A FAST SOLVER FOR THE EULER EQUATIONS ON UNSTRUCTURED GRIDS USING A NEWTON-GMRES METHOD

Max Blanco* David W. Zingg†

University of Toronto Institute for Aerospace Studies
4925 Dufferin Street, Downsview, Ontario
Canada M3H 5T6

Abstract

This paper discusses the use of two variations on Newton's method, quasi-Newton and full-Newton, for the solution of the Euler equations on unstructured triangular grids. The ILU(n)-preconditioned GMRES algorithm is employed in the solution of the Jacobian matrix problem which arises at each iteration. In the quasi-Newton method, a first-order approximation to the Jacobian matrix is used with the standard GMRES implementation. Both standard and matrix-free implementations of GMRES are used in the full-Newton scheme, and the latter is shown to be much faster and more efficient. A hybrid scheme is presented which makes use of the strengths of both full- and quasi-Newton implementations, resulting in very fast convergence to steady state. Finally, optimal preconditioning and reordering strategies are presented.

Introduction

Compressible flow solvers on unstructured grids have proven to be very efficient for steady inviscid flows. The most promising approaches generally use either the multigrid technique or some approximate form of Newton's method in order to converge to a steady-state solution. The appeal of the latter scheme is based on the potential for quadratic or superlinear convergence. At each iteration of Newton's method, a non-symmetric system of equations arising from the linearisation of the residual function must be solved. Iterative methods are generally used for the solution of this system, as direct methods have been shown by Venkatakrishnan [20] to be impractical for large problems. Krylov subspace methods such as GMRES [19] are often employed due to the non-symmetric nature of the system. Fortunately, the approximate solution obtained using iterative methods can be sufficient to maintain superlinear convergence [8]. In addition, because Krylov methods tend to fare poorly with ill-conditioned systems, a suitable preconditioner must be employed.

The linearisation of the residual function must be virtually exact in order to obtain the desired superlinear convergence. However, explicit formation of the Jacobian matrix resulting from the exact linearization can require excessive storage. This requirement can be reduced by forming the matrix-vector product required by GMRES indirectly; the "matrix-free" algorithm has no adverse effect on the convergence of the resulting full-Newton method. If the ILU preconditioner is employed, a lower-order approximation to the Jacobian matrix can be used in its calculation in order to conserve memory resources. Full-Newton algorithms using either full-matrix or matrix-free implementations of Krylov methods are described in references [2, 4, 10, 11, 14, 16]. The storage requirement can also be reduced by using a quasi-Newton method in which the exact Jacobian matrix is replaced by a lower-order approximation [1, 3, 5, 7, 12, 17, 21]. Although the number of Newton iterations to convergence may increase, the cost per GMRES iteration is reduced due to the improved conditioning of the matrix and the reduced operation count. Venkatakrishnan and Mavriplis [21] found this quasi-Newton strategy to be competitive with explicit multigrid solvers, albeit with increased storage requirements.

Other factors which can influence the efficiency of a Newton-Krylov algorithm include the ordering of the un-

*Graduate Student.
†Associate Professor, Member AIAA.

knowns [9] and the type of preconditioner used. Possible reordering algorithms include reverse Cuthill-McKee (RCM), nested dissection (ND), and quotient minimum degree (QMD). The choice of a preconditioner is dependent on the tradeoff between speed and memory. Many authors have used the incomplete lower-upper factorization with no additional fill, ILU(0). Other possibilities include ILU(n), in which fill entries are permitted based on a "level-of-fill" parameter n, and ILUT(p, t), which employs a threshold drop tolerance t, to introduce a maximum of p fill entries in each factor [18]. Full-Newton algorithms may encounter difficulties in the start-up phase, during which the iterates may diverge or may converge slowly. Various strategies have been proposed to deal with this problem. These generally make use of a less expensive solution procedure during the early iterations [2, 14, 16].

The goal of the present paper is to examine the tradeoffs inherent in the various Newton-GMRES implementations and to develop a fast, robust algorithm applicable to unstructured grids. In particular, we compare two full-Newton algorithms, using matrix-free and standard implementations of GMRES, to a quasi-Newton strategy. Optimal strategies for the choice of ILU(n) fill level and reordering technique are also investigated.

