

## Operator DEFI\_GROUP

---

### 1 Goal

---

To define in an existing grid, new groups of nodes or meshes. This can facilitate the definition of new loci for inputs or postprocessings.

To create new groups, one uses topological, logical or geometrical criteria.

Modify a structure of data of the type `grid`, `skeleton` or `grid`.

## Contents

1 Goal.....	1
2 Syntax.....	4
3 Operands.....	8
3.1 General information on the operands.....	8
3.2 Operands GRID and GRID.....	8
3.3 Keywords DETR_GROUP_MA and DETR_GROUP_NO.....	9
3.4 Keyword CREA_GROUP_MA.....	10
3.4.1 Operand NAME.....	10
3.4.2 Operand TYPE_MAILLE.....	10
3.4.3 Operand ALL.....	11
3.4.4 Operand INTERSEC.....	11
3.4.5 Operand UNION.....	11
3.4.6 Operand DIFFE.....	11
3.4.7 Sub-group of an existing group: keywords GROUP_MA / POSITION / NUME_INIT / NUME_FIN.....	12
3.4.8 Operand OPTION = 'FACE_NORMALE'.....	13
3.4.8.1 Operand ANGL_NAUT.....	14
3.4.8.2 Operand VECT_NORMALE.....	14
3.4.8.3 Operand ANGL_PREC.....	14
3.4.8.4 Operand VERI_SIGNE.....	14
3.4.9 Operand OPTION = 'SPHERE'.....	14
3.4.9.1 Operand NOT.....	14
3.4.9.2 Operand GROUP_NO_CENTRE.....	14
3.4.9.3 Operand RAY.....	14
3.4.10 Operand OPTION = 'CYLINDER'.....	15
3.4.10.1 Operand NOT.....	15
3.4.10.2 Operand GROUP_NO_CENTRE.....	15
3.4.10.3 Operand RAY.....	15
3.4.10.1 Operand ANGL_NAUT.....	15
3.4.10.2 Operand VECT_NORMALE.....	15
3.4.11 Operand OPTION = 'BAND'.....	15
3.4.11.1 Operand NOT.....	15
3.4.11.2 Operand GROUP_NO_CENTRE.....	16
3.4.11.3 Operand ANGL_NAUT.....	16
3.4.11.4 Operand VECT_NORMALE.....	16
3.4.11.5 Operand DIST.....	16
3.4.12 Operand OPTION = 'SUPPORT'.....	16
3.4.13 Operand OPTION = 'FISS_XFEM'.....	16

3.5 Keyword CREA_GROUP_NO.....	17
3.5.1 Operand NAME.....	17
3.5.2 Operand INTERSEC.....	17
3.5.3 Operand UNION.....	17
3.5.4 Operand DIFFE.....	17
3.5.5 Under group of an existing group: keywords GROUP_NO / POSITION / NUME_INIT / NUME_FIN.....	17
3.5.6 Operand OPTION = 'ENV_SPHERE'.....	18
3.5.6.1 Operand NOT.....	18
3.5.6.2 Operand GROUP_NO_CENTRE.....	18
3.5.6.3 Operand RAY.....	18
3.5.6.4 Operand PRECISION.....	18
3.5.7 Operand OPTION = 'ENV_CYLINDRE'.....	18
3.5.7.1 Operand NOT.....	18
3.5.7.2 Operand GROUP_NO_CENTRE.....	18
3.5.7.3 Operand RAY.....	18
3.5.7.4 Operand ANGL_NAUT.....	19
3.5.7.5 Operand VECT_NORMALE.....	19
3.5.7.6 Operand PRECISION.....	19
3.5.8 Operand OPTION = 'PLAN'.....	19
3.5.8.1 Operand NOT.....	19
3.5.8.2 Operand GROUP_NO_CENTRE.....	19
3.5.8.3 Operand ANGL_NAUT.....	19
3.5.8.4 Operand VECT_NORMALE.....	19
3.5.8.5 Operand PRECISION.....	20
3.5.9 Operand OPTION = 'SEGM_DROI_ORDO'.....	20
3.5.10 Operand OPTION = 'NOEUD_ORDO'.....	20
3.5.10.1 Case of the closed lines.....	21
3.5.11 Operand OPTION = 'TUNNEL'.....	21
3.5.12 Operand OPTION = 'INCLUSION'.....	21
3.5.13 Operand OPTION = 'INTERVALLE_VALE'.....	22
3.5.14 Operand OPTION = 'FISS_XFEM'.....	22
3.5.15 Operand RAYON_TORE.....	22
3.5.16 Operand OPTION = 'RELA_CINE_BP'.....	23
3.5.17 Operands GROUP_MA and NAME.....	23
3.5.18 Operand TOUT_GROUP_MA.....	23
3.5.19 Operand ALARM = 'YES' [DEFECT] / 'NOT'.....	23
3.5.20 Operand INFORMATION.....	23
4 Examples.....	24

