Structure de données sd_solveur

Résumé :

Ce document décrit la structure de données sd_solveur. Celle-ci définit la méthode de résolution des systèmes linéaires ainsi que les paramètres attenants au solveur choisi.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

Table des Matières

Table of Contents

Généralités....2
Arborescence....2
Contenu des objets de base....2
Cas LDLT ou MULT_FRONT....3
Cas GCPC....3
Cas PETSc....4
Cas MUMPS....5

1 Généralités

Cet objet de type SOLVEUR a pour fonction de stocker et de véhiculer entre les différentes routines du code (en interne d'une commande ou entre commandes), les informations liées aux paramétrages des solveurs linéaires. En particulier, il définit la méthode de résolution des systèmes linéaires mono-domaine (LDLT, MULT_FRONT, PETSC, MUMPS ou GCPC). Cet objet est créé sur la base volatile (le cas le plus fréquent) ou sur la base globale (commandes éclatées).

Le plus souvent, il est créé et rempli *via* CRESOL/CRSV** (deux premières lettres du solveur : MU pour MUMPS solveur direct et CRSMSP pour MUMPS pré-conditionneur, PE pour PETSC, LD pour LDLT, GC pour GCPC et MF pour MULT_FRONT).
Pour répondre à quelques cas particuliers, d'autres routines dédiées créent et remplissent une SD_SOLVEUR : CRSOLV [1], OP0014 [2] et CRSINT [3]. On essaie d'en limiter le nombre et de privilégier l'usage de la routine principale : CRESOL.
Quelle que soit la routine qui crée les objets de la structure de données SOLVEUR , le dimensionnement se fait exclusivement dans une routine chapeau : SDSOLV .

Lors d'une modification de cette structure de données il faut donc veiller à :
- *mettre en cohérence, si nécessaire, les sources mentionnées ci-dessus,*
- *mettre à jour, si nécessaire, le catalogue sd_solveur.py ,*
- *mettre à jour les documentations (cette doc. D et si nécessaire la doc. U4.50.01),*
- *enrichir ou modifier, si nécessaire, quelques cas-tests.*

2 Arborescence

```
SOLVEUR (K19)   ::=record
  ♦     '.SLVK'   :   OJB   S V K24  long=14     (initialisé à 'XXXX' )
  ♦     '.SLVR'   :   OJB   S V R    long=4   (initialisé à 0.d0 )
  ♦     '.SLVI'   :   OJB   S V I    long=9   (initialisé à -9999 )
   ♦     '.SLV O '   :   OJB   S V K80  long= 31   (initialisé à ' '  )
```

3 Contenu des objets de base

[1] Pour CALC_CORR_SSD, MODE_STATIQUE, NUME_DDL_GENE, NUME_DDL, MACR_ELEM_STAT.
[2] Pour FACTORISER.
[3] Pour un appel en sous-main de MUMPS dans MODE_STATIQUE et CALC_CORR_SSD.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

3.1 Cas LDLT ou MULT_FRONT

SLVK :
SLVK(1) : méthode de résolution ('LDLT' ou 'MULT_FRONT'),
SLVK(2) : inutilisé,
SLVK(3) : inutilisé,
SLVK(4) : méthode de renumérotation (mot-clé RENUM). Les valeurs possibles sont :
'RCMK' ou 'SANS' (si LDLT)
'MD' ou 'MDA' ou 'METIS' (si MULT_FRONT),
SLVK(5) à SLVK(12) : inutilisés.
SLVK(13) : élimination des équations de Lagrange (ELIM_LAGR='OUI'/'NON').
SVLK(14) : activation du pré-conditionneur XFEM ('OUI'/'NON') . La valeur par défaut est 'NON' . Cet emplacement interagit avec la commande MODI_MODELE_XFEM ; si PRETRAITEMENT='SANS', alors l'emplacement prend la valeur ' NON '; si PRETRAITEMENT='AUTO', alors l'emplacement peut prendre la valeur 'OUI'.

