Prise en compte de l'hypothèse des contraintes planes dans les comportements non linéaires

Résumé :

Ce document décrit une méthode générale d'intégration des modèles de comportements non linéaires (élastoplastiques, viscoplastiques, endommageants,...) en contraintes planes.

Ceci est réalisé par une méthode de condensation statique due à R. de Borst.

Cette méthode permet d'utiliser la modélisation C_PLAN, ou bien les modélisations COQUE_3D, DKT et TUYAU pour tous les modèles de comportements incrémentaux de (STAT/DYNA)_NON_LINE disponibles en axisymétrique ou en déformations planes.

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

1 Introduction

On présente ici une méthode générale d'intégration des modèles de comportements non linéaires (plasticité, viscoplasticité, endommagement) en contraintes planes. Si le comportement choisi n'est pas intégré analytiquement en contraintes planes, la méthode de R. De Borst est activée automatiquement pour les modélisations C_PLAN, DKT, COQUE3D et TUYAU.

2 Difficulté d'intégration des comportements non linéaires en contraintes planes

La modélisation C_PLAN, (ainsi que les modélisations COQUE_3D, DKT, TUYAU) suppose que l'état de contraintes local est plan, c'est à dire que $\sigma_{zz}=0$, z représentant la direction de la normale à la surface. Les tenseurs de contraintes et de déformations ont donc l'allure suivante (en C_PLAN) :

$$\varepsilon=\begin{pmatrix}\varepsilon_{xx} & \varepsilon_{xy} & 0\\ \varepsilon_{xy} & \varepsilon_{yy} & 0\\ 0 & 0 & \varepsilon_{zz}\end{pmatrix}$$
$$\sigma=\begin{pmatrix}\sigma_{xx} & \sigma_{xy} & 0\\ \sigma_{xy} & \sigma_{yy} & 0\\ 0 & 0 & \sigma_{zz}\end{pmatrix} \qquad \text{eq.2-1}$$

Remarque :

Pour les coques, il faut ajouter des termes dus au cisaillement transverse ($\sigma_{xzn}\sigma_{yz}$), mais ceux-ci sont traités élastiquement et n'interviennent pas dans la résolution du comportement local.

Cette hypothèse implique que la déformation correspondante est a priori indéterminée (contrairement aux autres modélisations bidimensionnelles où l'on fait une hypothèse directement sur ε_{zz}). Elle ne peut être déterminée qu'à l'aide de la relation de comportement. Or la condition $\sigma_{zz}=0$ n'est pas anodine pour l'intégration du comportement, où l'on calcule un accroissement de contrainte $\Delta\sigma$ en fonction de l'accroissement de déformation $\Delta\varepsilon$ fourni par l'algorithme de Newton. Dans le cas de l'élasticité linéaire, la prise en compte de cette condition est simple et permet de trouver :

$$\varepsilon_{zz}=-\frac{\nu}{1-\nu}(\varepsilon_{xx}+\varepsilon_{yy})$$

Mais si le comportement est non linéaire, $\Delta\varepsilon_{zz}$ ne peut pas être calculé uniquement à partir de Δu et ne se déduit pas simplement des autres composantes du tenseur des déformations. La prise en compte de cette hypothèse doit alors être faite (quand c'est réalisable) d'une façon spécifique à chaque comportement, et amène bien souvent à des difficultés de résolution supplémentaires : c'est le cas en particulier pour le comportement de Von Mises à l'écrouissage isotrope [R5.03.02]. De ce fait, beaucoup de modèles de comportement ne sont pas disponibles en contraintes planes.

La méthode présentée ici a le gros avantage de ne nécessiter aucun développement particulier dans l'intégration du comportement pour satisfaire à l'hypothèse des contraintes planes. Elle est utilisable dès que le modèle de comportement est disponible en axisymétrique ou en déformations planes.

