STRUCTURAL TAILORING OF ENGINE BLADES (STAEBL)

By:
C. E. Platt, T. K. Pratt and K. W. Brown

UNITED TECHNOLOGIES CORPORATION
POWER SECTOR
PRATT & WHITNEY AIRCRAFT GROUP
COMMERCIAL PRODUCTS DIVISION

Prepared for:
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

NASA Lewis Research Center
Contract NAS 3-22525

11-9-1982
A mathematical optimization procedure was developed for the Structural Tailoring of Engine Blades and was used to structurally tailor two engine fan blades constructed of composite materials without midspan shrouds. The first was a solid blade made from superhybrid composites, and the second was a hollow blade with metal-matrix composite inlays. Three major computerized functions were needed to complete the procedure: approximate analysis with the established input variables, optimization of an objective function, and refined analysis for design verification.
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>SUMMARY</td>
<td>1</td>
</tr>
<tr>
<td>2.0</td>
<td>INTRODUCTION</td>
<td>3</td>
</tr>
<tr>
<td>3.0</td>
<td>STAEBL PROCEDURE</td>
<td>5</td>
</tr>
<tr>
<td>3.1</td>
<td>Overview</td>
<td>5</td>
</tr>
<tr>
<td>3.2</td>
<td>General Approach to STAEBL Procedure Design</td>
<td>5</td>
</tr>
<tr>
<td>3.3</td>
<td>Input to Fan Blade Structural Tailoring Procedure</td>
<td>8</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Aerodynamic Stage</td>
<td>8</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Support Structure</td>
<td>8</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Operating Conditions</td>
<td>8</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Materials</td>
<td>9</td>
</tr>
<tr>
<td>3.3.5</td>
<td>Objective Function</td>
<td>9</td>
</tr>
<tr>
<td>3.3.6</td>
<td>Constraints</td>
<td>9</td>
</tr>
<tr>
<td>3.3.7</td>
<td>Design Variables</td>
<td>10</td>
</tr>
<tr>
<td>3.4</td>
<td>Approximate Analysis</td>
<td>10</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Stress and Vibration Analysis</td>
<td>10</td>
</tr>
<tr>
<td>3.4.1.1</td>
<td>Approach</td>
<td>10</td>
</tr>
<tr>
<td>3.4.1.2</td>
<td>Solution Procedure</td>
<td>17</td>
</tr>
<tr>
<td>3.4.1.3</td>
<td>Verification</td>
<td>23</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Foreign Object Damage Analysis</td>
<td>33</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Local Foreign Object Damage Analysis</td>
<td>34</td>
</tr>
<tr>
<td>3.4.4</td>
<td>Approximate Analysis System</td>
<td>35</td>
</tr>
<tr>
<td>3.5</td>
<td>Optimization Method</td>
<td>36</td>
</tr>
<tr>
<td>3.5.1</td>
<td>General Optimization Theory and Background</td>
<td>39</td>
</tr>
<tr>
<td>3.5.2</td>
<td>COPES/CONMIN Exact Analysis</td>
<td>40</td>
</tr>
<tr>
<td>3.5.2.1</td>
<td>Choice of Search Parameters for COPES/CONMIN</td>
<td>44</td>
</tr>
<tr>
<td>3.5.2.2</td>
<td>Scaling of Design Variables in COPES/CONMIN</td>
<td>45</td>
</tr>
<tr>
<td>3.5.2.3</td>
<td>Number of Function Calls for COPES/CONMIN</td>
<td>46</td>
</tr>
<tr>
<td>3.5.3</td>
<td>COPES/CONMIN Approximate Analysis</td>
<td>50</td>
</tr>
<tr>
<td>3.5.4</td>
<td>COPES/CONMIN Interfaces to Vibration, Flutter, and Stress Programs</td>
<td>53</td>
</tr>
<tr>
<td>3.5.5</td>
<td>Test Cases for STAEBL Optimization Procedure</td>
<td>57</td>
</tr>
<tr>
<td>3.6</td>
<td>Refined Analysis</td>
<td>61</td>
</tr>
<tr>
<td>4.0</td>
<td>RESULTS</td>
<td>61</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>61</td>
</tr>
<tr>
<td>4.2</td>
<td>Demonstration Considerations</td>
<td>61</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Aerodynamic Design</td>
<td>61</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Objective Function</td>
<td>61</td>
</tr>
<tr>
<td>4.2.2.1</td>
<td>Engine Weight</td>
<td>63</td>
</tr>
<tr>
<td>4.2.2.2</td>
<td>Engine Cost</td>
<td>65</td>
</tr>
<tr>
<td>4.2.2.3</td>
<td>Maintenance Cost</td>
<td>65</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Constraints on Behavior Variables</td>
<td>65</td>
</tr>
<tr>
<td>4.2.3.1</td>
<td>High Frequency Fatigue</td>
<td>65</td>
</tr>
<tr>
<td>4.2.3.2</td>
<td>Flight Cycle Fatigue</td>
<td>66</td>
</tr>
<tr>
<td>4.2.3.3</td>
<td>Bird Ingestion</td>
<td>67</td>
</tr>
<tr>
<td>4.3</td>
<td>STAEBL Procedure Demonstration</td>
<td>67</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Reference Blade</td>
<td>67</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Tailored Blades</td>
<td>69</td>
</tr>
<tr>
<td>4.3.2.1</td>
<td>Hollow Blade with Composite Inlay</td>
<td>69</td>
</tr>
<tr>
<td>4.3.2.2</td>
<td>Superhybrid Composite Blade</td>
<td>72</td>
</tr>
<tr>
<td>5.0</td>
<td>CONCLUSIONS AND RECOMMENDATIONS</td>
<td>77</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>BLADE OPTIMIZATION PROGRAM ORGANIZATION</td>
<td>79</td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>FORTRAN LISTING FOR HOLLOW BLADE TAILORING</td>
<td>85</td>
</tr>
<tr>
<td>APPENDIX C</td>
<td>FORTRAN LISTING REVISIONS FOR TAILORING SUPERHYBRID</td>
<td>329</td>
</tr>
<tr>
<td></td>
<td>BLADE</td>
<td></td>
</tr>
<tr>
<td>REFERENCES</td>
<td></td>
<td>339</td>
</tr>
<tr>
<td>DISTRIBUTION LIST</td>
<td></td>
<td>341</td>
</tr>
</tbody>
</table>
### LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2-1</td>
<td>The Structural Tailoring of Engine Blades Procedure</td>
<td>6</td>
</tr>
<tr>
<td>3.2-2</td>
<td>The Objective Function Relates Airline Economics to Blade Design Variables</td>
<td>6</td>
</tr>
<tr>
<td>3.2-3</td>
<td>Blade Chord Optimization Appears to be a Simple Design Problem</td>
<td>7</td>
</tr>
<tr>
<td>3.2-4</td>
<td>Design Problem Complexity is Introduced by Structural Constraints</td>
<td>7</td>
</tr>
<tr>
<td>3.4-1</td>
<td>Model Used in Approximate Analyses of a Spanwise Increment of Blade</td>
<td>12</td>
</tr>
<tr>
<td>3.4-2</td>
<td>Beam Model Verification with NASTRAN Vibration Analysis of Assembly of Rectangular, Isotropic, Cantilevered Beams</td>
<td>12</td>
</tr>
<tr>
<td>3.4-3</td>
<td>Torsional Frequencies Obtained with Planes Remain Plane Section Constraint</td>
<td>13</td>
</tr>
<tr>
<td>3.4-4</td>
<td>Low Torsional Frequencies Produced by the Free Warping Condition</td>
<td>14</td>
</tr>
<tr>
<td>3.4-5</td>
<td>Constrained Warping Condition and Thin Section Torsional Stiffness Correlated Well Using an Approximate Section Warping Function</td>
<td>15</td>
</tr>
<tr>
<td>3.4-6</td>
<td>Results Obtained for a Twisted Flat Plate</td>
<td>16</td>
</tr>
<tr>
<td>3.4-7</td>
<td>Radially Stacked Beam Element as Represented by the Laminated Composite Beam Theory</td>
<td>17</td>
</tr>
<tr>
<td>3.4-8</td>
<td>Shape Functions for the Element Displacement Response</td>
<td>18</td>
</tr>
<tr>
<td>3.4-9</td>
<td>Rectangular Beam Ply Layup</td>
<td>20</td>
</tr>
<tr>
<td>3.4-10</td>
<td>Tilted Flat Plate Used in Approximate Analysis Test Case</td>
<td>25</td>
</tr>
<tr>
<td>3.4-11</td>
<td>Twisted Flat Plate Used in Approximate Analysis Test Case</td>
<td>26</td>
</tr>
<tr>
<td>3.4-12</td>
<td>Airfoil Warping Defined by Spanwise Twist Gradient</td>
<td>29</td>
</tr>
<tr>
<td>3.4-13</td>
<td>Approximate Root Centrifugal Stress in the Energy Efficient Engine Fan Blade Compared with NASTRAN</td>
<td>30</td>
</tr>
<tr>
<td>3.4-14</td>
<td>NASTRAN Hollow Blade Model</td>
<td>31</td>
</tr>
<tr>
<td>Figure</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>3.4-15</td>
<td>Hollow Airfoil Beam</td>
<td>31</td>
</tr>
<tr>
<td>3.4-16</td>
<td>Laminated Airfoil Beam</td>
<td>32</td>
</tr>
<tr>
<td>3.5-1</td>
<td>Feasible Region Is Union of All Points that Satisfy All Constraints</td>
<td>37</td>
</tr>
<tr>
<td>3.5-2</td>
<td>Line Search Terminates Either at Minimum of Objective Function or at a Constant Boundary. Sequence of line searches converge to ( x_{opt} ).</td>
<td>40</td>
</tr>
<tr>
<td>3.5-3</td>
<td>New Search Direction, ( s_j ), Lies in the Usable Feasible Sector. The value of the push-off factor, ( \theta_j ), determines the orientation of the new search direction.</td>
<td>41</td>
</tr>
<tr>
<td>3.5-4</td>
<td>Constraint Thickness Parameter, ( CT ), Determines when a Constraint is Satisfied, Violated, or Active</td>
<td>43</td>
</tr>
<tr>
<td>3.5-5</td>
<td>For Proper Choice of ( CT ), Two Constraints Become Simultaneously Active so that Search Proceeds down the &quot;Valley&quot; Formed by the Constraints</td>
<td>43</td>
</tr>
<tr>
<td>3.5-6</td>
<td>Sequence of Approximate Analyses Using the Method of Feasible Directions Becomes Increasingly More Accurate as More Terms in the Taylor Series Are Determined.</td>
<td>48</td>
</tr>
<tr>
<td>3.5-7</td>
<td>COPES/CONMIN is Linked via Subroutine ANALIZ to Approximate Analyses</td>
<td>51</td>
</tr>
<tr>
<td>3.5-8</td>
<td>STAEBL Optimization Procedure May Not Converge to Global Optimum if Feasible Region is Disjoint</td>
<td>53</td>
</tr>
<tr>
<td>3.5-9</td>
<td>STAEBL Optimization Procedure with Three Flutter Constraints Converges to Global Optimum</td>
<td>54</td>
</tr>
<tr>
<td>3.5-10</td>
<td>Exact and Approximate Analyses for STAEBL Procedure Converge to Same Local Optimum</td>
<td>55</td>
</tr>
<tr>
<td>3.6-1</td>
<td>Refined Analysis of the Energy Efficient Engine Hollow Fan Blade</td>
<td>58</td>
</tr>
<tr>
<td>3.6-2</td>
<td>Energy Efficient Engine Hollow Fan Blade Airfoil Root Stress Predicted by Refined Analysis</td>
<td>59</td>
</tr>
<tr>
<td>3.6-3</td>
<td>Energy Efficient Engine Hollow Fan Blade Internal Surface Stress at Solid-to-Hollow Transition Predicted by Refined Analysis</td>
<td>59</td>
</tr>
<tr>
<td>Figure</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>3.6-4</td>
<td>Refined Analysis of the Energy Efficient Engine Hollow Fan Blade Supports the Conclusion that It Would Not Flutter</td>
<td>60</td>
</tr>
<tr>
<td>4.2-1</td>
<td>Fan System Weight Is a Simple Function of Individual Airfoil Weight</td>
<td>64</td>
</tr>
<tr>
<td>4.2-2</td>
<td>Full Span Hollow Fan Blade Life Is Derived from Solid Blade Experience Limits</td>
<td>66</td>
</tr>
<tr>
<td>4.3-1</td>
<td>Reference Blade Resonance Diagram</td>
<td>68</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4-I</td>
<td>NASTRAN TEST CASE FOR PANEL WITH 30 DEGREE PRETWIST</td>
<td>16</td>
</tr>
<tr>
<td>3.4-II</td>
<td>COMPARISON OF STATIC ANALYSIS PROCEDURE WITH NASTRAN TEST CASES</td>
<td>24</td>
</tr>
<tr>
<td>3.4-III</td>
<td>TILTED FLAT PLATE TIP DEFLECTIONS</td>
<td>25</td>
</tr>
<tr>
<td>3.4-IV</td>
<td>TWISTED (30 Degrees) FLAT PLATE TIP DEFLECTIONS</td>
<td>26</td>
</tr>
<tr>
<td>3.4-V</td>
<td>APPROXIMATE ANALYSIS NATURAL FREQUENCIES COMPARED WITH NASTRAN</td>
<td>27</td>
</tr>
<tr>
<td>3.4-VI</td>
<td>APPROXIMATE ANALYSIS NATURAL FREQUENCIES COMPARED WITH NASTRAN</td>
<td>27</td>
</tr>
<tr>
<td>3.4-VII</td>
<td>APPROXIMATE ANALYSIS NATURAL FREQUENCIES COMPARED WITH NASTRAN</td>
<td>27</td>
</tr>
<tr>
<td>3.4-VIII</td>
<td>APPROXIMATE PROCEDURE ANALYSIS OF FAN BLADE NATURAL FREQUENCY COMPARED WITH NASTRAN</td>
<td>28</td>
</tr>
<tr>
<td>3.4-IX</td>
<td>HOLLOW BLADE FREQUENCY COMPARISON</td>
<td>32</td>
</tr>
<tr>
<td>3.4-X</td>
<td>COMPOSITE REINFORCED BLADE FREQUENCY COMPARISON</td>
<td>32</td>
</tr>
<tr>
<td>3.4-XI</td>
<td>COMPOSITE REINFORCED BLADE FREQUENCY COMPARISON</td>
<td>33</td>
</tr>
<tr>
<td>3.4-XII</td>
<td>APPROXIMATE ANALYSIS SYSTEM MODULE OUTPUTS</td>
<td>36</td>
</tr>
<tr>
<td>3.5-I</td>
<td>UNSCALED AND SCALED VALUES OF GRADIENT OF OBJECTIVE FUNCTION RELATIVE TO THE DESIGN VARIABLES FOR HOLLOW FAN BLADE</td>
<td>45</td>
</tr>
<tr>
<td>3.5-II</td>
<td>FORTRAN NAMES AND GLOBAL LOCATIONS IN BLOBCM FOR DESIGN VARIABLES, OBJECTIVE FUNCTION, AND CONSTRAINTS FOR STAEBL BLADE APPLICATIONS</td>
<td>52</td>
</tr>
<tr>
<td>3.5-III</td>
<td>HOLLOW FAN BLADE HAS 3 ACTIVE CONSTRAINTS FOR FINAL DESIGN</td>
<td>56</td>
</tr>
<tr>
<td>4.2-I</td>
<td>ENERGY EFFICIENT ENGINE STUDY AIRPLANE CHARACTERISTICS</td>
<td>62</td>
</tr>
<tr>
<td>4.2-II</td>
<td>ENERGY EFFICIENT ENGINE AIRLINE ECONOMIC MODEL</td>
<td>63</td>
</tr>
<tr>
<td>4.3-I</td>
<td>HOLLOW BLADE WITH COMPOSITE INLAY</td>
<td>69</td>
</tr>
<tr>
<td>4.3-II</td>
<td>MATERIALS PROPERTIES</td>
<td>71</td>
</tr>
<tr>
<td>4.3-III</td>
<td>HOLLOW COMPOSITE BLADE WITH COMPOSITE INLAY NATURAL FREQUENCY</td>
<td>72</td>
</tr>
<tr>
<td>Table</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>4.3-IV</td>
<td>SUPERHYBRID COMPOSITE BLADE</td>
<td>73</td>
</tr>
<tr>
<td>4.3-V</td>
<td>SUPERHYBRID COMPOSITE BLADE NATURAL FREQUENCIES</td>
<td>75</td>
</tr>
<tr>
<td>A-I</td>
<td>APPROXIMATE ANALYSIS SUBROUTINE LIST</td>
<td>80</td>
</tr>
<tr>
<td>A-II</td>
<td>PRATT &amp; WHITNEY AIRCRAFT PROPRIETARY ROUTINES</td>
<td>84</td>
</tr>
</tbody>
</table>
1.0 SUMMARY

A program was conducted to develop an optimization procedure for the Structural Tailoring of Engine Blades (STAEBL) and to demonstrate the procedure by using it to design two fan blades of composite materials.

The fan stage of the Energy Efficient Engine which was designed under NASA Contract NAS3-20646, was selected as the aerodynamic configuration upon which to base the tailoring demonstration.

Three major computerized capabilities were needed to complete the procedure: approximate analysis with the established input variables and trial values of design variables, optimization of an objective function, and refined analysis for design verification. To perform approximate analyses, an existing procedure was modified to evaluate low cycle fatigue, vibratory fatigue and foreign object damage for composite blade designs. To incorporate the effects of Direct Operating Cost plus Interest (the objective function) into the design, the COPES/CONMIN optimization program was used. NASTRAN was used to verify that the approximate analysis and optimization procedure had designed a blade that met all design criteria.

To demonstrate the STAEBL procedure in real design situations, it was used to structurally tailor two engine fan blades constructed of composite materials without midspan shrouds. The first was a solid blade made from superhybrid composites, and the second was a hollow blade with metal-matrix composite inlays.

Mathematical optimization applied to shroudless fan blade structural tailoring has been demonstrated to be a very powerful automated design procedure. It provides the capacity to simultaneously evaluate many design variables to optimize a comprehensive objective function while satisfying numerous design constraints.
2.0 INTRODUCTION

Fan and compressor blades are designed to provide aerodynamic performance and structural durability at minimum cost to the aircraft/engine operator through alternating aerodynamic and structural design iterations. The current structural design procedure requires that specified criteria be satisfied. These criteria have been derived by correlating particular analyses with extensive empirical experience. The designer is provided with an interactive computer system that conducts vibration, steady state stress, and ingestion analyses of proposed designs, modifies designs for reanalysis, compares results of analysis with criteria and assembles input for non-interactive flight cycle life analysis. The structural designer uses his personal experience to establish the path to follow to improve the design and decide when to terminate the search for the best design within the limits of the proposed aerodynamic configuration.

Thus, the current design procedures for turbine engine blades are partly engineering and partly art. The quality of the design is often the result of the judgment and experience of the engineer or engineering team that performed the task. The penalties for less than optimum designs are weight and cost. The cost penalty may appear as low efficiency or the wasted time of a long development cycle, fixing failures and improving performance. This usually results in less than optimum designs because the constraints of correcting a problem are always more severe than an original design, i.e., the space for the stage is fixed, the performance of other components interacts or is dependent on the part under development, etc. Once the design has been corrected, it is usually at the expense of cost or weight and degradation of the overall engine performance must be accepted.

It is apparent that current blade design procedures are limited by the need for the design engineer to incorporate his experience in trading design variables against each other. This problem is not peculiar to structural blade design; it arises to substantially the same degree in the design of other components. For these reasons, it is appropriate to initiate development of automated procedures to permit the optimized trade-off of variables against each other to improve the blade design and establish a foundation for application to other components. Such formalized optimum design procedures have been developed and used with considerable success for optimum structural design of linear static structures, and are now being developed and used with some success for the aeroelastic tailoring of fixed aircraft wings. The objective of the Structural Tailoring of Engine Blades Program, hereinafter referred to as STAEBL, was to develop a formalized optimum design procedure for engine blades which will meet all the aerothermomechanical design requirements in an aircraft engine environment. The STAEBL procedure will reduce human error in the blade design process by automating with mathematical precision what was formerly user judgement on an interactive system.

To meet the objective of the STAEBL program, six technical tasks were established as part of NASA Contract NAS3-22525:
Task I: STAEBL Procedure - Design of the general STAEBL procedure.

Task II: Input - Definition of STAEBL procedure input parameters including initial blade geometry, material properties, loads, weight and cost models, and design constraints.

Task III: Approximate Analyses - Modification of existing beam analyses to perform vibration, stress and foreign object damage evaluations of composite blades.

Task IV: Optimization Procedure - Identify a procedure which optimizes the objective function, direct operating cost plus interest, within limits of specified constraints.

Task V: Refined Analyses - Establish a procedure for using NASTRAN to validate optimized blade designs.

Task VI: Demonstration and Documentation - Demonstrate and document the STAEBL procedure by using it to tailor two alternate designs of the shroudless Energy Efficient Engine fan blade: one a solid blade made from superhybrid composites; and the second, a hollow blade with metal-matrix composite inlays.

The facility used for the STAEBL program was an IBM System 370 computing system. Most engineering problems were currently programmed for solution on three existing IBM 370 computers. Using IBM's latest virtual storage technology, these computers could accommodate fully computerized interactive design systems, general time-sharing, teleprocessing, real time management/information systems, and management and scientific batch processing.

Section 3.0 of this report presents a description of the STAEBL program design. Section 4.0 describes the results of the demonstration of the procedure and Section 5.0 presents Conclusions and Recommendations emanating from this program.

Appendix A presents the STAEBL procedure organization, identifying the various subroutines used in the overall system. Appendix B presents a complete FORTRAN listing of the STAEBL procedure for the hollow blade. The revisions to this FORTRAN listing for the superhybrid blade are provided in Appendix C.
3.0 STAEBL PROCEDURE

3.1 OVERVIEW

Airfoil structural design is a critical part of the aircraft turbine engine development process. The limitations imposed by durability requirements for the airfoils have a direct bearing on the aerodynamic performance that can be achieved. In addition, a significant portion of engine weight and engine cost is a simple multiple of airfoil weight. The airfoil design problem is complex. Chord, thicknesses at several locations, and internal constructions are selected to simultaneously satisfy vibration, ingestion and flight cycle durability requirements. Mathematical optimization techniques have been developed to expedite solution of this kind of tailoring problem which involves many design variables and many requirements. The airfoil application is particularly appropriate because the complex shapes defined by optimization do not increase manufacturing cost. The basic airfoil aerodynamic shapes are fabricated in accordance with three-dimensional numerical definitions which are readily modified to accept the results of structural tailoring.

Problems associated with structural tailoring of engine blades include: 1) engine blades are designed to operate in a dynamic environment by application of constraints which differ substantially from those applied to linear static structures; 2) analysts and/or designers have hesitated to develop optimization procedures for blades made from homogeneous materials because acceptable designs can be derived from past experience; and 3) finite element analyses, which are too time consuming to be used effectively in an optimization procedure, have been used in designing blades having advanced constructions such as those to be designed in this program.

The approach taken to assemble a procedure which solves these problems is described in Section 3.2. Inputs to the procedure are defined in 3.3. The approximate analyses and controlling method used in automated optimization are described in Sections 3.4 and 3.5. The substantive refined analyses are described in Section 3.6.

3.2 GENERAL APPROACH TO STAEBL PROCEDURE DESIGN

Figure 3.2-1 summarizes the a procedure for the Structural Tailoring of Engine Blades. Design variables are initialized by input to the procedure and varied during optimization. Approximate analyses for low cycle fatigue, flutter, resonance, and foreign object damage are applied to evaluate position relative to constraints.

The objective function optimized in the STAEBL procedure is derived from the relationships illustrated in Figure 3.2-2. The complexity encountered in finding the design which optimizes this function can be illustrated by examining its relationship to blade chord (Figure 3.2-3). It appears to be simple, but becomes complicated when structural constraints are introduced (Figure 3.2-4). The design that the procedure selects must optimize user economics without violating the imposed constraints.
Determine gradients of objective function and constraints. Establish search directions, and determine designs iteratively until feasible design is established.

Evaluate constraint bounds. Calculate gradient of objective function and constraints, and determine feasible direction for search. Minimize objective function in feasible direction without violating constraints. Is this feasible design optimal? No

Modify constraints so that approximate analysis will produce fine tuned design. Does design meet all constraints? Yes

Evaluate optimal approximate design using refined analyses. Display structurally tailored blade. Stop.

Figure 3.2-1 The Structural Tailoring of Engine Blades Procedure

Blade design variables

Blades
- Initial cost
- Weight
- Maintenance cost

Airplane
- Takeoff gross weight
- Fuel burned

Engine
- Weight
- Costs

Direct operating cost plus interest

Figure 3.2-2 The Objective Function Relates Airline Economics to Blade Design Variables
Direct operating cost plus interest

Easily damaged blades

Large number of blades

Heavy blades

Large quantity of raw material

Figure 3.2-3 Blade Chord Optimization Appears to be a Simple Design Problem

Even integer number of blades

Foreign object damage

Forced response

Life cycle fatigue

Figure 3.2-4 Design Problem Complexity is Introduced by Structural Constraints
The most effective technique available for solving nonlinear optimization problems was selected from those available. The COPES/CONMIN (CONtrol Program for Engineering Synthesis/CONstrained MINimization) optimization program, a general purpose routine based on the method of feasible directions and developed by G. N. Vanderplaats of the Naval Postgraduate School, was chosen for the optimization procedure.

The above efforts in the STAEBL procedure identify a fine tuned optimum blade design that is validated by NASTRAN refined analysis. The procedure was demonstrated by the design of two composite material shroudless fan blades.

This procedure will reduce human error in the blade design process by automating with mathematical precision what was formerly user judgement in an interactive system.

3.3 INPUT TO FAN BLADE STRUCTURAL TAILORING PROCEDURE

3.3.1 Aerodynamic Stage

The starting point for structural tailoring of an engine blade is a candidate aerodynamic stage design which will deliver the required airflow and pressure ratio. The geometry of this candidate design is input to the structural tailoring procedure in the following form:

- coordinate definitions of a series of airfoil sections (define stagger, camber, edge radii, chord and thickness, all functions of radius);
- flowpath boundaries (root and tip radii and convergence angles);
- number of blades.

3.3.2 Support Structure

The dominant variables which control structural tailoring are frequency dependent and sensitive to blade attachment flexibility. Since the space available for the attachment varies with the airfoil design parameters, attachment flexibility is recognized by increasing the effective length of the candidate aerodynamic blade design. The additional input is:

- effective inner radius,
- dimensions of a rectangular section in the extended region.

3.3.3 Operating Conditions

Airfoil peak steady stress is calculated at maximum normal speed to determine life. Fatigue is prevented by tuning to avoid critical resonances at any speed above minimum cruise. Flutter stability and response to ingestion of a standard bird are calculated at maximum takeoff rotor speed. The inputs required to make these calculations are:
3.3.4 Materials

Blade centrifugal stresses and vibratory characteristics result from body loads and are, therefore, fully dependent upon the properties of the blade materials. Blade life is dependent on the strength of the material subjected to a particular stress condition. Composites materials, such as those to be used in the blades tailored in this program, are composed of a fixed proportion of fiber and matrix elements and can be considered to be homogeneous materials with directional properties. Similarly, adhesively bonded plies of metal matrix composite can be considered to be a single material. The net criticality of a local stress state is determined by evaluating a parameter which is a function of the relative criticality of each individual stress component. The inputs which define the required properties for each material are:

- density,
- directional moduli and Poisson's ratios,
- directional cyclic strengths.

3.3.5 Objective Function

The STAEBL procedure optimizes a single benefit which can be related to the final design. The benefit may be as simple as airfoil weight or it may be total value to the engine operator which considers trades between weight, initial cost, maintenance cost and even aerodynamic performance. The benefit expression is kept in generalized form by introducing a FORTRAN definition of:

- an objective function of design variables or quantities which are defined by the design variables (constant terms are not required).

3.3.6 Constraints

The durability objectives of a blade design are accomplished by imposing limits on the quantities that are calculated in the structural analyses. Margins are established relative to idealized limits to recognize the effects of geometric, material, and operational tolerances and to compensate for approximations in the analyses or underlying assumptions. Inputs to the STAEBL procedure are:

- minimum allowable predicted aerodynamic damping,
- minimum allowable difference between predicted frequencies and critical multiples of rotor speed,
maximum allowable local and root bird ingestion stress parameters,

- limits on design variables (for consistency with various assumptions).

3.3.7 Design Variables

Scaling techniques are provided within the STAEBL procedure to vary the coordinates that define any airfoil section in proportion with changes in chord or maximum thickness (fairing to constant edge radii). Logic is also included to identify the particular material at any point in a composite blade by references to quantities which define the relative position of the limits of that material. A fiber orientation angle is associated with each composite material. Relevant inputs are coded identification of design variables and initial values for starting the iteration and include:

- root chord (constant scale for all stations),
- thickness/chord (independent stations),
- composite material location limits (including the cavity as a zero properties composite),
- composite material fiber orientation angles.

3.4 APPROXIMATE ANALYSIS

The Pratt & Whitney Aircraft computerized system for designing conventional blade structures consists of a set of analysis programs which are linked to a common data library. Computational efficiency is an important consideration in the selection of these programs because the system is used so frequently. Approximate analyses are used wherever they have been successfully correlated with extensive durability experience. This system provides a proven base for selection of approximate analyses for the STAEBL system.

3.4.1 Stress and Vibration Analysis

3.4.1.1 Approach

The bladed disk vibration analysis program in the existing Pratt & Whitney Aircraft system evaluates a blade with coupled bending and torsional degrees of freedom mounted on a flexible disk and restrained by offset shrouds. The program can operate on blades with two or less shrouds and blade alone analyses can be performed. Resonance diagrams, mode shape plots and input data to the supersonic flutter prediction program are generated.

This beam-type analysis was formulated for application to airfoils with cross section shapes which are highly variable in the spanwise direction. It simulates increments of blade between independently defined airfoil sections. The analysis recognizes the effects of unequal deviation-from-radial of the directions of principal stiffness of the airfoil's leading and trailing edges.
Careful attention to edge stiffness inclination is given because the edge material is located at the greatest distance from the section shear center.

In this system, dynamic influence equations for a section of blade are derived by writing the six degree of freedom compatibility equations and six dynamic equilibrium equations of a rotating vibrating beam blade increment in finite difference form. These increments are then joined by a stacking matrix which rotates the variable vector into the local coordinate system of each successive blade increment, as defined by the line passing through the shear centers of the sections describing the ends of the increment. The shear centers are found by integration of the strength of materials thin section beam formula.

This analysis system was adapted to advanced blade constructions, including composite materials and hollow regions, by introducing new beam blade increment equilibrium and compatibility relationships. With these modifications, a spanwise blade increment is visualized as consisting of a stack of slices, Figure 3.4-1, where the stiffness properties of each slice are derived from lamination theory. Slice load resultants are determined by displacing one end of the beam relative to the other, evaluating slice strain levels, and multiplying by the local slice stiffness. Area integration of the load resultants provides the overall beam stiffness relationship.

The stack of slices analysis concept was verified with a NASTRAN analysis of an assembly of isotropic cantilevered beams of rectangular cross-section as shown in Figure 3.4-2. Figures 3.4-3 through 3.4-5 show the results of several comparisons with a NASTRAN baseline using a plate breakup. When the beam assembly was first analyzed, nodes at each cross-section were constrained using rigid body elements, forcing each plane to deform rigidly as shown in Figure 3.4-3. The results of this analysis were:

1. flapwise bending modes gave good agreement,
2. torsion mode frequencies were too high due to excessive section warping constraints,
3. stiffwise bending modes were too low due to shear flexibilities introduced by the guided cantilevered effect of the parallel beam components.

As shown in Figures 3.4-4 and 3.4-5, the torsion mode problem was alleviated by relaxing the warping rotation constraints along the cross-section. Stiffwise frequencies were improved by refining the spanwise sectional breakup. Figure 3.4-4 shows a no warping constraint condition and individual beam torsional stiffnesses which resulted in a torsional frequency that is too low.
Figure 3.4-1 Model Used in Approximate Analyses of a Spanwise Increment of Blade

Figure 3.4-2 Beam Model Verification with NASTRAN Vibration Analysis of Assembly of Rectangular, Isotropic, Cantilevered Beams
Figure 3.4-3 Torsional Frequencies Obtained with Planes Remain Plane Section Constraint
<table>
<thead>
<tr>
<th>CYCLES PER SECOND</th>
<th>FREE WAR Ping</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLATE</td>
<td></td>
</tr>
<tr>
<td>1ST F</td>
<td>90.1</td>
</tr>
<tr>
<td>1ST T</td>
<td>524.1</td>
</tr>
<tr>
<td>2ND F</td>
<td>568.3</td>
</tr>
<tr>
<td>1ST S</td>
<td>821.7</td>
</tr>
</tbody>
</table>

*THE SUM OF BEAM TORSIONAL STIFFNESS DOES NOT EQUAL SECTION TORSIONAL STIFFNESS*

Figure 3.4-4  Low Torsional Frequencies Produced by the Free Warping Condition
The low torsional stiffnesses were resolved by introducing thin section torsional stiffness coefficients and an approximate warping function. The warping function was evaluated using:

$$u = -yz \left( \frac{d\theta}{dx} \right)$$  \hspace{1cm} (1)

This warping function corresponds to the first term of the elasticity solution for the torsion of a rectangular section. The effect on natural frequencies of imposing the warping function on the NASTRAN solution is shown on Figure 3.4-5. As a result of the warping function prescription, each cross-section is now reduced to six degrees of freedom, which is consistent with the existing beam analysis procedure. Since the warping function that was prescribed corresponds rather well with the elasticity solution, little system stiffness was added, and the torsion mode frequency was increased by only 2.5 percent.

The beam assembly procedure was then tested on a plate with a 30 degree pretwist between root and tip. Results of this test case are shown on Table 3.4-1 and Figure 3.4-6. All frequencies from the combined beam analysis procedure were found to be within 10 percent of the NASTRAN plate solution.

These test cases verify the concept that a beam section may be assembled from a collection of beam elements with acceptable results for bending and torsion frequencies.
TABLE 3.4-I

NASTRAN TEST CASE FOR PANEL WITH 30 DEGREE PRETWIST
(Cycles per Second)

<table>
<thead>
<tr>
<th></th>
<th>NASTRAN Plate Analysis</th>
<th>Connected Beams, Free Warping</th>
<th>Connected beams, Warping Imposed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Flap</td>
<td>90.1</td>
<td>87.6</td>
<td>87.6</td>
</tr>
<tr>
<td>2nd Flap</td>
<td>480.1</td>
<td>438.8</td>
<td>438.8</td>
</tr>
<tr>
<td>1st Torsion</td>
<td>535.1</td>
<td>567.9</td>
<td>585.5</td>
</tr>
</tbody>
</table>

* WARPING PRESCRIBED, $\phi = -yz \theta, x$

* TORSIONAL STIFFNESS, $K_T = 1/3 BT^3$

COEFFICIENT

![Figure 3.4-6 Results Obtained for a Twisted Flat Plate](image)

Figure 3.4-6 Results Obtained for a Twisted Flat Plate
3.4.1.2 Solution Procedure

The approximate modelling technique for the design of a composite fan blade uses a laminated composite beam theory. In this theory, the airfoil is divided into radial segments with each radial segment being further divided into a chordwise array of beams as shown in Figure 3.4-7. Each beam has a rectangular cross-section to simplify the composite material characterization. The laminated beam derivation parallels classical laminated plate equations. Differences in stress components and the definition of the stress and moment resultants occur between beam and plate equations.

Figure 3.4-7 Radially Stacked Beam Element as Represented by the Laminated Composite Beam Theory
The laminated beam formulation employs a direct stiffness finite element approach. Slice stiffnesses are summed to give the section stiffness. The section stiffness matrix is then transformed into transfer matrix form, so that it may now replace the influence coefficients of the original analysis system.

The laminated beam formulation begins with the assumption of cubic lateral displacement functions and linear membrane and twist displacement functions, as shown in Figure 3.4-8. The \( o \) subscripts in Figure 3.4-8 refer to neutral axis displacements.

\[
\begin{align*}
V_0 &= a_1 + a_2 x + a_3 x^2 + a_4 x^3 \\
W_0 &= a_5 + a_6 x + a_7 x^2 + a_8 x^3 \\
U_0 &= a_9 + a_{10} x \\
\theta_0 &= a_{11} + a_{12} x
\end{align*}
\]

Figure 3.4-8 Shape Functions for the Element Displacement Response

The coefficients of the displacement functions, \( a \), are solved for in terms of nodal point displacements, giving:

\[
\{a\} = [\bar{N}] \{\Delta\}.
\]

The element shape functions, \([N]\), may, therefore, be readily evaluated.

Strains may be evaluated from shape function derivatives: second derivatives in bending, first derivatives in membrane and twist. Thus, the neutral axis strains and curvatures may be related to the shape coefficients, as

\[
\{\xi_0\} = [X] \{a\}.
\]

Away from the neutral axis, with the assumption that planes remain plane in bending, displacements become:

\[
\begin{align*}
\{u\}(x,y,z) &= u_0(x) - y v_{0,x}(x) - z w_{0,x}(x) + C y z \theta ,x \\
\{v\}(x,y,z) &= v_0(x) - c_1 y u_{0,x}(x) - x z \theta ,x \\
\{w\}(x,y,z) &= w_0(x) - c_2 z u_{0,x}(x) - x y \theta ,x
\end{align*}
\]

where \( C \) is a warping function coefficient, and \( c_1 \) and \( c_2 \) are Poisson ratios. Differentiation of these displacements gives the necessary strain relationships, which may now be related to nodal point displacements.
For the k'th lamina, the rotated stress-strain relation is

\[
\begin{bmatrix}
\sigma_x \\
\sigma_{xy} \\
\sigma_{xz}
\end{bmatrix}_k =
\begin{bmatrix}
Q_{11} & 0 & Q_{13} \\
0 & Q_{22} & 0 \\
Q_{13} & 0 & Q_{33}
\end{bmatrix}
\begin{bmatrix}
e_x \\
g_{xy} \\
g_{xz}
\end{bmatrix}_k
\]  

(5)

Integration of the stresses over the laminate gives the beam stress resultants,

\[
\begin{bmatrix}
N_x \\
N_{xy} \\
N_{xz}
\end{bmatrix} = \sum_{K=1}^{N} \int_{A_K} \begin{bmatrix}
\sigma_x \\
\sigma_{xy} \\
\sigma_{xz}
\end{bmatrix}_K \, dA_K
\]

(6)

where \( N \) is the total number of lamina in the laminate.

Substituting the lamina stress-strain law and the strain-displacement relations into the above, and integrating over the rectangular cross-section gives:

\[
\begin{bmatrix}
N_x \\
M_x \\
M_y \\
M_z
\end{bmatrix} = \sum_{K=1}^{N} \int_{A_K} \begin{bmatrix}
y \sigma_{xz} - z \sigma_{xy} \\
-y \sigma_x \\
-z \sigma_x
\end{bmatrix} \, dA_K
\]

(7)

where:

\[D_{33} = 2D_{33} + \frac{b^2}{6} A_{22}\]

\[A_{i,j} = a \sum_{K=1}^{N} Q_{i,j_K} (Y_K - Y_{K-1})\]

(8)

\[B_{i,j} = a/2 \sum_{K=1}^{N} Q_{i,j_K} (Y_K^2 - Y_{K-1}^2)\]

and
with the rectangle dimensions and layup as illustrated in Figure 3.4-9.

\[ D_{1j} = \frac{a}{3} \sum_{K=1}^{N} Q_{1jK} \left( Y_K^3 - Y_{K-1}^3 \right) \]

Figure 3.4-9  Rectangular Beam Ply Layup

The element force resultants at the beam ends may now be expressed in terms of the neutral axis strains as:

\[ \{f\} = [E] \{\varepsilon_0\} \]  \hspace{1cm} (9)

The beam forces may be transferred into the finite element coordinate system, using an equilibrium matrix, giving:

\[ \{F\} = [A] \{f\} \]  \hspace{1cm} (10)

The previous relations may now be combined to yield the element stiffness relation:

\[ \{F\} = [A] [E] [X] [\bar{N}] \{\Delta\} = [K_B] \{\Delta\} \]  \hspace{1cm} (11)
The section stiffness may be generated as the summation of the individual stacked beam stiffnesses, when offsets and cross-section warping are accounted for through a rigid-body linkage:

\[
\{F_s\} = \sum_{i=1}^{M} \left( [R^T_i][K_B][R]_i \right) \{\Delta_s\} = [S_K] \{\Delta_s\} \quad (12)
\]

where there are \( M \) stacked beams on the section. At the blade root, the warping function is set to zero, to impose warping restraint effects on the blade model.

Once the section stiffness has been generated, to make the new element compatible in form with the existing beam blade analysis, the section stiffness must be transformed to transfer matrix form. Partitioning the element equilibrium equation for ends 1 and 2 gives:

\[
\begin{bmatrix}
F_1 \\
F_2
\end{bmatrix} =
\begin{bmatrix}
S_{K11} & S_{K12} \\
S_{K21} & S_{K22}
\end{bmatrix}
\begin{bmatrix}
\Delta_1 \\
\Delta_2
\end{bmatrix} \quad (13)
\]

Reordered to transfer matrix form, the element equation becomes:

\[
\begin{bmatrix}
\Delta_2 \\
\Delta_2
\end{bmatrix} =
\begin{bmatrix}
-S_{K12}^{-1}S_{K11} & -S_{K12}^{-1} \\
S_{K21} - S_{K22}S_{K12}^{-1}S_{K11} & S_{K22}
\end{bmatrix}
\begin{bmatrix}
\Delta_1 \\
\Delta_1
\end{bmatrix} = [S_K] \begin{bmatrix}
\Delta_1 \\
\Delta_1
\end{bmatrix} \quad (14)
\]

The section masses are treated as lumped mass points at the two end node locations. In blade vibration, equilibrium across a concentrated mass point, gives:

\[
F_2' = F_2 + m\ddot{U}_2 = F_2 - w^2mU_2. \quad (15)
\]

In matrix form, the mass effect may be expressed as:

\[
\begin{bmatrix}
U \\
F_{2'}
\end{bmatrix} =
\begin{bmatrix} 1 & 0 \\
-w^2m & 1
\end{bmatrix}
\begin{bmatrix}
U \\
F
\end{bmatrix} = [m]
\begin{bmatrix}
U \\
F
\end{bmatrix} \quad (16)
\]

Hence, in traversing from the beginning of a beam segment to its end, with inertia effects included, it is found:
The product above is the final step in making the present theory compatible with the existing beam analysis. Notably, only the array has terms which are frequency dependent. Hence, this procedure can be made very efficient for frequency extraction.

For a rotating blade, centrifugal accelerations tend to stiffen the blade, and must be included in the calculation for vibration frequencies. In the present analysis, centrifugal effects are included through a section differential stiffness matrix. The section differential stiffness matrix employed, found in Reference 1, utilizes the element load resultants and geometry to calculate the stiffness increment.

To determine the static load resultants, and also for the calculation of static stresses, a static analysis is performed.

A lumped mass representation is employed in the static analysis as done previously for vibration analysis. At station \( i+1 \), a static load, \( P_{i+1} \), results from the centrifugal accelerations of the point mass. Thus, with mass effects included:

\[
\begin{bmatrix} U_{i+1} \\ F_{i+1} \end{bmatrix} = \begin{bmatrix} K_k \end{bmatrix} \begin{bmatrix} U_i \\ F_i \end{bmatrix} + \begin{bmatrix} 0 \\ P_{i+1} \end{bmatrix}
\]  

For the full blade, from station 1 to station \( n \),

\[
\begin{bmatrix} U_n \\ F_n \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} \\ K_{12} & K_{22} \end{bmatrix} \begin{bmatrix} U_1 \\ F_1 \end{bmatrix} + \begin{bmatrix} 0 \\ P_T \end{bmatrix}
\]  

At station 1, the blade root boundary condition,

\[
U_1 = C F_1,
\]

allows the root to be spring supported or cantilevered. The final equation system thus becomes:

\[
\begin{bmatrix} U_n \\ F_n \end{bmatrix} = \begin{bmatrix} C K_{11} + K_{12} \\ C K_{21} + K_{22} \end{bmatrix} \begin{bmatrix} F_1 \\ P_{T1} \end{bmatrix} + \begin{bmatrix} P_{T1} \\ P_{T2} \end{bmatrix}
\]  

At station \( n \), the blade tip boundary condition:

\[
\{ F_n \}' = 0.
\]
The root load can then be determined,
\[
\{F_1\} = -[C \ K_{21} + K_{22}]^{-1} \{P_{T2}\}
\]
(23)

Now, knowing the root loads and deflections, section loads and deflections are calculated according to the recursion relationship of (18).

The components of the centrifugal force acting on a vibrating blade contain terms which are proportional to the displacement. These displacement dependent terms form the "centrifugal mass matrix" of Reference 2. The "centrifugal mass matrix" for the present analysis has been generated at the individual beam level. The application of appropriate planar constraints enables the reduction of the centrifugal mass to the six degree-of-freedom section level. This capability has also been included in the analysis.

3.4.1.3 Verification

Due to intentional similarities between the present stacked beam analysis and the NASTRAN Bar element, analysis verification was greatly simplified.

The approximate static analysis procedure was verified through comparisons with NASTRAN test cases for a flat plate and for a plate with a 30 degree twist. The model consisted of 8 cross-sections, with 11 beams per section. Table 3.4-II summarizes the results of the two test cases, showing nearly exact agreement with NASTRAN for both deflections and reaction loads.

The differential stiffness and centrifugal mass capabilities of the approximate analysis were demonstrated by comparing the results of two test cases with the results of NASTRAN analyses. In the tailoring operation, these restoration effects would only be applied to the vibration analysis but the verification was obtained from iterated static analyses.

The first case, shown in Figure 3.4-10, consisted of a rotating, tilted, flat plate. Table 3.4-III shows almost exact comparisons between the present analysis and a corresponding NASTRAN beam analysis for the local tip deflections.

The second test case, shown on Figure 3.4-11, consisted of a twisted flat plate stacked along a radial line, with a 30 degree twist from root to tip. As shown on Table 3.4-IV, almost exact agreement exists between the present analysis and a corresponding NASTRAN beam analysis.

The vibration analysis capability of the stacked beam solution was verified by comparing predicted natural frequencies with NASTRAN predicted natural frequencies. Table 3.4-V shows excellent agreement between the approximate analysis procedure and a NASTRAN plate model for analysis of a flat plate 30.5cm x 10.2cm x 1.02cm (12in x 4in x 0.4in).

As shown in Table 3.4-VI, when the plate was given a 30 degree twist between root and tip, bending frequency agreement remained good between the approximate analysis and NASTRAN. The first torsion mode frequencies differed by less than 5 percent.
To examine a cross-section more representative of an airfoil, a doubly wedged plate was analyzed. For this model, the edge thicknesses were taken as one quarter of the mid-chord thickness. Good agreement with NASTRAN was obtained, as seen on Table 3.4-VII.

### TABLE 3.4-II

**COMPARISON OF STATIC ANALYSIS PROCEDURE WITH NASTRAN TEST CASES**

<table>
<thead>
<tr>
<th>NASTRAN</th>
<th>Static Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Flat Plate (11 Beams/Section-8 Sections)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Tip Deflections</strong></td>
<td></td>
</tr>
<tr>
<td>Radial, cm (in)</td>
<td>0.4171-3 (0.1642-3)</td>
</tr>
<tr>
<td>Twist, (radians)</td>
<td>0</td>
</tr>
<tr>
<td><strong>Restraint Forces</strong></td>
<td></td>
</tr>
<tr>
<td>Radial, n (lb)</td>
<td>-2327. (-0.5232+3)</td>
</tr>
<tr>
<td>Twist, n-m (in-lb)</td>
<td>0</td>
</tr>
<tr>
<td><strong>Twisted Plate (11x8 Breakup, 30° twist)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Tip Deflections</strong></td>
<td></td>
</tr>
<tr>
<td>Radial, cm (in)</td>
<td>0.4244-3 (0.1671-3)</td>
</tr>
<tr>
<td>Twist, (radians)</td>
<td>0.3828-4</td>
</tr>
<tr>
<td><strong>Restraint Forces</strong></td>
<td></td>
</tr>
<tr>
<td>Radial, n (lb)</td>
<td>-2330. (-0.5239+3)</td>
</tr>
<tr>
<td>Twist, n-m (in-lb)</td>
<td>3.11 (0.2754+2)</td>
</tr>
</tbody>
</table>
Figure 3.4-10  Tilted Flat Plate Used in Approximate Analysis Test Case

### TABLE 3.4-III

<table>
<thead>
<tr>
<th>Local Tip Deflections</th>
<th>$T_1$, cm (in)</th>
<th>$T_2$, cm (in)</th>
<th>$R_3$, radians</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Static Analysis</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NASTRAN</td>
<td>0.04881</td>
<td>1.9461</td>
<td>0.08524</td>
</tr>
<tr>
<td></td>
<td>(0.019217)</td>
<td>(0.76617)</td>
<td></td>
</tr>
<tr>
<td>STAEBL</td>
<td>0.04890</td>
<td>1.9463</td>
<td>0.08526</td>
</tr>
<tr>
<td></td>
<td>(0.019251)</td>
<td>(0.76627)</td>
<td></td>
</tr>
<tr>
<td><strong>Iterated Static Analysis</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NASTRAN</td>
<td>0.04870</td>
<td>0.88321</td>
<td>0.03474</td>
</tr>
<tr>
<td></td>
<td>(0.019175)</td>
<td>(0.34772)</td>
<td></td>
</tr>
<tr>
<td>STAEBL</td>
<td>0.04875</td>
<td>0.88326</td>
<td>0.34474</td>
</tr>
<tr>
<td></td>
<td>(0.019192)</td>
<td>(0.34774)</td>
<td></td>
</tr>
</tbody>
</table>
Figure 3.4-11 Twisted Flat Plate Used in Approximate Analysis Test Case

<table>
<thead>
<tr>
<th>Local Tip Deflections</th>
<th>$T_1$, cm (in) $\times 10^{-3}$</th>
<th>$R_1$ (radians) $\times 10^{-4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Static Analysis</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NASTRAN</td>
<td>0.4246 (0.16715)</td>
<td>0.38276</td>
</tr>
<tr>
<td>STAEBL</td>
<td>0.4245 (0.16711)</td>
<td>0.38272</td>
</tr>
<tr>
<td><strong>Iterated Static Analysis</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NASTRAN</td>
<td>0.4246 (0.16715)</td>
<td>0.38261</td>
</tr>
<tr>
<td>STAEBL</td>
<td>0.4245 (0.16711)</td>
<td>0.38264</td>
</tr>
</tbody>
</table>
### TABLE 3.4-V

**APPROXIMATE ANALYSIS NATURAL FREQUENCIES COMPARED WITH NASTRAN**  
*(Flat Plate)*

<table>
<thead>
<tr>
<th>Mode</th>
<th>Cycles per second</th>
<th>Percent Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>NASTRAN</td>
<td>STAEBL</td>
<td></td>
</tr>
<tr>
<td>First flap</td>
<td>90</td>
<td>88.5</td>
</tr>
<tr>
<td>First torsion</td>
<td>524</td>
<td>529.7</td>
</tr>
<tr>
<td>Second flap</td>
<td>558</td>
<td>552.3</td>
</tr>
<tr>
<td>First stiff</td>
<td>822</td>
<td>822.5</td>
</tr>
</tbody>
</table>

### TABLE 3.4-VI

**APPROXIMATE ANALYSIS NATURAL FREQUENCIES COMPARED WITH NASTRAN**  
*(30 Degree Twisted Plate)*

<table>
<thead>
<tr>
<th>Mode</th>
<th>Cycles per second</th>
<th>Percent Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>NASTRAN</td>
<td>STAEBL</td>
<td></td>
</tr>
<tr>
<td>First flap</td>
<td>90</td>
<td>88.5</td>
</tr>
<tr>
<td>Second flap</td>
<td>480</td>
<td>474.0</td>
</tr>
<tr>
<td>First torsion</td>
<td>536</td>
<td>561.5</td>
</tr>
<tr>
<td>First stiff</td>
<td>941</td>
<td>945.0</td>
</tr>
</tbody>
</table>

### TABLE 3.4-VII

**APPROXIMATE ANALYSIS NATURAL FREQUENCIES COMPARED WITH NASTRAN**  
*(Double-Wedge Plate)*

<table>
<thead>
<tr>
<th>Mode</th>
<th>Cycles per second</th>
<th>Percent Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>NASTRAN</td>
<td>STAEBL</td>
<td></td>
</tr>
<tr>
<td>First flap</td>
<td>65.6</td>
<td>65.0</td>
</tr>
<tr>
<td>Second flap</td>
<td>404.8</td>
<td>405.5</td>
</tr>
<tr>
<td>First torsion</td>
<td>442.5</td>
<td>460.0</td>
</tr>
<tr>
<td>First stiff</td>
<td>706.4</td>
<td>705.5</td>
</tr>
</tbody>
</table>
Further calibration of the analysis system was obtained by comparing the approximate analysis procedure with NASTRAN for frequency predictions of a rotating fan blade. The comparison, shown in Table 3.4-VIII, gave agreement consistent with the anticipated accuracy of an approximate analysis.

**TABLE 3.4-VIII**

APPROXIMATE PROCEDURE ANALYSIS OF FAN BLADE NATURAL FREQUENCY COMPARED WITH NASTRAN

<table>
<thead>
<tr>
<th>Mode</th>
<th>Cycles per second</th>
<th>Percent Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NASTRAN</td>
<td>STAEBL</td>
</tr>
<tr>
<td>First flap</td>
<td>92.9</td>
<td>93.6</td>
</tr>
<tr>
<td>First torsion</td>
<td>185.3</td>
<td>163.2</td>
</tr>
<tr>
<td>Second flap</td>
<td>210.2</td>
<td>205.7</td>
</tr>
<tr>
<td>First stiff</td>
<td>370.9</td>
<td>349.0</td>
</tr>
</tbody>
</table>

When stress distribution predictions were compared between the approximate analysis and previous beam analysis, poor correlations were noted. The problem was identified to be an assumption that warping faired uniformly from zero at the airfoil root to free-beam, as defined by the equations presented in Figure 3.4-12, at the end of the first increment. This assumption failed to recognize the fact that high root stress is a local result of restrained warping. A modification changed the torsional stiffness to recognize that:

\[
\text{twist gradient } \alpha = 1 - e^{-x/A}
\]

\[
\frac{\text{twist moment}}{x = \text{distance from root}}
\]

\[
A = \text{characteristic length associated with warping restraint}
\]

The resulting root centrifugal stress distribution is shown in Figure 3.4-13. Agreement with NASTRAN plate element blade analysis is as good as can be expected of a beam analysis.

An additional processing step was added to the approximate static and vibratory stress calculations in anticipation of the use of composite material in some layers of a tailored blade. A single quantity incorporating all stress components is evaluated. It is \( F_1 \sigma_1 + F_2 \sigma_2 + F_6 \sigma_6 + F_{11} \sigma_{11}^2 + F_{22} \sigma_{22}^2 + F_{66} \sigma_{66}^2 + 2 F_{12} \sigma_1 \sigma_2 \) where \( \sigma_1 \) and \( \sigma_2 \) are the principal tensile (or compressive) stresses and \( \sigma_6 \) is the shear stress. \( F_i \) and \( F_{ij} \) are input directional materials properties which are defined to yield an evaluation of the Tsai-Wu failure criteria for a composite material (or octahedral shear stress to strength ratio squared in metal).
The accuracy of the natural frequency prediction in the STAEBL procedure was subjected to an additional check by applying it to the analysis of a hollow blade. The same blade geometry, shown in Figure 3.4-14, was analyzed using NASTRAN plate elements with properties defined by lamination theory. Bending frequencies were in agreement but the error in the torsional frequency prediction was excessive. A similar result was obtained from analysis of a hollow rectangular section beam. It was found that the torsional stiffness equation could be redefined to be in agreement with the formula for a hollow rectangular section beam without changing the successful results that had previously been obtained for a solid airfoil.

The revised stiffness expression, for the thin walled airfoil pictured in Figure 3.4-15 and 3.4-16, is:

\[
K = 4G \int_{-t/2}^{b-t/2+\delta} \int_{-t/2}^{b/2} y^2 dydz + 4G \int_{0}^{t/2-\delta} \int_{0}^{b} y^2 dydz \tag{25}
\]

which agrees with the hollow beam stiffness when thickness, \( t \), approaches a constant. The revised stiffness expression also solved the airfoil problem as is illustrated by the good agreement presented in Table 3.4-IX.

- RADIAL DEFLECTION, \( u \)

\[
\bar{W}_s = \frac{1}{\text{AREA}} \int_{0}^{S} t W_s ds
\]

\[
W_s = \int_{0}^{S} R_s ds
\]

\( \Theta = \text{TWIST GRADIENT} \)

- ROTATIONS, \( R_2, R_3 \)

\[
R_2 = -R_N \Theta
\]

\[
R_3 = -R_T \Theta
\]

Figure 3.4-12 Airfoil Warping Defined by Spanwise Twist Gradient
Figure 3.4-13 Approximate Root Centrifugal Stress in the Energy Efficient Engine Fan Blade Compared with NASTRAN
Figure 3.4-14 NASTRAN Hollow Blade Model

Figure 3.4-15 Hollow Airfoil Beam
TABLE 3.4-IX
HOLLOW BLADE FREQUENCY COMPARISON

<table>
<thead>
<tr>
<th></th>
<th>NASTRAN (cps)</th>
<th>Approximate Analysis (cps)</th>
<th>Percent Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Frequency</td>
<td>110.7</td>
<td>113.5</td>
<td>+2.5</td>
</tr>
<tr>
<td>Second Frequency</td>
<td>267.4</td>
<td>278.8</td>
<td>+4.3</td>
</tr>
<tr>
<td>First Torsion</td>
<td>289.8</td>
<td>295.8</td>
<td>+2.1</td>
</tr>
</tbody>
</table>

A final check of the approximate analyses was accomplished by conducting comparative NASTRAN and approximate airfoil frequency analyses with composite material layers added. The results are presented in Tables 3.4-X and 3.4-XI. The trends are correct and the agreement is within the limits that can be expected of approximate analysis.

TABLE 3.4-X
COMPOSITE REINFORCED BLADE FREQUENCY COMPARISON
(0° Fiber Orientation)

<table>
<thead>
<tr>
<th>Mode</th>
<th>NASTRAN (cps)</th>
<th>Approximate Analysis (cps)</th>
<th>Percent Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Bending</td>
<td>108.4</td>
<td>111.8</td>
<td>3.1</td>
</tr>
<tr>
<td>Second Bending</td>
<td>261.8</td>
<td>276.2</td>
<td>5.5</td>
</tr>
<tr>
<td>First Torsion</td>
<td>298.8</td>
<td>312.9</td>
<td>4.7</td>
</tr>
</tbody>
</table>
TABLE 3.4-XI
COMPOSITE REINFORCED BLADE FREQUENCY COMPARISON
(35° Fiber Orientation)

<table>
<thead>
<tr>
<th>Mode</th>
<th>NASTRAN (cps)</th>
<th>Approximate Analysis (cps)</th>
<th>Percent Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Bending</td>
<td>108.4</td>
<td>111.1</td>
<td>2.5</td>
</tr>
<tr>
<td>Second Bending</td>
<td>259.6</td>
<td>270.0</td>
<td>4.0</td>
</tr>
<tr>
<td>First Torsion</td>
<td>295.4</td>
<td>298.8</td>
<td>1.2</td>
</tr>
</tbody>
</table>

3.4.2 Foreign Object Damage Analysis

Bird ingestion is a nonlinear transient structural dynamics problem with fluid structure interaction. The blade can fail in the impacted region when local strain exceeds material ductility, or away from the impacted region, when a few cycles of post-impact strain oscillation substantially exceed material strength. Analyses which simulate this behavior have been developed, but high computation time prohibits incorporation in the current design system. Simple chordwise and spanwise bending stress parameters are included in the current Pratt & Whitney Aircraft design system. They represent the effects of moment distributions resulting from a bird slice ingestion at the blade tip. They have been correlated with titanium blade ingestion experience.

3.4.2.1 Spanwise Bending Damage

A more refined linear analysis of spanwise bending response was derived by examination of an impact event where the differential equation of motion on the blade is:

$$[m] \ddot{x} + [k] x = \{F(t)\}$$  \hspace{1cm} (26)

Considering the response to be a superposition of n natural modes of the blade, the equilibrium equation may be decoupled into n modal equations of the form:

$$\ddot{\xi}_k + \omega_k^2 \xi_k = \frac{1}{m_k} P_k(t)$$  \hspace{1cm} (27)

where $\xi_k(t)$ is the modal amplification factor of the k'th mode, $\omega_k$ is the natural frequency, $m_k$ is the modal generalized mass, and $P_k = \lambda_k \{F(t)\}$ is the modal forcing function.

Assuming that the time of load application is short relative to the time when the root stress is most critical, (27) may be solved, giving:

$$\xi_k(t) = \frac{I_k}{m_k \omega_k} \sin(\omega_k t)$$  \hspace{1cm} (28)

where $I_k$ is the modal impulse.
Equation (28) provides the means for evaluating the blade root stresses as a function of time. Experience has shown that the highest root stresses occur at the quarter cycle of the first bending time point.

3.4.2.2 Local Damage Analysis

The current design system local impact damage looks at chordwise bending stress at the radius of the center of impact.

The analytical model consists of a blade cross section of incremental span. The loading is distributed over a distance determined by blade to gap and the velocity of the bird relative to the blade. The applied load is assumed to be taken out by the transverse shear and torsional restraint provided by the blade cross sections directly above and below the impacted region. Based on these considerations, the bending moment variation along the chord can be calculated and, since bending stress is inversely proportional to the square of the thickness, the stress parameter is moment/thickness$^2$. It peaks at a part chord location.

Scaling to determine an allowable for composite structures has been accomplished based upon the plastic hinge moment capability of the airfoil section.

For a titanium cross-section of thickness, $h$, the plastic hinge moment is:

$$M_{\text{max}} = \frac{1}{4} \sigma Y_{\text{Ti}} h^2$$  \hspace{1cm} (29)

For a cross-section of a superhybrid blade, the moment carrying capability of the graphite/epoxy has been neglected, because the low ductility of this material would cause it to fracture well before the other blade components reach their maximum loads. In the boron/aluminum, sufficient aluminum ductility exists in the transverse direction for inclusion in the parameter calculation. Hence, on a composite cross-section, the maximum moment becomes:

$$M_{\text{max}} = \sum_{i=1}^{n} \sigma Y_i \bar{\xi}_i \Delta \xi_i$$  \hspace{1cm} (30)

The allowable composite local ingestion parameter then becomes the allowable metallic parameter multiplied by the scale factor:

$$\frac{4}{\sigma Y_{\text{Ti}} h^2} \sum_{i=1}^{n} \sigma Y_i \bar{\xi}_i \Delta \xi_i$$ \hspace{1cm} (31)

3.4.3 Flutter Analysis

Flutter is "a self-excited oscillation of an aerodynamic lifting surface". During flutter, the aerodynamic forces couple with the blade elastic and inertia forces and increase the kinetic energy of the blade (negative damping). When this aerodynamic energy exceeds the positive mechanical damping
energy, the blade oscillations grow to destructive amplitudes. Thus, it is imperative that flutter conditions must be avoided to prevent high frequency fatigue failure of blades.

Supersonic flutter of fan stages is evaluated by means of an existing analysis program. Individual modes of vibration and steady state aerodynamic conditions are input to the analysis. Unsteady aerodynamic loads resulting from vibratory motion are calculated by the appropriate aerodynamic analysis. Work done on both the forward and the backward traveling wave implementation of each mode is determined by spanwise integration of the product of resultant unsteady load and input vibratory velocity. Work done is non-dimensionalized by dividing kinetic energy by input vibratory mode and expressed as a logarithmic decrement. The absolute value of the lowest decrement for any mode traveling in either direction represents stage stability limit.

In the analysis of supersonic unstalled flutter, the method of superposition of the basic wave solutions of the linearized flow equation is used to simulate the unsteady aerodynamics of a flat plate cascade oscillating in compressible flow. The method is generalized to cover supersonic relative flow with either subsonic or supersonic axial component ($M_\infty > 1$), and thus provide a wide range of application.

The basic assumptions of the analysis are the following:

- Flow is two-dimensional, unsteady, compressible, inviscid, irrotational and isentropic;
- Cascade is infinite, flat plate, at zero incidence and unstalled;
- Vibratory motion is small, constant interblade phase angle, 2 degree of freedom (twist and flap) at a blade/disk system natural frequency.

This existing design system analysis was used with evaluation of coefficients at six representative airfoil strips to provide the STAEUBL system approximate flutter analysis.

3.4.4 Approximate Analysis System

A comprehensive approximate analysis system has been developed using the approximate stress and vibration analysis, the root and local foreign object damage analysis, and an existing Pratt & Whitney Aircraft aerodynamic damping analysis. The flutter and root damage analyses depend on outputs from the blade vibration analysis; specifically, the natural frequencies and the mode shapes. The intermediate and constraint outputs of the component modules of the approximate analyses are listed on Table 3.4-XII.
TABLE 3.4-XII
APPROXIMATE ANALYSIS SYSTEM MODULE OUTPUTS

<table>
<thead>
<tr>
<th>Analysis</th>
<th>Intermediate Outputs</th>
<th>Constraints Calculated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static Analysis</td>
<td>Differential Stiffness</td>
<td>Blade Static Stress</td>
</tr>
<tr>
<td>Vibration Analysis</td>
<td>Frequencies, Mode Shapes, Speed Sensitivity</td>
<td>Resonance Margins</td>
</tr>
<tr>
<td>Flutter</td>
<td></td>
<td>Aerodynamic Damping Coefficient</td>
</tr>
<tr>
<td>Local Foreign Object Damage</td>
<td></td>
<td>Local Foreign Object Damage Parameter</td>
</tr>
<tr>
<td>Root Foreign Object Damage</td>
<td></td>
<td>Root Foreign Object Damage Parameter</td>
</tr>
</tbody>
</table>

3.5 OPTIMIZATION METHOD

A common engineering design problem is the determination of values for design variables which minimize design quantity such as weight, drag, or cost, while satisfying a set of auxiliary conditions. In the STAEBL program, the structural design of a composite or hollow fan blade is accomplished by varying airfoil section thicknesses, chord, titanium skin thickness, etc. to minimize a combination of weight and cost subject to constraints on resonance, flutter, stress, and foreign object damage.

3.5.1 General Optimization Theory and Background

The engineering design process can be modeled as a mathematical programming problem in optimization theory. In theoretical terms, this constrained minimization problem can be expressed as follows:

\[
\text{minimize } f(x),
\]

subject to the auxiliary conditions,

\[
g_i(x) \leq 0, \ i = 1, \ldots, m.
\]

The quantity \( x = (x_1, \ldots, x_n) \) is the vector of \( n \) design variables. The scalar function to be minimized, \( f(x) \), is the objective function; and \( g_i(x) \leq 0, \ i = 1, \ldots, m \), are the \( m \) inequality constraints. Upper and lower bounds on the design variables, e.g.,

\[
L_i \leq x_i \leq U_i, \ i = 1, \ldots, n,
\]

are referred to as side constraints. The \( n \)-dimensional space spanned by the design variables is design space. If \( f(x) \) and \( g_i(x), \ i = 1, \ldots, m \), are all
linear functions of $x$, then the optimization problem is a linear programming problem (LP) which can be solved by well-known techniques such as Dantzig's simplex method. If $f(x)$ or any of the $g_i(x)$'s are nonlinear, then it is a nonlinear programming (NP) problem for which a number of solution techniques are also available. If the objective function, $f(x)$, is to be maximized, then the equivalent problem of minimizing $-f(x)$ is considered.

Any choice of variables, $x$, in design space that satisfies all the constraints, (33) and (34), is a feasible point. As shown in Figure 3.5-1, the union of all feasible points comprises the feasible region. The locus of points which satisfy $g_i(x) = 0$, for some $i$, forms a constraint surface. On one side of the surface, $g_i(x) < 0$ and the constraint is satisfied; on the other side, $g_i(x) > 0$ and the constraint is violated. Points in the interior of the feasible region are free points; points on the boundary are bound points. If it is composed of two or more distinct sets, the feasible region is disjoint. A design point in the feasible region that minimizes the objective function is an optimal feasible point and is a solution of the problem posed in (32) through (34). As in any nonlinear minimization problem, there can be multiple local minima. In this case, the global minimum is the optimal feasible point. If a design point is on a constraint surface (i.e., $g_i(x) = 0$ for some $i$), then that particular constraint is active. A solution to a structural optimization problem is almost always on the boundary of the feasible region, and is usually at the intersection of two or more constraint surfaces (i.e., there are two or more active constraints).

![Figure 3.5-1 Feasible Region Is Union of All Points that Satisfy All Constraints](image-url)
There are two basic approaches to solving the constrained optimization problem posed in (32) through (34): direct methods (e.g., methods of feasible directions) and indirect methods (e.g., penalty function methods).

In a direct method, the objective function and constraints are evaluated independently, and the constraints are treated as limiting surfaces. Zoutendijk's method of feasible directions is an example of a direct method and will be discussed further in Sections 3.5.2 and 3.5.3.

In an indirect method, the problem is reformulated so that (32) through (34) are replaced by a single unconstrained minimization problem. For example, in an exterior penalty function method, violations of the constraints are added onto the objective function to form an augmented objective function,

\[
\phi(x; R_k) = f(x) + R_k \sum_{i=1}^{m} \left[ g_i(x) \right]^2,
\]

where:

\[
\left[ g_i(x) \right]_+ = \begin{cases} 
0, & \text{if } g_i(x) \leq 0, \\
g_i(x), & \text{if } g_i(x) > 0.
\end{cases}
\]

Thus, if a particular constraint is satisfied, then it contributes nothing to the summation in (35). If a constraint is violated, however, then a penalty term is added onto the objective function, which increases as the square of the violation. The design points, \( x \), must all satisfy the side constraints (34).

The augmented objective function \( \phi(x; R_k) \) is minimized for successively increasing values of the penalty parameter \( R_k \), i.e.,

\[
R_1 < R_2 < R_3 < \ldots \rightarrow \infty.
\]

Under rather mild conditions, the sequence of minima \( \{x_1, x_2, x_3, \ldots\} \) corresponding to (37), converge to a local optimum of the constrained optimization problem identified in (32) through (34). One advantage of this approach is that each of the minimization problems can be solved using a standard unconstrained function minimization technique, such as a conjugate gradient or quasi-Newton method.

Several programs are generally available in software libraries (e.g., IMSL = International Mathematical and Statistical Libraries, Inc., and HARWELL) that can solve the constrained minimization problem using either direct or indirect techniques. Because of its versatility in solving structural optimization problems at Pratt & Whitney Aircraft, NASA/Langley, General Motors, and Ford Motor Co., the COPES/CONMIN computer program was selected for the STAEBL contract. This program was developed by G. N. Vanderplaats of the Naval
Postgraduate School and has the added capability of solving both constrained minimization problems (32) through (34) and unconstrained minimization problems (32). COPES (COntrol Program for Engineering Synthesis) is a user-oriented FORTRAN program that prepares an input data set for the optimization program CONMIN (CONstrained MINimization). Two solution techniques are available for the constrained minimization problem.

1. **Exact analysis** - utilizes the method of feasible directions applied to the actual objective function and constraints. This approach is discussed in Section 3.5.2.

2. **Approximate analysis** - utilizes the method of feasible directions applied to Taylor series approximations and to the objective function and constraints. This approach is discussed in Section 3.5.3.

### 3.5.2 COPES/CONMIN Exact Analysis: Method of Feasible Directions

In this method, a sequence of designs \((x_0, x_1, \ldots)\) is produced which converges to a local optimum design, \(x_{\text{opt}}\), provided a feasible region exists. The successive designs are generated iteratively as a sequence of one-dimensional line searches, i.e.,

\[
x_{i+1} = x_i + \alpha s_i,
\]

for \(i = 0, 1, 2, \ldots\), where \(s_i\) is the search direction and \(\alpha\) are chosen so that once the feasible region has been entered, all subsequent iterates remain feasible and the magnitude of the objective function is reduced at each step. If the initial design, \(x_0\), is infeasible, then gradients of the violated constraints are calculated so that search directions can be established which lead to the feasible region, provided one exists.

Once the feasible region has been entered, a particular direction is pursued until either: a) a local minimum of the objective function, \(f(x)\), has been determined or, b) a constraint boundary has been reached. The value of \(\alpha\) in (38) at the termination point of this one-dimensional line search in the \(s_i\) direction is determined by interpolating polynomial fits of several trial values of the objective function and constraints. A schematic of a typical case is shown in Figure 3.5-2. The initial design, \(x_0\), is infeasible. The design point, \(x_i\), is a relative minimum of the objective function. The remaining search directions terminate at constant boundaries until \(x_{\text{opt}}\) is reached.

If a local minimum of the objective function has been reached, then the gradient of the objective function is calculated, and the procedure continues in the direction opposite to this (i.e., the "path of steepest descent"). If a constraint boundary has been reached first, however, then a new search direction can be determined using Zoutendijk's method of feasible directions as follows. A direction, \(s_i\), is usable if the objective function initially does not increase along this path, i.e.,

\[
s_i \cdot \nabla f(x_i) < 0.
\]

(39)
In addition, $s_i$ is feasible if no active constraints are initially violated along this path, i.e.,

$$s_i \cdot \nabla g_j(x_i) \leq 0, \quad j = 1, \ldots, \text{NAC},$$

where a subscript, $j$, is chosen for each of the constraints that are active at $x_i$. As shown schematically in Figure 3.5-2, allowable paths that emanate from $x_i$ comprise the usable feasible sector.

3.5.2.1 Choice of Search Parameters for COPES/CONMIN

In Zontendijk's method, the search direction, $s_i$, is determined by solving a sub-optimization problem, i.e.,

maximize $\beta$,
subject to:

$$s_i \cdot \nabla f(x_i) + \beta \leq 0,$$

$$s_i \cdot \nabla g_j(x_i) + \theta_j \beta \leq 0, \quad j = 1, \ldots, \text{NAC}\quad (41)$$

$$|s_i| \text{ bounded.}$$

The parameter $\theta_j$, the push-off factor, determines the orientation of the new search direction vector, $s_i$, in the usable feasible sector by pushing the
search away from the constraints into the feasible region. As shown in Figure 3.5-3, \( s_j \) approaches the constraint surface, \( g_j(x) \), tangentially as \( \theta_j \to 0 \), and \( s_j \) approaches a level curve to the objective function tangentially as \( \theta_j \to \infty \). For a linear constraint, \( \theta_j \) can be set to zero and the search can proceed along that particular constraint surface. If \( \theta_j \) is too small, then for nonlinear constraints with convex curvature, the same constraint will be immediately re-encountered. In this case, the search will "skid" along the same constraint boundary with little change in the objective function. If \( \theta_j \) is too large, then the search will "zigzag" back and forth between two or more constraints, and the objective function will again not be reduced rapidly enough. A compromise value of \( \theta_j = 1 \) is the default value used by COPES/CONMIN for the initial iteration. Since many of the constraints (e.g., flutter, resonance, etc.) in the STAEBL optimization problems were nearly linear (at least locally), the value \( \theta_j = 0.3 \) was used for the initial iteration to give more rapid convergence.

The rate of convergence is also affected by the value of \( \nu \), the constraint thickness parameter in COPES/CONMIN. For theoretical purposes, the \( i \)th constraint is satisfied if \( g_i(x) \leq 0 \) and is active if \( g_i(x) = 0 \). For computational purposes (as shown in Figure 3.5-4), COPES/CONMIN considers the
The $i$th constraint to be satisfied if $g_i(x) \leq CT$ and to be active if $|g_i(x)| \leq -CT$, where $CT$ is a negative number. If $|CT|$ is too small, then one or more constraints can be active on one iteration and inactive on the next, only to become active again on a subsequent iteration - another instance of "zigzagging". A proper choice of $CT$ ensures that two or more constraints will often be simultaneously active when a new search direction is chosen. In this case, as shown in Figure 3.5-5, the search will proceed down the "valley" formed by the constraint surfaces. The default value in COPES/CONMIN is $CT = -0.1$ (i.e., a constraint is considered active if it is within 10 percent of its specified value). For many STAEBL applications, a value $CT = -0.1$ was too large since too many constraints were simultaneously active during the early iterations, and new search directions could not be established. Consequently, the value $CT = -0.05$ was used.

During the COPES/CONMIN optimization procedure, the values of $CT$ and $\theta_j$ are updated as follows. After the first few iterations, the value of $CT$ is decreased monotonically so that fewer constraints will be active when new search directions are established. A minimum value of $|CT|$ is given by $CTMIN$; the default value in COPES/CONMIN is $CTMIN = 0.004$. In addition, the value of the push-off factor, $\theta_j$, is also readjusted at each iteration according to the value of the active constraint to which it applies and to the current value of $CT$. Thus, $\theta_j$ is a quadratic function of these parameters, i.e.,

$$\theta_j = \theta_0 \left( \frac{g_1(x_1)}{CT} - 1 \right)^2,$$

(42)

where $\theta_0$ is the initial value of $\theta_j$ (for STAEBL we have chosen $\theta_0 = 0.3$). A maximum value of $\theta_j = 50$ is also imposed.

The iteration is terminated under three conditions in COPES/CONMIN:

1. If the objective functions for three successive iterates are all within a prescribed error tolerance, then the procedure has converged to a local optimum. COPES/CONMIN uses default values of $DELFUN = 0.0001$ for the relative change in objective function and $(DABFUN = 0.0001) \times$ initial objective value for absolute change in the objective function as its convergence criteria. For STAEBL application, 1 percent differences in the objective function were adequate for convergence so that $DELFUN = DABFUN = 0.01$. These increased values also reduced the number of function calls required for convergence.

2. If convergence has not been obtained after a certain number of iterations inside the feasible region, the procedure is terminated. Either this design can be accepted or else the optimization procedure can be restarted if progress toward an optimum is obviously being made. COPES/CONMIN uses a default value of 20 for the total number of iterations.

3. If the feasible region cannot be located after a certain number of iterations (the COPES/CONMIN default value is 10), then the process is terminated. At this time, either a new starting guess should be chosen, or else the objective function and constraints should be examined to determine whether or not a feasible region exists.
Figure 3.5-4 Constraint Thickness Parameter, CT, Determines when a Constraint is Satisfied, Violated, or Active

Figure 3.5-5 For Proper Choice of CT, Two Constraints Become Simultaneously Active so that Search Proceeds down the "Valley" Formed by the Constraints
3.5.2.2 Scaling of Design Variables in COPES/CONMIN

Performance of the method of feasible directions can be greatly affected by the scaling of the design variables. At the beginning of each iteration in COPES/CONMIN, a new search direction is established according to Zoutendijk's method (41). This procedure is based upon the gradient of the objective function and each constraint with respect to each of the design variables. The choice of the search direction is very sensitive to the components of these gradients. For example, in a two design variable problem, suppose that a 1 percent change in $x_1$ leads to a 10 percent change in the objective function, $f(x)$; whereas a 1 percent change in $x_2$ leads to only a 0.1 percent change in $f(x)$. To reduce the objective function most rapidly, the search direction will be primarily in the $x_1$ direction. The "weak" variable, $x_2$, will be virtually unchanged, at least for several iterations. To obtain the optimal design, a relatively large change in $x_2$ must be made to affect the objective function and constraints.

In a well-formulated problem, the components of the gradient of the objective function with respect to the design variables should all be roughly the same order of magnitude. The scaling option in COPES/CONMIN can be used to equilibrate the gradient components as follows. The $i^{th}$ design variable, $x_i$, is scaled by dividing it by its initial value $x_i^0$, i.e.;

$$
\xi_i = \frac{x_i}{x_i^0}
$$

provided $x_i^0$ is nonzero. Using (43) in the chain rule, the $i^{th}$ component of the scaled gradient with respect to the nondimensional variable, $\xi_i$, is given by:

$$
\frac{\partial f}{\partial \xi_i} = \frac{\partial x_i}{\partial \xi_i} \frac{\partial f}{\partial x_i} = x_i^0 \frac{\partial f}{\partial x_i}
$$

Thus, the ratio of the $i^{th}$ components in the scaled gradients is given by:

$$
\frac{\partial f}{\partial \xi_i} / \frac{\partial f}{\partial x_i} = x_i^0
$$

The effect of scaling was exhibited by the optimal design of a hollow fan blade with a titanium outer skin. The four design variables were: $x_1$ and $x_2$ (blade thicknesses, $t_{\text{root}}$ and $t_{\text{tip}}$, at root and tip, respectively); $x_3$ (distance, $d_{\text{root}}$, from bottom of hollow section to the blade root); and $x_4$ (titanium skin thickness, $t_{\text{Ti}}$). The objective function, $f(x)$, was blade weight. As shown in Table 3.5-1, the components of the gradient of the objective function varied by two orders of magnitude since $\partial f/\partial x_3 = 0.027$ and $\partial f/\partial x_4 = 2.7$. Consequently, without scaling, the minimization proceeded by varying the titanium skin thickness, $x_4$, and leaving $x_3$ virtually unchanged - an optimum design was not achieved. When scaling was introduced according to (43), the gradient components in Table 3.5-I were all of the same order of magnitude. In this case, the search direction varied all four design variables (including the hollow section location) simultaneously so that an optimal design was obtained.
TABLE 3.5-1

UNSCALED AND SCALED VALUES OF GRADIENT OF OBJECTIVE FUNCTION RELATIVE TO THE DESIGN VARIABLES FOR HOLLOW FAN BLADE

<table>
<thead>
<tr>
<th>Design Variable, $x_i$</th>
<th>Initial value cm (inches)</th>
<th>$i^{th}$ Component of Gradient $= \frac{\partial f}{\partial x_i}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1 = t_{\text{root}}$</td>
<td>2.22 (0.875)</td>
<td>Unscaled 0.460  Scaled 0.403</td>
</tr>
<tr>
<td>$x_2 = t_{\text{tip}}$</td>
<td>0.85 (0.334)</td>
<td>Unscaled 0.140  Scaled 0.047</td>
</tr>
<tr>
<td>$x_3 = d_{\text{root}}$</td>
<td>24.51 (9.650)</td>
<td>Unscaled 0.027  Scaled 0.260</td>
</tr>
<tr>
<td>$x_4 = t_{\text{Ti}}$</td>
<td>0.30 (0.120)</td>
<td>Unscaled 2.700  Scaled 0.324</td>
</tr>
</tbody>
</table>

The scaling options in COPES/CONMIN are controlled by the input parameter NSCAL and are given as follows:

\[
\begin{align*}
\text{NSCAL} & > 0 : \quad \text{Rescale design variables by dividing by current values every NSCAL iteration,} \\
\text{NSCAL} & = 0 : \quad \text{No scaling (default value),} \\
\text{NSCAL} & < 0 : \quad \text{Scale design variables by dividing by user-input scaling variables}
\end{align*}
\]

For STAEBL demonstration, scaling was always used. The value $\text{NSCAL} = n+1$ (where $n =$ number of design variables) was recommended by G. Vanderplaats since this strategy worked well for unconstrained minimization problems using the conjugate gradient method.

3.5.2.3 Number of Function Calls for COPES/CONMIN

Engineering design problems are considered small or large according to the number of design variables as follows:

- Small: $n \leq 10$,
- Moderate: $10 < n \leq 50$,
- Large: $n > 50$.

The number, $N$, of function calls required for convergence of the method of feasible directions for COPES/CONMIN can be approximated as follows. As indicated in Figure 3.5-2, each iteration consists of a gradient evaluation of the objective function and constraints to determine the search direction, followed by a one-dimensional line search in that direction. When the gradients are not known analytically (as is the case for the STAEBL application), a backward difference gradient approximation is used. For $n$ design variables, $n$ function calls are required for the finite difference
gradient calculation. The one-dimensional line search usually requires 3 additional function evaluations to update the objective function and constraints and to determine where the search should terminate. Thus, for \( m \) iterations, with \( n + 3 \) function calls per iteration, we have:

\[
N = m(n + 3). \tag{48}
\]

Typically, convergence is attained in approximately 10 iterations so that \( N \approx 10n + 30 \). Note that \( N \) increases roughly linearly as a function of the number, \( n \), of design variables.

The limiting feature in these analyses is the computer time required per function call to evaluate the objective function and constraints. For example, calculation of three natural frequencies, mode shapes, resonance function, flutter parameters, stresses, weight, and bird ingestion parameters for a gas turbine engine fan blade requires approximately 25 seconds on an IBM 3033 computer. In this case, 10 iterations for a 12 design-variable optimization problem would require approximately 150 function calls and 1 hour of computer time. Sometimes the number of function calls can be reduced by using the approximate analysis version of COPES/CONMIN, discussed in Section 3.5.3.

3.5.3 COPES/CONMIN Approximate Analysis: Taylor Series Expansions

In the exact analysis version of COPES/CONMIN for the method of feasible directions, the actual vibration, flutter and stress computer programs are implemented for every function call to calculate the objective function and constraints. In the approximate analysis version of COPES/CONMIN, the objective function and constraints are all represented numerically by second order Taylor series expansions about some nominal design point. After the objective function and constraints have been evaluated for several distinct design points, linear and quadratic polynomial surfaces can be curve-fit through these points. Then, instead of implementing the actual vibration, flutter, and stress programs, the objective function and constraints can be closely approximated by evaluating the Taylor series expansions. As higher order terms in the series are determined, the approximate analyses become increasingly accurate.

In general, the Taylor series expansion through quadratic terms for an arbitrary function, \( f \), (representing either the objective function or constraints) is given by:

\[
f(x) \doteq f(x_0) + \overline{f}(x_0) \cdot (x-x_0) + \frac{1}{2} (x-x_0) \cdot H(x_0) \cdot (x-x_0), \tag{49}
\]

where \( x = (x_1, x_2, ..., x_n)^T \) is a vector of \( n \) design variables, \( x_0 \) is the nominal design (center of expansion), and \( \cdot \) denotes dot product, i.e., \( u \cdot v = u^T v \), where \( T \) denotes transpose. The gradient vector, \( \overline{f}(x_0) \), and Hessian matrix, \( H(x_0) \), are given by:
The coefficients in the Taylor series expansion (i.e., \( f(x_0) \), and all the terms in the gradient vector and Hessian matrix) in (49) are unknown. Since the Hessian matrix is symmetric, the total number, \( N_Q \), of unknowns in (49) is given by:

\[
N_Q = 1 + n + n(n+1)/2.
\] (51)

After \( N_Q \) linearly independent function evaluations have been made, all unknown coefficients in (49) are determined, and the quadratic Taylor series approximations are available. Thus, the method of feasible directions can then be applied to the quadratic approximations themselves instead of calling the actual vibration and stress programs. By evaluating the weight, natural frequencies, aerodynamic damping parameters, etc. using the Taylor series approximations, COPES/CONMIN can reduce the number of expensive calls to the vibration programs and thereby reduce the total computer time of the optimization program. In addition, since the objective function and constraints are expressed in terms of quadratic polynomials, the gradient calculations required by the method of feasible directions can be performed analytically rather than by finite differences.

The approximate optimization technique is performed sequentially. The first design, \( x_0 \), (the nominal design) is used to determine the constant term, \( f(x_0) \), in the Taylor series approximation. Another nearby design is selected so that one term in the gradient vector in (49) can be determined. As more designs are evaluated, more terms in the Taylor series can be evaluated. After each new approximation is made, the method of feasible directions is applied to the current approximation to obtain a new local optimum. This new approximate optimum design is then evaluated using the actual vibration and stress programs. This new design point is then appended to the sequence of previous designs so that the next term in the quadratic Taylor series can be determined. After \( n + 1 \) linearly independent designs have been determined, then all the first order (gradient) terms in the Taylor series (49) are known. The next \( n(n+1)/2 \) design evaluations are used to determine the Hessian matrix. A sequence of local optimum designs determined by applying the method of feasible directions to a sequence of local Taylor series approximations is
illustrated schematically in Figure 3.5-6. In Figure 3.5-6(A), only linear approximations to the objective function and constraints can be made. In Figure 3.5-6(B), more than n+1 design points are known so that some curvature in the objective function and constraint surfaces can be modeled. Eventually, in Figure 3.5-6(C), more than NQ design points have been evaluated so that a full quadratic approximation can be made.

**Figure 3.5-6** Sequence of Approximate Analyses Using the Method of Feasible Directions Becomes Increasingly More Accurate as More Terms in the Taylor Series Are Determined.
Since some of the initial approximations to the problem can have unbounded solutions, upper and lower bounds are placed on the design variables to limit the excursion between successive iterates. Once all \( N_Q \) terms in (49) have been determined, successive iterates should be close together and converge to a local optimum design. Convergence is obtained when the local optima of two successive approximate problems agree within a certain tolerance. If more than \( N_Q \) linearly independent designs are determined, then a weighted least squares fit of the redundant data is used to determine the Taylor series coefficients in (49). A default value for the upper limit in the number of function calls is approximately \( 3N_Q \) in COPES/CONMIN.

