A93-33971

AIAA-93-1416-CP

Mode Tracking Issues in Optimization

M. S. Eldred*
The University of Michigan, Ann Arbor, MI 48109

V.B. Venkayya†
Wright-Patterson Air Force Base, Dayton, OH 45433

W. J. Anderson‡
The University of Michigan, Ann Arbor, MI 48109

Abstract

Within the context of optimization of finite element models, methodology is developed for the tracking of eigenpairs during perturbations in the eigenproblem, for both self-adjoint and nonself-adjoint cases. One goal is to eliminate difficulties caused by "mode-switching" (i.e. frequency crossing). Out of several candidate methods, two methods for mode tracking are successful. The first method, the higher order eigenpair perturbation algorithm, is based on a perturbation expansion of the eigenproblem. It replaces the reanalysis step of an optimization routine with the important feature of maintaining the correspondence between the baseline and perturbed eigenpairs. The second method is a cross-orthogonality check method which uses mass orthogonality to reestablish correspondence after a standard reanalysis. Modified eigenpair extraction routines (Lanczos, subspace iteration, inverse power) were unsuccessful in tracking modes. Applications of mode tracking technology that are presented are frequency-constrained optimization, optimization with mode shape constraints, and V-g flutter analysis. Each application procedure is outlined and examples are given. Recommendations are made based on method speed and robustness in the example problems.

Nomenclature

$[A]$ = aerodynamic matrix from the doublet lattice method; complex,nonsymmetric

b = reference semi-chord of a wing

$[C]$ = cross-orthogonality check matrix

c = weighting factor determining the contribution of the homogeneous soln. to the total soln. for the eigenvector perturbation

$[D]$ = coefficient matrix in the eigenvector perturbation solutions; singular

$\{F\}$ = static pseudo-load vector appearing in eigenvector perturbation calculations

g = artificial structural damping in V-g flutter analysis

$[K]$ = stiffness matrix; symmetric

$[k]$ = modal stiffness matrix; diagonal

k = reduced frequency ($\frac{\omega b}{V}$)

$[M]$ = mass matrix; symmetric

$[m]$ = modal mass matrix; diagonal

$\{V\}$ = particular solution for eigenvector perturbation calculated via Nelson's method

V = airspeed

$\{z\}$ = flutter mode of normal mode participation factors; eigenvector in V-g analysis ($[k]\{z_R\}_i = \lambda_i([m]+[A])\{z_R\}_i$)

$[Z]$ = matrix of $\{z\}$ column vectors

A = perturbation symbol denoting exact change from a reference

λ = eigenvalue (real in self-adjoint problem, complex in nonself-adjoint problem)

ξ = collection of terms in solution for c_i

$\{\phi\}$ = normal mode shape; eigenvector in structural eigenproblem ($[K]\{\phi\}_i = \lambda_i[M]\{\phi\}_i$); normalized with respect to appropriate $[M]$

‘Graduate Research Assistant, Dept. of Aerospace Engineering, Student Member AIAA

†Principal Scientist, Analysis and Optimization Branch, Associate Fellow AIAA

Professor of Aerospace Engineering, Senior Member AIAA

[(a] = matrix of $\{\phi\}$ column vectors
ω = circular frequency of harmonic oscillation in V-g flutter analysis

Subscripts
i = associated with the i^{th} eigenpair
L = associated with the left eigenvector in nonself-adjoint problems
R = associated with the right eigenvector in nonself-adjoint problems

Superscripts
(k) = optimization iteration number
T = standard transpose, hermitian transpose denoted by $\overline{(\ )}^T$
0 = baseline values
1 = perturbed values resultant from parameter change, e.g. $(\ )^1 = (\ )^0 + \Delta(\ )$
$*$ = a user specified quantity
$\sim$ = denotes error from a specified value

Introduction

As optimization technology has matured, the range of disciplines that can appear in an optimization process has increased dramatically. Numerous codes are now capable of complex multi-disciplinary optimization. In ASTROS (Automated Structural Optimization System), for example, constraints can be applied on stress, displacements, buckling loads, natural frequencies, and static and dynamic aeroelastic constraints. In optimization for free vibration characteristics, previous work has been dominated by the frequency-constrained problem. Eigenvector control is a relatively new technology and can appear as a stand-alone constraint on the optimization. In both frequency-constrained and mode shape-constrained optimization, mode tracking is an important technology which allows for proper bookkeeping on constrained frequencies and mode shapes.

When defining constraints on the dynamic characteristics of a structure, specific frequencies and modes must be referenced in some manner. In vibration problems, the eigenvalues and eigenvectors are ordered by eigenvalue magnitude. In aeroelastic optimization, flutter frequencies and modes are ordered either by critical airspeed or frequency (double eigenvalue problem). When design variable perturbations are performed, frequencies and critical airspeeds will drift and mode crossings can occur. Thus, it is possible for a constraint to be enforced on an entirely different mode than was intended. This causes problems both from the design standpoint, in that the optimum found will not reflect the design goals, and from the mathematical viewpoint, in that convergence can be destroyed.

Another concern when optimizing is that mode shapes can change drastically in character. Rather than smoothly varying with changes in design variables, mode shapes can jump to entirely different shapes. For example, a rectangular plate having modes with horizontal and vertical nodal lines might have modes with diagonal nodal lines after a design perturbation. In some design scenarios, it may be desirable to handle this type of behavior; in all cases it is desirable to detect these abrupt changes, even if no action will be taken.

Theory of Mode Tracking

Mode tracking algorithms perform an eigenproblem analysis with the added feature of maintaining correspondence between baseline and perturbed (current and projected) modes. Thus, they can be used to replace the reanalysis phase of an optimization algorithm. A typical flowchart for optimization with mode tracking is shown in Figure 1. The first step in structural optimization is the generation of a baseline model. After thorough analysis of the baseline model, criteria are developed for desired improvements and are implemented through a design objective and design constraints. Constraint and objective function sensitivities are then calculated and used to find the design space search direction. After the optimal projection along the search path has been found, an optimization algorithm must

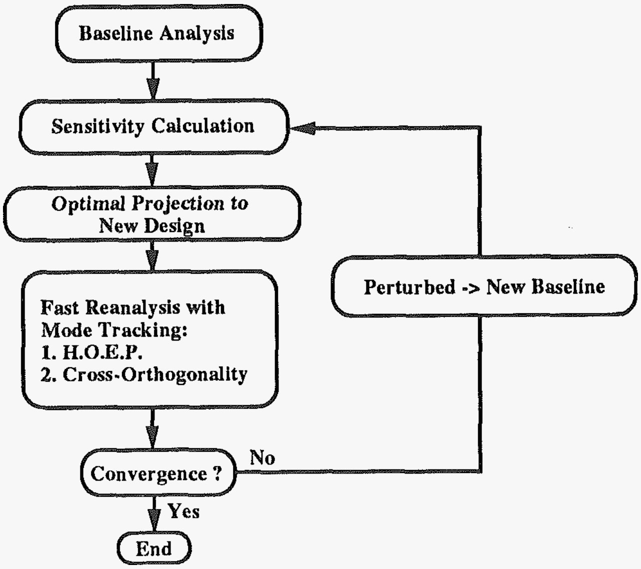

Figure 1: Optimization algorithm with mode tracking

reanalyze the new design configuration in order to evaluate the new objective function and constraints. The mode tracking algorithms perform this reanalysis with the added feature of retaining parent/offspring data for the modes throughout the optimization iterations. Two successful methods are proposed for use in mode tracking for both self-adjoint and nonself-adjoint problems, and one unsuccessful class of methods is briefly discussed. One successful method, the higher order eigenpair perturbation algorithm, is based on perturbation expansions of the eigenproblems. The unsuccessful method is the modified eigenpair extraction routine, of which three cases are discussed. The second successful method is the cross-orthogonality check method, which uses mass orthogonality to reestablish correspondence after a standard reanalysis.

