Intégration implicite et explicite des relations de comportements non linéaires

Résumé

Ce document décrit deux méthodes d'intégration (une implicite et une explicite) pour la résolution de problèmes avec comportements non linéaires dans les opérateurs STAT_NON_LINE [R5.03.01] et DYNA_NON_LINE [R5.05.05].

La méthode numérique implicite présentée est celle de Newton, avec ou sans recherche linéaire.

La méthode numérique explicite est celle de Runge Kutta d'ordre 2 à pas adaptatif.

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

Table des Matières

1 Généralités sur l'intégration des lois de comportement....3
1.1 Algorithme de résolution du problème quasi-statique....3
1.2 But de l'intégration....4
1.3 Méthodes de résolution....4
2 Intégration explicite d'une relation de comportement incrémentale....4
2.1 Utilisation d'une méthode explicite....5
2.2 Contrôle du pas de temps....5
2.3 Influence sur l'étape globale....6
3 Intégration implicite d'une relation de comportement incrémentale....8
3.1 Intégration implicite par la méthode de Newton....8
3.2 Opérateur tangent....9
3.3 Intégration implicite par la méthode de Newton avec recherche linéaire....10
3.3.1 Présentation de l'algorithme général de recherche linéaire....10
3.3.2 Mise en œuvre de la recherche linéaire dans PLASTI....12
4 Implantation d'un nouveau modèle de comportement....13
4.1 Méthodes possibles....13
4.2 Implantation d'un nouveau modèle de comportement par la méthode explicite....13
4.3 Implantation complète d'un nouveau comportement dans PLASTI....13
4.4 Implantation implicite facile d'un nouveau modèle de comportement dans l'environnement PLASTI....14
5 Bibliographie....15
6 Description des versions du document....16

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

1 Généralités sur l'intégration des lois de comportement

1.1 Algorithme de résolution du problème quasi-statique

Quand le comportement des matériaux composant la structure n'est pas linéaire, la résolution de l'équilibre global de la structure à chaque instant est un problème non linéaire dont la formulation variationnelle dans le cas des petites déformations peut s'écrire, dans le cas quasi-statique, sous la forme suivante, déduite de l'expression du principe des travaux virtuels (cf. [R5.03.01]) :

$$\begin{cases} \mathbf{v}^T.\mathbf{L}^{\text{int}}(\mathbf{u},t) = \mathbf{v}^T.\mathbf{L}^{\text{ext}}(t) \quad \forall \mathbf{v} \text{ tel que } \mathbf{B}.\mathbf{v}=0 \\ \mathbf{B}.\mathbf{u}=\mathbf{u}^d(t) \end{cases} \tag{1}$$

Où :

- t représente la variable d'instant
- $\mathbf{u}$ est le champ de déplacement pris à partir d'une configuration de référence
- $\mathbf{v}$ est le champ de déplacement virtuel cinématiquement admissible
- la relation $\mathbf{B}.\mathbf{u}=\mathbf{u}^d(t)$ correspond aux conditions aux limites imposées en déplacements
- $\mathbf{L}^{\text{ext}}(t)$ est le chargement mécanique extérieur auquel est soumis la structure
- $\mathbf{L}^{\text{int}}$ représente les forces internes du problème de mécanique quasi-statique non linéaire. $\mathbf{L}^{\text{int}}(\mathbf{u},t)$ est relié au champ de contraintes $\boldsymbol{\sigma}$ par l'opérateur du travail des déformations virtuelles $\mathbf{Q}^T$ suivant la relation suivante:

$$\mathbf{L}^{\text{int}}(\mathbf{u},t)=\mathbf{Q}^T(\mathbf{u}).\boldsymbol{\sigma} \tag{2}$$

La dualisation des conditions aux limites de Dirichlet $\mathbf{B}.\mathbf{u}=\mathbf{u}^d(t)$ conduit au problème suivant [R3.03.01] :

$$\begin{cases} \mathbf{L}^{\text{int}}(\mathbf{u},t)+\mathbf{B}^T.\boldsymbol{\lambda}=\mathbf{L}^{\text{ext}}(t) \\ \mathbf{B}.\mathbf{u}=\mathbf{u}^d(t) \end{cases} \tag{3}$$

La formulation du problème quasi-statique discrétisé consiste à exprimer l'équilibre de la structure pour une suite d'instants de calcul $(t_i)_{1\le i\le I}$ qui paramètrent le chargement, on notera les quantités à l'instant t_i par l'indice i (par exemple $\mathbf{L}^{\text{int}}(\mathbf{u_i},t_i)=\mathbf{L_i}^{\text{int}}$) :

$$\begin{cases} \mathbf{L}_i^{\text{int}}+\mathbf{B}^T.\boldsymbol{\lambda}_i=\mathbf{L}_i^{\text{ext}} \\ \mathbf{B}.\mathbf{u}_i=\mathbf{u}_i^d \end{cases} \tag{4}$$

Ce problème non linéaire est résolu par une méthode de Newton (ou quasi-Newton), cf [R5.03.01]).
Le système à résoudre s'écrit alors, pour chaque itération n de la phase de correction :

$$\begin{cases} \mathbf{K}_i^n.\delta\mathbf{u}_i^{n+1}+\mathbf{B}^T.\delta\boldsymbol{\lambda}_i^{n+1}=\mathbf{L}_i^{\text{méca}}-\mathbf{L}_i^{\text{int},n}-\mathbf{B}^T.\boldsymbol{\lambda}_i^n \\ \mathbf{B}.\delta\mathbf{u}_i^{n+1}=0 \end{cases} \tag{5}$$

Le vecteur des forces internes $\mathbf{L}_i^{\text{int},n}$ est calculé à partir des contraintes $\boldsymbol{\sigma}_i^n$, celles-ci étant calculées à partir des déplacements $\mathbf{u}_i^n$ par l'intermédiaire de la relation de comportement du matériau. En fait, dans le cas des comportements incrémentaux, $\boldsymbol{\sigma}_i^n$ est calculé à partir de

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

$(\sigma_{i-1}, \alpha_{i-1})$ et de l'incrément de déformation $\varepsilon(\Delta \mathbf{u}_i^n)$ induit par l'incrément de déplacement depuis le début du processus itératif.

