A COMPARISON OF THE OPTIMIZATION AND ANALYSIS OF DOUBLY CURVED SHELLS USING MSC/NASTRAN AND ASTROS

THESIS

John Richard Dewsnup
Captain, USAF

AFIT/GA/ENY/90D-1

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

Wright-Patterson Air Force Base, Ohio
A COMPARISON OF THE OPTIMIZATION AND ANALYSIS OF DOUBLY CURVED SHELLS USING MSC/NASTRAN AND ASTROS

THESIS
John Richard Dewsnap
Captain, USAF

AFIT/GA/ENY/90D-4

Approved for public release; distribution unlimited.
A COMPARISON OF THE OPTIMIZATION AND
ANALYSIS OF DOUBLY CURVED SHELLS
USING MSC/NASTRAN AND ASTROS

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

John Richard Dewsnup, B.S.
Captain, USAF

December, 1990

Approved for public release; distribution unlimited.
Acknowledgments

The completion of any difficult project is a reflection of the investment others have made in our lives. The real work on this Thesis was done by my parents Richard and Alice Dewsnap who gave the knowledge to succeed and faith to persevere, my wife Liz who gave the emotional support necessary to stand up under pressure, my children Hannah, Aaron and Abigail who were a constant reminder of what’s really important in life, my grandfather Major Stanley Dewsnap who imparted a love for engineering, and The Allmighty who gave life itself.

I am also indebted to Professor Howard Gans for leading along the way and helping me avoid numerous pitfalls, and to Professors William Elrod and Bradley Liebst for serving on my committee and providing good advice and guidance.

Due a special note of thanks are Ray Kolonie, who helped me understand the complexities of ASTROS, and my friend, Captain Ken Ryder, who was a sounding board for ideas and frustrations. My close friend Emanuel Davidson invested hours with me sifting through data, planning strategy and proofing the text. To him I will remain eternally indebted.

John Richard Dewsnap
# Table of Contents

<table>
<thead>
<tr>
<th>Acknowledgments</th>
<th>i</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table of Contents</td>
<td>ii</td>
</tr>
<tr>
<td>List of Figures</td>
<td>iv</td>
</tr>
<tr>
<td>Abstract</td>
<td>vi</td>
</tr>
<tr>
<td><strong>I. INTRODUCTION</strong></td>
<td>1</td>
</tr>
<tr>
<td>Purpose And Objectives</td>
<td>1</td>
</tr>
<tr>
<td>Literature Search</td>
<td>1</td>
</tr>
<tr>
<td>Literature In Optimization:</td>
<td>1</td>
</tr>
<tr>
<td>Literature In Plate And Shell Structures:</td>
<td>2</td>
</tr>
<tr>
<td>Literature In Nozzles:</td>
<td>2</td>
</tr>
<tr>
<td>Literature In Finite Elements And Software:</td>
<td>3</td>
</tr>
<tr>
<td><strong>II. OPTIMIZATION BACKGROUND</strong></td>
<td>5</td>
</tr>
<tr>
<td>The General Optimization Problem</td>
<td>5</td>
</tr>
<tr>
<td>Methods For Moving About The Design Space</td>
<td>8</td>
</tr>
<tr>
<td>Method Of Feasible Directions:</td>
<td>9</td>
</tr>
<tr>
<td>Optimality Criteria Method:</td>
<td>10</td>
</tr>
<tr>
<td>Optimization In The Context Of Finite Elements</td>
<td>13</td>
</tr>
<tr>
<td><strong>III. ASTROS AND MSC/NASTRAN COMPARED</strong></td>
<td>17</td>
</tr>
<tr>
<td>Software Differences</td>
<td>17</td>
</tr>
<tr>
<td>Comparison With Exact Solutions</td>
<td>21</td>
</tr>
<tr>
<td>Round Plate:</td>
<td>21</td>
</tr>
<tr>
<td>Plane-strain Cylinder:</td>
<td>23</td>
</tr>
</tbody>
</table>
**List of Figures**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Model of Clamped Round Plate With Ten Elements in Radial Direction</td>
<td>22</td>
</tr>
<tr>
<td>2.</td>
<td>Plot of Center Displacement Versus Grid Refinement, Round Plate</td>
<td>22</td>
</tr>
<tr>
<td>3.</td>
<td>Plot of First Natural Frequency Versus Grid Refinement, Round Plate</td>
<td>23</td>
</tr>
<tr>
<td>4.</td>
<td>Model Of A Plane Strain Tube With 32 Circumferential Elements</td>
<td>24</td>
</tr>
<tr>
<td>5.</td>
<td>Plot of Mid-plane Radial Displacement Versus Grid Refinement, Plane-strain Tube</td>
<td>25</td>
</tr>
<tr>
<td>6.</td>
<td>Plot of Hoop Stress Versus Grid Refinement, Plane-strain Tube</td>
<td>25</td>
</tr>
<tr>
<td>7.</td>
<td>Uniform Spherical Dome With Rigidly Fixed Edges Subjected To Uniform Pressure</td>
<td>27</td>
</tr>
<tr>
<td>8.</td>
<td>Model Of A Clamped Dome With 5 Elements In The Radial Direction</td>
<td>27</td>
</tr>
<tr>
<td>9.</td>
<td>Plot of Mid-plane $\sigma_\theta$ Versus Angle $\phi$, Clamped Dome</td>
<td>28</td>
</tr>
<tr>
<td>10.</td>
<td>Plot of Mid-plane $\sigma_\theta$ Versus Angle $\phi$, Clamped Dome</td>
<td>29</td>
</tr>
<tr>
<td>11.</td>
<td>Plot of Top Surface $\sigma_\theta$ Versus Angle $\phi$, Clamped Dome</td>
<td>29</td>
</tr>
<tr>
<td>12.</td>
<td>Plot of Top Surface $\sigma_\theta$ Versus Angle $\phi$, Clamped Dome</td>
<td>30</td>
</tr>
<tr>
<td>13.</td>
<td>Iteration History Of Wall Thickness, Plane-strain Tube</td>
<td>31</td>
</tr>
<tr>
<td>14.</td>
<td>Finite Element Model Of Quartered Nozzle, Grid Fineness Of 32 Elements Per Circumference</td>
<td>33</td>
</tr>
<tr>
<td>15.</td>
<td>Finite Element Model Of Nozzle Using Single Element Width Strip, Grid Fineness Of 32 Elements Per Circumference</td>
<td>33</td>
</tr>
<tr>
<td>16.</td>
<td>Nozzle Pressure Loads</td>
<td>35</td>
</tr>
<tr>
<td>17.</td>
<td>Nozzle Temperature Profile</td>
<td>36</td>
</tr>
<tr>
<td>18.</td>
<td>Comparison Of Optimization Progress Using Default And Modified Optimization Parameters In MSC/NASTRAN, Nozzle Model With Grid Fineness Of 32 Elements Per Circumference</td>
<td>37</td>
</tr>
<tr>
<td>19.</td>
<td>Comparison Of Optimization Progress Using Default And Modified Optimization Parameters In ASTROS, Nozzle Model With Grid Fineness Of 32 Elements Per Circumference</td>
<td>38</td>
</tr>
<tr>
<td>20.</td>
<td>Comparison Of Optimization Progress Using MSC/NASTRAN and ASTROS, Nozzle Model With Grid Fineness Of 32 Elements Per Circumference</td>
<td>39</td>
</tr>
</tbody>
</table>


24. Principle Stress Distribution Resulting From Thermal Loads Using MSC/NASTRAN, Nozzle Model With Grid Fineness Of 32 Elements Per Circumference ..................... 42

25. Conical Shell Element Cross Section ................................................................. 44
Abstract

This study identified techniques and software available for the optimization of doubly curved shells and applied them in the context of a large nozzle shape. An optimality criteria scheme that can reduce solution time was evaluated and compared to the Method of Feasible Directions. MSC/NASTRAN and ASTROS were used to perform finite element analysis and optimization, and the results were compared to theory. The programs give virtually identical results, and if plates and shells are carefully modeled, then stresses, displacements and modes are accurate to within ten percent. A Mindlin-type axisymmetric finite element was implemented in ASTROS that preserved accuracy and reduced the size of the stiffness matrix by a factor of four.

Nozzle optimization was performed using static pressure and thermal loads, constrained by the Von Mises stress criteria. Software errors in ASTROS were documented, and four characteristic stress regions identified for the optimized nozzle.
A COMPARISON OF THE OPTIMIZATION AND ANALYSIS OF DOUBLY CURVED SHELLS USING MSC/NASTRAN AND ASTROS

I. INTRODUCTION

Purpose And Objectives

Rigorous optimization of aerospace systems is required to extend range, reduce costs, and meet performance objectives. The trend toward complex, high performance systems requires early system level multidisciplinary optimization. This has driven the implementation of multidisciplinary optimization schemes in finite element analysis software.

Doubly curved shell structures are a key element in aerospace systems. Their high strength has led to use in aircraft skins, rocket fuel tanks and nozzles, among other things. The purpose of this study is to identify techniques available for the optimization of doubly curved shells and apply them to an aerospace structure. This will be done by researching robust optimization algorithms and the software that implements them. The software will be tested for accuracy, then used to optimize the wall thickness of a large nozzle.

Literature Search

Literature In Optimization: Structural weight optimization is important to the design of Aerospace vehicles since weight has a great impact on vehicle performance. Optimization is generally performed in the context of constraints, and various methods have been developed to deal with them. The constrained optimization problem may be changed to an unconstrained problem by transforming the cost and constraint functions into a new cost function that penalizes constraint
violations. The problem is then optimized using unconstrained techniques. Arora (2) details techniques that fall into this category, and their inherent instabilities near constraint boundaries. More popular and powerful methods use constraints directly in the optimization process using Lagrangian multipliers. Miura (17) describes a well conditioned Modified Feasible Direction algorithm that is faster than steepest descent methods. Venkayya (26, 27) details an Optimality Criteria Method that resizes or scales the design based on Kuhn-Tucker conditions and the location of the constraints.

*Literature In Plate And Shell Structures:* In general, a plate structure is defined as the solid material enclosed between two closely spaced planer surfaces, and a shell structure as the material between two curved surfaces (7:Gibson). The utility of plate and shell structures is illustrated in creation. Insect exoskeletons are commonly made of thin shells, and plate like structures are seen in the plant world. Plate and shell structures have a high strength to weight ratio that is useful in aerospace applications. Nozzles are one important application of doubly curved shells.

Closed form solutions to the structure and vibration problem exist for a number of simple plate and shell structures. Timoshenko (23) has solved for stresses and displacements in a uniformly loaded round plate with uniform thickness and clamped edges. Meirovitch (16) illustrates the derivation of natural frequencies and mode shapes for this problem. A spherical dome with rigidly fixed edges is another doubly curved shell structure for which a closed form solution exists. Timoshenko (23) illustrates both the derivation of the exact solution and a simpler approximate solution which was first developed by Hetenyi. Gibson (7) illustrates the FORTRAN coding of the approximate solution.

*Literature In Nozzles:* Supersonic convergent/divergent nozzles are used in a variety of applications including rocket and turbine engines and supersonic wind tunnels. Nozzle shape and length may be optimized based on the characteristics of the flow and mission profile. Oates (19) illustrates the relationship between expansion ratio and thrust at different altitudes. The nozzle's structural design is driven by material strength and design stability, subject to pressure loading.
and other mission loads. By assuming one dimensional flow, the pressure loading may be approximated using the Laval nozzle equations. Kuethe and Chow (12) detail the development of these equations. Mission loads may include transient thermal effects from the gas flow, thermal effects from nuclear bursts, erosive effects, and a vibration spectral density environment. For instance, erosion is the primary concern in analyzing solid propellant rocket nozzles, followed by thermal stresses and conductivity (6:Galati). The steady state operating temperature of the nozzle is an important consideration for liquid propellant rocket engines. Barrere, Jaumotte, DeVeubeke and Vandenkerckhove (3) illustrate methods developed by Ciniaref and Dobrovolski for calculating these temperatures in a liquid propellant rocket nozzle with regenerative cooling.

**Literature In Finite Elements And Software:** With increasing demands for vehicle performance there arose the need for multidisciplinary optimization early in the design process. In the past, large software packages performed analysis in one discipline at a time. It was up to the user to collect the necessary design sensitivity information from the various disciplines and put them in a form that could be optimized. The Air Force Wright Aeronautical Laboratories initiated development of ASTROS, short for Automated Structural Optimization System. This is a large software package that collects static, modal, aerodynamic and dynamic analysis together for multidisciplinary optimization (10:Johnson). The MacNeal Schwendler Corporation developed a new MSC/NASTRAN solution sequence, SOL 200, that collects static, dynamic, and buckling analysis together for the same purpose (17:Miura).

Finite elements for shells have been difficult to develop. Curved elements closely model the physical situation but are complicated. Modeling the shell as a set of finely faceted flat elements is less complex and works well enough to compete with curved elements (4:Cook). An alternative for problems with axial symmetry is an axisymmetric shell element. Cook details the development and accuracy of a Mindlin type axisymmetric shell element with membrane, transverse shear and bending stiffness. MSC/NASTRAN (14:MacNeal) incorporates a conical axisymmetric shell ele-
ment with all but drilling degrees of freedom at each nodal ring. A Fourier analysis scheme is used to obtain the response to non-axisymmetric loads.
II. OPTIMIZATION BACKGROUND

The General Optimization Problem

Optimization has been pursued in a variety of disciplines, but regardless of the discipline the optimization problem can be reduced to a general form. Numerous mathematical optimization techniques have been developed based on this standard form, which is defined by Arora (2) as follows:

Given the design variables

\[ x = (x_1, ..., x_n) \]  

and the function

\[ F(x) = F(x_1, ..., x_n) \]  

find the \( x \) that will minimize \( F(x) \) subject to the following constraints:

\[ z_j(x_1, ..., x_n) \leq 0 \quad j = 1 \ldots k \]  
\[ z_j(x_1, ..., x_n) = 0 \quad j = k + 1 \ldots l \]  
\[ x_j - x_j^{\text{upper}} \leq 0 \quad j = 1 \ldots n \]  
\[ -x_j - x_j^{\text{lower}} \leq 0 \quad j = 1 \ldots n \]

Equation 3 represents the \( k \) inequality constraints, Equation 4 the \( l - k \) equality constraints, and Equations 5 and 6 represent the side constraints on the design variables. The function \( F(x) \) may be referred to as the cost or objective function (2:Arora). In general, the objective function and constraints may be nonlinear, implicit functions of the design variables.

The design variables define an \( n \) dimensional design space. Any arbitrary \( x' \) is a design. If the \( x' \) produces inequality constraints that are more positive than some user defined constant \( \epsilon \), or equality constraints that are not zero to within \( \pm \epsilon \), the constraints are violated. Similarly, inequality constraints between zero and \( -\epsilon \), and equality constraints between \( \pm \epsilon \) are considered satisfied and
active. Inequality constraints that are more negative than $-\varepsilon$ at the design are characterized as satisfied and inactive. The activity or inactivity of a constraint relates to its involvement in moving about the design space. At a given design, active constraints provide important information about the direction toward the optimum. Inactive constraints remain part of the problem but the information they provide is less critical. For numerical solutions the user defined variable $\varepsilon$ is necessary because in a practical sense it is impossible to exactly satisfy a constraint. Without some finite $\varepsilon$, all the constraints would be characterized as violated or inactive, and the optimization routines would have to act on all the variables. This would be inefficient.

Given the design $\mathbf{z}^v$, we may characterize it as feasible or infeasible based on the condition of the constraints. If there are any violated inequality or equality constraints, and the design does not lie within the bounds specified by the side constraints, then the design is infeasible. On the other hand, if all the constraints are satisfied then the design is feasible. A feasible design may have some active constraints, in which case it is a constrained optimization problem. Otherwise it is an unconstrained problem. At any given $\mathbf{z}^v$ there are three possible design states:

1. The design is feasible and there are no active constraints.
2. The design is feasible with one or more active constraints.
3. The design is infeasible since one or more of the constraints are violated.

The design state is important in determining the action to be taken during the optimization process.

The intent of the design optimization process is to find the feasible design $\mathbf{z}^*$ that minimizes the objective function. To do this, the conditions that must exist at the minima need to be identified. If the minima is bounded and the problem unconstrained, then the familiar calculus concept of the vanishing gradient at the minima is a necessary condition, or:

$$\nabla F(x) = \left[ \frac{\partial F(x)}{\partial x_1}, \ldots, \frac{\partial F(x)}{\partial x_n} \right] = 0$$ (7)
Arfken (1) has demonstrated that if the design at the minima includes active constraints, then the \( \frac{\partial F(x)}{\partial x_i} \) are no longer arbitrary and in general do not vanish. This leads to formulation of the Lagrangian Function.

If we define some vector \( S \) to be an arbitrary unit direction vector in \( n \) space, then the dot product of \( \nabla F(x^r) \) and \( S \) yields the rate of change of the objective function in direction \( S \). If the dot product is negative then the objective function is getting smaller in that direction, and if positive it is increasing. If zero, the gradient and \( S \) are orthogonal and there is nothing to be gained by moving positive or negative in that direction. Applying the same thought to the constraints, if the dot product of \( \nabla z_j(x) \) and \( S \) is zero, then \( S \) lies on some iso-potential surface of the constraint. If the constraint is active then this direction is tangent to the constraint boundary.