Assuming that \( m = 10 \) iterations are required for convergence for the exact analysis, the number of function calls is given approximately by (48) as \( N = 10n + 30 \). For the quadratic Taylor series approximation, there are \( N_Q = 1 + n + n(n+1)/2 \) unknowns. Equating \( N \) and \( N_Q \), we find that the break even point is approximately \( n = 20 \) design variables. Further, since the number of function calls for the quadratic Taylor series approximation and exact analysis are proportional to \( n^2 \) and \( n \), respectively, the quadratic Taylor series method rapidly becomes noncompetitive for \( n > 2 \).

In some cases, certain reduced approximations can be made which can lead to further reductions in the number of function calls required for convergence. Two options are available in COPES/CONMIN in which not all \( N_Q \) unknowns in (49) need to be determined. First, if the off-diagonal terms in the Hessian matrix are neglected, then there are only \( n \) diagonal terms to be determined. In this model, the total number of unknowns in equation (49) is reduced to:

\[
N_Q = 1 + n + n,
\]

so that \( N_Q \) is now a linear function of \( n \). Before using the diagonal Hessian matrix option, it is sometimes instructive to analyze the problem using the full Hessian matrix option to determine how large the off-diagonal terms are compared to the diagonal terms. If there are several large off-diagonal terms in critical functions (i.e., objective function or active constraints), then the full quadratic approximation may converge more rapidly than the diagonal Hessian option.

A second reduced approximation is available in COPES/CONMIN by neglecting all the quadratic terms in the Taylor series. The approximating model is now strictly linear and the total number of unknowns in (49) is given by:

\[
N_L = 1 + n.
\]

Since a sequence of constrained linear optimization problems is solved until convergence is obtained, this approach is known as "sequential linear programming". Each linear subproblem can be solved using either the method of feasible directions or the Simplex method.

In summary, the linear and diagonal Hessian Taylor series approximations require the fewest function evaluations and should be used whenever possible. The success of the approximate techniques depends upon how accurately the Taylor series approximate the objective function and active constraints in a
neighborhood of a local optimum. If the actual functions are relatively smooth (e.g., differentiable with no rapid fluctuations) then the Taylor series fits should be very accurate. On the other hand, if the objective function or constraints are nonsmooth (e.g., discontinuous or rapidly varying), then the approximate techniques may not provide accurate representations of these surfaces near a local optimum. (It should be pointed out that the finite difference gradient calculations for the exact analysis may also experience difficulty in these cases.) Plots of level curves in two-variable function space (i.e., plotting objective function and constraint values as functions of only two design variables while holding the other design variables constant) demonstrate that airfoil weight, resonance margin, and aerodynamic damping exhibit nearly linear variation (at least locally) with blade thickness throughout a large portion of design space. Thus, in many STAEBL applications, quadratic, diagonal Hessian, and sequential linear Taylor series approximations have provided rapid and accurate results.

Convergence difficulties for the approximate optimization methods can occur due to linear dependence among the designs. The unknown coefficients in the Taylor series approximation are determined by solving systems of simultaneous equations. For example, if (49) were evaluated at \( N_0 \) designs, then there would be \( N_0 \) simultaneous linear equations to be solved for the \( N_0 \) unknown coefficients for the objective function and each of the constraints. If two of the \( N_0 \) designs were identical or numerically very "close" however, then the coefficient matrices for the linear systems of equations would be singular or numerically singular, respectively (i.e., the designs would be linearly dependent or numerically dependent, respectively). This situation can arise, for example if an initial design either coincides with, or was very close to, a local optima. In this case, it is very possible that this local optimum design point would be generated several times during the first \( N_0 \) iterations. Whenever a previously generated design is appended to the sequence of designs, the resulting system of equations would be linearly dependent and could not be solved. By taking several new starting points, COPES/CONMIN attempts to determine a new linearly independent design. There is also an option for the user to supply a priori his own set of linearly independent design vectors. In this case, COPES/CONMIN evaluates the objective function and constraints at these user-supplied points and then determines the surface fits through these points.

The approximate analysis option in COPES/CONMIN is especially useful if only a limited amount of data for the objective function and constraints are available. For example, some finite element programs (such as NASTRAN or MARC) for structural analyses are so time consuming that only a few function calls can be afforded for an optimization program. In other cases, data may be obtainable only through experiments so that only a limited number may be available. In the approximate analysis option, the available data can be entered a priori and the method of feasible directions can be applied to the Taylor series approximations.

3.5.4 COPES/CONMIN Interfaces to Vibration, Flutter, and Stress Programs

The COPES/CONMIN program is limited via subroutine ANALIZ to the approximate vibration, flutter, stress, and foreign object damage programs used for the structural analysis of blades at Pratt & Whitney Aircraft/Commercial Products
Division via subroutine analyse as shown in Figure 3.5-7. Once an optimal feasible design has been obtained by COPES/CONMIN, this blade design must be evaluated by the refined analysis (finite element program) for further tailoring and possible re-optimization.

**Figure 3.5-7 COPES/CONMIN Is Linked Via Subroutine ANALYSE to Approach analyses**

Subroutine ANALIZ is called by COPES/CONMIN in order to evaluate the objective function and constraints. There are 3 options, designated by different values of the parameter ICALC, utilized by COPES/CONMIN when calling subroutine ANALIZ:

- **ICALC = 1**: Read data, set the parameters that are used throughout the analysis, and analyze the initial design.
- **ICALC = 2**: Analyze the current design.
- **ICALC = 3**: Write output data and parameters and results of analysis on final design.
In order to accomplish these tasks, subroutine ANALIZ calls the vibration, flutter, stress, and FOD programs whenever necessary. The transfer of information between COPES/CONMIN and these approximate analyses is accomplished by accessing the data in common block GLOBCM. In Table 3.5-II, the global locations in GLOBCM and the FORTRAN names are given for the design variables, objective function, and constraint information used in the STAEBL procedure included in Appendix B.

### Table 3.5-II

<table>
<thead>
<tr>
<th>Global Location</th>
<th>FORTRAN Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>OBJF</td>
<td>Airfoil weight</td>
</tr>
<tr>
<td>2-6</td>
<td>FN</td>
<td>Frequencies (Root 1 - Root 5)</td>
</tr>
<tr>
<td>7-11</td>
<td>DLAR</td>
<td>Aerodynamic damping parameter (Root 1 - Root 5)</td>
</tr>
<tr>
<td>12-32</td>
<td>THKVAL</td>
<td>Airfoil thicknesses (max. of 21 stations)</td>
</tr>
<tr>
<td>33-37</td>
<td>RF</td>
<td>Resonance margin for 1st order (Root 1 - Root 5)</td>
</tr>
<tr>
<td>38-42</td>
<td>RF</td>
<td>Resonance margin for 2nd order (Root 1 - Root 5)</td>
</tr>
<tr>
<td>43-47</td>
<td>RF</td>
<td>Resonance margin for 3rd order (Root 1 - Root 5)</td>
</tr>
<tr>
<td>48-52</td>
<td>RF</td>
<td>Resonance margin for 4th order (Root 1 - Root 5)</td>
</tr>
<tr>
<td>53</td>
<td>BRCC</td>
<td>Root Chord</td>
</tr>
<tr>
<td>54-74</td>
<td>FODLSB</td>
<td>LSBIP = Local stress bird ingestion parameter</td>
</tr>
<tr>
<td>75</td>
<td>DLE</td>
<td>Distance to hole from leading edge</td>
</tr>
<tr>
<td>76</td>
<td>DTE</td>
<td>Distance to hole from trailing edge</td>
</tr>
<tr>
<td>77</td>
<td>DROOT</td>
<td>Distance to hole from root edge</td>
</tr>
<tr>
<td>78</td>
<td>DTIP</td>
<td>Distance to hole from tip edge</td>
</tr>
<tr>
<td>79</td>
<td>TTI</td>
<td>Thickness of titanium</td>
</tr>
<tr>
<td>80</td>
<td>TLT</td>
<td>Thickness of borsic titanium</td>
</tr>
<tr>
<td>81</td>
<td>OBJFUN</td>
<td>Objective function</td>
</tr>
<tr>
<td>82-89</td>
<td>SMAXTS</td>
<td>Root Tsai-Wu (layer 1-8)</td>
</tr>
<tr>
<td>90-89</td>
<td>SMAX2S</td>
<td>Hole Tsai-Wu (layer 1-8)</td>
</tr>
<tr>
<td>98-105</td>
<td>SMAX3S</td>
<td>Spanwise bending ingestion Tsai-Wu (layer 1-8)</td>
</tr>
<tr>
<td>106-113</td>
<td>SMAXLS</td>
<td>Leading edge Tsai-Wu (layer 1-8)</td>
</tr>
<tr>
<td>114-121</td>
<td>SMAXTS</td>
<td>Trailing edge Tsai-Wu (layer 1-8)</td>
</tr>
<tr>
<td>122-128</td>
<td>Theta</td>
<td>Fiber direction (7 layers)</td>
</tr>
<tr>
<td>129</td>
<td>HLRTIO</td>
<td>Hole to length ratio</td>
</tr>
<tr>
<td>130</td>
<td>ECRTIO</td>
<td>Edge to chord ratio</td>
</tr>
<tr>
<td>131-151</td>
<td>TOVB</td>
<td>t/b's</td>
</tr>
<tr>
<td>152</td>
<td>FODMAX</td>
<td>Max LSBIP</td>
</tr>
<tr>
<td>153</td>
<td>TIS</td>
<td>Titanium skin thickness</td>
</tr>
<tr>
<td>154</td>
<td>TIC</td>
<td>Titanium center thickness</td>
</tr>
<tr>
<td>155</td>
<td>PCBA</td>
<td>% boron aluminum</td>
</tr>
<tr>
<td>156</td>
<td>VAL</td>
<td>2/TIS + TIC)/THKVAL(1)</td>
</tr>
</tbody>
</table>
3.5.5 Test Cases for STAEBL Optimization Procedure

Several test cases were used to demonstrate certain features of the STAEBL optimization procedure. Since the first three examples involved two design variables, the two-variable function space option of COPES/CONMIN was used. With this feature, the objective function and constraints were evaluated over a rectangular grid of points so that the constraint curves and level curves for the objective function could be presented.

Case 1: Local optimum design for disjoint feasible region

Determine \( (t_r, t_t) \) that minimize blade weight:

Subject to:

\[
\begin{align*}
0.25 < t_r < 0.5 \\
0.1 < t_t < 0.3,
\end{align*}
\]

\[
\begin{align*}
130 < \omega_1 < 173 \\
272 < \omega_2 < 320 \\
390 < \omega_3 < 504.
\end{align*}
\]

For this problem, \( (t_r, t_t) \) are the maximum blade thicknesses (in inches) at the root and tip, respectively, and \( \omega_i \), \( i = 1, 2, 3 \), are the first three blade frequencies (in Hertz). The level curves for the blade weight were equally spaced lines. Due to the strongly nonlinear behavior of \( \omega_3 \), the feasible region was composed of two distinct components. As shown in Figure 3.5-8, the initial design was infeasible, and the STAEBL optimization procedure converged to a local optimum design after five iterations.

Figure 3.5-8 STAEBL Optimization Procedure May Not Converge to Global Optimum if Feasible Region is Disjoint
Case 2: Global optimum design for disjoint feasible region with three flutter constraints

This example had the same objective function and constraints as Case 1 except that the following constraints on the first three aerodynamic damping parameters were imposed: $\delta_i < 0.015$, $i = 1, 2, 3$. As shown in Figure 3.5-9, these flutter constraints forced the search into a relatively small feasible region containing the global optimum, so that convergence was achieved rather quickly.

Figure 3.5-9 STAEBL Optimization Procedure with Three Flutter Constraints Converges to Global Optimum
Case 3: Exact and approximate optimization procedures converge to the same local optimum.

Again, this example had the same objective function and constraints as Case 1. In this case, a comparison was made between the exact method of feasible directions, and the quadratic Taylor series approximate optimization method. Starting from the same infeasible initial design shown in Figure 3.5-10, both methods converged to the same local optimum design. The exact method required 19 function calls. The quadratic Taylor series had $N_Q = 6$ unknowns and achieved convergence using only six function calls.

<table>
<thead>
<tr>
<th></th>
<th>FUNC CALLS (SEC.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXACT</td>
<td>19 139</td>
</tr>
<tr>
<td>APPROX.</td>
<td>6 48</td>
</tr>
</tbody>
</table>

Figure 3.5-10 Exact and Approximate Analyses for STAEBL Procedure Converge to Same Local Optimum.
Case 4: Hollow fan blade with titanium skin

Determine \((t_r, t_t, d_r, t_{Ti})\) that minimizes blade weight + manufacturing cost + maintenance cost:

Subject to:

\[
\begin{align*}
{t_r} &\geq 0 & \delta_i &\geq 0 & \text{Root stress} \leq \text{max.} \\
{t_t} &\geq 0 & 0.025 \leq {t_r}/b \leq 0.1 & \text{Hole stress} \leq \text{max.} \\
{d_r} &\geq 2.0 & 0.025 \leq {t_t}/b \leq 0.1 & \text{Leading edge stress} \leq \text{max.} \\
{t_{Ti}} &\geq 0.03 & \text{LSBIP} \leq \text{max.} & \text{Trailing edge stress} \leq \text{max.} \\
\end{align*}
\]

For this problem, \(d_r\) is the distance between blade root and the bottom of the hole, \(t_{Ti}\) is the thickness of the titanium skin, \(b\) is the blade chord, LSBIP = Local Stress Bird Ingestion Parameter, and BBIP = Bending Bird Ingestion Parameter. After beginning with a feasible initial design, up to five constraints were active for some iterates. As shown in Table 3.5-III, convergence was achieved after seven iterations and three constraints were active at the final design.

**TABLE 3.5-III**

**HOLLOW FAN BLADE HAS 3 ACTIVE CONSTRAINTS FOR FINAL DESIGN**

<table>
<thead>
<tr>
<th>Iteration</th>
<th>(t_r)</th>
<th>(t_t)</th>
<th>(d_r)</th>
<th>(t_{Ti})</th>
<th>Objective Function</th>
<th>Active Constraints*</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.8750</td>
<td>0.3340</td>
<td>9.65</td>
<td>0.1200</td>
<td>1.464</td>
<td>32,33</td>
</tr>
<tr>
<td>1</td>
<td>0.8744</td>
<td>0.3388</td>
<td>9.07</td>
<td>0.0974</td>
<td>1.379</td>
<td>1,32,33</td>
</tr>
<tr>
<td>2</td>
<td>0.8742</td>
<td>0.3396</td>
<td>9.02</td>
<td>0.0920</td>
<td>1.360</td>
<td>1,7,8,32,33</td>
</tr>
<tr>
<td>3</td>
<td>0.8741</td>
<td>0.3397</td>
<td>8.01</td>
<td>0.0922</td>
<td>1.328</td>
<td>1,7,8,32,33</td>
</tr>
<tr>
<td>4</td>
<td>0.8988</td>
<td>0.3318</td>
<td>7.04</td>
<td>0.0930</td>
<td>1.304</td>
<td>1,7,8,32,33</td>
</tr>
<tr>
<td>5</td>
<td>0.8974</td>
<td>0.3362</td>
<td>5.06</td>
<td>0.0931</td>
<td>1.229</td>
<td>1,7,8,32,33</td>
</tr>
<tr>
<td>6</td>
<td>0.9000</td>
<td>0.3358</td>
<td>5.04</td>
<td>0.0920</td>
<td>1.224</td>
<td>1,7,8</td>
</tr>
<tr>
<td>7</td>
<td>0.9078</td>
<td>0.3318</td>
<td>4.88</td>
<td>0.0922</td>
<td>1.221</td>
<td>1,8,33</td>
</tr>
</tbody>
</table>

*Active Constraints Notes

1: \(\delta_1\) (initially violated)

7: LSBIP (next to last relevant station)

8: LSBIP (last relevant station)

32: \(t_r/b = 0.1\)

33: \(t_t/b = 0.025\)
3.6 Refined Analysis

NASTRAN was selected for use as the refined analysis method for the STAEBL procedure. It is used regularly to determine steady stresses in solid titanium fan blades for flight cycle life evaluation. A plate element blade model is analyzed in this application. NASTRAN is also used to calculate the vibratory characteristics of composite material fan blades. Equivalent anisotropic material properties are calculated for each finite element using thin laminated plate theory.

Engineering effort in setting up and analyzing solid titanium and composite material blades is minimized through the use of pre-and post-processors. Available processing capabilities include:

- An airfoil pre-processor which generates a NASTRAN plate model of a blade from the airfoil coordinate descriptions,
- A laminate pre-processor which calculates the laminate effective stiffness matrices for each finite element and outputs them in a form acceptable to NASTRAN as input data,
- A NASTRAN module to calculate laminate strains from element stresses,
- A post-processor to calculate ply stresses from NASTRAN element stresses.

The flight cycle life and vibratory characteristics of the hollow titanium Energy Efficient Engine fan blade were also evaluated using NASTRAN analysis. But separate models of the concave and convex airfoil walls were employed to verify that a sufficient number of ribs were provided. This made the analysis cumbersome and impractical for use in the STAEBL procedure.

It was proposed that hollow blades could be analyzed using a laminated plate model with the central lamina having zero stiffness and density. The Energy Efficient Engine was re-analyzed to substantiate this approach. The airfoil breakup was chosen so that internal ribs are coincident with loci of nodal points. The rib properties are represented by beam elements connecting these nodes. Vibration analysis of the lamination model agrees very well with the more cumbersome original analysis as shown in Figure 3.6-1. The breakup in the region of the airfoil root and solid-to-hollow transition was refined and the centrifugal stresses presented in Figures 3.6-2 and 3.6-3 were obtained. These stresses are consistent with those predicted by the original design analysis.

The procedure for predicting supersonic flutter of fan stages can evaluate the stability of structural modes which are defined by finite plate element analysis. This ability combines the chordwise bending degree of freedom with the flap and twist degrees of freedom included in approximate beam blade analysis. The use of the lamination model for blade modal analysis makes it practical to use the expanded flutter prediction procedure for refined analysis. Results of flutter analyses of the Energy Efficient Engine blade are compared in Figure 3.6-4. The original design flutter analysis reduced the plate element blade mode shapes to equivalent beam blade modes and concluded
that the blade would not flutter. The refined analysis supports this conclusion and provides a more accurate technique to evaluate thinner airfoils which are likely to result from structural tailoring.

![NASTRAN PLATE ANALYSIS](image)

Figure 3.6-1 Refined Analysis of the Energy Efficient Engine Hollow Fan Blade
Figure 3.6-2  Energy Efficient Engine Hollow Fan Blade Airfoil Root Stress Predicted by Refined Analysis

Figure 3.6-3  Energy Efficient Engine Hollow Fan Blade Internal Surface Stress at Solid-to-Hollow Transition Predicted by Refined Analysis
Figure 3.6-4  Refined Analysis of the Energy Efficient Engine Hollow Fan Blade Supports the Conclusion that It Would Not Flutter
4.0 RESULTS

4.1 INTRODUCTION

Structural Tailoring of Engine Blades (STAEBL) was demonstrated by tailoring two shroudless fan blades: 1) a hollow titanium blade with metal matrix composite inlays, and 2) a solid blade made from superhybrid composites. These blades were alternate constructions of the Energy Efficient Engine shroudless fan blade designed by Pratt & Whitney Aircraft as part of NASA Contract NAS3-20646. As a consequence of demonstration blade selection, several features of the Energy Efficient Engine were taken into account in the STAEBL demonstration. Those features and other design considerations will be presented in Section 4.2 while a description of the actual tailored blades will be presented in Section 4.3.

4.2 DEMONSTRATION CONSIDERATIONS

4.2.1 Aerodynamic Design

Engine configuration-dependent parameters of the Energy Efficient Engine fan component design were held constant while others were allowed to vary. Those parameters that were held constant include: the airflow through the fan component of the Energy Efficient Engine, which is 622.7 kg/sec (1372.8 lbm/sec); the tip speed of 456 m/sec (1496 ft/sec); and the average pressure ratio of 1.7. The hub/tip radius ratio of 0.34, the tip/root chordal taper of 1.46, and associated airfoil stagger and camber angles were also held constant during structural tailoring.

The basic blade chord was allowed to vary and the number of blades was changed inversely with chord to maintain constant aerodynamic gap/chord ratio at any radial location. The spanwise distribution of airfoil section maximum thickness was allowed to vary but the ratio of thicknesses at any two fraction of chord positions was held approximately constant. Maximum thickness was never allowed to exceed ten percent of local chord.

4.2.2 Objective Function

The procedure used to establish an objective function for the STAEBL demonstration was modeled after the economic assessment procedure used to guide the design of the Energy Efficient Engine. The Energy Efficient Engine design process was guided by an economic performance assessment of a study airplane defined in the Component Development and Integration Program phase of the contract. The overall airplane characteristics are shown in Table 4.2-I. The aircraft is designed for the full specified payload and range, but the economic analysis is conducted for the typical mission payload and range.

The economic analysis evaluates the effect of changes in engine weight, maintenance cost and first cost against the changes in the aircraft takeoff gross weight and fuel burned to assess the economic effect on the airline. The basis for this analysis is a well developed trade factor technique derived from consideration of airplane aerodynamics, flight mechanisms, propulsion system integration, and weight estimation. The changes in airplane takeoff
gross weight reflect a "rubber" airplane analysis, i.e. improvements to the engine configuration will result in further improvements to the airplane configuration. For example, a concept which reduces engine weight will result in a fuel savings which, in turn, further reduces aircraft weight and aircraft structural component weight, permitting reductions in wing size and engine thrust requirements. Consequently, the initial engine weight benefit "snowballs" in its impact on the aircraft benefit.

TABLE 4.2-I
ENERGY EFFICIENT ENGINE STUDY AIRPLANE CHARACTERISTICS

<table>
<thead>
<tr>
<th></th>
<th>Domestic Trijet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Engines</td>
<td>3</td>
</tr>
<tr>
<td>Range - kilometers (nautical miles)</td>
<td></td>
</tr>
<tr>
<td>Design mission</td>
<td>5550 (3000)</td>
</tr>
<tr>
<td>Typical mission</td>
<td>1295 (700)</td>
</tr>
<tr>
<td>Passengers</td>
<td></td>
</tr>
<tr>
<td>Design mission</td>
<td>400</td>
</tr>
<tr>
<td>Typical mission</td>
<td>55% load factor</td>
</tr>
<tr>
<td>Design takeoff gross weight - kilograms (pounds)</td>
<td>231,000 (510,000)</td>
</tr>
<tr>
<td>Cruise Mach number</td>
<td>0.80</td>
</tr>
<tr>
<td>Initial Cruise Altitude - meters (feet)</td>
<td>10,700 (35,000)</td>
</tr>
<tr>
<td>Takeoff field length - meters (feet)</td>
<td>2440 (8,000)</td>
</tr>
</tbody>
</table>

The life cycle ownership costs determined in this analysis are expressed as direct operating costs plus interest (DOC + I). A trade factor technique derived from considerations of total airline economics provides the basis for this analysis, and includes crew cost, fuel cost, airframe and engine depreciation, airframe and engine maintenance cost, insurance cost, and overhead cost. These trade factors are applied to the specific engine and airplane for which each engine change has been determined. Trade factors for changes in engine weight, maintenance cost, and first cost are applied independently to determine the effect of each engine change on a given economic parameter. Individual effects are then combined to evaluate the total effect of the advanced concept on that parameter. DOC + I is an extension of DOC in that it includes the "cost of money". In other words, it includes an expected return to the airline for their investment in the aircraft/engine system. DOC + I is an appropriate substitution for ROI (return on investment) and includes all of the engine related terms in ROI, and is, therefore, an appropriate parameter for evaluating the effect of engine changes on an airline's economics.
The ground rules for the airline economic model are shown in Table 4.2-II for the Energy Efficient Engine. The 15 percent cost of capital shown is the "interest" in the parameter "DOC + Interest" (DOC + I).

TABLE 4.2-II

ENERGY EFFICIENT ENGINE AIRLINE ECONOMIC MODEL

- 1977 Dollars
- $0.105/Liter ($0.40/Gallon) Domestic Fuel Cost
- 0.5% Per Year Insurance
- Spares - 5% Airframe, 30% Engine
- Maintenance - Labor Rate = $9.70/Hr, Burden = 200%
- Airplane Price - P&W Equation
- Depreciation - 15 Year Straight Line to 10% Residual
- Non-Revenue Flying - 2% Factor on Fuel and Maintenance
- Ground Time - 15 minutes (Domestic) - 20 minutes (International)
- Cost of Capital = 15%

The resulting function is:

\[ \Delta \% \ (DOC + I) = 1.14 \times \left( \frac{\Delta \text{ engine wt (kg)}}{1000} \right) + 0.54 \times \left( \frac{\Delta \text{ engine cost ($)}}{100,000} \right) + 0.80 \times \left( \frac{\Delta \text{ engine maint. cost ($/EFH)}}{10} \right) \]

This function was redefined in terms of blade design variables to provide the objective function for the STAEBL procedure demonstration.

4.2.2.1 Engine Weight

Individual airfoil weight is an output of the subroutine which was assembled to modify existing approximate analyses for application to composite blades and it is a function of all design variables. Total airfoil weight is the product of individual airfoil weight and number of blades, which is inversely dependent upon the blade chord design variable. Disk and containment case design requirements relate to individual and/or total airfoil weight.
A study was conducted to evaluate these relationships using preliminary design procedures which are regularly applied in estimating engine weight. Three different blade constructions were assumed and blade chord was varied over a range of relevant aspect ratios. Individual foil weights and fan system weight were calculated. Cross-plotting the results generated the unexpected conclusion that, within engineering accuracy, fan system weight is a simple function of individual airfoil weight. This function is depicted in Figure 4.2-1.

Figure 4.2-1 Fan System Weight Is a Simple Function of Individual Airfoil Weight
4.2.2.2 Engine Cost

Information from reference 3 was used as a basis for estimating costs of individual fan blades. The hollow blade is made from laminations of titanium and borsic-titanium sheet with hollow cavities produced using leachable iron cores. The stacked laminates are canned, hipped and isothermally forged to shape. The superhybrid blade is made from plies of graphite-epoxy, boron-aluminum, and titanium with adhesive ply bonding. Stacked plies are vacuum debulked and molded. Materials cost depends on amount of each component material which is related to the design variables by the composite blade approximate analysis subroutine. Labor cost depends on blade size as indicated by the design variables root chord and root thickness. Total blade cost is the product of individual blade cost and the number of blades which is inversely proportional to the blade chord design variable. The change in engine cost for changes in design variables is the change in total blade cost plus an experience-based assessment of costs of related structures which reduces to cost per unit engine length multiplied by change in blade chord.

4.2.2.3 Maintenance Cost

Engine maintenance histories show that the dominant factor in fan maintenance cost is the number of blades which must be discarded after an ingestion event because they are damaged beyond repair. Service experience provides a definition of the frequency of major ingestion events, the percentage of blades damaged by an event and the solid titanium blade repair/scrap ratio. A hollow blade is expected to have a lower ratio because damage in or near the cavity is not repairable. The controlling parameter is expected to be the design variable distance from the airfoil leading edge to the forward boundary of the cavity. Experience and judgement have been applied to generate the definition, shown in Figure 4.2-2, of hollow blade scrap life from known end points. Life is increased in proportion with swept flowpath area when the cavity inner bound is outboard of the airfoil root. Superhybrid blades are assumed to be unrepairable. Total maintenance cost per flight hour is equal to blade set fabrication cost divided by scrap life.

4.2.3 Constraints on Behavior Variables

Constraints were imposed during tailoring to ensure that the optimized designs would not be prone to failure from high frequency fatigue, would endure repeated cyclic acceleration to takeoff power, and could inject seagulls without experiencing damage which would cause engine shutdown. These constraints were consistent with criteria observed during the Energy Efficient Engine fan component design.

4.2.3.1 High Frequency Fatigue

High frequency fatigue in resonant vibration was avoided by tuning natural frequencies so they did not coincide with the frequencies of strong excitations. In the Energy Efficient Engine fan component design, the critical excitation frequencies were recognized to be the first through fourth integer order multiples of rotor speed when the engine is operating at any power between minimum cruise and maximum takeoff. High frequency fatigue in flutter
is avoided by deriving a design which has positive predicted aerodynamic damping in any natural mode of vibration at airflow conditions corresponding to maximum takeoff power. In the Energy Efficient Engine fan component design, minimum stability for any feasible interblade phase angle was evaluated for vibration in each of the first three modes of individual blade vibration.

Figure 4.2-2 Full Span Hollow Fan Blade Life Is Derived from Solid Blade Experience Limits

4.2.3.2 Flight Cycle Fatigue

Flight cycle fatigue is evaluated by assuming that operating stresses cycle between zero at shutdown and maximum at standard day takeoff power. In the Energy Efficient Engine fan component design, the operating stress was assumed to be that imposed by centrifugal force on a radially stacked blade because bending caused by tilt is equal and opposite to bending caused by air load at the critical root section.
4.2.3.3 Bird Ingestion

Two types of damage are recognized in designing a fan blade to tolerate bird ingestion: 1) loss of an excessive quantity of local leading edge material, or 2) fracture of the airfoil root. Both of these types of failures of the Energy Efficient Engine fan blade were evaluated using an ingestion stress parameter based on linear elastic analysis calibrated by application to many ingestion experiences.

4.3 STAEBL PROCEDURE DEMONSTRATION

4.3.1 Reference Blade

The Energy Efficient Engine shroudless fan blade design provides a logical baseline for evaluating the blade designs defined by the tailoring demonstration. It is an all titanium blade which is hollow over the outer two-thirds of the airfoil span. The cavity wall thickness is tapered in the spanwise and chordwise directions. A dynamically equivalent constant wall thickness hollow titanium blade with the same external shape and cavity planform location was derived. It provided a reference design consistent with the composite material blade designs which could be defined by the parameters that were varied in the demonstration.

The reference blade dynamic characteristics are equivalent to those of the Energy Efficient Engine shroudless fan blade design. Therefore, the dynamic resonance and flutter constraints observed in designing that blade were directly applicable to tailoring demonstration blades.

The reference blade stress levels are similar to, but not identical with, those in the Energy Efficient Engine blade. The demonstration was made most meaningful by setting limits equals to reference blade levels multiplied by Energy Efficient Engine blade allowable-to-actual ratios. Realistic stress concentrations and local material conditions are automatically covered by this approach.

The local bird ingestion stress parameter was limited to the Energy Efficient Engine blade design allowable level. The root dynamic stress response to bird ingestion was a new uncalibrated analysis. The limit was set to be in proportion with the Energy Efficient Engine blade root stress parameter margin of safety. This limit is much higher than material static strength.

The Tsai-Wu parameter for each material is calculated from ratios of directional stress to directional strength. Therefore, a search for the maximum value found in any material yields a single critical level which is compared with a material independent limit level. The local bird ingestion stress parameter is evaluated in a consistent fashion by ratioing each layer's contribution in proportion with chordwise directional strength.

Preliminary studies had shown that resonance constraints, which create disjoint feasible regions, could lead to tailoring a locally optimum blade design which is not globally optimum. Therefore, the initial optimization of each of the two types of blades was conducted without resonance constraints
and then, starting with the partially tailored design, was finalized with resonance constraints added. Both of the resultant tailored blades had the same basic resonance characteristics as the reference hollow titanium blade (Figure 4.3-1), increasing confidence that global optima were achieved.

Figure 4.3-1 Reference Blade Resonance Diagram
4.3.2 Tailored Blades

Each tailoring demonstration started with a configuration that was similar to the reference blade (2.5 aspect ratio based on root chord and average length). Each demonstration consisted of two passes through automated exact optimization and subsequent separate refined analyses. During the automated optimization, seventy-four percent of the central processing unit time was spent in approximate structural analysis and twenty-six percent of the time was spent in unsteady aerodynamic (flutter) analysis. The time spent on optimization calculations was negligible. The four optimization efforts successfully identified feasible designs as evaluated by approximate analyses. The first refined analyses generated correction factors which were applied to the approximate analyses for reoptimization. The particular demonstrations are described in the following paragraphs.

4.3.2.1 Hollow Blade with Composite Inlay

In Table 4.3-I, data describing the tailoring of a hollow titanium blade with a borsic titanium inlay in the area of the cavity are compared with the reference blade. Thirteen geometric quantities were varied in this demonstration. Chord at every station was changed in proportion with root chord. The properties which were assumed for borsic-titanium are compared with titanium in Table 4.3-II.

**TABLE 4.3-I**

<table>
<thead>
<tr>
<th>Design Variables</th>
<th>Reference Blade</th>
<th>Limits</th>
<th>First Tailoring</th>
<th>Second Tailoring</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root Chord - cm (inches)</td>
<td>23.2 (9.12)</td>
<td>-</td>
<td>16.9 (6.64)</td>
<td>19.0 (7.46)</td>
</tr>
<tr>
<td>Thickness/Chord</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Root</td>
<td>0.096</td>
<td>0.075</td>
<td>0.077</td>
<td>0.084</td>
</tr>
<tr>
<td>25%</td>
<td>0.075</td>
<td>0.025≤</td>
<td>0.089</td>
<td>0.092</td>
</tr>
<tr>
<td>50%</td>
<td>0.055</td>
<td>t/b≤0.100</td>
<td>0.084</td>
<td>0.045</td>
</tr>
<tr>
<td>75%</td>
<td>0.033</td>
<td></td>
<td>0.038</td>
<td>0.025</td>
</tr>
<tr>
<td>Tip</td>
<td>0.025</td>
<td></td>
<td>0.033</td>
<td>0.028</td>
</tr>
<tr>
<td>Cavity Boundaries - cm (inches)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>From Leading Edge</td>
<td>2.20 (1.00)</td>
<td>≥1.27</td>
<td>3.12 (1.23)</td>
<td>1.37 (0.54)</td>
</tr>
<tr>
<td>From Trailing Edge</td>
<td>2.20 (1.00)</td>
<td>≥1.27</td>
<td>2.48 (0.98)</td>
<td>3.48 (1.37)</td>
</tr>
<tr>
<td>From Root</td>
<td>25.2 (9.90)</td>
<td>≥5.46</td>
<td>12.97 (5.10)</td>
<td>7.52 (2.96)</td>
</tr>
<tr>
<td>From Tip</td>
<td>0.635 (0.25)</td>
<td>≥0.635</td>
<td>0.635 (0.25)</td>
<td>0.635 (0.25)</td>
</tr>
</tbody>
</table>
TABLE 4.3-1 (Continued)

<table>
<thead>
<tr>
<th>Reference Blade Limits</th>
<th>First Tailoring</th>
<th>Second Tailoring</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cavity Wall - cm (inches)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Titanium Skin</td>
<td>0.305 ≥ 0.076 (0.120) (0.030)</td>
<td>0.076 (0.030)</td>
</tr>
<tr>
<td>Composite</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Borsic Fiber Angle - rad</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Blade Weight - kilograms (pounds)</td>
<td>8.7 (19.2)</td>
<td>-</td>
</tr>
<tr>
<td>Objective Function - Δ (%DOC+I)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Engine Weight</td>
<td>-0.42</td>
<td>-0.33</td>
</tr>
<tr>
<td>Engine Cost</td>
<td>-0.14</td>
<td>-0.15</td>
</tr>
<tr>
<td>Maintenance Cost</td>
<td>-0.09</td>
<td>+0.03</td>
</tr>
<tr>
<td>Total</td>
<td>-0.65</td>
<td>-0.45</td>
</tr>
</tbody>
</table>

Constraints

| Resonance Margin - Δfn/fn |                  |                  |
| First Mode 2E             | 0.08 ≥ 0.05      | 0.34             | 0.28             |
| Second Mode 3E            | 0.05 ≥ 0.05      | -0.02            | -0.03            |
| Second Mode 4E            | 0.08 ≥ 0.05      | 0.16             | 0.19             |
| Third Mode 4E             | 0.05 ≥ 0.05      | 0.01             | 0.06             |

| Flutter - Log Decrement  |                  |                  |
| First Mode               | 0.006 ≥ 0        | 0.004            | 0.005            |
| Second Mode              | 0.001 ≥ 0        | 0.011            | 0.001            |
| Third Mode               | 0.007 ≥ 0        | 0.001            | 0.002            |

| Bird Ingestion           |                  |                  |
| Local Stress Parameter   | 52.0 ≤ 80.0      | 66.0             | 48.0             |
| Root Tsai-Wu             | 0.9 ≤ 10.0       | n.g.             | 2.5              |

| Steady Stress - Tsai-Wu  |                  |                  |
| Root Edge                | 0.20 ≥ 0.22      | 0.32             | 0.08             |
| Root Local               | 0.25 ≥ 0.29      | 0.32             | 0.21             |
| Cavity Local             | 0.26 ≥ 0.31      | 0.32             | 0.08             |
TABLE 4.3-I

MATERIALS PROPERTIES (Divided by Titanium Value)

<table>
<thead>
<tr>
<th></th>
<th>Borsic-Titanium</th>
<th>Adhesively Laminated Boron-Aluminum</th>
<th>Graphite Epoxy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elastic Modulus</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fiber Direction</td>
<td>2.06</td>
<td>1.73</td>
<td>1.15</td>
</tr>
<tr>
<td>Normal Direction</td>
<td>1.80</td>
<td>1.11</td>
<td>0.10</td>
</tr>
<tr>
<td>Shear</td>
<td>3.70</td>
<td>1.41</td>
<td>0.14</td>
</tr>
<tr>
<td>Density</td>
<td>0.81</td>
<td>0.55</td>
<td>0.35</td>
</tr>
<tr>
<td>Strength</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fiber Tension</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Direction</td>
<td>1.54</td>
<td>1.14</td>
<td>1.45</td>
</tr>
<tr>
<td>Fiber Compression</td>
<td></td>
<td>1.54</td>
<td>1.45</td>
</tr>
<tr>
<td>Normal Tension</td>
<td></td>
<td>0.45</td>
<td>0.12</td>
</tr>
<tr>
<td>Normal Compression</td>
<td></td>
<td>0.45</td>
<td>0.07</td>
</tr>
<tr>
<td>Direction</td>
<td>0.45</td>
<td>0.26</td>
<td>0.23</td>
</tr>
<tr>
<td>Shear</td>
<td>0.47</td>
<td>0.20</td>
<td>0.16</td>
</tr>
</tbody>
</table>

Thickness/chord distribution was defined by straight lines connecting independent levels at five stations. A lower limit was imposed to prevent chordwise bending flutter and an upper limit to avoid aerodynamic performance penalties. The cavity quadrilateral planform boundaries were parallel to adjacent airfoil planform extremities and the cavity walls were laminates of titanium and borsic-titanium. The titanium was of uniform thickness and was located at the airfoil surface. The borsic-titanium was also of uniform thickness and it was located between the titanium surface and the internal cavity. Design variables were the distances from the cavity boundaries to the airfoil extremities and the thickness of the individual materials which comprised the cavity wall. Limits were imposed based on anticipated maintenance penalties in excess of those recognized by the objective function (the root limit ensured that the supporting attachment would be solid titanium). Borsic fiber angle was the final variable.

The first tailoring converged in thirteen iterations. The root Tsai-Wu due to bird ingestion was observed to have varied in an illogical manner which was traced to an error in the attachment section scaling with root chord and thickness. No other parameters were seriously effected. Refined analysis results, which were presented in Table 4.3-I, showed that approximate analysis error had permitted violation of resonance limits, that the second mode frequency be at least five percent higher than three times rotor speed and that the third mode frequency be at least five percent higher than four times rotor speed, both at maximum speed. Root steady stress limits were also violated at the edge and away from the edge.
The attachment scaling logic was corrected and correction factors from the refined analysis of the first tailored blade were applied to the approximate analyses predicted frequencies and steady stresses. A second tailoring, starting with the configuration defined by the first tailoring, converged in ten iterations. Refined analysis showed that the use of correction factors had eliminated all but one limit violation. Comparison of approximate and refined analysis predicted frequencies, Table 4.3-III, showed that the approximate analysis error in the second mode frequency prediction was erratic. Tailoring was terminated since there was no reason to expect that continued iteration would lead to convergence of corrected approximate and refined frequency analyses or to a more realistic design. Central processing unit time averaged 5.9 minutes per iteration.

**TABLE 4.3-III**

<table>
<thead>
<tr>
<th>Mode</th>
<th>First Tailoring</th>
<th>Second Tailoring</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Approximate Analysis</td>
<td>Refined Analysis</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>88</td>
</tr>
<tr>
<td>2</td>
<td>187</td>
<td>208</td>
</tr>
<tr>
<td>3</td>
<td>300</td>
<td>287</td>
</tr>
</tbody>
</table>

*Without correction

The weight of the individual tailored blade is less than half the reference blade weight and the optimized objective function indicates that total weight and cost are both reduced even though more blades are required to hold aerodynamic gap/chord. Maintenance cost penalty is minimal. These benefits were achieved by the use of composite material, chord reduction and extension of the cavity toward the airfoil root. The limiting constraints appear to be resonance and flutter fine tuned by the location of the transition from the thick root to the thin tip and by the chordwise location of the cavity.

### 4.3.2.2 Superhybrid Composite Blade

Table 4.3-IV presents data describing the tailoring of a solid blade made of superhybrid composite material. Eleven quantities were varied in this demonstration. Chord and thickness were defined as they were in the hollow blade optimization. The blade was sheathed with a uniform thickness titanium skin. The internal construction consisted of a uniform thickness central titanium ply with the balance being composite material, boron-aluminum external to graphite epoxy. The titanium thicknesses and the constant fraction of composite which was boron-aluminum were design variables. A limit was imposed on the titanium skin thickness to avoid a maintenance penalty in
excess of that recognized by the objective function. Boron and graphite fiber angles were also varied. The properties which were assumed for the composite materials were compared with titanium in Table 4.3-II.

TABLE 4.3-IV
SUPERHYBRID COMPOSITE BLADE

<table>
<thead>
<tr>
<th>Design Variables</th>
<th>Limits</th>
<th>First Tailoring</th>
<th>Second Tailoring</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root Chord - cm (inches)</td>
<td>-</td>
<td>19.12</td>
<td>22.2</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>(7.53)</td>
<td>(8.32)</td>
</tr>
<tr>
<td>Thickness/Chord</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Root</td>
<td>0.100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25%</td>
<td>0.099</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50%</td>
<td>0.049</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75%</td>
<td>0.026</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tip</td>
<td>0.026</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Titanium Thickness - cm (inches)</td>
<td>≥ 0.076</td>
<td>0.147</td>
<td>0.142</td>
</tr>
<tr>
<td></td>
<td>(0.030)</td>
<td>(0.058)</td>
<td>(0.056)</td>
</tr>
<tr>
<td>Skin</td>
<td>0</td>
<td>0.066</td>
<td>0.026</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0 (0.026)</td>
<td></td>
</tr>
<tr>
<td>B/Al/Composite</td>
<td>-</td>
<td>0.828</td>
<td>0.835</td>
</tr>
<tr>
<td>Boron Fiber Angle - radians</td>
<td>-</td>
<td>0.002</td>
<td>0.002</td>
</tr>
<tr>
<td>Graphite Fiber Angle - radians</td>
<td>-</td>
<td>0.093</td>
<td>0.093</td>
</tr>
<tr>
<td>Blade Weight - kilograms (pounds)</td>
<td>-</td>
<td>4.13</td>
<td>5.49</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>(9.1)</td>
<td>(12.1)</td>
</tr>
<tr>
<td>Objective Function - Δ(%DOC+I)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Engine Weight</td>
<td>-0.34</td>
<td></td>
<td>-0.23</td>
</tr>
<tr>
<td>Engine Cost</td>
<td>-0.23</td>
<td></td>
<td>-0.18</td>
</tr>
<tr>
<td>Maintenance Cost</td>
<td></td>
<td></td>
<td>+0.05</td>
</tr>
<tr>
<td>Total</td>
<td>-0.57</td>
<td></td>
<td>-0.36</td>
</tr>
<tr>
<td>Constraints - Δfn/fn</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resonance Margin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Mode 2E</td>
<td>0.05</td>
<td>0.19</td>
<td>0.15</td>
</tr>
<tr>
<td>Second Mode 3E</td>
<td>0.05</td>
<td>0.05</td>
<td>0.12</td>
</tr>
<tr>
<td>Second Mode 4E</td>
<td>0.05</td>
<td>0.15</td>
<td>0.09</td>
</tr>
<tr>
<td>Third Mode 4E</td>
<td>0.05</td>
<td>0.03</td>
<td>0.12</td>
</tr>
</tbody>
</table>
A review of tailoring progress after five iterations exposed undue influence of the leading edge stress limit applied to the forward extremity of the composite core material. Since this limit is based on the particular characteristics of titanium material, it was subsequently applied only to the titanium skin. The local limit was applied to the composite material at any location.

The first tailoring converged after ten additional iterations. The refined analysis results, which were presented in Table 4.3-IV, showed that approximate analysis error had permitted violation of the third mode resonance limit and the first mode flutter stability. A second tailoring with correction factors converged in thirteen iterations. The differences between the approximate and refined analysis predicted frequencies, Table 4.3-V, were less than they were in the hollow blade demonstration and the refined analysis showed all limits to be satisfied. Central processing unit time averaged 5.1 minutes per iteration.

The tailored superhybrid composite blade weight and optimized objective function are not as beneficial as those provided by the tailored hollow blade with composite inlays, but they represent considerable improvement over the reference blade. Chord reduction and the boron aluminum portion of the composite material core appear to have provided the benefits achieved. Flutter stability appears to be the dominant constraint. Local bird ingestion capacity established the thickness of the titanium skin.
<table>
<thead>
<tr>
<th>Mode</th>
<th>First Tailoring</th>
<th>Second Tailoring</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Approximate Analysis</td>
<td>Refined Analysis</td>
</tr>
<tr>
<td>1</td>
<td>106</td>
<td>104</td>
</tr>
<tr>
<td>2</td>
<td>229</td>
<td>218</td>
</tr>
<tr>
<td>3</td>
<td>302</td>
<td>289</td>
</tr>
</tbody>
</table>

*Uncorrected
5.0 CONCLUSIONS AND RECOMMENDATIONS

Mathematical optimization applied to shroudless fan blade structural tailoring has been demonstrated to be a very powerful automated design procedure. It provides the capacity to simultaneously evaluate many design variables to optimize a comprehensive objective function while observing numerous design constraints. The fan blade application of this study is relatively novel because dynamic design constraints preponderate and, since critical excitations can be avoided by making structural frequencies higher or lower, disjoint feasible regions are a practical possibility. This did not introduce limiting problems.

The composite blade tailoring application demonstrated the capacity of the STAEBL procedure to select values for a large number of design variables because the fabricated internal constructions could be changed in many ways without affecting aerodynamic performance. But checkout studies also demonstrated that the procedure is a useful tool for homogeneous material blade tailoring. The various natural modes of vibration are sensitive to spanwise distribution of airfoil thickness to a level of refinement that can only be defined by several variables. Even the most experienced design analyst would only be able to find an approximation of the best distribution and, in some instances, he might not find any acceptable distribution when the feasible region is small.

Unresolved discrepancies between the final corrected approximate analysis of one of the demonstration blades and the subsequent refined analysis exposed an apparent limitation of the STAEBL procedure. It is suspected that the approximations which are inherent in simple beam theory limit accuracy when the procedure is applied to a composite material blade and, to a lesser degree, when it is applied to a shroudless fan blade of any material. But the fact that a finite beam element approach was used to derive a practical composite material blade analysis indicates that a special purpose plate element blade analysis could also be used in a tailoring procedure. It would eliminate the problem by improving consistency between the approximate and refined analyses.

Spanwise thickness distribution has been shown to be an important characteristic of an airfoil dynamic design. Chordwise distribution within aerodynamically allowable limits may also be significant but it can not be pursued effectively because of the inexact nature of the stress parameter used to evaluate local edge damage caused by bird ingestion. An existing nonlinear finite element analysis has the desired technical capability but it is too cumbersome for inclusion in a tailoring procedure. It is feasible that the critical features of the local response simulation could be incorporated in a fast linear analysis. It is recommended that the STAEBL procedure be revised to include a special purpose plate element for airfoil stress and vibration analyses. It is also recommended that the STAEBL procedure be revised to include another special purpose plate element for analysis of linearized large deflection local edge response to foreign object damage impact.
APPENDIX A
BLADE OPTIMIZATION PROGRAM ORGANIZATION

The COPES/CONMIN program forms the basis for the blade optimization program. COPES/CONMIN, through subroutine ANALIZ, calls the blade vibration, flutter, and foreign object damage analyses for evaluation of the constraints and the objective function of the design vector.

Table A-I shows a tabulation of the 74 subroutines developed or acquired under this contract, along with a list of other routines called by each routine, and a brief statement of the function of each subroutine. Table A-I, then, serves as an effective flow chart, detailing the precise program organization. On Table A-II are listed the Pratt & Whitney Aircraft proprietary routines and/or analysis systems which were used but not developed in the current program. The curve fit and matrix inversion routines may be replaced by standard routines from mathematical packages. MNW137 does the frequency and mode shape determination for the STAEBL generated beam stiffness, using a determinant search technique. MNT983 is the driving routine for a proprietary supersonic flutter analysis.
<table>
<thead>
<tr>
<th>Routine</th>
<th>External References</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Program</td>
<td>MYTIME COPE05</td>
<td>Main Program for COPES/CONMIN Constrained Optimization Program</td>
</tr>
<tr>
<td></td>
<td>COPE01 COPE06</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ANALIZ COPE07</td>
<td></td>
</tr>
<tr>
<td></td>
<td>COPE02 COPE09</td>
<td></td>
</tr>
<tr>
<td></td>
<td>COPE18 COPE14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>COPE04</td>
<td></td>
</tr>
<tr>
<td>ANALIZ</td>
<td>MYTIME MNW137</td>
<td>Constraint Calculation for COPES/CONMIN</td>
</tr>
<tr>
<td></td>
<td>RDW137 RSFUNC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RDDATA OBJTV</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TMAX ITT983</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WTW137 WTT983</td>
<td></td>
</tr>
<tr>
<td>BBIP</td>
<td>TSAIWV MATPRN</td>
<td>Multi-Mode Root Bird Ingestion Stress</td>
</tr>
<tr>
<td>BCOORD</td>
<td>MATPRN MATMPY</td>
<td>Beam Local Coordinate Systems</td>
</tr>
<tr>
<td>BEAM</td>
<td>MATPRN</td>
<td>Beam Stiffness Calculation</td>
</tr>
<tr>
<td>BIRDF</td>
<td>MATPRN</td>
<td>Bird Impulse Calculation</td>
</tr>
<tr>
<td>CALCTH</td>
<td>MATPRN</td>
<td>Input Blade Thickness Calculation</td>
</tr>
<tr>
<td>CNMN01</td>
<td>CNM04</td>
<td>Gradient Information</td>
</tr>
<tr>
<td>CNMN02</td>
<td>CNM04</td>
<td>Descent Vector</td>
</tr>
<tr>
<td>CNMN03</td>
<td>CNM04</td>
<td>One-Dimensional Search</td>
</tr>
<tr>
<td>CNMN04</td>
<td>CNM04</td>
<td>One-Dimensional Function Minimum</td>
</tr>
<tr>
<td>CNMN05</td>
<td>CNM08</td>
<td>Search Direction</td>
</tr>
<tr>
<td>CNMN06</td>
<td>CNM07 CNM04</td>
<td>One-Dimensional Search</td>
</tr>
<tr>
<td>CNMN07</td>
<td>CNM06</td>
<td>Zero of I-D Function</td>
</tr>
<tr>
<td>CNMN08</td>
<td>CNM01 CNM03</td>
<td>Special Linear Optimization</td>
</tr>
<tr>
<td>CONMIN</td>
<td>CNM02 CNM05</td>
<td>Constrained or Unconstrained Function Minimization</td>
</tr>
<tr>
<td>Routine</td>
<td>External References</td>
<td>Function</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>COPE01</td>
<td>COPE08</td>
<td>Control Input for COPES</td>
</tr>
<tr>
<td>COPE02</td>
<td>CONMIN COPE03</td>
<td>Control Optimization</td>
</tr>
<tr>
<td>COPE03</td>
<td>ANALIZ</td>
<td>Buffer Between CONMIN and COPES</td>
</tr>
<tr>
<td>COPE04</td>
<td>ANALIZ COPE02</td>
<td>Sensitivity Information</td>
</tr>
<tr>
<td>COPE05</td>
<td></td>
<td>Print Sensitivity Information</td>
</tr>
<tr>
<td>COPE06</td>
<td></td>
<td>Two Variable Function Space</td>
</tr>
<tr>
<td>COPE07</td>
<td></td>
<td>Print Two Variable Function Space</td>
</tr>
<tr>
<td>COPE08</td>
<td></td>
<td>Convert Data Input</td>
</tr>
<tr>
<td>COPE09</td>
<td>COPE10 COPE12 COPE15</td>
<td>Approximate Optimization</td>
</tr>
<tr>
<td>COPE10</td>
<td>COPE11</td>
<td>Set-Up Taylor Expansion</td>
</tr>
<tr>
<td>COPE11</td>
<td></td>
<td>Nominal Design for Approx. Optimization</td>
</tr>
<tr>
<td>COPE12</td>
<td>COPE13</td>
<td>Least Squares Fit</td>
</tr>
<tr>
<td>COPE13</td>
<td></td>
<td>Least Squares Fit Evaluation</td>
</tr>
<tr>
<td>COPE14</td>
<td></td>
<td>Print Approximate Optimization Results</td>
</tr>
<tr>
<td>COPE15</td>
<td>COPE16 COPE17</td>
<td>Function Evaluation, Approx. Optimization</td>
</tr>
<tr>
<td>COPE16</td>
<td></td>
<td>Approximate Function Evaluation</td>
</tr>
<tr>
<td>COPE17</td>
<td></td>
<td>Gradient Evaluation, Approx. Optimization</td>
</tr>
<tr>
<td>COPE18</td>
<td></td>
<td>Print Optimization Results</td>
</tr>
<tr>
<td>CORD2R</td>
<td>BCOORD</td>
<td>Blade Coordinate Systems</td>
</tr>
<tr>
<td>ESTIFF</td>
<td>MATPRN</td>
<td>Beam Material Stiffness Matrix</td>
</tr>
<tr>
<td>EXNECK</td>
<td>MATPRN MATMPY</td>
<td>Rectangular Extended Airfoil Neck</td>
</tr>
<tr>
<td>FOD</td>
<td>MATPRN</td>
<td>Local Ingestion Parameter</td>
</tr>
<tr>
<td>FRPM</td>
<td>MATMPY MATPRN</td>
<td>Centrifugal Force Calculation</td>
</tr>
<tr>
<td>Routine</td>
<td>External References</td>
<td>Function</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>HOLLOW</td>
<td>MATPRN</td>
<td>Hollow Blade Lamination</td>
</tr>
<tr>
<td>HOLZER</td>
<td>THICK PARTN MATMPY</td>
<td>Holzer Transform Generation</td>
</tr>
<tr>
<td>INPUT</td>
<td>MATPRN</td>
<td>Material and Geometry Input</td>
</tr>
<tr>
<td>ITT983</td>
<td>WTT983 MNT983 NDT983</td>
<td>Flutter Nodal Diameter Iteration</td>
</tr>
<tr>
<td>KDGG</td>
<td>MATPRN</td>
<td>Differential Stiffness Calculation</td>
</tr>
<tr>
<td>LAMINA</td>
<td>MATPRN</td>
<td>Lamina Stress-Strain Relation</td>
</tr>
<tr>
<td>LAMINI</td>
<td>MATPRN</td>
<td>Composite Blade Layup</td>
</tr>
<tr>
<td>LOAD2</td>
<td>MATPRN MATMPY</td>
<td>Loads for Holzer Solution</td>
</tr>
<tr>
<td>MATADD</td>
<td></td>
<td>Perform Matrix Addition</td>
</tr>
<tr>
<td>MATMPY</td>
<td></td>
<td>Perform Matrix Multiplication</td>
</tr>
<tr>
<td>MATPRN</td>
<td></td>
<td>Matrix Print</td>
</tr>
<tr>
<td>MODMAS</td>
<td>SHAPE MATMPY</td>
<td>Modal Mass determination</td>
</tr>
<tr>
<td>MODSTR</td>
<td>SHAPE MODMAS ZSTRES</td>
<td>Modal Stress Driver</td>
</tr>
<tr>
<td>NDT983</td>
<td></td>
<td>Flutter Nodal Diameter Selection</td>
</tr>
<tr>
<td>OBJTV</td>
<td></td>
<td>Objective Function Calculation</td>
</tr>
<tr>
<td>PARTN</td>
<td>MATPRN</td>
<td>Matrix Partition Routine</td>
</tr>
<tr>
<td>PP1</td>
<td>MATPRN</td>
<td>Matrix Reordering</td>
</tr>
<tr>
<td>RBE2</td>
<td>MATMPY MATPRN</td>
<td>Rigid Body Coupling Matrices</td>
</tr>
<tr>
<td>RDDATA</td>
<td></td>
<td>Optimization Input Information</td>
</tr>
<tr>
<td>RDT983</td>
<td></td>
<td>Flutter Input</td>
</tr>
<tr>
<td>RDW137</td>
<td></td>
<td>Blade Model Input</td>
</tr>
</tbody>
</table>

TABLE A-I (Cont'd.)
<table>
<thead>
<tr>
<th>Routine</th>
<th>External References</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSFUNC</td>
<td></td>
<td>resonance function</td>
</tr>
<tr>
<td>SECTN</td>
<td>BCCORD MATPRN ESTIFF BEAM</td>
<td>section stiffness matrix</td>
</tr>
<tr>
<td>SHAPE</td>
<td>PPI MATPRN</td>
<td>eigenvector transformation</td>
</tr>
<tr>
<td>SMULT</td>
<td>MATPRN MATPRN</td>
<td>holzer string multiplication</td>
</tr>
<tr>
<td>STABEL</td>
<td>INPUT CORDZR LAMINA PPI HOLZER STATIC</td>
<td>stacked beam analysis driver (versions differ for hollow and superhybrid blades)</td>
</tr>
<tr>
<td>STAEBL</td>
<td>PBMFIT CUBIC LAMINI</td>
<td>blade interpolation for approximate analysis (versions differ for hollow and superhybrid blades)</td>
</tr>
<tr>
<td>STATIC</td>
<td>MATPRN PARTN MATPRN MATMPY</td>
<td>blade static solution</td>
</tr>
<tr>
<td>THICK</td>
<td></td>
<td>layer thickness evaluation</td>
</tr>
<tr>
<td>TMAX</td>
<td></td>
<td>section maximum thickness</td>
</tr>
<tr>
<td>TSAIWU</td>
<td>MATPRN</td>
<td>tsai-wu failure criterion</td>
</tr>
<tr>
<td>WARP</td>
<td>MATPRN</td>
<td>thin section warping</td>
</tr>
<tr>
<td>WTT983</td>
<td></td>
<td>flutter output routine</td>
</tr>
<tr>
<td>WTW137</td>
<td></td>
<td>blade analysis output</td>
</tr>
<tr>
<td>ZSTRES</td>
<td>MATPRN MATMPY</td>
<td>composite layer stress</td>
</tr>
<tr>
<td>Routine</td>
<td>External References</td>
<td>Function</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CUBIC</td>
<td></td>
<td>Function Evaluation for Parametric Beam Fit</td>
</tr>
<tr>
<td>MINV</td>
<td></td>
<td>Matrix Inversion by Gaussian Elimination</td>
</tr>
<tr>
<td>MNT983</td>
<td></td>
<td>Pratt &amp; Whitney Aircraft Flutter Analysis System</td>
</tr>
<tr>
<td>MNW137</td>
<td>STABEL</td>
<td>Pratt &amp; Whitney Aircraft Beam Analysis System, Modified Through STABEL Analysis Package for Hollow and Composite Blades</td>
</tr>
<tr>
<td>MYTIME</td>
<td></td>
<td>Clock Times, Used to Bookkeep Analysis Time Consumption</td>
</tr>
<tr>
<td>PBMFIT</td>
<td></td>
<td>Parametric Beam Fit Coefficient Evaluation</td>
</tr>
</tbody>
</table>
APPENDIX B

COMPILED FORTRAN FOR HOLLOW BLADE
DEMONSTRATION CASE
DATA SET U490SSTAEB AT LEVEL 019 AS OF 03/15/82
DATA SET U490SSTAEB AT LEVEL 018 AS OF 01/20/82
DATA SET U490SSTAEB AT LEVEL 015 AS OF 10/06/81
DATA SET U490SSTAEB AT LEVEL 013 AS OF 08/24/81
DATA SET U490SSTAEB AT LEVEL 012 AS OF 08/14/81

ISH 0002
SUBROUTINE STAEEL

ISH 0003 COMMON /BLK4/ XP(3,53,21), NP(21), NIP(21), INF
ISH 0004 COMMON /BLK 2/ HA4CR, NSTA, NSTA1, POTISSN, MST2, MST3
ISH 0005 COMMON /BLK 4/ RRM, XHOACR, BR, PI, TERM, N3S
ISH 0006 COMMON /BLK 7/ AREA(I), CF(I), AA(I), AKG(I), IT(I), T(I)
ISH 0007 COMMON /BLK12/ XBARX(I), XMAX(I), XYMN(I), YBARY(I)

1 TLTA(I)

ISH 0008 COMMON /BLKAA/ ALPHIN(21), XSCC(21), YSCC(21), XX(I), YY(I)
ISH 0009 COMMON /BLK6/ XH(I), I'TLc(10), VARI(235), TMX(I), HALPFA(I)
ISH 0010 COMMON /BLK 5/ ALPHA(I), HA(I), HNRT(I), HALPH(I)

ISH 0011 XH(I) = H1NERT(I)

ISH 0012 COMMON /INPUT/ XSAVE(1000), YSAVE(1000), ZSAVE(1000), TSAVE(1000)
ISH 0013 COMMON /SC/ XSCSV(21), YSCSV(21), ZSCSV(21)
ISH 0014 1 ALSAVE(21), POLARN(I), ASAVE(21)

ISH 0015 NBR, TSKIN, TCENT, PXT, F-XT, F-T, F-E

ISH 0016 DIMENSION MM53(21), XM53(21), TMAX53(21), T(120), MM53(21)
ISH 0017 1, XHF(50,21), YHF(50,21), TMLF(50,21), XX53(50,21), YCG(50,21)
ISH 0018 2, XS5CC(21), YSCC(I), X5CC(21), Y5CC(21), ZSCC(21)
ISH 0019 3 XG(50,21), YG(50,21), ZG(50,21)
ISH 0020 4, X5CC(21), YSCC(21), X5CC(21), Y5CC(21)
ISH 0021 5, X(50,21), Y(50,21), Z(50,21)

ISH 0022 REAL*4 A151, B51, C51, D51, A151, B151, C151, D151,
ISH 0023 1, A251, B251, C251, D251, A351, B351, C351,
ISH 0024 2, D351, S151, S251

ISH 0025 REAL*4 XM51, YN51, XM51, XMF25, YM25, TMLF25
ISH 0026

ISH 0027 10 CONTINUE
ISH 0028 20 CONTINUE

ISH 0029 NBR = 1
ISH 0030 DO 20 I = NBR, NSTA
ISH 0031 NPS = NP(I)
ISH 0032 DO 10 N = 1, NPS
ISH 0033 YMN(I) = XP(2, N, I) + X(3, N, I) / 2.0
ISH 0034 XM(N, I) = XP(N, I)
ISH 0035 THC(I, I) = XP(2, N, I) - XP(3, N, I)
ISH 0036 10 CONTINUE
ISH 0037
ISH 0038 10 CONTINUE
ISH 0039
ISH 0040 DO 40 I = NBR, NSTA
ISH 0041 NPS = NP(I) - 1
ISH 0042 THL(I, I) = 0.0
ISH 0043 THL(NPS + 1, I) = 0.0
ISH 0044 DO 30 N = 2, NPS
ISH 0045 TH1 = ATAN2((YMN(I) - YMN(I - 1)), (XM(N, I) - XM(N - 1, I)))
ISH 0046 TH2 = ATAN2((XM(N + 1, I) - XM(I)), (XM(N + 1, I) - XM(I)))
ISH 0047
THT = (THT1 + THT2) / 2.0
TMLN(I) = TMCN(I) * COS(THT)
40 CONTINUE
NSTNB = NSTA - NCR
DO 45 I = NER,NSTA,NSTNB
NPS = NP(I)
WRITE (6,900) (YN(I),I=1,NPS)
WRITE (6,900) (TN(I),I=1,NPS)
WRITE (6,900) (THL(I),I=1,NPS)
CONTINUE
SAVE MEAN Y VALVES AND X VALVES IN AN ARRAY AND INTERPOLATE
FOR EQUAL INCREMENT X VALVES. REPEAT FOR THICKNESSES
DO 110I = NER,NSTA
NPF = 15
NPS = NP(I)
FILL IN DUMMY ARRAYS XM,YN,TN FROM XM,YN,THL VECTORS
DO 201 K = 1,NPS
TM(K) = THL(K,I)
YN(K) = YN(K,I)
201 YN(K) = YN(K,I)
CALL PENFIT FOR CURVE FIT
CALL PENFIT(XM,YM,1,1,NPS,A,B,C,D,A1,B1,C1,D1,S1)
CALL PENFIT(XM,YM,1,1,NPS,A2,B2,C2,D2,A3,B3,C3,D3,S2)
EQUAL BREAKUP ARC-LENGTH
SARC = S1(NPS) / NPF
NOW SEARCH S1 ARRAY FOR INTERVAL VALUE
NODE = 1
ARC1 = SARC / 2.
N = 1
205 IF(ARC1 .LE. SI(N)) GO TO 220
N = N + 1
210 IF(ARC1 .LE. SI(N)) GO TO 220
GO TO 210
DIST = (ARC1 - SI(N-1)) / (SI(N) - SI(N-1))
SOME = (SI(N) - SI(N-1)) / DIST
S120 = (S12(N) - S2(N-1)) / DIST
CALL CUSIC(A(N-1),B(N-1),C(N-1),D(N-1),SOME,XMLF(NODE))
CALL CUSIC(A(N-1),B(N-1),C(N-1),D(N-1),SOME,YMLF(NODE))
CALL CUSIC(A(N-1),B(N-1),C(N-1),D(N-1),SOME,TMLF(NODE))
NODE = NODE + 1
ARC1 = ARC1 + SARC
IF(NODE .LE. NPF) GO TO 205
FILL IN THE XMF, YMF, THLF VECTORS WITH THE INTERPOLATED VALUES
C TRANSLATE COORDINATES TO AN ENGINE AXIS SYSTEM

DO 130 I = NSR,NSTA
XCHORD = XH(I)
XSCL = XCHORD - XSC(I)
XSCG(I) = XSCL - (XCHORD - XBARI(I))

DO 120 N = 1,NPF
XCGW,I) = XFW,I) - X8ARX(I)
YCGW,I) = YFW,I) - YBARY(I)

120 CONTINUE
130 CONTINUE

SHIFT XY PLANE TO ENGINE AXIS YZ

DO 150 I = NSR,NSTA
XSCG(I) = R(I)
YSCG(I) = YSCG(I)
ZSCG(I) = XSCG(I)

DO 140 N = 1,NPF
XG(N,I) = R(I)
YG(N,I) = YG(N,I)
ZG(N,I) = ZG(N,I)

140 CONTINUE
150 CONTINUE

ROTATE THRU ALPHA CHORD - 90.0

DO 170 I = NSR,NSTA
ALPHA(I) = 90.0 * .0174533
ANG = -ALPHA(I) + 90.0 * .0174533
EN = COS(ANG)
EH = SIN(ANG)

XSCR(I) = XSCG(I)
YSCR(I) = YSCG(I) * EN + EH * ZSCG(I)
ZSCR(I) = EH * ZG(N,I) - EN * YG(N,I)

WRITE (6,900) ANG, EH, EN, XSCR(I), YSCR(I), ZSCR(I), AREA(I)

DO 160 N = 1,NPF
XG(N,I) = XG(N,I)
YG(N,I) = YG(N,I) * EN + EH * ZG(N,I)
ZG(N,I) = EH * ZG(N,I) - EN * YG(N,I)

160 CONTINUE
170 CONTINUE

C DO 170 I = NSR,NSTA,NSTB
C WRITE (6,900) (XCG(N,I),N=1,NPF)
C WRITE (6,900) (YCG(N,I),N=1,NPF)
C WRITE (6,900) (XG(N,I),N=1,NPF)
C WRITE (6,900) (YD(N,I),N=1,NPF)
C WRITE (6,900) (ZD(N,I),N=1,NPF)
C WRITE (6,900) (XF(I,N),N=1,NPF)
C WRITE (6,900) (YF(I,N),N=1,NPF)
C WRITE (6,900) (ZF(I,N),N=1,NPF)
C WRITE (6,900) (THLF(N,I),N=1,NPF)
C 175 CONTINUE

IJ = 1
IF(IJ .EQ. 1) GO TO 811
C TO3902 GENERATED NODES AND THICKNESSES
C
DO 800 I = NBR,NSTA
DO 801 J = 1,NPF
801 READ(5,602) Y(J,I),Z(J,I)
802 FORMAT(32X,2F8.0)
C DO 805 J = 1,NPF
READ(5,603) P1,P2
803 FORMAT(24X,F8.0,/,24X,F8.0)
805 TILF(J,I) = (P1 + P2) / 2.
C WRITE(6,810) I
C010 FORMAT(5X,'TO39 NODES AND THICKNESSES FOR SECTION ',I5)
C WRITE(6,900) (X(N,I),N=1,NPF)
C WRITE(6,900) (Y(N,I),N=1,NPF)
C WRITE(6,900) (Z(N,I),N=1,NPF)
C WRITE(6,900) (THLF(N,I),N=1,NPF)
C C 800 CONTINUE
811 CONTINUE
J = 0
K = 0
DO 190 I = NBR,NSTA
K = K + 1
XSCSV(K) = XSCR(I)
YSCSV(K) = YSCR(I)
ZSCSV(K) = ZSCR(I)
ALSAVE(K) = ALPHAI(I)
POLARI(K) = X1MINI + X1MAXI
ASAVE(K) = AREAI(I)
DO 180 N = 1,NPF
J = J +1
XSAVE(J) = X(N,I)
YSAVE(J) = Y(N,I)
ZSAVE(J) = Z(N,I)
TSAVE(J) = THLF(N,I)
C C038 IF(J .EQ. 1) GO TO 180
WRITE (7,901) XSAVE(J), YSAVE(J), ZSAVE(J), TSAVE(J)
C C 180 CONTINUE
ISH 0143  NTHN = NSTA - NBR + 1  00232
ISH 0145  IF (NCD .EQ. 1)  00233
         1 CALL HOLLOW (DLED, DTED, DROOT, DTIP,  00234
         2 TTID, TLTD, HPF, NTHNR)  00235
ISH 0147  IF (NCD .EQ. 2)  00236
         1 CALL LAMIN (TSKIN, TCENTR, PBT, PGE, HPF, NTHNR)  00237
ISH 0149  901 FORMAT (6E12.5)  00238
ISH 0150  900 FORMAT (1X,9F8.5)  00239
ISH 0151  RETURN  00240
ISH 0152  END  00241

*OPTIONS IN EFFECT*NAME=MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOOBJ(DBL4)
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NOCMAP NOFORMAT NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS*  SOURCE STATEMENTS =  151, PROGRAM SIZE =  141916, SUBPROGRAM NAME =STAEBL
*STATISTICS*  NO DIAGNOSTICS GENERATED
**** END OF COMPILATION ******  
2972K BYTES OF CORE NOT USED
DATA SET U498DSTABE AT LEVEL 048 AS OF 03/31/82

SUBROUTINE BBIP ( NC )

IMPLICIT REAL*(4A-H,O-Z)

MULTI-MODE BENDING BIRD INGESTION PARAMETER

COMMON /FAIL/ VV(42),TSAI8(6,25)

COMMON /SMAXX/,SMAXX(8),SMAX(8),SMAXE(8),SMAXTE(8)

COMMON /MODAL/ STRESS(3,8,25),FACTOR(3),RMASS(3)

COMMON /BLK15/,DIANO(100),FNI(100),FILL(300)

DIMENSION SEFF(3,8,25)

CALCULATE TIME TO 1/4 OF FIRST BENDING

TIME = 0.25 * FNI(11)

WRITE(6,200) TIME

200 FORMAT(//,5X,'BBIP CALCULATION, MODAL SUPERPOSITION METHOD',/,
       15X,'QUARTER OF FIRST BENDING (SECONDS) =',E12.5,/
C EFFECTIVE STRESS CALCULATION - MODAL SUPERPOSITION FOR EACH LAYER, EACH BEAM - THREE COMPONENTS
C
ARG = FNI(11) * 2. * 3.141593 * TIME
FS1 = FACTOR(1) * DSIN ARG
ARG = FNI(2) * 2. * 3.141593 * TIME
FS2 = FACTOR(2) * DSIN ARG
ARG = FNI(3) * 2. * 3.141593 * TIME
FS3 = FACTOR(3) * DSIN ARG

DO 100 I = 1,3
DO 100 J = 1,8
DO 100 K = 1,NC
SEFF(I,J,K) = STRESS(1,I,J,K) * FS1 + STRESS(2,I,J,K) * FS2 +
      1 STRESS(3,I,J,K) * FS3

TSAI-WU FAILURE CALCULATION
C
CALL TSAINU ( SEFF,NC )

PICK OUT THE MAX TSAI-WU FOR EACH LAYER OVER ALL BEAMS
C
DO 110 I = 1,8
DO 110 J = 1,NC
IF(SMAX3(I) .LT. TSAI(I,J)) SMAX3(I) = TSAI(I,J)

WRITE(6,300)

300 FORMAT(//,5X,'** MAX TSAI-WU ROOT STRESS ANALYSIS **')

CALL MATFRN(SMAX1,8,1,'MAX1')

WRITE(6,301)

CALL MATFRN(SMAX2,8,1,'MAX2')

WRITE(6,302)
**BBIP**

**SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)**

**DATE**

**82.141/10.47.43**

**PAGE** 2

**ISH 0035** 302 **FORMAT(//,5X,'** MAX TSAI-WU BBIP ANALYSIS **')** 00055*36

**ISH 0036** CALL MATPRN(SMAXX,8,1,'MAX3') 00056*36

**ISH 0037** WRITE(6,303) 00057*37

**ISH 0038** 303 **FORMA**T(//,5X,'** TSAI-WU AT LE AND TE **') 00058*37

**ISH 0039** CALL MATPRN(SMAXL,8,1,' LE ') 00059*37

**ISH 0040** CALL MATPRN(SMAXK,8,1,' TE ') 00060*37

**ISH 0041** RETURN 00061*35

**ISH 0042** END 00062*35

**ISH 0043**

**RETURN**

**ISH 0044** **END**

**OPTIONS IN EFFECT** NAME(HAIX) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTO DBL(DBL4)

**OPTIONS IN EFFECT** SOURCE EBCDIC NOLIST NODECK OBJECT NOMAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)

**STATISTICS**

SOURCE STATEMENTS = 41, PROGRAM SIZE = 6244, SUBPROGRAM NAME = BBIP

**STATISTICS**

NO DIAGNOSTICS GENERATED

****** END OF COMPIIIATION *******

3012K BYTES OF CORE NOT USED
SUBROUTINE BIRD( NR, NC, ISTA, BFORCE, SQUASH )

IMPLICIT REAL*8 (A-H, O-Z)

CALCULATES THE BIRD IMPACT FORCE

ASSUMPTIONS
1. 2 INCHES FROM TIP
2. 2.5 LB BIRD - 4 INCH DIAMETER
3. VBIRD = 180 KNOTS
4. LOADED OVER SQUASH-UP TIME
5. SLICE FROM CENTER OF SPHERE

COMMON /INPUTT/ (X(1000), Y(1000), Z(1000), T(1000))
COMMON /BLK 3/ FN, BLADES, BETA, THR, HT1, HT2, NBB
COMMON /C(6,6), RPH
COMMON /SC/ YSC(21), ZSC(21), ALPHA(21), SECIP(21), SEC3A(21)
COMMON /BLK 6/ SHB(21), ITTLE(18), VARI(235), THMAX(21), HALPHA(21)
DIMENSION BFORCE(6,1)

IMPACT STATION, ISTA

ISTA = 1
ISTOP = X(NR * NC) - 2.
NODE = (ISTA - 1) * NC + 1
IF (X(NODE) < XTOP) GO TO 50
ISTA = ISTA + 1
GO TO 51

50 CONTINUE

IST = ISTA - 1
NODE = NODE - NC
WRITE(6,300) NODE, ISTA
FORMAT(//,5X, '** IN BIRD **, NODE, ISTA', 215)

CALCULATE THE IMPACT PARAMETERS

VBLADE = X(NODE) * RPH
VREL = DSQRT(VBLADE**2 + VBIRD**2)
ARG = VBLADE / VBLADE
PHI = DATAN(ARG)
THETA = ALPHA(ISTA) - PHI
GAP = 2. * 3.141593 * X(NODE) / BLADES
SL = GAP * DTAN(PHI)
RBI = 2.
SQUASH = 2. * RBI / VREL
RHD = 0.036 / 366.4
BMASS = SL * 3.141593 * RBI / RHD

F = BMASS * VREL * DSIN(THETA) / SQUASH
CENTER OF IMPACT, COI

COI = SL / 2. / DSIN(PHI)
C MOMENT ARM

C TORQUE = SIN(ISTA) / 2. - COI
C TORQUE = TORQUE * F

C FORCE VECTOR, BFORCE

C BFORCE(1,1) = 0.
C BFORCE(2,1) = F.
C BFORCE(3,1) = 0.
C BFORCE(4,1) = TORQUE
C BFORCE(5,1) = 0.
C BFORCE(6,1) = 0.

C CALL MATPRM (BFORCE,6,1,'BFOR')

C WRITE(6,200) ISTA,NODE,vBLADE,vBIRD,vREL,PHI,THETA,SL,RBIRD,
C 1  SQUASH,RHO,BMASS,F,COI,TORQUE
C200  FORMAT(/,5X,'BBIP ANALYSIS',/,'STATION,NODE,vBLADE,vBIRD',
C 1    2IE12.5,/,5X,'vREL,PHI,THETA,SL,RBIRD,SQUASH',6E12.5,
C 2    /,5X,'RHO,BMASS,F,COI,TORQUE',6E12.5,)

C RETURN

*OPTIONS IN EFFECT* NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(DBL4)
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NODIAK NOFORMAT GOSTNT NOPOS NOALC NOALSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 42, PROGRAM SIZE = 1068, SUBPROGRAM NAME = BIRDF
*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILED *****

3012K BYTES OF CORE NOT USED
SUBROUTINE BCOORD(IP, IPP, BLOCAL)

THIS ROUTINE CALCULATES THE BEAM LOCAL COORDINATE SYSTEM, BLOCAL

THE SYSTEM IS LOCATED AT THE BEAM END 1

LONGITUDINAL VECTOR, X

Z UNIT VECTOR = X CROSS Y

THE LOCAL Y UNIT VECTOR = Z CROSS X

RETURN

* OPTIONS IN EFFECT * NAME (MAIN) OPTIMIZE (3) LINECOUNT (60) SIZE (MAX) AUTODBL (DBL4)

* OPTIONS IN EFFECT * SOURCE EBCDIC NOLIST NODECK OBJECT NOHAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG (I)

*STATISTICS* SOURCE STATEMENTS = 30, PROGRAM SIZE = 718, SUBPROGRAM NAME =BCOORD

*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF-compilation *****

3016K BYTES OF CORE NOT USED
SUBROUTINE BEAN(NC,IBEAM,BLOCAL,BCHGG,RFN)

THIS ROUTINE DEFINES THE FOLLOWING MATRICES:

CH(12,12) -- COEFFICIENT MATRIX

X(6,12) -- STRAIN-DISPLACEMENT MATRIX

A(12,6) -- EQUILIBRIUM MATRIX

EM(12,12) -- THE BEAN MASS (NO ROTATIONAL INERTIA)

BCHGG(12,12) -- THE BEAN CENTRIFUGAL RESTORING MATRIX

THEN FINDS THE BEAM STIFFNESS MATRIX, BK(12,12)

DEJNICATIONS:

A-H, O-Z

FIRST ZERO ALL MATRICES TO BE USED

DO 100 I = 1,12
    DO 100 J = 1,12
    BK(I,J) = 0.
    CMGG(I,J) = 0.
    CH(I,J) = 0.

    S = BSPAN(IBEAM)
    S2 = S**2
    S3 = S**3
    CH(1,1) = 1.
    CH(2,2) = -1.
    CH(3,1) = -1. / S
    CH(3,3) = 1. / S
    CH(4,2) = -1. / S
    CH(4,4) = 1. / S
    CH(5,5) = 1.
    CH(6,6) = 1.
    CH(7,7) = -3. / S2
    CH(7,6) = -2. / S
    CH(7,7) = 3. / S2

BEGIN LOOP FOR BEAM (J)

THE COEFFICIENT MATRIX, CH(12,12)
+VERSION 1.3.0 (01 MAY 80) BEAM SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 82.141/10.48.06 PAGE 2

ISH 0036  
ISH 0037  
ISH 0038  
ISH 0039  
ISH 0040  
ISH 0041  
ISH 0042  
ISH 0043  
ISH 0044  
ISH 0045  
ISH 0046  
ISH 0047  
ISH 0048  
ISH 0049  
ISH 0050  

C STRAIN-DISPLACEMENT MATRIX, X(6,12)

ISH 0051  
ISH 0052  
ISH 0053  
ISH 0054  
ISH 0055  
ISH 0056  
ISH 0057  
ISH 0058  

C THE EQUILIBRIUM MATRIX, A(12,8)

ISH 0059  
ISH 0060  
ISH 0061  
ISH 0062  
ISH 0063  
ISH 0064  
ISH 0065  
ISH 0066  
ISH 0067  
ISH 0068  
ISH 0069  
ISH 0070  
ISH 0071  
ISH 0072  
ISH 0073  
ISH 0074  
ISH 0075  
ISH 0076  
ISH 0077  
ISH 0078  

C BK = A(12,8)*E(8,6)*X(6,12)*CN(12,12)  
CALL MATMPYCX(E,DI,D2,8,6,12)  

ISH 0079  
ISH 0000  

C CALL MATMPY(X,CH,D1,6,12,12)  
CALL MATMPY(E,D1,D2,8,6,12)
CALL MATMUL(A,BK,12,9,12)

REORDER THE BEAM STIFFNESS, BK, TO LOOK LIKE A NASTRAN VECTOR

BK = P * BK * P(TRANS)

CALL MATMUL(BK,PT,12,12,12)

CALL MATMUL(P,12,BK,12,12)

FORM THE BEAM MASS MATRIX, BM

DO 15 I = 1,12
  DO 15 J = 1,12
  BM(I,J) = 0.

