Introduire de nouvelles conditions aux limites cinématiques

Résumé :

Ce document présente les deux routines utilitaires permettant d'introduire assez facilement de nouveaux types de conditions aux limites "cinématiques" (i.e des relations linéaires entre degrés de libertés inconnus).

Table des Matières

1 Introduction

Ce que l'on appelle "chargement" dans Aster (vocabulaire de la "mécanique") est ce que l'utilisateur définit dans les commandes AFFE_CHAR_* . On distingue en général les chargements en "force" [D5.03.01] et les chargements en "déplacements" (ou "cinématiques").

Ce document explique comment introduire de nouveaux chargements cinématiques.

2 Qu'est-ce qu'une relation linéaire ?

Cette expression désigne une contrainte linéaire sur les degrés de liberté du système à étudier :

- les degrés de liberté de la grandeur TEMP_R pour le phénomène thermique,
- les degrés de liberté des grandeurs DEPL_R ou DEPL_C pour le phénomène mécanique,
- les degrés de liberté de la grandeur PRES_C pour le phénomène acoustique.

Les coefficients de cette relation linéaire sont des constantes réelles (ou complexes), le second membre peut être réel, complexe ou de type " fonction " (K8) .

Une relation linéaire peut s'écrire :

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

où

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

Les degrés de liberté ddl_i sont des degrés de liberté portés par un ou plusieurs nœuds différents.

Exemples de relations linéaires :

$DX(N1) = 0.$	blocage de la composante DX du nœud $N1$
$TEMP(N3) = 100.$	température imposée à $100.$ pour le nœud $N3$
$DY(N1) - DY(N2) = 0.$	les nœuds $N1$ et $N2$ ont le même déplacement DY
$\cos\alpha\, DX(N1) + \sin\alpha\, DY(N1) = 0.$	le nœud $N1$ est astreint à se déplacer sur la droite perpendiculaire au vecteur ($\cos\alpha$, $\sin\alpha$) (en 2D).

3 Comment introduit-on des relations linéaires dans une modélisation ?

Les relations linéaires que l'on a définies dans le paragraphe §2 :

- contraignent la solution que l'on cherche,
- elles font partie de ce que l'on appelle en général les "conditions aux limites",
- dans *Code_Aster* elles sont une des composantes des charges (types char_acou , char_ther , char_meca).

Ces relations linéaires sont donc introduites par l'utilisateur via les commandes AFFE_CHAR_MECA (_F) , AFFE_CHAR_THER (_F) , AFFE_CHAR_ACOU , ou AFFE_CHAR_CINE .

Ces relations linéaires peuvent être “traitées” de deux façons :

- on élimine une inconnue pour chaque relation linéaire : méthode d’élimination [D3.03.01],
- on “dualise” la relation en lui ajoutant 2 inconnues supplémentaires : les paramètres de Lagrange [R3.03.01].

Dans *Code_Aster* , la méthode d’élimination est utilisée pour les relations issues de la commande AFFE_CHAR_CINE . On parlera dans ce cas de relations linéaires “cinématiques”, bien que ce terme ne soit pas très judicieux. On se limite alors à des relations du type :

```
DDL = cste
```

Les autres relations issues des commandes AFFE_CHAR_MECA , AFFE_CHAR_THER et AFFE_CHAR_ACOU sont toujours dualisées.

Exemples de mots clés facteur engendrant des relations linéaires :

Commandes	Mots Clés Facteur	traitement
AFFE_CHAR_CINE	MECA_IMPO	élimination
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

La commande AFFE_CHAR_CINE permet d’introduire facilement toutes les relations linéaires simples (DDL = cste) que l’on peut définir.

En revanche, bien qu’en théorie (grâce au mot clé LIAISON_DDL), on puisse introduire n’importe quelle relation linéaire, le nombre de coefficients à calculer peut devenir très grand. Penser par exemple aux relations linéaires qu’il faut écrire pour dire que 4 nœuds sont solidaires (reliés par un solide indéformable).

Les nombreux mots clés permettant à l’utilisateur de définir ces relations linéaires sont là pour lui faciliter le travail :

LIAISON_OBLIQUE	pour des appuis glissants dans un repère oblique
TEMP_IMPO	pour imposer une température
LIAISON_GROUP	pour relier des nœuds deux à deux
…	
et LIAISON_DDL	pour les autres cas …

Ce grand nombre de mot clés (qui ne pourra que croître) nécessite de se donner des outils logiciels permettant :

- de ne pas dupliquer inutilement du code,
- de faciliter l’introduction de nouveaux mots clés dans les commandes AFFE_CHAR_MECA , AFFE_CHAR_ACOU et AFFE_CHAR_THER .

C’est de ces outils dont nous parlerons dans les paragraphes suivants.

