
To introduce a new behavior

Summary:

How to introduce a new behavior?

One describes here the addition of a new behavior to solve a nonlinear problem posed on a structure, with `STAT_NON_LINE` or `DYNA_NON_LINE`, for all the elements 2D/3D (and multifibre hulls, pipes, beams,...)

Crucial steps :

- Writing of the reference material R (equations of the law of behavior)
- Modification of the catalogue of `DEFI_MATERIAU` (parameters material of the law of behavior)
- Addition of the catalogue python of the relation of behavior
- Choice of the method of integration among the following possibilities:
 - writing of an autonomous routine `lc0nn` integrating the behavior in a point of integration
 - integration clarifies (`ALGO_INTE=' RUNGE_KUTTA'`) and writing of the associated routines
 - implicit integration in the environment `PLASTI`, establishment “supplements” (`ALGO_INTE=' NEWTON'`)
 - implicit integration in the environment `PLASTI`, “easy” establishment (`ALGO_INTE=' NEWTON_PERT'`)
- To produce tests!

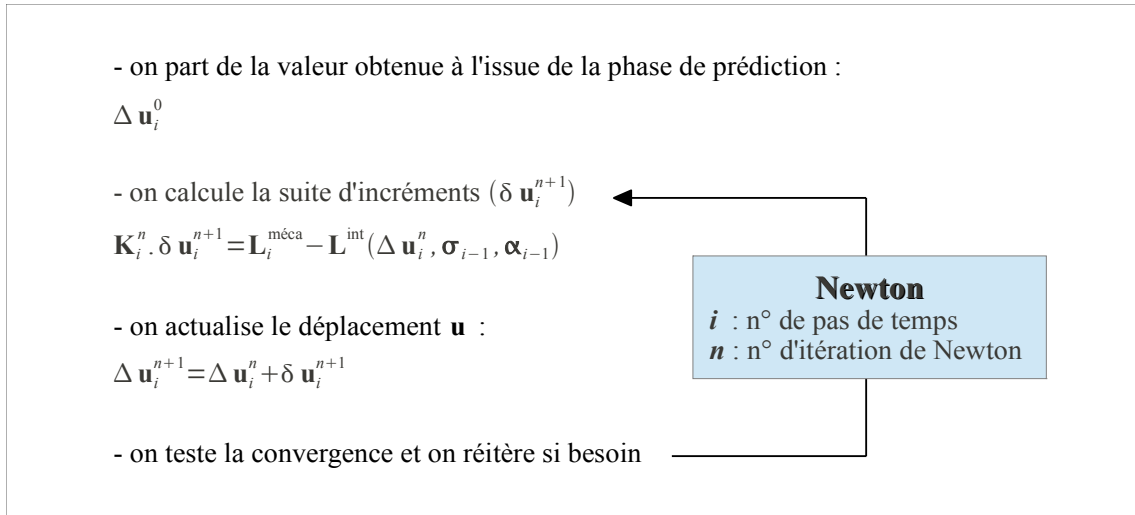
Notice : it is also possible to program laws of behavior in a routine of the type `Umat` (cf [U2.10.01]), that is to say using `MFront` (cf [U2.10.02] and the paragraph 5 this document).

Contents

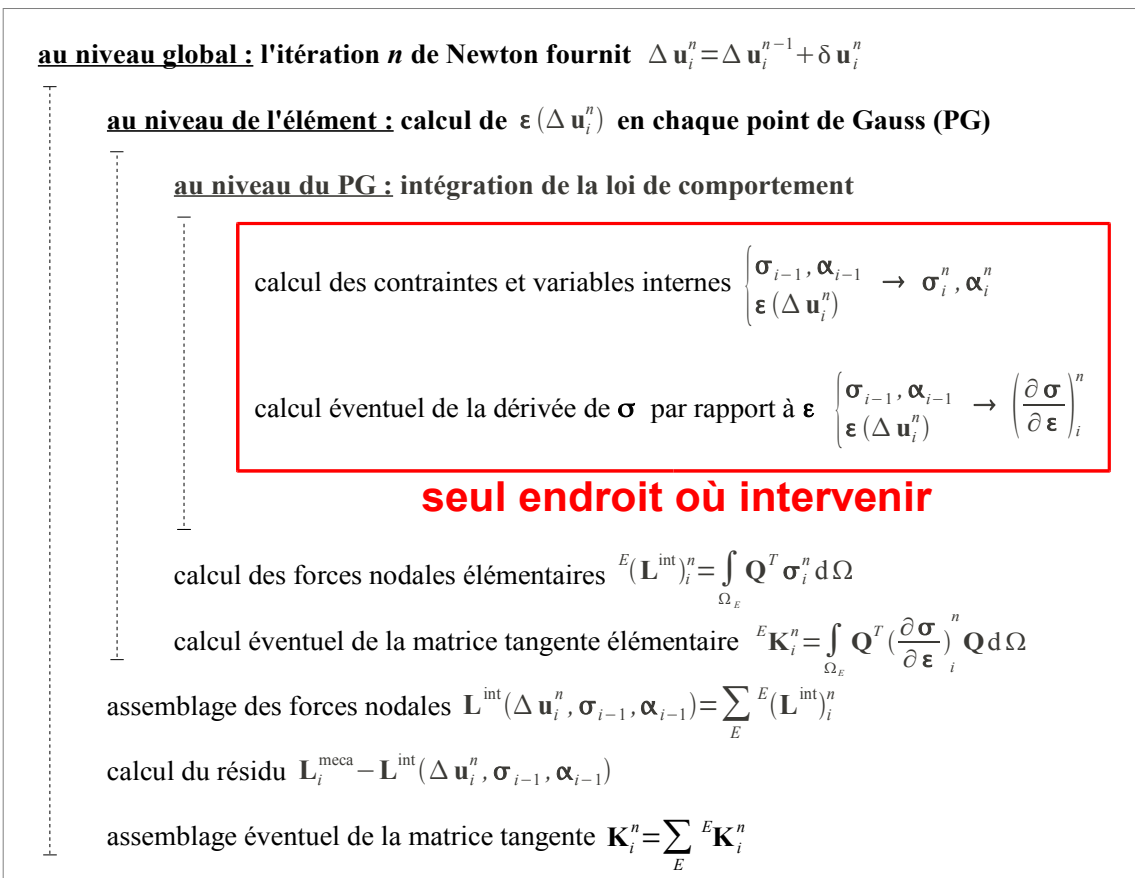
1 Outline general of resolution in STAT_NON_LINE.....	3
1.1 Options of calculation concerned.....	4
1.1.1 Option FULL_MECA (Full Newton).....	4
1.1.2 Option RAPH_MECA (Newton-Raphson).....	4
1.1.3 Option RIGI_MECA_TANG (calculation of the tangent matrix in prediction).....	4
1.2 Remarks concerning the calculation of the residue and the tangent matrix.....	5
1.2.1 Calculation of the residue.....	5
1.2.2 Calculation of the tangent matrix.....	5
2 And choice reference material of the method of integration.....	6
3 Procedure: catalogues.....	7
3.1 Modification of the catalogue of DEFI_MATERIAU.....	7
3.2 Modification of the catalogue C_RELATION.....	7
3.3 To add the catalogue of the law of behavior.....	7
4 Procedure: routines to be written.....	8
4.1 First possibility: to introduce a new explicit behavior – diagram of RUNGE - KUTTA.....	9
4.2 Second possibility: introduction “supplements” of a new behavior in PLASTI (implicit).....	10
4.3 Third possibility: use of the routines of integration clarifies in an implicit integration with PLASTI	13
4.3.1 Principle.....	13
4.3.2 Example.....	14
4.4 Fourth possibility: to write a routine lc00nn autonomous.....	15
4.4.1 lc00nn : routine relative to a point of integration of an element, specific to a law of behavior	15
4.4.2 Organization of the routine to be written.....	16
5 Typical case of the MFront laws.....	19
5.1 Catalogue for DEFI_MATERIAU and RELATION.....	19
5.2 Catalogue behavior.....	19
6 Validation and maintenance.....	19

