

USER'S MANUAL FOR THE LAMINATED COMPOSITE
INELASTIC SOLVER COMPUTER PROGRAM

David D. Robertson, Maj, USAF

AFIT/ENY/TR96-01

Approved for public release, distribution unlimited

19970502 240

DTIC QUALITY INSPECTED 4

User's Manual
for the
Laminated composite Inelastic SOLver
(LISOL)
Computer Program

David D. Robertson, Maj, USAF

July 1996

Department of Aeronautics and Astronautics
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433

Approved for public release, distribution unlimited

Table of Contents

<i>Introduction.....</i>	<i>3</i>
<i>Micromechanics Background</i>	<i>3</i>
<i>Running the Program.....</i>	<i>6</i>
<i>Material Properties.....</i>	<i>7</i>
<i>Format for LISOL Material Properties File</i>	<i>9</i>
<i>Load Sequence</i>	<i>10</i>
<i>Format for LISOL Load File.....</i>	<i>11</i>
<i>Convergence Parameters</i>	<i>12</i>
<i>Memory Considerations</i>	<i>14</i>
<i>Summary.....</i>	<i>14</i>
<i>References</i>	<i>15</i>
<i>Appendix - A, Material Property Files</i>	<i>17</i>
<i>Appendix - B Program Listing</i>	<i>22</i>

Introduction

LISOL is a computer program developed at the Air Force Institute of Technology to model the nonlinear behavior of metal matrix composite laminates[1]. It uses a micromechanics approach to develop a set of constitutive relations which are automatically assembled for a specified layup. It uses the unified viscoplastic theory of Bodner and Partom to model the matrix material and assumes the fiber is thermoelastic. Temperature dependent material properties may be input for both the fiber and matrix. An interfacial failure scheme based on a statistical approach is employed to model the progressive failure of the fiber/matrix interface. Property files for the SCS6/Ti-15-3 and SCS6/Ti- β 21s (sometimes referred to as TIMETAL21s) systems are currently available.

The program was designed with thermomechanical fatigue (TMF) cyclic loading in mind and to be run interactively. Therefore, it is menu driven so that the user can modify the load sequence and output from within the program. Also, symmetric layups are assumed which enhances the speed of the program, so only the ply angles required to characterize the composite need be entered and not the entire layup.

Micromechanics Background

The micromechanics approach used in the LISOL program is a modified method of cells approach [2-3] where the representative volume element for a single ply is modeled by a unit cell consisting of six regions each of uniform stress. A single fiber region, three matrix regions, and two infinitely thin interface regions, as shown in Figure 1, are

employed where the fiber and matrix are assumed to possess equivalent normal strain in the fiber direction. Also, the average strain along the external faces of the analysis cell are set equal to the ply strain. Equilibrium is accomplished by ensuring that the average stress through any cross section of the unit cell is in equilibrium with the ply stress and by forcing the stresses normal to the faces between adjacent regions to be equal. Slip between adjacent regions is allowed only in the 2-3 plane where continuity of displacements is maintained only along the external faces of the unit cell. The resulting equations for the unit cell are essentially the same as the method of cells except for the axial shear response. The original method of cells approach as proposed by Aboudi ensures equilibrium is satisfied between adjacent regions while relaxing the requirement for continuity of displacement by satisfying it only in an average sense [4]. The equations used in the LISOL program satisfy continuity of displacement in axial shear while satisfying equilibrium only in an average sense (similar to the displacement-based finite element method). All other relations are identical to the method-of-cells.

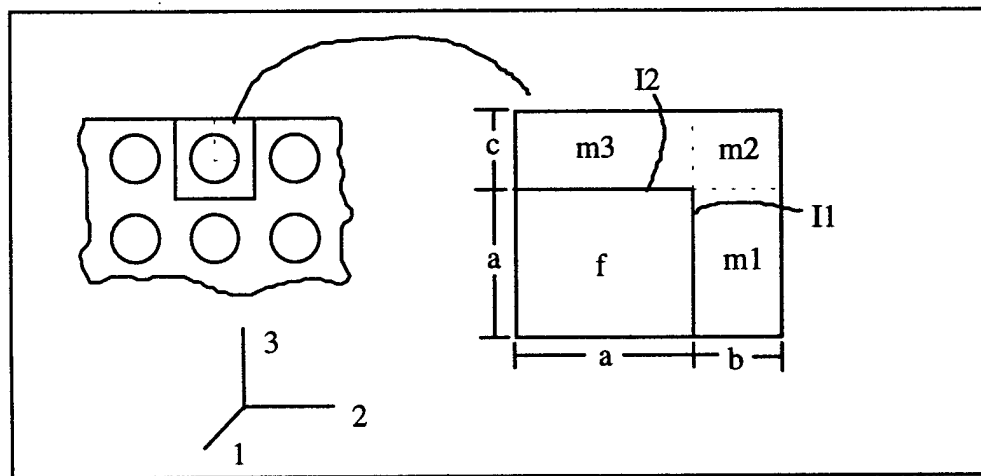


Figure 1. Schematic of the Micromechanics Model for a Single Ply

The equations for a single ply may be simplified down so that only a few normal stresses of the various regions remain coupled through a matrix equation. All the remaining normal and shear stresses may be determined by one-line equations. The general form of the equations are as follows:

$$\begin{bmatrix} P \end{bmatrix} \begin{Bmatrix} \vdots \\ \sigma_{reg} \\ \vdots \end{Bmatrix} = \begin{Bmatrix} f_{\Delta T} \\ \bar{\sigma} \end{Bmatrix} - \begin{bmatrix} P_I \end{bmatrix} \begin{Bmatrix} \vdots \\ \epsilon_{reg}^I \\ \vdots \end{Bmatrix} \quad (1)$$

where σ_{reg} represents the region stresses for each ply, $f_{\Delta T}$ is the thermal component of load, $\bar{\sigma}$ is the applied stress on the ply, ϵ_{reg}^I represents the region plastic strains, and the matrices, P and P_I , simply relate the various quantities according to the given assumptions. All shear stresses as well as the remaining normal stresses are obtained from single-line equations[5].

The laminate analysis is accomplished by using the classical laminated plate theory assumption which states that any plane perpendicular to the midplane before deformation remains both plane and perpendicular to the midplane after deformation. The micromechanics equations for each ply, Eq (1), are assembled according to this assumption which results in the following general form for the laminate analysis:

$$\begin{bmatrix} P^{com} \end{bmatrix} \begin{Bmatrix} \epsilon_o \\ \kappa \\ \sigma_{reg} \\ \vdots \end{Bmatrix} = \begin{Bmatrix} \vdots \\ f_{\Delta T} \\ \vdots \end{Bmatrix} + \begin{bmatrix} P_N^{com} \end{bmatrix} \begin{Bmatrix} N \\ M \end{Bmatrix} - \begin{bmatrix} P_p^{com} \end{bmatrix} \begin{Bmatrix} \vdots \\ \epsilon_{reg}^p \\ \vdots \end{Bmatrix} \quad (2)$$

where ϵ and κ are the midplane strain and curvature, respectively, σ_{reg} represents the region stresses for each ply, $f_{\Delta T}$ is the thermal component, N and M are the applied forces and moments on the laminate, ϵ^p_{reg} represents the region plastic strains for each ply, and the matrices P^{com} , P_N^{com} and P_p^{com} are associated with the given composite layup and relate the various quantities according to the given assumptions [6].

Running the Program

The only file that need exist prior to running the program is the material property file. Once that exists, simply execute the program. The main menu will immediately come up. The desired layup is input by selecting the first submenu (*Type of Model*). The function of all other submenus are self-explanatory. The property file must be input through selecting the *Constituent Properties* submenu, and the desired load sequence may be entered either interactively or through an existing file through the *Loading* submenu. Once a solution is obtained by selecting the *Solve for Response to Applied Loading* submenu, output may be examined by selecting *Examine Results*.

Most output will be generated by selecting *Create Tables* from within the *Examine Results* submenu. The user then specifies the number of columns and what quantities are desired in those columns for output to either the screen or a file. The entire calculation may be saved and reloaded at a later date by selecting *Write existing data to a file*. The unformatted file created in this manner is a duplicate of the scratch file the program has created to this point. Reloading this file allows the user to examine results at a later date.

Material Properties

The material properties file in LISOL carries around some excess baggage due to a previous version of the program. For instance, the program flags on the second line are from the unidirectional/elastic-plastic version which possessed a more interactive capability for changing the material properties. Also, the elastic-plastic material properties (type of hardening, strain hardening parameter, and yield stress) are throwbacks from this earlier version. The elastic-plastic material model has not been carried over to the laminate analysis.

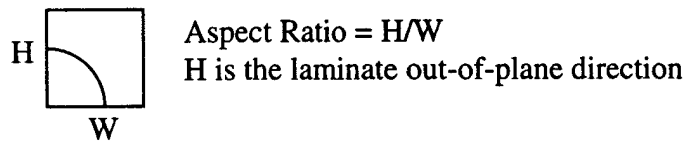
Nonlinear material models currently available in LISOL are the Bodner-Partom model with directional hardening and the original isotropic hardening Bodner-Partom model. To switch from one material model to another simply change what subroutine (BODDIR, or BODISO) is called in the subroutine VISC as well as switching the appropriate material property commented lines in VISC. The Bodner-Partom model with back stress has also been used with the program in the past. As a result, some of the values in the properties file are unique to this model. Also, if the isotropic hardening B-P model is desired, then the associated material properties should be entered using the text fields in the properties file corresponding to the B-P directional hardening model.

The LISOL program employs an interfacial failure scheme that provides for progressive failure of the interface during loading[7-8]. Three parameters are required to characterize the interface normal from its face and two are required in the tangential direction. The following paragraph gives a brief tabular description of these parameters. For a complete description of the interfacial model see the references.

Two properties files are included with the program. They are for the metal matrix composites systems SCS6/Ti-15-3 and SCS6/TIMETAL®21S.

Explanations of properties unique to LISOL:

1. Aspect Ratio. Ratio of height to width of the representative volume element.



2. σ_{IC} . Intercept on stress axis the interfacial stress-displacement curve returns to during unloading. Only applies for the normal direction from the interface.
3. S_{Ifn} . Initial interfacial failure stress in the normal direction.
4. u_{Ifn} . Relative displacement of interface at which point the interface has completely failed.
5. S_{IfT} , u_{IfT} . Same as above only for the tangential direction from the interface.

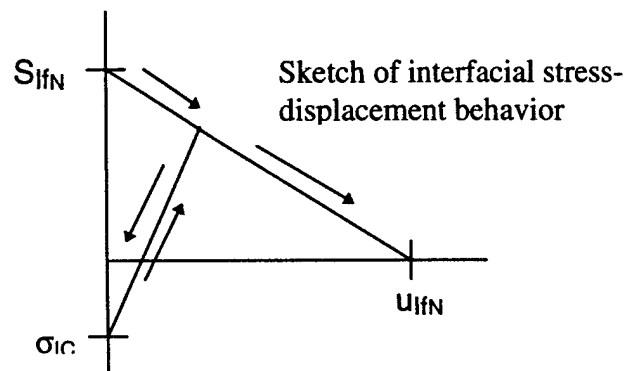


Figure 2. Interfacial Failure Model Used in LISOL

Format for LISOL Material Properties File

Note: All numerical values are in list directed read format

Name of Composite (40 characters or less) ← 1st Line

Y Y N Y (program flags - always the same, 4A2 format)

(blank line)

of fiber points, aspect ratio, fiber volume fraction

Temp (C), E_{f11} (GPa), ν_{f12} , E_{f22} , ν_{f23} , G_{f12}

α_{f1} ($^{\circ}\text{C}^{-1}$), α_{f2}

(blank line)

Repeat for all fiber points

(blank line)

of matrix points, type of hardening, D_0 , T_{REF}

Temp (C), E_{m11} (GPa), ν_{m12} , E_{m22} , ν_{m23} , G_{m12}

α_{m1} ($^{\circ}\text{C}^{-1}$), α_{m2} , strn hrdng param, yield str

O_{SAT} , $f1$, $f3$, Z_0 , n (B-P back stress params)

Z_0 , n , m_1 , Z_1 , r_1 , A_1

m_2 , Z_2 , r_2 , A_2 , Z_3 (B-P dir. hrdng params)

(blank line)

Repeat for all matrix points

(blank line)

of interface points, σ_{IC}

Temp (C), S_{Ifn} (MPa), u_{Ifn} (normalized to fib dimension)

S_{IfT} , u_{IfT}

(blank line)

Repeat for all
interface points

Load Sequence

The load file in LISOL is designed for a stress-based load sequence. Unlike the material properties file, the load file may be created and modified within LISOL. It is mostly self explanatory. The load is applied in LISOL through a preload, cyclic, and post-load. To do this, two load points within the load file must be specified as defining the repeating cycle. Two distinct points that define the repeating cycle must always be given; even if a monotonic load is desired (simply specify a cycle number of 1).

The first load point is assumed to be the stress-free processing temperature of the composite. This point simply sets the zero stress state. Also, it is important to remember that output is only saved at the load points entered in the load file (not at any subincrements). Therefore, if output is desired at a specific point, that point must be present in the load file.

Also, the load is specified per unit length (edge load). This edge load is applied to the total thickness (sum of ply thicknesses) the user has specified in the MODEL submenu. Please note that this is applied directly to the total thickness listed in the submenu (*i.e.* you do not need to double it since it is a symmetric layup). Remember, the LISOL program assumes a symmetric layup, so the ply thicknesses entered for the model should correspond to the sum of all thicknesses for a given orientation angle.

Definition of terms:

1. Δt . Change in time from the previous load point. Any positive nonzero number may be specified for the first cycle.
2. # of subincrements . Number of increments to subdivide the load step.
3. N_x , N_y , N_{xy} . Plate edge forces per unit length.

Format for LISOL Load File

Program flags

First line: 2A2, 4I7 format
subsequent lines: list directed

N Y # of points in file point to start cycle point to end cycle # of cycles
 Δt (sec), Temp(C), N_x (MPa-mm), N_y , N_{xy} , # of subincrements

⋮

Convergence Parameters

LISOL is a stress-based program which means it requires a stress input and solves for the strain. The associated numerical algorithm which solves the nonlinear equations contains a few convergence parameters which may be modified for a specific problem. A brief explanation of the convergence parameters is given here, but for a detailed discussion of the numerical algorithm see reference 4.

Four convergence parameters are used in the program and can individually be modified before the solution sequence. They are:

1. Multiple of elastic rates before incrementing, TOLV1. This was an effort to automate the program so it would automatically increase the number of subincrements at a given load if needed. The ratio of the calculated effective plastic strain rate to the effective elastic strain rate is checked before each subincrement. This represents a measure of the level of nonlinearity for the given point. If this ratio is above TOLV1, then the number of remaining subincrements for that load point is doubled. If the solution error message "maximum subincrements exceeded" appears, then this automation feature is causing a problem. It may be shut-off by increasing TOLV1 arbitrarily high. If the error message still appears, then the solution is diverging.
2. Instability convergence factor, TOLV2. This factor controls the rate of convergence. A higher number produces a slower but more stable convergence. From experience the default value of 15 is satisfactory for almost all cases.

3. Convergence error tolerance, TOLV3. This value controls when to stop iterating and apply the next load increment. If the change in the calculated effective plastic strain rate from the present to the previous iteration divided by the effective elastic strain rate is less than TOLV3, then the iteration ceases.

4. Amount above interface failure before incrementing, TOLV4. This performs a similar function for interfacial failure as TOLV1 does for viscoplasticity. If the calculated interfacial stress is more than this amount above the interfacial failure stress, then the remaining number of subincrements for that load point is doubled.

The stress-based numerical solution is advantageous for many cyclic type loadings because most cyclic tests are stress-controlled. However, some convergence problems can occur with stress-based solutions. For instance, the stress strain curve not only flattens out, but also begins to decrease for many materials. A stress-based program cannot converge in such cases. A way to get around this for monotonic loading is to alternatively increase and decrease the load steps to approximate the stress strain curve as shown below.

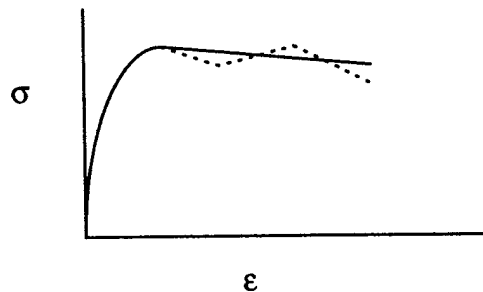


Figure 3. Approximating a Decreasing Stress-Strain Curve With a Stress-Based Algorithm.

Also, there seems to be a problem with the B-P directional hardening constants for both the Ti- β 21s and Ti-15-3 materials under large reversed viscoplastic strain. For instance, after several cycles in some high stress cyclic loading cases, the solution fails to converge during unloading. Little can be done about this problem without using a different material model or reevaluating the viscoplastic constants.

Memory Considerations

Before running any calculations of large numbers of cycles (in the thousands), please check the available memory on your machine. All the composite information for each load point is written to a scratch file while the program is running. If LISOL consumes all the available memory, it will crash and possibly cause other applications to crash. To avoid this, first run LISOL for a smaller number of cycles (e.g. 500). Save the results to an unformatted file, and multiply the size of this file appropriately. If the machine does not have at least this amount of memory remaining along with a substantial buffer, do not attempt to run the program for such a high number of cycles without freeing up some memory.

Summary

As indicated by the material property files given with the program, the main use of the LISOL code to date has been in the area of titanium-based metal matrix composites. However, any continuously reinforced composite system may be analyzed with the code

if the appropriate material properties are used. Also, the code was created as part of a research effort, and as such, is a research code. Therefore, it is not maintained and updated like a commercial code. Any comments or questions concerning the code as well as requests for the code should be directed to the Department of Aeronautics and Astronautics, Air Force Institute of Technology, 2950 P Street, Wright-Patterson AFB, OH, 45433-7765.

