Macro commande SIMU_POINT_MAT

1 But

Calculer l'évolution mécanique d'un point matériel, en quasi-statique non linéaire.

Tous les comportements disponibles dans STAT_NON_LINE [U4.51.11] le sont également ici.

Le but de cette macro-commande est de simplifier au maximum les données : il suffit de fournir :
1) Le comportement et le matériau ;
2) Les fonctions définissant l'évolution des composantes de contraintes ou de déformations choisies ;
3) La discrétisation en temps.

Ceci permet en particulier de calculer l'évolution du tenseur des contraintes dans le cas de déformations imposées, ou l'inverse (cas courants en identification de paramètres matériau)

Produit une structure de données de type table contenant, en fonction du temps, l'évolution de toutes les composantes des tenseurs de contraintes et de déformations, ainsi que les variables internes.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

2 Syntaxe

```
tabres [table] = SIMU_POINT_MAT (

♦  /   COMPORTEMENT     =_F (voir le document [U4.51.11]),

♦ MATER             =  / mat ,  [mater]
                       / l_mat, [l_mater]

◊  MASSIF            = /'ANGL_REP'                      [R]
                       /'ANGL_EULER'                    [R]
◊  ANGLE             = angz,                            [R]

♦  INCREMENT        =_F ( voir le document [U4.51.03]),

◊  NEWTON           =_F (  voir le document [U4.51.03]),

 ◊ CONVERGENCE      =_F(
      /RESI_GLOB_RELA     =  1.E-6,                     [DEFAUT]
      /|RESI_GLOB_MAXI    =  resmax,                    [R]
       |RESI_GLOB_RELA    =  resrel,                    [R]
       ITER_GLOB_MAXI     =  /10,                       [DEFAUT]
                             /maglob,                   [I]
                  ),
◊  SUPPORT =   /'ELEMENT'

            ◊  MODELISATION      =     /'3D'        [DEFAUT]
                                       /'C_PLAN'
                                       /'D_PLAN'

            ◊  RECH_LINEAIRE    =_F (  voir le document [U4.51.03]),

            ◊  ARCHIVAGE        =_F (  voir le document [U4.51.03]),

             # variables de commandes fonction scalaires du temps

            ◊  SUIVI_DDL        =_F (  voir le document [U4.51.03]),

            ◊  AFFE_VARC     = _F(♦ NOM_VARC =  /'TEMP',
                                                /'CORR',
                                                /'IRRA',
                                                /'HYDR',
                                                /'SECH',
                                                /'EPSA',
                                                /'NEUT1',
                                                /'NEUT2',
                                    ◊ VALE_REF   =vref        [R]
                                   ♦ VALE_FONC   =foncvarc    [fonction]

                                                /'M_ACIER',
                                   /  ◊ V1 =foncv1            [fonction]
                                   /  ◊ V2 =foncv2            [fonction]
                                   /  ◊ V3 =foncv3            [fonction]
                                   /  ◊ V4 =foncv4            [fonction]
                                   /  ◊ V5 =foncv5            [fonction]
                                   /  ◊ V6 =foncv6            [fonction]
                                   /  ◊ V7 =foncv7            [fonction]
                                   /  ◊ V8 =foncv8            [fonction]
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

```
                                    / ◊ V 9 =foncv 9              [fonction]
                                                  /'M_ZIRC',
                                    / ◊ V1  =foncv1               [fonction]
                                    / ◊ V2  =foncv2               [fonction]
                                    / ◊ V3  =foncv3               [fonction]
                                    / ◊ V4  =foncv4               [fonction]
                                    / ◊ V 5  =foncv 5             [fonction]
                                    ),
                  ◊   SIGM_IMPOSE=_F( ◊  SIXX = sigxx              [fonction]
                                      ◊  SIYY = sigyy              [fonction]
                                     ◊  SIZZ = sigzz              [fonction]
                                     ◊  SIXY = sigxy              [fonction]
                                     ◊  SIXZ = sigxz              [fonction]
                                     ◊  SIYZ = sigyz              [fonction]
                  ),
                  ◊  EPSI_IMPOSE=_F( ◊  EPXX = epsxx             [fonction]
                                     ◊  EPYY = epsyy             [fonction]
                                     ◊  EPZZ = epszz             [fonction]
                                     ◊  EPXY = epsxy             [fonction]
                                     ◊  EPXZ = epsxz             [fonction]
                                     ◊  EPYZ = epsyz             [fonction]
                  ),


