A FULLY COUPLED NEWTON-KRYLOV SOLVER FOR TURBULENT AERODYNAMICS FLOWS

Todd Chisholm and David W. Zingg
University of Toronto Institute for Aerospace Studies
4925 Dufferin Street, Toronto, ON
Canada, M3H 5T6
todd@oddjob.utias.utoronto.ca

Keywords: *airfoil, Navier–Stokes, high-lift, Newton's method, GMRES*

Abstract

A fast Newton-Krylov algorithm is presented for solving the compressible Navier-Stokes equations on structured multi-block grids with application to turbulent aerodynamic flows. The one-equation Spalart-Allmaras model is used to provide the turbulent viscosity. The optimization of the algorithm is discussed. ILU(4) is suggested for a preconditioner, operating on a modified Jacobian matrix. An efficient startup method to bring the system into the region of convergence of Newton's method is given. Three test cases are used to demonstrate convergence rates. Single-element cases are solved in less than 100 seconds on an engineering workstation, while the solution of a multi-element case can be found in less than 25 minutes.

1 Introduction

Recently, Newton-Krylov methods have been shown to be very effective in reducing the time required to compute numerical solutions to the Navier-Stokes equations. Blanco and Zingg [4] studied the solution of the Euler equations on unstructured grids with a matrix-free Newton-Krylov method. Geuzaine [7] used a similar method with the compressible Navier-Stokes equations, modeling turbulence with the Spalart-Allmaras model. Barth and Linton [3] studied a parallel implemention of a Newton-Krylov solver on unstructured grids for two- and three-dimensional flows. Pueyo and Zingg [14] solved the turbulent, compressible Navier-Stokes equations on structured grids. They have demonstrated that this approach is competitive with state of the art multigrid methods. However, their work was limited to representing turbulence with the algebraic Baldwin-Lomax model on single block grids. Here we discuss the solution of the Navier-Stokes equations on single- and multi-element airfoils, using the one-equation Spalart-Allmaras turbulence model [17].

2 Algorithm Description

2.1 Governing Equations

We study the solution of the steady compressible thin-layer Navier-Stokes equations on structured grids. A generalized curvilinear coordinate transformation is used to map the physical space to a rectangular computational domain. The use of multiple blocks allows for complex geometries such as multi-element airfoils. A circulation correction is used to reduce the effect of the farfield boundary. The Spalart-Allmaras turbulence model, including trip terms, is implemented as described in [8], with a small change in the calculation of the modified voriticity factor, first used by Ashford [1]

$$\tilde{S} = S f_{v3} + \frac{\tilde{\nu}_T}{\kappa^2 d^2} f_{v2} \quad (1)$$

$$f_{v2} = \left(1 + \frac{\chi}{c_{v2}}\right)^{-3} \quad (2)$$

$$f_{v3} = \frac{(1+\chi f_{v1})(1-f_{v2})}{\chi} \quad (3)$$

with $c_{v2} = 5.0$. The original form, which allowed $\tilde{S}$ to become negative, introduced a local minimum quite close to the solution root at some nodes at the edge of recirculation bubbles. This would cause the residual to hang, despite the majority of the flow being converged. The new form seems to avoid this problem.

2.2 Spatial Discretization

The spatial discretization follows that used by Nelson, et al. [11]. Second-order centred differences are used to approximate derivatives. Both Jameson's [9] scalar and Swanson and Turkel's [18] matrix second- and fourth-difference dissipation models can be used to stabilize the centred difference scheme. A pressure switch is used to control the activation of second-difference dissipation. The matrix dissipation model uses two switches V_l and V_n to avoid the effect of overly small eigenvalues in the flux Jacobian matrix. We use $V_l = V_n = 0.1$ for subsonic cases, and $V_l = 0.025$ and $V_n = 0.25$ for transonic cases. The turbulent viscosity convection and diffusion terms are discretized using first-order upwinding and second-order centred differencing, respectively, as suggested in [17], with two differences. The turbulence equation is scaled by J^{-1}, the grid-metric Jacobian, and the state variable $\tilde{\nu}$ is replaced with $J^{-1}\tilde{\nu}$. These changes help keep the entries in each block of the Jacobian of similar magnitude. This improves the conditioning of the system, making the Krylov solver more efficient.

