Large Order Modal Analysis Module in The Aeroelastic Design Optimization Program (ADOP)

George T.J. Tzong, Ph.D.
Gregory D. Sikes
Alan J. Dodd

Douglas Aircraft Company
McDonnell Douglas Corporation
Long Beach, California

Abstract

Two advanced mode extraction methods, the block Lanczos and the accelerated subspace iteration methods, have been implemented in the Aeroelastic Design Optimization Program (ADOP), a structural optimization program developed at McDonnell Douglas Corporation. Numerical errors most commonly observed in modal analysis have been overcome in both methods, using partial and selective orthogonalization schemes in the Lanczos method, and a selective technique in the subspace iteration method. Vibration modes for medium and large structural models are computed to demonstrate the ADOP modal analysis capability. In the paper a comparison between the block Lanczos and the subspace iteration methods is made in terms of CPU time usage. Since these two methods have never been compared on an equal basis before, i.e. using the same matrix storage format for the stiffness and mass matrices and the same routines to perform matrix-vector multiplication, and the numerical error treatments were not considered in the previous comparisons, this paper presents a more valid comparison.

Introduction

Structural vibration modes are calculated for various dynamic analyses applicable to the aerospace industry. In the past, a structure was either represented by a simple model with only a few degrees of freedom or by a model whose size was significantly reduced using dynamic reduction techniques [1]. Currently, large structural models are common, and direct modal analysis on the original (unreduced) structural models becomes possible because of the rapid development of advanced computer hardware and the feasibility of fast computing methods.

The power method [2,3], inverse iteration [2,3], subspace iteration [3,4,5] and the Lanczos method [6,7,8,9,10,11] are four popular eigensolution schemes to directly solve the vibration modes of structures. An accelerated subspace iteration [4,5] and a block Lanczos method [10] are used to develop a modal analysis capability in the Aeroelastic Design Optimization Program (ADOP) at Douglas Aircraft Company.

Subspace iteration extracts an eigenvector using a fixed-size subspace. By using a procedure to update the subspace continuously, the eigenvector will fall onto the subspace. The Lanczos method performs differently and extracts the eigenvector by expanding another subspace, starting with a small dimension.

Numerical error is a primary concern in developing a large-order modal analysis technique. The error due to finite computer precision is amplified in both methods. If no correction is made in the modal analysis, wrong eigenvalues or duplicates will be computed. Therefore, reliable error correction schemes [5,7,12] for both the subspace iteration and Lanczos method were developed and implemented in ADOP.

Both subspace iteration and the Lanczos method are efficient for large size problems. This paper presents direct comparison of their performance. Previously published comparisons [11,13] were either based on the number of matrix-vector operations [13] or the use of different computer programs [11,13]. A more valid comparison can now be made using ADOP, since both schemes use the same matrix-vector operation routines and the same storage for stiffness and mass matrices. The CPU times recorded in the ADOP modal analysis module for medium and large numerical examples are used to evaluate the performance of subspace iteration and the block Lanczos method. The reported CPU time not only includes that for the eigenvalue and eigenvector extraction in both methods but also contains the CPU time expended in numerical error correction. In addition, the numbers of solutions in both the subspace iteration and the Lanczos method are listed, along with the CPU time, to show the effect of numerical error correction on the overall performance of the two methods. The vibration modes are also computed using MSC/NASTRAN, and the results are used to verify the ADOP solution.

Accelerated Subspace Iteration

This method uses a small subspace (in ADOP, eight vectors are used) and continuously updates the subspace to match the invariant subspace spanned by all the required eigenvectors. During the iteration procedure, converged eigenvectors are removed from the subspace and the iteration continues for the remaining eigenvectors. In addition, accelerated subspace iteration [4] takes advantage of elaborate numerical techniques such as stiffness shifting and an overrelaxation factor.

The governing equation for modal analysis in structural dynamics using stiffness shifting can be written as

$$(\underline{K} - \mu \underline{M})\, \underset{\sim}{\phi} = (\lambda - \mu)\, \underline{M} \underset{\sim}{\phi} \qquad [1]$$

where $\underline{K}$ and $\underline{M}$ are the stiffness and mass matrices of the structure, respectively, λ is the eigenvalue of the system, $\underset{\sim}{\phi}$ is the corresponding eigenvector, and μ is a shift point.

Starting with an arbitrary set of vectors, $\underline{X}_1$, the subspace iteration algorithm in ADOP is written as follows:

(1) For $k = 1, 2, \ldots$ iterate from the *kth* subspace to the $(k+1)th$ subspace by solving

$$(\underline{K} - \mu\underline{M})\,\overline{\underline{X}}_{k+1} = \underline{M}\,\underline{X}_k \qquad [2]$$

where $\underline{X}_k$ is the matrix of vectors of the *kth* subspace, and $\overline{\underline{X}}_{k+1}$ is the matrix of intermediate vectors to be computed.

(2) Calculate the projectors of the stiffness and mass matrices.

$$\underline{K}^*_{k+1} = \overline{\underline{X}}^T_{k+1}\,(\underline{K} - \mu\underline{M})\,\overline{\underline{X}}_{k+1} \qquad [3a]$$

$$\underline{M}^*_{k+1} = \overline{\underline{X}}^T_{k+1}\,\underline{M}\,\overline{\underline{X}}_{k+1} \qquad [3b]$$

(3) Solve the eigenvalue problem of the projected system.

$$\underline{K}^*_{k+1}\,\underline{Q}_{k+1} = \underline{M}^*_{k+1}\,\underline{Q}_{k+1}\,\underline{\Omega}_{k+1} \qquad [4]$$

in which $\underline{Q}_{k+1}$ and $\underline{\Omega}_{k+1}$ are matrices of the eigenvectors and eigenvalues, respectively, of the projected system.

(4) The $(k+1)th$ subspace is then obtained by updating the *kth* subspace as

$$\underline{X}_{k+1} = \underline{X}_k + (\overline{\underline{X}}_{k+1}\,\underline{Q}_{k+1} - \underline{X}_k)\underline{\alpha} \qquad [5]$$

where $\underline{\alpha}$ is a diagonal matrix with its elements equal to the corresponding vector overrelaxation factors. The factors are determined by the convergence rate of subspace vectors toward the corresponding eigenvectors [4].

(5) Check the error growth with an error-monitoring scheme [5] for each subspace vector against each converged eigenvector, and identify the eigenvectors on which to perform Gram-Schmidt orthogonalization.

(6) Check if a shift is more advantageous [4] by comparing the computations with and without the shift.

(7) Check the convergence of eigenvalues, store the converged eigenvectors to backup storage, and continue the iteration until all the required eigenvalues and eigenvectors are found.

Steps 5 and 6 are discussed later in more detail.

Numerical Error Monitoring and Correction: The converged eigenvectors are stored in back-up storage and their positions in the subspace are filled by new vectors. If some of the converged eigenvalues are closer to the shift point than the unconverged eigenvalues, the corresponding converged eigenvectors may return because of numerical error, thereby squeezing the unconverged eigenvectors out of the subspace. The result is repeated computation of old eigenvectors and problems in searching for the subsequent eigenvectors. Gram-Schmidt orthogonalization is used in subspace iteration to avoid the return of converged eigenvectors. It is relatively inexpensive compared with the cost for iteration. However, in a large problem when more and more eigenvectors have converged, the cost of performing a full orthogonalization for all subspace vectors against all converged eigenvectors in every iteration becomes significant. Hence, a monitoring technique [5] was developed to detect the return of converged eigenvectors to the subspace.

