Operator CREA_CHAMP

1 Goal

To create a field of the type cham_no, map or cham_elem. One can create a field

- by assignment of values on nodes or meshs,
- by assembling (and/or combining) pieces of existing fields,
- by evaluating the functions of a field of functions to make a field of realities of it,
- by modifying the geometrical representation of a field (passage nodes ⇔ points of Gauss for example),
- by extracting a field from a structure of data of the type SD_RESULTAT (evol_ther, evol_noli, mode_meca,...).
- starting from the values contained in a structure of data of the type TABLE,
- …

One can also make use of this order to combine several fields linearly, but one can make "complicated" combinations more: multiplication of fields…

Product a structure of data of the type cham_no or map or cham_elem

Note: The order is not D-entering except for Lbe operationS ADZE and COMB.
In this case, the keyword CHAM_GD is obligatory.

Contents

2 Syntax

```
ch2 [*] = CREA_CHAMP   (

      ◊  CHAM_GD   = ch1  ,                                        [field]

      ◊  TITLE =   title  ,                                        [l_Kn]
      ◊  INFORMATION  =  /1  ,
   [DEFECT]
                    /2  ,
      ♦  TYPE_CHAM =  /'NOEU_xxx',
                      /'CART_xxx',
                      /'ELNO_xxx',
                      /'ELGA_xxx',
                      /'ELEM_xxx',

      # to impose the classification of the field result
      # (only allowed for the cham_no):
      ◊  /  NUME_DDL = naked ,
   [nume_ddl]
          / CHAM_NO = cnonu,                                       [cham_no]

      # If the "structure" of the field result (cham_elem or cham_no)
      # requires values which one cannot determine:
      ◊  PROL_ZERO =      /  'NOT',                                [DEFECT]
                          /  'YES',

      # If one wishes to create a cham_elem at "under-points" for which
      # all the under-points (of the same point) have the same value:
      ◊  AFFE_SP= _F  (  ♦  CARA_ELEM    =  carele,),              [cara_elem]


      /  OPERATION =            'AFFE',
      #  ==============================

         ♦  /  GRID   =         my,                    [grid]

      .     /  MODEL     =         Mo,                    [model]

         ♦  AFFE= (_F (  ♦  /  ALL    =  'YES',
                           /     |  GROUP_MA  =  grma   ,  [l_grma]
                                 |  GROUP_NO  =  grno   ,  [l_grno]

                         ♦  NOM_CMP   = lcmp ,              [l_K8]
                         ♦  /  VALE   = lvaler  ,           [l_R]
                            /  VALE_I = lvalei  ,           [l_I]
                            /  VALE_C = lvalec  ,           [l_C]
                            /  VALE_F = lvalef  ,     /     [l_fonction]
                                                      /     [l_formule]
                      ),),
```

```
      /  OPERATION =            'ADZE',
      #  ============================

         ♦  /  GRID   =         my ,                      [grid]
            /  MODEL     =        Mo ,                     [model]
               ◊  OPTION    =     option                   [KN]

         ♦  ASSE= (_F (  ♦      /  ALL    =  'YES',
                                /  |  GROUP_MA  =  grma   ,  [l_grma]
                                   |  GROUP_NO  =  grno   ,  [l_grno]

                         ♦  CHAM_GD      = ch1  ,           [field]

                         ◊  NOM_CMP      = lcmp ,           [l_K8]
                            ◊  NOM_CMP_RESU = lcmp_resu,    [l_K8]

                         ◊  /  OFFICE PLURALITY =         'NOT'  ,
[DEFECT]
                            /  OFFICE PLURALITY =         'YES'  ,
                               ◊  /  COEF_R =  /  1. ,      [DEFECT]
                                               /  coefr, [R]
                                  /  COEF_C =     coefc, [C]
                     ),),

      /  OPERATION =            'COMB',
      #  ============================
         ♦  COMB= (_F (
               ♦  CHAM_GD      = ch1  ,                   [cham_no]
               ♦  COEF_R       =  coefr  ,                [R]
         ),),

      /  OPERATION =            'EVAL',
      #  ============================

         ♦  CHAM_F =  ch_fonc   ,           /      [cham_no (NEUT_F)]
                                            /      [cham_elem (NEUT_F)]
         ♦  CHAM_PARA=   will l_ch_para   ,              [l_champ]

      /  OPERATION =            'DISC',
      #  ============================
         #  if the field result is a CHAM_ELEM:
         ◊  MODEL     =         Mo,                      [model]
            ◊  OPTION    = option                         [KN]

         ♦  CHAM_GD   =      ch1 ,                       [field]

      /  OPERATION =            'NORMAL',
      #  ===================================
         ♦  MODEL           =         Mo,                [model]
         ♦  GROUP_MA        =         l_gma,             [l_group_ma]
```

```
        /   OPERATION =            'R2C',
        #   ================================
            # to transform a real field with the nodes into complex field
            # (with worthless imaginary part)
            ♦  CHAM_GD             =         chR,                [cham_no]

        /   OPERATION =            'C 2R',
        #   ================================
            # to transform a field with the nodes complexes in real field:
            ♦  CHAM_GD      =         chC,                       [cham_no]
            ♦  PART         =      /   'REAL',   # left real
                                   /   'IMAG',   # left imaginary
                                   /   'MODULE', # absolute value
                                   /   'PHASE',  # "phase" (in degrees)


        /   OPERATION =            'EXTR',
        #   ==============================

            ♦  #  extraction of the field of geometry of a grid:
               /  ♦  GRID = my  ,                            [grid]
                  ♦  NOM_CHAM =    'GEOMETRY',

               #  extraction of the curvilinear field of X-coordinate of a
grid:
               /  ♦  GRID = my  ,                            [grid]
                  ♦  NOM_CHAM =    'ABSC_CURV',

               #  extraction in a table:
               /  ♦   TABLE    =  tabl   ,                      [table]
                  ♦  /  GRID   =  my   ,                    [grid]
                     /  MODEL  =  Mo   ,                      [model]
                        ◊  OPTION =  option  ,                [kN]

               #  extraction of the "level set" of a SD fiss_xfem:
               /  ♦  CRACK =   fxfem  ,                       [fiss_xfem]
                  ♦  NOM_CHAM =    /   'LTNO',
                                   /   'LNNO',
                                   /   'GRLTNO',
                                   /   'GRLNNO',
                                   /   'STNO',
                                   /   'STNOR',
                                   /   'BASLOC',

               #  extraction of a field of a SD cara_elem:
               /  ♦  CARA_ELEM =  carele ,                    [cara_elem]
                  ♦  NOM_CHAM =   nomch,                   [KN]

               #  extraction of a field of a SD char_meca:
               /  ♦  LOAD   =  load   ,                      [char_meca]
                  ♦  NOM_CHAM  =  nomch,                    [KN]
```

```
#  extraction of a field of a SD_RESULTAT:
             /  ♦  RESULT =  resu,
                ♦  NOM_CHAM =    /   'ACCE',
                                 /   … (cf [3.9.4]),

                ♦  /  #  Selection of a sequence number in the SD_RESULTAT
                   /  NUME_ORDRE    =  nuordr ,            [I]
                   /  NUME_MODE     =  numode ,            [I]
                   /  NOEUD_CMP     =  (node, cmp),        [l_K8]
                   /  NOM_CAS       =  nocas,              [KN]
                   /  ANGLE         =  alpha ,             [R]
                   /  ♦  /  INST          =  inst,         [R]
                      /  FREQ          =  freq,            [R]

                         ◊  /  CRITERION =  / 'RELATIVE',[DEFECT]
                                ◊  PRECISION =  / prec ,  [R]
                                                / 1.0E-6, [DEFECT]
                            /  CRITERION =  / 'ABSOLUTE',
                               ♦  PRECISION =    prec ,  [R]

                         ◊  Interpol     =  / 'NOT'   ,   [DEFECT]
                                            / 'FLAX'  ,

                   /  #  Calculation of a field containing the "extrema"
of one
                      #  SD_RESULTAT
                      ♦  TYPE_MAXI =  /  'MAXIMUM',
                                      /  'MINI',
                                      /  'MAXI_ABS',
                                      /  'MINI_ABS',
                                      /  'NORM_TRAN',

                       # if TYPE_MAXI is worth MAXI_ABS or MINI_ABS
                      ◊  TYPE_RESU =  /  'VALE' ,            [DEFECT]
                                      /  'INST' ,
                                      /  'VALE_ABS',
                       # if not
                      ◊  TYPE_RESU =  /  'VALE' ,            [DEFECT]
                                      /  'INST' ,

                      ◊  /  TOUT_ORDRE = 'YES'  ,            [DEFECT]
                         /  LIST_INST  = linst  ,            [listr8]
                         /  LIST_FREQ  = lfreq  ,            [listr8]

                            ◊  /  CRITERION =  / 'RELATIVE',[DEFECT]
                                   ◊  PRECISION =  / prec ,  [R]
                                                   / 1.0E-6, [DEFECT]
                               /  CRITERION =  / 'ABSOLUTE',
                                  ♦  PRECISION =    prec ,  [R]

                      )

If     TYPE_CHAM    =      'NOEU_xxx'          then   [*] =  cham_no
                           'CART_xxx'                        map
                           'ELNO_xxx'                        cham_elem
                           'ELGA_xxx'                        cham_elem
                           'ELEM_xxx'                        cham_elem
```

