Structure of the crystalline behaviors

Summarized:

This document describes structure of the integration of the crystalline behaviors (cf R5.03.11) and the actions to be undertaken to add a new crystalline behavior, with an aim of carrying out the computations of aggregates, or the computations homogenized using STAT_NON_LINE and SIMU_POINT_MAT.

Note: it is also possible to use the crystalline constitutive laws available in the modulus *Zmat* (cf [U2.10.01]).

Contents

1 Description general of the crystalline behaviors

a crystalline behavior uses, besides DEFI_MATERIAU, DEFI_COMPOR.

The name of the behavior in COMP_INCR is generic: MONOCRISTAL or POLYCRISTAL.

The algorithm of resolution is with the choice:
- for the MONOCRISTAL: NEWTON, NEWTON_RELI, NEWTON_PERT and RUNGE_KUTTA
- for the POLYCRISTAL: RUNGE_KUTTA

One describes in this document:
- the architecture of DEFI_COMPOR
- the architecture of the explicit resolution with error controlled by Runge_Kutta (cf [R5.03.14])
- architecture of the implicit resolution by Newton and its alternatives (environment PLASTI cf [R5.03.14])

to apprehend this document, the reading of R5.03.11 is highly advised.

2 Structure of DEFI_COMPOR

2.1 Illustration on an example: test SSNV194

Let us start from an example (test ssnv194).

2.1.1 Modelization a: a small aggregate comprising 10 grains:

```
ACIER=DEFI_MATERIAU (ELAS=_F (E=145200.0, NU=0.3,),
                     MONO_VISC1=_F (N=10.0, K=40.0, C=1.0,),
                     MONO_ISOT1=_F (R_0=75.5, Q=9.77, B=19.34),
                     MONO_CINE1=_F (D=36.68,),);

MONO1 =DEFI_COMPOR (MONOCRISTAL= (_F (  MATER=ACIER, ELAS=' ELAS",
                                    ECOULEMENT=' MONO_VISC1',
                                    ECRO_ISOT=' MONO_ISOT1',
                                    ECRO_CINE=' MONO_CINE1',
                                    FAMI_SYST_GLIS=' BCC24',),),);

ORIEN=AFFE_CARA_ELEM (MODELE=TROISD, MASSIF= (
_F (GROUP_MA=' GM1', ANGL_EULER= (- 150.646, 33.864, 55.646,),),
_F (GROUP_MA=' GM2', ANGL_EULER= (- 137.138, 41.5917, 142.138,),),
_F (GROUP_MA=' GM3', ANGL_EULER= (- 166.271, 35.46958, 171.271,),),
_F (GROUP_MA=' GM4', ANGL_EULER= (- 77.676, 15.61819, 154.676,),),
_F (GROUP_MA=' GM5', ANGL_EULER= (- 78.6463, 33.864, 155.646,),),
_F (GROUP_MA=' GM6', ANGL_EULER= (- 65.1378, 41.5917, 142.138,),),
_F (GROUP_MA=' GM7', ANGL_EULER=(- 94.2711, 35.46958, 71.271,),),
_F (GROUP_MA=' GM8', ANGL_EULER= (- 5.67599, 15.61819, 154.676,),),
_F (GROUP_MA=' GM9', ANGL_EULER= (- 6.64634, 33.864, 155.646,),),
_F (GROUP_MA=' GM10', ANGL_EULER= (6.86224, 41.5917, 142.138,),),
                                  ),);

SOLNL=STAT_NON_LINE (MODELE=…, CHAM_MATER=…, EXCIT=_…,
                     CARA_ELEM=ORIEN,
                     COMP_INCR=_F (RELATION=' MONOCRISTAL',
                                   COMPOR=MONO1,
                                   ),)
```

2.1.2 Modelization C: polycrystal comprising 10 grains:

A material point, 10 grains of identical fractions voluminal (0.1), and directional senses similar to those of the modelization A (what makes it possible to find the same average solution):

```
MONO1=DEFI_COMPOR (MONOCRISTAL= (… identical to modelization A)

COMPORP=DEFI_COMPOR (POLYCRISTAL= (
_F  (MONOCRISTAL=COMPORT,  FRAC_VOL=0.1,  ANGL_EULER=  (-  150.646,  33.864,
55.646,),),
_F  (MONOCRISTAL=MONO1,  FRAC_VOL=0.1,  ANGL_EULER=  (-  137.138,  41.5917,
142.138,),),
_F  (MONOCRISTAL=MONO1,  FRAC_VOL=0.1,  ANGL_EULER=  (-  166.271,  35.46958,
171.271,),),
_F  (MONOCRISTAL=MONO1,  FRAC_VOL=0.1,  ANGL_EULER=  (-  77.676,  15.61819,
154.676,),),
_F  (MONOCRISTAL=MONO1,  FRAC_VOL=0.1,  ANGL_EULER=  (-  78.6463,  33.864,
155.646,),),
_F  (MONOCRISTAL=MONO1,  FRAC_VOL=0.1,  ANGL_EULER=  (-  65.1378,  41.5917,
142.138,),),
_F  (MONOCRISTAL=MONO1,  FRAC_VOL=0.1,  ANGL_EULER=  (-  94.2711,  35.46958,
71.271,),),
_F  (MONOCRISTAL=MONO1,  FRAC_VOL=0.1,  ANGL_EULER=  (-  5.67599,  15.61819,
154.676,),),
_F  (MONOCRISTAL=MONO1,  FRAC_VOL=0.1,  ANGL_EULER=  (-  6.64634,  33.864,
155.646,),),
_F  (MONOCRISTAL=MONO1,  FRAC_VOL=0.1,  ANGL_EULER=  (6.86224,  41.5917,
142.138,),),
),
LOCALIZATION  = ' BETA', DL=0., DA=0.,);

SOLNL=SIMU_POINT_MAT (COMP_INCR=_F (RELATION=' POLYCRISTAL',
                                    COMPOR=COMPORP,
                                    ALGO_INTE=' RUNGE_KUTTA',),
                      NEWTON=_F (MATRICE=' ELASTIQUE', REAC_ITER=0),
                      MATER =…, NB_VARI_TABLE=6, INCREMENT=…, EPSI_IMPOSE=…
                            );
```

