Notice d'utilisation de MFront avec code_aster

Résumé :

On décrit ici l'utilisation du couplage entre **code_aster** et le module d'intégration de lois de comportement MFront, qui est diffusé en Open Source sous licence GPL et CECILL-A (https://thelfer.github.io/tfel/web/index.html).

MFront permet de définir une loi de comportement de façon simple, proche des équations physiques, sans avoir à se soucier des méthodes de résolution. Il offre des classes de manipulation de données (tenseurs, par exemple) de façon conviviale et met à disposition des algorithmes de résolution des équations non-linéaires associées efficaces en termes de performances. MFront est adapté aussi bien aux schémas temporels implicites qu'explicites.

Ces comportements sont utilisables dans **code_aster** pour les modélisations 3D et 2D, coques, THM et pour les éléments de joints. Les grandes déformations peuvent être utilisées avec GDEF_LOG. De plus, MFront produit un code efficace, car les opérations tensorielles sont optimisées comme le montrent les *benchmarks* réalisés avec des comportements équivalents de **code_aster**.

Il faut bien noter que l'utilisation de ces lois de comportement « à façon » (mot-clé RELATION='MFRONT' sous COMPORTEMENT) implique une validation spécifique pour l'étude envisagée, car on se place hors du domaine qualifié de **code_aster**.

Ce document décrit la façon d'utiliser un comportement défini par l'utilisateur à l'aide de MFront dans un fichier de commandes de **code_aster**.

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

1 Définition d'une loi de comportement avec MFront

1.1 Écriture d'un comportement MFront

On écrit la loi de comportement dans le fichier MFront, suivant les conseils fournis dans le tutoriel[1], ou en s'inspirant d'un comportement déjà disponible dans les tests de **code_aster** (*cf.* §2).

Nous prendrons ici pour exemple le comportement visco-plastique de Lemaître écrit sous forme implicite. Le fichier « `Lemaitre.mfront` » s'écrit :

```
@DSL Implicit;
@Behaviour Lemaitre;
@Algorithm NewtonRaphson_NumericalJacobian;
@UseQt true;

@Author EDF R&D ERMES;
@Date 21/11/2023;
@Description{
  Lemaitre behaviour law (cf. Aster doc. R5-03-08 for details):
  eto=eel+dp_dt*nv,
  sig=D:eel,
  nv=1.5*dev(sig)/seq,
  dp_dt=((1/K)*seq/p^(1/M))^Np;
}

@Theta 0.5;
@Epsilon 1.0e-12;
@IterMax 100;

@MaterialProperty stress young;
@MaterialProperty real poisson;
@MaterialProperty real Np;
@MaterialProperty result_type<real,stress,OpDiv> UN_SUR_K;
@MaterialProperty real UN_SUR_M;

@StateVariable real p;

@LocalVariable StiffnessTensor Dv;

@InitLocalVariables{
  const auto lambda=computeLambda(young,poisson);
  const auto mu=computeMu(young,poisson);
  Dv=lambda*Stensor4::IxI()+real(2.0)*mu*Stensor4::Id();
}

@ComputeStress{
  sig=Dv*eel;
}

@Integrator{
  const auto seq=sigmaeq(sig);
  const auto p_mts=max(p+theta*dp,real(1.0e-8));
  const auto slem=seq*UN_SUR_K/pow(p_mts,UN_SUR_M);
  const auto vp=frequency(1.0)*pow(slem,Np);
  const auto iseq=real(1.0)/max(seq,young*real(1.e-14));
  const auto n=real(1.5)*deviator(sig)*iseq;
```


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

```
  feel+=vp*dt*n;
  feel-=deto;
  fp-=vp*dt;
}

@TangentOperator{
  Stensor4 Je;
  getPartialJacobianInvert(Je);
  Dt=Dv*Je;
}
```

Cette loi est équivalente à la loi LEMAITRE, disponible dans **code_aster** (cf. [R5.03.08]).

1.2 Création de la bibliothèque dynamique à partir d'un comportement MFront

Pour être utilisable dans **code_aster**, le comportement MFront ainsi programmé doit être transformé en bibliothèque dynamique. Pour cela, il faut le « compiler ». Cela peut être fait de deux façons :

1) **Solution recommandée : utilisation de CREA_LIB_MFRONT**

La macro-commande CREA_LIB_MFRONT permet de construire la bibliothèque automatiquement. Il suffit de fournir le fichier MFront en entrée sur une unité logique ainsi que le nom du comportement décrit dans ce fichier (derrière le mot-clé @Behaviour) :

```
behav_mfront=CREA_LIB_MFRONT(NOM_COMPOR="Lemaitre",
                             UNITE_MFRONT=10,)
```

Ce mode de fonctionnement est à privilégier, notamment car il facilite l'utilisation dans AsterStudy. Pour plus d'informations, voir [U7.03.04].