4 Introduire un nouveau mot-clef de type « relation linéaire »

Nous donnons dans ce paragraphe un canevas pour l'écriture d'une routine "réalisant" un mot clé de la commande AFFE_CHAR_MECA (ou _THER ou _ACOU), ce mot clé facteur permettant à l'utilisateur de définir des relations linéaires.

Soient :

- MFAC le mot clé facteur
- CAMFAC le nom de la routine lui correspondant

Le but de la routine CAMFAC est de "scruter" les données de l'utilisateur derrière le mot clé MFAC , de traduire ces données en relations linéaires et de stocker ces relations dans la charge (ici de type char_meca) que l'utilisateur est en train de définir.

Pour cela, on dispose de deux routines utilitaires :

AFRELA	pour affecter une relation linéaire à une SD de type LISTE_RELA (liste de relations linéaires)
AFRLCH	pour "ajouter" une SD LISTE_RELA à une SD CHARGE

Ces routines imposent de passer par une SD intermédiaire (temporaire) de type LISTE_RELA . Cela alourdit un peu la programmation mais présente les avantages suivants :

- gains de performance, car la routine AFRLCH est coûteuse en CPU ,
- une grande flexibilité pour réaliser le principe de surcharge (Voir le § 6).

Le canevas de la routine CAMFAC est donc le suivant :

```
        SUBROUTINE CAMFAC(ch)
        CHARACTER * (*) ch
C  in  jxvar ch :       SD CHAR_MECA à enrichir
C  but :        enrichir la charge ch des relations linéaires définies
C               sous le mot clé facteur MFAC

boucle sur les relations linéaires

• acquisition des coefficients de la relation linéaire :
(routines GETVXX),
• ajout de la relation linéaire à la SD LISTE_RELA
Call AFRELA (... , '&&CAMFAC.LISTE_RELA')

fin boucle

• ajout de la SD LIST_RELA à la CHARGE : ch
Call AFLRCH ('&&CAMFAC.LIST_RELA', ch)

END
```

Remarques :

La SD LISTE_RELA *(temporaire) est propre à la routine* CAMFAC *, son nom respecte la convention des noms d'objets de travail :* '&&nom_routine.XXXX' ,
Le principe de surcharge (cf. [U2.01.00 §3.7]) ne concerne donc que les occurrences du mot clé MFAC ,
Cette SD est détruite lors de l'appel à AFLRCH .

5 Les routines AFRELA et AFLRCH

5.1 La routine AFRELA

```
      SUBROUTINE AFRELA (COEFR, COEFC, DDL, NOEUD , NDIM, DIRECT,
     +          NBTERM, BETAR, BETAC, BETAF, TYPCOE, TYPVAL, LISREL)
C----------------------------------------------------------------------
C BUT : AFFECTATION D'UNE  RELATION ENTRE DDLS A UNE SD LISTE_RELA
C       (SI L'OBJET LISREL N'EXISTE PAS, IL EST CREE)
C
C----------------------------------------------------------------------
C COEFR(NBTERM) - IN - R   - : TABLEAU DES COEFFICIENTS DE LA RELATION
C                            LES COEFFICIENTS SONT REELS
C----------------------------------------------------------------------
C COEFC(NBTERM) - IN - C   - : TABLEAU DES COEFFICIENTS DE LA RELATION
C                            LES COEFFICIENTS SONT COMPLEXES
C----------------------------------------------------------------------
C DDL(NBTERM)   - IN - K8  - : TABLEAU DES DDL DE LA RELATION
C----------------------------------------------------------------------
C NOEUD(NBTERM) - IN - K8  - : TABLEAU DES NOEUDS DE LA RELATION
C----------------------------------------------------------------------
C NDIM(NBTERM)  - IN - I   - : DIMENSION DU PROBLEME (0, 2 OU 3)
C                            SI = 0 PAS DE CHANGEMENT DE REPERE
C                            LA RELATION EST DONNEE DANS LA BASE
C                            GLOBALE
C----------------------------------------------------------------------
C DIRECT(3,NBTERM)- IN - R - : TABLEAU DES VECTEURS RELATIFS A CHAQUE
C                            TERME DEFINISSANT LA DIRECTION DE LA
C                            COMPOSANTE QUE L'ON VEUT CONTRAINDRE
C----------------------------------------------------------------------
C NBTERM        - IN - I   - : NOMBRE DE TERMES DE LA RELATION
C----------------------------------------------------------------------
C BETAR         - IN - R   - : VALEUR REELLE DU SECOND MEMBRE
C----------------------------------------------------------------------
C BETAC         - IN - C   - : VALEUR COMPLEXE DU SECOND MEMBRE
C----------------------------------------------------------------------
C BETAF         - IN - K8  - : VALEUR FONCTION DU SECOND MEMBRE
C----------------------------------------------------------------------
C TYPCOE        - IN - K4  - : TYPE DES COEFFICIENTS DE LA RELATION :
C                            = 'REEL' OU 'COMP'
C----------------------------------------------------------------------
C TYPVAL        - IN - K4  - : TYPE DU SECOND MEMBRE
C                            = 'REEL' OU 'COMP' OU 'FONC'
C----------------------------------------------------------------------
C LISREL        - IN - K19 - : NOM DE LA SD LISTE_RELA
C               - JXVAR    -
C----------------------------------------------------------------------
```

