To introduce new boundary conditions kinematics

Summary:

This document presents the two utility routines making it possible to rather easily introduce new types of boundary conditions “kinematics” (i.e of the linear relations between unknown degrees of freedom).

Contents

1 Introduction

What one calls “loading” in Aster (vocabulary of “mechanics”) is what the user defines in the orders AFFE_CHAR_* . One distinguishes the loadings in general of “forces” [D5.03.01] and the loadings in “displacements” (or “kinematics”).

This document explains how to introduce new loadings kinematics.

2 What a linear relation?

This expression indicates a linear constraint on the degrees of freedom system to be studied:

- degrees of freedom of the size TEMP_R for the thermal phenomenon,
- degrees of freedom of the sizes DEPL_R or DEPL_C for the mechanical phenomenon,
- degrees of freedom of the size PRES_C for the acoustic phenomenon.

The coefficients of this linear relation are real constants (or complexes), the second member can be real, complex or of type “ function ” (K8) .

A linear relation can be written:

$$\alpha_1 ddl_1 + \alpha_2 ddl_2 + ... + \alpha_n ddl_n = \alpha_0$$

where

$\alpha_i \in \mathbb{R}$ (or $\mathbb{C}$) ($i = 1, n$)
$\alpha_0 \in \mathbb{R}$ (or $\mathbb{C}$) (or function)

Degrees of freedom ddl_i are degrees of freedom carried by one or more different nodes.

Linear examples of relations:

$DX(N1) = 0.$	blocking of the component DX node $N1$
$TEMP(N3) = 100.$	temperature imposed on $100.$ for the node $N3$
$DY(N1) - DY(N2) = 0.$	nodes $N1$ and $N2$ have same displacement DY
$\cos\alpha\, DX(N1) + \sin\alpha\, DY(N1) = 0.$	the node $N1$ east compels to move on the line perpendicular to the vector ($\cos\alpha$, $\sin\alpha$) (in 2D).

3 How does one introduce linear relations into a modeling?

The linear relations which one defined in the paragraph §2:

- force the solution which one seeks,
- they belong to what one in general calls the “boundary conditions”,
- in *Code_Aster* they are one of the components of the loads (types char_acou , char_ther , char_meca).

These linear relations are thus introduced by the user via the orders AFFE_CHAR_MECA (_F) , AFFE_CHAR_THER (_F) , AFFE_CHAR_ACOU , or AFFE_CHAR_CINE .

These linear relations can be “dealt” with two ways:

- one eliminates an unknown factor for each linear relation: method of elimination [D3.03.01],
- one “dualise” the relation by adding 2 additional unknown factors to him: parameters of Lagrange [R3.03.01].

In *Code_Aster* , the method of elimination is used for the relations resulting from the order AFFE_CHAR_CINE . One will speak in this case about linear relations “kinematics”, although this term is not very judicious. One limits oneself then to relations of the type:

```
DDL = cste
```

Other relations resulting from the orders AFFE_CHAR_MECA , AFFE_CHAR_THER and AFFE_CHAR_ACOU are always dualisées.

Examples of keywords factor generating of the linear relations:

Orders	Keywords Factor	treatment
AFFE_CHAR_CINE	MECA_IMPO	elimination
AFFE_CHAR_MECA_F	LIAISON_OBLIQUE	dualisation
AFFE_CHAR_THER	TEMP_IMPO	dualisation
AFFE_CHAR_MECA	LIAISON_DDL	dualisation
AFFE_CHAR_MECA	LIAISON_SOLIDE	dualisation

The order AFFE_CHAR_CINE allows to introduce all the simple linear relations easily (DDL = cste) that one can define.

On the other hand, although in theory (thanks to the keyword LIAISON_DDL), one can introduce any linear relation, the number of coefficients to be calculated can become very tall. To think for example of the linear relations which it should be written to say that 4 nodes are interdependent (connected by an indeformable solid).

The many keywords making it possible to the user to define these linear relations are there to facilitate work to him:

LIAISON_OBLIQUE	for supports slipping into an oblique reference mark
TEMP_IMPO	to impose a temperature
LIAISON_GROUP	to connect nodes two to two
…	
and LIAISON_DDL	for the other cases…

This a large number of keyword (which will be able to only grow) requires to give itself tools software allowing:

- not to duplicate a code unnecessarily,
- to facilitate the introduction of new keywords into the orders AFFE_CHAR_MECA , AFFE_CHAR_ACOU and AFFE_CHAR_THER .

It is of these tools about which we will speak in the following paragraphs.

4 To introduce a new keyword of type “linear relation”

We give in this paragraph a groundwork for the writing of a routine “carrying out” a keyword of the order AFFE_CHAR_MECA (or _THER or _ACOU), this keyword factor allowing the user to define linear relations.

