Macro order SIMU_POINT_MAT

1 Goal

To calculate the mechanical evolution of a material point, into quasi-static nonlinear.

All behaviors available in `STAT_NON_LINE` [U4.51.11] are it also here.

The goal of this macro-order is to simplify to the maximum the data: it is enough to provide:
1) The behavior and the material;
2) Functions defining the evolution of the components of constraints or selected deformations;
3) Discretization in time.

This makes it possible in particular to in the case of calculate the evolution of the tensor of the constraints imposed deformations, or the reverse (current cases in identification of parameters material)

Product a structure of data of the type `table` container, according to time, the evolution of all the components of the tensors of constraints and deformations, as well as the internal variables.

2 Syntax

```
tabres [table] = SIMU_POINT_MAT (

♦  /   BEHAVIOR     =_F (see the document [U4.51.11]),

♦ MATER             =   / chechmate,  [to subdue]
                        / l_mat, [l_mater]

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

♦  INCREMENT       =_F ( to see the document [U4.51.03]),

◊  NEWTON          =_F (  to see the document [U4.51.03]),

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

             ◊  MODELING      =    /'3D'        [DEFECT]
                                      /'C_PLAN'
                                      /'D_PLAN'

             ◊  RECH_LINEAIRE   =_F (  to see the document [U4.51.03]),

             ◊  FILING       =_F (  to see the document [U4.51.03]),

              # variable of orders function scalars of time

             ◊  SUIVI_DDL       =_F (  to see the document [U4.51.03]),

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

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

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


      ◊   SUPPORT=   /'NOT'                          [DEFECT]
                    ◊   NB_VARI_TABLE = nvar  [I]
                    ◊   FORMAT_TABLE  = /'CMP_COLONNE'   [DEFECT]
                                       /'CMP_LIGNE'
                    ◊   OPER_TANGENT  = /'NOT'           [DEFECT]
                                       /'YES'
                    ◊   FILING =_F   (
                          ◊   LIST_INST = linst (see document [U4.51.03]),
                          ◊   PRECISION = prec  ),
                    /   ◊   SIGM_IMPOSE=_F ( ◊  SIXX = sigxx        [function]
                                           ◊  SIYY = ifgyy        [function]
                                           ◊  SIZZ = sigzz        [function]
                                           ◊  SIXY = sigxy        [function]
                                           ◊  SIXZ = sigxz        [function]
                                           ◊  SIYZ = sigyz        [function]),
                        ◊    EPSI_IMPOSE=_F (    ◊  EPXX = epsxx
              [function]
                                           ◊  EPYY = epsyy        [function]
                                           ◊  EPZZ = epszz        [function]
                                           ◊  EPXY = epsxy        [function]
                                           ◊  EPXZ = epsxz        [function]
                                           ◊  EPYZ = epsyz        [function]),
                    /   ◊   GRAD_IMPOSE=_F ( ◊  F11 = fonc         [function]
                                           ◊  F12 = fonc         [function]
                                           ◊  F12 = fonc         [function]
                                           ◊  F13 = fonc         [function]
                                           ◊  F21 = fonc         [function]
                                           ◊  F22 = fonc         [function]
                                           ◊  F23 = fonc         [function]
                                           ◊  F31 = fonc         [function]
                                           ◊  F32 = fonc         [function]
                                           ◊  F33 = fonc         [function]),
                    /   ◊   MATR_C1=_F ( ♦  VALE = cij              [R]
                                       ♦  NUME_LIGNE = numlig      [I]
                                       ♦  NUME_COLONNEE = numcol    [I]
                                      ),
                        ◊   MATR_C2=_F ( ♦  VALE = cij              [R]
                                       ♦  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     [ function]
                                ◊ 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] ),
    ◊   INFORMATION =    / 1,                       [DEFECT]
             / 2 , );
```

3 Operands

3.1 Operand MATER

♦ **MATER** =/chechmate, [to subdue]
/ l_mat, [l_mater]

This keyword makes it possible to inform the name of material (to subdue) defined by DEFI_MATERIAU [U4.43.01], where are provided the parameters necessary to the selected behavior. In the case of the polycrystals, one can have to give several materials (cf ssnv194c).