1 Outline general of resolution in STAT_NON_LINE

Iteration of Newton on the complete system



Reference: Principle of resolution (algorithm of Newton) [R5.03.01] and modulates Nonlinear course Aster



1.1 Options of calculation concerned

1.1.1 Option FULL_MECA (Full Newton)

With the step of time i and with the iteration of Newton n , starting from the internal constraints and variables with preceding balance $(\sigma_{i-1}, \alpha_{i-1})$ and of the increment of deformation $\varepsilon(\Delta \mathbf{u}_i^n)$ (and possibly with the parameters: temperature, hydration,...), calculation in each point of Gauss of each finite element:

- internal constraints and variables (SIEF_ELGA, VARI_ELGA):

$$\begin{cases} \sigma_{i-1}, \alpha_{i-1} & \rightarrow & \sigma_i^n, \alpha_i^n \\ \varepsilon(\Delta \mathbf{u}_i^n) & & \end{cases}$$

- of the tangent operator:

$$\begin{cases} \sigma_{i-1}, \alpha_{i-1} & \rightarrow & \left(\frac{\partial \sigma}{\partial \varepsilon} \right)_i^n \\ \varepsilon(\Delta \mathbf{u}_i^n) & & \end{cases}$$

This option is calculated if REAC_ITER= m in the command file, and that the number of iteration n is multiple of m (reactualization of the coherent tangent matrix).

1.1.2 Option RAPH_MECA (Newton-Raphson)

With the step of time i and with the iteration of Newton n , starting from the internal constraints and variables with preceding balance $(\sigma_{i-1}, \alpha_{i-1})$ and of the increment of deformation $\varepsilon(\Delta \mathbf{u}_i^n)$ (and possibly with the parameters: temperature, hydration,...), calculation in each point of Gauss of each finite element:

- internal constraints and variables (SIEF_ELGA, VARI_ELGA):

$$\begin{cases} \sigma_{i-1}, \alpha_{i-1} & \rightarrow & \sigma_i^n, \alpha_i^n \\ \varepsilon(\Delta \mathbf{u}_i^n) & & \end{cases}$$

This option is calculated if REAC_ITER=0 or REAC_ITER= m in the command file, and that the number of iteration n is not multiple of m .

1.1.3 Option RIGI_MECA_TANG (calculation of the tangent matrix in prediction)

With the iteration 0 step of time i (initialization of the algorithm of Newton), one chooses like tangent matrix of prediction the tangent matrix to preceding balance ($i-1$), that is to say $\mathbf{K}_i^0 = \mathbf{K}_{i-1}$. With to leave the internal constraints and variables to preceding balance $(\sigma_{i-1}, \alpha_{i-1})$, calculation in each points of Gauss of each finite element:

- of the tangent operator in prediction:

$$\sigma_{i-1}, \alpha_{i-1} \rightarrow \left(\frac{\partial \sigma}{\partial \varepsilon} \right)_i^0$$

This option is calculated if `REAC_INCR= m` in the command file, and that the number of step of time i is multiple of m (reactualization of the tangent operator in prediction).

1.2 Remarks concerning the calculation of the residue and the tangent matrix

1.2.1 Calculation of the residue

The exact calculation of the residue $\mathbf{L}_i^{\text{meca}} - \mathbf{L}^{\text{int}}(\Delta \mathbf{u}_i^n, \boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1})$ (and thus of the constraints and the internal variables) is fundamental: it guarantees that one will converge towards the solution of the problem. A small error in the evaluation of the residue can have serious consequences .

1.2.2 Calculation of the tangent matrix

Matrix tangent known as coherent or consistent (Option `FULL_MECA`) :

Reactualized with each iteration, it ensures the best speed of convergence (quadratic) the algorithm of Newton (figure 1.2.2-1). Its calculation remains however expensive, and if a direct solver is used, it is necessary to add to the cost of each reactualization that of a factorization. Lastly, for great increments of loading, the coherent tangent matrix can lead to divergences of the algorithm.

Other “tangent” matrices:

One can make errors or approximations in the calculation of the “tangent” matrix: this resulted in degrading the speed of convergence compared to that which is obtained with the coherent tangent matrix reactualized with each iteration, but the solution obtained remains right as long as the residue is calculated in an exact way. There exist several alternatives (methods of quasi-Newton) possible authorized by `STAT_NON_LINE` (for more details to see [R5.03.01]):

- Elastic matrix (figure 1.2.2-2)
 - calculated only once (economic) starting from the parameters of elasticity
 - recommended in the event of discharge
 - slow but assured convergence
- Reactualized tangent matrix all them i_0 increments of load (figure 1.2.2-3) or all them n_0 iterations of Newton (figure 1.2.2-4)
 - less cost
 - direction less better evaluated
 - diverge sometimes in the zones from strong not linearity

