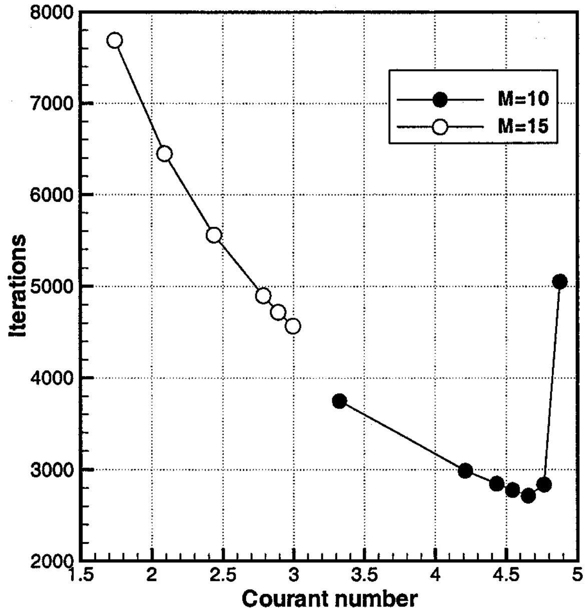

Fig. 3 Optimal CFL selection with perfect gas, fine grid, and ADI method.

Because the errors introduced by factorization and linearization are a function of the time-step itself, the largest CFL value allowing a converged solution on a given grid is not necessarily the best choice in terms of convergence rate. An example is shown in Fig. 3, where the number of iterations needed to reach convergence with the ADI method on the fine grid at $M = 10$ and 15 is plotted as a function of the Courant number. While at $M = 15$ the maximum converged CFL is, indeed, the optimal one, at $M = 10$, the optimal CFL is lower than the maximum value allowing convergence.

In addition to the factorization error, there is another reason for limiting the optimal CFL value, namely, the flux linearization introduced in the discretisation of the flow equations. Such linearization is performed both in time and in solving an approximate Riemann problem at each cell interface (through Roe's ARS) and affects particularly the initial transient phase of the calculation when, starting from a uniform undisturbed field, the bow shock is formed and moves upstream till it reaches its steady-state location. When the shock moves from one cell to another, an exceedingly high value of the CFL number may give rise to spurious oscillations in the solution, leading to physically impossible negative values for the gas density and internal energy, and thus preventing reaching convergence.

E. Rate of Convergence

From the convergence histories shown in Figs. 4 and 5, the initial transient phase may be clearly detected. The initial phase is characterized by strong peaks of the maximum error, as opposed to the solution refinement phase, which features a much smoother behavior and a strong reduction of the residual. All of the convergence histories presented herein refer to optimal CFL runs.

Observe that the NR5-LUSGS1 method, which practically eliminates the error due to the time–linearization, does not feature significant residual peaks during the initial transient on the coarse grid. However, this behavior is not confirmed on the fine grid calculations (Fig. 5), suggesting that the main source of error during the transient phase is Roe's linearization itself.

The addition of a certain amount of artificial dissipation ($\delta = 0.2$) yields a reduction of the error peaks during the transient phase on both grids, as shown in Figs. 6 and 7. Such a reduction is particularly impressive for $M \leq 10$ and on the fine grid. In general, the solution refinement phase is also favorably influenced by $\delta > 0$, leading to an increase of the convergence rate. There are some exceptions to this behavior, like the LUSGS5 method at $M = 10$ on the coarse grid (compare Fig. 4 with Fig. 6).

Increasing the Mach number yields an obvious reduction of the convergence rate, as shown for the ADI factorization in Figs. 8 and 9.

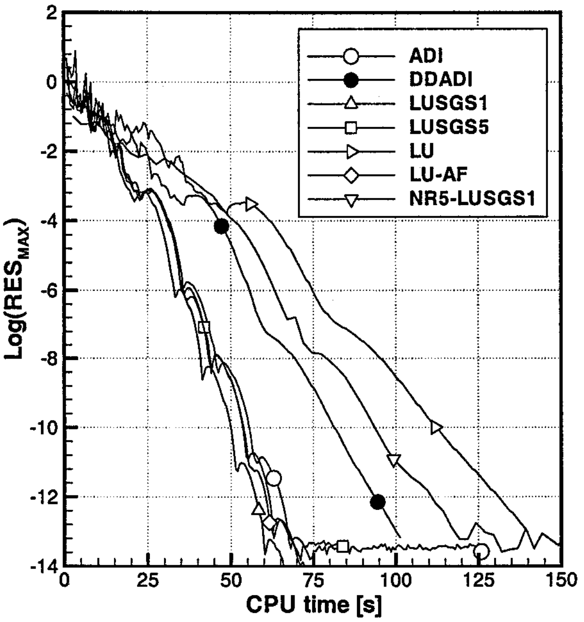

Fig. 4 Convergence history with perfect gas, coarse grid, $M = 10$, and $\delta = 0$.

