Introduire un nouveau calcul élémentaire

Résumé :

Ce document décrit ce qu'il faut faire pour introduire un nouveau calcul élémentaire dans *Code_Aster* .

En quelques mots, il faut :

- introduire un petit bloc de texte dans le catalogue d'un `type_element`
- écrire une nouvelle routine fortran de nom `te00ij.f` où `00ij` est un nombre à 4 chiffres

Table des Matières

1 Introduction....3
2 Modification du catalogue du type_element THER_PENTA15....4
2.1 Trouver le nom du fichier à modifier....4
2.2 Modifier le fichier gener_th3d_3.cata....5
2.2.1 Champ de sortie....6
2.2.2 Champs d'entrée....7
2.2.2.1 DDL_THER....8
2.2.2.2 NGEOMER....8
2.2.2.3 CMATERC....9
2.2.2.4 CCAMASS....9
2.2.2.5 CTEMPSR....9
3 Écrire (ou modifier) la routine fortran te0062.f....9
3.1 Arguments de la routine....10
3.2 Présentation de quelques utilitaires utilisés dans le te0062.f....10
3.2.1 Routine JEVECH....10
3.2.2 Routine ELREFE_INFO....11
3.2.3 Routine DFDM3D....11
3.2.4 Routine RCVALA....11
3.3 Routine TE0062....12
4 Détails non utilisés dans l'exemple choisi....13
4.1 Description de l'entête d'un type_element....13
4.2 Modes Locaux ELNO__ DIFF__....13
4.3 Conventions de noms pour les modes locaux....13
4.4 À compléter …Noms réservés pour certains modes locaux (ddl_meca, ddl_ther, ddl_acou) ….13
4.5 Champs locaux de type vecteur élémentaire ou matrice élémentaire....13
4.6 Champs facultatifs, routine tecach.f....13
4.7 Calcul élémentaire non-disponible : « -1 »....14
4.8 Famille de points de Gauss "MATER"....14

1 Introduction

Pour Code_Aster, un calcul élémentaire correspond à un couple (type d'élément fini, option de calcul). Exemples de types d'élément finis (type_element) :

- MEDKTR3 : élément DKT triangulaire à 3 nœuds
- THER_PENTA15 : élément de thermique pentaèdre à 15 nœuds

Exemples d'options de calcul (option) :

- RIGI_MECA : calcul de la rigidité (comportement élastique)
- FLUX_ELGA : calcul du flux thermique connaissant la température aux nœuds

Dans le reste de ce document, l'exemple qui nous servira de fil conducteur sera celui du calcul du flux thermique aux points d'intégration (FLUX_ELGA) des éléments PENTA15 de la modélisation '3D ' du phénomène ' THERMIQUE ' (type_element = THER_PENTA15).

On va supposer que ce calcul élémentaire n'existe pas encore mais que l'option FLUX_ELGA existe déjà (pour d'autres éléments finis) et que le type_element THER_PENTA15 existe également (il sait déjà calculer d'autres options). Dans ce document, nous essaierons de répondre aux questions :

- Que faut-il faire pour réaliser ce nouveau calcul élémentaire ?
- Quels fichiers sources faut-il modifier ou ajouter ?

D'autres questions relatives aux éléments finis sont traitées dans d'autres documents :

- [D5.02.03] Comment introduire une nouvelle option de calcul élémentaire ? (par exemple un nouveau post-traitement)
- [D5.02.04] Comment introduire une nouvelle famille d'éléments finis (modelisation) ?
- [D5.02.01] Comment introduire une nouvelle grandeur ou de nouvelles composantes dans une grandeur existante ?
- [D5.02.02] Comment introduire un nouveau type de maille (type_maille) ou un nouvel élément de référence (ELREFE) ?

Nous avons déjà dit dans le résumé que l'introduction d'un nouveau calcul élémentaire dans Code_Aster nécessitait 2 actions :

- l'ajout d'un bloc de texte dans le catalogue du type_element (ici THER_PENTA15)
- l'ajout (ou la modification) d'une routine fortran de nom te00ij.f

Nous allons détailler successivement ces deux actions.