Since a Newton solver is used to solve the resulting nonlinear system, it is important that all of the boundaries be handled fully implicitly. This includes the interfaces between blocks. The Navier-Stokes equations are solved on these interfaces in the same manner as the interior nodes.

2.3 Newton-Krylov Algorithm

The Nonlinear System. After spatial discretization, we have a system of the form

$$R(\hat{Q}^*) = 0 \quad (4)$$

where each block of $\hat{Q}$, the conservative state variables with the turbulence variable, is

$$\hat{Q}_i = J_i^{-1} Q_i = J_i^{-1}[\rho_i, \rho u_i, \rho v_i, e_i, \tilde{\nu}_i]^T$$

To find $\hat{Q}^*$ which satisfies Eq. 4, we apply the implicit Euler method repeatedly until some convergence criterion, typically $\|R\|_2 < 10^{-12}$, is reached:

$$\left[\frac{I}{\Delta t_n} - \frac{\partial R_n}{\partial \hat{Q}_n}\right] \Delta\hat{Q}_n = R_n$$

$$\hat{Q}_{n+1} = \hat{Q}_n + \Delta\hat{Q}_n$$

We call these the *outer* iterations. When the time step is increased towards infinity, Newton's method is approached. If Δt is increased appropriately as $\|R\|$ decreases, the quadratic convergence characteristic of Newton's method can be achieved, while dramatically increasing the region of convergence. Note that, in order for Newton's method to converge quadratically, $\frac{\partial R}{\partial Q}$ must be accurate. This requires that the equations be fully coupled.

The Linear System. In order for $\Delta\hat{Q}_n$ to be found, a linear system needs to be solved. This system tends to be very large, so that direct solution is prohibitive in both memory and time. Fortunately, finding the exact $\Delta\hat{Q}$ is not necessary, and we may settle for finding an approximation. This is an inexact-Newton method. There are a number of popular methods of finding the approximate solution of the linear system. The proper selection and use of this method is crucial to the success of the overall solver [2]. The most successful class are the Krylov iterative methods. Specifically, the preconditioned Generalized Minimum Residual (GMRES)[16] has proven to be effective for aerodynamic systems. We call these linear iterations the *inner* iterations.

Over-solving the linear system needs to be avoided for efficiency. A stopping criterion is needed for the inner iterations. There are two considerations. First, we use a target reduction in the inner residual. Pueyo [13] found a one order of magnitude reduction ideal to balance outer and inner iteration efficiency. This is appropriate in the turbulent case during the final Newton stage, but not in the startup. This will be discussed in the next section. The second consideration is setting the maximum number of iterations of GMRES. The amount of memory and CPU time increases with each GMRES iteration, so a limit is required. GMRES may be restarted, which keeps the memory requirements lower, while allowing further solution of the linear system. However, this can significantly slow the linear system convergence, due to the very poor conditioning seen in these systems. Typically, we do not use restarting for this reason.

The convergence rate of GMRES is very sensitive to the condition number of the matrix. Since the Jacobian of the equations being solved is typically extremely ill-conditioned, a good preconditioner is required to limit the number of inner iterations. Pueyo and Zingg [14] have shown that an incomplete LU preconditioner (ILU) [5] with two levels of fill minimizes solution time. They also found that a preconditioner based on a first-order Jacobian is more efficient than the exact Jacobian, both in saving memory and CPU time. The first-order Jacobian is formed by using only second-difference dissipation. This reduces the number of entries per equation to five instead of nine. It tends to give a better conditioned matrix, which leads to a more stable LU factorization. The coefficient of the second-difference dissipation used in the Jacobian matrix is found by

$$\varepsilon_2^l = \varepsilon_2^r + \sigma\varepsilon_4^r$$

where σ is found empirically.

Pueyo and Zingg used scalar artificial dissipation. The use of matrix dissipation significantly worsens the conditioning of the preconditioner, due to a reduction in diagonal dominance. This requires further modifications. Two methods have been investigated. The most obvious is to include a time step. In order to achieve preconditioner stability, a sufficiently small time step needs to be taken. This is due to the existence of negative entries on the diagonal. Large time steps (with correspondingly small additions to the diagonal) have the possibility of worsening the condition of the matrix by reducing the diagonal. Small time steps are obviously not desirable, as they dramatically slow outer residual convergence. For this reason, we add the time step only to the matrix used for the preconditioner.