Let us assume a constraint is active and direction \( S \) is tangent to its boundary. If the dot product of the objective function and \( S \) is nonzero, the objective function can be reduced by moving along \( S \). If this is the case then the design cannot be a minimum for the objective function can be reduced without violating the constraint. However, if the dot product is zero then no move from the present design that remains in the feasible region will reduce the objective function; hence the present design is at least a local minimum. Also, since the dot product of \( S \) with the gradients is zero, the gradients must be collinear. Therefore an arbitrary \( \lambda \) can be chosen such that

\[
\nabla F(x^r) - \lambda \nabla z(x^r) = 0,
\]

which introduces the concept of the Lagrangian function and Lagrangian multipliers.

As Venkayya (26) has demonstrated, the constrained optimization problem is typically reformulated with a Lagrangian function \( L(x, \lambda) \), defined as:

\[
L(x, \lambda) = F(x) - \sum_{j=1}^{p} \lambda_j z_j(x)
\]

(8)

The \( z_j(x) \) are the \( p \) active constraints and the \( \lambda \)'s are the Lagrangian multipliers corresponding to these constraints. At the minima the gradients of the objective function and the constraints are
collinear and there exists some collection of Lagrangian multipliers that will nullify the sum of the gradients, or:

\[ \frac{\partial L}{\partial x_i} = \frac{\partial F}{\partial x_i} - \sum_{j=1}^{p} \lambda_j \frac{\partial z_j}{\partial x_i} = 0 \quad i = 1, \ldots, n \]  

(9)

The system of Equations at 9 represent the necessary conditions of optimality and are known as the Kuhn-Tucker conditions. If we define a matrix \( e \) such that

\[ e_{ij} = \frac{\partial z_j}{\partial x_i} \]  

(10)

then the Kuhn-Tucker conditions may be reformulated as:

\[ \sum_{j=1}^{p} e_{ij} \lambda_j = 1 \quad i = 1, \ldots, n \]  

(11)

If the Kuhn-Tucker conditions are met and the constraints are satisfied, then the design has at least reached a local minima. If the objective function and inequality constraints are convex, and equality constraints are linear, then this is also the global minimum (2:Arora). Alternatively, a rigorous search through the feasible design space in search of other global minima candidates may satisfy the user that the global minimum has been reached.

Methods For Moving About The Design Space

The design optimization process requires moving through the design space from some \( x' \) in search of the \( x^* \) that minimizes \( F(x) \). Much work has gone into developing algorithms for performing this task. One common approach is known as the Method of Feasible Directions (17:Miura). Its robustness and efficiency have led to its implementation in large structural analysis software packages such as MSC/NASTRAN and ASTROS. This method follows a two-phase approach; first determine a search direction, then determine the distance to step in that direction. A less common approach is defined as the Optimality Criteria Method (26:Venkayya). It is also a two-phase method.
which either resizes the $z^r$ toward the optimum or scales it toward the constraint boundaries. The decision to resize or scale is based on the design state.

Method Of Feasible Directions: In the Method of Feasible Directions (17:Miura), the design $z^r$ is iteratively updated by the expression

$$z^{r+1} = z^r + a^* S \tag{12}$$

where $S$ is a unit pointing vector used to identify a search direction and the $a^*$ is a scalar move parameter that defines the distance to move along $S$. The first critical step in the optimization task is to find the search direction $S$. If the design is feasible and there are no active constraints, then the search direction that reduces the objective function most rapidly is $S = -\nabla F$. This is known as the steepest descent direction. Experience has shown that convergence is greatly enhanced if $S$ is modified towards the trend direction of the last several iterations. If the design is feasible but there are active constraints, then the steepest descent direction will likely violate a constraint. In this case a sub-optimization must be performed to find an $S$ that minimizes $\nabla F(z^r) \cdot S$ subject to the constraints $\nabla z_j(z^r) \cdot S \leq 0$, where the $z_j$ are the active constraints at the design. If the current design is infeasible, then a sub-optimization must be performed as above but with the objective function penalized and the constraint relaxed proportional to the magnitude of the constraint violation. This allows $S$ to point back toward the feasible region.

After a suitable search direction has been chosen, the scalar step distance $a^*$ must be determined. This is done by a one dimensional search. By sampling the $F(z)$ and $z_j(z)$ and their gradients along $S$, interpolation may be used to select an $a^*$ that activates a new constraint or minimizes the objective function. With a suitable step direction and distance in hand, the design is updated by Equation 12.

At this point the third critical element of the Method of Feasible Directions comes into play;
detection of convergence to the optimum. Failure to find a feasible search direction $S$ while in possession of a feasible design indicates that the Kuhn-Tucker necessary conditions for optimality are satisfied and the current design has reached an optimum. Another criteria is that of having the relative or absolute change in the objective function be below some user specified tolerance. The optimization must also be terminated if after a reasonable number of iterations a feasible design cannot be found. If convergence is detected by one of these criteria then the optimization process is stopped. Otherwise a new search direction and step parameter are sought and the process goes on.

**Optimality Criteria Method:** In the Optimality Criteria Method (26, 27: Venkayya) the design $x^r$ is iteratively updated by one of the following expressions:

\[ x^{r+1}_i = x^r_i \left[ \sum_{j=1}^{p} c_{ij} \lambda_j \right]^{\frac{1}{2}} \quad i = 1, \ldots, n \quad (13) \]

\[ x^{r+1}_i = A_i x^r_i \quad i = 1, \ldots, n \quad (14) \]

Equation 13 resizes the $x^r$ by stepping toward the optimum design and Equation 14 scales the design by stepping toward constraint boundaries. The $A_i$ in Equation 14 is a scale factor chosen to modify design variable $i$ so as to bring the design to a constraint boundary.

The resizing scheme is based on the Kuhn-Tucker conditions expressed in Equation 11. The bracketed sum at Equation 13 equals 1 at the optimum design, indicating satisfaction of the Kuhn-Tucker conditions and producing no change in $x^r$. If the design is not at the optimum then the bracketed summation can be thought of as the ratio of the $z^r_i$ at the optimum and the current $z^r_i$. Since the objective function and constraints are typically nonlinear, the relationship will not be exact, but it provides a factor that will move the design in the right direction. The "twiddle" factor $\alpha$ is used to soften the magnitude of the move and prevent oscillations. It is adjusted throughout the optimization process. Large values of $\alpha$ increase the number of iterations but provide smoother
convergence. An $\alpha < 1$ speeds up the iteration but may miss the optimum. Experience indicates that using an $\alpha$ value of 2 is good in the early iterations but to smooth convergence it should be increased to 3 or 4 as the optimum is approached. For the bracketed summation to accurately reflect the Kuhn-Tucker conditions it would seem necessary to know what constraints are active at the optimum and the values of the Lagrangian multipliers. But Venkayya (26) has noted that the optimization process is a series of iterations based on approximations, and the approximations will eventually converge to the appropriate constraint set and Lagrangian multipliers. He then details a technique for getting good first approximations for the $\lambda$'s using only gradients of the objective function and constraints, weighted by each design variable's contribution to the objective function.

The scaling scheme in Equation 14 moves the design toward constraint boundaries by applying a scale factor $A_i$ to each design variable $x_i$, as in Equation 14. Each design variable may have a number of candidate scale factors to choose from since each one may in general affect each constraint, and the constraints may be violated or satisfied to differing degrees. Venkayya and Tischler (27) have developed a compound scaling algorithm that uses constraint function gradients, constraint function values, and design variable values in a weighting scheme that selects the proper factor. Two key quantities are used in compound scaling; candidate scale factors and the relative sensitivity of the design variables to the constraints.

In developing candidate scale factors it is important to note that the constraints at Equations 3 through 6 are a generalization of constraints which are typically expressed in some variation of:

$$z_j(x_1, \ldots, x_n) \leq \bar{z}_j$$

(15)

The $\bar{z}_j$ is some constraint specified by the user in formulating the problem, such as a limitation on stress or displacement. If we define a parameter $\beta$ as

$$\beta_j = \frac{\bar{z}_j}{z_j}$$

(16)
then two candidate scale factors are:

\[
\Lambda_j = \left[ \frac{1}{\beta_j} \right]^{\frac{1}{2}}
\]

(17)

\[
\Lambda_j = [\beta_j]^{\frac{1}{2}}
\]

(18)

The \( \alpha \) is again a "twiddle" factor used to control convergence. Either scale factor may apply to the design variable. The sign of \( \frac{\partial z_i}{\partial z_j} \) is important in scale factor selection since it indicates the direction the constraint equation moves in response to the variable. The sign of the \( z_j \) is also important for it indicates what direction the constraint needs to move; whether from the violated or feasible region. In order to determine whether \( \Lambda_j \) or \( \Lambda_j \) is appropriate for a particular constraint, a new parameter is defined:

\[
\mu_{ij} = \frac{\partial z_j}{\partial z_i} z_i
\]

(19)

The sign of the parameter determines which scale factor to use. If for constraint \( j \) and design variable \( i \) the parameter \( \mu_{ij} \) is positive, then \( \Lambda_j \) is the correct scale factor, otherwise \( \Lambda_j \) should be used. Applying this process to all the design variables leads to generation of a scale factor table. This is the collection of candidate factors for each design variable based on the relationship of that variable to each constraint. It now remains to select the appropriate candidate for each design variable.

In order to make a proper selection, the design variables most sensitive to each constraint must be identified. This is done by defining a new parameter

\[
t_{ij} = \frac{\mu_{ij}}{\bar{\mu}_j}
\]

(20)

where \( \bar{\mu}_j \) is the maximum \( \mu_{ij} \) for constraint \( z_j \). The magnitude of parameter \( \mu_{ij} \) provides a measure of the sensitivity of constraint \( z_j \) to percentage changes in variable \( z_i \). The new parameter \( t_{ij} \) normalizes this sensitivity for each constraint. The design variables most sensitive to the constraints
are identified by a 1 and the remaining are scaled with respect to these. The largest $t_{ij}$ associated with design variable $z_i$ identifies the constraint $z_j$ that is most reactive to changes in the design variable, and thus the appropriate scale factor. With this in hand, the expression in Equation 14 can be used to update the design. The entire scaling process is essentially bookkeeping. It requires collecting the products and ratios of gradients, design variables, and constraint values into tables, then applying rules to those tables in order to select scale factors.

The decision to scale or resize is based on the state of the design. If the design is unconstrained or infeasible, then it should be scaled toward the constraints. If it is constrained then it should be resized toward the optimum. After updating the design by Equation 13 or 14, a test for convergence is applied as in the Method of Feasible Directions. This will lead to termination of the optimization process or a new scale or resize decision.

Optimization In The Context Of Finite Elements

The finite element model of a structural system is typically formulated as follows: Let $z$ represent the $n$ component vector of design variables for the system. Let $U$ and $P(z)$ be $I$ component vectors representing the generalized displacements and loads at nodal points of the system. An equilibrium equation is formulated as

$$[K(z)] U = P(x) \quad (21)$$

Where $[K(z)]$ is an $I$ by $I$ matrix called the stiffness matrix. The stiffness matrix and, in general, the load vector are functions of the design variables. The displacements are obtained by inverting the stiffness matrix and manipulating Equation 21 to the form:

$$U = [K(z)]^{-1} P(x) \quad (22)$$
Stress, strain, and other responses are then obtained by the expression

\[ R = [S(z)]U = [S(z)][K(z)]^{-1}P(z) \]  

(23)

where \( R \) is the response vector of interest and \([S(z)]\) is a response recovery matrix that relates responses to displacements. Like the stiffness matrix, the response recovery matrix is a property of the structural system and depends explicitly on the design variables, material properties and geometry of the system. Because of the inverted stiffness matrix in Equations 22 and 23, the explicit functional form of \( U \) and \( R \) cannot generally be written; they are implicit functions of the design variables. Since constraints are placed on displacements and other responses, our \( z_j \) in Equations 3 and 4 are implicit functions of the design variables. This means that to use the finite element model in the optimization process requires the inversion of the stiffness matrix at each new sample point. This is a very inefficient and impractical approach since the stiffness matrix for most practical design problems is extremely large, and each element may have many design constraints applied to it.

The optimization problem is made tractable by linearizing the finite element model about the current design and ignoring constraints that are not likely to affect the design at that point. Efficient design sensitivity analysis procedures have been developed for calculating derivatives of implicit functions with respect to the design variables (2:Arora). With values of the objective function, constraints, and gradients at design \( z_0 \), where \( z_0 = (x_{o1}, \ldots, x_{on}) \), an approximate model of the optimization problem may be stated in the form of Equations 1 through 6 as follows:

Given the design variables

\[ x = (x_1, \ldots, x_n) \]  

(24)
and the objective function

\[
F(x) = F(x_0) + \left[ (x_1 - x_{o1}) \frac{\partial F(x)}{\partial x_1} + \cdots + (x_n - x_{on}) \frac{\partial F(x)}{\partial x_n} \right]
\]  

(25)

find the \( x \) that will minimize the objective function, subject to the following constraints:

\[
z_j(x_o) + \left[ (x_1 - x_{o1}) \frac{\partial z_j(x)}{\partial x_1} + \cdots + (x_n - x_{on}) \frac{\partial z_j(x)}{\partial x_n} \right] \leq 0 \quad j = 1 \ldots k
\]  

(26)

\[
z_j(x_o) + \left[ (x_1 - x_{o1}) \frac{\partial z_j(x)}{\partial x_1} + \cdots + (x_n - x_{on}) \frac{\partial z_j(x)}{\partial x_n} \right] = 0 \quad j = k + 1 \ldots l
\]  

(27)

\[
x_j - x_{j,\text{upper}} \leq 0 \quad j = 1 \ldots n
\]  

(28)

\[
x_j - x_{j,\text{lower}} \leq 0 \quad j = 1 \ldots n
\]  

(29)

Since for approximate optimization we only need the relative size of the objective function, \( F(x) \) may be simplified to

\[
F_r(x) = \left[ x_1 \frac{\partial F(x)}{\partial x_1} + \cdots + x_n \frac{\partial F(x)}{\partial x_n} \right]
\]  

(30)

where \( F_r \) denotes the approximated relative objective function. The other terms in Equation 25 are constant.

Using these approximations, optimization of structures modeled by finite elements is carried out in the following manner:

1. Create an approximate model at the current design.

2. Optimize the approximate model to convergence.

3. Reevaluate the finite element model at the new design.

4. Test for convergence at the finite element level. Convergence is based on the relative and absolute movement of the objective function between the finite element model of step 1 and that of step 3.
5. If the design has not converged at the finite element level, create another approximate model and continue the process.

In this way movement about the design space is driven by the approximate model. This avoids many costly inversions of the stiffness matrix and renders the problem tractable.
III. ASTROS AND MSC/NASTRAN COMPARED

MSC/NASTRAN is a large industry standard multidisciplinary engineering analysis tool. ASTROS is a relative newcomer that has challenged conventional approaches to analysis by pioneering the multidisciplinary optimization of entire aerospace systems. Since both are the progeny of NASA's NASTRAN analysis software package developed in the 1970's, they are notable for their similarities. There are also clear differences in how each program is directed, in how they store and manipulate information, and in their capabilities.

Software Differences

MSC/NASTRAN is described (15:MacNeal) as a large-scale digital computer program which solves a wide variety of engineering problems by the finite element method. Capabilities include linear and nonlinear static analysis with thermal loads, dynamic structural analysis, heat transfer, acoustics, electromagnetism and other types of field problems. The latest version includes a structural optimization capability that can automatically modify the structural design and associated analysis model to satisfy the criteria prescribed by the user (17:Miura). Contrary to Miura (17), this capability was preceded by ASTROS and is therefore not unique.

ASTROS likewise is a finite element based program created to assist in the design of aerospace structures (10:Johnson). Capabilities include linear static structural analysis with thermal loads, normal mode analysis, aerodynamic analysis with steady state aeroelasticity and flutter, and dynamic analysis with transient and frequency response (18:Neill). Capabilities are clearly slanted toward the "aero" disciplines of aerospace and are inferior in quantity to those available in MSC/NASTRAN. The program performs multidisciplinary optimization within the context of its analysis capabilities.

The user interfaces with these programs through an input data file and the subsequent output generated during execution. The input data files for MSC/NASTRAN and ASTROS take a very similar form. Both include portions which control the solution sequence and input the physical
characteristics of the problem at hand. Differences lie in the logical arrangement of the solution sequence portion, the available options for describing the physical problem, and in the description of the optimization problem.

Control of the solution sequence in MSC/NASTRAN is performed in two sections of the input data file; the Executive section and Case Control section (Reymond). The Executive section identifies the job and type of solution and declares the general conditions under which the solution will be obtained, such as maximum time allowed. Any modifications to the normal solution sequence are also declared here. Case Control defines the subcase structure of the problem, such as what loads and constraints are to be considered. This is where requests for output are made. In ASTROS the portions of the input data stream that control the solution sequence are the Executive System Packet and Solution Control Packet (Neill). The input file begins with a preface statement to establish the database, then the Executive System Packet calls a user-supplied solution sequence or modifies the standard one, if necessary. If no Executive is included then the standard sequence is executed. Next, the Solution Control Packet defines the subcase structure. The most notable difference in this area is in the way disciplines are enabled for analysis and optimization. MSC/NASTRAN has many solution sequences available for the different disciplines, and these are called by declaring a solution number in the Executive. ASTROS has only one standard solution sequence that encompasses all the available disciplines. The selection of disciplines is made by simply rearranging the subcase structure in the Solution Control Packet. MSC/NASTRAN’s optimization solution sequence SOL 200 is basically equivalent, but the ASTROS approach is arguably easier to understand and apply.

