# ANSYS USER Material Subroutine USERMAT

**Mechanics Group**

**Development Department**

**ANSYS, Inc.**

**Southpointe**

**275 Technology Drive**

**Canonsburg, PA 15317**

**November, 1999**

# CONTENTS

# 1. INTRODUCTION

The user material routine, USERMAT, is an ANSYS user-programmable feature for use with the 18x family elements, which include LINK180, SHELL181, PLANE182, PLANE183, SOLID185, SOLID186, SOLID187, BEAM188 and BEAM189. Its function is to allow user to write their own material constitutive equations within a newly developed general material framework for the 18x family elements. This subroutine is called at all material integration points of these elements during the solution phase. The input parameters for the USERMAT subroutine is defined by command **TB**,**USER**.

This report provides guidelines on how to write a user material routine using USERMAT.

# 2. USER MATERIAL ROUTINE (USERMAT)

The user material routine, USERMAT, is used to define material stress-strain relation, the mechanical constitutive behavior of materials. It can be used in any ANSYS analysis procedure that requires mechanical behavior. If the state variables are used in the USERMAT routine, the number of state variables must be defined by the command **TB**, **STATE**. For every Newton-Raphson iteration, USERMAT is called at every material integration point. ANSYS passes in stresses, strains and state variables as the values at the beginning of the time increment, and the current strain increment, USERMAT then updates the stresses and state variables to the values at the end of the time increment. USERMAT must also provide the material Jacobian matrix, $\partial\sigma_{ij}/\partial\varepsilon_{ij}$ .

Stress, strain and material Jacobian tensors are stored in vector or matrix forms. The order of components for the stress, the strain and other tensors is

3D stress state:

11, 22, 33, 12, 23, 13,

2D plane stress/strain and axisymmetric stress states:

11, 22, 33, 12,

Stress state for BEAM188/189:

11, 13, 12.

Stress state for LINK180:

11.

The order of components for material Jacobian matrix is

3D stress state:

| | | | | | |
|---|---|---|---|---|---|
| 1111 | 1122 | 1133 | 1112 | 1123 | 1113 |
| 2211 | 2222 | 2233 | 2212 | 2223 | 2213 |
| 3311 | 3322 | 3333 | 3312 | 3323 | 3313 |
| 1211 | 1222 | 1233 | 1212 | 1223 | 1213 |
| 2311 | 2322 | 2333 | 2312 | 2323 | 2313 |
| 1311 | 1322 | 1333 | 1312 | 1323 | 1313 |

2D plane strain and axisymmetric stress states:

| | | | |
|---|---|---|---|
| 1111 | 1122 | 1133 | 1112 |
| 2211 | 2222 | 2233 | 2212 |
| 3311 | 3322 | 3333 | 3312 |
| 1211 | 1222 | 1233 | 1212 |

2D plane stress states:

| | | |
|---|---|---|
| 1111 | 1122 | 1112 |
| 2211 | 2222 | 2212 |
| 1211 | 1222 | 1212 |

BEAM188/189:

| | | |
|---|---|---|
| 1111 | 1113 | 1112 |
| 1311 | 1313 | 1312 |
| 1211 | 1213 | 1212 |

LINK180:

1111.

User subroutine interface USERMAT:

```
      subroutine usermat(
     &                   matId, elemId,kDomIntPt, kLayer, kSectPt,
     &                   ldstep,isubst,keycut,
     &                   nDirect,nShear,ncomp,nStatev,nProp,
     &                   Time,dTime,Temp,dTemp,
     &                   stress,statev,dsdePl,sedEl,sedPl,epseq,
     &                   Strain,dStrain, epsPl, prop, coords,
     &                   tsstif, epsZZ,
     &                   var1, var2, var3, var4, var5,
     &                   var6, var7, var8)

      INTEGER
     &                 matId, elemId,
     &                 kDomIntPt, kLayer, kSectPt,
     &                 ldstep,isubst,keycut,
     &                 nDirect,nShear,ncomp,nStatev,nProp
      DOUBLE PRECISION
     &                 Time,    dTime,   Temp,    dTemp,
     &                 sedEl,   sedPl,   epseq,   epsZZ
      DOUBLE PRECISION
     &                 stress  (ncomp  ), statev  (nStatev),
     &                 dsdePl  (ncomp,ncomp),
     &                 Strain  (ncomp  ), dStrain (ncomp  ),
     &                 epsPl   (ncomp  ), prop    (nProp  ),
     &                 coords  (3),       rotateM (3,3),
     &                 defGrad_t(3,3),    defGrad(3,3),
     &                 tsstif  (2)
```

# 3. TABLE COMMANDS

## TB, USER command

To use the user material option, the **TB, USER** command must be first issued so that the user material can be defined. The table command for USER material option is:

TB,USER, matId, NTEMPS, NPTS

matId material reference number,

NTEMPS Number of temperature points (temperature is defined by **TBTEMP**),

NPTS Number of material constants at a given temperature point (material constants is defined by **TBDATA**).

