

1N-39  
79762  
P-104

NASA Contractor Report 189091

# A Finite Element Program for Postbuckling Calculations (PSTBKL)

G.T. Simitzes, R.L. Carlson, and R. Riff  
*Georgia Institute of Technology*  
*Atlanta, Georgia*

December 1991

Prepared for  
Lewis Research Center  
Under Grant NAG3-534



(NASA-CR-189091) A FINITE ELEMENT PROGRAM  
FOR POSTBUCKLING CALCULATIONS (PSTBKL) Final  
Report (Georgia Inst. of Tech.) 104 p  
CSCS 20K

N92-20584  
Unclas  
G3/39 0079762

# A Finite Element Program for Postbuckling Calculations (PSTBKL)

G.T. Simitzes, R.L. Carlson, and R. Riff  
Georgia Institute of Technology  
Atlanta, Georgia 30332

## ABSTRACT

The object of the research reported herein was to develop a general mathematical model and solution methodologies for analyzing the structural response of thin, metallic shell structures under large transient, cyclic, or static thermomechanical loads. Among the system responses associated with these loads and conditions are thermal buckling, creep buckling, and ratcheting. Thus geometric and material nonlinearities (of high order) can be anticipated and must be considered in developing the mathematical model. The methodology is demonstrated through different problems of extension, shear and of planar curved beam. Moreover, importance of the inclusion of large strains is clearly demonstrated, through the chosen applications. This report describes the computer program resulting from the research.

## Introduction

Program PSTBKL is developed to study the thermo-elastoviscoplastic postbuckling behavior of shell-like structures. The main features of the program include:

1. Buckling and postbuckling predictions of shell-like structures
2. Response of the structure at elevated temperatures
3. Creep buckling predictions
4. Freedom to choose different thermo-mechanical loading path
5. Bodner-Partom's constitutive equations as an elastoviscoplastic material model
6. Walker's constitutive equations as another elastoviscoplastic material model
7. Nonlinear elastic calculations
8. Crisfield's iteration schemes for limit point load problems
9. Tanaka-Miller's method used to integrate the unified constitutive equations

The program works for material B1900+Hf now. With minor change, it can work for other materials.

## Input Format

File DT is the main input data file. File RD is used only when the program needs to resume a unfinished job. File RD can be copied from file WRT which is an output file in the last execution.

The format of file DT is the following:

- (1). Control data (lines 1 through 8)

Line 1: I1, I2, I3, I4

I1—number of elements, I2—number of nodes, I3—number of steps planed to run, I4—maximum number of iterations allowed in each load step

Line 2: A1, A2, A3, A4, A5, A6

A1—elastic modulus of the material, A2—Poisson's ratio, A3—thickness of the structure, A4—load coefficient (take 1.0), A5—load coefficient (take 1.0), A6—initial load step (take 1.0, not use now)

Line 3: I1, I2, I3

I1—the node number of the output displacement, I2—the component of the output displacement, I3—the control variable

Line 4: I1, I2, A1, A2

I1—determine whether the execution from the beginning (choose 0) or from the last execution (choose 1), I2—number of loading steps before the program write data for further execution, A1—the displacement increment of control variable, A2—the increase rate of A1 in next step (take 1.0 generally)

Line 5: I1, I2, I3, I4, A1, A2, A3, A4

I1—determine whether the thermal expansion is considered (take 1) or not (take 0), I2—number of steps for the change of temperature, I3—number of iterations executed before writing temporary data, I4—maximum number of iterations allowed in the equilibrium iterations, A1—thermal expansion coefficient, A2—initial temperature, A3—increment of temperature, A4—highest temperature

Line 6: I1, I2, A1, A2

I1—option whether to use unified constitutive equations (1 for yes, 0 for no), I2—option of which constitutive model to use (1 for Bodner-Partom's model and 2 for Walker's model), A1—calculation coefficient (take 1.0), A2—the increment of time in a load interval

Line 7: I1, I2

I1—option for creep calculation (1 for yes and 0 for no), I2—number of steps beyond which creep is calculated

Line 8: I1, I2

I1 and I2 are used to control the output of the calculated results. The value of I1 can be an integer from 1 to 6 which correspond to the stretch of bar, plate, cylindrical shell under axial compression, cylindrical shell under pressure and cylindrical shell under torsion. I2 controls the way of output (see NTV in subroutine OUTPUT).

(2). Initial nodal coordinates

format: I1, A1, A2, A3

I1—node number (it does not matter whatever to write, but the real nodal number must in order of 1, 2, 3...), A1—X, A2—Y, A3—Z

(3). Constraint specification

format: I1, I2, I3, I4, I5, I6

I1—node number, I2—displacement in x direction, I3—displacement in y direction, I4—displacement in z direction, I5—rotation along local x axis, I6—rotation along local y axis (0 for free movement and 1 for constraint)

(4). Applied load

format: I1, A1, A2, A3, A4, A5

I1—node number, A1—load applied in x direction, A2—load applied in y direction, A3—load applied in z direction, A4—moment applied in local x direction, A5—moment applied in local y direction

(5). Element and its corresponded nodes

format: I1, I2, I3, I4, I5, I6, I7, I8, I9

I1—element number, I2 through I9—the node number of the element

(6). Direction cosines of the structure

format: I1, A1, A2, A3

I1—element number, A1 through A3—the initial direction cosines of local coordinates to global coordinates at position of the node

(7). Radius and length of the shell

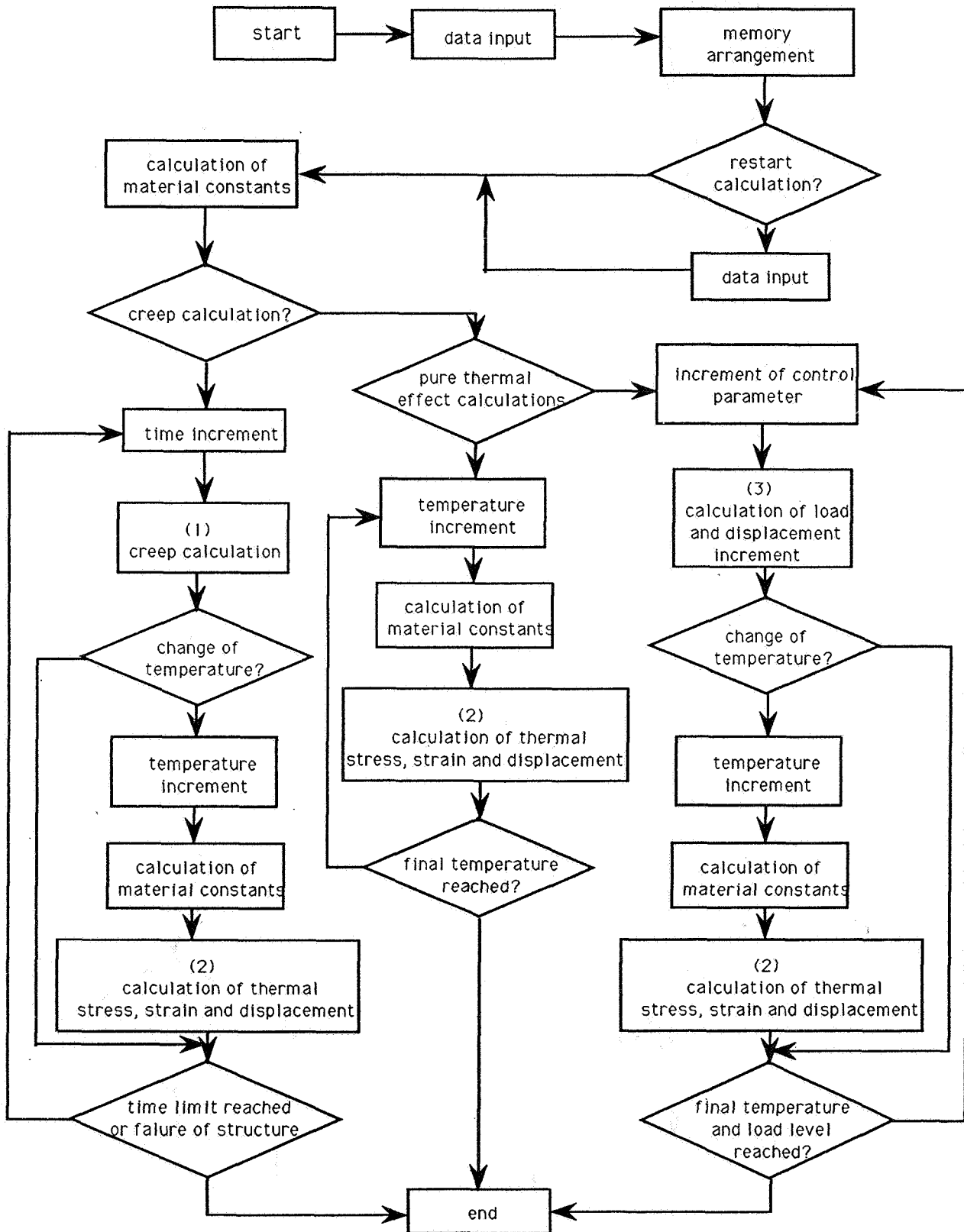
## Output Files

The output files are WRT, OUT, OT, OUT1, OT2 and OUT3. File WRT contains the necessary data for further execution. File OUT is the data used to locate any problem occurred during execution. Files OT, OT1, OT2 and OT3 are output files for the calculated results controlled by subroutine OUTPUT. In the subroutine, D1(I,J) is the displacement matrix where I and J are the nodal number and displacement component number, respectively. The updated coordinates of node I are XX(I), YY(I) and ZZ(I). The corresponding load can be calculated as the product of TROOT (a variable in the subroutine), load coefficient and the applied load (given in file DT). Files OT, OT1, OT2 and OT3 are associated with tape 3, 9, 11 and 12. Users can change subroutine OUTPUT to get desired output.

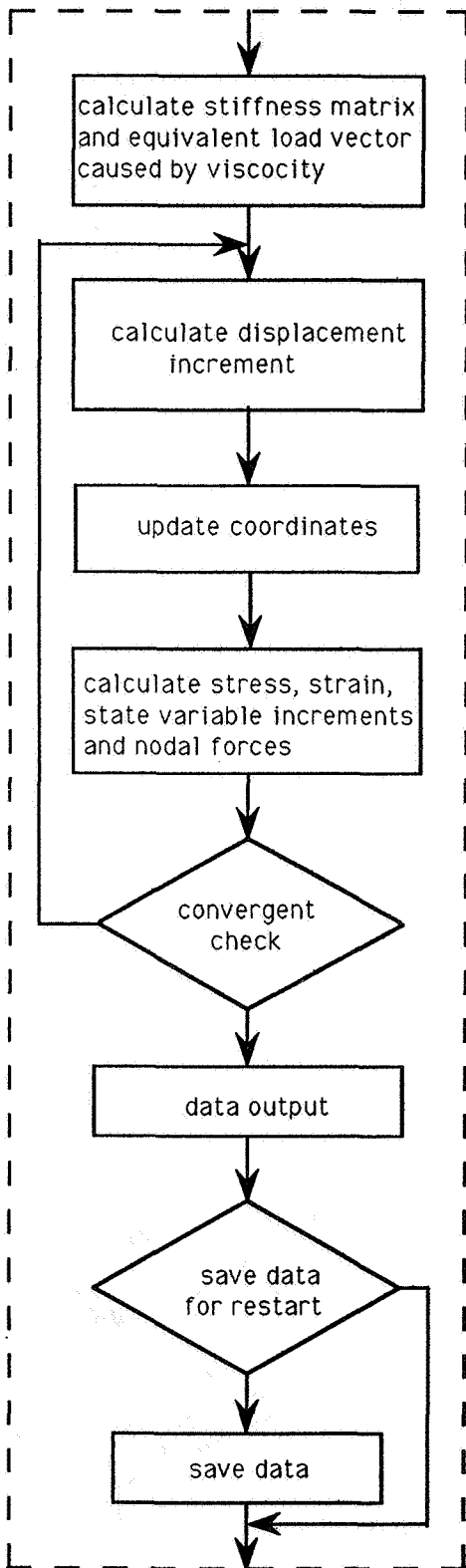
### Subroutines from Library

The subroutine LINRG from software IMSL is called in the program to invert the stiffness matrix. The corresponding version in Cyber is LINV3F.

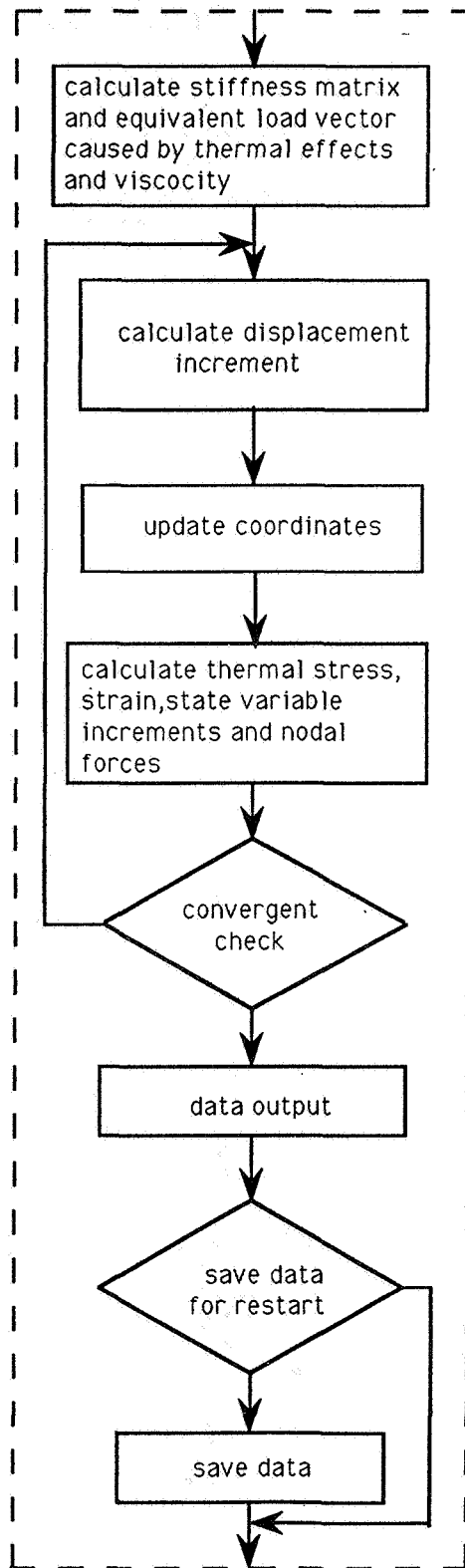
# Main Flowchart



(1). creep calculation

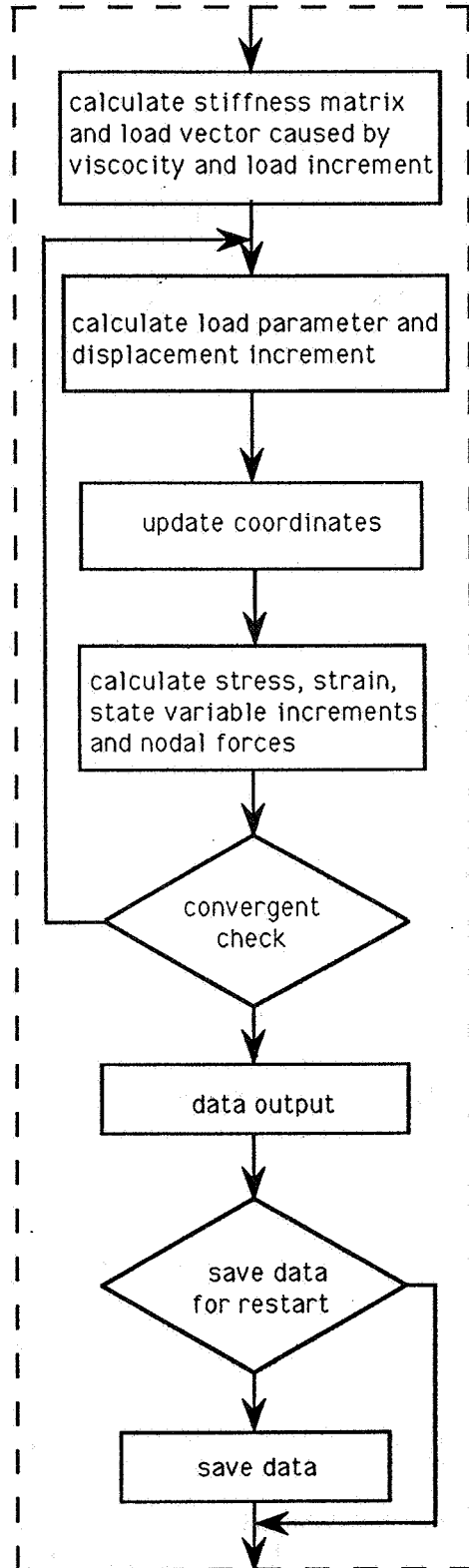


(2). thermal effects calculation





(1). calculation of load and displacement increment



```

C*****C
C   Program pstbkl is for the postbuckling analysis with either C
C   Bodner-Partom's or Walker's material model. The program can C
C   deals with the following problems: C
C   1. Postbuckling responses of thin-walled structures under C
C       normal loading C
C   2. Creep buckling analysis C
C   3. Thermal effects C
C*****C

```

```

C
C   PROGRAM PSTBKL
C   IMPLICIT REAL*8 (A-H,O-Z)
C   IMPLICIT INTEGER*8 (I-N)
C   PARAMETER (MAXR=150000,MAXI=5000)
C   DIMENSION RWKSP (100000)
C   COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
C   COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1      IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2      IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3      IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4      IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5      IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
C   COMMON /RLVEC/ VR (MAXR)
C   COMMON /INTVEC/ IPT (MAXI)
C   COMMON /WORKSP/ RWKSP

```

```

C
C   If the program is used in cyber, active 1r41=1r23 statement.
C

```

```

C
C   OPEN (3,FILE='ot')
C   OPEN (4,FILE='rd')
C   OPEN (5,FILE='dt')
C   OPEN (6,FILE='out')
C   OPEN (7,FILE='wrt')
C   OPEN (9,FILE='ot1')
C   OPEN (11,FILE='ot2')
C   OPEN (12,FILE='ot3')

```

```

C
C   CALL CMPT1
C
C   Call cmpt1 to make initial memory arangement
C

```

```

C
C   CALL IWKIN (100000)
C
C   IWKIN is used to set work space for subroutine LINRG wich is
C   given in IMSL library.
C

```

```

C
C   CALL PREPC (IPT (IP1),IPT (IP2),IPT (IP3),VR (IR1),VR (IR2),
1      VR (IR3),VR (IR4),VR (IR5),VR (IR6),VR (IR7))
C
C   STOP
C   END

```

```

C
C   Subroutine PREPC is used to read input data and make memory
C   arrangement
C

```

```

C
C   SUBROUTINE PREPC (IEL, ID, IID, XX, YY, ZZ, DD1, DD2, DLOAD, HORIZ)
C   IMPLICIT REAL*8 (A-H,O-Z)
C   IMPLICIT INTEGER*8 (I-N)
C   DIMENSION IEL (NELM,8), ID (1), IID (NNODE,5), XX (1), YY (1), ZZ (1),
1      DD1 (1), DD2 (1), DLOAD (1), HORIZ (1)
C
C   COMMON /SCHALR1/ NELM,NNODE,NT
C   COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,
1      NSHOW3,HRZ,ITRLM,FACTOR
C   COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10

```

```

COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3 IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4 IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5 IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50

```

```

COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
COMMON /DIRCS/ IR60,IR61,IR62,IR63,IR64,IR65
COMMON /DISV1/ IR70,IR71,IR72,IR73,IR74,IR75

```

```

COMMON /UNICT/ NCONS,MODEL,ETAA,TDELTA,TINIT
COMMON /RLVEC/ VR(1)
COMMON /INTVEC/ IPT(1)

```

```

CALL GETDT (IPT (IP1) , IPT (IP2) , IPT (IP3) , IPT (IP4) , IPT (IP5) ,
1 IPT (IP6) , IPT (IP7) , IPT (IP8) , VR (IR1) , VR (IR2) , VR (IR3) ,
2 VR (IR4) , VR (IR5) )

```

Call GETDT to read data. Call CMPT2 to make memory arrangement.  
Call RDSUP to get further data input.

```

CALL CMPT2
CALL RDSUP (VR (IR60) , VR (IR61) , VR (IR62) , VR (IR63) , VR (IR64) , VR (IR65) ,
1 VR (IR75) )

```

```

CALL PROCS (VR (IR6) , VR (IR4) , VR (IR5) , VR (IR9) , VR (IR27) , VR (IR20) ,
1 VR (IR43) , VR (IR44) , VR (IR45) , VR (IR1) , VR (IR2) , VR (IR3) ,
1 VR (IR47) , VR (IR42) , VR (IR10) , VR (IR51) , VR (IR58) , VR (IR39) )

```

```

CLOSE (3)
CLOSE (4)
CLOSE (5)
CLOSE (6)
CLOSE (7)
CLOSE (9)
CLOSE (11)
CLOSE (12)
RETURN
END

```

Subroutine procs is used to arrange the loading scheme, so that  
the normal loading, creep and temperature effects can be considered  
either simultaneously or separately.

```

SUBROUTINE PROCS (DLOAD,DD1,DD2,PLD,ACMDIS,SIGMA,XX1,YY1,ZZ1,
1 XX,YY,ZZ,UPSIG,FRCINC,FRCO,BETA,UPBET,EM)
IMPLICIT REAL*8 (A-H,O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION DLOAD (1) , DD1 (1) , DD2 (1) , PLD (1) , ACMDIS (1) ,
1 SIGMA (NELM,2,2,2,9) , XX (1) , YY (1) , ZZ (1) , XX1 (1) , YY1 (1) ,
2 ZZ1 (1) , UPSIG (NELM,2,2,2,9) , FRCINC (1) , FRCO (1) ,
3 BETA (NELM,2,2,2,12) , UPBET (NELM,2,2,2,12) , EM (6,6)

```

```

COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,
1 NSHOW3,HRZ,ITRLM,FACTOR
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3 IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4 IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5 IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
COMMON /DIRCS/ IR60,IR61,IR62,IR63,IR64,IR65

```

```

COMMON /DISVC/ IR66,IR67,IR68,IR69
COMMON /DISV1/ IR70,IR71,IR72,IR73,IR74,IR75
COMMON /UNICT/ NCONS,MODEL,ETAA,TDEL,TINIT
COMMON /RLVEC/ VR(1)
COMMON /INTVEC/ IPT(1)
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
COMMON /CNTRL/ DETMNT
COMMON /CONTN/ INSIDT,KPDT,DTLM1
COMMON /ABDFST/ ISEC
COMMON /SQ/ SQQ
COMMON /NMBITR/ NUM
COMMON /DISCT/ NDC,NDBC
COMMON /CREEP/ ICRP,NBCRP,NBDN,CRPTM,IPON
COMMON /TMPEF/ IDO,NTEM,NITR,NANM,CEXPN,TMMIN,TMINC,TMMAX,TMPP
C
DO 10 I=1,NNODE
  XX1(I)=XX(I)
  YY1(I)=YY(I)
  ZZ1(I)=ZZ(I)
10 CONTINUE
C
  IF (INSIDT.EQ.1) THEN
C    If the execution is based on the previous calculation, get
C    additional information
    CALL RDCDT (VR (IR27),VR (IR20),VR (IR43),VR (IR44),VR (IR45),
1      VR (IR1),VR (IR2),VR (IR3),VR (IR47),VR (IR10),
2      VR (IR51),VR (IR58),VR (IR60),VR (IR61),VR (IR62),
3      VR (IR63),VR (IR64),VR (IR65),VR (IR15),VR (IR71),
4      VR (IR75))
    END IF
C
    DO 200 J=1,NT
      DLOAD (J)=DD1 (J)*COEF1
200 CONTINUE
      ROOT=0.0
      DTLAM=FACTOR
      ROOT=ROOT+DTLAM
      SGN=1.0
      ISEC=1
C
C    Calculate material constants according to the chosen model
C
    IF (IDO.EQ.0) THEN
      TMPP=TMMIN
      IF (NCONS.EQ.0) THEN
        E=198700.0+16.78*TMPP-0.1034*TMPP*TMPP
1      +0.00001143*TMPP*TMPP*TMPP
      ELSE
        IF (MODEL.EQ.1) CALL BDCNS (TMPP)
        IF (MODEL.EQ.2) CALL WKCNS (TMPP)
      END IF
    END IF
C
C    Calculate the elastic matrix
C
    CALL ELSMTR (EM)
C
C
    DO 220 J=1,NT
      DLOAD (J)=DD2 (J)*COEF2
      PLD (J)=0.0
220 CONTINUE
C
C    Next iteration is to calculate the thermal effect
C
    IF (IDO.EQ.1) THEN

```

```

DO 205 I=1,NTEM
  NUM=1
  TMPP=TMINC+TMPP
  IF (NCONS.EQ.0) THEN
    E=198700.0+16.78*TMPP-0.1034*TMPP*TMPP
1    +0.00001143*TMPP*TMPP*TMPP
  ELSE
    IF (MODEL.EQ.1) CALL BDCNS (TMPP)
    IF (MODEL.EQ.2) CALL WKCNS (TMPP)
  END IF

C
  IF (TMPP.GT.TMMAX) THEN
    WRITE (6,*) 'THE MAXIMAM LIMIT OF TEMPERATURE IS REACHED, STOP'
    STOP
  END IF

C
  CALL THRML (1, IPT (IP1), IPT (IP2), IPT (IP3), IPT (IP4), IPT (IP5),
1    IPT (IP9), VR (IR1), VR (IR2), VR (IR3), VR (IR6), VR (IR8),
2    VR (IR9), VR (IR10), VR (IR11), VR (IR12), VR (IR13), VR (IR14),
3    VR (IR15), VR (IR16), VR (IR17), VR (IR21), VR (IR22), VR (IR23),
4    VR (IR24), VR (IR18), VR (IR26), VR (IR27), VR (IR42), VR (IR43),
5    VR (IR44), VR (IR45), VR (IR46), VR (IR47), VR (IR20), VR (IR48),
6    VR (IR49), VR (IR19), VR (IR50), VR (IR51), VR (IR58), VR (IR59),
7    VR (IR60), VR (IR61), VR (IR62), VR (IR63), VR (IR64), VR (IR65),
8    VR (IR4))
205 CONTINUE
  END IF

C
C Next iteration is to calculate creep responses (with or without
C thermal effects) or the normal loading responses (with or without
C thermal effects)
C
DO 900 I=1,NSTEP
  ROOT=0.0
  NUM=1
  IF (NBDN.GT.NBCRP.AND.ICRP.EQ.1) THEN
    CALL NTCRP (1, IPT (IP1), IPT (IP2), IPT (IP3), IPT (IP4), IPT (IP5),
1    IPT (IP9), VR (IR1), VR (IR2), VR (IR3), VR (IR6), VR (IR8),
2    VR (IR9), VR (IR10), VR (IR11), VR (IR12), VR (IR13), VR (IR14),
3    VR (IR15), VR (IR16), VR (IR17), VR (IR21), VR (IR22), VR (IR23),
4    VR (IR24), VR (IR18), VR (IR26), VR (IR27), VR (IR42), VR (IR43),
5    VR (IR44), VR (IR45), VR (IR46), VR (IR47), VR (IR20), VR (IR48),
6    VR (IR49), VR (IR19), VR (IR50), VR (IR51), VR (IR58), VR (IR59),
7    VR (IR60), VR (IR61), VR (IR62), VR (IR63), VR (IR64), VR (IR65),
8    VR (IR66), VR (IR67), VR (IR68), VR (IR69), VR (IR71), VR (IR72),
9    VR (IR73), VR (IR75), VR (IR74))
  ELSE
    CALL ARCLS (1, IPT (IP1), IPT (IP2), IPT (IP3), IPT (IP4), IPT (IP5),
1    IPT (IP9), VR (IR1), VR (IR2), VR (IR3), VR (IR6), VR (IR8),
2    VR (IR9), VR (IR10), VR (IR11), VR (IR12), VR (IR13), VR (IR14),
3    VR (IR15), VR (IR16), VR (IR17), VR (IR21), VR (IR22), VR (IR23),
4    VR (IR24), VR (IR18), VR (IR26), VR (IR27), VR (IR42), VR (IR43),
5    VR (IR44), VR (IR45), VR (IR46), VR (IR47), VR (IR20), VR (IR48),
6    VR (IR49), VR (IR19), VR (IR50), VR (IR51), VR (IR58), VR (IR59),
7    VR (IR60), VR (IR61), VR (IR62), VR (IR63), VR (IR64), VR (IR65),
8    VR (IR66), VR (IR67), VR (IR68), VR (IR69), VR (IR71), VR (IR72),
9    VR (IR73), VR (IR75), VR (IR74))

C
  END IF
  IF (IDO.EQ.2) THEN
    TMPP=TMINC+TMPP
    IF (NCONS.EQ.0) THEN
      E=198700.0+16.78*TMPP-0.1034*TMPP*TMPP
1    +0.00001143*TMPP*TMPP*TMPP
    ELSE
      IF (MODEL.EQ.1) CALL BDCNS (TMPP)

```

```

      IF (MODEL.EQ.2) CALL WKCNS (TMPP)
      END IF
      CALL THRML (1, IPT (IP1), IPT (IP2), IPT (IP3), IPT (IP4), IPT (IP5),
1         IPT (IP9), VR (IR1), VR (IR2), VR (IR3), VR (IR6), VR (IR8),
2         VR (IR9), VR (IR10), VR (IR11), VR (IR12), VR (IR13), VR (IR14),
3         VR (IR15), VR (IR16), VR (IR17), VR (IR21), VR (IR22), VR (IR23),
4         VR (IR24), VR (IR18), VR (IR26), VR (IR27), VR (IR42), VR (IR43),
5         VR (IR44), VR (IR45), VR (IR46), VR (IR47), VR (IR20), VR (IR48),
6         VR (IR49), VR (IR19), VR (IR50), VR (IR51), VR (IR58), VR (IR59),
7         VR (IR60), VR (IR61), VR (IR62), VR (IR63), VR (IR64), VR (IR65),
9         VR (IR4))

      DO 221 J=1,NT
          DLOAD (J)=DD2 (J)*COEF2
          PLD (J)=0.0
221      CONTINUE
      END IF
900      CONTINUE

      RETURN
      END

C
C
C      Subroutine ARCLS is used for normal loading calculation.
C      Arc-length method is used in the iteration scheme.
C
      SUBROUTINE ARCLS (INUM, IEL, ID, IID, L, MAXA, LD, XX, YY, ZZ, DLOADT, D,
1         PLD, FRCO, DD, DLDINC, VTEMP, VF, D1, VFE, DDD, AM, PD,
2         P, A, TDLD, HISINC, ACMDIS, FRCINC, XX1, YY1, ZZ1, DELTA,
3         UPSIG, SIGMA, DLTINC, DLTTMP, STIFFN, EXLVC, BETA, UPBET,
4         ACTFRC, GCL1, GCL2, GCL3, UCL1, UCL2, UCL3, ADC, ADD, AD,
5         ADVC, TLTY, TY1, TY2, ANGL, DBVC)
      IMPLICIT REAL*8 (A-H, O-Z)
      IMPLICIT INTEGER*8 (I-N)

      DIMENSION IEL (NELM, 8), ID (1), IID (NNODE, 5), L (1), MAXA (1), LD (1)
      DIMENSION XX (1), YY (1), ZZ (1), DD (NNODE, 5), D (1), PLD (1),
1         DLOADT (1), DLDINC (1), VTEMP (1), VF (NNODE, 5),
2         D1 (NNODE, 5), VFE (NT, 1), DDD (1), VRT (4),
3         A (NEQT, NEQT), AM (40, 40), PD (1), TDLD (1),
4         HISINC (1), ACMDIS (1), FRCINC (1), XX1 (1), YY1 (1), ZZ1 (1),
5         DELTA (1), FRCO (1), UPSIG (NELM, 2, 2, 2, 9), ACTFRC (1),
6         SIGMA (NELM, 2, 2, 2, 9), DLTINC (1), DLTTMP (1), COEEQ (5),
7         DEFVRT (4), STIFFN (NT, NT), ETT (4), EXLVC (1),
8         BETA (NELM, 2, 2, 2, 12), UPBET (NELM, 2, 2, 2, 12), GCL1 (NNODE, 3),
9         GCL2 (NNODE, 3), GCL3 (NNODE, 3), UCL1 (NNODE, 3),
1        UCL2 (NNODE, 3), UCL3 (NNODE, 3), ADC (NDBC, NDBC),
2        ADD (NDBC, NEQT), AD (NEQT, NDBC), ADVC (1), TLTY (1), TY1 (1),
3        TY2 (1), ANGL (1), DBVC (1)

      COMMON /SCHALR1/ NELM, NNODE, NT
      COMMON /SCHALR2/ NEQT, NSTEP, NHBW, COEF1, COEF2, NSHOW1, NSHOW2,
1         NSHOW3, HRZ, ITRLM, FACTOR
      COMMON /PNTRIN/ IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10
      COMMON /PNTRRL/ IR1, IR2, IR3, IR4, IR5, IR6, IR7, IR8, IR9, IR10,
1         IR11, IR12, IR13, IR14, IR15, IR16, IR17, IR18,
2         IR19, IR20, IR21, IR22, IR23, IR24, IR25, IR26,
3         IR27, IR28, IR29, IR30, IR31, IR32, IR33, IR34,
4         IR35, IR36, IR37, IR38, IR39, IR40, IR41, IR42,
5         IR43, IR44, IR45, IR46, IR47, IR48, IR49, IR50
      COMMON /UNIFBD/ IR51, IR52, IR53, IR54, IR55, IR56, IR57, IR58, IR59
      COMMON /DIRCS/ IR60, IR61, IR62, IR63, IR64, IR65
      COMMON /DISCT/ NDC, NDBC
      COMMON /DISVC/ IR66, IR67, IR68, IR69
      COMMON /DISV1/ IR70, IR71, IR72, IR73, IR74, IR75
      COMMON /UNICT/ NCONS, MODEL, Etaa, TDELTA, TINIT

```

```

C   Begin iteration
C
C   III=1
C
C   CALL MNU(NNODE,5,VF)
C   DO 200 I=1,NT
C     DLDINC(I)=DLOADT(I)
200 CONTINUE
C
C   DO 195 I=1,ND
C     TDLD(I)=0.0
C     HISINC(I)=0.0
195 CONTINUE
210 FORMAT('I,LDINC,LOADT,PLD IS',113,3F8.3)
579 CONTINUE
C
C   Call ASSMBL is to form the stiffness matrix
C
C     CALL ASSMBL(III,IPT(IP1),IPT(IP2),IPT(IP3),IPT(IP4),IPT(IP5),
1     IPT(IP9),VR(IR1),VR(IR2),VR(IR3),VR(IR6),VR(IR8),
2     VR(IR12),VR(IR14),VR(IR15),VR(IR16),VR(IR19),VR(IR21),
3     VR(IR23),VR(IR24),VR(IR19),VR(IR41),VR(IR50),VR(IR52),
4     VR(IR66),VR(IR67),VR(IR68),VR(IR74))
C
C
C   ICDD=1
C   IF(III.GT.2) GOTO 577
C   IF(NDC.EQ.1) THEN
C     For displacement boundary value problem, calculate ADVc
C     CALL DISBN(VR(IR69),VR(IR75))
C     DO 570 I=1,ND
C       DDD(I)=0.0
C       DO 570 J=1,NDBC
C         DDD(I)=DDD(I)+AD(I,J)*ADVC(J)
570 CONTINUE
533 FORMAT(113,6F9.3)
C     DO 572 I=1,ND
C       DDD(I)=D(I)-DDD(I)
572 CONTINUE
C     END IF
C     IF(NDC.EQ.0) THEN
C       DO 573 I=1,ND
C         DDD(I)=D(I)
573 CONTINUE
C     END IF
16  FORMAT('D(I) AND DDD(I): ',113,2F14.5)
C
577 CONTINUE
C   WRITE(6,36) III
36  FORMAT('THIS IS THE ITERATION ',113)
C   IF(III.EQ.ITRLM) THEN
C     WRITE(6,*) 'ITERATION LIMIT REACHED. STOP.'
C     STOP
C   END IF
C
C   IF(III.EQ.1) THEN
C     DO 444 I=1,ND
C       DO 444 J=1,ND
C         TDLD(I)=TDLD(I)+A(I,J)*DDD(J)
444 CONTINUE
C
C     DO 755 I=1,ND
C       VTEMP(I)=0.0
C       DO 755 J=1,ND
C         VTEMP(I)=VTEMP(I)+STIFFN(I,J)*TDLD(J)
755 CONTINUE

