AIAA JOURNAL
Vol. 42, No. 5, May 2004

Hopf Bifurcation Calculations for a Symmetric Airfoil in Transonic Flow

K. J. Badcock,* M. A. Woodgate,† and B. E. Richards‡
University of Glasgow, Glasgow, Scotland G12 8QQ, United Kingdom

The application of a sparse matrix solver for the direct calculation of Hopf bifurcation points arising for an airfoil moving in pitch and plunge in a transonic flow is considered. The iteration scheme for solving the Hopf equations is based on a modified Newton's method. Direct solution of the linear system for the updates has previously been restrictive for application of the method, and the sparse solver overcomes this limitation. Results of experiments with the approximation to the Jacobian matrix driving the iteration to convergence are presented. Finally, it is shown that an entire flutter boundary for the NACA0012 airfoil can be traced out in a time comparable to that required for a small number of time-response calculations.

I. Introduction

COMPUTATIONAL fluid dynamics (CFD) has matured to the point where it is being applied to complex problems in external aerodynamics. Aeroelastic analysis relies on high-fidelity predictions of aerodynamics, particularly for phenomena associated with shock motions or separation. These two observations have motivated the development of CFD-based aeroelastic simulation, a field now being called computational aeroelasticity.

Developments in computational aeroelasticity have mainly been focused on time-marching calculations, where the temporal response of a system to an initial perturbation is calculated to determine growth or decay, and from this to infer stability. This type of simulation has developed significantly in the past decade, with efforts concentrating on mesh movement, load and displacement transfer between the aerodynamic and structural grids,[1] and sequencing of solutions.[2,3] Recent and impressive example calculations have been made for complete aircraft configurations.[4,5]

The time-marching method will remain a powerful tool in computational aeroelasticity because of its generality. However, the cost of these calculations motivates attempts to find quicker ways of evaluating stability while still retaining the detailed aerodynamic predictions given by CFD. One way of doing this is to boil down the CFD into a reduced-order model that still retains the essence of the aerodynamics. Various approaches have been proposed, with an expansion of the flowfield in a truncated series of modes derived from proper orthogonal decomposition currently receiving much attention.[6]

A second approach proposed by Morton and Beran from the U.S. Air Force is to use dynamic systems theory to characterize the nature of the aeroelastic instability and then to use this additional information to concentrate the use of the CFD. Aeroelastic instabilities that are commonly termed flutter are of the Hopf type, where an eigenvalue of the system Jacobian matrix crosses the imaginary axis at the flutter point. A model problem was used to evaluate the approach[7] in which the main difficulties associated with the method (calculation of the Jacobian matrix, solution of the augmented system by Newton's method, solution of a large sparse linear system) were considered. The method was applied to an aeroelastic system consisting of an airfoil moving in pitch and plunge in Ref. 8. The linear system was solved using a direct method, and this motivated the use of an approximate Jacobian matrix to reduce the cost. Some robustness problems were encountered when applying the method, particularly at transonic Mach numbers. A complex variable formulation of the problem was introduced in Ref. 9, which resolved some of these problems. An approach considered to reduce the difficulties of applying a direct solver to large linear systems was to use domain decomposition to reduce the size of the system at the expense of an outer iteration over the domains. This was tested on the model problem in Refs. 7 and 9.

The main problems with applying the bifurcation technique can be traced to using a direct solver for the linear system, both in terms of the approximations to the Newton iteration to reduce the cost of solving these systems and in the application to large problems. The aim of the current work is to circumvent this problem by applying sparse matrix solvers. The paper continues with the CFD and computational structural dynamics (CSD) formulation followed by the formulation of the Hopf calculation. The two main challenges for implementing the Hopf calculation are then considered, namely, the generation of the Jacobian matrix and the solution of the linear system. Based on results for these two topics, an iteration scheme is proposed for the solution of the Hopf problem, and results are then presented to illustrate the performance of the scheme.

II. Aerodynamic and Structural Simulations

The two-dimensional Euler equations can be written in conservative form and Cartesian coordinates as

$$\frac{\partial \boldsymbol{w}_f}{\partial t} + \frac{\partial \boldsymbol{F}^i}{\partial x} + \frac{\partial \boldsymbol{G}^i}{\partial y} = 0 \tag{1}$$

where $\boldsymbol{w}_f = (\rho, \rho u, \rho v, \rho E)^T$ denotes the vector of conserved variables. The flux vectors $\boldsymbol{F}^i$ and $\boldsymbol{G}^i$ are

$$\boldsymbol{F}^i = \begin{bmatrix} \rho U^* \\ \rho u U^* + p \\ \rho v U^* \\ U^*(\rho E + p) + \dot{x} \end{bmatrix} \tag{2}$$

$$\boldsymbol{G}^i = \begin{bmatrix} \rho V^* \\ \rho u V^* \\ \rho v V^* + p \\ V^*(\rho E + p) + \dot{y} \end{bmatrix} \tag{3}$$

In Eqs. (2) and (3), ρ, u, v, p, and E are the density, the two Cartesian components of the velocity, the pressure, and the specific total energy, respectively. U^* and V^* are the two Cartesian components

Received 13 August 2002; revision received 25 September 2003; accepted for publication 25 November 2003. Copyright © 2004 by the authors. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0001-1452/04 $10.00 in correspondence with the CCC.

*Reader, Department of Aerospace Engineering.

†Research Assistant, Department of Aerospace Engineering.

‡Mechan Professor, Department of Aerospace Engineering. Member AIAA.

of the velocity relative to the moving coordinate system that has local velocity components $\dot{x}$ and $\dot{y}$, that is,

$$U^* = u - \dot{x} \tag{4}$$

$$V^* = v - \dot{y} \tag{5}$$

The flow solution in the current work is obtained using the Glasgow University code parallel multiblock (PMB). A summary of the applications examined using PMB may be found in Ref. 10. A fully implicit steady solution of the Euler equations is obtained by advancing the solution forward in time by solving the discrete nonlinear system of equations

$$\left(\boldsymbol{w}_f^{n+1} - \boldsymbol{w}_f^n\right)\Big/\Delta t = \boldsymbol{R}_f\left(\boldsymbol{w}_f^{n+1}\right) \tag{6}$$

The term on the right-hand side, called the residual, is the discretization of the convective terms, given here by Osher's approximate Riemann solver (see Ref. 11), MUSCL interpolation,[12] and Van Albada's limiter. The sign of the definition of the residual is opposite to convention in CFD, but this is to provide a set of ordinary differential equations that follows the convention of dynamic systems theory, as will be discussed in the next section. Equation (6) is a nonlinear system of algebraic equations. These are solved by an implicit method,[13] the main features of which are an approximate linearization to reduce the size and condition number of the linear system and the use of a preconditioned Krylov subspace method to calculate the updates. The steady-state solver is applied to unsteady problems within a pseudotime-stepping iteration.[14]

In this paper, the airfoil is allowed to move in pitch and plunge, free of mechanical friction according to the equations

$$\frac{\mathrm{d}\boldsymbol{w}_s}{\mathrm{d}t} = \boldsymbol{R}_s \tag{7}$$

where

$$\boldsymbol{R}_s = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & \frac{1}{2}x_\alpha \\ 0 & 0 & 1 & 0 \\ 0 & x_\alpha & 0 & \frac{1}{2}r_\alpha^2 \end{bmatrix}^{-1} \left\{ \begin{bmatrix} 0 \\ (2/\mu_s\pi)C_l \\ 0 \\ (4/\mu_s\pi)C_m \end{bmatrix} - \begin{bmatrix} 0 & -1 & 0 & 0 \\ (2\omega_R/\bar{U})^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 2r_\alpha^2/\bar{U}^2 & 0 \end{bmatrix} \boldsymbol{w}_s \right\} \tag{8}$$

and $\boldsymbol{w}_s = [h, \dot{h}, \alpha, \dot{\alpha}]^T$, where h is the plunge and α the incidence. Here $r_\alpha = \sqrt{(I_\alpha/m)}$ is the radius of gyration defined in terms of the pitch moment of inertia I_α and the airfoil mass per unit span m, x_α is the offset between the center of mass and the elastic axis, $\mu_s = m/\pi\rho_\infty b^2$ is the airfoil-to-fluid-mass ratio defined in terms of the fluid freestream density ρ_∞ and the semichord b, ω_R is the ratio of the natural frequencies of plunging to pitching, $\bar{U}$ is the reduced velocity, and C_l and C_m are the lift and moment coefficients about the elastic axis, respectively. These equations are solved by a two-stage Runge–Kutta method.