BM(1,1) = BM
BM(2,2) = BM
BM(3,3) = BM
BM(7,7) = BM
BM(8,8) = BM
BM(9,9) = BM

THE BEAM CENTRIFUGAL RESTORING MATRIX, BCMG

-- FIRST DEFINE THE TRANSFORMATION MATRIX, BT

IRON = 1

DO 90 I = 1,4
  DO 90 J = 1,3
  ICOL = (I-1) * 3 + 1

DO 91 K = 1,3
  BL(IRON,ICOL) = BLOCAL(J,K)

ICOL = ICOL + 1

IRJW = IRJW + 1

DO 92 I = 1,12
  DO 92 J = 1,12
  BT(I,J) = BL(J,I)

NONZERO VALUES OF THE CENTRIFUGAL RESTORING MATRIX

WHEN DEFINED IN THE GLOBAL COORDINATE SYSTEM

CHGG(2,2) = BMASS(IBEAM) / 2. * RPM**2

CHGG(8,8) = CHGG(2,2)

TRANSFORM CHGG TO THE LOCAL BEAM SYSTEM TO FIND BCHSG

BCHSG = BL * CHGG * BLT

CALL MATMUL(CHSG,BLT,12,12,12)

CALL MATMUL(BL,12,BCHSG,12,12)

WRITE(6,7)

FORMAT(5X,'BEAM STIFFNESS,BK',/)

CALL MATPRN(BL,12,12,' BK ')

WRITE(6,16)

FORMAT(5X,'BEAM MASS , BM',/)

CALL MATPRN(BM,12,12,' BM ')

--WRITE(6,16)
C 00362
ISH 0109 RETURN
ISH 0110 END

*OPTIONS IN EFFECT=NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(DBL4)
*OPTIONS IN EFFECT=SOURCE EBCDIC NOLIST NODECK OBJECT NCHAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 109, PROGRAM SIZE = 10198, SUBPROGRAM NAME = BEAM
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPILATION *****

2996K BYTES OF CORE NOT USED
C C SUBROUTINE COR2R(XP,YP,ZP,ISPAN,ICHORD) 00365
C C IMPLICIT REAL*(A-H,O-Z) 00366
C C COMMON /COORD1/ LOCAL(3,3,1000) 00367
C C DIMENSION XP(1),YP(1),ZP(1),BLOCAL(3,3) 00368
C C NLCNPC=ICHORD 00369
C C NLRNFR=ISPAN 00370
C C THIS ROUTINE CALCULATES THE COR2R FOR THE BLADE 00371
C C ISN 0002 00372
C C IP = 1 00373
C C DO 100 I = 1,ISPAN 00374
C C IP = (I-1) * NLCNPC + 1 00375
C C DO 101 J = 1,ICHORD 00376
C C IF(I.EQ.1.AND.J.EQ.1) GO TO 200 00377
C C IF(I.EQ.1.AND.J.EQ.1) GO TO 1 00378
C C IF(I.EQ.1.AND.J.GT.1.AND.J.LT.NLCNPC) GO TO 2 00383
C C IF(I.EQ.1.AND.J.GT.1.AND.J.LT.NLCNPC) GO TO 3 00394
C C IF(I.EQ.1.AND.J.GT.1.AND.J.LT.NLCNPC) GO TO 4 00385
C C IF(I.EQ.1.AND.J.GT.1.AND.J.LT.NLCNPC) GO TO 5 00386
C C IF(I.EQ.1.AND.J.GT.1.AND.J.LT.NLCNPC) GO TO 6 00387
C C IF(I.EQ.1.AND.J.GT.1.AND.J.LT.NLCNPC) GO TO 7 00388
C C IF(I.EQ.1.AND.J.GT.1.AND.J.LT.NLCNPC) GO TO 8 00389
C C IF(I.EQ.1.AND.J.GT.1.AND.J.LT.NLCNPC) GO TO 9 00390
C C GO TO 10 00391
C C X1 = XP(IP+NLCNPC) - XP(1) 00392
C C X2 = YP(IP+NLCNPC) - YP(1) 00393
C C X3 = ZP(IP+NLCNPC) - ZP(1) 00394
C C Z1 = 0. 00395
C C Z2 = YP(IP+1) - YP(1) 00396
C C Z3 = ZP(IP+1) - ZP(1) 00397
C C GO TO 10 00398
C C X1 = XP(IP+NLCNPC) - XP(IP) 00399
C C X2 = YP(IP+NLCNPC) - YP(IP) 00400
C C X3 = ZP(IP+NLCNPC) - ZP(IP) 00401
C C Z1 = 0. 00402
C C Z2 = YP(IP+1) - YP(IP-1) 00403
C C Z3 = ZP(IP+1) - ZP(IP-1) 00404
C C GO TO 10 00405
C C X1 = XP(IP+NLCNPC) - XP(IP) 00406
C C X2 = YP(IP+NLCNPC) - YP(IP) 00407
C C X3 = ZP(IP+NLCNPC) - ZP(IP) 00408
C C Z1 = 0. 00409
C C Z2 = YP(IP) - YP(IP-1) 00410
C C Z3 = ZP(IP) - ZP(IP-1) 00411
C C GO TO 10 00412
C C X1 = XP(IP+NLCNPC) - XP(IP) 00413
C C X2 = YP(IP+NLCNPC) - YP(IP) 00414
C C X3 = ZP(IP+NLCNPC) - ZP(IP) 00415
C C Z1 = 0. 00416
C C Z2 = YP(IP+1) - YP(IP) 00417
C C Z3 = ZP(IP+1) - ZP(IP) 00418
C C GO TO 10 00419
ISH 0061 5 X1 = XP(IP+HLCNPC) - XP(IP-HLCNPC) 00420
ISH 0062 X2 = YP(IP+HLCNPC) - YP(IP-HLCNPC) 00441
ISH 0063 X3 = ZP(IP+HLCNPC) - ZP(IP-HLCNPC) 00422
ISH 0064 Z1 = 0. 00423
ISH 0065 Z2 = YP(IP+1) - YP(IP-1) 00424
ISH 0066 Z3 = ZP(IP+1) - ZP(IP-1) 00425
ISH 0067 GO TO 10 00426
ISH 0068 6 X1 = XP(IP+HLCNPC) - XP(IP-HLCNPC) 00427
ISH 0069 X2 = YP(IP+HLCNPC) - YP(IP-HLCNPC) 00428
ISH 0070 X3 = ZP(IP+HLCNPC) - ZP(IP-HLCNPC) 00429
ISH 0071 Z1 = 0. 00430
ISH 0072 Z2 = YP(IP) - YP(IP-1) 00431
ISH 0073 Z3 = ZP(IP) - ZP(IP-1) 00432
ISH 0074 GO TO 10 00433
ISH 0075 7 X1 = XP(IP) - XP(IP-HLCNPC) 00434
ISH 0076 X2 = YP(IP) - YP(IP-HLCNPC) 00435
ISH 0077 X3 = ZP(IP) - ZP(IP-HLCNPC) 00436
ISH 0078 Z1 = 0. 00437
ISH 0079 Z2 = YP(IP+1) - YP(IP) 00438
ISH 0080 Z3 = ZP(IP+1) - ZP(IP) 00439
ISH 0081 GO TO 10 00440
ISH 0082 8 X1 = XP(IP) - XP(IP-HLCNPC) 00441
ISH 0083 X2 = YP(IP) - YP(IP-HLCNPC) 00442
ISH 0084 X3 = ZP(IP) - ZP(IP-HLCNPC) 00443
ISH 0085 Z1 = 0. 00444
ISH 0086 Z2 = YP(IP+1) - YP(IP-1) 00445
ISH 0087 Z3 = ZP(IP+1) - ZP(IP-1) 00446
ISH 0088 GO TO 10 00447
ISH 0089 9 X1 = XP(IP) - XP(IP-HLCNPC) 00448
ISH 0090 X2 = YP(IP) - YP(IP-HLCNPC) 00449
ISH 0091 X3 = ZP(IP) - ZP(IP-HLCNPC) 00450
ISH 0092 Z1 = 0. 00451
ISH 0093 Z2 = YP(IP) - YP(IP-1) 00452
ISH 0094 Z3 = ZP(IP) - ZP(IP-1) 00453
ISH 0095 10 CONTINUE 00454

C FORM THE UNIT VECTORS , FIRST LOCAL Z THEN Y AND FINALLY X
C
ISH 0096 ZMAG =DSQRT(Z1**2 + Z2**2 + Z3**2) 00455
ISH 0097 Z1 = Z1 / ZMAG 00456
ISH 0098 Z2 = Z2 / ZMAG 00457
ISH 0099 Z3 = Z3 / ZMAG 00458
ISH 0100 XMAG =DSQRT(X1**2 + X2**2 + X3**2) 00459
ISH 0101 X1 = X1 / XMAG 00460
ISH 0102 X2 = X2 / XMAG 00461
ISH 0103 X3 = X3 / XMAG 00462

C LOCAL Y UNIT VECTOR , Z CROSS X
ISH 0104 Y1 = Z2*X3 - Z3*X2 00463
ISH 0105 Y2 = -Z1*X3 + Z3*X1 00464
ISH 0106 Y3 = Z1*X2 - Z2*X1 00465
ISH 0107 YMAG =DSQRT(Y1**2 + Y2**2 + Y3**2) 00466
ISH 0108 Y1 = Y1 / YMAG 00467
ISH 0109 Y2 = Y2 / YMAG 00468
ISH 0110 Y3 = Y3 / YMAG 00469

C LOCAL X UNIT VECTOR , Y CROSS Z
ISH 0111 X1 = Y2*Z3 - Y3*Z2 00470
ISH 0112 X2 = -Y1*Z3 + Y3*Z1 00471
ISH 0113 X3 = Y1*Z2 - Y2*Z1 00472
ISH 0114 A1=0 00473
ISH 0115 A2=0 00474
ISH 0116 A3=0 00475
CONTINUE

IF (ICHORD .EQ. 1) GO TO 201
GO TO 202
CLOCAL(2,1,IP) = 0.
CLOCAL(2,2,IP) = 1.
CLOCAL(2,3,IP) = 0.
II = IP
IF (I .EQ. ISPAN) II = IP - 1
III = II + 1
CALL BCOORD(II,III,BLOCAL)
DO 400 K = 1,3
DO 400 L = 1,3
CLOCAL(K,L,IP) = BLOCAL(K,L)
GO TO 302
CONTINUE
IIP=IP+1
CLOCAL(IIP,2,IP) = XI
CLOCAL(IIP,2,IP) = X2
CLOCAL(IIP,2,IP) = X3
CLOCAL(IIP,3,IP) = Y1
CLOCAL(IIP,3,IP) = Y2
CLOCAL(IIP,3,IP) = Y3
CLOCAL(IIP,3,IP) = Z1
CLOCAL(IIP,3,IP) = Z2
CLOCAL(IIP,3,IP) = Z3
CONTINUE
5X, 'CLOCAL FOR NODE', IS)
(5X,3EI2.5)
IP = IP + 1
CONTINUE
RETURN
EXECUTABLE ****** 2992K BYTES OF CORE NOT Used
SUBROUTINE ESTIFF(ISEC,NC,NR,I)
C THIS ROUTINE DETERMINES THE MATERIAL STIFFNESS MATRIX, E, FOR EACH BEAM. EVALUATED AT EACH END.
C
IMPLICIT REAL*8(A-H,O-Z)
COMMON /STIF/E(8,6)
COMMON /QJ/Q(3,3,7)
COMMON /LAYER/TH(7,25),BMASS(25),BSPAN(25),BWIDTH(25)
COMMON /INPUT/X(1000),Y(1000),Z(1000),T(1000)
COMMON /BLK/SH(21),ITLE(18),VARI(235),TMAX(21),HALPHA(21)
DIMENSION Y(8)

DETERMINE THE LAYER THICKNESSES
Y(J) = -(TH(1,J) + TH(2,J) + TH(3,J) + TH(4,J)/2.)
DO 101 J = 2,8
Y(J) = Y(J-1) + TH(J-1)
FINA A11,A22
A11 = 0.
A22 = 0.
DO 102 J = 1,7
A11 = A11 + Q(1,1,J) * (Y(J+1) - Y(J))
A22 = A22 + Q(2,2,J) * (Y(J+1) - Y(J))
A11 = A11 * BWIDTH(I)
A22 = A22 * BWIDTH(I)
FINA B11,B13
B11 = 0.
B13 = 0.
DO 103 J = 1,7
B11 = B11 + Q(1,1,J) * (Y(J+1)*2 - Y(J)*2)
B13 = B13 + Q(3,3,J) * (Y(J+1)*2 - Y(J)*2)
B11 = B11 * BWIDTH(I)/2.
B13 = B13 * BWIDTH(I)/2.
ROUND OFF BIJ VALUES TO ZERO IF LESS THAN 0.
CHECK = 10.
IF(DABS(B11) .LT. CHECK) B11 = 0.
IF(DABS(B13) .LT. CHECK) B13 = 0.
FINA D11,D33,D13
D11 = 0.
D13 = 0.
D33 = 0.
DO 104 J = 1,7
D11 = D11 + Q(1,1,J) * (Y(J+1)*3 - Y(J)*3)
D13 = D13 + Q(1,3,J) * (Y(J+1)*3 - Y(J)*3)
D33 = D33 + Q(3,3,J) * (Y(J+1)*3 - Y(J)*3)
D11 = D11 * BWIDTH(I)/3.
D13 = D13 * BWIDTH(I)/3.
D33 = D33 * BWIDTH(I)/3.
C ROUND OFF D13 TO ZERO IF LESS THAN CHECK = 10.

C WRITE(6,1) I,A11,A22,B11,B13,D11,D13, D33

C FORMAT(5X,'ESTIFF FOR BEAM',I5,5X,'A11,A22,B11,B13,D11,D13,D33', 00576
C $/5X, F10.3)

C FORM THE E MATRIX
C
ISN 0042 105 DO 105 J = 1,6

ISN 0046 105 E(J,J) = 0.

ISN 0047 E(1,1) = A11

ISN 0048 E(1,2) = 2. * B13

ISN 0049 E(1,3) = -B11

ISN 0050 E(2,1) = B13

ISN 0061 TOTAL = Y(O) - Y(1)

ISN 0051 E(2,2) = 4. * D33

C

ISN 0059 E(2,3) = -D13

ISN 0060 E(3,1) = A11

ISN 0061 E(3,2) = 2. * B13

ISN 0062 E(3,3) = -B11

ISN 0063 E(4,1) = B13

ISN 0065 E(4,2) = E(2,2)

C

ISN 0067 E(5,1) = B13

ISN 0068 E(5,2) = -2. * D13

ISN 0069 E(5,3) = D11

ISN 0070 E(6,1) = -B11

ISN 0071 E(6,2) = -2. * D13

ISN 0072 E(6,3) = D11

ISN 0073 E(7,5) = 2.*B13/12. * A11

ISN 0074 E(8,6) = E(7,5)

C ROOT WARPING RESTRANDED ( (2,2) TERM NOT CHANGED)
C

C CALL MATRICE(8,6,' E ')

C

ISN 0072 RETURN

ISN 0073 END

*OPTIONS IN EFFECT* NAME(0MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTO DBL(DBL4)

*OPTIONS IN EFFECT* SOURCE EBCDIC NOHIST NOEDIT OBJECT NOMAP NOFORMAT GOSTH NOREF NOALC NOMAP TERM IBM FLG(1)

*STATISTICS* SOURCE STATEMENTS = 72, PROGRAM SIZE = 1290, SUBPROGRAM NAME = ESTIFF

*STATISTICS* NO DIAGNOSTICS GENERATED

******** END OF COMPIALATION ********
SUBROUTINE EXNECK (B)

EXTENDED NECK STIFFNESS - RECTANGULAR - TITANIUM
15% THICK NECK

COMMON /STIFS/ RBE(12,12,25,21),FOUT(12,21),SL(25,21),SW(25,21)
COMMON /STIFF/ SK(12,12),SIM(6,6,21)
COMMON /BLKAA/ AA(21),XSC(21),YSC(21),XYY(42)
COMMON /BLK12/ XBARX(21),XMAX(42),YBARY(21),TLTA(21)
COMMON /BLKET/ RBE(12,12),RBE(12,12),RBE(12,12)

T = 0.15 * B

RIMIN = B * T**3 / 12.
RIMAX = B**3 / T**12.
TORS = B**3 / T**12 * 0.21 * T**4 * (1. - T**4/12./B**4)
TORS = RMIN + RIMAX

TORS = 1. * TORS

E = 16.166
G = 6.0566
BL = SL(1,1)
AREA = B * T

WRITE(6,10) B,T,RMIN,RIMAX,TORS,E,G,BL,AREA

DO 100 I = 1,12
DO 100 J = 1,12

100 SK(I,J) = 0.

SK(1,1) = AREA * E / BL
SK(1,7) = -SK(1,1)
SK(2,2) = 12. * E * RIMAX / BL**3
SK(2,6) = 6. * E * RIMAX / BL**2
SK(2,6) = -SK(2,2)
SK(2,12) = SK(2,6)
SK(3,3) = 12. * E * RIMAX / BL**3
SK(3,5) = -6. * E * RIMAX / BL**2
SK(3,9) = -SK(3,3)
SK(3,5) = SK(3,5)
SK(4,4) = TORS * G / BL
SK(4,10) = -SK(4,4)
SK(5,5) = 4. * E * RIMAX / BL
SK(5,11) = -SK(5,11)
SK(5,11) = 2. * E * RIMAX / BL
SK(6,6) = 4. * E * RIMAX / BL
SK(6,6) = -SK(2,12)
SK(6,12) = 2. * E * RIMAX / BL
SK(7,7) = SK(1,1)
SK(8,8) = SK(2,2)
SK(8,12) = -SK(2,12)
ISN 0042  SK(9,9) = SK(3,3)
ISN 0043  SK(9,11) = -SK(3,11)
ISN 0044  SK(10,10) = SK(4,4)
ISN 0045  SK(11,11) = SK(5,5)
ISN 0046  SK(12,12) = SK(6,6)

C SYMMETRY CONDITION
C
ISN 0047  DO 110 I = 1,12
ISN 0048  DO 110 J = 1,12
ISN 0049  IF(J .GE. I) GO TO 110
ISN 0050  SK(I,J) = SK(J,I)
ISN 0051  110 CONTINUE

C CALL MATPRN(SK,12,12,'SKRT')
C TRANSLATE TO CG LOCATION
ISN 0052  DZ = XSC(2) - XBARX(2)
ISN 0053  DY = YBARY(2) - YSC(2)
C WRITE(6,11) DZ,DY
C FORMAT(5X,'DZ,DY',2E12.5)

ISN 0054  DO 200 I = 1,12
ISN 0055  DO 200 J = 1,12
ISN 0056  RBET(I,J) = 0.
ISN 0057  200 CONTINUE
      C
ISN 0058  RBET(1,5) = DZ
ISN 0059  RBET(1,6) = -DY
ISN 0060  RBET(2,4) = -DZ
ISN 0061  RBET(3,4) = DY
ISN 0062  RBET(7,11) = DZ
ISN 0063  RBET(7,12) = -DY
ISN 0064  RBET(8,10) = -DZ
ISN 0065  RBET(9,10) = DY
               C
C CALL MATPRN(RBET,12,12,'RBET ')
C RBE TRANSPOSE , RBET
ISN 0066  DO 210 I = 1,12
ISN 0067  DO 210 J = 1,12
ISN 0068  RBET(I,J) = RBET(J,I)
ISN 0069  210 CONTINUE
      C
ISN 0070  CALL MATNHY(SK,RBET,01,12,12,12)
ISN 0071  CALL MATNHY(RBET,SK,12,12,12,12)
C CALL MATPRN(SK,12,12,'SKRT')
ISN 0072  RETURN
ISN 0073  END

*OPTIONS IN EFFECT=NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(OBL4)
*OPTIONS IN EFFECT=SOURCE EBCDIC NOLIST NODECK OBJECT NOCMP NOFORMAT GOSTMT NOHREF NOALC NOANSF TERM IBM FLG(I)
*STATISTICS= SOURCE STATEMENTS = 74, PROGRAM SIZE = 4850, SUBPROGRAM NAME =EXNECK
*STATISTICS= NO DIAGNOSTICS GENERATED
***** END OF COMPIILATION *****
SUBROUTINE FRPM(J,IP,X,Y,BMASS,RPM,FBEAM,BLOCAL)
C C CALCULATING THE FORCES DUE TO RPM ON NODE IP FOR BEAM J
C C DOUBLE PRECISION X(1),Y(1),BMASS(1),RPM,FBEAM(6,1),BLOCAL(3,3)
C C DOUBLE PRECISION F(3,1),FF(3,1),A(6)

DO 100 K = 3,6
A(K) = 0.
100 FBEAM(K,1) = 0.
C C NONZERO ACCELERATION COMPONENTS
C C A(1) = RPM**2 * X(IP)
C C A(2) = RPM**2 * Y(IP)
C C FORCES
C C F(1,1) = BMASS(J)/2. * A(1)
C C F(2,1) = BMASS(J)/2. * A(2)
C C F(3,1) = 0.

WRITE(6,20) J,IP
C20 FORMAT(5X,'** IN FRPM-GLOBAL RPM LOADS AT BEAM, NODE',2I5)
C CALL MATHPY1BLOCAL,F,FF,3,3,1)
C INSERT INTO FBEAM
DO 200 K = 1,3
FBEAM(K,1) = FF(K,1)
200 WRITE(6,20) J,IP
C1 FORMAT(5X,'** IN FRPM **, BEAM AND NODE',2I5)
C CALL MATHPY1BLOCAL,FBEAM,6,1,'FBEM')

RETURN
END

** OPTIONS III EFFECT** NAME(HAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOOBL(DBL4)
* OPTIONS IN EFFECT* SOURCE EBCDIC NODECK OBJECT NOAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
* STATISTICS* SOURCE STATEMENTS = 16, PROGRAM SIZE = 706, SUBPROGRAM NAME = FRPM
* STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPIATION *****
SUBROUTINE HOLZERILL(NR,NC,RP)

THIS ROUTINE CALCULATES ALL SECTION STIFFNESSES AND MASSES AND THEN TRANSFORMS THESE INTO HOLZER MATRICES

IMPLICIT REAL*8(A-H.O-Z)
COMMON /STIFF/ SK(12,12),SMH(6,6,21)
COMMON /ERORDR/ PT(12,12)
COMMON /STIFF/ SKP(12,12),SKPP(12,12,21)
COMMON /INPUT/ X(1000),Y(1000),Z(1000),T(1000)
DIMENSION SK(11,6),SK(6,6),SK(11,6),D1(12,12),DD(6,6)

BEGIN
THE SECTION STIFFNESS SOLUTION MATRIX SK

NRll = NR - 1
ISKIP = 1
GO TO 98
DO 100
J = 1,12
DO 100 K = 1,12
SK(J,K) = SKK(J,K,1)
CONTINUE
CALL HAPRN(SK,12,12,' SK ')
PARTITION THE SECTION STIFFNESS MATRIX ,SK, INTO:
SK1 , SK2 , SK21 , SK22
CALL PARTH(SK,12,12,SK1,SK12,SK21,SK22,6,6)
INVERT SK12
CALL HIPV(SK12,6,6)
FORM THE SKP MATRIX FROM PARTITIONED SK COMPONENTS
CALL HATPRY(SK12,SK11,SK1,6,6,6)
I.

CHANGE SIGN ON UPPER QUARTER OF SKP

DO 115 K = 1,6
   DO 115 L = 1,6
   SKP(K,L) = -SK1(K,L)
   LM = L - 6
   SKP(K,L) = SK12(K,LM)

   THE SIGN ON SKP21 AND SKP22 ARE NEGATIVE FOR EQUIL.

   CALL MATHPY(SK22,SK1,SK11,6,6,6)
   DO 117 K = 7,12
      KK = K - 6
      J = 1
      DO 117 L = 1,6
         ISN
      JSK1(KL) = SK1(KK,LJ) - SK2(KK,LJ)
      J = J + 1
      FILL THE SKPP ARRAY WITH THE SKP MATRIX, THIS SAVES SKP
      ISN

   WRITE(6,101) I
   C101 FORMAT(6,SK,'SKP IS THE HOLZER STIFFNESS FOR SECTION',I5)
   CALL MATRIP(K,L,SKP)
   C

   CONTINUE

RETURN

END

*VERSION 1.3.0 (01 MAY 80) HOLZER SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 02/14/10.40.36 PAGE 2

00827
00820
00829
00830
00831
00832
00833
00834
00835
00836
00837
00838
00839
00840
00841
00842
00843
00844
00845
00846
00847
00848
00849
00850
00851
00852
00853
00854
00855
00856
00857
00858
00859
00860
00861
00862
00863
00864
00865
00866
00867
00868
00869

*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOMDL(DBL4)

*OPTIONS IN EFFECT*SOURCE EBCDIC HOLIST NODECK OBJECT NOFORMAT GOSTHM NOXREF NOALC NOANSF TERM IBM FLAG(I)

*STATISTICS* SOURCE STATEMENTS = 54, PROGRAM SIZE = 2792, SUBPROGRAM NAME =HOLZER

*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION *****

3004K BYTES OF CORE NOT USED
SUBROUTINE INPUT (NR, NC)

MATERIAL PROPERTIES AND GEOMETRY INPUT

IMPLICIT REAL*8(A-H,O-Z)

COMMON /ZCOEF/ E11(7),E22(7),E33(7),G12(7),G23(7),G13(7),
V12(7),V13(7),V23(7)

COMMON /PLY1/ PLY(21,25,7),THETA(7),ROH(7)

COMMON /FAIL/ X1T(7),X1C(7),X2T(7),X2C(7),S6P(7),S6M(7),TSAI(8,25)1000879*35

WRITE(0,1)
1 FORMAT(5X,'IS THIS AN ISOTROPIC BLADE. I=YES')

READ(8,*) ISO
IF(ISO.NE.1) GO TO 2

WRITE(8,3) E,V,R

E = 16100000.
V = .33
R = .16

DO 100 I = 1,7

X1T(I) = YIELD
X1C(I) = YIELD
X2T(I) = YIELD
X2C(I) = YIELD
S6P(I) = SHEAR
S6M(I) = SHEAR

Ell(I) = E
E22(I) = E
E33(I) = E
G12(I) = E / 2. / (1. + V)
G13(I) = G12(I)
G23(I) = G12(I)
V12(I) = V
V13(I) = V
V23(I) = V

DO 100 I = 1,7

RHO(I) = R / 386.4

WRITE(8,4) ISO
READ(8,*)

Ein(1) = 33.2E6

INPUT MATERIAL STUFF FOR Ti/BORON LAYER
C THETA(2) = 35./180. * 3.14159265
C THETA(6) = THETA(2)

*STATISTICS* SOURCE = 71, PROGRAM SIZE = 936, SUBPROGRAM NAME = INPUT
*STATISTICS* NO DIAGNOSTICS GENERATED
****** END OF COMPILATION ******

3004K BYTES OF CORE NOT USED
SUBROUTINE KDDG(ISEC,FF,FOUT1,DSKK)

Calculation of the differential stiffness matrix for section ISEC, DSKK, (see the NASTRAN theoretical manual for explanation 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

It implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49

CALCULATES THE DIFFERENTIAL STIFFNESS MATRIX FOR SECTION ISEC, DSKK, (SEE THE NASTRAN THEORETICAL MANUAL FOR EXPLANATION 7.2-8).

Implicit REAL*8 (A-H, O-Z)

DATE 82.141/10.48.49
C
ISH 0060  DO 110 I = 1,12
ISH 0061  DO 110 J = 1,12
ISH 0062  IF ( J .LE. I ) GO TO 110
ISH 0064  DSKK(I,J,ISEC) = DSKK(I,J,ISEC)
ISH 0065  110 CONTINUE
C
ISH 0066  DO 200 I = 1,12
ISH 0067  DO 200 J = 1,12
ISH 0068  200 DSKK(I,J) = DSKK(I,J,ISEC)
C CALL MATPRN(DSKK,12,12,'DSKK')
C
ISH 0069  RETURN
ISH 0070  END

*OPTIONS IN EFFECT NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOBL(DBL4)
*OPTIONS IN EFFECT SOURCE EBCDIC NOLIST NODECK OBJECT NOAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS SOURCE STATEMENTS = 69, PROGRAM SIZE = 2544, SUBPROGRAM NAME = KDGG
*STATISTICS NO DIAGNOSTICS GENERATED
**** END OF COMPILATION ****

3004K BYTES OF CORE NOT USED
 SUBROUTINE LAMINA  

 THIS ROUTINE GENERATES THE LAMINA STRESS-STRAIN MATRIX, Q  

 IMPLICIT REAL*8(A-H,O-Z)  
 COMMON /C0EF/ E11(7), E22(7), E33(7), G12(7), G23(7), G13(7), V12(7), V13(7), V23(7)  
 COMMON /PLY1/ PLY1(21,25,7), THETA(7), RHO(7)  
 COMMON /RQIJ/ Q(3,3,7)  

 N = 7  
 DO 100 I = 1, N  
 WRITE(6,1) I  
 1 FORMAT(/,5X,'LAMINA, LAYER ',I4,/)  
 CHECK FOR HOLLOW LAYER  
 IF(E11(1) .EQ. 0.) GO TO 200  
 DETERMINE THE POISSON RATIOS  
 V21 = V12(1) * E22(1) / E11(1)  
 V31 = V13(1) * E33(1) / E11(1)  
 V32 = V23(1) * E33(1) / E22(1)  
 DET = 1. - V12(1)*V21 - V13(1)*V31 - V23(1)*V32 - V21*V13(1)*V32 - V31*V12(1)*V23(1)  
 WRITE(6,2) V21, V31, V32, DET  
 2 FORMAT(5X,4E12.5,/)  
 ROTATE THRU ANGLE THETA (I) ABOUT THE 3 AXIS  
 C =DCOS( THETA(I) )  
 S =DSIN( THETA(I) )  
 CP11 = C**4*C11 + 2.*C**2*G12*C12 + 2.*C**2*G23*C13 + 2.*G12*C23  
 CP12 = C**2*G12*C12 + (C**4+G**4)*C12  
 CP13 = C**2*C12 + G**2*C13  
 CP16 = -C*S*(C**2*C11 + G**2*C22) + C*S*(C**2*C11 + G**2*C22)  
 CP22 = S**4*C11 + 2.*C**2*G12*C12 + 2.*C**2*G23*C13 + 2.*G12*C23
**VERSION 1.3.0 (01 MAY 80)**

**LAMINA**

**SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)**

**DATE** 08.14/10.48.54

**PAGE** 2

---

**ISN 0041**

\[
CP23 = S \times 2 \times C13 + C \times 2 \times C23
\]

**ISN 0042**

\[
CP26 = -C \times S \times (S \times 2 \times C11 - C \times 2 \times C22) - S \times C \times (C \times 2 \times S \times 2 \times C12 \times 2 \times C66)
\]

**ISN 0043**

\[
CP33 = C33
\]

**ISN 0044**

\[
CP36 = C \times S \times (C \times 2 \times C13 - C13)
\]

**ISN 0045**

\[
CP44 = C \times 2 \times C44 + S \times 2 \times C55
\]

**ISN 0046**

\[
CP45 = C \times S \times (C44 - C55)
\]

**ISN 0047**

\[
CP55 = C \times 2 \times C55 + S \times 2 \times C44
\]

**ISN 0048**

\[
CP66 = C \times 2 \times S \times (C11 \times C22 - 2 \times C12) + (C \times 2 \times S \times 2 \times C12 \times 2 \times C66)
\]

---

**ISN 0049**

\[
C = \text{CHECK} = \text{10.}
\]

**ISN 0050**

IF(ABS(CP16) .LT. CHECK) CP16 = 0.

**ISN 0052**

IF(ABS(CP26) .LT. CHECK) CP26 = 0.

**ISN 0054**

IF(ABS(CP36) .LT. CHECK) CP36 = 0.

**ISN 0056**

IF(ABS(CP45) .LT. CHECK) CP45 = 0.

---

**ISN 0058**

WRITE(6,6) CP11,CP12,CP13,CP16

**ISN 0059**

6 FORMAT(1X,3E12.5,2X,E12.5)

**ISN 0060**

WRITE(6,7) CP22,CP23,CP26

**ISN 0061**

7 FORMAT(1X,2E12.5,2X,E12.5)

**ISN 0062**

WRITE(6,8) CP33,CP36

**ISN 0063**

8 FORMAT(2X,E12.5,2X,E12.5)

**ISN 0064**

WRITE(6,9) CP44,CP45

**ISN 0065**

9 FORMAT(2X,E12.5)

**ISN 0066**

WRITE(6,10) CP55

**ISN 0067**

10 FORMAT(2X,E12.5)

**ISN 0068**

WRITE(6,11) CP66

**ISN 0069**

11 FORMAT(2X,E12.5,

---

**ISN 0070**

\[
Z1 = (CP23 \times CP13 - CP12 \times CP33) / (CP22 \times CP33 - CP23 \times CP32)
\]

**ISN 0071**

\[
Z2 = (CP26 \times CP33 - CP23 \times CP36) / (CP22 \times CP33 - CP23 \times CP32)
\]

**ISN 0072**

\[
Z3 = (-CP13 - CP23 - Z1) / CP33
\]

**ISN 0073**

\[
Z4 = (CP36 - CP23 \times Z2) / CP33
\]

---

**ISN 0074**

\[
Q(1,1,1) = CP11 + CP13 \times Z3 + CP12 \times Z1
\]

**ISN 0075**

\[
Q(1,1,2) = 0.
\]

**ISN 0076**

\[
Q(1,3,1) = CP13 \times Z4 + CP12 \times Z2 - CP16
\]

**ISN 0077**

\[
Q(2,1,1) = 0.
\]

**ISN 0078**

\[
Q(2,2,1) = CP55
\]

**ISN 0079**

\[
Q(2,3,1) = 0.
\]

**ISN 0080**

\[
Q(3,1,1) = Q(1,3,1)
\]

**ISN 0081**

\[
Q(3,2,1) = 0.
\]

**ISN 0082**

\[
Q(3,3,1) = -CP36 \times Z4 - CP26 \times Z2 + CP66
\]

---

**ISN 0083**

GO TO 201

**ISN 0084**

200 CONTINUE

**ISN 0085**

DO 210 K = 1,3

**ISN 0086**

DO 210 L = 1,3

**ISN 0087**

210 Q(K,L,1) = 0.

**ISN 0088**

201 CONTINUE

---

**ISN 0089**

WRITE(6,13)

**ISN 0090**

13 FORMAT(5X,'QII')

**ISN 0091**

WRITE(6,12) (Q(K,J,1),J=1,3),II=1,3)

**ISN 0092**

12 FORMAT(5X,3E12.5)

**ISN 0093**

100 CONTINUE

---
*VERSION 1.3.0 (01 MAY 80) LAMINA SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 02.141/10.40.54 PAGE 3
ISH 0094 RETURN
ISH 0095 END
*OPTIONS IN EFFECT:NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(DBL4)
*OPTIONS IN EFFECT:SOURCE EBCDIC Nolist NODECK OBJECT NOMAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 94, PROGRAM SIZE = 2554, SUBPROGRAM NAME =LAMINA
*STATISTICS* NO DIAGNOSTICS GENERATED
END OF COMPIATION

2996K BYTES OF CORE NOT USED
C SUBROUTINE LAMIN (TSKIN, TCENTR, PBT, PGE, NC, NR)
C
C PREPROCESSOR FOR THE COMPOSITE BLADE
C
C INPUT: TSKIN = TITANIUM SKIN THICKNESS IN INCHES
C TCENTR = Ti CENTER LAYER THICKNESS IN INCHES
C PBT = PERCENTAGE OF REMAINING THICKNESS OF BORON/TI
C PGE = PERCENTAGE OF REMAINING THICKNESS OF GRAPHITE/EPO
C
C NOTE ** PBT + PGE = 1.0
C
C
ISN 0002
C
C IMPLICIT REAL*8(A-H,O-Z)
C COMMON /PLY1/ PLY(21,25,7),THETA(7),RHO(7)
C COMMON /INPUT/ X(1000),Y(1000),Z(1000)
C
C NR01 = NR - 1
C DO 100 I = 1,NR01
C TAVE = (T(INODE) + T(NODE+NC))/2.
C CHECK ON TOTAL TITANIUM LAYER THICKNESS
C
C TTI = 2. * TSKIN + TCENTR
C TCHECK = TAVE - TTI
C
C INITIALIZE THICKNESSES
C
C TS = TSKIN
C TC = TCENTR
C TBT = 0.
C TGE = 0.
C
C IF(TCHECK) 110,110,110
C
C NO G/E OR TBT LAYERS DUE TO MINIMUM THICKNESS EXCEEDED
C
C 110 TCHECK = TAVE - 2. * TSKIN
C
C IF(TCHECK) 111,111,112
C
C TS = TAVE / 2.
C
C TC = 0.
C
C GO TO 200
C
C TAVE = TAVE - 2. * TS
C
C GO TO 200
C
C INSERT G/E AND TBT LAYERS
C
C ISN 0012
C
C TBT = PBT * TCHECK / 2.
C TGE = PGE * TCHECK / 2.
C
C CONTINUE
C
C FILL THE PLY ARRAY
C
C ISH 0028
C PLY(I,J,1) = TS
C PLY(I,J,2) = TBT
I.

**VERSION 1.3.0 (01 MAY 80) LAMIN1 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 82.141/10.49.01 PAGE 2

ISH 0030 PLY(I,J,3) = TGE
ISH 0031 PLY(I,J,4) = TC
ISH 0032 PLY(I,J,5) = TGE
ISH 0033 PLY(I,J,6) = TBT
ISH 0034 PLY(I,J,7) = TS

C WRITE(6,300) I,J,NODE,TAVE,TBT,TGE,TC
C300 FORMAT(5X, 'I,J,NODE,TAVE,TBT,TGE,TC',/,5X,
C 1 315,5E12.5)
C
C TOTAL = 2 * (TS + TBT + TGE) + TC
C WRITE(6,301) TOTAL
C301 FORMAT(5X, 'TOTAL SUMMED THICKNESS =', E12.5)
C
C ISN 0035 100 NODE = NODE + 1
C
ISH 0036 RETURN
ISH 0037 END

*OPTIONS IN EFFECT* NAME (MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(DBL4)
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NOMAP NOFORMAT GOSTHT NOXREF NOALC NOASF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 36, PROGRAM SIZE = 922, SUBPROGRAM NAME = LAMIN1
*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION *****

3016K BYTES OF CORE NOT USED
SUBROUTINE LOAD2(NR,FF)

C PRODUCE A SINGLE LOAD VECTOR, FF, FROM ALL
C THE STIFFNESSES-LOAD VECTOR PRODUCTS IN THE HOLZER METHOD

IMPLICIT REAL*8(A-H,O-Z)

C
C ISN 0002
C
C SUBROUTINE LOAD2(NR,FF)
C
C PRODUCE A SINGLE LOAD VECTOR, FF, FROM ALL
C THE STIFFNESSES-LOAD VECTOR PRODUCTS IN THE HOLZER METHOD
C
C ISN 0003
C IMPLICIT REAL*8(A-H,O-Z)
C ISN 0004
C COMMON /STATC/ SH(12,12,20),FF(12,21)
C ISN 0005
C DIMENSION S1(12,12),F1(12,12,1),F2(12,1)
C ISN 0006
C INITIALIZE FF, AND INSERT THE LOAD VECTOR AT NODE NR
C
C ISN 0007
C WRITE(6,10)
C ISN 0008
C 50 FF(I,1) = F(I,NR)
C CALL MATPRN(FF,12,1,' FF ')
C ISN 0009
C NRM1 = NR - 1
C ISN 0010
C IF(NRM1 .EQ. 1) RETURN
C ISN 0012
C DO 100 I = 2,NRM1
C FILL IN DUMMY ARRAYS
C
C ISN 0013
C DO 110 J = 1,12
C ISN 0014
C F1(J,1) = F(J,I)
C ISN 0015
C DO 110 K = 1,12
C ISN 0016
C 110 SI(J,K) = SH(J,K,NR-I)
C WRITE(6,10)
C 10 FORMAT(5X,'*** IN LOAD2 ****')
C CALL MATPRN(S1,12,12,' S1 ')
C CALL MATPRN(F1,12,1,' F1 ')
C PERFORM MATRIX MULTIPLICATION
C
C ISN 0017
C CALL MATPMY(S1,F1,F2,12,12,1)
C CALL MATPRN(F2,12,1,' F2 ')
C SUM INTO THE FF ARRAY
C
C ISN 0018
C DO 120 J = 1,12
C ISN 0019
C 120 FF(J,1) = FF(J,1) + F2(J,1)
C ISN 0020
C CONTINUE
C CALL MATPRN(FF,12,1,' FF ')
C RETURN
C ISN 0021
C END
SUBROUTINE MATADD(A,B,C,I,J,S)

C MATRIX ADDITION A(I,J) + S*B(I,J) = C(I,J)
C WITH, S = SIGN OF B (ADDITION OR SUBTRACTION)

C DOUBLE PRECISION A(I,J),B(I,J),C(I,J),S

DO 100 K = 1,I
   DO 100 L = 1,J
      C(K,L) = A(K,L) + S*B(K,L)
100 CONTINUE

RETURN

END
SUBROUTINE MATMIPY(A,B,C,I,J,K)

THIS ROUTINE MULTIPLIES MATRICES A * B = C

DO 100 L = 1, I

DO 100 N = 1, J

C(L,M) = C(L,M) + A(L,N) * B(N,M)

RETURN

END
**REQUESTED OPTIONS:** AUTODBL, SOURCE

**OPTIONS IN EFFECT:** NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL

**SOURCE EBCDIC NOLIST NODATE OBJECT NODUMP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)**

```fortran
C 01327
ISN 0002 SUBROUTINE MATPRN(A,I,J,TITLE) 01328
ISN 0003 DOUBLE PRECISION A(I,J) 01329
ISN 0004 C 01330
C 01331
C MATRIX OUTPUT 01332
C 01333
ISN 0005 WRITE(6,1) TITLE 01334
ISN 0006 1 FORMAT(/,5X,'MATRIX OUTPUT FOR ',A4,/) 01335
ISN 0007 2 FORMAT(5X,'ROW',5X,'COL',5X,'VALUE',/) 01336
ISN 0008 DO 100 K = 1,1 01337
ISN 0009 DO 100 L = I,J 01338
ISN 0010 IF(A(K,L) .EQ. 0.) GO TO 100 01339
ISN 0011 WRITE(6,3) K,L,A(K,L) 01340
ISN 0012 100 CONTINUE 01341
ISN 0013 3 FORMAT(5X,I5,3X,I5,3X,E12.5) 01342
C 01343
ISN 0015 RETURN 01344
ISN 0016 END 01345
```

**OPTIONS IN EFFECT** NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL

**OPTIONS IN EFFECT** SOURCE EBCDIC NOLIST NODATE OBJECT NODUMP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)

**STATISTICS**

- SOURCE STATEMENTS = 15,
- PROGRAM SIZE = 542,
- SUBPROGRAM NAME = MATPRN

**STATISTICS**

- NO DIAGNOSTICS GENERATED

*END OF COMPILATION***
SUBROUTINE MINV(A,N,D,L,H)

PURPOSE
INVERT A MATRIX

DESCRIPTION OF PARAMETERS
A - INPUT MATRIX, DESTROYED IN COMPUTATION AND REPLACED BY
RESULTANT INVERSE.
N - ORDER OF MATRIX A
D - RESULTANT DETERMINANT
L - WORK VECTOR OF LENGTH N
H - WORK VECTOR OF LENGTH N

REMARKS
MATRIX A MUST BE A GENERAL MATRIX

SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
NONE

METHOD
THE STANDARD GAUSS-JORDAN METHOD IS USED. THE DETERMINANT IS ALSO CALCULATED. A DETERMINANT OF ZERO INDICATES THAT THE MATRIX IS SINGULAR.

DIMENSION A(I),L(I),M(I)

IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION STATEMENT WHICH Follows.

DOUBLE PRECISION A,D,BIGA,HOLD

THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS ROUTINE.

THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS. DABS IN STATEMENT 10 MUST BE CHANGED TO DDABS.

------------------------------------------------------------------
SEARCH FOR LARGEST ELEMENT

D = 1.0  

IJ = I

DO 10 K = 1, N

NK = NK + K

L(K) = I

M(K) = K

BIGA = A(KK)

DO 20 J = K, N

IZ = I*(J - 1)

DO 10 I = K, N

IJ = IZ + I

IF (DABS(BIGA) - DABS(A(IJ))) > 15,20,20

BIGA = A(IJ)

M(K) = J

INTERCHANGE ROWS

J = L(K)

IF (J - K) > 35, 35, 25

DO 25 K = 1, N

DO 20 I = 1, N

KI = K + I

INTERCHANGE COLUMNS

J = L(K)

IF (J - K) > 35, 35, 38

DO 38 JP = N*(J - 1)

DO 30 J = 1, N

JK = JK + J

DO 40 A(JI) = HOLD

DIVIDE COLUMN BY MINUS PIVOT (VALUE OF PIVOT ELEMENT IS CONTAINED IN BIGA)

DO 45 IF (BIGA) < 48, 46, 48

DO 44 0 = 0.0

RETURN

DO 48 I = 1, N

IF (I - K) > 50, 55, 50

IK = IK + I

CONTINUE

REDUCE MATRIX

DO 65 I = 1, N
DIVIDE ROW BY PIVOT

PRODUCT OF PIVOTS

REPLACE PIVOT BY RECIPROCAL

FINAL ROW AND COLUMN INTERCHANGE

*OPTIOTIS HI EFFECT*SCURCE EBCDIC NO LIST HODECK OBJECT NOMAP NOFORMAT GOSTNT NOXREF NOALe NOANSF TERM IBM FLAG(1)

*STATISTICS* SOURCE STATEMENTS = 91, PROGRAM SIZE = 1502, SUBPROGRAM NAME = MINV
**REQUESTED OPTIONS:** AUTO.DOBL(3), SOURCE
**OPTIONS IN EFFECT:** NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTO.DOBL(3)

**SOURCE** EBCDIC NO.LIST NO.DEC23 OBJECT NO.MAP NO.FORMAT GOSTHT NO.XREF NO.ALC NO.ANSF TERM IBM FLAG(I)

---

SUBROUTINE MODMAS(I,EX,NR,RMASS)

IMPLICIT REAL*8(A-H, O-Z)

C MODAL MASS FOR THE I MODE

COMMON /STIFF/ SK(12,12),SXX(12,12,21),SMM(6,6,21)

DIMENSION EX(12,21,2),RMASS(3),DISP(6,1),SMM(6,6),DISPT(1,6)

DIMENSION OISP(6,1)

RMASS(I) = 0.

DO 100 J = 1, NR

C CONVERT THE W137 EIGENVECTOR, EX, TO STABE Format

C CALL SHAPE(J,EX,DISP)

NOW MULTIPLY - DISPT * SMM * DISP

Do 110 K = 1, 6

DO 110 L = 1, 6

110 SMM(K,L) = SMM(K,L)

Do 120 K = 1, 6

DISPT(1,K) = DISP(K,1)

C CALL MATMPY(SMM,DISP,D1,6,6,1)

C CALL MATMPY(DISPT,D1,D2,1,6,1)

RMASS(I) = RMASS(I) + D2

RETURN

END

---

*OPTIONS IN EFFECT* NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTO.DOBL(3)

*STATISTICS* SOURCE STATEMENTS = 18, PROGRAM SIZE = 1034, SUBPROGRAM NAME =MODMAS

*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION *****

3016K BYTES OF CORE NOT USED
SUBROUTINE MODSTR (I,IC,NR)

C THIS ROUTINE IS THE DRIVER FOR THE MODAL STRESS CALCULATION

COMMON /CALCA/ FILL1(1668),MSL2(1588),EX(12,21,2),SS(12,2)
COMMON /STRS/ RBE(12,12,25,21),FOUT(12,21),SL(25,21),SN(25,21)
COMMON /MODAL/ SIGMA(3,6,25)
COMMON /MODAL1/ STRESS(3,3,8,25),FACTOR(3),RMASS(3)
COMMON /BLK1/ DIAGON(100),FNI(100),FILLI(300)
DIMENSION DSP(6,1),BFORCE(6,1),DISP(1,6),PLOAD(1,1)

C CONVERT 1N37 EIGENVECTOR TO STABLE FORMAT FOR STATIONS 1 AND 2
C
CALL SHAPE(2,EX,DISP)
CALL MATPRMN(DISP,6,1,'DISP')
DO 100 J = 1,6
FOUT(J,2) = DISP(J,1)
100 CONTINUE

C
CALL SHAPE(3,EX,DISP)
CALL MATPRMN(DISP,6,1,'DISP')
DO 110 J = 1,6
FOUT(J,3) = DISP(J,1)
110 CONTINUE

C CALCULATE ROOT STRESS FOR MODE I
C
CALL ZSTRES(2,NC,SHAX)
C
STORE THE STRESS VALUES FOR MODE I IN STRESS(3,3,8,25)
C
DO 120 JJ = 1,NC
DO 120 J = 1,8
120 CONTINUE

C CALCULATE THE MODAL MASS FOR MODE I
C
CALL MODMAS (I,EX,NC,SHAX)

C CALCULATE THE BIRD FORCE AND MOMENT
C
CALL BIRDF(NR,NC,ISTA,BFORCE,SQUASH)

C FIND THE STABLE MODE SHAPE AT STATION ISTA
C
CALL SHAPE ( ISTA,EX,DISP )
CALL MATPRMN(DISP,6,1,'DISP')

DO 130 J = 1,6
DISP(J,7) = DISP(J,1)
130 CONTINUE

C
C MODAL LOAD, PLOAD
CALL MATPRN ( PLOAD, 1, 1, 'PLOAD' )

PARTICIPATION FACTOR, FACTOR

PI = PLOAD (1, 1)

TO = SQUASH

FREQ = FNI (I) * 2. * 3.141593

FACTOR (I) = PI * TO / RMASS (I) / FREQ

CALL MATPRN ( FNI, 100, 1, 'FNI' )

CALL MATPRN ( RMASS, 3, 1, 'RMASS' )

CALL MATPRN ( FACTOR, 3, 1, 'FACT' )

RETURN

END
SUBROUTINE PARTN(SK,II,JJ,SK11,SK12,SK21,SK22,KK,LL)

THIS ROUTINE PARTITIONS THE SK MATRIX INTO SK11,SK12,
SK21 AND SK22.

DIMENSION SK(II,JJ),SK11(KK,LL),SK12(KK,LL),SK21(KK,LL)

DO 100 I = 1,6
   DO 100 J = 1,6
      SK11(I,J) = SK(I,J)
   CONTINUE

DO 101 I = 1,6
   L = 1
   DO 101 J = 7,12
      SK12(I,L) = SK(I,J)
      L = L + 1
   CONTINUE

DO 102 I = 7,12
   K = I - 6
   DO 102 J = 1,6
      SK21(K,J) = SK(I,J)
   CONTINUE

DO 103 I = 7,12
   K = I - 6
   DO 103 J = 7,12
      SK22(K,L) = SK(I,J)
      L = L + 1
   CONTINUE

CALL MATPRN(SK11,6,6,'SK11')
CALL MATPRN(SK12,6,6,'SK12')
CALL MATPRN(SK21,6,6,'SK21')
CALL MATPRN(SK22,6,6,'SK22')

RETURN
END
C SUBROUTINE PPI(II)
C
C THIS ROUTINE SETS UP THE REORDER MATRICES
C P AND P1 ARE CONSTANT AND DO NOT CHANGE
C
C IMPLICIT REAL*8(A-H,O-Z)

ISH 0003 COMMON /REORDER/ P(12,12),PT(12,12)

C INITIALIZE

ISH 0005 DO 100 I = 1,12
ISH 0006 DO 100 J = 1,12
ISH 0007 100 P(I,J) = 0.

C SET NON-ZERO VALUES OF P

ISH 0009 IF(I.I.GT.1) GO TO 50

ISH 0010 P(1,1) = 1.
ISH 0011 P(2,5) = 1.
ISH 0012 P(3,9) = 1.
ISH 0013 P(4,2) = 1.
ISH 0014 P(5,10) = 1.
ISH 0015 P(6,6) = 1.
ISH 0016 P(7,3) = 1.
ISH 0017 P(8,7) = 1.
ISH 0018 P(9,11) = 1.
ISH 0019 P(10,4) = 1.
ISH 0020 P(11,12) = 1.
ISH 0021 P(12,8) = 1.
ISH 0022 GO TO 51
ISH 0023 50 P(1,7) = 1.
ISH 0024 P(2,6) = 1.
ISH 0025 P(3,12) =-1.
ISH 0026 P(4,9) =-1.
ISH 0027 P(5,11) = 1.
ISH 0028 P(6,10) = 1.
ISH 0029 P(7,1) = 1.
ISH 0030 P(8,6) =-1.
ISH 0031 P(9,2) = 1.
ISH 0032 P(10,5) = 1.
ISH 0033 P(11,3) =-1.
ISH 0034 P(12,4) = 1.
ISH 0035 51 CONTINUE

C NOW DETERMINE THE TRANSPOSE MATRIX,PT

ISH 0036 DO 200 I = 1,12
ISH 0037 DO 200 J = 1,12
ISH 0038 200 PT(I,J) = P(J,I)
ISH 0039 200 CONTINUE

C CALL MATPRN(P,12,12,' P ')
C CALL MATPRN(PT,12,12,'PT ')
C
SUBROUTINE RBE2(X,Y,Z,ISEC,IBEAM,NC,BLOCAL)

THIS ROUTINE CALCULATES THE RIGID BODY TRANSFORMATION

MATSIRC RBL12 AND RBL1

RBL12 IS WRITTEN IN THE LOCAL BEAM COORD. SYSTEM

IMPLICIT REAL*8(A-H,O-Z)

COMMON /RBE/ RBL12(12,12),RBL4(12,12)

COMMON /HARP/ WS(26,21),RH(25,21),RT(25,21)

COMMON /XSC/ XSC(21),YSC(21),ZSC(21),ALPHA(21),SECIP(21),SECA(21)

DIMENSION RBL(6,6),RL(6),RH(6)

DIMENSION X(1),Y(1),Z(1),XG(3),XGB(3),XAH(3)

DIMENSION XA(3),XB1(3),XB2(3),XGB(3),BLOCAL(3,3)

DIMENSION V1(3),V2(3),VL1(3),VL2(3)

FORCING RBL12

TRANSFORM GLOBAL COORD. POINTS TO SECTION SYSTEM

LAYER 1-BEAM COORD IS XAI,SECTION COORD IS XB1

IP = (ISEC-1) * NC + IBEAM

POSITION VECTORS FOR LAYERS 1 AND 2, V1 AND V2

CALL MATMPY(BLOCAL,V1,VL1,3,3,1)

CALL MATMPY(BLOCAL,V2,VL2,3,3,1)

ROTATE FROM GLOBAL SYSTEM TO BEAM LOCAL

CALL MATMPY(BLOCAL,V1,VL1,3,3,1)

CALL MATMPY(BLOCAL,V2,VL2,3,3,1)

DO 102 I = 1,12
    DO 102 J = 1,12
        RBL1(I,J) = 0.
    RBL2(I,J) = 0.

DO 103 I = 1,12
    RBL1(I,I) = 1.
*VERSION 1.3.0 (01 MAY 80)  RBE2  SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 82.141/10.49.55  PAGE 2

ISH 0033   103 RBL12(I,I) = 1.

C

ISH 0034   RBL12(I,4) = W5(IDEAM,ISEC) / XL
ISH 0035   RBL12(I,5) = Z1
ISH 0036   RBL12(I,6) = Y1
ISH 0037   RBL12(I,10) = -RBL12(I,4)
ISH 0038   RBL12(2,4) = -Z1
ISH 0039   RBL12(2,6) = X1
ISH 0040   RBL12(3,4) = Y1
ISH 0041   RBL12(3,5) = -X1
ISH 0042   RBL12(5,4) = RT(IDEAM,ISEC) / XL
ISH 0043   RBL12(5,10) = -RBL12(5,4)
ISH 0044   RBL12(6,4) = RH(IDEAM,ISEC) / XL
ISH 0045   RBL12(6,10) = -RBL12(6,4)
ISH 0046   RBL12(7,4) = W5(IDEAM,ISEC+1) / XL
ISH 0047   RBL12(7,10) = -RBL12(7,4)
ISH 0048   RBL12(7,11) = Z2
ISH 0049   RBL12(7,12) = -Y2
ISH 0050   RBL12(6,10) = -Z2
ISH 0051   RBL12(6,12) = X2
ISH 0052   RBL12(9,10) = Y2
ISH 0053   RBL12(9,11) = -X2
ISH 0054   RBL12(11,4) = RT(IDEAM,ISEC+1) / XL
ISH 0055   RBL12(11,10) = -RBL12(11,4)
ISH 0056   RBL12(12,4) = RH(IDEAM,ISEC+1) / XL
ISH 0057   RBL12(12,10) = -RBL12(12,4)

C CALL MATPAN(RBL12,12,12,'RBL12')
ISH 0058   RBL12(1,5) = RBL12(1,5)
ISH 0059   RBL12(1,6) = RBL12(1,6)
ISH 0060   RBL12(2,4) = RBL12(2,4)
ISH 0061   RBL12(2,6) = RBL12(2,6)
ISH 0062   RBL12(3,4) = RBL12(3,4)
ISH 0063   RBL12(3,5) = RBL12(3,5)
ISH 0064   RBL12(7,11) = RBL12(7,11)
ISH 0065   RBL12(7,12) = RBL12(7,12)
ISH 0066   RBL12(8,10) = RBL12(8,10)
ISH 0067   RBL12(8,12) = RBL12(8,12)
ISH 0068   RBL12(9,10) = RBL12(9,10)
ISH 0069   RBL12(9,11) = RBL12(9,11)

C CALL MATPAN(RBL12,12,12,'RBL12')
ISH 0070   IF( ISEC .EQ. 2 ) GO TO 300
ISH 0072   IF( ISEC .GT. 1 ) RETURN

C

C RESET RBL12 FOR EXTENDED NECK REGION - NO WARPING

C

ISH 0074   DO 200 I = 1,12
ISH 0075   DO 200 J = 1,12
ISH 0076   200 RBL12(I,J) = RBL1(I,J)
ISH 0077   300 CONTINUE
ISH 0078   DO 305 I = 1,6
ISH 0079   DO 305 J = 1,12
ISH 0080   305 RBL12(I,J) = RBL1(I,J)

C

ISH 0081   RETURN
ISH 0082   END
+VERSION 1.3.0 (01 MAY 80) RBE2 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 82.141/10.49.55

*STATISTICS* SOURCE STATEMENTS = 81, PROGRAM SIZE = 1496, SUBPROGRAM NAME = RBE2

*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION *****

3004K BYTES OF CORE NOT USED
BEGIN THE SECTION STIFFNESS SOLUTION FOR SECTION I

FORM THE SECTION COORDINATE SYSTEM, TS

TS FOR LAYER ONE

TS FOR LAYER TKO

LOCAL X UNIT VECTOR

II = I + 1

IF(I.EQ. 1) II = 1

IF(I.EQ. NRM1) II = NRM1

XSX = XSC(II+1) - XSC(II)

XSY = YSC(II+1) - YSC(II)

XSZ = ZSC(II+1) - ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)

WRITE(6,4) XSC(II),YSC(II),ZSC(II)
LOCAL Z UNIT VECTOR (INITIALLY)

ZSX = 0.
ZSY = DCOS (ALPHA(II))
ZSZ = DSIN (ALPHA(II))

LOCAL Y UNIT VECTOR YS
YSX = ZSY * XSZ - ZSZ * XSY
YSY = ZSZ * XSX - ZSX * XSZ
YSZ = ZSX * XSY - ZSY * XSX

WRITE(6,4) YSNAG, I, II

YSX = YSX 1
YSY = YSY 1
YSZ = YSZ 1

REDEFINE THE LOCAL ZS VECTOR, ZS = XS CROSS YS

IF(I.EQ.UNI .AND. NRNI .NE. 1) II = II + 1

TS(3,I,II) = XSY * YSZ - XSZ * YSY
TS(3,2,II) = X5Z * YSX XSX * YSZ
TS(3,3,II) = X5X * YSZ - X5Y * YS( I.H.EQ.1) II = II + 1

IF(NRNI .EQ. 1 .AND. II .EQ. 2) GO TO 98
IF(I.EQ.1 .AND. II .EQ. 2) GO TO 99

WRITE(6,2) I
C2 FORMAT(/,5X/'TS MATRIX FOR SECTION',I5,/) II = I+1
C DD 161 L = I,II
C WRITE(6,162) L
C162 FORMAT(5X,'SECTION NODE',I5)
C161 WRITE(6,1) ((TS(J,K,L),K=1,3),J=1,3)
C1 FORMAT(IX,3(E12.5,2X),/)
I
...

BEGIN LOOP FOR EACH BEAM

DO 200 J = 1, NC

CALCULATE THE BEAM COORDINATE SYSTEM, BLOCAL(3,3)

IP = (I-1) * NC + J

IPP = IP + NC

CALL BCOORD(IP, IPP, BLOCAL)

CALL MATPRN(BLOCAL, 3, 3, 'BLOC')

CALCULATE THE BEAM MATERIAL STIFFNESS MATRIX, E

CALL ESTIFF(I, NC, NR, J)

DETERMINE THE BEAM STIFFNESS MATRIX, BK

THE BEAM CENTRIFUGAL RESTORING MATRIX, BCHSG

CALL BEAM(NC, J, BLOCAL, BCHSG, RPH)

THE BEAM(J) STIFFNESS MATRIX IS NOW BUILT AND MUST

BE SUMMED INTO THE SECTION (I) STIFFNESS MATRIX, SK

FORM THE TBS TRANSFORMATION MATRIX

TBS = BLOCAL * TS(TRANSPOSE)

DO 103 K = 1, 3

DO 103 L = 1, 3

TS1(K, L) = TS1(L, K, I)

TS2(K, L) = TS2(L, K, I+1)

CALL MATMYP(BLOCAL, TS1, 3, 3)

CALL MATMYP(BLOCAL, TS2, 3, 3)

WRITE(6, 3) I

WRITE(6, 1) ((TS1(K, L), L=1,3), K=1,3)

WRITE(6, 1) ((TS2(K, L), L=1,3), K=1,3)

WRITE(6, 1) ((TS3(K, L), L=1,3), K=1,3)

THE RIGID BODY TRANSFORMATION MATRIX, RBL2

ALSO CALCULATE THE RIGID BODY MATRIX FOR STATIC LOADS WHICH

DOES NOT INCLUDE HARPPING- ONLY EQUILIBRIUM, RBL1

CALL REBE2(X, Y, Z, I, J, NC, BLOCAL)

RESTRAINED HARPPING AT THE ROOT

IF(I .EQ. 1) GO TO 190

GO TO 191

190 HARPP = 1

IF(NHARPP .GT. 1) GO TO 191

WRITE(6, 77)

FORMAT(/, 5X, '*** PLANES REMAIN PLANE ROOT CONDITION ***', I5, /)

RBL2(1, 4) = 0.

RBL2(1, 10) = 0.
**DEFINE THE TBSS MATRIX**

```fortran
C CONTINUE
C DEFINE THE TBSS MATRIX
C
ISH 0107 DO 105 K = 1,12
ISH 0108 DO 105 L = 1,12
ISH 0109 105 TBSS(K,L) = 0.
ISH 0110 IROW = 1
ISH 0111 DO 106 K = 1,4
ISH 0112 DO 106 L = 1,3
ISH 0113 ICOL = (K-1) * 3 + 1
ISH 0114 DO 108 M = 1,3
ISH 0115 TBSS(IROW,ICOL) = TBSS(L,M)
ISH 0116 IF(K .GT. 2) TBSS(IROW,ICOL) = TBSS(L,M)
ISH 0118 108 ICOL = ICOL + 1
ISH 0119 106 IROW = IROW + 1
C WRITE(6,51) I
C5 FOMAT(15,'TBSS MATRIX FOR SECTION',I5)
C CALL MATPRN(TBSS,12,12,'TBSS')
C CALCULATE THE R MATRIX, R = RBL12 * TBSS
C CALCULATE THE RF MATRIX, RF = RBL1 * TBSS
C
ISH 0120 CALL MATPRNY(RBL12,TBSS,R,12,12,12)
C CALL MATPRNY(RBL12,RF,12,12,12,RF)
ISH 0121 CALL MATPRNY(R1,12,12,RF)
C FILL THE RBE ARRAY FOR LATER USE IN THE STATIC ANALYSIS
C ALSO STORE THE INFO FOR BEAM DIMENSIONS
C
ISH 0122 DO 80 K = 1,12
ISH 0123 DO 80 L = 1,12
ISH 0124 80 RBE(K,L,J,I) = R(K,L)
ISH 0125 SL(J,I) = BSPAN(J)
ISH 0126 SW(J,I) = BWDTH(J)
C CALCULATE R1 = TSS * RBL1 * TBSS
C IT HAS ALSO BEEN SHOWN THAT R1 = R(TRANSPOSE)
C ALSO TRANSPOSE RF AND STORE IN RBL1
C
ISH 0127 DO 201 K = 1,12
ISH 0128 DO 201 L = 1,12
ISH 0129 RBL1(K,L) = RF(L,K)
ISH 0130 201 R1(K,L) = R(L,K)
C CALL MATPRNY(R1,12,12,'R1')
C CALL MATPRNY(RBL1,12,12,'RFT')
C BEAM(J) CONTRIBUTION TO SECTION (I) STIFFNESS IS:
C SK(K,L) = R1(K,M) * BK(H,N) * R(N,L)
C
ISH 0131 CALL MATPRNY(BK,R1,12,12,12)
ISH 0132 CALL MATPRNY(R1,12,SKOLD,12,12,12)
C BEAM(J) CONTRIBUTION TO SECTION(I) MASS , SM
```
C SL(K,L) = R1(K,M) * BN(M,N) * R(N,L)
C
C ISN 0133
CALL MATMPY(EM,R,01,12,12,12)
CALL MATMPY(R1,01,02,12,12,12)
C
C ISN 0134
CALL MATMPY(BEAMJ,01,12,12,12)
CALL MATMPY(EM,R,01,BCMG,12,12,12)
C
C ISN 0135
CALL MATMPY(BCMGG,R,01,12,12,12)
CALL MATMPY(R1,01,BCMGG,12,12,12)
C
C ISN 0136
CALL MATMPY(BCMGG,R,D1,12,12,12)
CALL MATMPY(R1,01,BCMGG,12,12,12)
C
C ISN 0137
CALL MATMPYIRB1,SILK,FF,12,12,1)
DO 170 K = 1,6
FSEC(K,I) = FBEAM1(K,I)
FSEC(K+6,1) = FBEAM2(K,I)
C
C ISN 0138
CALL MATMPYIRB1,FTOT1,6,1,FTOT1
CALL MATMPYIRB1,FTOT2,6,1,FTOT2
C
C ISN 0139
IP = (I-1) * NC + J
CALL FRPM(J,IP,X,Y,BMASS,RPM,FBEAM1,BLOCAL)
CALL MATPRN(FBEAM1,6,1,'FBEA1')
C
C ISN 0140
CALL FRPM(J,IP,X,Y,BMASS,RPM,FBEAM2,BLOCAL)
CALL MATPRN(FBEAM2,6,1,'FBEA2')
C
C ISN 0141
DO 170 K = 1,6
FSEC(K,1) = FBEAM1(K,1)
FSEC(K+6,1) = FBEAM2(K,1)
C
C ISN 0142
CALL MATMPY(R1L,FSEC,FF,12,12,1)
DO 175 K = 1,6
FTOT1(K,1) = FTOT1(K,1) + FF(K,1)
C
C ISN 0143
DO 175 K = 1,6
FTOT2(K,1) = FTOT2(K,1) + FF(K+6,1)
C
C ISN 0144
CALL MATPRN(FF,12,1,'FF')
C
C ISN 0145
CALL MATPRN(FF,12,1,'FF')
C
C ISN 0146
CALL MATPRN(FF,12,1,'FF')
C
C ISN 0147
CALL MATPRN(FF,12,1,'FF')
C
C ISN 0148
DO 101 K = 1,12
DO 101 L = 1,12
C
C ISN 0149
CMG(K,L) = CMG(K,L) + BCMGG(K,L)
SH(K,L) = SH(K,L) + D2(K,L)
C
C ISN 0150
SK(K,L) = SK(K,L) + SKOLD(K,L)
C
C ISN 0151
WRITE(6,13) I,J
C C FORMAT(5X,'THE SECTION STIFFNESS AND MASS FOR SECTION',I5,/) C C ISN 0152
WRITE(6,13) I,J
C C FORMAT(5X,'FOR BEAMS ONE THRU ',I5,/) C C ISN 0153
WRITE(6,13) I,J
C C FORMAT(5X,'FOR WARPING FUNCTION UPDATE TO ACCOUNT FOR RESTRAINT EFFECTS') C
A = 0.5 # SMB (2)

XDIST = XSC(I) - XSC(2)

ARG = - XDIST / A

WARPI = 1. - DEXP( ARG )

IF(I .LE. 2) WARPI = 1.

XDIST = XSC(I+1) - XSC(2)

ARG = - XDIST / A

WARPI = 1. - DEXP( ARG )

IF(I .EQ. 1) WARPI = 1.

SK(4,4) = SK(4,4) / WARPI

SK(4,10) = SK(4,10) / WARPI

SK(10,4) = SK(10,4) / WARPI

SK(10,10) = SK(10,10) / WARPI

C REPLACE EXTENDED NECK STIFFNESS WITH RECTANGULAR SECTION NECK

C IF(I .EQ. 1 .AND. NCO .EQ. 1) CALL EXNECK( STIB(2)

C INSERT SK INTO SKK FOR STORAGE

DO 110 J = 1,12

DO 110 K = 1,12

CMG(J,K,I) = CMG(J,K)

110 SKK(J,K,I) = SK(J,K)

C FILL IN THE MASS MATRIX, SMB, AND THE LOAD VECTOR, F.

DO 115 J = 1,12

F(J,I) = 0.

DO 115 J = 1,12

SM(J,K) = SM(J,K)

115 S1200(J,K) = S1200(J,K)

GO TO 126

125 CONTINUE

DO 118 J = 1,12

F(J,I) = 0.

DO 118 J = 1,12

SM(J,K) = SM(J,K)

118 S1200(J,K) = S1200(J,K)

GO TO 126

DO 115 J = 1,12

F(J,1) = 0.

DO 115 J = 1,12

SM(J,K) = SM(J,K)

115 S1200(J,K) = S1200(J,K)

GO TO 126
ISH 0201 SIM(J,K,I) = SM2OLD(J,K) + SM1(J,K)
ISH 0202 118 SM2OLD(J,K) = SM2(J,K)
ISH 0203 126 IF(I .NE. NRNL) RETURN
ISH 0205 DO 119 J = 1,6
ISH 0206 DO 119 K = 1,6
ISH 0207 F(J,I+1) = 0.
ISH 0208 F(J+6,I+1) = FBEAM2(J,1)
ISH 0209 119 SIM(J,K,I+1) = SM2(J,K)
C ADD A TIP MOMENT
C F(10,I+1) = 1000.
ISH 0210 RETURN
ISH 0211 END

*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(DBL4)
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NOMAP NOFORMAT GOSTMR NOXREF NOALC NOAHSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 210, PROGRAM SIZE = 17296, SUBPROGRAM NAME = SECTN
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPILATION *****

2964K BYTES OF CORE NOT USED
SUBROUTINE SHAPE (NODE, EX, DISP)

IMPLICIT REAL*8 (A-H, O-Z)

GIVEN THE W137 EIGENVECTOR CALCULATE THE STABLE SYSTEM EIGENVECTORS FOR A GIVEN NODE

DIMENSION EX(12, 21, 2), DISP(6, 1), W137(6, 1), PPT(6, 6)

EXTRACT THE W137 EIGENVECTOR COMPONENT FOR THE NODE

DO 100 I = 1, 6

100 W137(I, 1) = EX(I+6, NODE, 1)

CALL MATPIN(W137, 6, 1, 'W137')

TRANSFORMATION ARRAY, PPT

CALL PP1(2)

DO 110 I = 1, 6

110 PPT(I, J) = PT(I, J+6)

CALL MATPIN(PPT, 6, 6, 'PPT')

CALL MATMPY(PPT, W137, DISP, 6, 6, 1)

RETURN

END

**OPTIONS IN EFFECT** NAME = SHAPE

**REQUESTED OPTIONS:** AUTODBU (DB4), SOURCE

**OPTIONS IN EFFECT:** NAME (MAIN) OPTIMIZE (3) LINECOUNT (60) SIZE (MAX) AUTODBU (DB4)

**SOURCE EBCDIC NOLIST NODECK OBJECT NOLIST NOSHORT GOSTMT NOXREF NOLC NOANSF TERM IBM FLAG(I)***

**STATISTICS**

SOURCE STATEMENTS = 13, PROGRAM SIZE = 832, SUBPROGRAM NAME = SHAPE

**STATISTICS** NO DIAGNOSTICS GENERATED

**** END OF COMPILATION ****

3016K BYTES OF CORE NOT USED
SUBROUTINE SMULT(NR)
C A SLAVE ROUTINE TO PERFORM THE HOLZER STIFFNESS MATRIX STRING MULTIPLICATION FOR THE STATIC ANALYSIS
C
C
INITIALIZE SH MATRIX
C
NRM1 = NR - 1
DO 100 I = 1,12
DO 100 J = 1,12
100 SH(I,J,1) = SKPP(I,J,NRM1)
C IF(NRMI .EQ. 1) RETURN
C
DO 200 I = 2,NRMI
IP = NRM1 - I + 1
DO 110 J = 1,12
DO 110 K = 1,12
110 SH(I,J,K) = SKPP(J,K,IP)
C CALL MATMPY(D1,D2,D3,12,12,12)
C
DO 200 J = 1,12
DO 200 K = 1,12
200 SH(J,K,I) = D3(J,K)
C WRITE(6,10)
C10 FORMAT('IN SMULT FOR STRING OPERATION')
C CALL MATPRN(D3,12,12,12)
C
CONTINUE
C
RETURN
SUBROUTINE STABLE(NR,NC,RPM1,RPM2,RPM3,FREQ,ILoop,ISERCH)

THIS PROGRAM IS THE DRIVER FOR A STACKED BEAM ANALYSIS
A STATIC ANALYSIS IS PERFORMED WITH THE HOLZER METHOD
WRITTEN BY NICK MARTIN AND KEN BROWN

IMPLICIT REAL*8(A-H,O-Z)
COMITION /INPUTT/ XC 1000, YHOOO, ZC 1000, T( 1000)
COMITION /SC/ CC6,6), RPM
COMITION /STIFF/ SK(12,12), SSKK(12,12,21), SMIX(6,6,21)
COMITION /DSTIFF/ CNGG(12,12,20), DSKK(12,12,20)
COMITION /SC/ XSC(21), YSC(21), ZSC(21), ALPHAC(21), SECIP(21), SECA(21)
COMITION /HSTIFF/ SKP(12,12), SKPP(12,12,21)
COMITION /BLKAC/ AA(12,12,20), BB(12,12,20), CC(12,12,20),
STAHAT(12,12,20)
COMITION /REORD/ F(12,12), PT(12,12)
COMITION /STAT/ SH(12,12,20), F(12,21)
COMITION /HOLE/ HOLE
COMITION /SHAX/ SHAX(8), SMAX2(8), SMAX3(8), SMAXL(8), SMAXTE(8)
COMITION /HODAL/ SIGMA(3,6,25)
COMITION /FAIL/ XXX(42), TSAI(8,25)
DIMENSION SKPP(12,12,20), SKP(12,12,20)
DIMENSION DS(12,12), CM(12,12), SKKO(12,12,20), FOLD(12,21)
DIMENSION DS(12,12)

DO 50 I = 1,6
DO 50 J = 1,6
50 C(I,J) = 0.0
C

IF( LLoop .GT. 1 ) GO TO 999
LOOP = 0
RPM = RPM1

FOR CHECK OUT RUNS IN W137 THIS CALL TO INPUT IS MADE, REMOVE
AFTER SUITABLE INPUT IS AVAILABLE FROM W137

CALL INPUT(NR,NC)
CALL CORD2R(X,Y,Z,NR,NC)
CALCULATE THE LAMINA STRESS-STRAIN RELATION ,QIJ

CALL LAMINA
DEFIN THE REORDER MATRICES P AND P1

CALL PPI(1)
SET UP THE BOUNDARY CONDITIONS FOR A FIXED ROOT CONDITION

DO 50 I = 1,6
DO 50 J = 1,6
50 CI(J) = 0.0
C

FIND EACH SECTION STIFFNESS AND CORRESPONDING HOLZER STIFFNESS
ALSO DETERMINE THE NODEH LUMPED MASS MATRICES
CALL HOLZER(1,NR,NC,RPM1)

PERFORM THE STATIC ANALYSIS

WRITE(6,100) RPM1

FORMAT(//,5X,'STATIC ANALYSIS, RPM =',E12.5,//)

CALL STATIC(NR,NC,1)

SET THE INITIAL VALUE OF SKKOLD - THE ORIGINAL STIFFNESSES

NRM1 = NR - 1
DO 201 I = 1,NRM1
DO 201 J = 1,12
DO 201 K = 1,12
SKKOLD(J,K,I) = SKK(J,K,1)
DO 203 J = 1,12
DO 203 I = 1,NR
FOLD(J,I) = F(J,I)

LOCAL BIRD INGESTION PARAMETER CALCULATION

CALL FOD( NR,NC,RPM1,ALPHA )

IF(ISERCH .GT. 1) GO TO 998

UPDATE THE STIFFNESS MATRICES FOR DIFFERENTIAL STIFFNESS AND CENTRIFUGAL RESTORATION - ALSO UPDATE THE LOAD VECTOR, F

LOOP = LOOP + 1
RATIO = 1.
WRITE(6,500) ISERCH,LOOP,RATIO
FORMAT(//,5X,**',E12.5,**',/)

DO 400 I = 1,NRM1
DO 400 J = 1,12
DO 400 K = 1,12
IF(LOOP .EQ. 1) SKPP1(J,K,I) = SKK(J,K,1)
IF(LOOP .EQ. 2) SKPP2(J,K,I) = (DSKK(J,K,1) - CMGG(J,K,1))
CONTINUE

IF(LOOP .EQ. 1) GO TO 888

CALL HOLZER(2,NR,NC,RPM1)

FOR RPM1 - HOLZER STIFFNESS MATRIX IS SKPP1(12,12,20)
RPM2 - HOLZER STIFFNESS MATRIX IS SKPP2(12,12,20)

DO 400 I = 1,NRM1
DO 400 J = 1,12
DO 400 K = 1,12
IF(LOOP .EQ. 1) SKPP1(J,K,I) = SKPP(J,K,I)
IF(LOOP .EQ. 2) SKPP2(J,K,I) = SKPP(J,K,I)
CONTINUE

IF(LOOP .EQ. 1) GO TO 888
C RESET THE HOLZER MATRIX FOR THE STATIC ANALYSIS

C WRITE(6,501)
C FORMAT(/,'***RESET SKPP MATRIX***',/)

C501 FORMAT(/,'***RESET SKPP MATRIX***',/)

ISN 0068 DO 410 I = 1,NR1
ISN 0069 DO 410 J = 1,12
ISN 0070 DO 410 K = 1,12
ISN 0071 410 SKP(J,K,I) = SKPP(J,K,I)

C STATIC ANALYSIS WITH PRESTRESS INCLUDED

ISN 0072 WRITE(6,101) RPM1,RPM2
ISN 0073 101 FORMAT(/,'STATIC ANALYSIS WITH PRESTRESS,','',/)

1'RPM1 = ','E12.5,5X,'RPM2 = ','E12.5,/'

C CALL STATIC(NR,NC,2)
C PERFORM THE STRESS ANALYSIS
C CALL ZSTRES ( 2,NC,SMAX1 )
C CALCULATE TSAI-WU FOR LE AND TE ROOT STRESS - SMAXLE,SMAXTE
C CALL TSAIWU(SIGMA,HC)

DO 330 I = 1,8
SHAXLE(I) = TSAI(I,NC)
330 SI1AXTE(I) = TSAI(I,NC)

C HOLLOW SECTION STRESS
C IF(IHOLE .NE. 0) CALL ZSTRES(IHOLE,NC,SMAX2)

IHOLE1 = IHOLE + IHOLE2
IF(IHOLE .NE. 0) CALL ZSTRES(IHOLE1,NC,SMAX3)
C IF(IHOLE .EQ. 0) GO TO 501
C
DO 500 I = 1,8
IF(ISI1AX3(I) .GT. ISI1AX2(I)) SMAX2(I) = ISI1AX3(I)
500 CONTINUE
501 CONTINUE

C FORM MASS * STIFFNESS PRODUCTS TO BE SENT TO W137 SOLVER
C -MUST CHANGE SIGN ON 21 AND 22 QUADRANTS OF SKPP FIRST
C SKPP = HOLZER STIFFNESS MATRICES FOR EACH SECTION

ISN 0082 IF(IHOLE .NE. 0) CALL ZSTRES(IHOLE,NC,SMAX2)
IHOLE = IHOLE + 1
IF(IHOLE .NE. 0) CALL ZSTRES(IHOLE1,NC,SMAX3)
ISN 0085 IF(IHOLE .EQ. 0) GO TO 501
ISN 0087 DO 500 I = 1,8
IF(ISI1AX3(I) .GT. ISI1AX2(I)) SMAX2(I) = ISI1AX3(I)
500 CONTINUE
501 CONTINUE

C FORM MASS * STIFFNESS PRODUCTS TO BE SENT TO W137 SOLVER
C -MUST CHANGE SIGN ON 21 AND 22 QUADRANTS OF SKPP FIRST
C SKPP = HOLZER STIFFNESS MATRICES FOR EACH SECTION

ISN 0092 998 FREQ = FREQ * FREQ
ISN 0093 IF(FREQ .LT. 0.) RETURN
ISN 0095 WRITE(6,11) RPM1,RPM3,FREQ
ISN 0096 11 FORMAT(/,'FREQUENCY SEARCH',5X,'RPM1=','E12.5,/,'
1 5X,'RPM2=','E12.5,/,21X,'STARTING FREQ.=',E12.5,/

ISN 0097 DO 300 ISEC = 1,NR1
ISN 0098 DO 301 I = 1,12
ISN 0099 DO 301 J = 1,12
ISN 0100 SKP(I,J) = SKPP(I,J,ISEC)
ISN 0101 IF(RPM3 .NE. RPM1) SKP(I,J) = SKPP2(I,J,ISEC)
ISN 0103 IF(I .GT. 6 ) SKP(I,J) = -SKP(I,J)
301 CONTINUE
C FOR MASS
MATRIX
A
STORE IT IN SK ARRAY 02447

C 02448
ISH 0106 DO 305 I = 1,12
ISH 0107 DO 305 J = 1,12
ISH 0108 305 SK(I,J) = 0.
ISH 0109 DO 306 I = 1,6
ISH 0110 SK(I,I) = 1.
ISH 0111 306 SK(I+6,I+6) = -1.
C
C NOW INSERT SMM INTO THE 21 QUADRANT OF THE MASS MATRIX SK
C ACTUALLY THE PRODUCT FRE**2 * SMM IS INSERTED
C
ISH 0112 DO 310 I = 1,6
ISH 0113 DO 310 J = 1,6
ISH 0114 310 SK(I+6,J) = SK(I+6,J) + FRE**2 * SMM(I,J,ISEC+1)
C
ISH 0115 CALL MATHPY(SK,SKP,DS1,12,12,12)
C CALL MATHPRN(DS1,12,12,'DS1')
C REORDER THE MASS STIFFNESS PRODUCT MATRIX TO AGREE WITH
C THE W137 ORDER
C
ISH 0116 CALL PP1(2)
ISH 0117 CALL MATHPY(DS1,PT,CH,12,12,12)
ISH 0116 CALL MATHPY(P,CH,DS,12,12,12)
C CALL MATHPRN(DS,12,12,'DSCP')
C FILL IN THE SMK MATRIX
C
ISH 0119 DO 320 I = 1,12
ISH 0120 DO 320 J = 1,12
ISH 0121 320 STAMAT(I,J,ISEC) = DS(I,J)
C
ISH 0122 300 CONTINUE
C
ISH 0123 RETURN
C
ISH 0124 END

*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(DBL4)
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NOHAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 133, PROGRAM SIZE = 77400, SUBPROGRAM NAME =STABEL
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPILATION *****

2904K BYTES OF CORE NOT USED
SUBROUTINE STATICINR(NC,ISKIPI)

THIS ROUTINE PERFORMS THE STATIC SOLUTION WITH THE HOLZER METHOD

IMPLICIT REAL*8(A-H,O-Z)

COMMON /STIFF/ SK(12,12),SKK(12,12,21),SM(6,6,21)

COMMON /STIFF/ CHGG(12,12,20),DSKK(12,12,20)

COMMON /HSTIFF/ SKP(12,12),SKPP(12,12,21)

COMMON /STATIC/ SH(12,12,20),F12(12,21)

COMMON /EC/ CI(6,6),RPI

COMMON /STRS/ RBE(12,12,25,21),FOUT(12,21),SL(25,21),SW(25,21)

DIMENSION SH11(6,6),SH12(6,6),SH21(6,6),SH22(6,6)

DIMENSION SH(12,12,20)

DIMENSION F(12,21)

DIMENSION SHH(12,12)

DIMENSION STRS1(6,6,25,21),F1(6,1),F12(12,1)

DIMENSION STRS2(6,1),F2(12,1)

DIMENSION SDISPL(12,1),SFORCE(12,1),BSDISPL(12,1)

DIMENSION RB(12,12)

FORM THE HOLZER STIFFNESS MATRIX STRING MULTIPLICATION

CALL SHULT(NR)

CALL LOAD2(NR,FF)

PARTITION THE SHH INTO ONE DUMMY MATRIX

DO 50 I = 1,12

DO 50 J = 1,12

50 SHI(J,I) = SH(I,J,NR-1)

CALL MATPNR(SHH,12,12,'SHH ')

CALL PARTN(SHH,12,12,SH,SH11,SH12,SH21,SH22,6,6)

SOLVE FOR THE FORCE VECTOR AT NODE 1

F1 = (SH21*W + SH22*1)* - F12(F7-12)

WITH -I- MEANING THE INVERSE

DO 100 I = 1,6

FF2(I,1) = F1(I+6,1)

CALL MATPNN(F2,6,1,'F2 ')

CALL MATPNY(SH21,C,D1,6,6,6)

CALL MADD1(D1,SH22,SH21,6,6,1)

CALL MATPNN(SH21,6,6,'SH21 ')

CALL MNDV(Sh21,6,6,RL,R1)

CALL MATPNN(SH21,6,6,'SH21.')

CALL MATPNY(SH22,FF2,6,6,6,1)

CALL MATPNN(F1,6,1,'F1 ')

SOLVE FOR THE DISPLACEMENT AT NODE NR

02484

02485

02486

02487

02488

02489

02490

02491

02492

02493

02494

02495

02496

02497

02498

02499

02500

02501

02502

02503

02504

02505

02506

02507

02508

02509

02510

02511

02512

02513

02514

02515

02516

02517

02518

02519

02520

02521

02522

02523

02524

02525

02526

02527

02528

02529

02530

02531

02532

02533

02534

02535

02536

02537

02538
DO 110 I = 1,6
F(I,1) = FF(I,1)

CALL MATPY(SH11,C,SH21,6,6,6)
CALL MATADD(SH21,SH12,SH11,6,6,1.)
CALL MATPY(SH11,FF2,6,6,1.)
CALL MATPRN(U,6,' U ')

THE VECTOR OF DISPL. AND FORCES AT EACH NODE WILL BE STORED IN + FOUT(12,21) - DISPL. IN 1-6 AND FORCES IN 7-12.

FIRST SOLVE FOR DEFORMATIONS AND FORCES AT NODE 2

THIS SECTION ALSO GENERATES THE DIFFERENTIAL STIFFNESS MATRIX AND THE CENTRIFUGAL RESTORING STIFFNESS MATRIX

DO 120 I = 1,6
FOUT(I,NR) = U(I,1)
FOUT(I+6,NR) = F(I+6,NR)
FOUT(I,1) = 0.
FOUT(I+6,1) = F(I,1)
FF(I,1) = 0.
FF(I+6,1) = F(I+6,1)

GET THE HOLZER STIFFNESS FOR SECTION 1

DO 130 I = 1,6
DO 130 J = 1,6
SKP(I,J) = SKPP(I,J,1)
SKP(I+6,J) = SKPP(I+6,J,1)
SKP(I,J+6) = SKPP(I,J+6,1)
SKP(I+6,J+6) = SKPP(I+6,J+6,1)

CALL MATPYCSKP(FF,FOUT1,12,12,1)
CALL MATPRN(FF,12,1, 'FF ')
CALL MATPRN(FOUT1,12,1,'FOUT1 ')

THE DIFFERENTIAL STIFFNESS FOR SECTION 1

IF(ISKP .LE. 1) CALL KDGG11(1,FF,FOUT1,DSSK)

BEGIN LOOP TO SOLVE FOR THE REMAINING NODES 3 - (NR-1)

NRM1 = NR - 1
IF(NRM1 .EQ. 1) GO TO 201
DO 200 I = 3,NRM1

CALL MATPY(SKP,FF,FOUT1,12,12,1)
CALL MATPRN(FF,12,1, 'FF ')
CALL MATPRN(FOUT1,12,1,'FOUT1 ')

THE EQUILIBRIUM EQ WILL BE USED

DO 210 J = 1,6
FF(J,1) = FOUT(J,I-1)
FF(J+6,1) = -FOUT(J+6,I-1) + F(J+6,I-1)

GET THE HOLZER STIFFNESS MATRIX FOR SECTION ( I-1 )
**VERSION 1.3.0 (01 MAY 80)**

**STATIC SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)**

**DATE 82.141/10.50.28 PAGE 3**

```
ISH 0058   DO 220 J = 1,6
ISH 0059   DO 220 K = 1,6
ISH 0060   SKP(J,K) = SKPP(J,K+6)
ISH 0061   SKP(J+6,K) = SKPP(J+6,K+6)
ISH 0062   SKP(J+6,K+6) = -SKPP(J+6,K+6)
ISH 0063   220 SKP(J+6,K+6) = -SKPP(J+6,K+6)

ISH 0064   CALL MATPYP(SKP,FF,FOUT1,12,12,1)
ISH 0065   CALL MATPRN(FF,12,1,'FF ')
ISH 0066   CALL MATPRTN(FOUT1,12,1,'FOUT1')

C THE DIFFERENTIAL STIFFNESS FOR SECTION , I-1

ISH 0067   ISEC = I-1
ISH 0068   IF(ISKIP.LE.1) CALL KDGG(ISEC,FF,FOUT1,DSKK)
ISH 0069   DO 225 J = 1,12
ISH 0070   225 FOUT(J,I) = FOUT1(J,I)
ISH 0071   CONTINUE
C THE DIFFERENTIAL STIFFNESS FOR SECTION NR-1

ISH 0076   ISEC = NR - 1
ISH 0077   DO 140 I = 1,6
ISH 0078   FOUT(I,1) = FOUT(I,NR)
ISH 0079   FOUT(I+6,1) = FOUT(I+6,NR)
ISH 0080   FF(I,1) = FOUT(I,ISEC)
ISH 0081   FF(I+6,1) = FOUT(I+6,ISEC) + F(I+6,ISEC)
ISH 0082   CALL MATPRN(FF,12,1,'FF ')
ISH 0083   CALL MATPRN(FOUT1,12,1,'FOUT1')
ISH 0084   IF(ISKIP.LE.1) CALL KDGG(ISEC,FF,FOUT1,DSKK)
ISH 0085   140 CONTINUE
C FIND THE LOCAL BEAM DEFLECTIONS AND FORCES FOR EACH SECTION

ISH 0089   WRITE(6,305) I
ISH 0090   305 FORMAT(/,5X,'DEFLECTIONS AND FORCES FOR SECTION',I5,/)  
ISH 0091   DEFINE SECTION ARRAYS FOR DISPL. AND FORCES

ISH 0096   DO 301 J = 1,6
ISH 0097   SDISPL(J,1) = FOUT(J,1)
ISH 0098   SDISPL(J+6,1) = FOUT(J+6,1)
ISH 0099   SFORCE(J,1) = FOUT(J+6,1)
ISH 0100   SFORCE(J+6,1) = FOUT(J+6+6,1)

C DEFFLECTIONS AND FORCES FOR EACH BEAM

ISH 0105   DO 400 J = 1,NC
ISH 0106   DO 401 K = 1,12
ISH 0107   DO 401 L = 1,12
```
C 0100 WRITE(6,402) (EDISPL(K,1),K=1,6) 02667
C 0101 WRITE(6,403) (EDISPL(K,1),K=7,12) 02668
C 0104 WRITE(6,405) 02669
C 0105 WRITE(14X,'(FORCES)') 02670
C 0106 WRITE(6,404) (EFORCE(K,1),K=1,6) 02671
C 0107 WRITE(6,403) (EFORCE(K,1),K=7,12) 02672
C 0108 400 IP = IP + 1 02673
C 0109 300 CONTINUE 02674
C 0110 WRITE(6,240) 02675
C 0111 240 FORMAT(/,5X,' THE STATIC SOLUTION',/5X,'DISPL. AND ROTATIONS',/ 02676
15X,'U',10X,'V',10X,'W',10X,'RX',10X,'RY',10X,'RZ',/) 02677
C 0112 DO 250 I = 1,HR 02678
C 0113 250 WRITE(6,260) (FOUT(I,J),J=1,6) 02679
C 0114 260 WRITE(6,261) 02680
C 0115 WRITE(6,270) 02681
C 0116 270 FORMAT(/,5X,'FORCES AND MOMENTS',/5X,'FX',10X,'FY',10X,'FZ', 02682
10X,'MX',10X,'MY',10X,'MZ',/) 02683
C 0117 DO 280 I = 1,HR 02684
C 0118 280 WRITE(6,260) (FOUT(I,J),J=7,12) 02685
C C
C 0119 RETURN 02686
C 0120 END 02687

*OPTIONS IN EFFECT*NAME=MAIN OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTO32(DBL4)
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NOHAP NOFORMAT GOSTMT NOXREF NOALC NOAPSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 119, PROGRAM SIZE = 7468, SUBPROGRAM NAME =STATIC
*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION *****

2988K BYTES OF CORE NOT USED
SUBROUTINE ZSTRES (ISEC,NC,SMAX)

THIS ROUTINE PERFORMS THE STRESS ANALYSIS FOR EACH COMPOSITE BEAM IN THE SECTION

IMPLICIT REAL*(4-9,0-Z)

COMMON /STRS/ RBE(12,12,25,21),FOUT(12,21),SL(25,21),SW(25,21)
COMMON /PLY1/ PLY(1,25,7),THETA(7),RHO(7)
COMMON /ZCQEF/ E1(7),E2(7),E3(7),G12(7),G13(7),G23(7),G13(7),
1 V1(7),V3(7),V2(7)

COMMON /REORDR/ P(12,12),PT(12,12)
COMMON /FAIL/ VS(42),TSAI(8,25)
COMMON /MODAL/ SIGMA(3,8)
DIMENSION BDISPL(12,11),SDISPL(12,1),STRN(6,1),STRN(6,1)
DIMENSION R(12,12),TRANS(6,6),STR(3,1),STR(3,1)
DIMENSION D3(3,1),D4(3,1),Q(3,3),SIGMA(3,1),SIGMA(3,1)
DIMENSION T3(3,3),SHAV(6)
DIMENSION TTR(3,3),TT(3,3),DB(3,3),QB(3,3)

TRANS(6,5) = -1.