◊  SUPPORT=   /'POINT'                              [DEFAUT]
              ◊  NB_VARI_TABLE = nvar  [I]
              ◊  FORMAT_TABLE  =  /'CMP_COLONNE'   [DEFAUT]
                                  /'CMP_LIGNE'
              ◊  OPER_TANGENT  =  /'NON'           [DEFAUT]
                                  /'OUI'
              ◊  ARCHIVAGE =_F (
                    ◊  LIST_INST = linst (voir document [U4.51.03]),
                    ◊  PRECISION = prec  ),
              /  ◊  SIGM_IMPOSE=_F( ◊  SIXX = sigxx            [fonction]
                                    ◊  SIYY = sigyy            [fonction]
                                    ◊  SIZZ = sigzz            [fonction]
                                    ◊  SIXY = sigxy            [fonction]
                                    ◊  SIXZ = sigxz            [fonction]
                                    ◊  SIYZ = sigyz            [fonction]),
                 ◊   EPSI_IMPOSE=_F( ◊  EPXX = epsxx           [fonction]
                                    ◊  EPYY = epsyy            [fonction]
                                    ◊  EPZZ = epszz            [fonction]
                                    ◊  EPXY = epsxy            [fonction]
                                    ◊  EPXZ = epsxz            [fonction]
                                    ◊  EPYZ = epsyz            [fonction]),
              /  ◊  GRAD_IMPOSE=_F( ◊  F11 = fonc              [fonction]
                                    ◊  F12 = fonc              [fonction]
                                    ◊  F12 = fonc              [fonction]
                                    ◊  F13 = fonc              [fonction]
                                    ◊  F21 = fonc              [fonction]
                                    ◊  F22 = fonc              [fonction]
                                    ◊  F23 = fonc              [fonction]
                                    ◊  F31 = fonc              [fonction]
                                    ◊  F32 = fonc              [fonction]
                                    ◊  F33 = fonc              [fonction]),
              /  ◊  MATR_C1=_F(  ♦  VALE = cij                 [R]
                                 ♦  NUME_LIGNE = numlig        [I]
                                  ♦ NUME_COLONNEE = numcol     [I]
                                 ),
                  ◊ MATR_C2 =_F(  ♦ VALE = cij                  [R]
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

```
                                ♦  NUME_LIGNE = numlig          [I]
                                ♦  NUME_COLONNEE = numcol       [I]
                                ),
                    ◊ VECT_IMPO =_F(   ♦ VALE = cij                  [R]
                                ♦  NUME_LIGNE = numlig          [I]
                                ),
              ◊  AFFE_VARC     = _F(♦ NOM_VARC =  /'TEMP',
                                                  /'IRRA',
                                                  /'SECH',
                                   ♦ VALE_FONC   =foncvarc      [fonction]
                                   ◊ VALE_REF    =vref          [R]
                             ),
              ◊   SIGM_INIT=_F(  ◊  SIXX = sigxx                [R]
                                 ◊  SIYY = sigyy                [R]
                                 ◊  SIZZ = sigzz                [R]
                                 ◊  SIXY = sigxy                [R]
                                 ◊  SIXZ = sigxz                [R]
                                 ◊  SIYZ = sigyz                [R] ),
              ◊   EPSI_INIT=_F(  ◊  EPXX = epsxx                [R]
                                 ◊  EPYY = epsyy                [R]
                                 ◊  EPZZ = epszz                [R]
                                 ◊  EPXY = epsxy                [R]
                                 ◊  EPXZ = epsxz                [R]
                                 ◊  EPYZ = epsyz                [R] ),
◊  VARI_INIT=_F(    ◊  VALE = vari                              [R] ),
◊  INFO =     / 1,                          [DEFAUT]
              / 2 , );
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

3 Opérandes

3.1 Opérande MATER

♦ **MATER** = / mat , [mater]
/ l_mat, [l_mater]

Ce mot-clé permet de renseigner le nom du matériau (mater) défini par DEFI_MATERIAU [U4.43.01], où sont fournis les paramètres nécessaires au comportement choisi.
Dans le cas des polycristaux, on peut avoir à donner plusieurs matériaux (cf. ssnv194c).

3.2 Mot-clé COMPORTEMENT

La syntaxe de ce mot clé est décrite dans le document [U4.51.11].

Certaines lois de comportement utilisent des variables d'état de type « intrinsèque », c'est-à-dire des variables d'état (de commande) qui ne sont pas définies par AFFE_VARC. Il y a deux cas :

- Les variables d'état correspondant à la taille d'un élément. Il s'agit des comportements BETON_DOUBLE_DP , ENDO_PORO_BETON , FLUA_ENDO_PORO et RGI_BETON . Ces comportements ne sont pas utilisables dans SIMU_POINT_MAT
- La variable d'état correspondant au gradient de vélocité utilisée dans MONOCRISTAL . Par défaut, on met cette variable à zéro

Remarque : dans le mode SUPPORT='POINT', on suppose que la les coordonnées du point de Gauss sont $(0,0,0)$.

3.3 Mots clés INCREMENT / ARCHIVAGE/ NEWTON

La syntaxe de ces mots-clés est décrite dans le document [U4.51.03].

Le mot-clé INCREMENT définit les intervalles de temps pris dans la méthode incrémentale.

Le mot-clé ARCHIVAGE définit les instants où sont stockés les résultats dans la table tabres. Dans le cas SUPPORT='POINT', ces instants ne peuvent être définis que par le mot-clé LIST_INST avec la précision relative PRECISION.

Le mots-clé NEWTON, facultatif, permet de modifier les valeurs par défaut des paramètres de convergence de la méthode de Newton.

3.4 Mot clé CONVERGENCE

◊ CONVERGENCE =_F()

Si aucun des deux opérandes suivants n'est présent, alors tout se passe comme si : RESI_GLOB_RELA = 1.E-6.

3.4.1 Opérande RESI_GLOB_RELA/RESI_GLOB_MAXI

◊ |RESI_GLOB_RELA = resrel , [R]

L'algorithme continue les itérations globales tant que :

$$\max_{i=1,\ldots,nbddl} |\boldsymbol{F}_i^n| > \text{resrel} \,.\, max|\boldsymbol{L}|$$

où $\boldsymbol{F}^n$ est le résidu de l'itération n et $\boldsymbol{L}$ le vecteur du chargement imposé et des réactions d'appuis (Cf. [R5.03.01] pour plus de détails).

Lorsque le chargement et les réactions d'appui deviennent nuls, c'est-à-dire lorsque $\boldsymbol{L}$ est nul (par exemple dans le cas d'une décharge totale), on essaie de passer du critère de convergence relatif

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

RESI_GLOB_RELA au critère de convergence absolu RESI_GLOB_MAXI. Cette opération est transparente pour l'utilisateur (message d'alarme émis dans le fichier .mess). Lorsque le vecteur L redevient différent de zéro, on repasse automatiquement au critère de convergence relatif RESI_GLOB_RELA.
Toutefois, ce mécanisme de basculement ne peut pas fonctionner au premier pas de temps. En effet, pour trouver une valeur de RESI_GLOB_MAXI raisonnable de manière automatique (puisque l'utilisateur ne l'a pas renseigné), on a besoin d'avoir eu au moins un pas convergé sur un mode RESI_GLOB_RELA . Dès lors, si le chargement est nul dès le premier instant, le calcul s'arrête. L'utilisateur doit déjà alors vérifier que le chargement nul est normal du point de vue de la modélisation qu'il réalise, et si tel est le cas, trouver un autre critère de convergence (RESI_GLOB_MAXI par exemple).