2.2 Description of DEFI_COMPOR

2.2.1 Structure:

The purpose of routine OP0050 is producing data structure described in [D4.06.24]: the objects composing this SD are different according to whether one treats a monocrystal (routine OP5901) or a polycrystal (routine OP5902).

2.2.2 Addition of a crystalline behavior to the catalog of DEFI_MATERIAU/DEFI_COMPOR

If the new crystalline behavior uses material parameters different from those which are already available in key words MONO_* of DEFI_MATERIAU it is enough to introduce these new parameters of behavior, either under only one key word (case of behaviors MONO_DD_*, or by separating the coefficients relating to isotropic hardening, kinematic hardening, and flow (cf MONO_ISOT*, MONO_CINE*, MONO_VISC*,). These parameters will be exploited in integration (routines LCMMAT, LCMMAP), and the key word corresponding factors will be used in DEFI_COMPOR.

Example:

```
MONO_DD_CFC =FACT (statut=' f',
regles= ( UN_PARMI ("H", "H1"),
          PRESENT_PRESENT ("H1", "H2", "H3", "H4", "H5"),
          PRESENT_ABSENT ("H", "H1", "H2", "H3", "H4", "H5"),),

GAMMA0 =SIMP (statut=' f', typ=' R', defaut=0.001, units: S ** - 1"),
TAU_F =SIMP (statut=' o', typ=' R', fr= " in unit of stresses ex 20 MPa"),
A =SIMP (statut=' f', typ=' R', defaut=0.13, fr= " parameter A, without
unit"),
B =SIMP (statut=' f', typ=' R', defaut=0.005, fr= " parameter B, without
unit"),
N =SIMP (statut=' f', typ=' R', defaut=200., fr= " parameter N, without
unit"),
Y =SIMP (statut=' o', typ=' R', fr= " in unit of length ex 2.5 A"),
ALPHA=SIMP (statut=' f', typ=' R', defaut=0.35, fr= " parameter alpha"),
BETA =SIMP (statut=' o', typ=' R', fr= " parameter B, in unit of length"),
…),
```

This makes it possible to describe each coefficient, its optional character (with a possible value by default) or compulsory (for more precise details, to refer to [D5,01,01]).

The catalog of DEFI_COMPOR is, according to the cases:

```
MONOCRISTAL =FACT (statut=' f', max=5,
    MATER =SIMP (statut=' o', typ=mater_sdaster, max=1),
    ECOULEMENT=SIMP (statut=' o', typ=' TXM', into= ("MONO_VISC1",
    "MONO_VISC2",
                                                "MONO_DD_CFC", "MONO_DD_CC",…),
                                        viscoplastic flow fr= " standard"),
    ELAS =SIMP (statut=' f', typ=' TXM',),

    # case of a behavior of the type MONO_VISC*
    b_non_dd=BLOC (condition= " ECOULEMENT==' MONO_VISC1'
                           gold ECOULEMENT==' MONO_VISC2',
                   ECRO_ISOT=SIMP (statut=' f', typ=' TXM', max=1,
                      fr= " To give the isotropic type of hardening"),
                   ECRO_CINE=SIMP (statut=' f', typ=' TXM', max=1,
                      fr= " To give standard kinematic hardening"),
                   FAMI_SYST_GLIS=SIMP (statut=' f', typ=' TXM',
                      into= ("OCTAEDRIQUE", "BCC24", "CUBIQUE1", "CUBIQUE2",
                            "ZIRCONIUM", "UNIAXIAL", "UTILISATEUR"),),
                   b_util =BLOC (condition= " FAMI_SYST_GLIS==' UTILISATEUR'
                   ",
                    TABL_SYST_GLIS =SIMP (statut=' f', typ=table_sdaster,),),),),

    # case of a behavior of the type DD
    b_dd_cc=BLOC (condition= " ECOULEMENT==' MONO_DD_CC'",
                   FAMI_SYST_GLIS=SIMP   (statut='   f',   typ='   TXM',   into=
                   ("CUBIQUE1", "UTILISATEUR",),
                   b_util=BLOC (condition= " FAMI_SYST_GLIS==' UTILISATEUR'",
                   TABL_SYST_GLIS=SIMP (statut=' f', typ=table_sdaster),),),),),

MATR_INTER =SIMP (statut=' f', typ=table_sdaster, max=1,),

ROTA_RESEAU=SIMP  (statut='  f',  typ='  TXM',  max=1,  into=  ("NON",  "POST",
"CALC"),
                   defaut='NON', fr= " rotation of network: NON, POST, CALC"),
```

```
POLYCRISTAL =FACT (statut=' f', max=' ** ",
              regles= (UN_PARMI ("ANGL_REP", "ANGL_EULER"),),

   MONOCRISTAL=SIMP (statut=' O", typ=compor_sdaster, max=1),
   FRAC_VOL  =SIMP (statut=' o', typ=' R', fr= " voluminal fraction "),
   ANGL_REP=SIMP (statut=' f', typ=' R', max=3, fr= " nautical angles in
   degrees"),
   ANGL_EULER=SIMP (statut=' f', typ=' R', max=3, fr= " Eulerian angles in
   degrees"),

LOCALIZATION=SIMP (statut=' f', typ=' TXM', max=1, into= ("BZ", "BETA"),
   fr= " To give the name of the rule of localization"),
   b_beta      =BLOC (condition = "LOCALIZATION==' BETA'",
                DL=SIMP (statut=' o', typ=' R', max=1),
                DA=SIMP (statut=' o', typ=' R', max=1),)
```

