

```

1
2  *deck,usermat3d      USERDISTRIB  parallel                                gal
3      subroutine usermat3d_EPVP (
4          &                matId, elemId,kDomIntPt, kLayer, kSectPt,
5          &                ldstep,isubst,keycut,
6          &                nDirect,nShear,ncomp,nStatev,nProp,
7          &                Time,dTime,Temp,dTemp,
8          &                stress,ustatev,dsdePl,sedEl,sedPl,epseq,
9          &                Strain,dStrain, epsPl, prop, coords,
10         &                var0, defGrad_t, defGrad,
11         &                tsstif, epsZZ, cutFactor,
12         &                var1, var2, var3, var4, var5,
13         &                var6, var7)
14 #include "impcom.inc"
15     !*****!
16     !! SUBROTINA: USERMAT3D_EPVP                                !!
17     !!                                                                !!
18     !! Objetivo: atualiza as tensões, variáveis de estado e matriz    !!
19     !!                constitutiva para o modelo constitutivo EPVP    !!
20     !!                                                                !!
21     !! Situação: OK (10/11/2021)                                       !!
22     !!                                                                !!
23     !*****!
24     !
25     !*****!
26     ! Declaração variáveis de entrada e saída da subrotina          !
27     !*****!
28     INTEGER
29     &                matId, elemId,
30     &                kDomIntPt, kLayer, kSectPt,
31     &                ldstep,isubst,keycut,
32     &                nDirect,nShear,ncomp,nStatev,nProp
33     DOUBLE PRECISION
34     &                Time,      dTime,      Temp,      dTemp,
35     &                sedEl,      sedPl,      epseq,      epsZZ,      cutFactor
36     DOUBLE PRECISION
37     &                stress  (ncomp ), ustatev (nStatev),
38     &                dsdePl  (ncomp,ncomp),
39     &                Strain  (ncomp ), dStrain (ncomp ),
40     &                epsPl   (ncomp ), prop    (nProp ),
41     &                coords  (3),
42     &                defGrad (3,3),      defGrad_t(3,3),
43     &                tsstif  (2)
44     DOUBLE PRECISION var0, var1, var2, var3, var4, var5,
45     &                var6, var7
46     !
47     !*****!
48     ! Informações sobre as variáveis locais (precisão, tipo, escopo) !
49     !*****!
50     c  ! aux1                (dp,sc,l)                denominador no dlambdVP
51     c  ! c                   (dp,sc,l)                coesão
52     c  ! c1,c2,c3            (dp,sc,l)                cs do vetor de fluxo
53     c  ! cp,ci,cr            (dp,sc,l)                coesão inicial, pico e residual
54     c  ! czao                (dp,sc,l)                constante para def. plast. equiv.
55     c  ! czaoVP              (dp,sc,l)                constante para def. viscop. equiv.
56     c  ! cVP                 (dp,sc,l)                coesão do modelo viscoplastico
57     c  ! ddlam               (dp,sc,l)                ddlam pelo NR do corretor plástico
58     c  ! dfds                (dp,ar(ncomp),l)          df/dsigma
59     c  ! dfds_m              (dp,ar(1,ncomp),l)        df/dsigma^T para aplicar MATMUL
60     c  ! dgds                (dp,ar(ncomp),l)          dg/dsigma
61     c  ! dgdsVP              (dp,ar(ncomp),l)          dg/dsigma viscoplastico
62     c  ! dgds_m              (dp,ar(ncomp,1),l)        dg/dsigma para aplicar MATMUL
63     c  ! dgdsVP_m            (dp,ar(ncomp,1),l)        dg/dsigmaVP para aplicar MATMUL
64     c  ! dhdq,dfdq,dfdc,dcde (dp,sc,l)                dh/dq, df/dq, df/dc, dc/de
65     c  ! dgPHitD              (dp,ar(ncomp,ncomp),l)    produto dg/dsigma*PHI^T*D
66     c  ! dlam                (dp,sc,l)                delta lambda elastoplastico
67     c  ! dlambdVP             (dp,sc,l)                delta lambda viscoplastico
68     c  ! depsEP               (dp,ar(ncomp),l)          incremento deformações plásticas
69     c  ! depsVP               (dp,ar(ncomp),l)          incremento deformações viscoplásticas
70     c  ! depsEPeq             (dp,sc,l)                incremen. defor. plástica equivalente
71     c  ! DgftD                (dp,ar(ncomp,ncomp),l)    produto D*dg/dsigma*df/dsigma^T*D
72     c  ! dsdeEl               (dp,ar(ncomp,ncomp),l)    matriz constitutiva elastica
73     c  ! dsdeElinv            (dp,ar(ncomp,ncomp),l)    inversa de dsdeEl

```

```

74 c ! dt1,dt2 (dp,sc,l) tempos limites do modelo viscoplastico
75 c ! eps1,eps2,eps3 (dp,sc,l) deformações plast. equiva. amol/endur.