3 Operands

3.1 Operands generals

3.1.1 Operand TYPE_CHAM, assignment of a type to the field result

This keyword (obligatory) is initially used to typify the field result of the order. It is made of 2 "words" connected by a "underlined white" (_) :

TYPE_CHAM= '' where:

XXXX=	/	'NOEU'	Field with the nodes	(cham_no)
	/	'CART'	Constant field by mesh	(map)
	/	'ELNO'	Field by elements with the nodes	(cham_elem)
	/	'ELGA'	Field by elements at the points of Gauss	(cham_elem)
	/	'ELEM'	Constant field by element	(cham_elem)
GD=	/	'DEPL_R'	displacement	
	/	'SIEF_R'	constraint	
	/	'TEMP_R'	temperature	
	/	'FLUX_R'	flow	
	/	…		

The type of the field result is deduced from information given by the user. For example:

```
TYPE_CHAM= 'NOEU_DEPL_R'          - >    cham_no   (DEPL_R)
TYPE_CHAM= 'CART_SIEF_R'          - >    map    (SIEF_R)
TYPE_CHAM= 'ELNO_EPSI_R'          - >    cham_elem (EPSI_R)
TYPE_CHAM= 'ELGA_VARI_R'          - >    cham_elem (VARI_R)
```

This keyword is also used to specify (for the order) which must be the nature of the field wanted as a result. It is essential for the operations 'AFFE', 'ADZE' and 'DISC'.

Examples:

```
  OPERATION= 'AFFE'  + TYPE_CHAM= 'CART_DEPL_R'   => a map of DEPL_R.
  OPERATION= 'ADZE'  + TYPE_CHAM= 'NOEU_EPSI_R'   => a cham_no of EPSI_R.
  OPERATION= 'DISC'  + TYPE_CHAM= 'NOEU_SIEF_R'   => a cham_no of SIEF_R.
```

There are only two operations for which this keyword is a constraint useless (but obligatory!) for the user (OPERATION=' EVAL' and OPERATION=' EXTR') because for these two operations, the nature of the field result is imposed by the choice of the operation.

The information of the keyword TYPE_CHAM is (unfortunately) tiresome forOPERATION=' EXTR'. He results from the keyword NOM_CHAM. The correspondence is given in [3.9.1].

Notice important

The possibility of creating cham_elem of any size is conditioned by the level of development (data-processing) of the types of finite elements of the model. All is not yet possible; for example, to create one cham_elem of FLUX_R on a model containing the elements DKT , it is necessary that this finite element envisaged to do it (what is not the case today).

One cannot give here the precise list of the sizes allowed for each type of finite element. One will be satisfied to say roughly:

- *for the isoparametric elements of mechanics, the sizes are allowed: GEOM_R , INST_R , NEUT_R , NEUT_F , EPSI_R , SIEF_R , VARI_R , DOMMAG and HYDR_R ,*
- *for the isoparametric elements of thermics, the sizes are allowed: GEOM_R , INST_R , NEUT_R and NEUT_F .*

3.1.2 Operand CHAM_GD

With the operations ADZE and COMB, one can re-use an existing field. In this case, it is necessary to indicate here which field is re-used.

3.1.3 Operand GRID = netted

When one is created CHAM_NO or one MAP, it is necessary to specify in general on which grid this field will be based. For that, the keyword is used GRID.

3.1.4 Operand PROL_ZERO

When one is created CHAM_ELEM, the values existing in the field are determined by the finite elements of the model. For example, a field SIGM_ELGA on a model 2D **must** to contain the 4 components SIXX, SIYY, SIZZ and SIXY.
If the construction of the field does not make it possible to calculate all the expected values, one is confronted with a problem. If the keyword PROL_ZERO is worth 'YES', the missing values will be put at zero.
If the keyword PROL_ZERO is worth 'NOT', the code will stop in error.

This problem also relates to (but more rarely) them CHAM_NO when one wants to force the classification of their components (see the keywords NUME_DDL and CHAM_NO).

3.1.5 Operands MODEL, OPTION

When one is created CHAM_ELEM, it is necessary to specify on which finite elements the field will be defined. For that, the keyword is used MODEL.

To describe the structure of one CHAM_ELEM, it is not enough to give (via MODEL) a kind of element for each mesh, because a kind of element can know several "forms" for a given size. To create the desired field, the user can use the keyword OPTION. If for example, he writes: X= CREA_CHAMP (… MODELE=mo, OPTION=' SIEF_ELGA',…) , the field created by CREA_CHAMP the same form will have that if it had been calculated by CALC_CHAMP / CONTRAINTE=' SIEF_ELGA'. If the user does not employ the keyword OPTION, CREA_CHAMP (if it can it) a form by default will choose.

Note: for the operation 'ADZE', when one assembles cham_elem, it is better in general not to provide the keyword OPTION. The option which will be selected will be that of the field provided in the 1st occurrence of the keyword factor ADZE if the size associated with this field is the same one as that of the field result.

3.1.6 Keywords AFFE_SP/CARA_ELEM

The keyword factor (not répétable) AFFE_SP allows to ask for the creation of one cham_elem at "under-points" (for the elements of structure: multi-layer hulls, multifibre beams,…).
In cham_elem product, each point (node or not of Gauss) will be represented by N under-points (N being determined by the choices of the user in the order AFFE_CARA_ELEM).
N under-points of a point will carry the same components (with the same values).

