To introduce a new Summarized

behavior:

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 shells, pipes, beams,…)

Crucial steps :

- Writing of documentation of reference R (equations of the constitutive law)
- Modification of the catalog of `DEFI_MATERIAU` (material parameters of the constitutive law)
- Addition of the catalog python of the behavior model
- Choice of the integration method among the following possibilities:
  - writing of an autonomous routine `lc0nn` integrating the behavior in a point of explicit
  - integration integration (`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!

Note: it is also possible to program constitutive laws apart from the environment Aster, either using the Zmat *modulus*, or in a routine of the type *Umat* (cf [U2.10.01]).

Contents

1 general Outline of resolution in STAT_NON_LINE

Iteration of Newton on the complete system

- on part de la valeur obtenue à l'issue de la phase de prédiction :
$\Delta \mathbf{u}_i^0$

- on calcule la suite d'incréments $(\delta \mathbf{u}_i^{n+1})$
$\mathbf{K}_i^n . \delta \mathbf{u}_i^{n+1} = \mathbf{L}_i^{\text{méca}} - \mathbf{L}^{\text{int}}(\Delta \mathbf{u}_i^n, \boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1})$

- on actualise le déplacement $\mathbf{u}$:
$\Delta \mathbf{u}_i^{n+1} = \Delta \mathbf{u}_i^n + \delta \mathbf{u}_i^{n+1}$

- on teste la convergence et on réitère si besoin

Newton
i : n° de pas de temps
n : n° d'itération de Newton

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

au niveau global : l'itération n de Newton fournit $\Delta \mathbf{u}_i^n = \Delta \mathbf{u}_i^{n-1} + \delta \mathbf{u}_i^n$

au niveau de l'élément : calcul de $\varepsilon(\Delta \mathbf{u}_i^n)$ **en chaque point de Gauss (PG)**

au niveau du PG : intégration de la loi de comportement

calcul des contraintes et variables internes $\left\{ \begin{array}{l} \boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1} \\ \boldsymbol{\varepsilon}(\Delta \mathbf{u}_i^n) \end{array} \right. \rightarrow \boldsymbol{\sigma}_i^n, \boldsymbol{\alpha}_i^n$

calcul éventuel de la dérivée de $\boldsymbol{\sigma}$ par rapport à $\boldsymbol{\varepsilon}$ $\left\{ \begin{array}{l} \boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1} \\ \boldsymbol{\varepsilon}(\Delta \mathbf{u}_i^n) \end{array} \right. \rightarrow \left(\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}} \right)_i^n$

seul endroit où intervenir

calcul des forces nodales élémentaires ${}^E(\mathbf{L}^{\text{int}})_i^n = \int_{\Omega_E} \mathbf{Q}^T \boldsymbol{\sigma}_i^n \, d\Omega$

calcul éventuel de la matrice tangente élémentaire ${}^E\mathbf{K}_i^n = \int_{\Omega_E} \mathbf{Q}^T \left(\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}} \right)_i^n \mathbf{Q} \, d\Omega$

assemblage des forces nodales $\mathbf{L}^{\text{int}}(\Delta \mathbf{u}_i^n, \boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1}) = \sum_E {}^E(\mathbf{L}^{\text{int}})_i^n$

calcul du résidu $\mathbf{L}_i^{\text{meca}} - \mathbf{L}^{\text{int}}(\Delta \mathbf{u}_i^n, \boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1})$

assemblage éventuel de la matrice tangente $\mathbf{K}_i^n = \sum_E {}^E\mathbf{K}_i^n$

1.1 time step Computation options

1.1.1 concerned Option FULL_MECA (Full Newton

) With i and with the iteration of Newton n , starting from the stresses and local variables with the preceding equilibrium $(\boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1})$ and of the increment of strain $\boldsymbol{\varepsilon}(\Delta \mathbf{u}_i^n)$ (and possibly with the parameters: temperature, hydration,…), computation in each Gauss point of each finite element:

- stresses and local variables (SIEF_ELGA, VARI_ELGA):

$$\begin{cases} \boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1} \\ \boldsymbol{\varepsilon}(\Delta \mathbf{u}_i^n) \end{cases} \rightarrow \boldsymbol{\sigma}_i^n, \boldsymbol{\alpha}_i^n$$