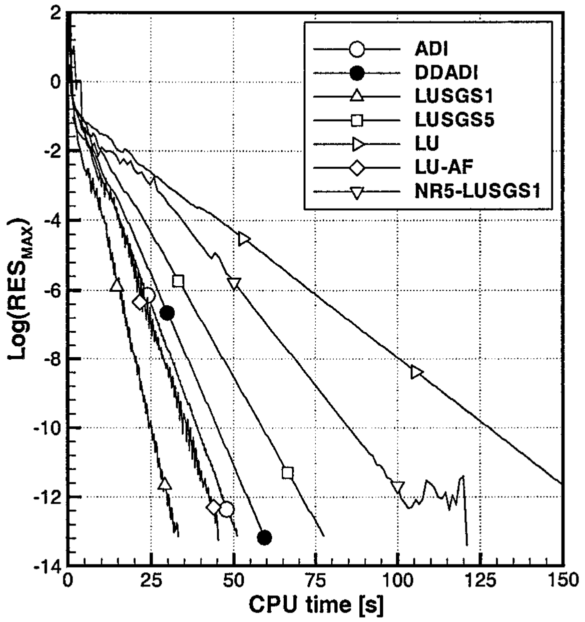

Fig. 6 Convergence history with perfect gas, coarse grid, $M = 10$, and $\delta = 0.2$.

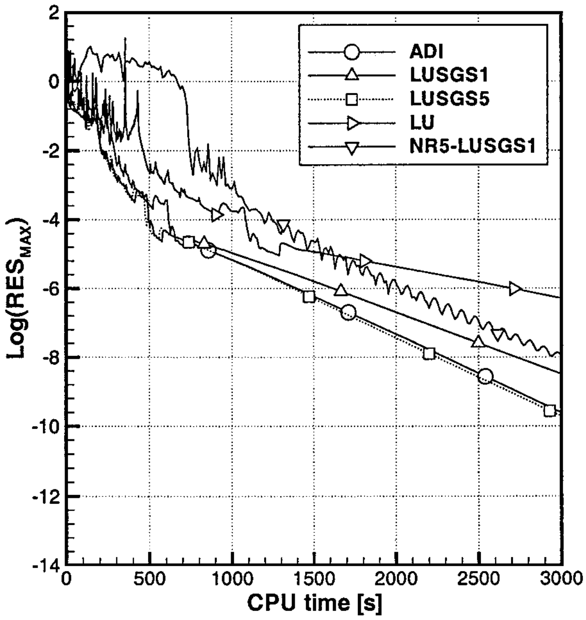

Fig. 5 Convergence history with perfect gas, fine grid, $M = 10$, and $\delta = 0$.

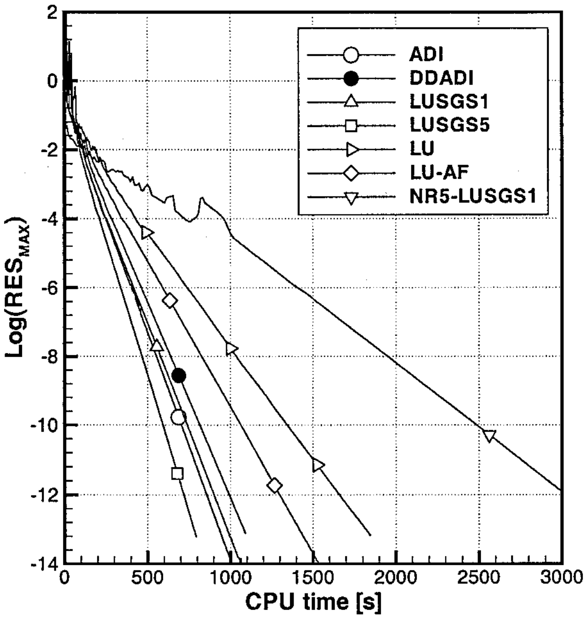

Fig. 7 Convergence history with perfect gas, fine grid, $M = 10$, and $\delta = 0.2$.

