
Opérateur RESOUDRE

1 But

Résoudre un système d'équations linéaires (méthode directe ou itérative)

Les méthodes de résolutions implantées dans *Code_Aster* et utilisables par cette commande sont :

- 1) la méthode `MULT_FRONT` (méthode directe),
- 2) la méthode `LDLT` (méthode directe),
- 3) la méthode `MUMPS` (méthode directe),
- 4) la méthode `GCPC` (méthode itérative),
- 5) la méthode `PETSC` (méthode itérative).

Le choix effectif de la méthode se fait au travers de la commande `FACTORISER` [U4.55.01].

Pour les méthodes directes, la matrice doit avoir été préalablement factorisée par la commande `FACTORISER` [U4.55.01]. Dans le cas des méthodes itératives avec pré-conditionnement, la matrice de pré conditionnement est fournie elle-aussi par l'opérateur `FACTORISER` [U4.55.01].

L'opérateur permet des résolutions complexes pour les méthodes "directes" (pas pour les méthodes itératives).

Produit une structure de données de type `cham_no`.

2 Syntaxe

```

U [cham_no_*] = RESOUDRE
(
  ◊ reuse = U,
  ◆ MATR = A,
  # Si méthode LDLT, MULT_FRONT, MUMPS :
                                / [matr_asse_DEPL_R]
                                / [matr_asse_DEPL_C]
                                / [matr_asse_TEMP_R]
                                / [matr_asse_TEMP_C]
                                / [matr_asse_PRES_R]
                                / [matr_asse_PRES_C]

  # Si méthode GCPC ou PETSC :
                                / [matr_asse_DEPL_R]
                                / [matr_asse_TEMP_R]
                                / [matr_asse_PRES_R]

  ◆ CHAM_NO = B, / [cham_no]
  ◊ CHAM_CINE = vcine, / [cham_no]

  # si méthode PETSC :
  ◊ ALGORITHME = / 'GMRES', [DEFAULT]
                / 'CG',
                / 'CR',
                / 'GCR',

  # si méthode MUMPS, GCPC, PETSC :
  ◊ RESI_RELA = / 1.e-6, [DEFAULT]
                / eps, [R]

  # si méthode GCPC ou PETSC :
  ◊ ◆ MATR_PREC = precondition, / [matr_asse_DEPL_R]
                                / [matr_asse_TEMP_R]
                                / [matr_asse_PRES_R]
  ◊ NMAX_ITER = / niter, [I]
                / 0, [DEFAULT]

  # si méthode MUMPS :
  ◊ POSTTRAITEMENTS = ... (voir mot clé SOLVEUR [U4.50.01])

  ◊ TITRE = titr , [l_K80]
  ◊ INFO = / 1 , [DEFAULT]
          / 2 ,
)

```

Si CHAM_NO : [cham_no_DEPL_R] alors (*) → DEPL_R
 [cham_no_TEMP_R] → TEMP_R
 [cham_no_PRES_C] → PRES_C

3 Généralités

Cette commande permet de résoudre :

- par une méthode directe, le système linéaire $\mathbf{AX}=\mathbf{B}$, où \mathbf{A} est une matrice préalablement "factorisée" par la commande `FACTORISER` [U4.51.01].
- par une méthode itérative (GCPC ou PETSC), le système linéaire $\mathbf{P}^{-1}\mathbf{AX}=\mathbf{P}^{-1}\mathbf{B}$, où \mathbf{P}^{-1} est une matrice de pré-conditionnement déterminée par la commande `FACTORISER` [U4.51.01] et \mathbf{A} la matrice assemblée initiale.

La résolution est possible pour des conditions aux limites de Dirichlet (conditions aux limites cinématiques) dualisées ou éliminées [U2.01.02]. Dans ce dernier cas, si le chargement $\mathbf{X}=\mathbf{X}_0$ sur le « bord » Γ_0 est appliqué avec une charge cinématique (opérateur `AFFE_CHAR_CINE` [U4.44.03]) prise en compte dans la matrice assemblée (opérateur `ASSE_MATRICE` [U4.61.22]), la « valeur » de ce chargement (\mathbf{X}_0) , calculée par l'opérateur `CALC_CHAR_CINE` [U4.61.03] doit être fournie par le mot clé `CHAM_CINE`.

4 Opérandes

4.1 Opérande `MATR`

♦ `MATR = A,`

Nom de la matrice du système à résoudre :

- Pour les méthodes directes, on fournit à `MATR` le concept modifié par l'opérateur `FACTORISER` ; cette matrice peut être réelle ou complexe, symétrique ou non.
- Pour les méthodes itératives, on fournit à `MATR` la matrice initiale. La matrice de pré-conditionnement est à fournir avec le mot-clé `MATR_PREC`.

4.2 Opérande `CHAM_NO`

♦ `CHAM_NO = B,`

Nom du vecteur second membre (en général obtenu par la commande `ASSE_VECTEUR`).

4.3 Opérande `CHAM_CINE`

♦ `CHAM_CINE = vcine,`

Nom du vecteur représentant la « valeur » des conditions aux limites de Dirichlet éliminées (c'est-à-dire appliquées avec une des commandes `AFFE_CHAR_CINE` ou `AFFE_CHAR_CINE_F`).