Self-Adjoint Eigenvalue Problems

Higher order eigenpair perturbations (HOEP)

This method [1] performs a perturbation expansion on the standard undamped structural eigenproblem:

$$[K]\{\phi\}_i = \lambda_i [M]\{\phi\}_i \tag{1}$$

All perturbation terms are retained, which leads to coupled equations for $\Delta\lambda_i$ and $\{\Delta\phi\}_i$. These equations, which follow, must be solved iteratively and will converge *to* the exact eigenpair perturbations.

The full-order eigenvalue perturbation equation is:

$$\Delta\lambda_i = \frac{\{\phi^0\}_i^T([\Delta K] - \lambda_i^0[\Delta M])\{\phi^1\}_i}{\{\phi^0\}_i^T[M^1]\{\phi^1\}_i} \tag{2}$$

In the context of a reanalysis phase in an optimization process, the 0-superscript denotes the current design and the l-superscript denotes the projected design. Unlike linear sensitivity methods, in which only the current eigenvector appears, Eq. **2** shows coupling with the projected eigenvector.

The corresponding eigenvector perturbation equation is

$$[D^1]_i\{\Delta\phi\}_i = \{F^{nl}\}_i \tag{3}$$

where

$$[D^1]_i \equiv [K^1] - \lambda_i^1[M^1] \tag{4}$$

is singular, and

$$\{F^{nl}\}_i \equiv (\Delta\lambda_i[M^1] + \lambda_i^0[\Delta M] - [\Delta K])\{\phi^0\}_i \tag{5}$$

is a static pseudo-load. Equation **3** corresponds to the pathological Fredholm alternative in which the coefficient matrix is singular and there is a non-zero load. Such equations cannot be solved in general. This equation is solvable, however, since it is "consistent," i.e. $\{F^{nl}\}_i$ is orthogonal to $\{\phi^1\}_i$.

The total solution for $\{\Delta\phi\}_i$ is made up of homogeneous and particular solutions. For singular $[D^1]_i$, $\{\phi^1\}_i$ is a homogeneous solution for $\{\Delta\phi\}_i$ in Eq. **3** since $[D^1]_i\{\phi^1\}_i = \{0\}$. The total solution for $\{\Delta\phi\}_i$ is then a sum of the particular solution $\{V\}_i$ and a weighted $\{\phi^1\}_i$. The weighting factor c_i is introduced because the scaling of the homogeneous solution is initially indeterminate :

$$\{\Delta\phi\}_i = c_i\{\phi^1\}_i + \{V\}_i \tag{6}$$

Equation 6 must be altered since $\{\phi^1\}_i$ is unknown:

$$\begin{aligned} \{\Delta\phi\}_i &= c_i(\{\phi^0\}_i + \{\Delta\phi\}_i) + \{V\}_i && (7)\\ \{\Delta\phi\}_i &= \frac{c_i}{1-c_i}\{\phi^0\}_i + \frac{1}{1-c_i}\{V\}_i && (8)\end{aligned}$$

Employing conditions of mass normalization for the current and projected systems and substituting Eq. 8 for $\{\Delta\phi\}_i$ yields a quadratic equation for c_i. The desired root is:

$$c_i = 1 - \sqrt{1+\xi_i} \tag{9}$$

where

$$\begin{aligned}\xi_i = \ & \{\phi^0\}_i^T[\Delta M]\{\phi^0\}_i + 2\{\phi^0\}_i^T[M^1]\{V\}_i \\ & +\{V\}_i^T[M^1]\{V\}_i && (10)\end{aligned}$$

The algorithm has evolved considerably since its original publication [1]. It has been greatly simplified and its convergence characteristics have been improved. The current algorithm flowchart is shown in Figure **2.** Iteration 0 consists of obtaining an initial estimate of the eigenvalue perturbations from a first order approximation to Eq. **2.** The nonlinear iterations are then performed, which consist of solution of the almost singular eigenvector perturbation equation (Eq. **3)** followed by the full-order update for the eigenvalue perturbation (Eq. 2). Solution for the eigenvector perturbation requires the computation of the particular solution *{V};*of Eq. **3** by Nelson's method [2], calculation of c_i from Eqs. 9 and 10, and finally solution of Eq. 8 for $\{\Delta\phi\}_i$. These iterations continue until the convergence criterion is satisfied. The important feature for mode tracking is the fact that the computations involve data from the i^{th} eigenpair only. There is a direct correspondence between current and projected data and frequency crossings are of no concern. The method is computationally intensive, but can converge for very large prescribed changes in design variables.

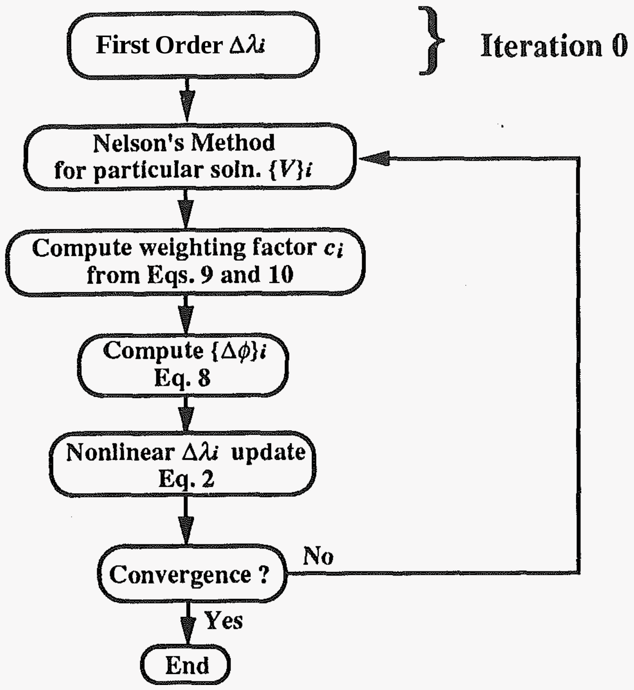
First Order $\Delta\lambda i$
Iteration 0
Nelson's Method for particular soln. $\{V\}i$
Compute weighting factor c_i from Eqs. 9 and 10
Compute $\{\Delta\phi\}i$ Eq. 8
Nonlinear $\Delta\lambda i$ update Eq. 2
Convergence ?
No
Yes
End

Figure 2: HOEP algorithm

Modified eigenpair extraction routines

The block Lanczos, subspace iteration, and inverse power eigenpair extraction routines were identified as having potential for mode tracking. In all cases, the idea was to preserve the baseline eigenpair ordering by using the previous iteration eigenpair data for trial vectors and shift values. In standard usage, random trial vectors are generated, shift values are widely spaced, and the converged eigenpairs are returned in ascending eigenvalue order. This reordering destroys the one-to-one correspondence if mode-switching has occurred. The goal of the modifications was to eliminate the reordering by altering the convergence characteristics of the extraction methods. In all cases, however, the ability to track modes was not achieved.

Modification of the block Lanczos method [3, 4] was eliminated from consideration because the iterates in the Lanczos recursion, the Lanczos vectors, are not directly related to the eigenvectors and therefore would not respond favorably to the use of baseline eigenvectors as trial vectors.