the architecture of routines OP5901 and OP5902 is simple, and consists in filling the data structure sd_compor, intended to prepare computations. For that, several information is deduced from the data of the user:

- For the monocrystal:
  - the number of sliding systems, is by calling on the routine **LCMMSG,** which defines sliding system families pre-established, that is to say by reading the array provided for each family (which itself is stored in the sd_compor)
  - the number of local variables which results amongst systems from slidings total, which will be associated with behavior MONOCRISTAL in COMP_INCR.
- For the polycrystal:
  - the various monocrystals relative to each grain, with the voluminal fraction and the directional sense
  - regulates it localization and its parameters.
  - The total number of local variables, deduced from the monocrystals and amongst grains, which will be associated with behavior POLYCRISTAL in COMP_INCR.

The addition of a crystalline behavior is thus reduced, in DEFI_COMPOR, with the modification of the catalog for part MONOCRISTAL (for the syntactic checking). The addition of a family of sliding systems also does not represent a modification of the catalog of DEFI_COMPOR, with possible blocks to manage the possibilities of association between flow models and sliding system families.
For part POLYCRISTAL, the addition of a crystalline behavior does not modify the catalog of DEFI_COMPOR, One of the only modifications would consist of the addition of a rule of localization.

3 Structure of integration

3.1 Recovery of the coefficients material

Some is the selected type of integration (implicit or clarifies) the recovery of the characteristics material and behavior, resulting from DEFI_MATERIAU and DEFI_COMPOR, is done via routine LCMMAT, called by LCMATE, for the MONOCRISTAL , and by routine LCMMAP, also called by LCMATE, for the POLYCRISTAL.

These routines have several functions:

- Recovery of the values of the key word defining the parameters, mainly using the general routine RCVALB, and storage in two tables (different only if the coefficients depend on the temperature): MATERD defining the parameters in previous time, i.e. the beginning of time step, and MATERF at current time, therefore at the end of time step). These tables make it possible to pass the material parameters to the routines of resolution;

- Reading of the sd_compor and information storage (sliding system families, matrix of interaction,…) in tables used during the resolution.

Structure of LCMMAT:

LCMMJV : reading of the sd_compor resulting from DEFI_COMPOR
for each family of systems:
LCMMSG provides the number of LCMMJS
sliding systems (if it is about a family "user")
LCMAFL recovers the coefficients material relating to flow
LCMHSR+LCMHDD : specific call in this routine for MONO_DD_KR
LCMAEC coefficients material relating to kinematic hardening,
LCMAEI coefficients material relating to isotropic hardening,
LCMHSR: computation or reading of the matrix of interaction
DMAT3D, D1MA3D : operator of elasticity and his reverse
CALCMM LCMMSG. : computation and storage of the tensors of directional sense of all the systems, to optimize the performances.

In the case of a new single-crystal behavior, it is enough a priori to intervene in routines LCMAFL, LCMAEI and possibly LCMAEC, by adding in each one of these routines the block of instructions necessary to the recovery of the coefficients material of this behavior.