Another method of increasing diagonal dominance is to increase the value of the matrix dissipation switches (V_n and V_l) used in the first-order Jacobian matrix the factorization is based on. These switches set the minimum level of dissipation, making them a natural choice for controlling small diagonals. As they are increased toward unity, scalar dissipation is approached. Note that the switches in the residual evaluation remain the same, so that the final solution is unaffected. The switches in the Jacobian used for the outer iterations are also unchanged, as inaccuracies there can adversely affect convergence. There are two conflicting effects that have to be balanced to optimize the preconditioner settings. The preconditioner must be conditioned well enough to be stable, but remain close enough to the true Jacobian to provide adequate clustering of the eigenvalues. A useful tool in evaluating a preconditioner is its condition number estimate, as discussed by Chow and Saad [5]. This is simply the L_2 norm of the solution to $LU \cdot \vec{c} = \vec{1}$. If a preconditioner is performing poorly, and has a high condition number estimate of the order of 10^7 or greater, more diagonal dominance is needed. It is usually better to err on the side of being too well conditioned. While this will slow convergence somewhat, the linear solves are much more robust.

To help reduce the effect of entries dropped from the preconditioner, reverse Cuthill-McKee reordering is used [6]. Good reordering is especially important in the multiblock case, due to the increased number of far off-diagonal entries resulting from the block boundaries.

The GMRES algorithm only requires matrix-vector multiplies, and does not explicitly require the matrix, except in forming the preconditioner. A Jacobian-free implementation of GMRES may be used, which has been found by Pueyo [13] to be faster, as well as resulting in significant memory savings. Unfortunately, the necessary changes in the Jacobian during startup eliminate this benefit when the Spalart-Allmaras turbulence model is used.

Startup. Grid sequencing is used to help rapidly eliminate initial transients. This involves partially solving on one or more coarse grids, each formed by removing every other grid line in each direction from the next finer grid. The solution is interpolated from the coarse grid to the fine grid. This helps to bring the solution on the fine grid closer to the region of convergence of Newton's method, allowing higher initial time steps. Grid sequencing is also particularly helpful in initializing the turbulence quantities. The model will tend to take a number of iterations at a low time step before tripping occurs. These iterations are very expensive to perform on the fine grid, but can be rapidly done on a twice-coarsened grid. It is important to ensure that the flow on the coarsened grids are tripped before passing to the fine grid. If a region is not properly tripped when the Newton's stage is started, divergence is likely. Raising the trip coefficient c_{t1} in the turbulence model help encourage transition on the coarse grids.

Examination of the production and destruction terms of the turbulence model reveals that these terms are unstable with negative $\tilde{\nu}$. Therefore, it is crucial to take steps to ensure that $\tilde{\nu}$ remain positive. This is a particular problem during the early iterations, when the solution is rapidly changing. There are a number of strategies which can be used to avoid negative values. Spalart and Allmaras [17] recommend that a modified linearization of the equations be used during startup. This modifies the turbulence model Jacobian so that it becomes an M matrix. The flow portion of the matrix is unchanged. While this prevents quadratic convergence, it was found that this modification is only necessary during the implicit Euler stages. This modification will be effective only if the linear system is solved sufficiently well. If the same tolerance appropriate for the laminar equations is used, large negative turbulence viscosities show up very quickly, despite using a small time step. Inner tolerances of approximately 10^{-4} are necessary to see the advantage of the modified turbulence model Jacobian. This is not nearly as detrimental as it would seem at first glance. The first few orders of magnitude reduction of the linear residual happen much faster than in the laminar case, often in only two or three iterations. This phenomenon only seems to occur when the modified Jacobian is used. Note that during startup, Jacobian-free GMRES cannot be used, since non-negative turbulent viscosities cannot be guaranteed.

Nemec and Zingg [12] used approximate factorization with the modified Jacobian during startup with good results. This is an inexpensive approach to approximately solving the linear system. The turbulence quantities are decoupled from the mean flow equations, and the linear system for each block is solved separately.

The approach followed by Geuzaine [7] was to use a variable time step based on the switched evolution relaxation method of Mulder and van Leer [10]. The time step in the early iterations is limited, so that the updates to the turbulence quantity are well bounded.