The subspace iteration algorithm is an eigenpair extraction routine which, unlike the Lanczos method, iterates directly on eigenvectors (see [5]). The modifications fail in the task of mode tracking because convergence in the subspace iteration method occurs according to eigenvalue magnitude and not according to eigenvector similarity to trial vectors. While convergence **is** improved when using the baseline eigenvectors since the starting subspace is an excellent approximation to the least-dominant (i.e. converged) subspace, convergence still occurs in eigenvalue order due to the minimization of the Rayleigh quotient inherent in the method. Shifting was also ruled out since it would be inefficient to shift on each baseline eigenvalue and, more importantly, since the procedure would not be foolproof. It is easy to envision scenarios where the closest eigenvalue to a shift is not the correct eigenvalue. This could occur for closely spaced eigenvalues or for frequency crossings, the exact cases of the most concern in this paper.

The inverse power method with shifting and sweeping is the final candidate extraction routine. This method was also unsuccessful, for the same reasons that subspace iteration was not successful, i.e. convergence occurs on eigenvalue magnitude rather than trial vector similarity and shifting is neither fail-safe nor efficient.

Thus, while some of the modified eigenpair extraction methods could be used for fast reanalysis due to their improved convergence speed, none of the methods exhibit a reliable ability to track modes.

*Cross-orthogonality **check** (CORC)*

This method, proposed by Gibson [6], performs a mass orthogonality check after reanalysis. The orthogonality information is held in the following mass triple product:

$$[C] = [\Phi^{(k-1)}]^T[M^{(k)}][\Phi^{(k)}] \tag{11}$$

where k and $k-1$ are the current and previous iterations, respectively. If the [C] matrix is diagonally dominant, then no mode-switching has occurred; and if the matrix is not diagonally dominant, the locations of the dominant values can be used to recorrelate the current iteration modes. The method does not attempt to directly track modes, but rather tries to reestablish correspondence after the changes in design and mode-switches have occurred. While attractive due to its simplicity, the method is unattractive in that the standard reanalysis does not make use of available baseline information and, in a sense, starts from scratch. This method could be combined with one of the modified eigenpair extraction routines. For example, the subspace iteration algorithm with baseline eigenvector trial vectors would enable an efficient reanalysis, and the cross-orthogonality check method would perform the mode tracking.

There are many other methods that can help identify *or* correlate modes, which appear in the system identification and model correlation literature ([7, 8] for example). The general spirit of these methods is similar to the cross-orthogonality check method and will not be reviewed here.

Nonself-Adioint Eigenvalue Problems

Complex higher **order** *eigenpair perturbations (C-HOEP)*

This method is an extension of [1] and performs a perturbation expansion on the nonself-adjoint eigenproblem for which the associated eigenpairs are complex. The approach is general and will be illustrated by way of a specific example — perturbation of the reduced frequency in the V-g flutter analysis modal equation:

$$[(1+ig)[k]-\omega^2([m]+[A])]\{z_R\}=\{0\} \qquad (12)$$

where *[A]*is the complex, non-symmetric aerodynamic matrix derived from the doublet lattice method [9, 10], **g** is the artificial structural damping necessary to sustain harmonic oscillation (which is required for the unsteady aerodynamics), *w* is the frequency of the harmonic oscillation, $[k]$ and *[m]*are diagonal modal stiffness and mass respectively, and $\{z_R\}$ is the right flutter eigenvector made up of complex normal mode participation factors. This equation can be rewritten:

$$[[k]-\lambda([m]+[A])]\{z_R\}=\{0\} \qquad (13)$$

where

$$\lambda=\frac{\omega^2}{1+ig} \qquad (14)$$

is called the complex eigenvalue. The associated left eigenproblem is:

$$\overline{\{z_L\}}^T[[k]-\lambda([m]+[A])]=\{0\} \qquad (15)$$

Perturbing the reduced frequency k affects the aerodynamic matrix and the eigenpairs in Eq. **13,** giving the following equation written for the ith eigenpair:

$$[[k]-(\lambda_i^0+\Delta\lambda_i)([m]+[A^0]+[\Delta A])](\{z_R^0\}_i+\{\Delta z_R\}_i)=\{0\} \qquad (16)$$

As before, all perturbation terms are retained, leading to coupled equations for $\Delta\lambda_i$ and $\{\Delta z_R\}_i$. The full-order eigenvalue perturbation equation has two forms:

$$\Delta\lambda_i=\frac{\overline{\{z_L^0\}}_i^T[\Delta A]\{z_R^1\}_i}{\overline{\{z_L^0\}}_i^T([m]+[A^1])\{z_R^1\}_i} \qquad (17)$$

or

$$\Delta\lambda_i=\frac{\overline{\{z_R^0\}}_i^T([k]-\lambda_i^0([m]+[A^1]))\{\Delta z_R\}_i-\lambda_i^0\overline{\{z_R^0\}}_i^T[\Delta A]\{z_R^0\}_i}{\overline{\{z_R^0\}}_i^T([m]+[A^1])\{z_R^1\}_i} \qquad (18)$$

depending on whether *or* not the left eigenvector data is computed at each optimization iteration. If the left eigenvector data is available, Eq. **17** is preferable to Eq. **18** due to its convergence characteristics (see example problem in Applications section). Also, the conjugate transpose appears on $\{z_R^0\}_i$ in Eq. **18** because it has been numerically determined to have better convergence characteristics than the standard transpose.

The corresponding eigenvector perturbation equation is

$$[D^1]_i\{\Delta z_R\}_i=\{F_R\}_i \qquad (19)$$

where

$$[D^1]_i \equiv [k]-\lambda_i^1([m]+[A^1]) \qquad (20)$$

is singular, and

$$\{F_R\}_i \equiv (\lambda_i^0[\Delta A]+\Delta\lambda_i([m]+[A^1]))\{z_R^0\}_i \qquad (21)$$

is a static pseudo-load. The equation is consistent since $\{F_R\}_i$ is orthogonal to $\{z_L^1\}_i$. Likewise, the eigenvector perturbation equation for $\{\Delta z_L\}_i$ (not shown) has its pseudo-load vector orthogonal to $\{z_R^1\}_i$. This biorthogonality of the pseudo-load vectors is true for nonself-adjoint problems in general, and the perturbation equations are solvable since they are consistent.

As in the self-adjoint case, the total solution for $\{\Delta z_R\}_i$ is made up of homogeneous and particular solutions. Since $[D^1]_i\{z_R^1\}_i=\{0\}$, $\{z_R^1\}_i$ is a homogeneous solution for $\{\Delta z_R\}_i$ in Eq. **19.** The total solution for $\{\Delta z_R\}_i$ is then a sum of the particular solution $\{V_R\}_i$ and a weighted $\{z_R^1\}_i$:

$$\{\Delta z_R\}_i=c_i\{z_R^1\}_i+\{V_R\}_i \qquad (22)$$

Equation **22** must be altered since $\{z_R^1\}_i$ is unknown:

$$\{\Delta z_R\}_i=\frac{c_i}{1-c_i}\{z_R^0\}_i+\frac{1}{1-c_i}\{V_R\}_i \qquad (23)$$

Employing conditions of magnitude normalization for the current and projected systems

$$\overline{\{z_R^0\}}_i^T\{z_R^0\}_i=\overline{\{z_R^1\}}_i^T\{z_R^1\}_i=1 \qquad (24)$$

and substituting Eq. 23 for $\{\Delta z_R\}_i$ yields the following equation for c_i:

$$c_i\overline{c_i}-c_i-\overline{c_i}=\xi_i \qquad (25)$$

where

$$\xi_i = \overline{\{z_R^0\}}_i^T \{V_R\}_i + \overline{\{V_R\}}_i^T \{z_R^0\}_i + \overline{\{V_R\}}_i^T \{V_R\}_i \quad (26)$$

is a real number. Equation **25** is a real equation in two unknowns, because the weighting factor c_i can be complex in general. An additional condition must be imposed since there are more unknowns than equations. Two approaches can be taken. The first approach is to enforce a phase correction condition on each C-HOEP iteration, that is, use c_i to rotate the homogeneous solution in Eq. 22 such that the phase of the perturbed eigenvector satisfies the phase correction condition. This approach is somewhat involved, and as it turns out, unnecessary. The second approach is to take the phase of c_i to be zero, equivalent to assuming that c_i is real. In this case, the homogeneous solution is not rotated, and the phase correction condition is not enforced on each C-HOEP iteration. Instead, the phase of the perturbed eigenvectors is corrected after convergence. This approach is simpler, faster, and more robust.

