JOURNAL OF COMPUTATIONAL PHYSICS 97, 398–413 (1991)

Eigenvalue Calculation Procedure for an Euler/Navier–Stokes Solver with Application to Flows over Airfoils

APARAJIT J. MAHAJAN,* EARL H. DOWELL, AND DONALD B. BLISS

Department of Mechanical Engineering and Materials Science, School of Engineering, Duke University, Durham, North Carolina 27706

Received November 22, 1989; revised September 20, 1990

A Lanczos procedure is applied to a Navier–Stokes solver for computing eigenvalues and eigenvectors. These eigenvalues and eigenvectors are associated with small perturbation analysis of a finite difference representation of the Navier–Stokes equations for flows over airfoils. A combination of block tridiagonal matrices is converted into a two-dimensional matrix for this eigensystem calculation. This matrix is very large, sparse, real, and nonsymmetric. A separate procedure, based on lopsided iteration, is also used to determine the eigensystem. The results from these two procedures are compared. The Lanczos procedure provides complete spectral information about the eigenvalues, whereas the lopsided iteration provides only a few of the eigenvalues which are largest in magnitude and the corresponding eigenvectors. Such eigensystem information is central to transient stability analysis of Navier–Stokes solvers, for determining the modal behavior of fluid in a fluid-structure interaction problem and for development of reduced order models based on variational principles for Navier–Stokes solvers.

INTRODUCTION

Eigensystem analysis of dynamic equilibrium equations for a structural system has been a standard procedure among researchers for many years. Dynamic equilibrium equations for a structure are written either in a finite difference or a finite element form. The mass, stiffness, and damping matrices are calculated in order to formulate an eigenvalue problem for the structure. With the advent of larger and faster computers, it is possible to analyze structures of increasing complexity, starting with strings, beams, plates, shells, truss assemblies, etc. Various methods have been developed to determine eigenvalues and corresponding eigenvectors for these structural dynamics equations. The matrices involved in eigensystem calculation are real, symmetric, and sometimes positive definite. Also they can be formed easily once the governing equations are discretized. See [1] for a comprehensive treatment of eigensystem problems in structural dynamics.

In fluid dynamics, nonlinearity in the governing equations and the use of Eulerian reference frame make it very difficult to formulate an eigensystem analysis.

* Presently at the University of Toledo and NASA Lewis Research Center.

0021-9991/91 $3.00

Discretization schemes, block treatment of variables, and linearized solution procedures used in fluid dynamics problems further complicate the formulation of an eigensystem analysis. The existing eigensystem analysis procedures developed for structural dynamics problems are thus clearly incapable of modeling fluid dynamics problems.

Over the last few years, many researchers have developed algorithms and codes to obtain steady state solutions for fluid dynamics problems. The Euler or Navier–Stokes equations are solved in a time marching fashion with approximate initial starting data (usually uniform flow conditions) and are converged to a steady state flowfield in an iterative manner. The main motivation behind previous attempts to perform an eigensystem analysis of fluid dynamics problems was to improve the convergence rates of these iterative schemes. Eigensystem information was used to reduce the number of time steps required to reach a steady state solution and accelerate the convergence.

Lomax *et al.* [2] formulated and determined an eigensystem for a one-dimensional diffusion equation expressed in a finite difference form. They formed the matrix analytically and used standard eigenvalue-computing routines to calculate the eigenvalues numerically. These routines are limited in application to matrices up to, at most, order 200. Eriksson and Rizzi [3] formed an eigensystem analysis for transonic Euler equations. They used Frechet derivatives to develop the Jacobian matrix with Arnoldi's method to condense it into an upper Hessenberg matrix. The eigensystem was determined about an arbitrary solution point during the time marching and depended on the time step used for time marching. A spectrum transformation was used to extract "interesting" eigenvalues from Arnoldi's method. Cheer *et al.* [4] used a simular procedure to determine the eigensystem of a one-dimensional nozzle flow and to improve the convergence rates. Saleem [5] used Arnoldi's method with Frechet derivatives for a one-dimensional nozzle flow and a two-dimensional Navier–Stokes solver for flow over airfoils. He applied the eigensystem information to spectrum shifting algorithms for accelerating the convergence to a steady state flowfield. Nordström [6] studied the eigensystem corresponding to linearized, symmetrized two-dimensional Navier–Stokes equations. The equations were Fourier transformed in the y-direction and Laplace transformed in the t-direction. He analyzed the effect of inflow, outflow type boundary conditions in the x-direction, on the stability of the solution algorithm. Eigenvalues were calculated from a continuous and also a semi-discrete model. Nordström [6] and also Engquist and Gustafsson [7] have studied substantially simpler physical problems and smaller numerical problems with the purpose of accelerating numerical convergence employing some of the techniques that are also used in this paper.

These existing ways for formulating an eigensystem analysis have a narrow application and that is to improve the convergence rates for a steady state solution, where accurate time histories are not important. (A complete eigensystem may be needed for some other applications, such as developing reduced order models, in the time accurate sense). Due to the partial evaluation of the Jacobian matrix about

a solution point with Frechet derivatives and subsequent use of Arnoldi's method, one can never obtain the complete eigensystem. Moreover, the eigenvalues calculated using these methods are time step dependent. Finally, both analytical and numerical temporal discretizations are used in forming these eigenvalue problems. By contrast, in classical, exact eigenvalue problems, spatial discretization is done numerically and temporal dependence is expressed analytically.