```





```
ASL=0.0
DO 857 I=1,ND
  ASL=ASL+VTEMP(I)*TDLD(I)
857 CONTINUE
  WRITE(6,*) 'ASL ',ASL
```

```
ETA=1.0
```

```
Next statement is important. It determines the controvariable.
```

```
FAC=DTLM1/ABS(TDLD(NSHOW3))
FAC=DTLM1/ABS(TDLD(ND-NSHOW3))
WRITE(6,*) 'TDELTA=',TDELTA
IF(ASL.LT.0.0) THEN
  FAC=-FAC
  WRITE(6,*) 'CHANGED SIGN OF FAC'
END IF
IF(DETMNT.LT.0.0) WRITE(6,*) 'NEG. DET. STOP'
IF(DETMNT.GT.0.0) FAC=ABS(FAC)
DO 550 I=1,ND
  DLTMP(I)=0.0
  DELTA(I)=0.0
  VTEMP(I)=0.0
  FRCINC(I)=0.0
```

```
550 CONTINUE
END IF
```

```
Finish iii=1 calculation.
Next to calculate the start point displacement HISINC(I)
```

```
ACCELERATION COMPUTATION
```

```
IF((III.EQ.1).OR.(III.EQ.2)) GOTO 624
D55=D5
D66=D6
D77=D7
E11=E1
E22=E2
```

```
Prepare the coefficients of the equation which determines the
load parameter.
```

```
CALL CALCDT(ND,DTL,ROOT,FAC,C1,C2,D11,D2,D3,D4,D5,D6,D7,A4,
1 VR(IR18),VR(IR17),VR(IR26),VR(IR46),VR(IR42))
```

```
ETA0=ETA
ROOT0=ROOT
KK=0
```

```
RTL=ROOT
WRITE(6,*) 'RTL=',RTL
Calculate the root of the equation
CALL CLCRT(ETA0,ETA,ATERM,C1,D11,D2,D3,D4,A4,DTL,ROOT)
ETA=1.0
```

```
624 CONTINUE
```

```
No acceleration iteration
```

```
IF((III.EQ.1).OR.(III.EQ.2)) THEN
```

```
For first and second iterations, there is no acceleration calculation
```

```
ETA=1.0
CALL CALCDT(ND,DTL,ROOT,FAC,C1,C2,D11,D2,D3,D4,D5,D6,D7,A4,
```

```

1      VR (IR18) ,VR (IR17) ,VR (IR26) ,VR (IR46) ,VR (IR42))
C
      IF (III.EQ.1) GOTO 625
      CALL CLCRT (ETAO,ETA,ATERM,C1,D11,D2,D3,D4,A4,DTL,ROOT)
      END IF
      WRITE (6,*) 'III=',III
C
625 CONTINUE
C
      Calculate the displacement increment
C
      DO 635 I=1,ND
      DLTINC (I)=0.0
      IF (III.EQ.1) THEN
      IF (NCONS.EQ.1) THEN
      DO 634 J=1,ND
      DLTINC (I)=DLTINC (I)+A (I,J)*EXLVC (J)
634      CONTINUE
      DLTINC (I)=FAC*TDLD (I)+DLTINC (I)
      ELSE
      DLTINC (I)=FAC*TDLD (I)
      END IF
      ROOT=FAC
      ELSE
      DLTINC (I)=ETA*(HISINC (I)+ROOT*TDLD (I))
      END IF
      DELTA (I)=DLTTMP (I)+DLTINC (I)
635 CONTINUE
      IF (III.EQ.1) THEN
      WRITE (6,*) 'FIRST ITERATION OF STEP ',NUM
      END IF
      I=NEQT
C      WRITE (6,*) 'CURRENT ROOT ',ROOT
C      WRITE (6,*) 'TDLD (25) ',TDLD (I)
C      WRITE (6,*) I,' ROOT*TDLD ',ROOT*TDLD (I)
C      WRITE (6,*) I,' FRCINC ',FRCINC (I)
C      WRITE (6,*) I,' HISINC ',HISINC (I)
C      WRITE (6,*) I,' DLTINC ',DLTINC (I)
C      WRITE (6,*) I,' DELTA ',DELTA (I)
C
C
      K=1
      KK=1
      DO 580 I=1,NNODE
      DO 580 J=1,5
      IF (IID (I,J).EQ.0) THEN
      VF (I,J)=DLTINC (K)
      DD (I,J)=DLTINC (K)
      K=K+1
      END IF
      IF (IID (I,J).EQ.2) THEN
      VF (I,J)=(ROOT-RTL)*ADVC (KK)
      DD (I,J)=VF (I,J)
      KK=KK+1
      END IF
580 CONTINUE
586 FORMAT (113,5F12.8)
C
C
      DO 901 I=1,NNODE
      DO 901 J=1,5
      VFE (I*5-5+J,1)=VF (I,J)
901 CONTINUE
302 FORMAT ('I,VFE (I) IS: ',2I2,1F12.6)
C
C      Estimation of the new coordinates

```

```

C
C
C   TINC=1.0
C
C   Update the coordinates
C
C   DO 900 I=1,NNODE
C     XX (I)=XX (I)+TINC*DD (I,1)
C     YY (I)=YY (I)+TINC*DD (I,2)
C     ZZ (I)=ZZ (I)+TINC*DD (I,3)
C     TMP=0.0
C     DO 903 J=1,3
C       GCL3 (I,J)=GCL3 (I,J)+TINC*(-GCL2 (I,J)*DD (I,4)+GCL1 (I,J)*DD (I,5))
C       TMP=TMP+GCL3 (I,J)*GCL3 (I,J)
903  CONTINUE
C     TMP=TMP**0.5
C     DO 902 J=1,3
C       GCL3 (I,J)=GCL3 (I,J)/TMP
902  CONTINUE
C     WRITE (6,267) I,XX (I),YY (I),ZZ (I)
900  CONTINUE
C
C   Update the directional cosines
C
C   CALL CNND (VR (IR60),VR (IR61),VR (IR62))
C
C   Calculate internal forces
C
C   CALL INTFRG (I11,IPT (IP1),VR (IR1),VR (IR2),VR (IR3),
1      VR (IR14),VR (IR22),VR (IR28),VR (IR9))
C
C   SHRINK THE INTERNAL FORCE VECTOR
C
C   DO 500 I=1,NT
C   DO 500 M=1,ND
C     IF (I.EQ.L (M)) THEN
C       FRCINC (M) = (PLD (I) -FRCO (M))
C       ACTFRC (M) =PLD (I)
C     END IF
500  CONTINUE
C
C
C   DO 447 I=1,ND
C     HISINC (I)=0.0
447  CONTINUE
C   DO 448 I=1,ND
C     DO 449 J=1,ND
C       HISINC (I)=HISINC (I)-A (I,J)*FRCINC (J)
449  CONTINUE
C     WRITE (6,*) I,' HISINC=',HISINC (I)
448  CONTINUE
C
C   DO 549 I=1,ND
C     EXLVC (I)=0.0
C     TDLD (I)=0.0
C     DO 446 J=1,ND
C       TDLD (I)=TDLD (I)+A (I,J)*DDD (J)
446  CONTINUE
549  CONTINUE
C
C   Check whether to step out of the iterations
C
C   ISWTC=0
C   ISEC=ISEC+1
C   IF (ISEC.GT.10) ISEC=10
C   WRITE (6,*) 'I, DDD (I), ROOT*DDD (I), FRCINC (I), EXLVC (I) '
C

```

```

DO 665 I=1,ND
  DLTTMP(I)=DELTA(I)
  ACMDIS(I)=ACMDIS(I)+DLTINC(I)
C WRITE(6,*) I,' ACMDIS ',ACMDIS(I)
665 CONTINUE
C
K=1
DO 585 I=1,NNODE
  DO 585 J=1,5
    IF(IID(I,J).EQ.0) THEN
      D1(I,J)=ACMDIS(K)
      K=K+1
    END IF
585 CONTINUE
C
CALL CRITR1(III,ND,VR(IR8),VR(IR42),VR(IR59),VR(IR17),
1          VLINIT,ICNC1,VALS)
WRITE(6,*) 'VLINIT=',VLINIT
C IF(ICNC1.EQ.0) THEN
C   IF(III.EQ.1) VLS1=VALS
C   IF(III.EQ.2) VLS2=VALS
C   IF(III.GT.2) THEN
C     IF(VALS.GT.VLS1.AND.VALG.GT.VLS2) THEN
C       WRITE(6,*) 'BREAK=',LIM
C       DTLM1=DTLM1/2.0
C       LIM=LIM+1
C       IF(LIM.EQ.20) THEN
C         WRITE(6,*) 'Break limit reached, stop'
C         STOP
C       END IF
C       GOTO 1000
C     ELSE
C       VLS1=VLS2
C       VLS2=VALS
C       LIM=0
C     END IF
C   END IF
C END IF
C
IF((ICONCL.EQ.1).OR.(ICNC1.EQ.1)) THEN
C   IF(III.LT.3.AND.NUM.LT.24) DTLM1=DTLM1*SQQ
C   DTLM1=DTLM1*SQQ
C   IF(III.LE.4) DTLM1=DTLM1*1.1
C   IF(III.GE.8.AND.111.LT.10) DTLM1=DTLM1/1.1
C   IF(III.GE.10.AND.111.LT.15) DTLM1=DTLM1/1.2
C   IF(III.GE.15) DTLM1=DTLM1/1.0
C   WRITE(6,*) 'FIN VAL OF III=',111,' NDTLM1=',DTLM1
C   TROOT=TROOT+ROOT
C
C   For displacement boundary problem:
C
C   IF(NDC.EQ.1) THEN
C     KK=1
C     DO 590 I=1,NNODE
C       DO 590 J=1,5
C         IF(IID(I,J).EQ.2) THEN
C           D1(I,J)=D1(I,J)+ROOT*ADVC(KK)
C           KK=KK+1
C         END IF
590 CONTINUE
C     DO 599 I=1,20
C       WRITE(6,*) I,' D1=',(D1(I,J),J=1,5)
599 CONTINUE
C   CALCULATE BOUNDARY TRACTION
C     TTLD=0.0
C     DO 636 I=1,NDBC

```

```

        TY1 (I)=0.0
        TY2 (I)=0.0
        DO 637 J=1,ND
        637   TY1 (I)=TY1 (I)+ADD (I,J)*DELTA (J)
        CONTINUE
        DO 638 J=1,NDBC
        638   TY2 (I)=TY2 (I)+ADC (I,J)*ADVC (J)*ROOT
        CONTINUE
        TLTY (I)=TLTY (I)+TY1 (I)+TY2 (I)-DBVC (I)
        C     WRITE (6,*) I,' TLTY=',TLTY (I)
        TTLD=TTLD+TLTY (I)
        C     WRITE (6,*) I,' TY1=',TY1 (I),' TY2=',TY2 (I),' TLTY=',TLTY (I)
        636   CONTINUE
        WRITE (6,*) 'TTLD=',TTLD
        END IF
C
C     CRPTM=CRPTM+TDELT
C
C     For a successful iteration, write the output data.
C
C     CALL OUTPUT (TTLD,VR (IR15),VR (IR75),VR (IR71),VR (IR1),VR (IR2),
1         VR (IR3))
C
C     ITYPE=1
C
C     For successful iteration, update some variables.
C
C     CALL UPDT (ITYPE,IPT (IP3),VR (IR1),VR (IR2),VR (IR3),VR (IR12),
1         VR (IR15),VR (IR27),VR (IR43),VR (IR44),VR (IR45),
2         VR (IR46),VR (IR47),VR (IR20),VR (IR48),VR (IR49),
3         VR (IR51),VR (IR58),VR (IR60),VR (IR61),VR (IR62),
4         VR (IR63),VR (IR64),VR (IR65),VR (IR75))
C
C     ELSE
C
C     If the iteration requirement is not satisfied, calculate the
C     following coefficients and go back to the iterations again.
C
        III=III+1
        E1=0.0
        E2=0.0
        DO 510 I=1,ND
        510   E1=E1+HISINC (I)*FRCINC (I)
        E2=E2+TDLD (I)*FRCINC (I)
        CONTINUE
        ICDD=ICDD+1
        C     IF (ICDD.GT.4) THEN
        C     GOTO 579
        C     ELSE
        C     GOTO 577
        C     END IF
        END IF
        670 CONTINUE
C
        DO 555 I=1,ND
        DO 555 J=1,ND
        VTEMP (I)=VTEMP (I)+STIFFN (I,J)*DELTA (J)
        C     IF (I.EQ.J) THEN
        C     WRITE (6,*) 'STIFFN2 ',STIFFN (I,J)
        C     END IF
        555   CONTINUE
C
        ASLOP=0.0
        DO 557 I=1,ND
        557   ASLOP=ASLOP+VTEMP (I)*DELTA (I)
        CONTINUE

```

```

ASLOP=ASLOP/ABS (ASLOP)
IF (NUM.EQ.1) ASO=ASLOP/ROOT/ROOT
ASI=ASLOP/ROOT/ROOT
WRITE (6,*) 'NUM ',NUM
WRITE (6,*) 'ASO, ASI ',ASO,ASI
SP=ASO/ASI
WRITE (6,*) 'SP ',SP

DO 730 I=1,ND
  FRCO(I)=FRCO(I)+FRCINC(I)
730 CONTINUE
  IF (KPDT.EQ.NUM) THEN

    If the required number of iterations has reached, save the
    nessisary data in harddisk. It can be used for further calculation.

    CALL WTCDT (VR (IR27),VR (IR20),VR (IR43),VR (IR44),
1             VR (IR45),VR (IR1),VR (IR2),VR (IR3),
1             VR (IR47),VR (IR10),VR (IR51),VR (IR58),VR (IR60),
3             VR (IR61),VR (IR62),VR (IR15),VR (IR71),VR (IR75))
    END IF
1000 CONTINUE
RETURN
END
END ARCLS

Subroutine CALCDT is used to calculate the coefficients of
the equation which determines the load parameter

SUBROUTINE CALCDT (ND,DTL,ROOT,FAC,C1,C2,D11,D2,D3,D4,D5,D6,D7,
1             A4,TDLD,D,HISINC,DELTA,FRCINC)
IMPLICIT REAL*8 (A-H,O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION TDLD (1),D (1),HISINC (1),DELTA (1),FRCINC (1)
COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,
1             NSHOW3,HRZ,ITRLM,FACTOR
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1             IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2             IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3             IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4             IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5             IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /RLVEC/ VR (1)
COMMON /INTVEC/ IPT (1)

C1=0.0
C2=0.0
D11=0.0
D2=0.0
D3=0.0
D4=0.0
D5=0.0
D6=0.0
D7=0.0
A4=0.0

DO 652 I=1,ND
WRITE (6,*) 'TDLD ',TDLD (I),'HISINC ',HISINC (I),'DELTA ',DELTA (I)
WRITE (6,*) 'I= ',I,'D (I) ',D (I),'FRCINC ',FRCINC (I)
C1=C1+TDLD (I)*TDLD (I)
C2=C2-TDLD (I)*D (I)
D11=D11+TDLD (I)*DELTA (I)
D2=D2+TDLD (I)*HISINC (I)

```

```

        D3=D3+HISINC (1) *HISINC (1)
        D4=D4+HISINC (1) *DELTA (1)
        D5=D5-HISINC (1) *D (1)
        D6=D6+HISINC (1) *FRCINC (1)
        D7=D7+TDLD (1) *FRCINC (1)
652 CONTINUE
C   WRITE (6,*) 'C1=',C1,' D1=',D11,' D2=',D2
C   WRITE (6,*) 'D3=',D3,' D4=',D4
C
        DTL=FAC*FAC*C1
        DO 660 I=1,ND
            A4=A4+DELTA (1) *DELTA (1)
660 CONTINUE
C   WRITE (6,*) 'A4, DTL ',A4,DTL
C
        A4=A4-DTL
        WRITE (6,*) 'A4 FIN. ',A4
C
        RETURN
        END
C
        Nextsubroutine culcalutes the roots of eqs. for lamda (i+1)
C
        SUBROUTINE CLCRT (ETAO,ETA,ATERM,C1,D1,D2,D3,D4,A4,DTL,ROOT)
        IMPLICIT REAL*8 (A-H,O-Z)
        IMPLICIT INTEGER*8 (I-N)
C
        K=0
20 CONTINUE
        K=K+1
        IF (K.EQ.10) THEN
            WRITE (6,*) 'NEGATIVE VALUE FOR SQRT OPER. APPROXM. GIVEN'
            WRITE (6,*) 'THE SQUARE VALUE ',UDRT
            ROOT=-A2/2.0/A1
            GOTO 200
        END IF
        A1=ETA*C1+ATERM
        A2=2.0*D1+2.0*ETA*D2
        A3=ETA*D3+2.0*D4
        WRITE (6,*) 'A1,A2,A3 ',A1,' ',A2,' ',A3
        IF (ABS (A3) .LT.0.0000000001) THEN
            ROOT=-A2/A1
            WRITE (6,*) 'ATTENTION: A3=0'
            RETURN
        END IF
C
C
C   SOLVE THE EQUATION FOR LAMDA (I+1)
C
        UDRT=A2*A2-4.0*A1*A3
        IF (UDRT.LT.0.0) THEN
            WRITE (6,*) 'NEGATIVE VALUE FOR THE ROOT, STOP.'
            STOP
            ETA=(ETA+ETAO) /2.0
            GOTO 20
        END IF
        ROOT1=(-A2+SQRT (UDRT)) /2.0/A1
        ROOT2=(-A2-SQRT (UDRT)) /2.0/A1
        CS1=1.0+ETA*(D4+ROOT1*D1) /DTL
        CS2=1.0+ETA*(D4+ROOT2*D1) /DTL
C   WRITE (6,*) 'ROOT1,ROOT2 ',ROOT1,ROOT2
C   WRITE (6,*) 'CS1,CS2 ',CS1,CS2
        IF ((CS1.LT.0.0) .AND. (CS2.GT.0.0)) THEN
            ROOT=ROOT2
        ELSE
            IF ((CS2.LT.0.0) .AND. (CS1.GT.0.0)) THEN

```

```

        ROOT=ROOT1
      ELSE
        IF (ABS (ROOT1+A3/A2) .LT. ABS (ROOT2+A3/A2)) THEN
          IF (ABS (ROOT1-1.0) .LT. ABS (ROOT2-1.0)) THEN
            ROOT=ROOT1
          ELSE
            ROOT=ROOT2
          END IF
        END IF
      END IF
    END IF
  END IF
C
200 CONTINUE
RETURN
END
C
Subroutine ASSMBL install the stiffness matrix and the load vector
C
SUBROUTINE ASSMBL (III, IEL, ID, IID, L, MAXA, LD, XX, YY, ZZ, DD, D,
1         DLDINC, VF, D1, VFE, TS, AM, P, A, STIFFN, AINV, EXLVC,
2         TXVC, ADC, ADD, AD, DBVC)
  IMPLICIT REAL*8 (A-H, O-Z)
  IMPLICIT INTEGER*8 (I-N)
  DIMENSION IEL (NELM, 8), ID (1), IID (NNODE, 5), L (1), MAXA (1), LD (1)
  DIMENSION XX (1), YY (1), ZZ (1), DD (1), D (1), EXLVC (1),
3         DLDINC (1), VF (NNODE, 5), TXVC (1),
4         D1 (NNODE, 5), VFE (NT, 1), TS (NT, NT), P (1), EXLD (40),
        A (NEQT, NEQT), AM (40, 40), AINV (1), STIFFN (NT, NT),
        ADC (NDBC, NDBC), ADD (NDBC, NEQT), AD (NEQT, NDBC), DBVC (1)
C
COMMON /SCHALR1/ NELM, NNODE, NT
COMMON /SCHALR2/ NEQT, NSTEP, NHBW, COEF1, COEF2, NSHOW1, NSHOW2,
1         NSHOW3, HRZ, ITRLM, FACTOR
COMMON /PNTRIN/ IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10
COMMON /PNTRRL/ IR1, IR2, IR3, IR4, IR5, IR6, IR7, IR8, IR9, IR10,
1         IR11, IR12, IR13, IR14, IR15, IR16, IR17, IR18,
2         IR19, IR20, IR21, IR22, IR23, IR24, IR25, IR26,
3         IR27, IR28, IR29, IR30, IR31, IR32, IR33, IR34,
4         IR35, IR36, IR37, IR38, IR39, IR40, IR41, IR42,
5         IR43, IR44, IR45, IR46, IR47, IR48, IR49, IR50
COMMON /UNIFBD/ IR51, IR52, IR53, IR54, IR55, IR56, IR57, IR58, IR59
COMMON /DIRCS/ IR60, IR61, IR62, IR63, IR64, IR65
COMMON /UNICT/ NCONS, MODEL, ETAA, TDELTA, TINIT
COMMON /RLVEC/ VR (1)
COMMON /INTVEC/ IPT (1)
COMMON /CNTRL/ DETMNT
COMMON /DISCT/ NDC, NDBC
COMMON /DISVC/ IR66, IR67, IR68, IR69
COMMON /TIDF/ IDF
C
CALL MNU (NT, NT, TS)
DO 20 I=1, NT
  EXLVC (I)=0.0
  TXVC (I)=0.0
20 CONTINUE
C
Calculation in defferent element
C
DO 140 I=1, NELM
  I1=IEL (I, 1)
  I2=IEL (I, 2)
  I3=IEL (I, 3)
  I4=IEL (I, 4)
  I5=IEL (I, 5)
  I6=IEL (I, 6)
  I7=IEL (I, 7)

```



```

18=IEL(1,8)

Calculate the element stiffness.

CALL CESM(111,1,11,12,13,14,15,16,17,18,VR(1R21),VR(1R1),
1      VR(1R2),VR(1R3),VR(1R14),VR(1R25),EXLD,VR(1R60),
2      VR(1R61),VR(1R62))

C
C Build the globe stiffness matrix
C
DO 140 J=1,8
  DO 140 K=1,5
    JJ=IEL(1,J)*5-5+K
    J1=J*5-5+K
    IF(NCONS.EQ.1) THEN
      TXVC(JJ)=TXVC(JJ)+EXLD(J1)
    END IF
    DO 140 M=1,8
      DO 140 N=1,5
        MM=IEL(1,M)*5-5+N
        M1=M*5-5+N
        IF(MM.LE.JJ) THEN
          TS(JJ,MM)=TS(JJ,MM)+AM(J1,M1)
        WRITE(6,143) IEL(1,J),JJ,MM,J1,M1,TS(JJ,MM)
        END IF
      END IF
    END IF
  END IF
140 CONTINUE
143 FORMAT('ST IS',5I4,1F15.3)
1200 CONTINUE

C
J=1
JD=1
DO 150 I=1,NT
  IF(ID(I).EQ.0) THEN
    L(J)=I
    J=J+1
  END IF
  IF(ID(I).EQ.2) THEN
    LD(JD)=I
    JD=JD+1
  END IF
150 CONTINUE

C
IDF=J-1
JJD=JD-1
WRITE(6,*) 'JJD=',JJD,' IDF=',IDF

C
idf is the number of unknown disp.
C
jld is the number of given disp.
C
200 CONTINUE

C
210 FORMAT('I,LDINC,LOADT,PLD IS',1I3,3F8.3)

C
C Shrinking the load vector and stiff matrix.
C
DO 500 I=1,NT
DO 500 M=1,IDF
  IF(I.EQ.L(M)) THEN
    D(M)=DLINC(I)
    IF(NCONS.EQ.1) THEN
      EXLVC(M)=TXVC(I)
    WRITE(6,*) M,' EXLVC IN ASSMB: ',EXLVC(M)
    END IF
  DO 510 J=1,NT
  DO 510 N=1,IDF

```

```

        IF (J.EQ.L(N)) THEN
            A(M,N)=TS(I,J)
        END IF
510    CONTINUE
        IF (NDC.EQ.1) THEN
            DO 505 J=1,NT
                DO 505 N=1,JJD
                    IF (J.EQ.LD(N)) THEN
                        AD(M,N)=TS(I,J)
                    END IF
205    CONTINUE
        END IF
    END IF
500 CONTINUE
C
C
        IF (NDC.EQ.1) THEN
            DO 600 I=1,NT
                DO 600 M=1,JJD
                    IF (I.EQ.LD(M)) THEN
                        IF (NCONS.EQ.1) THEN
                            DBVC(M)=TXVC(I)
C                            WRITE(6,*) M, ' EXLVC IN ASSMB: ', EXLVC(M)
                        END IF
                        DO 610 J=1,NT
                            DO 610 N=1,IDF
                                IF (J.EQ.L(N)) THEN
                                    ADD(M,N)=TS(I,J)
                                END IF
610    CONTINUE
                                DO 605 J=1,NT
                                    DO 605 N=1,JJD
                                        IF (J.EQ.LD(N)) THEN
                                            ADC(M,N)=TS(I,J)
                                        END IF
605    CONTINUE
                                END IF
600 CONTINUE
        END IF
C
C
        K=0
        DO 550 I=1,NEQT
            DO 550 J=1,NEQT
C                K=K+1
C                P(K)=A(I,J)
                STIFFN(I,J)=A(I,J)
550 CONTINUE
C
C    Inverse the stiffness matrix
C
        IJOB=1
        DD1=1.0
C        CALL LINV3F(A,BB,IJOB,NEQT,NEQT,DD1,DD2,AINV,IER)
        CALL LINRG(NEQT,A,NEQT,A,NEQT)
        DETMNT=DD1*(2**DD2)
        IF (IER.EQ.130) THEN
            WRITE(6,*) ' INVERSE PROB.
            STOP
        END IF
C
C    WRITE(6,*) ' END ASSEM'
        RETURN
        END
C    (END ASSEMBL)
C

```

```

C      Next subroutine is used to calculate the nodal force
C
SUBROUTINE INTFRC (III, IEL, XX, YY, ZZ, VF, PD, PDL, PLD)
IMPLICIT REAL*8 (A-H, O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION XX (1), YY (1), ZZ (1), VF (NNODE, 5), PD (1), PDL (1), PLD (1)
DIMENSION H (2), P (2), R (8), S (8), X (8), Y (8), Z (8), ND (8), IEL (NELM, 8),
1      VFE (40)
COMMON /SCHALR1/ NELM, NNODE, NT
COMMON /PNTRIN/ IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10
COMMON /PNTRRL/ IR1, IR2, IR3, IR4, IR5, IR6, IR7, IR8, IR9, IR10,
1      IR11, IR12, IR13, IR14, IR15, IR16, IR17, IR18,
2      IR19, IR20, IR21, IR22, IR23, IR24, IR25, IR26,
3      IR27, IR28, IR29, IR30, IR31, IR32, IR33, IR34,
4      IR35, IR36, IR37, IR38, IR39, IR40, IR41, IR42,
5      IR43, IR44, IR45, IR46, IR47, IR48, IR49, IR50
COMMON /UNIFBD/ IR51, IR52, IR53, IR54, IR55, IR56, IR57, IR58, IR59
COMMON /DIRCS/ IR60, IR61, IR62, IR63, IR64, IR65
COMMON /UNICT/ NCONS, MODEL, ETAA, TDELTA, TINIT
COMMON /RLVEC/ VR (1)
COMMON /INTVEC/ IPT (1)
COMMON /A3/ CL1 (8), CM1 (8), CN1 (8), CL2 (8), CM2 (8), CN2 (8),
1      CL3 (8), CM3 (8), CN3 (8)

C
C      DO 30 I=1,NT
C          PLD (I)=0.0
30 CONTINUE

C
C      DO 700 I=1, NELM
C          I1=IEL (I, 1)
C          I2=IEL (I, 2)
C          I3=IEL (I, 3)
C          I4=IEL (I, 4)
C          I5=IEL (I, 5)
C          I6=IEL (I, 6)
C          I7=IEL (I, 7)
C          I8=IEL (I, 8)

C
C      Calculate the nodal force for each element
C
C      CALL UPDATA (III, I, I1, I2, I3, I4, I5, I6, I7, I8, VR (IR1), VR (IR2), VR (IR3),
1      VR (IR14), VR (IR22), VR (IR28), VR (IR60), VR (IR61), VR (IR62))

C
C
C          DO 700 J=1, 8
C              DO 700 K=1, 5
C                  JJ=IEL (I, J) *5-5+K
C                  J1=J*5-5+K
C                  PLD (JJ)=PLD (JJ)+PD (J1)
C                  write (6, 110) I, JJ, J1
700 CONTINUE

C
C      RETURN
C      END
C      (END INTFRC)

C
C      Subroutine CESM is used to calculate the stiffness matrix for
C      each element
C
SUBROUTINE CESM (III, IL, I1, I2, I3, I4, I5, I6,
1      I7, I8, SM, XX, YY, ZZ, VF, ESM, EXLD, GCL1, GCL2, GCL3)
IMPLICIT REAL*8 (A-H, O-Z)
IMPLICIT INTEGER*8 (I-N)

C
C      DIMENSION XX (1), YY (1), ZZ (1), VF (NNODE, 5), SM (40, 40), ESM (40, 40),

```

```

1          H (2) , P (2) , R (8) , S (8) , X (8) , Y (8) , HH (4) , PP (4) ,
2          Z (8) , ND (8) , VFE (40) , EXED (40) , EXLD (40) ,
3          GCL1 (NNODE , 3) , GCL2 (NNODE , 3) , GCL3 (NNODE , 3)
C
COMMON /SCHALR1/ NELM , NNODE , NT
COMMON /A3/ CL1 (8) , CM1 (8) , CN1 (8) , CL2 (8) , CM2 (8) , CN2 (8) ,
1          CL3 (8) , CM3 (8) , CN3 (8)
COMMON /PNTRIN/ IP1 , IP2 , IP3 , IP4 , IP5 , IP6 , IP7 , IP8 , IP9 , IP10
COMMON /PNTRRL/ IR1 , IR2 , IR3 , IR4 , IR5 , IR6 , IR7 , IR8 , IR9 , IR10 ,
1          IR11 , IR12 , IR13 , IR14 , IR15 , IR16 , IR17 , IR18 ,
2          IR19 , IR20 , IR21 , IR22 , IR23 , IR24 , IR25 , IR26 ,
3          IR27 , IR28 , IR29 , IR30 , IR31 , IR32 , IR33 , IR34 ,
4          IR35 , IR36 , IR37 , IR38 , IR39 , IR40 , IR41 , IR42 ,
5          IR43 , IR44 , IR45 , IR46 , IR47 , IR48 , IR49 , IR50
COMMON /UNIFBD/ IR51 , IR52 , IR53 , IR54 , IR55 , IR56 , IR57 , IR58 , IR59
COMMON /DIRCS/ IR60 , IR61 , IR62 , IR63 , IR64 , IR65
COMMON /CONTN/ INSIDT , KPDT , DTLM1
COMMON /UNICT/ NCONS , MODEL , ETAA , TDELTA , TINIT
COMMON /RLVEC/ VR (1)
COMMON /INTVEC/ IPT (1)
C
C
ND (1) = 11
ND (2) = 12
ND (3) = 13
ND (4) = 14
ND (5) = 15
ND (6) = 16
ND (7) = 17
ND (8) = 18
C
CALL MNU (40 , 40 , SM)
DO 20 I = 1 , 40
    EXLD (I) = 0.0
20 CONTINUE
C
DO 250 I = 1 , 8
    X (I) = XX (ND (I))
    Y (I) = YY (ND (I))
    Z (I) = ZZ (ND (I))
C    ( Change displacemet field from matrix to vector.)
C
DO 250 J = 1 , 5
    VFE (I * 5 - 5 + J) = VF (ND (I) , J)
250 CONTINUE
C
C
R (1) = -1
S (1) = -1
R (2) = 1
S (2) = -1
R (3) = 1
S (3) = 1
R (4) = -1
S (4) = 1
C
R (5) = 0
S (5) = -1
R (6) = 1
S (6) = 0
R (7) = 0
S (7) = 1
R (8) = -1
S (8) = 0
C
WRITE (6 , 157) IL
C

```

```

DO 344 I=1,8
CL1 (1)=GCL1 (ND (1) ,1)
CM1 (1)=GCL1 (ND (1) ,2)
CN1 (1)=GCL1 (ND (1) ,3)
CL2 (1)=GCL2 (ND (1) ,1)
CM2 (1)=GCL2 (ND (1) ,2)
CN2 (1)=GCL2 (ND (1) ,3)
CL3 (1)=GCL3 (ND (1) ,1)
CM3 (1)=GCL3 (ND (1) ,2)
CN3 (1)=GCL3 (ND (1) ,3)
C
344 CONTINUE
346 FORMAT (112,9F7.4)
C
C
H (1)=1.0
H (2)=1.0
C
P (1)=0.577352692
P (2)=-P (1)
C
C
HH (1)=0.3478548451
HH (2)=H (1)
C
HH (3)=0.6521451548
HH (4)=H (3)
C
PP (1)=0.8611363115
PP (2)=-P (1)
C
PP (3)=0.3399810435
PP (4)=-P (3)
C
C
DO 150 I=1,2
DO 150 J=1,2
DO 150 K=1,2
U=P (I)
V=P (J)
W=P (K)
C
C
Calculate the stiffness matrix at every integration point
C
CALL CB (111,IL,I,J,K,U,V,W,X,Y,Z,DETJ,VR (IR25),VR (IR28),
1 VR (IR29),VR (IR30),VR (IR31),VR (IR32),VR (IR33),
2 VR (IR34),VR (IR35),VR (IR36),VR (IR37),VR (IR38),
3 VR (IR39),VR (IR40),VR (IR47),EXED,VR (IR53),VR (IR56),
4 VR (IR57))
C
C
DO 150 M=1,40
IF (NCONS.EQ.1) THEN
EXLD (M)=EXLD (M)+H (I) *H (J) *H (K) *EXED (M) *DETJ
END IF
DO 150 N=1,40
SM (M,N)=SM (M,N)+H (I) *H (J) *H (K) *ESM (M,N) *DETJ
C
150 CONTINUE
C
C
WRITE (6,*) 'DETJ=',DETJ
154 FORMAT ('M,N,SM (M,N) IS:',2I3,1F12.4)
C
RETURN
END
C
C
NEXT SUBROUTINE IS USED TO CALCULATE THE DIRECTION
COSINES AT NODE POINTS.HERE R,S,X,Y ARE THE NODE
COORD. IN REF.AND CART. COORD. RESPECTIVELY. CXR..
CZN ARE THE DIRECTION COSINES.

```

```

SUBROUTINE CN (R,S,X,Y,Z,CXR,CYR,CZR,
1           CXS,CYS,CZS,CXN,CYN,CZN)

```

```

IMPLICIT REAL*8 (A-H,O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION X (8),Y (8),Z (8),FR (8),FS (8)

```

```

C XS.. MEANS DX/DS AND SO ON
C S2=S*S
C R2=R*R

```

```

C WRITE (6,*) R,S
C WRITE (6,*)
C DO 20 I=1,8
C WRITE (6,10) I,X (I),Y (I),Z (I)
C 20 CONTINUE
10 FORMAT ('X,Y,Z (I) ARE: ',1I3,3F10.4)

```

```

C FR (1) = (2.0*R+S) * (1.0-S) /4.0
C FR (2) = (2.0*R-S) * (1.0-S) /4.0
C FR (3) = (2.0*R+S) * (1.0+S) /4.0
C FR (4) = (2.0*R-S) * (1.0+S) /4.0
C FR (5) = -R * (1.0-S)
C FR (6) = (1.0-S2) /2.0
C FR (7) = -R * (1.0+S)
C FR (8) = - (1.0-S2) /2.0

```

```

C FS (1) = (1.0-R) * (2.0*S+R) /4.0
C FS (2) = (1.0+R) * (2.0*S-R) /4.0
C FS (3) = (1.0+R) * (2.0*S+R) /4.0
C FS (4) = (1.0-R) * (2.0*S-R) /4.0
C FS (5) = - (1.0-R2) /2.0
C FS (6) = - (1.0+R) *S
C FS (7) = (1.0-R2) /2.0
C FS (8) = - (1.0-R) *S

```

```

C XR=0
C YR=0
C ZR=0
C XS=0
C YS=0
C ZS=0

```

```

C DO 315 I=1,8
C XR=XR+FR (I) *X (I)
C YR=YR+FR (I) *Y (I)
C ZR=ZR+FR (I) *Z (I)
C XS=XS+FS (I) *X (I)
C YS=YS+FS (I) *Y (I)
C ZS=ZS+FS (I) *Z (I)

```

```

315 CONTINUE

```

```

C GRR,GSS,GRS ARE THE METRIC TENSOR IN THE REFERENCE COORD.