Il est possible d'ajouter DEBUG= " OUI " à la fin de la macro-commande CREA_LIB_MFRONT si l'utilisateur souhait construire la bibliothèque en mode DEBUG .

2) **Pour les experts : construction préalable de la bibliothèque à la main**

Dans un terminal, il faut d'abord « compiler » le fichier MFront :

```
mfront --obuild --interface=generic Lemaitre.mfront
```

Cela crée deux répertoires : include et src. Ce dernier contient en particulier la bibliothèque dynamique libBehaviour.so.

Celle-ci doit ensuite être fournie en entrée sur une unité logique dans la macro-commande CREA_LIB_MFRONT, comme expliqué précédemment :

```
behav_mfront=CREA_LIB_MFRONT(NOM_COMPOR="Lemaitre",
                             UNITE_LIBRAIRIE=11,)
```

Enfin, notons qu'il est possible d'ajouter l'option --debug dans la ligne de commande mfront précédente si l'utilisateur souhaite construire la bibliothèque en mode DEBUG.

Remarque :

Les deux méthodes décrites précédemment sont exactement identiques d'un point de vue informatique, la première cachant à l'utilisateur la compilation du fichier Mfront, réalisée « *à la volée* ». Dans les deux cas, la macro-commande CREA_LIB_MFRONT produit une structure de données compor_mgis qui pourra pourra être utilisée plus tard dans le fichier de commandes.

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

1.3 Utilisation du comportement dans le fichier de commandes

L'utilisation du comportement MFront dans le fichier de commandes s'effectue de la façon suivante :

1) Les données matériaux sont fournies dans DEFI_MATERIAU [U4.43.01], sous MFRONT / MFRONT_FO.
Par exemple, pour la loi de Lemaître (les unités sont ici en MPa pour young et K) :

```
young=210000.0
poisson=0.3
Np=11.0
UN_SUR_K=0.00032840723
UN_SUR_M=0.178571429

mat_mfront=DEFI_MATERIAU(
                      ELAS=_F(E=young, NU=poisson),
                      MFRONT=_F(LISTE_COEF=(young, poisson,
                                  Np, UN_SUR_K, UN_SUR_M,),
                       ),
                              )
```

Il est important de noter que les valeurs de LISTE_COEF (young, poisson , ...) sont à donner **dans l'ordre** des @MaterialProperties du fichier MFront.

Remarque :

Pour des utilisateurs avancés de MFront, il est possible de lister les propriétés matériaux d'un fichier MFront dans le bon ordre à l'aide de l'exécutable mfront-query via la commande:

```
mfront-query --material-properties Lemaitre.mfront
```

.

2) Sous le mot-clé COMPORTEMENT des commandes de calcul non-linéaire STAT_NON_LINE, DYNA_NON_LINE, ou SIMU_POINT_MAT :

```
sol_mfront=STAT_NON_LINE(MODELE=MO,
                    CHAM_MATER=CMF,
                    INCREMENT=_F(LIST_INST=L_INST, NUME_INST_FIN=5,
                          ),
                    NEWTON=_F(REAC_ITER=1),
                    EXCIT=(_F(CHARGE=CH1),
                            _F(CHARGE=CH2),
                      ),
                    COMPORTEMENT=_F(RELATION='MFRONT',
                          COMPOR_MFRONT=behav_mfront,
                          RESI_INTE_MAX=1.0E-12,
                          ITER_INTE_MAXI=100,
                          ),
                )
```

Dans le cas où RELATION='MFRONT', notons que seul le mot-clé COMPOR_MFRONT est obligatoire et que le nom prescrit (ici behav_mfront) doit être cohérent avec le nom utilisé en sortie de CREA_LIB_MFRONT pour construire la bibliothèque.

Cependant, les deux mot-clés optionnels RESI_INTE_MAXI et ITER_INTE_MAX sont **automatiquement** transmis à MFront, même lorsque ceux-ci ne sont pas renseigné côté Aster.