2 Modification du catalogue du type_element THER_PENTA15

2.1 Trouver le nom du fichier à modifier

La première difficulté à résoudre est de trouver le nom du fichier catalogue à modifier. Remarquons déjà que le nom du type_element qui nous concerne (THER_PENTA15) ne nous est pas forcément familier. Nous pouvons le découvrir en utilisant la commande AFFE_MODELE sur un maillage contenant des PENTA15 :

```
MOTH = AFFE_MODELE(  MAILLAGE = MAIL,
                     AFFE=_F( TOUT = 'OUI', MODELISATION = '3D',
                                PHENOMENE = 'THERMIQUE'))
```

Dans le fichier .mess nous pouvons alors voir :

```
 SUR LES          132 MAILLES DU MAILLAGE MAIL
    ON A DEMANDE L'AFFECTATION DE          132
    ON A PU EN AFFECTER                    124

    MODELISATION      ELEMENT FINI      TYPE MAILLE          NOMBRE
    3D                THER_PENTA15      PENTA15                40
    3D                THER_FACE6        TRIA6                   4
    3D                THER_FACE8        QUAD8                  80
```

Nous voyons que la modélisation appelée ' 3D ' appliquée sur des PENTA15 conduit à l'affectation d'éléments finis de type THER_PENTA15 . C'est le nom que nous cherchions.

Nous aurions pu aussi trouver ce nom en consultant le catalogue :

```
.../NEW11/catalo/compelem/phenomene_modelisation__.cata
...
   PHENOMENE__  THERMIQUE       CODE__  'TH'
 ...
       MODELISATION__ '3D'               DIM__ 3 3   CODE__ '3D_'
              ATTRIBUT__  DIM_TOPO_MAILLE=X3
              MAILLE__ HEXA8     ELEMENT__ THER_HEXA8
              MAILLE__ PENTA6    ELEMENT__ THER_PENTA6
              MAILLE__ TETRA4    ELEMENT__ THER_TETRA4
              MAILLE__ PYRAM5    ELEMENT__ THER_PYRAM5
              MAILLE__ QUAD4     ELEMENT__ THER_FACE4
              MAILLE__ TRIA3     ELEMENT__ THER_FACE3
              MAILLE__ HEXA27    ELEMENT__ THER_HEXA27
              MAILLE__ HEXA20    ELEMENT__ THER_HEXA20
              MAILLE__ PENTA15   ELEMENT__ THER_PENTA15
              MAILLE__ TETRA10   ELEMENT__ THER_TETRA10
              MAILLE__ PYRAM13   ELEMENT__ THER_PYRAM13
              MAILLE__ QUAD9     ELEMENT__ THER_FACE9
              MAILLE__ QUAD8     ELEMENT__ THER_FACE8
              MAILLE__ TRIA6     ELEMENT__ THER_FACE6
```

Ce catalogue donne le nom de tous les type_element associés aux différents éléments finis de la modélisation ' 3D '. Pour le type de maille PENTA15 , le nom du type_element est bien THER_PENTA15 .

Une fois connu le nom THER_PENTA15 , pour trouver le nom du catalogue concernant cet élément fini, il faut faire un grep dans les fichiers de catalo/typelem/* :
grep -l THER_PENTA15 /aster/NEW11/catalo/typelem/* . Ce que devrait donner :

/ aster/NEW11/typel em/ gener_th3d_3.cata

C'est le nom du fichier que nous devons modifier.