DO 100 I = 1,NC
TRANSFER THE SECTION NODE DISPLACEMENTS IN THE FOUT ARRAY TO BEAM I

DO 110 J = 1,12
DO 110 K = 1,12
R(J,K) = RBE(J,K,I,ISEC)
CALL MATFNR(R,12,12,' R ')
DO 120 J = 1,6
SDISPL(J,1) = FOUT(J,ISEC)
110 DO 120 J = 6,12
SDISPL(J+6,1) = FOUT(J+6,ISEC+1)
CALL MATFNR(SDISPL,12,1,'SDISPL')

CALL MATMPY(R,SDISPL,BDISPL,12,12,1)
CALL MATFNR(BDISPL,12,1,'BDISPL')

CALL MATFNR(SDISPL,12,1,'BDISPL')
CALL MATFNR(BDISPL,12,1,'BDISPL')

SHAPE FUNCTION COEFFICIENT EVALUATION

BL = SL(I,ISEC)
U1 = BDISPL(1,1)
V1 = BDISPL(2,1)
W1 = BDISPL(3,1)
RX1 = BDISPL(4,1)
RY1 = BDISPL(5,1)
RZ1 = BDISPL(6,1)
U2 = BDISPL(7,1)
V2 = BDISPL(8,1)
W2 = BDISPL(9,1)
RX2 = BDISPL(10,1)
RY2 = BDISPL(11,1)
RZ2 = BDISPL(12,1)
C CALCULATE THE DEFLECTIONS AND SLOPES AT 10% OF BEAM LENGTH
C

X = 0.25 * SL(I,ISEC)

C

BDISPL(7,1) = A2 + B2 * X
BDISPL(11,1) = -2. * C1 * X - 3. * D1 * X**2 - B1
BDISPL(12,1) = 2. * C * X + 3. * D * X**2 + B

C

BEGIN LOOP FOR THE SEVEN LAYERS IN EACH BEAM

T = 0.
DO 51 K = 1,7
Y = T / 2.
Z = SW(I,ISEC) / 2.

C DETERMINE THE RADIAL DEFLECTIONS AT THE BEAM CORNERS

U1 = BDISPL(7,1) + T/2. * BDISPL(12,1) - Z * BDISPL(11,1)
U2 = BDISPL(7,1) - T/2. * BDISPL(12,1) - Z * BDISPL(11,1)
U3 = BDISPL(7,1) - T/2. * BDISPL(12,1) + Z * BDISPL(11,1)
U4 = BDISPL(7,1) + T/2. * BDISPL(12,1) + Z * BDISPL(11,1)

C

WRITE(6,5001) I,U1,U2,U3,U4
FORIAnSX, '** BEAt!, CORNER DEFLECS.
C

SX1 = U1 / X
SX2 = U2 / X
SX3 = U3 / X
SX4 = U4 / X

C

WRITE(6,5001) SL(I,ISEC),SX1,SX2,SX3,SX4
FORIAn5X, 'LENGTH,CORNER STRAINS',SE1Z.S)
C

C

C

SLOPE1 = ( SX2 - SX1 ) / T
SLOPE2 = ( SX3 - SX4 ) / T

C

WRITE(6,5002) SLOPE1,SLOPE2

C

SLOPE1 = SLOPE1 / T
SLOPE2 = SLOPE2 / T

C

WRITE(6,5002) SLOPE1,SLOPE2

C

C

SLOPE1 = SLOPE1 / T
SLOPE2 = SLOPE2 / T

C

WRITE(6,5002) SLOPE1,SLOPE2

C

C

DO 200 LAYER = 1,8

C

TEFF = T/2. + Y

C

WRITE(6,5002) T/2. + Y
SIDE1 = SLOPE1 * TEFF + SXI
SIDE2 = SLOPE2 * TEFF + SX4

WRITE(6,503) I,LAYER,T,Y,TEFF,SIDE1,SIDE2
C503 FORMAT(5X,'BEAM,LAYER,T,Y,TEFF,51,S2',/5X,2I5,5X,5E12.5)

YOLD = Y

IF(LAYER .GT. 7) GO TO 201

II = LAYER
IF(II .GE. 5) II = II - 1

DETERMINE THE LAMINA GEOMETRIC STRAIN COMPONENTS, EY AND EXY
TRANSFORMATION MATRIX IS TT
TTI = TT INVERSE
TTR = TT TRANSPOSE

CC = DCOS(THETA(II))
SS = DSIN(THETA(II))
C2 = CC * CC
S2 = SS * SS
CS = CC * SS

TT(1,1) = CC
TT(1,2) = CS
TT(1,3) = 2. * CS
TT(2,1) = S2
TT(2,2) = C2
TT(2,3) = -2. * CS
TT(3,1) = -CS
TT(3,2) = C3
TT(3,3) = CC - S2

DO 155 KK = 1,3
DO 155 LL = 1,3
TTR(KK,LL) = TT(KK,LL)
CONTINUE

TII(1,3) = -TTI(1,3)
TTI(2,3) = -TT(2,3)
TTI(3,1) = TTI(3,1)
TTI(3,2) = TTI(3,2)

DO 155 KK = 1,3
DO 155 LL = 1,3
TTR(KK,LL) = TTI(KK,LL)

CALL MATPRN(TT,3,3,'TT')
CALL MATPRN(TTI,3,3,'TTI')
CALL MATPRN(TTR,3,3,'TTR')

IF(E11(II).EQ.0 .OR. PLY(ISEC,I,II).EQ.0.) GO TO 300

V21 = V12(II) * E22(II) / E11(II)
FACTOR = 1. - V12(II) * V21
Q(I,1) = E11(II) / FACTOR
Q(I,2) = V21 * Q(I,1)
ISN 0106  Q(1,3) = 0.
ISN 0107  Q(2,1) = Q(1,2)
ISN 0108  Q(2,2) = E22(II) / FACTOR
ISN 0109  Q(2,3) = 0.
ISN 0110  Q(3,1) = 0.
ISN 0111  Q(3,2) = 0.
ISN 0112  Q(3,3) = G12(II)

C ROTATED LAMINA STIFFNESS MATRIX, QB(I,J)
C
ISN 0113  CALL MATMPY(Q,TTR,QB1,3,3,3)
ISN 0114  CALL MATMPY(TTI,QB1,QB,3,3,3)
C
ISN 0115  QF1 = QB(1,2) - QB(2,3) * QB(1,3) / QB(3,3)
ISN 0116  QF2 = QB(2,2) - QB(2,3) * QB(3,3) / QB(3,3)
C
ISN 0117  STR1(1,1) = SIDE1
ISN 0118  STR1(2,1) = -QF1 / QF2 * SIDE1
ISN 0119  STR1(3,1) = (QB(2,3)/QB(3,3))*QF1/QF2-QB(1,3)/QB(3,3)) * SIDE1
C
ISN 0120  STR2(1,1) = SIDE2
ISN 0121  STR2(2,1) = -QF1 / QF2 * SIDE2
ISN 0122  STR2(3,1) = (QB(2,3)/QB(3,3))*QF1/QF2-QB(1,3)/QB(3,3)) * SIDE2
C
C HOLLOW LAYER - ZERO Q(I,J)
C
ISN 0123  GO TO 301
ISN 0124  300 CONTINUE
ISN 0125  DO 310 KK = 1,3
ISN 0126  STR1(KK,1) = 0.
ISN 0127  STR2(KK,1) = 0.
ISN 0128  DO 310 LL = 1,3
ISN 0129  QB(KK,LL) = 0.
ISN 0130  DO 310 KK = 1,3
ISN 0131  310 CONTINUE
C
C CALL MATPRN(Q,3,3,'Q')
C CALL MATPRN(QB,3,3,'QB')
C CALL MATPRN(STR1,3,1,'STR1')
C CALL MATPRN(STR2,3,1,'STR2')
C
C CALL MATMPY(TTR,STR1,03,3,3,3,1)
C CALL MATMPY(TTR,STR2,04,3,3,3,1)
C
C CALCULATE THE LAMINA STRESSES
C
C CALL MATPRN(Q,3,3,'QIJ')
C CALL MATMPY(Q,D3,SIGMA1,3,3,1)
C CALL MATMPY(Q,D4,SIGMA2,3,3,1)
C
C AVERAGE THE SIDE STRESSES - STORE IN SIGMA
C
 ISSUE 0136 DO 210 J = 1,3
 ISSUE 0137 210 SIGMA(J,LAYER,I) = (SIGMA(J,1) + SIGMA(J,1)) / 2.
 C
 ISSUE 0138 WRITE(6,212) (SIGMA(JJ,LAYER,I),JJ=1,3)
 ISSUE 0139 212 FORMAT(5X,3E12.5)
 C
 ISSUE 0140 WRITE(6,213)
 ISSUE 0141 213 (SIGMA(JJ,LAYER,I),JJ=1,3)
 C
 ISSUE 0142 CONTINUE
 ISSUE 0143 CONTINUE
 C
 ISSUE 0144 DO 220 I = 1,8
 ISSUE 0145 220 SMAX(I) = 0.
 ISSUE 0146 DO 220 J = 1,NC
 ISSUE 0147 220 IF(SMAX(I) .LT. TSAI(J)) SMAX(I) = TSAI(J)
 C
 ISSUE 0148 CALL MATPRN(SMAX,8,1,'SMAX')
 ISSUE 0149 RETURN
 ISSUE 0150 END

*OPTIONS IN EFFECT* NAME(MAX) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(DBL4)
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NOEXP NOFORMAT GOSTAT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 146, PROGRAM SIZE = 5184, SUBPROGRAM NAME = ZSTRES
*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION *****

2984K BYTES OF CORE NOT USED
SUBROUTINE THICKCII,NR,NC,X,Y,Z,THK,ISKIP)

C DETERMINE THE LAYER THICKNESS, TH, ASSUMING THICKNESS SYMMETRY ABOUT THE CENTER AXIS

IMPLICIT REAL*8(A-H,O-Z)

COMMON IPLY1/ PPLY(21,25,7),THETAC7),RHOC7)
COMMON ILAYERI THC 7,25 )'BMASS((25),BSPAN(25),BWIDTH(25)
COMMON /NIEG1H(1)
COMMON DUM6(6),NCO ,NCK,DUM14(4)
COMMON /BILT 81,SBIC 21)' ITTLEC 18),DUM111C 277)
DISTRIBUTION PLYC 7)
IP = (II-1) * NC
AVERAGED THICKNESS CALCULATION
DO 100 I = 1,NC
IP = IP + 1
T(IP) = (THKCIP) + THKCIP+NC)) / 2.
AVERAGE THE X VALUES FOR BOTH ENDS TO FIND LAYER THICKNESS
TOTAL = THKCIP)
DO 101 J = 1,7
101 THCJ,I) = PPLY(I,I,J)
DETERMINE THE BEAM LENGTH
XX = X(IP) - X(IP+1)
YY = Y(IP) - Y(IP+1)
ZZ = Z(IP) - Z(IP+1)
BSPAN(I) =DSGRT(XX**2 + YY**2 + ZZ**2)
AVERAGED WIDTH
XX = X(IP) - X(IP+1)
YY = Y(IP) - Y(IP+1)
ZZ = Z(IP) - Z(IP+1)
DIST(I) =DSGRT(XX**2 + YY**2 + ZZ**2)
DIST(I) = (DIST(I) + DSGRT(XX**2 + YY**2 + ZZ**2)) / 2.
CONTINUE
BEAM WIDTHS
NCH1 = NC - 1
IP(NCH1 , .EQ. 0) GO TO 201
BWIDTH(I) = DIST(I)
ISt! 0035 BWIDTH(NC) = DIST(NC-1) 03005
ISt! 0036 DO 200 I = 2, NC(M1) 03006
ISt! 0037 200 BWIDTH(I) = (DIST(I-1) + DIST(I))/2. 03007
ISt! 0038 201 CONTINUE 03008
C C BEAM MASS 03009
C
ISt! 0039 DO 300 I = 1, NC 03010
ISt! 0040 BMASS(I) = 0. 03011
ISt! 0041 AREA = BWIDTH(I) * BSPAN(I) 03013
ISt! 0042 DO 11 J = 1, 7 03014
ISt! 0043 11 BMASS(I) = BMASS(I) + AREA * TH(J, I) * RHO(J) 03015
ISt! 0044 300 CONTINUE 03016

C C LAYER WEIGHT CALCULATION, RESULTS STORED IN WL(7) 03017
C
ISt! 0045 IF(ISKIP .GT. 1) RETURN 03019
ISt! 0047 IF(II .GT. 1) GO TO 400 03020
ISt! 0049 DO 401 I = 1, 7 03021
ISt! 0050 401 WL(I) = 0. 03022
ISt! 0051 IF(NCD .EQ. 2) GO TO 400 03023
ISt! 0052 400 CONTINUE 03024
ISt! 0053 IF(NCD .EQ. 2) GO TO 400 03025
ISt! 0054 WL(I) = WL(I) + RHO(I) * BSPAN(I) * TH(J, I) * TH(I, J) 03026
ISt! 0055 WL(7) = WL(7) 03027
ISt! 0056 RETURN 03028
ISt! 0057 400 CONTINUE 03029
ISt! 0058 DO 410 J = 1, NC 03030
ISt! 0059 410 WL(7) = WL(7) + RHO(I) * BSPAN(I) * TH(I, J) 03031
ISt! 0060 IF(I .EQ. NRM1) CALL MATPRN(WL, 7, I, 'WL') 03032
ISt! 0061 NRM1 = NRM1 - 1 03033
ISt! 0062 IF(I .EQ. NRM1) CALL MATPRN(WL, 7, I, 'WL') 03034
ISt! 0063 C C END OF COMPIILATION 03035
ISt! 0064 RETURN 03036
ISt! 0065 END 03037

*OPTIONS IN EFFECT NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(DBL4)
*OPTIONS IN EFFECT SOURCE EBCDIC NOLIST NODECK OBJECT NOHAP NOFORMAT GOSTH NOXREF NOALC NODANSF TERM IBM FLAG(I)
*STATISTICS SOURCE STATEMENTS = 64, PROGRAM SIZE = 9726, SUBPROGRAM NAME = THICK
*STATISTICS NO DIAGNOSTICS GENERATED

3004K BYTES OF CORE NOT USED
SUBROUTINE TSAIWU (SIGMA, NC)

THE TSAI-WU FAILURE CRITERION IS EVALUATED

COMMON /FAIL/ X1T(7), X1C(7), X2T(7), X2C(7), S6P(7), S6M(7), TSAI(0,25)

DIMENSION SIGMA(3,6,25), F1(I), F2(I), F6(I), F11(I), F22(I), F66(I)

C DETERMINE THE COEFFICIENTS

L = 0
DO 100 I = 1, 8
L = L + 1
IF(I .EQ. 5) L = L - 1

C BEGIN LOOP FOR EACH LAYER OF EACH BEAM
DO 110 J = 1, NC

110 TSAI(I,J) = F1(I) * SIGMA(1,1,J) + F2(I) * SIGMA(2,1,J) +
1 F6(I) * SIGMA(3,1,J) * SIGMA(2,1,J) + F11(I)
2 F22(I) * SIGMA(2,2,J) * SIGMA(2,2,J) + F66(I)
3 + F12(I) * SIGMA(1,1,J) + SIGMA(2,1,J)
C CALL MATPRN(TSAI,6,25,'TSAI')
C CALL MATPRN(F1,6,1,'F1')
C CALL MATPRN(F2,8,1,'F2')
C CALL MATPRN(F6,6,1,'F6')
C CALL MATPRN(F11,6,1,'F11')
C CALL MATPRN(F22,8,1,'F22')
C CALL MATPRN(F66,6,1,'F66')
C CALL MATPRN(F12,8,1,'F12')

BEGIN LOOP FOR EACH LAYER OF EACH BEAM
DO 100 J = 1, NC

100 F12(I) = -DSQRT( F11(I) * F22(I) )
C CALL MATPRN(TSAI,8,2.5,'TSAI')
C CALL MATPRN(F1,8,1,'F1')
C CALL MATPRN(F2,8,1,'F2')
C CALL MATPRN(F6,8,1,'F6')
C CALL MATPRN(F11,8,1,'F11')
C CALL MATPRN(F22,8,1,'F22')
C CALL MATPRN(F66,8,1,'F66')
C CALL MATPRN(F12,8,1,'F12')

BEGIN LOOP FOR EACH LAYER OF EACH BEAM
DO 100 J = 1, NC

RETURN

END

*STARTS* NO STATISTICS GENERATED

*STARTS* NO STATISTICS GENERATED

*STARTS* NO STATISTICS GENERATED

*STARTS* NO STATISTICS GENERATED

END OF COMPILATION

3016K BYTES OF CORE NOT USED
ISH 0002 SUBROUTINE WARP(ISEC, NR, NC)

THIS ROUTINE CALCULATES THE WARPING TERMS REQUIRED FOR THE
WARPING FUNCTION OF A THIN SECTION - THE VALUES WILL BE USED
IN SUBROUTINE RBE - (SEE THE THEORY OF ELASTIC STABILITY,
TIMOSHENKO AND GERE)

IMPLICIT REAL*8(A-H,O-Z)

COMMON /SC/XSC(21), YSC(21), ZSC(21), ALPHA(21), SECP(21), SECA(21)
COMMON /INPUT/ X(1000), Y(1000), Z(1000), T(1000)
COMMON /LAYER/ TH(7, 25), BMASS(25), BSPAN(25), BWIDTH(25)
COMMON /HARP/ HS(26, 21), RTH(25, 21), RTT(25, 21)
COMMON /COORD1/ CLOCAL(3, 3, 1000)
COMMON /COORD2/ CLOCAL(3, 3, 1000)
COMMON /RT/ RT(26), THETA(26), SS(25), HSS(25), ROUT(26)
DIMENSION RT(26), THETA(26), SS(25), HSS(25), ROUT(26)

C ARC LENGTH VARIABLE, S, AND SS.
C SS = CUMULATIVE ARC LENGTH FOR ELEMENT INTERVALS

SS(I) = BWIDTH(I)
DO 60 I = 2, NC
SS(I) = SS(I-1) + BWIDTH(I)
CALL MATPRH( SS, 25, 1, ' SS ')

Determine the value, HS, beginning at the leading edge
- Must also calculate the perpendicular distances from the
tangent, RT and from the normal, RN to the shear center
WS = evaluated at node points
WSS = evaluated at interval ends

NODE = (ISEC-1) * NC + 1
DO 100 I = 1, NC
RX = XSC( ISEC ) - X(NODE)
RY = YSC( ISEC ) - Y(NODE)
RZ = ZSC( ISEC ) - Z(NODE)
RHAG = DSQRT( RX**2 + RY**2 + RZ**2 )
IF(RHAG .EQ. 0.) GO TO 50
RX = RX / RHAG
RY = RY / RHAG
RZ = RZ / RHAG

ANGLE BETWEEN VECTORS R AND LOCAL Z, THETA

ARG = RX * CLOCAL(3, 1, NODE) + RY * CLOCAL(3, 2, NODE) +
      RZ * CLOCAL(3, 3, NODE)
THETA(I) = DARCOS( ARG )

Determine the direction of shear, CH or CCH, for the sign on
WSS - NOTE: THIS CHECK USES THE COORDINATE POINTS NOT THE
SIDES ON THE BEAM, SO THE CHECK IS APPROXIMATE

IF(I .EQ. NC) SIGN(I) = SIGN(I-1)
IF(I .EQ. NC) GO TO 400
**VERSION 1.3.0 (01 MAY 80) WORK SYSTEM/370 FORTRAN I EXTENDED (ENHANCED) DATE 02.141/10.51.01 PAGE 2**

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C V1 CROSS V2 - THE RESULTANT X COMPONENT SETS SIGN OF, SIGN.</td>
</tr>
<tr>
<td>2</td>
<td>XCOMP GT. 0. - SIGN = 1</td>
</tr>
<tr>
<td>3</td>
<td>XCOMP LT. 0. - SIGN = -1</td>
</tr>
</tbody>
</table>

**FORTRAN EXTENDED**

**INTEGRAL**

```
SETI 0032
ISH 0033
V2Y = Y(NODE+1) - YSEC(ISEC)
ISH 0034
V2Z = Z(NODE+1) - ZSEC(ISEC)
C
V1 CROSS V2 - 
```

**RESULTANT**

```
XCOMP GT. 0.
- SIGN = 1
XCOMP LT. 0.
- SIGN = -1

XCOMP = V1Y * V2Z - V1Z * V2Y
```

**SIGN I**

```
IF(XCOMP .LT. 0.) SIGNI = -1.
WRITE(6,111) I,NC,SIGNI)
```

**CONTINUE**

```
C
RN(I,ISEC) = -RMAG * DCOS(THETA(I))
RT(I) = RMAG * DSIN(THETA(I))
RTT(I,ISEC) = RT(I) * SIGNI)
ROUT(I) = RN(I,ISEC)
C
IF( I .EQ. 1 ) GO TO 101
WSS(I) = WSS(I-1) + SIGN(I) * RT(I) * (SS(I) - SS(I-1))
GO TO 100
```

**CONTINUE**

```
C
WSS(I) = SIGN(I) * RT(I) * SS(I)
WRITE(6,112) I,SIGN(I),WSS(I)
```

**FORMAT**

```
FORMAT(5X,'I,SIGN(I),WSS(I),',5X,'E12.5)
```

**WRITE**

```
WRITE(6,52) (WSSS(I),II=1,NC)
```

**FORMAT**

```
FORMAT(5X,E12.5)
```

**CALL**

```
CALL MATPRN(ROUT,26,1,' RN ')
CALL MATPRN(RT,26,1,' RT ')
CALL MATPRN(THETA,26,1,'THET')
CALL MATPRN(SIGN,25,1,'SIGN')
```

**CALCULATE THE AREA**

```
AREA = 0.
NODE = ISEC-1) * NC + 1
DO 110 I = 1,NC
AREA = AREA + T(NODE) * BWIDTH(I)
```

**CONTINUE**

```
C
WRITE(6,53) AREA
```

**FORMAT**

```
FORMAT(5X,'AREA=',E12.5)
```

**AVERAGE WARPING**, **WSBAR**

```
HSBAR = 0.
NODE = (ISEC-1) * NC + 1
DO 200 I = 1,NC
WSBAR = HSBAR + T(NODE) * (WSSS(I) + WSSS(I-1)) / 2. * SSS
```

**ADD THE FIRST TERM AND DIVIDE BY THE TOTAL AREA**

```
NODE = (ISEC-1) * NC + 1
```
ISN 0063  
WSBAR = WSBAR + T(NODE) * WSS(1) / 2. * SS(1)

ISN 0064  
C54  
WRITE(*,54) WSBAR

ISN 0065  
C  
FINALLY CALCULATE THE WARPING VALUE FOR RADIAL DISPLACEMENT, WS
C  
WS = WSBAR - WS, NOTE; THIS IS NOT COMPLETE BECAUSE NO THIST
C  
GRADIENT HAS BEEN USED

ISN 0066  
WS(1,ISEC) = WSBAR - WSS(1) / 2.

ISN 0067  
DO 300 I = 2, NC

ISN 0068  
RETURN

ISN 0069  
END

*OPTIONS IN EFFECT* NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(DBL4)

*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NOLAP NOFORMAT GOSTMT NOLANCL NOLANSF TERM IBM FLAG(I)

*STATISTICS* SOURCE STATEMENTS = 68, PROGRAM SIZE = 2582, SUBPROGRAM NAME = WARP

*STATISTICS* NO DIAGNOSTICS GENERATED

****** END OF COMPIILATION ******

3004K BYTES OF CORE NOT USED
SUBROUTINE HOLLOW(DLE,DTE,DROOT,DTIP,TI,TBT,NC,NR)

C REQUESTED OPTIONS: AUTODBL, DBL4, SOURCE
C OPTIONS IN EFFECT: NAME(MAIN) OPTIMIZE(3) LENCOUNT(60) SIZE(MAX) AUTODBL(DBL4)
C
C VERSION 1.3.0 101 MAY 80) SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 62.141/10.51.05 PAGE 1

C DATA SET U498D10L0 AT LEVEL 003 AS OF 12/10/81
C DATA SET U498D10L04 AT LEVEL 001 AS OF 10/30/81

ISN 0002 00001

SUBROUTINE HOLLOW(DLE,DTE,DROOT,DTIP,TI,TBT,NC,NR)

C THIS ROUTINE CALCULATES THE LAYER THICKNESSES FOR EACH BEAM
C ALSO FOUND ARE THE BEAM LENGTH, BSPAN, AND THE WIDTH, BWIDTH

CALL COMMON /LAYER/ TH(7,25),BMASS(25),BSPAN(25),BWIDTH(25)
CALL COMMON /INPUT/ X(1000),Y(1000),Z(1000),T(1000)
CALL COMMON /PLY/ PLY(21,25,7),THETA(7),RHO(7)

C DIMENSION DIST(25)
C A = CHORDWISE DISTANCE OF HOLLOW SECTION PENETRATION OF A BEAM
C B = SAME AS A BUT IN THE SPAN DISRECTION
C RBOT = RADIAL DISTANCE TO BOTTOM OF CAVITY
C RTIP = RADIAL DISTANCE TO TOP OF CAVITY
C
C CHECK FOR THE CAVITY EXISTENCE

IHOLE = 0
CAVITY = DLE + DTE + DROOT + DTIP
IF(CAVITY .EQ. 0.) GO TO 150
GO TO 151

GO TO 150
DO 150 NR1 = NR - 1
DO 150 I = 1, NR1
DO 150 J = 1, NC

C IF (I-1) * NC + 1
TAVE = (T(INODE) + T(INODE+NC)) / 2.
PLY(I,J,1) = TAVE / 2.
PLY(I,J,2) = 0.
PLY(I,J,3) = 0.
PLY(I,J,4) = 0.
PLY(I,J,5) = 0.
PLY(I,J,6) = 0.
PLY(I,J,7) = TAVE / 2.

DO 150 NODE1 = NODE
C CHORD = 0.

RETURN
CONTINUE

RTIP = X(NC/2) + DROOT
RBT = X(NC*NR-NC/2) - DROOT
WRITE(6,130) RBOT,RTIP
C130 FORMAT(5X,'RBOT,RTIP',2E12.5)

DO 160 NR1 = NR - 1
DO 160 I = 1, NR1
DO 160 J = 1, NC

C CHECK FOR THE CAVITY EXISTENCE

IHOLE = 0
CAVITY = DLE + DTE + DROOT + DTIP
IF(CAVITY .EQ. 0.) GO TO 150
GO TO 151

GO TO 150
DO 150 NR1 = NR - 1
DO 150 I = 1, NR1
DO 150 J = 1, NC

C IF (I-1) * NC + 1
TAVE = (T(INODE) + T(INODE+NC)) / 2.
PLY(I,J,1) = TAVE / 2.
PLY(I,J,2) = 0.
PLY(I,J,3) = 0.
PLY(I,J,4) = 0.
PLY(I,J,5) = 0.
PLY(I,J,6) = 0.
PLY(I,J,7) = TAVE / 2.

DO 150 NODE1 = NODE
C CHORD = 0.

RETURN
CONTINUE

RTIP = X(NC/2) + DROOT
RBT = X(NC*NR-NC/2) - DROOT
WRITE(6,130) RBOT,RTIP
C130 FORMAT(5X,'RBOT,RTIP',2E12.5)

DO 160 NR1 = NR - 1
DO 160 I = 1, NR1
DO 160 J = 1, NC

C CHECK FOR THE CAVITY EXISTENCE

IHOLE = 0
CAVITY = DLE + DTE + DROOT + DTIP
IF(CAVITY .EQ. 0.) GO TO 150
GO TO 151

GO TO 150
DO 150 NR1 = NR - 1
DO 150 I = 1, NR1
DO 150 J = 1, NC

C IF (I-1) * NC + 1
TAVE = (T(INODE) + T(INODE+NC)) / 2.
PLY(I,J,1) = TAVE / 2.
PLY(I,J,2) = 0.
PLY(I,J,3) = 0.
PLY(I,J,4) = 0.
PLY(I,J,5) = 0.
PLY(I,J,6) = 0.
PLY(I,J,7) = TAVE / 2.

DO 150 NODE1 = NODE
C CHORD = 0.

RETURN
CONTINUE

RTIP = X(NC/2) + DROOT
RBT = X(NC*NR-NC/2) - DROOT
WRITE(6,130) RBOT,RTIP
C130 FORMAT(5X,'RBOT,RTIP',2E12.5)

DO 160 NR1 = NR - 1
DO 160 I = 1, NR1
DO 160 J = 1, NC

C CHECK FOR THE CAVITY EXISTENCE

IHOLE = 0
CAVITY = DLE + DTE + DROOT + DTIP
IF(CAVITY .EQ. 0.) GO TO 150
GO TO 151

GO TO 150
DO 150 NR1 = NR - 1
DO 150 I = 1, NR1
DO 160 J = 1, NC

C IF (I-1) * NC + 1
TAVE = (T(INODE) + T(INODE+NC)) / 2.
PLY(I,J,1) = TAVE / 2.
PLY(I,J,2) = 0.
PLY(I,J,3) = 0.
PLY(I,J,4) = 0.
PLY(I,J,5) = 0.
PLY(I,J,6) = 0.
PLY(I,J,7) = TAVE / 2.

DO 150 NODE1 = NODE
C CHORD = 0.

RETURN
CONTINUE

RTIP = X(NC/2) + DROOT
RBT = X(NC*NR-NC/2) - DROOT
WRITE(6,130) RBOT,RTIP
C130 FORMAT(5X,'RBOT,RTIP',2E12.5)
C BEAM LENGTH, BSPAN

C

ISN 0037  XX = X(NODE) - X(NODE+NC)
ISN 0038  YY = Y(NODE) - Y(NODE+1)
ISN 0039  ZZ = Z(NODE) - Z(NODE+NC)
ISN 0040  BSPAN(J) = DSQRT(XX**2 + YY**2 + ZZ**2)
C CALL MATPRN(BSPAN,25,1,'SPAN')
C
C AVERAGED WIDTH, BWIDTH
C
ISN 0041  XX = X(NODE) - X(NODE+1)
ISN 0042  YY = Y(NODE) - Y(NODE+1)
ISN 0043  ZZ = Z(NODE) - Z(NODE+1)
ISN 0044  DIST(J) = DSQRT(XX**2 + YY**2 + ZZ**2)
ISN 0045  XX = X(NODE+NC) - X(NODE+NC+1)
ISN 0046  YY = Y(NODE+NC) - Y(NODE+NC+1)
ISN 0047  ZZ = Z(NODE+NC) - Z(NODE+NC+1)
ISN 0048  DIST(J) = (DIST(J) + DSQRT(XX**2 + YY**2 + ZZ**2)) / 2.
C
C
ISN 0049  110 NODE = NODE + 1
ISN 0050  NODE = NODE + 1
C
ISN 0051  NCH1 = NC - 1
ISN 0052  IF(NCH1 .LE. 0) GO TO 201
ISN 0053  BWIDTH(N1) = DIST(1)
ISN 0054  BWIDTH(NC) = DIST(NC-1)
ISN 0055  DO 111 JJ = 2,NCH1
ISN 0056  
ISN 0057  111 BWIDTH(JJ) = (DIST(JJ-1) + DIST(JJ)) / 2.
I CALL MATPRN(BWIDTH,25,1,'BWIDTH')
C
ISN 0058  201 CONTINUE
ISN 0059  DO 112 JJ = 1,NC
ISN 0060  112 CHORD = CHORD + BWIDTH(JJ)
C WRITE(6,131) I,CHORD
C 131 FORMAT(5X,'STATION, CHORD',I5,5X,E12.5)
C
ISN 0061  DO 100 J = 1,NC
ISN 0062  NODE = NODE + 1
C
ISN 0063  IF(X(NODE+NC) .LE. RBOT) GO TO 10
ISN 0064  IF(X(NODE+NC) .GT. RBOT .AND. X(NODE+NC) .LE. RTIP) GO TO 20
ISN 0065  IF(X(NODE+NC) .GT. RTIP) GO TO 30
C
C THIS RADIAL SECTION IS SOLID
C
ISN 0066  10 B = 0.
ISN 0070  GO TO 50
C
C THIS RADIAL SECTION IS HOLLOW / SOLID
C
ISN 0071  20 B = BSPAN(J)
ISN 0072  IF(X(NODE) .LT. RBOT) B = X(NODE+NC) - RBOT
ISN 0074  IF(IHOLE .EQ. 0) IHOLE = I
ISN 0076  GO TO 50
C
C THIS RADIAL SECTION IS NEAR THE TIP, MAY INCLUDE SOME HOLLOW
C
ISN 0077  30 B = 0.
ISN 0078  IF(X(NODE) .LT. RTIP) B = RTIP - X(NODE)
-BEGIN CHORDWISE SEARCH FOR - A - DIMENSION

\[
\text{IF}(B \cdot \text{GT.} \cdot 0.) \text{ GO TO 60}
\]

\[
\text{GO TO 61}
\]

\[
\text{THIS IS A HOLLOW BEAM}
\]

\[
\text{CDIST} = 0.\]

\[
\text{DO 120 JJ = 1,J}
\]

\[
\text{CDIST} = \text{CDIST} + \text{BWIDTH(JJ)}
\]

\[
\text{CDIST} = \text{DISTANCE FROM LE TO BEAM (J)}
\]

\[
\text{CTE} = \text{DISTANCE FROM LE TO CHORDWISE END OF CAVITY}
\]

\[
\text{CTE} = \text{CHORD - DTE}
\]

\[
\text{IF}(\text{CDIST} \cdot \text{LE. DLE}) \text{ GO TO 61}
\]

\[
\text{IF}(\text{CDIST} \cdot \text{GT. DLE} \cdot \text{AND. CDIST} \cdot \text{LE. CTE}) \text{ GO TO 62}
\]

\[
\text{CDIST1} = \text{CDIST} - \text{BWIDTH(JJ)}
\]

\[
\text{IF}(\text{CDIST1} \cdot \text{LT. CTE} \cdot \text{AND. CDIST} \cdot \text{GT. CTE}) \text{ GO TO 63}
\]

\[
\text{IF}(\text{CDIST1} \cdot \text{GT. CTE}) \text{ GO TO 64}
\]

\[
\text{SOLID LE SECTION}
\]

\[
\text{A} = 0.\]

\[
\text{GO TO 65}
\]

\[
\text{HOLLOW \ / SOLID SECTION - LE}
\]

\[
\text{A} = \text{CDIST} - \text{DLE}
\]

\[
\text{IF}(\text{A} \cdot \text{GT. BWIDTH(J)}) \text{ A} = \text{BWIDTH(J)}
\]

\[
\text{GO TO 65}
\]

\[
\text{HOLLOW \ / SOLID SECTION - TE}
\]

\[
\text{A} = \text{CTE} - \text{CDIST1}
\]

\[
\text{GO TO 65}
\]

\[
\text{SOLID TE SECTION}
\]

\[
\text{A} = 0.\]

\[
\text{CONTINUE}
\]

\[
\text{DEFINE THICKNESS OF EACH LAYER IN BEAM (J)}
\]

\[
\text{IF}(\text{B} \cdot \text{EQ.} \cdot 0. \cdot \text{OR.} \cdot \text{A} \cdot \text{EQ.} \cdot 0.) \text{ GO TO 80}
\]

\[
\text{TAVE} = (\text{TINODE} \cdot \text{+ TINODE+HC}) / 2.
\]

\[
\text{TCHECK} = \text{TAVE} - 2. \cdot \text{TTI} - 2. \cdot \text{TBT}
\]

\[
\text{IF}(\text{TCHECK}) 90,90,95
\]

\[
\text{NO HOLLOW SECTION DUE TO MINIMUM THICKNESS}
\]

\[
\text{TCHK1} = \text{TAVE} - 2. \cdot \text{TTI}
\]

\[
\text{IF}(\text{TCHK1}) 91,91,92
\]

\[
\text{TTII} = \text{TAVE} / 2.
\]

\[
\text{TBTI} = 0.\]
ISH 0116  THOL = 0.
ISH 0117  GO TO 200
ISH 0118  92  TBT1 = TCHECK / 2.
ISH 0119  TTI1 = TTI
ISH 0120  THOL = 0.
ISH 0121  GO TO 200
C
C  HOLLOW SECTION THICKNESS IS NONZERO
C
ISH 0122  95  THOL = A * B / BWIDTH(J) / BSPAN(J) * TCHECK
ISH 0123  TBT1 = A * B / BWIDTH(J) / BSPAN(J) * TBT
ISH 0124  TTI1 = TAVE - THOL - 2. * TBT1
ISH 0125  TTI1 = TTI1 / 2.
ISH 0126  GO TO 200
C
C  SOLID SECTION - ALL TITANIUM
C
ISH 0127  80  TTI1 = (T(NODE) + T(NODE+NC)) / 4.
ISH 0128  TBT1 = 0.
ISH 0129  THOL = 0.
C
ISH 0130  200  CONTINUE
C  WRITE(6,132) I,J,A,B
C132  FORMAT(5X,'STATION,BEAM,A,B',2I5,2E12.5)
ISH 0131  PLY(I,J,1) = TTI1
ISH 0132  PLY(I,J,2) = TBT1
ISH 0133  PLY(I,J,3) = 0.
ISH 0134  PLY(I,J,4) = THOL
ISH 0135  PLY(I,J,5) = 0.
ISH 0136  PLY(I,J,6) = TBT1
ISH 0137  PLY(I,J,7) = TTI1
C  WRITE(6,300) I,J,(PLY(I,J,K),K=1,7)
C300  FORMAT(5X,'STATION,BEAM,PLY',2I5,/,5X,7E12.5)
C
ISH 0138  100  CONTINUE
C
ISH 0139  RETURN
ISH 0140  END

*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTO DBL(DBL4)
*OPTIONS IN EFFECT*SOURCE EBCDIC HOLEST NODECK OBJECT NOHAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 139, PROGRAM SIZE = 2480, SUBPROGRAM NAME = HOLLOW
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPILATION *****

2992K BYTES OF CORE NOT USED
SUBROUTINE BMFIT2 (X, Y, YPI, YPN, N, IP, A, B, C, D)

IF (N - 2) 10, 30, 40

10  WRITE (6, 20) H

20 FORMAT (6H THE NUMBER OF DATA POINTS IS LESS THAN 2. N =, I4)

GO TO 240

30 0(1) = 0.0

C(1) = 0.0

40  6(1) = (Y(2) - Y(1)) / (X(2) - X(1))

A(1) = Y(1)

GO TO 240

C(1) = X(2) - X(1)

T = H - 1

H = H - 1

DO 50 I = 2, HI

C 00001

C 00002

C 00003

C 00004

C 00005

C 00006

C 00007

C 00008

C 00009

C 00010

C 00011

C 00012

C 00013

C 00014

C 00015

C 00016

C 00017

C 00018

C 00019

C 00020

C 00021

C 00022

C 00023

C 00024

C 00025

C 00026

C 00027

C 00028

C 00029

C 00030

C 00031

C 00032

C 00033

C 00034

C 00035

C 00036

C 00037

C 00038

C 00039

C 00040

C 00041

C 00042

C 00043

C 00044

C 00045

C 00046

C 00047

C 00048

C 00049

C 00050

C 00051

C 00052

C 00053

C 00054
GO TO 160

D(I) = 1.0

IF ( IP - I2 ) 170, 160, 190

D(I) = -YPI - YPI

A(I) = 0.0

IF ( IP - 2 ) 150, 160, 190

C

C PERIODIC SPLINE

C

C LAMBDA(N) = H(1) / ( H(1) + H(N-1) )

C

C SD(N) = 6. *(((Y(2)-Y(N))/H(1) - (Y(N)-Y(N-1))/H(N-1))/H(N-1)+H(1))

C

A(N) = 6. *(((Y(2)-Y(N))/C(1) - (Y(N)-Y(N-1))/C(N-1)) / (C(N-1)+C(1)))

C

B(2) = ( D(2) - 1.) / 2

A(2) = A(2) / 2

D(2) = -D(2) / 2

C

S(I) = S(I-1)*LAMBDA(I),1-1) / (2.*Q(I-1)*Q(1.-LAMBDA(I)))

C

U(I) = (SD(I)-U(I-1)) * (1.-LAMBDA(I)) / (2.*Q(I-1)*Q(I-1.-LAMBDA(I)))

C

Q(I) = -LAMBDA(I) / (2.*Q(I-1)*Q(1.-LAMBDA(I)))

C

DO 310 I = 3, N1

B(I) = (B(I-1)*D(I) - 1.) / (2.*D(I-1)*(1.-D(I)))

A(I) = (A(I)-A(I-1)*D(I)) / (2.*D(I-1)*(1.-D(I)))

D(I) = -D(I) / (2.*D(I-1)*(1.-D(I)))

DO 310 CONTINUE

C

T(N-1) = Q(N-1) + S(N-1)

V(N-1) = U(N-1)

T(K) = Q(K) * T(K+1) + S(K)

V(K) = Q(K) * V(K+1) + U(K)

C

B(N1) = D(N1) + B(N1)

D(N1) = A(N1)

DO 320 I = 2, N2

K = N - I

B(K) = D(K) * B(K+1) + B(K)

D(K) = D(K) * D(K+1) + A(K)

DO 320 CONTINUE

C

C MIN = ( SD(N) - V(N-1) + LAMBDA(N) * ( V(N-1) - V(2) ) ) / ( 2. + T(N-1) - LAMBDA(N) * ( T(N-1) - T(2) ) )

C

H(1) = MIN

C

A(N) = (A(N)-D(N)*D(N)*D(N)-D(2))/2.+B(N1)*D(N)*B(N1)-B(2)100106

C

A(1) = A(N)

DO 330 I = 1, N2

K = N - I

M(K) = T(K) * M(N) + V(K)

C

A(K) = B(K) * A(N) + D(K)

C

PAGE 2
C SET RIGHT END CONDITION
C
C ISN 0075 A(N) = 0.0
C ISN 0076 GO TO 200
C ISN 0077 B(N) = 1.0
C ISN 0078 A(N) = ( YPN - (Y(N)-Y(H1))/C(H1) ) * ( 6. / C(H1) )
C ISN 0079 GO TO 200
C ISN 0080 B(N) = 0.0
C ISN 0081 A(N) = YPN + YPN
C ISN 0082 B(N) = 0.0
C ISN 0083 A(N) = -2. * ( ( C(H1) + C(N-2) ) / C(N-2) )
C ISN 0084 D(N) = ( C(N) + C(H1) ) / C(N-2)
C
C G(1) = (LAMBDA(1)*SD(2)-2.*SD(1)) / (LAMBDA(1)*(1.-LAMBDA(2))-4.)
C H(1) = (LAMBDA(1)*LAMBDA(2)-2.*H) / (LAMBDA(1)*(1.-LAMBDA(2))-4.)
C G(2) = (SD(2)-G(1))*LAMBDA(2) / 2.
C H(2) = (LAMBDA(2)-W(1))*LAMBDA(2) / 2.
C
C 200 A(1) = (D(1)*A(2)-A(1)-A(1)) / (D(1)*(1.-D(2))-4.)
C ISN 0087 B(1) = (D(1)*D(2)-C(N)*C(N)) / (D(1)*(1.-D(2))-4.)
C ISN 0088 A(2) = (A(2)-A(1)*(1.-D(2))) / 2.
C ISN 0089 B(2) = (D(2)-B(1)*(1.-D(2))) / 2.
C
C G(I) = (SD(I-1)-G(I-1))*LAMBDA(I) / (2-W[I-1]*(1-LAMBDA(I)))
C H(I) = LAMBDA(I) / (2-H[I-1]*(1.-LAMBDA(I)))
C
C 210 CONTINUE
C
C M(N) = ( SD(N)-S*(G(N-2)-W(N-2)+G(N-1)-TAS*G(N-1) ) / ( 2-(TAS-S)*
C H(N-2) ) ) )
C MK = G(K) - W(K) * MK+1
C
C ISN 0094 A(N) = (A(N) - D(N) * (A(N2) - B(N2) * A(H1)) - B(N) * A(N1)) / 2.
C 1 / ( 2. - ( B(N) - D(N) * B(N2) ) * B(N1) )
C ISN 0095 DO 220 I = 1, N2
C ISN 0096 K = N - I
C ISN 0097 A(K) = A(K) - B(K) * A(K+1)
C ISN 0098 220 CONTINUE
C
C M(1) = G(1) - W(1) * M(3)
C
C ISN 0099 A(1) = A(1) - B(1)*A(3)
C
C COMPUTE COEFFICIENTS
C
C ISN 0100 DO 230 I = 1, N1
C D(I) = ( M(I+1) - M(I) ) / ( 6. * H(I) )
C B(I) = ( Y(I+1) - Y(I) ) / H(I) - H(I) * ( ( M(I+1) + 2 * M(I) ) / 6.00172
C
C ISN 0101 CONTINUE
C C(I) = M(I) / 2
C A(I) = Y(I)
C
ISH 0101  D(I) = ( A(I+1) - A(I) ) / ( 6. * C(I) )
ISH 0102  B(I) = ( Y(I+1) - Y(I) ) / ( C(I) - ( A(I+1)+A(I)+A(I) ) * (C(I) / 6. ) )
ISH 0103  C(I) = A(I) / 2
ISH 0104  A(I) = Y(I)
ISH 0105  230 CONTINUE
ISH 0106  240 RETURN
ISH 0107  END

*OPTIONS IN EFFECT* NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODEL(DBL4)
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NOBJAP NFORMAT COSTNT NOXREF NOALC NOANSF TERM IBM FLAGS(I)
*STATISTICS* SOURCE STATEMENTS = 106, PROGRAM SIZE = 2398, SUBPROGRAM NAME =BMFIT2
*STATISTICS* NO DIAGNOSTICS GENERATED
**** END OF COMPILATION ****

*STATISTICS* NO DIAGNOSTICS THIS STEP

2984K BYTES OF CORE NOT USED
**F64-LEVEL LINKAGE EDITOR OPTIONS SPECIFIED**

| MAP, LIST, LET, SIZE=(256K, 36K) |

**VARIABLE OPTIONS USED** - SIZE=(262144, 36864)

<table>
<thead>
<tr>
<th>OPTIONS</th>
<th>SIZE=(256K, 36K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUBIC</td>
<td></td>
</tr>
<tr>
<td>PDFM</td>
<td></td>
</tr>
<tr>
<td>FCD</td>
<td></td>
</tr>
</tbody>
</table>

**MODULE MAP**

<table>
<thead>
<tr>
<th>CONTROL SECTION</th>
<th>ENTRY</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>ORIGIN</td>
</tr>
<tr>
<td>STAEBL</td>
<td>00</td>
</tr>
<tr>
<td>GBIP</td>
<td>2A60</td>
</tr>
<tr>
<td>BIRDY</td>
<td>242CB</td>
</tr>
<tr>
<td>ECOORD</td>
<td>266F8</td>
</tr>
<tr>
<td>BEAM</td>
<td>246C8</td>
</tr>
<tr>
<td>COORDR</td>
<td>271A0</td>
</tr>
<tr>
<td>ESTIFF</td>
<td>276C0</td>
</tr>
<tr>
<td>EXHECK</td>
<td>280D0</td>
</tr>
<tr>
<td>FRP1</td>
<td>293CA</td>
</tr>
<tr>
<td>HOLZER</td>
<td>29690</td>
</tr>
<tr>
<td>INPUT</td>
<td>2A170</td>
</tr>
<tr>
<td>KDAG</td>
<td>2A520</td>
</tr>
<tr>
<td>LAMINA</td>
<td>2AF10</td>
</tr>
<tr>
<td>LAMIN1</td>
<td>2B910</td>
</tr>
<tr>
<td>LOGA2</td>
<td>2BCBA</td>
</tr>
<tr>
<td>NATADD</td>
<td>2C430</td>
</tr>
<tr>
<td>NATIPY</td>
<td>2C5E8</td>
</tr>
<tr>
<td>MATFRN</td>
<td>2C820</td>
</tr>
<tr>
<td>MINV</td>
<td>2CA40</td>
</tr>
<tr>
<td>MODAS</td>
<td>2D020</td>
</tr>
<tr>
<td>MODSTR</td>
<td>2D430</td>
</tr>
<tr>
<td>PARTH</td>
<td>2DE8F</td>
</tr>
<tr>
<td>PPI</td>
<td>2DC78</td>
</tr>
<tr>
<td>RBE2</td>
<td>2DE90</td>
</tr>
<tr>
<td>SECTN</td>
<td>2E468</td>
</tr>
<tr>
<td>SHAPE</td>
<td>327F8</td>
</tr>
<tr>
<td>SIULT</td>
<td>32E33</td>
</tr>
<tr>
<td>STABEL</td>
<td>33E48</td>
</tr>
<tr>
<td>STATIC</td>
<td>469A8</td>
</tr>
<tr>
<td>ZSTRES</td>
<td>43E00</td>
</tr>
<tr>
<td>THICK</td>
<td>49B10</td>
</tr>
<tr>
<td>TAIKU</td>
<td>4C110</td>
</tr>
<tr>
<td>HARP</td>
<td>4C5E8</td>
</tr>
<tr>
<td>HOLLOW</td>
<td>4CF60</td>
</tr>
<tr>
<td>EDOMFIT</td>
<td>4D930</td>
</tr>
<tr>
<td>IHOLASCN*</td>
<td>4E299</td>
</tr>
</tbody>
</table>

**IHOLATH2**

- **IHODARCS** 4E290  DARCOS  4E290  IHODARN  4E2A8  DARSIN  4E2A8

**IHOLCOS** 4E779

- **DCOS** 4E770  IHODCOS  4E770
<table>
<thead>
<tr>
<th>NAME</th>
<th>ORIGIN</th>
<th>LENGTH</th>
<th>NAME</th>
<th>LOCATION</th>
<th>NAME</th>
<th>LOCATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>IHOLSIN</td>
<td>4E9F0</td>
<td>2F0</td>
<td>DSIN</td>
<td>4E9F0</td>
<td>IHOLSIN</td>
<td>4E9F0</td>
</tr>
<tr>
<td>IHOLSRT</td>
<td>4ECE0</td>
<td>170</td>
<td>DSQRT</td>
<td>4ECE0</td>
<td>IHOLSRT</td>
<td>4ECE0</td>
</tr>
<tr>
<td>IHOLTINCT</td>
<td>4EE50</td>
<td>2F0</td>
<td>DCTAN</td>
<td>4EE50</td>
<td>IHOLTINCT</td>
<td>4EE50</td>
</tr>
<tr>
<td>FIOEXEC</td>
<td>4F140</td>
<td>14EE</td>
<td>TYPSTK#</td>
<td>4F38E</td>
<td>FOEXEC</td>
<td>50656</td>
</tr>
<tr>
<td>FIOPRET</td>
<td>50630</td>
<td>3500</td>
<td>FOXCALL</td>
<td>4F5E5</td>
<td>FIOPRET</td>
<td>53100</td>
</tr>
<tr>
<td>FIOCOMPL</td>
<td>53B30</td>
<td>2846</td>
<td>FOXCOUNT</td>
<td>53A8</td>
<td>FIOCOMPL</td>
<td>53B3A</td>
</tr>
<tr>
<td>IHOECOMM</td>
<td>56378</td>
<td>E54</td>
<td>DYNEMENT</td>
<td>53B50</td>
<td>IHOECOMM</td>
<td>53B7A</td>
</tr>
<tr>
<td>FIOAP#</td>
<td>571D0</td>
<td>6IC</td>
<td>IBCON#</td>
<td>563AC</td>
<td>FIOAP#</td>
<td>57110</td>
</tr>
<tr>
<td>IHOCOMH2#</td>
<td>577F0</td>
<td>AD4</td>
<td>INTSHTCH</td>
<td>57110</td>
<td>IHOCOMH2#</td>
<td>577F0</td>
</tr>
<tr>
<td>IHOCVTH*</td>
<td>582C8</td>
<td>CEA</td>
<td>SEQDASD</td>
<td>57BD6</td>
<td>IHOCVTH*</td>
<td>582C8</td>
</tr>
<tr>
<td>IHOEFNTH*</td>
<td>58F8B</td>
<td>800</td>
<td>ADCON#</td>
<td>582CB</td>
<td>IHOEFNTH*</td>
<td>58F8B</td>
</tr>
<tr>
<td>IHOLEXP#</td>
<td>597D0</td>
<td>3DC</td>
<td>ARITH#</td>
<td>58FB8</td>
<td>IHOLEXP#</td>
<td>597D8</td>
</tr>
<tr>
<td>IHOFIOS#</td>
<td>55B98</td>
<td>11C</td>
<td>DEXP</td>
<td>597B8</td>
<td>IHOFIOS#</td>
<td>55B98</td>
</tr>
<tr>
<td>IHOFIOS2#</td>
<td>5AD28</td>
<td>642</td>
<td>FIOCS#</td>
<td>5939E</td>
<td>IHOFIOS2#</td>
<td>5AD28</td>
</tr>
<tr>
<td>IHOERR1*</td>
<td>56370</td>
<td>624</td>
<td>ERRMON</td>
<td>58370</td>
<td>IHOERR1*</td>
<td>56370</td>
</tr>
<tr>
<td>INQUIATBL*</td>
<td>5B998</td>
<td>328</td>
<td>FQCONI#</td>
<td>5C188</td>
<td>INQUIATBL*</td>
<td>5B998</td>
</tr>
<tr>
<td>IHOUOPT#</td>
<td>58CC0</td>
<td>4C0</td>
<td>FQCONI#</td>
<td>5C188</td>
<td>IHOUOPT#</td>
<td>58CC0</td>
</tr>
<tr>
<td>IHOFCONI*</td>
<td>5C168</td>
<td>416</td>
<td>FQCONI#</td>
<td>5C188</td>
<td>IHOFCONI*</td>
<td>5C168</td>
</tr>
<tr>
<td>IHOFCOD#</td>
<td>5C5A0</td>
<td>8B8</td>
<td>FQCONO#</td>
<td>5C5A0</td>
<td>IHOFCOD#</td>
<td>5C5A0</td>
</tr>
<tr>
<td>IHOETRCH#</td>
<td>5CE58</td>
<td>2AE</td>
<td>IHOTRCH</td>
<td>5C5A0</td>
<td>IHOETRCH#</td>
<td>5CE58</td>
</tr>
<tr>
<td>PHPPOSTDP#</td>
<td>50108</td>
<td>128</td>
<td>ERRTRA</td>
<td>5CE60</td>
<td>PHPPOSTDP#</td>
<td>50108</td>
</tr>
<tr>
<td>IHOFTEGR</td>
<td>50230</td>
<td>220</td>
<td>FTEN#</td>
<td>5D230</td>
<td>IHOFTEGR</td>
<td>50230</td>
</tr>
<tr>
<td>BLKCH</td>
<td>59450</td>
<td>6904</td>
<td></td>
<td></td>
<td>BLKCH</td>
<td>59450</td>
</tr>
<tr>
<td>BLK2</td>
<td>63D58</td>
<td>20</td>
<td></td>
<td></td>
<td>BLK2</td>
<td>63D58</td>
</tr>
<tr>
<td>BLK4</td>
<td>63D78</td>
<td>2C</td>
<td></td>
<td></td>
<td>BLK4</td>
<td>63D78</td>
</tr>
<tr>
<td>BLK7</td>
<td>63D8A</td>
<td>4EC</td>
<td></td>
<td></td>
<td>BLK7</td>
<td>63D8A</td>
</tr>
<tr>
<td>BLK12</td>
<td>64298</td>
<td>348</td>
<td></td>
<td></td>
<td>BLK12</td>
<td>64298</td>
</tr>
<tr>
<td>BLKAA</td>
<td>645E0</td>
<td>348</td>
<td></td>
<td></td>
<td>BLKAA</td>
<td>645E0</td>
</tr>
<tr>
<td>BLKB</td>
<td>64928</td>
<td>998</td>
<td></td>
<td></td>
<td>BLKB</td>
<td>64928</td>
</tr>
</tbody>
</table>
DIAGNOSTIC MESSAGE DIRECTORY
IEN0132 ERROR - SYMBOL PRINTED IS AN UNRESOLVED EXTERNAL REFERENCE.
<table>
<thead>
<tr>
<th>SOURCE STATEMENTS</th>
<th>PROGRAM SIZE</th>
<th>SUBPROGRAM NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>63</td>
<td>846</td>
<td>COPE08</td>
</tr>
<tr>
<td>544</td>
<td>9868</td>
<td>COPE09</td>
</tr>
<tr>
<td>69</td>
<td>2038</td>
<td>COPE10</td>
</tr>
<tr>
<td>57</td>
<td>1134</td>
<td>COPE11</td>
</tr>
<tr>
<td>125</td>
<td>2844</td>
<td>COPE12</td>
</tr>
<tr>
<td>23</td>
<td>566</td>
<td>COPE13</td>
</tr>
<tr>
<td>95</td>
<td>2632</td>
<td>COPE14</td>
</tr>
<tr>
<td>63</td>
<td>1720</td>
<td>COPE15</td>
</tr>
<tr>
<td>29</td>
<td>878</td>
<td>COPE16</td>
</tr>
<tr>
<td>26</td>
<td>680</td>
<td>COPE17</td>
</tr>
<tr>
<td>49</td>
<td>1408</td>
<td>COPE18</td>
</tr>
<tr>
<td>157</td>
<td>3780</td>
<td>SIIICON</td>
</tr>
<tr>
<td>248</td>
<td>3980</td>
<td>MAIN</td>
</tr>
</tbody>
</table>
C DATA SET U498SFOD AT LEVEL 006 AS OF 03/01/82 00001
C DATA SET U498SFOD AT LEVEL 004 AS OF 02/08/82 00002
C DATA SET U498SFOD AT LEVEL 002 AS OF 11/16/81 00003

SUBROUTINE FODUIR,NC,RPIIl,AlPHAl
C
C LOCAL STRESS BIRD INGESTION PARAMETER
C
IMPLICIT REAL*8(A-H,O-Z)
REAL*4 VALT,BRSU,TLS
COMMON /INPUT/(X(1000),Y(1000),Z(1000),T(1000)) 00004
COMMON /BLK 3/FH,BLAD,BETA,THT,NST1,NSE 00005
COMMON /BLKAA/ALPMIN(21),XSC(21),YSC(21),XXI(21),YYY(21) 00006
COMMON /BLK 8/SNB(21),ITLTE(18),VARI(25),TMX(21),HALPHA(21) 00007
COMMON /BLAY/TH(7,25),EMASS(25),BPSAH(25),BUDTH(25) 00008
COMMON /BIRD/BLSNAX(21) 00009
COMMON /PLY/PLY(21,25,7),THETA(7),RHT1(21) 00010
COMMON /STRS/RE(12,12,25,21),OUT1(12,21),SL(25,21),SH(25,21) 00011
COMMON /SIML/P01(21,ittle(181),VARI(235),THAX(21),HALPHA(21) 00012
DIMENSION TB1(25),TB2(25),XXI(25),ALPHAl(1) 00013
WRITE(6,270) 00014
DATA SET U498SFOD AT LEVEL 004 AS OF 02/08/82 00015
DATA SET U498SFOD AT LEVEL 002 AS OF 11/16/81 00016
DATA SET U498SFOD AT LEVEL 006 AS OF 03/01/82 00017

CALCULATE LSBIP AT EACH STATION, LOCATING A MAXIMUM FOR EACH
STATION AND STORING IT IN BLSNAX(21)

DO 500 KK=1,NR 00018
C
VBIRD = 180.0 00019
NB = BLADES 00020
PI = 3.14159265 00021
ALPHA = ALPHAI(NSTA) 00022
RPN = RPM * 30. / PI 00023
V1 = .42 * 12. * VBIRD * DSIN(ALPHA) 00024
NODE = (N斯塔-1) * NC + 1 00025
V2 = PI * X(NODE) * RPM / 30. * DSIN(ALPHA) 00026
VBIADE = X(NODE)/12. * RPM / PI / 30. 00027
WRITE(6,400) N斯塔,NC,RPN,ALPHA,NB,VBIRD,XSC(N斯塔) 00028
400 FORMAT(/5X,'FOD INPUT',5X,'STATION = ',15X,'19X','HC = ',15X,'19X', 00029
1 'I15X,'RPM(REV/MIN) = ',F7.2,'/19X,'ALPHA CHG3D(RADIAN) = ',F7.2,'/19X', 00030
2 'I15X,'E12.5.,/19X,'NUMER OF BLADES = ',E12.5.,/19X', 00031
3 'BIRD VELOCITY (FT/SEC) = ',E12.5.,/19X', 00032
4 'SHEAR CENTER (FROM TE) = ',E12.5.,/ 00033

WRITE(6,270) 00034
DO 500 KK = 1,NR 00035
NODE = (N斯塔-1) * NC + 1 00036
VBIRD = X(NODE)/12. * RPM / PI / 30. 00037
VBIADE = X(NODE)/12. * RPM / PI / 30. 00038
WRITE(6,270) 00039

DATA SET U498SFOD AT LEVEL 002 AS OF 11/16/81 00040
DATA SET U498SFOD AT LEVEL 004 AS OF 02/08/82 00041
DATA SET U498SFOD AT LEVEL 006 AS OF 03/01/82 00042

DATA SET U498SFOD AT LEVEL 004 AS OF 02/08/82 00043
DATA SET U498SFOD AT LEVEL 002 AS OF 11/16/81 00044
DATA SET U498SFOD AT LEVEL 006 AS OF 03/01/82 00045

C PRINT MATRIX VALUES
C
C DO 500 KK = 1,NR 00046
C
V2 = PI * X(NODE) * RPM / 30. * DSIN(ALPHA) 00047
VBIADE = X(NODE)/12. * RPM / PI / 30. 00048

C ZERO MATRICES
C
DATA SET U498SFOD AT LEVEL 002 AS OF 11/16/81 00049
DATA SET U498SFOD AT LEVEL 004 AS OF 02/08/82 00050
DATA SET U498SFOD AT LEVEL 006 AS OF 03/01/82 00051

DO 500 KK = 1,NR 00052
TC1(KK) = 0. 00053
TB2(KK) = 0. 00054

C DATA SET U498SFOD AT LEVEL 006 AS OF 03/01/82 00055
C DATA SET U498SFOD AT LEVEL 004 AS OF 02/08/82 00056
C DATA SET U498SFOD AT LEVEL 002 AS OF 11/16/81 00057
C AREA CALCULATION - METAL AREA ONLY

C AREA(I) = BWIDTH(I) * TB(I)
IS 0052 AREA(I) = BWIDTH(I) * TB(I)
DO 101 I = 2,NC
IS 0054 AREA(I) = AREA(I-1) + BWIDTH(I) * TB(I)
C CALL MATPRNAREAAREA,25,1,'AREA')
C C EVALUATING THE DOUBLE AREA INTEGRAL , AAREA

C AAREA(I) = AAREA(I) * BWIDTH(I)
IS 0055 AAREA(I) = AAREA(I) * BWIDTH(I)
DO 102 I = 2,NC
IS 0057 AAREA(I) = AAREA(I-1) + AAREA(I) * BWIDTH(I)
C CALL MATPRN(AAREA,25,1,'AAREA')
C C TORSION TERMS - THIN SECTION ASSUMPTION

C TAVE = ( T(NODE) + T(NODE1) ) / 2.
IS 0054 TAVE = ( T(NODE) + T(NODE1) ) / 2.
IS 0056 TAVE = 0.
IS 0058 TAVE = TAVE + PLY(I,I,I,J)
C C TORSION TERMS - THIN SECTION ASSUMPTION

C NODE = (NSTA-1) * NC + 1
IS 0058 NODE = (NSTA-1) * NC + 1
IS 0059 NODE1 = NODE + NC
IS 0060 IF (NODE1 .GE. NRNC) NODE1 = NODE - NC
IS 0061 IF (NODE1 .GE. NRNC) NODE1 = NODE - NC
IS 0062 DO 200 I = 1,NC
IS 0064 TAVE = ( T(NODE) + T(NODE1) ) / 2.
IS 0066 DO 201 J = 1,7
IS 0068 NODE1 = NODE1 + 1
IS 0069 TAVE = TAVE + PLY(I,I,I,J)
IS 0070 THOL = TAVE - TB(I)
IF(TST(I)+T62(I)+.0001).LT. TAVE)
TORS(I) = TOR2
IF(I.EQ.1) GO TO 200
TORS(I) = TORS(I-1) + TORS(I)

DO 200 NODE = NODE + 1
C
C CALL MATPFL(TORS,25,1,'TORS')
C
C DISTANCE FROM L.E., XX
C
XX(I) = EWIDTH(I)/2.
DO 210 I = 2,NC
XX(I) = XX(I-1) + BWIDTH(I-1)/2. + BWIDTH(I)/2.
C CALL MATPFL(XX,25,1,'XX')
C
XBAR = DISTANCE FROM LOAD CENTER TO C.G.
C
TDX = 0.
DO 350 I = 1,NC
IF(XX(I).LT.XF) GO TO 333
XX .GT. XF
BIP(I) = P/TB(I)**2 * (XX(I) - XF/2. - AREA(I)/AREA(NC) - 1)
XBAR / TORS(NC) * TORS(I) )
GO TO 300
C
XX .LE. XF
C
BIP(I) = P/TB(I)**2 * (XX(I) - XF/2. - AREA(I)/AREA(NC) - 1)
XBAR / TORS(NC) * TORS(I) )
C
GO TO 300
C
CALL MATPFL(BIP,25,1,'BIP')
C
SEARCH FOR MAXIMUM BIP FOR SECTION KK
C
BLSMAX(KK) = 0.
DO 405 I = 1,NC2
IF(BIP(I).GT. BLSMAX(KK)) BLSMAX(KK) = BIP(I)
CONTINUE
C
CALL MATPFL(6,506,'BIP', STATION = ',I5,5X,E12.5)
C
WRITE(6,506) BLSMAX(KK)
C
DO 500 CONTINUE
C 00173
ISN 0105 RETURN
ISN 0106 END
*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
*OPTIONS IN EFFECT*SOURCE EBCDIC NODIST NODECK OBJECT NODMAP NODFORMAT GOSTMT NODIRF NODAHC NOANSF TERM ICH FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 105, PROGRAM SIZE = 4294, SUBPROGRAM NAME = FOD
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPILATION *****

2988K BYTES OF CORE NOT USED
REQUESTED OPTIONS: SOURCE, NOLIST, NOXREF, NOLOB, OPT(1), AUTO abide(NONE), NOALC

OPTIONS IN EFFECT: NAME(MAIN) OPTIMIZE(3) LINCOUNT(60) SIZE(HAX) AUTObol(NONE)

SOURCE EBCDIC NOLIST NOEDGE OBJECT NOHAP NOFORHAT GOSTML NOXREF NOALC NOHANSF TERM IBM FLAG(I)

**PCBAD**
+VERSION 1.3.0 (01 MAY 80) ANALZ SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)

ISN 0042 IF (NRF .NE. 0) CALL RDT993 (ICALC)
ISN 0044 CALL THAX (ICALC)
ISN 0045 CALL WTHI37 (ICALC)
ISN 0046 DO 150 I = 1,NTIS
ISN 0047 THKVAL(I) = VALT(I)
ISN 0048 150 CONTINUE
ISN 0049 BRCC = BRSV
ISN 0050 CALL MYTIME (ITIME)
ISN 0051 ITIMEC = 3
ISN 0052 WRITE (26,3000) ITIMEC, ITIME
ISN 0053 RETURN
ISN 0054 200 IF (ICALC .GE. 2) GO TO 300
ISN 0056 CALL MYTIME (ITIME)
ISN 0057 ITIMEC = 4
ISN 0058 WRITE (26,3000) ITIMEC, ITIME
ISN 0059 NBRS = ERS + .05
ISN 0060 IF (NCD .EQ. 2) GO TO 210
ISN 0062 HLRTIO = (DTIP + DROG) / (R(NSTA) - R(1))
ISN 0063 ECRTIO = (DLE + DTE) / BRCC
ISN 0064 WRITE (161,1000)
ISN 0065 WRITE (161,238) DLE, DTE, DROG, DTIP, TTI, TLT
ISN 0066 GO TO 220
ISN 0067 WRITE (161,1030)
ISN 0068 VAL = (2.0 * TIS + TIC) / (THKVAL(I))
ISN 0069 WRITE (161,238) TIS, TIC, PCBA, PCGE, VAL
ISN 0070 220 CONTINUE
ISN 0071 WRITE (161,1010)
ISN 0072 WRITE (161,238) (THETA(I), I=1,7)
ISN 0073 CALL RWHI37 (ICALC)
ISN 0074 CALL CALCTH
ISN 0075 CALL THAX (ICALC)
ISN 0076 CALL WTHI37 (ICALC)
ISN 0077 CALL MH4137
ISN 0078 CALL RSFUNC
ISN 0079 G5JF = BLADV
ISN 0080 CALL OBJTV
ISN 0081 WRITE (161,930)
ISN 0082 WRITE (161,238) OBJF, OBJFH
ISN 0083 NROOT = ROOT + .01
ISN 0084 DO 230 I = 1,NROOT
ISN 0085 FH(I) = FREQ(I)
ISN 0086 230 CONTINUE
ISN 0087 DO 232 I = 1,NTIS
ISN 0088 N = IST(I)
ISN 0089 TOVB(I) = TCC(N) / BCC(N)
ISN 0090 232 CONTINUE
ISN 0091 WRITE (161,1020)
ISN 0092 WRITE (161,238) (TOVB(I), I=1,NTIS)
ISN 0093 DO 235 I = NBRH,NSTA
ISN 0094 FODSIB(I) = BLSMAX(I)
ISN 0095 235 CONTINUE
ISN 0096 FODMAX = 0.0
ISN 0097 RCHKR = R(NSTA) + 2.0 + .005
ISN 0098 RCHKT = R(NSTA) - 2.0 + .005
ISN 0099 DO 234 K = NBRH,NSTA
ISN 0100 IF (R(K) LT. RCHKR) GO TO 234
ISN 0102 IF (R(K) GT. RCHKT) GO TO 234
ISN 0104 BMAX = BLSMAX(K)

DATE 82.141/10.52.24 PAGE 2
I'-1.3.0 (01 MAY 80) ANALIZ SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 82.141/10.52.24 PAGE 3

ISH 0105  FODMAX = AMAX1(FODMAX,BMAX)  00114
ISH 0106  IF (FODMAX .EQ. BMX) RMAX = R(K)  00115
ISH 0108  WRITE (I6I,230) FODMAX, RMAX  00116
ISH 0109  234 CONTINUE  00117
ISH 0110  WRITE (I6I,970)  00118
ISH 0111  WRITE (I6I,238) (FODL syll, I=NBRS,NSTA)  00119
ISH 0112  236 FORMAT (/3(3X,E12.5))  00120
ISH 0113  238 FORMAT (8(3X,E12.5))  00121
ISH 0114  DO 237 I = 1,N  00122
ISH 0115  SMAXJ(I) = SMAX(I)  00123
ISH 0116  SMAX2(I) = SMAX2(I)  00124
ISH 0117  SMAX3(I) = SMAX3(I)  00125
ISH 0118  SMAX 4(I) = SMAX4(I)  00126
ISH 0119  SMAXT(I) = SMAXT(I)  00127
ISH 0120  237 CONTINUE  00128
ISH 0121  WRITE (I6I,940)  00129
ISH 0122  WRITE(I6I,238) (SMAXI (I), I=1,N)  00130
ISH 0123  WRITE (I6I,950)  00131
ISH 0124  WRITE(I6I,230) (SMAX2(I), I=1,N)  00132
ISH 0125  WRITE (I6I,960)  00133
ISH 0126  WRITE(I6I,238) (SMAX3(I), I=1,N)  00134
ISH 0127  WRITE (I6I,980)  00135
ISH 0128  WRITE (I6I,238) (SMAX4(I), I=1,N)  00136
ISH 0129  WRITE (I6I,990)  00137
ISH 0130  WRITE (I6I,238) (SMAXT(I), I=1,N)  00138
ISH 0131  CALL MYTIME (ITIME)  00139
ISH 0132  ITIMEC = 5  00140
ISH 0133  WRITE (26,3000) ITIMEC, ITIME  00141
ISH 0134  IF (NRF .EQ. 0) GO TO 240  00142
ISH 0135  CALL IT983 (ICALC)  00143
ISH 0137  WRITE (I6I,920)  00144
ISH 0138  DO 240 I = 1,NRF  00145
ISH 0139  WRITE (I6I,910) ND(I), DLSV(I)  00146
ISH 0140  DLAR(I) = DLSV(I)  00147
ISH 0141  240 CONTINUE  00148
ISH 0142  CALL MYTIME (ITIME)  00149
ISH 0143  ITIMEC = 6  00150
ISH 0144  WRITE (26,3000) ITIMEC, ITIME  00151
ISH 0145  RETURN  00152
ISH 0146  300 CONTINUE  00153
ISH 0147  CALL MYTIME (ITIME)  00154
ISH 0148  ITIMEC = 7  00155
ISH 0149  WRITE (26,3000) ITIMEC, ITIME  00156
ISH 0150  CALL RMN37 (ICALC)  00157
ISH 0151  CALL CMNTH  00158
ISH 0152  CALL TMAX (ICALC)  00159
ISH 0153  CALL MN137 (ICALC)  00160
ISH 0154  ICALC = 4  00161
ISH 0155  CALL MN137 (ICALC)  00162
ISH 0156  ICALC = 3  00163
ISH 0157  IF (NRF .EQ. 0) GO TO 350  00164
ISH 0158  INAVE = -1  00165
ISH 0159  IND = 4  00166
ISH 0160  MODE = 1  00167
ISH 0161  CALL WTN37 (ICALC,MODE,IND,INAVE)  00168
ISH 0162  MODE = 2  00169
ISH 0163  CALL WTN37 (ICALC,MODE,IND,INAVE)  00170
ISH 0164  MODE = 3  00171
ISH 0165  CALL WTN37 (ICALC,MODE,IND,INAVE)  00172
ISH 0166  350 CONTINUE
ISH 0167     NBR5 = ERS + .05
ISH 0168     WRITE (I6I,900) (TIX(I),I=NBR5,NSTA)
ISH 0169     WRITE (I6I,900) (TCX(I),I=NBR5,NSTA)
ISH 0170     900 FORMAT (/312(X,F10.6))
ISH 0171     910 FORMAT (5X,E5.5X,K12.5)
ISH 0172     920 FORMAT (/6X,'NO-SAVE DELAERO-SAVE')
ISH 0173     930 FORMAT (/6X,'HEIGHT OBJECT FUNCTION')
ISH 0174     940 FORMAT (/5X,'SHAX1(I) I = 1,8')
ISH 0175     950 FORMAT (/5X,'SHAX2(I) I = 1,8')
ISH 0176     960 FORMAT (/5X,'SHAX3(I) I = 1,8')
ISH 0177     970 FORMAT (/5X,'LSBIP(I) I = 1,NTIS')
ISH 0178     980 FORMAT (/5X,'SHAXLE(I) I = 1,8')
ISH 0179     990 FORMAT (/5X,'SHAXTE(I) I = 1,8')
ISH 0180     1000 FORMAT (/8X,'DLE',12X,'DTE',11X,'DROOT',10X,'DTIP',
ISH 0181                11X,'TII',11X,'TBT',12X,'HLRATIO',9X,'ECRATIO')
ISH 0181     1010 FORMAT (/5X,'THETA(I) I = 1,7')
ISH 0182     1020 FORMAT (/5X,'T/B(I) I = 1,NTIS')
ISH 0183     1030 FORMAT (/6X,'TIS',12X,'TIC',11X,'PCBA',11X,'PCGE',9X,'T RATIO')
ISH 0184     CALL MYTIME (ITIME)
ISH 0185     ITIMEC = 8
ISH 0186     WRITE (26,3000) ITIMEC, ITIME
ISH 0187     RETURN
ISH 0188     END

*OPTIONS IN EFFECT=NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOINL(NONE)
*OPTIONS IN EFFECT=SOURCE EBCDIC NOLIST NODECK NODMAP NODSMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 167, PROGRAM SIZE = 3568, SUBPROGRAM NAME = ANALIZ
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPILATION *****
2980K BYTES OF CORE NOT USED
DATA SET U98SRDATA AT LEVEL 005 AS OF 03/15/82
DATA SET U98SRDATA AT LEVEL 004 AS OF 02/24/82
DATA SET U98SRDATA AT LEVEL 003 AS OF 02/23/82
DATA SET U98SRDATA AT LEVEL 002 AS OF 01/29/82
DATA SET U98SRDATA AT LEVEL 001 AS OF 11/12/81
DATA SET U500RDATA AT LEVEL 003 AS OF 07/16/81
DATA SET U500RDATA AT LEVEL 002 AS OF 06/18/81
DATA SET U500RDATA AT LEVEL 001 AS OF 06/08/81

SUBROUTINE RDDATA

COMMON /UIOS/ ISI, I6I
COMMON /ANAL54/ ND(5), DLSV(5)
COMMON /ANAL30/ NTIS, NRF, IST(21), VALT(21), BRSV
COMMON /ANAL31/ NSFA, ISTH(21)
COMMON /ANAL32/ RMIN, RMAX, RORD(4), NORD
COMMON /ANAL33/ DLED, DTED, DROOTD, DTFPD, TTD, TLTD, NCD

1, HCK, TISO, TID, PCBAD, PCGED

COMMON /ANAL04/ NSTD, NS1, NS2, NOACR, I1T, LONG, ISTE
1 DENG, EB, POIB, BR, SERIES, BRS, BTS, IOPP

COMMON /FLY/ FLY(21,25,7), THETAD(7), RIG(7)
COMMON /GLCSCH/ OBJF, FH(5), DLAR(5), THKVAL(21), RF(5,4)
1, BRCC, FODLS(21), DLE, DTE, DROOT, DTFP, TTI, TLTD
2, OBJFUN, SMA(8), SMA(8), SMA(8)
3, SMA(8), SMA(8), THET(7), HLTSTO, ECRTIO, TOVU(21)
4, FODMAX, TIS, TIC, PCBA

DOUBLE PRECISION DLED, DTED, DROOTD, DTFPD, TTD, TLTD
1, THETAD, FLY, RHO, TISO, TID, PCBAD, PCGED

READ NTIS - NUMBER OF THICKNESS INPUT STATIONS (MAX OF 21)
READ NRF - NUMBER OF ROOTS CALCULATED BY FLUTTER ANALYSIS
READ NSFA - NUMBER OF H137 OUTPUT STATIONS USED FOR THE FLUTTER ANALYSIS (MAX OF 21)
READ NORD - NUMBER OF ORDERS FOR RESONANCE FUNCTION CALCULATION (MAX OF 4)

READ (ISI,900) NTIS, NRF, NSFA, NORD, BRSV
READ NTIS VALUES OF STATION NUMBER AND THICKNESS
READ (ISI,901) (IST(I),VALT(I), I = 1,NTIS)
READ NRF VALUES OF NODAL DIAMETER (1 PER ROOT) FOR INITIAL STARTING VALUES FOR THE FLUTTER CALCULATION
READ H137 OUTPUT STATION NUMBERS TO BE USED IN THE FLUTTER ANALYSIS

IF (NRF .EQ. 0) GO TO 70
IF (ND(1) .EQ. 0) ND(1) = -4
IF (ND(2) .EQ. 0) ND(2) = 4
IF (ND(3) .EQ. 0) ND(3) = 4
IF (ND(4) .EQ. 0) ND(4) = -4
IF (ND(5) .EQ. 0) ND(5) = 4

READ H137 OUTPUT STATION NUMBERS TO BE USED IN THE FLUTTER ANALYSIS

REQUESTED OPTIONS: SOURCE, NOLIST, NOXREF, NODECK, OPT(3), AUTOOB(NONE), NOALC
OPTIONS IN EFFECT: NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOOB(NONE)
CONTINUE
READ (15,903) ISTH!,I
$I_0$: $I_0$ + .05

$L_0$ : $L_0$ + 1

IF ($L_0$.NE. NSFA) GO TO 70

CONTINUE
70 CONTINUE

READ DATA ASSOCIATED WITH HOLLOW FOIL

DLE = DISTANCE TO HOLE FROM LEADING EDGE
DTE = DISTANCE TO HOLE FROM TRAILING EDGE
DROOT = DISTANCE TO HOLE FROM FOIL ROOT
DTIP = DISTANCE TO HOLE FROM FOIL TIP
TTI = THICKNESS OF TITANIUM
TLT = THICKNESS OF BORIC TITANIUM

READ (15,906) DLE, DTE, DROOT, DTIP, TTI, TLT

IF (NCD .EQ. 2) GO TO 90

CONTINUE

READ DATA ASSOCIATED WITH COMPOSITE FOIL

TIS = TITANIUM SKIN THICKNESS
TIC = TITANIUM CENTER THICKNESS
PCBA = PERCENT OF BORON ALUMINUM

READ (15,907) TIS, TIC, PCBA

00039
00040
00041
00042
00043
00044
00045
00046
00047
00048
00049
00050
00051
00052
00053
00054
00055
00056
00057
00058
00059
00060
00061
00062
00063
00064
00065
00066
00067
00068
00069
00070
00071
00072
00073
00074
00075
00076
00077
00078
00079
00080
00081
00082
00083
00084
00085
00086
00087
00088
00089
00090
00091
00092
00093
00094
00095
00096
00097
00098
00099
00100
00101
00102
00103
00104
00105
00106
00107
00108
00109
00110
00111
00112
00113
**VERSION 1.3.0 (01 MAY 80)  RDDATA  SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 02.141/10.52.30  PAGE 3**

1. **PCGED = 1.0 - PCBAD**
2. **PCBAD 00114**
3. **PCBAD 00115**
4. **PCBAD 00116**
5. **PCBAD 00117**
6. **PCBAD 00118**
7. **PCBAD 00119**
8. **PCBAD 00120**
9. **PCBAD 00121**
10. **PCBAD 00122**
11. **PCBAD 00123**
12. **PCBAD 00124**
13. **PCBAD 00125**

**OPTIONS IN EFFECT**
- NAME (MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NO)
- SOURCE EBCDIC NOLIST NODECK OBJECT NOLAP NOFORMAT GOSTHT NOXREF NOLC NOSIF TERM IBM FLAG(I)
- SOURCE STATEMENTS = 70, PROGRAM SIZE = 1244, SUBPROGRAM NAME = RDDATA

**STATISTICS**
- NO DIAGNOSTICS GENERATED

*** END OF COMPILATION ***
DATA SET U498SCALCT AT LEVEL 010 AS OF 01/29/82
DATA SET U498SCALCT AT LEVEL 009 AS OF 01/26/82
DATA SET U498SCALCT AT LEVEL 008 AS OF 10/19/81
DATA SET U498SCALCT AT LEVEL 007 AS OF 07/23/81
DATA SET U498SCALCT AT LEVEL 006 AS OF 07/17/81
DATA SET US00SCALCT AT LEVEL 005 AS OF 07/16/81
DATA SET US00SCALCT AT LEVEL 004 AS OF 06/17/81

SUBROUTINE CALCTH

COMMON /NIOUS/ IST, I61
COMMON /ANAL04/ NSTA, MIST1, MIST2, NOACR, ILT, ISTE, 1 DENS, EB, POID, BR, SERIES, BRS, ITS, IGPP

COMMON /ANAL05/ R(21), BBSR(21), TOB(21), THSTH(21), ALPHA(21), 1 SOB(21), RLE(21), RTE(21), ADIT(21), O(21)

COMMON /ANAL17/ XPS(3,53,211, THKVEL(21), TCC(21), BMX(21)

COMMON /ANAL30/ NTIS, RF, IST(21), VALT(21), BRSV

COMMON /GLOBCM/ OBJF, FN(5), DLAR(5), THKVAL(21), RF(5,4), 1, BRCC

DIMENSION RIST(21), THKVEL(21)

WRITE (161,910) ISTA, THKVAL(21)
DO 10 I = 1, ISTA
    RIST(I) = R(N)
    THKVEL(I) = THKVEL(I)
    WRITE (161,910) I, THKVEL(I)
10 CONTINUE

WRITE (161,910) ISTA, RIST(21), THKVEL(21)

WRITE (161,920) ISTA THICKNESS CHORD
TCC(I) = THKVEL(I) - THKVAL(I)
IF (R(I).LE. RIST(I)) GO TO 15
N = N + 1
IF (N.GT. NTIS) GO TO 30
GO TO 15
15 TCC(I) = THKVEL(N-1) - THKVEL(N-1) + THKVEL(N) + R(I) - RIST(N-1)
    I = RIST(I)
    WRITE (161,910) I, TCC(I), BRCC
10 CONTINUE

*VERSION 1.3.0 (01 MAY 80)  CALCITH SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 82.141/10.52.33  PAGE 2

C WRITE (I6I,910) N , THKVEL(1)  00055
ISN 0039          RETURN  00056
ISN 0040          30 WRITE (I6I,900) N , NTIS  00057
ISN 0041          900 FORMAT (' N = ',I5,' NTIS = ',I5,' N EXCEEDS NTIS'/)  00058
ISN 0042          STOP  00059
ISN 0043          END  00060

*OPTIONS IN EFFECT*MAIN optimize(3) linecount(60) size(max) autodbl(None)
*OPTIONS IN EFFECT*source EBCDIC nolist nodeck object nodmap nodformat goshti notemp nodalc nodasmf term IBM flag(i)
*STATISTICS* source statements = 42, program size = 1032, subprogram name =CALCITH
*STATISTICS* no diagnostics generated
****** END OF COMPILATION ******

3012k bytes of core not used
DATA SET U498SIT983 AT LEVEL 009 AS OF 03/10/82
DATA SET U498SIT983 AT LEVEL 009 AS OF 03/10/82
DATA SET U5000IT983 AT LEVEL 009 AS OF 03/10/82

ISN 0002
SUBROUTINE ITT983 (ICALC)

ISN 0004
COMMON /UIGS/ ISI, I61

ISN 0005
COMMON /ANALOG/ NTIS, NRF, I57(21), VALT(21)

ISN 0006
COMMON /SECO/ FLAP(21,21), THST(21,21), ACOEF(21,4), ZODALD(21),
1 RED(21), DELAER(21), F2HSD, F2PSD

ISN 0007
COMMON /JMPCHK/ JMP(5), MODE

ISN 0008
COMMON /ANALOG/ MAY, MAX, ISTR, NBB, DENA, ED

ISN 0009
NMIN = 2

ISN 010
MNMIN = -2

ISN 011
NNAX = ( (BLADES + .05) / 2. ) -2.