The Navier–Stokes solver used in the present work represents the formulation and equations used by computational fluid dynamics (CFD) researchers and practitioners for analyzing complex flows such as unsteady, transonic, viscous flows over airfoils. These CFD codes typically run for hours on supercomputers and have a very large number of degrees of freedom. The eigensystem information of such solvers is important in many respects. It can be used for determining the necessity of fine grids in the physical domain which result in a very large number of degrees of freedom. The order of the system can be reduced by modal reduction techniques based on the eigensystem (similar to structural analysis). The eigenmodes corresponding to the continuous physical problem, the discretization scheme, the physical and numerical boundary conditions, the steady state flowfield, the artificial dissipation, and the grid density and size can be distinguished to understand the physical problem and its numerical model better.

In the present work, the first such effort is made to formulate a classical, exact eigensystem problem for a Navier–Stokes solver with application to flows over airfoils. The eigenvalue problem associated with small amplitude time dependent perturbation analysis of a finite difference representation of the full Navier–Stokes equations is formulated with respect to *steady flow* over airfoils. It should be noted that this steady flow calculation is *independent* of the time step used to reach it and the eigenvalue problem formulated about this steady flow uses a semi-discrete method. In this eigenvalue problem, the spatial discretization is done numerically (based on the grid), whereas the temporal discretization is done analytically. The resulting eigenvalue problem is thus *independent* of the time step, in contrast to the earlier eigenvalue problem formulations about transient solutions. The Navier–Stokes solver uses a combination of block tridiagonal matrices to determine the flowfield at each time step. In this eigensystem calculation, these matrices are converted into a two-dimensional matrix. This conversion is accomplished by writing the independent flow variables into a single array (instead of a two-dimensional matrix) and then transforming the corresponding four-dimensional coefficient matrix into a two-dimensional matrix. This matrix is very large ($24{,}000 \times 24{,}000$), sparse, real, and unsymmetric.

The state of the art software available for eigenvalue calculation (EISPACK, etc.) is NOT capable of storing a $24{,}000 \times 24{,}000$ matrix or utilizing the sparsity and nonsymmetry of the present problem for obtaining a solution in reasonable computer time. A modified Lanczos recursive procedure with no reorthogonalization is used to calculate the eigenvalues and the corresponding eigenvectors. A separate procedure, based on lopsided iteration, is also used to determine the eigensystem as an independent check. The results from these two procedures are in good agree-

ment. The Lanczos procedure provides complete spectral information about the eigenvalues. It calculates the eigenvalues that are at both ends of the spectrum and then the ones in the middle of the spectrum [8]. However, the lopsided iteration procedure provides only a few of the eigenvalues that are largest in magnitude or at only one end of the spectrum [9].

This eigensystem information has many attractive and important applications. It can be used in transient stability analysis of Navier–Stokes solvers. The fluid modal behavior in a fluid-structure interaction problem (e.g., flutter) can also be determined from a knowledge of the eigensystem associated with the fluid dynamics equations. Models of reduced order can be developed, using variational principles and eigenvector transformation to modal coordinates, to calculate system response. At present, these types of methods are widely used in structural dynamics. The present work is intended to provide the information necessary for extension of these methods to fluid dynamics.

Navier–Stokes Solver

A Navier–Stokes code capable of calculating unsteady, transonic and separated flows for different airfoil motions, such as pitching and plunging, was selected for the eigenvalue analysis. This finite difference, time marching code was developed by Sankar, based on the Beam–Warming algorithm [10]. This code solves the unsteady, two-dimensional Navier–Stokes equations on a body-fitted moving coordinate system in a strong conservative form using the alternate direction implicit (ADI) procedure with approximate factorization [11]. The convective terms are treated implicitly and the viscous terms are treated explicitly. The body-fitted grid is of the C-type and is shown in Fig. 1. This code is the latest in the series of Navier–Stokes solvers currently in use for transonic, viscous unsteady problems. It has received wide attention from the aerodynamic research community. Only those parts of the Navier–Stokes solver essential for understanding the present work are described below.

All the calculations are performed in a transformed coordinate system (ξ, η, τ) which is linked to the moving, body-fitted coordinate system by equations of the following form:

$$\xi = \xi(x, y, t), \qquad \eta = \eta(x, y, t), \qquad \tau = \tau(t). \tag{1}$$

The Jacobian of transformation is given by

$$J = \xi_x \eta_y - \xi_y \eta_x = 1/(x_\xi y_\eta - x_\eta y_\xi). \tag{2}$$

The two-dimensional unsteady Navier–Stokes equations in the transformed coordinate system are written as

$$\tilde{q}_\tau + \tilde{F}_\xi + \tilde{G}_\eta = \tilde{R}_\xi + \tilde{S}_\eta, \tag{3}$$

where

$$\tilde{q} = J^{-1}\{\rho, \rho u, \rho v, e\}; \tag{4}$$

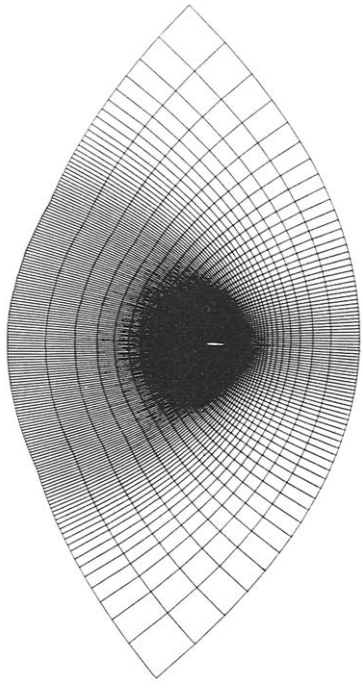

FIG. 1. The C-grid around a NACA 0012 airfoil used in the Navier–Stokes code.