We have found that a combination of these methods works well. Starting with the modified Jacobian allows larger time steps while maintaining stability. Variable time steps progress the solution rapidly. The method of adjusting the reference time step is somewhat different than SER, though similar in spirit:

$$\Delta t_{ref} = \alpha \left[\frac{1}{||R||_2} \right]^{\beta} \tag{5}$$

Choosing $\alpha = 10.0$ and $\beta = 1.0$ gives good results. The time step of the both the Navier-Stokes and turbulence model equations also vary spacially. The geometric time step following Pulliam [15] is used for the flow equations:

$$\Delta t = \frac{\Delta t_{ref}}{1 + \sqrt{J}} \tag{6}$$

where J is the Jacobian of the metric of the curvilinear coordinate transformation. The turbulence model uses a time step based on the distance to the closest wall:

$$\Delta t = \Delta t_{ref} \cdot \tau \cdot d^2 \quad (7)$$

This is inspired by the destruction term of the model, which is proportional to the inverse of the square of the distance. Virtually all instabilities occur in the region close to the body, where the destruction term is strong. τ allows us to set an appropriate scaling for the turbulence model time step. 10.0 gives good convergence while keeping the model stable.

Even with these changes, negative values of $\tilde{\nu}$ are likely to be found, especially after interpolation between grids. These values are clipped to zero after each update. Experiments have been carried out with other techniques including modifying the production and destruction terms for negative values, with varying degrees of success. Simple clipping seems to be the most robust, and allows more aggressive time steps.

Matrix dissipation tends to be somewhat more unstable than scalar, especially during startup. Scalar dissipation is used on the coarse grids for this reason. The interpolation error also seems to eliminate any gains which could be had by using matrix dissipation on the coarse grids.

3 Test Cases

Three test cases are presented. Two are single-element, one subsonic, the other transonic. The third case is an airfoil with a detached flap and slat, at low Mach number and high angle of attack. The flow conditions are shown in Table 1, grid details in Table 2. Off-wall spacing is given relative to chord length. Figure 1 shows the grid around the multi-element airfoil.

The GMRES iterations for the single-element cases are limited to thirty search directions, with one restart. This is not sufficient for the larger, multiblock case. Sixty search directions with no restarts allow the linear solver to converge sufficiently for this case.

Case	Mach	Alpha	Re$\cdot 10^6$	Airfoil
1	0.3	6.0°	9.0	NACA0012
2	0.729	2.31°	6.5	RAE2822
3	0.197	20.18°	3.52	A2

Table 1 Flow conditions

Case	Dimensions	Nodes	Offwall Spacing
1	305× 57	17385	10^{-6}
2	257× 57	14619	$2 \cdot 10^{-6}$
3	–	71868	10^{-6}

Table 2 Grids

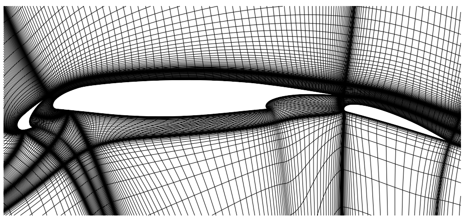

Fig. 1 Case 3 grid

	Inner iterations	
σ	Case 1	Case 2
4	430	div
5	323	263
6	334	291
7	379	308
8	401	321
9	483	345

Table 3 Inner iterations vs. σ

4 Algorithm Optimization

Choosing a proper startup sequence is crucial to obtaining maximum efficiency. The Newton steps will zero the residual much more quickly than the implicit Euler steps. We have had success with the following. Use $\Delta t_{ref} = 50$ on the coarsest grid, until the turbulence model fully trips. This is indicated by a peak in the turbulent residual. While the maximum value varies strongly by case, a minimum number of iterations of 15 should ensure tripping. At this point, the turbulence has stabilized enough to use a time step of 500. When $||R||_2 < 0.01$, the solution is interpolated to the next grid. One iteration at a time step of 50 is used to smooth the interpolation error. After this $\Delta t_{ref} = 500$ is used until $||R||_2 < 0.01$, and the solution is interpolated to the final grid. At this point, the variable time step is used along with the modified Jacobian, until $||R||_2 < 0.01$. The unmodified Jacobian and variabled time step are used until convergence. Note that the residual vector is comprised of both the four mean flow equations and the turbulence equation. The norm, and therefore the transition between stages, is typically dominated by the turbulence residual. This is appropriate, considering that the turbulence model is less stable than the flow equations. This method seems to be reasonably robust for the aerodynamic flows under consideration. Further convergence of the coarse grids does not appear to be beneficial. This seems to result from the turbulent viscosity not interpolating well. Three grids are used in all cases.