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

3 Principe du traitement des contraintes planes par la méthode De Borst

3.1 Rappel de la méthode de Newton

En statique non-linéaire on est amené à résoudre l'équation suivante ([cf. R5.03.01]) :

$$R(u)=f^{\text{int}}(u)-f^{ext}(t)\rightarrow 0\ ,$$ eq. 3.1-1

où f^{ext} est la force extérieure et f^{int} la force intérieure, définie comme,

$$f^{\int(u)}=\int_V B^T\sigma(\varepsilon(u))\,dV\ .$$ eq. 3.1-2

En utilisant la méthode des éléments finis (MEF), les forces intérieures, f^{int}, sont obtenues à partir du champ de contrainte, qui lui est déterminé, par une loi de comportement, à partir du champ de déformation, le tenseur de déformation discrétise, ε, étant défini comme :

$$\varepsilon=B\,u\ .$$

Selon la méthode Newton on résout le système dans [eq. 3.1-1] avec le processus itératif suivant :

1) $R^{(n)}=R(u^{(n)})$
2) $\Delta u^{(n+1)}=\underbrace{-\left(\frac{\partial R^{(n)}}{\partial u^{(n)}}\right)}_{K^{(n)^{-1}}}R^{(n)}=-K^{(n)^{-1}}R^{(n)}$
3) $u^{(n+1)}=u^{(n)}+\Delta u^{(n)}$, répéter 1) jusqu'à $R^{(n)}<tolérance$.

L'indice (n) signifie que la variable concernée correspond à la n-ième itération, dite *globale,* puisqu'elle concerne le calcul du champ de déplacement, contrairement au processus dit *local*, qui permet de calculer la composante ε_{zz} pour que les contraintes soit planes et lequel sera détaillé dans la suite.

La matrice de rigidité, $K^{(n)}$, se calcule comme :

$$K^{(n)}(u)=\int_V B^T\underbrace{\frac{\partial\sigma^{(n)}}{\partial u^{(n)}}}_{D^{(n)}}B\,dV=\int_V B^T D^{(n)}B\,dV\ ,$$

où $D^{(n)}$ est la matrice tangente correspondante à la loi de comportement utilisée. Pour étendre la méthode de Newton à l'utilisation de lois de comportement quelconques, dans [bib1] on propose de condenser la valeur de ε_{zz} à ce que $\sigma_{zz}<tolérance$ à l'état convergé. On présente ici deux variantes de cette approche : dans l'approche originale, les variables $u^{(k+1)}$ et $\varepsilon_{zz}^{(k+1)}$ sont corrigées simultanément pour chaque itération globale, tandis que dans l'approche modifiée un processus itératif local est rajouté, pour que la condition $\sigma_{zz}<tolérance$ soit satisfaite pour chaque valeur de $u^{(k+1)}$ et non-pas seulement pour l'état convergé. Il est intéressant d'utiliser l'approche modifiée notamment lorsque la convergence sur l'équilibre global est plus rapide que la convergence pour la satisfaction des contraintes planes. Dans certains cas, mais pas toujours, l'approche modifiée peut réduire le nombre d'itérations globales et ainsi accélérer les calculs. En terme d'utilisation, l'approche originale correspond à la valeur du paramètre ITER_CPLAN_MAXI = 1 et l'approche modifiée à ITER_CPLAN_MAXI > 1.

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

En plus du calcul de ε_{zz} au niveau local qui conduit à modifier les contraintes locales, l'approche dite de DEBORST consiste surtout à intervenir au niveau de la matrice de rigidité,

$$\hat{K}^{(n)}(u)=\int_V B^T \hat{D}^{(n)} B\, dV ,$$

où D a été remplacé par $\hat{D}$. Le calcul de $\hat{D}$ aussi bien que de ε_{zz} sont détaillés dans la suite.

3.2 Approche d'origine

L'idée de la méthode due à R. de Borst [bib1] consiste à traiter la condition de contraintes planes non pas au niveau de la loi de comportement mais au niveau de l'équilibre. On obtient ainsi au cours des itérations de l'algorithme de résolution globale de STAT_NON_LINE des champs de contraintes qui tendent vers un champ de contraintes planes au fur et à mesure des itérations :

$$\sigma_{zz}^{(n)} \to 0$$

où n désigne le numéro d'itération de Newton.
On obtient donc la condition de contrainte plane non pas exactement, mais de façon approchée, à convergence des itérations de Newton, pour chaque incrément calculé. On vérifie, comme précisé par la suite, que la composante ci-dessus est inférieure à une tolérance donnée.