3.2 Word-key BEHAVIOR

The syntax of this keyword is described in the document [U4.51.11].

3.3 Words keys INCREMENT/FILING NEWTON

The syntax of these keywords is described in the document [U4.51.03].

The keyword INCREMENT defines the time intervals taken in the incremental method.

The keyword FILING defines the moments when the results in the table are stored tabres. In the case SUPPORT=' POINT', these moments can be defined only by the keyword LIST_INST with the relative precision PRECISION.

Keywords NEWTON, Facultatif, allows to modify the values by default of the parameters of convergence of the method of Newton.

3.4 Keyword CONVERGENCE

◊ CONVERGENCE =_F ()

If none of the two operands following is present, then all occurs like if: RESI_GLOB_RELA = 1.E-6.

3.4.1 Operand RESI_GLOB_RELA/RESI_GLOB_MAXI

◊ |RESI_GLOB_RELA = resrel , [R]

The algorithm continues the total iterations as long as:

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

where $\boldsymbol{F}^n$ is the residue of the iteration n and $\boldsymbol{L}$ the vector of the imposed loading and the reactions of supports (cf [R5.03.01] for more details).

When the loading and the reactions of support become worthless, i.e. when $\boldsymbol{L}$ is null (for example in the case of a total discharge), one tries to pass from the relative convergence criteria RESI_GLOB_RELA with the absolute convergence criteria RESI_GLOB_MAXI. This operation is transparent for the user (message of alarm emitted in the file .mess). When the vector $\boldsymbol{L}$ becomes again different from zero, one passes by again automatically with the relative convergence criteria RESI_GLOB_RELA.
However, this mechanism of swing cannot function with the first step of time. Indeed, to find a value of RESI_GLOB_MAXI reasonable in an automatic way (since the user did not inform it), one needs to have had at least a step converged on a mode RESI_GLOB_RELA . Consequently, if the loading is null as of the first moment, calculation stops. The user must already then check that the null loading is normal from the point of view of the modeling which it carries out, and if such is the case, to find another convergence criteria (RESI_GLOB_MAXI for example).

If this operand is absent, the test is carried out with the value by default, except if RESI_GLOB_MAXI is present.

◊ |RESI_GLOB_MAXI = resmax , [R]

The algorithm continues the total iterations as long as:

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

where F^n is the residue of the iteration n (Cf [R5.03.01] for more details). If this operand is absent, the test is not carried out.

If RESI_GLOB_RELA and RESI_GLOB_MAXI both are present, the two tests are carried out.

3.4.2 Operand ITER_GLOB_MAXI

```
◊    ITER_GLOB_MAXI =   /10                   [DEFECT]
                        /maglob
```

Maximum iteration count carried out to solve the total problem at every moment (10 by defaults).

3.5 Word key RECH_LINEAIRE

The syntax of these keywords is described in the document [U4.51.03].

The keyword RECH_LINEAIRE allows, in the case SUPPORT=' ELEMENT', to activate linear research to help with the convergence of the algorithm of Newton. This functionality is not available for SUPPORT=' POINT', because it does not seem necessary.

3.6 Word key MODELING

The keyword MODELING allows, in the case SUPPORT=' ELEMENT', to carry out calculation on an element 3D or an element 2D, in plane constraints or plane deformations. It is not available in the case SUPPORT=' POINT', because it is enough to impose a zero value on the components corresponding to the plane constraints or the plane deformations to get the same result.

This keyword makes it possible to define the dimension of with the dealt problem: 3D (by default) or 2D: plane strain or plane stress. In the case 2D, the components of the tensors provided under the keywords SIGM_IMPOSE, EPSI_IMPOSE, SIGM_INIT, EPSI_INIT are 4: XX, YY, ZZ, XY.

3.7 Operand ANGLE

This keyword makes it possible to specify an angle (in degrees) to carry out an overall rotation around Z applied at the same time to the loading, the grid, and the examination. This especially makes it possible to check the reliability of the integration of the behavior, as in the tests **COMP001, COMP002**. By default, rotation is identically worthless.

