Note of use of the handling of fields and tables

Summary:

The objective of this document is to provide to the user some receipts related to the handling of the fields or the tables.

In this document the cases of handling of the following fields are described:

- To continue a calculation STAT_NON_LINE after having withdrawn or addhaving added elements in the model (ex: digging of a tunnel, installation of the layers of a stopping);
- To define a state of initial stresses (or internal variables) starting from “analytical formulas”;
- To apply a pressure to a part (not envisaged in the grid) of the edge of a structure 3D;
- To calculate the integral of the constraints or a quantity derived on the edge from a structure 3D;
- To read again a constant field by elements and to project it with the nodes of the grid.

And for the tables:

- To evaluate a damping on the calculated Eigen frequencies.

Contents

1 How to continue a calculation STAT_NON_LINE after having withdrawn or addhaving added elements in the model

1.1 Problems

The resumption of a calculation with a model made up of additional elements or contrary to a restricted model with certain elements requires some care. Indeed, calculation is not carried out any more on the same number of elements, which means that the fields defining the initial state of resumption of calculation (fields of displacement, constraints and internal variables) must undergo some adjustments to be able to be taken into account in the operator of resolution.

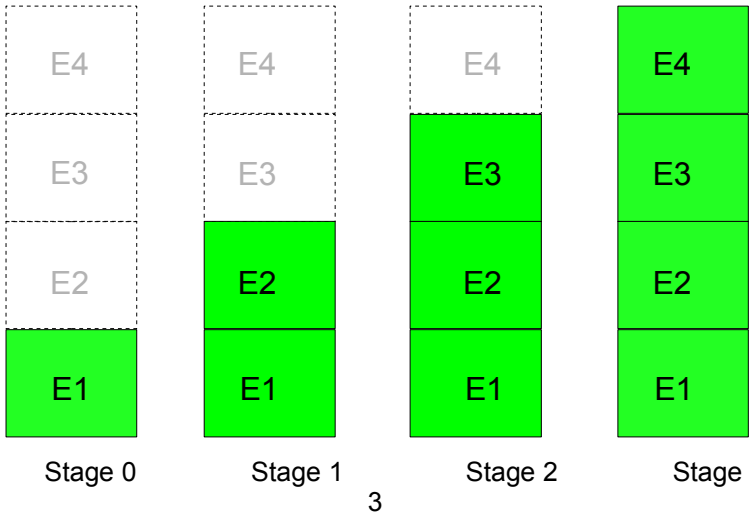
When one withdraws finite elements of the model, the problem is a priori rather simple: it is enough “to forget” information. On the other hand, when one wants to add elements, it is necessary of “to invent”.

Therefore in the continuation of this document, we will treat the case of the addition of elements.

1.2 Addition of elements in the model

1.2.1 Objective

We will present a methodology which makes it possible to carry out this kind of calculation starting from an example.



In this example one wishes to simulate the progressive addition of elements in the study such as for example simulating the construction of a stopping by adding layers of elements successively.

Four stages will be necessary to carry out these calculations: At the first stage, calculation will be carried out on a model made up of the group of meshs $E1$, at the second on a model made up of the groups of meshs $E1+E2$, and so on until stage 4. The user will have taken care during the construction of the grid to define the groups of meshs explicitly $E1$, $E2$, $E3$ and $E4$.

It will start by carrying out the stage 0, which consists in carrying out calculation on the grid restricted with the group $E1$. For the following stages, who consist in continuing calculation carried out previously, several “shapes” of implementation are possible:

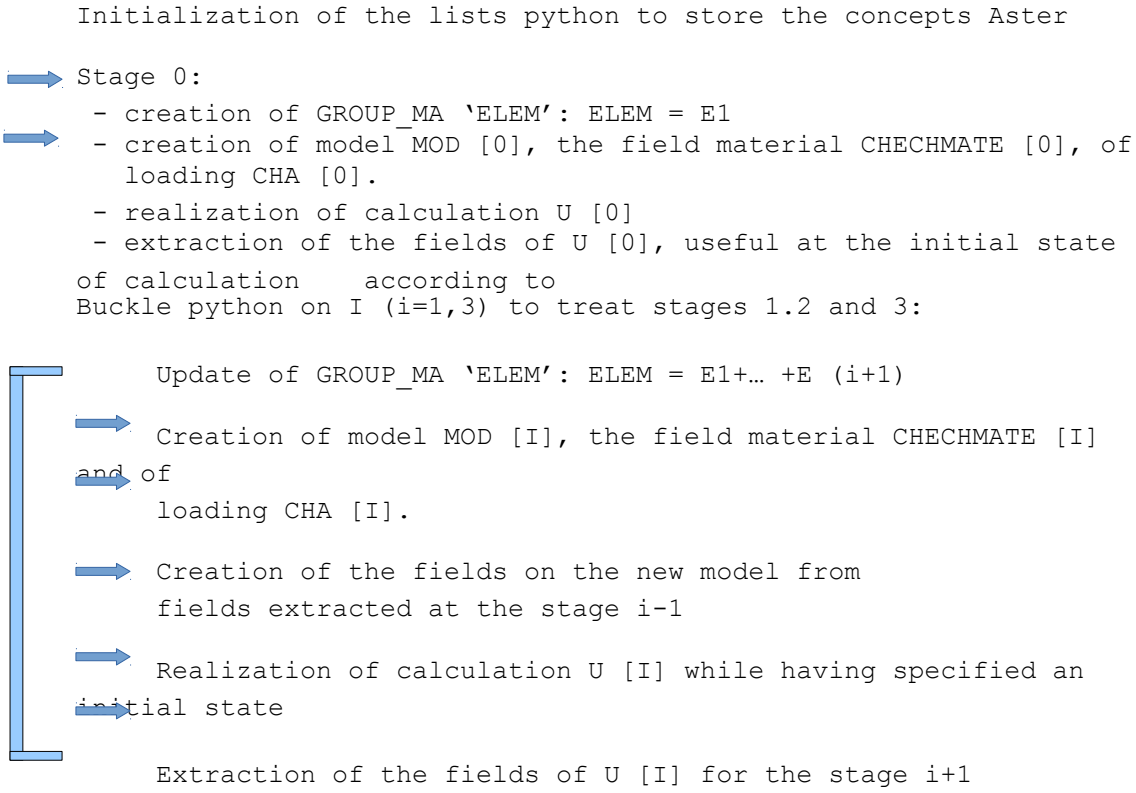
- either a continuation is carried out (new command file): this choice is not relevant if the number of layers of elements is consequent,