The influence of the grid size on the qualitative trend is rather small in this case.

Since the main objective of the present numerical experiments was to assess the computational efficiency of the examined factorization methods, the content of Tables 3–5 is displayed in graphical form in Figs. 10–12 in order to show the convergence performance in terms of overall CPU time as a function of the Mach number.

The analysis of the perfect gas performance with increasing Mach number is not obvious, although some trends may be clearly observed. First, the NR subiteration approach does not achieve good results on either the coarse or the fine grid. To see the lack of good results, compare the NR5-LUSGS1 and LUSGS1 plots. The partial elimination of the time-linerization error, obtained carrying out 5 subiterations, generally allows increasing the CFL number and, correspondingly, reducing the required number of iterations. However, such a reduction is not sufficient to balance the large increase in computational costs per iteration. Freezing the Jacobian (FJ) matrix at its first subiteration value (NR5-LUSGS1-FJ) leads only to a marginal improvement in the performance at high Mach numbers.

The ADI group of factorizations (ADI and DDADI) presents a rather smooth increase of CPU time with Mach number, independent of the grid size. The DDADI method becomes slightly superior to the ADI only at the highest Mach numbers ($M \geq 15$ on the coarse grid and $M = 20$ on the fine grid).

Among the LU group of factorizations (LU, LUSGS1, LUSGS5, and LU-AF), the LU-SGS approach is the most successful. The standard LU method achieves poor results at all Mach numbers and on both grids. The LU-AF scheme performs well at low M, but its efficiency decreases markedly with increasing Mach numbers, becoming worse than that of the LU method at $M = 20$. This effect is emphasized on the fine grid.

The addition of 5 Gauss–Seidel iterations, which reduces the factorization error at the expense of a higher unit computational cost, is sometimes successful in reducing the overall CPU time. Comparing the results obtained with LUSGS1 and LUSGS5 on the two different

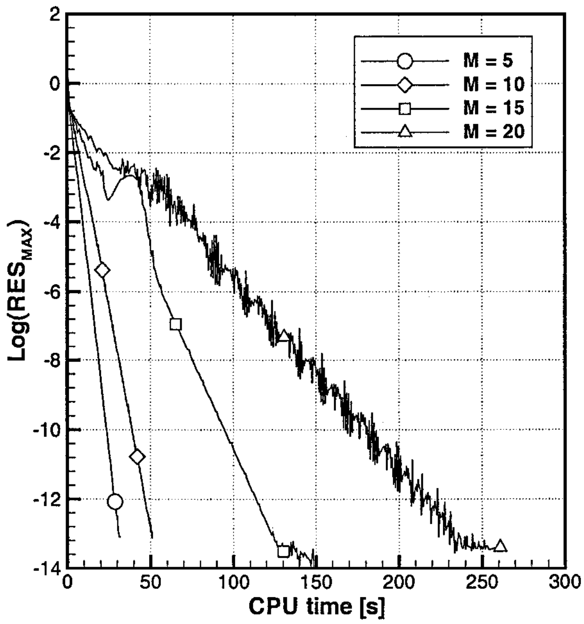

Fig. 8 Convergence history with perfect gas, coarse grid, ADI method, and $\delta = 0.2$.

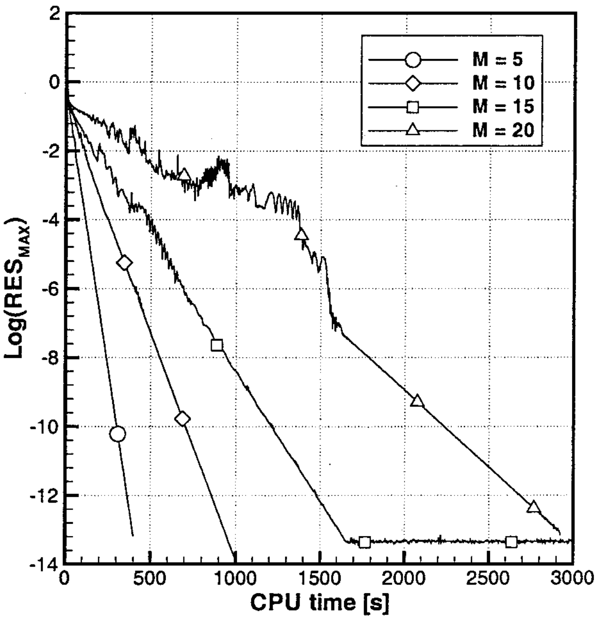

Fig. 9 Convergence history with perfect gas, fine grid, ADI method, and $\delta = 0.2$.

