

## Structures of data sd\_num\_d, sd\_num\_equa, sd\_stockage and sd\_prof\_chno

---

### Summary:

This document describes the structures of data NUM\_D, NUM\_EQUA, STORAGE and PROF\_CHNO used to define the classification of the unknown factors of the linear systems and the storage of the assembled matrices and the vectors solution and second member.

## Contents

---

<a href="#">1 General information.....</a>	<a href="#">3</a>
<a href="#">2 Tree structures.....</a>	<a href="#">4</a>
<a href="#">3 The object NUME_DDL.....</a>	<a href="#">6</a>
<a href="#">4 The object NUME_EQUA.....</a>	<a href="#">6</a>
<a href="#">5 The object PROF_CHNO.....</a>	<a href="#">6</a>
<a href="#">5.1 Object PRNO.....</a>	<a href="#">7</a>
<a href="#">5.2 Object LILL.....</a>	<a href="#">8</a>
<a href="#">5.3 Object NUEQ.....</a>	<a href="#">8</a>
<a href="#">5.4 Object DEEQ.....</a>	<a href="#">10</a>
<a href="#">6 The object NUML_EQUA.....</a>	<a href="#">11</a>
<a href="#">7 The object STORAGE.....</a>	<a href="#">12</a>
<a href="#">7.1 The object STOC_LIGN_CIEL.....</a>	<a href="#">12</a>
<a href="#">7.2 The object STOC_MORSE.....</a>	<a href="#">13</a>
<a href="#">7.3 The object MULT_FRONT.....</a>	<a href="#">14</a>
<a href="#">8 Examples.....</a>	<a href="#">15</a>

## 1 General information

One NUME\_DDL is used to define the classification of the unknown factors (and equations) of a linear system. It is pointed out that the unknown factors of such a system are components carried by nodes of GRID (or of the late nodes of LIGREL).

The matrices which one wants to describe are square, hollow, symmetrical or not.

When a matrix is not-symmetrical, its hollow structure is symmetrical.

The classification of the unknown factors of a system is similar to that of the components of one CHAM\_NO . Moreover the SD PROF\_CHNO described in this document is that referred by the SD CHAM\_NO [D4.06.05 - Structure of Data field].

The SD NUME\_DDL also the description of the tables of storage of the values of the assembled matrices [D4.06.10 - Structure of Data contains sd\_matr\_asse ]. One calls STORAGE this part of the structure of data.

The relations of dependence between these objects can be represented by:

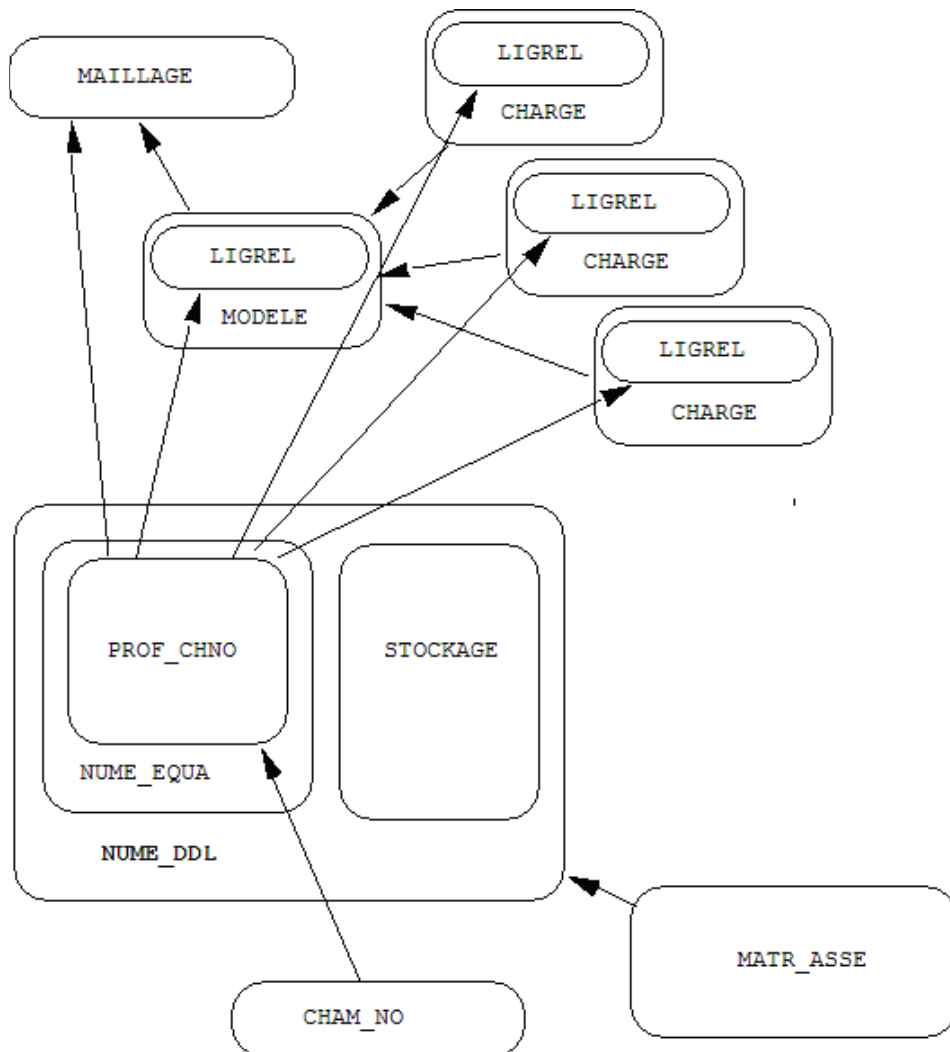


Figure 1-a: Links of the NUME\_DDL, NUME\_EQUA, STORAGE and PROF\_CHNO with the other structures of data

## 2 Tree structures

Here the tree structure of the various objects presented in this document.

```
NUME_DDL (K14):: =record
  ◆ .NUMÉRIQUE      NUME_EQUA
  ◆ .NUML           NUML_EQUA
  ◆ .NSLV           OJB S V K24 dim=1

NUME_EQUA (K19):: =record
  ◆ .NEQU           OJB S V I dim=2
  ◆ .REFN           OJB S V K24 dim=4
  ◆ .DELG           OJB S V I

PROF_CHNO (K19):: =record
  ◆ .PRNO           OJB XC V I NUM
  ◆ .LILI           OJB S NR K24
  ◆ .NUEQ           OJB S V I
  ◆ .DEEQ           OJB S V I

NUML_EQUA (K19):: =record
  ◆ .JOIN           OJB S V I
  ◆ .PRNO           OJB XC V I
  ◆ .NEQU           OJB S V I dim=2
  ◆ .DELG           OJB S V I
  ◆ .NLGP           OJB S V I
  ◆ .NUEQ           OJB S V I
  ◆ .NULG           OJB S V I
  ◆ .NUGL           OJB S V I
  ◆ .PDDL           OJB S V I

STORAGE (K14):: =record
  ◆ .SMOS           STOC_MORSE # to describe the initial hollow matrix
  ◇ % storage adapted to the solver 'LDLT'
  .SLCS            STOC_LCIEL # is used to describe the factorized matrix
  ◇ % after factorization by 'MULT_FRONT'
  .MLTF            MULT_FRONT # to describe the factorized matrix

STOC_LCIEL (K19):: =record
  ◆ .SCBL           OJB S V I
  ◆ .SCDI           OJB S V I
  ◆ .SCDE           OJB S V I
  ◆ .SCHC           OJB S V I
  ◆ .SCIB           OJB S V I
  ◇ % If storage is that adopted to factorize with LDLT:
  % line storage of sky uses classification 'RCMK'
  % storage "Morse" uses initial classification
  ◆ .M2LC           OJB S V I
  ◆ .LC2M           OJB S V I

STOC_MORSE (K19):: =record
  ◆ .SMDI           OJB S V I
  ◆ .SMDE           OJB S V I
  ◆ .SMHC           OJB S V I

MULT_FRONT (K19):: =record
  ◆ .ADNT           OJB S V I4
```

