Algorithmes de propagation de fissures avec X-FEM

Résumé :

Ce document présente les algorithmes de propagation de fissures dans le cadre de la méthode X-FEM [R7.02.12]. La fissure est alors représentée par des level-sets. La propagation de la fissure consiste donc à mettre à jour les level-sets compte-tenu de l'avancée du front de fissure.

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

Table des Matières

1 Introduction....3
2 Calcul du champ de vitesse de propagation....4
2.1 Vitesse d'avancée du fond de fissure....4
2.2 Champ de vitesse étendu au maillage....4
2.3 Calcul des vitesses pour chaque nœud du maillage....6
2.4 Calcul de la base locale....7
2.5 Ajustement du champ de vitesse normale....9
2.6 Prise en compte du déversement....9
2.7 Détermination des angles de bifurcation....10
3 Mise à jour des level-sets....12
3.1 Équations de mise à jour....12
4 Réinitialisation et réorthogonalisation des level-sets....14
4.1 Introduction....14
4.2 Résolution par la méthode de fast marching....15
4.2.1 Méthode « Upwind »....15
4.2.2 Méthode « Simplexe »....23
4.3 Calcul des level-sets à proximité des iso-zéros....29
5 Localisation du domaine de calcul des level-sets....31
5.1 Sélection des nœuds/points et des éléments du domaine de calcul localisé....32
5.2 Déplacement du domaine localisé en suite à la propagation de la fissure....33
5.3 Limitations sur la valeur du rayon du domaine localisé entre deux propagation successives....34
6 Calcul direct des nouvelles level-sets après avancée....37
6.1 Méthode de calcul des level-sets....37
6.2 Traitement des points problématiques....38
6.3 Discontinuité de la level-set normale et limitations de calcul....39
6.4 Localisation du domaine....40
7 Méthode maillage pour la mise à jour des level-sets....41
7.1 Présentation de la méthode....41
7.2 Algorithme....41
8 Séparation et fusion des fronts de fissure....43
8.1 Méthode maillage....43
8.2 Méthode upwind : introduction d'un fond virtuel....45
8.3 Méthode géométrique et simplexe....50
9 Bibliographie....51
10....Description des versions
....53

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

1 Introduction

Un avantage conséquent de la méthode X-FEM étant la possibilité d'étudier plusieurs géométries de fissure sur un même maillage, on peut aisément étudier l'évolution de la fissure au cours du temps. L'opérateur PROPA_FISS [U4.82.11] calcule la fissure résultante d'une propagation en fatigue selon une loi de Paris locale.

À partir de la donnée d'une fissure à un instant donné et des coefficients d'intensité des contraintes correspondants (Opérateur CALC_G [U4.82.03]), l'opérateur calcule les level-sets après avancée de la fissure, puis la structure de données fiss_xfem est enrichie de la même manière qu'avec l'opérateur DEFI_FISS_XFEM [U4.82.08].

Une introduction à la méthode des Level sets est donnée dans [R7.02.12]. La phase de propagation de la fissure se traduit simplement par la propagation des level-sets

À l'heure actuelle on peut calculer les nouvelles level-sets après avancée par trois approches différentes: par mise à jour des level-sets suivant une méthode de type fast marching, par mise à jour après un calcul géométrique direct ou par mise à jour à partir d'un maillage propagé reconstruit (le maillage de la fissure est reconstruit à partir des données de propagation et les level-sets sont calculées dans un second temps par distance au maillage). Les différentes étapes de la propagation de fissure X-FEM dans *Code_Aster* changent selon l'approche utilisée.

Néanmoins l'étape de calcul de la vitesse d'avancée du fond de fissure est commune à toutes les méthodes (cf. [§ 2]). On définit les level-sets traduisant la nouvelle géométrie de fissure et on s'assure que celles-ci gardent des définitions proches de fonctions distances signées (cf. [§ 4]). La propagation d'une leve l set nécessite ainsi les trois étapes successives suivantes (bib 2) :

- extension de la vitesse connue sur l'iso-zero vers le domaine entier,
- propagation de la level-set a partir de ce champ de vitesse,
- réinitialisation de la fonction level-set afin de conserver une fonction distance signée.

Au contraire, dans le cas où on utilise le calcul direct, on recalcule tout de suite les nouvelles valeurs des deux level-sets (cf. [§6]).

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

2 Calcul du champ de vitesse de propagation

Les premières étapes consistent à calculer deux champs de vitesses, normale et tangentielle, définissant l'évolution dissociée des deux level-sets. On se place dans le cadre d'une propagation en mode mixte.