ρ is the fluid density, u and v are the cartesian components of fluid velocity, and e is the total energy of the fluid per unit volume. The quantities $\tilde{F}$, $\tilde{G}$, $\tilde{R}$, and $\tilde{S}$ are given by

$$\begin{aligned} \tilde{F} &= (\xi_x F + \xi_y G + \xi_t \tilde{q})/J \\ \tilde{G} &= (\eta_x F + \eta_y G + \eta_t \tilde{q})/J \\ \tilde{R} &= (\xi_x R + \xi_y S)/J \\ \tilde{S} &= (\eta_x R + \eta_y S)/J \end{aligned} \tag{5}$$

and

$$\begin{aligned} F &= (\rho u, \rho u^2 + p, \rho uv, u(e+p)) \\ G &= (\rho v, \rho uv, \rho v^2 + p, v(e+p)) \\ R &= (0, \tau_{xx}, \tau_{xy}, R_4) \\ S &= (0, \tau_{xy}, \tau_{yy}, S_4) \\ R_4 &= u\tau_{xx} + v\tau_{xy} + K(a^2)_x \\ S_4 &= u\tau_{xy} + v\tau_{yy} + K(a^2)_y, \end{aligned} \tag{6}$$

where a is the speed of sound and K is the thermal conductivity.

In Eq. (3), central differencing is used for spatial derivatives and forward differencing is used for time derivatives. The resulting equation in the discretized form at any point (i, j) is

$$\frac{\Delta\tilde{q}_{ij}^{n+1}}{\Delta\tau} + \delta_\xi \tilde{F}_{ij}^{n+1} + \delta_\eta \tilde{G}_{ij}^{n+1} = \delta_\xi \tilde{R}_{ij}^{n+1} + \delta_\eta \tilde{S}_{ij}^{n+1} - \varepsilon_E D_{ij}^n, \tag{7}$$

where $\Delta\tilde{q}_{ij}^{n+1} = \tilde{q}_{ij}^{n+1} - \tilde{q}_{ij}^n$, and the superscripts n and $n+1$ refer to two successive time levels. D is the artificial viscosity term containing spatial derivatives of second and fourth order and ε_E controls the amount of this *explicit* artificial viscosity. The highly nonlinear terms F and G are then expanded in a Taylor series about time level n. The resulting equation is then approximately factored into two operators so as to allow for two sweeps, a ξ- and an η-sweep, during which block tridiagonal matrix equations are solved. To allow for the explicit treatment of the viscous terms, implicit smoothing terms are added to the left-hand side of Eq. (7). These terms contain spatial derivatives of second order and ε_I controls the amount of *implicit* smoothing. The final form of the governing equation is

$$\left(I + \Delta\tau\left(\delta_\xi A - \frac{\varepsilon_I}{J}\delta_{\xi\xi} J\right)\right)\left(I + \Delta\tau\left(\delta_\eta B - \frac{\varepsilon_I}{J}\delta_{\eta\eta} J\right)\right)\Delta\tilde{q}^{n+1}$$
$$= [-(\delta_\xi \tilde{F}^n + \delta_\eta \tilde{G}^n) + \delta_\xi \tilde{R}^n + \delta_\eta \tilde{S}^n - \varepsilon_E D^n]\,\Delta\tau, \tag{8}$$

where $A = [\partial\tilde{F}/\partial\tilde{q}]^n$ and $B = [\partial\tilde{G}/\partial\tilde{q}]^n$.

The above approximate factorization leads to two sweeps, a ξ- and an η-sweep, during which block tridiagonal matrix equations are solved. All the boundary conditions are explicitly applied at each time step. At the far field boundaries, except in the downstream boundary, the flow field is assumed to be undisturbed. At the downstream boundary, the velocity field u, v and the entropy are extrapolated from the interior, so as to allow for vorticity transport. The pressure at the downstream boundary is prescribed to be the freestream pressure. On the solid boundary, for inviscid flows, the normal velocity is set to zero and, for viscous flows, the normal and tangential velocities are set to zero. The density is extrapolated from the interior. For calculation of viscous, turbulent flows, a two-layer Baldwin–Lomax eddy viscosity model is used [12].

Artificial Viscosity Terms

In this Navier–Stokes solver, a switched dissipation scheme as proposed by Jameson [13], is used. The dissipation terms are written, in conservation form, as a combination of second- and fourth-order terms. A sensor, based on the second derivative of pressure, turns on the second-order dissipation term in the vicinity of shocks and suppresses the fourth-order dissipation term. Jameson's approach leads to crisp, three-point shocks in most cases, without the overshoots. This switching was employed only in the streamwise (ξ-) direction in this Navier–Stokes solver.