References

- [1] Robertson, David D., "A Nonlinear Three-Dimensional Micromechanics Model for Fiber-Reinforced Laminated Composites," Ph.D. Dissertation, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, AFIT/DS/AA/93-3, November 1993.
- [2] Aboudi, J. "Micromechanical Analysis of Composites by the Method of Cells," *Applied Mechanics Review*, **42**(7), July 1989, pp.193-221
- [3] Robertson, D. D., and Mall, S., "Micromechanical Relations for Fiber-Reinforced Composites Using the Free Transverse Shear Approach," *Journal of Composites Technology & Research*, **15**(3), (1993), pp. 181-192.
- [4] Aboudi, J. "Closed Form Constitutive Equations for Metal-Matrix Composites," *International Journal of Engineering Science*, **25** (1987), pp.1229-40

- [5] Robertson, D. D., and Mall, S., "Micromechanical Analysis for Thermoviscoplastic Behavior of Unidirectional Fibrous Composites," *Composites Science and Technology*, **50**, 1994, pp. 483-496.
- [6] Robertson, D. D., and Mall, S., "A Nonlinear Micromechanics Based Analysis of Metal Matrix Composite Laminates," *Composites Science and Technology*, **52**(3), (1994), pp. 319-331
- [7] Robertson, D. D., and Mall, S., "Micromechanical Analysis of Metal Matrix Composite Laminates With Fiber/Matrix Interfacial Damage," *Composites Engineering*, **4**(12), 1994, pp 1257-1274
- [8] Robertson, David D., and Mall, Shankar, "Analysis of the Thermo-Mechanical Fatigue Response of Metal Matrix Composite Laminates With Interfacial Normal and Shear Failure," *Thermo-Mechanical Fatigue Behavior of Materials: 2nd Volume, ASTM STP 1263*, Michael J. Verrilli and Michael G. Castelli, Eds., American Society for Testing and Materials, Philadelphia, 1996, pp. 236-251

Appendix - A, Material Property Files

Property File for SCS6/Ti-15-3

SCS6/Ti-15-3

Y Y N Y

10., 1.0, 0.35

21.11, 393., 0.2, 393., 0.2, 157.2

3.52911e-06, 3.52911e-06

93.33, 390., 0.2, 390., 0.2, 156.0

3.56562e-06, 3.56562e-06

204.44, 386., 0.2, 386., 0.2, 154.4

3.58091e-06, 3.58091e-06

315.56, 382., 0.2, 382., 0.2, 152.8

3.61545e-06, 3.61545e-06

426.67, 378., 0.2, 378., 0.2, 151.2

3.67056e-06, 3.67056e-06

537.78, 374., 0.2, 374., 0.2, 149.6

3.73890e-06, 3.73890e-06

648.89, 370., 0.2, 370., 0.2, 148.0

3.81488e-06, 3.81488e-06

760.00, 365., 0.2, 365., 0.2, 146.0

3.89375e-06, 3.89375e-06

871.11, 361., 0.2, 361., 0.2, 144.4

3.97195e-06, 3.97195e-06

1093.3, 354., 0.2, 354., 0.2, 141.6

4.09657e-06, 4.09657e-06

7, 0., 10000., 23.

25., 91.8, 0.36, 91.8, 0.36, 33.75

8.4800e-06, 8.4800e-06, 0., 0.

0., 0., 0., 0., 0.

1200., 4.5, 0., 1300., 3., 1.0e-08

.005, 1200., 3., 1.0e-08, 250.

315., 80.44, 0.36, 80.44, 0.36, 29.57
9.16e-06, 9.16e-06, 0., 0.
0., 0., 0., 0., 0.
1070., 2.9, 0., 1300., 3., 4.4e-06
.04, 1070., 3., 4.4e-06, 454.

427., 77.5, 0.36, 77.5, 0.36, 27.5
9.4e-06, 9.4e-06, 0., 0.
0., 0., 0., 0., 0.
1020., 2.7, 0., 1300., 3., 1.0e-05
.05, 1020., 3., 1.0e-05, 550.

482., 72.24, 0.36, 72.24, 0.36, 26.56
9.71e-06, 9.71e-06, 0., 0.
0., 0., 0., 0., 0.
850., 1.6, 0., 1300., 3., 1.
5, 850., 3., 1.0, 1100.

566., 64.4, 0.36, 64.4, 0.36, 23.68
9.98e-06, 9.98e-06, 0., 0.
0., 0., 0., 0., 0.
700., 1.05, 0., 1300., 3., 0.79
8., 700., 3., 0.79, 2400.

649., 53.00, 0.36, 53.00, 0.36, 19.49
10.260e-06, 10.260e-06, 0., 0.
0., 0., 0., 0., 0.
600.0, 0.7, 0., 1300., 3., 0.2
10.00, 600.0, 3., 0.2, 3800.

900., 25.00, 0.36, 25.00, 0.36, 9.19
10.5e-06, 10.5e-06, 0., 0.
0., 0., 0., 0., 0.
150., 0.5, 0., 1300., 3., 0.2
20., 150., 3., 0.2, 5000.

3, -75.
25., 95., .04
130., .20

538., 80., .04
75., .05

650., 75., .04
60., .05

Property File for SCS6/TIMETAL®21S

SCS6/TIMETAL21s

Y Y N Y

10., 1.4, 0.35

21.11, 393., 0.25, 393., 0.25, 157.2

3.52911e-06, 3.52911e-06

93.33, 390., 0.25, 390., 0.25, 156.0

3.56562e-06, 3.56562e-06

204.44, 386., 0.25, 386., 0.25, 154.4

3.58091e-06, 3.58091e-06

315.56, 382., 0.25, 382., 0.25, 152.8

3.61545e-06, 3.61545e-06

426.67, 378., 0.25, 378., 0.25, 151.2

3.67056e-06, 3.67056e-06

537.78, 374., 0.25, 374., 0.25, 149.6

3.73890e-06, 3.73890e-06

648.89, 370., 0.25, 370., 0.25, 148.0

3.81488e-06, 3.81488e-06

760.00, 365., 0.25, 365., 0.25, 146.0

3.89375e-06, 3.89375e-06

871.11, 361., 0.25, 361., 0.25, 144.4

3.97195e-06, 3.97195e-06

1093.3, 354., 0.25, 354., 0.25, 141.6

4.09657e-06, 4.09657e-06

16, 0., 10000., 23.

23., 112., 0.34, 112., 0.34, 41.79

6.3100e-06, 6.3100e-06, 0., 0.

0., 0., 0., 0., 0.

1550., 4.8, 0., 1600., 3., 0.

0.35, 1550., 3., 0., 100.

260., 108., 0.34, 108., 0.34, 40.30

7.2600e-06, 7.2600e-06, 0., 0.

0., 0., 0., 0., 0.
 1300., 3.5, 0., 1600., 3., 0.
 0.35, 1300., 3., 0., 300.

315., 106.133, 0.34, 106.133, 0.34, 39.60
 7.483e-06, 7.483e-06, 0., 0.
 0., 0., 0., 0., 0.
 1250.5, 3.054, 0., 1600., 3., 0.000044
 1.5, 1250.5, 3., 0.000044, 390.

365., 104.089, 0.34, 104.089, 0.34, 38.84
 7.684e-06, 7.684e-06, 0., 0.
 0., 0., 0., 0., 0.
 1205.4, 2.649, 0., 1600., 3., 0.000274
 2.55, 1205.4, 3., 0.000274, 500.

415., 101.74, 0.34, 101.74, 0.34, 37.96
 7.884e-06, 7.884e-06, 0., 0.
 0., 0., 0., 0., 0.
 1160.4, 2.243, 0., 1600., 3., 0.001304
 3.60, 1160.4, 3., 0.001304, 660.

465., 99.085, 0.34, 99.085, 0.34, 36.97
 8.085e-06, 8.085e-06, 0., 0.
 0., 0., 0., 0., 0.
 1115.3, 1.838, 0., 1600., 3., 0.00503
 4.64, 1115.3, 3., 0.00503, 960.

482., 98.113, 0.34, 98.113, 0.34, 36.60
 8.150e-06, 8.150e-06, 0., 0.
 0., 0., 0., 0., 0.
 1100.0, 1.700, 0., 1600., 3., 0.0076
 5.00, 1100.0 3., 0.0076, 1100.

500., 97.045, 0.34, 97.045, 0.34, 36.21
 8.225e-06, 8.225e-06, 0., 0.
 0., 0., 0., 0., 0.
 1089.3, 1.500, 0., 1600., 3., 0.01165
 5.763, 1089.3 3., 0.01165, 1300.

525., 95.497, 0.34, 95.497, 0.34, 35.63
 8.325e-06, 8.325e-06, 0., 0.
 0., 0., 0., 0., 0.
 1074.4, 1.280, 0., 1600., 3., 0.0203
 6.822, 1074.4 3., 0.0203, 1670.

550., 93.873, 0.34, 93.873, 0.34, 35.03
8.426e-06, 8.426e-06, 0., 0.
0., 0., 0., 0., 0.
1059.5, 1.100, 0., 1600., 3., 0.0342
7.881, 1059.5 3., 0.0342, 2100.

575., 92.172, 0.34, 92.172, 0.34, 34.39
8.526e-06, 8.526e-06, 0., 0.
0., 0., 0., 0., 0.
1044.6, 0.97, 0., 1600., 3., 0.0559
8.941, 1044.6 3., 0.0559, 2600.

600., 90.395, 0.34, 90.395, 0.34, 33.73
8.626e-06, 8.626e-06, 0., 0.
0., 0., 0., 0., 0.
1029.8, 0.82, 0., 1600., 3., 0.0887
10.00, 1029.8 3., 0.0887, 3700.

650., 86.612, 0.34, 86.612, 0.34, 32.31
8.830e-06, 8.830e-06, 0., 0.
0., 0., 0., 0., 0.
1000.0, 0.74, 0., 1600., 3., 0.2100
10.00, 1000.0 3., 0.2100, 3800.

760., 77.216, 0.34, 77.216, 0.34, 28.81
9.270e-06, 9.270e-06, 0., 0.
0., 0., 0., 0., 0.
600.0, 0.58, 0., 1600., 3., 1.00
15.00, 600.0 3., 1.00, 4000.

815., 71.964, 0.34, 71.964, 0.34, 26.87
9.490e-06, 9.490e-06, 0., 0.
0., 0., 0., 0., 0.
300.0, 0.55, 0., 1600., 3., 2.00
30.00, 300.0 3., 2.00, 4100.

900., 63.122, 0.34, 63.122, 0.34, 23.55
9.779e-06, 9.779e-06, 0., 0.
0., 0., 0., 0., 0.
300.0, 0.55, 0., 1600., 3., 2.00
30.00, 300.0 3., 2.00, 4300.

1, -5.
25., 3., .01
100., .01

Appendix - B Program Listing

```
C
C
C
C * * * * *
C * * * * *
C
C     LISOL stands for Laminated composite Inelastic SOLver.
C     It is a three dimensional nonlinear micromechanics model that
C     provides solutions for nonlinear composite laminate
C     behavior.
C * * * * *
C * * * * *
C
C     Copyright 1995 U.S. Government
C     All rights reserved
C
C This material may be reproduced by or for the U.S. Government
C
C Programmed by:
C
C David D. Robertson (513) 255-3636 ext. 4597
C Air Force Institute of Technology
C Wright-Patterson AFB, OH 45433-7765
C
C-----
C THIS SOFTWARE AND ANY ACCOMPANYING DOCUMENTATION IS RELEASED AS
C IS. THE U.S. GOVERNMENT, ITS CONTRACTORS AND THEIR SUBCONTRACTORS
C MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, CONCERNING THIS
C SOFTWARE AND ANY ACCOMPANYING DOCUMENTATION, INCLUDING, WITHOUT
C LIMITATION, ANY WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
C PARTICULAR PURPOSE. IN NO EVENT WILL THE U.S. GOVERNMENT, ITS
C CONTRACTORS AND THEIR SUBCONTRACTORS BE LIABLE FOR ANY DAMAGES,
C INCLUDING LOST PROFITS, LOST SAVINGS OR OTHER INCIDENTAL DAMAGES,
C EVEN IF INFORMED IN ADVANCE OF THE POSSIBILITY OF SUCH DAMAGES.
C
C "This software is being used at the user's own risk: Neither the
C Government Agency nor its contractors assure software's accuracy
C or its appropriate use."
C-----
C
C
C * * * * *
C
C
C
C
C     PROGRAM LISOL
C
C     REAL NUF12(40),NUF23(40),NUM12(40),NUM23(40),NX(9999),NY(9999),
C     > NXY(9999)
C
C     CHARACTER FIB*1,MATE*1,MATP*1,MATV*1,TCHARP*1,TCHARV*1,
C     >COMP*40, ECHAR*1,LCHAR*1
```

COMMON/TYPEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

COMMON/MATERIAL/NUF12(40),NUF23(40),NUM12(40),NUM23(40),GM12(40),
>EM22(40),EM11(40),AR,VF,GF12(40),EF22(40),EF11(40),HPRAMPTS(40),
>YSPTS(40),TEMPROPF(40),TEMPROP(40),NTEMF,NTEMM,HD,ALPHAF1(40),
>ALPHAF2(40),ALPHAM1(40),ALPHAM2(40),OSAT(40),F1(40),F3(40),Z0(40),
>BN(40),DNOT,Z0D(40),BND(40),BM1D(40),Z1D(40),R1D(40),
>A1D(40),BM2D(40),Z2D(40),R2D(40),A2D(40),Z3D(40),TEMPREF,NTEMI,
>SIC,TEMPROPI(40),SIFN1(40),UIFN1OA(40),SIFT1(40),UIFT1OA(40)

COMMON/LOADING/NC1,NC2,NUMP,NUMCYC,DT(9999),TEMP(9999),NX(9999),
>NY(9999),NXY(9999),M(9999)

COMMON/ELASTPROP/SSF11(40),SSF22(40),SSF12(40),SSF23(40),
>SSF44(40),SSM11(40),SSM22(40),SSM12(40),SSM23(40),SSM44(40),
>A(6),B(6),C(6)

COMMON/TOLERANCE/TOLP1,TOLP2,TOLP3,TOLV1,TOLV2,TOLV3,TOLV4

COMMON/WORDS/FIB,MATE,MATP,MATV,TCHARP,TCHARV,COMP,ECHAR,LCHAR

CHARACTER CHAR1*1

OPEN(8,STATUS='SCRATCH',FORM='UNFORMATTED')

C

C * * DEFAULTS * *

C

NA = 3
NUMPLY=2
ZPOS(1)=-.5
ZPOS(2)=0.
ZPOS(3)=0.5
ANGLE(1)=0.
ANGLE(2)=90.
FIB = 'N'
MATE = 'N'
MATP = 'N'
MATV = 'N'
TCHARP = 'N'
TCHARV = 'N'
YSPTS(1) = 0.
HPRAMPTS(1) = 0.
TEMPROPF(1) = 21.
TEMPROP(1) = 21.
HD=0.
NUMCYC = 1
ECHAR = 'N'
LCHAR = 'N'
COMP = 'NO NAME'
TOLP1=.05
TOLP2=.005

```

TOLP3=1.0
TOLV1= 200.
TOLV2= 15.0
TOLV3=.001
TOLV4=50.

10  WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*) ' * * * * * '
    WRITE(6,*) ' *               LISOL MAIN MENU               * '
    WRITE(6,*) ' * * * * * '
    WRITE(6,*)
    WRITE(6,*) '          (1)  Type of Model  '
    WRITE(6,*)
    WRITE(6,*) '          (2)  Constituent Properties  '
    WRITE(6,*)
    WRITE(6,*) '          (3)  Loading  '
    WRITE(6,*)
    WRITE(6,*) '          (4)  Solve for Response to Applied Loading'
    WRITE(6,*)
    WRITE(6,*) '          (5)  Examine Results  '
    WRITE(6,*)
    WRITE(6,*) '          (6)  Quit  '
    WRITE(6,*)
    WRITE(6,*) '  Select a submenu (1 to 7): '

    READ(5,*) NF

    IF(NF.EQ.1) THEN

        CALL MODEL

    ELSE IF(NF.EQ.2) THEN

        CALL PROP

    ELSE IF(NF.EQ.3) THEN

        CALL LOAD

    ELSE IF(NF.EQ.4) THEN

        IF(FIB.EQ.'N'.OR.MATE.EQ.'N'.OR.

```

```

      > (TCHARP.EQ.'N'.AND.TCHARV.EQ.'N')) THEN
        WRITE(6,100)
100  FORMAT(' Constituent properties and loading must first be ',
>'specified.')
        READ(5,*)
        GO TO 10
      END IF

      REWIND 8

      LCHAR = 'Y'

      IF(NA.EQ.2) THEN

        WRITE(6,*)
        WRITE(6,*)
        WRITE(6,*)
        WRITE(6,*) ' Set New Convergence Tolerances (y/n)'
        WRITE(6,*)
        READ(5,*) CHAR1

        PERTOL1 = TOLP1*100.
        PERTOL2 = TOLP2*100.

        IF(CHAR1.EQ.'y'.OR.CHAR1.EQ.'Y') THEN
1          WRITE(6,*)
          WRITE(6,1000) PERTOL1,PERTOL2,TOLP3
1000  FORMAT(' (1) Percent load past yield before incrementing = ',
>F5.2,/, ' (2) Percent yield surface convergence tolerance = ',
>F5.2,/, ' (3) Instability convergence factor = ',F5.2,/,
>' (4) Solve',/, 'Enter selection:')
          READ(5,*) NTOL
          IF(NTOL.EQ.1) THEN
            WRITE(6,*) ' Enter new incrementing parameter:'
            READ(5,*) PERTOL1
            GO TO 1
          ELSEIF(NTOL.EQ.2) THEN
            WRITE(6,*) ' Enter new yield surface thickness:'
            READ(5,*) PERTOL2
            GO TO 1
          ELSEIF(NTOL.EQ.3) THEN
            WRITE(6,*) ' Enter new convergence factor:'
            READ(5,*) TOLP3
            GO TO 1
          ELSEIF(NTOL.NE.4) THEN
            GO TO 1
          END IF

          TOLP1=PERTOL1/100.
          TOLP2=PERTOL2/100.
        END IF

        ELSEIF(NA.EQ.3) THEN

```

```

        WRITE(6,*)
        WRITE(6,*)
        WRITE(6,*)
        WRITE(6,*) ' Set New Convergence Tolerances (y/n)'
        WRITE(6,*)
        READ(5,*) CHAR1

        IF(CHAR1.EQ.'y'.OR.CHAR1.EQ.'Y') THEN
2          WRITE(6,*)
          WRITE(6,2000) TOLV1,TOLV2,TOLV3,TOLV4
2000 FORMAT(' (1) Multiple of elastic rates before incrementing = ',
>F6.1,/, ' (2) Instability convergence factor = ',
>F5.2,/, ' (3) Convergence error tolerance = ',F7.5,/,
>' (4) Amount above interface failure before incrementing = ',
>F5.0,/,
>' (5) Solve',/, ' (6) Return to main menu',/, 'Enter selection:')
          READ(5,*) NTOL
          IF(NTOL.EQ.1) THEN
            WRITE(6,*) ' Enter new incrementing parameter:'
            READ(5,*) TOLV1
            GO TO 2
          ELSEIF(NTOL.EQ.2) THEN
            WRITE(6,*) ' Enter new factor (increase for more stability):'
            READ(5,*) TOLV2
            GO TO 2
          ELSEIF(NTOL.EQ.3) THEN
            WRITE(6,*) ' Enter new error tolerance:'
            READ(5,*) TOLV3
            GO TO 2
          ELSEIF(NTOL.EQ.4) THEN
            WRITE(6,*) ' Enter new interface failure increment:'
            READ(5,*) TOLV4
            GO TO 2
          ELSEIF(NTOL.EQ.6) THEN
            GO TO 10
          ELSEIF(NTOL.NE.5) THEN
            GO TO 2
          END IF

        END IF

        END IF

        CALL SOLV

        ENDFILE 8

        ELSE IF(NF.EQ.5) THEN

          CALL RESULT