Ref. 5 shows that the component of the *ith* eigenvector in the subspace is amplified by a factor of $1.0/(\lambda_i - \mu)$ after each iteration. The *pth* eigenvector may be excluded from the subspace by the return

of the *ith* converged eigenvector. The number of iterations before the problem occurs can be expressed as

$$n_{eff,ip} = \gamma \frac{\log|\varepsilon| - \log|\alpha|}{\log|\lambda_i - \mu| - \log|\lambda_p - \mu|} \qquad [6]$$

where ε is the computer accuracy(i.e., 10^{-14}). α is a random number representing the component of the *pth* eigenvector in the subspace; chosen as $\alpha \leq 1.0/\sqrt{n}$, n being the dimension of the problem. A safety factor, γ, set to 0.25 in ADOP provides early detection of the return of an eigenvector. If the number of iterations after the previous orthogonalization of the unconverged eigenvector p against the converged eigenvector i is larger than $n_{eff,ip}$, the Gram-Schmidt technique is performed. The procedure is continued until all the required eigenvalues have converged. The number of necessary orthogonalizations is therefore much less than with the full orthogonalization.

Selection of Shift Points: The main benefit of shifting the stiffness matrix is that the eigenvalue convergence rate is improved from $(\lambda_i/\lambda_{q+1})^2$ to $(\lambda_i - \mu)^2/(\lambda_{q+1} - \mu)^2$ [4], where λ_{q+1} is the first eigenvalue excluded from subspace. Shifting also reduces the risk of numerical error. However, a prudent selection of the shift point is crucial, because an eigenvector component cancellation can occur by shifting in the middle of two eigenvalues.

Let $\underline{X}_0 = \underline{\Phi}\begin{bmatrix} \underline{\delta} \\ \underline{\alpha} \end{bmatrix}$ be a subspace. The projectors of the stiffness and mass matrices according to Eq. 3 then have the form

$$\underline{K}^* = \underline{\delta}^T \underline{\Lambda}_1^{-1} \underline{\delta} + \underline{\alpha}^T \underline{\Lambda}_2^{-1} \underline{\alpha} \qquad [7a]$$

$$\underline{M}^* = \underline{\delta}^T \underline{\Lambda}_1^{-2} \underline{\delta} + \underline{\alpha}^T \underline{\Lambda}_2^{-2} \underline{\alpha} \qquad [7b]$$

where $\underline{\Phi}$ is the matrix of all eigenvectors, $\underline{\Lambda}_1$ is a diagonal matrix with the terms equal to the converged eigenvalues less μ, $\underline{\Lambda}_2$ is a diagonal matrix with the terms equal to the remaining eigenvalues less μ, $\underline{\delta}$ is the matrix of residuals of the previously converged eigenvectors in the subspace, and $\underline{\alpha}$ is the matrix containing the fraction of the unconverged eigenvectors.

An example assuming three converged eigenvalues, where the subspace dimension is four, is used here to demonstrate the cancellation. In addition, an equal magnitude for all elements in the $\underline{\delta}$ matrix is assumed for simplicity. The projector of the stiffness matrix is

$$\underline{K}^* = \begin{bmatrix} \Delta & & & \\ \Delta & \Delta & & \text{symm.} \\ \Delta & \Delta & \Delta & \\ \Delta & \Delta & \Delta & \Delta \end{bmatrix} + \underline{\alpha}^T \underline{\Lambda}_2^{-1} \underline{\alpha} \qquad [8]$$

where $\Delta = \sum_{i=1}^{3} \delta^2/(\lambda_i - \mu)$. If the shift is made midway between the second and the third eigenvalues, Δ is reduced to $\Delta = \delta^2/(\lambda_1 - \mu)$. The cancellation causes the components of the second and third eigenvectors to disappear from the stiffness matrix projector. However, the projector of the mass matrix does not have such a cancellation. Since the δ is being enlarged in every iteration, the cancellation induced by making such a shift causes the information related to the growth of error in the stiffness matrix projector to be lost. If the iteration procedure is continued and if the orthogonalization is not thoroughly performed, the wrong eigenvalues are computed because the error growth in the stiffness and mass projectors are not consistent.

The shift point selection scheme in ADOP is based on experience from numerical testing. The formulae for determining the shift point are

$$\mu = \lambda_{i-1} + 0.625\,(\lambda_i - \lambda_{i-1}) \qquad [9a]$$

$$\mu = \bar{\lambda}_{i-1} + 0.375\,(\bar{\lambda}_i - \bar{\lambda}_{i-1}) \qquad [9b]$$

$$\mu = \lambda_{i-1} + 0.625\,(\bar{\lambda}_i - \lambda_{i-1}) \qquad [9c]$$

where λ and $\bar{\lambda}$ are the converged and "nearly converged" eigenvalues, respectively. The selected shift μ is subject to the condition that $1.003\,\lambda_{i-1}$ (or $\bar{\lambda}_{i-1}$) $< \mu < 0.995\,\lambda_i$ (or $\bar{\lambda}_i$). The criterion defining "near convergence" of an eigenvalue is that $|(\lambda^{(k+1)} - \lambda^{(k)})/\lambda^{(k)}| < 1.0^{-5}$, where $\lambda^{(k)}$ is the *ith* eigenvalue approximation at the *kth* iteration.

The shift point determination tries to minimize the computation of the subsequent eigenvalues. In addition, the algorithm in ADOP verifies that the shift falls safely between two bracketing eigenvalues. If the space between them satisfies the condition set for Eqs. 9a-c, the shift will be made; otherwise, it will be adjusted left (toward the lower eigenvalues) into the preceding bracket. This procedure is repeated until an appropriate shift is found.

The use of 0.375 instead of 0.625 in Eq. 9b implies that $(i-1)th$ eigenvector will converge before the other eigenvectors. When a factor greater than 0.5 is used in Eq. 9b, the $(i-1)th$ eigenvector will not converge until all of the other subspace vectors have converged to their associated eigenvectors, resulting in a high computation cost for the convergence. This problem is mainly due to the cancellation discussed before, which causes the $(i-1)th$ eigenvector component to hide in the stiffness projector. Using 0.375 in Eq. 9b makes the fraction of $(i-1)th$ eigenvector dominant and significantly reduces the damage due to the cancellation in the iteration procedure.

Cancellation between the components of two eigenvectors enclosing the shift point will occur regardless of whether Eq. 9a, b, or c is used; however, the next iteration will return the procedure to the normal condition due to the unequal amplification of the components of the two eigenvectors. A more aggressive shift into a cluster of unconverged eigenvalues was tested and failed because of the multiple cancellations caused by the unconverged eigenvalues on both sides of the shift point.

Block Lanczos Method

The Lanczos vectors form a subspace, i.e. the Lanczos subspace, which attempts to include the invariant subspace spanned by all the required eigenvectors. The Lanczos method generates new vectors and expands the Lanczos subspace step-by-step until the invariant subspace is included. The block Lanczos method, which generates a small set of Lanczos vectors in every step, is adopted in ADOP. The block size can be one vector, which represents the simple Lanczos method, or several vectors, depending on the number of eigenvectors corresponding to any repeated eigenvalues in the problem.