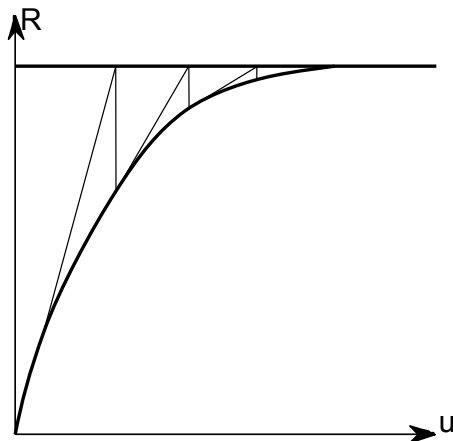


Figure 1.2.2-1: tangent matrix
reactualized with each iteration

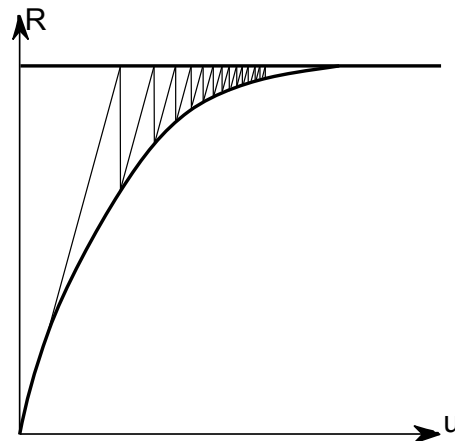


Figure 1.2.2-2: elastic matrix

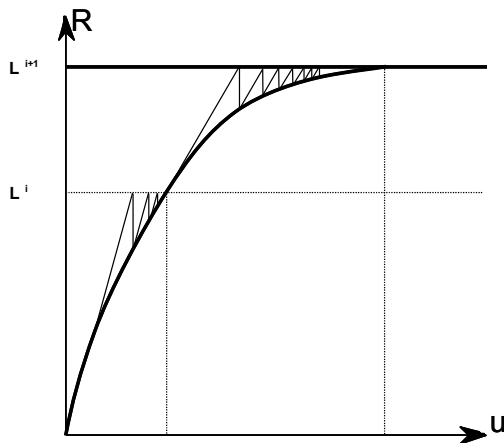


Figure 1.2.2-3: tangent matrix
reactualized with each increment of
loading

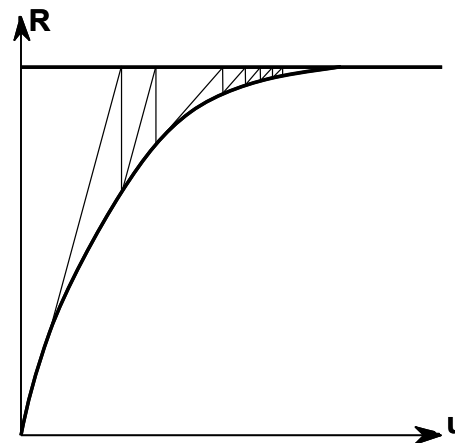


Figure 1.2.2-4: reactualized tangent
matrix all two iterations of Newton

2 And choice reference material of the method of integration

The writing of the Reference material is preliminary to the phase of development. The document (cf. for example [R5.03.02]) must specify according to the type of modeling concerned (continuous mediums 2D D_PLAN / AXIS and 3D , or models with local plasticity such as the hulls, the plates and the pipes and C_PLAN ...):

- the choice of the method of integration (see the various possibilities detailed with § 4);
- equations allowing to calculate the internal constraints and variables;
- equations allowing to calculate the two tangent matrices (options RIGI_MECA_TANG and FULL_MECA).

One can quote other examples of Reference materials:

- [R5.03.04]: *Relations of behavior élasto-visco-plastic of Chaboche.*
- [R5.03.16]: *Elastoplastic relation of behaviour to linear and isotropic work hardening kinematic nonlinear.*
- [R5.03.20]: *Relation of nonlinear elastic behavior in great displacements.*
- [R5.03.21]: *Elastoplastic modeling with isotropic work hardening in great deformations.*

3 Procedure: catalogues

3.1 Modification of the catalogue of `DEFI_MATERIAU`

The goal of the operator `DEFI_MATERIAU` [U4.43.01] is to introduce parameters of behavior. These parameters can be common to several relations of behavior (see the example below for the relations of behavior `VMIS_ISOT_LINE` and `VMIS_CINE_LINE`).

It is thus necessary hasjouter in the catalogue `defi_materiau.capy` (repertoire `catapy/order`) a keyword factor corresponding to the type of behavior which one wishes to introduce, and under this keyword factor, to add the simple keywords representing the parameters of this kind of behavior.

Example:

```
ECRO_LINE = FACT (statut=' f',  
                  D_SIGM_EPSI = SIMP (statut=' o', typ=' R',),  
                  SY           = SIMP (statut=' o', typ=' R',),),...
```

mean that the two keywords `SY` and `D_SIGM_EPSI` are obligatory for `ECRO_LINE` (for more precise details, to refer to [U1.03.01]: *Process control supervisor and language*)

3.2 Modification of the catalogue `C_RELATION`

It is necessary to add to the list returned by `C_RELATION` () the name chosen for the relation of behavior which one wishes to introduce (`'MA_RELATION'` in the example below). The catalogue to be modified is `c_relation.capy` (repertoire `catapy/common`).

Example:

```
def C_RELATION (): return (  
    "ELAS", #COMMUN#  
    ...  
    "LAIGLE",  
    "LEMAITRE",  
    "LEMAITRE_IRRA",  
    "LEMA_SEUIL",  
    "LETK",  
    "MA_RELATION",  
    "MAZARS",  
    "MAZARS_1D",  
    ...  
)
```

3.3 To add the catalogue of the law of behavior

This catalogue is to be added in the repertoire `bibpyt/Behavior`.

Example: `vmis_cine_line.py`

```
law = LoiComportement (  
    name           = 'VMIS_CINE_LINE',  
    Doc.           = """ Law of Von Mises... [R5.03.02] """,  
    num_lc         = 3,  
    nb_vari        = 7,  
    nom_vari       = ('XCINXX', 'XCINY', 'XCINZZ', 'XCINXY', 'XCINXZ',  
                     'XCINYZ', 'INDIPLAS',),  
    mc_mater       = ('ELAS', 'ECRO_LINE'),
```

```
modeling = ('3D', 'AXIS', 'D_PLAN', '1D'),  
deformation = ('SMALL', 'PETIT_REAC',  
               'GROT_GDEP', 'GDEF_LOG', 'GDEF_HYPO_ELAS'),  
nom_varc = ('TEMP',),  
algo_inte = ('ANALYTICAL',),  
type_matr_tang = ('DISTURBANCE', 'CHECKING'),  
properties = Nun,  
)
```

One thus provides in this catalogue most of the relative information to the behavior:

- name : name of the law, identical to that provided for `COMPR_INCR / RELATION`
- num_lc : number of routine `lc00nn`
- nb_vari/nom_vari : many internal variables, and their names (K8)
- mc_mater : keywords used in `DEFI_MATERIAU`
- modeling : possible types of modelings, for the behaviors of continuous mediums: `3D`, `D_PLAN`, `AXIS`, `C_PLAN`, `COMP1D`, `INCO`, `GRADEPSI`, `GRADVARI`,...
- deformation : type of possible deformations: `'SMALL'`, `'PETIT_REAC'`, `'GROT_GDEP'`, `'GDEF_LOG'`, `'GDEF_HYPO_ELAS'`.
- nom_varc : name of the variables of orders taken into account
- algo_inte : possible diagrams of integration: implicit (`'ANALYTICAL'`, `'NEWTON_PERT'`...), clarifies (`'RUNGE_KUTTA'`)
- type_matr_tang : types of tangent matrices available. In addition to the matrix by disturbance, one can also use the secant matrices, and the combination `TANGENTE_SECANTE`.

Notice :

The names of the internal variables of the whole of the behaviors are defined in the catalogue python `cata_vari.py`, in order to name in an identical way internal variables of the same significance. This catalogue is available in the repertoire `bibpyt/Behavior`. For a new behavior, it is desirable to re-use already existing names. If new names are added, an error occurs with the execution; it is thus necessary to modify `cata_vari.py`, by justifying its choice during the restitution.