The Euler Equations

The Euler equations, which model compressible inviscid fluid flows, can be derived by writing expressions for the conservation of mass, momentum and energy. In integral form, for an arbitrary two-dimensional control "volume" Ω bounded by "surface" $\partial\Omega$, they are given by

$$\oint_{\partial\Omega} (\mathbf{F} \mathrm{d}y - \mathbf{G} \mathrm{d}x) = \mathbf{0} \tag{1}$$

where

$$\mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u \left(e + \frac{p}{\rho}\right) \end{bmatrix} \quad \mathbf{G} = \begin{bmatrix} \rho v \\ \rho v u \\ \rho v^2 + p \\ \rho v \left(e + \frac{p}{\rho}\right) \end{bmatrix} \tag{2}$$

$$e = \frac{p}{(\gamma - 1)\rho} + \frac{u^2 + v^2}{2} \tag{3}$$

Equations 2 are written using the conservative flow variable formulation, $\mathbf{Q} = [\rho, \rho u, \rho v, \rho e]^T$, in order to admit shock capturing. Equation 3, the equation of state for a perfect gas, permits closure of the system by establishing a relation between the thermodynamic variables.

Numerical Method

Discretisation of equation 1 over an unstructured triangular grid using a finite-volume flux integration results in a system of nonlinear equations. This system is written as

$$\mathbf{R}(\mathbf{Q}) = \mathbf{0} \tag{4}$$

where $\mathbf{R}$ is referred to as the residual. The centred finite-volume spatial discretisation, including the nonlinear scalar artificial dissipation, is similar to that used by Mavriplis and Jameson [13]. Details of this particular implementation, including the boundary conditions, can be found in references [6] and [15].

Newton's method for the iterative solution of a system of equations in the form of equation 4 can be written as follows:

$$\left(\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}\right)^k \Delta \mathbf{Q}^k = -\mathbf{R}\left(\mathbf{Q}^k\right) \tag{5}$$

where the superscript k is the iteration index. The term $\partial\mathbf{R}/\partial\mathbf{Q}$ is the Jacobian matrix. Solution of this matrix equation for the unknown, $\Delta\mathbf{Q}^k$ and update of the solution vector, $\mathbf{Q}^{k+1} = \mathbf{Q}^k + \Delta\mathbf{Q}^k$, is continued until the process is deemed to have converged. All cases presented below are initiated with uniform freestream flow conditions. The L_2-norm of the full residual vector is used to measure convergence:

$$L_2(\mathbf{R}) = \sqrt{\sum_{i=1}^{N} R_i^2} \tag{6}$$

where N is equal to four times the number of nodes.

The data structure is such that the Jacobian matrix is sparsely populated by four-by-four blocks of entries. Although the structure of the non-zero blocks is symmetric, the matrix itself is non-symmetric. This leads to the use of the iterative GMRES algorithm of Saad and Schultz [19] in the solution of the matrix problem of equation 5 . Because the matrix problem is solved with an iterative method, a distinction must be drawn between these "inner" iterations and the "outer" Newton iterations. The inner iterations are terminated when the residual norm reaches an exit tolerance σ which depends on the outer residual $\mathbf{R}$ at each outer iteration. The optimal choice of σ differs for subsonic and transonic cases, and is determined at each outer iteration as follows:

- subsonic cases:

$$\sigma^k = 10^{-1} L_2\left(\mathbf{R}^k\right) \tag{7}$$

- transonic cases:

$$\sigma^k = 10^{-2} L_2\left(\mathbf{R}^k\right) \tag{8}$$

This procedure is discussed in detail in reference [6].

The scalar ILU(n) preconditioner is used to accelerate the convergence of the inner iterations. This is necessary due to the ill-conditioning of the Jacobian matrix. The preconditioner is frozen once the outer residual falls below 10^{-5}. A number of reordering algorithms can be used in order to reduce the amount of fill caused by the preconditioner when using $n > 0$. The ordering can have a significant impact on the quality of the preconditioner and therefore the number of inner iterations required to reach a given tolerance.

Test Cases

Results are presented for subsonic ($M_\infty = 0.63$, $\alpha = 2.0^o$) and transonic ($M_\infty = 0.85$, $\alpha = 1.0^o$) flows over the NACA 0012 airfoil. Calculations were performed on two grids: one grid consisting of 14,193 nodes, 350 of which are located on the airfoil surface, and 120 of which are located on the far-field boundary; and another grid consisting of 1,214 nodes, 60 of which are on the airfoil surface, and 36 of which are located on the far-field boundary. The latter is employed only for results in which comparison is made with the full-Newton scheme using standard GMRES. Both grids were generated using an algorithm due to Predovic [15] which employs a combination of advancing front and Delaunay triangulation techniques. A magnified view of the fine grid is shown in figure 1.