The governing equation for the block Lanczos procedure is, for all steps $j = 1, 2, 3, \ldots$,

$$\underline{R}_j = \underline{Q}_{j+1}\,\underline{\beta}^T_{j+1} = (\underline{K} - \mu\,\underline{M})^{-1}\,\underline{M}\underline{Q}_j - \underline{Q}_j\,\underline{\alpha}_j - \underline{Q}_{j-1}\,\underline{\beta}_j \qquad [10]$$

where $\underline{Q}_{j+1}$, dimensioned n by s, is the matrix of Lanczos vectors of the $(j+1)th$ step and is the normalization of $\underline{R}_j$, n is the dimension of the problem and s is the number of vectors in a Lanczos block. In addition, $\underline{Q}_j$ is orthonormal with respect to the mass matrix ($\underline{Q}^T_i\,\underline{M}\,\underline{Q}_j = \underline{I}\,\delta_{ij}$ in which δ_{ij} is the Kronecker delta and $\underline{I}$ is the identity matrix). The coefficient matrices $\underline{\alpha}_j$ and $\underline{\beta}_j$, dimensioned s by s, are

$$\underline{\alpha}_j = (\underline{M}\,\underline{Q}_j)^T\,(\underline{K} - \mu\,\underline{M})^{-1}\,(\underline{M}\,\underline{Q}_j) \qquad [11a]$$

$$\underline{\beta}_j = (\underline{M}\,\underline{Q}_{j-1})^T (\underline{K} - \mu\,\underline{M})^{-1} (\underline{M}\,\underline{Q}_j) \qquad [11b]$$

where $\underline{\beta}_j$ is a lower triangular matrix. The block Lanczos method starts with a set of random vectors $\underline{R}_0$ and generates a new set of Lanczos vectors in every step until all the required eigenvalues and eigenvectors are found.

Premultiplying Eq. 1 by $((\underline{K} - \mu\,\underline{M})^{-1}\underline{MV})^T$, the eigenvalue problem is reduced to

$$\underline{T}\,\underline{y} = \frac{1}{(\lambda - \mu)}\,\underline{y} \qquad [12]$$

where $\underline{V} = \{\underline{Q}_1, \underline{Q}_2, \underline{Q}_3, \ldots, \underline{Q}_j\}$ is the subspace spanned by all Lanczos vectors up to the *jth* step. Matrix $\underline{T} = (\underline{M}\,\underline{V})^T (\underline{K} - \mu\,\underline{M})^{-1}\,\underline{M}\,\underline{V}$ is symmetric and banded with the bandwidth of the Lanczos block size. $\underline{y}$ contains the multiplication factors for all Lanczos vectors and is obtained using inverse iteration. The eigenvector is then written as $\underline{\phi} = \underline{V}\,\underline{y}$.

An error bound [8] is used as a criterion to extract eigenvalues and eigenvectors. This error bound is defined as

$$\sqrt{\underline{r}^T\,\underline{M}\,\underline{r}} = \sqrt{\underline{y}_{i,s}^T\,\underline{\beta}_{j+1}\,\underline{\beta}_{j+1}^T\,\underline{y}_{i,s}} \le \sqrt{\varepsilon}\,/\,|\,\theta_{i,j} - \mu\,| \qquad [13]$$

where $\underline{r} = (\underline{K} - \mu\,\underline{M})^{-1}\,\underline{MV}\,\underline{y}_i - \dfrac{1}{\theta_{i,j} - \mu}\,\underline{V}\underline{y}_i$ is the residue of the *ith* eigenvector approximation, $\theta_{i,j}$ is the *ith* eigenvalue approximation after the *jth* Lanczos step, $\underline{y}_{i,s}$ contains the last s elements in $\underline{y}_i$, and ε is the computer accuracy. $\underline{\beta}_{j+1}$ has an important role in computing the error bound. A conclusion made in Ref. 9 for the simple Lanczos method states that a sudden drop of the magnitude of $\underline{\beta}_{j+1}$ (or the largest diagonal term of the lower triangular matrix $\underline{\beta}_{j+1}$ in the block Lanczos method) indicates the convergence of an eigenvalue.

Numerical Error and Correction: There are two numerical difficulties in the Lanczos vector calculation: the return of converged eigenvectors and the duplication of existing Lanczos vectors. The error terms are due to the finite precision of computers and, although small, contain components of all eigenvectors. They also contain the components of all possible Lanczos vectors (the maximum number of Lanczos vectors is equal to the dimension of the problem). As seen in the following, the error terms are amplified in each step.

Assuming that no eigenvalue converges between the *jth* and $(j-1)$*th* Lanczos steps and the errors in these two steps are small, the Lanczos vectors $\underline{Q}_j$ and $\underline{Q}_{j-1}$ can be written as

$$\underline{Q}_j = \underline{\Phi}\left\{\begin{matrix}\underline{\varepsilon} \\ \underline{d}\end{matrix}\right\} \qquad [14a]$$

$$\underline{Q}_{j-1} = \underline{\Phi}\left\{\begin{matrix}\underline{\varepsilon}_e \\ \underline{d}_e\end{matrix}\right\} \qquad [14b]$$

where $\underline{\Phi}$ is the matrix of all eigenvectors. $\underline{\varepsilon}$ and $\underline{\varepsilon}_e$, dimensioned l by s, are small residues of converged eigenvectors in $\underline{Q}_j$ and $\underline{Q}_{j-1}$, respectively. l is the number of converged eigenvectors. $\underline{d}$ and $\underline{d}_e$ are the matrices containing the components of the unconverged eigenvectors in the Lanczos vectors.

The generated new Lanczos vectors are then

$$\underline{Q}_{j+1} = \underline{\Phi}\left[\underline{\Lambda}^{-1}\begin{Bmatrix}\underline{\varepsilon}\\ \underline{d}\end{Bmatrix} - \begin{Bmatrix}\underline{\varepsilon}\\ \underline{d}\end{Bmatrix}\underline{\alpha}_j - \begin{Bmatrix}\underline{\varepsilon}_e\\ \underline{d}_e\end{Bmatrix}\underline{\beta}_j\right]\underline{\beta}_{j+1}^{-T} \qquad [15]$$

where $\underline{\Lambda}$ is a diagonal matrix with terms equal to all the eigenvalues less μ.

The first term in the right-hand side of Eq. 15 shows that every eigenvector component in the new Lanczos vectors is amplified by a factor of $\frac{1}{(\lambda_i - \mu)}$, where λ_i corresponds to the *ith* eigenvector in the matrix $\underline{\Phi}$. In addition, the eigenvector components are amplified by a small $\underline{\beta}_{j+1}$, when any eigenvector converges.

Similar to Eq. 14, the Lanczos vectors of the *jth* and $(j-1)$*th* steps can be written as

$$\underline{Q}_j = \underline{V}_j\begin{Bmatrix}\underline{\delta}\\ \underline{i}\end{Bmatrix} \qquad [16a]$$

$$\underline{Q}_{j-1} = \underline{V}_j\begin{Bmatrix}\underline{\delta}_1\\ \underline{i}_1\\ \underline{0}\end{Bmatrix} \qquad [16b]$$

where $\underline{V}_j$ is the subspace formed by all exact Lanczos vectors up to Step j. The exact Lanczos vectors $\overline{Q}_j$ and $\overline{Q}_{j-1}$ are the error-free vectors and are slightly different from $\underline{Q}_j$ and $\underline{Q}_{j-1}$. $\underline{\delta}$ and $\underline{\delta}_1$ contain the residues of the exact Lanczos vectors of previous steps in $\underline{Q}_j$ and $\underline{Q}_{j-1}$, respectively. $\underline{i}$ and $\underline{i}_1$ are close to the identity matrix and are the components of $\overline{Q}_j$ and $\overline{Q}_{j-1}$ in $\underline{Q}_j$ and $\underline{Q}_{j-1}$, respectively.