## 2 Syntax

```
my (grid) =      DEFI_GROUP    (

  ◊ reuse =/my,
                / gr.,
  ◆ | GRID = my ,                /                [grid]
                /                [skeleton]
  | GRID = gr.,                /                [grid]

  ◆ | DETR_GROUP_MA = _F (
      ◆ NAME = lgma                ),                [l_group_ma]
  | DETR_GROUP_NO = _F (
      ◆ NAME = lgno                ),                [l_group_no]

  | CREA_GROUP_MA = (_F (
      ◆ NAME = gma                ,                [identifiant]
      ◊ TYPE_MESH = / 'ALL'                [DEFECT]
                / '3D'/'2D'/'1D'
                / 'SEG2'/'TRIA3'/'QUAD4'
                / 'QUAD8'/'... / 'PYRAM13'

      ◆ / ALL = 'YES' ,
        / INTERSEC = lgma ,                [l_group_ma]
        / UNION = lgma ,                [l_group_ma]
        / DIFFE = lgma ,                [l_group_ma]
        / GROUP_MA = gma ,                [group_ma]
          / NUME_INIT = / nuini ,                [I]
          / 1 ,                [DEFECT]
          NUME_FIN = nufin ,                [I]

          / POSITION = / 'INIT' ,
                    / 'FINE' ,
                    / 'MEDIUM' ,

      / OPTION = 'FACE_NORMALE' ,
      ◆ / ANGL_NAUT = (has, b) ,                [l_R]
        / VECT_NORMALE= (X, there, Z) ,                [l_R]
      ◊ ANGL_PREC = / has ,                [R]
                / 0.5,                [DEFECT]
      ◊ VERI_SIGNE = / 'NOT',                [DEFECT]
                / 'YES',

      / OPTION = 'SPHERE',
      ◆ / NOT = (X, there, Z),                [l_R]
        / GROUP_NO_CENTRE = grno,                [group_no]
      ◆ RAY = R,                [R]

      / OPTION = 'CYLINDER',
      ◆ / NOT = (X, there, Z),                [l_R]
        / GROUP_NO_CENTRE = grno,                [group_no]
      ◆ RAY = R,                [R]
      ◆ / ANGL_NAUT = (has, b),                [l_R]
        / VECT_NORMALE= (X, there, Z),                [l_R]
```

```
/ OPTION = 'BAND',
♦ / NOT = (X, there, Z), [l_R]
/ GROUP_NO_CENTRE = grno, [group_no]
♦ / ANGL_NAUT = (has, b), [l_R]
/ VECT_NORMALE= (X, there, Z), [l_R]
♦ DIST = D, [R]

/ OPTION = 'SUPPORT',
♦ GROUP_NO = lgno, [l_group_no]
♦ TYPE_APPUI = / 'AU_MOINS_UN'
/ 'ALL'
/ 'TOP'
/ 'MAJORITY'

/ OPTION = 'FISS_XFEM',
♦ CRACK = (fiss1, fiss2,...), [l_fiss_xfem]
♦ TYPE_GROUP = / 'XFEM'
/ 'HEAVISIDE'
/ 'CRACKTIP'
/ 'MIXED'
/ 'FISSUREE'

),),

| CREA_GROUP_NO = (_F (
/ ♦ NAME = gno, [identifrier]
♦ / INTERSEC = lgno, [l_group_no]
/ UNION = lgno, [l_group_no]
/ DIFFE = lgno, [l_group_no]
/ GROUP_NO = gno, [group_no]
/ NUME_INIT = / nuini, [I]
/ 1, [DEFECT]
NUME_FIN = nufin, [I]
/ POSITION = / 'INIT',
/ 'FINE',
/ 'MEDIUM',

/ OPTION = 'ENV_SPHERE',
♦ / NOT = (X, there, Z), [l_R]
/ GROUP_NO_CENTRE = grno, [group_no]
♦ RAY = R, [R]
♦ PRECISION = eps, [R]

/ OPTION = 'ENV_CYLINDRE',
♦ / NOT = (X, there, Z), [l_R]
/ GROUP_NO_CENTRE = grno, [group_no]
♦ RAY = R, [R]
♦ / ANGL_NAUT = (has, b), [l_R]
/ VECT_NORMALE= (X, there, Z), [l_R]
♦ PRECISION = eps, [R]

/ OPTION = 'PLAN',
♦ / NOT = (X, there, Z), [l_R]
/ GROUP_NO_CENTRE = grno, [group_no]
♦ / ANGL_NAUT = (has, b), [l_R]
/ VECT_NORMALE= (X, there, Z), [l_R]
♦ PRECISION = eps, [R]

/ OPTION = 'SEGM_DROI_ORDO',
♦ GROUP_NO = gn2, [group_no]
```

```
♦ GROUP_NO_ORIG= gnoA , [group_no]
♦ GROUP_NO_EXTR= gnoB , [group_no]
♦ PRECISION = prec, [R]
♦ CRITERION =/ 'RELATIVE' ,
/ 'ABSOLUTE' ,

/ OPTION = 'NOEUD_ORDO',
♦ GROUP_MA = gmaAB , [group_ma]
◊ GROUP_NO_ORIG= gnoA , [group_no]
◊ GROUP_NO_EXTR= gnoB , [group_no]
◊ VECT_ORIE = (vx, vy, [vz]), [l_R]

/ OPTION = 'TUNNEL',
♦ / ALL = 'YES'
/ GROUP_MA = lgma , [l_group_ma]
♦ GROUP_MA_AXE = gnoA , [l_group_ma]
♦ GROUP_NO_ORIG= gnoA , [group_no]
♦ RAY = R , [R]
◊ LENGTH = long , [R]

/ OPTION = 'INCLUSION',
♦ GROUP_MA = lgma , [l_group_ma]
♦ CAS_FIGURE = / '2D'
/ '3D'
/ '2.5D'
◊ DISTANCE_MAX = distma [R]
♦ GROUP_MA_INCL = lgma_inc , [l_group_ma]
◊ MAILLAGE_INCL = ma_inc , [grid]

/ OPTION = 'INTERVALLE_VALE',
♦ CHAM_GD = chno , [cham_no]
♦ NOM_CMP = cmp, [TXM]
♦ VALE = (vmin, vmax) [R]

/ OPTION = 'FISS_XFEM',
♦ CRACK = (fiss1, fiss2,...), [l_fiss_xfem]
♦ TYPE_GROUP = / 'XFEM'
/ 'HEAVISIDE'
/ 'CRACKTIP'
/ 'MIXED'
/ 'ZONE_MAJ'
/ 'TORUS'
# If TYPE_GROUP = 'TORUS':
♦ RAYON_TORE = R, [R]

/ OPTION = 'RELA_CINE_BP',
♦ CABLE_BP = cable_bp, [cable_precont]
♦ PREF_GRNO = pref, [TXM]
= 'RCBP' [DEFECT]

/ GROUP_MA = lgma, [l_identificator]
◊ NAME = lgn, [l_group_no]
◊ CRIT_NOEUD = / 'ALL' , [DEFECT]
/ 'TOP',
/ 'MEDIUM',
/ 'CENTER',

/ TOUT_GROUP_MA: 'YES',
```

```
),),  
  
◇ ALARM = / 'YES', [DEFECT]  
          / 'NOT',  
  
◇ INFORMATION = / 1, [DEFECT]  
                / 2,  
                )
```

Type of the result:

```
if GRID : grid then : grid  
          : skeleton skeleton  
if GRID : grid then : grid
```

## 3 Operands

### 3.1 General information on the operands

This order treats the concepts of the type in the same way `grid` or `skeleton`. In the continuation one will use the vocabulary "grid".

This order makes it possible to define new groups of meshes (or groups of nodes) in an existing grid: the grid is enriched `my`.

