INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 22, 1023–1035 (1996)

EFFECTS OF IMPLICIT PRECONDITIONERS ON SOLUTION ACCELERATION SCHEMES IN CFD

J. O. HAGER AND K. D. LEE

Aeronautical and Astronautical Engineering, University of Illinois, Urbana, IL 61801, U.S.A.

SUMMARY

Several solution acceleration techniques, used to obtain steady state CFD solutions as quickly as possible, are applied to an implicit, upwind Euler solver to evaluate their effectiveness. The implicit system is solved using either ADI or ILU and the solution acceleration techniques evaluated are quasi-Newton iteration, Jacobian freezing, multigrid and GMRES. ILU is a better preconditioner than ADI because it can use larger time steps. Adding GMRES does not always improve the convergence. However, GMRES preconditioned with ILU and multigrid can take advantage of Jacobian freezing to produce an efficient scheme that is relatively independent of grid size and grid quality.

KEY WORDS: convergence acceleration; preconditioning; multigrid; GMRES; ADI; ILU

INTRODUCTION

Computational fluid dynamics has seen great advances in the range of aerodynamic problems that can be solved. Higher-order upwind schemes, improved turbulence models and advanced chemistry models have improved the flow model accuracy for complex flows, while multiblock grids and chimera schemes have improved the spatial accuracy for complex geometries. However, not many advances have been made in the solution methods for these advanced capability solvers. Solution acceleration schemes, which are used to obtain steady state CFD solutions as quickly as possible, are now in demand.

There are three broad classes of techniques used for solution acceleration: increased signal propagation, optimization, and reduced computation. One of the most common increased signal propagation schemes is implicit time integration. The increased stability of implicit schemes compared with explicit schemes allows a much larger time step to be used. The high cost of directly inverting the implicit correction matrix has led to the use of approximate factorization schemes such as ADI and ILU to obtain simpler matrix systems. An extension of implicit time integration is switched evolution/relaxation. The usual implicit time integration scheme is converted to a Newton iteration scheme as the solution converges by letting the time step approach infinity.

Other popular increased signal propagation schemes are local time stepping and multigrid. Local time stepping uses the largest time step allowable at each control point to increase the signal propagation through each cell. Multigrid uses the faster signal propagation across the domain on a coarse grid while maintaining the higher solution accuracy produced on a fine grid.

The optimization-based schemes are those related to conjugate gradient schemes. The non-linear GMRES scheme is very similar to numerical optimization. It obtains the optimum solution in a Krylov subspace that is spanned by the search direction vectors. The search directions and the optimum step size are obtained using finite difference sensitivity calculations.

CCC 0271–2091/96/101023–13

Received March 1995
Revised August 1995

One reduced computation scheme that is only applicable to implicit schemes is Jacobian freezing. As the solution converges, the correction matrix becomes almost constant. Freezing the Jacobians and therefore the correction matrix can significantly reduce the CPU time per iteration. This is also particularly useful when GMRES is used, because the Jacobians can be frozen during the sensitivity analysis and not only from one iteration to the next.

Preconditioning is an important concept in solution acceleration. Iterative schemes will converge quicker when the eigenvalues of the system are clustered. Preconditioning is used to modify the basic system in order to improve the performance of other solution schemes. Because the best system is the identity operator, where all of the eigenvalues are identically one, most preconditioning is accomplished by using an approximate inverse operator to premultiply the system.

This paper presents a summary of the solution techniques, followed by a parametric study of the effectiveness of the various schemes. Several grids are used to examine the relative strengths of the methods. The baseline CFD code is a first-order, upwind Euler equation solver that uses Steger/Warming flux vector splitting for both the residual and implicit correction operators.

TIME INTEGRATION OF THE EULER EQUATIONS

The two-dimensional, unsteady Euler equations in curvilinear co-ordinates can be written in the form

$$\frac{\partial h\mathbf{u}}{\partial t} + \mathbf{r}(\mathbf{u}) = \mathbf{0}, \qquad \mathbf{r}(\mathbf{u}) = \frac{\partial \mathbf{f}(\mathbf{u})}{\partial \xi} + \frac{\partial \mathbf{g}(\mathbf{u})}{\partial \eta}, \tag{1}$$

where h is the cell area, $\mathbf{u} = (\rho, \rho u, \rho v, \rho e)^{\mathrm{T}}$, $\mathbf{r}(\mathbf{u})$ is the residual and $\mathbf{f}(\mathbf{u})$ and $\mathbf{g}(\mathbf{u})$ are the flux vectors in the ξ- and η-direction respectively. For steady state solutions, $\mathbf{r}(\mathbf{u}) = \mathbf{0}$ and we are interested in driving the residual to zero as quickly as possible. The equations will be presented in finite difference form. However, the calculations are performed using a cell-centred finite volume scheme.

Time integration schemes are generally applied in an iterative fashion as

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta\mathbf{u}, \tag{2a}$$

$$\mathbf{N}\Delta\mathbf{u} = -\mathbf{L}(\mathbf{u}^n), \tag{2b}$$

where $\mathbf{N}$ is a linear operator and $\mathbf{L}(\mathbf{u})$ is a non-linear operator given by

$$\mathbf{L}(\mathbf{u}) = \Delta\tau\mathbf{r}(\mathbf{u}), \qquad \Delta\tau = \Delta t/h. \tag{3}$$

For fully implicit schemes the correction operator can be rewritten as

$$\mathbf{N} = \mathbf{I} + \Delta\tau\partial_\xi\mathbf{A} + \Delta\tau\partial_\eta\mathbf{B}, \tag{4}$$

where the flux Jacobians $\mathbf{A}$ and $\mathbf{B}$ are given by

$$\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{u}}, \qquad \mathbf{B} = \frac{\partial \mathbf{g}}{\partial \mathbf{u}}. \tag{5}$$

The matrix $\mathbf{N}$ has five non-zero block diagonals for a first-order, upwind scheme approximation of the flux Jacobians. This matrix is costly to invert, so approximate factorization schemes have often been used to solve the system.