ISN 012
MNMAX = -NNAX

ISN 013
LIN = NRF

ISN 014
DO 5 I = 1,5

ISN 015
JMP(I) = 0

ISN 016
5 CONTINUE

ISN 017
WRITE (161,110)

ISN 018
MODE = 1

ISN 019
10 INC = 1

ISN 020
IHAVE = 1

ISN 021
IF (ND(MODE) .NE. NMIN) GO TO 13

ISN 022
ND(MODE) = NMIN

ISN 023
GO TO 17

ISN 024
13 IF (ND(MODE) .EQ. MNMAX .OR. ND(MODE) .EQ. MMIN)
1 ND(MODE) = ND(MODE) + 1

ISN 025
IF (ND(MODE) .EQ. MNMAX .OR. ND(MODE) .EQ. MMIN)
1 ND(MODE) = ND(MODE) - 1

ISN 026
17 IF ( ND(MODE) .LT. 0 ) IHAVE = -1

ISN 027
NODE = IABS (ND(MODE))

ISN 028
CALL WTT983 (ICALC,MODE,NODE,IHAVE)

ISN 029
CALL MMT983

ISN 030
CALL DELSAV = DELAER(1)

ISN 031
NSAV = ND(MODE)

ISN 032
WRITE (161,100) DELSAV, DELAER(1), NSAV, ND(MODE)

ISN 033
NOD = ND(MODE)

ISN 034
CALL NDT983 (NOD,NODE,IHAVE,INC)

ISN 035
ND(MODE) = NODE * IHAVE

ISN 036
CALL WTT983 (ICALC,MODE,NODE,IHAVE)

ISN 037
CALL MMT983

ISN 038
IF (DELAER(1) .EQ. DELSAV) GO TO 40

ISN 039
IF (DELAER(1) .LT. DELSAV) GO TO 20

ISN 040
INC = -1

ISN 041
WRITE (161,100) DELSAV, DELAER(1), NSAV, ND(MODE)

ISN 042
ND(MODE) = ND(MODE) + INC

ISN 043
IF (ND(MODE) .EQ. 1) ND(MODE) = MMIN

ISN 044
GO TO 30

ISN 045
20 DELSAV = DELAER(1)

ISN 046
NSAV = ND(MODE)

ISN 047
WRITE (161,100) DELSAV, DELAER(1), NSAV, ND(MODE)

ISN 048
ND(MODE) = ND(MODE) + INC

ISN 049
IF (ND(MODE) .EQ. 1) ND(MODE) = MMIN

ISN 050
GO TO 30

ISN 051
30 NOD = ND(MODE)
ISH 0056 CALL NDT983 (NCD,NODE,INAVE,INC) 00055
ISH 0057 ND(MODE) = NODE * INAVE 00056
ISH 0058 CALL WRT983 (ICALC,MODE,NODE,INAVE) 00057
ISH 0059 CALL MHT983 00058
ISH 0060 IF (DELAER(1) .GE. DELSAV) GO TO 40 00059
ISH 0062 DELSAV = DELAER(1) 00060
ISH 0063 NSAV = ND(MODE) 00061
ISH 0064 WRITE (16I,100) DELSAV, DELAER(1), NSAV, ND(MODE) 00062
ISH 0065 GO TO 30 00063
ISH 0066 40 CONTINUE 00064
ISH 0067 WRITE (16I,100) DELSAV, DELAER(1), NSAV, ND(MODE) 00065
ISH 0068 ND(MODE) = NSAV 00066
ISH 0069 DLSV(MODE) = DELSAV 00067
ISH 0070 JMP(MODE) = 0 00068
ISH 0071 MODE = MODE + 1 00069
ISH 0072 IF (MODE .GT. LIH) GO TO 50 00070
ISH 0074 GO TO 10 00071
ISH 0075 50 RETURN 00072
ISH 0076 100 FORMAT (5X,E12.5,5X,E12.5,5X,I5,5X,I5) 00073
ISH 0077 110 FORMAT (/5X,'DELAERO-SAVE',6X,'DELAERO',9X,'ND-SAVE',4X,'ND') 00074
ISH 0078 END 00075

*OPTIONS IN EFFECT* NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NONMAP NOFORMAT NOGOST NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* NO DIAGNOSTICS GENERATED
**** END OF COMPIlATION ****

3004K BYTES OF CORE NOT USED
SUBROUTINE NDT983 (NOD, NODE, NAVE, INC)

THIS ROUTINE SELECTS A NODAL DIAMETER (NOD) BETWEEN -(NB/2-2) AND -2 AND BETWEEN 2 AND (NB/2-2) AND SETS NODE TO THE ABS(NOD) AND THE WAVE DIRECTION EQUAL TO THE SIGN OF NOD. THIS ROUTINE INCREASES NOD BY 1(INC) UNTIL DELTA AERO IN THE CALLING ROUTINE IS MINIMIZED.

COMMON /ANAL11/ MAY, NAX, ISTR, HBB, DEND, DENA, ED

1, DIAND, BLADES, POID

ISH 0001 COMMON /JMPCHK/ JMP(5), NODE

ISH 0002 NMIN = 2

ISH 0003 NMAX =((BLADES+.05)/2.) - 2.

ISH 0004 IF ( IABS(NOD) .GT. NMIN .AND. IABS(NOD) .LT. NMAX ) GO TO 10

ISH 0005 IF ( JMP(NODE) .NE. 0 ) GO TO 10

ISH 0006 JMP(MODE) = 1

ISH 0007 NOD = NOD *(-1)

ISH 0008 GO TO 20

ISH 0009 10 NOD = NOD + INC

ISH 0010 20 NAVE = 1

ISH 0011 IF ( NOD .LT. 0 ) NAVE = -1

ISH 0012 NODE = IABS(NOD)

ISH 0013 RETURN

ISH 0014 END
DATA SET U498SRD983 AT LEVEL 002 AS OF 06/04/81
DATA SET U500RDT983 AT LEVEL 001 AS OF 05/12/81

READ TITLE CARD
READ (IXI,900) TITL
READ CONTROL CARD
READ (IXI,900) TIAERO
READ AERO CASE TITLE
READ (IXI,900) TIAERO DATA
READ AERO DATA
DO 20 I = 1, MAERO
READ (IXI,902) RAERO(I), ANACHI(I), VEL1(I), TEMP1(I)
1, PRES(I), PSI1(I), BETA1(I)
20 CONTINUE
RETURN

*OPTIONS IN EFFECT=SOURCE,NOHAP,NOXREF,NOFORMAT,GOCTL,NOHDFORMAT,NOHDR,NOALC
*OPTIONS IN EFFECT=SOURCE, NOHAP, NOXREF, NOFORMAT, GOCTL, NOHDR, NOALC, NOANSF, TERM IBM, FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 18, PROGRAM SIZE = 558, SUBPROGRAM NAME = RDT983
*STATISTICS* NO DIAGNOSTICS GENERATED
**** END OF COMPILATION ****

3016K BYTES OF CORE NOT USED
1.0

ISN 0001

SUBROUTINE RCHI17 (ICALC)

COMMON /UGOS/ ISI, ISI

COMMON /ANAL01/ NSTE

COMMON /ANAL02/ ITLLE(16), IOP1, IOP2, IOP3

COMMON /ANAL03/ RPH, FN, DFN, TOLL, ROOT, DRPM, RPMM, RPMMX

COMMON /ANAL04/ NSTA, HST1, HST2, NOACR, ILT, LONG, ISTE

1. DENB, EB, POIB, BR, SERIES, BRB, BTS, IOPP

COMMON /ANAL05/ R(21), BCDR(21), THSTH(21), ALPH(21),

1. SOB(21), RLE(21), RRE(21), RFE(21), ADWT(21), OI(21)

COMMON /ANAL06/ TLTA(21), TLTT(21), XADWT(21), YADWT(21)

1. AKX(21), AKY(21), AKZ(21)

COMMON /ANAL07/ XP(3,53,21), HR(21)

COMMON /ANAL08/ THET, THER, AKMIN, AKMAX, ERAING

COMMON /ANAL09/ WTS1, XSHR1, YSHR1, ARSH1, XISH1, YISH1

1. AKSH1, SHPOI1, AKSH2, WSHL

COMMON /ANAL10/ WTS2, YSHR2, ARSH2, XISH2, YISH2

1. AKSH2, SHPOI2, AKSH3

COMMON /ANAL11/ MAY, NAX, ISTR, N3B, DENB, DENA, ED

1. DIAND, BLADES, POID

COMMON /ANAL12/ RAQD(32), THE (32)

COMMON /ANAL13/ SIGDOR, RAQDST(32), TANST(32)

COMMON /ANAL14/ ELP, HP, HBP, DBP, ISIP, DNP, EL, W

1. HS, DS, WZ, DT, RZ, THZ, TTH, OF

2. HT, EMU, DLX, RRT, BHP, THP, VR, RDR

COMMON /ANAL15/ SA, SB, SC, SD, SE, SF, SG, SH, SI, SJ

1. SK, SL, SH, REF, WI, T11, RZ, D2, D3, D4, D5, D6

2. DLR, T1, T2, T3, T4, RA, SETD

COMMON /ANAL16/ JXN(30), VAL(30,12), III

DIMENSION X(3,53)

1.4

READ FIRST FOUR CARDS OF THE W137 INPUT

CARD 1 --- NEW CASE CONTROL CARD

CARD 2 --- TITLE CARD AND RING AND RIM PROPERTIES OPTIONS

CARD 3 --- SPEED, FREQUENCIES, ROOT, ECT.

CARD 4 --- W137 CONTROL CARD

IXI = ISI

IF (ICALC .GT. 1) IXI = 24

IF (ICALC .EQ. 3) IXI = 5

IF (IXI .EQ. 24 .OR. IXI .EQ. 5) REWIND IXI

READ (IXI,900) NSTE

READ (IXI,901) ITLLE, IOP1, IOP2, IOP3

READ (IXI,902) RPH, FN, DFN, TOLL, ROOT, DRPM, RPMM, RPMMX

READ (IXI,903) NSTA, HST1, HST2, NOACR, ILT, LONG, ISTE, DENB, EB,

1. POIB, BR, SERIES, BRB, BTS, IOPP

DO 20 I = 1, NSTA

READ (IXI,904) R(1), BCDR(1), TCD(1), THSTH(1), ALPH(1), SGB(1),

1. RLE(1), RRE(1), ADWT(1), OI(1)

READ TILT AND SPRING CARD IF:

1. --- TILT IS INPUT AT EACH STATION (ILT = 1)
C 2--- TILT IS INPUT AT THE TIP STATION (ILT = 2) AND THE TIP STATION IS BEING READ (I = NO. OF STATION)
C 3--- ADDED WEIGHT HAS BEEN INPUT
C
ISN 0032 IF (ILT .EQ. 1) GO TO 5
ISN 0034 IF (ILT .EQ. 2) AND I .EQ. NSTA) GO TO 5
ISN 0036 IF (ADNT(I)) GT.5.5
ISN 0037 5 READ (IXI,905) TLTA(I), TLTT(I), XADNT(I), YADNT(I),
          1 AKX(I), AKY(I), AKZ(I)
C
ISN 0038 IF T/B IS POSITIVE --- STOP READING AIRFOIL DATA
C IF T/B IS NEGATIVE --- READ COORDINATES
C
ISN 0040 N = 0(I) + .05
ISN 0041 DO 10 JJ = 1, N
          10 CONTINUE
ISN 0042 READ (IXI,905) (X(JJ,IN), IN = 1,N)
ISN 0043 10 CONTINUE
ISN 0044 DO 15 M = 1, N
           15 CONTINUE
ISN 0045 XP(1,M) = X1,M)
ISN 0046 XP(2,M) = X2,M)
ISN 0047 XP(3,M) = X3,M)
ISN 0048 15 CONTINUE
ISN 0049 NPI(I) = N
ISN 0050 20 CONTINUE
C
ISN 0051 READ (IXI,906) THET, THER, AKMIN, AKMAX, BRANG
ISN 0052 READ SHROUD DATA (MST1 = 0, NO SHROUD)
C
ISN 0053 IF (MST1 .LT. 1) GO TO 30
ISN 0054 READ (IXI,907) WTS1, XSHR1, YSHR1, ARSH1, XISH1, XISH3,
          1 AKSH1, SHROI1, ANGS1, NSL
ISN 0055 IF (MST2 .LT. 2) GO TO 30
ISN 0056 READ (IXI,907) WTS2, XSHR2, YSHR2, ARSH2, XISH2, XISH4,
          1 AKSH2, SHROI2, ANGS2
C
ISN 0057 READ DISC CONTROL DATA
ISN 0058 30 CONTINUE
ISN 0059 READ (IXI,908) MAY, NAX, ISTR, NSB, DEND, DENA, ED
          1, DIAND, BLADES, POID
C
ISN 0060 IF (MAY .NE. 0) GO TO 40
ISN 0061 READ (IXI,909) (RAD(I),THE(I), I = 1,MAY)
ISN 0062 40 CONTINUE
C
ISN 0063 READ BORE STRESS, STRESS AT EACH STATION, OR NO STRESS
C
ISN 0064 IF (ISTR) 50,70,60
ISN 0065 50 READ (IXI,910) SIGBOR
ISN 0066 GO TO 70
ISN 0067 60 J = NAX + 1
ISN 0068 READ (IXI,909) (RADST(I), TANST(I), I = J, MAY)
ISN 0069 70 CONTINUE
C    READ DOVETAIL INPUT OR SKIP TO RING INPUT
C
    IF (I01 .EQ. 0) GO TO 80
C
    READ (IXI,902) EL,HP,HP0,DP,HPN,DPN,EL,W
C    READ (IXI,902) WS,DS,HZ,DT,RZ,THZ,TH,DF
C    READ (IXI,902) WT,ENU,DLX,RRT,ENP,TNP,HR
C    IF (HR .NE. 0 ) READ (IXI,910) RDR
C    80 CONTINUE
C
C    READ DISC RIM INPUT IOP2 = 1 OR 2
C
    IF (I0P2 - 1 ) 110,110,90
C
    90 READ (IXI,902) SA, SB, SC, SD, SE, SF, SG, SH
C    READ (IXI,902) SI, SJ, SK, SL, SH, REF, WI
C    IF ( WI ) 95,110,95
C    95 READ (IXI,902) T11, R2
C    GO TO 110
C
    100 READ (IXI,902) D2, D3, D4, D5, D6, DLR, WI
C    READ (IXI,902) T1, T2, T3, T4, RA, BETD
C    IF ( WI ) 105,110,105
C    105 READ (IXI,902) T1, R2
C    CONTINUE
C
C    READ SPRING AND RING PROPERTY INFORMATION
C
    I = 1
C
    120 READ (IXI,911) JXH(I) ,(VALI,L),L=1,12
C    IF ( JXH(I) .EQ. 0 ) GO TO 130
C    I = I + 1
C    GO TO 120
C
    130 I = I + 1
C    READ (IXI,911) JXH(I)
C
    I = I - 2
C    RETURN
C
    900 FORMAT(72X,I1)
C    901 FORMAT(16A4,1X,3I1)
C    902 FORMAT(6F8.0)
C    903 FORMAT(7I2,2X,7F8.0,1X,I1)
C    904 FORMAT(9F8.0,1X,F4.0)
C    905 FORMAT(9F8.0)
C    906 FORMAT(5E8.0)
C    907 FORMAT(9F8.0,1X,I1)
C    908 FORMAT(4I2,8X,6F8.0)
C    909 FORMAT(8F8.2)
C    910 FORMAT(8F8.2)
C    911 FORMAT(12,6(F3.0,F8.0))
C    END

*OPTIONS IN EFFECT=NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
*OPTIONS IN EFFECT=SOURCE EEDIC NODEC NOFORMAT GOSTMT NOXREF NOLAC NOSF TERM IBM FLAG(I)

*STATISTICS* SOURCE STATEMENTS = 110, PROGRAM SIZE = 3472, SUBPROGRAM NAME =RDW137
*STATISTICS* NO DIAGNOSTICS GENERATED

******** END OF COMPILATION ********

2988K BYTES OF CORE NOT USED
*VERSION 1.3.0 (01 MAY 80)  SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 82.141/10.52.54  PAGE 1
REQUESTED OPTIONS: SOURCE, NOSMAP, NOKREF, NOLIST, NODECK, OPT((3), AUTO DBLBL(NONE), NOALC
OPTIONS IN EFFECT: NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTO DBLBL(NONE)
SOURCE EBCDIC NOLIST NODECK OBJECT NOSMAP NOFORMAT GOSTH NOKREF NOALC NOSMF TERMIN IBM FLAG(I)

C DATA SET U498SRSFUN AT LEVEL 003 AS OF 01/28/82
C DATA SET U498SRSFUN AT LEVEL 002 AS OF 06/24/81
C DATA SET U500SRSFUN AT LEVEL 001 AS OF 06/22/81

ISN 0002 SUBROUTINE RSFUNC
ISN 0003 COMMON /USOS/ IS1, IS2
ISN 0004 COMMON /AHAL03/ RPM, FNS, DHF, TOLL, ROOT, DPHM, RFMIN, RFMAX
ISN 0005 COMMON /AHAL21/ FNRQ(100), SPEED(100), BTA(100)
ISN 0006 COMMON /AHAL32/ RFMIN, RFMAX, EORD(4), NORD
ISN 0007 COMMON /GLOBCL/ OBJF, FMIN(5), DLR(5), TKVAL(21), RF(5,4)
ISN 0008 DOUBLE PRECISION FNRQ, SPEED, BTA

C SAVE THE LARGER VALUE OF RFMAX AND RFMIN FOR EACH ORDER AND EACH ROOT

ISN 0009 NROOT = ROOT + .05
ISN 0010 WRITE (16I, 910) 00011
ISN 0011 910 FORMAT(5X, 'FREQUENCY', 8X, 'SPEED', 10X, 'BETA')
ISN 0012 DO 3 I = 1, NROOT
ISN 0013 WRITE (16I, 900) FNRQ(I), SPEED(I), BTA(I)
ISN 0014 3 CONTINUE
ISN 0015 900 FORMAT(5(3X, E12.5))
ISN 0016 WRITE (16I, 920) 00017
ISN 0017 920 FORMAT(6X, 'RPM-MAX', 6X, 'RPM-MAX', 9X, 'ND(1)', 10X, 'ND(2)', 10X, 'ND(3)', 10X, 'ND(4)')
ISN 0018 WRITE (16I, 900) FNRQ, RFMIN, RFMAX, EORD(1), EORD(2), EORD(3)
ISN 0019 WRITE (16I, 900) FNRQ, RFMIN, RFMAX, EORD(1), EORD(2), EORD(3)
ISN 0021 DO 10 I = 1, NROOT
ISN 0022 DO 5 J = 1, NORD
ISN 0023 DMAX = FNRQ(I) ** 2 + BTA(I) * (RFMAX ** 2 - SPEED(I) ** 2)
ISN 0024 DMIN = FNRQ(I) ** 2 + BTA(I) * (RFMIN ** 2 - SPEED(I) ** 2)
ISN 0025 RFMAX = 60. * SQRT(DMAX) / EORD(J) / RFMIN - 1.0
ISN 0026 RFMIN = 1.0 - 60. * SQRT(DMIN) / EORD(J) / RFMIN
ISN 0027 RF(I,J) = RFMAX
ISN 0028 IF (RFMIN .GT. RFMAX) RF(I,J) = RFMIN
ISN 0029 WRITE (16I, 900) DMAX, DMIN, RFMAX, RFMIN, RF(I,J)
ISN 0030 10 CONTINUE
ISN 0031 5 CONTINUE
ISN 0032 10 CONTINUE
ISN 0033 RETURN
ISN 0034 END

*OPTIONS IN EFFECT=NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTO DBLBL(NONE)
*OPTIONS IN EFFECT=SOURCE EBCDIC NOLIST NODECK OBJECT NOSMAP NOFORMAT GOSTH NOKREF NOALC NOSMF TERMIN IBM FLAG(I)
*STATISTICS** SOURCE STATEMENTS = 33, PROGRAM SIZE = 1048, SUBPROGRAM NAME =RSFUNC
*STATISTICS* NO DIAGNOSTICS GENERATED
***END OF COMPILED***

3012K BYTES OF CORE NOT USED
SUBROUTINE MAX (ICALC)

C DETERMINE MAX THICKNESS OF EACH INPUT STATION AND SAVE COORDINATES

DO 30 I = NBRS, NSTA
NPP = NP(I)

TMX = XP(2,1,I) - XP(3,1,I)

BMX(I) = XP(1,NPP,I) - XP(1,1,I)

DO 20 J = 1, NPP

T = XP(2,J,I) - XP(3,J,I)

THMAX = AMAX1 (THMAX, T)

30 CONTINUE

DO 90 I = NBRS, NSTA

NPP = NP(I)

DO 80 J = 1, NPP

DO 70 K = 1, 3

XPS(K,J,I) = XP(K,J,I)

70 CONTINUE

DO 90 I = NBRS, NSTA

NPP = NP(I)

DO 80 J = 1, NPP

DO 70 K = 1, 3

XPS(K,J,I) = XP(K,J,I)

80 CONTINUE

90 CONTINUE

C RATIO THE Y-UPPERS AND Y-LOKERS TO REFLECT THE NEW
THICKNESS FROM COPES-CONNIN

DO 90 I = NBRS, NSTA

NPP = NP(I)

DO 80 J = 1, NPP

DO 70 K = 1, 3

XPS(K,J,I) = XP(K,J,I) + (PER - 1) * .5 * T * XS

70 CONTINUE

90 CONTINUE

C RATIO THE Y-UPPERS AND Y-LOKERS TO REFLECT THE NEW
C CHORD FROM COPES-CONSIN
C
ISH 0046     DO 110 I = NBRJ,NSTA
ISH 0047     NPP = NP(I)
ISH 0048     IF (BCC(I) .EQ. 0.0 .OR. BMX(I) .EQ. 0.0) GO TO 110
ISH 0050     PER = BCC(I) / BMX(I)
ISH 0051     DO 100 J = 1, NPP
ISH 0052     XP(1,J,I) = XP(1,J,I) + (PER - 1.0) * XP(1,J,I)
ISH 0053     XP(2,J,I) = XP(2,J,I) + (PER - 1.0) * XP(2,J,I)
ISH 0054     XP(3,J,I) = XP(3,J,I) + (PER - 1.0) * XP(3,J,I)
ISH 0055     100 CONTINUE
ISH 0056     110 CONTINUE
ISH 0057     900 FORMAT (14,ICALC = ',I10)
ISH 0058     999 RETURN
ISH 0059     END

*OPTIONS IN EFFECT*MAIN OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
*OPTIONS IN EFFECT*SOURCE EBCDIC NOECK NODECK OBJECT NOEAP NOFORMAT GOSTHT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 58, PROGRAM SIZE = 1098, SUBPROGRAM NAME = TMAX
*STATISTICS* NO DIAGNOSTICS GENERATED
****** END OF COMPILATION *******

3004K BYTES OF CORE NOT USED
+VERSION 1.3.0 (01 MAY 80)  SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 82.141/10.53.00  PAGE 1
REQUESTED OPTIONS: SOURCE, NOLAP, NOXREF, NOLIST, NODECK, OPT(3), AUTODBL(NONE), NOALC
OPTIONS IN EFFECT: NAMO(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
SOURCE EBCDIC NOLIST NODECK OBJECT NOLAP NOFORMAT GOSTM370 XREF NOALC NOMSF TERM IBM FLAG(1)

C DATA SET U498SHT983 AT LEVEL 014 AS OF 01/29/82
C DATA SET U498SHT983 AT LEVEL 013 AS OF 12/02/81
C DATA SET U498SHT983 AT LEVEL 012 AS OF 09/14/81
C DATA SET U500SHT983 AT LEVEL 011 AS OF 09/09/81
C DATA SET U500SHT983 AT LEVEL 010 AS OF 06/09/81

ISN 0002
SUBROUTINE NTY83 (ICALC, NODE, INO, IWAWE)

ISN 0003
COMMON/USIS/ ISI, ICI

ISN 0004
COMMON/ANALOG/ NSTA, MSTI, MSTD, NOCAR, INT, LONG, ISTE,
1, DEHI, EB, POID, BR, SERIES, BRS, BTS, IOPP

ISN 0005
COMMON /ANAL11/ NAX, NSTR, NBB, DEHI, DEHA, ED
1, DIAMD, BLADES, POID

ISN 0006
COMMON /ANAL17/ XPS(3,53,21), TMX(21), TCC(21), BMX(21)

ISN 0007
1, BCC(21)

COMMON/ANAL20/ SPEED(5), AND(5), FRFN(5), TKEB(5)
1, RAD(5,21), CHD(5,21), SCDT(5,21)

ISN 0008
COMMON/ANAL5/ TITL(20)

ISN 0009
COMMON/ANAL51/ MACH, MAER0, NUMA, M1137, INO, IBETA, IGUST

ISN 010
COMMON/ANAL52/ TIAERO(20)

ISN 0011
COMMON/ANAL5S/ RAERO(21), AMACHI(21), VELI(21), TEHP(21)

ISN 012
COMMON /ANAL5/ NSFA, ISTH(21)

ISN 013
DOUBLE PRECISION SPEED, AND, FRFN, TKEB
1, RAD, CHD, SCDT, ALPH, PFD, PFAG, PFD, PHAG

IF (ICALC.EQ. 2) REWIND 5

ISN 016
NBR = BRS + .05

ISN 017
IYI = I61

ISN 018
IF (ICALC .EQ. 2) IYI = 5

ISN 020
WRITE TITLE CARD - - CARD 1

ISN 021
WRITE (IYI,900) TITL

ISN 022
WRITE CONTROL CARD - - CARD 2

ISN 023
NUMA = 1

ISN 024
NUMA = 1

ISN 025
NUMA = 1

ISN 026
NUMA = 1

ISN 027
NUMA = 1

ISN 028
IPARM = 0

ISN 029
IPARM = 0

IF (ICALC .EQ. 2) IYI = 5

ISN 030
WRITE (IYI,901) MACH, MAERO, NUMA, M1137, IWAWE, NIN, MN
1, IBETA, IGUST, KGUST, NGUST, IPARM, IPARM

ISN 031
WRITE (IYI,900) TIAERO

ISN 032
WRITE AERO CASE TITLE - - CARD 3

ISN 033
WRITE (IYI,900) TIAERO

ISN 034
WRITE AERO DATA - - CARD 4

ISN 035
DO 10 I = 1, MAERO
1, PRES(I), PSI(I), BETAI(I)
10 CONTINUE

WRITE (1, 902) RAERO(I), AMACH(I), VEL(I), TEMP(I)

WRITE (1, 904) RADI(I), CHDC(I), SCTI(I), ALPH(I), PFDF(I), PFAG(I), PWDFN(I)

IF (IYYI .EQ. 5) REWIND 5

RETURN

END
DATA SET U98SHLT17 AT LEVEL 005 AS OF 03/11/82
DATA SET U98SHLT17 AT LEVEL 004 AS OF 01/28/82
DATA SET U98SHLT17 AT LEVEL 003 AS OF 12/02/81
DATA SET U98SHLT17 AT LEVEL 002 AS OF 05/28/81
DATA SET U98SHLT17 AT LEVEL 001 AS OF 04/06/81

SUBROUTINE WHT137 (ICALC)

COMMON /UOS/ ISI, ISI
COMMON /ANAL0/ NTST, NTST
COMMON /ANAL02/ ITTLT(16), IPO1, IPO2, IPO3
COMMON /ANAL03/ RPM, FN, DFN, TOLL, ROOT, DRPN, RPMN, RPMX
COMMON /ANAL04/ NSTA, HST1, HST2, NOACR, ILT, LONG, ISTE
COMMON /ANAL05/ R(21), BSR(21), TSTH(21), ALPHA(21), USC211
COMMON /ANAL06/ TLT(21), TLT(21), XAYDHT(21), YADHT(21)
COMMON /ANAL07/ AKXI, SHP02, ANGS2
COMMON /ANAL11/ MAY, NAX, ISTR, NDB, DND, DENA, ED
COMMON /ANAL12/ RADI(32), THE(32)
COMMON /ANAL13/ SIGOR, RADST(32), TANST(32)
COMMON /ANAL14/ ELP, WP, WBP, DAP, WKN, DINP, EL, W
COMMON /ANAL15/ HRT(21), SHR1, YSHR1, ARSH1, XISH1, XISH2, XISH3
COMMON /ANAL16/ WTS2, XSRH2, YSRH2, ARSH2, XISH2, XISH3
COMMON /ANAL17/ XPS, XPS, XPS
COMMON /ANAL18/ XPS, XPS, XPS
COMMON /ANAL19/ XJS(30.132), VAL(30.12), III
COMMON /ANAL20/ XPS, XPS, XPS

WRITE FIRST FOUR CARDS OF THE W137 INPUT
WRITE NEW CASE CONTROL CARD
WRITE TITLE CARD AND RHM AND RHM PROPERTIES OPTIONS
WRITE SPEED, FREQUENCIES, ROOT, ECT.
WRITE W137 CONTROL CARD

IYI = I61
IF (ICALC .EQ. 1) IYI = 24
IF (ICALC .EQ. 2) IYI = 24
IF (ICALC .EQ. 4) IYI = 7
IF (IYI .EQ. 5 .OR. IYI .EQ. 24) REWIND IYI
WRITE (IYI, 900) NTEST
WRITE (IYI, 901) ITTLT, IPO1, IPO2, IPO3
WRITE (IYI, 902) RPM, FN, DFN, TOLL, ROOT, DRPN, RPMN, RPMX
WRITE (IYI, 903) HST1, HST2, NOACR, ILT, LONG, ISTE
DO 20 I = 1, NSTA
WRITE AIRFOIL DATA
WRITE AIRFOIL DATA
C WRITE TILT AND SPACING CARD IF:
C 1 --- TILT IS INPUT AT EACH STATION (ILT = 1)
C 2 --- TILT IS INPUT AT THE TIP STATION (ILT = 2) AND THE
C TIP STATION IS BEING READ (I = NO. OF STATIONS)
C 3 --- ADDED WEIGHT HAS BEEN INPUT
C
C IF T/B IS POSITIVE --- STOP READING AIRFOIL DATA
C IF T/B IS NEGATIVE --- READ COORDINATES
C
C IF (TOB(I) .GT. 0.0) GO TO 20
DO 10 J = 1 , 3
WRITE (IYI,906) (XP(J,IH,I) , IN = 1 , H)
10 CONTINUE
20 CONTINUE
C WRITE DISC CONTROL DATA
C
C WRITE DISC RADII AND THICKNESSES IF MAY .NE. 0
C
C IF (MAY .EQ. 0) GO TO 50
WRITE (IYI,911) SIGBOR
50 CONTINUE
IStl 0069 GO TO 70 00114
IStl 0070 60 J = MAX + 1
IStl 0071 WRITE (IYI,911) (RADI(I), TANST(I), I = J, MAX)
IStl 0072 70 CONTINUE

C WRITE DOVE TAIL INPUT OR SKIP TO RING INPUT
C IF (IOP1 .EQ. 0) GO TO 80
IStl 0073 WRITE (IYI,910) ELP, WP, WBP, DBP, WNP, ONP, WR
IStl 0074 IF (HR .NE. 0.) WRITE (IYI,910) RUN
IStl 0075 60 CONTINUE

C WRITE DISC RIM INPUT IF IOP2 = 1 OR 2
C IF (IOP2 .LT. 1) 110,100,90
IStl 0081 IF (IOP2 .EQ. 1) 110,100,90
IStl 0082 100 WRITE (IYI,910) T11, R2
IStl 0083 110 CONTINUE

C WRITE SPRING AND RING PROPERTY INFORMATION
IStl 0092 DO 120 I = 1 , III
IStl 0093 WRITE (IYI,912) JXN(I), (VAL(I,L), L = 1 , 12)
IStl 0094 120 CONTINUE
IStl 0095 IXN = 0
IStl 0096 WRITE (IYI,913) IXN
IStl 0097 IXN = 3
IStl 0098 IF (IYI .EQ. 5) REWIND IYI
IStl 0100 RETURN
IStl 0101 900 FORMAT (72X,I1)
IStl 0102 901 FORMAT (18A4,1X,3I1)
IStl 0103 902 FORMAT (3F8.1,F8.0,4F8.1)
IStl 0104 903 FORMAT (72X,2F8.5,E8.5,3F8.1,1X,I1)
IStl 0105 904 FORMAT (3F8.5,2F8.4,4F8.5,1X,F4.0)
IStl 0106 905 FORMAT (4F8.5,3E8.1)
IStl 0107 906 FORMAT (9F8.5)
IStl 0108 907 FORMAT (2F8.4,2F8.0,F8.4)
IStl 0109 908 FORMAT (3F8.5,F8.6,4F8.7,F8.4,4X,I1)
IStl 0110 909 FORMAT (4I2,8X,2F8.5,E8.3,3F8.4)
IStl 0111 910 FORMAT (8F8.5)
IStl 0112 911 FORMAT (6F8.1)
IStl 0113 912 FORMAT (I2,F3.0,E8.3,5(F3.0,F8.5))
IStl 0114 913 FORMAT (7X,I1)
IStl 0115 FORMAT

*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOBLKLINE
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NOMAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 114, PROGRAM SIZE = 2984, SUBPROGRAM NAME = WTH137
*STATISTICS* NO DIAGNOSTICS GENERATED

*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOBLKLINE
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NOMAP NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 114, PROGRAM SIZE = 2984, SUBPROGRAM NAME = WTH137
*STATISTICS* NO DIAGNOSTICS GENERATED

EN D OF COMPI LATION ** ** ** ** 2984 K BYTES OF CORE NOT USED
REQUESTED OPTIONS: SOURCE,NOMAP,NOLIST,NODECK,OPT(3),AUTODBL(NONE),NOALC

OPTIONS IN EFFECT: NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)

SOURCE EBCDIC NOLIST NODECK OBJECT NOHAP NOFORMAT COSTHT NOKREF NOALC NOANSF TERM IBM FLAG(I)

DATA SET U498SOBJTV AT LEVEL 007 AS OF 03/16/82
DATA SET U498SOBJTV AT LEVEL 006 AS OF 03/03/82
DATA SET U498SOBJTV AT LEVEL 005 AS OF 02/03/82
DATA SET U498SOBJTV AT LEVEL 004 AS OF 12/02/81
DATA SET U498SOBJTV AT LEVEL 003 AS OF 11/23/81
DATA SET U498SOBJTV AT LEVEL 002 AS OF 11/20/81
DATA SET U498SOBJTV AT LEVEL 001 AS OF 11/19/81

ISN 0002 SUBROUTINE OBJTV

ISN 0003 COMMON /GLOBCM/ OBJF, FN(5), DLAR(5), THKVAL(21), RF(5,4)
1, ERCC, FODLSB(21), BLC, DL, DUE, DROOT, BTIP, TTI, TLT
2, OBJFUN

ISN 0004 COMMON /PLY1/ PLY(21, 25, 7), THEATA(7), RHO(7)

ISN 0005 COMMON /WEIGHT/ HL(7)

ISN 0006 COMMON /ANAL3/ DLED, DTED, DROOT, DTIPD, TTID, TLTD, NCD

ISN 0007 COMMON /ANAL7/ XPS(3, 35, 21), TIX(21), TCC(21), BHX(21)
1, ECC(21)

ISN 0008 COMMON /UOS/ I5I, I6I

ISN 0009 COMMON /ANAL04/ HSTA, MSTA, MST2, NOACR, ILT, LONG, ISTE,
1 DENS, EB, POID, BR, SERIES, ERS, BTS, IOPP

ISN 0010 COMMON /ANAL05/ R(21), BOBR(21), TOS(21), THBA(21), ALPHA(21),
1 SOD(21), RLE(21), RTE(21), ADIF(21), O(21)

ISN 0011 DOUBLE PRECISION PLY, THEATA, RHO, WL, DLED, DTED, DROOTD
1, DTIPD, TTID, TLTD

ISN 0012 RC = BRCC

ISN 0013 IF (BRCC EQ. 0.0) RC = BHX(1)

ISN 0015 RT = THKVAL(1)

ISN 0016 IF (NCD .EQ. 2) GO TO 10

ISN 0018 TIVOL = (HL(1) + HL(7)) / RHO(1)

ISN 0019 BTIVOL = (HL(2) & HL(6)) / RHO(2)

ISN 0020 EMC = 2000. * (.000958 * TIVOL + 1.183 * (BTIVOL/116.6)**1.08 )

ISN 0021 ELC = 2000.*.273 * (RC / 9.25)**.35 + .95 * (RT / 9.25)**.86

ISN 0022 EDC = 264.42 * (EMC + ELC) / RC +3700. + RC

ISN 0023 WITE (161, 500) RC, RT, TIVOL, BTIVOL, EMC, ELC, EDC

ISN 0024 906 FORMAT (7(3X,E12.5/))

ISN 0025 TR = R(HSTA)

ISN 0026 NBRS = BRS + .05

ISN 0007 NCD = 1 (HOLLOW FOIL)

ISN 0008 NCD = 2 (SUPERHYBRID FOIL)

ISN 0010 TIVOL = (HL(1) + HL(7)) / RHO(1)

ISN 0019 BTIVOL = (HL(2) & HL(6)) / RHO(2)

ISN 0020 EMC = 2000. * (.000958 * TIVOL + 1.183 * (BTIVOL/116.6)**1.08 )

ISN 0021 ELC = 2000.*.273 * (RC / 9.25)**.35 + .95 * (RT / 9.25)**.86

ISN 0022 EDC = 264.42 * (EMC + ELC) / RC +3700. + RC

ISN 0023 WITE (161, 500) RC, RT, TIVOL, BTIVOL, EMC, ELC, EDC

ISN 0024 906 FORMAT (7(3X,E12.5/))

ISN 0025 TR = R(HSTA)

ISN 0026 NBRS = BRS + .05
ISN 0027  RR = R(NDRS)                         00055
ISN 0026  ARATIO = (TR*TR - (DROOT - 2.)*2) / (TR*TR - RR*RR) 00056
ISN 0029  IF ((DROOT - 2.) .LT. RR) ARATIO = 1.0 00057
ISN 0031  EDLE = DLE 00058
ISN 0032  IF ( (DLE - 0.001) .LT. 0.) EDLE = 20.0 00029
ISN 0034  EDMMC = (EMC + ELC) * (4.0 + 6.6 * ARATIO/EDLE) / 220.0
          1 / 25000.0 / RC 00060
          00061
          00062
          00063
          00064
          00065
          00066

ISN 0035  WRITE (I6I,910) NSTA , NDRS,TR, RR, DROOT, DLE, ARATIO, EDMMC 00067
ISN 0036  910 FORMAT (215,6(3X,E12.5)/) 00068
ISN 0037  FW = (WL(1) + WL(7) + WL(2) + WL(6) ) / 396.4 00069
ISN 0038  DIWT = 99.5 * FW - 1.233 * FW*2 00070
ISN 0039  OBJFUN = .54 * EDC/100000. + .50 * EDMMC/10. + .52 * DIWT/1000. 00071
          00072
          00073
          00074
          00075
ISN 0040  WRITE (I6I,900) WL(1),WL(2),WL(6),WL(7),FW,DW,OBJFUN 00076
ISN 0041  GO TO 50 00078
ISN 0042  10 CONTINUE 00079
          00080
          00081
          00082
          00083
ISN 0043  TIVOL = VOLUME OF TITANIUM 00084
ISN 0044  BAVOL = VOLUME OF BORON ALUMINUM 00085
ISN 0045  GEVOL = VOLUME OF GRAPHITE EPOXY 00086
          00087
          00088
          00089
ISN 0046  EMC = MATERIAL COST 00090
ISN 0047  ELC = LABOR COST 00091
ISN 0048  EDC = DELTA COST RELATIVE TO E3 00092
ISN 0049  EDMMC = DELTA MAINTENANCE AND MATERIAL COST 00093
          00094
          00095
          00096
ISN 0046  EMC = 2000.0 * (.483 * (GEVOL / 116.6)**1.05
          1 + .637 * (. (BAVOL + TIVOL ) / 116.6)**1.08
          2 + .1445 * (. (RC / 9.25)**1.774) 00097
ISN 0047  ELC = 2000.0 * (.063 * (RC / 9.25)**1.15
          1 + .121 * (RT / .925)**.62) 00098
ISN 0048  EDC = 246.42 * (EMC + ELC) / RC + 3700.0 * RC
ISN 0049  EDMMC = 222.0 * (EMC + ELC) / (12500.0 * RC) 00099
          00100
          00101
          00102
                FW = FOIL WEIGHT 00103
                DIWT = DELTA WEIGHT RELATIVE TO E3 00104
                OBJFUN = OBJECTIVE FUNCTION 00105
          00106
                FW = (WL(1) +WL(2) +WL(3) +WL(4) +WL(5) +WL(6) +WL(7)) *386.4
ISN 0050  DIWT = 99.5 * FW - 1.233 * FW*2 00107
ISN 0051  OBJFUN = .54 * EDC/100000. + .60 * EDMMC/10. + .52 * DIWT/1000. 00108
ISN 0052  WRITE (I6I,900) RC, RT, TIVOL, BAVOL, GEVOL 00109
ISN 0053  WRITE (I6I,900) (WL(I), I=1,7) 00110
ISN 0054  WRITE (I6I,900) (E3(I), I=1,7) 00111
ISN 0055  WRITE (I6I,900) (RHO(I), I=1,7) 00112
ISN 0056  WRITE (I6I,900) EMC, ELC, EDC, EDMMC, FW, DIWT, OBJFUN 00113
ISN 0057  50 CONTINUE
+VERSION 1.3.0 (01 MAY 80) OBJTV SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 82.141/10.53.08 PAGE 3

ISH 0058 RETURN 00114
ISH 0059 END 00115

*OPTIONS IN EFFECT* NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOCBL(NONE)
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECX OBJECT NOXAP NOFORMAT GOSTIT NOXREF NDALC NDANSF TERM IEM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 59, PROGRAM SIZE = 1994, SUBPROGRAM NAME = OBJTV
*STATISTICS* NO DIAGNOSTICS GENERATED

****** END OF COMPILATION ******

3000K BYTES OF CORE NOT USED
C
DATA SET U477CHNN1 AT LEVEL 001 AS OF 02/13/81
C
DATA SET 9186CHNN1 AT LEVEL 001 AS OF 07/10/80

ISN 0002 SUBROUTINE CNN001 (JGOTO,X,D,F,G,ISC,IC,A,G1,VLB,VUB,SCAL,C,NCAL,DX) 00001
1,DLX,FX,III,HI,NI,NI,H(N4)
00003

ISN 0003 COMMON /CHRN1/ DELFUN,DABFUN,FCH,FCHM,CT,CTMN,CTL,CTLMN,ALPHAX..00004
1,ABCDJ,THETA,OBJ,NB,NCON,NSIDE,INFO,H,FDG,SCAL,LINOBJ,THINX,ITON
2,CHDIR,JGOTO,NAC,INFO,INFOG,ITER
00006

ISN 0004 DIMENSION X(N1),DF(N1),G(N2),ISC(N2),IC(N3),A(N1,N4),G1(N2),G00007
1,VLB(N1),VUB(N1),SCAL(N1),NCAL(2),C(N4)

C ROUTINE TO CALCULATE GRADIENT INFORMATION BY FINITE DIFFERENCE. 00009
C BY G. H. VANDERPLOAT JULY, 1972. 00010
C NASA-AMES RESEARCH CENTER, MOUNT FULFORD FIELD, CALIF. 00011

ISN 0005 IF (JGOTO.EQ.1) GO TO 10 00012
ISN 0007 IF (JGOTO.EQ.1) GO TO 70 00013
ISN 0009 INFO=0 00014
ISN 0010 INFO=INFO 00015
ISN 0011 NAC=0 00016
ISN 0012 IF (LINOBJ.NE.0 .AND. ITER.GT.1) GO TO 10 00017
CGRADIENT OF LINEAR OBJECTIVE 00018
C
ISN 0014 IF (FDG.EQ.2) JGOTO=1 00019
ISN 0016 IF (FDG.EQ.2) RETURN 00020
ISN 0018 10 CONTINUE 00021
ISN 0019 JGOTO=0 00022
ISN 0020 IF (FDG.EQ.2 .AND. NCON.EQ.0) RETURN 00023
ISN 0022 IF (NCON.EQ.0) GO TO 40 00024
C ** DETERMINE WHICH CONSTRAINTS ARE ACTIVE OR VIOLATED ** 00025
C
ISN 0024 DO 20 I=1,NCON 00026
ISN 0025 IF (G(I).LT.CT) GO TO 20 00027
ISN 0027 IF (ISC(I).GT.0 .AND. G(I).LT.CT) GO TO 20 00028
ISN 0029 NAC=NAC+1 00029
ISN 0030 IF (NAC.GE.N3) RETURN 00030
ISN 0032 IC(NAC)=I 00031
ISN 0033 20 CONTINUE 00032
ISN 0034 IF (FDG.EQ.2 .AND. NAC.EQ.0) RETURN 00033
ISN 0036 IF ((LINOBJ.GT.0 .AND. ITER.GT.1).AND. NAC.EQ.0) RETURN 00034
CSTORE VALUES OF CONSTRAINTS IN G1 00035
C
ISN 0038 DO 30 I=1,NCON 00036
ISN 0039 30 G(I)=G(I) 00037
ISN 0040 CONTINUE 00038
C IF (NAC.EQ.0 .AND. NFDG.EQ.2) RETURN 00039
C
ISN 0041 JGOTO=0 00040
ISN 0042 IF (NAC.EQ.0 .AND. NFDG.EQ.2) RETURN 00041
C
ISN 0044 INFOG=1 00042
ISN 0045 INFO=1 00043
ISN 0046 IF=0 00044
ISN 0047 50 IF=IF+1 00045
ISN 0048 50 IF=IF+1 00046

C CALCULATE GRADIENTS 00047
C

C FUNCTION EVALUATION
C
ISH 0065 JGOTO=2 00066
ISH 0066 RETURN 00067
ISH 0067 70 CONTINUE 00068
ISH 0068 X(III)=XI 00069
ISH 0069 IF (NFDG.EQ.O) DF(III)=DX1*(OBJ-FI) 00070
ISH 0071 IF (NAC.EQ.O) GO TO 90 00071
C DETERMINE GRADIENT COMPONENTS OF ACTIVE CONSTRAINTS 00072
C
ISH 0073 DO 80 J=1,NAC 00073
ISH 0074 IL=IC(J) 00074
ISH 0075 80 A(III,J)=DX1*(G(I))-G(I)) 00075
ISH 0076 90 CONTINUE 00076
ISH 0077 IF (III.LT.NDY) GO TO 50 00077
ISH 0078 INFOS=0 00078
ISH 0079 INFO=INF 00079
ISH 0080 JGOTO=0 00080
ISH 0081 OBJ=FI 00081
ISH 0082 IF (NCHQ.EQ.O) RETURN 00082
C STORE CURRENT CONSTRAINT VALUES BACK IN G-VECTOR 00083
C
ISH 0085 DO 100 I=1,NCON 00084
ISH 0086 100 G(I)=G(I) 00085
ISH 0087 RETURN 00086
ISH 0088 ENO 00087
endants of active constraints

C DETERMINE GRADIENT COMPONENTS OF ACTIVE CONSTRAINTS
C
ISH 0073 DO 80 J=1,NAC
ISH 0074 IL=IC(J)
ISH 0075 80 A(III,J)=DX1*(G(I))-G(I))
ISH 0076 90 CONTINUE
ISH 0077 IF (III.LT.NDY) GO TO 50
ISH 0078 INFOS=0
ISH 0079 INFO=INF
ISH 0080 JGOTO=0
ISH 0081 OBJ=FI
ISH 0082 IF (NCHQ.EQ.O) RETURN
C STORE CURRENT CONSTRAINT VALUES BACK IN G-VECTOR
C
ISH 0085 DO 100 I=1,NCON
ISH 0086 100 G(I)=G(I)
ISH 0087 RETURN
ISH 0088 ENO

*OPTIONS IN EFFECT=NAME(HAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOABB(NONE)
*OPTIONS IN EFFECT=SOURCE EBCDIC NOLIST NODIR OBJECT NOIMAP NOFORMAT GOSTIT NOREF NOALC NOANS NTERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 87, PROGRAM SIZE = 1730, SUBPROGRAM NAME =CN::NOI
*STATISTICS* NO DIAGNOSTICS GENERATED
****** END OF COMPIATION ******

3004K BYTES OF CORE NOT USED
DATA SET U477CHN02 AT LEVEL 001 AS OF 02/13/81
DATA SET 9160CHN02 AT LEVEL 001 AS OF 07/10/80

SUBROUTINE CMM02 (NCALC,SLOPE,DFTDFI,DF,S,N)

COMMON /CHN/ DFLFUN,DFBDFUN,DFCH,DFCHM,CT,CTMIN,ALPHACK,ABOBJ1,THETA,OSJ,TIDV,NCON,TSIDE,IPHT,NFDG,LINCAL,IT0

C ROUTINE TO DETERMINE CONJUGATE DIRECTION VECTOR OR DIRECTION
OF STEEPEST DESCENT FOR UNCONSTRAINED FUNCTION MINIMIZATION.
BY G. N. VANDERPLAATS  APRIL, 1972.
NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.
NCALC = CALCULATION CONTROL.
NCALC = 0, S = STEEPEST DESCENT.
NCALC = 1, S = CONJUGATE DIRECTION.
CONJUGATE DIRECTION IS FOUND BY FLETCHER-REEVES ALGORITHM.
------------------------------------------------------------------
CALCULATE NORM OF GRADIENT VECTOR
------------------------------------------------------------------
DFTDF=0.
DO 10 I=1,NDV
DFI=DF(I)
DFTDF=DFTDF+DFI*DFI
------------------------------------------------------------------
********** FIND DIRECTION S ****************
------------------------------------------------------------------
IF (NCALC.NE.1) GO TO 30
IF (DFTDF1.LT.1.0E-20) GO TO 30
BETA=DFTDF/OFTDF1
SLOPE=0.
DO 20 I=1,NOV
DFI=DF(I)
SI=BETA*S(I)-DFI
SLOPE=SLOPE+SI*DFI
S(I)=SI
GO TO 50
CONTINUE
NCALC=0
------------------------------------------------------------------
CALCULATE DIRECTION OF STEEPEST DESCENT
------------------------------------------------------------------
DO 40 I=1,NDV
S(I)=-DF(I)
SLOPE=-DFTDF
CONTINUE
------------------------------------------------------------------
NORMALIZE S TO MAX ABS VALUE OF UNITY
------------------------------------------------------------------
S1=0.
DO 60 I=1,NDV
S2=ABS(S(I))
IF (S2.GT.S1) S1=S2
CONTINUE

C ROUTINE TO DETERMINE CONJUGATE DIRECTION VECTOR OR DIRECTION
OF STEEPEST DESCENT FOR UNCONSTRAINED FUNCTION MINIMIZATION.
BY G. N. VANDERPLAATS  APRIL, 1972.
NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.
NCALC = CALCULATION CONTROL.
NCALC = 0, S = STEEPEST DESCENT.
NCALC = 1, S = CONJUGATE DIRECTION.
CONJUGATE DIRECTION IS FOUND BY FLETCHER-REEVES ALGORITHM.
------------------------------------------------------------------
CALCULATE NORM OF GRADIENT VECTOR
------------------------------------------------------------------
DFTDF=0.
DO 10 I=1,NDV
DFI=DF(I)
DFTDF=DFTDF+DFI*DFI
------------------------------------------------------------------
********** FIND DIRECTION S ****************
------------------------------------------------------------------
IF (NCALC.NE.1) GO TO 30
IF (DFTDF1.LT.1.0E-20) GO TO 30
BETA=DFTDF/OFTDF1
SLOPE=0.
DO 20 I=1,NOV
DFI=DF(I)
SI=BETA*S(I)-DFI
SLOPE=SLOPE+SI*DFI
S(I)=SI
GO TO 50
CONTINUE
NCALC=0
------------------------------------------------------------------
CALCULATE DIRECTION OF STEEPEST DESCENT
------------------------------------------------------------------
DO 40 I=1,NDV
S(I)=-DF(I)
SLOPE=-DFTDF
CONTINUE
------------------------------------------------------------------
NORMALIZE S TO MAX ABS VALUE OF UNITY
------------------------------------------------------------------
S1=0.
DO 60 I=1,NDV
S2=ABS(S(I))
IF (S2.GT.S1) S1=S2
CONTINUE

C ROUTINE TO DETERMINE CONJUGATE DIRECTION VECTOR OR DIRECTION
OF STEEPEST DESCENT FOR UNCONSTRAINED FUNCTION MINIMIZATION.
BY G. N. VANDERPLAATS  APRIL, 1972.
NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.
NCALC = CALCULATION CONTROL.
NCALC = 0, S = STEEPEST DESCENT.
NCALC = 1, S = CONJUGATE DIRECTION.
CONJUGATE DIRECTION IS FOUND BY FLETCHER-REEVES ALGORITHM.
------------------------------------------------------------------
CALCULATE NORM OF GRADIENT VECTOR
------------------------------------------------------------------
DFTDF=0.
DO 10 I=1,NDV
DFI=DF(I)
DFTDF=DFTDF+DFI*DFI
------------------------------------------------------------------
********** FIND DIRECTION S ****************
------------------------------------------------------------------
IF (NCALC.NE.1) GO TO 30
IF (DFTDF1.LT.1.0E-20) GO TO 30
BETA=DFTDF/OFTDF1
SLOPE=0.
DO 20 I=1,NOV
DFI=DF(I)
SI=BETA*S(I)-DFI
SLOPE=SLOPE+SI*DFI
S(I)=SI
GO TO 50
CONTINUE
NCALC=0
------------------------------------------------------------------
CALCULATE DIRECTION OF STEEPEST DESCENT
------------------------------------------------------------------
DO 40 I=1,NDV
S(I)=-DF(I)
SLOPE=-DFTDF
CONTINUE
------------------------------------------------------------------
NORMALIZE S TO MAX ABS VALUE OF UNITY
------------------------------------------------------------------
S1=0.
DO 60 I=1,NDV
S2=ABS(S(I))
IF (S2.GT.S1) S1=S2
CONTINUE

C ROUTINE TO DETERMINE CONJUGATE DIRECTION VECTOR OR DIRECTION
OF STEEPEST DESCENT FOR UNCONSTRAINED FUNCTION MINIMIZATION.
BY G. N. VANDERPLAATS  APRIL, 1972.
NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.
NCALC = CALCULATION CONTROL.
NCALC = 0, S = STEEPEST DESCENT.
NCALC = 1, S = CONJUGATE DIRECTION.
CONJUGATE DIRECTION IS FOUND BY FLETCHER-REEVES ALGORITHM.
------------------------------------------------------------------
CALCULATE NORM OF GRADIENT VECTOR
------------------------------------------------------------------
DFTDF=0.
DO 10 I=1,NDV
DFI=DF(I)
DFTDF=DFTDF+DFI*DFI
------------------------------------------------------------------
********** FIND DIRECTION S ****************
------------------------------------------------------------------
IF (NCALC.NE.1) GO TO 30
IF (DFTDF1.LT.1.0E-20) GO TO 30
BETA=DFTDF/OFTDF1
SLOPE=0.
DO 20 I=1,NOV
DFI=DF(I)
SI=BETA*S(I)-DFI
SLOPE=SLOPE+SI*DFI
S(I)=SI
GO TO 50
CONTINUE
NCALC=0
------------------------------------------------------------------
CALCULATE DIRECTION OF STEEPEST DESCENT
------------------------------------------------------------------
DO 40 I=1,NDV
S(I)=-DF(I)
SLOPE=-DFTDF
CONTINUE
------------------------------------------------------------------
NORMALIZE S TO MAX ABS VALUE OF UNITY
------------------------------------------------------------------
S1=0.
DO 60 I=1,NDV
S2=ABS(S(I))
IF (S2.GT.S1) S1=S2
CONTINUE

C ROUTINE TO DETERMINE CONJUGATE DIRECTION VECTOR OR DIRECTION
OF STEEPEST DESCENT FOR UNCONSTRAINED FUNCTION MINIMIZATION.
BY G. N. VANDERPLAATS  APRIL, 1972.
NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.
NCALC = CALCULATION CONTROL.
NCALC = 0, S = STEEPEST DESCENT.
NCALC = 1, S = CONJUGATE DIRECTION.
CONJUGATE DIRECTION IS FOUND BY FLETCHER-REEVES ALGORITHM.
------------------------------------------------------------------
CALCULATE NORM OF GRADIENT VECTOR
------------------------------------------------------------------
DFTDF=0.
DO 10 I=1,NDV
DFI=DF(I)
DFTDF=DFTDF+DFI*DFI
------------------------------------------------------------------
********** FIND DIRECTION S ****************
------------------------------------------------------------------
IF (NCALC.NE.1) GO TO 30
IF (DFTDF1.LT.1.0E-20) GO TO 30
BETA=DFTDF/OFTDF1
SLOPE=0.
DO 20 I=1,NOV
DFI=DF(I)
SI=BETA*S(I)-DFI
SLOPE=SLOPE+SI*DFI
S(I)=SI
GO TO 50
CONTINUE
NCALC=0
------------------------------------------------------------------
CALCULATE DIRECTION OF STEEPEST DESCENT
------------------------------------------------------------------
DO 40 I=1,NDV
S(I)=-DF(I)
SLOPE=-DFTDF
CONTINUE
------------------------------------------------------------------
NORMALIZE S TO MAX ABS VALUE OF UNITY
------------------------------------------------------------------
S1=0.
DO 60 I=1,NDV
S2=ABS(S(I))
IF (S2.GT.S1) S1=S2
CONTINUE
IF (S1.LT.1.0E-20) S1=1.0E-20

S1=1./S1

DFTDF=DFTDF*S1

DO 70 I=1,NV

S(I)=S1*S(I)

SLOPE=S1*SLOPE

RETURN

END

*OPTIONS IN EFFECT=NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOOBL(NONE)
*OPTIONS IN EFFECT=SOURCE EBCDIC NOLIST NODECK OBJECT NCMAP NOFORMAT GOSTHOT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 40, PROGRAM SIZE = 654, SUBPROGRAM NAME =CN02
*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION *****

3012K BYTES OF CORE NOT USED
SUBROUTINE CN2NN3 (X,S,SLOPE,ALP,FFF,A1,A2,A3,A4,F1,F2,F3,F4,APP,IMPROV,NCAL,KCOUNT,JGOTO)  

C Routines to solve one-dimensional search in unconstrained minimization using 2-point quadratic interpolation, 3-point cubic interpolation and 4-point cubic interpolation. By G. N. VANDERPLAATS, 1972. NASA-AMES RESEARCH CENTER, MOUNT FLEET, CALIF. 

C ALP = proposed move parameter. SLOPE = initial function slope = S-transpose times Df. SLOPE must be negative. OBJ = initial function value. ZRO = 0.  

C INITIAL INFORMATION (ALPHA = 0)  

C INITIAL INFORMATION (ALPHA = 0)  

C DATA SET U477CN303 AT LEVEL 002 AS OF 03/13/81  

C DATA SET U477CN303 AT LEVEL 001 AS OF 02/13/81  

C DATA SET 9186CN303 AT LEVEL 001 AS OF 07/10/80  

C SOURCE IBM NCA I XML EEBDIC (ENHANCED) NOFORMAT NOXREF NOALC NOSHIFT OPTIONS (3) LINECOUNT (60) SIZE (MAX) AUTOFLAG (NONE) 

C SOURCE IBM NCA I XML EEBDIC (ENHANCED) NOFORMAT NOXREF NOALC NOSHIFT OPTIONS (3) LINECOUNT (60) SIZE (MAX) AUTOFLAG (NONE) 

C DATA SET U477CN303 AT LEVEL 002 AS OF 03/13/81  

C DATA SET U477CN303 AT LEVEL 001 AS OF 02/13/81  

C DATA SET 9186CN303 AT LEVEL 001 AS OF 07/10/80  

C SOURCE IBM NCA I XML EEBDIC (ENHANCED) NOFORMAT NOXREF NOALC NOSHIFT OPTIONS (3) LINECOUNT (60) SIZE (MAX) AUTOFLAG (NONE) 

C SOURCE IBM NCA I XML EEBDIC (ENHANCED) NOFORMAT NOXREF NOALC NOSHIFT OPTIONS (3) LINECOUNT (60) SIZE (MAX) AUTOFLAG (NONE)
C CHECK FOR ILL-CONDITIONING

ISH 0043 IF (KOUNT.GT.5) GO TO 60
ISH 0045 FF=2. *ABS(F1)
ISH 0046 IF (F2.LT.FF) GO TO 90
ISH 0048 FF=5. *ABS(F1)
ISH 0049 IF (F2.LT.FF) GO TO 60
ISH 0051 A2=.5*A2
ISH 0052 AP=-A2
ISH 0053 ALP=A2
ISH 0054 GO TO 30
ISH 0055 F3=F2
ISH 0056 A3=A2
ISH 0057 A2=.5*A2

C UPDATE DESIGN VECTOR AND FUNCTION VALUE

ISH 0058 AP=A2-ALP
ISH 0059 ALP=A2
ISH 0060 DO 70 I=1,NDV
ISH 0061 70 X(I)=X(I)+AP*S(I)
ISH 0062 IF (IPRINT.GT.4) WRITE (161,3701 A2
ISH 0064 IF (IPRINT.GT.4) WRITE (161,380) (X(I),I=1,NDV)
ISH 0066 NCAL(I)=NCAL(I)+1
ISH 0067 JGOTO=2
ISH 0068 RETURN
ISH 0069 CONTINUE
ISH 0070 F2=OBJ
ISH 0071 IF (IPRINT.GT.4) WRITE (161,390) F2
C PROCEED TO CUBIC INTERPOLATION.
ISH 0073 GO TO 160
ISH 0074 CONTINUE

C ******* 2-POINT QUADRATIC INTERPOLATION *******

ISH 0075 JJ=1
ISH 0076 II=1
ISH 0077 CALL CNM104 (II,APP,ZRO,A1,F1,SLOPE,A2,F2,ZRO,ZRO,ZRO,ZRO)
ISH 0078 IF (APP.LT.ZRO.OR.APP.GT.A2) GO TO 120
ISH 0080 F3=F2
ISH 0081 A3=A2
ISH 0082 A2=APP
ISH 0083 JJ=0

C UPDATE DESIGN VECTOR AND FUNCTION VALUE

ISH 0084 AP=A2-ALP
ISH 0085 ALP=A2
ISH 0086 DO 100 I=1,NDV
ISH 100 X(I)=X(I)+AP*S(I)
ISH 0088 IF (IPRINT.GT.4) WRITE (161,370) A2
ISH 0090 IF (IPRINT.GT.4) WRITE (161,380) (X(I),I=1,NDV)
ISH 0092 NCAL(I)=NCAL(I)+1
ISH 0093 JGOTO=3
ISH 0094 RETURN
ISH 0095 CONTINUE
ISH 0096 F2=OBJ
ISH 0097 IF (IPRINT.GT.4) WRITE (161,390) F2
GO TO 150

AP=A3-ALP
ALP=A3
DO 130 I=1,NDV
130 XI(I)=XI(I)+AP*S(I)
IF (IPRINT.GT.4) WRITE (161,370) A3
IF (IPRINT.GT.4) WRITE (161,380) (XI(I),I=1,NDV)
NCAL(1)=NCAL(1)+1
JGOTO=4
RETURN
CONTINUE
F3=OBJ
IF (IPRINT.GT.4) WRITE (161,390) F3
CONTINUE
IF (F3.LT.F2) GO TO 190
CONTINUE
C

CALL CHNNO4 (II,APP,ZRO,A1,F1,SLOPE,A2,F2,A3,F3,ZRO,ZRO)
IF (APP.LT.ZRO.OR.APP.GT.A3) GO TO 190
C
AP=APP-ALP
ALP=APP
DO 170 I=1,NDV
170 XI(I)=XI(I)+AP*S(I)
IF (IPRINT.GT.4) WRITE (161,370) ALP
IF (IPRINT.GT.4) WRITE (161,380) (XI(I),I=1,NDV)
NCAL(II)=NCAL(I)+1
JGOTO=5
RETURN
CONTINUE
IF (UPRWT.GT.4) WRITE (161,390) OBJ
C
AA=XX-APP/A2
AB2=ABS(F2)
AB3=ABS(XX)
AB=ABS
IF (AB3.GT.AB) AB=AB3
IF (AB.LT.1.0E-15) AB=1.0E-15
AB=(AB2-AB3)/A2
IF (ABS(AB).LT.1.0E-15.AND.ABS(AA).LT.0.01) GO TO 330
A3=APP
F3=OBJ
IF (A3.GT.A2) GO TO 230
A3=A2
F3=OBJ
CONTINUE
IF (IPRINT.GT.4) WRITE (161,390) OBJ
C

CHECK CONVERGENCE
C

SAVE XXXX,IPRINT,APP,ZRO,A1,F1,SLOPE,A2,F2,A3,F3,ZRO,ZRO
C ************ 4-POINT CUBIC INTERPOLATION ************

C UPDATE DESIGN VECTOR AND FUNCTION VALUE.

AP=AP-ALP
ALP=AP
DO 260 I=1,NDV
X(I)=X(I)+AP*SCI)
IF (IPRINT.GT.4) WRITE (161,370) ALP
IF (IPRINT.GT.4) WRITE (161,380) (X(I),I=1,NDV)
NCAL(I)=NCAL(I)+1
JGOTO=7
RETURN
270 CONTINUE
IF (IPRINT.GT.4) WRITE (161,390) OBJ
280 CONTINUE

C CHECK FOR ILL-CONDITIONING

AP=APP-ALP
ALP=APP
DO 260 I=1,NDV
X(I)=X(I)+AP*SCI)
IF (IPRINT.GT.4) WRITE (161,370) ALP
IF (IPRINT.GT.4) WRITE (161,380) (X(I),I=1,NDV)
NCAL(I)=NCAL(I)+1
JGOTO=7
RETURN
270 CONTINUE
IF (IPRINT.GT.4) WRITE (161,390) OBJ
280 CONTINUE

IF (OBJ.GT.F2.0 OR OBJ.GT.F3) GO TO 290
AP=AP+ALP
OBJ=OBJ+F3
IFI (OBJ.LE.F2) GO TO 330
AP=AP-ALP
OBJ=OBJ-F3
IFI (F2.LT.F1) GO TO 300
AP=AP-ALP
OBJ=OBJ-F2

DO 320 I=1,NDV
X(I)=X(I)+AP*S(I)

C -- UPDATE DESIGN VECTOR --

DO 320 I=1,NDV
X(I)=X(I)-ALP*S(I)
ALP=0
OBJ=FFF
CONTINUE

C -- CHECK FOR MULTIPLE MINIMA --

IF (OBJ.LE.FFF) GO TO 350
INITIAL FUNCTION IS MINIMUM.

DO 340 I=1,NDV
X(I)=X(I)+AP*S(I)

C -- FORTRAN --

FORMAT (//,5X,6H* * UNCONSTRAINED ONE-DIMENSIONAL SEARCH INFO*),
1MATION * * *)

*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOIDL NONE

*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NOHAP NOFORMAT GOSTMT NOLREF NODLALC NOHNSF TERM IBM FLAG(I)

*STATISTICS* SOURCE STATEMENTS = 249, PROGRAM SIZE = 3520, SUBPROGRAM NAME = CN2NO3

*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION *****

2972K BYTES OF CORE NOT USED
SUBROUTINE CNMN04 (II,XBAR,EPS,X1,Y1,SLOPE,X2,Y2,X3,Y3,X4,Y4)

ROUTINE TO FIND FIRST XBAR.GE.EPS CORRESPONDING TO A MINIMUM
OF A ONE-DIMENSIONAL REAL FUNCTION BY POLYNOMIAL INTERPOLATION.

BY G. N. VANDERPLAATS APRIL, 1972.

NASA-AIRES RESEARCH CENTER, MOFFETT FIELD, CALIF.

II = CALCULATION CONTROL.
1: 2-POINT QUADRATIC INTERPOLATION, GIVEN X1, Y1, SLOPE, X2 AND Y2.
2: 3-POINT QUADRATIC INTERPOLATION, GIVEN X1, Y1, X2, Y2, X3, Y3.
3: 3-POINT CUBIC INTERPOLATION, GIVEN X1, Y1, SLOPE, X2, Y2, X3, Y3.
4: 4-POINT CUBIC INTERPOLATION, GIVEN X1, Y1, X2, Y2, X3, Y3, X4, Y4.

EPS MAY BE NEGATIVE.

IF REQUIRED MINIMUM ON Y DOES NOT EXITS, OR THE FUNCTION IS
ILL-CONDITIONED, XBAR = EPS-1.0 WILL BE RETURNED AS AN ERROR
INDICATOR.

IF DESIRED INTERPOLATION IS ILL-CONDITIONED, A LOWER ORDER
INTERPOLATION, CONSISTANT WITH INPUT DATA, WILL BE ATTEMPTED,
AND II WILL BE CHANGED ACCORDINGLY.

XBAR1=EPS-1.

XBAR=XBAR1

DX=X1-X2

IF (ABS(DX).LT.1.0E-20) RETURN

AA=(SLOPE+Y2-Y1)/DX/DX

IF (AA.LT.1.0E-20) RETURN

BB=SLOPE-2.*AA*X1

XBAR=-.5*BB/AA

IF (XBAR.LT.EPS) XBAR=XBAR1

RETURN

II=1: 2-POINT QUADRATIC INTERPOLATION

II=2: 3-POINT QUADRATIC INTERPOLATION

II=3: 3-POINT CUBIC INTERPOLATION

II=4: 4-POINT CUBIC INTERPOLATION

DATA SET U477CNMN04 AT LEVEL 001 AS OF 02/13/81

DATA SET 9188CNMN04 AT LEVEL 001 AS OF 07/10/80

SUBROUTINE CNMN04 (II,XBAR,EPS,X1,Y1,SLOPE,X2,Y2,X3,Y3,X4,Y4)

ROUTINE TO FIND FIRST XBAR.GE.EPS CORRESPONDING TO A MINIMUM
OF A ONE-DIMENSIONAL REAL FUNCTION BY POLYNOMIAL INTERPOLATION.

DATA SET U477CNMN04 AT LEVEL 001 AS OF 02/13/81
**II=3: 3-POINT CUBIC INTERPOLATION**

1. Calculate $BB = (Y_2 - Y_1) / X_2 - AA * (X_2 * X_2 + X_1 * X_2 - 2 * X_11)$
2. Calculate $XBAR = -0.5 * BB / AA$
3. If $(XBAR < EPS)$, then $XBAR = XBAR_1$
4. Return

**II=4: 4-POINT CUBIC INTERPOLATION**

1. Calculate $XBAR_1 = X_11 * X_1$
2. Calculate $XBAR_2 = X_22 * X_2$
3. Calculate $XBAR_3 = X_33 * X_3$
4. If $(ABS(q2).LT.1.0E-30)$, then return
5. Calculate $Q_2 = X_11 * X_2 - X_22 * X_1$
6. Calculate $Q_4 = X_3 * X_3 * X_2$
7. Calculate $Q_5 = X_1 * X_1 * X_2$
8. Calculate $Q_6 = X_3 * X_3 * X_2$
9. Calculate $Q_7 = X_11 * X_22 * X_22 * X_11$
10. Calculate $Q_8 = X_22 * X_22 * X_11 * X_22$
11. Calculate $Q_9 = X_11 * X_22 * X_22 * X_11$
12. Calculate $Q_{10} = X_22 * X_22 * X_11 * X_22$
13. Calculate $Q_{11} = X_11 * X_22 * X_22 * X_11$
14. If $(ABS(DNONE).LT.1.0E-30)$, then return
15. Calculate $AA = ((X_11 * X_11 * (Y_2 - Y_1) - X_22 * X_22 * (Y_3 - Y_1) + (X_22 * X_22 * (Y_4 - Y_2) - X_33 * X_33 * (Y_4 - Y_2))) / (Q_2 * Q_4 - Q_2 * Q_6))$
16. Calculate $BB = ((Y_2 - Y_1) / X_2 - SLOPE * AA * X_2 * X_2 + X_1 * X_2 - 2 * X_11) / X_2$
17. Calculate $CC = SLOPE - 3 * AA * X_11 - 2 * BB * X_1$
18. Calculate $BAC = EB * BB - 3 * AA * CC$
19. If $(BAC < 0)$, then return
20. Calculate $BAC = SQRT(BAC)$
21. Calculate $XBAR = (BAC - BB) / (3 * AA)$
22. If $(XBAR < EPS)$, then $XBAR = EPS$
23. Return
ISH 0097 BAC=SQRT(BAC) 00114
ISH 0098 XBAR=(BAC-GB)/(3.*AA) 00115
ISH 0099 IF (XBAR.LT.EPS) XBAR=XBAR1 00116
ISH 0101 RETURN 00117
ISH 0102 60 CONTINUE 00118
ISH 0103 IF (INSDO.EQ.1) GO TO 40 00119
ISH 0105 GO TO 20 00120
ISH 0106 END 00121

*OPTIONS IN EFFECT: NAME(HAIN) OPTIMIZE(3) LIMECOUNT(60) SIZE(MAX) AUTODEPLICATE

*OPTIONS IN EFFECT: SOURCE EBCDIC NOLIST NODECK OBJECT NOPAP NOFORMAT GOSTMT NOXREF NDALC NOSF TERM IBM FLAG(I)

*STATISTICS: SOURCE STATEMENTS = 105, PROGRAM SIZE = 1754, SUBPROGRAM NAME = CN3104

*STATISTICS: NO DIAGNOSTICS GENERATED

***** END OF COMPILED *****/

3004K BYTES OF CORE NOT USED
SUBROUTINE CNNH05 (G,DF,AS,B,C,SLOPE,PHI,ISC,IC,N1,N2,N3,N4,N5)
DATA SET U477CNMH05 AT LEVEL 002 AS OF 03/13/81
DATA SET U477CNMH05 AT LEVEL 001 AS OF 02/13/81
DATA SET U477CNMH05 AT LEVEL 001 AS OF 07/10/80
COMMON /CHM1/ DELFUN, DABFUN, FDCH, FDCHN, CT, CTMIN, CTMAX, ALPHAX
1, ABOBJJ, THETA, OBJ, NOV, NCON, NSIDE, IPRINT, NFDF, JSCL, LINDJ, ITMAX, ITT00006
2RM, INDIR, IGOTO, TAC, IHFO, INFO:
C ROUTINE TO SOLVE DIRECTION FINDING PROBLEM IN MODIFIED METHOD OF
C FEASIBLE DIRECTIONS.
C BY G. N. VAIDERPLAATS MAY, 1972.
C NASA-AMES RESEARCH CENTER, Moffett Field, Calif.
C NORM OF S VECTOR USED HERE IS S-TRANSPOSE TIMES S.LE.1.
C IF NVC = 0 FIND DIRECTION BY ZOUTENDIJK'S METHOD. OTHERWISE
C FIND MODIFIED DIRECTION.
C *** NORMALIZE GRADIENTS, CALCULATE THETA'S AND DETERMINE NVC ***
C
C NDV1=NDV+1
C NDV2=NDV+2
C NAC1=NAC+1
C NVC=0
C THMAX=0.
C CT=ABS(CT)
C CT1=1./CT
C CT1N=ABS(CT1N)
A1=1.
DO 40 I=1,NAC
CALCULATE THETA
NC=IC(I)
IF (NCI.LE.NCON) NCJ=ISC(NCI)
C1=GC(NCI)
CT=CT1
CTB=ABS(CTB)
IF (CT1 .LT. CTB) GO TO 10
CT2=CTB
CT2M=ABS(CT2M)
A(NDV2,II)=I.}
C DO 40 I=1,NAC
C CALCULATE THETA
NC=IC(I)
IF (NCI.LE.NCON) NCJ=ISC(NCI)
C1=GC(NCI)
CT=CT1
CTB=ABS(CTB)
IF (CT1 .LT. CTB) GO TO 10
CT2=CTB
CT2M=ABS(CT2M)
A(NDV2,II)=I.}
C DO 40 I=1,NAC
C CALCULATE THETA
NC=IC(I)
IF (NCI.LE.NCON) NCJ=ISC(NCI)
C1=GC(NCI)
CT=CT1
CTB=ABS(CTB)
IF (CT1 .LT. CTB) GO TO 10
CT2=CTB
CT2M=ABS(CT2M)
A(NDV2,II)=I.}
IF (NCI.GT.NCON) GO TO 40

DO 20 J=1,NDV

A1=A1+A(J,I)**2

20 CONTINUE

IF (A1.LT.1.0E-20) A1=1.0E-20

A1=SQRT(A1)

A(NDV2,I)=A1

A1=1/A1

DO 30 J=I,NDV

A(J,I)=A1*A(J,I)

30 CONTINUE

C NORMALIZE GRADIENT OF OBJECTIVE FUNCTION AND STORE IN NAC+1

C COLUMN OF A

C

A1=0.

DO 50 I=1,NDV

A1=A1+DF(I)**2

50 CONTINUE

IF (A1.LT.1.0E-20) A1=1.0E-20

A1=SQRT(A1)

A1=1/A1

DO 60 I=1,NDV

60 A(I,HAC1)=A1*DF(I)

C BUILD VECTOR.

C

IF (NYC.GT.0) GO TO 80

C BUILD FOR CLASSICAL METHOD

C

NDB=HAC1

A(NDV1,NDB)=1.

DO 70 J=1,NDB

70 C(J)=-A(NDV1,J)

GO TO 110

80 CONTINUE

C BUILD FOR MODIFIED METHOD

C

NDB=NAC

A(NDV1,NAC)=PHI

C SCALE THETA'S SO THAT MAXIMUM THETA IS UNITY

C

IF (THMAX.GT.0.00001) THMAX=1./THMAX

DO 100 I=1,NDB

A(NDV1,I)=A(NDV1,I)*THMAX

100 CONTINUE

C BUILD B MATRIX

C

DO 120 J=1,NDB

120 B(I,J)=0.

C BUILD B MATRIX
ISH 0089 DO 120 K=1,NDV1
ISH 0090 120 B(I,J)=B(I,J)-A(K,I)*A(K,J)
ISH 0091 CALL CNMN08 (WDB,ỊER,C,MS1,B,N3,N4,N5)
ISH 0092 IF (ỊPRINT.GT.1.AND.ỊER.GT.0) WRITE (161,180)
ISH 0094 C CALCULATE RESULTING DIRECTION VECTOR, S.
ISH 0095 SLOPE=0.
ISH 0096 C------------------------------------------------------------------
ISH 0097 C SOLVE SPECIAL L. P. PROBLEM
ISH 0098 C------------------------------------------------------------------
ISH 0099 DO 140 I=I,NDV
ISH 1000 140 S(I)=SI
ISH 1001 C------------------------------------------------------------------
ISH 1002 C USABLE-FEASIBLE DIRECTION
ISH 1003 C------------------------------------------------------------------
ISH 1004 DO 130 J=I,NOB
ISH 1005 130 SI=SI-A(I,J)*C(J)
ISH 1006 SLOPE=SLOPE+SI*DF(I)
ISH 1007 140 S(I)=SI
ISH 1008 C------------------------------------------------------------------
ISH 1009 C NORMALIZE S TO MAX ABS OF UNITY
ISH 1010 C------------------------------------------------------------------
ISH 1011 S1=0.
ISH 1012 DO 160 I=I,NOV
ISH 1013 160 SI=ABS(S(I))
ISH 1014 IF (SI.GT.1.E-10) 161 CONTINUE
ISH 1015 161 SI=1./SI
ISH 1016 IF (SI.LT.1.E-10) RETURN
ISH 1017 SI=1.
ISH 1018 DO 170 I=1,NOV
ISH 1019 170 S(I)=SI*S(I)
ISH 1020 SLOPE=SI*SLOPE
ISH 1021 170 CONTINUE
ISH 1022 FORMAT (/5X,46H* DIRECTION FINDING PROCESS DID NOT CONVERGE/5X,0157
129H* S-VECTOR MAY NOT BE VALID)
ISH 1023 RETURN
ISH 1024 C------------------------------------------------------------------
ISH 1025 C FORMATS
ISH 1026 C------------------------------------------------------------------
ISH 1027 180 FORMAT (/5X,46H* SOURCE STATEMENTS = 122, PROGRAM SIZE = 2548, SUBPROGRAM NAME =CMN05
129H* NO DIAGNOSTICS GENERATED
129H* OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOOBJ(NONE)
129H* OPTIONS IN EFFECT*SOURCE EDDIC NOLIST NODIAG NOHEADER NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
129H* STATISTICS* SOURCE STATEMENTS = 122, PROGRAM SIZE = 2548, SUBPROGRAM NAME =CMN05
129H* END OF COMPIILATION ******
129H* 2992K BYTES OF CORE NOT USED
SUBROUTINE CHMN06 (X,V1B,VUB,G,SCAl,DFlH2,SIH2,GlH2,CTAM,CTBN,SL0PE,ALP,Fl,F2,F3,CV1,CV2,CV3,CV4,AlPCA,AlPFES,AlPM,xAct,xFnc,x !nfo, xGoto)

C ROUTINE TO SOLVE ONE-DIMENSIONAL SEARCH PROBLEM FOR CONSTRAINED FUNCTION MINIMIZATION. BY G. N. VANDERPLAATS, NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF. AUG., 1974.

C PROPOSED MOVE.
C ------------------------------------------------------------------
C ***** BEGIN SEARCH OR IMPOSE SIDE CONSTRAINT MODIFICATION *****
C ------------------------------------------------------------------

ZRO=0.0. 00027
JGOTO.EQ.0) GO TO 10 00029
GO TO (140,310,520),JGOTO 00030
10 IF (IPRINT.GE.5) WRITE (161,730) 00031
AlPSAV=AlP 00032
ICOUNT=ICOUNT+1 00033
AlPSID=I.0E+20 00034
C INITIAL ALPHA AND OBJ.
C ------------------------------------------------------------------
C **** BEGIN SEARCH OR IMPOSE SIDE CONSTRAINT MODIFICATION ****
C ------------------------------------------------------------------

ZAO=O.
IF (NSIDE.EQ.O) GO TO 70
GO TO (140,310,520),JGOTO
10 IF (IPRINT.GE.5) WRITE (161,730)
A2=AlPSAV
ICOUNT=ICOUNT+1
ALPSID=I.0E+20
C INITIAL ALPHA AND OBJ.
C ------------------------------------------------------------------
C **** BEGIN SEARCH OR IMPOSE SIDE CONSTRAINT MODIFICATION ****
C ------------------------------------------------------------------

ZAO=O.
IF (NSIDE.EQ.O) GO TO 70
GO TO (140,310,520),JGOTO
10 IF (IPRINT.GE.5) WRITE (161,730)
A2=AlPSAV
ICOUNT=ICOUNT+1
ALPSID=I.0E+20
C INITIAL ALPHA AND OBJ.
C 30 CONTINUE
ISN 0034 X1=X(I)
ISN 0035 SI=1./SI
ISN 0036 IF (SI.GT.0.) GO TO 40
ISN 0037 X12=VUB(I)
ISN 0038 X11=ABS(X12)
ISN 0039 IF (X11.LT.1.1) X11=1.
ISN 0040 CONRAIN VALUE.
ISN 0041 GI=(X12-X11)/X11
ISN 0042 IF (GI.GT.-1.0E-6) GO TO 50
ISN 0043 ALPA=(X12-X11)*SI
ISN 0044 IF (ALPA.LT.ALPSID) ALPSID=ALPA
ISN 0045 GO TO 60
ISN 0046 40 CONTINUE
C UPPER BOUND.
ISN 0050 X12=VUB(I)
ISN 0051 X11=ABS(X12)
ISN 0052 IF (X11.LT.1.1) X11=1.
ISN 0053 CONRAIN VALUE.
ISN 0054 GI=(X1-X11)/X11
ISN 0055 IF (GI.GT.-1.0E-6) GO TO 50
ISN 0056 ALPA=(X1-X11)*SI
ISN 0057 ALPSID=ALPA
ISN 0058 IF (ALPA.LT.ALPSID) ALPSID=ALPA
ISN 0059 GO TO 60
ISN 0060 50 CONTINUE
C MOVE WILL VIOLATE SIDE CONSTRAINT. SET S(I)=0.
ISN 0062 SLOPE=SLOPE-S(I)*DF(I)
ISN 0063 S(I)=0.
ISN 0064 KSID=KSID+1
ISN 0065 60 CONTINUE
C ALPSID IS UPPER BOUND ON ALPHA.
ISN 0066 IF (A2.GT.ALPSID) A2=ALPSID
ISN 0067 70 CONTINUE
C CHECK ILL-CONDITIONING
ISN 0069 IF (KSID.EQ.NDV.OR.ICOUNT.GT.10) GO TO 710
ISN 0070 ICOUNT=1.
ISN 0071 IF (NVC.EQ.0.AND.SLOPE.GT.0.) GO TO 710
C STORE CONSTRAINT VALUES IN GI.
ISN 0080 DO 80 I=1,NCON
ISN 0081 GI(I)=GI(I)
ISN 0082 80 CONTINUE
ISN 0083 90 CONTINUE

---

DO 60 I=1,NDV
S(I)=S(I)
IF (ABS(S(I)).GT.1.0E-20) GO TO 30
C ITH COMPONENT OF S IS SMALL. SET TO ZERO.
S(I)=0.
SLOPE=SLOPE-S(I)*DF(I)
DO 60
C IF (SI.GT.0.) GO TO 40
XI=X(I)
SI=1./SI
IF (SI.GT.0.) GO TO 40
XI2=VUB(I)
X11=ABS(X12)
IF (X11.LT.1.1) X11=1.
CONRAIN VALUE.
GI=(X12-X11)/X11
IF (GI.GT.-1.0E-6) GO TO 50
ALPA=(X12-X11)*SI
IF (ALPA.LT.ALPSID) ALPSID=ALPA
GO TO 60
CONTINUE
C UPPER BOUND.
XI2=VUB(I)
X11=ABS(X12)
IF (X11.LT.1.1) X11=1.
CONRAIN VALUE.
GI=(X1-X11)/X11
IF (GI.GT.-1.0E-6) GO TO 50
ALPA=(X1-X11)*SI
IF (ALPA.LT.ALPSID) ALPSID=ALPA
GO TO 60
CONTINUE
C MOVE WILL VIOLATE SIDE CONSTRAINT. SET S(I)=0.
SLOPE=SLOPE-S(I)*DF(I)
S(I)=0.
KSID=KSID+1
CONTINUE
ALPSID IS UPPER BOUND ON ALPHA.
IF (A2.GT.ALPSID) A2=ALPSID
CONTINUE
CHECK ILL-CONDITIONING
IF (KSID.EQ.NDV.OR.ICOUNT.GT.10) GO TO 710
IF (NVC.EQ.0.AND.SLOPE.GT.0.) GO TO 710
ALPSID=ALPSID
ALPHN1=A1=ALPSID
ALPN=ALPSID
ALPHA=ALPSID
IF (HCON.EQ.0.) GO TO 90
 STORE CONSTRAINT VALUES IN GI.
DO 80 I=1,NCON
GI(I)=GI(I)
CONTINUE
MOVE A DISTANCE A2*S

ALPTOT=ALPTOT+A2

DO 100 I=1,NDV
X(I)=X(I)+A2*S(I)

CONTINUE

IF (IPRINT.LT.5) GO TO 130
WRITE (I61,740) A2
IF (INSCAL.EQ.0) GO TO 120
DO 110 I=1,NOV
G(I)=SCAL(I)*X(I)
WRITE (I61,750) (G(I),I=1,NDV)
GO TO 130
WRITE (I61,750) (X(I),I=1,NDV)

UPDATE FUNCTION AND CONSTRAINT VALUES

NCAL(I)=NCAL(I)+1
JGOTO=1
RETURN

CONTINUE

F2=OBJ
IF (IPRINT.GE.5) WRITE (I61,760) F2
IF (INCON.EQ.0) GO TO 150
WRITE (I61,770)
WRITE (I61,750) (G(I),I=1,NCON)
CONTINUE

IDENTIFY ACCEPTABILITY OF DESIGNS F1 AND F2

IGOOD1=0
IGOOD2=0
CV1=0.
CV2=0.
NVC1=0
NVC2=0
IF (INCON.EQ.0) GO TO 170
DO 160 I=1,NCON
CC=CTAM
IF (ISC(I).GT.0) CC=CTBM
C1=G(I)-CC
C2=G(I)-CC
IF (C2.GT.0.) NVC1=NVC1+1
IF (C1.GT.CV1) CV1=C1
IF (C2.GT.CV2) CV2=C2
CONTINUE
GO TO 160
CONTINUE
IF (CV1.GT.0.) IGOOD1=1
IF (CV2.GT.0.) IGOOD2=1
CONTINUE
ALP=A2
OBJ=F2

IF (F2 VIOLATES FEWER CONSTRAINTS THAN F1 BUT STILL HAS CONSTRAINT VIOLATIONS) RETURN

CONTINUE

IF (NVC1.LT.NVC.AND.NVC1.GT.0) GO TO 710

CONTINUE

IF (NVC1.GT.0) GO TO 710

CONTINUE
IDENTIFY BEST OF DESIGNS F1 ANF F2

IF CONSTRAINTS ARE VIOLATED, IBEST CORRESPONDS TO MINIMUM
CONVIATION.

IF (IGOOD1.EQ.0.AND.IGOOD2.EQ.0) GO TO 100
VOLTED CONSTRAINTS. PICK MINIMUM VIOLATION.