```

```

C GRR=XR*XR+YR*YR+ZR*ZR
C GSS=XS*XS+YS*YS+ZS*ZS
C GRS=XR*XS+YR*YS+ZR*ZS

```

```

C GRRH=SQRT (GRR)
C GSSH=SQRT (GSS)
C GRSHH=GRRH*GSSH

```

```

C WRITE (6,408) R,S,GRR,GSS,GRS
408 FORMAT ('THE METRIC AT NODE R= ',1F2.0,'S= ',1F2.0,3F10.5)
C WRITE (6,409) R,S,GRRH,GSSH,GRSHH

```

```

409 FORMAT('THE METRIC AT NODE R= ',1F2.0,'S= ',1F2.0,3F10.5)
C
C
C   CXR IS THE DIRECTION COSINE BETWEEN THE AXES X AND R.THE
C   SAME MEANING THROUGH CZS.
C
C   CXR=XR/GRRH
C   CYR=YR/GRRH
C   CZR=ZR/GRRH
C
C   CXS=XS/GSSH
C   CYS=YS/GSSH
C   CZS=ZS/GSSH
C
C
C   THE CXN..ARE THE DIRECTION COSINES BETWEEN THE UNIT NORMAL
C   AND THE COORD. X,Y,Z.
C
C   CXN=(YR*ZS-ZR*YS)/GRSHH
C   CYN=(ZR*XS-XR*ZS)/GRSHH
C   CZN=(XR*YS-YR*XS)/GRSHH
C
C   RETURN
C   END
C
C
C   THIS IS A PROCEDURE TO MULTIPLY TWO MATRIX
C
C   SUBROUTINE MMT(I,K,J,A1,A2,A)
C   IMPLICIT REAL*8(A-H,O-Z)
C   IMPLICIT INTEGER*8(I-N)
C   DIMENSION A1(I,K),A2(K,J),A(I,J)
C
C   CALL MNU(I,J,A)
C   DO 20 M=1,I
C     DO 20 N=1,J
C       DO 20 L=1,K
C         TEMP=A1(M,L)*A2(L,N)
C         A(M,N)=A(M,N)+TEMP
20 CONTINUE
C   RETURN
C   END
C
C
C   THIS IS A PROCEDURE TO MAKE NULL MATRIX
C
C   SUBROUTINE MNU(I,J,A)
C   IMPLICIT REAL*8(A-H,O-Z)
C   IMPLICIT INTEGER*8(I-N)
C   DIMENSION A(I,J)
C   DO 30 M=1,I
C     DO 30 N=1,J
C       A(M,N)=0.0
30 CONTINUE
C   RETURN
C   END
C
C   Subroutine transp is to make transpose matrix.
C
C   SUBROUTINE TRANSP(I,J,XI,XO)
C   IMPLICIT REAL*8(A-H,O-Z)
C   IMPLICIT INTEGER*8(I-N)
C   DIMENSION XI(I,J),XO(J,I)
C
C   DO 10 M=1,I

```

```

DO 10 N=1,J
  XO(N,M)=X1(M,N)
10 CONTINUE
RETURN
END

```

Subroutine GetGeom(r,s,t,t0,x,y,z,rj,detj) is to calculate the geometric property at an intergration point. Here input is: r,s - the intergration position,t0 - the thickness of the the shell, the x,y,z - the nodes's coordinates.The Jacobin and the reversed Jacobin matrix,as well as the determinate of the Jacobin matrix are calculated. A,B,C,D,E,G are the outputs.

```

SUBROUTINE GEOM(R,S,T,TO,X,Y,Z,DETJ,A,B,C,D,E,G)
  IMPLICIT REAL*8 (A-H,O-Z)
  IMPLICIT INTEGER*8 (I-N)
  DIMENSION X(8),Y(8),Z(8),RJ(3,3),F(8),FR(8),FS(8),CJ(3,3),
2      A(8),B(8),C(8),D(8),E(8),G(8)
  COMMON /A3/CL1(8),CM1(8),CN1(8),CL2(8),CM2(8),CN2(8),
1      CL3(8),CM3(8),CN3(8)

```

```

S2=S*S
R2=R*R
S3=S2*S
R3=R2*R

```

C  
C  
C

F(k) is the shape function evaluated at node k.

```

F(1)=(1.0-R)*(1.0-S)*(-R-S-1.0)/4.0
F(2)=(1.0+R)*(1.0-S)*(R-S-1.0)/4.0
F(3)=(1.0+R)*(1.0+S)*(R+S-1.0)/4.0
F(4)=(1.0-R)*(1.0+S)*(-R+S-1.0)/4.0
F(5)=(1.0-R2)*(1.0-S)/2.0
F(6)=(1.0+R)*(1.0-S2)/2.0
F(7)=(1.0-R2)*(1.0+S)/2.0
F(8)=(1.0-R)*(1.0-S2)/2.0

```

C  
C  
C

FR(k) is the derivetive w.r.t. r of the shape function

```

FR(1)=(2.0*R+S)*(1.0-S)/4.0
FR(2)=(2.0*R-S)*(1.0-S)/4.0
FR(3)=(2.0*R+S)*(1.0+S)/4.0
FR(4)=(2.0*R-S)*(1.0+S)/4.0
FR(5)=-R*(1.0-S)
FR(6)=(1.0-S2)/2.0
FR(7)=-R*(1.0+S)
FR(8)=-R*(1.0-S2)/2.0

```

C  
C  
C

FR(k) is the derivetive w.r.t. s of the shape function

```

FS(1)=(1.0-R)*(2.0*S+R)/4.0
FS(2)=(1.0+R)*(2.0*S-R)/4.0
FS(3)=(1.0+R)*(2.0*S+R)/4.0
FS(4)=(1.0-R)*(2.0*S-R)/4.0
FS(5)=-R*(1.0-S)
FS(6)=-R*(1.0+S)
FS(7)=(1.0-R2)/2.0
FS(8)=-R*(1.0-S2)/2.0

```

C  
C  
C

CJ is the Jacobin matrix.

```

CALL MNU(3,3,CJ)

```

C

```

DO 346 I=1,8
  CJ(1,1)=CJ(1,1)+FR(I)*(X(I)+T*TO*CL3(I)/2.0)
  CJ(1,2)=CJ(1,2)+FR(I)*(Y(I)+T*TO*CM3(I)/2.0)

```



```

CJ (1,3)=CJ (1,3)+FR (1) * (Z (1)+T*TO*CN3 (1) /2.0)
CJ (2,1)=CJ (2,1)+FS (1) * (X (1)+T*TO*CL3 (1) /2.0)
CJ (2,2)=CJ (2,2)+FS (1) * (Y (1)+T*TO*CM3 (1) /2.0)
CJ (2,3)=CJ (2,3)+FS (1) * (Z (1)+T*TO*CN3 (1) /2.0)
CJ (3,1)=F (1) *TO*CL3 (1) /2.0+CJ (3,1)
CJ (3,2)=F (1) *TO*CM3 (1) /2.0+CJ (3,2)
CJ (3,3)=F (1) *TO*CN3 (1) /2.0+CJ (3,3)

```

346 CONTINUE

C  
C  
C

Detj is the determinate of the Jacobin matrix.

```

DETJ=CJ (1,1) * (CJ (2,2) *CJ (3,3) -CJ (3,2) *CJ (2,3))
1  -CJ (1,2) * (CJ (2,1) *CJ (3,3) -CJ (3,1) *CJ (2,3))
2  +CJ (1,3) * (CJ (2,1) *CJ (3,2) -CJ (3,1) *CJ (2,2))

```

C  
C  
C

WRITE (6,347) DETJ

347 FORMAT ('DETJ IS',1F12.9)

C  
C  
C  
C

RJ is the inverse of the jacobin matrix.

```

RJ (1,1)=(CJ (2,2) *CJ (3,3) -CJ (3,2) *CJ (2,3)) /DETJ
RJ (1,2)-- (CJ (1,2) *CJ (3,3) -CJ (3,2) *CJ (1,3)) /DETJ
RJ (1,3)=(CJ (1,2) *CJ (2,3) -CJ (2,2) *CJ (1,3)) /DETJ

```

C  
C  
C  
C

```

RJ (2,1)-- (CJ (2,1) *CJ (3,3) -CJ (3,1) *CJ (2,3)) /DETJ
RJ (2,2)=(CJ (1,1) *CJ (3,3) -CJ (3,1) *CJ (1,3)) /DETJ
RJ (2,3)-- (CJ (1,1) *CJ (2,3) -CJ (2,1) *CJ (1,3)) /DETJ

```

C  
C  
C

```

RJ (3,1)=(CJ (2,1) *CJ (3,2) -CJ (3,1) *CJ (2,2)) /DETJ
RJ (3,2)-- (CJ (1,1) *CJ (3,2) -CJ (3,1) *CJ (1,2)) /DETJ
RJ (3,3)=(CJ (1,1) *CJ (2,2) -CJ (2,1) *CJ (1,2)) /DETJ

```

C  
C  
C

DO 360 I=1,8

```

A (I)=RJ (1,I) *FR (I) +RJ (1,2) *FS (I)
B (I)=RJ (2,I) *FR (I) +RJ (2,2) *FS (I)
C (I)=RJ (3,I) *FR (I) +RJ (3,2) *FS (I)
D (I)=TO* (A (I) *T+RJ (1,3) *F (I)) /2.0
E (I)=TO* (B (I) *T+RJ (2,3) *F (I)) /2.0
G (I)=TO* (C (I) *T+RJ (3,3) *F (I)) /2.0

```

360 CONTINUE

C  
C  
C

RETURN  
END

C  
C  
C  
C

Subroutine Rotsmatrix is to get the rotate transformation matrix. Here the input is r,s,x,y,z. Output is transformation matrix tl.

SUBROUTINE ROTMTRX (R,S,X,Y,Z,TL)

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

IMPLICIT INTEGER\*8 (I-N)

DIMENSION X (8), Y (8), Z (8), TL (6,6)

COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10

COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,

1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,

2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,

3 IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,

4 IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,

5 IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50

COMMON /RLVEC/ VR (1)

COMMON /INTVEC/ IPT (1)

C  
C

CALL CN (R,S,X,Y,Z,PL1,PM1,PN1,PL2,PM2,PN2,PL3,PM3,PN3)

C  
C  
C

WRITE (6,\*) 'PL1=',PL1,' PL2=',PL2,' PL3=',PL3

WRITE (6,\*) 'PM1=',PM1,' PM2=',PM2,' PM3=',PM3

```

C WRITE (6,*) 'PN1=',PN1,' PN2=',PN2,' PN3=',PN3
C TL (1,1)=PL1**2
C TL (2,1)=PL2**2
C TL (3,1)=PL3**2
C TL (4,1)=PL1*PL2*2.0
C TL (5,1)=PL2*PL3*2.0
C TL (6,1)=PL3*PL1*2.0
C
C TL (1,2)=PM1**2
C TL (2,2)=PM2**2
C TL (3,2)=PM3**2
C TL (4,2)=PM1*PM2*2.0
C TL (5,2)=PM2*PM3*2.0
C TL (6,2)=PM3*PM1*2.0
C
C TL (1,3)=PN1**2
C TL (2,3)=PN2**2
C TL (3,3)=PN3**2
C TL (4,3)=PN1*PN2*2.0
C TL (5,3)=PN2*PN3*2.0
C TL (6,3)=PN3*PN1*2.0
C
C TL (1,4)=PL1*PM1
C TL (2,4)=PL2*PM2
C TL (3,4)=PL3*PM3
C TL (4,4)=PL1*PM2+PL2*PM1
C TL (5,4)=PL2*PM3+PL3*PM2
C TL (6,4)=PL3*PM1+PL1*PM3
C
C TL (1,5)=PM1*PN1
C TL (2,5)=PM2*PN2
C TL (3,5)=PM3*PN3
C TL (4,5)=PM1*PN2+PM2*PN1
C TL (5,5)=PM2*PN3+PM3*PN2
C TL (6,5)=PM3*PN1+PM1*PN3
C
C TL (1,6)=PN1*PL1
C TL (2,6)=PN2*PL2
C TL (3,6)=PN3*PL3
C TL (4,6)=PN1*PL2+PN2*PL1
C TL (5,6)=PN2*PL3+PN3*PL2
C TL (6,6)=PN3*PL1+PN1*PL3
C
C RETURN
C END
C
C Subroutine nonlm is to get the nonlinear part of the matrix B. Here
C the input is the geometric parameters a,b,c,d,e,g and the direction
C cosines.The parameter ss is the stress calculated in last iteration.
C The output is the matrix bn1(40,40) and bn2(40,40)
C
C SUBROUTINE NONLM(A,B,C,D,E,G,SS,SS1,BN1,BN2,BN3,B1,BIT,TMPSS)
C IMPLICIT REAL*8 (A-H,O-Z)
C IMPLICIT INTEGER*8 (I-N)
C DIMENSION A (8) ,B (8) ,C (8) ,D (8) ,E (8) ,G (8) ,SS (9,9) ,SS1 (9,9) ,
1 BN1 (40,40) ,BN2 (40,40) ,BN3 (40,40) ,B1 (9,40) ,
2 BIT (40,9) ,TMPSS (40,9)
C
C COMMON /SCHALR1/ NELM,NNODE,NT
C COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
C COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3 IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4 IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,

```

```

5          IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /RLVEC/ VR(1)
COMMON /INTVEC/ IPT(1)
COMMON /A3/CL1(8),CM1(8),CN1(8),CL2(8),CM2(8),CN2(8),
1          CL3(8),CM3(8),CN3(8)

```

```

C
C
CALL MNU(9,40,B1)
C
C

```

```

DO 413 I=1,8
  B1(1,I*5-4)=A(I)
  B1(2,I*5-4)=B(I)
  B1(3,I*5-4)=C(I)

```

```

C
  B1(4,I*5-3)=A(I)
  B1(5,I*5-3)=B(I)
  B1(6,I*5-3)=C(I)

```

```

  B1(7,I*5-2)=A(I)
  B1(8,I*5-2)=B(I)
  B1(9,I*5-2)=C(I)

```

```

C
  B1(1,I*5-1)=-D(I)*CL2(I)
  B1(2,I*5-1)=-E(I)*CL2(I)
  B1(3,I*5-1)=-G(I)*CL2(I)
  B1(4,I*5-1)=-D(I)*CM2(I)
  B1(5,I*5-1)=-E(I)*CM2(I)
  B1(6,I*5-1)=-G(I)*CM2(I)
  B1(7,I*5-1)=-D(I)*CN2(I)
  B1(8,I*5-1)=-E(I)*CN2(I)
  B1(9,I*5-1)=-G(I)*CN2(I)

```

```

C
  B1(1,I*5)=D(I)*CL1(I)
  B1(2,I*5)=E(I)*CL1(I)
  B1(3,I*5)=G(I)*CL1(I)
  B1(4,I*5)=D(I)*CM1(I)
  B1(5,I*5)=E(I)*CM1(I)
  B1(6,I*5)=G(I)*CM1(I)
  B1(7,I*5)=D(I)*CN1(I)
  B1(8,I*5)=E(I)*CN1(I)
  B1(9,I*5)=G(I)*CN1(I)

```

```

413 CONTINUE

```

```

C
C
DO 430 I=1,40
  DO 430 J=1,9
    BIT(I,J)=B1(J,I)

```

```

430 CONTINUE

```

```

C
CALL MMT(40,9,9,BIT,SS,TMPSS)
CALL MMT(40,9,40,TMPSS,B1,BN1)

```

```

C
CALL MMT(40,9,9,BIT,SS1,TMPSS)
CALL MMT(40,9,40,TMPSS,B1,BN3)

```

```

C
B2=B1 NOW.
CALL MNU(9,40,B1)

```

```

C
DO 414 I=1,8
  B1(1,I*5-4)=A(I)
  B1(2,I*5-4)=B(I)/2.0
  B1(3,I*5-4)=C(I)/2.0
  B1(4,I*5-4)=B(I)/2.0
  B1(7,I*5-4)=C(I)/2.0

```

```

B1 (2,1*5-3)=A (1) /2.0
B1 (4,1*5-3)=A (1) /2.0
B1 (5,1*5-3)=B (1)
B1 (6,1*5-3)=C (1) /2.0
B1 (8,1*5-3)=C (1) /2.0
C
B1 (3,1*5-2)=A (1) /2.0
B1 (6,1*5-2)=B (1) /2.0
B1 (7,1*5-2)=A (1) /2.0
B1 (8,1*5-2)=B (1) /2.0
B1 (9,1*5-2)=C (1)
C
B1 (1,1*5-1)=-D (1) *CL2 (1)
B1 (2,1*5-1)=- (E (1) *CL2 (1)+D (1) *CM2 (1)) /2.0
B1 (3,1*5-1)=- (G (1) *CL2 (1)+D (1) *CN2 (1)) /2.0
B1 (4,1*5-1)=- (E (1) *CL2 (1)+D (1) *CM2 (1)) /2.0
B1 (5,1*5-1)=-E (1) *CM2 (1)
B1 (6,1*5-1)=- (G (1) *CM2 (1)+E (1) *CN2 (1)) /2.0
B1 (7,1*5-1)=- (G (1) *CL2 (1)+D (1) *CN2 (1)) /2.0
B1 (8,1*5-1)=- (G (1) *CM2 (1)+E (1) *CN2 (1)) /2.0
B1 (9,1*5-1)=-G (1) *CN2 (1)
C
B1 (1,1*5)=D (1) *CL1 (1)
B1 (2,1*5)= (E (1) *CL1 (1)+D (1) *CM1 (1)) /2.0
B1 (3,1*5)= (G (1) *CL1 (1)+D (1) *CN1 (1)) /2.0
B1 (4,1*5)= (E (1) *CL1 (1)+D (1) *CM1 (1)) /2.0
B1 (5,1*5)=E (1) *CM1 (1)
B1 (6,1*5)= (G (1) *CM1 (1)+E (1) *CN1 (1)) /2.0
B1 (7,1*5)= (G (1) *CL1 (1)+D (1) *CN1 (1)) /2.0
B1 (8,1*5)= (G (1) *CM1 (1)+E (1) *CN1 (1)) /2.0
414 CONTINUE
C
DO 432 I=1,40
DO 432 J=1,9
BIT (I,J)=B1 (J,I)
B2T (I,J)=B2 (J,I)
C
432 CONTINUE
C
CALL MMT (40,9,9,BIT,SS,TMPSS)
CALL MMT (40,9,40,TMPSS,B1,BN2)
C
RETURN
END
(end nonlm)
C
C
C
C
Subroutine ELSMTR is used to calculate the elastic matrix
C
SUBROUTINE ELSMTR (EM)
IMPLICIT REAL*8 (A-H,O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION EM (6,6)
COMMON /MTL/ E,EU
U=EU
CALL MNU (6,6,EM)
EM (1,1)=E / (1.0-U*U)
C
WRITE (6,*) 'EM=',EM (1,1)
EM (2,2)=EM (1,1)
EM (3,3)=1.0
EM (1,2)=E*U / (1.0-U*U)
EM (2,1)=EM (1,2)
EM (5,5)=E/2 / (1+U)
EM (4,4)=EM (5,5)
EM (6,6)=EM (5,5)

```

```

RETURN
END
C (enc eismtr)
C
C This procedure is used to calculate the nodal force in
C every element
C
SUBROUTINE UPDATA (I11, I1, I2, I3, I4, I5, I6, I7, I8, XX, YY, ZZ,
1 VF, PD, PDL, GCL1, GCL2, GCL3)

IMPLICIT REAL*8 (A-H, O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION XX (1), YY (1), ZZ (1), VF (NNODE, 5), PD (1), PDL (1)
DIMENSION H (2), P (2), R (8), S (8), X (8), Y (8), Z (8), ND (8),
1 VFE (40), GCL1 (NNODE, 3), GCL2 (NNODE, 3), GCL3 (NNODE, 3),
2 HH (4), PP (4)
COMMON /SCHALR1/ NELM, NNODE, NT
COMMON /PNTRIN/ IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10
COMMON /PNTRRL/ IR1, IR2, IR3, IR4, IR5, IR6, IR7, IR8, IR9, IR10,
1 IR11, IR12, IR13, IR14, IR15, IR16, IR17, IR18,
2 IR19, IR20, IR21, IR22, IR23, IR24, IR25, IR26,
3 IR27, IR28, IR29, IR30, IR31, IR32, IR33, IR34,
4 IR35, IR36, IR37, IR38, IR39, IR40, IR41, IR42,
5 IR43, IR44, IR45, IR46, IR47, IR48, IR49, IR50
COMMON /UNIFBD/ IR51, IR52, IR53, IR54, IR55, IR56, IR57, IR58, IR59
COMMON /DIRCS/ IR60, IR61, IR62, IR63, IR64, IR65
COMMON /UNICT/ NCONS, MODEL, ETAA, TDELTA, TINIT
COMMON /RLVEC/ VR (1)
COMMON /INTVEC/ IPT (1)
COMMON /CONTN/ INSIDT, KPDT, DTLMI
COMMON /A3/ CL1 (8), CM1 (8), CN1 (8), CL2 (8), CM2 (8), CN2 (8),
1 CL3 (8), CM3 (8), CN3 (8)
C
C
ND (1) = I1
ND (2) = I2
ND (3) = I3
ND (4) = I4
ND (5) = I5
ND (6) = I6
ND (7) = I7
ND (8) = I8
C
DO 250 I = 1, 8
X (I) = XX (ND (I))
Y (I) = YY (ND (I))
Z (I) = ZZ (ND (I))
C WRITE (6, 260) I, X (I), Y (I), Z (I), ND (I)
DO 250 J = 1, 5
VFE (I*5-5+J) = VF (ND (I), J)
250 CONTINUE
260 FORMAT (1X, 'THE COORDINATES OF NODE', I2, 1X, 'ARE:', 3F12.8, I12)
C
C
R (1) = -1
S (1) = -1
R (2) = 1
S (2) = -1
R (3) = 1
S (3) = 1
R (4) = -1
S (4) = 1
C
R (5) = 0
S (5) = -1
R (6) = 1

```

```

S (6)=0
R (7)=0
S (7)=1
R (8)=-1
S (8)=0

DO 344 I=1,8
  CL1 (I)=GCL1 (ND (I), 1)
  CM1 (I)=GCL1 (ND (I), 2)
  CN1 (I)=GCL1 (ND (I), 3)
  CL2 (I)=GCL2 (ND (I), 1)
  CM2 (I)=GCL2 (ND (I), 2)
  CN2 (I)=GCL2 (ND (I), 3)
  CL3 (I)=GCL3 (ND (I), 1)
  CM3 (I)=GCL3 (ND (I), 2)
  CN3 (I)=GCL3 (ND (I), 3)
C
344 CONTINUE
346 FORMAT (112,9F7.4)
C
DO 348 I=1,40
  PD (I)=0.0
348 CONTINUE
C
H (1)=1.0
H (2)=1.0
C
WRITE (6,*) SM (1, 1), SM (2, 2)
P (1)=0.577352692
P (2)=-P (1)
C
HH (1)=0.3478548451
HH (2)=H (1)
HH (3)=0.6521451548
HH (4)=H (3)
PP (1)=0.8611363115
PP (2)=-P (1)
PP (3)=0.3399810435
PP (4)=-P (3)
C
DO 150 I=1,2
  DO 150 J=1,2
    DO 150 K=1,2
      U=P (I)
      V=P (J)
      W=P (K)
C
WRITE (6, 157) IL
C
CALL CBUPDT (111, IL, ND, I, J, K, U, V, W, X, Y, Z, VR (IR14), VR (IR28),
1      DETJ, VR (IR31), VR (IR32), VR (IR33), VR (IR29),
2      VR (IR37), VR (IR38), VR (IR36), VR (IR39), VR (IR40),
3      VR (IR30), VR (IR20), VR (IR47), VR (IR54), VR (IR55),
4      VR (IR57))
C
DO 150 M=1,40
  PD (M)=PD (M)+H (I)*H (J)*H (K)*PDL (M)*DETJ
C
write (6, 10) m, pd1 (m), pd1 (m), detj
C
WRITE (6,*) 'PD (M) ', PD (M)
150 CONTINUE
C
DO 151 I=1,40
  WRITE (6,*) 'PD (M) ', PD (I)
C
151 CONTINUE
C
10 format ('integ.i, pd1 (i), pd (i), DETJ is: ', 1i3, 3f13.5)
C
WRITE (6, 153) DETJ
C
153 FORMAT ('DETJ IS:', 1F12.4)

```

```

C
C
RETURN
END
C
C (end update)
C
C
SUBROUTINE GETDT (IEL, ID, IID, NEQ, MX, NHLF, NN, MEQT,
1      XX, YY, ZZ, DD1, DD2)
  IMPLICIT REAL*8 (A-H, O-Z)
  IMPLICIT INTEGER*8 (I-N)
C
C Subroutine GETDT is designed to read data from data file. The data
C needed are:
C
C      nelm: The numberr of elements in the structure.
C      nnode: The number of node of the structure.
C      nstep: The number of load step to be taken.
C      ncrtr: The max. iterations to balance the node force.
C      xx,yy,zz: initial coordinates of the nodes
C      iel(i,j): The node name, here i is the element name
C                and j is the node sequence in the local
C                corordinate.
C      id(i) (i=5*nnode): The the constrain for displacement
C      iid(i,j): The boundary constrain for displacement.
C                 here i--element j--generalized displacement.
C      dd(i,j): The load at node i corespond to the direction j.
C
C Data calculated:
C      NHBW: the half-band-width of the problem.
C      neqt: The number of equation to be solved.
C
C
C      DIMENSION IEL (NELM, 8), ID (1), IID (NNODE, 5), NEQ (NNODE, 5),
1      MX (1), NHLF (1), MN (1), MEQT (NELM, 40)
C      DIMENSION XX (1), YY (1), ZZ (1), DD1 (1), DD2 (1)
C      COMMON /SCHALR1/ NELM, NNODE, NT
C      COMMON /SCHALR2/ NEQT, NSTEP, NHBW, COEF1, COEF2, NSHOW1, NSHOW2,
1      NSHOW3, HRZ, ITRLM, FACTOR
C      COMMON /PNTRIN/ IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10
C      COMMON /PNTRRL/ IR1, IR2, IR3, IR4, IR5, IR6, IR7, IR8, IR9, IR10,
1      IR11, IR12, IR13, IR14, IR15, IR16, IR17, IR18,
2      IR19, IR20, IR21, IR22, IR23, IR24, IR25, IR26,
3      IR27, IR28, IR29, IR30, IR31, IR32, IR33, IR34,
4      IR35, IR36, IR37, IR38, IR39, IR40, IR41, IR42,
5      IR43, IR44, IR45, IR46, IR47, IR48, IR49, IR50
C      COMMON /RLVEC/ VR (1)
C      COMMON /INTVEC/ IPT (1)
C      COMMON /MTL/ E, EU
C      COMMON /GEO/ TO
C      COMMON /DISCT/ NDC, NDBC
C      COMMON /OUTVR/ NPT, NPV
C
C
C      WRITE (6, 10) NELM
10  FORMAT (' THE NUMBER OF ELEMENT IS: ', I13)
C      WRITE (6, 20) NNODE
20  FORMAT (' THE NUMBER OF NODES IS: ', I15)
C      WRITE (6, 30) E, EU
30  FORMAT (' THE MATERIAL CONSTANTS E AND NU ARE: ', 2F13.3)
C      WRITE (6, *) ' THE THICKNESS OF THE SHELL IS: ', TO
C      DO 100 NODE=1, NNODE
C          READ (5, *) KK, XX (NODE), YY (NODE), ZZ (NODE)
C          WRITE (6, 101) NODE, XX (NODE), YY (NODE), ZZ (NODE)
100 CONTINUE
101 FORMAT (' THE COORDINATES OF NODE ', I12, ' IS: ', 3F12.5)
C
C

```

```

DO 106 I=1,NNODE
  READ(5,*) KK, (IID(I,J),J=1,5)
  WRITE(6,107) I, (IID(I,J),J=1,5)
  WRITE(6,107) I,IID(I,1),IID(I,2),IID(I,3),
  1 IID(I,4),IID(I,5)
106 CONTINUE
107 FORMAT(' THE CONSTRAIN AT NODE ',113,' IS',513)
  NDBC=0
  DO 108 I=1,NNODE
    DO 108 J=1,5
      ID(1*5-5+J)=IID(I,J)
      IF(ID(1*5-5+J).EQ.2) NDBC=NDBC+1
108 CONTINUE
  NDC=0
  IF(NDBC.NE.0) NDC=1
C
C WRITE(6,*) 'The first group load is:'
C DO 110 I=1,NNODE
C   READ(5,*) KK, (DD1(1*5-5+J),J=1,5)
C   K=1*5-5
C   WRITE(6,114) I,DD1(K+1),DD1(K+2),DD1(K+3),DD1(K+4),DD1(K+5)
C 110 CONTINUE
C
C   WRITE(6,*) 'THE SECOND GROUP LOAD IS:'
C   DO 112 I=1,NNODE
C     READ(5,*) KK, (DD2(1*5-5+J),J=1,5)
C     K=1*5-5
C     WRITE(6,114) I,DD2(K+1),DD2(K+2),DD2(K+3),DD2(K+4),DD2(K+5)
112 CONTINUE
114 FORMAT('THE LOAD CORRESP. TO NODE ',112,' IS: ',5F8.3)
C
C   DO 122 I=1,NELM
C     READ(5,*) KK, (IEL(I,J),J=1,8)
C     WRITE(6,126) I,IEL(I,1),IEL(I,2),IEL(I,3),IEL(I,4),
C     1 IEL(I,5),IEL(I,6),IEL(I,7),IEL(I,8)
122 CONTINUE
C
C   READ(15,*) NPT,NPV
126 FORMAT(' THE NODE NUMBER FOR ELEMENT ',112,' IS: ',814)
C
C   Next part is to calculate the half band width of the stiffness matrix.
C
C   For every unknown disp. get the correspond equation number: NEL(I,J)
C   K=1
C   DO 200 I=1,NNODE
C     DO 200 J=1,5
C       IF(IID(I,J).EQ.1) THEN
C         NEQ(I,J)=0
C       ELSE
C         IF(IID(I,J).EQ.0) THEN
C           NEQ(I,J)=K
C           K=K+1
C         END IF
C       END IF
200 CONTINUE
  NEQT=K-1
  WRITE(6,400) NEQT
400 FORMAT('THE NUMBER OF EQUATIONS IS: ',116)
C
C   CALL MNU(NELM,40,MEQT)
C
C   Get all the equation number in element i : MEQT(I,K) (k=1..40) here.
C
C   DO 240 I=1,NELM
C     K=1
C     DO 240 J=1,8

```



```

C      DO 240 M=1,5
C          MEQT (I,K)=NEQ (IEL (I,J),M)
C          WRITE (6,500) I,K,MEQT (I,K)
C          K=K+1
C 240 CONTINUE
C
C 500 FORMAT ('THE EQ. NUMBER IN ELM(I) (K=1..40) IS: ',3I6)
C      DO 600 K=1,40
C          WRITE (6,515) K,MEQT (I,K)
C 600 CONTINUE
C 515 FORMAT ('THE MEQT (I,K) IS:',2I5)
C
C      Get the max and min eq. number in an element. The difference is the
C      half-band-width of the stiffness matrix in the element
C
C      DO 280 I=1,NELM
C          MX (I)=0
C          MN (I)=NT
C          DO 300 K=1,40
C              IF (MEQT (I,K).GT.MX (I)) THEN
C                  MX (I)=MEQT (I,K)
C                  WRITE (6,490) I,K,MEQT (I,K),MX (I)
C 490          FORMAT ('I,K,MEQT (I,K),MX (I):',4I5)
C              END IF
C              IF ((MEQT (I,K).GT.0).AND.(MEQT (I,K).LT.MN (I))) THEN
C                  MN (I)=MEQT (I,K)
C              END IF
C 300 CONTINUE
C          NHLF (I)=MX (I)-MN (I)
C          WRITE (6,460) I,MX (I),MN (I),NHLF (I)
C 280 CONTINUE
C 460 FORMAT ('The max,min and half band width in el(i) is: ',4I5)
C
C      Get the half-band-width of the stiffness matrix of the structure
C
C      NHBW=0
C      DO 320 I=1,NELM
C          IF (NHLF (I).GT.NHBW) NHBW=NHLF (I)
C 320 CONTINUE
C
C          WRITE (6,440) NHBW
C 440 FORMAT ('THE HALF-BAND-WIDTH OF THE STIFFNESS MATRIX IS: ',1I5)
C      RETURN
C      END
C
C
C      SUBROUTINE CRITRI (II,ND,D,FRCINC,ACTFRC,DDD,VLIMN,ICNC1,VALS)
C      IMPLICIT REAL*8 (A-H,O-Z)
C      IMPLICIT INTEGER*8 (I-N)
C
C      Subroutine CRITRI is to build an exit criteria for the equilibrium
C      iterations.
C      input:
C      ii = The ii'th number iteration
C      DLDINC = The load increament
C      DLOADT = Te load level at that iteration.
C      PLD = The node force in last iteration
C      DVEC = The unknown solved in last iteration
C      VLINIT = the criteria value calculated in the first iteration.
C      Output:
C      ICONCL = The conclusion : Exit the loop or not.
C              1 = exit
C              0 = Keep inside the loop.
C
C      DIMENSION D (1),FRCINC (1),ACTFRC (1),DDD (1)
C      COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10

```

```

COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3 IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4 IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5 IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /RLVEC/ VR(1)
COMMON /INTVEC/ IPT(1)
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
COMMON /DISCT/ NDC,NDBC

C
C
AINS=0.0
COEFF=90.0
VLIMNO=VLIMN
VAL=0.0
IF (11.EQ.1) THEN
  VLIMN=0.0
  DO 10 I=1,ND
    C IF (NDC.EQ.0) THEN
    C TEMP=D(I)*ROOT-FRCINC(I)
    C ELSE
    C TEMP=DDD(I)*ROOT-FRCINC(I)
    C END IF
    C AINS=AINS+TEMP
    C VLIMN=VLIMN+TEMP*TEMP
    C IF (1.LT.11) THEN
    C WRITE (6,90) 11,I,D(I)*ROOT,FRCINC(I),ACTFRC(I)
    C END IF
    C WRITE (6,80) 11,I,D(I)*ROOT,FRCINC(I),TEMP,VAL
    C 80 FORMAT ('11,I,D(I),FRCINC,TEMP: ',2I4,4F12.3)
    C 10 CONTINUE
    C VLIMN=SQRT(VLIMN)
    C VAL=VLIMN
    C WRITE (6,*) 'VAL=',VAL
    C ELSE
    C DO 20 I=1,ND
    C IF (NDC.EQ.0) THEN
    C TEMP=D(I)*ROOT-FRCINC(I)
    C ELSE
    C TEMP=DDD(I)*ROOT-FRCINC(I)
    C END IF
    C VAL=VAL+TEMP*TEMP
    C AINS=AINS+TEMP
    C IF ((1.EQ.2).OR.(1.EQ.7)) THEN
    C IF (1.LT.10) THEN
    C WRITE (6,90) 11,I,D(I)*ROOT,FRCINC(I),ACTFRC(I)
    C END IF
    C 90 FORMAT ('11,I,D(I),FRCINC,ACTF: ',2I4,3F14.6)
    C 20 CONTINUE
    C VAL=SQRT(VAL)
    C END IF
    C WRITE (6,*) 'AINS ',AINS
    C
    C ICNC1=0
    C VALS=VAL*COEFF
    C IF (VLIMN.GT.10.0) VLIMN=10.0
    C IF (NDC.EQ.1.AND.VLIMN.LT.0.005) ICNC1=1
    C IF (VALS.LT.VLIMNO) ICNC1=1
    C WRITE (6,50) VAL*COEFF,VLIMN,ICNC1
    C 50 FORMAT ('VAL1,CRIT1,CONCL ARE: ',2F14.4,1I3)
    C
    C RETURN

```

END

SUBROUTINE CRITR3(II,ND,D,FRCINC,ACTFRC,DDD,VLIMN,ICONCL,VALS)  
IMPLICIT REAL\*8(A-H,O-Z)  
IMPLICIT INTEGER\*8(I-N)

Subroutine CRITR3 is to build an exit criteria for the equilibrium iterations

input:

ii = The ii'th number iteration

DLDINC = The load increament

DLOADT = Te load level at that iteration.

PLD = The node force in last iteration

DVEC = The unknown solved in last iteration

VLINIT = the criteria value calculated in the first iteration.

Output:

ICONCL = The conclusion : Exit the loop or not.

1 = exit

0 = Keep inside the loop.