Are:

- MFAC the keyword factor
- CAMFAC the name of the routine corresponding to him

The goal of the routine CAMFAC is “to scan” the data of the user behind the keyword MFAC , to translate these data into linear relations and to store these relations in the load (here of type char_meca) that the user is defining.

For that, one has two utility routines:

AFRELA	to assign a linear relation to one SD of type LISTE_RELA (list of linear relations)
AFRLCH	“to add” one SD LISTE_RELA with one SD LOAD

These routines force to pass by one SD intermediary (temporary) of type LISTE_RELA . That weighs down a little the programming but has the following advantages:

- performance profits, because the routine AFRLCH is expensive in CPU ,
- a great flexibility to carry out the principle of overload (See the § 6).

The groundwork of the routine CAMFAC is thus the following:

```
        SUBROUTINE CAMFAC (CH)
        CHARACTER * (*) CH
C  in  jxvar CH:       SD CHAR_MECA to be enriched
C  goal:        to enrich the CH load by the definite linear relations
C               under the keyword factor MFAC

buckle on the linear relations

• acquisition of the coefficients of the linear relation:
(routines GETVXX),
• addition of the linear relation with SD LISTE_RELA
Cal AFRELA (… , '&&CAMFAC.LISTE_RELA')

fine buckles

• addition of SD LIST_RELA to the LOAD: CH
Cal AFLRCH ('&&CAMFAC.LISTE_RELA', CH)

END
```

Note:

SD LISTE_RELA *(temporary) is specific to the routine* CAMFAC *, its name respects the convention of the names of objects of work: '*&&nom_routine.XXXX' *,*
The principle of overload (cf [U2.01.00 §3.7]) thus relates to only the occurrences of the keyword MFAC *,*
This SD is destroyed at the time of the call to AFLRCH *.*

5 Routines AFRELA and AFLRCH

5.1 The routine AFRELA

```
      SUBROUTINE AFRELA (COEFR, COEFC, DDL, NODE, NDIM, DIRECT,
     +          NBTERM, BETAR, BETAC, BETAF, TYPCOE, TYPVAL, LISREL)
C----------------------------------------------------------------------
C DRANK: ASSIGNMENT OF ONE  RELATION BETWEEN DDLS WITH A SD LISTE_RELA
C       (IF OBJECT LISREL DOES NOT EXIST, IT IS CREATES)
C
C----------------------------------------------------------------------
C COEFR (NBTERM) - IN - R   - : TABLE OF THE COEFFICIENTS OF THE RELATION
C                             THE COEFFICIENTS ARE REAL
C----------------------------------------------------------------------
C COEFC (NBTERM) - IN - C   - : TABLE OF THE COEFFICIENTS OF THE RELATION
C                             THE COEFFICIENTS ARE COMPLEX
C----------------------------------------------------------------------
C DDL (NBTERM)   - IN - K8  - : TABLE OF THE DDL OF THE RELATION
C----------------------------------------------------------------------
C NODE (NBTERM) - IN - K8  - : TABLE OF THE NODES OF THE RELATION
C----------------------------------------------------------------------
C NDIM (NBTERM)  - IN - I   - : DIMENSION OF THE PROBLEM (0, 2 OR 3)
C                             IF = 0 STEP OF CHANGE OF REFERENCE MARK
C                             THE RELATION IS GIVEN IN THE BASE
C                             TOTAL
C----------------------------------------------------------------------
C DIRECT (3, NBTERM) - IN - R -: TABLE OF THE RELATIVE VECTORS WITH EACH
C                             TERM DEFINING THE DIRECTION OF
C                             COMPONENT WHICH ONE WANTS TO FORCE
C----------------------------------------------------------------------
C NBTERM        - IN - I   - : MANY TERMS OF THE RELATION
C----------------------------------------------------------------------
C BETAR         - IN - R   - : ACTUAL VALUE OF THE SECOND MEMBER
C----------------------------------------------------------------------
C BETAC         - IN - C   - : VALUE COMPLEXES OF THE SECOND MEMBER
C----------------------------------------------------------------------
C BETAF         - IN - K8  - : VALUE FUNCTION OF THE SECOND MEMBER
C----------------------------------------------------------------------
C TYPCOE        - IN - K4  - : TYPE OF THE COEFFICIENTS OF THE RELATION:
C                             = 'REAL' OR 'COMP'
C----------------------------------------------------------------------
C TYPVAL        - IN - K4  - : TYPE OF THE SECOND MEMBER
C                             = 'REAL' OR 'COMP' OR 'FONC'
C----------------------------------------------------------------------
C LISREL        - IN - K19 -: NAME OF SD LISTE_RELA
C               - JXVAR    -
C----------------------------------------------------------------------
```

Two cases are to be considered:

- the degrees of freedom to be connected are given in the absolute reference mark: DX , DY ,...
- some degrees of freedom to connect are given in a local reference mark.

Case A (all in the absolute reference mark):

- NBTERM is the number of degrees of freedom connected by the relation.
- NDIM is a vector filled of 0

- DIRECT is useless.