Si nous éditons ce fichier, nous voyons qu'il contient un certain nombre de rubriques d'entête :

```
ENTETE__ ELEMENT__ THER_HEXA20      MAILLE__ HEXA20
   ELREFE__   H20       GAUSS__   RIGI=FPG27  MASS=FPG27  GANO=FPG8
   ELREFE__   QU8       GAUSS__   RIGI=FPG9   MASS=FPG9
ENTETE__ ELEMENT__ THER_HEXA27      MAILLE__ HEXA27
   ELREFE__   H27       GAUSS__   RIGI=FPG27  MASS=FPG27  GANO=FPG8
   ELREFE__   QU9       GAUSS__   RIGI=FPG9   MASS=FPG9
ENTETE__ ELEMENT__ THER_HEXA8       MAILLE__ HEXA8
   ELREFE__   HE8       GAUSS__   RIGI=FPG8   MASS=FPG8   GANO=FPG8
   ELREFE__   QU4       GAUSS__   RIGI=FPG4   MASS=FPG4
ENTETE__ ELEMENT__ THER_PENTA15     MAILLE__ PENTA15
   ELREFE__   P15       GAUSS__   RIGI=FPG21  MASS=FPG21  GANO=FPG21
   ELREFE__   QU8       GAUSS__   RIGI=FPG9   MASS=FPG9
   ELREFE__   TR6       GAUSS__   RIGI=FPG6   MASS=FPG6
```

L'une de ces entêtes concerne notre élément (THER_PENTA15), mais ce fichier catalogue concerne aussi tous les autres type_element des autres entêtes.

Ce qui veut dire que notre nouveau calcul élémentaire sera (par défaut) accessible à tous les éléments finis décrits dans le fichier catalogue.

En général, c'est ce que l'on souhaite car si on sait faire un calcul sur un PENTA15 , on sait également le faire pour le TETRA4 , le PENTA6 , ...

Dans la suite du document, on continuera à se référer à l'élément THER_PENTA15 , mais en réalité, notre développement concernera tous les éléments volumiques de la modélisation ' 3D ' (THER_TETRA4 , THER_TETRA10 , ..., THER_HEXA27).

Si l'on ne souhaitait pas rendre ce calcul élémentaire disponible pour les HEXA8 par exemple, on modifierait l'entête de l ' HEXA8 :

```
ENTETE__ ELEMENT__ THER_HEXA8       MAILLE__ HEXA8
   ELREFE__   HE8       GAUSS__   RIGI=FPG8   MASS=FPG8   GANO=FPG8
   ELREFE__   QU4       GAUSS__   RIGI=FPG4   MASS=FPG4
   OPTION__    FLUX_ELGA        -1
```

le nombre "-1" en regard de l'option FLUX_ELGA indique que ce calcul élémentaire n'est pas disponible pour les HEXA8 . Une erreur fatale sera émise si l'utilisateur tente de l'utiliser.

2.2 Modifier le fichier gener_th3d_3.cata

Le bloc de texte à ajouter est :

```
FLUX_ELGA   62    IN__   CCAMASS  PCAMASS  NGEOMER  PGEOMER
                         CMATERC  PMATERC  DDL_THER PTEMPER
                         CTEMPSR  PTEMPSR
                   OUT__  EFLUXPG  PFLUX_R
```

Dans ce bloc de texte, on peut reconnaitre différents items :

- le nom de l'option que l'on veut "réaliser" (FLUX_ELGA)
- Le nombre "62" qui est le numéro de la routine te00ij.f associée au calcul élémentaire (ici : te0062.f)

- le bloc des champs d'entrée du calcul élémentaire (ce qui suit le mot clé IN__)
- le bloc des champs de sortie du calcul élémentaire (ce qui suit le mot clé OUT__)

Remarque :

un champ est soit " IN " soit " OUT ", il ne peut pas être " INOUT ". Les blocs des champs " IN " et " OUT " sont décrits par des listes de couples (mode_local , parametre).
Ici, il y a 5 champs " IN " et un seul champ " OUT ".

2.2.1 Champ de sortie

Commençons par le champ "OUT" (but du calcul élémentaire) :

PFLUX_R (le paramètre) est le nom donné au champ de flux résultat pour l'option FLUX_ELGA . Ce nom a été choisi lorsque le catalogue de l'option (/aster/NEW11/catalo/options/FLUX_ELGA.cata) a été introduit dans le code. Ce nom est utilisé dans tous les catalogues des éléments finis qui calculent l'option FLUX_ELGA .

EFLUXPG (le mode_local) est un nom local au catalogue que l'on modifie. Il permet de décrire la structure du champ local de flux pour les éléments THER_PENTA15 . Nous y reviendrons plus tard.