The aerodynamic grid positions and speeds depend on $\boldsymbol{w}_s$. Because no airfoil deformation is present, the initial grid can be rotated and translated according to

$$\begin{bmatrix} x_{i,j} \\ y_{i,j} \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x_{i,j}^0 - x_{\mathrm{ea}} \\ y_{i,j}^0 - y_{\mathrm{ea}} \end{bmatrix} + \begin{bmatrix} x_{\mathrm{ea}} \\ y_{\mathrm{ea}} + h \end{bmatrix} \tag{9}$$

where the superscript 0 indicates the initial position of the i, jth grid point and the subscript ea indicates the location of the elastic axis. The grid speeds can then be calculated from

$$\begin{bmatrix} \dot{x}_{i,j} \\ \dot{y}_{i,j} \end{bmatrix} = \begin{bmatrix} \dot{\alpha}(y_{i,j} - h) \\ -\dot{\alpha}x_{i,j} \end{bmatrix} \tag{10}$$

For coupled CFD–CSD calculations, the aerodynamic and structural solutions must be sequenced. For steady solutions, taking one step of the CFD solver followed by one step of the structural solver will result in the correct equilibrium. However, for time-accurate calculations, more care must be taken to avoid introducing additional errors. The exact formulation used to avoid this is discussed in Ref. 15.

III. Formulation of Hopf Analysis

Consider the semidiscrete form of the coupled CFD–CSD system

$$\frac{\mathrm{d}\boldsymbol{w}}{\mathrm{d}t} = \boldsymbol{R}(\boldsymbol{w}, \mu) \tag{11}$$

where

$$\boldsymbol{w} = [\boldsymbol{w}_f, \boldsymbol{w}_s]^T \tag{12}$$

is a vector containing the fluid unknowns $\boldsymbol{w}_f$ and the structural unknowns $\boldsymbol{w}_s$ and

$$\boldsymbol{R} = [\boldsymbol{R}_f, \boldsymbol{R}_s]^T \tag{13}$$

is a vector containing the fluid residual $\boldsymbol{R}_f$ and the structural residual $\boldsymbol{R}_s$. The residual also depends on a parameter μ, which is independent of $\boldsymbol{w}$. In the case of the pitch–plunge airfoil, $\mu = \bar{U}$. An equilibrium of this system $\boldsymbol{w}_0(\mu)$ satisfies $\boldsymbol{R}(\boldsymbol{w}_0, \mu) = \boldsymbol{0}$.

Dynamic systems theory gives criteria for an equilibrium to be stable.[16] In particular, all eigenvalues of the Jacobian matrix of Eq. (11), given by $A = \partial\boldsymbol{R}/\partial\boldsymbol{w}$, must have negative real part. A Hopf bifurcation with respect to the parameter μ occurs in the stability of the equilibrium at values of μ such that $A(\boldsymbol{w}_0, \mu)$ has one eigenvalue $i\omega$ that crosses the imaginary axis. When the corresponding eigenvector is denoted by $\boldsymbol{P} = \boldsymbol{P}_1 + i\boldsymbol{P}_2$, a critical value of μ is one at which there is an eigenpair ω and $\boldsymbol{P}$ such that

$$A\boldsymbol{P} = i\omega\boldsymbol{P} \tag{14}$$

This equation can be written in terms of real and imaginary parts as $A\boldsymbol{P}_1 + \omega\boldsymbol{P}_2 = \boldsymbol{0}$ and $A\boldsymbol{P}_2 - \omega\boldsymbol{P}_1 = \boldsymbol{0}$. A unique eigenvector is chosen by scaling against a constant real vector $\boldsymbol{S}$ to produce a fixed complex value, taken to be $0 + 1i$. This yields two additional scalar equations $\boldsymbol{S}^T\boldsymbol{P}_1 = 0$ and $\boldsymbol{S}^T\boldsymbol{P}_2 - 1 = 0$.

A bifurcation point can be calculated directly by solving the system of equations

$$\boldsymbol{R}_A(\boldsymbol{w}_A) = \boldsymbol{0} \tag{15}$$

where

$$\boldsymbol{R}_A = \begin{bmatrix} \boldsymbol{R} \\ A\boldsymbol{P}_1 + \omega\boldsymbol{P}_2 \\ A\boldsymbol{P}_2 - \omega\boldsymbol{P}_1 \\ \boldsymbol{S}^T\boldsymbol{P}_1 \\ \boldsymbol{S}^T\boldsymbol{P}_2 - 1 \end{bmatrix} \tag{16}$$

and $\boldsymbol{w}_A = [\boldsymbol{w}, \boldsymbol{P}_1, \boldsymbol{P}_2, \mu, \omega]^T$. If there are n components in $\boldsymbol{w}$, then $\boldsymbol{w}_A$ has $3n + 2$ components, as does $\boldsymbol{R}_A$. Hence, Eq. (15) is closed. The catch is that this is a large sparse system of nonlinear equations.

Newton's method can be used to solve this type of problem. A sequence of approximations $\boldsymbol{w}_A^n$ to a solution is generated by solving the linear system

$$\frac{\partial\boldsymbol{R}_A}{\partial\boldsymbol{w}_A}\Delta\boldsymbol{w}_A = -\boldsymbol{R}_A^n \tag{17}$$

where $\Delta\boldsymbol{w}_A = \boldsymbol{w}_A^{n+1} - \boldsymbol{w}_A^n$. The Jacobian matrix on the left-hand side of Eq. (17) is given in expanded form as

$$\frac{\partial\boldsymbol{R}_A}{\partial\boldsymbol{w}_A} = \begin{bmatrix} A & 0 & 0 & \boldsymbol{R}_\mu & 0 \\ (A\boldsymbol{P}_1)_w & A & I\omega & (A\boldsymbol{P}_1)_\mu & \boldsymbol{P}_2 \\ (A\boldsymbol{P}_2)_w & -I\omega & A & (A\boldsymbol{P}_2)_\mu & -\boldsymbol{P}_1 \\ 0 & \boldsymbol{S}^T & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{S}^T & 0 & 0 \end{bmatrix} \tag{18}$$

There are three key issues for the application of Eq. (17). First, a good initial guess is required, or the iterations will diverge. Second, the Jacobian matrix $\partial \boldsymbol{R}_A/\partial \boldsymbol{w}_A$ is required. Third, the large sparse linear system given in Eq. (17) must be solved. These points will be considered in the following sections.

One simplification arises if we are dealing with a symmetric problem, for example, a symmetrical airfoil at zero incidence.[8] In this case, the equilibrium solution $\boldsymbol{w}_0$ is independent of μ, and hence, can be calculated from Eq. (11) independently of the other Hopf conditions in Eq. (15). Then, a smaller system can be solved for the bifurcation parameter given by

$$\boldsymbol{R}_A = \begin{bmatrix} A\boldsymbol{P}_1 + \omega \boldsymbol{P}_2 \\ A\boldsymbol{P}_2 - \omega \boldsymbol{P}_1 \\ \boldsymbol{S}^T \boldsymbol{P}_1 \\ \boldsymbol{S}^T \boldsymbol{P}_2 - 1 \end{bmatrix} \tag{19}$$

with $\boldsymbol{w}_A = [\boldsymbol{P}_1, \boldsymbol{P}_2, \mu, \omega]^T$. The Jacobian matrix in Newton's method then becomes

$$\frac{\partial \boldsymbol{R}_A}{\partial \boldsymbol{w}_A} = \begin{bmatrix} A & I\omega & A_\mu \boldsymbol{P}_1 & \boldsymbol{P}_2 \\ -I\omega & A & A_\mu \boldsymbol{P}_2 & -\boldsymbol{P}_1 \\ \boldsymbol{S}^T & 0 & 0 & 0 \\ 0 & \boldsymbol{S}^T & 0 & 0 \end{bmatrix} \tag{20}$$

For the rest of this paper we will concentrate on solving the symmetric problem.

IV. Calculation of the Jacobian Matrix

The difficult terms to form in the Jacobian matrix of the augmented system are A and A_μ. The calculation of A is most conveniently done by partitioning the matrix as

$$A = \begin{bmatrix} \dfrac{\partial \boldsymbol{R}_f}{\partial \boldsymbol{w}_f} & \dfrac{\partial \boldsymbol{R}_f}{\partial \boldsymbol{w}_s} \\ \dfrac{\partial \boldsymbol{R}_s}{\partial \boldsymbol{w}_f} & \dfrac{\partial \boldsymbol{R}_s}{\partial \boldsymbol{w}_s} \end{bmatrix} = \begin{bmatrix} A_{ff} & A_{fs} \\ A_{sf} & A_{ss} \end{bmatrix} \tag{21}$$

The block A_{ff} describes the influence of the fluid unknowns on the fluid residual and has by far the largest number of nonzeros for the pitch–plunge airfoil problem. The fluid residual is calculated using the Osher scheme, and the Jacobian matrix is calculated analytically in two stages. First, the dependence of the Osher flux approximations on the left and right states is calculated. Second, the dependence of the left and right states on the local cell values is calculated from the expression for the MUSCL interpolation. The chain rule can then be used to calculate the required Jacobian. In two dimensions, there are seven nonzero 4 × 4 blocks for every cell in the grid. In three dimensions, there are 13 nonzero 5 × 5 blocks. The Jacobian calculated in this way is referred to as second order throughout this paper.