Example 1:

one wants to impose: $3.\times DX(N1)+2.\times DY(N2)-4.\mathrm{DRZ}(N1)=F$ (function)

```
NBTERM  = 3
TYPCOE  = 'REAL'
TYPVAL  = 'FONC'
COEFR   = (3.  , 2. , -4. )
NDIM    = (0   , 0  ,  0  )
DDL     = ('DX', 'DY', 'DRZ')
NODE   = ('N1', 'N2', 'N1')
BETAF   = 'F'
```

Case B (local reference mark):

For each node implied in the relation, one can give a local reference mark in which the relation is simpler (the normal on a surface for example).

Example 2:

that is to say n , an unit vector of components (nx, ny, nz) .
It is wanted that following displacement n with the node $N3$ that is to say no one.

```
NBTERM  = 1
TYPCOE  = 'REAL'
TYPVAL  = 'REAL'
COEFR   = (1.)
NDIM    = (3)
DIRECT  = (nx, ny, nz)
DDL     = ('DEPL')
NODE   = ('N1')
BETAR   = 0.
```

Note:

NBTERM is not the number of terms of the final relation here (: 3).
When one employs (for a "term") the possibility of a local reference mark NDIM/= 0 the name of the degree of freedom must be conventionally ' DEPL ' or 'ROTA '

Example 3:
RC

are	n1 : an unit vector of components (n1x , n1y , n1z)
	N2 : an unit vector of components (n2x , n2y , n2z)

following data:

```
NBTERM  = 3
TYPCOE  = 'REAL'
TYPVAL  = 'REAL'
COEFR   = (4. , 2. , - 3.)
NDIM    = (3,0,3)
DIRECT  = (n1x, n1y, n1z, rbid, rbid, rbid, n2x, n2y, n2z)
DDL     = ('DEPL', 'DX', 'ROTA')
NODE   = ('N1', 'N3', 'N2')
BETAR   = 5.
```

describe the relation in the 7 terms:

```
4.* (n1x*DX (N1) +n1y*DY (N1) +n1z*DZ (N1))
+  2.*DX (N3)
+ -3.* (n2x*DX (N2) +n2y*DY (N2) +n2z*DZ (N2))   = 5.
```

5.2 The routine AFLRCH

```
  SUBROUTINE AFLRCH (LISREL, LOAD)
C ----------------------------------------------------
C  ADDITION OF A LISTE_RELA IN A LOAD
C
C  THE IDENTICAL RELATIONS WITHIN LISTE_RELA ARE
C  ELIMINEES. THE PRINCIPLE OF OVERLOAD IS BRACKET:
C  IT IS THE LAST SECOND MEMBER WHO IS PRESERVE.
C ----------------------------------------------------
C  LISREL  IN/JXVAR - K19   - : NAME OF SD LISTE_RELA
C                               THE LISTE_RELA IS DESTROYED
C                               AT THE END OF THE ROUTINE
C ----------------------------------------------------
C  LOAD  IN/JXVAR  - K8   - : NAME OF THE SD CHARGES
C                               THE LOAD IS ENRICHED
C ----------------------------------------------------
```

6 Principle of overload

It can happen that the user defines several times the same linear relation (except for a multiplying coefficient).

Example:

```
3.DX (N1) -1.DY (N2) = 4.
6.DX (N1) -2.DY (N2) = 8.
3.DX (N1) -1.DY (N2) = 5.
```

Here, the first 2 equations are identical. Third is contradictory with the preceding ones (because of second member).

If two equations of a linear system to solve have same the 1 1st member, one cannot reverse the matrix, because the equations are not independent. It is thus necessary to eliminate all the equations which are multiple from/to each other.

One wants to be able to apply the principle of “overload” [U2.01.00 §3.7]: it is thus the last second member who is preserved.

This elimination of the “redundant” relations is made at the time or one adds LISTE_RELA with LOAD (routine AFLRCH). One eliminates the doubled blooms from LISTE_RELA , the eliminated relations are printed, then one adds the relations preserved at LOAD .

If one keeps the diagram advised here with the § 4: only one LISTE_RELA by keyword factor, the principle of overload is thus naturally applied for each keyword. The last occurrences take precedence over the first.

If one wanted (it today is not wanted!) an overload between various keywords (for example: DDL_IMPO bonus on FACE_IMPO), it would be enough that these 2 keywords are associated has the same one LISTE_RELA:

```
CAL FACIMPO (CH, LISREL)
CAL DDLIMPO (CH, LISREL)
CAL AFLRCH  (LISREL, CH)
```

7 Example:

To illustrate the use of the two routines presented higher, one will be able to consult the source of the routine caliai.f.

This routine treats the keyword LIAISON_DDL orders:

- AFFE_CHAR_MECA (_F)
- AFFE_CHAR_THER (_F)