The temperatures and material constants are defined through commands **TBTEMP** and **TBDATA** as standard ANSYS procedures. The material properties at an intermediate temperature point are interpolated and passed to the USERMAT subroutine. An example of defining a user material option is as follows:

```
tb,user,1,2,4                    ! define mat. 1 as user mat. with 2
                                 ! temperatures and 4 data points
                                 ! at each temp. point.
tbtemp,1.0                       ! first temp.
tbdata,1,19e5, 0.3, 1e3,100,     ! 4 mat constants for 1 temp.
tbtemp,2.0                       ! second temp.
tbdata,1,21e5, 0.3, 2e3,100,     ! 4 mat constants for 2 temp.
```

## TB, STATE command

If state variables are used in the USERMAT subroutine, the number of the state variables need to be firstly defined by command **TB, STATE**:

TB,STATE, matId, , NPTS

matId material reference number,

NPTS Number of the state variables to be used in USERMAT.

This command is used only for defining the number of the state variables and must be always associated with a user material option. No temperatures or data are associated with this command. State variables are initialized to zero at beginning of an analysis. An example of defining the number of the state variables is as follows:

```
tb,state,1,,4,                   ! define mat. 1 has 4 state variables
```

# 4. VARIABLE DESCRIPTION

## Input arguments

The following variables are passed in as information for use by the subroutine, they must not be changed in the user material subroutine USERMAT.

| | |
|---|---|
| matId | Integer variable contains material index number. |
| elemId | Integer variable contains element number. |
| kDomIntPt | Integer variable contains material integration point number. |
| kLayer | Integer variable contains integer variable, Layer number. |
| kSectPt | Integer variable contains section point number. |
| ldstep | Integer variable contains load step number. |
| isubst | Integer variable contains substep number |
| nDirect | Number of direct components of the stress or strain vector at material point. |
| nShear | Number of shear components of the stress or strain vector at material point, this is engineering components. |
| ncomp | Total number of the stress or strain components at material point (nDirect + nShear). |
| nstatev | Number of state variables. This is defined by NPTS through command **TB, STATE**. |
| nProp | Number of material constants, defined by NPTS in command **TB,USER**. |
| Temp | Double precision variable contains the current temperature. |
| dTemp | Double precision variable contains the current temperature increment. |
| Time | Double precision variable contains the total time at the beginning of the time increment. |
| dTime | Double precision variable contains the current time increment. |
| Strain | Double precision array contains the total strains at the beginning of the time increment. Array size is ncomp. If there are thermal strains, the strain passed |

into USERMAT are the mechanical strains only. The thermal strains (defined by command **MP, ALPHA** and temperature load) have been subtracted from the total strains. For large deformation problem (NLGEOM, ON), the strain components have been updated to account for rigid body rotation before they are passed to USERMAT and are approximately the logarithmic strains.

dStrain — Double precision array contains current strain increments. Array size is ncomp. As "Strain" array, it contains the mechanical strain increments only. The thermal strain increments (if there is any) have been subtracted from total strains increments.

prop — Double precision array contains the material constants defined by **TB,USER** and **TBDATA**. Array size is nProp. Array prop contains the material constants at current temperature point.

coords — Double precision array contains the current coordinates of the material integration points. Array size is 3.

rotateM — Double precision matrix contains the incremental rotation matrix. Matrix size is 3x3. This matrix is the increment of rigid body rotation to account for large deformation. It is a unit matrix for small deformation analysis.

defGrad_t — Double precision matrix contains deformation gradient at the beginning of the time increment. The matrix size is 3x3. The matrix components DefGrad_t(i,j) are equivalent to deformation gradient Fij at the beginning of the time increment.

defGrad — Double precision matrix contains current deformation gradient. The matrix size is 3x3. The matrix components DefGrad(i,j) are equivalent to deformation gradient Fij at the current time.