The physical problem being considered is described within the Bulk Data section of the input data file. The Bulk Data entries available in ASTROS are essentially the same as their counterparts in MSC/NASTRAN, with the exception being the description of the design model for optimization. MSC/NASTRAN allows a wide variety of property characteristics to be identified.
as design variables. ASTROS allows only one type of design variable for each element type; a variable related to system weight, such as thickness or area. MSC/NASTRAN identifies a variety of responses that can be constraints or the cost function. The responses that can be used in this way are structural weight and volume; static displacement, stress and strain; internal force; natural frequency; buckling responses and composite lamina responses (17:Miura). ASTROS has Bulk Data entries that identify constraints related to aileron effectiveness, lift effectiveness, flutter, modal frequency, static displacement, stress, and strain (18:Neill). But the program assumes all optimization is based on system weight. Because of this and the limited design variables, the Bulk Data entries for directing optimization in ASTROS are simpler and incompatible with those in MSC/NASTRAN. The slight increase in complexity is a small price to pay for the improved capabilities.

Other incompatibilities in the bulk data arise from MSC/NASTRAN capabilities that ASTROS does not support. Except for the optimization differences, an ASTROS data deck will generally run in MSC/NASTRAN. The opposite is not true, however. MSC/NASTRAN includes numerous elements, loads, parameter adjusters, output requests, and other capabilities that are not supported in ASTROS. This can limit the analyst.

A significant development in ASTROS that is invisible to the user is the structure of the data base. The developers recognized a need for a data base that could handle three distinct types of data (10:Johnson). First, it had to store and retrieve large, sparse matrices. The second requirement is to access individual data items in entities such as tables, and the third is to access heterogeneous collections of unstructured data efficiently. As no such data base was available commercially, it was developed for ASTROS and designated the Computer Automated Design Data Base or CADDB. The CADDB supports these different data types and provides a common structure for accessing all three in an efficient manner. The CADDB also allows the data base to expand and contract with the problem at hand rather than resorting to fixed size arrays.
A notable similarity between MSC/NASTRAN and ASTROS is their utilization of Vanderplaats MICRO-DOT algorithm to perform optimization (17, 10:Miura, Johnson). Different default parameters are used by each program, however. For instance MSC/NASTRAN imposes a 0.2 relative move limit while that in ASTROS is 2.0. As a result, optimization in the former may grind along at a much slower pace unless the limit is relaxed.

When both programs are used side by side on the same problem, MSC/NASTRAN is notable for it's robustness and flexibility. Difficulties that are automatically cared for or worked around in MSC/NASTRAN may crop up as fatal errors in ASTROS. For instance, modal analysis of flat plates in ASTROS requires omission of rotational degrees of freedom; something that is not necessary in MSC/NASTRAN. Also, ASTROS does not support some useful outputs such as nodal stress values, preventing the analyst from obtaining edge stresses.

Some bugs still exist in the ASTROS code. For example, if the number of constraints is about ten times greater than the number of design variables, optimization may halt with a fatal array dimensioning error. When static thermal analysis is attempted on the Sun 4 using either Versions 4 or 5, analysis terminates with a fatal error and the system accuses the user of defining more than one temperature per grid point. The same problems run correctly on the VAX. The Flight Dynamics Lab has confirmed that ASTROS was poorly ported to the Sun, and is working on corrections. Another interesting discrepancy has to do with Version 5's implementation of the four noded quadrilateral element QUAD4. Models using this element that run successfully in Version 4 often terminate with fatal matrix singularities in Version 5. The QUAD4 element was modified for Version 5 to enhance accuracy with composite materials (24). It appears that greater composite material accuracy has driven the element to singularities in other applications.
Comparison With Exact Solutions

Round Plate: Timoshenko (23) shows that the center deflection $w$ of a uniform round plate with clamped edges and loaded by a uniform pressure is given by

$$w = \frac{qa^4}{64D} \quad (31)$$

where $q$ is the uniform pressure, $a$ the plate radius, and $D$ its flexural rigidity. Meirovitch (16) shows that the first natural frequency $\omega$ of such a plate in Hertz is given by:

$$\omega = 1.015^2 \frac{\pi}{2a^2} \sqrt{\frac{D}{\rho}} \quad (32)$$

where $\rho$ is the area density of the plate. The flexural rigidity $D$ is given by

$$D = \frac{Eh^3}{12(1-\nu^2)} \quad (33)$$

where $E$ is Young’s Modulus, $h$ is the plate’s uniform thickness and $\nu$ is Poisson’s Ratio for the material.

A numerical experiment was performed letting $a$ equal 15 inches, $h$ equal 0.25 inches, $E$ equal 10700 ksi, $\nu$ equal 0.33, $P$ equal 10 psi and $\rho$ equal $6.475 \times 10^{-5}$ slug inches. Material properties are nominal values for 2024-T3 bare aluminum alloy (5:Denno). The theoretical center point deflection given by Equation 31 is 0.50593 inches and the first natural frequency given by Equation 32 is 111.76 Hertz. The same plate was next evaluated using finite elements. Figure 1 illustrates a finite element model of the plate and Figure 2 plots center point deflection in inches versus the number of radial elements in the model. Both MSC/NASTRAN and ASTROS converged to within 3.7 percent of the theoretical result with ASTROS slightly more accurate. Figure 3 plots the first natural frequency of the plate in Hertz versus the radial grid refinement. Both
Figure 1. Model of Clamped Round Plate With Ten Elements in Radial Direction

Figure 2. Plot of Center Displacement Versus Grid Refinement, Round Plate
programs converged to within 0.27 percent of the theoretical result with ASTROS again slightly more accurate.

**Plane-strain Cylinder:** Saada (21) shows that the mid-plane radial deflection $\delta$ of a plane-strain tube loaded by internal pressure is given by

$$\delta = \frac{Pa^2(1+\nu)}{2(b-a)E} \left[ \frac{4b^2}{(a+b)^2} - 2\nu + 1 \right]$$

where $P$ is the internal pressure, $b$ the outer radius and "a" the inner radius of the tube. He also shows that the mid-plane hoop stress $\sigma_h$ is given by

$$\sigma_h = \frac{a^2 P}{b^2 - a^2} \left[ 1 + \frac{4b^2}{(a+b)^2} \right]$$

A numerical experiment was performed with "a" equal 14.875 inches and $b$ equal 15.125 inches. This gave a plane-strain tube with a mid plane radius of 15 inches and thickness of 0.25 inches. Using the $E$ and $\nu$ for aluminum and an internal pressure equal to 10 psi gives with
Equation 34 a theoretical mid-plane radial deflection of 0.00074629 inches. The hoop stress is 594.98 psi per Equation 35. The tube was then analyzed with finite elements. Element boundaries were skewed with respect to the ends of the tube to convert any inter-element bending tendencies to membrane stresses. Figure 4 illustrates a finite element model of the tube and Figure 5 plots mid-plane displacement in inches versus the number of circumferential elements. MSC/NASTRAN converged to within 0.2 percent of the theoretical result while ASTROS only converged to within 3.7 percent. Figure 6 is a plot of the hoop stress $\sigma_h$ of the tube in psi versus the number of circumferential elements. Both programs converge to within about 0.6 percent of the theoretical result, but MSC/NASTRAN is about 0.05 percent better. ASTROS is slightly better on models having a coarse finite element grid.

Clamped Dome: In the case of a uniform partial-dome loaded by uniform pressure $P$, Timoshenko (23) illustrates the general hypergeometric series solution and Kraus (11) gives an example of the infinite series form of the solution for the special case of a semi-dome. The solution is an infinite series of Gamma functions which becomes ill-conditioned and difficult to evaluate as the thickness of the shell becomes small compared with its radius (23:Timoshenko). A very good ap-
Figure 5. Plot of Mid-plane Radial Displacement Versus Grid Refinement, Plane-strain Tube

Figure 6. Plot of Hoop Stress Versus Grid Refinement, Plane-strain Tube
proximation can be made by assuming a shape for the shearing force that damps out toward the center of the dome. With a radius to thickness ratio of 30, Timoshenko demonstrates that the approximation gives stress resultants that are good to within 10 percent at the extremes and follow the theoretical curves in every respect. The approximation improves as the radius to thickness ratio increases. With this approximation Gibson (7) has solved for the stress resultant forces $N_\phi$ and $N_\theta$ and the moments $M_\phi$ and $M_\theta$, as follows:

\[
N_\phi = \frac{-Pa}{2}
\]

\[
N_\theta = \frac{Pa}{2}(1 - \nu)e^{\beta(\phi - \alpha_1)}[\cos(\beta(\alpha_1 - \phi)) + \sin(\beta(\alpha_1 - \phi))] - \frac{Pa}{2}
\]

\[
M_\phi = \frac{P\alpha^2(1 - \nu)}{4\beta^2}e^{\beta(\phi - \alpha_1)}[\sin(\beta(\alpha_1 - \phi)) - \cos(\beta(\alpha_1 - \phi))]
\]

\[
M_\theta = \frac{\nu P\alpha^2(1 - \nu)}{4\beta^2}e^{\beta(\phi - \alpha_1)}[\sin(\beta(\alpha_1 - \phi)) - \cos(\beta(\alpha_1 - \phi))]
\]

In these equations "a" is the dome radius and $\alpha_1$ the half angle subtended by the dome. The parameter $\beta$ is defined as:

\[
\beta = \sqrt{3(1 - \nu^2)}\frac{a}{h}
\]

The $\theta$ represents the circumferential direction and $\phi$ the angle from the dome center point, as seen in Figure 7 from Gibson (7). The stresses within the shell can be derived from the stress resultants by the following Equation (4;Cook):

\[
\sigma_\theta = \frac{N_\theta}{h} + \frac{12zM_\theta}{h^3}
\]

where the $\ast$ is replaced by $\theta$ or $\phi$ as desired, and $z$ is the distance from the mid-plane.

A numerical experiment was performed with "a" equal 15 inches, $h$ equal 0.25 inches, $\alpha_1$ equal 35 degrees, $E$ and $\nu$ for aluminum and pressure $P$ equal 10 psi. The radius to thickness ratio was 60, which is double that used by Timoshenko to demonstrate the quality of the approximate solution. Figure 8 illustrates a finite element model of the dome. The theoretical values obtained
Figure 7. Uniform Spherical Dome With Rigidly Fixed Edges Subjected To Uniform Pressure

Figure 8. Model Of A Clamped Dome With 5 Elements In The Radial Direction
using Equations 36 through were compared with those found by finite element analysis. Figure 9 is a plot of mid-plane \( \sigma_\phi \) in psi versus the angular distance from the center point. Likewise Figure 10 plots \( \sigma_\phi \). The top surface \( \sigma_\phi \) is plotted in Figure 11 and \( \sigma_\phi \) at the top surface is shown in Figure 12. The stresses in each case are in psi. MSC/NASTRAN and ASTROS yield virtually identical results with the former slightly better where differences exist. The finite element solutions lie within about 9 percent of the theoretical results and their extreme value always exceeds theoretical. This is a positive attribute for an analysis tool as the design is driven by stresses not likely to be exceeded in the physical world. It is also interesting to note that in Figure 9 the finite element results err in the direction of the exact hypergeometric series solution illustrated by Timoshenko (23).

**Optimization:** A numerical optimization experiment was performed on the 64 element plane-strain tube. With the radial displacement from static analysis as an upper constraint, the model was optimized for minimum weight to see how close MSC/NASTRAN and ASTROS came to the theoretical 0.25 inch wall thickness. Optimization started with a thickness of 10 inches. With default optimizer parameters ASTROS converged within five iterations, but MSC/NASTRAN took fifteen. This is because the latter imposes a default 20 percent move limit while the former has a
Figure 10. Plot of Mid-plane $\sigma_\phi$ Versus Angle $\phi$, Clamped Dome

Figure 11. Plot of Top Surface $\sigma_\theta$ Versus Angle $\phi$, Clamped Dome
200 percent limit. With the MSC/NASTRAN move limit relaxed to 90 percent, convergence was obtained in five iterations. Both programs converge to within 0.02 percent of the theoretical value. The design variable history is plotted in Figure 13 where wall thickness is given in inches.
Figure 13. Iteration History Of Wall Thickness, Plane-strain Tube
IV. OPTIMIZATION OF A DOUBLY CURVED SHELL

The analysis of simple plate and shell constructions has given confidence in the capabilities of MSC/NASTRAN and ASTROS, and provided a foundation for the analysis of more complex structures. In this chapter optimization is probed more deeply in the context of a large nozzle.

Finite Element Model Of A Nozzle

Physical Characteristics: A finite element model was created using the deployed shape of the Peacekeeper Stage II nozzle. This large nozzle was developed by the Missile, Ordnance and Space Group at Hercules Corporation. It has a length of 78.38 inches, throat diameter of 8.60 inches and exit diameter of 69.46 inches (8:Hercules). The nozzle was modeled using four noded quadrilateral elements having bending, membrane and shear stiffness. A computer program was developed that fits the finite element mesh to the nozzle geometry with IMSL cubic spline routines (9), then generates the executive, case control and bulk data files needed for analysis in MSC/NASTRAN and ASTROS.

Analysis time was saved by exploiting the symmetry of the geometry and loads. A quartered model and single element strip model were used with good results. A full nozzle with 32 elements per circumference has 960 elements, but only 30 of these are independent. A quartered model reduces the number of elements to 240, while a strip model uses the 30 independent elements. The model generation program has the flexibility to create either full, quartered, or strip models at the user's discretion. It also lets the user vary the fineness of the mesh. Figure 14 is an illustration of a quartered nozzle and Figure 15 shows the same nozzle modeled by a single strip of elements.

Material properties were those for alloy AISI 687 (13:Lynch); a high temperature, high strength alloy of Nickel, Chromium, Cobalt and other trace elements. This alloy was chosen for its yield strength at the nearly 1700 degrees fahrenheit at the throat. AISI 687 has a modulus of
Figure 14. Finite Element Model Of Quartered Nozzle, Grid Fineness Of 32 Elements Per Circumference

Figure 15. Finite Element Model Of Nozzle Using Single Element Width Strip, Grid Fineness Of 32 Elements Per Circumference
elasticity of 32400 ksi, coefficient of thermal expansion of 7.5 \times 10^{-6} \text{ per degree fahrenheit}, and an interpolated yield strength of 68 ksi at 1700 degrees.

*Pressure And Thermal Loads:* Pressure loads were approximated using the one dimensional Laval nozzle equations for isentropic flow. The development of the relation between pressure and channel area is illustrated by Kuethe and Chow (12), resulting in the equation

\[
\left( \frac{A}{A^*} \right)^2 = \frac{\frac{2}{\gamma+1} \left[ \frac{P}{P^*} \right]^{\frac{\gamma+1}{\gamma-1}}}{1 - \left[ \frac{P}{P^*} \right]^{\frac{2}{\gamma-1}}} \]

(42)

where the "A" is the nozzle area at the point of interest and \( A^* \) the area at the throat. Pressure at the point of interest is denoted by \( P \), and \( P^* \) is the chamber pressure. The specific heat ratio for the gas is \( \gamma \).

With chamber pressure and nozzle geometry known, Equation 42 may be solved numerically for pressure at any axial location. The relation is double valued except at the throat. The correct value is determined by whether the point of interest is up or downstream of the throat, and the pressure outside the nozzle. For this analysis the nozzle was assumed to be operating in a vacuum, thus the high pressure value belongs on the chamber side of the throat and the low pressure value toward the nozzle exit. The model generation program numerically solves Equation 42 for \( P \) using the nozzle crosssectional area at the element in question, the throat area, a \( \gamma \) of 1.18 and a chamber pressure of 1000 psi. This was applied to the element as a pressure load. Element loads versus location are plotted in Figure 16.

The parameters used in the development of pressure loads were chosen to be as realistic as possible. A 1000 psi chamber pressure is reasonable for a nozzle of this size (8:Hercules). The \( \gamma \) is for a nitric acid oxidizer and kerosene fuel mixture used by Barrere (3) in an example calculation of nozzle temperatures. In a real nozzle the \( \gamma \) would not be constant, nor the flow isentropic, as assumed here. However, Equation 42 is good first approximation to the pressure loads and generally
within 15 percent of experimental values (6: Galati), thus it provided a good representative load for nozzle analysis.

Temperature loads were modeled using the example developed by Barrere. His values were converted to degrees fahrenheit and fit to this nozzle's geometry. The example was of a nitric acid and kerosene fueled rocket motor using the nitric acid to regeneratively cool the nozzle. The resulting nozzle wall temperatures are plotted in Figure 17. Although the example is a little outdated, it nonetheless provided a good representative temperature profile for analysis and material selection.

