ANALYSIS OF MODAL BEHAVIOR AT FREQUENCY CROSS-OVER

THESIS

Robert N. Costa Jr., Captain, USAF

AFIT/GA/ENY/94D-6

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GA/ENY/94D-6

ANALYSIS OF MODAL BEHAVIOR AT FREQUENCY CROSS-OVER

THESIS

Robert N. Costa Jr., Captain, USAF

AFIT/GA/ENY/94D-6

AFIT/GA/ENY/94D-6

ANALYSIS OF MODAL BEHAVIOR AT FREQUENCY CROSS-OVER

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Astronautical Engineering

Robert N. Costa Jr., B.S.A.E.

Captain, USAF

December 1994

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	28 Nov 94	Master's Thesis - Final

4. TITLE AND SUBTITLE

Analysis of Modal Behavior at Frequency Cross-Over

5. FUNDING NUMBERS

6. AUTHOR(S)

Capt Robert Costa Jr., USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Department of Astronautical Engineering
2950 P Street
Wright-Patterson AFB,OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GA/ENY/94D-6

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

WL/FIBAD
Wright-Patterson AFB, OH 45431

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release;
distribution unlimited

12b. DISTRIBUTION CODE

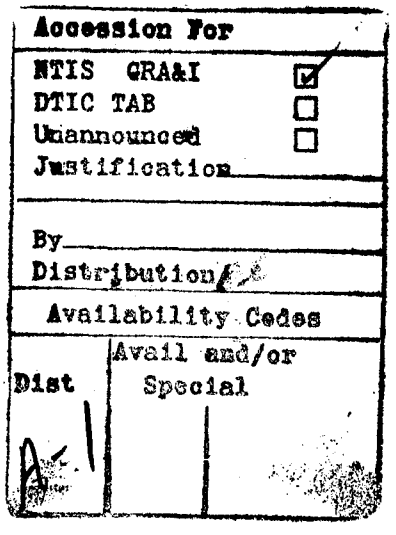
13. ABSTRACT (Maximum 200 words)

The existence of the mode crossing condition is detected and analyzed in the Active Control of Space Structures Model 4 (ACOSS4). The condition is studied for its contribution to the inability of previous algorithms to successfully optimize the structure and converge to a feasible solution. A new algorithm is developed to detect and correct for mode crossings. The existence of the mode crossing condition is verified in ACOSS4 and found not to have appreciably affected the solution. The structure is then successfully optimized using new analytic methods based on modal expansion. An unrelated error in the optimization algorithm previously used is verified and corrected, thereby equipping the optimization algorithm with a second analytic method for eigenvector differentiation based on Nelson's Method. The second structure is the Control of Flexible Structures (COFS). The COFS structure is successfully reproduced and an initial eigenanalysis completed.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Mode Crossing;Nelson's Method;ACOSS4;COFS; Modal Expansion;Cross-Orthogonality Check	248
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.



Acknowledgments

I would like to first and foremost thank my principal thesis advisor, Major (Select) Robert Canfield, for his unwavering support and technical guidance throughout this research effort. His patience, faith, and caring attitude have left me with far more than just research experience. I would also like to thank my thesis committee, Dr. Bradley Liebst and Dr. Curtis Spenny for their support during the research. In addition, I'd like to thank our sponsors at Wright Laboratories for the use of ASTROS and CADS and their technical assistance in using these programs.

Few students get through AFIT without strong peer support -- in that respect, I am grateful to Capts Blaylock, Carpenter, Decker, and Zadzora for their help and friendship. I am also especially grateful to Major Coullette for his kind words and unbiased support which lifted my spirits during some very dark days. I'd like to also thank Dr. Peterson at the WP Medical Center, for having an open mind and the medical acumen to diagnose and treat a ruptured disk that had gone unappreciated for so many months.

No words, however, can come close to capturing the gratitude I feel toward my wife Kimberly for her loving support and confidence through the physical and mental anguish we have been confronted with these past 18 months. I doubt I could have completed the degree requirements without her support, nor would the success have meant as much if she was not part of my life. Finally, I would like to thank my parents, Robert and Colleen Costa, whose countless personal sacrifices provided me the avenue for this and other opportunities, and a special thanks to my sister Sharon, who listened attentively and advised me on so many occasions.

Table of Contents

List of Figures

List of Tables

Abstract

The existence of the mode crossing condition is detected and analyzed in the Active Control of Space Structures Model 4 (ACOSS4). The condition is studied for its contribution to the inability of previous algorithms to successfully optimize the structure and converge to a feasible solution. A new algorithm is developed to detect and correct for mode crossings. The existence of the mode crossing condition is verified in ACOSS4 and found not to have appreciably affected the solution. The structure is then successfully optimized using new analytic methods based on modal expansion. An unrelated error in the optimization algorithm previously used is verified and corrected, thereby equipping the optimization algorithm with a second analytic method for eigenvector differentiation based on Nelson's Method. The second structure is the Control of Flexible Structures (COFS). The COFS structure is successfully reproduced and an initial eigenanalysis completed.

ANALYSIS OF MODAL BEHAVIOR AT FREQUENCY CROSS-OVER

I. Introduction

As large, deployable space structures became feasible in the latter half of the 1970's, structural and control engineers began to turn their attention to the problem of controlling the dynamics of these structures. The sizing constraints imposed by launch vehicles on spin-stabilized satellites, together with the lack of precision afforded by the crude attitude control systems inherent to those platforms, led to the development of computer-controlled, 3-axis stabilized vehicles. Because 3-axis stabilized vehicles generally rotate relative to their target (usually the earth) and not the sun, they require large areas dedicated to solar cells which rotate to maintain normality to the sun at all times. These solar cells are typically mounted on solar panels -- large flexible appendages connected to the main body of the vehicle by yokes or struts.

Forces from a variety of sources act upon these spacecraft continually. Such forces arise from internal sources, such as thruster activity, internal mechanical movement (e.g. gimbaled antennae, so-called "whisk broom sensors", etc.), as well as from external forces, such as solar pressure (which causes the majority of attitude deviations in the geosynchronous belt). These forces induce vibrations into these structures which in turn

affect the pointing accuracy of sensor suites and other sensitive payload equipment having little tolerance for such error. The undesirable flexible behavior of these systems can be minimized by employing means to dampen these vibrations. Depending on the payload requirements, the complexity of these control systems range from purely passive means (e.g. spin or gravity gradient stabilization, inclusion of visco-elastic material, etc.) to active means. These latter methods employ attitude control computers to execute control laws which in turn command actuators (e.g. reaction wheels, active structures, etc.) to dampen vibrations and thus inhibit the magnitude of attitude excursions [Liebst, 1994].

Vibrations naturally arise in all structures, but more so in space structures. Structures slated for operation in space are extremely flexible and inherently of closely-coupled, low frequency design. Low frequencies arise indirectly as a result of launch costs -- the cost of payload deployment demands light-weight, load-bearing structures to support payloads. Frequency is proportional to the square root of the ratio of stiffness to mass. The only source of stiffness is the light-weight (and therefore flexible) support structure. However, *both* the payload (where most of the total mass is typically concentrated) and the support structure contribute to the mass and thus the mass term in the denominator grows more rapidly than does the stiffness term in the numerator. In addition, as beams grow in size, stiffness typically decreases while mass increases proportionally to length. Both factors drive frequencies *down*.

Finally, the typical space structure is modeled (mathematically) as a truss. Trusses have *three* degrees of freedom for *every* pin-connected joint. Each degree of freedom in turn is associated with one natural frequency and one natural mode. Trusses are joint-

dominated structures and thus even small designs (modeled as *discrete* structures) have many natural modes. (Again, this discussion assumes a discrete model of the structure -- continuous models have an infinite number of natural frequencies and natural modes because they have an infinite number of degrees of freedom. This thesis will address only discretized structural models.) The number of natural modes coupled with the structure's inherent low frequency bandwidth result in the structure exhibiting high modal density as well as a general lack of stiffness.

As a result of the high modal density, some natural frequencies may be repeated, or may be numerically so close as to appear to be repeated to the system identification equipment monitoring the structure's attitude. Whether or not the frequencies are exactly repeated or just *nearly* repeated, their associated natural modes in most cases are distinct. Distinct eigenvectors create an insurmountable problem for a active control laws, which may apply actuator forces to dampen the wrong mode due to their inability to distinguish closely spaced natural frequencies.

Consider the following example of an *n*-degree of freedom cantilevered beam equipped with position and velocity sensors and actuators to control its natural modes. The natural frequencies and modes have previously been identified as listed in Figure 1-1 and used to build the feedback matrices which control the actuators. The beam is excited at 0.145 hz. Environmental factors have slightly perturbed the cross-sectional area of the beam, so that the feedback matrices no longer exactly represent the required feedback. Thus the actuators generate a force to control 1st Torsion although the beam has actually

deformed into a bending mode along the z-axis. The mode of course remains uncontrolled and the entire system may eventually become unstable.

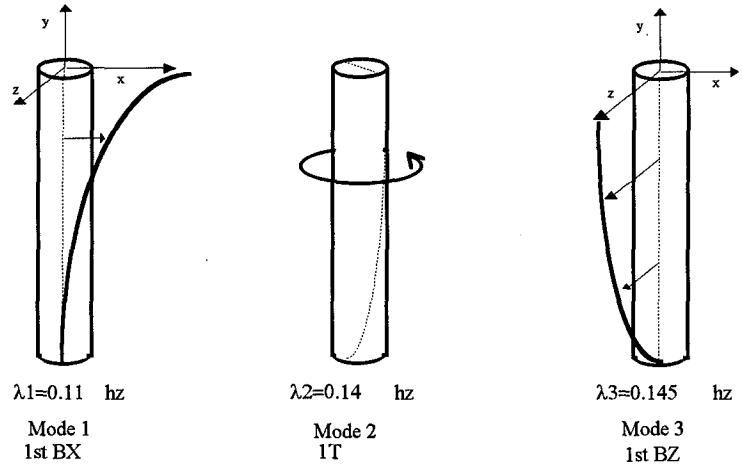


Figure 1-1: : Effect of Closely Spaced Frequencies on Modal Control

Low stiffness and its associated narrow frequency range cause problems in iterative structural *design* as well. At each iteration, the optimization program must complete at least a partial set of normal mode analyses, resulting in a vector of eigen*values*, the structure's natural frequencies, and an eigen*matrix*, a column-wise set of eigen*vectors*, each of which is a natural mode and which *initially* has a 1-1 correspondence with the set of eigenvalues. These programs typically order the eigenvalues by magnitude (with λ_1 being the smallest non-zero value) and the initial mode ordering is inherently carried along. At each iteration, adjacent frequencies (which may differ only by mere fractions of a hertz), may switch their order.

Consider the same problem of the cantilever beam in Figure 1-1 from a design perspective. Let the natural frequencies and modes listed above correspond to those from

the first iteration of a normal mode analysis. Figure 1-2 shows the results of the second design iteration where some properties of the beam have just been perturbed during an optimization scheme. After completing Iteration 2, 1st Torsion becomes associated with the *third* highest frequency, a frequency formerly associated with 1st bending deformation along the z-axis. An algorithm simulating the control response, does not detect the decrease in modal shape its response should have induced, and reconfigures the search to try a new direction. This action may eventually lead to non-convergence.

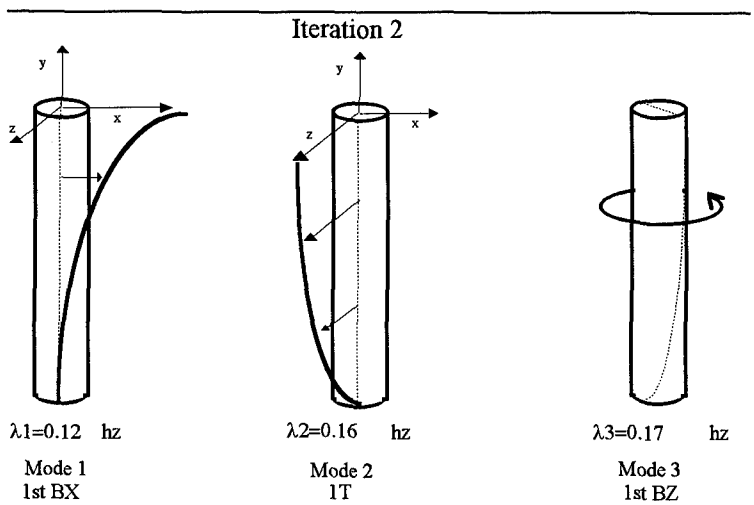


Figure 1-2: Results of Iteration 2

This *design* problem (as well as others) arises as a result of having combined the two key optimization problems -- the minimization of structural weight and the minimization of the control energy -- energy required primarily as a result of the flexibility induced by the former. In the past, structures were designed independently of the control system which would ultimately contain its behavior. The structure was designed and optimized, and its design passed to the control engineers who would then design an optimum controller based on that design. Results were satisfactory, but not truly optimum. With new

methods and more powerful computing platforms available, structural and control engineers soon realized the benefits of interdisciplinary design.

This optimization problem is commonly referred to as *the integrated structural design and control problem* and its solution has been aggressively pursued during this last decade. In the context of the mode crossing condition during the integrated design process, frequency cross-over results in the analytical solutions exhibiting two undesirable behaviors: 1) eigenvectors are non-unique, and 2) differentiating the non-unique eigenvectors analytically becomes difficult and costly, even if the issue of discontinuity can be addressed.

These problems have been observed in structures designed as test beds for active control algorithms. ACOSS4 (Active Control of Space Structures) is one such structure designed as an analytical tool to study the integrated structural design and control problem. The ACOSS4 structure is a tetrahedron modeled with actuators in each of the two legs at each of the three apexes of its triangular base. During optimization runs initiated by independent design teams in separate studies, the ACOSS4 model would not converge to a solution to the integrated problem. A similar convergence problem occurred with the Control of Flexible Structures (COFS) model, a three-dimensional truss comprised of 54 triangular bays, originally designed for shuttle deployment. Each bay consists of one strong and two weak longerons to promote close coupling of the frequency modes. Although COFS is an example of the structural design problem only, it has more degrees of freedom and is a more realistic model than ACOSS4. During the

optimization process, COFS engineers continued to use the analytical derivatives, but were unable to converge to a feasible solution.

Much of the literature on this subject of integrated design is dedicated to the simultaneous structural design in the presence of damping forces. Liebst, for example, minimized the performance index of a modified Linear Quadratic Regulator (LQR) in simultaneously designing active and passive dampers while minimizing the control energy required for these same damping sources. [Liebst, 1993: 1]. Likewise, Grandhi employed multivariable control techniques in using LQRs to optimize structures with behavior constraints imposed on the closed loop problem in the presence of damping. [Grandhi, 1988: 860]. Canfield used multiobjective optimization techniques to solve the integrated structural design and control problem. He sought to optimize vibration control as well as account for aeroelastic tailoring requirements but did not employ a control method involving passive damping forces [Canfield, 1992, 3]. Like Grandhi, Canfield used ACOSS4 as a test bed for the program he authored. This program, ***FRAME***, could not converge to a feasible solution when forced to use analytical derivatives of eigenvectors to determine an optimum search direction. It did, however, converge to the optimum solution when reprogrammed to use finite difference derivatives for the same purpose.

The problems described above may be the result of the mode crossing condition occurring in the integrated structural design and control problem. These, and other problems associated with mode swaps can be prevented by re-ordering the modes before the modal matrix is used in subsequent calculations. Such can be done *manually* through

the finite element program IDEAS Master Series™. IDEAS requires visual detection of the mode swaps, and then assists the user in manually realigning the modal matrix [IDEAS, 1990: 39-5]. Algorithms to detect and correct for mode swaps *autonomously*, however, have only recently been applied. These methods range from computationally-intensive perturbation methods, to efficient (but less robust) methods rooted in the basic properties of eigenvectors.

This research effort was designed to study the reason Canfield's solution based on analytical derivatives would not converge in the ACOSS4 model. It was believed the analytical solution failed to converge due the mode swapping condition, i.e. when the optimizer was faced with a mode swap, the mode shapes became discontinuous, and the optimizer could not re-establish a correct trajectory toward the optimum solution thereafter. It was also conjectured that this same phenomena prevented the COFS structure from successful optimization as well. To ensure the circumstances associated with modal crossings existed in these structures, an extensive literature survey was conducted to examine the problems associated with repeated or near repeated frequencies, which, as noted above, is a frequently cited cause of these mode crossings.

This thesis is divided into five chapters. Chapter 1 introduces the integrated structural design and control problem and its relation to the mode swapping condition. Chapter 2 is a summary of the literature survey completed in order to determine if 1) the subject structures were candidates for the mode crossing condition, and 2) if they were candidates, what means were currently in use in the field to locate and correct the condition. Chapter 3 is a summary of the mathematical theory behind the mode swapping

condition and of finite element modeling techniques. It includes a review of Nelson's method, a standard method in computing eigenvector derivatives, to determine its suitability to the integrated structural design and control problem. The ACOSS4 and COFS design programs are also summarized. Chapter 4 highlights the major results of this research effort and explains the reasoning behind the course of action taken in solving the problem. Chapter 5 contains conclusions drawn from this study and possible follow on research areas. Finally, the appendices have been tailored to contain only the most critical data, to include complete NASTRAN files portable to a variety of finite element software packages. Throughout the document titles of non-commercial programs are highlighted in capital letters in a bold italic font, subroutines are in capital letters and in an italic font, and program variables are in italics.

In summary, the goal of this research was to use proven numerical optimization to accomplish the integrated design on the ACOSS4 model with analytic derivatives. In the end, that goal was achieved, resulting in a solution calculated in 1/10 the time required for the previous approach using finite difference derivatives. In addition, the mode crossing condition was to be investigated as the cause of previous optimization failures.

Over the course of the research, it was determined that the mode swapping condition did **not** exist in the structure and that the original hypothesis that its existence had caused the original optimization failure was incorrect. A great deal of valuable material was developed in the process of arriving at that conclusion and is now available to the community. One key contribution was the development of a new, more efficient automated mode tracking algorithm based on cross-orthogonality checks. A mode

characterization scheme for visual identification of ACOSS4 modes was created where none had existed previously. A complete ACOSS4 optimization history is likewise now available. In addition, when frequency crossover was shown not to be the cause of the convergence failure, the existing analytic method for eigenvector differentiation based on Nelson's Method was replaced with one based on modal expansion. This new method not only resulted in convergence to the optimum design, it also highlighted the error in the original coding of Nelson's Method. This error was identified and corrected, thereby re-equipping ***FRAME*** with its original code intact and at the same time providing ***FRAME*** with a secondary analytic method for sensitivity analysis. Finally, although the COFS structure could not be completely optimized, the community was provided with an updated finite element model of the structure and complete an initial eigenanalysis on the +1400 element design.

II. Literature Review

2.1 Overview

Five main subject areas were studied from the current literature. The first main section of this chapter describes the first of the two models optimized in this study, the **A**ctive **C**ontrol **o**f **S**pace **S**tructures model (ACOSS4). Section 2.3 summarizes results on the integrated problem in literature associated with ACOSS4 and describes the results from other optimization studies. The next two sections are set up similarly for the **C**ontrol **o**f **F**light **S**tructures (COFS) model. The chapter concludes with an examination of three mode tracking algorithms currently in use in the field and laboratory. This last section is discussed more thoroughly, as it references more current publications and is more applicable to this particular thesis.

2.2 ACOSS4 Physical Model

ACOSS 4 (**A**ctive **C**ontrol **o**f **S**pace **S**tructures 4), is a mathematical model designed by Draper Labs for its research in the field of vibration control of large flexible spacecraft. The model was intentionally designed to have closely-spaced natural frequencies. Such a design challenges the abilities of new control algorithms to correctly distinguish, and subsequently control, the correct modes associated with each frequency. ACOSS 4 is simple in design, having only twelve degrees of freedom, but complex enough to provide several closely-coupled modes having the characteristics of various simple modes (e.g. 1st

Bending, 1st Torsion, etc.) The structure is a theoretical model only. Draper chose this design as it has under 20 modes yet still shares the same shape and characteristics of radar and optical mounting configurations where controlling the motion of the apex (Node 1) is the prime concern.

The nominal ACOSS 4 is a truss constructed of twelve hollow rods with various (circular) cross-sectional areas as shown in Figure 2-1. These twelve areas comprise the set of structural design variables for the standard optimization problem. Six rods form a tetrahedron and comprise the main body of the model. The triangular base is supported by two rods at each of three vertices which form the base. Each of the six legs is pinned at ground level. The structure is statically indeterminate and has 12 independent modes. The base and all three faces of the main body are equilateral triangles. [Strunce, 1980:1-3] In the integrated structural design and control problem, each of the bipods are equipped with actuators capable of generating axial forces, and co-located sensors returning position and velocity data. Parameters associated with these actuators comprise the set of control design variables. The finite element model is described in Appendix E.1 and the corresponding NASTRAN file is included in Appendix E.2.

2.3 ACOSS4 Optimization History

ACOSS4 was designed as a testbed and as such is a standard model used in testing new integrated problems. Source material was abundant. The design of the nominal structure is readily available from a wide variety of sources. All literature reviewed

stressed the enormous benefits achievable when the structure and its control system are designed simultaneously rather than separately as done in the past.

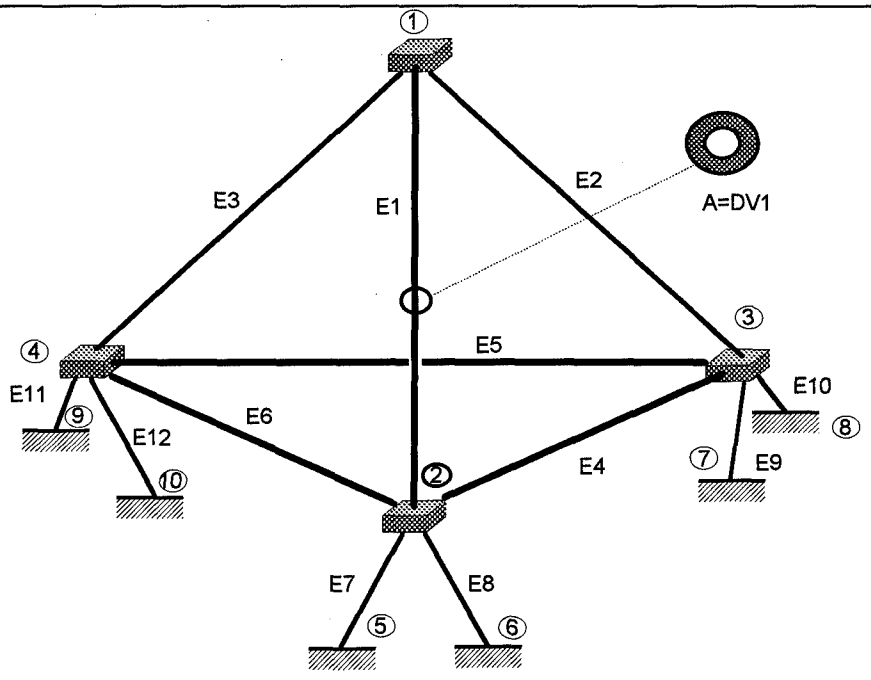


Figure 2-1: ACOSS4

In 1988, Grandhi employed ACOSS4 to test the accuracy of three different integrated optimization packages, IDESIGN, NEWSUMT-A, and VMCON. Weight was the objective function, the performance index of the LQR was also minimized, and the closed loop poles of the eigenvalues and damping parameters were both constrained. The design variable vector consisted of the twelve cross-sectional areas and the three programs iterated the design of an LQR using constant gain feedback. Grandhi's approach was to first determine the nominal design weight and the structure's natural frequencies and modes. These latter two values were used to assemble the plant and input matrices of the standard state space representation of the equations of motion. Closed loop eigenvalues

and damping ratios were determined and their sensitivities computed with the solution of the algebraic Lyapunov equations. These sensitivities were input into the three programs and the structure optimized. Constraints were as follows:

1. $\tilde{\omega}_1 = 1.341$
2. $\tilde{\omega}_2 \geq 1.5$
3. $\xi_1 = \xi_2 = \xi_3 = \xi_4 = 0.1093$

where ξ_i is the damping parameter. The nominal weight was 43.69 units. Each of the three algorithms tested took an unreasonable amount of iterations to converge upon the optimum solution, if they converged at all. IDESIGN converged to a weight of 26 after *110* iterations, NEWSUMT-A converged to a weight of 25 after 80 iterations, and VMCON failed to converge to any solution. [Grandhi, 1989:142]

In 1988, Khot studied two different approaches to the integrated problem. In the first approach, he defined structural weight as the objective function while constraining the two lower frequencies and the damping parameters as shown above. His controller too is an LQR using constant gain feedback. (The second approach defined the Frobenius norm as the objective function and will not be discussed.) Khot was able to achieve an optimized weight of 20.75 units in only 22 iterations, using NEWSUMT-A. [Khot, 1988: 359]

Oz and Khot optimized the ACOSS4 structure under a variety of constraint conditions, including equality and inequality constraints on the lower structural frequencies. Although their paper did not state the iteration history of the optimization process, weight reductions on the order of 350-400% were achieved when the efficiency of a structure-control system was also constrained. The FEM in this study was reduced so

only the lower eight modes were studied. They concluded that if structure-control interaction efficiencies were higher than 60%, and the truncated frequencies greater than 1 rad/sec, the results from the reduced model would compare favorably with those obtained from optimizing the full model, while providing marked decreases in computational time.

Canfield applied Independent Modal Space Control (IMSC) to the ACOSS4 problem in his dissertation on the integrated structural design and control problem. In IMSC, or so-called *natural* control, the closed loop equations retain the independence characteristics of the open loop equations. [Canfield, 1992, 35] Canfield first minimized the structure independently of any control minimization and showed that the characteristics of natural control prevented modal gains from significantly changing, commenting that any increases in the gains could be attributed to new modal shapes. Canfield then solved the integrated problem using an LQR (again with constant feedback) controlling six actuators in the bipods of the ACOSS4 structure in the presence of unit disturbances in the x and y directions, respectively, to evaluate typical actuator forces. He obtained the true optimum (minimized) weight of 14.69 units, and noted eigenvector sensitivity played a key role.

One key feature of Canfield's research was the closed form expression for all derivatives with respect to control design variables (modal gains). Finite difference perturbation methods were employed to calculate eigenvector sensitivities. These sensitivities were required in the semi-analytic derivatives with respect to the design variables and used to determine optimum search directions. Canfield's initial solution did incorporate Nelson's method, the state-of-the-art method for determining eigenvector

sensitivities. This method, however, fails in the presence of repeated or near repeated frequencies. Ojalvo and Mills-Curran both have provided methods to extend Nelson's Method to the case of repeated eigenvalues, but these extensions were not incorporated into Canfield's solution. Using sensitivities based on finite difference perturbation methods did burden the computational resources, as this methods requires $(n+1)$ finite element analyses. Here, n is the number of design variables, which number eighteen in Canfield's formulation (twelve cross-sectional areas and six control gains.) The analytical methods of eigenvector differentiation (Nelson's method, modal expansion, etc.) require only one FEM analysis.

Canfield resorted to finite difference methods when the optimizer failed to converge to a feasible solution based on sensitivities calculated with Nelson's method. This initial optimization may have failed due to the presence of repeated frequencies occurring as the structure was perturbed during the optimization process. This can lead to the mode crossing condition as discussed in the introduction. Nevertheless, Canfield's multi-objective optimization method did return a global minimum of 16.92 units when the objective function was defined as minimized weight. [Canfield, 1992: 89]

2.4 COFS Physical Model

Unlike the simple 12-element mathematical model afforded by ACOSS, (which if ever actually built could provide somewhat accurate results through ground based testing), the sheer complexity and size of COFS makes such testing difficult and provides unreliable

results, mostly due to the inability to prevent or, in the absence of prevention, account for the effects of gravity.

In response to the myriad of problems associated with ground testing [Hanks, 1984:13-15], NASA Langley concluded that the most accurate method to validate analytical results was to perform an on-orbit test of a deployable space structure. NASA designed a structure similar in design to most space trusses that was light enough to be carried aloft in one shuttle mission, which could be deployed, tested, and retracted by the shuttle crew, and one whose physical characteristics and configuration could be altered during testing to offer the broadest spectrum of test data. In addition, the structure was required to maintain cantilever end conditions, have a near-zero thermal expansion coefficient and integrate with the Space Technology Experiments Program (STEP) platform, a platform offering generic electronic and testing equipment for payloads deployed from the Shuttle cargo bay. These test results could then be compared to those generated by the analytical (finite-element) models, and model deficiencies could be corrected. The result was the Mast Flight System, the main feature of which is the Deployable Mast Subsystem, also known simply as the Mast, the MAST, or COFS. This thesis will use the latter nomenclature. The mast was to be built by the Astro Aerospace Corporation based in California. An artist rendering of the COFS structure fully deployed is shown in Figure 2-2.

COFS is a statically indeterminate, truss-dominated space structure comprised of 54 individual "bays" having the cross-section of an equilateral triangle 1.4 meters in diameter. Each bay is 1.124 meters high. The structure's main component is a simple 2-bay element

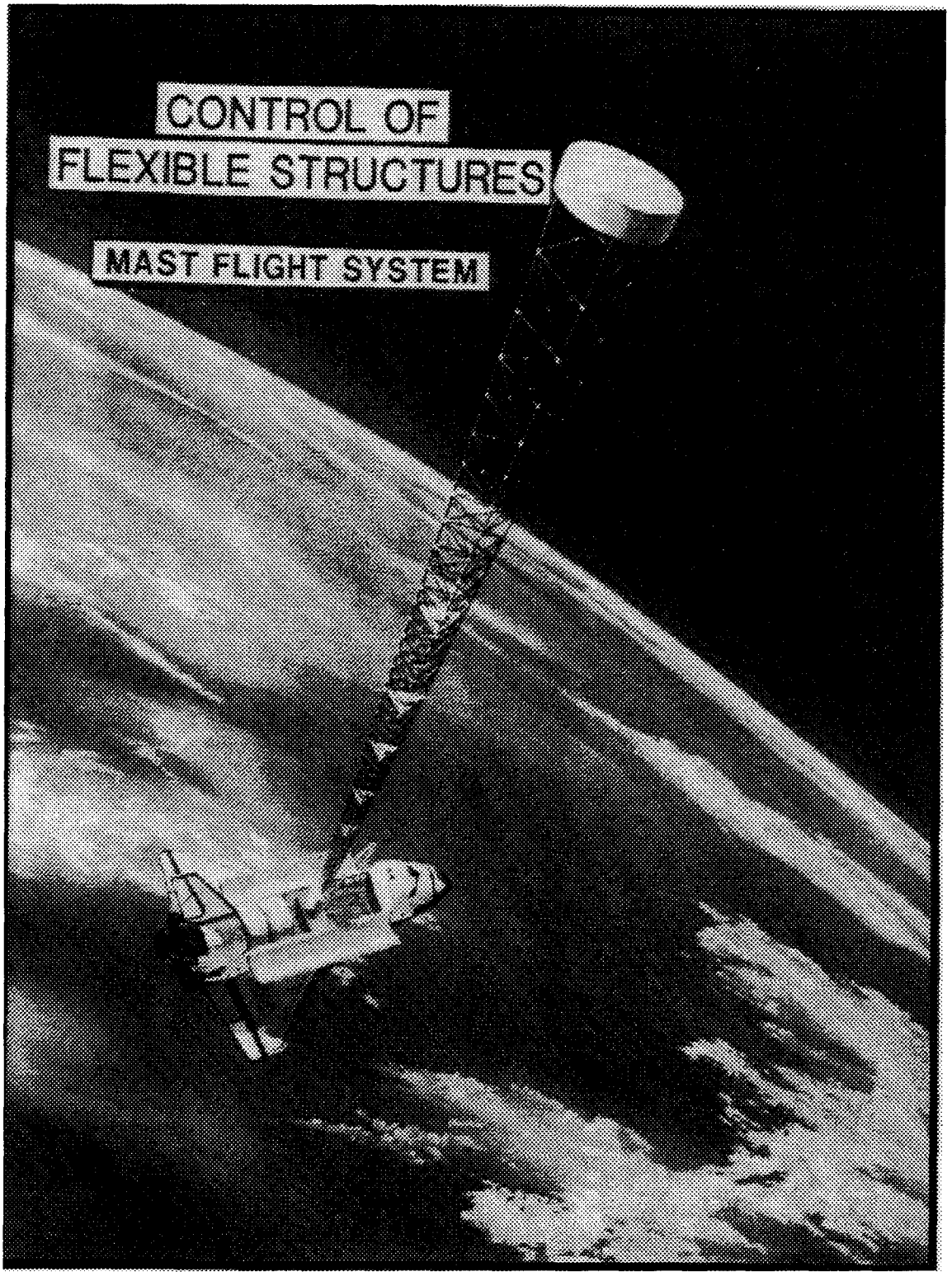


Figure 2-2: Artist Rendering of COFS Fully Deployed

(shown in Figure 2-3) replicated 27 times in the z-direction, for a total height of 60.7 meters. (For ease of reference, one bay of the 2-bay element is designated the 'A'-bay, the other the 'B' bay -- thus there are 27 A-bays alternating with 27 B-bays.) The truss is stowed in a package just over 2 meters high in the shuttle bay directly integrated with the STEP platform. Sensor suites are located on platforms at Bays 24 and 38, and actuators are located at Bays 12, 30, 44, and 54. A large tip mass contains the Parameter Modification Device (PMD) located on top of Bay 54. The PMD sits symmetrically on top of the structure, but its cross section is significantly larger than the 1.4 meter triangular cross section of the main structure.

Each bay is composed of three longitudinal members (longerons) to provide bending stiffness, three diagonals to provide torsional and shear stiffness, and three transverse members to stabilize the longerons. All members are made of graphite epoxy tubes having various elastic moduli all of an order of magnitude of $100*10^9$ N/m^2. All dimensions for the nominal design are provided in Appendix E.2. That appendix, as well as the information to follow, was compiled from the works by Colladay, Horta, and Talcott.

Longerons: Of the three longerons, one is designated the *strong* longeron and faces sternward (the artist rendering does show the incorrect orientation) along the centerline of the orbiter. The strong longeron is so named as it has a larger cross sectional area than the weaker longerons. COFS design engineers used different cross-sectional areas to ensure the nominal design had closely spaced, non-repeated lower structural modes. The two *weak* longerons are located at the other two vertices of the triangle. To ease packaging and deploying the truss, all longerons have nearly equal outside diameters with

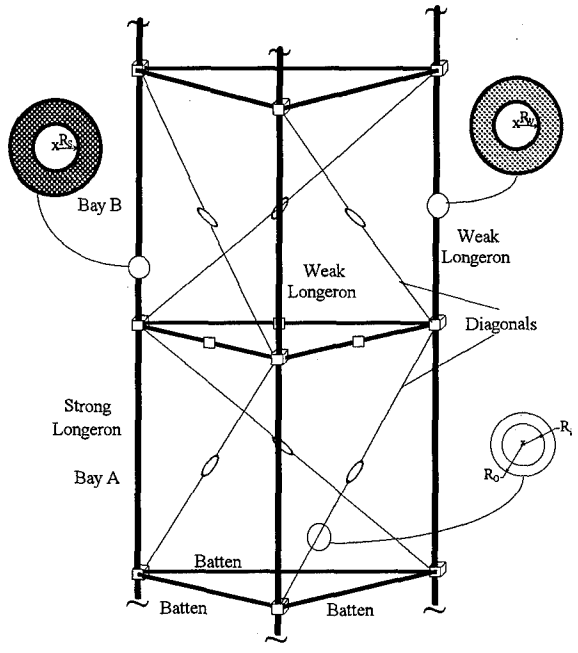


Figure 2-3: COFS 2-Bay Element

inner radii differing by over 40%. The strong and weak longerons in the first bay of the 2-bay element are identical to those in the second bay. The inner radius of the strong longeron (R_s) and the inner radius of the weak longerons (R_w) are the first two design variables (of a total of four) in the optimization algorithm.

Battens: All three battens have equal cross sectional areas in one bay and are designed to provide longitudinal stability. The battens in Bay A are continuous members and fit into titanium hinges at the vertices. The battens in Bay B are hinged at the midpoint to provide for packaging and re-stowing. Upon deployment, these hinges lock into place and unlock themselves during retraction with the introduction of compressive forces. The battens contribute no design variables.

Diagonals: All three diagonals have equal cross-sectional areas in each bay and are designed to provide torsional and shear stiffness. Diagonals alternate in Bay A and Bay B meaning that the endpoints of the diagonals pass through the same corner hinge at the batten line. All six diagonals in the 2-bay element are non-continuous members, being hinged at the midpoint for purposes of stowing and deployment. The outer radius (R_o) and the inner radius (R_i) of the diagonals provide the last two design variables in the optimization algorithm.

The centerlines of two adjacent battens from one bay, two corresponding longerons from a two bay element and two alternating diagonals from a two-bay element all pass through a common point at the vertex. All six elements are connected through a titanium end fitting at each vertex.

Tip Remote Station: The tip remote station (TRS) is the primary sensor/actuator suite housed in a package on top of Bay 54. It consists of four actuators (two along the x-axis and two along the y-axis). Each set can be commanded in phase to impart linear forces, or out-of-phase to generate torques about the z-axis. (See Figure. 2-4) The four actuators can be used simultaneously to provide a range of forces and torques. The actuators are standard Linear Direct Current Motors (LDCM) generating up to 30 Newtons of force with a peak-to-peak range of 32 centimeters. The tip station also houses two linear accelerometers (one for the x-axis, one of the y-axis) and one rotational accelerometer to measure z-axis torsion.

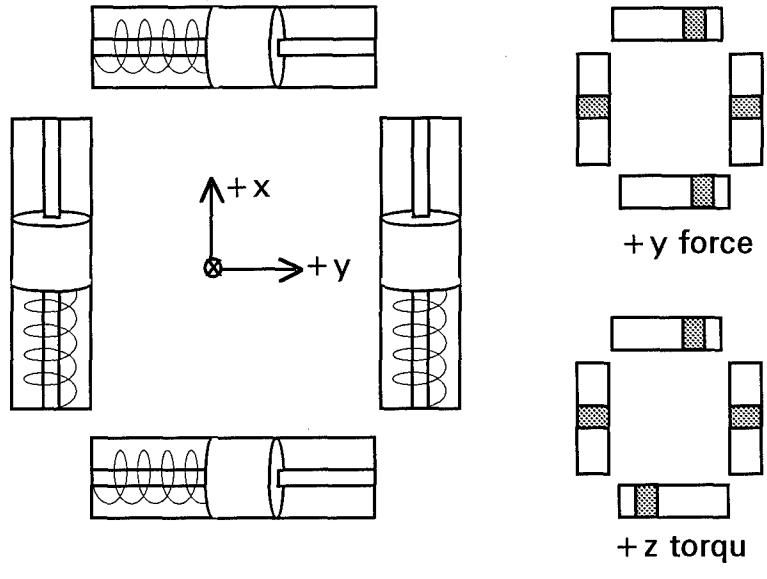


Figure 2-4: Tip Remote Station Layout and Operation

Secondary Actuator Suites: There are secondary actuator suites located at bays 12, 30 and 44. The equipment is attached to rigid platforms at the batten plane. Each platform holds two LDCM actuators (one for the x-axis, one for the y-axis) a co-located linear accelerometer for each actuator, and one rotational accelerometer. The primary function of the secondary actuators is to control bending modes in the x and y axes. Used in conjunction with the primary actuators in the TRS, the secondary actuators can provide insight into broadband excitation and control methods. [Colladay, 1986:11]

Secondary Sensor Suites: There are secondary sensor suites located at Bays 24 and 38. Each suite consists of three accelerometers (one x-axis, one y-axis, and the only z-axis accelerometers on the truss), and one rotational accelerometer. Since no actuators are present, these suites were included to provide researchers with insight into the non-minimum phase characteristics associated with non-co-located actuator/sensor pairs.

Parameter Modification Device: The parameter modification device (PMD) consists of four 20-kg masses guided along four independent linear tracks by pinion drive motors. These large masses can be independently displaced to offset the center of mass and/or to alter the torsional moment of inertia. It is housed in a single squat cylinder along with the TRS actuators and is bolted to the top of Bay 54.

Table 2-1 summarizes the location of all sensors and actuators.

Table 2-1: Summary of Actuator and Sensor Locations

	ACTUATORS		SENSORS				
BAY	+X LDCM	+Y LDCM	+X Acc.	+Y Acc.	+Z Acc.	Ang Acc.	Rate Gyro
TRS	2	2	1	1	1	1	3
44	1	1	1	1	1	1	0
38	0	0	1	1	0	1	0
30	1	1	1	1	1	1	0
24	0	0	1	1	1	1	0
12	1	1	1	1	0	1	0

2.5 COFS Optimization History

Recall, COFS, like ACOSS4, was designed as a testbed for space structure control algorithms. Thus there were several inherent operational constraints which had to be satisfied to ensure the structure would provide the opportunity to test the nuances of the subject control algorithms. Since COFS would be deployed in space, however, its weight still required minimization. In addition, because it was deployed from the shuttle, there were other physical (dimensional constraints), as well as frequency constraints to prevent interference with shuttle operation. These inequality constraints are listed below, and the corresponding optimization problem posed in Equation 2-1. These constraints hold for all literature reviewed, except where noted:

Objective Function	:	Minimize Mass
Design Variables	:	Inner Radii of Strong and Weak Longerons
	:	Inner and Outer Radii of Diagonal
1st Constraint	:	1st Natural frequency no less than 0.18hz
2nd Constraint	:	1st Torsion frequency within 1% of 2nd Bending frequency
3rd Constraint	:	1st Natural Frequency of Diagonal no less than 0.15hz
4th Constraint	:	Radius of strong longeron at least 0.25 mm larger than that of weak longeron
5th Constraint	:	Diagonal has wall thickness of at least 0.56 mm

$$F(X) = \text{total mass} \Rightarrow \text{Minimum} \qquad (2-1a)$$

$$X(dv) = \left[R_s \quad R_w \quad R_i \quad R_o\right]^T \qquad (2-1b)$$

$$g_1 = 1 - \frac{\lambda_1^2}{(0.18hz)^2} \le 0 \qquad (2-1c)$$

$$g_2 = \left| \frac{\lambda^2_{Torsion} - \lambda^2_{Bending}}{\lambda^2_{Torsion}} \right| - 1\% \le 0 \qquad (2-1d)$$

$$g_3 = 1 - \frac{\lambda^2_{1_D}}{(0.15hz)^2} \le 0 \qquad (2-1e)$$

$$g_4 = 0.254mm - \left(R_w - R_S\right) \le 0 \qquad (2-1f)$$

$$g_5 = 0.56mm - \left(R_0 - R_i\right) \le 0 \qquad (2-1g)$$

In 1986, Talcott and Colladay independently found the non-optimized structure's lower ten frequencies and modes to be as shown in Table 2-2. Talcott's results included the mass of the DRA and orbiter, as did Colladay's model. In the model used, all LDCMs are locked and the PMD configured to induce maximum inertia. Results are reported to the same number of significant digits as reported in the literature. Note the differences in the frequencies after mode 3. Although the difference between Colladay's and Talcott's results continue to grow with increasing mode number, a 1-1 correspondence with the mode shapes is maintained throughout.

Table 2-2: Comparison of COFS Natural Frequencies and Modes

	Talcott [Talcott, 1986: 260]		[Colladay, 1986: 31]		[Horta, 1986: 519]	
	Non-Optimized		Non-Optimized		Optimized	
Mode#	Freq (hz)	Mode Shape	Freq (hz)	Mode Shape	Freq (hz)	Mode Shape
1	0.1813	1B-X	0.21	1B-X	0.1888	1B-Y(X)
2	0.2387	1B-Y	0.24	1B-Y	0.2414	1B-X(Y)
3	1.2276	2B-Y	1.78	2B-Y	1.291	2B-X(Y)
4	1.2773	2B-X	1.97	2B-X	1.338	1T
5	1.3004	1T	2.18	1T	1.339	2B-Y(X)
6	3.5079	3B-Y	5.47	3B-Y	3.686	3B-X(Y)
7	3.6584	3B-X	6.07	3B-X	3.831	3B-Y(X)
8	4.3637	2T	8.12	2T	4.303	2T
9	6.0100	4B-Y	11.15	4B-Y	6.713	4B-X(Y)
10	6.2370	4B-X	12.46	4B-X	6.946	4B-Y(X)

Horta, et. al, completed one of the initial optimization studies of the COFS structure. He used a model of COFS built with the FEM package Engineering Analysis Language (EAL) and optimized the structure using the package CONMIN which uses constrained function minimization techniques to minimize weight while maintaining modal coupling [Horta, et al, 1986:519]. Like Collady and Horta, the FEM includes the DRA and the orbiter mass properties. Results are included in Table 2-2, above. (Note, Horta used a reference frame different than the first two, i.e. his x- and y-axes are interchanged when compared to the others. Reference with respect to a standard reference frame is included in parenthesis). Horta's results are very similar to those reported by Talcott, the only difference being the reversal of mode shapes 4 and 5 (1st Torsion and 2nd Bending in the (common) x-plane. Unfortunately Horta did not report the mode shapes of his baseline model and thus we cannot determine if a mode swap between modes 4 and 5 occurred.

Walsh optimized the COFS model in her paper describing an algorithm to design a large space structure having closely spaced frequencies. The same FEM is used. Walsh employs EAL to compute analytical eigenvalues and eigenvectors for the mass-normalized optimization problem, as well as the analytical eigen*value* derivatives. CONMIN requires these derivatives and the derivatives of the objective and constraint functions as well to determine a proper search direction. The latter derivatives are computed through piece-wise linear approximation methods using a 1st order Taylor series approximation and constrained with move limits. Walsh noted mode switches occurred at iterations 9, 13, and 20 but states this is rectified during a full analysis. (She does not state if the mode swaps simply did not occur in the full analysis, or whether the mode switching was

autonomously or manually rectified.) Nevertheless, Walsh concluded after 25 iterations that no feasible design exists which can satisfy the constraints listed in Equation 2-1 by merely varying the set of design variables provided. She additionally stated that more design freedom is required to solve the optimization problem. (It is interesting to note that CONMIN can supply its own eigenvalue derivatives through finite difference perturbation methods, similar to Canfield's secondary approach. Walsh instead had EAL compute them *analytically*. It would be interesting to study the optimization results had CONMIN used its own FD derivatives.) [Walsh, 1987: 5]

In their study of multiobjective optimization in the face of both equality and inequality frequency constraints, Grandhi and Venkayya used a different optimization algorithm than that used in the previous studies cited above. The optimization algorithm was based on an *optimality criterion*. This criterion uses a scaling procedure to determine the constraint boundary. The study used a different finite element model and optimization problem as well. COFS was modeled as a truss rather than as a space frame, the orbiter and DRA were removed from the structure, and a cantilever end condition was added. In addition, all elements were initially the same cross sectional area, eliminating the need for the *strong* and *weak* designations. (This additional symmetry promotes repeated frequencies as verified by the results.) The original optimization problem was replaced with the following

$$F(X) = \text{total mass} \Rightarrow \text{Minimum} \qquad (2\text{-}2a)$$

$$X = \left[R_{L_i} \quad R_{B_j} \quad R_{D_i} \quad \cdots \quad R_{L_{162}} \quad R_{B_{165}} \quad R_{D_{162}} \right] \qquad (2-2b)$$

$$g_1 = 1 - \frac{\lambda_1^2}{(0.18hz)^2} = 0 \qquad (2-2c)$$

$$g_2 = R_L - 0.1in \geq 0 \qquad (2-2d)$$

where

R_{L_i} ≡ radius of the ith longeron

R_{D_j} ≡ radius of the kth batten

R_{B_i} ≡ radius of the ith diagonal

There are a total of **489** design variables -- i.e. the cross-sectional areas of all battens, longerons, and diagonals are design variables. (Recall, the nominal COFS model has only 4 design variables). The initial cross-sectional area of all 489 elements was 1.0 in^2. Optimization results are shown in Table 2-3 for the first four frequencies. Final results correspond to the design after 20 iterations. Note that the initial symmetry does cause the lower frequencies to repeat, although the eigenvectors are distinct. This result could lead to a control problem as explained in Chapter I. The study reports that the two lower frequencies no longer repeat by the end of the optimization. We contend that the two lower frequencies in the optimized design may be *numerically* distinct, but would be indistinguishable in a real-world operational system. Note that this optimization problem results in very similar lower frequencies despite the use of 489 design variables. The mode shapes, however, are re-ordered. The paper also concludes that mode switching caused "abrupt" changes in the variable distribution which led to several oscillatory cycles before convergence [Grandhi, 1989:15-17]. The study used no mode swapping algorithm.

Table 2-3: Optimization Results from Grandhi's Formulation

Parameter	Nominal	Final	Mode
Weight (lbs)	1407.42	258.35	n/a
1st Freq (hz)	0.2058	0.1800	1B-X
2nd Freq (hz)	0.2058	0.1802	2B-X
3rd Freq (hz)	1.4570	1.1068	1T
4th Freq (hz)	1.4570	1.1221	not reported

2.6 Mode Tracking Algorithms

As mentioned in the introduction, only recently has the structures community dedicated the necessary time and resources to develop efficient mode tracking algorithms. Three separate mode tracking algorithms were reviewed and the potential for incorporation into this research effort assessed. The following summarizes each of these three efforts.

2.6.1 Higher Order Perturbation Method

The most robust automated method thus far appears to be the Higher Order Perturbation Method (HOEP) initially developed by Eldred in 1992. The basis of HOEP lies in the notion that the modes -- the eigenvectors -- are merely mathematical identities which can be monitored by tracking the perturbation in all structural parameters. Baseline eigenvectors are established at the initial design parameters. As these parameters are modified through some iterative process, the former eigenvectors are mathematically correlated with the current modes, and the "eigenpairs" [Eldred, 1992:1870] then tracked. The method is considered to be a forward analysis method and is based on a ". . .

perturbation expansion of the eigenproblem." [Eldred, 1994: 1] The method tracks the effect these perturbations have on the eigenpairs through subsequent iterations.

The HOEP method has several advantages over the methods based on the cross-orthogonality checks to follow, and one serious disadvantage.

1) This method is more robust than the following methods, as it truly tracks the mathematical effects of perturbations on the structure rather than basing modal tracking on an "after-the-fact" correlation.

2) The method allows the user to monitor only the constrained eigenpairs, vice requiring the algorithm to correlate all the modes.

3) Large perturbations of the design space and optimization problems using large move limits can still be effectively tracked with HOEP. These large perturbations can take the form of the total annihilation of a beam element in large space truss, or simply starting an iterative design process with a very non-optimum design. Other methods allow only small perturbations and small move limits.

HOEP's major disadvantage is that, besides requiring a massive programming effort to implement, it requires additional CPU time, even when only select eigenpairs are tracked. In his second paper on the HOEP method, Eldred compared HOEP to Gibson's cross-orthogonality check (CORC) and noted in models requiring small perturbations, CORC was up to 23% faster than HOEP and just as accurate. He also showed that other models with larger move limits do not however, fare so well under CORC, and may have corruption coefficients of up to 100%. These cross-orthogonality checks, however do warrant strong consideration, as describe below.

2.6.2 Cross- Orthogonality Checks

The Automated Structural Optimization System (ASTROS) is a design and optimization software package written for Wright Laboratories and used in the analysis of finite element models [Neil, et al, 1990:1021]. In 1992 under contract for the Air Force, Gibson authored an extension to the software to detect and correct for mode swaps which could occur during the optimization process. [Gibson, 1992: i]. Gibson bases his method on the notion that not only is every mode from a mass-orthonormalized matrix orthonormal to every other mode from that matrix, but every mode is also *relatively* orthonormal to every other mode in subsequent iterations of that matrix. This holds only if the design changes between iterations are small. The CORC matrix is computed as follows:

$$CORC = [\Theta]^{(n-1)^T}[M]^n[\Theta]^n \qquad (2-3)$$

where $\mathbf{\Phi}^{(n-1)}$ is modal matrix from the previous iteration, and $\mathbf{M}^n$ and $\mathbf{\Phi}^n$ are the mass orthogonalized mass matrix and the modal matrix, respectively, from the current iteration.

The individual elements of the CORC matrix indicate the degree to which the modes in the current matrix align with the modes from the previous iteration. Systems that are relatively orthonormal will have some non-zero off-diagonal terms, but these terms will be much smaller in magnitude than the diagonal terms; i.e. the system is said to be "diagonally dominant" [Gibson, 1992:6]. The row number, *i*, of the largest element in each individual column of a diagonally dominant matrix should therefore correspond to the column number, *j*, of the previous mode.

Systems that are not orthonormal may have some off-diagonal terms that are larger than the terms on the diagonal. If the row number and column number differ, it is the row number which indicates the mode swap that occurred between the i^{th} and j^{th} modes, i.e. the jth mode from the previous iteration is now the ith mode. For example, a CORC matrix whose 2nd column is $[.77\ .11\ .51]^T$ indicates that in this iteration, Mode 2 is mostly comprised of the same behavior as is Mode 1 from the previous iteration. It also indicates that it is also influenced by Mode 3 of the previous iteration, and is negligibly affected by Mode 2. The column can be normalized to determine strict percentages of modal influence but this is rarely necessary in well-behaved problems.

A complete flowchart of Gibson's Method (including a numerical example) is included in Appendix E. In general terms, Gibson's method works as follows. A vector is built with a 1-1 correspondence with the modes of the previous iteration. Since the modes in the last iteration are considered to be sequentially ordered, that vector is just a sequence of numbers the same length as the order of the problem. The CORC matrix is computed as shown above. Each column is processed individually. It is sorted and the row of the element having the largest absolute value is noted. This row number (actually the potential matching mode), is then compared to the element of the sequential vector residing in the same column as the column being processed in the current iteration. The mode is recorded in a tracking vector. If the mode just defined has already been assigned, that column is re-processed, now basing the modal assignment on the element with the next largest magnitude.

This method is computationally very efficient when compared to HOEP. It does have two major drawbacks however, as noted below:

1) Because each mode is processed individually, the direction in which the modes are analyzed is critical. Once a column (an eigenvector) has been processed, its mode is permanently assigned, regardless of other eigenvectors in the matrix. Thus it is possible that in processing the modes in one direction, say from column 1 to column 'n', the algorithm will detect no mode swaps. However, in processing the modes from 'n' down to 1, several mode swaps could be detected. (See the numerical example in Appendix D.1)

2) Modes between iterations only remain relatively orthogonal if the perturbations between modes are very small. Thus, this method does not work for large perturbations in the design space.

Gibson's method does have two advantages over HOEP:

1) The method is computationally much faster than HOEP and give results just as accurate as HOEP when move limits are small.

2) This method allows the emergence of new modes not originally detected in the baseline iteration. (This, however, can have mixed results if CORC is trying to correlate an entirely new mode with a set of former modes.) For this reason, Eldred suggests the use of a "correlation coefficient", the ratio of the two largest elements in a column, which warn the user of this occurrence when that ratio is greater than 0.5. [Eldred, 1992: 5]

In addition, others have used cross-orthogonality checks for the same purpose, lending validity to the method in general.

Ting, in his 1993 paper on an automated mode tracking strategy, developed a modified version of the PAREDYM FEM package. That package is primarily used for large scale finite element models. He demonstrated his method on a model of a Sikorsky Black Hawk helicopter where he showed his method effectively reduced mode shape errors in two lower structural modes from several hundred percent to just under 17%. [Ting, 1994: 975]

Like Gibson's, Ting used a cross-orthogonality check of the same form as Eq. 2-3. However, Ting went one step further by introducing a technique to partition the matrix, extracting only those modes warranting consideration -- a valuable procedure when dealing with large scale models and in line with the general thinking that it is the lower modes which best characterize the nature of a model. This technique also has the advantage of just monitoring constrained eigenpairs, as does the HOEP method.

Ting first calculated a matrix of Modal Assurance Coefficients (MAC) as follows:

$$[MAC]_{JI} = \left[\phi^{k-1}\right]_J^T \left[M^k\right] \left[\phi^k\right]_I$$

Note this is just the CORC matrix in Gibson's Method. Each row is then normalized by the absolute value of the largest term in that particular row, and any term less than one (which is every term except the largest value) is zeroed. This new matrix $[P]^k$ is called the Boolean operator and becomes the identity matrix of rank j for a diagonally dominant matrix. The Boolean operator then post-multiplies the current eigenvector matrix which effectively permutes the individual eigenvectors. (The paper makes no mention of using

the operator to also permute the eigenvalues as well, so we must assume this is also done.) This operator can be reduced and modified to extract only certain eigenvectors for analysis as well.

In the case of strongly coupled modes, Ting's procedure used small perturbation techniques to first locate the coupled modes and their locations in the eigenmatrix. Ting believed these modes can not be effectively tracked and should therefore be ignored during that iteration. The procedure removes these modes from consideration during that iteration by replacing them with the corresponding modes from the previous iteration (assuming these could be tracked in that iteration). The actual modes are returned only after a subsequent iteration shows they have been decoupled.

Although Ting achieved substantial gains using this procedure, we found this method to have several major drawbacks when considered for a general problem:

1) Like Gibson's method, because the entire matrix is not considered as a whole, it will suffer from the same ordering problem; i.e. it may detect mode swaps using one sequence, but detect no such swaps if the sequence is changed.

2) No mention is made as to the route taken when a subsequent mode has its largest element on the same row as a previous row.

3) Ignoring strongly coupled modes could lead to losing all knowledge of these modes during subsequent iterations. If the modes remain coupled, neglecting them only prevents subsequent correlation when their characterization is most critical.

4) The method only works for optimization techniques using small perturbations.

Despite the drawbacks noted above, Gibson's method was used initially to determine the mode swaps in the ACOSS4 problem. A cross-orthogonality check was much simpler to program than was the HOEP method. The time savings afforded by a simpler programming task would be critical if it turned out that a mode swap did not occur, (or if one did occur, had had a negligible effect on the result), as it would allow time to pursue other solutions. In addition, some of Gibson's drawbacks could be eliminated with additional programming steps. These advantages were noted as good reasons to use CORC instead of HOEP [Eldred, 1994:10-11], although that particular reference does go on to state that using a hybrid approach -- using CORC when the problem is well-behaved but autonomously switching to HOEP if it became unruly -- is the best algorithm.

III. Mathematical Theory

3.1 Overview

The following chapter is an overview of the mathematical foundation upon which this thesis is based. Section 3.2 contains the background of the finite element process as it is applied to this thesis. It is somewhat more thorough than would normally be expected, but its detail was required due to the reprogramming effort necessitated by our initial (and inconclusive) thesis results, as will be seen in Chapter 4. Section 3.3 introduces the eigenvalue problem to include a summary of the problems caused by repeated eigenvalues. It includes a discussion of the physical manifestation of an eigenvector. Section 3.4 describes the four methods by which eigenvalue and eigenvector derivatives were calculated for this research effort. The last section is an overview of basic optimization principles as they pertain to this thesis.

3.2 Finite Element Formulation

The equations of motion governing a system can by determined by several means. The two most common methods are through the application of classical dynamics (which requires free-body diagrams and Newton's second law) -- the other through Lagrange's equations which is based on principles of variational calculus. One serious drawback of

the former is the need to determine forces at each point where forces and moments act, even though they may be internal to the structure and will cancel out each other when the system's equations of motion are determined. The latter approach enables the derivation of finite element equations directly in terms of the nodal displacements. If internal forces are desired, they are available as a post-processing step in finite element analysis once the displacements have been found.

Craig presents a derivation based on Hamilton's principle [Craig, 1991: 243-257] which when supplemented with material from Meirovitch [Meirovitch, 1970:72-76] is complete. He first finds the local stiffness and mass matrices in a truss structure. This formulation can be used in conjunction with the method of assumed modes. The method is an extension of virtual displacement principles which uses an *admissible function* (a function which satisfies all geometric boundary conditions and provides the required number of continuous derivatives) to approximate the deformation of a continuous system. [Craig, 1981: 252] The displacement is a function of position and time and is the product of a the shape function, $\psi(x)$, and a time-dependent generalized coordinate $u(t)$. Deflections in n-degree systems are approximated by the sum of the products

$$\boldsymbol{u}(\boldsymbol{x},\boldsymbol{t}) = \sum_{i=1}^{n} \varphi_i(\boldsymbol{x})\boldsymbol{u}_i(\boldsymbol{t}) \qquad (3\text{-}1)$$

The assumed modes method forms the basis for finite element theory as well, where the mode shapes describe the deflections of the individual finite elements. These elements are the discretized portions of the structure. In the two structures studied, the elements are actually the entire length of a beam or rod, or a large mass placed at a node. Nonethe-

less, the deflections can be modeled as the sum of only two products of the form given by Eq. 3-1, with stiffness and mass computed from

$$k_{ij} = \int_0^L EA\varphi_i'(x)\varphi_j'(x)dx \qquad (3\text{-}2.a)$$

$$m_{ij} = \int_0^L \rho A\varphi_i(x)\varphi_j(x)dx \qquad (3\text{-}2.b)$$

ACOSS4 is a *truss*, while the COFS is a *space frame*. A truss is a pin-connected structure comprised of axial members, and as such, its individual members have no bending deformation, unlike a frame. The COFS structure is comprised of beam elements. Each element of the individual structures was considered a finite element and no further discritization of individual elements was necessary. To find an element's local stiffness and mass matrices, a local coordinate system is employed to find the deflection as shown in Figure 3-1 [Craig, 1981: 397].

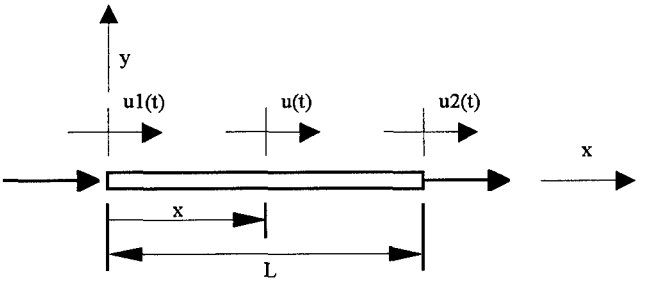


Figure 3-1: Local Coordinate System for Axial Element

The deflection is then simply the sum of the two products

$$u(x,t) = \varphi_1(x)u_1(t) + \varphi_2(x)u_2(t) \qquad (3\text{-}3)$$

where

$$\varphi(x)_1 = 1 - \frac{x}{L} \quad \text{and} \quad \varphi_2(x) = \frac{x}{L}$$

Substituting these modes shapes into Eqs. 3-2, the elemental (i.e. local) stiffness and mass matrices are found to be

$$\boldsymbol{k} = \frac{\boldsymbol{AE}}{\boldsymbol{L}}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad \boldsymbol{m} = \frac{\rho \boldsymbol{AL}}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \qquad (3\text{-}4.a\text{-}b)$$

In the derivations to follow, matrices are denoted by bold uppercase letters. Determining the global stiffness and mass matrices is tedious, and best left to automated procedures. In general, the global stiffness and mass matrices are determined in a 4-step process. Consider the stiffness matrix. First, the local stiffness matrix is found using Eq. 3-4.a. That matrix is then rotated into the global coordinate system using the coordinate transformation matrix **T** (defined below), and the partitions of the transformed matrix placed into the corresponding locations in the global stiffness matrix. The last step involves reducing the global stiffness matrix as required.

This process is demonstrated for ACOSS4 which consists of 12 elements connected through 10 nodes. Consider element 3, which is connected between nodes 2 and 4. Each node has 3 associated degrees of freedom, thus the overall stiffness matrix will be 30x30.

1) Determine the local stiffness matrix

$$\boldsymbol{k}_3 = \frac{\boldsymbol{A}_3\boldsymbol{E}_3}{\boldsymbol{L}_3}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad (3\text{-}5)$$

2) Determine the direction cosines and build transformation matrix **T**

$$\boldsymbol{T}^T = \begin{bmatrix} 0 & 0 & 0 & \boldsymbol{l} & \boldsymbol{m} & \boldsymbol{n} \\ \boldsymbol{l} & \boldsymbol{m} & \boldsymbol{n} & 0 & 0 & 0 \end{bmatrix} \qquad (3\text{-}6)$$

and transform the local stiffness matrix, $\boldsymbol{k}^3$, into global coordinates:

$$\hat{\boldsymbol{k}}^3 = \boldsymbol{T}\boldsymbol{k}^3\boldsymbol{T}^T \qquad (3-7)$$

$\hat{\boldsymbol{k}}$ is now a 6x6 matrix comprised of 4 partitioned 3x3 matrices. The partitioned matrices on the diagonal represent the contribution to stiffness at the nodes while the off-diagonals represent stiffness coupling between the two nodes

$$\hat{\boldsymbol{k}}^3 = \left[\begin{array}{c|c} \hat{\boldsymbol{k}}^3_{11} & \hat{\boldsymbol{k}}^3_{12} \\ \hline \hat{\boldsymbol{k}}^3_{21} & \hat{\boldsymbol{k}}^3_{22} \end{array}\right] \qquad (3-8)$$

where each $\hat{\boldsymbol{k}}_{ij}$ is a 3x3 matrix relating the stiffness term at that local node or between the local nodes. In this example, $\hat{\boldsymbol{k}}^3_{11}$, is the stiffness term of element 3 at node 2, while $\hat{\boldsymbol{k}}^3_{21}$ is the stiffness term of element 3 between nodes 2 and 4.

The global stiffness matrix, **K**, is then constructed of individual partitions of these local 3x3 matrices. It is built by substituting the corresponding partitions from $\hat{\boldsymbol{k}}^3$, as shown below. Note the local stiffness matrix $\hat{\boldsymbol{k}}^3_{12}$, which represents the stiffness between nodes *2* and *4* resident in *row 2* and *column 4* of the global matrix (where each global element is a 3x3 local matrix.)

$$\boldsymbol{K} = \left[\begin{array}{c|c|c|c|c} \cdots & \cdots & \cdots & \cdots & \cdots \\ \hline \cdots & \hat{\boldsymbol{k}}^3_{11} & \cdots & \hat{\boldsymbol{k}}^3_{12} & \cdots \\ \hline \cdots & \cdots & \cdots & \cdots & \cdots \\ \hline \cdots & \hat{\boldsymbol{k}}^3_{21} & \cdots & \hat{\boldsymbol{k}}^3_{22} & \cdots \\ \hline \cdots & \cdots & \cdots & \cdots & \ddots \end{array}\right] \qquad (3-9)$$

The global mass matrix is constructed in an analogous fashion. Both global matrices can then be reduced by eliminating those rows and columns where all elements of both

vectors are identically zero. The finite element software program, IDEAS™ [IDEAS, 1990:35-2] was used to construct these matrices. The program uses a graphical interface, where the operator places nodes in a 3-dimensional grid, and interconnects the nodes with various element types (e.g. axial, beam, mass, etc.) The internal software then generates the corresponding mass and stiffness matrices and reduces them as appropriate. These matrices are used in subsequent calculations, to include the determination of eigenvalues and their associated mode shapes.

Partial derivatives of the local stiffness and mass matrices (Eqs. 3-4.a and b, respectively), taken with respect to the element's area (the design variables in both ACOSS4 and COFS) are simply

$$\boldsymbol{k}' = \frac{\boldsymbol{E}}{\boldsymbol{L}}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad \boldsymbol{m}' = \frac{\rho \boldsymbol{L}}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \qquad (3\text{-}10.a\text{-}b)$$

The partial derivatives of the global stiffness and mass matrices are then formulated using an analogous 4-step process, but with Eqs. 3-10 replacing Eqs. 3-4. This procedure was actually completed manually for several ACOSS4 elements (augmented with MATLAB code written by the author) while investigating some suspect procedures in the original ***FRAME*** code [Canfield, 1992]. Finally, a similar 4-step procedure can be followed for beam elements [Craig, 1981:385-392].