Si cet opérande est absent, le test est effectué avec la valeur par défaut, sauf si RESI_GLOB_MAXI est présent.

◊ |RESI_GLOB_MAXI = resmax , [R]

L'algorithme continue les itérations globales tant que :

$$\max_{i=1,\ldots,nbddl} |F_i^n| > \text{resmax}$$

où F^n est le résidu de l'itération n (Cf. [R5.03.01] pour plus de détails). Si cet opérande est absent, le test n'est pas effectué.

Si RESI_GLOB_RELA et RESI_GLOB_MAXI sont présents tous les deux, les deux tests sont effectués.

3.4.2 Opérande ITER_GLOB_MAXI

◊ ITER_GLOB_MAXI = /10 [DEFAUT]
/maglob

Nombre d'itérations maximum effectué pour résoudre le problème global à chaque instant (10 par défaut).

3.5 Mot clé RECH_LINEAIRE

La syntaxe de ces mots-clés est décrite dans le document [U4.51.03].

Le mot-clé RECH_LINEAIRE permet, dans le cas SUPPORT='ELEMENT', d'activer la recherche linéaire pour aider à la convergence de l'algorithme de Newton. Cette fonctionnalité n'est pas disponible pour SUPPORT='POINT', car elle ne semble pas nécessaire.

3.6 Mot clé MODELISATION

Le mot-clé MODELISATION permet, dans le cas SUPPORT='ELEMENT', d'effectuer le calcul sur un élément 3D ou sur un élément 2D, en contraintes planes ou en déformations planes. Il n'est pas disponible dans le cas SUPPORT='POINT', car il suffit d'imposer une valeur nulle aux composantes correspondantes aux contraintes planes ou aux déformations planes pour obtenir le même résultat.

Ce mot-clé permet de définir la dimension du problème traité : 3D (par défaut) ou bien 2D : déformation plane ou contrainte plane. Dans le cas 2D, les composantes des tenseurs fournis sous les mots-clés SIGM_IMPOSE, EPSI_IMPOSE, SIGM_INIT, EPSI_INIT sont au nombre de 4 : XX, YY, ZZ, XY.

3.7 Opérande ANGLE

Ce mot-clé permet de spécifier un angle (en degrés) pour effectuer une rotation d'ensemble autour de Z appliquée à la fois au chargement, au maillage, et au dépouillement. Ceci permet surtout de vérifier la fiabilité de l'intégration du comportement, comme dans les tests **COMP001, COMP002**.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Par défaut, la rotation est identiquement nulle.

Dans le cas de matériaux possédant une orientation intrinsèque (orthotropie, comportements cristallins), il convient d'utiliser également le mot-clé MASSIF, avec une première valeur d'angle identique à celle fournie sous ANGLE.

3.8 Mot clé MASSIF

3.8.1 Opérandes ANGL_EULER / ANGL_REP

Ces mot-clés permettent de définir une orientation intrinsèque au matériau (orthotropie, comportements cristallins), et permettent de faire appel dans la macro-commande au mot-clé **MASSIF** de **AFFE_CARA_ELEM** [U4.42.01] .
Par défaut, l'orientation est nulle, et on ne fait pas appel à **AFFE_CARA_ELEM**.

3.9 Mots clés SIGM_INIT/EPSI_INIT/VARI_INIT

Ces mots clés permettent de définir un état initial par la donnée :
1) des composantes des contraintes initiales (toutes les composantes ne sont pas nécessaires, par défaut on prend la valeur 0),
2) des composantes des déformations initiales (si le mot clé EPSI_INIT est présent, il faut fournir toutes les composantes des déformations initiales : 4 en 2D, et 6 en 3D)
3) l'ensemble des variables internes initiales pour le comportement utilisé.

Cette fonctionnalité est illustrée dans le test SSNV160E.

3.10 Mots clés SIGM_IMPOSE/EPSI_IMPOSE

3.10.1 Opérandes SIXX, SIYY, SIZZ, SIXY, SIXZ, SIYZ

Ces mot-clés permettent de définir des composantes du tenseur de contraintes imposées au point matériel, par l'intermédiaire de fonctions du temps. Ces fonctions peuvent être définies à l'aide de DEFI_FONCTION [U4.31.02] ou à l'aide de FORMULE [U4.31.05].

Par défaut, les composantes non affectées sont identiquement nulles.

3.10.2 Opérandes EPXX, EPYY, EPZZ, EPXY, EPXZ, EPYZ

Ces mot-clés permettent de définir des composantes du tenseur de déformation imposées au point matériel, par l'intermédiaire de fonctions du temps. Ces fonctions peuvent être définies à l'aide de DEFI_FONCTION [U4.31.02] ou à l'aide de FORMULE [U4.31.05] .

Par défaut, les composantes non affectées sont laissées sans valeur (pas de déformation imposée).

Il convient de noter que, dans le cas du modèle de déformation PETIT_REAC, il n'est pas possible d'imposer exactement la déformation à l'aide d'EPSI_IMPOSE. En effet, à cause de la nature *incrémentale* de ce modèle, les déformations obtenues par ce modèle à la fin du calcul SIMU_POINT_MAT seront différentes de ce qui aura été imposé, sauf en petites déformations. Pour les grandes déformations il donc est préférable d'utiliser le modèle GDEF_LOG qui ne souffre pas de cet inconvénient.

3.11 Mots clé GRAD_IMPOSE

3.11.1 Opérandes F11, F12, F13, F21, F22, F23,F31, F32, F33

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Ces mot-clés permettent de définir toutes les composantes du tenseur gradient de transformation imposé, en grandes déformations (DEFORMATION='SIMO_MIEHE') cf. test ssnd113).

3.12 Mots clés MATR_C1 / MATR_C2 / VECT_IMPO

Ces mot-clés permettent, dans le cas SUPPORT='POINT', de définir directement les coefficients des matrices $C1$, $C2$ et du vecteur g décrits au §4.2 : cela permet ainsi de définir des conditions linéaires sur les inconnues (contraintes et déformations du point matériel) plus générales que les composantes imposées par les mots clés SIGM_IMPOSE/EPSI_IMPOSE. Tous les termes des matrices $C1$ et $C2$ non précisés sont nuls. Pour un exemple d'utilisation, voir le test WTNV134B [V7.31.134].

3.13 Opérande AFFE_VARC

Ce mot-clé permet de spécifier une variable de commande (cf . [U4.43.03]) dont le nom est défini sous le mot clé NOM_VARC ; la fonction définissant l'évolution temporelle de cette variable de commande est fournie via le mot clé VALE_FONC. La valeur éventuelle de référence vref est donnée par VALE_REF.

Dans le cas SUPPORT='ELEMENT', toutes les variables de commande sont autorisées. De plus, pour M_ZIRC (resp. M_ACIER), il faut fournir les évolutions des 4 (resp. 7) phases métallurgiques en fonction du temps.

Dans le cas SUPPORT='POINT', seules les variables de commande ' TEMP ', ' SECH ' et ' IRRA ' sont autorisées.

3.14 Mot clé NB_VARI_TABLE

Le mot-clé NB_VARI_TABLE permet, dans le cas SUPPORT='POINT', de limiter le nombre de variables internes écrites dans la table. En effet pour les milieux polycristallins, celui-ci peut atteindre plusieurs milliers. On limite alors le nombre de colonnes de la table à nvar. Par contre les calculs sont bien sûr effectués avec la totalité des variables internes : celles-ci ne sont tronquées que dans la table en résultat.

3.15 Mot clé FORMAT_TABLE

◊ FORMAT_TABLE = /'CMP_COLONNE' [DEFAUT]
/'CMP_LIGNE'

Le mot-clé FORMAT_TABLE permet, dans le cas SUPPORT='POINT', de définir le mode de stockage des grandeurs dans la table résultat (le test SSNV194C illustre ces deux formats) . Dans le cas où le nombre de variables internes dépasse le maximum de colonnes autorisé pour une table (9999, cf. D4,02,05), le format bascule automatiquement en : FORMAT_TABLE = /'CMP_LIGNE'.