The keyword CARA_ELEM is obligatory: it is the name of the structure of data of the type cara_elem who allows to determine the number of the under-points.

3.1.7 Keywords NUME_DDL and CHAM_NO

These two keywords make it possible to impose a classification for the field result (if this one is one CHAM_NO). One gives via these keywords a "model" of classification for the field result.

If one gives Naked NUME_DDL=, one will take as classification that of naked. This possibility is valid only for the fields of displacements (phenomenon 'MECHANICS') or for fields of temperature (phenomenon'THERMICS') or of the fields of acoustic pressure (phenomenon'ACOUSTICS').

If one gives CHAM_NO= chno, one will take the numeration of chno.

So Lagrange resulting from boundary conditions dualized are present, a checking is made by the operator. Indeed, if the fields considered do not satisfy the same limiting conditions, they are not compatible. An alarm is then emitted: Lagrange will be put at zero in the resulting field.

Notice on the disk space used:

Sometimes these 2 keywords make it possible to save much place on the "Total" basis. When for example, one extracts from many cham_no of one SD RESULT , if one does not use one of these keywords, one duplicates the profile of the field for each one of them. If one uses one of these 2 keywords, all these fields will be based on the profile contained in chno (or naked).

3.1.8 Operand TITLE = titr

Title which one wants to give to the field result [U4.03.01].

3.1.9 Operand INFORMATION = /1 /2

INFORMATION = 1
No impression.

INFORMATION = 2
Impression on the file 'MESSAGE' field result.

3.2 Operand OPERATION =/'AFFE'/'ADZE'/'EVAL'/'DISC'/'EXTR'/'R2C'/'C 2R'/'COMB'

This operand is used to choose the "mode" of manufacturing of the field result. One can create a field:

- by assignment of values on nodes or meshs (OPERATION=' AFFE'),
- by assembling pieces of fields defined on pieces of grids (OPERATION=' ASSE'),
- by modifying the geometrical representation (discretization) of a field (passage nodes <-> points of Gauss for example) (OPERATION=' DISC'),
- by extracting a field from a SD of the type SD_RESULTAT (evol_ther, evol_noli, mode_meca,...) (OPERATION=' EXTR').
- while extracting from the digital values of a table whose columns have preset names: 'MESH', 'NODE'...
- by combining fields linearly (OPERATION=' ASSE'),
- in "combining" (multiplication, exponential,...) fields (OPERATION=' EVAL'),
- by evaluating the functions of a field of functions to make a field of realities of it (OPERATION=' EVAL'),
- by transforming a real field with the nodes into field complexes (or reciprocally) (OPERATION=' R2C'/'C 2R'),
- by making a linear combination of several cham_no having same classification (OPERATION=' COMB'). Unlike the operation 'ADZE', the field result will preserve the ddls of Lagrange associated with the dualisation with the boundary conditions.

3.3 Operands for OPERATION = 'AFFE'

This operation makes it possible to affect values (real, whole, complex or function) on geometrical entities (nodes or meshs) of a grid.

The size associated with the field is implicitly given by the keyword TYPE_CHAM (above).

3.3.1 Keyword factor AFFE

The operands are gathered under the keyword factor AFFE. This keyword is répétable. The principle of overload is applied between the various occurrences of the keyword AFFE : if one geometrical entity is affected several times, the last assignment carries it.

3.3.2 Operands TOUT=' OUI', GROUP_MA and GROUP_NO

The geometrical entities that one wants to affect are given by the operands TOUT=' OUI', GROUP_MA, and GROUP_NO.

If TYPE_CHAM=' NOEU_xxx', nodes are affected; the use of operand GROUP_MA is possible and means that one affects all the nodes of the specified meshs.
If TYPE_CHAM: 'EL. _xxx' (or 'CART_xxx'), meshs are affected; the use of operand GROUP_NO is then prohibited.

Note:
For the keyword factor AFFE, the keyword TOUT=' OUI', wants to say:
- "all them nodes grid" for cham_no,
- "all meshs of the grid" for cards,
- "all elements of the model" for cham_elem,

3.3.3 Operand NOM_CMP

The names of the components which one wants to affect are given by the operand NOM_CMP.
If the size is 'VARI_R', the components must be named 'V1', 'v2', 'V3',...
If the size is 'VARI_R', and that one chooses PROL_ZERO=' OUI', the components whose number is lower than greatest affected number the are assigned to zero. For more details to see the document [U2.01.09].

3.3.4 Operands VALE, VALE_I, VALE_C or VALE_F

The values to be affected are given by the operands VALE, VALE_I, VALE_C or VALE_F according to nature (reality, entirety, complex, function (or formulates)) components of the size (DEPL_R : reality, DEPL_C : complex, TEMP_F : function/formula,….).

3.3.5 Notice

The rule of remanence (see U1.03.00) applies for the various components which one can affect.

3.3.6 Examples

Creation of a field to the nodes of displacement. One wants to impose the classification of the field (that of cnomod) :

```
DEPL1 = CREA_CHAMP (OPERATION= 'AFFE',
          TYPE_CHAM=' NOEU_DEPL_R', GRID = MY  , CHAM_NO= CNOMOD,
          AFFE= (
          _F  (TOUT=' OUI', NOM_CMP= ('DX', 'DY', 'DZ'), VALE= (0. , 0. ,
0.),),
          _F  (GROUP_MA= ('GM1', 'GM2'), NOM_CMP= 'DX', VALE= 3.5e-2),
          _F  (GROUP_NO= ('GN5', 'GN7', 'GN9'), NOM_CMP= 'DY', VALE= 1.6e-
2),
                    )
)
```

Creation of a map of temperature (functions):

```
TEMPF = CREA_CHAMP (OPERATION= 'AFFE',
          TYPE_CHAM=' CART_TEMP_F',  GRID = MY,
          AFFE= (   _F (TOUT=' OUI', NOM_CMP= ('TEMP'), VALE_F= F1),
                 _F (GROUP_MA= ('GM1','GM2'), NOM_CMP= ('TEMP'), VALE_F= F2),)
)
```

3.4 Operands for OPERATION = 'ADZE'

3.4.1 General information

This operator "assembles" "pieces of fields" to manufacture new. Each occurrence of the keyword ADZE a piece of field defines. One calls a piece of field, the restriction of an existing field (map / cham_no or cham_elem) on a set of geometrical entities (meshs or nodes) and on a set of components.

There is a principle of overload of the occurrences of the keyword ADZE if the pieces recover the ones the others.

Currently, one can manufacture:

- one cham_no by assembling pieces of cham_no.
- one cham_elem by assembling pieces of cham_elem and/or of cards.
- one map by assembling pieces of cards and/or of cham_elem/ELEM.

The operation 'ADZE'also allows to change the size associated with a field; for example to transform a field of deformation (EPSI_R) in stress field (SIEF_R). For that it is necessary to use the keywords NOM_CMP and NOM_CMP_RESU.

The assembly of the pieces of fields can be done by cumulating the pieces (keywords OFFICE PLURALITY and COEF_R). That makes it possible to use this order to make linear combinations of CHAM_NO or of CHAM_ELEM.

3.4.2 Operands GRID, MODEL

Even use that for OPERATION= 'AFFE' [§3.3.1] and [§3.3.2].