3.3 The Eigenvalue Problem

In deriving the system's equations of motion Craig uses Hamilton's principle [Craig, 1981: 243-255] to show equations of motion for an unforced, undamped, n-degree system to be

$$\begin{aligned} m_{11}\ddot{u}_{11} + k_{11}u_{11} &= 0 \\ m_{22}\ddot{u}_{22} + k_{22}u_{22} &= 0 \\ &\vdots \\ m_{nn}\ddot{u}_{nn} + k_{nn}u_{nn} &= 0 \end{aligned} \tag{3-11}$$

These *n* equations are then simplified, and the resulting equations, in matrix form are

$$M\ddot{U} + KU = 0 \tag{3-12}$$

where

$$M = \begin{bmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & \ddots & \vdots \\ m_{n1} & \cdots & m_{nn} \end{bmatrix} \quad K = \begin{bmatrix} k_{11} & \cdots & k_{1n} \\ \vdots & \ddots & \vdots \\ k_{n1} & \cdots & k_{nn} \end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix} U_{11} & \cdots & U_{1n} \\ \vdots & \ddots & \vdots \\ U_{n1} & \cdots & U_{nn} \end{bmatrix}$$

This is the equation of a simple harmonic oscillator, one solution of which is

$$u = U\sin(\omega t - \alpha) \tag{3-13}$$

Substituting this solution into Eq. 3-12 yields the well-known eigenvalue problem

$$\left(K - \omega^2 M\right)U = 0 \tag{3-14}$$

Linear algebra shows the solution of this equation to be non-trivial only if the determinant of the coefficient matrix is identically 0, or

$$\det\left(K - \omega^2 M\right) = 0 \tag{3-15}$$

Upon expansion, the solution of Eq. 3-15 yields *n* natural frequencies, $\omega_1, \omega_2, \ldots \omega_n$. These values can then be individually substituted into Eq. 3-12 to yield the individual eigenvectors U_r. These eigenvectors, or natural modes, are not unique -- they can be scaled by any positive *or* negative constant and still satisfy Eq. 3-14

$$\phi_r = c_r U_r \qquad (3-16)$$

and Eq. 3-14 becomes

$$[K - \lambda M]\phi = 0 \qquad (3-17)$$

The ability to scale by any arbitrary coefficient allows the mode to be normalized -- scaled in such a manner that all the elements of a particular eigenvector ϕ_r, have a unique value. Defining the generalized mass,

$$M_r = \phi_r^T m \phi_r \equiv 1 \qquad (3-18)$$

mass normalizes the r^{th} eigenvector. Although other scaling algorithms are commonly used, the structures herein were mass normalized. One other key feature of eigenvectors is that of orthogonality. That is, each eigenvector is orthogonal to every other eigenvector [Craig, 1981: 303]. This property is the result of the linear independence of the set of eigenvectors.

Determining the mode shape, the shape the structure assumes when excited at one of its natural frequencies, is critical to structural analysis in all engineering designs. In many cases, an engineer may have to prevent a structure from assuming a particular shape when excited at certain natural frequencies. For example, engineers designing computer disc

drives may have to prevent certain mode shapes from occurring to ensure some frequency does not deform the platter into a shape which would drive the disk into the head.

Mode shapes are computed by a variety of methods. One of the most efficient methods, and the method used by both ***FRAME*** and IDEAS [SDRC, 1990: E-10-12], is simultaneous vector iteration (SVI). A full derivation can be found in the above reference. Basically, each iteration consists of an inverse power step followed by a subspace orthogonalization:

$$\Psi = \boldsymbol{K}^{-1}\boldsymbol{M}\Phi_{k-1} \qquad (3-19.a)$$

$$\Phi_k = \Psi\Lambda \qquad (3-19.b)$$

where Λ is a diagonal matrix of the eigenvalues and Ψ is a vector of length n. Eqs. 3-19 are iterated until

$$\Psi^T \boldsymbol{K}\Psi\Lambda = \Psi^T \boldsymbol{M}\Psi\Lambda\lambda \qquad (3-20)$$

at which time they yield the natural eigenvalues and eigenvectors.

Once the eigenvalues are determined, one sometimes finds a frequency is repeated. This occurs predominantly in structures exhibiting some type of symmetry. If a frequency is repeated , or nearly repeated, yet is associated with a unique eigenvector, a structure excited at that frequency can describe two totally distinct mode shapes. (Note: there currently exists no standard for determining when two adjacent frequencies are close enough to be considered repeated. Craig [Craig, 1981: 304] implies it could be as close as 1%, while [Bernard and Bronowicki, 1994: 1500] use 5% as their criterion, a percentage

considered high for space structure analysis). Control actuators using feedback sensors which report the frequency, may then induce forces to contain one mode when another is actually present, exacerbating the control problem. Repeated eigenvalues also cause numerous mathematical difficulties. Consider Eq. 3-17 as an example. It is the eigenvalue problem and by definition singular, with rank = $n-1$. Section 3.4.4 will show that Nelson was able to solve the matrix equation (with rank $n-1$) by condensing the matrix and thus eliminating the one singularity. [Nelson, 1976: 1202-4]. However, his very same technique fails in the presence of repeated eigenvalues (which drive the rank lower).

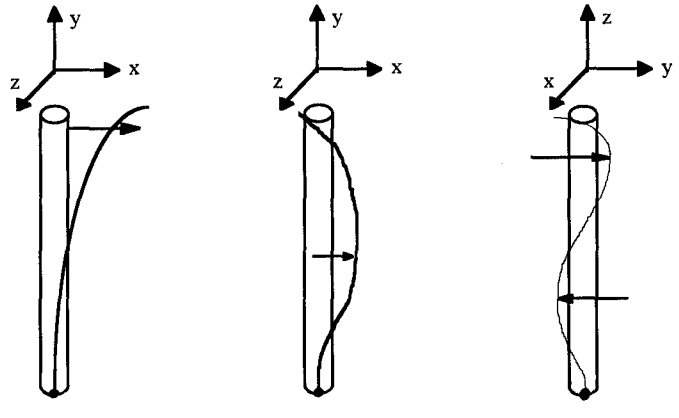
An eigenvector represents the physical displacement an element undergoes when that element is excited at the vector's corresponding natural frequency. Each element of the eigenvector in turn describes the physical displacement of one degree of freedom of that element. In fact, natural modes are in one sense, simply one discrete sample of a vibration. A vibration is a time-variant phenomena -- it is simply the time history of the set of displacements of the structure's degrees of freedom. That time history unfolds over a period governed by the natural frequency and the structure deforms into a shape corresponding to that natural frequency. That deformation then repeats.

Engineers have designed a unique nomenclature to describe the modes as they occur in structures. Some modes are simple -- that is, they consist entirely of one type of motion. More complex modes may consist of the superposition of multiple, simple modes. No such standard naming convention exists for 3-dimensional objects. Attempts to analogize modes occurring in 3-dimensional objects with those occurring in two-dimensional objects is possible for simple structures at its lower frequencies. Identification

of higher modes, however, which tend to consist of coupled motions and occur in multiple planes, soon becomes unique to each structure. Such a naming convention was determined for ACOSS4. (Section 4.3)

Some of the more common mode modal shapes for two dimensional structures are described below where the diameter of the rod is considered much smaller than the length. Although any undergraduate text in structural mechanics could serve as a reference here, mode identification is a major component of this research effort. Thus the mode shapes are reproduced here where they can better serve as a proper reference for the 3-dimensional mode identification to follow.

Bending motion refers to cantilevered motion, where at least one node exists. (Note: A *node* is a point which is stationary under excitation. It does not have the same meaning as the term node in finite element nomenclature which refers to the point at which two elements are connected and is therefore more accurately termed a *joint*.) Bending modes are numbered ordinally by the number of nodes which occur during the deformation. These modes are also given a designation indicating the plane into which they deflect. The



1st XY Bending (1-XY or 1X) | 2nd XY Bending (2-XY or 2X) | 3rd YZ Bending (3-YZ ro 3Y)

Figure 3-2: Bending Modes

designation can be a plane designation, or if the plane is known by default, one letter can serve as the reference. The eigenvectors for the figures above could be vectors of length 5, where the mode corresponding to 1st Bending would be $[0\ .1\ .28\ .52\ .79]^T$. The next two bending modes could have vectors $[0\ .1\ .3\ .1\ 0]^T$ and $[0\ -1\ 0\ +1\ 0]^T$. Other common modes are axial modes (compression and extension) and torsional modes, shown in Figure 3-3:

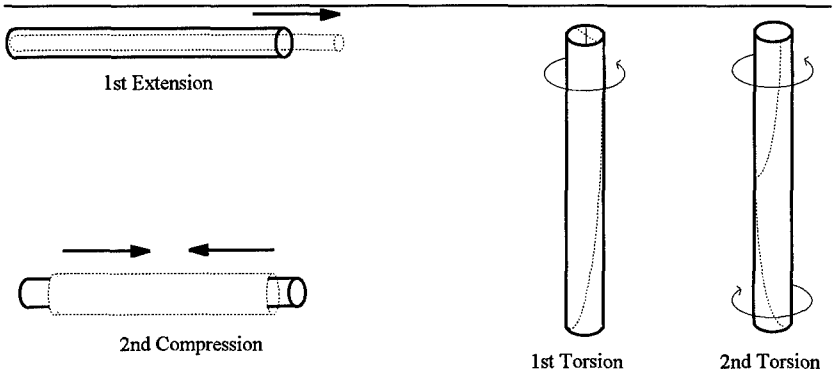


Figure 3-3: : Axial and Torsional Modes

Finally, these eigenvectors, which are assumed to have been mass-orthogonalized in accordance with Eq. 3-18, can be placed columnwise into a single matrix. This matrix is then termed the *modal matrix* given as

$$\Phi = [\phi_1 \quad \phi_2 \quad \cdots \quad \phi_n] \qquad (3-21)$$

It can be shown that the modal matrix spans the entire subspace. [Fox and Kapoor, 1968: 2426] Using the definition of the modal matrix along with Eq. 3-18, one can mass orthogonalize the system of equations to find the modal mass matrix

$$\boldsymbol{M} = \Phi^T \boldsymbol{m}\, \Phi \qquad (3-22)$$

This equation is the basis for the cross-orthogonality checks reviewed in Section 2.6.2 and the mode-swapping routine developed in Chapter 4. Note that the modal mass matrix is diagonal as a result of the linear independence of the eigenvectors. Using Eq. 3-22, consider the k^{th} iteration in an optimization problem which results in a mass matrix $\boldsymbol{m}^k$ and a modal matrix $\boldsymbol{\Phi}^k$. Replace the first modal matrix in Eq. 3-22 by a slightly perturbed modal matrix

$$\boldsymbol{M}^k = \left[\Phi^{\varepsilon}\right]^T \boldsymbol{m}^k \Phi^k \qquad (3-23)$$

where

$$\Phi^n = \begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1n} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nn} \end{bmatrix}$$

and

$$\Phi^{\varepsilon} = \begin{bmatrix} \phi_{11}+\varepsilon_{11} & \phi_{12}+\varepsilon_{12} & \cdots & \phi_{1n}+\varepsilon_{1n} \\ \phi_{21}+\varepsilon_{12} & \phi_{22}+\varepsilon_{22} & \cdots & \phi_{2n}+\varepsilon_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{n1}+\varepsilon_{1n} & \phi_{n2}+\varepsilon_{n2} & \cdots & \phi_{nn}+\varepsilon_{nn} \end{bmatrix}$$

Equation 3-23 should still result in a diagonally dominant matrix if all elements of the perturbation matrix, $\boldsymbol{\varepsilon}$, are very small in relation to the elements of Φ. This can be seen if one takes the limit as the matrix of perturbations approaches the null matrix

$$\lim_{\varepsilon_{ij} \to [0]} \Phi^{\varepsilon} \to \Phi^n \qquad (3-24)$$

which is a diagonal matrix in accordance with Eq. 3-19. The cross-orthogonality checks consider the modal matrix from the previous iteration to be a slightly perturbed version of the current modal matrix , i.e. $\Phi^{\varepsilon} = \Phi^{k-1}$. Note that this method is highly dependent on

the adjacent iterations being only slightly perturbed versions of one another. Optimization routines which allow large deviations between iterations violate this fundamental principle and thus will not work.

In computing the CORC matrix of an optimization problem, one must decide which mass matrix to use in the computation. Averaging, or using a weighted sum of the two mass matrices may be an appropriate choice. This was not, however, an issue in the ACOSS4 problem because the non-structural mass was several orders of magnitude greater than the structural mass used in Eq. 3-23.

3.4 Eigensolution Sensitivity

Sensitivity in systems indicates the degree to which the solution changes upon introduction of small perturbations. Very sensitive systems will undergo large changes when their design parameters are varied only slightly, while the parameters of insensitive systems can undergo major changes with little effect on the final output. Sensitivity therefore indicates how well an analytical model will work in the real world where non-linearities and noise could cause overly sensitive systems to fail. Sensitivity to modifications induced by an optimization algorithm can be determined using derivatives taken with respect to design variables (e.g. length, area, moduli, etc.). [Ojalvo, 1986:1-3] One common method is to determine the gradients of the eigenvectors with respect to some system parameter. There are several methods to determine these eigenvector derivatives. The following briefly outlines three complete methods and one hybrid method used for sensi-

tivity analysis in this thesis. The finite difference approximation and two related hybrid approaches based on semi-analytical derivatives is first presented. The two analytical methods, modal expansion and Nelson's Method, then follow.

3.4.1 Finite Difference Approximation

One method to determine eigenvector derivatives is by finite difference approximation. This method uses difference equations vice differential equations, perturbing one design variable at each iteration to find the corresponding change in the eigenvector

$$\frac{\partial \phi}{\partial k_i} \approx \frac{\Delta \phi}{\Delta k_i} \qquad i = 1,2,\cdots,n \qquad (3-25)$$

Once the change in the eigenvector due to the perturbation in the i^{th} variable is calculated, the k^{th} design variable is reset to its initial value and then the next design variable perturbed. This operation continues until each design variable has been perturbed separately. The individual effects of the perturbations on the eigenvector are then combined to determine the eigenvector gradient.

There are several drawbacks to this method. First and foremost, it requires a finite element re-analysis for *each* design variable (as well as one for the initial analysis.) For example, the ACOSS4 structure has eighteen design variables -- twelve cross-sectional areas and six control gains. An optimizer programmed to use finite difference derivatives thus must analyze nineteen *different* finite element models -- one initial analysis and one *additional* analysis for *each* design variable. This strains computer resources for all but simple models.

In addition, the use of IMSL routines [IMSL, 1991: 1110-16] to determine finite difference gradients restricts the analyst. End-users have no vehicle available to change the method, or even the parameters by which IMSL's finite difference routines compute the gradients. This restriction prevents using optimal finite difference step sizes; in fact the user has no control over the step size, a critical component to the solution, *at all.* IMSL bases step size on the machine precision and comes with a warning of "possible poor performance" [IMSL, 1991: 1118]. If the design is highly sensitive, a large step will invalidate the fundamental concept of the finite difference method -- that it can approximate the tangent to a function by a secant division. Alternatively taking too small a step may leave a function stepping at a rate comparable to the level of noise in the system inducing oscillations in a non-feasible sector of the design space. In addition, even non-optimal step sizes introduced into highly sensitive problems could result in irrecoverable path deviations. Finally, users are unable to monitor the progress of the difference operation as it occurs because all computations are imbedded within the proprietary code. Although computationally intensive, these difference approximations do provide accurate eigenvector derivatives when provided the **appropriate** step size.

3.4.2 Semi-Analytic Methods

One hybrid approach to sensitivity analysis incorporates elements of both analytic differentiation and finite differencing. The form of the partial derivative equation for a constraint $\mathbf{g}$, (e.g. actuator forces, constraint forces, quadratic performance index, etc.) in the integrated structural design and control problem is

$$\frac{\partial g}{\partial X} = \begin{Bmatrix} 0 \\ \hline \partial g / \partial x_c \end{Bmatrix} + \begin{Bmatrix} \partial g / \partial x_s \\ \hline 0 \end{Bmatrix} \qquad (3-26)$$

where x_c is a design variable in the control formulation (e.g. an actuator gain) and x_s is a structural design variable (e.g. a cross-sectional area.)

The equation in this form can be comprised of elements from two independent formulations -- one analytic and one finite difference. If the first term is differentiated analytically, and the second term replaced with its finite difference counterpart, Eq. (3-23) becomes

$$\frac{\partial g}{\partial X} = \begin{Bmatrix} 0 \\ \hline \partial g / \partial x_c \end{Bmatrix} + \begin{Bmatrix} \Delta g / \Delta x_s \\ \hline 0 \end{Bmatrix} \qquad (3-27)$$

This is the Semi-Analytic-*Constraint* version or SAC. Likewise, if the second term is computed via the chain rule, and the second factor replaced with its finite difference counterpart

$$\frac{\partial g}{\partial X} = \begin{Bmatrix} 0 \\ \hline \partial g / \partial x_c \end{Bmatrix} + \begin{Bmatrix} \frac{\partial g}{\partial \Phi} \bullet \frac{\partial \Phi}{\partial x_s} \\ \hline 0 \end{Bmatrix} \qquad (3-28a)$$

and

$$\cong \begin{Bmatrix} 0 \\ \hline \partial g / \partial x_c \end{Bmatrix} + \begin{Bmatrix} \frac{\partial g}{\partial \Phi} \bullet \frac{\Delta \Phi}{\Delta x_s} \\ \hline 0 \end{Bmatrix} \qquad (3-28b)$$

where $\bullet$ is defined as an inner product $\sum_{i}^{n} \sum_{j}^{m} \frac{\partial g}{\partial \phi_{ij}} \frac{\partial \phi_{ij}}{\partial x_k}$

then the formulation is termed a Semi-Analytic *Differentiation* operation. These semi-analytic methods were used as investigative tools to distinguish which terms were in error.

Section 3.4.3 Modal Expansion Method

Bernard and Bronowicki recently extended an analytical method put forth by Fox and Kapoor in 1968 to calculate derivatives of eigenvectors based on the modal expansion theorem. Their method allows determination of eigenvector sensitivity in the cases of repeated eigenvalues as well as in the case of repeated eigenvalue derivatives. Recall that the eigenvectors of a mass normalized *n x n* modal matrix are orthogonal to each other, thus they are all linearly independent. [Strang, 1988: 80-83] Fox and Kapoor showed that because these vectors also span the vector space, they form a basis and thus any *n*-dimensional vector can be represented by the linear combination of the vectors of that basis -- to include the derivatives of the eigenvectors. The full derivation can be found in the paper by Fox and Kapoor [Fox and Kapoor, 1968:2426-29], and is extended as noted above by Bernard and Bronowicki [Bernard and Bronowicki, 1994: 1500-1506].

The modal expansion theorem states

$$\phi_i' = \sum_{j=1}^{n} c_{ij} \phi_j \qquad (3-29)$$

Taking the derivative of the eigenvalue problem in Eq. 3-17 with respect to a design variable, x_k, , denoted as $\frac{\partial}{\partial x_k}(\cdot) \equiv (\ )'$, yields

$$[K - \lambda_i M]'\phi_i + [K - \lambda_i M]\phi_i' = 0 \qquad (3-30)$$

Premultiplying by ϕ_i^T and noting the symmetry of **K** and **M** in Eq. 3-17 yields

$$\phi_j^T \left[K' - \lambda_i M' - \lambda_i' M \right] \phi_j = 0 \qquad (3-31)$$

Finally, the mass-orthogonality condition (Eq. 3-18), can be used in conjunction with Eq. 3-31 to find an expression for the partial derivative of the eigenvalue

$$\lambda_i' = \phi_i^T [K' - \lambda_i M'] \phi_i \qquad (3-32)$$

Expressions for the constant c_{ij} are now required to find the partial derivative of the eigenvector with respect to an element area. Substituting Eq. 3-29 into Eq. 3-30 and premultiplying by ϕ_j^T for $j \neq i$, one finds

$$c_{ij} = \frac{\phi_j^T \left[K' - \lambda_i M' \right] \phi_i}{\lambda_i - \lambda_j} \text{ when } i \neq j \qquad (3-33)$$

To find c_{ij} for i=j, the mass normalization equation (Eq. 3-22) is differentiated with respect to the k^{th} design variable resulting in

$$c_{ii} = \frac{-\phi_i^T M' \phi_i}{2} \qquad (3-34)$$

This formulation was used to determine the eigenvector derivatives which led to the discovery of the original error in the implementation of Nelson's method in ***FRAME***

[Canfield, 1992], as explained in Section 4.9. The complete ForTran 77 code implementing modal expansion is included in Appendix F.3.

Section 3.4.4 Nelson's Method (with Extensions)

Nelson's method is the preferred approach for gradient evaluation when not all eigenvectors are available (as in the case of large systems) and when eigenvalues are distinct. In 1976, Nelson proved that computation of the eigenvalue and eigenvectors derivatives could be simplified by overcoming the inherent singularity in the eigenvalue problem. Nelson's solution consists of a homogeneous and particular solution. [Nelson, 1976:1201-3].

Expanding Eq. 3-30 and rearranging yields

$$\left[\boldsymbol{K} - \lambda_j \boldsymbol{M}\right]\phi_j{}' = \left[\lambda_i{}' \boldsymbol{M} + \lambda_j \boldsymbol{M}' - \boldsymbol{K}'\right]\phi_j \qquad (3-35)$$

While Eq. 3-32 provides the j^{th} eigen*value* derivative, Eq. 3-35 cannot be solved directly for the eigen*vector* derivative -- by definition, its matrix coefficient is singular and thus cannot be inverted. Further manipulation is required. Nelson writes the eigenvector derivative as a sum of a particular and a homogeneous solution

$$\phi' = \boldsymbol{V} + c\phi \qquad (3-36)$$

Nelson next assumes all eigenvalues are unique (i.e. the rank of **M** is *n-1*). The row and column where the singularity exists are eliminated and the corresponding component of V zeroed. The resulting equation is just a nonsingular linear matrix equation which can be solved for $\tilde{V}$

$$\left[\tilde{K} - \lambda \tilde{M}\right]\tilde{V} = [\tilde{K} - \lambda_j \tilde{M}]'\tilde{\phi}_j \qquad (3\text{-}37)$$

where ~ denotes a reduced matrix or vector. Eq. 3-37 is solved for $\tilde{V}$, the particular solution. The eigenvector coefficient, c, can be computed by differentiating the equation for mass normalization (Eq. 3-18) to find

$$c = -\phi^T M V - \frac{1}{2}\phi^T M' \phi \qquad (3\text{-}38)$$

The results of Eqs. 3-37 and 38 are substituted back into Eq. 3-36 and the derivative of the j^{th} eigenvector determined.

Nelson's method can only be used in cases where the eigenvalues are distinct as Eq. 3-20 depends on the existence of only one singularity. Several authors have submitted methods to extend Nelson's method to the case of repeated eigenvalues. Ojalvo's [Ojalvo, 1986: 1] original method was found to only work in the case of diagonal mass and stiffness matrices, as pointed out in separate papers by both Daily [Daily, 1989: 486] and Mills-Curran [Mills-Curran, 1994:867-8]. Daily solves the problem of repeated eigenvalues by differentiating the eigenvalue equation (Eq. 3-17) twice. The approach by Mills-Curran sets up a secondary eigenvalue problem. It was the method intended for use in this research had repeated eigenvalues been found to be the cause of the problem as initially expected. A full derivation can be found in the above reference and a summary as it applies to this research effort follows.

Eigenvalues appearing *p* times result in a system of rank *n-p*. If the eigenvalue, λ_j, appears twice for example, any linear combination of the two eigenvectors is also an eigenvector. A unique pair of eigenvectors corresponding to the repeated eigenvalue must

first be determined before the eigenvalue derivative can be computed. This requirement is apparent when the two equations for the eigenvalue derivative (Eq. 3-32) are viewed together

$$\lambda_j' = \phi_j^T \left[\boldsymbol{K}' \phi_j - \lambda_j \boldsymbol{M}' \phi_j \right] \phi_j \qquad (3\text{-}39.a.\text{-}b.)$$

and

$$\lambda_{j+1}' = \phi_{j+1}^T \left[\boldsymbol{K}' \phi_{j+1} - \lambda_{j+1} \boldsymbol{M}' \phi_{j+1} \right] \phi_{j+1}$$

Note that the derivatives are ambiguous, because either eigenvector can be used in these equations. The correct eigenvector to use in each case is a linear combination of the two original eigenvectors

$$\bar{\phi}_j = \Phi \begin{bmatrix} a \\ b \end{bmatrix} \quad \text{and} \quad \bar{\phi}_{j+1} = \Phi \begin{bmatrix} c \\ d \end{bmatrix} \qquad (3\text{-}40a.\text{-}b.)$$

where $\Phi \equiv \left[\phi_j \quad \phi_{j+1} \right]$. All that is required now is the determination of the vector constants. Substituting Eq. 3-40a. into Eq. 3-31 and simplifying yields

$$\Phi^T (\boldsymbol{K} - \lambda_j \boldsymbol{M})' \Phi \mu + \Phi^T (\boldsymbol{K} - \lambda_j \boldsymbol{M}) \Phi' \mu = 0 \qquad (3\text{-}41)$$

where μ is a vector representing $[a\ b]^T$ or $[c\ d]^T$ in Eqs. 3-40. To simplify, recall that $(\mathbf{K}\text{-}\lambda_j\mathbf{M})\Phi = 0$. Expand the derivative, noting that $\Phi^T\mathbf{M}\Phi = [\mathbf{I}]$ to find

$$\left[\Phi^T \left(\boldsymbol{K}' - \lambda_j \boldsymbol{M}' \right) \Phi - \lambda_j' \Phi^T \boldsymbol{M} \Phi \right] \mu = 0$$

or

$$\left[\Phi^T \left(\boldsymbol{K}' - \lambda_j \boldsymbol{M}' \right) \Phi \right] \mu = \lambda_j' \mu \qquad (3\text{-}42)$$

which is just a new eigenvalue problem. [Mill-Curran, 1988: 867-9] Once the constant vectors are determined, the correct eigenvector satisfying Eq. 3-35 can be determined. While this approach does not work for systems with rigid body modes, such modes in the structures studied were not computed and thus the approach is valid. This approach was studied and outlined for insertion into FRAME when it appeared a mode swap (Section 4.5) had indeed been the cause of the original failure of the solution equipped with analytical derivatives.

3.5 Non-Linear Optimization Theory

The basic goal of an optimization problem is to find some extremum of a given design. In the context of the integrated structural design and control problem, that extremum is usually multidisciplinary -- for example, solving for minimum weight while at the simultaneously expending the minimum control energy. The design parameter to be minimized is termed the *objective function*. The objective function is minimized by numerically iterating a set of design parameters according to some computational algorithm until the design goal is reached. The algorithm minimizes the objective function in strict adherence to a number of design *constraints*. *Equality* constraints require that a function of the design variables match a specified value exactly, while inequality constraints place an upper (or lower) bound on a function of the design variables. *Side* constraints restrict the design variable directly. The algorithm need not meet all con-

straints at each iteration but must do so on the final iteration before a design is considered *feasible*. [Vanderplaats,1984:1-10]. The mathematical equivalence is shown in Table 3-1:

Table 3-1: Optimization Formulation

Action	Nomenclature	Mathematical Expression	Range	Equation
Minimize	Objective Function:	F(**X**)		(3-43a)
by iterating on	Design Vector	**X**=[$X_1, X_2, \ldots X_n$]	i = 1 . . . n	(3-43b)
According to:				
	1) Inequality Constraints	$g_i(X) \leq 0$ or $g_i(X) \geq 0$	i = 1 . . . m	(3-43c)
	2) Equality Constraints	$h_j(X)=0$	i = 1 . . . j	(3-43d)
	3) Side Constraints	$X_k^L \leq X_k \leq X_k^U$	i = 1 . . . k	(3-43e)

Using ACOSS4 as an example, the objective function is to minimize mass. The design vector is the set comprised of cross sectional areas and actuator gains and thus has a length of eighteen. It has an equality constraint, forcing the lowest frequency to be 1.341 rad/sec, and an inequality constraint restricting the second lowest frequency to values less than 1.5 rad/sec. In addition, there are constraints on the actuators as well as on the Quadratic Performance Index.

A program iterates the set of design variables according to an algorithm which computes a search path. This search path is constructed in two phases. A search direction must first be found, and then the step size in that direction determined [Vanderplaats, 1984: 10-11]. ***FRAME*** uses a line search algorithm

$$X^q = X^{q-1} + \alpha^* S^q \qquad (3-44)$$

where

q ≡ iteration number
α^* ≡ step size
S ≡ search direction

The optimizer uses eigenvalue derivatives and eigenvector gradients (Section 3.4) to determine **S** and α^*. The new vector **S** is scaled by the step size and the result added to the old search vector $\mathbf{X}^{q-1}$. The largest step size allowed in this implementation is unity. The optimizer is particularly susceptible to repeated frequencies because it requires an initial direction in which to search. Specifically, the mode swapping condition, or passing through a repeated frequency, causes two problems:

1) During the line search, modes may become associated with the wrong control gains. The control force generated by the solution to the integrated problem may then be applied to the wrong mode, exacerbating the control problem.

2) The gradients may become discontinuous thereby corrupting the approximation to the Hessian. *NCONG* and *NCONF* [IMSL, 1991: 1096-97, 1103-06] are the IMSL sub-routines used to solve a general non-linear programming problem using analytical or finite difference derivatives respectively. Both routines are based on a sequential (or successive) quadratic programming algorithm. SQP requires the problem functions (the eigenvector derivatives here) be continuously differentiable. The method finds a search direction based on solving a subproblem using a quadratic objective. Constraints are reformulated as linear. The optimization subproblem becomes

$$\text{Minimize: } Q(S) = F(X) + \nabla F(X) \bullet S + \frac{1}{2} S^T B S \qquad (3-45)$$

Here, **S** is the design vector, and **B** is the approximation to the Hessian matrix of the Lagrangian. [Vanderplaats, 1984: 195-197]. (Lagrangian multipliers were used as weighting factors to augment the objective function with the constraint functions. A full derivation can be found in the preceding reference.) Eq. 3-45 is used to find the search direction. IMSL [IMSL, 1991:1104] initially assumes a unit step size and updates the design vector **X**. The new design vector is used in a subsequent eigenanalysis, the constraints are again determined, and a new step size, α^*, is determined using a quadratic extrapolation. The key point in this summary is that the Hessian is updated by the constraint gradients which in turn are based on *eigenvector derivatives*. If these derivatives become misaligned due to a mode swap, the Hessian will not be updated correctly, and the search direction will be corrupted. This corruption is carried along on every subsequent iteration. If it happens early enough in the iteration history (where its effects will be greater), or its effects are not minimized over the next several iterations, the optimizer may be unable to recover. This problem is explained graphically in Figure 3-4.

The figure shows the ambiguity hampering the optimizer upon checking for a gradient at the point where the modes cross. At this point, the eigenvalues are equal. If the optimizer had been iterating along the λ_2 trajectory when a mode swap occurs, and then instead continues along the original λ_1 trajectory as a result of the ambiguity, the optimizer sees a discontinuity in the gradient -- a cusp occurring at the mode crossing. Even if the optimizer can continue through the discontinuity, it will now update the Hessian approximation with disassociated eigenvector derivatives and the search direction will lose its

integrity. If the optimizer can re-establish the correct trajectory, this one deviation may not be enough to prevent eventual convergence (See Section 4.8.1). This is particularly true if the Hessian is corrupted late in the iterative process, where the impact will not be as significant as if it had occurred earlier in the process. However, if the optimizer never re-establishes the correct trajectory in subsequent iterations, the solution will not converge.

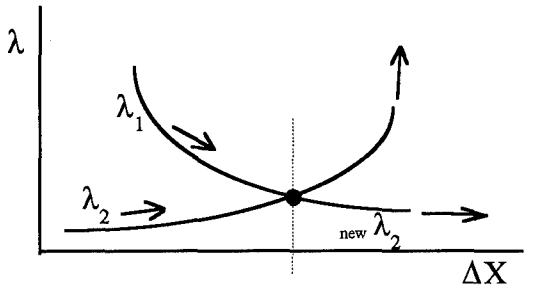


Figure 3-4: Effect of Modal Crossing During Optimization

IV. Results

4.1 Overview

This chapter details the results of the research effort. The goal of the research is first re-presented. Section 4.2 then explains how the initial research path, that of investigating the mode swapping condition, was logically chosen over other paths. Section 4.3 details the initial mode characterization of the primary structure being researched, ACOSS4. Sections 4.4, 4.5, and 4.6 use that work to determine qualitatively and quantitatively that the mode swapping condition exists. A new algorithm to detect and neutralize the mode swapping condition is presented in Section 4.7 and its superiority over existing methods proven by example. Because the results of the mode swapping condition are shown to be inconclusive, additional methods of sensitivity analysis are presented in Sections 4.8-4.11. The issue of mode swapping is then re-addressed in Section 4.12. The last section reports the results of the attempt to optimize the second structure, COFS-I, using the corrected analytic methods.

As mentioned previously, while researching multiobjective optimization as a method for solving the integrated structural design and control problem, Canfield experienced some difficulty in using analytic derivatives of eigenvectors (computed using Nelson's Method) to determine optimum search directions for ACOSS4. Specifically, using these derivatives resulted in non-convergence and the optimization problem could not be solved as originally intended. The optimizer did converge to the correct solution when equipped with derivatives

computed by finite difference methods. Feeling the original non-convergence (using the analytic derivatives) was a result of modes switching their order during the optimization process, Canfield retained his original data runs for investigation at a later time. The goal of this present research was to determine the validity of his hypothesis. If the mode swapping condition was indeed found to have caused the problem, its effects were to be neutralized. If the condition was not present, or was present but not the cause of the problem, additional research would be conducted to determine the actual cause and possibly neutralize it.

4.2 Formulation of the Problem

Canfield's methodology and original data runs were first reviewed to determine the best research path. It was concluded from this review that the original solution may have failed as a result of four possible problems:

1) The numerical accuracy of the eigensolver could not correctly distinguish the modes of closely spaced (nearly repeated) frequencies during the optimization.

2) The mixed use of single and double precision calculations may have affected the accuracy of the analytic derivatives.

3) A mode swap occurred and corrupted the optimization solution.

4) Two modes may not have actually *swapped* places, but one may have transformed into a mode completely distinct from anything in previous iterations and this mode was therefore untrackable.

The possibilities that each of these problems could individually be the cause of the failure was discussed. A preliminary review showed that any *one* of these four possibilities

would consume the bulk of the allotted research time -- thus the most likely cause of the original problem had to first be determined to ensure the highest probability of success. The effort required to investigate each of these possible causes was outlined and balanced with the likelihood that each was *individually* the cause.

1) Numerical Accuracy of the Eigensolver: As detailed in Chapter 1, space structures are characterized by high modal density. Many of the frequencies may be only fractions of a hertz away from two or three neighboring frequencies yet all of them will correspond to different mode shapes -- vibration modes the control engineer is trying to actively suppress. The eigensolver used was from the IMSL library, a comprehensive assortment of ForTran 77 routines used for mathematical applications [IMSL, 1991]. The eigensolver's computations were done in 32-bit (double) precision, which should have provided the necessary discretion even for this problem. It was determined that if the eigensolver was at fault, it was an indirect correlation only -- i.e. it may have failed as a result of being fed an inaccurate three-dimensional model. The possibility of such an input error was determined to be slim.

2) Mixed Precision: The high modal density inherent to these structures demands a much higher level of precision. The ***FRAME*** [Canfield, 1992] code used consists of over 13,000 lines of code comprising more than 130 separate sub-routines ranging from simple dynamic memory allocation routines to more sophisticated routines to determine the effects of wind gusts on aeroelasticity models. Many routines use single precision matrices. Other routines parameterize these same matrices as input and output vectors. Due to subsequent calculations with double-precision matrices, these single precision routines must first be

converted into double precision numbers. In some cases, the subsequent calculations must then be re-converted for compatibility.

These conversions introduce some error into the calculations -- error which could be enough to corrupt the eigenvector derivatives (computed in single precision) of adjacent modes (computed in double precision). While this problem is well understood, fixing it would have required phenomenal effort since virtually all 130 subroutines would be affected -- subroutines which the author had little experience with, the result of which would be what was deemed an unhealthy involvement level by the thesis advisor. More importantly, the likelihood that this was the main cause of the problem was determined to be slim.

3&4) Modal Transformation or Modal Swaps: Determining that the solution based on analytic derivatives failed to converge was due to either a modal transformation or a modal swap required virtually the same type of investigation and the same level of effort. Only the eventual solution was different. (Note: This thesis defines modal *transformation* as the case where an entirely new mode suddenly appears in the iteration history). Literature on the subject was reviewed and the small differences in the design variables between the unsuccessful run and the successful run over several iterations studied. It was surmised that the most likely cause of the unsuccessful run was due to a modal swap early on in the optimization process (i.e. within the first six iterations). Unfortunately, a mathematical proof showing the modes did indeed swap would certainly take the bulk of the allotted research time.

Proving the analytic breakdown was a result of 1) or 2) vice 3) or 4) would result in totally different solutions, only one of which could be completed in the time allotted. Because it was believed that the most probable cause was the modal swap, it was determined that

proving or disproving this should first be done. Our sponsors in the Flight Dynamics Directorate at Wright Laboratories were consulted and it was decided that prudence dictated the modal swap should first be proven *visually* using FEM models. If the modal swap could be proven visually, using the allotted research time to then *mathematically* prove the mode swapping condition existed would be justified. At that point, additional mode tracking code could be written and implemented into ***FRAME***, the ACOSS4 model re-run, and optimal convergence (hopefully) achieved. This approach combined the best of both worlds -- the most probable cause would be investigated first, and most importantly, by first verifying the modal swap visually instead of plowing head-on into a difficult, time-consuming mathematical solution, time to pursue fixes consistent with the first two hypothesis would still be available if no mode swap could be identified visually.

4.3 Development

The IDEAS Master Series™ software [SDRC, 1990] resident on the Silicon Graphics Workstations was used to verify the modal swap visually. This FEM software proved critical to this type of identification. Visually identifying the modes required a five step process:

1) A thorough understanding of mode-identification theory detailed in Chapter 3 including the creation of a naming convention which was both robust and acceptable to the structures community,

2) Building the nominal ACOSS4 FEM and matching the results in literature,

3) Building the FEMs at each subsequent iteration,

4) Identifying the modes of each iteration using the naming convention from Step 1,

5) Determining if a mode swap occurred by assimilating the results of Step 4.

Some difficulty in creating a robust naming convention for structural modes in a three-dimensional model was experienced. While the naming convention for *one* dimensional models is well-known and well understood in the structures community, a naming convention for multi-dimensional structures, which must account for activity in multiple planes is not so generally accepted. Completing Step 2) *before* Step 1 was therefore necessary to get a feel for an acceptable naming convention.

4.3.1 ACOSS4 Finite Element Model

The nominal ACOSS model was built using the finite element data in Appendix A.1. Resulting natural frequencies and mode shapes are included in Table 4-1. These results agreed to the original results published by Draper Labs to four significant digits (only four significant digits were provided) [Strunce, 1980:7]. The modal shapes computed also agreed with those provided by the Draper Labs. This effort confirmed an accurate model had been built.

Table 4-1: ACOSS4 Nominal Design Results

Mode Number	Mode Name	Frequency (hz)
1	1st Bending (Out-of-Plane)	0.2135874
2	1st Bending (In-Plane)	0.2649485
3	1st Rocking (In-Plane)	0.4600698
4	2nd Bending (Out-of-Plane)	0.4706857
5	2nd Rocking (In-Plane)	0.5408387
6	1st Torsion	0.6691620
7	1st Shearing (Out-of-Plane)	0.7419893
8	1st Shearing (In-Plane)	0.7568212
9	1st Compression (In-Plane)	1.359087
10	1st Compression (Out-of-Plane)	1.472268
11	1st Extension + 1st Torsion	1.636868
12	1st Breathing	2.053906

4.3.2 Output Display

Two techniques were employed to accurately characterize the mode shapes. The first technique used IDEAS' built-in capability to animate modes. This was critical to those less-experienced in mode identification and provided the determining factor in identifying many of the coupled modes. The second technique used multiple view ports of the ACOSS4 structure during its deformation. Using multiple views allowed a better representation of a three-dimensional structure in two dimensions. In addition, using multiple views allowed the researchers to focus on a particular characteristics of motion (e.g. out-of-plane motion, in plane motion, etc.) by using different viewing angles to alternately emphasize and de-emphasize these characteristics.

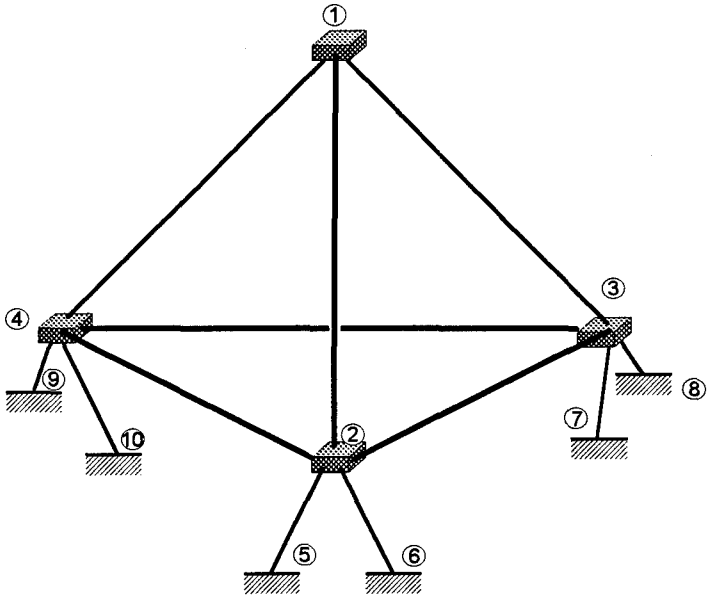


Figure 4-1: ACOSS4 Nominal Design

Because the nominal ACOSS4 has one strong member in the upper tetrahedron, it provided a convenient reference point. The nominal ACOSS4 structure is repeated in Figure

4-1. Plots of the nominal ACOSS4 structure shown in Figure 4-10 required only three viewports due to the structure's inherent symmetry, (i.e. a fourth view provided no additional information.) Motion perpendicular to the strong member was termed *out-of plane* motion, and motion parallel to the beam was termed *in-plane* motion.

The top view provided the most information to independently characterize the modes and was therefore allocated the larger viewport on the left for the nominal design. The plot in the top right corner of the diagram is the front view and was used to highlight out-of-plane motion. The plot in the bottom left corner is the side view and is used to highlight in-plane motion. This ability to emphasize and de-emphasize is clearly shown in Figure 4-10d. This view shows pure out-of-plane bending motion -- this purity comes again from the symmetry of the nominal design. The second view clearly indicates a large out-of-plane component while the third view (the in-plane view) shows little deformation at all and provides no useful information for characterization.

For subsequent iterations where significant modal coupling could have caused ambiguity in mode identification, a fourth view was added. This view effectively rotated the structure so that its plane of motion was coincident with the plane of the paper. The rotation scheme to obtain the view was dependent on the whether the motion was predominantly in-plane or out-of-plane as detailed in Figure 4-2. The axes listed are the screen axes -- not the structure's axes. The angle α is the angle between the strong member and *the line of apses* (a line drawn between the nominal and displaced apexes). This view provided additional insight into the most prominent mode shapes of significantly coupled modes.

TOP VIEW	$X=0^0$ $Y=0^\circ$ $Z=150^\circ$	**FRONT VIEW**	$X=270^0$ $Y=60^\circ$ $Z=0^\circ$
SIDE VIEW	$X=90^0$ $Y=30^\circ$ $Z=180^\circ$ PHASE: **IN** : Node 4 (↑←) & Apex(↑→) : **OUT** : Node 4 (↓←) & Apex(↓→)	**DIRECT VIEW**	$\lvert\alpha\rvert<45^0$: 90, (30-α), 180 $\lvert\alpha\rvert>45^0$: 270, [60-comp(α)], 0

Figure 4-2: View Angle Geometry

In Figure 4-3, the six bipods, the boundary conditions and the lumped masses have been removed for clarity. The medium solid lines represent the undeformed structure in that

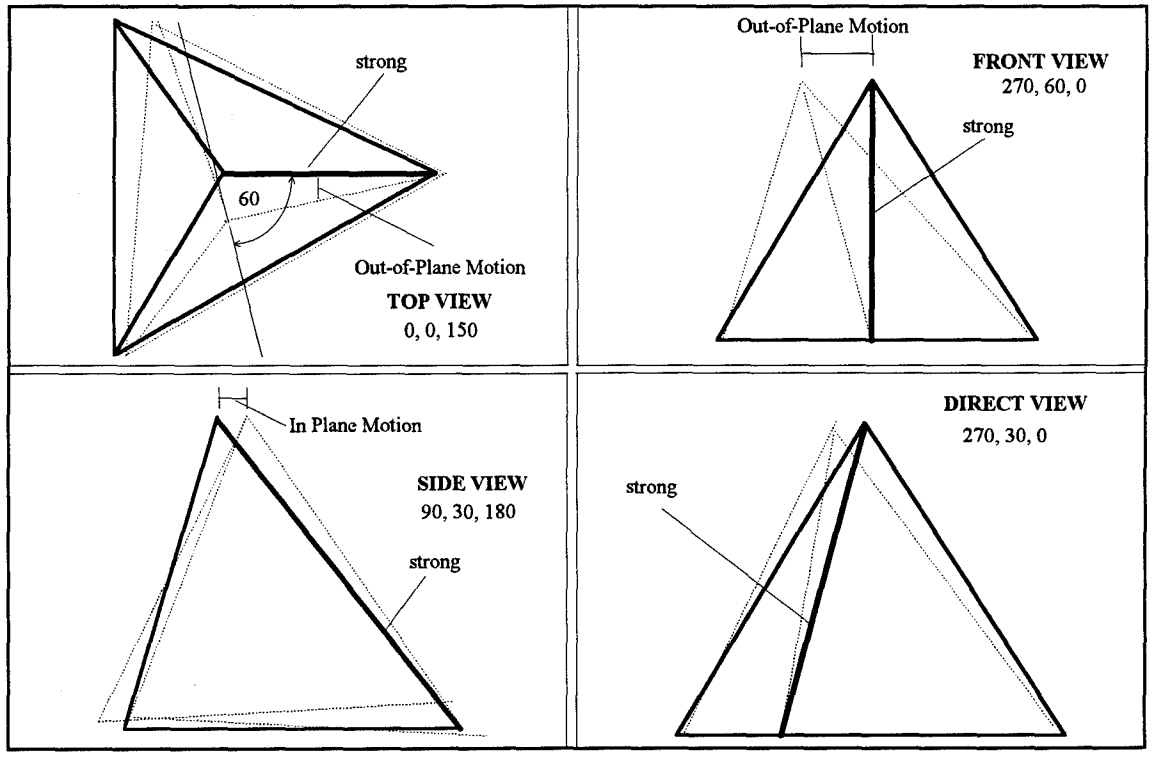


Figure 4-3: ACOSS4 Planar Motion

particular view and the dashed lines show the deformed structure just as they would appear in actual plot from IDEAS.

4.3.3 Definitions

The modal shapes next had to be defined such that they met the following criteria:

a) Be analogous to the standard beam modal shape deformations accepted in the structures community for one dimensional objects (e.g. beams, rods),

b) Provide the necessary fidelity to correctly distinguish the modes from one another,

c) Be robust enough to ensure no mode could fit more than one description (and therefore have more than definition).

Through trial and error, the following definitions were developed:

Axial Motion: Translational motion in the z-direction where an imaginary perpendicular line connecting the apex to the centroid of the triangular base would exhibit axial motion.

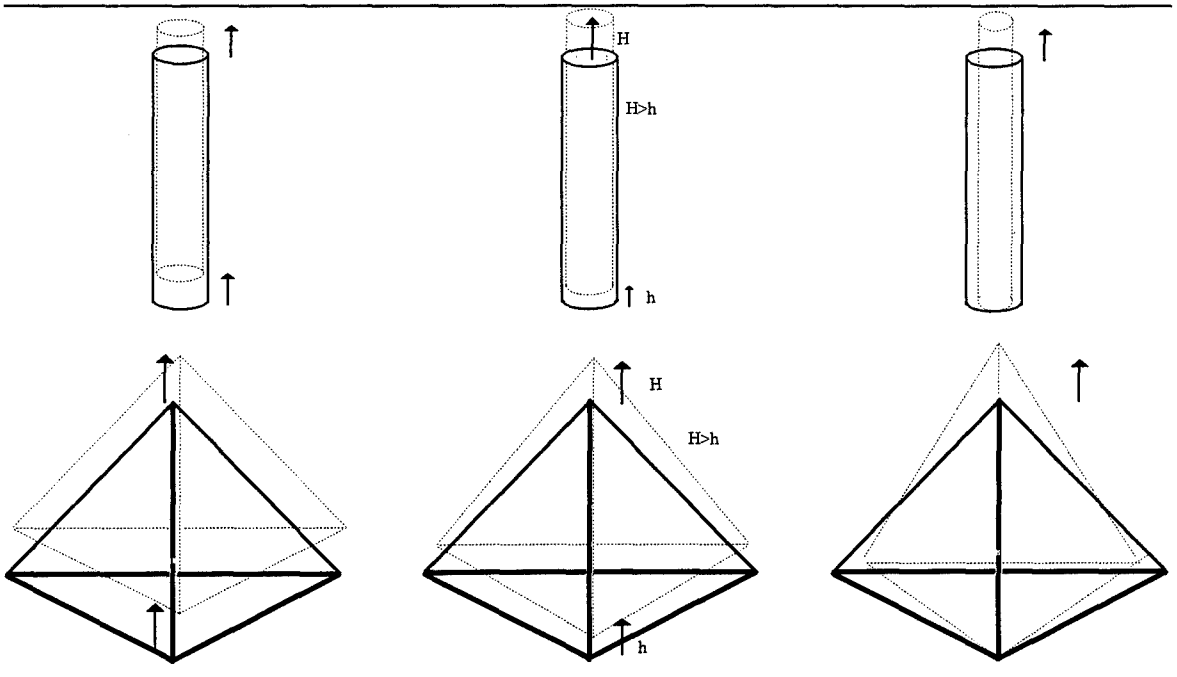


Figure 4-4: Axial Mode Equivalence

This includes motion where the entire truss moves as a rigid body in the z-direction, or where the apex moves slightly more or slightly less than the centroid (Figure 4-4). (Note: the deformed structures in the figures were drawn slightly more narrow or slightly offset to ensure the deformed and undeformed structures were distinguishable).

1st Bending: 1st Bending is defined as motion perpendicular to the z-direction where an imaginary line connecting the apex to the centroid of the triangular base would exhibit 1st bending motion (i.e. perpendicular line deviating slightly from one node). The three-dimensional nature of the structure requires the bending plane also be specified. See Figure 4-5 below.

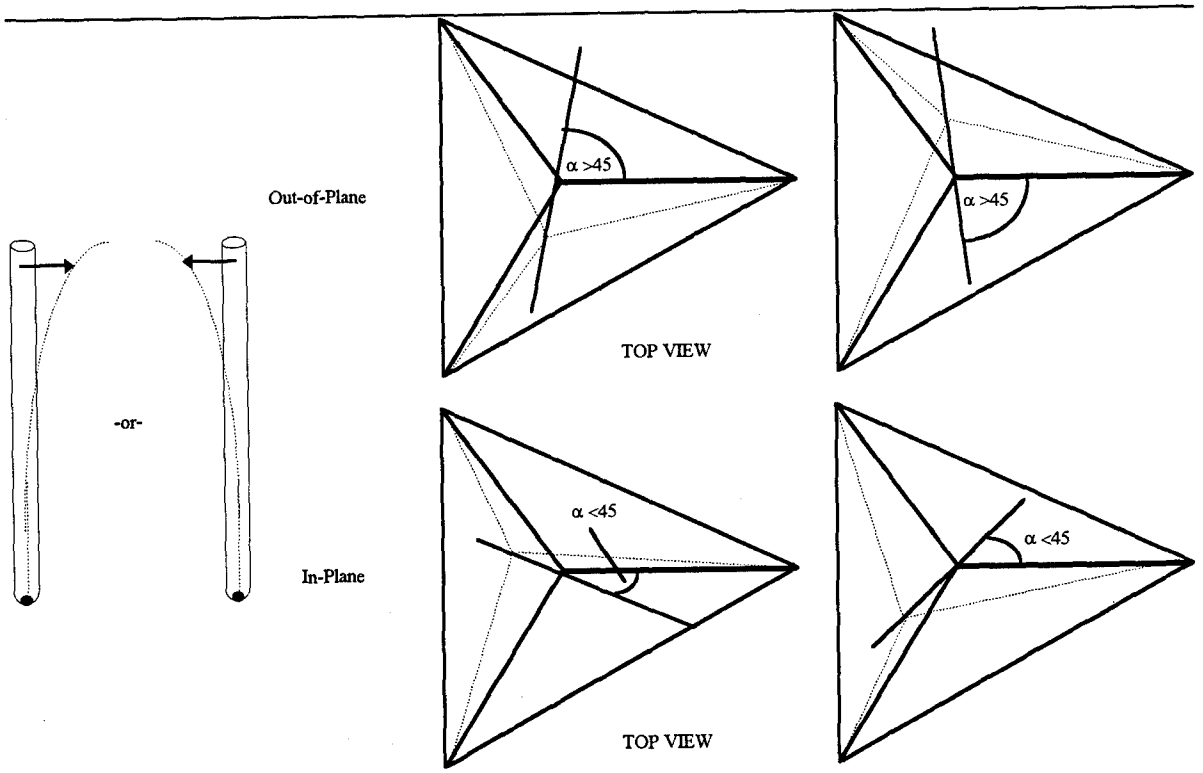


Figure 4-5: 1st Bending Motion Equivalence

2nd Bending: 2nd Bending is defined as motion perpendicular to the z-direction where an imaginary line connecting the apex to the centroid of the triangular base would exhibit 2nd bending motion (i.e. shallow sine wave connecting two nodes). Again, the dimensionality of the structure requires the bending plane be specified. See Figure 4-6 below.

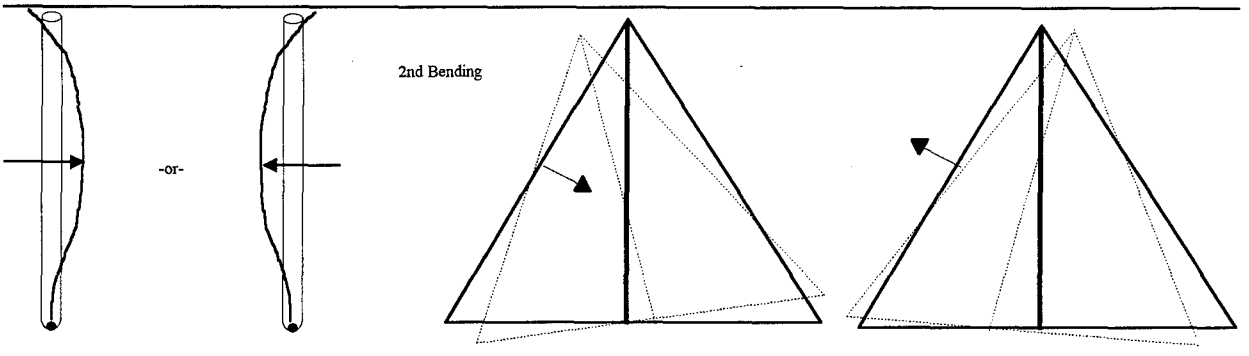


Figure 4-6: : 2nd Bending Mode Equivalence

1st Torsion: 1st Torsional is defined as motion in the z-direction where an imaginary line connecting the apex to the centroid of the triangular base would exhibit 1st Torsional motion (twisting motion), as shown below.

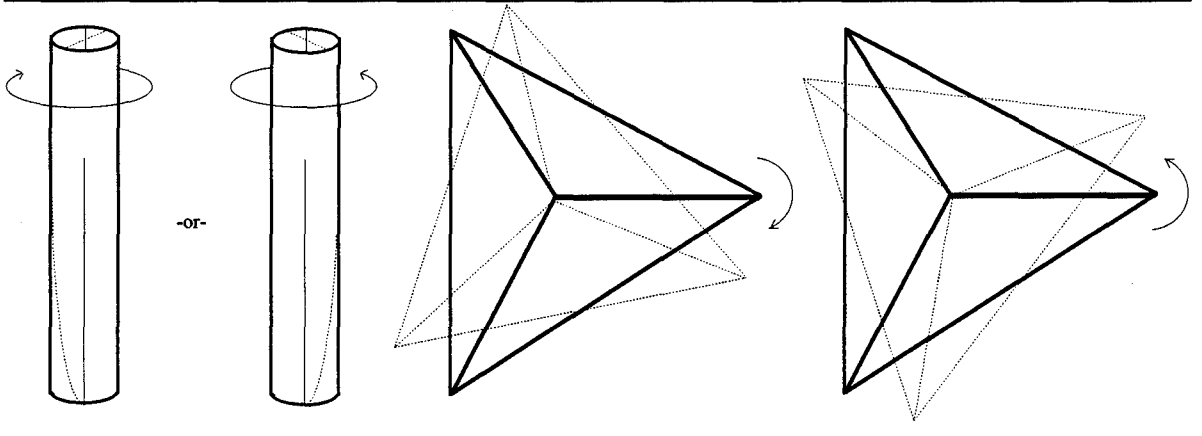


Figure 4-7: Torsional Mode Equivalence

Phase Motion: Structures may have two *distinct* coupled modes which are comprised of exactly the *same* two uncoupled modes. The coupled modes are completely distinct due the nature of the coupling of the equivalent modes. Figure 4-8 shows two distinct coupled modes in a 2-dimensional element comprised of exactly the same two singular modes (1st Axial and 1st Bending). These two new coupled modes were defined as 1st Rocking. *In-phase* rocking was defined to be the superposition of 1st axial motion in the +z direction with 1st bending when the tip deflects in the +x direction (to the reader's right). *Out-of-phase* was defined as the superposition of 1st axial motion in the -z direction with the same bending motion (tips deflection to the right). When the complete motion of the apex is traced out, the rocking motion which gives this mode its name, is clearly apparent in the last column. Rocking motion in the subject structure is apparent in Figure 4-10.c and 10.e.

ACOSS4, even as a symmetric object, exhibited phase motion as well. First bending and 1st axial motion appeared coupled in two separate cases and in an analogous fashion as shown in Figure 4-8. Unfortunately, defining the phase for the three-dimensional structure proved much more difficult than for the one-dimensional beam. Alternately using either the apex or Node 4 (the lower left point on the structure in the viewports in Figure 4-9), as the single reference point led to ambiguity in later mode characterization. Instead, both reference points, the apex *and* Node 4, were needed to completely define the phase with no ambiguity. The view port for phase determination was also designated as the side view (Frame 3). For ACOSS4, the definitions of Phase are shown in Figure 4-9. The only modes to exhibit phase behavior were "rocking" modes, as described below:

1st Rocking (In-Plane) : 1st Axial *in phase* with 1st In-Plane Bending
2nd Rocking (In-Plane) : 1st Rocking *out of phase* with 1st In-Plane Bending
1st Rocking (Out-of Plane): 1st Axial In-Phase with 1st Out-of-Plane Bending

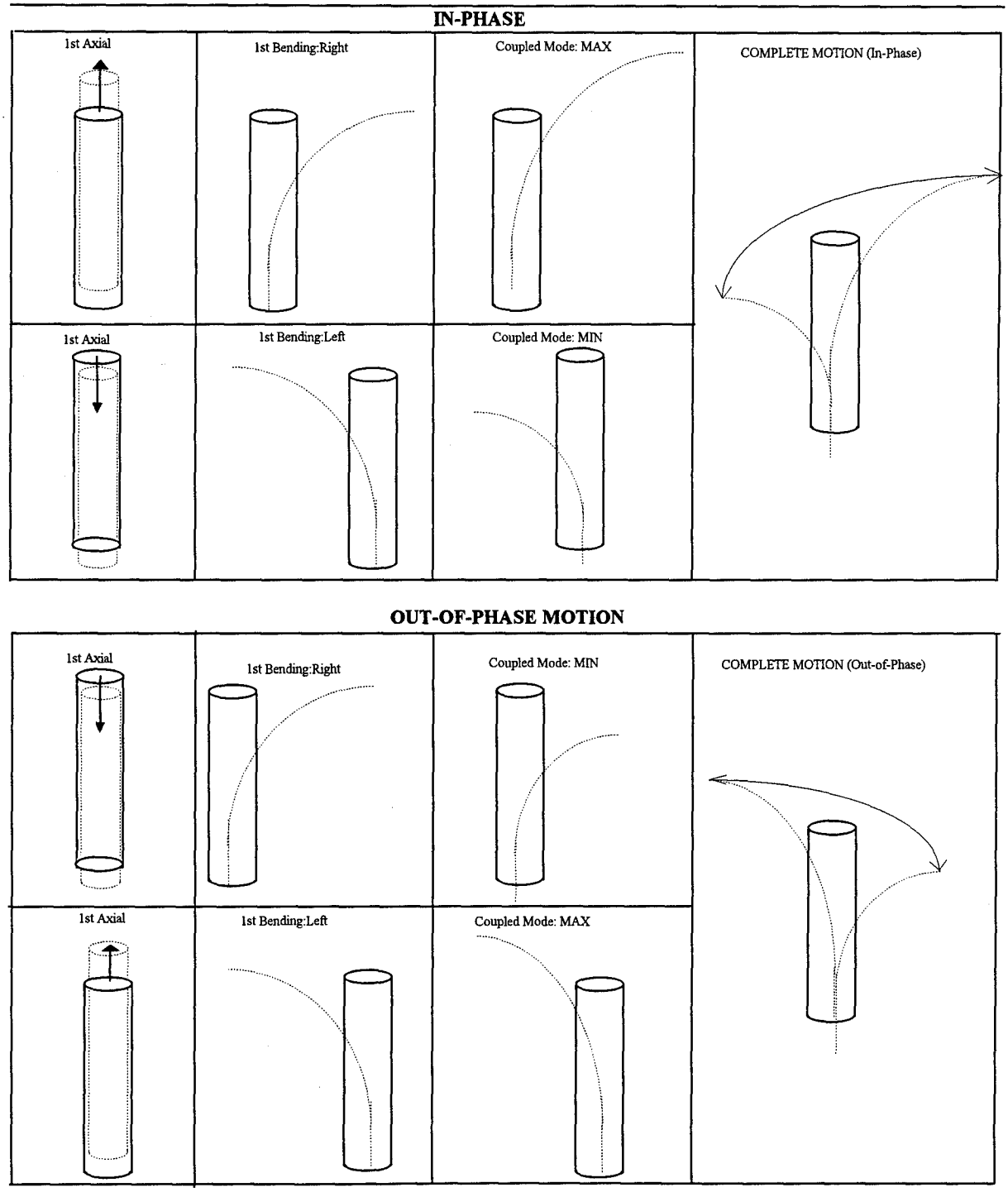


Figure 4-8: One-Dimensional Analogy to Rocking Motion

The nodes at Node 4 and the apex have been emphasized in the following figure for clarity.

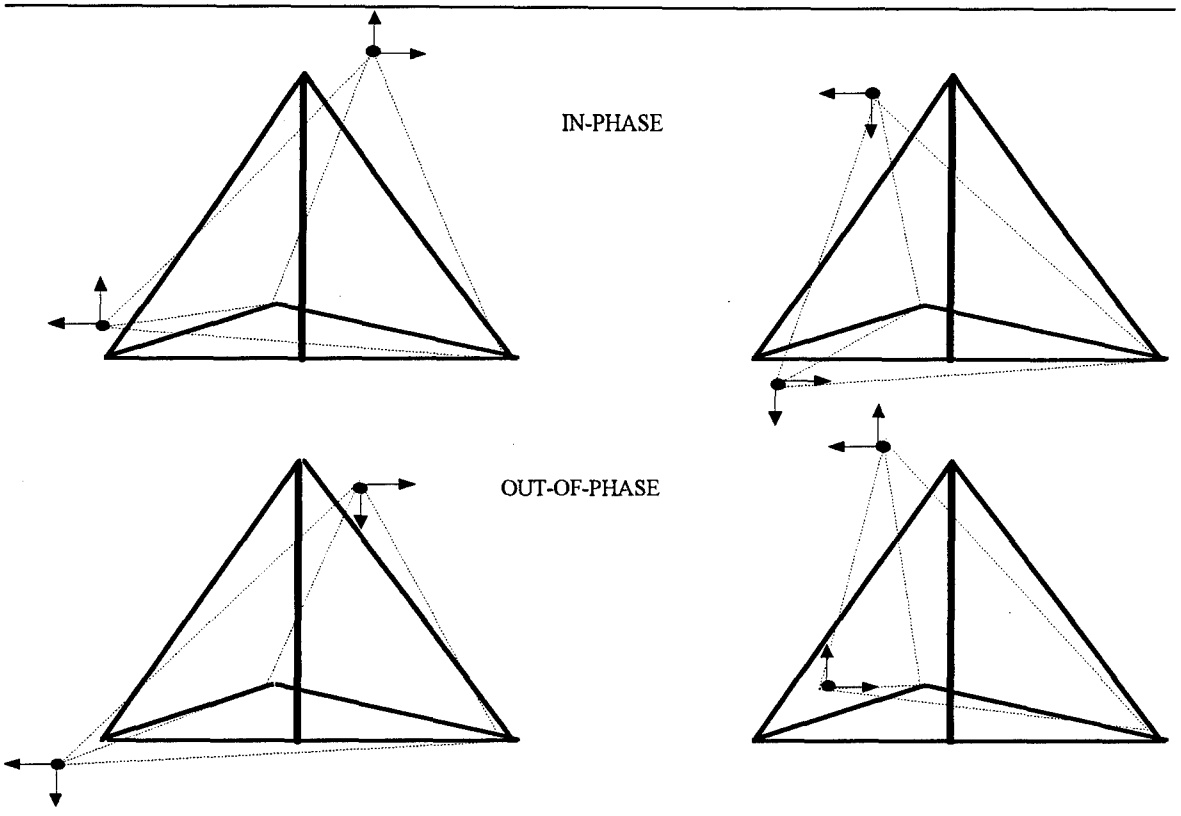


Figure 4-9: Phase Characteristics of Rocking Motion

Because the ACOSS4 model employs only six actuators (one actuator in each of the six bipod legs), which control only the lower six modes, modal crossings in only the first seven modes were investigated. Modal swaps contained completely within the six higher modes (Modes 7-12), would not affect the optimizer since these modes were not controlled. Modal swaps between the lower six and the higher six modes were considered unlikely, (with the possible exception of a mode swap between a controlled mode (Mode 6) and an uncontrolled mode (Mode 7), and were therefore not investigated.

The complete finite element model of the ACOSS4 truss is included in Appendix A.1. Determining the number of natural frequencies (and therefore the number of nodes) for a truss is found using the following equation from [Meirovitch, 1970:47]:

$$\boldsymbol{n} = 3N - \boldsymbol{c} \qquad (4-1)$$

where
$\boldsymbol{n}$ = total # of resulting DOFs
N = # DOFs of each node
$\boldsymbol{c}$ = total # of constraints

ACOSS4 has 10 nodes. Six of these nodes are clamped and thus have three constraints each. Substituting these values into Eq. 4-1 produces

$$\boldsymbol{n} = 3\frac{DoF}{node} * 10 nodes - [6 nodes * 3\frac{constraints}{node}] = 12\boldsymbol{DOF}$$

Thus, ACOSS4 has 12 degrees of freedom and therefore 12 natural frequencies and 12 natural modes. All 12 eigenvalues and eigenvectors are included in Table 4-1 for completeness, although only the first 7 modes were investigated. Plots of the modal deformations are shown in Figures 4-10.a-g. and the remaining modes included in Appendix B. The methodology for mode identification was followed as described above for the first seven iterations and the mode identities included in Table 4-2. The mode characterization process used in identifying the first seven modes of both the nominal design and of the first iteration are included in detail for completeness. In all cases of mode identification, the mode was characterized by the **majority** of the motion present in the modal deformation.