For real c_i, the desired root, as before in Eq. 9, is:

$$c_i = 1 - \sqrt{1 + \xi_i} \quad (27)$$

where ξ_i is defined by **Eq.** 26.

After convergence has occurred, the phase of the complex eigenvector must be corrected, since it is initially arbitrary. This phase correction allows for a unique complex eigenvector, which is mandatory if optimization constraints are to be placed upon it. The phase correction used is to force the maximum magnitude component to have zero phase by multiplying the complex eigenvector by a unit magnitude scalar of opposite phase. This correction is unique despite the phase discontinuity at $\pm\pi$. It is interesting to note that the length of the complex eigenvector is very important and it must be normalized on each C-HOEP iteration (Eqs. 22 through 27), whereas the phase is relatively unimportant and only needs to be corrected once (at convergence).

Computation of the left eigenvector perturbations proceeds identically as for the right eigenvector perturbations described above. The left eigenvectors will be required if Eq. 17 is to be used in an incremental process. The same inverted $[D^1]_i$ matrix (modified by the singularity removal by Nelson's method) may be used in the computations for $\{\Delta z_L\}_i$ and $\{\Delta z_R\}_i$ so long as the same pivotal element is used for the left and right systems (this is the standard procedure; consult [2] for details of Nelson's method).

The algorithm flowchart for C-HOEP is basically the same as that for HOEP and is shown in Figure **3.** Iteration 0 consists of obtaining an initial estimate of the eigenvalue perturbations from a first order approximation to Eq. 17 or 18. The nonlinear iterations are then performed, which consist of solution of the approximately singular eigenvector perturbation equation (Eq. 19) followed by the full-order update for the eigenvalue perturbation (Eq. 17 or 18). Solution for the eigenvector perturbation requires the computation of the particular solution $\{V_R\}_i$ of Eq. 19 by Nelson's method, calculation of c_i from Eqs. 26 and 27, and finally solution of Eq. 23 for $\{\Delta z_R\}_i$. These iterations continue until the convergence criterion is satisfied. As in the self-adjoint case, mode tracking is possible because the computations involve data from the ith eigenpair only, creating a direct correspondence between current and projected data.

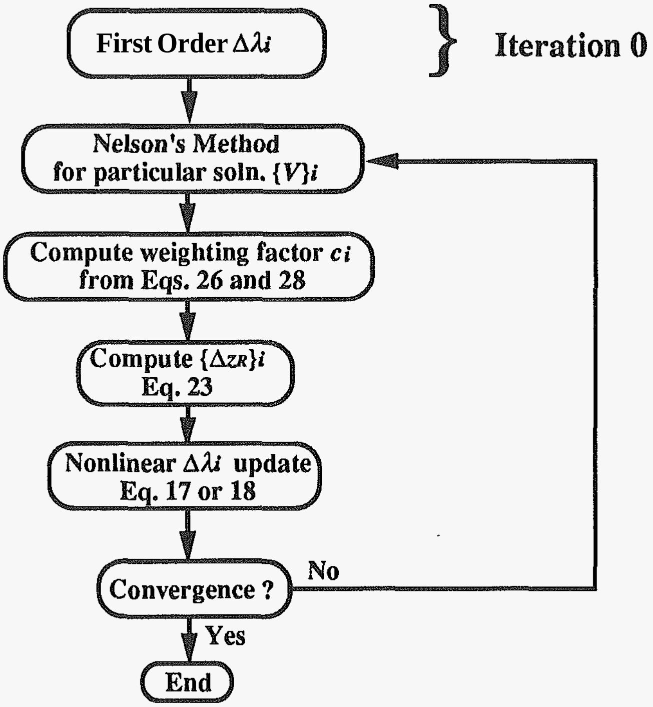
First Order $\Delta\lambda i$
Iteration 0
Nelson's Method
for particular soln. $\{V\}i$
Compute weighting factor ci
from Eqs. 26 and 28
Compute $\{\Delta z_R\}i$
Eq. 23
Nonlinear $\Delta\lambda i$ update
Eq. 17 or 18
Convergence ?
No
Yes
End

Figure **3:** C-HOEP algorithm

Complex cross-orthogonality ***check*** *(C-CORC)*

For the V-g flutter analysis problem, one must make use of the biorthogonality property to recorrelate the complex modes. The expression for the $[C]$ matrix is then:

$$[C] = \overline{[Z_L^{(k-1)}]}^T ([m^{(k)}] + [A^{(k)}])[Z_R^{(k)}] \quad (28)$$

where k and $k-1$ are the current and previous iterations, respectively. Obviously, this requires left eigenvector information which is not generally computed. As before, if the [C] matrix is diagonally dominant, then no mode-switching has occurred; and if the matrix is not diagonally dominant, the locations of the dominant values can be used to recorrelate the current iteration modes.

Applications

Optimization with Frequency Constraints

Problems of structural optimization subject to frequency constraints are commonly formulated in the following manner:

$$\begin{array}{ll} \text{minimize} & f(\{b\}) \\ \text{subject to} & g_j(\{b\}) = \lambda_i - \lambda_i^* \geq 0 \end{array} \qquad (29)$$

The linear sensitivity of the frequency constraint to a design variable is:

$$\frac{dg_j}{db_k} = \frac{d\lambda_i}{db_k} = \frac{\{\phi\}_i^T([\frac{dK}{db_k}] - \lambda_i[\frac{dM}{db_k}])\{\phi\}_i}{\{\phi\}_i^T[M]\{\phi\}_i} \qquad (30)$$

where the denominator is unity in the case of mass normalized eigenvectors.

Since the eigenvectors do not appear in the above problem formulation (Eq. 29), it is typical for the analyst to ignore the eigenvectors altogether. Even with modest design changes, this practice can fail due to the possibilities of mode-switches and drastic modal character changes. When either possibility occurs, the optimization iteration history can exhibit a jump since the constraint sensitivity is being calculated for a vastly different mode shape. The iterations may even oscillate unpredictably if the mode-switching is recurrent. Thus, even if frequency constraints are the only concern, mode tracking is an important technology.

Example 1: Simple Cantilevered Wing Box

A simple cantilevered wing box (Fig. **4** has been devised for use in optimization with vibration constraints. The ASTROS multi-disciplinary optimization code is being used with MAPOL (Matrix Programming Oriented Language) coding and FORTRAN modules performing all nonstandard tasks.

The simple wing box is to undergo weight minimization subject to frequency constraints. The third and fourth modes are closely spaced in the original design and have exhibited a tendency to switch.