- either one enriches the command file: the size of the command file is function amongst continuations, and thus amongst layers; this choice is not inevitably judicious.
- either a loop python is used: we privilege this choice because it has the advantage of being easily evolutionary to take into account the addition of additional layers.

In the "content", the procedure is the same one: with an aim of providing an initial state to calculation, she calls upon the operator CREA_CHAMP :

- to extract fields of displacements, constraints, and internal variables,
- to extend these fields to the new model.

The following diagram illustrates the simplified structure of the command file:



In the table below we present the orders carried out as well as the produced concepts. It will be noticed how the concepts from one stage to another forward.

		Stage 0	Stage 1	Stage 2	Stage 3
Grid	LIRE_MAILLAGE DEFI_GROUP	E1	E1+E2	E1+E2+E3	E1+E2+E3+E4
Model	AFFE_MODELE	MOD_0	MOD_1	MOD_2	MOD_3
Material	AFFE_MATERIAU	MAT_0	MAT_1	MAT_2	MAT_3
Load	AFFE_CHAR_MECA	CHA_0	CHA_1	CHA_2	CHA_3
Initial state	CREA_CHAMP/ASSE		DEP_0* SIG_0* VARI_0*	DEP_1* SIG_1* VARI_1*	DEP_2* SIG_2* VARI_2*
Solution	STAT_NON_LINE	U_0	U_1	U_2	U_3
Extraction	CREA_CHAMP/EXTR	DEP_0 SIG_0 VARI_0	DEP_1 SIG_1 VARI_1	DEP_2 SIG_2 VARI_2	DEP_3 SIG_3 VARI_3

Table 1.1 : produced concepts

(*: computed field at the stage i, waited at the stage $i+1$)

One presents in the continuation the various stages and their implementations in the command file.

1.2.2 Implementation

1.2.2.1 Initialization

Each concept produced by these operators will be stored in a list python. It is thus advisable to allocate these lists. That is to say n the number of layers of elements: In this example, $n=4$.
One will write in the command file:

- MOD= [Nun] *n,
- MAT= [Nun] *n,
- CHA= [Nun] *n,
- U= [Nun] *n,
- DEP= [Nun] *n,
- SIG= [Nun] *n,
- VARI= [Nun] *n.

Note:
It is enough to modify the value of n to take into account new layers of elements.

1.2.2.2 Stage 0: sleep $E1$

- **the group is defined 'ELEM' who will evolve according to the various stages:**

```
DEFI_GROUP (MAILLAGE=MA, CREA_GROUP_MA=_F (NOM=' ELEM', GROUP_MA='
E1'),)
```

- **Creation of the new model, assignment of material and the loading:**

```
MOD [0] =AFFE_MODELE (MAILLAGE=MA, AFFE=_F (GROUP_MA=' ELEM',…)

CHECHMATE [0] =AFFE_MATERIAU (MODELE=MOD [0], AFFE=_F (TOUT=' OUI',
MATE=…),…)

CHA [0] =AFFE_CHAR_MECA (MODELE=MOD [0],…)
```

- **Calculation:**

```
U [0] =STAT_NON_LINE (MODELE=MOD [0], CHAM_MATER=MAT [0],
                      EXCIT=_F (CHARGE=CHA [0]),…)
```

- **Extraction of the fields of displacements, constraints, variables internal:**

```
EPD [0] =CRÉA_CHAMP (TYPE_CHAM=' NOEU_DEPL_R',
                  OPERATION=' EXTR',
                  RESULTAT=U [0],
                  NOM_CHAM=' DEPL',
                  …);

SIG [0] =CRÉA_CHAMP (TYPE_CHAM=' ELGA_SIEF_R',
                   OPERATION=' EXTR',
                   RESULTAT=U [0],
                   NOM_CHAM=' SIEF_ELGA',
                   …);

VAR [0] =CRÉA_CHAMP (TYPE_CHAM=' ELGA_VARI_R',
                   OPERATION=' EXTR',
                   RESULTAT=U [0],
                   NOM_CHAM=' VARI_ELGA',
                   …);
```

1.2.2.3 Stage i : layers $El+...+Ei$

At this stage, calculations were already carried out at the stages `0` with `i-1`. We are in the loop python "`for I in arranges (1, N):` " with the index `I` :