The generated new Lanczos vectors are then

$$\underline{Q}_{j+1} = \underline{V}_j\left[\underline{T}_j\begin{Bmatrix}\underline{\delta}\\ \underline{i}\end{Bmatrix} - \begin{Bmatrix}\underline{\delta}\\ \underline{i}\end{Bmatrix}\underline{\alpha}_j - \begin{Bmatrix}\underline{\delta}_1\\ \underline{i}_1\\ \underline{0}\end{Bmatrix}\underline{\beta}_j\right]\underline{\beta}_{j+1}^{-T} + \underline{Q}_{j+1}\underline{\beta}_{j+1}^{T}\underline{i}\,\underline{\beta}_{j+1}^{-T} \qquad [17]$$

In exact arithmetics the terms in the bracket cancel and this would give an exact new Lanczos vector set, $\overline{Q}_{j+1}$. However, the real new Lanczos vectors contain the residues of the previous Lanczos vectors and the residues are amplified by the matrix $\underline{\beta}_{j+1}^{-T}$. According to Eqs. 13, 15 and 17, a sudden drop in the magnitude of $\underline{\beta}_{j+1}$ not only indicates the convergence of an eigenvector, but also induces the return of converged eigenvectors and a loss of orthogonality between Lanczos vectors. This finding was first reported by Paige [9]. Comparing Eqs. 15 and 17 shows that the numerical problem of the return of eigenvectors is more severe than the loss of orthogonality in Lanczos vectors due to double amplification in Eq. 15.

A full orthogonalization performed on the new generated Lanczos vectors against all preceding Lanczos vectors and all converged eigenvectors would definitely avoid the numerical problems, but is too expensive. Two orthogonalization schemes [7,12], one correcting the return of eigenvectors and the other avoiding the loss of orthogonality between Lanczos vectors, are adopted in ADOP. A selective orthogonalization scheme [7] monitors the growth of numerical errors due to the converged eigenvectors and detects any intolerable error. Orthogonalization is then performed on the Lanczos vectors against only the eigenvectors to which the error terms correspond. The equation for the detection of error growth used in ADOP is

$$\underline{\tau}_{j+1} \simeq \left|\left(\overline{\underline{\Lambda}}^{-1}\underline{\tau}_j - \underline{\tau}_j\underline{\alpha}_j\right)\underline{\beta}_{j+1}^{-T}\right| + \left|\underline{\tau}_{j-1}\underline{\beta}_j\underline{\beta}_{j+1}^{-T}\right| \qquad [18]$$

where $\underline{\tau}_j \equiv \overline{\underline{\Phi}}^T\underline{M}\,\underline{Q}_j$. $\overline{\underline{\Phi}}$ is the matrix of all converged eigenvectors, and $\overline{\underline{\Lambda}}$ is a diagonal matrix with diagonals equal to the converged eigenvalues less μ. The absolute sign is designated to the entries of the enclosed matrix products. The element e_{lk} in matrix $\underline{\tau}_{j+1}$ represents the error of the *lth* eigenvector in the *kth* Lanczos vector of the $(j+1)$*th* step. If the element e_{lk} exceeds the tolerance,

defined as $\sqrt{\varepsilon}$, the new Lanczos vectors and the Lanczos vectors generated in the previous step are orthogonal to the *Ith* eigenvector. Since the error in the new Lanczos vectors depends on the previous two steps, removing (or reducing) the error from the two steps delays the error growth in the subsequent Lanczos vectors.

A partial orthogonalization scheme [12], designed to monitor the error growth in the Lanczos vectors and decide which vector is to be orthogonalized, is written as

$$\underline{w}_{j+1}\underline{\beta}_{j+1}^{T} \simeq \underline{T}_j\underline{w}_j - \underline{w}_j\underline{\alpha}_j - \underline{w}_{j-1}\underline{\beta}_j \qquad [19]$$

where $\underline{w}_{j+1} \equiv \underline{V}^T\underline{M}\underline{Q}_{j+1}$. The entry f_{lk} in matrix $\underline{w}_{j+1}$ represents the numerical error of the *Ith* Lanczos vector in the *kth* Lanczos vector of the $(j+1)$*th* step. If any entry in $\underline{w}_{j+1}$ exceeds the tolerance $\sqrt{\varepsilon}$, the Lanczos vectors in both the *jth* and $(j+1)$*th* steps are orthogonalized against a range of Lanczos vectors. The range includes all Lanczos vectors with error terms greater than $\varepsilon^{3/4}$, because the growth of numerical error always starts from an intermediate Lanczos vector and spreads towards the neighboring vectors [12]. If only the Lanczos vectors with errors exceeding $\sqrt{\varepsilon}$ are orthogonalized, the errors in the Lanczos vectors closest to the ones being orthogonalized will exceed the tolerance in the next step.

Matrix Shifting and Selection of Starting Vectors: Shifting of the stiffness matrix can accelerate the convergence of the higher modes and reduce the numerical error growth in Lanczos vectors. Furthermore, the computation for eigensolution of the reduced matrix (Eq. 12) increases quadratically with the number of Lanczos vectors. Making a shift and restarting the Lanczos procedure with new starting vectors can reduce the computation.

The decision when to shift is based on the comparison between computational cost of the eigensolution of the reduced system, the orthogonalization and the formation of the converged eigenvector, i.e. $\phi = \underline{V}\,y$, and that of a new decomposition. If the cost of the former operations is more than 20% of that of the latter, a shift is made by selecting a shifted point using Eq. 9.

But stiffness shifting brings new problem. Numerical tests have shown that the shifting of the stiffness matrix disturbs the selective orthogonalization scheme. The scheme is still effective for the newly converged eigenvectors, but is no longer applicable to those eigenvectors converged before the shift. An external selective orthogonalization scheme [9] was suggested by considering an additional term in Eq. 18. This term is induced by the residue of converged eigenvectors, i.e. $\underline{d} = (\underline{K} - \lambda\underline{M})\phi$. However, it is found in this study that similar saving in orthogonalization to that in the external selective scheme can be achieved by providing different tolerances for both the newly and previously converged eigenvectors. The additional computation involved in the external selective scheme is then avoided. In ADOP a tolerance $tol = 0.01\sqrt{\varepsilon}$ is defined for the old converged eigenvectors and $\sqrt{\varepsilon}$ is used for the new eigenvectors.

A good selection of starting vectors can accelerate the convergence of eigenvectors. In theory if a starting vector is an eigenvector, it will converge in one step. The selection of "good vectors" is possible, after a new stiffness shifting, by using some nearly converged eigenvectors from the last shift as the new starting vectors. However, the fast convergence due to good initial selection disturbs the search for subsequent eigenvectors for the following reason. Generally, after Gram-Schmidt orthogonalization, starting vectors contain very little of the converged eigenvectors, but each contains an approximately equal magnitude of the remaining eigenvectors, i.e. the magnitude of $\underline{d}$ and $\underline{d}_e$ in Eqs. 14a and 14b is much larger than that of $\underline{\varepsilon}$ and $\underline{\varepsilon}_e$. When some eigenvectors converge, the new generated Lanczos vectors still have roughly equal contributions from all the remaining unconverged eigenvectors. On the other hand, the good starting vectors contain mostly the nearly converged eigenvectors, but very little of the remaining eigenvectors. The extraction of the converged vectors leaves only small residues for the remaining eigenvectors. In this case the magnitude of $\underline{d}$ and $\underline{d}_e$ is close to that of $\underline{\varepsilon}$ and $\underline{\varepsilon}_e$. The next Lanczos step therefore tends to regenerate the eigenvectors that converged before the shift and were closest to the shift point. The selective orthogonalization scheme fails to detect the error growth caused by the rapid convergence because the initial errors set in the matrix $\underline{\tau}_j$ assume a large ratio in magnitude between $\underline{d}$ and $\underline{\varepsilon}$. Using random vectors as starting vectors prevents this problem at the cost of a few more Lanczos steps.