# Code\_Aster

Version  
default

Titre : Structures de données sd\_numd\_dcl, sd\_numd\_equa, s[...]  
Responsable : PELLET Jacques

Date : 07/08/2017 Page : 5/15  
Clé : D4.06.07 Révision :  
c6b8d5b774ba

◆ .GLOB	OJB	S	V	I4
◆ .LOCL	OJB	S	V	I4
◆ .PNTI	OJB	S	V	I4

## 3 The object `NUME_DDL`

The object `NUME_DDL` is used to define the classification of the unknown factors (and equations) of a linear system.

```
NUME_DDL (K14):: =record
  ♦ .NUMÉRIQUE      NUME_EQUA
  ♦ .NUML           NUML_EQUA
  ♦ .NSLV           OJB S V K24 dim=1
```

It thus contains three objects. First refers to the object `NUME_EQUA` (see §4):

```
NUME_DDL (1:14) / '.NUMÉRIQUE'=NUMÉRIQUE_EQUA
```

Second refers to the object `NUML_EQUA` (see §6):

```
NUME_DDL (1:14) / '.NUML'=NUML_EQUA
```

The last object `.NSLV` store the name of `sd_solvor` who will be used by default.

```
NSLV (1) : name of sd_solvor who will be used by default.
```

## 4 The object `NUME_EQUA`

The object `NUME_EQUA` refers to the classification of the equations. It contains three objects:

```
NUME_EQUA (K19):: =record
  ♦ .NEQU           OJB S V I      dim=2
  ♦ .REFN           OJB S V K24   dim=4
  ♦ .DELG           OJB S V I
```

The object `.REFN` store some information:

- `REFN (1)` : name of the grid subjacent with `NUME_DDL` ;
- `REFN (2)` : name of the size simple partner (`TEMP_R`, `DEPL_R`, `PRES_C`,...);
- `REFN (3)` : linear solvor by default (`MUMPS`, `MULT_FRONT`);
- `REFN (4)` : I nutilized.

The object `.NEQU` store dimensions:

- `NEQU (1)` : full number of equations (length of `.VALE` nodal fields) ;
- `NEQU (2)` : full number of degrees of freedom (length of `.NUEQ` in `PROF_CHNO`).

The object `.DELG` described (a little) the equations of the type "Lagrange":

- `DELG (ieq) =-1` : SI the equation `ieq` corresponds to the component `LAGR` range by second node of a mesh `SEG3` carrying an element of Lagrange (Lagrange 1) ;
- `DELG (ieq) =-2` : if the equation `ieq` corresponds to the component `LAGR` range by the third node of a mesh `SEG3` carrying an element of Lagrange (Lagrange 2);
- `DELG (ieq) =0` : the equation is not of type "Lagrange".

## 5 The object `PROF_CHNO`

The structure of data `PROF_CHNO` the profile of classification defines, i.e. the description of the unknown factors. In *Code\_Aster*, the unknown factors are necessarily carried by nodes. Nodes of one `PROF_CHNO` are of two types:

- "Physical" nodes of the grid `my` (concerned with the model);

- Late nodes<sup>1</sup> of one (or several) `LIGREL` who are based on the grid `my`.

Objects of `PROF_CHNO` are the following:

```
PROF_CHNO (K19):: =record
    ◆ .PRNO          OJB XC V I NUM
    ◆ .LILI          OJB S  NR K24
    ◆ .NUEQ          OJB S  V I
    ◆ .DEEQ          OJB S  V I
```

## 5.1 Object `PRNO`

The object `.PRNO` is a numbered contiguous collection of vector of entireties.

```
◆ .PRNO          OJB XC V I NUM
```

This collection describes which are the components carried by the nodes implied in `PROF_CHNO` by indicating the name of `LIGREL` of model and those of `LIGREL` with meshes and/or late nodes. If it is about one `LIGREL` with meshes and late nodes (`DDL_IMPO`, `LIAISON_DDL` ...), that makes it possible to identify the late nodes implied in `PROF_CHNO`. On the other hand, if it is about one `LIGREL` only with late meshes (`FORCE_NODALE` ...), it does not point towards any object of collection of `.PRNO`.