ADI

One common technique used to solve the system (2b) is the alternating direction implicit (ADI) scheme.[1] The correction matrix is factored as

$$\mathbf{N} = (\mathbf{I} + \Delta\tau\partial_\xi\mathbf{A}^n)(\mathbf{I} + \Delta\tau\partial_\eta\mathbf{B}^n) + O(\Delta t^2), \tag{6}$$

where the two matrix factors are block tridiagonal and are much easier to invert than the full N-matrix. The advantage of this scheme is that each factor can be solved one row or one column at a time, thus reducing the storage requirements. However, the factorization error places a restriction on the allowable time step.

ILU

Incomplete LU decomposition is another technique used to solve the system (2b). The ILU(l) class of schemes is particularly attractive.[2] The LU factors are formed such that the sparsity pattern of the factors is directly related to the sparsity pattern of the original matrix and the level of fill-in is denoted by l:

$$\mathbf{N} \approx \mathbf{LU}, \qquad \mathbf{LU} = \mathbf{N} + \mathbf{R}. \tag{7}$$

The matrix **R** has terms restricted to locations outside the sparsity pattern of the **N**-matrix. In other words, the product **LU** equals the original sparsity pattern terms of **N** exactly, plus terms that fall outside the original sparsity pattern. The most basic scheme is the no-fill-in, ILU(0) scheme, which has non-zero blocks restricted to the location of non-zero blocks of the **N**-matrix. This scheme has the advantage that there is no splitting error associated with the desired matrix terms. However, the error terms are dependent on the time step. One disadvantage, which can be turned into an advantage, is that the entire system must be solved at once, which increases the storage required.

Only the ILU(0) scheme will be presented in this paper. The ILU(1) scheme produces better convergence in terms of iterations, but the increased CPU time per iteration decreases the convergence in terms of CPU time.

SOLUTION ACCELERATION TECHNIQUES

Switched evolution/relaxation

Newton iteration schemes converge very quickly in a neighbourhood of the final solution but can diverge in regions away from the final solution. Note that if $\Delta t \to \infty$, equation (2b) becomes a Newton iteration scheme. The switched evolution/relaxation (SER) scheme is formed by letting Δt grow as the solution evolves in order to avoid the divergence problems away from the solution.[3] In order to retain the advantages of local time stepping, the CFL number is defined as

$$\mathrm{CFL} = \mathrm{CFL}_0 / r_{\mathrm{ave}}, \tag{8}$$

where CFL_0 is the initial CFL number and r_{ave} is the normalized density residual norm.

Frozen Jacobians

When the solution becomes close to the converged solution, the system becomes more linear and the Jacobian matrices **A** and **B** become nearly constant. This behaviour can be exploited by freezing the Jacobian terms for several iterations to reduce the computational cost.[4,5] Now the system becomes

$$\mathbf{N}(\mathbf{u}^m)\Delta\mathbf{u} = -\mathbf{L}(\mathbf{u}^n), \quad m \leqslant n. \tag{9}$$

In the current implementation, m is determined from r_{ave}, the normalized norm of the density residual. Initially, $m = n$ until $r_{\mathrm{ave}} < 10^{-2}$, then m is reset to n when r_{ave} is reduced by a specified order of magnitude.

The increased storage of the ILU(l) scheme now becomes an advantage. If the Jacobian terms are constant, then the LU factors can be retained and the new system inversion only requires two triangular

solves. However, the reduced storage for the ADI scheme is now a disadvantage: the tridiagonal matrix systems must be resolved even though the matrix terms are the same.

Multigrid

Multigrid is a solution acceleration scheme that maintains the solution accuracy available on a fine grid but takes advantage of the increased signal propagation speed and reduced computational cost of a coarse grid.[6–8] For the non-linear problem on a fine mesh f,

$$\mathbf{L}_f(\mathbf{u}_f) = \mathbf{0}, \tag{10}$$

an equivalent problem exists on a coarse mesh c,

$$\mathbf{L}_c(\mathbf{u}_c) + \mathbf{C}_c^f(\mathbf{L}_f(\mathbf{u}_f)) - \mathbf{L}_c(\mathbf{C}_c^f(\mathbf{u}_f)) = \mathbf{0}, \tag{11}$$

where $\mathbf{C}_c^f$ is a collection operator from f to c. The coarse grid equation is solved for $\Delta\mathbf{u}_c$ and the fine coarse grid correction is obtained using

$$\mathbf{u}_f^{n+1} = \mathbf{u}_f^n + \mathbf{D}_f^c(\Delta\mathbf{u}_c), \tag{12}$$

where $\mathbf{D}_f^c$ is a distribution operator from c to f. The coarse grid is obtained by dropping every other fine grid node point in each direction. The algorithm is applied in a V-cycle, where equation (11) is solved on the way down, with no subiterations, and equation (12) is used on the way up. The multigrid scheme will be denoted MG(N_{grid}), where N_{grid} indicates the number of multigrid grid levels.

GMRES

The non-linear GMRES algorithm can be applied to non-symmetric, non-linear systems.[9] It locates the best solution to the problem over a k-dimensional Krylov subspace. The main difference from the linear GMRES algorithm is that the subspace is built using finite difference sensitivities of the residual to changes in the solution. For the differential system

$$\mathbf{F}(\mathbf{u}) = \mathbf{0} \tag{13}$$

of K non-linear equations in K unknowns, the directional derivative of $\mathbf{F}$ at $\mathbf{u}$ in the $\mathbf{p}$-direction is defined as

$$\tilde{\mathbf{F}}(\mathbf{u}; \mathbf{p}) = \lim_{\varepsilon \to 0} \frac{\mathbf{F}(\mathbf{u} + \varepsilon\mathbf{p}) - \mathbf{F}(\mathbf{u})}{\varepsilon}. \tag{14}$$

Computationally, the directional differential is evaluated using a finite difference where ε is selected based on the magnitude of $\mathbf{F}(\mathbf{u})$.