- **Actualization of the group of meshs `'ELEM'` grid `MY`**

```
DEFI_GROUP (MAILLAGE=MA,
            CREA_GROUP_MA=_F (NOM=' ELEM0', GROUP_MA=' ELEM'),…)

DEFI_GROUP (MAILLAGE=MA, DETR_GROUP_MA=_F (NOM=' ELEM'),…)

DEFI_GROUP (MAILLAGE=MA,
            CREA_GROUP_MA= (_F (NOM=' ELEM', OPTION=' UNION',
                               GROUP_MA= ('ELEM0', 'E'+str (i+1)),),…)
```

- **Creation of the new model, assignment of material and the loading**

```
MOD [I] =AFFE_MODELE (MAILLAGE=MA, AFFE=_F (GROUP_MA=' ELEM',…)

CHECHMATE [I] =AFFE_MATERIAU (MODELE=MOD [I], AFFE=_F (TOUT=' OUI',
MATE=…),…)

CHA [I] =AFFE_CHAR_MECA (MODELE=MOD [I],…)
```

- **Creation of the fields of displacements, constraints and variables internal for the initial state of the stage i**

```
DEPBID=CRÉA_CHAMP (TYPE_CHAM=' NOEU_DEPL_R',
                   OPERATION=' AFFE',
                   MODELE=MOD [I],
                   AFFE=_F (GROUP_MA=' ELEM',
                            NOM_CMP= ('DX', 'DY', 'DZ',…),
                            VALE= (1.0, 1.0, 1.0,…),),…)
```

```
INIDEP=CRÉA_CHAMP (TYPE_CHAM=' NOEU_DEPL_R',
                   OPERATION=' ASSE',
                   MODELE=MOD [I],
                   OPTION=' DEPL',
                   PROL_ZERO=' OUI',
                   ASSE= (_F (GROUP_MA=' ELEM',
                              CHAM_GD=DEPBID,
                              CUMUL=' OUI',
                              COEF_R=0.0),
                          _F (GROUP_MA=' ELEM0',
                              CHAM_GD=DEP [i-1],
                              CUMUL=' OUI',
                              COEF_R=1.0,),),);

 SIEBID=CRÉA_CHAMP (TYPE_CHAM=' ELGA_SIEF_R',
                    OPERATION=' AFFE',
                    MODELE=MOD [I],
                    AFFE=_F (GROUP_MA=' ELEM',
                             NOM_CMP= ('SIXX', 'SIYY', 'SIZZ',…),
                             VALE= (1.0, 1.0, 1.0,…),),…)

 INISIG=CRÉA_CHAMP (TYPE_CHAM=' ELGA_SIEF_R',
                    OPERATION=' ASSE',
                    MODELE=MOD [I],
                    PROL_ZERO=' OUI',
                    ASSE= (_F (GROUP_MA=ELEM,
                               CHAM_GD=SIEBID,
                               CUMUL=' OUI',
                               COEF_R=0.0),
                           _F (GROUP_MA=' ELEM0',
                               CHAM_GD=SIG [i-1],
                               CUMUL=' OUI',
                               COEF_R=1.0,),),);

CART = CREA_CHAMP (TYPE_CHAM=' ELGA_NEUT_R',
                   OPERATION=' AFFE',
                   MODELE=MOD [I],
                   PROL_ZERO=' OUI',
                   AFFE=_F (GROUP_MA=ELEM,
                            NOM_CMP = ('X1', 'X2', 'X3',…),
                            VALE    = (0.0, 0.0, 0.0,…),),)

INIVAR=CRÉA_CHAMP (TYPE_CHAM=' ELGA_VARI_R',
                   OPERATION=' ASSE',
                   MODELE=MOD [I],
                   PROL_ZERO=' OUI',
                   ASSE= (_F (GROUP_MA=ELEM,
                              CHAM_GD=CART,
                              NOM_CMP = ('X1', 'X2', 'X3',…),
                              NOM_CMP_RESU= ('V1', 'V2', 'V3',…),
                              CUMUL=' OUI', COEF_R=0.0),
                          _F (GROUP_MA=ELEM0,
                              CHAM_GD=VAR [i-1],
                              CUMUL=' OUI', COEF_R=1.0),))

U [I] =STAT_NON_LINE (MODELE=MOD [I], CHAM_MATER=MAT [I],
                      ETAT_INIT=_F (SIGM=INISIG, VARI=INIVAR, DEPL=INIDEP),
                      EXCIT=_F (CHARGE=CHA [I]),…)
```

```
EPD [I] =CRÉA_CHAMP (TYPE_CHAM=' NOEU_DEPL_R',
                     OPERATION=' EXTR',
                     RESULTAT=U [I],
                     NOM_CHAM=' DEPL',);

SIG [I] =CRÉA_CHAMP (TYPE_CHAM=' ELGA_SIEF_R',
                     OPERATION=' EXTR',
                     RESULTAT=U [I],
                     NOM_CHAM=' SIEF_ELGA',);

VAR [I] =CRÉA_CHAMP (TYPE_CHAM=' ELGA_VARI_R',
                     OPERATION=' EXTR',
                     RESULTAT=U [I],
                     NOM_CHAM=' VARI_ELGA',);
```

- **Suppression of the concepts INIVAR, INIDEP, INISIG, CART**

```
TO DESTROY (CONCEPT= (_F (NOM=INIDEP),
                     _F (NOM=INISIG),
                     _F (NOM=INIVAR),
                     _F (NOM=CART),),)
```

2 How to define a state of initial stresses (or internal variables) with analytical formulas

2.1 Objective

The objective is to be able to manufacture two of the fields constituting the initial state of a non-linear calculation: the stress field and the field of internal variables.
In both cases, the solution consists in connecting a certain number of orders CREA_CHAMP.

