93/150R
Submitted as a Reprint

were quasisteady. For values larger than 0.009, they observed that the roll-up time, growth, and release of the periodic vortical structures were strongly dependent on the oscillation frequency. This is consistent with the present results.


## Acknowledgment

This work was supported by the Air Force Office of Scientific Research under Contract F49620-86-0133, monitored by H. Helin.


## References


[1]Francis, M. S., Keesee, J. E., Lang, J. D., Sparks, G. W., Jr., and Sisson, G. E., "Aerodynamic Characteristics of an Unsteady Separated Flow," *AIAA Journal*, Vol. 17, No. 12, 1979, pp. 1332–1339.

[2]Ramiz, M. A., and Acharya, M., "The Detection of Flow State in an Unsteady Separated Flow," *AIAA Journal*, Vol. 30, No. 1, 1992, pp. 117–123.

[3]Ramiz, M. A., "The Development of a Simple, Non-Intrusive Technique for Flow-State Detection in a Model, Leading-Edge Unsteady Separation," M.S. Thesis, Mechanical and Aerospace Engineering Dept., Illinois Inst. of Technology, Chicago, IL, May 1989.

[4]Reynolds, W. C., and Carr, L. W., "Review of Unsteady, Driven, Separated Flows," AIAA Paper 85-0527, March 1985.

[5]Koga, D. J., "Control of Separated Flowfields Using Forced Unsteadiness," Ph.D. Dissertation, Mechanical and Aerospace Engineering Dept., Illinois Inst. of Technology, Chicago, IL, 1983.

[6]Miau, J., Chen, M., and Chou, J., "Frequency Effect of an Oscillating Plate Immersed in a Turbulent Boundary Layer," AIAA Paper 89-1016, March 1989.


# Efficient Iterative Methods for the Transonic Small Disturbance Equation


A. S. Lyrintzis,* A. M. Wissink,† and A. T. Chronopoulos‡
*University of Minnesota, Minneapolis, Minnesota 55455*


## Nomenclature

| | |
|---|---|
| $A$ | = term defined in Eq. (3) |
| $b$ | = right-hand-side vector |
| $C$ | = coefficient matrix of system of equations |
| $c_1, c_2$ | = constants defined in Eqs. (6) |
| $D_x$ | = operator defined in Eqs. (6) |
| $F, F'$ | = functions used in Newton's method, Eq. (7) |
| $f$ | = intermediate value for the velocity potential |
| $g$ | = airfoil surface shape function |
| $M_\infty$ | = freestream Mach number |
| $u, \hat{u}, \bar{u}, \tilde{u}$ | = velocities defined in Eqs. (6) |
| $\gamma$ | = ratio of specific heats |
| $\delta, \Delta$ | = difference operators defined in Eqs. (6) |
| $\epsilon$ | = switching flag |
| $\tau$ | = airfoil thickness |
| $\Phi$ | = disturbance velocity potential |

*Subscripts*

| | |
|---|---|
| $i, j$ | = $x$ and $y$ coordinates of solution mesh |
| $x, y$ | = direction for difference operators $d$ and $D$ |

*Superscript*

| | |
|---|---|
| $n$ | = iteration number |

Received Nov. 15, 1991; revision received Feb. 13, 1992; accepted for publication Feb. 28, 1992. 


*Assistant Professor, Department of Aerospace Engineering and Mechanics. Member AIAA.

†Graduate Research Assistant, Department of Aerospace Engineering and Mechanics. Student Member AIAA.

‡Assistant Professor, Department of Computer Science.


## Introduction

THE steady two-dimensional transonic small disturbance (TSD) equation is solved using a new method. Approximate factorization techniques[1,2] traditionally have been used for the solution of this equation. Other methods have used finite difference techniques to discretize the nonlinear TSD equation, approximate factorizations to achieve linearization, and alternating direction implicit (ADI) techniques to solve the resulting linear tridiagonal systems.

In this Note we present a different method of solving the TSD equation. After the finite difference discretization, we use a Newton method to solve the resulting nonlinear system of equations. Conjugate gradient methods can be used for the solution of the linear system of equations in each time step. These methods are fully vectorizable and very efficient. In this work we used a preconditioned algorithm called Orthomin.[3] An efficient parallel and vector implementation of Orthomin appears in Ref. 4. The preconditioning method used incomplete LU decomposition (ILU), is obtained from the incomplete factorization of the matrix, proposed by Meijerink and van der Vorst.[5] We use a vectorizable version of ILU.[4] The idea of using Newton's method in transonic flow is not new. Hafez and Palaniswamy[6] used a direct system solver for each Newton step. The method presented here (i. e., Orthomin with ILU preconditioner) is much more efficient.