Optimization Scheme: Optimization was performed with the objective being to minimize weight by varying the wall thickness. The Von Mises stress criteria was used with a yield strength of 68 ksi as the constraint. This criteria is defined as (20: Reymond):

\[ \tau_V = \sqrt{\sigma_x^2 - \sigma_x \sigma_y + \sigma_y^2 - 3\tau_y} \]  

The minimum wall thickness was constrained to 0.01 inches and the exit lip constrained to a maximum deflection of 2 inches in any direction. Wall thickness was assumed uniform in the
circumferential direction, consistent with the symmetric loading and geometry. ASTROS Versions Four and Five do not support nodal temperature loads on the SUN 4, therefore it was run without thermal loads. MSC/NASTRAN was run with and without thermal loads for comparison purposes.

The Modified Feasible Directions search scheme was used by MSC/NASTRAN and ASTROS. Both programs employ the MICRO-DOT algorithm for optimization but initiate it with different default optimizer parameters. Of the 45 parameters in the algorithm (25: Vanderplaats), MSC/NASTRAN provides for the easy modification of 13 and ASTROS documents 26. Both programs allow the experienced user to vary all the parameters. Numerous optimization runs were performed to compare the goodness of the default parameters and identify those which have the best effect on the optimization process.

Results

The strip model ran well in MSC/NASTRAN with adjustment of the optimizer parameters, but did not converge with defaults. Figure 18 compares the optimization progress with different parameters. The dramatic improvement in the first step was obtained by changing the move limit
parameter \textit{DELP} from its default of 0.2 to 2.0. This resulted in a slight overshoot that settled in four iterations and converged within ten. Adjusting the convergence detection parameters \textit{CONV1} and \textit{CONV2} provided a further improvement in the objective of about 2 percent. These parameters determine the relative and absolute change in the objective for convergence. The default values are 0.001 and 0.01 respectively. They were both changed to $0.1 \times 10^{-12}$.

The strip model and the quartered model were used in ASTROS. Figure 19 compares the optimization progress between the two models. Although the ASTROS optimizer default values were an improvement over those in MSC/NASTRAN, the design never reached the optimum. Varying the optimizer parameters had little effect. Convergence detection parameters were adjusted to $0.1 \times 10^{-12}$ and $0.1 \times 10^{-18}$, and the design variables permitted to step to one thousandth of their initial value on the first step. But the design still changed at the same rate and ground to a halt short of the optimum. The quartered model was then run and it converged at the same rate but settled to the MSC/NASTRAN optimum in eleven iterations. Apparently ASTROS has difficulty optimizing extremely light structures without modification of some yet undiscovered parameter.
Moving to an equivalent but more massive model overcame this difficulty.

The best MSC/NASTRAN and ASTROS optimization runs are compared in Figure 20. Both programs converged to final values that agree to within 1.3 percent. However, MSC/NASTRAN arrived in the neighborhood of the optimum much more quickly than ASTROS. The resulting thickness distribution is plotted in Figure 21 with thickness in inches. The programs are in good general agreement and reach the 0.01 inch minimum thickness constraint at about 20 inches axially. The distributions at the throat reflect the rapidly changing loads and geometry. MSC/NASTRAN has more sharply defined thickness variations in the throat area resulting from the fourteen iterations allowed it near the optimum. In designing a real nozzle, the peak values from an analysis such as this would be used to establish a smoothly varying thickness distribution from chamber to exit, with some margin of safety added. Despite the discrepancies at the throat, both programs would provide good data for such a distribution. If ASTROS were used, however, there would be some risk of exceeding margins, since the peak values in ASTROS are consistently below those in MSC/NASTRAN.
Figure 20. Comparison Of Optimization Progress Using MSC/NASTRAN and ASTROS, Nozzle Model With Grid Fineness Of 32 Elements Per Circumference

Figure 21. Comparison Of Final Thickness Distribution Using MSC/NASTRAN And ASTROS, Nozzle Model With Grid Fineness Of 32 Elements Per Circumference
The resulting major and minor principle stresses are plotted in Figures 22 and 23. Once again both programs are in very good agreement with the exception of an interesting deviation at the throat. Major principle stresses are in the circumferential direction and minor stresses are in the axial direction. The circumferential stress is generally in tension since the radius of curvature lies on the pressure side of the structure. However, at the throat the axial radius of curvature lies outside the nozzle. This causes a large compressive spike in the axial distribution which the geometry couples into the circumferential direction as a trend toward compression. This is seen in both the MSC/NASTRAN and ASTROS stress distributions. The ASTROS model is about 0.01 inches thicker in the vicinity of the throat, and this support keeps the element from going into outright compression. The MSC/NASTRAN model is thinner, so the throat does go into compression. Despite these variations, the Von Mises stress criteria are satisfied in both cases, and the stresses have been shown to be reliable. The discrepancy at the throat is a result of slightly different thickness distributions, and these in turn are a product of the differences in the optimizer parameters. The nozzle was also optimized using a finite element model with twice the grid refinement. There was little change in the thickness and stress distributions.
The nozzle was optimized in MSC/NASTRAN with static thermal loads. This was done to quantify the thermal stresses being overlooked because of ASTROS' inability to support thermal analysis on the Sun 4. Stresses were low and the optimization quickly moved the thickness distribution to the 0.01 minimum throughout. The resulting major principle stress distribution is plotted at Figure 23. With the exception of a 3200 psi tensile load at the throat, the loads are minimal and of a few hundred psi. The throat load is the result of the rapid change in nozzle diameter and, interestingly, opposes the stress resulting from the pressure load. When the model was optimized in MSC/NASTRAN with both pressure and thermal loads, the final objective was about 0.2 percent smaller than that obtained with pressure loads alone. This reflects the balancing of opposing stresses at the throat, and could lead to a thickness distribution that is inadequate at engine ignition when the nozzle is cool. In this application the inclusion of thermal loads does not significantly improve the analysis. The real utility in knowing the thermal distribution is in the material selection process, where use of high temperature yield values is critical. Thermal analysis should be done, and it is unfortunate that ASTROS was unsupportive, but good results can be obtained nonetheless.
Figure 24. Principle Stress Distribution Resulting From Thermal Loads Using MSC/NASTRAN, Nozzle Model With Grid Fineness Of 32 Elements Per Circumference
V. AXISYMMETRIC FINITE ELEMENT ANALYSIS IN ASTROS

Axisymmetric shell finite elements offer an alternative to the analysis of axially symmetric shells with flat elements. We have seen the results that may be obtained with flat, quadrilateral elements. This chapter will discuss the development of a Mindlin type axisymmetric shell element and illustrate its use in ASTROS.

Element Development

The following development carries through the steps alluded to but not explicitly demonstrated by Cook (4) in his development of a Mindlin axisymmetric shell element. Element geometry and coordinates are illustrated in Figure 25. As in the figure, $t$ denotes element thickness and $z$ is the distance from the midplane. The length of the element is $L$ and the degrees of freedom supported at each node ring are $u, w$ and $\beta$. The $u$ and $w$ may be rotated to the global coordinates $U$ and $W$. The angle the element makes with global coordinate $U$ is denoted $\phi$. The circumferential direction is not shown but is denoted by $\theta$.

The element nodal displacement vector $d_e$ is configured as:

$$d_e = \begin{bmatrix} w_1 & u_1 & \beta_1 & w_2 & u_2 & \beta_2 \end{bmatrix}^T$$  \hspace{1cm} (44)

In the axisymmetric case these displacements refer to nodal rings rather than the more familiar nodal points. Because of this, elements containing nodal points cannot be attached to axisymmetric elements.

Using the generalized coordinate $\xi$ where $\xi = \frac{2z}{L}$, the shape functions are:

$$N_1 = \frac{1}{2}(1 - \xi)$$  \hspace{1cm} (45)

$$N_2 = \frac{1}{2}(1 + \xi)$$  \hspace{1cm} (46)
These are assembled into an element shape function matrix \([N]\) of the form:

\[
[N] = \begin{bmatrix}
0 & N_1 & 0 & 0 & N_2 & 0 \\
N_1 & 0 & N_3 & N_2 & 0 & -N_3 \\
0 & 0 & N_1 & 0 & 0 & N_2 
\end{bmatrix}
\] (48)

If we define an operator matrix \([D]\) as

\[
[D] = \begin{bmatrix}
\frac{\partial}{\partial \xi} & 0 & 0 \\
\frac{\cos \phi}{r} & \frac{\sin \phi}{r} & 0 \\
0 & 0 & \frac{-\delta}{\sin \phi} \\
0 & 0 & \frac{\delta}{\sin \phi} \\
0 & \frac{\delta}{\sin \phi} & -1 
\end{bmatrix}
\] (49)

then the element strains are obtained by the operation:

\[
\epsilon = [D][N]d = [B]d
\] (50)

This gives \(\epsilon\) in the form \(\epsilon = \left[ \epsilon_1, \epsilon_\theta, \kappa_1, \kappa_\theta, \gamma_{zz} \right]^T\) where the \(\epsilon\) and \(\gamma\) are strains and the \(\kappa\) are curvature changes.

Figure 25. Conical Shell Element Cross Section

\[N_3 = \frac{L}{8}(1 - \xi^2)\] (47)
The material property matrix is denoted by \([E]\) and defined as:

\[
[E] = \begin{bmatrix}
\frac{E_t}{1-\nu^2} & \frac{\nu E_t}{1-\nu^2} & 0 & 0 & 0 \\
\frac{\nu E_t}{1-\nu^2} & \frac{E_t}{1-\nu^2} & 0 & 0 & 0 \\
0 & 0 & \frac{E_t^3}{12(1-\nu^2)} & \frac{\nu E_t^3}{12(1-\nu^2)} & 0 \\
0 & 0 & \frac{\nu E_t^3}{12(1-\nu^2)} & \frac{E_t^3}{12(1-\nu^2)} & 0 \\
0 & 0 & 0 & 0 & \frac{5E_t}{12(1+\nu)}
\end{bmatrix} \tag{51}
\]

Premultiplying \(\epsilon\) by \([E]\) returns membrane forces \(N_\{\cdot\}\), and bending moments \(M_\{\cdot\}\) in the shell. The \((\cdot)\) is replaced by \(s\) or \(\theta\) depending on which stress resultant is desired. Using the relation

\[
\sigma_\{\cdot\} = \frac{N_\{\cdot\}}{t} + \frac{12zM_\{\cdot\}}{t^3} \tag{52}
\]

stresses are obtained through matrix \([S_\cdot]\) of the form:

\[
[S_\cdot] = \begin{bmatrix}
\frac{1}{t} & 0 & \frac{6}{t^2} & 0 & 0 \\
0 & \frac{1}{t} & 0 & \frac{6}{t^2} & 0 \\
\frac{1}{t} & 0 & 0 & 0 & 0 \\
0 & \frac{1}{t} & 0 & 0 & 0 \\
\frac{1}{t} & 0 & \frac{6}{t^2} & 0 & 0 \\
0 & \frac{1}{t} & 0 & \frac{6}{t^2} & 0 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix} \tag{53}
\]

If postmultiplied by the element strain vector, this matrix will return stresses in the element coordinate system. These stresses are at the outer surface, midplane and inner surface of the shell. The form of the stress vector returned is:

\[
\sigma = \begin{bmatrix}
\sigma_{\text{outer}} \\
\sigma_{\text{outer}} \\
\sigma_{\text{midplane}} \\
\sigma_{\text{midplane}} \\
\sigma_{\text{inner}} \\
\sigma_{\text{inner}} \\
T_{\text{g}}
\end{bmatrix} \tag{54}
\]

With these matrices the element stiffness matrix \([k]\) is formed by the following integration:

\[
[k] = \frac{L}{2} \int_{-1}^{1} [B]^T[E][B]d\xi \tag{55}
\]
The matrix is then rotated to the global coordinate system for assembly into a global stiffness matrix.

Loads in general do not fall at node rings. A vector of equivalent nodal loads $f_{eni}$ is formed for pressure load $P(\xi)$ on the element by the integration:

$$f_{eni} = \frac{L}{2} \int_{-1}^{1} [N]^T P(\xi) d\xi$$  \hspace{1cm} (56)

After the finite element solution has been obtained, element stresses are extracted from the global displacement vector $d_g$ as follows:

$$\sigma \cdot [S_g] d_g = [S_e] [E] [B] [R] d_g$$  \hspace{1cm} (57)

The $[R]$ is a transformation matrix that rotates the global displacements $d_g$ to displacements in the element coordinate system. $[S_g]$ is the stress recovery matrix which converts global displacements to element level stresses.

**ASTROS Implementation**

The Mindlin Axisymmetric element was implemented in ASTROS by externally programming the necessary matrices, creating a pseudo axisymmetric problem in ASTROS with rod elements, then overwriting the internal matrices with those created externally. The program discussed in Appendix B was developed to create the global stiffness matrix, load vector, and stress recovery matrix. The integrations required by Equations 55 and 56 are performed using a seventh order Gaussian Quadrature scheme with weights and function evaluation points provided by Stroud and Secrest (22). The element level matrices thus obtained are rotated to the global coordinate system and assembled into global level matrices. Likewise the element level stress recovery matrix is obtained by performing the multiplications of Equation 57. These are assembled into a global level
stress recovery matrix. The matrices are then put in a direct matrix input form for bulk data entry into ASTROS.

To flush out the unwanted matrices and load in the externally created ones, ASTROS requires modification of the standard MAPOL solution sequence at the executive level. For ASTROS version 5 this was done through the following edit sequence:

```
INSERT 88
MATRIX [MYKGG],[MYPL],[MYSREC],[STROUT];
INSERT 1613
[KGG] := [MYKGG];
INSERT 1638
[PG] := [MYPL];
REPLACE 2303,2311
[STROUT] := [MYSREC] * [UG];
CALL UTMPRT(0,[STROUT],[UG]);
```

The insertion at line 88 creates matrix entities MYKGG, MYPL, and MYSREC within ASTROS. These are the global stiffness, load, and stress recovery matrices to be loaded via direct matrix input. The STROUT matrix outputs stresses. At the appropriate point the ASTROS global stiffness matrix $KGG$ and load vector $PG$ are replaced by the externally generated substitutes. Solution occurs normally, albeit with radically modified matrices. The normal output sequence from lines 2303 through 2311 is no longer useful, and is replaced with a matrix multiplication to produce stresses and an output command to directly print stresses and displacements.

All that remained was to create a pseudo axisymmetric bulk data problem with CONROD elements. This arranged the solution sequence, constraints, matrix sizes and the internal structure of ASTROS for solution of the externally generated matrices. CONROD is a rod finite element that has an element stiffness matrix and degrees of freedom of similar size and compatible with the axisymmetric finite element. Because of this it is useful for creating the pseudo problem. The grid points must match the location of the nodal rings used to create the stiffness, load, and stress recovery matrices, and they must lie in the X-Z plane. The Z coordinate in ASTROS matches
the global $U$ coordinate along the axial direction in the axisymmetric problem. Likewise the X coordinate matches the global $W$. Constraints and CONROD elements are applied to these grid points as they would be in the axisymmetric problem. Loads and material property entries are necessary for proper internal sizing, but can contain dummy values since externally generated data will take over before the solution begins.

**Results**

A numerical experiment was performed using the geometry and properties of the plane strain tube used in Chapter 3. The direct matrix input data created for the axisymmetric problem were merged with a CONROD pseudo model of the tube and the appropriate executive modifications. The theoretical mid-plane radial deflection is 0.00074629 inches and theoretical hoop stress is 594.98 psi. The ASTROS axisymmetric solution returned a radial deflection of 0.00074952 inches and hoop stress of 600.00 psi. This is an error of 0.43 percent for displacement and 0.84 percent for stress, comparing favorably with the 0.6 percent obtained using 32 quadrilateral flat elements.

This problem was formulated with a 12 by 12 matrix containing the 6 by 6 axisymmetric stiffness matrix. In contrast, the quadrilateral element problem required solution of a 1932 by 1932 stiffness matrix for the full tube or a 24 by 24 matrix if symmetry was fully exploited. The axisymmetric element offered comparable accuracy while reducing the size of the problem, but in this formulation disallowed consideration of non-axisymmetric loads.

The optimized 32 element per circumference nozzle model of Chapter 4 was also reformulated as an axisymmetric problem. This produced an extremely large bulk data deck which ASTROS was not successful in solving. In describing the bulk data entry for direct matrix input, the ASTROS User's Manual (18:Neill) implies that only nonzero entries need be loaded. In fact, if the entire matrix is not loaded including all the zeros, a number of strange errors crop up in the solution sequence. In both versions 4 and 5 the program typically accuses the user of trying to load beyond
the dimensioned size of the matrix, or of improperly using complex arguments. These problems disappeared when the entire matrix was loaded, but this drove the size of the direct matrix entry to the order of 10,000 lines. This resulted in input/output problems on the Sun 4.
VI. CONCLUSIONS

The Utility Of MSC/NASTRAN And ASTROS

MSC/NASTRAN and ASTROS provide accurate analysis and optimization results on plate and shell structures, providing the finite element model is carefully constructed with a fine grid structure. Results on simple shells have been demonstrated to within 10 percent of theoretical values, and responses obtained for a large nozzle make good physical sense. From an analysis standpoint the differences between the programs are negligible. Optimization results are similar, but each program steps down differing iteration paths driven by different optimizer parameters.