Table 3 shows the effect of varying σ on the total number of inner iterations needed to reach $||R||_2 < 10^{-12}$ on the fine grid for cases 1 and 2. *div* indicates that the case diverged. Based on these results, $\sigma = 5$ was chosen.

The transonic case is the best conditioned of the tests. This is likely due to the higher Mach number, increased dissipation, and relatively low number of grid nodes. It is appropriate to simply apply a time step of 100.0 to the preconditioner to stabilize the LU factorization when solving transonic cases. Cases 1 and 3 demand more attention, due to the low Mach numbers, and in case 3, the large number of nodes. In these cases a combination of modified switch values and a time step is appropriate. $V_l = V_n = 0.4$ with $\Delta t = 100.0$ is a good choice. Note that the switch values in the residual and exact Jacobian remain unchanged.

As mentioned previously, GMRES does not require the matrix to be explicitly formed. There is a trade-off in speed between matrix-free and matrix-explicit GMRES, which depends on the number of linear iterations. The former requires one residual evaluation per iteration. The latter requires a matrix construction when beginning the linear solve, plus a matrix-vector multiply per iteration. Since the matrix-vector multiply is cheaper than a residual evaluation, matrix-free GMRES becomes less efficient with more difficult systems. Figure 2 compares these two methods. The matrix-free solver requires 25% more time. This case, which has a particularly ill-conditioned matrix, takes roughly 30 inner iterations for each Newton step. This makes the matrix-explicit GMRES more attractive. Figure 3 shows a different situation for case 2, the transonic case. The matrix here is better conditioned, and needs about 15 inner iterations per Newton step. Due to the presence of the shock, the pressure switch has activated second-difference dissipation. Adding the linearization of the switch and the dissipation coefficient requires significant amounts of time, and slows the linear solves. However, without these terms in the Jacobian matrix, the convergence of the outer iterations is affected. Figure 3 compares matrix-free GMRES with matrix-explicit without the differentiation of

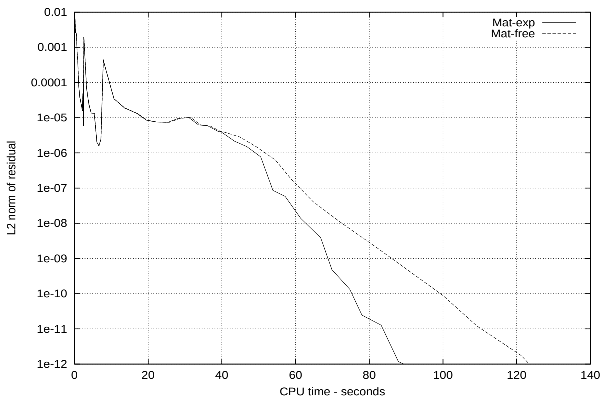

Fig. 2 Case 1 matrix-free vs. matrix-explicit GMRES

the switch and dissipation coefficient. Matrix-free is clearly the best choice for transonic cases.

The choice of level of fill in the preconditioner is important in balancing the memory use and CPU time. There is an optimum level of fill to minimize CPU time. Higher fills produce more powerful preconditioners, which reduce the number of inner iterations, but require significantly more time to factorize and apply. Table 4 shows residual convergence times for cases 1 and 2. Both cases show that ILU(4) is a good choice for minimizing CPU time. This adjustable parameter has the advantage of allowing a trade-off between memory and speed. The matrix formed when a finite time step is used is quite well conditioned, so that a preconditioner with a fill of two