Block Lanczos Algorithm: The Lanczos algorithm implemented in ADOP is summarized as follows:

(1) Select starting vectors $\underline{R}_0$ using a random number generator, orthogonalize them against the previously converged eigenvectors, and decompose the new shifted stiffness matrix.
(2) Compute the first set of Lanczos vectors $\underline{Q}_1$ by normalizing the random vectors and the matrix $\underline{\alpha}_1$ using Eq. 11a.
(3) Generate new Lanczos vectors using Eq. 10.
(4) Check for the numerical error growth. This includes several sub-steps:
    (a) Check for the return of converged eigenvectors using Eq. 18. If none are detected, go to Step 4-c.
    (b) Purge the returned eigenvectors from the Lanczos vectors of the current and previous steps.
    (c) Check the entries in $\underline{w}_{j+1}$ for the error growth associated with preceding Lanczos vectors in the new Lanczos vectors.
    (d) Identify and purge the preceding Lanczos vectors from those generated in the current and last steps if any error is found intolerable.
    (e) Reset the error terms corresponding to the Lanczos vectors or eigenvectors involved in the orthogonalization in the matrices $\underline{w}_{j+1}$ and $\underline{\tau}_{j+1}$ by the default number $\sqrt{n}\,\varepsilon$.
(5) Check the convergence of eigenvectors using Eq. 13 and compute them by $\underline{\phi} = \underline{V}\,\underline{y}$.
(6) Check for termination of the modal analysis if all the required eigenvalues and eigenvectors have been found.
(7) Check if a new shift is necessary and if so determine the new shift point.
(8) If a new shift is not required, go to Step 3; otherwise go to Step 1.

ADOP Modal Analysis Module

The ADOP modal analysis module is designed to compute structural vibration modes of large order finite element models. This module contains two modern eigenvalue solution techniques -- the block Lanczos method and accelerated subspace iteration. The user can select either method. Furthermore, the modal analysis module in the ADOP has the following options:

(1) An independent rigid body mode calculation: The rigid body modes can be calculated independently by balancing the inertia force caused by the corresponding motion. After those modes are computed, they are treated as converged eigenvectors in the modal analysis.

(2) Requested modes: Users can request a number of lowest structural vibration modes or modes in a range of frequencies of interest.

(3) Variable subspace and Lanczos block sizes: A default size of eight vectors is used in the subspace iteration and can be increased by the user. The Lanczos block size can be one to seven vectors with a default size of seven.

(4) Restart capability: If additional modes are required, the user can restart the modal analysis, assuming all related diskfiles for stiffness and mass matrices and eigenvectors from the previous run are saved. The module will automatically shift the stiffness matrix close to the first expected eigenvalue in the new run.

(5) Normalization and translation of modes: A mode, by default, is normalized to the mass matrix, but can also be normalized to its largest entry. A support point can be specified about which the modes translate and rotate (the default is at the center of gravity).

(6) Sturm sequence check: This is used to check if all required modes are found.

Numerical Examples

Three examples are presented to compare the accelerated subspace iteration to the block Lanczos method. The last two examples also compare ADOP results against those from MSC/NASTRAN. The computing facility used is an IBM 3090 without vectorization.

The first example is a transport aircraft main landing gear model, as shown in Fig. 1. This model, composed of 275 beams and 13 lumped mass elements, has 1319 degrees of freedom and the stiffness matrix has an average bandwidth of 33.

Separate analyses were made to compute the first 10, 20, 30, 40 and 50 vibration modes using both the subspace iteration and the block Lanczos method in which three block sizes of one vector, three vectors and seven vectors were used. Both the CPU time (excluding the first decomposition necessary to start the modal analysis) and number of solutions for each analysis are shown in Figs. 2, 3 and 4 (in which ASSI represents the accelerated subspace iteration and BLM-NV indicates results from the block Lanczos method with a block size of N vectors). The number of solutions (an estimation of computation) is the total forward and backward solutions for a single vector. The first ten natural frequencies of the model are shown in Table 1.

Fig. 2 shows that the number of solutions required for the subspace iteration is much more than that for the block Lanczos method. However, the CPU time usage does not increase proportionally (Fig. 3). This is because a significant amount of CPU time for the block Lanczos method is used for orthogonalization. The effect of the orthogonalization on the overall performance is even more significant when the CPU time for the decomposition of the shifted stiffness matrix is excluded, as shown in Fig. 4. When computing the first 50 modes, two shifts were made in the block Lanczos method with a block size of one vector and one shift for both block sizes of three and seven vectors. Ten shifts were made in the subspace iteration.

The second example is a transport aircraft wing model. This model contains 2157 rods and 1290 membrane elements and has 2422 degrees of freedom. The average bandwidth of the stiffness matrix is 615. Numerical tests similar to those for the main landing gear model were performed. The first ten natural frequencies, listed in Table 2, compare very well with those from the block Lanczos method in MSC/NASTRAN. The small difference is attributed to the different finite element formulations used in ADOP and in MSC/NASTRAN. (The hybrid element formulations [14, 15] have been used to compute the element stiffness matrices for the QUAD4 membrane and shell elements and the TRIA3 shell element in ADOP.) The fifth mode shape of the model is shown in Fig. 5.

Fig. 6 shows the number of solutions required for both the subspace iteration and the block Lanczos method, and Fig. 7 shows the corresponding CPU times. Because two additional stiffness shifts were made between the computation for 30 and 40 modes in the subspace iteration, the increase in the number of solutions is small. Fig. 7 also shows the CPU time excluding shifts in the subspace iteration and the Lanczos method. When computing the first 50 modes, six shifts were made in the subspace iteration and only one shift in the Lanczos method with a block size of one vector. No shift was made using block sizes of three and seven vectors. In Fig. 7, the ratios of CPU time between two methods is approximately equal to that in the number of solutions (Fig. 6).

The reason that the performance of the subspace iteration and the block Lanczos method in the above two examples is so different is that the bandwidths of the two stiffness matrices are very distinct. The main landing gear model has a small bandwidth of 2.5%. The CPU time for orthogonalization in the block Lanczos method is therefore a relative large part of the overall CPU time usage. The wing model has a large bandwidth (25% of the number of degrees of freedom). Most of the CPU time is used to perform the matrix-vector operations in the solution, and the time for orthogonalization is hence minor.

The performance of the MSC/NASTRAN block Lanczos method in computing the first 50 modes of the wing model is summarized. (The authors do not intend to compare MSC/NASTRAN with ADOP because (1) ADOP is still a development program; (2) the programs may be compiled with different code optimization levels which impact performance.) The MSC/NASTRAN Lanczos method used seven vectors in a block, two decompositions and 24 solutions and the total CPU time was 96.04 seconds. The CPU time, including two decompositions, used by the Lanczos method was 54.17 seconds. For this particular example, a better finite element nodal numbering was performed by MSC/NASTRAN than ADOP. The bandwidth of the MSC/NASTRAN stiffness matrix is only 10% of the ADOP stiffness matrix, resulting in a significantly lower solution time.