```

```

ELSE IF(NF.EQ.6) THEN

    CLOSE(8)
    STOP

END IF

GO TO 10

END

C  * * * * *
C
C  BIGPMAT enters the matrix P for the LISOL model.
C
C  * * * * *

SUBROUTINE BIGPMAT

COMMON/TPEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

COMMON/ELASTPROP/SSF11(40),SSF22(40),SSF12(40),SSF23(40),
>SSF44(40),SSM11(40),SSM22(40),SSM12(40),SSM23(40),SSM44(40),
>A(6),B(6),C(6)

COMMON/VISCOPLAST/H(50,50),HINV(50,50),T(6,3,3),TINT(6,3,3),
>TH(50,20),BIGP(50,50),N,NCYC,TEM0,TIM0,PZI(6,3),LVEC0(50),
>EPSMP0(6,3,4),SM0(6,3,4),UIF0(6,2,2),SI0(6,2,2),UI0(6,2,2),
>K2(6,3),BETA0(6,3,4),OMEG0(6,3,4),EPSTHM1,EPSTHM2,EPSTHF1,
>EPSTHF2,NDUMP

COMMON/PMATRIX/PP(6,8,12),S11F,S12F,S22F,S23F,S44F,S11M,S12M,
>S22M,S23M,S44M,SSI(6,2),SSIT(6,2)

REAL LVEC0(50),K2(6,3)

R1=S11M/S12M

DO I=1,NUMPLY

    T1=A(I)+B(I)
    T2=A(I)+C(I)
    T3=B(I)/A(I)
    T4=C(I)/A(I)

    AA=(1+T4)*(S44F+SSIT(I,1))
    BB=S44M+T4*(S44F+SSIT(I,1))
    I8=8*I
    I5=5*I

    DO J=1,3
        BIGP(I8-7,J)=TINT(I,1,J)
    
```

```

      BIGP(I8-6,J)=-T1*TINT(I,2,J)
      BIGP(I8-5,J)=0.
      BIGP(I8-4,J)=0.
      BIGP(I8-3,J)=0.
      BIGP(I8-2,J)=0.
      BIGP(I8-1,J)=0.
      BIGP(I8,J)=T1/(S44M*(A(I)*AA/BB+B(I)))*TINT(I,3,J)
END DO

DO J=4,6
      BIGP(I8-7,J)=-TH(I8-7,J)
      BIGP(I8-6,J)=-TH(I8-6,J)
      BIGP(I8-5,J)=-TH(I8-5,J)
      BIGP(I8-4,J)=-TH(I8-4,J)
      BIGP(I8-3,J)=-TH(I8-3,J)
      BIGP(I8-2,J)=-TH(I8-2,J)
      BIGP(I8-1,J)=-TH(I8-1,J)
      BIGP(I8,J)=-TH(I8,J)
END DO

      BIGP(I8-7,I5+2)=-S11F
      BIGP(I8-7,I5+3)=-S12F
      BIGP(I8-7,I5+4)=0.
      BIGP(I8-7,I5+5)=T3*S12F
      BIGP(I8-7,I5+6)=0.

      BIGP(I8-6,I5+2)=A(I)*S12F
      BIGP(I8-6,I5+3)=A(I)*S22F+B(I)*S22M+A(I)*SSI(I,1)
      BIGP(I8-6,I5+4)=B(I)*S12M
      BIGP(I8-6,I5+5)=B(I)*(S23M-S23F)
      BIGP(I8-6,I5+6)=0.

      BIGP(I8-5,I5+2)=S11F
      BIGP(I8-5,I5+3)=S12F-S12M
      BIGP(I8-5,I5+4)=-S11M
      BIGP(I8-5,I5+5)=-S12M-T3*S12F
      BIGP(I8-5,I5+6)=0.

      BIGP(I8-4,I5+2)=A(I)*S12F
      BIGP(I8-4,I5+3)=A(I)*(S22F-S22M+SSI(I,1))
      BIGP(I8-4,I5+4)=B(I)*S12M-R1*S22M*T1
      BIGP(I8-4,I5+5)=-T1*S12M/R1+B(I)*(S23M-S23F)
      BIGP(I8-4,I5+6)=-T1*(S12M-R1*S22M)

      BIGP(I8-3,I5+2)=A(I)*S12F
      BIGP(I8-3,I5+3)=A(I)*(S23F-S23M)
      BIGP(I8-3,I5+4)=-A(I)*S12M
      BIGP(I8-3,I5+5)=C(I)*(1+T3)*S12M/R1-B(I)*S22F-(T2+T3*C(I))
      >          *S22M-B(I)*SSI(I,2)
      BIGP(I8-3,I5+6)=0.

      BIGP(I8-2,I5+2)=A(I)**2./(T1*T2)
      BIGP(I8-2,I5+3)=0.
      BIGP(I8-2,I5+4)=A(I)*B(I)/(T1*T2)

```

```

      BIGP(I8-2,I5+5)=C(I)/(T2*R1)
      BIGP(I8-2,I5+6)=C(I)/T2

      BIGP(I8-1,I5+2)=0.
      BIGP(I8-1,I5+3)=1.
      BIGP(I8-1,I5+4)=C(I)*R1/T2
      BIGP(I8-1,I5+5)=0.
      BIGP(I8-1,I5+6)=-C(I)*R1/T2

      BIGP(I8,I5+2)=0.
      BIGP(I8,I5+3)=0.
      BIGP(I8,I5+4)=0.
      BIGP(I8,I5+5)=0.
      BIGP(I8,I5+6)=0.

      END DO

      RETURN

      END

C      * * * * *
C
C      BIGPPMAT enters the matrix Pplast for each ply
C
C      * * * * *

      SUBROUTINE BIGPPMAT

      COMMON/TYPEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

      COMMON/ELASTPROP/SSF11(40),SSF22(40),SSF12(40),SSF23(40),
>SSF44(40),SSM11(40),SSM22(40),SSM12(40),SSM23(40),SSM44(40),
>A(6),B(6),C(6)

      COMMON/PMATRIX/PP(6,8,12),S11F,S12F,S22F,S23F,S44F,S11M,S12M,
>S22M,S23M,S44M,SSI(6,2),SSIT(6,2)

      R1=S12M/S11M
      DO I=1,NUMPLY

      T4=C(I)/A(I)

      AA=(1+T4)*(S44F+SSIT(I,1))
      BB=S44M+T4*(S44F+SSIT(I,1))

      DO J=1,12
        PP(I,1,J)=0.
      END DO
      DO J=1,4
        PP(I,2,J)=0.
      END DO
      PP(I,2,5)=B(I)

```

```

DO J=6,12
  PP(I,2,J)=0.
END DO
DO J=3,7
  DO K=1,3
    PP(I,J,K)=0.
  END DO
END DO
DENOM=S44M*(A(I)*AA/BB+B(I))
PP(I,8,1)=-B(I)/(DENOM*(1+C(I)/A(I)))
PP(I,8,2)=-B(I)*C(I)/(DENOM*(A(I)+C(I)))
PP(I,8,3)=-A(I)*(1-S44M/BB)/DENOM
DO J=4,12
  PP(I,8,J)=0.
END DO

PP(I,3,4)=-1.
DO J=5,12
  PP(I,3,J)=0.
END DO

APB=A(I)+B(I)
APC=A(I)+C(I)

PP(I,4,4)=-APB*S22M/S12M
PP(I,4,5)=B(I)
PP(I,4,6)=0.
PP(I,4,7)=-A(I)*R1+APB*S22M/S12M
PP(I,4,8)=-B(I)
PP(I,4,9)=0.
PP(I,4,10)=A(I)*R1
PP(I,4,11)=-A(I)
PP(I,4,12)=0.

PP(I,5,4)=0.
PP(I,5,5)=0.
PP(I,5,6)=-A(I)
PP(I,5,7)=C(I)*R1
PP(I,5,8)=0.
PP(I,5,9)=-C(I)
PP(I,5,10)=-C(I)*R1
PP(I,5,11)=0.
PP(I,5,12)=C(I)

PP(I,6,4)=0.
PP(I,6,5)=0.
PP(I,6,6)=0.
PP(I,6,7)=A(I)*C(I)/(APB*APC*S11M)
PP(I,6,8)=0.
PP(I,6,9)=0.
PP(I,6,10)=-PP(I,6,7)
PP(I,6,11)=0.
PP(I,6,12)=0.

```

```

      PP(I,7,4)=C(I)/(APC*S12M)
      PP(I,7,5)=0.
      PP(I,7,6)=0.
      PP(I,7,7)=-PP(I,7,4)
      PP(I,7,8)=0.
      PP(I,7,9)=0.
      PP(I,7,10)=0.
      PP(I,7,11)=0.
      PP(I,7,12)=0.

      END DO
      RETURN

      END

C
C
C * * * * *
C
C   The subroutine BODDIR calculates the inelastic strain rate using
C   the Bodner-Partom model with directional hardening.
C
C * * * * *
C
C
      SUBROUTINE BODDIR(Z0D1,BND1,BM1D1,Z1D1,R1D1,A1D1,BM2D1,Z2D1,R2D1,
>A2D1,Z3D1,DZ1DT,DZ2DT,DZ3DT,DNOT,TDOT,DTIM,SMP,SM,RT,EPSEFFDP,
>BETA,BETA0,ZI,ZI0,EPSCALDP)

      COMMON/TYEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

      REAL EPSEFFDP(6,3),SMP(6,3,4),RT(6,3),EPSCALDP(6,3),K2(6,3),
>SM(6,3,4),BETA(6,3,4),BETA0(6,3,4),ZI(6,3),ZI0(6,3),
>EPSMPD(6,3,4),WPD(6,3),U(6,3,4),DZI(6,3),DBETA(6,3,4)

      DO I=1,NUMPLY

        DO J=1,3

          K2(I,J)=0.5*(SMP(I,J,1)**2+SMP(I,J,2)**2+SMP(I,J,3)**2+
> 2.*SMP(I,J,4)**2)
          RT(I,J)=SQRT(3.*K2(I,J))
          DENOM = SQRT(SM(I,J,1)**2+SM(I,J,2)**2+SM(I,J,3)**2+
> 2.*SM(I,J,4)**2)
          IF(RT(I,J).LT.5.) THEN
            EPSCALDP(I,J)=0.0
          ELSE
            DO K=1,4
              U(I,J,K)=SM(I,J,K)/DENOM
              EPSMPD(I,J,K) = 1.5*EPSEFFDP(I,J)*SMP(I,J,K)/RT(I,J)
            END DO
          END IF
        END DO
      END DO

```

```

      EPSMPD(I,J,4)=2.*EPSMPD(I,J,4)
      WPD(I,J)=SM(I,J,1)*EPSMPD(I,J,1)+SM(I,J,2)*EPSMPD(I,J,2)
>      +SM(I,J,3)*EPSMPD(I,J,3)+SM(I,J,4)*EPSMPD(I,J,4)
      Q=-BM1D1*WPD(I,J)-A1D1*Z1D1**(1.-R1D1)*(ZI(I,J)-Z2D1)**R1D1
>      /ZI(I,J)+(DZ1DT-DZ2DT)/(Z1D1-Z2D1)*TDOT
      DZI(I,J)=(BM1D1*Z1D1*WPD(I,J)+(-Z2D1*DZ1DT+Z1D1*DZ2DT)*TDOT/
>      (Z1D1-Z2D1)+ZI0(I,J)*Q)/(1/DTIM-Q)
      DUM=SQRT(BETA(I,J,1)**2+BETA(I,J,2)**2+BETA(I,J,3)**2+2.*
>      BETA(I,J,4)**2.)
      Q2 = -BM2D1*WPD(I,J)-A2D1*(DUM/Z1D1)**(R2D1-1.)+DZ3DT*
>      TDOT/Z3D1
      DO K=1,4
        DBETA(I,J,K)=(BM2D1*Z3D1*WPD(I,J)*U(I,J,K)+BETA0(I,J,K)*
>        Q2)/(1./DTIM - Q2)
        BETA(I,J,K)=BETA0(I,J,K)+DBETA(I,J,K)
      END DO
      ZI(I,J)=ZI0(I,J)+DZI(I,J)
      IF(ZI(I,J).LT.Z2D1) THEN
        ZI(I,J)=Z2D1
      ELSEIF(ZI(I,J).GT.Z1D1) THEN
        ZI(I,J)=Z1D1
      END IF
      ZD = BETA(I,J,1)*U(I,J,1)+BETA(I,J,2)*U(I,J,2)+BETA(I,J,3)*
>      U(I,J,3)+2.*BETA(I,J,4)*U(I,J,4)
      EPSCALDP(I,J) = 2./SQRT(3.)*DNOT*EXP(-0.5*((ZI(I,J)+ZD)/
>      RT(I,J))**(2*BND1))
      END IF

```

END DO

END DO

RETURN

END

C

C

C * * * * *

C

C The subroutine BODISO calculates the inelastic strain rate using
C the original Bodner-Partom model with isotropic hardening only.

C

C * * * * *

C

C

```

      SUBROUTINE BODISO(Z0A1,PNA1,PMA1,Z1A1,DNOT,DTIM,SMP,SM,RT,
>ZI,ZI0,EPSEFFDP,EPSCALDP)

```

```

      COMMON/TYEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

```

```

      REAL EPSEFFDP(6,3),SMP(6,3,4),RT(6,3),
>EPSCALDP(6,3),K2(6,3),SM(6,3,4),ZI(6,3),ZI0(6,3),

```

```

>EPSMPD(6,3,4),WPD(6,3)

DO I=1,NUMPLY
  DO J=1,3
    K2(I,J)=0.5*(SMP(I,J,1)**2+SMP(I,J,2)**2+SMP(I,J,3)**2+
    >      2.*SMP(I,J,4)**2)
    RT(I,J)=SQRT(3.*K2(I,J))
    IF(RT(I,J).LT.5.) THEN
      EPSCALDP(I,J)=0.0
    ELSE
      DO K=1,4
        EPSMPD(I,J,K) = 1.5*EPSEFFDP(I,J)*SMP(I,J,K)/RT(I,J)
      END DO
      EPSMPD(I,J,4)=2.*EPSMPD(I,J,4)
      WPD(I,J)=SM(I,J,1)*EPSMPD(I,J,1)+SM(I,J,2)*EPSMPD(I,J,2)
    >      +SM(I,J,3)*EPSMPD(I,J,3)+SM(I,J,4)*EPSMPD(I,J,4)
      Q=PMA1*WPD(I,J)/Z0A1
      DZI=Q*(Z1A1-ZI0(I,J))/(1/DTIM+Q)
      ZI(I,J)=ZI0(I,J)+DZI
      IF(ZI(I,J).LT.Z0A1) THEN
        ZI(I,J)=Z0A11
      ELSEIF(ZI(I,J).GT.Z1A1) THEN
        ZI(I,J)=Z1A1
      END IF
      POW = -0.5*(PNA1+1)/PNA1*(ZI(I,J)/RT(I,J))**(2*PNA1)
      EPSCALDP(I,J) = 2./SQRT(3.)*DNOT*EXP(POW)
    END IF
  END DO
END DO

RETURN

END
C * * * * *
C
C GAUSS performs Gauss elimination to produce the solution to
C a matrix-vector equation.
C
C * * * * *

SUBROUTINE GAUSS(N,MAT,FVEC,XVEC)

REAL MAT(50,50),FVEC(50),XVEC(50),AMAT(50,51),SMAT(50,51)

C
C Create the augmented matrix consisting of an N X (N+1) matrix
C where MAT is the NxN portion and FVEC is the final column.
C
DO I=1,N
  DO J=1,N
    AMAT(I,J)=MAT(I,J)
  END DO

```

```

        AMAT(I,N+1)=FVEC(I)
    END DO

C
C Begin Gauss elimination
C

    DO K=1,N-1

C
C First find the maximum coefficient in column K whose row is
C greater than or equal to K.
C

        DUM=0.

        DO I=K,N
            IF (ABS (AMAT(I,K)) .GT. ABS (DUM)) THEN
                J=I
                DUM=AMAT(I,K)
            END IF
        END DO
        IF (DUM.EQ.0.) THEN
            WRITE(6,*)
            WRITE(6,*)
            WRITE(6,*)
            WRITE(6,*) '      Matrix has no inverse !'
            WRITE(6,*)
            WRITE(6,*) ' Attempted to invert a singular matrix'
            WRITE(6,*) ' in subroutine GAUSS.'
            WRITE(6,*)
            READ(5,*)
            STOP
        END IF

C
C Exchange rows K & J.
C

        IF (J.NE.K) THEN
            DO L=1,N+1
                SMAT(J,L) = AMAT(J,L)
                AMAT(J,L) = AMAT(K,L)
                AMAT(K,L) = SMAT(J,L)
            END DO
        END IF

C
C Perform Gauss elimination
C

        DO J=K+1,N

            AMULT=AMAT(J,K)/AMAT(K,K)

```

```

      DO I=K+1,N+1

          AMAT(J,I)=AMAT(J,I)-AMULT*AMAT(K,I)

      END DO

END DO

END DO

C
C  Start back substitution
C
      IF (AMAT(N,N).EQ.0.) THEN
          WRITE(6,*)
          WRITE(6,*)
          WRITE(6,*) '      Matrix has no inverse !'
          WRITE(6,*)
          WRITE(6,*) ' Attempted to invert a singular matrix'
          WRITE(6,*) ' in subroutine GAUSS.'
          WRITE(6,*)
          READ(5,*)
          STOP
          END IF

      XVEC(N)=AMAT(N,N+1)/AMAT(N,N)

      DO K=1,N-1
          I=N-K
          DUM=0.
          DO ISUM=I+1,N
              DUM=DUM+AMAT(I,ISUM)*XVEC(ISUM)
          END DO
          XVEC(I)=(AMAT(I,N+1)-DUM)/AMAT(I,I)
      END DO

      RETURN

      END
C * * * * *
C
C  I N V E R S E
C
C * * * * *
C
C      The subroutine INVERSE calculates the inverse of a square
C      matrix, A, of size N and returns it as AINV. Gauss-Jordan
C      elimination
C      is used in the calculations. Arguments include the matrix size (N),
C      the matrix (A), and the inverse matrix (AINV).

