Manuel d'Utilisation
Fascicule U4.5- : Méthodes de résolution
Document : U4.55.04

Opérateur RESO_GRAD

1 But

Résoudre un système linéaire par la méthode du gradient conjugué pré-contionné. Le préconditionnement s'obtient auparavant par l'opérateur FACT_GRAD [U4.55.03] et est nécessaire pour obtenir une convergence satisfaisante en un minimum d'itérations.

Opérateur parfois réentrant (dans le cas où on reprend une solution déjà obtenue par RESO_GRAD).

Produit une structure de données de type cham_no_*.

2 Syntaxe

```
& S [cham_no_*] = RESO_GRAD

          (   ♦  MATR_ASSE =  A,                  /  [matr_asse_DEPL_R]
                                                  /  [matr_asse_TEMP_R]
                                                  /  [matr_asse_PRES_R]

              ♦  CHAM_NO =    b,                  /  [cham_no_DEPL_R]
                                                  /  [cham_no_TEMP_R]
                                                  /  [cham_no_PRES_R]

              ◊  CHAM_CINE =  vcine,              /  [cham_no_DEPL_R]
                                                  /  [cham_no_TEMP_R]
                                                  /  [cham_no_PRES_R]

              ◊  MATR_FACT =  precond,            /  [matr_asse_DEPL_R]
                                                  /  [matr_asse_TEMP_R]
                                                  /  [matr_asse_PRES_R]

              ◊  REPRISE =    /  'OUI'  ,
                              /  'NON'  ,            [DEFAUT]

              ◊  RESI_RELA =  /  1.E-6  ,            [DEFAUT]
                              /  resi   ,            [R]

              ◊  NMAX_ITER =  /  niter  ,            [I]
                              /  0      ,            [DEFAUT]

              ◊  INFO =    /  1  ,
                           /  2  ,

          )
```

Si CHAM_NO : [cham_no_DEPL_R] alors (*) → DEPL_R
[cham_no_TEMP_R] → TEMP_R
[cham_no_PRES_R] → PRES_R

3 Opérandes

3.1 Opérande MATR_ASSE

♦ `MATR_ASSE = A`

Nom de la matrice assemblée du système à résoudre.

3.2 Opérande CHAM_NO

♦ `CHAM_NO = b`

Nom du `cham_no` second membre du système.

3.3 Opérande CHAM_CINE

◊ `CHAM_CINE = vcine`

Nom du vecteur représentant la "valeur" des conditions limites de type degré de liberté imposé traduites sous forme de chargement cinématique (c'est-à-dire par utilisation de la commande `AFFE_CHAR_CINE(_F)`. Ce `cham_no` provient de l'exécution de l'opérateur `CALC_CHAR_CINE` sur la liste des `CHAR_CINE` (chargement cinématique) associée à la matrice assemblée `A`. Cf. [U4.61.03].

3.4 Opérande MATR_FACT

◊ `MATR_FACT = precond`

Matrice de pré-conditionnement, obtenue par l'opérateur `FACT_GRAD` [U4.55.03]. Le pré-conditionnement est nécessaire pour obtenir une bonne convergence en un minimum d'itérations.

3.5 Opérande REPRISE

◊ `REPRISE =`

Indique si l'on est ou non en reprise d'un calcul précédent qui n'aurait pas convergé suffisamment :

le calcul est initialisé par :

$x^{(0)} = 0$ vecteur nul si `REPRISE = 'NON'`

$x^{(0)} = S$ le `cham_no` solution `'S'` si `REPRISE = 'OUI'` ; dans ce cas le nom du concept produit est précédé d'un &.

La valeur par défaut est `'NON'`.

3.6 Opérande RESI_RELA

◊ `RESI_RELA =`

Critère de convergence de l'algorithme ; c'est un critère relatif sur le résidu :

$$\frac{\|r_m\|}{\|b\|} \leq resi$$

r_m est le résidu à l'itération m

b est le second membre et $\|\cdot\|$ la norme euclidienne.

La valeur par défaut est 1.E-6.

3.7 Opérande NMAX_ITER

◊ `NMAX_ITER = niter`

Nombre d'itérations maximum de l'algorithme.

Si `NITER` = `0` alors le nombre maximum d'itérations est calculé comme suit :

`niter` = `nequ/2` où `nequ` est le nombre d'équations du système.

La valeur par défaut est 0.

3.8 Opérande INFO

◊ `INFO =`

Paramètre d'impression :

`1 =` rien,
`2 =` on imprime le n° de l'itération à laquelle on a atteint le critère de convergence et la norme du résidu avec un message de convergence.

4 Algorithme du gradient conjugué par LDLT

Soient :

A : la matrice du système à inverser,

C : la matrice de préconditionnement, ou D = diag (A),

b : le vecteur second membre du système.

4.1 Initialisation

$x^{(o)}$ = vecteur nul si `REPRISE : 'NON'`
`cham_no` produit si `REPRISE : 'OUI'`

$$r^{(o)} = Ax^{(o)} - b$$

$$\gamma_0 = 0.$$

4.2 Corps de l'algorithme

Pour $m = 0$ à `niter` et tant que $\dfrac{\|r_m\|}{\|b\|} \leq resi$ faire

soit résoudre $C\tilde{r}^{(m)} = r^{(m)}$ où $C = LDL^T$ incomplet,

soit

$$\tilde{r}^{(m)} = D^{-1} r^{(m)}$$

$$\gamma_m = \left(r^{(m)}, \tilde{r}^{(m)}\right)$$

si $m = 0$ $\quad p^{(0)} = \tilde{r}^{(0)}$

si $m > 0$ $\quad p^{(m)} = \tilde{r}^{(m)} + \dfrac{\gamma_m}{\gamma_{m-1}} p^{(m-1)}$

$$\rho^{(m)} = \frac{\left(r^{(m)}, p^{(m)}\right)}{\left(p^{(m)}, A.p^{(m)}\right)}$$

$x^{(m+1)} = x^{(m)} - \rho^{(m)} p^{(m)}$ itéré $m+1$

$r^{(m+1)} = r^{(m)} - \rho^{(m)} Ap^{(m)}$ résidu $m+1$

Fin pour m

5 Exemple d'utilisation

```
nu     =  NUME_DDL( MATR_RIGI= mel, METHODE= 'GCPC' , RENUM= 'SANS',)
matas  =  ASSE_MATRICE (   MATR_ELEM= mel, NUME_DDL= nu
                       )
vecas  =  ASSE_VECTEUR (   VECT_ELEM= vel, NUME_DDL= nu
                       )
kmatas =  FACT_GRAD    (   MATR_ASSE= matas
                       )
dep    =  RESO_GRAD    (   CHAM_NO  = vecas ,  MATR_ASSE= matas,
                           MATR_FACT= kmatas,
                           NMAX_ITER= 1000  ,  RESI_RELA= 1e-07
                       )
```

6 Bibliographie

[1] G.V. PAOLINI & G. RADICATI di BROZOLO - Data structures to vectorize C.G. algorithms for general sparsity patterns, Bit 29, pp 617-718 (1989).

[2] J.P. GREGOIRE : Implantation et optimisation de l'algorithme du gradient conjugué [R6.01.02].