MSC/NASTRAN is more robust with a large element set and many other options available to the analyst. It has well developed documentation suitable for both new and experienced users. Optimizer parameters are clearly documented and can greatly speed the optimization process. The program is capable of analyzing thermal loads on the Sun 4. Software bugs were not encountered.

ASTROS is hindered by a few software errors. If the ratio of constraints to design variables exceeds about ten, the optimizer is driven into an array dimensioning error. The program as ported to the Sun Workstation has difficulties with thermal analysis. The documentation available for ASTROS is also a limiting factor. As a whole the documentation is accurate but not particularly user friendly. The standard complement of manuals is available, but these do not accurately reflect the newer versions of ASTROS. System Release dates must be used in conjunction with the manuals to obtain correct information in some cases. The small element set and limited capabilities can also hinder the analyst.

On the positive side, with the capabilities it has ASTROS offers cost free equivalence to MSC/NASTRAN. Software bugs are being actively pursued and corrected, and an optimality criteria optimization scheme is being implemented. The inclusion of this optimality criteria search technique in the next release should significantly reduce the consumption of computer time on large scale problems.
The Optimization Process

Both MSC/NASTRAN and ASTROS use the Method of Feasible Directions for optimization. This is a robust technique that moves about the design space based upon gradient information. Movement is meticulous and deliberate. A sub-optimization must be performed in order to determine the direction to move. A one-dimensional line search must be performed to determine the magnitude of the movement. This requires sampling the design space and constraints at several points and interpolating based on the results. The method is well conditioned but its utility is limited to about 300 design variables, as are all other popular methods.

The Optimality Criteria Method also moves in the design space based on gradient information, but gradients are used differently. The design leaps about the design space driven by simple ratios of the design information, modified by "twiddle" factors. No sub-optimizations or one dimensional searches are required. The method is sensitive to "twiddle" factors and may be unstable if they are not just right. On the other hand, moving about the design space is less computationally burdensome and less sensitive to the number of design variables in the problem. This method should move optimization beyond present design variable limits.

Nozzle Analysis And Optimization

The doubly curved nozzle shape can be adequately analyzed in both MSC/NASTRAN and ASTROS. Key areas of consideration are thermal effects and stress distributions. The analyst may also be concerned with displacements, but these are likely to be of little consequence. The displacements seen in the analysis done here were of the order of a tenth of an inch at the most.

Nozzle temperatures are extremely high and constrained by material limitations. Because of the thermodynamics of nozzle flow, the highest temperatures are near the throat. Material properties may be strongly temperature dependent, particularly yield strength. For instance, the room temperature yield strength of alloy AISI 687 (13.Lynch) is double that at 1700 degrees
fahrenheit. If the room temperature yield strength were used in optimization at the 1700 degree
throat, catastrophic failure would result. Therefore high temperature material properties must be
used for nozzle analysis.

Static thermal loads also induce stresses, but these have been shown to be lower order effects
for the nozzle considered here. Thermal loads using composite materials and transient or erosive
effects could be significant, but are beyond the scope of this discussion. Thermal stresses arise in
shells from nodal constraints, changes of curvature, and thermal gradients. All three of these effects
are strong in the vicinity of the throat and can affect optimum thickness distributions. Thermal
stresses at the throat oppose those which are pressure induced, leading to a thinner optimum wall.
The thickness may be adequate at operating temperatures but insufficient for loads at ignition
when the nozzle is cold. Nozzles should be analyzed with and without thermal effects to guard
against this.

Pressure induced stresses are more critical than those induced thermally. Four distinct stress
distribution regions can be identified and linked to the geometry and loading of the nozzle. Region
I is where the nozzle terminates at the chamber. Region II is the throat area. Region III lies in
the high pressure region aft of the throat. Region IV is the lower pressure region aft of the throat
where stress constraints are no longer active.

Because of the nature of supersonic nozzle flow, pressures rapidly drop aft of the throat. Due
to this drop in pressure, the nozzle wall thickness toward the exit is no longer constrained by stress
but is driven by other considerations, such as manufacturability, buckling, and dynamic loads. For
the nozzle evaluated here, the optimization process quickly moved wall thickness aft of 20 inches
to the hypothetical manufacturability constraint of 0.01 inches. This area is denoted as region IV.
The dominant radius of curvature is in the circumferential plane on the pressure side, making the
dominant stress a tensile hoop stress in the circumferential direction. The wall is tilted with respect
to the axial direction, so there is a cumulative compressive axial direction stress. But pressures are
low and the angle of the wall with the axial direction is small, making this a secondary effect.

Stress distributions near the throat are much more interesting and unpredictable. As the throat is approached from the aft end, pressures rise and eventually deactivate the side constraints, signaling entry into region III. Wall thickness is driven by a stress constraint such as the Von Mises stress criteria. Hoop stress varies proportional, and axial stress inversely proportional to radius. Therefore in region III a tradeoff occurs between axial and hoop stress to keep the Von Mises stress maximized. This is seen in Figures 23 and 24 where the circumferential or major principle stress decreases toward the throat, while the axial or minor principle stress increases in magnitude.

Continuing up the nozzle, the shell curvature inflects to form the throat, signaling the onset of region II. In this area the radius of curvature in the axial plane approaches and may exceed that in the circumferential plane. The change in curvature superimposes a compressive hoop stress on the large compressive axial stress. The total axial stress dominates and, because of the geometry, imparts a compressive circumferential stress. This, combined with the circumferential hoop stress reduction due to decreasing diameter, leads to the trend reversal seen in circumferential stress at the throat. The reversal allows a thinning of the nozzle wall in this region.

At region I there is a dramatic axial stress reduction due to the high pressure load component opposing the cumulative effects from the rest of the nozzle. Radial stresses again dominate, and there is a considerable thickening of the nozzle wall to support the high pressures.

**Axisymmetric Elements**

Axially symmetric problems such as the nozzle can be modeled using axisymmetric finite elements. The Mindlin axisymmetric element is nearly as accurate as a four noded quadrilateral element model with a grid fineness of 160 elements per circumference. The axisymmetric model tested here uses a 6 by 6 stiffness matrix to solve the problem posed by a 1932 by 1932 quadrilateral element stiffness matrix. This is a significant simplification. The matrix flushing scheme provided
a quick test of the Mindlin element's accuracy and demonstrated ASTROS' flexibility, ease of modification, and potential for the implementation of such an element.

The axisymmetric element considered here is not attachable to elements with nodal points and does not support non-axisymmetric loads. The loading difficulty can be overcome by Fourier analysis (14:MacNeal). The inability to attach to other element types is a disadvantage common to axisymmetric elements, precluding their use in large complex models.

Lessons Learned In The Optimization Of Shells

The following general guidelines have been gleaned from this investigation and will prove useful to the analyst seeking the "correct" answer to a shell problem:

1. The analyst should not be deceived by the remarkably accurate results finite element analysis gives for flat plates and singly curved shells under simple loading. A complex, doubly curved shell is not likely to give results with an error less than the ± 10 percent seen for the clamped dome.

2. Thermal loads will induce stresses in response to constraints, curvature changes, and thermal gradients. Material properties used in analysis must also take temperatures into account.

3. Optimization should be driven to convergence. Unless the plot of the objective or cost function asymptotically approaches some value, the optimum has not been reached. This is seen in Figure 19 where ASTROS terminated optimization before reaching the minimum wall thickness, but claimed convergence.

4. Optimizer parameters should be varied. Accepting default values may give very slow convergence.
5. Grid fineness should be varied. Accurate results are obtained when the results are no longer affected by increasing grid refinement. This was seen in Figures 2, 3, 5 and 6 with the round plate and tube.

6. Formulating the problem with a different finite element model may be helpful and collaborate results. Going to a quartered model was the only way to get good optimization results in ASTROS. One may also try other elements, such as triangular or axisymmetric, or other nodal arrangements.

7. Results that don't make good physical sense should be reevaluated. The nozzle stress results are believable because ASTROS and MSC/NASTRAN were shown to return accurate values and because the odd results at the throat do make good physical sense under careful analysis.

**Thesis Contribution And Suggestions For Future Work**

This Thesis has provided a comprehensive comparison of MSC/NASTRAN and ASTROS as applied to the analysis and optimization of shells. Problems with the porting of ASTROS to the Sun Workstations were revealed. Finite element results have been compared to exact solutions and a new nozzle analysis has probed the strengths and weaknesses of each program.

The nozzle analysis included development of a nozzle model generation program. This program has the flexibility to encompass different geometries and material properties. The analysis identified the dominant effects in key regions of the nozzle and issues of concern to the analyst.

Work was also done with an axisymmetric element. The potential for implementing this element in ASTROS was demonstrated, and a modularized FORTRAN program developed for its efficient generation. This flexible program can be used alone to generate shell element models, or it could provide the core for implementing axisymmetric analysis and optimization in ASTROS.

Future work should refine the nozzle analysis to include more accurate pressure loads, composite materials, and a regenerative cooling scheme. Additional testing of the axisymmetric element
should be done and its incorporation into ASTROS pursued. Other areas of investigation could
include optimization with the axisymmetric element and its attachment to other element types.
Appendix A. COMPUTER IMPLEMENTATION

Hardware

Finite element analysis and optimization were performed on the Sun 4 workstation using MSC/NASTRAN Version 66A and ASTROS Versions 4 and 5. Models were generated on the Sun 4, with the exception of the nozzle model, which required IMSL subroutines available on the VAX. Some post-processing used SDRC/IDEAS on the Sun 4.

Nozzle Model Generation Program

The nozzle generation program entailed significant effort and is attached for reference purposes. The program inputs geometry, material, thermal, and flow information and produces a finite-element model of a nozzle using four noded quadrilateral elements. The user may opt to exploit symmetry, and determines whether an ASTROS or MSC/NASTRAN data file is generated. An input file for the axisymmetric element program may also be created, allowing the generation of an axisymmetric model of the nozzle.

```
PROGRAM NQGEN
  DIMENSION ZI(20), RI(20), SC(20), Y(4), BREAK(20), CSCDE(20,4),
  &ZIT(20), TH(20), BREAKT(20), CSCDE(20,4)
  EXTERNAL F
  COMMON GM, ARATIO
  CHARACTER INFILE*8, DTJFILE*8,
  &TEMP*78, HEADER*8,
  &EXTN*8, TMP2*8, LABEL*1, RTYPE*8
  FORMAT(A78)
  1 FORIAT(A)

C GET STARTUP INFO FROM USER
C
PRINT*, 'INPUT FILE NAME?'
READ(*,2) INFILE
PRINT*, 'OUTPUT FILE NAME?'
READ(*,2) OUTFILE
PRINT*, 'HEADER FILE NAME?'
READ(*,2) HEADER
PRINT*, '1 - IF CREATING A NASTRAN DATA FILE'
PRINT*, '2 - IF CREATING AN ASTROS DATA FILE'
PRINT*, '3 - IF CREATING AN AXYSYMMETRIC DATA FILE'
READ*, MODEL
IF (MODEL .EQ. 1 .AND. MODEL .NE. 2 .AND. MODEL .NE. 3) THEN
  PRINT*, 'MUST BE 1, 2 OR 3 '
  STOP
ENDIF
IF (MODEL .EQ. 3) GO TO 5
PRINT*, '1 - IF DESIRE FULL MODEL'
PRINT*, '2 - IF DESIRE QUARTER MODEL'
PRINT*, '3 - IF DESIRE STRIP MODEL'
```
READ*, IQUART
IF((IQUART.NE.1.AND.:QUART.NE.2.AND.:QUART.NE.3) THEN
   PRINT*, 'MUST BE 1, 2 OR 3 !'
ENDIF

PRINT*, 'HOW MANY ELEMENTS/CIRCUMFERENCE ??'
READ*, M
PRINT*, '1 - FOR NO THERMAL, 2 - FOR YES'
READ*, MDTMP

C INITIALIZE FILES
OPEN(UNIT=6, FILE=INFILE, STATUS='OLD')
OPEN(UNIT=7, FILE=OUTFILE, STATUS='NEW')
OPEN(UNIT=8, FILE=HEADER, STATUS='OLD')
IF(MODEL.EQ.3) THEN
   OPEN(UNIT=9, FILE='AXYDATA', STATUS='NEW')
ENDIF

C INPUT MODEL INFORMATION FROM INFILE
READ(6,*) NI !NUMBER OF INTERPOLATING POINTS
READ(6,*) (ZI(I, RI), I=1,NI) !INTERPOLATING POINTS
READ(6,*) NTHRT !NTHRT - DS NO FOR THROAT
READ(6,*) NIT !NUMBER OF INTERPOLATING POINTS FOR TEMPERATURE
READ(6,*) (ZIT(I), THUI), I=1,NIT) !INTERPOLATING POINTS
READ(6,*), GM,PSO !READ IN GAMMA AND STAGNATION PRESSURE
READ(6,*), EMOD,POISS,RHO !MODULUS, NU, AND MASS DENSITY
READ(6,*), XINIT,XLB,XUB !DESIGN VARIABLE BOUNDS
READ(6,*), XLALL1,XUALL1 !LOWER & UPPER ALLOWABLES
READ(6,*), XLALL2,XUALL2
READ(6,*), XLALL3,XUALL3
READ(6,*), XLALL4,XUALL4

C SET UP FOR QUARTER MODEL
IF((IQUART.EQ.1) MM=M
IF((IQUART.EQ.2) MM=(M/4)+1
IF((IQUART.EQ.3) MM='2

C USE IMSL TO ESTABLISH SPLINE FIT
CALL CSAKM(NI, ZI, RI, THUI, NTHRT,CSI)
CALL CSAKM(NIT, ZIT,T11, PREAKT, CSOE)

C FORMATS
70 FORMAT('CONROD',418,F8.3)
72 FORMAT('FORCE',318,4F8.3)
80 FORMAT('DESVAR',218,3E8.2)
81 FORMAT('PLIST',18,AS,18)
82 FORMAT('DCONSTR',18,AS)
83 FORMAT('DCONDSP',218,AS,3A8,218,3E8.3)
90 FORMAT('EIGR',I8,AS,24X,AS,18)
91 FORMAT('TEMP',9,AS,218,3E8.3)
100 FORMAT('BEGIN BULK')
120 FORMAT('GRID+',216,2E16.9,A3,11,/,A3,11,4F8.3,216,2E16.9,A3,11,/,A3,11,4F8.3)
READ HEADER AND WRITE TO DATA FILE

IF(MODEL.EQ.3) THEN
   NF=9
ELSE
   NF=7
ENDIF
READ(8,*) NLINES ! INPUT HEADER INFORMATION
DO 155 I=1,NLINES
   READ(8,1) TEMP
   WRITE(NF,1) TEMP ! WRITE HEADER TO FILE
CONTINUE

FIT MESH TO NOZZLE GEOMETRY

RM=FLOAT(M)
PI=3.141592653589793
Z=0.0 'START AT END
NZ=0 'NUMBER OF ELEMENTS IN Z DIRECTION
STEP=2.*SIN(PI/RM)
160 NZ=NZ+1
   R=CSVAL(Z,NI-1,BREAK,CSCOEF)
   D=R*STEP 'DEAPTH TO STEP
   Z=Z+D
   IF(Z.LT.ZIGJI)) GO TO 160
   IF(Z-ZI(NI).LE.D/2.0) THEN
      FACTOR=ZI(NI)/Z
   ELSE
      FACTOR=ZI(NI)/(Z-D)
   NZ=NZ-1
   ENDIF

