A Three-Dimensional Multi-Block Newton-Krylov Flow Solver for the Euler Equations

Jason C. Nichols[*] and David W. Zingg[†]

University of Toronto Institute for Aerospace Studies

4925 Dufferin Street, Toronto, Ontario, M3H 5T6, Canada

A three-dimensional multi-block Newton-Krylov flow solver for the Euler equations has been developed for steady aerodynamic flows. The solution is computed through a Jacobian-free inexact-Newton method with an approximate-Newton method for startup. The linear system at each outer iteration is solved using a Generalized Minimal Residual (GMRES) Krylov subspace algorithm. An incomplete lower/upper (ILU) factored preconditioner with reverse Cuthill-McKee reordering is utilized to increase the efficiency of GMRES. The parameters in the solver are optimized to provide a balance between speed and robustness. Tests are performed using a variety of flow conditions and grid sizes. The solver demonstrates fast convergence and good correlation with experimental data.

I. Introduction

A fast, accurate and robust algorithm is necessary to solve three-dimensional aerodynamic flows for use in shape optimization. Until recently, solving the flow over a wing in three dimensions has been hindered by large grid sizes, inefficient solution methods and maximum computer clock speeds. New numerical methods and faster computers allow full solutions in three dimensions to be calculated in a reasonable amount of time.[1] The objective of this research is to create a multi-block three-dimensional Newton-Krylov flow solver for inviscid flow using structured grids. As an implicit scheme, Newton's method is appealing due to its property of quadratic convergence. Advantages of Newton-Krylov methods have been proven by authors such as Venkatakrishnan and Mavriplis,[2] Nielsen et al.,[3] Luo et al.[4] and Knoll and Keyes.[5] The work presented in this paper follows on successful two-dimensional algorithms of Pueyo and Zingg,[6] Blanco and Zingg,[7] Nemec and Zingg[8] and Chisholm and Zingg.[9] This algorithm is intended to lay the ground-work for a 3D Navier-Stokes and 3D optimization program.

The feasibility of a Newton-Krylov method has been well established, but the crux of the method is in the startup. A balance between the speed versus robustness of the algorithm is a challenging issue. Tests were conducted, and the results and justification of the parameters selected are presented in this paper. The flow solver as presented in this work is known as TYPHOON.[10]

II. Three-dimensional Euler Equations

The conservative form of the three-dimensional Euler equations is:

$$\frac{\partial Q}{\partial t} + \frac{\partial E}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial G}{\partial z} = 0 \tag{1}$$

[*] Aeronautical Engineer, E.I.T., Pratt & Whitney Canada, and AIAA Member

[†] Professor, Senior Canada Research Chair in Computational Aerodynamics, 2004 Guggenheim Fellow, Senior AIAA Member

The non-dimensional conservative variables and inviscid fluxes are given by:

$$Q = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{pmatrix} \quad E = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(e+p) \end{pmatrix} \quad F = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vw \\ v(e+p) \end{pmatrix} \quad G = \begin{pmatrix} \rho w \\ \rho wu \\ \rho wv \\ \rho w^2 + p \\ w(e+p) \end{pmatrix} \tag{2}$$

A general transformation is used to map the curvilinear grid (x,y,z) into a computational domain (ξ,η,ζ) where the spacing is uniform and equal to unity. Transforming the grid points from physical to computational space involves calculation of nine metric terms and the metric Jacobian (J) as follows:

$$\begin{array}{lll} \xi_x = J\left(y_\eta z_\zeta - y_\zeta z_\eta\right) & \eta_x = J\left(z_\xi y_\zeta - y_\xi z_\zeta\right) & \zeta_x = J\left(y_\xi z_\eta - z_\xi y_\eta\right) \\ \xi_y = J\left(z_\eta x_\zeta - x_\eta z_\zeta\right) & \eta_y = J\left(x_\xi z_\zeta - x_\zeta z_\xi\right) & \zeta_y = J\left(x_\eta z_\xi - x_\xi z_\eta\right) \\ \xi_z = J\left(x_\eta y_\zeta - y_\eta x_\zeta\right) & \eta_z = J\left(y_\xi x_\zeta - x_\xi y_\zeta\right) & \zeta_z = J\left(x_\xi y_\eta - y_\xi x_\eta\right) \end{array} \tag{3}$$

$$J^{-1} = \left(x_\xi y_\eta z_\zeta + x_\zeta y_\xi z_\eta + x_\eta y_\zeta z_\xi - x_\xi y_\zeta z_\eta - x_\eta y_\xi z_\zeta - x_\zeta y_\eta z_\xi\right) \tag{4}$$

Since centered differencing is used to calculate the metrics in three-dimensions, the metric invariants will not be automatically satisfied as in 2D. As shown by Pulliam[11] and Steger,[12] averaging the central differences of the grid values $(x_\xi, x_\eta, x_\zeta$ etc.) and then calculating the metrics will satisfy the metric invariants. For example, ξ_x is calculated as:

$$\xi_x = J\left[(\mu_\zeta \delta_\eta y)(\mu_\eta \delta_\zeta z) - (\mu_\eta \delta_\zeta y)(\mu_\zeta \delta_\eta z)\right] \tag{5}$$

where δ is the central difference operator and μ is an average operator:

$$\mu_\eta x_k = \frac{1}{2}\left(x_{k+1} + x_{k-1}\right) \tag{6}$$

III. Three-dimensional Multi-block

Figure 1 shows a twelve-block, 28,000-node grid around the ONERA M6 wing. The multi-block grid is set up such that each block is independent while the overall process remains fully implicit. The block faces share physical space but exist as two separate computational planes of points. Formation of the equations is completed inside each block separately and block to block communication is carried out using temporary or halo nodes that hold information from the neighbouring block. The equations are not solved on the halo nodes, they are only used to provide the current block with information about the neighbouring block. The entire left-hand-side matrix is formed, reordered, and then the system of equations is solved. This method of information transfer has proven successful by authors such as De Rango,[13] De Rango and Zingg,[14] Epstein and Seror[15] and Kim and Lee.[16]

IV. Numerical Method

Spatial discretization

TYPHOON solves the governing equations using central-differencing with the scalar artificial dissipation of Jameson et al.[17] Following the spatial discretization, the system has the form

$$\frac{d\widehat{Q}}{dt} + R(\widehat{Q}) = 0 \tag{7}$$

where $\widehat{Q}$ is the discrete vector of conservative dependent flow variables with length five times the number of nodes.