DIMENSION D(1),FRCINC(1),ACTFRC(1),DDD(1)  
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10  
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,  
1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,  
2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,  
3 IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,  
4 IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,  
5 IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50

COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59  
COMMON /SCHALR1/ NELM,NNODE,NT  
COMMON /RLVEC/ VR(1)  
COMMON /INTVEC/ IPT(1)  
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP  
COMMON /DISCT/ NDC,NDBC

AINS=0.0  
COEFF=50.0  
ZR=0.0  
VLIMNO=VLIMN  
VAL=0.0  
IF(II.EQ.1) THEN  
VLIMN=0.0  
DO 10 I=1,ND  
IF(NDC.EQ.0) THEN  
TEMP=-FRCINC(I)  
ELSE  
TEMP=DDD(I)\*ROOT-FRCINC(I)  
END IF  
AINS=AINS+TEMP  
VLIMN=VLIMN+TEMP\*TEMP  
IF(I.LT.11) THEN  
WRITE(6,90) II,I,ZR,FRCINC(I),ACTFRC(I)  
END IF  
WRITE(6,80) II,I,D(I)\*ROOT,FRCINC(I),TEMP,VAL  
80 FORMAT('II,I,D(I),FRCINC,TEMP:',2I4,4F12.3)  
10 CONTINUE  
VLIMN=SQRT(VLIMN)  
VAL=VLIMN  
WRITE(6,\*) 'VAL=',VAL  
ELSE  
DO 20 I=1,ND  
IF(NDC.EQ.0) THEN  
TEMP=-FRCINC(I)

```

C          ELSE
C          TEMP=DDD (I) *ROOT-FRCINC (I)
C          END IF
C          VAL=VAL+TEMP*TEMP
C          AINS=AINS+TEMP
C          IF ((I.EQ.2).OR.(I.EQ.7)) THEN
C          IF (I.LT.10) THEN
C          WRITE (6,90) I,I,ZR,FRCINC (I),ACTFRC (I)
C          END IF
90      FORMAT ('I,I,D (I),FRCINC,ACTF: ',2I4,3F14.6)
20      CONTINUE
      VAL=SQRT (VAL)
      END IF
      ICNC1=0
      VALS=VAL*COEFF
      IF (VLIMN.GT.10.0) VLIMN=10.0
      IF (NDC.EQ.1.AND.VLIMN.LT.0.005) ICNC1=1
      IF (VALS.LT.VLIMNO) ICNC1=1
      WRITE (6,50) VAL*COEFF,VLIMN,ICNC1
50     FORMAT ('VAL1,CRIT1,CONCL ARE: ',2F14.4,1I3)
C
      RETURN
      END
C
C
C      SUBROUTINE CMPT1
C
C      CMPT1 is used to make a initial arrangement of
C      the real and integer vector.
C      The parameters are :
C          NEIM  -- The number of elements in the shell.
C          NNODE -- The number of nodes in the shell.
C          NT    -- NNODE*5
C          ND    -- The number of unknown displacements.
C          NO    -- 2*nd
C          NSTEP -- Number of load steps.
C
C      IMPLICIT REAL*8 (A-H,O-Z)
C      IMPLICIT INTEGER*8 (I-N)
C      CHARACTER TITLE*80
C      COMMON /SCHALR1/ NELM,NNODE,NT
C      COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,
1      NSHOW3,HRZ,ITRLM,FACTOR
C      COMMON /MTL/ E,EU
C      COMMON /LNGTHI/ L11,L12,L13,L14,L15,L16,L17,L18,L19,L110
C      COMMON /LNGTHR/ LR1,LT2,LR3,LR4,LR5,LR6,LR7,LR8,LR9,LR10,
1      LR11,LR12,LR13,LR14,LR15,LR16,LR17,LR18,
2      LR19,LR20,LR21,LR22,LR23,LR24,LR25,LR26
C      COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
C      COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1      IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2      IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3      IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4      IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5      IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
C      COMMON /UNICT/ NCONS,MODEL,ETAA,TDELTA,TINIT
C      COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
C      COMMON /RLVEC/ VR (1)
C      COMMON /INTVEC/ IPT (1)
C      COMMON /GEO/ TO
C      COMMON /CONTN/ INSIDT,KPDT,DTLM1
C      COMMON /BOD/ DO,ZCO,ZC1,ZC2,ZC3,ZM1,ZM2,CA1,CA2,CR1,CR2,ZNO
C      COMMON /WAL/ WK,WB,WN2,WN3,WN4,WN5,WN6,WN8,WN9,WN10,WN11,WRO
C      COMMON /SQ/ SQQ
C      COMMON /DISCT/ NDC,NDBC
C      COMMON /CREEP/ ICRP,NBCRP,NBDN,CRPTM,IPON

```

```

COMMON /CRPC/ CRPC1,CRPC2
COMMON /TMPEF/ IDO,NTEM,NITR,NANM,CEXPN,TMMIN,TMINC,TMMAX,TMPP
COMMON /OUTVR/ NPT,NPV

READ (5,*) NELM,NNODE,NSTEP,ITRLM,E,EU,TO,COEF1,COEF2,FACTOR,
1 NSHOW1,NSHOW2,NSHOW3,INSIDT,KPDT,DTLMI,SQQ
WRITE (6,20) NELM,NNODE
20 FORMAT ('THE NUMBER OF ELEMENTS IS:',I13,' THE NUMBER OF NODE IS:'
1 ,I14)

READ (5,*) IDO,NTEM,NITR,NANM,CEXPN,TMMIN,TMINC,TMMAX
READ (5,*) NCONS,MODEL,ETAA,TDELTA
TINIT=TDELTA
READ (5,*) ICRP,NBCRP
READ (5,*) NPT,NPV
CRPC1=1.0
CRPC2=1.0
IF (NCONS.EQ.0.AND.ICRP.EQ.1) THEN
WRITE (6,*) 'ELASTIC MODEL CAN NOT BE USED TO CALCULATE CREEP.'
1 STOP
STOP
END IF
MODEL=1..BODNER, MODEL=2..WALKER
READ (5,*) DO,ZC0,ZC1,ZC2,ZC3,ZM1,ZM2,CA1,CA2,CR1,CR2,ZNO
READ (5,*) WK,WB,WN2,WN3,WN4,WN5,WN6,WN8,WN9,WN10,WN11,WRO
NE8=NELM*8
ND3=NNODE*3
ND5=NNODE*5
NT=ND5
ND5S=ND5*ND5

L11=NE8
L12=ND5
L13=ND5
L14=ND5
L15=ND5
L16=NELM
L17=NELM
L18=NELM*40
L19=NDBC
L110=NDBC

LRI=NNODE
LR2=NNODE
LR3=NNODE
LR4=ND5
LR5=ND5
LR6=ND5
LR7=NSTEP

IP1=1
IP2=IP1+L11
IP3=IP2+L12
IP4=IP3+L13
IP5=IP4+L14
IP6=IP5+L15
IP7=IP6+L16
IP8=IP7+L17
IP9=IP8+L18
IP10=IP9+L19
IP11=IP10+L110
WRITE (6,*) 'NUMBER OF INTEGER:',IP11

IR1=1
IR2=IR1+LR1
IR3=IR2+LR2

```



```

LR8=NEQT
LR9=ND5
LR10=ND5
LR11=ND5
LR12=ND5
LR13=ND5
LR14=ND5
LR15=ND5
LR16=ND5
LR17=NEQT
LR18=ND5
LR19=ND5S
LR20=NELM*72
LR21=1600
LR22=40
C   LR23=NEQT*(NEQT+1)/2
LR23=1
C   LR23 is for P(I), if use skylight, then active it.
LR24=NEQT*NEQT
LR25=1600
LR26=ND5
LR27=ND5
LR28=1600
LR29=1600
LR30=1600
LR31=360
LR32=360
LR33=360
LR34=81
LR35=81
LR36=36
LR37=36
LR38=36
LR39=36
LR40=36
LR41=1
C   LR41=LR23
LR42=ND5
LR43=NNODE
LR44=NNODE
LR45=NNODE
LR46=ND5
LR47=NELM*72
LR48=ND5
LR49=ND5
LR50=ND5
LR51=NELM*96
C   2*2*2*12=96 FOR BOTH BODNER AND WALKER'S MODEL
C   64=2*2*2*(6+1) 6=BETA(I,J) 7=Zi (THE STATE VARIABLE FOR
C   BODNER'S MODEL
LR52=ND5
LR53=6
LR54=NELM*8*24*6
LR55=NELM*8*24
LR56=NELM*8*6
LR57=NELM*8*36
LR58=NELM*8*12
LR59=ND5
LR60=NNODE*3
LR61=NNODE*3
LR62=NNODE*3
LR63=NNODE*3
LR64=NNODE*3
LR65=NNODE*3
LR66=NDBC*NDBC
LR67=NDBC*NEQT

```

LR68=LR67  
LR69=NDBC  
LR70=NDBC  
LR71=NDBC  
LR72=NDBC  
LR73=NDBC  
LR74=NDBC  
LR75=NDBC

IP1=1  
IP2=IP1+L11  
IP3=IP2+L12  
IP4=IP3+L13  
IP5=IP4+L14  
IP6=IP5+L16  
IP7=IP6+L17  
IP8=IP5+L15  
IP9=IP8+L18

IR9=IR8+LR8  
IR10=IR9+LR9  
IR11=IR10+LR10  
IR12=IR11+LR11  
IR13=IR12+LR12  
IR14=IR13+LR13  
IR15=IR14+LR14  
IR16=IR15+LR15  
IR17=IR16+LR16  
IR18=IR17+LR17  
IR19=IR18+LR18  
IR20=IR19+LR19  
IR21=IR20+LR20  
IR22=IR21+LR21  
IR23=IR22+LR22  
IR24=IR23+LR23  
IR25=IR24+LR24  
IR26=IR25+LR25  
IR27=IR26+LR26  
IR28=IR27+LR27  
IR29=IR28+LR28  
IR30=IR29+LR29  
IR31=IR30+LR30  
IR32=IR31+LR31  
IR33=IR32+LR32  
IR34=IR33+LR33  
IR35=IR34+LR34  
IR36=IR35+LR35  
IR37=IR36+LR36  
IR38=IR37+LR37  
IR39=IR38+LR38  
IR40=IR39+LR39  
IR41=IR40+LR40  
IR42=IR41+LR41  
IR43=IR42+LR42  
IR44=IR43+LR43  
IR45=IR44+LR44  
IR46=IR45+LR45  
IR47=IR46+LR46  
IR48=IR47+LR47  
IR49=IR48+LR48  
IR50=IR49+LR49  
IR51=IR50+LR50  
IR52=IR51+LR51  
IR53=IR52+LR52  
IR54=IR53+LR53  
IR55=IR54+LR54



```

IR56=IR55+LR55
IR57=IR56+LR56
IR58=IR57+LR57
IR59=IR58+LR58
IR60=IR59+LR59
IR61=IR60+LR60
IR62=IR61+LR61
IR63=IR62+LR62
IR64=IR63+LR63
IR65=IR64+LR64
IR66=IR65+LR65
IR67=IR66+LR66
IR68=IR67+LR67
IR69=IR68+LR68
IR70=IR69+LR69
IR71=IR70+LR70
IR72=IR71+LR71
IR73=IR72+LR72
IR74=IR73+LR73
IR75=IR74+LR74

```

C

```

WRITE (6,*) 'INTEGER=',IP9
MEMOR=IR75+LR75
IF (MEMOR.LT.MAXR) THEN
WRITE (6,*) 'THE PREDIFINED MEMORY IS NOT ENOUGH.'
WRITE (6,*) 'MEMORY: ',MEMOR
STOP
END IF
WRITE (6,*) 'MEMORY: ',MEMOR
IF (MEMOR.GT.100) STOP

```

C

C

```

RETURN
END

```

C

C

C\*\*\*\*\*

C Subroutine Bolsul is the solution phase using Bodner's constitutive equation.

C Inputs are:

C BL used to find the local strain.

C VFE the displace increament. epsln=bl.vfe

C SVT3D and SVBLD are the data calculated in the processing face.

C State variable BETA(..7) (1..6-directional, 7-isotropic) are updated.

C The derivative of the state variables STVDF and the derivative of the

C nonlinear strain EPSND are calculated.

C The stress increament is also calculated.

C\*\*\*\*\*

C

```

CALL BODSUL (IL,II, JJ, KK, VR (IR31), VR (IR29), VR (IR54),
1 VR (IR55), VR (IR51), SD, VR (IR56), VR (IR57))

```

C

```

SUBROUTINE BODSUL (IAA, IA, IB, IC, BL, VFE, SVT3D, SVBLD, BETA, SD,
1 BDSV, EM4, AA)

```

C

```

IMPLICIT REAL*8 (A-H, O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION BL (6, 40), VFE (1), SVT3D (NELM, 2, 2, 2, 144), TMVEC (24),
1 SVBLD (NELM, 2, 2, 2, 24), BETA (NELM, 2, 2, 2, 12), SD (6, 1),
2 BDSV (NELM, 2, 2, 2, 6), EM4 (NELM, 2, 2, 2, 36),
3 DLBET (6), TMV (19), AA (6, 1)

```

C

```

COMMON /SCHALR1/ NELM, NNODE, NT
COMMON /SCHALR2/ NEQT, NSTEP, NHBW, COEF1, COEF2, NSHOW1, NSHOW2,
1 NSHOW3, HRZ, ITRLM, FACTOR
COMMON /PNTRIN/ IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10

```

```

COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3 IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4 IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5 IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /RLVEC/ VR(1)
COMMON /INTVEC/ IPT(1)
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
COMMON /GEO/ TO
COMMON /CNTRL/ DETMNT
COMMON /CONTN/ INSIDT,KPDT,DTLM1
COMMON /UNICT/ NCONS,MODEL,ETAA,TDELT,TINIT
COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
COMMON /BOD/ DO,ZCO,ZC1,ZC2,ZC3,ZM1,ZM2,CA1,CA2,CR1,CR2,ZNO
COMMON /CREEP/ ICRP,NBCRP,NBDN,CRPTM,IPON

C
IPR=0
IF ((IA.EQ.1).AND.(IB.EQ.1).AND.(IC.EQ.1)) IPR=1

C
C
59 FORMAT(6F12.4)
C if(ipr.eq.1) then
C do 220 i=1,19
C write(6,59) (-svt3d(iaa,ia,ib,ic,i*6-6+j),j=1,6)
C220 continue
C end if
DO 80 I=1,19
TMVEC(I)=0.0
DO 80 J=1,6
TMVEC(I)=TMVEC(I)-SVT3D(IAA,IA,IB,IC,I*6-6+J)*AA(J,1)
80 CONTINUE
C if(ipr.eq.1) write(6,*) ' vblD, tmv, tmvec in FACE2: '
DO 60 I=1,19
TMV(I)=TMVEC(I)
TMVEC(I)=SVBLD(IAA,IA,IB,IC,I)+TMVEC(I)
C IF(IPR.EQ.1) then
C write(6,*) I,' ',svbld(iaa,ia,ib,ic,i),' ',TMV(I),' ',tmvec(i)
C end if
60 CONTINUE
C
DO 100 I=1,6
SD(I,1)=TMVEC(I)
DLBET(I)=TMVEC(I+13)
C WRITE(6,*) I,' D(Zd/DT): ',STVDF(IAA,IA,IB,IC,I)
100 CONTINUE
C
C IF(IPR.EQ.1) THEN
C WRITE(6,*) 'PUELAS:'
C WRITE(6,8) (TMV(I),I=1,6)
C WRITE(6,8) (SD(I,1),I=1,6)
C 8 FORMAT(6F12.8)
C END IF
C
DO 120 I=1,6
BETA(IAA,IA,IB,IC,I)=BETA(IAA,IA,IB,IC,I)+DLBET(I)
IF(BETA(IAA,IA,IB,IC,I).GT.ZC3) BETA(IAA,IA,IB,IC,I)=ZC3
IF(BETA(IAA,IA,IB,IC,I).LT.-ZC3) BETA(IAA,IA,IB,IC,I)=-ZC3
C WRITE(6,*) I,' BETA: ',BETA(IAA,IA,IB,IC,I)
120 CONTINUE
BETA(IAA,IA,IB,IC,7)=BETA(IAA,IA,IB,IC,7)+TMVEC(13)
IF(BETA(IAA,IA,IB,IC,7).GT.ZC1) BETA(IAA,IA,IB,IC,7)=ZC1
IF(BETA(IAA,IA,IB,IC,7).LT.(2.0*ZCO-ZC1)) BETA(IAA,IA,IB,IC,7)=
1 2.0*ZCO-ZC1
C if(ipr.eq.1) WRITE(6,*) '8 =Zi BETA: ',BETA(IAA,IA,IB,IC,7)
C

```

C STVDF(1) is the dirivative of the undirectional variable.  
 C BETA(7) is the undirectional variable.

C RETURN  
 C END  
 C END (BODSOL)

C \*\*\*\*\*  
 C \* Subroutine Bodner is to prepare the stiffness matrix and the \*  
 C \* residue force. Input is the state variable and current stress. \*  
 C \* Output is EM2 (to form stiffness matrix by cb), BDL \*  
 C \* (to form the force term by cb), SVT3D and SVBLD (will be used \*  
 C \* in the sulation face) \*  
 C \*\*\*\*\*

C SUBROUTINE BODNER(III,IAA,IA,IB,IC,SIG,ZZZ,EM2,S,BETA,BDL,  
 1 SVT3D,SVBLD,ZZR,BDSV,EM4,AINV)

C IMPLICIT REAL\*8 (A-H,O-Z)  
 C IMPLICIT INTEGER\*8 (I-N)

C DIMENSION SIG (3,3),ZZZ (19,19),EM2 (6,6),S (3,3),BETA (NELM,2,2,2,12),  
 1 BDL (1),SVT3D (NELM,2,2,2,144),SVBLD (NELM,2,2,2,24),  
 2 ZZR (19,6),VEC1 (19),VCTL (19),GA (19),BETAA (7),AINV (1),  
 3 VEPS (6),SS (6),SECTM (6),T3D (19,6),VEPSLN (3,3),  
 4 BDSV (NELM,2,2,2,6),EM4 (NELM,2,2,2,36),SIGVC (6)  
 5 ,AAA (6,6),BBB (6,6),CCC (6,6),DDD (6,6),VECC (19)

C COMMON /BOD/ DO,ZCO,ZC1,ZC2,ZC3,ZM1,ZM2,CA1,CA2,CR1,CR2,ZNO  
 COMMON /UNICT/ NCONS,MODEL,ETAA,TDELTA,TINIT  
 COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59  
 COMMON /SCHALR1/ NELM,NNODE,NT  
 COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,  
 1 NSHOW3,HRZ,ITRLM,FACTOR  
 COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10  
 COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,  
 1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,  
 2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,  
 3 IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,  
 4 IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,  
 5 IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50

C COMMON /RLVEC/ VR (1)  
 COMMON /INTVEC/ IPT (1)  
 COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP  
 COMMON /GEO/ TO  
 COMMON /CNTRL/ DETMNT  
 COMMON /CONTN/ INSIDT,KPDT,DTLM1  
 COMMON /ABDFST/ ISEC  
 COMMON /NCTT/ NCT (12,2,2,2)  
 COMMON /NMBITR/ NUM

C ZNO,DO are input constants in kinematical equation.

C ACS,ZC1,ZC2,ZC3,CM1,CM2,CR1,CR2 are constants in state variable equations.

C S(i,j) is the stress deviator  
 C  $DJ2=1/2*S(I,J)*S(I,J)$   
 C  $SJ2=SIG(I,J)*SIG(I,J)$   
 C  $ZV1=Zi$   
 C  $SIGM(6)---SIG(3,3) \dots\dots\dots$   
 C  $VSTV=D(Z)/DT$   
 C  $VSTV1=D(ZV1)/DT$

C  $ET=-ETA*TDELTA$  where eta and deltat are given.

```

C      IPR=0
      IF ((IA.EQ.1).AND.(IB.EQ.1).AND.(IC.EQ.1)) IPR=1
C
      DO 20 I=1,7
        BETAA(I)=BETA(IAA,IA,IB,IC,I)
20    CONTINUE
C      WRITE(6,*) 'NUM=',NUM
      IF ((NUM.EQ.1.OR.NUM.EQ.2).AND.(INSIDT.NE.1)) THEN
        BETA(IAA,IA,IB,IC,7)=ZCO
        ZV1=ZCO
      ELSE
        ZV1=BETA(IAA,IA,IB,IC,7)
      END IF
C
      ET=-ETAA*TDELTA
C
      SAV=(SIG(1,1)+SIG(2,2)+SIG(3,3))/3.0
C
      IF (IPR.EQ.1) THEN
C        WRITE(6,*) 'SIGMA IN BODNER'
C        DO 80 I=1,3
C          WRITE(6,32) (SIG(I,J),J=1,3)
C      80  CONTINUE
C      32  FORMAT(3F12.4)
C        END IF
C
      DO 90 I=1,3
        DO 90 J=1,3
          IF (I.EQ.J) THEN
            S(I,J)=SIG(I,J)-SAV
          ELSE
            S(I,J)=SIG(I,J)
          END IF
        90  CONTINUE
C
      DJ2=0.0
      SJ2=0.0
C
      DO 100 I=1,3
        DO 100 J=1,3
          DJ2=DJ2+0.5*S(I,J)*S(I,J)
          SJ2=SJ2+SIG(I,J)*SIG(I,J)
100    CONTINUE
C      IF (IPR.EQ.1) WRITE(6,*) 'DJ2,SJ2 IS: ',DJ2,SJ2
C
C      ZZ is state variable. ZZ=Zi+Zd
C      Now calculate ZD and ZZ
C
      ZD=0.0
      ZD=SIG(1,1)*BETAA(1)+SIG(2,2)*BETAA(2)+SIG(3,3)*BETAA(3)
      +2*(SIG(1,2)*BETAA(4)+SIG(2,3)*BETAA(5)+SIG(1,3)*BETAA(6))
      ZD=ZD/SJ2**0.5
      ZZ=ZV1+ZD
      ZZ2=ZZ*ZZ
      IF (IPR.EQ.1) THEN
        WRITE(6,*) 'STATE VAR Z1,ZD,ZZ ',ZV1,ZD,ZZ
      END IF
C
C      WRITE(6,*) 'CONTTOL VAR: ',0.5*(ZZ2/DJ2/3.0)**ZNO
      IF ((0.5*(ZZ2/DJ2/3.0)**ZNO).GT.60) THEN
        FAC1=0.0
      ELSE
C        WRITE(6,*) 'COMMING'
        FAC1=DO*(EXP(-0.5*(ZZ2/DJ2/3.0)**ZNO))/DJ2**0.5
      END IF

```

```

DO 40 I=1,3
  DO 40 J=1,3
    VEPSLN(I,J)=S(I,J)*FAC1
40 CONTINUE
C
VEPS(1)=VEPSLN(1,1)
VEPS(2)=VEPSLN(2,2)
VEPS(3)=VEPSLN(3,3)
VEPS(4)=VEPSLN(1,2)
VEPS(5)=VEPSLN(2,3)
VEPS(6)=VEPSLN(1,3)

NCT(IAA,IA,IB,IC)=1
C
  if (i.pr.eq.1) then
C    write(11,*)
C    end if
C    write(11,253) (veps(i),i=1,6)
253 FORMAT(6F12.10)
VEPSLN(1,1)=VEPS(1)
VEPSLN(2,2)=VEPS(2)
VEPSLN(3,3)=VEPS(3)
VEPSLN(1,2)=VEPS(4)
VEPSLN(2,3)=VEPS(5)
VEPSLN(1,3)=VEPS(6)
C
SS(1)=S(1,1)
SS(2)=S(2,2)
SS(3)=S(3,3)
SS(4)=S(1,2)
SS(5)=S(2,3)
SS(6)=S(1,3)
C
SIGVC(1)=SIG(1,1)
SIGVC(2)=SIG(2,2)
SIGVC(3)=SIG(3,3)
SIGVC(4)=SIG(1,2)
SIGVC(5)=SIG(2,3)
SIGVC(6)=SIG(1,3)
C
FAC1=FAC1*ET
C Now -eta*deltat is included in the formula in first 6*6 matrix.
C
FAC2=ZZ2*ZNO*(ZZ2/DJ2/3.0)**(ZNO-1.0)/6.0/DJ2/DJ2-0.5/DJ2
FAC3=FAC1*FAC2
FAC4=-FAC1*ZNO*(1/3.0/DJ2)**ZNO*(ABS(ZZ)**(2.0*ZNO-1.0))
IF (ZZ.GT.0.0) THEN
  FAC4=FAC4
ELSE
  FAC4=-FAC4
END IF
C
FAC5=FAC4/(SJ2)**0.5
C
CALL MNU(19,19,ZZZ)
C
ZZZ(7,1)=FAC1*(2.0/3.0+S(1,1))*S(1,1)*FAC2)
ZZZ(7,2)=FAC1*(-1.0/3.0+S(1,1))*S(2,2)*FAC2)
ZZZ(7,3)=FAC1*(-1.0/3.0+S(1,1))*S(3,3)*FAC2)
ZZZ(7,4)=FAC3*S(1,1)*S(1,2)
ZZZ(7,5)=FAC3*S(1,1)*S(2,3)
ZZZ(7,6)=FAC3*S(1,1)*S(1,3)
C
ZZZ(8,1)=FAC1*(-1.0/3.0+S(2,2))*S(1,1)*FAC2)
ZZZ(8,2)=FAC1*(2.0/3.0+S(2,2))*S(2,2)*FAC2)

```

ZZZ (8, 3) = FAC1 \* (-1.0/3.0 + S (2, 2) \* S (3, 3) \* FAC2)  
ZZZ (8, 4) = FAC3 \* S (2, 2) \* S (1, 2)  
ZZZ (8, 5) = FAC3 \* S (2, 2) \* S (2, 3)  
ZZZ (8, 6) = FAC3 \* S (2, 2) \* S (1, 3)

C

ZZZ (9, 1) = FAC1 \* (-1.0/3.0 + S (3, 3) \* S (1, 1) \* FAC2)  
ZZZ (9, 2) = FAC1 \* (-1.0/3.0 + S (3, 3) \* S (2, 2) \* FAC2)  
ZZZ (9, 3) = FAC1 \* (2.0/3.0 + S (3, 3) \* S (3, 3) \* FAC2)  
ZZZ (9, 4) = FAC3 \* S (3, 3) \* S (1, 2)  
ZZZ (9, 5) = FAC3 \* S (3, 3) \* S (2, 3)  
ZZZ (9, 6) = FAC3 \* S (3, 3) \* S (1, 3)

C

ZZZ (10, 1) = FAC3 \* S (1, 2) \* S (1, 1)  
ZZZ (10, 2) = FAC3 \* S (1, 2) \* S (2, 2)  
ZZZ (10, 3) = FAC3 \* S (1, 2) \* S (3, 3)  
ZZZ (10, 4) = FAC1 \* (1 + S (1, 2) \* S (1, 2) \* FAC2)  
ZZZ (10, 5) = FAC3 \* S (1, 2) \* S (2, 3)  
ZZZ (10, 6) = FAC3 \* S (1, 2) \* S (1, 3)

C

ZZZ (11, 1) = FAC3 \* S (2, 3) \* S (1, 1)  
ZZZ (11, 2) = FAC3 \* S (2, 3) \* S (2, 2)  
ZZZ (11, 3) = FAC3 \* S (2, 3) \* S (3, 3)  
ZZZ (11, 4) = FAC3 \* S (2, 3) \* S (1, 2)  
ZZZ (11, 5) = FAC1 \* (1 + S (2, 3) \* S (2, 3) \* FAC2)  
ZZZ (11, 6) = FAC3 \* S (2, 3) \* S (1, 3)

C

ZZZ (12, 1) = FAC3 \* S (1, 3) \* S (1, 1)  
ZZZ (12, 2) = FAC3 \* S (1, 3) \* S (2, 2)  
ZZZ (12, 3) = FAC3 \* S (1, 3) \* S (3, 3)  
ZZZ (12, 4) = FAC3 \* S (1, 3) \* S (1, 2)  
ZZZ (12, 5) = FAC3 \* S (1, 3) \* S (2, 3)  
ZZZ (12, 6) = FAC1 \* (1.0 + S (1, 3) \* S (1, 3) \* FAC2)

C

ZZZ (7, 7) = 1.0  
ZZZ (8, 8) = 1.0  
ZZZ (9, 9) = 1.0  
ZZZ (10, 10) = 1.0  
ZZZ (11, 11) = 1.0  
ZZZ (12, 12) = 1.0

C

ZZZ (7, 13) = FAC4 \* S (1, 1)  
ZZZ (8, 13) = FAC4 \* S (2, 2)  
ZZZ (9, 13) = FAC4 \* S (3, 3)  
ZZZ (10, 13) = FAC4 \* S (1, 2)  
ZZZ (11, 13) = FAC4 \* S (2, 3)  
ZZZ (12, 13) = FAC4 \* S (1, 3)

C

ZZZ (7, 14) = FAC5 \* S (1, 1) \* SIG (1, 1)  
ZZZ (8, 14) = FAC5 \* S (2, 2) \* SIG (1, 1)  
ZZZ (9, 14) = FAC5 \* S (3, 3) \* SIG (1, 1)  
ZZZ (10, 14) = FAC5 \* S (1, 2) \* SIG (1, 1)  
ZZZ (11, 14) = FAC5 \* S (2, 3) \* SIG (1, 1)  
ZZZ (12, 14) = FAC5 \* S (1, 3) \* SIG (1, 1)

C

ZZZ (7, 15) = FAC5 \* S (1, 1) \* SIG (2, 2)  
ZZZ (8, 15) = FAC5 \* S (2, 2) \* SIG (2, 2)  
ZZZ (9, 15) = FAC5 \* S (3, 3) \* SIG (2, 2)  
ZZZ (10, 15) = FAC5 \* S (1, 2) \* SIG (2, 2)  
ZZZ (11, 15) = FAC5 \* S (2, 3) \* SIG (2, 2)  
ZZZ (12, 15) = FAC5 \* S (1, 3) \* SIG (2, 2)

C

ZZZ (7, 16) = FAC5 \* S (1, 1) \* SIG (3, 3)  
ZZZ (8, 16) = FAC5 \* S (2, 2) \* SIG (3, 3)  
ZZZ (9, 16) = FAC5 \* S (3, 3) \* SIG (3, 3)  
ZZZ (10, 16) = FAC5 \* S (1, 2) \* SIG (3, 3)  
ZZZ (11, 16) = FAC5 \* S (2, 3) \* SIG (3, 3)

```

ZZZ (12, 16) = FAC5 * S (1, 3) * SIG (3, 3)
C
ZZZ (7, 17) = FAC5 * S (1, 1) * SIG (1, 2)
ZZZ (8, 17) = FAC5 * S (2, 2) * SIG (1, 2)
ZZZ (9, 17) = FAC5 * S (3, 3) * SIG (1, 2)
ZZZ (10, 17) = FAC5 * S (1, 2) * SIG (1, 2)
ZZZ (11, 17) = FAC5 * S (2, 3) * SIG (1, 2)
ZZZ (12, 17) = FAC5 * S (1, 3) * SIG (1, 2)
C
ZZZ (7, 18) = FAC5 * S (1, 1) * SIG (2, 3)
ZZZ (8, 18) = FAC5 * S (2, 2) * SIG (2, 3)
ZZZ (9, 18) = FAC5 * S (3, 3) * SIG (2, 3)
ZZZ (10, 18) = FAC5 * S (1, 2) * SIG (2, 3)
ZZZ (11, 18) = FAC5 * S (2, 3) * SIG (2, 3)
ZZZ (12, 18) = FAC5 * S (1, 3) * SIG (2, 3)
C
ZZZ (7, 19) = FAC5 * S (1, 1) * SIG (1, 3)
ZZZ (8, 19) = FAC5 * S (2, 2) * SIG (1, 3)
ZZZ (9, 19) = FAC5 * S (3, 3) * SIG (1, 3)
ZZZ (10, 19) = FAC5 * S (1, 2) * SIG (1, 3)
ZZZ (11, 19) = FAC5 * S (2, 3) * SIG (1, 3)
ZZZ (12, 19) = FAC5 * S (1, 3) * SIG (1, 3)
C
C
C
Next part is -[G, epsilon n]
C
PWR=0.0
DO 150 I=1,3
  DO 150 J=1,3
    PWR=PWR+SIG (I, J) * VEPSLN (I, J)
150 CONTINUE
C
WRITE (6, *) 'PLASTIC WORK IS: ', PWR
C
C
Row 13 is for state variable Zi.
C
FAC6 = -ZM1 * (ZC1 - ZV1)
C
IF (IPR.EQ.1) THEN
C
WRITE (6, *) 'FAC1: ', FAC1
C
WRITE (6, *) 'FAC2: ', FAC2
C
WRITE (6, *) 'FAC3: ', FAC3
C
WRITE (6, *) 'FAC4: ', FAC4
C
WRITE (6, *) 'FAC5: ', FAC5
C
WRITE (6, *) 'FAC6: ', FAC6
C
END IF
ZZZ (13, 7) = FAC6 * SIG (1, 1)
ZZZ (13, 8) = FAC6 * SIG (2, 2)
ZZZ (13, 9) = FAC6 * SIG (3, 3)
ZZZ (13, 10) = FAC6 * SIG (1, 2) * 0.5
ZZZ (13, 11) = FAC6 * SIG (2, 3) * 0.5
ZZZ (13, 12) = FAC6 * SIG (1, 3) * 0.5
C
ZZZ (13, 13) = 1.0 + ET * (-ZM1 * PWR - CA1 * CR1 *
1 (ABS ((ZV1 - ZC2) / ZC1)) ** (CR1 - 1.0))
C
C
Row 8..13 are for state variable Zd or BETAij.
C
The order for BETAij is as stress or strain: 11, 22, 33, 12, 23, 13.
C
FAC7 = ZC3 / SJ2 * 0.5
C
WRITE (6, *) 'FAC7 ', FAC7
C
FAC8 = -ZM2 * (FAC7 * SIG (1, 1) - BETAA (1))
ZZZ (14, 7) = FAC8 * SIG (1, 1)
ZZZ (14, 8) = FAC8 * SIG (2, 2)
ZZZ (14, 9) = FAC8 * SIG (3, 3)
ZZZ (14, 10) = FAC8 * SIG (1, 2) * 0.5
ZZZ (14, 11) = FAC8 * SIG (2, 3) * 0.5
ZZZ (14, 12) = FAC8 * SIG (1, 3) * 0.5

```

```

C      FAC8=-ZM2*(FAC7*SIG(2,2)-BETAA(2))
C
C      ZZZ(15,7)=FAC8*SIG(1,1)
C      ZZZ(15,8)=FAC8*SIG(2,2)
C      ZZZ(15,9)=FAC8*SIG(3,3)
C      ZZZ(15,10)=FAC8*SIG(1,2)*0.5
C      ZZZ(15,11)=FAC8*SIG(2,3)*0.5
C      ZZZ(15,12)=FAC8*SIG(1,3)*0.5
C
C      FAC8=-ZM2*(FAC7*SIG(3,3)-BETAA(3))
C
C      ZZZ(16,7)=FAC8*SIG(1,1)
C      ZZZ(16,8)=FAC8*SIG(2,2)
C      ZZZ(16,9)=FAC8*SIG(3,3)
C      ZZZ(16,10)=FAC8*SIG(1,2)*0.5
C      ZZZ(16,11)=FAC8*SIG(2,3)*0.5
C      ZZZ(16,12)=FAC8*SIG(1,3)*0.5
C
C      FAC8=-ZM2*(FAC7*SIG(1,2)-BETAA(4))
C
C      ZZZ(17,7)=FAC8*SIG(1,1)
C      ZZZ(17,8)=FAC8*SIG(2,2)
C      ZZZ(17,9)=FAC8*SIG(3,3)
C      ZZZ(17,10)=FAC8*SIG(1,2)*0.5
C      ZZZ(17,11)=FAC8*SIG(2,3)*0.5
C      ZZZ(17,12)=FAC8*SIG(1,3)*0.5
C
C      FAC8=-ZM2*(FAC7*SIG(2,3)-BETAA(5))
C
C      ZZZ(18,7)=FAC8*SIG(1,1)
C      ZZZ(18,8)=FAC8*SIG(2,2)
C      ZZZ(18,9)=FAC8*SIG(3,3)
C      ZZZ(18,10)=FAC8*SIG(1,2)*0.5
C      ZZZ(18,11)=FAC8*SIG(2,3)*0.5
C      ZZZ(18,12)=FAC8*SIG(1,3)*0.5
C
C      FAC8=-ZM2*(FAC7*SIG(1,3)-BETAA(6))
C
C      ZZZ(19,7)=FAC8*SIG(1,1)
C      ZZZ(19,8)=FAC8*SIG(2,2)
C      ZZZ(19,9)=FAC8*SIG(3,3)
C      ZZZ(19,10)=FAC8*SIG(1,2)*0.5
C      ZZZ(19,11)=FAC8*SIG(2,3)*0.5
C      ZZZ(19,12)=FAC8*SIG(1,3)*0.5
C
C      RBT=0.0
C
C      DO 160 I=1,3
C          RBT=RBT+BETAA(I)*BETAA(I)
160 CONTINUE
C      DO 170 I=4,6
C          RBT=RBT+2*BETAA(I)*BETAA(I)
170 CONTINUE
C
C      FAC9=-ET*CA2*(ZC1** (1.0-CR2)) *(RBT** ((CR2-1.0)/2.0))
C      IF (ABS(RBT) .LT. 0.000000001) THEN
C          FAC10=0.0
C      ELSE
C          FAC10=FAC9*(CR2-1.0)/RBT
C      END IF
C
C      WRITE(6,*) 'FAC9: ',FAC9,' FAC10: ',FAC10
C      EPT=-ET*PWR*ZM2
C