An approximate Jacobian matrix, referred to as modified order, is also used in the iteration scheme defined later and has been used with success to accelerate CFD-only calculations.[17] The approximation is to equate the terms arising from a flux calculation associated with cells to the left and right of an interface with the dependence on the left and right states calculated from the MUSCL interpolation. With this approximation, the number of nonzero contributions arising from each flux calculation is reduced from four blocks to two. This scheme is similar to calculating the exact Jacobian matrix for a first-order spatial discretization, with the modification that the MUSCL interpolated values at the interface are used in the evaluation rather than the cell values that would be used for a first-order spatial scheme.

The dependence of the fluid residual on the structural unknowns is partially hidden by the notation used. The fluid residual depends not only on the fluid cell values but also on the location of the grid points themselves. The fluid and structural unknowns are independent variables, and hence, to calculate the term A_{fs} the fluid unknowns are kept fixed. The influence of the structural unknowns is felt through the moving grid. For example, for an airfoil moving in pitch and plunge, the grid is translated and rotated according to the current values of the structural solution. In addition the residual also depends on the mesh speeds. The easiest way of computing A_{fs} is, keeping $\boldsymbol{w}_f$ fixed, increment the structural unknowns in turn, that is, α, $\dot{\alpha}$, h, and $\dot{h}$, update the grid locations and speeds, reevaluate the fluid residual, and use a finite difference to calculate the Jacobian terms one column at a time. This requires n_s fluid residual evaluations where n_s is the number of structural unknowns and is relatively cheap if n_s is small, as is the case for the pitch–plunge airfoil.

The term A_{sf} essentially involves calculating the dependence of integrated fluid forces on the fluid unknowns. For example, for the pitch–plunge airfoil the fluid variables contribute to the structural equations through the lift and moment coefficients. In turn, these coefficients are calculated using the a linear combination of the values of pressure in the two cells adjacent to the airfoil surface. It is, therefore, straightforward to calculate the exact terms in the Jacobian matrix.

Finally, the exact Jacobian matrix for the dependence of the structural equations on the structural unknowns is easy to calculate from Eq. (8).

For the two-degree-of-freedom airfoil, the bifurcation parameter ($\bar{U}$ in this case) only appears in the structural equations and in terms involving the structural unknowns. Therefore, for this case,

$$A_\mu = \begin{bmatrix} 0 & 0 \\ 0 & \dfrac{\partial^2 \boldsymbol{R}_s}{\partial \mu \partial \boldsymbol{w}_s} \end{bmatrix} \tag{22}$$

Because of the simple algebraic expression for $\partial \boldsymbol{R}_s/\partial \boldsymbol{w}_s$, it is straightforward to calculate the required term analytically.

A simplification is used to reduce storage requirements for the evaluation of the augmented residual, which requires the products $A\boldsymbol{P}_1$ and $A\boldsymbol{P}_2$. This can be done using a matrix-free formulation as

$$A\boldsymbol{x} \approx \frac{\boldsymbol{R}(\boldsymbol{w} + h\boldsymbol{x}) - \boldsymbol{R}(\boldsymbol{w} - h\boldsymbol{x})}{2h} \tag{23}$$

where $\boldsymbol{x}$ denotes the real or imaginary part of the critical eigenvalue and h is the increment applied. Computing this expression is not costly because it requires only two residual evaluations. This gives a very accurate approximation to the required product without having to evaluate and store A. Matrix A is required for the left-hand-side coefficient matrix, but the modified-order approximation is used for this purpose, which reduces the storage. Hence, using the matrix-free evaluation of the augmented residual reduces the memory requirements for the scheme overall and simplifies the code considerably. The use of automatic differentiation[18] tools, with some effort put into recoding the residual calculations, would allow the required terms for the right-hand side to be evaluated exactly. This should be considered for future versions of the code, although the matrix-free calculation has been found to work effectively for the current purpose.

V. Solution of the Linear System

The calculation of the Newton updates requires the solution of the large sparse linear system in Eq. (17). Experience with solving CFD-only problems[19] shows that the system can potentially be solved efficiently by Krylov subspace type iterative solvers (see Ref. 20). The majority of the nonzero terms in the matrix are associated with the eigenvector real and imaginary parts. Hence, initial experiments for the linear solver were carried out for the system with coefficient matrix

$$C = \begin{bmatrix} A & I\omega \\ -I\omega & A \end{bmatrix} \tag{24}$$

where the matrix A is evaluated at an equilibrium solution for the NACA0012 airfoil at a freestream Mach number of 0.5 and zero incidence. The reduced velocity was chosen to be the bifurcation value and ω the imaginary component of the critical eigenvector. The calculations shown here were done on the medium grid, as described later, and the matrix is of modified accuracy unless otherwise stated.

For the formulation of the Newton iterations the matrix is presented in a block form, which is convenient for calculating, coding, and describing the various contributions to the iteration. These blocks are not used in the linear solution, which operates on nonzeros in the matrix regardless of their origin. However, within the Jacobian matrix A, there is also a natural block structure because the discretization of the Euler equations is expressed cell by cell, with four conserved variables in each cell. This means that A consists of 4×4 blocks. For the discussion of the preconditioning of the iterative solver, this latter block structure is either exploited, that is, operations in the factorization are done on the 4×4 blocks, or it is ignored, in which case operations are done directly on the elements of the matrix. The former case is referred to as block and the latter as pointwise.

The key issue for iterative linear solvers is usually the preconditioner. The incomplete lower–upper (ILU) factorization family[20] can be very effective at approximating the inverse of the coefficient matrix with a small number of terms. For CFD calculations, block ILU (BILU) factorizations with no fill in have proved very successful.[10] Here no fill in means that the factorization has the same sparsity pattern as the coefficient matrix.

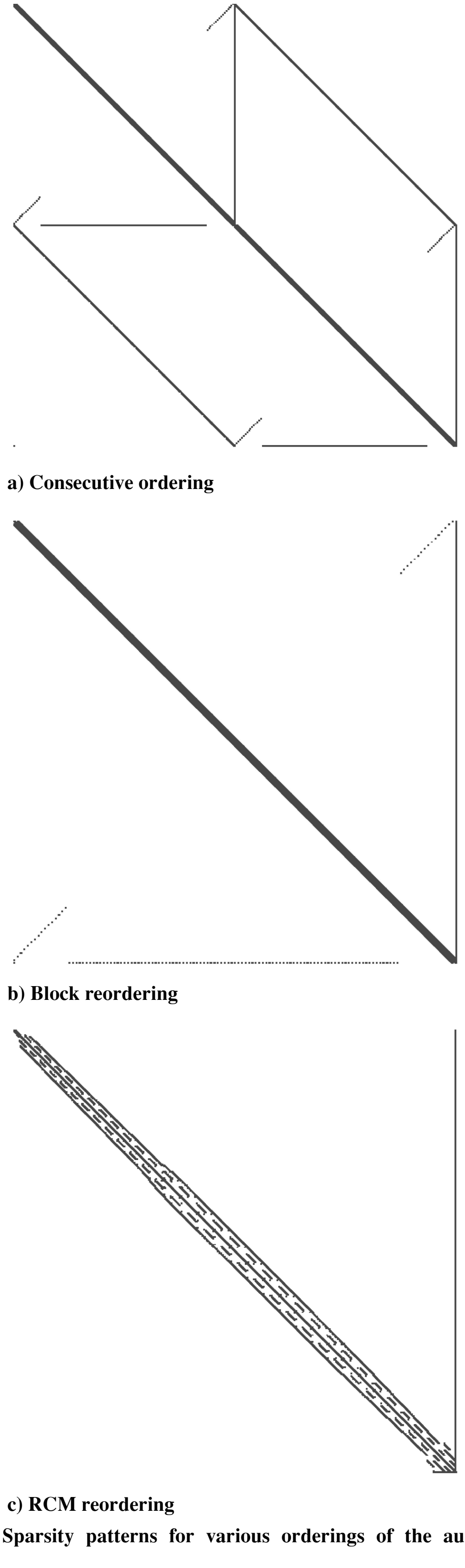
Because of the structure of the coefficient matrix and the previous success of calculating effective preconditioners for matrix A, initial attempts focused on using a factorization of the matrix

$$\begin{bmatrix} A & I\omega \\ -I\omega & A \end{bmatrix} = \begin{bmatrix} I & 0 \\ -\omega A^{-1} & I \end{bmatrix} \begin{bmatrix} A & I\omega \\ 0 & A + \omega^2 A^{-1} \end{bmatrix} \tag{25}$$

However, manipulating the term $A + \omega^2 A^{-1}$ efficiently is not straightforward (in particular, the requirement of the inverse of this term), and so these efforts were abandoned. The BILU factorization of the matrix C is calculated directly as opposed to being constructed in terms of a factorization of A.