Regardons ce que contient le catalogue de l'option FLUX_ELGA concernant son champ de sortie :

```
FLUX_ELGA
  <<  FLUX_ELGA : CALCUL DU FLUX AUX POINTS DE GAUSS >>
OPTION__
   IN__
        PGEOMER  GEOM_R  << COORDONNEES DES NOEUDS >>
...

   OUT__
        PFLUX_R    FLUX_R    ELGA__
```

On voit que le champ paramètre PFLUX_R est un champ de la grandeur FLUX_R et que c'est un champ par éléments aux points de Gauss (ELGA__)

C'est le "cahier des charges" du développement que nous devons réaliser : nous devons calculer un champ de FLUX_R sur les points de Gauss du THER_PENTA15 . Le choix du type de champ (ici ELGA) est imposé par l'option. En revanche les composantes de la grandeur FLUX_R que l'on va utiliser sont au choix de l'élément THER_PENTA15 . C'est l'objet du mode_local EFLUXPG .

Ce mode_local est décrit dans un bloc de texte du catalogue (dans la rubrique MODE_LOCAL__) :

```
MODE_LOCAL__
    ...
    EFLUXPG  = FLUX_R   ELGA__ RIGI      (FLUX    FLUY    FLUZ)
```

On y voit que ce mode_local concerne bien la grandeur FLUX_R et qu'il décrit bien la structure d'un champ " ELGA " (c'est dans le cahier des charges imposé par l'option). Le choix des composantes FLUX , FLUY et FLUZ n'a rien d'étonnant puisque l'élément est 3D .

Le nom des composantes que l'on peut utiliser dans la description d'un mode_local est donné dans le catalogue des grandeurs (fichier /aster/NEW11/catalo/compelem/grandeur_simple__.cata). On peut y lire concernant la grandeur FLUX_R :

```
FLUX_R   = R    FLUX     FLUY     FLUZ   FLUX_SUP   FLUY_SUP   FLUZ_SUP
                                         FLUX_INF   FLUY_INF   FLUZ_INF
```

Normalement, dans ce catalogue, une ligne de commentaire explique la signification de chacune de ces composantes :

```
<< FLUX_R   Type:R   Flux vectoriel de chaleur en un point matériel du
   domaine continu : PHI = -lambda.gradient(T)
      FLUX : composante suivante OX de PHI
      FLUY : composante suivante OY de PHI
      FLUZ : composante suivante OZ de PHI
      ...
```

2.2.2 Champs d'entrée

La liste des champs d'entrée a la même structure que celle les champs de sortie : c'est une liste de couples (mode_local , parametre). Les paramètres sont imposés par le catalogue de l'option. Pour FLUX_ELGA on trouve dans le catalogue :

```
FLUX_ELGA
       <<  FLUX_ELGA : CALCUL DU FLUX AUX POINTS DE GAUSS >>
OPTION__
   IN__
        PGEOMER  GEOM_R
       <<  PGEOMER : COORDONNEES DES NOEUDS >>
        PMATERC  ADRSJEVE
       <<  PMATERC : CHAMP DE MATERIAU >>
        PCAORIE  CAORIE
       <<  PCAORIE : ORIENTATION LOCALE D'UN ELEMENT DE POUTRE OU DE TUYAU,
           ISSUE DE AFFE_CARA_ELEM MOT CLE ORIENTATION >>
        PCADISK  CADISK
       <<  PCADISK : CARACTERISTIQUE DE DISCRET,
           NECESSITE DE FOURNIR LE CONCEPT PRODUIT PAR AFFE_CARA_ELEM  >>
        PCAGNPO  CAGNPO
       <<  PCAGNPO : CARACTERISTIQUES GEOMETRIQUES D'UNE SECTION DE POUTRE,
           NECESSITE DE FOURNIR LE CONCEPT PRODUIT PAR AFFE_CARA_ELEM  >>
        PCACOQU  CACOQU
       <<  PCACOQU : CARACTERISTIQUE DE COQUE,
           NECESSITE DE FOURNIR LE CONCEPT PRODUIT PAR AFFE_CARA_ELEM  >>
        PCAMASS  CAMASS
       <<  PCAMASS : CARACTERISTIQUE DE MASSIF,
           NECESSITE DE FOURNIR LE CONCEPT PRODUIT PAR AFFE_CARA_ELEM  >>
        PTEMPER  TEMP_R
       <<  PTEMPER : TEMPERATURES INSTANT ACTUEL >>
        PTEMPSR  INST_R
       <<  PTEMPSR :  INSTANT ACTUEL >>
        PNUMCOR  NUMC_I
       <<  PNUMCOR : NIVEAU ET COUCHE D'UN MATERIAU MULTICOUCHE >>
        PHARMON  HARMON
       <<  PHARMON : NUMERO D'HARMONIQUE DE FOURIER >>
```