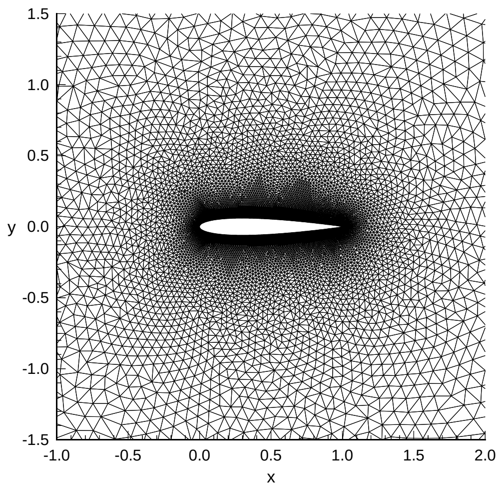

Figure 1: 14,193-node grid over NACA 0012 airfoil.

Convergence histories for all solvers presented here were obtained using the same workstation and compiler optimisations. Comparison is made on the basis of CPU time consumption since iteration or work unit counts do not fully reflect the relative performance. The restarted GMRES algorithm was not employed since the maximum number of inner iterations for all test cases was set at either 20 or 25.

Quasi-Newton Solver

The scheme termed "quasi-Newton" results from an incomplete linearisation of the residual. The biharmonic dissipation stencil is truncated so as to leave Jacobian entries in positions corresponding to nearest-neighbour nodes only, as described in reference [6]. The storage required for the Jacobian matrix is therefore significantly reduced vis-a-vis a complete linearisation of the residual. The ILU preconditioner is calculated from this first-order approximation to the Jacobian matrix.

The departure from Newton's method due to the incomplete linearisation of the residual results in an increase in the number of outer iterations required to achieve convergence. Figure 2 summarises the convergence history of the quasi-Newton solver for both test cases on the fine grid. Table 1 shows the CPU time spent in calculating the ILU factors, t_{prec}, in the GMRES algorithm, t_{GMRES}, and overall, t_{total}, as well as the average number of inner iterations at each outer iteration, N_{in}, and the total number of outer iterations, N_{out}.

Case	t_{prec}	t_{GMRES}	t_{total}	N_{in}	N_{out}
Subsonic	159.7	873.7	23:46	8.2	75
Transonic	303.2	1224.1	34:03	8.9	95

Table 1: Convergence history data for the quasi-Newton solver on the fine grid.

The rate of convergence of the quasi-Newton scheme is roughly the same for both subsonic and transonic cases, except that during the initial few iterations of transonic cases, instabilities may develop. Therefore, when solving transonic flows, the Jacobian is modified by the addition of entries to its diagonal, which results in the much more stable implicit Euler pseudo-time marching scheme. This diagonal conditioning is decreased as the solution progresses, and can be turned off after approximately 20 outer iterations.

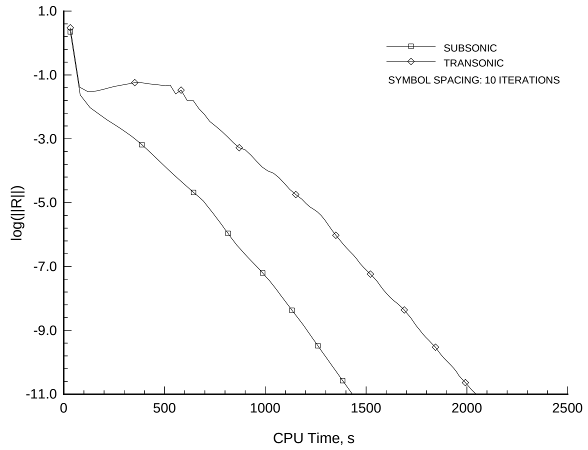

Figure 2: Convergence of the quasi-Newton solver on the fine grid.

Full-Newton Solvers

Standard GMRES Implementation

The kernel of the GMRES procedure is the matrix-vector product. One way of forming this product involves the explicit multiplication of the sparse matrix and vector, both of which are stored in memory. The stencil of the biharmonic dissipation operator employed encompasses both nearest- and next-to-nearest-neighbours for every interior node. Due to its scalar form, it only contributes non-zero entries to the block diagonals only. Since the flux integration requires only a nearest-neighbour stencil, the linearisation of the residual function causes only the diagonal entries on the blocks corresponding to next-to-nearest neighbours are non-zero. For the same reasons, only the diagonal entries in the diagonal blocks to be non-zero. The flexibility of the scalar matrix storage scheme ensures that the resulting benefits are exploited. The storage costs of the scalar system are reduced with respect to those of a block-based system since "empty" off-diagonal non-zero entries are not allocated. The ILU preconditioner calculations are based on this "second-order" Jacobian matrix.