SLVR :
SLVR(1) : inutilisé,
SLVR(2) : inutilisé,
SLVR(3) : taille bloc (si LDLT)
c'est la taille des blocs de l'objet .UALF d'un stockage en ligne de ciel,
SLVR(4) : inutilisé.

SLVI :
SLVI(1) : valeur de NPREC,
SLVI(2) : inutilisé,
SLVI(3) : istop
Comportement voulu en cas de singularité lors de la factorisation
0 : erreur <F> en cas de singularité ou de quasi-singularité
1 : erreur <F> en cas de singularité
alarme <A> en cas de quasi-singularité
2 : Aucun message en cas de singularité ou de quasi-singularité
SLVI(4) à SLVI(8) : inutilisés.

3.2 Cas GCPC

SLVK :
SLVK(1) : méthode de résolution 'GCPC',
SLVK(2) : préconditionnement de la matrice de travail
(PRECOND='LDLT_INC', 'LDLT_SP' ou 'SANS'),
SLVK(3) : nom de la SD solveur Mumps dans le cas PRECOND='LDLT_SP',
SLVK(4) : valeur de RENUM ('RCMK' ou 'SANS'),
SLVK(5) à SLVK(12) : inutilisés.
SLVK(13) : élimination des équations de Lagrange (ELIM_LAGR='OUI'/'NON').
SLVK(14) : inutilisé.

SLVR :
SLVR(1) : valeur de RESI_RELA (copie pour NEWTON_KRYLOV),
SLVR(2) : valeur de RESI_RELA,
SLVR(3) à SLVR(4) : inutilisés.

SLVI :
SLVI(1) : inutilisé,
SLVI(2) : valeur de NMAX_ITER,

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

SLVI(3) : inutilisé,
SLVI(4) : valeur de NIVE_REMPLISSAGE,
SLVI(5) : nombre d'itérations pour atteindre la convergence dans le cas PRECOND='LDLT_SP',
SLVI(6) : valeur de REAC_PRECOND dans le cas PRECOND='LDLT_SP',
SLVI(7) : valeur de PCENT_PIVOT dans le cas PRECOND='LDLT_SP',
SLVI(8) : istop
Comportement souhaité en cas d'erreur lors de la résolution linéaire itérative
0 : erreur <F> en cas d'échec
2 : aucun message en cas d'échec, code retour non nul

3.3 Cas PETSc

SLVK :
SLVK(1) : méthode de résolution 'PETSC',
SLVK(2) : préconditionnement de la matrice de travail
('LDLT_INC', 'LDLT_SP', 'JACOBI', 'SOR', 'ML', 'BOOMER', 'GAMG', 'SANS'),
SLVK(3) : nom de la SD solveur Mumps dans le cas PRECOND='LDLT_SP',
SLVK(4) : valeur de RENUM ('RCMK' ou 'SANS'),
SLVK(5) : inutilisé,
SLVK(6) : nom de la méthode itérative utilisée
('CG', 'CR', 'GMRES', 'GCR'),
SLVK(7) à SLVK(9) : inutilisés.
SLVK(10) : inutilisé
SLVK(11) à SLVK(12) : inutilisés.
SLVK(13) : élimination des équations de Lagrange (ELIM_LAGR='OUI'/'NON').
SLVK(14) : inutilisé.

SLVR :
SLVR(1) : valeur de RESI_RELA (copie pour NEWTON_KRYLOV),
SLVR(2) : valeur de RESI_RELA,
SLVR(3) : valeur de REMPLISSAGE,
SLVR(4) : valeur de RESI_RELA_PC (mot-clé caché).