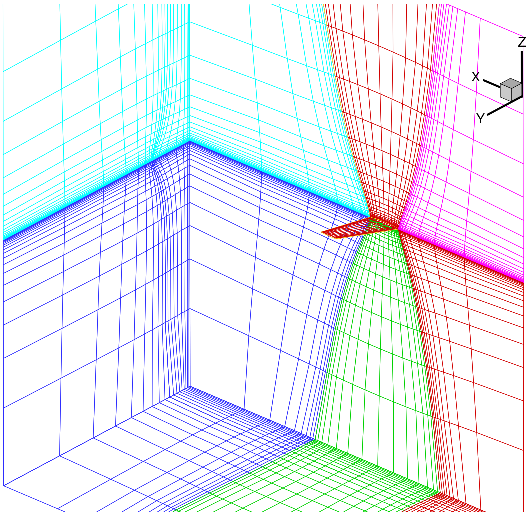


Figure 1. 28,000 node ONERA M6 grid

Inexact-Newton method

Once the equations are discretized, implicit Euler time-marching is applied. The resulting system is:

$$\left[\frac{I}{\Delta t} + \mathcal{A}\right]^{(n)} \Delta\widehat{Q}^{(n)} = -R^{(n)} \tag{8}$$

where $\mathcal{A}$ is the flow Jacobian and Δt is the time step. It can be noted that the implicit Euler time-marching method with local time-linearization yields Newton's method as $\Delta t \rightarrow \infty$.

There are many methods available to solve large sparse linear systems. Solving directly would be prohibitively expensive computationally. Several methods exist that solve the system iteratively, which is appealing due to smaller memory and computational requirements. It has been found to be faster to solve the linear system a few orders of magnitude rather than exactly.[18] This leads to the inexact-Newton method:

$$||R(\widehat{Q}^n) + \frac{\partial R}{\partial \widehat{Q}}\Delta\widehat{Q}|| \leq \tilde{\eta}||R(\widehat{Q}^n)|| \tag{9}$$

This equation allows the selection of the convergence tolerance ($\tilde{\eta}$). The full Newton method is recovered if $\tilde{\eta} = 0$ is selected. A value of $\tilde{\eta}$ that is too small will force *oversolving* of the system. This means that the linear system is solved past what is needed to have an effective update to the outer iteration. Conversely, if an $\tilde{\eta}$ that is too large is selected there is a possibility of *undersolving* the system. This could lead to the outer iterations stalling because a sufficiently accurate update to the linear problem was not found.

For aerodynamic applications it has been found by many authors[9, 19–23] that a Krylov subspace method works very well and is typically faster than other iterative solvers. The Generalized Minimal Residual (GMRES) method, developed by Saad and Schultz,[24] is a Krylov subspace method that iteratively solves a problem in the form of Eq. (8) ($Ax = b$). Two methods of solving the equations using GMRES were investigated: an approximate-Newton method which uses a first-order approximation of the Jacobian as well as a Jacobian-free method.

An interesting property of GMRES is that it only requires a matrix-vector product and does not need the actual Jacobian matrix. An approximation to the matrix-vector product can be made by taking the forward difference of the residual vector and adding a term associated with the finite timestep:

$$\left[\mathcal{A} + \frac{I}{\Delta t}\right] v \simeq \frac{R(\widehat{Q} + \varepsilon v) - R(\widehat{Q})}{\varepsilon} + \frac{v}{\Delta t} \tag{10}$$

where ε is a small scalar used to perturb the state quantities $\widehat{Q}$ in the direction of v. The value of ε is a balance between an inaccurate approximation (ε too large) and introducing roundoff error (ε too small). The value of ε is based on work by Nielsen[3] and is calculated using the following equation

$$\varepsilon \simeq \frac{\sqrt{\varepsilon_m}}{\bar{v}} \tag{11}$$

where $\bar{v}$ is the root mean square of v, and ε_m is the value of "machine zero" for the hardware being used. The accuracy of this approximation is better than that of the approximate-Newton method and therefore yields better convergence rates.

Preconditioning

The performance of iterative solvers can be increased significantly by preconditioning the linear system of equations.[25–27] Following preconditioning, the eigenvalue spectrum will be clustered near unity, and the efficiency and performance of GMRES will be greatly improved.[23] The preconditioner used in this work is based on an incomplete lower-upper (ILU) factorization of a first-order approximation to the flow Jacobian. The off-diagonal dominance of the Jacobian can be reduced by using only second-difference dissipation when forming it.[28] The left hand side coefficient of dissipation is formed using a linear combination of the right hand side according to the following formula:

$$\varepsilon_2^{(l)} = \varepsilon_2^{(r)} + \sigma\varepsilon_4^{(r)} \tag{12}$$

where σ is a constant left as a free parameter, and ϵ_2 and ϵ_4 are the second- and fourth-difference artificial dissipation terms, respectively. Using this approximation the preconditioner will be more diagonally dominant and easier to solve while still retaining adequate accuracy for the linear solver. The reverse Cuthill-McKee (RCM)[29] re-ordering scheme has been found to improve the efficiency of a preconditioner based on an ILU factorization[2,6,7] and is applied in TYPHOON.