Deux cas de figure sont à considérer :

- les degrés de liberté à relier sont donnés dans le repère absolu : DX , DY , ...
- certains degrés de liberté à relier sont donnés dans un repère local.

Cas A (tout dans le repère absolu) :

- NBTERM est le nombre de degrés de liberté reliés par la relation.
- NDIM est un vecteur rempli de 0

- DIRECT est inutile.

Exemple 1 :

on veut imposer : $3.\times DX(N1)+2.\times DY(N2)-4.\mathrm{DRZ}(N1)=F$ (fonction)

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

Cas B (repère local) :

Pour chaque nœud impliqué dans la relation, on peut donner un repère local dans lequel la relation est plus simple (la normale à une surface par exemple).

Exemple 2 :

soit n , un vecteur unitaire de composantes (nx, ny, nz) .
On veut que le déplacement suivant n au nœud $N3$ soit nul.

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

Remarques :

NBTERM n'est pas le nombre de termes de la relation finale (ici : 3).
Quand on emploie (pour un "terme") la possibilité d'un repère local NDIM /= 0 le nom du degré de liberté doit être conventionnellement 'DEPL' ou 'ROTA '

Exemple 3 :
RC

soient	n1 : un vecteur unitaire de composantes (n1x , n1y , n1z)
	n2 : un vecteur unitaire de composantes (n2x , n2y , n2z)

les données suivantes :

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

décrivent la relation à 7 termes :

```
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 La routine AFLRCH

```
  SUBROUTINE AFLRCH (LISREL, CHARGE)
C ----------------------------------------------------------
C  AJOUT D'UNE LISTE_RELA DANS UNE CHARGE
C
C  LES RELATIONS IDENTIQUES AU SEIN DE LISTE_RELA SONT
C  ELIMINEES. LE PRINCIPE DE SURCHARGE EST APPLIQUE :
C  C'EST LE DERNIER SECOND MEMBRE QUI EST CONSERVE.
C ----------------------------------------------------------
C  LISREL  IN/JXVAR - K19   - : NOM DE LA SD LISTE_RELA
C                               LA LISTE_RELA EST DETRUITE
C                               A LA FIN DE LA ROUTINE
C ----------------------------------------------------------
C  CHARGE  IN/JXVAR  - K8   - : NOM DE LA SD CHARGE
C                               LA CHARGE EST ENRICHIE
C ----------------------------------------------------------
```

6 Principe de surcharge

Il peut arriver que l'utilisateur définisse plusieurs fois une même relation linéaire (à un coefficient multiplicateur près).

Exemple :

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

Ici, les 2 premières équations sont identiques. La troisième est contradictoire avec les précédentes (à cause du second membre).

Si deux équations d'un système linéaire à résoudre ont le même 1^{ier} membre, on ne peut inverser la matrice, car les équations ne sont pas indépendantes. Il faut donc éliminer toutes les équations qui sont multiples les unes des autres.

On veut pouvoir appliquer le principe de "surcharge" [U2.01.00 §3.7] : c'est donc le dernier second membre qui est conservé.

Cette élimination des relations "redondantes" est faite au moment ou on ajoute la LISTE_RELA à la CHARGE (routine AFLRCH). On élimine les doublons de la LISTE_RELA , on imprime les relations éliminées, puis on ajoute les relations conservées à la CHARGE .

Si on garde le schéma conseillé ici au § 4 : une seule LISTE_RELA par mot clé facteur, le principe de surcharge est donc naturellement appliqué pour chaque mot clé. Les dernières occurrences priment sur les premières.

Si l'on voulait (on ne le veut pas aujourd'hui !) une surcharge entre différents mots clés (par exemple : DDL_IMPO prime sur FACE_IMPO), il suffirait que ces 2 mots clés soient associés a la même LISTE_RELA :

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

7 Exemple :

Pour illustrer l'usage des deux routines présentées plus haut, on pourra consulter le source de la routine caliai.f.

Cette routine traite le mot clé LIAISON_DDL des commandes :

- AFFE_CHAR_MECA(_F)
- AFFE_CHAR_THER(_F)