Remarque :

Les textes écrits entre « ...» sont des commentaires qui sont imprimés dans certains messages d'erreur afin d'aider l'utilisateur à comprendre le contexte de son erreur. Il ne faut pas hésiter à les améliorer ou à les enrichir. Les 11 paramètres d'entrée de l'option FLUX_ELGA sont donc :

```
PGEOMER  GEOM_R
PMATERC  ADRSJEVE
PCAORIE  CAORIE
PCADISK  CADISK
PCAGNPO  CAGNPO
PCACOQU  CACOQU
PCAMASS  CAMASS
```

```
PTEMPER  TEMP_R
PTEMPSR  INST_R
PNUMCOR  NUMC_I
PHARMON  HARMON
```

Il faut choisir dans cette liste les paramètres qui seront utiles au type_element THER_PENTA15 .
Après réflexion, on retient les 5 paramètres suivants :

- PTEMPER : c'est le champ de température dont il va falloir calculer le gradient
- PGEOMER : le champ de géométrie de l'élément (ses coordonnées) est nécessaire pour calculer le gradient.
- PMATERC : le champ de matériau est nécessaire pour pouvoir connaitre la conductivité thermique (lambda)
- PCAMASS : c'est le champ contenant l'éventuel repère local des éléments 3D . Il est nécessaire si le matériau n'est pas isotrope : les caractéristiques anisotropes sont données dans un repère local.
- PTEMPSR : ce champ sert à transmettre l'instant du calcul. Il est nécessaire ici car, en thermique, lorsque les coefficients matériau sont des fonctions (DEFI_MATERIAU / THER_FO par exemple), ces fonctions peuvent dépendre du temps.

Une fois que l'on a retenu les paramètres utiles au type_element , il faut leur attribuer à chacun un mode_local . On va choisir :

```
DDL_THER PTEMPER
NGEOMER  PGEOMER
CCAMASS  PCAMASS
CMATERC  PMATERC
CTEMPSR  PTEMPSR
```

Ces modes locaux doivent être décrits dans la rubrique MODE_LOCAL__ du catalogue :

```
MODE_LOCAL__
    ...
    DDL_THER = TEMP_R   ELNO__ IDEN__   (TEMP      )
    ...
    NGEOMER  = GEOM_R   ELNO__ IDEN__   (X         Y         Z         )
    ...
    CMATERC  = ADRSJEVE ELEM__          (I1        )
    ...
    CCAMASS  = CAMASS   ELEM__      (C         ALPHA     BETA      KAPPA
                                     X         Y         Z)
    ...
    CTEMPSR  = INST_R   ELEM__      (INST      DELTAT    THETA     KHI
                                     R         RHO       )
    ...
```

Commentons ces modes locaux :

2.2.2.1 DDL_THER

Ce mode_local indique que le champ local attendu par l'élément fini est un champ sur les nœuds de l'élément (ELNO__).

Tous les nœuds de l'élément (15 pour un PENTA15) portent les mêmes composantes (IDEN__).
La liste des composantes portées par ces nœuds est réduite à une seule composante de la grandeur TEMP_R : TEMP .

2.2.2.2 NGEOMER

Ce mode_local indique que le champ local attendu par l'élément fini est un champ sur les nœuds de l'élément (ELNO__).