Caution :

When that one adds a new catalogue, to check well the presence of the map of addition at the top of file. It specifies in which library python to place the file here (`Behavior`):

```
#@ ADDITION maloidecomportement Behavior
```

4 Procedure: routines to be written

It is on this level which must be made the choice of the type of integration. There exist four possibilities:

1. To use the architecture of the environment of integration clarifies by a diagram of Runge-Kutta of order 2 [R5.03.14] (`ALGO_INTE=' RUNGE_KUTA'`):
 - it is the simplest method. In addition to the recovery of the data materials, it is enough to write a routine calculating the derivative of the internal variables
 - the calculation of the tangent operator is not available under this environment, it is the elastic operator of rigidity which is used
2. Establishment "supplements" new behavior in the implicit environment of integration `PLASTI` [R5.03.14] (`ALGO_INTE=' NEWTON'`):
 - resolution of the local nonlinear system by the method of Newton. In addition to the recovery of the data materials, it is necessary to write several routines called in the

- algorithm of local Newton (evaluation of the threshold, calculation of the residue, analytical calculation of the matrix jacobienne...)
- The coherent operator is obtained directly starting from the jacobienne of the local system, and the tangent operator in prediction is by default the elastic operator of rigidity
 - `PLASTI` does not allow to obtain models optimized in time calculation
3. “Easy” establishment of the new behavior in the implicit environment of integration `PLASTI` with [R5.03.14] (`ALGO_INTE=' NEWTON_PERT'`):
- the local non-linear system can be rewritten so that the evaluation of the residue require on behalf of the developer to specify the expression of the derivative of the internal variables written within the framework of method 2 (explicit integration by RK2). One can thus also carry out an implicit integration in `PLASTI` with the two only routines necessary to explicit integration.
 - The jacobienne of the local system is calculated by disturbance, calculation is thus even more expensive than with method 2. Same manner, the coherent tangent operator is obtained directly starting from the jacobienne of the local system
4. To create an autonomous routine of complete integration of the behavior:
- often allows to obtain the most powerful models (for example, by reducing the system to be solved with only one scalar equation, nonlinear, to see for example [R5.03.04], [R5.03.16], [R5.03.21],...)
 - require more work “on paper” to optimize the equations

Caution :

In case 4, one will choose a number of routine `nn` and the routine will be written `lc00nn`. In the other cases one will choose like entrance point number 32: `LC0032` call `PLASTI` or `NMVPRK` (Runge-Kutta) according to the value of `ALGO_INTE` chosen by the user.

4.1 First possibility: to introduce a new explicit behavior – diagram of RUNGE - KUTTA

This kind of integration corresponds to `ALGO_INTE=' RUNGE_KUTTA'`, C'fastest D is the way' of introducing a new behavior.

A routine should be written initially `XXXMAT` called by the routine of shunting `LCMATE` in order to recover the parameters materials and the size of the non-linear differential connection to integrate.

```
SUBROUTINE XXXMAT (FAMI, KPG, KSP, MOD, IMAT, NMAT, MATERD, MATERF,  
                  MATCST, NDT, NDI, NR, NVI, VIND)
```

Arguments as starter:

```
FAMI, KPG, KSP: family and number of point of gauss/under-point  
IMAT          : address of material  
MOD           : type of modeling  
NMAT         : dimension of MATERD/MATERF
```

Arguments at exit:

```
MATERD       : coefficients material has T  
MATERF       : coefficients material has t+dt  
               MATERx (*, 1) = characteristic rubber bands  
               MATERx (*, 2) = characteristic plastics  
MATCST       : 'yes' if material has T = material has t+dt
```

```
        'not' if not
NDT      : total Nb of components of the tensors
NDI      : Nb of direct components of the tensors
NR       : Nb of components of the non-linear system
NVI      : Nb of internal variables
```

One gives below an example for each of the two principal functions which this routine must fill

- **ASSIGNMENT OF DIMENSIONS OF THE LOCAL PROBLEM (NDT, NDI, NR, NVI)**

```
NVI=7
IF (MOD .EQ. '3D') THEN
  NDT = 6
  NDI = 3
  NR  = NDT+2
ELSE IF (MOD .EQ. 'D_PLAN' .OR. MOD .EQ. 'AXIS') THEN
  NDT = 4
  NDI = 3
  NR  = NDT+2
ELSE
  CAL U2MESS ('F',...)
ENDIF
```

- **RECOVERY OF MATERIAL**

```
NOMC (1) = 'E'
NOMC (2) = 'NAKED'
NOMC (3) = 'ALPHA'
NOMC (4) = 'SY'
NOMC (5) = 'D_SIGM_EPSI'
CAL RCVALB (FAMI, KPG, KSP, '-', IMAT, '', 'ELAS', 0, '',
&          0.D0,3, NOMC (1), MATERD (1.1), ICODRE, 1)
CAL RCVALB (FAMI, KPG, KSP, '-', IMAT, '', 'ECRO_LINE', 0, '',
&          0.D0,2, NOMC (4), MATERD (1.2), ICODRE, 1)
```

A routine should then be written `RKDXXX` called by the routine of shunting `LCDVIN` and giving the derivative temporal of the internal variables.

Examples of routine `RKDXXX` : `RKDCHA`, `RKDVEC`, `RKDHAY`.

4.2 Second possibility: introduction “supplements” of a new behavior in `PLASTI` (implicit)

This type of integration corresponds to `ALGO_INTE='NEWTON'`. Environment `PLASTI` allows to integrate in a systematic way of the non-linear relations of behavior by a local method of Newton (on the level of the point of Gauss). Knowing the internal constraints and variables at the moment $i-1$ as well as the increment of total deflection $\Delta \epsilon_i^n$ given by the algorithm of total Newton, the local system of equations to be solved in purely implicit form is written in the following way:

$$R(\Delta \mathbf{y}) = \begin{pmatrix} g(\Delta \mathbf{y}) \\ l(\Delta \mathbf{y}) \\ f(\Delta \mathbf{y}) \end{pmatrix} = 0 \quad \text{with} \quad \Delta \mathbf{y} = \begin{pmatrix} \Delta \boldsymbol{\sigma} \\ \Delta \text{vari} \\ \Delta p \end{pmatrix}$$

The first equation represents for example the elastic relation stress-strain (6 equations with 6 unknown factors), with \mathbf{A} the operator of elasticity (possibly modified for the laws with damage), $\Delta \epsilon^p$ variation of plastic deformation and $\Delta \epsilon^th$ thermal variation of deformation:

$$g(\Delta \mathbf{y}) = \Delta \boldsymbol{\sigma} - \mathbf{A}(\Delta \boldsymbol{\varepsilon}_i^n - \Delta \boldsymbol{\varepsilon}^{th} - \Delta \boldsymbol{\varepsilon}^p) = 0$$

the second represents the whole of the laws of evolution of the various scalar and/or vectorial internal variables (n_v scalar equations with n_v unknown factors)

$$l(\Delta \mathbf{y}) = 0$$

the last represents the possible criterion of plasticity (1 equation)

$$f(\Delta \mathbf{y}) = 0$$

This system of $6 + n_v(+1)$ equations with $6 + n_v(+1)$ unknown factors is solved by a method of Newton:

$$\begin{cases} \frac{\partial R}{\partial \Delta \mathbf{y}}(\Delta \mathbf{y}_k) \cdot d[\Delta \mathbf{y}_k] = -R(\Delta \mathbf{y}_k) \\ \Delta \mathbf{y}_{k+1} = \Delta \mathbf{y}_k + d(\Delta \mathbf{y}_k) \end{cases}$$

With convergence, one thus obtains the increments of constraints and internal variables. The coherent tangent operator as for him is calculated in a systematic way starting from the jacobienne of the local system by the routine LCOPTG (see [R5.03.14] for the detail of the equations). It is thus necessary to program has minimum a routine defining the residue (formed by the equations above) as well as a routine building the matrix jacobienne. One describes briefly below the general architecture of PLASTI, by indicating the list of the routines progressively to be written.