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

En effet, soit RESI_INTE_MAXI est renseigné sous COMPORTEMENT et c'est alors la valeur prescrite qui est utilisée, soit c'est la valeur par défaut (1.0E-8) qui est utilisée. Dans les deux cas, une valeur est fournie et elle vient toujours écraser celle prescrite dans le fichier MFront **sans prévenir l'utilisateur**. Côté MFront, il s'agit du paramètre epsilon automatiquement défini à 1.0E-8 lorsque qu'une méthode implicite est invoquée, et modifiable via le mot-clé @Epsilon dans le fichier MFront.

De manière similaire, la valeur fournie côté Aster pour ITER_INTE_MAXI (soit celle renseignée sous COMPORTEMENT , soit celle par défaut *i.e* . 100) vient toujours automatiquement écraser celle prescrite dans le fichier MFront, sans prévenir l'utilisateur. Côté MFront, il s'agit du paramètre iterMax automatiquement défini à 100 lorsque qu'une méthode implicite est invoquée, et modifiable via le mot-clé @IterMax (ou @MaximumNumberOfIterations) dans le fichier MFront.

Remarque :

Comme c'est une source fréquente d'erreur, il est recommandé de **toujours** spécifier le mot-clé optionnel RESI_INTE_MAXI dans le cas d'un comportement utilisant MFront, et de prescrire la valeur fournie dans le fichier MFront pour @E psilon . Et de même pour ITER_INTE_MAXI si @IterMax (ou @MaximumNumberOfIterations) apparaît dans le fichier MFront.

Dans le cas où RELATION='MFRONT', il est également possible d'ajouter sous COMPORTEMENT les autres mots-clés suivants (*cf.* [U4.51.11] pour plus de détails) :

- VERI_BORNE ('ARRET' par défaut, sinon 'SANS' ou 'MESSAGE') permet de piloter le comportement de MFront dans le cas où la valeur d'un des paramètres matériaux dépasse des bornes spécifiées **à l'avance** dans le fichier MFront (via le mot-clé @Bounds). Par défaut, l'intégration de la loi échoue, sinon l'erreur est passée sous silence, l'impression d'un message étant généré ou non côté MFront (malheureusement non-récupéré par Aster pour le moment).

Remarque :

Contrairement au mot-clé @Bound piloté par VERI_BORNE côté Aster, on mentionne que le mot-clé @PhysicalBounds dans un fichier MFront entraîne automatiquement une erreur si le paramètre matériau associé dépasse les bornes prescrites (échec automatique de la loi de comportement).

- SYME_MATR_TANG ('OUI' par défaut, sinon 'NON') permet de spécifier si la matrice de comportement est symétrique ou non.
- ALGO_CPLAN ('DEBORST' par défaut, sinon 'ANALYTIQUE') permet de définir la modélisation dans le cas des contraintes planes (*cf.* §1.4 qui suit).
- DEFORMATION ('PETIT' par défaut, sinon GDEF_LOG) permet de préciser le type de petites déformations utilisées. Malheureusement, les modèles de grandes déformations ne sont actuellement plus accessibles pour les comportements issus d'un fichier MFront (*cf.* § 1.5 suivant).
- TOUT='OUI' (mot-clé par défaut) ou l'autre mot-clé habituel GROUP_MA pour prescrire un ou plusieurs groupes de mailles.
- TYPE_MATR_TANG permet de vérifier la matrice tangente, calculée dans ce cas par MFront. La valeur 'PERTURBATION' permet de la remplacer par une matrice tangente perturbée tandis que 'VERIFICATION' va les comparer. On mentionne que VALE_PERT_RELA (1.0E-5 par défaut) permet de préciser la valeur de la perturbation relative utilisée. On rappelle aussi qu'on peut utiliser RESI_RADI_RELA à la place de TYPE_MATR_TANG (*cf.* §4.12 de [U4.51.11]) .

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

Enfin, sous COMPORTEMENT dans le cas où RELATION='MFRONT', mentionnons que les mots-clés suivants sont ignorés (*cf.* [U4.51.11]):

- ITER_INTE_PAS
- ALGO_INTE
- RIGI_GEOM
- PARM_THETA
- REGU_VISC

ou exclu

- RESI_INTE_RELA
- POST_ITER
- POST_INCR
- COMPOR
- NB_VARI
- LIBRAIRIE
- NOM_ROUTINE

Le reste du fichier de commandes est inchangé. Par exemple, si on utilise la loi de Lemaître définie dans Lemaitre.mfront pour le test ssna01a, on peut comparer les résultats avec ceux obtenus par la loi de LEMAITRE dans **code_aster**.