The definition of a new group can be done in several ways:

- by Boolean operation on existing groups: intersection (`INTERSEC`), meeting (`UNION`) or difference (`DIFFE`),
- according to a geometrical criterion: meshes whose node belongs to a given sphere,...
- for the groups of nodes, by referring to existing groups of meshes. The group of nodes thus defined contains **all** nodes of the meshes of the group of meshes origin (keywords `TOUT_GROUP_MA` and `GROUP_MA`).

The operator treats initially the keyword `CREA_GROUP_MA` so that one can make use of the groups of meshes thus defined in the keyword `CREA_GROUP_NO`.

With each occurrence of a keyword `CREA_GROUP_MA` (`_NO`) one defines a new group named (keyword `NAME`). This new group can then be re-used in the following occurrences to define new groups by intersection, meeting,...

Keywords `DETR_GROUP_MA` and `DETR_GROUP_NO` allow "to destroy" groups of meshes or nodes. The meshes and the nodes of these groups are not removed, they are only the definitions of the groups which are unobtrusive. These keywords are useful for example in the loops python when one wants to create a group with each iteration of the loop: one starts by destroying this group then one recreates it under the same name. That avoids changing name of group to each iteration.

One can note that most features of `DEFI_GROUP` are feasible directly in module `SMESH` of `SALOMÉ`, namely:

- Boolean operation on existing groups,
- creation of group per type of mesh (filter `SMESH Type`),
- according to a geometrical criterion (filter `SMESH LyingOn`),
- according to a criterion of angle of normal for surface meshes (filter `SMESH Coplanar`),
- for the groups of meshes being based on groups of nodes (function `SMESH 'Group based one nodes of other groups'`),
- for the groups of nodes, by referring to existing groups of meshes (function `SMESH 'Group based one nodes of other groups'`),
- suppression of groups.

### 3.2 Operands GRID and GRID

◆ | `GRID = my`

`my` is the name of the grid which one wants "to enrich".

| `GRID = gr.`

`gr.` is the name of the auxiliary grid which one wants "to enrich".

## 3.3 Keywords DETR\_GROUP\_MA and DETR\_GROUP\_NO

These two keywords factor make it possible to remove the definition of groups of meshes or nodes. These keywords are sometimes necessary because the code stops in fatal error if one tries to create a group whose name is already used. It is necessary to destroy the group before being able to re-use its name. The behavior of the two keywords is similar and we will speak here only about DETR\_GROUP\_MA.

Syntax:

```
DETR_GROUP_MA=_F (NOM= (gm1, gm2,...)),
```

The keyword factor DETR\_GROUP\_MA is a priori répétable but it is never necessary because the keyword NAME allows to indicate a list of names of groups to be destroyed (gm1, gm2,...).

It is important to know that all the occurrences of the keyword `DETR_GROUP_MA` are treated **front** those of the keyword `CREA_GROUP_NO` because the objective of this keyword is to be able to re-use the destroyed name. It also should be known that the destruction of a non-existent group does not involve any message of alarm. These choices make it possible for example to make in a loop python:

```
for I in arranges (N):
    DEFI_GROUPE (reuse=MA, MAILLAGE=MA,
                DETR_GROUP_MA=_F (NAME ('GM1',)),
                CREA_GROUP_MA=_F (NOM=' GM1',...
```

At the time of the first iteration, the group `'GM1'` do not exist, one asks his destruction but no message of alarm is transmitted.

## Note:

*As the destruction takes place at the beginning of the order, it is impossible to modify a group by making only one call with `DEFI_GROUP`. For example, one cannot make "enlarge" (in a loop) a group by adding a small group to him (`b1`):*

```
for I in arranges (N):
    b1=nouveau group..
    DEFI_GROUP (reuse=MA, MAILLAGE=MA,
                CREA_GROUP_MA=_F (NOM=' tout', UNION= ('all', 'b1'))))
```

*To do that, it is necessary to invite twice `DEFI_GROUP` :*

```
for I in arranges (N):
    b1=nouveau group..
    DEFI_GROUP (reuse=MA, MAILLAGE=MA,
                DETR_GROUP_MA=_F (NOM=' tout2'),
                CREA_GROUP_MA=_F (NOM=' tout2', UNION= ('all', 'b1'))))
    DEFI_GROUP (reuse=MA, MAILLAGE=MA,
                DETR_GROUP_MA=_F (NOM=' tout'),
                CREA_GROUP_MA=_F (NOM=' tout', UNION= ('tout2', 'b1'))))
```

## 3.4 Keyword `CREA_GROUP_MA`

| `CREA_GROUP_MA`

An occurrence of this keyword factor makes it possible to define a new group of meshes.

### 3.4.1 Operand `NAME`

◆ `NAME = gma`

One gives here the name (with "quotes") of the new group of meshes.

### 3.4.2 Operand `TYPE_MAILLE`

◇ `TYPE_MAILLE= /'ALL'/'3D'/'2D'/'1D' (DEFAULT=' TOUT')`  
`/'SEG2'/'TRIA3'/'QUAD4' /.../'PYRAM13'`

This keyword makes it possible to filter the meshes which one will put in the new group of meshes. By default, it does not have there a filter, but if the user writes for example: `TYPE_MAILLE=' 2D'`, the group created will contain only surface meshes.

The user can also filter the group to be created for a kind of mesh individual (`TRIA3`, `HEXA27`, ...). All types of meshes (`POI1`, `SEG2`, `SEG3`, `SEG4`, ..., `PYRAM13`) are authorized.

Examples:

*Warning : The translation process used on this website is a "Machine Translation". It may be imprecise and inaccurate in whole or in part and is provided as a convenience.*

Copyright 2019 EDF R&D - Licensed under the terms of the GNU FDL (<http://www.gnu.org/copyleft/fdl.html>)

CREA\_GROUP\_MA=\_F (NOM=' VOLUM', 'TOUT=' YES', TYPE\_MAILLE=' 3D')  
allows to create the group of all the voluminal meshes (HEXA, PENTA,...) grid.

CREA\_GROUP\_MA=\_F (NOM=' VOLH27', 'GROUP\_MA=' GMA1', TYPE\_MAILLE='  
HEXA27')  
allows to create the group of all the meshes of the type 'HEXA27' contained in GROUP\_MA.  
GMA1.

### 3.4.3 Operand ALL

/ ALL = 'YES'

This keyword makes it possible to define a group containing all the meshes of the grid.

### 3.4.4 Operand INTERSEC

/ INTERSEC = (gma1, gma2, gma3,...),

The new group of meshes will be obtained by taking all the meshes of gma1 who also belong to gma2, gma3,... The order of the meshes remains that of gma1.

