Convergence Acceleration of Viscous and Inviscid Hypersonic Flow Calculations

S. Cheung*
MCAT Institute, San Jose, California 95127
A. Cheer† and M. Hafez‡
University of California, Davis, Davis, California
and
J. Flores§
NASA Ames Research Center, Moffett Field, California 94035

The convergence of inviscid and viscous hypersonic flow calculations using a two-dimensional flux-vector-splitting code is accelerated by applying a Richardson-type overrelaxation method. Successful results are presented for various cases, and a 50% savings in computer time (convergence rate is increased by a factor of 2) is usually achieved. An analytical formula for the overrelaxation factor is derived, and the performance of this scheme is confirmed numerically. Moreover, application of this overrelaxation scheme produces a favorable preconditioning for Wynne's ε-algorithm. Both techniques have been extended to viscous three-dimensional flows and applied to accelerate the convergence of the compressible Navier-Stokes code. A savings of 40% in computer time is achieved in this case.

Introduction

TIME requirements for hypersonic flow calculations are generally long,[1] especially compared with transonic flow calculations. From the spectral analysis done by Cheer et al.[2] and the one in this paper, it is noticed that the spectral radii of the iterative matrices in hypersonic calculations are larger than that in the transonic flow calculation. This is why the rate of convergence in hypersonic calculations is generally slower. Thus, the search for more efficient numerical algorithms has assumed increasing importance.

Iterative methods are commonly used in computational fluid dynamics (CFD) calculations. A typical iterative scheme to solve a linear system, $Ax = b$, is given by

$$x_{n+1} = Gx_n + k$$

with $A = M - N$, and $G = M^{-1}N = (I - M^{-1}A)$, $k = M^{-1}b$. G is called an iterative matrix of the scheme and the vector sequence $\{x_n\}$ is said to be generated by G. If all of the eigenvalues of G are less than unity in modulus, the sequence $\{x_n\}$ converges to the limit (or the solution) of the scheme x^*, i.e., $x^* = Gx^* + k$. The rate of convergence is governed by the largest eigenvalue(s) of G.

Generally speaking, convergence acceleration is a transformation or a black box such that when the previous iterates $x_n, x_{n-1}, \ldots, x_{n-k}$ (with $k < n$) are fed in, it produces $S(x_n, x_{n-1}, \ldots, x_{n-k})$, which is a better approximation than x_{n+1} to the solution. It has been more than 20 years since Shanks derived his famous transformation[3]; since then, convergence acceleration has developed interest of its own. As usual in the development of any scientific field, the first 20 years were devoted to the basic knowledge of the subject, that is, algorithms for sequence transformations and their convergence and acceleration properties. Some researchers in Europe, such as Delahaye and Germain-Bonne,[4] have raised more fundamental theoretical questions. For example what is possible to achieve with convergence acceleration methods and what is not? What can be expected and when? In other words, what is the information needed in order to know whether a sequence or a set of sequences can be accelerated? Interested readers are referred to Ref. 5.

There are generally two main branches of this subject. One is the fundamental theory, as mentioned earlier. The other is the investigation of more practical and powerful algorithms. Of course, both have to go hand in hand. As far as the algorithms are concerned, there are three main sets of convergence acceleration methods applied in CFD calculations, namely, overrelaxation methods, polynomial extrapolation methods, and conjugate gradient methods.

In the 1970s, relaxation processes assumed a more prominent role in computational aerodynamics. This occurred when Murman and Cole (1971) succeeded in computing transonic flows on digital computers, the strength and location of the embedded shock being automatically captured in the finite difference mesh by extremely simply means. After that, relaxation techniques were applied to the two- and three-dimensional transonic Euler calculations.[6,7] Young (1974) demonstrated that symmetric successive overrelaxation (SSOR) could be very effectively used with linear potential problems.

Along an entirely different line, attempts were made to accelerate the convergence of the Euler calculation for transonic flow by extrapolation techniques such as the Aitken and Shanks transformations. Both Hafez and Cheng[8,9] and Martin and Lomax[10] reported some success with this procedure. In 1985, Hafez et al.[11] applied the power method and minimal residual method to accelerate the convergence rate of iterative solutions of the Euler equations for transonic flow calculations. In 1987, Wynn's vector ε-algorithm (a handy algorithm for Shanks transformation) was employed by Hafez et al.[12] to accelerate CFD calculations as well. More recently, Sidi[13] employed the minimal polynomial extrapolation (MPE) and reduced rank extrapolation (RRE) in CFD calculations and compared their performance with Wynn's ε-algorithm. All of these methods were applied to two- and three-dimensional

Presented as Paper 89-1875 at the AIAA 20th Fluid Dynamics, Plasma Dynamics, and Lasers Conference, Buffalo, NY, June 12–16, 1989; received Sept. 15, 1989; revision received Sept. 17, 1990; accepted for publication Nov. 27, 1990. Copyright © 1989 by the American Institute of Aeronautics and Astronautics, Inc. No copyright is asserted in the United States under Title 17, U.S. Code. The U.S. Government has a royalty-free license to exercise all rights under the copyright claimed herein for Governmental purposes. All other rights are reserved by the copyright owner.

*Research Scientist, 3933 Blue Gum Drive. Member AIAA.
†Associate Professor, Department of Mathematics.
‡Associate Professor, Department of Mechanics.
§Research Scientist. Member AIAA.

inviscid flow calculations. Eriksson and Rizzi[14] introduced a technique of using Arnoldi's methods as a tool to analyze the spectrum of the iterative matrix in the iterative scheme, from which useful information could be extracted. With the help of this spectral analysis technique, Cheer et al.[2] accelerated the convergence of two-dimensional viscous and inviscid flow calculations computed by ARC2D.[15]

Besides overrelaxation methods and polynomial extrapolation methods, conjugate gradient methods are also applied in CFD calculations. In the past few years, several attempts were made to use conjugate-gradient-like methods in CFD calculations.[16,17] However, difficulties were encountered in applying these methods to the nonsymmetric matrices that arise in CFD. In 1985, an algorithm called generalized minimal residual (GMRES), which was developed in 1983, was employed by Wigton et al.[18] Unfortunately, GMRES was not extended to viscous flow calculations.

The aim of this study is to accelerate the convergence rates of hypersonic flow calculations. In this paper, an overrelaxation method and Wynn's ε-algorithm[19] are utilized. The Wynn's ε-algorithm has been successfully applied to accelerate the convergence of inviscid and viscous flows in subsonic and transonic flow regimes. The overrelaxation method is easy to implement and no extra computer memory is needed. This is why the previously mentioned methods are chosen.

The governing equations and the numerical scheme are described first. Then a simple successive overrelaxation technique is presented and applied to accelerate the convergence of the Steger-Warming two-dimensional flux-vector-splitting code[20] applied to hypersonic flow conditions. Application of this relaxation method yields a 50% savings in computer time (convergence rate is increased by a factor of 2). Even faster convergence is achieved when the relaxation method is applied along with Wynn's ε-algorithm. Finally, a so-called N-relaxation factor is introduced and examined numerically. Application of this new formula yields even better results. This technique is extended to accelerate the convergence of the compressible Navier-Stokes (CNS) code in three dimensions.[21]

Governing Equation and Numerical Scheme

In two dimensions, the Euler equation can be written as

$$\partial tQ + \partial x\boldsymbol{F}(Q) + \partial y\boldsymbol{G}(Q) = 0 \tag{1}$$

where

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix}, \quad \boldsymbol{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(e+p) \end{bmatrix}, \quad \boldsymbol{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(e+p) \end{bmatrix}$$

ρ is the density, u and v the velocity components in the x and y directions, respectively, e the total energy, and p the pressure given by $p = (\gamma - 1)[e - \rho(u^2 + v^2)/2]$ with $\gamma = 1.4$ for ideal gases.