```
FORMAT_TABLE  =  /'CMP_LIGNE' :
...  V845          V846          V847          V848          NB_ITER
...  1.16186E-17   1.32359E-17   1.11751E-17   1.00000E+00   1.00000E+00
...  1.29473E-16   1.47341E-16   1.24474E-16   1.00000E+00   1.00000E+00
...  8.90739E-16   1.00875E-15   8.55093E-16   1.00000E+00   1.00000E+00
...  4.40109E-15   4.92817E-15   4.21938E-15   1.00000E+00   1.00000E+00
...  1.70332E-14   1.87022E-14   1.63484E-14   1.00000E+00   1.00000E+00
...  5.44940E-14   5.80870E-14   5.25904E-14   1.00000E+00   1.00000E+00
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

```
FORMAT_TABLE  =  /'CMP_COLONNE' :
...  INST          GRANDEUR CMP       VALEUR
....
  4.97867E-03 VARI     V845     -1.43828E+01
  4.97867E-03 VARI     V846     -2.63548E+01
  4.97867E-03 VARI     V847      2.80907E+01
  4.97867E-03 VARI     V848      1.00000E+00
  5.00000E-03 EPSI     EPXX     -2.20535E-03
  5.00000E-03 EPSI     EPYY     -1.96506E-03
  5.00000E-03 EPSI     EPZZ      5.00000E-03
  5.00000E-03 EPSI     EPXY     -1.98892E-04
  5.00000E-03 EPSI     EPXZ     -2.11427E-04
  5.00000E-03 EPSI     EPYZ     -3.00870E-04
  5.00000E-03 V7.32.119SIGM     SIXX      1.67146E-04
  5.00000E-03 SIGM     SIYY      2.78713E-05
  5.00000E-03 SIGM     SIZZ      3.01140E+02
  5.00000E-03 SIGM     SIXY      9.15194E-05
  5.00000E-03 SIGM     SIXZ      1.62000E-04
  5.00000E-03 SIGM     SIYZ      6.86376E-05
  5.00000E-03 SIEQ     VMIS      3.01140E+02
  5.00000E-03 SIEQ     TRACE     3.01140E+02
  5.00000E-03 VARI     V1       -1.58316E-03
  5.00000E-03 VARI     V2       -1.34287E-03
  5.00000E-03 VARI     V3        2.92604E-03
  5.00000E-03 VARI     V4       -2.81276E-04
....
```