The full-Newton solver converges within ten iterations once the solution vector lies within the radius of convergence of Newton's method. Otherwise, it can fail to converge or become unstable. This is a problem only with transonic flows, while subsonic flows allow the use of the full-Newton algorithm from the first iteration. When solving transonic flows, the Jacobian is modified to resemble the implicit Euler time-marching scheme as is done for the quasi-Newton method. This modification is also necessary for only the first 20 iterations.

Figure 3 presents convergence histories for the subsonic test case using the coarse grid. The relevant CPU data are listed in table 2. Although this scheme is faster than the quasi-Newton approach, it requires more memory.

Solver	t_{prec}	t_{GMRES}	t_{total}	N_{in}	N_{out}
Quasi-Newton	13.2	20.7	1:06.3	2.4	78
Standard	11.3	13.7	0:36.6	9.4	7
Matrix-Free	3.4	15.5	0:23.8	13.0	7

Table 2: Convergence history data for the subsonic case on the coarse grid.

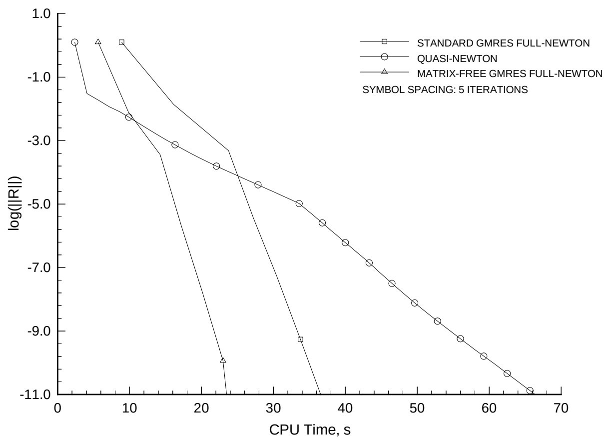

Figure 3: Convergence of the standard full-Newton and quasi-Newton schemes for the subsonic test case on the coarse grid.

Matrix-Free GMRES Implementation

The matrix-vector product required by GMRES can also be formed without explicitly performing the multiplication. This is done by making use of the Fréchet derivative of the residual function $\mathbf{R}$, which can be defined as

$$\frac{\partial \mathbf{R}}{\partial \mathbf{Q}} \cdot \mathbf{v} = \frac{\mathbf{R}(\mathbf{Q} + \epsilon \mathbf{v}) - \mathbf{R}(\mathbf{Q})}{\epsilon} \tag{9}$$

where $\mathbf{v}$ is the vector to be multiplied. The parameter ϵ is calculated at every inner iteration as a function of machine zero, ϵ_{mz}, and the magnitude of $\mathbf{v}$, as described in reference [14]:

$$\epsilon = \frac{\sqrt{\epsilon_{mz}}}{L_2(\mathbf{v})} \tag{10}$$

The "matrix-free" GMRES procedure which results from using the Fréchet derivative to form the matrix-vector product is effectively the same as using a full linearisation of the residual operator. It therefore yields a full-Newton scheme.

The chief advantage of this procedure is that the full matrix need not be stored. However, some form of matrix is still required in order to form the ILU preconditioner due to ill-conditioning of the matrix. The preconditioner is based on a first-order matrix using a truncated dissipation stencil for two reasons: first, as compared to the use of a full second-order matrix for preconditioning, a large savings in storage is gained; second, results reported for structured grids by Pueyo and Zingg [16] indicate that their full-Newton scheme is significantly more efficient when using the first-order matrix to form the preconditioner.

Figure 3 shows that the matrix-free GMRES implementation of the full-Newton scheme requires less CPU time than the standard GMRES implementation for the subsonic case on the coarse grid. This can be attributed to the reduced number of operations required in each matrix- or preconditioner-related calculation.

The matrix-free scheme is also notable for a large savings in memory over the standard scheme for this case. The storage requirements of the matrix-free solver are exactly the same as those of the quasi-Newton solver for a given level of preconditioner fill. However, as described below, the optimum preconditioner is larger for the matrix-free scheme than for the quasi-Newton scheme.

Figure 4 shows the matrix-free solver to be three times faster than the quasi-Newton scheme when using the fine grid. This is due to the reduction in the number of outer iterations. Table 3 presents the relevant CPU data for the matrix-free solution of the subsonic case on the fine grid.

Case	t_{prec}	t_{GMRES}	t_{total}	N_{in}	N_{out}
Subsonic	62.5	393.3	8:50	18.2	9

Table 3: Convergence history data for the matrix-free GMRES full-Newton solver on the fine grid.