76 c ! epsEP (dp,ar(ncomp),l) deformações plásticas
77 c ! epsVP (dp,ar(ncomp),l) deformações viscoplasticas
78 c ! epsEPeq (dp,sc,l) deformação plástica equivalente
79 c ! epsVPeq (dp,sc,l) deformação viscoplastica equivalente
80 c ! f (dp,sc,l) função de escoamento elastoplástico
81 c ! fVP (dp,sc,l) função de escoamento viscoplastico
82 c ! fi (dp,sc,l) ângulo de atrito elastoplástico
83 c ! fiVP (dp,sc,l) ângulo de atrito viscoplastico
84 c ! ftDg (dp,sc,l) produto df/dsigma^T*D*dg/dsigma
85 c ! g1,g2,g3 (dp,ar(ncomp),l) componentes diretoras do vetor de fluxo
86 c ! i,j,k (int,sc,l) contadores
87 c ! I1 (dp,sc,l) primeiro invariante das tensões
88 c ! J2 (dp,sc,l) segundo invariante do desviador
89 c ! J3 (dp,sc,l) terceiro invariante do desviador
90 c ! n,eta,f0 (dp,sc,l) constantes do modelo de Perzyna
91 c ! ncompgt (int,sc,l) componentes de tensões iniciais
92 c ! nrmax (int,sc,l) quantidade máxima de interações de NR
93 c ! PHI (dp,sc,l) função de sobretensão
94 c ! PHItDdeps (dp,sc,l) produto PHI^T*D*deps
95 c ! PHItDg (dp,sc,l) produto dPHI/ds^T*D*dg/ds
96 c ! psi (dp,sc,l) angulo de dilatação
97 c ! psiVP (dp,sc,l) angulo de dilatação viscoplastico
98 c ! s (dp,ar(ncomp),l) tensor desviador
99 c ! sigi (dp,ar(ncomp),l) tensão inicial
100 c ! sigmap (dp,ar(ncomp),l) tensão inicial a descontar
101 c ! stressn (dp,ar(ncomp),l) tensão no inicio do passo
102 c ! stresstrial (dp,ar(ncomp),l) tensão tentativa
103 c ! theta (dp,sc,l) ângulo de Lode
104 c ! thetaVP (dp,sc,l) ângulo de Lode do modelo viscoplastico
105 c ! young (dp,sc,l) módulo de Young
106 c ! posn (dp,sc,l) coeficiente de Poisson
107 c ! q1,dq (dp,sc,l) variaveis auxiliares
108 c ! superficief (int,sc,l) f : 1-DPI, 2-DPII, 3-DPIII
109 c ! superficiefVP (int,sc,l) fVP : 1-DPI, 2-DPII, 3-DPIII
110 c ! superficiegVP (int,sc,l) g : 1-DPI, 2-DPII, 3-DPIII
111 c ! superficieg (int,sc,l) gVP : 1-DPI, 2-DPII, 3-DPIII
112 c ! vPi (dp,sc,l) Pi
113 c !
114 c ! *****!
115 c ! INFORMAÇÕES DAS VARIÁVEIS DE ESTADO !
116 c ! *****!
117 c !
118 c ! nstatev = 20
119 c ! ustatev(1) = epsEPeq
120 c ! ustatev(2) = dlam
121 c ! ustatev(3) = c
122 c ! ustatev(4) = q
123 c ! ustatev(5) = f
124 c ! ustatev(6) = epsVPeq
125 c ! ustatev(7) = dt1
126 c ! ustatev(8) = dt2
127 c ! ustatev(9:ncomp) = espEP(1:ncomp)
128 c ! ustatev(15:ncomp) = espVP(1:ncomp)
129 c !
130 c ! OBS: as variáveis de estados podem ser plotadas na GUI do ANSYS
131 c ! utilizando o comando PLESOL,SVAR,[componente] ou PLNESOL,SVAR,[componente]
132 c !
133 c ! *****!
134 c ! Declaração das variáveis locais !
135 c ! *****!
136 c !
137 c ! Variáveis comuns a ambos modelos
138 c INTEGER
139 c & i, j, k, ncompgt
140 c DOUBLE PRECISION
141 c & I1,J2,J3,theta,
142 c & c1,c2,c3,
143 c & vPi,
144 c & dstress(ncomp),sigi(ncomp),
145 c & dsdeEl(ncomp,ncomp),
146 c & s(ncomp),

```

```

147      &          g1(ncomp),g2(ncomp),g3(ncomp)
148      DOUBLE PRECISION
149      &          young,posn
150      PARAMETER      (
151      &          vPi = 3.14159265358979323846d0
152      &          )
153      EXTERNAL
154      &          vzero, vmove, get_ElmData, get_ElmInfo,
155      &          matrizD,invars,normatensor,calcula_czao,
156      &          matinv,yield,clc2c3,glg2g3
157  c      !
158  c      ! variáveis do modelo viscoplástico
159      DOUBLE PRECISION
160      &          fVP,PHI,
161      &          dlambdVP,PHItDdeps,PHItDg,aux1,
162      &          epsVPeq,depsVPeq,czaoVP
163      DOUBLE PRECISION
164      &          dgdsVP(ncomp),dPHIds(ncomp),
165      &          dPHIds_m(1,ncomp),dgdsVP_m(ncomp,1),
166      &          epsVP(ncomp),depsVP(ncomp),
167      &          dgPHItD(ncomp,ncomp),
168      &          sigmap(ncomp),dt1,dt2
169      INTEGER
170      &          superficiefVP,superficieiegVP
171      DOUBLE PRECISION
172      &          cVP,fiVP,psiVP,n,eta,f0,thetaVP
173  c      !
174  c      ! variáveis do modelo elastoplástico
175      INTEGER
176      &          nrmax, dalg
177      DOUBLE PRECISION
178      &          tolEP,
179      &          f,
180      &          dlam,ddlam,ftDg,
181      &          dhdq,dfdq,dqdc,dcde,
182      &          epsEPeq,depsEPeq,czao,
183      &          q,dq,Xi
184      DOUBLE PRECISION
185      &          dsdeElinv(ncomp,ncomp),
186      &          stresstrial(ncomp),
187      &          dgds(ncomp),dfds(ncomp),
188      &          dfds_m(1,ncomp),dgds_m(ncomp,1),
189      &          epsEP(ncomp),depsEP(ncomp),
190      &          DgftD(ncomp,ncomp),
191      &          stressn(ncomp)
192      INTEGER
193      &          superficief,superficieieg
194      DOUBLE PRECISION
195      &          c,fi,psi,ci,cp,cr,eps1,eps2,eps3
196      PARAMETER      (nrmax = 100000,
197      &          tolEP = 0.00000000000001d0
198      &          )
199      EXTERNAL
200      &          calcula_Xi,calcula_dcde
201  c      !
202  c      ! *****!