General architecture of PLASTI :

```
CAL LCMATE (...)  
→ writing necessary of one specific routine XXXMAT of recovery of  
material identical to RUNGE_KUTTA  
  
IF (OPT .EQ. 'RAPH_MECA' .OR. OPT .EQ. 'FULL_MECA') THEN  
ELASTIC INTEGRATION ON DT  
CAL LCELAS (...)  
→ writing possible of one specific routine XXXELA (by default  
LCELIN: linear elasticity)  
  
ELASTIC PREDICTION STATE WITH T+DT  
CAL LCCNVX (... , THRESHOLD)  
→ writing necessary of one specific routine XXXCVX of evaluation of  
threshold  
  
IF (THRESHOLD .GE. 0.D0) THEN  
CAL LCPLAS (...)  
ENDIF  
ENDIF
```

The routine LCPLAS call LCPLNL, which realizes **the loop of Newton** the structure is the following one:

Notations:

YD= (SIGD, VIND) : vector of the unknown factors (of dimension $6 + n_v$) at the moment T
YF= (SIGF, VINF) : vector of the unknown factors at the moment T+DT
DY : increment of the vector of the unknown factors between the moments T and T+DT
DDY : increment of vector of the unknown factors between two successive iterations of Newton
R : residue

DRDY : jacobienne

One thus solves:

$$R(DY) = 0$$

By a method of Newton

$$DRDY(DYK) DDYK = -R(DYK)$$

$$DYK+1 = DYK + DDYK \quad (DY0 \text{ BEGINNING})$$

and one reactualizes

$$YF = YD + DY$$

CALCULATION OF THE INITIAL OUTPUT SOLUTION D TEST OF SYSTEM NL OUT OF DY
CAL LCINIT (... DY,...)

→ writing possible of one specific routine XXXINI (by default DY is initialized with 0)

ITERATIONS OF NEWTON

ITER = 0

1 CONTINUOUS

ITER = ITER + 1

INCREMENTING OF YF = YD + DY

CAL LCISOVN (NR, YD, DY, YF)

CALCULATION OF THE TERMS OF THE SYSTEM WITH T+DT = - R (DY)

CAL LCRESI (... , DY, R, IRET)

→ writing necessary of one specific routine XXXRES of calculation of residue

CALCULATION OF THE JACOBIEN OF THE SYSTEM WITH T+DT = DRDY (DY)

if ALGO_INTE=' NEWTON' exact calculation of the jacobienne

CAL LCJACB (... DY,... DRDY, IRET)

→ writing necessary of one specific routine XXXJAC

if not if ALGO_INTE=' NEWTON_PERT', calculation by disturbance (cf §

4.3)

CAL LCJACP (... DRDY,...)

RESOLUTION OF LINEAR SYSTEM DRDY (DY) .DDY = - R (DY)

CAL LCEQMN (NR, DRDY, DRDY1)

CAL LCEQVN (NR, R, DDY)

CAL MGAUSS ('NCWP', DRDY1, DDY, NR, NR, 1, RBID, IRET)

DY REACTUALIZATION = DY + DDY

CAL LCISOVN (NR, DDY, DY, DY)

LINEAR RESEARCH in case ALGO_INTE=' NEWTON_RELI '

CAL LCRELI (...)

ESTIMATE OF CONVERGENCE

CAL LCCONV (DY, DDY, NR, ITMAX, TOLER,..., R,..., IRTET)

→ writing possible of one specific routine XXXCVG of the criterion of convergence (relative criterion by default in LCCONG)

IF (IRTET.GT.0) GOTO 1

CONVERGENCE - > INCREMENTING OF YF = YD + DY

CAL LCISOVN (NDT+NVI, YD, DY, YF)

UPDATE OF SIGF, VINP

CAL LCEQVN (NDT, YF (1), SIGF)

CAL LCEQVN (NVI-1, YF (NDT+1), VINP)

In short, for an introduction "supplements" of a new behavior in PLASTI, necessarily should be written the following specific routines:

- XXXMAT called by LCMATE : recovery of material and the size of the local problem
- XXXCVX called by LCCNVX : evaluation of the threshold
- XXXRES called by LCRESI : calculation of the residue
- XXXJAC called by LCJACB : calculation of the jacobienne

It can also be useful, according to the need, to write the following specific routines:

- XXXELA called by LCELAS : elastic integration (if non-linear elasticity)
- XXXINI called by LCINIT : initialization (for an initialization other than DY0=0)
- XXXCVG called by LCCONV : to modify the convergence criteria

4.3 Third possibility: use of the routines of integration clarifies in an implicit integration with PLASTI

4.3.1 Principle

This last case corresponds to `ALGO_INTE_=' NEWTON_PERT'`, it is the method D'“easy” mplantation of the new behavior in the implicit environment of integration `PLASTI`. It is possible to use the two routines directly `XXXMAT` and `RKDXXX` (recovery of the data material, and derived from the internal variables) used with `ALGO_INTE_=' RUNGE_KUTTA'` to carry out an implicit integration. Indeed, the system of differential equations solved by `RUNGE_KUTTA` can be written:

$$\begin{cases} \Delta \sigma = A (\Delta \varepsilon_i^n - \Delta \varepsilon^{th} - \Delta \varepsilon^p(\mathbf{Y})) \\ \frac{d\mathbf{Y}}{dt} = F(\mathbf{Y}, t; \sigma) \end{cases}$$

where \mathbf{Y} represent the whole of the internal variables of the model. The relation between the tensor of the constraints and the elastic part of the tensor of the deformations is generally linear, but can be evaluated in a nonlinear way by a specific expression.

Once programmed the routine `RKDXXX` allowing to calculate $\frac{d\mathbf{Y}}{dt} = F(\mathbf{Y}, t; \sigma)$, it is possible to use it for an implicit integration, which consists in solving (cf. [R5.03.14]):

$$R(\Delta \mathbf{Z}) = 0 = \begin{bmatrix} R_1(\Delta \mathbf{Z}) \\ R_2(\Delta \mathbf{Z}) \end{bmatrix}, \text{ with } \Delta \mathbf{Z} = \begin{pmatrix} \Delta \sigma \\ \Delta \mathbf{Y} \end{pmatrix} = \mathbf{Z}(t + \Delta t) - \mathbf{Z}(t)$$

- The first system of equations represents the elastic relation stress-strain

$$R_1(\Delta \mathbf{Z}) = A^{-1} \sigma - (\Delta \varepsilon_i^n - \Delta \varepsilon^{th} - \Delta \varepsilon^p(\mathbf{Y})) = A^{-1} \sigma - G(\mathbf{Y}) = 0$$

By convention, first values of \mathbf{Y} represent the variation of plastic deformation, to facilitate the calculation of $G(\mathbf{Y})$ (see the routine `LCRESA` for more details)

- The second expresses the laws of evolution of the various internal variables, that is to say after temporal discretization by a diagram of implicit Euler:

$$R_2(\Delta \mathbf{Z}) = \Delta \mathbf{Y} - \Delta t \cdot F(\mathbf{Y}, \sigma) = 0$$

This system is solved by the method of Newton suggested in the environment `PLASTI` and described in the preceding paragraph:

$$\begin{cases} \frac{\partial R}{\partial \Delta Z} d(\Delta Z_k) = -R(\Delta Z_k) \\ \Delta Z_{k+1} = \Delta Z_k + d(\Delta Z_k) \end{cases}$$

Quantities G and F intervening in the residue are calculated by the routine "clarifies" `RKDXXX` to write, and the residue is built automatically by the routine `LCRESA`. The matrix jacobienne is calculated automatically by disturbance (routine `LCJACP`). The coherent tangent operator as for him is calculated in a systematic way starting from the jacobienne (routine `LCOPTG`, to see [R5.03.14] for the detail of the equations).