```



```

ZZZ (14, 14) = FAC10*BETAA (1) *BETAA (1) +1.0+FAC9+EPT
ZZZ (14, 15) = FAC10*BETAA (1) *BETAA (2)
ZZZ (14, 16) = FAC10*BETAA (1) *BETAA (3)
ZZZ (14, 17) = FAC10*BETAA (1) *BETAA (4)
ZZZ (14, 18) = FAC10*BETAA (1) *BETAA (5)
ZZZ (14, 19) = FAC10*BETAA (1) *BETAA (6)
C
ZZZ (15, 14) = FAC10*BETAA (2) *BETAA (1)
ZZZ (15, 15) = FAC10*BETAA (2) *BETAA (2) +1.0+FAC9+EPT
ZZZ (15, 16) = FAC10*BETAA (2) *BETAA (3)
ZZZ (15, 17) = FAC10*BETAA (2) *BETAA (4)
ZZZ (15, 18) = FAC10*BETAA (2) *BETAA (5)
ZZZ (15, 19) = FAC10*BETAA (2) *BETAA (6)
C
ZZZ (16, 14) = FAC10*BETAA (3) *BETAA (1)
ZZZ (16, 15) = FAC10*BETAA (3) *BETAA (2)
ZZZ (16, 16) = FAC10*BETAA (3) *BETAA (3) +1.0+FAC9+EPT
ZZZ (16, 17) = FAC10*BETAA (3) *BETAA (4)
ZZZ (16, 18) = FAC10*BETAA (3) *BETAA (5)
ZZZ (16, 19) = FAC10*BETAA (3) *BETAA (6)
C
ZZZ (17, 14) = FAC10*BETAA (4) *BETAA (1)
ZZZ (17, 15) = FAC10*BETAA (4) *BETAA (2)
ZZZ (17, 16) = FAC10*BETAA (4) *BETAA (3)
ZZZ (17, 17) = FAC10*BETAA (4) *BETAA (3) +1.0+FAC9+EPT
ZZZ (17, 18) = FAC10*BETAA (4) *BETAA (5)
ZZZ (17, 19) = FAC10*BETAA (4) *BETAA (6)
C
ZZZ (18, 14) = FAC10*BETAA (5) *BETAA (1)
ZZZ (18, 15) = FAC10*BETAA (5) *BETAA (2)
ZZZ (18, 16) = FAC10*BETAA (5) *BETAA (3)
ZZZ (18, 17) = FAC10*BETAA (5) *BETAA (4)
ZZZ (18, 18) = FAC10*BETAA (5) *BETAA (5) +1.0+FAC9+EPT
ZZZ (18, 19) = FAC10*BETAA (5) *BETAA (6)
C
ZZZ (19, 14) = FAC10*BETAA (6) *BETAA (1)
ZZZ (19, 15) = FAC10*BETAA (6) *BETAA (2)
ZZZ (19, 16) = FAC10*BETAA (6) *BETAA (3)
ZZZ (19, 17) = FAC10*BETAA (6) *BETAA (4)
ZZZ (19, 18) = FAC10*BETAA (6) *BETAA (5)
ZZZ (19, 19) = FAC10*BETAA (6) *BETAA (6) +1.0+FAC9+EPT
C
C Equation Zi+BETA(I,J)*U(I,J)=Z in increamental form.
C
C ZZZ (14, 13) =1.0
C
C SJR=1.0/SJ2**0.5
C
C
C ZZZ (1, 1) =1.0
C ZZZ (2, 2) =1.0
C ZZZ (3, 3) =1.0
C ZZZ (4, 4) =1.0
C ZZZ (5, 5) =1.0
C ZZZ (6, 6) =1.0
C
C DO 333 I=1,6
C DO 333 J=1,6
C ZZZ (I, J+6) =EM2 (I, J)
333 CONTINUE
C
C
C Now the matrix [zzz] is formed.
C Next step is to find vector part.
C
C

```

```

C      VCTL(1..6) is the difference of d(epslon)/dt and f.
C
DO 200 I=1,6
  VEC1(I+6)=TDELTA*VEPS(I)
200 CONTINUE

C      SECTM(I) is (G,epslon*d(epslon)/dt)
C
DO 220 I=1,7
  SECTM(I)=0.0
  DO 220 J=1,6
    ZZZ(I+12,J+6)=0.0
    SECTM(I)=SECTM(I)+ZZZ(I+12,J+6)*VEPS(J)
    SECTM(I)=SECTM(I)+ZZZ(I+12,J+6)*EPSND(IAA,IA,IB,IC,J)
220 CONTINUE

C      IF (IPR.EQ.1) THEN
C      WRITE(6,*) 'VEPS:'
C      WRITE(6,211) (VEPS(I),I=1,6)
C      DO 210 I=1,7
C      WRITE(6,211) (ZZZ(I+12,J+6),J=1,6)
210 CONTINUE
211 FORMAT(6F13.4)
C      WRITE(6,*) 'PWR=',PWR
C      END IF
C      GA is the state variable g.
C
      GA(1)=ZM1*(ZC1-ZV1)*PWR-CA1*ZC1*ABS(((ZV1-ZC2)/ZC1))*CRI
C      IF (IPR.EQ.1) THEN
C      WRITE(6,*) 'ZM1=',ZM1,' ZC1=',ZC1,' ZV1=',ZV1
C      WRITE(6,*) 'ZM2=',ZM2,' ZC3=',ZC3
C      END IF
C      WRITE(6,*) 'GA(1)=Zi: ',GA(1)
C      DO 240 I=1,6
C      GA(I+1)=ZM2*(ZC3*SIGVC(I)/SJ2**0.5-BETAA(I))*PWR+FAC9*BETAA(I)/ET
240 CONTINUE

C      VCTL(7..13) is the difference between the derivative of the
C
DO 280 I=1,7
  VEC1(I+12)=TDELTA*(GA(I)+SECTM(I))
  IF (IPR.EQ.1) WRITE(6,*) I,' SEC=',SECTM(I),' GA=',GA(I)
280 CONTINUE

C
DO 300 I=1,6
  VEC1(I)=0.0
300 CONTINUE

C
CALL MNU(19,6,ZZR)
DO 180 I=1,6
  IF (ABS(BETAA(I)).GT.(ZC3-1.0)) THEN
    DO 190 J=1,190
      ZZZ(I+13,J)=0.0
190 CONTINUE
      ZZZ(I+13,I+13)=1.0
      VEC1(I+13)=0.0
    END IF
180 CONTINUE

C
IF (BETAA(7).GT.(ZC1-1.0).OR.BETAA(7).LT.(2.0*ZC0-ZC1+1.0)) THEN
  DO 191 I=1,19
    ZZZ(13,I)=0.0
191 CONTINUE
    ZZZ(13,13)=1.0
    VEC1(13)=0.0
  END IF

```

```

C
DO 370 I=1,6
  DO 370 J=1,6
    ZZR(I,J)=-EM2(I,J)
370 CONTINUE
C
C   ZZR=-D*
C
  IJOB=3
  IBOD=19
  DD1=1.0
C
C
  CALL LINRG(1BOD,ZZZ,1BOD,ZZZ,1BOD)
  DO 978 I=1,1BOD
    VECC(I)=0.0
  DO 978 J=1,1BOD
    VECC(I)=ZZZ(I,J)*VECC(J)+VECC(I)
978 CONTINUE
  DO 972 I=1,1BOD
    VEC1(I)=VECC(I)
972 CONTINUE
C
C   For cyber:
C   CALL LINV3F(ZZZ,VEC1,IJOB,1BOD,1BOD,DD1,DD2,AINV,IER)
C
  DETMNT=DD1*(2**DD2)
  WRITE(6,*) 'The determinant of bodner matrix is: ',DETMNT
C
  IF (IER.EQ.130) THEN
    WRITE(6,*) 'INVERSE PROBLEM IN BODNER MATRIX, STOP.'
    STOP
  END IF
C
  CALL MMT(19,19,6,ZZZ,ZZR,T3D)
  IF (IPR.EQ.1) THEN
    write(6,*) 'element=',iaa
    write(6,*) 'em2:'
    DO 940 I=1,6
      WRITE(6,970) (EM2(I,J),J=1,6)
C 940 CONTINUE
    END IF
C
  DO 360 I=1,6
    DO 360 J=1,6
      EM2(I,J)=-T3D(I,J)
      EM4(IAA,IA,IB,IC,I*6-6+J)=EM2(I,J)
360 CONTINUE
  IF (IPR.EQ.1) THEN
    write(6,*) 'TDELT=',tdelt
    DO 980 I=1,6
      WRITE(6,970) (EM2(I,J),J=1,6)
      WRITE(6,970) (-T3D(I,J),J=1,6)
C 980 CONTINUE
    END IF
  970 FORMAT(6F12.1)
C
  DO 380 I=1,6
    BDL(I)=-VEC1(I)
    BDSV(IAA,IA,IB,IC,I)=BDL(I)
C 380 CONTINUE
    IF (IPR.EQ.1) WRITE(6,*) 'BDL(I):=',BDL(I)
C
  EM2 and BDL will be back to subroutine cb for assemble.
C
  DO 400 I=1,19

```

```

          SVBLD(IAA,IA,IB,IC,I)=VEC1(I)
400 CONTINUE
C
C   WRITE(6,*) 'T3D IN BODNER'
      DO 420 I=1,19
          DO 422 J=1,6
              SVT3D(IAA,IA,IB,IC,I*6-6+J)=T3D(I,J)
422 CONTINUE
C   WRITE(6,423) (T3D(I,K),K=1,6)
420 CONTINUE
C
      RETURN
      END
C
C   END BODNER
C
C*****
C   Subroutine walsul is the solution phase using Walker's constitutive
C   equation.
C   Input:
C   BL- used to find the local strain.
C   VFE- the displace increament. epsln=bl.vfe
C   SVT3D and SVBLD are the data calculated in the processing face.
C   State variable BETA(..12) are updated.
C   The derivative of the statevariable STVDF and the derivative of the
C   nonlinear strain EPSND are calculated.
C*****
C
C   SUBROUTINE WALSUL(IAA,IA,IB,IC,BL,VFE,SVT3D,SVBLD,BETA,SD,
1          BDSV,EM4,AA)
C
C   IMPLICIT REAL*8(A-H,O-Z)
C   IMPLICIT INTEGER*8(I-N)
C   DIMENSION BL(6,40),VFE(1),SVT3D(NELM,2,2,2,144),TMVEC(24),
1          SVBLD(NELM,2,2,2,24),BETA(NELM,2,2,2,12),SD(6,1),
2          BDSV(NELM,2,2,2,6),EM4(NELM,2,2,2,36),
3          DBTA1(6),DBTA2(6),AA(6,1)
C
C   COMMON /SCHALR1/ NELM,NNODE,NT
C   COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,
1          NSHOW3,HRZ,ITRLM,FACTOR
C   COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
C   COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1          IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2          IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3          IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4          IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5          IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
C   COMMON /RLVEC/ VR(1)
C   COMMON /INTVEC/ IPT(1)
C   COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
C   COMMON /GEO/ TO
C   COMMON /CNTRL/ DETMNT
C   COMMON /CONTN/ INSIDT,KPDT,DTLM1
C   COMMON /WAL/ WK,WB,WN2,WN3,WN4,WN5,WN6,WN8,WN9,WN10,WN11,WRO
C   COMMON /UNICT/ NCONS,MODEL,ETAA,TDELTA,TINIT
C   COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
C   COMMON /WKLMT/ WAL1,WAL2
C
C   IPR=0
C   IF((IA.EQ.1).AND.(IB.EQ.1).AND.(IC.EQ.1)) IPR=1
C   WRITE(6,*) 'IAA= ',IAA,' IA..IC ',IA,IB,IC
C   WRITE(6,*) 'WHERE CHANGED IN BODSUL'
C   DO 52 I=1,19
C       WRITE(6,53) (SVT3D(I,1,1,1,I*6-6+J),J=1,6)

```

```

52 CONTINUE
53 FORMAT (6F12.4)
59 FORMAT (6F12.4)
DO 60 I=1,24
    TMVEC (I)=0.0
    DO 80 J=1,6
        TMVEC (I)=TMVEC (I)-SVT3D (IAA, IA, IB, IC, I*6-6+J)*AA (J, I)
80 CONTINUE
    TMVEC (I)=SVBLD (IAA, IA, IB, IC, I)+TMVEC (I)
    IF (IPR.EQ.1) THEN
        WRITE (6,*) I, ' TMVEC (I) IN SOLFACE: ', TMVEC (I)
    END IF
90 CONTINUE

DO 100 I=1,6
    SD (I, I)=TMVEC (I)
    DBTA1 (I)=TMVEC (I+12)
    DBTA2 (I)=TMVEC (I+18)
    WRITE (6,*) I, ' D (Zd/DT): ', STVDF (IAA, IA, IB, IC, I)
100 CONTINUE

DO 120 I=1,6
    BETA (IAA, IA, IB, IC, I)=BETA (IAA, IA, IB, IC, I)+DBTA1 (I)
    BETA (IAA, IA, IB, IC, I+6)=BETA (IAA, IA, IB, IC, I+6)+DBTA2 (I)
    IF (BETA (IAA, IA, IB, IC, I).GT.WAL1) BETA (IAA, IA, IB, IC, I)=WAL1
    IF (BETA (IAA, IA, IB, IC, I).LT.-WAL1) BETA (IAA, IA, IB, IC, I)=-WAL1
    IF (BETA (IAA, IA, IB, IC, I+6).GT.WAL2) BETA (IAA, IA, IB, IC, I+6)=WAL2
    IF (BETA (IAA, IA, IB, IC, I+6).LT.-WAL2) BETA (IAA, IA, IB, IC, I+6)=-WAL2
    if (ipr.eq.1) then
        write (6,*) i, ' dta1=', dbta1 (i), ' dbta2=', dbta2 (i)
        WRITE (6,*) I, ' BA1: ', BETA (IAA, IA, IB, IC, I),
        1 ' ba2=', beta (iaa, ia, ib, ic, i+6)
    end if
120 CONTINUE

RETURN
END
END (WALSOL)

C *****
C * Subroutine WALKER is to prepare the stiffness matrix and the *
C * residue force. Input is the state variable and current stress. *
C * Output is EM2 (to form stiffness matrix by cb), BDL D *
C * (to form the force term by cb), SVT3D and SVBLD (will be used *
C * in the solution face) *
C *****
C
SUBROUTINE WALKER (III, IAA, IA, IB, IC, SIG, ZZZ, EM2, S, BETA, BDL D,
1 SVT3D, SVBLD, ZZR, BDSV, EM4, AINV)
C
IMPLICIT REAL*8 (A-H, O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION SIG (3, 3), ZZZ (24, 24), EM2 (6, 6), S (3, 3), BETA (NELM, 2, 2, 2, 12),
1 BDL D (1), SVT3D (NELM, 2, 2, 2, 144), SVBLD (NELM, 2, 2, 2, 24),
2 ZZR (24, 6), VEC1 (24), VCTL (24), GA (24), BETAA (6), AINV (1),
3 VEPS (6), SS (6), SECTM (12), T3D (24, 6), VEPSLN (3, 3),
4 BDSV (NELM, 2, 2, 2, 6), EM4 (NELM, 2, 2, 2, 36), SIGVC (6)
5 , AAA (6, 6), BBB (6, 6), CCC (6, 6), DDD (6, 6), BTA1 (6), BTA2 (6)
6 , DEFW (6), SGNW (6), VECC (24)
C
COMMON /WAL/ WK, WB, WN2, WN3, WN4, WN5, WN6, WN8, WN9, WN10, WN11, WRO
COMMON /UNICT/ NCONS, MODEL, ETAA, TDEL T, TINIT
COMMON /UNIFBD/ IR51, IR52, IR53, IR54, IR55, IR56, IR57, IR58, IR59
COMMON /SCHALR1/ NELM, NNODE, NT
COMMON /SCHALR2/ NEQT, NSTEP, NHBW, COEF1, COEF2, NSHOW1, NSHOW2,

```

```

1          NSHOW3,HRZ,ITRLM,FACTOR
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1          IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2          IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3          IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4          IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5          IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /RLVEC/ VR(1)
COMMON /INTVEC/ IPT(1)
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
COMMON /GEO/ TO
COMMON /CNTRL/ DETMNT
COMMON /CONTN/ INSIDT,KPDT,DTLMI
COMMON /ABDFST/ ISEC
COMMON /WKLMT/ WAL1,WAL2
COMMON /NCTT/ NCT(12,2,2,2)

```

ZNO,DO are input constants in kinematical equation.

ACS,ZC1,ZC2,ZC3,CM1,CM2 are constants in state variable equations.  
CR1,CR2 AS WELL.

S(i,j) is the stress deviator

$DJ2=1/2*S(I,J)*S(I,J)$

$SJ2=SIG(I,J)*SIG(I,J)$

ZV1=Zi

SIGM(6)---SIG(3,3) .....

$VSTV=D(Z)/DT$

$VSTV1=D(ZV1)/DT$

ET=-ETA\*TDELTA where eta and deltat are given.

IPR=0

IF((IA.EQ.1).AND.(IB.EQ.1).AND.(IC.EQ.1)) IPR=1

DO 20 I=1,6

BTA1(I)=BETA(IAA,IA,IB,IC,I)

BTA2(I)=BETA(IAA,IA,IB,IC,I+6)

20 CONTINUE

DO 30 I=1,6

BETAA(I)=BTA1(I)+BTA2(I)

30 CONTINUE

ET=-ETAA\*TDELTA

SAV=(SIG(1,1)+SIG(2,2)+SIG(3,3))/3.0

DO 90 I=1,3

DO 90 J=1,3

IF(I.EQ.J) THEN

S(I,J)=SIG(I,J)-SAV

ELSE

S(I,J)=SIG(I,J)

END IF

90 CONTINUE

SS(1)=S(1,1)

SS(2)=S(2,2)

SS(3)=S(3,3)

SS(4)=S(1,2)

SS(5)=S(2,3)

SS(6)=S(1,3)

DO 60 I=1,6

DEFW(I)=1.5\*SS(I)-BETAA(I)

```

      IF (DEFW(1).GE.0.0) THEN
        SGNW(1)=1.0
      ELSE
        SGNW(1)=-1.0
      END IF
C     if (ipr.eq.1) then
C     write(6,*) i,' bta1=',bta1(i),' bta2=',bta2(i),' def=',defw(i)
C     end if
60  CONTINUE
C
      WJ2=0.0
      SJ2=0.0
C
      DO 80 I=1,6
        IF (I.LE.3) THEN
          WJ2=WJ2+DEFW(I)*DEFW(I)
        ELSE
          WJ2=WJ2+2.0*DEFW(I)*DEFW(I)
        END IF
80  CONTINUE
C
      COW1=(2.0/3.0)**0.5
      WJSQ=WJ2**0.5
      RTW=COW1*WJSQ
      WJSE=(EXP(RTW/WK)-1.0)/WB
C
      write(6,*) 'iii in walk',iii
      ISEE=NCT(IAA,IA,IB,IC)
      DO 40 I=1,6
        VEPS(I)=DEFW(I)*WJSE/RTW
C     EPSND(IAA,IA,IB,IC,I)=VEPS(I)
40  CONTINUE
      NCT(IAA,IA,IB,IC)=1
C     if ((ia.eq.1).and.(ib.eq.1).and.(ic.eq.1)) then
C     write(6,*) (veps(i),i=1,6)
C     end if
C
      FAC1=3.0*ET*WJSE/RTW/2.0
      FAC2=(EXP(RTW/WK)/WB/WK/RTW/RTW-WJSE/RTW**3)*ET
      BTTN=(BETAA(1)+BETAA(2)+BETAA(3))/3.0
C
      if (ipr.eq.1) then
C     write(6,*) 'sig=',sig(2,2),' defw=',defw(2)
C     write(6,*) 'rtw=',rtw,' wjse=',wjse,' j2=',wj2
C     write(6,*) 'fac1=',fac1,' fac2=',fac2
C     write(6,*) 'defw(1)=' ,defw(1),' st=' ,bttn
C     end if
      Now -eta*deltat is included in the formula in first 6*6 matrix.
C
      CALL MNU(24,24,ZZZ)
C
      BTTN=0.0
      ZZZ(7,1)=FAC1*2.0/3.0+FAC2*DEFW(1)*(DEFW(1)+BTTN)
      ZZZ(7,2)=-FAC1/3.0+FAC2*DEFW(1)*(DEFW(2)+BTTN)
      ZZZ(7,3)=-FAC1/3.0+FAC2*DEFW(1)*(DEFW(3)+BTTN)
      ZZZ(7,4)=FAC2*DEFW(1)*DEFW(4)
      ZZZ(7,5)=FAC2*DEFW(1)*DEFW(5)
      ZZZ(7,6)=FAC2*DEFW(1)*DEFW(6)
C
      ZZZ(8,1)=-FAC1/3.0+FAC2*DEFW(2)*(DEFW(1)+BTTN)
      ZZZ(8,2)=FAC1*2.0/3.0+FAC2*DEFW(2)*(DEFW(2)+BTTN)
      ZZZ(8,3)=-FAC1/3.0+FAC2*DEFW(2)*(DEFW(3)+BTTN)
      ZZZ(8,4)=FAC2*DEFW(2)*DEFW(4)
      ZZZ(8,5)=FAC2*DEFW(2)*DEFW(5)
      ZZZ(8,6)=FAC2*DEFW(2)*DEFW(6)
C

```

ZZZ (9, 1) = -FAC1/3.0 + FAC2\*DEFW (3) \* (DEFW (1) + BTTN)  
ZZZ (9, 2) = -FAC1/3.0 + FAC2\*DEFW (3) \* (DEFW (2) + BTTN)  
ZZZ (9, 3) = FAC1\*2.0/3.0 + FAC2\*DEFW (3) \* (DEFW (3) + BTTN)  
ZZZ (9, 4) = FAC2\*DEFW (3) \* DEFW (4)  
ZZZ (9, 5) = FAC2\*DEFW (3) \* DEFW (5)  
ZZZ (9, 6) = FAC2\*DEFW (3) \* DEFW (6)

ZZZ (10, 1) = FAC2\*DEFW (4) \* (DEFW (1) + BTTN)  
ZZZ (10, 2) = FAC2\*DEFW (4) \* (DEFW (2) + BTTN)  
ZZZ (10, 3) = FAC2\*DEFW (4) \* (DEFW (3) + BTTN)  
ZZZ (10, 4) = FAC2\*DEFW (4) \* DEFW (4) + FAC1  
ZZZ (10, 5) = FAC2\*DEFW (4) \* DEFW (5)  
ZZZ (10, 6) = FAC2\*DEFW (4) \* DEFW (6)

ZZZ (11, 1) = FAC2\*DEFW (5) \* (DEFW (1) + BTTN)  
ZZZ (11, 2) = FAC2\*DEFW (5) \* (DEFW (2) + BTTN)  
ZZZ (11, 3) = FAC2\*DEFW (5) \* (DEFW (3) + BTTN)  
ZZZ (11, 4) = FAC2\*DEFW (5) \* DEFW (4)  
ZZZ (11, 5) = FAC2\*DEFW (5) \* DEFW (5) + FAC1  
ZZZ (11, 6) = FAC2\*DEFW (5) \* DEFW (6)

ZZZ (12, 1) = FAC2\*DEFW (6) \* (DEFW (1) + BTTN)  
ZZZ (12, 2) = FAC2\*DEFW (6) \* (DEFW (2) + BTTN)  
ZZZ (12, 3) = FAC2\*DEFW (6) \* (DEFW (3) + BTTN)  
ZZZ (12, 4) = FAC2\*DEFW (6) \* DEFW (4)  
ZZZ (12, 5) = FAC2\*DEFW (6) \* DEFW (5)  
ZZZ (12, 6) = FAC2\*DEFW (6) \* DEFW (6) + FAC1

C  
C

ZZZ (7, 7) = 1.0  
ZZZ (8, 8) = 1.0  
ZZZ (9, 9) = 1.0  
ZZZ (10, 10) = 1.0  
ZZZ (11, 11) = 1.0  
ZZZ (12, 12) = 1.0

C

FAC3 = -FAC1/3.0\*2.0  
FAC4 = -FAC2\*2.0/3.0

C

ZZZ (7, 13) = FAC4\*DEFW (1) \* DEFW (1) + FAC3  
ZZZ (8, 13) = FAC4\*DEFW (2) \* DEFW (1)  
ZZZ (9, 13) = FAC4\*DEFW (3) \* DEFW (1)  
ZZZ (10, 13) = FAC4\*DEFW (4) \* DEFW (1)  
ZZZ (11, 13) = FAC4\*DEFW (5) \* DEFW (1)  
ZZZ (12, 13) = FAC4\*DEFW (6) \* DEFW (1)

C

ZZZ (7, 14) = FAC4\*DEFW (1) \* DEFW (2)  
ZZZ (8, 14) = FAC4\*DEFW (2) \* DEFW (2) + FAC3  
ZZZ (9, 14) = FAC4\*DEFW (3) \* DEFW (2)  
ZZZ (10, 14) = FAC4\*DEFW (4) \* DEFW (2)  
ZZZ (11, 14) = FAC4\*DEFW (5) \* DEFW (2)  
ZZZ (12, 14) = FAC4\*DEFW (6) \* DEFW (2)

C

ZZZ (7, 15) = FAC4\*DEFW (1) \* DEFW (3)  
ZZZ (8, 15) = FAC4\*DEFW (2) \* DEFW (3)  
ZZZ (9, 15) = FAC4\*DEFW (3) \* DEFW (3) + FAC3  
ZZZ (10, 15) = FAC4\*DEFW (4) \* DEFW (3)  
ZZZ (11, 15) = FAC4\*DEFW (5) \* DEFW (3)  
ZZZ (12, 15) = FAC4\*DEFW (6) \* DEFW (3)

C

ZZZ (7, 16) = FAC4\*DEFW (1) \* DEFW (4)  
ZZZ (8, 16) = FAC4\*DEFW (2) \* DEFW (4)  
ZZZ (9, 16) = FAC4\*DEFW (3) \* DEFW (4)  
ZZZ (10, 16) = FAC4\*DEFW (4) \* DEFW (4) + FAC3  
ZZZ (11, 16) = FAC4\*DEFW (5) \* DEFW (4)  
ZZZ (12, 16) = FAC4\*DEFW (6) \* DEFW (4)



```

ZZZ (7, 17)=FAC4*DEFW (1) *DEFW (5)
ZZZ (8, 17)=FAC4*DEFW (2) *DEFW (5)
ZZZ (9, 17)=FAC4*DEFW (3) *DEFW (5)
ZZZ (10, 17)=FAC4*DEFW (4) *DEFW (5)
ZZZ (11, 17)=FAC4*DEFW (5) *DEFW (5)+FAC3
ZZZ (12, 17)=FAC4*DEFW (6) *DEFW (5)

ZZZ (7, 18)=FAC4*DEFW (1) *DEFW (6)
ZZZ (8, 18)=FAC4*DEFW (2) *DEFW (6)
ZZZ (9, 18)=FAC4*DEFW (3) *DEFW (6)
ZZZ (10, 18)=FAC4*DEFW (4) *DEFW (6)
ZZZ (11, 18)=FAC4*DEFW (5) *DEFW (6)
ZZZ (12, 18)=FAC4*DEFW (6) *DEFW (6)+FAC4

DO 120 I=7,12
  DO 120 J=1,6
    ZZZ (I, J+18)=ZZZ (I, J+12)
120 CONTINUE

C
C Next part is -[G, epsilon n]
C
PWR=0.0
DO 145 I=1,6
  IF (I.LE.3) THEN
    PWR=PWR+VEPS (I) *VEPS (I)
  ELSE
    PWR=PWR+2.0*VEPS (I) *VEPS (I)
  END IF
145 CONTINUE

C
PWR=(2.0*PWR/3.0)**0.5
C
C WRITE (6,*) 'PLASTIC WORK IS: ',PWR
C
IF (PWR.GT.WRO) THEN
  FAC5=(WRO/PWR)**WN5
  FAC7=-2.0*WN5*(WRO**WN5)*(PWR**(-WN5-1.0))*WN4/3.0
ELSE
  FAC5=(PWR/WRO)**WN5
  FAC7=2.0*(PWR** (WN5-1.0)) / (WRO**WN5) /3.0*WN4
END IF

C
FAC6=2.0*(WN3+WN4*FAC5)/3.0/PWR+FAC7
C
C WRITE (6,*) 'FAC6: ',FAC6
C
FAC8=2.0*WN9/3.0/PWR
DO 150 I=1,6
  DO 150 J=1,6
    IF (I.EQ.J) THEN
      ZZZ (12+I, 6+J)=FAC6*BTA1 (I) *VEPS (J) -WN2
      ZZZ (18+I, 6+J)=FAC8*BTA2 (I) *VEPS (J) -WN11
    ELSE
      ZZZ (12+I, 6+J)=FAC6*BTA1 (I) *VEPS (J)
      ZZZ (18+I, 6+J)=FAC8*BTA2 (I) *VEPS (J)
    END IF
150 CONTINUE

C
C Next part: dg/dx
C
FAC9=-((WN4*FAC5+WN3)*PWR+WN6)*ET
FAC10=- (WN9*PWR+WN10)*ET
C
DO 160 I=1,6
  DO 160 J=1,6

```

```

        IF (I.EQ.J) THEN
            ZZZ (12+I, 12+J) = 1.0+FAC9
            ZZZ (18+I, 18+J) = 1.0+FAC10
            ZZZ (I, J) = 1.0
        END IF
160 CONTINUE
C
    DO 333 I=1,6
        DO 333 J=1,6
            ZZZ (I, J+6) = EM2 (I, J)
333 CONTINUE
C
C    Now matrix [zzz] is formed.
C    Next step is to find the vector part.
C
    SIGVC (1) = SIG (1, 1)
    SIGVC (2) = SIG (2, 2)
    SIGVC (3) = SIG (3, 3)
    SIGVC (4) = SIG (1, 2)
    SIGVC (5) = SIG (2, 3)
    SIGVC (6) = SIG (1, 3)
C
C    VCTL(1..6) is the difference of d(epsilon)/dt and f.
C
    DO 200 I=1,6
        VEC1 (I+6) = TDELTA*VEPS (I)
200 CONTINUE
C
C    SECTM(i) is -(G, epsilon*d(epsilon)/dt)
C
    DO 220 I=1,12
        SECTM (I) = 0.0
        DO 220 J=1,6
            ZZZ (I+12, J+6) = 0.0
220 CONTINUE
C
C    GA is the state variable g
C
    FAC12 = PWR*(WN3+WN4*FAC5) + WN6
    FAC13 = WN9*PWR + WN10
    DO 240 I=1,6
        GA (I) = WN2*VEPS (I) - BTA1 (I) * FAC12
        GA (I+6) = WN11*VEPS (I) - BTA2 (I) * FAC13
C    WRITE (6, *) 'GA 2..7=Zd: ', GA (I+1)
240 CONTINUE
C
C
    DO 280 I=1,12
        VEC1 (I+12) = TDELTA*GA (I)
280 CONTINUE
C
    DO 300 I=1,6
        VEC1 (I) = 0.0
        IF (ABS (BTA1 (I)) .GT. (WAL1-1.0)) VEC1 (I+12) = 0.0
300 CONTINUE
C
    CALL MNU (24, 6, ZZR)
C
    DO 370 I=1,6
        DO 370 J=1,6
            ZZR (I, J) = -EM2 (I, J)
370 CONTINUE
C
C    ZZR = -D*
C
    IJOB = 3

```

```

      IBOD=24
      DD1=1.0
C     DO 310 I=1,24
C       WRITE (6,*) 'I= ',I,' VEC1 (I): ',VEC1 (I)
C310  CONTINUE
      DO 320 I=1,19
      WRITE (6,330) (ZZZ (I,J),J=1,12)
C320  CONTINUE
C     DO 340 I=1,19
C       WRITE (6,350) (ZZZ (I,J),J=13,19)
C340  CONTINUE
C330  FORMAT (12F6.1)
C350  FORMAT (7F9.2)
C
C     For cyber:
C     CALL LINV3F (ZZZ,VEC1,IJOB,IBOD,IBOD,DD1,DD2,AINV,IER)
C     CALL LINRG (IBOD,ZZZ,IBOD,ZZZ,IBOD)
      DO 978 I=1,IBOD
      VECC (I)=0.0
      DO 978 J=1,IBOD
      VECC (I)=ZZZ (I,J)*VEC1 (J)+VECC (I)
978   CONTINUE
      DO 972 I=1,IBOD
      VEC1 (I)=VECC (I)
972   CONTINUE
C
C     DETMNT=DD1*(2**DD2)
C
C     WRITE (6,*) 'The determinant of bodner matrix is: ',DETMNT
C
C     IF (IER.EQ.130) THEN
      WRITE (6,*) 'INVERSE PROBLEM IN BODNER MATRIX, STOP.'
      STOP
    END IF
C
C     CALL MMT (24,24,6,ZZZ,ZZR,T3D)
C     IF (IPR.EQ.1) THEN
C     DO 940 I=1,6
C       WRITE (6,970) (EM2 (I,J),J=1,6)
C940  CONTINUE
C     END IF
C
      DO 360 I=1,6
      DO 360 J=1,6
      EM2 (I,J)=-T3D (I,J)
      EM4 (IAA,IA,IB,IC,I*6-6+J)=EM2 (I,J)
360  CONTINUE
C     IF (IPR.EQ.1) THEN
C     DO 980 I=1,6
C       WRITE (6,970) (EM2 (I,J),J=1,6)
C       WRITE (6,970) (-T3D (I,J),J=1,6)
C980  CONTINUE
C     END IF
C970  FORMAT (6F12.1)
C
      DO 380 I=1,6
      BDL (I)=-VEC1 (I)
      BDSV (IAA,IA,IB,IC,I)=VEC1 (I)
C     WRITE (6,*) 'BDL (I) :=-ZITA ',BDL (I)
380  CONTINUE
C
C     EM2 and BDL will be back to subroutine cb for assemble.
C
      DO 400 I=1,24
      SVBL (IAA,IA,IB,IC,I)=VEC1 (I)
400  CONTINUE

```

```

C
C WRITE (6,*) 'T3D IN BODNER'
C DO 420 I=1,24
C   DO 422 J=1,6
C     SVT3D(IAA,IA,IB,IC,I*6-6+J)=T3D(I,J)
422 CONTINUE
420 CONTINUE
C
C SVT3D and SVBLD will be used in processing face.
C
C RETURN
C END
C
C (END WALKER)
C
C Subroutine is used to calculate the material constants of
C Bodner-Partom type of constitutive equations. The material
C used is B1900+Hf. For different material, this subroutine
C should be modified.
C
C SUBROUTINE BDCNS(TMPP)
C IMPLICIT REAL*8(A-H,O-Z)
C IMPLICIT INTEGER*8(I-N)
C COMMON /BOD/ DO,ZC0,ZC1,ZC2,ZC3,ZM1,ZM2,CA1,CA2,CR1,CR2,ZNO
C COMMON /MTL/ E,EU
C
C E=198700.0+16.78*TMPP-0.1034*TMPP*TMPP
1 +0.00001143*TMPP*TMPP*TMPP
C WRITE (6,*) 'BODNER CONST: E=',E
C EU=0.3
C DO=10000.0
C ZC0=2700.0
C ZC1=3000.0
C ZC2=2700.0
C ZC3=1150.0
C ZM1=0.27
C ZM2=1.52
C CA1=0.0
C CA2=0.0
C CR1=2.0
C CR2=2.0
C ZNO=1.055
C IF (TMPP.LT.760.0) THEN
C   ZC0=2700.0
C   CA1=0.0
C   ZNO=1.055
C END IF
C IF ((TMPP.GE.760.0).AND.(TMPP.LT.871.0)) THEN
C   ZC0=2700.0-(TMPP-760.0)/111.0*300.0
C   CA1=(TMPP-760.0)/111.0*0.0055
C   ZNO=1.055-(TMPP-760.0)/111.0*0.025
C END IF
C IF ((TMPP.GE.871.0).AND.(TMPP.LT.982.0)) THEN
C   ZC0=2400.0-(TMPP-871.0)/111.0*500.0
C   CA1=(TMPP-871.0)/111.0*0.0145+0.0055
C   ZNO=1.03-(TMPP-871.0)/111.0*0.18
C END IF
C IF ((TMPP.GE.982.0).AND.(TMPP.LT.1093.0)) THEN
C   ZC0=1900.0-(TMPP-982.0)/111.0*700.0
C   CA1=(TMPP-982.0)/111.0*0.23+0.02
C   ZNO=0.85-(TMPP-982.0)/111.0*0.15
C END IF
C CA2=CA1
C ZC2=ZC0
C
C WRITE (6,*) 'ELASTIC MODULUS=',E