Given a starting solution $\mathbf{u}^n$, a new solution is obtained by locating the optimum solution in a k-dimensional subspace spanned by the search direction vectors $\mathbf{p}_j$:

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \sum_{j=1}^{n} a_j \mathbf{p}_j. \tag{15}$$

The k orthonormal search directions are obtained as follows. The first search direction is set opposite to the residual vector:

$$\tilde{\mathbf{p}}_1 = -\mathbf{F}(\mathbf{u}^n), \qquad \mathbf{p}_1 = \tilde{\mathbf{p}}_1 / \|\tilde{\mathbf{p}}_1\|. \tag{16}$$

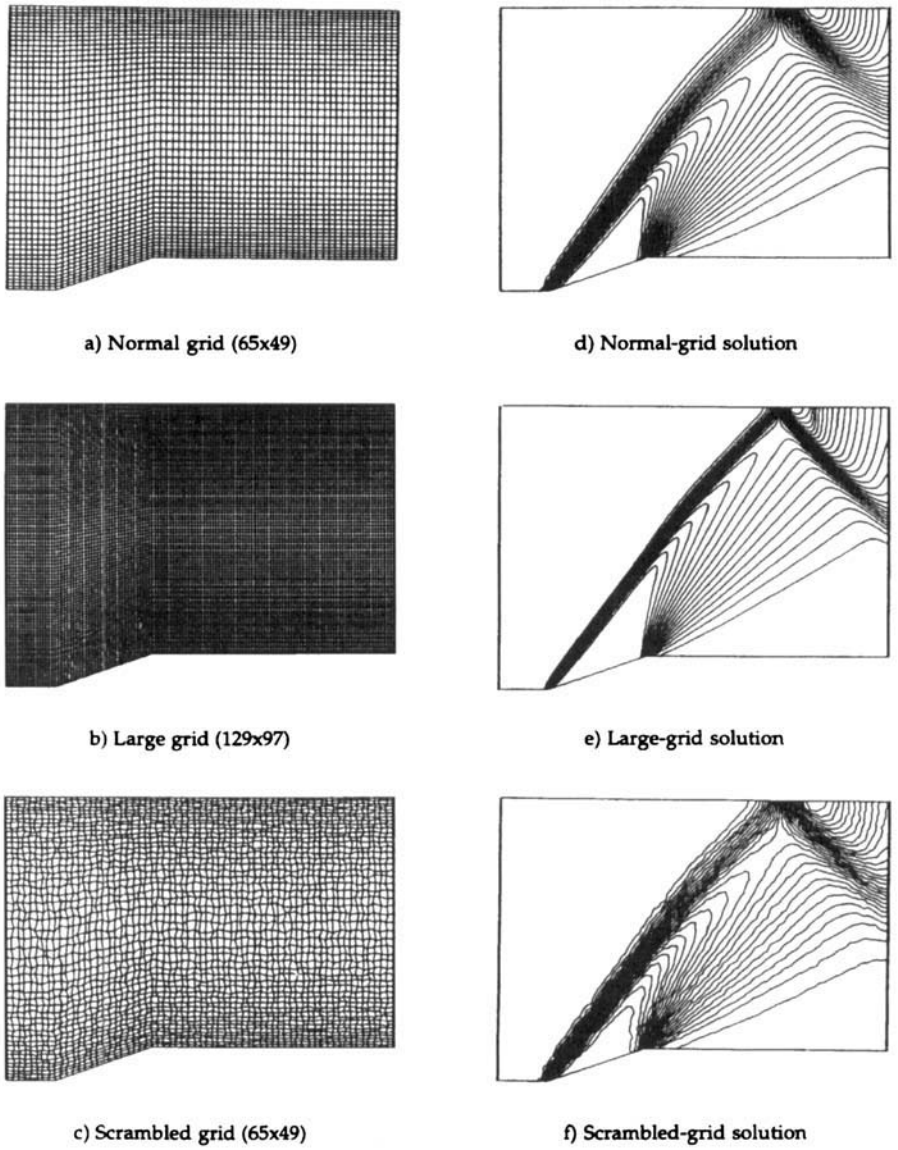


Figure 1. Grids and pressure distributions for $M = 2{\cdot}00$, 20° ramp duct using plain ADI scheme (left, inflow; right, outflow; top, symmetry; bottom, solid wall)

The remaining search directions are obtained by performing the following for $j = 1, 2, \ldots, k-1$:

$$\tilde{\mathbf{p}}_{j+1} = \bar{\mathbf{F}}(\mathbf{u}^n; \mathbf{p}_j) - \sum_{i=1}^{j} h_{ij}\mathbf{p}_i, \tag{17a}$$

$$\mathbf{p}_{j+1} = \tilde{\mathbf{p}}_{j+1}/\|\tilde{\mathbf{p}}_{j+1}\|, \tag{17b}$$

where

$$h_{ij} = \bar{\mathbf{F}}(\mathbf{u}^n; \mathbf{p}_j) \cdot \mathbf{p}_i, \tag{18}$$

so that each new search direction is orthogonal to the previous search directions.