3.16 Mot clé OPER_TANGENT

◊ OPER_TANGENT = /'NON' [DEFAUT]
/'OUI'

Le mot-clé OPER_TANGENT permet, dans le cas SUPPORT='POINT', d'ajouter à la table résultat les 36 valeurs de l'opérateur tangent issu du comportement.

3.17 Opérande INFO

◊ **INFO = inf**

Permet d'effectuer dans le fichier message diverses impressions intermédiaires.

4 Fonctionnement de la macro_commande

Cette macro_commande a pour but de restreindre au strict nécessaire les données relatives à une simulation sur un point matériel pour un modèle de comportement incrémental.

Le fonctionnement interne allège donc le fichier de commandes de l'utilisateur, en réalisant des opérations répétitives pour ce genre de situations.

4.1 Cas SUPPORT='ELEMENT'

Le fonctionnement est :

1. création d'un maillage d'un seul élément à un seul point de Gauss (un tétraèdre à quatre nœuds en 3D, un triangle à trois nœuds en 2D) (voir par exemple [V6.04.176]).
2. affectation d'un modèle 3D ou C_PLAN ou D_PLAN

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

3. affectation du matériau sur ce maillage ;
4. affectation des chargements :
5. en ce qui concerne les déformations imposées, pour chaque composante affectée via un des mots-clés de EPSI_IMPOSE , création d'un chargement unitaire en déformation qui sera multiplié par la fonction du temps fournie pour cette composante par l'utilisateur ;
6. en ce qui concerne les contraintes imposées, pour chaque composante affectée via un des mots-clés de SIGM_IMPOSE , création d'un chargement unitaire en contraintes qui sera multiplié par la fonction du temps fournie pour cette composante par l'utilisateur ;
7. Appel à STAT_NON_LINE . Tous les mots-clés ayant des valeurs par défaut sont utilisés, sauf s'ils sont surchargés par l'utiliseur (NEWTON, CONVERGENCE, SUIVI_DDL, ARCHIVAGE, RECH_LINEAIRE) et de l'état initial.

L'ensemble des résultats (six ou quatre composantes de contraintes et de déformations, variables internes) sont stockés dans une table (tabres). Pour chaque composante (colonne de la table) figure l'évolution en fonction du temps.