It is also advisable to allot a number to him: indeed, to optimize the performances, it is preferable to read and compare integers rather than these character strings; in the routines of integration, rather than to test:

```
IF (NECOUL.EQ. "MONO_VISC1") THEN.
```

one will test:

```
IF (NUCOUL.EQ.1) THEN….
```

The nomenclature of the numbers of flow models is defined in LCMAFL :

Name of the flow model	Number associated
MONO_VISC1	1
MONO_VISC2	2
MONO_DD_KR	4
MONO_DD_CC (athermic)	5
MONO_DD_CFC	5
MONO_DD_FAT	6

Table 3.1-1

numbers with models of isotropic hardening are defined in LCMAEI :

Name of the flow model	Number associated
MONO_ISOT1	1
MONO_ISOT2	2
MONO_DD_CFC	3
MONO_DD_CC (athermic)	3
MONO_DD_FAT	4

Table 3.1-2

numbers with models of isotropic hardening are defined in LCMAEC :

Name of the flow model	associated Number
MONO_CINE1	1
MONO_CINE2	2

Table 3.1-3

Structures of LCMMAP:
Reading of the sd_compor of type polycrystal

For each behavior monocrystal (5 at the most) used by all the grains
for each family of systems: reading of the characteristics as LCMMAT
LCMMSG provides the number of LCMAFL
sliding systems recovers the coefficients material relating to flow
LCMAEC coefficients material relating to kinematic hardening,
LCMAEI coefficients material relating to isotropic hardening, stamps interaction
DMAT3D, D1MA3D : operator of elasticity and his reverse
Storage of the relative information to the monocrystals used for each phase in specific tables. Right before the call to the resolution clarifies by Runge_Kutta (routine GERPAS), call to specific routine CALCMS allowing to store in a single table the sliding systems relative ones to all the grains, to optimize the performances.

3.2 Explicit integration – diagram of RUNGE - KUTTA

3.2.1 Case of the monocrystal:

It is the fastest way (but the least optimal) to introduce a new single-crystal behavior in small strains: it is enough to write derivatives of the local variables in routine LCMMON, called by RDIF01.

The purpose of routine LCMMON is calculating derivatives of the local variables. One must solve a system of $6+3*n_s$ differential equations, standard:

- $\dot{\varepsilon}^{vp}=\sum_s \mu_s \dot{\gamma}_s$ 6 equations
- for each system of sliding (on all the families of systems) 3 relations:
  - $\dot{\gamma}_s=\dot{p}_s\left(\tau_s(\sigma),\alpha_s,\gamma_s,R_s(p)\right)\eta(\tau_s,\alpha_s)$ where $\eta(\tau_s,\alpha_s)=\pm 1$
  - $\dot{\alpha}_s=h\left(\tau_s,\alpha_s,\gamma_s,p_s\right)$
  - $R_s(p)$

where $\tau_s=\mu_s:\sigma$ where σ is deduced from the relation: $\sigma=\Lambda\left(\varepsilon-\varepsilon^{vp}\right)$

In practice, the resolution is carried out in the following way:

Routine LCMMON and the calculates the stresses by the relation of elasticity (isotropic or orthotropic

) CAL CALSIG (...) what provides the tensor SIG σ

= Then it calculates $6+3*n_s$ derivatives resulting from the differential equations above, stores in a table DVIN:

- Buckle on sliding system families:

```
C IFA=1, NBFSYS
   …
    Recovery amongst sliding systems
    CAL LCMMSG (NOMFAM, NBSYS, 0, PGL, ms, NG, LG, 0, Q)

    Buckles on sliding systems of family IFA:
    C IS=1, NBSYS

       CAL LCMMSG (.)    computation of the tensor of MuS directional sense

       CALCUL OF REDUCED CISSION
       TAUS= SIG (I) *MuS (I) for i=1,6

        CAL LCMMFI (=> RP)      routine of computation of isotropic hardening

        CAL LCMMFE (=> DGAMMA, DP)        routine of computation of flow

        CAL LCMMEC (=> DALPHA) routine of computation of kinematic hardening

        Computation of the total viscoplastic strain
       C ITENS=1,6
          DEVI (ITENS) =DEVI (ITENS) +MuS (ITENS) *DGAMMA
       ENDDO

       storage of derivatives of the local variables for the system of sliding IS
       DVIN (NUVI-2) =DALPHA
       DVIN (NUVI-1) =DGAMMA
       DVIN (NUVI  ) =DP

    ENDDO

ENDDO

storage of tensor derived from the viscoplastic strain.
C ITENS=1,6
    DVIN (ITENS) = DEVI (ITENS)
ENDDO
```

the algorithm of Runge-Kutta then manages the integration of these differential equations, by controlling the error, and while refining time step until obtaining an error lower than the required accuracy (RESI_INTE_RELA) [R5.03.14].

A priori, the structure of routine LCMMON, should not evolve at the time of the addition of a new single-crystal behavior; only routines LCMMFI, LCMMFE and LCMMEC are to be modified.

They are built in the following way:

```
LCMMFI
C----------------------------------------------------------------------
C     FOR NEW A TYPE OF ISOTROPIC HARDENING, TO ADD A BLOCK IF
C----------------------------------------------------------------------
C     IF (NECRIS.EQ. "MONO_ISOT1") THEN
      IF (NUEISO.EQ.1) THEN
       ......
            RP=…
C     ELSEIF (NECRIS.EQ. "MONO_ISOT2") THEN
      ELSEIF (NUEISO.EQ.2) THEN
      ............
```

```
            RP=…

C       ELSEIF (NECRIS.EQ. "MONO_DD_CFC") THEN
        ELSEIF (NUEISO.EQ.3) THEN
            ..........
            RP=MU*SQRT (RP) *CEFF
        ENDIF
```

Note : one uses here the numbers associated with each type of behavior rather than the names, to optimize the performances.