The dissipation term is written as

$$D_{ij} = J^{-1}\delta_{\eta\eta\eta\eta}(J\tilde{q}) + \delta_{\xi}\frac{C_1}{J_{i+1/2,j}}(\tilde{q}_{i+2,j} - 3\tilde{q}_{i+1,j} + 3\tilde{q}_{ij} - \tilde{q}_{i-1,j})$$

$$-\frac{C_2}{J_{i+1/2,j}}(\tilde{q}_{i+1,j} - \tilde{q}_{ij}), \tag{9}$$

where

$$C_2 = \text{Max}(\beta_{ij}, \beta_{i+1,j})$$

and

$$\beta_{ij} = H\frac{|(p_{i+1,j} - 2p_{ij} + p_{i-1,j})|}{(p_{i+1,j} + 2p_{ij} + p_{i-1,j})}$$

and

$$C_1 = \text{Max}(0, 1 - C_2).$$

Here H is a user-input value, of the order of 10. The quantity β_{ij} senses the second derivative of pressure and is large only near shocks. This leads to a non-zero value for C_1, near shocks. The value $J_{i+1/2,j}$ needed in the above calculation is computed as

$$\frac{1}{J_{i+1/2,j}} = \frac{1}{2}\left(\frac{1}{J_{ij}} + \frac{1}{J_{i+1,j}}\right). \tag{10}$$

The effect of these artificial viscosity terms on the transient stability of the Navier–Stokes solver is described in detail in [14].

Eigenvalue Problem

An eigenvalue problem is formulated for the Navier–Stokes code. Since $\tilde{F}$, $\tilde{G}$, $\tilde{R}$, $\tilde{S}$ are functions of $\tilde{q}$, Eq. (3) can be rewritten as

$$\tilde{q}_{\tau} = -\tilde{F}_{\xi} - \tilde{G}_{\eta} + \tilde{R}_{\xi} + \tilde{S}_{\eta} = \tilde{Q}(\tilde{q}). \tag{11}$$

Here $\tilde{Q}(\tilde{q})$ also contains the artificial viscosity terms. For small perturbations about steady flow, the following eigenvalue problem results. Substitute $\tilde{q} = \bar{q} + \hat{q}$ in Eq. (11), where $\bar{q}$ is the steady state value; $\hat{q}$ is the small time dependent perturbation. This gives

$$\hat{q}_{\tau} = \tilde{Q}(\bar{q}) + \left[\frac{d\tilde{Q}}{d\tilde{q}}\right]_{\bar{q}}\hat{q}. \tag{12}$$

In the results discussed here, only inviscid flows are considered. The same procedure applies to viscous flows also. At present, the viscosity model used in the Navier–Stokes solver is of an empirical nature. Thus, to determine the contribution of viscous terms to $[d\tilde{Q}/d\tilde{q}]_{\bar{q}}$, the $24{,}000 \times 24{,}000$ derivatives will have to be evaluated numerically and the computational costs are prohibitive. If an analytical viscosity model were used, these derivatives could be evaluated analytically.

At steady state flow conditions, $\tilde{Q}(\bar{q})$ is equal to zero, which reduces Eq. (12) to

$$\hat{q}_\tau = \left[\frac{d\tilde{Q}}{d\tilde{q}}\right]_{\bar{q}} \hat{q}. \tag{13}$$

Substitute $\hat{q} = qe^{\lambda t}$ in Eq. (13):

$$\lambda q e^{\lambda \tau} = \left[\frac{d\tilde{Q}}{d\tilde{q}}\right]_{\bar{q}} q e^{\lambda \tau}. \tag{14}$$

Equation (14) can be written in matrix form as

$$\lambda\{q\} = [P]\{q\}. \tag{15}$$

Consider Eq. (8) before approximate factorization,

$$\left[\frac{d\tilde{Q}}{d\tilde{q}}\right]_{\bar{q}} = -\delta_\xi A - \delta_\eta B + \frac{\varepsilon_I}{J}\delta_{\xi\xi} J + \frac{\varepsilon_I}{J}\delta_{\eta\eta} J + \left[\frac{dD}{d\tilde{q}}\right]_{\bar{q}}, \tag{16}$$

where $A = [\partial\tilde{F}/\partial\tilde{q}]_{\bar{q}}$ and $B = [\partial\tilde{G}/\partial\tilde{q}]_{\bar{q}}$. Analytical expressions are written for A, B, J and $[dD/d\tilde{q}]_{\bar{q}}$. The contribution of convective terms to this derivative (A and B) is determined analytically. The analytical nature of these contributions in both η- and ξ-directions are given in detail in [15].

The following discussion explains the conversion of block matrices (commonly used in CFD) into two-dimensional matrices and vectors (commonly used in eigenvalue problems). This conversion can sometimes pose a significant problem as block CFD schemes are usually optimized for computer memory storage and vector operations. In the Navier–Stokes solver, q is a matrix of dimension (4, *imax*, *jmax*), where *imax* is the maximum number of grid lines in x-direction and *jmax* is the maximum number of grid lines in y-direction. At each grid point (i, j), four flow variables are present as given in Eq. (4). This matrix is converted to a vector containing $(4 \times imax \times jmax)$ number of variables. This vector contains four flow variables in a serial order for *jmax* number of points on *imax* number of lines. In the Navier–Stokes solver, the matrices A and B have dimension of (4, 4, *imax*, *jmax*), the matrix J has dimension of (*imax*, *jmax*) and the matrix D has dimension of (4, *imax*, *jmax*). The same transformation that converted the matrix q into a vector, is applied to the matrices A, B, J, and D to give a resultant matrix P of dimension $(4 \times imax \times jmax, 4 \times imax \times jmax)$. This matrix is sparse and is stored in a rowwise, unordered format. For a typical flow condition this matrix is about 0.09% full.