203  c      ! Entrada de dados!
204  c      ! *****!
205  c      !
206  c      ! Variavel que controla o método da bisseção caso não haja convergência
207      keycut      = 0
208  c      !
209  c      ! Propriedades elásticas
210      young      = prop(2)
211      posn      = prop(3)
212  c      !
213  c      ! Propriedades do modelo EP
214      superficief = prop(4)
215      superficieieg = prop(5)
216      fi      = prop(6)*vPi/180
217      psi      = prop(7)*vPi/180
218      ci      = prop(8)
219      cp      = prop(9)

```

```

220      cr                = prop(10)
221      eps1              = prop(11)
222      eps2              = prop(12)
223      eps3              = prop(13)
224      dalg              = prop(14)
225  c      !
226  c      ! Propriedades do modelo VP
227      superficieVP      = prop(15)
228      superficieVP      = prop(16)
229      fiVP              = prop(17)*vPi/180
230      psiVP             = prop(18)*vPi/180
231      cVP               = prop(19)
232      n                 = prop(20)
233      eta               = prop(21)
234      f0                = prop(22)
235      thetaVP          = prop(23)
236  c      !
237  c      ! Limpando variáveis do modelo EP
238      depsEP            = 0.0d0
239      depsEPeq          = 0.0d0
240      dq                = 0.0d0
241      f                 = 0.0d0
242      stressn           = 0.0d0
243      dcde              = 0.0d0
244      k                 = 0
245  c      !
246  c      ! Limpando variáveis do modelo VP
247      depsVP            = 0.0d0
248      fVP               = 0.0d0
249      dt1               = 0.0d0
250      dt2               = 0.0d0
251  c      !
252  c      !*****!
253  c      ! Coletando variáveis de estado e deformações plásticas do passo convergido!
254  c      !*****!
255  c      ! Modelo EP
256      epsEPeq = ustatev(1)
257      dlam    = ustatev(2)
258      c       = ustatev(3)
259      q       = ustatev(5)
260      CALL vmove(ustatev(9), epsEP(1), ncomp)
261  c      !
262  c      ! Inicializa o valor da coesão
263      IF(c.EQ.0.0d0)c=ci
264      IF(q.EQ.0.0d0)THEN
265          CALL calcula_Xi(superficieVP,fi,Xi)
266          q=Xi*ci
267      ENDIF
268  c      !
269  c      ! Modelo VP
270      epsVPeq = ustatev(6)
271      CALL vmove(ustatev(15), epsVP(1), ncomp)
272  c      !
273  c      !*****!
274  c      ! Calculo da matriz constitutiva !
275  c      !*****!
276      dsdeEl      = 0.0d0
277      CALL MatrizD(young,posn,ncomp,dsdeEl)
278      dsdePl = dsdeEl
279  c      !
280  c      !*****!
281  c      ! Calculo módulo de rigidez transversal para hourglass !
282  c      !*****!
283      tsstif(1) = 0.5d0*(young/(1.0d0+posn))
284  c      !
285  c      !*****!
286  c      ! Coletando tensões iniciais !
287  c      !*****!
288      call get_ElmInfo('NCOMP', ncompgt)
289      call vzero(sigi(1),ncompgt)
290      call get_ElmData ('ISIG', elemId,kDomIntPt, ncompgt, sigi)
291  c      !
292  c      !*****!

```

```

293 c      ! Calculo da tensão no passo n
294 c      !*****!
295 stress = MATMUL(dsdeEl(1:ncomp,1:ncomp),
296 &          strain(1:ncomp)
297 &          -epsVP(1:ncomp)
298 &          -epsEP(1:ncomp))+sigi
299 c      !
300 c      !*****!
301 c      ! Calculo da função de sobretensão
302 c      !*****!
303 PHI = 0.0d0
304 CALL invars(stress,ncomp,I1,J2,J3,theta,s)
305 CALL yield(superficiefVP,I1,J2,theta,cVP,fiVP,fVP)
306 PHI = (fVP/f0)**n
307 IF(PHI.LE.0.0d0) PHI = 0.0d0
308 c      !
309 c      !*****!
310 c      ! Calculo dgdsvp
311 c      !*****!
312 CALL glg2g3(s,ncomp,J2,g1,g2,g3)
313 CALL clc2c3(J2,theta,superficieVP,psiVP,c1,c2,c3)
314 dgdsVP = c1*g1 + c2*g2 + c3*g3
315 c      !
316 c      !*****!
317 c      ! Calculo dPHIds,PHItDg,PHItDdeps,aux1
318 c      !*****!
319 CALL clc2c3(J2,theta,superficiefVP,psiVP,c1,c2,c3)
320 dPHIds = c1*g1 + c2*g2 + c3*g3
321 IF(PHI.LE.0.0d0) dPHIds = 0.0d0
322 c      !
323 c      !*****!
324 c      ! Verificação do incremento de tempo
325 c      !*****!
326 IF(PHI.GT.0.0d0) THEN
327 dt1 = 4.0d0/3.0d0*eta/PHI*(1.0d0+posn)/young*DSQRT(3.0d0*J2)
328 dt2 = eta*f0/(n*(fVP/f0)**(n-1))*
329 &      (1.0d0+posn)*(1.0d0-2.0d0*posn)/young*
330 &      ((3.0d0-DSIN(fiVP))**2)/
331 &      (3.0d0/4.0d0*(1.0d0-2.0d0*posn)*(3.0d0-DSIN(fiVP))**2+
332 &      6.0d0*(1.0d0+posn)*DSIN(fiVP)**2)
333 IF(dtime.GT.dt1.OR.dtime.GT.dt2) THEN
334     keycut = 1
335     RETURN
336 ENDIF
337 ENDIF
338 c      !
339 c      !*****!
340 c      ! Atualização do módulo constitutivo
341 c      !*****!