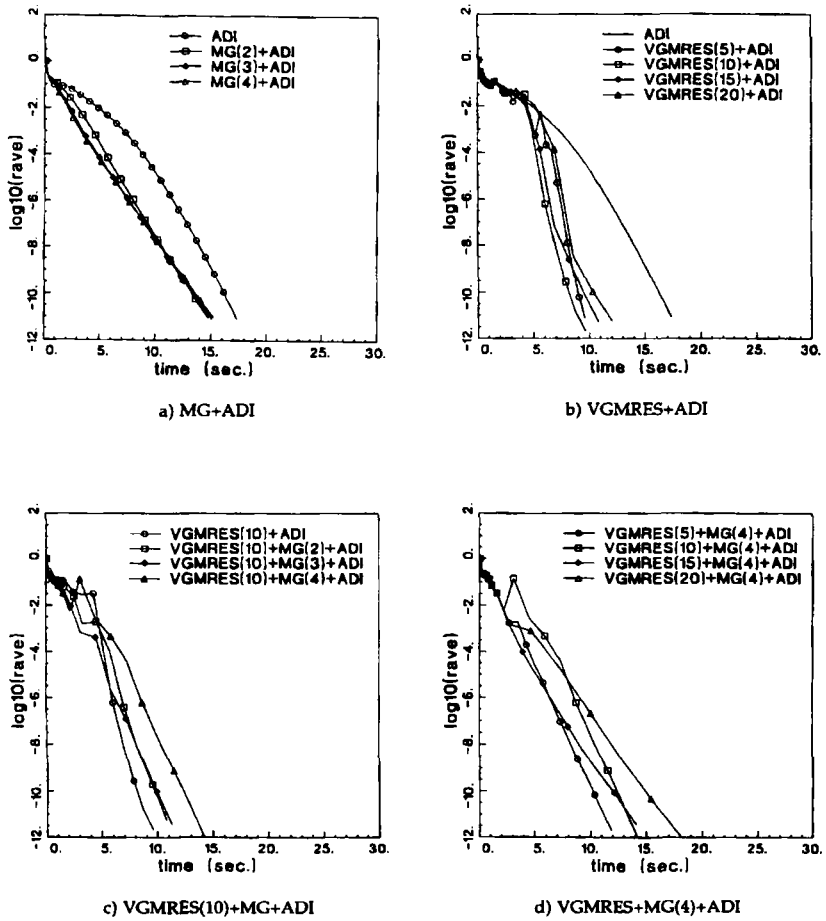


Figure 2. Convergence histories for ADI-based schemes on normal grid (CFL = 10)

The optimum weighting coefficients a_j are chosen to minimize the new residual norm

$$J = \|\mathbf{F}(\mathbf{u}^{n+1})\|^2 \approx \left\| \mathbf{F}(\mathbf{u}^n) + \sum_{j=1}^{k} a_j \bar{\mathbf{F}}(\mathbf{u}^n; \mathbf{p}_j) \right\|^2 . \tag{19}$$

Solving this least squares problem leads to a $k \times k$ system that is easy to solve.

Using GMRES with a large number of search directions will not improve the initial convergence rate for highly non-linear problems. This is because the optimum new solution is derived using a Taylor series that has been truncated to first order, so the new solution is only optimum in a linear sense. Therefore, until the correction matrix has become more linear, a small number of search directions will reduce the number of computations while not effecting the convergence. Computationally, this is performed using a variable-k GMRES scheme, denoted VGMRES(k), where k denotes the final number of search directions. The local k varies according to

$$k_{\text{local}} = \begin{cases} k_{\text{init}} + (k - k_{\text{init}}) \dfrac{n-1}{n_k - 1} & \text{for } n \leqslant n_k, \\ k & \text{for } n \geqslant n_k, \end{cases} \tag{20}$$

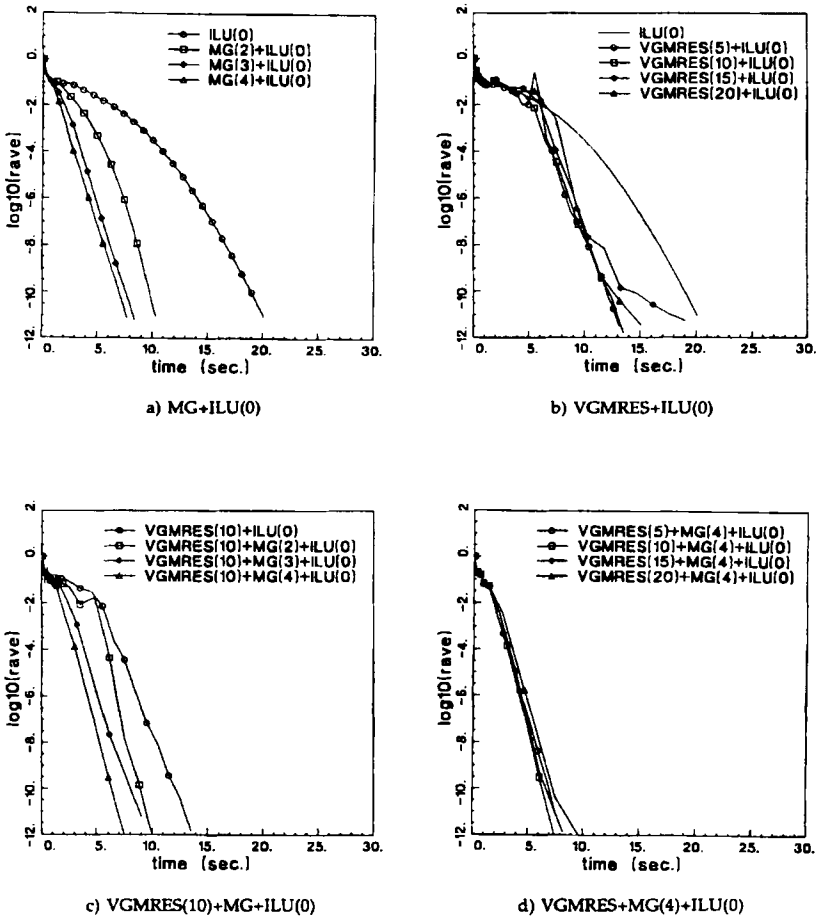


Figure 3. Convergence histories for ILU(0)-based schemes on normal grid (CFL = 10)

where n_k is the VGMRES iteration number where $k_{\text{local}} = k$ and remains constant for larger n. For this study, $k_{\text{init}} = 2$ and $n_k = 5$. In addition, VGMRES(k) is not begun until a 1·1-order-of-magnitude reduction of the residual is reached using the preconditioner schemes alone. This further reduces the non-linearity of the system before the VGMRES scheme is applied.