What wants to say that the object `PRNO` (1) inform about the degrees of freedom carried by the nodes of the grid (on which necessarily the model is based) subjacent with `PROF_CHNO` and that other objects of the collection `.PRNO` (when they exist) inform about the degrees of freedom carried by the nodes which do not belong to the grid (late nodes).

The late nodes are defined in `LIGREL`. With each `LIGREL` `nomlig` an object corresponds `.PRNO` (`ili`) of index `ili`. The number of `LIGREL` is that corresponding to `nomlig` in the pointer of name `.LILI`.

For example, the collection `.PRNO` will contain:

- `.PRNO` (1) : nodes of the grid `my` ;
- `.PRNO` (2) : late nodes of `LIGREL` whose name is in `.LILI` (2) ;
- `.PRNO` (3) : late nodes of `LIGREL` whose name is in `.LILI` (3) ;
- ...

If `nec` is the number of coded entireties of the size associated with `PROF_CHNO` :

- `.PRNO` (1) is length  $(nb\_noeuds(my)) * (2+nec)$  ;
- `.PRNO` (`lili`) is length  $(nb\_noeuds\_tardifs(LIGREL(ili))) * (2+nec)$ .

Each node is described by two entireties and a vector of coded entireties length `nec` that one calls the descriptor-size [D4.06.05 - Structure of Data field].

Let us take the example of the nodes of the grid `my` :

- `nb_cmp` is the component count increased by the node `ino` grid;
- `ieq` is the address in the object `.NUEQ` first component carried by the node `ino`.

```
v = PROF_CHNO (1)
v (ino-1) * (2+nec) +1) = ieq
v (ino-1) * (2+nec) +2) = nb_cmp
<
v (ino-1) * (2+nec) +2+1) |
... | Descriptor-size of the node ino
v (ino-1) * (2+nec) +2+nec |
<
```

Note:

1 A late node in Code\_Aster corresponds to a node added by a dualized limiting condition (`DDL_IMPO`, `LIAISON_DDL`,...). This node was added during the creation of the load and was stored in `LIGREL` corresponding to carry Lagrange. They do not appear in the structure of data grid and they have negative numbers.

All the components carried by the same node are consecutive in `.NUEQ`. This is why one does not store that the address of first. The components are ordered in the order of the catalogue of the sizes. Unfortunately, size associated with `.PRNO` is not stored in `PROF_CHNO`.

## 5.2 Object LILI

The object `.LILI` is a pointer of names which gives access the elements of the collection `PRNO`.

◆ `.LILI` OJB S NR K24

By convention, one stores in `LILI (1)` the value `&MAILLA`, which wants to say that the object `PRNO (1)` inform about the degrees of freedom carried by the nodes of the grid subjacent with `PROF_CHNO`. Nevertheless the name of the grid is not stored.

Other objects of the collection `.PRNO` (when they exist) inform about the degrees of freedom carried by the nodes which do not belong to the grid (late nodes). These late nodes are defined in `LIGREL`. With each `LIGREL nomlig` an object corresponds `.PRNO (ili)`. The number of `LIGREL` is that corresponding to `nomlig` in the pointer of name `.LILI`.

## 5.3 Object NUEQ

The object `.NUEQ` is a vector of entreties.