The Steger-Warming flux-vector-splitting approach makes use of the property that the flux vectors for the Euler equations are homogeneous of degree one with respect to the primary conservation variable Q. That is, in one dimension, we have

$$F = AQ \tag{2}$$

where $A \equiv \partial F/\partial Q$. The matrix A can be decomposed into an A^+ and an A^- such that the eigenvalues of A^+ are nonnegative and the eigenvalues of A^- are nonpositive in the following way:

$$A = X\Lambda X^{-1} = X(\Lambda^+ + \Lambda^-)X^{-1} = A^+ + A^- \tag{3}$$

where the columns of X are eigenvectors of A and Λ is a diagonal matrix of eigenvalues of A. The diagonal elements of $\Lambda^{\pm}$ are

$$\lambda^{\pm} = \frac{\lambda \pm |\lambda|}{2} \tag{4}$$

Thus, F can be written as

$$F = AQ = (A^+ + A^-)Q = F^+ + F^- \tag{5}$$

In this approach, the one-dimensional Euler equations can be approximated by

$$\partial_t Q + \delta_x^b F^+ + \delta_x^f F^- = 0 \tag{6}$$

where $\delta_x^b F^+$ and $\delta_x^f F^-$ are backward and forward difference operators in x, respectively.

This algorithm for two dimensions was implemented in generalized coordinates by Buning and Steger.[22] Euler equation (1), in generalized coordinates (ξ, η), where $\xi = \xi(x, y)$ and $\eta = \eta(x, y)$, can be written as

$$\partial_t \hat{Q} + \partial_\xi \hat{F}(\hat{Q}) + \partial_\eta \hat{G}(\hat{Q}) = 0 \tag{7}$$

where

$$\hat{Q} = Q/J, \qquad \hat{F} = (\xi_x F + \xi_y G)/J, \qquad \hat{G} = (\eta_x F + \eta_y G)/J$$

$$J = \xi_x\eta_y - \xi_y\eta_x = 1/(x\xi y_\eta - x_\eta y\xi), \qquad \xi_x = y_\eta J$$

$$\xi_y = -x_\eta J, \qquad \eta_x = -x_\xi J, \qquad \eta_y = -x_\xi J$$

To obtain the thin-layer Navier-Stokes equations (TLNS) in two dimensions, an explicit boundary-layer-type viscous term $\hat{G}_v$ is added to the right side of Eq. (7), yielding

$$\partial_t \hat{Q} + \partial_\xi \hat{F}(\hat{Q}) + \partial_\eta \hat{G}(\hat{Q}) = \partial_\eta \hat{G}_v \tag{8}$$

where

$$\hat{G}_v = J^{-1} \begin{bmatrix} 0 \\ \mu(\eta_x^2 + \eta_y^2)u_\eta + \frac{\mu}{3}(\eta_x u_\eta + \eta_y v_\eta)\eta_x \\ \mu(\eta_x^2 + \eta_y^2)v_\eta + \frac{\mu}{3}(\eta_x u_\eta + \eta_y v_\eta)\eta_y \\ \frac{\gamma\kappa}{Pr}(\eta_x^2 + \eta_y^2)e_\eta + \beta \end{bmatrix}$$

$$\beta = \mu[(\eta_x^2 + \eta_y^2)\tfrac{1}{2}(u^2 + v^2)\,\eta + (\mu/3)(\eta_x u + \eta_y v)(\eta_x u_\eta + \eta_y v_\eta)]$$

μ is the viscosity, κ the conductivity, and $Pr = 0.72$ the Prandtl number.

First-order differencing applied to this flux-vector-splitting approach yields a large sparse system of equations, which is solved by means of an approximate factorization technique.[22] One of the difficulties encountered in solving this problem in generalized coordinates is that the invariance of the metric differencing must be satisfied in order to maintain a uniform flow solution in the absence of disturbances.[23] To cancel out the metric differencing error in the solution, a simple approach of freestream subtraction is used. In this approach, the Euler equations can be rewritten in delta form as follows:

$$\begin{aligned} &[I + \Delta t(\nabla_\xi \hat{A}^+ + \nabla_\eta \hat{B}^+)^n][I + \Delta t(\Delta_\xi \hat{A}^- \\ &\quad + \Delta_\eta \hat{B}^-)^n]\Delta \hat{Q}^n = -\Delta t(\delta_\xi^b \hat{F}^+ + \delta_\xi^f \hat{F}^- + \delta_\eta^b \hat{G}^+ \\ &\quad + \delta_\eta^f \hat{G}^-)^n - \Delta t(\delta_\xi^b \hat{F}^+ + \delta_\xi^f \hat{F}^- + \delta_\eta^b \hat{G}^+ + \delta_\eta^f \hat{G}^-)_\infty \end{aligned} \tag{9}$$

where $\Delta\hat{Q}^n = \hat{Q}^{n+1} - \hat{Q}^n$ and ∞ indicates that the last term is evaluated at freestream.

Putting this into a matrix formulation yields

$$M\Delta x_n = -\Delta t A x_n + \Delta t b \tag{10}$$

or

$$x_{n+1} = G x_n + \Delta t k \tag{11}$$

where $x_n = \hat{Q}^n$, $\Delta x_n = \Delta\hat{Q}^n$, $G = (I - \Delta t M^{-1} A)$, and $k = M^{-1} b$.

The matrix M is symbolically denoted by

$$M = [I + \Delta t(\nabla_\xi \hat{A}^+ + \nabla_\eta \hat{B}^+)^n][I + \Delta t(\Delta_\eta \hat{A}^- + \Delta_\eta \hat{B}^-)^n]$$

$$A = (\delta_\xi^b \hat{A}^+ + \delta_\xi^f \hat{A}^- + \delta_\eta^b \hat{B}^+ + \delta_\eta^f \hat{B}^-)^n$$

$$b = -(\delta_\xi^b \hat{F}^+ + \delta_\xi^f \hat{F}^- + \delta_\eta^b \hat{G}^+ + \delta_\eta^f \hat{G}^-)_\infty$$

Reference 22 contains detailed information on the numerical algorithm.

The test case in this study is a hypersonic flow over a blunt cone. The inflow boundary is freestream with high Mach number. The strong bow shock extends to the outflow, and the boundary condition is zeroth-order extrapolation. A no-slip boundary condition is used at the body when viscous calculation is performed.

Richardson Relaxation and Wynn's ε-Algorithm

Various convergence-acceleration techniques have been successfully applied to subsonic and transonic flow calculations. In this section, applications of a Richardson's overrelaxation-type method (RF method) and Wynn's ε-algorithm are described. Results of these methods applied to viscous and invicid hypersonic flow calculations are presented.

In order to calculate the relaxation factor for the RF method, the largest and the smallest eigenvalues of the iterative matrix need to be determined. In this study, Arnoldi's method[24] is used to reveal a subset of the eigenvalues of the iterative matrix in the flux-vector-splitting algorithm. Figures 1 and 2 are plots of typical subsets of the eigenvalues of G. The subset contains the largest and the smallest eigenvalues.