PRECONDITIONING

The convergence rate of the GMRES algorithm depends on the eigenvalue distribution of the system matrix. The more the eigenvalues are clustered together near unity, the faster GMRES will converge. Preconditioning changes the system to provide a more favourable eigenvalue distribution.[10] Because the identity operator has the best eigenvalue distribution (they are all equal to one), most preconditioning is done using an approximate inverse of the residual operator.

Consider the set of flow equations of the form

$$\mathbf{L}(\mathbf{u}) = \mathbf{0}. \tag{21}$$

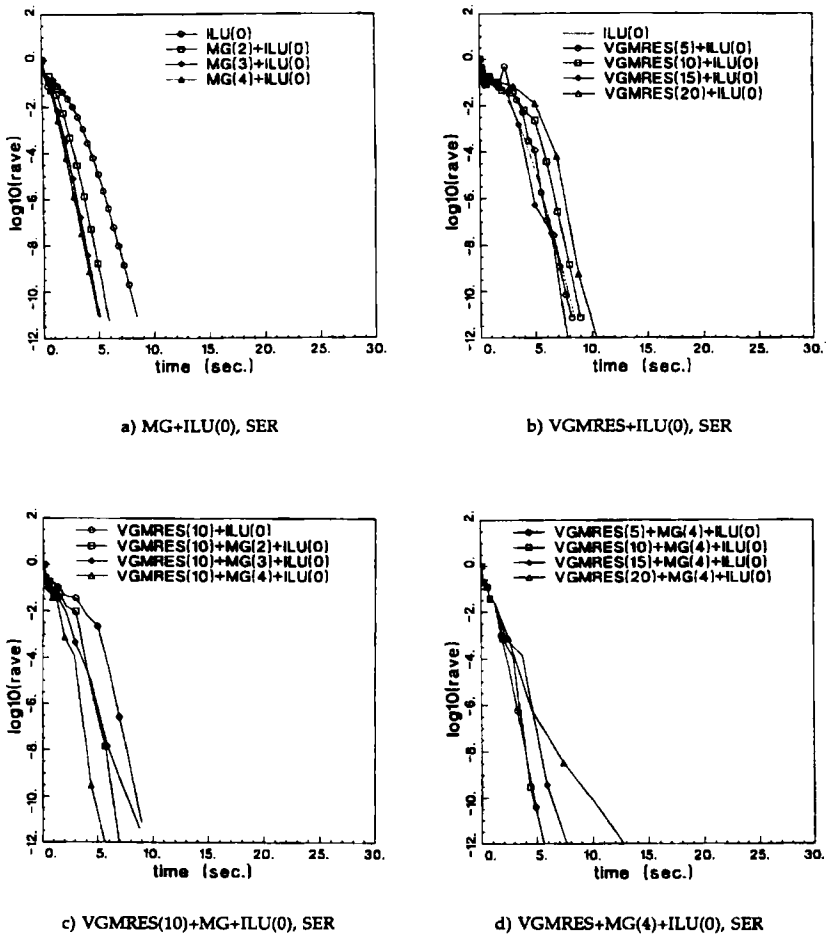


Figure 4. Convergence histories for ILU(0)-based SER schemes on normal grid (CFL = 10)

The GMRES algorithm can be applied directly to this system by setting $\mathbf{F}(\mathbf{u}) = \mathbf{L}(\mathbf{u})$. However, the eigenvalues of the system $\mathbf{L}$ will be scattered and the convergence will be slow. Computer codes that use an iterative approach already have a scheme, e.g. ADI and multigrid, that is used to advance the solution. Each scheme includes some form of approximate inverse to the operator in (21). In other words, the existing solution method is of the form

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta\mathbf{u}, \qquad \Delta\mathbf{u} = \mathbf{N}_{\text{pre}}^{-1}\mathbf{L}(\mathbf{u}^n). \tag{22}$$

The system matrix is preconditioned by the matrix $\mathbf{N}^{-1}$. The GMRES algorithm can be applied to the preconditioned system by setting

$$\mathbf{F}(\mathbf{u} = \mathbf{N}_{\text{pre}}^{-1}\mathbf{L}(\mathbf{u}^n). \tag{23}$$

In other words, preconditioned GMRES operates on the correction provided by the preconditioner.