342 PHItDg = DOT_PRODUCT(dPHIds,MATMUL(dsdeEl,dgdsVP))
343 aux1 = (eta/dtime + thetaVP*PHItDg)
344 dPHIds_m(1,:) = dPHIds
345 dgdsVP_m(:,1) = dgdsVP
346 DgPHItD = MATMUL(MATMUL
347 &      (MATMUL(dsdeEl,dgdsVP_m),dPHIds_m),dsdeEl)
348 dsdePl = dsdePl - thetaVP*DgPHItD/aux1
349 c      !
350 c      !*****!
351 c      ! Calculo do p
352 c      !*****!
353 sigmap = PHI*MATMUL(dsdeEl(1:ncomp,1:ncomp),
354 &      dgdsVP(1:ncomp))/aux1
355 c      !
356 c      !*****!
357 c      ! Calculo da deformação viscoplastica
358 c      !*****!
359 PHItDdeps = DOT_PRODUCT(dPHIds,MATMUL(dsdeEl,dstrain))
360 dlambaVP = (PHI + thetaVP*PHItDdeps)/aux1
361 depsVP = dlambaVP*dgdsVP
362 c      !
363 c      ! Calcula as deformações viscoplasticas totais
364 epsVP = epsVP + depsVP
365 CALL normatensor(epsVP,ncomp,epsVPeq)

```

```

366 CALL calcula_Czao(superficieVP,fiVP,CzaoVP)
367 epsVPeq = CzaoVP*epsVPeq
368 C !
369 C !
370 C !*****!
371 C ! Calculo preditor elástico !
372 C !*****!
373 stresstrial = 0.0d0
374 stresstrial = MATMUL(dsdeEl(1:ncomp,1:ncomp),
375 & strain(1:ncomp)+dstrain(1:ncomp)
376 & -epsEP(1:ncomp)-epsVP(1:ncomp)) + sigi
377 stress = stresstrial
378 C !
379 C !*****!
380 C ! Calcula função de escoamento !
381 C !*****!
382 CALL invars(stresstrial,ncomp,I1,J2,J3,theta,s) ! Invariantes
383 CALL yield(superficieVP,I1,J2,theta,c,fi,f) ! função de escoamento
384 C !
385 C !*****!
386 C ! Verifica o critério de escoamento !
387 C !*****!
388 IF(f.GT.0.0d0) THEN
389 C !
390 C ! Aplica o corretor plástico
391 C !
392 C !*****!
393 C ! Calcula dgds !
394 C !*****!
395 dgds = 0.0d0
396 stressn = stresstrial
397 CALL invars(stressn,ncomp,I1,J2,J3,theta,s)
398 CALL glg2g3(s,ncomp,J2,g1,g2,g3)
399 CALL c1c2c3(J2,theta,superficieVP,psi,c1,c2,c3)
400 dgds = c1*g1 + c2*g2 + c3*g3
401 C !
402 C !*****!
403 C ! Calcula dhdq referente ao endurecimento e amolecimento !
404 C !*****!
405 dhdq = 0.0d0
406 dfdq = -1.0d0
407 CALL calcula_Xi(superficieVP,fi,Xi)
408 dqdc = Xi
409 CALL calcula_dcde(ci,cp,cr,eps1,eps2,eps3,epsEPEq,dcde)
410 dhdq = -dfdq*dqdc*dcde
411 C !
412 C !*****!
413 C ! Interações de NR local do modelo constitutivo !
414 C !*****!
415 k = 0
416 DO
417 C !*****!
418 C ! Calculo de ddlamb !
419 C !*****!
420 C !
421 C ! Calcula dfds
422 dfds = 0.0d0
423 CALL invars(stress,ncomp,I1,J2,J3,theta,s)
424 CALL glg2g3(s,ncomp,J2,g1,g2,g3)
425 CALL c1c2c3(J2,theta,superficieVP,fi,c1,c2,c3)
426 dfds = c1*g1 + c2*g2 + c3*g3
427 C !
428 C ! Calcula dfdq
429 dfdq = -1.0d0
430 C !
431 C ! Calcula denominador de ddlamb
432 ftDg = DOT_PRODUCT(dfds,MATMUL(dsdeEl,dgds))
433 C !
434 C ! Calcula ddlamb
435 ddlam = f/(ftDg - dfdq*dhdq)
436 C !
437 C !*****!
438 C ! Calculo do corretor plástico !

```

```

439      C      !*****!
440      dstress = -ddlam*MATMUL(dsdeEl,dgds)
441      dq      = -ddlam*(-dhdq)
442      C      !
443      C      !*****!
444      C      ! Incremento das tensões e do dlam
445      C      !*****!
446      stress  = stress + dstress
447      c       = c + dq/Xi
448      dlam    = dlam + ddlam
449      q       = q + dq
450      k = k + 1
451      C      !
452      C      !*****!
453      C      ! Calcula deformação plástica equivalente
454      C      !*****!
455      call matinv(ncomp,dsdeEl,dsdeElinv)
456      depsEP = depsEP-MATMUL(dsdeElinv,dstress)
457      C      !
458      C      !*****!
459      C      ! Verifica critério de escoamento
460      C      !*****!
461      CALL invars(stress,ncomp,I1,J2,J3,theta,s)
462      CALL yield(superficief,I1,J2,theta,c,fi,f)
463      IF(f.LE.tolEP)EXIT
464      C      !
465      C      ! Caso atinja o número de iterações limites, faça a bisseção
466      IF(k.EQ.nrmax)THEN
467          keycut = 1
468          RETURN
469      ENDIF
470      ENDDO
471      C      !
472      C      !*****!
473      C      ! Atualizando o módulo constitutivo
474      C      !*****!
475      IF(dalg.EQ.1)THEN
476          dfds_m(1,:) = dfds
477          dgds_m(:,1) = dgds
478          DgftD= MATMUL(MATMUL(MATMUL(dsdeEl,dgds_m),dfds_m),dsdeEl)
479          dsdePl = dsdePl - DgftD/(ftDg - dfdq*dhdq)