In the above two examples, the CPU time is not proportional to the number of solutions in the block Lanczos method using different block sizes. (In the second example, the CPU time for the block size

of seven vectors is less than those for the other two block sizes, although the seven-vector block requires more solutions.) A larger ratio of CPU time to the number of solutions for smaller block sizes is due to: (1) more orthogonalizations needed for smaller blocks; (2) more Lanczos steps and therefore more eigensolutions of a banded matrix (Eq. 12) performed. An orthogonalization (Eq. 6) is also performed in the subspace iteration, but is much cheaper than that in the block Lanczos method.

The final example is a transport aircraft inboard flap model, composed of 2303 bars, 3549 beams, and 6426 shells. There are 6411 nodes and 35577 degrees of freedom and the first 20 modes are computed using both accelerated subspace iteration and the block Lanczos method in ADOP. The eighth mode shape along with the undeformed structure is shown in Fig. 8. The first five vibration modes are also computed by MSC/NASTRAN using the modified Givens' method with dynamic reduction.

ADOP and MSC/NASTRAN results compare reasonably well and are shown in Table 3. This analysis required 36 iterations and two shifts in subspace iteration, and 27 steps in the block Lanczos method with seven vectors in a block. Subspace iteration used 197.11 CPU minutes and block Lanczos method used 53.25 minutes.

The MSC/NASTRAN block Lanczos method was also used to analyze the flap model. The first 20 modes required 229.11 CPU minutes, in which five decompositions and 49 solutions were made with a block size of three vectors. The CPU time, excluding the first and last decompositions, used by the Lanczos method was 152.61 minutes. (The frequencies are not presented because MSC/NASTRAN detected several local modes which were automatically restrained by ADOP.)

Conclusions

Two advanced large-order modal analysis techniques, the accelerated subspace iteration and the block Lanczos method, have been implemented in the Aeroelastic Design Optimization Program (ADOP). Numerical error correction schemes are also incorporated in both methods to avoid extracting duplicate eigenvectors. Numerical examples are tested to verify the ADOP modal analysis module and to compare the performance of the accelerated subspace iteration and the Lanczos method. The following conclusions are made:

(1) The performance of block Lanczos method is superior to that of the accelerated subspace iteration. However, the numerical error correction (or orthogonalization) in the block Lanczos method deteriorates its performance, especially for problems with a narrow banded stiffness matrix.

(2) Although more solutions are needed using a larger block size in the Lanczos method, the CPU time does not increase proportionally because fewer Lanczos steps and orthogonalizations are required than for a smaller block size.

(3) More stiffness shifts are required in the accelerated subspace iteration than in the block Lanczos method. Each shift requires the decomposition of a modified stiffness matrix and is very expensive for large size problems. Therefore, the block Lanczos method is recommended for large problems.

References

1. The MSC/NASTRAN Theoretical Manual, Level 15.5, The MacNeal-Schwendler Corporation, Los Angeles, CA, 1972.

2. Clough, R. W. and Penzien, J., *Dynamics of Structures*, McGraw Hill, New York, 1975.

3. Bathe, K. J., *Finite Element Procedures in Engineering Analysis*, Prentice-Hall, Inc., New Jersey, 1982.

4. Bathe, K. J. and Ramaswamy, S., "An Accelerated Subspace Iteration Method," *Computer Methods in Applied Mechanics and Engineering*, Vol. 23 , North-Holland Publishing Company, 1980.

5. Tzong, T. J., Sikes, G. D., and Loikkanen, M. J., "Large Order Modal Analysis Techniques in the Aeroelastic Design Optimization Program (ADOP)," SAE Technical Paper Series No. 892323, Anaheim, California, September 1989.

6. Lanczos, C., "An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators," *Journal of Research of the Numerical Bureau of Standards*, Vol. 45, pp 255-282, October 1950.

7. Parlett, B. N., and Scott, D. S., "The Lanczos Algorithm with Selective Orthogonalization," *Mathematics of Computation*, Vol. 33, pp 217-238, 1979.

8. Hughes, T. J. R., *The Finite Element Method* , Prentice-Hall, Inc., New Jersey, 1987.

9. Paige, C. C., "The Computations of Eigenvalues and Eigenvectors of Very Large Sparse Matrices," *Ph.D. Dissertation*, Institute of Computer Science, University of London, England, 1971.

10. Grimes, R. G., Lewis, J. G., and Simon, H. D., "The Implementation of a Block Lanczos Algorithm with Reorthogonalization Method," Boeing Computer Services, ETA-TR-91, May 1988.

11. Jones, M. T. and Patrick, M. L., "The Use of Lanczos's Method to Solve the Large Generalized Symmetric Definite Eigenvalue Problem," NASA Contract Report 181914, NASA Langley Research Center, September 1989.

12. Simon, H. D., "The Lanczos Algorithm for Solving Symmetric Linear Systems," *Ph.D. Dissertation*, University of California, Berkeley, June 1982.

13. Nour-Omid, B., Parlett, B. N., and Taylor, R. L., "Lanczos Versus Subspace Iteration for Solution of Eigenvalue Problems," *International Journal for Numerical Methods in Engineering*, Vol. 19, pp 859-871, 1983.

14. Loikkanen, M. J., "A 4-Node Thin Hybrid Finite Element", *Engineering Computations*, U.K., Vol. 2, pp 151-154, 1985.

15. Razzaque, A., "Program for Triangular Bending Elements with Derivative Smoothing", *InternationalJournal for Numerical Methods in Engineering*, Vol. 6, pp 333-343, 1973.

TABLE 1
NATURAL FREQUENCIES (Hz) OF
THE MAIN LANDING GEAR MODEL

FREQUENCY NUMBER	ADOP
1	9.009
2	9.785
3	18.959
4	25.043
5	30.273
6	31.204
7	39.899
8	49.534
9	53.189
10	55.480

TABLE 2
NATURAL FREQUENCIES (Hz)
OF THE WING MODEL

FREQUENCY NUMBER	ADOP	MSC NASTRAN
1	4.136	4.144
2	4.887	4.897
3	11.144	11.172
4	14.981	15.021
5	18.040	18.078
6	24.752	24.797
7	31.898	31.962
8	33.860	33.949
9	39.076	39.135
10	42.800	42.834

TABLE 3
NATURAL FREQUENCIES (Hz)
OF THE INBOARD FLAP MODEL

FREQUENCY NUMBER	ADOP	MSC NASTRAN
1	6.722	6.67
2	17.516	16.87
3	19.819	21.00
4	24.690	-
5	30.177	28.40
6	31.080	31.08
7	34.536	-
8	38.563	-
9	40.394	-
10	44.988	-