Tous les nœuds portent les mêmes composantes (IDEN__).
La liste des composantes portées par ces nœuds est X , Y , Z car l'élément est 3D .

2.2.2.3 CMATERC

Ce mode_local indique que le champ local attendu par l'élément fini est un champ constant sur l'élément (ELEM__).

La liste des composantes portées par cet élément est I1 .

Remarque :
Le champ de matériau est toujours associé à un mode_local *(en général appelé* CMATERC *)*
défini de la même façon : CMATERC = ADRSJEVE ELEM__ (I1)

Cela traduit le fait qu'on ne peut pas affecter (par exemple) des matériaux différents sur les nœuds d'un élément. Il n'y a qu' un seul matériau par maille.

Par ailleurs, pour des raisons de performance, la structure informatique représentant le matériau au niveau d'un élément est dite "codée" (on parle alors de matériau codé), elle est représentée par un nombre entier (composante I1 de la grandeur ADRSJEVE) qui est une adresse mémoire. On verra plus tard comment ce matériau est utilisé dans les routines utilitaires.

2.2.2.4 CCAMASS

Ce mode_local indique que le champ local attendu par l'élément fini est un champ constant sur l'élément (ELEM__).

Les 7 composantes (C , ALPHA , ...) sont des nombres réels qui permettent de représenter le changement de repère Global → Local

Nous n'en dirons pas plus ici.

2.2.2.5 CTEMPSR

Ce mode_local indique que le champ local attendu par l'élément fini est un champ constant sur l'élément (ELEM__).

On remarque que 6 composantes sont indiquées dans le mode_local : INST , DELTAT , ... Pourtant seule la valeur de l'instant INST nous est utile pour pouvoir évaluer d'éventuelles fonctions du temps.
Les autres composantes ne seront pas utilisées et elles pourraient être retirées du mode_local .

Pourtant, il ne faut pas le faire trop brutalement. En effet, le mode_local CTEMPSR est utilisé par d'autres options qui elles ont besoin de plus d'informations (DELTAT : valeur du pas de temps, ...)

Si on voulait améliorer la lisibilité du catalogue de notre élément, on pourrait dédoubler ce mode_local :

```
CTEMPSR  = INST_R   ELEM__   (INST  DELTAT  THETA  KHI  R  RHO  )
CTEMPS1  = INST_R   ELEM__   (INST)
```

Pour notre option SIEF_ELGA_TEMP , on pourrait alors décrire le champ local d'instant par le couple (CTEMPS1 PTEMPSR)

3 Écrire (ou modifier) la routine fortran te0062.f

Nous avons déjà vu que le catalogue de l'élément THER_PENTA15 indiquait que le calcul de l'option FLUX_ELGA se passera dans la routine fortran te0062.f . L'objet de cette routine est de calculer (pour un élément fini) le champ local de flux sur les points de Gauss de l'élément.
De façon générale, une routine te00ij.f a toujours comme but de calculer des champs " OUT " à partir se ses champs " IN ".

3.1 Arguments de la routine

Toutes les routines te00ij.f ont les mêmes arguments. La raison en est que la routine qui les appelle (te0000.f) est écrite une fois pour toutes et qu'on ne veut pas la changer à chaque fois que l'on ajoute un calcul élémentaire.

Les deux seuls arguments des routines te00ij.f sont des arguments d'entrée :

OPTION est une chaine de caractères contenant le nom de l'option (pour nous : FLUX_ELGA)

NOMTE est une chaine de caractères contenant le nom du type_element (pour nous : THER_PENTA15)

Ces 2 arguments peuvent être utilisés (ou pas).

On utilise l'argument OPTION lorsque l'on traite dans une même routine te00ij.f 2 options (ou plus) qui se ressemblent.

On peut alors écrire des tests comme :

```
...
IF (OPTION.EQ.'FLUX_ELGA') THEN
   ...
ELSE IF (OPTION.EQ.'FLUX_ELNO') THEN
   ...
ENDIF
```

De la même façon, on traite en général tous les éléments d'une même modélisation (TETRA4 , TETRA10 , ...) dans la même routine te00ij.f . L'argument NOMTE permet de distinguer certains traitements. On pourrait par exemple imaginer un petit bloc de code propre aux éléments pyramidaux.