4.2 Cas SUPPORT='POINT'

Dans ce cas, plutôt que d'utiliser un élément fini (même unique) pour réaliser le calcul, SIMU_POIN_MAT fait appel à une commande dédiée, CALC_POINT_MAT, qui englobe en Fortran l'appel direct à la routine 3D d'intégration des comportements, NMCOMP. Ceci n'est disponible que dans le cas des petites déformations.

Rappelons que NMCOMP est la routine générale d'intégration des lois de comportement, appelée par tous les éléments finis 3D et 2D. Elle permet le calcul en un point (ce point étant le point d'intégration pour un élément fini) des contraintes et variables internes à l'instant actuel, connaissant les contraintes et variables internes à l'instant précédent; et l'accroissement de déformation actuel. (cf. [D5.04.01] et [R5.03.14]). Plus précisément, à l'instant t_i, et à l'itération n le tenseur des contraintes $\boldsymbol{\sigma}_i^n$, en un point est calculé à partir de $(\boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1})$ et de l'incrément de déformation $\Delta \boldsymbol{\varepsilon}_i^n$.

Quand toutes les composantes du tenseur des déformations sont fournies, l'algorithme est immédiat : il s'agit d'une simple boucle en temps, contenant pour chaque incrément temporel la donnée de l'état mécanique de l'incrément précédent et le tenseur (symétrique) correpondant à l'accroissement de déformation connu.

Mais dans le cas contraire, soit que l'on fournit seulement n composantes de l'histoire des déformations, $n<6$, soit que l'on fournit n composantes de l'histoire des contraintes, l'algorithme est le suivant :

- par défaut, toute composante non spécifiée correspond à une composante de contrainte assujettie à rester nulle (condition de Neumann)
- les équations à résoudre sont (en utilisant la notation sous forme vectorielle des tenseurs symétriques d'ordre 2) :
- $\boldsymbol{\sigma}_\mathbf{i} = F(\Delta \boldsymbol{\varepsilon}_\mathbf{i} ; \sigma_{i-1}, \alpha_{i-1})$ où F représente le résultat de l'intégration du comportement par NMCOMP
- pour j variant de 1 à 6 :
  - soit $(\sigma_i)_j = g_j(t_i)$
  - soit $(\varepsilon_i)_j = g_j(t_i)$
    - où $\bar{\sigma}_j(t)$ et $\bar{\varepsilon}_j(t)$ sont données par SIGM_IMPOSE / EPSI_IMPOSE.

Ceci peut s'écrire encore :

pour chaque instant t_i, résoudre :

$$R(Y_i)=0 \text{ avec } Y_i = Y(t_i) = \begin{Bmatrix} [\boldsymbol{\sigma}_\mathbf{i}] \\ [\boldsymbol{\varepsilon}_\mathbf{i}] \end{Bmatrix} \text{ et } R(Y_i) = \begin{Bmatrix} [\boldsymbol{\sigma}_\mathbf{i} - F(\Delta \boldsymbol{\varepsilon}_\mathbf{i} ;\ \boldsymbol{\sigma}_\mathbf{i-1}, \alpha_{i-1})] \\ [C_1]\boldsymbol{\sigma}_\mathbf{i} \ + \ [C_2]\boldsymbol{\varepsilon}_\mathbf{i} - \mathbf{g}(t_i) \end{Bmatrix}$$

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

qui est un système non linéaire d'ordre 12.

La dernière relation traduit les condtitions de contraintes ou déformations imposées : les matrices C_1 et C_2 ne contiennent que des termes sur la diagonale, valant 1 si la composante correspondante est imposée, sachant que l'on ne peut avoir à la fois contrainte et déformation imposée.

Par exemple, si la déformation ε_{yy} est imposée, la dernière relation s'écrit :

$$[C_1]\boldsymbol{\sigma}_\mathbf{i} + [C_2]\boldsymbol{\varepsilon}_\mathbf{i} - \mathbf{g}(t_i) = \begin{bmatrix} 1&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&1&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&1&0\\ 0&0&0&0&0&1 \end{bmatrix} \begin{pmatrix} \sigma_{xx}\\ \sigma_{yy}\\ \sigma_{zz}\\ \sigma_{xy}\\ \sigma_{xz}\\ \sigma_{yz} \end{pmatrix} + \begin{bmatrix} 0&0&0&0&0&0\\ 0&1&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0\\ 0&0&0&0&0&0 \end{bmatrix} \begin{pmatrix} \varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\\ \varepsilon_{xy}\\ \varepsilon_{xz}\\ \varepsilon_{yz} \end{pmatrix} = \begin{pmatrix} 0\\ \varepsilon_{yy}(t_i)\\ 0\\ 0\\ 0\\ 0 \end{pmatrix}$$

Dans le cas où l'utilisateur a précisé (via les mots clés `MATR_C1` , `MATR_C2` et `VECT_IMPO`) des relations supplémentaires, celles-ci sont prises en compte directement dans les matrices C_1 et C_2.