In the case of materials having an intrinsic orientation (orthotropism, behaviors crystalline), it is advisable to also use the keyword SOLID MASS, with a first value of angle identical to that provided under ANGLE.

3.8 Keyword SOLID MASS

3.8.1 Operands ANGL_EULER/ANGL_REP

These keyword make it possible to lay down an intrinsic orientation in the material (orthotropism, behaviors crystalline), and make it possible to appeal in the macro-order the keyword **SOLID MASS** of **AFFE_CARA_ELEM** [U4.42.01].
By default, the orientation is worthless, and one does not call on **AFFE_CARA_ELEM**.

3.9 Keywords SIGM_INIT/EPSI_INIT/VARI_INIT

These keywords make it possible to define an initial state by the data:
1) components of the initial constraints (all the components are not necessary, by default one takes value 0),
2) components of the initial deformations (if the keyword EPSI_INIT is present, it is necessary to provide all the components of the initial deformations: 4 in 2D, and 6 in 3D)
3) the whole of the initial internal variables for the behavior used.

This functionality is illustrated in test SSNV160E.

3.10 Keywords SIGM_IMPOSE/EPSI_IMPOSE

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

These keyword make it possible to define components of the tensor of constraints imposed on the material point, via functions of time. These functions can be defined using DEFI_FONCTION [U4.31.02] or using FORMULA [U4.31.05].

By defaults, the nonaffected components are identically worthless.

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

These keyword make it possible to define components of the tensor of deformation imposed on the material point, via functions of time. These functions can be defined using DEFI_FONCTION [U4.31.02] or using FORMULA [U4.31.05].

By defaults, the nonaffected components are left without value (not imposed deformation).

It should be noted that, in the case of the model of deformation PETIT_REAC, it is not possible to impose the deformation exactly usingEPSI_IMPOSE. Indeed, because of nature *incremental* this model, deformations obtained by this model at the end of calculation SIMU_POINT_MAT will be different from what will have been imposed, except in small deformations. For the great deformations it thus is preferable to use the model GDEF_LOG who does not suffer from this disadvantage.

3.11 Keywords GRAD_IM POSE

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

These keyword make it possible to define all the components of the tensor gradient of transformation imposed, in great deformations (DEFORMATION=' SIMO_MIEHE') cf test ssnd113).

3.12 Keywords MATR_C1/MATR_C2/VECT_IMPO

These keyword allow, in the case SUPPORT=' POINT', to define the coefficients of the matrices directly $C1$, $C2$ and of the vector g described with the §4.2 : that thus makes it possible to define linear conditions on the unknown factors (forced and deformations of the material point) more general than the components imposed by the mots key SIGM_IMPOSE/EPSI_IMPOSE. All terms of the matrices $C1$ and $C2$ not specified are worthless. For an example of use, to see test WTNV134B [V7.31.134].

3.13 Operand AFFE_VARC

This keyword makes it possible to specify a variable of order (cf [U4.43.03]) whose name is defined under the keyword NOM_VARC ; the function defining the temporal evolution of this variable of order is provided via the keyword VALE_FONC. The possible value of reference vref is given by VALE_REF.

In the case SUPPORT=' ELEMENT', all variables of order are authorized. Moreover, for M_ZIRC (resp. M_ACIER), it is necessary to provide the evolutions of the 4 (resp. 7) metallurgical phases according to time.

In the case SUPPORT=' POINT', only variables of order ' TEMP ',' SECH 'and' IRRA 'are authorized.

3.14 Word key NB_VARI_TABLE

The keyword NB_VARI_TABLE allows, in the case SUPPORT=' POINT', to limit the number of internal variables written in the table. Indeed for the polycrystalline mediums, this one can reach several thousands. One then limits the number of columns of the table to nvar. On the other hand calculations are of course carried out with the totality of the internal variables: those are truncated only in the table as a result.