IEST=1
IBEST=1
GO TO 190
CONTINUE
IEST=1
IBEST=1
IF (F2.LE.F1) IBEST=2
CONTINUE
II=1
IF (HCON.EQ.0.) GO TO 230

CIDEITFY 
BEST OF DESIGNS 
9ANF 
F2
00173
00174
00175
00176
00177
00178
00179
00180
00181
00182
00183
00184
00185
00186
00187
00188
00189
00190
00191
00192
00193
00194
00195
00196
00197
00198
00199
00200
00201
00202
00203
00204
00205
00206
00207
00208
00209
00210
00211
00212
00213
00214
00215
00216
00217
00218
00219
00220
00221
00222
00223
00224
00225
00226
00227
00228
00229
00230
00231
00232
00233
00234
00235
00236
00237
00238
00239
00240
00241
00242
00243
00244
00245
00246
00247
00248
00249
00250
00251
00252
00253
00254
00255
00256
00257
00258
00259
00260
00261
00262
00263
00264
00265
00266
00267
00268
00269
00270
00271
00272
00273
00274
00275
00276
00277
00278
00279
00280
00281
00282
00283
00284
00285
00286
00287
00288
00289
00290
00291
00292
00293
00294
00295
00296
00297
00298
00299
00300
00301
00302
00303
00304
00305
00306
00307
00308
00309
00310
00311
00312
00313
00314
00315
00316
00317
00318
00319
00320
00321
00322
00323
00324
00325
00326
00327
00328
00329
00330
00331
C LIMIT MOVE TO NEW CONSTRAINT ENCOUNTER. 00232
ISN 0192 IF (A3.GT.ALPLN) A3=ALPLN 00233
ISN 0194 IF (A3.GT.ALPLN) A3=ALPLN 00234
C MAKE A3 NON-ZERO. 00235
ISN 0196 IF (A3.LE.1.0E-20) A3=1.0E-20 00236
C IF A3=A2=ALPSID AND F2 IS BEST, GO INVOKE SIDE CONSTRAINT 00237
ISN 0198 MODIFICATION. 00238
CISN 0200 ISN 0201 IF (ABS(ALPB).LT.1.0E-10.AND.ABS(ALPA).LT.1.0E-10) JBEST=1 00239
ISN 0203 IF (JBEST.EQ.1.AND.JBEST.EQ.2) GO TO 20 00240
C SIDE CONSTRAINT CHECK NOT SATISFIED. 00241
ISN 0205 IF (NCON.EQ.0) GO TO 260 00242
C STORE CONSTRAINT VALUES IN G2. 00243
ISN 0207 DO 250 I=1,NCON 00244
ISN 0208 G(I)=G(I) 00245
CONTINUE 00246
CONTINUE 00247
ISN 0209 250 CONTINUE 00248
ISN 0210 260 CONTINUE 00249
C IF A3=A2, SET A3=.9*A2. 00250
ISN 0211 IF (ABS(ALPB).LT.1.0E-10) A3=.9*A2 00251
C MOVE AT LEAST .01*A2. 00252
ISN 0213 IF (A3.LT.(1.01*A2)) A3=.01*A2 00253
C LIMIT MOVE TO 5.*A2. 00254
ISN 0215 IF (A3.GT.(5.*A2)) A3=5.*A2 00255
C LIMIT MOVE TO ALPSID. 00256
ISN 0217 IF (A3.GT.ALPSID) A3=ALPSID 00257
C MOVE A DISTANCE A3*S. 00258
ISN 0219 ALP=A3-A2 00259
ISN 0220 ALPTOT=ALPTOT+ALP 00260
ISN 0221 DO 270 I=1,NDV 00261
ISN 0222 X(I)=X(I)+ALP*SI(I) 00262
CONTINUE 00263
ISN 0223 270 CONTINUE 00264
ISN 0224 IF (IPRINT.LT.5) GO TO 300 00265
ISN 0225 WRITE (I61,780) 00266
ISN 0226 WRITE (I61,740) A3 00267
ISN 0227 WRITE (I61,740) F3 00268
ISN 0228 IF (NSCAL.EQ.0) GO TO 290 00269
ISN 0229 DO 280 I=1,NDV 00270
ISN 0230 DO 280 I=1,NDV 00271
ISN 0231 280 G(I)=SCAL(I)*X(I) 00272
ISN 0232 WRITE (I61,750) (G(I),I=1,NDV) 00273
ISN 0233 GO TO 300 00274
ISN 0234 WRITE (I61,750) (X(I),I=1,NDV) 00275
ISN 0235 290 WRITE (I61,750) (X(I),I=1,NDV) 00276
CONTINUE 00277
C UPDATE FUNCTION AND CONSTRAINT VALUES 00278
C UPDATE FUNCTION AND CONSTRAINT VALUES 00279
ISN 0236 NCAL(I)=NCAL(I+1) 00280
ISN 0237 JGOTO=2 00281
ISN 0238 RETURN 00282
ISN 0239 310 CONTINUE 00283
ISN 0240 F3=OBJ 00284
ISN 0241 IF (IPRINT.GE.5) WRITE (I61,760) F3 00285
ISN 0242 IF (IPRINT.LT.5.OR.NCON.EQ.0) GO TO 320 00286
ISN 0243 WRITE (I61,770) 00287
ISN 0244 WRITE (I61,750) (G(I),I=1,NCON) 00288
ISN 0245 WRITE (I61,750) (X(I),I=1,NCON) 00289
ISN 0246 CONTINUE 00290
CONTINUE 00291
C UPDATE FUNCTION AND CONSTRAINT VALUES 00292
C UPDATE FUNCTION AND CONSTRAINT VALUES 00293
ISN 0247 320 CONTINUE 00294
C CALCULATE MAXIMUM CONSTRAINT VIOLATION AND PICK BEST DESIGN 00295
C CALCULATE MAXIMUM CONSTRAINT VIOLATION AND PICK BEST DESIGN 00296
C CALCULATE MAXIMUM CONSTRAINT VIOLATION AND PICK BEST DESIGN 00297
ISH 0246  CV3=0.
ISH 0249  IGOOD3=0.
ISH 0250  NVC1=0.
ISH 0251  IF (NCON.EQ.0) GO TO 340.
ISH 0253  DO 330 I=I,NCOH.
ISH 0255  CC=CTAH.
ISH 0257  CI=G(I)-CC.
ISH 0259  IF (C1.GT.CV3) CV3=C1.
ISH 0260  IF (C1.GT.0.) NVC1=NVC1+1.
ISH 0262  CONTINUE.
ISH 0265  IF (CV3.GT.0.) IGOOD3=1.
ISH 0266  IF (IBEST.EQ.2) GO TO 360.
ISH 0267  IF (IGOOD1.EQ.0.AND.IGOOD3.EQ.0) GO TO 350.
ISH 0268  IF (CV1.GE.CV3) IBEST=3.
ISH 0269  GO TO 380.
ISH 0270  IF (F3.LE.Fl) IBEST=3.
ISH 0271  IF (HVCl.NT.NVCI GO TO 710.
ISH 0272  IF (LINEOBJ.NE.0.AND.IICON.EQ.NICON) GO TO 710.
ISH 0273  IF (A3 = ALPNI AND F3 IS BOTH GOOD AND BEST RETURN).
ISH 0274  IF ((ABS(ALPNI).LT.1.0E-20.AND.IBEST.EQ.3).AND.(IGOOD3.EQ.0)) GO TO 370.
ISH 0275  CONTINUE.
ISH 0276  IF (F3 VIOLATES FEWER CONSTRAINTS THAN F1 RETURN).
ISH 0277  IF (NVC1.LT.NVC) GO TO 710.
ISH 0278  IF OBJECTIVE AND ALL CONSTRAINTS ARE LINEAR, RETURN.
ISH 0279  IF (LINEOBJ.EQ.0.AND.IICON.EQ.NICON) GO TO 710.
ISH 0280  IF A3 = ALPNI AND F3 IS BOTH GOOD AND BEST RETURN.
ISH 0281  IF ((ABS(ALPNI).LT.1.0E-20.AND.IBEST.EQ.3) GO TO 20.
ISH 0282  CONTINUE.
ISH 0283  IF A3 = ALPSID AND F3 IS BEST, GO INVOKE SIDE CONSTRAINT MODIFICATION.
ISH 0284  ALP=A3.
ISH 0285  ALPC=A3.
ISH 0286  OBJ=F3.
ISH 0287  IF F3 VIOLATES FEWER CONSTRAINTS THAN F1 RETURN.
ISH 0288  IF (NVC1.LT.NVC) GO TO 710.
ISH 0289  IF OBJECTIVE AND ALL CONSTRAINTS ARE LINEAR, RETURN.
ISH 0290  IF (LINEOBJ.EQ.0.AND.IICON.EQ.NICON) GO TO 710.
ISH 0291  IF A3 = ALPNI AND F3 IS BOTH GOOD AND BEST RETURN.
ISH 0292  IF ((ABS(ALPNI).LT.1.0E-20.AND.IBEST.EQ.3).AND.(IGOOD3.EQ.0)) GO TO 370.
ISH 0293  1.
ISH 0294  IF A3 = ALPSID AND F3 IS BEST, GO INVOKE SIDE CONSTRAINT MODIFICATION.
ISH 0295  ALP=A3.
ISH 0296  IF (ABS(ALPNI).LT.1.0E-20.AND.IBEST.EQ.3) GO TO 20.
ISH 0297  CONTINUE.
ISH 0298  ALPC=A3.
ISH 0299  ALPES=-1.
ISH 0300  ALPNI=-1.
ISH 0301  IF (IICON.EQ.0) GO TO 440.
ISH 0302  III=0.
ISH 0303  III=III+1.
ISH 0304  CONTINUE.
ISH 0305  CI=G(I).
ISH 0306  C2=G2(I).
ISH 0307  C3=G3(I).
ISH 0308  IF (ISCI.EQ.0) GO TO 400.
ISH 0309  CONTINUE.
ISH 0310  CONTINUE.
ISH 0311  CONTINUE.
ISH 0312  CONTINUE.
ISH 0313  CONTINUE.
ISH 0314  CONTINUE.
ISH 0315  CONTINUE.
ISH 0316  CONTINUE.
ISH 0317  CONTINUE.
ISH 0318  CONTINUE.
ISH 0319  CONTINUE.
ISH 0320  CONTINUE.
ISH 0321  CONTINUE.
ISH 0322  CONTINUE.
ISH 0323  CONTINUE.
ISH 0324  CONTINUE.
ISH 0325  CONTINUE.
ISH 0326  CONTINUE.
ISH 0327  CONTINUE.
ISH 0328  CONTINUE.
ISH 0329  CONTINUE.
ISH 0330  CONTINUE.
ISH 0331  CONTINUE.
ISH 0332  CONTINUE.
ISH 0333  CONTINUE.
ISH 0334  CONTINUE.
ISH 0335  CONTINUE.
ISH 0336  CONTINUE.
ISH 0337  CONTINUE.
ISH 0338  CONTINUE.
ISH 0339  CONTINUE.
ISH 0340  CONTINUE.
ISH 0341  CONTINUE.
ISH 0342  CONTINUE.
ISH 0343  CONTINUE.
ISH 0344  CONTINUE.
ISH 0345  CONTINUE.
ISH 0346  CONTINUE.
ISH 0347  CONTINUE.
ISH 0348  CONTINUE.
ISH 0349  CONTINUE.
IF (CT. LE. CTBMI GO TO 430
11=1
CALL CMM07 (II, ALP, ZRO, ZRO, C1, A2, C2, A3, C3)
IF (ALP .GT. ALPFES) ALPFES = ALP
GO TO 430
CONTINUE
C NON-LINEAR CONSTRAINT
------------------------------------------------------------------
11=2
CALL CMM07 (II, ALP, ZRO, ZRO, C1, A2, C2, A3, C3)
IF (ALP .LE. ZROI) GO TO 430
IF (C1 .GE. CT. AND. C1 .LE. 0.) GO TO 410
IF (C1 .GT. CTAM .OR. C1 .LT. 0.) GO TO 420
ALP IS MINIMUM MOVE. UPDATE FOR NEXT CONSTRAINT ENCOUNTER.
ALPA = ALP
CALL CMR:N07 UI, ALP, ALPA, ZRO, C1, A2, C2, A3, C3
IF (ALP .LT. ALPCA. AND. ALP .GE. ALPA) ALPCA = ALP
GO TO 430
CONTINUE
IF (ALP .GT. ALPFES .AND. C1 .GT. CTAM) ALPFES = ALP
IF (ALP .LT. ALPNC .AND. C1 .LT. 0.) ALPNC = ALP
ALPHA = ALP
CONTINUE
IF (III .LT. NCOHI) GO TO 390
CONTINUE
IF (LINOBJ .GT. 0. OR. SLOPE .GT. 0.) GO TO 450
CALCULATE ALPHA TO MINIMIZE FUNCTION
------------------------------------------------------------------
11=3
IF (A2 .GT. A3 .AND. (IGOOD2 .EQ. 0. AND. IBEST .EQ. 2)) II=2
CALL CMM04 (II, ALPMIN, ZRO, ZRO, F1, SLOPE, A2, F2, A3, F3, ZRO, ZRO)
CONTINUE
------------------------------------------------------------------
MOTION AT LEAST ENOUGH TO OVERCOME CONSTRAINT VIOLATIONS.
A4 = ALPFES
MOVE TO MINIMIZE FUNCTION.
IF (ALPMIN .GT. A4) A4 = ALPMIN
IF (A4 .LE. 0.) SET A4 = ALPSID.
IF (A4 .LT. 0.) A4 = ALPSID
LIMIT MOVE TO NEW CONSTRAINT ENCOUNTER.
IF (A4 .GT. ALPHA) A4 = ALPHA
IF (A4 .GT. ALPHC) A4 = ALPHC
LIMIT MOVE TO RE-ENCOUNTER CURRENTLY ACTIVE CONSTRAINT.
IF (A4 .GT. ALPCA) A4 = ALPCA
LIMIT A4 TO 5. * A3.
IF (A4 .GT. (5. * A3)) A4 = 5. * A3
UPDATE DESIGN.
IF (IBEST .NE. 3. OR. NCON .EQ. 0.) GO TO 470
STORE CONSTRAINT VALUES IN G2. F3 IS BEST. F2 IS NOT.
DO 460 I = 1, NCON
G2(I) = G(I)
CONTINUE
------------------------------------------------------------------
IF A4 = A3 AND IGOOD1 = 0 AND IGOOD3 = 1, SET A4 = 0.9*A3.
IF (IGOOD1 .EQ. 0. AND. IGOOD3 .EQ. 1.) .AND. (ABS(ALP) .LT. 1.0E-20) A4 = 0.9*A00408
C MOVE A DISTANCE A4*S

ALP=A4-A3
ALP=ALP+ALP
DO 460 I=1,NDV
X(I)=X(I)+ALP*SII)
460 CONTINUE
IF (IPRINT.LT.5) GO TO 510
WRITE 1161,720) A4
IF (NSCAL.EQ.O) GO TO 500
DO 490 I=I,NDV
G(I)=SCAL(II*X(I)
WRITE (161,750) (G(I),I=I,NDV)
GO TO 510
WRITE (161,750) (X(I),I=I,NDV)
CONTINUE
NCAL(I)=NCAL(I)+1
JGOTO=3
RETURN
F4=OBJ
IF (IPRINT.GE.5) WRITE (1161,750) F4
IF (IPRINT.LT.5.OR.NCON.EQ.0) GO TO 530
WRITE (161,750) (G(I),I=I,NCON)
CONTINUE
IGOOD4=0
CV4=0.
IF (NCON.EQ.O) GO TO 550
DO 540 I=I,IICON
CC=CTAN
IF (ISC(I).GT.O) CC=CTBM
Cl=G(I)-CC
IF (Cl.GT.CV4) CV4=Cl
CONTINUE
IF (CV4.GT.0.) IGOO4=1
CONTINUE
ALP=A4
OBJ=F4
CONTINUE
GO TO (560,610,660),IBEST
CONTINUE
CHOOSE BETWEEN F1 AND F4.
IF (IGOOD1.EQ.0.AND.IGOOD4.EQ.0) GO TO 570
IF (CV4.GT.0.) IGOO4=1
CONTINUE
IF (F4.LE.Fl) GO TO 710
CONTINUE
Fl IS BEST.
ISN 0426 ALP'TOT=ALP'TOT-A4
ISN 0427 OBJ=F1
ISN 0428 DO 590 I=1,NDV
ISN 0429 XI(I)=XI(I)-A4*SI(I)
ISN 0430 CONTINUE
ISN 0431 IF (NCON.EQ.0) GO TO 710
ISN 0432 DO 600 I=1,NCON
ISN 0433 GI(I)=GI(I)
ISN 0434 600 CONTINUE
ISN 0435 610 CONTINUE
C CHOOSE BETWEEN F2 AND F4.
ISN 0437
ISN 0438 IF (IGOOD2.EQ.0.AND.IGOOD4.EQ.0) GO TO 620
ISN 0440 IF (CV4.GT.CV4) GO TO 710
ISN 0442 GO TO 630
ISN 0443 620 CONTINUE
ISN 0444 IF (F4.LE.F2) GO TO 710
ISN 0446 630 CONTINUE
C F2 IS BEST.
ISN 0447 OBJ=F2
ISN 0448 A2=A4-A2
ISN 0449 ALP'TOT=ALP'TOT-A2
ISN 0450 DO 640 I=1,NDV
ISN 0451 XI(I)=XI(I)-A2*SI(I)
ISN 0452 640 CONTINUE
ISN 0453 IF (NCON.EQ.0) GO TO 710
ISN 0455 DO 650 I=1,NCON
ISN 0456 GI(I)=GI(I)
ISN 0457 650 CONTINUE
ISN 0458 GO TO 710
ISN 0459 660 CONTINUE
C CHOOSE BETWEEN F3 AND F4.
ISN 0460 IF (IGOOD3.EQ.0.AND.IGOOD4.EQ.0) GO TO 670
ISN 0462 IF (CV3.GT.CV4) GO TO 710
ISN 0464 GO TO 680
ISN 0465 670 CONTINUE
ISN 0466 IF (F4.LE.F3) GO TO 710
ISN 0468 680 CONTINUE
C F3 IS BEST.
ISN 0469 OBJ=F3
ISN 0470 A3=A4-A3
ISN 0471 ALP'TOT=ALP'TOT-A3
ISN 0472 DO 690 I=1,NDV
ISN 0473 XI(I)=XI(I)-A3*SI(I)
ISN 0474 690 CONTINUE
ISN 0475 IF (NCON.EQ.0) GO TO 710
ISN 0477 DO 700 I=1,NCON
ISN 0478 GI(I)=G3(I)
ISN 0479 700 CONTINUE
ISN 0480 710 CONTINUE
ISN 0481 ALP=ALP'TOT
ISN 0482 IF (IPRINT.GE.5) WRITE (6,790)
ISN 0484 JGOTO=0
ISN 0485 RETURN
C---------------------------------------------------------------
C FORMATS
C---------------------------------------------------------------
**VERSION 1.3.0 (01 MAY 80) CIN606 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 82.141/10.53.39 PAGE 10**

ISH 0486  720  FORMAT (/5X,25THREE-POINT INTERPOLATION) 00527
ISH 0487  730  FORMAT (//,50H* * CONSTRAINED ONE-DIMENSIONAL SEARCH INFORMATION) 00528
ISH 0488  740  FORMAT (/5X,15HPROPOSED DESIGN/5X,7HALPHA =,E12.5/5X,6HV-VECTOR) 00530
ISH 0489  750  FORMAT (1X,6E12.4) 00531
ISH 0490  760  FORMAT (/5X,5OBJ =,E13.5) 00532
ISH 0491  770  FORMAT (/5X,17HCONSTRAINT VALUES) 00533
ISH 0492  780  FORMAT (/5X,23HTWO-POINT INTERPOLATION) 00534
ISH 0493  790  FORMAT (/5X,35H* * END OF ONE-DIMENSIONAL SEARCH) 00535
ISH 0494  800  END 00536

*OPTIONS IN EFFECT*NANE(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOCDL(NONE)
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NOHAP NOTOFMAT GOSTMT NOREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 493, PROGRAM SIZE = 6746, SUBPROGRAM NAME =CIN606
*STATISTICS* NO DIAGNOSTICS GENERATED

****** END OF COMPILATION ******

2924K BYTES OF CORE NOT USED
SUBROUTINE CMN107 (II,XBAR,EPS,X1,Y1,X2,Y2,X3,Y3)

ROUTINE TO FIND FIRST XBAR.GE.EPS CORRESPONDING TO A REAL ZERO OF A ONE-DIMENSIONAL FUNCTION BY POLYNOMIAL INTERPOLATION.

BY G. N. VANDERPLAATS APRIL, 1972.
NASA AMES RESEARCH CENTER, JOFFETT FIELD, CALIF.

II = CALCULATION CONTROL.
1: 2-POINT LINEAR INTERPOLATION, GIVEN X1, Y1, X2 AND Y2.
2: 3-POINT QUADRATIC INTERPOLATION, GIVEN X1, Y1, X2, Y2, X3 AND Y3.

EPS MAY BE NEGATIVE.

IF REQUIRED ZERO ON Y DOES NOT EXISTS, OR THE FUNCTION IS ILL-CONDITIONED, XBAR = EPS+1.0 WILL BE RETURNED AS AN ERROR INDICATOR.

IF DESIRED INTERPOLATION IS ILL-CONDITIONED, A LOWER ORDER INTERPOLATION, CONSISTANT WITH INPUT DATA, WILL BE ATTEMPTED AND II WILL BE CHANGED ACCORDINGLY.

II = 1:

II = 1: 2-POINT LINEAR INTERPOLATION

II = 2:

II = 2: 3-POINT QUADRATIC INTERPOLATION

C DATA SET U47CN107 AT LEVEL 001 AS OF 02/13/81
C DATA SET 916CN107 AT LEVEL 001 AS OF 07/10/80

C SUBROUTINE CMN107 (II,XBAR,EPS,X1,Y1,X2,Y2,X3,Y3)
C ROUTINE TO FIND FIRST XBAR.GE.EPS CORRESPONDING TO A REAL ZERO OF A ONE-DIMENSIONAL FUNCTION BY POLYNOMIAL INTERPOLATION.
C BY G. N. VANDERPLAATS APRIL, 1972.
C NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.
C II = CALCULATION CONTROL.
C 1: 2-POINT LINEAR INTERPOLATION, GIVEN X1, Y1, X2 AND Y2.
C 2: 3-POINT QUADRATIC INTERPOLATION, GIVEN X1, Y1, X2, Y2, X3 AND Y3.
C EPS MAY BE NEGATIVE.
C IF REQUIRED ZERO ON Y DOES NOT EXISTS, OR THE FUNCTION IS ILL-CONDITIONED, XBAR = EPS+1.0 WILL BE RETURNED AS AN ERROR INDICATOR.
C IF DESIRED INTERPOLATION IS ILL-CONDITIONED, A LOWER ORDER INTERPOLATION, CONSISTANT WITH INPUT DATA, WILL BE ATTEMPTED AND II WILL BE CHANGED ACCORDINGLY.
C II = 1:

C II = 1: 2-POINT LINEAR INTERPOLATION

C II = 2:

C II = 2: 3-POINT QUADRATIC INTERPOLATION

C CONTINUE
ISH 0040  BB=(Y2-Y1)/X21-AA*(X1+X2)
ISH 0041  CC=Y1-X1*(AA*X1+BB)
ISH 0042  BAC=BB*BB-4.*AA*CC
ISH 0043  IF (BAC.LT.0.) GO TO 10
ISH 0045  BAC=SQRT(BAC)
ISH 0046  AA=.5/AA
ISH 0047  XBAR=AA*(BAC-BB)
ISH 0048  XB2=-AA*IBAC+BB
ISH 0049  IF (XBAR.LT.EPS) XBAR=XB2
ISH 0051  IF (XB2.LT.XBAR.AND.XB2.GT.EPS) XBAR=XB2
ISH 0053  IF (XBAR.LT.EPS) XBAR=XBAR1
ISH 0055  RETURN
ISH 0056  END

*OPTIONS IN EFFECT*NAME(CHMN07) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODATE OBJECT NOLOAD NOFORMAT GOSTMT NOXREF NOXREF NOXREF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 55, PROGRAM SIZE = 936, SUBPROGRAM NAME =CHMN07
*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION ******
3916K BYTES OF CORE NOT USED
SUBROUTINE CINN08 (NOB,NER,C,MS1,B,N3,N4,N5) 00002
DIMENSION C(N4), B(N3,N3), MS1(N5) 00003
ROUTINE TO SOLVE SPECIAL LINEAR PROBLEM FOR IMPOSING S-TRANSPOSE TIMES S.LE.1 BOUNDS IN THE MODIFIED METHOD OF FEASIBLE DIRECTIONS.00005
BY G. N. VANDERPLAATS APRIL, 1972.00006
NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.00007
U AND LAST NOB COMPONENTS OF X CONTAIN VECTOR U AND V. CONSTRAINTS ARE U.GE.0, V.GE.0. AND U-TRANSPOSE TIMES V = 0.0011
HER = ERROR FLAG. IF NER.NE.0 ON RETURN, PROCESS HAS NOT CONVERGED IN 5*NOB ITERATIONS.00013
VECTOR MS1 IDENTIFIES THE SET OF BASIC VARIABLES.00015
------------------------------------------------------------------00017
CHOOSE INITIAL BASIC VARIABLES AS V, AND INITIALIZE VECTOR MSI.00019
------------------------------------------------------------------00020
NER=1 00020
M2=2*NOB 00021
C = 
CALCULATE CBMIN AND EPS AND INITIALIZE MSI.00022
EPS=-1.0E+10 00023
CBMIN=0. 00024
DO 10 I=1,NOB 00025
BI=B(I,I) 00026
CBMAX=0. 00027
IF (BI.LT.-1.0E-6) CBMAX=C/I/BI 00028
IF (BI.GT.EPS) EPS=BI 00029
IF (CBMAX.GT.CBMIN) CBMIN=CBMAX 00030
M51(I)=0 00031
EPS=.0001*EPS 00032
IF (EPS.LT.-1.0E-10) EPS=-1.0E-10 00033
IF (EPS.GT.-.0001) EPS=-.0001 00034
CBMIN=CBMIN*1.0E-6 00035
CBMAX=CBMAX*1.0E-10 00036
ITER1=0 00037
NMAX=5*NOB 00038
------------------------------------------------------------------00039
************** BEGIN NEW ITERATION **************00040
------------------------------------------------------------------00041
IF (ITER1.GT.NMAX) RETURN 00043
C = 
FIND MAX. C(I)/BI1(I) FOR I=1,NOB.00044
CBMAX=.9*CBMIN 00045
ICHK=0 00046
DO 30 I=1,NOB 00047
CI=C(I) 00048
BI1=BI1(I) 00049
IF (BI1.GE.0.) GO TO 30 00050
IF (CI.GE.0.) GO TO 30 00051
IF (CBMAX.GE.0.) GO TO 30 00052
ICHK=I 00053
CBMAX=CB 00054
ISH 0043  30 CONTINUE
ISH 0044  IF (CBMAX.LT.CBMIN) GO TO 70
ISH 0046  IF (ICHK.EQ.0) GO TO 70
C UPDATE VECTOR MS1.
ISH 0048  JJ=ICHK
ISH 0049  IF (S(J).EQ.0) JJ=ICHK+NOB
ISH 0051  KK=JJ+NOB
ISH 0052  IF (KK.GT.M2) KK=JJ-NOB
ISH 0054  JJ=ICHK
ISH 0055  JJ=ICHK
C --------------------------------------------
C PIVOT OF B(ICHK,ICHK) 
C --------------------------------------------
ISH 0056  BB=1./B(ICHK,ICHK)
ISH 0057  DO 40 J=1,NOB
ISH 0058  B(ICHK,J)=BB*B(ICHK,J)
ISH 0059  C(ICHK)=CBMAX
ISH 0060  B(ICHK,ICHK)=BB
C ELIMINATE COEFFICIENTS ON VARIABLE ENTERING BASIS AND STORE
C COEFFICIENTS ON VARIABLE LEAVING BASIS IN THEIR PLACE.
ISH 0061  DO 60 I=1,NDB
ISH 0062  IF (I.EQ.ICHK) GO TO 60
ISH 0064  BB1=B(I,ICHK)
ISH 0065  B(I,ICHK)=O.
ISH 0066  DO 50 J=1,NDB
ISH 0067  B(I,J)=B(I,J)-BB1*B(ICHK,J)
ISH 0068  C(I)=C(I)-BB1*CBMAX
ISH 0069  60 CONTINUE
ISH 0070  GO TO 20
ISH 0071  70 CONTINUE
ISH 0072  NER=O
C -------------------------------------------------------------
C STORE ONLY COMPONENTS OF U-VECTOR IN 'C'. USE B(I,1) FOR
C TEMPORARY STORAGE 
C -------------------------------------------------------------
ISH 0073  DO 80 I=1,NDB
ISH 0074  B(I,1)=C(I)
ISH 0075  80 CONTINUE
ISH 0076  DO 90 I=1,NDB
ISH 0077  C(I)=O.
ISH 0078  J=HS(I)
ISH 0079  IF (J.GT.0) C(I)=B(J,1)
ISH 0081  IF (C(I).LT.0.) C(I)=O.
ISH 0083  90 CONTINUE
ISH 0084  RETURN
ISH 0085  END

*OPTIONS IN EFEFF=NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOBL(NONE)
*OPTIONS IN EFFECT=SOURCE EBCDIC NOLIST NODECK OBJECT NOFORMAT GOSTH T NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 64, PROGRAM SIZE = 1350, SUBPROGRAM NAME =CN:~1108
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPIILATION *****

3004K BYTES OF CORE NOT USED
SUBROUTINE CONMIN (X,VLB,VUB,G,SCAL,DF,A,S,G1,G2,B,C,ISC,IC,MS)

C      COMMON /COMIN/ DELLUN, DABFUN, FDCH, FDCHM, CT, CTMIN, CTL, CTLMIN, ALPHAX
1, ABOBJJ, THEBA, OBJ, NCOND, NSIDE, IPRINT, NDF, NSCAL, LINOBJ, ITMAX, IT0
0, ERF, DCMIR, IGOTO, HAC, INFO, INFOG, ITER

C      COMMON /UOVS/ ISI, I6I

C      DIMENSION X(N1), VLB(N1), VUB(N1), G(N1), SCAL(N1), DF(N1), A(N1), N00004
13), S(N1), G1(N1), G2(N1), B(N3,N3), C(N4), ISC(N2), IC(N3), M5(N00011
25)

C      COMMON /CONSAV/ DH1, DH2, DM1, DM2, DM3, DM4, DM5, DM6, DM7, DM9, DM9, DM10, DM11, DM1200006
0, DCT, DCTL, PHA1, ABOBJ, CT, CTM, CTB, OBJJ, SLOPE, DX, DN1, F1, F2, DFTDF1, A00014
2, LP, FEF, Al, A2, A3, A4, F1, F2, F3, F4, CV1, CV1, CV3, CV4, APP, ALPC, ALPFS, A00015
3, 3PLN, ALPHMIN, ALPHNC, ALPSAV, ALPSID, ALPPTOT, RSPACE, IC:1, IGD, IHD3, JDIF, I00016
0, OBJ, KOSJ, KCOUNT, KHALO(2), NFEAS, NSCAL, NCOND, NVCOUNT, ICOUNT, IG00008
0, SIGOOD2, IG0004, IBEST, I3I, NUNC, JGOTO, IPSAVE(2)

C      ROUTINE TO SOLVE CONSTRAINED OR UNCONSTRAINED FUNCTION

C      MINIMIZATION.

C      BY G. N. VANDERPLAATS APRIL, 1972.

C      *************** JUNE, 1979 VERSION ***************

C      NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.

C      REFERENCE: CONMIN - A FORTRAN PROGRAM FOR CONSTRAINED FUNCTION
C      MINIMIZATION: USER'S MANUAL, BY G. N. VANDERPLAATS,

C      STORAGE REQUIREMENTS:
C      PROGRAM - 7000 DECIMAL WORDS (CDC COMPUTER)
C      ARRAYS - APPROX. 2*(NOV**2)+26*NOV+4
C      WHERE NOV = NDV+2.

C      RE-SCALE VARIABLES IF REQUIRED.

C      IF (NSCAL.EQ.0.OR.IGOTO.EQ.0) GO TO 20

C      DD 10 I=1,NDV

C      IF (OBJ.GT.-1.0E+40) GO TO 30

C      CONTINUE

C      STOP IF OBJ IS LESS THAN -1.0E+40

C      IF (OBJGT.-1.0E+40) GO TO 30

C      WRITE (I6I,900)

C      GO TO 010

C      WRITE (I6I,900)

C      CONTINUE

C      CONTINUE

C      IF (LINOBJ.EQ.0.OR.(NCON.GT.0.OR.NSIDE.GT.0)) GO TO 50
C TOTALLY UNCONSTRAINED FUNCTION WITH LINEAR OBJECTIVE.
C SOLUTION IS UNBOUNDED.

I0027 WRITE (161,970) LINOBJ,NCON,NSIDE
I0028 RETURN
I0029 50 CONTINUE
I0030 IDM1=ITRM
I0031 IDM2=ITHAX
I0032 IDM3=ICHDIR
I0033 DM1=DELFUN
I0034 DM2=DABFUN
I0035 DM3=CT
I0036 DM4=CTMIN
I0037 DM5=CTL
I0038 DM6=CTLMIN
I0039 DM7=THETA
I0040 DM8=PHI
I0041 DM9=FDCH
I0042 DM10=FDCHM
I0043 DM11=ABOBJ1
I0044 DM12=ALPHAX
C DEFAULTS
C
I0045 IF (ITRML.LE.0) ITRM=3
I0046 IF (ITHAX.LE.0) ITHAX=20
I0049 NDV1=NDV+1
I0050 IF (ICHDIR.EQ.0) ICHDIR=NDV1
I0052 IF (DELFUN.LE.0.) DELFUN=.0001
I0054 CT=-ABS(CT)
I0055 IF (CT.GE.0.) CT=-.1
I0057 CTMIN=ABS(CTMIN)
I0058 IF (CTMIN.LE.0.) CTMIN=.004
I0060 CTL=-ABS(CTL)
I0061 IF (CTL.GE.0.) CTL=.01
I0063 CTLMIN=ABS(CTLMIN)
I0064 IF (CTLMIN.LE.0.) CTLMIN=.001
I0066 IF (THETA.LE.0.) THETA=1.
I0068 IF (ABOBJ1.LE.0.) ABOBJ1=.1
I0070 IF (ALPHAX.LE.0.) ALPHAX=.1
I0072 IF (FDCH.LE.0.) FDCH=.01
I0074 IF (FDCHM.LE.0.) FDCHM=.01
C INITIALIZE INTERNAL PARAMETERS
C
I0076 INFOG=O
I0077 ITR=O
I0078 JDIR=O
I0079 IOBJ=O
I0080 KOBJ=O
I0081 KLEV=KLEV+2
I0082 KCOUNT=O
I0083 NCA=1
I0084 NCA(2)=O
I0085 NAC=O
I0086 NFEAS=O
I0087 Mscal=NSCAL
I0088 CT1=ITRM
I0089 CT1=1./CT1
I0090 DCT=CTMIN/ABS(CT)**CT1
C  CALCULATE NUMBER OF LINEAR CONSTRAINTS, NLNC.

C  CHECK TO BE SURE THAT SIDE CONSTRAINTS ARE SATISFIED

C  INITIALIZE SCALING VECTOR, SCAL
C ****** CALCULATE INITIAL FUNCTION AND CONSTRAINT VALUES ******
C ***************************************************************
C
ISH 0151  INFO=1  00175
ISH 0152  NCAL(1)=1  00176
ISH 0153  IGO=1  00177
ISH 0154  GO TO 950  00178
ISH 0155  160 CONTINUE  00179
ISH 0156  OBJ=OBJ  00180
ISH 0157  IF (DABFUN.LE.0.) DABFUN=-.001*ABS(OBJ)  00181
ISH 0159  IF (DABFUN.LT.1.0E-10) DABFUN=1.0E-10  00182
ISH 0161  IF (IPRINT.LE.0.) GO TO 270  00183
C   ------------------------------------------------------------------
C   PRINT INITIAL DESIGN INFORMATION
C   ------------------------------------------------------------------
C
ISH 0163  IF (IPRINT.LE.1) GO TO 230  00184
ISH 0165  IF (NSIDE.EQ.0.AND.NCON.EQ.0) WRITE (I61,1290)  00185
ISH 0167  IF (NSIDE.NE.0.OR.NCON.GT.0) WRITE (I61,1290)  00186
ISH 0169  WRITE(I61,1240)IPRINT,ITMAX,NSIDE,ICNDR,NSCAL,NDGBL,LINOBO190
1J,ITRM,NI,N5,N4,N3,N2,N1  00187
ISH 0170  WRITE (I61,1260) CT,CMIN,CTL,CMIN,THETA,PHI,DELFUN,DABFUN  00188
ISH 0171  WRITE (I61,1250) FOCII,FCHM,ALPHAX,ABOBJ  00189
ISH 0172  IF (ITLST.EQ.O) GO TO 190  00190
ISH 0174  WRITE (I61,1270)  00191
ISH 0175  DO 170 I=1,NDV,6  00192
ISH 0176  M1=MINOINDV.I+5)  00193
ISH 0177  WRITE (I61,1280)  00194
ISH 0178  WRITE (I61,1280)  00195
ISH 0179  DO 180 I=1,NDV,6  00196
ISH 0180  M1=MINOINDV.I+5)  00197
ISH 0181  WRITE (I61,1290)  00198
ISH 0182  CONTINUE  00199
ISH 0183  IF (NSCAL.GE.0) GO TO 200  00200
ISH 0185  WRITE (I61,1300)  00201
ISH 0186  WRITE (I61,1460) (SCAL(I),I=1,NDV)  00202
ISH 0187  WRITE (I61,1420)  00203
ISH 0188  CONTINUE  00204
ISH 0189  IF (NCON.EQ.0) GO TO 230  00205
ISH 0190  IF (NLNC.EQ.0.OR.NLNC.EQ.NCON) GO TO 220  00206
ISH 0192  WRITE (I61,1020)  00207
ISH 0193  DO 210 J=1,NCON,15  00208
ISH 0194  M1=MINOINCON.I+14)  00209
ISH 0195  WRITE (I61,1030)  00210
ISH 0196  GO TO 230  00211
ISH 0197  WRITE (I61,1040)  00212
ISH 0199  IF (NLNC.EQ.0) WRITE (I61,1050)  00213
ISH 0201  CONTINUE  00214
ISH 0202  WRITE (I61,1440) OBJ  00215
ISH 0203  WRITE (I61,1450)  00216
ISH 0204  DO 240 J=1,NDV  00217
ISH 0205  XI=I.  00218
ISH 0206  IF (NSCAL.NE.0) XI=SCAL(I)  00219
ISH 0208  240 GI(I)=XI(I)*XI  00220
ISH 0209  DO 250 J=1,NDV,6  00221
ISH 0210  M1=MINOINDV.I+5)  00222
ISH 0211  WRITE (I61,1010) I.GI(I),J=I,M1)  00223
ISH 0212  IF (NCON.EQ.0) GO TO 270  00224
ISH 0214  WRITE (I61,1470)  00225
ISH 0215  DO 260 J=1,NCON,6  00226
ISH 0216  M1=MINOINCON.I+5)  00227
ISH 0217  WRITE (I61,1010) I,(GI(J),J=I,M1)  00228
*VERSION 1.3.0  (01 MAY 80) CONMIN  SYSTEM/370 FORTRAN IV EXTENDED (ENHANCED)  DATE 82.141/10.53.56  PAGE 5

**C** CONTINUE 00210
**C** IF (IPRINT.GT.1) WRITE (I6,I,1360) 00211

**C** BEGIN MINIMIZATION

**C** CONTINUE 00220

**C** ITER=ITER+1 00221

**C** IF (ABOBJ.LT. .0001) ABOBJ= .0001 00222

**C** IF (ABOBJ.GT. .2) ABOBJ= .2 00223

**C** IF (ALPHAX.LT. .001) ALPHAX= .001 00224

**C** IF (ALPHAX.GT. .001) ALPHAX= .001 00225

**C** IF (IPRINT.GT.2) WRITE (I6,I,1310) ITER 00226

**C** IF (IPRINT.GT.3.AND.KCON.GT.0) WRITE (I6,I,1320) CT, CTL, PHI 00227

**C** CT=ABS(CT) 00228

**C** IF (NOOBJ.EQ.0) GO TO 340 00229

**C** NO MOVE ON LAST ITERATION. DELETE CONSTRAINTS THAT ARE NO LONGER ACTIVE.

**C** CONTINUE 00230

**C** NAC=NAC 00231

**C** DO 310 I = 1, NAC 00232

**C** SCAL(I) = SI 00233

**C** XI=SI*XII 00234

**C** SIB=SI 00235

**C** IF (INSCAL.GT.0) SI=ABSXI 00236

**C** IF (ISI.LT.1.0E-10) GO TO 310 00237

**C** CONTINUE 00238

**C** SCALE VARIABLES

**C** CONTINUE 00240

**C** SCALE VARIABLES

**C** WRITE 1161,1360) ITER 00241

**C** WRITE 1161,1320) CT, CTL, PHI 00242

**C** WRITE 1161,1330) KCON 00243

**C** WRITE 1161,1340) IPRINT 00244

**C** WRITE 1161,1350) IABOBJ 00245

**C** WRITE 1161,1360) ALPHAX 00246

**C** WRITE 1161,1370) ITSOLV 00247

**C** WRITE 1161,1380) ITSOLV 00248

**C** WRITE 1161,1390) ITSOLV 00249

**C** WRITE 1161,1400) ITSOLV 00250

**C** WRITE 1161,1410) ITSOLV 00251

**C** WRITE 1161,1420) ITSOLV 00252

**C** WRITE 1161,1430) ITSOLV 00253

**C** WRITE 1161,1440) ITSOLV 00254

**C** WRITE 1161,1450) ITSOLV 00255

**C** WRITE 1161,1460) ITSOLV 00256

**C** WRITE 1161,1470) ITSOLV 00257

**C** WRITE 1161,1480) ITSOLV 00258

**C** WRITE 1161,1490) ITSOLV 00259

**C** WRITE 1161,1500) ITSOLV 00260

**C** WRITE 1161,1510) ITSOLV 00261

**C** WRITE 1161,1520) ITSOLV 00262

**C** WRITE 1161,1530) ITSOLV 00263

**C** WRITE 1161,1540) ITSOLV 00264

**C** WRITE 1161,1550) ITSOLV 00265

**C** WRITE 1161,1560) ITSOLV 00266

**C** WRITE 1161,1570) ITSOLV 00267

**C** WRITE 1161,1580) ITSOLV 00268

**C** WRITE 1161,1590) ITSOLV 00269

**C** WRITE 1161,1600) ITSOLV 00270

**C** WRITE 1161,1610) ITSOLV 00271

**C** WRITE 1161,1620) ITSOLV 00272

**C** WRITE 1161,1630) ITSOLV 00273

**C** WRITE 1161,1640) ITSOLV 00274

**C** WRITE 1161,1650) ITSOLV 00275

**C** WRITE 1161,1660) ITSOLV 00276

**C** WRITE 1161,1670) ITSOLV 00277

**C** WRITE 1161,1680) ITSOLV 00278

**C** WRITE 1161,1690) ITSOLV 00279
ISH 0294 350 CONTINUE 00295
ISH 0295 IF (IPRT.LT.4.0.R.(NSCAL.LT.0.AND.ITER.GT.1)) GO TO 360 00296
ISH 0296 WRITE (6,1330) 00297
ISH 0297 WRITE (6,1460) (SCAL(I),I=1,NDV) 00298
ISH 0299 360 CONTINUE 00299
ISH 0300 NSCAL=MSCAL+1 00301
ISH 0301 NAC=0 00302

C OBTAIN GRADIENTS OF OBJECTIVE AND ACTIVE CONSTRAINTS 00303
C------------------------------------------------------------------00304
C OBTAIN GRADIENTS OF OBJECTIVE FUNCTION. 00305
DO 400 I=1,NDV 00306
400 DF(I)=DF(I)*SCAL(I) 00307
IF (INFDG.EQ.2.0.R.NAC.EQ.0) GO TO 420 00308
C SCALE GRADIENTS OF ACTIVE CONSTRAINTS. 00309
DO 410 J=1,NDV 00310
SCJ=SCAL(J) 00311
DO 410 I=1,HAC 00312
AIJ,II=AIJ,II*SCJ 00313
410 AIJ,II 00314
420 CONTINUE 00315

C SCALE GRADIENTS 00316
C------------------------------------------------------------------00317
C SCALE GRADIENT OF OBJECTIVE FUNCTION. 00318
DO 440 J=1,NDV 00319
440 DF(J)=DF(J)*SCAL(J) 00320
C SCALE GRADIENTS OF ACTIVE CONSTRAINTS. 00321
DO 470 J=1,NDV 00322
SCJ=SCAL(J) 00323
DO 470 I=1,HAC 00324
AIJ,II=AIJ,II*SCJ 00325
470 AIJ,II 00326
420 CONTINUE 00327

C PRINT ACTIVE AND VIOLATED CONSTRAINT NUMBERS. 00328
H1=0 00329
M1=0 00330
M2=N3 00331
IF (NAC.EQ.0) GO TO 450 00332
DO 440 I=1,HAC 00333
440 I 00334
450 CONTINUE 00335

C ACTIVE CONSTRAINT. 00336
M1=M1+1 00337
M1(M1)=J 00338
**Versión 1.3.0 (01 May 80) Conmin System/370 Fortran H Extended (Enhanced) Date 02.141/10.53.56 Page 7**

**ISP 0341**
GO TO 440 00350

**ISP 0342**
430 M2=M2+1 00351

**ISP 0343**
C VIOLATED CONSTRAINT. 00352

**ISP 0344**
M3=M2+1 00353

**ISP 0345**
CONTINUE 00354

**ISP 0346**
440 CONTINUE 00354

**ISP 0347**
450 N3=N2+1 00355

**ISP 0348**
WRITE (161,1060) M1 00356

**ISP 0349**
IF (M1.EQ.0) GO TO 460 00357

**ISP 0350**
WRITE (161,1070) M3 00358

**ISP 0351**
WRITE (161,1480) (M1(I),I=1,M1) 00359

**ISP 0352**
WRITE (161,1080) M3 00360

**ISP 0353**
WRITE (161,1480) (M3(I),I=1,M3) 00361

**ISP 0354**
CONTINUE 00362

**ISP 0355**
------------------------------------------------------------------ 00363
**ISP 0356**
C CALCULATE GRADIENTS OF ACTIVE SIDE CONSTRAINTS 00364

**ISP 0357**
IF (HSIDE.EQ.0) GO TO 530 00365

**ISP 0358**
MCI=MCI+1 00366

**ISP 0359**
M1=0 00367

**ISP 0360**
DO 510 I=1,HDV 00368

**ISP 0361**
C LOWER BOUND. 00369

**ISP 0362**
XI=X(I) 00370

**ISP 0363**
XID=VLB(I) 00371

**ISP 0364**
X12=ABS(XID) 00372

**ISP 0365**
IF (X12.LT.1.) X12=1. 00373

**ISP 0366**
GI=(XID-XI)/X12 00374

**ISP 0367**
IF (GI.LT.-1.0E-6) GO TO 490 00375

**ISP 0368**
HAC=HAC+1 00376

**ISP 0369**
IF (GI.LT.-1.0E-6) GO TO 510 00377

**ISP 0370**
A(J,HAC)=0. 00378

**ISP 0371**
A(I,HAC)=1. 00379

**ISP 0372**
IC(NAC)=NCH 00380

**ISP 0373**
IC(NCH1)=1 00381

**ISP 0374**
IC(MCN1)=GI 00382

**ISP 0375**
ISC(MCH1)=1 00383

**ISP 0376**
ISC(MCH1)=1 00384

**ISP 0377**
ISP 0378**
DO 480 J=1,NDV 00385

**ISP 0379**
480 A(J,HAC)=0. 00386

**ISP 0380**
A(I,HAC)=1. 00387

**ISP 0381**
IC(NAC)=MCN1 00388

**ISP 0382**
G(MCN1)=GI 00389

**ISP 0383**
ISC(MCH1)=1 00390

**ISP 0384**
ISC(MCH1)=1 00391

**ISP 0385**
CONTINUE 00392

**ISP 0386**
------------------------------------------------------------------ 00393
**ISP 0387**
C UPPER BOUND. 00394

**ISP 0388**
490 XID=VUB(I) 00395

**ISP 0389**
X12=ABS(XID) 00396

**ISP 0390**
IF (X12.LT.1.) X12=1. 00397

**ISP 0391**
G(I1)=GI 00398

**ISP 0392**
G(I1)=GI 00399

**ISP 0393**
IF (XI.GE.N3) GO TO 810 00400

**ISP 0394**
IF (XI.GE.N3) GO TO 810 00401

**ISP 0395**
MCI=MCI+1 00402

**ISP 0396**
A(I,HAC)=0. 00403

**ISP 0397**
A(I,HAC)=1. 00404

**ISP 0398**
IC(NAC)=MCN1 00405

**ISP 0399**
IC(NAC)=MCN1 00406

**ISP 0400**
G(MCN1)=GI 00407

**ISP 0401**
ISC(MCH1)=1 00408

**ISP 0402**
CONTINUE 00409

**ISP 0403**
------------------------------------------------------------------ 00410
**ISP 0404**
C CALCULATE GRADIENTS OF ACTIVE SIDE CONSTRAINTS 00411

**ISP 0405**
------------------------------------------------------------------ 00412
**ISP 0406**
C UPPER BOUND. 00413

**ISP 0407**
------------------------------------------------------------------ 00414
**ISP 0408**
C CALCULATE GRADIENTS OF ACTIVE SIDE CONSTRAINTS 00415

**ISP 0409**
------------------------------------------------------------------ 00416
**ISP 0410**
C UPPER BOUND. 00417

**ISP 0411**
------------------------------------------------------------------ 00418
**ISP 0412**
C CALCULATE GRADIENTS OF ACTIVE SIDE CONSTRAINTS 00419

**ISP 0413**
------------------------------------------------------------------ 00420
**ISP 0414**
C UPPER BOUND. 00421

**ISP 0415**
------------------------------------------------------------------ 00422
C PRINT ACTIVE SIDE CONSTRAINT NUMBERS.

C ---~--------------------------------------------------------------00411
C PRINT GRADIENTS OF ACTIVE AND VIOLATED CONSTRAINTS. 00412
C IF (IPRINT.LT.3) GO TO 530 00413
C WRITE (161,1090) HI 00414
C IF (H1.EQ.0) GO TO 530 00415
C WRITE (161,1100) I-IRITE!I6I,1480) (HSl(Jl.J=I,Hll 00416
C 530 CONTINUE 00417
C C PRINT GRADIENTS OF ACTIVE AND VIOLATED CONSTRAINTS. 00419
C IF (IPRINT.LT.4) GO TO 570 00420
C WRITE (161,1340) 00421
C DO 540 I=I,HDV,6 00422
C M=MINO!tlDV,I+5) 00423
C 540 WRITE (161,1010) I,(DF(J),J=I,Ml) 00424
C IF (HAC.EQ.0) GO TO 570 00425
C WRITE (161,1350) 00426
C DO 560 I=l,HAC 00427
C Hl=IC(I) 00428
C H2=HI-HCON 00429
C H3=0 00430
C IF (H2.GT.0) H3=IABS(HSl(H2)) 00431
C IF (H2.LE.0) WRITE (161,990) HI 00432
C IF (H2.GT.0) WRITE (161,1000) H3 00433
C DO 550 K=l,NOV,6 00434
C M=MINO(HOV,K+5) 00435
C 550 WRITE (161,1010) K,(A(J,I),J=K,Ml) 00436
C 560 WRITE (161,1360) 00437
C 570 CONTINUE 00438
C C *************** DETERMINE SEARCH DIRECTION *******************00440
C C ALP=1.0E+20 00442
C IF (NAC.GT.O) GO TO 580 00443
C C UNCONSTRAINED FUNCTION 00445
C C FIND DIRECTION OF STEEPEST DESCENT OR CONJUGATE DIRECTION. 00447
C NVC=O 00448
C NFEAS=O 00449
C KCOUNT=KCOUNT+1 00450
C IF KCOUNT.GT.ICNDIR RESTART CONJUGATE DIRECTION ALGORITHM. 00451
C IF (KCOUNT.GT.ICNDIR.OR.IGOBJ.EQ.2) KCOUNT=l 00452
C IF (KCOUNT.EQ.1) JDIR=O 00453
C IF JDIR = 0 FIND DIRECTION OF STEEPEST DESCENT. 00454
C CALL CHMN02 (JOIR,SLOPE,DF,TDF1,DF,S,N1) 00455
C GO TO 630 00456
C 630 CONTINUE 00457
C C FIND USABLE-FEASIBLE DIRECTION. 00461
C KCOUNT=O 00462
C JDIR=O 00463
C CALL CHMN05 (G,DF,AS,B,C,SLOPE,PHI,ISC,IC,HSl,NVC,N1,N2,N3,N4,N5)00467
IF (IPRINT.LT.3) GO TO 660
WRITE (161,1380) 00525
DO 650 I=1,NDV,6

------------------------------------------------------------------
C *************** ONE-DIMENSIONAL SEARCH **************************
C------------------------------------------------------------------
C
IF (S(NDV1).LT.1.0E-6.AND.NVC.EQ.0) GO TO 710
C
------------------------------------------------------------------
C
IF (HVC.EQ.0) GO TO 630
ALP=-1.
DO 620 I=1,NAC
NCI=IC(I)
Cl=G(NCI)
CTC=CTAM
IF (ISC(Cl).GT.0) CTC=CTBM
IF (Cl.LE.CTC) GO TO 620
ALP1=0.
DO 610 J=I,NDV
ALP1=ALP1+S(J)*A(J,I)
ALP1=ALP1*A(NDV2,I)
IF (ABS(ALP1).LT.1.0E-20) GO TO 620
ALP1=-Cl/ALP1
IF (ALP1.GT.ALP) ALP=ALP1
620 CONTINUE
630 CONTINUE
C------------------------------------------------------------------
C
LIMIT CHANGE TO ADJBJ*OBJ
C------------------------------------------------------------------
ALP1=1.0E+20
DO 640 I=1,NDV
SI=ABS(S(I))
XI=ABS(X(I))
IF (SI.LT.1.0E-10.0R.XI.LT.0.1) GO TO 640
ALP1=ALPHAX*XI/SI
IF (ALP1.LT.1.0E+20) ALP1=ALPHAX*XI/SI
IF (ALP1.LE.1.0E-20) ALP1=1.0E-20
IF (IPRINT.LT.3) GO TO 660
WRITE (161,1380) 00525
DO 650 I=1,NDV,6

------------------------------------------------------------------
C LIMIT CHANGE IN VARIABLE TO ALPHAX
C------------------------------------------------------------------
ALP11=1.0E+20
DO 660 I=1,NDV
XI=ABS(X(I))
IF (SI.LT.1.0E-10.0R.XI.LT.0.1) GO TO 660
ALP1=ALPHAX*XI/SI
IF (ALP1.LT.1.0E+20) ALP1=ALPHAX*XI/SI
IF (ALP1.LE.1.0E-20) ALP1=ALPHAX*XI/SI
IF (IPRINT.LT.3) GO TO 660
WRITE (161,1380) 00525
DO 650 I=1,NDV,6
H1=M1=MINO(NDV,IXS) 00527
WRITE (161,1010) I,(S(J),J=I,NI) 00528
WRITE (161,1110) SLOPE,ALP 00529
CONTINUE 00530
IF (NCON.GT.0.OR.NSIDE.GT.0) GO TO 680 00531
C DO ONE-DIMENSIONAL SEARCH FOR UNCONSTRAINED FUNCTION 00532
C 00533
CONTINUE 00534
WRITE (161,1010) I,(S(J),J=I,NI) 00535
WRITE (161,1110) SLOPE,ALP 00536
CONTINUE 00537
CALL CN2003 (X,SLOPE,ALP,FFF,A1,A2,A3,A4,F1,F2,F3,F4,APP,N1,NCAL) 00538
1,KOUNT,JGOTO) 00539
C 00540
CONTINUE 00541
CALL CN2006 (X,VLB,VUB,G,SCAL,DF,S,G,GI,GT,CTAM,CTBM,SLOPE,ALP,A2,A3,A4,F1,F2,F3,F4, 00542
APP,N1,NCAL,G,COUNT,GGOOD1,GGOOD2,GGOOD3,GGOOD4,GGOOD5) 00543
C 00544
CONTINUE 00545
JGOTO=4 00546
IF (JGOTO.GT.0) GO TO 950 00547
JDIR=1 00548
C PROCEED TO CONVERGENCE CHECK. 00549
GO TO 700 00550
C 00551
C**************************** UPDATE ALPHAX ************************** 00552
C 00553
CONTINUE 00554
DO 720 I=1,NDV 00555
SI=ABS(S(I)) 00556
XI=ABS(X(I)) 00557
IF (XI.LT.1.0E-10) GO TO 720 00558
ALP1=ALP*SI/XI 00559
IF (ALP1.GT.ALPI1) ALPI1=ALP1 00560
ALPHAX=ALPI1 00561
NCOBJ=NCOBJ+1 00562
ABSOLUTE CHANGE IN OBJECTIVE. 00563
OBJO=OBJ1-JOB 00564
OBJB=ABS(OBJO) 00565
IF (OBJB.LT.1.0E-10) OBJB=0. 00566
IF (NAC.EQ.0) JDIR=1 00567
IF (OBJB.GT.0.) NCOBJ=0 00568
IF (NCOBJ.GT.1) NCOBJ=0 00569
C 00570
CEND OF MAIN PROGRAM 00571
C 00572
*VERSION 1.3.0 (01 MAY 80) COMMENTS SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 02.141/10.53.56 PAGE 11

C FRINT MOVE PARAMETER, NEW X-VECTOR AND CONSTRAINTS.

ISN 0570 IF (IPRINT.LT.3) GO TO 730

ISN 0572 WRITE (161,1590) ALP

ISN 0573 730 IF (IPRINT.LT.2) GO TO 800

ISN 0575 IF (OBJB.GT.0.) GO TO 740

ISN 0577 IF (IPRINT.EQ.2) WRITE (161,1400) ITER,OJB

ISN 0579 IF (IPRINT.GT.2) WRITE (161,1410) OBJ

ISN 0581 GO TO 760

ISN 0582 740 IF (IPRINT.EQ.2) GO TO 750

ISN 0584 WRITE (161,1420) OBJ

ISN 0585 GO TO 760

ISN 0586 750 WRITE (161,1430) ITER

ISN 0587 760 WRITE (161,1440)

ISN 0588 DO 770 I=1,NDV

ISN 0589 FF1=1.

ISN 0590 IF (INSCAL.NE.0) FF1=SCAL(i)

ISN 0591 DO 780 I=1,NDV,6

ISN 0592 M1=MIHO(NDV,I+5)

ISN 0593 WRITE (161,1010) I,(G(i,j)),J=I,M1

ISN 0594 M1=MENO(NDV,I+5)

ISN 0595 780 WRITE (161,1010) I,(G(i,j)),J=I,M1

ISN 0596 IF (NCON.EQ.0) GO TO 800

ISN 0598 DO 790 I=1,NCON,6

ISN 0599 H1=MIHO(NCON,I+5)

ISN 0600 WRITE (161,1010) I,(H(i,j)),J=I,M1

ISN 0601 CONTINUE

C CHECK FEASIBILITY

C IF(NCON.LE.0) GO TO 808

ISN 0603 DO 804 I=1,NCON

ISN 0605 CI=CTAM

ISN 0606 CI=CTBM

ISN 0607 IF(G(i,j).LE.CI) GO TO 804

ISN 0609 NFEAS=NFEAS+1

ISN 0611 GO TO 806

ISN 0612 CONTINUE

ISN 0613 804 IF(NFEAS.GT.0) ABOBJ1=.05

ISN 0614 NFEAS=0

ISN 0615 PHI=5.

ISN 0616 IF(HFEAS.GE.10) GO TO 810

ISN 0617 CONTINUE

ISN 0618 806 IF(NFEAS.GT.10) GO TO 810

ISN 0619 CONTINUE

C CHECK CONVERGENCE

C IF(ITER.GE.ITMAX) GO TO 810

ISN 0620 CONTINUE

C STOP IF ITER EQUALS ITMAX.

ISN 0621 IF (ITER.GE.ITMAX) GO TO 810

C ABSOLUTE CHANGE IN OBJECTIVE

ISN 0623 OBJ=ABS(OBJ1)

ISN 0624 KOBJ=KOBJ+1

ISN 0625 IF (OBJB.GE.DABFUN.OR.NFEAS.GT.0) KOBJ=0

ISN 0627 IF (ABS(OBJ1).GT.1.0E-10) OBJD=OBJD/ABS(OBJ1)

ISN 0628 CONTINUE

ISN 0629 ABOBJ1=.5*(ABS(ABOBJ)+ABS(OBJ1))
(01 MAY 80) CONMIN SYSTEH FORTRAN  H EXTENDED (ENHANCED) DATE 82.141/10.53.56

**C**

```
ABOBJ=ABS(OBJD)

IF(OBJJ+OBJJ+1) GO TO 810
IF(OBJJ.0 OR.OBJJ.GE.DELFUH) OBJJ=0
IF (OBJJ.OF.OBJJ.GE.0) GO TO 810
OBJJ=OBJJ
```

**C** REDUCE CT IF OBJECTIVE FUNCTION IS CHANGING SLOWLY

```
CT=CT*CT
CTL=CTL*DCTL
IF(ABS(CT).LT.CTHIN) CT=-CTMIN
IF(ABS(CTL).LT.CTLMIN) CTL=-CTLMIN
```

**C**

```
810 CONTINUE
```

**C**

```
820 C
830 C
840 C
850 C
860 C
870 C
```

**C** UN-SCALE THE DESIGN VARIABLES.

```
DO 820 I=1,HDV
XI=SCAL(I)
IF(NSIDE.EQ.0) GO TO 820
VLB(I)=XI*VLB(I)
VUB(I)=XI*VUB(I)
X(I)=XI*X(I)
```

**C**

```
820 X(I)=XI*X(I)
```

**C**

```
830 IF (IPRINT.EQ.0.OR.NAC.GE.N3) GO TO 940
WRITE (161,1500)
WRITE (161,1420) OBJ
WRITE (161,1450)
DO 840 I=1,NDV,6
M1=INO(NDV,1+5)
WRITE (161,1010)
I,(X(J),J=I,H1)
```

**C** DETERMINE WHICH CONSTRAINTS ARE ACTIVE AND PRINT.

```
NAC=0
NVC=0
DO 870 I=1,NCON
CTA=CTAH
IF (ISC(I).GT.O) CTA=CTBH
GI=GI
IF (GI.GT.CTA) GO TO 860
IF (GI.LT.CTA.AND.ISC(I).EQ.O) GO TO 870
IF (GI.LT.CTA.AND.ISC(I).GT.O) GO TO 870
NAC=NAC+1
ICIN./;C)=I
NVC=NVC+1
MS1CNVC)=I
CCNTItlUE
WRITE (161,1060) NAC
```
ISH 0692 IF (NAC.EQ.0) GO TO 880 00704
ISH 0694 WRITE (IGI,1070) 00705
ISH 0695 WRITE (IGI,1480) (IC(J),J=1,NAC) 00706
ISH 0696 WRITE (IGI,1080) NVC 00707
ISH 0697 IF (NVC.EQ.0) GO TO 890 00708
ISH 0699 WRITE (IGI,1480) (HS1(J),J=1,NVC) 00710
ISH 0701 880 WRITE (161,1080) NAC 00707
ISH 0702 900 CONTINUE 00712
ISH 0703 IF (INSIDE.EQ.0) GO TO 930 00713

C DETERMINE WHICH SIDE CONSTRAINTS ARE ACTIVE AND PRINT.
ISH 0705 NAC=0 00714
ISH 0706 DO 920 I=1,NOV 00716
ISH 0707 XI=X(I) 00717
ISH 0708 XID=VUB(I) 00718
ISH 0709 X12=ABS(XID) 00719
ISH 0710 IF (X12.LT.1.) XI2=1. 00720
ISH 0712 GI=(XID-XI2)/X12 00721
ISH 0713 IF (GI.LT.-1.0E-6) GO TO 910 00722
ISH 0715 NAC=NAC+1 00723
ISH 0716 MS1(NAC)=-1 00724
ISH 0717 910 CONTINUE 00725
ISH 0718 WRITE (161,1090) NAC 00726
ISH 0719 IF (NAC.EQ.0) GO TO 930 00734
ISH 0720 WRITE (161,1100) 00727
ISH 0721 WRITE (161,1480) (HS1(J),J=1,NAC) 00736
ISH 0722 930 CONTINUE 00737
ISH 0723 WRITE (161,1150) 00738
ISH 0724 DO 920 I=1,NOV 00739
ISH 0725 XI=X(I) 00740
ISH 0726 XID=VUB(I) 00741
ISH 0727 X12=ABS(XID) 00742
ISH 0728 IF (X12.LT.1.) XI2=1. 00743
ISH 0729 GI=(XID-XI2)/X12 00744
ISH 0730 IF (GI.LT.-1.0E-6) GO TO 920 00745
ISH 0731 NAC=NAC+1 00746
ISH 0732 MS1(NAC)=-1 00747
ISH 0733 WRITE (161,1160) NAC 00748
ISH 0734 IF (NAC.EQ.0) GO TO 930 00749
ISH 0735 WRITE (161,1170) 00750
ISH 0736 WRITE (161,1180) 00751
ISH 0737 WRITE (161,1190) 00752
ISH 0738 WRITE (161,1200) 00753
ISH 0739 WRITE (161,1210) 00754
ISH 0740 WRITE (161,1220) 00755
ISH 0741 WRITE (161,1230) 00756
ISH 0742 WRITE (161,1240) 00757
ISH 0743 WRITE (161,1250) 00758
ISH 0744 WRITE (161,1260) 00759
ISH 0745 WRITE (161,1270) 00760
ISH 0746 WRITE (161,1280) 00761
ISH 0747 WRITE (161,1290) 00762

C ------------------------------------------------------------------
C RE-SET BASIC PARAMETERS TO INPUT VALUES
C ------------------------------------------------------------------
ISH 0750 940 ITRM=IDM1 00751
ISH 0751 ITMAX=IDM2 00752
ISH 0752 IC301=IDM3 00753
ISH 0753 DELFUN=DM1 00754
ISH 0754 DABFUN=DM2 00755
ISH 0755 CT=DM3 00756
ISH 0756 CTMIN=DM4 00757
ISH 0757 CTL=DM5 00758
ISH 0758 CTMIN=DM6 00759
ISH 0759 THETA=DM7 00760
ISH 0760 FHI=DM8 00761
ISH 0761 FDCH=DM9 00762
CONTINUE 0076
C IF (.NOT.ISCAL.EQ.0.OR.IGOTO.EQ.0) RETURN 0077
DO 950 I=1,NDV 0078
C(I)=X(I) 0079
RETURN 0080
C --------------------------------------------------------------- 0081
C FORMATS 0082
C ------------------------------------------------------------------ 0083
C 0084
C 0085
C 0086 FORMAT (//5X,17HCOMPLETELY UNCONSTRAINED FUNCTION WITH A 0087
S LINEAR OBJECTIVE IS SPECIFIED///10X,8HOBJ =,E12.4,8X,15HCON =,E12.4,60X,15H 0088
2HSIDE =,E15.15,35HCONTROL RETURNED TO CALLING PROGRAM) 0089
C 0090 FORMAT (//5X,6HCONMIN HAS ACHIEVED A SOLUTION OF OBJ LESS THAN 0091
11.0E+40/5X,32HAN OPTIMIZATION TERMINATED) 0092
C 0093 FORMAT (//5X,26HALL CONSTRAINTS ARE LINEAR) 0094
C 0095 FORMAT (//5X,24HALL CONSTRAINTS ARE NON-LINEAR) 0096
C 0097 FORMAT (//5X,23HACTIVE CONSTRAINTS) 0098
C 0099 FORMAT (//5X,24HACTIVE SIDE CONSTRAINTS) 0099
C 0100 FORMAT (//5X,21HDECISION VARIABLES AT LOWER OR UPPER BOUND) 0101
C 0102 FORM3E (//5X,20HINITIAL SLOPE =,E12.4,2X,8HAFTER PROPOSED ALPHA =,E12.4) 0103
C 0104 FORMAT (//5X,20HSET XI=VUBL/I) 0105
C 0106 FORMAT (//5X,20HSET XII=VUBL/I) 0107
C 0108 FORMAT (//5X,20HSET XII=VUBL/I)
DATA SET U477COPE01 AT LEVEL 003 AS OF 05/17/81 0001
DATA SET U477COPE01 AT LEVEL 002 AS OF 05/17/81 0002
DATA SET U477COPE01 AT LEVEL 001 AS OF 05/13/81 0003
DATA SET 9146COPE01 AT LEVEL 001 AS OF 07/10/80 0004
SUBROUTINE COPE01 (RA1,IA,NRA,INDIA) 0005

C ROUTINE TO READ CONTROL INPUT FOR COPES. 0006
****************************************************************** 0007
BY G. H. VANDERPLAATS HAR. 1973. 0008
NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF. 0009
------------------------------------------------------------------ 0010
READ CARD IMAGES AND STORE ON UNIT ISCR2. STORE ON UNIT ISCR1 0011
WITHOUT COMMENT CARDS 0012
------------------------------------------------------------------ 0013
READ (I5,I,5801 IRA(I),I=1,801 0014
ICARD=ICARD+1 0015
NCARDS=NCARDS+1 0016
WRITE (ISCR2,5901 I,NCARDS,IRA(I),I=1,80) 0017
IF IRA(I).EQ.COHI GO TO 10 0018
IF IRA(I).EQ.END1.AND.IRA(I21).EQ.END2.AHD.RAI31).EQ.END31) GO TO 20 0019
IF INCHG.NE.OI GO TO 30 0020
TITLE OR END CARD. 0021
WRITE (ISCR1,5801 IRA(I),I=1,801 0022
IF INCHG.GT.OI GO TO 70 0023
IT WAS THE TITLE CARD. 0024
CONTINUE 0025
FOR DATA AS REQUIRED. 0026
C FORMAT DATA AS REQUIRED. 0027
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0028
C CALL COPE03 (RA(NA),RA(NB),IFORM,NFLD) 0029
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0030
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0031
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0032
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0033
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0034
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0035
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0036
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0037
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0038
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0039
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0040
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0041
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0042
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0043
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0044
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0045
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0046
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0047
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0048
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0049
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0050
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0051
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0052
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0053
C DETERMINE NUMBER OF CARDS OF DATA AND ADD BLANKS TO FILL. 0054
*VERSION 1.3.0 (01 MAY 80) COPE01 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 02.161/10.54.12  PAGE 2

**ISN 0040**
DO 40 I=NA,NB 00055

**ISN 0041**
40 RA(I)=BLANK 00056

**ISN 0042**
50 CONTINUE 00057

**ISN 0043**
WRITE (ISCR1,580) (RA(I),I=81,NB) 00058

**ISN 0044**
IF (IFORM.GT.0) GO TO 10 00059

**ISN 0045**
C DATA WAS NOT PREVIOUSLY FORMATTED. 00060

**ISN 0046**
N1=1 00061

**ISN 0047**
DO 60 II=1,NBC 00062

**ISN 0048**
N1=N1+80 00063
N2=N2+79 00064

**ISN 0049**
WRITE (ISCR2,590) NCARDS,(RA(Il),I=H1,N2) 00065

**ISN 0050**
60 ICARD=ICARD+1 00066

**ISN 0051**
GO TO 10 00067

**ISN 0052**
CONTINUE 00068

**ISN 0053**
REWIND ISCR1 00069

**ISN 0054**
REWIND ISCR2 00070

**ISN 0055**
------------------------------------------------------------------00071

**ISN 0056**
C GENERAL SYNTHESIS INFORMATION 00072

**ISN 0057**
C TITLE. 00073

**ISN 0058**
C --- DATA BLOCK A. 00074

**ISN 0059**
READ (ISCR1,1190) (ATITLE(I),I=1,20) 00075

**ISN 0060**
C CONTROL PARAMETERS. 00076

**ISN 0061**
C --- DATA BLOCK B. 00077

**ISN 0062**
READ (ISCR1,1200) HCALC,NOV,HSV,N2VAR,HXAPRX,IPINPUT,IPDBG 00078

**ISN 0063**
IF (HCALC.LT.0.OR.NCALC.GT.61) WRITE 1161,1220) HCALC 00079

**ISN 0064**
IF (HCALC.LT.0.OR.NCALC.GT.61) RETURN 00080

**ISN 0065**
IF (IPINPUT.GT.1) GO TO 100 00081

**ISN 0066**
WRITE (161,970) 00082
WRITE (161,980) 00083
WRITE (161,990) (ATITLE(I),I=1,20) 00084

**ISN 0067**
------------------------------------------------------------------00085

**ISN 0068**
C CARD IMAGE PRINT 00086

**ISN 0069**
C --- DATA BLOCK C. 00087

**ISN 0070**
READ (ISCR1,1210) IPRIHT,ITHAX,ICNOIR,NSCAL,ITR1,LTLINE,NACMX1,NFDOO10

**ISN 0071**
C --- DATA BLOCK D. 00088

**ISN 0072**
READ (ISCR2,590) NCARDS,(RA(J),J=1,80) 00089

**ISN 0073**
WRITE (161,890) NCARDS,(RA(J),J=1,80) 00090

**ISN 0074**
REWIND ISCR2 00091

**ISN 0075**
CONTINUE 00092

**ISN 0076**
WRITE (161,900) (ATITLE(I),I=1,20) 00093

**ISN 0077**
WRITE (161,901) (ATITLE(I),I=1,20) 00094

**ISN 0078**
WRITE (161,910) (ATITLE(I),I=1,20) 00095

**ISN 0079**
100 NACMX1=0 00096

**ISN 0080**
NCHOT=0 00097

**ISN 0081**
NCNOA=0 00098

**ISN 0082**
NCNO=0 00099

**ISN 0083**
IF (NCH.LE.0) GO TO 270 00100

**ISN 0084**
------------------------------------------------------------------00101

**ISN 0085**
C OPTIMIZATION INFORMATION 00102

**ISN 0086**
C OPTIMIZATION CONTROL VARIABLES. - CQM IN DEPENDENT. 00103

**ISN 0087**
C --- DATA BLOCK E. 00104

**ISN 0088**
READ (ISCR1,1210) IPRIH,ITMAX,ICNDIR,NISCAL,ITRM,LINOBJ,NACMX1,NFDOO10

**ISN 0089**
IG 00105

**ISN 0090**
C --- DATA BLOCK F. 00106

**ISN 0091**
READ (ISCR1,1210) FDCH,FDCHM,CT,CMIN,CTL,CTLM,THETA,PHI,DELFIN,00107

**ISN 0092**
------------------------------------------------------------------00108

**ISN 0093**
------------------------------------------------------------------00109

**ISN 0094**
------------------------------------------------------------------00110

**ISN 0095**
------------------------------------------------------------------00111

**ISN 0096**
------------------------------------------------------------------00112
C --- DATA BLOCK E.
C TOTAL NO. OF D. V. , OBJECTIVE GLOBAL NUMBER, SIGN
C ON OPTIMIZATION OBJECTIVE.

I$N 0087 READ (ISCR1,920) NDVTOT,IOBJ,SGNOPT
I$N 0088 IF (NDVTOT.LT.LDVTOT) NDVTOT=LDVTOT
I$N 0090 IF (HCALC.EQ.6.AND.HACMXI.EQ.0) HACMXI=2*NDVTOT+2
I$N 0092 IF (HACMXI.LT.0) HACMXI=NDVTOT+2
I$N 0094 IF (MPUT.GE.6.2) GO TO 110
I$N 0096 IF (ABS(SGNOPT).LT.1.0E-10) SGNOPT=1.
I$N 0098 WRITE (1,10170) IOBJ,SGNOPT
I$N 0099 WRITE (1,10170) IOBJ,SGNOPT
I$N 1000 WRITE (1,10170) IOBJ,SGNOPT
U$N 0101 N2=NDVTOT+3
U$N 0102 N3=N2+NDVTOT+2
U$N 0103 N4=N3+NDVTOT+2
C --- DATA BLOCK F.
C DESIGN VARIABLE INFORMATION, LB, UB, INITIAL VALUE, SCAL.
I$N 0104 IF (IPUT.LT.2) WRITE (1,10270) 0
I$N 0105 N5=N4+NDVTOT+2
I$N 0106 NS=NDVTOT+1
I$N 0107 WRITE (1,10270)
I$N 0108 WRITE (1,10270)
U$N 0109 WRITE (1,10270)
U$N 0110 WRITE (1,10270)
U$N 0111 LOC(J)=N5
U$N 0112 120 CONTINUE
U$N 0113 NSIDE=0
U$N 0114 DO 130 I=1,NDV
U$N 0115 NSIDE=NSIDE+1
U$N 0116 READ (ISCR1,960) RAIN2),RAIN3),RAIN4),RAIN4),RAIN4),RAIN4),RAIN4),RAIN4),RAIN4),RAIN4)
U$N 0117 IF (RAIN2).GT.1.0E+15.OR.RAIN3).LT.-1.0E+15) NSIDE=1
U$N 0118 IF (RAIN2).LE.-1.0E+15) RAIN2)=-1.0E+15
U$N 0119 IF (RAIN3).GT.1.0E+15) RAIN3)=1.0E+15
U$N 0120 WRITE (1,10270)
U$N 0121 IF (MPUT.LT.2) WRITE (1,10270)
I$N 0122 N5=NS+1
I$N 0123 N6=NS+1
I$N 0124 N7=NS+1
I$N 0125 N8=NS+1
I$N 0126 130 CONTINUE
I$N 0127 NSIDE=0
I$N 0128 130 CONTINUE
C --- DATA BLOCK G.
C D. V. NO., GLOBAL LOCATION, MULTIPLYING FACTOR.
I$N 0129 IF (IPUT.LT.2) WRITE (1,1030)
I$N 0130 NS=NS+1
I$N 0131 N5=NS+1
I$N 0132 MZ=NDVTOT+1
I$N 0133 N5=NS+NDVTOT
I$N 0134 M3=M2+NDVTOT
I$N 0135 IF (N6.LE.NDRA) GO TO 140
I$N 0136 WRITE (1,1030)
I$N 0137 WRITE (1,1030)
I$N 0138 WRITE (1,1030)
I$N 0139 LOC(J)=N5
I$N 0140 GO TO 550
I$N 0141 140 CONTINUE
I$N 0142 IF (M3.LE.NDRA) GO TO 150
I$N 0143 WRITE (1,1030)
I$N 0144 WRITE (1,1030)
I$N 0145 WRITE (1,1030)
I$N 0146 LOC(J)=M3
I$N 0147 GO TO 550
I$N 0148 150 CONTINUE
I$N 0149 GO 160 I=1,NDVTOT

--- End of Natural Text ---
C --- DATA BLOCK H.
C NUMBER OF CONSTRAINT SETS.
ISN 0159 READ (ISCR1,920) NCONS 00182
ISN 0160 IF (INPUT.LT.2) WRITE (161,1110) NCONS 00183
ISN 0164 IF (NCONS.EQ.0) GO TO 270 00184
ISN 0165 H2=H2+1 00176
ISN 0166 N5=N5+1 00177
ISN 0168 CONTINUE 00178
ISN 0158 NCONS=0 00179
C --- DATA BLOCK I.
C NUMBER OF VARIABLES IN THIS SET.
ISN 0191 NVAR=NCONI-ICOHI+1 00207
ISN 0192 IF (NVAR.LT.1) NVAR=1 00208
ISN 0194 NCONI=NCONI+NVAR 00209
C HOW MANY CONSTRAINTS?
ISN 0195 J1=0 00210
ISN 0196 IF (RA(N6).LE.-1.0E+15) J1=1 00212
ISN 0197 IF (RA(N6+2).GE.1.0E+15) J1=J1+1 00213
ISN 0200 NCONI=J1+NVAR 00214
ISN 0201 NCONI=NCONI+NCONI 00215
ISN 0202 IF (J1.EQ.0) GO TO 160 00216
C ADD LINEAR CONSTRAINT IDENTIFIERS TO ISC.
ISN 0205 DO 170 J=1,NVAR1 00218
ISN 0206 N4=N4+1 00219
ISN 0207 IF (M4+1.ME.0) GO TO 260 00220
ISN 0209 170 IA(N4)=ICONI 00222
ISN 0210 CONTINUE 00223
C ADD LB, UB AND SCAL TO BLU IF NVAR.GT.1.
ISN 0211 IF (NVAR.EQ.1) GO TO 200 00224
ISN 0214 DO 190 J=1,NVAR 00225
ISN 0215 N5=N5+1 00226
ISN 0216 IF ((M5+1.GE.0)GO TO 250 00227
ISN 0218 RA(N5+1)=RA(N5) 00228
ISN 0219 RA(N5+5)=RA(N5+1) 00229
ISH 0220  RA(NS+6)=RA(NS+2)
ISH 0221  RA(NS+7)=RA(NS+3)
ISH 0222  NS=NS+4
ISH 0223  190  CONTINUE
ISH 0224  200  CONTINUE
C ADD CONSTRAINED VARIABLE GLOBAL IDENTIFIERS TO ICON.
ISH 0225  ICON=ICON1
ISH 0226  MNS=M4+NVAR-1
ISH 0227  IF (MM.MT.GT.NDIA) GO TO 260
ISH 0228  DO 230  J=1,NVAR
ISH 0229  IF (J.EQ.1) GO TO 220
C SHIFT ISC VECTOR.
ISH 0230  LI=M4+1
ISH 0231  L2=M4
ISH 0232  DO 210  K=M4,M4+1
ISH 0233  IAI(L1)=IAI(L2)
ISH 0234  L1=L1-1
ISH 0235  210  L2=L2-1
ISH 0236  H4=1+1
ISH 0237  H4A=M4ATL
ISH 0238  IF (INPUT.LT.2) WRITE 1161,1100) J,ICON,JCON,RA(NS5),RA(N6+1)
ISH 0239  N6=N6+4
ISH 0240  L=TICON+1
ISH 0241  240  CONTINUE
ISN 0242  C STARTING LOCATIONS FOR APPROXIMATION INFORMATION.
ISH 0243  NAPR=4*NVTOT+4*NCONA+9
ISH 0244  NAPI=2*(NV+NVFIN+1)+NCONA+1
ISH 0245  NF=0
ISH 0246  MAXTRM=0
ISH 0247  IF (INPUT.LT.2) WRITE (1161,600) NF,NPS,NPFS,NPA,INOH,ISCRX,ISCRXF
ISH 0248  270  CONTINUE
C CONTROL PARAMETERS.
ISH 0249  READ (ISCR1,1200) NF,NPS,NPFS,NPA,INOH,ISCRX,ISCRXF,IPAPRX
ISH 0250  IF (NPME.NE.0) NPA=1
ISH 0251  IF (NPS.EQ.0.AND.NPS.EQ.0) NPA=1
ISH 0252  IF (ISCRX.EQ.0.AND.ISCRX.EQ.0) ISCRX=5
ISH 0253  IF (ISCRX.EQ.0.AND.ISCRXF.EQ.0) ISCRXF=5
ISH 0254  IF (INPUT.LT.2) WRITE (1161,600) NF,NPS,NPFS,NPA,INOH,ISCRX,ISCRXF,IPAPRX
**Version 1.3.0 (01 May 80)**

**Cope01**

**System/370 Fortran H Extended (Enhanced)**

**Date 82.141/10.54.12**

**Page 6**

**ISN 0279**

**NPSF=NPS+HPFS**

**ISH 0280**

**IF (NPSF.LT.2) NPA=1**

**ISH 0282**

**NPTOT=NPS+HPFS+NPA**

**ISH 0285**

**IF (NPTOT.LT.2) HPTOT=2**

**ISH 0286**

**READ (ISCR1,1200) KMAX,KMAX,NPMAX,JNOM,INXLOC,INFLOC,MAXTRM**

**ISH 0288**

**IF (INXLOC.EQ.0) NKAPRX=NDV**

**ISH 0289**

**H=NXAPRX+1**

**ISH 0291**

**IF (KMAX.EQ.0) KMAX=3+H-NPTOT+1**

**ISH 0293**

**IF (KMIN.EQ.0) KMIN=2*NDV-HPTOT+1**

**ISH 0295**

**IF (KMIN.LT.0) KMIN=0**

**ISH 0297**

**IF (KMAX.GT.0) KMAX=KMIN**

**ISH 0299**

**IF (JNOM.EQ.0) JNOM=2*M**

**ISH 0301**

**IF (MAXTRM.LT.1) MAXTRM=3**

**ISH 0303**

**IF (IPHPUT.LT.2) WRITE (161,610) KHIN,KMAX,NPMAX,JNOM,INXLOC,INFLOC,MAXTRM**

C --- DATA BLOCK K, PART 1.

**ISH 0305**

**IF (NDV.LE.0) GO TO 290**

**ISH 0307**

**N7=NDV**

**ISH 0308**

**NH7=N7+NDV-1**

**ISH 0309**

**IF (NH7.LE.NDIA) GO TO 280**

**ISH 0311**

**WRITE (161,780) I=H7,HH7**

**ISH 0312**

**WRITE (161,560)**

**ISH 0313**

**CONTINUE**

**ISH 0314**

**READ (ISCR1,1210) (RAII),I=H7,HH7**

**ISH 0315**

**IF (IPHPUT.LT.2) WRITE (161,970) I=H7,HH7**

**ISH 0317**

**C --- DATA BLOCK K, PART 2.**

**C DELX BOUNDS ON APPROXIMATE OPTIMIZATION.**

**ISH 0317**

**C --- DATA BLOCK K, PART 2.**

**C --- DATA BLOCK K, PART 2.**

**C DELX BOUNDS ON APPROXIMATE OPTIMIZATION.**

**ISH 0317**

**C --- DATA BLOCK L.**

**C --- DATA BLOCK L.**

**C GLOBAL LOCATIONS OF X-VARIABLES.**

**ISH 0319**

**READ (ISCR1,1210) XFAC1,XFACT2**

**ISH 0320**

**IF (XFAC1.LT.1.0E-10) XFAC1=1.5**

**ISH 0322**

**IF (XFAC2.LT.1.0E-10) XFAC2=2.0**

**ISH 0324**

**IF (IPHPUT.LT.2) WRITE (161,620) XFAC1,XFACT2**

**ISH 0326**

**CONTINUE**

**ISH 0327**

**C --- DATA BLOCK L.**

**C --- DATA BLOCK L.**

**C GLOBAL LOCATIONS OF X-VARIABLES.**

**ISH 0328**

**MS=MS1**

**ISH 0329**

**MS=MS+NXAPRX-1**

**ISH 0331**

**WRITE (161,630)**

**ISH 0332**

**WRITE (161,630)**

**ISH 0333**

**LOC1(25)=MS**

**ISH 0334**

**GO TO 550**

**ISH 0335**

**CONTINUE**

**ISH 0336**

**IF (INXLOC.EQ.0) GO TO 310**

**ISH 0338**

**READ (ISCR1,1200) (IAII),I=MS,MS5**

**ISH 0339**

**GO TO 330**

**ISH 0340**

**CONTINUE**

**ISH 0341**

**X-LOCATIONS ARE DEFAULTED TO DESIGN VARIABLE LOCATIONS.**

**ISH 0342**

**GO 320 I=1,NXAPRX**

**ISH 0343**

**IA(MS)=IAII**

**ISH 0344**

**MS=NAPI**

**ISH 0345**

**CONTINUE**

**ISH 0346**

**IF (IPHPUT.LT.2) WRITE (161,640) (IAII),I=MS,MS5**

**ISH 0348**

**IF (IPHPUT.LT.2) WRITE (161,1160) (IAII),I=MS,MS5**

**ISH 0349**

**C --- DATA BLOCK M.**

**C --- DATA BLOCK M.**

**C GLOBAL LOCATIONS OF FUNCTIONS.**

**ISH 0349**

**C DATA BLOCK M.**

**C DATA BLOCK M.**

**C GLOBAL LOCATIONS OF FUNCTIONS.**
C FUNCTION LOCATIONS ARE DEFAULTED TO OBJECTIVE AND CONSTRAINT LOCATIONS.

NF1 = 1
M3 = M3 + 1

C --- DATA BLOCK N.
C READ INPUT X-VECTORS AND STORE ON UNIT ISCR2.

REND I SCR2
IF (NPS.EQ.0) GO TO 410
N7 = NAP + NDV
NN7 = N7 + NXAPRX
IF (NN7.LE.NN7) NN7 = NN7 + HF - 1
WRITE (I6I,780)
WRITE (I6I,670)
LOCIR(25) = NN7
GO TO 550
IF (INPFL.T.LT.2) WRITE (I6I,660)
IF (INPFL.T.LT.2) WRITE (I6I,1180)
C --- DATA BLOCK O.
C READ INPUT X-F PAIRS AND STORE ON UNIT ISCR2.