```

```

C
C
C * * * * *
      SUBROUTINE INVERSE(N,A,AINV)

      REAL A(50,50),AINV(50,50),AI(50,100),S(50,100)
      INTEGER N,I,J,K,M

C
C      Create the augmented matrix, AI, which is an N x 2N matrix
C      containing the matrix A in the first N columns and the identity
C      matrix in columns N+1 to 2N.
C

      DO J = 1, N
        DO I = 1, N
          AI(I,J) = A(I,J)
          AI(I,J+N) = 0.
        END DO
      END DO

      DO I = 1,N
        AI(I,I+N) = 1.
      END DO

C
C * *      Begin Gauss-Jordan elimination. * *
C

      DO J=1,N

C
C      First find a row in column J with a nonzero coefficient.
C

        DUM=0.

        DO I=J,N
          IF(ABS(AI(I,J)).GT.ABS(DUM)) THEN
            K=I
            DUM=AI(I,J)
          END IF
        END DO
        IF(DUM.EQ.0.) THEN
          WRITE(6,*)
          WRITE(6,*)
          WRITE(6,*)
          WRITE(6,*) '      Matrix has no inverse !'
          WRITE(6,*)
          WRITE(6,*) ' Attempted to invert a singular matrix'
          WRITE(6,*) ' in subroutine INVERSE.'
          WRITE(6,*)
          READ(5,*)
          STOP
        END IF
      END DO

```

```

END IF

C
C   Exchange rows J & K.
C

      IF(K.NE.J) THEN
      DO L = 1,2*N
        S(K,L) = AI(K,L)
        AI(K,L) = AI(J,L)
        AI(J,L) = S(K,L)
      END DO
    END IF

C
C   Perform Gauss-Jordan elimination of each column, J.
C

      DO I=1,N
      IF(I.NE.J) THEN
        AMULT = AI(I,J)/AI(J,J)
        DO M=J+1,2*N
          AI(I,M) = AI(I,M) - AMULT*AI(J,M)
        END DO
      END IF
    END DO

C
C   Must operate on pivoted row last
C

      DO M=J+1,2*N
        AI(J,M) = AI(J,M)/AI(J,J)
      END DO

    END DO

C
C   Create AINV from the new augmented matrix
C

      DO I=1,N
        DO J=1,N
          AINV(I,J) = AI(I,J+N)
        END DO
      END DO

    RETURN

  END

C
C   * * * * *
C   *

```

```

C *                A P P L I E D                *
C *                L O A D I N G                *
C *                *                *                *
C * * * * * * * * * * * * * * * * * * * * * * *
C
C   The LOAD subroutine sets the applied loading to be
C   used with the program FTS.
C
C * * * * * * * * * * * * * * * * * * * * * * *
C

```

```

SUBROUTINE LOAD

```

```

CHARACTER CHAR1*1, CYCSYM(999)*1, INLOD*15, OUTLOD*15

```

```

CHARACTER FIB*1,MATE*1,MATP*1,MATV*1,TCHARP*1,TCHARV*1,
>COMP*40, ECHAR*1,LCHAR*1

```

```

COMMON/TPEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

```

```

COMMON/LOADING/NC1,NC2,NUMP,NUMCYC,DT(9999),TEMP(9999),NX(9999),
>NY(9999),NXY(9999),M(9999)

```

```

COMMON/WORDS/FIB,MATE,MATP,MATV,TCHARP,TCHARV,COMP,ECHAR,LCHAR

```

```

REAL NX(9999),NY(9999),NXY(9999)

```

```

3      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*) ' * * * * * '
      WRITE(6,*) ' *   Applied Loading Menu   * '
      WRITE(6,*) ' * * * * * '
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*) '      (1) Read input file '
      WRITE(6,*) '      (2) Save existing load sequence to a file '
      WRITE(6,*) '      (3) Composite load history '
      WRITE(6,*) '      (4) Return to main menu '
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*) ' Enter Selection: '
      READ(5,*) NL

      IF(NL.EQ.1) THEN

```

```

WRITE(6,*)
WRITE(6,*)' Enter the name of the load file you wish to read.'
READ(5,2001) INLOD
OPEN(UNIT=20,FILE=INLOD,STATUS='OLD')
READ(20,2003) TCHARP,TCHARV,NUMP,NC1,NC2,NUMCYC
2003 FORMAT(2A2,4I7)
DO I=1,NUMP
    READ(20,*) DT(I),TEMP(I),NX(I),NY(I),NXY(I),M(I)
END DO
CLOSE(UNIT=20)
GO TO 3

ELSE IF(NL.EQ.2) THEN

    IF(TCHARP.EQ.'N'.AND.TCHARV.EQ.'N') THEN
        WRITE(6,1032)
1032 FORMAT(' No load history has yet been entered. Please return to',
>' the Applied ',/, ' Loading menu by pressing <return>.')
        READ(5,*)
        GO TO 3
    END IF

    WRITE(6,*)
    WRITE(6,*)' Enter the name of the load file you wish to save.'
    READ(5,2001) OUTLOD
2001 FORMAT(A15)
    OPEN(UNIT=21,FILE=OUTLOD,STATUS='NEW')
    WRITE(21,2003) TCHARP,TCHARV,NUMP,NC1,NC2,NUMCYC
    DO I=1,NUMP
        WRITE(21,*) DT(I),TEMP(I),NX(I),NY(I),NXY(I),M(I)
    END DO
    CLOSE(UNIT=21)
    GO TO 3

ELSE IF(NL.EQ.3) THEN

    IF(NA.LE.2) THEN

        ELSE

14      DO I=1,NUMP
            CYCSYM(I) = ' '
            IF(I.EQ.NC1.OR.I.EQ.NC2) CYCSYM(I) = '*'
        END DO

        WRITE(6,1035)
1035 FORMAT(////, '      dt (s)   T (deg C)   NX (MPa-mm)   NY (MPa-mm)',
>'      NXY (MPa-mm) #Incr',/)

        DO I=1,NUMP
            WRITE(6,1036) CYCSYM(I),I,DT(I),TEMP(I),NX(I),NY(I),
>            NXY(I),M(I)

```

```

1036 FORMAT(A1, '(' ,I2,')' , 2F9.2, 3F11.2,I4)
      IF(I.GT.100) THEN
        WRITE(6,*) ' *****>'
        GO TO 101
      END IF
    END DO

101      WRITE(6,1027) NUMP+1,NUMP+2,NUMP+3,NUMCYC,NUMP+4
      READ(5,*) NLP
      NUMPAD = 0
1027 FORMAT(' (' ,I2,')  Add points',/,
> '(' ,I2,')  Define a repeating cycle (points from * to * are ',
> 'repeated N times)',/,
> '(' ,I2,')  Change number of cycles (present no. = ',I7,')',/,
> '(' ,I2,')  Save and return to Applied Load menu',/,
> ' Point to modify/delete or other selection:')

      DO I=1,NUMP

        IF(NLP.EQ.I) THEN
          WRITE(6,*)
          WRITE(6,*) ' Delete or Modify point',I,' ? (d/m) '
          READ(5,*) CHAR1
          IF(CHAR1.EQ.'d') THEN
            DO N=I,NUMP
              DT(N) = DT(N+1)
              TEMP(N) = TEMP(N+1)
              NX(N) = NX(N+1)
              NY(N) = NY(N+1)
              NXY(N) = NXY(N+1)
              M(N) = M(N+1)
            END DO
            NUMPAD = -1
          ELSE IF(CHAR1.EQ.'m') THEN
            WRITE(6,1028) I
1028 FORMAT(/,' Enter the applied loading for point ',I2)

            WRITE(6,1039)
1039 FORMAT('  dt (s)  T (deg C)  NX (MPa-mm)  NY (MPa-mm)',
> ' NXY (MPa-mm) #Incr')
            READ(5,*) DT(I),TEMP(I),NX(I),NY(I),NXY(I),M(I)
          END IF
        END IF

      END DO

      IF(NLP.EQ.NUMP+1) THEN
        WRITE(6,*)
        WRITE(6,*) ' Enter the number of points to add.'
        READ(5,*) NUMPAD
        WRITE(6,*)
        WRITE(6,*) ' Enter load for ',NUMPAD,' additional points.'
        WRITE(6,1039)

```

```

        DO I=NUMP+1,NUMP+NUMPAD
            READ(5,*) DT(I),TEMP(I),NX(I),NY(I),NXY(I),M(I)
        END DO
        TCHARV='Y'
    END IF

    IF(NLP.EQ.NUMP+2) THEN
        WRITE(6,1030)
1030 FORMAT(/,' Enter the load point that begins the repeating cycle.')

        READ(5,*) NC1
        WRITE(6,1031)
1031 FORMAT(/,' Enter the load point that ends the repeating cycle.')
        READ(5,*) NC2
    END IF

    IF(NLP.EQ.NUMP+3) THEN
        WRITE(6,*)
        WRITE(6,*) ' Enter the number of desired cycles.'
        READ(5,*) NUMCYC
    END IF

    IF(NLP.EQ.NUMP+4) THEN
        GO TO 3
    END IF

    NUMP=NUMP+NUMPAD
    GO TO 14

END IF

ELSE IF(NL.NE.4) THEN
    GO TO 3

END IF

RETURN

END

C
C  * * * * *
C  *
C  *                               M O D E L
C  *
C  * * * * *
C
C  The MODEL subroutine sets the type of analysis model to be
C  used with the program LISOL.
C
C  * * * * *
C
C
SUBROUTINE MODEL

CHARACTER CHAR1*1

```

```

COMMON/TPEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

REAL THICK(6)

1      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*) ' * * * * * '
      WRITE(6,*) ' *          Type of Model          * '
      WRITE(6,*) ' * * * * * '
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*) ' * Type of analysis * '
      WRITE(6,*)
      WRITE(6,*) '      **N/A** (1) Linear Elastic '
      WRITE(6,*) '      **N/A** (2) Elastic-Plastic '
      WRITE(6,*) '      (3) Visco-plastic '
      WRITE(6,*)
      WRITE(6,*) ' Present analysis selection: ',NA
      WRITE(6,*)
      WRITE(6,*) ' Number of plies = ',NUMPLY
      WRITE(6,*)
      WRITE(6,*) ' Ply thickness (mm)          Ply angle'

      DO I=1,NUMPLY
          THICK(I) = ZPOS(I+1)-ZPOS(I)
          WRITE(6,1000) THICK(I), ANGLE(I)
1000  FORMAT(5X,2F15.4)
      END DO
      WRITE(6,*)
      WRITE(6,*) '          Change above selections (y/n) ? '
      READ(5,*) CHAR1

      IF (CHAR1.EQ.'y'.OR.CHAR1.EQ.'Y') THEN

          WRITE(6,*)
          WRITE(6,*) ' Enter new analysis type (#,#):'
          READ(5,*) NA
          WRITE(6,*)
          WRITE(6,*) ' Enter number of plies:'
          READ(5,*) NUMPLY
          WRITE(6,*)
          WRITE(6,*) ' Enter thickness and angle for each ply '
          DO I=1,NUMPLY
              WRITE(6,*) ' thickness and angle for ply ',I
              READ(5,*) THICK(I),ANGLE(I)
          END DO
          TOT = 0.0
          DO I=1,NUMPLY
              TOT = TOT + THICK(I)
          END DO
          ZPOS(1)=-TOT/2.
          DO I=1,NUMPLY

```

```

        ZPOS(I+1)=ZPOS(I)+THICK(I)
    END DO
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    GO TO 1

ELSE IF (CHAR1.NE.'n') THEN
    GO TO 1

END IF

RETURN

END

C
C * * * * *
C *
C *          C O N S T I T U E N T
C *          P R O P E R T I E S
C *
C * * * * *
C
C
C   The PROP subroutine sets the constituent properties
C   for the program LISOL.
C
C
C * * * * *
C
SUBROUTINE PROP

REAL NUF12(40), NUF23(40), NUM12(40), NUM23(40)

CHARACTER FIB*1, MATE*1, MATP*1, MATV*1, TCHARP*1, TCHARV*1,
>COMP*40, ECHAR*1, LCHAR*1

CHARACTER CHAR1*1, INPROP*15, OUTPROP*15

COMMON/TYPEMOD/NA, NUMPLY, ZPOS(7), ANGLE(6)

COMMON/MATERIAL/NUF12(40), NUF23(40), NUM12(40), NUM23(40), GM12(40),
>EM22(40), EM11(40), AR, VF, GF12(40), EF22(40), EF11(40), HPRAMPTS(40),
>YSPTS(40), TEMPROPF(40), TEMPROM(40), NTEMP, NTEMM, HD, ALPHAF1(40),
>ALPHAF2(40), ALPHAM1(40), ALPHAM2(40), OSAT(40), F1(40), F3(40), Z0(40),
>BN(40), DNOT, Z0D(40), BND(40), BM1D(40), Z1D(40), R1D(40),
>A1D(40), BM2D(40), Z2D(40), R2D(40), A2D(40), Z3D(40), TEMPREF, NTEMI,
>SIC, TEMPROMI(40), SIFN1(40), UIFN1OA(40), SIFT1(40), UIFT1OA(40)

```

COMMON/WORDS/FIB,MATE,MATP,MATV,TCHARP,TCHARV,COMP,ECHAR,LCHAR

```

2      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*) ' * * * * * '
      WRITE(6,*) ' *   Constituent Properties   * '
      WRITE(6,*) ' * * * * * '
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*) '      (1) Read input file '
      WRITE(6,*) '      (2) Save existing properties to a file '
WRITE(6,*) '      (3) Composite Material Name '
      WRITE(6,*) '      (4) Fiber properties '
      WRITE(6,*) '      (5) Matrix properties '
      WRITE(6,*) '      (6) Return to main menu '
WRITE(6,*)
WRITE(6,*)
WRITE(6,*) ' Enter Selection: '
      READ(5,*) NP

      IF(NP.EQ.1) THEN

          WRITE(6,*)
          WRITE(6,*) ' Enter the name of the input file '
          READ(5,2001) INPROP
2001  FORMAT(A15)
          OPEN(UNIT=10,FILE=INPROP,STATUS='OLD')
          READ(10,2000) COMP
2000  FORMAT(A40)
          READ(10,2002) FIB,MATE,MATP,MATV
2002  FORMAT(4A2)
          READ(10,*)
          READ(10,*) NTEMF,AR,VF
          DO I=1,NTEMF
              READ(10,*) TEMPROPF(I),EF11(I),NUF12(I),EF22(I),NUF23(I),
>              GF12(I)
              READ(10,*) ALPHAF1(I),ALPHAF2(I)
              READ(10,*)
          END DO
          READ(10,*)
          READ(10,*) NTEMM,HD,DNOT,TEMPREF
          DO I=1,NTEMM
              READ(10,*) TEMPROPM(I),EM11(I),NUM12(I),EM22(I),NUM23(I),

```

```

>          GM12(I)
      READ(10,*) ALPHAM1(I),ALPHAM2(I),HPRAMPTS(I),YSPTS(I)
      READ(10,*) OSAT(I), F1(I),F3(I), Z0(I), BN(I)
      READ(10,*) Z0D(I),BND(I),BM1D(I),Z1D(I),R1D(I),A1D(I)
      READ(10,*) BM2D(I),Z2D(I),R2D(I),A2D(I),Z3D(I)
      READ(10,*)
    END DO
  READ(10,*)
    READ(10,*) NTEMI,SIC
  DO I=1,NTEMI
    READ(10,*) TEMPROPI(I),SIFN1(I),UIFN10A(I)
    READ(10,*) SIFT1(I),UIFT10A(I)
    READ(10,*)
  END DO
  CLOSE(UNIT=10)
  GO TO 2

ELSE IF(NP.EQ.2) THEN

      IF(COMP.EQ.'NO NAME') THEN
        WRITE(6,1015)
1015 FORMAT(/,' The composite has not yet been named. Please return',
      >' to the',/, ' Constituent Properties menu by pressing <return> or'
      >,' <enter>',/, ' and name the composite. ')
        READ(5,*)
        GO TO 2

      ELSE IF(FIB.EQ.'N') THEN
        WRITE(6,1016)
1016 FORMAT(/,' Fiber properties have not yet been specified. Please',
      >' return to',/, ' the Constituent Properties menu by pressing ',
      > '<return> or <enter>',/, ' and enter the fiber properites. ')
        READ(5,*)
        GO TO 2

      ELSE IF(MATE.EQ.'N') THEN
        WRITE(6,1017)
1017 FORMAT(/,' Matrix properties have not yet been specified. Please'
      >,' return to',/, ' the Constituent Properties menu by pressing ',
      > '<return> or <enter>',/, ' and enter the matrix properites. ')
        READ(5,*)
        GO TO 2
      END IF

      WRITE(6,*)
      WRITE(6,*) ' Enter the name of the file to be saved '
      READ(5,2001) OUTPROP
      OPEN(UNIT=11,FILE=OUTPROP,STATUS='UNKNOWN')
      WRITE(11,2000) COMP
      WRITE(11,2002) FIB,MATE,MATP,MATV
      WRITE(11,*)
      WRITE(11,*) NTEMF,AR,VF
      DO I=1,NTEMF
        WRITE(11,*) TEMPROPF(I),EF11(I),NUF12(I),EF22(I),NUF23(I),

```

```

>          GF12(I)
        WRITE(11,*) ALPHAF1(I),ALPHAF2(I)
        WRITE(11,*)
      END DO
      WRITE(11,*)
      WRITE(11,*) NTEMM,HD,DNOT,TEMPREF
      DO I=1,NTEMM
        WRITE(11,*) TEMPROM(I),EM11(I),NUM12(I),EM22(I),NUM23(I),
>          GM12(I)
        WRITE(11,*) ALPHAM1(I),ALPHAM2(I),HPRAMPTS(I),YSPTS(I)
        WRITE(11,*) OSAT(I), F1(I),F3(I), Z0(I), BN(I)
        WRITE(11,*) Z0D(I),BND(I),BM1D(I),Z1D(I),R1D(I),A1D(I)
        WRITE(11,*) BM2D(I),Z2D(I),R2D(I),A2D(I),Z3D(I)
        WRITE(11,*)
      END DO
      WRITE(11,*)
      WRITE(11,*) NTEMI,SIC
      DO I=1,NTEMI
        WRITE(11,*) TEMPROM(I),SIFN1(I),UIFN1OA(I)
        WRITE(11,*) SIFT1(I),UIFT1OA(I)
        WRITE(11,*)
      END DO
      CLOSE(UNIT=11)
      GO TO 2

ELSE IF(NP.EQ.3) THEN
  WRITE(6,*)
  WRITE(6,*) ' Composite Description: ',COMP
  WRITE(6,*)
  WRITE(6,*) ' Do you wish to change the description (y/n)?'
  READ(5,*) CHAR1

  IF(CHAR1.EQ.'y') THEN
    WRITE(6,1019)
1019 FORMAT(/,' Enter a description of the composite (40 characters or'
>,' less) :')
    READ(5,2000) COMP
  END IF
  GO TO 2

ELSE IF(NP.EQ.4) THEN

  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*) 'HD = ',HD,' AR = ',AR,' VF = ',VF
  WRITE(6,*) ' TEMPREF=',
> TEMPREF
  WRITE(6,*)
  WRITE(6,1001)

```

```

DO I=1,NTEMP
  WRITE(6,1000) TEMPPOPF(I),EF11(I),EF22(I),GF12(I),NUF12(I),
>    NUF23(I),ALPHA1(I),ALPHA2(I)
  END DO
1000  FORMAT(4F11.2,2F11.4,2E11.4)
1001  FORMAT(' TEMP (C)   E11 (GPA)  E22 (GPA)  G12 (GPA)   NU12   ',
>    NF23      ALPHA1      ALPHA2')

ELSE IF(NP.EQ.5) THEN
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)
  WRITE(6,*)'DNOT = ', DNOT
  WRITE(6,*)
  WRITE(6,1001)
  DO I=1,NTEMM
    WRITE(6,1000) TEMPPOPM(I),EM11(I),EM22(I),GM12(I),NUM12(I),
>    NUM23(I),ALPHAM1(I),ALPHAM2(I)
    END DO
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,1002)
1002  FORMAT(' H   Y       OSAT           F1           F3           Z0           ',
>    'N')
    DO I=1,NTEMM
      WRITE(6,1003)HPRAMPTS(I),YSPTS(I),OSAT(I),F1(I),F3(I),Z0(I),
>    BN(I)
    END DO
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*) '           Directional hardening constants'
    WRITE(6,1004)
1004  FORMAT('TEMP   Z0           N           M1           Z1           R1           ',
>    'A1 ')
    DO I=1,NTEMM
      WRITE(6,1005) TEMPPOPM(I),Z0D(I),BND(I),BM1D(I),Z1D(I),
>    R1D(I),A1D(I)
    END DO
    WRITE(6,*)
    WRITE(6,1006)
    DO I=1,NTEMM
      WRITE(6,1007) TEMPPOPM(I),BM2D(I),Z2D(I),R2D(I),A2D(I),
>    Z3D(I)
    END DO
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*)
    WRITE(6,*) '           Interfacial damage characteristics '
    WRITE(6,*)