1.2 But de l'intégration

Ce schéma nécessite, à partir de l'estimation des déplacements à l'itération n, de calculer en chaque point de Gauss :

- le tenseur des contraintes σ_i^n qui vérifie la loi de comportement, et qui permet de calculer $\mathbf{L}_i^{\text{int},n}$ et les variables internes spécifiques à chaque comportement,
- l'opérateur de comportement tangent cohérent $\left(\frac{\partial \sigma}{\partial \epsilon}\right)_i^n$, pour obtenir la matrice tangente :

$$\mathbf{K}_i^n = \sum_e \mathbf{K}_{e,i}^n \text{ avec } \mathbf{K}_{e,i}^n = \int_{\Omega_e} \mathbf{Q}^T \left(\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\epsilon}}\right)_i^n \mathbf{Q}\, d\,\Omega_e$$

1.3 Méthodes de résolution

Pour chaque élément fini, en chaque point d'intégration, il faut donc intégrer le comportement, c'est à dire calculer le tenseur des contraintes et les variables internes solutions du problème ainsi que l'opérateur tangent.

Nous présentons dans les paragraphes suivant deux algorithmes généraux permettant le calcul de la solution : soit la résolution par une méthode de Newton, basée sur une discrétisation implicite des équations, soit par une méthode de Runge-Kutta d'ordre 2 avec contrôle d'erreur, basée sur une discrétisation explicite de ces équations.

D'autres méthodes de résolution, spécifiques à chaque comportement, sont également employées. Elle sont fondées sur une optimisation du système d'équations non linéaires, permettant de réduire le nombre d'équations à résoudre, bien souvent à une seule équation. La méthode utilisée alors est une simple recherche de zéro de fonction (voir par exemple [R5.03.02], [R5.03.04], [R5.03.16]).

Pour tous ces algorithmes, il est possible, en cas de non convergence, de procéder à un re-découpage local du pas de temps. ; pour cela, on adopte une interpolation linéaire du déplacement et des variables de commande (par exemple la température) au cours de l'intervalle de temps, ce qui conduit, pour tout incrément de temps à :

$$\forall \tau \in [t_{i-1}, t_i] \quad T(\tau) = T_{i-1} + \frac{\tau - t_{i-1}}{\Delta t}[T - T_{i-1}] \text{ et } \varepsilon_{ij}^{\mathbf{k}}(\tau) = \varepsilon_{ij}(\mathbf{u}_{i-1}) + \frac{\tau - t_{i-1}}{\Delta t}[\varepsilon_{ij}^{\mathbf{k}}(\mathbf{u}_i^n) - \varepsilon_{ij}(\mathbf{u}_{i-1})]$$

2 Intégration explicite d'une relation de comportement incrémentale

Pour résoudre le système non linéaire relatif à la loi de comportement,on peut utiliser une méthode d'intégration explicite de type Runge Kutta d'ordre 2 (option 'RUNGE_KUTTA' de l'opérande ALGO_INTE du mot-clé facteur COMPORTEMENT) pour les problèmes incrémentaux de comportement non linéaire.

Les relations concernées actuellement sont :

VISCOCHAB	Comportement élasto-viscoplastique de Chaboche, à deux centres, avec effet de mémoire et de restauration
VENDOCHAB	Comportement viscoplastique avec endommagement de Chaboche
POLYCRISTAL	Comportement élasto-viscoplastique polycristallin homogénéisé

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

MONOCRISTAL	Comportements viscoplastiques monocristallins des métaux
HAYHURST	Comportement viscoplastique avec endommagement de Hayhurst

La relation POLYCRISTAL n'est accessible que par la méthode explicite présentée ci-après.

Ce type d'intégration permet d'implanter très facilement un nouveau modèle de comportement [bib2]. On décrit le calcul du champ de contraintes à partir d'un incrément de déformation donné en suivant l'évolution des variables internes.

2.1 Utilisation d'une méthode explicite

On suppose que le système à résoudre peut s'exprimer sous la forme :

$$\frac{dY}{dt}=F(Y,t;\Delta\varepsilon_i^n,\sigma_{i-1})$$

où Y représente l'ensemble des variables internes du modèle, y compris la variation de déformation plastique $\Delta\varepsilon^p$.
Leurs dérivées en temps ne sont supposées dépendre que des variables internes et des contraintes à l'instant précédent, et de l'accroissement de déformation totale fourni par l'algorithme global de Newton $\Delta\varepsilon_i^n$.

Cette hypothèse repose en particulier sur la relation linéaire (loi de Hooke) entre le tenseur des contrainte à l'instant actuel et la partie élastique du tenseur des déformations : $\Delta\boldsymbol{\sigma}=\mathbf{A}(\Delta\boldsymbol{\varepsilon}_\mathbf{i}^\mathbf{n}-\Delta\boldsymbol{\varepsilon}^\mathbf{p})$, ce qui permet d'exprimer σ en fonction de $\Delta\varepsilon^p$ dans les équations différentielles $\frac{dY}{dt}=F(Y,t;\boldsymbol{\sigma}(\Delta\varepsilon_i^n,\sigma_{i-1},Y))$

L'une des techniques les plus simples à mettre en œuvre pour résoudre ces équations différentielles est l'utilisation de méthodes explicites. Pour qu'elles restent efficaces numériquement, il est indispensable de leur associer un contrôle de pas automatique pour conserver un bon compromis coût précision.