SLVI :
SLVI(1) : inutilisé,
SLVI(2) : valeur de NMAX_ITER,
SLVI(3) : inutilisé,
SLVI(4) : valeur de NIVE_REMPLISSAGE,
SLVI(5) : nombre d'itérations pour atteindre la convergence dans le cas PRECOND='LDLT_SP',
SLVI(6) : valeur de REAC_PRECOND dans le cas PRECOND='LDLT_SP',
SLVI(7) : valeur de PCENT_PIVOT dans le cas PRECOND='LDLT_SP',
SLVI(8) : istop
Comportement souhaité en cas d'erreur lors de la résolution linéaire itérative
0 : erreur <F> en cas d'échec
2 : aucun message en cas d'échec, code retour non nul
SLVI(9) : longueur utile du vecteur SLVO

SLVO :
Le vecteur SLVO contient les options à passer à la librairie PETSc par le mot-clé OPTION_PETSC. Il s'agit d'une chaîne de caractères de longueur libre (elle est quand même limitée à 2500 caractères dans le code). Elle est découpée en SLVI(9) blocs de 80 caractères, chacun stocké dans les entrées SLVO(1), SLVO(2), etc.
SLVO(1) : 80 premiers caractères de la la chaîne OPTION_PETSC
SLVO(2) : 80 caractères suivants de la la chaîne OPTION_PETSC
etc.

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

3.4 Cas MUMPS

SLVK :
- SLVK(1) : méthode de résolution ('MUMPS'),
- SLVK(2) : prétraitements (PRETRAITEMENTS='AUTO' ou 'SANS'),
- SLVK(3) : algorithme de résolution souhaité (TYPE_RESOL=)
  - 'NONSYM' : matrice non-symétrique (factorisation LU)
  - 'SYMGEN' : matrice symétrique "générale"
  - 'SYMDEF' : matrice symétrique "définie positive"
  - 'AUTO' : choix automatique fait au vu des caractéristiques de la matrice
- SLVK(4) : renuméroteur souhaité (RENUM='AUTO','AMD','AMF','QAMD','PORD', 'PTSCOTCH'/SCOTCH' ou 'PARMETIS'/'METIS'),
- SLVK(5) : type d'accélération activée (ACCELERATION='AUTO', 'FR', 'FR+', 'LR' ou 'FR+'),
- SLVK(6) : élimination «virtuelle» de la 2ieme famille de Lagrange lorsqu'on transmet la matrice Aster à MUMPS (ELIM_LAGR='LAGR2'),
- SLVK(7) : précision mixte (MIXER_PRECISION='OUI'/'NON'),
- SLVK(8) : utilisation en préconditionneur simple précision pour le GCPC ('OUI'/'NON'),
- SLVK(9) : gestion de la mémoire allouée par MUMPS (GESTION_MEMOIRE='IN_CORE'/'OUT_OF_CORE'/'AUTO'/'EVAL'),
- SLVK(10) : inutilisé
- SVLK(11) : gestion des post-traitements (POSTTRAITEMENTS='SANS','AUTO','FORCE'),
- SVLK(12) : numéro de version de MUMPS (par exemple: '5.1.0'). Il s'agit d'un numéro version licite, c'est-à-dire testé et approuvé par la version de *Code_Aster* considérée. Dans la cas contraire on s'arrête en ERREUR_F avant le remplissage de ce champ. Valeur remplie juste après l'initialisation de l'occurence MUMPS uniquement *via* les routines amump*/mumpu.F.
- SLVK(13) : élimination des équations de Lagrange (ELIM_LAGR='OUI'/'NON'/'LAGR2' sauf si calcul modal, alors ELIM_LAGR='NON'/'LAGR2').
- SVLK(14) : activation du pré-conditionneur XFEM ('OUI'/'NON') . La valeur par défaut est 'NON' . Cet emplacement interagit avec la commande MODI_MODELE_XFEM ; si PRETRAITEMENT='SANS', alors l'emplacement prend la valeur ' NON '; si PRETRAITEMENT='AUTO', alors l'emplacement peut prendre la valeur 'OUI'.