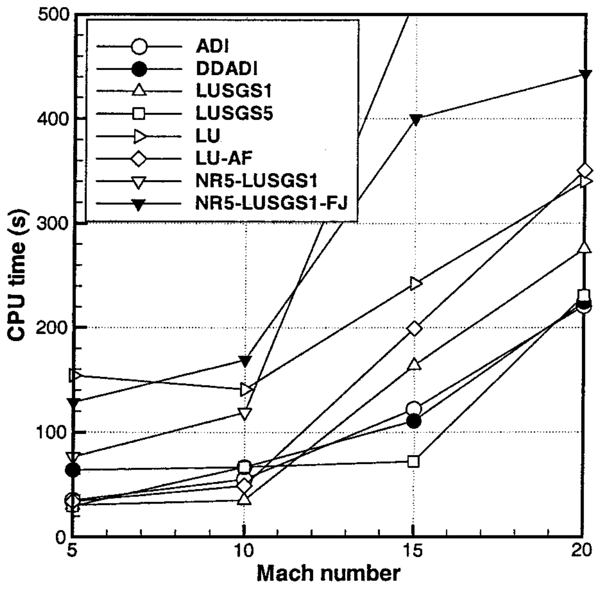

Fig. 10 CPU time variation with Mach number using perfect gas and a coarse grid.

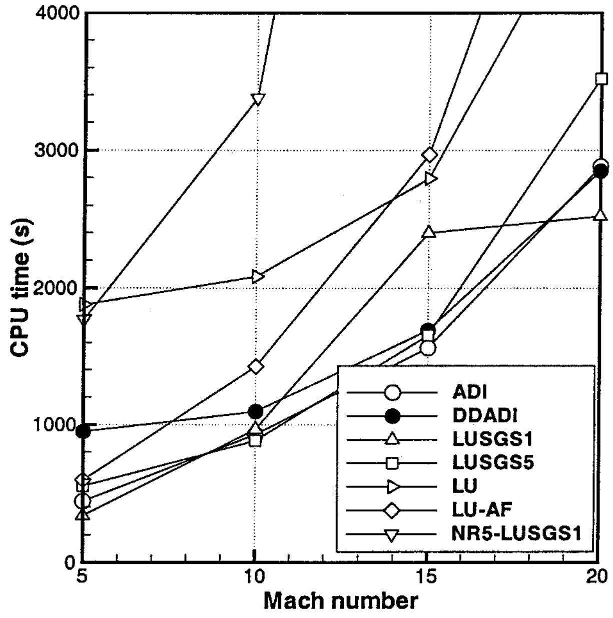

Fig. 11 CPU time variation with Mach number using perfect gas and a fine grid.

grids, similar trends can be noticed. On the coarse grid, LUSGS5 improves the efficiency over LUSGS1 at a high Mach ($M \geq 15$), whereas on the fine grid, it performs better in the intermediate Mach range ($10 \leq M \leq 15$). A marked reduction of the performance of the LUSGS1 scheme at $M = 15$ can be observed, independent of the grid resolution. This reduction could be due to a grid-shock alignment effect. An exceedingly tight alignment between the shape of the bow shock and that of the grid lines, occurring at this Mach number, may produce oscillations in the solution that are more effectively damped when using Gauss–Seidel subiterations. In such a case, the addition of a single subiteration may noticeably improve the results. For instance, on the coarse grid, a LUSGS2 scheme reaches convergence in half the CPU time needed by the LUSGS1 method.

The tested factorizations behave in much the same way when air is treated as an equilibrium real gas (Fig. 12). The ADI method performs better than the DDADI, whereas the LUSGS1 gives better results than the LUSGS5, with the LUSGS1 method generally being the most efficient. On the coarse grid, notice that the optimal CFL number values are virtually the same for all the factorizations when $M \geq 15$. The same results happen on the fine grid already at $M = 5$. In these conditions, it is more convenient to use the LUSGS1 factorization, which features the best computational cost/iteration ratio. For Mach number values greater than 10, it was not possible to reach convergence to machine accuracy on the fine grid, even with large reductions of the CFL number. The calculations converged to $\mathrm{err} = 10^{-6}$, a value still acceptable from an engineering viewpoint, but one not reported here for consistency.