Algorithm startup

Algorithm startup is the most critical phase when solving the equations using Newton's method. Speed is a dominant factor in designing a program, but robustness is equally important. By definition Newton's method works extremely well when the initial guess is near to the solution, but can diverge otherwise. The current algorithm uses a variable time step to modify Newton's method (Implicit Euler), which is an approach used successfully by many authors, such as Venkatakrishnan and Mavriplis,[2] Orkwis,[22] Chisholm and Zingg[30,31] and Manzano et al.[32]

TYPHOON uses a variable time step in conjunction with an approximate-Newton method and Jacobian-free GMRES in the solution process. The advantages of both the approximate-Newton and Jacobian-free strategies are combined into one hybrid method, as suggested by Blanco and Zingg.[7] The robustness in startup of the approximate-Newton method is combined with the rapid convergence of the Jacobian-free method.

The solution is converged two orders of magnitude using the approximate-Newton method. The Jacobian-free method is then used to converge the solution to machine zero. A drawback with the Jacobian-free method is that due to the presence of large non-linearities, even with small time steps this method has the potential to diverge. In the early iterations, inaccurate forward difference approximations to the flow Jacobian lead to possible solution divergence. It is desirable to reduce the amount of time spent using the approximate-Newton method since the convergence is slower, but if sufficient convergence is not attained, the Jacobian-free method could fail.

The Δt term in Eq. (8) is used to stabilize the system during startup and is also known as pseudo-transient timestepping. This time term strengthens the diagonal of the Jacobian, thereby increasing the efficiency of the linear solver. Based on Pulliam[11] the time step is scaled to the size of each cell thereby making it a local time step:

$$\Delta t^{(n)}_{j,k,m} = \frac{\Delta t^{(n)}_{ref}}{1 + \sqrt{J_{j,k,m}}} \tag{13}$$

Δt_{ref} is the reference time step and is increased as the solution proceeds toward convergence.

Approximate-Newton

This phase uses a first-order approximation ($\mathcal{A}_1$) to the flow Jacobian based on Eq. (12) in place of the true Jacobian. The system becomes less accurate, but requires less computation than the true Jacobian. If the first-order Jacobian is substituted into the system in Eq. (8) the result is:

$$\left[\frac{I}{\Delta t} + \mathcal{A}_1\right]^{(n)} \Delta\widehat{Q}^{(n)} = -R^{(n)} \tag{14}$$

This system is more diagonally dominant and easier to solve than one using the true Jacobian. This strategy is also found to be more robust than the Jacobian-free method during startup. Since this system is solving an approximate system, the convergence rate of the outer iterations is linear.

The time step used in the first phase is a simple geometric formula as used in the flow solver HURRICANE,[33, 34] which is given by:

$$\Delta t^0_{ref} = A; \qquad \Delta t^n_{ref} = B\Delta t^{n-1}_{ref} \tag{15}$$

where A and B are constants with values of $A = 1.0$ and $B = 1.7$. The algorithm begins using this preset time step sequence combined with the approximate-Newton method.

Jacobian-free GMRES

The second phase of the startup procedure involves a variable time step based on the residual and Jacobian-free GMRES. The Switched Evolution Relaxation method of Mulder and van Leer[35] is used to calculate the reference time step:

$$\Delta t^n_{ref} = \max\left[\frac{\alpha}{{R_d}^{\beta}}, \Delta t_{min}\right] \tag{16}$$

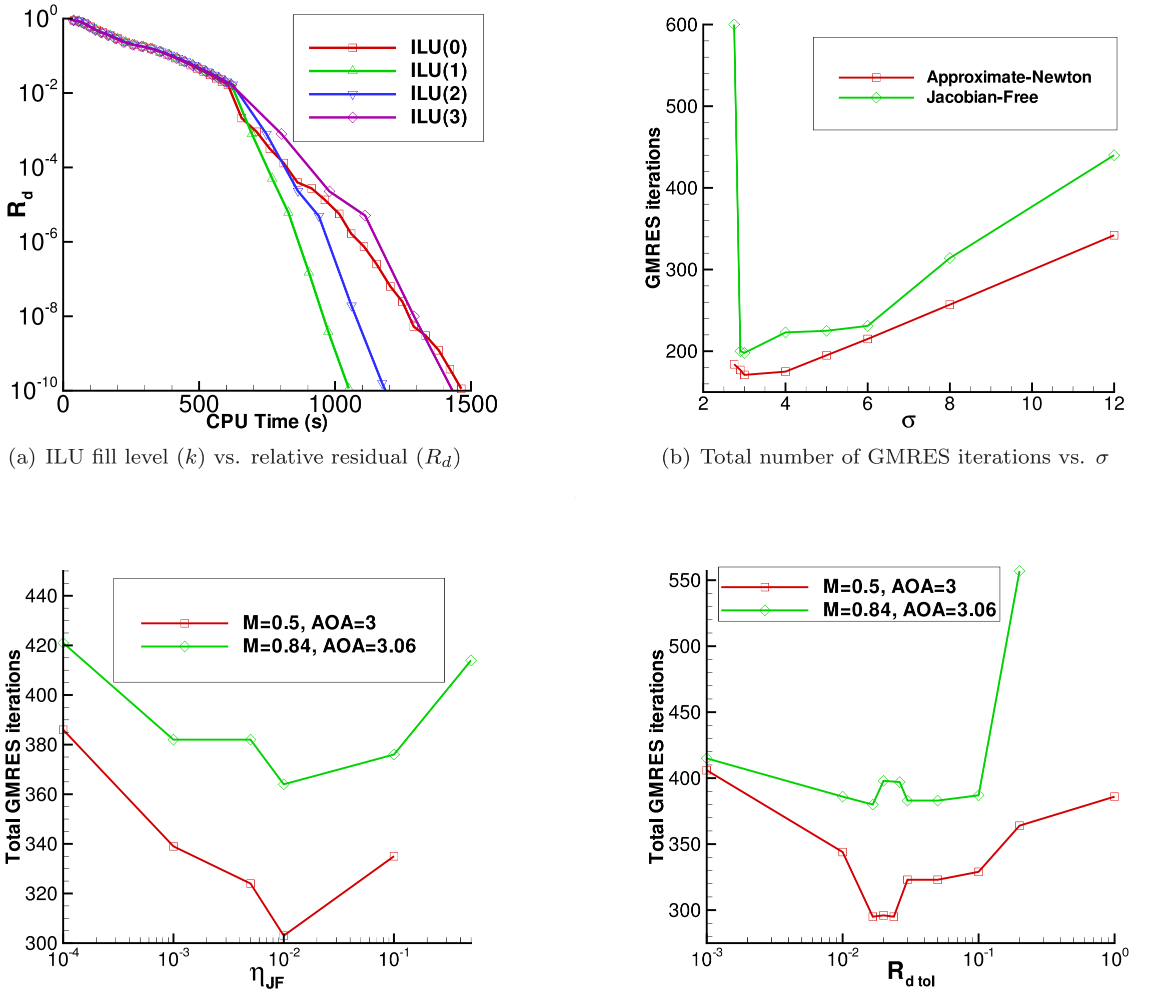
where α and β are constants, and Δt_{min} is set to 50. Typically, $\alpha = 1.0$ and $\beta = 1.3$. The relative residual $R^{(n)}_d$ is calculated as follows:

$$R^{(n)}_d = \frac{||R^{(n)}||}{||R^0||} \tag{17}$$

where $||R^0||$ is the starting residual.

V. Parametric studies

Parametric studies were conducted to investigate the effect of several key variables on the performance and stability of the flow solver. Since nonlinear equations are being solved, the parameters must be coupled; therefore, finding a truly optimum set of parameters is an iterative process. In most cases, a clear optimum can be found by assuming that the parameters are independent of each other. The parameters tested are: ILU fill level (k), preconditioner parameter (σ), linear system tolerance ($\tilde{\eta}$), and approximate-Newton convergence parameter ($R_{d_{tol}}$). The ONERA M6 wing geometry combined with a grid of $100,000$ nodes was used to test all the parameters at two flight conditions: subsonic $M = 0.5$, $\alpha = 3°$ and transonic $M = 0.84$, $\alpha = 3.06°$. Each parameter was tested individually while the others remained fixed at the following values: $k = 1$, $\sigma = 5$, $\tilde{\eta} = 0.01$ and $R_d = 0.01$.



(a) ILU fill level (k) vs. relative residual (R_d)

(b) Total number of GMRES iterations vs. σ

(c) $\tilde{\eta}_{JF}$ vs. total number of GMRES iterations

(d) Values of $R_{d_{tol}}$ vs. total number of GMRES iterations

Figure 2. Results of parameter studies

ILU fill level (k)

The effect of the fill level on the approximate-Newton and Jacobian-free phases was tested using various fill levels for both subsonic and transonic cases. The effect of level of fill on the convergence rate of the Jacobian-free method for the transonic case is shown in Figure 2(a). It can be seen that a fill level of unity achieves the fastest convergence. The results from this set of tests demonstrate that a fill level of unity ($k = 1$) is the best balance between a sufficiently accurate preconditioner and the computational time required to form and apply it. Similar results were also found for the approximate-Newton startup.

Preconditioner parameter (σ)

The formation of the preconditioner uses a parameter σ to combine the fourth-difference with the second-difference dissipation. Figure 2(b) shows how different values of σ affect the performance of the solver for the transonic case. Values of $\sigma \approx 3$ require fewer inner GMRES iterations and therefore less CPU time overall, but it was found that serious stability issues can arise when values of $\sigma \leq 3$ are used. It is recommended that $\sigma = 5$ be used as a good balance between speed and robustness.

Linear system tolerance ($\tilde{\eta}$)

The linear system tolerance ($\tilde{\eta}$) sets the convergence tolerance of the inner GMRES iterations. A value of 0.1 for the approximate-Newton linear system tolerance ($\tilde{\eta}_{AN}$) was found to be sufficient to minimize the total number of GMRES iterations. The results for the Jacobian-free parameter ($\tilde{\eta}_{JF}$) test are shown in Figure 2(c). It is clearly shown that a value of $\tilde{\eta} = 0.01$ minimizes the number of GMRES iterations.

Approximate-Newton convergence parameter ($R_{d_{tol}}$)

The parameter $R_{d_{tol}}$ controls the switch from the approximate-Newton phase to the Jacobian-free phase. The program must be converged sufficiently before switching to Jacobian-free operation or the solution can diverge. Figure 2(d) shows the results of the $R_{d_{tol}}$ parameter tests. Values of $R_{d_{tol}} = 0.02$ and $R_{d_{tol}} = 0.015$ were found to be minimums for the subsonic and transonic cases respectively. For flows with shocks, values of $R_{d_{tol}} \geq 0.1$ should be avoided. Difficult cases should use lower values of $R_{d_{tol}}$ to ensure that the solution is converged enough for the Jacobian-free method to take over.

Optimized algorithm

Based on the parameter studies, the optimal parameters are summarized below:

- ILU(1) preconditioner based on the first-order Jacobian with $\sigma = 5$
- Reverse Cuthill-McKee reordering based on an initial node located downstream
- Approximate-Newton startup convergence tolerance: $R_{d_{tol}} = 0.02$
- Approximate-Newton linear system tolerance: $\tilde{\eta}_{AN} = 0.1$
- Jacobian-free linear system tolerance: $\tilde{\eta}_{JF} = 0.01$
- GMRES limited to 40 search directions and no restart

VI. Test cases

TYPHOON is tested using four grids around the ONERA M6 wing geometry. Table 1 summarizes the details of the grids used. All have a 12 block H-mesh structured grid configuration and were created using the elliptical grid generator: GEM3D.[36,37] Four test cases containing a variety of flow conditions are used to show the results and performance of TYPHOON. Table 2 shows the details of these test cases. All test cases are inviscid without roll or yaw. Only the angle of attack (pitch) and Mach number are varied.