La méthode consiste à décomposer les champs (de déformations ou de contraintes) en une partie purement plane (spécifiée par un « chapeau ») et une composante suivant z. On fait alors apparaître explicitement les composantes « zz » dans l'expression des tenseurs de déformations et de contraintes :

$$\varepsilon=\begin{pmatrix}\hat{\varepsilon}\\ \varepsilon_{zz}\end{pmatrix} \quad \text{et} \quad \sigma=\begin{pmatrix}\hat{\sigma}\\ \sigma_{zz}\end{pmatrix}.$$

ainsi que dans l'expression de l'opérateur tangent :

$$D^{(n)}=\frac{\partial \sigma^{(n)}}{\partial \varepsilon^{(n)}}=\begin{pmatrix}D_{11}^{(n)} & D_{12}^{(n)}\\ D_{21}^{(n)} & D_{22}^{(n)}\end{pmatrix} . \qquad \text{eq 3.2-1}$$

Enfin, à chaque itération Newton on corrige les valeurs de déformation,

$$\hat{\varepsilon}^{(n+1)}=\hat{\varepsilon}^{(n)}+\Delta\hat{\varepsilon}^{(n)} \qquad \text{et} \qquad \varepsilon_{zz}^{(n+1)}=\varepsilon_{zz}^{(n)}+\Delta\varepsilon_{zz}^{(n)} .$$

En utilisant [eq. 3.2-1] on peut écrire,

$$\begin{pmatrix}\Delta\hat{\sigma}\\ \Delta\sigma_{zz}\end{pmatrix}=D^{(n)}\begin{pmatrix}\Delta\hat{\varepsilon}\\ \Delta\varepsilon_{zz}\end{pmatrix} \text{, avec } \Delta\sigma_{zz}=\sigma_{zz}^{(n+1)}-\sigma_{zz}^{(n)} ,$$

et on obtient :

$$\sigma_{zz}^{(n+1)}\approx\sigma_{zz}^{(n)}+D_{21}^{(n)}\Delta\hat{\varepsilon}^{(n)}+D_{22}^{(n)}\Delta\varepsilon_{zz}^{(n)}\to 0 , \qquad \text{eq 3.2-2}$$

A partir de [eq. 3.2-2] on peut condenser $\Delta\varepsilon_{zz}^{(n)}$ comme,

$$\Delta\varepsilon_{zz}^{(n)}=-\frac{\sigma_{zz}^{(n)}+D_{21}^{(n)}\Delta\hat{\varepsilon}^{(n)}}{D_{22}^{(n)}} . \qquad \text{eq 3.2-3}$$

C'est exactement la condensation dans [eq 3.2-3] qui nous permet d'utiliser le cadre 2D pour la résolution avec la MEF. Au final, on cherche à corriger les composantes 2D (notées par l'indice [1]), aussi bien pour les contraintes que pour l'opérateur tangent, de sorte que la condition de contraintes planes soit satisfaite.

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

L'algorithme de contraintes planes est entièrement local, appliqué dans le cadre d'une architecture éléments finis globale correspondant à une modélisation 2D équivalente à celle de la déformation plane. On se positionne en un point de Gauss, où on connaît la valeur de la déformation à l'itération (n+1),

$\hat{\varepsilon}^{(n+1)}=\hat{\varepsilon}^{(n)}+\Delta\hat{\varepsilon}^{(n)}$, et on doit calculer les valeurs de contrainte $\hat{\sigma}^{(n+1)}$ et de l'opérateur tangent, $\hat{D}^{(n+1)}$.

Algorithme 1 :

1) Actualiser $\varepsilon_{zz}^{(n)}$,

$$\varepsilon_{zz}^{(n+1)}=\varepsilon_{zz}^{(n)}-\frac{\sigma_{zz}^{(n)}+D_{\mathbf{21}}^{(n)}\Delta\hat{\varepsilon}^{(n)}}{D_{22}^{(n)}}$$

2) Calculer les contraintes et l'opérateur tangent intermédiaires $\sigma^{(n+1)}$, $D^{(n+1)}$ par la loi de comportement 3D, imposant $\varepsilon_{xz}=\varepsilon_{yz}=0$,

$$\sigma^{(n+1)}=\sigma(\hat{\varepsilon}^{(n+1)},\varepsilon_{zz}^{(n+1)}) \quad \text{et} \quad D^{(n+1)}=D(\hat{\varepsilon}^{(n+1)},\varepsilon_{zz}^{(n+1)})$$