START BULK DATA DECK

WRITE(NF,100)
IF(MODEL.NE.3) THEN
   WRITE(7,116)100.,.O,.O,,10,.
ENDIF

MATERIAL

WRITE(NF,106) 1,EMOD,POISS,RHO,TECOEF,TREF,'+MT1 ',&'±MT1 ',BST,BSC,BSS

FIRST GRID RING WITH SPC'S

IF(MODEL.EQ.3) THEN
   WRITE(2,102) 1,0,RI(1),0.,'*GR',1,'*GR',1,0.,0.
   WRITE(3,103) 1.1,123456,0.
ELSE
   DO 165 I=1,MM
      PX=RI(1)*COS(2.*PI*FLOAT(I-1)/RM)
      PY=RI(1)*SIN(2.*PI*FLOAT(I-1)/RM)
      IF(I.LT.10) THEN
         WRITE(7,102) I,PX,PY,'*GR',I,'*GR',I,0.,0.1
      ELSEIF(I.LT.100) THEN
         WRITE(7,122) I,PX,PY,'*GR',I,'*GR',I,0.,0.1
      ELSEIF(I.LT.1000) THEN
         WRITE(7,132) I,PX,PY,'*GR',I,'*GR',I,0.,0.1
      ELSEIF(I.LT.10000) THEN
         WRITE(7,142) I,PX,PY,'*GR',I,'*GR',I,0.,0.1
      ELSE
         PRINT*, 'LOGIC ERROR LOOP 165'
   ENDIF
END
STOP
ENDIF
WRITE(7,103) 1,1,23456.0, 'WRITE SPG'S
IF(MODTMP.EQ.2) THEN
WRITE(7,91) S,1,TH(1)
ENDIF
165 CONTINUE
ENDIF
C WRITE AXELEM FIRST DATA
C IF(MODEL.NE.3) GO TO 167
WRITE(7,*), DBLE(EMOO), DBLE(POISS)
WRITE(7,*), NZ
WRITE(7,*), DOLE(EMOO), DBLE(RI(1))
167 CONTINUE
C INITIALIZE VALUES FOR LOOPING TO END OF NOZZLE
C ATHT=RI(NTHRT)*RI(NTHRT) 'THREAT AREA WITHOUT PI
Z2=0.
NZ=0 'THIS IS THE NUMBER OF TIMES STEPPED IN Z DIRECTION
D=RI(1)*STEP
C LOOP TO END OF NOZZLE
C NZ=NZA+1
Z1=Z2
Z2=Z2+D
ZIC=ZI*FACTOR
Z2C=Z2*FACTOR
R=CSVAL(Z2,NI-1,BREAK,CSCOEF)
THETMP=CSVAL(Z2C,NI-1,BREAK,CSCOET)
D=R*STEP 'SET D FOR NEXT LOOP
R2C=CSVAL(Z2C,NI-1,BREAK,CSCOEF)
IF(R2C.LE.RI(NTHRT)) THEN
R2C=RI(NTHRT)
ENDIF
IF(MODEL.EQ.3) THEN
II=NZA*M+1
ELSE IF(II.LT.10) THEN
WRITE(9,102) II,0,R2C,THETMP,0.,'+GR',II,'*GR',II,Z2C,0
ELSEIF(II.LT.100) THEN
WRITE(9,122) II,0,R2C,THETMP,0.,'+GR',II,'*GR',II,Z2C,0
ELSEIF(II.LT.1000) THEN
WRITE(9,132) II,0,R2C,THETMP,0.,'+GR',II,'*GR',II,Z2C,0
ELSEIF(II.LT.10000) THEN
WRITE(9,142) II,0,R2C,THETMP,0.,'+GR',II,'*GR',II,Z2C,0
ELSE
PRINT*, 'LOOP TOO BIG AFTER 170'
STOP
ENDIF
WRITE(9,103) 1,1,246.0.
II=NZA*M+1
WRITE(9,70) II,II,II+M,1,1. !DUMMY CONROD
C DO 175 I=1,MM
C PX=R2C*COS(2.*PI*FLOAT(I-1)/RM)
C PY=R2C*SIN(2.*PI*FLOAT(I-1)/RM)
C THETA=FLOAT(I-1)*360./RM
C ICORD=1
C II=NZA*M+1
C IF(I1.LT.10) THEN 'WRITE GRID RING
C WRITE(7,102) II,ICORD,R2C,THETA,'*GR',II,'*GR',II,Z2C,ICORD
C ELSEIF(I1.LT.100) THEN
C WRITE(7,122) II,ICORD,R2C,THETA,'*GR',II,'*GR',II,Z2C,ICORD
C ELSEIF(I1.LT.1000) THEN
C WRITE(7,132) II,ICORD,R2C,THETA,'*GR',II,'*GR',II,Z2C,ICORD
C ELSEIF(I1.LT.10000) THEN
C WRITE(7,142) II,ICORD,R2C,THETA,'*GR',II,'*GR',II,Z2C,ICORD
C ELSEIF(I1.LE.100000) THEN
C PRINT*, 'LOGIC ERROR LOOP 175'
C STOP
C ENDIF
C IF(MODTMP.EQ.2) THEN
C WRITE(7,91) S,1,THETMP
C ENDIF
IF(IQUART.EQ.2) THEN  'WRITE SPC'S IF QUARTERED MODEL
  WRITE(7,103)  1,II,246,0.0
ELSEIF(I.EQ.1) THEN
  WRITE(7,103) 1,II,246,0.0
ENDIF
ELSEIF(IQUART.EQ.3) THEN  'WRITE SPC'S FOR STRIP MODEL
  WRITE(7,103) 1,II,246,0.0.
ENDIF

CONTINUE
DO 177 I=1,MM-1  'WRITE QUAD ELEMENTS
  IF(IQUART.EQ.2) THEN
    WRITE(7,104) II,NZA,II+1,II+M,II+1
  ELSEIF(IQUART.EQ.3) THEN
    WRITE(7,104) II,NZA,II+1,II+M,II+1
  ENDIF
CONTINUE

IF(II.EQ.1) THEN  'IF NOT QUARTERED, COMPLETE RING
  WRITE(7,104) NZA+M,NZA,NZA+M-1,NZA+1,NZA+M-1
ENDIF

WRITE(7,107) NZA,1,1,1,1,1  'WRITE PSHELL FOR RING
ENDIF

178 ZP=(Z2C+ZlC)/2.0  'Z LOCATION FOR PRESSURE
RP=CSVAL(ZP,NI-i,BREAK,CSCOEF)  'RADIUS AT Z LOCATION
IF(RP.LE.RI(NTHIRT)) THEN  'COUNT LET RP< GIVEN R THROAT
  R=RI(NTHIRT)
ENDIF

ARATIO=RP*RP/ATHRT  'AREA RATIO
PPOT=(2.0/(GM+1.0))**(GM/(GM-1.0))
IF(ZP.EQ.ZI(NTHIRT)) THEN  'ITERATE TO PRESSURE
  P=P*PPOT
ELSEIF(ZP.LT.ZI(NTHIRT)) THEN
  A=A*PPOT
ENDIF

135 IF((B+A)/2.0).LE.0.0 THEN
  B=(B+A)/2.0
ELSE
  A=(B+A)/2.0
ENDIF

137 IF((A+B)/2.0).LE.0.0 THEN
  B=(A+B)/2.0
ELSE
  A=(A+B)/2.0
ENDIF

IF(MODEL.EQ.3) THEN
  WRITE(7,*) DBLE(Z2C),DBLE(R2C),DBLE(R2)
  WRITE(9,72) 1,NZA*M+1,0,1,1,1,1,1  'DUMMY LOAD
  GO TO 210
ENDIF

DO 190 I=1,MM-1  'WRITE PLOAD ON ELEMENTS
  IF(I.EQ.1) THEN
    WRITE(7,105) 1,P,II,II+1,II+M,II+1
  ELSE
    WRITE(7,105) 1,P,NZA+M,NZA+M-1,NZA+M+1,NZA+M+M
  ENDIF
CONTINUE

IF(IQUART.EQ.1) THEN  'IF NOT QUARTERED MODEL, COMPLETE PLOAD
  WRITE(7,105) 1,P,NZA+M,NZA+M-1,NZA+M+1,NZA+M+M
ENDIF

IF(MODEL.EQ.1) THEN  'WRITE DESVAR, DVPREL FOR NASTRAN
  IF(NZA.LT.10) THEN
    WRITE(7,106) NZA,'T',NZA,XINIT,XLII,XLII
    WRITE(7,110) NZA,NZA,ADM,'+DP',NZA,'+DP',NZA,NZA
  ELSEIF(NZA.LT.100) THEN
    WRITE(7,110) NZA,NZA,ADM,'+DP',NZA,'+DP',NZA,NZA
  ELSEIF(NZA.LT.1000) THEN
    WRITE(7,110) NZA,NZA,ADM,'+DP',NZA,'+DP',NZA,NZA
  ELSEIF(NZA.LT.10000) THEN
    WRITE(7,110) NZA,NZA,ADM,'+DP',NZA,'+DP',NZA,NZA
  ELSEIF(NZA.GE.10000) THEN
    PRINT*, 'LOGIC ERROR BETWEEN NGA AND S-
  STOP
ENDIF
NOS=2 'NUMBER OF STRESS COMPONENTS FOR CONSTRAINTS
DD 200 I=1,NOS 'WRITE DRESP1 FOR NASTRAN
IF(I.EQ.1) THEN
  LABEL='S'
  RTYPE='STRESS'
  PTYPE='PSHELL'
  IATTA=9
  IATT1=NZA
ELSEIF(I.EQ.2) THEN
  LABEL='S'
  RTYPE='STRESS'
  PTYPE='PSHELL'
  IATTA=17
  IATT1=NZA
ENDIF

IF((10*NZA+I).LT.100) THEN
  WRITE(7,111) NOS*NZA-NOS+I, LABEL, 10*NZA+I, RTYPE, PTYPE, & IATTA, IATT1
ELSEIF((10*NZA+1).LT.1000) THEN
  WRITE(7,121) NOS*NZA-NOS+I, LABEL, 10*NZA+I, RTYPE, PTYPE, & IATTA, IATT1
ELSEIF((10*NZA+1).LT.10000) THEN
  WRITE(7,131) NOS*NZA-NOS+I, LABEL, 10*NZA+I, RTYPE, PTYPE, & IATTA, IATT1
ELSEIF((10*NZA+1).LT.100000) THEN
  WRITE(7,141) NOS*NZA-NOS+I, LABEL, 10*NZA+I, RTYPE, PTYPE, & IATTA, IATT1
ENDIF

200 CONTINUE
WRITE(7,112) NOS*NZA-1, XLALL1, XUALL1 'WRITE DCONSTR FOR NASTRAN
WRITE(7,112) NOS*NZA, XLALL2, XUALL2
ELSEIF(MODEL.EQ.2) THEN 'WRITE DESVARP, PLIST FOR ASTROS
WRITE(7,80) NZA, NZA, XLB, XUB, XINIT
WRITE(7,81) NZA, 'PSHELL ', NZA
ENDIF

C C NOW THAT AT END OF NOZZLE, FINISH OFF MODEL
C IF(MODEL.EQ.1) THEN 'FINAL NASTRAN DRESP1, DESOBJ, DOPTPRM, PARAM
  PTYPE='P
  WRITE(7,111) 88888,'D', 88, 'DISP ', PTYPE, 1, NZ*M+1
  WRITE(7,112) 88888, XLALL3, XUALL3
  WRITE(7,113) 99999, 'W', 'WEIGHT ', W
  WRITE(7,114) 99999, 'MIN
  WRITE(7,115) 2, 3, 20, _5_ 01
  WRITE(7,118)
ELSEIF(MODEL.EQ.2) THEN 'WRITE FINAL DCONSTR FOR ASTROS
  WRITE(7,82) 1, 'VMISES
  WRITE(7,83) 1, 1, 'LOWER ', XLALL3, 'D1
  WRITE(7,83) 1, 2, 'UPPER ', XLALL3, 'D2
  WRITE(7,83) 1, 3, 'LOWER ', XLALL4, 'D3
  WRITE(7,83) 1, 4, 'UPPER ', XLALL4, 'D4
ELSEIF(MODEL.EQ.3) THEN
  GO TO 250
ENDIF
WRITE(7,90) 75, 'GIV
C NOW CLOSE OFF FILES AND EXIT PROGRAM
C 250 CLOSE(6)
CLOSE(7)
CLOSE(8)
IF(MODEL.EQ.3) THEN
  CLOSE(9)
ENDIF
STOP
END
C C IDEAL CONVERGENT/DIVERGENT NOZZLE EQN
C REAL FUNCTION F(X)
COMMON GM, GAMMA
A=GM+1
P=(M+1)
R/X
Axisymmetric Finite Element Generation Program

The axisymmetric element program also involved significant effort and is attached for reference. The FORTRAN program inputs geometry and material properties and produces global stiffness, load and stress recovery matrices in direct matrix form. It can be easily modified since it is modularized and well documented internally. A Gaussian quadrature integration scheme is used, and the order of integration and weighting parameters can be easily changed in the preface.

Double precision is used throughout to retain accuracy.

```fortran
PROGRAM AXYSYM
   DOUBLE PRECISION E,T(100),DNU,X(100),R(100),PR(100),DKGELM(6,6),
    &PWT(2,10),TEMP(2),TEM(6),TEM(14,9),SREC(7,6),
    &TEMP(9,3)
   CHARACTER iNr'ILE*8, OUTFILE*8

   ! EXPLANATION OF VARIABLES
   E == MG/jUS
   T == VECTOR CONTAINING ELEMENT THICKNESSES
   DNU == POISSONS RATIO
   Xi, Ri == VECTORS CONTAINING COORDINATES OF END POINTS
   FR == VECTOR CONTAINING PRESSURE ON ELEMENTS NORMAL & CENTERED
   DKGELM == ELEMENT STIFFNESS MATRIX, GLOBAL
   NOD == NODE CONNECTIVITY TABLE
   NG == NUMBER OF GRID POINTS
   NE == NUMBER OF ELEMENTS
   NOQ == ORDER OF GAUSSIAN QUADRATURE
   PWT == ARRAY CONTAINING PSI AND WEIGHTS FOR QUADRATURE (I)
   TEM1 == TEMPORARY STORAGE
   TEM1 == MATRIX W/COLUMNS BEING ELEMENT LOAD VECTORS, GLOBAL
   SREC == STRESS RECOVERY MATRIX

   ! ESTABLISH GAUSSIAN QUADRATURE DATA
   NOQ=7
   PWT(1,1)=-0.949107912342158524526189684048
   PWT(2,1)=0.129484966168889693270611432679
   PWT(1,2)=-0.741531185599394439863864773281
   PWT(2,2)=0.27970539148927666790146771424
   PWT(1,3)=-0.4058451513773971669066666412077
   PWT(2,3)=0.417959183673469394305050504048
   PWT(1,4)=-0.000000000000000000000000000000
   PWT(2,4)=0.417959183673469394305050504048
   PWT(1,5)=PWT(1,4)
   PWT(2,5)=PWT(2,4)
   PWT(1,6)=PWT(1,5)
   PWT(2,6)=PWT(2,5)
   PWT(1,7)=PWT(1,6)
   PWT(2,7)=PWT(2,6)

   ! GET STARTUP FILE INFORMATION FROM USER
   /I FORMAT(A)
   /I FORMAT(A)
```

The program stores and processes various geometry-related parameters and uses a Gaussian quadrature scheme for integration. It also outlines variables and their meanings for clarity.
FORMAT(E16.8)
PRINT*, 'INPUT FILE NAME ??'
READ(*,2) INFILE
PRINT*, 'OUTPUT FILE NAME ??'
READ(*,2) OUTFILE

C INITIALIZE FILES
C OPEN(UNIT=6, FILE=INFILE, STATUS='OLD')
OPEN(UNIT=7, FILE=OUTFILE, STATUS='NEW')

C INPUT DATA
C READ(6,*), E,DNU
READ(6,*), NE
READ(6,*), X(1),R(1)
DO 10 I=1,NE
READ(6,*), X(I+1),R(I+1),PR(I)
10 CONTINUE
DO 20 I=1,NE
READ(6,*), T(I)
20 CONTINUE

C CREATE AND OUTPUT KG
C OPEN(UNIT=8, FILE='SCRATCH1', STATUS='NEW')
DO 200 I=1,NE+1
IF(I.EQ.1) THEN
CALL KELME(T(I),DNU,X(I),R(I),X(I+1),R(I+1),NOQ,PWT,DKGELM)
DO 150 J=1,6
WRITE(8,3), J
WRITE(8,3), K
WRITE(8,3), 0.00+00
CONTINUE
150 CONTINUE
ELSEIF(I.EQ.NE+1) THEN
CALL KELME(T(I-1),DNU,X(I-1),R(I-1),X(I),R(I),NOQ,PWT,DKGELM)
DO 160 J=4,6
WRITE(8,3), J
WRITE(8,3), K
WRITE(8,3), 0.00+00
CONTINUE
160 CONTINUE
ENDIF
DO 149 K=1,NE*6+6
WRITE(8,3), 0.00+00
149 CONTINUE
DO 154 K=1,6*NE-6
WRITE(8,3), 0.00+00
154 CONTINUE
ELSE
CALL KELME(T(I-1),DNU,X(I-1),R(I-1),X(I),R(I),NOQ,PWT,DKGELM)
DO 160 J=4,6
WRITE(8,3), J
WRITE(8,3), K
WRITE(8,3), 0.00+00
CONTINUE
160 CONTINUE
ENDIF
DO 158 K=1,6
WRITE(8,3), 0.00+00
158 CONTINUE