Les méthodes de Runge et Kutta explicites et emboîtées [bib1], [bib2] sont sans doute les schémas d'intégration les plus simples respectant ces critères. Leur principe est d'associer deux schémas d'intégration d'ordre différent pour contrôler le pas de temps en fonction d'une précision requise. En fonction de l'ordre d'intégration choisi, plusieurs algorithmes sont disponibles et le schéma le plus simple est une méthode d'ordre deux.

On intègre selon le schéma suivant :

- $Y_{t+h}=Y^{(2)}$ si le critère de précision est satisfait ;
- $Y^{(2)}=Y+\frac{h}{2}[F(Y,t)+F(Y^{(1)},t+h)]$ avec $Y^{(1)}=Y+hF(Y,t)$

La différence entre $Y^{(2)}$ (schéma d'ordre 2) et $Y^{(1)}$ (schéma d'ordre 1, Euler) fournit une estimation de l'erreur d'intégration et permet de contrôler la taille du pas de temps h qui est initialisé à Δt_i pour la première tentative. Ainsi, la méthode reste efficace si le comportement reste élastique au cours de l'incrément et on dispose naturellement de sous pas locaux indépendants de l'incrément global pour intégrer, avec une meilleure précision, l'évolution des variables internes aux points de Gauss où la non linéarité du comportement est la plus significative.

2.2 Contrôle du pas de temps

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

La stratégie du contrôle du pas est définie à partir d'une norme de l'écart entre les deux méthodes d'intégration : $||Y^{(2)}-Y^{(1)}||$ et de la précision requise par l'utilisateur η (mot-clé : RESI_INTE_RELA). Le critère retenu est le suivant, où l'on note $Y=(y_{1,}y_{2,}...y_N)$:

$$\delta Y(t)=\sup_{j=1...N}\left\{\frac{|y_j^{(2)}-y_j^{(1)}|}{max[\varepsilon,|y_j(t)|]}\right\}<\eta$$

Le paramètre ε est fixé à 0,001. La précision d'intégration souhaitée η doit être cohérente avec le niveau de précision requis pour l'étape globale. La valeur par défaut $\eta=10^{-6}$ s'avère souvent pénalisante en termes de temps calcul. Elle peut être souvent être augmentée d'un ordre ou deux, en particulier pour les modèles élasto-visco-plastiques, sans réelle perte de précision (à voir au cas par cas).

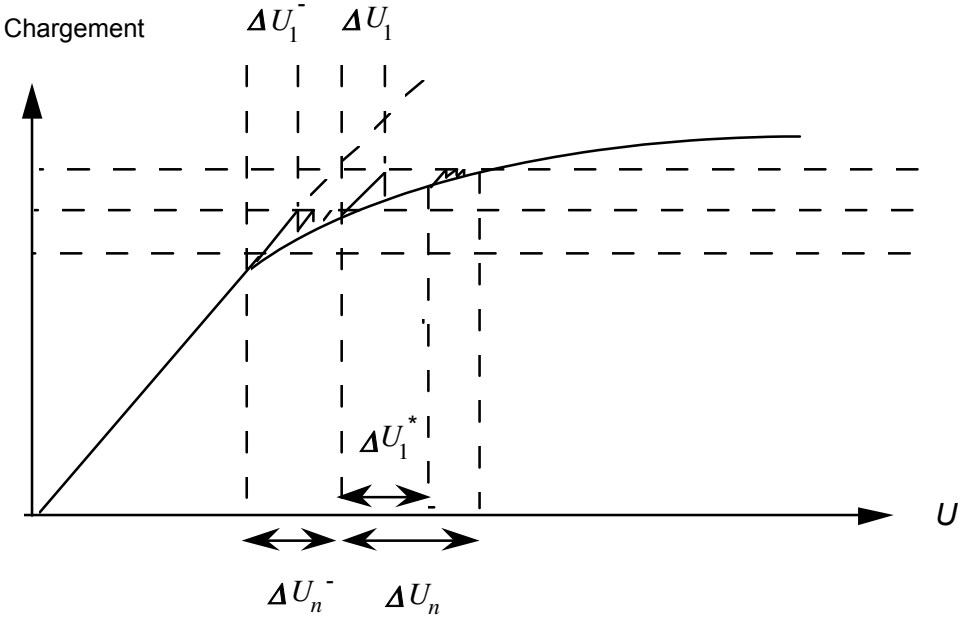
Si le critère n'est pas vérifié, le pas de temps est re-découpé selon une méthode heuristique. Lorsque le pas de temps devient trop faible ($h<$ précision machine), un code retour est émis, et le re-découpage global du pas de temps est activé, si demandé par l'utilisateur.

2.3 Influence sur l'étape globale

Dans sa version actuelle, la méthode ne fournit pas de matrice tangente ; son évaluation est toutefois possible (mais coûteuse) par une technique de perturbations (activée sous COMPORTEMENT le mot-clé TYPE_MATR_TANG='PERTURBATION') . Si on utilise la matrice élastique pour l'équilibre global, le nombre d'itérations de Newton pour converger peut devenir très important, ce qui amène souvent à nombre d'incréments plus important pour que la résolution globale soit facilitée.