- of the tangent operator:

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

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 time step i and the iteration of Newton n , starting from the stresses and local variables with the preceding equilibrium $(\boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1})$ and of the increment of strain $\boldsymbol{\varepsilon}(\Delta \mathbf{u}_i^n)$ (and possibly with the parameters: temperature, hydration,…), computation in each Gauss point of each finite element:

- stresses and local variables (SIEF_ELGA, VARI_ELGA):

$$\begin{cases} \boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1} \\ \boldsymbol{\varepsilon}(\Delta \mathbf{u}_i^n) \end{cases} \rightarrow \boldsymbol{\sigma}_i^n, \boldsymbol{\alpha}_i^n$$

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 (computation of the tangent matrix in prediction)

With the iteration 0 time step i (initialization of the algorithm of Newton), one chooses like tangent matrix of prediction the tangent matrix to the preceding equilibrium ($i-1$), that is to say $\mathbf{K}_i^0 = \mathbf{K}_{i-1}$. From the stresses and local variables with the preceding equilibrium $(\boldsymbol{\sigma}_{i-1}, \boldsymbol{\alpha}_{i-1})$, computation in each Gauss points of each finite element:

- of the tangent operator in prediction:

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

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

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

1.2.1 Computation of the residue

The computation exact 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 stresses and the local variables) is fundamental: it guarantees that one will converge towards the solution of the problem. U small error in the evaluating of the residue can have serious consequences

1.2.2 Computation of the tangent matrix

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

Reactualized with each iteration, it ensures the best velocity of convergence (quadratic) the algorithm of Newton (figure 1.2.2-1). Its computation 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” matrixes:

One can make errors or approximations in the computation of the “tangent” matrix: this resulted in degrading the velocity 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]):