In the same way the structure of routine LCMMFC is :

```
C-----------------------------------------------------------------------
C   FOR NEW A TYPE OF KINEMATIC HARDENING, TO ADD A BLOCK IF
C-----------------------------------------------------------------------
C       IF (NECRCI.EQ. "MONO_CINE1") THEN
        IF (NUECIN.EQ.1) THEN
            DALPHA=…
C       ELSEIF (NECRCI.EQ. "MONO_CINE2") THEN
        ELSEIF (NUECIN.EQ.2) THEN
        ENDIF
```

And in a similar way, the structure of routine LCMMFC is :

```
C-------------------------------------------------------------
C       FOR NEW A TYPE OF ECOULEMENT, TO CREATE A BLOCK IF
C-------------------------------------------------------------
C       IF (NECOUL.EQ. "MONO_VISC1") THEN
        IF (NUECOU.EQ.1) THEN
            DP=…
            DGAMMA=…
        ELSEIF (NUECOU.EQ.2) THEN
        ENDIF
```

Note : Routines LCMMFE, LCMMFI, LCMMFC are also used by the explicit integration of the polycrystal and implicit integration. This simplifies the implementation of a new crystalline behavior.

3.2.2 Case of the polycrystal:

The integration of the polycrystal rests largely on that of the monocrystal: there one calculates still derivatives of the local variables for each monocrystal of each grain g , in routine LCMMOP, called by RDIF01.

One must solve a system of $6+n_g\left(6+3*n_s(g)\right)$ differential equations, standard:

•for each grain defined by a directional sense and a proportion f_g , a relation of localization of the stresses, general form:

$$\boldsymbol{\sigma_g}=L\left(\boldsymbol{\Sigma},\boldsymbol{E^{vp}},\varepsilon_g^{vp},\beta_g\right) \quad \text{with,} \quad \boldsymbol{\Sigma}=\boldsymbol{\Lambda}(\boldsymbol{\Lambda}_{-}^{-1})\boldsymbol{\Sigma}^{-}+\boldsymbol{\Lambda}(\Delta\boldsymbol{E}-\Delta\boldsymbol{E}^{th}-\Delta\boldsymbol{E}^{vp})$$

•for each of $n_s(g)$ sliding systems of each grain g , 3 relations:

- $\dot{\gamma}_s=\dot{p}_s\left(\tau_s(\boldsymbol{\sigma}),\alpha_s,\gamma_s,R_s(p)\right)\eta(\tau_s,\alpha_s)$ where $\eta(\tau_s,\alpha_s)=\pm1$
- $\dot{\alpha}_s=h\left(\tau_s,\alpha_s,\gamma_s,p_s\right)$
- $R_s(p)$

•the scale of the grain has, the equations of homogenization: $\dot{\boldsymbol{\varepsilon}}_g^{vp}=\sum_s \boldsymbol{\mu}_s\dot{\gamma}_s$

In practice, the resolution is carried out in the following way:

Routine LCMMOP and the calculates the stresses by the relation of elasticity (isotropic or orthotropic) CAL CALSIG (...) what provides the tensor SIG $\boldsymbol{\Sigma}$ (

) Then it calculates $6+n_g\left(6+3*n_s(g)\right)$ derivatives resulting from the differential equations above, stores in a table DVIN:

•Buckle on the grains:

```
C IGRAIN=1, NGRAIN
   CAL LCLOCA () relation of localization making it possible to calculate SIGG (σ_g)
```

•Buckles on sliding system families:

```
C IFA=1, NBFSYS
   …
      Recovery amongst sliding systems
      CAL LCMMSG (...)

      Buckles on sliding systems of the IFA family:
      C IS=1, NBSYS

         CAL LCMMSG (.)    computation of the tensor of MuS directional sense

         CALCUL OF REDUCED CISSION
         TAUS= SIGG (I) *MuS (I) for i=1,6

          CAL LCMMFI (=> RP)      routine of computation of isotropic hardening

          CAL LCMMFE (=> DGAMMA, DP)          routine of computation of flow

          CAL LCMMEC (=> DALPHA) routine of computation of kinematic hardening

          Computation of the total viscoplastic strain
         C ITENS=1,6
cDEVG          (ITENS) =DEVG (ITENS) +MuS (ITENS) *DGAMMA
 (DEVG=ε_g^vp)
         ENDDO

         storage of derivatives of the local variables for the system of sliding IS
         DVIN (NUVI-2) =DALPHA
         DVIN (NUVI-1) =DGAMMA
         DVIN (NUVI  ) =DP

      ENDDO

   ENDDO
   homogenization of viscoplastic strains
   C I=1,6
      DEVI (I) =DEVI (I) +FV*DEVG (I)
   ENDDO
storage of the tensor derived from the viscoplastic strain.
C ITENS=1,6
   DVIN (ITENS) = DEVI (ITENS)
ENDDO
```

the seventh local variable contains cumulated equivalent viscoplastic strain

```
DVIN (7) = DVINEQ
```

the algorithm of Runge-Kutta then manages the integration of these differential equations, by controlling the error, and while refining time step until obtaining an error lower than the required accuracy (RESI_INTE_RELA) [R5.03.14].