SLVR :
- SLVR(1) : valeur de FILTRAGE_MATRICE,
- SLVR(2) : valeur de RESI_RELA (calcul et contrôle de la qualité de la solution, déclenchement des post-traitements suivant la valeur de POSTTRAITEMENTS),
- SLVR(3) :inutilisé.
- SLVR(4) : taux de compression 'low-rank' (mot-clé LOW_RANK_SEUIL).

SLVI :
- SLVI(1) : valeur de NPREC (comme LDLT et MULT_FRONT),
- SLVI(2) : pourcentage de mémoire supplémentaire nécessaire aux pivotages tardifs (valeur de PCENT_PIVOT)
- SLVI(3) : valeur de ISTOP (comme LDLT et MULT_FRONT),
- SLVI(4) : indicateur pour dire à MUMPS de ne pas stocker les termes de sa factorisée (intéressant si par exemple, on a juste besoin d'un résultat global type calcul de déterminant ou critère de Sturm). Si il vaut 1, on ne stocke pas cette factorisée (gros gain mémoire), sinon on la garde suivant le mode standard.
  Cette fonctionnalité n'est activée qu'à partir de MUMPS 4.10.0 (pour les versions en deçà on ne fait rien et on émet un message UTMESS_I (si INFO=2). Valeur temporaire (on remet, si nécessaire, après usage sa valeur initiale) remplie dans les routines vpstur.f et apchar.f .

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)

SLVI(5) : indicateur pour dire à MUMPS de calculer en plus le déterminant de la matrice. Si il vaut 1, on le calcule et on le stocke dans l'objet temporaire ' &&AMUMP.DETERMINANT ' (cf. amumpu.F), sinon on ne le calcule pas.
Cette fonctionnalité n'est activée qu'à partir de MUMPS 4.10.0 (pour les versions en deçà on s'arrête en ERREUR_F . Valeur temporaire (on remet, si nécessaire, après usage sa valeur initiale) remplie dans les routines vpstur.f et apchar.f .

SLVI(6) : indicateur pour dire à MUMPS le nombre de matrices factorisées que l'on va construire en même temps. Ce paramètre n'est pris en compte que pour les options GESTION_MEMOIRE='AUTO' et 'EVAL'. Il est initialisé à 1 par défaut. Juste après sa lecture il y'a un arrêt en ASSERT si ce chiffre est illicite (<1 ou >nmxins).
Il est de la responsabilité du programmeur de modifier ce chiffre lorsqu'il sait qu'il aura plusieurs factorisées à gérer en simultané. Si cela n'est pas fait, la gestion mémoire sera sous-optimale et le calcul ralentit (' AUTO ') ou les chiffres affichés seront minorés (' EVAL ').
Ce cas de figure a été, par exemple, pris en compte dans STAT_NON_LINE + ' CRIT_STAB '.
RQ : On suppose que les factorisées simultanées requièrent le même espace et on ne gére pas les précédences (sur 2 matrices, à la première on consacre la moitié de l'espace disponible, à la seconde, on fait de même, alors qu'on pourrait tout lui donner).

SLVI(7) : valeur du mot-clé REDUCTION_MPI ; c'est-à-dire le facteur de réduction du nombre de processus MPI qui va s'appliquer uniquement dans MUMPS. Celui-ci va alors augmenter, dans la même proportion, le nombre de threads OpenMP. Licite uniquement sur une version parallèle MPI et activé, uniquement si > 1 et avec MUMPS en usage solveur direct.

SLVI(8) : valeur du mot-clé NB_RHS ; c'est-à-dire le nombre de seconds membres ('Right Hand Side' dans le jargon) que l'on donne à MUMPS pour une descente-remontée simultanée. Activé uniquement avec MUMPS en usage solveur direct. Si négatif, les RHS sont considérés comme très creux au lieu du format dense standard (valeur positive).

Document diffusé sous licence GNU FDL (http://www.gnu.org/copyleft/fdl.html)