480      ENDIF
481      C      !
482      ELSE
483          depsEp = 0.0d0
484      ENDIF
485      C      !
486      C      !*****!
487      C      ! Guardando deformações plásticas totais
488      C      !*****!
489      C      !
490      C      ! Calcula as deformações plásticas totais
491      epsEP = epsEP + depsEP
492      CALL normatensor(epsEP,ncomp,epsEPeq)
493      CALL calcula_Czao(superficief,fi,Czao)
494      epsEPeq = Czao*epsEPeq
495      C      !
496      C      ! Retorna as deformações inelásticas totais e equivalentes (só tem plásticas)
497      epsPl = epsEP+epsVP
498      epseq = epsEPeq+epsVPeq
499      C      !
500      C      ! Calcula o trabalho elástico
501      sedEl = 0.0d0
502      sedEl = 1.0d0/2.0d0*
503      & DOT_PRODUCT(stress,strain+dstrain-epsPl-epsVP)
504      C      !
505      C      !
506      C      ! Calcula o trabalho inelástico (só tem plástico)
507      CALL normatensor(depsEP,ncomp,depsEPeq)
508      CALL calcula_Czao(superficief,fi,Czao)
509      depsEPeq = Czao*depsEPeq
510      CALL normatensor(depsVP,ncomp,depsVPeq)
511      CALL calcula_Czao(superficief,fi,Czao)

```

```

512     depsVPeq = Czaos*depsVPeq
513     sedPl = sedPl + (q)*depsEPeq+depsVPeq
514 c     !
515 c     ! Guarda valores nas variáveis de estado
516     ustatev(1) = epsEPeq
517     ustatev(2) = dlam
518     ustatev(3) = c
519     ustatev(4) = q
520     ustatev(5) = f
521     ustatev(6) = epsVPeq
522     ustatev(7) = dt1
523     ustatev(8) = dt2
524     CALL vmove(epsEP(1), ustatev(9), ncomp)
525     CALL vmove(epsVP(1), ustatev(15), ncomp)
526 c     !
527     RETURN
528     END
529     !
530     SUBROUTINE matrizD(E,Poisson,ncomp,D)
531 c     !*****!
532 c     !** Função: matrizD **!
533 c     !** **!
534 c     !** Objetivo: calcula a matriz consitutiva do material isotrópico **!
535 c     !**          adaptado de Smith, Griffiths e Margetts (2014, p.42-44) **!
536 c     !** **!
537 c     !** Situação: (28-09-2016) OK **!
538 c     !** **!
539 c     !*****!
540     IMPLICIT NONE
541     DOUBLE PRECISION E           ! módulo de elasticidade
542     DOUBLE PRECISION Poisson     ! coeficiente de Poisson
543     INTEGER ncomp                ! numero de componentes
544     DOUBLE PRECISION D(ncomp,ncomp) ! matriz constitutiva elástica isotrópica
545 c     !
546     D= .0d0
547     D( ,1)=(E*(1.0d0-Poisson))/((1.0d0+Poisson)*(1.0d0-2.0d0*Poisson))
548     D( ,2)=(E*Poisson)/((1.0d0+Poisson)*(1.0d0-2.0d0*Poisson))
549     D( ,3)=D(1,2)
550     D( ,1)=D(1,2)
551     D( ,2)=D(1,1)
552     D( ,3)=D(1,2)
553     D( ,1)=D(1,3)
554     D( ,2)=D(2,3)
555     D( ,3)=D(1,1)
556     D( ,4)=(E)/((1.0d0+Poisson)*2.0d0)
557 c     !
558     IF(ncomp.EQ.6) THEN
559         D(ncomp-1,ncomp-1)=D(4,4)
560         D(ncomp,ncomp)=D(4,4)
561     ENDIF
562 c     !
563     END SUBROUTINE MatrizD
564 c     !
565     SUBROUTINE invars(stress,ncomp,I1,J2,J3,theta,s)
566 c     !*****!
567 c     !** Subrotina: invars **!
568 c     !** **!
569 c     !** Objetivo: calcula os invariantes do tensor de tensões e o tensor **!
570 c     !**          desviador. Adaptado de Chen e Han (1998, p.57-72) **!
571 c     !** **!
572 c     !** Situação: (10-11-2021) OK **!
573 c     !** **!
574 c     !*****!
575     IMPLICIT NONE
576     DOUBLE PRECISION stress(ncomp) ! tensões
577     INTEGER ncomp                ! numero de componentes
578     DOUBLE PRECISION I1          ! primeiro invariante do tensor de tensões
579     DOUBLE PRECISION J2,J3       ! segundo e terceiro invariante do desviador
580     DOUBLE PRECISION theta       ! angulo de Lode
581     DOUBLE PRECISION s(6)        ! desviador
582     DOUBLE PRECISION p,q         ! pressão hidrostática e tensão eq. de vm
583     DOUBLE PRECISION sine        ! variavel auxiliar
584 c     !

```



```

585 c      ! Inicializando variaveis
586 I1      = 0.0d0
587 J2      = 0.0d0
588 p       = 0.0d0
589 q       = 0.0d0
590 s       = 0.0d0
591 J3      = 0.0d0
592 theta   = 0.0d0
593 sine    = 0.0d0
594 c      !
595 c      ! Calculo do I1 (p. 53)
596 I1 = stress(1) + stress(2) + stress(3)
597 c      !
598 c      ! Calculo do J2 (p. 58)
599 J2 = 1/6.0d0*((stress(1)-stress(2))**2+(stress(2)-stress(3))**2+
600 &      (stress(3)-stress(1))**2)+
601 &      stress(4)**2
602 c      !
603 IF(ncomp.EQ.6) THEN
604     J2 = J2 + stress(ncomp-1)**2+stress(ncomp)**2
605 ENDIF
606 c      !