The behaviour of the matrix-free implementation is similar to that of the standard GMRES solver in that, for transonic cases, the full-Newton algorithm cannot be used initially. The implicit Euler approach can also be used with the matrix-free implementation, but we prefer a hybrid approach, presented in the next section, because of the high cost per iteration of the matrix-free scheme. Table 4 summarises the cost per inner iteration for the subsonic test case using the coarse grid. Obviously, the quasi-Newton solver will provide the required initial solution for the full-Newton solver at a reduced cost. This fact is the basis for the development of the hybrid scheme.

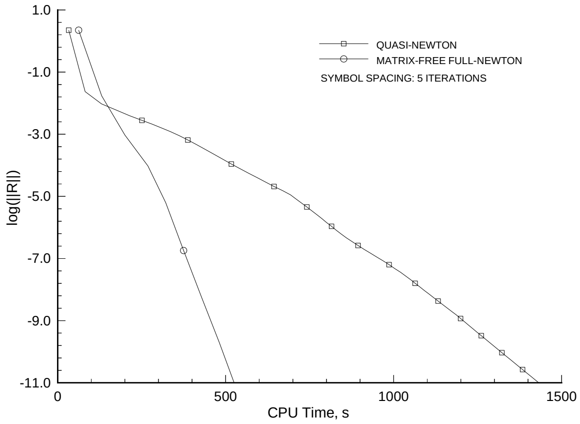

Figure 4: Convergence of the standard full-Newton and quasi-Newton schemes for the subsonic test case on the fine grid.

Solver	GMRES Type	Prec.	$\bar{t}_{GMRES}$
Quasi-Newton	Standard	ILU(2)	1.4
Full-Newton	Matrix-Free	ILU(4)	2.5

Table 4: Average CPU time, in seconds, per inner iteration on the coarse grid.

Hybrid Newton-GMRES Solver

A hybrid solver has been developed which makes use of the initial stability of the quasi-Newton solver in order to provide a startup solution for the full-Newton solver lying within its radius of convergence. The switch from quasi- to full-Newton is done when the outer residual falls below a user-selected level, ϕ_{MF}. Although it is desirable to switch to full-Newton mode as early as possible, instability can result from premature changeover. This problem is typically encountered with transonic cases due to shock formation and "motion" as the solution evolves toward steady state. Testing has shown that the solver will converge for most transonic cases when switching occurs at an outer residual equal to 10^{-3}. Figure 5 demonstrates the performance of the hybrid solver for the transonic case using various values of ϕ_{MF}. Subsonic cases do not require any initial quasi-Newton iterations, and full-Newton solution proceeds directly from the first iteration.

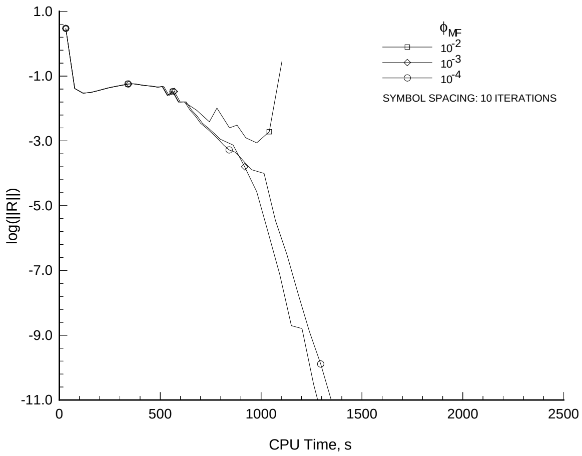

Figure 5: Effect of ϕ_{MF} on the convergence history of the hybrid solver using the fine grid.

Figure 6 compares the hybrid Newton-GMRES with the quasi-Newton solver results shown previously for the transonic case on the fine grid. These results show the hybrid solver to be almost twice as fast as the quasi-Newton scheme, while requiring a comparable amount of storage.

ILU(n) Preconditioner

To date, most researchers have employed the block ILU(0) preconditioner when using GMRES in implicit solvers.

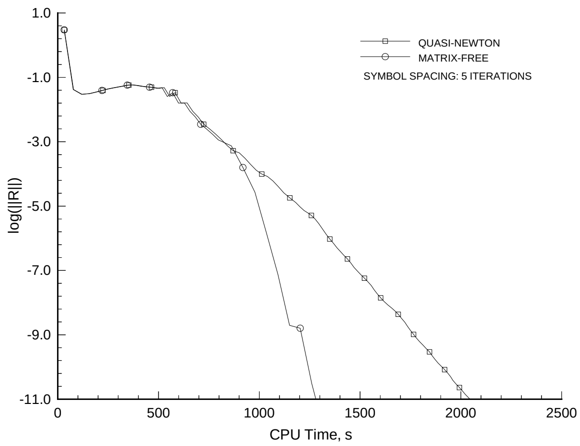

Figure 6: Convergence histories of the quasi-Newton and matrix-free solvers for the transonic case on the fine grid.

