Manuel d'Utilisation
Fascicule U4.5- : Méthodes de résolution
Document : U4.50.01

Mot clé SOLVEUR

1 But

Choisir le mode de stockage des matrices et l'algorithme de résolution. Ce mot clé facteur se retrouve dans un certain nombre de commandes conduisant à la résolution de systèmes linéaires. Pour les algorithmes de résolution il permet de choisir entre factorisation "classique" de type "GAUSS" (`'LDLT'`) ou factorisation multi frontale (`'MULT_FRONT'`) ou gradient conjugué préconditionné (`'GCPC'`).

Pour chaque type de solveur, certains paramètres numériques facultatifs sont accessibles et sont décrits ici.

Par défaut, c'est le solveur `'MULT_FRONT'` qui est utilisé.

A priori la consommation en "espace disque" est croissante en passant de `GCPC` à `MULT_FRONT` puis à `LDLT`.

2 Syntaxe

```
◊  SOLVEUR = _F (

   # Factorisation de type "multi-frontale" :

   /  METHODE = 'MULT_FRONT' ,                    [DEFAUT]
      ◊  RENUM            =  /  'METIS' , [DEFAUT]
                             /  'MD'    ,
                             /  'MDA'   ,
      ◊  STOP_SINGULIER   =  /  'OUI'  ,  [DEFAUT]
                             /  'NON'  ,
      ◊  NPREC            =  /  8      ,  [DEFAUT]
                             /  nprec  ,  [I]

   # Factorisation "classique" de type GAUSS :

   /  METHODE = 'LDLT' ,                          [DEFAUT]

      ◊  RENUM            =  /  'RCMK' ,  [DEFAUT]
                             /  'SANS' ,
      ◊  STOP_SINGULIER   =  /  'OUI'  ,  [DEFAUT]
                             /  'NON'  ,
      ◊  NPREC            =  /  8      ,  [DEFAUT]
                             /  nprec  ,  [I]

   # Méthode itérative du gradient conjugué :

   /  METHODE = 'GCPC' ,

      ◊  NMAX_ITER    =      / 0,          [DEFAUT]
                             / nmax_iter,  [I]
      ◊  RESI_RELA    =      / 10-6 ,      [DEFAUT]
                             / prec ,      [R]
      ◊  PRE_COND     =      'LDLT_INC' ,  [DEFAUT]
      ◊  NIVE_REMPLISSAGE =  / 0    ,      [DEFAUT]
                             / niv  ,

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

),
```

3 Opérandes

3.1 Opérande METHODE

◊ METHODE =

Ce mot clé permet de choisir la méthode de résolution des systèmes linéaires :

/ 'MULT_FRONT' méthode directe "multi-frontale". Le stockage est 'MORSE'.
Cette méthode est parallélisée et peut être exécutée sur plusieurs processeurs (via l'interface Asterix).

/ 'LDLT' méthode directe avec factorisation de la matrice. Le stockage des matrices est alors "ligne de ciel".

/ 'GCPC' gradient conjugué avec préconditionnement : méthode itérative. Le stockage de la matrice est alors 'MORSE'.

Les valeurs par défaut des autres mots clés sont alors prises automatiquement en fonction de la méthode choisie.

3.2 METHODE : 'MULT_FRONT'

◊ RENUM =

Cet argument permet de renuméroter les nœuds du modèle :

/ 'MD' ("Minimum Degré") cette numérotation des nœuds minimise le remplissage de la matrice lors de sa factorisation.

/ 'MDA' ("Minimum Degré Approché") cette numérotation est en principe moins optimale que 'MD' en ce qui concerne le remplissage mais elle est plus économique à calculer. Elle est toutefois préférable à 'MD' pour les gros modèles (≥ 50 000 ddls).

/ 'METIS' Autre méthode de numérotation basée sur une dissection emboîtée. Cette méthode n'est possible que sur la machine compaq sauf à installer soi-même l'exécutable METIS. Sur cette machine, c'est la méthode la plus efficace (en temps CPU et en mémoire).

◊ STOP_SINGULIER = 'OUI' / 'NON'

Lorsqu'au terme de la factorisation, on constate qu'un terme diagonal d' est devenu très petit (par rapport à ce qu'il était avant la factorisation d), c'est que la matrice est presque singulière.

Soit $n = \log\left|\frac{d}{d'}\right|$

ce nombre n indique que sur une équation (au moins) on a perdu n chiffres significatifs.

Si n > nprec (mot clé NPREC ci-dessous), on considère que la matrice est singulière.

Si l'utilisateur a indiqué : STOP_SINGULIER = 'OUI', le code s'arrête en erreur fatale, sinon l'exécution se poursuit avec émission d'une alarme.