607 c      ! Calculo do p (p. 57)
608 p = 1.0d0/3.0d0*I1
609 c      !
610 c      ! Calculo do desviador s (p. 57)
611 s(1) = stress(1) - p
612 s(2) = stress(2) - p
613 s(3) = stress(3) - p
614 s(4) = stress(4)
615 IF(ncomp.EQ.6) THEN
616     s(ncomp-1) = stress(ncomp-1)
617     s(ncomp) = stress(ncomp)
618 ENDIF
619 c      !
620 c      ! Calculo do J3 (p. 58)
621 J3 = s(1)*s(2)*s(3)-s(3)*s(4)*s(4)
622 IF(ncomp.EQ.6) THEN
623     J3 = J3 -s(1)*s(ncomp-1)*s(ncomp-1)-s(2)*s(ncomp)*s(ncomp)+
624 &      2.0d0*s(4)*s(ncomp-1)*s(ncomp)
625 ENDIF
626 c      !
627 c      ! Calculo do ângulo de Lode (p. 70) e Owen e Hinton (1980, p.229)
628 q = DSQRT(3.0d0*J2)
629 IF(q < 1.E-7) THEN
630     theta = 0.0d0
631 ELSE
632     sine = -3.0d0*DSQRT(3.0d0)*J3/(2.0d0*DSQRT(J2)**3)
633     IF(sine>1.0d0)sine=1.0d0
634     IF(sine<-1.0d0)sine=-1.0d0
635     theta=DASIN(sine)/3.0d0
636 END IF
637 c      !
638 END SUBROUTINE
639 c      !
640 SUBROUTINE glg2g3(s,ncomp,J2,g1,g2,g3)
641 c      !*****!
642 c      !** Subrotina: glg2g3 **!
643 c      !** **!
644 c      !** Objetivo: calcula as direções do vetor de fluxo. Adaptado de Owen e **!
645 c      !** Hinton (1980, p.231 e 233) **!
646 c      !** **!
647 c      !** Situação: (10-11-2021) OK **!
648 c      !** **!
649 c      !*****!
650 IMPLICIT NONE
651 DOUBLE PRECISION s(ncomp)           ! desviador
652 INTEGER          ncomp               ! numero de componentes
653 DOUBLE PRECISION J2                 ! segundo invariante do desviador
654 DOUBLE PRECISION g1(ncomp), g2(ncomp), g3(ncomp)
655 c      !
656 c      ! Inicializando variaveis
657 g1 = 0.0d0

```

```

658      g2 = 0.0d0
659      g3 = 0.0d0
660      c      !
661      c      ! Calculo do g1
662      g1(1) = 1.0d0
663      g1(2) = 1.0d0
664      g1(3) = 1.0d0
665      c      !
666      c      ! Calculo do g2
667      g2(1) = s(1)
668      g2(2) = s(2)
669      g2(3) = s(3)
670      g2(4) = 2.0d0*s(4)
671      IF(ncomp.EQ.6) THEN
672          g2(ncomp-1) = 2.0d0*s(ncomp-1)
673          g2(ncomp) = 2.0d0*s(ncomp)
674      ENDIF
675      IF(J2.EQ.0.0d0) THEN
676          g2 = 0.0d0
677      ELSE
678          g2 = 1.0d0/(2.0d0*DSQRT(J2))*g2
679      ENDIF
680      c      !
681      c      ! Calculo do g3
682      g3(1) = s(2)*s(3) + J2/3.0d0
683      g3(2) = s(1)*s(3) + J2/3.0d0
684      g3(3) = s(1)*s(2) - s(4)**2 + J2/3.0d0
685      g3(4) = 2.0d0*(-s(3)*s(4))
686      IF(ncomp.EQ.6) THEN
687          g3(1) = g3(1) - s(ncomp-1)**2
688          g3(2) = g3(2) - s(ncomp)**2
689          g3(4) = g3(4) + 2.0d0*s(ncomp-1)*s(ncomp)
690          g3(ncomp-1) = 2.0d0*(s(ncomp)*s(4)-s(1)*s(ncomp-1))
691          g3(ncomp) = 2.0d0*(s(4)*s(ncomp-1)-s(2)*s(ncomp))
692      ENDIF
693      c      !
694      END SUBROUTINE
695      c      !
696      c      !
697      SUBROUTINE c1c2c3(J2,theta,superficie,fi,c1,c2,c3)
698      c      !*****!
699      c      !** Subrotina: c1c2c3 **!
700      c      !** **!
701      c      !** Objetivo: calcula a magnitude das componentes do vetor de fluxo. **!
702      c      !** Adaptado de Bernaud (1991, p.90, 91) **!
703      c      !** Adaptado de Owen e Hinton (1980, p.231) **!
704      c      !** **!
705      c      !** Situação: (10-11-2021) OK **!
706      c      !** **!
707      c      !*****!
708      IMPLICIT NONE
709      DOUBLE PRECISION J2          ! segundo invariante do desviador
710      DOUBLE PRECISION theta      ! ângulo de Lode
711      DOUBLE PRECISION fi         ! Ângulo de atrito
712      INTEGER          superficie  ! 1-DPI, 2-DPII, 3-DPIII
713      DOUBLE PRECISION c1,c2,c3   ! magnitude das componentes do vetor
714      DOUBLE PRECISION beta1,beta2,beta3 ! parâmetros do DP
715      DOUBLE PRECISION k          ! coeficiente de empuxo
716      c      !
717      c      ! Seleciona o modelo
718      SELECT CASE (superficie)
719          CASE(1)
720              c      !
721              c      ! DPI
722              k = (1.0d0+DSIN(fi))/(1.0d0-DSIN(fi))
723              c1 = (k-1.0d0)/3.0d0
724              c2 = (k+2.0d0)/DSQRT(3.0d0)
725              c3 = 0.0d0
726              c      !
727          CASE(2)
728              c      !