## Input output arguments

stress — Double precision array contains the stresses. Its size is defined by ncomp. The stress measure is the "true" stress. It is passed as the values of stresses at beginning of time increment and must be updated to the values of stress at the end of the time increment. For finite deformation problems, the stresses have been rotated to account for rigid body motion before they are passed in

using the Hughes-Winget rotation matrix, and thus only co-rotational part of stress integration needs to be done in the USERMAT.

statev — Double precision array contains the state variables. Its size is defined by command **TB, STATE**. It is passed as the values of state variables at the beginning of the time increment and must be updated to the values of the state variables at the end of the time increment. For finite deformation problems, any vector or tensor type of state variables must be rotated to account for rigid body motion before they are used any constitutive calculations. Rotation matrix, rotatM, is passed in for the purpose.

epspl — Double precision array contains the plastic strains. The strain measure is the "true" strain. Its size is defined by ncomp. It is passed as the values of the plastic strains at the beginning of the time increment and must be updated to the values of the plastic strains at the end of the time increment. For finite deformation problems, the plastic strains have been rotated to account for rigid body motion before they are passed in using the Hughes-Winget rotation matrix (Hughes, 1980).

sedEl — Elastic work. It is used for output purpose only and has no effect on the solution.

sedPl — Plastic work. It is used for output purpose only and has no effect on the solution.

sedCr — Creep work. It is used for output purpose only and has no effect on the solution.

## Output arguments

Following list of variables must be updated in the user material subroutine.

keycut — Integer variable as key for loading bisection/cut control:

0 - no bisect/cut,

1 - bisect/cut,

By default, keycut is 0. Set keycut to 1 when USERMAT experiences convergency difficulty in the solution of the integration of the constitutive equations. The bisect/cut factor is determined by ANSYS solution control.

epsZZ — strain component at out of plane direction for plane stress state

Required when the thickness change is accounted in plane stress or shell elements

tsstif(2) Transver shear stiffness

Tsstif(1) - GXZ

Tsstif(2) - GYZ

dsdePl(ncomp,ncomp)

Double precision array contains the material Jacobian matrix, $\partial\sigma_{ij}/\partial\varepsilon_{ij}$ . Here $\partial\sigma_{ij}$ and $\partial\varepsilon_{ij}$ are the stress and the strain increments respectively. dsdePl(i,j) denotes the change in the i-th stress component at the end of the time increment caused by an change of the j-th strain component. By default, ANSYS assumes that the element stiffness matrix is symmetric, therefore user must provide a symmetric material Jacobian matrix even it is unsymmetric. However, if an unsymmetric material Jacobian matrix is desired, an element key option, KEYOPT(5)=1, can be used to define the unsymmetric stiffness matrix for the PLANE and the SOLID elements.

# 5. SPECIFICATIONS

User material subroutine, USERMAT, is currently applicable for the elements 180, 181, 182, 183, 185, 186, 187, 188 and 189 with all the key options. However, a different material constitutive integration must be provided for the various stress states such as general 3D, plane stress and beam with or without shear stress components. For SHELL181, a plane stress algorithm for the material constitutive integration must be used. To ensure the overall numerical stability, the user should make sure that the integration scheme implemented in this subroutine is stable. ANSYS always uses full Newton-Raphson scheme for global solution to achieve better convergence rate. The material Jacobian matrix, dsdePl(i,j), must be consistent with the material constitutive integration scheme for the better convergence rate of the overall Newton-Raphson scheme.

# 6. RESTRICTIONS

This user subroutine is only applicable to elements LINK180, SHELL181, PLANE182, PLANE183, SOLID185, SOLID186, SOLID187, BEAM188 and BEAM189.

Currently the state variables used in USERMAT.F are not available in ANSYS 5.6 post-processors, such as POST1 and POST26, for post-processing purposes. However, it is expected that in ANSYS 5.6.1, the post-processors will be able to process the state variables.

USERMAT is not intended for application of modeling incompressible elastic materials, such as hyperelastic materials. A special treatment such as penalty approach may be needed to ensure the incompressibility. In any case, if the material exhibits nearly incompressible behavior, the user must ensure that a finite tangent bulk modulus is used.

# 7. UTILITY ROUTINES

vzero(**a**,n)

Initialize array **a** to zero. n is dimension of array **a**.

vmult(**a**,**b**,n,c)

Multiply a vector **a** by a constant c and output as vector **b**, **b**=**a***c, n is dimension of arrays.

vmult1(**a**,n,c)

Multiply a vector **a** by a constant c and output as itself, **a**=**a***c, n is dimension of array.

maxb(**a**,**b**,**c**,na,nb,nc,n1,n2,n3)

Multiply two double precision matrices and output as **c**, **c**=**a***__b__, na number of rows in matrix **a**, nb number of rows in matrix **b**, and nc number of rows in matrix **c**. n1 number of rows in matrix **c** to fill, n2 number of columns in matrix **c** to fill, n3 number of columns in matrix **a** and number of rows in matrix **b** to work with (the two need to be the same for the inner product).