Due to the alternate direction implicit (ADI) procedure used in the Navier–Stokes solver, the matrices A and B are evaluated for different ordering of vector containing q to maintain the tridiagonal form. The first one orders q as *jmax* number of points for *imax* number of lines (ξ-sweep) and the second one orders q as *imax* number of points for *jmax* number of lines (η-sweep). The order used in forming the resultant matrix P is *jmax* number of points at *imax* number of lines. The matrices A, B, J, and D are calculated in sparse format, before adding their spatial derivatives. For sparse matrix operations and algebra, all the subroutines are written explicitly to keep the code portable and independent of the computer system being used [16]. All the matrices are converted to a form, where a standard eigenvalue analysis can be applied. It should be noted that in the limit of Δt going to 0, $\Delta \tilde{q}^{n+1}$ is equivalent to $\dot{q}$.

Eigensystem Calculation

In Eq. (15), P is a sparse, real, nonsymmetric matrix of order n, where $n = 4 \times imax \times jmax \approx 24{,}000$. The state of the art software available for eigenvalue calculation (EISPACK, etc.) is NOT capable of storing a $24{,}000 \times 24{,}000$ matrix or utilizing the sparsity and nonsymmetry of the present problem for obtaining a solution with reasonable computer time. A procedure that would exploit the sparsity of the matrix for storage and calculation purposes is needed. The iterative methods, for eigenvalue calculations on this scale, need a very good initial guess for convergence. Also they are not capable of providing spectral information about the eigenvalues. The "reduction to condensed forms" methods, however, provide a suitable procedure to attempt a solution to this eigenvalue problem. These methods reduce a general matrix to a smaller matrix with some special properties. The original matrix and the reduced matrix are proven to have a very similar eigensystem [17].

Modified Lanczos Procedure

A modified Lanczos recursive procedure with no reorthogonalization is used to calculate the eigenvalues. For very large real, symmetric matrices, the Lanczos procedure is one of the most widely used ways to estimate the eigenvalues due to its efficiency and range of eigenvalues calculated [17]. This procedure was modified by Cullum and Willoughby [18] for nonsymmetric matrices. A family of complex, symmetric, tridiagonal matrices which represent orthogonal projections of the given original matrix P onto the corresponding Krylov subspaces are calculated during this recursive procedure. The eigenvalues of these Lanczos matrices are the eigenvalues of the operators obtained for P by restricting P to the Krylov subspaces. These eigenvalues are found to be independent of the starting vectors used in the recursion. A more general summary of Lanczos procedure and its modifications is given in [8]. The theoretical background of the Lanczos procedure is beyond the

scope and size of this paper and is explained in [8, 17]. Only the computational implementation of the modified procedure is explained in this paper.

Let P be a real $n \times n$ matrix. Let v_1 and w_1 be two $n \times 1$ vectors with their Euclidian "inner product" $v_1^{\mathrm{T}} w_1 = 1$. These starting vectors can be complex vectors. For $i = 1, 2, ..., m$, the following recursions are used to define Lanczos vectors $W_m \equiv \{w_1, ..., w_m\}$ and $V_m \equiv \{v_1, ..., v_m\}$ and scalars α_i and β_{i+1} such that

$$\begin{aligned} \beta_{i+1} v_{i+1} &= P v_i - \alpha_i v_i - \beta_i v_{i-1} \equiv r_{i+1} \\ \beta_{i+1} w_{i+1} &= P^{\mathrm{T}} w_i - \alpha_i w_i - \beta_i w_{i-1} \equiv t_{i+1} \\ \alpha_i &\equiv (\alpha_i^v + \alpha_i^w)/2, \qquad \beta_{i+1}^2 \equiv (r_{i+1}^{\mathrm{T}} t_{i+1}) \\ \alpha_i^v &\equiv w_i^{\mathrm{T}} (P v_i - \beta_i v_{i-1}), \qquad \alpha_i^w \equiv v_i^{\mathrm{T}} (P^{\mathrm{T}} w_i - \beta_i w_{i-1}). \end{aligned} \tag{17}$$

Recursions are started with $w_1 = v_1$; thus the only nonsymmetry in this calculation is due to the nonsymmetry of P. Even if these starting vectors are real, the nonsymmetry of P soon gives rise to complex vectors. The coefficients α_i and β_i are chosen such that the sets of Lanczos vectors $W_n \equiv \{w_1, ..., w_m\}$ and $V_m \equiv \{v_1, ..., v_m\}$ are (real) biorthogonal. The corresponding complex, symmetric, tridiagonal Lanczos matrix is defined by the scalars α_i and β_{i+1} as

$$T_m = \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \ddots & & \\ & & & \alpha_{m-1} & \beta_{m-1} \\ & & & \beta_m & \alpha_m \end{bmatrix}. \tag{18}$$

This can also be written as $T_m = W_m^{\mathrm{T}} P V_m$. Initially, subroutines from EISPACK were used to calculate the eigenvalues for a general complex matrix, which restricted the number of eigenvalues that could be calculated because the memory requirement was $O(n^2)$ and the time requirement was $O(n^3)$. A procedure based on the QL algorithm was later incorporated in the code. This procedure has a memory requirement of $O(n)$ and a time requirement of $O(n^2)$ [19]. The eigenvalues of T_m are very close approximations to the eigenvalues of P. As m increases, these approximations get better. Normally, good approximations are obtained for m greater than $\sqrt{n}$.

To improve computational efficiency, no reorthogonalization is used. This gives rise to "spurious" eigenvalues. After a particular eigenvalue of T_m has converged to an eigenvalue of P and the corresponding Ritz vector has converged to an eigenvector, the subsequent recursions give rise to linearly dependent Ritz vectors and the biorthogonality of Lanczos vectors is soon lost. This when compounded with the finite precision arithmetic, gives both "good" and "spurious" eigenvalues. This can be corrected by employing reorthogonalization during the recursion process. An alternative is to distinguish between "good" and "spurious" eigenvalues. The most common and computationally efficient way of doing this is to calculate eigenvalues of T_m for some m and then repeat the calculation for a larger m [8]. The

"good" eigenvalues repeat in these calculations for different m, whereas the "spurious" eigenvalues do not repeat. Also, since P is real, its eigenvalues are either real or complex conjugates, but T_m is a complex matrix and may have eigenvalues that are not necessarily the complex conjugates. In the present work, m is varied between 1000 and 1500.