3.4.3 Operands for the keyword factor ADZE

Each occurrence of the keyword factor ADZE allows to define a piece of field which one assembles in the field result.

3.4.3.1 Operand CHAM_GD = ch1

ch1 is the field (existing) with which one wants to manufacture a piece of field.

3.4.3.2 Operands ALL = ' OUI', GROUP_MA and GROUP_NO

These operands are used to define the geometrical restriction of the field ch1. If ch1 is one CHAM_NO, one can use all these operands. If ch1 is one CHAM_ELEM (or one MAP), one cannot use operands GROUP_NO.
The keyword TOUT=' OUI' wants to say here: "all nodes, meshs or elements which carry components (of the list NOM_CMP) in the field ch1 ".

3.4.3.3 Operands NOM_CMP and NOM_CMP_RESU

The operand NOM_CMP is used to define the components on which one wants to restrict the field ch1. If NOM_CMP is absent, one takes all the components of ch1.

The operand NOM_CMP_RESU is used to re-elect (if it is wished) the components of ch1. If NOM_CMP_RESU is provided, NOM_CMP the being also must and the two lists in correspondence must be of the same length.

Example 1: to transform a field of EPSI_R in field of VARI_R

```
CHVARI=CRÉA_CHAMP (OPERATION=' ASSE', TYPE_CHAM=' ELGA_VARI_R',
   MODELE=MO,
   ASSE=_F (CHAM_GD=CHEPSI, TOUT=' OUI',
      NOM_CMP  = ('EPXX', 'EPYY'),
      NOM_CMP_RESU= ('V3', 'V1'),   ),)
```

Example 2: to permute the cmps SIXX and SIYY of a field of SIEF_R

```
CHS2=CRÉA_CHAMP (OPERATION=' ASSE', TYPE=' NOEU_SIEF_R',
   MAILLAGE= MY,
   ASSE=_F (CHAM_GD=CHS1, TOUT=' OUI',
      NOM_CMP  = ('SIXX', 'SIYY'),
      NOM_CMP_RESU= ('SIYY', 'SIXX',),  ))
```

3.4.3.4 Operands OFFICE PLURALITY, COEF_R and COEF_C

The operand CUMUL=' OUI' wants to say that the values of the occurrence concerned will be added with the possible already existing values.

If CUMUL=' NON', the affected value replaces the value possibly already present (CUMUL=' OUI' is invalid for the fields of "text" (k8/k16,...) of course).

The operand COEF_R = coefr allows multiplication of the piece of field by the real coefficient coefr before assembling it with the field result.

Example:

To manufacture the cham_elem : ch3 = 2. * ch1 + 3. * ch2

```
CH3= CREA_CHAMP (OPERATION= 'ADZE',
          MODEL = MO    ,  TYPE_CHAM = ' ELGA_EPSI_R',
          ADZE = (_F ( CHAM_GD = CH1, ALL = ' OUI',
                       CUMUL=' OUI', COEF_R = 2.),
                   _F ( CHAM_GD = CH2, ALL = ' OUI',
   CUMUL=' OUI', COEF_R = 3.),)
 )
```

Notice concerning the complex fields

The keyword COEF_C is accepted only if the field result (CH3) and the fields arguments (CH1 and CH2) are all complex. To make a linear combination with complex coefficients of real fields, it is necessary to transform as a preliminary the real fields into complex fields. See OPERATION = 'R2C'.

3.4.4 Examples

Example 1

To manufacture one cham_no of temperature by extracting a field already calculated (in one evol_ther) and by redefining it (with 25. degrees) on the group of meshs soudur1 .

```
CH1= CREA_CHAMP (OPERATION= 'EXTR', TYPE_CHAM=' NOEU_TEMP_R',
          RESULTAT= EVOTH,  NOM_CHAM= 'TEMP', INST = 12.)
CH2= CREA_CHAMP (OPERATION = 'AFFE', TYPE_CHAM=' NOEU_TEMP_R',
          GRID =MA,
          AFFE=_F (ALL = ' OUI', NOM_CMP = ' TEMP', VALE = 25.) )

CH3= CREA_CHAMP (OPERATION = 'ADZE',
          GRID = MY, TYPE_CHAM = ' NOEU_TEMP_R',
   ADZE = (_F (CHAM_GD = CH1, ALL = ' OUI',),
                  _F (CHAM_GD = CH2, GROUP_MA = SOUDUR1),)
```

)

Example 2:

To manufacture one cham_elem of VARI_R (to use it as an initial state for STAT_NON_LINE) while recovering internal variables (6 and 8) of a law of behavior to make of them variables 1 and 2 of (new) the law of behavior which will be used in STAT_NON_LINE to come.

```
CH1= CREA_CHAMP (OPERATION= 'EXTR', TYPE_CHAM=' ELGA_VARI_R',
          RESULTAT= STNL,  NOM_CHAM= 'VARI_ELGA', INST = 4.)

CH2= CREA_CHAMP (OPERATION= 'ADZE',
          MODEL = MO    ,   TYPE_CHAM = ' ELGA_VARI_R',
          ADZE = _F (CHAM_GD = CH1, ALL = ' OUI',
                     NOM_CMP  =    ('V6', 'V8'),
                     NOM_CMP_RESU = ('V1', 'V2'),  ))
```

3.5 Operands for the keyword factor COMB

This keyword makes it possible to calculate the linear combination of several cham_no having same classification. Unlike the operation 'ADZE', the field result will also contain the coefficients of Lagrange corresponding to the dualisation of the boundary conditions.

Each occurrence of the keyword factor COMB allows to define an element of the linear combination.

3.5.1 Operand CHAM_GD = ch1

ch1 is the cham_no (existing) that one wants to combine linearly.

3.5.2 Operand COEF_R = coefr

coefr is the real coefficient applied to ch1 for the combination.

3.5.3 Example

To calculate C = 1.*A - 2.*B, one writes:

```
C= CREA_CHAMP (OPERATION=' COMB', TYPE_CHAM=' NOEU_DEPL_R',
       COMB= (
           F (CHAM_GD=A, COEF_R= 1.),
           F (CHAM_GD=B, COEF_R=-2.),
       ))
```

3.6 Operands for OPERATION = 'EVAL'

This operation is used to transform a field of functions into fields of realities by evaluating the functions of the field of functions.

The field of functions is obligatorily a field of the size 'NEUT_F' and the field result will be always a field of 'NEUT_R'. This field could be transformed into field of another unspecified size by calling one second time on the order CREA_CHAMP/OPERATION=' ASSE'.

An example of the use of the operation 'EVAL' is given in the document [U2.01.09] "analytical Definition of a stress field…"

3.6.1 Operand CHAM_F= chf

chf is the name of the field of functions to be evaluated (CHAM_NO, MAP or CHAM_ELEM).

3.6.2 Operand CHAM_PARA= will l_chpara

will l_chpara is the list of the fields “parameters” for the evaluation of the functions. All fields of will l_chpara must be discretized in the same way that chf. For example, if chf is one CHAM_ELEM/ELGA, it is necessary that all the fields of will l_chpara are also CHAM_ELEM/ELGA.

It is necessary that the list of the fields parameters is sufficient to allow the evaluation of all the functions referred in chf.