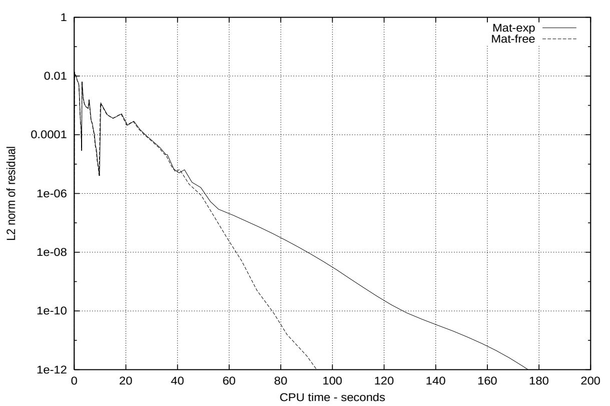

Fig. 3 Case 2 matrix-free vs. matrix-explicit GMRES

ILU fill	Case 1	Case 2
2	168	129
3	94	112
4	89	92
5	93	143

Table 4 Convergence time (seconds) vs. ILU fill

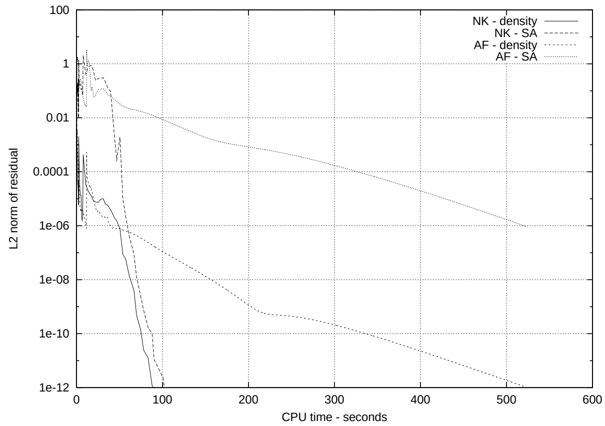

Fig. 4 Case 1 residual history

is sufficient during the startup phase.

5 Results

Figures 4, 5, and 6 compare the convergence histories of the Newton-Krylov solver with an approximately-factored solver. All cases used an ILU(4) preconditioner, and were run on an Alpha 667MHz EV67 CPU. Cases 1 and 3 are much faster than the approximate factored solver, while case 2 shows only a marginal improvement. This is due to the difficulties with the dissipation terms in the Jacobian, and the resulting use of matrix-free GMRES. Case 3 is not fully converged by the approximately factored solver. There are recirculation bubbles behind the slat, the cove on the main body, and at the trailing edge of the flap. Difficulties arise at the edges of these bubbles with the highly nonlinear destruction term of the turbulence model. Outside of these few nodes, the model is converged.