The following is a nonrigorous analysis using the Gershgorin circle theorem[25] to demonstrate the reasoning. Consider

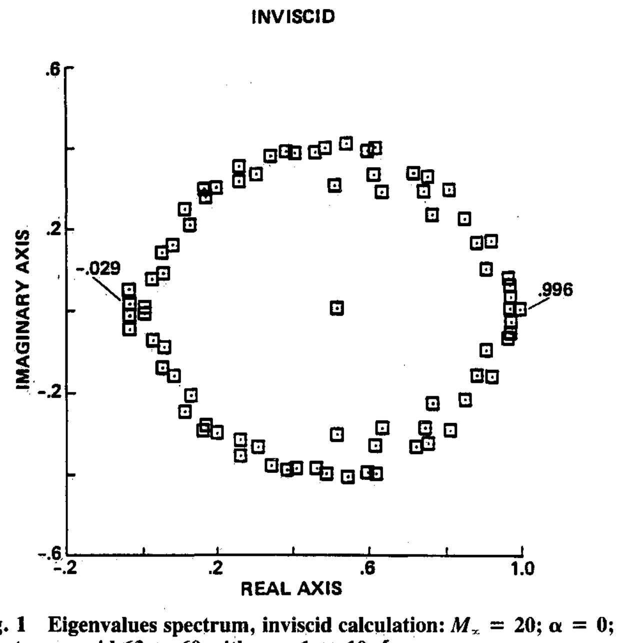

Fig. 1 Eigenvalues spectrum, inviscid calculation: $M_\infty = 20$; $\alpha = 0$; blunt cone grid 63 × 60 with $s = 1 \times 10^{-5}$.

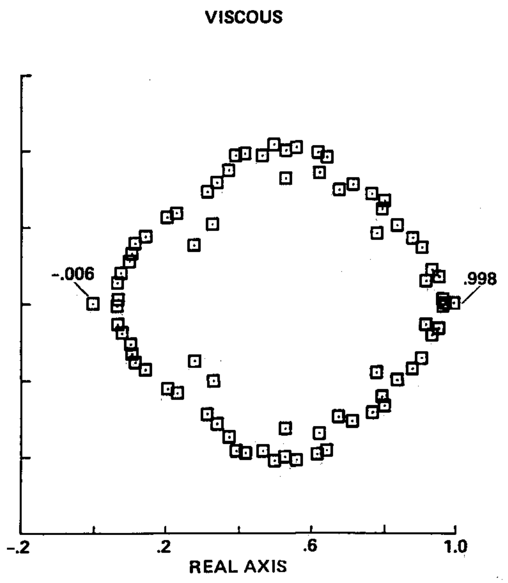

Fig. 2 Eigenvalues spectrum, viscous calculation: $M_\infty = 20$; $Re_L = 4 \times 10^6$; $\alpha = 0$; blunt cone grid 63 × 60 with $s = 1 \times 10^{-5}$.

the convergent flux-vector-splitting algorithm in one dimension. From Eqs. (10) and (11), we have a five-banded matrix symbolized as

$$M = [I + \Delta t(\nabla_\xi \hat{A}^+ + \Delta_\xi \hat{A}^-)^n] \tag{12}$$

For one-dimensional hypersonic flow, only the region near the shock value of $(\nabla_\xi \hat{A}^+ + \Delta_\xi \hat{A}^-)$ is large. For the iterative scheme of Eq. (10) to be convergent, the value of Δt should be kept small or the CFL number is too large. Let the second term of Eq. (12) be small, then a binomial expansion can be applied and yields

$$M^{-1} \approx [I - \Delta t(\nabla_\xi \hat{A}^+ + \Delta_\xi \hat{A}^-)^n] + \mathcal{O}(\Delta t^2)$$

Substitute this into the iterative matrix G, we have

$$G \approx I - \Delta t(\nabla_\xi \hat{A}^+ + \Delta_\xi \hat{A}^-)^n + \mathcal{O}(\Delta t^2) \tag{13}$$

Now, Gershgorin's circle theorem can be applied to Eq. (13), indicating that the spectrum of G is expected to have the real parts of the largest eigenvalues close to 1 and to have the imaginary parts close to the real axis.

Consider the following linear iterative scheme

$$M\Delta x_n = -\Delta t A x_n + \Delta t b \tag{14}$$

or

$$x_{n+1} = (I - \Delta t M^{-1} A) x_n + \Delta t M^{-1} b \tag{15}$$

Let ω be a relaxation factor, such that

$$x'_{n+1} = x_n + \omega \Delta x_n \tag{16}$$

where x'_n is called a relaxed solution, which will be used as an old-level solution in Eq. (14), or Eq. (15), to produce a new-level solution. This new-level solution is then put into Eq. (16) to obtain a better approximation of the solution. This combination of Eqs. (14) and (16) is the relaxation process. We are going to derive a formula for the optimal value of ω that gives the fastest rate of convergence.

Substitution of Eq. (14) into Eq. (16) yields

$$x'_{n+1} = (I - \omega \Delta t M^{-1} A) x_n + \omega M^{-1} b \tag{17}$$

Let x^* be the exact steady solution and define the error vector ε_n and ε' by $\varepsilon_n = x_n - x^*$ and $\varepsilon' = x' - x^*$, respectively. From Eqs. (15) and (17),

$$\varepsilon_n = (I - \Delta t M^{-1}A)\varepsilon_n \tag{18a}$$

$$\varepsilon' = (I - \omega\Delta t M^{-1}A)\varepsilon_n \tag{18b}$$

Let G and $G(\omega)$ be the iterative $(I - \Delta t M^{-1}A)$ matrices and $(I - \omega\Delta t M^{-1}A)$, respectively, and let λ and $\lambda(\omega)$ be the eigenvalues of G and $G(\omega)$, respectively. From Eqs. (18), and from a simple algebraic calculation (with $\omega \geq 1$),

$$\lambda(\omega) = 1 - \omega(1 - \lambda) \tag{19}$$

From Eq. (19), it is clear that, for all fixed ω, the transformation that maps λ into $\lambda(\omega)$ is linear. Thus, the largest and the smallest eigenvalues of G are mapped to the largest and the smallest eigenvalues of $G(\omega)$ with the following real and imaginary parts.

Real part:

$$\mathrm{Re}[\lambda(\omega)] = 1 - \omega + \omega\mathrm{Re}(\lambda)$$

Imaginary part:

$$\mathrm{Im}[\lambda(\omega)] = \omega\mathrm{Im}(\lambda)$$

Since the imaginary parts are small compared with the real parts in the hypersonic calculation of flux-vector-splitting algorithm, as shown by Gershgorin's circle theorem, only the real parts are considered in the following derivation.

Let $\Pi(G)$ and $\pi(G)$ be the right and the left eigenvalues of G, respectively. Note that since only the real parts are considered $\Pi(G)$ and $\pi(G)$ are real. Rewriting Eq. (19) yields

$$\Pi[G(\omega)] = 1 - \omega[1 - \Pi(G)] \tag{20a}$$

$$\pi[G(\omega)] = 1 - \omega[1 - \pi(G)] \tag{20b}$$

The relaxation factor is chosen such that the following condition is satisfied: $\Pi(G) = -\pi(G)$. This condition gives

$$\omega_R = \frac{2}{2 - \pi(G) - \Pi(G)} \tag{21}$$

In the following, ω_R will be called the Richardson relaxation factor (or RF factor). A similar formula is given in Ref. 26 for the solution of the Laplace equation.