```

```

        WRITE(6,*) 'SIC = ', SIC
        WRITE(6,*)
        WRITE(6,1008)
1008 FORMAT(' TEMP      SF1      UF1/a')
        DO I=1,NTEMI
            WRITE(6,1009) TEMPROPI(I),SIFN1(I),UIFN1OA(I)
            WRITE(6,1010) SIFT1(I),UIFT1OA(I)
        END DO
1003 FORMAT(2F3.0,2F11.1,F11.4,F11.1,F11.3)
1005 FORMAT(7F11.3)
1006 FORMAT('TEMP      M2      Z2      R2      A2      Z3')
1007 FORMAT(6F11.3)
1009 FORMAT(F10.2,F10.2,F10.5)
1010 FORMAT(T11,F10.2,F10.5)
        READ(5,*)

        ELSE IF(NP.EQ.6) THEN

            RETURN

        ELSE

            GO TO 2

        END IF

    END

C
C
C  * * * * *
C
C    The subroutine READDAT reads either a machine dependent
C    unformatted file of a solution set from an FTS calculation or
C    a global ASCII file that may be read by any machine.
C
C  * * * * *
C
C
C

SUBROUTINE READDAT

    CHARACTER FIB*1,MATE*1,MATP*1,MATV*1,TCHARP*1,TCHARV*1,
>COMP*40, ECHAR*1, LCHAR*1

    COMMON/TYPEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

    COMMON/MATERIAL/NUF12(40),NUF23(40),NUM12(40),NUM23(40),GM12(40),
>EM22(40),EM11(40),AR,VF,GF12(40),EF22(40),EF11(40),HPRAMPTS(40),
>YSPTS(40),TEMPROPF(40),TEMPROP(40),NTEMF,NTEMM,HD,ALPHAF1(40),
>ALPHAF2(40),ALPHAM1(40),ALPHAM2(40),OSAT(40),F1(40),F3(40),Z0(40),
>BN(40),DNOT,Z0D(40),BND(40),BM1D(40),Z1D(40),R1D(40),
>A1D(40),BM2D(40),Z2D(40),R2D(40),A2D(40),Z3D(40),TEMPREF,NTEMI,
>SIC,TEMPROPI(40),SIFN1(40),UIFN1OA(40),SIFT1(40),UIFT1OA(40)

```

```

COMMON/LOADING/NC1,NC2,NUMP,NUMCYC,DT(9999),TEMP(9999),NX(9999),
>NY(9999),NXY(9999),M(9999)

COMMON/ELASTPROP/SSF11(40),SSF22(40),SSF12(40),SSF23(40),
>SSF44(40),SSM11(40),SSM22(40),SSM12(40),SSM23(40),SSM44(40),
>A(6),B(6),C(6)

COMMON/TOLERANCE/TOLP1,TOLP2,TOLP3,TOLV1,TOLV2,TOLV3,TOLV4

COMMON/WORDS/FIB,MATE,MATP,MATV,TCHARP,TCHARV,COMP,ECHAR,LCHAR

REAL NX(9999),NY(9999),NXY(9999),EPSCOM(3),SIGPLY(6,3),EPSF(6,4),
>SF(6,4),EPSM(6,3,4),SM(6,3,4),UI(6,2,2),SI(6,2,2),NUF12(40),
>NUF23(40),NUM12(40),NUM23(40)

CHARACTER INDAT*15

```

2 REWIND 8

```

WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*) ' * * * * * '
WRITE(6,*) ' * Read Results from a File * '
WRITE(6,*) ' * * * * * '
WRITE(6,*)
WRITE(6,*)
WRITE(6,*) ' (1) Unformatted file '
WRITE(6,*) ' **N/A**(2) Global ASCII file '
WRITE(6,*) ' (3) Return to results menu '
WRITE(6,*)
WRITE(6,*)
WRITE(6,*) ' Enter Selection: '
READ(5,*) ND

IF(ND.EQ.1) THEN

WRITE(6,*) ' Enter name of the unformatted file to be read.'
READ(5,201) INDAT
201 FORMAT(A15)
OPEN(40,FILE=INDAT,STATUS='OLD',FORM='UNFORMATTED')
READ(40) NA,NUMPLY,NTEMF,NTEMM,NTEMI,AR,VF,HD,DNOT,SIC
> TEMPREF,NC1,NC2,NUMP,NUMCYC,FIB,MATE,MATP,MATV,TCHARP,
> TCHARV,COMP,ECHAR,LCHAR
READ(40) (ZPOS(I),I=1,NUMPLY+1),(ANGLE(J),J=1,NUMPLY)

```

```

      READ(40) (TEMPROPF(I),NUF12(I),NUF23(I),GF12(I),EF22(I),
>             EF11(I),ALPHAF1(I),ALPHAF2(I),I=1,NTEMF)
      READ(40) (TEMPROP(I),NUM12(I),NUM23(I),GM12(I),EM22(I),
>             EM11(I),ALPHAM1(I),ALPHAM2(I),HPRAMPTS(I),YSPTS(I),
>             OSAT(I),F1(I),F3(I),Z0(I),BN(I),Z0D(I),BND(I),
>             BM1D(I),Z1D(I),R1D(I),A1D(I),BM2D(I),Z2D(I),
>             R2D(I),A2D(I),Z3D(I),I=1,NTEMM)
      READ(40) (TEMPROPI(I),SIFN1(I),UIFN1OA(I),
>             SIFT1(I),UIFT1OA(I),I=1,NTEMI)

```

```

DO I=1,NUMP
  READ(40) DT(I),TEMP(I),NX(I),NY(I),NXY(I),M(I)
END DO

```

```

      DO N=1,NC1-1
        READ(40) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
> ((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

        WRITE(8) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
> ((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

      END DO

```

```

DO NCYC=1,NUMCYC

```

```

      DO N=NC1,NC2
        READ(40) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
> ((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

        WRITE(8) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
> ((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

      END DO

```

```

END DO

```

```

DO N=NC2+1,NUMP
  READ(40) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
> ((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

  WRITE(8) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
> ((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),

```

```

>I=1,NUMPLY)

      END DO

      ELSEIF (ND.EQ.2) THEN

      ELSEIF (ND.EQ.3) THEN

      RETURN

      END IF

      GO TO 2

      END
C
C
C * * * * *
C
C      The subroutine RESULT examines the most recent solution from the
C      present session or past solutions that have been saved to a formatted
C      file. The most recent solution may also be saved to a formatted or
C      unformatted file using this subroutine.
C
C * * * * *
C
C
      SUBROUTINE RESULT

      REAL NUF12(40), NUF23(40), NUM12(40), NUM23(40), NX(9999),
>NY(9999), NXY(9999)

      CHARACTER FIB*1, MATE*1, MATP*1, MATV*1, TCHARP*1, TCHARV*1,
>COMP*40, ECHAR*1, LCHAR*1

      COMMON/TPEMOD/NA, NUMPLY, ZPOS(7), ANGLE(6)

      COMMON/MATERIAL/NUF12(40), NUF23(40), NUM12(40), NUM23(40), GM12(40),
>EM22(40), EM11(40), AR, VF, GF12(40), EF22(40), EF11(40), HPRAMPTS(40),
>YSPTS(40), TEMPROPF(40), TEMPROPM(40), NTEMF, NTEMM, HD, ALPHAF1(40),
>ALPHAF2(40), ALPHAM1(40), ALPHAM2(40), OSAT(40), F1(40), F3(40), Z0(40),
>BN(40), DNOT, Z0D(40), BND(40), BM1D(40), Z1D(40), R1D(40),
>A1D(40), BM2D(40), Z2D(40), R2D(40), A2D(40), Z3D(40), TEMPREF, NTEMI,
>SIC, TEMPROPI(40), SIFN1(40), UIFN1OA(40), SIFT1(40), UIFT1OA(40)

      COMMON/LOADING/NC1, NC2, NUMP, NUMCYC, DT(9999), TEMP(9999), NX(9999),
>NY(9999), NXY(9999), M(9999)

```

```
COMMON/ELASTPROP/SSF11(40),SSF22(40),SSF12(40),SSF23(40),
>SSF44(40),SSM11(40),SSM22(40),SSM12(40),SSM23(40),SSM44(40),
>A(6),B(6),C(6)
```

```
COMMON/TOLERANCE/TOLP1,TOLP2,TOLP3,TOLV1,TOLV2,TOLV3,TOLV4
```

```
COMMON/WORDS/FIB,MATE,MATP,MATV,TCHARP,TCHARV,COMP,ECHAR,LCHAR
```

```
1  WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*) ' * * * * * '
   WRITE(6,*) ' *      Results of Solution      * '
   WRITE(6,*) ' * * * * * '
   WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*) '      (1) Read saved results file '
   WRITE(6,*) '      (2) Write existing data to a file '
   WRITE(6,*) '      (3) Create Tables '
   WRITE(6,*) '      (4) Return to main menu '
   WRITE(6,*)
   WRITE(6,*)
   WRITE(6,*) ' Enter Selection: '
   READ(5,*) NS

   IF(NS.EQ.1) THEN

       CALL READDAT

   ELSEIF(NS.EQ.2) THEN

       CALL WRITEDAT

   ELSEIF(NS.EQ.3) THEN

       IF(LCHAR.NE.'Y') THEN
           WRITE(6,*)
           WRITE(6,*) ' No dataset has been calculated.'
```

```

        WRITE(6,*)

    ELSE

        CALL TABLE

    END IF

    ELSEIF(NS.EQ.4) THEN

        RETURN

    END IF

    GO TO 1

END

C
C
C * * * * *
C
C    The SOLV subroutine determines the composite response to a given
C    thermomechanical load history for a LISOL model.
C
C * * * * *
C
C

SUBROUTINE SOLV

    REAL NUF12(40), NUF23(40), NUM12(40), NUM23(40)

    CHARACTER FIB*1,MATE*1,MATP*1,MATV*1,TCHARP*1,TCHARV*1,
    >COMP*40, ECHAR*1,LCHAR*1

    COMMON/TYPMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

    COMMON/MATERIAL/NUF12(40),NUF23(40),NUM12(40),NUM23(40),GM12(40),
    >EM22(40),EM11(40),AR,VF,GF12(40),EF22(40),EF11(40),HPRAMPTS(40),
    >YSPTS(40),TEMPROPF(40),TEMPROP(40),NTEMF,NTEMM,HD,ALPHAF1(40),
    >ALPHAF2(40),ALPHAM1(40),ALPHAM2(40),OSAT(40),F1(40),F3(40),Z0(40),
    >BN(40),DNOT,Z0D(40),BND(40),BM1D(40),Z1D(40),R1D(40),
    >A1D(40),BM2D(40),Z2D(40),R2D(40),A2D(40),Z3D(40),TEMPREF,NTEMI,
    >SIC,TEMPROPI(40),SIFN1(40),UIFN1OA(40),SIFT1(40),UIFT1OA(40)

    COMMON/LOADING/NC1,NC2,NUMP,NUMCYC,DT(9999),TEMP(9999),NX(9999),
    >NY(9999),NXY(9999),M(9999)

    COMMON/ELASTPROP/SSF11(40),SSF22(40),SSF12(40),SSF23(40),
    >SSF44(40),SSM11(40),SSM22(40),SSM12(40),SSM23(40),SSM44(40),
    >A(6),B(6),C(6)

```

```

COMMON/TOLERANCE/TOLP1,TOLP2,TOLP3,TOLV1,TOLV2,TOLV3,TOLV4

REAL BIGT(50,20),NX(9999),NY(9999),NXY(9999),K2(6,3),LVEC0(50)

COMMON/VISCOPLAST/H(50,50),HINV(50,50),T(6,3,3),TINT(6,3,3),
>TH(50,20),BIGP(50,50),N,NCYC,TEM0,TIM0,PZI(6,3),LVEC0(50),
>EPSMP0(6,3,4),SM0(6,3,4),UIF0(6,2,2),SIO(6,2,2),UI0(6,2,2),
>K2(6,3),BETA0(6,3,4),OMEG0(6,3,4),EPSTHM1,EPSTHM2,EPSTHF1,
>EPSTHF2,NDUMP

COMMON/WORDS/FIB,MATE,MATP,MATV,TCHARP,TCHARV,COMP,ECHAR,LCHAR

C
C
C Find the dimensions of the representative volume element
C
C
      DO I=1,NUMPLY
        A(I) = (ZPOS(I+1)-ZPOS(I))*SQRT(VF/AR)
        C(I) = ZPOS(I+1)-ZPOS(I)-A(I)
        B(I) = (A(I)+C(I))/AR-A(I)
      END DO

C
C
C Use property relations to find the fiber and matrix compliance
C matrices. Also, converts compliance to units of 1/MPa.
C
C
      CONVER = 10.**(-3)

      DO N=1,NTEMF

        SSF11(N) = 1/EF11(N)
        SSF22(N) = 1/EF22(N)
        SSF12(N) = -NUF12(N)/EF11(N)
        SSF23(N) = -NUF23(N)/EF22(N)
        SSF44(N) = 1/GF12(N)

        SSF11(N) = SSF11(N)*CONVER
        SSF22(N) = SSF22(N)*CONVER
        SSF12(N) = SSF12(N)*CONVER
        SSF23(N) = SSF23(N)*CONVER
        SSF44(N) = SSF44(N)*CONVER

      END DO

      DO N=1,NTEMM

        SSM11(N) = 1/EM11(N)
        SSM22(N) = 1/EM22(N)

```

```

SSM12(N) = -NUM12(N)/EM11(N)
SSM23(N) = -NUM23(N)/EM22(N)
SSM44(N) = 1/GM12(N)

```

```

SSM11(N) = SSM11(N)*CONVER
SSM22(N) = SSM22(N)*CONVER
SSM12(N) = SSM12(N)*CONVER
SSM23(N) = SSM23(N)*CONVER
SSM44(N) = SSM44(N)*CONVER

```

```

END DO

```

```

C
C
C

```

```

Set the zero stress thermal strains

```

```

AM1 = TDEP(NTEMM,TEMPROP,ALPHAM1,TEMP(1))
AF1 = TDEP(NTEMF,TEMPROPF,ALPHA1,TEMP(1))
AM2 = TDEP(NTEMM,TEMPROP,ALPHAM2,TEMP(1))
AF2 = TDEP(NTEMF,TEMPROPF,ALPHA2,TEMP(1))
EPSTHM1 = AM1*(TEMP(1)-TEMPREF)
EPSTHF1 = AF1*(TEMP(1)-TEMPREF)
EPSTHM2 = AM2*(TEMP(1)-TEMPREF)
EPSTHF2 = AF2*(TEMP(1)-TEMPREF)

```

```

C
C
C

```

```

ENTER THE [H] AND [T] MATRICES AND DETERMINE THE [TH] MATRIX

```

```

DO N=1,NUMPLY
H(1,3*N-2)=ZPOS(N+1)-ZPOS(N)
H(1,3*N-1)=0.
H(1,3*N)=0.
H(2,3*N-2)=0.
H(2,3*N-1)=ZPOS(N+1)-ZPOS(N)
H(2,3*N)=0.
H(3,3*N-2)=0.
H(3,3*N-1)=0.
H(3,3*N)=ZPOS(N+1)-ZPOS(N)
H(4,3*N-2)=0.5*(ZPOS(N+1)**2.-ZPOS(N)**2.)
H(4,3*N-1)=0.
H(4,3*N)=0.
H(5,3*N-2)=0.
H(5,3*N-1)=0.5*(ZPOS(N+1)**2.-ZPOS(N)**2.)
H(5,3*N)=0.
H(6,3*N-2)=0.
H(6,3*N-1)=0.
H(6,3*N)=0.5*(ZPOS(N+1)**2.-ZPOS(N)**2.)
END DO
DO N=7,3*NUMPLY
DO I=1,3*NUMPLY
H(N,I)=0.
END DO

```

```

END DO
DO N=7,3*NUMPLY
  H(N,N)=1.
END DO

DO N=1,NUMPLY
  CS=COSD(ANGLE(N))
  SN=SIND(ANGLE(N))
  T(N,1,1)=CS**2.
  T(N,1,2)=SN**2.
  T(N,1,3)=2.*SN*CS
  T(N,2,1)=T(N,1,2)
  T(N,2,2)=T(N,1,1)
  T(N,2,3)=-T(N,1,3)
  T(N,3,1)=-SN*CS
  T(N,3,2)=-T(N,3,1)
  T(N,3,3)=CS**2.-SN**2.
END DO

```

```

C
C  INVERSE-TRANSPPOSE OF THE [T] MATRIX
C

```

```

DO N=1,NUMPLY
  TINT(N,1,1)=T(N,1,1)
  TINT(N,1,2)=T(N,1,2)
  TINT(N,1,3)=T(N,1,3)/2.
  TINT(N,2,1)=T(N,2,1)
  TINT(N,2,2)=T(N,2,2)
  TINT(N,2,3)=T(N,2,3)/2.
  TINT(N,3,1)=T(N,3,1)*2.
  TINT(N,3,2)=T(N,3,2)*2.
  TINT(N,3,3)=T(N,3,3)
END DO

NPLY3=3*NUMPLY

CALL INVERSE(NPLY3,H,HINV)

DO N=1,8*NUMPLY
  DO I=1,3*NUMPLY
    BIGT(N,I)=0.
  END DO
END DO

DO N=1,NUMPLY
  BIGT(8*N-2,3*N-2)=T(N,1,1)
  BIGT(8*N-2,3*N-1)=T(N,1,2)
  BIGT(8*N-2,3*N)=T(N,1,3)
  BIGT(8*N-1,3*N-2)=T(N,2,1)
  BIGT(8*N-1,3*N-1)=T(N,2,2)
  BIGT(8*N-1,3*N)=T(N,2,3)
  BIGT(8*N,3*N-2)=T(N,3,1)
  BIGT(8*N,3*N-1)=T(N,3,2)

```

```

        BIGT(8*N,3*N)=T(N,3,3)
    END DO

C
C  DETERMINE THE [TH] MATRIX BY MATRIX MULTIPLICATION
C

    DO N=1,8*NUMPLY
        DO I=1,3*NUMPLY
            TH(N,I)=0.
            DO J=1,3*NUMPLY
                TH(N,I)=TH(N,I)+BIGT(N,J)*HINV(J,I)
            END DO
        END DO
    END DO

C
C  ENTER COMPONENTS OF [TH] THAT APPEAR IN THE [BIGP] MATRIX
C

    DO I=5*NUMPLY+7,8*NUMPLY
        DO N=1,8*NUMPLY
            BIGP(N,I)=-TH(N,I-5*NUMPLY)
        END DO
    END DO

C
C  Initialize the components of the BIGP matrix containing the PB
C  matrices
C

    DO I=1,5*NUMPLY
        DO J=1,8*NUMPLY
            BIGP(J,I+6)=0.0
        END DO
    END DO

C
C  Perform calculations.
C
C

    IF(NA.EQ.1) THEN

C
C
C      Linear Elastic Case.
C
C

    ELSEIF(NA.EQ.2) THEN