In short, this process allows, with the two only routines necessary to explicit integration, (coefficients material and calculation of the derivative of the internal variables) to use an implicit integration, and to profit from a tangent matrix. This process is economic in terms of time of development, but a *priori* less effective in time CPU than a matrix jacobienne programmed explicitly.

One details below the calculation of the tangent operator by disturbance, as well as the convergence criteria of the algorithm of local Newton.

Calculation by disturbance (`LCJACP`): finished differences in order 2

- Initialization of the disturbance: $\eta = 10^{-7} \|\Delta Z\|$
- buckle on the columns j matrix to be filled:
 - calculation of $R(\Delta Z + \eta I_j)$ with $I_j = [0 \ 0 \ \dots \ 1 \ \dots \ 0 \ 0]^T$ null vector except with the line j
 - calculation of $R(\Delta Z - \eta I_j)$
 - calculation of the column j : $\left[\frac{\partial R}{\partial \Delta Z} \right]_{\dots, j} \simeq \frac{R(\Delta Z + \eta I_j) - R(\Delta Z - \eta I_j)}{2\eta}$

Convergence criteria (`LCCONG`):

The two blocks are separated R_1 and R_2 residue to avoid the problems due to different orders of magnitude. With each iteration k algorithm of local Newton, one calculates:

$$err_1 = \frac{\|R_1(\Delta Z_k)\|_\infty}{\|R_1(\Delta Z_0)\|_\infty}, \text{ with } R_1(\Delta Z_0) = \Delta \varepsilon_i^n \text{ because one has } \Delta Z_0 = 0 \text{ with initialization. (cf §$$

4.2)

$$err_2 = \frac{\|R_2(\Delta Z_k)\|_\infty}{\|Y(t) + \Delta Y_k\|_\infty}$$

The criterion of stop is then the following:

$$\max(err_1, err_2) < \xi, \text{ where } \xi \text{ is given by RESI_INTE_REL.}$$

4.3.2 Example

An example: the law visco-élasto-plastic of Hayhurst.

Routine of reading of the coefficients material (called by the routine of shunting `LCMATT`):

- HAYMAT

Routine of calculation of the derivative of the internal variables (called by the routine of shunting `LCDVIN`):

- RKDHAY

The developments clarifies/implicit this relation of behavior are tested and compared in the CAS-test ssnv225 [V6.04.225]. It is about a test at the material point of creep in great deformations making it possible to validate the capacities of the model of HAYHURST to represent primary education, secondary and tertiary creep. Here characteristics of execution for this test in version 11.2:

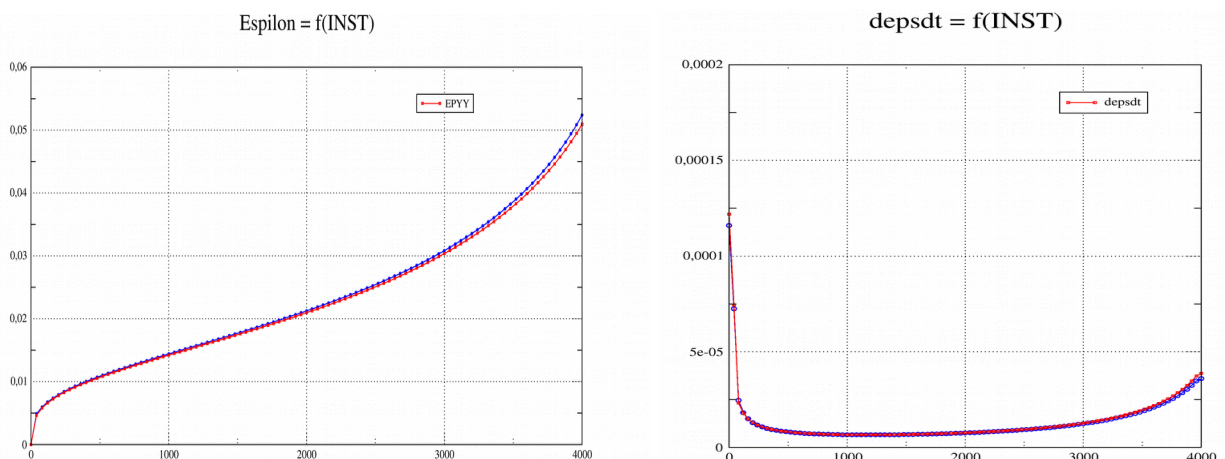
Modeling a: ALGO_INTE=' RUNGE_KUTTA '

- time CPU: 90.59 S
- Many steps of time: 1660
- Iteration count of Newton: 7615

Modeling b: ALGO_INTE=' NEWTON_PERT ' :

- time CPU: 27.38 S
- Many steps of time: 520
- Iteration count of Newton: 1404

The results are nearly identical:



4.4 Fourth possibility: to write a routine 1c00nn autonomous

4.4.1 1c00nn : routine relative to a point of integration of an element, specific to a law of behavior

To seek in the repertoire `bibfor/algorithm` a number of routine `1c00nn` not used ($50 < nn < 100$) and to start from this empty routine.

Notice :

The call to 1c00nn by 1c0000 (which is the routine calling all the routines of integration of the behavior available in Code_Aster) is already written. However, it should be made sure that the arguments (as well as the order of these arguments) selected with the declaration of 1c00nn by the developer are the same ones as with the call in 1c0000.

Arguments of entry of a routine `1c00nn` are *has minimum* :

FAMI family of points of gauss (RIGI, FARMHOUSE, ...)
KPG, KSP number of the point of gauss and the under-point

NDIM	dimension of L spaces (3d=3 , 2d=2 , 1d=1)
IMATE	address of material
COMPOR	information on the behavior compor (1) = relation of behavior (vmis_cine...) compor (2) = number internal variables compor (3) = type of deformation (small, green...)
CRIT	criteria buildings crit (1) = number D maximum iterations has convergence (ITER_INTE_MAXI) crit (3) = value of the tolerance of convergence (RESI_INTE_RELA)
INSTAM	moment T
INSTAP	moment T = T + dt
EPSM	total deflection has t- (or possibly the gradient of transformation according to the type of deformation: the arguments of the appealing routine, lc0000, contain the dimension of EPSM, and LIFO),
LIFO	increment of total deflection (even notices)
SIGM	constraint has T
VIM	internal variables has T
OPTION	option of calculation RIGI_MECA_TANG - > dsidep (T) FULL_MECA - > dsidep (t+dt), sig (t+dt) RAPH_MECA - > sig (t+dt)
ANGMAS	three angles (nautical or of Euler) of the massive mot_clef
TYPMOD	type of modeling: 3D, AXIS, D_PLAN, C_PLAN,...
ICOMP	meter for the local recutting of the step of time
NVI	many internal variables of the behavior

the arguments of exit are, according to the option of calculation :