Remarque :

Toute perte importante de chiffres significatifs lors d'une factorisation est un indicateur d'un problème mal posé. Plusieurs causes sont possibles (liste non exhaustive) :

- *des conditions aux limites de blocage de la structure insuffisantes,*
- *des relations linéaires redondantes,*
- *des données numériques très hétérogènes (termes de pénalisation trop grands), ...*

◊ NPREC = nprec

C'est le nombre qui sert à déterminer si la matrice est singulière (ou non) (cf. mot clé STOP_SINGULIER ci-dessus).

3.3 METHODE : 'LDLT'

◊ RENUM =

Cet argument permet de renuméroter si on le désire les nœuds du modèle :

'SANS' on garde l'ordre initial donné dans le fichier de maillage,

'RCMK' "Reverse CUTHILL-MacKEE" cet algorithme de renumérotation est souvent efficace pour réduire la place nécessaire au stockage "ligne de ciel" de la matrice assemblée et pour réduire le temps nécessaire à la factorisation de la matrice. Cette renumérotation est faite par défaut.

◊ STOP_SINGULIER = 'OUI' / 'NON'

Voir [§3.2].

◊ NPREC = nprec

Voir [§3.2].

3.4 METHODE : 'GCPC'

◊ PRE_COND = 'LDLT_INC'

Méthode de préconditionnement : la matrice de préconditionnement est obtenue par une décomposition LDLT incomplète de la matrice assemblée.

◊ NIVE_REMPLISSAGE = / 0
/ niv

La matrice de préconditionnement (P) utilisée pour accélérer la convergence du gradient conjugué est obtenue en factorisant de façon plus ou moins complète la matrice initiale (A).

Si niv = 0 (défaut)

P a le même stockage que A. La factorisation est incomplète car on n'utilise pour les calculs que les termes que l'on peut stocker dans P.

P représente donc une approximation (médiocre) de A^{-1} ; son stockage est faible.

Si `niv = 1`

On stocke dans P en plus des termes qui avaient leur place dans le stockage initial, les "descendants" de première génération des termes initiaux. En effet lors de la factorisation, un terme nul dans A peut devenir non nul dans P. On obtient ainsi le remplissage de niveau 1.

Si `niv = 2, ...`

Le même procédé est repris : la matrice P remplie au niveau `niv-1` crée les termes de la matrice P au niveau `niv`.

Plus `niv` est grand, plus la matrice P est proche de A^{-1} et donc plus le gradient conjugué converge vite (en nombre d'itérations).

En revanche, plus `niv` est grand plus le stockage de P devient volumineux (en mémoire et sur disque) et plus les itérations sont coûteuses en CPU.

Les premiers essais ont montré (approximativement) que la taille de P valait :

- 3,5* taille (A) pour `niv = 1`
- 7,5* taille (A) pour `niv = 2`

Notre expérience de ce mot clé est encore limitée et nous conseillons d'utiliser la valeur par défaut (`niv = 0`).

Si `niv = 0` ne permet pas au gradient conjugué de converger, on essaiera successivement les valeurs `niv = 1, 2, 3`.

◊ `NMAX_ITER = niter`

Nombre d'itérations maximum de l'algorithme de résolution itératif.

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.

◊ `RESI_RELA =`

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

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

r_m est le résidu à l'itération m

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

3.5 Mot clé SYME

◊ `SYME = / 'OUI'`
`/ 'NON'`

Si la matrice du système linéaire (A) est non-symétrique, le mot clé `SYME = 'OUI'` permet de symétriser cette matrice avant la résolution du système.

La matrice A est remplacée par $A' = \frac{1}{2}\left(A + A^T\right)$

Attention :

La symétrisation de la matrice A conduit donc à résoudre un autre problème que celui que l'on cherche à résoudre ! En réalité, cette possibilité (`SYME = 'OUI'`) n'est utile que dans les commandes non-linéaires (comme `STAT_NON_LINE` par exemple), pour lesquelles la convergence vers la solution est obtenue par itérations successives. Chaque itéré est obtenu par "estimation" et l'on vérifie ensuite qu'il est "solution". Dans ce cas, une erreur légère sur les itérés n'empêche pas de converger vers la bonne solution. L'intérêt de ce mot clé est de gagner du temps lors de la résolution des systèmes linéaires. Le tout est de savoir si la symétrisation perturbe beaucoup (ou non) la solution du système linéaire ? On peut citer (par exemple) le cas des modèles 3D (ou coque) avec pression suiveuse pour lesquels la symétrisation fait gagner beaucoup de temps.

4 Exemples

4.1 Solveur par défaut

Il n'y a rien à écrire ! Mais on peut aussi écrire : `SOLVEUR=_F()`

4.2 Gradient conjugué

On veut utiliser le gradient conjugué. On pense que la convergence sera plus efficace si l'on autorise un pré-conditionnement plus poussé (`NIVE_REPLISSAGE=1`).

```
SOLVEUR=_F(METHODE ='GCPC', NIVE_REMPLISSAGE=1,)
```