```

```
WRITE (6,*) 'ELASTIC MODULUS=',E,' ZO=',ZCO,' A=',CA1,' N=',ZNO
```

```
RETURN  
END
```

```
Subroutine is used to calculate the material constants of  
Walker type of constitutive equations. The material  
used is B1900+Hf. For different material, this subroutine  
should be modified.
```

```
SUBROUTINE WKCNS (TMPP)  
IMPLICIT REAL*8 (A-H,O-Z)  
IMPLICIT INTEGER*8 (I-N)  
COMMON /WAL/ WK,WB,WN2,WN3,WN4,WN5,WN6,WN8,WN9,WN10,WN11,WRO  
COMMON /MTL/ E,EU  
COMMON /WKLMT/ WAL1,WAL2
```

```
TEM=TMPP  
WK=12.4  
WB=1.73E11  
WN2=2.41E6  
WN3=4794.0  
WN4=0.0  
WN5=0.3117  
WN6=0.0  
WN7=0.0  
WN8=0.0  
WN9=11.87  
WN10=0.0  
WN11=4.7E3  
WRO=1.0E-4  
E=1.9E5  
IF ((TEM.GT.-0.01).AND.(TEM.LT.427.0)) THEN  
STE=TEM/427.0  
EU=0.322+(0.328-0.322)*STE  
END IF  
IF ((TEM.GE.427.0).AND.(TEM.LT.538.0)) THEN  
STE=(TEM-427.0)/(538.0-427.0)  
EU=0.328+(0.331-0.328)*STE  
END IF  
IF ((TEM.GE.538.0).AND.(TEM.LT.649.0)) THEN  
STE=(TEM-538.0)/(649.0-538.0)  
E=1.9E5+(1.8E5-1.9E5)*STE  
EU=0.331+(0.334-0.331)*STE  
WB=1.73E11+(3.862E10-1.73E11)*STE  
WN2=2.41E6+(8.27E5-2.41E6)*STE  
WN3=4794.0+(1714.0-4794.0)*STE  
WN9=11.87+(16.64-11.87)*STE  
END IF
```

```
IF ((TEM.GE.649.0).AND.(TEM.LT.760.0)) THEN  
STE=(TEM-649.0)/(760.0-649.0)  
E=1.8E5+(1.655E5-1.8E5)*STE  
EU=0.334+(0.339-0.334)*STE  
WK=12.4+(13.8-12.4)*STE  
WB=3.862E10+(2.55E10-3.862E10)*STE  
WN2=8.27E5  
WN3=1714.0+(1880.0-1714.0)*STE  
WN4=-585.0*STE  
WN9=16.64+(19.83-16.64)*STE  
WN10=2.44E-3*STE  
END IF
```

```
IF ((TEM.GE.760.0).AND.(TEM.LT.871.0)) THEN
```

```

STE=(TEM-760.0)/(871.0-760.0)
E=1.655E5+(1.438E5-1.655E5)*STE
EU=0.339+(0.324-0.339)*STE
WK=13.8+(16.6-13.8)*STE
WB=2.55E10+(5.5E12-2.55E10)*STE
WN2=8.27E5+(2.36E5-8.27E5)*STE
WN3=1880.0+(621.2-1880.0)*STE
WN4=-585.0+585.0*STE
WN6=8.73E-4*STE
WN9=19.83+(59.33-19.83)*STE
WN10=2.44E-3
WN11=4.70E3+(9.65E2-4.7E3)*STE
END IF

```

```

C
IF ((TEM.GE.871.0).AND.(TEM.LT.982.0)) THEN
STE=(TEM-871.0)/(982.0-871.0)
E=1.438E5+(1.249E5-1.438E5)*STE
EU=0.324+(0.351-0.324)*STE
WK=16.6+(13.8-16.6)*STE
WB=5.5E12+(4.2E10-5.5E12)*STE
WN2=2.36E5+(9.65E4-2.36E5)*STE
WN3=621.2+(400.0-621.2)*STE
WN4=0.0
WN6=8.73E-4+(4.29E-4-8.73E-4)*STE
WN9=59.33+(136.0-59.33)*STE
WN10=2.44E-3
WN11=9.65E2+(-9.65E2)*STE
END IF

```

```

C
IF ((TEM.GE.982.0).AND.(TEM.LE.1093.0)) THEN
STE=(TEM-982.0)/(1093.0-982.0)
E=1.249E5+(1.161E5-1.249E5)*STE
EU=0.351
WK=13.8+(9.0-13.8)*STE
WB=4.2E10+(5.57E9-4.2E10)*STE
WN2=9.65E4+(2.36E4-9.65E4)*STE
WN3=400.0+(278.7-400.0)*STE
WN4=0.0
WN6=4.29E-4+(4.83E-2-4.29E-4)*STE
WN9=136.0
WN10=2.44E-3
WN11=0.0
END IF

```

```

C
IF (TEM.GT.1093.0) THEN
WRITE (6,*) 'MATERIAL CONSTANTS ARE NOT AVAILABLE'
STOP
END IF
WAL1=WN2/WN3
WAL2=WN11/WN9

```

```

C
WRITE (6,*) 'WK=',WK
WRITE (6,*) 'WB=',WB
WRITE (6,*) 'WN2=',WN2
WRITE (6,*) 'WN3=',WN3
WRITE (6,*) 'WN4=',WN4
WRITE (6,*) 'WN5=',WN5
WRITE (6,*) 'WN6=',WN6
WRITE (6,*) 'WN8=',WN8
WRITE (6,*) 'WN9=',WN9
WRITE (6,*) 'WN10=',WN10
WRITE (6,*) 'WN11=',WN11
WRITE (6,*) 'WRO=',WRO
WRITE (6,*) 'WLMT1=',WAL1
WRITE (6,*) 'WLMT2=',WAL2

```

RETURN  
END

C  
C Subroutine THRML is for the calculation of thermal effects  
C of the structure. Newton-Raphson's iteration scheme is used  
C in the equilibrium iterations.

C  
C SUBROUTINE THRML (INUM, IEL, ID, IID, L, MAXA, LD, XX, YY, ZZ, DLOADT,  
1 D, PLD, FRCO, DD, DLDINC, VTEMP, VF, D1, VFE, DDD,  
2 AM, PD, P, A, TDLD, HISINC, ACMDIS, FRCINC, XX1, YY1,  
3 ZZ1, DELTA, UPSIG, SIGMA, DLTINC, DLTTMP, STIFFN,  
4 EXLVC, BETA, UPBET, ACTFRC, GCL1, GCL2, GCL3, UCL1,  
5 UCL2, UCL3, DD1)  
IMPLICIT REAL\*8 (A-H, O-Z)  
IMPLICIT INTEGER\*8 (I-N)

C  
C DIMENSION IEL (NELM, 5), ID (1), IID (NNODE, 5), L (1), MAXA (1)  
C DIMENSION XX (1), YY (1), ZZ (1), DD (NNODE, 5), D (1), PLD (1),  
1 DLOADT (1), DLDINC (1), VTEMP (1), VF (NNODE, 5),  
2 D1 (NNODE, 5), VFE (NT, 1), DDD (1), P (1), VRT (4),  
3 A (NEQT, NEQT), AM (40, 40), PD (1), TDLD (1),  
4 HISINC (1), ACMDIS (1), FRCINC (1), XX1 (1), YY1 (1), ZZ1 (1),  
5 DELTA (1), FRCO (1), UPSIG (NELM, 2, 2, 2, 9), ACTFRC (1),  
6 SIGMA (NELM, 2, 2, 2, 9), DLTINC (1), DLTTMP (1), COEEQ (5),  
7 DEFVRT (4), STIFFN (NT, NT), ETT (4), EXLVC (1), DD1 (1),  
8 BETA (NELM, 2, 2, 2, 12), UPBET (NELM, 2, 2, 2, 12), GCL1 (NNODE, 3),  
9 GCL2 (NNODE, 3), GCL3 (NNODE, 3), UCL1 (NNODE, 3),  
1 UCL2 (NNODE, 3), UCL3 (NNODE, 3)

C  
C COMMON /SCHALR1/ NELM, NNODE, NT  
C COMMON /SCHALR2/ NEQT, NSTEP, NHBW, COEF1, COEF2, NSHOW1, NSHOW2,  
1 NSHOW3, HRZ, ITRLM, FACTOR  
C COMMON /PNTRIN/ IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10  
C COMMON /PNTRRL/ IR1, IR2, IR3, IR4, IR5, IR6, IR7, IR8, IR9, IR10,  
1 IR11, IR12, IR13, IR14, IR15, IR16, IR17, IR18,  
2 IR19, IR20, IR21, IR22, IR23, IR24, IR25, IR26,  
3 IR27, IR28, IR29, IR30, IR31, IR32, IR33, IR34,  
4 IR35, IR36, IR37, IR38, IR39, IR40, IR41, IR42,  
5 IR43, IR44, IR45, IR46, IR47, IR48, IR49, IR50  
C COMMON /UNIFBD/ IR51, IR52, IR53, IR54, IR55, IR56, IR57, IR58, IR59  
C COMMON /DIRCS/ IR60, IR61, IR62, IR63, IR64, IR65  
C COMMON /DISVC/ IR66, IR67, IR68, IR69  
C COMMON /DISV1/ IR70, IR71, IR72, IR73, IR74, IR75  
C COMMON /UNICT/ NCONS, MODEL, ETAA, TDELTA, TINIT  
C COMMON /RLVEC/ VR (1)  
C COMMON /INTVEC/ IPT (1)  
C COMMON /ITESCH/ ROOT, DTLAM, SGN, IPP, TROOT, ASO, SP  
C COMMON /GEO/ TO  
C COMMON /CNTRL/ DETMNT  
C COMMON /CONTN/ INSIDT, KPDT, DTLM1  
C COMMON /ABDFST/ ISEC  
C COMMON /MTL/ E, EU  
C COMMON /NMBITR/ NUM  
C COMMON /CNTR/ ICNTR  
C COMMON /TMPCO/ ICTMP  
C COMMON /TMPEF/ IDO, NTEM, NITR, NANM, CEXP, TMMIN, TMINC, TMMAX, TMPP

C  
C ICTMP=1

C (The switch to the effects of the change of temperature is on)

C ND=NEQT

C ICNTR=ICNTR+1

C  
C Initiate some variables.

C  
C CALL INIT (VR (IR1), VR (IR2), VR (IR3), VR (IR4), VR (IR5),

```

1          VR (IR60) ,VR (IR61) ,VR (IR62) ,VR (IR63) ,VR (IR64) ,VR (IR65) ,
2          VR (IR47) ,VR (IR20) ,VR (IR51) ,VR (IR58))

Begin iteration

III=1

DO 195 I=1,ND
    TDLD (I)=0.0
195 CONTINUE

CALL MNU (NNODE,5,DD)

Form stiffness matrix.

CALL ASSMBL (III, IPT (IP1) ,IPT (IP2) ,IPT (IP3) ,IPT (IP4) ,IPT (IP5) ,
1          IPT (IP9) ,VR (IR1) ,VR (IR2) ,VR (IR3) ,VR (IR6) ,VR (IR8) ,
2          VR (IR12) ,VR (IR14) ,
3          VR (IR15) ,VR (IR16) ,VR (IR19) ,
4          VR (IR21) ,VR (IR23) ,VR (IR24) ,VR (IR19) ,VR (IR41) ,VR (IR50) ,
5          VR (IR52) ,VR (IR66) ,VR (IR67) ,VR (IR68) ,VR (IR74))

Calculate the equivalent load vector

CALL INLDV (IPT (IP1) ,VR (IR1) ,VR (IR2) ,VR (IR3) ,
1          VR (IR14) ,VR (IR22) ,VR (IR28) ,VR (IR4))

DO 200 I=1,NT
    DLDINC (I)=DD1 (I)
    WRITE (6,*) I, ' DD1 (I)=', DD1 (I)
200 CONTINUE

CALL REDC (IPT (IP4) ,VR (IR8) ,VR (IR12))

DO 570 I=1,ND
    DD1 (I)=0.0
    EXLVC (I)=D (I)
    WRITE (6,*) I, ' D (I)=', D (I)
570 CONTINUE
    WRITE (6,*) ITRLM
    WRITE (6,36) III
36  FORMAT ('THIS IS THE ITERATION ',I3)
571 CONTINUE

DO 444 I=1,ND
    TDLD (I)=0.0
    DO 444 J=1,ND
        TDLD (I)=TDLD (I)+A (I,J)*D (J)
444  CONTINUE

DO 505 I=1,NT
    DO 505 M=1,ND
        IF (I.EQ.L (M)) THEN
            DLOADT (I)=TDLD (M)
        END IF
505 CONTINUE

WRITE (6,*) 'Temperature related displacement:'
DO 506 I=1,NNODE
    DO 506 J=1,5
        VF (I,J)=DLOADT (I*5-5+J)
        DD (I,J)=DD (I,J)+VF (I,J)
    WRITE (6,*) 'I=',I, ' ',VF (I,1) ,' ',VF (I,2) ,' ',VF (I,3)
506 CONTINUE

```



```

C
C Estimate the new coordinates
C
TINC=1.0
IF (III.EQ.NANM) STOP
C
DO 900 I=1,NNODE
  XX(I)=XX(I)+VF(I,1)
  YY(I)=YY(I)+VF(I,2)
  ZZ(I)=ZZ(I)+VF(I,3)
  TMP=0.0
  DO 903 J=1,3
    GCL3(I,J)=GCL3(I,J)+TINC*(-GCL2(I,J)*VF(I,4)+GCL1(I,J)*VF(I,5))
    TMP=TMP+GCL3(I,J)*GCL3(I,J)
903  CONTINUE
    TMP=TMP**0.5
    DO 902 J=1,3
      GCL3(I,J)=GCL3(I,J)/TMP
902  CONTINUE
  C WRITE(6,*) 'I=',I,' ',VF(I,1),' ',VF(I,2),' ',VF(I,3)
  C WRITE(6,267) I,XX(I),YY(I),ZZ(I)
900 CONTINUE
  CALL CNND(VR(IR60),VR(IR61),VR(IR62))
C
C Calculate internal forces
C
CALL INTFRC(III,IPT(IP1),VR(IR1),VR(IR2),VR(IR3),
1 VR(IR14),VR(IR22),VR(IR28),VR(IR9))
C
C SHRINK THE INTERNAL FORCE VECTOR
C
DO 500 I=1,NT
  C WRITE(6,*) 'PLD ',I,' ',PLD(I)
  DO 500 M=1,ND
    IF (I.EQ.L(M)) THEN
      FRCINC(M)=PLD(I)-FRCO(M)
      ACTFRC(M)=PLD(I)
504  FORMAT('THE LOAD COL D,IS:',112,' ',2F12.5)
    END IF
500 CONTINUE
  DO 502 I=1,ND
  C WRITE(6,*) I,' RD PLD=',ACTFRC(I),' DD1=',DD1(I)
502 CONTINUE
C
C Check whether to step out the equilibrium iterations
C
CALL CRITR2(III,ND,VR(IR8),VR(IR42),VR(IR59),VLINIT,ICNC1)
C
IF (III.EQ.40) THEN
  WRITE(6,*) 'ITER LIMIT IN TEM. REACHED,STOP'
  STOP
END IF
IF (ICNC1.EQ.0) THEN
  ICTMP=0
C (The switch to the effects of the change of temperature is off)
  DO 700 I=1,ND
  C WRITE(6,*) I,' 3,D=',D(I),' FRCINC',FRCINC(I)
    D(I)=-FRCINC(I)
700 CONTINUE
    III=III+1
    GOTO 571
  END IF
C
701 CONTINUE
  ISEC=ISEC+1
  IF (ISEC.GT.10) ISEC=10

```

```

      K=1
      DO 589 I=1,NNODE
        DO 589 J=1,5
          IF (IID(I,J).EQ.0) THEN
            ACMDIS(K)=ACMDIS(K)+DD(I,J)
            DI(I,J)=ACMDIS(K)
            K=K+1
          END IF
589    CONTINUE
C
      DO 689 I=1,NNODE
        DO 689 J=1,5
          DD(I,J)=0.0
689    CONTINUE
C
      ITYPE=2
C
C      Update some of the variables if equilibrium iteration is succeeded.
C
      CALL UPDT(ITYPE,IPT(IP3),VR(IR1),VR(IR2),VR(IR3),VR(IR12),
1      VR(IR15),VR(IR27),VR(IR43),VR(IR44),VR(IR45),
2      VR(IR46),VR(IR47),VR(IR20),VR(IR48),VR(IR49),
3      VR(IR51),VR(IR58),VR(IR60),VR(IR61),VR(IR62),
4      VR(IR63),VR(IR64),VR(IR65),VR(IR75))
C
C      Data output
C
      CALL OUTPUT(TTLD,VR(IR15),VR(IR75),VR(IR71),VR(IR1),VR(IR2),
1      VR(IR3))
C
      IF (NITR.EQ.NUM) THEN
C
C      Write necessary data for further use.
C
      CALL WTCDT(VR(IR27),VR(IR20),VR(IR43),VR(IR44),
1      VR(IR45),VR(IR1),VR(IR2),VR(IR3),
1      VR(IR47),VR(IR10),VR(IR51),VR(IR58),VR(IR60),
3      VR(IR61),VR(IR62),VR(IR15),VR(IR71),VR(IR75))
      END IF
C
      RETURN
      END
C
C      Subroutine is used to calculate the equivalent load vector
C      caused by the change of temperature.
C
      SUBROUTINE INLDV(IEL,XX,YY,ZZ,
1      VF,PD,PDL,PLD)
      IMPLICIT REAL*8(A-H,O-Z)
      IMPLICIT INTEGER*8(I-N)
      DIMENSION XX(1),YY(1),ZZ(1),VF(NNODE,5),PD(1),PDL(1),PLD(1)
      DIMENSION H(2),P(2),R(8),S(8),X(8),Y(8),Z(8),ND(8),IEL(NELM,8),
1      VFE(40)
C
      COMMON /SCHALR1/ NELM,NNODE,NT
      COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
      COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1      IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2      IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3      IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4      IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5      IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
      COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
      COMMON /DIRCS/ IR60,IR61,IR62,IR63,IR64,IR65
      COMMON /UNICT/ NCONS,MODEL,ETAA,TDELT,TINIT
      COMMON /RLVEC/ VR(1)

```

```

COMMON /INTVEC/ IPT(1)
COMMON /A3/ CL1(8),CM1(8),CN1(8),CL2(8),CM2(8),CN2(8),
1      CL3(8),CM3(8),CN3(8)

```

```

DO 30 I=1,NT
  PLD(I)=0.0
30 CONTINUE

```

```

DO 700 I=1,NELM
  I1=IEL(I,1)
  I2=IEL(I,2)
  I3=IEL(I,3)
  I4=IEL(I,4)
  I5=IEL(I,5)
  I6=IEL(I,6)
  I7=IEL(I,7)
  I8=IEL(I,8)

```

```

CALL UPILD(I,I1,I2,I3,I4,I5,I6,I7,I8,VR(IR1),VR(IR2),VR(IR3),
1      VR(IR14),VR(IR22),VR(IR28),VR(IR60),VR(IR61),VR(IR62))

```

```

DO 700 J=1,8
  DO 700 K=1,5
    JJ=IEL(I,J)*5-5+K
    J1=J*5-5+K
    PLD(JJ)=PLD(JJ)+PD(J1)

```

```

700 CONTINUE

```

```

RETURN
END
(END INLDV)

```

Subroutine UPILD is used to evaluate the equivalent load vector caused by the change of temperature at every element.

```

SUBROUTINE UPILD(IL,I1,I2,I3,I4,I5,I6,I7,I8,XX,YY,ZZ,
1      VF,PD,PDL,GCL1,GCL2,GCL3)

```

```

IMPLICIT REAL*8(A-H,O-Z)
IMPLICIT INTEGER*8(I-N)
DIMENSION XX(1),YY(1),ZZ(1),VF(NNODE,5),PD(1),PDL(1),
1      H(2),P(2),R(8),S(8),X(8),Y(8),Z(8),ND(8),
2      VFE(40),GCL1(NNODE,3),GCL2(NNODE,3),GCL3(NNODE,3),
3      HH(4),PP(4)
COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1      IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2      IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3      IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4      IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5      IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
COMMON /DIRCS/ IR60,IR61,IR62,IR63,IR64,IR65
COMMON /UNICT/ NCONS,MODEL,ETAA,TDELTA,TINIT
COMMON /RLVEC/ VR(1)
COMMON /INTVEC/ IPT(1)
COMMON /CONTN/ INSIDT,KPDT,DTLM1
COMMON /A3/ CL1(8),CM1(8),CN1(8),CL2(8),CM2(8),CN2(8),
1      CL3(8),CM3(8),CN3(8)

```

```

ND(1)=11
ND(2)=12

```

```

ND (3)=13
ND (4)=14
ND (5)=15
ND (6)=16
ND (7)=17
ND (8)=18
C
C
DO 250 I=1,8
  X (I)=XX (ND (I))
  Y (I)=YY (ND (I))
  Z (I)=ZZ (ND (I))
C
  WRITE (6,260) I,X (I),Y (I),Z (I),ND (I)
  DO 250 J=1,5
    VFE (I*5-5+J)=VF (ND (I),J)
250 CONTINUE
260 FORMAT (1X,'THE COORDINATES OF NODE',12,1X,'ARE:',3F12.8,112)
C
C
R (1)=-1.0
S (1)=-1.0
R (2)=1.0
S (2)=-1.0
R (3)=1.0
S (3)=1.0
R (4)=-1.0
S (4)=1.0
C
R (5)=0.0
S (5)=-1.0
R (6)=1.0
S (6)=0.0
R (7)=0.0
S (7)=1.0
R (8)=-1.0
S (8)=0.0
C
C
WRITE (6,157) IL
C
DO 344 I=1,8
  CL1 (I)=GCL1 (ND (I),1)
  CM1 (I)=GCL1 (ND (I),2)
  CN1 (I)=GCL1 (ND (I),3)
  CL2 (I)=GCL2 (ND (I),1)
  CM2 (I)=GCL2 (ND (I),2)
  CN2 (I)=GCL2 (ND (I),3)
  CL3 (I)=GCL3 (ND (I),1)
  CM3 (I)=GCL3 (ND (I),2)
  CN3 (I)=GCL3 (ND (I),3)
344 CONTINUE
C
DO 348 I=1,40
  PD (I)=0.0
348 CONTINUE
C
H (1)=1.0
H (2)=1.0
P (1)=0.577352692
P (2)=-P (1)
C
C
HH (1)=0.3478548451
C
HH (2)=H (1)
C
HH (3)=0.6521451548
C
HH (4)=H (3)
C
PP (1)=0.8611363115
C
PP (2)=-P (1)

```

```

C      PP (3)=0.3399810435
C      PP (4)=-P (3)

C      DO 150 I=1,2
C          DO 150 J=1,2
C              DO 150 K=1,2
C                  U=P (I)
C                  V=P (J)
C                  W=P (K)
C          CALL INTFC (IL,ND,I,J,K,U,V,W,X,Y,Z,VR (IR14),VR (IR28),
1          DETJ,VR (IR31),VR (IR32),VR (IR33),VR (IR29),
2          VR (IR37),VR (IR38),VR (IR36),VR (IR39),VR (IR40),
3          VR (IR30),VR (IR20),VR (IR47),VR (IR54),VR (IR55))

C          DO 150 M=1,40
C              PD (M)=PD (M)+H (I)*H (J)*H (K)*PDL (M)*DETJ
150 CONTINUE

C      RETURN
C      END
C      (END UPILD)

C      Subroutine UPILD is used to evaluate the equivalent load vector
C      caused by the change of temperature at every integration point.
C
C      SUBROUTINE INTFC (IL,ND,II,JJ,KK,R,S,T,X,Y,Z,VF,PDL,DETJ,BL,
1      TBL,TMPBL,VFE,TL,TT,TMP,EM,EM2,PDLL,SIGMA,
1      UPSIG,SVT3D,SVBLD)

C      IMPLICIT REAL*8 (A-H,O-Z)
C      IMPLICIT INTEGER*8 (I-N)
C      DIMENSION X (8),Y (8),Z (8),VF (NNODE,5),PDL (1),
1      BL (6,40),TBL (40,6),TMPBL (6,40),VFE (40),
2      A (8),B (8),C (8),D (8),E (8),G (8),ND (8),
3      TL (6,6),TT (6,6),TMP (6,6),EM (6,6),EM2 (6,6),
4      PDLL (40,1),SIGMA (NELM,2,2,2,9),UPSIG (NELM,2,2,2,9),
5      SIG (3,3),GRT (3,3),DV (3,3),SVT3D (NELM,1,2,2,144),
6      SS1 (3,3),SS2 (3,3),SS3 (3,3),AA (3,3),SA (6,1),STA (6,1),
7      SD (6,1),GAU (3,3),DGR (3,3),DGRT (3,3),EM3 (6,6),
8      GRD (9),GR (3,3),DW (3,3),SVBLD (NELM,2,2,2,24)

C      COMMON /SCHALR1/ NELM,NNODE,NT
C      COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
C      COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1      IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2      IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3      IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4      IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5      IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
C      COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
C      COMMON /UNICT/ NCONS,MODEL,ETAA,TDELTA,TINIT
C      COMMON /RLVEC/ VR (1)
C      COMMON /INTVEC/ IPT (1)
C      COMMON /GEO/ TO
C      COMMON /ABDFST/ ISEC
C      COMMON /CONTN/ INSIDT,KPDT,DTLM1
C      COMMON /A3/ CL1 (8),CM1 (8),CN1 (8),CL2 (8),CM2 (8),CN2 (8),
1      CL3 (8),CM3 (8),CN3 (8)
C      COMMON /TMPEF/ IDO,NTEM,NITR,NANM,CEXPN,TMMIN,TMINC,TMMAX,TMPP

C      DO 10 I=1,8
C          A (I)=0.0
C          B (I)=0.0

```

```

C          C (1)=0.0
C          D (1)=0.0
C          E (1)=0.0
C          G (1)=0.0
C
C 10  CONTINUE
C
C          CALL GEOM(R,S,T,TO,X,Y,Z,DETJ,A,B,C,D,E,G)
C
C          Get the geometric property at the integration point.
C
C          CALL MNU(6,40,BL)
C
C          DO 380 I=1,8
C
C              BL (1,I*5-4)=A (I)
C              BL (4,I*5-4)=B (I)
C              BL (6,I*5-4)=C (I)
C
C              BL (2,I*5-3)=B (I)
C              BL (4,I*5-3)=A (I)
C              BL (5,I*5-3)=C (I)
C
C              BL (3,I*5-2)=C (I)
C              BL (5,I*5-2)=B (I)
C              BL (6,I*5-2)=A (I)
C
C              BL (1,I*5-1)=-D (I)*CL2 (I)
C              BL (2,I*5-1)=-E (I)*CM2 (I)
C              BL (3,I*5-1)=-G (I)*CN2 (I)
C              BL (4,I*5-1)=-E (I)*CL2 (I)-D (I)*CM2 (I)
C              BL (5,I*5-1)=-G (I)*CM2 (I)-E (I)*CN2 (I)
C              BL (6,I*5-1)=-D (I)*CN2 (I)-G (I)*CL2 (I)
C
C              BL (1,I*5)=D (I)*CL1 (I)
C              BL (2,I*5)=E (I)*CM1 (I)
C              BL (3,I*5)=G (I)*CN1 (I)
C              BL (4,I*5)=E (I)*CL1 (I)+D (I)*CM1 (I)
C              BL (5,I*5)=G (I)*CM1 (I)+E (I)*CN1 (I)
C              BL (6,I*5)=D (I)*CN1 (I)+G (I)*CL1 (I)
C
C 380  CONTINUE
C
C          CALL MNU(6,6,TL)
C
C          CALL ROTMTRX(R,S,X,Y,Z,TL)
C
C          Get the rotation transformation matrix [T].
C
C          CALL TRANSP(6,6,TL,TT)
C
C          tt = t transpose.
C
C          SA (1,1)=CEXPN*TMINC
C          SA (2,1)=CEXPN*TMINC
C          SA (3,1)=CEXPN*TMINC
C          SA (4,1)=0.0
C          SA (5,1)=0.0
C          SA (6,1)=0.0
C
C          IF ((NCONS.EQ.1).AND.(III.GT.2)) THEN
C              CALL MMT(6,6,1,EM2,SA,EM3)
C          ELSE
C              CALL MMT(6,6,1,EM,SA,EM3)
C          END IF

```

```

C      WRITE (6,*) (EM3(I,1),I=1,6)
C      Get the elastic costant and will be changed by further consideration.
C
C      CALL MMT(6,6,1,TT,EM3,TMP)
C      WRITE (6,*) (TMP(I,1),I=1,6)
C
C      DO 720 I=1,6
C          STA(I,1)=TMP(I,1)
720 CONTINUE
C
C      CALL TRANSP(6,40,BL,TBL)
C      CALL MMT(40,6,1,TBL,STA,PDLL)
C
C      DO 80 I=1,40
C          PDL(I)=PDLL(I,1)
80 CONTINUE
C      RETURN
C      END
C      ( end INFC)
C
C      SUBROUTINE CRITR2(II,ND,DD1,FRCINC,ACTFRC,VLIMN,ICNC1)
C      IMPLICIT REAL*8(A-H,O-Z)
C      IMPLICIT INTEGER*8(I-N)
C
C      Subroutine CRITR2 is to build an exit criteria for the equilibrium
C      iterations.
C      input:
C      ii = The ii'th number iteration
C      DLDINC = The load increament
C      DLOADT = Te load level at that iteration.
C      PLD = The nodal force in last iteration
C      DVEC = The unknown solved in last iteration
C      VLINIT = the criteria value calculated in the first iteration.
C      Output:
C      ICONCL = The conclusion : Exit the loop or not.
C          1 = exit
C          0 = Keep inside the loop.
C
C      DIMENSION DD1(1),FRCINC(1),ACTFRC(1)
C      COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
C      COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1          IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2          IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3          IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4          IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5          IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
C
C      COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
C      COMMON /SCHALR1/ NELM,NNODE,NT
C      COMMON /RLVEC/ VR(1)
C      COMMON /INTVEC/ IPT(1)
C      COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
C
C      AINS=0.0
C      COEFF=70.0
C      VLIMNO=VLIMN
C      VAL=0.0
C      IF (II.EQ.1) THEN
C          VLINT1=0.0
C          DO 10 I=1,ND
C              TEMP=FRCINC(I)
C              AINS=AINS+TEMP
C              VLIMN=VLIMN+TEMP*TEMP
C          IF (I.LT.6) THEN

```

```

        WRITE (6,90) 11,1,DD1 (1),FRCINC (1),TEMP,ACTFRC (1)
        END IF
80    FORMAT ('11,1,D(1),FRCINC,TEMP: ',2I4,4F12.3)
10    CONTINUE
        VLIMN=SQRT (VLIMN)
        VAL=VLIMN
        ELSE
            DO 20 I=1,ND
                TEMP=-FRCINC (I)
                VAL=VAL+TEMP*TEMP
                IF (I.LT.6) THEN
                    WRITE (6,90) 11,1,DD1 (I),FRCINC (I),TEMP,ACTFRC (I)
                END IF
90    FORMAT ('11,1,D(1),FRCINC,ACTF: ',2I4,4F12.4)
20    CONTINUE
        VAL=SQRT (VAL)
        END IF

```

```

ICNC1=0
IF (VLIMN.GT.10.0) VLIMN=10.0
IF ((VAL*COEFF).LT.VLIMN) ICNC1=1
WRITE (6,50) VAL*COEFF,VLIMN,ICNC1
50 FORMAT ('VAL1,CRIT1,CONCL ARE: ',2F14.3,1I3)

```

```

RETURN
END

```

C  
C Subroutine RDCDT reads necessary data saved at last execution.  
C So the program can stop and resume the previous work.  
C

```

SUBROUTINE RDCDT (ACMDIS,SIGMA,XX1,YY1,ZZ1,XX,YY,ZZ,UPSIG,
1          FRCO,BETA,UPBET,GCL1,GCL2,GCL3,UCL1,UCL2,
3          UCL3,D1,TLTY,ANGL)
IMPLICIT REAL*8 (A-H,O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION DLOAD (1),DD1 (1),DD2 (1),PLD (1),ACMDIS (1),ANGL (1),
1          SIGMA (NELM,2,2,2,9),XX (1),YY (1),ZZ (1),XX1 (1),YY1 (1),
2          ZZ1 (1),UPSIG (NELM,2,2,9),FRCINC (1),FRCO (1),
3          BETA (NELM,2,2,2,12),UPBET (NELM,2,2,2,12),D1 (NNODE,5),
4          GCL1 (NNODE,3),GCL2 (NNODE,3),GCL3 (NNODE,3),
5          UCL1 (NNODE,3),UCL2 (NNODE,3),UCL3 (NNODE,3),TLTY (1)

```

C

```

COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,
1          NSHOW3,HRZ,ITRLM,FACTOR
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1          IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2          IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3          IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4          IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5          IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
COMMON /TMPEF/ IDO,NTEM,NITR,NANM,CEXP,N,TMIN,TMINC,TMMAX,TMPP
COMMON /DISV1/ IR70,IR71,IR72,IR73,IR74,IR75
COMMON /UNICT/ NCONS,MODEL,ETAA,TDELT,TINIT
COMMON /RLVEC/ VR (1)
COMMON /INTVEC/ IPT (1)
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
COMMON /CONTN/ INSIDT,KPDT,DTLM1
COMMON /SQ/ SQQ
COMMON /DISCT/ NDC,NDBC
COMMON /OUTVR/ NPT,NPV
COMMON /CREEP/ ICRP,NBCRP,NBDN,CRPTM,IPON
COMMON /CNTR/ ICNTR

```

C



```

READ (4,*) ICNTR
READ (4,*) TROOT
READ (4,*) DTLM1
READ (4,*) SQQ
READ (4,*) TMPP

C
IF (ICRP.EQ.1) THEN
  READ (4,*) NBDN,CRPTM
END IF

C
DO 689 I=1,NNODE
  READ (4,*) XX (1),YY (1),ZZ (1)
  WRITE (2,*) XX (1),YY (1),ZZ (1)
  XX1 (1)=XX (1)
  YY1 (1)=YY (1)
  ZZ1 (1)=ZZ (1)
689 CONTINUE
DO 687 I=1,NNODE
  READ (4,*) (GCL1 (I,J),J=1,3)
  READ (4,*) (GCL2 (I,J),J=1,3)
  READ (4,*) (GCL3 (I,J),J=1,3)
  DO 688 J=1,3
    UCL1 (I,J)=GCL1 (I,J)
    UCL2 (I,J)=GCL2 (I,J)
    UCL3 (I,J)=GCL3 (I,J)
688 CONTINUE
687 CONTINUE

C
DO 269 I=1,NELM
  DO 269 J=1,2
  DO 269 K=1,2
  DO 269 M=1,2
  DO 269 N=1,9
    READ (4,*) SIGMA (I,J,K,M,N)
    WRITE (2,*) SIGMA (I,J,K,M,N)
    UPSIG (I,J,K,M,N)=SIGMA (I,J,K,M,N)
269 CONTINUE

C
DO 669 I=1,NEQT
  READ (4,*) ACMDIS (I)
  WRITE (2,*) ACMDIS (I)
669 CONTINUE

C
DO 730 I=1,NEQT
  READ (4,*) FRCO (I)
  WRITE (2,*) FRCO (I)
730 CONTINUE

C
IF (NCONS.EQ.1) THEN
  DO 299 I=1,NELM
  DO 299 J=1,2
  DO 299 K=1,2
  DO 299 M=1,2
  DO 299 N=1,12
    READ (4,*) BETA (I,J,K,M,N)
    WRITE (2,*) BETA (I,J,K,M,N)
    UPBET (I,J,K,M,N)=BETA (I,J,K,M,N)
299 CONTINUE
END IF
IF (NDC.EQ.1) THEN
  DO 320 I=1,NNODE
  DO 320 J=1,5
    READ (4,*) D1 (I,J)
320 CONTINUE
DO 420 I=1,NDBC
  READ (4,*) TLTY (I)