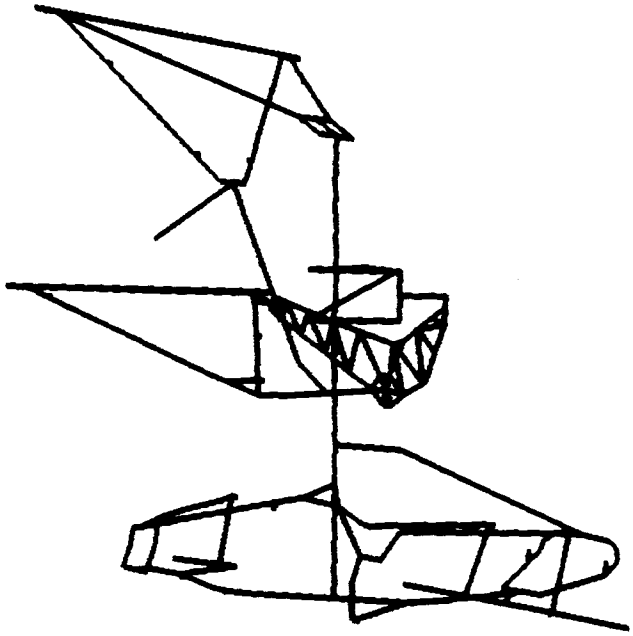

Fig. 1. Finite Element Model of a Transport Aircraft Main Landing Gear

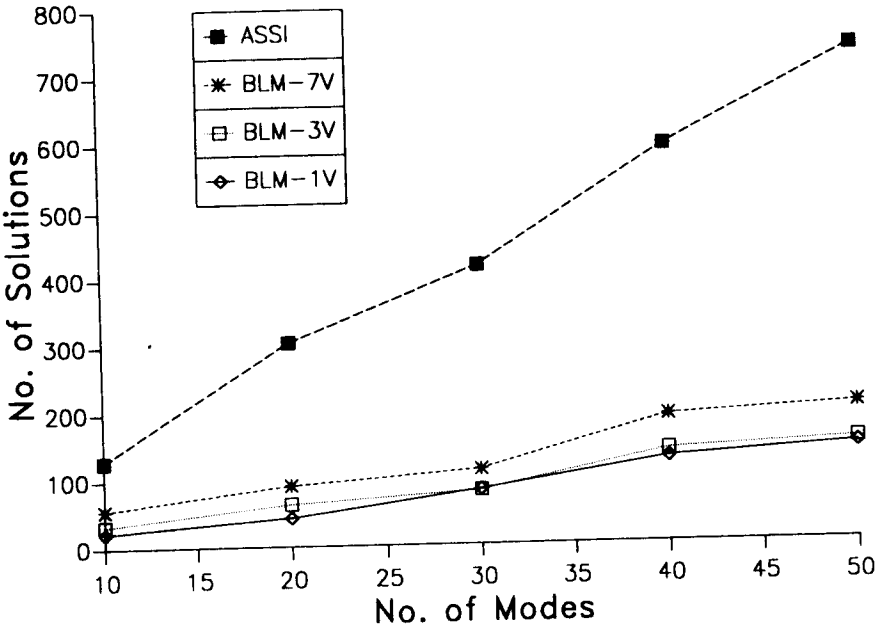

Fig. 2. Comparison of No. of Solutions for Main Landing Gear Model

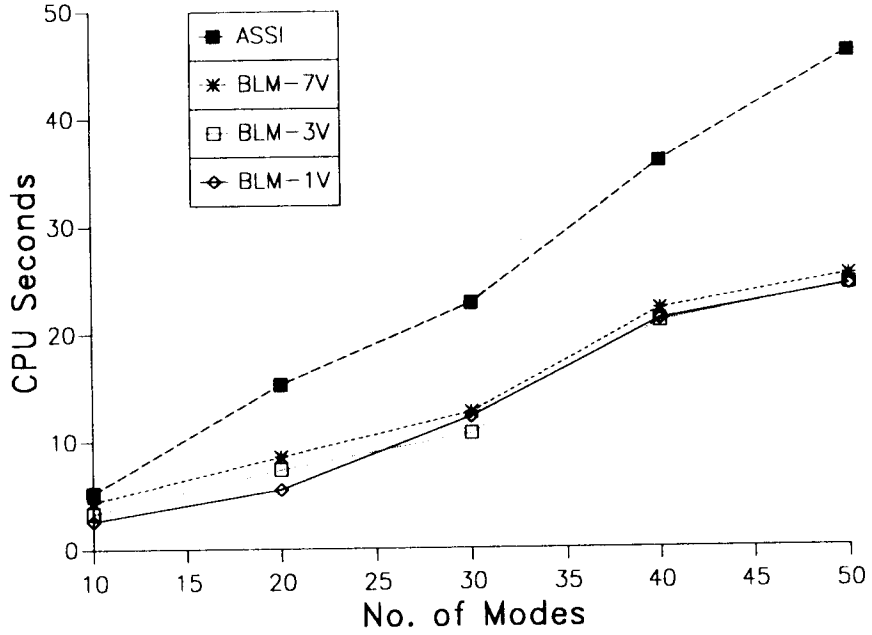

Fig. 3. Comparison of Total CPU Times for Main Landing Gear Model

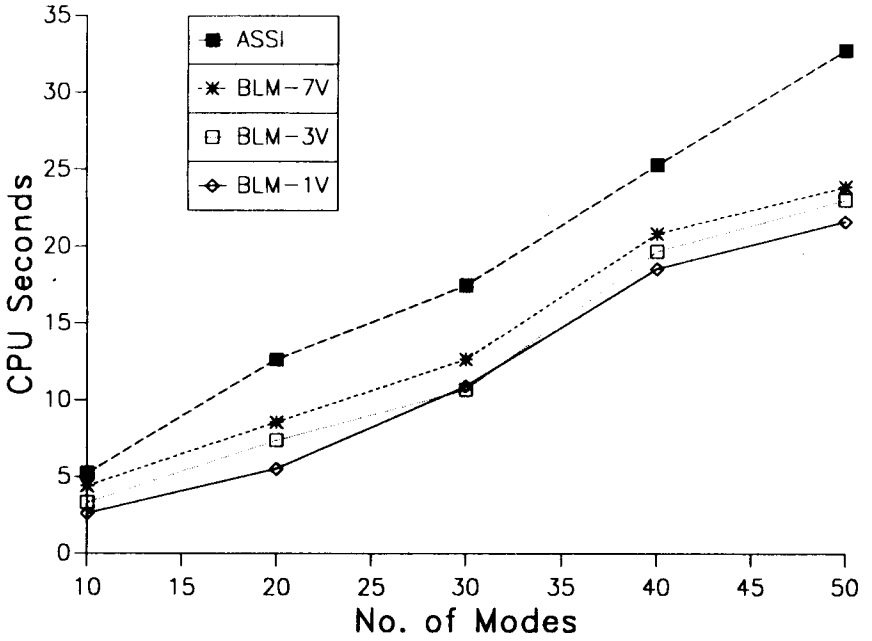

Fig. 4. Comparison of CPU Times Excluding Stiffness Shift for Main Landing Gear Model