REND ISCR2
IF (NPS.EQ.0) GO TO 440
N7 = NAP + NDV
NN7 = N7 + NXAPRX
IF (NN7.LE.NN7) NN7 = NN7 + HF - 1
WRITE (I6I,780)
WRITE (I6I,670)
LOCIR(25) = NN7
GO TO 550
IF (INPFL.T.LT.2) WRITE (I6I,660)
IF (NN8.LE.NN7) GO TO 420
WRITE (161,780) 00410
WRITE (161,690) 00411
LOC(25)=IN8 00412
GO TO 550 00413
CONTINUE 00414
NN8=NN7+NF-1 00415
IF (IPNPUT.LT.2) WRITE (161,7001 ISCRXF 00416
DO 430 I=1,NPFS 00417
C X-VECTOR.
READ IF (ISCRXF.NE.5, 00418
IF (ISCRXF.NE.5) READ (ISCRXF) (RA(J),J=N7,NN7) 00419
C FORTRAN READ IF (ISCRXF.EQ.5, 00420
IF (ISCRXF.EQ.5) READ (ISCR1,1210) (RA(J),J=N7,NN7) 00421
II=I+NPS 00422
IF (IPNPUT.LT.2) WRITE (161,710) II 00423
IF (IPNPUT.LT.2) WRITE (161,720) II 00424
WRITE IISCR2) (RA(J).J=N7.NN7) 00425
C FUNCTION VALUES.
WRITE IISCR3) (RA(J).J=N7.NN7) 00426
CONTINUE 00427
CONTINUE 00428
CONTINUE 00429
NSOBJ=O 00430
NSVTOT=O 00431
C STARTING LOCATIONS FOR SENSITIVITY INFORMATION. 00432
NSVR=NPFR+NFV 00433
NSVI=NPFR+NHAPRX+NF 00434
IF (NSV.LE.0) GO TO 500 00435
C ------------------------------------------------------------------ 00436
C SENSITIVITY INFORMATION 00437
C ------------------------------------------------------------------ 00438
IF (IPNPUT.LT.2) WRITE (161,1020) NSOBJ,IPSENS 00439
C --- DATA BLOCK P, PART 1. 00440
C NSOBJ, IPSENS 00441
READ (ISCR1,1200) NSOBJ,IPSENS 00442
C --- DATA BLOCK P, PART 2. 00443
C NSSENS.
M15=NSVI 00444
M15=NSJ+NSOBJ-1 00445
IF (M15.LE.NDIA) GO TO 460 00446
WRITE (161,810) 00447
WRITE (161,830) 00448
LOC(25)=HM15 00449
GO TO 550 00450
CONTINUE 00451
CONTINUE 00452
CONTINUE 00453
CONTINUE 00454
CONTINUE 00455
CONTINUE 00456
CONTINUE 00457
CONTINUE 00458
CONTINUE 00459
CONTINUE 00460
CONTINUE 00461
CONTINUE 00462
CONTINUE 00463
CONTINUE 00464
CONTINUE 00465
CONTINUE 00466
CONTINUE 00467
+VERSION 1.3.0 (01 MAY 80) COPE01 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 02.141/10.54.12  PAGE 9

ISH 0478  M17=M16+HSV  00465
ISH 0479  GO 490 I=1,HSV  00466
C --- DATA BLOCK Q., PART 1.
C ISENS, NSENS.  00470

ISH 0480  READ (ISCR1,1200) I,IA(M16),NN1  00471
ISH 0481  NN1S=NN1S+NN1-1  00473
ISH 0482  IF (NN1S.LE.NDRA) GO TO 470  00474
ISH 0484  WRITE (I61,780)  00475
ISH 0485  WRITE (I61,640)  00476
ISH 0486  LOC(R25)=NN1S  00477
ISH 0487  GO TO 550  00478
ISH 0488  470 CONTINUE  00479
C --- DATA BLOCK Q., PART 2.
C SESS.

ISH 0489  READ (ISCR1,1210) (RA(J),JJ)=NN1S,NN1S  00480
ISH 0490  IF (INPUT.GE.2) GO TO 480  00481
ISH 0492  JJ=NN1S  00482
ISH 0493  IF (J.J.GT.NN1S) JJ=NN1S  00483
ISH 0495  WRITE (I61,1040) I,IA(M16),(RA(J),J=NN1S,JJ)  00484
ISH 0496  JJ=JJ+1  00485
ISH 0497  IF (J.J.LE.NN1S) WRITE (I61,1050) (RA(J),J=JJ,NN1S)  00486
ISH 0499  460 CONTINUE  00487
ISH 0500  NSVTOT=NSVTOT+NN1S  00488
ISH 0501  IAA(M17)=NN1S  00489
ISH 0502  N1S=NN1S+1  00490
ISH 0503  M16=M16+1  00491
ISH 0504  M17=M17+1  00492
ISH 0505  490 CONTINUE  00493
ISH 0506  500 CONTINUE  00494
ISH 0507  M2VX=0  00495
ISH 0508  M2VY=0  00496
C STARTING LOCATIONS FOR TWO-VARIABLE FUNCTION SPACE INFORMATION.

ISH 0509  N2VR=NSVR+NSVTOT-1  00497
ISH 0510  N2VI=NSVI+NSOBJ+2*NSV  00498
ISH 0511  IF (N2VAR.LE.0) GO TO 540  00499
C --- DATA BLOCK R.
C VARIABLE NUMBERS AND NUMBER OF VALUES OF X AND Y.

ISH 0513  READ (ISCR1,1200) N2XHR,M2XHR,N2XVR,M2XVR,IP2VAR  00500
ISH 0514  N20=N2VR  00501
ISH 0515  M20=M2VR  00502
ISH 0516  M20=M20+M2VAR-1  00503
ISH 0517  IF (M20.LE.NDRA) GO TO 510  00504
ISH 0519  WRITE (I61,610)  00505
ISH 0520  WRITE (I61,650)  00506
ISH 0521  LOC(R25)=M20  00507
ISH 0522  GO TO 550  00508
ISH 0523  510 CONTINUE  00509
C --- DATA BLOCK S.
C GLOBAL VARIABLE NUMBERS CORRESPONDING TO FUNCTIONS OF X AND Y.

ISH 0524  READ (ISCR1,1200) (IA(I),I=1,M20,NN20)  00510
ISH 0525  IF (INPUT.LT.2) WRITE (I61,1170) IP2VAR  00511
ISH 0527  IF (INPUT.LT.2) WRITE (I61,1180) (IA(I),I=1,M20,NN20)  00512
C --- DATA BLOCK T.
C VALUES OF X COMPONENTS.

ISH 0529  NN20=N20+M2XVR-1  00513
ISH 0530  IF (NN20.LE.NDRA) GO TO 520  00514
C -- DATA BLOCK U.
C VALUES OF Y COMPONENTS.
N21=N20+N2VX
NN21=N21+H2VY-1
IF (NN21.LE.NDRA) GO TO 530
WRITE (161,780)
WRITE (161,750)
LOCR(25)=NN21
GO TO 550
530 CONTINUE
IN20=NN21
READ (ISCR1,1210) (RA(I),I=N21,NN21)
IF (IPNPUT.LT.2) WRITE (161,1150) N2VY
IF (IPNPUT.LT.2) WRITE (161,1140) IRA(I,I=N21,NN21)
540 CONTINUE
C ------------------------------------------------------------------
C DYNAMIC STORAGE ALLOCATION
C ------------------------------------------------------------------
NDV2=NDV+2
REAL VARIABLES.
X.

LOCR(1)=1
VBL.
LOCR(2)=NDV+3
VUB.
LOCR(3)=LOCR(2)+NDV2
SCAL.
LOCR(4)=LOCR(3)+NDV2
AJUMP.
LOCR(5)=LOCR(4)+NDV2
XBL.
LOCR(6)=LOCR(5)+NDVTOT
DELX.
LOCR(7)=LOCR(6)+MCNAG
LOCR(8)=LOCR(7)+NDV
SENS.
LOCR(15)=LOCR(8)
XMV.
LOCR(20)=LOCR(15)+NSVTOT
YM2V.
LOCR(21)=LOCR(20)+NSVTOT
567
LOCR(21)=LOCR(20)+NSVTOT
LOCR(22)=LOCR(21)+NSVTOT
START OF EXECUTION STORAGE.
LOCR(23)=LOCR(22)
INTEGER VARIABLES.
IDSGN.
LOC(1)=1
HDSGN.
LOC(2)=NDVTOT+1
ICON.
LOC(3)=LOC(2)+NDVTOT
ISC.
ISH 0574  LOC(4)=LOC(3)+NCONA
ISH 0575  LOC(5)=LOC(4)+2*(NDV+NCONA)
ISH 0576  LOC(6)=LOC(5)+NXAPRX
ISH 0577  LOC(7)=LOC(6)+HF
ISH 0578  NSERNZ.
ISH 0579  LOC(16)=LOC(15)+NSOBJ
ISH 0580  LOC(17)=LOC(16)+NV
ISH 0581  LOC(18)=LOC(17)+NV
ISH 0583  START OF EXECUTION STORAGE.
ISH 0584  LOC(23)=LOC(21)
ISH 0585  EXECUTION STORAGE REQUIREMENTS.
ISH 0586  NRI=NDV
ISH 0587  IF (NACMX1.GT.NRI) NRI=NACMX1
ISH 0588  NR2=3*NCON12+HDV+NACMX1*(NDV2+NACMX1)*3*NRI+12
ISH 0589  NIS=NACMX12+2*NRI+2*NDV+NCON
ISH 0590  NR3=NSV
ISH 0591  IF (NSOBJ.GT.NR3) NR3=NSOBJ
ISH 0593  NR4=2*VAR
ISH 0594  NR5=NR2+NR3
ISH 0595  NR6=3*VARX+6+NDV+2*NV+N*NV
ISH 0596  NCONA+NXAPRX
ISH 0598  START OF TEMPORARY STORAGE.
ISH 0599  LCR(24)=LCR(23)
ISH 0600  IF (NCALC.EQ.2) LCR(24)=LCR(23)+2*NDV
ISH 0601  IF (NCALC.EQ.2) LCR(24)=LCR(23)+VAR
ISH 0602  NCR(24)=NCR(23)+2*VAR
ISH 0603  NCR(24)=NCR(23)+3*VARX
ISH 0604  NCR(24)=NCR(23)+4*VARX
ISH 0605  NCR(24)=NCR(23)+5*VAR
ISH 0606  IF (NCALC.EQ.6) NCR(24)=NCR(23)+3*VARX
ISH 0607  IF (NCALC.EQ.6) NCR(24)=NCR(23)+4*VARX
ISH 0608  TOTAL STORAGE REQUIREMENTS.
ISH 0609  LCR(25)=LCR(24)
ISH 0610  IF (NCALC.EQ.1) LCR(24)=LCR(23)+2*NDV
ISH 0611  IF (NCALC.EQ.1) LCR(24)=LCR(23)+VAR
ISH 0612  IF (NCALC.EQ.1) LCR(24)=LCR(23)+3*VARX
ISH 0613  IF (NCALC.EQ.1) LCR(24)=LCR(23)+4*VARX
ISH 0614  IF (NCALC.EQ.1) LCR(24)=LCR(23)+5*VARX
ISH 0615  IF (NCALC.EQ.1) LCR(24)=LCR(23)+6*VARX
ISH 0616  IF (NCALC.EQ.1) LCR(24)=LCR(23)+7*VARX
ISH 0617  IF (NCALC.EQ.1) LCR(24)=LCR(23)+8*VARX
ISH 0618  IF (NCALC.EQ.1) LCR(24)=LCR(23)+9*VARX
ISH 0619  IF (NCALC.EQ.1) LCR(24)=LCR(23)+10*VARX
ISH 0620  IF (NCALC.EQ.1) LCR(24)=LCR(23)+11*VARX
ISH 0621  IF (NCALC.EQ.1) LCR(24)=LCR(23)+12*VARX
ISH 0622  IF (NCALC.EQ.1) LCR(24)=LCR(23)+13*VARX
ISH 0623  IF (NCALC.EQ.1) LCR(24)=LCR(23)+14*VARX
ISH 0624  IF (NCALC.EQ.1) LCR(24)=LCR(23)+15*VARX
ISH 0625  IF (NCALC.EQ.1) LCR(24)=LCR(23)+16*VARX
ISH 0626  IF (NCALC.EQ.1) LCR(24)=LCR(23)+17*VARX
ISH 0627  IF (NCALC.EQ.1) LCR(24)=LCR(23)+18*VARX
ISH 0628  IF (NCALC.EQ.1) LCR(24)=LCR(23)+19*VARX
ISH 0629  IF (NCALC.EQ.1) LCR(24)=LCR(23)+20*VARX
ISH 0630  IF (NCALC.EQ.1) LCR(24)=LCR(23)+21*VARX
ISH 0631  IF (NCALC.EQ.1) LCR(24)=LCR(23)+22*VARX
ISH 0632  IF (NCALC.EQ.1) LCR(24)=LCR(23)+23*VARX
ISH 0633  IF (NCALC.EQ.1) LCR(24)=LCR(23)+24*VARX
ISH 0634  IF (NCALC.EQ.1) LCR(24)=LCR(23)+25*VARX
ISH 0635  IF (NCALC.EQ.1) LCR(24)=LCR(23)+26*VARX
ISH 0636  IF (NCALC.EQ.1) LCR(24)=LCR(23)+27*VARX
ISH 0637  CONTINUE
ISH 0638  RETURN
FOURTHPACE/10/11/80

(01 MAY 80) COPE01 SYSTEM/370 FORTRAN II EXTENDED (ENHANCED) DATE 02.14.010.54.12

1/4X,19HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727
1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)

1/4X,14HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727

1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)

1/4X,14HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727

1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)

1/4X,14HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727

1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)

1/4X,14HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727

1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)

1/4X,14HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727

1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)

1/4X,14HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727

1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)

1/4X,14HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727

1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)

1/4X,14HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727

1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)

1/4X,14HCONTROL PARAMETERS/5X,42CALCULATION CONTROL, 00727

1 NCALC = ,I5/5X,42NUMBER OF GLOBAL DESIGN VARIABLES, 00728
2NDV = ,I5/5X,42NUMBER OF SENSITIVITY VARIABLES, NSV = ,I5/5X,4200729
3NUMBER OF FUNCTIONS IN TWO-SPACE, N2VAR = ,I5/5X,42NUMBER OF APP00730
4ROXIMATING VARY, NAPRX = ,I5/5X,42HINFORMATION PRINT CODE, IFDBG 00731
5E, IPNPUT = ,I5/5X,42DEBUG PRINT CODE, =,IS)

1/4X,12HPRINT CONTROL, IPSENS = ,I5/5X,34HNUMBER 00733
1OF SENSITIVITY OBJECTIVES = ,I5//5X,53HGLOBAL NUMBERS ASSOCIATED H100714
2TH SENSITIVITY OBJECTIVES)
DATA SET U477COPE02 AT LEVEL 001 AS OF 02/13/81

SUBROUTINE COPE02 (ARRAY,RA,IA,ARRAY,NDRA,NDIA)

------------------------------------------------------------------

DIMENSION ARRAY(RA,RA(NDRA,IA)), RA(NDRA,IA)

******************************************************************

ROUTINE TO CONTROL OPTIMIZATION.


NASA-AIRES RESEARCH CENTER, MOFFETT FIELD, CALIF.

------------------------------------------------------------------

ARRAY STARTING LOCATIONS

------------------------------------------------------------------

X, VLB, VUB, DF, A, S, G1, G2, C, B, SCAL, ISC, IC, MS1

------------------------------------------------------------------

ARRAY DIMENSIONS

------------------------------------------------------------------

ISN 0005

******************************************************************

SOURCE EBCDIC NODECK OBJECT NOMAP NOFORMAT GNOSTMT NOXREF NOALC NODEC

TERM IBM FLAG (I)

DATA SET 918COPE02 AT LEVEL 001 AS OF 07/10/60

SUBROUTINE COPE02 (ARRAY,RA,IA,ARRAY,NDRA,NDIA)

------------------------------------------------------------------

DIMENSION ARRAY(RA,RA(NDRA,IA)), RA(NDRA,IA)

******************************************************************

ROUTINE TO CONTROL OPTIMIZATION.


NASA-AIRES RESEARCH CENTER, MOFFETT FIELD, CALIF.

------------------------------------------------------------------

ARRAY STARTING LOCATIONS

------------------------------------------------------------------

X, VLB, VUB, DF, A, S, G1, G2, C, B, SCAL, ISC, IC, MS1

------------------------------------------------------------------

ARRAY DIMENSIONS

------------------------------------------------------------------

ISN 0005

******************************************************************

SOURCE EBCDIC NODECK OBJECT NOMAP NOFORMAT GNOSTMT NOXREF NOALC NODEC

TERM IBM FLAG (I)
1505 IGOTO=0 00055
C CALL CONMIN (X,VLB,VUB,G,SCAL,DF,A,S,G1,G2,B,C,ISC,IC,MS1,N1,N2,N3)00056
1 = N4,N5)
C 00057
10 CONTINUE 00058
11 CALL CONMIN (RANX, RANVLB, RANVUB, RANNSCAL, RAN(NDF), RAN00059
1 (RAN(N1), RAN(N2), RAN(NH1), RAN(NH2), RAN(NH3), IAH(NISC), IAH(NIC), IAH(NS100060
2), N1, N2, N3, N4, N5)
C ANALIZE. 00061
00062
1 CALL COPE03 (ARRAY, NARRAY, RANX, RANV, RAN(NH1), RAN(NH2), RAN(NH3), RAN(NH4), RAN(NH5), RAN(NH6), RAN(NH7), ITER, OBJ)
1 ISGH, IA(NND50N), IA(NICON), N11, N12, N16, N17, ITER, OBJ)
C 00064
2 IF (IGOTO.GT.0) GO TO 10 00065
1 RETURN 00066
1 END 00067
*OPTIONS IN EFFECT* NAME=MAIN OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOCLASS(NONE)
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NOHAP NOFORMAT GOSTAT NOXREF NOAHC TERM IBM FLAG(1)
*STATISTICS* SOURCE STATEMENTS = 41, PROGRAM SIZE = 1352, SUBPROGRAM NAME =COPE02
**STATISTICS** NO DIAGNOSTICS GENERATED
***** END OF-compilation *****
3012k BYTES OF CORE NOT USED
DATA SET U477COPEO3 AT LEVEL 002 AS OF 03/16/81
DATA SET U477COPEO3 AT LEVEL 001 AS OF 02/13/81
DATA SET U477COPEO3 AT LEVEL 001 AS OF 07/10/80

SUBROUTINE COPEO3 (ARRAY,NARRAY,X,G,AMULT,BLU,IDSIGN,DSDGN,ICON,NH)
1,N2,NNS,NNS,INTER,GO)
COMMON /COPEO3/ SIGNOPT,SSPAR,SSPAP,NSV,NSS,KMAX,KMAX,Y0,KMAX,Y0,KS
12Y,NSVAR/IPESN,IPVAP,IPDBG,CHCN1,NDVTOT,LCR(25),LCI(25),ISCR10005
21,ISCR2,NSAPRX,NSH,NHS,HPH,HAP,HPH,HPH,HPH,HPH,HPH,HPH,HPH,HPH
3,NAM2,NAM3,NAM4,NMPTOT,JNM,JNM,JNM,JNM,JNM,JNM,JNM,JNM,JNM

COMMON /UOS/ ISI,161
DIMENSION ARRAY(NARRAY),X(NNS),G(NNS),AMULT(NNS),BLU(NNS),NNS)

******************************************************************
BUFFER BETWEEN COMMON AND COPES FUNCTION EVALUATION.
****************************************1(*************************
NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.
INITIAL ANALYSIS HAS BEEN DONE. IF ITER = 0, GO EVALUATE
OBJECTIVE AND CONSTRAINTS.
IF (ITER.EQ.0) ITER1=0
IF (ITER.LT.1) GO TO 40
------------------------------------------------------------------
PRINT OUTPUT IF DEBUG CONTROL IS TURNED ON
------------------------------------------------------------------
DEBUG OUTPUT AS REQUIRED.
IF (IPDBG.LT.1) GO TO 20
IF (ITER.EQ.ITER1) OR (ITER.LE.1) GO TO 20
XSAV2=X(I)
X(I)=XSAV1
N5=LOCR(5)
M2=LCR(2)
DO 10 I=1,NDVTOT
N=IDSIGN(I)
M=IDSIGN(I)
DO 20 I=1,NDVTOT
IF (X(I).GT.0) ARRAY(I)=X(N)AMULT(I)
10 CONTINUE
ICALC=3
ICALC=3
NM3=NM3+1
CALL ANALIZ (ICALC)
WRITE (161,70)
ITER1=ITER
DO 30 I=1,NDVTOT
X(I)=XSAV2
30 CONTINUE
------------------------------------------------------------------
TRANSFER DESIGN VARIABLES TO USER ARRAY
------------------------------------------------------------------
TRANSFER DESIGN VARIABLES TO USER ARRAY
------------------------------------------------------------------
CALL ANALIZ (ICALC)
WRITE (161,70)
ITER1=ITER
DO 30 I=1,NDVTOT
X(I)=XSAV2
30 CONTINUE
------------------------------------------------------------------
ANALIZE
------------------------------------------------------------------

ANALYZE

**VERSION 1.3.0 (01 MAY 80)**

COPE03 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 02.14/10.54.28

---

**ISN 0040**
ICalc=2

**ISN 0041**
NAME=NAME+1

**ISN 0042**
CALL ANALIZ (ICalc)

**ISN 0043**
SAVE X(1)

**ISN 0044**
CONTINUE

**ISN 0045**
OBJ=SGN0PT*ARRAY(IOBJ)

**ISN 0046**
IF (NCONA.EQ.0) RETURN

**ISN 0048**
J=1

**ISN 0049**
N=0

**ISN 0050**
DO 60 I=1,NCONA

**ISN 0051**
NN=ICON(I)

**ISN 0052**
CC=ARRAY(NN)

**ISN 0053**
BB=BLU(1,I)

**ISN 0054**
IF (BB.LT.-1.0E+15) GO TO 50

**ISN 0056**
CI=BLU(2,I)

**ISN 0057**
N=N+1

**ISN 0058**
G(N)=(BB-CC)/CI

**ISN 0059**
50 BB=BLU(3,I)

**ISN 0060**
CI=BLU(4,I)

**ISN 0061**
J=J+4

**ISN 0062**
IF (BB.GT.1.0E+15) GO TO 60

**ISN 0064**
N=N+1

**ISN 0065**
G(N)=CC-BB)/CI

**ISN 0066**
60 CONTINUE

**ISN 0067**
RETURN

---

**ISN 0068**
70 FORMAT (1H1)

**ISN 0069**
END

---

*OPTIONS IN EFFECT* NAME=MAIN OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBU(NONE)

*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECOD OBJECT NOIAP NOFORMAT GOSTMT NOXREF NOALC NODANSF TERM IBM FLAG(I)

*STATISTICS* SOURCE STATEMENTS = 68, PROGRAM SIZE = 1246, SUBPROGRAM NAME =COPE03

*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPIlATION *****

3004K BYTES OF CORE NOT USED
DATA SET U477COPE04 AT LEVEL 001 AS OF 02/13/81

DATA SET 9166COPE04 AT LEVEL 001 AS OF 07/10/80

COMMON /COPES1/ TITL(20)                          00001
COMMON /COPES3/ SGTP, NSV, NSOBJ, NCONA, N2VX, N2VY, M2VX, M2VY, MNOOA5, N2VAR, IPSENS, IP2VAR, IXFACT1, IXFACT2

COMMON /COPES5/ DLFUN, DABFUN, FDCH, FDCH1, CT, CTLMII, ALPHAX00002

C ROUTINE TO PROVIDE SENSITIVITY INFORMATION WITH RESPECT TO
C A PRESCRIBED SET OF DESIGN VARIABLES.
C STORE OUTPUT ON UNIT ISCRI.
C
C TITLE. 00003
WRITE (ISCRI,130) (TITLE(I),I=1,20) 00004

C NCALC, NSV, NSOBJ 00005
WRITE (ISCRI,340) NCALC, NSV, NSOBJ 00006

C ISENS(I),I=1,NSV 00007
WRITE (ISCRI,340) ISENS(I),I=1,NSV 00008

C NCALC=3 00009
C NCALC=2 00010
NDVS=NDV 00011

C SAVE X, VLB, VUB AND SCAL IN TEMPORARY STORAGE. 00012
C
C IF (NCALC.EQ.5) GO TO 10 00013

C IF (IPSENS.GT.0) NANS=NANS+1 00014
C IF (IPSENS.GT.0) CALL ANALIZ (JCALC) 00015
C GO TO 13 00016

C CONTINUE 00017
C SAVE X, VLB, VUB AND SCAL IN TEMPORARY STORAGE. 00018
C
C  SHIFT DESIGN VARIABLE INFORMATION IF ANY SENSITIVITY VARIABLE IS ALSO A DESIGN VARIABLE.

C IS THIS ALSO A DESIGN VARIABLE.

IF (L.EQ.N) GO TO 50

M2=M2+1
C SENSITIVITY VARIABLE IS NOT A DESIGN VARIABLE.

GO TO 90

CONTINUE

SENSITIVITY VARIABLE IS ALSO A DESIGN VARIABLE.

NDV2=NDV+2

DO 40 J=1,NDVTOT

M2=IA(J)
C SENSITIVITY VARIABLE IS NOT A DESIGN VARIABLE.

GO TO 90

CONTINUE

SENSITIVITY VARIABLE IS ALSO A DESIGN VARIABLE.

NDV=NDV-1

IDV=IA(M2)
C ELIMINATE THIS DESIGN VARIABLE AND REDUCE HIGHER NUMBER DESIGN VARIABLES BY ONE.

C VARIABLES BY ONE.

M2=LOC(M2)

DO 70 J=1,NDVTOT

IF (IA(M2).NE.IDV) GO TO 60

IA(M2)=0
C SET DESIGN VARIABLE VALUES TO SENSITIVITY VARIABLE VALUE.

K=IA(J)

M=LOC(K)+J-1

ARRAY(K)=ARRAY(N)*RA(M)

CONTINUE

IF (M.NE.IDV) GO TO 90

SHIFT X, VLB, VUB AND SCAL.

DO 80 J=IDV,NDV

X.

RA(J)=RA(J+1)
C VLB.

VUB.

RA(K)=RA(K+1)
C VUB.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.

K=K+NDV2

RA(K)=RA(K+1)
C SCAL.
**VERSION 1.3.0 (01 MAY 80) COPE04 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) **

**DATE 82.141/10.54.31 PAGE 3**

```plaintext
ISN 0043 L=LOCR(24) 00114
ISN 0044 DO 110 I=1,N 00115
ISN 0045 RA(I)=-RA(I) 00116
ISN 0046 110 L=L+1 00117
C PUT IDGN AND NDSGN BACK. 00118
ISN 0047 N=2*HDVTOT 00119
ISN 0048 L=LOCBI(24) 00120
ISN 0049 DO 120 I=1,N 00121
ISN 0050 IAI(I)=IA(I) 00122
ISN 0051 120 L=L+1 00123
ISN 0052 CONTINUE 00124
C WRITE NOMINAL RESULTS ON UNIT ISCR1 00125
C
C SENS(I,I). 00126
ISN 0053 M=1 00127
ISN 0054 DO 140 I=1,NSV 00128
ISN 0055 TEMP(I)=SENS(NI 00129
ISN 0056 140 M=H+HSENS(I) 00130
C SENSITIVITY OBJECTIVES, OBJZ. 00131
ISN 0057 DO 150 I=1,NSOBJ 00132
ISN 0058 M=NSENSZ(I) 00133
ISN 0059 WRITE (ISCR1,350) (TEMP(I,I=1,NSV) 00134
C ********** SENSITIVITIES ***l'******00135
C
C GLOBAL LOCATION OF SEHSITIVITY VARIABLE. 00136
ISN 0060 ISENSV=ISENS(III 00137
C NUMBER OF SENSITIVITY VARIABLES, NSENSV. 00138
ISN 0061 NSENSV=NSENS(I) 00139
C WRITE ISENSV AND NSENSV-1 ON UNIT ISCR1. 00140
ISN 0062 NSENSV=NSENSV-1 00141
C WRITE (ISCR1,350) ISENSV.NSENSV 00142
C IF (NSSENSV.LE.1) GO TO 320 00143
C ISENSV IS NOT A DESIGN VARIABLE. 00144
C MODIY OPTIMIZATION INFORMATION. 00145
ISN 0063 NDV=NDSY+2 00146
ISN 0064 NDV=NDSV+1 00147
C SAVE X, VLB, VUB AND SCAL FOR THIS DESIGN VARIABLE AND SHIF 00148
C REMAINING VARIABLES. 00149
C
C IS THIS SENSITIVITY VARIABLE ALSO A DESIGN VARIABLE. 00150
ISN 0065 NDV=NDV3AV 00151
ISN 0066 DO 160 I=1,HDVTOT 00152
ISN 0067 JJ=I 00153
ISN 0068 IF (IA(I).EQ.ISENSV) GO TO 170 00154
ISN 0069 160 CONTINUE 00155
C ISENSV IS NOT A DESIGN VARIABLE. 00156
C GO TO 210 00157
ISN 0069 170 CONTINUE 00158
C ISENSV IS A DESIGN VARIABLE. MODIFY OPTIMIZATION INFORMATION. 00159
ISN 0070 NDV=NDV3AV 00160
ISN 0071 NDV=NDV+2 00161
C SAVE X, VLB, VUB AND SCAL FOR THIS DESIGN VARIABLE AND SHIF 00162
C REMAINING VARIABLES. 00163
C
ISN 0072 M2=LOCBI(2)+JJ-1 00164
ISN 0073 ID1=IA(M2) 00165
ISN 0074 SAVX=RA(ID1) 00166
ISN 0075 K=ID1+NDV2 00167
```
SAVL=RAIKI(0)
K=K+NDV2
SAVU=RAIKI(0)
K=K+NDV2
SAV=RAIKI(0)
C SHIFT

IF (ID1.GT.NDV) GO TO 190
DO 180 I=ID1,NDV
RA(I)=RA(I+1)
K=I+NDV2
RAIKI=RAIKI+1
K=K+NDV2
RAIKI=RAIKI+1
180 RAIKI=RA(K+1)
190 CONTINUE

C MODIFY NDSGN.
M2=LOCII(2)
DO 200 I=I,NDVTOT
IF (IA(M2).EQ.ID1) IA(M2)=0
IF (IA(M2).GT.ID1) IA(M2)=IA(M2)-1
200 M2=M2+1
210 CONTINUE

C VARY THE VALUE OF THE SENSITIVITY PARAMETER
NSVAL1=NSVAL1+1
NSVALN=NSVAL1
DO 280 JJ=2,NSENSV
NSVAL1=NSVAL1+1
ARRAY(J)=SENS(NSVAL1, J)
WRITE (ISCR1,320) SENS(NSVAL1, J)

IF (NCALC.EQ.5) GO TO 220
NAN2=NAN2+1
CALL ANALIZ (ICALC)
IF (IPSENS.GT.O) NAN3=NAN3+1
IF (IPSENS.GT.0) CALL ANALIZ (JCALC)
GO TO 260
220 CONTINUE
NAN2=NAN2+1
CALL ANALIZ (ICALC)
IF (NDV.LE.O) GO TO 250
CALL CORE02 (ARRAY,RA,IA,NARRAY,NDRA,NOIA)
250 CONTINUE
IF (IPSENS.GT.0) NAN3=NAN3+1
ISN 0186  IF (IPSENS.GT.0) CALL ANALIZ (JCALC) 00232
ISN 0188 260 CONTINUE 00233
C WRITE SENSITIVITY RESULTS ON UNIT ISCR 00234
C OBJZ. 00235
ISH 0189 DO 270 I=1,NSOBJ 00236
ISH 0190 M=NSENSZ(I) 00237
ISH 0191 270 TEI1PII)=ARRAY(M) 00238
ISH 0192 WRITE (ISCR1,350) (ITEMP(Il),I=1,NSOBJ) 00239
ISH 0193 280 CONTINUE 00240
ISH 0194 ARRAY(NSENSV)=RENSINSVALU 00241
ISH 0195 IF (NCALC.NE.5.0R.ID1.EQ.0) GO TO 320 00242
C RESTORE X, VLBI, VUB AND SCAL. 00243
ISH 0197 NOV=NOVSAV 00244
ISH 0198 IF (ID1.EQ.NDV) GO TO 300 00245
ISH 0200 L=NOV-1 00246
ISH 0201 L=L 00247
ISH 0202 DO 290 (I=1,NOV 00248
ISH 0203 RA(L+l)=RA(L) 00249
ISH 0204 K=ID1+NDV2 00250
ISH 0205 RAK)=SAVL 00251
ISH 0206 K=K+NDV2 00252
ISH 0207 RA(K+I)=RA(K) 00253
ISH 0208 K=K+NDV2 00254
ISH 0209 RA(K+l)=RA(K) 00255
ISH 0210 290 L=L-1 00256
ISH 0211 RA(ID1)=SAVX 00257
ISH 0212 K=ID1+NDV2 00258
ISH 0213 RAK)=SAVL 00259
ISH 0214 K=K+NDV2 00260
ISH 0215 RA(K)=SAVL 00261
ISH 0216 K=K+NDV2 00262
ISH 0217 RA(K)=SAVL 00263
ISH 0218 300 CONTINUE 00264
ISH 0219 C RESTORE NDSGN. 00265
ISH 0219 M2=LOCI(2) 00266
ISH 0220 DO 310 I=1,NDVTOT 00267
ISH 0221 IF (IA(M2).GE.ID1) IA(M2)=IA(M2)+1 00268
ISH 0222 IF (IA(M2).EQ.0) IA(M2)=ID1 00269
ISH 0223 310 M2=M2+1 00270
ISH 0224 320 CONTINUE 00271
ISH 0225 RETURN 00272
ISH 0226 330 FORMAT (20A4) 00273
ISH 0228 C FORMATS 00274
ISH 0228 340 FORMAT (1615) 00275
ISH 0229 350 FORMAT (5E15.8) 00276
ISH 0230 360 FORMAT (120A4) 00277
ISH 0231 END 00278

*OPTIONS IN EFFECT*NAME=MAIN) OPTIMIZE(1) LINECOUNT(60) SIZE(MAX) AUTODBL(NOHE)
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NDECK OBJECT NCOAP NFORMAT GOSTMT NOXREF NDAUL NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 230, PROGRAM SIZE = 3454, SUBPROGRAM NAME = COPE04
*STATISTICS* NO DIAGNOSTICS GENERATED

****** END OF COMPIlATION ******

2972K BYTES OF CORE NOT USED
SUBROUTINE COPEOS (IRA,IA,NDRA,NDIA,ISCRI)

DIMENSION RA(NDRA), I(NDIA)

ROUTINE TO PRINT SENSITIVITY INFORMATION STORED ON UNIT ISCRI.

BY G. N. VANDERPLAATS
JULY, 1974.
NOMENCLATURE RESEARCH CENTER, MOFFETT FIELD, CALIF.

READ ISCRI,70) (RA(I),I=1,20)
READ ISCRI,80) NCALC, NSV, NSOBJ
IF NCALC.NE.3.AND. NCALC.NE.5) RETURN
IF NCALC.EQ.3) WRITE (161,90) 00007
WRITE (161,60) IRA(I),I=1,20)
WRITE (161,100) NSV, NSOBJ
ISENSI), I=1, NSV.
READ ISCRI,80) (IA(I),I=1, NSV)
WRITE (161,110) 00010
WRITE (161,60) IRA(I),I=1, NSV.
READ ISCRI,80) (IA(I),I=1, NSOBJ)
WRITE (161,120) 00013
WRITE (161,130) 00016
WRITE (161,140) (RA(I),I=1, NSV)
WRITE (161,150) 00019
WRITE (161,160) (RA(I),I=1, NSV)
DO 10 J=1, NSOBJ 00022
SENS(I,J)=1, NSV.
WRITE (161,170) 00025
WRITE (161,180) 00028
DO 20 J=1, NSOBJ 00031
SENS(I,J)=1, NSOBJ.
WRITE (161,190) 00034
WRITE (161,200) 00037
WRITE (161,210) 00040
WRITE (161,220) 00043
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
SENS(I,J)=1, NSOBJ.
READ (ISCR1,140) SENSIJ 00057
ISN 0039
READ (ISCR1,140) (RA(I),I=1,NSOBJ) 00058
ISN 0040
WRITE (161,210) SENSIJ,
RA(I),I=1,NSOBJ 00059
ISN 0041
N=(NSOBJ-1)/4 00060
IF (N.LT.1) GO TO 20 00061
ISN 0042
LI=5 00062
ISN 0043
DO 10 I=1,N 00063
ISN 0044
L2=LI+3 00064
ISN 0045
L2=MINO(L2,NSOBJ) 00065
ISN 0046
WRITE (161,220) RA(J),J=L1,L2 00066
ISN 0047
10 LI=LI+4 00067
ISN 0048
20 CONTINUE 00068
ISN 0049
CONTINUE 00069
ISN 0050
CONTINUE 00070
ISN 0051
CONTINUE 00071
ISN 0052
RETURN 00072
C --------------- 00073
C FORHATS 00074
C --------------- 00075
ISN 0053
SO FORMAT (1H1,4X,46HOPTIMUt1 SENSITIVITY ANALYSIS RESULTS (NCALC=S)) 00076
ISN 0054
60 FCRYMAT (/ISX,5HTITLE/5X,20A4) 00077
ISN 0055
70 FORMAT (20A4) 00078
ISN 0056
80 FORMAT (16IS) 00079
ISN 0057
90 FORMAT (1H1,4X,47I1STANDARD SENSITIVITY ANALYSIS RESULTS (NCALC=3)) 00080
ISN 0058
100 FORMAT (//5X,36HNUMBER OF SENSITIVITY VARIABLES, NSV,9X,1H=,IS/SX,00081
139HNUMBER OF SENSITIVITY OBJECTIVES, NSOBJ,4X,1H=,IS) 00082
ISN 0059
110 FORMAT (/ISX,S2HGLOBAL NUMBERS ASSOCIATED WITH SENSITIVITY VARIABLES) 00083
ISN 0060
120 FORMAT (1H1,S3HGLOBAL NUMBERS ASSOCIATED WITH SENSITIVITY OBJECTIVES) 00084
ISN 0061
130 FORMAT (5X,5HGLOBAL NUMBERS ASSOCIATED WITH SENSITIVITY VARIABLES) 00085
15ES) 00086
ISN 0062
140 FORMAT (5E15.8) 00087
ISN 0063
150 FORMAT (1H1,4X,31HVALUES OF SENSITIVITY VARIABLES) 00088
11TIVITY VARIABLES) 00089
ISN 0064
160 FORMAT (5X,E13.5) 00090
ISN 0065
170 FORMAT (1H1,4XVALUES OF SENSITIVITY OBJECTIVE FUNCTIONS) 00091
ISN 0066
180 FORMAT (1H1,2HSENSITIVITY ANALYSIS RESULTS) 00092
ISN 0067
190 FORMAT (1H1,15HGLOBAL VARIABLE, IS//10X,1HX,20X,4HF(X)1) 00093
ISN 0068
200 FORMAT (1H1,35HTHE NOMINAL VALUE IS THE ONLY VALUE//5X,4HSPECIFIED00094
1 FOR THIS VARIABLE) 00095
ISN 0069
210 FORMAT (1H1,3X,E12.4,3X,E13.4) 00096
ISN 0070
220 FORMAT (10X,E14.4) 00097
ISN 0071
END 00098
ISN 0072
C OPTIONS IN EFFECT=NAME( MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOCLONE(NONE)
* STATISTICS* SOURCE STATEMENTS = 71, PROGRAM SIZE = 2210, SUBPROGRAM NAME =COPE05
**** END OF COMPIILATION ***** 2996K BYTES OF CORE NOT USED
SUBROUTINE COPE06 (ARRAY, RA, IA, NARRAY, NDATA, NDATA)

DIMENSION ARRAY(NARRAY), RA(NDATA), IA(NDATA)

**ROUTINE TO CALCULATE FUNCTIONS OF TWO DESIGN VARIABLES FOR ALL COMBINATIONS OF A SET OF PRESCRIBED VALUES OF THESE VARIABLES.**

WRITE OUTPUT INFORMATION ON SCRATCH UNIT ISCR1.

BY G. N. VANDERPLAATS AUG., 1974.

NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.

------------------------------------------------------------------

WRITE ISCR1,40) TITLE, I=1,20)

WRITE (ISCR1,50) NVAR, M2VAR, M2VX, M2VY, M2VZ.

WRITE (ISCR1,50) LOC(20), LOC(21) - 1.

WRITE (ISCR1,50) IA(I), I = M20, H21.

------------------------------------------------------------------

TWO-VARIABLE FUNCTION SPACE

IC1CALC = 2
KC1CALC = 3
ISIGN = 1

DO 20 J = 1, M2VY

N21 = N21 + ISIGN

ARRAY(N2VY) = RAIN21

ANALYZ.

NAN2 = NAN2 + 1

CALL ANALYZ (ICALC)

IF (IP2VAR.GT.0) CALL ANALYZ (KCALC)

IF (IP2VAR.GT.0) NAN3 = NAN3 + 1

WRITE ISCR1,60) RA(N20), RAIN21) FOR (X,Y) VALUES.

------------------------------------------------------------------

WRITE X, Y.

WRITE ISCR1,60) RA(N20), RA(N21) FOR (X,Y) VALUES.
RA(N24)=ARRAY(N)
M20=M20+1
COHTINUE
N24=N23+N2VAR-1
WRITE (ISCR1,60) (RA(K),K=H23,H24)
CONTINUE
N21=N21+ISIGN
H20=H20+1
ISIGN=-ISIGN
CONTINUE
RETURN

C ------------------------------------------------------------------
C FORMATS
C ------------------------------------------------------------------
C
ISH 0046 40 FORMAT (20A4)
ISH 0047 50 FORMAT (1615)
ISH 0048 60 FORMAT (SE15.0)
ISH 0049 END

**OPTIONS IN EFFECT** NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOCODE(NONE)
**OPTIONS IN EFFECT** SOURCE EBCDIC NOLIST NODECK OBJECT NOMAP N2FORMAT GOSTMT NOREF NOALC NONSF TERM IBM FLAG(1)
**STATISTICS** SOURCE STATEMENTS = 48, PROGRAM SIZE = 1118, SUBPROGRAM NAME =COPE06
**STATISTICS** NO DIAGNOSTICS GENERATED
****** END OF COMPILATION ******

3008K BYTES OF CORE NOT USED
SUBROUTINE COPE07 (RA, IA, NORA, NOIA, ISCR1)

TITLE.
READ (ISCR1, 60) (RA(I), I=1, 20)
READ (ISCR1, 70) (NCALC, N2VAR, N2XV, N2YV, N2YV)
IF (NCALC.NE.4.AND.NCALC.HE.6) RETURN
WRITE (161, 50) 0001
WRITE (161, 40) (RA(I), I=1, 20)
N2VZ(I), I=1, N2VAR.
READ (ISCR1, 70) (IA(I), I=1, N2VAR)
WRITE (161, 120) N2VX, N2VY
WRITE (161, 130) N2VZ.
WRITE (161, 80) (IA(I), I=1, N2VAR)

TWO-VARIABLE FUNCTION SPACE INFORMATION

DO 30 I=1, M2VX
DO 30 J=1, H2VY
X, Y.
READ (ISCR, 150) XX, YY, F(X, Y).
READ (ISCR1, 150) (RA(K), K=1, N2VAR)
N=4
IF (N2VAR.LT.4) N=H2VAR
IF (J.EQ.1) WRITE (161, 100) XX, YY, (RA(K), K=1, N)
IF (J.GT.1) WRITE (161, 90) YY, (RA(K), K=1, N)
N=10
H=(N2VAR-1)/4
K=1, H
WRITE (161, 110) (RA(KK), KK=H, L1)
C FORMATS 00055
C
------------------------------------------------------------------
ISN 0041 40 FORMAT (/5X,5HTITLE/5X,20A4) 00056
ISN 0042 50 FORMAT (1H1,4X,35HTWO-VARIABLE FUNCTION SPACE RESULTS) 00058
ISN 0043 60 FORMAT (20A4) 00059
ISN 0044 70 FORMAT (16I5) 00060
ISN 0045 80 FORMAT (5X,10I5) 00061
ISN 0046 90 FORMAT (/15X,E12.4,3X,4E13.4) 00063
ISN 0047 100 FORMAT (/3X,2E12.4,3X,4E13.4) 00064
ISN 0048 110 FORMAT (30X,4E13.4) 00065
ISN 0049 120 FORMAT (/5X,49HTWO-VARIABLE FUNCTION SPACE RESULTS) 00066
ISN 0050 130 FORMAT (/5X,49HTWO-VARIABLE FUNCTION SPACE RESULTS) 00067
ISN 0051 140 FORMAT (/10X,1HX,1HY,20X,6HF(X,Y)) 00068
ISN 0052 150 FORMAT (5E15.8) 00069
ISN 0053 END 00070

*OPTIONS IN EFFECT* NAME( MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX)
AUTODBL(NONE)

*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NODATA
FORMAT COSTMT NDXREF NOALC NOANSF TERM IBM FLAG(I)

*STATISTICS* SOURCE STATEMENTS = 52, PROGRAM SIZE = 1470, SUBPROGRAM NAME =COPE07

*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPILATION *****

3004K BYTES OF CORE NOT USED
SUBROUTINE COPE08 (A,B,IFORM,NFLD)
DIMENSION All), B(1), C(10)

DATA CONMA/IH,/,BlANK/IH

C ROUTINE TO CONVERT UNFORMATTED DATA TO FORMATTED DATA IN FIELDS OF 10 EACH FIELD RIGHT JUSTIFIED.
C NASA AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.

C --- INPUT.
C A - ARRAY OF DATA SEPERATED BY COMMAS. MINIMUM DIMENSION OF ISN 0002
C IS A(80).
C --- OUTPUT.
C B - ARRAY OF DATA IN FIELDS OF 10 AND RIGHT JUSTIFIED.
C MINIMUM DIMENSION OF B IS 10*NFLD.
C IFORM - 0 IF A WAS UNFORMATTED.
C 1 IF A WAS ALREADY FORMATTED.
CNFLD - NUMBER OF FIELDS OF DATA.
C --- NOTE.
C 1) DATA IS ASSUMED TO BE REAL OR INTEGER DATA WITH NO EMBEDDED BLANKS WITHIN A GIVEN FIELD.
C 2) DATA IS CONSIDERED UNFORMATTED IF:
C A) A COMMA IS FOUND.
C B) LAST NON-BLANK CHARACTER IS IN COLUMN 1-10 AND THERE IS NO DECIMAL AND IT IS NOT RIGHT JUSTIFIED.
C
C ISN 0003

C SEARCH FOR LAST NON-BLANK CHARACTER AND SEARCH FOR COMMA.
C CALCULATE NUMBER OF NON-BLANK SETS.

C 1) DATA MAY BE FORMATTED.
C NO COMMA WAS FOUND. DATA MAY BE FORMATTED.
C IF (LST.GE.10) GO TO 93
C IF MORE THAN ONE SETS OF CHARACTERS, DATA IS ASSUMED FORMATTED.
C IF (KNDH.GT.1) GO TO 93
C
C ISN 0004

C DATA IS UNFORMATTED.

C ISN 0005

C SEARCH FOR LAST NON-BLANK CHARACTER AND SEARCH FOR COMMA.
C CALCULATE NUMBER OF NON-BLANK SETS.

C 1) DATA IS ASSUMED TO BE REAL OR INTEGER DATA WITH NO EMBEDDED BLANKS WITHIN A GIVEN FIELD.
C 2) DATA IS CONSIDERED UNFORMATTED IF:
C A) A COMMA IS FOUND.
C B) LAST NON-BLANK CHARACTER IS IN COLUMN 1-10 AND THERE IS NO DECIMAL AND IT IS NOT RIGHT JUSTIFIED.
C
C ISN 0006

C INON=0
C KNON=0
C LST=0
C DO 10 I=1,80
C IF (A(I).EQ.COMMA) GO TO 20
C INON=INON+1
C LST=LST+1
C IF (A(I).NE.BlANK) GO TO 30
C
C ISN 0007

C CALCULATE NUMBER OF NON-BLANK SETS.
C
C ISN 0008

C SEARCH FOR LAST NON-BLANK CHARACTER AND SEARCH FOR COMMA.
C CALCULATE NUMBER OF NON-BLANK SETS.

C 1) DATA MAY BE FORMATTED.
C NO COMMA WAS FOUND. DATA MAY BE FORMATTED.
C IF (LST.GE.10) GO TO 93
C IF MORE THAN ONE SETS OF CHARACTERS, DATA IS ASSUMED FORMATTED.
C IF (KNDH.GT.1) GO TO 93
C
C ISN 0009

C DATA IS UNFORMATTED.

C ISN 0010

C INON=0
C KNON=0
C LST=0
C DO 10 I=1,80
C IF (A(I).EQ.COMMA) GO TO 20
C INON=INON+1
C LST=LST+1
C IF (A(I).NE.BlANK) GO TO 30
C
C ISN 0011

C 1) DATA IS ASSUMED TO BE REAL OR INTEGER DATA WITH NO EMBEDDED BLANKS WITHIN A GIVEN FIELD.
C 2) DATA IS CONSIDERED UNFORMATTED IF:
C A) A COMMA IS FOUND.
C B) LAST NON-BLANK CHARACTER IS IN COLUMN 1-10 AND THERE IS NO DECIMAL AND IT IS NOT RIGHT JUSTIFIED.
C
C ISN 0012

C 1) DATA MAY BE FORMATTED.
C NO COMMA WAS FOUND. DATA MAY BE FORMATTED.
C IF (LST.GE.10) GO TO 93
C IF MORE THAN ONE SETS OF CHARACTERS, DATA IS ASSUMED FORMATTED.
C IF (KNDH.GT.1) GO TO 93
C
C ISN 0013

C DATA IS UNFORMATTED.
ISH 0036 JJ=0
ISH 0037 DO 40 J=I,80
ISH 0038 IF (A(J).EQ.COMMA.OR.A(J).EQ.BLANK) GO TO 50
ISH 0040 JJ=JJ+1
ISH 0041 C(JJ)=A(J)
ISH 0042 NFLO=NFLO+1
ISH 0043 I=I+JJ
C BLANK FIELD NFLO OF B.
ISH 0044 K1=K2-9
ISH 0045 DO 60 K=K1,K2
ISH 0046 B(K)=BLANK
C STORE C IN FIELD NFLO OF B, RIGHT JUSTIFIED.
ISH 0047 IF (JJ.EQ.0) GO TO 80
ISH 0049 JJ=JJ
ISH 0050 K=K2
ISH 0051 DO 70 L=I,JJ
ISH 0052 B(L)=C(JJ)
ISH 0053 K=K-1
ISH 0054 J1=J1
ISH 0055 K2=K2+10
ISH 0056 GO TO 30
ISH 0057 CONTINUE
C formatted input. Store A directly in B.
ISH 0058 IFORM=1
ISH 0059 NFLO=8
ISH 0060 DO 100 I=1,80
ISH 0061 B(I)=A(I)
ISH 0062 100 CONTINUE
ISH 0063 RETURN
ISH 0064 END

*OPTIONS IN EFFECT* NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOCOLINCHE
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NOAP NOSTDIN GOSTDIN NOXREF NOALC NOALSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 63, PROGRAM SIZE = 846, SUBPROGRAM NAME =COPE08
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPILATION *****
3012K BYTES OF CORE NOT USED
DATA SET U477COPE09 AT LEVEL 002 AS OF 03/16/81
DATA SET U477COPE09 AT LEVEL 001 AS OF 02/13/81
DATA SET 9108COPE09 AT LEVEL 002 AS OF 11/14/80

SUBROUTINE COPE09

COMMON /CHMIN/ DELFUN, DABFUN, FDCHM, CTMIN, CTLMIN, ALPHAX

1, ABSCT, THETA, NXV, NCV, NCO, NC, INFO, IFDGS, APTR, HST, ITMAX, TIM0
2, CMIN, CMAX, NCC, INFO, IFDGS, ITMAX

COMMON /COPES2/ IA(ISOE), IA(1000)

COMMON /COPES3/ SINDG, NCALC, IBS, NISV, NCO, NCC, NNV, M2VX, NVY, M2VY, M2VZ, NVZ, M2VZ

COMMON /COPES4/ IRPENS, IPENAR, IPDGS, NAC, NHT, LCR(25), LCR(25), LCR(25), LCR(25), NAC

11, ISCR, ICA, HAPRX, HPS, HPS, HPS, HPS, HPS, HPS, HPS, HPS, HPS, HPS, HPS

3, NAXR, NAXR, NAXR, NAXR, NAXR, NAXR, NAXR, NAXR, NAXR, NAXR, NAXR, NAXR, NAXR

IF (ITMAX .LT. 3) GO TO 160
IF (KMAX .LT. 1) GO TO 160
IF (CTMIN .LE. .0005) CTMIN = .0005
IF (CTLHM .LE. .0001) CTLHM = .0001
IF (ABSCT .LE. .0001) CT = .0001
IF (ABSCT .LE. .0001) CTL = .0001
IF (DELFUN .LE. .0001) DELFUN = .0001
IF (IPDGS .LT. 1) IPDGS = 1
IF (ITMAX .LT. 50) ITMAX = 50

HIDE = 1
KOUNT = 0

---- ARRAY STARTING LOCATIONS

NNI = NAPRX + 1
NKV = LCR(23)
NVLB = NVLB + NNI
NCO = NVLB + NNI
NDX = NCO + NAPRX
NFHOM = NDX + NDV
NSTAY = NFHOM + NF
NDR = NDR + (NAPRX * NAPRX + 1) / 2
IF (MAXTRM .LT. 3) NDR = MAXTRM * NAPRX
NTHP = NSTAY + NDR + NF
NBLU = LCC(6)
NISC = LCC(4)
HGHN = LCC(23)
NIDV = NIGHN + NCC
IF (KMAX .LT. 0) GO TO 160
C CONVIN ARRAYS.
C DIMENSIONS.
ISN 0049
NN1 = IDV + 2
ISN 0050
NN2 = 2*IDV + NCON
ISN 0051
NN3 = NACHX1
ISN 0052
NN4 = NN3
ISN 0053
IF (IDV.GT. NN4) NN4 = IDV
ISN 0055
NN5 = 2*NN4
ISN 0056
SCAL, DF, G, A, S, GI, G2, C, B, ISC, IC, MS1.
ISH 0057
NINCAL = LCCR(4)
ISH 0058
NDF = NTHP
ISH 0059
NS = NDF + NN1
ISH 0060
NS = NA + NN2
ISH 0061
NG1 = NS + NN1
ISH 0062
NG2 = NS1 + NN2
ISH 0063
NC = NC + N4
ISH 0064
M9 = NC + N4
ISH 0065
MISC = LOCI(4)
ISH 0066
NIC = NIVD + NXAPRX
ISH 0067
MISC1 = NIC + NN3
------------------------------------------------------------------
C DETERMINED IBOJA, ARRAYS IGFH AND IDV.
C
C IBOJA.
ISH 0068
M6 = LOCI(6)
ISH 0069
d 10 I = I + NF
ISH 0070
J = IA(M6)
ISH 0071
IBOJA = J
ISH 0072
IF (J .EQ. IBOJ) GO TO 20
ISH 0074
M6 = M6 + 1
ISH 0075
10 CONTINUE
C ERROR - IBOJA NOT FOUND.
ISH 0076
20 CONTINUE
ISH 0077
IF (NCONA .EQ. 0) GO TO 60
C IGFH ARRAY.
ISH 0079
M3 = LOCI(3)
ISH 0080
M23 = LOCI(23)
ISH 0081
d 50 I = 1 + NCONA
ISH 0082
COMMENT: ----GLOBAL LOCATIONS OF CONSTRAINED PARAMETERS.---------
ISH 0083
J = IA(M3)
ISH 0084
M3 = M3 + 1
ISH 0085
COMMENT: ----LOCAL VARIABLE, F, LOCATION.-----------------------------
ISH 0086
M6 = LOCI(6)
ISH 0087
d 30 K = 1 + NF
ISH 0088
KK = K
ISH 0089
L = IA(M6)
ISH 0090
IF (L .EQ. J) GO TO 40
ISH 0091
M6 = M6 + 1
ISH 0092
30 CONTINUE
C COMMENT: ERROR - CONSTRAINED VARIABLE IS NOT AN APPROXIMATE FUNCTION.
ISH 0093
40 CONTINUE
ISH 0094
M23 = KK
ISH 0095
M23 = M23 + 1
ISH 0096
50 CONTINUE
ISH 0097
60 CONTINUE
C COMMENT: ------JOV ARRAY.---------------------------------------------
ISH 0098
N3 = NIDV
ISH 0099
N5 = LOCI(5)
**VERSION 1.3.0 (01 MAY 80) COPE09 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 82.141/10.54.59 PAGE 3**

```
+ISN 0099  DO 90  I=1,NXAPRX
  00114
  IAIN3) = 0
  00115
COMENT: ----GLOBAL LOCATION.-------------------------------------------00116
  11 = IAIN(S)
  00117
ISN 0102  NS = NS + 1
  00118
COMENT: ----FIND CORRESPONDING DESIGN VARIABLE.------------------------00119
  11 = IA(N)
  00120
ISH 0103  N1 = LOCI(1)
  00121
ISH 0104  N2 = LOCI(2)
  00122
ISH 0105  DO 70  J=1,NDVTOT
  00123
ISH 0106  IF (IA(NI) .EQ. II) GO TO 80
  00124
ISH 0108  N1 = N1 + 1
  00125
ISH 0109  N2 = N2 + 1
  00126
ISH 0110  70 CONTINUE
  00127
COMENT: ----THIS APPROXIMATING VARIABLE IS NOT A DESIGN VARIABLE.-----00128
  00129
ISH 0111  GO TO 90
  00130
ISH 0112  90 CONTINUE
  00131
ISH 0113  IAIN(S) = IA(N)
  00132
ISH 0114  NS = NS + 1
  00133
ISH 0115  90 CONTINUE
  00134
COMENT: ----CHECK TO BE SURE EACH INDEPENDENT DESIGN VARIABLE IS ALSO AN00135
  00136
ISH 0116  CONTINUE
  00137
ISH 0117  DO 120  I=1,NDV
  00138
ISH 0118  N1 = LOCI(1)
  00139
ISH 0119  N2 = LOCI(2)
  00140
ISH 0120  DO 111  J=1,NDVTOT
  00141
ISH 0121  IF (IA(NI) .NE. II) GO TO 110
  00142
ISH 0122  COMMENT: ----YES.--------------------------------------------00143
  00144
ISH 0123  IGLOB = IA(NI)
  00145
ISH 0124  N1 = N1 + 1
  00146
ISH 0125  NS = LOCI(5)
  00147
ISH 0126  DO 100  K=1,NXAPRX
  00148
ISH 0127  COMMENT: ----IS THIS THE SAME AS IGLOB.----------------------00149
  00150
ISH 0128  IF (IA(NS) .EQ. IGLOB) GO TO 120
  00151
ISH 0129  COMMENT: ----NO.---------------------------------------------00152
  00153
ISH 0130  NS = NS + 1
  00154
ISH 0131  111 CONTINUE
  00155
ISH 0132  120 CONTINUE
  00156
ISH 0133  COMMENT: --ERROR -- DESIGN VARIABLE IS NOT AN APPROXIMATING VARIABLE.--00157
  00158
ISH 0134  120 CONTINUE
  00159
ISH 0135  BEGIN SEQUENTIAL APPROXIMATE OPTIMIZATION.
  00160
ISH 0136
```

The code snippet appears to be a FORTRAN program related to optimization or solving a particular problem, involving loops, conditional statements, and array indexing. The specific purpose or context of the code is not immediately clear from the snippet alone. The program seems to be handling array elements and making decisions based on their values, possibly for a simulation or algorithmic purpose.
CENTER CONSTRAINTS.

WRITE (161,560) OC175

N1 = NIGFN

WRITE (1161,570) UAII),I=N1,N2) 00178

CONTINUE 00179

CONTINUE:

DESIGN VARIABLES.

WRITE (1161,580) H2

WRITE (1161,570) UAII),I=H1,N2) 00184

CONTINUE 00185

CONTINUE:

ITERATION NUMBER.

WRITE (1161,680) KOUNT 00187

HP = NPTOT

C

------------------------------------------------------------------
C SET UP ARRAYS XNON, FNon, XI, Y. 00191
C

------------------------------------------------------------------

NXI = NTP

NY = NI* NXAPRX*NPTOT

NIGHT = NY + NF*NPTOT

IF (IKOUNT .LT. 2 .AND. DABFUH .LE. 0.) DABFUH = .001*ABSIOBJSAV)00206

IF (IPAPRX .LT. 1 .OR. IPAPRX .EQ. 3) GO TO 170 00208

COMMENT:------------------ PRINT CURRENT NOMINAL.

WRITE (161,690) INOM 00210

H2 = NXNOM + NF - 1

WRITE (161,700) IRA(I),I=NXNOM,N2) 00212

WRITE (161,710) 00213

N2 = NFHOM + NF - 1

WRITE (161,700) IRA(I),I=NFHOM,N2) 00215

CONTINUE 00216

C

------------------------------------------------------------------
C LEAST SQUARES FIT FOR TAYLOR SERIES EXPANSION.
C

------------------------------------------------------------
COMMENT:------------------PRINT TAYLOR SERIES COEFFICIENTS.------------------
WRITE (161,720)
DO 160 J=1,100
H3 = LOCI(6)
HI = H3TAY
DO 160
J=I,HF
N2 = HI + M - 1
WRITE (161,600) J,IA(H3)
WRITE (161,700) (RA(I),I=N1,N2)
HI = HI + HBR
H3 = N3 + 1
160 CONTINUE
190 CONTINUE
C
C
IF (KMAX .GT. KMAX) GO TO 470
180 CONTINUE
190 CONTINUE
C
C
INITIALIZE XV, DX, VLB, VUB.
C
C
DO 200 I=1,HXAPRX
RA(HI) = 0.
N1 = HI + 1
200 CONTINUE
H2 = HVLB
H3 = HUB
N4 = HXNOM
N5 = LOCR(2)
N6 = LOCR(3)
N7 = LOCR(7)
N8 = NDX
L1 = M - NDV
L2 = L1 - NDV
ICK = ICK1 + ICK2 + ICK3
DO 210 I=I,NDV
RA(HS) = 0.
XFACT = 1.
IF (I .LE. L1) XFACT = XFACT1
L2 = L2 - NDV + 1
IF (L2 .GE. 0) XFACT = XFACT2
DX = XFACT*RA(H7)
XX = RA(H4)
DXL = XX - RA(N5)
IF (DXL .GT. DX) DXL = DX
DXU = RA(H6) - XX
IF (DXU .GT. DX) DXU = DX
RA(N2) = -DXL
RA(N3) = DXU
N2 = N2 + 1
N3 = N3 + 1
N4 = N4 + 1
N5 = N5 + 1
N6 = N6 + 1
N7 = N7 + 1
N8 = N8 + 1
CONTINUE 00291
IF (IPAPRX .LT. 2 .AND. IPAPRX .NE. 4) GO TO 220 00292
WRITE (161,610) 00293
WRITE (161,700) (RA(I),I=NVLB,NL) 00294
WRITE (161,620) 00295
WRITE (161,700) (RA(I),I=NVLB,NL) 00296
WRITE (161,730) 00297
CONTINUE 00298
CONTINUE 00299
C ----------------------------------------------- 00300
C OPTIMIZE APPROXIMATE FUNCTION. 00301
C ----------------------------------------------- 00302
COMMENT:------------------------------------------ 00303
ISGOTO = 0 00304
C CALL CONMIN (X,VLB,VUB,G,SCAL,DF,A,S,G1,G2,B,C,ISC,IC,MS1,N1,N2,H3) 00305
230 CONTINUE 00306
COMMENT:------------------------------------------ 00307
CALL CONMIN (RA(NXV),RA(NVUB),RA(NVUB),RA(NVISCAL),RA(NDF),R0) 00308
211,NV1,NV2,NV3,NV4,NV5 00309
COMMENT:------------------------------------------ 00310
TRANSFER VARIABLES FROM X TO XV. 00311
N1 = NTDV 00312
N2 = NVV 00313
DO 240 I=1,NXAPRX 00314
II = IAT(NI) 00315
N4 = NDV + II - 1 00316
IF (II .GT. 0 .AND. II .LE. NOVB) RA(N2) = RA(N4) 00317
N1 = NV + 1 00318
N2 = NV + 1 00319
N3 = NV + 1 00320
240 CONTINUE 00321
COMMENT:------------------------------------------ 00322
APPROXIMATE ANALYSIS. 00323
COMMENT:------------------------------------------ 00324
CALL COPEIS (RA(NXV),RA(NVUB),RA(NVUB),RA(NVISCAL),RA(NDF),R0) 00325
211,NV1,NV2,NV3,NV4,NV5 00326
COMMENT:------------------------------------------ 00327
IF (IGOTO .GT. 0) GO TO 230 00328
C IF DESIGN PRODUCED ZERO DELTA-X TWICE IN A ROW AND KOUNT.GE.KMII, 00329
C TERMINATE. 00330
ICKI = ICKI + 1 00331
SUM = 0. 00332
N1 = NVV 00333
DO 250 I=1,NV 00334
SUM = SUM + RA(N1)**2 00335
N1 = N1 + 1 00336
250 CONTINUE 00337
COMMENT:------------------------------------------ 00338
INSURE NEW X-VECTOR IS INDEPENDENT 00339
C ----------------------------------------------- 00340
JJJ = 0 00341
SUM = 0. 00342
N1 = NVV 00343
DO 250 I=1,NV 00344
SUM = SUM + RA(N1)**2 00345
N1 = N1 + 1 00346
250 CONTINUE 00347
COMMENT:------------------------------------------ 00348
INSURE NEW X-VECTOR IS INDEPENDENT 00349
C ----------------------------------------------- 00350
JJJ = JJJ + 1 00351
SUM = 0. 00352
N1 = NVV 00353
DO 250 I=1,NV 00354
SUM = SUM + RA(N1)**2 00355
N1 = N1 + 1 00356
250 CONTINUE 00357
COMMENT:------------------------------------------ 00358
INSURE NEW X-VECTOR IS INDEPENDENT 00359
C ----------------------------------------------- 00360
JJJ = JJJ + 1 00361
SUM = 0. 00362
N1 = NVV 00363
DO 250 I=1,NV 00364
SUM = SUM + RA(N1)**2 00365
N1 = N1 + 1 00366
250 CONTINUE
+VERSION 1.3.0 (01 MAY 80) COFE09 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 82.141/10.54.59 PAGE 7

ISN 0303 N2 = NDXG1 00350
ISN 0304 N3 = NXV 00351
ISN 0305 DO 270 I=1,NXAPRX 00352
ISN 0306 RA(N1) = RA(N2) + RA(N3) 00353
ISN 0307 N1 = N1 + 1 00354
ISN 0308 N2 = N2 + 1 00355
ISN 0309 N3 = N3 + 1 00356
ISN 0310 270 CONTINUE 00357

COMMENT:--------- READ X-VECTORS ONE AT A TIME AND COMPARE TO XNOM. 00358
ISN 0311 REWIND ISCR2 00359
ISN 0312 N1 = NTMP + NXAPRX 00360
ISN 0313 N2 = N1 + NXAPRX - 1 00361
ISN 0314 N3 = N2 + 1 00362
ISN 0315 N4 = N3 + NF - 1 00363
ISN 0316 DO 290 J=1,NPTOT 00364
ISN 0317 KK = J 00365

COMMENT:--------- X-VECTOR. 00366
ISN 0318 READ (ISCR2) (RA(I),I=N1,N2) 00367

COMMENT:--------- Y-VECTOR. NOT USED. READ TO POSITION ISCR2. 00368
ISN 0319 READ (ISCR2) (RA(I),I=N3,N4) 00369

ISN 0320 N5 = N1 00370
ISN 0321 N6 = NTMP 00371
ISN 0322 SUM = 0. 00372
ISN 0323 DO 280 I=1,NXAPRX 00373
ISN 0324 SUM = SUM + (RA(N5) - RA(N5))**2 00374
ISN 0325 N5 = N5 + 1 00375
ISN 0326 N6 = N6 + 1 00376
ISN 0327 280 CONTINUE 00377
ISN 0328 IF (SUM.LE.1.0E-10) GO TO 300 00378

COMMENT:--------- COMPARE X WITH XNOM. 00379
ISN 0329 CONTINUE 00380
ISN 0330 IF (SUM.LE.1.0E-10) GO TO 300 00381
ISN 0331 300 CONTINUE 00382

C THIS DESIGN IS SAME AS A PREVIOUS DESIGN. 00383
C MODIFY DELTA-X VECTOR. 00384
ISN 0333 N5=NVUB 00385
ISN 0334 N7=NXV 00386
ISN 0335 N8=NVUB 00387
ISN 0336 N9=NXV+NDV-1 00388
ISN 0337 IF (IPAPRX.LT.1.OR.IPAPRX.EQ.3) GO TO 310 00389
ISN 0338 WRITE (I61,630) 00390
ISN 0339 WRITE (I61,640) 00391
ISN 0340 WRITE (I61,700) (RA(I),I=N7,N9) 00392
ISN 0341 WRITE (I61,750) 00393
ISN 0342 WRITE (I61,700) (RA(I),I=N7,N9) 00394
ISN 0343 WRITE (I61,640) 00395
ISN 0344 WRITE (I61,640) 00396
ISN 0345 310 CONTINUE 00397

C MODIFY DELTA-X VECTOR. 00398
ISN 0346 AMULT=.01*FLOAT(IJJ) 00399
ISN 0347 DO 320 I=1,NDV 00400
ISN 0348 BU=RA(N5) 00401
ISN 0349 BL=RA(N3) 00402
ISN 0350 IF (BL.LT.-1.0E+15) BL=0. 00403
ISN 0351 IF (BU.GT.1.0E+15) BU=0. 00404
ISN 0352 DB=ABS(BU-BL) 00405
ISN 0353 IF (DB.LT.1.0E-10) DB=.1 00406
ISN 0354 DX=RA(N7)+AMULT*DB 00407
ISN 0355 IF (DX.GT.RA(N6)) DX=DX-2.*AMULT*DB 00408
ISN 0356 IF (DX.LT.RA(N8)) DX=DX+1.5*AMULT*DB 00409
ISN 0357
C 101 MAY 80 COPE09 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED) DATE 02.141/10.54.59 PAGE 8

ISN 0363 RA(N7)=DX 00409
ISN 0364 N5=N5+1 00410
ISN 0365 N7=N7+1 00411
ISN 0366 N8=N8+1 00412
ISN 0367 320 CONTINUE 00413
ISN 0368 CALL COPE15 (RA(NKV),RA(NJ),RA(NDF),RA(NA),IA(NISC),IA(NIC),N1,RA00414
1(RBLU),N1,XJ,OBJA,M,RA(NFHON),RA(NFHIM),RA(HSTAT),N6R,IA(HIGHF),CT,00415
2CTL,INHF,HAC,NCOHA,MDV,NC,OBJ,GSOPT) 00416
ISN 0369 IF (J111.LT.4) GO TO 260 00417
C FOUR Tries HAVE FAILED to PRODUCE A USABLE X-VECTOR. 00418
C USE LATEST TRY. 00419
ISN 0371 IF (KK.EQ.NHPTOT) GO TO 340 00420
KK=KK+1 00421
ISN 0373 DO 330 J=KK,NHPTOT 00422
ISN 0374 READ (ISCR2) (RA(I),I=N1,N2) 00423
ISN 0376 330 READ (ISCR2) (RA(I),I=N3,N4) 00424
ISN 0377 340 CONTINUE 00425
ISN 0378 IF (IPAPRX.LT.1.OR.IPAPRX.EQ.3) GO TO 350 00426
ISN 0380 IF (J111.EQ.4) WRITE (161,650) 00427
ISN 0382 350 CONTINUE 00428
ISN 0383 360 CONTINUE 00429
ISN 0384 370 00430
C UPDATE ANALYSIS. 00431
C------------------------------------------------------------------00432
C XNOM. 00433
C N1=ANOM 00434
C N2=NXV 00435
C DO 370 I=I1,XAPRX 00436
C RAIN11=RAIN1)+RAIN2) 00437
C N1=N1+1 00438
C N2=N2+1 00439
C GLOBAL VARIABLES. 00440
C N3=NXNOM 00441
C N4=NXDV 00442
C DO 410 I=I1,XAPRX 00443
C DESIGN VARIABLE NUMBER. 00444
C II=IA(N4) 00445
C IF (II.EQ.0) GO TO 400 00446
C DESIGN VARIABLE UPDATE. 00447
C N1=LOGC1(I) 00448
C N2=LOGC1(2) 00449
C N5=LOGC5(5) 00450
C DO 390 J=1,NHVTOT 00451
C IF (JAI(NH2).NE.II) GO TO 380 00452
C UPDATE VARIABLE J. 00453
C JJ=IA(N1) 00454
C ARRAY(JJ)=RA(N3)*RA(N5) 00455
C N1=N1+1 00456
C N2=N2+1 00457
C N5=N5+1 00458
C N4=N4+1 00459
C IF (IPAPRX.LT.1.OR.IPAPRX.EQ.3) GO TO 420 00460
C PRINT APPROXIMATE OPTIMIZATION INFORMATION. 00461
C WRITE (161,740) 00462
C N2=NXV+NDV-1 00463
C WRITE (161,700) (RA(I),I=NXV,N2) 00464
C
ISH 0415 WRITE (161,760) 00468
ISH 0416 N2=NXNOM+NXAPRX-1 00469
ISH 0417 WRITE (161,700) (RA(I),I=NXNOM,N2) 00470
ISH 0418 WRITE (161,760) 00471
ISH 0419 N2=NFNEW+NF-1 00472
ISH 0420 WRITE (161,700) (RA(I),I=NFNEW,N2) 00473
ISH 0421 CONTINUE 00474
ISH 0422 IF (ICK1.EQ.2.AND.KOUNT.GE.KMIN).AND.(IPAPRX.GT.0.AND.IPAPRX.NE.300475
1) WRITE (161,540) 00476
ISH 0424 IF (ICK1.EQ.2.AND.KOUNT.GE.KMIN) GO TO 460 00477
ISH 0426 ICALC=2 00478
ISH 0427 N2=N2+1 00479
ISH 0428 CALL ANALIZ (ICALC) 00480
C NEW FUNCTION VALUES. 00481
ISH 0429 NI=NFNON 00482
ISH 0430 M6=LOCII6) 00483
ISH 0431 DO 430 I=I,NF 00484
ISH 0432 M6=I6+1 00485
ISH 0433 RAIH I=ARRAYM6) 00486
ISH 0434 N2=HFHm1+NF-l 00487
ISH 0435 CONTINUE 00488
ISH 0436 IF (IPDBG.LT.1) GO TO 440 00489
C DEBUG OUTPUT. 00490
ISH 0438 N=3=4+1 00491
ISH 0439 ICALC=3 00492
ISH 0440 CONTINUE 00493
ISH 0441 IF (IPAPRX.LT.1) GO TO 450 00494
C PRINT PRECISE FUNCTION VALUES. 00495
ISH 0444 WRITE (161,770) 00496
ISH 0445 WRITE (161,700) (RA(I),I=NFNON,N2) 00497
ISH 0446 CONTINUE 00498
C NEW OBJECTIVE. 00499
ISH 0448 Obj=-RAIN1)*SGtnOPT 00500
C ------------------------------------------------------------------00501
C WRITE NEW X AND F VALUES ON ISCR2. 00502
C ------------------------------------------------------------------00503
ISH 0450 NI=NXNOM+NXAPRX-1 00504
ISH 0451 WRITE (161,760) (RA(I),I=NXNOM,N1) 00505
C FUNCTIONS. 00506
ISH 0452 NI=NFNON+NF-1 00507
ISH 0453 WRITE (161,760) (RA(I),I=NFNON,N1) 00508
C UPDATE PARAMETERS. 00509
ISH 0454 NPTOT=NPTOT+1 00510
ISH 0455 IF (J.JJ.LT.2.OR.KOUNT.LT.KMIN) INGH=NPTOT 00511
C ------------------------------------------------------------------00512
C CONVERGENCE CHECK. 00513
ISH 0457 IF (KOUNT.LT.KMIN) GO TO 130 00514
ISH 0459 ICK2=ICK2+1 00515
ISH 0460 ICK3=ICK3+1 00516
ISH 0461 DEL=ABS(OBJ) 00517
ISH 0462 IF (DEL.LT.1.0E-6) DEL=1.0E-6 00518
ISH 0464 DEL=OBJ/DEL 00519
ISH 0465 DEL=ABS(DEL) 00520
ISH 0466 IF (DEL.GT.DELFUN) ICK2=0 00521
ISH 0468  DEL=ABS(OBJ-OBJSAV) 00527
ISH 0469  IF (DEL.GT.DABFUN) ICK3=0 00528
ISH 0471  IF (ICK2.GE.2 .OR .ICK3.GE.2) GO TO 460 00529
ISH 0473  IF (KOUNT.LT.KMAX) GO TO 130 00530
ISH 0475  CONTINUE 00531
C FINAL INFORMATION. 00532
C --00533
ISH 0476  IF (IFAPRX.GT.0 .AND .IFAPRX.NE.3) WRITE (I6I,660) 00534
ISH 0478  INOM=O 00535
ISH 0479  KOUNT=KMAX+1 00536
ISH 0480  IF (KOUNT.LT.JHOM) KOUNT=JHOM+1 00537
ISH 0482  CTSAV=CT 00538
ISH 0483  CTLSAV=CTL 00539
ISH 0484  IF (ABS(CT).LT.1.0E-10) CT=-.004 00540
ISH 0486  IF (ABS(CTL).LT.1.0E-10) CTL=-.001 00541
ISH 0488  GO TO 160 00542
ISH 0489  CONTINUE 00543
ISH 0490  CT=CTSAV 00544
ISH 0491  CTL=CTLSAV 00545
C STORE FINAL VALUES OF XNOM IN GLOBAL ARRAY. 00546
ISH 0492  N3=NOMX 00547
ISH 0493  N4=NIDV 00548
ISH 0494  DO 510 I=1,NXAPRX 00549
C DESIGN VARIABLE NUMBER. 00550
ISH 0495  II=IA(N4) 00551
ISH 0496  IF (II.EQ.O) GO TO 500 00552
C DESIGN VARIABLE UPDATE. 00553
ISH 0498  II=IA(N4) 00554
ISH 0499  II=IA(N4) 00555
ISH 0500  NS=LCR(N5) 00556
ISH 0501  DO 490 J=1,NDVTOT 00557
ISH 0502  IF (IA(NH2).NE.II) GO TO 480 00558
C UPDATE VARIABLE J. 00559
ISH 0504  JJ=IA(N4) 00560
ISH 0505  ARRAY(JJ)=RA(H3)*RA(HS) 00561
ISH 0506  ARRAY(JJ)=RA(N3)*RA(N5) 00562
ISH 0507  N1=NS+1 00563
ISH 0508  N2=NS+1 00564
ISH 0509  N5=NS+1 00565
ISH 0510  CONTINUE 00566
ISH 0511  N4=NH4+1 00567
C STORE FINAL VALUES OF FHOM IN GLOBAL ARRAY. 00568
ISH 0512  M6=LOCII(1) 00569
ISH 0513  M6=LOCII(1) 00570
ISH 0514  DO 520 I=1,NF 00571
C DESIGN VARIABLE NUMBER. 00572
ISH 0515  II=IA(M6) 00573
ISH 0516  M6=NH4+1 00574
ISH 0517  ARRAY(II)=RA(N1) 00575
ISH 0518  ARRAY(II)=RA(N1) 00576
ISH 0519  CONTINUE 00577
ISH 0520  RETURN 00578
C FORMATS 00579
C ---------------00580
ISH 0521  540 FORMAT (/5X,7HTWO CONSECUTIVE APPROXIMATE OPTIMIZATIONS HAVE PRODUCED THE SAME DESIGN,SX,2ND OPTIMIZATION TERMINATED) 00581
ISH 0522  550 FORMAT (/5X,2ND APPROXIMATING FUNCTION IS THE OBJECTIVE) 00582
APPROXIMATING FUNCTIONS
ASSOCIATED WITH CONSTRAINTS

ISN 0523 560 FORMAT (5X,51HAPPROXIMATING FUNCTIONS ASSOCIATED WITH CONSTRAINTS)

ISN 0524 570 FORMAT (5X,10I5)

ISN 0525 550 FORMAT (5X,63HDESIGN VARIABLE NUMBERS ASSOCIATED WITH APPROXIMATING)

111G VARIABLES)

ISN 0526 590 FORMAT (5X,59H* LEAST SQUARES FIT TO APPROXIMATION DATA IS SINGULAR *

/*5X,24HRESULTS MAY NOT BE VALID)

ISN 0527 600 FORMAT (5X,15HFUNCTION NUMBER,15,25H -- GLOBAL VARIABLE NUMBER,15/0

15X,12HCOEFFICIENTS)

ISN 0528 610 FORMAT (5X,44HSIDE CONSTRAINTS ON APPROXIMATE OPTIMIZATION//5X,120

0595

ISN 0529 620 FORMAT (5X,12HUPPER BOUNDS)

ISN 0530 630 FORMAT (5X,76HFOR OPTIMIZATION HAS PRODUCED AN X-VECTOR WHICH IS THE

SAME AS A PREVIOUS DESIGN)

ISN 0531 640 FORMAT (5X,51HTHE FOLLOWING DESIGN IS NOT THE APPROXIMATE OPTIMUM)

ISN 0532 650 FORMAT (5X,60HFOUR ATTEMPTS HAVE FAILED TO PRODUCE AN INDEPENDENT

X-VECTOR/SX,S2HOPTIMIZATION WILL CONTINUE WITH MOST RECENT X-VECTOR)

ISN 0533 660 FORMAT (5X,40HFINAL RESULT OF APPROXIMATE OPTIMIZATION)

ISN 0534 670 FORMAT (1H1,4X,40HBEGIN OPTIMIZATION ITERATION HISTORY)

ISN 0535 680 FORMAT (5X,22HGLOBAL VARIABLE NUMBER,15)

ISN 0536 690 FORMAT (5X,21HGLOBAL DESIGN NUMBER =,15//5X,68HVECTOR)

ISN 0537 700 FORMAT (5X,5E13.5)

ISN 0538 710 FORMAT (5X,15HFUNCTION VALUES)

ISN 0539 720 FORMAT (5X,26HTAYLOR SERIES COEFFICIENTS)

ISN 0540 730 FORMAT (5X,35HRESULTS OF APPROXIMATE OPTIMIZATION)

ISN 0541 740 FORMAT (5X,14HDELTA-X VECTOR)

ISN 0542 750 FORMAT (5X,61HVECTOR)

ISN 0543 760 FORMAT (5X,27HAPPROXIMATE FUNCTION VALUES)

ISN 0544 770 FORMAT (5X,23HPRECISE FUNCTION VALUES)

ISN 0545 END

OPTIONS IN EFFECT : NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOQBL(NONE)

OPTIONS IN EFFECT : SOURCE EBCDIC NOLIST NODECK OBJECT NOCAP NMAP NOFORMAT GOSTMT HI

REF NOALC NDNSF TERM IBM FLAG(I)

STATISTICS* SOURCE STATEMENTS = 544, PROGRAM SIZE = 9668, SUBPROGRAM NAME = COPE09

*STATISTICS* NO DIAGNOSTICS GENERATED

***** END OF COMPIILATION ******

2904K BYTES OF CORE NOT USED
SUBROUTINE COPE10 (XI, Y, XNOM, FNO1, NF, NPTOT, KO1NT, BLU, IGFN, IOBJA, ISC)

DIMENSION XI(NXAPRX,1), Y(INF,1), XNOM(1), FHOM(1), BLU(4,1), IGFN(1), WGGHT(1)

C ******************************************************************************
C ROUTINE TO SET UP ARRAYS FOR TAYLOR SERIES EXPANSION.
C ******************************************************************************
C BY G. N. VANDERPLAATS
C
C******************************************************************************
C FIND BEST NOMINAL IF REQUIRED.
C******************************************************************************
IF (KOUNT.LE.1.AND.INOM.GT.0) GO TO 20
IF (KOUNT.GT.1.AND.KOUNT.LE.JNOM) GO TO 20
CALL COPE11 (NPTOT, Y, NF, INOM, BLU, NCONA, IGFN, IOBJA, SNOPT, CTIN, CT0022)

CONTINUE

C******************************************************************************
C CREATE XNOM AND FNON.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C SHIFT XI AND Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C SHIFT XI AND Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C SHIFT XI AND Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C SHIFT XI AND Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C SHIFT XI AND Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C SHIFT XI AND Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C SHIFT XI AND Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C SHIFT XI AND Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C SHIFT XI AND Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C REPLACE XI BY DELTA-XI AND Y BY DELTA-Y.
C******************************************************************************
DO 110 J=1,NP
DO 90 I=1,NXAPRX
XI(I,J)=XI(I,JI-XNOM(I)
DO 100 I=1,NF
Y(I,J)=Y(I,J)-FNON(I)
CONTINUE

C******************************************************************************
C WEIGHTING FACTORS.
C !-------------------------------------------------- 00055
ISN 0034
ISH 0035 DO 130 J=1,NP
ISH 0036 SUM=0.
ISH 0037 DO 120 I=1,NXAPRX
ISH 0038 SUM=SUM+XI(I,J)**2
ISH 0039 IF (SUM.GT.SMAX) SMAX=SUM
ISH 0040 SMAX=SQRT(SMAX)
ISH 0041 DO 140 I=1,NP
ISH 0042 WEIGHT(I)=SQRT(SMAX)
ISH 0043 IF (NP.LE.NPMAX) RETURN
ISH 0044 C !-------------------------------------------------- 00065
ISH 0045 REDUCE THE NUMBER OF DESIGNS TO NPMAX. 00066
ISH 0046 C !-------------------------------------------------- 00067
ISH 0047 NPMAX=NP+1
ISH 0048 NPSAV=NP
ISH 0049 DO II=NPMAX,NPSAV
ISH 0050 FIND DESIGN WITH MINIMUM WEIGHTING FACTOR. 00073
ISH 0051 WMIN=WEIGHT(I)
ISH 0052 IMN=I
ISH 0053 DO 150 I=2,NP
ISH 0054 IF (WEIGHT(I).GE.WMIN) GO TO 150
ISH 0055 IMN=I
ISH 0056 IF (IMN.EQ.NP) GO TO 190
ISH 0057 CONTINUE
ISH 0058 IF (IMN.EQ.NP) GO TO 190
ISH 0059 C ! SHIFT XI, Y AND WEIGHT. 00082
ISH 0060 NPH=NP-1
ISH 0061 DO 160 J=1,NPH
ISH 0062 DO 160 I=1,NXAPRX
ISH 0063 XI(I,J)=XI(I,J+1)
ISH 0064 DO 170 I=1,NF
ISH 0065 Y(I,J)=Y(I,J+1)
ISH 0066 WEIGHT(I)=WEIGHT(I+1)
ISH 0067 NP=NP-1
ISH 0068 CONTINUE
ISH 0069 RETURN
ISH 0070 END

*OPTIONS IN EFFECT=NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOCELL(NONE)
*OPTIONS IN EFFECT=SOURCE EBCDIC NODIAG NOSECRET OCTOPH NDVAL NDSSFS TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 69, PROGRAM SIZE = 2038, SUBPROGRAM NAME =COPEI0
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPILED *****

3000K BYTES OF CORE NOT USED
SUBROUTINE COPE11 (NPTOT, Y, NYR, IND, BLU, NCONA, IGFN, IBOJA, SGNOPT, CT)

DATA SET US7COPE11 AT LEVEL 001 AS OF 02/13/81
DATA SET US8COPE11 AT LEVEL 001 AS OF 07/10/80

DIMENSION (NYR,1), BLU(4,1), IGFN(1), ISC(1)

ROUTINE TO DETERMINE NOMINAL DESIGN FOR APPROXIMATE OPTIMIZATION.

BY G. N. VANDERPLAATS
JAN., 1979.

NOMINAL DESIGN IS THE ONE WITH LOWEST OBJECTIVE SATISFYING ALL CONSTRAINTS.
IF ALL DESIGNS VIOLATE CONSTRAINTS, THE DESIGN WITH
THE LEAST VIOLATION IS FOUND.
II IS DESIGN WITH LOWEST MAXIMUM CONSTRAINT VALUE.
I2 IS THE DESIGN WITH THE LOWEST OBJECTIVE SATISFYING ALL CONSTRAINTS.