Ce `cham_no` provient de l'exécution de l'opérateur `CALC_CHAR_CINE` sur la liste des `char_cine` (chargements cinématiques) associée à la matrice assemblée `A` [U2.01.02].

4.4 Opérande `ALGORITHME`

♦ `ALGORITHME = / 'GMRES' [DEFAULT]
/ 'CG'
/ 'CR'
/ 'GCR'`

Ce mot clé sert à choisir l'algorithme de la méthode itérative `PETSC`. Les différents algorithmes disponibles sont documentés dans le mot-clé `SOLVEUR[U4.50.01]`.

4.5 Opérande `MATR_PREC`

◇ `MATR_PREC = precondition`

Matrice de pré-conditionnement, obtenue par l'opérateur `FACTORISER [U4.55.01]`.

Le pré-conditionnement est nécessaire dans les méthodes itératives pour obtenir une bonne convergence avec un minimum d'itérations.

Avec la méthode `GCPC`, la matrice de pré-conditionnement est une matrice distincte de la matrice du problème (mot clé `MATR`).

En revanche, avec la méthode `PETSC`, on utilise la même matrice pour `MATR_PREC` et `MATR`, ce qui veut dire que la commande `FACTORISER` doit être faite « en place » (avec le mot clé `reuse`). Voir exemple ci-dessous.

4.6 Opérande `RESI_RELA`

◇ `RESI_RELA = / 1.e-6, [DEFAULT]`
`/ eps, [R]`

Ce mot-clé est décrit dans `[U4.50.01]`

Pour les méthodes itératives `GCPC` et `PETSC`, il s'agit du critère de convergence de l'algorithme. Pour la méthode `MUMPS`, ce mot-clé permet de vérifier la qualité de la solution.

4.7 Opérande `NMAX_ITER`

◇ `NMAX_ITER = niter`

Nombre d'itérations maximum de l'algorithme itératif.

Si `niter = 0` alors l'algorithme choisit un nombre d'itérations par défaut.

4.8 Opérande `TITRE`

◇ `TITRE = titr,`

Titre que l'on veut donner au résultat produit `[U4.03.01]`.

4.9 Opérande `INFO`

◇ `INFO =`

1 : pas d'impression.

2 : impressions

5 Exemples

5.1 Résolution par la méthode directe `MULT_FRONT`

- Constitution de la matrice assemblée et du second membre :

On a calculé auparavant les termes élémentaires `KEL`, `FEL`.

```
NU =NUME_DDL( MATR_RIGI=KEL )
K  =ASSE_MATRICE( MATR_ELEM=KEL, NUME_DDL=NU, )
F  =ASSE_VECTEUR( MATR_ELEM=FEL, NUME_DDL=NU, )
```

- Factorisation :

```
K  =FACTORISER( reuse=K, MATR_ASSE=K, METHODE='MULT_FRONT', )
```

- Résolution :

```
U  =RESOUDRE( MATR=K, CHAM_NO=F, )
```

- pour l'utilisation des charges cinématiques (avec élimination des degrés de liberté imposés), voir l'exemple donné dans la commande `AFFE_CHAR_CINE` [U4.44.03].

5.2 Résolution par la méthode `MUMPS`

```
NU      = NUME_DDL( MATR_RIGI= KEL)

K       = ASSE_MATRICE ( MATR_ELEM= KEL, NUME_DDL= NU)
F       = ASSE_VECTEUR ( VECT_ELEM= FEL, NUME_DDL= NU )
K       = FACTORISER   ( reuse= K, MATR_ASSE= K, METHODE= 'MUMPS' )
dep     = RESOUDRE     ( CHAM_NO = F , MATR= K )
```

5.3 Résolution par la méthode du gradient conjugué pré conditionné

```
NU      = NUME_DDL( MATR_RIGI= KEL)

K       = ASSE_MATRICE ( MATR_ELEM= KEL, NUME_DDL= NU )
F       = ASSE_VECTEUR ( VECT_ELEM= FEL, NUME_DDL= NU )
KPREC  = FACTORISER   ( MATR_ASSE= K, METHODE= 'GCPC',
                       PRE_COND='LDLT_INC' )
dep     = RESOUDRE     ( CHAM_NO = F , MATR= K, MATR_PREC= KPREC,
                       NMAX_ITER= 1000 , RESI_RELA= 1e-07
                       )
```

5.4 Résolution par la méthode `PETSC`

```
NU      = NUME_DDL( MATR_RIGI= KEL)

K       = ASSE_MATRICE ( MATR_ELEM= KEL, NUME_DDL= NU )
F       = ASSE_VECTEUR ( VECT_ELEM= FEL, NUME_DDL= NU )
K       = FACTORISER   ( reuse=K, MATR_ASSE= K, METHODE= 'PETSC')
dep     = RESOUDRE     ( CHAM_NO = F , MATR= K, MATR_PREC= K,
                       ALGORITHM='GMRES',
                       NMAX_ITER= 1000 , RESI_RELA= 1e-07 )
```