A representative eigenvalue constellation is shown in Fig. 2. Inviscid flow over a NACA 0012 airfoil at an angle of attack of zero degrees and a Mach number of 0.8 is considered. The eigenvalues that repeat for different m are the good eigenvalues and they also appear as complex conjugates. The eigenvalues with the largest and the smallest *magnitudes* (not necessarily the real parts), i.e., at both ends of the spectrum, are calculated first. The eigenvalues in the middle of the spectrum are calculated as m is increased further. Various eigenvalue constellations for different amounts of artificial viscosity are presented in [14].

It turns out, fortunately, that the eigenvalues with the smallest magnitudes (smallest frequencies) generally have the most positive real parts, i.e., the least damping. See Figs. 2, 3, and 4 and also further results in [14].

The eigenvectors of P corresponding to the "good" eigenvalues are determined as follows. First, the eigenvectors $\{z\}$ of T_m corresponding to the "good" eigenvalues are computed. The transformation $\{q\} = [V_m]\{z\}$ gives the eigenvectors $\{q\}$ of P. These eigenvectors are complex and of the dimension $(4 \times imax \times jmax)$ or n. At present, the calculation procedure calculates both the eigenvalues and the corresponding eigenvectors. The eigenvectors are not shown in this paper as they are vectors consisting of 24,000 complex numbers and could not be plotted on a highly clustered grid.

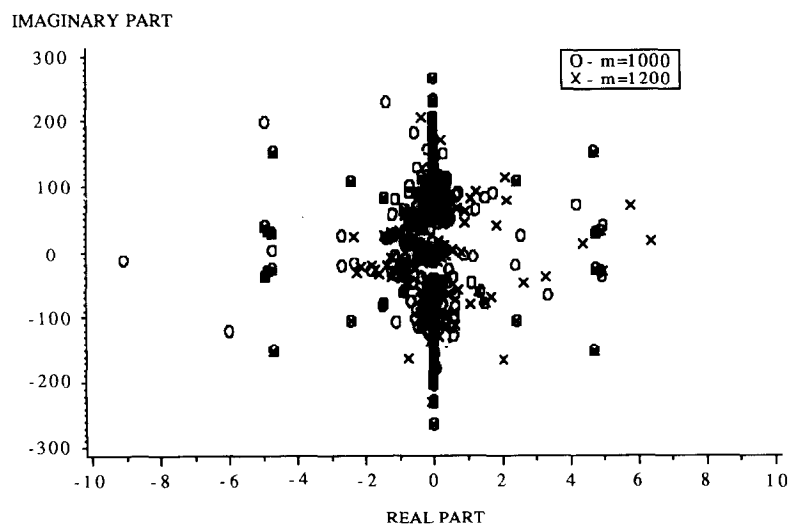

FIG. 2. Eigenvalue constellation for different values of m (NACA 0012 airfoil, $M = 0.8$, inviscid flow, $\alpha = 0°$, $\varepsilon_I = 0$, $\varepsilon_E = 0$).

Lopsided Iteration Procedure

Simultaneous iteration methods are extensions of the power method whereby iteration is carried out with a number of trial vectors that converge onto the eigenvectors corresponding to the dominant eigenvalues. For nonsymmetric matrices, a bi-iteration technique is used to predict left and right eigenvectors simultaneously. When only one set of eigenvectors is required, lopsided iteration technique is used. The algorithm used in the present work is given in detail in [9].

Each iteration cycle of the lopsided iteration involves premultiplication and reorientation, followed by normalization and a tolerance test. The basic procedure is as follows:

Premultiplication. Let P be a matrix of order n for which eigenvalues and right eigenvectors are required. P is premultiplied by a $n \times m$ set of normalized vectors $U \equiv \{u_1, ..., u_m\}$. If the resulting set of vectors is $V \equiv \{v_1, ..., v_m\}$, then

$$V = PU. \tag{19}$$

This step is carried out k number of times after substituting V in place of U for successive multiplications. This results in a substantial reduction in computation time for a sparse matrix as premultiplication washes out the lower eigenvector components from the initial trial vectors.

Reorientation. In this phase, an $m \times m$ interaction matrix C is obtained from the solution of a linear system of equations given by

$$GC = H, \qquad \text{where} \quad G = U^{\mathrm{T}}U \qquad \text{and} \qquad H = U^{\mathrm{T}}V. \tag{20}$$

In this calculation, the values of U and V are those obtained from premultiplication phase. The complete eigensolution of C is then obtained. This eigensolution contains complex conjugate pairs as eigenvalues. The eigenvalues of C correspond to the eigenvalues of P after a sufficiently large number of iterations. The $m \times m$ matrix Y contains right eigenvectors of C.

Sorting. The eigenvalues of C are sorted out in decreasing order according to their magnitude; the corresponding eigenvectors are also arranged accordingly. The eigenvectors of P are then given as

$$X = VY. \tag{21}$$

Normalization and tolerance test. These eigenvectors are then normalized. Depending on the number of eigenvectors being calculated, they are checked for convergence. If no convergence is observed, X is substituted for U in the premultiplication phase and the entire process is repeated.

