Structure of Data sd_matr_asse

Summary:

This document describes the structures of data sd_matr_asse : hollow matrices.

Contents

1 General information

Objects of the type sd_matr_asse represent the square assembled matrices (within the meaning of the finite elements). It is in general of large objects. These matrices are hollow, which explains why their structure is not simply a square table.

One sd_matr_asse can come from an assembly from sd_matr_elem or of a linear combination of others sd_matr_asse.

Tables describing the storage of sd_matr_asse are in the structure sd_stockage of one sd_nume_ddl [D4.06.07].

There exists sd_matr_asse symmetrical and of sd_matr_asse not-symmetrical. But it is supposed that the topology of the matrix is always symmetrical. I.e. that the nonworthless terms are laid out symmetrically compared to the diagonal.

2 Tree structures

sd_matr_asse (K19) :: =record

♦ '.REFA' : OJB S V K24

♦ '.VALM' : OJB XD V R/C NUM () nbobj=1/2

◊ # if there exist ddls of Lagrange:
♦ '.CONL': OJB S V R

◊ # if sd_matr_asse comes in one way or another from elementary matrices:
♦ '.LIME': OJB S V K24

◊ # if sd_matr_asse (routine "was factorized" preres. F90) :
/ # if factorized with MULT_FRONT :
♦ '.VALF : OJB XD V R/C NUM ()
◊ # if the matrix is nonsymmetrical:
♦ '.WALF: OJB XD V R/C NUM ()
/ # if factorized with LDLT :
♦ '.UALF : OJB XD V R/C NUM ()
/ # if factorized with GCPC + LDLT_INC :
♦ '.PERM : OJB S V I
/ # if factorized with LDLT or MULT_FRONT :
♦ '.DIGS : OJB S V R/C

◊ # if there exist loads kinematics:
♦ '.CCID' : OJB S V I
♦ '.CCLL' : OJB S V I
♦ '.CCVA' : OJB S V R/C
♦ '.CCII' : OJB S V I

3 Contents of the OJB

3.1 Object .REFA

.REFA S V K24 dim=20

.REFA (1) = name of sd_maillage subjacent.

.REFA (2) = name of sd_nume_ddl.

.REFA (3) = '' / 'ELIMF' / 'ELIML'
'' : There do not exist loads kinematics.
'ELIMF' : There exist loads kinematics.
Part of .VALM was copied in .CCVA
.VALM was partially replaced by a matrix unit (on the eliminated ddls)
'ELIML' : There exist loads kinematics.
.CCVA was not created yet
.VALM was not recopied yet in .CCVA
to see mtmchc.f for the details

.REFA (4) = name ofOPTION of calculation (or the chain: '&&MELANGE').

.REFA (5) = ''

.REFA (6) = ''

.REFA (7) = '' / nomsolv
nomsolv : name of the solvor to be used (by default) at the time of the resolutions.

.REFA (8) = '' / 'ADZE' / 'DECT' / 'DECP'
'' or 'ADZE' : matrix in its initial state (not factorized)
'DECT' : entirely factorized matrix
'DEPT' : partially factorized matrix (possible only if LDLT)

.REFA (9) = 'Ms' / 'MR.'
'Ms' : symmetrical matrix
'MR.' : not-symmetrical matrix

.REFA (10) = / 'NOEU': the ddls of the matrix are carried by nodes
/ 'GENE': the ddls of the matrix are generalized ddls.

.REFA (11) = 'MPI_COMPLET' / 'MPI_INCOMPLET' / 'MATR_DISTR'
'MPI_COMPLET' : Objects .VALM (and .CCVA) are "complete"
'MPI_INCOMPLET' : Objects .VALM (and .CCVA) are "incomplete" because of a calculation MPI distributed. Each processor does not assemble that the finite elements which are affected for him.
'MATR_DISTR' : If MATR_DISTRIBUEE=' OUI'
This option dimensions with just the matrix assembled according to the number of unknown factors present on the current processor.

Notice :
if 'MPI_INCOMPLET', objects .VALM and .CCVA are allocated with their normal size, but they are partially filled of "0". if 'MATR_DISTR', the object .VALM is this time of a size smaller than in the case 'MPI_INCOMPLET'.

.REFA (17) = '' / 'XFEM_PRECOND'
'' : No pre-conditioner XFEM was applied to the matrix. The resolution continues without particular treatment, relating to prepacking XFEM.
'XFEM_PRECOND' : The matrix was modified using pre-conditioner XFEM. It is necessary to take into account this modification in the continuation of calculation.

Notice :

If the matrix is modified by pre-conditioner XFEM, that implies, that there is at least a pre-conditioner object stored in memory. This object is useful for the continuation of calculation (for the modification of the second member and the solution).

.REFA (18) = '' / '&&XFEM_PC_1'

'' : No pre-conditioner XFEM is stored.

'&&XFEM_PC_1' : Pre-conditioner XFEM is stored in a site " JEVEU " **fixed**.

Concerning the format of storage of the object " JEVEU ", the matrix of prepacking is stored in a news sd_matr_asse. New is generated *.VALM* (not-symmetrical) and new *NUME_DDL* respecting the classification of the equations of the initial matrix. The management of the object pre-conditioner XFEM is carried out as one "sd_matr_asse" classical, described in this document.

Notice :

Even if the address of the pre-conditioner is stored in.REFA original matrix, there does not exist chaining of the addresses "sd_matr_asse". Indeed, the address of the pre-conditioner is fixed and one does not need to know the address of matr_asse to test the existence of a pre-conditioner, which simplifies management in memory of these two interdependent objects.
In addition, as pre-conditioner XFEM is effective only in sequential calculation, the reciprocal application of the address of a pre-conditioner towards matr_asse original is not built for the moment.