This is due to the use of a block-based matrix storage scheme and the ease with which the ILU(0) preconditioner can be implemented. This paper presents results for an implementation based on scalar storage of matrices and the ILU(n) preconditioner. The use of a scalar data structure provides much greater flexibility and allows new third-party numerical routines (often written for compressed sparse row format) to be incorporated with ease. For our purposes, the benefits of this flexibility outweigh the 10% penalty in storage needs for the Jacobian matrix. The ILU(0) preconditioner is not the optimal choice when using a scalar storage implementation. Data presented in tables 5 and 6 for the transonic and subsonic flows, respectively, show that the scalar ILU(0) preconditioner can be completely inadequate. Figures 7 and 8 present these results graphically.

Increasing the amount of fill leads to a higher cost per application of the preconditioner and increased storage requirements. These must be weighed against the reduction in the number of inner, and sometimes outer, iterations. In terms of CPU expense, ILU(2) is optimal for the quasi-Newton solver and ILU(4) is optimal for the matrix-free full-Newton solver. Table 6 shows that the memory requirements of the matrix-free solver can be reduced by decreasing the amount of fill in the preconditioner, with a corresponding increase in CPU time. Comparison of tables 1 and 6 shows that the matrix-free solver remains considerably faster than the quasi-Newton solver even with ILU(1) preconditioning.

Prec.	$NPNZ$	t_{prec}	t_{GMRES}	t_{total}	N_{in}	N_{out}
ILU(0)	1297932	Did not converge.				
ILU(1)	2064290	183.4	1556.2	37:35	12.9	98
ILU(2)	2871430	303.2	1224.1	34:03	8.9	95
ILU(3)	3694428	465.9	1120.1	35:19	7.0	97
ILU(4)	4520330	669.4	1050.3	37:27	5.8	96
ILU(5)	5356792	894.9	970.3	39:26	4.9	96
ILU(6)	6201994	1151.8	910.7	43:14	4.2	96

Table 5: Effect of the fill parameter n on the performance of the quasi-Newton solver for the transonic case.

Prec.	$NPNZ$	t_{prec}	t_{GMRES}	t_{total}	N_{in}	N_{out}
ILU(0)	1297932	Did not converge.				
ILU(1)	2064290	40.7	843.1	16:53	19.4	24
ILU(2)	2871430	49.4	624.4	12:48	19.0	16
ILU(3)	3694428	54.5	454.7	9:45	18.5	11
ILU(4)	4520330	62.5	393.3	8:50	18.2	9
ILU(5)	5356792	85.2	369.2	8:54	18.0	8
ILU(6)	6201994	111.8	367.6	9:30	16.9	8

Table 6: Effect of the fill parameter n on the performance of the matrix-free solver for the subsonic case.

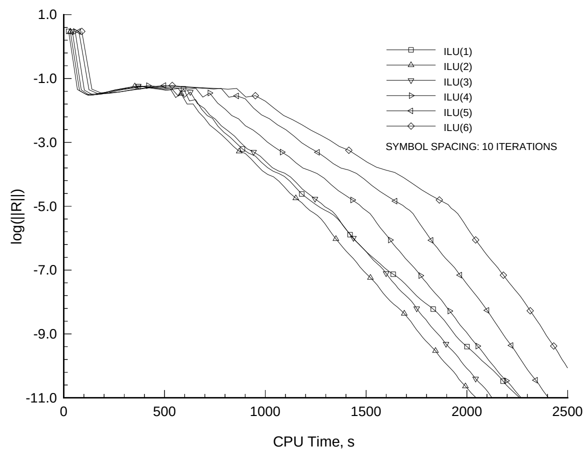

Figure 7: Effect of the fill parameter n on the performance of the quasi-Newton solver for the transonic case.

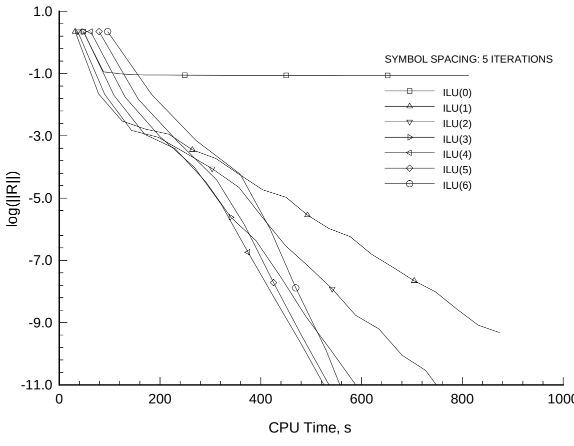

Figure 8: Effect of fill the parameter n on the performance of the matrix-free solver for the subsonic case.