4.4 ACOSS4 Mode Characterization

4.4.1 Nominal Model

Several nuances of the display were also transferred to the hardcopy plots and must first be explained. The first two result from having a video driver that was not 100% compatible with the software. First, the lumped masses grew by several orders of magnitude for plotting and would have appeared as large (3/8") squares. Thus, the lumped masses have been hidden to prevent obscuring the primary motion. Secondly, many of the deformed lines of the eigenvectors are plotted as *curved*. These lines would not actually be curved, since pinned truss members only permit axial deformation. Finally, note that the displacements have been purposely magnified to span 20% of the screen in order to highlight the deformed motion.

Mode 1 (1st Out-Of-Plane Bending): The top view during animation shows pure Out-of-Plane motion. There is little deviation at the nodes where the bipods are connected (Nodes 2-4). As mentioned previously, there is no motion apparent in the side view indicating no in-plane motion. (See Fig. 4-10a).

Mode 2 (1st In-Plane Bending): Again, the top view during animation and in the static view included in the first panel of Figure 4-10.b show pure in-plane motion. The structure deforms perfectly parallel to the strong upright which serves as the reference. The front view indicates no out-of-plane motion, while the side view indicates pure In-Plane bending. (See Fig. 4-10b).

Mode 3 (1st In-Plane Rocking): This is the first coupled mode. The front view shows some significant axial deformation completely perpendicular to the base of the structure, and 2nd Bending is apparent in the third panel. The in plane motion is clearly shown in the top and side views. Note how even in the front view one can see the base of the structure *swinging* as if the tetrahedron was connected to a hinge at the apex. Note how as the base swings *out*, Element 5 (E5) remains perfectly parallel to the original structure and nodes 3 and 4 have symmetric displacements. The phase motion which distinguished this rocking motion from that in Mode 5 is not very apparent in the nominal case, again due to symmetry. However, one notices that in the side view, the apex is moving up and to the right while Node 4 is moving up and to the left, i.e. in phase. (See Fig. 4-10c).

Mode 4 (2nd Out-of-Plane Bending): Viewing Modes 3 and 4 in sequence one might be tempted to identify this mode as a rocking motion as well, where the rocking occurs 120° from the rocking plane in Mode 3. However, what prevents this mode from being labeled as a rocking motion is the total lack of 1st axial deformation missing from the front view. The compression in member E3 characterizes this as 2nd Bending. (See Fig. 4-10d).

Mode 5 (2nd In-Plane Rocking): This is the second coupled mode. Axial motion is apparent in the front view and 2nd Bending is noted from the top view, especially apparent due to the compression in the two weak uprights, Elements 2 and 3. From the side view (Panel 3), it is clear that the apex is moving up and to the left while Node 4 is moving up and to the right characterizing the axial motion as out of phase with the bending motion. The effect of symmetry in the bipod legs is still apparent causing the deformed structure to move parallel to the base of the tetrahedron during the rocking motion. (See Fig. 4-10e).

Mode 6 (Torsion): The effect of symmetry is most clearly demonstrated in Mode 6. The structure appears to be rotating perfectly (i.e. no precession of the apex) about an axis through the apex and which is perpendicular to the tetrahedral base. Nodes 2, 3 and 4 connecting the bipod legs to the tetrahedron describe perfect lines with no deviation apparent during the motion. Neither planar motion nor phase motion can be detected in the front and side views. (See Fig. 4-10f).

Mode 7 (1st Out-of-Plane Shearing): This motion initially appeared to be another rocking motion but the absence of a clear axial component made that definition imprecise. By animating the displacements and rotating through several angles, it was apparent that the structure was being sheared -- i.e. it appeared that opposing forces were acting tangential to the out-of-plane surface, alternately increasing and decreasing the angles at the vertices. (See Fig. 4-10g).

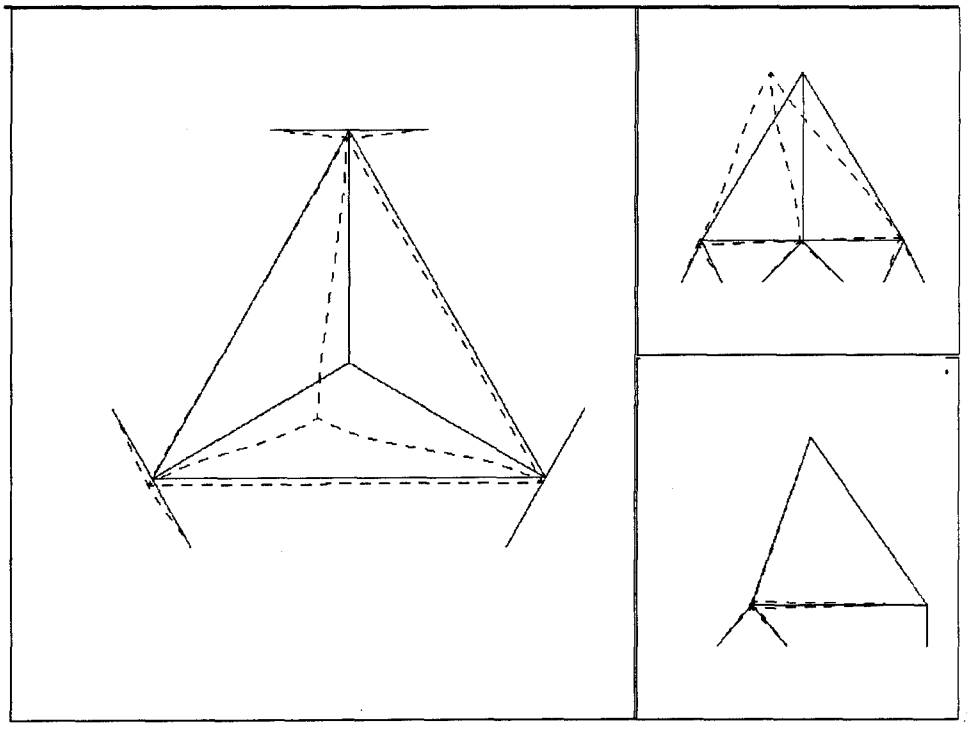


Figure 4-10a: Nominal/Analytic Mode 1 (1st Out-of-Plane Bending)

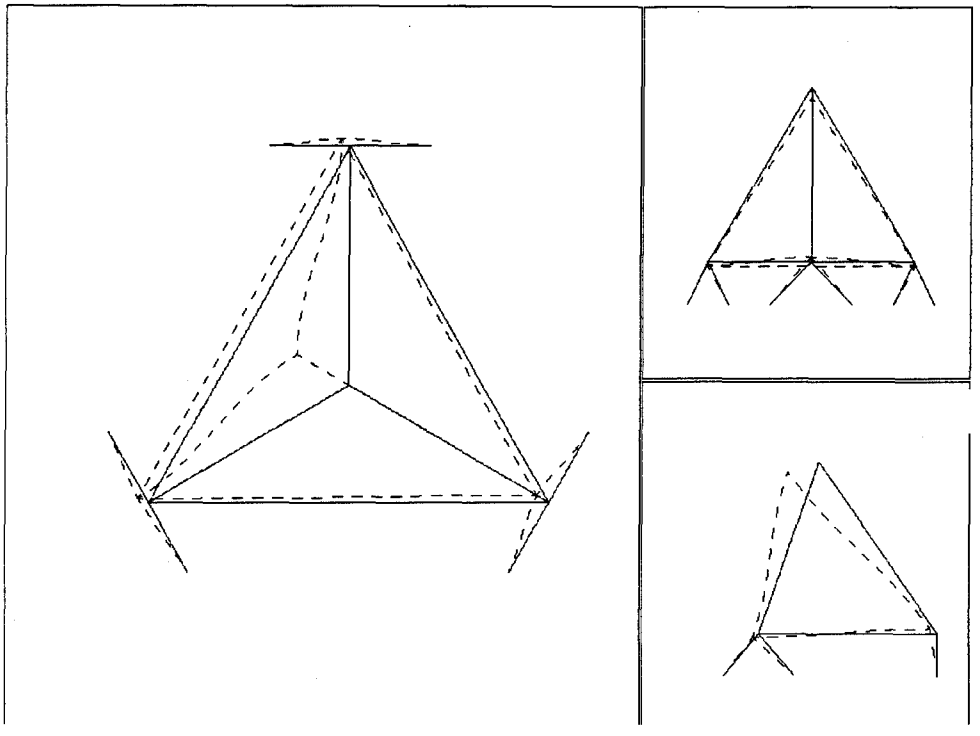


Figure 4-10b: Nominal/Analytic Mode 2 (1st In-Plane Bending)

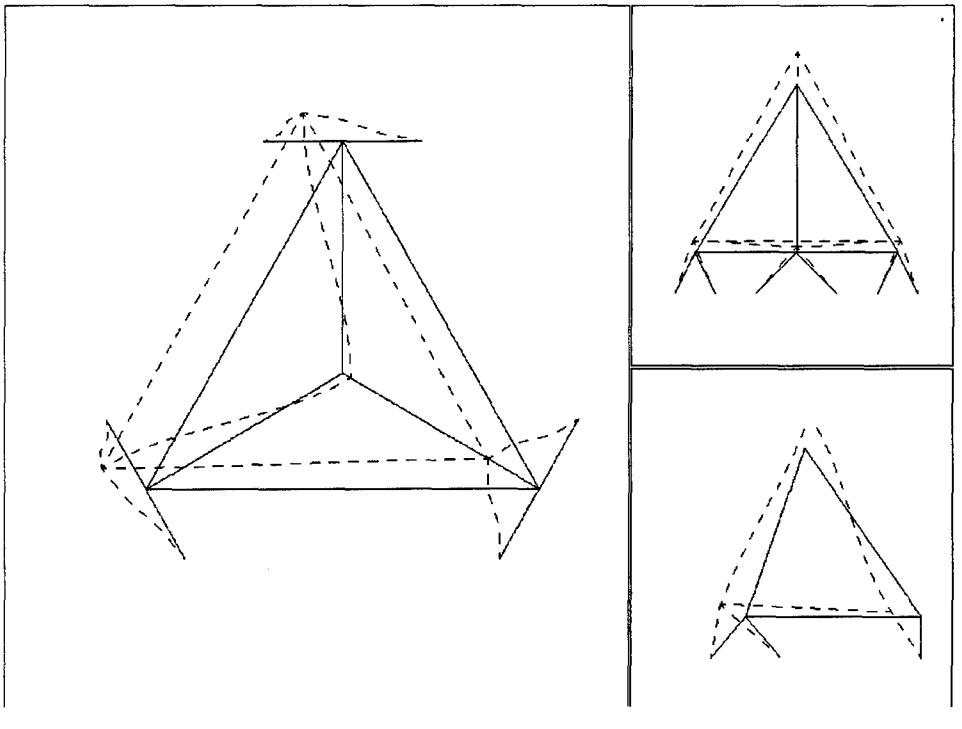


Figure 4-10c: Nominal/Analytic Mode 3 (1st In-Plane Rocking)

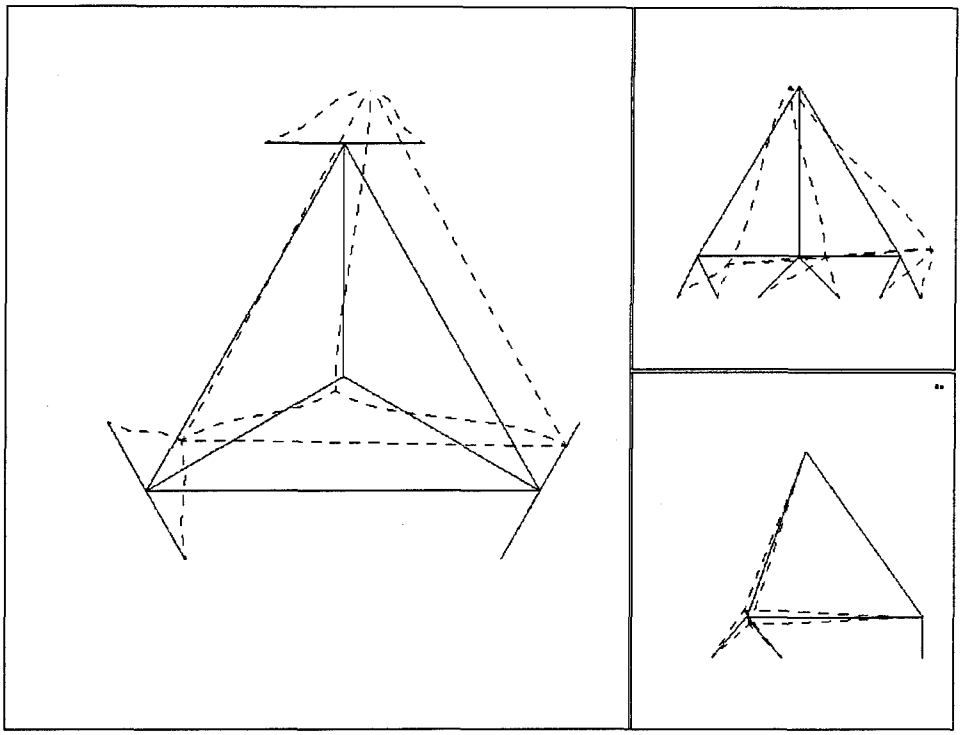


Figure 4-10d: Nominal/Analytic Mode 4 (2nd Out-of-Plane Bending)

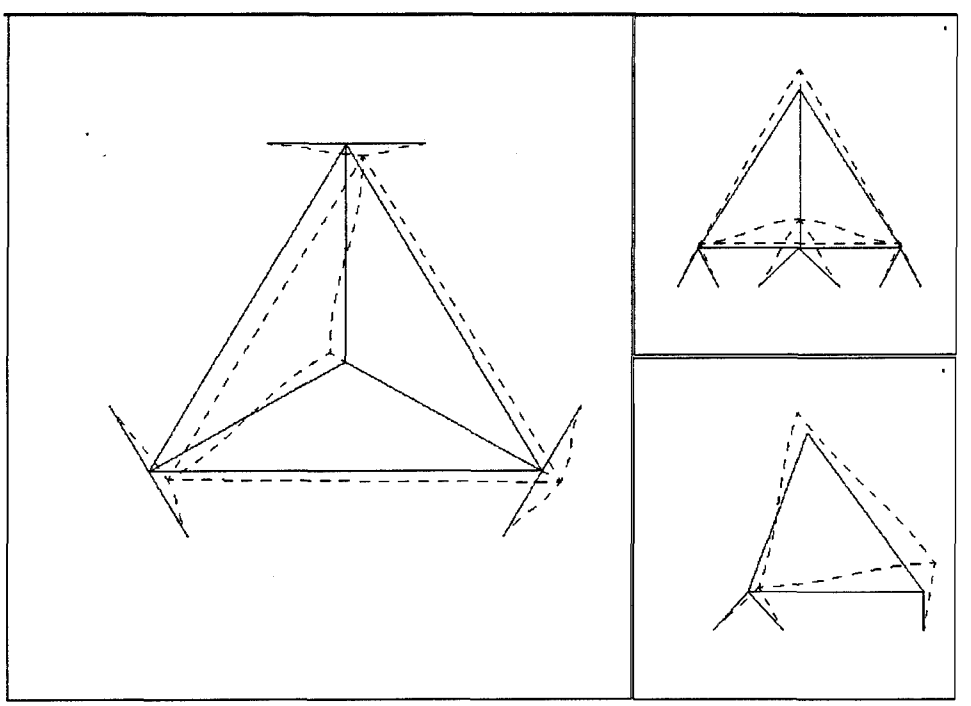


Figure 4-10e: Nominal/Analytic Mode 5 (1st In-Plane Rocking)

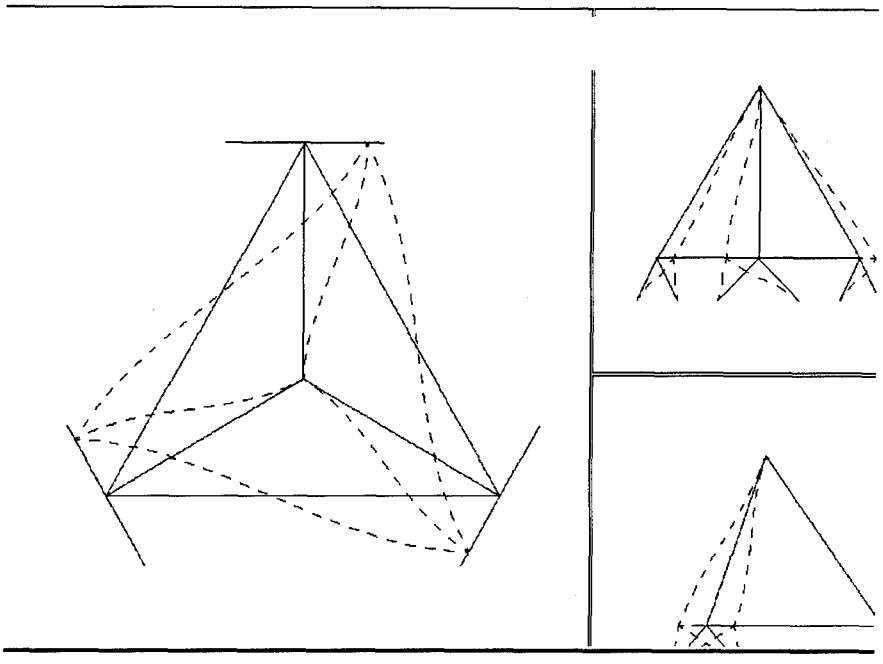


Figure 4-10f: Nominal Analytic Mode 6 (1st Torsion)

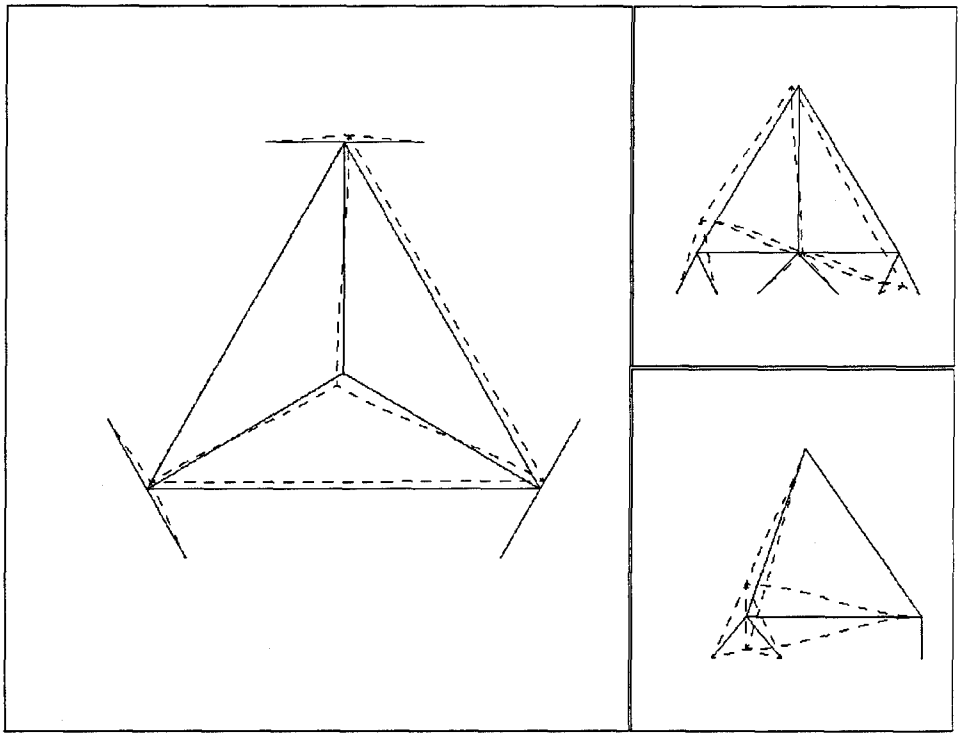


Figure 4-10g: Nominal Analytic Mode 7 (1st Out-of-Plane Shearing)

Characterization of the nominal modes lent confidence to the mode definitions. The seven modes in the next six iterations were then characterized. This required that six additional FEMs be built, using the cross-sectional areas output from the (failed) optimization solution based on analytic derivatives [Canfield, 1992]. Characterization of all of the lower seven modes in several of the initial iterations was necessary for visual identification of a mode swap. Results of the analysis for the first four iterations are included in Table 4-2. Mode characterizations for Iteration 1 are included in detail to emphasize the significant modal coupling which occurs in just the first iteration. The results of the subsequent mode characterizations are summarized in Tables 4-2 and 4-3.

Whereas the natural frequencies for the nominal ACOSS4 structure are well documented in the literature (and therefore verifiable), the accuracy of the finite element models built for subsequent iterations was verified by comparing the models' lower two constrained frequencies to those from the original optimization history. The frequency constraints were formed as

$$g = 1 - \frac{\lambda}{\lambda_{allow}} \qquad (4-2)$$

where $\lambda = \left(\omega \frac{rad}{sec}\right)^2$. Algebraic manipulation will show this becomes

$$\omega_{hz} = \frac{\sqrt{\lambda_{allow}(1-g)}}{2\pi} \qquad (4-3)$$

which was the form used in the verification process. The first two frequencies for each iteration were validated in this manner. Characterization of the first seven modes of the first iteration follow.

Table 4-2: ACOSS4 Nominal Frequency Distribution

Mode#/ It #	0	1	2	3	4	5	6
	Frequency Distribution (in hz)						
1	0.2135874	0.1813903	0.1808355	0.1822452	0.1849712	0.1877405	0.1894244
2	0.2649485	0.1939295	0.1953476	0.2011477	0.2068810	0.2123655	0.2165920
3	0.4600698	0.3067506	0.2945747	0.2911318	0.2888258	0.2887076	0.2920095
4	0.4706857	0.3428443	0.3683392	0.3896224	0.4029963	0.4034913	0.3951084
5	0.5408387	0.4616504	0.4518428	0.4555888	0.4578784	0.4645500	0.4494889
6	0.6691620	0.4947067	0.4912449	0.5143530	0.5340067	0.5550178	0.5770200
7	0.7419893	0.5812024	0.6305590	0.6773335	0.7055712	0.7007192	0.6942277

4.4.2 ACOSS4 Iteration 1

Mode 1: (1st Out-of-Plane Bending): This mode is similar to the nominal design but some differences are immediately apparent. Whereas in the nominal design the deformed base remained parallel to the actual base, here the deformed base is no longer parallel. There is now a non-negligible in-plane bending component as well, apparent in the side view, although it is not nearly as significant as the out-of-plane component. (See Fig. 4-11a).

Mode 2 (1st In-Plane Bending): Again, the mode is similar to Mode 2 of the nominal design but with some significant differences. Whereas in the nominal design, the bending motion apparent in the strong upright (element E1) remained perfectly parallel to undeformed model, the deformed upright in this iteration does not. This motion contributes to the small out-of-plane component noticeable in the front view. Also, there is a *shifting* motion, a

translational motion in the x-direction of the tetrahedron, apparent in the side view. (See Fig. 4-11b).

Mode 3 (1st In-Plane Rocking - In-Phase): Axial motion is apparent in the front view but of diminished contribution in comparison to the nominal case. While remaining essentially parallel to the undeformed structure, the base in the deformed model (during the rocking motion) now has some motion out-of-plane noticeable in the top view. Axial and bending motions are in phase with each other. (See Fig. 4-11c).

Mode 4 (2nd Out-of-Plane Bending): Coupled motion is very apparent. The majority of this motion is indeed 2nd Bending as evidenced by the compression in Element E3. Still, no axial motion is evident preventing characterization as a rocking motion as in Modes 3 and 5. However, there is a significant torsion component clearly demonstrated in the top view. (See Fig. 4-11d).

Mode 5 (1st In-Plane Rocking - Out-of-Phase): Little axial motion is apparent but there is some contribution. Some torsion is evident but not nearly at the level in the previous mode. The 2nd Bending motion is apparent in element E2. The phase is difficult to distinguish here, but using a strict interpretation of the definition, axial motion was determined to be out-of-phase with the bending component. (See Fig. 4-11e).

Mode 6 (1st Torsion): This motion is very similar to the nominal design, however some rocking motion is evident in the front view. Note the corruption of the pure torsion evident in the nominal design. The imaginary axis around which the tetrahedron was rotating in the nominal design is now precessing during the motion. Also, Nodes 2,3, and 4 no longer describe perfect lines during the rotation but deviate unsymmetrically. (See Fig. 4-11f).

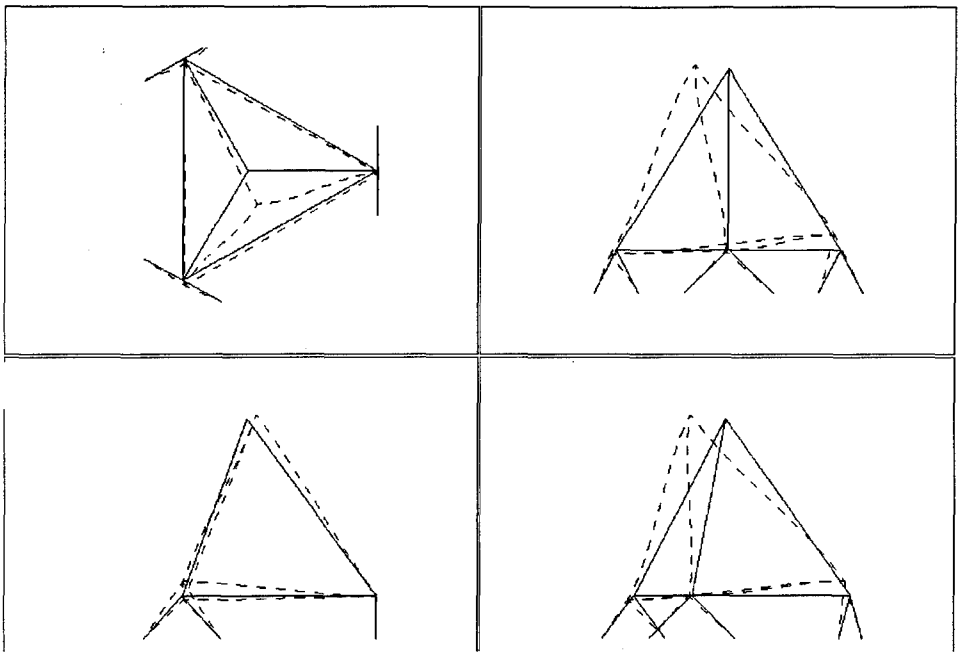


Figure 4-11a: Iteration 1/Analytic Mode 1(1st Out-of-Plane Bending)

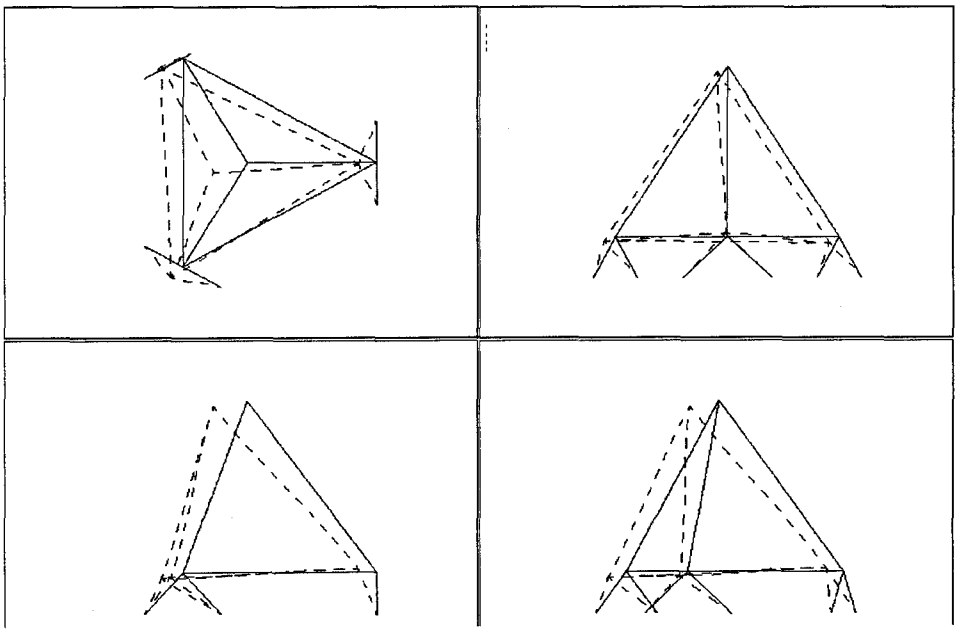


Figure 4-11b: Iteration 2/Analytic (1st In-Plane Bending)

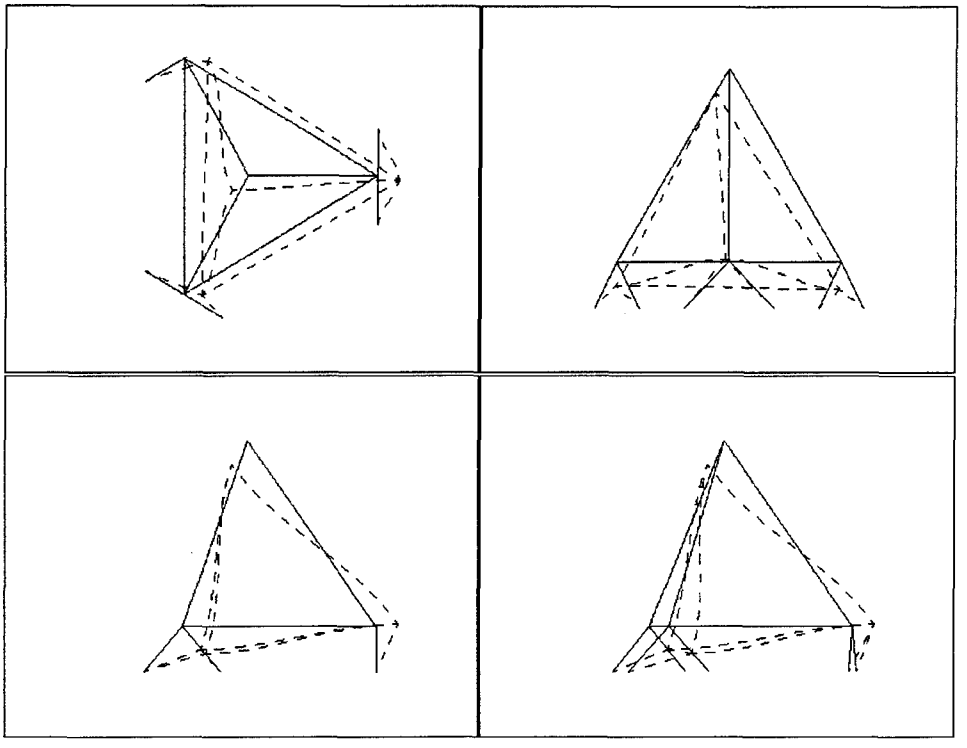


Figure 4-11c: Iteration 1/Analytic Mode 3 (1st In-Plane Rocking [In-Phase])

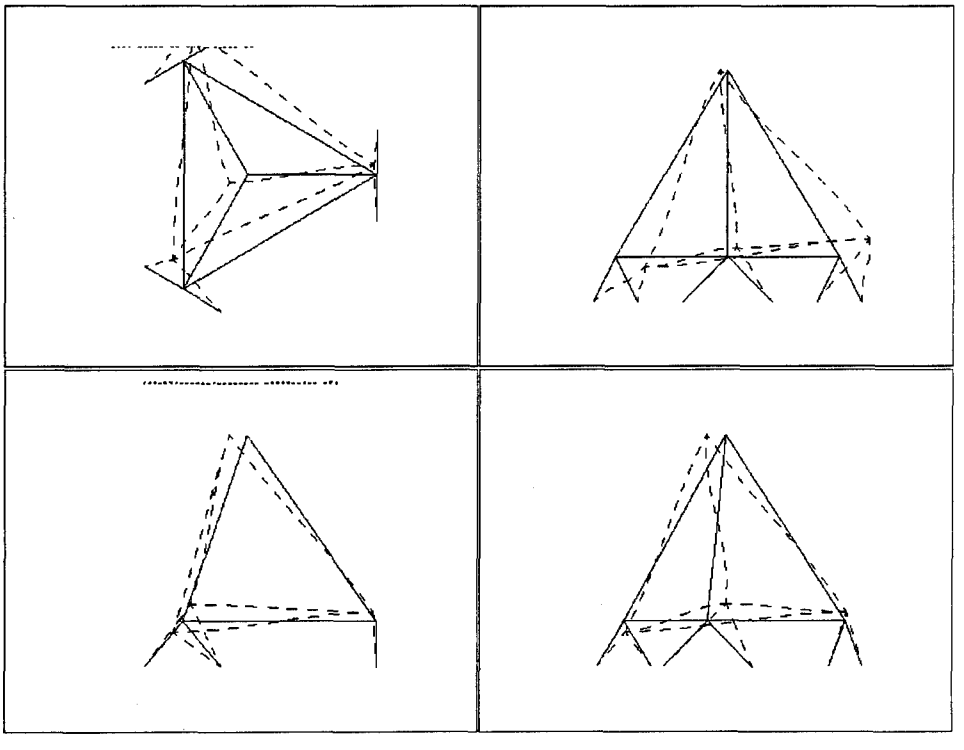


Figure 4-11d: Iteration 1/Analytic Mode 4 (2nd Out-of-Plane Bending)

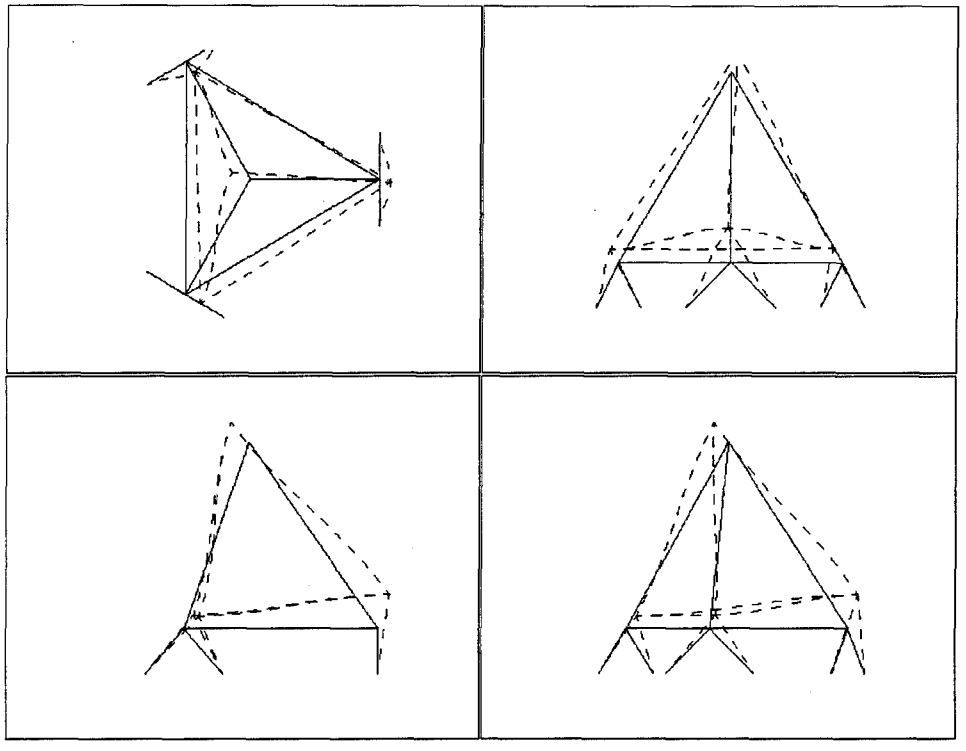


Figure 4-11e: Iteration 1/Analytic Mode 5 (1st In-Plane Rocking [Out-of-Phase])

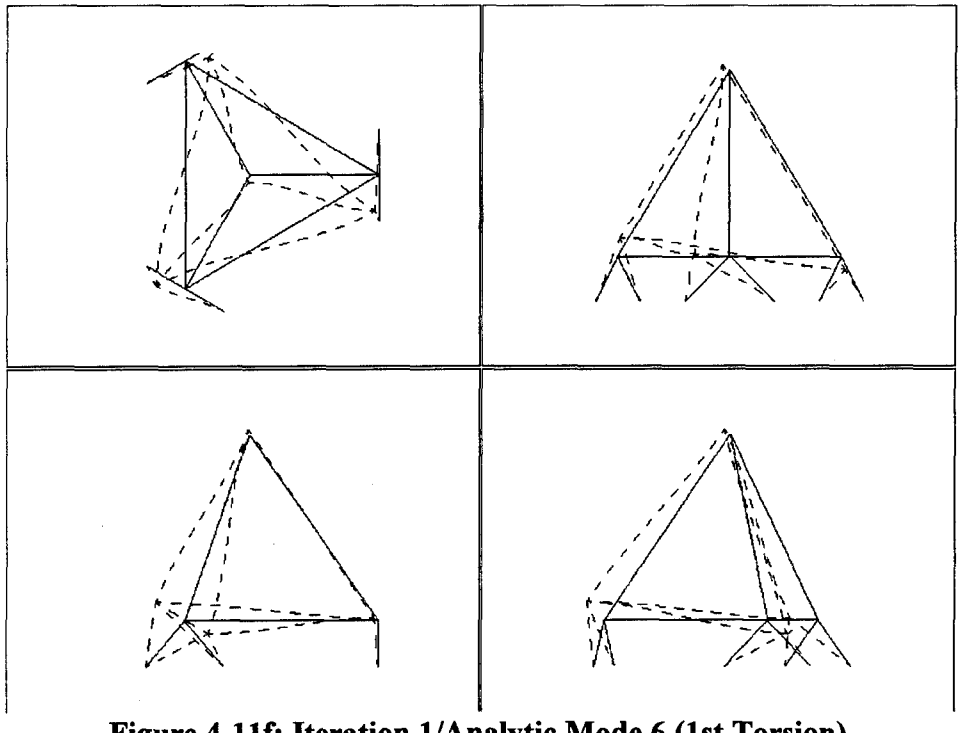


Figure 4-11f: Iteration 1/Analytic Mode 6 (1st Torsion)

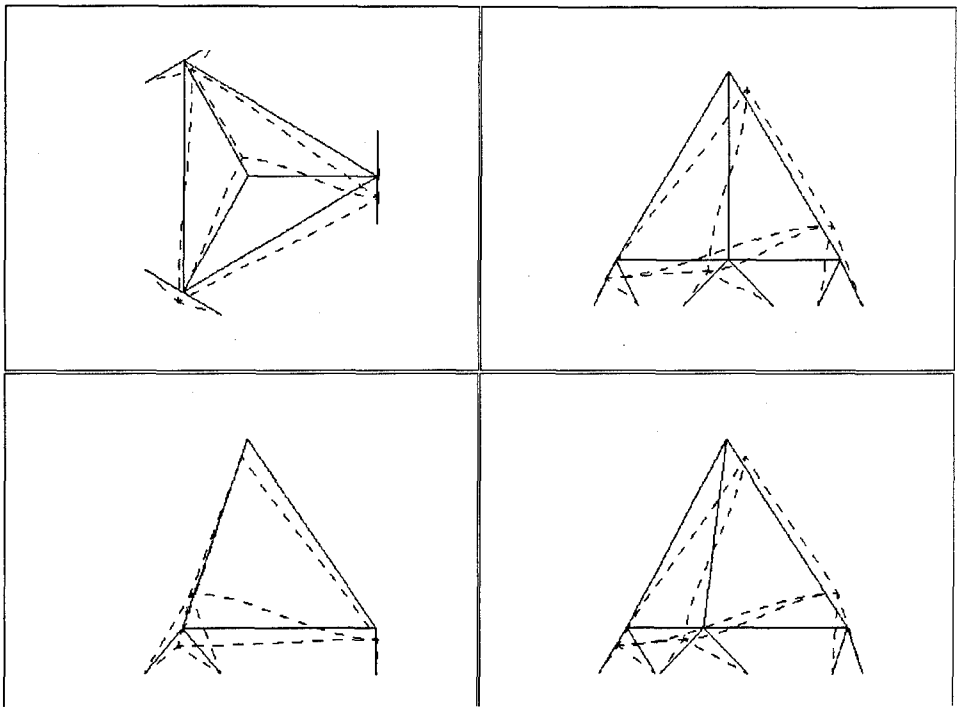


Figure 4-11g: Iteration 1/Analytic Mode 7 (1st Shearing)

Mode 7 (1st Out-of-Plane Shearing): This motion is very similar to the motion in the nominal design but a small axial component is present. The swinging motion apparent in the nominal design is greatly diminished and the shearing motion more defined. (See Fig. 4-11g).

The next five iterations were characterized in a similar manner. As the coupling between modes became more significant, the need to apply the characterization algorithm to the majority of the deformed motion became more clear. Extreme care was taken, however, to ensure in characterizing these modes, an entirely new mode was not negligently dismissed by defining it in terms of a previous mode in the interest of conformity; the emergence of an entirely new mode could have been the source of the optimizer's failure to converge.

To determine if there was a mode swap between iterations, the frequencies for each of the seven modes for the first seven iterations, where the nominal case (Iteration 0 in the

FRAME analysis) is represented as Iteration **1** in the ensuing plots. This plot is included in Figure 4-12. The 'column' of data points above each iteration number correspond to the seven different frequencies for each mode at each iteration. Each plot was analyzed and the mode characterization algorithm rigorously employed. Identifying each mode by its ordinal frequency allowed the points to be connected as shown. Results are shown in Table 4-3. From Figure 4-12, one can see that no mode swaps between iterations were found; each mode retained a 1-1 correspondence with its ordinal frequency throughout the iteration history.

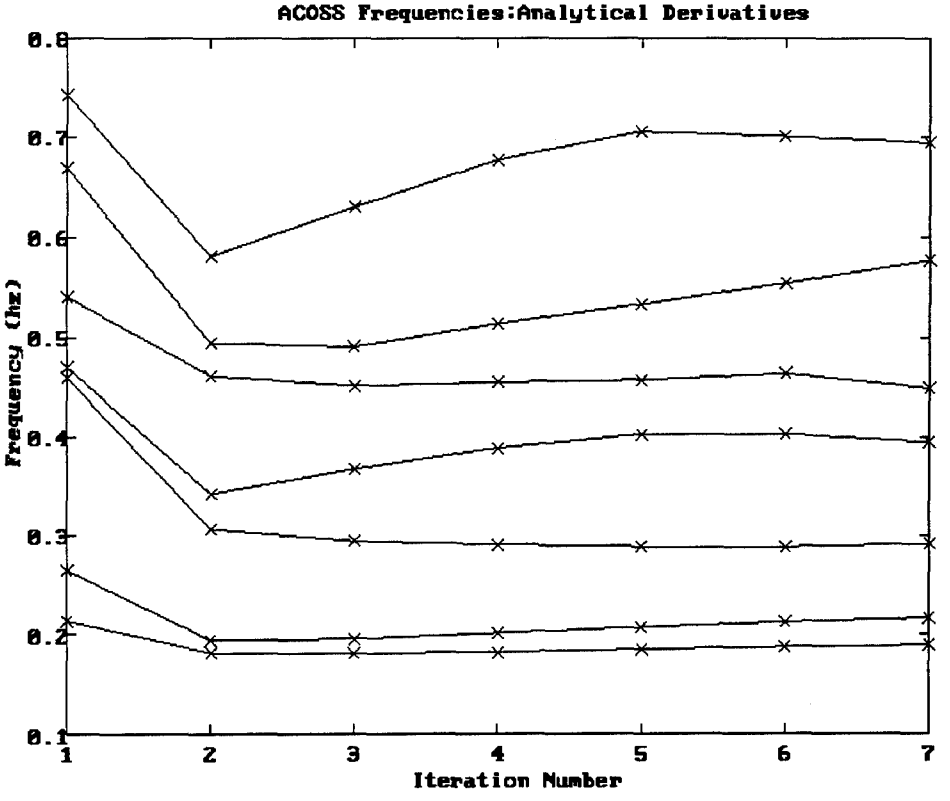


Figure 4-12: ACOSS4 Natural Frequency History

Table 4-3: ACOSS4 Mode Characterization using Analytic Derivatives

MODE	Nominal			IT 1			IT 2			IT 3			IT 4			IT 5			IT 6		
Char.	M	P	ϕ	M	P	ϕ	M	P	ϕ	M	P	ϕ	M	P	ϕ	M	P	ϕ	M	P	ϕ
1	1B	O		1B	O		1B	O		1B	O		1B	O		1B	O		1B	O	
2	1B	I		1B	I		1B	I		1B	I		1B	I		1B	I		1B	I	
3	1R	I	I	1R	I	I	1R	I	I	1R	I	I	1R	I	I	1R	I	I	1R	I	I
4	2B	O		2B	O		2B	O		2B	O		2B	O		2B	O		2B	O	
5	2R	I	O	2R	I	O	2R	I	O	2R	I	O	2R	I	O	2R	I	O	2R	I	O
6	1T			1T			1T			1T			1T			1T			1T		
7	1S	O	I	1S	O	I	1S	O	I	1S	O	I	1S	O	I	1S	O	I	1S	O	I

4.5 Mode Characterization for Finite Difference Solution

Although no mode swaps were detected, it was decided not to immediately revert to one of the first two hypothesis, feeling it would be prudent to first determine visually where the modes deviated between the unsuccessful solution based on analytic derivatives and the successful solution based on the finite difference derivatives. Six new FEMs were built using the cross sectional areas from the original data runs when ***FRAME*** had been equipped with finite difference derivatives. The same mode characterization algorithm was then applied to the finite element solutions. The nominal solution was the same for the solution based on analytic derivatives as was Iteration 1. Close scrutiny of Modes 3 and 4 of Iteration 2, however, were not as expected, as detailed below. The frequency iteration history is shown in Table 4-4. Plots have been excluded (except as noted below) in the interest of brevity.

Table 4-4: ACOSS4 FD Frequency History

Mode#/ It #	0	1	2	3	4	5	6
	Frequency Distribution (in hz)						
1	0.2135874	0.2053558	0.1656328	0.1705601	0.1748678	0.1864965	0.1919765
2	0.2649485	0.2451654	0.2063847	0.2103761	0.2141715	0.2251522	0.2293247
3	0.4600698	0.3836274	0.2956890	0.3014011	0.3066098	0.3245196	0.3343967
4	0.4706857	0.4175960	0.3448448	0.3559204	0.3682598	0.3905437	0.3957293
5	0.5408387	0.4907984	0.3736763	0.3892873	0.4864272	0.4482020	0.4687221
6	0.6691620	0.5617776	0.4942248	0.5028366	0.5170048	0.5493836	0.5608690
7	0.7419893	0.6325130	0.6554744	0.6595717	0.6692428	0.6906056	0.6939048

4.5.1 Iteration 1

Mode 3 (1st In-Plane Rocking): Axial motion is perceptible in the front view. 2nd In-Plane bending is seen in the top view with the compression of the weak uprights, members E2 and E3. During deformation, the rocking motion describes an arc almost parallel to the strong upright -- very little out-of-plane motion. The side view shows the apex moving down and to the readers left while node 4 is moving down and to the right indicating the bending and axial motions are in phase. This characterization matches that of the nominal design. (See Figure 4-13a.)

Mode 4 (2nd Out-of-Plane Bending): There is significant out-of-plane motion with correspondingly little in-plane motion as seen in mode 3. No axial motion is at all apparent. Slight torsion is detectable. This motion too was characterized as the same motion as in the nominal design. (See Figure 4-13b.)

4.5.2 Iteration 2

Mode 3 (*2nd Out-of-Plane Bending*): There is negligible axial motion in the front, side and direct views. The front view shows negligible in-plane motion. Note the large out-of-plane motion apparent in the top view, which also shows compression of member E2 indicative of 2nd Bending. As the axial motion is no longer present, no phase characterization need be applied. (See Fig. 4-14a.)

Mode 4 (*1st In-Plane Rocking*): There is noticeable axial motion present in the front and direct views. 2nd Bending is noticeable in compression of members E2 and E3. Although strict interpretation of definitions identifies this as in-plane motion, one can see similar magni-

tudes of in-plane and out-of plane motion in the front and side views, respectively. This motion does not strictly occur in-plane. The apex is deforming down and to the left while node 4 is deforming downward and to the right indicating axial motion is *in* phase with the significant 2nd bending component. Rocking motion is very apparent in animation. (See Fig. 4-14b.)

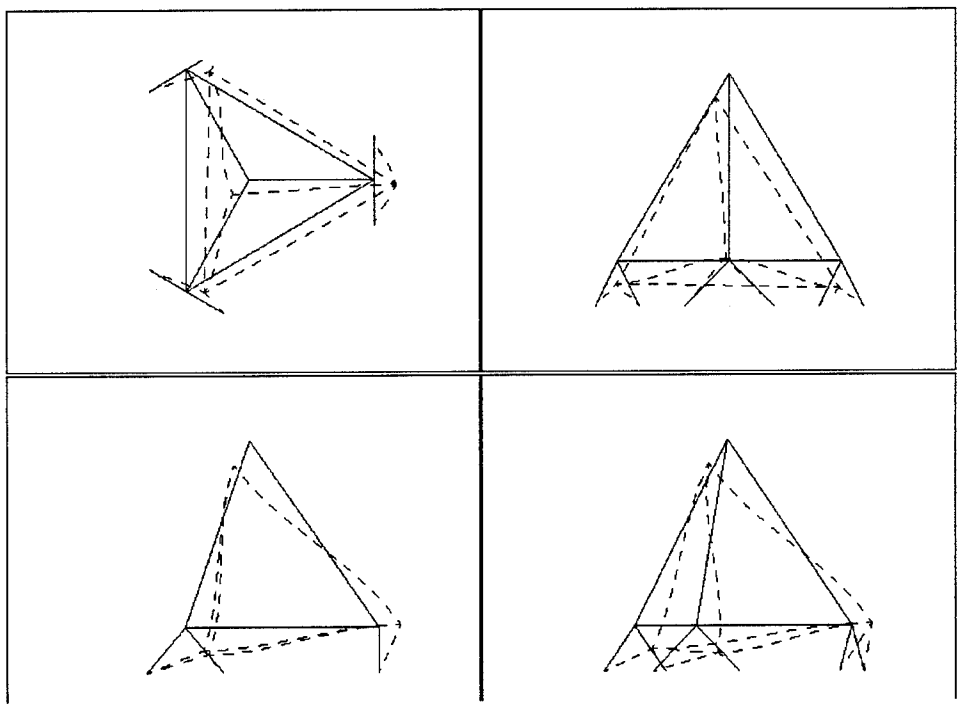


Figure 4-13a: Iteration 1/FD Mode 3 (1st In-Plane Rocking)

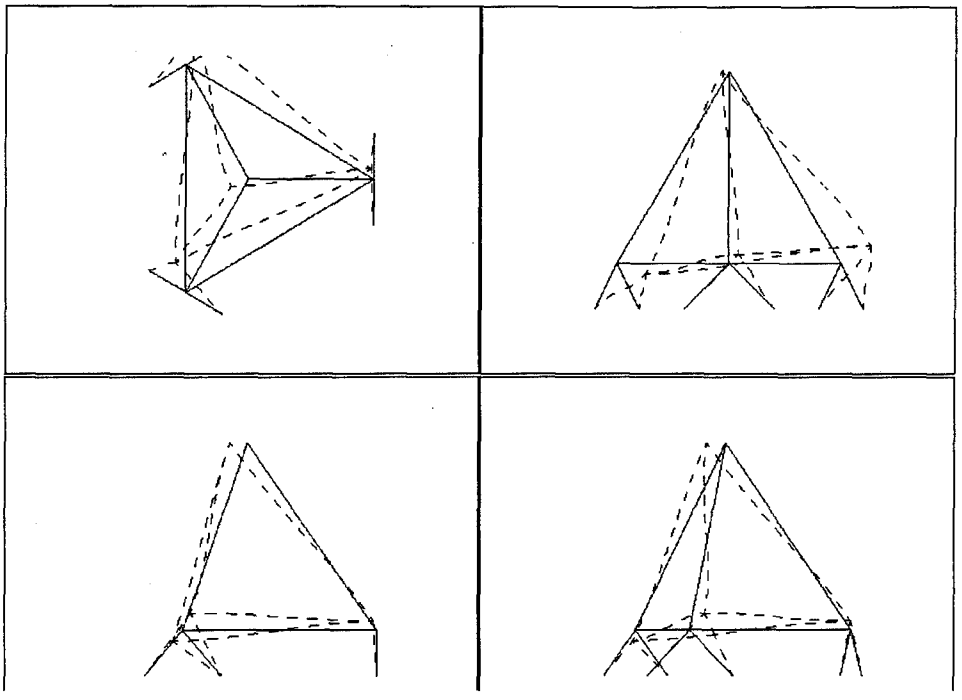


Figure 4-13b: Iteration 1/FD Mode 4 (2nd Out-of-Plane Bending)

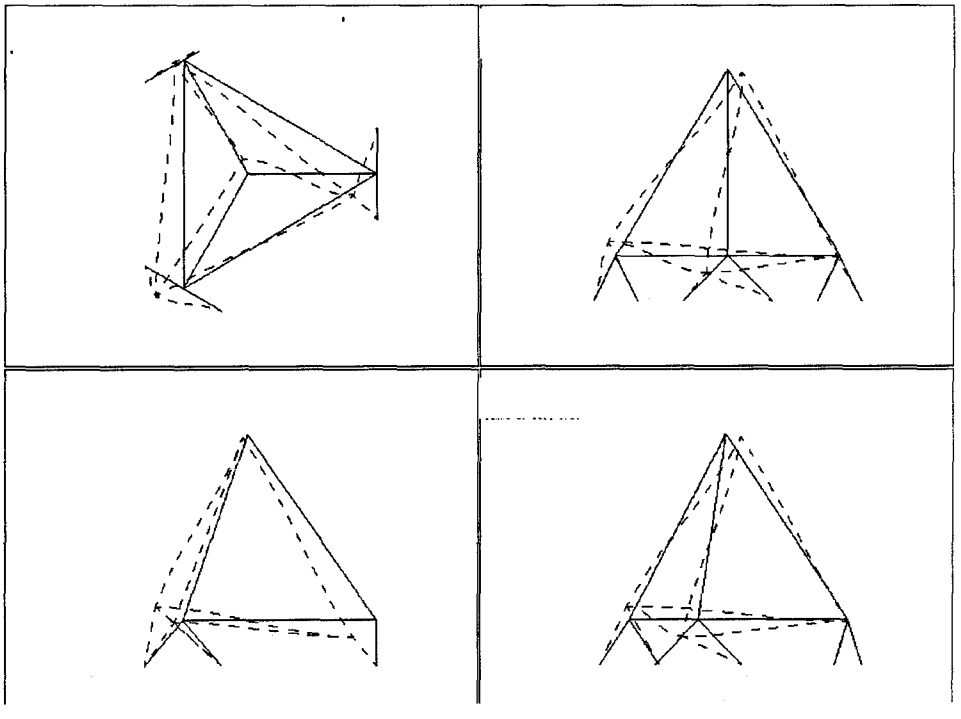


Figure 4-14a: Iteration 2/FD Mode 3 (2nd Out-of-Plane Bending)

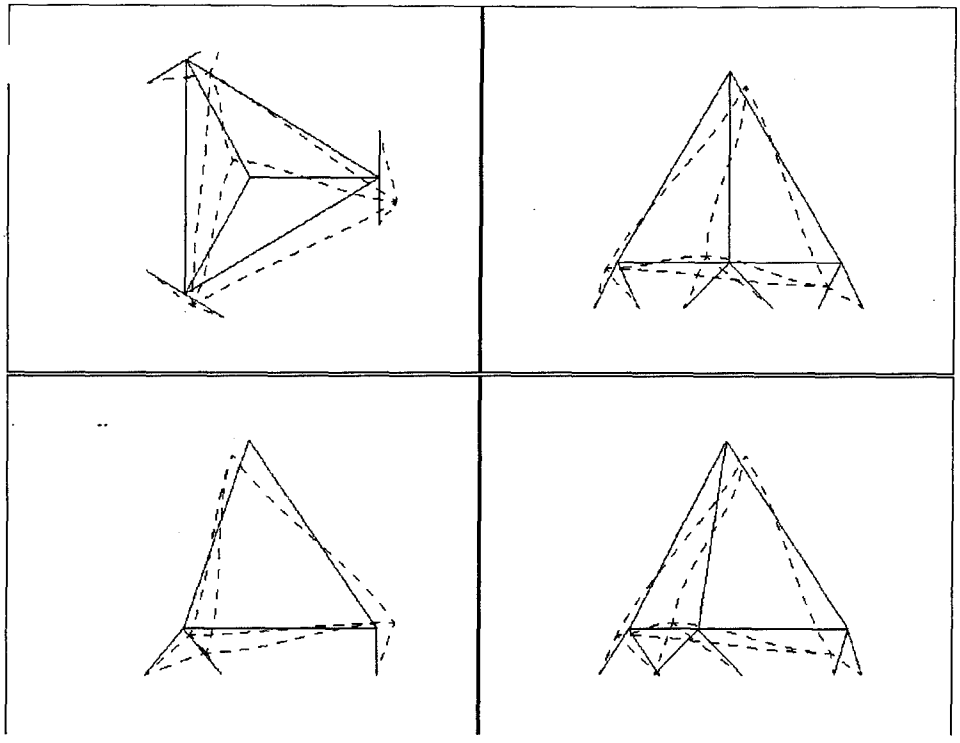


Figure 4-14b: Iteration 2/FD Mode 4 (1st In-Plane Rocking)

Although there was some concern that the planar motion in Mode 4 of Iteration 2 did not clearly indicate a preference for in-plane motion, there was no mistaking 1) the absence of axial motion in Iteration 2, mode 3; 2) the switch from almost completely *in*-plane motion in Iteration 1 to almost completely *out*-of-plane motion in the subsequent iteration; and 3) the phase carry-over between Iteration 1: Mode 3 and Iteration 2: Mode 4. Thus it was clear that a mode swap in the finite difference solution had occurred. Note the location of the third eigenvalue to that of the fourth eigenvalue at the onset of the nominal design -- they are the closest two frequencies on the chart. Modes 3 and 4 were connected to show the general flow of the iterations.

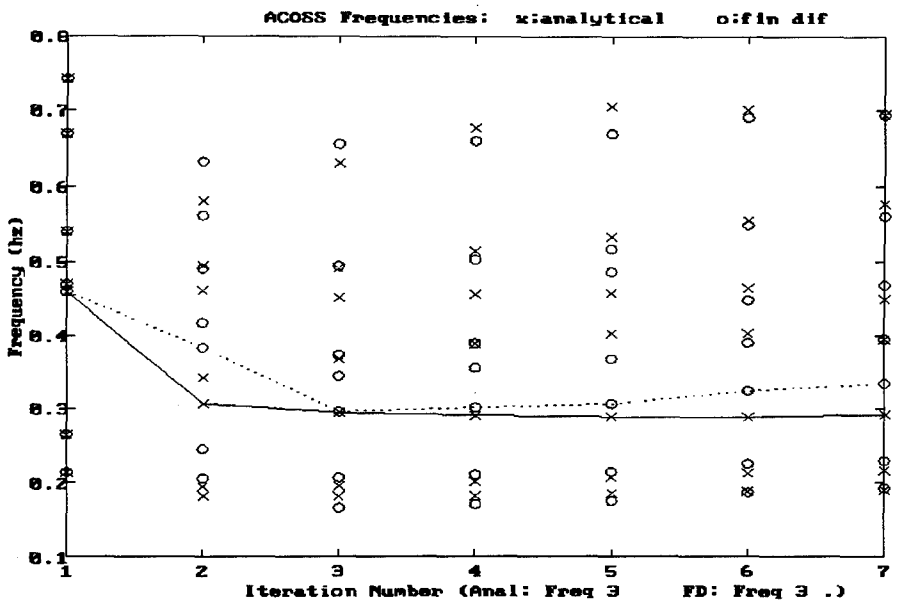


Figure 4-15: Superposition of Iteration Histories

The results of the finite difference iterations were plotted in the same fashion as those for the analytic methods as shown in Figure 4-15, although not all lines have been included for clarity. These plots were then superimposed with the identical plots from the analytic derivative solution and the finite difference solution to determine trends. Notice the continuing divergence of the Mode 3 frequencies between the analytic and finite difference solutions.

Table 4-5

ACOSS4 Mode Characterization using FD Derivatives

MODE	Nominal			IT 1			IT 2			IT 3			IT 4			IT 5			IT 6		
Char.	M	P	ϕ	M	P	ϕ	M	P	ϕ	M	P	ϕ	M	P	ϕ	M	P	ϕ	M	P	ϕ
1	1B	O		1B	O		1B	O		1B	O		1B	O		1B	O		1B	O	
2	1B	I		1B	I		1B	I		1B	I		1B	I		1B	I		1B	I	
3	1R	I	I	1R	I	I	2B	O		2B	O		2B	O		2B	O		2B	O	
4	2B	O		2B	O		1R	I	I	1R	I	I	1R	I	I	1R	I	I	1R	I	I
5	2R	I	O	2R	I	O	2B	O		2R	I	O	2R	I	O	2R	I	O	2R	I	O
6	1T			1T			1T			1T			1T			1T			1T		
7	1S	O	I	1S	O	I	1S	O	I	1S	O	I	1S	O	I	1S	O	I	1S	O	I

Recall, the original premise was that the optimizer, when using analytic derivatives in calculating the eigenvector gradients could not recover from a mode swap - literally applying at least two elements with reactive actuator control forces to which they did not correspond. There were no mode swap found in the analytic solution. This mode swap *seemingly* had to occur for the solution to converge to the correct feasible solution. However, the optimizer, when supplied with finite difference gradients, did allow the critical mode swap and the correct solution could be calculated. Clearly this development warranted additional investigation and the next plot was completed to determine if there was a connection.

The eigenvalues, together with the eigenvalues from the solution based on analytic derivatives, were plotted along with the objective function, $\mathbf{F(X)}$, i.e. the minimized weight of the ACOSS4 structure. This plot is shown in Figure 4-16. Note that the mode swap occurs in Modes 3 and 4 in the finite difference solution (solid lines) just as it seems to home in on the eventual correct solution. The solution based on analytic derivatives (dashed lines) at the same juncture seems to get on, and then follow an incorrect path. So that the magnitude of the constraint forces were not neglected by narrow-mindedly concentrating only on the activity of the objective function, the magnitudes of the three highest constraint forces were also analyzed (Figures 4-17 and 4-17b). These have been plotted with the objective function against iteration. Since the objective function is several orders of magnitude higher than the constraint forces, the constraint forces have been separately plotted in Figures 4-18a and 4-18b for clarity.

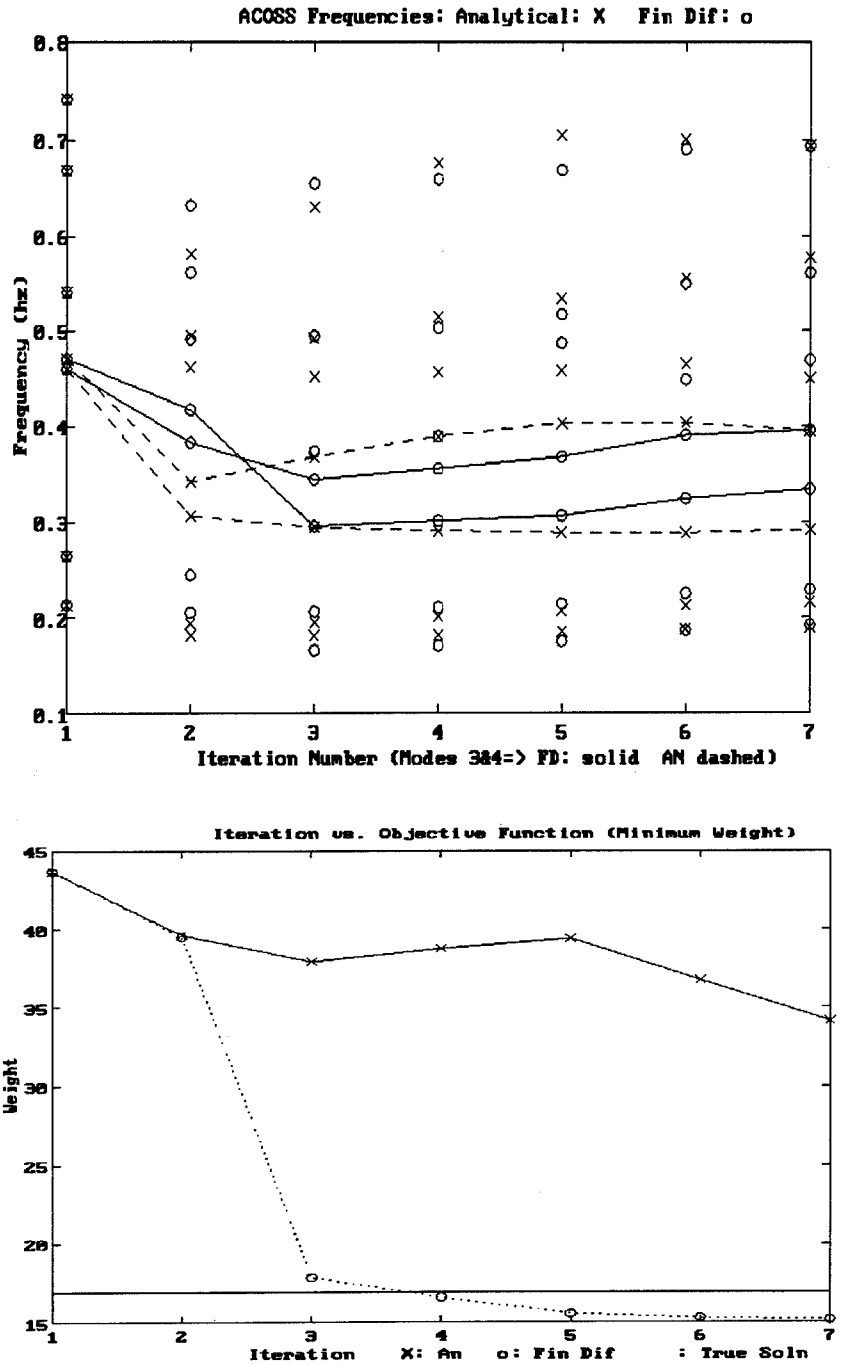


Figure 4-16: Effect of Frequency Cross-Over on Objective Function

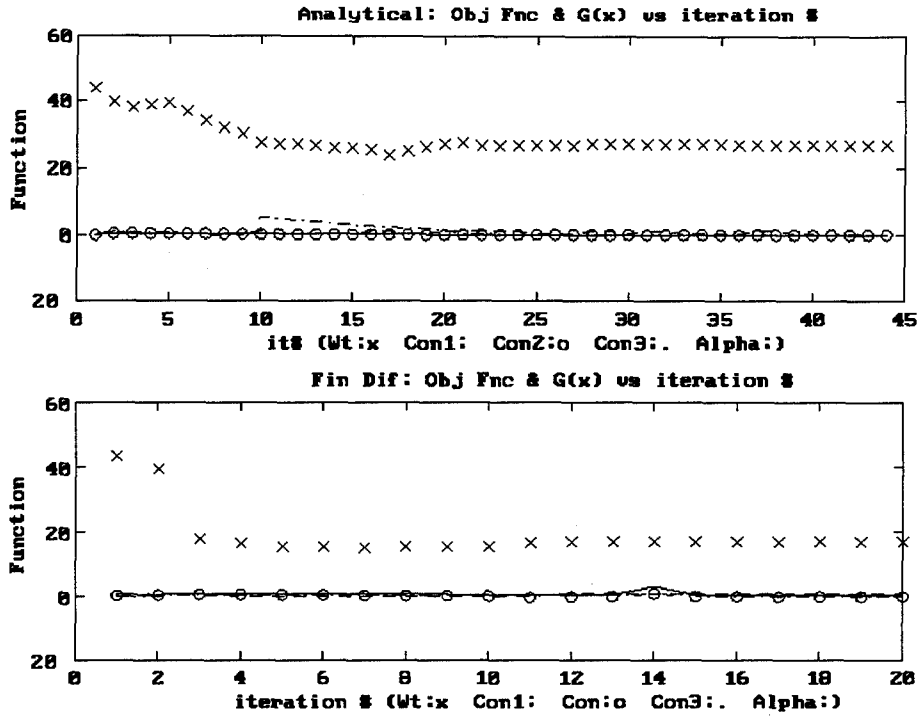


Figure 4-17a&b: Maximum Constraints vs. Iteration Number

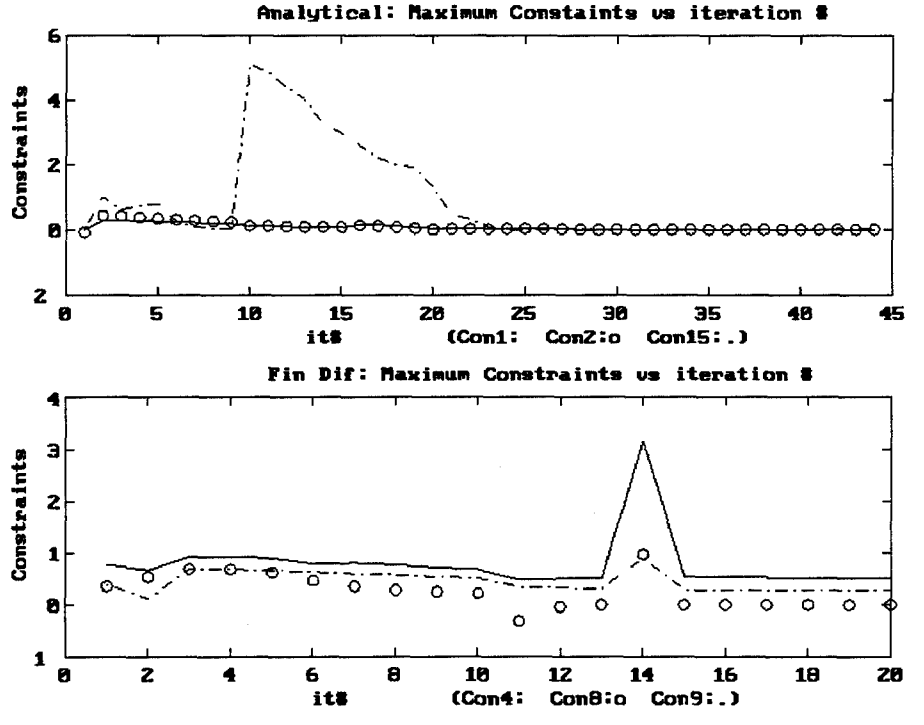


Figure 4-18a&b: Maximum Constraints vs. Iteration Number (Magnified)

As the reason was uncertain as to why the finite difference solution allowed a mode swap which apparently led to the correct solution, and why the analytic solution did not allow the critical mode swap, some preliminary checks of the first two possible hypothesis -- those concerning the accuracy of the eigensolver -- were investigated.