The structure of routine LCMMOP, should not evolve at the time of the addition of a new single-crystal behavior; only routines LCMMFI, LCMMFE and LCMMEC are to be modified (what is already made in theory for the integration of the monocrystal). An additional modification is to be carried out in routine LCLOCA at the time of the addition of a new rule of localization.

3.3 Implicit integration of the monocrystal by Newton in PLASTI

One integrates this time the single-crystal behavior by a method of Newton. This method is programmed in PLASTI [R5.03.14]. It is thus necessary to provide to this algorithm to write the system of equations to be solved in purely implicit form, in the following way: $\boldsymbol{R}(\boldsymbol{Y})=0$

$$R(Y)=\left\{\begin{array}{l}\boldsymbol{\Lambda}^{-1}\boldsymbol{\Sigma}-(\boldsymbol{\Lambda}_{-}^{-1})\boldsymbol{\Sigma}^{-}-(\Delta\boldsymbol{E}-\Delta\boldsymbol{E}^{\mathrm{th}}-\Delta\boldsymbol{E}^{\mathrm{vp}})\\ \Delta\boldsymbol{E}^{\mathrm{vp}}-\sum_s \boldsymbol{\mu}_s\Delta\gamma_s\\ n_s\left\{\begin{array}{l}\Delta\alpha_s-h(\tau_s^+,\alpha_s^+,\gamma_s^+,p_s^+)\\ \Delta\gamma_s-g(\tau_s^+,\alpha_s^+,\gamma_s^+,p_s^+)\\ \Delta p_s-f(\tau_s^+,\alpha_s^+,\gamma_s^+,p_s^+)\end{array}\right.\end{array}\right. \quad =\mathbf{0}$$

It is a system of $6+6+3n_s$ nonlinear equations (of the same size than that which is integrated by the method of Runge-Kutta into explicit).

In order to optimize the performances, one solves in fact a reduced, of size, $6+n_s$ in the following way built system of equations [R5.03.11]:

- In the form of the 6 components of the tensor of the stresses, $\Delta\boldsymbol{E}^{vp}$ can be expressed according to $\sum_s \boldsymbol{\mu}_s\Delta\gamma_s$ thus 6 equations can be eliminated from the total system to solve.
- Like $\Delta p_s=|\Delta\gamma_s|$, the equation can Δp_s about it be eliminated, and for all the crystalline behaviors currently considered, either $\Delta\alpha_s$ is expressed directly according to $\Delta\gamma_s$, in the case of the kinematic hardening, or $\Delta\gamma_s$ expresses itself according to $\Delta\alpha_s$ which then represents the density of dislocation (except for a factor) for the behaviors of the type MONO_DD_*.

The following system is thus obtained:

- **Small strains:**

$$\begin{array}{l}\boldsymbol{R}_1(\boldsymbol{\sigma},\Delta\beta)=\boldsymbol{\Lambda}^{-1}.\Delta\boldsymbol{\sigma}-\Delta\boldsymbol{\varepsilon}+\Delta\boldsymbol{\varepsilon}^{th}+\sum_s \Delta\gamma_s\boldsymbol{\mu}_s=0\\ \boldsymbol{R}_2(\boldsymbol{\sigma},\Delta\beta)=\Delta\beta_s-k_s(\tau_s(\boldsymbol{\sigma}),\Delta\beta)=0\end{array} \quad \text{where the unknown is:} \quad \boldsymbol{Y}=\left[\begin{array}{c}\boldsymbol{\sigma}\\ \Delta\beta\end{array}\right]$$

with $\tau_s(\boldsymbol{\sigma})=\boldsymbol{\sigma}:\boldsymbol{\mu}_s$

- **Large deformations**

$$\begin{array}{l}\boldsymbol{R}_1(\boldsymbol{S},\Delta\beta)=\boldsymbol{\Lambda}^{-1}.\boldsymbol{S}-\frac{1}{2}\left(\boldsymbol{F}^{\mathrm{eT}}\boldsymbol{F}^{\mathrm{e}}-\boldsymbol{I}_d\right)=0\\ \boldsymbol{R}_2(\boldsymbol{S},\Delta\beta)=\Delta\beta_s-k_s(\tau_s(\boldsymbol{S}),\Delta\beta)=0\end{array} \quad \text{where the unknown is:} \quad \boldsymbol{Y}=\left[\begin{array}{c}\boldsymbol{S}\\ \Delta\beta\end{array}\right]$$

with $\tau_s(\boldsymbol{S})=\left[\left(2\boldsymbol{\Lambda}^{-1}\boldsymbol{S}+\boldsymbol{I_d}\right)\boldsymbol{S}\right]:\boldsymbol{m_s}\otimes\boldsymbol{n_s}$ and $\boldsymbol{F}^e_{n+1}=\Delta\boldsymbol{F}\,\boldsymbol{F}^e_n\left(\Delta\boldsymbol{F}^p(\Delta\gamma_s)\right)^{-1}$