729              c      ! DPII
730              k = (1.0d0+DSIN(fi))/(1.0d0-DSIN(fi))

```

```

731         c1 = (k-1.0d0)/3.0d0
732         c2 = (2.0d0*k+1.0d0)/DSQRT(3.0d0)
733         c3 = 0.0d0
734     c      !
735         CASE(3)
736     c      !
737     c      ! DPIII - Owen e Hinton (1980, p.231)
738         c1 = 2.0d0*DSIN(fi)/(DSQRT(3.0d0)*(3.0d0+DSIN(fi)))
739         c2 = 1.0d0
740         c3 = 0.0d0
741     c      !
742     ENDSELECT
743     END SUBROUTINE
744     !
745     SUBROUTINE yield(superficie,I1,J2,theta,c,fi,f)
746     c      !*****!
747     c      !** Subrotina: yield **!
748     c      !** **!
749     c      !** Objetivo: calcula o critério de escoamento. **!
750     c      !** Adaptado de Bernaud (1991, p.90, 91) **!
751     c      !** Adaptado de Souza Neto, Peri, Owen (2008, p. 162-167) **!
752     c      !** **!
753     c      !** Situação: (10-11-2021) OK **!
754     c      !** **!
755     c      !*****!
756     IMPLICIT NONE
757     INTEGER      superficie      ! 1-DPI, 2-DPII, 3-DPIII
758     DOUBLE PRECISION I1          ! primeiro invariante do tensor de tensões
759     DOUBLE PRECISION J2          ! segundo invariante do desviador
760     DOUBLE PRECISION theta      ! ângulo de Lode
761     DOUBLE PRECISION c          ! coesão
762     DOUBLE PRECISION fi         ! Ângulo de atrito
763     DOUBLE PRECISION f          ! função de escoamento
764     DOUBLE PRECISION betal,beta2,beta3 ! Parametros para DP
765     DOUBLE PRECISION k          ! coeficiente de empuxo
766     c      !
767     c      ! Seleciona o modelo
768     SELECT CASE (superficie)
769     CASE(1)
770     c      !
771     c      ! DPI
772         k = (1.0d0+DSIN(fi))/(1.0d0-DSIN(fi))
773         betal = (k-1.0d0)/3.0d0
774         beta2 = (k+2.0d0)/DSQRT(3.0d0)
775         beta3 = 2.0d0*DSQRT(k)*c
776         f = betal*I1 + beta2*DSQRT(J2)-beta3
777     c      !
778     CASE(2)
779     c      !
780     c      ! DPII
781         k = (1.0d0+DSIN(fi))/(1.0d0-DSIN(fi))
782         betal = (k-1.0d0)/3.0d0
783         beta2 = (2.0d0*k+1.0d0)/DSQRT(3.0d0)
784         beta3 = 2.0d0*DSQRT(k)*c
785         f = betal*I1 + beta2*DSQRT(J2)-beta3
786     c      !
787     CASE(3)
788     c      !
789     c      ! DPIII Souza Neto, Peri, Owen (2008, p. 162-167)
790         betal = 2.0d0*DSIN(fi)/(DSQRT(3.0d0)*(3.0d0+DSIN(fi)))
791         beta2 = 1.0d0
792         beta3 = 6.0d0*DCOS(fi)/(DSQRT(3.0d0)*(3.0d0+DSIN(fi)))*c
793         f = betal*I1 + beta2*DSQRT(J2)-beta3
794     c      !
795     ENDSELECT
796     END SUBROUTINE
797     !
798     subroutine matinv(n,a,ainv)
799     c      !*****!
800     c      !** Subrotina: matinv **!
801     c      !** **!
802     c      !** Objetivo: inverte uma matriz pela técnica de pivotamento **!
803     c      !** **!

```

```

804 c      !** Situação: (26-10-2016) OK                                     **!
805 c      !**                                                                 **!
806 c      !*****!
807         INTEGER          n          ! dimensão do sistema
808         DOUBLE PRECISION a(n,n)      ! matriz dos coeficientes
809         DOUBLE PRECISION ainv(n,n)   ! matriz inversa
810         DOUBLE PRECISION b(n,2*n)    ! matriz aumentada
811         DOUBLE PRECISION pivot       ! pivô
812         DOUBLE PRECISION xnum        ! auxiliar
813         INTEGER           i,j,k      ! contador
814 c      !
815 c      ! Fazer matriz aumentada
816         do i=1,n
817             do j=1,n
818                 b(i,j) = 0.0d0
819                 b(i,j+n) = 0.0d0
820                 b(i,j)=a(i,j)
821                 if(i.eq.j) then
822                     b(i,j+n)=1.0d0
823                 endif
824             enddo
825         enddo
826 c      !
827         do i=1,n
828 c          ! Escolher o elemento não nulo mais a esquerda como pivot
829             do j=1,n
830                 if (dabs(b(i,j)).gt.0.0d0) then
831                     pivot=b(i,j)
832                     exit
833                 endif
834             enddo
835 c          !
836 c          ! Passo 1: alterar o pivo escolhido
837             do j=1,2*n
838                 b(i,j)=b(i,j)/pivot
839             enddo
840             pivot=b(i,i)
841 c          !
842 c          ! Passo 2: mudando o restante da coluno do pivo para 0,
843 c          ! adicionando a cada linha um multiplo adequado do pivot
844             do k=1,n
845                 if(k.ne.i) then
846                     xnum=b(k,i)/pivot
847                     do j=1,2*n
848                         b(k,j)=b(k,j)-xnum*b(i,j)
849                     enddo
850                 endif
851             enddo
852         enddo
853 c      !
854 c      ! Prepara a matriz inversa final
855         do i=1,n
856             do j=1,n
857                 ainv(i,j)=b(i,j+n)
858             enddo
859         enddo
860         return
861     end
862 c      !
863     SUBROUTINE normatensor(tensor,ncomp,norma)
864 c      !*****!