Pour améliorer la convergence, il est possible d'initialiser les itérations de Newton à partir de la solution précédemment calculée : au lieu de considérer l'incrément de déplacement pour la première itération à l'instant courant (obtenu via la matrice élastique), on initialise l'incrément de déplacement recherché pour la première itération ΔU_1 à partir de la valeur ΔU_n^-, solution à l'incrément précédent (cf. figure page suivante). Le mot-clé PREDICTION='EXTRAPOLE' (sous NEWTON) [U4.51.03] permet d'initialiser avec l'incrément convergé du pas précédent (pondéré par le rapport des pas de temps). Cette estimation est projetée sur le champ des déplacements cinématiquement admissibles pour que la solution finale vérifie bien les conditions aux limites de Dirichlet.

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



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

3 Intégration implicite d'une relation de comportement incrémentale.

Pour résoudre les systèmes d'équations non linéaires relatifs aux problèmes incrémentaux de comportement non linéaire (opérande RELATION du mot clé facteur COMPORTEMENT). il est possible d'utiliser une méthode d'intégration implicite de type Newton-Raphson, à laquelle on peut adjoindre une étape de recherche linéaire (respectivement option 'NEWTON' et option 'NEWTON_RELI' de l'opérande ALGO_INTE du mot-clé facteur CONVERGENCE).

Pour plus de détails consulter le document [U4.51.03] du manuel utilisateur.

Les relations concernées sont par exemple :

VISCOCHAB	Comportement élasto-viscoplastique de J.L. Chaboche, à deux centres, avec effet de mémoire et de restauration
MONOCRISTAL	Comportements viscoplastiques monocristallins des métaux
IRRAD3M	Comportement élasto-plastique sous irradiation des aciers inoxydables
...	

Les relations VISCOCHAB et MONOCRISTAL peuvent également être intégrées par la méthode explicite exposée au §2.

3.1 Intégration implicite par la méthode de Newton

L'environnement PLASTI permet dans *Code_Aster* d'intégrer par une méthode de Newton, de façon systématique, des modèles de comportement non linéaire. La démarche pour résoudre les équations du modèle peut se schématiser de la manière suivante : connaissant toutes les variables (contraintes, variables internes) à l'instant précédent t_{i-1}, et l'accroissement de déformation totale fourni au point considéré par l'algorithme global de Newton $\boldsymbol{\varepsilon}(\Delta \mathbf{u}_i^n)$, noté aussi $\Delta \varepsilon_i^n$, il faut trouver la solution du système :

$$f(\Delta Y)=0=\begin{bmatrix} g(\Delta y) \\ l(\Delta y) \\ i(\Delta y) \\ j(\Delta y) \\ k(\Delta y) \end{bmatrix} \quad \text{avec} \quad \Delta y=\begin{Bmatrix} \Delta \boldsymbol{\sigma} \\ \Delta \boldsymbol{\varepsilon}^{\mathbf{p}} \\ \Delta X_1 \\ \Delta X_2 \\ \Delta p \end{Bmatrix} = y_i - y_{i-1}\,.$$

La première équation représente par exemple le relation contrainte-déformation élastique $g(\Delta y)=\Delta \boldsymbol{\sigma}-\mathbf{A}(\Delta \boldsymbol{\varepsilon}_{\mathbf{i}}^{\mathbf{n}}-\Delta \boldsymbol{\varepsilon}^{\mathbf{p}})$, la deuxième exprimant l'écoulement plastique, et les suivantes correspondent aux lois d'évolution des différentes variables internes, vectorielles ou scalaires, notées ici ΔX_1, ΔX_2 et Δp.

Le nombre d'équations à résoudre varie évidemment en fonction du comportement (voir par exemple les comportements MONOCRISTAL [R5.03.11], VISCOCHAB [R5.03.12], IRRAD3M [R5.03.23]).

On résout ce système par la méthode de Newton proposée dans l'environnement PLASTI, soit :

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

$$\begin{cases} \dfrac{\partial f^l}{\partial \Delta y_k} d(\Delta y_k) = - f^l(\Delta y_k) \\ \Delta y_{k+1} = \Delta y_k + d(\Delta y_k) \end{cases}$$ en itérant en k jusqu'à convergence.

Il reste à calculer la matrice jacobienne du système : $\dfrac{df}{dy_k}$. Elle a la forme suivante :

$$J = \begin{bmatrix} \dfrac{\partial g}{\partial \Delta \sigma} & \dfrac{\partial g}{\partial \Delta \varepsilon^p} & \dfrac{\partial g}{\partial \Delta X_1} & \dfrac{\partial g}{\partial \Delta X_2} & \dfrac{\partial g}{\partial \Delta p} \\ \dfrac{\partial l}{\partial \Delta \sigma} & \dfrac{\partial l}{\partial \Delta \varepsilon^p} & \dfrac{\partial l}{\partial \Delta X_1} & \dfrac{\partial l}{\partial \Delta X_2} & \dfrac{\partial l}{\partial \Delta p} \\ \dfrac{\partial i}{\partial \Delta \sigma} & \dfrac{\partial i}{\partial \Delta \varepsilon^p} & \dfrac{\partial i}{\partial \Delta X_1} & \dfrac{\partial i}{\partial \Delta X_2} & \dfrac{\partial i}{\partial \Delta p} \\ \dfrac{\partial j}{\partial \Delta \sigma} & \dfrac{\partial j}{\partial \Delta \varepsilon^p} & \dfrac{\partial j}{\partial \Delta X_1} & \dfrac{\partial j}{\partial \Delta X_2} & \dfrac{\partial j}{\partial \Delta p} \\ \dfrac{\partial k}{\partial \Delta \sigma} & \dfrac{\partial k}{\partial \Delta \varepsilon^p} & \dfrac{\partial k}{\partial \Delta X_1} & \dfrac{\partial k}{\partial \Delta X_2} & \dfrac{\partial k}{\partial \Delta p} \end{bmatrix}$$

Le critère d'arrêt des itérations porte, suivant les comportements, sur la nullité du résidu : $\dfrac{f(y_k)}{f(y_0)} < \varepsilon$

ou la stationnarité de la solution : $|y_{k+1} - y_k| < \varepsilon$.