VIP	internal variables has the current moment (options RAPH_MECA and FULL_MECA)
SIGP	constraints has the current moment (options RAPH_MECA and FULL_MECA)
DSIDEP	coherent tangent operator or of speed (option FULL_MECA or RIGI_MECA_TANG).
CODRET	code return allowing to indicate (if it is nonnull) a problem of integration local, therefore to carry out one recutting of the step of time

Remarks :

- if necessary, it is possible to also use a table of work as starter (*WKIN* in *LC0000*, of dimension *NWKIN*). This table contains additional arguments, for example a characteristic length in the case of the nonlocal models...
- In the same way, it is possible to transfer from the values at exit of the routine *lc00nn* (table *WKOUT*). But in this case it is necessary to define the dimension and the use of this table in the elementary routines calling the behavior – in plasticity 3D HP, for example, *TE0139/NMPL3D/NMCOMP*)
- Withtention, in the case *RIGI_MECA_TANG*, the tables relating to the constraints and internal variables at the end of the step of time are not allocated. They thus should not be used to calculate the tangent matrix of prediction.

4.4.2 Organization of the routine to be written

One takes as with § 3.3 the example of the law of Von Mises with linear kinematic work hardening *VMIS_CINE_LINE* (num_lc=3 in *vmis_cine_line.py*):

```
SUBROUTINE LC0003 (FAMI, KPG, KSP, NDIM, IMATE, COMPOR, CRIT, INSTAM,
```



```
&          INSTAP, EPSM, LIFO, SIGM, VIM, OPTION, ANGMAS, SIGP,  
VIP,  
&          PLUG, TYPMOD, ICOMP, NVI, DSIDE, CODRET)
```

This routine is in fact a routine of shunting for the behaviors `VMIS_CINE_LINE` and `VMIS_ECMI_*`. The integration of `VMIS_CINE_LINE` is realized in the routine `NMCINE`, called by `lc0003` and whose contents are described here briefly:

- **READING OF THE ELASTIC CHARACTERISTICS OF THE MATERIAL (TIME T = +)**

```
NOMRES (1) = ' E '  
NOMRES (2) = ' NU '  
CAL RCVALB (FAMI, KPG, KSP, '+', IMATE, '', 'ELAS', 0, '',  
&          0.D0, 2, NOMRES, VALRES, ICODRE, 2)  
E = VALRES (1)  
NAKED = VALRES (2)
```

Note:

RCVALB is a general routine allowing to interpolate the values of the coefficients materials compared to the variables of orders of which they depend (see the utilities).
RCVARC is a routine which makes it possible to recover the value of variables of orders (temperature, drying, irradiation,...) at the moment considered, and the point of Gauss considered (see the utilities). Example:

```
CAL RCVARC ('', 'TEMP', '-', FAMI, KPG, KSP, MT, IRET)  
CAL RCVARC ('', 'TEMP', '+', FAMI, KPG, KSP, TP, IRET)
```

- **READING OF THE CHARACTERISTICS OF WORK HARDENING**

```
NOMRES (1) = ' D_SIGM_EPSI '  
NOMRES (2) = ' SY '  
CAL RCVALB (FAMI, KPG, KSP, '+', IMATE, '', 'ECRO_LINE', 0, '',  
&          0.D0, 2, NOMRES, VALRES, ICODRE, 2)  
DSDE=VALRES (1)  
SIGY=VALRES (2)  
C = 2.D0/3.D0*DSDE/(1.D0-DSDE/E)
```

- **CALCULATION OF THE ELASTIC CONSTRAINTS AND THE CRITERION OF VON MISES**

```
C 110 K=1, 3  
  DEPSTH (K) = LIFO (K) - EPSTHE  
  DEPSTH (K+3) = LIFO (K+3)  
110 CONTINUOUS  
  EPSMO = (DEPSTH (1) +DEPSTH (2) +DEPSTH (3))/3.D0  
C 115 K=1, NDIMSI  
  DEPSDV (K) = DEPSTH (K) - EPSMO * KRON (K)  
115 CONTINUOUS  
  SIGMO = (SIGM (1) +SIGM (2) +SIGM (3))/3.D0  
  SIELEQ = 0.D0  
C 114 K=1, NDIMSI  
  SIGDV (K) = SIGM (K) - SIGMO*KRON (K)  
  SIGDV (K) = DEUXMU/DEUMUM*SIGDV (K)  
  SIGEL (K) = SIGDV (K) + DEUXMU * DEPSDV (K)  
  SIELEQ = SIELEQ + (SIGEL (K) - C/CM*VIM (K))** 2  
114 CONTINUOUS  
  SIGMO = TROISK/TROIKM * SIGMO  
  SIELEQ = SQRT (1.5D0*SIELEQ)  
  THRESHOLD = SIELEQ - SIGY  
  DP = 0.D0
```

```
PLASTI=VIM (7)
```

- **CALCULATION OF THE CONSTRAINTS AND THE INTERNAL VARIABLES**

Expressions of the constraints and the internal variables (RAPH_MECA and FULL_MECA) are given in [R5.03.02]

```
IF (OPTION (1:9) .EQ. 'RAPH_MECA' .OR.  
&    OPTION (1:9) .EQ. 'FULL_MECA') THEN  
  IF (SEUIL.LT.0.D0) THEN  
    VIP (7) = 0.D0  
    DP = 0.D0  
    SIELEQ = 1.D0  
    A1 = 0.D0  
    A2 = 0.D0  
  ELSE  
    VIP (7) = 1.D0  
    DP = THRESHOLD (1.5D0* (DEUXMU+C))  
    A1 = (DEUXMU/(DEUXMU+C)) * (SEUIL/SIELEQ)  
    A2 = (C / (DEUXMU+C)) * (SEUIL/SIELEQ)  
  ENDIF  
  PLASTI=VIP (7)  
  C 160 K = 1, NDIMSI  
    SIGDV (K) = SIGEL (K) - A1* (SIGEL (K) - VIM (K) *C/CM)  
    SIGP (K) = SIGDV (K) + (SIGMO + TROISK*EPSMO) *KRON (K)  
    VIP (K) = VIM (K) *C/CM + A2* (SIGEL (K) - VIM (K) *C/CM)  
160 CONTINUOUS  
ENDIF
```