Mais les véritables arguments des routines te00ij.f sont en réalité leurs champs " IN " et leurs champs " OUT ". Ces arguments sont "souterrains" et on y accède via les 2 routines utilitaires JEVECH et TECACH que l'on va présenter dans le paragraphe suivant.

3.2 Présentation de quelques utilitaires utilisés dans le te0062.f

3.2.1 Routine JEVECH

Cette routine permet de récupérer l'adresse mémoire d'un champ local. Par exemple, dans la routine te0062.f , on trouve :

```
CALL JEVECH('PGEOMER','L',IGEOM)
```

Le 1er argument de la routine JEVECH est le nom du paramètre qui nous intéresse.
Le 2ème argument est "documentaire" (il n'est pas utilisé dans le code). Il indique si le parametre est un champ " IN " (accès en lecture L) ou un champ "OUT" (accès en écriture E).
Le 3ème argument (de sortie) est l'adresse de la zone mémoire contenant le champ local.

C'est au programmeur de la routine te0062.f d'assurer la cohérence entre l'usage qu'il fait de cette adresse et ce qui est écrit dans le catalogue de l'élément concernant ce paramètre :

Puisque le paramètre PGEOMER est associé à la grandeur GEOM_R et que cette grandeur est de type réel, l'adresse IGEOM est une adresse dans le common JEVEUX ZR

Puisque le mode_local NGEOMER (associé à PGEOMER) est décrit dans le catalogue par NGEOMER = GEOM_R ELNO *IDEN* (X Y Z) , c'est au programmeur de savoir que le champ local représenté en mémoire à l'adresse ZR(IGEOM) est de longueur 3*15 (3 composantes X , Y , Z pour chacun des 15 nœuds).

Plus précisément, le programmeur doit savoir qu'il va trouver dans la mémoire JEVEUX :

```
ZR(IGEOM-1+1) -> 'X' du noeud 1
ZR(IGEOM-1+2) -> 'Y' du noeud 1
ZR(IGEOM-1+3) -> 'Z' du noeud 1
ZR(IGEOM-1+4) -> 'X' du noeud 2
ZR(IGEOM-1+5) -> 'X' du noeud 2
...
ZR(IGEOM-1+45) -> 'Z' du noeud 15
```

3.2.2 Routine ELREFE_INFO

À compléter ...

3.2.3 Routine DFDM3D

À compléter …

3.2.4 Routine RCVALA

À compléter ...