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

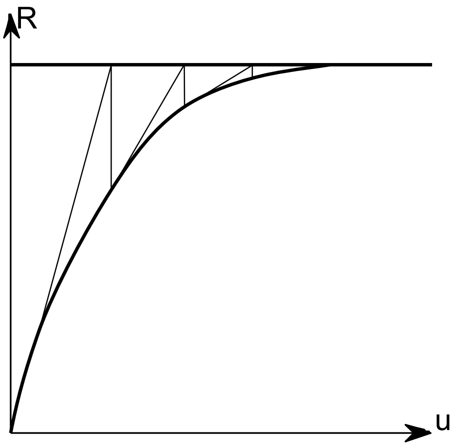


Figure 1.2.2-1: stamp tangent reactualized with each iteration

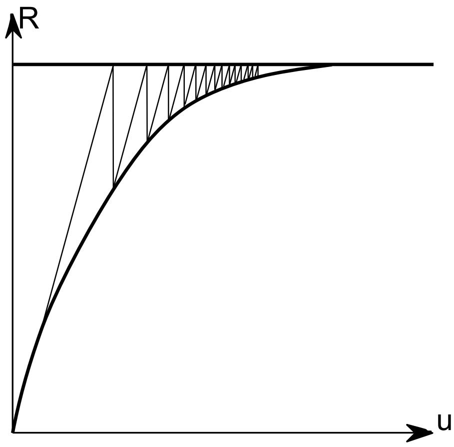


Figure 1.2.2-2: stamp elastic

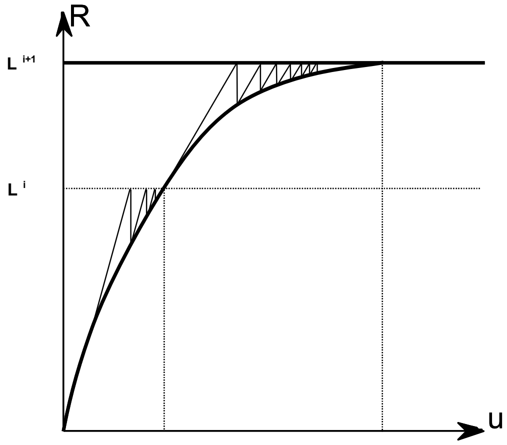


Figure 1.2.2-3: stamp tangent reactualized with each increment of loading

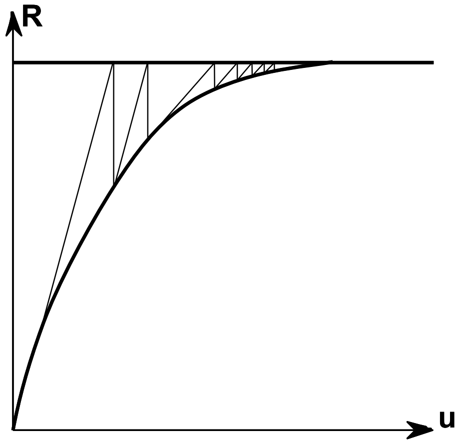


Figure 1.2.2-4: stamp tangent reactualized all the two iterations of Newton

2 Documentation of reference and choice of the integration method

the writing of Documentation of reference is preliminary to the phase of development. The document (*cf* for example [R5.03.02]) must specify according to the type of modelization concerned (continuums `2D` `D_PLAN` / `AXIS` and `3D` , or models with local plasticity such as the shells, plates and the pipes and `C_PLAN` …):

- the choice of the integration method (see the various possibilities detailed with the § 8);
- equations allowing to calculate the stresses and local variables;
- equations allowing to calculate the two tangent matrixes (options `RIGI_MECA_TANG` and `FULL_MECA`).

One can quote other examples of Documentations of reference:

- [R5.03.04]: *Behavior models élasto-visco-plastic of Chaboche.*
- [R5.03.16]: *Elastoplastic behavior model with linear and isotropic kinematic hardening nonlinear*.
- [R5.03.20]: *Nonlinear elastic behavior model in large displacements.*
- [R5.03.21]: *Elastoplastic modelization with isotropic hardening in large deformations.*

3 Procedure: the catalogs

3.1 Modification of the catalog of DEFI_MATERIAU

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

It is thus necessary to add in the catalog `defi_materiau.capy (directory` catapy `/commande)` a key word factor corresponding to the type of behavior which one wishes to introduce, and under this key word 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',),),… means
```

that two key words SY and D_SIGM_EPSI are compulsory for ECRO_LINE (for more precise details, to refer to [U1.03.01]: Process control *supervisor and language) Modification*

3.2 of the C_ catalog RELATION It

is necessary RELATION to add to the list returned by C_ () the name chosen for the behavior model which one wishes to introduce ("MA_RELATION" in the example below). The catalog to be modified is c_ relation.capy (directory catapy /commun). Example:

def C

```
_RELATION (): return ("ELAS                ", #COMMUN            #… "
                                          LAIGLE
                                          ", "LEMAITRE
                                          ", "LEMAITRE_IRRA
                                          ", "LEMA_SEUIL
                                          ", "LETK
                                          ", "LMARC_IRRA
                                          ", "MA_RELATION
                                          ", "MAZARS
                                          ", "MAZARS
                                          _1D",…)
                                          To add
```

3.3 the catalog of the constitutive law This catalog

is to be added in the directory bibpyt /Comportement. Example:

vmis_ cine_line.py model = LoiComportement