```

```

C
C
C   Elastic-Plastic case
C
C
C
C
C   ELSEIF(NA.EQ.3) THEN
C
C
C   Viscoplastic case
C
C
C   Set initial values for some of the variables for the first
C   increment.
C
C
C   TEM0=TEMP(1)
C   TIM0=0.
C   IF(DT(1).EQ.0.0) DT(1) = 0.000001
C       NCYC=0
C       NDUMP=0
C       DO I=1,NUMPLY
C           DO J=1,3
C               PZI(I,J)=0.0
C           END DO
C       END DO
C       DO I=1,8*NUMPLY
C           LVEC0(I)=0.
C       END DO
C       DO I=1,NUMPLY
C           DO J=1,3
C               DO K=1,4
C                   EPSMP0(I,J,K)=0.
C                   BETA0(I,J,K)=0.
C                   OMEG0(I,J,K)=0.
C                   SM0(I,J,K)=0.
C               END DO
C               K2(I,J)=0.0
C           END DO
C       DO K=1,2
C           UIF0(I,K,1)=0.0000001
C           UIF0(I,K,2)=0.0000001
C           DO J=1,2
C               SI0(I,K,J)=0.0
C               UI0(I,K,J)=0.0
C           END DO
C       END DO
C   END DO
C
C   DO N=1,NC1-1
C
C       CALL VISC

```

```

        IF (NDUMP.EQ.1) RETURN

        END DO

C
C
C   Perform cyclic loading sequence
C
C
        DO NCYC=1,NUMCYC
            WRITE(6,*) 'CYCLE ',NCYC
            DO N=NC1,NC2

                CALL VISC
                IF (NDUMP.EQ.1) RETURN

            END DO

        END DO

C
C
C   Perform the post-load sequence
C
C
        DO N=NC2+1,NUMP

            CALL VISC
            IF (NDUMP.EQ.1) RETURN

        END DO

        END IF

        RETURN

        END

C
C
C * * * * *
C
C   The subroutine STRESS calculates the stresses for a LISOL
C   model given a plastic strain and the solution to the matrix equation.
C
C * * * * *

```

C
C

```
SUBROUTINE STRESS (NUMPLY, EPSMP, XVEC, TINT, A, B, C, SM, SF, SI, SPLY,  
> SSIT, S44F, S11M, S12M, S44M)
```

```
REAL EPSMP(6,3,4), XVEC(50), TINT(6,3,3), A(6), B(6), C(6),  
> SM(6,3,4), SF(6,4), SI(6,2,2), SPLY(6,3), SSIT(6,2), GAM(6)
```

```
INTEGER NUMPLY
```

```
R1 = S11M/S12M
```

```
DO I=1, NUMPLY
```

```
  I5=5*I
```

```
    SF(I,1)=XVEC(I5+2)
```

```
    SF(I,2)=XVEC(I5+3)
```

```
    SM(I,1,1)=XVEC(I5+4)
```

```
    SM(I,1,3)=XVEC(I5+5)
```

```
    SM(I,2,1)=XVEC(I5+6)
```

```
    SF(I,3)=-B(I)/A(I)*SM(I,1,3)
```

```
    SM(I,1,2)=SF(I,2)
```

```
    SM(I,2,2)=SM(I,1,2)+R1*SM(I,1,1)-R1*SM(I,2,1)+EPSMP(I,1,1)  
>    /S12M - EPSMP(I,2,1)/S12M
```

```
    SI(I,1,1)=SF(I,2)
```

```
    SM(I,2,3)=SM(I,1,3)
```

```
    SM(I,3,2)=SM(I,2,2)
```

```
    SM(I,3,3)=SF(I,3)
```

```
    SI(I,2,1)=SF(I,3)
```

```
    SM(I,3,1)=SM(I,2,1)+SM(I,2,3)/R1-SM(I,3,3)/R1+  
>    EPSMP(I,2,1)/S11M-EPSMP(I,3,1)/S11M
```

```
    GAM(I)=TINT(I,3,1)*XVEC(1)+TINT(I,3,2)*XVEC(2)+  
>    TINT(I,3,3)*XVEC(3)
```

```
T4=C(I)/A(I)
```

```
AA=(1+T4)*(S44F+SSIT(I,1))
```

```
BB=S44M+T4*(S44F+SSIT(I,1))
```

```
D1=S44M*(A(I)*AA/BB+B(I))
```

```
SPLY(I,3)=(A(I)+B(I))*GAM(I)/D1 - B(I)*EPSMP(I,1,4)/(D1*(1+T4))  
> - B(I)*T4*EPSMP(I,2,4)/(D1*(1+T4)) - A(I)*(1-S44M/BB)*  
> EPSMP(I,3,4)/D1
```

```
SM(I,3,4)=AA/BB*SPLY(I,3)-EPSMP(I,3,4)/BB
```

```
SF(I,4)=(1+T4)*SPLY(I,3) - T4*SM(I,3,4)
```

```
SM(I,2,4)=SPLY(I,3)+(EPSMP(I,1,4)-EPSMP(I,2,4))/(S44M*(1+T4))
```

```
SM(I,1,4)=(1+T4)*SPLY(I,3) - T4*SM(I,2,4)
```

```
SI(I,1,2)=SF(I,4)
```

```
SI(I,2,2)=0.
```

END DO

RETURN

END

C
C * * * * *
C
C The subroutine TABLE creates tables of results from an LISOL
C solution.
C
C * * * * *
C
C

SUBROUTINE TABLE

COMMON/TPEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)

COMMON/LOADING/NC1,NC2,NUMP,NUMCYC,DT(9999),TEMP(9999),NX(9999),
>NY(9999),NXY(9999),M(9999)

REAL SC(3),EPSCOM(3),SIGPLY(6,3),SF(6,4),EPSF(6,4),SM(6,3,4),
>EPSM(6,3,4),SI(6,2,2),UI(6,2,2),TBL(10),NX(9999),NY(9999),
>NXY(9999)

INTEGER ITYP(10),ICOM(10)

CHARACTER CHAR1(100)*18,CHAR2(20)*18,OUTFIL*20

REWIND 8

WRITE(6,*)
WRITE(6,*) ' How many columns in new table (<=7)?'
READ(5,*) NCOL

DO LL=1,NCOL
ICOM(LL)=0
END DO

NUMM=3

DO I=1,NCOL

WRITE(6,1001)
1001 FORMAT(//////, ' (1) Time',/,
>' (2) Temperature',/,
>' Composite: (3) Stress (4) Strain')

```

      DO IP=1,NUMPLY
        WRITE(6,1011) IP,14*IP-9,14*IP-8
1011  FORMAT('          Ply : ',I1,5X,'(',I2,') Stress      (',
          > I2,') Strain')

        WRITE(6,1002) 14*IP-7,14*IP-6
1002  FORMAT('      Fiber Region : ',5X,'(',I2,') Stress      (',I2,
          >') Strain')

      DO IIM=1,NUMM
        WRITE(6,1003) IIM,14*IP+2*IIM-7,14*IP+2*IIM-6
1003  FORMAT('      Matrix Region',I2,': ',6X,'(',I2,') Stress      (',
          >I2,') Strain')
      END DO

      DO III = 1,NUMM-1
        WRITE(6,1010) III,14*IP+2*III-1,14*IP+2*III
1010  FORMAT(' Interface Region',I2,': ',6X,'(',I2,') Stress      (',
          >I2,') Displacement')
      END DO
      WRITE(6,*)

    END DO

    WRITE(6,1004) I
1004  FORMAT('/', ' Select desired quantity for column ',I2)

    READ(5,*) ITYP(I)

    IF(ITYP(I).GT.2) THEN
      WRITE(6,1005)
1005  FORMAT('/', '      Component:  (1) 1-1, X    (2) 2-2, Y    ',
          >' (3) 3-3, XY    (4) 1-2',/,
          >'      Select stress/strain component:')
      READ(5,*) ICOM(I)
    END IF

    END DO

    WRITE(6,*)
    WRITE(6,1008)
1008  FORMAT(' Write table to  (1) screen  (2) file  (3) both',
          >'      (4) Cancel')
    READ(5,*) IOUT

    IF(IOUT.EQ.4) RETURN

    IF(IOUT.NE.1) THEN
      WRITE(6,*) 'Enter the name of the file'
      READ(5,2222) OUTFIL
2222  FORMAT(A20)
      OPEN(20,FILE=OUTFIL,STATUS='NEW')
    END IF

```

```

CHAR1(1) = 'Time (sec)'
CHAR1(2) = 'Temperature (C)'
CHAR1(3) = 'Composite Stress'
CHAR1(4) = 'Composite Strain'

CHAR1(5) = 'Ply 1 Stress'
CHAR1(6) = 'Ply 1 Strain'
CHAR1(19) = 'Ply 2 Stress'
CHAR1(20) = 'Ply 2 Strain'
CHAR1(33) = 'Ply 3 Stress'
CHAR1(34) = 'Ply 3 Strain'
CHAR1(47) = 'Ply 4 Stress'
CHAR1(48) = 'Ply 4 Strain'
CHAR1(61) = 'Ply 5 Stress'
CHAR1(62) = 'Ply 5 Strain'
CHAR1(75) = 'Ply 6 Stress'
CHAR1(76) = 'Ply 6 Strain'

DO IP=1,NUMPLY

CHAR1(14*IP-7) = 'Fib Reg Stress'
CHAR1(14*IP-6) = 'Fib Reg Strain'
CHAR1(14*IP-5) = 'Mat Reg 1 Stress'
CHAR1(14*IP-4) = 'Mat Reg 1 Strain'
CHAR1(14*IP-3) = 'Mat Reg 2 Stress'
CHAR1(14*IP-2) = 'Mat Reg 2 Strain'
CHAR1(14*IP-1) = 'Mat Reg 3 Stress'
CHAR1(14*IP) = 'Mat Reg 3 Strain'
CHAR1(14*IP+1) = 'Int Reg 1 Stress'
CHAR1(14*IP+2) = 'Int Reg 1 Dsplmt'
CHAR1(14*IP+3) = 'Int Reg 2 Stress'
CHAR1(14*IP+4) = 'Int Reg 2 Dsplmt'

END DO

CHAR2(0) = ''
CHAR2(1) = 'Component 1-1/XX'
CHAR2(2) = 'Component 2-2/YY'
CHAR2(3) = 'Component 3-3/XY'
CHAR2(4) = 'Component 1-2'

IF(IOUT.NE.2) THEN
  WRITE(6,1006) (CHAR1(ITYP(I)),I=1,NCOL)
1006 FORMAT(7(2X,A16))
1009 FORMAT(7(2X,A16),/)
  WRITE(6,1009) (CHAR2(ICOM(I)),I=1,NCOL)
END IF

IF(IOUT.NE.1) THEN
  WRITE(20,1006) (CHAR1(ITYP(I)),I=1,NCOL)
  WRITE(20,1009) (CHAR2(ICOM(I)),I=1,NCOL)
END IF

```

```

DO N=1,NC1-1
  READ(8) TIM1,TEM1,(EPSCOM(K),K=1,3),(SC(K),K=1,3),
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

DO I=1,NCOL
  IF(ITYP(I).EQ.1) TBL(I)=TIM1
  IF(ITYP(I).EQ.2) TBL(I)=TEM1
  IF(ITYP(I).EQ.3) TBL(I)=SC(ICOM(I))
  IF(ITYP(I).EQ.4) TBL(I)=EPSCOM(ICOM(I))

DO IP=1,NUMPLY

  IF(ITYP(I).EQ.14*IP-9) TBL(I)=SIGPLY(IP,ICOM(I))
  IF(ITYP(I).EQ.14*IP-8) TBL(I)=EPSCOM(ICOM(I))
  IF(ITYP(I).EQ.14*IP-7) TBL(I)=SF(IP,ICOM(I))
  IF(ITYP(I).EQ.14*IP-6) TBL(I)=EPSF(IP,ICOM(I))

DO IIM=1,NUMM
  IF(ITYP(I).EQ.14*IP+2*IIM-7) TBL(I)=SM(IP,IIM,ICOM(I))
  IF(ITYP(I).EQ.14*IP+2*IIM-6) TBL(I)=EPSM(IP,IIM,
>      ICOM(I))
END DO

DO III = 1,NUMM-1
  IF(ITYP(I).EQ.14*IP+2*III-1) TBL(I)=SI(IP,III,ICOM(I))
  IF(ITYP(I).EQ.14*IP+2*III) TBL(I)=UI(IP,III,ICOM(I))
END DO

END DO

END DO

IF(IOUT.NE.2) THEN
  WRITE(6,1007) (TBL(I),I=1,NCOL)
1007 FORMAT(7F18.9)
END IF

IF(IOUT.NE.1) THEN
  WRITE(20,1007) (TBL(I),I=1,NCOL)
END IF

END DO

DO NCYC=1,NUMCYC

DO N=NC1,NC2
  READ(8) TIM1,TEM1,(EPSCOM(K),K=1,3),(SC(K),K=1,3),
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

DO I=1,NCOL
  IF(ITYP(I).EQ.1) TBL(I)=TIM1
  IF(ITYP(I).EQ.2) TBL(I)=TEM1
  IF(ITYP(I).EQ.3) TBL(I)=SC(ICOM(I))
  IF(ITYP(I).EQ.4) TBL(I)=EPSCOM(ICOM(I))

DO IP=1,NUMPLY

  IF(ITYP(I).EQ.14*IP-9) TBL(I)=SIGPLY(IP,ICOM(I))
  IF(ITYP(I).EQ.14*IP-8) TBL(I)=EPSCOM(ICOM(I))
  IF(ITYP(I).EQ.14*IP-7) TBL(I)=SF(IP,ICOM(I))
  IF(ITYP(I).EQ.14*IP-6) TBL(I)=EPSF(IP,ICOM(I))

DO IIM=1,NUMM
  IF(ITYP(I).EQ.14*IP+2*IIM-7) TBL(I)=SM(IP,IIM,ICOM(I))
  IF(ITYP(I).EQ.14*IP+2*IIM-6) TBL(I)=EPSM(IP,IIM,
>      ICOM(I))
END DO

DO III = 1,NUMM-1
  IF(ITYP(I).EQ.14*IP+2*III-1) TBL(I)=SI(IP,III,ICOM(I))
  IF(ITYP(I).EQ.14*IP+2*III) TBL(I)=UI(IP,III,ICOM(I))
END DO

END DO

END DO

IF(IOUT.NE.2) THEN
  WRITE(6,1007) (TBL(I),I=1,NCOL)
END IF

IF(IOUT.NE.1) THEN
  WRITE(20,1007) (TBL(I),I=1,NCOL)
END IF

END DO

END DO

DO N=NC2+1,NUMP
  READ(8) TIM1,TEM1,(EPSCOM(K),K=1,3),(SC(K),K=1,3),
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

DO I=1,NCOL
  IF(ITYP(I).EQ.1) TBL(I)=TIM1
  IF(ITYP(I).EQ.2) TBL(I)=TEM1
  IF(ITYP(I).EQ.3) TBL(I)=SC(ICOM(I))
  IF(ITYP(I).EQ.4) TBL(I)=EPSCOM(ICOM(I))

DO IP=1,NUMPLY

```

```

      IF(ITYP(I).EQ.14*IP-9) TBL(I)=SIGPLY(IP,ICOM(I))
      IF(ITYP(I).EQ.14*IP-8) TBL(I)=EPSCOM(ICOM(I))
      IF(ITYP(I).EQ.14*IP-7) TBL(I)=SF(IP,ICOM(I))
      IF(ITYP(I).EQ.14*IP-6) TBL(I)=EPSF(IP,ICOM(I))

      DO IIM=1,NUMM
        IF(ITYP(I).EQ.14*IP+2*IIM-7) TBL(I)=SM(IP,IIM,ICOM(I))
        IF(ITYP(I).EQ.14*IP+2*IIM-6) TBL(I)=EPSM(IP,IIM,
>          ICOM(I))
      END DO

      DO III = 1,NUMM-1
        IF(ITYP(I).EQ.14*IP+2*III-1) TBL(I)=SI(IP,III,ICOM(I))
        IF(ITYP(I).EQ.14*IP+2*III) TBL(I)=UI(IP,III,ICOM(I))
      END DO

      END DO

      END DO

      IF(IOUT.NE.2) THEN
        WRITE(6,1007) (TBL(I),I=1,NCOL)
      END IF

      IF(IOUT.NE.1) THEN
        WRITE(20,1007) (TBL(I),I=1,NCOL)
      END IF

      END DO

      IF(IOUT.NE.2) THEN
        READ(5,*)
      END IF

      RETURN

      END

C
C * * * * *
C
C This function calculates the temperature dependent properties
C
C * * * * *
C

      FUNCTION TDEP(NTEM,TEMPROP,YSPTS,TEMP)

      REAL TEMPROP(40),YSPTS(40)

      IF(NTEM.EQ.1) THEN

        TDEP = YSPTS(NTEM)

```

```

ELSEIF (TEMP.LE.TEMPROP(1)) THEN

    TDEP = YSPTS(1) + (TEMP-TEMPROP(1))/(TEMPROP(2)-TEMPROP(1))*
> (YSPTS(2)-YSPTS(1))

ELSEIF (TEMP.GE.TEMPROP(NTEM)) THEN

    TDEP = YSPTS(NTEM-1) + (TEMP-TEMPROP(NTEM-1))/
> (TEMPROP(NTEM)-TEMPROP(NTEM-1))*(YSPTS(NTEM)-YSPTS(NTEM-1))

ELSE

    DO I=2,NTEM

        IF (TEMP.LE.TEMPROP(I)) THEN
            TDEP = YSPTS(I-1) + (TEMP-TEMPROP(I-1))/
> (TEMPROP(I)-TEMPROP(I-1))*(YSPTS(I)-YSPTS(I-1))
            GO TO 1
        END IF

    END DO

END IF

1 RETURN

END

C
C
C * * * * *
C
C * * * * *
C
C    The subroutine VISC calculates and writes the stresses and
C    strains of a single load point for an elastic-viscoplastic
C    model.
C
C * * * * *
C
C

SUBROUTINE VISC

    REAL NUF12(40), NUF23(40), NUM12(40), NUM23(40), NX(9999), NY(9999),
> NXY(9999)

    COMMON/TPEMOD/NA, NUMPLY, ZPOS(7), ANGLE(6)

    COMMON/MATERIAL/NUF12(40), NUF23(40), NUM12(40), NUM23(40), GM12(40),
> EM22(40), EM11(40), AR, VF, GF12(40), EF22(40), EF11(40), HPRAMPTS(40),
> YSPTS(40), TEMPROPF(40), TEMPROPM(40), NTEMF, NTEMM, HD, ALPHAF1(40),
> ALPHAF2(40), ALPHAM1(40), ALPHAM2(40), OSAT(40), F1(40), F3(40), Z0(40),

```

```

>BN(40),DNOT,Z0D(40),BND(40),BM1D(40),Z1D(40),R1D(40),
>A1D(40),BM2D(40),Z2D(40),R2D(40),A2D(40),Z3D(40),TEMPREF,NTEMI,
>SIC,TEMPROPI(40),SIFN1(40),UIFN10A(40),SIFT1(40),UIFT10A(40)

```

```

COMMON/LOADING/NC1,NC2,NUMP,NUMCYC,DT(9999),TEMP(9999),NX(9999),
>NY(9999),NXY(9999),M(9999)

```

```

COMMON/ELASTPROP/SSF11(40),SSF22(40),SSF12(40),SSF23(40),
>SSF44(40),SSM11(40),SSM22(40),SSM12(40),SSM23(40),SSM44(40),
>A(6),B(6),C(6)

```

```

COMMON/TOLERANCE/TOLP1,TOLP2,TOLP3,TOLV1,TOLV2,TOLV3,TOLV4

```

```

REAL LVEC1(50),LVEC0(50),DLVEC(50),ZIO(6,3),VI(50),VEPSMP0(6,12),
>SMP0(6,3,4),SIF(6,2,2),SSI(6,2),SSIT(6,2),UIF(6,2,2),PPVEC(50),
>FVEC(50),EPSEDOT(6,3,4),EPSEFFDE(6,3),DUIF(6,2,2),
>RT(6,3),K2(6,3),ZI(6,3),BETA(6,3,4),OMEG(6,3,4),SM(6,3,4),
>SMP(6,3,4),DSMP(6,3,4),SI(6,2,2),EPSEFFDP(6,3),EPSCALDP(6,3),
>DEPSDP(6,3),DEPSMP(6,3,4),EPSMP(6,3,4),VEPSMP(6,12),UI(6,2,2),
>EPSF(6,4),EPSM(6,3,4),SF(6,4),SPLY(6,3),SIGPLY(6,3),EPSC(3),
>SIFN(3),UIFN(3),SIFT(3),UIFT(3),XVEC(50)

```

```

COMMON/VISCOPLAST/H(50,50),HINV(50,50),T(6,3,3),TINT(6,3,3),
>TH(50,20),BIGP(50,50),N,NCYC,TEM0,TIM0,PZI(6,3),LVEC0(50),
>EPSMP0(6,3,4),SM0(6,3,4),UIF0(6,2,2),SIO(6,2,2),UI0(6,2,2),
>K2(6,3),BETA0(6,3,4),OMEG0(6,3,4),EPSTHM1,EPSTHM2,EPSTHF1,
>EPSTHF2,NDUMP

```

```

COMMON/PMATRIX/PP(6,8,12),S11F,S12F,S22F,S23F,S44F,S11M,S12M,
>S22M,S23M,S44M,SSI(6,2),SSIT(6,2)

```

```

CHARACTER CHAR1*1

```

```

FACT1 = 1.