It can be shown that, if G is independent of the iteration n, then the relaxation ω_R defined in Eq. (21) is optimal. To prove this assertion, it is necessary to show that the spectral radius of $G(\omega_R)$ is less than the spectral radius of $G(\omega)$ for all possible ω. Since we are only concerned with the real parts of the spectrum, let $|\Pi(G)|>|\pi(G)|$. It is clear from Eqs. (20) that

$$\Pi[G(\omega)] - \Pi[G(\omega_R)] = (\omega_R - \omega)[1 - \Pi(G)] \tag{22a}$$

This implies that $\Pi[G(\omega)] > \Pi[G(\omega_R)]$ if $\omega_R > \omega$; that is, ω_R is a better relaxation factor. Similarly,

$$\pi[(G(\omega)] - \pi[G(\omega_R)] = (\omega_R - \omega)[1 - \pi(G)] \tag{22b}$$

This implies that $|\pi(G(\omega)]|>|\pi[G(\omega_R)]| = \Pi[G(\omega_R)]$ if $\omega_R < \omega$; that is, ω_R is a better relaxation factor. Thus, ω_R is the optimal relaxation factor.

In order to apply the overrelaxation method with $\omega = \omega_R$, $\Pi(G)$ and $\pi(G)$ must be known. The value of $\Pi(G)$ is approximated by the ratio of the residues at any two levels, that is,

$$\Pi(G) = \kappa_n = \frac{\|\text{Residue at level } n + 1\|}{\|\text{Residue at level } n\|} \tag{23}$$

This approximation is true for linear iteration. In application, if $\kappa_n > 1$, then ω is simply set to 1, otherwise ω is computed from Eq. (21). However, there is no easy method by which to approximate $\pi(G)$. Since the smallest eigenvalues of G are close to zero in the flux-vector-splitting algorithm, $\pi(G)$ is set to be zero. The relaxation factor given by Eq. (21) can then be replaced by a simpler relaxation factor $\tilde{\omega}_R$,

$$\tilde{\omega}_R = \frac{2}{2 - \Pi(G)} \tag{24}$$

In Figs. 1 and 2, subsets of eigenvalues near the steady state (large n), obtained by using Arnoldi's method, are plotted for the Euler and TLNS iterative matrices, respectively. Notice that the distribution for the TLNS case is shifted to the right relative to that in the Euler case. The largest eigenvalues are real with magnitude of 0.996 for the Euler case and 0.998 for the TLNS case. The smallest eigenvalues for both cases are close to zero. This indicates that $\tilde{\omega}_R$ is close to ω_R. Figures 3 and 4 show that a 50% savings in number of iterations is achieved when $\tilde{\omega}_R$ is applied to the Euler and TLNS calculations, respectively. This work is undertaken with the idea in mind to apply the convergence acceleration strategy for engineering purposes. Therefore, in Figs. 3 and 4, the L2 norm of residues are only dropped three orders of magnitude from its initial value.

In addition to overrelaxation, the convergence of the iterative scheme is also accelerated using Wynn's ε-algorithm, which gives a handy way to implement the pth-order Shank transformation, which annihilates the first p largest eigenvalues of the iterative matrix. This algorithm is defined by the following formulas

$$\varepsilon_{-1}^n = 0, \qquad \varepsilon_0^n = Q^n \qquad \forall\, n \geq 0$$

$$\varepsilon_{p+1}^n = \varepsilon_{p-1}^{n+1} + (\varepsilon_{p-1}^{n+1} - \varepsilon_p^n)/\theta, \qquad \forall\, n,p \geq 0 \tag{25}$$

where

$$\theta = (\varepsilon_{p-1}^{n+1} - \varepsilon_p^n)\cdot(\varepsilon_{p-1}^{n+1} - \varepsilon_p^n)$$

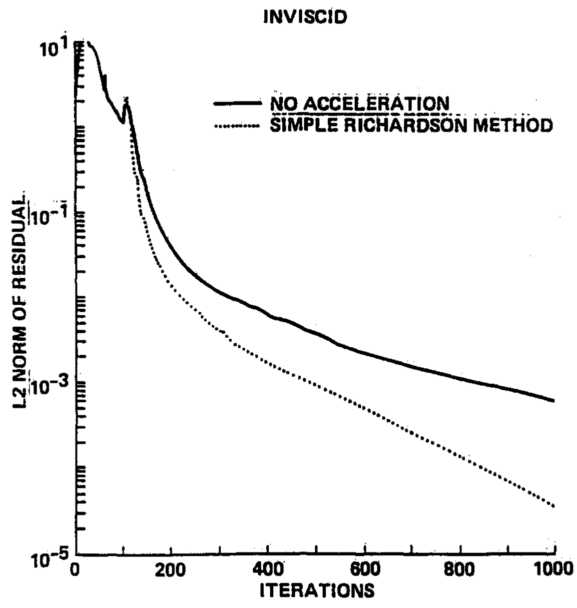

Fig. 3 History of residuals, inviscid calculation: $M_\infty = 20$; $\alpha = 0$; blunt cone grid 63 × 60 with $s = 1 \times 10^{-5}$.

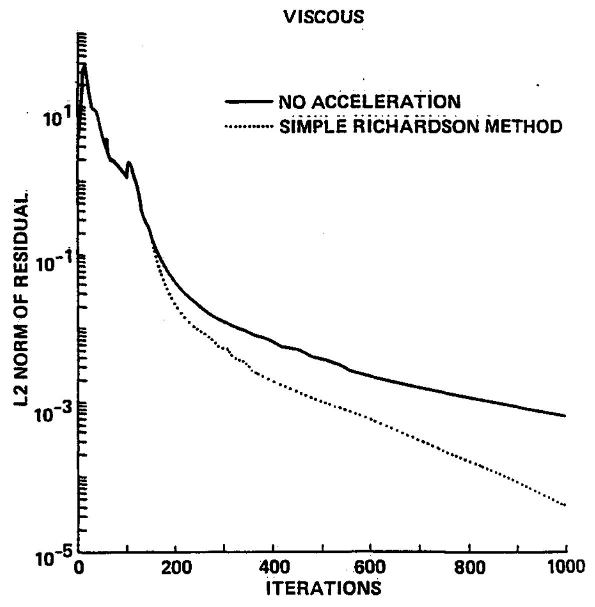

Fig. 4 History of residuals, viscous calculation: $M_\infty = 20$; $Re_L = 4 \times 10^6$; $\alpha = 0$; blunt cone grid 63 × 60 with $s = 1 \times 10^{-5}$.

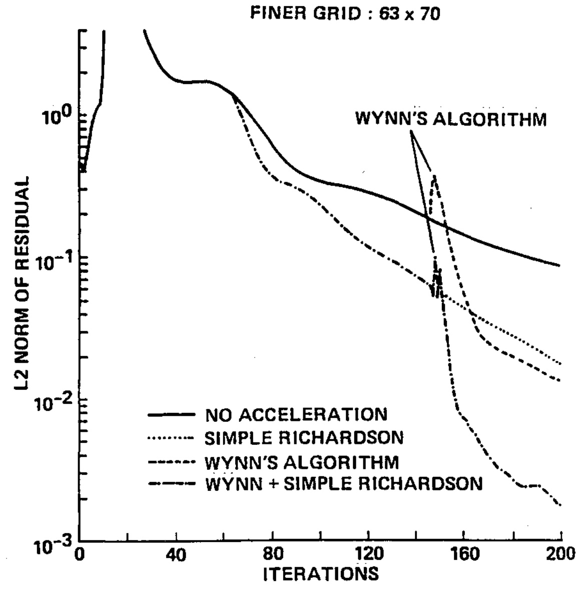

Fig. 6 History of residuals, inviscid calculation: $M_\infty = 12$; $\alpha = 0$; blunt cone grid 63 × 70 with $s = 1 \times 10^{-6}$.