```
(name = "
   VMIS_CINE_LINE            ", Doc. = """
   Model               of Von Mises… [R5.03.02] """, num_lc
   = 3, nb_            vari
   = 7, name           _vari
   = ("XCINXX          ", "XCINYY", "XCINZZ", "XCINXY", "XCINXZ", "XCINYZ"
                        , "INDIPLAS",), mc_mater
   = ("ELAS         ", "ECRO_LINE"), modelization
   = ("3D",    "AXIS", "D_PLAN", "1D"), strain
   = ("PETIT     ", "PETIT_REAC", "GROT_GDEP
                       ", "GDEF_LOG", "GDEF_HYPO_ELAS"), nom_varc
   = ("TEMP         ",), algo_inte
   = ("ANALYTIQUE       ",), type_matr
   _tang = ("PERTURBATION", "VERIFICATION"), properties
   = Nun,      ) One thus
```

provides in this catalog most of information relating to the behavior: name: name

- of the model, identical to that provided for COMPR_INCR /RELATION num_lc:
- number of routine lc00nn Nb _vari
- /nom_vari: many local variables, and their names (K8) mc_mater
- : key words used in DEFI_MATERIAU modelization

•: possible types of modelizations, for the behaviors of continuums: 3D, D_PLAN , AXIS, C_PLAN, COMP1D, INCO, GRADEPSI, GRADVARI,… strain
•: type of possible strains: "PETIT", "PETIT_REAC", "GROT_GDEP", "GDEF_LOG", "GDEF_HYPO_ELAS". nom_ varc
•: name of the command variables taken into account algo_inte
•: diagrams of possible integration: implicit ("ANALYTIQUE", "NEWTON_PERT"…), clarifies ("RUNGE_KUTTA") type_matr
•_tang: types of tangent matrixes available. In addition to the matrix by disturbance, one can also use the secant matrixes, and combination TANGENTE_SECANTE . Note:

The names

> *of the local variables of all the behaviors are defined in the catalog python cata_vari.py , in order to name in an identical way of the same local variables meaning. This catalog is available in the directory 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; cata_vari.py thus should be modified, by justifying its choice during the restitution. Caution:*

When that *one*

> *adds a new catalog, to check well the presence of the card of addition at the top of file. It specifies in which library python to place the file (here Behavior) : #@ AJOUT maloidecomportement*
> *Behavior Procedure*

4 : the routines to write It is on this level

which must be made the choice of the type of integration. There exist four possibilities: To use the architecture

1. 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 derivatives of the local variables the computation of
- the tangent operator is not available under this environment, it is the elastic operator of stiffness which is used "complete

1. " Establishment of the new behavior in the implicit environment of integration PLASTI [R5.03.14] (ALGORITHME _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 (evaluating of the threshold, computation of the residue, analytical computation of the jacobian matrix…) The coherent operator
- is obtained directly from the jacobienne of the local system, and the tangent operator in prediction is by default the elastic operator of stiffness PLASTI does not allow
- to obtain models optimized in time "easy"

1. computation Establishment of the new behavior in the implicit environment of integration PLASTI with [R 5.03.14] (ALGO_INTE=' NEWTON _PERT'): the local nonlinear

- system can be rewritten so that the evaluating of the residue only requires on behalf of the developer to specify the form of derivatives of the local variables written in the frame 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, computation is thus even more expensive than with method 2. Same way, the coherent tangent operator is obtained directly from the jacobienne of the local system To create an autonomous

1.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 Attention:

In case *4,*

one will choose a number of routine nn and one will write the routine lc00nn. In `the other` *cases one will choose like entrance point number 32 : LC0032 calls* `PLASTI` *or NMVPRK* `(Runge - Kutta)` *according to the value of ALGO_INTE chosen* `by` *the user. First possibility*

4.1 : to introduce a new explicit behavior – diagram of RUNGE - KUTTA This kind of integration

corresponds to ALGO_INTE=' RUNGE `_KUTTA', it is the` fastest way to introduce a new behavior. It is necessary to initially write

a routine XXXMAT called `by` the routine of shunting LCMATE in order to `recover` materials parameters and cuts it nonlinear 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/subpoint IMAT: addresses
material          MOD: type of
modelization          NMAT: dimension
of         MATERD/MATERF Arguments in output
```

: MATERD: coefficients

```
       material  has T MATERF: coefficients
       material  has t+dt MATERx (*, 1) =
                characteristic MATERx elastics (*, 2) =
                characteristic plastics MATCST: "yes
" if material        has T  = material has t+dt "not" if not NDT
                : total Nb
of          components  of tensors NDI: Nb of
direct           components of the tensors NR: Nb of components
            of nonlinear system NVI: Nb of
local variables           One gives below
```

an example for each of the two principal functions which this routine ASSIGNMENT must fill OF

- **DIMENSIONS OF PROBLEM LOCAL (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
```

- **MATERIAU NOMC (1) = "E"**