The sparse matrix package Aztec[21] was used to carry out experiments for the solution of this system. This package has three main solvers available, namely, GMRES, CGS, and TFQMR, although the differences in performance for the current problem were found to be small. The last solver was found to work best for the current problems, and so is used throughout this work. A variety of preconditioners are also available including pointwise ILU, that is, working on the elements, and BILU, working on the matrix in its block structured form. Various levels of fill in can be generated in the factorization. The pointwise ILU preconditioner allows reordering to minimize the bandwidth by the reverse Cuthill McGee (RCM) algorithm. This is not available for the BILU factorization. Two different orderings have also been used for the matrix when generated. The first lists all of the unknowns associated with the real and then the imaginary parts of the eigenvector, and the second orders the real and imaginary parts of the eigenvector components associated with each cell in the grid consecutively, referred to as block reordering. The sparsity patterns for these two orderings and the RCM reordering are shown in Fig. 1 and verify that the RCM reordering is effective in minimizing the bandwidth of the matrix.

Various calculations were carried out with the test matrix. First, the value of ω was set to zero to obtain a system that is close to that of the CFD–CSD-only problem. Second, the problem was solved with the correct value for ω and with the various orderings for ILU and BILU factorizations. Finally, one of these cases was rerun with a second-order Jacobian matrix for A. The results are summarized in Fig. 2. First, the system with ω set to zero was most easily solved, and the performance of the iterative solver in this case is comparable with previous experience for the CFD–CSD-only system. The RCM reordering makes the largest difference between all of the options for the modified-order matrix, but the performance for all three orderings is similar. Also, the ILU and BILU factorizations give similar convergence behavior. Finally, the second-order system does not converge when using preconditioning with no fill in. For comparison, a calculation was run using level one fill in for the factorization. This results in about 10 times the number of terms being generated in the factorization, which means it is a better preconditioner but is much more expensive to calculate and use. Although the level-one preconditioned system required fewer iterative steps to converge, the CPU time required for the level-one solution was around 20 times longer, and the memory required is an order of magnitude higher.



a) Consecutive ordering

b) Block reordering

c) RCM reordering

Fig. 1 Sparsity patterns for various orderings of the augmented system.

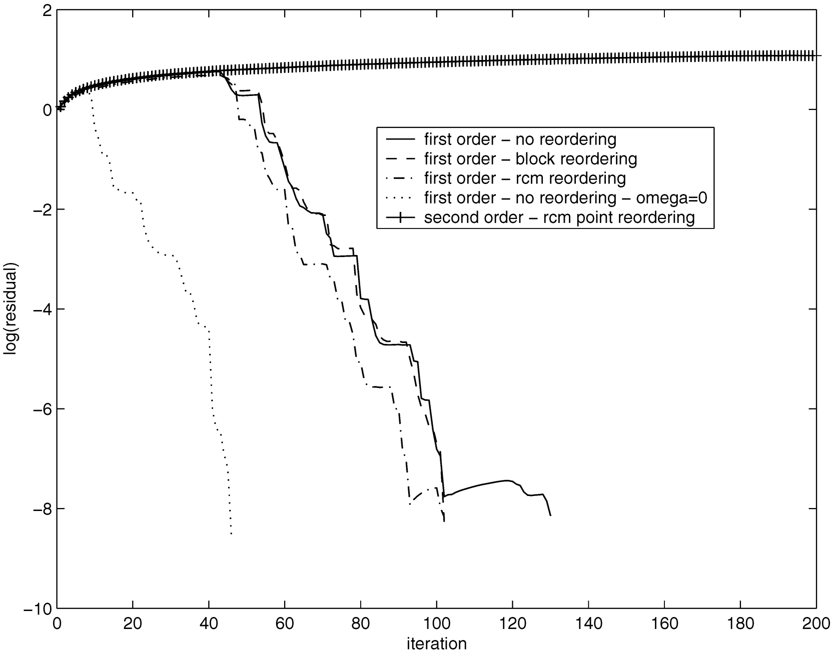


Fig. 2 Convergence histories for TFQMR solution of augmented system using several preconditioning options.

The following conclusions were drawn for the solution of the linear system:

1) The preconditioning for the augmented system cannot easily be based on factorizations of A alone.

2) The augmented linear systems are significantly more difficult to solve than CFD–CSD-only systems.

3) The second-order Jacobians cannot be solved with zero fill in preconditioners, whereas the modified-order Jacobians can.

4) Using RCM reordering marginally improves the convergence rate.

VI. Iteration scheme for Flutter Boundaries

We are free to make any approximations to the coefficient matrix on the left-hand side of Eq. (17) that still lead to a convergent iteration scheme. Approximations will tend to reduce the rate of convergence (and in particular will lead to the loss of quadratic convergence). However, the potential gain if the linear system is made easier to solve can outweigh this effect. This has been exploited for CFD solvers where, for example, the Jacobian matrix associated with a first-order spatial scheme has been used to drive a higher-order scheme to convergence.[17] The advantages are, first, the linear system is much better conditioned and can be solved in a smaller number of iterations, second, the number of nonzero blocks in the matrix is reduced by a factor of 7/13, and, finally, because the stencil is reduced, parallel communication is also reduced during the linear solve. For inviscid flows around airfoils, a reduction in the time to convergence for a CFD-only solver by a factor of four has been achieved.[17]

The results from the tests on the linear solver suggest that the linear system associated with the second-order Jacobian matrix is too badly conditioned to be solved efficiently by the methods used here. The modified-order Jacobian is, therefore, considered as a replacement. This means that the iteration scheme is given by

$$\frac{\partial \bar{\boldsymbol{R}}_A}{\partial \boldsymbol{w}_A}\Delta \boldsymbol{w}_A = -\boldsymbol{R}_A^n \tag{26}$$

where

$$\frac{\partial \bar{\boldsymbol{R}}_A}{\partial \boldsymbol{w}_A} = \begin{bmatrix} \bar{A} & I\omega & \bar{A}_\mu \boldsymbol{P}_1 & \boldsymbol{P}_2 \\ -I\omega & \bar{A} & \bar{A}_\mu \boldsymbol{P}_2 & -\boldsymbol{P}_1 \\ \boldsymbol{S}^T & 0 & 0 & 0 \\ 0 & \boldsymbol{S}^T & 0 & 0 \end{bmatrix} \tag{27}$$

Here

$$\bar{A} = \begin{bmatrix} \bar{A}_{ff} & A_{fs} \\ A_{sf} & A_{ss} \end{bmatrix} \tag{28}$$

where $\bar{A}_{ff}$ is the modified-order fluid Jacobian as described earlier.

The starting solution for the iteration is likely to be important. We assume that the method will be used to trace out a stability boundary for varying values of a parameter, which in the current work is the freestream Mach number. At low values of Mach number, linear aerodynamic theory gives a good estimate of the bifurcation parameter and frequency of the unstable solution (the critical μ and ω). Alternatively, time-marching calculations can be used to find these values at one Mach number. We adopt the notation that the nth approximations to the critical values at the kth Mach number M_∞^k are denoted by $\mu^{n,k}$ and $\omega^{n,k}$ and the converged values by μ^k and ω^k. With this notation, the chosen values for $\mu^{1,1}$ and $\omega^{1,1}$ are assumed to be good estimates to μ^1 and ω^1. Also, the converged values at the previous Mach number give a reasonable initial guess for the next one, that is, $\mu^{1,k+1} = \mu^k$ and $\omega^{1,k+1} = \omega^k$ are satisfactory starting values at M_∞^{k+1}.