Remarque :

Lorsque la loi de comportement échoue côté MFront, un message d'erreur est généré mais il n'est malheureusement pas récupérable par code_aster pour le moment. En particulier, il se peut que la loi de comportement échoue pour une raison différente que la non-convergence de l'algorithme (division par zéro, racine d'un nombre négative ou paramètre matériau en dehors de ses bornes de validité par exemple).

1.4 Modélisations disponibles

Les lois de comportement MFront sont utilisables nativement en 3D (et ses dérivés comme 3D_SI et 3D_INCO_*), en déformations planes (D_PLAN_*) et en axisymétrique (AXIS_*). Pour le cas des contraintes planes (C_PLAN), il y a deux situations :

- soit on utilise le mode contraintes planes natif de MFront en sélectionnant ALGO_CPLAN='ANALYTIQUE' dans le mot-clef facteur COMPORTEMENT. Dans ce cas, le fichier MFront devra utiliser, suivant la valeur de @ModellingHypothesis, l'intégrateur adéquat ;

- soit on utilise l'algorithme de De Borst (voir [R5.03.03]) avec ALGO_CPLAN='DEBORST' dans le mot-clef facteur COMPORTEMENT. Dans ce cas, le fichier MFront devra pouvoir utiliser une modélisation de type axisymétrique.

  Côté Aster, on rappelle que la convergence concernant la vérification de l'état des contraintes planes peut être dissociée de la convergence de l'algorithme de Newton grâce au mot-clé RESI_CPLAN_RELA (1.0E-6 par défaut) ou RESI_CPLAN_MAXI pour le critère d'arrêt relative ou absolu respectivement, ainsi que ITER_CPLAN_MAXI (10 par défaut) pour le nombre maximal d'itérations laissé à l'algorithme de De Borst au sein de chaque itération de l'algorithme de résolution des équations d'équilibre (schémas emboîtés).

Pour les comportements unidimensionnels (TUYAU_*, POU_D_EM et POU_D_TGM), il faut nécessairement utiliser l'algorithme de De Borst car MFront ne supporte pas ces modélisations.

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

1.5 Modèles de grandes déformations

Tous les modèles de grandes déformations existants dans code_aster ne sont malheureusement plus utilisables pour le moment avec des comportements MFront. C'est le cas de `'PETIT_REAC'`, `'GROT_GDEP'`, `'SIMO_MIEHE'` et `'GREEN-LAGRANGE'`.

Cependant, comme le modèle `GDEF_LOG` de code_aster s'appuient nécessairement sur une loi écrite en petites déformations, il est possible d'utiliser `DEFORMATION='GDEF_LOG'` (à la place de `'PETIT'`, valeur par défaut). Il n'y a donc rien à faire de particulier dans le fichier MFront.

2 Exemples d'utilisation

D'autres exemples sont fournis dans le tutoriel disponible sur le site internet de MFront (https://thelfer.github.io/tfel/web/gallery.html).

Les cas-tests `mfron*` permettent de vérifier le mot clé `RELATION='MFRONT'`, avec un fichier MFront fourni en donnée du test.

Remarque :

> Les tutoriels et informations concernant l'interface entre Aster et MFront issues du nouveau site MFront (https://thelfer.github.io/tfel/web/index.html) et de l'ancien (https://tfel.sourceforge.net) décrive l'ancienne manière dont code_aster communiquait avec MFront. Depuis la version 16.3.13, code_aster n'utilise plus l'interface historique (option `--interface=aster` dans la ligne de commande MFront) mais l'interface générique via son support appelé `mgis` et dont une description est disponible sur https://thelfer.github.io/mgis/web/index.html .
>
> En particulier, l'option `--@AsterGenerateMtestFileOnFailure=true` n'est plus disponible durant la compilation du fichier Mfront. Elle permettait de récupérer automatiquement les données de simulation d'Aster sur un point matériel (fichier `*.mtest`) lorsque l'intégration échouait sur celui-ci.

3 Références

[1] T. Helfer, J.-M. Proix. *Écriture de lois de comportement avec MFront : tutoriel*. Décembre 2014. http://tfel.sourceforge.net/documents/tutoriel/tutoriel.pdf

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