ELSE
DO 166 J = 1, 3
DO 165 K = 1, 9
 TEMP4(K, J) = 0.0D+00
165 CONTINUE
166 CONTINUE
CALL KELM(E, T(I - 1), DNU, X(I - 1), R(I - 1), NOQ, PWT, DKGELM)
DO 170 J = 1, 3
DO 168 K = 1, 6
 TEMP4(K, J) = DKGELM(K, J + 3)
168 CONTINUE
170 CONTINUE
CALL KELM(E, T(I), DNU, X(I), R(I), X(I + 1), R(I + 1), NOQ, PWT, DKGELM)
DO 180 J = 1, 3
DO 178 K = 1, 6
 TEMP4(K + 3, J) = TEMP4(K + 3, J) + DKGELM(K, J)
178 CONTINUE
180 CONTINUE
WRITE(8, 3) 2
WRITE(8, 3) 2*(3*I) - 5 + 2*(J - 1)
WRITE(8, 3) 1
184 CONTINUE
DO 188 K = 1, 9
 WRITE(8, 3) 1
 WRITE(8, 4) TEMP4(K, J)
 WRITE(8, 3) 1
 WRITE(8, 4) 0.0D+00
188 CONTINUE
IF (I.EQ.2) GO TO 184
DO 184 K = 1, 6*I - 12
 WRITE(8, 3) 1
 WRITE(8, 4) 0.0D+00
184 CONTINUE
DO 188 K = 1, 9
 WRITE(8, 3) 1
 WRITE(8, 4) TEMP4(K, J)
 WRITE(8, 3) 1
 WRITE(8, 4) 0.0D+00
188 CONTINUE
IF (I.EQ.NE) GO TO 189
DO 189 K = 6*I + 8, NE*6 + 6
 WRITE(8, 3) 1
 WRITE(8, 4) 0.0D+00
189 CONTINUE
DO 220 J = 4, 6
 WRITE(8, 3) 1
 WRITE(8, 4) TEMP2(J)
 WRITE(8, 3) 1
 WRITE(8, 4) 0.0D+00
220 CONTINUE
ELSE IF (I.EQ.NE + 1) THEN
 CALL PLICAD(X(I), R(I), X(I + 1), R(I + 1), PR(I), NOQ, PWT, TEMP2)
 WRITE(8, 3) 2
 WRITE(8, 3) 1
 WRITE(8, 3) 2
 WRITE(8, 3) 1
 DO 220 J = 4, 6
 WRITE(8, 3) 1
 WRITE(8, 4) TEMP2(J)
 WRITE(8, 3) 1
 WRITE(8, 4) 0.0D+00
220 CONTINUE
C CREATE NODAL LOAD VECTOR
C OPEN(UNIT=8, FILE='SCRATCH2', STATUS='NEW')
 DO 300 I = 1, NE + 1
 IF (I.EQ.1) THEN
 CALL PLICAD(X(I), R(I), X(I + 1), R(I + 1), PR(I), NOQ, PWT, TEMP2)
 WRITE(8, 3) 2
 WRITE(8, 3) 1
 WRITE(8, 3) 2
 WRITE(8, 3) 1
 DO 220 J = 4, 6
 WRITE(8, 3) 1
 WRITE(8, 4) TEMP2(J)
 WRITE(8, 3) 1
 WRITE(8, 4) 0.0D+00
220 CONTINUE
C
C ELSEIF (I.EQ.NE + 1) THEN
C CALL PLICAD(X(I - 1), R(I - 1), X(I), R(I), PR(I - 1), NOQ, PWT, TEMP2)
C DO 220 J = 4, 6
C WRITE(8, 3) 1
C WRITE(8, 3) 1
C WRITE(8, 3) TEMP2(J)
C WRITE(8, 3) 1
C WRITE(8, 4) 0.0D+00
220 CONTINUE
C ELSE
CALL PLOAD(X(I-1),R(I-1),X(I),R(I),NOQ,PWT,TMP2)
TEMP(1)=TEMP2(4)
TEMP(2)=TEMP2(5)
TEMP(3)=TEMP2(6)
CALL PLOAD(X(I),R(I),X(I+1),R(I+1),NOQ,PWT,TMP2)
TEMP(1)=TEMP(1)+TEMP2(1)
TEMP(2)=TEMP(2)+TEMP2(2)
TEMP(3)=TEMP(3)+TEMP2(3)
DO 240 J=1,3
   WRITE(8,3) 1
   WRITE(8,4) TEMP(J)
   WRITE(8,3) 1
   WRITE(8,4) 0.0D+00
240 CONTINUE

ENDIF
300 CONTINUE
WRITE(8,3) 3
CALL ASTOUT('MYPL','RDP ','REC ','NE*6+6,1,
# 'SCRATCH2',7)
CLOSE(8)

C CREATE AND OUTPUT STRESS RECOVERY MATRIX
C
OPEN(UNIT=S,FILE='SCRATCH3',STATUS='NEW')
DO 400 I=1,NE+1
   IF(I.EQ.1) THEN
      CALL RECOVER(X(I),R(I),X(I+1),R(I+1),DNU,E,T(I),SREC)
      DO 330 J=1,3
         WRITE(8,3) 2
         WRITE(I) T(I)
         WRITE(8,3) 2
         WRITE(8,4) SREC(K,J)
      330 CONTINUE
      IF(NE.EQ.1) GO TO 329
      ELSE
         DO 328 K=8,7*NE
            WRITE(8,3) 3
            WRITE(8,4) 0.0D+00
      328 CONTINUE
      ELSEIF(I.EQ.NE+1) THEN
         CALL RECOVER(X(I-1),R(I-1),X(I),R(I),DNU,E,T(I-1),SREC)
         DO 340 J=4,6
            WRITE(8,3) 2
            WRITE(8,3) 1+6*NE+2*(J-4)
            WRITE(8,3) 2
            WRITE(8,4) 0.0D+00
      340 CONTINUE
      ELSE
         CALL RECOVER(X(I-1),R(I-1),X(I),R(I),DNU,E,T(I-1),SREC)
      400 CONTINUE
      ENDIF

DO 350 I=1,7
   DO 348 J=1,6
      TEMP3(K,J)=SREC(K,J)
   CONTINUE
348 CONTINUE
CALL RECOVER(X(I),R(I),X(I+1),R(I+1),DMU,E,T(I),SREC)
DO 360 J=1,3
   WRITE(8,3) 2
   WRITE(8,3) 6*(I-1)+2*J-1
   WRITE(8,3) 1
   IF(I.EQ.2) GO TO 354
   DO 354 K=1,7*I-14
      WRITE(8,3) 1
      WRITE(8,4) 0.D+00
354 CONTINUE
   DO 356 K=1,7*NE
      WRITE(8,3) 1
      WRITE(8,4) TEMP3(K,J+3)
356 CONTINUE
   DO 358 K=1,7*NE
      WRITE(8,3) 1
      WRITE(8,4) SREC(K,J)
358 CONTINUE
   IF(I.EQ.NE) GO TO 359
   DO 359 K=7*I+2,7*NE
      WRITE(8,3) 1
      WRITE(8,4) 0.D+00
359 CONTINUE
   WRITE(9,3) 2
   WRITE(8,3) 6*I+2*J-6
   WRITE(8,3) 2
   WRITE(8,3) 1
   DO 360 K=1,7*NE
      WRITE(8,3) 1
      WRITE(8,4) 0.D+00
360 CONTINUE
ENDIF
400 CONTINUE
WRITE(8,3) 3
CALL ASTOUT('MYSREC ','RDP ','REC ',7*NE,NE*6+6,'SCRATCH3',7)
CLOSE(8)
STOP
END
C** THIS SUBROUTINE INPUTS GEOMETRY AND MATERIAL DATA AND RETURNS AN ELEMENT
C STIFFNESS MATRIX (DKELM) IN GLOBAL COORDINATES.
C
C SUBROUTINE KELM(E,T,DMU,X1,R1,X2,R2,NOQ,PWT,DKGELM)
DOUBLE PRECISION E,T,DMU,X1,R1,X2,R2,PWT(2,NOQ),
&DK(6,6),DKGELM(6,65,B(5,6),G(5,5),DL
C
C EXPLANATION OF VARIABLES, (I) INPUT, (O) OUTPUT
C E == MODULUS (I)
C T == ELEMENT THICKNESS (I)
C DMU == POISSONS RATIO (I)
C X1, R1 == COORDINATES OF END POINTS (I)
C NOQ == ORDER OF GAUSSIAN QUADRATURE (I)
C PWT == ARRAY CONTAINING PSI AND WEIGHTS FOR QUADRATURE (I)
C DKGELM == ELEMENT MATRIX, GLOBAL (O)
C DK == ELEMENT MATRIX FOR INTERNAL USE, SQUARE, LOCAL COORDINATES
C B == B MATRIX
C G == E MATRIX
C DL == LENGTH OF ELEMENT
C
C CLEAR DK MATRIX
C DO 10 I=1,6
   DO 8 J=1,6
      DK(I,J)=0.0D+00
8 CONTINUE
10 CONTINUE
C PERFORM GAUSSIAN QUADRATURE TO FORM ELEMENT K MATRIX, DL/2 IS 1.0
C DO 100 I=1,NOQ
   CALL EMAT(X1,R1,X2,R2,PWT(I,1),H,DL)
   CALL EMAT(DMU,E,T,G)
100 CONTINUE
DO 90 J1=1,6
DO 90 J2=1,5
DO 90 J3=1,5
DO 90 J4=1,6
DK(J1,J4)=DK(J1,J4)+PWT(2,1)*B(J2,J1)*G(J2,J3)*B(J3,J4)
90 CONTINUE
100 CONTINUE
DO 145 I=1,6
DO 145 J=1,6
DK(I,J)=DK(I,J)*DL/2.0D+00
145 CONTINUE
150 CONTINUE
C ROTATE TO GLOBAL COORDINATES
C CALL ROTATE(X1,R1,X2,R2,6,6,DX,5,DKELM)
C FINISHED
C RETURN
END

C THIS SUBROUTINE INPUTS GEOMETRY, PRESSURE, AND GAUSSIAN DIRECTIONS, AND
C OUTPUTS ELEMENTAL NODAL LOAD VECTOR IN GLOBAL COORDINATES.
C
C SUBROUTINE PLOAD(XI,RI,X2,R2,PR,NOQ,PWT,PGENL)
DOUBLE PRECISION XI,RI,X2,R2,PR,PWT(2,NOQ),PGENL(6),PEENL(6),
&DN(3,6),DL,P(3)

C EXPLANATION OF VARIABLES, (I) INPUT, (O) OUTPUT
C XI, RI == COORDINATES OF END POINTS (I)
C PR == STATIC PRESSURE (I)
C NOQ == ORDER OF GAUSSIAN QUADRATURE (I)
C PWT == ARRAY CONTAINING PSI AND WEIGHTS FOR QUADRATURE (I)
C PGENL == VECTOR OF GLOBAL EQUIVALENT NODAL LOADS (O)
C PEENL == EQUIVALENT NODAL LOADS IN ELEMENT COORDINATES
C DN == SHAPE FUNCTION MATRIX
C DL == LENGTH OF ELEMENT
C P == LOAD VECTOR FOR INTERNAL USE, ELEMENT COORDINATES
C CLEAR PEENL AND ESTABLISH P
C DO 10 I=1,6
PEENL(I)=0.0D+00
10 CONTINUE
P(1)=0.0D+00
P(2)=PR
P(3)=0.0D+00
C PERFORM GAUSSIAN QUADRATURE TO GENERATE EQUIVALENT NODAL LOADS
C DO 100 I=1,NOQ
CALL NMAT(X1,RI,X2,R2,PWT(I,1),DN,DL)
DO 90 J1=1,6
DO 90 J2=1,3
PEENL(J1)=PEENL(J1)+(PWT(2,1)*DN(J2,J1)*P(J2))
90 CONTINUE
100 CONTINUE
DO 110 I=1,6
PEENL(I)=PEENL(I)*DL/2.0D+00
110 CONTINUE
C ROTATE TO GLOBAL COORDINATES
C CALL ROTATE(X1,RI,X2,R2,6,1,PEENL,2,PGENL)
C FINISHED
C RETURN
END

C THIS SUBROUTINE INPUTS GEOMETRY AND MATERIAL PROPERTIES AND RETURNS THE
C STRESS RECOVERY MATRIX. THIS MATRIX WHEN PREMULTIPLYING GLOBAL
C DISPLACEMENTS YIELDS STRESSES AT THE MIDSPAN AND +/- T/2 IN ELEMENT
C COORDINATES AT THE CENTER OF THE ELEMENT.
C
C SUBROUTINE RECOVERY(X1,X2,X2,X2,R1,R2,E,T,SHC)
DOUBLE PRECISION X1,X2,X2,X2,R1,R2,E,T,SHC(7,0),G(5,5),K(5,6),
&L(7,7),M(7,7),N(7,7),M(7,7)}
C EXPLANATION OF VARIABLES, (I) INPUT, (O) OUTPUT
C Xi, Ri == COORDINATES OF END POINTS (I)
C DNU == POISSON'S RATIO (I)
C E == MODULUS (I)
C T == ELEMENT THICKNESS (I)
C SREC == STRESS RECOVERY MATRIX (O)
C G == MATERIAL PROPERTY MATRIX
C B == B MATRIX
C DL == LENGTH OF ELEMENT
C BMC == BENDING MOMENT CONVERSION MATRIX
C TINV == INVERSE OF THICKNESS
C Ci, TEMPi == DUMMY CONSTANTS & MATRICES
C CLEAR SREC, TEMP AND BMC MATRICES
C DO 10 J=1,7
  DO 8 J=1,6, TEMP(I,J)=0.0D+00
  CONTINUE
DO 9 J=1,5
  BMC(I,J)=0.0D+00
  CONTINUE
10 CONTINUE
C CREATE CONSTANTS FOR BMC CREATION
C TINV=1.0D+00/T
C CI=6.0D+00/(T*T)
C CREATE BMC, G, AND B MATRICES
BMC(1,1)=TINV
BMC(1,3)=CI
BMC(2,2)=TINV
BMC(2,4)=CI
BMC(3,1)=TINV
BMC(4,2)=TINV
BMC(5,1)=TINV
BMC(5,3)=CI
BMC(5,5)=TINV
BMC(6,4)=CI
BMC(7,5)=1.0D+00
CALL BMAT(X1,RI,X2,R2,0.0D+00,B,DL)
CALL EMAT(DNU,E,T,G)
C DO BMC*E*B
C DO 100 J1=1,7
  DO 100 J2=1,5
    DO 100 J3=1,5
      DO 100 J4=1,6
        TEMP(JIJ4)=TEMP(J1,J4)+BMC(J1,J2)*G(J2,J3)*B(J3,J4)
  100 CONTINUE
C TRANSFORM TO ALLOW OPERATION OFF GLOBAL DISPLACEMENTS
C CALL ROTATE(X1,RI,X2,R2,7,6,TEMP,3,SREC)
C FINISHED
C RETURN
END
C THIS SUBROUTINE INPUTS GEOMETRY AND THE GENERALIZED COORDINATE OF INTEREST
C AND RETURNS THE B MATRIX AND ELEMENT LENGTH.
C SUBROUTINE BMAT(X1,R1,X2,R2,DKSI,B,DL)
DOUBLE PRECISION X1,R1,X2,R2,DKSI,B(6),DL,R,PHI,CPR,SPR,R,
&DKMDKP, DLINV
C EXPLANATION OF VARIABLES, (I) INPUT, (O) OUTPUT
C Xi, Ri == COORDINATES OF END POINTS (I)
C DKSI == KSI TRANSFORMED VARIABLE, S -- KSI (I)
C R == B MATRIX, DERIVATIVE OF SHAPE FUNCTION, SO TO SPEAK (O)
C DL == LENGTH OF ELEMENT (O)
C PHI == RADIAN AT CENTER OF ELEMENT
C
C THIS SUBROUTINE INPUTS GEOMETRY AND THE GENERALIZED COORDINATE OF INTEREST
C AND RETURNS THE B MATRIX AND ELEMENT LENGTH.
C SUBROUTINE BMAT(X1,R1,X2,R2,DKSI,B,DL)
DOUBLE PRECISION X1,R1,X2,R2,DKSI,B(6),DL,R,PHI,CPR,SPR,R,
&DKMDKP, DLINV
C EXPLANATION OF VARIABLES, (I) INPUT, (O) OUTPUT
C Xi, Ri == COORDINATES OF END POINTS (I)
C DKSI == KSI TRANSFORMED VARIABLE, S -- KSI (I)
C R == B MATRIX, DERIVATIVE OF SHAPE FUNCTION, SO TO SPEAK (O)
C DL == LENGTH OF ELEMENT (O)
C PHI == RADIAN AT CENTER OF ELEMENT
C
C CLEAR G MATRIX
C     DO 10 I=1,5
     DO 8 J=1,5
     G(I,J)=0.0D+00
8     CONTINUE
10    CONTINUE
C CALCULATE RO,L,PHI
C
R0=(R2+R1)/2.0D+00
PHI=DATAN2(R2-R1,X2-X1)
DL=(X2-X1)/DCOS(PHI)
C
C CREATE CONSTANTS FOR ELEMENT CREATION
C
R=R0+(DKSI*DL*DSIN(PHI)/2.0D+00)
CPR=DCOS(PHI)/R
SPR=DSIN(PHI)/R
DKM=0.5D+00*(1.0D+00-DKSI)
DKP=0.5D+00*(1.0D+00+DKSI)
DLINV=1.0D+00/DL
C
C NOW CREATE B MATRIX
C
B(1,2)=-DLINV
B(1,5)=DLINV
B(2,2)=DKM*SPR
B(2,5)=DKP*SPR
B(2,1)=DKM*CPR
B(2,4)=DKP*CPR
B(2,3)=DLINV*(1.0D+00-(DKSI*DKSI))*CPR/8.0D+00
B(2,6)=-B(2,3)
B(3,3)=DLINV
B(3,6)=-DLINV
B(4,3)=DKM*SPR
B(4,6)=DKP*SPR
B(5,1)=-DLINV
B(5,4)=DLINV
B(5,3)=(-(DKSI/2.0D+00)-DKM
B(5,6)=(DKSI/2.0D+00)-DKP
C
C FINISH'D
C
RETURN
END
C
C******************************************************************************
C THIS SUBROUTINE INPUTS MATERIAL PROPERTIES AND RETURNS THE MATERIAL
C MATRIX FOR USE IN CREATING THE STIFFNESS MATRIX.
C
SUBROUTINE EMAT(DNU,E,T,G)
DOUBLE PRECISION DNU,E,T,G(5,5),C1
C
EXPLANATION OF VARIABLES, (I) INPUT, (O) OUTPUT
DNU == POISSONS RATIO (I)
E == MODULUS (I)
T == ELEMENT THICKNESS (I)
G == MATERIAL PROPERTY MATRIX (O)
C
C CLEAR G MATRIX
C     DO 10 I=1,5
     DO 8 J=1,5
     G(I,J)=0.0D+00
8     CONTINUE
10    CONTINUE
C
C CREATE CONSTANTS FOR ELEMENT CREATION
C
C1=E*T/(1.0D+00-(DNU*DNU))
C
C NOW CREATE EMAT MATRIX
C
G(I,J)=C1
G(I,J)=DNU*C1
G(J,J)=G(I,J)
G(J,I)=C1
G(J,J)=T*C1/(2.0D+00)
G(3,4) = DN1*G(3,3)  
G(4,3) = G(3,4)  
G(4,4) = G(3,3)  
G(5,5) = 5.0D+00*E*T/(12.0D+00*(1.0D+00+DN1))