3) Calculer la contrainte finale en utilisant la correction intermédiaire de $\tilde{\varepsilon}_{zz}^{(n+1)}$,

$$\Delta\tilde{\varepsilon}_{zz}^{(n+1)}=-\frac{\sigma_{zz}^{(n+1)}}{D_{22}^{(n+1)}} \qquad \text{eq 3.2-4}$$

laquelle nous permet d'écrire la contrainte finale $\hat{\sigma}^{(n+1)}$ comme :

$$\hat{\sigma}^{(n+1)}=\sigma^{(n+1)}+D_{12}^{(n+1)}\Delta\tilde{\varepsilon}_{zz}^{(n+1)}=\sigma^{(n+1)}-\frac{D_{12}^{(n+1)}\sigma_{zz}^{(n+1)}}{D_{22}^{(n+1)}}$$

4) Calculer l'opérateur tangent final :

$\hat{D}^{(n+1)}=D_{11}^{(n+1)}-\frac{D_{12}^{(n+1)}D_{\mathbf{21}}^{(n+1)}}{D_{22}^{(n+1)}}$.

Remarque 1 *: Dans 2) la contrainte σ est calculée en tant que tenseur en 3D,*
$\sigma=(\sigma_{xx}\quad\sigma_{yy}\quad\sigma_{zz}\quad\sigma_{xy}\quad\sigma_{xz}\quad\sigma_{yz})^T$.
Par contre, on n'utilise que $\sigma=(\sigma_{xx}\quad\sigma_{yy}\quad\sigma_{zz}\quad\sigma_{xy})^T$, les composantes σ_{xz} et σ_{yz} n'ayant pas besoin d'être extraites. Ainsi, D est de taille 4×4 .

Remarque 2 *: Dans 3) l'expression de $\Delta\tilde{\varepsilon}_{zz}^{(n+1)}$ est obtenue en supposant que la valeur extrapolée de $\sigma_{zz}^{n+2}\approx0$,*

$$\sigma_{zz}^{(n+2)}\approx\sigma_{zz}^{(n+1)}+D_{22}^{(n+1)}\Delta\tilde{\varepsilon}_{zz}^{(n+1)}=0 \ .$$

Remarque 3 *: Dans 4) on calcule l'opérateur tangent comme :*

$$\hat{D}^{(n+1)}=\frac{\delta\sigma^{(n+1)}}{\delta\hat{\varepsilon}^{(n+1)}}=\frac{\partial\sigma^{(n+1)}}{\partial\hat{\varepsilon}^{(n+1)}}+\frac{\partial\sigma^{(n+1)}}{\partial\varepsilon_{zz}^{(n+1)}}\frac{\partial\varepsilon_{zz}^{(n+1)}}{\partial\hat{\varepsilon}^{(n+1)}}=D_{11}^{(n+1)}-\frac{D_{12}^{(n+1)}D_{\mathbf{21}}^{(n+1)}}{D_{22}^{(n+1)}} \ ,$$

où δ signifie la dérivée totale contrairement à ∂ qui signifie une dérivée partielle. En général, cet opérateur tangent n'est pas cohérent par rapport à $\hat{\sigma}^{(n+1)}$,

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

$$\hat{D}^{(n+1)} \neq \frac{\delta \hat{\sigma}^{(n+1)}}{\delta \hat{\varepsilon}^{(n+1)}} \quad .$$

L'opérateur tangent cohérent peut être obtenu de la manière suivante :

$$\frac{\delta \hat{\sigma}^{(n+1)}}{\delta \hat{\varepsilon}^{(n+1)}} = \frac{\delta \sigma^{(n+1)}}{\delta \hat{\varepsilon}^{(n+1)}} - \frac{D_{12}^{(n+1)}}{D_{22}^{(n+1)}} \frac{\delta \sigma_{zz}^{(n+1)}}{\delta \hat{\varepsilon}^{(n+1)}} \quad ,$$