# REFERENCES

1. Hughes, T.J.R. and Winget, J. (1980), “Finite Rotation Effects in Numerical Integration of Rate Constitutive Equations Aarising in Large-Deformation Analysis”, International Journal for Numerical Methods in engineering, Vol. 15, No. 9, pp. 1413-1418.

# APPENDIX

## 1. Example of a user material subroutine

An example of a simple bilinear plasticity material model, which is the same as TB,BISO, is used to demonstrate the user material subroutine USERMAT. The subroutine is for 3D, plane strain and axisymmetric stress states. The 3D solid element 185 is used for the analysis. Comparison is made with the prediction by ANSYS **TB,BISO** material option.

The example is a two elements test case under simple tension. Element 1 has material defined by TB,USER option, while element 2 has material defined by the TB, BISO option. A 100% of deformation is applied to both elements. Finite deformation (NLGEOM, ON) is considered. POST26 results of stresses components (Sxx, Syy ) and plastic strain components (EPxx, EPyy) are printed for both elements. There are expected to be the same.

## 2. INPUT data:

```
/batch,list
/title, mvpl-um01, gal, usermat.F test case
/com,
/com, This is a single element test case for testing usermat.F
/com, usermat.F is user materials subroutine for 18x elements.
/com, The material subroutine provided as the example
/com, is the same as the TB, BISO.
/com, A side by side comparison is made for two 185 elements,
/com, among which one is defined by TB,BISO, and another
/com, is defined as TB,USER. They are expected to produce
/com, the same results.
/com, uniaxial tension stress, large deformation.
/com,
/nopr
/nolist
/prep7

ele1=185
ele2=185
mat1=1
mat2=2


et,1,ele1
keyopt,1,2,1
mat,mat1
block,0,1,0,1,0,1
esize,,1
vmesh,1
```

```
mat,mat2
block,0,1,0,1,0,1
esize,,1
vmesh,2

elist

! define material 1 by tb,biso

mp,ex  ,mat1,20e5
mp,nuxy,mat1,0.3
tb,biso,mat1,2,4
tbtemp,1.0
tbdata,1,1e3,100,
tbtemp,2.0
tbdata,1,2e3,100,

! define material 2 by tb,user

tb,user,mat2,2,4
tbtemp,1.0                          ! first temp.
tbdata,1,19e5, 0.3, 1e3,100,        ! E, posn, sigy, H
tbtemp,2.0
tbdata,1,21e5, 0.3, 2e3,100,
tb,state,mat2,,1

! boundary condition

nsel,s,loc,x
d,all,ux
nall
nsel,s,loc,y
d,all,uy
nall
nsel,s,loc,z
d,all,uz
nall
fini

/solu
tunif,1.5
nlgeom,on
nsel,s,loc,y,1
nsubst,20,100,1
d,all,uy,1.0
time,1
nall
outres,,-10
outpr,all,-10
solv

fini
/post26
esol,2,1,,s,x,SX_BISO
esol,3,2,,s,x,SX_USER
esol,4,1,,s,y,SY_BISO
esol,5,2,,s,y,SY_USER
```

```
esol,6,1,,eppl,x,EPX_BISO
esol,7,2,,eppl,x,EPX_USER
esol,8,1,,eppl,y,EPY_BISO
esol,9,2,,eppl,y,EPY_USER

prvar,2,3,4,5
prvar,6,7,8,9

fini

/exit,no save
```

## 3. Output of POST26 results:

```
          ***** ANSYS POST26 VARIABLE LISTING *****


   TIME          1 S   X       2 S   X       1 S   Y       2 S   Y
                 SX_BISO       SX_USER       SY_BISO       SY_USER
 0.10000       -0.188102E-02 -0.188102E-02   1509.45       1509.45
 0.28750       -0.110968     -0.110968       1525.07       1525.07
 0.45625       -0.814415     -0.814415       1536.67       1536.67
 0.66204        -1.73160      -1.73160       1548.95       1548.95
 0.89592        -1.86240      -1.86240       1561.97       1561.97
  1.0000       -0.176924E-01 -0.176924E-01   1569.16       1569.16


          ***** ANSYS POST26 VARIABLE LISTING *****


   TIME          1 EPPLX       2 EPPLX       1 EPPLY       2 EPPLY
                 EPX_BISO      EPX_USER      EPY_BISO      EPY_USER
 0.10000       -0.472687E-01 -0.472687E-01  0.945374E-01  0.945374E-01
 0.28750       -0.125917     -0.125917      0.251834      0.251834
 0.45625       -0.187417     -0.187417      0.374835      0.374835
 0.66204       -0.253409     -0.253409      0.506818      0.506818
 0.89592       -0.319141     -0.319141      0.638282      0.638282
  1.0000       -0.345853     -0.345853      0.691707      0.691707
```