The initial guess for the eigenvector is crucial to obtaining convergence. If a good estimate for an eigenvalue is known, then the inverse power method can be used to calculate the corresponding eigenvector.[22] For a matrix A, the inverse power method iteration is given by

$$(A - i\omega_s I)\boldsymbol{P}^m = \boldsymbol{x}_p^{m-1} \tag{29}$$

$$\boldsymbol{x}_p^m = \boldsymbol{P}^m \big/ \|\boldsymbol{P}^m\|_\infty \tag{30}$$

This iteration converges to the eigenvector $\boldsymbol{P}$, which corresponds to the eigenvalue in the spectrum of A that is closest to $i\omega_s$. Writing out the system in Eq. (29) in real form leads to a coefficient matrix of the form given in Eq. (24), and so the linear system to be solved is close to that of the augmented system. Therefore, the eigenvector is calculated for the modified-order Jacobian $\bar{A}$ to again allow easier solution of the linear system. The power method is used to generate the initial approximation to the critical eigenvector at the first Mach number. At subsequent Mach numbers, the converged eigenvector from the previous one is used as the initial guess.

Because the Jacobian matrix has been approximated, it is interesting to see if additional approximations can be made, particularly because it has already been seen that the linear system without the $I\omega$ terms in the off-diagonal blocks is much easier to solve. In addition, the part of one of these terms corresponding to the fluid unknowns was set to zero in Ref. 8 to allow for a more efficient direct solution of the linear system. Experiments were carried out to solve the augmented system at one Mach number (0.5) with various combinations of these terms left out. The convergence rates omitting neither (full), $-I\omega$ (lower), $I\omega$ (upper), and both of these terms is shown in Fig. 3. For the case when one of the terms is omitted, the iteration fails to converge. When both terms are omitted, the iteration converges but to the wrong value of μ. Hence, it appears that in general making further approximations to the augmented Jacobian adversely effects the performance of the scheme.

The iteration scheme for calculating the bifurcation behavior at a new Mach number M_∞^k is, therefore, as follows:

1) Calculate $\bar{A}$ at the converged fluid–structure steady state (all of which except A_{ss} are independent of μ due to symmetry).

2) Set starting values for the iteration as $\omega^{1,k}=\omega^{k-1}$, $\mu^{1,k}=\mu^{k-1}$, and $\boldsymbol{P}^{1,k}=\boldsymbol{P}^{k-1}$.

3) Solve Eq. (26) and update solution by $\boldsymbol{w}_A^{n+1,k}=\boldsymbol{w}_A^{n,k}+\psi\Delta\boldsymbol{w}_A$, where ψ is a relaxation parameter chosen to be between 0 and 1, repeating until convergence.

VII. Results for Symmetric Problem

The test problem considered to illustrate the performance of the proposed scheme is for the NACA0012 airfoil at zero incidence. The parameters for the structural model are given in Table 1. Two cases are considered for varying airfoil mass. The first, called the light case, has $\mu_s=10$, and the second, called the heavy case, has $\mu_s=100$.

The starting grid used for the calculations is of C topology and has 257 points wrapped around the airfoil and 65 points normal (shown in Fig. 4). The mesh is divided into three blocks for the solver, and the block boundaries are indicated in Fig. 4, running normal and streamwise from the trailing edge. The far field is located 15 chords away, and the first spacing on the airfoil surface is 100th of the chord. A medium grid was defined by taking every second point in each direction, a coarse grid by taking every fourth point, and a very coarse grid by taking every eighth point.

Table 1 Structural model parameters

Parameter	Value
r_α	0.539
x_α	−0.2
ω_R	0.343
μ_s	100.0 (heavy case)
μ_s	10.0 (light case)
x_{ea}	0.4
y_{ea}	0.0

To check the mesh used, a steady-state calculation was made for zero incidence and $M_\infty=0.8$, and the results on the fine and medium grids are shown in Fig. 5 and agree closely, with only minor differences in the shock resolution. These results give confidence in the medium grid, which is mainly used for the bifurcation calculations.

A check on the augmented solver can be made for the very coarse grid by computing, using MATLAB® for the light case, the complete eigenvalue spectrum of A at fixed values of μ. The computed augmented solution on this grid and for a Mach number of 0.5 is $\mu=1.6311$. The eigenspectrum for values of μ of 1.62, 1.6311, and

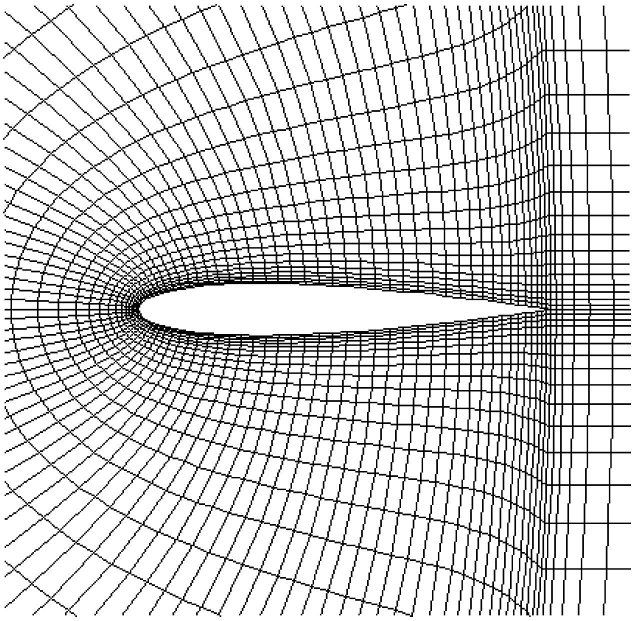


Fig. 4 Fine mesh for NACA0012 airfoil.

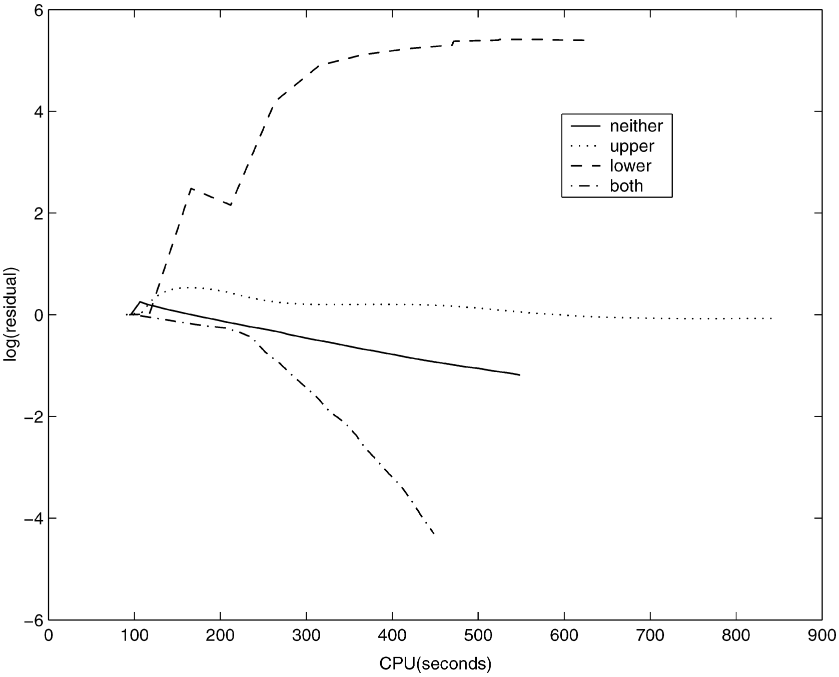


Fig. 3 Comparison of convergence rate for retaining various combinations of $I\omega$ terms in augmented Jacobian matrix.