Attempts were made to improve the convergence rate of several methods using a variable time step, either with a prescribed sequence or as a piecewise constant or a piecewise linear function of the maximum error. However, all of the tested approaches were found extremely dependent on both the asymptotic conditions and the grid resolutions. Owing to the influence of the grid size on the error behavior, the overall convergence rate could be improved by obtaining a solution on a coarse grid (to reduce the start-up problems) and interpolating it on the desired fine grid as the initial condition for the final accurate solution.

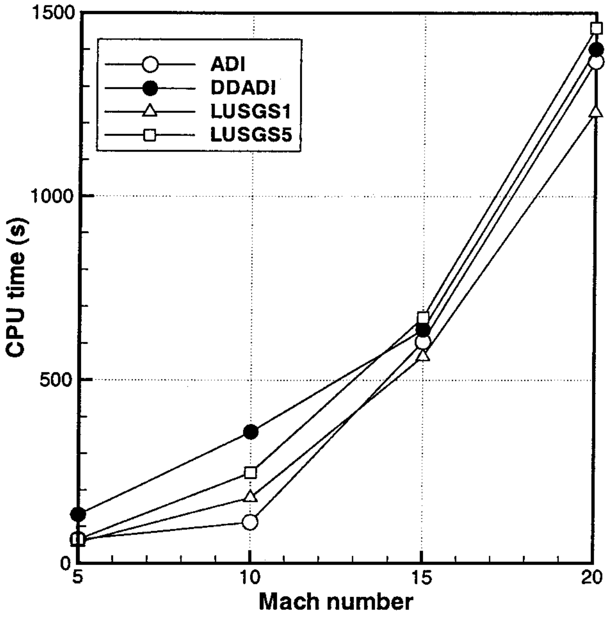

Fig. 12 CPU time variation with Mach number using equilibrium real gas.

IV. Conclusion

Some ADI-like factorizations making use of a diagonalization technique[17,18] have not been considered in the present work. Also, the application to a single test case, although significant for high Mach flow, does not allow for a definite judgment. However, the numerical experiments presented herein for high-Mach-number steady flows are deemed interesting because the use of a single spatial discretisation scheme (Roe's ARS) has allowed a clear comparison among several factorization methods.

The most efficient methods, in terms of CPU time, are the ADI (ADI and DDADI) and the LU-SGS (LUSGS1 and LUSGS5) factorization groups, with comparable performances.

The results obtained with the ADI-like factorizations for perfect gas are rather grid independent and vary smoothly with increasing Mach number. However, the performance of the LU-SGS approach, although often representing the absolute best, is strongly influenced by the number of Gauss–Seidel iterations carried out. This influence depends on both the Mach number and the grid size. The present analysis suggests that an intermediate approach between LUSGS1 and LUSGS5 (carrying out just two relaxation iterations, for instance) could lead to a smoother trend.

The performance of all the tested factorizations is significantly improved by adding a certain amount of numerical dissipation. This improvement is especially evident when the fine grid is used. The selected value for the dissipation coefficient $\delta = 0.2$ is customary in the literature and eventually does not markedly alter the quantitative behavior of the solutions as monitored by the stagnation temperature values.

For equilibrium real gas, the convergence rates obtained with the tested factorizations are very similar, suggesting that the linearization error is much more binding than the factorization error. Generally, the most efficient approach is to use LUSGS1, for its low computational cost/iteration ratio. However, the ADI factorization gives comparable results.

In conclusion, for high-Mach-number two-dimensional steady flow, the Euler equations may be integrated in an efficient and robust way using an ADI-like factorization with time linearization. An improvement in the performance may be achieved with the LU-SGS factorization at the expense of a case-by-case tuning of the number of relaxation steps.

Acknowledgments

The authors are indebted to the referees' criticisms and suggestions that led to more accurate results.

References

[1] Briley, W. R., and McDonald, H., "Solution of the Multidimensional Compressible Navier–Stokes Equations by a Generalized Implicit Method," *Journal of Computational Physics*, Vol. 24, No. 4, 1977, pp. 372–397.

[2] Beam, R., and Warming, R. F., "An Implicit Factored Scheme for the Compressible Navier–Stokes Equations," *AIAA Journal*, Vol. 16, No. 4, 1978, pp. 393–402.