## 4. List of user subroutine USERMAT.F:

```
      subroutine usermat(
     &                   matId, elemId,kDomIntPt, kLayer, kSectPt,
     &                   ldstep,isubst,keycut,
     &                   nDirect,nShear,ncomp,nStatev,nProp,
     &                   Time,dTime,Temp,dTemp,
     &                   stress,statev,dsdePl,sedEl,sedPl,epseq,
     &                   Strain,dStrain, epsPl, prop, coords,
     &                   rotateM, defGrad_t, defGrad,
     &                   tsstif, epsZZ,
```

```
     &                   var1, var2, var3, var4, var5,
     &                   var6, var7, var8)
c*************************************************************************
c     *** primary function ***
c
c           user defined material constitutive model
c
c      Attention:
c           User must define material constitutive law properly
c           according to the stress state such as 3D, plain strain
c           and axisymmetry, plane stress and beam.
c
c           a 3D material constitutive model can be used for
c           plain strain and axisymmetric cases.
c
c           When using shell elements, the plane stress algorithm
c           must be used.
c
c       The following demonstrates a USERMAT subroutine for
c       a plasticity model in 3D stress state. The plasticity
c       model is the same as TB, BISO.
c       See "ANSYS user material subroutine USERMAT" for detailed
c       description of how to write a USERMAT routine.
c
c*************************************************************************
c
c     input arguments
c     ===============
c      matId     (int,sc,i)              material #
c      elemId    (int,sc,i)              element #
c      kDomIntPt (int,sc,i)              "k"th domain integration point
c      kLayer    (int,sc,i)              "k"th layer
c      kSectPt   (int,sc,i)              "k"th Section point
c      ldstep    (int,sc,i)              load step number
c      isubst    (int,sc,i)              substep number
c      nDirect   (int,sc,in)             # of direct components
c      nShear    (int,sc,in)             # of shear components
c      ncomp     (int,sc,in)             nDirect + nShear
c      nstatev   (int,sc,l)              Number of state variables
c      nProp     (int,sc,l)              Number of material ocnstants
c
c      Temp      (dp,sc,in)              temperature at beginning of
c                                        time increment
c      dTemp     (dp,sc,in)              temperature increment
c      Time      (dp,sc,in)              current time
c      dTime     (dp,sc,in)              current time increment
c
c      Strain   (dp,ar(ncomp),i)         Strain at beginning of time increment
c      dStrain  (dp,ar(ncomp),i)         Strain increment
c      prop     (dp,ar(nprop),i)         Material constants defined by TB,USER
c      coords   (dp,ar(3),i)             current coordinates
c      rotateM  (dp,ar(3,3),i)           Rotation matrix for finite deformation
update
c      defGrad_t(dp,ar(3,3),i)           Deformation gradient at time t
c      defGrad  (dp,ar(3,3),i)           Deformation gradient at time t+dt
c
c     input output arguments
```

```
c     ======================
c      stress   (dp,ar(nTesn),io)         stress
c      statev   (dp,ar(nstatev),io)       statev
c      sedEl    (dp,sc,io)                elastic work
c      sedPl    (dp,sc,io)                plastic work
c      epseq    (dp,sc,io)                equivalent plastic strain
c      var?     (dp,sc,io)                not used, they are reserved arguments
c                                         for further development
c
c     output arguments
c     ================
c      keycut   (int,sc,io)               loading bisect/cut control
c                                         0 - no bisect/cut
c                                         1 - bisect/cut