The reader is invited to consult the documentatione [U2.01.09] (“Danalytical éfinition of a stress field and a field of internal variables initial “) which answers these problems explicitly. It can also look at the command file of the CAS-test zzzz130a who illustrates this implementation.

We present below a “brief” receipt for each case.

2.2 Manufacturing of an analytical stress field

2.2.1 Receipt

- To define the formulas: starting from the analytical expression of each constraint, one builds the concept Aster corresponding starting from the operator FORMULA.
  Example :
  ```
  SIZZ = FORMULA (REEL= ''' (REAL: Z) = RHO*G*Z ''').
  ```
  (the formula thus defined depends on the variable Z (geometrical coordinate)).

- Creation of the field of formulas: one builds a neutral field (without particular size) to associate with each formula previously definite, a component of this field.
  Example :
  ```
  SIZF=CRÉA_CHAMP (OPERATION=' AFFE', TYPE_CHAM=' ELGA_NEUT_F', AFFE=_F
  (TOUT= (OUI', 'NOM_CMP'=' X1', VALE_F=SIZZ),…);
  ```

- Creation of the field of parameters:
  We must have a field whose components are parts of the variables of the formulas of the field of evaluation. In this example, we must have the geometrical field (component Z) with the points of Gauss. One proceeds in two stages:
  - extraction of the geometrical field to the nodes:
    ```
    CHXN = CREA_CHAMP (OPERATION=' EXTR',
                       TYPE_CHAM=' NOEU_GEOM_R', NOM_CHAM=' GEOMETRIE',…)
    ```
  - transformation of the field to the nodes into a geometrical field at the points of Gauss:
    ```
    CHXG = CREA_CHAMP (OPERATION=' DISC',
                       TYPE_CHAM=' ELGA_GEOM_R',…)
    ```

- Creation of the field by evaluation of the functions:
  Now, one can evaluate the field: the operation is used EVAL :
  ```
  SIGZ2=CRÉA_CHAMP (OPERATION=' EVAL', TYPE_CHAM=' ELGA_NEUT_R',
                    CHAM_F=SIZF, CHAM_PARA=CHXG,…)
  ```

- Creation of the stress field: to obtain our analytical stress field, the size should be allotted SIEF_R with the field of evaluation.
  ```
  SIGZ=CRÉA_CHAM (OPERATION=' ASSE', TYPE_CHAM=' ELGA_SIEF_R',
                  ASSE=_F (CHAM_GD=SIGZ2,
                           NOM_CMP=' X1', NOM_CMP_RESU=' SIZZ'),…).
  ```

2.2.2 Orders used

Orders	Functionality
FORMULA	Creation of the formulas
CREA_CHAMP/OPERATION=' AFFE'	Creation of the field of formulas
CREA_CHAMP/OPERATION=' EXTR'	Creation of the field of parameters
CREA_CHAMP/OPERATION=' EVAL'	Creation of the field of evaluation
CREA_CHAMP/OPERATION=' DISC'	Creation of the stress field wished after change of size

Table 2.1

2.3 Manufacturing of a field of internal variables not no one

Two methods are possible to create a field of internal variables not no one.

2.3.1 First method

2.3.1.1 Receipt

This method requires knowledge amongst internal variables of each law of behavior of your STAT_NON_LINE. Doc. U4.51.11 allows to obtain this information. This method rests on the assignment of all the internal variables of the laws provided to your STAT_NON_LINE. The operator of assignment is used AFFE of CREA_CHAMP for this effect.
This operation is carried out in two stages:

- one creates initially one MAP of NEUT_R
  Example:

```
VAIN = CREA_CHAMP (OPERATION=' AFFE', TYPE_CHAM=' CART_NEUT_R',
                   AFFE= (_F (GROUP_MA = ' GM1',
                              NOM_CMP= ('X1', 'X2'), VALE= (0, 2,)),
                          _F (GROUP_MA = ' GM2',
                              NOM_CMP= ('X1', 'X2', 'X3', X4', 'X5'),
                              VALE= (0,0,0,9,2,1)),…)
```

- one transforms the map of NEUT_R in field at the point of Gauss of VARI_R via the operation 'ADZE':

```
VARI=CRÉA_CHAMP (OPERATION=' ASSE', TYPE_CHAM=' ELGA_VARI_R',
                 ASSE= (_F (CHAM_GD=VAIN,
                            GROUP_MA=' GM1', NOM_CMP= ('X1', 'X2'),
                            NOM_CMP_RESU= ('V1', 'V2'))
                        _F (CHAM_GD=VAIN,
                            GROUP_MA=' GM2',
                            NOM_CMP= ('X1', 'X2', 'X3', 'X4', 'X5'),
                            NOM_CMP_RESU= ('V1', 'V2', 'V3', 'V4',
'V5'),),),
                 …)
```

The field of internal variables VARI contains 2 internal variables for the elements of the group $GM1$ and 5 for $GM2$. Thus the law of behavior to be applied to the group $GM1$ model in the operator of resolution will have to comprise 2 internal variables exactly.

2.3.1.2 Orders used

Orders	Functionality
CREA_CHAMP/AFFE	Creation of a map where all components are affected.
CREA_CHAMP/OPERATION=' ASSE'	Transformation of the map into field of internal variables.