```

C
C
C
C
C
C

```

load vector for desired point

```

```

IF(NCYC.LT.2) WRITE(6,*)'POINT ',N
    AM1 = TDEP(NTEMM,TEMPROP,ALPHAM1,TEMP(N))
    AF1 = TDEP(NTEMF,TEMPROPF,ALPHA1,TEMP(N))
    AM2 = TDEP(NTEMM,TEMPROP,ALPHAM2,TEMP(N))
    AF2 = TDEP(NTEMF,TEMPROPF,ALPHA2,TEMP(N))

DO I=1,NUMPLY
    LVEC1(8*I-7) = AF1*(TEMP(N)-TEMPREF)-EPSTHF1
    LVEC1(8*I-6) = -(A(I)*AF2+B(I)*AM2)*(TEMP(N)-TEMPREF)+
>    A(I)*EPSTHF2+B(I)*EPSTHM2
    LVEC1(8*I-5) = (AM1-AF1)*(TEMP(N)-TEMPREF)-(EPSTHM1-EPSTHF1)
    LVEC1(8*I-4) = A(I)*((AM2-AF2)*(TEMP(N)-TEMPREF) -
>    (EPSTHM2-EPSTHF2))
    LVEC1(8*I-3) = A(I)*((AM2-AF2)*(TEMP(N)-TEMPREF) -
>    (EPSTHM2-EPSTHF2))

```

```

      LVEC1(8*I-2) = TH(8*I-2,1)*NX(N)+TH(8*I-2,2)*NY(N) +
>      TH(8*I-2,3)*NXY(N)
      LVEC1(8*I-1) = TH(8*I-1,1)*NX(N)+TH(8*I-1,2)*NY(N) +
>      TH(8*I-1,3)*NXY(N)
      LVEC1(8*I) = TH(8*I,1)*NX(N)+TH(8*I,2)*NY(N) +
>      TH(8*I,3)*NXY(N)
END DO

```

```

NR=3

```

```

Z1D0=TDEP(NTEMM,TEMPROP, Z1D,TEM0)
Z2D0=TDEP(NTEMM,TEMPROP, Z2D,TEM0)
DO I=1,NUMPLY
  DO J=1,NR
    ZI0(I,J)=PZI(I,J)*(Z1D0-Z2D0)+Z2D0
  END DO
END DO

```

C Initialize interface load vector

```

MATSIZ=8*NUMPLY
DO I=1,MATSIZ
  VI(I)=0.
END DO

```

```

DTEM = (TEMP(N)-TEM0)/M(N)
DTIM = DT(N)/M(N)

```

```

MM=1
DO J=1,MATSIZ
  DLVEC(J)=(LVEC1(J)-LVEC0(J))/M(N)
END DO

```

```

DO I=1,NUMPLY
  DO J=1,NR
    VEPSMP0(I,J)=EPSMP0(I,J,4)
    VEPSMP0(I,3*J+1)=EPSMP0(I,J,1)
    VEPSMP0(I,3*J+2)=EPSMP0(I,J,2)
    VEPSMP0(I,3*J+3)=EPSMP0(I,J,3)
  END DO
END DO

```

```

DO I=1,NUMPLY
  DO J=1,NR
    HYDRO = (SM0(I,J,1)+SM0(I,J,2)+SM0(I,J,3))/3.
    SMP0(I,J,1)=SM0(I,J,1)-HYDRO
    SMP0(I,J,2)=SM0(I,J,2)-HYDRO
    SMP0(I,J,3)=SM0(I,J,3)-HYDRO
    SMP0(I,J,4)=SM0(I,J,4)
  END DO
END DO

```

C

```

C
C   Enter the matrix BIGP and Pplast
C
C
1   TEM1=TEM0+DTEM

C   First determine temperature dependent properties required to find
C   the BIGP matrix

S11F = TDEP(NTEMF,TEMPROPF,SSF11,TEM1)
S12F = TDEP(NTEMF,TEMPROPF,SSF12,TEM1)
S22F = TDEP(NTEMF,TEMPROPF,SSF22,TEM1)
S23F = TDEP(NTEMF,TEMPROPF,SSF23,TEM1)
S44F = TDEP(NTEMF,TEMPROPF,SSF44,TEM1)

S11M = TDEP(NTEMM,TEMPROP,SSM11,TEM1)
S12M = TDEP(NTEMM,TEMPROP,SSM12,TEM1)
S22M = TDEP(NTEMM,TEMPROP,SSM22,TEM1)
S23M = TDEP(NTEMM,TEMPROP,SSM23,TEM1)
S44M = TDEP(NTEMM,TEMPROP,SSM44,TEM1)

SIFN(1) = TDEP(NTEMI,TEMPROPI,SIFN1,TEM1)
UIFN(1) = 0.
SIFN(2) = 0.
UIFN(2) = TDEP(NTEMI,TEMPROPI,UIFN10A,TEM1)
SIFN(3) = 0.
UIFN(3) = UIFN(2)+0.05

SIFT(1) = TDEP(NTEMI,TEMPROPI,SIFT1,TEM1)
UIFT(1) = 0.
SIFT(2) = 0.5
UIFT(2) = TDEP(NTEMI,TEMPROPI,UIFT10A,TEM1)
SIFT(3) = 0.5
UIFT(3) = UIFT(2)+0.05

C   Determine stress point where interfacial failure begins.

DO I=1,NUMPLY
  DO K=1,2
    SIF(I,K,1) = TDEP(3,UIFN,SIFN,UIF0(I,K,1))
    SIF(I,K,2) = TDEP(3,UIFT,SIFT,UIF0(I,K,2))
    IF(SI0(I,K,1).LE.SIC.OR.UI0(I,K,1).LT.0.) THEN
      SSI(I,K) = 0.
    ELSE
      SSI(I,K)=UIF0(I,K,1)/(SIF(I,K,1)-SIC)
    END IF
    SSIT(I,K)=UIF0(I,K,2)/SIF(I,K,2)
  END DO
END DO

C
C   Initialize interfacial failure displacement
C
DO J=1,2
  UIF(I,K,J)=UIF0(I,K,J)
END DO
END DO

```

```

        END DO

C
C Find the BIGP and PP matrices

        CALL BIGPMAT
        CALL BIGPPMAT

        ITER=1
C
C Multiply Pplast by the accumulated plastic strain for each ply,
C and determine the plastic component of the forcing vector.
C
        DO J=1,MATSIJ
            PPVEC(J)=0.
        END DO

        DO I=1,NUMPLY

            DO J=1,8

                DO K=1,12

                    PPVEC(8*I-(8-J)) = PPVEC(8*I-(8-J)) + PP(I,J,K)*
>                                     VEPSMPO(I,K)

                END DO

            END DO

        END DO

C
C Add up the complete forcing vector.
C
        DO I=1,NUMPLY
            VI(8*I-6)=A(I)*SSI(I,1)*SIC
            VI(8*I-4)=A(I)*SSI(I,1)*SIC
            VI(8*I-3)=A(I)*SSI(I,2)*SIC
        END DO

        DO I=1,MATSIJ
            FVEC(I) = DLVEC(I)+LVEC0(I)-PPVEC(I)+VI(I)
        END DO

C
C Solve by matrix inversion for coupled stress terms
C

        CALL GAUSS(MATSIJ,BIGP,FVEC,XVEC)

C
C
C Stresses in matrix and interface regions for elastic response.
C

```

```

C      CALL STRESS(NUMPLY, EPSMPO, XVEC, TINT, A, B, C, SM, SF, SI, SPLY,
>SSIT, S44F, S11M, S12M, S44M)

C
C      Elastic strain rate if step is completely elastic response.
C
      DO I=1, NUMPLY
        DO J=1, NR
          V1=SM(I, J, 1)-SM0(I, J, 1)
          V2=SM(I, J, 2)-SM0(I, J, 2)
          V3=SM(I, J, 3)-SM0(I, J, 3)
          V4=SM(I, J, 4)-SM0(I, J, 4)
          EPSEDOT(I, J, 1) = (S11M*V1+S12M*V2+S12M*V3)/DTIM
          EPSEDOT(I, J, 2) = (S12M*V1+S22M*V2+S23M*V3)/DTIM
          EPSEDOT(I, J, 3) = (S12M*V1+S23M*V2+S22M*V3)/DTIM
          EPSEDOT(I, J, 4) = S44M*V4/DTIM
          EPSEFFDE(I, J) = SQRT(2./3.*(EPSEDOT(I, J, 1)**2+
> EPSEDOT(I, J, 2)**2+EPSEDOT(I, J, 3)**2+0.5*(EPSEDOT(I, J, 4)**2)))
        END DO
      END DO

C
C      Set initial effective plastic strain rate for the step to zero
C
      DO I=1, NUMPLY
        DO J=1, NR
          EPSEFFDP(I, J) = 0
        END DO
      END DO

C
C      Set the initial interface failure displacement increment to zero
C
      DO I=1, NUMPLY
        DO J=1, 2
          DO K=1, 2
            DUIF(I, J, K) = 0.0
          END DO
        END DO
      END DO

C
C      Find temperature dependent constants for the type of viscoplastic
C      analysis chosen
C
C      Bodner-Partom with back stress
C
      FS3=TDEP(NTEMM, TEMPROPM, F3, TEM1)
      FS1=TDEP(NTEMM, TEMPROPM, F1, TEM1)
      OMEGSAT = TDEP(NTEMM, TEMPROPM, OSAT, TEM1)
      Z=TDEP(NTEMM, TEMPROPM, Z0, TEM1)

```

```

C      EN = TDEP (NTEMM, TEMPROPM, BN, TEM1)

C      Bodner-Partom with directional hardening

      BND1=TDEP (NTEMM, TEMPROPM, BND, TEM1)
      BM1D1=TDEP (NTEMM, TEMPROPM, BM1D, TEM1)
      Z1D1=TDEP (NTEMM, TEMPROPM, Z1D, TEM1)
      R1D1=TDEP (NTEMM, TEMPROPM, R1D, TEM1)
      A1D1=TDEP (NTEMM, TEMPROPM, A1D, TEM1)
      BM2D1=TDEP (NTEMM, TEMPROPM, BM2D, TEM1)
      Z2D1=TDEP (NTEMM, TEMPROPM, Z2D, TEM1)
      R2D1=TDEP (NTEMM, TEMPROPM, R2D, TEM1)
      A2D1=TDEP (NTEMM, TEMPROPM, A2D, TEM1)
      Z3D1=TDEP (NTEMM, TEMPROPM, Z3D, TEM1)

      DZ1DT = TDEP (NTEMM, TEMPROPM, Z1D, (TEM1+1.))-Z1D1
      DZ2DT = TDEP (NTEMM, TEMPROPM, Z2D, (TEM1+1.))-Z2D1
      DZ3DT = TDEP (NTEMM, TEMPROPM, Z3D, (TEM1+1.))-Z3D1
      TDOT = DTEM/DTIM

C
C
C      Bodner-Partom with isotropic hardening only.
C
C
C      PNA1=TDEP (NTEMM, TEMPROPM, BND, TEM1)
C      PMA1=TDEP (NTEMM, TEMPROPM, BM1D, TEM1)
C      Z1D1=TDEP (NTEMM, TEMPROPM, Z1D, TEM1)
C      Z2D1=TDEP (NTEMM, TEMPROPM, Z2D, TEM1)
C
C
C
C      DO I=1,NUMPLY
C        DO J=1,NR
C          RT(I,J) = SQRT(3.*K2(I,J))
C          ZI(I,J) = PZI(I,J)*(Z1D1-Z2D1)+Z2D1
C          DO K=1,4
C            BETA(I,J,K) = BETA0(I,J,K)
C            OMEG(I,J,K) = OMEG0(I,J,K)
C          END DO
C        END DO
C      END DO

C      Begin iteration
C
C
C      Calculate the deviatoric stress and change in deviatoric stress
C      from the previous step.
C
C
2  DO I=1,NUMPLY
    DO J=1,NR
      HYDRO = (SM(I,J,1)+SM(I,J,2)+SM(I,J,3))/3.
      SMP(I,J,1)=SM(I,J,1)-HYDRO

```

```

      SMP(I,J,2)=SM(I,J,2)-HYDRO
      SMP(I,J,3)=SM(I,J,3)-HYDRO
      SMP(I,J,4)=SM(I,J,4)
      DO K=1,4
        DSMP(I,J,K)=SMP(I,J,K)-SMP0(I,J,K)
      END DO
      END DO
      END DO
C
C   Call the appropriate analysis method
C
C   Bodner-Partom with back stress
C
C      CALL BODBAK(FS1,FS3,OMEGSAT,EN,Z,DTIM,RT,DSMP,EPSEFFDP,SMP,
C      >OMEG,EPSCALDP)
C
C   Bodner-Partom with directional hardening
C
C      CALL BODDIR(Z0D1,BND1,BM1D1,Z1D1,R1D1,A1D1,BM2D1,Z2D1,R2D1,
C      >A2D1,Z3D1,DZ1DT,DZ2DT,DZ3DT,DNOT,TDOT,DTIM,SMP,SM,RT,EPSEFFDP,
C      >BETA,BETA0,ZI,ZI0,EPSCALDP)
C
C   Bodner-Partom with isotropic hardening only
C
C      CALL BODISO(Z2D1,PNA1,PMA1,Z1D1,DNOT,DTIM,SMP,SM,RT,
C      >ZI,ZI0,EPSEFFDP,EPSCALDP)
C
C
C   If load increment is too great, split it in half.
C
C
      IF(ITER.EQ.1) THEN
        DO I=1,NUMPLY
          DO J=1,2
            IF(SI(I,J,1).GT.(SIF(I,J,1)+TOLV4)) GO TO 7
            IF(ABS(SI(I,J,2)).GT.ABS(SIF(I,J,2)+TOLV4)) GO TO 7
          END DO
          DO J=1,NR
            IF(EPSCALDP(I,J)/TOLV1.GT.EPSEFFDE(I,J)) THEN
              IF(EPSEFFDE(I,J).LT.1.0E-06) THEN
                DO II=1,NUMPLY
                  DO JJ=1,NR
                    EPSEFFDE(II,JJ) = 1.0E-06
                  END DO
                END DO
                GO TO 8
              END IF
            END IF
          END DO
        END DO
      7      M(N)=2*M(N)-MM+1
            IF(M(N).GT.1000) THEN
              WRITE(6,*)'MAXIMUM Subincrements Exceeded'
              WRITE(6,*)'Last incrementing ratio = ',
                >EPSCALDP(I,J)/EPSEFFDE(I,J)
            END IF

```

```

        READ(5,*)
        NDUMP = 1
        RETURN
        END IF
        DTEM=DTEM/2.
        DTIM=DTIM/2.
        DO JJ=1,MATSI2
            DLVEC(JJ)=DLVEC(JJ)/2.
        END DO
        GO TO 1
        END IF
    END DO

END DO

END IF

C
C Find new effective plastic strain rate and change in plastic
C strain based on the calculation.
C

8 DO I=1,NUMPLY
    DO J=1,NR
        DEPSDP(I,J) = EPSCALDP(I,J)-EPSEFFDP(I,J)

        IF(DEPSDP(I,J).GT.0.) THEN
            RATIO = DEPSDP(I,J)/(DEPSDP(I,J)+TOLV2*EPSEFFDE(I,J))
        ELSEIF(DEPSDP(I,J).LT.0.) THEN
            RATIO = -DEPSDP(I,J)/(DEPSDP(I,J)-TOLV2*EPSEFFDE(I,J))
        ELSE
            RATIO = 0.
        END IF
        EPSEFFDP(I,J)=EPSEFFDP(I,J)+RATIO*EPSEFFDE(I,J)

        IF(RT(I,J).EQ.0.0) THEN
            DO K=1,4
                DEPSMP(I,J,K)=0.0
            END DO
        ELSE
            DO K=1,4
                DEPSMP(I,J,K)=DTIM*1.5*EPSEFFDP(I,J)*(SMP(I,J,K)-
> OMEG(I,J,K))/RT(I,J)
            END DO
        END IF
        DEPSMP(I,J,4)=2*DEPSMP(I,J,4)
    END DO

    DO J=1,2
        IF(SI(I,J,1).LE.SIF(I,J,1).AND.DUIF(I,J,1).EQ.0.0) THEN
            DUIF(I,J,1)=0.0
        ELSE
            DUIF(I,J,1)=FACT1*SSI(I,J)*(SI(I,J,1)-SIF(I,J,1))
        END IF
    END IF

```

```

      IF (ABS(SI(I,J,2)) .LE. ABS(SIF(I,J,2)) .AND. DUIF(I,J,
>                                     2) .EQ. 0.0) THEN
          DUIF(I,J,2) = 0.0
      ELSE
          DUIF(I,J,2) = FACT1 * SSIT(I,J) * (ABS(SI(I,J,2)) -
>                                     ABS(SIF(I,J,2)))
      END IF
      DO K=1,2
          UIF(I,J,K) = UIF(I,J,K) + DUIF(I,J,K)
      END DO
  END DO
END DO

C
C   Calculate new matrix stresses for this iteration
C
  DO I=1,NUMPLY
      DO J=1,NR
          DO K=1,4
              EPSMP(I,J,K) = DEPSMP(I,J,K) + EPSMP0(I,J,K)
          END DO
          VEPSMP(I,J) = EPSMP(I,J,4)
          VEPSMP(I,3*J+1) = EPSMP(I,J,1)
          VEPSMP(I,3*J+2) = EPSMP(I,J,2)
          VEPSMP(I,3*J+3) = EPSMP(I,J,3)
          END DO
      END DO

C   Determine stress point where interfacial failure begins.

      DO I=1,NUMPLY
          DO J=1,2
              SIF(I,J,1) = TDEP(3,UIFN,SIFN,UIF(I,J,1))
              SIF(I,J,2) = TDEP(3,UIFT,SIFT,UIF(I,J,2))
              UI(I,J,1) = SSI(I,J) * (SI(I,J,1) - SIC)
              UI(I,J,2) = SSIT(I,J) * SI(I,J,2)
              IF (SI(I,J,1) .LE. SIC .OR. UI(I,J,1) .LT. 0.) THEN
                  SSI(I,J) = 0.
              ELSE
                  SSI(I,J) = UIF(I,J,1) / (SIF(I,J,1) - SIC)
              END IF
              SSIT(I,J) = UIF(I,J,2) / SIF(I,J,2)
          END DO
      END DO

      CALL BIGPMAT
      CALL BIGPPMAT

C
C   Multiply Pplast by the accumulated plastic strain for each ply,
C   and determine the plastic component of the forcing vector.
C
      DO J=1,MATSIZ
          PPVEC(J) = 0.