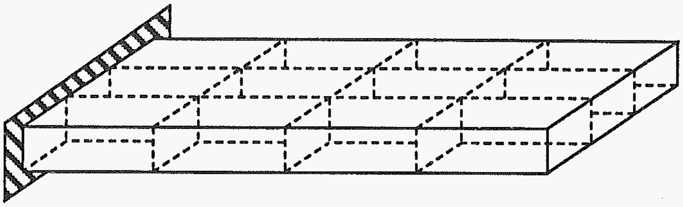

Figure **4**: Simple cantilevered wing box

The first optimization procedure is the standard reanalysis procedure implemented in ASTROS. When a lower-bound constraint is placed on ω_3 ($\omega_3 \geq 110$ Hz), optimization without mode tracking exhibits large oscillations (see Fig. 5). The modes first switch on iteration **3,** causing the constraint sensitivity to be calculated using the wrong mode shape and resulting in an erroneous projection to the iteration **4** design. The modes have switched back in the iteration **4** design and the optimization begins its nonconverging oscillations.

When mode tracking by the HOEP or CORC algorithms is inserted in place of the reanalysis routine, the mode-switch on iteration **3** is properly tracked and the correct sensitivities are calculated. Figure 5 shows convergence to the optimal design in 6 iterations.

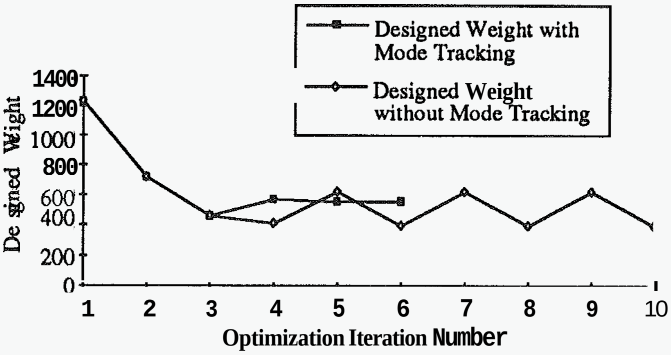

Figure 5: Weight minimization with a frequency constraint

Both HOEP and CORC are robust enough to handle the mode-switching in this example problem. More examples are to be presented at the conference. The question of efficiency depends on the method of reanalysis used prior to CORC. If all eigenpairs must be extracted prior to the orthogonality check, then HOEP has a definite advantage in tracking only the constrained eigenpairs.

Optimization with Mode Shape Constraints

The obvious extension of the frequency-constrained problem is to include constraints on the mode shapes. Typically, it may be desired to constrain a mode to have some specified shape. This could involve control of nodal lines to minimize vibration in a region of the structure. Or, it might involve constraining against an anticipated abrupt modal character change. While constraints of this type can be formulated in numerous ways due to the vector nature of a mode shape, the approach employed here will involve vector norms. If, for example, it is desired to keep the ith mode shape close to some specified shape, then the optimization problem can be formulated **as:**

$$\begin{array}{ll} \text{minimize} & f(\{b\}) \\ \text{subject to} & g_j = \mu_i^* - \mu_i \geq 0 \end{array} \tag{31}$$

where

$$\mu_i = ||\{\widetilde{\Delta\phi}\}_i||$$

and

$$\{\widetilde{\Delta\phi}\}_i = \{\phi\}_i - \{\phi^*\}_i$$

Here, $\{\phi\}_i$ is the current mode shape and $\{\phi^*\}_i$ is the specified mode shape. The μ_i^* is user specified to denote how close the current eigenvector must be to the prescribed eigenvector. In order to lead to a simpler sensitivity formula, the constraint can be reformulated with the square of the vector norm:

$$g_j = (\mu_i^*)^2 - ||\{\widetilde{\Delta\phi}\}_i||^2 \geq 0 \tag{32}$$

or

$$g_j = (\mu_i^*)^2 - \{\widetilde{\Delta\phi}\}_i^T\{\widetilde{\Delta\phi}\}_i \geq 0 \tag{33}$$

This does not alter the nature of the constraint. The sensitivity of this new constraint to a design variable is

$$\frac{\partial g_j}{\partial b} = -2\{\widetilde{\Delta\phi}\}_i^T \frac{d\{\phi\}_i}{db} \tag{34}$$

where the fact that

$$\frac{d\{\widetilde{\Delta\phi}\}_i}{db} = \frac{d\{\phi\}_i}{db}$$

has been used, since

$$\frac{d\{\phi^*\}_i}{db} = 0$$

Implementing this sensitivity (Eq. **34)** requires calculation of the eigenvector derivative $\frac{d\{\phi\}_i}{db}$ (by Nelson's method **[2]** since it is already coded for the HOEP algorithm). Current $\{\widetilde{\Delta\phi}\}_i$ information, the difference of the current and prescribed eigenvectors, is a known quantity and is available for the constraint calculation (unlike $\{\Delta\phi\}_i$, which is an unknown in the HOEP algorithm).

This constraint formulation is very general since any portion of an eigenvector can be prescribed. If only certain degrees of freedom are to be constrained, then only those freedoms appear in the vector calculations of Eqs. **33** and **34.** Common choices for the prescribed eigenvector would include the baseline mode shape, an experimental mode shape to be correlated, or a region of zero vibration (enforcing a nodal line).

Once again, mode tracking will be very important to a successful optimization. The vector norm constraint formulation will preclude any abrupt changes in modal character, but will be very susceptible to mode-switches. The same convergence problems could occur **as** in the frequency-constrained case if mode-switches are not properly tracked.

Example 1: Sample Cantilevered Wing Box

The same simple cantilevered wing box (Fig. **4)** will be used to illustrate optimization with eigenvector constraints.

Initial testing has been conducted to define "iteration-invariant" normalization and sign convention methods. The goal is to enable the eigenvector perturbation vector norms to measure changes in mode shape only, separate from changes in sign or normalization.

Mass normalization of all eigenvectors to the baseline mass matrix $[M^0]$ has been identified **as** the best invariant normalization method. This normalization deletes changes in scaling from the vector norms. Care must be taken in calculating the eigenvalue sensitivities from Eq. **30,** however, since the scaling of $\{\phi\}_i$ affects the sensitivity calculation and the subsequent optimal projection.

The best sign convention method involves making the maximum magnitude baseline eigenvector component greater than zero, and retaining the same component **as** positive in all subsequent eigenvectors. Through this use of the baseline eigenvectors **as** a reference, the sign convention is consistent for all iterations.

Figures 6 and 7 show a before and after view of iteration-invariance. Vector norms of eigenvector change measured from the baseline are shown for the seven lowest modes of the cantilever wing box. In Figure 6, changes in sign and normalization obscure the changes in mode shape. Figure 7 shows the effect of using an invariant sign convention and normalization. Pure changes in mode shape are plotted, and a mode-switch has become evident through the relatively large vector norms for the switched modes.