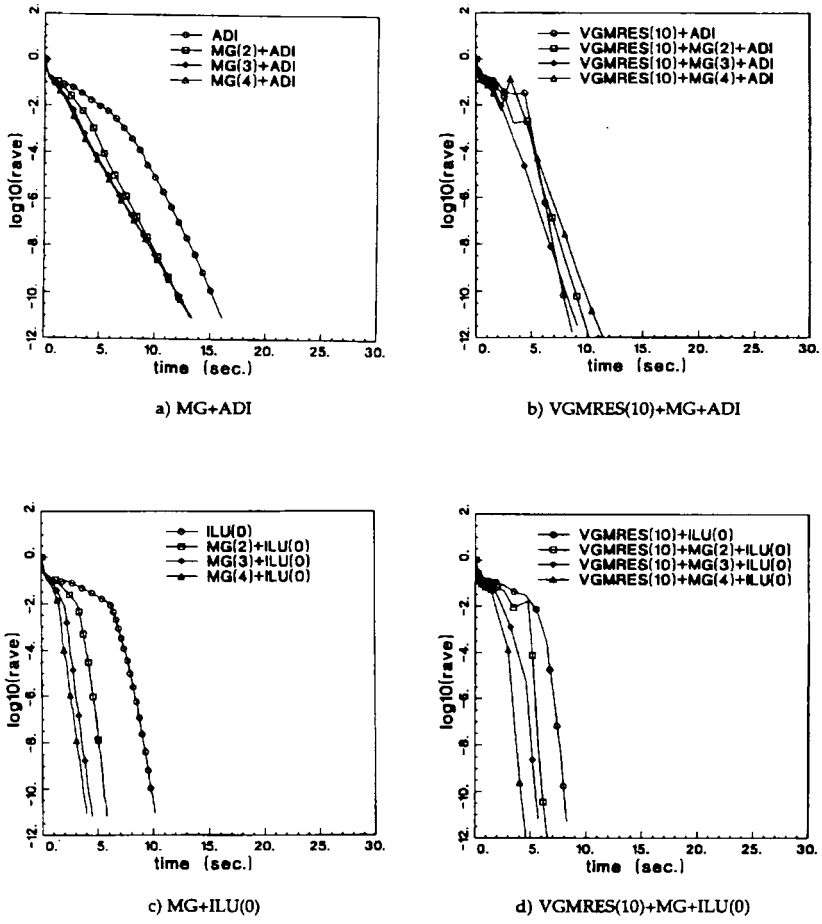
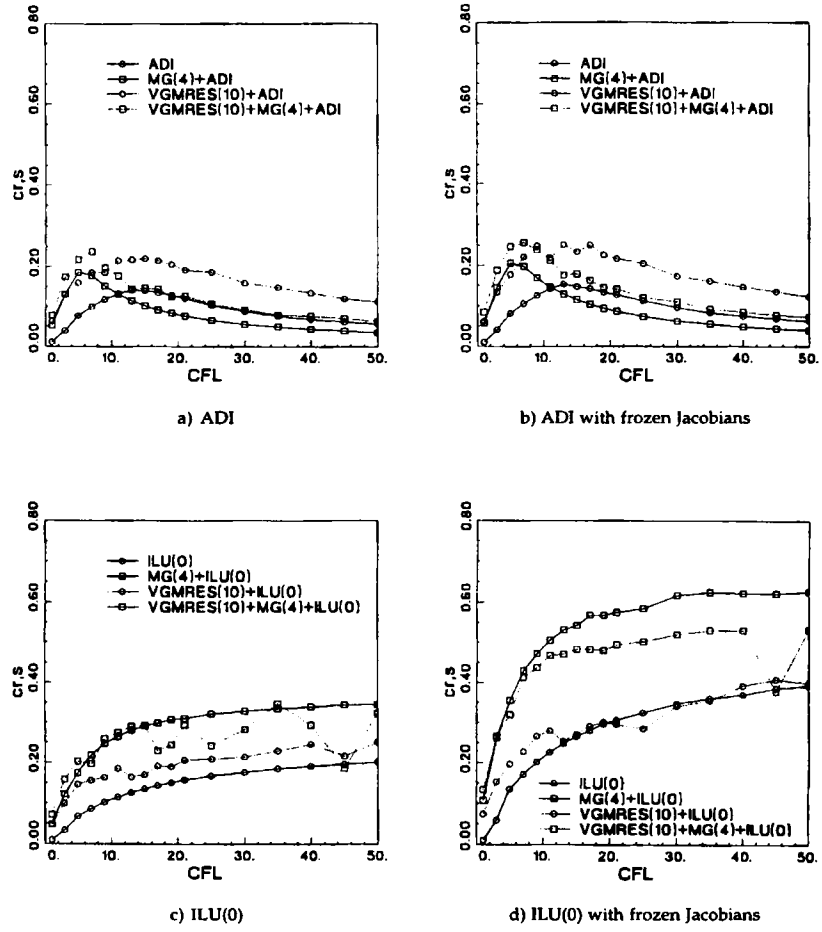


Figure 5. Convergence histories with frozen Jacobian on normal grid (CFL = 10)

RESULTS AND DISCUSSION

A parametric study was performed to evaluate the effects of grid size and grid quality on the convergence of the various schemes. The problem is an $M = 2{\cdot}00$, inviscid flow through a 20° ramp duct. The three grids that were used are shown in Figure 1, along with a solution obtained using the plain ADI scheme. CFL = 10 is used unless stated otherwise and local time stepping is used throughout. Calculations were run on a Cray C-90 for 400 work units or until the average residual was reduced by 11 orders of magnitude. One work unit represents the evaluation of the residual on the fine grid. When MG or VGMRES is used, each iteration requires more than one work unit. After the Jacobians are frozen, they are updated after a two-order-of-magnitude reduction of the residual for the normal and scrambled grids and after a one-order-of-magnitude reduction of the residual for the large grid. Note that the convergence history plots have markers at every second iteration with VGMRES and at every 10th iteration without VGMRES.



a) ADI

b) ADI with frozen Jacobians

c) ILU(0)