Table 1. ONERA M6 grids used to test TYPHOON

Grid	Grid Size	Far Field (semi-spans)	Off-wall Spacing ($\times 10^{-3}$ semi-spans)	Leading Edge Clustering ($\times 10^{-3}$ semi-spans)	Root Spacing ($\times 10^{-3}$ semi-spans)	Tip Spacing ($\times 10^{-3}$ semi-spans)	Nodes on Wing Surface
A	99,840	5	2.0	2.0	10.0	10.0	1,200
B	248,000	10	1.0	2.0	10.0	5.0	3,000
C	500,000	10	1.0	1.0	5.0	2.0	3,600
D	1,008,000	20	1.0	1.0	5.0	5.0	4,800

Table 2. Flow conditions for the four inviscid test cases

Case	Flow Condition	Mach Number	Angle of Attack (degrees)
1	Subsonic	0.3	8.0
2	Subsonic	0.5	3.0
3	Transonic	0.699	3.06
4	Transonic	0.84	3.06

Table 3. Convergence data for the coefficient of lift and drag for case 4

Convergence	TYPHOON grid size		
criterion	100,000	500,000	1,000,000
0.5% of C_L	5.9 min	42 min	90 min
0.1% of C_L	6.3 min	47 min	105 min
0.01% of C_L	6.5 min	67 min	127 min
0.5% of C_D	7.0 min	47 min	98 min
0.1% of C_D	8.0 min	64 min	118 min
0.01% of C_D	8.3 min	71 min	125 min

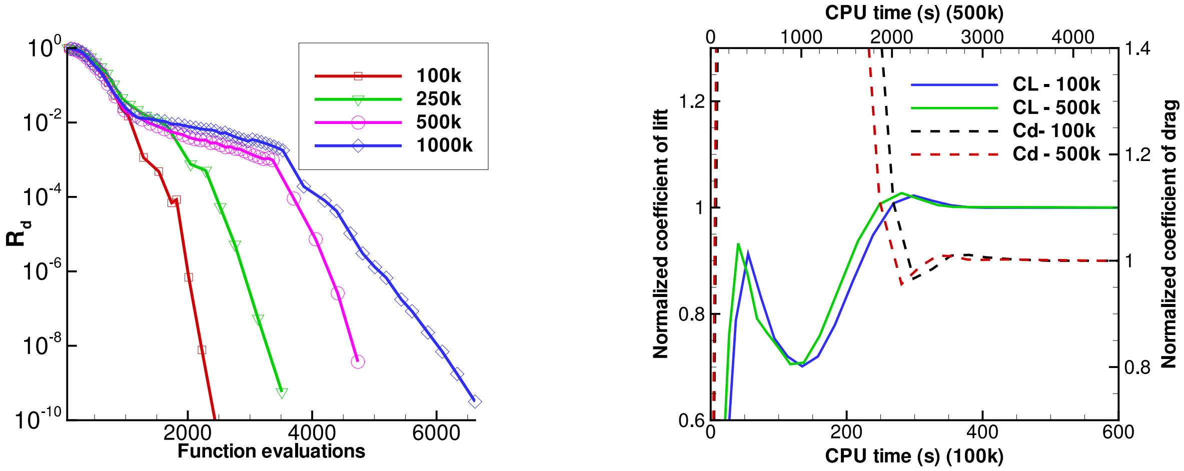
VII. Algorithm performance

The unit used to benchmark the performance of TYPHOON that is independent of computer architecture is the number of equivalent function evaluations (FE). A function evaluation is defined as one right-hand-side evaluation or one residual evaluation. The total number of FE's is calculated by dividing the total CPU time by the CPU time needed to calculate one right-hand-side. A single function evaluation includes calculation of the fluxes, pressure field, artificial dissipation and boundary conditions. First, TYPHOON is compared across all the grids using the subsonic case 2. Figure 3(a) shows the CPU time in function evaluations and allows comparison of the wide range of grid sizes on a similar scale. The 100,000 node grid requires about 2,400 function evaluations to converge to machine zero, while the 1,000,000 node grid requires about 6,300. A single 1000MHz EV68CB Alpha processor was used for testing and this corresponds to about 10 minutes for 100k nodes and 6.3 hours for the one million node case.

One of the goals of the flow solver is to calculate the coefficients of lift and drag, and another measure of code performance is how quickly these values are attained. Figure 3(b) shows the convergence history for the lift and drag coefficients on the 100k and 500k grids. Table 3 shows the convergence times to given criteria for the 100k, 500k and 1000k grids. On the 100k grid, TYPHOON produces results for the coefficient of lift to four decimal places in 6.5 minutes. For the same target, TYPHOON will solve the 500k node grid in 67 minutes and the one million node grid in 127 minutes. The resulting C_L and C_d values for the ONERA M6 were calculated to be 0.3048 and 0.0116 for the 100k grid, 0.3014 and 0.0108 for the 500k grid and 0.3031 and 0.0107 for the 1000k grid.

Grid scaling

An important aspect of a flow solver is the correlation between the CPU time and the problem size. From the tests conducted (cases 1 to 4) using grids A through D as well as a 50,000 and 75,000 node grid, the CPU time in function evaluations for convergence to machine zero is plotted as a function of the grid size in Figure 4. A similar analysis was performed by Pueyo and Zingg[28] on the PROBE solver[23] and ARC2D.[11] The results from Pueyo's tests along with the TYPHOON data are plotted. The equation of a line of best



(a) Residual vs. function evaluations for $M = 0.5$, $\alpha = 3°$

(b) Normalized C_L and C_D vs. CPU time for case 4

Figure 3. Algorithm performance graphs

Table 4. Newton-Krylov statistics

Grid	Case	Approximate-Newton		Jacobian-Free		Total Outer	Total Inner	$\frac{i-it}{o-it}$	FE
		Outer	Inner	Outer	Inner	Iterations	Iterations		
		Iters	Iters	Iters	Iters	(o-it)	(i-it)		
A	1	12	92	8	280	20	372	18.6	2796
A	2	12	88	8	229	20	317	15.9	2576
A	3	16	137	6	183	22	320	14.5	2616
A	4	17	165	8	220	25	385	15.4	3048
D	1	24	390	22	684	46	1074	23.3	6703
D	2	40	484	13	523	53	1007	19.0	6619
D	3	46	456	15	471	61	927	15.2	6619
D	4	31	412	9	467	40	879	22.0	5025

fit on the log-log plot can be represented by:

$$\omega = \kappa N^{\beta} \tag{18}$$

where N is the grid size, ω is the CPU time in function evaluations, and κ and β are constants. The ideal case for a flow solver is to have the function evaluations remain constant with the grid size ($\beta = 0$). For ARC2D: $\beta = 0.73$, PROBE: $\beta = 0.325$ and TYPHOON: $\beta = 0.288$. Therefore, the scalability of the 3D multi-block solver TYPHOON is found to be very good.