3.6.3 Examples

3.6.3.1 Example 1

One wants to create one cham_elem (SIEF_R) at the points of Gauss whose components are analytical functions of the geometry and time. For this example, it is supposed that one already manufactured two fields at the points of Gauss CHGEOMG : field of geometry and CHINSTG : field of moments.

```
RHO=1000.
G=10.
KP=3.
SIZZ = FORMULA (NOM_PARA = 'Z', VALE = 'RHO*G*Z')
SIXX = FORMULA (NOM_PARA = ('Z', 'INST'), VALE = 'KP*SIZZ (Z) +3.*INST')

# Assignment of the functions:
# ---------------------------------------------
SIG1=CRÉA_CHAMP (OPERATION=' AFFE', TYPE_CHAM=' ELGA_NEUT_F',
     MODELE=MO, PROL_ZERO=' OUI',
     AFFE=_F (TOUT=' OUI', NOM_CMP= ('X1', 'X2'),
                       VALE_F= ('SIXX', 'SIZZ')))

# Evaluation of the functions:
# ---------------------------------------------
SIG2= CREA_CHAMP (OPERATION=' EVAL', TYPE_CHAM=' ELGA_NEUT_R',
      MODELE=MO, CHAM_F=SIG1, CHAM_PARA= (CHGEOMG, CHINSTG)   )

 # transformation of the field of NEUT_R into SIEF_R:
 # ---------------------------------------------
SIG3=CRÉA_CHAMP (OPERATION=' ASSE', TYPE_CHAM=' ELGA_SIEF_R',
      MODELE=MO, PROL_ZERO=' OUI',
      ASSE=_F (TOUT=' OUI',   CHAM_GD=SIG2,
              NOM_CMP=     ('X1'  , 'X2'),
              NOM_CMP_RESU= ('SIXX', 'SIZZ'),
           ))
```

3.6.3.2 Example 2

One wants to calculate a field of temperature to the nodes ($CH3$) containing the product of 2 other fields to the nodes of temperature ($CH1$ and $CH2$)

```
# 1) transformation of cham_no/TEMP_R (CH1 and CH2) into cham_no/NEUT_R:
# ---------------------------------------------------------------------
CH1N=CRÉA_CHAMP (OPERATION=' ASSE', TYPE_CHAM=' NOEU_NEUT_R', MAILLAGE=MA,
           ASSE=_F (ALL = 'YES', CHAM_GD = CH1,
           NOM_CMP = ('TEMP',),  NOM_CMP_RESU = ('X1',)),  )

CH2N=CRÉA_CHAMP (OPERATION=' ASSE', TYPE_CHAM=' NOEU_NEUT_R', MAILLAGE=MA,
           ASSE=_F (ALL = 'YES', CHAM_GD = CH2,
           NOM_CMP = ('TEMP',),  NOM_CMP_RESU = ('X2',)),  )
```

```
# 2) multiplication   CH3N = CH1N * CH2N:
# ---------------------------------------
FMULT = FORMULA (NOM_PARA = ('X1', 'X2'), VALE = 'X1*X2')

CHFMU=CRÉA_CHAMP (OPERATION=' AFFE',  TYPE_CHAM=' NOEU_NEUT_F', MAILLAGE=MA,
                  AFFE=_F (ALL = 'YES', NOM_CMP = 'X3', VALE_F = FMULT))


CH3N=CRÉA_CHAMP (OPERATION=' EVAL', TYPE_CHAM=' NOEU_NEUT_R',
                CHAM_F=CHFMU, CHAM_PARA= (CH1N, CH2N,))


# 3) transformation of cham_no/NEUT_R (CH3N) into cham_no/TEMP_R:
# ---------------------------------------------------------------
CH3=CRÉA_CHAMP (OPERATION=' ASSE', TYPE_CHAM=' NOEU_TEMP_R', MAILLAGE=MA,
           ASSE=_F (ALL = 'YES', CHAM_GD = CH3N,
           NOM_CMP = ('X3',),  NOM_CMP_RESU = ('TEMP',)),  )
```

3.7 Operands for OPERATION = 'DISC'

This operation is used to modify the "discretization" of an existing field. For example, to transform a field with the nodes into fields at the points of Gauss.

The two fields (given and result) are associated with the same size. There exists an exception: the case of the operation DISC who allows to transform one cham_no_VAR2_R (or one carte_VAR2_R) in cham_elem_VARI_R (to see for example the test hplp100b).

The discretization wanted by the user for his field result is indicated by the keyword TYPE_CHAM.

Only the following cases are currently treated by the order:

```
MAP                 - > CHAM_ELEM/ELNO
MAP                 - > CHAM_ELEM/ELGA
MAP                 - > CHAM_ELEM/ELEM
MAP                 - > CHAM_NO
CHAM_NO             - > CHAM_ELEM/ELNO
CHAM_NO             - > CHAM_ELEM/ELGA
CHAM_NO             - > CHAM_ELEM/ELEM
CHAM_ELEM/ELNO      - > CHAM_NO
CHAM_ELEM/ELGA      - > CHAM_ELEM/ELNO
CHAM_ELEM/ELGA      - > CHAM_NO
CHAM_ELEM/ELEM      - > CHAM_ELEM/ELGA
CHAM_ELEM/ELEM      - > CHAM_ELEM/ELNO
```

The ingredients of the treatments are:

MAP/ELEM - > ELxx :

- The value (single) carried by a mesh is recopied on all the points of the mesh.

NOEU - > ELxx :

- the passage of the values of the nodes at the internal points of the mesh is done by using the functions of form of the finite elements of the model.

ELGA - > ELNO :

- the passage of the values of the internal points to the nodes of the mesh is done by using the matrix of extrapolation Gauss- >Nœuds.

ELNO - > NOEU :

- the passage to the values of the nodes of the grid is done by arithmetic mean of the values carried by the nodes of elements convergent.

3.7.1 Operand CHAM_GD= ch1

ch1 is the field which one wants to modify the “discretization”.

3.7.2 keyword MODEL

Even use that for OPERATION = 'AFFE' [3.3.2]

3.7.3 Example

```
# CHXG = FIELD OF GEOMETRY AT THE POINTS OF GAUSS:
# ------------------------------------------------------
CHXN =CRÉA_CHAMP (OPERATION=' EXTR', TYPE_CHAM=' NOEU_GEOM_R',
      NOM_CHAM=' GEOMETRIE', MAILLAGE=MA)
CHXG= CREA_CHAMP (OPERATION=' DISC', TYPE_CHAM=' ELGA_GEOM_R',
      MODELE=MO, CHAM_GD= CHXN   )
```

3.8 Operands for OPERATION = 'NORMAL'

This operation is used to calculate the “normal” vectors with the facets of a model. The user must indicate with the keywords MODEL, and GROUP_MA the name of the model concerned as well as the “facets” of which it wishes to calculate the normals. The “facets” can be elements of “skin” of a grid 3D or elements of plate/hull.
For the grids 2D, the “facets” are linear elements.

The produced field is one cham_no (size GEOM_R) whose components are named X, Y, Z. The normal carried by a node is obtained by realising the normals of the facets concourrant in this node. The “normal” vector is length 1.

3.9 Operands for OPERATION = 'EXTR'

This operation is in general used to extract a field from a SD as type resultat_sdaster.