3.3 Routine TE0062

```
      SUBROUTINE TE0062(OPTION,NOMTE)
      IMPLICIT NONE
      CHARACTER*16 OPTION,NOMTE

C --- DEBUT DECLARATIONS NORMALISEES JEVEUX ----------------------------
      INTEGER ZI
      COMMON /IVARJE/ZI(1)
      REAL*8 ZR
      COMMON /RVARJE/ZR(1)
      COMPLEX*16 ZC
      COMMON /CVARJE/ZC(1)
      LOGICAL ZL
      COMMON /LVARJE/ZL(1)
      CHARACTER*8 ZK8
      CHARACTER*16 ZK16
      CHARACTER*24 ZK24
      CHARACTER*32 ZK32
      CHARACTER*80 ZK80
      COMMON /KVARJE/ZK8(1),ZK16(1),ZK24(1),ZK32(1),ZK80(1)
C --- FIN DECLARATIONS NORMALISEES JEVEUX ------------------------------

      INTEGER ICODRE
      CHARACTER*8 NOMRES(1)
      REAL*8 LAMBDA,FLUXX,FLUXY,FLUXZ,DFDX(27),DFDY(27),DFDZ(27),POIDS
      INTEGER JGANO,IPOIDS,IVF,IDFDE,IGEOM,IMATE,NNO,KP,NPG1,I,IFLUX,
     &        ITEMPS,ITEMPE,NDIM,NNOS
C.......................................................................


C     -- récupération d'informations concernant l'élément de référence :
      CALL ELREFE_INFO(FAMI='RIGI',NPG=NPG1,
     &                 JPOIDS=IPOIDS,JVF=IVF,JDFDE=IDFDE)

C     -- récupération des adresses des champs locaux :
      CALL JEVECH('PGEOMER','L',IGEOM)
      CALL JEVECH('PMATERC','L',IMATE)
      CALL JEVECH('PTEMPSR','L',ITEMPS)
      CALL JEVECH('PTEMPER','L',ITEMPE)
      CALL JEVECH('PFLUX_R','E',IFLUX)

C     -- récupération de la conductivité LAMBDA :
      CALL RCVALA(ZI(IMATE),' ','THER',1,'INST',ZR(ITEMPS),1,'LAMBDA',
     &            LAMBDA,ICODRE,1)


C     -- boucle sur les points de Gauss :
      DO 20 KP=1,NPG1
C       -- récupération des dérivées des fonctions de forme
C          (sur l'élément réel) :
        CALL DFDM3D(NNO,KP,IPOIDS,IDFDE,ZR(IGEOM),DFDX,DFDY,DFDZ,POIDS)

C       -- calcul du gradient du champ de température :
        FLUXX=0.0D0
        FLUXY=0.0D0
        FLUXZ=0.0D0

        DO 10 I=1,NNO
          FLUXX=FLUXX+ZR(ITEMPE-1+I)*DFDX(I)
```

```
              FLUXY=FLUXY+ZR(ITEMPE-1+I)*DFDY(I)
              FLUXZ=FLUXZ+ZR(ITEMPE-1+I)*DFDZ(I)
   10      CONTINUE

C          -- calcul du flux et stockage dans le champ résultat:
           ZR(IFLUX+(KP-1)*3)   =-LAMBDA*FLUXX
           ZR(IFLUX+(KP-1)*3+1)=-LAMBDA*FLUXY
           ZR(IFLUX+(KP-1)*3+2)=-LAMBDA*FLUXZ
   20    CONTINUE

         END
```

Commentaires sur cette routine :

Il ne faut pas oublier que cette routine doit pouvoir traiter tous les éléments 3D : TETRA4 , ..., HEXA27 . C'est pourquoi, Les tableaux DFDX , DFDY et DFDZ sont dimensionnés à 27 qui est le "max" du nombre de nœuds d'un élément volumique.

La famille de points de Gauss qui est utilisée ici est ' RIGI '. C'est elle qui est donnée en argument de la routine ELREFE_INFO . Cette famille est cohérente avec le choix fait dans le catalogue :
EFLUXPG = FLUX_R ELGA__ RIGI (FLUX FLUY FLUZ)

La routine RCVALA est utilisée pour récupérer la valeur de la conductivité thermique (LAMBDA). Comme ce paramètre peut être une fonction du temps, on fournit la valeur de l'instant en argument d'entrée de la routine RCVALA .

le calcul du gradient de la température est fait en deux temps. La routine DFDM3D permet de calculer le gradient des fonctions de forme sur l'élément en 1 point de Gauss donné. On peut ensuite utiliser le gradient des fonctions de forme et la valeur de la température sur les nœuds pour calculer le gradient de la température.

Le stockage des flux calculés à l'adresse IFLUX respecte bien la convention de stockage des champs locaux : 3 composantes FLUX , FLUY et FLUZ pour chaque point de Gauss.

4 Détails non utilisés dans l'exemple choisi

4.1 Description de l'entête d'un type_element

À compléter …

4.2 Modes Locaux ELNO__ DIFF__

À compléter …

4.3 Conventions de noms pour les modes locaux

À compléter …

4.4 À compléter …Noms réservés pour certains modes locaux (ddl_meca, ddl_ther, ddl_acou)

À compléter …

4.5 Champs locaux de type vecteur élémentaire ou matrice élémentaire

À compléter …

4.6 Champs facultatifs, routine tecach.f

À compléter …

4.7 Calcul élémentaire non-disponible : « -1 »

À compléter …

4.8 Famille de points de Gauss "MATER"

À compléter …