d) ILU(0) with frozen Jacobians

Figure 6. Convergence rate behaviour on normal grid

A useful measure of the convergence rate is given by

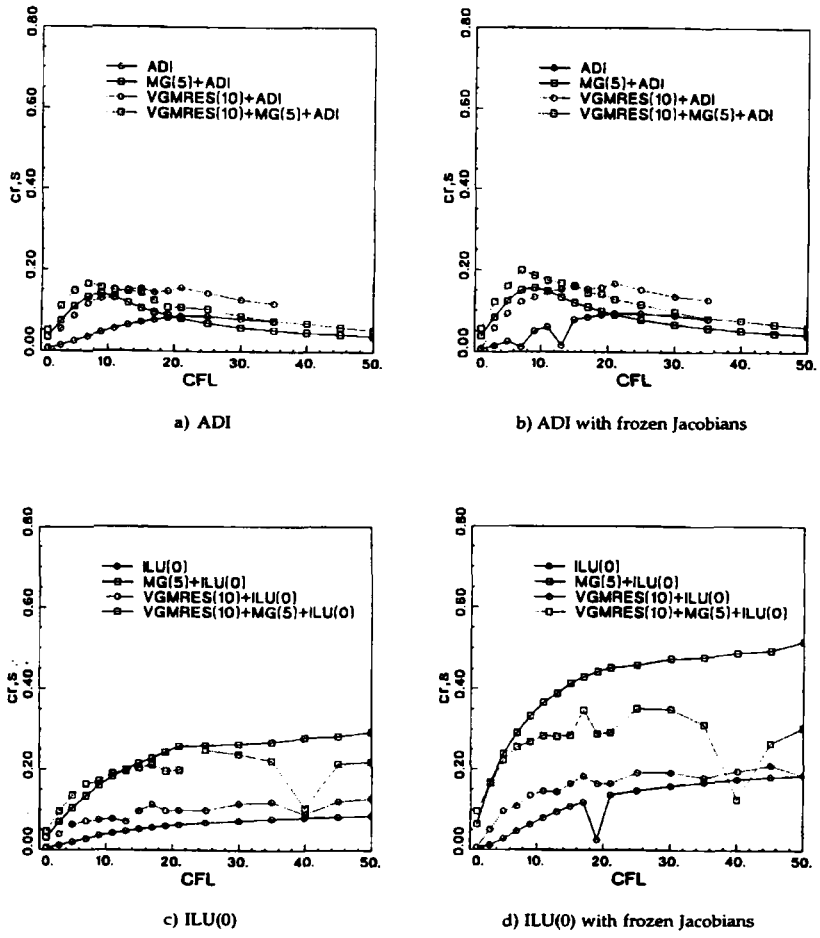
$$cr = 1 - \left(\frac{r_{\text{ave}_2}}{r_{\text{ave}_1}}\right)^p, \quad p = \frac{1}{work_2 - work_1}, \tag{24}$$

which represents the average fractional amount of error removed for each work unit. The average convergence rate data are based on the entire flow calculation. The values are scaled using

$$cr_s = cr\frac{C_{\text{ref}}}{C}, \tag{25}$$

where C is the CPU time per work node. This scaling takes into account the varying computational costs associated with the work units for various schemes. The reference C is from the ILU(0) solution on the normal grid.

Convergence histories for the normal grid with the ADI-based schemes are shown in Figure 2. Figure 2(a) shows the results for MG(N_{grid}) + ADI. Using more than two multigrid levels at this CFL number does not improve the convergence. This is because the time step on the coarse grids is larger than the time step on the fine grid. Therefore the factorization error on the coarse grids is larger than the



a) ADI

b) ADI with frozen Jacobians

c) ILU(0)

d) ILU(0) with frozen Jacobians

Figure 7. Convergence rate behaviour on large grid

factorization error on the fine grid and this prevents the corrections from the coarsest grid levels from helping the solution. Figure 2(b) shows the results for VGMRES(k) + ADI. Adding VGMRES improves the final convergence rate significantly and $k = 10$ converges the fastest. Figure 2(c) shows the results for VGMRES(10) + MG(N_{grid}) + ADI. Adding MG does not improve the convergence for this case. Figure 2(d) shows the results for VGMRES(k) + MG(4) + ADI. Increasing k decreases the convergence rate for this case.

Convergence histories for the normal grid with the ILU(0)-based schemes are shown in Figure 3. Figure 3(a) shows that plain ILU(0) converges slower than plain ADI. However, increasing the number of multigrid levels improves the convergence, so that MG(4) + ILU(0) converges about twice as fast as MG(4) + ADI. Figure 3(b) shows that $k = 5$ produces the best convergence for the VGMRES(k) + ILU(0) schemes. Figure 3(c) shows that adding MG to VGMRES(10) + ILU(0) improves the convergence for this case. Figure 3(d) shows that $k = 10$ produces the best VGMRES(k) + MG(4) + ILU(0) case.

Convergence histories for the normal grid with the ILU(0)-based SER scheme are shown in Figure 4. Figure 4(a) shows that plain ILU(0), SER converges more than twice as fast as plain ILU(0). SER also improves the other schemes, except for VGMRES(20) + MG(4) + ILU(0).