### 3.4.5 Operand UNION

/ UNION = (gma1, gma2, gma3,...)

The new group of meshes will be obtained by taking all the meshes of gma1, then by adding the meshes of gma2 who do not belong to gma1, then those of gma3 who do not belong nor to gma1 nor with gma2, etc.

### 3.4.6 Operand DIFFE

/ DIFFE = (gma1, gma2, gma3,...)

The new group of meshes will be obtained by taking all the meshes of gma1 who do not belong to the other groups of the list. The order of the meshes remains that of gma1.

## 3.4.7 Sub-group of an existing group: keywords `GROUP_MA` / `POSITION` / `NUME_INIT` / `NUME_FIN`

One can create a new group of mesh by selecting certain meshes of an existing group.

### 1<sup>ère</sup> possibility:

A group is created **from only one nets** while specifying by the keyword `POSITION` the required mesh.

### Example:

```
CREA_GROUP_MA = _F ( GROUP_MA = G1, POSITION = 'INIT', NAME = G1I)
```

The group `G1I` the 1 contains<sup>ère</sup> mesh of the group `G1`.

### 2<sup>ème</sup> possibility:

One creates a group containing the meshes ranging between the rows `nuini` and `nufin` (included) in an existing group.

### Example:

```
CREA_GROUP_MA=_F (GROUP_MA = G1, NUME_INIT = 3, NUME_FIN = 7, NAME = G1P)
```

The group `G1P` contains meshes 3,4,5,..., 7 of `G1`.

### Caution:

*These keywords use the concept of **order** meshes in a group of meshes. This order is often unknown to the user. It can depend on the preprocessor. It is the order of the meshes at the time of the definition of `GROUP_MA` in the file of grid Aster.*

## 3.4.8 Operand `OPTION = 'FACE_NORMALE'`

/ `OPTION = 'FACE_NORMALE'`

This option makes it possible to define one `GROUP_MA` constituted by surface meshes whose normal is parallel to the direction of the vector defined by its components if the keyword is used `VECT_NORMALE` or with that of the first vector of the new base defined by the change of reference mark due to the nautical angles.

In 3D, it is supposed that the surface meshes are plane facets. They are of type `TRIA3`, `TRIA6`, `QUAD4`, `QUAD8` or `QUAD9`. If one calls  $X_1$ ,  $X_2$ , and  $X_3$  the vectors position of the first three nodes tops of the element, the normal is determined by the vector product:  $(X_2 - X_1) \wedge (X_3 - X_1)$ .

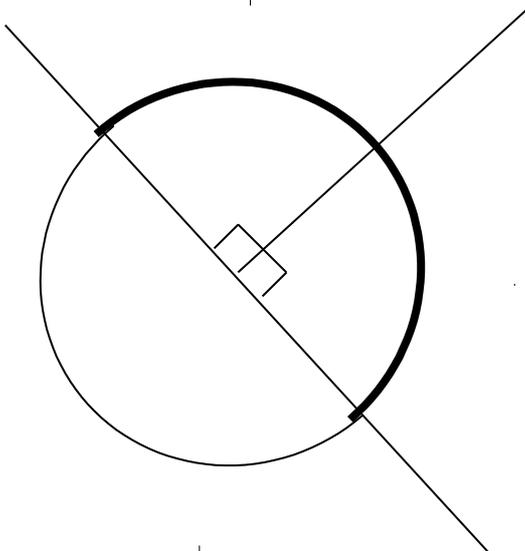
In 2D, it is supposed that the surface meshes are right segments. They are of type `SEG2` or `SEG3`. If one calls  $X_1$  and  $X_2$  the vectors position of the two nodes ends of the element, the normal is defined by  $(X_2 - X_1) \wedge z$  where  $z$  is the unit vector perpendicular to the plan and where one has affected 0. like third component with  $(X_2 - X_1)$ .

### Note:

A mesh "facet" will be retained if its normal is colinéaire with the normal vector defined by `VECT_NORMALE`. This condition must be checked except for a certain precision (keyword `ANGL_PREC`).

When one is chosen `ANGL_PREC` (for example 30. degrees), one defines in fact the group of the meshes whose normal belongs to the cone of axis `VECT_NORMALE` and of point angle `ANGL_PREC`.

This can be used (for example) to gather the meshes of a half wraps spherical (`ANGL_PREC = 90.`).



## 3.4.8.1 Operand ANGL\_NAUT

◆ / ANGL\_NAUT = has in 2D  
(has, B) in 3D

Nautical angles (has, b) defined in degrees, are the angles allowing to pass from the total reference mark of definition of the coordinates of the nodes to a reference mark whose first vector indicates the direction according to which the normal of the surface meshes is directed that one wishes to recover.

For the definition of the nautical angles, to see the operator AFFE\_CARA\_ELEM [U4.42.01] operand ORIENTATION.

## 3.4.8.2 Operand VECT\_NORMALE

◆ / VECT\_NORMALE = (X, there) in 2D  
(X, there, Z) in 3D

Coordinates X, there, Z are those giving the direction according to which the normal of the surface meshes is directed that one wishes to recover.

## 3.4.8.3 Operand ANGL\_PREC

◇ ANGL\_PREC = has

It is the tolerance, in degrees, that one accepts on the angle formed by the vector provided by the user and the normal vector to the surface element to affirm that these two vectors have the same direction.

The value by default of has is 0.5 degree.

## 3.4.8.4 Operand VERI\_SIGNE

◇ VERI\_SIGNE = / 'NOT' ,  
/ 'YES' , [DEFECT]

If the value is affected 'NOT' with VERI\_SIGNE, it GROUP\_MA will be made up by the surface meshes whose normal is parallel to the vector given by the user.

If the value is affected 'YES', it GROUP\_MA will be made up by the surface meshes whose normal is parallel and has the same orientation as the vector given by the user.

The value by default is 'YES'.

## 3.4.9 Operand OPTION = 'SPHERE'

/ OPTION = 'SPHERE'

This option makes it possible to define one GROUP\_MA constituted by the meshes whose at least node belongs to a sphere (a circle in 2D) defined by its centre and its.

### 3.4.9.1 Operand NOT

◆ / NOT = (X, there) in 2D  
(X, there, Z) in 3D

X there Z are the coordinates of the center of the sphere.

### 3.4.9.2 Operand GROUP\_NO\_CENTRE

◆ GROUP\_NO\_CENTRE = grno

This keyword makes it possible to indicate which is the node coinciding with the center of the sphere.