2.3.2 Second method

2.3.2.1 Receipt

This method makes it possible to affect explicitly only the meshs which have nonworthless components. One proceeds in the following way:

- One carries out the first bogus calculation (to profit *a posteriori* of a field of internal variables "models"):

```
UBID=STAT_NON_LINE (COMPORTEMENT= (_F (GROUP_MA=' GM1', RELATION=…),
                                   _F (GROUP_MA=' GM2', RELATION=…),),
…)
```

- One extracts the field from internal variables VBID result UBID :

```
VBID=CRÉA_CHAMP (OPERATION=' EXTR', RESULTAT=UBID,
                 TYPE_CHAM=' ELGA_VARI_R', NOM_CHAM=' VARI_ELGA',…)
```

- One affects this time the not-worthless values:

```
VAIN1 = CREA_CHAMP (OPERATION=' AFFE', TYPE_CHAM=' CART_NEUT_R',
                    AFFE= (_F (GROUP_MA=' GM1',
                               NOM_CMP= ('X2'), VALE= (2,)),
                           _F (GROUP_MA = ' GM2',
                               NOM_CMP= ('X3', X4', 'X5'), VALE=
(9,2,1)),),
                    …)
```

- One puts at zero the field of internal variables "models" and one overloads it nonworthless values:

```
VAIN2 = CREA_CHAMP (OPERATION=' ASSE', TYPE_CHAM=' ELGA_VARI_R'
                    ASSE= (_F (TOUT=' OUI', CHAM_GD=VBID,
                               CUMUL=' OUI', COEF_R=0.),
                           _F (GROUP_MA=' GM1', CHAM_GD=VAIN1,
                               CUMUL=' OUI', COEF_R=1.,
                               NOM_CMP= ('X2'), NOM_CMP_RESU= ('V2'),),
                           _F (GROUP_MA=' GM2', CHAM_GD=VAIN1,
                               CUMUL=' OUI', COEF_R=1.,
                               NOM_CMP= ('X3', 'X4', 'X5'),
                               NOM_CMP_RESU= ('V3', 'V4', 'V5'),),),…)
```

2.3.2.2 Orders used

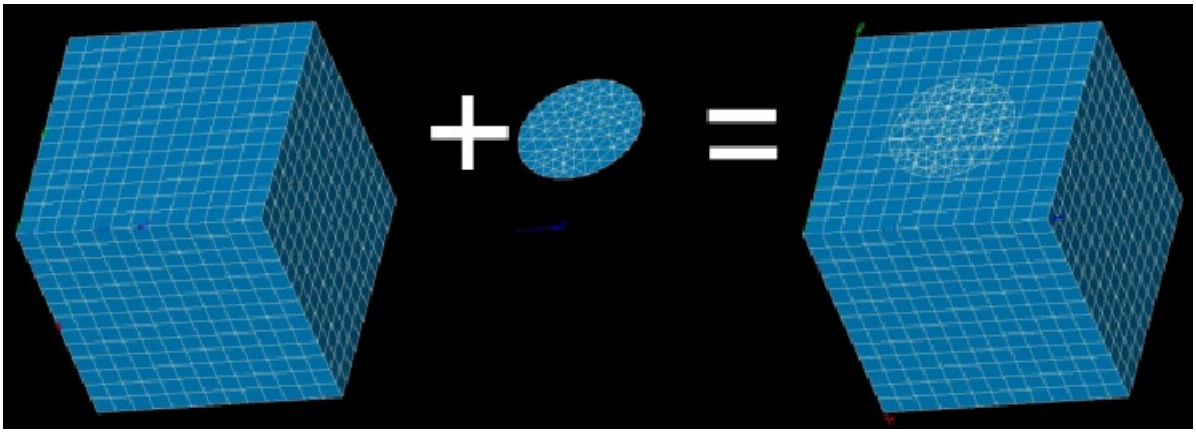
Orders	Functionality
STAT_NON_LINE	Creation of a bogus result
CREA_CHAMP/EXTR	Extraction of the field of internal variables.
CREA_CHAMP/AFFE	Creation of a map where only components not-worthless are affected.
CREA_CHAMP/ASSE	Transformation of the map into field of variables

	interns in order to assemble it with the field extracted the result (given beforehand to zero).

3 How to apply a pressure to a restricted part (nonwith a grid exactly) of the edge of a structure 3D

It is possible with *Code_Aster* to affect a field of pressure on a geometrical zone of the part to be calculated not necessarily with a grid. Indeed, it is enough to create the grid of skin support of the zone of pressure, to create there a field of pressure which one projects on the grid of the part. The setting in data is then similar to recovery/projection of a field of pressure calculated by a code of CFD.

In the example below, one wants to affect a pressure on a circular zone centered on a face of a cube, uniformly with a grid in HEXA8. The grid of the disc is carried out except for.



Note:

One checks the value of the total resultant of effort by a calculation of the integral of pressure on the affected zones. Indeed, the operation of projection induces a loss (proportional to the coarseness of the grid) which it is appropriate to measure.

We will be interested in creation of the load CHA_PROJ**.** The various stages are the following ones:

- reading of the grid (disc) carrying the effort of pressure;
- creation of the field of pressure on the disc;
- projection of the field of pressure since the disc towards the calculated structure;
- creation of the load.

We will detail each one of these stages.