c                                         (factor will be determined by ANSYS
solution control)
c      dsdePl   (dp,ar(ncomp,ncomp),io)   material jacobian matrix
c
c*************************************************************************
c
c      ncomp   6   for 3D
c      ncomp   4   for plane strain/stress, axisymmetric
c      ncomp   1   for 1D
c
c      stresss and strains, plastic strain vectors
c          11, 22, 33, 12, 23, 13    for 3D
c          11, 22, 33, 12            for Plane strain/stress, axisymmetry
c          11                        for 1D
c
c      material jacobian matrix
c        3D
c           dsdePl    |  1111   1122   1133   1112   1123   1113 |
c           dsdePl    |  2211   2222   2233   2212   2223   2213 |
c           dsdePl    |  3311   3322   3333   3312   3323   3313 |
c           dsdePl    |  1211   1222   1233   1212   1223   1213 |
c           dsdePl    |  2311   2322   2333   2312   2323   2313 |
c           dsdePl    |  1311   1322   1333   1312   1323   1313 |
c        plane strain/stress, axisymmetric
c           dsdePl    |  1111   1122   1133   1112 |
c           dsdePl    |  2211   2222   2233   2212 |
c           dsdePl    |  3311   3322   3333   3312 |
c           dsdePl    |  1211   1222   1233   1212 |
c           for plane stress  entities in third colum/row are zero
c        1d
c           dsdePl    |  1111 |
c
c*************************************************************************
#include "impcom.inc"
c
      INTEGER
     &                 matId, elemId,
     &                 kDomIntPt, kLayer, kSectPt,
     &                 ldstep,isubst,keycut,
     &                 nDirect,nShear,ncomp,nStatev,nProp
      DOUBLE PRECISION
     &                 Time,    dTime,   Temp,    dTemp,
     &                 sedEl,   sedPl,   epseq,   epsZZ
```

```
      DOUBLE PRECISION
     &                 stress  (ncomp  ), statev  (nStatev),
     &                 dsdePl  (ncomp,ncomp),
     &                 Strain  (ncomp  ), dStrain (ncomp  ),
     &                 epsPl   (ncomp  ), prop    (nProp  ),
     &                 coords  (3),       rotateM (3,3),
     &                 defGrad_t(3,3),    defGrad(3,3),
     &                 tsstif  (2)
c
c***************** User defined part **********************************
c
c --- parameters
c
      INTEGER          NEWTON, mcomp
      DOUBLE PRECISION HALF, THIRD, ONE, TWO, SMALL, ONEHALF,
     &                 ZERO, TWOTHIRD, ONEDM02, ONEDM05, sqTiny
      PARAMETER       (ZERO       = 0.d0,
     &                 HALF       = 0.5d0,
     &                 THIRD      = 1.d0/3.d0,
     &                 ONE        = 1.d0,
     &                 TWO        = 2.d0,
     &                 SMALL      = 1.d-08,
     &                 sqTiny     = 1.d-20,
     &                 ONEDM02    = 1.d-02,
     &                 ONEDM05    = 1.d-05,
     &                 ONEHALF    = 1.5d0,
     &                 TWOTHIRD   = 2.0d0/3.0d0,
     &                 NEWTON     = 10,
     &                 mcomp      = 6
     &                 )
     c
     c --- local variables
     c
     c      sigElp   (dp,ar(6  ),l)           trial stress
     c      dsdeEl   (dp,ar(6,6),l)           elastic moduli
     c      sigDev   (dp,ar(6  ),l)           deviatoric stress tensor
     c      dfds     (dp,ar(6  ),l)           derivative of the yield function
     c      JM       (dp,ar(6,6),l)           2D matrix for a 4 order tensor
     c      pEl      (dp,sc     ,l)           hydrostatic pressure stress
     c      qEl      (dp,sc     ,l)           von-mises stress
     c      pleq_t   (dp,sc     ,l)           equivalent plastic strain at
     c                                        beginnig of time increment
     c      pleq     (dp,sc     ,l)           equivalent plastic strain at end
     c                                        of time increment