Les valeurs initiales y_0 sont choisies nulles par défaut, mais peuvent prendre des valeurs particulières suivant les comportements.

Plus précisément, les tests d'arrêt et les valeurs initiales sont :

Comportements	Critère d'arrêt	Initialisation
VISCOCHAB	$\|y_{k+1} - y_k\| < \varepsilon \|y_{k+1}\|$	Solution explicite et modification de l'initialisation en cas de mauvaise convergence
MONOCRISTAL	$\|f(y_k)\| < \varepsilon \|f(y_0)\|$	Solution nulle
IRRAD3M	$\|f(y_k)\| < \varepsilon \|f(y_0)\|$	Solution élastique

La méthode utilisée permet un re-découpage local du pas de temps, soit systématique, soit en cas de non convergence.

3.2 Opérateur tangent

L'opérateur tangent peut être obtenu directement à partir du système précédent. En effet, le système formé des équations du modèle à convergence, $\left(f(\Delta y_{CV}) = 0\right)$ est vérifié en fin d'incrément. Pour une petite variation de f, en considérant cette fois l'accroissement de déformation totale $\Delta \varepsilon_i^n$ comme variable et non comme paramètre, le système reste à l'équilibre et on vérifie $df = 0$, c'est-à-dire :

$$\frac{\partial f}{\partial \Delta \boldsymbol{\sigma}} \delta \Delta \boldsymbol{\sigma} + \frac{\partial f}{\partial \Delta \boldsymbol{\varepsilon}_\mathbf{i}^\mathbf{n}} \delta \Delta \boldsymbol{\varepsilon}_\mathbf{i}^\mathbf{n} + \frac{\partial f}{\partial \Delta \boldsymbol{\varepsilon}^\mathbf{p}} \delta \Delta \boldsymbol{\varepsilon}^\mathbf{p} + \frac{\partial f}{\partial \Delta X_1} \delta \Delta X_1 + \frac{\partial f}{\partial \Delta X_2} \delta \Delta X_2 + \frac{\partial f}{\partial \Delta p} \delta \Delta p = 0$$

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

Ce système peut encore s'écrire :

$$\frac{\partial f}{\partial y}\delta(y)=X,\ \text{avec}\ \ Y=\begin{bmatrix}\Delta\boldsymbol{\sigma}\\ \Delta Z_s\end{bmatrix}\ \text{et}\ \ X=\begin{bmatrix}\delta\Delta\boldsymbol{\varepsilon}_{\mathbf{i}}^{\mathbf{n}}\\ 0\end{bmatrix}$$

où ΔZ représente les inconnues complémentaires de $\Delta\boldsymbol{\sigma}$ dans y .

En écrivant la matrice jacobienne sous la forme :

$$J.\delta Y=\begin{bmatrix}Y_0 & Y_1\\ Y_2 & Y_3\end{bmatrix}\begin{bmatrix}\Delta\boldsymbol{\sigma}\\ \Delta Z\end{bmatrix}$$

puis en opérant par éliminations et substitutions successives, l'opérateur tangent recherché peut s'écrire directement :

$$\left(\frac{\partial\Delta\boldsymbol{\sigma}}{\partial\Delta\boldsymbol{\varepsilon}}\right)_i^n=\left(Y_0-Y_1Y_3^{-1}Y_2\right)^{-1}$$

Les équations précédentes montrent que l'on est conduit à réutiliser la même matrice jacobienne J que précédemment pour évaluer l'opérateur tangent.

3.3 Intégration implicite par la méthode de Newton avec recherche linéaire

Afin d'améliorer la robustesse de la méthode de Newton, on offre la possibilité d'activer la recherche linéaire.

3.3.1 Présentation de l'algorithme général de recherche linéaire

Soit un système d'équations non-linéaires à résoudre : $f(x)=0$

On associe à f une fonctionnelle F.

La méthode de Newton avec recherche linéaire consiste à trouver à chaque itération un pas d'avancement suivant la direction de descente qui minimise (au sens fort ou au sens faible) la fonctionnelle. Cela se ramène en un enchainement des étapes suivantes :

- calcul de la direction de descente : $d_k=-[\nabla f(x_k)]^{-1}.f(x_k)$
- calcul du pas d'avancement : ρ tel que $F(x_k+\rho.d_k)<F(x_k)$, le meilleur étant le ρ qui minimise la fonctionnelle $\rho=\text{argmin}[F(x_k+\rho.d_k)]$
- actualisation des inconnues : $x_{k+1}=x_k+\rho\,d_k$

On associe à f la fonctionnelle $F(\rho)=\frac{1}{2}\|f(x_k+\rho.d_k)\|^2$.

Contrairement à la recherche linéaire de Newton global où l'on détermine ρ par la résolution de $F'(\rho)=0$, on cherche ici un $\rho\in[0,1]$, $\rho\neq 0$ qui diminue suffisamment la fonctionnelle F. On ne va pas chercher un pas d'avancement supérieur au pas de Newton ($\rho=1$). Le principe est décrit dans [1].