```

```

END DO

DO I=1,NUMPLY

DO J=1,8

DO K=1,12

PPVEC(8*I-(8-J)) = PPVEC(8*I-(8-J)) + PP(I,J,K)*
>
VEPSMP(I,K)

END DO

END DO

END DO

C
C Add up the complete forcing vector.
C
DO I=1,NUMPLY
VI(8*I-6)=A(I)*SSI(I,1)*SIC
VI(8*I-4)=A(I)*SSI(I,1)*SIC
VI(8*I-3)=A(I)*SSI(I,2)*SIC
END DO

DO I=1,MATSIZ
FVEC(I) = DLVEC(I)+LVEC0(I)-PPVEC(I)+VI(I)
END DO

C
C Solve by matrix inversion for coupled stress terms
C

CALL GAUSS(MATSIZ,BIGP,FVEC,XVEC)

C
C
C Stresses in matrix regions
C
C

CALL STRESS(NUMPLY,EPSMP,XVEC,TINT,A,B,C,SM,SF,SI,SPLY,
>SSIT,S44F,S11M,S12M,S44M)

C
C Is iteration complete?
C
IF(ITER.GT.3000) THEN

WRITE(6,1090) N,NCYC
1090 FORMAT(' * * * ERROR * * *',/, ' Maximum iterations exceeded',

```

```
>' while attempting to calculate',/, ' load point',I3,' in ',
>'loading cycle',I7)
```

```
      WRITE(6,*) 'M = ',M(N)
      WRITE(6,*) 'SUBINCREMENT POINT FURTHER?'
      READ(5,*) CHAR1
      IF (CHAR1.EQ.'Y'.OR.CHAR1.EQ.'y') GO TO 7
        NDUMP = 1
        RETURN
      ELSE

        DO I=1,NUMPLY
          DO J=1,2
            DO K=1,2
              IF (ABS(DUIF(I,J,K)).GT.TOLV3*UIF(I,J,K)) THEN
                ITER=ITER+1
                GO TO 2
              END IF
            END DO
          END DO
        END DO

        DO J=1,NR
          IF (ABS(DEPSDP(I,J)).GT.ABS(TOLV3*EPSEFFDE(I,J))) THEN
            ITER=ITER+1
            GO TO 2
          END IF
        END DO
      END DO

    END IF
```

C
C
C
C
C
C

Calculate the strains for this increment from the stress-strain relations.

```
    DO I=1,NUMPLY
      EPSF(I,1) = S11F*SF(I,1)+S12F*SF(I,2)+S12F*SF(I,3)+
    >   AF1*(TEM1-TEMPREF) - EPSTHF1
      EPSF(I,2) = S12F*SF(I,1)+S22F*SF(I,2)+S23F*SF(I,3)+
    >   AF2*(TEM1-TEMPREF) - EPSTHF2
      EPSF(I,3) = S12F*SF(I,1)+S23F*SF(I,2)+S22F*SF(I,3)+
    >   AF2*(TEM1-TEMPREF) - EPSTHF2
      EPSF(I,4) = S44F*SF(I,4)
      DO J=1,NR
        EPSM(I,J,1) = S11M*SM(I,J,1)+S12M*SM(I,J,2)+S12M*SM(I,J,3)+
    >   AM1*(TEM1-TEMPREF)-EPSTHM1+EPSMP(I,J,1)
        EPSM(I,J,2) = S12M*SM(I,J,1)+S22M*SM(I,J,2)+S23M*SM(I,J,3)+
    >   AM2*(TEM1-TEMPREF)-EPSTHM2+EPSMP(I,J,2)
        EPSM(I,J,3) = S12M*SM(I,J,1)+S23M*SM(I,J,2)+S22M*SM(I,J,3)+
    >   AM2*(TEM1-TEMPREF)-EPSTHM2+EPSMP(I,J,3)
        EPSM(I,J,4) = S44M*SM(I,J,4)+EPSMP(I,J,4)
      END DO
    END DO
```

```

        UI(I,1,2) = SSIT(I,1)*SI(I,1,2)
        UI(I,2,2) = SSIT(I,2)*SI(I,2,2)
        UI(I,1,1) = SSI(I,1)*(SI(I,1,1)-SIC)
        UI(I,2,1) = SSI(I,2)*(SI(I,2,1)-SIC)
    END DO

C
C   Ply stresses
C
    DO I=1,NUMPLY

        SPLY(I,1)=(A(I)**2.*SF(I,1)+A(I)*B(I)*SM(I,1,1)+B(I)*C(I)
>        *SM(I,2,1)+A(I)*C(I)*SM(I,3,1))/(A(I)+B(I))*(A(I)+C(I))
        SPLY(I,2)=(A(I)*SM(I,1,2)+C(I)*SM(I,2,2))/(A(I)+C(I))

    END DO

    DO I=1,NUMPLY

        SIGPLY(I,1)=TINT(I,1,1)*SPLY(I,1)+TINT(I,2,1)*SPLY(I,2)+
>        TINT(I,3,1)*SPLY(I,3)
        SIGPLY(I,2)=TINT(I,1,2)*SPLY(I,1)+TINT(I,2,2)*SPLY(I,2)+
>        TINT(I,3,2)*SPLY(I,3)
        SIGPLY(I,3)=TINT(I,1,3)*SPLY(I,1)+TINT(I,2,3)*SPLY(I,2)+
>        TINT(I,3,3)*SPLY(I,3)

    END DO

        DO I=1,3
            EPSC(I)=XVEC(I)
        END DO

C
C   Time
C
        TIM1=DTIM+TIM0

C
C   Set values for next increment
C
        DO I=1,NUMPLY
            DO J=1,NR
                DO K=1,4
                    SM0(I,J,K)=SM(I,J,K)
                    SMP0(I,J,K)=SMP(I,J,K)
                    EPSMP0(I,J,K)=EPSMP(I,J,K)
                    OMEG0(I,J,K)=OMEG(I,J,K)
                    BETA0(I,J,K)=BETA(I,J,K)
                END DO
                PZI(I,J)=(ZI(I,J)-Z2D1)/(Z1D1-Z2D1)
                ZI0(I,J)=ZI(I,J)
            END DO
        END DO

```

```

END DO
DO I=1,NUMPLY
  DO J=1,2
    DO K=1,2
      SI0(I,J,K)=SI(I,J,K)
      UI0(I,J,K)=UI(I,J,K)
      UIF0(I,J,K)=UIF(I,J,K)
    END DO
    PUIF1=UIF(I,J,1)/UIFN(2)
    PUIF2=UIF(I,J,2)/UIFT(2)
    IF(PUIF1.GT.PUIF2) THEN
      UIF0(I,J,2)=PUIF1*UIFT(2)
      UIF0(I,J,1)=UIF(I,J,1)
    ELSE
      UIF0(I,J,1)=PUIF2*UIFN(2)+0.0000001
      UIF0(I,J,2)=UIF(I,J,2)
    END IF
  END DO
END DO
DO I=1,MATSIZ
  LVEC0(I)=DLVEC(I)+LVEC0(I)
END DO
TEM0=TEM1
TIM0=TIM1

C
C   Check if desired load point has been achieved.
C

      IF(MM.LT.M(N)) THEN
        MM = MM+1
        GO TO 1
      END IF

C
C
C   Write to scratch file
C
C

      SCX=NX(N)/(ZPOS(NUMPLY+1)-ZPOS(1))
      SCY=NY(N)/(ZPOS(NUMPLY+1)-ZPOS(1))
      SCXY=NX(N)/(ZPOS(NUMPLY+1)-ZPOS(1))

      WRITE(8) TIM1,TEM1,(EPSC(K),K=1,3),SCX,SCY,SCXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,NR),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

C
C
C   Save present load vector as initial point for subsequent load.
C

```

C

```
      DO I=1,MATSIZ
        LVEC0(I)=LVEC1(I)
      END DO
      TEM0=TEMP(N)
```

```
      RETURN
```

```
      END
```

C

C * * * * *

C

C The subroutine WRITEDAT is for writing solution sets from the
C FTS method to formatted or unformatted files for either printing
C or saving for future analysis.

C

C * * * * *

```
      SUBROUTINE WRITEDAT
```

```
      REAL NUF12(40), NUF23(40), NUM12(40), NUM23(40)
```

```
      CHARACTER FIB*1,MATE*1,MATP*1,MATV*1,TCHARP*1,TCHARV*1,  
>COMP*40, ECHAR*1, LCHAR*1
```

```
      COMMON/TYPEMOD/NA,NUMPLY,ZPOS(7),ANGLE(6)
```

```
      COMMON/MATERIAL/NUF12(40),NUF23(40),NUM12(40),NUM23(40),GM12(40),  
>EM22(40),EM11(40),AR,VF,GF12(40),EF22(40),EF11(40),HPRAMPTS(40),  
>YSPTS(40),TEMPROPF(40),TEMPROP(40),NTEMP,NTEMM,HD,ALPHAF1(40),  
>ALPHAF2(40),ALPHAM1(40),ALPHAM2(40),OSAT(40),F1(40),F3(40),Z0(40),  
>BN(40),DNOT,Z0D(40),BND(40),BM1D(40),Z1D(40),R1D(40),  
>A1D(40),BM2D(40),Z2D(40),R2D(40),A2D(40),Z3D(40),TEMPREF,NTEMI,  
>SIC,TEMPROPI(40),SIFN1(40),UIFN10A(40),SIFT1(40),UIFT10A(40)
```

```
      COMMON/LOADING/NC1,NC2,NUMP,NUMCYC,DT(9999),TEMP(9999),NX(9999),  
>NY(9999),NXY(9999),M(9999)
```

```
      COMMON/ELASTPROP/SSF11(40),SSF22(40),SSF12(40),SSF23(40),  
>SSF44(40),SSM11(40),SSM22(40),SSM12(40),SSM23(40),SSM44(40),  
>A(6),B(6),C(6)
```

```
      COMMON/TOLERANCE/TOLP1,TOLP2,TOLP3,TOLV1,TOLV2,TOLV3,TOLV4
```

```
      COMMON/WORDS/FIB,MATE,MATP,MATV,TCHARP,TCHARV,COMP,ECHAR,LCHAR
```

```
      REAL NX(9999),NY(9999),NXY(9999),EPSCOM(3),SIGPLY(6,3),EPSF(6,4),  
>SF(6,4),EPSM(6,3,4),SM(6,3,4),UI(6,2,2),SI(6,2,2)
```

```
      CHARACTER OUTDAT*15,CYCSYM(99)*1
```

3 REWIND 8

```

WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*)
WRITE(6,*) ' * * * * * '
WRITE(6,*) ' *      Write Out Results      * '
WRITE(6,*) ' * * * * * '
WRITE(6,*)
WRITE(6,*)
WRITE(6,*) '      (1) Unformatted file '
WRITE(6,*) '      (2) Global ASCII file '
WRITE(6,*) '      **N/A**(3) Formatted file for printing '
WRITE(6,*) '      (4) Return to results menu '
WRITE(6,*)
WRITE(6,*)
WRITE(6,*) ' Enter Selection: '
READ(5,*) NW

IF(NW.EQ.1) THEN
  IF(LCHAR.EQ.'N') THEN
    WRITE(6,101)
101  FORMAT(/,' No solution set exists in the program database.')
    READ(5,*)
    GO TO 3
  END IF

  WRITE(6,*) ' Enter name of the unformatted file to be created.'
  READ(5,201) OUTDAT
201  FORMAT(A15)
  OPEN(30,FILE=OUTDAT,STATUS='NEW',FORM='UNFORMATTED')

  WRITE(30) NA,NUMPLY,NTEMF,NTEMM,NTEMI,AR,VF,HD,DNOT,SIC
>    TEMPREF,NC1,NC2,NUMP,NUMCYC,FIB,MATE,MATP,MATV,TCHARP,
>    TCHARV,COMP,ECHAR,LCHAR

  WRITE(30) (ZPOS(I),I=1,NUMPLY+1),(ANGLE(J),J=1,NUMPLY)

  WRITE(30) (TEMPROPF(I),NUF12(I),NUF23(I),GF12(I),EF22(I),
>    EF11(I),ALPHAF1(I),ALPHAF2(I),I=1,NTEMF)
  WRITE(30) (TEMPROPM(I),NUM12(I),NUM23(I),GM12(I),EM22(I),
>    EM11(I),ALPHAM1(I),ALPHAM2(I),HPRAMPTS(I),YSPTS(I),
>    OSAT(I),F1(I),F3(I),Z0(I),BN(I),Z0D(I),BND(I),
>    BM1D(I),Z1D(I),R1D(I),A1D(I),BM2D(I),Z2D(I),
>    R2D(I),A2D(I),Z3D(I),I=1,NTEMM)

```

```

        WRITE(30) (TEMPROI(I),SIFN1(I),UIFN1OA(I),
>                SIFT1(I),UIFT1OA(I),I=1,NTEMI)

        DO I=1,NUMP
            WRITE(30) DT(I),TEMP(I),NX(I),NY(I),NXY(I),M(I)
        END DO

        DO N=1,NC1-1
            READ(8) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

            WRITE(30) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

        END DO

        DO NCYC=1,NUMCYC

            DO N=NC1,NC2
                READ(8) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

                WRITE(30) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

            END DO

        END DO

        DO N=NC2+1,NUMP
            READ(8) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

            WRITE(30) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

        END DO

        CLOSE(30)

        ELSEIF(NW.EQ.2) THEN

```

```

        IF(LCHAR.EQ.'N') THEN
        WRITE(6,101)
        READ(5,*)
        GO TO 3
    END IF

    WRITE(6,*) ' Enter name of the global ASCII file to be created.'
    READ(5,201) OUTDAT
    OPEN(30,FILE=OUTDAT,STATUS='NEW')

    WRITE(30,*) NA,NUMPLY,NTEMF,NTEMM,NTEMI,AR,VF,HD,DNOT,SIC
>    TEMPREF,NC1,NC2,NUMP,NUMCYC

    WRITE(30,1000) FIB,MATE,MATP,MATV,TCHARP,TCHARV,ECHAR,
>    LCHAR,COMP

1000 FORMAT(8A1,A40)

    WRITE(30,*) (ZPOS(I),I=1,NUMPLY+1),(ANGLE(J),J=1,NUMPLY)

    DO I=1,NTEMF
    WRITE(30,*) TEMPROPF(I),NUF12(I),NUF23(I),GF12(I),EF22(I),
>    EF11(I),ALPHAF1(I),ALPHAF2(I)
    END DO

    DO I=1,NTEMM
    WRITE(30,*) TEMPROMP(I),NUM12(I),NUM23(I),GM12(I),EM22(I),
>    EM11(I),ALPHAM1(I),ALPHAM2(I),HPRAMPTS(I),YSPTS(I),
>    OSAT(I),F1(I),F3(I),Z0(I),BN(I),Z0D(I),BND(I),
>    BM1D(I),Z1D(I),R1D(I),A1D(I),BM2D(I),Z2D(I),
>    R2D(I),A2D(I),Z3D(I)
    END DO

    DO I=1,NTEMI
    WRITE(30,*) TEMPROMPI(I),SIFN1(I),UIFN10A(I),
>    SIFT1(I),UIFT10A(I)
    END DO

    DO I=1,NUMP
    WRITE(30,*) DT(I),TEMP(I),NX(I),NY(I),NXY(I),M(I)
    END DO

    DO N=1,NC1-1
    READ(8) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

    WRITE(30,*) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,

```

```

>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

END DO

```

```

DO NCYC=1,NUMCYC

```

```

DO N=NC1,NC2

```

```

READ(8) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

WRITE(30,*) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

END DO

```

```

END DO

```

```

DO N=NC2+1,NUMP

```

```

READ(8) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

WRITE(30,*) TIM1,TEM1,(EPSCOM(K),K=1,3),PNX,PNY,PNXY,
>((SIGPLY(I,K),K=1,3),(EPSF(I,K),SF(I,K),K=1,4),((EPSM(I,J,K),
>SM(I,J,K),K=1,4),J=1,3),((UI(I,J,K),SI(I,J,K),K=1,2),J=1,2),
>I=1,NUMPLY)

```

```

END DO

```

```

CLOSE(30)

```

```

ELSEIF(NW.EQ.3) THEN

```

```

IF(LCHAR.EQ.'N') THEN

```

```

WRITE(6,101)

```

```

READ(5,*)

```

```

GO TO 3

```

```

END IF

```

```

ELSEIF(NW.EQ.4) THEN

```

```

RETURN

```

```

END IF

```

```

GO TO 3

```

```

END

```

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE July 1996		3. REPORT TYPE AND DATES COVERED Program Manual
4. TITLE AND SUBTITLE User's Manual for the <u>L</u> aminated Composite <u>I</u> nelastic <u>S</u> olver Computer Program				5. FUNDING NUMBERS
6. AUTHOR(S) David D. Robertson, Maj, USAF				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology, Wright-Patterson AFB, OH 45433-7765				8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/ENY/TR96-01
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Brian Sanders AFOSR/NA Bolling AFB, DC 20332				10. SPONSORING/MONITORING AGENCY REPORT NUMBER
Ted Nicholas WL/MLLN Wright-Patterson AFB, OH 45433				
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited				12b. DISTRIBUTION CODE A
13. ABSTRACT (Maximum 200 words) The program described in this document (LISOL) is for analyzing the thermal and inelastic behavior of continuous fiber-reinforced laminated composites with symmetric layups. Its main application has been in the area of metal matrix composites (MMCs). In particular, titanium-based MMCs have been extensively analyzed using LISOL. The program is very adept at performing thermomechanical cyclic loading. Inelastic material behavior is modeled using the Bodner-Partom viscoplastic theory, and the interface between fiber and matrix may be modeled with a progressive failure scheme. These nonlinearities as well as the thermal and elastic properties of the constituents are combined through a micromechanics approach which assumes an average representative volume element for a single ply. The equations for a single ply are then assembled for the entire layup through the classical laminated plate theory. LISOL is designed to require modest computer resources. For example, most solutions (10 cycles or less) can be accomplished in a few minutes on a personal computer.				
14. SUBJECT TERMS Fiber-Reinforced Composites, Composite Micromechanics, Metal Matrix Composites, Thermomechanical Fatigue				15. NUMBER OF PAGES 86
				16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	