[3] Steger, J. L., and Warming, R. F., "Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite Difference Methods," *Journal of Computational Physics*, Vol. 40, No. 2, 1981, pp. 263–293.

[4] Jameson, A., and Turkel, E., "Implicit Schemes and L-U Decompositions," *Mathematics of Computation*, Vol. 37, No. 156, 1981, pp. 385–397.

[5] Yoon, S., and Kwak, D., "Implicit Navier–Stokes Solver for Three-Dimensional Compressible Flows," *AIAA Journal*, Vol. 30, No. 11, 1992, pp. 2653–2659.

[6] Jameson, A., and Yoon, S., "Lower-Upper Implicit Schemes with Multiple Grids for the Euler Equations," *AIAA Journal*, Vol. 25, No. 7, 1987, pp. 929–935.

[7] Yoon, S., and Jameson, A., "Lower-Upper Symmetric Gauss–Seidel Method for the Euler and Navier–Stokes Equations," *AIAA Journal*, Vol. 26, No. 9, 1988, pp. 1025–1026.

[8] Whitfield, D. L., "Newton-Relaxation Schemes for Non-Linear Hyperbolic Systems," Aerospace Engineering Dept., MSSU-EIRS-ASE-90-3, Mississippi State Univ., Mississippi State, MS, Oct. 1990.

[9] Briley, W. R., Neerarambam, S. S., and Whitfield, D. L., "Implicit Lower-Upper/Approximate-Factorization Schemes for Incompressible Flows," *Journal of Computational Physics*, Vol. 128, No. 1, 1996, pp. 32–42.

[10] Yee, H. C., Klopfer, G. H., and Montagné, J.-L., "High-Resolution Shock-Capturing Schemes for Inviscid and Viscous Hyperbolic Flows," *Journal of Computational Physics*, Vol. 88, No. 1, 1990, pp. 31–61.

[11] Chen, C. L., McCrosky, W. J., and Obayashi, S., "Numerical Solutions of Forward-Flight Rotor Flow Using an Upwind Method," AIAA Paper 89-1846, June 1989.

[12] Grasso, F., and Marini, M., "Lower Upper Implicit Total Variation Diminishing Solution of Viscous Hypersonic Flows," *AIAA Journal*, Vol. 30, No. 9, 1992, pp. 2184–2185.

[13] Ju, Y., "Lower-Upper Scheme for Chemically Reacting Flow with Finite Rate Chemistry," *AIAA Journal*, Vol. 33, No. 8, 1995, pp. 1418–1425.

[14] Shuen, J.-S., "Upwind Differencing and LU Factorization for Chemical Non-Equilibrium Navier–Stokes Equations," *Journal of Computational Physics*, Vol. 99, No. 2, 1992, pp. 233–250.

[15] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372.

[16] McCormack, R. W., "A New Implicit Algorithm for Fluid Flow," AIAA Paper 97-2100, June 1997.

[17] Pulliam, T. H., and Chaussee, D. S., "A Diagonal Form of an Implicit Approximate-Factorization Algorithm," *Journal of Computational Physics*, Vol. 32, No. 2, 1981, pp. 347–363.

[18] Klopfer, G. H., Van der Wijngaart, R. F., Hung, C. M., and Onufer, J. T., "A Diagonalized Diagonal Dominant Alternating Direction Implicit (D3ADI) Scheme and Subiteration Correction," AIAA Paper 98-2824, June 1998.

[19] Vinokur, M., and Montagné, J.-L., "Generalized Flux-Vector Splitting and Roe Average for an Equilibrium Real Gas," *Journal of Computational Physics*, Vol. 89, No. 2, 1990, pp. 276–300.

[20] Mottura, L., Vigevano, L., and Zaccanti, M., "An Evaluation of Roe's Scheme Generalizations for Equilibrium Real Gas Flows," *Journal of Computational Physics*, Vol. 138, No. 2, 1997, pp. 354–399.

[21] Yee, H. C., "Construction of Explicit and Implicit Symmetric TVD Schemes and Their Applications," *Journal of Computational Physics*, Vol. 68, No. 1, 1987, pp. 151–179.

[22] Harten, A., and Hyman, J. M., "Self-Adjusting Grid Methods for One-Dimensional Hyperbolic Conservation Laws," *Journal of Computational Physics*, Vol. 50, No. 2, 1983, pp. 235–269.

K. Kailasanath
Associate Editor