Intervalle de confiance :

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

Pour savoir si un ρ diminue suffisamment la fonctionnelle F, on utilise la règle d'Armijo. Celle-ci permet de caractériser un intervalle de confiance en disant que ρ convient si $F(\rho)\leq F(0)+\omega F'(0)\rho$ où ω est un réel fixé inférieur à 1. Cette expression peut se représenter graphiquement par la Figure 3.3.1-a : la courbe en pointillés bleu est le graphe de la fonctionnelle $h(\rho)=(7\rho-4)^4+(7\rho-5)^3$, celle est violet est la tangente à l'origine, de pente $F'(0)$ et celle en pointillés rouge est la droite de pente $\omega F'(0)$ avec $\omega=0,1$. Dans cet exemple, ρ diminue suffisamment la fonctionnelle si $\rho\in[0;0,871]$.

Principe du calcul de ρ :

On commence d'abord par essayer le pas de Newton ($\rho=1$). Si ce pas satisfaisait la règle d'Armijo, il est retenu et on continue les itérations de Newton. Sinon, on fait une interpolation quadratique connaissant $F(0)$, $F'(0)$ et $F(1)$. On détermine alors un nouveau $\rho=\rho_q$. Si ce pas satisfaisait la règle d'Armijo, il est retenu et on continue les itérations de Newton. Sinon, on réalise une interpolation cubique connaissant $F(0)$, $F'(0)$, $F(1)$ et $F(\rho_q)$. On détermine alors un nouveau $\rho=\rho_c$. Si ce pas satisfaisait la règle d'Armijo, il est retenu et on continue les itérations de Newton. Sinon, on continue les interpolations cubiques entre 0 et les 2 derniers ρ. Le paragraphe ci-dessous détaille le calcul de ρ.

Détails du calcul de ρ :

On commence d'abord par essayer le pas de Newton ($\rho=1$). À ce stade, on connait $F(0)=\frac{1}{2}\|f(x_k)\|^2$ et $F'(0)=-\|f(x_k)\|^2$.

On calcule $F(1)=\frac{1}{2}\|f(x_k+d_k)\|^2$.

Si la règle d'Armijo n'est pas satisfaite, c'est-à-dire si $F(1)>F(0)+\omega F'(0)$, alors on on modélise $F(\rho)$ par un polynôme d'ordre 2 (interpolation quadratique) :

$$q(\rho)=[F(1)-F(0)-F'(0)]\rho^2+F'(0)\rho+F(0)$$

Le minimum de ce polynôme est obtenu pour $\rho_q=\frac{-F'(0)}{2[F(1)-F(0)-F'(0)]}$. Notons que si $F(1)$ est beaucoup plus grand que $F(0)$, alors le minimum du polynôme est atteint en 0. Or on ne souhaite pas avoir un pas d'avancement trop petit, sous peine de devoir faire un très grand nombre d'itérations de Newton. Pour cela, on impose un $\rho_{min}>0$ (par exemple $\rho_{min}=0,1$). On peut aussi imposer une borne supérieure ρ_{max} (par exemple $\rho_{max}=0,6$). Ainsi, si $\rho_q<\rho_{min}$ alors $\rho_q=\rho_{min}$ et si $\rho_q>\rho_{max}$ alors $\rho_q=\rho_{max}$.

Ensuite on calcule $F(\rho_q)=\frac{1}{2}\|f(x_k+\rho_q.d_k)\|^2$.

Si la règle d'Armijo n'est pas satisfaite, c'est-à-dire si $F(\rho_q)>F(0)+\omega F'(0)\rho_q$, alors on on modélise $F(\rho)$ par un polynôme d'ordre 3 (interpolation cubique) :

$$c(\rho)=a\rho^3+b\rho^2+F'(0)\rho+F(0)$$

avec

$$\begin{bmatrix} a \\ b \end{bmatrix}=\frac{1}{\rho_1-\rho_2}\begin{bmatrix} \frac{1}{\rho_1^2} & \frac{-1}{\rho_2^2} \\ \frac{-\rho_2}{\rho_1^2} & \frac{\rho_1}{\rho_2^2} \end{bmatrix}\begin{bmatrix} F(\rho_1)-F(0)-F'(0)\rho_1 \\ F(\rho_2)-F(0)-F'(0)\rho_2 \end{bmatrix}$$

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

où ρ_1 et ρ_2 sont les deux dernières valeurs échouées de ρ.

Le point minimal de $c(\rho)$ est :

$$\rho_c = \frac{-b+\sqrt{b^2-3a\,F'(0)}}{3a}.$$

On continue les interpolations cubiques tant que la règle d'Armijo n'est pas satisfaite. Dans la pratique, on limite à 2 le nombre d'interpolations cubiques.