Reordering

A number of reordering algorithms have been developed for use with direct solvers. The effects of three common reordering algorithms, reverse Cuthill-McKee (RCM), nested dissection (ND), and quotient minimum degree (QMD), are described in this section. The "natural" ordering of unknowns produced by the grid generation process is also tested.

As demonstrated in the previous section, the number of non-zeros in the preconditioner has a significant impact on the overall performance of an implicit solver using ILU-preconditioned GMRES. Because the ordering of unknowns in a matrix has a direct impact on the number of non-zeros generated by the ILU(n) preconditioner, the choice of ordering is of critical importance. Table 7 and figure 9 present results for the transonic flow obtained with the quasi-Newton solver and the ILU(2) preconditioner. Table 7 shows that the reorderings can affect

Reord.	$NPNZ$	t_{total}	N_{in}	N_{out}	$\bar{t}_{GMRES}$
None	2806526	0:33:21	8.8	96	1.40
RCM	2871430	0:34:03	8.9	95	1.45
QMD	3579453	1:12:18	20.1	97	1.63
ND	3771043	1:11:46	20.7	96	1.62

Table 7: Effect of reorderings on the performance of the quasi-Newton solver for the transonic case.

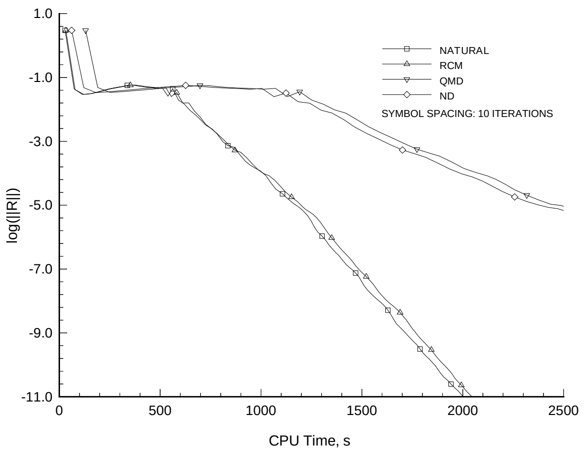

Figure 9: Effect of reorderings on the performance of the quasi-Newton solver.

the quality of the ILU preconditioner: because the inner iteration exit tolerance is fixed to $\sigma^k = 10^{-2} L_2 \left(\mathbf{R}^k\right)$, this is evidenced by the increase in the number of inner iterations required when using the QMD and ND reorderings. As a result, these two algorithms are considerably inferior to RCM for this application.

Study of the behaviour of all solvers presented here over a range of cases when using RCM and natural orderings has shown that although the latter ordering often results in fast convergence, it can also cause temporary stalling of the matrix-free full-Newton scheme, as demonstrated in figure 10. These data were obtained for the transonic case. Use of the RCM ordering prevents this undesirable behaviour.

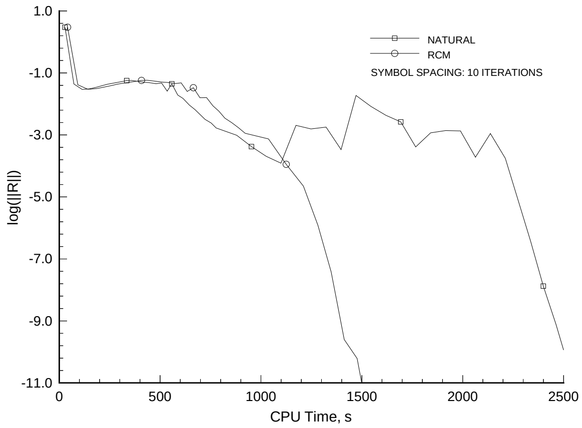

Figure 10: Stalling of the matrix-free solver when using the natural ordering.

Conclusions

We have shown that the Reverse Cuthill-McKee algorithm is the best reordering for our solver, and that scalar ILU(n) preconditioning can be an efficient strategy with fill levels ranging from two to four. Our most important conclusion is that the hybrid strategy, which combines a quasi-Newton method with a full-Newton method based on a matrix-free implementation of GMRES, is considerably faster than the quasi-Newton strategy alone. Considering the previous observation that the quasi-Newton method is competitive with fast multigrid solvers [21], this suggests that the present hybrid strategy is extremely promising. Future work will involve extension to turbulent flows and examination of techniques for reducing the storage requirements of Newton-Krylov methods.