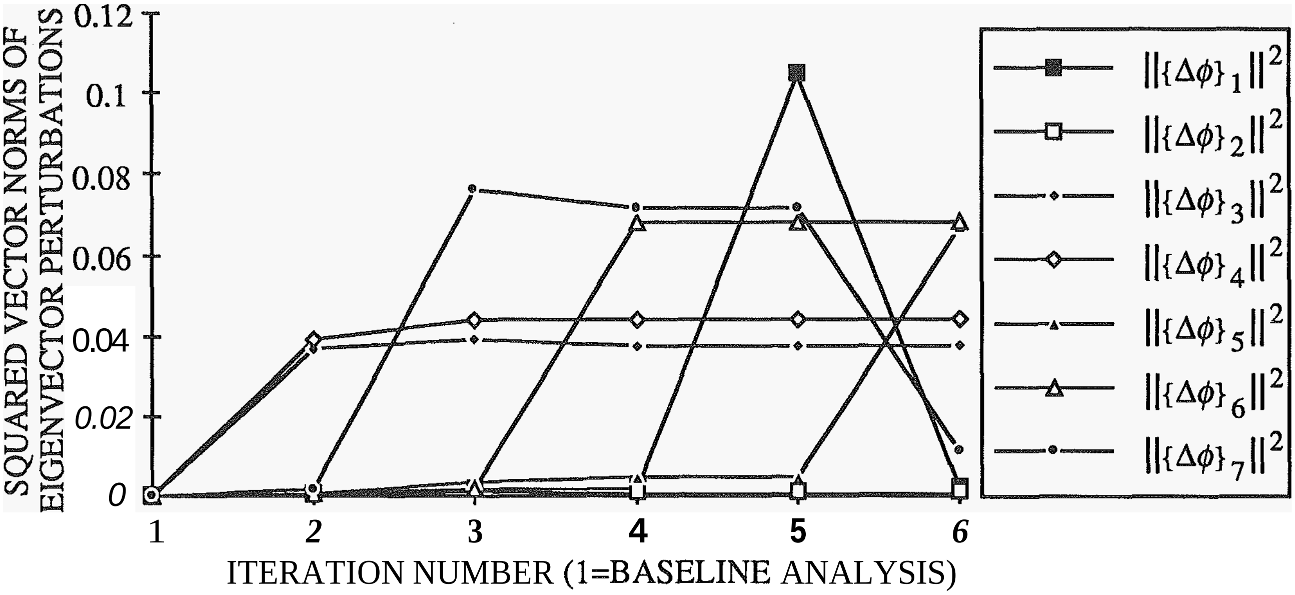

Figure 6: Before iteration-invariance

This concept of iteration invariance is easily extended to the complex case, e.g. a flutter mode. The length normalization is already iteration invariant since no mass operator is used (Eq. 24). The phase correction discussed previously is an extension of the sign convention for real eigenvectors. The phase correction is rendered iteration invariant by making the phase of the maximum magnitude baseline eigenvector component equal to zero, and retaining zero phase on the same component for all subsequent optimization iterations (not to be confused with C-HOEP iterations).

Examples of eigenvector control are not complete at printing time, but will be presented in the conference.

V-e: Flutter Mode Tracking

It is important to distinguish between two mode tracking needs in the areas of flutter analysis and optimization. First, in **V-g** type flutter analysis, pre-flutter modes are tracked as reduced frequency (k) is incremented. Typically, the analysis steps through k values, extracting complex eigenpairs on each step, until the desired flutter points are obtained (when the artificial damping required for harmonic oscillation is greater than that available from the structure). Second, in flutter optimization, critical flutter modes (ordered by airspeed) may switch order under a design change. This is a more difficult problem than the former, since each critical flutter mode denotes the result of a converged flutter analysis.

The former problem, pre-flutter mode tracking in flutter analysis, is handled as developed in the nonself-adjoint eigenvalue problems theory section. The following example shows an application of this technique.

Example 1: MATLAB pre-flutter made tracking for rectangular wing

The aerodynamic planform and doublet lattice discretization for the rectangular wing are shown in Figure 8. The underlying structure was kept **as** simple **as** possible and is an aluminum cantilever beam of rectangular cross section. **A** total of six normal modes are retained, three each of bending and torsion. For Mach 0.5 at sea level $(p = 1.225\frac{kg}{m^3})$, a V-g analysis for k varying from 1.0 to 0.0 was performed, resulting in Figures 9 and 10. The most important thing to note is apparent in Figure 10, where the modes are seen to veer away from one another. That is, frequency crossing is not a problem for this example, and simply ordering the modes by frequency magnitude is sufficient.

While frequency crossings may be rare for a clean wing (with modes that are strongly coupled by the aerodynamics), it is theorized that this will not be true for more complex configurations (where some modes will not be affected by the aerodynamic changes). For example, in a full airplane with stores, wing and store vibration modes may be weakly coupled by the aerodynamics and frequency crossings could easily oc-

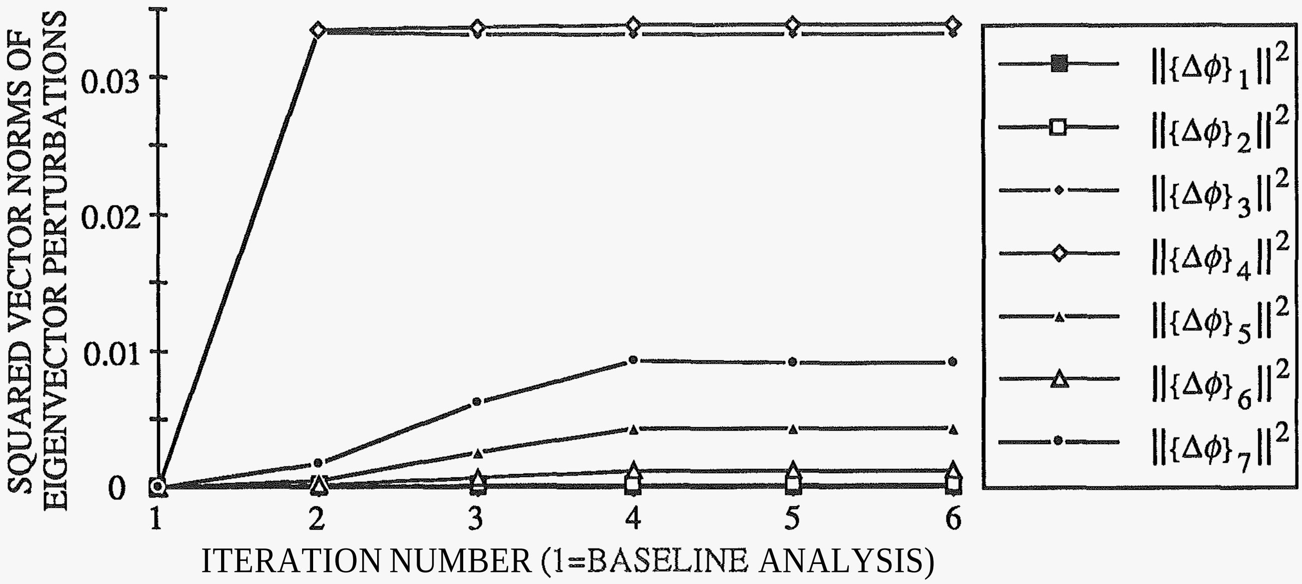

Figure 7: After iteration-invariance

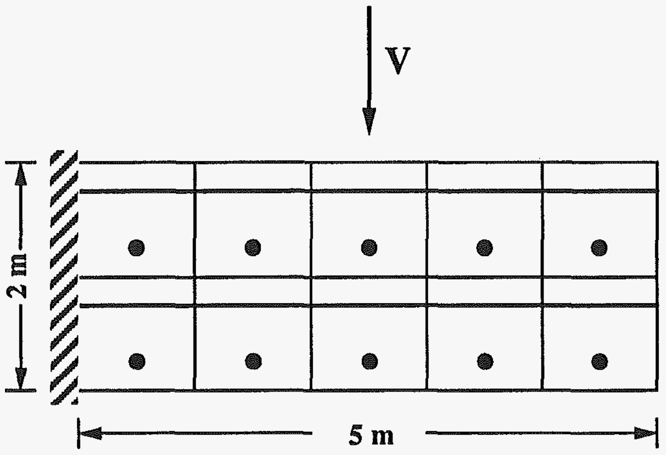

Figure 8: Rectangular wing with doublet lattice discretization

cur. Likewise, some wing/tail, wing/fuselage, and tail/fuselage interactions would be weakly coupled with the aerodynamics.

Performance of the mode tracking methods was evaluated for this example problem. The lack of frequency crossings allows evaluation of the mode tracking methods well past their region of convergence, since the proper mode identifications are known with assurance. In all cases, the initial value of k (the reduced frequency) is **1.0,** and the projected value is as shown. This is not an incremental process (no updates are made). The purpose here is to test the methods for increasing Ak.