Here, a set of iterative solutions $Q^1, Q^2, Q^3, \ldots, Q^n$ are stored in $\varepsilon_0^1, \varepsilon_0^2, \varepsilon_0^3, \ldots, \varepsilon_0^n$, respectively. In Eqs. (25), ε_{2p}^n is the same as the pth-order Shank transformation, which is supposed to be a better approximation to the steady-state solution. Application of Wynn's algorithm requires an extra storage of $2p$ iterative solutions. Detailed definition of the implementation of Wynn's algorithm is given in Ref. 19.

The result of applying the third-order ε-algorithm ($p = 1, \ldots, 6$) to the inviscid flow calculations with a freestream Mach number of 12 is plotted in Figs. 5 and 6. Figure 5 shows the resultant L2 norm on a grid (63 × 40) whose first grid spacing (the geometric distance between the geometry and the first grid point off the geometry, denoted by s) is 1×10^{-3}. Figure 6 presents the results on a finer grid (63 × 70) whose first grid spacing s is 1×10^{-5}. As a result of applying the Wynn's ε-algorithm to the calculations, the L2 norm of the residual drops very rapidly (see Fig. 5). If, however, the Wynn's ε-algorithm is applied earlier, wiggles develop in the residual, as shown in Fig. 6. Nevertheless, the convergence is still accelerated.

Application of the linear relaxation factor changes the iterative matrix G to $G(\bar{\omega}_R)$. Applying Wynn's ε-algorithm with and without relaxation leads to the annihilation of the three largest eigenvalues in $G(\bar{\omega}_R)$ and $G(\omega)$, respectively. The fourth largest eigenvalue(s) now dominates the rate of convergence of the two iterative schemes. Comparing the results when either relaxation (dotted line) or Wynn's (dash line) are applied alone to the results when they are applied together (dash-dotted line) in Figs. 5 and 6 indicates that the fourth largest eigenvalue(s) of $G(\bar{\omega}_R)$ is smaller than that of G.

Figures 7 and 8 are plots of the convergence history of the TLNS calculations with both the relaxation factor $\bar{\omega}_R$ and Wynn's ε-algorithm. Figure 7 corresponds to the calculations of a freestream Mach number of 19 with an angle of attack of zero ($\alpha = 0$) and Reynolds number Re of 1×10^6. For this case, the relaxation factor $\bar{\omega}_R$ is a favorable precondi-

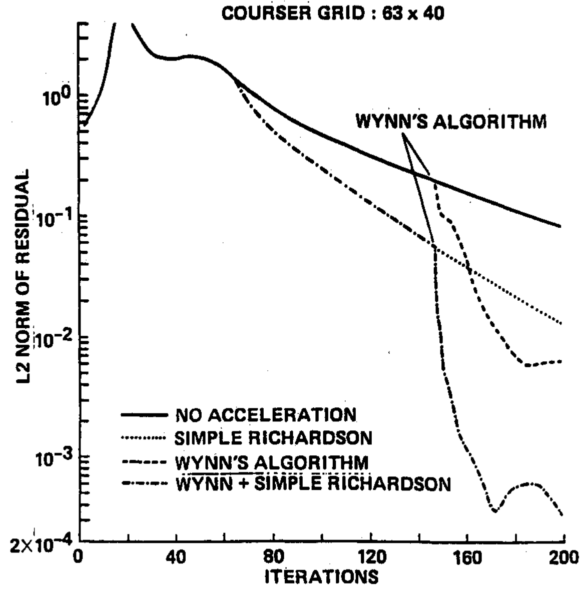

Fig. 5 History of residuals, inviscid calculation: $M_\infty = 12$; $\alpha = 0$; blunt cone grid 63 × 40 with $s = 1 \times 10^{-4}$.

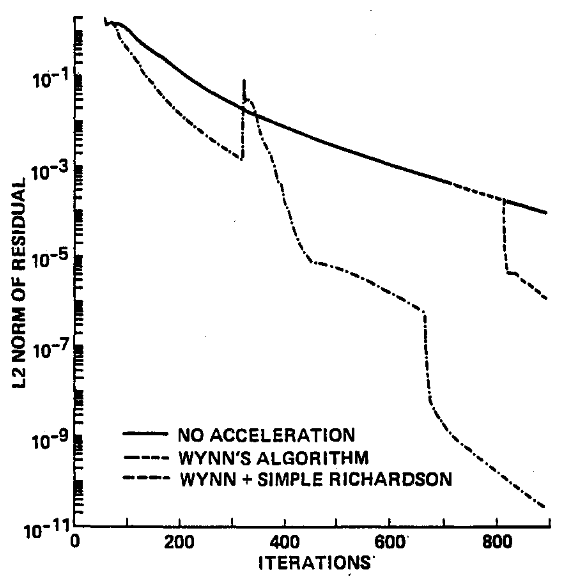

Fig. 7 History of residuals, TLNS calculation: $M_\infty = 19$; $Re_L = 1 \times 10^6$; $\alpha = 0$; blunt cone grid 63 × 60.

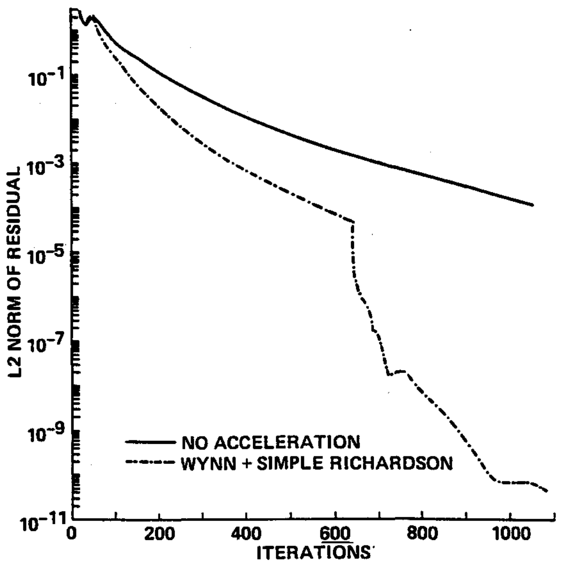

Fig. 8 History of residuals, TLNS calculation: $M_\infty = 12$; $Re_L = 1 \times 10^6$; $\alpha = 0$; blunt cone grid 63 × 60.

tioning for Wynn's ε-algorithm (compare the residual drop of the dashed line and the solid line in Fig. 7). Figure 8 corresponds to a calculation with a freestream Mach number of 12 and an angle of attack of 5 deg. It shows that combining the two methods yields a six-order-of-magnitude reduction in the residual compared to that in the case in which no acceleration technique is applied.

N-Relaxation Factor

Consider the ordinary differential equation (ODE)

$$\frac{dx}{dt} = -M^{-1}Ax - M^{-1}b \tag{26}$$

where M and A are matrices that depend on x (M is invertible), b is a constant vector, and Δt is a constant. First-order differencing in time will give

$$x_{n+1} = (I - \Delta t M^{-1}A)x_n + \Delta t M^{-1}b \tag{27}$$

where x_{n+1} is an approximation of the solution $x[t = (n + 1)\Delta t]$ of Eq. (26) and $G_n = (1 - \Delta t M^{-1}A)$ is dependent on n. Thus, Eq. (19) should be rewritten as

$$\lambda_n(\omega) = 1 - \omega_n(1 - \lambda_n) \tag{28}$$

and the optimal relaxation factor of this n-dependent iterative scheme should be like Eq. (21), except that $\Pi(G_n)$ and $\pi(G_n)$ are now n dependent. However, it is not practical to calculate the largest and the smallest eigenvalues of G_n for each n in terms of computer time and memory. Therefore, an approximate relation between λ and $\lambda(\omega)$ for fixed ω is described as follows.