où

$$\frac{\delta \sigma^{(n+1)}}{\delta \hat{\varepsilon}^{(n+1)}} = \frac{\partial \sigma^{(n+1)}}{\partial \hat{\varepsilon}^{(n+1)}} + \frac{\partial \delta^{(n+1)}}{\partial \varepsilon_{zz}^{(n+1)}} \frac{\partial \varepsilon_{zz}^{(n+1)}}{\partial \hat{\varepsilon}^{(n+1)}} = D_{11}^{(n+1)} - \frac{D_{12}^{(n+1)} D_{\mathbf{21}}^{(n)}}{D_{22}^{(n)}}$$

$$\frac{\delta \sigma_{zz}^{(n+1)}}{\delta \hat{\varepsilon}^{(n+1)}} = \frac{\partial \sigma_{zz}^{(n+1)}}{\partial \hat{\varepsilon}^{(n+1)}} + \frac{\partial \sigma_{zz}^{(n+1)}}{\partial \varepsilon_{zz}^{(n+1)}} \frac{\partial \varepsilon_{zz}^{(n+1)}}{\partial \hat{\varepsilon}^{(n+1)}} = D_{\mathbf{21}}^{(n+1)} - \frac{D_{22}^{(n+1)} D_{\mathbf{21}}^{(n)}}{D_{22}^{(n)}} \quad .$$

Au final, on obtient :

$$\frac{\delta \hat{\sigma}^{(n+1)}}{\delta \hat{\varepsilon}^{(n+1)}} = D_{11}^{(n+1)} - \frac{D_{12}^{(n+1)} D_{\mathbf{21}}^{(n)}}{D_{22}^{(n)}} - \frac{D_{12}^{(n+1)}}{D_{22}^{(n+1)}} \left(D_{\mathbf{21}}^{(n+1)} - \frac{D_{22}^{(n+1)} D_{\mathbf{21}}^{(n)}}{D_{22}^{(n)}} \right) \quad .$$

L'opérateur utilisé, $\hat{D}^{(n+1)}$, tend vers l'opérateur tangent cohérent lors de la convergence du critère des contraintes planes, puisque $D^{(n+1)} \to D^{(n)}$, lorsque $\sigma_{zz}^{(n)} \to 0$, ce qui mène donc à :

$$\hat{D}^{(n+1)} \overset{\sigma_{zz}^{(n+1)} \to 0}{\longrightarrow} \frac{\delta \hat{\sigma}^{(n+1)}}{\delta \hat{\varepsilon}^{(n+1)}} \quad .$$

3.3 Approche modifiée

Dans l'algorithme modifié on propose d'introduire une boucle supplémentaire par rapport au processus décrit ci-dessus pour mieux satisfaire les contraintes planes pour chaque itération globale de Newton, (n). Cette nouvelle boucle englobe les points 2) et 3) de l'algorithme présenté dans 3.2. Ainsi le nouvel algorithme s'écrit comme :

Algorithme 2 :

1) Actualiser $\varepsilon_{zz}^{(n)}$,

$$\varepsilon_{zz}^{(n+1)} = \varepsilon_{zz}^{(n)} - \frac{\sigma_{zz}^{(n)} + D_{\mathbf{21}}^{(n)} \Delta \hat{\varepsilon}^{(n)}}{D_{22}^{(n)}}$$

2) Initialiser la boucle

$$\tilde{\varepsilon}_{zz}^{(k=0,n+1)} = \varepsilon_{zz}^{(n+1)}$$

Début boucle $k = 0, K_{\max}$

3) Calculer les contraintes et l'opérateur tangent intermédiaires $\sigma^{(k,n+1)}$, $D^{(k,n+1)}$ par la loi de comportement 3D,

$$\sigma^{k,(n+1)} = \sigma\left(\hat{\varepsilon}^{(n+1)}, \tilde{\varepsilon}_{zz}^{(k,n+1)}\right) \quad \text{et} \quad D^{(k,n+1)} = D\left(\hat{\varepsilon}^{(n+1)}, \tilde{\varepsilon}_{zz}^{(k,n)}\right)$$

4) Calculer la correction intermédiaire de $\tilde{\varepsilon}_{zz}^{(k,n+1)}$,

$$\Delta \tilde{\varepsilon}_{zz}^{(k,n+1)} = -\frac{\sigma_{zz}^{(k,n+1)}}{D_{22}^{(k,n+1)}} \quad \text{et} \quad \tilde{\varepsilon}_{zz}^{(k+1,n+1)} = \tilde{\varepsilon}_{zz}^{(k,n+1)} + \Delta \tilde{\varepsilon}_{zz}^{(k,n+1)} \quad .$$

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

Terminer la boucle si $|\sigma_{zz}^{(k,n+1)}|<\sigma_{tol}$ ou si $k=K_{\max}$.
(Avec le paramètre σ_{tol} on définit la tolérance sur la valeur de σ_{zz} .)