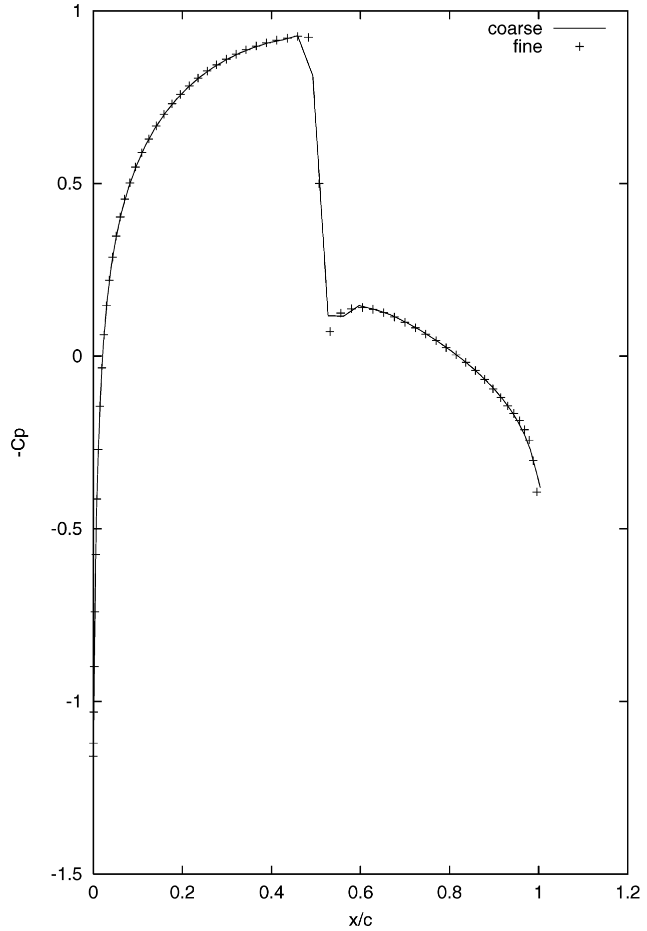
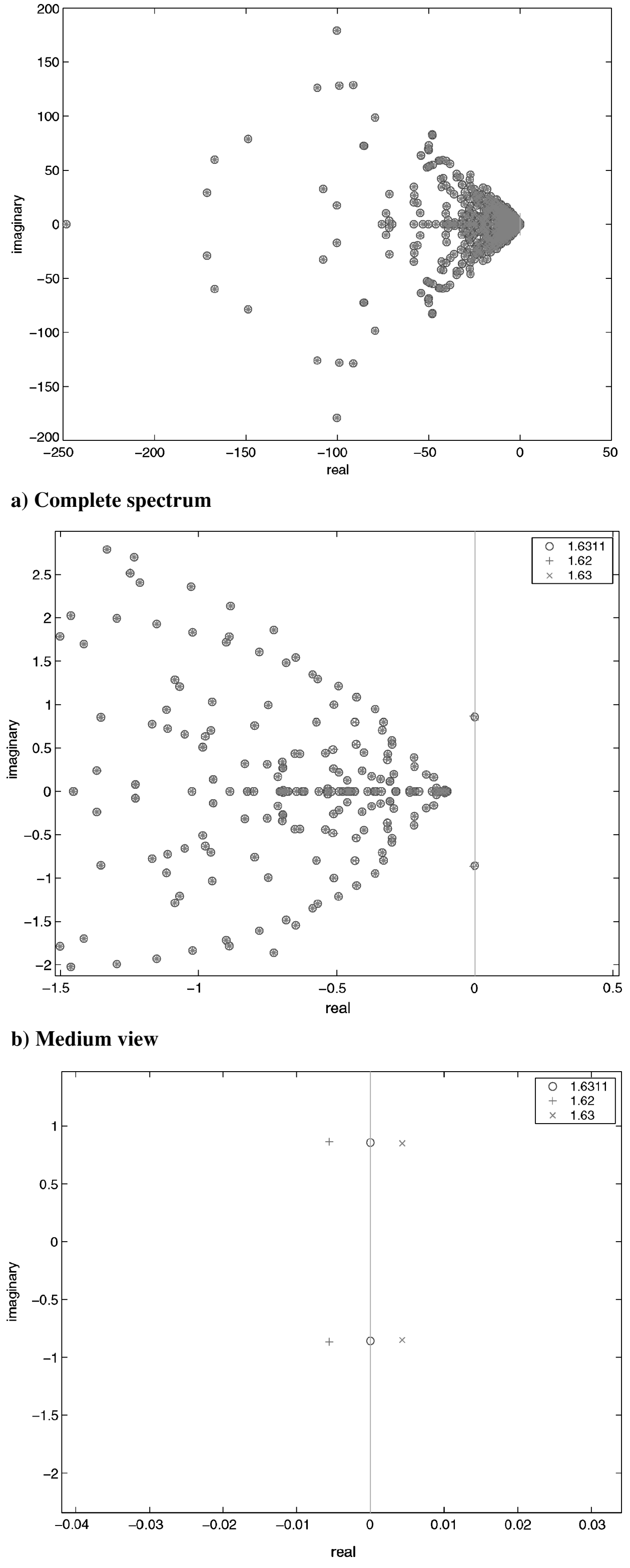


Fig. 5 Comparison of pressure distribution for NACA0012 airfoil at zero incidence and $M_\infty = 0.8$ on the coarse and fine grids.



a) Complete spectrum

b) Medium view

c) Closeup view

Fig. 6 Eigenspectrum for quoted values of μ on a very coarse grid at a Mach number of 0.5.

1.64 are plotted on various scales in Fig. 6. The critical eigenvalue crosses the imaginary axis at the value computed by the augmented solver, providing confirmation of the solver.

The scheme proposed in the preceding section was first used to compute the stability boundary for the light case between Mach numbers of 0.5 and 0.95. This range includes transonic effects. The initial values for μ and ω were found from time-marching calculations at the first Mach number. The bifurcation calculation was first made on the coarse grid for Mach number steps of 0.05 and then on the medium grid for similar steps. These calculations indicated that the behaviour in the region 0.8–0.95 had not been resolved adequately, and the resolution here was increased to steps of 0.01 on each grid. The resulting stability boundaries on the two grids are compared in Fig. 7 and are in good agreement. The augmented residual was reduced by 3–4 orders of magnitude, with up to 20 steps per Mach number used. This was sufficient to converge the bifurcation parameter to five significant figures and so is very conservative. The convergence behavior in terms of the original calculation on the medium grid is shown in Fig. 8, where the reduction in residual and the convergence of μ is shown as a function of the augmented solver iteration. The residual of the linear solver was reduced by two orders at each augmented step. On average this means that 30 Krylov steps are required per linear solve, partly due to the plateau encountered at the start of each solve. Hence, there is scope for improving on the current performance by modifying the preconditioner and relaxing the convergence criteria. Nevertheless, the stability boundary using the initial 10 Mach numbers was traced out for the medium mesh in 4500 CPU s on a 1-GHz processor. An additional 20 Mach numbers were calculated in 7578 s.

The main cost of the direct solution is divided almost evenly between the CFD–CSD calculation of the steady state and the augmented solution. The cost of calculating the flutter point is about equivalent to a CFD steady-state calculation at each Mach number. To put this in perspective, each time-marching calculation requires about 3300 s on a 1-GHz processor to compute four cycles of the response. Four cycles indicates whether or not a solution is diverging for simple problems such as the current one but may be insufficient to see the behavior for a complex system that involves a larger number of frequencies. Care was taken to ensure convergence of these solutions with respect to the time step. Two sets of tests were carried