In Table 1, performance of the C-HOEP algorithm is shown by virtue of the CPU time and iterations necessary to reach convergence, for analyses run without use of left eigenvector data (i.e. using Eqs. 18). The convergence criterion is average λ_i^1 change < 0.01%.

Table 1: Performance of C-HOEP with no left eigenvector data

new k	CPU/mode (sec, R only)	iterations
0.9	0.36	3
0.8	0.36	3
0.7	0.36	3
0.6	0.36	3
0.5	0.36	3
0.4	0.47	4
0.3	7.4	72
0.2	4.1	40 *
0.1	5.6	55 *

In Table 2, left eigenvector data is used (via Eq. 17), and CPU times per mode tracked are shown for computations of the perturbed eigenvalues and right perturbed eigenvectors (column **2)** and for computations of the perturbed eigenvalues and both the right and left perturbed eigenvectors (column **3).** The latter case is more realistic since left eigenvector data will have to

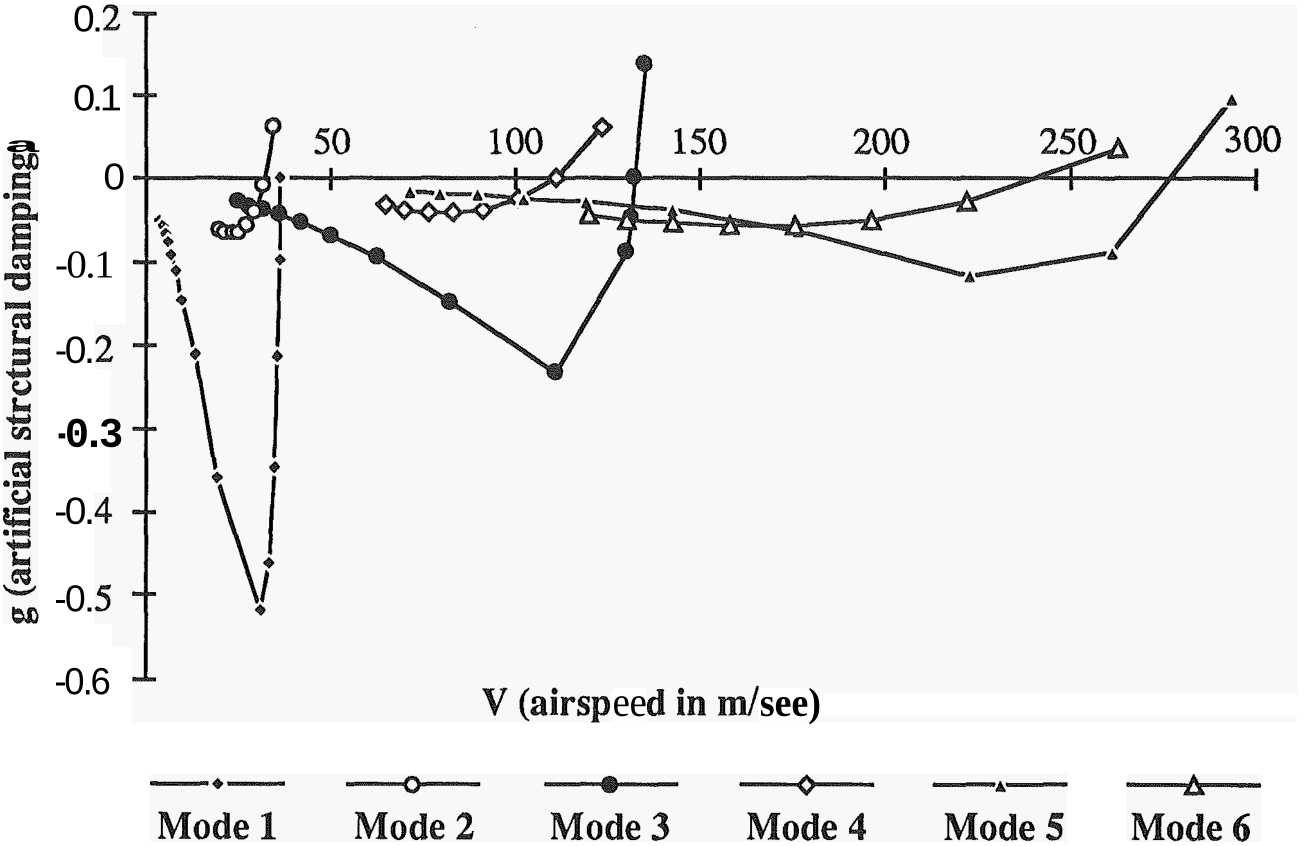

Figure 9: V-g plot for 0≤k≤1

be updated if **Eq. 17** is to be used in an incremental process.

Table **2:** Performance of C-HOEP with use of left eigenvector data

new k	CPU/mode (sec, R only)	CPU/mode (sec, R&L)	iterations
0.9	0.22	**0.30**	**2**
0.8	**0.22**	**0.30**	2
0.7	**0.22**	**0.30**	2
0.6	**0.22**	**0.30**	2
0.5	**0.31**	**0.43**	**3**
0.4	**0.39**	0.55	**4**
0.3	**3.1**	**4.6**	**37**
0.2	**2.6**	**3.8**	**31** *
0.1	**4.6**	6.8	55 *

The * designates failure in mode tracking in that the proper mode identifications are not made. The trend is obvious: as you push the perturbations higher, convergence is less assured. Use of the left eigenvector data is desirable, and only a relatively small penalty is paid for calculating the left eigenvector perturbations in addition to the right eigenvector perturbations.

In Table **3,** the performance of C-CORC is shown through CPU times and "corruption" data. **A** corruption index for a mode in the C-CORC method is the largest magnitude value in the column of [C] different from the correlated value (with a reference value of **1).** For example, if a column of $[C]$ has first and second largest component magnitudes of **1.2** and **0.3,** then the value of **1.2** correlates the mode and the corruption index for that mode is $0.3/1.2 = 0.25$. The "average corruption" index denotes the average of the corruption indices for the **6** columns in **[C],** and the "maximum corruption" index shows the largest corruption index. The smaller the corruption indices are, the better the assurance of proper mode correlation. Since the proper correlations are known a priori in this example, a corruption index of greater than 1.0 shows correlation failure.