3.15 Word key FORMAT_TABLE

```
◊  FORMAT_TABLE  =  /'CMP_COLONNE'   [DEFECT]
                    /'CMP_LIGNE'
```

The keyword FORMAT_TABLE allows, in the case SUPPORT=' POINT', to define the mode of storage of the sizes in the table result (the test SSNV194C illustrate these two formats). If the number of internal variables exceeds the maximum of columns authorized for a table (9999, cf D4,02,05), the format rocks automatically in: 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

FORMAT_TABLE  =  /'CMP_COLONNE' :
…  INST         SIZE CMP      VALUE
….
  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 Word key OPER_TANGENT

◊ OPER_TANGENT = /'NOT' [DEFECT]
/'YES'

The keyword OPER_TANGENT allows, in the case SUPPORT=' POINT', to add to the table result the 36 values of the tangent operator resulting from the behavior.

3.17 Operand INFORMATION

◊ **INFORMATION =inf**

Allows to carry out in the file message various intermediate impressions.

4 Operation of the macro_commande

The purpose of this macro_commande is to restrict with bare essential the relative data for a simulation on a material point for a model of incremental behavior.

The inner working thus reduces the command file of the user, by carrying out repetitive operations for this kind of situations.

4.1 Case SUPPORT=' ELEMENT'

Operation is:

1. creation of a grid of only one element at only one point of Gauss (a tetrahedron with four nodes in 3D, a triangle with three nodes in 2D) (see for example [V6.04.176]).
2. assignment of a model 3D or C_PLAN or D_PLAN
3. assignment of material on this grid;
4. assignment of the loadings:
5. with regard to the imposed deformations, for each component affected via one of the keywords of EPSI_IMPOSE , creation of a unit loading in deformation which will be multiplied by the function of time provided for this component by the user;
6. with regard to the imposed constraints, for each component affected via one of the keywords of SIGM_IMPOSE , creation of a unit loading in constraints which will be multiplied by the function of time provided for this component by the user;
7. Call to STAT_NON_LINE . All the keywords having values by default are used, except if they are overloaded by the utilisor (NEWTON, CONVERGENCE, SUIVI_DDL, FILING, RECH_LINEAIRE) and of the initial state.

The whole of the results (six or four components of constraints and deformations, internal variables) are stored in a table (tabres). For each component (column of the table) the evolution according to time appears.

4.2 Case SUPPORT=' POINT'

In this case, plutôt to use a finite element (even single) to carry out calculation, SIMU_POIN_MAT fact call to a dedicated order, CALC_POINT_MAT, which includes in FORTRAN the direct call with the routine 3D of integration of the behaviors, NMCOMP. This is available only Dyears the case of the small deformations.

Let us recall that NMCOMP is the general routine of integration of the laws of behavior, called by all the finite elements 3D and 2D. It allows calculation in a point (this point being the point of integration for a finite element) of the constraints and internal variables the current moment, knowing the internal constraints and variables at the previous moment; and the current increase in deformation. (cf [D5.04.01] and [R5.03.14]). More precisely, at the moment t_i, and with the iteration n the tensor of the constraints $\boldsymbol{\sigma}_i^n$, in a point is calculated from $(\boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1})$ and of the increment of deformation $\Delta \boldsymbol{\varepsilon}_i^n$.

When all the components of the tensor of the deformations are provided, the algorithm is immediate: it is about a simple loop in time, containing for each temporal increment the data of the mechanical state of the preceding increment and the tensor (symmetrical) correpondant with the known increase in deformation.

But in the contrary case, that is to say which one only provides n components of the history of the deformations, $n<6$, that is to say that one provides n components of the history of the constraints, the algorithm is the following:

- by default, any component not specified corresponds to a component of subjugated constraint to remain worthless (condition of Neumann)
- the equations to be solved are (by using the notation in vectorial form of the symmetrical tensors of order 2):
- $\boldsymbol{\sigma}_\mathbf{i} = F(\Delta \boldsymbol{\varepsilon}_\mathbf{i} ; \sigma_{i-1}, \alpha_{i-1})$ where F represent the result of the integration of the behavior by NMCOMP
- for J varying from 1 to 6:
  - that is to say $(\sigma_i)_j = g_j(t_i)$
  - that is to say $(\varepsilon_i)_j = g_j(t_i)$
    - where $\bar{\sigma}_j(t)$ and $\bar{\varepsilon}_j(t)$ are given by SIGM_IMPOSE/EPSI_IMPOSE.