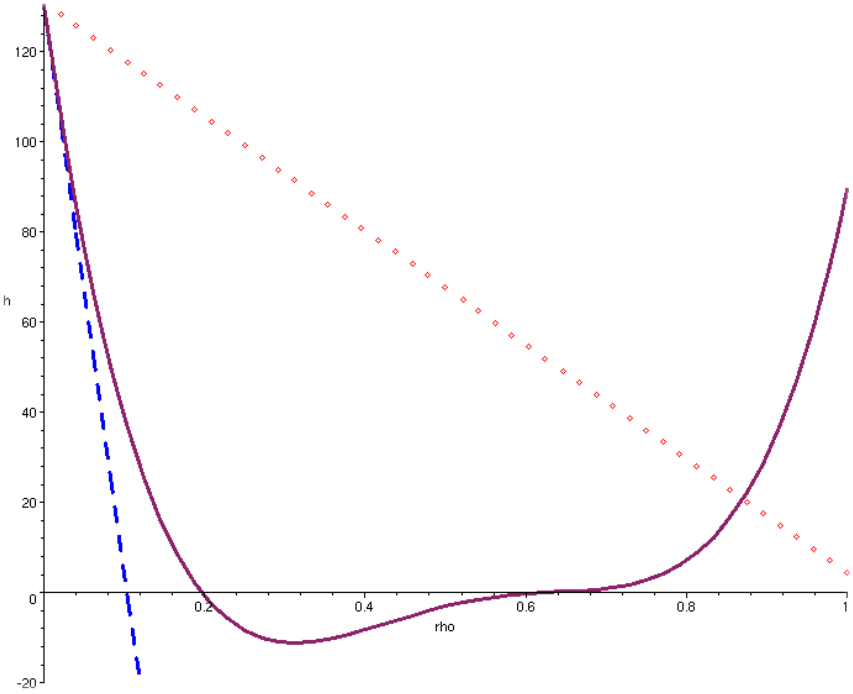


Figure 3.3.1-a : illustration de la règle d'Armijo

3.3.2 Mise en œuvre de la recherche linéaire dans PLASTI

Le système à résoudre est :

$$f(\Delta y)=0$$

On fait les itérations de Newton suivantes :

- mise à jour des variables à l'instant « + »
- calcul du résidu $f(\Delta y_i)=0$
- calcul de la Jacobienne $\frac{\partial f}{\partial y}(\Delta y_i)$
- résolution du système linéaire $\frac{\partial f}{\partial y}(\Delta y_i).\delta\Delta y_i = -f(\Delta y_i)$
- recherche linéaire : calcul de ρ
- actualisation de la solution : $\Delta y_{i+1}=\Delta y_i+\rho\,\delta\Delta y_i$

Pour l'étape de recherche linéaire :

On introduit la fonctionnelle $F(\rho)=\frac{1}{2}\|f(y_i+\rho.\delta\Delta y_i)\|^2$.

Les paramètres de la recherche linéaire sont fixés en dur :

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

Paramètre de la règle d'Armijo	$\omega = 0,1$
Borne min pour le rabattement	$\rho_{min} = 0,1$
Borne max pour le rabattement	$\rho_{max} = 0,5$
Nombre d'itérations d'interpolations cubiques	$it_{max} = 2$

Remarque :

Certains auteurs prennent des valeurs encore plus petites pour ω (par exemple 10^{-4}).

4 Implantation d'un nouveau modèle de comportement

4.1 Méthodes possibles

En utilisant les architectures précédentes, un nouveau modèle de comportement peut être développé relativement facilement :

– la méthode la plus simple est l'intégration explicite, par la méthode de RUNGE_KUTTA (voir §2) ;

– l'intégration implicite complète par le méthode de NEWTON dans l'environnement PLASTI (voir §3) demande un peu plus de développements, en particulier le calcul de la matrice jacobienne ;

– il est toutefois possible d'utiliser directement le développement réalisé pour l'intégration explicite (2 routines) , sans routine supplémentaire, pour effectuer des calculs avec intégration implicite, la matrice jacobienne étant alors calculée par perturbation.

4.2 Implantation d'un nouveau modèle de comportement par la méthode explicite

Pour implanter un modèle de comportement XXX, utilisable avec ALGO_INTE_='RUNGE_KUTTA', il suffit d'écrire 2 routines :

- une routine récupération des données matériau, appelée par la routine d'aiguillage LCMATE
- une routine calculant les dérivés temporelles des variables internes, appelée par la routine d'aiguillage LCDVIN.

La mise à jour des catalogues concerne uniquement DEFI_MATERIAU et C_COMPORTEMENT, ainsi que le catalogue de la loi de comportement (cf. [D5.04.01]).

4.3 Implantation complète d'un nouveau comportement dans PLASTI

Sans faire une description informatique précise des développements (voir à ce propos [D5.04.01]), on peut toutefois décrire l'architecture de PLASTI et les lieux où intervenir pour intégrer un nouveau comportement.

Architecture générale de PLASTI :

• Appel aux routines récupérant les paramètres matériau (à l'instant t_i car dépendant éventuellement des variables de commandes) : LCMATE => routines spécifiques de récupération du matériau, communes avec RUNGE_KUTTA (§3)

• Intégration élastique sur Δt : LCELAS => par défaut, élasticité linéaire isotrope ou anisotrope ; si nécessaire, appel à des routines spécifiques pour le calcul élastique XXXELA.

• Évaluation du seuil pour l'intégration élastique : LCCNVX => appel aux routines d'évaluation du seuil XXXCVX

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

• Si $seuil > 0$, alors résolution par Newton : LCPLAS qui appelle LCPLNL, qui réalise la boucle de Newton. Sa structure est la suivante :

• Initialisation : LCINIT => par défaut initialisation à zéro : $y^0 = 0$; si nécessaire, appel aux routines spécifiques d'initialisation XXXINI

• Itérations de Newton :

  • Incrémentation de la solution $y^k = y_{i-1} + \Delta y^k$

  • calcul du résidu $-f(\Delta y^k)$ LCRESI => appel aux routines calculant le résidu XXXRES

  • calcul de la matrice jacobienne $\frac{df}{dy_k}$: LCJACB => appel aux routines spécifiques XXXJAC

  • résolution du système linéaire $\frac{\partial f^l}{\partial \Delta y_k} d(\Delta y_k) = - f^l(\Delta y_k)$ par la méthode de Gauss

  • incrémentation de $\Delta y_{k+1} = \Delta y_k + d(\Delta y_k)$

  • test de convergence LCCONV => par défaut, test sur l'erreur relative sur la solution $|y_{k+1} - y_k| < \varepsilon |y_{k+1}|$; si nécessaire, appel aux routines spécifiques XXXCVG

    • si non convergence, poursuite des itérations, sauf si le nombre maximum est atteint, auquel cas on sort de PLASTI avec un code retour, pour activer le re-découpage local ou global du pas de temps (suivant les données utilisateur : ITER_INTE_PAS pour le re-découpage local et DEFI_LIST_INST pour le re-découpage global).