```
 NOMC (2) = "NU"
 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) It is then necessary
```

to write a routine RKDXXX called by the routine of shunting LCDVIN and giving temporal derivatives of the local variables. Examples of routine

RKDXXX: RKDCHA , RKDVEC , RKDHAY . Second possibility

4.2 : "complete" introduction of a new behavior into PLASTI (implicit) This type of integration

corresponds to ALGO_INTE=' NEWTON ". Environment PLASTI makes it possible to integrate in a systematic way of the nonlinear behavior models by a local method of Newton (on the level of the Gauss point). Knowing the stresses and the local variables at time formulates as well as $i-1$ of total deflection formulates given $\Delta \boldsymbol{\varepsilon}_i^n$ of total Newton, the local system D" equations to be solved in purely implicit form is written in the following way: formulate with

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

represents for example the elastic relation stress-strain (6 equations with 6 unknowns), with formula the operator $\mathbf{A}$ elasticity (possibly modified for the models with damage), formulates the variation $\Delta \boldsymbol{\varepsilon}^p$ plastic strain and formula the variation $\Delta \boldsymbol{\varepsilon}^{th}$ thermal strain: formulate the second

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

all the models D" evolution of the various local variables scalar and/or vectorial (formula scalar n_v with formula unknowns n_v the last represents

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

the possible criterion of plasticity (1 equation) This system of

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

unknown equations $6+n_v(+1)$ to formula $6+n_v(+1)$ is solved by a method of Newton: formulate convergence A

$$\begin{cases}\frac{\partial R}{\partial \Delta y}(\Delta y_k).d[\Delta y_k]=-R(\Delta y_k)\\ \Delta y_{k+1}=\Delta y_k+d(\Delta y_k)\end{cases}$$

one thus obtains the increments of stresses and local variables. The coherent tangent operator as for him is calculated in a systematic way from the jacobienne of the local system by 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 jacobian matrix. 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 a specific routine XXXMAT of recovery of the material
identical
      to RUNGE_KUTTA IF (OPT .EQ.

    "RAPH_MECA" .OR. OPT .EQ. "FULL_MECA") THEN INTEGRATION ELASTIC
      ON DT CAL LCELAS (...)
      → possible
      writing of a specific routine XXXELA (by default LCELIN: linear
        elasticity) PREDICTION ELASTIC

      STATE A T+DT CAL LCCNVX (…
      , SEUIL) → writing
      necessary of a specific routine XXXCVX of evaluating of the threshold
IF (SEUIL
        .GE

      . 0.D0) THEN CAL LCPLAS (...)
        ENDIF ENDIF
      routine
    LCPLAS
```

calls LCPLNL , which carries out the loop of Newton **of which the structure** is the following one: Notations: YD

= (SIGD, VIND)

: vector of the unknowns (of dimension formulates) at $6+n_v$ T YF= (SIGF, VINF)
: vector of the unknowns at time T+DT DY: increment
of the vector of the unknowns between times T and T+DT DDY : increment
of vector of the unknowns between two successive iterations of Newton R: residue DRDY
: jacobienne
One thus solves

: R (DY) = 0 By a method
of NewtonDRDY (DYK) DDYK = - R (DYK) DYK+1 = DYK +
DDYK (DY0 debut) and one réactualiseYF
= YD + DY CALCUL OF the SOLUTION

```
D TEST INITIALE OF SYSTEME NL IN DY CAL LCINIT (…
    DY,…) → possible
    writing of a specific routine XXXINI (by default DY is initialized to
```

```
      0) ITERATIONS OF

NEWTON ITER = 0 1 CONTINUE
   ITER
   = ITER +
     1 INCREMENTING

OF YF = YD + DY CAL  LCSOVN (NR
     , YD, DY, YF) CALCUL OF the TERMS

OF the SYSTEME A T+DT = - R (DY) CAL LCRESI (…
     , DY, R, IRET) → writing
   necessary of a specific routine XXXRES of computation of the residue
CALCUL
     OF the JACOBIAN

OF the SYSTEME A T+DT = DRDY (DY) if ALGO_INTE='
     NEWTON' exact computation of the jacobienne CAL LCJACB (…
       DY,… DRDY, IRET) → writing
       necessary of a specific routine XXXJAC if not if ALGO_
     INTE=' NEWTON_PERT', computation by disturbance (cf § 4.3) CAL LCJACP
13
       DRDY,…) RESOLUTION OF

SYSTEME LINEAIRE 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) REACTUALIZATION

OF DY = DY + DDY CAL LCSOVN (
     NR, DDY, DY, DY) SEARCHES LINEAIRE

in case ALGO_INTE=' NEWTON_RELI' CAL LCRELI (
     …) ESTIMATE OF

the CONVERGENCE CAL LCCONV (DY
     , DDY, NR, ITMAX, TOLER,…, R,…, IRTET) → possible
     writing of a specific routine XXXCVG of the convergence criterion
(relative
       criterion by default in LCCONG) IF (IRTET.GT.0
     ) GOTO 1 CONVERGENCE - >

INCREMENTING OF YF = YD + DY CAL  LCSOVN (
   NDT+NVI, YD, DY, YF) UPDATE OF

SIGF, VINF CAL LCEQVN (
   NDT, YF (1), SIGF  ) CAL    LCEQVN (
   NVI-1, YF (NDT+1), VINF) In short, for
```

a "complete" introduction of a new behavior into PLASTI, it is necessarily necessary to write following specific routines: XXXMAT called