5) Attribuer au tenseur 2D des contraintes les valeurs convergées :

$$\hat{\sigma}^{(n+1)}=\sigma^{(k,n+1)}+D_{12}^{(k,n+1)}\Delta\tilde{\varepsilon}_{zz}^{(k,n+1)}=\sigma^{(k,n+1)}-\frac{D_{12}^{(k,n+1)}\sigma_{zz}^{(k,n+1)}}{D_{22}^{(k,n+1)}}$$

En principe, le deuxième terme de l'équation ci-dessus peut être omis, si le paramètre du critère de convergence, σ_{tol} , est choisi suffisamment petit.

6) Calculer l'opérateur tangent final :

$$\hat{D}^{(n+1)}=D_{11}^{(n+1)}-\frac{D_{12}^{(n+1)}D_{21}^{(n+1)}}{D_{22}^{(n+1)}} .$$

Remarque 4 *: Dans la version modifiée, l'opérateur tangent,* $\hat{D}^{(n+1)}$ *, est cohérent par rapport à* $\hat{\sigma}^{(n+1)}$ *, contrairement à l'opérateur de la version d'origine (voir la remarque 3), puisque* $|\sigma_{zz}^{(n+1)}|<\sigma_{tol}$.

4 Aspects pratiques d'utilisation

Cette méthode est utilisée automatiquement dès que le comportement choisi n'est pas disponible en contraintes planes, pour des modélisations `C_PLAN` ou de type coque : `COQUE_3D, DKT, TUYAU`. En pratique, cela augmente (automatiquement) de 4 le nombre de variables internes du comportement.

Pour bien converger, il est conseillé de réactualiser la matrice tangente (si possible, à toutes les itérations : `REAC_ITER = 1`, ou toutes les n itérations, avec n petit).

Cette méthode permet donc une grande souplesse d'utilisation par rapport aux comportements : il suffit qu'un comportement soit disponible en axisymétrie ou en déformation plane pour qu'il soit aussi utilisable en contraintes planes.

Comme pour toutes les intégrations de modèles de comportements non linéaire, il est vivement conseillé de donner un critère de convergence petit (laisser la valeur par défaut à 10^{-6} .).

L'avantage de l'approche modifiée est une meilleure satisfaction de la condition de contraintes planes en chaque point de Gauss ($|\sigma_{zz}^{\mathrm{modif},(n)}|<<|\sigma_{zz}^{orig,(n)}|$ pour chaque n). Dans certains cas, elle est indispensable pour faire converger un calcul, notamment pour les lois de comportement adoucissantes.

En revanche, à cause d'une boucle supplémentaire la procédure modifiée est plus coûteuse, surtout parce que la boucle inclut l'appel au module « loi de comportement 3D ». Néanmoins, le surcoût à cause des calculs locaux plus lourds peut être compensé par un gain au niveau du nombre d'itérations globales de Newton, qui est le plus souvent moins élevé pour l'algorithme modifié. Ce gain en nombre d'itérations globales n'est pas garanties, ce qui a pour conséquence que la boucle itérative supplémentaire n'est pas activée par défaut (`ITER_CPLAN_MAXI`=1). On a également observé que du moment où on choisit `ITER_CPLAN_MAXI` > 1, il est préférable d'utiliser `ITER_CPLAN_MAXI` > 5.

5 Bibliographie

1 R de Borst "The zero normal stress condition in plane stress and shell elastoplasticity " Communications in applied numerical methods, Vol 7, 29-33 (1991)

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

6 Historique des versions du document

Version Aster	Auteur(s) ou contributeur(s), organisme	Description des modifications
5.4	J.M. PROIX, E. LORENTZ	Version initiale.
9.2	D. MARKOVIC	Rajout de la boucle itérative interne pour améliorer la convergence.
11.2	J.M.PROIX	fiche 18398

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