    • Si convergence, incrémentation $y^k = y_{i-1} + \Delta y^k$ et sortie.

Pour implanter un modèle de comportement XXX, il faut donc écrire a minima les routines :

- XXXMAT : récupération des données matériau, appelée par LCMATE
- XXXCVX pour l'évaluation du seuil initial
- XXXRES : calcul du résidu, appelée par LCRESI
- XXXJAC : calcul de la matrice jacobienne, appelée par LCJACB.
- On peut aussi modifier les routines suivantes, si besoin, pour définir de façon particulière :
- l'élasticité XXXELA appelée par LCELAS
- l'initialisation XXXINI appelée par LCINIT
- le test de convergence XXXCVG, appelé par LCCONV.

La mise à jour des catalogues concerne uniquement DEFI_MATERIAU et C_COMPORTEMENT, ainsi que le catalogue de la loi de comportement (voir [D5.04.01]).

4.4 Implantation implicite facile d'un nouveau modèle de comportement dans l'environnement PLASTI

Il est possible d'utiliser directement les 2 routines (récupération des données matériau, dérivées des variables internes) utilisées avec ALGO_INTE_='RUNGE_KUTTA' pour réaliser un intégration implicite. En effet, le système d'équation différentielles résolu par RUNGE_KUTTA peut s'écrire :

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

$$\boldsymbol{\sigma}=\boldsymbol{\sigma}_{i-1}+\Delta\boldsymbol{\sigma}=\boldsymbol{\sigma}_{i-1}+\mathbf{A}(\Delta\boldsymbol{\varepsilon}_{\mathbf{i}}^{\mathbf{n}}-\Delta\boldsymbol{\varepsilon}^{\mathbf{th}}-\Delta\boldsymbol{\varepsilon}^{\mathbf{p}}(Y))=G(Y)$$
$$\frac{dY}{dt}=F(Y,t;\boldsymbol{\sigma})$$

où Y représente l'ensemble des variables internes du modèle. On suppose ici que les premières valeurs de Y sont les composantes de la variation de déformation plastique $\Delta\varepsilon^p$. La relation entre le tenseur des contraintes et la partie élastique du tenseur des déformations est généralement linéaire, mais peut être évaluée de façon non linéaire par une expression spécifique.

Une fois programmée la routine permettant de calculer $\frac{dY}{dt}=F(Y,t;\boldsymbol{\sigma})$, il est possible de l'utiliser pour une intégration implicite, ce qui consiste à résoudre (cf. 3.1) :

$$R(\Delta Z)=0=\begin{bmatrix}R_1(\Delta Z)\\R_2(\Delta Z)\end{bmatrix} \text{ avec } \Delta Z=\begin{pmatrix}\Delta\boldsymbol{\sigma}\\\Delta\mathbf{Y}\end{pmatrix}=Z_i-Z_{i-1}.$$

- Le premier système d'équations représente la relation contrainte - déformation élastique : $R_1(\Delta Z)=\boldsymbol{\sigma}-G(Y)=0$;
- Le deuxième exprime les lois d'évolution des différentes variables internes : $R_2(\Delta Z)=\Delta Y-\Delta t.F(Y,t;\boldsymbol{\sigma})=0$, , où G et F sont calculés par la routine explicite.

Pour résoudre ce système par la méthode de Newton proposée dans l'environnement PLASTI, soit :

$$\begin{cases}\dfrac{\partial f^l}{\partial\Delta Z_k}d(\Delta Z_k)=-f^l(\Delta Z_k)\\ \Delta Z_{k+1}=\Delta Z_k+d(\Delta Z_k)\end{cases}$$, i l faut disposer de la matrice jacobienne $\frac{\partial f^l}{\partial\Delta Z_k}$.

Son calcul peut être effectué automatiquement par perturbation : il suffit d'utiliser ALGO_INTE='NEWTON_PERT'.

De plus, la matrice tangente est alors calculées directement à partir de la matrice jacobienne suivant la méthode décrite au § 3.2 .

En résumé, ce procédé permet, avec les deux seules routines nécessaires à l'intégration explicite, d'utiliser une intégration implicite, et de bénéficier d'un matrice tangente.

La mise à jour des catalogues concerne uniquement DEFI_MATERIAU et C_COMPORTEMENT, ainsi que le catalogue de la loi de comportement (cf. [D5.04.01]).

La mise en œuvre de cette méthode est illustrée dans le test SSNV225, pour une loi de fluage.

5 Bibliographie

1) CROUZEIX M., MIGNOT A.L. : “Analyse numérique des équations différentielles”, Masson, 1989.

2) CAILLETAUD G., PILVIN P. : “Utilisation de modèles polycristallins pour le caclul par éléments finis”, Revue européenne des éléments finis, vol.3, n°4, pp. 515-541, 1994.

3) ZIANI M., « Accélération de la convergence des méthodes de type Newton pour la résolution des systèmes non-linéaires », thèse de doctorat, 2008.

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

6 Description des versions du document

Version Aster	Auteur(s) Organisme(s)	Description des modifications
4	*P. GEYER, E. LORENTZ, C. VOGEL (EDF/RNE/EMA, IMA/MMN)*	Texte initial (intégration explicite)
10	*S. GENIAUT, J.M. PROIX*	Ajout de l'intégration implicite, et de la recherche linéaire.
11.2	*J.M. PROIX*	Ajout du §4.4

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