And, according to the behavior considered,

- $\Delta\gamma_s=\Delta p_s(\tau_s,\Delta\beta_s)\xi_s$ and $\xi_s=\frac{\tau_s}{|\tau_s|}$ ou $\frac{\tau_s-f(\alpha)}{|\tau_s-f(\alpha)|}$ corresponds to the sign of flow
- $\Delta\beta_s$ represents either the plastic increment of sliding $\Delta\gamma_s$, for models MONO_VISC* , or the variation of density of dislocations $\Delta\omega_s$ for models MONO_DD_ *

Note: : the extraction of the unknowns of the system starting from the local variables (which remain 3 per system of sliding) is done in routine LCAFYD. Conversely, after the resolution by NEWTON, the computation of the 3 local variables per system of sliding according to the basic variable used in the resolution is done in routine LCPLNF.

The general shape of the algorithm solved by Newton is

$$Y_{k+1}=Y_k-(\frac{dR}{dY_k})^{-1}R(Y_k)$$

It is thus necessary to define the initial values ΔY_0 (0 by default, in routine LCMMIN) , and to calculate the residue Y_k , as well as the jacobian matrix of the system: $\frac{dR}{dY_k}$

3.3.1 General architecture of PLASTI :

```
CAL LCMATE => LCMMAT, identical to RUNGE_KUTTA
```

elastic Prediction
```
   CAL LCELAS
```
Computation of the SEUIL
```
   CAL LCCNVX => routine LCMMVX evaluating of the threshold for MONOCRISTAL.
```

Computation the solution élasto-visco-plastic by the method of Newton:
```
IF (SEUIL .GE. 0.D0) THEN
   CAL LCPLAS/CAL LCPLNL
ENDIF
```

Computation of the tangent operator:
```
IF (OPT .EQ. "RIGI_MECA_TANG" .OR. OPT .EQ. "FULL_MECA") THEN
   CAL LCJPLC => CAL LCMMJP
ENDIF
```

Note : The tangent operator is calculated automatically according to the jacobian matrix of the local system of equations [R5.03.11]. This is carried out into small and large deformations in routine LCMMJP. There is thus a priori nothing to modify for this computation at the time of the addition a new crystalline behavior.

Routine LCCNVX makes it possible to detect if the threshold is crossed for at least a system of sliding. Its structure is the following one:
```
SEUIL=0.D0
C IFA=1, NBFSYS
   C IS=1, NBSYS
```

```
      VISCOPLASTIC CAL
LCMMFI     C ECOULEMENT
      CAL LCMMFE => DP  calculated from the elastic prediction
      IF (DP.GT.0.D0) SEUIL=1.D0
   ENDDO
ENDDO
```

One thus uses same routines LCMMFE and LCMMFI that for explicit and implicit integration.

Routine LCPLNL carries out **the loop of Newton**. Its structure is the following one:

LCPLNL

- LCAFYD (extraction of the local variables useful for the reduced system)
- LCINIT => LCMMIN : initialization of ΔY_0 , 0 by default
- LCRESI => LCMMRE : computation of the residue
  - Is LCJACB => LCMMJA : computation of the jacobian matrix
  - Is (if ALGO_INTE=NEWTON_PERT) LCJACP computation of the jacobian matrix by disturbance, which calls LCRESI
- MGAUSS resolution
- LCRELI searches linear (if ALGO_INTE=NEWTON_RELI), which calls LCRESI…
- LCCONV => LCMMCV convergence criterion
- LCPLNF => LCDPEC computation of all the local variables.

3.3.2 Computation of the residue

routine LCMMRE calculates the residue. Its structure is the following one:

```
C IFA=1, NBFSYS
   C IS=1, NBSYS
      CALTAU computation of  τ_s
      LCMMLC calculates quantities relating to the system of sliding:
         CAL LCMMFI (=> RP)        routine of computation of isotropic hardening
         CAL LCMMFE (=> DGAMMA, DP)        routine of computation of flow
         CAL LCMMEC (=> DALPHA) routine of computation of kinematic hardening
      computation of  k_s  and storage in  R_2(σ, Δβ)=Δβ_s−k_s(τ_s(σ), Δβ)

      In small strains: total viscoplastic strain
      C ITENS=1,6
         DEVI (ITENS) =DEVI (ITENS) +MuS (ITENS) *DGAMMA
      ENDDO
      In large deformations, computation of the terms necessary to
 ΔF^p(Δγ_s)
   ENDDO
ENDDO
```

- in small strains, computation of $\boldsymbol{R_1}(\boldsymbol{\sigma}, \Delta\beta)=\boldsymbol{\Lambda}^{-1}.\Delta\boldsymbol{\sigma}-\Delta\boldsymbol{\varepsilon}+\Delta\boldsymbol{\varepsilon}^{th}+\sum_s \Delta\gamma_s\boldsymbol{\mu}_s=0$
- in large deformations