Newton-Krylov solver statistics

Table 4 shows the details of TYPHOON's inner and outer iterations and performance for grids A and D. The average number of inner iterations per outer iteration remains relatively constant and independent of grid size. The number of outer iterations is higher for larger grids due to the need for more approximate-Newton iterations during startup. During Jacobian-free operation, more inner iterations per outer iteration are needed when compared to the approximate-Newton phase.

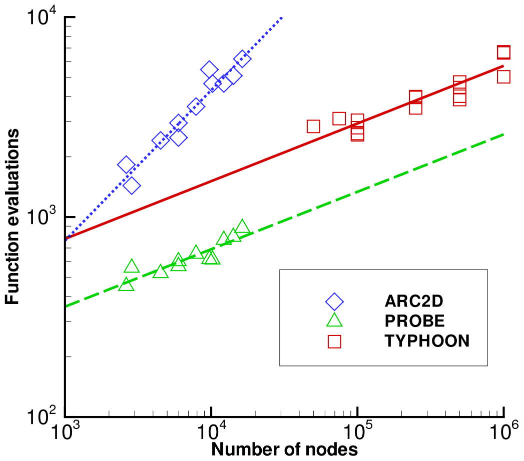


Figure 4. Grid scaling comparison

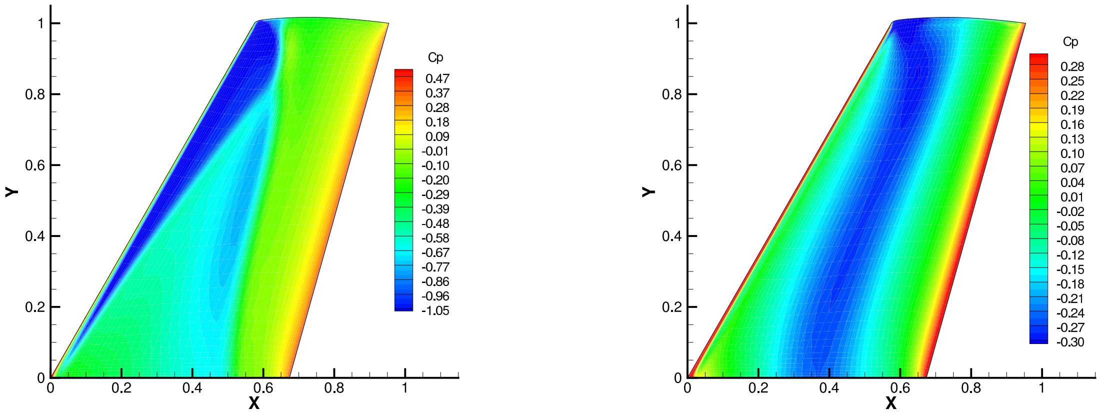
VIII. Solutions

Solutions are compared to a classic set of experimental results for the ONERA M6 wing by Charpin and Schmitt[38] in 1979. The test case used here is transonic flow at $M = 0.84$, $\alpha = 3.06°$ (Case 4 from Table 2). The most distinctive feature of this case is the pattern of the shock wave on the upper surface of the wing. The λ shock pattern on the upper surface can be seen clearly in Figure 5(a). This figure shows the coefficient of pressure (C_p) contours on the top surface (a) and on the bottom surface (b) for grid D. To compare the results of TYPHOON to the experiment, it is necessary to take spanwise slices on the wing and compare the coefficient of pressure (C_p). Six slices are plotted in Figure 6. Three grid sizes are overlaid along with the experimental data in this figure. Overall, the results correspond well to the experimental data. TYPHOON predicts the leading edge pressure rise very well and captures the shock with minimal overshoot.

IX. Conclusions

A three-dimensional multi-block Newton-Krylov flow solver for the Euler equations has been developed for steady aerodynamic flows. A Krylov subspace method (GMRES) is used to solve the linear system that arises at each Newton iteration. This linear system is preconditioned using an incomplete lower/upper (ILU) factorization, which increases the effectiveness of the linear solver. A combination of approximate-Newton and Jacobian-free solution methods are combined to form a robust and fast algorithm capable of handling a variety of flow conditions on various grids. The main focus of the work is to find a balance between speed and robustness for the algorithm. The approximate-Newton startup is found to be very robust, which makes it ideal for algorithm startup in the region where the Jacobian-free method is not stable. The objective is to switch to the Jacobian-free method early enough to benefit from the increased convergence rate, but late enough so that the solution is in the Newton convergence region.

Tests were conducted on grids up to 1,000,000 nodes and flow conditions from $M = 0.3$, $\alpha = 8°$ to $M = 0.84$, $\alpha = 3.06°$. The flow solver converges to machine zero in 2,400 function evaluations (10 minutes) for a 100,000 node grid and 6,300 (6.3 hours) for a one million node grid on a single processor. Lift and drag coefficients are converged in about eight minutes on a grid with 100,000 nodes, one hour on a grid with 500,000 nodes, and two hours on a grid with one million nodes. The scalability of TYPHOON is found to be good with the CPU cost proportional to the grid size to the power of 1.288.