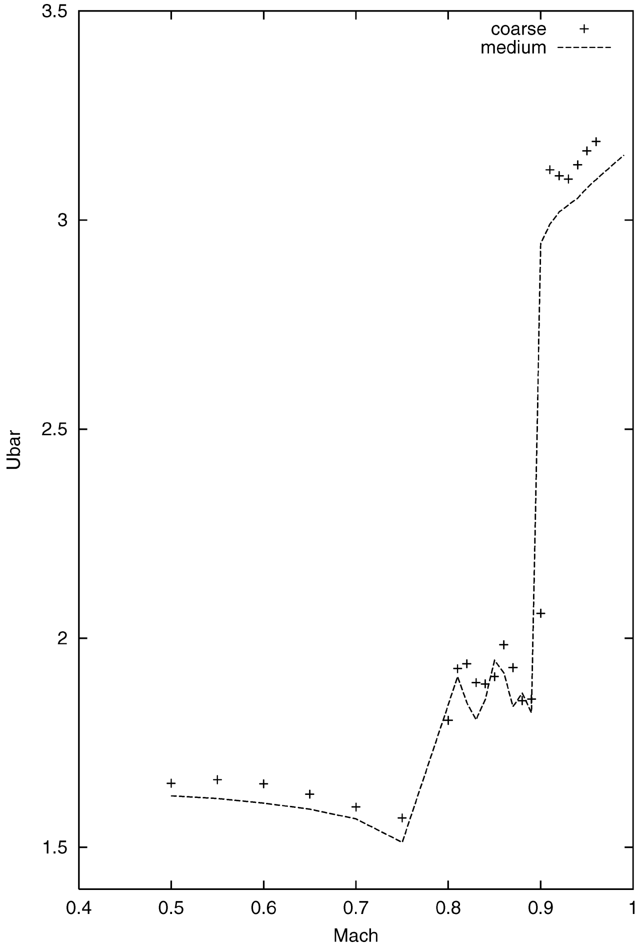
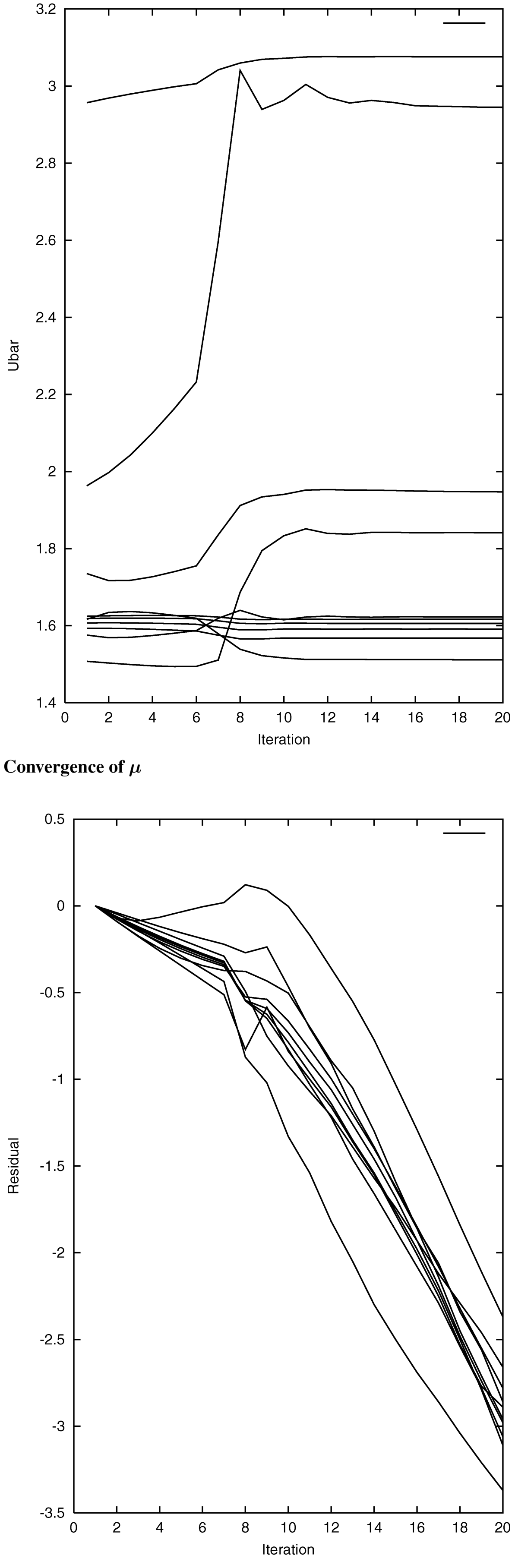


Fig. 7 Comparison of stability boundaries for the light case on the coarse and medium grids.

out. First, the convergence of the time histories with regard to the pseudotime-stepping tolerance was examined, and it was found that the residual had to be reduced by three orders of magnitude at each real time step, leading to between six and eight pseudoiterations. Second, a time-step convergence study was carried out, and again to achieve a converged prediction of the growth of the response, a time step of 0.125, corresponding to about 120 real time steps per pitching cycle, was required. It is considered that there is little scope for speeding up the time-marching calculations using the current solver because the number of time steps required is fixed by accuracy requirements and not solver requirements such as stability. For each Mach number, at least three time-marching calculations are required to locate the flutter speed, and several more would be required to locate the value to five significant figures. The behavior of the time-marching responses at conditions chosen to straddle the stability boundary are compared with the direct boundary in Fig. 9 on the medium grid and are in close agreement.

The heavy case proved more challenging for the augmented solver. This case has stability up to larger values of μ but also has two regions of high gradients in the transonic region of the μ–Mach stability curve. The initial calculations on the coarse grid successfully traced the curve over the entire Mach range and then for refined resolution in the transonic range, as for the light case. However, the augmented calculations on the medium grid diverged at the two values of Mach number (0.83 and 0.89) with maximum change in μ. The solution to this was to calculate the three regions separately, starting from information obtained on the coarse grid. The agreement between the calculations on the two grids is shown in Fig. 10 and again is close. The comparison with selected time-marching calculations is shown in Fig. 11 and again shows consistency. The costs of the calculations are as for the light case.



a) Convergence of μ

b) Convergence of residual

Fig. 8 Convergence at different Mach numbers for the light case on the medium grid.

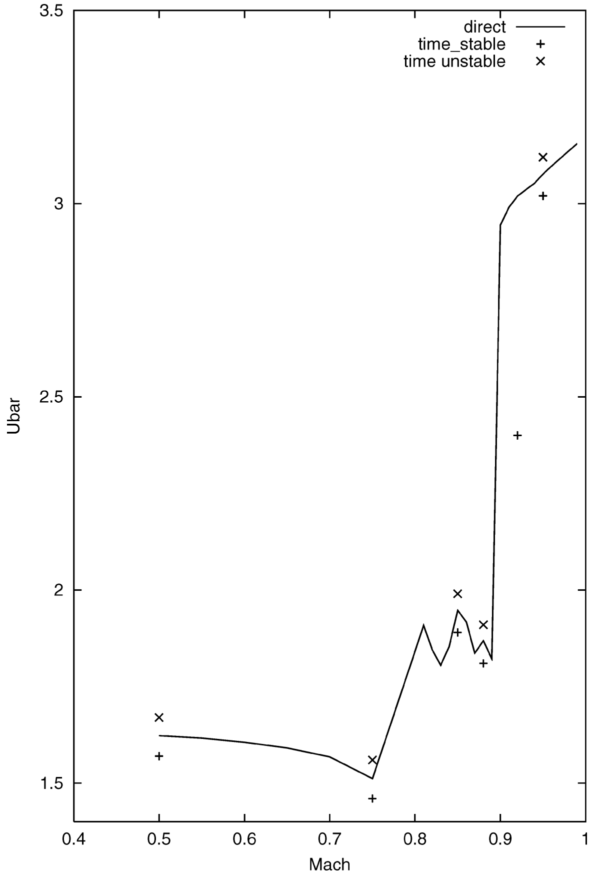


Fig. 9 Comparison of stability boundary for the light case on the medium grids with time-marching results.

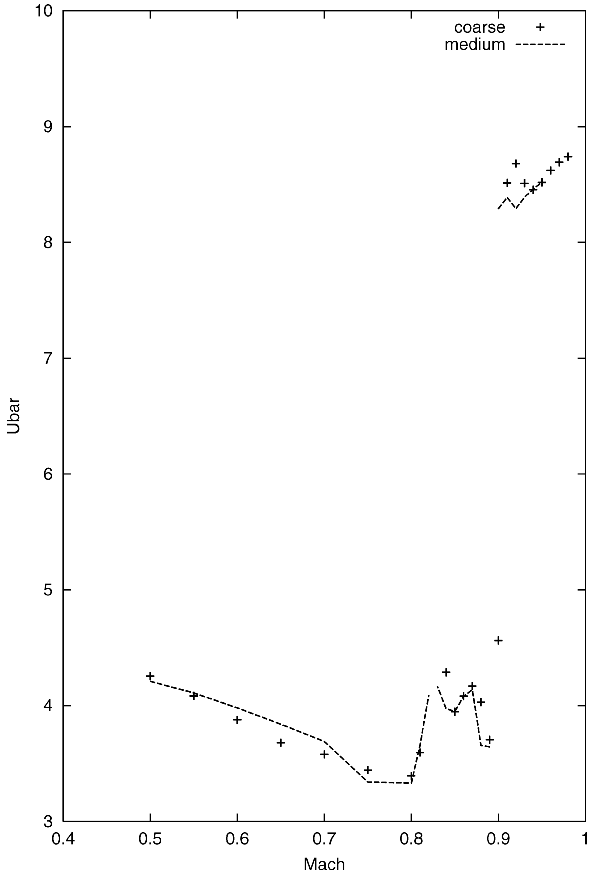


Fig. 10 Comparison of stability boundaries for the heavy case on the coarse and medium grids.