References

1. Anderson, W. K., "A grid generation and flow solution method for the Euler equations on unstructured grids," *Journal of Computational Physics*, vol. 110, 1994.

2. Anderson, W. K., R. D. Rausch and D. L. Bonhaus, "Implicit/multigrid algorithms for incompressible turbulent flows on unstructured grids," Tech. Rep. AIAA–95–1740, AIAA, 1995.

3. Barth, T. J., "Numerical aspects of computing viscous high Reynolds number flows on unstructured meshes," Tech. Rep. AIAA–91–0721, AIAA, 1991. 29th Aerospace Sciences Meeting.

4. Barth, T. J. and S. W. Linton, "An unstructured mesh Newton solver for compressible fluid flow and its parallel implementation," Tech. Rep. AIAA–95–0221, AIAA, 1995. 33rd Aerospace Sciences Meeting and Exhibit.

5. Batina, J. T., "A fast implicit upwind solution algorithm for three–dimensional unstructured dynamic meshes," Tech. Rep. AIAA–92–0447, AIAA, 1992. 30th Aerospace Sciences Meeting and Exhibit.

6. Blanco, M., "An implicit solution method for the Euler equations on unstructured triangular grids," Master's thesis, Department of Aerospace Science and Engineering, University of Toronto, September 1994.

7. Cabello, J., K. Morgan, R. Lohner, H. Luo and J. Baum, "An implicit solver for laminar compressible flows on unstructured grids," Tech. Rep. AIAA–95–0344, AIAA, 1995. 33rd Aerospace Sciences Meeting and Exhibit.

8. Coleman, T. F., *Large Sparse Numerical Optimization*, vol. 165 of *Lecture Notes in Computer Science*. Springer–Verlag, 1984.

9. Dutto, L. C., "The effect of ordering on preconditioned GMRES algorithm, for solving the compressible Navier–Stokes equations," *International Journal for Numerical Methods in Engineering*, vol. 36, pp. 457–497, 1993.

10. Forsyth, P. A. and H. Jiang, "Iterative methods for full Newton solution of the Euler equations," in *Sixth International Symposium on Computational Fluid Dynamics*, vol. 1, pp. 318–323, 1995.

11. Johan, Z., T. J. R. Hughes and F. Shakib, "A globally convergent matrix-free algorithm for implicit time-marching schemes arising in finite element analysis in fluids," *Computer Methods in Applied Mechanics and Engineering*, vol. 87, pp. 281–304, 1991.

12. Luo, H., J. Baum, R. Lohner and J. Cabello, "Implicit schemes and boundary conditions for compressible flows on unstructured meshes," Tech. Rep. AIAA–94–0816, AIAA, 1994. 32nd Aerospace Sciences Meeting and Exhibit.

13. Mavriplis, D. J. and A. Jameson, "Multigrid solution of the two–dimensional Euler equations on unstructured triangular meshes," Tech. Rep. AIAA–87–0353, AIAA, 1987. 25th Aerospace Sciences Meeting.

14. Nielsen, E. J., W. K. Anderson, R. W. Walters and D. E. Keyes, "Application of Newton-Krylov methodology to a three-dimensional unstructured Euler code," Tech. Rep. AIAA–95–1733, AIAA, 1995.

15. Predovic, D. T., "Multigrid solution of the Euler equations using unstructured adaptive meshes," Master's thesis, Department of Aerospace Science and Engineering, University of Toronto, April 1994.

16. Pueyo, A. and D. W. Zingg, "Airfoil computations using a Newton-GMRES method," in *Proceedings of the 4th Annual Conference*, CFD Society of Canada, 1996.

17. Rogers, S. E., "A comparison of implicit schemes for the incompressible Navier–Stokes equations with artificial compressibility," Tech. Rep. AIAA–95–0567, AIAA, 1995. 33rd Aerospace Sciences Meeting and Exhibit.

18. Saad, Y., "Krylov subspace techniques, conjugate gradients, preconditioning and sparse matrix solvers," in *Computational Fluid Dynamics* (H. Deconinck, ed.), vol. 5 of *Lecture Series*, Von Karmán Institute for Fluid Dynamics, 1994.

19. Saad, Y. and M. H. Schultz, "GMRES: A generalized minimum residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Stat. Computing*, vol. 7, no. 3, pp. 856–869, 1986.

20. Venkatakrishnan, V., "Newton solution of inviscid and viscous problems," Tech. Rep. AIAA–88–0413, AIAA, 1988. 26th Aerospace Sciences Meeting.

21. Venkatakrishnan, V. and D. J. Mavriplis, "Implicit solvers for unstructured meshes," *Journal of Computational Physics*, vol. 105, pp. 83–91, 1993.