The two algorithms (ADI and Newton-Orthomin) were implemented for different test cases. We looked at subsonic and transonic flowfields for a parabolic and a NACA 64A006 airfoil for different mesh sizes. Both algorithms gave the same accuracy, but the iterative method yields better efficiency.

## Theoretical Development

The TSD equation for two-dimensional steady flow can be written as

$$\left[\frac{1-M_\infty^2}{\tau^{2/3}} - (\gamma+1)M_\infty^2\Phi_x\right]\Phi_{xx} - \Phi_{yy} = 0 \tag{1}$$

The far-field boundary conditions applied to the problem are $\Phi = 0$ along the left, right, and top edges of the mesh. The flow tangency condition is applied along the surface of the airfoil. Thus, the conditions applied along the bottom of the mesh ($y = 0$) are

$$\frac{\partial\Phi}{\partial y} = \frac{\partial g}{\partial x} \text{ along airfoil,} \quad \Phi = 0 \text{ elsewhere} \tag{2}$$

Equation (1) is discretized with finite differences over a solution mesh. The mesh used is normalized with the chord length of the airfoil, extending five chord lengths in both the $x$ and $y$ directions. A nonuniform ($80 \times 30$) mesh is used for most tests, but a finer ($160 \times 60$) mesh is also tested. Only half of the physical region of the airfoil is needed since only symmetric airfoils are studied.

For notation purposes, we define the parameter

$$A = \left[\frac{1-M_\infty^2}{\tau^{2/3}} - (\gamma+1)M_\infty^2\Phi_x\right] \tag{3}$$

The value of $A$ is important because it governs the switching procedure from subcritical to supercritical flow. When $A$ is positive, Eq. (1) is elliptic (subsonic flow) and the equation can be solved using Eq. (1) and central differences for $\Phi_{xx}$ and $\Phi_{yy}$. However, when $A$ is negative, Eq. (1) is hyperbolic (supersonic flow) and must be solved using backward differences. Murman's switch (approximate factorization two, AF2 code) was used initially with good results.[7] Here we present a more advanced monotone switch that was introduced by Goorjian et al.[2] (monotone approximate factorization, MAF code) and is based on the ideas of Godanov. (The algorithm was slightly restructured to emulate the AF2 algorithm.) Since the TSD equation is nonlinear, the resulting system of equations is nonlinear. Approximate factorization is used for the linearization. The resulting two steps are

$x$ sweep:

$$(\alpha - A_{i,j}D_x)f^n_{i,j} = \alpha(A_{i,j}D_x\delta_x + \delta_{yy})\Phi^n_{i,j} \tag{4}$$

$yx$ sweep:

$$(\alpha\delta_x - \delta_{yy})\Phi^{n+1}_{i,j} = f^n_{i,j} \tag{5}$$

where

$$A_{i,j}D_x = \{[(1-\epsilon_{i,j})c_1 + c_2(\bar{u}^n_{i+\frac{1}{2},j} + \hat{u}^n_{i-\frac{1}{2},j}) \times (\tfrac{1}{2}c_1 + c_2\bar{u}^n_{i-\frac{1}{2},j})]\}\Delta_x + \{[\epsilon_{i-1,j}c_1 + c_2(\bar{u}^n_{i-\frac{1}{2},j} + \hat{u}^n_{i-3/2,j})(\tfrac{1}{2}c_1 + c_2\hat{u}^n_{i-\frac{1}{2},j})]\}\Delta_x$$

$$c_1 = 1 - M^2_\infty, \quad c_2 = -\tfrac{1}{2}(\gamma+1)M^2_\infty, \quad \bar{u} = -\frac{c_1}{2c_2}$$

$$\bar{u}^n_{i-\frac{1}{2},j} = \bar{u} + [(1-\epsilon_{i-\frac{1}{2},j})](\bar{u}^n_{i-\frac{1}{2},j} - \bar{u})$$

$$\hat{u}^n_{i-\frac{1}{2},j} = \bar{u} + \epsilon_{i-\frac{1}{2},j}(u^n_{i-\frac{1}{2},j} - \bar{u})$$

$$u^n_{i-\frac{1}{2},j} = \delta_x\Phi^n_{i,j} = \frac{\Phi^n_{i,j} - \Phi^n_{i-1,j}}{x_i - x_{i-1}}$$

$$\epsilon_{i,j} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \text{ for } u^n_{i-\frac{1}{2},j} + u^n_{i+\frac{1}{2},j} \begin{bmatrix} \geq 2\bar{u} \\ < 2\bar{u} \end{bmatrix}$$