La résolution de ce système d'équations non linéaires s'effectue par une méthode de Newton :

–initialisation : $Y_i^0 = Y_{i-1} + [K_i^0]^{-1} \begin{bmatrix} 0 \\ -[C_1]\boldsymbol{\sigma}_{\mathbf{i-1}} - [C_2]\boldsymbol{\varepsilon}_{\mathbf{i-1}} + \mathbf{g}(t_i) \end{bmatrix}$

–a l'itération n :

$$\delta Y_i^{n+1} = -[K_i^n]^{-1} R(Y_i^n) = [K_i^n]^{-1} \begin{bmatrix} F(\Delta\boldsymbol{\varepsilon}_\mathbf{i}^\mathbf{n}; \boldsymbol{\sigma}_{\mathbf{i-1}}, \alpha_{i-1}) - \boldsymbol{\sigma}_\mathbf{i}^\mathbf{n} \\ -[C_1]\boldsymbol{\sigma}_\mathbf{i}^\mathbf{n} - [C_2]\boldsymbol{\varepsilon}_\mathbf{i}^\mathbf{n} + \mathbf{g}(t_i) \end{bmatrix} ;$$

$$\Delta Y_i^n = \delta Y_i^n + \Delta Y_i^{n-1} \ ; \ Y_i^n = \Delta Y_i^n + Y_{i-1}$$

avec

$$[K_i^n] = \left[\frac{\partial R}{\partial Y}\right] = \begin{bmatrix} \mathbf{1} & -\left(\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}}\right)_i^n \\ [C_1] & [C_2] \end{bmatrix} \text{ et } [K_i^0] = \begin{bmatrix} \mathbf{1} & -\left(\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}}\right)_i^0 \\ [C_1] & [C_2] \end{bmatrix}$$

où $\left(\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}}\right)_i^0$ représente l'opérateur tangent de prédiction (option `RIGI_MECA_TANG` , cf. [R5.03.01, R5.03.02]) et $\left(\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}}\right)_i^n$ représente l'opérateur tangent cohérent (option `FULL_MECA)` , cf. [R5.03.01, R5.03.02]). Ces opérateurs peuvent être remplacés par l'opérateur d'élasticité suivant les mots-clés `PREDICTION` , `REAC_ITER` .

La seule non linéarité du problème provient du comportement : $F(\Delta\boldsymbol{\varepsilon}_\mathbf{i}^\mathbf{n}; \boldsymbol{\sigma}_{\mathbf{i-1}}, \alpha_{i-1})$.

Dans le cas d'un comportement linéaire, on vérifie que la solution du problème est obtenue à l'issue de phase de prédiction.

La convergence des itérations est vérfiée :

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

- soit en valeur relative, (mot clé RESI_GLOB_RELA) :

$$max\left(\frac{\underset{j=1,6}{max}\left|\left(R_i^n\right)_j\right|}{\underset{j=1,6}{max}\left|\left(\sigma_i^0\right)_j\right|};\frac{\underset{j=7,12}{max}\left|\left(R_i^n\right)_j\right|}{\underset{j=7,12}{max}\left|\left(R_i^0\right)_j\right|}\right)<\text{RESI_GLOB_RELA}$$

  Dans ce cas, les termes en contraintes et en déformations sont séparés pour l'examen du critère de convergence pour éviter les problèmes dus aux différences d'ordres de gandeur.

- soit en valeur absolue (mot clé RESI_GLOB_MAXI) ou vavleur du dénominateur proche de zéro dans le critère relatif ci-dessus :

$$\underset{j=1,12}{max}\left|\left(R_i^n\right)_j\right|<\text{RESI_GLOB_MAXI}$$

Les options de calcul de la rigidité tangente par perturbation et la gestion automatique du pas de temps sont également activées, comme dans [U4.51.03].

Dans la résolution précédente, les termes en contraintes sont adimensionnalisés, pour éviter un mauvais conditionnement de la matrice jacobienne. On divise donc pour la résolution tous les termes en contraintes par le max des termes diagonaux de l'opérateur d'élasticité ; il faut donc fournir dans DEFI_MATERIAU le mot-clé ELAS ou ELAS_ORTH ou ELAS_ISTR.

Dans le cas SUPPORT='POINT', certains mots clés n'ont pas d'utilité :
- dans le cas de la recherche linéaire, celle-ci n'est pas programmée dans la version actuelle
- dans le cas de l'archivage, seul le mot-clé LIST_INST est pris en compte
- dans le cas CONVERGENCE / RESI_REFE_RELA , ce mot-clé sans objet : le résidu par valeur de référence n'a pas de sens pour un point matériel.
- Dans le cas où la valeur du mot clé DEFORMATION n'est pas PETIT , on alarme l'utilisateur en précisant que le type de DEFORMATION choisi est incompatible avec SUPPORT=POINT , et que l'on utilise donc SUPPORT=ELEMENT, sauf dans le cas où on fournit toutes les composantes du gradient de transformation (mot clé GRAD_IMPOSE).