### 3.4.9.3 Operand RAY

◆ RAY = R

R is the radius of the sphere (circle in 2D).

### 3.4.10 Operand **OPTION = 'CYLINDER'**

/ OPTION = 'CYLINDER'

This option makes it possible to define one `GROUP_MA` constituted by the meshes whose at least node belongs to a cylinder defined by its axis and its ray.

The axis is defined by a vector and a point pertaining to this axis. This option has direction only in 3D.

#### 3.4.10.1 Operand **NOT**

◆ / NOT = (X, there, Z)

X there Z are the punctual coordinates located on the axis of the cylinder.

#### 3.4.10.2 Operand **GROUP\_NO\_CENTRE**

◆ GROUP\_NO\_CENTRE = grno

This MoT key makes it possible to indicate a node located on the axis of the cylinder.

#### 3.4.10.3 Operand **RAY**

◆ RAY = R

R is the ray of the cylinder.

#### 3.4.10.1 Operand **ANGL\_NAUT**

◆ / ANGL\_NAUT = (has, B)

Nautical angles `has, B` defined in degrees, are the angles allowing to pass from the total reference mark of definition of the coordinates of the nodes to a reference mark whose first vector indicates the direction of the axis of the cylinder.

For the definition of the nautical angles to see the operator `AFFE_CARA_ELEM [U4.42.01]` operand `ORIENTATION`.

#### 3.4.10.2 Operand **VECT\_NORMALE**

◆ / VECT\_NORMALE = (X, there, Z)

X there Z are the coordinates of a vector directing the axis of the cylinder.

### 3.4.11 Operand **OPTION = 'BAND'**

/ OPTION = 'BAND'

This option makes it possible to define one `GROUP_MA` constituted by the meshes whose at least node belongs to a "band" defined by a plan "medium" (a line in 2D) and the half-width on both sides of this plan.

The plan is defined by a normal vector in this plan and a point belonging to him.

#### 3.4.11.1 Operand **NOT**

◆ / NOT = (X, there) in 2D  
(X, there, Z) in 3D

X there Z are the punctual coordinates pertaining to the plan "medium" of the band.

## 3.4.11.2 Operand `GROUP_NO_CENTRE`

- ◆ `GROUP_NO_CENTRE = grno`

This keyword makes it possible to define one belonging to the plan “medium” of the band.

## 3.4.11.3 Operand `ANGL_NAUT`

- ◆ / `ANGL_NAUT = has in 2D`  
`(has . B) in 3D`

Nautical angles `has, B` defined in degrees, are the angles allowing to pass from the total reference mark of definition of the coordinates of the nodes to a reference mark whose first vector is orthogonal with the plan “medium” of the band.

For the definition of the nautical angles, to see the operator `AFFE_CARA_ELEM [U4.42.01]` operand `ORIENTATION`.

## 3.4.11.4 Operand `VECT_NORMALE`

- ◆ / `VECT_NORMALE = (X, there) in 2D`  
`(X, there, Z) in 3D`

`X there` and `Z` are the components of a vector perpendicular to the plan “medium” of the band.

## 3.4.11.5 Operand `DIST`

- ◆ `DIST = D`

`D` is the half-width of the band.

## 3.4.12 Operand `OPTION = 'SUPPORT'`

This option makes it possible to recover the group of the meshes whose certain nodes belong to the whole of the nodes specified by the keyword `GROUP_NO`.

```
GROUP_NO = l_gno
```

This keyword makes it possible to define the list of the nodes which will be used as support with the meshes. Let us call `lno1` this list.

- ◆ `TYPE_APPUI =`  
/ `'ALL'` : the mesh will be retained if **all** its nodes belong to `lno1`  
/ `'TOP'` : the mesh will be retained if **all** its nodes “**top**” belong to `lno1`  
/ `'AU_MOINS_UN'` : the mesh will be retained **at least one** of its nodes belongs to `lno1`  
/ `'MAJORITY'` : the mesh will be retained if **more half** of its nodes belong to `lno1`

## 3.4.13 Operand `OPTION = 'FISS_XFEM'`

This option makes it possible to recover the group of the meshes of the type X-FEM specified by the keywords `TYPE_GROUP`.

- ◆ `CRACK = (fiss1, fiss2,...)`
- ◆ `TYPE_GROUP =`  
/ `'HEAVISIDE'` : the mesh will be retained if it is of Heaviside type  
/ `'CRACKTIP'` : the mesh will be retained if it is of type Ace-tip  
/ `'MIXED'` : the mesh will be retained if it is of Mixed type (Heaviside and Ace-tip)  
/ `'XFEM'` : the mesh will be retained if it is of Heaviside type, Ace-tip or Mixed  
/ `'FISSUREE'` : the mesh will be retained if **all** its nodes are nouveau riches

For a definition specifies concepts of Heaviside mesh and mesh Ace-tip, to see R7.02.12, §3.2.5.

## 3.5 Keyword CREA\_GROUP\_NO

| CREA\_GROUP\_NO

An occurrence of this keyword factor makes it possible to define a new group of nodes (for the keywords GROUP\_MA and TOUT\_GROUP\_MA, one creates several groups of nodes “at a stretch”).

### 3.5.1 Operand NAME

/ ♦ NAME = gno

One gives here the name (with “quotes”) of the new group of nodes.

### 3.5.2 Operand INTERSEC

/ INTERSEC = (gno1, gno2, gno3,...)

The new group of nodes will be obtained by taking all the nodes of gno1 who also belong to gno2, gno3,... The order of the nodes remains that of gno1.

### 3.5.3 Operand UNION

/ UNION = (gno1, gno2, gno3,...)

The new group of nodes will be obtained by taking all the nodes of gno1, then by adding the nodes of gno2 who do not belong to gno1, then those of gno3 who do not belong nor to gno1 nor with gno2, etc.

### 3.5.4 Operand DIFFE

/ DIFFE = (gno1, gno2, gno3,...)

The new group of nodes will be obtained by taking all the nodes of gno1 who do not belong to the other groups of the list. The order of the nodes remains that of gno1.

## 3.5.5 Under group of an existing group: keywords GROUP\_NO / POSITION / NUME\_INIT / NUME\_FIN

One can create a new group of node by selecting certain nodes of an existing group.

### 1<sup>ère</sup> possibility:

A group is created **of only one node** while specifying by the keyword POSITION the required node.