3.1 Stage 1: creation of the field of pressure on the disc

- **Creation of a constant field of pressure by element**

```
VALLEY =CRÉA_CHAMP (   TYPE_CHAM =   'ELEM_PRES_R',
                 OPERATION =   'AFFE',
                 MODEL     =   modele2,
                 PROL_ZERO =   'YES',
                 AFFE   =_F (  GROUP_MA = 'disc',
                               NOM_CMP  = 'NEAR',
                               VALE     =  100000000.   ),
                 …)
```

- **Creation of a result of the type EVOL_CHAR starting from this field**

```
RES_PRES=CRÉA_RESU (OPERATION = 'AFFE',
                    TYPE_RESU = 'EVOL_CHAR',
                    NOM_CHAM  = 'NEAR',
                    AFFE      = _F (CHAM_GD   = VALLEY,
                                    MODEL    = modele2,
                                    INST      = 0.     ),)
```

3.2 Stage 2: projection of the field of pressure since the disc towards the calculated structure

```
RES_PROJ=PROJ_CHAMP (METHODE='COLLOCATION',
                     RESULTAT=RES_PRES,
                     MODELE_1=modele2,
                     MODELE_2=MODE,
                     DISTANCE_MAX=1.,
                     CAS_FIGURE=' 2.5D',
                     PROL_ZERO=' OUI',
                     VIS_A_VIS=_F (GROUP_MA_1=' disque',
                                   GROUP_MA_2=' Group_1',),
                     TOUT_ORDRE=' OUI',);
```

3.3 Stage 3: creation of the load

For the creation of the loading, one uses the evolutionary loadings in the time of the type evol_char products by LIRE_RESU [U7.02.01] (or CREA_RESU), and containing fields of pressure, densities of voluminal force in 2D or 3D and densities of surface force in 2D or 3D.

```
CHA_PROJ=AFFE_CHAR_MECA (MODELE=MODE,
                         EVOL_CHAR=RES_PROJ,)
```

3.4 Command file

```
BEGINNING ()

MAIL=LIRE_MAILLAGE (FORMAT=' MED',);

MAIL=MODI_MAILLAGE (reuse =MAIL,
                    MAILLAGE=MAIL,
                    ORIE_PEAU_3D=_F (GROUP_MA=' Group_1',),)

MAIL=DEFI_GROUP (reuse =MAIL,
                 MAILLAGE=MAIL,
                 CREA_GROUP_NO=_F (NOM=' Group_3',
                                   GROUP_MA=' Group_3'))

MA=DEFI_MATERIAU (ELAS=_F (  E = 2.E11,
                             NAKED = 0.3,
                             ALPHA = 1.E-5,
                             RHO=1.  ),
                  ECRO_LINE=_F (  D_SIGM_EPSI = 2.E9,
                                SY = 2.E8)  )

MODE=AFFE_MODELE (MAILLAGE=MAIL,
                  AFFE=_F (TOUT=' OUI',
                           PHENOMENE=' MECANIQUE',
                           MODELISATION=' 3D',),);

MATE=AFFE_MATERIAU (MAILLAGE=MAIL,
                    AFFE=_F (TOUT=' OUI',
                             MATER=MA,),);

########################################
##### creation of load CHA_PROJ #####
########################################

# reading of the grid (disc) carrying L effort of pressure
mail2=LIRE_MAILLAGE (UNITE=21,
                     FORMAT=' MED',);

mail2=MODI_MAILLAGE (reuse =mail2,
                     MAILLAGE=mail2,
                     ORIE_PEAU_2D=_F (GROUP_MA=' disque',),);

modele2=AFFE_MODELE (MAILLAGE=mail2,
                     AFFE=_F (TOUT=' OUI',
                              PHENOMENE=' MECANIQUE',
                              MODELISATION=' 3D',),)

# creation of the field of pressure on the disc
VALLEY =CRÉA_CHAMP (TYPE_CHAM=' ELEM_PRES_R',
          OPERATION=' AFFE',
          MODELE=modele2,
          PROL_ZERO=' OUI',
          AFFE=_F (GROUP_MA = 'disc',
             NOM_CMP  = 'NEAR',
             VALE     =  100000000.   ),
          INFO=1,)
```

```
RES_PRES=CRÉA_RESU (OPERATION = 'AFFE',
                    TYPE_RESU = 'EVOL_CHAR',
                    NOM_CHAM  = 'NEAR',
                    AFFE      = _F (CHAM_GD   = VALLEY,
                                    MODEL    = modele2,
                                    INST      = 0.     ),)

# calculation of the compressive force resulting on the disc from the first
grid
tab1=POST_ELEM (INTEGRAL = _F (GROUP_MA = 'disc',
                               DEJA_INTEGRE = 'NOT',
                NOM_CHAM = 'NEAR',
                NOM_CMP  = 'CLOSE'  ),
          RESULT  = RES_PRES,
          MODEL    = modele2   )

IMPR_TABLE (TABLE=tab1)

# projection of the field of pressure since the disc towards the structure
calculee
RES_PROJ=PROJ_CHAMP (METHODE='COLLOCATION',
                     RESULTAT=RES_PRES,
                     MODELE_1=modele2,
                     MODELE_2=MODE,
                     DISTANCE_MAX=1.,
                     CAS_FIGURE=' 2.5D',
                     PROL_ZERO=' OUI',
                     VIS_A_VIS=_F (GROUP_MA_1=' disque',
                                   GROUP_MA_2=' Group_1',),
                     TOUT_ORDRE=' OUI',);

# calculation of the resulting compressive force after projection
tab2=POST_ELEM (INTEGRAL = _F (GROUP_MA = 'Group_1',
                NOM_CHAM = 'NEAR',
                               DEJA_INTEGRE = 'NOT',
                NOM_CMP  = 'CLOSE'  ),
          RESULT  = RES_PROJ,
          MODEL    = MODE   )

IMPR_TABLE (TABLE=tab2)

# creation of the load
CHA_PROJ=AFFE_CHAR_MECA (MODELE=MODE,
                         EVOL_CHAR=RES_PROJ,)

################################################
##### fine of creation of load CHA_PROJ #####
################################################

BLOQ=AFFE_CHAR_MECA (MODELE=MODE,
                     DDL_IMPO=_F (GROUP_MA=' Group_3',
                                  DX=0.0,
                                  DY=0.0,
                                  DZ=0.0,),)

RESU=MECA_STATIQUE (MODEL     = MODE,
           CHAM_MATER = SUBDUE,
           EXCIT= (_F (CHARGE=BLOQ,),
              _F (CHARGE=CHA_PROJ,),))
```

```
RESU=CALC_CHAMP (reuse =RESU,
             FORCE=' REAC_NODA',
             RESULTAT=RESU,)

STANLEY ()

END ()
```