$$\epsilon_{i+\frac{1}{2},j} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \text{ for } u^n_{i+\frac{1}{2},j} \begin{bmatrix} \geq \bar{u} \\ < \bar{u} \end{bmatrix}$$

$$\Delta_x f^n_{i,j} = \frac{f^n_{i+1,j} - f^n_{i,j}}{\tfrac{1}{2}(x_{i+1} - x_{i-1})}, \quad \Delta_x f^n_{i,j} = \frac{f^n_{i,j} - f^n_{i-1,j}}{\tfrac{1}{2}(x_{i+1} - x_{i-1})}$$

$$\delta_{yy}\Phi_{i,j} = \frac{[(\Phi_{i+1,j} - \Phi_{i,j})/\Delta y_j] - [(\Phi_{i,j} - \Phi_{i-1,j})/\Delta y_{j-1}]}{\tfrac{1}{2}(y_{j+1} - y_{j-1})} \tag{6}$$

Both sweeps involve the solution of tridiagonal systems of equations that are solved using the Thomas algorithm.

In the Newton-Orthomin algorithm, the resulting nonlinear system of equations is solved directly using Newton's method. The method can be written as

$$F'(\Phi^n)\Delta\Phi^{n+1} = -F(\Phi^n) \tag{7}$$

where

$$F(\Phi) = (A_{i,j}D_x\delta_x + \delta_{yy})\Phi^n_{i,j} \tag{8}$$

**Table 1 CPU time comparison for Newton-Orthomin and MAF methods**

| | Newton-Orthomin | MAF | Speedup |
|---|---|---|---|
| 12% parabolic airfoil, $80 \times 30$ mesh, average residual $= 10^{-9}$ | | | |
| $M_\infty = 0.7141$ (subsonic) | 14.6 | 47.9 | 3.3 |
| $M_\infty = 0.8016$ (transonic) | 31.4 | 131.5 | 4.2 |
| NACA 64A006, $80 \times 30$ mesh, average residual $= 10^{-9}$ | | | |
| $M_\infty = 0.8004$ (subsonic) | 19.6 | 36.5 | 2.7 |
| $M_\infty = 0.8663$ (transonic) | 52.3 | 139.0 | 3.8 |
| 12% parabolic, $M = 0.8016$ (transonic) | | | |
| $(80 \times 30)$, residual $= 10^{-3}$ | 11.4 | 23.4 | 2.1 |
| $(160 \times 60)$, residual $= 4 \times 10^{-3}$ | 27.0 | 105.5 | 3.9 |
| Ratio: $(160 \times 60)/(80 \times 30)$ | 2.4 | 4.5 | |

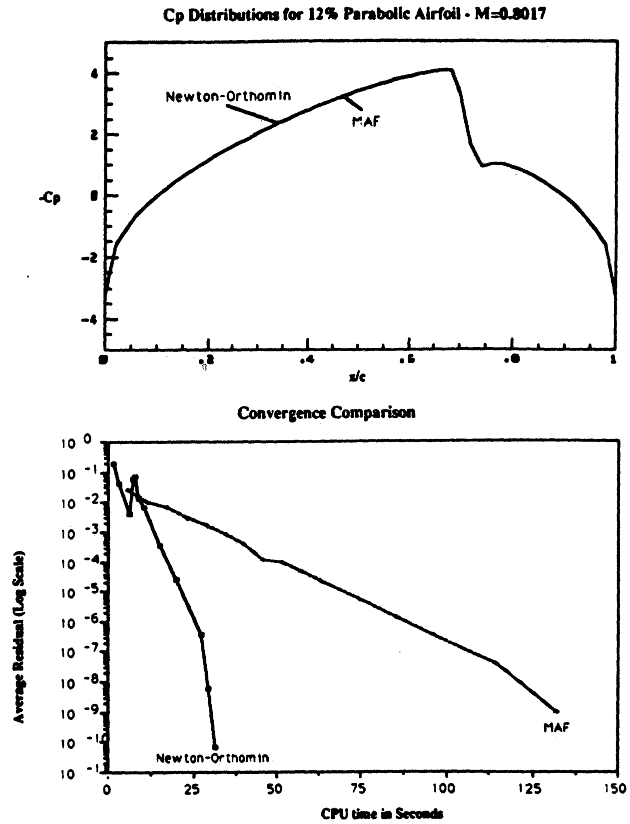


**Fig. 1 $C_p$ distribution (same in both methods) and convergence comparison for Newton-Orthomin and MAF methods; 12% thick parabolic airfoil, $M = 0.8016$.**