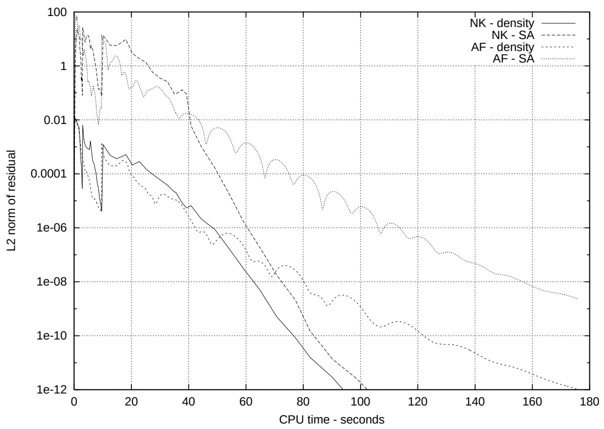

Fig. 5 Case 2 residual history

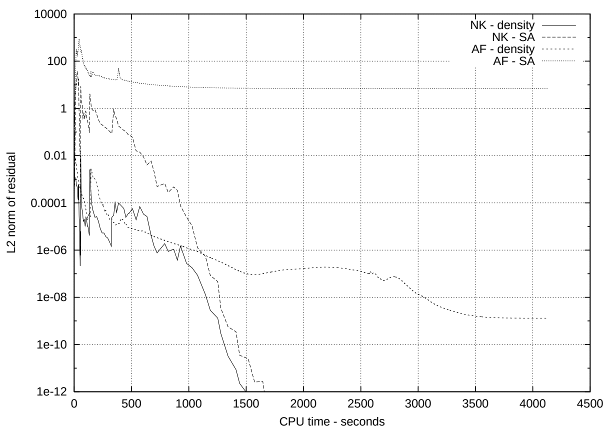

Fig. 6 Case 3 residual history

6 Conclusions

An efficient Newton-Krylov solver has been presented for the steady compressible Navier-Stokes equations governing turbulent flows over multi-element airfoils. Proper optimization is essential. This includes using grid sequencing and the implicit Euler method for startup. During this phase, the Jacobian matrix must be modified to ensure that the turbulent viscosity remains positive. Incomplete LU preconditioning is used. A level of fill of four was found to be optimal with respect to CPU time. The ILU factorization is applied to the first-order Jacobian matrix with modified second-difference dissipation. Time step selection and modified dissipation switches are important to stabilize the preconditioner. The single-element test cases can be solved to machine zero in less than 100 seconds, while the complex flow on the multi-element case can be found in less than 25 minutes. The subsonic cases converge three to five times faster than an approximately factored algorithm.

7 Acknowledgments

This research was supported by Bombardier Aerospace, and an OGSST grant from the Government of Ontario.

References

[1] Ashford G. A. *An Unstructured Grid Generation and Adaptive Solution Technique for High Reynolds Number Compressible Flows*. PhD thesis, University of Michigan, 1996.

[2] Barrett R, Berry M, Chan T, Demmel J, Donato J, Dongarra J, Eijkhout V, Pozo R, Romine C, and van der Vorst H. Templates for the solution of linear systems: building blocks for iterative methods. Technical report, SIAM, 1994.

[3] Barth T. J and Linton S. W. An unstructured mesh Newton solver for compressible fluid flow and its parallel implementation. 1995. AIAA-95-0221.

[4] Blanco M and Zingg D. W. Fast Newton-Krylov method for unstructured grids. *AIAA Journal*, Vol. 36, No 4, pp 607–612, April 1998.

[5] Chow E and Saad Y. Experimental study of ILU preconditioners for indefinite matrices. *J. of Comp. and Appl. Mathematics*, Vol. 87, pp 387–414, 1997.

[6] Cuthill E. H and McKee J. M. Reducing the bandwidth of sparse symmetric matrices. *Proc Proceedings of the 24th National Conference of the Association for Computing Machinery*, pp 157–172. Brondon Press, 1969.

[7] Geuzaine P. Newton-Krylov strategy for compressible turbulent flows on unstructured meshes. *AIAA Journal*, Vol. 39, No 3, pp 528–531, 1997.

[8] Godin P, Zingg D. W, and Nelson T. High-lift aerodynamics calculations with one- and two-equation turbulence models. *AIAA Journal*, Vol. 35, No 11, Feb. 1997.

[9] Jameson A, Schmidt W, and Turkel E. Numerical solutions of the Eeuler equations by finite volume methods using Runge-Kutta time-stepping schemes. June 1981. AIAA paper 81-1259.

[10] Mulder W. A and van Leer B. Experiments with implicit upwind methods for the Euler equations. *J. Comp. Phys.*, Vol. 59, pp 232–246, 1985.

[11] Nelson T. E, Zingg D. W, and Johnston G. W. Compressible Navier-Stokes computations of multielement airfoil flows using multiblock grids. *AIAA Journal*, Vol. 32, No 3, pp 506–511, 1997.

[12] Nemec M and Zingg D. W. Newton-Krylov algorithm for aerodynamic design using the Navier-Stokes equations. *AIAA Journal*, Vol. 40, No 6, pp 1146–1154, 2002.

[13] Pueyo A and Zingg D. W. Airfoil computations using a Newton-GMRES method. *Proc Proceedings of the Fourth Annual Conference of the CFD Society of Canada*. CFD96, June 1997.

[14] Pueyo A and Zingg D. W. Efficient Newton-Krylov solver for aerodynamic calculations. *AIAA Journal*, Vol. 36, No 11, pp 1991–1997, 1998.

[15] Pulliam T. H. Efficient solution methods for the Navier-Stokes equations. Lecture Notes for The Von Karman Institute, January 1986.

[16] Saad Y and Schultz M. H. GMRES: a generalized minimum residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Computing*, Vol. 7, pp 856–869, 1986.

[17] Spalart P and Allmaras S. A one-equation turbulence model for aerodynamic flows. 1992. AIAA paper 92-0439.

[18] Swanson R. C and Turkel E. On central-difference and upwind schemes. *J. Comp. Phys.*, Vol. 101, pp 292–306, 1992.