OBJMAX=-Y(IOBJA,1)*SGNOPT
DO 10 J=2,NPTOT
OBJ=-Y(IOBJA,JI)*SGNOPT
IF (OBJ.GT.OBJMAX) OBJMAX=OBJ
CONTINUE

GMAX=1.0E+20
II=1
DO 10 J=2,NPTOT
OBJ=-Y(IOBJA,JI)*SGNOPT
ICON=0
DO 30 I=1,NCONA
II=IGFN(II)
OBJ=OBJ-BlU(1,II)*GG
IF (OBJ.GT.OBJMAX) OBJMAX=OBJ
CONTINUE

UMAX=1.0E+20
II=1
DO 10 J=1,NPTOT
OBJ=-Y(IOBJA,JI)*SGNOPT
ICON=0
DO 30 I=1,NCONA
II=IGFN(II)
OBJ=OBJ-BlU(1,II)*GG
IF (OBJ.GT.OBJMAX) OBJMAX=OBJ
CONTINUE

OBJMAX=-Y(IOBJA,1)*SGNOPT
OBJ=-Y(IOBJA,JI)*SGNOPT
IF (OBJ.GT.OBJMAX) OBJMAX=OBJ
CONTINUE

GMAX=1.0E+20
II=1
DO 10 J=1,NPTOT
OBJ=-Y(IOBJA,JI)*SGNOPT
ICON=0
DO 30 I=1,NCONA
II=IGFN(II)
OBJ=OBJ-BlU(1,II)*GG
IF (OBJ.GT.OBJMAX) OBJMAX=OBJ
CONTINUE
ISH 0038  CT=CT1
ISH 0039  IF (ISCICON).GT.0) CT=CT1
ISH 0041  G=(GS-BLU(3,I))/BLU(4,I)-CT
ISH 0042  IF (G.GT.GL1) GI=G
ISH 0044  30  CONTINUE
ISH 0045  IF (GI.LT.0..OR.GL.GT.GMAX) GO TO 40
ISH 0047  IL=J
ISH 0048  GMAX=GI
ISH 0049  40  IF (OBJ.GT.OBJMAX.OR.GI.GT.0.) GO TO 50
ISH 0051  I2=J
ISH 0052  OBJMAX=OBJ
ISH 0054  50  CONTINUE
ISH 0058  I2=II
ISH 0059  IF (I2.GT.0) INOM=I2
ISH 0060  RETURN
ISH 0063  EL10

*OPTIONS IN EFFECT=*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODE(LNONE)
*OPTIONS IN EFFECT=*SOURCE EBCDIC NODLIST NODECK OBJECT NODAP NOFORMAT GOSTHT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* NO DIAGNOSTICS GENERATED
**END OF COMPILATION**
**RELEASE 1.3.0 (01 MAY 80)**

**REQUESTED OPTIONS:** SOURCE, NOPAP, NOXREF, NOSLIST, NODECK, OPT(3), AUTOCLASS(NONE), NOLAC

**OPTIONS IN EFFECT:** MAIN(HUGUE) OPTIMIZE(3) LINESECTION(60) SIZE(MAX) AUTOCLASS(NONE)

**SOURCE EBDCIC NOLIST NODECK OBJECT NOPAP NOFORMAT COSTHT NOXREF NOLAC NOSNFF TERM IBM FLAG(I)

```c
 DATA SET U477COPE12 AT LEVEL 001 AS OF 02/13/81 00001
 DATA SET 9186COPE12 AT LEVEL 001 AS OF 07/10/80 00002
 SUBROUTINE COPE12 00003
 ************************************************************************************
 ROUTINE TO PERFORM A LEAST SQUARES FIT OF AN ARBITRARY FUNCTION OF NV VARIABLES.
 Y = F(X1, X2, ..., XN) = B(1)*F(1) + B(2)*F(2) + ... + B(M)*F(M) 00004
 ************************************************************************************
 BY G. N. VANDERPLAATS 00005
 NASA AMES RESEARCH CENTER, MONTFELL FIELD, CALIF. 00006

ARGUMENTS.
X,Y - INPUT ARRAYS OF OBSERVATIONS OF NP POINTS 00009
NX - NUMBER OF INDEPENDENT VARIABLES OF WHICH Y IS A FUNCTION. 00010
NP - NUMBER OF OBSERVATION POINTS. 00011
NF - NUMBER OF SEPARATE CURVE FITS BEING DONE SIMULTANEOUSLY. 00012
M - NUMBER OF COMPONENTS OF THE FUNCTIONS TO BE FITTED. 00013
B - ARRAY OF M COEFFICIENTS OF FUNCTIONAL FIT TO DATE. 00014
A - (M+1)/2 WORK VECTOR. 00015
F - WORK VECTOR - F(M). 00016
G - WORK VECTOR - G(NF). 00017
NXR - DIMENSIONED ROWS OF X. 00018
NYR - DIMENSIONED ROWS OF Y. 00019
NBR - DIMENSIONED ROWS OF B. 00020
WGHT - ARRAY OF WEIGHTING FACTORS - WGHT(NP). 00021
NER - ERROR FLAG. IF NER.GT.0, DIAGONAL ELEMENT NER OF A IS LESS THAN 1.0E-10. 00022
USER SUPPLIED SUBROUTINE, COPE13. 00023

USAGE
CALL COPE13(X,Y,NX,NP,NF,M,B,A,F,G,NXR,NYR,NBR,WGHT,NER)

ROUTINE TO EVALUATE COMPONENTS F(1),...,F(M) WHICH ARE TO BE FITTED TO DATA.
TO DATA, ROUTINE EVALUATES THE FUNCTIONS FOR A SINGLE VECTOR OF XI AND STORES THE RESULTING VALUES IN VECTOR F.
ARGUMENTS.
XI - VECTOR OF INDEPENDENT VARIABLES AT WHICH FUNCTIONS ARE TO BE EVALUATED.
F - VECTOR OF FUNCTION VALUES.
NF - NUMBER OF INDEPENDENT VARIABLES OF WHICH Y IS A FUNCTION.
M - NUMBER OF FUNCTION COMPONENTS, ALSO REQUIRED DIMENSION OF NER 00030

SUBROUTINE COPE12 (X,Y,NX,NP,NF,M,B,A,F,G,NXR,NYR,NBR,WGHT,NER) 00031
DIMENSION X(NXR,1), Y(NYR,1), B(NBR,1), A(1), F(1), G(1) 00032
DIMENSION WGHT(1) 00033
IF (NX.LE.NP) GO TO 50 00034
SPECIAL CASE. FEWER OBSERVATIONS THAN DESIGN VARIABLES. USE AVERAGE FINITE DIFFERENCE FOR FIRST ORDER EXPANSION. 00035
NPI=NP+1 00036
NP1=NP+1 00037
AP=FLOAT(NP) 00038
DO 20 I=1,NX 00039
XI(NI,NP1)=AP 00040
DO 10 J=1,NP 00041
```

---

**SOURCE:** EBDCIC NOLIST NODECK OBJECT NOPAP NOFORMAT COSTHT NOXREF NOLAC NOSNFF TERM IBM FLAG(I)

**REQUESTED OPTIONS:** SOURCE, NOPAP, NOXREF, NOSLIST, NODECK, OPT(3), AUTOCLASS(NONE), NOLAC

**OPTIONS IN EFFECT:** MAIN(HUGUE) OPTIMIZE(3) LINESECTION(60) SIZE(MAX) AUTOCLASS(NONE)
IF (ABS(X(I,J)).GT.1.0E-10) GO TO 10
X(I,J)=X(I,J)-1.
CONTINUE
IF (X(I,NP1).LT.1.) X(I,NP1)=1.
CONTINUE
DO 40 J=1,NF
B(I,J)=0.
DO 30 K=1,NP
B(I,J)=B(I,J)+Y(I,K)/X(I,K)
40 B(I,J)=B(I,J)/X(I,NP1)
NER=0
RETURN
CONTINUE
GENERAL CASE. DO LEAST SQUARES FIT.
A=B=0.
DO 60 J=1,NF
DO 60 I=1,M
60 B(I,J)=0.
L=(L+1)/2
DO 70 J=1,L
70 A(J)=0.
LOWER TRIANGLE OF A IN SYMMETRIC MODE.
DO 100 K=1,NP
I=K
KK=I
DO 110 J=1,M
IF (K-J).LT.1.
90 B(I,J)=B(I,J)+Y(I,J)*F(I)
100 CONTINUE
SOLVE FOR B.
IF (M.LT.1) GO TO 200
C LDU DECOMPOSITION.
M1=M-1
K=0
DO 110 K=1,M1
IF (K.EQ.0) GO TO 100
IF (ABS(A(K)).LT.1.0E-20) GO TO 220
FACT=1./A(K)
A(K)=FACT
K1=K+1
KJ=K
DO 120 J=1,M1
IF (J.GT.K1) GO TO 130
K2=K
DO 120 I=1,J
I=I+1
KI=K+I
120 CONTINUE
100 CONTINUE
**VERSION 1.3.0 (01 MAY 80) COPE12 SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)**

<table>
<thead>
<tr>
<th>Line</th>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>0069</td>
<td>A(I,J)=A(I,J)-A(K,K)A(K,J)</td>
</tr>
<tr>
<td>0070</td>
<td>CONTINUE</td>
</tr>
<tr>
<td>0071</td>
<td>K=KK+M</td>
</tr>
<tr>
<td>0072</td>
<td>NER=M</td>
</tr>
<tr>
<td>0073</td>
<td>IF (ABS(A(K,K)).LT.1.0E-20) GO TO 220</td>
</tr>
<tr>
<td>0075</td>
<td>A(K,K)=1./A(K,K)</td>
</tr>
<tr>
<td>0076</td>
<td>C FORWARD SUBSTITUTION</td>
</tr>
<tr>
<td>0077</td>
<td>MPL=M+1</td>
</tr>
<tr>
<td>0078</td>
<td>DO 130 K=1,MP1</td>
</tr>
<tr>
<td>0079</td>
<td>KPI=K+1</td>
</tr>
<tr>
<td>0080</td>
<td>KK=KK+KK+K</td>
</tr>
<tr>
<td>0081</td>
<td>AKK=AKK(K)</td>
</tr>
<tr>
<td>0082</td>
<td>DO 120 L=1,MP</td>
</tr>
<tr>
<td>0083</td>
<td>B(K,L)=B(K,L)*AKK</td>
</tr>
<tr>
<td>0084</td>
<td>KI=KK</td>
</tr>
<tr>
<td>0085</td>
<td>DO 130 I=KPI,M</td>
</tr>
<tr>
<td>0086</td>
<td>KI=KI+I-1</td>
</tr>
<tr>
<td>0087</td>
<td>AKK=AKK(I)</td>
</tr>
<tr>
<td>0088</td>
<td>DO 140 J=1,MP</td>
</tr>
<tr>
<td>0089</td>
<td>B(J,I,J)=B(J,I,J)-AKK*B(K,J)</td>
</tr>
<tr>
<td>0090</td>
<td>CONTINUE</td>
</tr>
<tr>
<td>0091</td>
<td>KK=KK+M</td>
</tr>
<tr>
<td>0092</td>
<td>AKK=AKK(K)</td>
</tr>
<tr>
<td>0093</td>
<td>DO 140 J=1,MP</td>
</tr>
<tr>
<td>0094</td>
<td>B(M,J)=B(M,J)*AKK</td>
</tr>
<tr>
<td>0095</td>
<td>C BACK SUBSTITUTION.</td>
</tr>
<tr>
<td>0096</td>
<td>DO 190 I=5,M</td>
</tr>
<tr>
<td>0097</td>
<td>J=MP1-I</td>
</tr>
<tr>
<td>0098</td>
<td>JF=J*(J+1)/2</td>
</tr>
<tr>
<td>0099</td>
<td>JP1=J+1</td>
</tr>
<tr>
<td>0100</td>
<td>DO 150 L=I,MP</td>
</tr>
<tr>
<td>0101</td>
<td>G(L)=O.</td>
</tr>
<tr>
<td>0102</td>
<td>DO 170 K=JP1,M</td>
</tr>
<tr>
<td>0103</td>
<td>JK=JK+K-1</td>
</tr>
<tr>
<td>0104</td>
<td>AKJ=AKJ(K)</td>
</tr>
<tr>
<td>0105</td>
<td>DO 160 L=1,MP</td>
</tr>
<tr>
<td>0106</td>
<td>G(L)=G(L)+AKJ*B(K,L)</td>
</tr>
<tr>
<td>0107</td>
<td>CONTINUE</td>
</tr>
<tr>
<td>0108</td>
<td>AJJJ=AJJJ(J)</td>
</tr>
<tr>
<td>0109</td>
<td>DO 180 J=1,MP</td>
</tr>
<tr>
<td>0110</td>
<td>B(J,L)=B(J,L)-AJJ*G(L)</td>
</tr>
<tr>
<td>0111</td>
<td>CONTINUE</td>
</tr>
<tr>
<td>0112</td>
<td>NER=O</td>
</tr>
<tr>
<td>0113</td>
<td>RETURN</td>
</tr>
<tr>
<td>0114</td>
<td>CONTINUE</td>
</tr>
<tr>
<td>0115</td>
<td>AKK=A(1)</td>
</tr>
<tr>
<td>0116</td>
<td>NER=1</td>
</tr>
<tr>
<td>0117</td>
<td>IF (ABS(AKK).LT.1.0E-20) GO TO 220</td>
</tr>
<tr>
<td>0119</td>
<td>AKK=1./AKK</td>
</tr>
<tr>
<td>0120</td>
<td>DO 210 J=1,MP</td>
</tr>
<tr>
<td>0121</td>
<td>B(I,J)=B(I,J)*AKK</td>
</tr>
<tr>
<td>0122</td>
<td>NER=0</td>
</tr>
<tr>
<td>0123</td>
<td>RETURN</td>
</tr>
<tr>
<td>0124</td>
<td>CONTINUE</td>
</tr>
<tr>
<td>0125</td>
<td>RETURN</td>
</tr>
<tr>
<td>0126</td>
<td>END</td>
</tr>
</tbody>
</table>

*OPTIONS IN EFFECT* NAME=MAIN OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
*VERSION 1.3.0 (01 MAY 80)  COPE12  SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 82.141/10.55.15  PAGE 4
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NOUAP NOFORMAT NOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS*  SOURCE STATEMENTS = 125, PROGRAM SIZE = 2044, SUBPROGRAM NAME =COPE12
*STATISTICS*  NO DIAGNOSTICS GENERATED
***** END OF COMPILATION ******
2992K BYTES OF CORE NOT USED
SUBROUTINE COPE13 (X,F,NX,N)

SOURCE TO CALCULATE \( F \) VALUES FOR LEAST SQUARES FIT TO QUADRATIC \( Y-Y_0 = D_Y - TRANSPOSE TIMES X + (1/2) X - TRANSPOSE TIMES H \) TIMES X.

BY G. N. VANDERPLAATS JAN., 1979.

X CONTAINS \( x-x_0 \).

\( M \) = MAXIMUM NUMBER OF COEFFICIENTS TO BE CALCULATED.

\( M \) .LE. \( (NX+1)/2 \).

\( DY \) COEF.

DO \( I=1,NX \)

\( F(I)=X(I) \)

\( H \) COEF.

\( = X1*X1, X2*X2, ... XN*XN, X1*X2 ... X1*XN ... \)

------------------------------------------------------------------

DIAGONAL ELEMENTS.

------------------------------------------------------------------

DO \( I=1,NX \)

IF (\( II.GT.M \)) GO TO 40

\( F(II)=.5*(X(I)*X(I)) \)

IF (!\( /II.X.LT.2) \) RETURN

------------------------------------------------------------------

OFF-DIAGONAL ELEMENTS.

------------------------------------------------------------------

\( NX=NX-1 \)

DO \( 30 J=I+1,NX \)

\( F(J)=X(I)*X(J) \)

30 CONTINUE

RETURN

END
DATA SET U477COPE14 AT LEVEL 002 AS OF 03/16/81
DATA SET U477COPE14 AT LEVEL 001 AS OF 02/13/81
DATA SET 91SSCOPE14 AT LEVEL 001 AS OF 07/10/80

SUBROUTINE COPE14 (NXAPRX, HF, NPTOT, RA, IA, LCCR, LOCI, TITLE, INOM, NDV, 00003
IPAPRX, ISCR2, TITLERM) 00004

COM110M IUIOSI 151, 161 00005
DIMENSION RA(1), IA(1), LCCR(1), LOCI(1), TITLE(1) 00006

ROUTINE TO PRINT RESULTS OF APPROXIMATE ANALYSIS/OPTIMIZATION. 00008

BY G. N. VANDERPLAATS JAN., 1979. 00010
NASA AAMES RESEARCH CENTER, MOFFETT FIELD, CALIF. 00012
------------------------------------------------------------------

TITLE. 00015
------------------------------------------------------------------
WRITE (I6,90) (TITLE(I),I=1,20) 00017
------------------------------------------------------------------
GLOBAL LOCATION OF X AND F(X) 00020

GLOBAL LOCATIONS OF X. 00022
H5=LOCI(5) 00023
M5=H5+NXAPRX-1 00024
WRITE (161,100) (IAtI)'I=M5,Mt15) 00025
GLOBAL LOCATIONS OF F(X). 00026
H6=LOCI(6) 00027
M6=H6+HF-1 00028
WRITE (161,110) (FA(I)'I=M1,M2) 00030
------------------------------------------------------------------
X-VALUES AND FUNCTIONS, F(X) 00032

X-VALUES. 00034
H1=LCCR(23)+3*NXAPRX+6 00035
N2=N1+NXAPRX-1 00036
WRITE (161,120) (RA(1)'I=N1,N2) 00038
F(X) VALUES. 00039
M1=LCCR(23)+3*NXAPRX+6 00040
N1=N1+NXAPRX-1 00041
WRITE (161,130) (RA(1)'I=M1,N2) 00043
------------------------------------------------------------------
TAYLER SERIES COEFFICIENTS. 00045

WRITE (161,140) 00047
NP=NPTOT-1 00048
WRITE (161,150) 00049
HDV=NXAPRX*NDV 00050
WRITE (161,160) 00052
DO 30 JJ=1,NDV 00054
M6=LOCI(6)+JJ-1 00055
M6=IA(M6) 00056
DO 30
C LINEAR TERMS.
N2=NXAPRX
IF (N2.GT.NP) N2=NP
N2=N2+1
IF (MAXTRM.LT.2) GO TO 20
C LOCATION OF FIRST OFF-DIAGONAL ELEMENT.
H3=H2+1
C LOCATION OF LAST OFF-DIAGONAL ELEMENT.
N4=NP
IF (N4.GT.HP) H4=HP
N4=H4-2*NXAPRX+H3-1
C LOCATION OF LAST OFF-DIAGONAL ELEMENT - THIS RCH.
LL=LOCATION OF LAST OFF-DIAGONAL ELEMENT - THIS RCH.
WRITE (161,190)
II=1
DO 10 I=H1,H2
WRITE (I61,150) II
II=II+1
LL=N3+NXAPRX-LL
IF (LL.GE.H3) WRITE (161,130) RA(I),(RA(J),J=H3,LL)
H3=LL+1
CONTINUE
30 CONTINUE
NBTAY=NBTAY+NXAPRX
CONTINUE
50 FORMAT (1115X,IB1!SUMMARY OF DESIGNS)
**VERSION 1.3.0 (01 MAY 80)**

**COPE14**

**SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)**

**DATE** 30.05.80

**PAGE** 3

---

<table>
<thead>
<tr>
<th>ISN</th>
<th>FORMAT</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0006</td>
<td>100 FORMAT (///SX,31HGLOBAL LOCATIONS OF X-VARIABLES)</td>
<td>00114</td>
</tr>
<tr>
<td>0007</td>
<td>110 FORMAT (5X,10I5)</td>
<td>00115</td>
</tr>
<tr>
<td>0008</td>
<td>120 FORMAT (///SX,2SHAPPROXIMATION IS BASED ON,15,H DESIGNS///SX,31HGLOBAL00116</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>00117</td>
</tr>
<tr>
<td>0009</td>
<td>130 FORMAT (5X,5E13.4)</td>
<td>00118</td>
</tr>
<tr>
<td>0010</td>
<td>140 FORMAT (///SX,10IS)</td>
<td>00119</td>
</tr>
<tr>
<td>0011</td>
<td>150 FORMAT (///SX,2SHAPPROXIMATION IS BASED ON,15,H DESIGNS///SX,31HGLOBAL00120</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>00121</td>
</tr>
<tr>
<td>0012</td>
<td>160 FORMAT (///SX,5HLINEAR TERMS, DEL F)</td>
<td>00122</td>
</tr>
<tr>
<td>0013</td>
<td>170 FORMAT (///SX,3HCOEFFICIENTS OF TAYLOR SERIES EXPANSION)</td>
<td>00123</td>
</tr>
<tr>
<td>0014</td>
<td>180 FORMAT (///SX,5HNON-LINEAR TERMS, H, BEGINNING WITH DIAGONAL ELEMENTS)</td>
<td>00124</td>
</tr>
<tr>
<td></td>
<td></td>
<td>00125</td>
</tr>
<tr>
<td>0096</td>
<td>END</td>
<td>00126</td>
</tr>
</tbody>
</table>

**OPTIONS IN EFFECT**

- NAME(MAIN)
- OPTIMIZE(3)
- LINECOUNT(60)
- SIZE(MAX)
- AUTOUBL(NONE)

**OPTIONS IN EFFECT**

- SOURCE EBDCIC NOLIST NODECK OBJECT NOHAP NOFORMAT GOSTMT NOKREF NOALC NOAHSF TERM IBM FLAG(I)

**STATISTICS**

- SOURCE STATEMENTS = 95, PROGRAM SIZE = 2632, SUBPROGRAM NAME =COPE14

**STATISTICS**

- NO DIAGNOSTICS GENERATED

**** END OF COMPILATION ****

2996K BYTES OF CORE NOT USED
SUBROUTINE COPE15 (XV,G,DF,A,ISC,IC,NN1,BLU,NX1,IOBJA,M,FNOM,FNEW,BTAY,NSR,IGFH,CT,CTL,INFO,NAC,ILCONA,NDV,OBJ)

DATA SET U477COPE15 AT LEVEL 001 AS OF 02/13/81
DATA SET 9166COPE15 AT LEVEL 001 AS OF 07/10/80

FUNCTION EVALUATION FOR APPROXIMATE OPTIMIZATION.

BY G. N. VANDERPLAATS
NASA-AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.

OBJECTIVE.

CALL COPE16 (NX1,XV,IOBJA,BTAY,NBR,MOBJ=-FHEW(IOBJA)*SGHOPT
IF (INFO.EQ.1) GO TO 20
GRADIENT OF OBJECTIVE.
CALL COPE17 (NX1,XV,IOBJA,BTAY,NBR,M,DF)
DO 10 I=1,NAC
DF(IJ=-DF(II)*SGHOPT
CONTINUE
IF (NCONA.LE.0) GO TO 80

CONSTRAINTS.

IF (INFO.EQ.2) HAC=0
ICOH=0
DO 70 I=1,NCONA
J=IGFH(I)
G=FHEW(J)
LAUER SOUND.
IF (BLU(I,II.LT.-1.0E+15) GO TO 40
ICON=ICON+1
G(ICON)=(G(ICON)-GG)/BLU(2,II)
IF (INFO.EQ.1) GO TO 40
IS THIS CONSTRAINT ACTIVE OR VIOLATED.
CTI=CT
IF (ISC(ICON.JG.T) CTI=CTL
IF (G(ICON).LT.CTII GO TO 40
ACTIVE CONSTRAINT. CALCULATE GRADIENT.

IC/src/ope150f77
ISN 0046  CTI=CT  00055
ISN 0047  IF (ISC(ICON).GT.0) CTI=CTI  00056
ISH 0049  IF (G(ICON).LT.CTI) GO TO 60  00057
ISH 0051  NAC=NAC+1  00059
ISH 0052  IC(NAC)=ICON  00060
ISH 0053  MX=M  00061
ISH 0054  IF (ISC(ICON).GT.0) MX=NDV  00062
ISH 0056  CALL COPE15 (HX1,XV,J,BTAY,NBR,MM,A(1,NAC))  00063
ISH 0057  FF=1./BLU(4,I)  00064
ISH 0058  DO 50 K=1,NDV  00065
ISH 0059  50 A(K,NAC)=A(K,NAC)*FF  00066
ISH 0060  60 CONTINUE  00067
ISH 0061  70 CONTINUE  00068
ISH 0062  80 CONTINUE  00069
ISH 0063  RETURN  00070
ISH 0064  END  00071

*OPTIONS IN EFFECT* NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
*OPTIONS IN EFFECT* SOURCE EBCDIC NO LIST OBJECT NOMAP NOFORMAT GOSTMT NOXREF NOALC TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 63, PROGRAM SIZE = 1728, SUBPROGRAM NAME = COPE15
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF COMPILATION *****

304K BYTES OF CORE NOT USED
DATA SET U477COP16 AT LEVEL 001 AS OF 02/13/81
DATA SET 9188COP16 AT LEVEL 001 AS OF 07/10/80

**REQUESTED OPTIONS:** SOURCE, NOPAM, NODXREF, NOLIST, NODECK, OPT(3), AUTOABBL(HCNE), NOALC

**OPTIONS IN EFFECT:** NAME(NAM) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOABBL(HCNE)

**SOURCE** EBCDIC NODXREF NODECK object NOPAM NOFORMAT GOSTLI1X NOXREF

**NAME** IIIAItII OPTIMIZE(3)

**360**

**LINECOUNT** 60 **SIZE** MAX **AUTOHEADER**

**SOURCE** EBCDIC NOLIST NODECK OBJECT NOPAM NOFORMAT GOSTLI1X NOXREF

DATA SET U477COP16 AT LEVEL 001 AS OF 02/13/81
DATA SET 9188COP16 AT LEVEL 001 AS OF 07/10/80

**ROUTINE TO EVALUATE FUNCTIONS APPROXIMATED BY TAYLOR SERIES**

**EXPRESSION UP TO SECOND ORDER.**

**BY G. N. VAIDENPLAATS**

**MAR., 1978.**

**NAVAL SHIP R D CENTER.**

**ARGUMENTS.**

**NX** - NUMBER OF INDEPENDENT VARIABLES CONTAINED IN X.

**X** - VECTOR OF DELTA VARIABLES X-X0, DIMENSIONED X(NX).

**NF** - NUMBER OF FUNCTIONS TO BE EVALUATED.

**FNOM** - NOMINAL FUNCTION VALUES ABOUT WHICH TAYLOR SERIES EXPANSION WAS DONE.

**FNEW** - NEW APPROXIMATED VALUES, DIMENSIONED FNEW(NF).

**B** - MATRIX OF TAYLOR SERIES COEFFICIENTS. B(I,J) CONTAINS DEL F, I=1,NX

**B(NX+I,J) CONTAINS DEL2 TERMS, I = 1,NX*(NX+1)/2.**

**MINIMUM DIMENSIONS** - B(M,NF).

**NBR** - DIMENSIONED RDS OF B.

**H** - TOTAL NUMBER OF COEFFICIENTS CURRENTLY USED.

**DIMENSION X(1), FNOM(1), FNEW(1), B(NBR,1)**

**DO 50 J=1,NF**

**------------------------------------------------------------------**

**CONSTANT TERM.**

**------------------------------------------------------------------**

**F=FNOM(J)**

**------------------------------------------------------------------**

**FIRST ORDER TERMS.**

**------------------------------------------------------------------**

**DO 10 I=1,NX**

**IF (I.GT.H) GO TO 40**

**F=F+B(I,J)*X(I)**

**------------------------------------------------------------------**

**SECOND ORDER TERMS.**

**------------------------------------------------------------------**

**II=1**

**II=II+1**

**IF (II.GT.M) GO TO 40**

**F=F+B(I,J)*X(I)**

**------------------------------------------------------------------**

**DIAGONAL ELEMENTS.**

**------------------------------------------------------------------**

**IF (NX.LT.2) GO TO 40**

**NXM1=NX-1**

**DO 30 I=1,NXM1**
IF (II.GT.MI) GO TO 40
DO 30 K=IP1,HX
II=II+1
XX=X(I)
30 F=F+B(II,J)*XX*X(K)
CONTINUE
RETURN
END

*OPTIONS IN EFFECT* NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOOBL(NONE)
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NOTMPE NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 29, PROGRAM SIZE = 878, SUBPROGRAM NAME =COPE16
*STATISTICS* NO DIAGNOSTICS GENERATED
****** END OF COMPILATION *****
3016K BYTES OF CORE NOT USED
SUBROUTINE COPE17 (NX,J,B,NBR,M,GRAD)

ROUTINE TO CALCULATE GRADIENT OF THE J-TH FUNCTION APPROXIMATED BY TAYLOR SERIES EXPANSION UP TO SECOND ORDER.


NASA AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.

F = F0 + DELF TIMES X + .5 X-TRANSPOSE TIMES DEL2F TIMES X.

AGREEMENTS.

NX - NUMBER OF INDEPENDENT VARIABLES CONTAINED IN X.

J - FUNCTION FOR WHICH GRADIENT INFORMATION IS CALCULATED.

B - MATRIX OF TAYLOR SERIES COEFFICIENTS.

B(I,J) CONTAINS DELF, I = 1,NX.

B(NX+I,J) CONTAINS DEL2, I = 1,NS*(NX+1)/2.

MINIMUM DIMENSIONS - BM,NF).

NBR - DIMENSIONED ROWS OF B.

M - TOTAL NUMBER OF COEFFICIENTS CURRENTLY USED.

GRAD - GRADIENT OF J-TH FUNCTION. OUTPUT.

DIMENTIONS X(1), B(NBR,1), GRADO

FIRST ORDER TERMS.

DO 10 I=1,NX

IF (I.GT.M) GO TO 40

GRAD(I)=B(I,J)

10

SECOND ORDER TERMS.

II=NX

II=NX

DIAGONAL ELEMENTS.

DO 20 I=1,NX

II=I+1

IF (I.I.GT.M) GO TO 40

GRAD(I)=GRADO(I)+B(I,J)*X(I)

20

GRAD(I)=GRADO(I)+B(I,J)*X(I)

CONTINUE

RETURN

OFF-DIAGONAL ELEMENTS.

IF (NX.LT.2) GO TO 40

DO 30 I=1,NX

30

GRAD(I)=GRADO(I)+B(I,J)*X(I)

CONTINUE

RETURN
**ROUTINE TO PRINT OPTIMIZATION RESULTS**

**BY G. N. VAU:DERPLAATS M., 1979**

**NASA Ames Research Center, Moffett Field, Calif.**

**CC/CNS**/UIGOS/151, 161

**DIMENSION RA(1), IA(1), LOCRI(1), LOCII(1), ARRAY(1)**

**OBJECTIVE FUNCTION AND DESIGN VARIABLES.**

**WRITE (161, 30) IOBJ, ARRAY(IOBJ)**

**DO 10 I = 1, NDVTOT**

**DESIGN VARIABLE NUMBER.**

**DO 10 I = 1, NDVTOT**

**GLOBAL LOCATION.**

**IG = IA(I)**

**MULTIPLIER.**

**AMULT = RA(1)**

**LOWER BOUND.**

**BL = AMULT * RA**

**VALUE.**

**XX = ARRAY(I)**

**UPPER BOUND.**

**BU = AMULT * RA**

**IDENTIFICATION NUMBER.**

**WRITE (161, 50) I, IDV, IG, BL, XX, BU**

**CONTINUE**

**IF (NCONA .EQ. 0) RETURN**

**CONSTRAINTS.**

**WRITE (161, 50) I, H3, M3, XX**

**GLOBAL LOCATION.**

**IG = IA(H3)**

**LOWER BOUND.**

**BL = RA**

**VALUE.**

**XX = ARRAY(I)**

**UPPER BOUND.**

**BU = RA**

**IDENTIFICATION NUMBER.**
ISH 0038
JD=ID

ISH 0039
IF (BL.GT.-1.0E+15) ID=ID+1

ISH 0041
IF (BU.LT.1.0E+15) ID=ID+1

ISH 0043
WRITE (16I,60) JD,ID,BL,XX,BU

ISH 0044
20 CONTINUE

ISH 0045
RETURN

ISH 0046
C
C -----------------------------------------------
C FORMATS
C
ISH 0046
30 FORMAT (1H1,4X,20HOPTIMIZE RESULTS//ISX,12HOBJECTIVE FUNCTION/0064
1SX,15HGLOBAL LOCATION,ISX,14HFUNCTON VALUE,E12.5//5X,16HDESIGN0065
2 VARIABLES//14X,5I5D. V.,5X,6HGLOBAL,7X,5HLINDER,23X,5HUPPER/8X,2HID0066
3,5X,3HNO.,5X,6HVAR. NO.,6X,5HBOUND,9X,5HVALUE,9X,5HBOUND)

ISH 0047
40 FORMAT (11O,I7,I11,3X,3EI4.S)

ISH 0048
50 FORMAT (///5X,16HDESIGN CONSTRAINTS//15X,6HGLOBAL,7X,5HLINDER,23X,5H0069
1UPPER/9X,2HID,4X,6HVAR. NO.,6X,5HBOUND,9X,5HVALUE,9X,5HBOUND)

ISH 0049
60 FORMAT (11O,I9,3X,3EI4.5)

ISH 0050
END

*OPTIONS IN EFFECT*NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOGBL(NONE)
*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NODECK OBJECT NODUMP NAMECOPE18
*STATISTICS* SOURCE STATEMENTS = 49, PROGRAM SIZE = 1408, SUBPROGRAM NAME =COPE18
*STATISTICS* NO DIAGNOSTICS GENERATED
***** END OF Compilation *****

3012K BYTES OF CORE NOT USED
SUBROUTINE SIMCON (NOV, X, VLB, VUB, NCON, ITMAX, IPRINT, ISCAL, DELFUN, QA00002
IBFUN, Fun, IWK, UWK, 03J, G, IER)

--- PURPOSE ---
MINIMIZE OBJ AS A FUNCTION OF X(I), I=1, NOV
SUBJECT TO
G(J).LE.0, J=1, NCON
Vlb(I).LE.X(I).LE.VUB(I), I = 1, NDV

... NOTES ...
NCON MAY BE ZERO.
Vlb(I) IS IGNORED IF Vlb(I).LT.-1.0E+15.
Vub(I) IS IGNORED IF Vub(I).GT.1.0E+15.
IF NCON = 0 AND IT IS NOT ESSENTIAL TO LIMIT THE VARIABLES, X(I)
THEN SET Vlb(I)=-1.0E+16 AND Vub(I)=1.0E+16. THIS WILL IMPROVE
THE EFFICIENCY OF THE UNCONSTRAINED OPTIMIZATION.

--- REFERENCE ---
G. N. VANDERPLAATS
COINMIN - A FORTRAN PROGRAM FOR CONSTRAINED FUNCTION MINIMIZATION

--- ARGUMENTS ---
NOV - NUMBER OF INDEPENDENT DESIGN VARIABLES.
X - ARRAY OF INITIAL DESIGN VARIABLES.
REQUIRED DIMENSION = NOV + 2.
VLB - ARRAY OF LOWER BOUNDS ON X.
IF Vlb(I).LT.-1.0E+15 THE ITH VARIABLE IS NOT BOUNDED.
REQUIRED DIMENSION = NOV +2.
VUB - ARRAY OF UPPER BOUNDS ON X.
IF Vub(I).GT.1.0E+15 THE ITH VARIABLE IS NOT BOUNDED.
REQUIRED DIMENSION = NOV +2.
NCON - NUMBER OF CONSTRAINT VALUES STORED IN ARRAY G.
NCON MAY BE ZERO.
ITMAX - MAXIMUM PERMISSIBLE ITERATIONS IN COINMIN.
DEFAULT = 20.
DEFAULT = 0.
DEFAULT = 5.
SUGGESTED VALUE IS IPRINT = 3.
ISCAL - SCALING PARAMETER. IF ISCAL = 0 NO SCALING IS DONE.
IF ISCAL = 1 THE DESIGN VARIABLES ARE SCALED DURING
OPTIMIZATION. RECOMMENDED ISCAL = 1.
 DELFUN - CONVERGENCE TOLERANCE ON FRACTIONAL CHANGE IN OBJECTIVE
FUNCTION. IF OBJ DOES NOT CHANGE BY MORE THAN Delfun...
FOR THREE CONSECUTIVE ITERATIONS, OPTIMIZATION IS TERMINATED.

IF DELFUN = 0 IS INPUT, DELFUN = 0.0001 IS USED.

CONVERGENCE TOLERANCE ON ABSOLUTE CHANGE IN OBJECTIVE FUNCTION. IF OBJ DOES NOT CHANGE BY 0.0001 IN MAGNITUDE FOR THREE CONSECUTIVE ITERATIONS, OPTIMIZATION IS TERMINATED.

IF DABFUN = 0 IS INPUT, DABFUN = 0.001*ABS(INITIAL OBJ) IS USED.

NAME OF EXTERNAL SUBROUTINE WHICH EVALUATES OBJECTIVE AND CONSTRAINT FUNCTIONS.

REAL WORK ARRAY.

DIMENSION OF WORK ARRAY.

DIMENSION OF WORK ARRAY.

NAME OF EXTERNAL SUBROUTINE WHICH EVALUATES OBJECTIVE AND CONSTRAINT FUNCTIONS.

INTEGER WORK ARRAY.

DIMENSION OF WORK ARRAY.

DIMENSION = NCON + 2*NDV + 2*(NDV*NDV) + 2*NDV + 16

NAME OF EXTERNAL SUBROUTINE WHICH EVALUATES OBJECTIVE AND CONSTRAINT FUNCTIONS.

OBJECTIVE FUNCTION ASSOCIATED WITH X.

ARRAY CONTAINING THE NCON CONSTRAINT VALUES ASSOCIATED WITH X. IF NCON = 0, G IS NOT CALCULATED.

NUMBER OF INDEPENDENT DESIGN VARIABLES.

NUMBER OF CURRENT VALUES OF THE NDV DESIGN VARIABLES.

DIMENSION = NDV + 2.

NUMBER OF CONSTRAINT VALUES STORED IN G.

NCON MAY BE ZERO.

OBJECTIVE FUNCTION ASSOCIATED WITH X.

ARRAY CONTAINING THE NCON CONSTRAINT VALUES ASSOCIATED WITH X. IF NCON = 0, G IS NOT CALCULATED.

DIMENSION = NCON + 2*NDV IF NCON.GT.0.

DIMENSION = 1.

ERROR CODE.

IER = 0, NO STORAGE ERROR.

IER = 1, ARRAY WK OR WK IS NOT DIMENSIONED LARGE ENOUGH.

EXTERNAL USER-SUPPLIED ROUTINE TO EVALUATE OBJECTIVE AND CONSTRAINT FUNCTIONS.

FUNCTIONS.

SUBROUTINE FUN (NDV,X,NCON,OBJ,G)

INPUT

NDV - NUMBER OF INDEPENDENT DESIGN VARIABLES.

X - ARRAY CONTAINING CURRENT VALUES OF THE NDV DESIGN VARIABLES.

DIMENSION = NDV + 2.

NCON - NUMBER OF CONSTRAINT VALUES STORED IN G.

NCON MAY BE ZERO.

OUTPUT

OBJ - OBJECTIVE FUNCTION ASSOCIATED WITH X.

G - ARRAY CONTAINING THE NCON CONSTRAINT VALUES ASSOCIATED WITH X. IF NCON = 0, G IS NOT CALCULATED.

DIMENSION = NCON + 2*NDV IF NCON.GT.0.

DIMENSION = 1.

IER = 0, NO STORAGE ERROR.

IER = 1, ARRAY WK OR WK IS NOT DIMENSIONED LARGE ENOUGH.
**INITIALIZE COMMON PARAMETERS**

- NV=NDV
- NCC=NCON
- IPRINT=IPRINT
- ITMAX=ITMAX
- DLFUN=DLFUN
- DDFUN=DDFUN
- CT=0.
- CTMIN=CTMIN
- CTLIN=CTLIN
- CTLIN=CTLIN
- APLNH=APLNH
- ITHAX=ITHAX
- DABFUN=DABFUN
- CT=0.
- CTIN=CTIN
- CT=f=CT=f
- APLN=APLN
- THETA=THETA
- NFAG=NFAG
- NCON=NCON+1
- NSCl=IIDV+1
- IF (ISCAl.EQ.O) NSCAl=O
- NSIOE=O
- DO 10 =1,NDV
- 10 CONTINUE
- FCH=.001
- FDCHN=.001

**PRINT INPUT INFORMATION**

- WRITE (6,170)
- WRITE (6,160) NDV,NCC,ITMAX,IPRINT,ISCAl,DLFUN,DABFUN
- WRITE (6,150)
- WRITE (6,140) I,VLBI(I),X(I),VUB(I)
- CONTINUE
- FCH=.001
- FDCHN=.001
- IF (IPRINT.LT.1) GO TO 30

**COMMON ARRAY DIMENSIONS**

- IER=1
- N1=NDV+2
- IF (NCON.EQ.0.AND.NSIDE.EQ.0) GO TO 40
- N2=NCON+2*NDV
- N3=N1
- N4=N1
- N5=2*N1
- GO TO 50
- CONTINUE
- SPECIAL CASE. NCON = NSIDE = 0.
- N2=1
- N3=1
- N4=IDV
- N5=1
- CONTINUE
- CONTINUE
- CONTINUE
- IREQE=3*N1+2*N2+N3*(N1+N3)+N4
- N1=NDV+2
ISN 0057  NREQI=N2+N3+N5
ISN 0058  NREQG=NREQQ
ISN 0059  NREQA=NREQA
ISN 0060  IF (NREQR.GT.NWK.OR.NREQI.GT.NWK) GO TO 130
ISN 0061  IF (NCON.EQ.0.AND.NSIDE.EQ.0) GO TO 80
C FIND MAXIMUM POSSIBLE N3.
ISN 0062  II=II+1
ISN 0063  N3SAV=N3
ISN 0064  DO 60  I=II,N21
ISN 0065  11=1
ISN 0066  N4=N3
ISN 0067  IF (N4.LT.NDV) N4=NDV
ISN 0068  N5=2*N4
ISN 0069  NREQR=3*N1+2*N2+3*(N1*N3)+N4+N5
ISN 0070  IF (NREQR.GT.NWK.OR.NREQI.GT.NWK) GO TO 70
ISN 0071  N3SAV=I
ISN 0072  CONTINUE
C STORAGE ALLOCATION
ISN 0073  NNSCAL=1
ISN 0074  NDF=NNSCAL+N1
ISN 0075  NA=NDF+N1
ISN 0076  N5=NA+N1+N3
ISN 0077  NG1=6*4+4
ISN 0078  NG2=NG1+N2
ISN 0079  NB=NG2+N2
ISN 0080  NC=NB+N3+N3
ISN 0081  NS=NC+NC
C REQUIRED STORAGE.
ISN 0082  NREQR=NC+N4-1
ISN 0083  NREQI=NNS1+N5-1
ISN 0084  IF (NREQR.GT.NWK.OR.NREQI.GT.NWK) GO TO 130
ISN 0085  IF (NCON.LE.0) GO TO 100
C DEFINE ISC ARRAY SO ALL CONSTRAINTS ARE NONLINEAR.
ISN 0086  N=NISC
ISN 0087  DO 90  I=1,NCON
ISN 0088  IWKINI=O
ISN 0089  90  N=N+1
ISN 0090  WRITE (16,210) NWK,NREQRl,NREQR,NIWK,NREQI
ISN 0091  CONTINUE
C OPTIMIZATION
ISN 0092  IGOTO=O
ISN 0093  GOTO 120
ISN 0094  CALL COMIN (X,VLB,VUB,G,WK(NNSCAL),WK(NDF),WK(NA),WK(NS),WK(NS1),00231)
HAS STORAGE BEEN EXCEEDED.

IF (NSG.GE.N3) GO TO 100

CALL FUN (HDV,X,NCON,OBJ,G)

IF (IGOTO.GT.0) GO TO 120

IER=0

RETURN

------------------------------------------------------------------

REQUIRED STORAGE EXCEEDS AVAILABLE STORAGE

------------------------------------------------------------------

INSUFFICIENT STORAGE TO START OPTIMIZATION.

WRITE (6,220) NWK,NREQR,NREQR,HIWK,NREQI,NREQI

RETURN

INSUFFICIENT STORAGE FOR GRADIENTS DURING OPTIMIZATION.

WRITE (6,220) N3SAV=N3

N3=1

IF (ICON.EQ.0) GO TO 160

DO 150 I=1,NCON

IF (GIII.GT.-.21 N3=I3+1

150 CONTINUE

IF (NSIDE.GT.0) H3=H3+NDV

IF (H3.LT.H3SAV) N3=H3SAV+NDV

N4=H3

IF (N4.LT.NDVE

N4=NDV

NREQR=3*H1+2*N2+H3*IH1+H31+H4

NREQI=I2+H3+2*N4

WRITE (6,250) NWK,NIWK,NREQR,NREQI

RETURN

------------------------------------------------------------------

FORATS

------------------------------------------------------------------

1 FORMAT (1H1,1/12X,29(2H1/1/12X,1H*,55X,1H=1/12X,1H*,22X,11HS 100273
1 M C O N 22X,1H*,55X,1H=1/12X,1H*,10X,16HFORTRAN PROGRAM FC00274
28,16X,1H=/12X,1H*,55X,1H=1/12X,1H*,16X,23HSIMPLIFIED COMMENT USAGE,100275
36X,1H=1/12X,1H*,55X,1H=1/12X,2H=1)

1 FORMAT (1H1,1/12X,29(2H1/1/12X,1H*,55X,1H=1/12X,1H*,22X,11HS 100273
1 M C O N 22X,1H*,55X,1H=1/12X,1H*,10X,16HFORTRAN PROGRAM FC00274
28,16X,1H=/12X,1H*,55X,1H=1/12X,1H*,16X,23HSIMPLIFIED COMMENT USAGE,100275
36X,1H=1/12X,1H*,55X,1H=1/12X,2H=1)

1 FORMAT (1H1,1/12X,29(2H1/1/12X,1H*,55X,1H=1/12X,1H*,22X,11HS 100273
1 M C O N 22X,1H*,55X,1H=1/12X,1H*,10X,16HFORTRAN PROGRAM FC00274
28,16X,1H=/12X,1H*,55X,1H=1/12X,1H*,16X,23HSIMPLIFIED COMMENT USAGE,100275
36X,1H=1/12X,1H*,55X,1H=1/12X,2H=1)
ISN 0156 240 FORMAT (/22X,5HOPTIMIZATION CANNOT PROCEED//22X,28HRETURNING TO 00292
ICALLING PROGRAM//12X,47(1H*) ) 00293
ISN 0157 250 FORMAT ( /12X,66HAVAILABLE STORAGE FOR GRADIENT INFORMATION EXCEEDS 00294
1AVAILABLE STORAGE//12X,5HARRAY,14X,7HNSK(NIK),5X,7HNSK(NIK)/12X,9H00295
2DIMENSION,15X,16X,16//12X,19HSUGGESTED DIMENSION,5X,16X,16 ) 00296
ISN 0158 END 00297
*OPTIONS IN EFFECT* NAME.MAIN OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTODSL(NONE)
*OPTIONS IN EFFECT* SOURCE EBCDIC NOLIST NODECK OBJECT NOHEADER NOFORMAT GOSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 157, PROGRAM SIZE = 3760, SUBPROGRAM NAME =SIMCON
*STATISTICS* NO DIAGNOSTICS GENERATED
******** END OF COMPILATION ********
2996K BYTES OF CORE NOT USED
REQUESTED OPTIONS: SOURCE, NOLIST, NOXREF, NOCALC, OPT(3), AUTOCL(NONE), NOALC
OPTIONS IN EFFECT: HANDMAIN OPTIMIZE(3) LINESCOUNT(60) SIZE(MAX) AUTOCL(NOIDE)

ВЕРСИЯ 1.3.0 (01 МАЙ 80) СИСТЕМА/370 FORTRAN Х ЭНТЕНДИД (ЕНАНСИД).

**REQUESTED OPTIONS:**
- SOURCE, NOLIST, NOXREF, NOCALC, OPT(3), AUTOCL(NONE), NOALC

**OPTIONS IN EFFECT:**
- HANDMAIN OPTIMIZE(3) LINESCOUNT(60) SIZE(MAX) AUTOCL(NOIDE)

**SOURCE EBCDIC NOLIST NODECK OBJECT**
- NOLIST GOSTH蛞 THEREXTENDED
- OPTIMIZE(3)

**LINECOUNT(60) SIZE(MAX) AUTOCL(NOIDE)**

**SOURCE**
- EBCDIC NOLIST NODECK OBJECT

**NOXREF, NOCALC, NOANSF**
- TERM IBM FLAG(I)

**DATE**
- 82.141/10.55.45

**PAGE**
- 1

**SOURCE**
- EBCDIC NOLIST NODECK OBJECT

**NOXREF, NOCALC, NOANSF**
- TERM IBM FLAG(I)

**REQUESTED OPTIONS:**
- SOURCE, NOLIST, NOXREF, NOCALC, OPT(3), AUTOCL(NONE), NOALC

**OPTIONS IN EFFECT:**
- HANDMAIN OPTIMIZE(3) LINESCOUNT(60) SIZE(MAX) AUTOCL(NOIDE)

**SOURCE EBCDIC NOLIST NODECK OBJECT**
- NOLIST GOSTH蛞 THEREXTENDED
- OPTIMIZE(3)

**LINECOUNT(60) SIZE(MAX) AUTOCL(NOIDE)**

**SOURCE**
- EBCDIC NOLIST NODECK OBJECT

**NOXREF, NOCALC, NOANSF**
- TERM IBM FLAG(I)

**DATE**
- 82.141/10.55.45

**PAGE**
- 1

**SOURCE**
- EBCDIC NOLIST NODECK OBJECT

**NOXREF, NOCALC, NOANSF**
- TERM IBM FLAG(I)

**REQUESTED OPTIONS:**
- SOURCE, NOLIST, NOXREF, NOCALC, OPT(3), AUTOCL(NONE), NOALC

**OPTIONS IN EFFECT:**
- HANDMAIN OPTIMIZE(3) LINESCOUNT(60) SIZE(MAX) AUTOCL(NOIDE)

**SOURCE EBCDIC NOLIST NODECK OBJECT**
- NOLIST GOSTH蛞 THEREXTENDED
- OPTIMIZE(3)

**LINECOUNT(60) SIZE(MAX) AUTOCL(NOIDE)**

**SOURCE**
- EBCDIC NOLIST NODECK OBJECT

**NOXREF, NOCALC, NOANSF**
- TERM IBM FLAG(I)

**DATE**
- 82.141/10.55.45

**PAGE**
- 1

**SOURCE**
- EBCDIC NOLIST NODECK OBJECT

**NOXREF, NOCALC, NOANSF**
- TERM IBM FLAG(I)
IF (NORDA.LT.NDRA.AND.NDRA.LT.NDIA) GO TO 10
WRITE (360) NORDA,NDRA,NDIA,NDIA
GO TO 340
C READ USER INPUT.
CALL ANALIZ (ICALC)
IF (ICALC.LT.1.OR.ICALC.GT.6) GO TO 340
EXECUTION
IF (ICALC.LT.2.AND.ICALC.LT.5) GO TO 60
------------------------------------------------------------------
IF (ABS(X(I).GT.0.0) OVER-RIDE USER INPUT OF DECISION VARIABLES FOR
OPTIMIZATION.
------------------------------------------------------------------
DO 40 I=1,NDV
XX=ABS(RA(I))
OVER-RIDE ANALIZ INPUT.
NS=LOCRI(5)
M2=LOCRI(2)
DO 20 J=1,NDVTOT
N1=IA(M2)
M2=M2+1
IF (N1.LE.I) GO TO 20
N1=IA(J)
IF (XX.LT.1.0E-10) GO TO 30
ARRAY(N1)=RA(I)*RA(N5)
N5=N5+1
GO TO 40
RA(I)=ARRAY(N1)/RA(N5)
CONTINUE
M2=LOCRI(2)
NS=LOCRI(5)
DO 50 I=1,NDVTOT
II=IA(M2)
M=IA(I)
ARRAY(M)=RA(M)*RA(NS)
115=NS+1
M2=M2+1
CONTINUE
IF (ICALC.LT.6) GO TO 290
------------------------------------------------------------------
TRANSFER DESIGN VARIABLES TO ARRAY.
M2=LOCRI(2)
NS=LOCRI(5)
DO 50 I=1,NDVTOT
II=IA(M2)
M=IA(I)
ARRAY(M)=RA(M)*RA(NS)
115=NS+1
M2=M2+1
CONTINUE
IF (ICALC.LT.3.AND.ICALC.LT.5) GO TO 80
------------------------------------------------------------------
TRANSFER NOMINAL VALUES OF SENSITIVITY VARIABLES TO ARRAY.
M16=LOCRI(16)
M17=LOCRI(17)
M15=LOCRI(15)
DO 70 I=1,NSV
MN=IA(M16)
M16=M16+1
M15=M15+1
CONTINUE
IF (ICALC.LT.6) GO TO 290
C INITIALIZATION FOR APPROXIMATE ANALYSIS/OPTIMIZATION

C

IFS 0079
C ANALYZ INPUT DEFINES AN X-VECTOR.

IFS 0081
M5=LOCI(5)

IFS 0082
N23=LOCRTL23)

IFS 0083
DO 120 I=1,NXAPRX

IFS 0084
J=IA(MS)

IFS 0085
C IS THIS A DESIGN VARIABLE.

IFS 0086
DO 99 K=1,NDVTOT

IFS 0087
KK=K

IFS 0088
IF (IA(K).EQ.J) GO TO 100

IFS 0089
90 CONTINUE

IFS 0090
NO.

IFS 0091
AMULT=1.

IFS 0092
GO TO 110

IFS 0093
YES.

IFS 0094
K=LOCRTL5)+KK-1

IFS 0095
AMULT=RA(K)

IFS 0096
RA(N23)=ARRAY(J)/AMULT

IFS 0097
N23=N23+1

IFS 0098
CONTINUE

IFS 0099
IF (NPS.GT.0 .OR. NPS.F.GT.0) GO TO 190

IFS 0100
C ONLY ONE DESIGN VECTOR IS AVAILABLE. CREATE A SECOND X-VECTOR

IFS 0101
SO OPTIMIZATION CAN PROCEED.

IFS 0102
N23=LOCRTL23)

IFS 0103
DO 120 I=1,NXAPRX

IFS 0104
C GLOBAL LOCATION.

IFS 0105
IG=IA(M5)

IFS 0106
N5=NS+1

IFS 0107
C PROPOSED X-VALUE.

IFS 0108
XX=1.1*RA(N23)

IFS 0109
IF (ABS(XX).LT.1.0E-10) XX=.1

IFS 0110
C IS THIS A DESIGN VARIABLE.

IFS 0111
N5=LOCRTL5)

IFS 0112
DO 140 J=1,NDVTOT

IFS 0113
JJ=J

IFS 0114
AH=RA(N5)

IFS 0115
IF (IA(J).EQ.IG) GO TO 150

IFS 0116
C NO.

IFS 0117
N5=N5+1

IFS 0118
CONTINUE

IFS 0119
GO TO 170

IFS 0120
C INSURE XX IS WITHIN BOUNDS.

IFS 0121
ID=LOCRTL2)+JJ-1

IFS 0122
ID=IA(ID)

IFS 0123
N2=LOCRTL2)+ID-1

IFS 0124
N3=LOCRTL3)*ID-1

IFS 0125
BL=RA(N2)*ABS(AH)

IFS 0126
BU=RA(N3)*ABS(AH)

IFS 0127
IF (BL.LE.BU) GO TO 160

IFS 0128
BL=BU

IFS 0129
1st! 0130
C

IFS 0131
C

IFS 0132
C

IFS 0133
C

IFS 0134
C

IFS 0135
C

IFS 0136
C

IFS 0137
C

IFS 0138
C

IFS 0139
C

IFS 0140
C

IFS 0141
C

IFS 0142
C

IFS 0143
C

IFS 0144
C

IFS 0145
C

IFS 0146
C

IFS 0147
C

IFS 0148
C

IFS 0149
C

IFS 0150
C

IFS 0151
C

IFS 0152
C

IFS 0153
C

IFS 0154
C

IFS 0155
C

IFS 0156
C

IFS 0157
C

IFS 0158
C

IFS 0159
C

IFS 0160
C

IFS 0161
C

IFS 0162
C

IFS 0163
C

IFS 0164
C

IFS 0165
C

IFS 0166
C

IFS 0167
C

IFS 0168
C

IFS 0169
C

IFS 0170
C

IFS 0171
C

IFS 0172
C
IF (XX.LT.BL) XX=BL
IF (XX.GT.BU) XX=BU

DX=R(N+23)-XX
IF (ABS(DX).LT.1.0E-6) XX=I.0
IF (ABS(XX).LT.1.0E-6) XX=.001
RA(N+24)=XX
N23=N23+1
N24=N24+1

CONTINUE

READ ISCR2 (RA(I),I=NXI,NXIJ)

C TRANSMIT X-VALUES.
M6=LOC(6)
II=HY
DO 220 I=I,NXAPRX
II=IA(M6)
ARRAY(II)=RA(I1)
II=II+1

C ANALIZE.
NAN2=NAN2+1
CALL ANALIZE (ICALC)

C PUT FUNCTION VALUES IN Y-ARRAY.
M6=LOCI(6)
II=HY
DO 230 I=1,1,NF
II=IA(M6)
RA(I1)=ARRAY(II)
II=II+1

M6=M6+1

C Y-VECTOR.
NXI=NXI+NXAPRX
NY=NY+NF-1
READ ISCR2 (RA(I),I=NY,NYJ)

C X-VECTOR.
READ ISCR2 (RA(I),I=NXI,NXIJ)
NXI=NXI+NXAPRX
NY=NY+NYAPRX+NYPTOT
DO 280 I=1,NPTOT
NXIJ=NXI+NXAPRX-1
NYJ=NY+IIF-1
C X-VECTOR.
WRITE (ISCR2) (RA(J),J=HXI,NXIJ)
C V-VECTOR.
WRITE (ISCR2) (RA(JJ,J=NY,HYJI
NXI=NXIJ+1
280 HY=NYJ+1
290 CONTINUE
GO TO (300,300,310,320,310,330)NCALC
300 NAN2=NAN2+1
CALL ANALIZ (ICALC)
310 CONTINUE
C ARRAY STARTING LOCATIONS.
C SEN.
N1=LOCRI(5)
N9=LOCRI(16)-N1
N9=NSOBJ
N10=NSV
N4=LOCRI(17)
C TEMP.
N5=LOCRI(23)
CALL COPE04 (ARRAY,RA,IA,NARRAY,NDRA,NDIA)
C OUTPUT RESULTS.
CALL COPE02 (ARRAY,RA,IA,NARRAY,NDRA,NDIA)
CALL COPE18 (IOBJ,NDVTOT,NCOHA,RA,IA,LOCRI,LOCI,ARRAY)
CALL ANALIZ (JCALC)
CALL ANALIZ (JCALC)
GO TO 340
C SENSITIVITY ANALYSIS
C ARR.
C SEN.
C TEMP.
C OUTUT RESULTS.
CALL COPE05 (RA,IA,NDRA,NDIA,ISCR1)
GO TO 340
C---~--------------------------------------------------------------
C VARIABLE FUNCTION SPACE
C------------------------------------------------------------------
ISN  0228  CALL COPE06 (ARRAY,RA,IA,NARRAY,NDR,A,NYIA)
C OUTPUT RESULTS.
ISN  0229  CALL COPE07 (RA,IA,NDR,A,NDIA,ISCR1)
ISN  0230  GO TO 340
ISN  0231  330 CONTINUE
C------------------------------------------------------------------
C APPROXIMATE ANALYSIS/OPTIMIZATION.
C------------------------------------------------------------------
ISH  0232  CALL COPE09
C OUTPUT RESULTS.
ISH  0233  CALL COPE14 (NXAPRX,NF,NPTOT,RA,IA,LOCR,LOCI,TITLE,INCR,NDV,IPAPRX)
ISH  0234  IF (KHAX.LT.0) GO TO 340
ISH  0235  CALL COPE10 (IOBJ,NVTO,NCONA,RA,IA,LOCR,LOCI,ARRAY)
ISH  0236  IF (KHAX.LT.0) GO TO 340
ISH  0237  NAH3=NAH3+1
ISH  0238  CALL AHALIZ (JCALC)
ISH  0239  340 CONTINUE
ISH  0240  WRITE (161,350) NAN2,NAN3
ISH  0241  REIND ISCRI
ISH  0242  REIND ISCRI
ISH  0243  CALL MYTIME (ITIME)
ISH  0244  ITIMEC = 9
ISH  0245  WRITE (26,3000) ITIMEC , ITIME
ISH  0246  STOP
C------------------------------------------------------------------
C FORMATS
C------------------------------------------------------------------
ISH  0247  350 FORMAT (1H1,4X,2HPROGRAM CALLS TO AUALIZ//5X,5HICALC,2X,5HCALLS//,100322
ISH  0248  360 FORMAT (1H1,4X,2HPROGRAM CALLS TO AUALIZ//5X,5HICALC,2X,5HCALLS//,100323
ISH  0249  END

+OPTIONS IN EFFECT=NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTOBAL(NONE)
+OPTIONS IN EFFECT=SOURCE EBCDIC NOLIST NODECK OBJECT NOMAP NOFORMAT GSTMT NOXREF NOALC NOANSF TERM IBM FLAG(I)
+STATISTICS= SOURCE STATEMENTS = 246, PROGRAM SIZE = 3980, SUBPROGRAM NAME = MAIN
+STATISTICS= NO DIAGNOSTICS GENERATED
***** END OF COMPILATION *****

2968K BYTES OF CORE NOT USED
APPENDIX C

REVISED FORTRAN FOR SUPERHYBRID BLADE DEMONSTRATION CASE
C DATA SET U492STAE8 AT LEVEL THP AS OF 05/21/82
C DATA SET U492STAE8 AT LEVEL 018 AS OF 01/20/82
C DATA SET U492STAE8 AT LEVEL 019 AS OF 10/26/81
C DATA SET U492STAE8 AT LEVEL 012 AS OF 03/24/81
C DATA SET U492STAE8 AT LEVEL 012 AS OF 02/14/81

ISN 0002
C SUBROUTINE STABBL

ISN 0003
C COMMON /BLKCH/ XP(3,53,21), NP(21), IND(21), INP

ISN 0004
C COMMON /BLK 2/ HACR, NHCRI, NSTA, NSTA1, POISHN,MST2,MST3

ISN 0005
C COMMON /BLK 4/ RPR, XNODCR, RP, PI, TERM, HDS

ISN 0006
C COMMON /BLK 7/ AREA(21), CF(21), AA(21), AKG(21), IT7(21)

ISN 0007
C COMMON /BLK12/ XBARX(21), XMAX(21), XMIN(21), YBRY(21),
I TLYA(21)

ISN 0008
C COMMON /BLK13/ ALPHM(21), XSCL(21), YSC(21), XX(21), YY(21)

ISN 0009
C COMMON /BLK9/ SPS(21), ITLT(18), VAR(235), THAX(21), HALPHA(21)

ISN 0010
C COMMON /BLK A/ ALPHA(21), HA(21), HR(21), HALPHA(21)

ISN 0011
C HINTF(21)

ISN 0012
C COMMON /BLK 9/ DELTAZ(21), DELT(21), L(21), R(21), SHU(21)

ISN 0013
C COMMON /INPUT/ XSAVE(1000), YSAVE(1000), ZSAVE(1000), TSAVE(1000)

ISN 0014
C COMMON /SC/ XSCG(V(21)), YSCG(V(21)), ZSCG(V(21))

ISN 0015
C COMMON /SSAVE/(21), FSTATIC(21), ASAVE(21)

ISN 0016
C COMMON /ALPHA/ LDEL, DDEL, DROGO, GP, DHO, TIO, TIO, TLO, NCD

ISN 0017
C COMMON /YNS/ XYN(53,21), YMN(53,21), YNS(53,21), TYN(53,21)

ISN 0018
C COMMON /ZNS/ ZYN(53,21), ZMN(53,21), ZNS(53,21)

ISN 0019
C COMMON /TNS/ TYN(53,21), ZYN(53,21)

ISN 0020
C COMMON /ASAVE/ XSAVE(1000), YSAVE(1000), ZSAVE(1000), TSAVE(1000)

ISN 0021
C COMMON /TSAVE/ XSAVE(1000), YSAVE(1000), ZSAVE(1000), TSAVE(1000)

ISN 0022
C COMMON /HLOAD/ HLD(21), HLD(21), HLD(21), HLD(21)

ISN 0023
C COMMON /XLOAD/ XLD(21), XLD(21), XLD(21), XLD(21)

ISN 0024
C COMMON /YLOAD/ YLD(21), YLD(21), YLD(21), YLD(21)

ISN 0025
C COMMON /ZLOAD/ ZLD(21), ZLD(21), ZLD(21), ZLD(21)

ISN 0026
C COMMON /DELTAZ/ DELTAZ(21), DELTAZ(21), DELTAZ(21), DELTAZ(21)

ISN 0027
C COMMON /SLINE/ SLINE(21), SLINE(21), SLINE(21), SLINE(21)

ISN 0028
C COMMON /TLINE/ TLINE(21), TLINE(21), TLINE(21), TLINE(21)

ISN 0029
C COMMON /NLIN/ NLIN(21), NLIN(21), NLIN(21), NLIN(21)

ISN 0030
C COMMON /NYLIN/ NYLIN(21), NYLIN(21), NYLIN(21), NYLIN(21)

ISN 0031
C COMMON /TLOAD/ TLD(21), TLD(21), TLD(21), TLD(21)

ISN 0032
C COMMON /XLOAD/ XLD(21), XLD(21), XLD(21), XLD(21)

ISN 0033
C COMMON /YLOAD/ YLD(21), YLD(21), YLD(21), YLD(21)

ISN 0034
C COMMON /ZLOAD/ ZLD(21), ZLD(21), ZLD(21), ZLD(21)

ISN 0035
C COMMON /DELTAZ/ DELTAZ(21), DELTAZ(21), DELTAZ(21), DELTAZ(21)

ISN 0036
C COMMON /SLINE/ SLINE(21), SLINE(21), SLINE(21), SLINE(21)

ISN 0037
C COMMON /TLINE/ TLINE(21), TLINE(21), TLINE(21), TLINE(21)

ISN 0038
C COMMON /NLIN/ NLIN(21), NLIN(21), NLIN(21), NLIN(21)

ISN 0039
C COMMON /NYLIN/ NYLIN(21), NYLIN(21), NYLIN(21), NYLIN(21)

ISN 0040
C COMMON /TLOAD/ TLD(21), TLD(21), TLD(21), TLD(21)

ISN 0041
C COMMON /XLOAD/ XLD(21), XLD(21), XLD(21), XLD(21)

ISN 0042
C COMMON /YLOAD/ YLD(21), YLD(21), YLD(21), YLD(21)

ISN 0043
C COMMON /ZLOAD/ ZLD(21), ZLD(21), ZLD(21), ZLD(21)

ISN 0044
C COMMON /DELTAZ/ DELTAZ(21), DELTAZ(21), DELTAZ(21), DELTAZ(21)

ISN 0045
C COMMON /SLINE/ SLINE(21), SLINE(21), SLINE(21), SLINE(21)

ISN 0046
C COMMON /TLINE/ TLINE(21), TLINE(21), TLINE(21), TLINE(21)

ISN 0047
C COMMON /NLIN/ NLIN(21), NLIN(21), NLIN(21), NLIN(21)

ISN 0048
C COMMON /NYLIN/ NYLIN(21), NYLIN(21), NYLIN(21), NYLIN(21)
!VERSION 1.3.0 (01 MAY 80)   STAEBL SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)   DATE 08.14/10.58.59   PAGE 2

ISH  0035  NPS = NP(I)-1   00049
ISH  0036  NH1(I) = 0.0   00050
ISH  0037  NH2(NPS+1,I) = 0.0   00051
ISH  0038  DO 30 N = 2,NPS   00052
ISH  0039  THT1 = ATAN2(YH1(I+1) - YH1(I),XH1(I+1) - XH1(I))   00053
ISH  0040  THT2 = ATAN2(YH1(I) - YH1(I),XH1(I) - XH1(I))   00054
ISH  0041  THT = (THT1 + THT2) / 2.0   00055
ISH  0042  TILN(H,I) = TIL(T1N(I) + COS(THT))   00056
ISH  0043  30 CONTINUE   00057
ISH  0044  40 CONTINUE   00058
ISH  0045  NSTNB = NSTA - NER   00059
ISH  0046  DO 45 I = NER,NSTA,NSTNB   00060
ISH  0047  NPS = NPS(I)   00061
C WRITE (6,900) (YH1(I),I=1,NPS)   00062
C WRITE (6,900) (XH1(I),I=1,NPS)   00063
C WRITE (6,900) (TILN(I),I=1,NPS)   00064
C WRITE (6,900) (TILN(I),I=1,NPS)   00065
ISH  0048  45 CONTINUE   00066
C C SAVE MEAN Y VALUES AND X VALUES IN AN ARRAY AND INTERPOLATE   00067
C FOR EQUAL INCREMENT X VALVES . REPEAT FOR THICKNESSES   00068
C
ISH  0049  DO 110 I = NBR,NSTA   00069
ISH  0050  NP1 = 15   00070
ISH  0051  NPS = NPS(I)   00071
C C FILL IN DUMMY ARRAYS XM, YM, TIM FROM XI, YI, TIL VECTORS   00072
C
ISH  0052  DO 201 K = 1,NPS   00073
ISH  0053  XM(K) = TIL(K,I)   00074
ISH  0054  XM(K) = XM(K,I)   00075
ISH  0055  201   00076
ISH  0056  CALL PBFIT FOR CURVE FIT   00077
ISH  0057  CALL PBFIT(XM, YM, 'I', 1, NPS, A, B, C, D, A1, B1, C1, D1, S1)   00078
ISH  0058  CALL PBFIT(XM, YM, 'I', 1, NPS, A2, B2, C2, D2, A3, B3, C3, D3, S2)   00079
ISH  0059  EQUAL BREAKUP ARC-LENGTH   00080
ISH  0060  SARC = SI(NPS) / NPS   00081
C C NOW SEARCH SI ARRAY FOR INTERVAL VALUE   00082
C
ISH  0061  NODE = 1   00083
ISH  0062  ARC1 = SARC / 2.   00084
ISH  0063  N = 1   00085
ISH  0064  IF (ARC1 .LE. SI(N)) GO TO 220   00086
ISH  0065  N = N + 1   00087
ISH  0066  GO TO 210   00088
ISH  0067  210   00089
ISH  0068  220 DIST = (ARC1 - SI(N-1)) / (SI(N) - SI(N-1))   00090
ISH  0069  SONE = (SI(N) - SI(N-1)) * DIST   00091
ISH  0070  STNO = (SI(N) - SI(N-1)) * DIST   00092
ISH  0071  CALL CUBIC(A(N-1), B(N-1), C(N-1), D(N-1), SONE, XXIF(NODE))   00093
ISH  0072  CALL CUBIC(A(N-1), B(N-1), C(N-1), D(N-1), SONE, YYIF(NODE))   00094
ISH  0073  CALL CUBIC(A(N-1), B(N-1), C(N-1), D(N-1), STNO, TTHLF(NODE))   00095
FILL IN THE XMF, YMF, TMLF VECTORS WITH THE INTERPOLATED VALUES

DO 230 K = 1, NPF
    XMF(K, I) = XMF(K)
    YMF(K, I) = YMF(K)
C WRITE(6, 902) I, K, XMF(K, I), YMF(K, I)
FORMAT( 5X, I, K, XMF, YLF, 2E12.5)
TMLF(K, I) = TMLF(K)
C

CONTINUE

TRANSLATE COORDINATES TO AN ENGINE AXIS SYSTEM

DO 130 I = NBR, NSTA
    XSCLE = XCHORD
    XSCCG(I) = XSCLE - XSCII
    YSCCG(I) = YSCII - YBARYII)
C

CONTINUE

SHIFT XY PLANE TO ENGINE AXIS YZ

DO 150 I = NBR, NSTA
    R(I) = R(I)
    XSGG(I) = XSGG(I)
C

CONTINUE

ROTA T H/TH RU ALPHA CHORD - 90.0

DO 170 I = NBR, NSTA
    ALPHA(I) = 90.0 - .0174533
C

CONTINUE

WRITE(6,900) ANG, EN, EM, XSCR(I), YSCR(I), ZSCR(I), AREA(I)
+VERSION 1.3.0  (01 MAY 80)  STAEBL  SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 02.14/10.56.59  PAGE 4

ISH 0110  Y(N,I) = YG(N,I) + EM + EN + ZG(N,I)  00167
ISH 0112  Z(N,I) = EM + ZG(N,I) - EN + YG(N,I)  00168
ISH 0113  160 CONTINUE  00169
ISH 0114  170 CONTINUE  00170
C  DO 175 I = NCR,NSTA,NSTNO
C  WRITE (6,900) (XCG(N,I),N=1,NPF)  00171
C  WRITE (6,900) (YCG(N,I),N=1,NPF)  00172
C  WRITE (6,900) (ZCG(N,I),N=1,NPF)  00173
C  WRITE (6,900) (YG(N,I),N=1,NPF)  00174
C  WRITE (6,900) (ZG(N,I),N=1,NPF)  00175
C  WRITE (6,900) (THL(N,I),N=1,NPF)  00176
C  WRITE (6,900) (TH2(N,I),N=1,NPF)  00177
C  WRITE (6,900) (TH4(N,I),N=1,NPF)  00178
C  WRITE (6,900) (TH6(N,I),N=1,NPF)  00179
C  WRITE (6,900) (TH8(N,I),N=1,NPF)  00180
C  175 CONTINUE  00181
ISH 0115  IJ = 1  00182
ISH 0116  IF(IJ .EQ. 1) GO TO 811  00183
C  TO3902 GENERATED NODES AND THICKNESSES  00184
C  DO 800 I = NCR,NSTA  00185
ISH 0118  DO 801 J = 1,NPF  00186
ISH 0119  READ(5,802) Y(J,I),Z(J,I)  00187
ISH 0120  802 FORMAT(32X,2F3.0)  00188
C  DO 805 J = 1,NPF  00189
ISH 0122  READ(5,803) P1,P2  00190
ISH 0123  803 FORMAT(24X,F8.0,24X,F8.0)  00191
ISH 0124  805 THL(J,I) = (P1 + P2) / 2.  00192
C  WRITE(6,810) I  00193
C  FORMAT(5X,'TO39 NODES AND THICKNESSES FOR SECTION ',I5)  00194
C  WRITE(6,900) (X(N,I),N=1,NPF)  00195
C  WRITE(6,900) (Y(N,I),N=1,NPF)  00196
C  WRITE(6,900) (Z(N,I),N=1,NPF)  00197
C  WRITE(6,900) (THL(N,I),N=1,NPF)  00198
C  WRITE(6,900) (TH2(N,I),N=1,NPF)  00199
C  WRITE(6,900) (TH4(N,I),N=1,NPF)  00200
C  WRITE(6,900) (TH6(N,I),N=1,NPF)  00201
C  WRITE(6,900) (TH8(N,I),N=1,NPF)  00202
C  800 CONTINUE  00203
ISH 0126  811 CONTINUE  00204
C  ISH 0128  J = 0  00205
ISH 0129  K = 0  00206
ISH 0130  DO 190 I = NBR,NSTA  00207
ISH 0131  K = K + 1  00208
ISH 0132  XSCSV(K) = XSCR(I)  00209
ISH 0133  YSCSV(K) = YSCR(I)  00210
ISH 0134  ZSCSV(K) = ZSCR(I)  00211
ISH 0135  ASAVE(K) = ALPHA(I)  00212
ISH 0136  POLAR(K) = XIMIN(I) + XIMAX(I)  00213
ISH 0137  ASAVE(K) = AREA(I)  00214
ISH 0138  DO 160 N = 1,NPF  00215
ISH 0139  J = J +1  00216
ISH 0140  XSAVE(J) = X(N,I)  00217
ISH 0141  YSAVE(J) = Y(N,I)  00218
ISH 0142  ZSAVE(J) = Z(N,I)  00219
ISH 0143  TSAVE(J) = THL(N,I)  00220
C  WRITE (7,901) XSAVE(J) , YSAVE(J) , ZSAVE(J) , TSAVE(J)  00221
ISH 0144  180 CONTINUE  00222
ISH 0145  190 CONTINUE  00223
*VERSION 1.3.0 (01 MAY 80) STAEBL SYSTEM/370 FORTRAN H EXTENDED (ENHANCED)  DATE 82.141/10.50.59  PAGE 5

ISH 0146  K = 0  00226
ISH 0147  DO 200  I = NSR,NST A  00227
ISH 0148  K = K + 1  00228

C    WRITE (7,901) XCSV(K), YSCSV(K), ZSCSV(K), ALSAVE(K)
C    1, P L A R I(K), ASAVE(K)

ISH 0149  200 CONTINUE  00229
ISH 0150  NTMR = NSTA - NSR + 1  00230
ISH 0151  IF (NCD .EQ. 1)  00231
ISH 0152     IC A L L HOLLOW (DLED, DTE D, DROOD, DTIPD,  00232
ISH 0153     2 TTID, T T LD, NPF, NMT R)  00233
ISH 0153     IF (NCD .EQ. 2)  00234
ISH 0154     IC A L L L A H I I N (TSKIN, TCEHTR, PB T, PGE, NPF, NMT R)  00235
ISH 0155  901 FORMAT (6E12.5)  00236
ISH 0156  900 FORMAT (1X,9F8.5)  00237
ISH 0157  RETURN  00238
ISH 0158  END  00239

*OPTIONS IN EFFECT: NAME(MAIN) OPTIMIZE(3) LINECOUNT(60) SIZE(MAX) AUTO DBL(DBL4)
*OPTIONS IN EFFECT: SOURCE EBCDIC NOLIST NODECK OBJECT NCHAP NOFORMAT GOSTHT NO:REF NO:ALC NO:NSF TERM IBM FLAG(I)
*STATISTICS* SOURCE STATEMENTS = 157, PROGRAM SIZE = 141976, SUBPROGRAM NAME = STAEBL
*STATISTICS* NO DIAGNOSTICS GENERATED
**** END OF COMPI LATION *******

2972K BYTES OF CORE NOT USED
SUBROUTINE INPUT (NR, NC)  

MATERIAL PROPERTIES AND GEOMETRY INPUT  

COMMON /ZCOEF/ E11(7), E22(7), E33(7), G12(7), G23(7), G13(7),  
1 
V11(7), V13(7), V23(7)  

COMMON /PLY/ PLY(21, 25, 7), THETA(7), RHO(7)  

COMMON /FAIL/ X1T(7), X1C(7), X2T(7), X2C(7), S6P(7), S6M(7), TSAI(6, 25)  

WRITE(8, 1)  

FORMAT(SX, 'IS THIS AN ISOTROPIC BLADE, 1= YES')  

READ(8, *) ISO  

IF (ISO .NE. 1) GO TO 2  

ISOTROPIC BLADE  

WRITE(8, 3)  

FORMAT(SX, 'INPUT-E(PSI), V AND DENSITY(LB/IN3), FREE FORMAT')  

READ(8, *) E, V, R  

E = 16100000.  
V = .33  
R = .16  

DPLY = 1. / 7.  

YIELD = 110000.  
SHEAR = 0.577 * YIELD  

TI LAYERS  

DO 100 II = 1, 3  

X1T(II) = YIELD  
X1C(II) = YIELD  
X2T(II) = YIELD  
X2C(II) = YIELD  
S6P(II) = SHEAR  
S6M(II) = SHEAR  
E11(II) = E  
E22(II) = E  
E33(II) = E  
G12(II) = E / 2. / (1. + V)  
G13(II) = G12(II)  
G23(II) = G12(II)  
V12(II) = V  
V13(II) = V  
V23(II) = V  
RHO(II) = R / 366.4  

THETA(II) = 0.  

I = I + 3  

B/A AND G/E LAYERS  

I2 = 2
C E1BA = 31.0E6
C E1BA = 0.9 * E1BA
C E2BA = 20.0E6
C E2BA = 0.9 * E2BA
C V1CBA = 0.27
C G3A = 0.5E6
C GBA = 0.9 * GBA
C RBA = 0.097
C RSA = 0.9 * RSA
C X1TBA = 140000.0
C X1TBA = 0.9 * X1TBA
C X1CBA = 200000.0
C XICBA = 0.9 * XICBA
C X2TBA = 14800.0
C X2TBA = 0.9 * X2TBA
C X2CBA = 32200.0
C X2CBA = 0.9 * X2CBA
C SBA = 14500.0
C SBA = 0.9 * SBA
C E1GE = 18.5E6
C E2GE = 1.5(IE6
C V12GE = 0.3
C GGE = 0.85E6
C RGE = 0.056
C X1TGE = 160000.0
C X1CGE = 160000.0
C X2TGE = 7500.0
C X2CGE = 25000.0
C SGE = 10000.0
C DO 200 II = 1,2
C E11(I2) = E1BA
C E22(I2) = E2BA
C E33(I2) = E22(I2)
C G12(I2) = GBA
C G13(I2) = GBA
C G23(I2) = GBA
C V12(I2) = V12BA
C V13(I2) = V12BA
C V23(I2) = V12BA
C R12(I2) = RBA / 386.4
C X1T(I2) = X1TBA
C X1C(I2) = XICBA
C X2T(I2) = X2TBA
C X2C(I2) = X2CBA
C S6P(I2) = SBA
C S6N(I2) = SBA
C E11(I3) = E1GE
C E22(I3) = E2GE
C E33(I3) = E22(I3)
C G12(I3) = GGE
C G13(I3) = GGE
C G23(I3) = GGE
C V12(I3) = V12GE
C V13(I3) = V12GE
ISH 0099    RETURN
ISH 0100    END

*OPTIONS IN EFFECT=NAME(MAIN) OPTIMIZE(3) LINEDCOUNT(60) SIZE(MAX) AUTODELL(DBLA4)
*OPTIONS IN EFFECT=SOURCE EBCDIC NOLIST NCODEX OMIT NOFORMAT GOSTHT NOKREF NOKTAL NOASTEL TERM IBM (1)
*STATISTICS* SOURCE STATEMENTS = 99, PROGRAM SIZE = 1402, SUBPROGRAM NAME = INPUT
*STATISTICS* NO DIAGNOSTICS GENERATED
****** END OF COMPIILATION ******

3004K BYTES OF CORE NOT USED
REFERENCES


This Page Intentionally Left Blank
DISTRIBUTION LIST FOR INTERIM REPORT  
NASA CR-167949  

STRUCTURAL TAILORING OF ENGINE BLADES (STAEBL)  
CONTRACT NAS 3-22525

NASA Lewis Research Center  
21000 Brookpark Road  
Cleveland, OH 44135

Attn: Contracting Officer  
Technical Report Control Office  
Technology Utilization Office  
AFSC Liaison Office  
S&MT Division Contract File  
Library  
L. Berke  
R. H. Johns  
L. J. Kiraly  
P. C. Chamis  
M. S. Hirschbein  
J. A. Ziemianski  
D. H. Buckley  
E. V. Zaretsky  
D. P. Fleming  
R. E. Kielb  
G. V. Brown  
J. J. Adamczyk  
D. L. Nored

Mail Stop Copies
500-313 1  
5-5 1  
7-3 1  
501-3 1  
49-6 2  
60-3 1  
49-6 1  
49-6 1  
23-2 1  
49-6 4  
49-6 1  
49-6 1  
23-2 1  
6-1 1  
6-1 1  
49-6 1  
49-6 1  
5-9 1  
301-2 1

National Aeronautics & Space Administration  
Washington, DC 20546  

Attn: NHS-22/Library  
RTM-6/S. L. Venneri  
RTM-6/D. J. Weidman

NASA Ames Research Center  
Moffett Field, CA 94035  

Attn: Library  
202-3 1

NASA Goddard Space Flight Center  
Greenbelt, MD 20771  

Attn: 252/Library  
1

NASA John F. Kennedy Space Center  
Kennedy Space Center, FL 32931  

Attn: Library  
AD-CSO-1 1
NASA Langley Research Center
Hampton, VA 23665
Attn: Library 185 1
M. F. Card 244 1

NASA Lyndon B. Johnson Space Center
Houston, TX 77001
Attn: JM6/Library 1

NASA George C. Marshall Space Flight Center
Marshall Space Flight Center, AL 35812
Attn: AS61/Library 1

Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91103
Attn: Library 1
B. Wada 1
R. Levi 1

NASA S&T Information Facility
P. O. Box 8757
Baltimore-Washington Int. Airport, MD 21240
Attn: Acquisition Department 10

Air Force Aeronautical Propulsion Laboratory
Wright-Patterson AFB, OH 45433
Attn: Z. Gershon 1
E. Bailey 1

Air Force Systems Command
Aeronautical Systems Division
Wright-Patterson AFB, OH 45433
Attn: Library 1
C. W. Cowie 1
J. McBane 1

Aerospace Corporation
2400 E. El Segundo Blvd.
Los Angeles, CA 90045
Attn: Library-Documents 1

Air Force Office of Scientific Research
Washington, DC 20333
Attn: A. K. Amos 1

Department of the Army
U. S. Army Material Command
Washington, DC 20315
Attn: AMCRD-RC 1

U. S. Army Ballistics Research Laboratory
Aberdeen Proving Ground, MD 21005
Attn: Dr. Donald F. Haskell DRXBR-BM 1
General Motors Corporation
Warren, MI 48090
Attn: R. J. Trippet

AVCO Lycoming Division
550 South Main Street
Stratford, CT 06497
Attn: Mr. Herbert Kaehler

Beech Aircraft Corporation, Plant 1
Wichita, KA 67201
Attn: Mr. M. K. O'Connor

Bell Aerospace
P. O. Box 1
Buffalo, NY 14240
Attn: R. A. Gellatly

Boeing Aerospace Company
Impact Mechanics Lab
P. O. Box 3999
Seattle, WA 98124
Attn: Dr. R. J. Bristow

Boeing Commercial Airplane Company
P. O. Box 3707
Seattle, WA 98124
Attn: Dr. Ralph B. McCormick

Boeing Commercial Airplane Company
P. O. Box 3707
Seattle, WA 98124
Attn: Mr. David T. Powell

Boeing Commercial Airplane Company
P. O. Box 3707
Seattle, WA 98124
Attn: Mr. David T. Powell

Boeing Company
Wichita, KA 67201
Attn: Mr. C. F. Tiffany

McDonnell Douglas Aircraft Corporation
P. O. Box 516
Lambert Field, MO 63166
Attn: Library

Douglas Aircraft Company
3855 Lakewood Blvd.
Long Beach, CA 90846
Attn: Mr. M. A. O'Connor, Jr.
Garrett AiResearch Manufacturing Co.
111 S. 34th Street
P. O. Box 5217
Phoenix, AZ 85010
Attn: L. A. Matsch

Mr. R. Stockton
Garrett Turbine Engine Company
Rotor Integrity, 503-42
Mechanical Component Design
111 S 34th Street
P. O. Box 5217
Phoenix, AZ 85010

General Dynamics
P. O. Box 748
Fort Worth, TX 76101
Attn: Library

General Dynamics/Convair Aerospace
P. O. Box 1128
San Diego, CA 92112
Attn: Library

General Electric Company
Interstate 75, Bldg. 500
Cincinnati, OH 45215
Attn: Dr. L. Beitch
Dr. M. Roberts
Dr. V. Gallardo

General Electric Company
Aircraft Engine Group
Lynn, MA 01902
Attn: Mr. Herbert Garten

Grumman Aircraft Engineering Corp.
Bethpage, Long Island, NY 11714
Attn: Library

IIT Research Institute
Technology Center
Chicago, IL 60616
Attn: Library

Lockheed California Company
P. O. Box 551
Dept. 73-31, Bldg.90, PL. A-1
Burbank, CA 91520
Attn: Mr. D. T. Pland
Lockheed California Company
P. O. Box 551
Dept. 75-71, Bldg. 63, PL. A-1
Burbank, CA 91520
Attn: Mr. Jack E. Wignot

Northrop Space Laboratories
3401 West Broadway
Hawthorne, CA 90250
Attn: Library

North American Rockwell, Inc.
Rocketdyne Division
6633 Canoga Avenue
Canoga Park, CA 91304
Attn: Library, Dept. 596-306

North American Rockwell, Inc.
Space & Information Systems Div.
12214 Lakewood Blvd.
Downey, CA 90241
Attn: Library

Norton Company
Industrial Ceramics Div.
Armore & Spectramic Products
Worcester, MA 01606
Attn: Mr. George E. Buron

Norton Company
1 New Bond Street
Industrial Ceramics Division
Worcester, MA 01606
Attn: Mr. Paul B. Gardner

United Aircraft Corporation
Pratt & Whitney Group
Government Products Division
P. O. Box B2691
West Palm Beach, FL 33402
Attn: Library
R. A. Marmol

United Aircraft Corporation
Pratt & Whitney Aircraft Group
400 Main Street
East Hartford, CT 06108
Attn: Library
R. Liss
D. H. Hibner

United Aircraft Corporation
Hamilton Standard Division
Windsor Locks, CT 06096
Attn: Dr. G. P. Townsend
Dr. R. A. Cornell

347
End of Document