(a) Top surface

(b) Bottom surface

Figure 5. Pressure contours on the ONERA M6 wing, $M = 0.84, \alpha = 3.06°$, grid D, 1 million nodes

Acknowledgements

Financial assistance from the Natural Sciences and Engineering Research Council (NSERC) and the University of Toronto is gratefully acknowledged.

References

[1] Oran, E., “Matchsticks, Scramjets, and Black Holes: Numerical Simulation Faces Reality,” *AIAA Journal*, Vol. 40, No. 8, August 2002, pp. 1481–1494.

[2] Venkatakrishnan, V. and Mavriplis, D., “Implicit Solvers for Unstructured Meshes,” *Journal of Computational Physics*, , No. 105, 1993, pp. 83–91.

[3] Nielsen, E., Anderson, K., Walters, R., and Keyes, D., “Application of Newton-Krylov Methodology to a Three-Dimensional Unstructured Euler Code,” AIAA Paper 95-1733, 1995.

[4] Luo, H., Baum, J., and Löhner, R., “A Fast, Matrix-Free Implicit Method for Compressible Flows on Unstructured Grids,” *Journal of Computational Physics*, Vol. 146, 1998, pp. 664–690.

[5] Knoll, D. and Keyes, D., “Jacobian-Free Newton-Krylov Methods: A Survey of Approaches and Applications,” *Journal of Computational Physics*, Vol. 193, 2004, pp. 357–397.

[6] Pueyo, A. and Zingg, D. W., “Efficient Newton-Krylov Solver for Aerodynamic Computations,” *AIAA Journal*, Vol. 36, No. 11, 1998, pp. 1991–1997.

[7] Blanco, M. and Zingg, D. W., “Fast Newton–Krylov Method for Unstructured Grids,” *AIAA Journal*, Vol. 36, No. 4, 1998, pp. 607–612.

[8] Nemec, M. and Zingg, D. W., “Newton-Krylov Algorithm for Aerodynamic Design Using the Navier-Stokes Equations,” *AIAA Journal*, Vol. 36, No. 1, 2002, pp. 1146–1154.

[9] Chisholm, T. and Zingg, D. W., “A Fully Coupled Newton-Krylov Solver for Turbulent Aerodynamic Flows,” Paper 333, ICAS 2002, Toronto, ON, Sept. 2002.

[10] Nichols, J., *A Three-Dimensional Multi-Block Newton-Krylov Flow Solver for the Euler Equations*, Master’s thesis, University of Toronto, 2004.

[11] Pulliam, T., “Efficient Solution Methods for the Navier-Stokes Equations,” Lecture Notes for the Von Karman Institute for Fluid Dynamics Lecture Series: Numerical Techniques for Viscous Flow Computation in Turbomachinery Bladings, NASA Ames Research Center, January 1986.

[12] Pulliam, T. and Steger, J., “Implicit Finite-Difference Simulations of Three-Dimensional Compressible Flow,” *AIAA Journal*, Vol. 18, No. 2, 1980, pp. 159–167.

[13] De Rango, S., *Higher-Order Spatial Discretization for Turbulent Aerodynamic Flows*, Ph.D. thesis, University of Toronto, 2001.

[14] De Rango, S. and Zingg, D. W., “Higher-Order Spatial Discretization for Turbulent Aerodynamic Computations,” *AIAA Journal*, Vol. 39, No. 7, 2001, pp. 1296–1304.

[15] Epstein, B., Rubin, T., and Seror, S., “Accurate Multiblock Navier-Stokes Solver for Complex Aerodynamic Configurations,” *AIAA Journal*, Vol. 41, No. 4, 2003, pp. 582–594.

[16] Kim, J. and Lee, D., “Characteristic Interface Conditions for Multiblock High-Order Computation on Singular Structured Grid,” *AIAA Journal*, Vol. 41, No. 12, 2003, pp. 2341–2348.

[17] Jameson, A., Schmidt, W., and Turkel, E., “Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping,” AIAA Paper 81–1259, June 1981.

[18] Morini, B., “Convergence Behaviour of Inexact Newton Methods,” *American Math Society*, Vol. 68, No. 228, March 1999, pp. 1605–1613.

[19] Adami, P., Michelassi, V., and Martelli, F., “Performances of a Newton-Krylov Scheme against Implicit and Multigrid Solvers for Inviscid Flows,” AIAA Paper 98-2429, Albuquerque, NM, June 1998.

[20] McHugh, P. and Knoll, D., “Comparison of Standard and Matrix-Free Implementations of Several Newton-Krylov Solvers,” *AIAA Journal*, Vol. 32, No. 12, 1994, pp. 2394–2400.

[21] Nemec, M., *Optimal Shape Design of Aerodynamic Configurations: A Newton-Krylov Approach*, Ph.D. thesis, University of Toronto, 2003.

[22] Orkwis, P., “Comparison of Newton’s and Quasi-Newton’s Method Solvers for the Navier-Stokes Equations,” *AIAA Journal*, Vol. 31, No. 5, May 1993, pp. 832–836.

[23] Pueyo, A., *An Efficient Newton-Krylov Method for the Euler and Navier-Stokes Equations*, Ph.D. thesis, University of Toronto, 1998.

[24] Saad, Y. and Schultz, M. H., “GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Problems,” *SIAM J. Sci. Stat. Comp.*, Vol. 7, 1986, pp. 856–869.

[25] 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*, SIAM, Philadelphia, PA, 1994.

[26] Feng, D. and Pulliam, T., “Tensor-GMRES Method for Large Systems of Nonlinear Equations,” *SIAM Journal on Optimization*, Vol. 7, No. 3, 1997, pp. 757–779.

[27] Pulliam, T., Rogers, S., and Barth, T., “Practical Aspects of Krylov-Subspace Iterative Methods in CFD,” AGARD CFD Symposium, October 1995.

[28] Pueyo, A. and Zingg, D. W., “Improvements to a Newton-Krylov Solver for Aerodynamic Flows,” AIAA Paper 98-0619, 1998.