4 How to calculate the integral of the constraints or a quantity derived on the edge from a structure 3D

4.1 Objective

The user has a stress field (at the points of Gauss). It wishes to make the integral of each component on the edge of its model or to calculate a derived quantity like one axial moment.

$\int_S \sigma_{xx} ds$

$\int_S \sigma_{xx} . y \, ds$

Integral is calculated by the order POST_ELEM. When one wishes to integrate a quantity derived from a field, this one must be defined (FORMULA) and evaluated (CREA_CHAMP) beforehand.
The constraints at the points of Gauss are not calculated directly on the faces, also the user is seen confronted with some difficulties. A receipt was established to be able to carry out this calculation. It is presented below.

For more details, the CAS-test will be consulted tplv07a who implements this receipt within a thermal framework in order to calculate outgoing flow on an edge.

4.2 Receipt

4.2.1 Stage 1: calculation of the stress field 3D to the nodes

First of all, it is necessary to calculate the stress fields to the nodes.

```
CALC_CHAMP (CONTRAINTE=' SIGM_NOEU', RESULTAT=…,)
```

The following stage consists in building a neutral field of which values:
- are localised at the points of Gauss of the meshs (2D and 3D), and
- correspond to the values of the constraints.

4.2.2 Stage 2: calculation of a neutral field corresponding to the constraints

It is supposed that the model 3D (MO3D) who was used to calculate contains besides the elements "3D", of the elements of edge (facets) on all his edges.

4.2.2.1 Case of the integration of an existing field

1) Extraction of the stress field to the nodes:

```
SIGNO=CRÉA_CHAMP (OPERATION=' EXTR', TYPE_CHAM=' NOEU_SIEF_R',
                  NOM_CHAM=' SIEF_NOEU', RESULTAT=…, INST=…)
```

2) Transformation of the field SIGNO in neutral field:

```
NEUTNO=CRÉA_CHAMP (OPERATION=' ASSE', TYPE_CHAM=' NOEU_NEUT_R',
MODELE=MO3D,
                   PROL_ZERO=' OUI', ASSE=_F (TOUT=' OUI', CHAM_GD=SIGNO,
                   NOM_CMP= ('SIXX', 'SIYY', 'SIZZ'),
                   NOM_CMP_RESU= ('X1', 'X2', 'X3'),)
```

4.2.2.2 Case of the integration of a derived field (for example for the one axial moment calculation or a heat flux)

When the quantity to be integrated does not exist directly, it should be built. The approach is then exactly the same one as that described with the §2.2.1. One will start by defining the intégrande with the operator FORMULA, then a field of functions of the type NEUT (CREA_CHAMP/OPERATION=' ASSE') and finally it will be evaluated (CREA_CHAMP/OPERATION=' EVAL').
One will also refer to the CAS-test tplv07a for an example in thermics.

4.2.3 Stage 3: calculation of the integral on the edge

It is enough to integrate the field NEUTNO with the meshs of edge. For example, for the face 'HIGH', the order is the following one:

```
INT=POST_ELEM (CHAM_GD=NEUTNO, MODELE=MOD3D,
               INTEGRALE=_F (GROUP_MA=' HAUT',
                             NOM_CMP= ('X1', 'X2', 'X3'),),)
```

4.3 Orders used

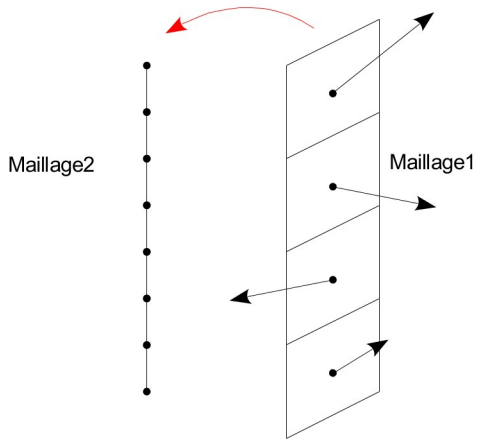
Orders	Functionality
CALC_CHAMP	Creation of the stress fields to the nodes by elements at the various moments.
CALC_CHAMP	Creation of the stress fields to the nodes at the various moments.
AFFE_MODELE	Creation of a model 3D.
CREA_CHAMP/EXTR	Extraction of the stress field to the nodes at the moment INST.
(FORMULA)	Definition of the intégrande if necessary.
CREA_CHAMP/ASSE (CREA_CHAMP/AFFE)	Transformation of the stress field into neutral field. Creation of a neutral field of functions.
(CREA_CHAMP/EVAL)	Evaluation of the field of functions if necessary.
POST_ELEM	Calculation of the integral by specifying the groups of meshs of face.