865 c      !** Subrotina: normatensor                                     **!
866 c      !**                                                                 **!
867 c      !** Objetivo: calcula a norma de um tensor escrito em notação de Voigt **!
868 c      !**                                                                 **!
869 c      !** Situação: (10-11-2021) OK                                     **!
870 c      !**                                                                 **!
871 c      !*****!
872     IMPLICIT NONE
873     INTEGER ncomp
874     DOUBLE PRECISION tensor(ncomp)
875     DOUBLE PRECISION norma
876 c      !

```

```

877      IF(ncomp.EQ.6) THEN
878          norma = DSQRT(tensor(1)**2 + tensor(2)**2 + tensor(3)**2 +
879      &                2*((tensor(4))**2 +
880      &                (tensor(5))**2 + (tensor(6))**2))
881      ELSEIF(ncomp.EQ.4) THEN
882          norma = DSQRT(tensor(1)**2 + tensor(2)**2 + tensor(3)**2 +
883      &                2*(tensor(4)**2))
884      ELSE
885      ENDIF
886      !
887      END SUBROUTINE
888      !
889      SUBROUTINE calcula_Czao(superficie,fi,Czao)
890      !*****!
891      !** Subrotina: Czao **!
892      !** **!
893      !** Objetivo: calcula o C utilizado no calculo da deformação plástica **!
894      !**          efetiva. Adaptado de Chen e Han (1988, p. 257-259). **!
895      !** **!
896      !** Situação: (10-11-2021) OK **!
897      !** **!
898      !*****!
899      IMPLICIT NONE
900      INTEGER          superficie      ! 1-DPI, 2-DPII, 3-DPIII
901      DOUBLE PRECISION fi              ! angulo de atrito
902      DOUBLE PRECISION Czao           ! constante da deformação plástica efetiva
903      DOUBLE PRECISION beta           ! constante referente a pressão hidrostática
904      !
905      ! Seleciona o modelo
906      SELECT CASE (superficie)
907      CASE(1)
908      !
909      ! DPI
910      beta = 2.0d0*DSIN(fi)/(DSQRT(3.0d0)*(3.0d0-DSIN(fi)))
911      !
912      CASE(2)
913      !
914      ! DPII
915      beta = 2.0d0*DSIN(fi)/(DSQRT(3.0d0)*(3.0d0+DSIN(fi)))
916      !
917      CASE(3)
918      !
919      ! DPIII
920      beta = 2.0d0*DSIN(fi)/(DSQRT(3.0d0)*(3.0d0+DSIN(fi)))
921      !
922      ENDSELECT
923      Czao = (beta+1.0d0/DSQRT(3.0d0))/
924      &      (DSQRT(3.0d0*beta**2+1.0d0/2.0d0))
925      END SUBROUTINE
926      !
927      SUBROUTINE calcula_Xi(superficie,fi,Xi)
928      !*****!
929      !** Subrotina: calcula_Xi **!
930      !** **!
931      !** Objetivo: calcula Xi **!
932      !**          Adaptado de Potts e Zdravkovic (1999, p. 158) **!
933      !** **!
934      !** **!
935      !** Situação: (10-11-2021) OK **!
936      !** **!
937      !*****!
938      IMPLICIT NONE
939      INTEGER          superficie      ! 1-DPI, 2-DPII, 3-DPIII
940      DOUBLE PRECISION fi              ! angulo de atrito
941      DOUBLE PRECISION Xi              ! derivada de f em relação a c
942      DOUBLE PRECISION k              ! coeficiente de empuxo
943      !
944      SELECT CASE (superficie)
945      CASE(1)
946      k = (1.0d0+DSIN(fi))/(1.0d0-DSIN(fi))
947      Xi = 2.0d0*DSQRT(k)
948      CASE(2)
949      k = (1.0d0+DSIN(fi))/(1.0d0-DSIN(fi))

```

```

950         Xi = 2.0d0*DSQRT(k)
951 CASE(3)
952     k = (1.0d0+DSIN(fi))/(1.0d0-DSIN(fi))
953     Xi = 6.0d0*DCOS(fi)/(DSQRT(3.0d0)*(3.0d0+DSIN(fi)))
954 END SELECT
955 END SUBROUTINE
956 c !
957 SUBROUTINE calcula_dcde(ci,cp,cr,eps1,eps2,eps3,epsPleq,
958 & dcde)
959 c !*****!
960 c !** Subrotina: calcula_dcde **!
961 c !** **!
962 c !** Objetivo: calcula dc/de **!
963 c !** Adaptado de Potts e Zdravkovic (1999, p. 158) **!
964 c !** **!
965 c !** Situação: (10-11-2021) OK **!
966 c !** **!
967 c !*****!
968 IMPLICIT NONE
969 DOUBLE PRECISION ci ! coesão inicial
970 DOUBLE PRECISION cp ! coesão no pico
971 DOUBLE PRECISION cr ! coesão residual
972 DOUBLE PRECISION eps1 ! deformação plastica equivalente 1
973 DOUBLE PRECISION eps2 ! deformação plastica equivalente 2
974 DOUBLE PRECISION eps3 ! deformação plastica equivalente 3
975 DOUBLE PRECISION epsPleq ! deformação plástica equivalente
976 DOUBLE PRECISION dcde ! dc/depsPleq
977 c !
978 IF(epsPleq.LT.eps1) THEN
979     dcde = (cp-ci)/(eps1)
980 ELSEIF((epsPleq.GE.eps1).AND.(epsPleq.LE.eps2)) THEN
981     dcde = 0.0d0
982 ELSEIF((epsPleq.GT.eps2).AND.(epsPleq.LT.eps3)) THEN
983     dcde = (cr-cp)/(eps3-eps2)
984 ELSEIF(epsPleq.GE.eps3) THEN
985     dcde = 0.0d0
986 ENDIF
987 c !
988 END SUBROUTINE
989

```