As the number of iterations and/or m increase, the largest eigenvalues converge at a faster rate. The rate of convergence for these largest eigenvalues depends on sparsity and bandedness of the matrix. Typically, for $m = 20$, $k = 10$, in

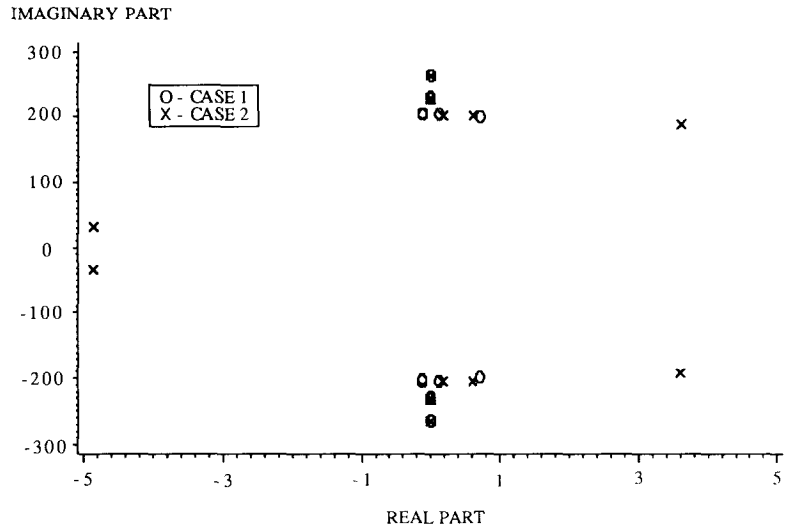

FIG. 3. Comparison of eigenvalues from the lopsided iteration procedure for two cases: Case 1. $m = 20$, $k = 10$, 15 iterations; Case 2. $m = 20$, $k = 5$, 10 iterations (NACA 0012 airfoil, $M = 0.8$, inviscid flow, $\alpha = 0°$, $\varepsilon_I = 0$, $\varepsilon_E = 0$).

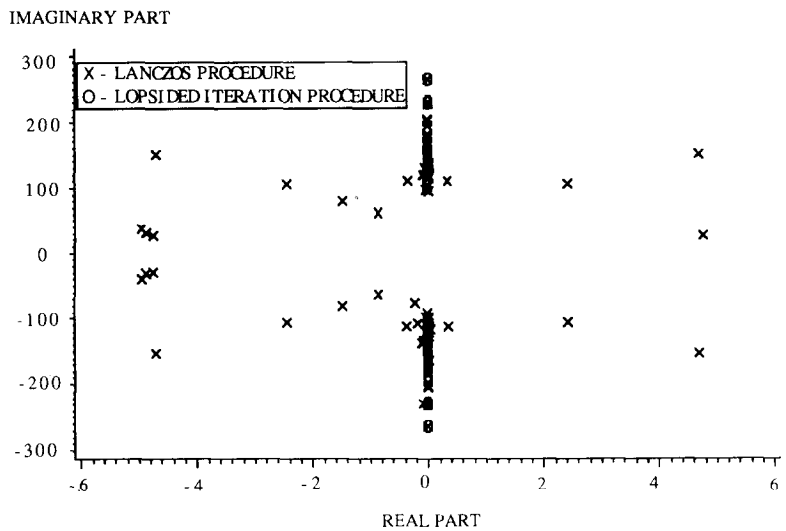

FIG. 4. Comparison of "good" eigenvalues from the Lanczos procedure with "converged" eigenvalues from the lopsided iteration procedure (NACA 0012 airfoil, $M = 0.8$, inviscid flow, $\alpha = 0°$, $\varepsilon_I = 0$, $\varepsilon_E = 0$).

15 iterations, about 12 eigenvalues converge. By increasing any of these parameters, a convergence can be obtained for a larger number of eigenvalues at the same rate or at a faster rate for the same number of eigenvalues. The eigenvalues from two different cases are compared in Fig. 3. In first case, $m = 20$, $k = 10$, and the number of iterations is 15. In the second case, $m = 20$, $k = 5$, and the number of iterations is 10. The largest eigenvalues converge quickly, the eigenvalues with smaller magnitudes converge slowly. Depending on the number of eigenvalues of interest to

TABLE I

Comparison of Eigenvalues From the Lanczos Procedure and the Lopsided Iteration Procedure ($m = 20$, $k = 10$, 15 Iterations)

Lanczos Procedure		Lopsided Iteration Procedure	
Real part	Imaginary part	Real part	Imaginary part
0.0	264.286865333	0.0	264.28685333
0.0	-264.286865333	0.0	-264.28685333
0.0	264.276867199	0.0	264.276867199
0.0	-264.276867199	0.0	-264.276867199
0.0	231.004612882	0.0	231.004612889
0.0	-231.004612882	0.0	-231.004612889
0.0	230.989773391	0.0	230.989773392
0.0	-230.989773391	0.0	-230.989773392
0.0	228.836872362	0.0	228.836872357
0.0	-228.836872362	0.0	-228.836872357
0.0	228.827440150	0.0	228.827440152
0.0	-228.827440150	0.0	-228.827440152
0.0	204.475652580	-0.12733604	204.921209110
0.0	-204.475652580	-0.12733604	-204.921209110
0.0	204.394821069	0.119820597	204.290977827
0.0	-204.394821069	0.119820597	-204.290977827
0.0	202.158284441	-0.12231994	202.121476826
0.0	-202.158284441	-0.12231994	-202.121476826
0.0	202.141394198	0.713979301	199.552068301
0.0	-202.141394198	0.713979301	-199.552068301

a researcher, m and k can be chosen accordingly to get a convergence within a specified number of iterations.