4.6 Eigensolver Accuracy

Access to the IMSL routines which perform the eigenanalysis was not an option. Therefore, the subroutine in ***FRAME*** directly involved with calculating the eigenvectors themselves was modified to print out the mass-normalized matrix $[\Phi^q]^T[M]^q[\Phi]^q$ if two parameters were both exceeded. The first parameter was the condition number of the matrix, a parameter which measured a matrix's sensitivity to data errors (or perturbations) [Strang, 1988:364]. (It should be noted that there is an error in the IMSL code of subroutine *DGPISP* which results in an incorrect condition number when *DGPISP* is sent multiple vectors. IMSL has been notified, but the error is not widely known. The correct condition number was calculated using an alternate subroutine.)

The second parameter, DELTA was a tolerance. The off-diagonals of a perfectly mass-orthonormalized matrix should all be zero, meaning the eigenvectors are perfectly orthonormal. Those matrices not exactly orthonormal will have non-zero off-diagonal elements. DELTA was the tolerance to which these off-diagonals terms were compared -- any elements greater than DELTA forced the subroutine to output the entire matrix.

Several of the matrices output from ***FRAME*** were analyzed and the following discussion applied to all. The diagonal terms were seven orders of magnitude higher than those terms on the off-diagonals as expected but some of the off-diagonals warranted suspicion. While most of the off-diagonals were on the order of 10^{-2} and 10^{-3}, six terms were an order of magnitude higher. The elements on the off-diagonals indicate the accuracy with which eigenvectors are calculated; the largest ones, those warranting note, are highlighted in the following discussion. (Note: as the matrix is symmetric, the row and column numbers in the following can be interchanged).

Element [9,10], at a magnitude of 0.101, indicated Modes 9 and 10 did not meet the orthogonality criteria precisely and thus were not "as orthogonal" as the majority of other terms in the matrix. These frequencies were also adjacent, meaning they were close enough to be repeated frequencies, since the eigensolver sorts the frequencies by magnitude. This *could* have indicated one of the problems caused by repeated frequencies as discussed in Chapter 2. However, these modes were both higher modes. The higher modes were not controlled in the ACOSS4 problem, and could not therefore have been the source of the problem. The next off-diagonal term warranting investigation was element [11,4] = 0.105, which indicated Modes 11 and 4 were not completely orthogonal. This *could* have indicated a problem since Mode 4 was a lower mode and was controlled. However, there were six frequencies between the two -- since the eigensolver sorts by magnitude there is virtually no chance these modes were adjacent, and therefore no chance they were repeated modes. This same argument was applied to the last relatively large off-diagonal term, element [10,6], at a magnitude of 0.120,

which indicated an orthogonality problem between Modes 6 and 10, but which too would not have affected the solution.

After applying this same reasoning to other mass-orthogonalized matrices output from the modified routine, it was clear that the source of the problem experienced while using analytic derivatives was **not** due to the numerical accuracy of the software or hardware, nor was it the result of mixed precision matrix operations. Therefore, the first two of the four original possibilities were abandoned. Because the only major difference between the two solutions was the visual verification of a mode swap in the finite difference solution, it was decided that this phenomena was the key to determining the reason the finite difference solution worked when the solution based on analytic derivatives did not. Efforts were therefore channeled into pursuing a quantitative -- *a mathematical* -- method to verify what had been verified qualitatively.

4.7 Mode Tracking Routines

4.7.1 Gibson's Cross-Orthogonality Check

As described in Chapter 2, the most applicable mode tracking algorithm for the ACOSS4 and COFS problem appeared to be Gibson's **C**ross **Or**thogonality **C**heck (CORC) [Gibson, 1992: 2]. To reiterate, this method is based on the fact that not only is every mode from a mass-orthonormalized matrix orthonormal to every other mode from that matrix, but every mode is also *relatively* orthogonal to every other mode in subsequent iterations of that matrix (assuming those iterations correspond to small changes in the design vector).

As explained in Section 2.6, the individual elements of the CORC matrix indicate the degree to which the modes in the current matrix align with the modes from the previous iteration. Systems that are relatively orthogonal will have some non-zero off-diagonal terms, but these terms will be much smaller in magnitude than the diagonal terms. The row number, *i*, of the largest element in each individual column of a diagonally dominant matrix should therefore correspond to the column number, *j*, of the previous mode. If the row number and column number differ, it is the row number which indicates the mode swap that occurred between mode *i* and mode *j*.

Steps must then be taken to interchange the i^{th} and j^{th} columns. A subroutine incorporating this method of first identifying and then completing the mode swap was written by the author. The method processes each column individually and uses only the magnitude of the individual elements. Consider the j^{th} column. The method is comprised of four steps:

1) The j^{th} column of the matrix is sorted in descending order along with a permutation vector representing the original order of that column. The first element of the permutation vector is the original row of the largest element.

2) That row is stored in the j^{th} slot of a tracking vector.

3) If that row has been previously assigned, the routine repeats Step 2 now using the next element (which represents the original position of the next largest element in the original column), until a row is found that has not yet been assigned.

4) Once steps 1-4 have been performed on all columns, the tracking vector is compared to a sequential vector. If the two vectors are not identical, a mode swap has occurred, as flagged by the tracking vector [Gibson, 1992:6].

A flow chart is included in Appendix E.2 and the actual ForTran 77 code in Appendix F.1. Because this method processes each row *individually,* the mode tracking algorithm is highly dependent on order. A numerical example best demonstrates this handicap.

Consider a system with only three degrees of freedom and assume the resulting CORC matrix is

$$\boldsymbol{CORC} = \begin{bmatrix} 0.7767 & 0.0936 & -0.0230 \\ 0.5126 & -0.8585 & 0.9893 \\ 0.1115 & 0.6295 & 0.0143 \end{bmatrix} \qquad (4-4)$$

Steps 1 - 3 will eventually result in the following index vectors:

$$\boldsymbol{indexV}_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \quad \boldsymbol{indexV}_2 = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} \quad \boldsymbol{indexV}_3 = \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix} \qquad (4-5)$$

In processing the first two columns, the routine outputs what one would expect for a diagonally dominant matrix, i.e. the largest elements are on the diagonal. Thus, the largest element in the *first* column of the CORC matrix was in row *1* and as expected the row and column match. Similarly, the largest element in the *second* column was in the *second* row and again the row and column match. Modes 1 and 2 have now been permanently assigned. The routine is now forced to process the third and last column. The largest number appears in the second row, but that mode has already been assigned. The next largest number appears in the first column, but that mode too has been assigned. Therefore the mode is assigned to the Mode 3, with a correlation factor of less than 2% versus the 100% correlation expected in a perfectly mass-orthonormal matrix. The order of the modes is [1 2 3]. This matches the

results of the previous optimization, and thus no mode swap is indicated, although one has occurred between Modes 2 and 3.

Processed in reverse order from column 3 to column 1 results in the same index vectors as shown above. However, the routine is forced to assign the row number of its largest element, resident in row 2, to the same mode as the column being processed, Mode 3. In processing the second column, the routine is forced to eliminate its first choice, which has already been assigned, in lieu of Mode 3. Row 1 is assigned as Mode 1 and the routine outputs a *CTRKR* of [1 3 2], which, when compared to the previous iteration's mode ordering of [1 2 3] correctly identifies the mode swap that did occur.

The routine was implemented in ***FRAME*** as described in above. Since the ACOSS4 problem has twelve DOF, the matrix product *CORC* is a 12x12 matrix. The following is a partition of the CORC matrix consisting of the first 6 rows and 6 columns. The remaining partitions of the matrix do not contain any terms that affect the following discussion.

$$C_{6x6} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} -0.7663 & 0.5193 & -0.2440 & 0.2586 & -0.0091 & -.0993 \\ 0.3542 & 0.79513 & 0.3507 & -0.1957 & -0.2400 & 0.1069 \\ 0.0452 & 0.0101 & 0.5496 & 0.7203 & 0.3634 & 0.1425 \\ -0.4005 & -.0727 & 0.3350 & -0.5298 & 0.3634 & 0.5390 \\ 0.2744 & 0.2875 & -0.3656 & -0.1113 & 0.8043 & -0.1427 \\ -0.1792 & -0.0519 & -0.4880 & -0.2674 & 0.1289 & -0.79518 \end{bmatrix} \end{array} \qquad (4-6)$$

The routine will find the largest elements in rows 1-3 to be on the diagonal as expected for a diagonally dominant matrix. Upon applying the algorithm to column 4, however, the largest element again is found in row 3 -- *Mode* 3. Mode 3 has already been assigned, how-

ever, and the routine is forced to find the next highest element, -0.5298, and assign it to Mode 4. Modes 5 and 6 are assigned as expected. Again, the elements of the CORC matrix represent the degree to which the current mode is influenced by the previous mode. Mode 4 is strongly correlated with Mode 3 (0.72), while Mode 3 is only loosely correlated with Mode 3 (0.54) Thus, Gibson's method missed the mode swap which had been verified visually. The program was therefore set back on its original course, and resulted in the same solution -- the program failed to converge to a feasible solution. However, in processing the 6x6 partition in reverse order, the routine does locate the mode swap. With the Gibson flaw referenced in [Eldred, 1994: 5] experienced, it was clear a more robust cross-orthogonality check which was not dependent on the order the columns were traversed had to be built.

4.7.2 Modified Cross-Orthogonality Check

As clearly evidenced from the sample problem and the actual results, Gibson's algorithm suffers because it is forced to analyze only one vector at a time while ignoring other critical data. Thus, a more robust CORC algorithm was needed to consider the entire matrix at one time, vice column by column. This new algorithm is designed to search the matrix for its largest element, assigning its row to a tracking vector just as was done in the previous method. Because that column will not provide another mode, the entire column is zeroed. Also, because that mode can not be assigned again, that row is likewise zeroed. This process continues until all mode are assigned. A flowchart of the routine is included in Appendix E.3. The actual ForTran 77 code is included in Appendix F.2.

One other check also seemed appropriate. Recall, a single eigenvector is uniquely determined except for an arbitrary scaling factor. Any eigenvector can be scaled by a constant or can be multiplied by -1. Scaling was not an issue in this problem, as the problem was mass-orthonormalized. However, between iterations, modes may have been scaled by a factor of -1. To ensure this was not an issue with the optimizer, a block was included to determine if the corresponding eigenvectors had opposite algebraic signs. The CORC matrix is a product of three matrices, one of which (the mass matrix) is always positive. Therefore, after calculating CORC, a negative element will indicate that the eigenvectors may have switched sign. Since the modes between iterations are different, however, not **all** the signs may have changed. Thus, it is necessary to base the sign change on the most critical element of that matrix, that element which establishes modal correspondence. For example, in the 3x3 matrix of Eq. 4-4,

$$\boldsymbol{CORC} = \begin{bmatrix} 0.7767 & 0.0936 & -0.0230 \\ 0.5126 & -0.8585 & 0.9893 \\ 0.1115 & 0.6295 & 0.0143 \end{bmatrix}$$

there was a mode swap between modes 2 and 3. $CORC_{2,3}$ is 0.9893, which is positive -- therefore the eigenvectors of each iteration had the same sign (whether they were both negative or both positive). Likewise, there was no mode swap between the first two modes, nor between the second two modes as verified by $CORC_{1,1}$ and $CORC_{3,2}$ respectively, which are both positive. Had the value of $CORC_{3,2}$ been -0.6295, the entire second column of the current eigenvector (i.e. Mode 2) would have been multiplied by -1.

The MCORC routine was then implemented in ***FRAME*** and the program re-run. When it was confronted with the same 12x12 CORC matrix, the routine found the six largest elements assigned to the partitioned diagonally dominant matrix in the lower left quarter of the CORC matrix. Thus, Modes 7-12 were immediately assigned. The routine was now passed the following matrix:

$$\left[\begin{array}{c|c} C_{1,1} & C_{1,2} \\ \hline C_{2,1} & C_{2,2} \end{array}\right]$$

where $C_{1,2}$, $C_{2,1}$, and $C_{2,2}$, are 6x6 matrices of zeroes. $C_{1,2}$ is the same as in Eq. 4-6. The algorithm next correlates Modes 5 and 6, then Modes 1 and 2. Finally, the modes with the weakest correlation, Modes 3 and 4, are processed and the mode swap observed (visually) earlier, is correctly identified. This example is included in detail in Appendix E.5.

This algorithm, because it is designed to consider the entire matrix each time, does not suffer from the ordering problem as does Gibson's. In addition, this algorithm is more efficient than is Gibson's. It is based on *searching* a matrix, vice *sorting* one, which in most cases is faster. Also, because it zeros out entire columns and rows in each assignment, it reduces the number of potential re-assignments due the structure of inequality checks.

Unfortunately, this code fails when presented with an infeasible structure, which results in a singular matrix. To avoid the confusion this would cause the user, additional code was built into *MCORC* to gracefully exit the *MCORC* subroutine and instead run the mode tracking algorithm designed by Gibson (described above) if certain validity tests fail. The program does alert the user of the input error, and that the mode tracking routines were switched.

MCORC was designed to assign modes to a CORC matrix with a rank defect of one using the process of elimination. Although *MCORC* is still executed rather than switching to Gibson's method (***GIBSON***), the user is alerted to the error so the structure can be corrected. The validity tests are described in Appendix E.4 which includes a flowchart of the routine. This improved mode tracking algorithm was incorporated into ***FRAME***, and the program executed with the hopes of convergence. Again, the program failed to converge at nearly the same point as it had originally.

4.8 Regroup: Formulation of Contingency Strategy

A summary of results thus far is appropriate to introduce the subsequent strategy, for which had not been originally planned. The original premise of this research was to determine the reason the integrated structural design and control problem failed to converge to a feasible solution when using analytic derivatives of eigenvectors to determine the search direction. The strongest hypothesis at the time was that a mode swap had occurred during the optimization and had corrupted the search direction to the point that it could not recover.

A mode swap between Modes 3 and 4 was observed visually and that same swap confirmed mathematically using a cross-orthogonality check. The mode swap occurred when the optimizer had been based on a search direction which in turn was based on finite difference derivatives of the eigenvectors. No such swap was seen when the optimizer was equipped with analytic derivatives. Because the swap occurred during the former approach, and the former approached converged successfully, it appeared to be the key to the problem.

The incorporation of the mode tracking algorithm in the optimization loop did not alter the results significantly. Figures 4-17, 4-18, and 4-19 indicate the divergence of the analytic solution had to occur between iterations 1 and 2. As explained in Section 3.5, finding the correct search path from

$$\boldsymbol{X}^q = \boldsymbol{X}^{(q-1)} + \alpha^* \boldsymbol{S}^q \qquad (3-45)$$

requires two things: 1) the correct direction, and 2) the correct step size. The program was run using analytic derivatives and finite difference derivatives of eigenvectors. We noted that between iterations 1 and 2 in the finite difference solution the optimizer took a unit step, which is desirable. However, the solution based on analytic derivatives took a step size of only 0.1643. This appeared to be a significant part of the problem and it was decided to more closely examine the behavior of the optimizer between iterations. The code in ***FRAME*** was modified to display results between iterations when equipped with analytic derivatives, the mode tracking algorithm was de-activated, and new FEMs of the resulting models were constructed. (This same procedure could not be applied to the finite difference routine, as the differencing routine is proprietary and thus can not be modified in such a way that the correct modal matrix can be retrieved.) The iteration number was defined as 1.1 where the decimal denotes the additional step required after the nominal (unit) step size failed. In the interest of conciseness, only the significant plots are included in Figures 4-19a-c. These can be compared to Figures 4-10c-e. Modes 1, 2, 3 and 7 closely matched those from Iteration 1, but each now had a small component of torsion. A brief analysis follows:

4.8.1 Iteration 1.1

Mode 4 (1st Torsion): A mode swap between Mode 6 of Iteration 1.0 and this interim solution is clearly evident. Mode 4 now has a significant axial component. This motion resembles the motion one would describe when screwing the lid off a can whose thread has a large pitch angle. (See Figure 4-19a.)

Mode 5 (2nd Out-of-Plane Bending): There is some axial motion but not as significant as in Mode 6. This motion most clearly resembles Mode 4 of Iteration 1.0. (See Figure 4-19b.)

Mode 6 (2nd In-Plane Rocking): There is a significant axial component present. 2nd Bending motion is apparent in Element E3. Some torsion is evident but it is not as significant as that in Mode 4. (See Figure 4-19c.)

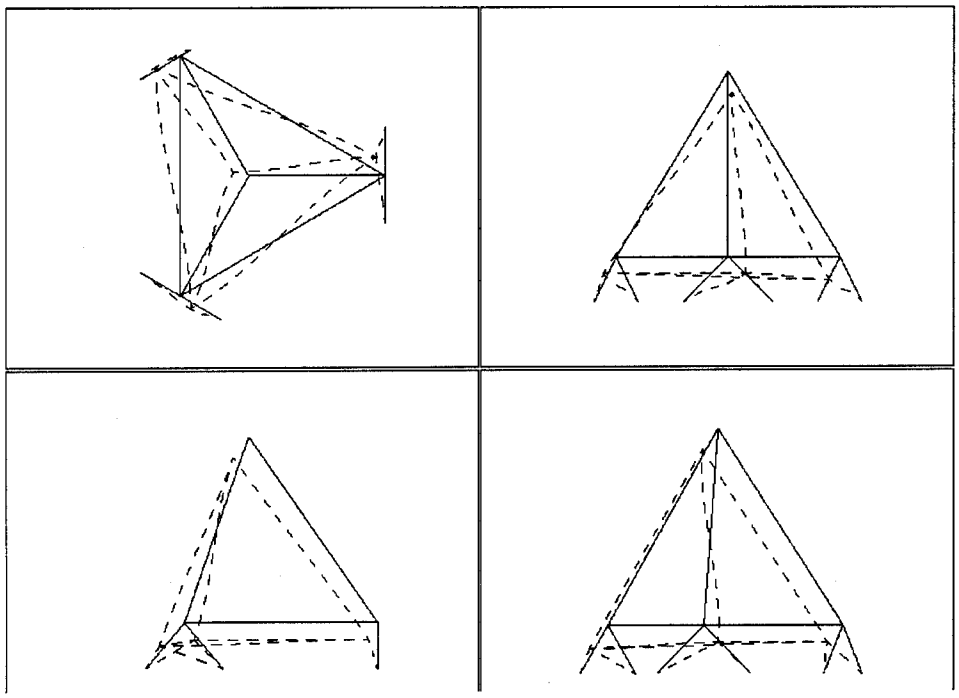


Figure 4-19a: Iteration 1.1/Analytic Mode 4 (1st Torsion)

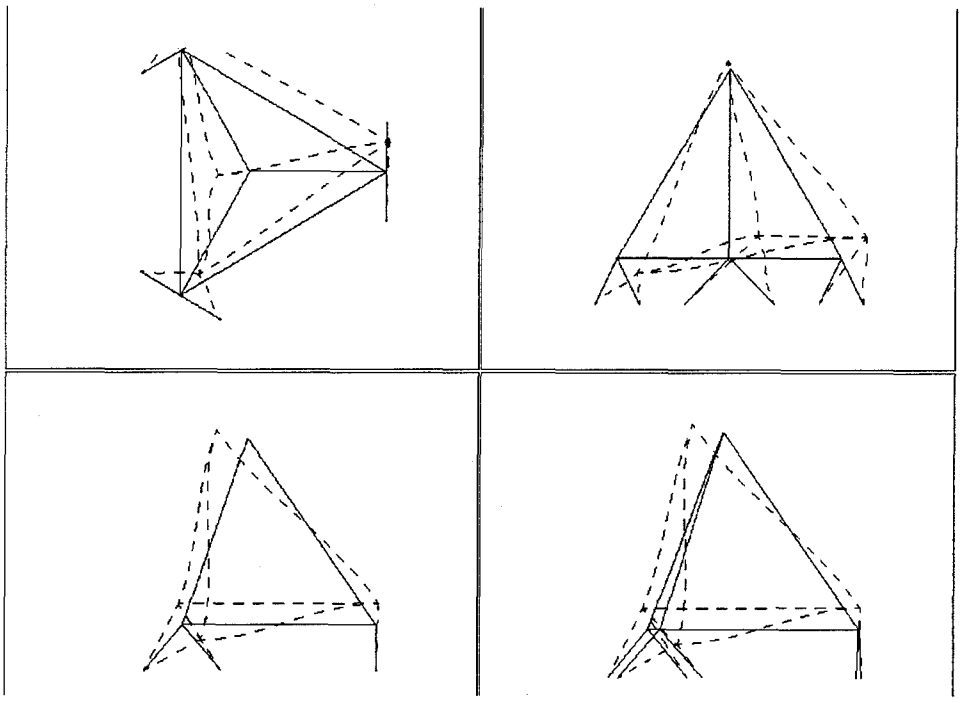


Figure 4-19b: Iteration 1.1/Analytic Mode 5 (2nd Bending)

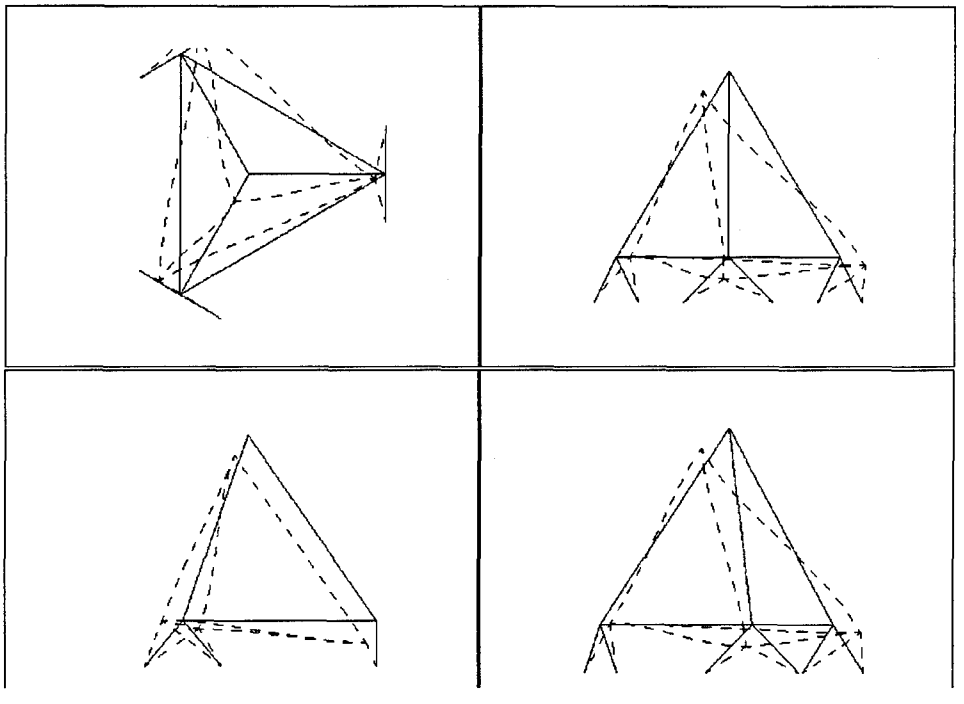


Figure 4-19c: Iteration 1.1/Analytic Mode 6 (1st Rocking [Out-of-Phase])

FRAME was then run using analytic derivatives. The mode tracking algorithm was left on, but modified so that no action would be taken once the mode swap was detected (i.e. the modes and eigenvalues were not permuted). Just as was done visually, the algorithm mathematically detected the same mode rotation of Modes 4, 5 and 6 between iterations 1 and 2. The ordering had gone from 1234567 to 1236457 when the optimizer had taken a full step of 1. It was unclear at this point whether or not the mode rotation had *caused* the optimizer to back off from a step size of 1. The end result was that the optimizer took a step size of only 0.1644, and while this small step did **not** cause a mode swap, it did cause the algorithm to head off in a new search direction from which it could never recover.

As had been suspected, mode swapping was indeed a problem between iterations 1 and 2 in the solution based on analytic derivatives. However, its effects appeared minimal, and it was determined that a mode swap was not the cause of the non-convergence problem in the solution based on analytic derivatives. The following was also concluded:

1) There existed a more complex error in the manner in which the analytic derivatives of the eigenvectors were calculated and,

2) While the finite difference result did experience a mode swap, the algorithm recovered and was still able to converge to a feasible solution. Graphically, one could imagine the problem (if reduced to a two variable space) to be as shown in Figure 4-20. The dashed line represents a possible route to the optimum solution which the finite difference solution could have taken if it had the benefit of mode tracking.

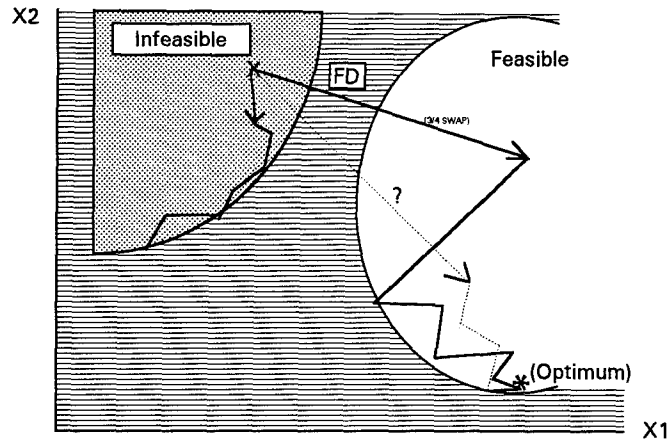


Figure 4-20: Graphical Representation of Hypothesis in $\Re^2$

Two causes of the failure now seemed possible:

1) There could be some unforeseen error in Nelson's Method which makes it inappropriate for the problem. (As explained in Chapter 3, the analytic derivatives of the eigenvectors are calculated using Nelson's Method.)

2) There could be an error in the ***FRAME*** code.

To determine which solution was most likely, another program which also used Nelson's Method to calculate the eigenvector derivatives was required as a reference. If the derivatives of *this* program compared more closely to the analytic derivatives computed from ***FRAME***, this would point to an error in Nelson's Method, since it was considered improbable that two programmers, working independently, would make the same coding error. However, if those same derivatives compared more closely to the finite difference derivatives, an error in the original coding of ***FRAME*** would be the more likely problem.

The finite element program ASTROS was made available from our sponsors at Wright Laboratories and served as the reference. The finite element model of ACOSS4 was converted into a NASTRAN model compatible with ASTROS. ASTROS computed and output the eigenvector derivatives, included in Appendix C.3. (A preliminary review of the ASTROS data indicated major differences. Upon closer examination, an error was found in the output subroutine of the ASTROS code. This error was noted and reported to our sponsors.) Likewise, ***FRAME*** was configured to alternately run with finite difference and analytic eigenvector derivatives, and the output assembled into tables. A reduced set of derivatives corresponding to only the first six modes for Iteration 0 are included in Appendices C.1 and C.2.

These results were then loaded into a statistical analysis package written in MATLAB by the author specifically for this purpose. The program is included in Appendix G. The raw data from the program was then compiled and the results included in Appendix C.6. In both cases, the program took the finite difference results to be the *actual* solution, and the ***ASTROS*** and ***FRAME*** solutions individually taken as what would be considered the *experimental* solutions.

A matrix of the finite difference eigenvector derivatives was subtracted from the corresponding matrix of eigenvector derivatives calculated by ***ASTROS*** and by ***FRAME***. Relative percentages were calculated using elementary statistics, and the mean and standard deviations of that relative matrix computed. Since the above difference operations were carried out in absolute value, and therefore did not account for sign, the signs of the eigenvectors were also analyzed. A cursory review of ***FRAME***'s eigenvector derivatives seemed to

indicate they differed from the FD derivatives by a factor of 2. Therefore, a sub-routine was written to determine how many elements in the matrix based on ***FRAME*** were within 25% and 50% of those elements in the FD matrix.

The results indicated there was no correlation between the analytic derivatives from ***FRAME*** and the correct eigenvectors calculated by finite difference methods. There was, however, a strong correlation between the derivative matrix computed by ***ASTROS*** and the FD matrix. The individual elements of the modal matrices, after accounting for sign, were off an average of only 1.3%. Also, of the 864 terms, the ***ASTROS*** derivatives differed in sign from the FD only 3 times (one term in Mode 4 and two terms in Mode 6). The presence of the strict zeros in the finite difference terms and the absence of the same in the other two modal matrices could not be explained.

4.9 Semi-Analytic Methods

Knowing the sensitivity computations were in error, but unsure of the location, the multiplicity, and the magnitude of the error(s), semi-analytic methods were employed. As noted in Section 3.4.2, the form of the semi-analytic differential equations, Eqs. 3-27-28

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{X}} = \begin{Bmatrix} 0 \\ \hline \partial \boldsymbol{g} / \partial \boldsymbol{x}_c \end{Bmatrix} + \begin{Bmatrix} \Delta \boldsymbol{g} / \Delta \boldsymbol{x}_s \\ \hline 0 \end{Bmatrix} \qquad (3-27)$$

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{X}} \cong \begin{Bmatrix} 0 \\ \hline \partial \boldsymbol{g} / \partial \boldsymbol{x}_c \end{Bmatrix} + \begin{Bmatrix} \frac{\partial \boldsymbol{g}}{\partial \Phi} \bullet \frac{\Delta \Phi}{\Delta \boldsymbol{x}_s} \\ \hline 0 \end{Bmatrix} \qquad (3-28)$$

can be used to help determine single point errors in code where derivatives are computed in separate subroutines. In addition, if an error could not be found, this hybrid approach offered some of the advantages of analytic computation (e.g. nominal increase in speed and flexibility, etc.) as well as a supposed guarantee of success (since the finite difference approach had converged previously).

Two approaches were used. Semi-analytic constraint (SAC) derivatives (Eq. 3-27) were processed through the VAX independently of the author to test the feasibility of the approach. The solution converged with a slight decrease in computational time. Since the second term in Eq. 3-27 (used in the SAC approach) was comprised of the finite difference derivatives (and known to be accurate from the previous section), and the $\partial \mathbf{g}/\partial \mathbf{X}$ terms were correct (the solution converged), the $\partial \mathbf{g}/\partial \mathbf{x}_c$ terms also had to be correct. The author then constructed a similar routine (*SEMPHI*) using semi-analytic derivatives. The purpose was to distinguish in which term of the chain rule the error resided. This method also provided convergence to the correct solution but was a full 33% slower than the strict finite difference approach and required almost twice as many iterations. Looking ahead to Figure 4-23, one can detect no wild oscillation present which might account for the additional iterations; rather, the optimizer appeared to approach the solution extremely cautiously. This could be a

manifestation of the non-optimal finite difference perturbation size, constraining the finite difference solution which is further hampered by the hybrid approach.

One last confusing aspect was that the solution based on SEMPHI had to have mode tracking enabled in order to converge. The alternate solution satisfied the original constraints set by FRAME and in that sense did arrive at the correct solution, but it failed to meet additional constraints imposed by IMSL within the required number of five steps per line search, and in that sense ultimately failed. A review of the output corresponding to the feasible solution could determine no direct link between the mode tracking and the success of the solution. Successful convergence seemed to indicate that the first term in the chain rule, $\partial \mathbf{g}/\partial \Phi$, was correct and that the error lay in the second term, $\partial \Phi/\partial x_s$. This observation was consistent with the ASTROS comparisons of $\partial \Phi/\partial x_s$. In the end, the solution was deemed unacceptable and the search for alternate means of analytic sensitivity analysis continued.

4.10 Expansion Theorem

The results from the hybrid approach using *SEMPHI* were non-spectacular. They did, however, strengthen the conviction that the eigenvector derivatives were probably in error, and that abandoning the original hypothesis on mode swapping had been a valid decision. A cursory review of the affected code, spread amongst some six separate subroutines, clearly indicated that there existed little chance the error could be found through mere scrutiny. Re-writing the code from scratch was considered. However, the data had thus far, (as had an expanded literature review), showed the ACOSS4 to be a deceptively simplistic structure and

could be extremely sensitive at times. Hence the possibility that Nelson's Method was coded correctly, but still providing inaccurate eigenvectors due to interaction with other facets of the ***FRAME*** code, had not yet been totally eliminated.

A different analytic approach was next introduced in order to meet the goal of equipping ***FRAME*** with a solution based on analytic derivatives. The modal expansion algorithm was selected as an alternative to Nelson's Method. As explained in Section 3.4.3, the modal expansion method would only work if the complete modal matrix could be found. The method was therefore appropriate for ACOSS4, but not so for COFS which had 435 columns in its a modal matrix.

Using the derivation included in Section 3.4.3, the ForTran 77 program *EXPNSN* (Appendix F.3) was written and integrated into the ***FRAME*** program. A flag was added to ***FRAME*** to allow the user to choose the method by which eigenvector derivatives would be computed. The program was run on the ACOSS4 model using IMSL routines to compute the eigenvalues and eigenvectors. The original subroutine *DKMDVP* was retained to compute the term

$$[\boldsymbol{K}'-\lambda\,\boldsymbol{M}']\phi$$

which is required by both Nelson's Method and the modal expansion method. The program failed to converge to a feasible solution, failing at the 39th iteration due to its inability to find a feasible search direction within the required number of five steps per line search. The mode swapping phenomena was again considered, and mode tracking was turned back on. This new solution failed to converge with mode tracking incorporated -- this time randomly

oscillating before failing in only five iterations. Mode tracking had seemingly made the design history far worse.

The new eigenvector derivatives were input into the statistics package used previously. Results are included in Appendix C.6. It was readily apparent that these derivatives differed from the finite difference derivatives by a consistent error of 50% -- i.e. they were consistently half the expected value. (Mode 6 was only 30% in error but this low figure was attributed to the large number of zeros returned from the finite difference routine and which had biased the results). In addition, there was only **one** sign difference error (out of 832 comparisons) between the results computed by *EXPNSN* and those computed through finite difference. It certainly appeared there was a missing factor of *2*. As a test, the eigenvector derivatives were scaled by a factor of 2 upon exiting the *EXPNSN* subroutine. The program was run, and converged in only thirteen iterations with little oscillation. The literature was again consulted and the *EXPNSN* code meticulously scrutinized for the missing factor, to no avail.

Recall there was no correlation between the eigenvector derivatives based on Nelson's Method yet the derivatives from *EXPNSN* were ***highly*** correlated. This difference did make it appear that there was no common link between the two failures. However, in comparing modal expansion and Nelson's Method, it was determined that the factor of 2 was apparent in *EXPNSN* because the routine is based on iterative summation; therefore a renegade factor of 2 could readily be factored by distributive law. This is not true in Nelson's approach as it is based on the sum of a homogeneous and a particular solution (Section 3.4.4), where a common factor would not be readily apparent.

The only routines common to both *GRDPHI* (where Nelson's Method is implemented) and *EXPNSN* were the IMSL routines used to calculate the eigenvalues and eigenvectors, and *DKMDVP*. The latter routine was tested first. The original scaling was applied to the output of *DKMDVP* rather than to that of *EXPNSN*. The iteration history was identical -- the program converged in only thirteen iterations. ***FRAME*** was then re-configured to execute *GRDPHI* vice *EXPNSN* using the doctored *DKMDVP* subroutine. Again, the iteration history was nearly identical. It was concluded that the error had to be in *DKMDVP*.

The *DKMDVP* code was carefully reviewed; the error was corrected and the worka-round scaling removed. ***FRAME*** was re-executed using *EXPNSN* and *GRDPHI* alternately to calculate the eigenvector derivatives. Every corresponding result from the two iteration histories was identical to the number of significant digits reported. The results are shown in Appendix D.3. Comparisons of the results from the original code and the corrected code using modal expansion and Nelson's Method are plotted in Figures 4-21.

4.11 Comparison of Methodologies

Finally, the iteration histories from the four methods which had successfully converged were compared. The plot is shown in Figure 4-23. Determining the total CPU time expended was simply a matter of recording results from a timing file inherent to ***FRAME***. Unfortu nately, the CPU times provided were not consistent over multiple runs. To minimize any error this inconsistency would generate, the timing results from multiple runs were averaged.

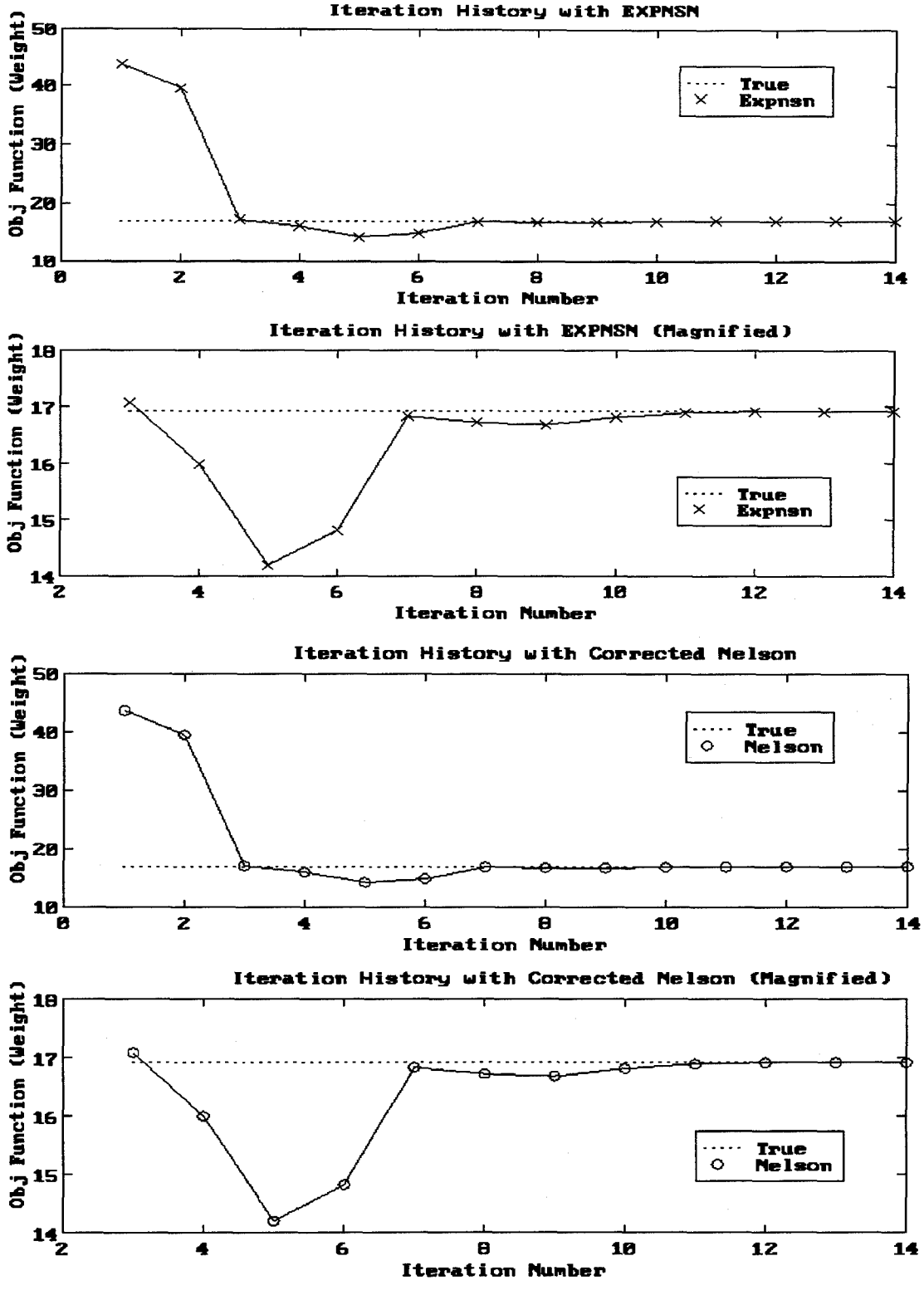


Figure 4-21: Iteration Histories for Modal Expansion (a&b); Nelson's Method (c&d)

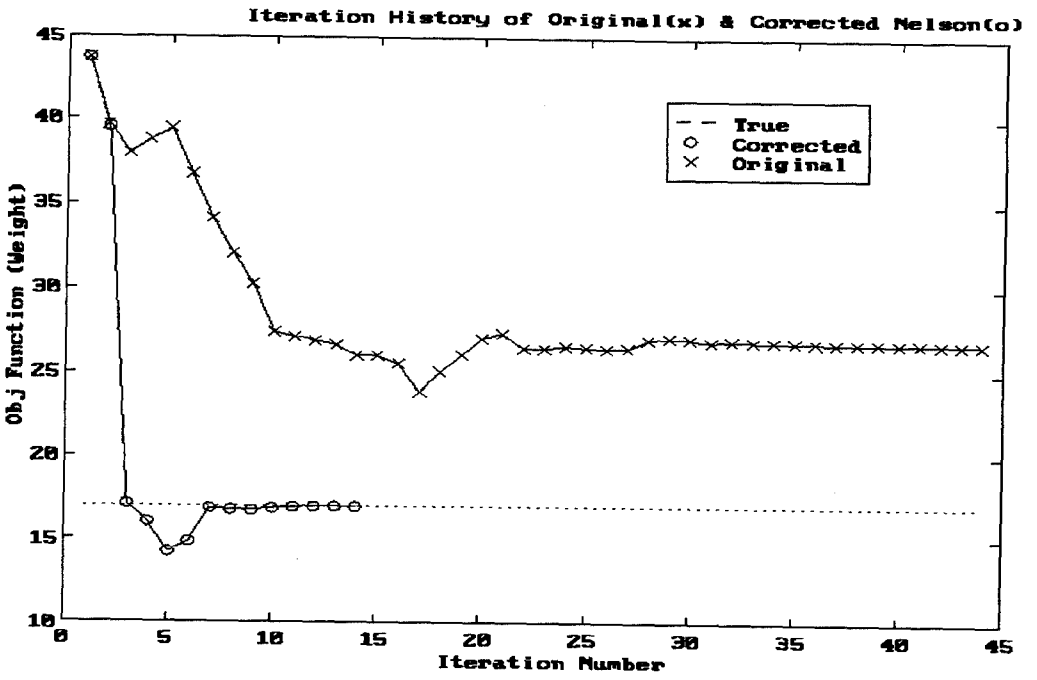


Figure 4-22: Comparison of Original and Corrected Nelson

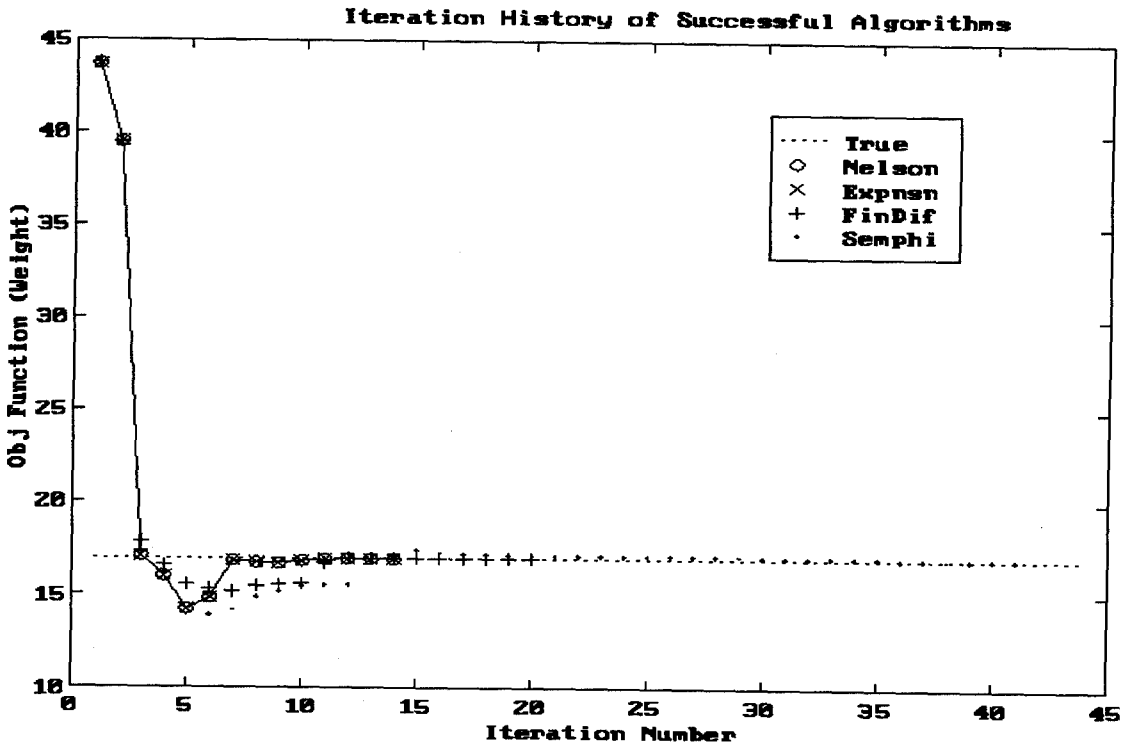


Figure 4-23: Comparison of Iteration Histories

The original finite difference run had required 20.7 CPU seconds, converging in 22 iterations. The objective function was consistent with that of the two analytic runs to four significant digits. That the finite difference solution required nine additional iterations than the analytic results was attributed to two factors. The first was the mode swap which was verified visually and mathematically. The Hessian matrix, being comprised of results from only two iterations at the time of the mode swap, was initially corrupted. Subsequent iterations, however, provided the Hessian the opportunity to recover. The iteration history was also impacted by the inflexibility of the IMSL finite difference routines -- specifically the inability to provide the routines with an optimal step size. There is some evidence to justify this assertion. The step size is based on the machine precision [IMSL, 1991: 1118]. When the optimization routine was run on the VAX, the solution converged in only 19 iterations. When the identical routine was run on the SPARC 20, convergence required 22 iterations -- three additional iterations solely due to the difference in machine precision.

The hybrid approach using *SEMPHI* took 30.47 CPU seconds to converge over 40 iterations. No clear reason why this solution should require more iterations than its finite difference counterpart could be determined. That the routine would only converge with mode tracking in place was also a mystery. Little time was devoted to studying these peculiarities, as the results were not useable.

The approach using eigenvector derivatives calculated by modal expansion required only 13 iterations and converged in 4.12 CPU seconds. The resulting objective function and design vector were consistent with *all* results based on Nelson's Method to eight significant digits. Such precision was unexpected. (The programs were even verified to ensure the

correct routines were being called.) With mode tracking de-activated, the solution converged in 4.11 seconds; the 0.01 difference was considered inconclusive. The numerical solutions with and without mode tracking were identical.

The solution based on Nelson's Method converged in only 2.65 seconds and likewise required 13 iterations. This time was reduced to 2.62 seconds with mode tracking disabled. Numerical results were identical with and without mode tracking activated.

Finally, both the modal expansion method and Nelson's method were tested alternatively with *MCORC* and *CORC* to determine if there were any differences. *CORC* required slightly more time than did *MCORC* with both sensitivity routines, but these difference were considered inconclusive due to the requirement to average CPU times. Superiority will have to be determined with a much larger finite element model.

The results cited above together with the final (optimized) frequencies are included in Appendix D.3 and D.4, respectively.

4.12 Mode Swapping Condition Analyzed

With FRAME equipped with an accurate method for determining eigenvector derivatives, the effect of mode tracking could now be analyzed. ***FRAME*** was configured to operate with mode tracking enabled and use eigenvector derivatives computed by Nelson's Method. Nelson's method was chosen over modal expansion as it was the faster and more flexible method.

Table 4-6 is a summary of the action the mode tracking logic took in optimizing ACOSS4. The table reports data on the number of mode swaps which were tracked by **both** MCORC and Gibson's CORC method, with only MCORC performing the permutations required. Critical data from the optimizer -- the number of steps the optimizer required along a search vector, and the size of the step (α) taken along that vector, is also included. If an iteration required more than one step, the results of each step are included but de-emphasized with smaller text and a dashed border. The total number of mode swaps detected by the independent mode trackers is reported in the corresponding row. The nature of the mode swap as detected by MCORC is listed in the column corresponding to that iteration. If the column of numbers under an iteration in not sequential, a mode swap was detected upon completion of that iteration. For example, if the element in the 7^{th} row (denoted as MODE 7) of column 3 is a *10*, this indicates that at the end of iteration 3, there was a mode swap -- Mode 10 is now more correlated with Mode 7 from the previous iteration. Finally, the table also reports the number of sign reversals (the number of eigenvectors scaled by a factor of -1 since the previous iteration). If a mode had a sign reversal since the last iteration, that element is shaded.

For example, in Iteration 2, there were no mode swaps detected by either mode swapping routine. However, there were a total of four sign reversals. These occurred in Modes 1, 4, 6, and 9. The optimizer required only one search, and that search provided the optimizer with the maximum step size of 1. One the other hand, the fourth iteration required two steps. There were a total of four mode swaps -- Modes 8 and 9 swapped places as did Modes 10 and 11, and six sign reversals. The optimizer took a step a size of only 0.4129 in proceeding to

Table 4-6: MCORC Mode Tracking History (with Nelson's Method)

Mode/IT#	0	1	2	3	3.1	4	4.1	5	5.1	6	7	8	9	10	11	12	13	#RevSin
MODE 1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	12
MODE 2	2	2	2	1	2	2	2	2	2	2	2	2	2	2	2	2	2	0
MODE 3	3	3	3	11	3	3	3	3	3	3	3	3	3	3	3	3	3	0
MODE 4	4	4	4	12	4	4	4	4	4	4	4	4	4	4	4	4	4	12
MODE 5	5	5	5	3	5	5	5	5	5	5	5	5	5	5	5	5	5	0
MODE 6	6	6	6	4	6	6	6	6	6	6	6	6	6	6	6	6	6	12
MODE 7	7	7	7	10	7	9	7	7	7	7	7	7	7	7	7	7	7	0
MODE 8	8	8	8	6	8	7	9	9	9	9	9	9	9	9	9	9	9	5
MODE 9	9	9	9	9	9	11	8	8	8	8	8	8	8	8	8	8	8	6
MODE 10	10	10	10	5	10	10	11	11	11	11	11	11	11	11	11	11	11	4
MODE 11	11	11	11	7	11	12	10	10	10	10	10	10	10	10	10	10	10	8
MODE 12	12	12	12	8	12	8	12	12	12	12	12	12	12	12	12	12	12	8
MCORC	0	0	0	11	0	5	4	4	4	4	4	4	4	4	4	4	4	
CORC	0	0	0	10	0	5	4	4	4	4	4	4	4	4	4	4	4	
RevSin	0	1	4	4	4	6	5	5	6	7	6	5	5	6	6	6	6	
# steps	1	1	1	-	2	-	2	-	2	1	1	1	1	1	1	1	1	
α	1	1	1	1	0.1	1	0.4139	1	0.2813	1	1	1	1	1	1	1	1	

Iteration 5. The following details the mode tracking history and the conclusions drawn from the subsequent analysis.

Iterations 1 & 2:

From the data in Appendix D.2, one sees that the modes in Iteration 1 are highly correlated with those of Iteration 0, as indicated by high correlation factors ranging from 0.85 (Mode 7) to 0.9999 (Mode 12). Likewise, the modes resulting from Iteration 2 are strongly correlated with those of Iteration 1, with the strongest association seen in the higher, uncontrolled modes. Note that Iteration 2 is the first of a series of sign reversal in Modes 1, 4, and 6 which occur in every subsequent iteration. No conclusion could be drawn to explain this behavior except that of pure randomness.

Iteration 3

Table 4-6 indicates this iteration resulted in 11 mode swaps. Upon closer examination, however, one sees that in taking the full (maximum) unit step, the optimizer produced a degenerate solution -- a solution with members having cross sectional areas of less than 0.01 units and 4 orders of magnitude smaller than their initial values. *MCORC* reported these as mode swaps in its attempt to blindly correlate one set of modes with a set that bore absolutely no resemblance to it. Thus *MCORC* failed when not properly constrained to small move limits. The optimizer then calculated a new search direction with a step size of only 0.1. Note that it was the optimizer which reset itself; it was not reset due to the mode swapping logic.

Iteration 4

Iteration 4 likewise required two steps, it's final step being only 0.4139. This second step was not required as the result of a degenerate solution (as in Iteration 3) but due instead to the normal activity of the optimizer. Iteration 4.1 resulted in four mode swaps. Because these swaps occurred in the higher modes, they did not affect the optimization. The optimizer still considered the solution viable and continued. In other words, *MCORC* did not detect the mode swap and thus force the optimizer to complete another search *due* to the mode swap -- it merely acted on the data it received from the optimizer. *MCORC* then permuted the resulting modal matrix. Extracting the applicable rows and columns from Appendix D.2, one sees *MCORC* did correctly detect the mode swap which actually occurred.

$$\boldsymbol{CORC}_{8:11x8:11} = \begin{array}{c|cccc} & \textit{8} & \textit{9} & \textit{10} & \textit{11} \\ \hline \textit{8} & 0.0578 & \mathbf{0.8879} & 0.0051 & 0.3679 \\ \textit{9} & \mathbf{-0.9131} & 0.1293 & 0.1940 & -0.1164 \\ \textit{10} & -0.0531 & -0.3214 & 0.6211 & \mathbf{0.6838} \\ \textit{11} & 0.2547 & 0.2292 & \mathbf{0.7530} & -0.5372 \end{array}$$

However, the mode swaps occurred in the higher, uncontrolled modes, and thus did not affect the optimizer. This was the first of a sequence of the same mode swaps which occurred between the same modes.

Iteration 5

The optimizer again backed off from taking a full step and found a step size of 0.2813 on its second search attempt. MCORC detected the same mode swaps (8-9, 10-11). This had the net effect of returning the modes to their original, sequential configuration.

Iteration 6

Iteration 6 required only one search and returned a unit step size. Note that in the previous three iterations, where the step size was less than the maximum, Figure 4-23 indicates the optimizer was *recovering*, having sunk below the true solution. This iteration marked the return to a more confident solution, with the optimizer taking full steps until final convergence. Note that the same mode swap occurred.

Iterations 7-13

Iterations 7 through 13 were non-spectacular. The optimizer slowly *tweaked* the solution from there on out. Note that at no time during these remaining seven iterations did the objective function differ from the optimum solution by more than 0.02%. The constraint forces occurring were equally non-significant with little interchange between constraint violations. The same mode swap occurred on every iteration, the net result being that the modes completed in the same order in which they originally had begun at Iteration 0. No conclusion could be drawn from this activity. The net result of the sign reversal logic was that only one mode, Mode 9, ended scaled by -1. Many sign reversals did occur, as noted by the

amount of shading evident in Table 4-6; however, each subsequent reversal negated the effects of the previous sign activity.

From these results, it was clear the mode swapping which did occur had no effect on the eventual solution. The sign reversal routine was equally insignificant. The latter result is due to the location of the mode swaps -- since all swaps occurred in the higher, uncontrolled modes, they did not corrupt the Hessian and thus did not affect SQP's (Section 3.5) ability to determine the best search vector. It was overwhelmingly evident that the original inability of the optimizer to converge to a feasible solution when equipped with analytic derivatives, was due to a logic error in the associated routine and that mode swapping played no part.

4.13 COFS Finite Element Model

4.13.1 Background

As with the ACOSS model, the first goal was to build an exact replica of the nominal COFS design found in the literature. This proved to be a somewhat daunting task as much of the most complete data sets were found in older literature published before the COFS model had fully evolved into what the community now defines as the "nominal" design. For example, Colladay's paper [Colladay, 1986] while very complete, used a *58*-bay COFS model, whereas Horta's model [Horta, 1986:517-23], while using the nominal 54-bay model, listed different mass properties for the cluster hinges located at alternate vertices of the 2-bay design. In addition, some of the data sets did not list the complete set of both material and physical

properties (which would have forced the use of data from two different references to fill in the gaps and thus lead to possible compromises in accuracy). Finally, some of the data did not report all significant digits, which might have led to rounding errors not acceptable in a model where natural frequencies differ from one another by only fractions of a hertz.

The most complete set of data was determined to be from Walsh [Walsh, 1987:7] because it could be cross-referenced with a hard-copy of the input file [Walsh, 1985] used in the finite element program EAL. This input file, the most valuable source, was unpublished and written in the native EAL syntax. The program EAL is no longer widely used in the structures community, nor could any published references to its syntax be found (not even in anthologies of modern finite element programs). Finally, the EAL database is not readily transportable to other finite element platforms. Prudence demanded all current data be compiled into one comprehensive document. In addition, a portable FEM of COFS, one that had an intuitive construction sequence and which could be transferred to a number of modern FEM platforms, was to be built.

The EAL program had been run in May 1987 by Horta, et al for a separate study. The corresponding input file was also available on disk but, as mentioned above, was not readily upconverted to any other finite element language. Two options by which to build the model were available. One option was to build a translator program. This program would read the EAL file into a dummy text file as ASCII text. It would then convert each EAL input line into the corresponding NASTRAN line, and write the new syntax into a NASTRAN ASCII file. This file could then be loaded into IDEAS. IDEAS could then convert it into an acceptable format.

The second option was to build the FEM model directly into IDEAS, just as had been done for ACOSS4. The latter option was chosen for two reasons. The first was the difficulty in obtaining an EAL reference document. Learning the syntax by intuition was not an option -- some of its syntax was even a mystery to an expert in commercial finite element programs. The second, primary, reason was that building the structure in IDEAS would serve mainly as a learning tool in mastering IDEAS. This experience would be critical to mode identification (as seen above with ACOSS) and in running subsequent optimizing schemes on COFS. By cross-referencing the two sources from Walsh, and using the papers by Hanks, Talcott, Horta, and Colladay as model checks, it was clear a model could be built to the required accuracy necessary for this type of analysis. The model construction scheme is included in Appendix A.3. The +70 page hardcopy file of the corresponding NASTRAN file was excluded due to its sheer bulk and limited utility in that form.

4.13.2 Finite Element Results

Several different configurations of the finite element model were examined in an attempt to attain as close a match as possible to those in the literature. Unfortunately, differences between the input file used in the original research, and the corresponding model configuration reported in that literature could not be reconciled. For example, the literature reported that plates were used to model sensor and actuator bays, and that actuator masses (it was assumed that the author was referring to both the primary and the secondary actuators), were set to zero. In addition, the model supposedly existed in a free-free condition. However, the input file showed all actuators to include their prescribed mass, the platforms to be

replaced with rigid bars, and the DRA's three connection points to be clamped. In addition, some changes had to be made in order to be compatible with the ***FRAME*** program, with which we hoped to eventually optimize COFS. The most noticeable difference was the removal of the orbiter stick model and the substitution of rigid elements with beam elements. (The surrogate members were provided with an unrealistically high modulus of elasticity and shear modulus in order to make them behave like a rigid element.)

Table 4-7: Finite Element Model Comparisons

	EAL FEM		IDEAS Model	
Mode	**Frequency (Hz)**	**Mode Shape**	**Frequency (Hz)**	**Mode Shape**
1	0.1888	1BY	0.1537334	1BY
2	0.2414	1BX	0.1613817	1BX
3	1.291	2BX	1.249438	1T
4	1.338	1T	1.374686	2BY
5	1.339	2BY	1.437559	2BX
6	3.686	3BX	1.738279	Orbiter Bending Mode
7	3.831	3BY	2.405682	Orbiter Rotation Mode
8	4.303	2T	4.059728	4BY
9	6.713	4BX	4.22252	4BX
10	6.946	4BY	4.783581	2T

The magnitudes of the frequencies in the two models are the within an acceptable tolerance given the construction differences mentioned above. In addition, the frequencies generally follow the same grouping and the two adjacent frequencies with the least separation in the EAL model are also extremely close in the IDEAS model. The most striking difference is the replacement of the 3rd bending modes in the IDEAS model with the defunct orbiter modes. Finally, note the mode rotations between the two models, corresponding to the different frequencies. This is a manifestation of the mode swapping condition from a *non-*

optimization perspective -- small differences in frequencies between models correspond to different mode shapes.

As this IDEAS model was incompatible with the ***FRAME*** program, a new model was built. The lower vertices of the DRA remained constrained, the orbiter stick model was removed, and the rigid elements replaced as noted above. The new frequencies were found to actually more closely mirror the results in literature, confirming speculation that the model in literature was not accurately reported. Again, the model produced values which were compatible with the main purpose of the model -- that of eventually optimizing the structure. Table 4-8 reports the natural frequencies and modes of the model intended for optimization.

Table 4-8: Finite Element Model Comparisons

	EAL FEM		IDEAS Model	
Mode	**Frequency (Hz)**	**Mode Shape**	**Frequency (Hz)**	**Mode Shape**
1	0.1888	1BY	0.1537336	1BY
2	0.2414	1BX	0.1613818	1BX
3	1.291	2BX	1.249422	1T
4	1.338	1T	1.374683	2BY
5	1.339	2BY	1.437555	2BX
6	3.686	3BX	4.059638	3BY
7	3.831	3BY	4.222431	3BX
8	4.303	2T	4.783452	2T
9	6.713	4BX	7.40866	4BY
10	6.946	4BY	7.66077	4BX

Again, there is some mode rotation between Modes 3, 4, and 5 due to their minimal separation. Consider for a moment that is a real-world deployed space system. One can readily see how differences of only hundredths of a hertz could lead to actuators trying to control, say the first torsional mode, when the structure has actually undergone second bending in the y-axis. This is the problem of mode swapping from a real-world perspective , as

originally introduced in Chapter 1, and the impetus for the COFS project -- to build a testbed with closely spaced frequencies to be used to check the fidelity of new control systems. Plots of the lower ten natural modes are included in Appendix H.

4.13.3 COFS Optimization

Transferring the COFS data deck from IDEAS to ***FRAME*** which would perform the optimization, proved to be much more difficult than originally foreseen. The process was tedious, and required the following steps:

1) Replace all rigid elements with beam elements. In addition, remove the orbiter, the model of which cannot be interpreted by ***FRAME***. Ensure the model still retains acceptable levels of equivalence with the nominal COFS design from literature;

2) Convert the IDEAS data deck into a NASTRAN file;

3) Replace the CBAR and PBAR cards with CROD and PROD cards and modify some data entries on the latter cards;

4) ***FRAME***'s numbering scheme is *inherent* -- i.e. Node 1 is the first node in the data deck, Node 2 the second, etc. Members are numbered the same way. To translate the IDEAS numbering scheme successfully into an input format compatible with ***FRAME***, the modified NASTRAN deck was run through CADS [Les, Manual, 1985], a translation program again provided by our sponsors at Wright Laboratories. CADS was to translate the 435 nodes into sequentially numbered grid points as required by ***FRAME.*** In addition, because CADS ignores lumped masses, the lumped masses after translation would lose their 1-1 correspondence with the respective nodes. Thus, the lumped masses had to first be converted into other elements (CONRODs were used) to force CADS to renumber them;

5) Replace any other element types CADS may have neglected in its translation;

6) Modify the resulting data deck to a final format compatible with ***FRAME.***

Steps 1-4 were completed. Unfortunately, CADS is limited to the number of nodes it can process and the COFS FEM far exceeded that limit. A modified version of CADS (with a larger limit) is currently being pursued by Wright Labs. Optimization, however, cannot be completed until that effort is completed.

V. Conclusion

The main goal of this research was to effect a successful optimization of an integrated structural and control design problem based on analytic derivatives. The first step was to determine if the existence of the mode crossing condition had prevented convergence of the integrated problem, ACOSS4. The problem had failed to converge when the corresponding optimization algorithm had been partially based on analytic derivatives of eigenvectors. It was believed that the solution failed because of the nonuniqueness of eigenvector gradients associated with at least two (nearly) repeated frequencies. The further possibility of the problem of frequency cross-over that accompanies the swapping of closely spaced modes was also suspected of exacerbating the failure to converge. The current literature on the problems associated with repeated frequencies and the mode crossing condition seemed to fit the description of the original problem.

While the primary goal of the thesis was accomplished, the original hypothesis proved to be incorrect. No mode swaps were found by visual or mathematical inspection of the ACOSS4 design history when the optimizer was equipped with analytical derivatives. A new mode characterization scheme for 3-dimensional structures was created where none had previously existed. All 12 nominal modes of ACOSS4 were characterized. In addition, this mode characterization scheme was applied to over 100 other modes from 14 distinct models and shown to be sound with little resulting ambiguity. The optimization history of the ACOSS4 was identified and recorded.

A mode swap was found (by visual inspection) to have occurred between Iterations 1 and 2 when the optimizer was equipped with eigenvector derivatives calculated by finite difference methods. Mode 3 (1st In-Plane Rocking) swapped its order with Mode 4 (2nd Out-Of-Plane Bending) when the 3rd frequency became larger than the 4th frequency upon completion of Iteration 2. This mode swap could not be confirmed quantitatively using automated mode swapping routines based on cross-orthogonality checks run in the nominal direction (checking Modes 1 through 12 in that order). The suspect mode-swapping routine originally used was CORC, as presented by Gibson. This method was shown to be highly dependent on order and therefore unreliable.

A new mode swapping routine, *MCORC* (for modified cross-orthogonality check), was created which initially appears to be more robust than the current cross-orthogonality checks in the literature, as it processes the CORC matrix in an order such that the strongest modes are correlated first. The method should be more efficient as well, as it depends on searching rather than sorting a matrix. MCORC did verify quantitatively the 3-4 mode swap which had been verified visually. In addition, MCORC was used to verify that a mode swap had occurred as a result of a unit step size between Iterations 2 and 3 during the run with *analytic* derivatives. This mode swap was also verified by visual inspection. The optimizer, however, rejected the unit step; its subsequent choice of step size did not result in a mode swap. It was determined that the optimizer backed off from the unit step size by its own volition -- not as a result of the mode swap.