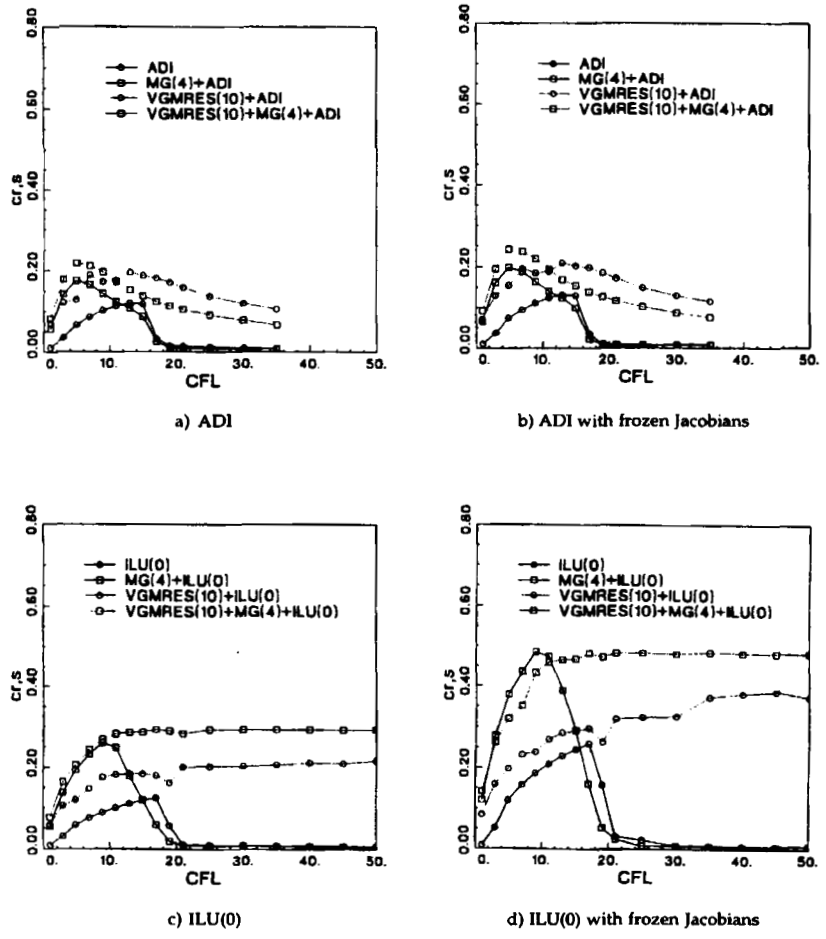


Figure 8. Convergence rate behaviour on scrambled grid

Figure 5 shows the convergence histories for selected ADI and ILU(0) schemes using frozen Jacobians. Freezing the Jacobians improves the ADI-based schemes slightly but improves the ILU(0) schemes significantly.

A summary of the average convergence rates for the normal grid is shown in Figure 6. Figure 6(a) compares the ADI-based schemes. There is an optimum CFL number due to the splitting error that is a function of the time step. Note that the schemes that use MG have a lower optimum CFL number. This is because the time step on the coarse grids is larger than the time step on the fine grid. Figure 6(b) compares the ADI-based schemes using frozen Jacobians. Freezing the Jacobians has a slight benefit in the scaled convergence rate. The oscillation in the convergence rate data is attributed to the non-linearity of the system. Figure 6(c) compares the ILU(0)-based schemes. The convergence rate increases over the tested range and does not show an optimum CFL number like the ADI-based schemes. However, the convergence rate does appear to be approaching an asymptote. Figure 6(d) compares the ILU(0)-based, frozen Jacobian schemes. Freezing the Jacobians has a dramatic effect on the convergence rate because it also freezes the LU factors.

A summary of the average convergence rates for the large grid is shown in Figure 7. The ADI-based schemes still have an optimum CFL number, but it is increased owing to the smaller cell sizes. The convergence rate of the MG(5) + ILU(0) scheme is almost as high as that of the MG(4) + ILU(0) scheme on the normal grid. This demonstrates the major advantage of multigrid: the convergence rate can be made relatively independent of system size.

A summary of the average convergence rates for the scrambled grid is shown in Figure 8. Including VGMRES helps all the schemes considerably. Without VGMRES, both the ADI- and ILU(0)-based schemes do not converge for CFL > 20. Adding VGMRES produces an acceptable scheme for these high CFL numbers.

CONCLUSIONS

The ILU(0) scheme can take full advantage of the frozen Jacobian and multigrid schemes to produce a very efficient solution technique. The time to reach a converged solution can be reduced by a factor of five compared with the plain ILU(0) scheme. The major advantage of the ILU(0) scheme over the ADI scheme is that the factorization error is not directly proportional to the time step. This allows the ILU(0) scheme to take advantage of larger time steps, which makes the multigrid scheme more effective. However, the ILU(0) factorization error does limit the effectiveness of the quasi-Newton scheme. Also, the ILU(0) scheme can take better advantage of the frozen Jacobians, because the LU factors do not have to be recomputed when the Jacobians are frozen, whereas the ADI scheme still requires two tridiagonal matrix solves. Adding VGMRES does not consistently improve convergence in the current implementation. For the two nice grids, when VGMRES is added to the multigrid, the convergence is reduced at high CFL numbers, but for the scrambled grid, VGMRES does produce an acceptable scheme for high CFL numbers when the other schemes break down. Adding VGMRES does, however, produce acceptable convergence rates that are relatively independent of both grid size and grid quality.

REFERENCES

1. R. M. Beam and R. F. Warming, 'An implicit finite-difference algorithm for hyperbolic systems in conservation-law form', *J. Comput. Phys.*, **22**, 87–110 (1976).
2. J. A. Meijerink and H. A. van der Vorst, 'Guidelines for the usage of incomplete decompositions in solving sets of linear equations as they occur in practical problems', *J. Comput. Phys.*, **44**, 134–155 (1981).
3. W. A. Mulder, 'Multigrid relaxation for the Euler equations', *J. Comput. Phys.*, **60**, 235–252 (1985).
4. H. E. Bailey and R. M. Beam, 'Newton's method applied to finite- difference approximations for the steady-state compressible Navier–Stokes equations', *J. Comput. Phys.*, **93**, 108–127 (1991).
5. D. B. Kim and P. D. Orkwis, 'Jacobian update strategies for quadratic and near-quadratic convergence of Newton and Newton-like implicit schemes', *AIAA Paper 93-0878*, 1993.
6. A. Brandt, 'Multi-level adaptive solutions to boundary-value problems', *Math. Comput.*, **31**, 333–390 (1977).
7. A. Jameson, 'Successes and challenges in computational aerodynamics', *Proc. AIAA 8th Computational Fluid Dynamics Conf., CP-87-1184*, pp. 1–35.
8. A. A. Arnone and S. S. Stecco, 'Inviscid cascade flow calculations using a multigrid method', *ASME Paper 89-GT-22*, AIAA, New York, 1987, 1989.
9. L. B. Wigton, N. J. Yu and D. P. Young, 'GMRES acceleration of computational fluid dynamics codes', *AIAA Paper 85-1494*, 1985.
10. G. Brussino and V. Sonnad, 'A comparison of direct and preconditioned iterative techniques for sparse, unsymmetric systems of linear equations', *Int. j. numer. methods eng.*, **28**, 801–815 (1989).