     c      dpleq    (dp,sc     ,l)           incremental equivalent plastic
     strain
     c      cpleq    (dp,sc     ,l)           correction of incremental
     c                                        equivalent plastic strain
     c      sigy_t   (dp,sc     ,l)           yield stress at beginnig of time
     increments
     c      sigy     (dp,sc     ,l)           yield stress at end of time
     c                                        increment
     c      young    (dp,sc     ,l)           Young's modulus
     c      posn     (dp,sc     ,l)           Poiss's ratio
     c      sigy0    (dp,sc     ,l)           initial yield stress
     c      dsigdep  (dp,sc     ,l)           plastic slope
     c      twoG     (dp,sc     ,l)           two time of shear moduli
```

```
c      threeG   (dp,sc     ,l)           three time of shear moduli
c      funcf    (dp,sc     ,l)           nonlinear function to be solved
c                                         for dpleq
c      dFdep    (dp,sc     ,l)           derivative of nonlinear function
c                                         over dpleq
c
c --- temporary variables for solution purpose
c      i, j
c      threeOv2qEl, oneOv3G, qElOv3G, con1, con2, fratio
c
       DOUBLE PRECISION sigElp(mcomp), dsdeEl(mcomp,mcomp), G(mcomp),
      &                 sigDev(mcomp), JM    (mcomp,mcomp), dfds(mcomp)

       DOUBLE PRECISION var1, var2, var3, var4, var5,
      &                 var6, var7, var8, var9, var10

       DATA G/1.0D0,1.0D0,1.0D0,0.0D0,0.0D0,0.0D0/
c
       INTEGER          i, j
       DOUBLE PRECISION pEl,   qEl,     pleq_t,  sigy_t , sigy,
      &                 cpleq, dpleq,   pleq,
      &                 young, posn,    sigy0,   dsigdep,
      &                 elast1,elast2,
      &                 twoG,  threeG,  oneOv3G, qElOv3G, threeOv2qEl,
      &                 funcf, dFdep,   fratio,  con1,    con2
c*********************************************************************
c
       keycut   = 0
       dsigdep  = ZERO
       pleq_t   = statev(1)
       pleq     = pleq_t
c *** get Young's modulus and Poisson's ratio, initial yield stress and others
       young    = prop(1)
       posn     = prop(2)
       sigy0    = prop(3)
c *** calculate the plastic slope
       dsigdep  = young*prop(4)/(young-prop(4))
       twoG     = young / (ONE+posn)
       threeG   = ONEHALF * twoG
c
c *** calculate elastic stiffness matrix (3d)
c
c
       elast1=young*posn/((1.0D0+posn)*(1.0D0-TWO*posn))
       elast2=young/(TWO*(1.0D0+posn))
       dsdeEl(1,1)=(elast1+TWO*elast2)*G(1)*G(1)
       dsdeEl(1,2)=elast1*G(1)*G(2)+elast2*TWO*G(4)*G(4)
       dsdeEl(1,3)=elast1*G(1)*G(3)+elast2*TWO*G(5)*G(5)
       dsdeEl(1,4)=elast1*G(1)*G(4)+elast2*TWO*G(1)*G(4)
       dsdeEl(1,5)=elast1*G(1)*G(5)+elast2*TWO*G(1)*G(5)
       dsdeEl(1,6)=elast1*G(1)*G(6)+elast2*TWO*G(4)*G(5)
       dsdeEl(2,2)=(elast1+TWO*elast2)*G(2)*G(2)
       dsdeEl(2,3)=elast1*G(2)*G(3)+elast2*TWO*G(6)*G(6)
       dsdeEl(2,4)=elast1*G(2)*G(4)+elast2*TWO*G(1)*G(4)
       dsdeEl(2,5)=elast1*G(2)*G(5)+elast2*TWO*G(1)*G(5)
       dsdeEl(2,6)=elast1*G(2)*G(6)+elast2*TWO*G(2)*G(6)
       dsdeEl(3,3)=(elast1+TWO*elast2)*G(3)*G(3)
```

```
      dsdeEl(3,4)=elast1*G(3)*G(4)+elast2*TWO*G(5)*G(6)
      dsdeEl(3,5)=elast1*G(3)*G(5)+elast2*TWO*G(5)*G(3)
      dsdeEl(3,6)=elast1*G(3)*G(6)+elast2*TWO*G(6)*G(3)
      dsdeEl(4,4)=elast1*G(4)*G(4)+elast2*(G(1)*G(2)+G(4)*G(4))
      dsdeEl(4,5)=elast1*G(4)*G(5)+elast2*(G(1)*G(6)+G(5)*G(4))
      dsdeEl(4,6)=elast1*G(4)*G(6)+elast2*(G(4)*G(6)+G(5)*G(2))
      dsdeEl(5,5)=elast1*G(5)*G(5)+elast2*(G(1)*G(3)+G(5)*G(5))
      dsdeEl(5,6)=elast1*G(5)*G(6)+elast2*(G(4)*G(3)+G(5)*G(6))
      dsdeEl(6,6)=elast1*G(6)*G(6)+elast2*(G(2)*G(3)+G(6)*G(6))
      do i=1,ncomp-1
        do j=i+1,ncomp
          dsdeEl(j,i)=dsdeEl(i,j)
        end do