The mode swap that had occurred in the finite difference run appears to be a result of the high sensitivity of the ACOSS4 problem. The finite difference solution required

60% more iterations than did a subsequent analytical routine and 10 times more CPU time. The finite difference solution was hampered by two unrelated phenomena. The first is the non-optimal design variable perturbation size used by the IMSL finite difference routine [IMSL, 1991: 1118]. The perturbation size is dependent on machine precision, and so sensitive that 3 additional iterations were required on a SPARC 20 than on the VAX due *only* to the differences in the platforms' respective precision. The second cause was the mode swap that did occur at the onset of the finite difference iteration history. The search direction appears to have been initially corrupted by the mode swap but eventually did recover. This second hypothesis is belied by the failure of a solution based on semi-analytic derivatives to converge when mode swapping was deactivated.

The solution based on semi-analytic derivatives was the only way to test the effect of mode tracking, as the proprietary optimizer did not allow access to its finite difference calculations. Unfortunately, small changes to the finite differencing were enough to prevent the mode swap from being observed again. Nevertheless, the semi-analytic approach was crucial in narrowing down the error that was corrected in the analytic derivatives.

A mode swap did **not** cause the original solution to fail to converge. Eigenvector sign reversal similarly had **no** effect. Rather, an error was found in the element sensitivity formulation in the portion of code comprising Nelson's Method for determination of eigenvector derivatives. The error was found when the correct implementation of modal expansion failed, forcing closer scrutiny of the two failures. The original coding of Nelson's method was corrected, thereby equipping ***FRAME*** with two analytic methods

for sensitivity determination. No differences between the solution based on Nelson's method and the solution based on modal expansion could be found to the 8th significant digit. The solution based on Nelson's method, however, was clearly more efficient than the solution based on modal expansion providing the identical results in half the time. Moreover, the achievement of the primary research goal -- the solution based on analytic derivatives -- was a *full order of magnitude more efficient* than the original finite difference solution.

The existence of the mode swapping condition in the COFS structure could not be verified due to the unexpected demands required when the original mode swapping hypothesis proved to be incorrect. However, a new finite element model of the COFS was built and an initial eigenanalysis shown to match the results in literature to the required degree of accuracy. Unlike previous finite element models of the structure, this new model, comprised of over 1900 individual components, is transportable to a number of software platforms, and has an intuitive, systematic numbering scheme.

In summary, the major accomplishments of this thesis are:

1) Implementation of a new analytical method (the primary goal) for eigenvector sensitivity analysis based on modal expansion,

2) Correction of the original optimization code based on Nelson's method, essentially equipping ***FRAME*** with two analytical methods for sensitivity analysis, where none had existed previously,

3) Development of a new, more robust mode tracking algorithm,

4) Determination that the mode swapping condition did not cause the failure of the original optimization problem,

5) Development of a new mode characterization algorithm for non-planar structures,

6) Creation of a finite element model of COFS along with completion of an initial eigenanalysis.

Recommendations

There are several recommendations for further study:

1) First and foremost, the nominal COFS model should be optimized using analytical methods and the mode-swapping condition investigated with the mode tracking code. If the mode swaps can be reproduced identically to those in the study by Walsh [Walsh, 1987: 5], their full impact can now be analyzed. More importantly, it can be determined if the mode swapping condition resulted in the original failure of the problem to converge to a feasible solution using only the nominal four design variables. Some work will be required to transform the NASTRAN deck into an input file which is compatible with ***FRAME***.

2) The analytical solution suffered from repetitive mode swaps which ultimately did not affect the solution as they were associated with uncontrolled modes. The mode tracking code could be modified to check for mode swaps amongst only the controlled modes. Such an enhancement may be critical in large models where an eigenanalysis may be too costly to perform on other than the set of controlled modes.

3) Eldred [Eldred, 1994:5] suggests CORC be used with corruption indices to determine the validity of the cross-orthogonality check. This corruption index is simply the ratio of the 2nd largest element in one column to the largest element of that column and is determined for every column of the modal matrix. If this ratio is greater than 0.5, Eldred suggests the check may be invalid. The code in Appendix F.1 can easily be equipped with such a validity check.

4) To determine if MCORC is truly superior to CORC, a different structure (one where mode swaps are known to occur) should be optimized using the two routines. Eldred [Eldred, 1994: 7] showed that CORC nearly failed when faced with a mode swap which had occurred during the optimization of an intermediate complexity wing (ICW) using ASTROS. A finite element model of the same ICW is available and needs only to be translated into an input file compatible with ***FRAME***.

5) The 489-element COFS model implemented by Grandhi [Grandhi and Venkayya, 1988: 18] was found to suffer from some oscillation due to repetitive mode swaps. This finite element model is available and likewise can be translated into a format compatible with ***FRAME***. Once the model is restructured, the mode tracking code can be used to determine if such an algorithm can better contain the oscillations.

6) It was conjectured that the additional iterations required in the finite difference solution was a result of two problems: 1) the non-optimum step size used by IMSL, and 2) the mode swap which occurred between iterations 1 and 2. Because the IMSL code is proprietary, further investigation into the mode swap hypothesis was not possible at the time. For those interested, it may be possible to either obtain a copy of that code or reproduce it, thereby providing the opportunity to include additional output lines which could provide insight into the true effect of the mode swap.

7) It is not clear how large the separation between eigenvalues can be before they are considered repeated. As mentioned in Chapter 4, Bernard and Bronowicki suggest it may be as large as 5%. Investigating the maximum separation may be useful to the structures community for working with the integrated structural design and control problem.

Appendix A
Finite Element Models

A.1 ACOSS4

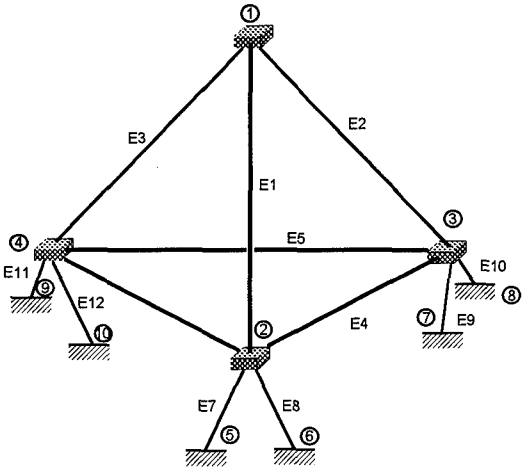


Figure A-1: ACOSS4

The basic elements of the ACOSS4 structure were two types of rods. The upper half of the structure is a tetrahedron. The base of the tetrahedron is an equilateral triangle comprised of rods having an initial cross-sectional area one order of magnitude larger than that of the smaller elements. Two of the uprights of the tetrahedron have the smaller cross sectional areas, and the third element has the larger area. All units are dimensionless and consistent. Six bipods comprise the lower half of the structure with two rods supporting the base at each of the three vertices. The bipods are likewise modeled as rods and all initially have the smaller cross sectional area.

The element types correspond to the element numbering scheme (i.e. Element 5 is "Beam" Type 5 in IDEAS). See Table A.1-2. All twelve elements are rods, are made of generic isotropic steel, and have a modulus of elasticity of 1. In addition, each of the twelve rods is afforded its own element type to expedite independent design iterations and the magnitudes referenced are initial values only.

It was thought that the small size of the model would eliminate the need to designate an independent color for each of the design variables as was done in modeling COFS.

Thus, the larger rods were all designated green and the eight smaller rods were designated as yellow. However some ambiguity could have been avoided if all the uprights had been different colors. In addition, if Elements 8, 10 and 11 had been given a different color than their counterpart bipods (Elements 7, 9 and 12, respectively), some confusion could have been avoided in determining plane designations.

The six bipod legs are clamped as is the apex. Each of the four vertices includes a lumped mass of 20000 mass units, modeled in IDEAS as blue.

Table A-1: Node Geometry

Node	X	Y	Z	Lumped Mass
1	0.0	0.0	10.165	$2.0*10^6$
2	-5.0	-2.8870	2.0	$2.0*10^6$
3	5.0	-2.8870	2.0	$2.0*10^6$
4	0.0	5.7735	2.0	$2.0*10^6$
5	-6.0	-1.1547	0.0	0.0
6	-4.0	-4.6188	0.0	0.0
7	4.0	-4.6188	0.0	0.0
8	6.0	-1.1547	0.0	0.0
9	-2.0	5.7735	0.0	0.0
10	2.0	5.7735	0.0	0.0

Table A-2: Member Geometry/Connections

Member	From Node	To Node	Area
1	1	2	10
2	2	3	10
3	2	4	10
4	3	4	10
5	1	3	1
6	1	4	1
7	5	2	1
8	6	2	1
9	7	3	1
10	8	3	1
11	9	4	1
12	10	4	1

A.2 ACOSS4 NASTRAN Bulk Data

```
BEGIN BULK
GRID       1      0 0.00000 0.00000 10.1650      0
GRID       2      0-5.00000-2.88700 2.00000      0
GRID       3      0 5.00000-2.88700 2.00000      0
GRID       4      0 0.00000 5.77350 2.00000      0
GRID       5      0-6.00000-1.15470 0.00000      0
GRID       6      0-4.00000-4.61880 0.00000      0
GRID       7      0 4.00000-4.61880 0.00000      0
GRID       8      0 6.00000-1.15470 0.00000      0
GRID       9      0-2.00000 5.77350 0.00000      0
GRID      10      0 2.00000 5.77350 0.00000      0
SEQGP      1      7      2      8      3      3      4      1
SEQGP      5      9      6     10      7      4      8      6
SEQGP      9      2     10      5
CROD       1      1      1      2
CROD       2      1      2      3
CROD       3      1      2      4
CROD       4      1      3      4
CROD       5      5      1      3
CROD       6      5      4      1
CROD       7      5      5      2
CROD       8      5      6      2
CROD       9      5      3      7
CROD      10      5      3      8
CROD      11      5      9      4
CROD      12      5      4     10
CONM2     13      3      0 2.00E+6 0.00000 0.00000 0.00000      +EA   13
+EA   13 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
CONM2     14      2      0 2.00E+6 0.00000 0.00000 0.00000      +EA   14
+EA   14 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
CONM2     15      4      0 2.00E+6 0.00000 0.00000 0.00000      +EA   15
+EA   15 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
CONM2     16      1      0 2.00E+6 0.00000 0.00000 0.00000      +EA   16
+EA   16 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
MAT1       2 1.00E+8 1.00000 0.29000 0.10000 6.50E-6          +MA    2
+MA    2 100000. 100000. 9862.56
PROD       1      2 10.0000 2.40E+6 0.00000 0.00000
PROD       5      2 1.00000 2.40E+6 0.00000 0.00000
PARAM   AUTOSPC YES
PARAM   POST   -2
ENDDATA
```

A.3 COFS Finite Element Model

The basic building block of the model is the two-bay structure. The bottom bay (i.e. the bay closest to the shuttle) is labeled the 'A' bay, and the top bay the 'B' bay. While many of the parts in the B-bay are the same as those in the A-bay (e.g. longerons, cluster hinges, etc.), it was necessary to keep all the elements in the bay distinct to maintain the numbering scheme with respect for those elements present in only one of the bays -- thus necessitating the 'A' and 'B' bay designations.

Once the 2-bay element was completed, it was simply replicated 27 times. To ensure this replication was ordered, each node and each element in each bay were given a separate identifier. The identifiers are spaced at intervals of 100. This allows us to simply set an interval of one when replicating the 2-bay structure, and all corresponding elements will be one digit apart, with no cross-overs. For example, all strong longerons in 'A' bays are numbered between 101 and 127. There are no other elements in this first "century" so there are no cross-overs. Each subset of elements is therefore in its own century. Take the cluster hinges on the B-bay batten plane as an example. They are numbered between 2801 and 2827 (2801, 2802, 2803 . . .2827). The first two digits, '25' designate that element as a cluster hinge on the 'B' batten planes. The last two digits, say '27' designate this element as the hinge on the 27th B-bay batten plane.

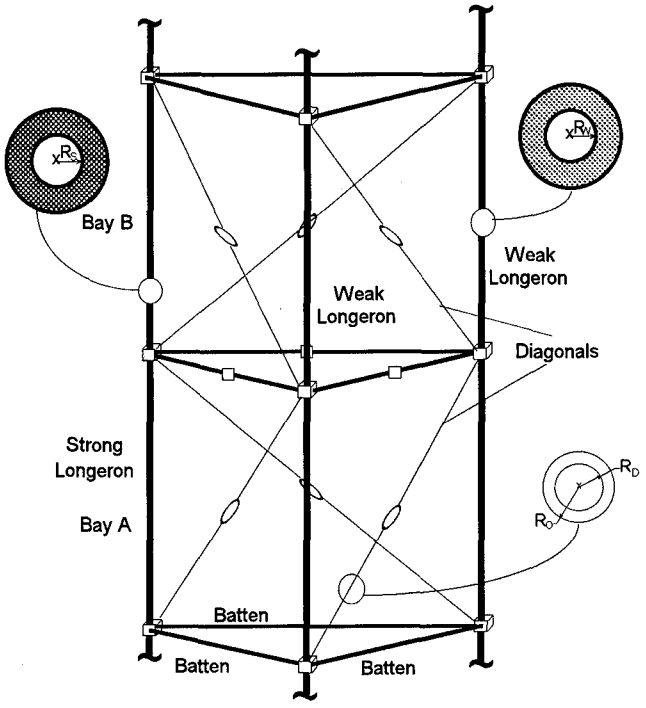


Figure A-2: COFS 2-Bay Element

Now, take the first four bays on the bottom of the COFS structure. The first strong longeron is in the 'A' bay, and has an identifier of 101. The strong longeron in the 'B' bay is identified as 401. The structure then repeats. Bay 3 is an A-bay and has a strong longeron numbered 102, while Bay 4's strong longeron is 402. Following this pattern, Bay 53's strong longeron is simply 127, and the top-most bay, Bay 54, has a strong longeron labeled as 427. Again, each of the 48 elements in the basic 2-bay structure is in its own century. Also note that many of the elements appear in subsets of 3; this pattern is a direct result of COFS triangular cross-section.

This numbering scheme presented two anomalies which we will mention for completeness. First, note that we did not start the elements at an even interval of 100, but at an [even interval + 1], i.e. 101. This was done since those last two digits designate the sequence number, and we did not want there to be a 'zeroth' (0th) element. The second anomaly is a result of the COFS being a 'closed' structure -- that is, there are battens running along the very bottom, as well as battens running across the very top. Therefore, there is one more set of battens for the A bays than the B-bays, even though the structure has the same number of A-bays as it does B-bays.

Also note, one error was made in designating centuries necessitating the elimination of centuries 1000, 1100, and 1200. This was done to keep all the battens in the 'B' bays together. Finally, the *node* numbering scheme and the *element* numbering scheme share some of the same numerical designations. This presented no problems in replicating the structure, since we replicated nodes first, and then elements, but it can be somewhat confusing if one does not acknowledge the alpha-designator ('N' for node, 'E" for element) when perusing the computer model.

The other COFS elements -- the actuators, sensors, and PMD masses, were added after the model was replicated. The stiff panels, on which the actuators and sensors are mounted were not included as plates as the literature suggested, but instead the sensors and actuators were attached using rigid beam elements. The sensor masses were small enough that they could be included as lumped masses at the vertices of the structure, while the actuators and PMD masses required additional nodes on which to attach the lumped masses. Actuator nodes were attached at the center of the bay. Cabling also had to be included. Like the sensor masses, the cabling was of such small mass that it could be distributed at each vertex as lumped masses. Having multiple lumped masses at one node presents one difficulty in IDEAS -- one can only see the last lumped mass added although in terms of the model, all other masses are still present.

We created the basic **node** structure by studying the input file [Walsh: 1985] and identifying sets of triangular patterns. One glaring inconsistency in most of the literature is

Table A-3: COFS Nodal Geometry

NODE	X	Y	Z
	Main	**Vertices**	
101	24.6140	0.0	11.952
201	23.5640	0.606	11.952
301	23.5640	-0.606	11.952
	Diagonal	**Midspan**	**Points**
401	23.5640	0.0	12.514
501	24.0890	0.303	12.514
601	24.0890	-0.303	12.514
	Batten	**Midspan**	**Points**
701	24.0890	0.0	13.076
801	23.5640	0.303	13.076
901	24.0890	-.303	13.076
	Actuator	**Nodes**	
1001	23.9620	0.0	25.440
1101	23.9620	0.0	45.672
1201	23.9620	0.0	61.408
	LDCM	**Nodes**	
1301	23.9140	0.4600	72.848
1401	24.3470	0.0	72.848
1501	23.9140	-0.4600	72.848
1601	23.4540	0.0	72.848
	PMD	**Nodes**	
1701	23.2140	0.000	72.848
1801	23.9140	-0.700	72.848
1901	23.3895	0.175	72.848
2001	24.0890	-0.525	72.848
	Tip	**Mass**	
2101	23.914	0.0	72.848

the orientation of the diagonals in the 'A' and 'B' bays. None of the literature is consistent, and some documents actually contradict themselves. To determine if the diagonals in the A-bay run up-from-left-to-right or up-from-right-to-left (as viewed from the front of the any of the A-bays), we graphed some of the data points from the input file referenced above. We determined the orientation was that of the former, that is the diagonals run up-from-left-to-right in the A-bays (and therefore up right-to-left in the B-bays.)

Three nodes were required for the basic cross section (one node at each vertex), and replicated at a Z-station 1.124 meters (the height of one bay element) above the first set. Three nodes were required to identify the hinge location at the midspan of each diagonal, and three nodes were required to identify the midpoint of each mid-span batten, which is also hinged. The latter nodes were incremented by 2 to maintain the numbering scheme necessary for simple replication. These 9 nodes were then replicated 27 times at a constant interval of 1.124 meters in the 'z' direction. Node assignments are included in Table A-3 where the coordinates correspond to a right hand coordinate system, +x is parallel to the shuttle's roll axis, +z is

up and node locations are all in meters. (Note, the diagonals and Batten B elements are not continuous members, being connected through a hinge. Because the hinge is a lumped mass, and because a lumped mass requires a node, the diagonals and battens were modeled as two separate beam elements.)

All structural elements were modeled as beams with hollow, circular cross-sections. We required 4 such beam elements, defined in the parameter table as 'STRONG LONGERON', 'WEAK LONGERON', 'DIAGONAL' and 'BATTEN'. Recall, the strong longeron and weak longerons have diameters corresponding to optimization parameters (R_s and R_w respectively), while the diagonal's inner and outer diameters (R_i and R_o) are both optimization parameters. Because the diagonals are connected with locked hinges at midspan, each diagonal is modeled as two symmetric tubes connected at midspan. The elements are coded for ease of reference.

Table A-4: COFS Element Properties

ELEMENT	Outer Diameter (m)	Inner Diameter (m)	Thickness (m)	Area (m^2)	Elasticity Modulus (N/m^2*e10)	Shear Modulus (N/m^2*e9)	Color
Strong Long.	0.0220	0.0104	0.0116	0.03644	13.25	49.700	red
Weak Long.	0.0229	0.0147	0.0082	0.02580	13.88	52.063	pink
Batten	0.0140	0.0114	0.0026	0.00819	12.80	48.012	dk. green
Diagonal	0.0191	0.0179	0.0012	0.00377	11.07	41.523	white
Rigid Beams	0.00208	-	-	-	1000	30000	lt. blue
DRA	0.00208	-	-	-	-	-	yellow

The battens at mid-bay (defined as Batten B) are also hinged and therefore modeled as two identical beams connected at mid-span. Diagonals are identified in Table A-6 using a 2-digit mnemonic. The first digit identifies the face and the second the half of the beam element of the 2-beam sequence (e.g. Diagonal A12 is the upper half of the diagonal on

the first face of Bay A. Cluster hinges, batten hinges, and diagonal hinges are modeled as lumped masses with appropriate magnitudes.

Once the truss structure was replicated, distributed cabling and sensor masses were included as lumped masses. Nodes for the actuators at Bays 12,30, 44 and the TRS were added, and lumped masses placed there. Nodes were added for the PMD as well, and nodes placed corresponding to the four 20-kg masses being in their neutral positions. Elements for the sensors, actuators, and PMD masses are numbered so that the third and fourth digits correspond to the actual bay at which they are located. For example, taking element 10242, the '10' identifies it as a sensor mass, the '24' identifies it as the sensor mass at Bay 24, and the 2 identifies it as the mass at the vertex of the second longeron. Each distinct element was color-coded for ease of identification in later iterations. All data is included in Table A-6.

The Deployment/Retraction Assembly was modeled as a truss comprised of elements as indicated in Table A-4. The 84 individual masses comprising the gearing and machinery were modeled as lumped masses and placed at the six vertices. (See Fig. A-4). The lower vertices were clamped, and are the only boundary conditions used in the optimization model. The DRA is connected to the midpoint of the orbiter cargo bay by rigid members.

Table A-5: Lumped Masses

Name	Mass (kg)	Color
Cluster Hinge A	0.907200	Magenta
Cluster Hinge B	0.907200	Lt Magenta
Batten Hinge	0.520000	Blue
Diagonal Hinge	0.300000	Yellow
Cable Mass	0.072727	Orange
Sensor Mass	3.430000	Gold Orange
Actuator Mass	18.80000	Red
Tip Mass*	72.900000 6.45 6.45 12.9	Yellow
PMD	20.000000	Magenta
Fixed LDCMs	8.256000	Dark Blue
DRA Elements	see bulk	Magenta
Orbiter*	92388.6 1.203e+6 9.190e+6 9.580e+6	White
Beam Elements	0	Lt Blue

Only the Orbiter and PMD have inertia properties. The values listed columnwise in Table A-5 are listed as I_{xx}, I_{yy}, and I_{zz}. Because the corresponding NASTRAN input deck was over 70 pages in length, it has been excluded from the appendix.

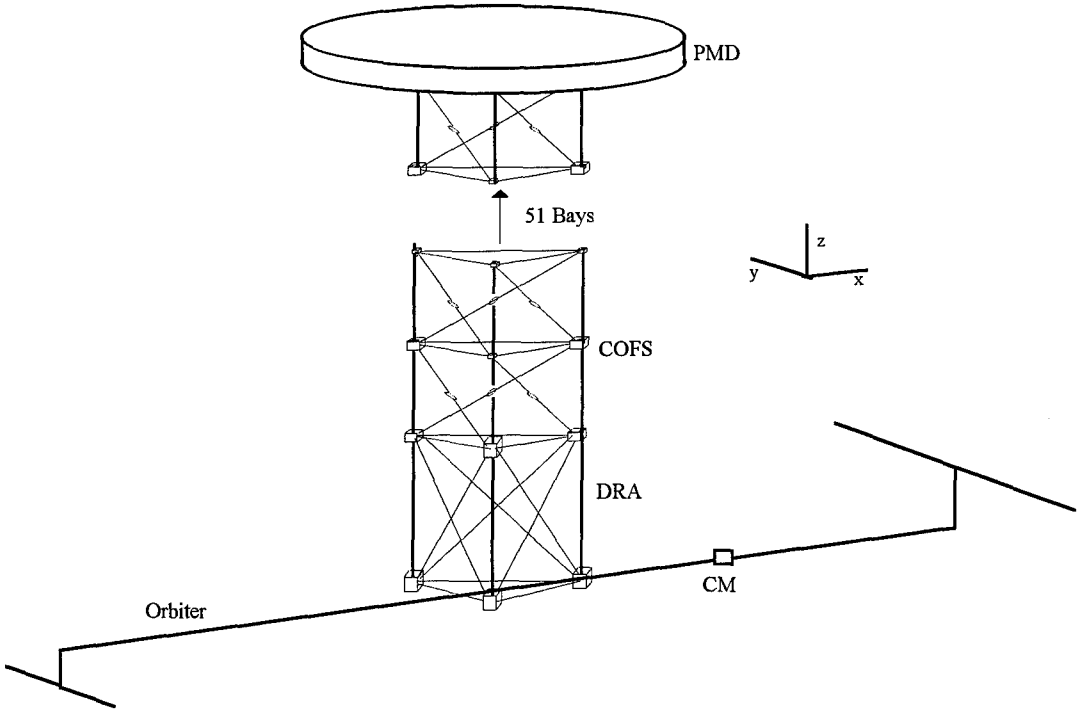


Figure A-3: Complete Finite Element Model

Table A-6: Element Identifiers

Element	Label	Init #	Tot
Strong Long. A	L1	101	27
1st Weak Long. A	L2	201	27
2nd Weak Long. A	L3	301	27
Strong Long. B	L4	401	27
1st Weak Long. B	L5	501	27
2nd Weak Long. B	L6	601	27
Batten A1	BA1	701	28
Batten A2	BA2	801	28
Batten A3	BA3	901*	28
Diagonal A11	DA11	1301	27
Diagonal A12	DA12	1401	27
Diagonal A21	DA21	1501	27
Diagonal A22	DA22	1601	27
Diagonal A32	DA31	1701	27
Diagonal A33	DA32	1801	27
Diagonal B11	DB11	1901	27
Diagonal B12	DB12	2001	27
Diagonal B21	DB21	2101	27
Diagonal B22	DB22	2201	27
Diagonal B32	DB31	2301	27
Diagonal B33	DB32	2401	27
Cluster Hinge A1	CHA1	2501	28
Cluster Hinge A2	CHA2	2601	28
Cluster Hinge A3	CHA3	2701	28
Cluster Hinge B1	CHB1	2801	27
Cluster Hinge B2	CHB2	2901	27
Cluster Hinge B3	CHB3	3001	27
Diag. Hinge A1	DHA1	3101	27
Diag. Hinge A2	DHA2	3201	27
Diag. Hinge A3	DHA3	3301	27
Diag. Hinge B1	DHB1	3401	27
Diag. Hinge B2	DHB2	3501	27
Diag. Hinge B3	DHB3	3601	27
Batten Hinge B1	BHB1	3701	27
Batten Hinge B2	BHB2	3801	27
Batten Hinge B3	BHB3	3901	27
Batten B11	BB11	4001	27
Batten B12	BB12	4101	27
Batten B21	BB21	4201	27
Batten B22	BB22	4301	27
Batten B31	BB31	4401	27
Batten B32	BB32	4501	27
Cable Mass A1	CMA1	4601	28
Cable Mass A2	CMA2	4701	28
Cable Mass A3	CMA3	4801	28
Cable Mass B1	CMB1	4901	27
Cable Mass B2	CMB2	5001	27
Cable Mass B3	CMB3	5101	27

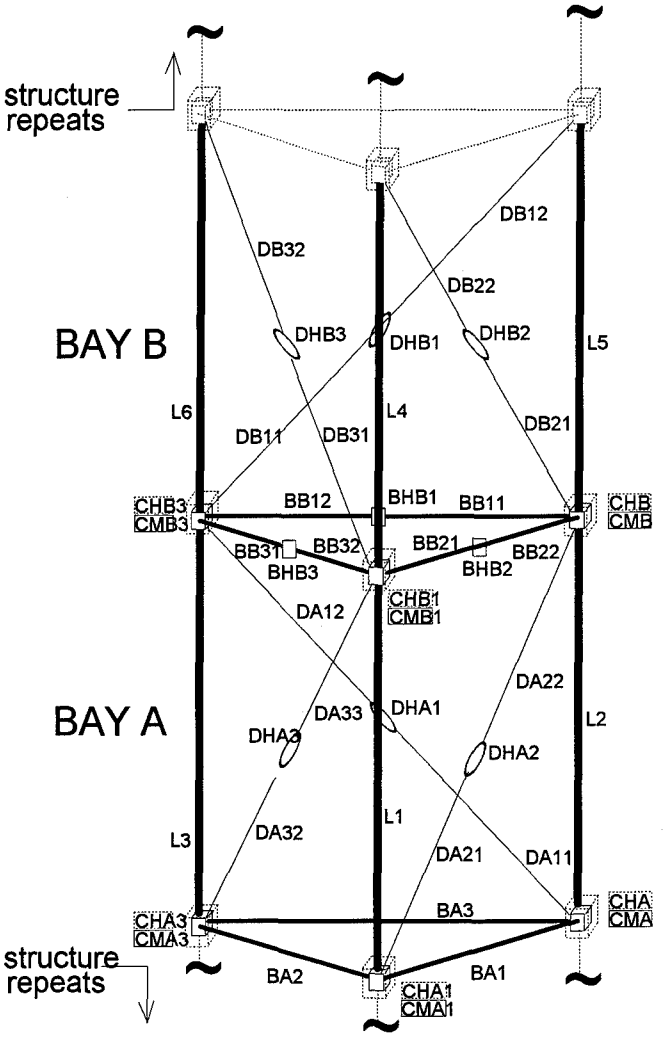


Figure A-4: Element Identifiers

Table A-5: Element Identifiers (Cont.)

Element	Label	Init #	Tot
Sensor Mass 24-1	SM241	10241	1
Sensor Mass 24-2	SM242	10242	1
Sensor Mass 24-3	SM243	10243	1
Sensor Mass 38-1	SM381	11381	1
Sensor Mass 38-2	SM382	11382	1
Sensor Mass 38-3	SM383	11383	1
Actuator Mass 12	AM121	12121	3
Actuator Mass 30	AM301	13301	3
Actuator Mass 44	AM441	14441	3
Rigid Elements	RE	80000	20
Par. Mod. Device	PMD	20001	4
Tip Mass	TP	54001	4
Orbiter	O	200000	15
DRA	D	400000	87

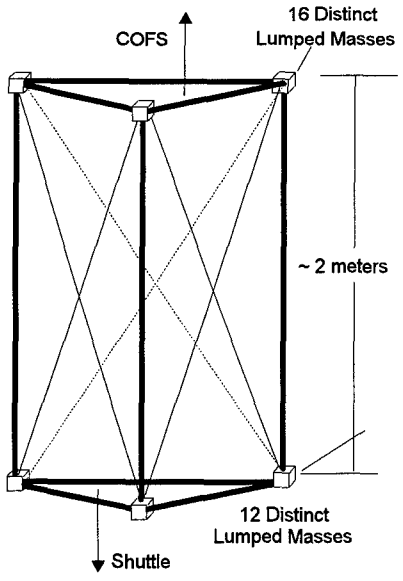


Figure A-5: DRA

A.4 COFS NASTRAN DECK

The +60 page NASTRAN deck has been excluded in the interest of space.

Appendix B

Modal Deformations of ACOSS4 FEM

DESIGN: Nominal

Derivatives Used: Analytic

Mode 8: 1st Shearing (In-Plane)

Mode 9: 1st In-Plane Compression In-Phase w/1st Axial

Mode 10: 1st In-Plane Compression In-Phase w/1st Torsion

Mode 11: 1st Extension (In-Plane)

Mode 12: 1st Breathing

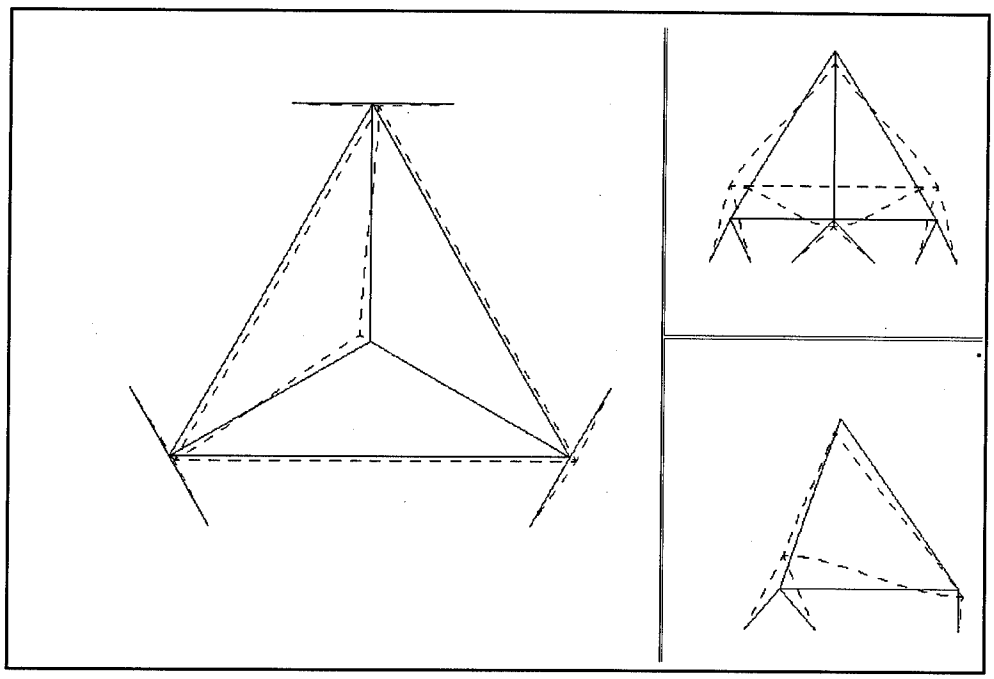


Figure B-1: Nominal/Analytic Mode 8 (1st In-Plane Shearing)

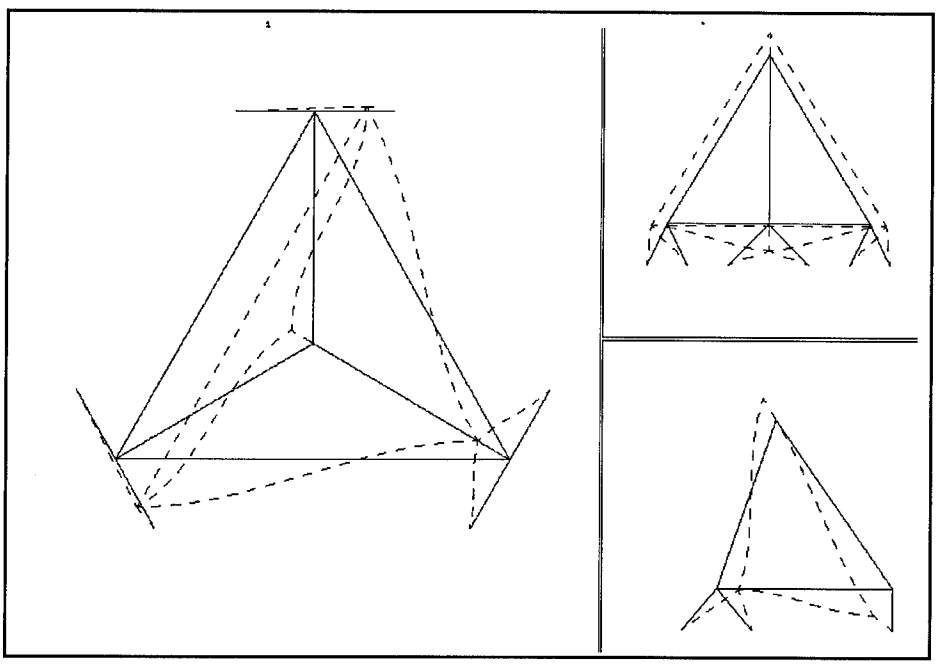


Figure B-2: Nominal/Analytic Mode 9

(1st In-Plane Compression In-Phase w/1st Axial)

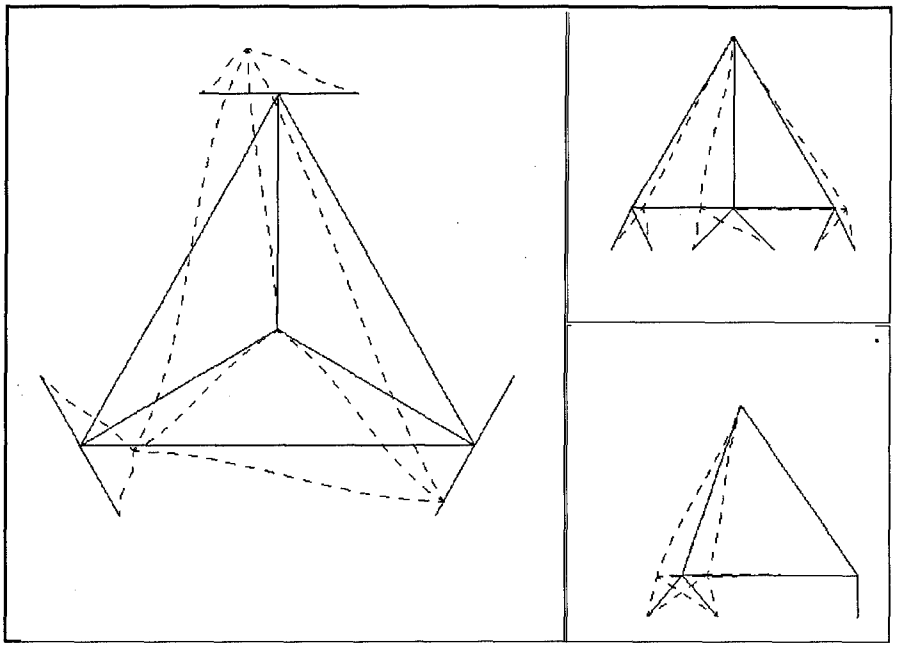


Figure B-3: Nominal/Analytic Mode 10

(1st In-Plane Compression In-Phase w/1st Torsion)

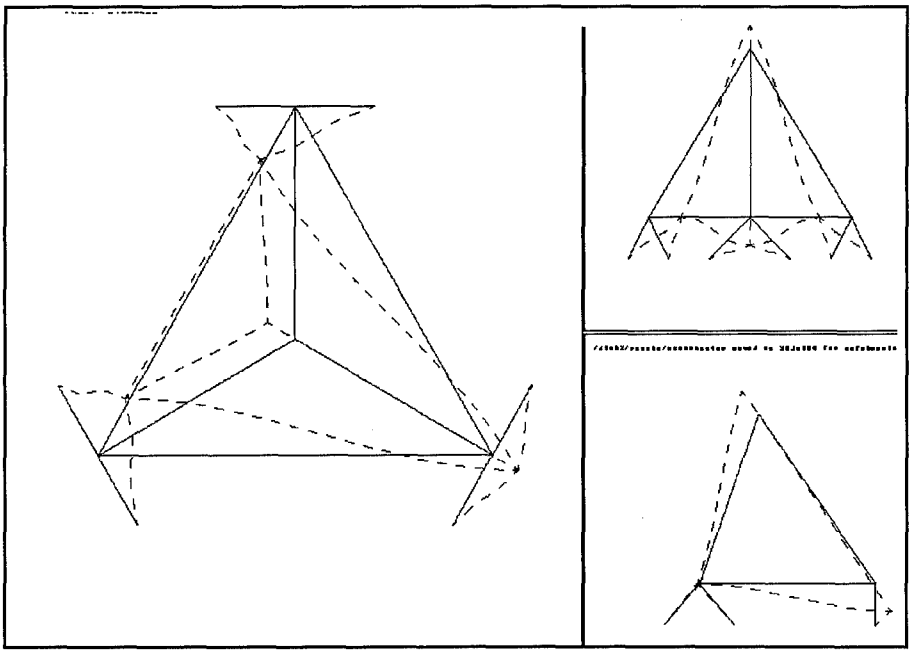


Figure B-4: Nominal/Analytic Mode 11 (1st In-Plane Extension)

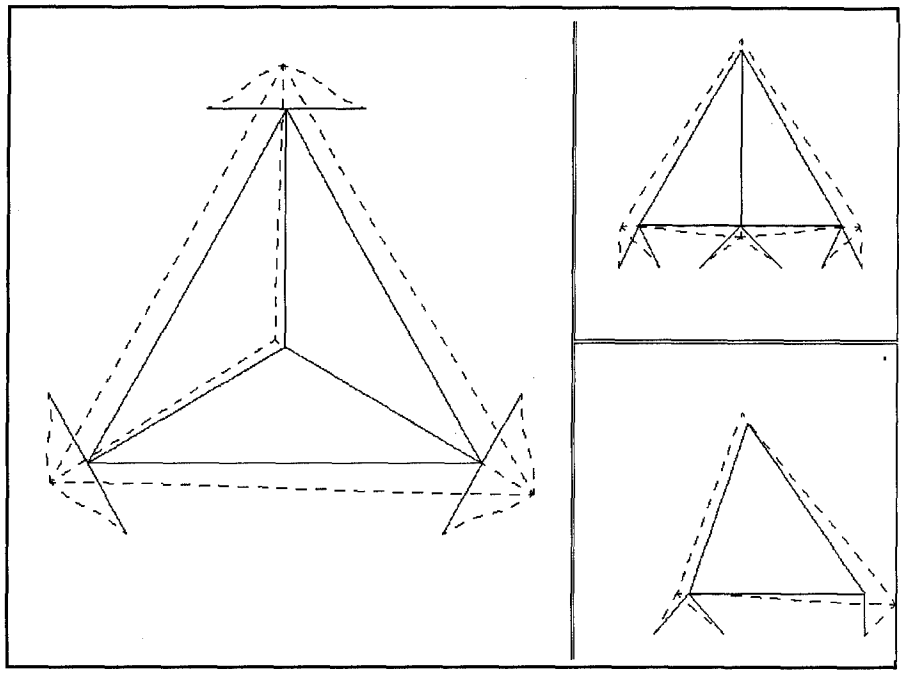


Figure B-5: Nominal/Analytic Mode 12 (1st Breathing)

Appendix C

The following tables show the results of the sensitivity analyses for iteration 0 for all six controlled modes using four methods of sensitivity analysis. Results from subsequent iterations have been excluded in the interest of space. The mode derivatives from the original formulation using Nelson's Method are also included for completeness. A separate set of tables for the semi-analytic derivatives is not included as the derivatives are the same as those in the finite difference table. The tables show the derivatives of the modes with respect to the 12 design variables horizontally across the table; while the derivatives with respect to the 12 degrees of freedom read down the table. For example, the element in row 3, column 5 of Mode 1 represents the partial derivative of Mode 1 with respect to the fifth design variable (the cross sectional area of Element 5) with respect to the third degree of freedom, or,

$$\frac{\partial \phi_{31}}{\partial X_5}.$$

In addition, the first four columns of the matrix of ASTROS eigenvector derivatives were scaled by a factor of 0.1 to account for scaling done in the formulation of the original program by ASTROS.

C.1: Mode Derivatives as computed from Original Formulation of Nelson's Method

C.2: Mode Derivatives as computed from Finite Difference Formulation

C.3: Mode Derivatives as computed from ASTROS (Nelson's Method)

C.4: Mode Derivatives as computed from Modal Expansion

C.5: Mode Derivatives as computed from Corrected Formulation of Nelson's Method

Table C-1

Eigenvector Derivatives from Original Nelson

MODE 1

DOF	$\partial M_1/\partial DV_1$	$\partial M_1/\partial DV_2$	$\partial M_1/\partial DV_3$	$\partial M_1/\partial DV_4$	$\partial M_1/\partial DV_5$	$\partial M_1/\partial DV_6$	$\partial M_1/\partial DV_7$	$\partial M_1/\partial DV_8$	$\partial M_1/\partial DV_9$	$\partial M_1/\partial DV_{10}$	$\partial M_1/\partial DV_{11}$	$\partial M_1/\partial DV_{12}$
1	3.718e-10	-1.545e-06	2.884e-07	3.730e-10	8.616e-05	-2.604e-04	2.091e-05	-4.441e-05	-3.410e-05	2.131e-05	-1.645e-05	-2.636e-05
2	1.210e-10	-7.763e-07	2.823e-07	2.097e-10	3.981e-05	-1.603e-04	1.581e-05	-2.191e-05	-1.589e-05	1.165e-05	-5.695e-06	-1.588e-05
3	-8.398e-15	2.363e-07	1.742e-07	-1.087e-10	-6.491e-05	1.298e-04	3.599e-06	3.599e-06	-2.928e-07	-7.686e-06	1.708e-05	9.680e-06
4	4.728e-11	-8.133e-07	5.386e-07	1.613e-10	-5.663e-06	-4.967e-05	1.037e-05	1.975e-06	-2.370e-05	7.430e-06	1.009e-05	-9.511e-06
5	1.634e-11	-4.388e-07	3.418e-07	9.512e-11	1.530e-05	-1.011e-05	-1.329e-05	-1.813e-05	-9.684e-06	5.304e-06	9.827e-06	-4.478e-06
6	4.091e-11	-2.241e-07	6.834e-13	4.167e-11	-9.128e-09	-3.545e-05	1.936e-05	-2.329e-05	-6.001e-06	2.071e-06	-3.162e-06	-3.160e-06
7	4.399e-11	2.174e-07	5.217e-07	1.514e-10	-1.006e-05	-4.842e-05	9.943e-06	2.799e-06	-2.330e-05	6.717e-06	9.318e-06	-9.139e-06
8	2.612e-12	-3.580e-07	-2.401e-07	-2.771e-11	1.716e-05	3.201e-06	-3.763e-06	-6.443e-06	-2.215e-05	9.856e-06	-4.530e-06	2.098e-06
9	-2.900e-11	2.155e-07	9.996e-08	-2.642e-11	8.310e-06	3.707e-05	8.861e-07	4.918e-06	-1.981e-05	-1.024e-05	6.157e-06	2.470e-06
10	3.550e-11	-2.280e-07	2.628e-08	4.821e-11	-6.257e-06	-3.752e-05	1.111e-06	-4.780e-06	-7.891e-06	1.901e-06	2.367e-05	-1.274e-05
11	1.232e-11	-7.168e-07	-5.124e-07	1.503e-10	2.060e-05	-5.657e-06	-8.192e-06	-1.303e-05	-1.626e-05	8.263e-06	3.174e-06	-1.595e-06
12	-1.191e-11	3.539e-08	-8.020e-08	-2.757e-11	-1.566e-05	1.311e-05	-2.548e-06	1.487e-06	-6.244e-07	-1.812e-06	2.534e-05	1.090e-05

MODE 2

DOF	$\partial M_2/\partial DV_1$	$\partial M_2/\partial DV_2$	$\partial M_2/\partial DV_3$	$\partial M_2/\partial DV_4$	$\partial M_2/\partial DV_5$	$\partial M_2/\partial DV_6$	$\partial M_2/\partial DV_7$	$\partial M_2/\partial DV_8$	$\partial M_2/\partial DV_9$	$\partial M_2/\partial DV_{10}$	$\partial M_2/\partial DV_{11}$	$\partial M_2/\partial DV_{12}$
1	7.870e-07	-3.329e-07	9.291e-07	1.729e-06	1.664e-04	-4.161e-05	3.556e-05	-8.563e-06	-5.771e-06	7.204e-05	9.269e-06	4.295e-05
2	-2.826e-06	5.575e-07	-1.628e-06	-2.670e-06	-3.000e-04	6.020e-05	-7.390e-05	2.509e-06	1.148e-05	-1.063e-04	-1.456e-05	-5.598e-05
3	-1.414e-06	-2.951e-07	-2.950e-07	-4.418e-07	3.796e-05	3.797e-05	-1.986e-05	-1.987e-05	-3.908e-06	-4.985e-06	-3.906e-06	-4.982e-06
4	5.128e-07	-4.087e-07	-3.358e-07	-6.001e-07	4.805e-05	1.322e-05	-5.677e-06	2.070e-05	-5.047e-06	-2.202e-06	-3.113e-06	-1.100e-05
5	-4.930e-08	-2.264e-07	-3.524e-07	-7.328e-07	-2.889e-05	3.143e-05	2.260e-05	-2.309e-05	-9.519e-07	-2.347e-05	-4.296e-06	-8.233e-06
6	-6.071e-09	2.523e-08	2.527e-08	2.354e-07	1.129e-05	1.129e-05	-2.845e-05	-2.846e-05	5.096e-08	8.616e-06	5.162e-08	8.614e-06
7	5.780e-07	4.254e-07	-2.854e-07	-5.133e-07	5.990e-05	4.298e-06	-4.146e-06	2.071e-05	-5.413e-06	2.524e-06	-2.328e-06	-1.152e-05
8	-1.734e-07	-2.242e-07	1.022e-07	2.756e-07	-4.223e-05	3.009e-05	3.643e-06	-7.772e-06	-4.526e-06	-5.610e-05	-4.420e-07	1.633e-05
9	-4.999e-07	7.132e-08	-3.117e-07	-5.743e-07	-5.136e-05	1.179e-05	-1.371e-05	-3.125e-07	-5.406e-06	2.476e-05	-2.960e-06	-1.331e-05
10	4.379e-07	1.029e-08	8.312e-08	3.165e-07	4.242e-05	7.593e-06	7.592e-06	5.047e-06	-1.315e-06	1.881e-05	-6.390e-06	-3.689e-05
11	-1.367e-07	-4.545e-07	3.244e-07	-1.392e-06	-4.574e-05	3.857e-05	1.223e-05	-1.501e-05	-2.360e-06	-4.338e-05	-2.987e-06	4.994e-06
12	4.234e-08	-1.948e-07	1.883e-07	3.210e-08	3.757e-05	-2.558e-05	6.879e-06	-6.516e-06	-2.539e-06	5.594e-06	-4.981e-06	4.367e-05

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-1 (Cont.)
Eigenvector Derivatives from Original Nelson

MODE 3

DOF	$\partial M_3/\partial DV_1$	$\partial M_3/\partial DV_2$	$\partial M_3/\partial DV_3$	$\partial M_3/\partial DV_4$	$\partial M_3/\partial DV_5$	$\partial M_3/\partial DV_6$	$\partial M_3/\partial DV_7$	$\partial M_3/\partial DV_8$	$\partial M_3/\partial DV_9$	$\partial M_3/\partial DV_{10}$	$\partial M_3/\partial DV_{11}$	$\partial M_3/\partial DV_{12}$
1	-2.636e-07	-2.517e-06	1.575e-06	1.021e-07	-5.669e-05	-1.144e-04	2.260e-05	-4.098e-05	-1.722e-04	-1.704e-05	-3.971e-05	5.190e-05
2	-1.525e-07	3.269e-06	-3.812e-06	5.601e-08	-9.938e-05	3.778e-07	-6.019e-05	4.979e-05	5.318e-05	6.982e-05	-1.761e-04	-4.958e-05
3	-1.096e-06	-1.015e-06	-1.022e-06	-5.922e-07	-6.328e-05	-6.308e-05	-3.564e-05	-3.563e-05	-7.419e-05	2.300e-06	-7.436e-05	2.125e-06
4	4.618e-07	5.000e-06	-5.529e-06	-6.372e-08	-1.311e-04	1.577e-04	-8.018e-05	1.143e-04	1.456e-04	1.037e-04	-6.207e-05	-1.118e-04
5	2.676e-07	-9.265e-06	8.959e-06	-2.799e-08	2.580e-04	-2.420e-04	1.778e-04	-1.587e-04	-1.547e-04	-1.890e-04	2.048e-04	1.840e-04
6	-6.042e-07	-9.024e-07	-9.043e-07	-3.186e-07	-1.111e-04	-1.110e-04	-5.106e-05	-5.119e-05	-1.486e-04	1.587e-05	-1.487e-04	1.579e-05
7	4.481e-07	8.992e-06	-7.450e-06	-1.331e-07	-1.816e-04	2.148e-04	-1.112e-04	1.442e-04	1.813e-04	1.424e-04	-1.038e-04	-1.534e-04
8	1.956e-07	-1.046e-05	1.118e-05	1.390e-06	2.910e-04	-2.710e-04	1.749e-04	-1.612e-04	-2.934e-04	-2.450e-04	3.209e-04	2.343e-04
9	-1.823e-07	3.038e-06	-2.949e-06	-2.308e-08	-1.201e-05	7.238e-05	-5.205e-05	4.094e-05	-3.610e-05	9.619e-05	-4.798e-05	-5.670e-05
10	3.927e-07	5.962e-06	-4.566e-06	1.130e-06	-1.277e-04	1.611e-04	-6.735e-05	9.623e-05	2.253e-04	1.265e-04	-1.643e-04	-1.409e-04
11	2.920e-07	-1.204e-05	1.301e-05	-7.942e-07	3.221e-04	-3.021e-04	2.048e-04	-1.842e-04	-2.485e-04	-2.503e-04	3.054e-04	2.454e-04
12	-1.818e-07	-2.947e-06	3.035e-06	-1.901e-08	7.248e-05	-1.177e-05	4.079e-05	-5.216e-05	-4.749e-05	-5.672e-05	-3.570e-05	9.599e-05

MODE 4

DOF	$\partial M_4/\partial DV_1$	$\partial M_4/\partial DV_2$	$\partial M_4/\partial DV_3$	$\partial M_4/\partial DV_4$	$\partial M_4/\partial DV_5$	$\partial M_4/\partial DV_6$	$\partial M_4/\partial DV_7$	$\partial M_4/\partial DV_8$	$\partial M_4/\partial DV_9$	$\partial M_4/\partial DV_{10}$	$\partial M_4/\partial DV_{11}$	$\partial M_4/\partial DV_{12}$
1	1.651e-10	-2.254e-06	1.930e-07	1.005e-10	6.794e-05	-6.198e-05	1.760e-05	-1.178e-04	-4.076e-05	1.983e-05	2.597e-05	-4.880e-05
2	1.269e-10	-8.850e-08	1.327e-06	4.317e-11	2.060e-05	-5.444e-05	6.127e-05	-1.695e-05	-1.010e-05	7.528e-06	2.851e-05	-3.207e-05
3	9.329e-14	2.099e-05	-2.342e-06	-3.341e-09	-1.955e-04	3.846e-04	5.167e-04	5.165e-04	3.175e-04	5.870e-04	-2.590e-04	1.027e-05
4	-6.918e-10	1.554e-05	-2.800e-06	-2.425e-09	-1.783e-04	3.231e-04	1.611e-04	6.760e-04	2.448e-04	3.845e-04	-2.127e-04	1.569e-05
5	-3.721e-10	1.048e-05	-1.184e-07	-1.479e-09	-1.058e-04	1.839e-04	2.492e-05	3.223e-04	1.552e-04	2.795e-04	-1.091e-04	6.633e-05
6	1.905e-10	5.009e-06	1.362e-09	-9.905e-10	-1.674e-08	5.090e-05	3.851e-04	-1.073e-04	6.643e-05	2.110e-04	-5.065e-05	-5.049e-05
7	-6.195e-10	1.648e-05	-2.503e-06	-2.065e-09	-1.541e-04	2.851e-04	1.252e-04	5.905e-04	2.094e-04	3.214e-04	-1.861e-04	1.468e-05
8	-3.126e-10	7.542e-06	-1.330e-06	-1.336e-09	-7.814e-05	1.293e-04	1.081e-04	3.196e-04	8.731e-05	1.253e-04	-9.509e-05	-5.514e-06
9	-8.376e-11	7.791e-06	-1.124e-06	-1.140e-09	-1.169e-04	1.416e-04	1.375e-04	2.095e-04	8.727e-05	2.853e-04	-9.473e-05	1.364e-06
10	-5.179e-10	1.347e-05	-3.701e-06	-2.207e-09	-1.422e-04	2.568e-04	4.108e-05	4.569e-04	2.100e-04	3.518e-04	-1.455e-04	8.531e-05
11	-4.050e-10	9.026e-06	-2.990e-06	-1.192e-09	-1.063e-04	1.708e-04	6.472e-05	3.621e-04	1.374e-04	2.254e-04	-1.143e-04	2.489e-05
12	-1.067e-10	7.750e-06	-1.171e-06	-1.141e-09	-7.453e-05	1.841e-04	1.574e-04	2.295e-04	1.156e-04	2.108e-04	-6.647e-05	-7.320e-05

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-1 (Cont.)
Eigenvector Derivatives from Original Nelson

MODE 5

DOF	$\partial M_5/\partial DV_1$	$\partial M_5/\partial DV_2$	$\partial M_5/\partial DV_3$	$\partial M_5/\partial DV_4$	$\partial M_5/\partial DV_5$	$\partial M_5/\partial DV_6$	$\partial M_5/\partial DV_7$	$\partial M_5/\partial DV_8$	$\partial M_5/\partial DV_9$	$\partial M_5/\partial DV_{10}$	$\partial M_5/\partial DV_{11}$	$\partial M_5/\partial DV_{12}$
1	1.717e-07	-2.082e-10	2.180e-10	-1.604e-07	-1.438e-05	-2.801e-06	4.178e-05	8.266e-05	7.473e-06	-3.503e-05	2.064e-06	4.276e-07
2	9.938e-08	3.090e-10	-4.312e-10	-9.261e-08	5.073e-06	-1.500e-05	7.133e-05	5.579e-07	-1.937e-06	2.071e-05	7.446e-06	-4.070e-05
3	-8.509e-07	-1.072e-09	-1.393e-09	-8.700e-07	1.911e-05	1.911e-05	5.585e-05	5.581e-05	-3.161e-05	2.237e-06	-3.164e-05	2.236e-06
4	1.585e-06	1.642e-09	-2.880e-09	-7.191e-07	2.116e-05	8.960e-06	9.775e-05	3.544e-05	-1.425e-05	4.299e-05	-3.464e-05	-7.796e-05
5	9.142e-07	-3.501e-09	4.100e-09	-4.153e-07	-1.879e-06	1.927e-05	-1.562e-05	9.237e-05	-3.172e-05	-1.148e-04	3.529e-06	9.466e-05
6	2.158e-06	-7.116e-10	-9.150e-10	-3.515e-07	-1.912e-07	-1.957e-07	-1.213e-05	-1.221e-05	-6.580e-06	1.885e-06	-6.588e-06	1.868e-06
7	1.351e-06	-1.096e-09	-3.380e-09	-5.644e-07	2.063e-05	5.426e-06	1.310e-04	-3.448e-05	-7.487e-06	6.222e-05	-3.286e-05	-8.838e-05
8	5.642e-07	-3.574e-09	2.342e-09	-1.280e-06	-4.900e-06	1.798e-05	-2.724e-04	3.088e-04	1.065e-06	-2.242e-05	-1.792e-05	2.101e-05
9	3.418e-07	-7.658e-11	-1.626e-09	-1.459e-07	4.106e-05	5.431e-06	1.166e-04	-9.072e-06	1.797e-05	-6.876e-05	-1.485e-05	-1.919e-05
10	1.164e-06	2.782e-10	-4.721e-09	-1.390e-06	1.828e-05	6.075e-06	2.502e-04	-1.704e-04	-3.192e-05	-2.600e-05	-2.824e-06	1.171e-05
11	8.864e-07	-3.160e-09	1.074e-09	1.507e-07	-4.296e-06	2.031e-05	-1.844e-04	2.495e-04	-1.948e-05	-8.705e-05	-7.023e-06	6.510e-05
12	3.416e-07	-1.256e-09	-1.071e-10	-1.461e-07	5.430e-06	4.108e-05	-9.104e-06	1.166e-04	-1.484e-05	-1.921e-05	1.798e-05	-6.879e-05

MODE 6

DOF	$\partial M_6/\partial DV_1$	$\partial M_6/\partial DV_2$	$\partial M_6/\partial DV_3$	$\partial M_6/\partial DV_4$	$\partial M_6/\partial DV_5$	$\partial M_6/\partial DV_6$	$\partial M_6/\partial DV_7$	$\partial M_6/\partial DV_8$	$\partial M_6/\partial DV_9$	$\partial M_6/\partial DV_{10}$	$\partial M_6/\partial DV_{11}$	$\partial M_6/\partial DV_{12}$
1	-1.450e-14	1.417e-15	-5.717e-15	1.079e-14	5.172e-14	-4.798e-15	-6.250e-05	6.251e-05	6.249e-05	-6.251e-05	-6.270e-09	-6.111e-09
2	2.815e-15	-3.418e-15	6.005e-15	-2.998e-15	-3.452e-14	3.041e-15	-3.609e-05	3.608e-05	-3.609e-05	3.609e-05	-7.217e-05	7.216e-05
3	7.760e-15	-9.858e-15	1.697e-14	-8.276e-15	-9.762e-14	8.788e-15	-1.021e-04	1.021e-04	-1.021e-04	1.021e-04	1.020e-04	-1.021e-04
4	2.649e-11	8.264e-11	-2.334e-11	-8.144e-13	4.640e-11	-5.152e-11	8.093e-05	3.149e-06	-2.644e-05	6.243e-05	1.228e-04	1.173e-04
5	-4.588e-11	-2.060e-11	-8.253e-11	1.785e-12	3.467e-11	-2.862e-11	6.750e-05	2.258e-05	-8.717e-06	-5.358e-05	7.749e-05	-2.193e-05
6	5.859e-10	-4.333e-11	4.349e-11	-1.358e-13	-4.075e-11	4.168e-11	-1.724e-04	8.904e-05	-5.728e-05	5.726e-05	-2.610e-05	-1.406e-04
7	5.054e-11	7.144e-11	1.241e-12	-2.491e-11	2.338e-11	-5.152e-11	6.244e-05	-2.647e-05	3.169e-06	8.093e-05	1.173e-04	1.228e-04
8	-3.030e-11	1.055e-11	-2.963e-12	7.511e-11	4.147e-11	2.862e-11	5.360e-05	8.702e-06	-2.254e-05	-6.753e-05	2.195e-05	-7.750e-05
9	4.162e-11	4.001e-11	1.741e-12	-4.178e-11	-5.859e-08	-4.168e-11	-5.722e-05	5.734e-05	3.981e-04	-3.148e-04	1.406e-04	2.611e-05
10	9.601e-13	1.949e-12	-4.420e-11	-5.142e-11	4.199e-12	-5.298e-11	5.967e-05	-2.370e-05	-2.371e-05	5.969e-05	7.193e-05	7.194e-05
11	-5.892e-11	-2.339e-12	-5.638e-11	5.873e-11	5.850e-11	-2.196e-15	8.620e-05	3.165e-05	-3.163e-05	-8.622e-05	4.490e-05	-4.489e-05
12	-4.166e-11	-1.661e-12	-3.985e-11	4.152e-11	4.129e-11	3.889e-15	-1.560e-05	9.892e-05	-9.891e-05	1.558e-05	-3.564e-04	3.564e-04

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-2

Eigenvector Derivatives from Finite Difference Method

MODE 1

DOF	$\partial M_1/\partial DV_1$	$\partial M_1/\partial DV_2$	$\partial M_1/\partial DV_3$	$\partial M_1/\partial DV_4$	$\partial M_1/\partial DV_5$	$\partial M_1/\partial DV_6$	$\partial M_1/\partial DV_7$	$\partial M_1/\partial DV_8$	$\partial M_1/\partial DV_9$	$\partial M_1/\partial DV_{10}$	$\partial M_1/\partial DV_{11}$	$\partial M_1/\partial DV_{12}$
1	5.821e-10	-1.921e-06	1.709e-06	5.821e-10	3.530e-04	-3.367e-04	6.129e-05	-6.785e-05	-2.103e-05	4.758e-05	1.398e-05	-4.647e-05
2	0.000E+00	-8.650e-07	1.232e-06	0.000E+00	1.851e-04	-2.132e-04	4.299e-05	-3.160e-05	-4.022e-06	2.622e-05	1.620e-05	-2.809e-05
3	-2.442e-14	6.177e-08	-6.174e-08	-2.150e-10	-1.934e-04	1.934e-04	-1.549e-08	1.617e-08	-1.718e-05	-1.714e-05	1.718e-05	1.713e-05
4	1.091e-10	-1.348e-06	1.329e-06	3.274e-10	3.083e-05	-5.714e-05	2.509e-05	8.491e-06	-3.618e-05	1.593e-05	3.073e-05	-1.749e-05
5	3.638e-11	-7.567e-07	7.888e-07	1.819e-10	4.816e-05	-2.636e-06	-2.430e-05	-3.386e-05	-1.460e-05	1.101e-05	2.404e-05	-8.292e-06
6	8.093e-11	-2.220e-07	2.220e-07	8.241e-11	3.541e-05	-3.542e-05	4.215e-05	-4.213e-05	-2.819e-06	5.162e-06	2.815e-06	-5.160e-06
7	7.276e-11	6.644e-07	1.267e-06	2.910e-10	1.760e-05	-5.907e-05	2.375e-05	9.623e-06	-3.654e-05	1.439e-05	2.803e-05	-1.689e-05
8	0.000E+00	-6.638e-07	-4.302e-07	-7.276e-11	4.109e-05	1.319e-05	-6.649e-06	-1.195e-05	-4.200e-05	1.966e-05	-7.125e-06	4.353e-06
9	-7.276e-11	2.858e-07	5.697e-08	-7.276e-11	-6.213e-06	5.121e-05	-7.240e-07	7.253e-06	-4.496e-05	-2.089e-05	6.426e-06	4.187e-06
10	7.276e-11	-2.611e-07	2.426e-07	1.091e-10	1.809e-05	-4.439e-05	5.537e-06	-6.118e-06	-7.845e-06	4.677e-06	5.465e-05	-2.420e-05
11	7.276e-11	-1.312e-06	-9.074e-07	2.910e-10	5.776e-05	5.290e-06	-1.432e-05	-2.389e-05	-2.783e-05	1.682e-05	1.065e-05	-2.633e-06
12	0.000E+00	-5.697e-08	-2.859e-07	-7.276e-11	-5.121e-05	6.229e-06	-7.258e-06	7.247e-07	-6.425e-06	-4.189e-06	4.497e-05	2.088e-05

MODE 2

DOF	$\partial M_2/\partial DV_1$	$\partial M_2/\partial DV_2$	$\partial M_2/\partial DV_3$	$\partial M_2/\partial DV_4$	$\partial M_2/\partial DV_5$	$\partial M_2/\partial DV_6$	$\partial M_2/\partial DV_7$	$\partial M_2/\partial DV_8$	$\partial M_2/\partial DV_9$	$\partial M_2/\partial DV_{10}$	$\partial M_2/\partial DV_{11}$	$\partial M_2/\partial DV_{12}$
1	-1.175e-06	-1.249e-06	1.250e-06	3.667e-07	1.993e-04	-2.145e-04	3.400e-05	-5.317e-05	-1.356e-05	4.710e-05	1.622e-05	-1.020e-05
2	-6.790e-07	2.164e-06	-2.163e-06	2.104e-07	-3.626e-04	3.538e-04	-8.102e-05	6.993e-05	2.657e-05	-3.898e-05	-2.501e-05	6.025e-05
3	-2.744e-06	-5.725e-07	-5.723e-07	-8.137e-07	7.909e-05	7.911e-05	-3.854e-05	-3.855e-05	-7.695e-06	-7.922e-06	-7.691e-06	-7.916e-06
4	5.927e-07	-9.006e-07	-7.563e-07	-1.662e-06	7.553e-05	6.020e-06	-1.686e-05	3.534e-05	-1.033e-05	-1.915e-05	-6.495e-06	-3.656e-05
5	3.420e-07	-3.536e-07	-6.031e-07	-9.596e-07	-3.665e-05	8.374e-05	5.054e-05	-3.987e-05	-1.544e-06	-3.117e-05	-8.166e-06	-1.003e-06
6	1.164e-07	7.763e-08	7.771e-08	6.101e-07	2.860e-05	2.859e-05	-5.460e-05	-5.461e-05	2.001e-07	2.154e-05	2.008e-07	2.154e-05
7	5.455e-07	7.133e-07	-6.946e-07	-1.687e-06	9.072e-05	-2.022e-05	-1.617e-05	3.304e-05	-1.119e-05	-1.594e-05	-5.077e-06	-4.373e-05
8	2.115e-07	-3.244e-07	3.222e-07	1.166e-06	-5.770e-05	8.660e-05	1.457e-05	-8.049e-06	-8.531e-06	-9.176e-05	-4.438e-07	5.167e-05
9	-4.130e-07	2.657e-07	-4.927e-07	-4.920e-07	-7.466e-05	5.125e-05	-1.945e-05	7.026e-06	-1.026e-05	6.921e-05	-5.412e-06	-6.260e-06
10	4.560e-07	-6.832e-08	7.582e-08	1.665e-07	6.488e-05	-4.623e-06	9.550e-06	4.531e-06	-2.925e-06	2.289e-05	-1.298e-05	-8.743e-05
11	3.664e-07	-7.629e-07	7.798e-07	-2.044e-06	-6.079e-05	1.074e-04	3.263e-05	-2.129e-05	-4.179e-06	-6.372e-05	-5.420e-06	3.208e-05
12	-4.128e-07	-4.929e-07	2.657e-07	-4.920e-07	5.124e-05	-7.466e-05	7.031e-06	-1.945e-05	-5.415e-06	-6.262e-06	-1.025e-05	6.922e-05