Consider the fact that, when $M^{-1}A$ is a constant matrix and its eigenvalues are real, then

$$x_1 < x^* < x(\tau) \tag{29}$$

where $x(\tau)$ is the solution of the linear ODE at time τ, x_1 is the solution of the iterative scheme (27) with time step $\Delta t = \tau$ (i.e., x_1 is solution obtained from one step), and x^* is the approximation of $x(\tau)$ obtained from more than one time step (note that the sum of time steps is τ). Also notice that the solution of the linear ODE can be written as linear combination of the eigenvectors of G. Thus, Eq. (29) gives

$$\sum d_i \lambda_i v_i \le \sum d_i \sigma_i v_i \le \sum d_i e^{\lambda_i - 1} v_i$$

or for each component, it gives

$$\lambda_i \le \sigma_i \le e^{\lambda_i - 1} \tag{30}$$

where λ_i and v_i are eigenpairs of G, and σ_i is unknown. For simplicity, the subscripts will be dropped. It is noticed that, same as eigenvalues λ, σ governs the rate of convergence of the iterative scheme. We saw earlier that a relaxation process will change the iterative matrix as Eq. (15) to Eq. (17). Repeat the process for x and x_1 and obtain

$$\begin{aligned} 1 + \omega(\lambda - 1) &= \lambda(\omega) \le \sigma(\omega) \\ &\le \nu(\omega) = 1 + \omega(e^{\lambda - 1} - 1) \end{aligned} \tag{31}$$

These inequalities are obtained from the assumption that $M^{-1}A$ is a fixed constant matrix and x^* is obtained by a series of different time steps whose sum is τ. However, in actual calculation, a fixed time step is chosen by users and $M^{-1}A = (M^{-1}A)_n$, which changes for every n. Suppose for large n we can view the iterative scheme Eq. (27) as if Δt is changing from one iteration to another and $(M^{-1}A)_n$ equal to some appropriate constant matrix $M^{-1}A$, that is,

$$G_n = [I - \Delta t(M^{-1}A)_n] = (I - \Delta t_n M^{-1}A)$$

Similarly, the iterative matrix of a relaxation scheme should be

$$G_n(\omega) = [I - \omega\Delta t(M^{-1}A)_n] = (I - \omega\Delta t_n M^{-1}A)$$

Therefore, the inequalities in Eqs. (30) and (31) are still true for actual calculation.

To approximate $\sigma(\omega)$, we look for non-negative constants ϕ_1 and ϕ_2 such that

$$\sigma(\omega) = 1 + \omega(e^{\lambda - 1} - 1) - \phi_1 = 1 + \omega(\lambda - 1) + \phi_2$$

Considering Eq. (27) and taking the truncation error of the scheme (27) into account (since $x_{n+1} = x_n + \Delta t\, \dot{x}|_n + (\Delta t/2)\, \ddot{x}|_n$) will yield the relation

$$\sigma(\omega) = 1 + \omega(e^{\lambda - 1} - 1) + \frac{\omega\Delta t\kappa}{2(\lambda - 1)} \tag{32}$$

where κ is given by $\ddot{x} = \Sigma\, c\kappa\nu$; ν are eigenvectors of G. Note that the subscripts are dropped. On the other hand, considering Eq. (26) with truncation error yields

$$\sigma(\omega) = 1 + \omega(\lambda - 1) + \frac{\omega\Delta t^2\kappa}{2} \tag{33}$$

That is,

$$\phi_1 = \frac{\omega\Delta t\kappa}{2(1 - \lambda)}, \qquad \phi_2 = \frac{\omega\Delta t^2\kappa}{2}$$

The value of κ can be determined by equating Eqs. (32) and (33), so that

$$\sigma(\omega) = 1 + \omega(e^{\lambda - 1} - 1) + \omega\frac{\chi(\lambda)}{\lambda - 2} \tag{34}$$

where $\chi(\lambda) = e^{\lambda - 1} - \lambda$. The details are given in Ref. 27.

The relaxation in Eq. (34) indicates that, for $\omega > 1$ and $|\lambda|$, maximum (minimum) σ maps to maximum (minimum) $\sigma(\omega)$. In order to obtain optimal value of ω, the same hypothesis that was used in the RF method is employed, i.e., the largest value (positive) of $\sigma(\omega)$ equals the absolute value of the smallest (negative) $\sigma(\omega)$. After manipulation,

$$\omega_N = \frac{2}{2 - e^{\Pi-1} - e^{\pi-1} - [\chi(\Pi)]/(\Pi - 2) - [\chi(\pi)]/(\pi - 2)} \tag{35}$$

We will call the relaxation factor given by Eq. (35) the *N*-relaxation factor.

Application of ω_N to the TLNS calculation with a freestream Mach number of 12 yields a 65% savings in number of iterations (see Fig. 9). In comparing Figs. 3, 4, and 9, it is clear that the application of the *N*-relaxation factor results in an additional 15% savings in number of iterations. This savings in CPU time is significant since the extra computations required in evaluation of ω_N are negligible in comparison.

In Table 1, the *N*-relaxation factor ω_N is compared with the relaxation factor ω_{exp}, which is picked by trail and error with different values of ω until the fastest rate of convergence is observed. The values in Table 1 agree reasonably well.

Tables 2 and 3 contain values of the smallest and largest eigenvalues of the code when a fixed relaxation factor is applied in late iteration (near steady state). These values are obtained from Arnoldi's method and Eq. (34). Tables 2 and 3 show how well Eq. (34) predicts in these test cases. The results show that Eq. (34) is reasonably accurate in estimating the eigenvalues. Furthermore, Eq. (34) allows us to compare the rate of convergence of G and $G(\omega)$. For example, for the TLNS calculation in a grid of 60 × 63 at Mach number of 16, $\Pi(G) \approx 0.9853$. If the value of $\pi(G)$ is chosen to be -0.1016, then Eq. (35) gives $\omega_N \approx 2.250$. Using these values in Eq. (34) to obtain $\Pi[G(\omega)] \approx 0.9670$ and the ratio of the rates of convergence of G and $G(\omega)$ is

$$\frac{-\ln[\Pi(G)]}{-\ln\{\Pi[G(\omega)]\}} \approx 0.45$$

i.e., the rate of convergence is increased by a factor of 1.82. This ratio is confirmed by Fig. 10.

Figure 11 shows a comparison of the overrelaxation method and the shifting method introduced by Saleem,[28] where

$$\Sigma = \text{shifting factor} = -[\Pi(G) + \pi(G)]/4 \tag{36}$$

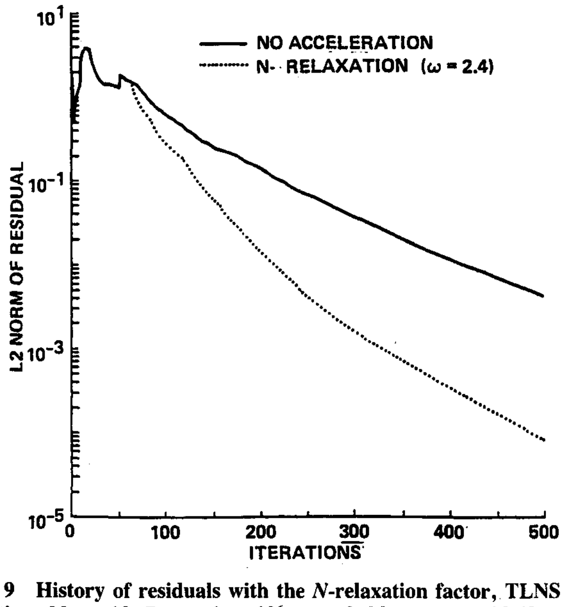

Fig. 9 History of residuals with the *N*-relaxation factor, TLNS calculation: $M_\infty = 12$; $Re_L = 1 \times 10^6$; $\alpha = 0$; blunt cone grid 63 × 60.