There are 5 additional possibilities:

- one can extract the field from geometry of the nodes of a grid. The keywords should then be used: GRID = my, NOM_CHAM = 'GEOMETRY' and TYPE_CHAM = 'NOEU_GEOM_R'.
- one can extract the curvilinear field of X-coordinate from a grid. The keywords should then be used: GRID = my, NOM_CHAM = 'ABSC_CURV' and TYPE_CHAM = 'CART_ABSC_R'.
- one can create a field by extracting from a table the values corresponding to parameters of pre-established names: MESH, NODE, NOT, SOUS_POINT, names of the components.
- one can extract the “level set” associated with one crack XFEM. The keywords should then be used:
  CRACK = fiss_xfem, NOM_CHAM = / 'LTNO' / 'LNNO'
  / 'GRLTNO' / 'GRLNNO'
  / 'STNO' / 'STNOR'
  / 'BASLOC'

  and TYPE_CHAM = 'NOEU_NEUT_R' except 'STNO' for which it will be 'NOEU_NEUT_I'

'LTNO'	level set tangent
'LNNO'	level set normal
'GRLTNO'	gradient of the level set tangent
'GRLNNO'	gradient of the level set normal
'STNO'	statute from the nodes, 1 whole value

'STNOR' statute from the nodes, 1 actual value
'BASLOC' base local in bottom of crack, with 6/9 actual values, 2/3 coordinates for the point and 2/3 coordinates for each vector of the local base in 2D/3D

- one can extract the various fields contained in the structures of data cara_elem and char_meca. This functionality is rather reserved for the developers. It makes it possible for example to test the contents of these structures of data.

  Example:
```
CACOQU=CRÉA_CHAMP ( TYPE_CHAM=' CART_CACOQU', OPERATION=' EXTR',
                        CARA_ELEM=CARA,  NOM_CHAM='.CARCOQUE',)
```

  The various fields which one can extract are given in the table below.

concept	NOM_CHAM	size	Type of the field
cara_elem	'.CAFIBR'	CAFI_R	cham_elem
cara_elem	'.CANBSP'	NBSP_I	cham_elem
cara_elem	'.CARARCPO'	CAARPO	map
cara_elem	'.CARCABLE'	CACABL	map
cara_elem	'.CARCOQUE'	CACOQU	map
cara_elem	'.CARDINFO'	CINFDI	map
cara_elem	'.CARDISCA'	CADISA	map
cara_elem	'.CARDISCK'	CADISK	map
cara_elem	'.CARDISCM'	CADISM	map
cara_elem	'.CARDNSCA'	CADISA	map
cara_elem	'.CARDNSCK'	CADISK	map
cara_elem	'.CARDNSCM'	CADISM	map
cara_elem	'.CARGENBA'	CAGNBA	map
cara_elem	'.CARGENPO'	CAGNPO	map
cara_elem	'.CARGEOPO'	CAGEPO	map
cara_elem	'.CARMASSI'	CAMASS	map
cara_elem	'.CARORIEN'	CAORIE	map
cara_elem	'.CARPOUFL'	CAPOUF	map
char_meca	'.CVENTCXF'	VENTCX_F	map
char_meca	'.CHME.EPSIN'	EPSI_R	map
char_meca	'.CHME.F1D1D'	FORC_R	map
char_meca	'.CHME.F1D2D'	FORC_R	map
char_meca	'.CHME.F1D3D'	FORC_R	map
char_meca	'.CHME.F2D2D'	FORC_R	map
char_meca	'.CHME.F2D3D'	FORC_R	map
char_meca	'.CHME.F3D3D'	FORC_R	map
char_meca	'.CHME.FCO2D'	FORC_R	map

char_meca	'.CHME.FCO3D'	FORC_R	map
char_meca	'.CHME.FELEC'	FELECR	map
char_meca	'.CHME.FL101'	FLAPLA	map
char_meca	'.CHME.FL102'	FLAPLA	map
char_meca	'.CHME.FLUX'	FTHM_R	map
char_meca	'.CHME.FORNO'	FORC_R	map
char_meca	'.CHME.IMPE'	IMPE_R	map
char_meca	'.CHME.ONDE'	ONDE_R	map
char_meca	'.CHME.ONDPL'	NEUT_R	map
char_meca	'.CHME.ONDPR'	NEUT_R	map
char_meca	'.CHME.PESAN'	PESA_R	map
char_meca	'.CHME.PRESS'	PRES_R	map
char_meca	'.CHME.ROTAT'	ROTA_R	map
char_meca	'.CHME.SIGIN'	SIEF_R	map
char_meca	'.CHME.SIINT'	NEUT_K8	map
char_meca	'.CHME.VNOR'	SOUR_R	map

3.9.1 Typing of the field result, keyword TYPE_CHAM

The keyword TYPE_CHAM (obligatory) [§3.2] must be well informed. Put except for the case of the extraction in a SD of the type DYNA_HARMO, the keyword TYPE_CHAM results from the reference symbol of the extracted field (NOM_CHAM). The table below gives the correspondence between these two keywords.

NOM_CHAM	TYPE_CHAM	NOM_CHAM	TYPE_CHAM
'ACCE'	'NOEU_DEPL_R'	'ETOT_ELEM'	'ELEM_ENER_R'
'ACCE_ABSOLU'	'NOEU_DEPL_R'	'ETOT_ELGA'	'ELGA_ENER_R'
'COHE_ELEM'	'ELEM_NEUT_R'	'ETOT_ELNO'	'ELGA_ENER_R'
'BEHAVIOR'	'CART_COMPOR'	'REINFORCEMENT'	'ELEM_FER2_R'
'COMPORTHER'	'CART_COMPOR'	'FLHN_ELGA'	'ELGA_FLHN_R'
'DEGE_ELGA'	'ELGA_EPSI_R'	'FLUX_ELGA'	'ELGA_FLUX_R'
'DEGE_ELNO'	'ELNO_EPSI_R'	'FLUX_ELNO'	'ELNO_FLUX_R'
'DEGE_NOEU'	'NOEU_EPSI_R'	'FLUX_NOEU'	'NOEU_FLUX_R'
'DEPL'	'NOEU_DEPL_R'	'FORC_AMOR'	'NOEU_DEPL_R'
'DEPL_ABSOLU'	'NOEU_DEPL_R'	'FORC_EXTE'	'NOEU_DEPL_R'
'DEPL_VIBR'	'NOEU_DEPL_R'	'FORC_LIAI'	'NOEU_DEPL_R'
'DERA_ELGA'	'ELGA_DERA_R'	'FORC_NODA'	'NOEU_DEPL_R'
'DERA_ELNO'	'ELNO_DERA_R'	'GEOMETRY'	'NOEU_GEOM_R'
'DERA_NOEU'	'NOEU_DERA_R'	'HYDR_ELNO'	'ELNO_HYDR_R'
'DISS_ELEM'	'ELEM_DISS_R'	'HYDR_NOEU'	'NOEU_HYDR_R'
'DISS_ELGA'	'ELGA_DISS_R'	'INDC_ELEM'	'ELEM_NEUT_I'
'DISS_ELNO'	'ELNO_DISS_R'	'INDL_ELGA'	'ELGA_INDL_R'
'DISS_NOEU'	'NOEU_DISS_R'	'INTE_ELNO'	'ELNO_INTE_R'
'DIVU'	'NOEU_EPSI_R'	'INTE_NOEU'	'NOEU_INTE_R'
'DURT_ELNO'	'ELNO_DURT_R'	'IRRA'	'NOEU_IRRA_R'
'DURT_NOEU'	'NOEU_DURT_R	'META_ELNO'	'ELNO_VARI_R'
'ECIN_ELEM'	'ELEM_ENER_R'	'META_NOEU'	'NOEU_VARI_R'
'EFGE_ELGA'	'ELGA_SIEF_R'	'MODE_FLAMB'	'NOEU_DEPL_R'
'EFGE_ELNO'	'ELNO_SIEF_R'	'MODE_STAB'	'NOEU_DEPL_R'
'EFGE_NOEU'	'NOEU_SIEF_R'	'NEUT'	'NOEU_NEUT_R'
'ENDO_ELGA'	'ELGA_SIEF_R'	'PDIL_ELGA'	'ELGA_PDIL_R'
'ENDO_ELNO'	'NOEU_SIEF_R'	'PRAC_ELNO'	'ELNO_PRAC_R'