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-2 (Cont.)
Eigenvector Derivatives from Finite Difference

MODE 3

DOF	$\partial M_3/\partial DV_1$	$\partial M_3/\partial DV_2$	$\partial M_3/\partial DV_3$	$\partial M_3/\partial DV_4$	$\partial M_3/\partial DV_5$	$\partial M_3/\partial DV_6$	$\partial M_3/\partial DV_7$	$\partial M_3/\partial DV_8$	$\partial M_3/\partial DV_9$	$\partial M_3/\partial DV_{10}$	$\partial M_3/\partial DV_{11}$	$\partial M_3/\partial DV_{12}$
1	-3.237e-07	-3.536e-06	4.512e-06	2.638e-06	-2.127e-05	-1.398e-04	4.871e-05	-6.986e-05	-1.126e-04	-1.424e-05	1.603e-04	1.185e-04
2	-1.873e-07	7.253e-06	-6.683e-06	1.521e-06	-1.492e-04	5.597e-05	-1.085e-04	9.671e-05	2.498e-04	1.453e-04	-2.228e-04	-8.480e-05
3	-1.968e-06	-6.577e-07	-6.746e-07	1.274e-06	-4.240e-05	-4.194e-05	-6.191e-05	-6.185e-05	8.230e-05	1.971e-05	8.186e-05	1.932e-05
4	8.426e-07	9.284e-06	-1.145e-05	-9.986e-07	-3.014e-04	2.829e-04	-1.534e-04	2.095e-04	2.181e-04	1.906e-04	-2.277e-04	-2.245e-04
5	4.875e-07	-1.858e-05	1.731e-05	-5.697e-07	5.007e-04	-5.107e-04	3.295e-04	-2.984e-04	-3.876e-04	-3.695e-04	3.838e-04	3.491e-04
6	-8.718e-07	5.120e-07	5.036e-07	3.355e-06	-7.722e-05	-7.703e-05	-8.515e-05	-8.532e-05	8.617e-05	5.993e-05	8.599e-05	5.965e-05
7	8.068e-07	1.708e-05	-1.528e-05	-1.242e-06	-4.068e-04	3.950e-04	-2.114e-04	2.652e-04	2.853e-04	2.646e-04	-3.260e-04	-3.051e-04
8	3.482e-07	-2.089e-05	2.171e-05	2.278e-06	5.695e-04	-5.675e-04	3.246e-04	-3.026e-04	-6.614e-04	-4.763e-04	6.216e-04	4.464e-04
9	-3.625e-07	5.940e-06	-5.846e-06	-6.854e-08	-3.006e-05	1.439e-04	-9.783e-05	7.568e-05	-6.622e-05	1.836e-04	-1.079e-04	-1.105e-04
10	7.043e-07	1.117e-05	-9.559e-06	1.345e-06	-2.945e-04	2.898e-04	-1.292e-04	1.760e-04	3.748e-04	2.346e-04	-4.311e-04	-2.798e-04
11	5.263e-07	-2.410e-05	2.521e-05	-2.202e-06	6.262e-04	-6.364e-04	3.797e-04	-3.463e-04	-5.913e-04	-4.882e-04	5.792e-04	4.663e-04
12	-3.619e-07	-5.845e-06	5.932e-06	-6.534e-08	1.439e-04	-2.976e-05	7.539e-05	-9.805e-05	-1.073e-04	-1.106e-04	-6.591e-05	1.831e-04

MODE 4

DOF	$\partial M_4/\partial DV_1$	$\partial M_4/\partial DV_2$	$\partial M_4/\partial DV_3$	$\partial M_4/\partial DV_4$	$\partial M_4/\partial DV_5$	$\partial M_4/\partial DV_6$	$\partial M_4/\partial DV_7$	$\partial M_4/\partial DV_8$	$\partial M_4/\partial DV_9$	$\partial M_4/\partial DV_{10}$	$\partial M_4/\partial DV_{11}$	$\partial M_4/\partial DV_{12}$
1	3.274e-10	-3.040e-06	1.766e-06	1.819e-10	1.522e-04	-1.118e-04	1.107e-04	-1.528e-04	-8.202e-05	8.647e-05	5.895e-05	-5.406e-05
2	2.183e-10	-2.889e-07	2.489e-06	7.276e-11	4.141e-05	-1.110e-04	1.124e-04	-3.976e-05	-2.058e-05	1.232e-05	6.097e-05	-6.876e-05
3	-2.842e-13	2.297e-05	-2.295e-05	-6.661e-09	-5.871e-04	5.866e-04	2.326e-05	-2.208e-05	6.084e-04	5.581e-04	-6.101e-04	-5.573e-04
4	-1.310e-09	1.691e-05	-1.918e-05	-4.802e-09	-5.017e-04	5.134e-04	-4.238e-04	5.576e-04	4.726e-04	3.096e-04	-4.942e-04	-3.967e-04
5	-8.731e-10	1.240e-05	-8.449e-06	-2.910e-09	-3.039e-04	2.826e-04	-3.984e-04	1.685e-04	2.964e-04	2.803e-04	-2.621e-04	-1.278e-04
6	3.768e-10	4.939e-06	-4.939e-06	-1.975e-09	-5.097e-05	5.092e-05	4.914e-04	-4.911e-04	1.236e-04	2.578e-04	-1.240e-04	-2.577e-04
7	-1.310e-09	2.056e-05	-1.684e-05	-4.220e-09	-4.336e-04	4.555e-04	-3.995e-04	4.888e-04	4.046e-04	2.427e-04	-4.313e-04	-3.435e-04
8	-5.821e-10	8.988e-06	-8.495e-06	-2.619e-09	-2.235e-04	1.964e-04	-1.060e-04	2.975e-04	1.605e-04	5.281e-05	-2.250e-04	-1.963e-04
9	-2.183e-10	8.986e-06	-8.573e-06	-2.328e-09	-3.029e-04	2.212e-04	-7.374e-05	5.235e-05	1.636e-04	3.561e-04	-2.209e-04	-1.969e-04
10	-1.019e-09	1.578e-05	-1.805e-05	-4.366e-09	-3.980e-04	4.098e-04	-5.013e-04	2.922e-04	4.092e-04	3.422e-04	-3.420e-04	-1.663e-04
11	-8.731e-10	1.036e-05	-1.332e-05	-2.328e-09	-2.969e-04	2.639e-04	-2.742e-04	2.936e-04	2.607e-04	2.001e-04	-2.712e-04	-1.836e-04
12	-2.183e-10	8.584e-06	-8.986e-06	-2.256e-09	-2.216e-04	3.028e-04	-5.203e-05	7.422e-05	2.204e-04	1.973e-04	-1.642e-04	-3.559e-04

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-2 (Cont.)
Eigenvector Derivatives from Finite Difference

MODE 5

DOF	$\partial M_5/\partial DV_1$	$\partial M_5/\partial DV_2$	$\partial M_5/\partial DV_3$	$\partial M_5/\partial DV_4$	$\partial M_5/\partial DV_5$	$\partial M_5/\partial DV_6$	$\partial M_5/\partial DV_7$	$\partial M_5/\partial DV_8$	$\partial M_5/\partial DV_9$	$\partial M_5/\partial DV_{10}$	$\partial M_5/\partial DV_{11}$	$\partial M_5/\partial DV_{12}$
1	-4.470e-07	-4.366e-10	4.366e-10	-4.943e-07	-3.063e-05	-7.464e-06	2.110e-05	1.012e-04	1.368e-05	-8.177e-05	2.903e-06	-1.151e-05
2	-2.580e-07	6.548e-10	-8.731e-10	-2.855e-07	9.072e-06	-3.107e-05	1.047e-04	-3.404e-05	-4.555e-06	3.391e-05	1.413e-05	-8.780e-05
3	-2.393e-06	-2.037e-09	-2.619e-09	-1.882e-06	3.651e-05	3.652e-05	5.237e-05	5.228e-05	-6.405e-05	-6.772e-06	-6.413e-05	-6.778e-06
4	2.296e-06	3.201e-09	-5.821e-09	-1.613e-06	4.038e-05	1.597e-05	1.258e-04	8.597e-06	-2.960e-05	7.262e-05	-7.021e-05	-1.673e-04
5	1.325e-06	-6.985e-09	8.149e-09	-9.310e-07	-4.879e-06	3.741e-05	-6.278e-05	1.402e-04	-6.389e-05	-2.350e-04	6.322e-06	1.804e-04
6	4.424e-06	-1.746e-09	-1.746e-09	-6.636e-07	-9.313e-08	-1.048e-07	-1.634e-05	-1.647e-05	-1.296e-05	5.442e-06	-1.297e-05	5.407e-06
7	2.064e-06	-2.183e-09	-6.839e-09	-1.254e-06	3.984e-05	9.447e-06	2.097e-04	-1.120e-04	-1.579e-05	1.143e-04	-6.633e-05	-1.843e-04
8	8.872e-07	-7.130e-09	4.584e-09	-2.587e-06	-1.033e-05	3.543e-05	-5.595e-04	5.965e-04	1.801e-06	-4.740e-05	-3.603e-05	3.783e-05
9	-3.492e-09	-2.183e-10	-3.274e-09	-4.413e-07	8.055e-05	9.261e-06	?.777e-04	-7.143e-05	3.476e-05	-1.473e-04	-3.060e-05	-4.853e-05
10	1.800e-06	4.366e-10	-9.459e-09	-2.867e-06	3.540e-05	1.099e-05	4.606e-04	-3.797e-04	-6.430e-05	-5.941e-05	-6.343e-06	1.611e-05
11	1.343e-06	-6.330e-09	2.037e-09	2.070e-07	-9.538e-06	3.967e-05	-3.953e-04	4.614e-04	-3.938e-05	-1.785e-04	-1.459e-05	1.227e-04
12	-3.347e-09	-2.474e-09	-2.183e-10	-4.416e-07	9.260e-06	8.059e-05	-7.143e-05	1.776e-04	-3.058e-05	-4.854e-05	3.479e-05	-1.473e-04

MODE 6

DOF	$\partial M_6/\partial DV_1$	$\partial M_6/\partial DV_2$	$\partial M_6/\partial DV_3$	$\partial M_6/\partial DV_4$	$\partial M_6/\partial DV_5$	$\partial M_6/\partial DV_6$	$\partial M_6/\partial DV_7$	$\partial M_6/\partial DV_8$	$\partial M_6/\partial DV_9$	$\partial M_6/\partial DV_{10}$	$\partial M_6/\partial DV_{11}$	$\partial M_6/\partial DV_{12}$
1	-1.776e-14	0.000E+00	-1.776e-14	0.000E+00	0.000E+00	0.000E+00	-1.257e-04	1.256e-04	1.236e-04	-1.238e-04	1.592e-07	1.577e-07
2	2.628e-15	-3.787e-15	5.967e-15	-2.996e-15	-6.218e-14	3.039e-15	-7.241e-05	7.270e-05	-7.154e-05	7.127e-05	-1.428e-04	1.428e-04
3	7.232e-15	-1.091e-14	1.687e-14	-8.269e-15	-1.758e-13	8.782e-15	-2.046e-04	2.046e-04	-2.014e-04	2.014e-04	2.014e-04	-2.014e-04
4	1.455e-10	1.455e-10	0.000E+00	0.000E+00	0.000E+00	0.000E+00	4.246e-05	-1.136e-04	-1.689e-04	6.582e-06	1.246e-04	1.136e-04
5	0.000E+00	0.000E+00	-2.910e-10	0.000E+00	0.000E+00	0.000E+00	1.067e-04	1.654e-05	-4.627e-05	-1.350e-04	1.232e-04	-7.327e-05
6	1.160e-09	-4.496e-11	4.526e-11	-2.709e-13	-3.726e-11	4.170e-11	-2.620e-04	2.621e-04	-3.121e-05	1.947e-04	3.114e-05	-1.947e-04
7	1.455e-10	1.455e-10	0.000E+00	0.000E+00	0.000E+00	0.000E+00	5.215e-06	-1.733e-04	-1.106e-04	4.289e-05	1.136e-04	1.246e-04
8	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	1.364e-04	4.615e-05	-1.423e-05	-1.031e-04	7.330e-05	-1.233e-04
9	4.164e-11	3.844e-11	3.553e-12	-4.185e-11	-1.160e-07	-4.121e-11	-2.000e-04	3.139e-05	7.067e-04	-7.068e-04	1.962e-04	-3.138e-05
10	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	4.670e-05	-1.208e-04	-1.203e-04	4.450e-05	6.771e-05	6.774e-05
11	-5.892e-11	-4.631e-12	-5.388e-11	5.855e-11	5.809e-11	0.000E+00	1.732e-04	6.366e-05	-6.171e-05	-1.696e-04	8.869e-05	-8.868e-05
12	-4.166e-11	-3.283e-12	-3.809e-11	4.139e-11	4.094e-11	2.776e-15	-3.126e-05	2.000e-04	-1.961e-04	3.140e-05	-7.067e-04	7.066e-04

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-3

Eigenvector Derivatives from ASTROS (Nelson's Method)

MODE 1

DOF	$\partial M_1/\partial DV_1$	$\partial M_1/\partial DV_2$	$\partial M_1/\partial DV_3$	$\partial M_1/\partial DV_4$	$\partial M_1/\partial DV_5$	$\partial M_1/\partial DV_6$	$\partial M_1/\partial DV_7$	$\partial M_1/\partial DV_8$	$\partial M_1/\partial DV_9$	$\partial M_1/\partial DV_{10}$	$\partial M_1/\partial DV_{11}$	$\partial M_1/\partial DV_{12}$
1	7.0314e-09	-1.9406e-05	1.7259e-05	7.4373e-09	3.5408e-04	-3.3915e-04	6.1974e-05	-6.8680e-05	-2.1246e-05	4.8214e-05	1.4064e-05	-4.7123e-05
2	3.1236e-09	-8.7282e-06	1.2446e-05	4.2377e-09	1.8719e-04	-2.1314e-04	4.3554e-05	-3.1901e-05	-3.9913e-06	2.6610e-05	1.6403e-05	-2.8446e-05
3	-2.4951e-13	6.2165e-07	-6.2134e-07	-2.1740e-09	-1.9476e-04	1.9476e-04	-2.1776e-09	2.8811e-09	-1.7365e-05	-1.7371e-05	1.7370e-05	1.7364e-05
4	8.9813e-10	-1.3613e-05	1.3427e-05	3.2348e-09	3.0661e-05	-5.7374e-05	2.5386e-05	8.6008e-06	-3.6550e-05	1.6153e-05	3.1037e-05	-1.7731e-05
5	4.0897e-10	-7.6431e-06	7.9689e-06	1.8876e-09	4.8533e-05	-2.3025e-06	-2.4597e-05	-3.4266e-05	-1.4737e-05	1.1161e-05	2.4286e-05	-8.4039e-06
6	8.1834e-10	-2.2419e-06	2.2415e-06	8.3351e-10	3.5440e-05	-3.5454e-05	4.2657e-05	-4.2643e-05	-2.8416e-06	5.2329e-06	2.8381e-06	-5.2306e-06
7	8.0198e-10	6.7130e-06	1.2800e-05	3.0431e-09	1.7296e-05	-5.9432e-05	2.4034e-05	9.7439e-06	-3.6923e-05	1.4586e-05	2.8305e-05	-1.7127e-05
8	1.6217e-10	-6.7043e-06	-4.3455e-06	-5.8275e-10	4.1526e-05	1.3612e-05	-6.7264e-06	-1.2086e-05	-4.2429e-05	1.9933e-05	-7.1972e-06	4.4181e-06
9	-4.9462e-10	2.8878e-06	5.7692e-07	-5.3431e-10	-5.8777e-06	5.1653e-05	-7.2200e-07	7.3438e-06	-4.5432e-05	-2.1178e-05	6.5013e-06	4.2479e-06
10	6.5387e-10	-2.6369e-06	2.4489e-06	9.8186e-10	1.7909e-05	-4.4615e-05	5.5946e-06	-6.1899e-06	-7.9211e-06	4.7389e-06	5.5211e-05	-2.4540e-05
11	3.6884e-10	-1.3257e-05	-9.1673e-06	2.9803e-09	5.8277e-05	5.7631e-06	-1.4491e-05	-2.4166e-05	-2.8110e-05	1.7052e-05	1.0761e-05	-2.6653e-06
12	-3.2375e-10	-5.7695e-07	-2.8889e-06	-5.4581e-10	-5.1655e-05	5.8940e-06	-7.3489e-06	7.2315e-07	-6.5006e-06	-4.2499e-06	4.5436e-05	2.1168e-05

MODE 2

DOF	$\partial M_2/\partial DV_1$	$\partial M_2/\partial DV_2$	$\partial M_2/\partial DV_3$	$\partial M_2/\partial DV_4$	$\partial M_2/\partial DV_5$	$\partial M_2/\partial DV_6$	$\partial M_2/\partial DV_7$	$\partial M_2/\partial DV_8$	$\partial M_2/\partial DV_9$	$\partial M_2/\partial DV_{10}$	$\partial M_2/\partial DV_{11}$	$\partial M_2/\partial DV_{12}$
1	-1.1859e-05	-1.2610e-05	1.2630e-05	3.7060e-06	2.0162e-04	-2.1450e-04	3.4538e-05	-5.3729e-05	-1.3692e-05	4.7843e-05	1.6389e-05	-1.0334e-05
2	-6.8558e-06	2.1860e-05	-2.1848e-05	2.1282e-06	-3.6398e-04	3.5660e-04	-8.1975e-05	7.0871e-05	2.6834e-05	-3.9561e-05	-2.5260e-05	6.1181e-05
3	-2.7718e-05	-5.7824e-06	-5.7813e-06	-8.2181e-06	7.8549e-05	7.8569e-05	-3.8994e-05	-3.9004e-05	-7.7732e-06	-8.0417e-06	-7.7687e-06	-8.0357e-06
4	5.9870e-06	-9.0944e-06	-7.6361e-06	-1.6778e-05	7.5811e-05	6.1355e-06	-1.7013e-05	3.5740e-05	-1.0426e-05	-1.9290e-05	-6.5574e-06	-3.6880e-05
5	3.4553e-06	-3.5696e-06	-6.0902e-06	-9.6884e-06	-3.6673e-05	8.3995e-05	5.1092e-05	-4.0282e-05	-1.5578e-06	-3.1456e-05	-8.2457e-06	-9.8018e-07
6	1.1786e-06	7.8492e-07	7.8574e-07	6.1624e-06	2.8765e-05	2.8755e-05	-5.5175e-05	-5.5189e-05	2.0320e-07	2.1764e-05	2.0440e-07	2.1761e-05
7	5.5101e-06	7.2036e-06	-7.0129e-06	-1.7033e-05	9.1030e-05	-2.0186e-05	-1.6314e-05	3.3403e-05	-1.1298e-05	-1.6048e-05	-5.1260e-06	-4.4128e-05
8	2.1364e-06	-3.2746e-06	3.2533e-06	1.1779e-05	-5.7815e-05	8.6834e-05	1.4716e-05	-8.1110e-06	-8.6150e-06	-9.2653e-05	-4.4781e-07	5.2197e-05
9	-4.1711e-06	2.6832e-06	-4.9771e-06	-4.9686e-06	-7.5007e-05	5.1309e-05	-1.9684e-05	7.1033e-06	-1.0357e-05	6.9836e-05	-5.4668e-06	-6.2991e-06
10	4.6058e-06	-6.8993e-07	7.6690e-07	1.6853e-06	6.5104e-05	-4.5662e-06	9.6796e-06	4.5868e-06	-2.9532e-06	2.3141e-05	-1.3102e-05	-8.8257e-05
11	3.7013e-06	-7.7020e-06	7.8749e-06	-2.0644e-05	-6.0882e-05	1.0775e-04	3.2982e-05	-2.1490e-05	-4.2184e-06	-6.4324e-05	-5.4733e-06	3.2424e-05
12	-4.1692e-06	-4.9778e-06	2.6835e-06	-4.9681e-06	5.1297e-05	-7.5012e-05	7.1085e-06	-1.9684e-05	-5.4698e-06	-6.3015e-06	-1.0352e-05	6.9845e-05

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-3 (Cont.)
Eigenvector Derivatives from ASTROS (Nelson's Method)

MODE 3

DOF	$\partial M_3/\partial DV_1$	$\partial M_3/\partial DV_2$	$\partial M_3/\partial DV_3$	$\partial M_3/\partial DV_4$	$\partial M_3/\partial DV_5$	$\partial M_3/\partial DV_6$	$\partial M_3/\partial DV_7$	$\partial M_3/\partial DV_8$	$\partial M_3/\partial DV_9$	$\partial M_3/\partial DV_{10}$	$\partial M_3/\partial DV_{11}$	$\partial M_3/\partial DV_{12}$
1	3.2741e-06	3.6021e-05	-4.5776e-05	-2.6649e-05	2.4735e-05	1.4004e-04	-5.1951e-05	7.5152e-05	1.0901e-04	1.4905e-05	-1.5586e-04	-1.2291e-04
2	1.8939e-06	-7.3671e-05	6.7975e-05	-1.5364e-05	1.4745e-04	-5.2094e-05	1.1648e-04	-1.0351e-04	-2.4258e-04	-1.5075e-04	2.1607e-04	8.8108e-05
3	1.9905e-05	5.9370e-06	6.1106e-06	-1.2862e-05	3.7555e-05	3.7107e-05	6.4498e-05	6.4428e-05	-8.7891e-05	-2.3853e-05	-8.7503e-05	-2.3431e-05
4	-8.5223e-06	-9.4884e-05	1.1569e-04	1.0064e-05	2.9393e-04	-2.8371e-04	1.6277e-04	-2.2623e-04	-2.0720e-04	-2.0055e-04	2.0819e-04	2.3046e-04
5	-4.9308e-06	1.8832e-04	-1.7616e-04	5.7424e-06	-4.9728e-04	5.0259e-04	-3.5420e-04	3.1890e-04	3.5889e-04	3.8210e-04	-3.5999e-04	-3.6396e-04
6	8.8162e-06	-5.3548e-06	-5.2693e-06	-3.3867e-05	7.7188e-05	7.7002e-05	9.1076e-05	9.1256e-05	-8.7545e-05	-6.3101e-05	-8.7388e-05	-6.2811e-05
7	-8.1615e-06	-1.7411e-04	1.5472e-04	1.2525e-05	3.9873e-04	-3.9414e-04	2.2515e-04	-2.8571e-04	-2.6834e-04	-2.7704e-04	3.0180e-04	3.1456e-04
8	-3.5232e-06	2.1196e-04	-2.2080e-04	-2.3013e-05	-5.6482e-04	5.5938e-04	-3.4854e-04	3.2364e-04	6.3266e-04	4.9382e-04	-5.9587e-04	-4.6489e-04
9	3.6644e-06	-6.0633e-05	5.9119e-05	6.9092e-07	2.4844e-05	-1.4394e-04	1.0417e-04	-8.1809e-05	7.4385e-05	-1.9221e-04	9.8149e-05	1.1358e-04
10	-7.1230e-06	-1.1402e-04	9.6544e-05	-1.3605e-05	2.8782e-04	-2.8975e-04	1.3717e-04	-1.8998e-04	-3.6460e-04	-2.4594e-04	4.1467e-04	2.8881e-04
11	-5.3256e-06	2.4449e-04	-2.5654e-04	2.2224e-05	-6.2142e-04	6.2704e-04	-4.0794e-04	3.7024e-04	5.5760e-04	5.0562e-04	-5.5014e-04	-4.8586e-04
12	3.6587e-06	5.9105e-05	-6.0546e-05	6.5972e-07	-1.4395e-04	2.4545e-05	-8.1497e-05	1.0439e-04	9.7644e-05	1.1367e-04	7.4079e-05	-1.9176e-04

MODE 4

DOF	$\partial M_4/\partial DV_1$	$\partial M_4/\partial DV_2$	$\partial M_4/\partial DV_3$	$\partial M_4/\partial DV_4$	$\partial M_4/\partial DV_5$	$\partial M_4/\partial DV_6$	$\partial M_4/\partial DV_7$	$\partial M_4/\partial DV_8$	$\partial M_4/\partial DV_9$	$\partial M_4/\partial DV_{10}$	$\partial M_4/\partial DV_{11}$	$\partial M_4/\partial DV_{12}$
1	-3.4383e-09	3.0771e-05	-1.8180e-05	-1.9437e-09	-1.5045e-04	1.0943e-04	-1.1455e-04	1.5626e-04	7.6978e-05	-8.5495e-05	-5.6518e-05	5.1761e-05
2	-2.3016e-09	3.2793e-06	-2.5021e-05	-9.7587e-10	-3.9658e-05	1.1042e-04	-1.1416e-04	4.2264e-05	2.0679e-05	-1.0222e-05	-5.6539e-05	6.8966e-05
3	2.9307e-12	-2.3343e-04	2.3329e-04	6.6807e-08	5.8064e-04	-5.8011e-04	-2.8638e-07	-9.4759e-07	-5.7589e-04	-5.7718e-04	5.7752e-04	5.7633e-04
4	1.3719e-08	-1.7176e-04	1.9524e-04	4.8833e-08	4.9821e-04	-5.0504e-04	4.4919e-04	-5.8121e-04	-4.4542e-04	-3.2344e-04	4.6981e-04	4.1436e-04
5	7.6492e-09	-1.2651e-04	8.5522e-05	2.8996e-08	2.9625e-04	-2.8337e-04	4.1088e-04	-1.8430e-04	-2.8399e-04	-2.9276e-04	2.4470e-04	1.3354e-04
6	-3.8085e-09	-5.0097e-05	5.0096e-05	1.9809e-08	5.1001e-05	-5.0954e-05	-4.9246e-04	4.9214e-04	-1.1694e-04	-2.6151e-04	1.1729e-04	2.6142e-04
7	1.2341e-08	-2.0806e-04	1.7162e-04	4.1641e-08	4.3180e-04	-4.4693e-04	4.2305e-04	-5.0821e-04	-3.8032e-04	-2.5379e-04	4.1103e-04	3.5976e-04
8	5.9208e-09	-9.1914e-05	8.5570e-05	2.6616e-08	2.1625e-04	-1.9869e-04	1.1058e-04	-3.1279e-04	-1.5591e-04	-6.1945e-05	2.0900e-04	1.9982e-04
9	1.7906e-09	-9.1070e-05	8.7290e-05	2.2798e-08	2.9976e-04	-2.1750e-04	8.4085e-05	-6.0334e-05	-1.5398e-04	-3.6313e-04	2.1014e-04	2.0471e-04
10	1.0672e-08	-1.5995e-04	1.8347e-04	4.4035e-08	3.9573e-04	-4.0264e-04	5.2428e-04	-3.0800e-04	-3.8526e-04	-3.5337e-04	3.2592e-04	1.7977e-04
11	7.9783e-09	-1.0607e-04	1.3432e-04	2.3491e-08	2.8833e-04	-2.6595e-04	2.8338e-04	-3.1172e-04	-2.5110e-04	-2.1240e-04	2.5246e-04	1.8872e-04
12	2.0210e-09	-8.7400e-05	9.1078e-05	2.2814e-08	2.1784e-04	-2.9961e-04	5.9996e-05	-8.4596e-05	-2.0968e-04	-2.0515e-04	1.5454e-04	3.6295e-04

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-3 (Cont.)
Eigenvector Derivatives from ASTROS (Nelson's Method)

MODE 5

DOF	$\partial M_5/\partial DV_1$	$\partial M_5/\partial DV_2$	$\partial M_5/\partial DV_3$	$\partial M_5/\partial DV_4$	$\partial M_5/\partial DV_5$	$\partial M_5/\partial DV_6$	$\partial M_5/\partial DV_7$	$\partial M_5/\partial DV_8$	$\partial M_5/\partial DV_9$	$\partial M_5/\partial DV_{10}$	$\partial M_5/\partial DV_{11}$	$\partial M_5/\partial DV_{12}$
1	4.5008e-06	3.9970e-09	-4.5652e-09	4.9858e-06	3.0606e-05	7.4437e-06	-2.1359e-05	-1.0308e-04	-1.3760e-05	8.2254e-05	-2.9410e-06	1.1349e-05
2	2.5971e-06	-6.3243e-09	8.5529e-09	2.8793e-06	-9.0829e-06	3.1057e-05	-1.0671e-04	3.4838e-05	4.5579e-06	-3.4381e-05	-1.4206e-05	8.8451e-05
3	2.4140e-05	2.1638e-08	2.8064e-08	1.8996e-05	-3.6562e-05	-3.6568e-05	-5.5866e-05	-5.5768e-05	6.4277e-05	6.4721e-06	6.4350e-05	6.4799e-06
4	-2.3206e-05	-3.2360e-08	5.8243e-08	1.6286e-05	-4.0360e-05	-1.5948e-05	-1.2890e-04	-4.2572e-06	2.9763e-05	-7.2927e-05	7.0545e-05	1.6898e-04
5	-1.3389e-05	7.0493e-08	-8.1819e-08	9.4030e-06	4.8940e-06	-3.7405e-05	6.9607e-05	-1.4636e-04	6.4175e-05	2.3720e-04	-6.3247e-06	-1.8180e-04
6	-4.4704e-05	1.4001e-08	1.8070e-08	6.6826e-06	2.2587e-08	3.1141e-08	1.2102e-05	1.2242e-05	1.2928e-05	-6.1554e-06	1.2943e-05	-6.1219e-06
7	-2.0857e-05	2.2366e-08	6.8256e-08	1.2668e-05	-3.9823e-05	-9.4237e-06	-2.1367e-04	1.1723e-04	1.5894e-05	-1.1499e-04	6.6651e-05	1.8623e-04
8	-8.9654e-06	7.1506e-08	-4.6326e-08	2.6110e-05	1.0338e-05	-3.5421e-05	5.6296e-04	-5.9940e-04	-1.7831e-06	4.8410e-05	3.6190e-05	-3.8457e-05
9	2.2763e-08	1.7980e-09	3.2560e-08	4.4543e-06	-8.0532e-05	-9.2705e-06	-1.7946e-04	7.1937e-05	-3.4910e-05	1.4807e-04	3.0722e-05	4.8933e-05
10	-1.8188e-05	-4.8006e-09	9.4665e-08	2.8945e-05	-3.5384e-05	-1.0968e-05	-4.6044e-04	3.8070e-04	6.4593e-05	5.9819e-05	6.4103e-06	-1.5580e-05
11	-1.3574e-05	6.3515e-08	-2.1096e-08	-2.0827e-06	9.5544e-06	-3.9660e-05	4.0130e-04	-4.6648e-04	3.9574e-05	1.8048e-04	1.4669e-05	-1.2382e-04
12	2.1813e-08	2.5161e-08	2.4114e-09	4.4575e-06	-9.2695e-06	-8.0566e-05	7.1942e-05	-1.7943e-04	3.0697e-05	4.8950e-05	-3.4940e-05	1.4811e-04

MODE 6

DOF	$\partial M_6/\partial DV_1$	$\partial M_6/\partial DV_2$	$\partial M_6/\partial DV_3$	$\partial M_6/\partial DV_4$	$\partial M_6/\partial DV_5$	$\partial M_6/\partial DV_6$	$\partial M_6/\partial DV_7$	$\partial M_6/\partial DV_8$	$\partial M_6/\partial DV_9$	$\partial M_6/\partial DV_{10}$	$\partial M_6/\partial DV_{11}$	$\partial M_6/\partial DV_{12}$
1	-1.4819e-13	1.4199e-14	-5.9779e-14	1.0768e-13	1.0120e-13	-4.7982e-15	-1.2499e-04	1.2503e-04	1.2500e-04	-1.2502e-04	-6.3485e-09	-6.0316e-09
2	2.6298e-14	-3.7944e-14	5.9673e-14	-2.9948e-14	-6.2607e-14	3.0412e-15	-7.2176e-05	7.2172e-05	-7.2166e-05	7.2182e-05	-1.4434e-04	1.4433e-04
3	7.2359e-14	-1.0927e-13	1.6871e-13	-8.2688e-14	-1.7704e-13	8.7884e-15	-2.0413e-04	2.0412e-04	-2.0410e-04	2.0415e-04	2.0410e-04	-2.0414e-04
4	2.6451e-10	8.8250e-10	-2.1646e-10	-6.4698e-12	3.8531e-11	-5.1523e-11	4.1789e-05	-1.1376e-04	-1.7288e-04	4.7364e-06	1.2563e-04	1.1448e-04
5	-4.5890e-10	-2.5623e-10	-9.0560e-10	1.8820e-11	3.8167e-11	-2.8621e-11	1.0722e-04	1.7380e-05	-4.5201e-05	-1.3496e-04	1.2721e-04	-7.1647e-05
6	1.1719e-08	-4.4989e-10	4.5291e-10	-2.7249e-12	-3.7226e-11	4.1676e-11	-2.6144e-04	2.6145e-04	-3.1232e-05	1.9793e-04	3.1159e-05	-1.9788e-04
7	5.0486e-10	6.5830e-10	3.4504e-11	-2.4782e-10	1.6544e-11	-5.1523e-11	4.8156e-06	-1.7301e-04	-1.1366e-04	4.1738e-05	1.1449e-04	1.2562e-04
8	-3.0295e-10	5.5256e-11	-4.2384e-11	7.5715e-10	3.7915e-11	2.8617e-11	1.3497e-04	4.5177e-05	-1.7323e-05	-1.0727e-04	7.1672e-05	-1.2722e-04
9	4.1581e-10	3.8361e-10	3.5440e-11	-4.1914e-10	-1.1719e-07	-4.1681e-11	-1.9781e-04	3.1314e-05	7.1285e-04	-7.1297e-04	1.9788e-04	-3.1150e-05
10	9.1998e-12	1.9488e-11	-3.6407e-10	-5.0828e-10	5.1929e-12	-5.2978e-11	4.7391e-05	-1.1934e-04	-1.1932e-04	4.7400e-05	7.1928e-05	7.1948e-05
11	-5.8917e-10	-4.6798e-11	-5.3826e-10	5.8541e-10	5.8077e-11	-2.1964e-15	1.7240e-04	6.3302e-05	-6.3269e-05	-1.7245e-04	8.9801e-05	-8.9791e-05
12	-4.1655e-10	-3.3170e-11	-3.8048e-10	4.1388e-10	4.0934e-11	3.0234e-15	-3.1186e-05	1.9784e-04	-1.9781e-04	3.1168e-05	-7.1284e-04	7.1282e-04

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-4

Eigenvector Derivatives from Expansion Theorem

MODE 1

DOF	$\partial M_1/\partial DV_1$	$\partial M_1/\partial DV_2$	$\partial M_1/\partial DV_3$	$\partial M_1/\partial DV_4$	$\partial M_1/\partial DV_5$	$\partial M_1/\partial DV_6$	$\partial M_1/\partial DV_7$	$\partial M_1/\partial DV_8$	$\partial M_1/\partial DV_9$	$\partial M_1/\partial DV_{10}$	$\partial M_1/\partial DV_{11}$	$\partial M_1/\partial DV_{12}$
1	7.4360e-10	-1.9406e-06	1.7260e-06	7.4620e-10	3.5400e-04	-3.3920e-04	6.1980e-05	-6.8680e-05	-2.1240e-05	4.8220e-05	1.4064e-05	-4.7120e-05
2	2.4200e-10	-8.7280e-07	1.2446e-06	4.1940e-10	1.8718e-04	-2.1320e-04	4.3560e-05	-3.1900e-05	-3.9920e-06	2.6600e-05	1.6402e-05	-2.8440e-05
3	-2.5440e-14	6.2160e-08	-6.2140e-08	-2.1740e-10	-1.9476e-04	1.9476e-04	-2.1780e-09	2.8820e-09	-1.7366e-05	-1.7372e-05	1.7370e-05	1.7364e-05
4	9.4560e-11	-1.3614e-06	1.3426e-06	3.2260e-10	3.0660e-05	-5.7380e-05	2.5380e-05	8.6000e-06	-3.6560e-05	1.6152e-05	3.1040e-05	-1.7732e-05
5	3.2680e-11	-7.6440e-07	7.9680e-07	1.9024e-10	4.8540e-05	-2.3020e-06	-2.4600e-05	-3.4260e-05	-1.4736e-05	1.1160e-05	2.4280e-05	-8.4040e-06
6	8.1840e-11	-2.2420e-07	2.2420e-07	8.3340e-11	3.5440e-05	-3.5460e-05	4.2660e-05	-4.2640e-05	-2.8420e-06	5.2320e-06	2.8380e-06	-5.2300e-06
7	8.7980e-11	6.7120e-07	1.2800e-06	3.0280e-10	1.7296e-05	-5.9440e-05	2.4040e-05	9.7440e-06	-3.6920e-05	1.4586e-05	2.8300e-05	-1.7128e-05
8	5.2260e-12	-6.7040e-07	-4.3460e-07	-5.5420e-11	4.1520e-05	1.3612e-05	-6.7260e-06	-1.2086e-05	-4.2420e-05	1.9934e-05	-7.1980e-06	4.4180e-06
9	-5.8020e-11	2.8880e-07	5.7700e-08	-5.2860e-11	-5.8780e-06	5.1660e-05	-7.2200e-07	7.3440e-06	-4.5440e-05	-2.1180e-05	6.5020e-06	4.2480e-06
10	7.1020e-11	-2.6360e-07	2.4480e-07	9.6440e-11	1.7908e-05	-4.4620e-05	5.5940e-06	-6.1900e-06	-7.9220e-06	4.7380e-06	5.5220e-05	-2.4540e-05
11	2.4640e-11	-1.3256e-06	-9.1680e-07	3.0060e-10	5.8280e-05	5.7640e-06	-1.4490e-05	-2.4160e-05	-2.8120e-05	1.7052e-05	1.0762e-05	-2.6660e-06
12	-2.3820e-11	-5.7700e-08	-2.8880e-07	-5.5160e-11	-5.1660e-05	5.8940e-06	-7.3480e-06	7.2320e-07	-6.5000e-06	-4.2500e-06	4.5440e-05	2.1160e-05

MODE 2

DOF	$\partial M_2/\partial DV_1$	$\partial M_2/\partial DV_2$	$\partial M_2/\partial DV_3$	$\partial M_2/\partial DV_4$	$\partial M_2/\partial DV_5$	$\partial M_2/\partial DV_6$	$\partial M_2/\partial DV_7$	$\partial M_2/\partial DV_8$	$\partial M_2/\partial DV_9$	$\partial M_2/\partial DV_{10}$	$\partial M_2/\partial DV_{11}$	$\partial M_2/\partial DV_{12}$
1	-1.1860e-06	-1.2610e-06	1.2630e-06	3.7060e-07	2.0160e-04	-2.1440e-04	3.4540e-05	-5.3720e-05	-1.3692e-05	4.7840e-05	1.6390e-05	-1.0334e-05
2	-6.8560e-07	2.1860e-06	-2.1840e-06	2.1280e-07	-3.6400e-04	3.5660e-04	-8.1980e-05	7.0880e-05	2.6840e-05	-3.9560e-05	-2.5260e-05	6.1180e-05
3	-2.7720e-06	-5.7820e-07	-5.7820e-07	-8.2180e-07	7.8540e-05	7.8560e-05	-3.9000e-05	-3.9000e-05	-7.7740e-06	-8.0420e-06	-7.7680e-06	-8.0360e-06
4	5.9860e-07	-9.0940e-07	-7.6360e-07	-1.6778e-06	7.5820e-05	6.1360e-06	-1.7012e-05	3.5740e-05	-1.0426e-05	-1.9290e-05	-6.5580e-06	-3.6880e-05
5	3.4560e-07	-3.5700e-07	-6.0900e-07	-9.6880e-07	-3.6680e-05	8.4000e-05	5.1100e-05	-4.0280e-05	-1.5578e-06	-3.1460e-05	-8.2460e-06	-9.8020e-07
6	1.1786e-07	7.8500e-08	7.8580e-08	6.1620e-07	2.8760e-05	2.8760e-05	-5.5180e-05	-5.5180e-05	2.0320e-07	2.1760e-05	2.0440e-07	2.1760e-05
7	5.5100e-07	7.2040e-07	-7.0120e-07	-1.7034e-06	9.1040e-05	-2.0180e-05	-1.6314e-05	3.3400e-05	-1.1298e-05	-1.6048e-05	-5.1260e-06	-4.4120e-05
8	2.1360e-07	-3.2740e-07	3.2540e-07	1.1780e-06	-5.7820e-05	8.6840e-05	1.4716e-05	-8.1120e-06	-8.6140e-06	-9.2660e-05	-4.4780e-07	5.2200e-05
9	-4.1700e-07	2.6840e-07	-4.9780e-07	-4.9680e-07	-7.5000e-05	5.1300e-05	-1.9684e-05	7.1040e-06	-1.0358e-05	6.9840e-05	-5.4660e-06	-6.3000e-06
10	4.6060e-07	-6.9000e-08	7.6700e-08	1.6854e-07	6.5100e-05	-4.5660e-06	9.6800e-06	4.5860e-06	-2.9540e-06	2.3140e-05	-1.3102e-05	-8.8260e-05
11	3.7020e-07	-7.7020e-07	7.8740e-07	-2.0640e-06	-6.0880e-05	1.0776e-04	3.2980e-05	-2.1480e-05	-4.2180e-06	-6.4320e-05	-5.4740e-06	3.2420e-05
12	-4.1700e-07	-4.9780e-07	2.6840e-07	-4.9680e-07	5.1300e-05	-7.5020e-05	7.1080e-06	-1.9684e-05	-5.4700e-06	-6.3020e-06	-1.0352e-05	6.9840e-05

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-4 (Cont.)
Eigenvector Derivatives from Expansion Theorem

MODE 3

DOF	$\partial M_3/\partial DV_1$	$\partial M_3/\partial DV_2$	$\partial M_3/\partial DV_3$	$\partial M_3/\partial DV_4$	$\partial M_3/\partial DV_5$	$\partial M_3/\partial DV_6$	$\partial M_3/\partial DV_7$	$\partial M_3/\partial DV_8$	$\partial M_3/\partial DV_9$	$\partial M_3/\partial DV_{10}$	$\partial M_3/\partial DV_{11}$	$\partial M_3/\partial DV_{12}$
1	-3.2740e-07	-3.6020e-06	4.5780e-06	2.6640e-06	-2.4740e-05	-1.4004e-04	5.1960e-05	-7.5160e-05	-1.0902e-04	-1.4906e-05	1.5586e-04	1.2290e-04
2	-1.8938e-07	7.3680e-06	-6.7980e-06	1.5364e-06	-1.4746e-04	5.2100e-05	-1.1648e-04	1.0352e-04	2.4260e-04	1.5074e-04	-2.1600e-04	-8.8100e-05
3	-1.9906e-06	-5.9380e-07	-6.1120e-07	1.2860e-06	-3.7560e-05	-3.7100e-05	-6.4500e-05	-6.4420e-05	8.7900e-05	2.3860e-05	8.7500e-05	2.3440e-05
4	8.5220e-07	9.4880e-06	-1.1568e-05	-1.0064e-06	-2.9400e-04	2.8380e-04	-1.6278e-04	2.2620e-04	2.0720e-04	2.0060e-04	-2.0820e-04	-2.3040e-04
5	4.9300e-07	-1.8832e-05	1.7616e-05	-5.7420e-07	4.9720e-04	-5.0260e-04	3.5420e-04	-3.1880e-04	-3.5880e-04	-3.8200e-04	3.6000e-04	3.6400e-04
6	-8.8160e-07	5.3540e-07	5.2680e-07	3.3860e-06	-7.7180e-05	-7.7000e-05	-9.1080e-05	-9.1260e-05	8.7540e-05	6.3100e-05	8.7380e-05	6.2820e-05
7	8.1620e-07	1.7410e-05	-1.5472e-05	-1.2526e-06	-3.9880e-04	3.9420e-04	-2.2520e-04	2.8580e-04	2.6840e-04	2.7700e-04	-3.0180e-04	-3.1460e-04
8	3.5240e-07	-2.1200e-05	2.2080e-05	2.3020e-06	5.6480e-04	-5.5940e-04	3.4860e-04	-3.2360e-04	-6.3260e-04	-4.9380e-04	5.9580e-04	4.6480e-04
9	-3.6640e-07	6.0640e-06	-5.9120e-06	-6.9100e-08	-2.4840e-05	1.4394e-04	-1.0416e-04	8.1800e-05	-7.4380e-05	1.9220e-04	-9.8140e-05	-1.1358e-04
10	7.1220e-07	1.1402e-05	-9.6540e-06	1.3604e-06	-2.8780e-04	2.8980e-04	-1.3718e-04	1.8998e-04	3.6460e-04	2.4600e-04	-4.1460e-04	-2.8880e-04
11	5.3260e-07	-2.4440e-05	2.5660e-05	-2.2220e-06	6.2140e-04	-6.2700e-04	4.0800e-04	-3.7020e-04	-5.5760e-04	-5.0560e-04	5.5020e-04	4.8580e-04
12	-3.6580e-07	-5.9100e-06	6.0540e-06	-6.5980e-08	1.4396e-04	-2.4540e-05	8.1500e-05	-1.0440e-04	-9.7640e-05	-1.1366e-04	-7.4080e-05	1.9176e-04

MODE 4

DOF	$\partial M_4/\partial DV_1$	$\partial M_4/\partial DV_2$	$\partial M_4/\partial DV_3$	$\partial M_4/\partial DV_4$	$\partial M_4/\partial DV_5$	$\partial M_4/\partial DV_6$	$\partial M_4/\partial DV_7$	$\partial M_4/\partial DV_8$	$\partial M_4/\partial DV_9$	$\partial M_4/\partial DV_{10}$	$\partial M_4/\partial DV_{11}$	$\partial M_4/\partial DV_{12}$
1	3.3020e-10	-3.0780e-06	1.8180e-06	2.0100e-10	1.5044e-04	-1.0944e-04	1.1456e-04	-1.5626e-04	-7.6980e-05	8.5500e-05	5.6520e-05	-5.1760e-05
2	2.5380e-10	-3.2800e-07	2.5020e-06	8.6340e-11	3.9660e-05	-1.1042e-04	1.1416e-04	-4.2260e-05	-2.0680e-05	1.0222e-05	5.6540e-05	-6.8960e-05
3	-7.9460e-14	2.3340e-05	-2.3320e-05	-6.6820e-09	-5.8060e-04	5.8020e-04	2.8740e-07	9.4680e-07	5.7580e-04	5.7720e-04	-5.7760e-04	-5.7640e-04
4	-1.3838e-09	1.7176e-05	-1.9524e-05	-4.8500e-09	-4.9820e-04	5.0500e-04	-4.4920e-04	5.8120e-04	4.4540e-04	3.2340e-04	-4.6980e-04	-4.1440e-04
5	-7.4440e-10	1.2650e-05	-8.5520e-06	-2.9580e-09	-2.9620e-04	2.8340e-04	-4.1080e-04	1.8430e-04	2.8400e-04	2.9280e-04	-2.4460e-04	-1.3354e-04
6	3.8100e-10	5.0100e-06	-5.0100e-06	-1.9812e-09	-5.1000e-05	5.0960e-05	4.9240e-04	-4.9220e-04	1.1694e-04	2.6160e-04	-1.1730e-04	-2.6140e-04
7	-1.2390e-09	2.0800e-05	-1.7162e-05	-4.1300e-09	-4.3180e-04	4.4700e-04	-4.2300e-04	5.0820e-04	3.8040e-04	2.5380e-04	-4.1100e-04	-3.5980e-04
8	-6.2520e-10	9.1920e-06	-8.5580e-06	-2.6720e-09	-2.1620e-04	1.9870e-04	-1.1058e-04	3.1280e-04	1.5590e-04	6.1940e-05	-2.0900e-04	-1.9982e-04
9	-1.6762e-10	9.1060e-06	-8.7300e-06	-2.2800e-09	-2.9980e-04	2.1760e-04	-8.4080e-05	6.0340e-05	1.5398e-04	3.6320e-04	-2.1020e-04	-2.0480e-04
10	-1.0360e-09	1.5996e-05	-1.8346e-05	-4.4140e-09	-3.9580e-04	4.0260e-04	-5.2420e-04	3.0800e-04	3.8520e-04	3.5340e-04	-3.2600e-04	-1.7978e-04
11	-8.1020e-10	1.0606e-05	-1.3432e-05	-2.3840e-09	-2.8840e-04	2.6600e-04	-2.8340e-04	3.1180e-04	2.5120e-04	2.1240e-04	-2.5240e-04	-1.8872e-04
12	-2.1340e-10	8.7400e-06	-9.1080e-06	-2.2820e-09	-2.1780e-04	2.9960e-04	-6.0000e-05	8.4600e-05	2.0960e-04	2.0520e-04	-1.5454e-04	-3.6300e-04

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-4(Cont.)
Eigenvector Derivatives from Expansion Theorem

MODE 5

DOF	$\partial M_5/\partial DV_1$	$\partial M_5/\partial DV_2$	$\partial M_5/\partial DV_3$	$\partial M_5/\partial DV_4$	$\partial M_5/\partial DV_5$	$\partial M_5/\partial DV_6$	$\partial M_5/\partial DV_7$	$\partial M_5/\partial DV_8$	$\partial M_5/\partial DV_9$	$\partial M_5/\partial DV_{10}$	$\partial M_5/\partial DV_{11}$	$\partial M_5/\partial DV_{12}$
1	-4.5020e-07	-4.1700e-10	4.3500e-10	-4.9860e-07	-3.0600e-05	-7.4440e-06	2.1360e-05	1.0308e-04	1.3760e-05	-8.2260e-05	2.9400e-06	-1.1348e-05
2	-2.5980e-07	6.1760e-10	-8.6300e-10	-2.8800e-07	9.0820e-06	-3.1060e-05	1.0672e-04	-3.4840e-05	-4.5580e-06	3.4380e-05	1.4206e-05	-8.8460e-05
3	-2.4140e-06	-2.1460e-09	-2.7880e-09	-1.8996e-06	3.6560e-05	3.6560e-05	5.5860e-05	5.5760e-05	-6.4280e-05	-6.4720e-06	-6.4340e-05	-6.4800e-06
4	2.3200e-06	3.2840e-09	-5.7600e-09	-1.6284e-06	4.0360e-05	1.5948e-05	1.2890e-04	4.2580e-06	-2.9760e-05	7.2920e-05	-7.0540e-05	-1.6898e-04
5	1.3388e-06	-7.0020e-09	8.2000e-09	-9.4020e-07	-4.8940e-06	3.7400e-05	-6.9600e-05	1.4636e-04	-6.4180e-05	-2.3720e-04	6.3240e-06	1.8180e-04
6	4.4700e-06	-1.4230e-09	-1.8298e-09	-6.6820e-07	-2.2640e-08	-3.1180e-08	-1.2102e-05	-1.2242e-05	-1.2928e-05	6.1560e-06	-1.2942e-05	6.1220e-06
7	2.0860e-06	-2.1920e-09	-6.7600e-09	-1.2668e-06	3.9820e-05	9.4240e-06	2.1360e-04	-1.1722e-04	-1.5894e-05	1.1498e-04	-6.6660e-05	-1.8624e-04
8	8.9660e-07	-7.1480e-09	4.6820e-09	-2.6100e-06	-1.0338e-05	3.5420e-05	-5.6300e-04	5.9940e-04	1.7830e-06	-4.8420e-05	-3.6180e-05	3.8460e-05
9	-2.3260e-09	-1.5370e-10	-3.2520e-09	-4.4540e-07	8.0540e-05	9.2700e-06	1.7946e-04	-7.1940e-05	3.4900e-05	-1.4806e-04	-3.0720e-05	-4.8940e-05
10	1.8188e-06	5.5600e-10	-9.4420e-09	-2.8940e-06	3.5380e-05	1.0968e-05	4.6040e-04	-3.8080e-04	-6.4600e-05	-5.9820e-05	-6.4100e-06	1.5580e-05
11	1.3574e-06	-6.3200e-09	2.1460e-09	2.0840e-07	-9.5540e-06	3.9660e-05	-4.0120e-04	4.6640e-04	-3.9580e-05	-1.8048e-04	-1.4668e-05	1.2382e-04
12	-2.2300e-09	-2.5120e-09	-2.1500e-10	-4.4580e-07	9.2700e-06	8.0560e-05	-7.1940e-05	1.7944e-04	-3.0700e-05	-4.8960e-05	3.4940e-05	-1.4812e-04

MODE 6

DOF	$\partial M_6/\partial DV_1$	$\partial M_6/\partial DV_2$	$\partial M_6/\partial DV_3$	$\partial M_6/\partial DV_4$	$\partial M_6/\partial DV_5$	$\partial M_6/\partial DV_6$	$\partial M_6/\partial DV_7$	$\partial M_6/\partial DV_8$	$\partial M_6/\partial DV_9$	$\partial M_6/\partial DV_{10}$	$\partial M_6/\partial DV_{11}$	$\partial M_6/\partial DV_{12}$
1	-2.9000e-14	2.8340e-15	-1.1434e-14	2.1580e-14	1.0362e-13	-9.5960e-15	-1.2498e-04	1.2502e-04	1.2500e-04	-1.2502e-04	-6.3500e-09	-6.0320e-09
2	5.6320e-15	-6.8360e-15	1.2010e-14	-5.9960e-15	-6.8860e-14	6.0820e-15	-7.2180e-05	7.2180e-05	-7.2160e-05	7.2180e-05	-1.4434e-04	1.4432e-04
3	1.5526e-14	-1.9716e-14	3.3940e-14	-1.6552e-14	-1.9470e-13	1.7576e-14	-2.0420e-04	2.0420e-04	-2.0400e-04	2.0420e-04	2.0420e-04	-2.0420e-04
4	5.2960e-11	1.6528e-10	-4.6680e-11	-1.6288e-12	8.9080e-11	-1.0304e-10	4.1780e-05	-1.1376e-04	-1.7288e-04	4.7360e-06	1.2564e-04	1.1448e-04
5	-9.1780e-11	-4.1200e-11	-1.6506e-10	3.5700e-12	6.8480e-11	-5.7240e-11	1.0722e-04	1.7380e-05	-4.5200e-05	-1.3496e-04	1.2720e-04	-7.1640e-05
6	1.1718e-09	-8.6660e-11	8.6980e-11	-2.7160e-13	-7.8920e-11	8.3360e-11	-2.6140e-04	2.6140e-04	-3.1240e-05	1.9794e-04	3.1160e-05	-1.9788e-04
7	1.0104e-10	1.4288e-10	2.4820e-12	-4.9820e-11	4.3040e-11	-1.0304e-10	4.8160e-06	-1.7300e-04	-1.1366e-04	4.1740e-05	1.1450e-04	1.2562e-04
8	-6.0600e-11	2.1100e-11	-5.9260e-12	1.5022e-10	8.3800e-11	5.7240e-11	1.3496e-04	4.5180e-05	-1.7324e-05	-1.0728e-04	7.1680e-05	-1.2722e-04
9	8.3200e-11	8.0020e-11	3.4820e-12	-8.3560e-11	-1.1718e-07	-8.3360e-11	-1.9780e-04	3.1320e-05	7.1280e-04	-7.1300e-04	1.9788e-04	-3.1160e-05
10	1.8980e-12	3.8980e-12	-8.8400e-11	-1.0284e-10	6.1640e-12	-1.0596e-10	4.7400e-05	-1.1934e-04	-1.1932e-04	4.7400e-05	7.1920e-05	7.1940e-05
11	-1.1784e-10	-4.6780e-12	-1.1276e-10	1.1746e-10	1.1700e-10	-4.3920e-15	1.7240e-04	6.3300e-05	-6.3260e-05	-1.7244e-04	8.9800e-05	-8.9800e-05
12	-8.3320e-11	-3.3220e-12	-7.9700e-11	8.3040e-11	8.2580e-11	7.7780e-15	-3.1180e-05	1.9784e-04	-1.9780e-04	3.1160e-05	-7.1280e-04	7.1280e-04

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-5

Eigenvector Derivatives from Corrected Nelson

MODE 1

DOF	$\partial M_1/\partial DV_1$	$\partial M_1/\partial DV_2$	$\partial M_1/\partial DV_3$	$\partial M_1/\partial DV_4$	$\partial M_1/\partial DV_5$	$\partial M_1/\partial DV_6$	$\partial M_1/\partial DV_7$	$\partial M_1/\partial DV_8$	$\partial M_1/\partial DV_9$	$\partial M_1/\partial DV_{10}$	$\partial M_1/\partial DV_{11}$	$\partial M_1/\partial DV_{12}$
1	7.031e-10	-1.941e-06	1.726e-06	7.437e-10	3.541e-04	-3.392e-04	6.197e-05	-6.868e-05	-2.125e-05	4.821e-05	1.406e-05	-4.712e-05
2	3.123e-10	-8.728e-07	1.245e-06	4.237e-10	1.872e-04	-2.131e-04	4.355e-05	-3.190e-05	-3.991e-06	2.661e-05	1.640e-05	-2.845e-05
3	-2.495e-14	6.217e-08	-6.213e-08	-2.174e-10	-1.948e-04	1.948e-04	-2.177e-09	2.881e-09	-1.737e-05	-1.737e-05	1.737e-05	1.736e-05
4	8.981e-11	-1.361e-06	1.343e-06	3.234e-10	3.066e-05	-5.737e-05	2.539e-05	8.601e-06	-3.655e-05	1.615e-05	3.104e-05	-1.773e-05
5	4.090e-11	-7.643e-07	7.969e-07	1.887e-10	4.853e-05	-2.303e-06	-2.460e-05	-3.427e-05	-1.474e-05	1.116e-05	2.429e-05	-8.404e-06
6	8.183e-11	-2.242e-07	2.242e-07	8.334e-11	3.544e-05	-3.545e-05	4.266e-05	-4.264e-05	-2.842e-06	5.233e-06	2.838e-06	-5.231e-06
7	8.020e-11	6.713e-07	1.280e-06	3.043e-10	1.730e-05	-5.943e-05	2.403e-05	9.744e-06	-3.692e-05	1.459e-05	2.830e-05	-1.713e-05
8	1.622e-11	-6.704e-07	-4.345e-07	-5.827e-11	4.153e-05	1.361e-05	-6.726e-06	-1.209e-05	-4.243e-05	1.993e-05	-7.197e-06	4.418e-06
9	-4.946e-11	2.888e-07	5.769e-08	-5.343e-11	-5.878e-06	5.165e-05	-7.220e-07	7.344e-06	-4.543e-05	-2.118e-05	6.501e-06	4.248e-06
10	6.538e-11	-2.637e-07	2.449e-07	9.818e-11	1.791e-05	-4.462e-05	5.595e-06	-6.190e-06	-7.921e-06	4.739e-06	5.521e-05	-2.454e-05
11	3.688e-11	-1.326e-06	-9.167e-07	2.980e-10	5.828e-05	5.763e-06	-1.449e-05	-2.417e-05	-2.811e-05	1.705e-05	1.076e-05	-2.665e-06
12	-3.237e-11	-5.770e-08	-2.889e-07	-5.458e-11	-5.166e-05	5.894e-06	-7.349e-06	7.232e-07	-6.501e-06	-4.250e-06	4.544e-05	2.117e-05

MODE 2

DOF	$\partial M_2/\partial DV_1$	$\partial M_2/\partial DV_2$	$\partial M_2/\partial DV_3$	$\partial M_2/\partial DV_4$	$\partial M_2/\partial DV_5$	$\partial M_2/\partial DV_6$	$\partial M_2/\partial DV_7$	$\partial M_2/\partial DV_8$	$\partial M_2/\partial DV_9$	$\partial M_2/\partial DV_{10}$	$\partial M_2/\partial DV_{11}$	$\partial M_2/\partial DV_{12}$
1	-1.186e-06	-1.261e-06	1.263e-06	3.706e-07	2.016e-04	-2.145e-04	3.454e-05	-5.373e-05	-1.369e-05	4.784e-05	1.639e-05	-1.033e-05
2	-6.856e-07	2.186e-06	-2.185e-06	2.128e-07	-3.640e-04	3.566e-04	-8.198e-05	7.087e-05	2.683e-05	-3.956e-05	-2.526e-05	6.118e-05
3	-2.772e-06	-5.782e-07	-5.781e-07	-8.218e-07	7.855e-05	7.857e-05	-3.899e-05	-3.900e-05	-7.773e-06	-8.042e-06	-7.769e-06	-8.036e-06
4	5.987e-07	-9.094e-07	-7.636e-07	-1.678e-06	7.581e-05	6.136e-06	-1.701e-05	3.574e-05	-1.043e-05	-1.929e-05	-6.557e-06	-3.688e-05
5	3.455e-07	-3.570e-07	-6.090e-07	-9.688e-07	-3.667e-05	8.400e-05	5.109e-05	-4.028e-05	-1.558e-06	-3.146e-05	-8.246e-06	-9.802e-07
6	1.179e-07	7.849e-08	7.857e-08	6.162e-07	2.876e-05	2.875e-05	-5.517e-05	-5.519e-05	2.032e-07	2.176e-05	2.044e-07	2.176e-05
7	5.510e-07	7.204e-07	-7.013e-07	-1.703e-06	9.103e-05	-2.019e-05	-1.631e-05	3.340e-05	-1.130e-05	-1.605e-05	-5.126e-06	-4.413e-05
8	2.136e-07	-3.275e-07	3.253e-07	1.178e-06	-5.782e-05	8.683e-05	1.472e-05	-8.111e-06	-8.615e-06	-9.265e-05	-4.478e-07	5.220e-05
9	-4.171e-07	2.683e-07	-4.977e-07	-4.969e-07	-7.501e-05	5.131e-05	-1.968e-05	7.103e-06	-1.036e-05	6.984e-05	-5.467e-06	-6.299e-06
10	4.606e-07	-6.899e-08	7.669e-08	1.685e-07	6.510e-05	-4.566e-06	9.680e-06	4.587e-06	-2.953e-06	2.314e-05	-1.310e-05	-8.826e-05
11	3.701e-07	-7.702e-07	7.875e-07	-2.064e-06	-6.088e-05	1.078e-04	3.298e-05	-2.149e-05	-4.218e-06	-6.432e-05	-5.473e-06	3.242e-05
12	-4.169e-07	-4.978e-07	2.683e-07	-4.968e-07	5.130e-05	-7.501e-05	7.109e-06	-1.968e-05	-5.470e-06	-6.302e-06	-1.035e-05	6.985e-05

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-5 (Cont.)
Eigenvector Derivatives from Corrected Nelson

MODE 3

DOF	$\partial M_3/\partial DV_1$	$\partial M_3/\partial DV_2$	$\partial M_3/\partial DV_3$	$\partial M_3/\partial DV_4$	$\partial M_3/\partial DV_5$	$\partial M_3/\partial DV_6$	$\partial M_3/\partial DV_7$	$\partial M_3/\partial DV_8$	$\partial M_3/\partial DV_9$	$\partial M_3/\partial DV_{10}$	$\partial M_3/\partial DV_{11}$	$\partial M_3/\partial DV_{12}$
1	-3.274e-07	-3.602e-06	4.578e-06	2.665e-06	-2.473e-05	-1.400e-04	5.195e-05	-7.515e-05	-1.090e-04	-1.491e-05	1.559e-04	1.229e-04
2	-1.894e-07	7.367e-06	-6.797e-06	1.536e-06	-1.475e-04	5.209e-05	-1.165e-04	1.035e-04	2.426e-04	1.507e-04	-2.161e-04	-8.811e-05
3	-1.991e-06	-5.937e-07	-6.111e-07	1.286e-06	-3.756e-05	-3.711e-05	-6.450e-05	-6.443e-05	8.789e-05	2.385e-05	8.750e-05	2.343e-05
4	8.522e-07	9.488e-06	-1.157e-05	-1.006e-06	-2.939e-04	2.837e-04	-1.628e-04	2.262e-04	2.072e-04	2.005e-04	-2.082e-04	-2.305e-04
5	4.931e-07	-1.883e-05	1.762e-05	-5.742e-07	4.973e-04	-5.026e-04	3.542e-04	-3.189e-04	-3.589e-04	-3.821e-04	3.600e-04	3.640e-04
6	-8.816e-07	5.355e-07	5.269e-07	3.387e-06	-7.719e-05	-7.700e-05	-9.108e-05	-9.126e-05	8.754e-05	6.310e-05	8.739e-05	6.281e-05
7	8.161e-07	1.741e-05	-1.547e-05	-1.253e-06	-3.987e-04	3.941e-04	-2.252e-04	2.857e-04	2.683e-04	2.770e-04	-3.018e-04	-3.146e-04
8	3.523e-07	-2.120e-05	2.208e-05	2.301e-06	5.648e-04	-5.594e-04	3.485e-04	-3.236e-04	-6.327e-04	-4.938e-04	5.959e-04	4.649e-04
9	-3.664e-07	6.063e-06	-5.912e-06	-6.909e-08	-2.484e-05	1.439e-04	-1.042e-04	8.181e-05	-7.439e-05	1.922e-04	-9.815e-05	-1.136e-04
10	7.123e-07	1.140e-05	-9.654e-06	1.360e-06	-2.878e-04	2.897e-04	-1.372e-04	1.900e-04	3.646e-04	2.459e-04	-4.147e-04	-2.888e-04
11	5.326e-07	-2.445e-05	2.565e-05	-2.222e-06	6.214e-04	-6.270e-04	4.079e-04	-3.702e-04	-5.576e-04	-5.056e-04	5.501e-04	4.859e-04
12	-3.659e-07	-5.911e-06	6.055e-06	-6.597e-08	1.440e-04	-2.455e-05	8.150e-05	-1.044e-04	-9.764e-05	-1.137e-04	-7.408e-05	1.918e-04

MODE 4

DOF	$\partial M_4/\partial DV_1$	$\partial M_4/\partial DV_2$	$\partial M_4/\partial DV_3$	$\partial M_4/\partial DV_4$	$\partial M_4/\partial DV_5$	$\partial M_4/\partial DV_6$	$\partial M_4/\partial DV_7$	$\partial M_4/\partial DV_8$	$\partial M_4/\partial DV_9$	$\partial M_4/\partial DV_{10}$	$\partial M_4/\partial DV_{11}$	$\partial M_4/\partial DV_{12}$
1	3.439e-10	-3.077e-06	1.818e-06	1.944e-10	1.504e-04	-1.094e-04	1.146e-04	-1.563e-04	-7.698e-05	8.550e-05	5.652e-05	-5.176e-05
2	2.302e-10	-3.279e-07	2.502e-06	9.760e-11	3.966e-05	-1.104e-04	1.142e-04	-4.226e-05	-2.068e-05	1.022e-05	5.654e-05	-6.897e-05
3	-2.906e-13	2.334e-05	-2.333e-05	-6.682e-09	-5.806e-04	5.801e-04	2.874e-07	9.468e-07	5.759e-04	5.772e-04	-5.775e-04	-5.763e-04
4	-1.372e-09	1.718e-05	-1.952e-05	-4.884e-09	-4.982e-04	5.050e-04	-4.492e-04	5.812e-04	4.454e-04	3.234e-04	-4.698e-04	-4.144e-04
5	-7.651e-10	1.265e-05	-8.552e-06	-2.900e-09	-2.962e-04	2.834e-04	-4.109e-04	1.843e-04	2.840e-04	2.928e-04	-2.447e-04	-1.335e-04
6	3.809e-10	5.010e-06	-5.010e-06	-1.981e-09	-5.100e-05	5.095e-05	4.925e-04	-4.921e-04	1.169e-04	2.615e-04	-1.173e-04	-2.614e-04
7	-1.234e-09	2.081e-05	-1.716e-05	-4.165e-09	-4.318e-04	4.469e-04	-4.230e-04	5.082e-04	3.803e-04	2.538e-04	-4.110e-04	-3.598e-04
8	-5.922e-10	9.191e-06	-8.557e-06	-2.662e-09	-2.162e-04	1.987e-04	-1.106e-04	3.128e-04	1.559e-04	6.194e-05	-2.090e-04	-1.998e-04
9	-1.791e-10	9.107e-06	-8.729e-06	-2.280e-09	-2.998e-04	2.175e-04	-8.408e-05	6.033e-05	1.540e-04	3.631e-04	-2.101e-04	-2.047e-04
10	-1.067e-09	1.599e-05	-1.835e-05	-4.404e-09	-3.957e-04	4.026e-04	-5.243e-04	3.080e-04	3.853e-04	3.534e-04	-3.259e-04	-1.798e-04
11	-7.980e-10	1.061e-05	-1.343e-05	-2.350e-09	-2.883e-04	2.660e-04	-2.834e-04	3.117e-04	2.511e-04	2.124e-04	-2.525e-04	-1.887e-04
12	-2.021e-10	8.740e-06	-9.108e-06	-2.282e-09	-2.178e-04	2.996e-04	-6.000e-05	8.460e-05	2.097e-04	2.052e-04	-1.545e-04	-3.630e-04