- by LCMATE: recovery of the material and the size of the local problem XXXCVX called
- by LCCNVX: evaluating of threshold XXXRES called
- by LCRESI: computation of residue XXXJAC called
- by LCJACB: computation 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 nonlinear elasticity) XXXINI called
- by LCINIT: initialization (for an initialization other than DY0=0) XXXCVG called
- by LCCONV: to modify the convergence criterion Third possibility

4.3 : use of the routines of explicit integration in an implicit integration with PLASTI Principle This last

4.3.1 case

corresponds to ALGO_INTE_=' NEWTON _PERT', it acts of the method of "easy" establishment of the new behavior in the environment of implicit integration PLASTI. It is possible to use directly two routines XXXMAT and RKDXXX (recovery of the material characteristics, and derived from the local variables) used with ALGO_INTE_=' RUNGE _KUTTA' to carry out an implicit integration. Indeed, the system of equations differentials solved by RUNGE_KUTTA can be written : formulate where

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

all $\boldsymbol{Y}$ the local variables of the model represent. The relation between the tensor of the stresses and the elastic part of the tensor of the strains is generally linear, but can be evaluated in a nonlinear way by a specific statement. Once programmed

routine RKDXXX making it possible to calculate, it is possible $\dfrac{d\boldsymbol{Y}}{dt} = F(\boldsymbol{Y}, t; \boldsymbol{\sigma})$ to use it for an implicit integration, which consists in solving (cf [R5.03.14]): , with the first

$$R(\Delta \boldsymbol{Z}) = 0 = \begin{bmatrix} R_1(\Delta \boldsymbol{Z}) \\ R_2(\Delta \boldsymbol{Z}) \end{bmatrix} \text{ system of equations } \Delta \boldsymbol{Z} = \begin{pmatrix} \Delta \boldsymbol{\sigma} \\ \Delta \boldsymbol{Y} \end{pmatrix} = \boldsymbol{Z}(t+\Delta t) - \boldsymbol{Z}(t)$$

- represents the elastic relation stress-strain By convention

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

  , the first values of represent $\boldsymbol{Y}$ the variation of plastic strain, to facilitate computation (see routine $G(\boldsymbol{Y})$ LCRESA for more details) the second expresses
- the laws of evolution of the various local variables, that is to say after temporal discretization by an implicit diagram of Eulerian: This system

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

is solved by the method of Newton suggested in environment PLASTI and described in the preceding paragraph: formulate the quantities

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

formulates and formulates G F the residue are calculated by routine "explicit" RKDXXX to write , and the residue is built automatically by routine LCRESA. The jacobian matrix is calculated automatically by disturbance (routine LCJACP). The coherent tangent operator as for him is calculated in a systematic way 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 computation of derivatives of the local 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 TEMPS CPU than an explicitly programmed jacobian matrix. One details below

the computation of the tangent operator by disturbance, as well as the convergence criterion of the algorithm of local Newton. Computation by disturbance

(LCJACP): finite differences of order 2 Initialization

- of the disturbance: buckle on $\eta=10^{-7}\|\Delta Z\|$
- the columns of the matrix j to fill: computation of with
  - null vector $R(\Delta Z+\eta I_j)$ except $I_j=[0\ \ 0\ \ ...\ \ 1\ \ ...\ \ 0\ \ 0]^T$ at line computation of computation j
  - of the column $R(\Delta Z-\eta I_j)$
  - formulates: Convergence criterion j $\left[\frac{\partial R}{\partial \Delta Z}\right]_{...,j} \simeq \frac{R(\Delta Z+\eta I_j)-R(\Delta Z-\eta I_j)}{2\eta}$

(LCCONG): One `separates`

the two blocks and from the residue R_1 R_2 to avoid the problems due to different orders of magnitude. A each iteration formulates algorithm k of local Newton, one calculates: formulate, with