Table 1 Comparison of optimal relaxation factors obtained from experiments and Eq. (35)[a]

M_∞	ω_N	ω_{exp}
12	2.40	2.449
16	2.21	2.329
19	2.03	1.986

[a]Test case: Blune cone, 63 × 70 grid, $Re = 1 \times 10^6$, $\Delta t = 0.1$.

Table 2 Smallest and largest eigenvalues obtained from Arnoldi's method and Eq. (34)[a]

Eigenvalues	Arnoldi's Method	Eq. (34)
$\Pi[G(\omega)]$	0.9676 ± i0	0.9701 ± i0
$\pi[G(\omega)]$	−0.9486 ± i0	−0.9650 ± i0

[a]Test case: 63 × 60 grid, $Re = 1 \times 10^6$, $M_\infty = 12$, $\Delta t = 0.1$, $\omega = 2.35$, $\Pi(G) = 0.9873$, $\pi(G) = -0.0344$.

Table 3 Smallest and largest eigenvalues obtained from Arnoldi's method and Eq. (34)[a]

Eigenvalues	Arnoldi's Method	Eq. (34)
$\Pi[G(\omega)]$	0.9921 ± i0	0.9893 ± i0
$\pi[G(\omega)]$	−0.9375 ± i0	−0.9231 ± i0

[a]Test case: 63 × 70 grid, $Re = 1 \times 10^6$, $M_\infty = 16$, $\Delta t = 0.2$, $\omega = 2.35$, $\Pi(G) = 0.9953$, $\pi(G) = -0.1016$.

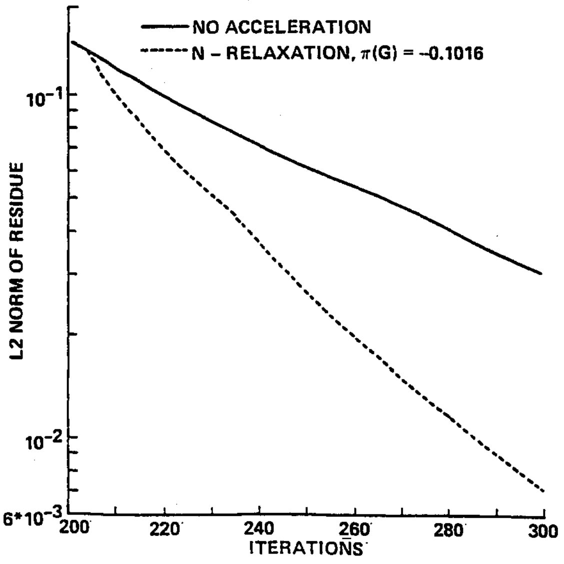

Fig. 10 History of residuals with the *N*-relaxation factor, TLNS calculation: $M_\infty = 16$; $Re_L = 1 \times 10^6$; $\alpha = 0$; blunt cone grid 63 × 60.

For the flux-vector-splitting algorithm, this shifting factor is applied to the left side of Eq. (9) as follows

$$[I + I\Sigma + \Delta t(\nabla_\xi \hat{A}^+ + \nabla_\eta \hat{B}^+)^n][I + I\Sigma + \Delta t(\Delta_\xi \hat{A}^- + \Delta_\eta \hat{B}^+)^n]\Delta \hat{Q}^n = \text{RHS [Eq. (9)]}$$

The results plotted in Fig. 11 indicate that the relaxation factor ω_N results in a faster convergence and a more favorable preconditioning for Wynn's ε-algorithm than the shifting method.

Both the *N*-relaxation method and Wynn's ε-algorithm are applied to a three-dimensional case using the CNS code.[7] The geometry of the problem is a blunt cone. The grid is (31 ×

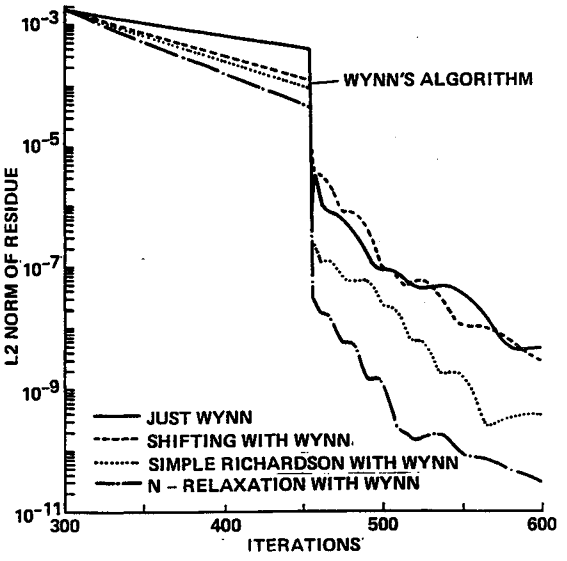

Fig. 11 Comparison of Wynn's algorithm with different preconditionings, TLNS calculation: $M_\infty = 12$; $Re_L = 1 \times 10^6$; $\alpha = 0$; blunt cone grid 63 × 34.

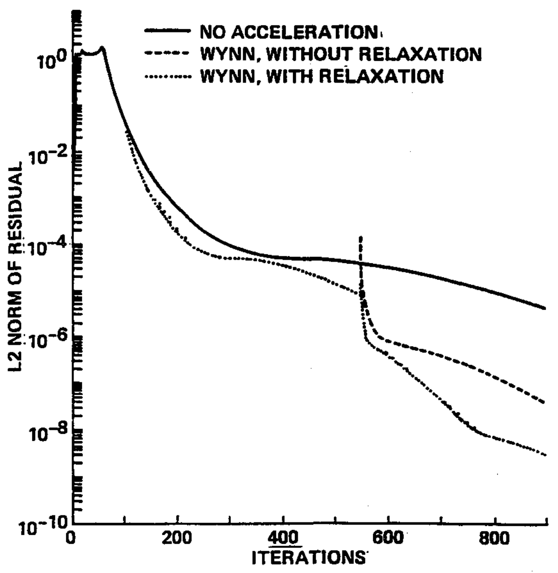

Fig. 13 Convergence of CNS code using Wynn's algorithm with and without the N-relaxation factor: $M_\infty = 15$; $Re_L = 5 \times 10^6$; $\alpha = 0$; blunt cone grid 31 × 34 × 23.

34 × 23) with a first grid spacing $s = 1 \times 10^{-4}$, and the freestream Mach number is 15. For this case, the residual drops very rapidly initially and then slows down considerably (see Figs. 12 and 13). Since the problem is three dimensional, determining the values of $\Pi(G)$ and $\pi(G)$ is expensive. The value of $\Pi(G)$ is approximated by κ_n as in Eq. (23) and, based on the eigenvalue analysis for the two-dimensional, flux-vector-splitting code (which is the only source to give an idea of how the spectrum of the iterative matrix in the flux-vector-splitting algorithm should look), the smallest eigenvalue is estimated to be about -0.1. Using the N-relaxation scheme with the earlier estimations for $\Pi(G)$ and $\pi(G)$, a savings of 40% in the number of iterations is obtained (see Fig. 12). A combination of Wynn's ε-algorithm (with $p = 6$, $k = 10$) and an application of the N-relaxation factor yields even greater savings, as shown in Fig. 13.