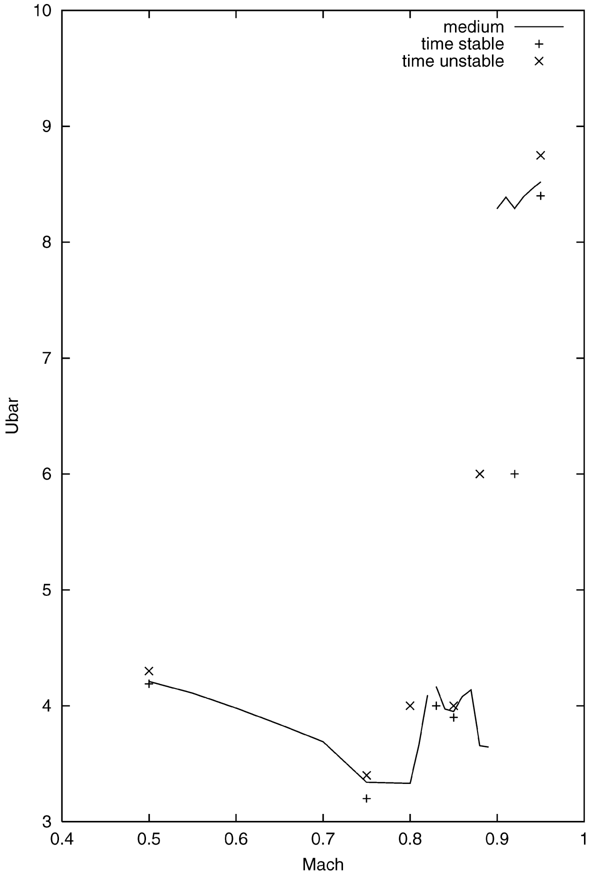


Fig. 11 Comparison of stability boundary for the heavy case on the medium grids with time-marching results.

VIII. Conclusions

A new iteration scheme for the direct calculation of aeroelastic instability boundaries has been proposed. The scheme builds on the original work of Morton and Beran by, first, using an iterative sparse linear solver to improve on the cost of direct methods and, second, approximating the Jacobian matrix in the iteration scheme without overly disrupting the convergence or robustness of the scheme. To improve robustness the inverse power method is used to obtain a starting solution for the critical eigenvector.

The method has been tested on a symmetric pitch–plunge airfoil problem. The stability boundary at zero incidence and 10 Mach numbers on the medium grid was traced out by the direct scheme in less than 1 h on a 1-GHz processor. There is scope for reducing this cost by relaxing the convergence criteria and by improving the preconditioning. In any case, the method already only requires a time to calculate the stability boundary at each Mach number, which is similar to a steady-state CFD calculation. The whole boundary defined at 25 Mach numbers requires the time needed for about three to four time-marching calculations. There appears to be little scope for reducing the cost of the time-marching calculations further.

Some difficulties were encountered with the basic continuation strategy used, which did not allow different branches on the stability curve to be traced automatically for the heavy case. With use of information from the coarse grid, the different braches of the solution were traced separately. Some work is required on this problem. However, it would be a lengthy business to map out the curve in all its detail using time marching.

The method has been developed with a view to generalization. First, building the CFD–CSD equation into the iteration loop to compute nonsymmetric problems is not likely to contribute greatly to the cost. The simplest approach is to iterate between the equilibrium calculation that provides a Jacobian matrix and the direct solver that provides the bifurcation parameter. The cost of this, especially because the previous equilibrium can be used to restart the coupled static solution and the previous critical eigenvectors to restart the direct solution, is likely to be low. Second, incorporating a grid movement technique to account for deforming geometries is a small modification, which has already been done based on transfinite interpolation. The extra Jacobian terms arising from the dependence of the fluid residual on the structural solution through the mesh deformation can be calculated by a combination of analytical terms and finite differences, although it has not been necessary to exploit this in the current paper. Finally, the Krylov linear solver techniques are practical for three-dimensional problems, and this extension will be reported in a future paper.

Acknowledgments

This work was supported by the Engineering and Physical Sciences Research Council, the Ministry of Defence, the Defence Evaluation and Research Agency, and BAE Systems.

References

[1] Goura, G. S. L., "Time Marching Analysis of Flutter Using Computational Fluid Dynamics," Ph.D. Dissertation, Dept. of Aerospace Engineering, Univ. of Glasgow, Glasgow, Scotland, U.K., Nov. 2001.

[2] Lesoinne, M., and Farhat, C., "High Order Subiteration-Free Staggered Algorithm for Nonlinear Transient Aeroelastic Problems," *AIAA Journal*, Vol. 36, No. 9, 1998, pp. 1754–1757.

[3] Djayapertapa, L., "A Computational Method for Coupled Aerodynamic–Structural Calculations in Unsteady Transonic Flow with Active Control Study," Ph.D. Dissertation, Dept. of Aerospace Engineering, Univ. of Bristol, Bristol, England, U.K., Nov. 2000.

[4] Farhat, C., Geuzaine, P., and Brown. G., "Application of a Three-Field Nonlinear Fluid–Structure Formulation to the Prediction of the Aeroelastic Parameters of an F-16 Fighter," *Computers and Fluids*, Vol. 32, No. 1, 2003, pp. 3–29.

[5] Melville, R, "Nonlinear Simulation of F-16 Aeroelastic Instability," AIAA Paper 2001-0570, Jan. 2001.

[6] Silva, W. A., Beran, P. S., Cesnik, C. E. S., Guendel, R. E., Kurdila, A., Prazenica, R. J., Librescu, L., Marzocca, P., and Raveh, D., "Reduced Order Modelling: Cooperative Research and Development at the NASA Langley Research Center," *Proceedings: International Forum for Aeroelasticity and Structural Dynamics*, Vol. 2, Asociacion de Ingeneros Aeronauticos de Espania, Madrid, 2001, pp. 159–174.

[7] Beran, P. S., and Carlson, C. D., "Domain-Decomposition Methods for Bifurcation Analysis," AIAA Paper 97-0518, 1997.

[8] Morton, S. A., and Beran, P. S., "Hopf-Bifurcation Analysis of Airfoil Flutter at Transonic Speeds," *Journal of Aircraft*, Vol. 36, 1999, pp. 421–429.

[9] Beran, P. S., "A Domain-Decomposition Method for Airfoil Flutter Analysis," AIAA Paper 98-0098, 1998.

[10] Badcock, K. J., Richards, B. E., and Woodgate, M. A., "Elements of Computational Fluid Dynamics on Block Structured Grids Using Implicit Solvers," *Progress in Aerospace Sciences*, Vol. 36, 2000, pp. 351–392.

[11] Osher, S., and Chakravarthy, S. R., "Upwind Schemes and Boundary Conditions with Applications to Euler Equations in General Coordinates," *Journal of Computational Physics*, Vol. 50 , 1983, pp. 447–481.

[12] Van Leer, B., "Towards the Ultimate Conservative Conservative Difference Scheme II: Monotonicity and Conservation Combined in a Second Order Scheme," *Journal of Computational Physics*, Vol. 14, 1974, pp. 361–374.

[13] Cantariti, F., Dubuc, L., Gribben, B., Woodgate, M., Badcock, K., and Richards, B., "Approximate Jacobians for the Solution of the Euler and Navier–Stokes Equations," Aerospace Engineering Rept. 9704, Univ. of Glasgow, Glasgow, Scotland, U.K., March 1997.

[14] Jameson, A., "Time Dependent Calculations Using Multigrid, with Applications to Unsteady Flows past Airfoils and Wings," AIAA Paper 91-1596, 1991.

[15] Goura, G. S. L., Badcock, K. J., Woodgate, M. A., and Richards, B. E., "Implicit Method for the Time Marching Analysis of Flutter," *Aeronautical Journal*, Vol. 105, No. 1046, 2001, pp. 199–214.

[16] Seydel, R., *Practical Bifurcation Analysis and Stability Analysis*, 2nd ed., Springer-Verlag, Berlin, 1994.

[17] Cantariti, F., Dubuc, L., Gribben, B., Woodgate, M., Badcock, K. J., and Richards, B. E., "Approximate Jacobians for the Solution of the Euler and Navier–Stokes Equations," Glasgow Univ., Aerospace Engineering Rept. 5, Glasgow, Scotland, U.K., March 1997.

[18] Bischof, C., Carle, A., Corliss, G., Griewank, A., and Hovland, P., "ADIFOR: Generating Derivative Codes from Fortran Programs," *Scientific Programming*, Vol. 1, IOS Press, Amsterdam, 1992, pp. 11–29.

[19] Badcock, K. J., Xu, X., Dubuc, L., and Richards, B. E., "Preconditioners for High Speed Flows in Aerospace Engineering," *Numerical Methods for Fluid Dynamics*, Vol. 5, Oxford Univ. Press, Oxford, 1996, pp. 287–294.

[20] Axelsson, O., *Iterative Solution Methods*, Cambridge Univ. Press, Cambridge, England, U.K., 1994.

[21] Tuminaro, R. S., Heroux, M., Hutchinson, S. A., and Shahid, J. N., "Official Aztec User's Guide," Ver. 2.1, Sandia Lab., Rept. SAND99-8801J, Albuquerque, NM, 1999.

[22] Burden, R. L., and Faires, J. D., *Numerical Analysis*, Prindle, Weber, and Schmidt, Boston, 1985, Chap. 8.

E. Livne
Associate Editor