'ENDO_NOEU'	'ELNO_SIEF_R'	'PRAC_NOEU'	'NOEU_PRAC_R'
'ENEL_ELEM'	'ELEM_ENER_R'	'CLOSE'	'NOEU_PRES_C'
'ENEL_ELGA'	'ELGA_ENER_R'	'PRME_ELNO'	'ELNO_PRME_R'
'ENEL_ELNO'	'ELNO_ENER_R'	'PTOT'	'NOEU_DEPL_R'
'ENEL_NOEU'	'NOEU_ENER_R'	'QIRE_ELEM'	'ELEM_ERRE_R'
'EPEQ_ELGA'	'ELGA_EPSI_R'	'QIRE_ELNO'	'ELNO_ERRE_R'
'EPEQ_ELNO'	'ELNO_EPSI_R'	'QIRE_NOEU'	'NOEU_ERRE_R'
'EPEQ_NOEU'	'NOEU_EPSI_R'	'QIZ1_ELEM'	'ELEM_ERRE_R'
'EPFD_ELGA'	'ELGA_EPSI_R'	'QIZ2_ELEM'	'ELEM_ERRE_R'
'EPFD_ELNO'	'ELNO_EPSI_R'	'REAC_NODA'	'NOEU_DEPL_R'
'EPFD_NOEU'	'NOEU_EPSI_R'	'SECO_ELEM'	'ELEM_NEUT_R'
'EPFP_ELGA'	'ELGA_EPSI_R'	'SIEF_ELGA'	'ELGA_SIEF_R'
'EPFP_ELNO'	'ELNO_EPSI_R'	'SIEF_ELNO'	'ELNO_SIEF_R'
'EPFP_NOEU'	'NOEU_EPSI_R'	'SIEF_NOEU'	'NOEU_SIEF_R'
'EPME_ELGA'	'ELGA_EPSI_R'	'SIEQ_ELGA'	'ELGA_SIEF_R'
'EPME_ELNO'	'ELNO_EPSI_R'	'SIEQ_ELNO'	'ELNO_SIEF_R'
'EPMG_ELGA'	'ELGA_EPSI_R'	'SIEQ_NOEU'	'NOEU_SIEF_R'
'EPMG_ELNO'	'ELNO_EPSI_R'	'SIGM_ELGA'	'ELGA_SIEF_R'
'EPMG_NOEU'	'NOEU_EPSI_R'	'SIGM_ELNO'	'ELNO_SIEF_R'
'EPMQ_ELGA'	'ELGA_EPSI_R'	'SIGM_NOEU'	'NOEU_SIEF_R'
'EPMQ_ELNO'	'ELNO_EPSI_R'	'SING_ELEM'	'ELEM_SING_R'
'EPMQ_NOEU'	'NOEU_EPSI_R'	'SING_ELNO'	'ELNO_SING_R'
'EPOT_ELEM'	'ELEM_ENER_R'	'SIPM_ELNO'	'ELNO_SIEF_R'
'EPSA_ELNO'	'ELNO_EPSI_R'	'SIPO_ELNO'	'ELNO_SIEF_R'
'EPSA_NOEU'	'NOEU_EPSI_R'	'SIPO_NOEU'	'NOEU_SIEF_R'
'EPSG_ELGA'	'ELGA_EPSI_R'	'SIRO_ELEM'	'ELEM_SIEF_R'
'EPSG_ELNO'	'ELNO_EPSI_R'	'SISE_ELNO'	'ELNO_SIEF_R'
'EPSG_NOEU'	'NOEU_EPSI_R'	'SIZ1_NOEU'	'NOEU_SIEF_R'
'EPSI_ELGA'	'ELGA_EPSI_R'	'SIZ2_NOEU'	'NOEU_SIEF_R'
'EPSI_ELNO'	'ELNO_EPSI_R'	'SOUR_ELGA'	'ELGA_SOUR_R'
'EPSI_NOEU'	'NOEU_EPSI_R'	'STRX_ELGA'	'ELGA_STRX_R'
'EPSP_ELGA'	'ELGA_EPSI_R'	'TEMP'	'NOEU_TEMP_R'
'EPSP_ELNO'	'ELNO_EPSI_R'	'THETA'	'NOEU_DEPL_R'
'EPSP_NOEU'	'NOEU_EPSI_R'	'UTXX_ELGA'	See remark below
'EPVC_ELGA'	'ELGA_EPSI_R'	'UTXX_ELNO'	See remark below
'EPVC_ELNO'	'ELNO_EPSI_R'	'UTXX_NOEU'	See remark below
'EPVC_NOEU'	'NOEU_EPSI_R'	'VAEX_ELGA'	'ELGA_NEUT_R'
'ERME_ELEM'	'ELEM_ERRE_R'	'VAEX_ELNO'	'ELNO_NEUT_R'
'ERME_ELNO'	'ELNO_ERRE_R'	'VAEX_NOEU'	'NOEU_NEUT_R'
'ERME_NOEU'	'NOEU_ERRE_R'	'CONT_NOEU'	'NOEU_INFC_R'
'ERTH_ELEM'	'ELEM_ERRE_R'	'VARC_ELGA'	'ELGA_VARC_R'
'ERTH_ELNO'	'ELNO_ERRE_R'	'VARI_ELGA'	'ELGA_VARI_R'
'ERTH_NOEU'	'NOEU_ERRE_R'	'VARI_ELNO'	'ELNO_VARI_R'
'ERZ1_ELEM'	'ELEM_ERRE_R'	'VARI_NOEU'	'NOEU_VAR2_R'
'ERZ2_ELEM'	'ELEM_ERRE_R'	'QUICKLY'	'NOEU_DEPL_R'
'ETHE_ELEM'	'ELEM_ENER_R'	'VITE_ABSOLU'	'NOEU_DEPL_R'

Note:

- Fields named "users" UT01_ELGA,..., UT19_ELNO can be associated with different sizes according to the context. To know the name of the size, one can look at the table written in the file .mess at the end of the execution associated with the structure of data result.

- For DYNA_HARMO, NOM_CHAM can take three values: 'DEPL', 'QUICKLY' and 'ACCE'. In the three cases, the type of the field result is one CHAM_NO/DEPL_C and it is thus necessary to inform: TYPE_CHAM='NOEU_DEPL_C'.