#### Example:

```
CREA_GROUP_NO = _F (GROUP_NO = G1, POSITION = 'INIT', NAME = G1I)
```

The group G1I the 1 contains<sup>er</sup> node of the group G1.

### 2<sup>ème</sup> possibility:

One creates a group containing the nodes ranging between the rows nuini and nufin (included) in an existing group.

#### Example:

```
CREA_GROUP_NO=_F (GROUP_NO = G1, NUME_INIT = 3 NUME_FIN = 7, NAME = G1P)
```

The group G1P node 3,4,5,..., 7 contains of G1.

## Caution:

*These keywords use the concept of **order** nodes in a group of nodes. This order is often unknown to the user. It can depend on the preprocessor. It is the order of the nodes at the time of the definition of `GROUP_NO` in the file of grid Aster.*

### 3.5.6 Operand `OPTION = 'ENV_SPHERE'`

```
/ OPTION = 'ENV_SPHERE'
```

This option makes it possible to define one `GROUP_NO` constituted by the nodes located on the envelope of a sphere except for a precision given.

#### 3.5.6.1 Operand `NOT`

◆ / `NOT = (X, there)` , in 2D  
`(X, there, Z)`, in 3D

X there Z are the coordinates of the center of the sphere.

#### 3.5.6.2 Operand `GROUP_NO_CENTRE`

◆ `GROUP_NO_CENTRE = grno`

This keyword makes it possible to define the node coinciding with the center of the sphere.

#### 3.5.6.3 Operand `RAY`

◆ `RAY = R`

R is the ray of the sphere.

#### 3.5.6.4 Operand `PRECISION`

◇ `PRECISION = eps`

`eps` is the tolerance with which one defines the membership of one node in the envelope of the sphere. This tolerance is to be taken with the following direction:

if  $d$  is the distance from a node in the center of the sphere, one says that this node belongs to the group if:

$$|d - r| \leq \text{eps}$$

### 3.5.7 Operand `OPTION = 'ENV_CYLINDRE'`

```
/ OPTION = 'ENV_CYLINDRE'
```

This option makes it possible to define one `GROUP_NO` constituted by nodes located on the envelope of a cylinder except for a precision given.

This option has direction only in 3D.

#### 3.5.7.1 Operand `NOT`

◆ / `NOT = (X, there, Z)`

X there Z are the punctual coordinates pertaining to the axis of the cylinder.

#### 3.5.7.2 Operand `GROUP_NO_CENTRE`

◆ `GROUP_NO_CENTRE = grno`

This keyword makes it possible to define a node belonging to the axis of the cylinder.

#### 3.5.7.3 Operand `RAY`

- ◆ RAY = R  
R is the ray of the cylinder.

### 3.5.7.4 Operand ANGL\_NAUT

- ◆ / ANGL\_NAUT = (has , B)  
Nautical angles has, B defined in degrees, are the angles allowing to pass from the total reference mark of definition of the coordinates of the nodes to a reference mark whose first vector indicates the direction of the axis of the cylinder.  
For the definition of the nautical angles, to see the operator AFFE\_CARA\_ELEM [U4.42.01] operand ORIENTATION.

### 3.5.7.5 Operand VECT\_NORMALE

- ◆ / VECT\_NORMALE = (X, there, Z)  
X there Z are the coordinates of a vector directing the axis of the cylinder.

### 3.5.7.6 Operand PRECISION

- ◇ PRECISION = eps  
eps is the tolerance with which one defines the membership of one node in the cylinder clothing.  
This tolerance is to be taken with the following direction:  
if  $d$  indicate the distance from the point running to the axis of the cylinder, one says that the point running belongs to the cylinder clothing if:  
 $|d-r| \leq \text{eps}$

### 3.5.8 Operand OPTION = 'PLAN'

This option makes it possible to define one GROUP\_NO constituted by nodes located on a line (in 2D) or in a plan (in 3D) except for a precision given.

#### 3.5.8.1 Operand NOT

- ◆ / NOT = (X, there) , in 2D  
(X, there, Z), in 3D  
X there Z are the punctual coordinates pertaining to the plan (with the right-hand side).

#### 3.5.8.2 Operand GROUP\_NO\_CENTRE

- ◆ GROUP\_NO\_CENTRE = grno  
This keyword makes it possible to define a node pertaining to the plan (with the right-hand side).

#### 3.5.8.3 Operand ANGL\_NAUT

- ◆ / ANGL\_NAUT = has , in 2D  
( , has B), in 3D  
Nautical angles has, B defined in degrees, are the angles allowing to pass from the total reference mark of definition of the coordinates of the nodes to a reference mark whose first vector is orthogonal with the plan 'medium' of the band.  
For the definition of the nautical angles, to see the operator AFFE\_CARA\_ELEM [U4.42.01] operand ORIENTATION.

#### 3.5.8.4 Operand VECT\_NORMALE

- ◆ / VECT\_NORMALE = (X, there) , in 2D  
(X, there, Z), in 3D

X there and Z are the components of a vector perpendicular to the plan (with the right-hand side).

### 3.5.8.5 Operand PRECISION

◇ PRECISION = eps

eps is the tolerance with which one defines the membership of a node in the plan (or with the right-hand side).

This tolerance is to be taken with the following direction:

if  $d$  indicate the distance from the node to the plan (or the right-hand side), it is said that this node belongs to this plan (or on this line) if:

$$|d| \leq \text{eps}$$

### 3.5.9 Operand OPTION = 'SEGM\_DROI\_ORDO'

This option is used to order a set of nodes roughly located on a segment of right-hand side AB.

◆ GROUP\_NO = gno2,

One defines the whole of the nodes which one wants to order.

◆ GROUP\_NO\_ORIG= gnoA ,

◆ GROUP\_NO\_EXTR= gnoB ,

The nodes are defined  $A$  and  $B$  , origin and end of the segment  $AB$  .

◆ PRECISION = prec,

◆ CRITERION = / 'RELATIVE' ,  
/ 'ABSOLUTE' ,

These two arguments are parapets, they are used to check that the nodes which one seeks to order (lno2 or gno2) are of course the segment  $AB$  . If the variation of a node with  $AB$  is higher than prec the code stops in fatal error.

If the selected criterion is 'RELATIVE' , the distance from a node with  $AB$  will be divided by the length of  $AB$  .