and where $\Delta\Phi^{n+1} = \Phi^{n+1} - \Phi^n$, and $F'$ is the Jacobian matrix; $F'$ can be found analytically by differentiating $F$ with respect to $\Phi$. Equation (7) is a linear system of equations that needs to be solved in each time step. The system is set up in the form $Cx = b$ with $C$ the matrix of the system coefficients; $C$ is sparse and has at most six diagonals ($\Phi_{i,j+1}$, $\Phi_{i,j-1}$, $\Phi_{i+1,j}$, $\Phi_{i-1,j}$, $\Phi_{i-2,j}$, $\Phi_{i,j}$). The vector $b$ is set up to satisfy the boundary conditions. Newton's method is incorporated for the nonlinear iterative process and the Orthomin solver is used for the efficient solution of this system. A description of this restructured Orthomin algorithm can be found in Ref. 4. ILU factorization preconditioning is added to the Orthomin method to speed up its convergence. For our test cases the number of iterations in Orthomin were reduced about five times.

The value of the average residual is used as the stopping criterion. The residual is found by substituting the solution into Eq. (8). The iterative process is exited when the average residual is found to be below a desired accuracy level. A different accuracy level can be used for the convergence of the linear system. The two accuracy levels are tuned for efficiency. The Newton-Orthomin algorithm can be summarized as follows:

1) Evaluate $F$ [from Eq. (7)] and $F'$ for a given $\Phi^n$.

2) Substitute in Eq. (6) and solve the linear system using Orthomin to obtain $\Delta\Phi^{n+1}$.

3) Repeat steps 1 and 2 until convergence.

## Results and Discussion

To test efficiency, both algorithms were timed for different test cases with equal exit accuracy levels. The two codes generated exactly the same solution in each test. Mach numbers are chosen for fully subsonic flow and transonic flow. The two airfoils tested are a 12% thick parabolic airfoil and a NACA 64A006 airfoil. The codes were timed on a Cray XMP, which is a four-processor vector machine with a 9.5-ns clock period. Iterations are terminated when the average residual is found to be less than $10^9$. (In both algorithms we used identical methods for calculating the average residual value.) In MAF, the size of the acceleration parameter sequence is adjusted to give best convergence. In the Newton-Orthomin algorithm, the number of orthogonal vectors used within Orthomin ($k$) and the Orthomin accuracy level are also tuned for optimal convergence (more details can be found in Ref. 8). All of the results for the measured CPU times are summarized in Table 1.

The first test is a parabolic airfoil with an $80 \times 30$ mesh. From Table 1 we can see that the Newton-Orthomin algorithm is 3.3 times as fast for subsonic flow ($M_\infty = 0.7141$) and is 4.2 times as fast for transonic flow ($M_\infty = 0.8016$). In the transonic cases we have higher speedups as the convergence of MAF slows down considerably, whereas the Newton-Orthomin algorithm slows down much less. The solution for the transonic case (identical for both algorithms) and the convergence of the two algorithms is shown in Fig. 1.

The second test is a NACA 64A006 airfoil for the same conditions. Table 1 shows the results of CPU time comparisons between the Newton-Orthomin algorithm and MAF in subsonic ($M_\infty = 0.8004$) and transonic ($M_\infty = 0.8663$) conditions. The results are nearly the same as those found for the parabolic airfoil. The Newton-Orthomin algorithm is 2.7 times as fast as MAF in subsonic flows and 3.8 times as fast in transonic flows. The convergence history (not shown) is very similar to the one shown in Fig. 1 for the parabolic airfoil.

The effect of increasing the size of the system is investigated by using a finer $160 \times 60$ mesh. This quadruples the number of equations in the system. The average residual value should be adjusted according to the mesh size (the double mesh has four times larger residual). The average residual for exit was $10^{-3}$ for the $80 \times 30$ mesh and $4 \times 10^{-3}$ for the $160 \times 60$ mesh. The results of the test using a 12% thick parabolic airfoil for transonic flow ($M_\infty = 0.8016$) are given in Table 1. We see that, as the size of the system is increased, the ratio of CPU times required for the larger mesh is much less for the Newton-Orthomin algorithm. Speedups increase from 2.1 for the regular mesh to 3.9 for the fine mesh. Similar results were found for different Mach numbers and average residuals. We expect the speedup to increase further for very large systems, as the ones used in three-dimensional calculations.