◆ `.NUEQ` OJB S V I

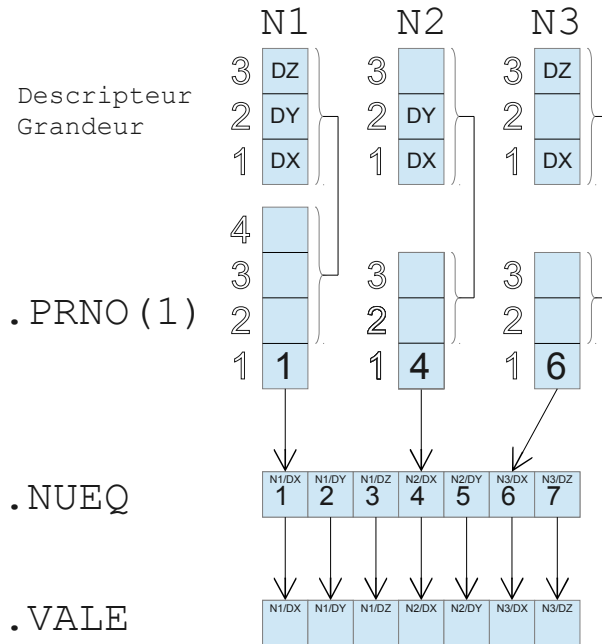
It is a vector of indirection between the object `.PRNO` and the object `.VALE CHAM_NO` who refer it `PROF_CHNO`. This vector of indirection makes it possible to be freed from the rule according to which the components of a node are followed in the order of the catalogue of the sizes but also that each component corresponds indeed to an unknown factor in `NUME_DDL`.

In practice, this vector is almost always the identity (`NUEQ (I) =I`). This case more the general mean two things:

1. There are as many unknown factors in components on the nodes corresponding to `LIGREL` model and on the late nodes of `LIGREL` late of the dualized loadings;
2. Classification is a tautology: the unknown factors are in the order of the components of the catalogue on each node.

On the following example, there are three nodes. The first node has as components `DX`, `DY` and `DZ`, the second node has as components `DX` and `DY` and the third node has as components `DX` and `DZ`. In this case, the object `.NUEQ` is such as `NUEQ (I) =I`:

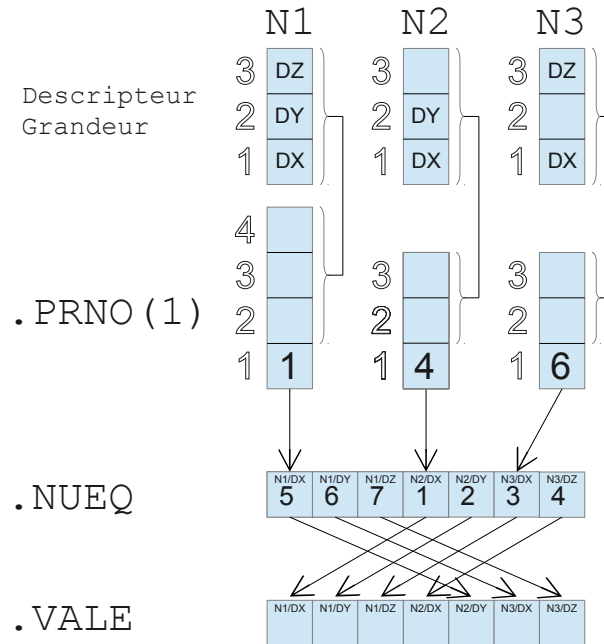




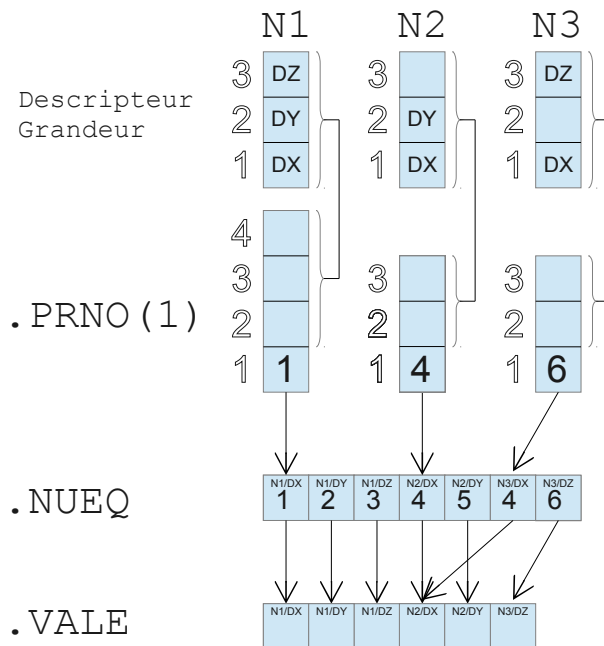
Nevertheless, this vector is not also any more commonplace in two precise cases:

- When one creates `macr_elem_stat` with an aim of separating the internal degrees of freedom from the external degrees of freedom of the macronutrient. One can then factorize the matrices partially (the part corresponding to the internal dds);
- When one defines relations of identity between certain degrees of freedom. One uses this case in certain situations (XFEM and contact of the LAKE type).