$err_1=\frac{\|R_1(\Delta Z_k)\|_\infty}{\|R_1(\Delta Z_0)\|_\infty}$ because there $R_1(\Delta Z_0)=\Delta \boldsymbol{\varepsilon}_i^n$ with the initialization $\Delta Z_0=0$. (cf § 4.2)

the stopping criteria 10

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

are then the following: , where is given

$max(err_{1,} err_2)<\xi$ by ξ RESI_INTE_RELA . `Example an example`

4.3.2 :

the model visco-élasto-plastic of Hayhurst. Routine of reading

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

- `of computation`

of derivatives of the local variables (called by the routine of shunting LCDVIN): RKDHAY the developments

-

clarifies/implicit this behavior model is tested and compared in the ssnv225 benchmark [V6.04.225]. It is about a test at the material point of creep in large 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: Modelization

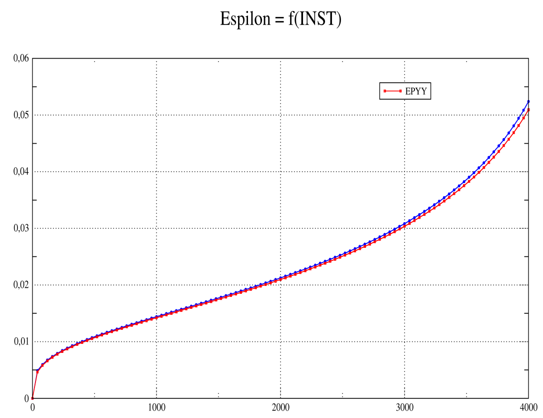
a: ALGO_INTE=' RUNGE `_KUTTA' TEMPS CPU:` 90.59

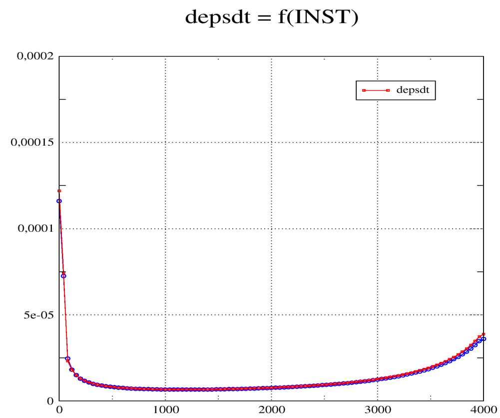
- S Number of time step
- : 1660 Nombre of iterations
- of Newton: 7615 Modelization

b: ALGO_INTE=' NEWTON _PERT': TEMPS CPU : 27.38

- S Number of time step
- : 520 Nombre of iterations
- of Newton: The 1404 results

are nearly identical: Fourth possibility





4.4 : to write an autonomous routine lc00nn `lc00nn` : routine

4.4.1 `relative` to a point of integration of an element, specific to a constitutive law To seek in

the directory bibfor/algorith `a number of` routine lc00nn not used formula) and to start $(50< nn< 100)$ this empty routine. Note:

The call to *lc*

> *00nn by lc `0000 (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 lc00nn by `the developer` are the same ones as with the call in lc0000. The arguments*

D" entered of a routine lc00nn are has `minimum` : FAMIfamille *of*

`points` of gauss (RIGI, FARMHOUSE,...) `KPG, KSP number`
`of the point` of gauss and subpoint NDIM dimension
`of L` spaces (formula, formula $3d=3$ $2d=2$ $1d=1$
`material` COMPOR information on
`the behavior` compor (1) = behavior model
(vmis_cine_...) `compor (2) =` many
`local variables` compor (3) = standard
`of strain` (small, green...) CRIT local
`criteria` crit (1) = number
`D maximum` iterations has convergence (ITER_INTE_MAXI) `crit (3) = value`
`of the tolerance` of convergence (RESI_INTE_RELA) `INSTAM urgent`
`urgent T` INSTAP
`T = T` - + dt EPSM `total`