This can be still written:

for each moment t_i, to solve:

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

who is a nonlinear system of order 12.

The last relation translates the condtitions of constraints or imposed deformations: matrices C_1 and C_2 only terms contain on the diagonal, being worth 1 if the corresponding component is imposed, knowing that one cannot have at the same time forced and imposed deformation.

For example, if deformation ε_{yy} is imposed, the last relation is written:

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

If the user specified (via the keywords `MATR_C1, MATR_C2 and VECT_IMPO`) additional relations, those are taken into account directly in the matrices C_1 and $C_{2.}$

The resolution of this system of nonlinear equations is carried out by a method of Newton:

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

–iteration N has:

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

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

with

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

where $\left(\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}}\right)_i^0$ represent the tangent operator of prediction (option `RIGI_MECA_TANG` , cf [R5.03.01, R5.03.02]) and $\left(\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}}\right)_i^n$ represent the coherent tangent operator (option `FULL_MECA)` , cf [R5.03.01, R5.03.02]). These operators can be replaced by the operator of elasticity according to the keywords `PREDICTION, REAC_ITER`.

The only not linearity of the problem comes from the behavior: $F(\Delta \boldsymbol{\varepsilon}_\mathbf{i}^\mathbf{n}; \boldsymbol{\sigma}_{\mathbf{i-1}}, \alpha_{i-1})$.

In the case of a linear behavior, one checks that the solution of the problem is obtained at the conclusion of phase of prediction.

The convergence of the iterations is vérfiée:

- maybe in relative value, (keyword `RESI_GLOB_RELA)` :

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

DaNS this case, the terms in constraints and deformations are separate for the examination of the convergence criteria to avoid the problems due to the differences in orders of gandor.

- maybe in absolute value (keyword RESI_GLOB_MAXI) or vavlor of the denominator close to zero in the relative criterion above:

$$\max_{j=1,12} \left|(R_i^n)_j\right| < \text{RESI\_GLOB\_MAXI}$$

The options of calculation of tangent rigidity by disturbance and the automatic management of the step of time are also activated, as in [U4.51.03].

In the preceding resolution, the terms in constraints are adimensionnalisés, to avoid a bad conditioning of the matrix jacobienne. One thus divides for the resolution all the terms into constraints by the max of the diagonal terms of the operator of elasticity; it is thus necessary to provide in DEFI_MATERIAU the keyword ELAS or ELAS_ORTH or ELAS_ISTR.

In the case SUPPORT=' POINT', certain keywords do not have utility:

- in the case of linear research, this one is not programmed in the current version
- in the case of filing, only the keyword LIST_INST is taken into account
- in the case CONVERGENCE/RESI_REFE_RELA , this keyword without object: the residue by value of reference does not have a direction for a material point.
- If the value of the keyword DEFORMATION is not SMALL , one alarms the user by specifying that the type of DEFORMATION chosen is incompatible with SUPPORT=POINT , and that one thus uses SUPPORT=ELEMENT, except if one provides all the components of the gradient of transformation (keyword GRAD_IMPOSE).

5 Example of use

This example is resulting from the test SSNV160E :

```
# TITRATES CA HYDROSTATIC TEST CAM_CLAY IN 3D WITH SIMU_POINT_MAT

#  CHARACTERISTICS OF MATERIAL
MATER=DEFI_MATERIAU (ELAS=_F (E=7.74E6, NU=0.285),
                    CAM_CLAY=_F (DRIVEN  = 6.E6,
                           PORO=0.66,
                           LAMBDA=0.25,
                           KAPA=0.05,
                           M=0.9,
                           PRES_CRIT=3.E5),);

#  LOADING
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),
          PROL_DROITE=' CONSTANT');

#  LIST OF THE MOMENTS OF CALCULATION
LI1=DEFI_LISTE_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 (
BEHAVIOR =_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)
```

The table result contains:

```
#--------------------------------------------------------------------------------
#
#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
 .......
```