C FINISHED
C RETURN
END

C*************************************************************************
C THIS SUBROUTINE INPUTS GEOMETRY AND THE GENERALIZED COORDINATE OF INTEREST
C AND RETURNS THE N MATRIX AND ELEMENT LENGTH
C*************************************************************************

SUBROUTINE NMAT(XIRI,X2,R2,DKSI,DN,DL)
DOUBLE PRECISION X1,R1,X2,R2,DKSI,DN(3,6),DL,DN1,DN2,DN3

C EXPLANATION OF VARIABLES, (I) INPUT, (O) OUTPUT
C X1, R1 == COORDINATES OF END POINTS (I)
C DKSI == KSI TRANSFORMED VARIABLE, S --> KSI (I)
C DN == N, THE SHAPE FUNCTION MATRIX (O)
C DL == LENGTH OF ELEMENT (O)
C DN1 == ELEMENTS OF SHAPE FUNCTION MATRIX

CLEAR DN MATRIX

DO 10 I=1,3
DO 8 J=1,6
DN(I,J)=O.OD+00
8 CONTINUE
10 CONTINUE

CALCULATE DL

DL=(X2-X1)/DCOS(DATAN2(R2-R1,X2-X1))
DN1=0.5D+00*(1.0D+00-DKSI)
DN2=0.5D+00*(1.0D+00+DKSI)
DN3=DL*(1.0D+00-(DKSI*DKSI))/8.0D+00

NOW CREATE DN MATRIX

DN(1,2)=DN1
DN(1,5)=DN2
DN(2,1)=DN1
DN(2,4)=DN2
DN(2,3)=DN3
DN(2,6)=-DN3
DN(3,3)=DN1
DN(3,6)=DN2

C FINISHED
C RETURN
END

C*************************************************************************
C THIS SUBROUTINE INPUTS GEOMETRY AND A MATRIX ENTITY, CREATES ROTATION
C MATRIX USING THE FORM u=Tu (CAPS GLOBAL), AND RETURNS A ROTATED ENTITY PER
C INSTRUCTION CODE. ENTITY MUST BE COMPATIBLE WITH THE OPERATION.
C*************************************************************************

SUBROUTINE ROTATE(X1,R1,X2,R2,NROW,NCOL,ENTIT,INSTR,ENOUT)
DOUBLE PRECISION X1,R1,X2,R2,ENTIT(NROW,NCOL),T(6,6),PHI,DC,DS,
& ENOUT(NROW,NCOL)

C EXPLANATION OF VARIABLES, (I) INPUT, (O) OUTPUT
C X1, R1 == COORDINATES OF END POINTS (I)
C NROW == NUMBER OF ROWS IN ENTIT (I)
C NCOL == NUMBER OF COLUMNS IN ENTIT (I)
C ENIT == ENTITY THAT IS BEING TRANSFORMED (I)
C INSTR == INSTRUCTIONS FOR ROTATION AS BELOW (I)
C 1 --> PREMULTIPLY BY T
C 2 --> PREMULTIPLY BY T TRANSPOSE
C 3 --> POSTMULTIPLY ENTIT BY T
C 4 --> POSTMULTIPLY ENTIT BY T TRANSPOSE
C 5 --> PRE & POST BY T TRANSPOSE & T, RESPECTIVELY
C 6 --> PRE & POST BY T & T TRANSPOSE, RESPECTIVELY
C ENOUT == TRANSFORMED ENTITY (O)
C T == TRANSFORMATION MATRIX TO GIVE u=Tu
C PHI == ANGLE FROM Z AXIS IN DEGREES, FIG 12.4-11
C
C CHECK FOR COMPATIBILITY


IF(NCOL.NE.6.AND.NROW.NE.6) GO TO 510

CALCULATE PHI, COS(PHI), SIN(PHI)

PHI=DATA2(R2-R1,X2-X1)
DC=DCOS(PHI)
DS=DSIN(PHI)

CLEAR ROTATION AND ENTITY OUTPUT MATRICES

DO 10 I=1,6
   DO 6 J=1,6
      T(I,J)=0.0D+00
   CONTINUE
10 CONTINUE

DO 20 I=1,NCOL
   DO 18 J=1,NROW
      ENOUT(J,I)=0.0D+00
18 CONTINUE
20 CONTINUE

CREATE ROTATION MATRIX

T(1,1)=DC
T(1,2)=-DS
T(2,1)=DS
T(2,2)=DC
T(4,4)=DC
T(4,5)=-DS
T(5,4)=DS
T(5,5)=DC
T(3,3)=1.0D+00
T(6,6)=1.0D+00

TRANSFORM ENTITY

IF(INSTR.EQ.1) THEN
   DO 100 J1=1,6
      DO 100 J2=1,6
         ENOUT(J1,J2)=ENOUT(J1,J2)+(T(J1,J2)*ENTIT(J1,J2))
100 CONTINUE
ELSEIF(INSTR.EQ.2) THEN
   DO 110 J1=1,6
      DO 110 J2=1,6
         DO 110 J3=1,NCOL
            ENOUT(J1,J3)=ENOUT(J1,J3)+(T(J1,J2)*ENTIT(J1,J3))
110 CONTINUE
ELSEIF(INSTR.EQ.3) THEN
   DO 120 J1=1,NROW
      DO 120 J2=1,6
         DO 120 J3=1,6
            ENOUT(J1,J3)=ENOUT(J1,J3)+(T(J1,J2)*ENTIT(J1,J3))
120 CONTINUE
ELSEIF(INSTR.EQ.4) THEN
   DO 130 J1=1,NROW
      DO 130 J2=1,6
         DO 130 J3=1,6
            ENOUT(J1,J3)=ENOUT(J1,J3)+(T(J1,J2)*ENTIT(J1,J3))
130 CONTINUE
ELSEIF(INSTR.EQ.5) THEN
   IF(NCOL.NE.6.OR.NROW.NE.6) GO TO 500
   DO 140 J1=1,6
      DO 140 J2=1,6
         ENOUT(J1,J2)=ENOUT(J1,J2)+(T(J1,J2)*ENTIT(J1,J2))
140 CONTINUE
ELSEIF(INSTR.EQ.6) THEN
   IF(NCOL.NE.6.OR.NROW.NE.6) GO TO 500
   DO 150 J1=1,6
      DO 150 J2=1,6
         DO 150 J3=1,6
            ENOUT(J1,J3)=ENOUT(J1,J3)+(T(J1,J2)*ENTIT(J1,J3))
150 CONTINUE
ELSE
   PRINT*, 'ERROR IN ROTATE, INCORRECT SIZING'
STOP
END
GO TO 600
C
C ERROR STATEMENTS
C 500 PRINT*, 'ERROR IN ROTATE, ENTIT NOT SQUARE'
STOP
510 PRINT*, 'ERROR IN ROTATE, ROWS AND/OR COLUMNS NOT COMPATIBLE'
STOP
C
C FINISHED
C 600 RETURN
END
C
C THIS SUBROUTINE INPUTS FILE DATA AND WRITES AN ASTROS COMPATIBLE DIRECT C MATRIX INPUT BULK DATA FILE USING LARGE FIELDS.
C
SUBROUTINE ASTOUT(MATRIX, PRECIS, FORM, NROWS, NCOLS, INFILE, NOFILE)
DOUBLE PRECISION AM4
DIMENSION INSTR(5), ICR(4)
CHARACTER MATRIX*8, PRECIS*8, FORM*8, INFILE*S
OPENCUNIT=8, FILE=INFILE, STATUS='OLD')
REWIND(8)
C
C EXPLANATION OF VARIABLES, (I) INPUT, (O) OUTPUT
C MATRIX == CHARACTER DENOTING MATRIX NAME (I)
C PRECIS == CHARACTER DENOTING PRECISION (I)
C FORM == CHARACTER DENOTING FORM OF MATRIX (I)
C NROWS, NCOLS == NUMBER OF ROWS AND COLUMNS IN MATRIX (I)
C INFILE == INPUT FILE NAME (I)
C NOFILE == FORTRAN OUTPUT FILE NUMBER (I)
C A == REAL MATRIX TERM
C INSTR == INSTRUCTION CODE READ FROM INPUT FILE NUMBER NIFILE
C 1 --> FOLLOWING NUMBER IS REAL, DOUBLE PRECISION
C 2 --> FOLLOWING NUMBER IN FILE IS INTEGER
C 3 --> THIS IS THE END OF THE FILE
C ICR == INTEGER COLUMN OR ROW NUMBER READ FROM INPUT FILE NIFILE
C IC2 == INSTRUCTION CODE, NUMBER OF VALUES TO OUTPUT, 1 --> 4
C IC3 == INSTRUCTION CODE
C 1 --> THIS IS LAST LINE TO OUTPUT
C 2 --> DON'T STOP, MORE LINES WILL FOLLOW
C
C ESTABLISH POSSIBLE FORMATS
C
1 FORMAT('DMI ',3AS,2I8,24X,'ABC')
2 FORMAT(*BC ',4E16.8,A3)
3 FORMAT(*BC ',I16,3E16.8,A3)
4 FORMAT(*BC ',I16,2E16.8,A3)
5 FORMAT(*BC ',E16.8,2E16.8,A3)
6 FORMAT(*BC ',2E16.8,2I16,A3)
7 FORMAT(*BC ',I16,3E16.8)
8 FORMAT(*BC ',E16.8)
9 FORMAT(*BC ',3E16.8)
10 FORMAT(*BC ',E16.8)
11 FORMAT(*BC ',2I16)
12 FORMAT(*BC ',I16,3E16.8)
13 FORMAT(*BC ',I16,2E16.8)
14 FORMAT(*BC ',I16,2E16.8)
15 FORMAT(*BC ',2E16.8,2I16)
16 FORMAT(*BC ',2E16.8,2I16,E16.8)
17 FORMAT(*BC ',E16.8,2I16,E16.8)
18 FORMAT(*BC ',E16.8)
19 FORMAT(*BC ',I16,3E16.8)
20 FORMAT(*BC ',I16,2E16.8)
21 FORMAT(*BC ',I16,2E16.8)
22 FORMAT(*BC ',2I16,2E16.8)
23 FORMAT(*BC ',2I16,2E16.8)
24 FORMAT(*BC ',E16.8,2I16,E16.8)
25 FORMAT(*BC ',E16.8,2I16,E16.8)
26 FORMAT(*BC ',E16.8,2I16,E16.8)
27 FORMAT(*BC ',E16.8,2I16,E16.8)
28 FORMAT(*BC ',E16.8,2I16,E16.8)
29 FORMAT(*BC ',E16.8,2I16,E16.8)
30 FORMAT(*BC ',E16.8,2I16,E16.8)
31 FORMAT(*BC ',E16.8)
C
C WRITE FIRST LINE
C WRITE(NOFILE, I) MATRIX, PRECIS, FORM, NROWS, NCOLS
C
C STEP THROUGH NIFILE AND OUTPUT PER APPROPRIATE FORMAT
C 100 READ(8,30) INSTR(1)
110 IF(INSTR(1).EQ. 3) THEN
PRINT*, 'FIRST RECORD READ IN SUB ASTOUT IS END ***'
STOP
ENDIF
120 DD 200 I=1,4
130 IF(INSTR(1).EQ. 1) THEN

.
ELSEIF INSTR(1).EQ.2 THEN
READ(NOFILE,11) A(1)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 THEN
WRITE(NOFILE,10) A(1),A(2)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
WRITE(NOFILE,13) A(1),A(2),A(3)
WRITE(NOFILE,16) A(1),A(2),A(3)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,19) A(1),A(2),A(3)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,22) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,25) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,28) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,31) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,34) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,37) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,40) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,43) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,46) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,49) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,52) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,55) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,58) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,61) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,64) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,67) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,70) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,73) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,76) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,79) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,82) A(1),A(2),A(3),A(4)
GO TO 700
ELSEIF INSTR(1).EQ.3 THEN
IF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1 A ND.INSTR(3).EQ.1 THEN
WRITE(NOFILE,85) A(1),A(2),A(3),A(4)
GO TO 700
& INSTR(4).EQ.2) THEN
  WRITE(NOFILE,6) A(1),A(2),ICR(3),ICR(4),'ABC'
  INSTR(1)=INSTR(5)
  GO TO 150
ELSEIF(INSTR(1).EQ.1.AND.INSTR(2).EQ.1.AND.INSTR(3).EQ.1.AND.
& INSTR(4).EQ.2) THEN
  WRITE(NOFILE,7) A(1),A(2),A(3),ICR(4),'ABC'
  INSTR(1)=INSTR(6)
  GO TO 150
ENDIF
GO TO 603
ENDIF
ENDIF
C
C ERROR OUTPUT
C
600 PRINT*, 'ERROR WRITING LAST TWO ELEMENT LINE IN ASTOUT'
STOP
601 PRINT*, 'ERROR WRITING LAST THREE ELEMENT LINE IN ASTOUT'
STOP
602 PRINT*, 'ERROR WRITING LAST FOUR ELEMENT LINE IN ASTOUT'
STOP
603 PRINT*, 'ERROR WRITING INTERIM FOUR ELEMENT LINE IN ASTOUT'
STOP
C
C FINISHED
C
700 RETURN
END
6. Galati, Terry; Air Force Astronautics Laboratory. Personal Correspondence, August 1990.


Vita

Captain John R. Dewsnap was born on 10 August 1956 in North Bend, Oregon. He graduated from Deer Park High School in Deer Park, Washington in 1975. He attended Multnomah School of the Bible in Portland, Oregon and Spokane Falls Community College in Spokane, Washington.

In 1978 he enlisted in the USAF as an aircraft radio technician. He maintained and supervised the maintenance of communication systems on F-4, B-52 and KC-135 aircraft, while serving tours of duty at Torrejon AFB, Spain; Loring AFB, Maine; and Fairchild AFB, Washington. In 1983 he was awarded a scholarship under the Airman's Education and Commissioning Program and began studies in 1984 at the University of Washington in Seattle, Washington.

After graduation in 1986 with a Bachelor of Science in Aeronautics and Astronautics, Captain Dewsnap attended Officer Training School, receiving a regular commission in the USAF. He served his first commissioned tour of duty at the Ballistic Missile Office, Norton AFB, California. He was the Peacekeeper Stage II Project Manager, overseeing the technical and financial aspects of this multi-million dollar development program. He entered the School of Engineering, Air Force Institute of Technology in May, 1989.

Permanent address: 162 Apricot Lane
Dayton, Ohio 45433
A COMPARISON OF THE OPTIMIZATION AND ANALYSIS OF DOUBLY CURVED SHELLS USING MSC/NASTRAN AND ASTROS

John R. Dewnap, Captain, USAF

Air Force Institute of Technology, WPAFB OH 45433-6583

Dr. Venkayya
Structural Optimization Branch
Flight Dynamics Lab
Wright Patterson AFB, OH 45433

Approved for public release; distribution unlimited

This study identified techniques and software available for the optimization of doubly curved shells and applied them in the context of a large nozzle shape. An optimality criteria scheme that can reduce solution time was evaluated and compared to the Method of Feasible Directions. MSC/NASTRAN and ASTROS were used to perform finite element analysis and optimization, and the results were compared to theory. The programs give virtually identical results, and if plates and shells are carefully modeled, then stresses, displacements, and modes are accurate to within ten percent. A Mindlin-type axisymmetric finite element was implemented in ASTROS that preserved accuracy and reduced the size of the stiffness matrix by a factor of four. Nozzle optimization was performed using static pressure and thermal loads, constrained by the Von Mises stress criteria. Software errors were documented in ASTROS, and four characteristic stress regions identified for the optimized nozzle.

ASTROS, COMPUTER PROGRAMMING, FINITE ELEMENT ANALYSIS, MSC/NASTRAN, NONLINEAR PROGRAMMING, OPTIMIZATION, ROCKET NOZZLES, STRESS ANALYSIS, SUPERSONIC NOZZLES

Unclassified | Unclassified | Unclassified | UL