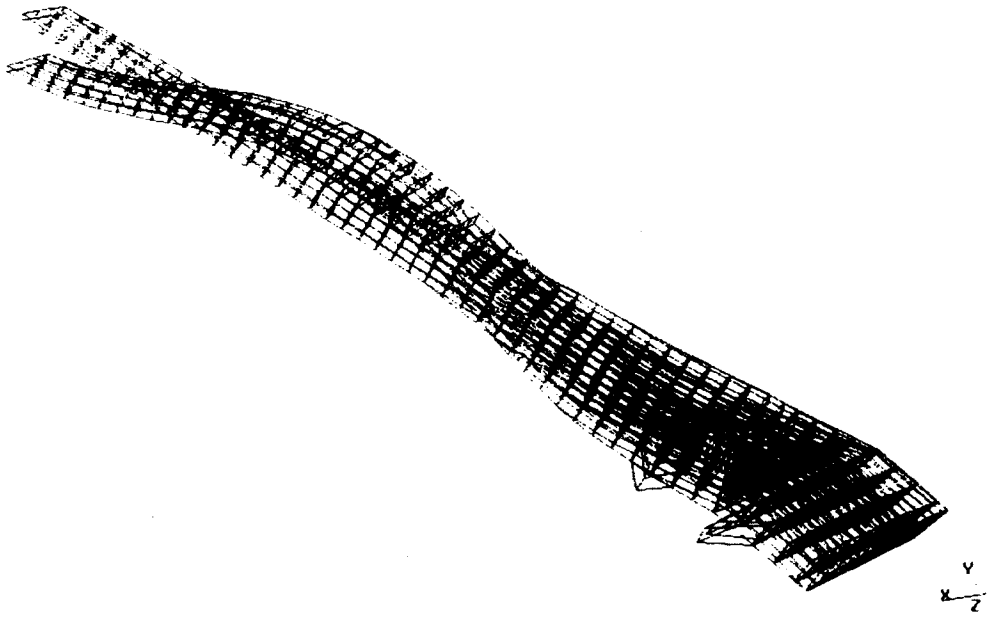

Fig. 5. Mode No. 5 of the Transport Aircraft Wing Model

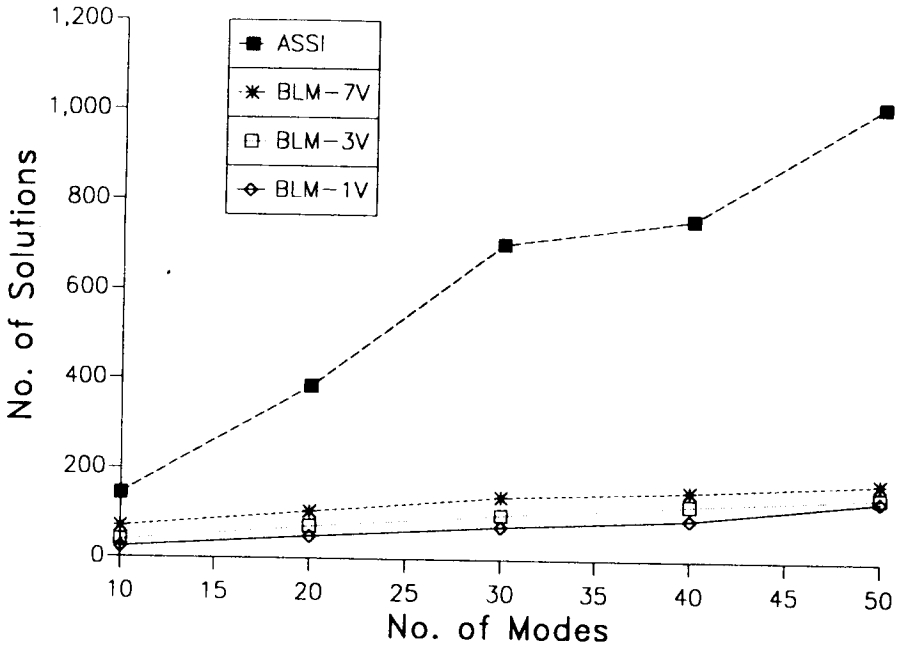

Fig. 6. Comparison of No. of Solutions for Wing Model

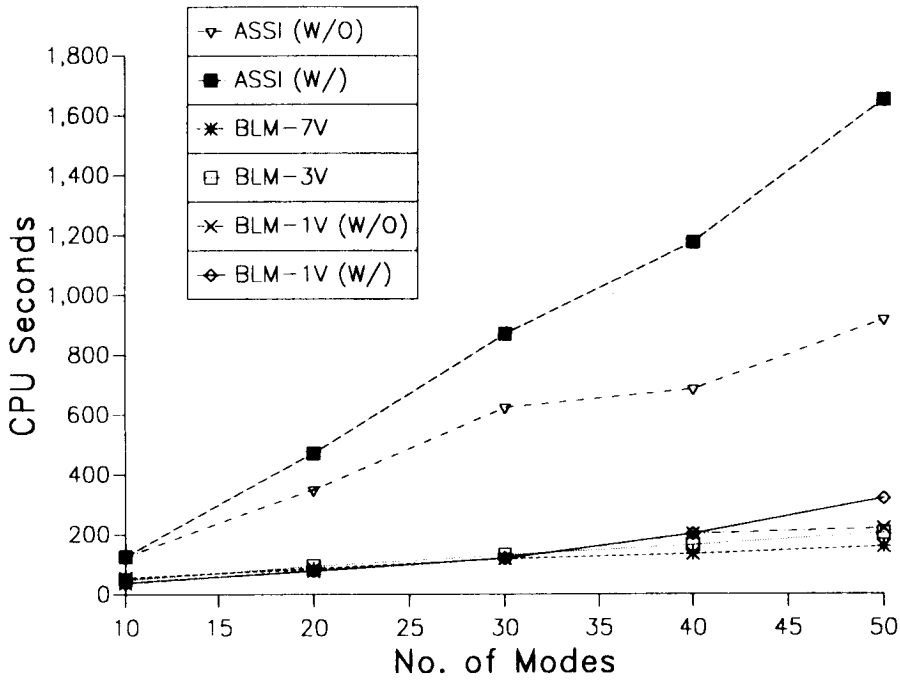

Fig. 7. Comparison of CPU Times With and Without Stiffness Shift for Wing Model

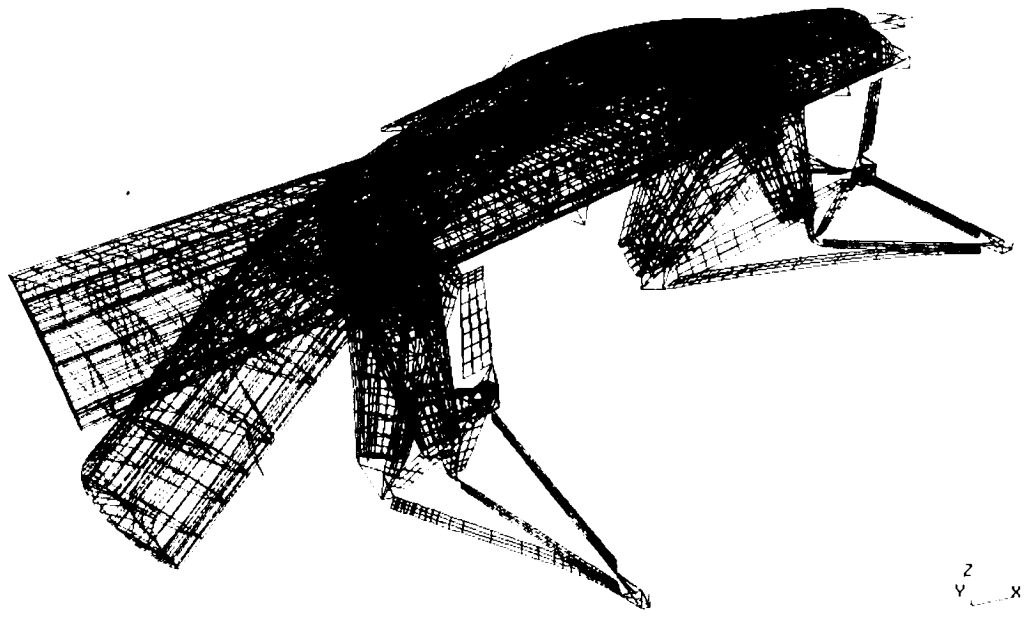

Fig. 8. Mode No. 8 of the Inboard Flap Model