.REFA (19) = matred/''

Name of the "reduced" matrix obtained by removing the equations of Lagrange (functionality SOLVEUR/ELIM_LAGR=' OUI'). One also speaks about matrix "girl"

.REFA (20) = matmere/''

Name of the matrix "mother" of the matrix if this one were obtained by removing the equations of Lagrange of matmere (functionality SOLVEUR/ELIM_LAGR=' OUI').

DOCU ('.REFA') = / 'ADZE' initial matrix,
/ 'DECT' completely factorized matrix,
/ 'DECP' partially factorized matrix.

3.2 Object .VALM

.VALM XD V R/C NUM ()

This object contains the values of the nonworthless terms of the matrix (with the format Morse). There are 1 (or 2) element (S) in this collection (2 if the matrix is nonsymmetrical).

The 1st element of the collection contains the upper part of the matrix,
2nd the lower part contains.

Remarks :

** For the not-symmetrical matrices, the diagonal is thus stored 2 times;*
One agrees to use only the diagonal stored in the upper part (VALM (1)) ;
** the arrangement of the terms of the matrix in the blocks is explained in documentation sd_stockage [D4.06.07];*
** Even if the matrix is not symmetrical, its profile (mapping of the nonworthless terms) remains symmetrical. Only one profile thus applies to its two parts higher and lower.*

3.3 Object .CONL

.CONL S V R dim=neq

neq is the number of equations of the system

This optional object is present only if there exists at least a ddl of the type 'LAGR' :
V (ieq) = C if ieq corresponds to one DDL named 'LAGR',
1. if not.
C is the coefficient of conditioning of the ddls of Lagrange.

3.4 Object .DIGS

.DIGS S V R/C dim=2*neq

neq is the number of equations of the system

This object is present only if the matrix were factorized by 'LDLT' or 'MULT_FRONT'

DIGS (1: neq) : values of the diagonal of the initial matrix
DIGS (neq+1: 2*neq) : values of the diagonal of the factorized matrix

3.5 Object .PERM

.PERM S V I dim=neq

neq is the number of equations of the system

This object is present only if the matrix is the result of prepacking by 'GCPC' + 'LDLT_INC' of another matrix. It is used to establish the correspondence between the classification of the equations in the initial matrix and its matrix of prepacking. Indeed, the matrix of prepacking was renumbered with the algorithm 'RCMK'(CUTHILL - Mc KEE Transfers)

That is to say: B=FACTORISER (MATR_ASSE=A, METHODE=' GCPC', PRE_COND=' LDLT_INC')

LEAVE (ieq_A) : ieq_B
where ieq_A and ieq_B are the numbers of equations in the matrices With and B.

This object has the same contents as the object . M2LC sd_stockage.

3.6 Object .LIME

List of the names of sd_matr_elem at the origin of sd_matr_asse.
This object exists if sd_matr_asse comes from the assembly from sd_matr_elem or of the combination of sd_matr_asse having to them-even an object.FILE.
This object always does not exist and it is to better avoid making use of it.

3.7 Objects .UALF, .VALF, .WALF

These collections contain the terms of the factorized matrix.

.UALF the matrix factorized contains with LDLT:
symmetrical : collection of size *nblocs*
not-symmetrical : collection of size *2*nblocs*

.VALF the matrix factorized contains with MULT_FRONT of the upper part
.WALF the matrix factorized contains with MULT_FRONT of the lower part

Remarks :

** the arrangement of the terms of the matrix in the blocks is explained in documentation sd_stockage [D4.06.07];*
** For storage LDLT (.UALF), the number of blocks is doubled in the event of a nonsymmetrical matrix. nblocs first correspond to the values associated with the upper part of the matrix while the last nblocs correspond to the lower part of the matrix.*
** Even if the matrix is not symmetrical, its profile (mapping of the nonworthless terms) remains symmetrical. Only one profile thus applies to its two parts higher and lower.*

4 Objects related to the presence of loads eliminated kinematics

4.1.1 Object .CCID

.CCID S V I dim=neq+1

.CCID (ieq) = 1 if the ddl ieq eliminated,
0 if not.
.CCID (neq) = nelim : number of ddls eliminated

4.1.2 Object .CCLL

.CCLL S V I dim=3*nelim

nelim is the number of ddls "eliminated".

```
.CCLL ((i-1) *3+1)     : ieq
.CCLL ((i-1) *3+2)     : i1
.CCLL ((i-1) *3+3)     : i2
```

ieq is the number of the equation corresponding to I^{ème} ddl eliminated,
i1 is the number of terms stored for the equation ieq
i2 is the cumulated number of terms stored for the equations J < I

4.1.3 Object .CCVA

.CCVA S V R/C

The object .CCVA contains the columns of the initial matrix corresponding to the ddls to eliminate (those which are imposed by a "kinematic" load). More precisely, one stores the submatrix of the terms on lines corresponding to DDLs free and columns with of DDLs imposed.

4.1.4 Object .CCII

.CCII S V I

The object .CCII the same structure has as the object .CCVA. It contains the indices of lines of the terms stored in .CCVA

4.1.5 Remarks concerning the "distributed" matrices

When the matrix is distributed,

- The object .CCID is identical on all the processors and its last value (nelim) is the full number (total) of ddls eliminated.
- The object .CCLL is of the same length on all the processors but its contents are different because it contains local numbers of equations.
- Objects .CCII and .CCVA are different lengths on the various processors and their contents are related to the local matrices.