### 3.5.10 Operand OPTION = 'NOEUD\_ORDO'

This option is used to create one group\_no **ordered** containing the nodes of a set of meshes formed by segments (SEG2, SEG3 or SEG4). The whole of these meshes must form a continuous line. The line can be "open" (with 2 ends) or "closed" (it is then a simple loop).

◆ GROUP\_MA = gmaAB

Name of group\_ma which one wants to order the nodes.

Meshes of gmaAB must form an open line.

◇ GROUP\_NO\_ORIG= gnoA ,

◇ GROUP\_NO\_EXTR= gnoB ,

The keywords make it possible to define the nodes  $A$  and  $B$  , origin and end of the line  $AB$  .

The node  $A$  will be numbered in first, then one makes use of the topology of the meshes of gmaAB to number the nodes gradually.

If the node  $A$  is not provided by the user, the program will choose like node "origin", the first node of gmaAB who belongs only to only one nets segment. The origin is thus arbitrary: the program could just as easily have fallen on the other end.

It is checked that the last numbered node is well  $B$  (if this one is given).

## 3.5.10.1 Case of the closed lines

If the line is a loop, one cannot determine his ends automatically. To define the origin of the curvilinear X-coordinates, the user can to inform the nodes origin and end. It is necessary that these 2 nodes are identical. But it can also write: `ORIGINE=' SANS '`. The node "origin" will then be taken randomly on the closed line.

To direct a closed line, one cannot make use of knowledge of the nodes origin and end. If it wishes it, the user can then inform the keyword `VECT_ORIE` (2 or 3 cordonnées according to the dimension of space). One will choose as direction of course of the loop, the mesh of `gmaAB` who touches the node origin and who minimizes the angle with the vector provides by `VECT_ORIE`.

## 3.5.11 Operand `OPTION = 'TUNNEL'`

This option is used to create it `group_no` formed by the nodes located inside a "tunnel" which one provides the axis and the ray. The nodes selected will be those whose distance to the axis is lower than the ray.

The axis of the "tunnel" is defined by the linear meshes provided via the keyword `GROUP_MA_AXE`.

The axis of the tunnel must have a "origin" defined by the keyword `GROUP_NO_ORIG`.

The keyword `RAY` is used to define the "ray" of the tunnel.

One can limit the tunnel by giving his length by the keyword `LENGTH`. This length is measured starting from the origin of the tunnel.

The nodes candidates to be part of the tunnel are those carried by the meshes defined by the keywords: `TOUT=' OUI '` and `GROUP_MA`.

## 3.5.12 Operand `OPTION = 'INCLUSION'`

```
/  OPTION =  'INCLUSION',  
  ♦  GROUP_MA      = lgma ,           [l_group_ma]  
  ♦  CAS_FIGURE    =      / '2D'  
                        / '3D'  
                        / '2.5D'  
  ♦  DISTANCE_MAX = distma  
  ♦  GROUP_MA_INCL = lgma_inc ,      [l_group_ma]  
  ♦  MAILLAGE_INCL = ma_inc ,       [grid]
```

This option makes it possible to create the group of the nodes of the meshes of `lgma` who are geometrically inside the meshes of `Lgma_inc`.

If `MAILLAGE_INCL` is not provided, `lgma_inc` is a list of `GROUP_MA` grid which one enriches (`my`). If not it is `GROUP_MA` of `ma_inc`.

The keyword `CAS_FIGURE` is obligatory, it is used to determine which are the meshes of `lgma_inc` who must be used to determine inclusion:

'2D' : one is interested only in the surface meshes (`SORTED` and `QUAD`) of a grid 2D (plan `XOY`).  
'3D' : one is interested only in the voluminal meshes (`TETRA`, `PENTA`,...)  
'2.5D' : one is interested only in the surface meshes (`SORTED` and `QUAD`) of a grid 3D (hull).

The keyword `DISTANCE_MAX` is optional. It is used to give a small tolerance to determine whether a node is included in a mesh. Indeed, a node located "just" on an interface between 2 meshes, is likely to be regarded as "outside" with the 2 meshes and thus like not being part of inclusion. This is why a value by default of `distma` is taken by the code. One chose 1% length of the smallest edge of the grid `ma_inc`.

In the case of a grid of type "hull" (2.5D), if surface is not plane, it is almost impossible that an element of facet is geometrically included in other meshes: there is almost always a variation in the "normal" direction on the surface. It will thus be necessary in general, in this case, to provide a value of `distma` higher than the value by default.

### 3.5.13 Operand `OPTION = 'INTERVALLE_VALE'`

This option is used to create it `group_no` formed by the nodes of which the value of a component (`cmp`) of a field to the nodes (`cham_no`) lies between two values (`vmin` and `vmax`).

The field and the component which will be used to select the nodes are given by the keywords `CHAM_GD` and `NOM_CMP`.

Values `vmin` and `vmax` are provided via the keyword `VALE`.

Example:

```
DEFI_GROUP (reuse = E-MAIL, GRID = E-MAIL,  
           CREA_GROUP_NO = _F (NOM=' GN700', OPTION=' INTERVALLE_VALE',  
                               CHAM_GD=TEMPER, NOM_CMP=' TEMP', VALE= (700. , 800.),),),);
```

`GROUP_NO 'GN700'` will be made of all the nodes of the grid `E-MAIL` of which the temperature in the field `TEMPER` is understood enters `700.` and `800.`

### 3.5.14 Operand `OPTION = 'FISS_XFEM'`

This option makes it possible to recover the group of the nodes of the type `XFEM` specified by the keywords `TYPE_GROUP`.

◆ `CRACK = (fiss1, fiss2,...)`

◆ `TYPE_GROUP =`

/ `'XFEM'` : the node will be retained if it is an enriched node

/ `'HEAVISIDE'` : the node will be retained if it is a node enriched by Heaviside type

/ `'CRACKTIP'` : the node will be retained if it is a node enriched by Cracktip type

/ `'MIXED'` : the node will be retained if it is a node enriched by Mixed type (Heaviside and

Cracktip)

/ `'ZONE_MAJ'` : the node will be retained if it is contained in the zone of update of the level

sets. If the grid of the crack passed by the keyword `GRID` ( § 3.2 ), the zone of update coincides:

- in the absence of an auxiliary grid associated with the crack, with the field of calculation around the bottom,

- in the presence of an auxiliary grid associated with the crack, with the field of projection between grid and grid , independently of the field of calculation used on the grid.

If a grid passed by the keyword `GRID` ( § 3.2 ) , the zone of update always coincides with the field of calculation used on the grid.