This procedure requires much more computer memory and time than the Lanczos procedure. Also it can calculate only the eigenvalues with the largest magnitude.

The comparison of "converged" eigenvalues from the lopsided iteration procedure with "good" eigenvalues from the Lanczos procedure is shown in Fig. 4. The agreement between these two different procedures for the estimated eigenvalues is very good. It should be noted that these eigenvalues are "accurate" estimates of the eigenvalues of the original matrix P. To illustrate the numerical accuracy of these procedures, numerical values of these eigenvalues are given in Table I. Table I contains all of the 20 eigenvalues calculated by the lopsided iteration procedure in case 1 as described above and the corresponding 20 eigenvalues from the Lanczos procedure. The eigenvalues from the lopsided iteration procedure that have not converged yet, also show a tendency towards convergence. The results from the modified Lanczos procedure could not be compared with the results from some standard routine from EISPACK, as EISPACK does not have a provision for handling very large, sparse matrices. A smaller problem could not be formed as for a small number of grid points, the Navier–Stokes solver was unstable and did not converge to a steady state. In this work, a comparison between the modified Lanczos procedure and the EISPACK routines was made for a mathematical example to check the computer programs. A more complete comparison for smaller physical problems is given in [18].

Conclusion

For the first time, a classical eigenvalue problem formulation and a practical calculation procedure for exact eigenvalues and corresponding eigenvectors are developed and applied to an Euler/Navier–Stokes solver. An algorithm is presented to convert a combination of block tridiagonal matrices into a two-dimensional matrix. Two different procedures are used for the eigensystem determination of a real, unsymmetric matrix. The results from these two procedures are in good agreement. The relative efficiency of these two procedures is also estimated.

For calculation of 1000 eigenvalues, the Lanczos procedure required 4 min of CPU time on an IBM 3090 computer. Whereas the lopsided iteration procedure required 90 min of CPU time for only 40 eigenvalues. For a practical application, the Lanczos procedure is recommended. Use of lopsided iteration procedure should be limited to testing purposes only.

Efforts are underway to use this eigensystem information for studying the transient stability of Navier–Stokes solvers, for developing reduced order models for nonlinear aerodynamics and for determining the modal behavior of a fluid in a fluid-structure interaction system.

ACKNOWLEDGMENTS

This research is sponsored, in part, by the Structural Dynamics Branch at NASA Lewis Research Center under Grant NAG 3-724. Doctors Krishna Rao V. Kaza and George Stefko are the technical monitors. The computer time is provided, in part, by the Cornell National Supercomputing Facility.

REFERENCES

1. K. J. BATHE, *Finite Element Procedures in Engineering Analysis* (Prentice-Hall, Englewood Cliffs, NJ, 1982).
2. H. LOMAX, T. H. PULLIAM, AND D. JESPERSON, AIAA Paper No. 81-1027, Washington, DC, 1981 (unpublished).
3. L. E. ERIKSSON AND A. RIZZI, *J. Comput. Phys.* **57**, 90 (1985).
4. A. Y. CHEER, M. SALEEM, T. H. PULLIAM, AND M. HAFEZ, AIAA Paper No. 88-3795-CP, Washington, DC, 1988 (unpublished).
5. M. SALEEM, Dissertation, Dept. of Mathematics, University of California, Davis, 1988 (unpublished).
6. J. NORDSTRÖM, *J. Comput. Phys.* **85**, 210 (1989).
7. B. ENGQUIST AND B. GUSTAFSSON, *Math. Comput.* **49**, 39 (1987).
8. J. K. CULLUM AND R. A. WILLOUGHBY, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. I, Theory* (Birkhäuser, Boston, 1985).
9. W. J. STEWART AND A. JENNINGS, *ACM Trans. Math. Software* **7**, 184 (1981).
10. N. L. SANKAR AND W. TANG, AIAA Paper No. 85-0129, Washington, DC, 1985 (unpublished).
11. R. M. BEAM AND R. F. WARMING, *AIAA J.* **16**, 393 (1978).
12. B. S. BALDWIN AND H. LOMAX, AIAA Paper No. 78-257, Washington, DC, 1978 (unpublished).
13. A. JAMESON, W. SCHMIDT, AND E. TURKEL, AIAA Paper No. 81-1259, Washington, DC, 1981 (unpublished).
14. A. J. MAHAJAN, E. H. DOWELL, AND D. B. BLISS, *AIAA J.* **29**, 555 (1991).
15. D. A. ANDERSON, J. C. TANNEHILL, AND R. H. PLETCHER, *Computational Fluid Mechanics and Heat Transfer* (Hemisphere, Washington, DC, 1984), p. 490.
16. S. PISSANETZKY, *Sparse Matrix Technology* (Academic Press, London, 1984).
17. J. H. WILKINSON, *The Algebraic Eigenvalue Problem* (Clarendon Press, Oxford, 1965).
18. J. K. CULLUM AND R. A. WILLOUGHBY, "A Practical Procedure for Computing Eigenvalues of Large Sparse Nonsymmetric Matrices," in *Large Scale Eigenvalue Problems*, edited by J. K. Cullum and R. A. Willoughby (Elsevier Science, New York, 1986), p. 193.
19. R. S. MARTIN AND J. H. WILKINSON, *Numer. Math.* **12**, 377 (1968).