## Concluding Remarks

A new efficient algorithm is introduced for the solution of the two-dimensional TSD equation. The new algorithm uses Newton's method to solve the nonlinear system of equations resulting from the discretization using finite differences. An efficient iterative linear solver (i.e., Orthomin) is used for the solution of the sparse linear system of equations in each Newton step. The proposed algorithm is compared with a traditionally used approximate factorization algorithm with monotone switches (MAF). The results show 2.1 to 4.5 speedups for various cases and mesh sizes. These speedups are expected to be higher in very large systems. The results justify the viability of our algorithm. The algorithm idea can be extended for different switches, more complex flow models (i.e., Euler and Navier-Stokes equations), and configurations (i.e., three-dimensional flow). Other iterative linear solvers and different preconditioners should be tried to increase efficiency and demonstrate the robustness of this new approach.

## Acknowledgments

This work was supported by the Minnesota Supercomputer Institute and the Army High Performance Computing Research Center (AHPCRC) at the University of Minnesota.

## References

[1]Ballhaus, W. F., Jameson, A., and Albert, J., "Implicit Approximate Factorization Schemes for Steady Transonic Flow Problems," *AIAA Journal*, Vol. 16, No. 6, 1978, pp. 573–579.

[2]Goorjian, P. M., Meagher, M. E., and Van Buskirk, R. D., "Monotone Switches in Implicit Algorithms for Potential Equations Applied to Transonic Flows," *AIAA Journal*, Vol. 23, No. 4, 1985, pp. 492–498.

[3]Vinsome, P. K. W., "Orthomin, An Iterative Method for Solvin Sparse Sets of Simultaneous Linear Equations," Society of Petroleur Engineers of AIME, Paper SPE 5729, Feb. 1976.

[4]Ma, S., and Chronopoulos, A. T., "Implementation of Iterativ Methods for Large Sparse Nonsymmetric Linear Systems on a Paralle Vector Machine," *International Journal of Supercomputing Applica tions*, Vol. 4, No. 4, Winter 1990, pp. 9–24.

[5]Meijerink, J. A., and van der Vorst, H. A., "An Iterative Solutio Method for Linear Systems of Which the Coefficient Matrix is Symmetric *M*-Matrix," *Mathematics of Computation*, Vol. 31, Nc 137, 1977, pp. 148–162.

[6]Hafez, M., and Palaniswamy, S., "Calculations of Transoni Flows with Shocks Using Newton's Method and a Direct Solver, Par I: Potential Flows." *Advances in Computer Methods for Partic Differential Equations*, VI, edited by R. Vichnevesky an R. Stepleman, IMACS, 1987, pp. 507–516.

[7]Lyrintzis, A. S., Wissink, A., and Chronopoulos, A. T., "The Us of Efficient Iterative Methods for Solving the Transonic Small Distur bance Equation," Univ. of Minnesota Supercomputer Inst. Researc Rept. UMSI 91/288, Minneapolis, MN, Nov. 1991.

[8]Wissink, A. M., "Efficient Transonic Flow Calculations," Minne sota Supercomputer Inst., Final Project Rept., Minneapolis, MN Aug. 1991.

# Approximate Riemann Solver for Hypervelocity Flows

P. A. Jacobs*
*NASA Langley Research Center,*
*Hampton, Virginia 23665*

## Nomenclature

| | |
|---|---|
| $a$ | = local speed of sound, m/s |
| $E$ | = total energy, J/kg |
| $e$ | = specific internal energy, J/kg |
| $h$ | = specific enthalpy, J/kg |
| $M$ | = Mach number |
| $P$ | = pressure, Pa |
| $Pr$ | = Prandtl number, $(C_p\mu/k)$ |
| $R$ | = gas constant, J/kg/K |
| $T$ | = temperature, K |
| $t$ | = time, s |
| $\bar{U}$ | = Riemann invariant |
| $u$ | = $x$ component of velocity, m/s |
| $v$ | = $y$ component of velocity, m/s |
| $ws$ | = wave speed, m/s |
| $x$ | = $x$ (axial) coordinate, m |
| $y$ | = $y$ (radial) coordinate, m |
| $Z$ | = intermediate variable |
| $\rho$ | = density, kg/m$^3$ |
| $\gamma$ | = ratio of specific heats |
| $\mu$ | = coefficient of viscosity, Pa.s |

*Subscripts*

| | |
|---|---|
| $e$ | = boundary-layer edge condition |
| $L, R$ | = left state, right state, respectively |
| MIN | = minimum allowable value |
| $x, y$ | = Cartesian components |

*Superscript*

| | |
|---|---|
| * | = intermediate states for the Riemann solver locally tangent to the cone surface |

Received Sept. 23, 1991; revision received March 30, 1992; accepte for publication March 31, 1992. 

*Visiting Scientist, Institute for Computer Applications in Scienc and Engineering; currently Research Fellow, Department of Mechan cal Engineering, University of Queensland, St Lucia 4072, Australi