CALCFE computation of $\boldsymbol{F}^e_{n+1}=\Delta\boldsymbol{F}\,\boldsymbol{F}^e_n\left(\Delta\boldsymbol{F}^p(\Delta\gamma_s)\right)^{-1}$

LCGRLA computation of $\boldsymbol{F}^e_{n+1}=\Delta\boldsymbol{F}\,\boldsymbol{F}^e_n\left(\Delta\boldsymbol{F}^p(\Delta\gamma_s)\right)^{-1}$ $\boldsymbol{E}^e_{GL}=\frac{1}{2}(\boldsymbol{F}^{eT}\boldsymbol{F}^e-\boldsymbol{I}_d)$

$\boldsymbol{S}=\boldsymbol{\Lambda}:\boldsymbol{E}^e_{GL}$. and $\boldsymbol{R_1}(\boldsymbol{S}, \Delta\beta)=\boldsymbol{\Lambda}^{-1}.\boldsymbol{S}-\frac{1}{2}\left(\boldsymbol{F}^{eT}\boldsymbol{F}^e-\boldsymbol{I}_d\right)$

It is thus noted there that one uses still same the routines as for explicit integration: LCMMFI, LCMMFE, LCMMEC are a priori the only routines to be modified at the time of the addition of a behavior, that it is into small or large deformations, for the computation of the residue, during implicit integration.

3.3.3 Computation of the jacobian matrix

routine LCMMJA calculates the jacobian matrix. Its structure is the following one:

```
C IFA=1, NBFSYS
   C IS=1, NBSYS
      LCMMJB : computation of derived terms
         LCMMJ2 : computation of the derived terms for MONO_DD_KR
         LCMMJD : computation of the derived terms for MONO_DD_CFC, MONO_DD_CC
         LCMMJ1 : computation of the derived terms for MONO_VISC1, MONO_VISC2
   ENDDO
ENDDO
```

the derivatives of these equations for the computation of the jacobian matrix can be written in a general way:

HP		GDEF	
$J_{11}=\frac{\partial \boldsymbol{R_1}(\boldsymbol{\sigma},\Delta\beta)_i}{\partial \boldsymbol{\sigma_j}}$	$J_{12}=\frac{\partial \boldsymbol{R_1}(\sigma,\Delta\beta)_i}{\partial \beta_s}$	$J_{11}=\frac{\partial \boldsymbol{R_1}(\boldsymbol{S},\Delta\beta)_i}{\partial \boldsymbol{S_j}}$	$J_{12}=\frac{\partial \boldsymbol{R_1}(\boldsymbol{S},\Delta\beta)_i}{\partial \beta_s}$
$J_{21}=\frac{\partial \boldsymbol{R_2}(\boldsymbol{\sigma},\Delta\beta)_s}{\partial \boldsymbol{\sigma_i}}$	$J_{22}=\frac{\partial \boldsymbol{R_2}(\boldsymbol{\sigma},\Delta\beta)_s}{\partial \beta_r}$	$J_{21}=\frac{\partial \boldsymbol{R_2}(\boldsymbol{S},\Delta\beta)_s}{\partial \boldsymbol{S_i}}$	$J_{22}=\frac{\partial \boldsymbol{R_2}(\boldsymbol{S},\Delta\beta)_s}{\partial \beta_r}$

Table 3.3.3-1

terms intervening in each submatrix in common have narrower terms with each behavior, which are calculated in routines LCMMJ* [R5,03,11 appendix 5]:

- $\frac{\partial \Delta\gamma_s}{\partial \tau_s}=\frac{\partial \Delta p_s}{\partial \tau_s}\xi_s$
- $\frac{\partial \Delta\gamma_r}{\partial \Delta\beta_s}=\frac{\partial \Delta p_r}{\partial \Delta\beta_s}\xi_r$,
- $\frac{\partial k_s}{\partial \tau_s}$
- $\frac{\partial k_r}{\partial \Delta\beta_s}$

In the case of a new behavior, it is thus necessary either to add the computation of these terms derived in an existing routine LCMMJ*, or to add a news of it.

Note: for a first test, one can do without the computation of the jacobian matrix, by means of the automatic construction of the jacobian matrix (by disturbance, ALGO_INTE=' NEWTON_PERT'). So it is very fast to introduce a new crystalline behavior into environment PLASTI: it is enough to calculate the residue in routine LCMMRE, called by LCRESI. On the other hand, time computation will be optimized only with one programmed jacobian matrix.

3.3.4 Convergence criterion and postprocessings

the convergence criterion is generic a priori, and does not depend on the behavior, but can be possibly modified:

- LCCONV => LCMMCV convergence criterion

the last routine to be modified (possibly, if one wants to calculate and add to the local variables in output of the values useful for postprocessing) is:

- LCPLNF => LCDPEC which recomputes all the local variables from the solution of the reduced system. She once again calls on the routine of behavior LCMMLC.