3.9.2 Operand TABLE

♦ TABLE = tabl

Name of concept table containing the values to be stored in the field. The names of the parameters of the table must comply with certain rules. The name of the meshs and the nodes must be coded on 8 characters, tables printed with the format 'TABLE' the chains print on 24 characters by default, it is thus preferable to use the format 'ASTER' during the second reading of the tables to specify the format character strings and to indicate K8 for the parameters NODE and MESH.

The columns containing the values of the field (realities) must be identified by their name of component in the size. For example: DX, DY, DZ for displacement (DEPL_R).

The other columns to be informed depend on the type of field to create:

Type of the field	Column 1	Column 2	Column 3
NOEUD_XXXX	NODE		
ELEM_XXXX	MESH	[SOUS-POINT]	
ELNO_XXXX	MESH	NODE	[SOUS-POINT]
ELGA_XXXX	MESH	NOT	[SOUS-POINT]

The parameter NODE the name of the node contains.
The parameter MESH the name of the mesh contains.
The parameter NOT the number of the point of Gauss in the mesh contains.
The parameter SOUS_POINT (necessary only for the fields to "under-points") the number of the under-point in the point of Gauss contains (or the node).

Caution: the table should not contain of another column only the expected columns (for example: NUME_ORDRE, INST,...). If the table contains useless columns, they should be removed using the order CALC_TABLE + OPERATION=' EXTR'.

3.9.3 Operand RESULT

♦ RESULT = resu

Name of concept result in which one wants to recover a field.

3.9.4 Operand NOM_CHAM

♦ NOM_CHAM

This keyword specifies the reference symbol of the field to be extracted [U4.71.00].

3.9.5 Operands NUME_ORDRE/NUME_MODE/NOM_MODE/.../INTERPOL

These keywords are used to specify which is sequence number of SD_RESULTAT that one wants to extract.

The choice of the keywords to be used depends on the type on SD_RESULTAT [U4.71.00].

Remarks

When one uses INTERPOL = 'FLAX' , the extracted field will be an interpolation between two fields of SD_RESULTAT . This interpolation always does not have a "physical" direction; for example on clean modes. This keyword should be used only for SD_RESULTAT of type " evol_xxxx "
When one uses an access of the "real" type (INST or FREQ), one seeks a field in a given interval. If one finds several fields in the interval, the program stops in fatal error.

3.9.6 Calculation of a field containing the "extrema" of one SD_RESULTAT

The idea is to create a field containing in each point of space the extreme value recorded during a transient (or the moment to which this value was recorded).

Today this paragraph relates to only the results of the type evol_ther, evol_elas, evol_noli and dyna_trans. The fields are always of "real" type.

One must specify:
- sequence numbers defining the transient: TOUT_ORDRE, LIST_INST,
- the selected type of extremality: TYPE_MAXI = 'MAXIMUM', ..., 'MINI_ABS', 'NORM_TRAN',
- what one wants: the extreme value or the moment when this value is reached (TYPE_RESU).

3.9.6.1 Operand TYPE_MAXI

/ 'MAXIMUM' one considers the maximum reached by the components during the transient,
/ 'MAXI_ABS' one considers the maximum reached by the absolute value of the components during the transient,
/ 'MINI' idem for the minima,
/ 'MINI_ABS'
/ 'NORM_TRAN' one considers the maximum reached by the quantity: DX ** 2 + DY ** 2 + DZ ** 2 .

- for the 4 values: 'MAXIMUM', ..., 'MINI_ABS', the components of the field are treated independently from/to each other: the extreme values can not be reached at the same moment,
- the fifth possible value: 'NORM_TRAN' is not possible that for the fields of depl_R. In a given point, one seeks the moment when the standard of the vector translation is maximum and one recopies in the field result all the components of the field at the found moment.

3.9.6.2 Operand TYPE_RESU

/ 'VALE', the field result contains the extreme values recorded during the transient,

Note:

Even if the extremum were obtained with an absolute value ('MAXI_ABS' or 'MINI_ABS'), the stored value is algebraic. It is necessary to use VALE_ABS to have the absolute value.

/ 'INST', the field result contains the values of the moments when the extreme values were recorded.

Note:

SI for example:
- *NOM_CHAM = 'FLUX_ELNO',*
- *TYPE_RESU = 'INST',*

The field result is a field of the type FLUX_R which contains values of moments!

/ 'VALE_ABS', available only with one TYPE_MAXI absolute ('MAXI_ABS' or 'MINI_ABS'), the field result then contains the extreme absolute values recorded during the transient.

3.9.6.3 Operands TOUT_ORDRE / LIST_INST / LIST_FREQ / PRECISION / CRITERION

These keywords make it possible to specify the extent of the transient to be examined.

If TOUT_ORDRE = 'YES' one reviews all the sequence numbers.
If LIST_INST = linst only the specified moments are considered.

3.9.7 Examples

3.9.7.1 Extraction of a field of temperature of a concept result of type evol_ther

```
temp10 =  CREA_CHAMP       (  OPERATION=' EXTR',
          NOM_CHAM  = 'TEMP'  ,  TYPE_CHAM  = 'NOEU_TEMP_R',
          RESULT = evoth      ,  INST = 10. )
```

temp10 is the field of temperature extracted the result evoth (of type evol_ther) at the moment 10.

3.9.7.2 Extraction of a field of displacement of a concept result of type mode_meca

```
mode4 =   CREA_CHAMP    (  OPERATION=' EXTR'   ,
          NOM_CHAM  = 'DEPL'  ,  TYPE_CHAM  = 'NOEU_DEPL_R',
          RESULT = modes      ,  NUME_MODE = 4)
```

mode4 is the 4ème clean mode of the result modes (of type mode_meca).

3.9.7.3 Extraction of the field of "temperature" containing the moments when the maximum temperature was reached during a transient

```
instmax =     CREA_CHAMP ( OPERATION=' EXTR'   ,
              NOM_CHAM  = 'TEMP'  ,  TYPE_CHAM  = 'NOEU_TEMP_R',
              RESULT  = evoth  ,
              TYPE_MAXI = 'MAXIMUM'   ,  TYPE_RESU = 'INST'     )
```

3.9.7.4 Extraction of a stress field in a table

That is to say the file (fort.81) containing the image of the following table:

```
#DEBUT_TABLE
FORCED #TITRE 'ELNO'
MESH  NOT   SIXX       SIYY      SIZZ
     K8      I       R          R         R
     M1      1    -1.632E+03  -2.553E+02  6.788E-01
     M1      2    -5.302E+03  -9.663E+01  6.018E+01
     M1      3    -3.638E+03  -1.058E+02  5.669E+01
     M2      1     5.632E+01   1.553E+02  3.788E-01
#FIN_TABLE
```

One can extract a field from "constraints" to the nodes of this table while making:

```
# reading of the table:
TA=LIRE_TABLE (UNITE=81, TYPE_TABLE=' TABLE', SEPARATEUR=' ')

# extraction of the field in the table:
CH=CRÉA_CHAMP (OPERATION=' EXTR', TYPE_CHAM=' ELNO_SIEF_R', TABLE=' TA',
               MODELE=MO, PROL_ZERO=' OUI', OPTION=' SIEF_ELNO_DEPL')
```

3.9.7.5 Calculation of the field of the "normals" on a group of meshs of edge

```
nor_DNOR = CREA_CHAMP (TYPE_CHAM = 'NOEU_GEOM_R',
                       'NORMAL' OPERATION=, MODELE= MO, GROUP_MA= 'FE'   );
```