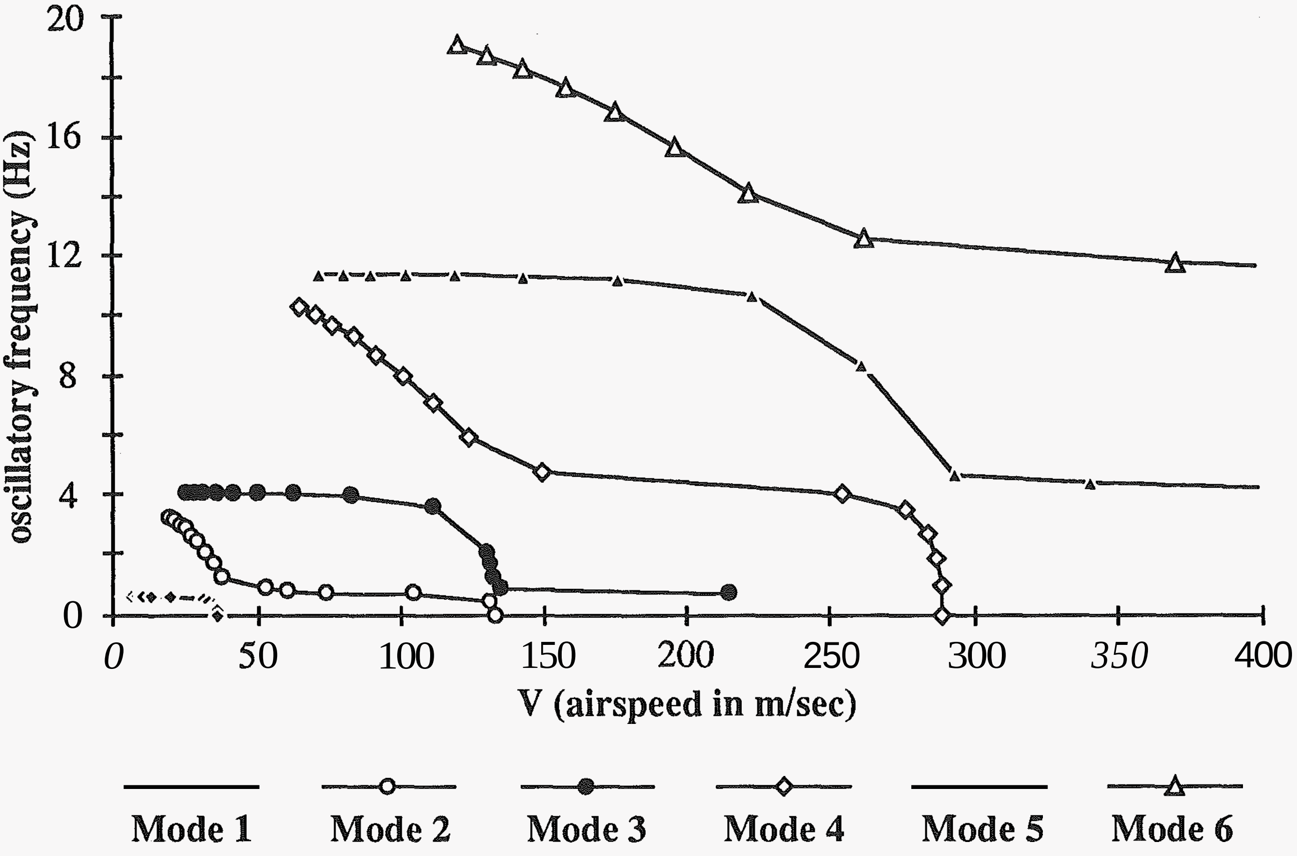

Figure 10: V-w plot for $0 \leq k \leq 1$

Table 3: Performance of C-CORC (left eigenvector data required)

new k	CPU (sec)	Avg Corrupt.	Max Corrupt.
0.9	1.2	0.018	0.043
0.8	1.2	0.037	0.084
0.7	1.2	0.059	0.12
0.6	1.2	0.087	0.16
0.5	1.2	0.13	0.19
0.4	1.2	0.21	0.33
0.3	1.2	0.47	1.1*
0.2	1.2	1.2	**3.2** *
0.1	1.2	2.7	6.9 *

The basic trend is the same: increasing the perturbation size decreases the performance of the method. Again, * denotes failure of the method.

There are large changes in pre-flutter modes 5 and 6 near $k=0.3$, causing the methods to have convergence problems. This is where the fifth and sixth frequencies can be seen to veer in Figure 10 and where the fifth and sixth modes can be seen to flutter in Figure 9. In this region, mode 5 correlates highly with mode 6 and visa versa, meaning that the mode shapes become very similar when near flutter. Mode tracking with the C-HOEP and C-CORC methods is tested in true incremental fashion for the range $0.15 \leq k \leq 0.35$, in order to test robustness in this difficult region. The initial k is 0.4 and increments are taken **of** 0.05 with the baseline updated at each increment. Table 4 shows the results.

Table 4: Robustness of C-HOEP and C-CORC near flutter points

incremental k	C-HOEP iterations	C-CORC Max Corrupt.
0.35	3	0.22
0.30	5	1.8*
0.25	10	1.9*
0.20	14	1.5*
0.15	12	**

Again, * denotes method failure in mode tracking in that the proper correlations are not made, and the

** denotes total failure in the C-CORC method since it correlated more than one mode with a single parent mode. It can be easily seen that C-HOEP is in fact more robust than C-CORC. Similar mode shapes near flutter cause C-CORC to fail, whereas C-HOEP can still track modes despite this mode shape similarity. Note that it takes more iterations to converge in the C-HOEP method as $k=0.0$ is approached since the change in airspeed is increasing. Once a mode tracking method fails, it is impossible to recover since the process is incremental and the following iteration modes are being correlated with a bad set.

Conclusions

Mode tracking techniques have been developed and applied to a range of problems in dynamic analysis and optimization. This often overlooked technology is an important bookkeeping tool which allows the analyst to maintain proper identification of modal data, thereby avoiding confusion caused by mode-switching. In optimization with frequency constraints, higher order eigenpair perturbations (HOEP) and the cross-orthogonality check (CORC) have both been shown to be effective in eliminating convergence problems caused by mode-switching. Relative efficiency of the methods depends on the reanalysis method used prior to CORC. If the reanalysis method must extract all eigenpairs, then HOEP has an advantage in only needing to track the constrained eigenpairs.

In V-g flutter analysis, C-HOEP has been shown to be more robust in the example problem than C-CORC. Near flutter points, different flutter modes can become very similar, causing orthogonality check correlation methods to fail. C-HOEP can successfully track modes near flutter despite mode shape similarity. Furthermore, C-HOEP can be more efficient than C-CORC if only a few eigenpairs need to be tracked.

If the problem requiring mode tracking technology is fairly well-behaved, then the orthogonality check methods may be preferable due to their simplicity. If, however, the problem is more difficult and large mode changes are possible, then the eigenpair perturbation methods are recommended due to their superior robustness.

References

[1] Eldred, M.S., Lerner, P.B., and Anderson, W.J., "Higher Order Eigenpair Perturbations," *AIAA Journal,* Vol. *30,* No. 7, July 1992, pp. 1870-1876.

[2] Nelson, R.B., "Simplified Calculation of Eigenvector Derivatives," *AIAA Journal,* Vol. 14, No. 9, Sept. 1976, pp. 1201-5.

[3] Tischler, V.A., and Venkayya, V.B., "Eigenvalue Routines in NASTRAN: A Comparison with the Block Lanczos Method," presented at Twentieth NASTRAN Users' Colloquium, April 27 - May 1, 1992.

[4] Komzsik, L., ed., "MSC/NASTRAN Handbook for Numerical Methods, MSC/NASTRAN Version 66," The MacNeal-Schwendler Corp., 1990, section 4.4, pp. 1-14.

[5] Bathe, K.J., and Wilson, E.L., *Numerical Methods in Finite Element Analysis,* Prentice Hall, Englewood Cliffs, New Jersey, 1976.

[6] Gibson, W., "ASTROS-ID: Software for System Identification using Mathematical Programming," to be published as WL-TR-92-XXXX.

[7] McGrew, J., "Orthogonalization of Measured Modes and Calculation of Influence Coefficients," *AIAA Journal,* Vol. 7, April 1969, pp. 774-776.

[8] Targoff, W.P., "Orthogonality Check and Correction of Measured Modes," *AIAA Journal,* Vol. 14, No. 2, Feb. 1976, pp. 164-167.

[9] Albano, E., and Rodden, W.P., "A Doublet-Lattice Method for Calculating Lift Distributions on Oscillating Surfaces in Subsonic Flows," *AIAA Journal,* Vol. 7, No. 2, Feb. 1969, pp. 279-285.

[10] Blair, M., "A Compilation of the Mathematics Leading to the Doublet Lattice Method," WL-TR-92-3028.