      end do
c
c *** calculate the trial stress and
c     copy elastic moduli dsdeEl to material Jacobian matrix
      do i=1,ncomp
         sigElp(i) = stress(i)
         do j=1,ncomp
            dsdePl(j,i) = dsdeEl(j,i)
            sigElp(i) = sigElp(i)+dsdeEl(j,i)*dStrain(j)
         end do
      end do
c *** hydrostatic pressure stress
      pEl = -THIRD * (sigElp(1) + sigElp(2) + sigElp(3))
c *** compute the deviatoric stress tensor
      sigDev(1) = sigElp(1) + pEl
      sigDev(2) = sigElp(2) + pEl
      sigDev(3) = sigElp(3) + pEl
      sigDev(4) = sigElp(4)
      sigDev(5) = sigElp(5)
      sigDev(6) = sigElp(6)
c *** compute von-mises stress
      qEl =
     &  sigDev(1) * sigDev(1)+sigDev(2) * sigDev(2)+
     &  sigDev(3) * sigDev(3)+
     &  TWO*(sigDev(4) * sigDev(4)+ sigDev(5) * sigDev(5)+
     &  sigDev(6) * sigDev(6))
      qEl = sqrt( ONEHALF * qEl)
c *** compute current yield stress
      sigy    = sigy0 + dsigdep * pleq
c
      fratio = qEl / sigy - ONE
c *** check for yielding
      IF (sigy .LE. ZERO.or.fratio .LE. -SMALL) GO TO 500
c
      sigy_t = sigy
      threeOv2qEl = ONEHALF / qEl
c *** compute derivative of the yield function
      DO i=1, ncomp
         dfds(i) = threeOv2qEl * sigDev(i)
      END DO
      oneOv3G  = ONE / threeG
      qElOv3G  = qEl * oneOv3G
c *** initial guess of incremental equivalent plastic strain
      dpleq    = (qEl - sigy) * oneOv3G
```

```
      pleq     = pleq_t + dpleq
c
c *** Newton-Raphosn procedure for return mapping iteration
      DO i = 1,NEWTON
         sigy   = sigy0 + dsigdep * pleq
         funcf  = qElOv3G -  dpleq - sigy * oneOv3G
         dFdep  = - ONE  - dsigdep * oneOv3G
         cpleq  = -funcf / dFdep
         dpleq  = dpleq + cpleq
c        --- avoid negative equivalent plastic strain
         dpleq  = max (dpleq, sqTiny)
         pleq   = pleq_t + dpleq
         fratio = funcf/qElOv3G
c ***    check covergence
         IF (((abs(fratio) .LT. ONEDM05        )  .AND.
     &        (abs(cpleq ) .LT. ONEDM02 * dpleq)) .OR.
     &       ((abs(fratio) .LT. ONEDM05        )  .AND.
     &        (abs(dpleq ) .LE. sqTiny         )))  GO TO 100
      END DO
c
c *** Uncovergence, set keycut to 1 for bisect/cut
      keycut   = 1
      GO TO 990
 100  CONTINUE
c
c ***  update stresses
      con1 =  twoG * dpleq
      DO i = 1 , ncomp
         stress(i) =  sigElp(i) - con1 * dfds(i)
      END DO
c
c ***  update plastic strains
      DO i = 1 , nDirect
         epsPl(i) = epsPl(i) + dfds(i) * dpleq
      END DO
      DO i = nDirect + 1 , ncomp
         epsPl(i) = epsPl(i) + TWO * dfds(i) * dpleq
      END DO
      epseq  = pleq
c *** Update state variables
      statev(1) = pleq
c *** Update plastic work
      sedPl = sedPl + HALF * (sigy_t+sigy)*dpleq
c
c *** Material Jcobian matrix
c
      IF (qEl.LT.sqTiny) THEN
         con1 = ZERO
      ELSE
         con1 = threeG * dpleq / qEl
      END IF
      con2 = threeG/(threeG+dsigdep) - con1
      con2 = TWOTHIRD * con2
      DO i=1,ncomp
         DO j=1,ncomp
            JM(j,i) = ZERO
         END DO
```

```
      END DO
      DO i=1,nDirect
         DO j=1,nDirect
            JM(i,j) = -THIRD
         END DO
         JM(i,i) = JM(i,i) + ONE
      END DO
      DO i=nDirect + 1,ncomp
         JM(i,i) = HALF
      END DO
      DO i=1,ncomp
         DO j=1,ncomp
            dsdePl(i,j) =    dsdeEl(i,j) - twoG
     &           * (  con2 * dfds(i) * dfds(j) + con1 * JM(i,j) )
         END DO
      END DO
c
      goto 600
  500 continue

c *** Update stress in case of elastic/unloading
      do i=1,ncomp
         stress(i) = sigElp(i)
      end do

  600 continue
c *** Claculate elastic work
      sedEl = ZERO
      DO i = 1 , ncomp
         sedEl = sedEl + stress(i)*(Strain(i)+dStrain(i)-epsPl(i))
      END DO
      sedEl    = sedEl * HALF
c
 990  CONTINUE
c
      return
      end
```