5 Exemple d'utilisation

Cet exemple est issu du test SSNV160E :

```
# TITRE CAS TEST HYDROSTATIQUE CAM_CLAY EN 3D AVEC SIMU_POINT_MAT

#  CARACTERISTIQUES DU MATERIAU
MATER=DEFI_MATERIAU(ELAS=_F(E=7.74E6,NU=0.285),
                    CAM_CLAY=_F(MU  = 6.E6,
                           PORO=0.66,
                           LAMBDA=0.25,
                           KAPA=0.05,
                           M=0.9,
                           PRES_CRIT=3.E5),);

#  CHARGEMENT
PRESS2=DEFI_FONCTION(NOM_PARA='INST', NOM_RESU='PRESSION',
            VALE=(0.0,0.0,
           100.0,-100000.0,
           600.0,-320000.0,
           1000.0,-350000.0,
           5000.0,-500000.0,
           8000.0,-800000.0),
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

```
                PROL_DROITE='CONSTANT');

    #  LISTE DES INSTANTS DE CALCUL
    LI1=DEFI_LIST_REEL(DEBUT=0.0,
           INTERVALLE=(_F(JUSQU_A=1000.0,NOMBRE=10,),
                       _F(JUSQU_A=1.E4,NOMBRE=60,),),);

    SXXINI= -7.99000E+05
    EXXINI= -1.82689E-02

    RESU3=SIMU_POINT_MAT(
    COMPORTEMENT =_F(RELATION='CAM_CLAY',ITER_INTE_MAXI=100,ITER_INTE_PAS=-10,),
              NEWTON=_F(MATRICE='TANGENTE',REAC_ITER=1,),
              CONVERGENCE=_F(ITER_GLOB_MAXI=20,),
              MATER      = MATER,
              INCREMENT=_F(LIST_INST=LI1,INST_INIT= 7990.,INST_FIN = 8000.),
              SIGM_INIT=_F(SIXX=SXXINI,SIYY=SXXINI,SIZZ=SXXINI,),
              EPSI_INIT=_F( EPXX=EXXINI,EPYY=EXXINI,EPZZ=EXXINI,
                            EPXY=0.,EPYZ=0.,EPXZ=0.,),
              VARI_INIT=_F( VALE=( 3.99500E+05,1.0,7.99000E+05,4.63066E-10,
                                   1.94773E-02,2.99086E-17,1.79821E+00),),
              SIGM_IMPOSE=_F(  SIXX=PRESS2, SIYY=PRESS2, SIZZ=PRESS2,),
              );
    IMPR_TABLE(TABLE=RESU3)
```

La table résultat contient :

```
#------------------------------------------------------------------------------
#
#CALC_POINT_MAT
 INST          EPXX          EPYY          ... SIXX          SIYY          ... TRACE         ... V1            V2
..
  0.00000E+00  0.00000E+00  0.00000E+00    -1.00000E+05 -1.00000E+05     0.00000E+00     0.00000E+00  0.00000E+00
  2.00000E+02 -2.06631E-03 -2.06631E-03    -1.44000E+05 -1.44000E+05    -4.32000E+05     3.00000E+05  0.00000E+00
  3.00000E+02 -3.57721E-03 -3.57721E-03    -1.88000E+05 -1.88000E+05    -5.64000E+05     3.00000E+05  0.00000E+00
  4.00000E+02 -4.76888E-03 -4.76888E-03    -2.32000E+05 -2.32000E+05    -6.96000E+05     3.00000E+05  0.00000E+00
  5.00000E+02 -5.75297E-03 -5.75297E-03    -2.76000E+05 -2.76000E+05    -8.28000E+05     3.00000E+05  0.00000E+00
  6.00000E+02 -6.59119E-03 -6.59119E-03    -3.20000E+05 -3.20000E+05    -9.60000E+05     3.00000E+05  0.00000E+00
  7.00000E+02 -6.72247E-03 -6.72247E-03    -3.27500E+05 -3.27500E+05    -9.82500E+05     3.00000E+05  0.00000E+00
  8.00000E+02 -6.85078E-03 -6.85078E-03    -3.35000E+05 -3.35000E+05    -1.00500E+06     3.00000E+05  0.00000E+00
  9.00000E+02 -6.97624E-03 -6.97624E-03    -3.42500E+05 -3.42500E+05    -1.02750E+06     3.00000E+05  0.00000E+00
  1.00000E+03 -7.09899E-03 -7.09899E-03    -3.50000E+05 -3.50000E+05    -1.05000E+06     3.00000E+05  0.00000E+00
  1.40000E+03 -7.33679E-03 -7.33679E-03    -3.65000E+05 -3.65000E+05    -1.09500E+06     3.00000E+05  0.00000E+00
  1.80000E+03 -7.56501E-03 -7.56501E-03    -3.80000E+05 -3.80000E+05    -1.14000E+06     3.00000E+05  0.00000E+00
 .......
```


Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)