5 How to read again a constant field by elements and to project it with the nodes of the grid

5.1 Objective

The user has a constant field with 3 components by elements with format MED. It wishes to be able to project it on the nodes of another grid.

More precisely, 2 grids are considered. The first is a grid made up of surface meshs on which a constant field by mesh was calculated (resulting for example from a fluid calculation). The second is a linear grid (representative for example a line of piping on which one wishes to project the computed field on the first grid). The challenge is thus to read again the field on grid 1 and to transfer it to the nodes from grid 2.



So that the technique presented here functions, it is enough simply that the grids are parallel one with the other (for example, it is necessary that they are both following axis Z) and that they have a beach of joint coordinates along the axis whereby they are parallel.

5.2 Method

5.2.1 Second reading of file MED

The second reading is done in two stages:

1) Second reading of grid 1:

```
MAIL1 = LIRE_MAILLAGE (FORMAT = 'MED',);
```

2) Second reading of the fields with LIRE_CHAMP. Indeed, it is LIRE_CHAMP who most simply allows to read again constant fields by elements via the production of the map (within the meaning of *Code_Aster*). The field has to read again is called here 'ForceLineique', one stores it in a map with 3 components of the indicating type of error (this last element is anecdotic because one makes use of it only of container):

```
CH1=LIRE_CHAMP (FORMAT      = 'MED',
                GRID    = MAIL1,
                UNIT       = 20,
                NOM_MED     = 'ForceLineique',
                INST        = 0.0001,
                TYPE_CHAM   = 'CART_ERRE_R',
                NOM_CMP     = ('ERREST', 'NUEST', 'SIGCAL',),
```

```
                    NOM_CMP_MED = ('FX', 'FY', 'FZ',),);
```

5.2.2 Creation of the result and projection of the field

To be able to project the field using PROJ_CHAMP, it should initially be arranged in a structure of data result by using the order CREA_RESU :

1) Creation of the result:

```
EVOL1=CRÉA_RESU (OPERATION = 'AFFE',
                 TYPE_RESU = 'EVOL_NOLI',
                 NOM_CHAM  = 'ERME_ELEM',
                 AFFE      = (_F (CHAM_GD = CH1,
                                  INST    = 0.0001,),),);
```

2) Projection:

```
EVOL2=PROJ_CHAMP (RESULT  = EVOL1,
                  NOM_CHAM  = 'ERME_ELEM',
                  MODELE_1  = MO1,
                  MODELE_2  = MO2,
                  TYPE_CHAM = 'NOEU',);
```

where MO1 and MO2 are models resting respectively on grid 1 and the grid 2 (whose second reading is not clarified here). These models must be coherent between them and obviously with respect to the use which one wishes to make of the field later. So their creation is not part of this explanation because it depends on the required application.

With final, one now obtains a new structure of data result containing a field with the nodes defines on grid 2. A last stage (optional) consists in modifying the noun and the type of the field in order to obtain a true field with the nodes. In the continuation, it is what one will seek to obtain by converting the field of error obtained into field of displacement (field to the nodes with 3 components).

5.2.3 Modification of the type of the field

The last stage breaks up into two under-stages:

1) Extraction of the field:

```
CHTMP1 = CREA_CHAMP (OPERATION = 'EXTR',
                     NOM_CHAM  = 'ERME_ELEM',
                     TYPE_CHAM = 'NOEU_ERRE_R',
                     RESULT  = EVOL2,
                     INST      = 0.0001,);
```

2) Conversion of the field:

```
CHNOFIN = CREA_CHAMP (OPERATION    = 'ADZE',
                      TYPE_CHAM    = 'NOEU_DEPL_R',
                      MODEL       = MO2,
                      ADZE         = (_F (CHAM_GD      = CHTMP1,
                                          COEF_R       = 1.0,
                                          ALL          = 'YES',
                                          NOM_CMP      = ('ERREST',
                                                          'NUEST',
                                                          'SIGCAL'),
                                          NOM_CMP_RESU = ('DX',
                                                          'DY',
                                                          'DZ'),),),);
```

6 How to use the value of the Eigen frequencies calculated to evaluate damping

One wishes to interpolate a function of damping $F(FREQ)=\text{f\_amor}$ on the Eigen frequencies of a model.

The frequencies and clean modes are obtained by CALC_MODES.

```
modes = CALC_MODES (…)
```

One extracts only the list from the frequencies calculated while making (one obtains a table made up of a column with the sequence numbers and a column with the frequencies):

```
= RECU_TABLE (RESU=modes,
                  NOM_PARA=' FREQ')
```

The trick consists in creating a function identity on the list of the frequencies $G(FREQ)=FREQ$ by recovering the values in the table:

```
g_freq = RECU_FONCTION (TABLE=tab,
                        PARA_X=' FREQ',
                        PARA_Y=' FREQ')
```

The damping evaluated on the Eigen frequencies is thus `f_amor = F (G (FREQ))`, that is to say:

```
amor = CALC_FONCTION (COMPOSE=_F (FONC_PARA=g_freq,
                                  FONC_RESU=f_amor))
```

This example is available in `sdld22a.com1`.