- **CALCULATION OF TANGENT OPERATOR 'DSIPSEP': RIGI_MECA_TANG (of speed) or FULL_MECA (coherent)**

```
IF (OPTION (1:14) .EQ. 'RIGI_MECA_TANG' .OR.  
&    OPTION (1:9) .EQ. 'FULL_MECA') THEN  
  CAL MATINI (6,6,0.D0, DSIDEP)  
  C 120 K=1,6  
    DSIDEP (K, K) = DEUXMU  
120 CONTINUOUS  
  IF (OPTION (1:14) .EQ. 'RIGI_MECA_TANG') THEN  
    C 174 K = 1, NDIMSI  
      SIGDV (K) = SIGDV (K) - VIM (K) *C/CM  
174 CONTINUOUS  
  ELSE  
    C 175 K = 1, NDIMSI  
      SIGDV (K) = SIGDV (K) - VIP (K)  
175 CONTINUOUS  
  ENDIF  
  SIGEPS = 0.D0  
  C 170 K = 1, NDIMSI  
    SIGEPS = SIGEPS + SIGDV (K) *DEPSDV (K)  
170 CONTINUOUS  
  A1 = 1.D0/(1.D0+1.5D0* (DEUXMU+C)*DP/SIGY)  
  A2 = (1.D0+1.5D0*C*DP/SIGY) *A1  
  IF (PLASTI.GE.0.5D0.AND.SIGEPS.GE.0.D0) THEN  
    COEFF = -1.5D0* (DEUXMU/SIGY) ** 2/(DEUXMU+C) * A1  
    C 135 K=1, NDIMSI  
      C 135 L=1, NDIMSI  
        DSIDEP (K, L) = A2 * DSIDEP (K, L) + COEF*SIGDV (K) *SIGDV (L)  
135 CONTINUOUS
```

```
        LAMBDA = AVERAGE + DEUXMU ** 2*A1*DP/SIGY/2.D0
    ENDIF
    C 130 K=1,3
      C 131 L=1,3
        DSIDEF (K, L) = DSIDEF (K, L) + AVERAGE
131     CONTINUOUS
130     CONTINUOUS
    ENDIF
```

5 Typical case of the MFront laws

In complement of the document [u2.10.02] which describes how to use a law of MFront behavior in prototype mode, one specifies in this paragraph the characteristics of the laws of behavior MFront which are officially (under quality assurance) integrated into code_aster.

The name of the MFront file must be identical in the name of the behavior described in the file.

When the official laws of MFront behavior are compiled, the procedure of construction must know which files are produced by the conversion of the MFront file into C++. Usually, two files are produced: one of the name of the named behavior and one `aster` + name of the behavior.

If another file is produced, it should be indicated in the MFront file for example (in `PlasticityTH.mfront`):

```
//output Acier_ElasticYield-mfront
```

5.1 Catalogue for DEFI_MATERIAU and RELATION

The catalogue of order of the parameters materials is automatically integrated in order DEFI_MATERIAU by the procedure of description (at the time DE the stage `waf` installation):

- the keyword factor is the name of the behavior (example: `AnisoLemaitre`),
- the name of the parameters materials corresponds to `@MaterialProperty`.

If the parameters are tables (example: `@MaterialProperty real has [3]`), they will be named `a_0`, `a_1` and `a_2`.

A second keyword factor suffix by `_FO` (example: `AnisoLemaitre_FO`) is created in which all the parameters are functions.

The name of the behavior is automatically added for the keyword RELATION.

5.2 Catalogue behavior

The description of the behavior is slightly different. C is needed to create an object of the type `LoiComportementMFront` and to indicate the name of the produced symbol during compilation.

6 Validation and maintenance

The reflection on the tests of validation and identification can be started very early, before even the development itself. One can distinguish:

- command files (often simple, using for example `SIMU_POINT_MAT`) allowing to identify certain parameters (using `MACR_RECAL`).
- Tests to be restored: it must make it possible to validate the behavior in all its aspects, for a range of value of parameters covering the field of validity correctly. Certain tests are almost obligatory:
- tests of robustness and invariance compared to a rotation, a change of unit (`COMP001`, `COMP002`, `COMP003`)
- tests checking the good taking into account of the variables of orders (`COMP008`, `COMP010`)

Moreover the developer can have to correct or analyze its behavior following an anomaly met at the time of a study. In this case, besides the usual techniques of debugging, it is possible to use a functionality put in work in the event of failure of integration of the behavior: first points in failure produce a small command file making it possible to play again the scene with SIMU_POINT_MAT, for better analyzing the problem, testing other methods. It is enough to recopy the definition of material; all other data (forced, deformations, initial internal variables, and increment of deformation make it possible to simulate the behavior at the moment and the point where the failure occurred. For example for the test forma03c:

```
#-----
# test to analyze the failure of integration on the mesh <M83>, not <3>
#-----
BEGINNING ()

# to recopy DEFI_MATERIAU (...)
# DIAGRAM TRACTION
CTRAC = LIRE_FONCTION (UNITE=21, NOM_PARA=' EPSI', PROL_DROITE=' CONSTANT',)

MAT=DEFI_MATERIAU (ELAS=_F (E=200000., NU=0.3), TRACTION=_F (SIGM=CTRAC))

LISTE=DEFI_LISTE_REEL (DEBUT=1.9440000000000000E+02,
                      INTERVALLE=_F (JUSQU_A= 2.1870000000000000E+02, NOMBRE=1))

DEFLIST = DEFI_LIST_INST (  DEFI_LIST=_F (LIST_INST=LISTE,),
                          ECHEC=_F (SUBD_NIVEAU=10, SUBD_PAS=4),)

EXX=DEFI_FONCTION (NOM_PARA=' INST',
                  VALE= (1.9440000000000000E+02, -9.407813329102166E-04,
                        2.1870000000000000E+02, 7.373776084156800E+06))

EYY=DEFI_FONCTION (NOM_PARA=' INST',
                  VALE= (1.9440000000000000E+02, 1.718362911427018E-04,
                        2.1870000000000000E+02, -5.221483651275956E+06))

EZZ=DEFI_FONCTION (NOM_PARA=' INST',
                  VALE= (1.9440000000000000E+02, 0.0000000000000000E+00,
                        2.1870000000000000E+02, - 9.224110429927666E+05))

EXY=DEFI_FONCTION (NOM_PARA=' INST',
                  VALE= (1.9440000000000000E+02, 3.621278829822514E-04,
                        2.1870000000000000E+02, -4.710098874951571E+06))

RESU=SIMU_POINT_MAT (INFO=1, MATER=MAT, INCREMENT=_F (LIST_INST=DEFLIST),
                    EPSI_IMPOSE=_F (EPXX=EXX, EPYY=EYY, EPZZ=EZZ, EPXY=EXY),
                    SUPPORT=' ELEMENT', MODELISATION=' C_PLAN',
                    EPSI_INIT=_F (EPXX=-9.407813329102166E-04,
                                    EPYY=1.718362911427018E-04,
                                    EPZZ= 0.0000000000000000E+00,
                                    EPXY=3.621278829822514E-04,
                                    EPXZ=0.0000000000000000E+00,
                                    EPYZ= 0.0000000000000000E+00),
                    SIGM_INIT=_F (SIXX=-1.887253339740370E+02,
                                    SIYY=-2.497632316081525E+01,
                                    SIZZ= 0.0000000000000000E+00,
                                    SIXY=7.537194347798921E+01,
                                    SIXZ=0.0000000000000000E+00,
                                    SIYZ= 0.0000000000000000E+00),
                    VARI_INIT=_F (VALE= (3.931095545774485E-05,
                                            1.0000000000000000E+00,)),
                    COMPORTEMENT=_F (RELATION=' VMIS_ISOT_TRAC',
                                    ITER_INTE_MAXI=20,),
                    NEWTON=_F (REAC_ITER=1),)

END ()
```