```

```

420  CONTINUE
      IF (NPT.EQ.6) THEN
          DO 620 I=1,NDBC
              READ (4,*) ANGL (I)
620  CONTINUE
      END IF
END IF

C
C
      RETURN
      END
      END RDCDT

C
C
      Subroutine WTCDT write necessary data in file wrt.
      So the program can resume the execution when desired.
C

SUBROUTINE WTCDT (ACMDIS, SIGMA, XX1, YY1, ZZ1, XX, YY, ZZ, UPSIG,
1          FRCO, BETA, UPBET, GCL1, GCL2, GCL3, D1, TLTY, ANGL)
      IMPLICIT REAL*8 (A-H, O-Z)
      IMPLICIT INTEGER*8 (I-N)
      DIMENSION DLOAD (1), DD1 (1), DD2 (1), PLD (1), ACMDIS (1), ANGL (1),
1          SIGMA (NELM, 2, 2, 2, 9), XX (1), YY (1), ZZ (1), XX1 (1), YY1 (1),
2          ZZ1 (1), UPSIG (NELM, 2, 2, 2, 9), FRCINC (1), FRCO (1),
3          BETA (NELM, 2, 2, 2, 12), UPBET (NELM, 2, 2, 2, 12), TLTY (1),
4          GCL1 (NNODE, 3), GCL2 (NNODE, 3), GCL3 (NNODE, 3), D1 (NNODE, 5)
      COMMON /SCHALR1/ NELM, NNODE, NT
      COMMON /SCHALR2/ NEQT, NSTEP, NHBW, COEF1, COEF2, NSHOW1, NSHOW2,
1          NSHOW3, HRZ, ITRLM, FACTOR
      COMMON /PNTRIN/ IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10
      COMMON /PNTRRL/ IR1, IR2, IR3, IR4, IR5, IR6, IR7, IR8, IR9, IR10,
1          IR11, IR12, IR13, IR14, IR15, IR16, IR17, IR18,
2          IR19, IR20, IR21, IR22, IR23, IR24, IR25, IR26,
3          IR27, IR28, IR29, IR30, IR31, IR32, IR33, IR34,
4          IR35, IR36, IR37, IR38, IR39, IR40, IR41, IR42,
5          IR43, IR44, IR45, IR46, IR47, IR48, IR49, IR50
      COMMON /UNIFBD/ IR51, IR52, IR53, IR54, IR55, IR56, IR57, IR58, IR59
      COMMON /DISV1/ IR70, IR71, IR72, IR73, IR74, IR75
      COMMON /UNICT/ NCONS, MODEL, ETAA, TDELTA, TINIT
      COMMON /RLVEC/ VR (1)
      COMMON /INTVEC/ IPT (1)
      COMMON /ITESCH/ ROOT, DTLAM, SGN, IPP, TROOT, ASO, SP
      COMMON /CONTN/ INSIDT, KPDT, DTLM1
      COMMON /SQ/ SQQ
      COMMON /DISCT/ NDC, NDBC
      COMMON /OUTVR/ NPT, NPV
      COMMON /CREEP/ ICRP, NBCRP, NBDN, CRPTM, IPON
      COMMON /CNTR/ ICNTR
      COMMON /TMPEF/ IDO, NTEM, NITR, NANM, CEXP, TMIN, TMAX, TMPP

C
      WRITE (7,*) ICNTR
      WRITE (7,*) TROOT
      WRITE (7,*) DTLM1
      WRITE (7,*) SQQ
      WRITE (7,*) TMPP

C
      IF (ICRP.EQ.1) THEN
          WRITE (7,*) NBDN, CRPTM
          END IF
          DO 689 I=1, NNODE
              WRITE (7,*) XX (I), YY (I), ZZ (I)
689  CONTINUE
          DO 687 I=1, NNODE
              WRITE (7,*) (GCL1 (I, J), J=1, 3)
              WRITE (7,*) (GCL2 (I, J), J=1, 3)
              WRITE (7,*) (GCL3 (I, J), J=1, 3)
687  CONTINUE

```

```

DO 269 I=1,NELM
DO 269 J=1,2
DO 269 K=1,2
DO 269 M=1,2
DO 269 N=1,9
WRITE (7,*) SIGMA (I,J,K,M,N)
269 CONTINUE
DO 669 I=1,NEQT
WRITE (7,*) ACMDIS (I)
669 CONTINUE

DO 730 I=1,NEQT
WRITE (7,*) FRCO (I)
730 CONTINUE

IF (NCONS.EQ.1) THEN
DO 299 I=1,NELM
DO 299 J=1,2
DO 299 K=1,2
DO 299 M=1,2
DO 299 N=1,12
WRITE (7,*) BETA (I,J,K,M,N)
299 CONTINUE
END IF
IF (NDC.EQ.1) THEN
DO 320 I=1,NNODE
DO 320 J=1,5
WRITE (7,*) D1 (I,J)
320 CONTINUE
DO 420 I=1,NDBC
WRITE (7,*) TLTY (I)
420 CONTINUE
IF (NPT.EQ.6) THEN
DO 620 I=1,NDBC
WRITE (7,*) ANGL (I)
620 CONTINUE
END IF
END IF

C
C
RETURN
END
C
END WTCDT

C
C
NEXT SUBROUTINE IS USED TO UPDATA THE DIRECTION
C
COSINES OF VECTOR V1 AND V2 AT EVERY NODE.
C
C
INPUT: GCL3
C
OUTPUT: GCL1,GCL2
C
C
SUBROUTINE CNND (GCL1,GCL2,GCL3)
C
IMPLICIT REAL*8 (A-H,O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION GCL1 (NNODE,3),GCL2 (NNODE,3),GCL3 (NNODE,3)
COMMON /DIRCS/ IR60,IR61,IR62,IR63,IR64,IR65
COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /RLVEC/ VR (1)
COMMON /INTVEC/ IPT (1)

C
C
DO 10 I=1,NNODE

```

```

      CMD=(GCL3(1,1)*GCL3(1,1)+GCL3(1,3)*GCL3(1,3))**.5
      GCL1(1,1)=GCL3(1,3)/CMD
      GCL1(1,2)=0.0
      GCL1(1,3)=-GCL3(1,1)/CMD
      TM1=GCL3(1,1)*GCL3(1,1)+GCL3(1,3)*GCL3(1,3)
      TM2=GCL3(1,2)*(GCL3(1,1)+GCL3(1,3))
      CMD=(TM1*TM1+TM2*TM2)**.5
      GCL2(1,1)=0.0
      GCL2(1,2)=TM1/CMD
      GCL2(1,3)=-TM2/CMD
10  CONTINUE
C
      RETURN
      END
C
C      Subroutine is for additional data input.
C
      SUBROUTINE RDSUP(GCL1,GCL2,GCL3,UCL1,UCL2,UCL3,ANGL)
      IMPLICIT REAL*8(A-H,O-Z)
      IMPLICIT INTEGER*8(I-N)
      DIMENSION GCL1(NNODE,3),GCL2(NNODE,3),GCL3(NNODE,3),
1      UCL1(NNODE,3),UCL2(NNODE,3),UCL3(NNODE,3),ANGL(1)
      COMMON /DIRCS/ IR60,IR61,IR62,IR63,IR64,IR65
      COMMON /SCHALR1/ NELM,NNODE,NT
      COMMON /RLVEC/ VR(1)
      COMMON /INTVEC/ IPT(1)
      COMMON /DISCT/ NDC,NDBC
      COMMON /OUTVR/ NPT,NPV
      COMMON /RADS/ RR,ZL
C
      DO 10 I=1,NNODE
      READ(5,*) IA,(GCL3(I,J),J=1,3)
10  CONTINUE
C
      CALL CNND(VR(IR60),VR(IR61),VR(IR62))
      DO 20 I=1,NNODE
      DO 30 J=1,3
      UCL1(I,J)=GCL1(I,J)
      UCL2(I,J)=GCL2(I,J)
      UCL3(I,J)=GCL3(I,J)
30  CONTINUE
C      WRITE(6,*) I,' UCL1=',(UCL1(I,J),J=1,3)
C      WRITE(6,*) I,' UCL2=',(UCL2(I,J),J=1,3)
C      WRITE(6,*) I,' UCL3=',(UCL3(I,J),J=1,3)
20  CONTINUE
C
      IF(NPT.EQ.6) THEN
      DO 50 I=1,NDBC
      READ(5,*) ANGL(I)
      WRITE(6,*) .ANGL(I)
50  CONTINUE
      READ(5,*) RR
      END IF
      IF(NPT.EQ.4.OR.NPT.EQ.5.OR.NPT.EQ.6) THEN
      READ(5,*) RR,ZL
      END IF
      RETURN
      END
C
C      Subroutine CB is to calculate the stiffness matrix at every
C      integeation point
C
      SUBROUTINE CB(III,IL,JL,KL,ML,R,S,T,X,Y,Z,DETJ,ESM,BN1,BN2,
1      BN3,BL,TBL,MP,EM2,SS,SS1,TMP,TL,TT,EM,EM2,UPSIG,
2      EXED,BDL,BDSV,EM4)
C

```

```

IMPLICIT REAL*8 (A-H,O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION X (8), Y (8), Z (8), ESM (40,40), BN1 (40,40), BN2 (40,40),
1      BN3 (40,40), BL (6,40), TBL (40,6), TMPEM2 (6,40), SS (9,9),
2      SSI (9,9), TMP (6,6), TL (6,6), TT (6,6), EM (6,6), EM2 (6,6),
3      A (8), B (8), C (8), D (8), E (8), G (8), SIG (3,3),
4      UPSIG (NELM,2,2,2,9), EXED (40), BDLD (1),
5      BDSV (NELM,2,2,2,6), EM4 (NELM,2,2,2,36)
COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1      IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2      IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3      IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4      IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5      IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
COMMON /UNICT/ NCONS,MODEL,ETAA,TDEL,TINIT
COMMON /A3/ CL1 (8),CM1 (8),CN1 (8),CL2 (8),CM2 (8),CN2 (8),
1      CL3 (8),CM3 (8),CN3 (8)
COMMON /RLVEC/ VR (1)
COMMON /INTVEC/ IPT (1)
COMMON /GEO/ TO
COMMON /ABDFST/ ISEC
COMMON /CONTN/ INSIDT,KPDT,DTLM1

C
IPR=0
IF (IL.EQ.1.AND.JL.EQ.1.AND.KL.EQ.1.AND.ML.EQ.1) IPR=1

C
CALL GEOM (R,S,T,TO,X,Y,Z,DETJ,A,B,C,D,E,G)

C
WRITE (6,*) R,S,T,TO,DETJ

C
CALL MNU (6,40,VR (IR31))

C
DO 440 I=1,3
  DO 440 J=1,3
    SIG (I,J)=UPSIG (IL,JL,KL,ML,I*3-3+J)
440 CONTINUE

C
Get the linear part of matrix [B].

C
DO 380 I=1,8
  BL (1,I*5-4)=A (I)
  BL (4,I*5-4)=B (I)
  BL (6,I*5-4)=C (I)

C
  BL (2,I*5-3)=B (I)
  BL (4,I*5-3)=A (I)
  BL (5,I*5-3)=C (I)

C
  BL (3,I*5-2)=C (I)
  BL (5,I*5-2)=B (I)
  BL (6,I*5-2)=A (I)

C
  BL (1,I*5-1)=-D (I)*CL2 (I)
  BL (2,I*5-1)=-E (I)*CM2 (I)
  BL (3,I*5-1)=-G (I)*CN2 (I)
  BL (4,I*5-1)=-E (I)*CL2 (I)-D (I)*CM2 (I)
  BL (5,I*5-1)=-G (I)*CM2 (I)-E (I)*CN2 (I)
  BL (6,I*5-1)=-D (I)*CN2 (I)-G (I)*CL2 (I)

C
  BL (1,I*5)=D (I)*CL1 (I)
  BL (2,I*5)=E (I)*CM1 (I)
  BL (3,I*5)=G (I)*CN1 (I)
  BL (4,I*5)=E (I)*CL1 (I)+D (I)*CM1 (I)

```

```

      BL (5,1*5)=G (1) *CM1 (1)+E (1) *CN1 (1)
      BL (6,1*5)=D (1) *CN1 (1)+G (1) *CL1 (1)
380 CONTINUE
C
      CALL ROTMTRX (R,S,X,Y,Z,TL)
C
C      Get the rotation transformation matrix [T].
C
      CALL TRANSP (6,6,TL,TT)
C
      tt = t transpose.
C
      CALL MMT (6,6,6,TT,EM,TMP)
      CALL MMT (6,6,6,TMP,TL,EM2)
C
      IEEC=0
      IF (ISEC.EQ.1.OR.ISEC.EQ.2) IEEC=1
      IF ((NCONS.EQ.1).AND.((ISEC.NE.1).OR.(INSIDT.EQ.1)).
1 AND.((I11.EQ.1).OR.(ISEC.EQ.2))) THEN
      IF (MODEL.EQ.1) THEN
          CALL BODNER (I11,IL,JL,KL,ML,SIG,VR (IR28),VR (IR40),VR (IR36),
1              VR (IR51),VR (IR53),VR (IR54),VR (IR55),
2              VR (IR30),VR (IR56),VR (IR57),VR (IR33))
      ELSE
          CALL WALKER (I11,IL,JL,KL,ML,SIG,VR (IR28),VR (IR40),VR (IR36),
1              VR (IR51),VR (IR53),VR (IR54),VR (IR55),
2              VR (IR30),VR (IR56),VR (IR57),VR (IR33))
      END IF
      END IF
C
      CALL TRANSP (6,40,BL,TBL)
      CALL TRANSP (6,40,VR (IR31),VR (IR32))
C
      tbl = b1 transpose.
C
      CALL MMT (6,6,40,EM2,BL,TPPEM2)
      CALL MMT (40,6,40,TBL,TPPEM2,ESM)
C
      IF (IPR.EQ.1) THEN
      DO 3 I=1,40
      C      WRITE (6,*) I,' ',ESM (I,1)=' ',ESM (I,1)
C 3 CONTINUE
      END IF
C
C
      IF (NCONS.EQ.1) THEN
      DO 350 I=1,40
      EXED (I)=0.0
      DO 349 J=1,6
      EXED (I)=EXED (I)+TBL (I,J)*BDLD (J)
C      WRITE (6,*) I,J,' EXED ',EXED (I),' TBL ',TBL (I,J),' ',BDLD (J)
C 349 CONTINUE
      IF (IPR.EQ.1) WRITE (6,*) 'EXED IN CB: ',EXED (I)
      WRITE (6,*) I,' EXED IN CB=',EXED (I)
C 350 CONTINUE
      END IF
C
      CALL MNU (9,9,SS)
C
      DO 520 I=1,3
      DO 520 J=1,3
      SS (I,J)=SIG (I,J)
      SS (I+3,J+3)=SIG (I,J)
      SS (I+6,J+6)=SIG (I,J)
C 520 CONTINUE
C

```

```

DO 530 I=1,3
  SS1 (I,I*3-2)=SIG (I,1)
  SS1 (I,I*3-1)=SIG (I,2)
  SS1 (I,I*3)=SIG (I,3)

  SS1 (I+3,I*3-2)=SIG (2,1)
  SS1 (I+3,I*3-1)=SIG (2,2)
  SS1 (I+3,I*3)=SIG (2,3)

  SS1 (I+6,I*3-2)=SIG (3,1)
  SS1 (I+6,I*3-1)=SIG (3,2)
  SS1 (I+6,I*3)=SIG (3,3)
530 CONTINUE

CALL NONLM (A,B,C,D,E,G,VR (IR34),VR (IR35),VR (IR28),
1          VR (IR29),VR (IR30),VR (IR31),VR (IR32),VR (IR33))

Get the nonlinear part (rotation invariant) of the matrix ESM.

DO 441 I=1,40
  DO 441 J=1,40
    ESM (I,J)=ESM (I,J)+BN1 (I,J)-2*BN2 (I,J)+BN3 (I,J)
    WRITE (6,460) I,J,ESM (I,J),BN1 (I,J),BN2 (I,J),BN3 (I,J)
441 CONTINUE
460 FORMAT ('ESM (I,J) IS:',2I3,4F10.3)

RETURN
END

Subroutine CBUPDT is to calculate the nodal forces at every
integration point and update stresses for that point.

SUBROUTINE CBUPDT (III,IL,ND,II,JJ,KK,R,S,T,X,Y,Z,VF,PDL,DETJ,BL,
1          TBL,TMPBL,VFE,TL,TT,TMP,EM,EM2,PDLL,SIGMA,
1          UPSIG,SVT3D,SVBLD,EM4)
  IMPLICIT REAL*8 (A-H,O-Z)
  IMPLICIT INTEGER*8 (I-N)
  DIMENSION X (8),Y (8),Z (8),VF (NNODE,5),PDL (1),BL (6,40),
1          TBL (40,6),TMPBL (6,40),VFE (40),A (8),B (8),C (8),
2          D (8),E (8),G (8),ND (8),TL (6,6),TT (6,6),TMP (6,6),
3          EM (6,6),EM2 (6,6),PDLL (40,1),SIGMA (NELM,2,2,2,9),
4          UPSIG (NELM,2,2,2,9),SIG (3,3),GRT (3,3),DV (3,3),
5          SVT3D (NELM,2,2,2,114),SS1 (3,3),SS2 (3,3),SS3 (3,3),
6          AA (3,3),SA (6,1),SD (6,1),GAU (3,3),DGR (3,3),DGRT (3,3),
7          AAAA (6,1),GRD (9),GR (3,3),DW (3,3),SVBLD (NELM,2,2,2,19),
8          EM4 (NELM,2,2,2,36)

  COMMON /SCHALR1/ NELM,NNODE,NT
  COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
  COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1          IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2          IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3          IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4          IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5          IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
  COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
  COMMON /UNICT/ NCONS,MODEL,ETAA,TDEL,TINIT
  COMMON /RLVEC/ VR (1)
  COMMON /INTVEC/ IPT (1)
  COMMON /GEO/ TO
  COMMON /ABDFST/ ISEC
  COMMON /CONTN/ INSIDT,KPDT,DTLM1
  COMMON /NMBITR/ NUM
  COMMON /TMPCO/ ICTMP
  COMMON /A3/ CL1 (8),CM1 (8),CN1 (8),CL2 (8),CM2 (8),CN2 (8),
1          CL3 (8),CM3 (8),CN3 (8)

```

COMMON /TMPEF/ IDO,NTEM,NITR,NANM,CEXPN,TMMIN,TMINC,TMMAX,TMPP

```
C
IPR=0
IF (11.EQ.1.AND.JJ.EQ.1.AND.KK.EQ.1) IPR=1
DO 10 I=1,8
  A(I)=0.0
  B(I)=0.0
  C(I)=0.0
  D(I)=0.0
  E(I)=0.0
  G(I)=0.0
10 CONTINUE
C
CALL GEOM(R,S,T,TO,X,Y,Z,DETJ,A,B,C,D,E,G)
C
DO 30 I=1,8
  DO 30 J=1,5
    VFE(I*5-5+J)=VF(ND(I),J)
30 CONTINUE
C
DO 695 I=1,9
  GRD(I)=0.0
695 CONTINUE
C
DO 700 I=1,8
  K=I*5
  GRD(1)=GRD(1)+A(I)*VFE(K-4)+D(I)*(-CL2(I)*VFE(K-1)
1      +CL1(I)*VFE(K))
  GRD(2)=GRD(2)+B(I)*VFE(K-4)+E(I)*(-CL2(I)*VFE(K-1)
1      +CL1(I)*VFE(K))
  GRD(3)=GRD(3)+C(I)*VFE(K-4)+G(I)*(-CL2(I)*VFE(K-1)
1      +CL1(I)*VFE(K))
C
  GRD(4)=GRD(4)+A(I)*VFE(K-3)+D(I)*(-CM2(I)*VFE(K-1)
1      +CM1(I)*VFE(K))
  GRD(5)=GRD(5)+B(I)*VFE(K-3)+E(I)*(-CM2(I)*VFE(K-1)
1      +CM1(I)*VFE(K))
  GRD(6)=GRD(6)+C(I)*VFE(K-3)+G(I)*(-CM2(I)*VFE(K-1)
1      +CM1(I)*VFE(K))
C
  GRD(7)=GRD(7)+A(I)*VFE(K-2)+D(I)*(-CN2(I)*VFE(K-1)
1      +CN1(I)*VFE(K))
  GRD(8)=GRD(8)+B(I)*VFE(K-2)+E(I)*(-CN2(I)*VFE(K-1)
1      +CN1(I)*VFE(K))
  GRD(9)=GRD(9)+C(I)*VFE(K-2)+G(I)*(-CN2(I)*VFE(K-1)
1      +CN1(I)*VFE(K))
700 CONTINUE
C
COMP=GRD(1)+GRD(5)+GRD(9)
CCOMP=1.0-COMP
C
DO 720 I=1,3
  DO 720 J=1,3
    GR(I,J)=GRD(I+J*3-3)
    IF (I.EQ.J) THEN
      DGR(I,J)=GR(I,J)+1.0
    ELSE
      DGR(I,J)=GR(I,J)
    END IF
    GRT(J,I)=GR(I,J)
    DGRT(J,I)=DGR(I,J)
720 CONTINUE
C
DETG=DGR(1,1)*DGR(2,2)*DGR(3,3)+DGR(2,1)*DGR(3,2)*DGR(1,3)
1  +DGR(3,1)*DGR(1,2)*DGR(2,3)-DGR(3,1)*DGR(2,2)*DGR(1,3)
2  -DGR(2,1)*DGR(1,3)*DGR(3,3)-DGR(1,1)*DGR(3,2)*DGR(2,3)
```



```

C WRITE (6,722) DETG
722 FORMAT ('DETG IS: ',1F10.6)
C
C DO 740 I=1,3
DO 740 J=1,3
GRT (I,J)=GR (J,I)
DV (I,J)=0.5*(GRT (I,J)+GR (I,J))
DW (I,J)=0.5*(GRT (I,J)-GR (I,J))
C WRITE (6,741) I,J,GRT (I,J),DV (I,J),DW (I,J)
740 CONTINUE
741 FORMAT ('I,J,GRT,DV,DW: ',2I3,3F12.5)
C
C DO 440 I=1,3
DO 440 J=1,3
SIG (I,J)=UPSIG (IL,II,JJ,KK,I*3-3+J)
440 CONTINUE
450 FORMAT ('SIG (I,J) IS: ',2I3,1F13.5)
C
C CALL MNU (6,40,BL)
C
C DO 380 I=1,8
BL (1,I*5-4)=A (I)
BL (4,I*5-4)=B (I)
BL (6,I*5-4)=C (I)
C
BL (2,I*5-3)=B (I)
BL (4,I*5-3)=A (I)
BL (5,I*5-3)=C (I)
CC
BL (3,I*5-2)=C (I)
BL (5,I*5-2)=B (I)
BL (6,I*5-2)=A (I)
CC
BL (1,I*5-1)=-D (I)*CL2 (I)
BL (2,I*5-1)=-E (I)*CM2 (I)
BL (3,I*5-1)=-G (I)*CN2 (I)
BL (4,I*5-1)=-E (I)*CL2 (I)-D (I)*CM2 (I)
BL (5,I*5-1)=-G (I)*CM2 (I)-E (I)*CN2 (I)
BL (6,I*5-1)=-D (I)*CN2 (I)-G (I)*CL2 (I)
C
BL (1,I*5)=D (I)*CL1 (I)
BL (2,I*5)=E (I)*CM1 (I)
BL (3,I*5)=G (I)*CN1 (I)
BL (4,I*5)=E (I)*CL1 (I)+D (I)*CM1 (I)
BL (5,I*5)=G (I)*CM1 (I)+E (I)*CN1 (I)
BL (6,I*5)=D (I)*CN1 (I)+G (I)*CL1 (I)
380 CONTINUE
C
CALL TRANSP (6,40,BL,TBL)
C
CALL MNU (6,6,TL)
C
CALL ROTMTRX (R,S,X,Y,Z,TL)
C
Get the rotation transformation matrix [T].
C
CALL TRANSP (6,6,TL,TT)
C
ICGO=0
C IF (IPR.EQ.1) WRITE (6,*) 'III=',III,' ISEC=',ISEC
IF (NUM.EQ.1.AND.INSIDT.EQ.0) GOTO 345
IF ((NCONS.EQ.1).AND.(III.NE.1)) THEN
DO 453 I=1,6
DO 453 J=1,6

```

```

      EM2 ( I , J ) = EM4 ( IL , II , JJ , KK , I * 6 - 6 + J )
453  CONTINUE
      ICGO = 1
      END IF
345  CONTINUE
      IF ( ICGO . EQ . 1 ) GOTO 988
      CALL MMT ( 6 , 6 , 6 , TT , EM , TMP )
      CALL MMT ( 6 , 6 , 6 , TMP , TL , EM2 )
988  CONTINUE
C
      CALL MNU ( 6 , 40 , VR ( IR33 ) )
C
      CALL MMT ( 6 , 40 , 1 , BL , VFE , AAAA )
C
      IF ( ICTMP . EQ . 1 ) THEN
C
      For thermal effects calculation
      EXPNS = CEXPN * TMINC
      AAAA ( 1 , 1 ) = AAAA ( 1 , 1 ) - EXPNS
      AAAA ( 2 , 1 ) = AAAA ( 2 , 1 ) - EXPNS
      AAAA ( 3 , 1 ) = AAAA ( 3 , 1 ) - EXPNS
      END IF
C
      CALL MMT ( 6 , 6 , 1 , EM2 , AAAA , SD )
      K = 1
C
      sd will be the stress increment
C
      CALL TRANSP ( 6 , 40 , BL , TBL )
280  CONTINUE
C
C
      IF ( NUM . EQ . 1 . AND . INSIDT . EQ . 0 ) GOTO 875
      IECC = 0
      IF ( ISEC . EQ . 1 . OR . ISEC . EQ . 2 ) IECC = 1
      IF ( NCONS . EQ . 1 . AND . III . EQ . 1 ) THEN
          IF ( IPR . EQ . 1 ) THEN
              WRITE ( 6 , * ) 'CALL BODSUL '
          END IF
          IF ( MODEL . EQ . 1 ) THEN
              CALL BODSUL ( IL , II , JJ , KK , VR ( IR31 ) , VR ( IR29 ) , VR ( IR54 ) ,
1              VR ( IR55 ) , VR ( IR51 ) , SD , VR ( IR56 ) , VR ( IR57 ) , AAAA )
          ELSE
              CALL WALSUL ( IL , II , JJ , KK , VR ( IR31 ) , VR ( IR29 ) , VR ( IR54 ) ,
1              VR ( IR55 ) , VR ( IR51 ) , SD , VR ( IR56 ) , VR ( IR57 ) , AAAA )
          END IF
C
          ELSE
              IF ( NCONS . EQ . 1 ) THEN
                  IF ( MODEL . EQ . 1 ) THEN
                      CALL BODS2 ( IL , II , JJ , KK , VR ( IR31 ) , VR ( IR29 ) , VR ( IR54 ) ,
1                      VR ( IR55 ) , VR ( IR51 ) , SD , VR ( IR56 ) , VR ( IR57 ) , AAAA )
                  ELSE
                      CALL WALS2 ( IL , II , JJ , KK , VR ( IR31 ) , VR ( IR29 ) , VR ( IR54 ) ,
1                      VR ( IR55 ) , VR ( IR51 ) , SD , VR ( IR56 ) , VR ( IR57 ) , AAAA )
                  END IF
              END IF
              END IF
875  CONTINUE
C
      GAU ( 1 , 1 ) = SD ( 1 , 1 )
      GAU ( 2 , 2 ) = SD ( 2 , 1 )
      GAU ( 3 , 3 ) = SD ( 3 , 1 )
      GAU ( 1 , 2 ) = SD ( 4 , 1 )
      GAU ( 2 , 1 ) = GAU ( 1 , 2 )
      GAU ( 2 , 3 ) = SD ( 5 , 1 )
      GAU ( 3 , 2 ) = GAU ( 2 , 3 )

```

```

GAU(3,1)=SD(6,1)
GAU(1,3)=GAU(3,1)

DO 758 I=1,3
  DO 758 J=1,3
    AA(I,J)=GAU(I,J)
758 CONTINUE

C
C
DO 760 I=1,3
  DO 760 J=1,3
    UPSIG(IL,II, JJ, KK, I*3-3+J)=UPSIG(IL,II, JJ, KK, I*3-3+J)
    1 AA(I,J)=UPSIG(IL,II, JJ, KK, I*3-3+J)
    +AA(I,J)
    AA(I,J)=UPSIG(IL,II, JJ, KK, I*3-3+J)
760 CONTINUE

C
C
SA(1,1)=AA(1,1)*CCOMP
SA(2,1)=AA(2,2)*CCOMP
SA(3,1)=AA(3,3)*CCOMP
SA(4,1)=AA(1,2)*CCOMP
SA(5,1)=AA(2,3)*CCOMP
SA(6,1)=AA(1,3)*CCOMP

C
C
CALL MMT(40,6,1,TBL,SA,PDLL)
900 CONTINUE
DO 80 I=1,40
  PDL(I)=PDLL(I,1)
80 CONTINUE
90 FORMAT('HERE PDL(I) IS: ',1I3,1F12.7)

C
RETURN
END

C
C*****
C Subroutine WAL2 is the solution phase using Walker's constitutive
C equation. It is called after the first iteration.
C Input:
C BL- used to find the local strain.
C VFE- the displace increament. epsln=bl.vfe
C SVT3D and SVBLD are the data calculated in the processing face.
C State variable BETA(..12) are updated.
C The derivative of the statevariable STVDF and the derivative of the
C nonlinear strain EPSND are calculated.
C*****
C
SUBROUTINE WAL2(IAA,IA,IB,IC,BL,VFE,SVT3D,SVBLD,BETA,SD,
1 BDSV,EM4,AA)
C
IMPLICIT REAL*8(A-H,O-Z)
IMPLICIT INTEGER*8(I-N)
DIMENSION BL(6,40),VFE(1),SVT3D(NELM,2,2,2,144),TMVEC(24),
1 SVBLD(NELM,2,2,2,24),BETA(NELM,2,2,2,12),SD(6,1),
2 BDSV(NELM,2,2,2,6),EM4(NELM,2,2,2,36),AA(6,1),
3 DBTA1(6),DBTA2(6)
C
COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,
1 NSHOW3,HRZ,ITRLM,FACTOR
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,

```

```

3          IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4          IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5          IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /RLVEC/ VR(1)
COMMON /INTVEC/ IPT(1)
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
COMMON /GEO/ TO
COMMON /CONTRL/ DETMNT
COMMON /CONTN/ INSIDT,KPDT,DTLM1
COMMON /WAL/ WK,WB,WN2,WN3,WN4,WN5,WN6,WN8,WN9,WN10,WN11,WRO
COMMON /UNICT/ NCONS,MODEL,ETAA,TDELT,TINIT
COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
COMMON /WKLMT/ WAL1,WAL2
C
C ) IPR=0
C   IF ((IA.EQ.1).AND.(IB.EQ.1).AND.(IC.EQ.1)) IPR=1
C
C   DO 60 I=1,24
C     TMVEC(I)=0.0
C     DO 80 J=1,6
C       TMVEC(I)=TMVEC(I)-SVT3D(IAA,IA,IB,IC,I*6-6+J)*AA(J,1)
80    CONTINUE
60    CONTINUE
C
C   DO 100 I=1,6
C     SD(1,1)=TMVEC(I)
C     DBTA1(I)=TMVEC(I+12)
C     DBTA2(I)=TMVEC(I+18)
100   CONTINUE
C
C   WRITE(6,*) 'DSIGX=',SD(1,1),' DSY=',SD(2,1),' DSZ=',SD(3,1)
C
C   DO 120 I=1,6
C     BETA(IAA,IA,IB,IC,I)=BETA(IAA,IA,IB,IC,I)+DBTA1(I)
C     BETA(IAA,IA,IB,IC,I+6)=BETA(IAA,IA,IB,IC,I+6)+DBTA2(I)
C     IF (BETA(IAA,IA,IB,IC,I).GT.WAL1) BETA(IAA,IA,IB,IC,I)=WAL1
C     IF (BETA(IAA,IA,IB,IC,I).LT.-WAL1) BETA(IAA,IA,IB,IC,I)=-WAL1
C     IF (BETA(IAA,IA,IB,IC,I+6).GT.WAL2) BETA(IAA,IA,IB,IC,I+6)=WAL2
C     IF (BETA(IAA,IA,IB,IC,I+6).LT.-WAL1) BETA(IAA,IA,IB,IC,I+6)=-WAL2
120   CONTINUE
C
C   RETURN
C   END
C   END(WALS2)
C
C*****
C   Subroutine BODS2 is the solution phase using Bodner's constitutive
C   equation. It is called after the first iteration.
C   Input:
C   BL- used to find the local strain.
C   VFE- the displace increament. epsln=bl.vfe
C   SVT3D and SVBLD are the data calculated in the processing face.
C   State variable BETA(..12) are updated.
C   The derivative of the statevariable STVDF and the derivative of the
C   nonlinear strain EPSND are calculated.
C*****
SUBROUTINE BODS2(IAA,IA,IB,IC,BL,VFE,SVT3D,SVBLD,BETA,SD,
1          BDSV,EM4,AA)
C
C   IMPLICIT REAL*8(A-H,O-Z)
C   IMPLICIT INTEGER*8(I-N)
C   DIMENSION BL(6,40),VFE(1),SVT3D(NELM,2,2,2,144),TMVEC(20),
1          SVBLD(NELM,2,2,2,24),BETA(NELM,2,2,2,12),SD(6,1),
2          BDSV(NELM,2,2,2,6),EM4(NELM,2,2,2,36),AA(6,1),
3          DLBET(6),TMV(19)
C

```

```

COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,
1      NSHOW3,HRZ,IITRLM,FACTOR
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1      IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2      IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3      IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4      IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5      IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /RLVEC/ VR(1)
COMMON /INTVEC/ IPT(1)
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
COMMON /GEO/ TO
COMMON /CONTRL/ DETMNT
COMMON /CONTN/ INSIDT,KPDT,DTLM1
COMMON /UNICT/ NCONS,MODEL,ETAA,TDELTA,TINIT
COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59
COMMON /BOD/ DO,ZCO,ZC1,ZC2,ZC3,ZM1,ZM2,CA1,CA2,CR1,CR2,ZNO
COMMON /CREEP/ ICRP,NBCRP,NBDN,CRPTM,IPON

C
IPR=0
IF ((IA.EQ.1).AND.(IB.EQ.1).AND.(IC.EQ.1)) IPR=1

C
DO 80 I=1,19
  TMV(I)=0.0
  TMVEC(I)=0.0
  DO 80 J=1,6
    TMVEC(I)=TMVEC(I)-SVT3D(IAA,IA,IB,IC,I*6-6+J)*AA(J,1)
80 CONTINUE
DO 60 I=1,19
  TMV(I)=TMVEC(I)
60 CONTINUE

C
DO 100 I=1,6
  SD(I,1)=TMVEC(I)
C
IF (IPR.EQ.1) WRITE(6,*) 'SD IN BODS2',SD(I,1)
DLBET(I)=TMVEC(I+13)
100 CONTINUE

C
DO 120 I=1,6
  BETA(IAA,IA,IB,IC,I)=BETA(IAA,IA,IB,IC,I)+DLBET(I)
  IF (BETA(IAA,IA,IB,IC,I).GT.ZC3) BETA(IAA,IA,IB,IC,I)=ZC3
  IF (BETA(IAA,IA,IB,IC,I).LT.-ZC3) BETA(IAA,IA,IB,IC,I)=-ZC3
120 CONTINUE
  BETA(IAA,IA,IB,IC,7)=BETA(IAA,IA,IB,IC,7)+TMVEC(13)
  IF (BETA(IAA,IA,IB,IC,7).GT.ZC1) BETA(IAA,IA,IB,IC,7)=ZC1
  IF (BETA(IAA,IA,IB,IC,7).LT.(2.0*ZCO-ZC1)) BETA(IAA,IA,IB,IC,7)=
1    2.0*ZCO-ZC1

C
RETURN
END
C
END(BODS2)
C
C*****
C Subroutine OUTPUT is used to arrange the output data. Here C
C D1(i,j) is the displacement matrix, where i and j are the node C
C number and displace component number respectively. The coordinates C
C of node i are XX(I), YY(I), ZZ(I). The corresponding load can be C
C calculated as the product of TROOT, load coefficient and the C
C applied load (given in file dt). C
C*****
C
C
SUBROUTINE OUTPUT (TTLD,D1,ANGL,TTLTY,XX,YY,ZZ)
IMPLICIT REAL*8 (A-H,O-Z)