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-5 (Cont.)
Eigenvector Derivatives from Corrected Nelson

MODE 5

DOF	$\partial M_5/\partial DV_1$	$\partial M_5/\partial DV_2$	$\partial M_5/\partial DV_3$	$\partial M_5/\partial DV_4$	$\partial M_5/\partial DV_5$	$\partial M_5/\partial DV_6$	$\partial M_5/\partial DV_7$	$\partial M_5/\partial DV_8$	$\partial M_5/\partial DV_9$	$\partial M_5/\partial DV_{10}$	$\partial M_5/\partial DV_{11}$	$\partial M_5/\partial DV_{12}$
1	-4.501e-07	-3.997e-10	4.565e-10	-4.986e-07	-3.061e-05	-7.444e-06	2.136e-05	1.031e-04	1.376e-05	-8.225e-05	2.941e-06	-1.135e-05
2	-2.597e-07	6.324e-10	-8.553e-10	-2.879e-07	9.083e-06	-3.106e-05	1.067e-04	-3.484e-05	-4.558e-06	3.438e-05	1.421e-05	-8.845e-05
3	-2.414e-06	-2.164e-09	-2.806e-09	-1.900e-06	3.656e-05	3.657e-05	5.587e-05	5.577e-05	-6.428e-05	-6.472e-06	-6.435e-05	-6.480e-06
4	2.321e-06	3.236e-09	-5.824e-09	-1.629e-06	4.036e-05	1.595e-05	1.289e-04	4.257e-06	-2.976e-05	7.293e-05	-7.055e-05	-1.690e-04
5	1.339e-06	-7.049e-09	8.182e-09	-9.403e-07	-4.894e-06	3.740e-05	-6.961e-05	1.464e-04	-6.418e-05	-2.372e-04	6.325e-06	1.818e-04
6	4.470e-06	-1.400e-09	-1.807e-09	-6.683e-07	-2.259e-08	-3.114e-08	-1.210e-05	-1.224e-05	-1.293e-05	6.155e-06	-1.294e-05	6.122e-06
7	2.086e-06	-2.237e-09	-6.826e-09	-1.267e-06	3.982e-05	9.424e-06	2.137e-04	-1.172e-04	-1.589e-05	1.150e-04	-6.665e-05	-1.862e-04
8	8.965e-07	-7.151e-09	4.633e-09	-2.611e-06	-1.034e-05	3.542e-05	-5.630e-04	5.994e-04	1.783e-06	-4.841e-05	-3.619e-05	3.846e-05
9	-2.277e-09	-1.798e-10	-3.256e-09	-4.454e-07	8.053e-05	9.270e-06	1.795e-04	-7.194e-05	3.491e-05	-1.481e-04	-3.072e-05	-4.893e-05
10	1.819e-06	4.801e-10	-9.466e-09	-2.895e-06	3.538e-05	1.097e-05	4.604e-04	-3.807e-04	-6.459e-05	-5.982e-05	-6.410e-06	1.558e-05
11	1.357e-06	-6.352e-09	2.110e-09	2.083e-07	-9.554e-06	3.966e-05	-4.013e-04	4.665e-04	-3.957e-05	-1.805e-04	-1.467e-05	1.238e-04
12	-2.182e-09	-2.516e-09	-2.411e-10	-4.457e-07	9.269e-06	8.057e-05	-7.194e-05	1.794e-04	-3.070e-05	-4.895e-05	3.494e-05	-1.481e-04

MODE 6

DOF	$\partial M_6/\partial DV_1$	$\partial M_6/\partial DV_2$	$\partial M_6/\partial DV_3$	$\partial M_6/\partial DV_4$	$\partial M_6/\partial DV_5$	$\partial M_6/\partial DV_6$	$\partial M_6/\partial DV_7$	$\partial M_6/\partial DV_8$	$\partial M_6/\partial DV_9$	$\partial M_6/\partial DV_{10}$	$\partial M_6/\partial DV_{11}$	$\partial M_6/\partial DV_{12}$
1	-1.482e-14	1.420e-15	-5.978e-15	1.077e-14	1.016e-13	-4.798e-15	-1.250e-04	1.250e-04	1.250e-04	-1.250e-04	-6.349e-09	-6.032e-09
2	2.632e-15	-3.794e-15	5.967e-15	-2.995e-15	-6.281e-14	3.041e-15	-7.218e-05	7.217e-05	-7.217e-05	7.218e-05	-1.443e-04	1.443e-04
3	7.242e-15	-1.093e-14	1.687e-14	-8.269e-15	-1.776e-13	8.788e-15	-2.041e-04	2.041e-04	-2.041e-04	2.041e-04	2.041e-04	-2.041e-04
4	2.645e-11	8.825e-11	-2.165e-11	-6.470e-13	3.853e-11	-5.152e-11	4.179e-05	-1.138e-04	-1.729e-04	4.736e-06	1.256e-04	1.145e-04
5	-4.589e-11	-2.562e-11	-9.056e-11	1.882e-12	3.817e-11	-2.862e-11	1.072e-04	1.738e-05	-4.520e-05	-1.350e-04	1.272e-04	-7.165e-05
6	1.172e-09	-4.499e-11	4.529e-11	-2.725e-13	-3.723e-11	4.168e-11	-2.614e-04	2.615e-04	-3.123e-05	1.979e-04	3.116e-05	-1.979e-04
7	5.049e-11	6.583e-11	3.450e-12	-2.478e-11	1.654e-11	-5.152e-11	4.816e-06	-1.730e-04	-1.137e-04	4.174e-05	1.145e-04	1.256e-04
8	-3.029e-11	5.526e-12	-4.238e-12	7.572e-11	3.792e-11	2.862e-11	1.350e-04	4.518e-05	-1.732e-05	-1.073e-04	7.167e-05	-1.272e-04
9	4.158e-11	3.836e-11	3.544e-12	-4.191e-11	-1.172e-07	-4.168e-11	-1.978e-04	3.131e-05	7.129e-04	-7.130e-04	1.979e-04	-3.115e-05
10	9.200e-13	1.949e-12	-3.641e-11	-5.083e-11	5.193e-12	-5.298e-11	4.739e-05	-1.193e-04	-1.193e-04	4.740e-05	7.193e-05	7.195e-05
11	-5.892e-11	-4.680e-12	-5.383e-11	5.854e-11	5.808e-11	-2.196e-15	1.724e-04	6.330e-05	-6.327e-05	-1.724e-04	8.980e-05	-8.979e-05
12	-4.165e-11	-3.317e-12	-3.805e-11	4.139e-11	4.093e-11	3.024e-15	-3.119e-05	1.978e-04	-1.978e-04	3.117e-05	-7.128e-04	7.128e-04

Note: Derivatives of Nodes 13-30 were all identically 0.0000 due to the constraints

Table C-6

Comparison to Derivatives Computed by ASTROS using Nelson's Method

MODE #	MIN AST-FD	MAX AST-FD	MAX (AST-FD)/FD	MIN (AST-FD)/FD	MEAN (AST-FD)/FD	STDV (AST-FD)/FD	SIGN Dif'nt AST<=>AN	SIGN Dif'nt AST<=>FD	Sign ANAL <> AST=FD	# AST 0's	# FD 0's
1	0	2.45	86	0	-1.2	12.3	13	4	13	0	4
2	0	2.8	2.3	0	0.89	.5	16	0	16	0	0
3	0	33.7	21.3	0.017	1.57	5.7	135	144	0	0	0
4	0	32.6	98.8	0.86	-0.55	12.9	127	143	1	0	0
5	0	6.8	75.7	0.01	-1.4	11.4	136	144	0	0	0
6	0	6.22	96.1	0	-3.7	17.4	4	32	4	0	30

Table C-6 (Cont.)
Comparison to Derivatives Computed by Original FRAME using Nelson's Method

MODE #	MIN ANL-FD	MAX ANL-FD	MIN (ANL-FD)/FD	MAX (ANL-FD)/FD	MEAN (ANL-FD)/FD	STDV (ANL-FD)/FD	SIGN Dif'nt ANL<=>AST	SIGN Dif'nt ANL<=>FD	Sign ANAL <> ANL=FD	# ANL 0's	# FD 0's
1	0	267	0	2313	282	2664	13	17	13	0	4
2	0.0018	293.6	0.4	1169	-19.6	135	16	16	16	0	0
3	0.03	368	9.16	167	-40.5	37.8	135	9	0	0	0
4	0	547	0.0393	2239	-7.3	266	127	18	1	0	0
5	1.1e-4	287	6.6	1010	94.1	1168	136	8	0	0	0
6	0	392	0	1047	-9.5	124	4	36	4	0	30

Table C-6(Cont.)
Comparison to Derivatives Computed by Corrected FRAME using Nelson's Method

MODE #	MIN ANL-FD	MAX ANL-FD	MIN (ANL-FD)/FD	MAX (ANL-FD)/FD	MEAN (ANL-FD)/FD	STDV (ANL-FD)/FD	SIGN Dif'nt ANL<=>AST	SIGN Dif'nt ANL<=>FD	Sign ANAL <> ANL=FD	# ANL 0's	# FD 0's
1	0	2.5	0	85.9	-1.2	12.2	0	0	0	0	4
2	0	2.8	0	2.3	0.89	0.5	0	0	0	0	0
3	0	33.7	0	21.3	1.6	5.7	144	0	0	0	0
4	0	32.6	0.1	104.0	-6.1	13.4	144	1	1	0	0
5	0	6.8	0	75.7	-1.42	11.4	144	0	0	0	0
6	0	6.2	0	104.0	-3.8	18.0	0	0	0	0	30

Table C-6 (Cont.)
Comparison to Derivatives Computed by Original FRAME using Expansion Theorem

MODE #	MIN EXP-FD	MAX EXP-FD	MIN (EXP-FD)/FD	MAX (EXP-FD)/FD	MEAN (EXP-FD)/FD	STDV (EXP-FD)/FD	SIGN Dif'nt EXP<=>AST	SIGN Dif'nt EXP<=>FD	Sign ANAL <> EXP=FD	# EXP 0's	# FD 0's
1	0	176	0	93/0	**-49.2**	10.5	0	4	0	0	4
2	0.03	180.6	49.0	51.1	**-49.6**	0.25	0	0	0	0	0
3	0.03	345.1	39.3	58.8	**-49.2**	2.84	144	0	0	0	0
4	0	321.3	40.7	102.1	**-50.6**	7.3	144	1	1	0	0
5	0	296.8	36.3	87.8	**-50.7**	5.8	144	0	0	0	0
6	0	350.3	0	102.0	-30.7	27.1	0	32	0	0	30

Table C-6 (Cont.)
Comparison to Derivatives Computed by Corrected FRAME using Expansion Theorem

MODE #	MIN EXP-FD	MAX EXP-FD	MIN (EXP-FD)/FD	MAX (EXP-FD)/FD	MEAN (EXP-FD)/FD	STDV (EXP-FD)/FD	SIGN Dif'nt EXP<=>AST	SIGN Dif'nt EXP<=>FD	Sign ANAL <> EXP=FD	# EXP 0's	# FD 0's
1	0	2.5	0	85.9	-1.21	12.8	0	4	0	0	4
2	0	2.8	0.04	2.6	0.90	0.5	0	0	0	0	0
3	0	33.7	0	21.3	1.57	5.67	144	0	0	0	0
4	0	32.6	0.06	104.3	-1.13	14.6	144	1	1	0	0
5	0	6.8	0.01	75.7	-1.5	11.6	144	0	0	0	0
6	0	6.2	0	180.2	17.7	45.0	0	32	0	0	30

Appendix D

D.1: Nodal Displacement Vectors: The final coordinates of the displacements at the four unpinned nodes (nodes 1-4) as calculated by Nelson's Method, Modal Expansion, and Finite Difference

D.2: Mode Tracking Results: Results computed from FRAME as equipped with Nelson's Method and MCORC.

D.3: Final Design Variables: Final Design Variables and parameter comparisons as computed by Nelson's Method, Modal Expansion, Finite Difference, and Semi-Analytic Methods.

D.4: Final natural frequencies: Final frequecies as computed by Nelson's Method, Modal Expansion, Finite Difference, and Semi-Analytic Methods.

Table D-1

Nodal Displacement Vectors

MD	DOF*	CORRECTED NELSON			MODAL EXPANSION			FINITE DIFFERENCE		
		X	Y	Z	X	Y	Z	X	Y	Z
1	1	-4.3221e-04	4.0112e-04	-3.1763e-05	-4.3221e-04	4.0112e-04	-3.1763e-05	-4.3236e-04	4.0105e-04	-3.1722e-05
	2	-1.4208e-04	4.3087e-05	-5.2235e-05	-1.4208e-04	4.3087e-05	-5.2235e-05	-1.4186e-04	4.3291e-05	-5.2338e-05
	3	-2.1544e-04	1.5144e-04	6.5559e-05	-2.1544e-04	1.5144e-04	6.5559e-05	-2.1541e-04	1.5148e-04	6.5645e-05
	4	-8.9429e-05	1.9434e-04	-8.4238e-05	-8.9429e-05	1.9434e-04	-8.4238e-05	-8.9322e-05	1.9433e-04	-8.4128e-05
2	1	4.1342e-04	4.5198e-04	2.1317e-05	4.1342e-04	4.5198e-04	2.1317e-05	4.1348e-04	4.5201e-04	2.1496e-05
	2	2.0465e-04	1.3626e-04	1.2219e-04	2.0465e-04	1.3626e-04	1.2219e-04	2.0463e-04	1.3615e-04	1.2197e-04
	3	8.6903e-05	8.0749e-05	-8.6636e-06	8.6903e-05	8.0749e-05	-8.6636e-06	8.6800e-05	8.0685e-05	-8.6269e-06
	4	3.2126e-05	1.8188e-04	-2.6564e-05	3.2126e-05	1.8188e-04	-2.6564e-05	3.2075e-05	1.8199e-04	-2.6532e-05
3	1	-1.2863e-04	-1.8756e-04	5.2095e-05	-1.2863e-04	-1.8756e-04	5.2095e-05	-1.2844e-04	-1.8795e-04	5.1463e-05
	2	2.1169e-04	3.4085e-04	-2.3151e-04	2.1169e-04	3.4085e-04	-2.3151e-04	2.1200e-04	3.4088e-04	-2.3170e-04
	3	8.5704e-05	1.2667e-04	4.3662e-05	8.5704e-05	1.2667e-04	4.3662e-05	8.5937e-05	1.2666e-04	4.3398e-05
	4	2.2424e-04	3.1438e-04	2.3782e-04	2.2424e-04	3.1438e-04	2.3782e-04	2.2388e-04	3.1448e-04	2.3741e-04
4	1	2.4337e-04	-1.6851e-04	-2.4159e-04	2.4337e-04	-1.6851e-04	-2.4159e-04	2.4330e-04	-1.6821e-04	-2.4168e-04
	2	-2.6459e-04	9.7964e-05	-5.4374e-05	-2.6459e-04	9.7964e-05	-5.4374e-05	-2.6463e-04	9.7953e-05	-5.4553e-05
	3	-3.0041e-04	1.2807e-04	-3.1565e-04	-3.0041e-04	1.2807e-04	-3.1565e-04	-3.0052e-04	1.2788e-04	-3.1588e-04
	4	-8.8420e-05	2.3653e-04	-3.7485e-05	-8.8420e-05	2.3653e-04	-3.7485e-05	-8.8603e-05	2.3618e-04	-3.7715e-05
5	1	4.4298e-05	-7.2405e-05	4.2324e-04	4.4298e-05	-7.2405e-05	4.2324e-04	-4.4399e-05	7.2718e-05	-4.2360e-04
	2	-1.6363e-04	1.0177e-05	3.7482e-04	-1.6363e-04	1.0177e-05	3.7482e-04	1.6338e-04	-1.0657e-05	-3.7405e-04
	3	-2.5725e-04	5.7204e-05	9.6595e-05	-2.5725e-04	5.7204e-05	9.6595e-05	2.5759e-04	-5.7712e-05	-9.6582e-05
	4	8.7750e-05	7.4936e-05	2.3279e-04	8.7750e-05	7.4936e-05	2.3279e-04	-8.7681e-05	-7.5100e-05	-2.3290e-04
6	1	2.1751e-05	1.7302e-04	-6.7153e-05	2.1751e-05	1.7302e-04	-6.7153e-05	2.1814e-05	1.7297e-04	-6.6658e-05
	2	-2.0522e-04	2.8899e-04	-9.2604e-05	-2.0522e-04	2.8899e-04	-9.2604e-05	-2.0517e-04	2.8897e-04	-9.2989e-05
	3	-1.2526e-04	-4.4810e-04	-2.5718e-05	-1.2526e-04	-4.4810e-04	-2.5718e-05	-1.2524e-04	-4.4818e-04	-2.5390e-05
	4	2.7334e-04	-1.7125e-04	9.8437e-05	2.7334e-04	-1.7125e-04	9.8437e-05	2.7340e-04	-1.7105e-04	9.8584e-05

* DOFs 5-10 all had a value 0.0000 and have been excluded in the interest of space

Table D-1
Nodal Displacement Vectors (Cont.)

MD	DOF*	CORRECTED NELSON			MODAL EXPANSION			FINITE DIFFERENCE		
		X	Y	Z	X	Y	Z	X	Y	Z
7	1	7.1327e-05	-9.4896e-05	2.0644e-04	7.1327e-05	-9.4896e-05	2.0644e-04	7.1801e-05	-9.4750e-05	2.0639e-04
	2	-3.3829e-05	4.3509e-04	-8.1987e-05	-3.3829e-05	4.3509e-04	-8.1987e-05	-3.4168e-05	4.3446e-04	-8.3071e-05
	3	-3.5319e-05	5.5183e-05	1.7915e-04	-3.5319e-05	5.5183e-05	1.7915e-04	-3.5093e-05	5.5344e-05	1.7987e-04
	4	-3.2012e-04	-1.6701e-04	-2.8172e-04	-3.2012e-04	-1.6701e-04	-2.8172e-04	-3.2010e-04	-1.6706e-04	-2.8183e-04
8	1	1.7587e-04	-1.3351e-05	5.5075e-05	1.7587e-04	-1.3351e-05	5.5075e-05	1.7662e-04	-1.3001e-05	5.4574e-05
	2	1.5686e-04	-2.3941e-04	-3.2688e-04	1.5686e-04	-2.3941e-04	-3.2688e-04	1.5432e-04	-2.4066e-04	-3.2847e-04
	3	-3.4689e-04	-1.7453e-04	3.2282e-04	-3.4689e-04	-1.7453e-04	3.2282e-04	-3.4515e-04	-1.7270e-04	3.2418e-04
	4	-7.6717e-05	1.2505e-04	2.3555e-05	-7.6717e-05	1.2505e-04	2.3555e-05	-7.6089e-05	1.2482e-04	2.4843e-05
9	1	1.0676e-04	-4.7632e-05	-2.4035e-04	1.0676e-04	-4.7632e-05	-2.4035e-04	1.0603e-04	-4.7466e-05	-2.4050e-04
	2	-3.5514e-04	-2.5797e-06	6.7912e-05	-3.5514e-04	-2.5797e-06	6.7912e-05	-3.5612e-04	-1.9570e-06	6.9450e-05
	3	2.2827e-04	7.1920e-05	4.6757e-04	2.2827e-04	7.1920e-05	4.6757e-04	2.2981e-04	7.2548e-05	4.6578e-04
	4	6.4943e-05	1.3201e-04	1.6596e-05	6.4943e-05	1.3201e-04	1.6596e-05	6.5302e-05	1.3202e-04	1.7020e-05
10	1	-1.8220e-05	6.0370e-05	3.0141e-05	-1.8220e-05	6.0370e-05	3.0141e-05	1.8026e-05	-6.0598e-05	-3.3344e-05
	2	-8.3389e-05	1.9819e-05	-2.8084e-05	-8.3389e-05	1.9819e-05	-2.8084e-05	8.5333e-05	-1.8773e-05	2.9935e-05
	3	1.9081e-04	-2.2866e-04	-7.6783e-05	1.9081e-04	-2.2866e-04	-7.6783e-05	-1.9258e-04	2.2914e-04	7.8366e-05
	4	-4.7362e-04	1.3271e-04	3.8790e-04	-4.7362e-04	1.3271e-04	3.8790e-04	4.7283e-04	-1.3411e-04	-3.8610e-04
11	1	1.2371e-04	1.4455e-04	1.2784e-04	1.2371e-04	1.4455e-04	1.2784e-04	-1.2274e-04	-1.4454e-04	-1.2802e-04
	2	-1.9185e-04	-6.7796e-05	-3.4057e-04	-1.9185e-04	-6.7796e-05	-3.4057e-04	1.9271e-04	6.7509e-05	3.3981e-04
	3	3.5075e-05	3.5297e-04	-4.5109e-05	3.5075e-05	3.5297e-04	-4.5109e-05	-3.7291e-05	-3.5377e-04	4.7053e-05
	4	4.8294e-05	-3.0659e-04	2.5668e-04	4.8294e-05	-3.0659e-04	2.5668e-04	-4.9885e-05	3.0640e-04	-2.5565e-04
12	1	-3.1927e-05	-8.2476e-06	-3.6488e-04	-3.1927e-05	-8.2476e-06	-3.6488e-04	-3.1256e-05	-7.3666e-06	-3.6424e-04
	2	2.4206e-04	1.3624e-04	2.0434e-04	2.4206e-04	1.3624e-04	2.0434e-04	2.4087e-04	1.3634e-04	2.0292e-04
	3	-2.3504e-04	1.3593e-04	1.4573e-04	-2.3504e-04	1.3593e-04	1.4573e-04	-2.3383e-04	1.3470e-04	1.4569e-04
	4	-1.0598e-04	-2.5859e-04	2.7167e-04	-1.0598e-04	-2.5859e-04	2.7167e-04	-1.0961e-04	-2.5836e-04	2.7514e-04

* DOFs 5-10 all had a value 0.0000 and have been excluded in the interest of space

Table D-2

CORC Iteration Histories

ITERATION 0

↓Former Mode **New Mode →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
3	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
5	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

ITERATION 1

↓Former Mode **New Mode →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.9910	0.0667	-0.0605	-0.0383	-0.0115	0.0426	-0.0717	0.0365	0.0007	0.0008	0.0003	0.0007
2	-0.0755	0.9834	-0.1329	-0.0415	0.0204	0.0104	0.0454	0.0719	-0.0050	0.0000	0.0023	0.0026
3	0.0408	0.1411	0.9582	-0.0968	0.0997	0.0798	-0.1057	-0.1482	-0.0373	-0.0041	0.0106	-0.0028
4	0.0440	0.0476	0.0721	0.9707	0.1185	0.1375	0.1166	-0.0377	-0.0051	-0.0176	0.0021	-0.0003
5	0.0008	-0.0315	-0.1254	-0.0986	0.9799	-0.0296	-0.0754	-0.0813	0.0072	0.0030	-0.0096	-0.0029
6	-0.0334	-0.0334	-0.0806	-0.1535	0.0102	0.9622	0.1884	-0.0759	0.0133	0.0204	-0.0034	-0.0001
7	0.0857	0.0010	0.0279	-0.0985	0.0163	-0.2134	0.8587	-0.4458	-0.0046	-0.0141	0.0004	-0.0016
8	0.0147	-0.0630	0.1758	-0.0421	0.1216	-0.0117	0.4352	0.8710	0.0100	0.0038	0.0012	0.0061
9	0.0013	0.0115	0.0354	0.0035	-0.0036	-0.0094	-0.0056	-0.0144	0.9984	-0.0165	0.0342	0.0046
10	0.0020	0.0025	0.0074	0.0190	-0.0009	-0.0199	0.0084	-0.0094	0.0155	0.9993	0.0129	0.0017
11	0.0009	0.0047	0.0130	0.0026	-0.0080	-0.0025	0.0000	-0.0002	0.0339	0.0122	-0.9992	0.0041
12	-0.0004	-0.0019	0.0015	-0.0001	0.0025	0.0001	-0.0018	-0.0068	-0.0049	-0.0017	0.0039	0.9999

Table D-2 (Cont.)
CORC Iteration Histories

ITERATION 2

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-0.8964	0.2901	-0.0164	0.2756	0.0140	-0.1555	-0.0929	0.0092	0.0087	0.0545	-0.0039	0.0000
2	0.2068	0.8847	0.0935	-0.2365	-0.3100	0.0221	-0.0244	0.0145	-0.0956	0.0326	0.0377	0.0296
3	0.1303	0.0962	0.7754	0.5042	0.1971	0.2168	0.0513	-0.1579	0.0478	-0.0093	0.0198	-0.0469
4	-0.3188	-0.0781	0.2649	-0.6438	0.1909	0.6014	-0.0757	-0.0314	-0.0070	0.0521	0.0012	0.0010
5	0.1251	0.3289	-0.3086	0.0383	0.8703	0.0262	0.0897	0.0773	0.0660	-0.0034	-0.0537	-0.0058
6	-0.0330	-0.0536	0.4352	-0.4367	0.2210	-0.7484	0.0299	-0.0555	0.0068	-0.0516	-0.0185	-0.0031
7	-0.1219	-0.0013	0.0628	-0.0181	-0.0683	0.0584	0.8495	0.3579	-0.1835	-0.2893	-0.0241	0.0710
8	0.0579	-0.0509	0.1731	0.0623	0.0372	-0.0143	-0.3477	0.8630	-0.1573	0.1898	-0.0913	0.1546
9	0.0035	-0.0577	-0.0321	0.0482	0.1144	-0.0181	-0.0399	-0.2218	-0.9328	0.1610	0.1683	0.0680
10	0.0130	-0.0247	0.0037	-0.0168	-0.0329	-0.0402	0.3590	-0.0397	0.1418	0.9182	-0.0494	0.0110
11	-0.0051	-0.0138	-0.0051	-0.0093	0.0401	-0.0182	0.0186	0.1333	0.1539	0.0275	0.9773	-0.0065
12	0.0002	-0.0079	0.0038	0.0190	0.0161	0.0065	-0.0039	-0.1528	0.1077	-0.0319	0.0109	0.9814

ITERATION 3

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-0.9958	0.0426	0.0424	-0.0282	-0.0301	-0.0103	0.0491	0.0037	-0.0092	0.0118	0.0167	0.0004
2	0.0343	0.9882	-0.0979	0.0740	0.0463	-0.0065	-0.0054	-0.0531	-0.0336	0.0187	-0.0252	-0.0107
3	0.0494	0.0978	0.9901	-0.0475	0.0217	-0.0334	0.0348	-0.0115	0.0158	0.0465	-0.0039	0.0057
4	0.0285	0.0597	-0.0617	-0.9811	0.1130	-0.1110	0.0526	-0.0020	-0.0321	-0.0025	0.0145	-0.0126
5	-0.0320	-0.0518	-0.0049	0.1052	0.9786	0.0893	0.0495	-0.0237	-0.0510	-0.0524	0.1037	-0.0197
6	-0.0026	-0.0291	-0.0233	0.1169	0.0837	-0.9761	-0.0823	-0.0333	-0.0945	0.0149	-0.0882	0.0179
7	0.0382	-0.0215	-0.0322	0.0498	-0.0342	-0.0430	0.9271	-0.2778	-0.0130	-0.0552	-0.2238	0.0398
8	0.0055	0.0367	-0.0238	0.0057	0.0125	-0.1337	0.1210	0.0578	0.8879	0.0051	0.3679	-0.1959
9	-0.0208	-0.0522	-0.0053	-0.0260	0.0136	0.0533	-0.2795	-0.9131	0.1293	0.1940	-0.1164	-0.1159
10	0.0302	-0.0104	-0.0268	0.0275	-0.0663	-0.0364	0.1753	-0.0531	-0.3214	0.6211	0.6838	-0.0401
11	-0.0072	-0.0133	-0.0385	-0.0096	0.1137	0.0414	-0.0044	0.2547	0.2292	0.7530	-0.5372	0.1027
12	0.0007	-0.0137	0.0086	0.0136	-0.0110	0.0062	0.0365	0.1161	-0.1582	0.0292	-0.1590	-0.9661

Table D-2 (Cont.)
CORC Iteration Histories

ITERATION 4

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-0.9958	0.0426	0.0424	-0.0282	-0.0301	-0.0103	0.0491	0.0037	-0.0092	0.0118	0.0167	0.0004
2	0.0343	0.9882	-0.0979	0.0740	0.0463	-0.0065	-0.0054	-0.0531	-0.0336	0.0187	-0.0252	-0.0107
3	0.0494	0.0978	0.9901	-0.0475	0.0217	-0.0334	0.0348	-0.0115	0.0158	0.0465	-0.0039	0.0057
4	0.0285	0.0597	-0.0617	-0.9811	0.1130	-0.1110	0.0526	-0.0020	-0.0321	-0.0025	0.0145	-0.0126
5	-0.0320	-0.0518	-0.0049	0.1052	0.9786	0.0893	0.0495	-0.0237	-0.0510	-0.0524	0.1037	-0.0197
6	-0.0026	-0.0291	-0.0233	0.1169	0.0837	-0.9761	-0.0823	-0.0333	-0.0945	0.0149	-0.0882	0.0179
7	0.0382	-0.0215	-0.0322	0.0498	-0.0342	-0.0430	0.9271	-0.2778	-0.0130	-0.0552	-0.2238	0.0398
8	0.0055	0.0367	-0.0238	0.0057	0.0125	-0.1337	0.1210	0.0578	0.8879	0.0051	0.3679	-0.1959
9	-0.0208	-0.0522	-0.0053	-0.0260	0.0136	0.0533	-0.2795	-0.9131	0.1293	0.1940	-0.1164	-0.1159
10	0.0302	-0.0104	-0.0268	0.0275	-0.0663	-0.0364	0.1753	-0.0531	-0.3214	0.6211	0.6838	-0.0401
11	-0.0072	-0.0133	-0.0385	-0.0096	0.1137	0.0414	-0.0044	0.2547	0.2292	0.7530	-0.5372	0.1027
12	0.0007	-0.0137	0.0086	0.0136	-0.0110	0.0062	0.0365	0.1161	-0.1582	0.0292	-0.1590	-0.9661

ITERATION 5

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-0.9997	0.0031	0.0016	0.0067	-0.0094	0.0081	0.0191	0.0002	0.0032	0.0024	-0.0011	-0.0010
2	0.0035	0.9996	0.0085	0.0183	0.0049	0.0110	0.0075	0.0099	0.0047	0.0010	0.0058	0.0050
3	0.0021	-0.0097	0.9981	0.0565	-0.0055	0.0009	0.0028	0.0135	-0.0039	0.0127	-0.0021	-0.0015
4	-0.0069	0.0186	0.0570	-0.9963	-0.0015	-0.0549	0.0020	-0.0212	0.0091	-0.0050	-0.0089	-0.0006
5	-0.0094	-0.0045	0.0053	0.0008	0.9990	-0.0307	0.0093	-0.0026	0.0093	0.0080	-0.0219	-0.0129
6	-0.0079	0.0102	-0.0013	0.0557	-0.0295	-0.9958	-0.0234	-0.0291	0.0207	-0.0317	0.0321	0.0204
7	0.0185	-0.0082	-0.0050	-0.0004	-0.0089	-0.0282	0.9894	0.1272	-0.0571	0.0144	0.0093	0.0062
8	-0.0049	0.0035	-0.0073	-0.0131	0.0075	-0.0307	-0.0904	0.2649	-0.9547	0.0778	-0.0504	-0.0163
9	-0.0014	-0.0095	-0.0102	-0.0181	0.0014	-0.0189	-0.1075	0.9534	0.2712	-0.0251	0.0573	-0.0345
10	0.0017	0.0052	-0.0043	0.0098	-0.0241	-0.0337	0.0054	0.0419	0.0725	0.1024	-0.9903	0.0020
11	0.0021	-0.0014	-0.0117	-0.0045	-0.0067	-0.0259	-0.0116	-0.0036	0.0758	0.9906	0.1088	0.0138
12	0.0012	0.0054	-0.0013	0.0026	-0.0139	-0.0193	0.0106	-0.0370	0.0074	0.0128	-0.0006	-0.9988

Table D-2 (Cont.)
CORC Iteration Histories

ITERATION 6

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-0.9969	-0.0213	-0.0346	0.0036	-0.0311	0.0154	0.0572	-0.0046	0.0081	0.0020	-0.0016	-0.0036
2	-0.0216	0.9990	0.0065	0.0050	0.0230	0.0072	0.0120	0.0164	0.0024	0.0147	0.0075	0.0150
3	-0.0376	-0.0075	0.9879	0.1305	0.0215	-0.0501	-0.0442	0.0181	0.0097	-0.0152	-0.0065	-0.0049
4	-0.0028	0.0063	0.1315	-0.9827	-0.0897	-0.0649	0.0609	-0.0240	0.0047	0.0068	-0.0157	-0.0049
5	-0.0287	-0.0209	-0.0175	-0.0801	0.9837	-0.1249	0.0450	-0.0172	0.0519	-0.0372	-0.0323	-0.0358
6	-0.0136	0.0088	-0.0573	0.0632	-0.1125	-0.9828	-0.0687	-0.0285	-0.0019	0.0540	0.0611	0.0422
7	0.0535	-0.0137	0.0299	0.0651	-0.0388	-0.0668	0.9633	0.2353	-0.0292	-0.0133	0.0062	0.0369
8	-0.0152	0.0056	-0.0078	-0.0185	0.0278	-0.0324	-0.0899	0.2425	-0.8514	-0.4428	-0.0967	0.0161
9	-0.0138	-0.0145	-0.0236	-0.0376	0.0118	-0.0101	-0.2144	0.9382	0.2208	0.1319	0.0096	-0.0701
10	0.0033	0.0083	-0.0132	0.0213	-0.0455	-0.0541	0.0006	-0.0059	0.1603	-0.0947	-0.9741	0.1035
11	0.0026	0.0120	-0.0172	-0.0026	-0.0547	-0.0316	-0.0075	0.0161	0.4434	-0.8786	0.1634	0.0044
12	0.0070	0.0179	-0.0059	0.0169	-0.0464	-0.0449	0.0450	-0.0546	-0.0139	-0.0270	-0.0992	0.9896

ITERATION 7

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-0.9965	-0.0732	-0.0010	-0.0023	-0.0311	-0.0161	0.0203	-0.0022	0.0027	0.0044	-0.0034	0.0002
2	-0.0731	0.9962	-0.0343	0.0072	-0.0220	0.0100	-0.0202	-0.0008	-0.0092	0.0039	0.0000	0.0009
3	-0.0006	0.0348	0.9773	-0.1968	-0.0406	-0.0497	0.0132	-0.0103	-0.0047	0.0214	-0.0029	0.0105
4	0.0085	-0.0030	-0.2001	-0.9616	-0.1779	0.0373	0.0407	-0.0002	-0.0117	0.0204	-0.0020	0.0038
5	-0.0333	0.0197	0.0068	-0.1829	0.9798	0.0544	-0.0379	0.0147	0.0168	-0.0062	-0.0037	-0.0018
6	0.0134	0.0100	-0.0560	-0.0378	0.0465	-0.9925	-0.0424	-0.0349	0.0675	0.0108	0.0073	-0.0104
7	0.0178	0.0230	-0.0073	0.0327	0.0457	-0.0362	0.9942	-0.0116	0.0685	-0.0182	-0.0005	0.0327
8	-0.0006	0.0076	0.0114	-0.0130	-0.0297	0.0499	-0.0599	0.3215	0.9201	-0.2048	-0.0290	0.0003
9	0.0008	0.0013	-0.0038	-0.0046	0.0033	0.0554	-0.0308	-0.9453	0.3061	-0.0881	0.0094	-0.0289
10	0.0034	0.0004	-0.0004	0.0014	-0.0025	-0.0077	0.0079	-0.0114	-0.0342	-0.0377	-0.9750	-0.2158
11	0.0047	-0.0021	-0.0141	0.0210	0.0053	0.0266	0.0041	-0.0165	0.2206	0.9725	-0.0335	-0.0531
12	-0.0007	0.0019	0.0110	-0.0054	-0.0018	0.0076	0.0331	0.0315	-0.0118	-0.0425	0.2175	-0.9739

Table D-2 (Cont.)
CORC Iteration Histories

ITERATION 8

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-0.9998	-0.0016	-0.0069	-0.0051	0.0051	0.0023	-0.0125	-0.0086	0.0045	-0.0009	0.0003	-0.0001
2	-0.0017	0.9999	0.0018	0.0035	-0.0116	-0.0006	0.0012	-0.0033	-0.0068	0.0071	0.0008	0.0011
3	-0.0070	-0.0019	0.9993	0.0235	0.0061	0.0242	0.0072	-0.0058	-0.0028	0.0015	0.0009	-0.0025
4	0.0053	0.0038	0.0230	-0.9983	0.0325	0.0201	-0.0275	0.0106	-0.0184	0.0054	-0.0069	-0.0106
5	0.0050	0.0116	-0.0070	0.0318	0.9990	0.0034	0.0092	-0.0135	0.0153	-0.0137	-0.0032	-0.0036
6	-0.0024	-0.0003	0.0248	-0.0202	0.0039	-0.9988	0.0059	0.0124	0.0196	-0.0217	0.0069	0.0144
7	-0.0127	-0.0012	-0.0066	-0.0273	-0.0077	0.0067	0.9987	0.0332	-0.0162	-0.0085	0.0015	0.0084
8	0.0040	0.0077	0.0031	-0.0185	-0.0160	0.0227	0.0130	0.0431	0.9909	-0.1200	-0.0193	-0.0136
9	0.0084	-0.0031	-0.0052	-0.0132	-0.0143	-0.0110	0.0334	-0.9981	0.0408	-0.0170	0.0080	-0.0028
10	-0.0004	0.0005	0.0015	0.0036	-0.0023	-0.0025	-0.0010	-0.0089	-0.0167	-0.0198	-0.9581	0.2850
11	-0.0003	-0.0062	-0.0009	0.0023	0.0116	-0.0188	0.0108	-0.0117	0.1214	0.9915	-0.0120	0.0351
12	0.0003	-0.0008	0.0021	-0.0118	0.0041	0.0170	-0.0089	0.0003	0.0144	-0.0318	0.2852	0.9576

ITERATION 9

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-0.9997	0.0180	-0.0075	-0.0145	0.0004	0.0037	-0.0014	-0.0018	-0.0010	0.0012	0.0009	0.0001
2	0.0179	0.9995	-0.0068	0.0080	0.0133	-0.0138	0.0046	-0.0004	0.0073	0.0050	-0.0017	0.0006
3	-0.0078	0.0062	0.9992	0.0264	0.0163	-0.0174	-0.0099	-0.0039	0.0074	0.0056	0.0004	0.0026
4	0.0143	0.0062	0.0245	-0.9950	0.0894	-0.0226	0.0080	0.0141	0.0147	0.0090	0.0007	0.0003
5	-0.0009	-0.0136	-0.0182	0.0882	0.9954	0.0118	-0.0057	-0.0019	-0.0229	-0.0089	0.0042	-0.0004
6	-0.0041	-0.0135	-0.0174	0.0217	0.0078	-0.9973	-0.0073	0.0044	-0.0599	-0.0268	-0.0003	-0.0005
7	-0.0015	-0.0044	0.0097	0.0077	0.0043	-0.0051	0.9986	-0.0359	-0.0370	-0.0018	-0.0004	0.0026
8	-0.0021	-0.0069	-0.0061	0.0173	0.0191	-0.0493	0.0408	0.1761	0.9506	-0.2439	-0.0075	-0.0312
9	-0.0018	0.0017	0.0055	0.0110	-0.0030	0.0151	0.0285	0.9824	-0.1823	0.0033	-0.0179	-0.0142
10	-0.0009	-0.0015	0.0007	-0.0010	0.0038	0.0013	-0.0014	-0.0215	-0.0105	-0.0129	-0.9945	-0.1010
11	-0.0004	0.0076	0.0082	-0.0148	-0.0132	0.0395	-0.0117	-0.0407	-0.2378	-0.9692	0.0158	0.0019
12	-0.0001	-0.0009	-0.0027	0.0009	0.0013	-0.0017	-0.0010	0.0175	0.0266	-0.0071	-0.1015	0.9943

Table D-2 (Cont.)
CORC Iteration Histories

ITERATION 10

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-0.9999	0.0109	0.0013	0.0000	0.0065	0.0063	-0.0018	-0.0001	-0.0016	-0.0018	0.0013	0.0012
2	0.0110	0.9998	0.0132	0.0012	0.0070	0.0037	0.0055	-0.0017	-0.0027	-0.0014	0.0003	-0.0008
3	0.0013	-0.0133	0.9973	0.0713	0.0033	0.0041	-0.0062	0.0002	-0.0024	0.0008	0.0012	-0.0043
4	0.0001	0.0002	0.0713	-0.9972	0.0140	-0.0110	-0.0126	-0.0030	-0.0031	-0.0045	0.0009	-0.0002
5	0.0063	-0.0069	-0.0044	0.0141	0.9995	-0.0228	-0.0121	0.0020	0.0029	0.0019	0.0029	-0.0021
6	-0.0065	0.0038	0.0036	0.0108	-0.0226	-0.9992	0.0170	-0.0218	0.0071	-0.0080	-0.0050	0.0040
7	-0.0018	-0.0057	0.0069	-0.0122	0.0126	0.0167	0.9994	-0.0019	0.0175	0.0081	-0.0014	-0.0150
8	0.0015	-0.0027	-0.0025	0.0026	0.0026	-0.0085	0.0177	0.0727	-0.9970	-0.0035	0.0138	-0.0085
9	-0.0004	0.0020	0.0004	-0.0031	-0.0026	-0.0213	0.0010	0.9968	0.0726	0.0211	-0.0050	0.0140
10	-0.0010	0.0003	0.0019	-0.0008	0.0034	0.0055	0.0012	-0.0063	-0.0156	0.0129	-0.9847	0.1725
11	-0.0018	0.0015	-0.0004	-0.0043	-0.0021	-0.0077	-0.0076	-0.0212	-0.0051	0.9993	0.0170	0.0214
12	-0.0015	-0.0004	-0.0041	-0.0002	-0.0019	-0.0037	-0.0152	0.0119	0.0065	0.0242	-0.1724	-0.9845

ITERATION 11

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-1.000	-0.001	0.004	0.003	-0.001	-0.002	0.001	-0.001	0.000	0.000	0.000	-0.001
2	-0.001	1.000	0.000	-0.001	0.000	0.001	-0.001	0.000	-0.001	0.000	0.000	0.001
3	0.004	0.000	1.000	-0.015	-0.008	0.002	0.004	-0.001	0.000	-0.002	0.000	0.000
4	-0.004	-0.001	-0.015	-1.000	-0.011	-0.001	0.003	0.000	-0.001	0.001	0.000	-0.002
5	-0.001	0.000	0.008	-0.011	1.000	0.005	0.005	-0.004	-0.002	0.002	0.000	0.000
6	0.002	0.001	0.002	0.001	0.005	-1.000	0.002	0.001	-0.003	0.000	-0.001	0.002
7	0.001	0.001	-0.004	0.004	-0.005	0.002	1.000	0.013	-0.009	0.004	0.001	0.004
8	0.000	-0.001	0.000	0.001	-0.002	0.003	-0.009	-0.004	-1.000	-0.001	0.001	-0.004
9	-0.002	0.000	0.001	0.000	0.004	0.001	-0.013	1.000	-0.004	0.001	-0.001	-0.003
10	0.000	0.000	0.000	0.000	0.000	0.001	0.001	-0.001	-0.001	0.003	-0.999	0.035
11	0.000	0.000	0.002	0.001	-0.002	0.000	-0.004	-0.001	-0.001	1.000	0.004	0.006
12	0.001	0.001	0.000	0.002	0.000	-0.002	0.004	-0.003	0.004	0.006	-0.035	-0.999

Table D-2 (Cont.)
CORC Iteration Histories

ITERATION 12

↓Former Mode **NEW MODE →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-1.000	-0.001	0.000	0.000	0.000	0.000	-0.001	0.000	0.000	0.000	0.000	0.000
2	-0.001	1.000	0.000	0.000	-0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.000	1.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000
4	0.000	0.000	0.000	-1.000	0.001	0.001	-0.001	-0.001	0.000	0.000	0.000	0.001
5	0.000	0.001	0.000	0.001	1.000	0.001	0.000	0.000	-0.001	0.000	0.000	0.000
6	0.000	0.000	0.001	-0.001	0.001	-1.000	0.000	0.000	-0.001	0.000	0.000	-0.001
7	-0.001	0.000	0.000	-0.001	0.000	0.000	1.000	-0.003	0.000	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	-0.001	0.001	0.000	-0.003	-1.000	0.000	-0.001	0.002
9	0.000	0.000	0.000	-0.001	0.000	0.000	0.003	1.000	-0.003	-0.002	0.000	0.001
10	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.001	-1.000	-0.003
11	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.000	1.000	0.001	-0.001
12	0.000	0.000	0.000	-0.001	0.000	0.001	0.000	0.001	-0.002	-0.001	0.003	-1.000

ITERATION 13

↓Former Mode **New Mode →**

	1	2	3	4	5	6	7	8	9	10	11	12
1	-1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.000	1.000	0.000	-0.001	0.001	0.000	0.000	0.000	0.000	0.000	0.000
4	0.000	0.000	0.000	-1.000	0.000	0.000	-0.001	0.000	0.000	0.000	0.000	0.000
5	0.000	0.000	0.001	0.000	1.000	0.000	0.000	0.000	-0.001	0.000	0.000	0.000
6	0.000	0.000	0.001	0.000	0.000	-1.000	0.000	0.000	-0.001	0.000	0.000	0.000
7	0.000	0.000	0.000	-0.001	0.000	0.000	1.000	0.000	-0.001	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	-0.001	0.001	-0.001	0.000	-1.000	0.000	0.000	0.001
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000
10	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	-1.000	0.003
11	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.001	0.000
12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	-0.001	0.000	-0.003	-1.000

Table D-3

Final Design Variables & Performance Indices

	Nelson's Method	Modal Expansion	Finite Difference	Semi-Analytical
Total Iterations	13	13	19	40
Final Mass	16.921516	16.921516	16.921535	16.921576
CPU Time (sec) w/MCORC	2.66/2.60/2.60 **2.62**	4.10/4.16/4.11 **4.12**	N/A	29.93/30.40/29.87 **30.07**
CPU Time (sec) w/CORC	2.64/2.60/2.71 **2.65**	4.51/4.48/4.53 **4.50**	N/A	29.80/31.21/31.20 **30.74**
CPU Time(sec) w/o MDTRKG	2.73/2.52/2.63 **2.62**	4.27/4.07/3.99 **4.11**	**20.70**	N/A
Area_1	0.30892541e+01	0.30892541e+01	0.30881789e+01	0.30788636e+01
Area_2	0.25096672e+01	0.25096672e+01	0.25096164e+01	0.25082769e+01
Area_3	0.17667323e+01	0.17667323e+01	0.17640684e+01	0.17723612e+01
Area_4	0.31764767e+01	0.31764767e+01	0.31771979e+01	0.31729827e+01
Area_5	0.23284152e+01	0.23284152e+01	0.23302929e+01	0.23325105e+01
Area_6	0.22477653e+01	0.22477653e+01	0.22467837e+01	0.22508605e+01
Area_7	0.62014741e+00	0.62014741e+00	0.62103188e+00	0.62081271e+00
Area_8	0.94880247e+00	0.94880247e+00	0.94796419e+00	0.95081973e+00
Area_9	0.87466598e+00	0.87466598e+00	0.87386447e+00	0.87527764e+00
Area_10	0.10496428e+01	0.10496428e+01	0.10492880e+01	0.10507542e+01
Area_11	0.62419724e+00	0.62419724e+00	0.62355286e+00	0.62587976e+00
Area_12	0.22572575e+01	0.22572575e+01	0.22667665e+01	0.22600608e+01
Gain_11	0.40259999e+00	0.40259999e+00	0.40259999e+00	0.40259999e+00
Gain_12	0.10417568e+00	0.10417568e+00	0.10396373e+00	0.10421369e+00
Gain_13	0.14694263e+00	0.14694263e+00	0.14689459e+00	0.14650612e+00
Gain_14	0.15802240e+00	0.15802240e+00	0.16020583e+00	0.26840001e+00
Gain_15	0.26840001e+00	0.26840001e+00	0.26840001e+00	0.15823942e+00
Gain_16	0.26840001e+00	0.26840001e+00	0.26840001e+00	0.26840001e+00
Constraint_1	0.49521145e-02	0.49521145e-02	0.45971065e-02	0.67654764e-02
Constraint_2	-0.11357029e-05	-0.11357029e-05	-0.14049945e-05	-0.64751048e-05
Constraint_3	-0.36152937e-06	-0.36152937e-06	-0.37764241e-05	-0.25302725e-04
Constraint_4	0.52055937e+00	0.52055937e+00	0.52021211e+00	0.52216810e+00
Constraint_5	0.70980957e-06	0.70980957e-06	-0.49920700e-05	0.22125470e-04
Constraint_6	-0.16661738e-05	-0.16661738e-05	-0.96335907e-05	0.21539165e-04
Constraint_7	0.30283993e-01	0.30283993e-01	0.31436592e-01	0.29456856e-01
Constraint_8	-0.37449811e-05	-0.37449811e-05	0.15987511e-05	-0.53746877e-04
Constraint_9	0.27212971e+00	0.27212971e+00	0.27066684e+00	0.27244985e+00
Constraint_10	0.56927092e-05	0.56927092e-05	0.43177524e-05	0.81438920e-07
Constraint_11	-0.96850449e-07	-0.96850449e-07	-0.16186649e-05	0.44102639e-04
Constraint_12	0.39976221e+00	0.39976221e+00	0.39806980e+00	0.40069133e+00
Constraint_13	0.21027813e-05	0.21027813e-05	0.37553596e-06	0.19431579e-05
Constraint_14	0.20025045e+00	0.20025045e+00	0.19516227e+00	0.19771661e+00
Constraint_15	-0.26375346e-04	-0.26375346e-04	-0.11327343e-04	-0.90949470e-06

Table D-4

Final Frequency Comparison

MODE	Nelson's Method Frequency (hz)	Modal Expansion Frequency (hz)	Finite Difference Frequency (hz)	Semi-Analytic Frequency (hz)
1	0.2139277	0.2139277	0.2138897	0.2141207
2	0.2545999	0.2545999	0.2545997	0.2545992
3	0.4066455	0.4066455	0.4065966	0.4069065
4	0.4465441	0.4465441	0.4464045	0.5171821
5	0.5168197	0.5168197	0.5168045	0.4468667
6	0.6468926	0.6468926	0.6468151	0.6472891
7	0.7240045	0.7240045	0.7238836	0.7247891
8	0.8907723	0.8907723	0.8365525	0.8909553
9	0.8367811	0.8367811	0.8905662	0.8371138
10	1.1142572	1.1142572	0.9526950	0.9521853
11	0.9526105	0.9526105	1.1156629	1.1145569
12	1.1645244	1.1645244	1.1646685	1.1645293

Appendix E

Algorithm Flowcharts

E.1 Mode Tracking Flowchart

E.2 Flowchart of Gibson's Method (CORC)

E.3 Flowchart of Modified CORC

E.4 Flowchart of MCORC's Extended Validity Tests

E.5 Numerical Example of MCORC

E.1 Mode Tracking Algorithm

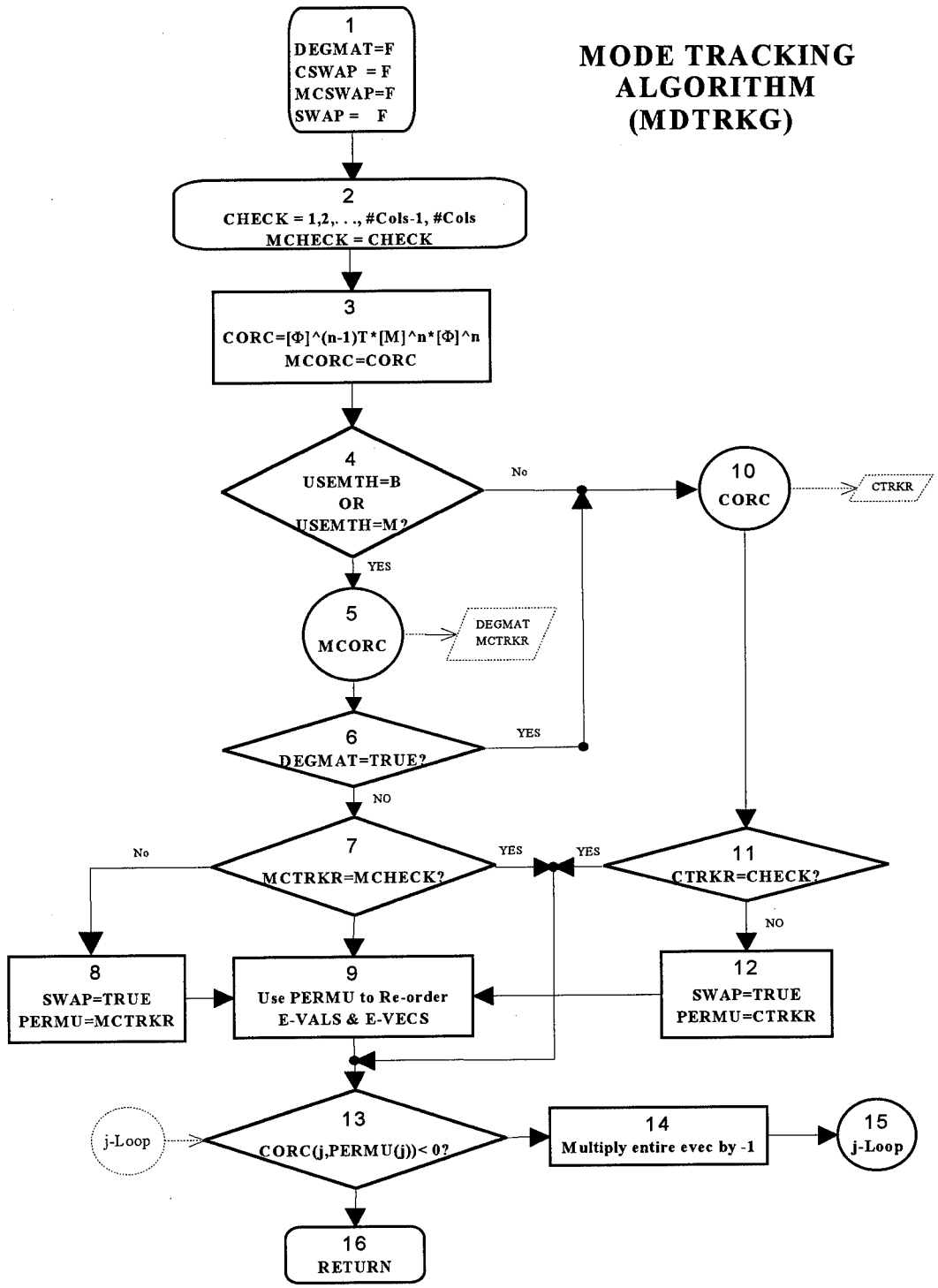


Figure E-1: Mode Tracking Flowchart

E.2 Gibson's Method (CORC)

The ***MDTRKG*** algorithm provides ***CORC*** with the mass-orthonormalized matrix, *CORC*, and a vector *CCHECK*, which is just the order of the modes in the previous iteration. As the modes in the previous iteration are assumed to be in the correct order, the elements in *CCHECK* are sequential.

Steps 1&2: The loop counter is set to process each and every column of CORC individually and the *k*th column is selected.

Step 3: An ordering vector (*indexV*) is initialized to represent the original order of the column of numbers. At the outset *indexV* and *CCHECK* are equal. If after the mode tracking algorithm is complete, these vectors are not equal, the ordering of the elements in *indexV* will indicate the mode swap that occurred.

Steps 4&5: IMSL routines as well as other routines written by the author are used to sort the column in descending order by the magnitude of the absolute value of the elements. The original order of the column is also sorted to track the subsequent permutation and stored as *indexV*. Thus the row number (assigned to '*large*') of the largest element will be in the first slot of *indexV*, the next largest in the second slot, etc. **This row number is the mode which should match the current column number (*k*) being processed.**

Steps 6&7: Initialization of variables used to determine if the mode has already been assigned is processed. *"kc"* is a boolean, (short for "keep checking"), which forces the routine to look in the next slot of *indexV* if the mode has already been assigned.

Steps 9-12: Since the index of the largest number is the first number in *indexV*, it is assumed to correspond to the column currently being processed. It is therefore assigned to that mode, which is tracked in *CTRKR*.

Steps 13-15: The previous elements of *CTRKR* are then checked to determine if that mode had already been assigned. If so, the routine is forced to advance down to the next element in *indexV*, which corresponds to the next largest element. This process continues until all elements are assigned, and all columns processed.

The main subroutine, ***MCTRKG***, then checks *CTRKR* against the mode order from the previous iteration, held in *CCHECK*. If a mode swap is indicated, an IMSL routine is used to permute both the eigenvectors matrix and eigenvalues vector.

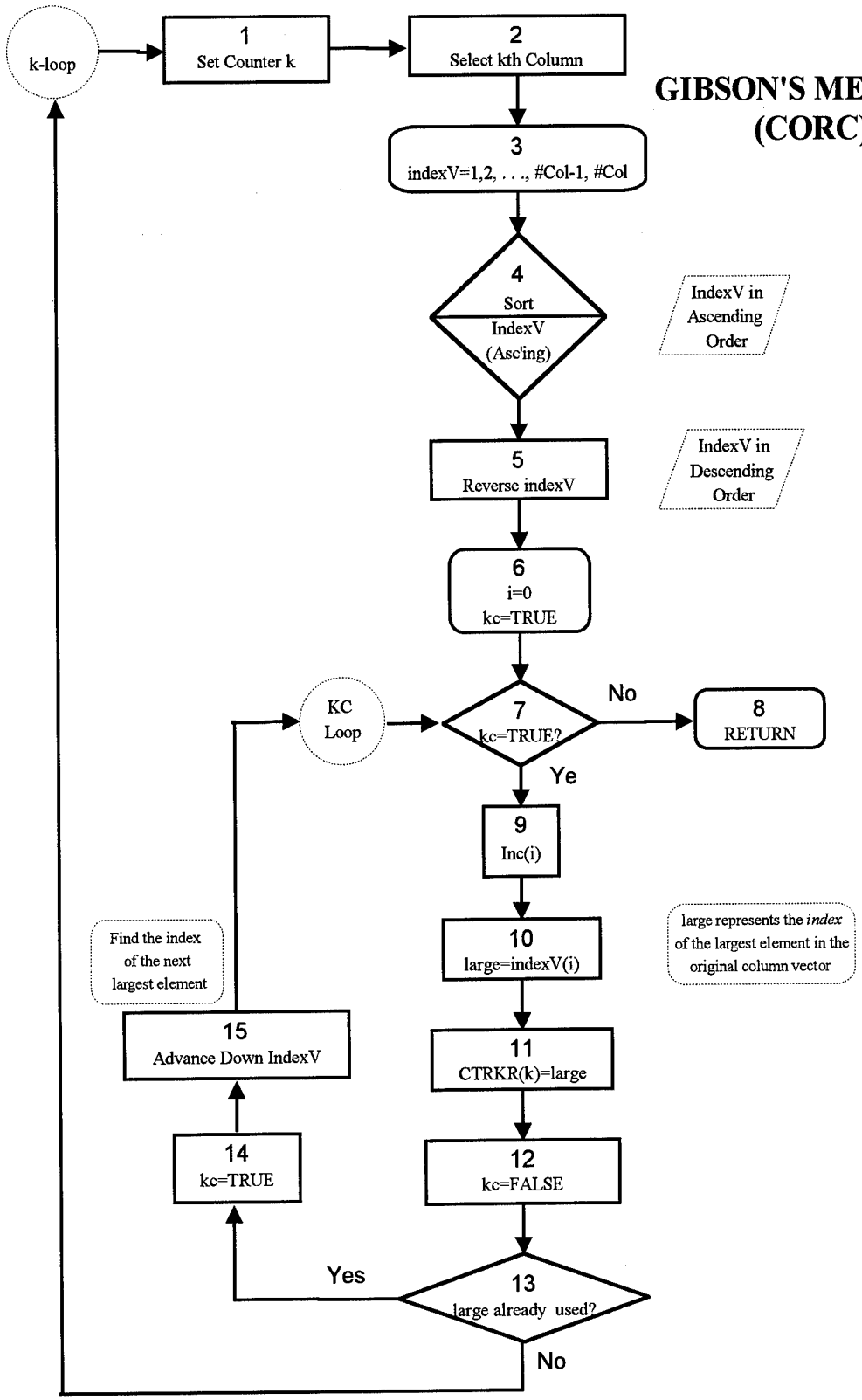


Figure E-2: CORC Flowchart

E.3 Modified CORC

Steps 1&2: The main ***MDTRKG*** routine calls ***MCORC***. Each element of the tracking vector *MCTRKR*, which is analogous to *CTRKR* in Gibson's method, is assigned to -1. This assignment will later serve as a flag indicating at least one mode is still unassigned. As in most search algorithms, the largest element is assumed to be the first element of the matrix, residing in slot [1,1].

Step 3: The matrix is searched for the largest element and its address.

Steps 4-7: If the row returned has already been assigned to a mode, a flag (*DUPLCT*) is set. That element is then set to a value of zero, so it will not be found again on the subsequent search for the largest number. *DUPLCT* then serves to set a shared flag to force the algorithm to repeat. These steps are included only as a safety check in case the matrix has multiple numerical 0's -- it should not normally be executed.

Step 11: This is the path executed if the row output from Step 3 has not been assigned to a mode. The row indicates the mode while the column indicates where that mode falls into place as compared to a perfectly diagonally dominant matrix. If the row and column are the same, the largest element lies on the diagonal. Each column must assign one and only one mode. To ensure that column will not be searched again, every element in the column is set to zero. Also, because that mode has already been assigned, no other element in the row of that matrix can represent a mode. Therefore, the row is also zeroed. This additional step makes the routine more efficient, since it reduces the number of potential re-assignments based on the inequality check in Step 3.

Steps 8-10: The algorithm interrogates *MCTRKR* to ensure all modes have been assigned. A flag is set if there are unassigned modes, otherwise, the algorithm returns to the main sub-routine, ***MDTRKG.***

In ***MDTRKG***, *MCTRKR* is compared to a sequential vector of numbers. If there is not a 1-1 correspondence between these two vectors, *MCTRKR* indicates at least one mode swap has occurred, and the eval vector and evec matrix are permuted.

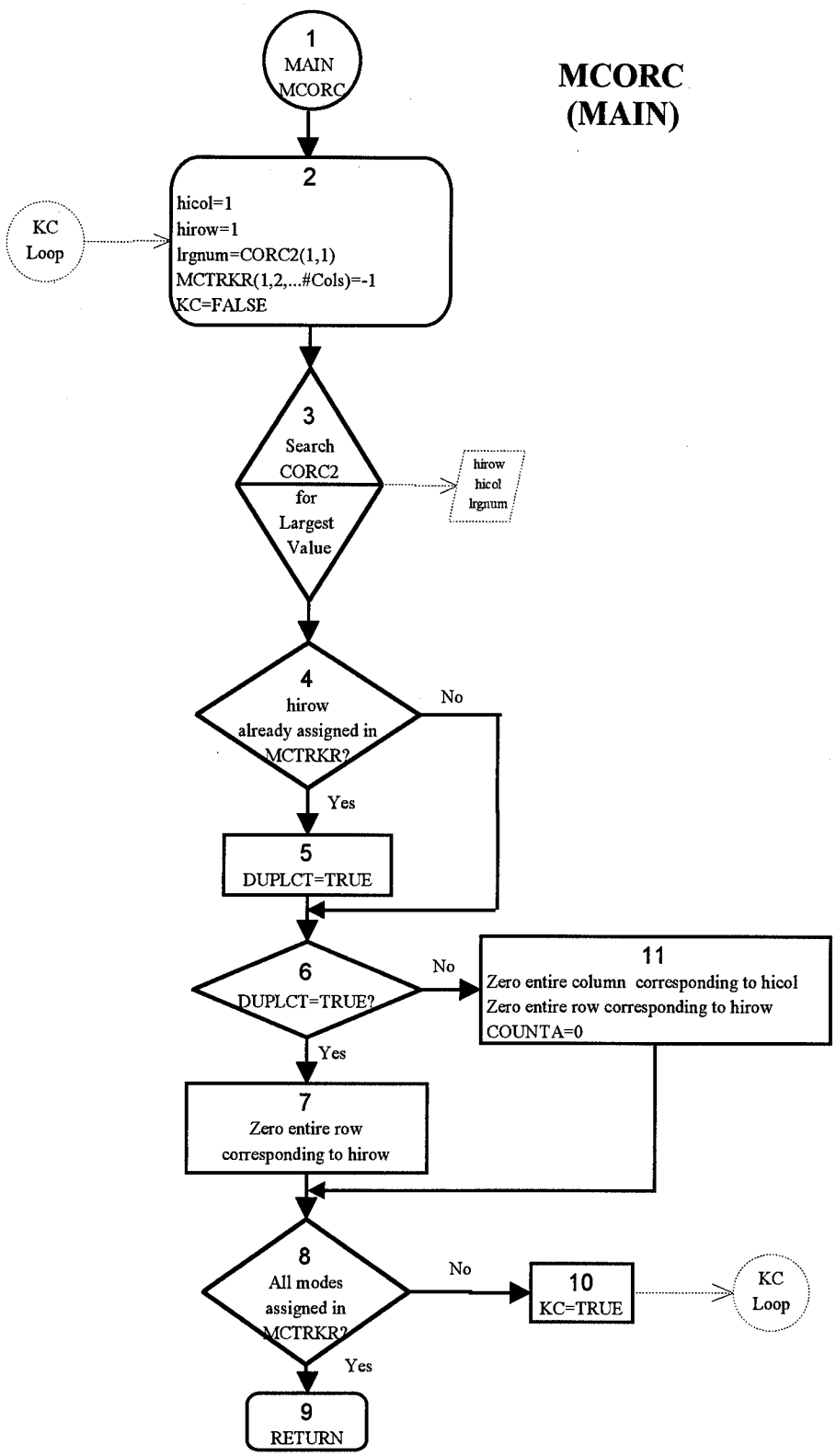


Figure E-3: MCORC Flowchart

E.4 MCORC Extended Validity Tests

Steps 1-2: The validity tests are entered and counters are initialized. *DEGMAT* is a counter which will report that the matrix has a rank defect greater than 1.

Step 3, 14: If all elements are zero, the program is immediately aborted.

Step 4: If the matrix is still being interrogated and the matrix has not yet been determined to be defective, the validity tests continue.

Steps 5-6: A counter is incremented every time the matrix is searched. The counter is reinitialized only when a mode has been assigned in Step 11 of the main MCORC logic. If the matrix has been searched more times than the number of elements, no mode will be assigned except through the process of elimination.