deflection has T (or possibly the gradient of transformation according to the type of strain: the arguments of the appealing routine, lc0000, contain the dimension of EPSM, and LIFO), LIFO increment
of total deflection (even notices) SIGM forced
has T VIM local variables
has T OPTION computation option
RIGI_MECA_TANG
- > dsidep (T) FULL_MECA - >
dsidep (t+dt), sig (t+dt) RAPH_MECA - >
sig (t+dt) ANGMAS the three
angles (nautical or of Eulerian) of standard mot_clef massive TYPMOD of
modelization : 3D, AXIS, D_PLAN , C_PLAN ,… ICOMP meter
for the local recutting of time step the NVI many
local variables of the behavior the arguments

of output are, according to the computation option: VIP local variables

has current time (options RAPH_MECA and FULL_MECA) SIGP forced
has current time (options RAPH_MECA and FULL_MECA) DSIDEP coherent
tangent operator or of velocity (option FULL_MECA or RIGI_MECA_ TANG). CODRET return code
allowing to indicate (if it is non-zero) a problem of local integration, therefore
to carry out a recutting of the Note: time step if necessary

- *, it is possible to use also working table 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 in output 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/NMPL3 D/NMCOMP) Attention, in*
- *the case RIGI_MECA_TANG , the tables relating to the stresses and local variables at the end of time step are not allocated. They thus should not be used to compute: the tangent matrix of prediction. Organization of*

4.4.2 the routine to be written One takes as

with the § 3.3 the example 7 the model of Von Mises with linear kinematic 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, DSIDEP, CODRET) This routine is
```

in fact a routine of shunting for behaviors VMIS_CINE_LINE and VMIS_ECMI_*. The integration of VMIS_CINE_LINE is carried out in routine NMCINE, is called by lc0003 and whose contents are described here briefly: READING OF the ELASTIC

- **CHARACTERISTICS OF MATERIAU (TEMPS 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) NU
 =  VALRES (2) Note:
 : RCVALB
```

is a general

routine making it possible to interpolate the values of the coefficients materials compared to the command variables of which they depend (see the utilities). RCVARC is a routine which makes it possible to recover the value of command variables (temperature, drying, irradiation,…) at time considered, and the Gauss point 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 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) CALCUL OF THE ELASTIC
```

- **STRESSES AND THE CRITERE OF VON MISES C 110 K=1,3 DEPSTH**

```
   (K) = LIFO
     (K) - EPSTHE   DEPSTH (K+3) = LIFO
     (K+3) 110 CONTINUE EPSMO
=  (DEPSTH (1
   ) +DEPSTH (2) +DEPSTH (3))/3.D0 C 115 K=1, NDIMSI
   DEPSDV (K) = DEPSTH
     (K) - EPSMO   * KRON (K) 115 CONTINUE 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 CONTINUE SIGMO
=  TROISK/TROIKM
   * SIGMO SIELEQ = SQRT (1.5
   D0*SIELEQ) SEUIL = SIELEQ -
   SIGY  DP = 0.D0 PLASTI=
   VIM (7) CALCUL
   OF the STRESSES
```

- **AND the Local variables the statements of**

the stresses and local 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.D
        0 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 CONTINUE ENDIF
CALCUL    OF
     TANGENT
```

•OPERATOR "DSIPSEP": RIGI_MECA_TANG (of `velocity) 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 CONTINUE IF
(OPTION    (1:14)
        .EQ. "RIGI_MECA_TANG") THEN C 174 K = 1, NDIMSI
          SIGDV (K) = SIGDV (
            K) - VIM (K) *C/CM 174 CONTINUE ELSE
C       175 K = 1
        , NDIMSI
          SIGDV (K) = SIGDV (
            K) - VIP (K) 175 CONTINUE ENDIF
SIGEPS      = 0.
        D0 C
        170 K = 1, NDIMSI
        SIGEPS = SIGEPS +
          SIGDV (K) *DEPSDV (K) 170 CONTINUE A1
= 1.D    0 (1.D0+1.5
        D0* (DEUXMU+C)*DP/SIGY) A2 = (1.D0+1.5D0*
        C*DP/SIGY) *A1 IF (PLASTI.GE.0.5D
        0.AND.SIGEPS.GE.0.D0) THEN COEF = -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 CONTINUE LAMBDA = LAMBDA
+ DEUXMU      ** 2*A1
          *DP/SIGY/2.D0 ENDIF C 130 K=1,3 C 131 L=1,3
        DSIDEP
        (K, L) = DSIDEP
          (K,L) + LAMBDA
            131 CONTINUE 130 CONTINUE ENDIF
```