[29] Cuthill, E. and McKee, J., “Reducing the Bandwidth of Sparse Symmetric Matrices,” *24th National Conference of the Association for Computing Machinery*, No. ACM P-69, Brandon Press, New York, 1969, pp. 157–172.

[30] Chisholm, T. and Zingg, D. W., “A Newton-Krylov Algorithm for Turbulent Aerodynamic Flows,” AIAA Paper 2003–0071, Reno, NV.

[31] Chisholm, T. and Zingg, D. W., “Start-up Issues in a Newton-Krylov Algorithm for Turbulent Aerodynamic Flows,” AIAA Paper 2003–3708, Orlando, FL, June 2003.

[32] Manzano, L., Lassaline, J., Wong, P., and Zingg, D. W., “A Newton-Krylov Algorithm for the Euler Equations Using Unstructured Grids,” AIAA Paper 2003-0274, 2003.

[33] Lassaline, J., *A Navier-Stokes Equation Solver using Agglomerated Multigrid Featuring Directional Coarsening and Line-Implicit Smoothing*, Ph.D. thesis, University of Toronto, 2003.

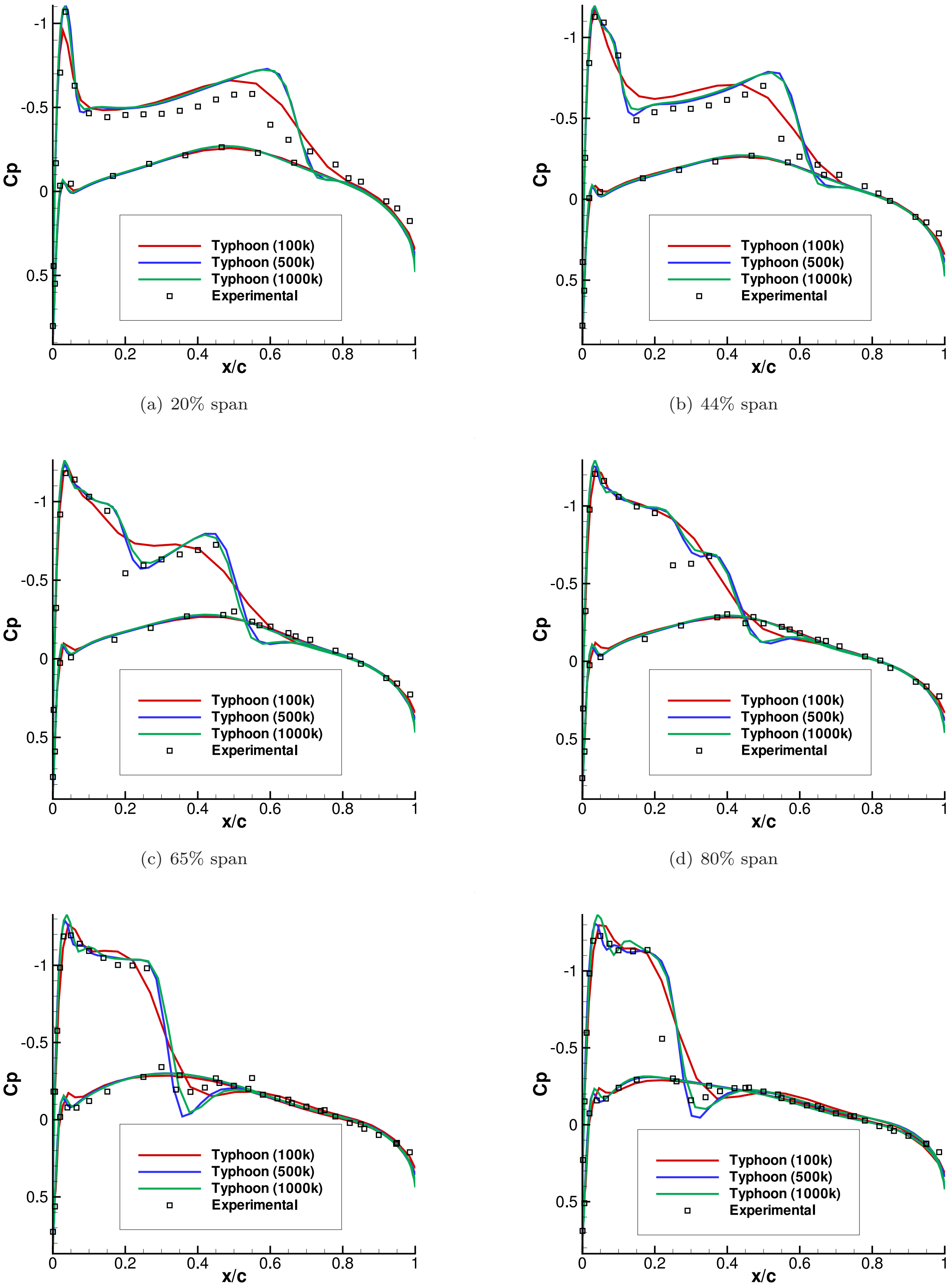
[34] Leung, T., *Parallel Implementation of a Newton-Krylov Flow Solver on Unstructured Grids*, Master’s thesis, University of Toronto, 2004.

[35] Mulder, W. and van Leer, B., “Experiments with an Implicit Upwind Method for the Euler Equations,” *Journal of Computational Physics*, Vol. 59, 1985, pp. 232–246.

[36] Hall, D., *Three-Dimensional Elliptic Grid Generation*, Master’s thesis, University of Toronto, 1992.

[37] Hall, D. and Zingg, D., *User’s Manual for the GEM3D Elliptic Grid Generator*, University of Toronto, April 1992.

[38] Schmitt, V. and Charpin, F., “Pressure Distributions on the ONERA-M6 Wing at Transonic Mach Numbers,” Paper, Office National D’Etudes et de Recherches Aerospatiales, 1979.



(a) 20% span

(b) 44% span

(c) 65% span

(d) 80% span

(e) 90% span

(f) 96% span

Figure 6. ONERA M6 wing C_p distribution, $M = 0.84, \alpha = 3.06°$