For the first case, one always has `.NUEQ` and `.VALE` of the same length. On the other hand, the order is not inevitably the same one. For example, one takes again the preceding case by saying that node 1 must be in the last (it corresponds to a macronutrient which must be in last position to be able to profit from partial factorizations in the solver `LDLT`). The diagram becomes the following:



It is always a bijection enters .NUEQ and .VALE but it is not any more one tautology (  $NUEQ(I) \neq I$  ). Finally the last case is the taking into account of relation of identity between two degrees of freedom. For example, the same case is taken again, except that this time, it is written that displacement DX second node is equal to displacement DX third node.



This time, there is not any more a bijection: there is less of unknown factors than of degrees of freedom.

Note:

If the solver is MULT\_FRONT, it is checked that the vector corresponds to the identity. What means that one cannot use this solver in the case of the static macronutrients or when there exist relations of identity between degrees of freedom. In these cases, solver MUMPS should be used.

## 5.4 Object DEEQ

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 2021 EDF R&D - Licensed under the terms of the GNU FDL (<http://www.gnu.org/copyleft/fdl.html>)

The object `.DEEQ` is a vector of entireties length  $2 * neq$ . With `neq` is the number of equations of `PROF_CHNO`.

```
◆ .DEEQ          OJB S  V I DIM=2*neq
```

It is a vector “reverses” `.PRNO` who describes (partially) the equations.

If `neq` is a number of equation (i.e addresses in the object `.VALE`), then:

```
V ((neq-1) *2+1): ino
V ((neq-1) *2+2): ICMP
```

With following convention:

- If `ino > 0` and `icmp > 0` : `neq` is the equation associated with `ICMP`<sup>ième</sup> component range by the node `ino` grid.
- If `ino > 0` and `icmp < 0` : `neq` is one of the two equations which dualisent the blocking of `ICMP`<sup>ième</sup> component node `ino` grid.
- If `ino = 0` and `icmp = 0` : `neq` is an equation of dualisation of a linear relation between several components.

*Remarque:*

In the case of the taking into account of the relations of identity in classification, as several nodes/component correspond to the same equation, by convention, one indicates the first couple node/component.

## 6 The object `NUML_EQUA`

The object `NUML_EQUA` refers to the classification of the equations if the solver used is `MUMPS` and that the user activated the option `MATR_DISTRIBUE=' OUI '`.

```
NUML_EQUA (K19):: =record
  ◆ .JOIN          OJB S  V I
  ◆ .PRNO          OJB XC V I
  ◆ .NEQU          OJB S  V I    dim=2
  ◆ .DELG          OJB S  V I
  ◆ .NLGP          OJB S  V I
  ◆ .NUEQ          OJB S  V I
  ◆ .NULG          OJB S  V I
  ◆ .NUGL          OJB S  V I
  ◆ .PDDL          OJB S  V I
```

◆ `.JOIN`

This vector is dimensioned with the number of couplings of the graph of communication. For each coupling, one finds the processor in link with the current processor. If the value is -1 then the processor running does not take part in this coupling.

◆ `.PRNO`

This object is identical (in its architecture) to `.PRNO` of one `PROF_CHNO`. The difference lies in the fact that the classification of the degrees of freedom is local for the processor considered.

◆ `.NEQU`

- `NEQU (1)` : local number of equations (length of `.VALE` nodal fields );
- `NEQU (2)` : local number of degrees of freedom (length of `.NUEQ` in `PROF_CHNO`).

◆ Local `.DELG` `dim=neq`

This object is similar to the object `.DELG` of one `NUME_EQUA`.