Steps 7-8: The *MCTRKR* vector is interrogated to determine the number of modes unassigned and the location of the last unassigned mode.

Steps 9-11, 15: If only one mode remains unassigned, it is determined through a process of elimination and the remaining unassigned mode is assigned to the mode returned from the previous step. If more than one mode remains unassigned, the matrix has a rank defect greater than one, and the ***MCORC*** routine is aborted.

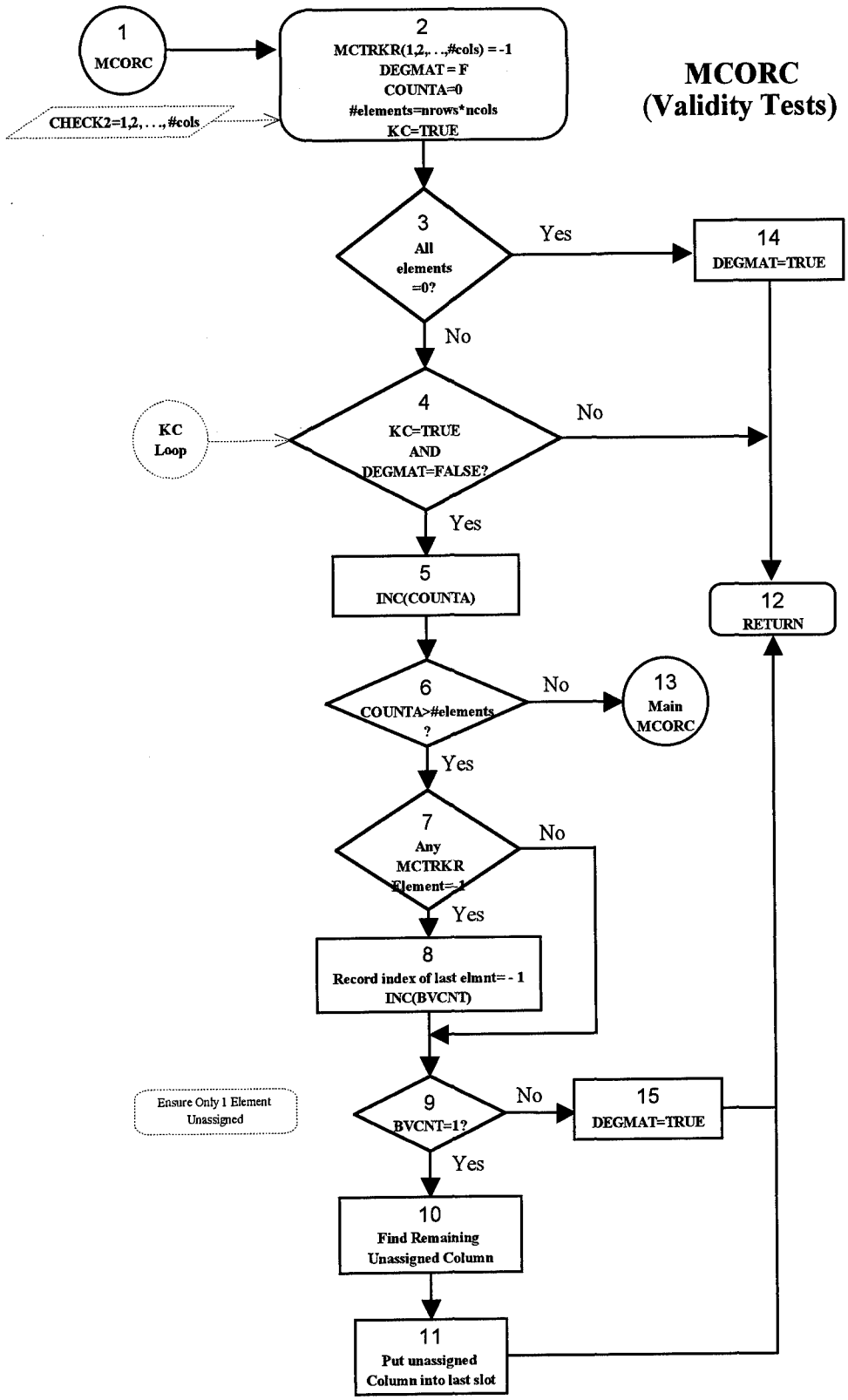


Figure E-4: MCORC Validity Tests

E.5 Numerical Example

The following example demonstrates MCORC's methodology in detecting the 3-4 mode swap between Iterations 1 and 2 when ***FRAME*** was equipped with finite difference derivatives. Modes 7-12 were immediately assigned and the remaining active partition is as follows:

	1	2	3	4	5	6
1	−0.7663	0.5193	−0.2440	0.2586	−0.0091	−.0993
2	0.3542	0.79513	0.3507	−0.1957	−0.2400	0.1069
3	0.0452	0.0101	0.5496	0.7203	0.3634	0.1425
4	−0.4005	−.0727	0.3350	−0.5298	0.3634	0.5390
5	0.2744	0.2875	−0.3656	−0.1113	**0.8043**	−0.1427
6	−0.1792	−0.0519	−0.4880	−0.2674	0.1289	**−0.79518**

⇓ Modes 5 and 6 are assigned

−0.7663	0.5193	−0.2440	0.2586	0	0
0.3542	**0.79513**	0.3507	−0.1957	0	0
0.0452	0.0101	0.5496	0.7203	0	0
−0.4005	−.0727	0.3350	−0.5298	0	0
0	0	0	0	**0**	0
0	0	0	0	0	**0**

⇓ Modes 1&2 are assigned

0	0	0	0	0	0
0	**0**	0	0	0	0
0	0	0.5496	**0.7203**	0	0
0	0	0.3350	−0.5298	0	0
0	0	0	0	**0**	0
0	0	0	0	0	**0**

⇓ Θ^{n}'s Mode **4** correlates more strongly with $\Theta^{(n-1)}$'s Mode **3**

0	0	0	0	0	0
0	**0**	0	0	0	0
0	0	0	**0**	0	0
0	0	**0.3350**	0	0	0
0	0	0	0	**0**	0
0	0	0	0	0	**0**

Finally, Mode **3** is assigned to Mode **4**

Appendix F

ForTran 77 Programs

F.1 *MDTRKG:* This subroutine is called from *MODES* and uses Gibson's method (CORC) and an improved version (*MCORC*) to check for mode swaps. *MCORC* is called from within *MDTRKG* while CORC is resident within this main subroutine. Both mode swapping methods are always executed but the results of *MCORC* are used by default. The user may request a particular method be used. If the user inputs a faulty design which leads to CORC being a singular matrix, *MCORC* may not complete nominally. Therefore, a warning message is displayed and the routine automatically defaults to CORC. If a mode swap is detected, IMSL routines are used to permute the eigenvalue and modal matrices. *MDTRKG* is a passive routine only, and cannot effect the main optimization routine. If a mode swap is detected and the search returned is feasible, *MDTRKG* will permute the matrices and return them to the optimization routine.

F.2 *MCORC:* This subroutine is called by *MDTRKG* and checks the CORC matrix for mode swaps. The routine does not sort as is done in Gibson's methodology but instead actively searches the matrix for the largest elements and assigns them to the corresponding mode by building a permutation vector. If that vector does not have a 1-1 correspondence with a sequential tracking vector, a mode swap has occurred and the eigenvalue and modal matrices are subsequently permuted according to that permutation vector.

F.3 *EXPNSN*: This subroutine uses the expansion method to determine eigenvector sensitivity. It is called by *SNSTVT*, which has been modified to allow the user to pick by which method eigenvector sensitivity is to be computed. The routine assumes all eigenvectors have been calculated, but does not require all eigenvectors be controlled. This subroutine must call *DKMDVP* to determine the term $(\mathbf{K}' - \lambda\mathbf{M}')$.

```
      SUBROUTINE MDTRKG(nrows,ncols,currmu,oldvec,temp,evec,eval)
C
C Called from modes.f
C
C This is 72 columns!!
C -----------------------------------------------------------------|
C
C nrows      : # of rows = neq
C ncols      : # of cols = neval
C currmu     : M^nU^n; passed as DCOR(MU)
C oldvec     : eigenmatrix of last iteration; passed as DCOR(P2OLDV)
C indexV     : The vector of indices of CORC
C evec       : The eigenmatrix of the current iteration, i.e. U of MU
C eval       : [eval|stiffness|mass]
C cswap      : Logical; tells Gibson's CORC routine found a mode swap
C mcswap     : Logical; tells MCORC routine found a mode swap
C (m)check   : Used as a basis to determine if trackers found swap
C corc((),2,3) : phi^(n-1)T*M^n*phi^n; need 3 copies since both CORC
C            and MCORC corrupt original as they track modes and
C            one copy needed for sign check
C (m)ctrkr   : Vector to track mode swaps in MCORC and CORC
C degmat     : BOOLEAN, tells you if MCORC found degenerate matrix
C            Used to switch to CORC if req'd
C
C
      INTEGER   MAXTRM
      PARAMETER  (MAXTRM=20)
C
      LOGICAL   kc, cswap, mcswap, swap, degmat
      EXTERNAL  PN2MEM
C
      INTEGER   i,j,k,CSP,large,indexV(MAXTRM), temp(MAXTRM), PN2MEM,
     +          check(MAXTRM), mdswap(MAXTRM), mctrkr(MAXTRM),
     +          mcheck(MAXTRM), ipermu(MAXTRM), mctotl,ctotl,revsin
      REAL*8    currmu(nrows,ncols), oldvec(nrows,ncols),
     +          corc(MAXTRM,MAXTRM), corc2(MAXTRM,MAXTRM),
     +          corc3(MAXTRM,MAXTRM),evec(nrows,ncols),eval(ncols,3)
C
C
C--INCLUDE CORE-------------------------------------------------------
      INTEGER MEMDIM, INT, RSP, RDP, CDP, KOR
      PARAMETER ( MEMDIM = 512 000 )
      PARAMETER ( INT=0, RSP=1, RDP=2, CDP=4 )
      REAL     COR( MEMDIM )
      COMPLEX CCOR( MEMDIM / 2 )
      DOUBLE PRECISION DCOR( MEMDIM / 2 )
      COMPLEX*16 ZCOR( MEMDIM / 4 )
      EQUIVALENCE ( KOR, COR, DCOR, CCOR, ZCOR )
      COMMON /MEMORY/ KOR( MEMDIM )
      SAVE   /MEMORY/
C
```

```
C  ** INITIALIZATION **
C
      CALL MARK('MDTRKG begin')
      cswap  = .FALSE.
      mcswap = .FALSE.
      swap   = .FALSE.
      ctotl  = 0
      mctotl = 0
      DO 30 i=1,ncols
        check(i)  = i
        mcheck(i) = 1
30    CONTINUE
C
C
C  ** Multiply the transpose of oldvec * mu using
C    DMXTYF(NRA,NCA,A,LDA,NRB,NCB,B,LDB,NRC,NCC,C,LDC) (pg 1195) **
C
*       WRITE(*,*)'Just before A^TB, oldvec is '
*       CALL DWRRRN('EVEC=>DCOR(P2OLDV)',nrows,ncols,
*    + oldvec,nrows,0)
C
      CALL DMXTYF(nrows, ncols, oldvec, nrows, nrows, ncols,
     + currmu, nrows, nrows, ncols, corc, MAXTRM)
      CALL DWRRRN('u^(n-1)T*M^n*u From corcy',nrows,ncols,corc,MAXTRM,0)
C
      CALL DCRGRG(nrows,corc,MAXTRM,corc2,MAXTRM)
      CALL DCRGRG(nrows,corc,MAXTRM,corc3,MAXTRM)
C
C  DSVRBP(N, RA, RB, IPERM) (pg. 1286) sorts by ascending order by || and
C  returns permutation vector.
C    NOTE: Don't need corc again so just re-assign as corc.
C    NOTE: DSVRGP sorts in ASC order so need to reverse vector using revvec
C    NOTE: Need to initalize indexV
C
C  RUN MCORC first since more robust.
C
      CALL MCORC(nrows,ncols,corc2,mctrkr,mcheck,MAXTRM,degmat)
      DO 40 i=1,ncols
         IF (mctrkr(i) .NE. check(i)) THEN
            mctotl = mctotl+1
            mcswap = .TRUE.
         ENDIF
40    CONTINUE
C
C ** CORC **
C
C This for loop will examine each col of C (found by CORC) independently
C and sort in ASCENDING order.  Then revvec will reverse order.
C
     DO 50 k=1, ncols
C
       DO 51 i=1, nrows
         indexV(i) = i
51     CONTINUE
```

```
      CALL DSVRBP(nrows,corc(1,k),corc(1,k), indexV)
C
      CALL REVVEC(nrows, indexV)
C
C       The index represents the magnitudes
C       of the original col.  The first row should contain
C       the original index of the highest magnitude.  All subsequent
C       elements also sorted so just keep stepping down the colmn.
C       kc = keepgoing -- a boolean that says you tried to assign a
C       subsequent mode to a mode that had previously been assigned
C
        i=0
        kc = .TRUE.
        DO WHILE (kc)
           i=i+1
           IF (i .GT. nrows) THEN
              kc = .FALSE.
           ELSE
              large=indexV(i)
              temp(k) = large
              kc = .FALSE.
              DO 70 j=1,k-1
                 IF (large .EQ. temp(j)) THEN
                    kc = .TRUE.
                 ENDIF
70            CONTINUE
           ENDIF
        END DO
50    CONTINUE
C
C
C       Now check for mode swaps by comparing check to temp;
C       store results in mdswap => -1 is no swap,
C       otherwise it swapped mode
C
      DO 100 k=1,ncols
        IF (temp(k) .EQ. check(k)) THEN
          mdswap(k) = -1
        ELSE
           ctotl = ctotl+1
           cswap = .TRUE.
           mdswap(k) = temp(k)
           CSP = PN2MEM('CSPVEC')
        ENDIF
100     CONTINUE
C
      IF (degmat) THEN
        swap = cswap
        WRITE(*,*)'Switching to CORC'
        DO 160, i=1,ncols
          ipermu(i) = temp(i)
160       CONTINUE
```

```
      ELSE
        swap = mcswap
        WRITE(*,*)'Using MCORC'
        DO 170, i=1,ncols
          ipermu(i) = mctrkr(i)
170       CONTINUE
      ENDIF
C
      WRITE(*,*)'mcswap = ',mcswap
      WRITE(*,*)'cswap  = ',cswap
      IF (mcswap .OR. cswap) THEN
          WRITE(*,*)'Mode Swap detected'
          WRITE(*,*)'MCORC        CORC'
          DO 180 i=1,ncols
            WRITE(*,*)mctrkr(i),'        ',temp(i)
180         CONTINUE
          WRITE(*,*)'MC# = ',mctotl,'   C# = ',ctotl
          WRITE(*,*)'CSP = ',CSP
            DO 200 j=CSP,CSP-1+12
               WRITE(*,*)COR(j)
200         CONTINUE
C
      ELSE
          WRITE(*,*)'No mode swap detected'
      ENDIF
C
C  If MCORC (not CORC!) caught a modeswap, mcswap will be true and only then
C  will a permutation of both evec and the (ncol,3) eval be permutated.
C  Use DPERMA(nra,nca,a,lda,ipermu,ipath,aper,ldaper) (pg. 1276)
C  Here a and aper are the same, i.e. evec and the permutated evec
C  ipath is 1 for rows (for eval), 2 for cols (for evec)
C
C
C  The following five lines are an interim convenient way to
C  test Gibson against MCORC
C
*         swap = cswap
*         WRITE(*,*)'Trying only CORC'
*         DO 183, i=1,ncols
*           ipermu(i) = temp(i)
*183      CONTINUE

      IF (swap) THEN

C                 ** permutate columns of evec **
          CALL DPERMA(nrows,ncols,evec,nrows,ipermu,2,evec,nrows)
C
C                 ** permutate rows of eval **
          CALL DPERMA(ncols,3,eval,ncols,ipermu,1,eval,ncols)
      ELSE
          WRITE(*,*)'No mode swap detected'
      ENDIF
```

```
C  CHECK evec sign
C
     revsin=0
      DO 300 j=1, ncols
        IF (corc3(check(j), ipermu(j)) .LT. 0) THEN
              WRITE(*,*)'sign change in evec ',j
              revsin=revsin+1
              DO 310 i=1, nrows
                 evec(i,j) = -evec(i,j)
310             CONTINUE
        ENDIF
300    CONTINUE
      WRITE(*,*)'revsin = ',revsin
C
C
      CALL MARK('MDTRKG END')
      RETURN
      END
```

```
      SUBROUTINE MCORC(nrows,ncols,corc2,mctrkr,mcheck,
     +                 MAXTRM, degmat)
C
C This is 72 columns:
C  ------------------------------------------------------------------|
C  This subroutine is a modified version of the CORC routine
C  created by Gibson for the x-orthogonality check.  This method,
C  unlike the original CORC, is not order dependent and therefore
C  more likely to detect a mode swap.
C
C  This routine computes corc2=phi^(n-1)T*M^n*phi^n and searches
C  the resulting matrix for its largest entry.  It returns the
C  corresponding row and column.  The row of the largest entry
C  corresponds to the mode.  The algorithm places that row number
C  into the same column of mctrk as the column returned.  If
C  there have been no mode swaps, the row and column number will
C  be the same.  Since that mode has now been designated, there is
C  no reason to check that column anymore.  Also, that mode can not be
C  assigned again.  Therefore, the routine zeros out that row and
C  column.
C
C  Inaccurate models may create degenerate matrices, (those with at
C  least one col of 0's).  Because we actively zero out cols, this
C  could lead to the alg. continuing to search a matrix of all 0's.
C  We filter out this contingency in three separate tests.
C    1)  If the matrix we read in is all 0's we immediately abort.
C    2)  If we can track all but one column, we use the process of
C        elimination to designate that last mode.
C    3)  If we are left with more than one column of zeros, we
C        gracefully abort.
C
C ** VARIABLES
C     hirow/col : row/col of the element with the largest magnitude
C     lrgnum    : Magnitude of largest element
C     MAXTRM    : dummy var from corc; largest possible model size
C     mctrkr    : vector which tracks the mode swaps using MCORC
C                 If no swaps, each ith element of mctrkr will be i
C     counta    : Ensures no more searches than there are elements
C     slot      : The last slot in mctrkr to be filled by process
C                 of elimination
C     used      : A vector which tracks which modes have been used;
C                 Like the 'Used Letter Board" on Wheel of Fortune
C     duplct    : BOOLEAN that tells you the mode you want is taken
C     kc        : keep checking; tells you mctrkr not yet filled
C     mcheck    : The full used-letter board you'll reduce
C     degmat    : BOOLEAN tells you if you had a corrupted matrix
C                 (i.e. more than one col of all zeros)
C **
C
C
```

```
      INTEGER   MAXTRM
      REAL*8    corc2(MAXTRM,MAXTRM), lrgnum
      INTEGER   i,j,nrows,ncols, hicol, hirow, mctrkr(ncols),
     +          mcheck(ncols), counta, nelmnt,used,slot
      LOGICAL   kc, duplct, degmat
C
C     ** INITIALIZATION **
            used = 0
            degmat = .FALSE.
            counta = 0
            nelmnt = nrows*ncols
            kc=.TRUE.
C
C     ** Check for matrix of all zeros **
C
      DO 10 i=1,nrows
        DO 20, j=1,ncols
          IF (corc2(i,j) .NE. 0) THEN
              kc = .FALSE.
          ENDIF
20      CONTINUE
10    CONTINUE
C
      IF (kc) THEN
         degmat = .TRUE.
         WRITE(*,*)'DEGENERATE MATRIX - all 0s.'
         WRITE(*,*)'MCORC will not work.'
         GO TO 1000
      ENDIF
C
*      CALL DWRRRN('corc2 in boblng',nrows,ncols,corc2,MAXTRM,0)
       DO 100 j=1, ncols
         mctrkr(j) = -1
100    CONTINUE
C
      kc= .TRUE.
C
      DO WHILE ((kc) .AND. .NOT. (degmat))
         counta = counta+1
*         WRITE(*,*)'counta = ',counta
C
C  ** Following IF should only be run if mctrkr not filled AND
C    you've re-searched every element of the remaining matrix **
C
         IF (counta .GT. nelmnt) THEN
             DO 120 j=1,ncols
                IF (mctrkr(j) .EQ. -1) THEN
                    used = used + 1
C                ** Find slot that's not filled yet **
                    slot = j
*                    WRITE(*,*)'slot = ',slot
                ENDIF
120             CONTINUE
```

```
C
C       ** ensure only 1 remaining slot to fill in mctrkr **
C       ** Create the full used-letter board to pick from **
C
          IF (used .EQ. 1) THEN
*           WRITE(*,*)'In bv, bv = ',used
            DO 140 i=1,ncols
              DO 150 j=1,ncols
C
C               ** Find the slot that's not filled
C                 w/ a legal "letter" from board **
C
                IF (mctrkr(i) .EQ. mcheck(j)) THEN
                  mcheck(j) =  -2
                ENDIF
150           CONTINUE
140         CONTINUE
*           WRITE(*,*)'mctrkr = '
*           DO 155 i=1,ncols
*             WRITE(*,*)'mctrkr(',i,') = ',i
*155        CONTINUE
*           DO 156 i=1,ncols
*             WRITE(*,*)'mcheck(',i,') = ',i
*156        CONTINUE
C
C             ** Put the remaining mode into
C               the one unused slot **
C
            DO 160 i=1,ncols
              IF (mcheck(i) .NE. -2) THEN
                mctrkr(slot) = i
              ENDIF
160         CONTINUE
            WRITE(*,*)'Had 1 column of zeros remaining'
            WRITE(*,*)'Put Mode ',mctrkr(slot),
     +              ' into slot ',slot, ' of mctrkr'
            GOTO 1000
C
C             ** more than one column of zeros **
C
          ELSE
            degmat = .TRUE.
            WRITE(*,*)'DEGENERATE MATRIX - all 0s.'
            WRITE(*,*)'MCORC will not work.'
            GO TO 1000
          ENDIF
        ENDIF
C
C
```

```
C  ** Back to real tracking logic, above just tests for bad data **
C
        hicol=1
        hirow=1
        lrgnum = DABS(corc2(hirow,hicol))
*         WRITE(*,*)'lrgnum just assigned to ',lrgnum
*         WRITE(*,*)'ist assignment'
*         WRITE(*,*)'hicol= ',hicol,' hi row= ',hirow,'
*    +            large = ',lrgnum
*         WRITE(*,*)
*         WRITE(*,*)'Just to check, corc2(1,1) =
*    +            ',DABS(corc2(1,1))
*          WRITE(*,*)'AND Just to check, corc2(12,12) = ',
*    +             DABS(corc2(12,12))
C
        kc =.FALSE.
        DO 200 i=1,nrows
           DO 300 j=1,ncols
                IF (DABS(corc2(i,j)) .GT. lrgnum) THEN
                   lrgnum = DABS(corc2(i,j))
                   hicol = j
                   hirow = i
*                    WRITE(*,*)'In Search :'
*                    WRITE(*,*)'hicol =',hicol,' hirow = ',
*    +                        hirow,' large = ',lrgnum
                ENDIF
300          CONTINUE
200       CONTINUE
C
*         WRITE(*,*)'mctrkr b4 eq -1 check'
*         DO 255 i=1,ncols
*           WRITE(*,*)mctrkr(i)
255       CONTINUE
        duplct = .FALSE.
        DO 280 i=1,ncols
C
C             ** If mode taken, try again **
           IF (hirow .EQ. mctrkr(i)) THEN
                duplct = .TRUE.
           ENDIF
C
280       CONTINUE
        IF (duplct) THEN
C                ** zero the element **
           corc2(hirow,hicol) = 0
           kc = .TRUE.
        ELSE
           mctrkr(hicol) = hirow
           counta = 0
C                ** zero the column **
           DO 250 i=1, nrows
              corc2(i,hicol) = 0
250          CONTINUE
        ENDIF
```

```
C
*         CALL DWRRRN('corc2 in dup check',nrows,ncols,
*    + corc2,MAXTRM,0)
*         WRITE(*,*)'mctrkr'
*         DO 258 i=1,ncols
*           WRITE(*,*)mctrkr(i)
*258       CONTINUE
C
C           ** Following asks if mctrkr is filled yet **
      DO 400 j=1, ncols
          IF (mctrkr(j) .EQ. -1) THEN
             kc = .TRUE.
          ENDIF
400     CONTINUE
      END DO
C
1000    CONTINUE
*         WRITE(*,*)' FINAL mctrkr'
*         DO 2000 i=1,ncols
*           WRITE(*,*)mctrkr(i)
2000      CONTINUE
C
      RETURN
      END
```

```
*
*DECK EXPNSN
* Last Modified: 12-Oct-94
*
      SUBROUTINE EXPNSN ( NRE, STIFF, LDK, KCOD, MASS, LDM, MCOD, PHIR,
     +                    NP, PHIG, NEQ, EIG, DEIG, NDP, MASNRM, DT,
     +                    KSTAR, FSTAR, V, MAPR2G, KEEP, STORE, F,
     +                    DPHIR, DPHI )
C
C*********************************************************************
C  The EXPNSN subroutine calculates eigenvector sensitivity
C  according to Modal Expansion
C*********************************************************************
C
C--INPUT
C
C  NRE..... Number of Reduced system Equations
C  NEQ..... Number of global  system Equations
C  STIFF.... Stiffness (symmetric banded) matrix, K
C  LDK..... Leading Dimension of K
C  KCOD..... Codiagonals in K
C  MASS..... Mass (symmetric banded) matrix, M
C  LDM..... Leading Dimension of M
C  MCOD..... Codiagonals in M
C  PHIR..... Matrix of eigenvectors (stored columnwise), Reduced DoFs
C  PHIG..... Matrix of eigenvectors, Global (unreduced) DoFs
C  NP..... Number of eigenvectors
C  EIG..... Eigenvectors
C  DEIG..... Eigenvector Gradient
C  NDP..... Number of Design Parameters
C  DT..... Derivative of generalized mass
C  MASNRM... Mass orthonormalization (or max elem norm)
C  MAPR2G... Map of Reduced to Global DoFs
C  KEEP..... Store DPHIR
C  STORE.... Store F
C
      INTEGER NRE, NEQ, LDK, KCOD, LDM, MCOD, NP, NDP, MAPR2G(NEQ)
      LOGICAL MASNRM, KEEP, STORE
      DOUBLE PRECISION STIFF(LDK,NRE), MASS(LDM,NRE), PHIR(NRE,NP),
     +                 PHIG(NEQ,NP), EIG( NP ), DEIG( NDP, NP ),
     +                 DT(NDP,NP)
C
      INTEGER          i,j,k
      DOUBLE PRECISION  NUM, DEN ,c(6,12)

C
C--OUTPUT
C
C  KSTAR.... LHS matrix multiplier, K - EIG*M
C  FSTAR.... RHS load vector, [DEIG*M - (K'-EIG*M') ]*PHIR
C  V..... Vector portion of DPHI due to other modes
C  F..... [LAMBDA'*M - (K'-LAMBDA*M')]*PHI
C  DPHIR.... Gradients of PHI in reduced coordinates
C  DPHI..... Gradients of PHI in global  coordinates
```

```
C
      DOUBLE PRECISION KSTAR(NRE,*), FSTAR(NEQ,4), V(NEQ),
     +                 F(NRE,NP,NDP),
     +               DPHIR(NRE,NP,NDP), DPHI(NEQ,NP,NDP)
C
C--Global variables
C
*     INCLUDE '(CORE)'
C--INCLUDE CORE----------------------------------------------------------
      INTEGER MEMDIM, INT, RSP, RDP, KOR
      PARAMETER ( MEMDIM = 512 000 )
      PARAMETER ( INT=0, RSP=1, RDP=2 )
      REAL     COR( MEMDIM )
      DOUBLE PRECISION DCOR( MEMDIM / 2 )
      EQUIVALENCE ( KOR, COR, DCOR )
      COMMON /MEMORY/ KOR( MEMDIM )
      SAVE   /MEMORY/
C-----------------------------------------------------------------------
C
C--Local variables
C
C  C..... Constant multiplier for current eigenvector
C  I..... Ith Design variable
C  J..... Jth eigenvector
C  K..... Max element of PHI_j
C  L..... Equation Line # for reduction
C  M..... Loop index for design parameters of current element
C  N..... Nth section property for current element
C  LDKS..... Leading Dimension of KS
C  NSP..... Number of Section Properties for current element
C          (if NSP<0, current element properties are not designed)
C  PTDMP.... PHI_j^T * M' * PHI_j
C
      INTEGER   NSP
      DOUBLE PRECISION  PTDMP(4), init
C
C  Pointers
C
C  CSPT..... CSP Table
C  KS..... KSTAR in Real Band storage mode
C  IPVT..... Pivot info to solve linear system as non-symmetric
C
      INTEGER CSPT
C
C--External functions
C
      DOUBLE PRECISION DDOT
      INTEGER  IDAMAX, IDAMIN, LK8MEM, PN2MEM
      EXTERNAL IDAMAX, IDAMIN, LK8MEM, PN2MEM, DDOT, DVCAL
```

```
C
C--BEGIN
C
      CALL MARK('EXPNSN BEGIN')

      CSPT = PN2MEM( 'CSPTBL' )
      init = 0.0

C
C
C     i: outer loop counter for 6 controlled evecs
C     k: middle loop counter which runs thru 12 evecs
C     j: inner loop counter acting as a summer to scale evecs
C       and add them in a running sum
C     At the end of the j loop I will have ONE 30x1 DPHI(i)/DA(k)
C     At the end of the k loop I will have a 30x12 matrix with
C     1-12 k columns and 30 DOFS in the rows
C
C     Fi is needed to pull the 1st column out of the 30x4 FSTAR
      DO 100 i = 1, NP
        DO 200 k=1, NDP
C                   *initialize runsum vector PROD*
          CALL DSET(NEQ,init,DPHI(1,i,k),1)
          CALL DSET(NRE,init,DPHIR(1,i,k),1)
C
          DO 300 j=1, NRE
C                   *DKMDVP rtns latter half of numerator,
C                    Fi = (K'-lambda(i)*M')*PHI(i) for i<>j
C                    or the phi(j)^T*M'*phi(j) if i=j*
C
            CALL DKMDVP(k,CSPT,EIG(i),PHIG(1,i),NEQ,NSP,FSTAR,PTDMP)


            IF (i .NE. j) THEN
C
              NUM = DDOT(NEQ,PHIG(1,j),1,FSTAR,1)
              DEN = EIG(i) - EIG(j)
              c(i,j) = NUM/DEN
            ELSE
              c(i,j) = -0.5*PTDMP(1)
            ENDIF
C                   *Now have my c(i,j) if i=j or i<>j
C                    Now dot it into PHI(j) and add
C                    it to running sum*
C
            CALL DAXPY(NEQ,c(i,j),PHIG(1,j),1,DPHI(1,i,k),1)

C                    * Now do the same for reduced set*
            CALL DAXPY(NRE,c(i,j),PHIR(1,j),1,DPHIR(1,i,k),1)
300         CONTINUE
C
```

```
C                   * At this point I have one vector each
C                     for the globals and the reduced set.
C                     That vector is derivative of PHI(i)
C                     wrt A single area*
C                   * The DPHI and DPHIR 3-D arrays are
C                     set up (NEQ,NP,NDP) or (NRE,NP,NDP)
C                     They are 30 (or 12) long, 6 wide, and
C                     12 deep, or 30 by i by k.
C

*           CALL DCOPY(NEQ,PRODG,1,DPHI(1,i,k),1)
C                   * Now do the same for the reduced set
*           CALL DCOPY(NRE,PRODR,1,DPHIR(1,i,k),1)

200     CONTINUE
100   CONTINUE

*     DO 1000 i=1,6
*       DO 2000 k=1,12
*           DO 3000 j=1,12
*               HOLD(j,k) = DPHIR(j,i,k)
*3000          CONTINUE
*2000      CONTINUE
*       WRITE(*,*)'MODE = ',i
**       CALL DWRRRN('DPHI,dof/DA(k)',12,12,HOLD,12,0)
*1000  CONTINUE

C
      CALL MARK('EXPNSN END')
      RETURN
      END
```

Appendix G

MATLAB Programs

STATS: This MATLAB subroutine is used to correlate the various eigenvector derivatives. The derivative matrices are converted to text using an outside program and loaded sequentially. Each mode is processed individually. In all runs, the finite difference derivatives are taken as the *primary* true solution and the derivatives returned from ASTROS taken as the *secondary* true solution. All analytical derivatives -- the original and corrected derivatives from **FRAME**'s implementation of Nelson's Method and the derivatives calculated by modal expansion -- are compared to the true solutions by manipulating the input names. Statistics are applied to each individual element and include straight differences, relative differences, mean and standard deviation. In addition, the routine determines sign correlation between the derivatives (where a '0' is assumed as neither positive or negative). It also determines if the derivatives being tested fall within 25% and 50% respectively, of the true finite difference solution.

Note: Several other plotting and test programs (programs used to test logic before being converted to ForTran 77), were written but were deemed of little utility to the thesis user. These programs are available from the author, or from Capt R. Canfield at the AFIT School of Engineering.

STATS.m

```
if prmode=='p'
  print
elseif prmode=='pg'
  print -dps cnstrnt2
else
  pause
end

for modenumber=1:6

if modenumber==1
 load fd1.txt
 load anc1.txt
 load ast1.txt
 fd=fd1;
 ana=anc1;
 ast=ast1;
elseif modenumber==2
 load fd2.txt
 load anc2.txt
 load ast2.txt
 fd=fd2;
 ana=anc2;
 ast=ast2;
elseif modenumber==3
 load fd3.txt
 load anc3.txt
 load ast3.txt
 fd=fd3;
 ana=anc3;
 ast=ast3;
elseif modenumber==4
 load fd4.txt
 load anc4.txt
 load ast4.txt
 fd=fd4;
 ana=anc4;
 ast=ast4;
```

```
elseif modenumber==5
 load fd5.txt
 load anc5.txt
 load ast5.txt
 fd=fd5;
 ana=anc5;
 ast=ast5;
else modenumber==6
 load fd6.txt
 load anc6.txt
 load ast6.txt
 fd=fd6;
 ana=anc6;
 ast=ast6;
end;


% BEGIN entire logic for each modal data set


disp('MODE is')
modenumber
disp(' ')
disp(' ')

disp('Multiplying ASTROS OUTPUT by 0.10 for Scaling')
disp(' ')
disp('Scaling all matrices by 10e6 to liberate digits')

ast=[ast(:,1:4)/10 ast(:,5:12)];
ast=ast*1000000
ana=ana*1000000
fd= fd *1000000

disp('MODE is')
modenumber
disp(' ')
disp(' ')


absfd=abs(fd);
absast=abs(ast);
absana=abs(ana);
```

```
notall3      = 0;
notanaandast  = 0;
notastandfd   = 0;
notanaandfd   = 0;
ana_noteqto_ast_and_fd = 0;
numanazeros=0;
numastzeros=0;
numfdzeros =0;

for i=1:12
 for j=1:12
  if ana(i,j) < 0
       signana(i,j) = 200;
  elseif ana(i,j) > 0
     signana(i,j) = 100;
  else
    signana(i,j) = 300;
    numanazeros = numanazeros+1;
  end

  if ast(i,j) < 0
       signast(i,j) = 20;
  elseif ast(i,j) > 0
     signast(i,j) = 10;
  else
    signast(i,j) = 30;
    numastzeros = numastzeros+1;
  end

  if fd(i,j) < 0
       signfd(i,j) = 2;
  elseif fd(i,j) > 0
     signfd(i,j) = 1;
  else
     signfd(i,j) = 3;
       numfdzeros = numfdzeros+1;
  end

  signcomp(i,j) = signana(i,j)+signast(i,j)+signfd(i,j);

 end;
end;
```

```
for i=1:12
  for j=1:12
    if ((signana(i,j)/100) ~= (signast(i,j)/10))
      if ((signana(i,j)/100) ~= signfd(i,j))
        if ((signast(i,j)/10)  ~= signfd(i,j))
           notall3 = notall3+1;
        end
      end
    end
    if ((signana(i,j)/100)  ~= (signast(i,j)/10))
           notanaandast = notanaandast+1;
    end
    if ((signana(i,j)/100) ~= signfd(i,j))
           notanaandfd = notanaandfd + 1;
    end
    if ((signast(i,j)/10) ~= signfd(i,j))
           notastandfd = notastandfd + 1;
    end
    if ((signana(i,j)/100) ~= (signast(i,j)/10))
      if ((signana(i,j)/100) ~= signfd(i,j))
        if ((signast(i,j)/10) == signfd(i,j))
            ana_noteqto_ast_and_fd = ana_noteqto_ast_and_fd+1;
        end
      end
    end
  end;
end;



astminusfd     = ast-fd;
anaminusfd     = ana-fd;
astminusfddivfd = ((ast-fd)./fd)*100;
anaminusfddivfd = ((ana-fd)./fd)*100;

disp('ASTROS - FD')
astminusfd

disp('MODE is')
modenumber
disp(' ')
disp(' ')
```

```
disp(' ')
disp('Max/Min Difference btwn ASTROS and FD - made INF=0')

temp=abs(astminusfd(:));
for i=1:size(temp)
 if ((temp(i) == Inf) | (temp(i) == -Inf))
  temp(i)=0;
 end
end

maxastm = max(temp)
minastm = min(temp)

disp('100*(ASTROS - FD)/FD')
astminusfddivfd
disp('Max/Min Relative Percentage btwn ASTROS and FD')

temp=abs(astminusfddivfd(:));
for i=1:size(temp)
 if ((temp(i) == Inf) | (temp(i) == -Inf))
  temp(i)=0;
 end
end
maxastr = max(temp)
minastr = min(temp)
disp(' ')

temp=astminusfddivfd(:);
for i=1:size(temp)
  if ((temp(i) == Inf) | (temp(i) == -Inf))
    temp(i) = 0;
  end
end

mean_astr = mean(temp)
std_astr  = std(temp)

disp('MODE is')
modenumber
disp(' ')
disp(' ')
```

```
disp(' ')
disp('ANALYTICAL - FD')
anaminusfd
disp(' ')
disp('Max/Min Difference btwn ANALYTICAL and FD')

temp=abs(anaminusfd(:));
for i=1:size(temp)
 if ((temp(i) == Inf) | (temp(i) == -Inf))
   temp(i)=0;
 end
end
maxanam = max(temp)
minanam = min(temp)

disp('MODE is')
modenumber
disp(' ')
disp(' ')

disp('100*(ANAL - FD)/FD')
anaminusfddivfd
disp('Max/Min Relative Percentage btwn ANAL and FD')

temp=abs(anaminusfddivfd(:));
for i=1:size(temp)
 if ((temp(i) == Inf) | (temp(i) == -Inf))
   temp(i)=0;
 end
end
maxanar = max(temp)
minanar = min(temp)
disp(' ')

temp=anaminusfddivfd(:);
for i=1:size(temp)
  if ((temp(i) == Inf) | (temp(i) == -Inf))
    temp(i) = 0;
  end
end

mean_anar = mean(temp)
std_anar  = std(temp)

disp('MODE is')
```

```
modenumber
disp(' ')
disp(' ')

disp(' ')
disp(' Sign Comparison')
disp(' ')
disp('ANA is in 100s Column; AST in 10s Col; FD in 1s Col')
disp('3:Zero; 2:Negative; 1:Positive')
signcomp

disp('All 3 signs are different:')
notall3
disp('The sign of ANAL and ASTROS is different')
notanaandast
disp('The sign of ANAL and FD is different')
notanaandfd
disp('The sign of ASTROS and FD is different')
notastandfd
disp('The sign of ANAL is different when the sign of')
disp('ASTROS and FD is the same')
ana_noteqto_ast_and_fd
disp(' ')
disp('Number of analytical zeros')
numanazeros
disp(' ')
disp(' ')
disp('Number of ASTROS zeros')
numastzeros
disp(' ')
disp(' ')
disp('Number of FD zeros')
numfdzeros
disp(' ')
disp('MODE is')
modenumber
disp(' ')
disp(' ')


% Check for analytical correlation

anatimes2 = ana*2;
a=abs(anatimes2);
low = abs(fd)*0.75;
```

```
hi  = abs(fd)*1.25;
numhits = 0;
for i=1:12
 for j=1:12
   if ((a(i,j) >= low(i,j)) | (a(i,j) <= hi(i,j)))
       c(i,j) = 'O';
       numhits = numhits+1;
   else
       c(i,j) = 'X';
   end
 end
end
disp(' ')
disp('0 => anal*2 is within 25% of FD; X : it is not')
c
disp('Number hits in range = ')
numhits

% Check if correlated within 50%

low = abs(fd)*0.5;
hi  = abs(fd)*1.5;
numhits=0;
for i=1:12
 for j=1:12
   if ((a(i,j) >= low(i,j)) | (a(i,j) <= hi(i,j)))
       c(i,j) = 'O';
       numhits = numhits+1;
   else
       c(i,j) = 'X';
   end
 end
end

disp(' ')
disp('0 => anal*2 is within 50% of FD; X : it is not')
c
disp('Number hits in range = ')
numhits

disp(' ')
disp(' ')
end; % the modeswap loop
```

Appendix H

H.1 Plots of first ten modes of COFS model as designed for eventual input to ***FRAME.*** Orbiter stick model has been removed but DRA remains. **All** lumped masses are active and the plates are modeled as very stiff beams. Defunct modes are replaced with 3rd bending modes as one would expect.

H-1: 1st Bending in YZ Plane

H-2: 1st Bending in XZ Plane

H-3: 1st Torsion (some breathing due to rigid beams)

H-4: 2nd Bending in YZ Plane

H-5: 2nd Bending in XZ Plane

H-6: 3rd Bending in YZ Plane

H-7: 3rd Bending in XZ Plane

H-8: 2nd Torsion (significantly more breathing than in 1T)

H-9: 4th Bending in YZ Plane

H-10: 4th Bending in XZ Plane

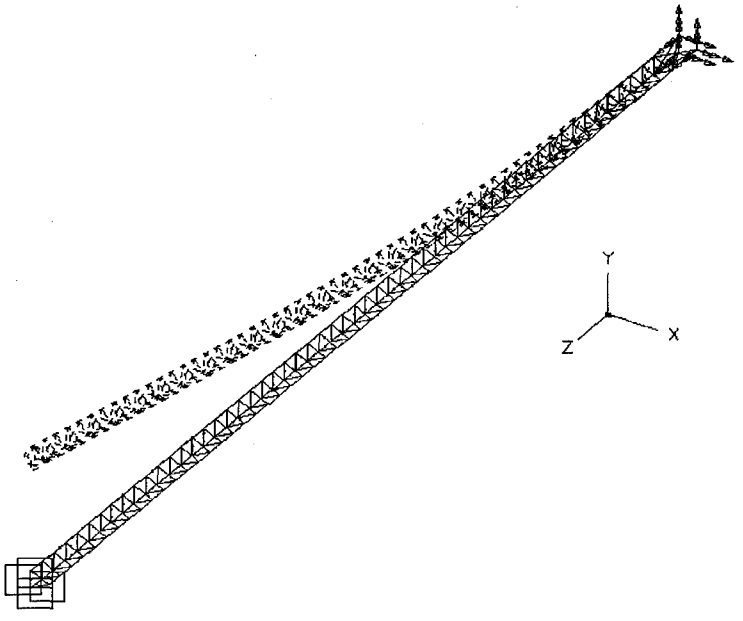


Figure H-1: COFS Mode 1 (1st YZ Bending)

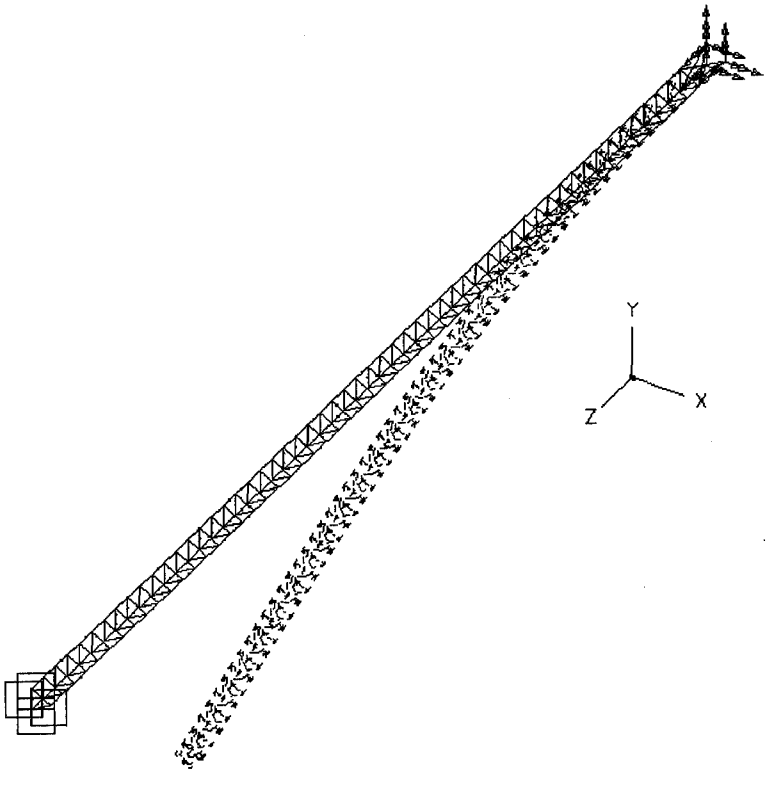


Figure H-2: COFS Mode 2 (1st XZ Bending)

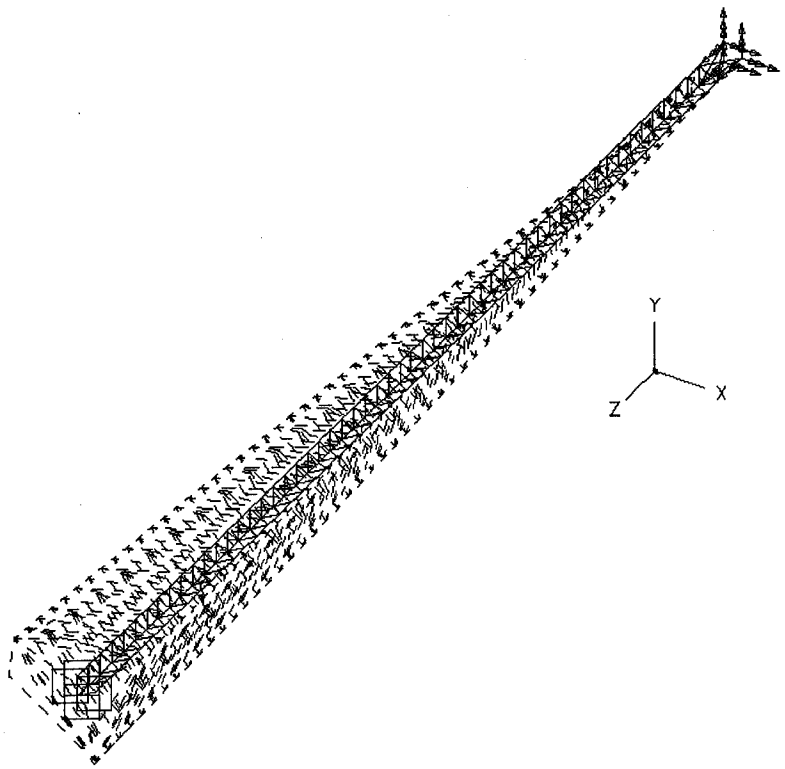


Figure H-3: COFS Mode 3 (1st Torsion)

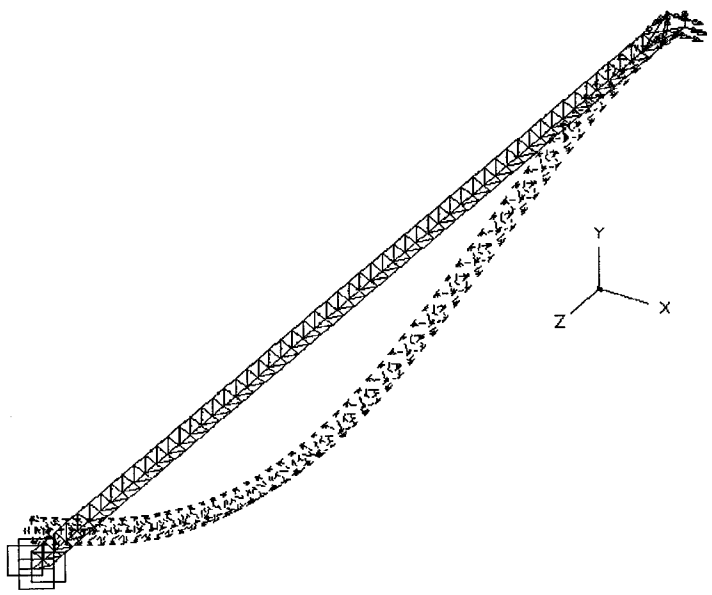


Figure H-4: COFS Mode 4 (2nd YZ Bending)

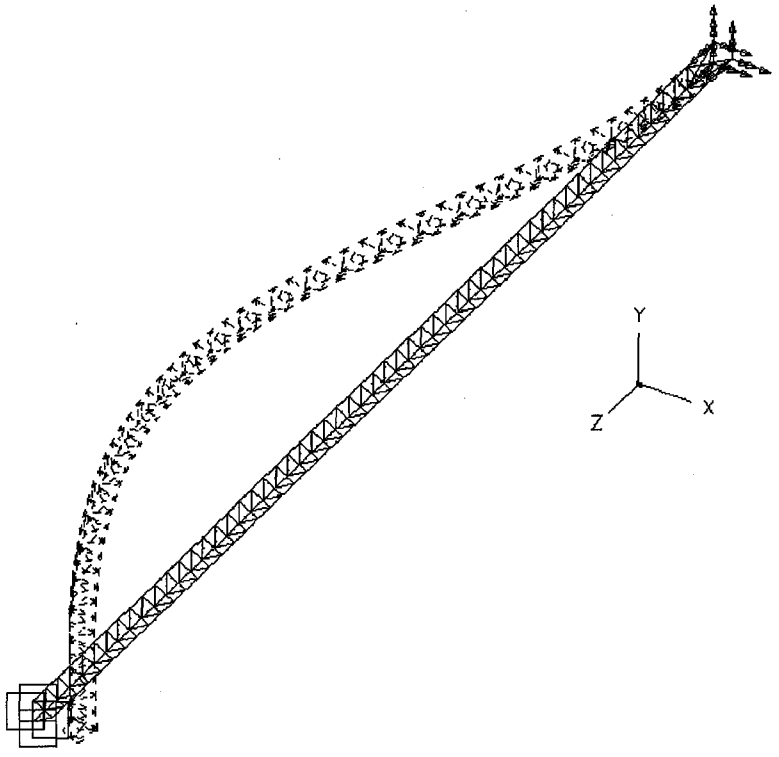


Figure H-5: COFS Mode 5 (2nd XZ Bending)

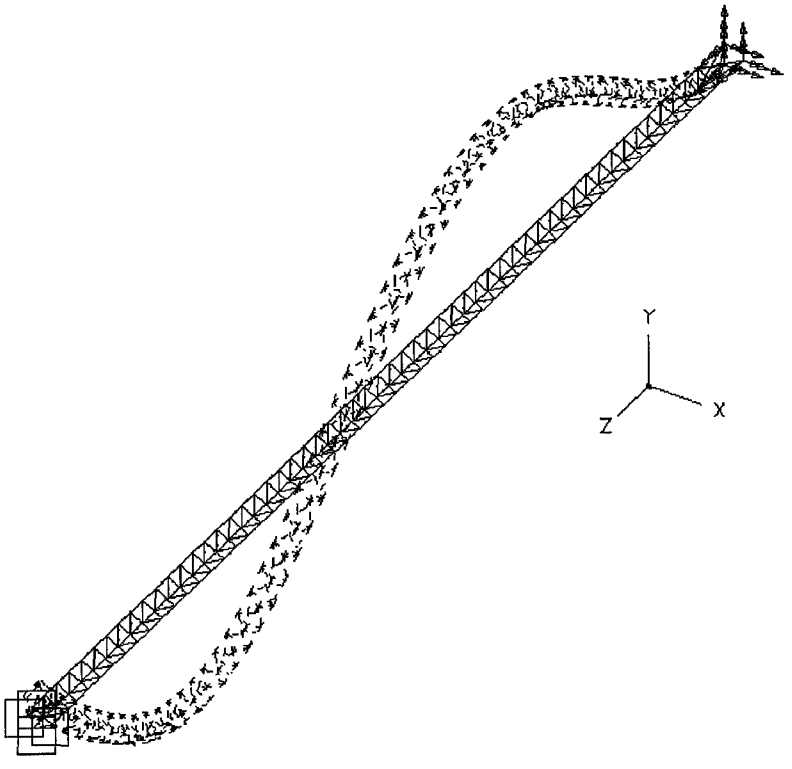


Figure H-6: COFS Mode 6 (3rd YZ Bending)

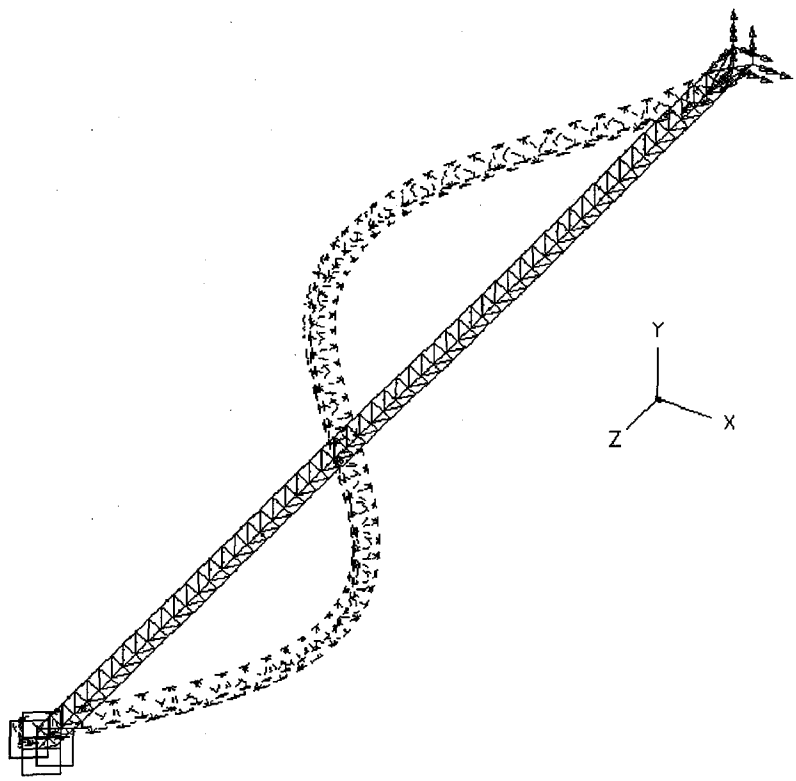


Figure H-7: COFS Mode 7 (3rd XZ Bending)

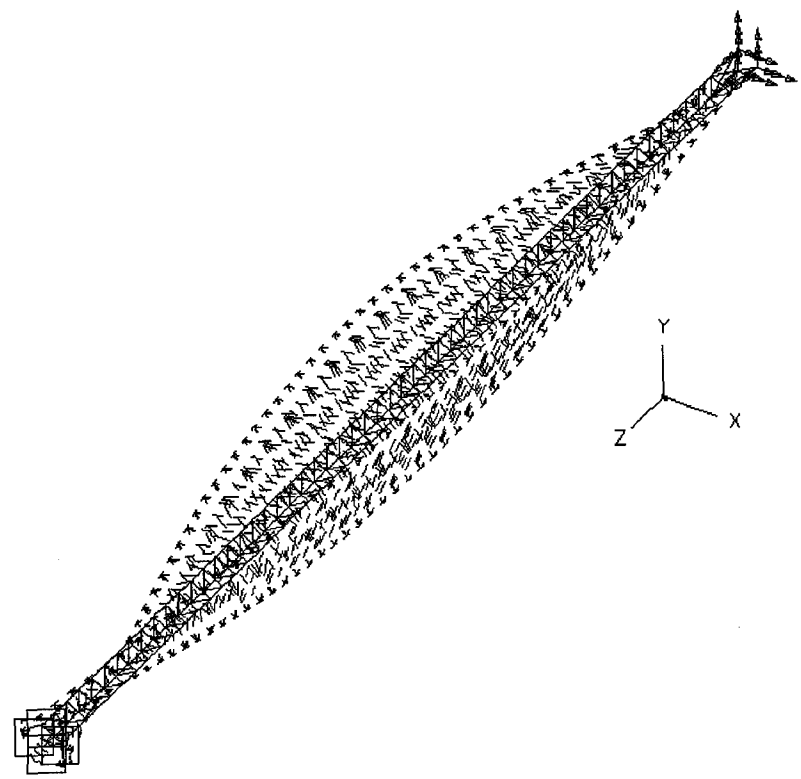


Figure H-8: COFS Mode 8 (2nd Torsion)

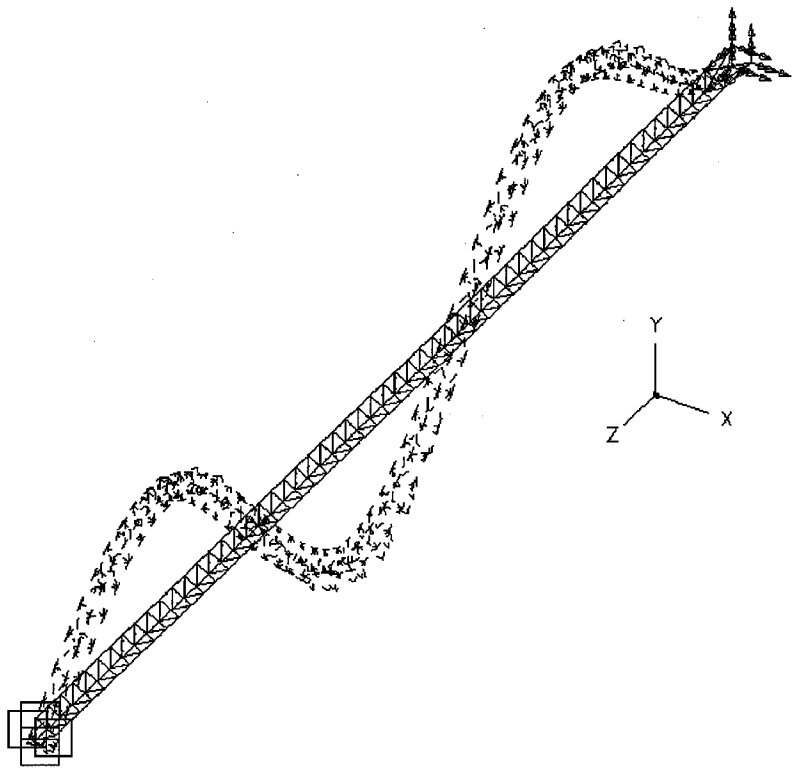


Figure H-9: COFS Mode 9 (4th YZ Bending)

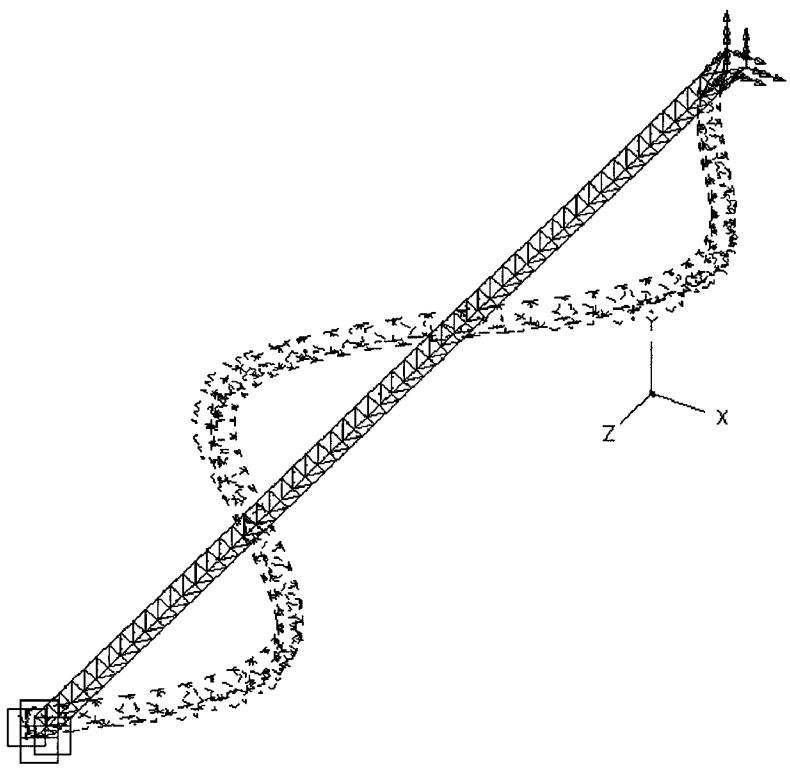


Figure H-10: COFS Mode 10 (4th XZ Bending)

Bibliography

1. Bernard, Michael L. and Allen J. Bronowicki. "Modal Expansion Method for Eigensensitivity with Repeated Roots," AIAA Journal, Vol. 32, No. 7: 1500-1506 (July 1994).

2. Canfield, Robert A. Associate Professor of Aerospace Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio. Personal Lecture on Semi-Analytic Derivatives, 10 Jul 1994.

3. Canfield, Robert A. Integrated Structural Design, Vibration Control, and Aeroelastic Tailoring by Multiobjective Optimization. PhD Dissertation. Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 1992.

4. Colladay, Raymond S. Office of Aeronautics & Space Technology Notice: Control of Flexible Structures (COFS) Technology Program. Washington D.C.: National Aeronautics and Space Administration, 1986.

5. Craig Jr., Roy R. Structual Dynamics: An Introduction to Computer Methods. New York: John Wiley and Sons, 1981.

6. Dailey, R. Lane. "Eigenvector Derivatives with Repeated Eigenvalues," AIAA Journal, Vol. 27, No. 4: 486-491, Apr 1989.

7. Eldred, M. S., V.B. Venkayya, and W.J. Anderson. "Mode Tracking Issues in Structural Optimization," unpublished technical paper, University of Michigan, July 1994.

8. Eldred, M.S., P.B. Lerner, and W.J. Anderson. "Higher Order Eigenpair Pertubations," AIAA Journal, Vol. 30, No. 7: 1870-1876 (July 1992).

9. Fox, R.L. and M. P. Kapoor. "Rates of Change of Eigenvalues and Eigenvectors," AIAA Journal, Vol. 6, No. 12: 2426-2429 (December 1968).

10. Gibson, Warren C. ASTROS-ID: Software for System Identification Using Mathematical Progamming. Contract C-F33615-90-C-3211. Flight Dynamics Directorate, Wright Laboratory, Wright Patterson AFB, September 1992.

11. Grandhi, R.V. and V.B. Venkayya. "Structural Optimization with Frequency Constraints," AIAA Journal, Vol. 26, No. 7: 858-866 (July 1988).

12. Grandhi, Ramana V. "Structural and Control Optimization of Space Structures," Computers and Structures, Vol. 31, No. 2: 139-150 (1989).

13. Hanks, B. R. "Control of Flexible Structures (COFS) Flight Experiment Background and Description," Large Space Antenna Systems Technology 1984. 893-902. December 1984.

14. Horta, Lucas G., Joanne L. Walsh, and Garnett C. Horner. "Analysis and Simulation of the MAST (COFS-I Flight Hardware)," First NASA/DoD Control/Structures Technology Conference. 515-532. November 1986.

15. IMSL, Inc. (1991). IMSL Math/Library ForTran Subroutines for Mathematical Applications. IMSL Problem-Solving Software Systems, Houston.

16. Khot, N.S. "An Integrated Approach to the Minimum Weight and Optimum Design of Space Structures," Large Space Structures: Dynamics and Control. 355-363. New York: Springer-Verlag, 1988.

17. Less Michael C. and Susan Manual. CADS - A Computer Aided Design System Volume II - User's Guide. El Segunda: Rockwell International Inc., 1985.

18. Liebst, Bradley S. "Simultaneous Optimization of Structural Damping and Active Control." Department of Aeronautics and Astronautics, Air Force Institute of Technology, Wright-Patterson AFB.

19. Liebst, Bradley S. Lecture, MECH 722, Control of Flexible Spacecraft. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, Spring Quarter 1994.

20. Meirovitch, Leonard. Methods of Analytical Dynamics. New York: McGraw-Hill, 1970.

21. Mills-Curran, William C. "Calculation of Eigenvector Derivatives for Structures with Repeated Frequencies," AIAA Journal, Vol. 26, No. 7: 867-871 (Jul 1988).

22. Neil, D.J, E. H. Johnson, and R. Canfield. "ASTROS - A Multidisciplinary Automated Structural Design Tool," Journal of Aircraft, Vol. 27, No.12: 1021-1027 (Dec 1990).

23. Nelson, Richard B. "Simplified Calculation of Eigenvector Derivatives," AIAA Journal, Vol. 14, No. 9: 1201-1205 (Sep 1976).

24. Ojalvo, I.U. "Gradients for Large Structural Models with Repeated Frequencies," Aerospace Technology Conference and Exposition, Society of Automotive Engineers. 1-8. Pennsylvania: Society of Automobile Engineers, Inc., 1986.

25. Oz, H., and Khot, N.S., "Structure-Control System Optimization with Fundamental Efficiency Constraint," Proceedings of of the 8th VPI&SU Symposium. 159-169. Virginia: Virginia Polytechnic Institute and State University, 1992.

26. Strang, Gilbert. Linear Algebra and Its Applications (Third Edition). Fort Worth: Harcourt Brace Jovanivich College Publishers, 1988.

27. Structural Dynamics Research Corporation (1990). I-DEAS System Dynamics Analysis™ User's Guide. SDRC Software Products Marketing Division, Milford.

28. Strunce, Robert R. ACOSS4 (Active Control of Space Structures) Theory Appendix. Contract F30602-78-C-0268. NewYork: The Charles Stark Draper Laboratory, Inc., June 1980.

29. Talcott, Ronald C. and John W. Shipley. "Description of the MAST Flight System," NASA/DoD Control Structures Interaction Technology 1986. 253-263. November 1986.

30. Ting, T. "An Automated Mode Tracking Strategy," AIAA-93-1414-CP: 970-976.

31. Vanderplaats, Garret N. Numerical Optimization Techniques for Engineering Design. New York: Mc-Graw Hill, Inc., 1984.

32. Walsh, Joanne L. "60 Meter Flight Truss. EAL Input Stack and Output File for COFS1 Model." (Program Output). May 1985.

33. Walsh, Joanne L. "Optimization Procedures to Control the Coupling of Vibration Modes in Flexible Space Structures," AIAA/ASME/ASCE/AHS 28th Structures, Structural Dynamics and Materials Conference. 1-29. Virginia: Langley Research Center, April 1987.

Vita

Robert Costa Jr. was born on 3 February 1965 in South Dartmouth, Massachusetts. Bob earned a regular commission through the AFROTC program at Boston University in 1987, where he received his Bachelor of Science Degree in Aeronautical Engineering. He declined an invitation to pursue his Master's Degree on a teaching fellowship at BU, and instead entered active duty as a 2nd Lieutenant at Falcon AFB in Jan 88.

Bob's first assignment was as an operations requirements officer for 2d Space Wing (now the 50th Operations Group) where he prioritized resources for AFSPACECOM's satellite tracking network. He then transferred to the 3rd Space Operations Squadron at Falcon where he served as a crew commander for the Defense Satellite Communications System Phase III (DSCS III) satellite program. Bob also served as the lead DSCS III vehicle engineer and as Chief, DSCS III Evaluations. After 5 1/2 years, he left Falcon AFB to pursue a Master of Science Degree in Astronautical Engineering at the Air Force Institute of Technology at Wright-Patterson AFB.

Bob married Miss Kimberly A. Pelletier of Caribou, Maine in January 1992. They have no children.