/ `'TORUS'` : the node will be retained if it is contained in a torus built around the bottom of crack of ray given by the keyword `RAYON_TORE` . If the localization of the field were used for the calculation of the crack, this option cannot be selected. In this case, the group of nodes is created by using the zone of update (one selects the option automatically `TYPE_GROUP=' ZONE_MAJ'` ) and the choice of the user is ignored.

For a definition specifies concepts of enriched node, node Heaviside, node Ace-tip, auxiliary grid, field of calculation and its localization, to see R7.02.12, §3.2.5.

### 3.5.15 Operand `RAYON_TORE`

# If `TYPE_GROUP=' TORE'`

◆ `RAYON_TORE = R,`

The ray is specified `R` torus to be used for the selection of the nodes.

### 3.5.16 Operand `OPTION = 'RELA_CINE_BP'`

For each triplet of connections (3 directions of space) contents in the list of relations kinematics of a concept `cable_precont` resulting from `DEFI_CABLE_BP`, this option created a group of nodes containing it node of cable and them nodes of concrete concerned. The name of this group is the value given to `PREF_GRNO` followed by the name of node of cable.

◆ `CABLE_BP = cable_bp`  
Name of the concept `cable_precont`.

◆ `PREF_GRNO = pref`  
Prefix given to the groups of nodes created.

### 3.5.17 Operands `GROUP_MA` and `NAME`

/ `GROUP_MA = lgma`

For each group of meshes of the list `lgma`, one creates a group of nodes formed by the nodes carried by the meshes of this group of meshes.

◇ `NAME = lgno`

If `lgno` is provided by the user, this list must be of the same length than `lgma`. These are the names that one wants to give to the new groups of nodes.

If `lgno` is not provided, the groups of nodes will bear the same names as the groups of meshes which gave them rise.

◇ `CRIT_NOEUD =`

/ `'ALL'` [`DEFECT`] : all the nodes of each mesh are taken.

/ `'TOP'` : one takes only the nodes "top" of the meshes (i.e. ends of the edges).

/ `'MEDIUM'` : one takes only the nodes "medium" of the edges of the meshes.

/ `'CENTER'` : one takes only the nodes which is neither "top" nor "medium" it is - with - to say the nodes to the center of the facets or the voluminal elements.

### 3.5.18 Operand `TOUT_GROUP_MA`

/ `TOUT_GROUP_MA = 'YES'`

This keyword has the same meaning as the precedent, except that one creates groups of nodes for **all** existing groups of meshes of the grid.

### 3.5.19 Operand `ALARM = 'YES' [DEFECT] / 'NOT'`

if `ALARM = 'NOT'`, the code does not emit alarm; for example when one asks him to create one `GROUP_NO` and that this group is empty. The value by default of this keyword is `'YES'`.

### 3.5.20 Operand `INFORMATION`

if `INFORMATION = 1`, one print in the file `'MESSAGE'`, the number of groups create and for each group, the name of the group and the number of entities the component.

if `INFORMATION = 2`, one prints in the file `'MESSAGE'`, the number of groups create and for each group, the name of the group, the number of entities the component then the list of the entities setting up the groups.

## 4 Examples

### Example 1 (topological criteria and logics):

That is to say `my` a grid containing the groups of meshes already:

```
M1 M2 M3
```

and groups of nodes:

```
N1 N2 N3
```

```
my = DEFI_GROUP (reuse = my, GRID = my,
  CREA_GROUP_MA = ( _F ( NAME = NM1, GROUP_MA = (GMA7, GMA9,...) ),
                    _F ( NAME = NM2, UNION = (M1, NM1) ),
                    _F ( NAME = NM3, DIFFE = (NM2, M2) ), ),
  CREA_GROUP_NO = _F ( TOUT_GROUP_MA = 'YES' ), )

my = DEFI_GROUP (reuse = my, GRID = my,
  CREA_GROUP_MA = _F ( NAME = NM4, GROUP_MA = (GMA7, GMA11, GMA13) )
  CREA_GROUP_NO = ( _F ( NAME = NN1, INTERSEC= (NM1, N1) ),
                   _F ( GROUP_MA = NM4 ) ) )
```

After these two calls to the order `DEFI_GROUP`, the grid contains then:

- groups of meshes:
  - M1, m2, m3 (initial)
  - NM1 = (meshs: MA7, MA9,...)
  - NM2 = M1 "union" NM1
  - Nm3 = NM2 "minus" m2
  - NM4 = (MESHS: MA7, MA11, MA13)
- groups of nodes:
  - N1, N2, N3 (initial)
  - M1, M2, M3, NM1, NM2, NM3: `group_no` containing the nodes of `group_ma` of same names. These `group_no` are created by the 1<sup>era</sup> order `DEFI_GROUP`.
  - NN1 = NM1 "intersection" N1
  - NM4 = (nodes of `group_ma` NM4)

### Example 2 (geometrical criteria):

```
my = DEFI_GROUP (reuse = my, GRID = my,
  CREA_GROUP_MA= ( _F (NAME = facesup , OPTION = 'FACE_NORMALE',
                      VECT_NORMALE = (0. , 0. , 1.)),
                  _F (NAME = S01 , OPTION = 'SPHERE',
                      NOT = (0. , 0. , 0.), RAY = 1.)),
  CREA_GROUP_NO = ( _F (NAME = BO_S01, OPTION = 'ENV_SPHERE',
                      POINT= (0. , 0. , 0.), RAYON=1., PRECISION=0.01),
                  _F (NAME = S01_1 , GROUP_MA = S01),
                  _F (NAME = S01_2 , OPTION = 'ENV_SPHERE',
                      POINT= (0. , 0. , 0.), RAYON=0.5,
                      PRECISION=0.5) ), )
```

Afterwards `DEFI_GROUP` grid `my` will contain 2 new `GROUP_MA` and 3 new `GROUP_NO` :

- `facesup` contains the facets whose normal is directed according to  $OZ$  (towards  $Z > 0$ ),
- `S01` contains **all** the meshes of which **one of the nodes** belongs to the sphere of ray 1. and centered in  $O$  (origin of the axes),
- `BO_S01` is the group of the nodes which are in the vicinity of the envelope of the preceding sphere (`S01`),
- `S01_1` is the group of all the nodes of the meshes of the group of meshes `S01`; caution: certain nodes of this group can be outside the sphere!
- `S01_2` is the group of the nodes included in the sphere `S01`:  $|d(M, O) - 0.5| \leq 0.5$