◆ `.NLGP`

This vector provides the total number of the ddl for PETSc. This solver needs contiguous blocks of lines. With this intention, one adapts classification for him.

- ◆ Local `.NUEQ dim=neq`  
It is a vector of indirection between the object `.PRNO` and the object `.VALE CHAM_NO`. See `.NUEQ` of one `NUME_EQUA`.
- ◆ `.NULG`  
It is a vector of indirection which makes it possible to pass from the local classification of the degrees of freedom (of `.NUML`) with the total classification (of `.NUMÉRIQUE`)
- ◆ `.NUGL`  
It is a vector of indirection which makes it possible to pass from the total classification of the degrees of freedom (of `.NUMÉRIQUE`) with the local classification (of `.NUML`)
- ◆ `.PDDL`  
It is a vector which makes it possible to know by which processor are had the local ddl.

## 7 The object STORAGE

The hollow matrices of Code\_Aster have a whole a symmetrical structure: if `has (I, J)` exist, then `has (J, I)` also exist.

For a hollow symmetrical matrix, one stores only his "higher" part (`has (I, J)` for  $J \geq I$ ). The structure of data `STORAGE` described how the nonworthless terms of the upper part of matrix are stored.

For a nonsymmetrical matrix, one stores the upper part and the lower part. As these 2 parts have the same structure (except for a transposition), the storage of the upper part is enough.

Note:

*Although theoretically, the nonworthless terms of a matrix can be had unspecified way, the filling of the matrices of Code\_Aster is such as all DDLs carried by a node are connected to all DDLs carried by the other nodes being next to this node via a finite element. The matrix is thus formed small rectangles full corresponding to connectivity with the nodes of the model. In particular, there exist small blocks on all the diagonal of the matrix.*

### 7.1 The object STOC\_LIGN\_CIEL

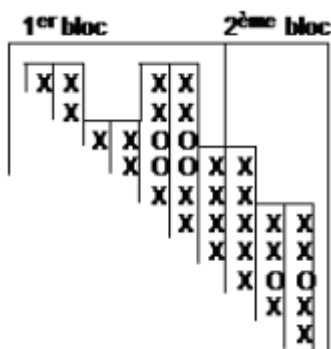
For the online storage of sky, the objects are the following:

```
STOC_LCIEL (K19):: =record
  ◆ .SCBL      OJB  S  V  I
  ◆ .SCDI      OJB  S  V  I
  ◆ .SCDE      OJB  S  V  I
  ◆ .SCHC      OJB  S  V  I
  ◆ .SCIB      OJB  S  V  I
  ◇ . M2LC     OJB  S  V  I
  ◇ .LC2M     OJB  S  V  I
```

<code>.SCDE</code>	<code>S V I dim=4</code>	
	(1)	many equations <code>neq</code>
	(2)	size of the blocks of the matrix <code>t_bloc</code>
	(3)	number of blocks necessary to the storage of the values of the matrix <code>n_bloc</code>
	(4)	maximum height of the columns of the matrix
<code>.SCHC</code>	<code>S V I dim=neq</code>	

	(I)	height of l <sup>ème</sup> column
.SCDI	S V I	dim=neq
	(I)	address of the diagonal term of l <sup>ème</sup> column in its block
.SCBL	S V I	dim=n_bloc+1
	(1) (K+1)	0 number of the last column of the block K notice : a column can belong only to one block
.SCIB	S V I	dim=neq
	(I)	number of the block which contains l <sup>ème</sup> column of the matrix
. M2LC	S V I	dim=neq
	ieq_m	ieq_lc : give the correspondence between the classification of storage "Morse" (ieq_m) and the classification of storage "line of sky" (ieq_lc)
.LC2M	S V I	dim=neq
	ieq_lc	ieq_m : give the correspondence between the classification of storage "line of sky" (ieq_lc) and the classification of storage "Morse" (ieq_m)

Example:



SCDI (1) =	1	SCDI (6) =	17
SCDI (2) =	3	SCDI (7) =	22
SCDI (3) =	4	SCDI (8) =	6
SCDI (4) =	6	SCDI (9) =	11
SCDI (5) =	11	SCDI (10) =	17
SCHC (1) =	1	SCHC (6) =	6
SCHC (2) =	2	SCHC (7) =	5
SCHC (3) =	1	SCHC (8) =	6
SCHC (4) =	2	SCHC (9) =	5
SCHC (5) =	5	SCHC (10) =	6
SCBL (1) =	0	SCIB (1 to 7) =	1
SCBL (2) =	7	SCIB (7 to 10) =	2
SCBL (3) =	10		

## 7.2 The object STOC\_MORSE

For Morse storage, the objects are the following:

STOC\_MORSE (K19):: =record

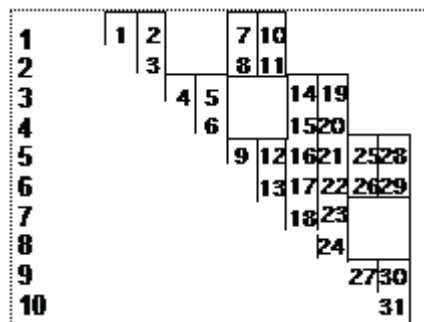
Warning : The translation process used on this website is a "Machine Translation". It may be imprecise and inaccurate in whole or in part and is provided as a convenience.

Copyright 2021 EDF R&D - Licensed under the terms of the GNU FDL (<http://www.gnu.org/copyleft/fdl.html>)

- ◆ .SMDI            OJB    S   V   I
- ◆ .SMDE           OJB    S   V   I
- ◆ .SMHC           OJB    S   V   I

.SMDE	S V I dim=3
(1)	many equations neq
(2)	many terms stored in the half-matrix n_termes
(3)	number of block 1
.SMHC	S V I dim=n_termes
(I)	number of line of I <sup>ème</sup> stored term
.SMDI	S V I dim=neq
(I)	address of the diagonal term of I <sup>ème</sup> column in the block It is necessary thus that all the diagonal terms are stored in the matrix

Example:



```

SMDI (1) = 1          SMDI (6) = 13
SMDI (2) = 3          SMDI (7) = 18
SMDI (3) = 4          SMDI (8) = 24
SMDI (4) = 6          SMDI (9) = 27
SMDI (5) = 9          SMDI (10) = 31

```

```

SMHC (1) = 1
SMHC (2) = 1
SMHC (3) = 2
SMHC (4) = 3
SMHC (5) = 3
SMHC (6) = 4
SMHC (7) = 1
SMHC (8) = 2
SMHC (9) = 5
...
SMHC (28) = 5
SMHC (29) = 6
SMHC (30) = 9
SMHC (31) = 10

```

## 7.3 The object MULT\_FRONT

For storage MULT\_FRONT., the objects are the following:

```

MULT_FRONT (K19):: =record
  ◆ .ADNT            OJB    S   V   I4
  ◆ .GLOB            OJB    S   V   I4

```

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 2021 EDF R&D - Licensed under the terms of the GNU FDL (<http://www.gnu.org/copyleft/fdl.html>)

◆ .LOCL OJB S V I4  
◆ .PNTI OJB S V I4

That is to say `lgind`, the sum amongst neighbors of the super-nodes.

- ◆ .GLOB OJB S V I4 dim=`lgind`  
This vector gives the whole of the neighbors of the super-nodes
- ◆ .LOCL OJB S V I4 dim=`lgind`  
This vector establishes for the numbers of lines of the super-nodes, the correspondence between the local classification of the son and the local classification of the father.
- ◆ .ADNT OJB S V I4 dim=`matr_init`  
Dimension `matr_init` is the size of the initial matrix (Morse). It is the vector of the addresses of the initial terms in the factorized matrix.
- ◆ .PNTI OJB S V I4 dim=`19*neq+10`  
Office plurality in the same vector of a succession of work tables.

## 8 Examples

---

An example of `NUMD_DDL` associated to the three linear solveurs ('LDLT', 'MULT\_FRONT', 'GCPC') in the document [D4.06.10 - Structure of Data is given `sd_matr_asse`].