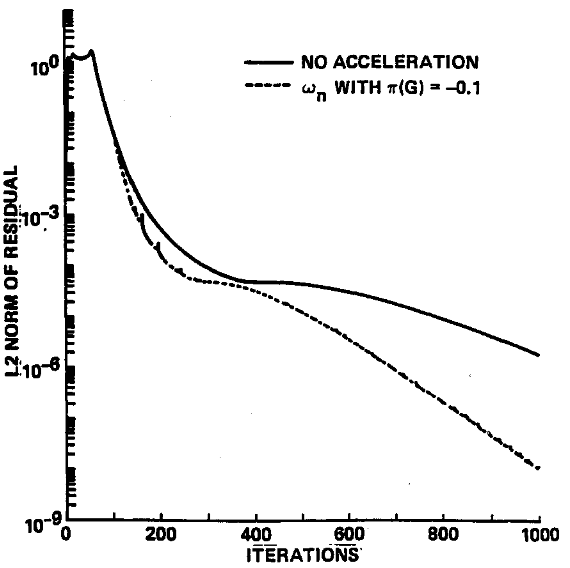

Fig. 12 Convergence of CNS code with and without the N-relaxation factor: $M_\infty = 15$; $Re_L = 5 \times 10^6$; $\alpha = 0$; blunt cone grid 31 × 34 × 23.

Conclusions

By applying Arnoldi's method of the flux-vector-splitting code, the eigenvalue distribution of the iterative matrix is known to have a pattern, as shown in Figs. 1 and 2. The RF method is successfully applied to accelerate the convergence of the iterative scheme [Eq. (11)]. Unlike Wynn's ε-algorithm, the relaxation method can be applied in the early stages of the iterative process. The relaxation method is also a good preconditioner for Wynn's ε-algorithm. A new relaxation factor ω_N is derived and it is confirmed to have the same properties as the RF relaxation factor. In addition, ω_N gives faster convergence than either the RF method or the shifting method. Also, ω_N is a better preconditioner for Wynn's ε-algorithm.

References

[1]Edwards, T., Chaussee, D., Lawrence, S., and Rizk, Y., "Comparisons of Four CFD Codes as Applied to a Hypersonic All-Body Vehicle," AIAA Paper 87-2642, 1987.

[2]Cheer, A., Saleem, M., Pulliam, T., and Hafez, M., "Analysis of the Convergence History of Flow Through Nozzles with Shocks," AIAA Paper 88-3795, July 1988.

[3]Shank, D., "Nonlinear Transformation of Divergent and Slowly Convergent Sequences," *Journal of Mathematics and Physics*, Vol. 34, 1955, pp. 1–42.

[4]Delahey, J. P., and Germain-Bonne, B., "The Set of Logarithmically Convergent Sequences Cannot be Accelerated," *Journal of Numerical Analysis*, Vol. 19, 1982, pp. 840–844.

[5]Brezenski, C., "Convergence Acceleration Methods: The Past Decade," *Journal of Computational and Applied Mathematics*, Vol. 12-13, 1975, pp. 29–36.

[6]South, J., Jr., and Jameson, A., "Relaxation Solutions for Inviscid Axisymmetric Transonic Flow over Blunt or Pointed Bodies," *Proceedings of AIAA CFD Conference*, AIAA, New York, 1973, pp. 8–17.

[7]Bailey, F., and Steger, J., "Relaxation Techniques for 3-D Transonic Flow About Wings," *AIAA Journal*, Vol. 1, No. 3, 1973, pp. 318–325.

[8]Hafez, M., and Cheng, H., "Convergence Acceleration and Shock Fitting for Transonic Aerodynamics Computation," AIAA Paper 75-51, 1975.

[9]Hafez, M., and Cheng, H., "Convergence Acceleration of Relaxation Solution for Transonic Flow Computations," *AIAA Journal*, Vol. 5, No. 3, 1977, pp. 329–336.

[10]Martin, E., and Lomax, H., "Rapid Finite-Difference Computation of Subsonic and Transonic Aerodynamic Flows," AIAA Paper 74-11, 1974.

[11]Hafez, M., Palaniswamy, S., Kuruvila, G., and Salas, M., "Convergence Acceleration of Iterative Solutions of Euler Equations for Transonic Flow Computation," AIAA Paper 85-1641, 1985.

[12]Hafez, M., Palaniswamy, S., Kuruvila, G., and Salas, M., "Application of Wynn's ε-Algorithm to Transonic Flow Calculations," AIAA Paper 87-1143, 1987.

[13]Sidi, A., "Convergence Acceleration for Vector Sequences and Application to Computation Fluid Dynamics," NASA TM 101327.

[14]Eriksson, L., and Rizzi, A., "A Computer Aided Analysis of the Convergence to Steady State of Discrete Approximations to the Euler Equations," *Journal of Computational Physics*, Vol. 57, No. 1, 1985, pp. 90–128.

[15]Pulliam, T., "Euler and Thin Layer Navier-Stokes Code: ARC2D, ARC3D," Notes for CFD User's Workshop, Univ. of Tennessee Space Inst., Tullahoma, TN, March 1984.

[16]Prince, T., "Conjugate Gradient Methods for Solution of Finite Element and Finite Difference Flow Problems," AIAA Paper 83-1923, 1983.

[17]Wong, Y., and Hafez, M., "Application of Conjugate Gradient Methods to Transonic Finite Difference and Finite Element Calculations," AIAA Paper 81-1032, 1981.

[18]Wigton, L., Yu, N., and Young, D., "GMRES Acceleration of Computational Fluid Dynamics Codes," AIAA Paper 85-1494, 1985.

[19]Wynn, P., "Acceleration Technique for Iterated Vector and Matrix Problems," *Mathematics and Computations*, Vol. 16, 1962, pp. 301–322.

[20]Steger, J., and Warming, F., "Flux Vector Splitting of the Inviscid Gasdynamics Equations with Application to Finite Difference Methods," *Journal of Computational Physics*, Vol. 13, No. 2, 1981, p. 263.

[21]Flores, J., Rtan, J. and Edwards, T., "CNS Code Development (U), Paper 12, 5th National Aero-Space Plane Symposium, Oct. 18–26, Hampton, VA., 1988.

[22]Buning, P., and Steger, J., "Solution of the Two-Dimensional Euler Equations with Generalized Coordinate Transformation Using Flux Vector Splitting," AIAA Paper 82-0971, 1982.

[23]Hindman, G., "Geometrically Induced Errors and Their Relationship to the Form of the Governing Equations and the Treatment of Generalized Mappings," AIAA Paper 81-1008, 1981.

[24]Saad, Y., "Variations on Arnoldi's Method for Computating Eigenvalues of Large Usymmetric Matrices," *Linear Algebra and Its Applications*, Vol. 23, 1980, pp. 269–295.

[25]Golub, G., and van Loan, C., *Matrix Computations*, Johns Hopkins Univ. Press, Baltimore, MD, 1983.

[26]Frankel, S., "Convergence Rates of Iterative Treatments of Partial Differential Equations," *Mathematical Tables and Other Aids to Computation*, Vol. 4, 1950, p. 65.

[27]Cheung, S., "Convergence Acceleration in Viscous and Inviscid Hypersonic Flow Calculations," Ph.D. Dissertation, Univ. of California, Davis, CA, 1989.

[28]Saleem, M., "Spectrum Analysis and Convergence Acceleration Techniques Applied to Implicit Finite Difference Approximations for the Euler and Navier-Stokes Equations," Ph.D. Dissertation, Univ. of California, Davis, CA, 1988.