```

```

IMPLICIT INTEGER*8 (I-N)
DIMENSION D1 (NNODE,5), ANGL (1), TTLY (1), XX (1), YY (1), ZZ (1)
COMMON /SCHALR1/ NELM,NNODE,NT
COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,
1      NSHOW3,HRZ,ITRLM,FACTOR
COMMON /RLVEC/ VR (1)
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP
COMMON /GEO/ TO
COMMON /MTL/ E,EU
COMMON /DISCT/ NDC,NDBC
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,
1      IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,
2      IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,
3      IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,
4      IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,
5      IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10
COMMON /INTVEC/ IPT (1)
COMMON /OUTVR/ NPT,NPV
COMMON /RADS/ RR,ZL
COMMON /CREEP/ ICRP,NBCRP,NBDN,CRPTM,IPON
COMMON /UNICT/ NCONS,MODEL,ETAA,TDELTA,TINIT
I=NSHOW1
C      NPT=1 : STRECH
C      NPT=2 : PLATE
C      NPT=3: PANEL
C      NPT=4: CYLINDRICAL SHELL UNDER AXIAL COMPRESSION
C      NPT=5: CYLINDRICAL SHELL UNDER PRESSURE
C      NPT=6: CYLINDRICAL SHELL UNDER TORSION
C
IF (NDC.EQ.0) THEN
  IF (NPT.EQ.1) THEN
    IF (ICRP.EQ.1) THEN
      WRITE (3,*) D1 (1,2)/20.0*100.0,' ',TROOT*2.0/TO/20.0,' ',
1      CRPTM
    ELSE
      WRITE (3,*) D1 (1,2)/20.0*100.0,' ',TROOT*2.0/TO/20.0
    END IF
  END IF
C
  IF (NPT.EQ.2) THEN
    DDK=3.14159**2*E*TO**3/12.0
    DDK2=3.14159**2*198700.0*TO**3/12.0
    IF (ICRP.EQ.1) THEN
      WRITE (3,*) D1 (1,3)/TO,' ',TROOT/DDK2,' ',TROOT/TO,' ',
1      CRPTM
    ELSE
      WRITE (3,*) D1 (1,3)/TO,' ',TROOT/DDK,' ',TROOT/DDK2,' ',
1      TROOT/TO
    END IF
    DO 55 J=1,NNODE
      WRITE (12,12) J,(D1 (J,KK)*1000.0,KK=1,3)
55    CONTINUE
12    FORMAT (115,3F12.5)
23    FORMAT (7F8.3,1F7.1)
  END IF
C
  IF (NPT.EQ.3) THEN
    WRITE (3,*) -D1 (1,3)*1000.0,' ',TROOT*4.0*1000.0
    IF (NPV.EQ.1) THEN
      WRITE (12,13) D1 (8,3)*1000.0,D1 (13,3)*1000.0,
1      D1 (16,3)*1000.0,D1 (21,3)*1000.0,TROOT*4.0*1000.0
13    FORMAT ('0.0',' ',4F10.5,1F11.5)
    END IF
  END IF
C

```

```

IF (NPT.EQ.4) THEN
  IF (NPV.EQ.1) THEN
    KKN=9
    KKO=33
  END IF
  IF (NPV.EQ.2) THEN
    KKN=5
    KKO=19
  END IF
  IF (NPV.EQ.3) THEN
    KKN=16
    KKO=60
  END IF
  IF (NPV.EQ.4) THEN
    KKN=32
    KKO=60
  END IF
  WT=0.0
  DO 100 I=1,KKN
    WT=WT+D1(I,2)
  CONTINUE
  WT=WT/REAL(KKN)
  WOUT=0.0
  DO 200 I=1,NNODE
    RDD=(D1(I,1)*D1(I,1)+D1(I,3)*D1(I,3))*0.5
    IF (I.LE.KKN) WOUT=WOUT+RDD
    IF (I.EQ.KKO) DPR=RDD
    WRITE(11,220) I,(D1(I,J)*1000.0,J=1,3),XX(I),
      YY(I),ZZ(I),RDD*1000.0
  CONTINUE
  WRITE(6,*) 'IN OUTPUT'
  WOUT=WOUT/REAL(KKN)
  AREA=2.0*3.1415926535*RR*TO
  WRITE(6,*) 'IN OUTPUT', ' AREA=',AREA
  WRITE(9,*) WT*2000.0, ' ',WOUT*1000, ' ',TROOT/AREA
  IF (ICRP.EQ.1) THEN
    WRITE(3,*) WT*2.0/ZL, ' ',TROOT/AREA, ' ',CRPTM
  ELSE
    WRITE(3,*) WT*2.0/ZL, ' ',TROOT/AREA
  END IF
  FORMAT(115,6F10.6,1F12.3)
  WRITE(11,*) '*'
END IF
IF (NPT.EQ.5) THEN
  TEMP=0.0
  IF (NPV.EQ.1.OR.NPV.EQ.3) THEN
    DO 410 I=1,7
      TEMP=TEMP+D1(I,2)
    CONTINUE
    TEMP=TEMP*1000.0/7.0
    WRITE(9,425) (D1(I,2)*1000.0,I=1,7),TEMP,TROOT,
      TROOT*RR**3*10.92/198700.0/TO**3
    FORMAT(7F7.4,1F7.4,2F11.6)
    WRITE(11,*) TEMP, ' * ',TROOT,TROOT*RR**3*10.92/198700.0/TO**3
    WRITE(11,427) (((XX(I)**2+ZZ(I)**2)**0.5-RR)*1000.0,I=8,11)
    WRITE(11,426) (((XX(I)**2+ZZ(I)**2)**0.5-RR)*1000.0,I=12,18)
    WRITE(11,427) (((XX(I)**2+ZZ(I)**2)**0.5-RR)*1000.0,I=19,22)
    WRITE(11,426) (((XX(I)**2+ZZ(I)**2)**0.5-RR)*1000.0,I=23,29)
    WRITE(11,427) (((XX(I)**2+ZZ(I)**2)**0.5-RR)*1000.0,I=30,33)
    WRITE(11,426) (((XX(I)**2+ZZ(I)**2)**0.5-RR)*1000.0,I=34,40)
    IF (NPV.EQ.3) THEN
      WRITE(11,427) (((XX(I)**2+ZZ(I)**2)**0.5-RR)*1000.0,I=41,44)
      WRITE(11,426) (((XX(I)**2+ZZ(I)**2)**0.5-RR)*1000.0,I=45,51)
      WRITE(3,*) -((XX(48)**2+ZZ(48)**2)**0.5-RR)*1000.0,TROOT,
        TROOT*RR**3*10.92/198700.0/TO**3
    ELSE

```

```

      IF (ICRP.EQ.1) THEN
        WRITE (3,*) - ((XX (37) **2+ZZ (37) **2) **0.5-RR) *1000.0,TROOT,
          CRPTM
      ELSE
        WRITE (3,*) - ((XX (37) **2+ZZ (37) **2) **0.5-RR) *1000.0,TROOT,
          TROOT*RR**3*10.92/198700./TO**3
      END IF
    END IF
    WRITE (11,*)
    426   FORMAT (7F10.7)
    427   FORMAT (1F10.7,'          ',1F10.7,'          ',1F10.7,
      1    ',1F10.7)
    ELSE
      DO 411 I=1,9
        TEMP=TEMP+D1 (I,2)
    411   CONTINUE
        TEMP=TEMP*1000.0/9.0
        WRITE (9,426) (D1 (I,2)*1000.0,I=1,9)
        WRITE (9,432) TEMP,TROOT,
      1    TROOT*RR**3*10.92/198700.0/TO**3
    429   FORMAT (9F8.4)
    432   FORMAT (3F12.6)
        WRITE (11,*) TEMP,' * ',TROOT,TROOT*RR**3*10.92/198700./TO**3
        WRITE (11,424) (((XX (I) **2+ZZ (I) **2) **0.5-RR) *1000.0,I=10,14)
        WRITE (11,423) (((XX (I) **2+ZZ (I) **2) **0.5-RR) *1000.0,I=15,23)
        WRITE (11,424) (((XX (I) **2+ZZ (I) **2) **0.5-RR) *1000.0,I=24,28)
        WRITE (11,423) (((XX (I) **2+ZZ (I) **2) **0.5-RR) *1000.0,I=29,37)
        WRITE (11,424) (((XX (I) **2+ZZ (I) **2) **0.5-RR) *1000.0,I=38,42)
        WRITE (11,423) (((XX (I) **2+ZZ (I) **2) **0.5-RR) *1000.0,I=43,51)
        WRITE (11,*)
    423   FORMAT (9F8.5)
    424   FORMAT (5F10.7)
        WRITE (3,*) - ((XX (47) **2+ZZ (47) **2) **0.5-RR) *1000.0,TROOT,
      1    TROOT*RR**3*10.92/198700./TO**3
    END IF
  END IF
ELSE
  IF (NPT.EQ.1) THEN
    WRITE (3,*) ' ',D1 (I,2) /20.0*100.0,' ',TTLD/TO/20.0
  END IF
C
  IF (NPT.EQ.2) THEN
    DDK=3.14159**2*E*TO**3/12.0
    WRITE (3,*) ' ',D1 (I,3) /TO,' ',TTLD*2.0/DDK
  END IF
  IF (NPT.EQ.6) THEN
    TOR=0.0
    DO 600 I=1,NDBC,2
      TOR=TOR+RR*(-TTLY (I) *SIN (ANGL (I))+TTLY (I+1) *COS (ANGL (I+1)))
    600   CONTINUE
      WRITE (3,*) TROOT,' ',TOR
      DO 400 I=1,NNODE
        WRITE (11,*) I,' ',(XX (I) **2+ZZ (I) **2) **0.5-RR
    400   CONTINUE
      WRITE (11,*) '* '
    END IF
  END IF
  RETURN
  END
C
C   Subroutine UPDT is to update some variables when the
C   equilibrium requirement is satisfied.
C
  SUBROUTINE UPDT (ITYPE,IID,XX,YY,ZZ,DLDINC,D1,ACMDIS,XX1,
  1    YY1,ZZ1,DELTA,UPSIG,SIGMA,DLTINC,DLTAMP,
  2    BETA,UPBET,GCL1,GCL2,GCL3,UCL1,UCL2,UCL3,ANGL)

```



IMPLICIT REAL\*8 (A-H,O-Z)  
IMPLICIT INTEGER\*8 (I-N)

DIMENSION IID (NNODE,5)  
DIMENSION XX (1), YY (1), ZZ (1), D1 (NNODE,5), ACMDIS (1), XX1 (1),  
1 YY1 (1), ZZ1 (1), DELTA (1), UPSIG (NELM,2,2,2,9),  
2 SIGMA (NELM,2,2,2,9), DLTINC (1), DLTTMP (1),  
3 BETA (NELM,2,2,2,12), UPBET (NELM,2,2,2,12),  
4 GCL1 (NNODE,3), GCL2 (NNODE,3), GCL3 (NNODE,3),  
5 UCL1 (NNODE,3), UCL2 (NNODE,3), UCL3 (NNODE,3), ANGL (1)

COMMON /SCHALR1/ NELM,NNODE,NT  
COMMON /SCHALR2/ NEQT,NSTEP,NHBW,COEF1,COEF2,NSHOW1,NSHOW2,  
1 NSHOW3,HRZ,ITRLM,FACTOR  
COMMON /PNTRIN/ IP1,IP2,IP3,IP4,IP5,IP6,IP7,IP8,IP9,IP10  
COMMON /PNTRRL/ IR1,IR2,IR3,IR4,IR5,IR6,IR7,IR8,IR9,IR10,  
1 IR11,IR12,IR13,IR14,IR15,IR16,IR17,IR18,  
2 IR19,IR20,IR21,IR22,IR23,IR24,IR25,IR26,  
3 IR27,IR28,IR29,IR30,IR31,IR32,IR33,IR34,  
4 IR35,IR36,IR37,IR38,IR39,IR40,IR41,IR42,  
5 IR43,IR44,IR45,IR46,IR47,IR48,IR49,IR50  
COMMON /UNIFBD/ IR51,IR52,IR53,IR54,IR55,IR56,IR57,IR58,IR59  
COMMON /DIRCS/ IR60,IR61,IR62,IR63,IR64,IR65  
COMMON /DISCT/ NDC,NDBC  
COMMON /UNICT/ NCONS,MODEL,ETAA,TDELTA,TINIT  
COMMON /RLVEC/ VR (1)  
COMMON /INTVEC/ IPT (1)  
COMMON /ITESCH/ ROOT,DTLAM,SGN,IPP,TROOT,ASO,SP  
COMMON /GEO/ TO  
COMMON /OUTVR/ NPT,NPV  
COMMON /CONTN/ INSIDT,KPDT,DTLM1

C  
ND=NEQT  
DO 689 I=1,NNODE  
XX1 (I)=XX (I)  
YY1 (I)=YY (I)  
ZZ1 (I)=ZZ (I)  
DO 688 J=1,3  
UCL1 (I,J)=GCL1 (I,J)  
UCL2 (I,J)=GCL2 (I,J)  
UCL3 (I,J)=GCL3 (I,J)  
688 CONTINUE  
WRITE (6,691) I,XX1 (I),YY1 (I),ZZ1 (I)  
691 FORMAT (' COOR: ',1I3,3F10.6)  
689 CONTINUE

C  
DO 269 I=1,NELM  
DO 269 J=1,2  
DO 269 K=1,2  
DO 269 M=1,2  
DO 269 N=1,9  
SIGMA (I,J,K,M,N)=UPSIG (I,J,K,M,N)  
269 CONTINUE

C  
C  
IF (NCONS.EQ.1) THEN

C  
DO 169 I=1,NELM  
DO 169 J=1,2  
DO 169 K=1,2  
DO 169 M=1,2  
DO 169 N=1,12  
UPBET (I,J,K,M,N)=BETA (I,J,K,M,N)  
169 CONTINUE  
END IF  
IF (ITYPE.EQ.2) GOTO 800

```

C
DO 669 I=1,ND
  DLTTMP (I)=DELTA (I)
  ACMDIS (I)=ACMDIS (I)+DLTINC (I)
669 CONTINUE
C
K=1
DO 589 I=1,NNODE
  DO 589 J=1,5
    IF (IID (I,J) .EQ.0) THEN
      DI (I,J)=ACMDIS (K)
      K=K+1
    END IF
589 CONTINUE
C
IF (NPT.EQ.6) THEN
  DO 620 I=1,NDBC
    ANGL (I)=ANGL (I)+DTLM1
620 CONTINUE
  END IF
800 CONTINUE
  RETURN
  END

C
C Subroutine DISBN is used to calculate the displacement increment
C in displacement boundary value problem for cylindrical shells.
C
SUBROUTINE DISBN (ADVC, ANGL)
  IMPLICIT REAL*8 (A-H, O-Z)
  IMPLICIT INTEGER*8 (I-N)
  DIMENSION ADVC (1), ANGL (1)
  COMMON /DISCT/ NDC, NDBC
  COMMON /DISVC/ IR66, IR67, IR68, IR69
  COMMON /RLVEC/ VR (1)
  COMMON /INTVEC/ IPT (1)
  COMMON /OUTVR/ NPT, NPV
  COMMON /RADS/ RR, ZL
C
C NPT=1 : STRECH
C NPT=2 : PLATE
C NPT=3 : PANEL
C NPT=4 : CYLINDRICAL SHELL UNDER AXIAL COMPRESSION
C NPT=5 : CYLINDRICAL SHELL UNDER PRESSURE
C NPT=6 : CYLINDRICAL SHELL UNDER TORSION
C
IF (NPT.EQ.1.OR.NPT.EQ.2) THEN
  DO 10 I=1,NDBC
    ADVC (I)=1.0
10 CONTINUE
  END IF
IF (NPT.EQ.6) THEN
  WRITE (6,*) 'RR=', RR
  DO 30 I=1,NDBC
    WRITE (6,*) I, ' ANGLE', ANGL (I)
30 CONTINUE
  K=1
  DO 20 I=1,NDBC,2
    ADVC (K)=-RR*SIN (ANGL (I))
    ADVC (K+1)=RR*COS (ANGL (I))
C
    WRITE (6,*) 'ADVC1=', ADVC (K), ' ADVC2=', ADVC (K+1)
    K=K+2
20 CONTINUE
  END IF
  RETURN
  END

C
C Subroutine NTCRP is for the calculation of creep buckling.

```

C Newton-Raphson's iteration scheme is employed in the equilibrium  
C iterations.

C  
SUBROUTINE NTCRP (INUM, IEL, ID, IID, L, MAXA, LD, XX, YY, ZZ, DLOADT, D,  
1 PLD, FRCO, DD, DLDINC, VTEMP, VF, D1, VFE, DDD,  
2 AM, PD, P, A, TDLD, HISINC, ACMDIS, FRCINC,  
3 XX1, YY1, ZZ1, DELTA, UPSIG, SIGMA, DLTINC, DLTTMP,  
4 STIFFN, EXLVC, BETA, UPBET, ACTFRC, GCL1,  
5 GCL2, GCL3, UCL1, UCL2, UCL3, ADC, ADD, AD, ADVC, TLTY,  
6 TY1, TY2, ANGL, DBVC)  
IMPLICIT REAL\*8 (A-H, O-Z)  
IMPLICIT INTEGER\*8 (I-N)

C  
DIMENSION IEL (NELM, 8), ID (1), IID (NNODE, 5), L (1), MAXA (1), LD (1),  
1 XX (1), YY (1), ZZ (1), DD (NNODE, 5), D (1), PLD (1), DLOADT (1),  
2 DLDINC (1), VTEMP (1), VF (NNODE, 5), D1 (NNODE, 5), VFE (NT, 1),  
3 DDD (1), VRT (4), A (NEQT, NEQT), AM (40, 40), PD (1), P1 (1),  
4 TDLD (1), HISINC (1), ACMDIS (1), FRCINC (1), XX1 (1), YY1 (1),  
5 ZZ1 (1), DELTA (1), FRCO (1), UPSIG (NELM, 2, 2, 2, 9), ACTFRC (1),  
6 SIGMA (NELM, 2, 2, 2, 9), DLTINC (1), DLTTMP (1), COEEQ (5),  
7 DEFVRT (4), STIFFN (NT, NT), ETT (4), EXLVC (1), DBVC (1),  
8 BETA (NELM, 2, 2, 2, 12), UPBET (NELM, 2, 2, 2, 12), GCL1 (NNODE, 3),  
9 GCL2 (NNODE, 3), GCL3 (NNODE, 3), UCL1 (NNODE, 3), UCL2 (NNODE, 3),  
1 UCL3 (NNODE, 3), ADC (NDBC, NDBC), ADD (NDBC, NEQT),  
2 AD (NEQT, NDBC), ADVC (1), TLTY (1), TY1 (1), TY2 (1), ANGL (1)

C  
COMMON /SCHALR1/ NELM, NNODE, NT  
COMMON /SCHALR2/ NEQT, NSTEP, NHBW, COEF1, COEF2, NSHOW1, NSHOW2,  
1 NSHOW3, HRZ, ITRLM, FACTOR  
COMMON /RLVEC/ VR (1)  
COMMON /INTVEC/ IPT (1)  
COMMON /PNTRIN/ IP1, IP2, IP3, IP4, IP5, IP6, IP7, IP8, IP9, IP10  
COMMON /PNTRRL/ IR1, IR2, IR3, IR4, IR5, IR6, IR7, IR8, IR9, IR10,  
1 IR11, IR12, IR13, IR14, IR15, IR16, IR17, IR18,  
2 IR19, IR20, IR21, IR22, IR23, IR24, IR25, IR26,  
3 IR27, IR28, IR29, IR30, IR31, IR32, IR33, IR34,  
4 IR35, IR36, IR37, IR38, IR39, IR40, IR41, IR42,  
5 IR43, IR44, IR45, IR46, IR47, IR48, IR49, IR50  
COMMON /UNIFBD/ IR51, IR52, IR53, IR54, IR55, IR56, IR57, IR58, IR59  
COMMON /DIRCS/ IR60, IR61, IR62, IR63, IR64, IR65  
COMMON /DISVC/ IR66, IR67, IR68, IR69  
COMMON /DISV1/ IR70, IR71, IR72, IR73, IR74, IR75  
COMMON /DISCT/ NDC, NDBC  
COMMON /UNICT/ NCONS, MODEL, ETAA, TDELTA, TINIT  
COMMON /ITESCH/ ROOT, DTLAM, SGN, IPP, TROOT, ASO, SP  
COMMON /GEO/ TO  
COMMON /CNTRL/ DETMNT  
COMMON /CONTN/ INSIDT, KPDT, DTLM1  
COMMON /ABDFST/ ISEC  
COMMON /MTL/ E, EU  
COMMON /SQ/ SQQ  
COMMON /BRLIM/ LIM  
COMMON /NMBITR/ NUM  
COMMON /OUTVR/ NPT, NPV  
COMMON /CRPC/ CRPC1, CRPC2  
COMMON /CREEP/ ICRP, NBCRP, NBDN, CRPTM, IPON  
COMMON /CNTR/ ICNTR  
COMMON /TMPCO/ ICTMP

C  
C  
C  
ICTMP=0  
C (The switch to the effects of the change of temperature is off)  
ICNTR=ICNTR+1  
RTL=0.0  
LIM=0  
VLS1=0.0

```

VLS2=0.0
CALL INIT (VR (IR1), VR (IR2), VR (IR3), VR (IR43), VR (IR44), VR (IR45),
1      VR (IR60), VR (IR61), VR (IR62), VR (IR63), VR (IR64), VR (IR65),
2      VR (IR47), VR (IR20), VR (IR51), VR (IR58))
C
WRITE (6,*) 'NUMBER:', INUM
ND=NEQT
IF (ICRP.EQ.1) THEN
  NBDN=NBDN+1
END IF
C
Begin iteration
C
III=1
C
CALL MNU (NNOE,5,VF)
DO 200 I=1,NT
  DLDINC (I)=DLOADT (I)
200 CONTINUE
DO 195 I=1,ND
  TDLD (I)=0.0
  HISINC (I)=0.0
195 CONTINUE
C
210 FORMAT ('I,LDINC,LOADT,PLD IS',113,3F8.3)
C
579 CONTINUE
C
Form the global stiffness matrix.
C
CALL ASSMBL (III, IPT (IP1), IPT (IP2), IPT (IP3), IPT (IP4), IPT (IP5),
1      IPT (IP9), VR (IR1), VR (IR2), VR (IR3), VR (IR6), VR (IR8),
2      VR (IR12), VR (IR14),
3      VR (IR15), VR (IR16), VR (IR19),
4      VR (IR21), VR (IR23), VR (IR24), VR (IR19), VR (IR41), VR (IR50),
5      VR (IR52), VR (IR66), VR (IR67), VR (IR68), VR (IR74))
C
ICDD=1
WRITE (6,*) 'ASSMBL CALLED'
IF (III.GT.2) GOTO 577
IF (NDC.EQ.1) THEN
  CALL DISBN (VR (IR69), VR (IR75))
  DO 570 I=1,ND
    WRITE (6,*) I, (AD (I,K), K=1,NDBC)
    DDD (I)=0.0
    DO 570 J=1,NDBC
      DDD (I)=DDD (I)+AD (I,J)*ADVC (J)
570 CONTINUE
533 FORMAT (113,6F9.3)
DO 572 I=1,ND
  DDD (I)=D (I)-DDD (I)
572 CONTINUE
END IF
IF (NDC.EQ.0) THEN
  DO 573 I=1,ND
    DDD (I)=D (I)
573 CONTINUE
END IF
16 FORMAT ('D (I) AND DDD (I): ',113,2F14.5)
C
577 CONTINUE
WRITE (6,36) III
36 FORMAT ('THIS IS THE ITERATION ',113)
IF (III.EQ.ITRLM) THEN
  WRITE (6,*) 'ITERATION LIMIT REACHED. STOP.'

```

```

      STOP
      END IF
C
      IF (III.EQ.1) THEN
C
        DO 755 I=1,ND
          VTEMP (I)=0.0
          DO 755 J=1,ND
            VTEMP (I)=VTEMP (I)+STIFFN (I,J)*TDLD (J)
C 755 CONTINUE
C
          ASL=0.0
          DO 857 I=1,ND
            ASL=ASL+VTEMP (I)*TDLD (I)
C 857 WRITE (6,*) I, ' TDLD=',TDLD (I)
            CONTINUE
            WRITE (6,*) 'ASL ',ASL
C
            WRITE (6,*) 'TDELT=',TDELT
            WRITE (6,*) 'DETMNT=',DETMNT
            IF (ASL.LT.0.0) THEN
              WRITE (6,*) 'CHANGED SIGN OF FAC'
            END IF
            IF (DETMNT.LT.0.0) WRITE (6,*) 'NEGATIVE DETERMINT'
            DO 550 I=1,ND
              DLTTMP (I)=0.0
              DELTA (I)=0.0
              VTEMP (I)=0.0
              FRCINC (I)=0.0
C 550 CONTINUE
            END IF
C
            WRITE (6,*) 'III=',III
C
C 625 CONTINUE
            DO 635 I=1,ND
              DLTINC (I)=0.0
              DO 634 J=1,ND
                DLTINC (I)=DLTINC (I)+A (I,J)*EXLVC (J)
C 634 CONTINUE
                IF (III.GT.1) DLTINC (I)=DLTINC (I)*CRPC1
                DELTA (I)=DLTTMP (I)+DLTINC (I)
C 635 CONTINUE
C
            IF (III.EQ.1) THEN
              WRITE (6,*) 'FIRST ITERATION OF STEP ',NUM
            END IF
            I=NEQT
            WRITE (6,*) 'CURRENT ROOT ',ROOT
            WRITE (6,*) 'TDLD (25) ',TDLD (I)
            WRITE (6,*) I, ' ROOT*TDLD ',ROOT*TDLD (I)
            WRITE (6,*) I, ' FRCINC ',FRCINC (I)
            WRITE (6,*) I, ' HISINC ',HISINC (I)
            WRITE (6,*) I, ' DLTINC ',DLTINC (I)
            WRITE (6,*) I, ' DELTA ',DELTA (I)
C
C
            K=1
            KK=1
            DO 580 I=1,NNODE
              DO 580 J=1,5
                IF (IID (I,J).EQ.0) THEN
                  VF (I,J)=DLTINC (K)
                  DD (I,J)=DLTINC (K)
                  K=K+1
                END IF

```

```

580 CONTINUE

      DO 901 I=1,NNODE
        DO 901 J=1,5
          VFE (I*5-5+J,1)=VF (I,J)
901 CONTINUE
302 FORMAT ('I,VFE (I) IS: ',2I2,1F12.6)

      TINC=1.0
      DO 900 I=1,NNODE
        XX (I)=XX (I)+DD (I,1)
        YY (I)=YY (I)+DD (I,2)
        ZZ (I)=ZZ (I)+DD (I,3)
        TMP=0.0
        DO 903 J=1,3
          GCL3 (I,J)=GCL3 (I,J)+TINC*(-GCL2 (I,J)*DD (I,4)+GCL1 (I,J)*DD (I,5))
          TMP=TMP+GCL3 (I,J)*GCL3 (I,J)
903 CONTINUE
          TMP=TMP**0.5
          DO 902 J=1,3
            GCL3 (I,J)=GCL3 (I,J)/TMP
902 CONTINUE
          WRITE (6,267) I,XX (I),YY (I),ZZ (I)
900 CONTINUE

      C
      C Calculate new directional cosines for all the nodes of elements.
      C
      CALL CNND (VR (IR60),VR (IR61),VR (IR62))

      C
      C Calculate internal forces
      C
      CALL INTFRC (III,IPT (IP1),VR (IR1),VR (IR2),VR (IR3),
1          VR (IR14),VR (IR22),VR (IR28),VR (IR9))

      C
      C
      DO 500 I=1,NT
        DO 500 M=1,ND
          IF (I.EQ.L (M)) THEN
            FRCINC (M)=(PLD (I)-FRCO (M))
            ACTFRC (M)=PLD (I)
          C WRITE (6,*) M,' PLD=',PLD (I),' FCO=',FRCO (M),' FIC=',FRCINC (M)
          END IF
500 CONTINUE

      C
      DO 549 I=1,ND
        EXLVC (I)=-FRCINC (I)
          C WRITE (6,*) M,' FCO=',FRCO (I),' FIC=',FRCINC (I)
          C 1 , 'ACTF=',ACTFRC (I)
549 CONTINUE

      C
      ISWTC=0
      ISEC=ISEC+1
      IF (ISEC.GT.10) ISEC=10

      C
      DO 665 I=1,ND
        DLTTMP (I)=DELTA (I)
          C WRITE (6,*) 'DELTA AFTER ',DELTA (I)
          C ACMDIS (I)=ACMDIS (I)+DLTTMP (I)
          C WRITE (6,*) I,' ACMDIS ',ACMDIS (I)
665 CONTINUE

      C
      C
      K=1
      DO 585 I=1,NNODE

```

```

DO 585 J=1,5
  IF (IID(I,J).EQ.0) THEN
    D1(I,J)=ACMDIS(K)
    K=K+1
  END IF
585 CONTINUE

```

```

C Check whether equilibrium requirement is satisfied.
C

```

```

C CALL CRITR3(III,ND,VR(IR8),VR(IR42),VR(IR59),VR(IR17),
1 VLIMIT,ICNC1,VALS)
C WRITE(6,*) 'VLIMIT=',VLIMIT
C IF (ICNC1.EQ.0) THEN
C   IF (III.EQ.1) VLS1=VALS
C   IF (III.EQ.2) VLS2=VALS
C   IF (III.GT.2) THEN
C     IF (VALS.GT.VLS1.AND.VALS.GT.VLS2) THEN
C       WRITE(6,*) 'BREAK=',LIM
C       DTLM1=DTLM1/2.0
C       LIM=LIM+1
C       IF (LIM.EQ.20) THEN
C         WRITE(6,*) 'Break limit reached, stop'
C         STOP
C       END IF
C       GOTO 1000
C     ELSE
C       VLS1=VLS2
C       VLS2=VALS
C       LIM=0
C     END IF
C   END IF
C END IF
C END IF

```

```

C IF ((ICONCL.EQ.1).OR.(ICNC1.EQ.1)) THEN
C   IF (III.LT.3.AND.NUM.LT.24) DTLM1=DTLM1*SQQ
C   DTLM1=DTLM1*SQQ
C   IF (III.GE.8.AND.III.LT.10) DTLM1=DTLM1/1.1
C   IF (III.GE.10.AND.III.LT.15) DTLM1=DTLM1/1.2
C   IF (III.GE.15) DTLM1=DTLM1/1.0
C   WRITE(6,*) 'FIN VAL OF III=',III,' NDTLM1=',DTLM1
C   CRPTM=CRPTM+TDELT

```

```

C Write output data
C

```

```

C CALL OUTPUT(TTLD,VR(IR15),VR(IR75),VR(IR71),VR(IR1),VR(IR2),
1 VR(IR3))

```

```

C ITYPE=1

```

```

C Update some variables.

```

```

C CALL UPDT(ITYPE,IPT(IP3),VR(IR1),VR(IR2),VR(IR3),VR(IR12),
1 VR(IR15),VR(IR27),VR(IR43),VR(IR44),VR(IR45),
2 VR(IR46),VR(IR47),VR(IR20),VR(IR48),VR(IR49),
3 VR(IR51),VR(IR58),VR(IR60),VR(IR61),VR(IR62),
4 VR(IR63),VR(IR64),VR(IR65),VR(IR75))

```

```

C ELSE

```

```

  III=III+1

```

```

  ICDD=ICDD+1

```

```

  GOTO 577

```

```

END IF

```

```

670 CONTINUE

```

```

C DO 555 I=1,ND

```

```

  DO 555 J=1,ND

```

```

    VTEMP(I)=VTEMP(I)+STIFFN(I,J)*DELTA(J)

```

```

555 CONTINUE

```

```

C
ASLOP=0.0
DO 557 I=1,ND
  ASLOP=ASLOP+VTEMP(I)*DELTA(I)
557 CONTINUE
ASLOP=ASLOP/ABS(ASLOP)
C
IF (KPDT.EQ.NUM) THEN
  CALL WTCDDT (VR (IR27) ,VR (IR20) ,VR (IR43) ,VR (IR44) ,
1          VR (IR45) ,VR (IR1) ,VR (IR2) ,VR (IR3) ,
1          VR (IR47) ,VR (IR10) ,VR (IR51) ,VR (IR58) ,VR (IR60) ,
3          VR (IR61) ,VR (IR62) ,VR (IR15) ,VR (IR71) ,VR (IR75))
  END IF
1000 CONTINUE
RETURN
END
C
C Subroutine Init is used to initiate some variables
C
SUBROUTINE INIT (XX,YY,ZZ,XX1,YY1,ZZ1,GCL1,GCL2,GCL3,
1          UCL1,UCL2,UCL3,UPSIG,SIGMA,BETA,UPBET)
IMPLICIT REAL*8 (A-H,O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION XX (1) ,YY (1) ,ZZ (1) ,XX1 (1) ,YY1 (1) ,ZZ1 (1) ,
1          UPSIG (NELM,2,2,2,9) ,SIGMA (NELM,2,2,2,9) ,
2          BETA (NELM,2,2,2,12) ,UPBET (NELM,2,2,2,12) ,
3          GCL1 (NNODE,3) ,GCL2 (NNODE,3) ,GCL3 (NNODE,3) ,
4          UCL1 (NNODE,3) ,UCL2 (NNODE,3) ,UCL3 (NNODE,3)
C
COMMON /SCHALR1/ NELM,NNODE,NT
C
DO 687 I=1,NNODE
  XX (I) =XX1 (I)
  YY (I) =YY1 (I)
  ZZ (I) =ZZ1 (I)
  DO 686 J=1,3
    GCL1 (I,J) =UCL1 (I,J)
    GCL2 (I,J) =UCL2 (I,J)
    GCL3 (I,J) =UCL3 (I,J)
686 CONTINUE
687 CONTINUE
DO 249 I=1,NELM
  DO 249 J=1,2
    DO 249 K=1,2
      DO 249 M=1,2
        DO 249 N=1,9
          UPSIG (I,J,K,M,N) =SIGMA (I,J,K,M,N)
249 CONTINUE
DO 164 I=1,NELM
  DO 164 J=1,2
    DO 164 K=1,2
      DO 164 M=1,2
        DO 164 N=1,12
          BETA (I,J,K,M,N) =UPBET (I,J,K,M,N)
164 CONTINUE
C
RETURN
END
C
C Subroutine REDC eliminates the redundant elements of a vector.
C
SUBROUTINE REDC (L,D,DLDDINC)
IMPLICIT REAL*8 (A-H,O-Z)
IMPLICIT INTEGER*8 (I-N)
DIMENSION L (1) ,D (1) ,DLDDINC (1)
COMMON /SCHALR1/ NELM,NNODE,NT

```



```
C
  DO 500 I=1,NT
    DO 500 M=1,IDF
      IF (I.EQ.L(M)) THEN
        D(M)=DLDINC(I)
      END IF
500 CONTINUE
C
  RETURN
  END
C
  (END REDC)
C
```

# 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.

<b>1. AGENCY USE ONLY</b> ( <i>Leave blank</i> )	<b>2. REPORT DATE</b> December 1991	<b>3. REPORT TYPE AND DATES COVERED</b> Final Contractor Report	
<b>4. TITLE AND SUBTITLE</b>  A Finite Element Program for Postbuckling Calculations (PSTBKL)		<b>5. FUNDING NUMBERS</b>  WU-553-13-00 G-NAG3-534	
<b>6. AUTHOR(S)</b>  G.T. Simitzes, R.L. Carlson, and R. Riff		<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>  None	
<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b>  Georgia Institute of Technology School of Civil Engineering Atlanta, Georgia 30332		<b>10. SPONSORING/MONITORING AGENCY REPORT NUMBER</b>  NASA CR-189091	
<b>9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)</b>  National Aeronautics and Space Administration Lewis Research Center Cleveland, Ohio 44135-3191		<b>11. SUPPLEMENTARY NOTES</b>  Project Manager, C.C. Chamis, Structures Division, NASA Lewis Research Center, (216) 433-3252.	
<b>12a. DISTRIBUTION/AVAILABILITY STATEMENT</b>  Unclassified - Unlimited Subject Category 39		<b>12b. DISTRIBUTION CODE</b>	
<b>13. ABSTRACT</b> ( <i>Maximum 200 words</i> )  The object of the research reported herein was to develop a general mathematical model and solution methodologies for analyzing the structural response of thin, metallic shell structures under large transient, cyclic, or static thermomechanical loads. Among the system responses associated with these loads and conditions are thermal buckling, creep buckling, and ratcheting. Thus geometric and material nonlinearities (of high order) can be anticipated and must be considered in developing the mathematical model. The methodology is demonstrated through different problems of extension, shear and of planar curved beam. Moreover, importance of inclusion of large strains is clearly demonstrated, through the chosen applications. This report describes the computer program resulting from the research.			
<b>14. SUBJECT TERMS</b>  Rate theory; Kinematics; Kinetics; Large strains; Viscoplasticity; Finite rotations; Thermodynamic state; Example problems			<b>15. NUMBER OF PAGES</b> 106
<b>17. SECURITY CLASSIFICATION OF REPORT</b> Unclassified			<b>16. PRICE CODE</b> A06
<b>18. SECURITY CLASSIFICATION OF THIS PAGE</b> Unclassified	<b>19. SECURITY CLASSIFICATION OF ABSTRACT</b> Unclassified	<b>20. LIMITATION OF ABSTRACT</b>	