2.1 Vitesse d'avancée du fond de fissure

L'avancée du fond de fissure en chacun de ses points, évolue dans la base locale selon une loi de Paris. Sa direction peut être déterminée par plusieurs critères détaillés dans le paragraphe [§2.7]. On peut généralement écrire :

$$\boldsymbol{V}=V_N\boldsymbol{e_1}+V_T\boldsymbol{e_2} \qquad \text{dans la base locale } (\boldsymbol{e}_1,\boldsymbol{e}_2,\boldsymbol{e}_3) \tag{1}$$

Soit encore :

$$\boldsymbol{V}=C\,G^m\left(\sin\beta\,\boldsymbol{e_1}+\cos\beta\,\boldsymbol{e_2}\right) \tag{2}$$

Et où C et m sont des caractéristiques du comportement en fatigue du matériau, G est le taux de restitution d'énergie local issu de la méthode G-Thêta, et β est l'angle de branchement. Cet angle vaut :

$$\beta=2\arctan\left[\frac{1}{4}.\left(\frac{K_I}{K_{II}}-sign\left(K_{II}\right).\sqrt{\left(\frac{K_I}{K_{II}}\right)^2+8}\right)\right] \tag{3}$$

Ou encore :

$$\beta=2\arctan\left[\frac{1}{4}.\left(\frac{1+x_I-\left(1-x_{III}\right)^p}{x_{II}}-sign\left(K_{II}\right).\sqrt{\left(\frac{1+x_I-\left(1-x_{III}\right)^p}{x_{II}}\right)^2+8}\right)\right] \tag{4}$$

Avec :

$$x_i=\frac{K_i}{K_I+\left|K_{II}\right|+\left|K_{III}\right|} \quad \text{et} \quad p=\frac{\sqrt{\pi}-5\nu}{4} \tag{5}$$

2.2 Champ de vitesse étendu au maillage

Les vitesses normale et tangente sont connues uniquement pour les points d'intersection entre le fond de la fissure et les faces des éléments du maillage. Ces points ne sont pas des nœuds du maillage. Pour la mise à jour des level-sets après une propagation, la vitesse doit être connue pour chaque nœud du maillage [§3.1]. Il est donc nécessaire qu'on étende la vitesse à tous les nœuds du maillage à partir des valeurs sur le fond de fissure. Pour cela on projette chaque nœud sur le fond de fissure en direction normale et on assigne au nœud la vitesse du point qui est la projection du nœud.

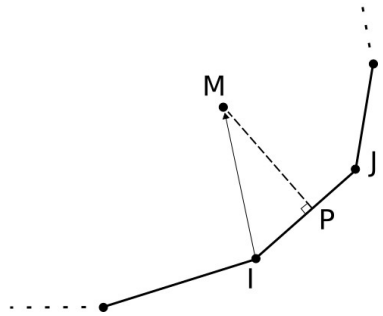


Figure 2.2-1: projection du nœud M sur le fond de fissure

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

La projection est mise en œuvre selon l'algorithme suivant (Figure 2.2-1) :

- Boucle sur les nœuds M du maillage
  - Initialisation: $d_{min}=\text{réel maxi}$
  - Boucle sur les segments $\vec{\mathrm{IJ}}$: entre deux points I et J consécutifs du fond de fissure
    - On calcule $s=\frac{\vec{\mathrm{IJ}}\cdot\vec{\mathrm{IM}}}{\|\vec{\mathrm{IJ}}\|^2}=\frac{\|\vec{\mathrm{IP}}\|\cdot\|\vec{\mathrm{IJ}}\|}{\|\vec{\mathrm{IJ}}\|^2}=\frac{\|\vec{\mathrm{IP}}\|}{\|\vec{\mathrm{IJ}}\|}$
    - Si $s<0$, alors $s=0$
    - Si $s>1$, alors $s=1$
    - Si on projette le nœud M sur $\vec{\mathrm{IJ}}$, on obtient le point P . La position de ce point est déterminée par le vecteur $\vec{\mathrm{IP}}$: $\vec{\mathrm{IP}}=s\cdot\vec{\mathrm{IJ}}$
    - On calcule la distance entre le nœud M et sa projection P : $d=\left\|\overrightarrow{\mathrm{MP}}\right\|$
    - Si $d<d_{min}$, alors on stocke d_{min} , $\mathrm{I}_{min}=\mathrm{I}$, $\mathrm{J}_{min}=\mathrm{J}$ et $s_{min}=s$
  - Fin boucle
- Fin boucle

Le paramètre s_{min} donne la position de la projection P de chaque nœud M du maillage par rapport à la longueur du vecteur $\overrightarrow{\mathrm{I}_{min}\mathrm{J}_{min}}$ du fond de fissure auquel la projection P appartient. La position de la projection P peut donc être regardée comme position paramétrique sur le vecteur $\overrightarrow{\mathrm{I}_{min}\mathrm{J}_{min}}$. Cela peut être utilisé pour calculer la valeur du vecteur vitesse V à assigner au nœud M .

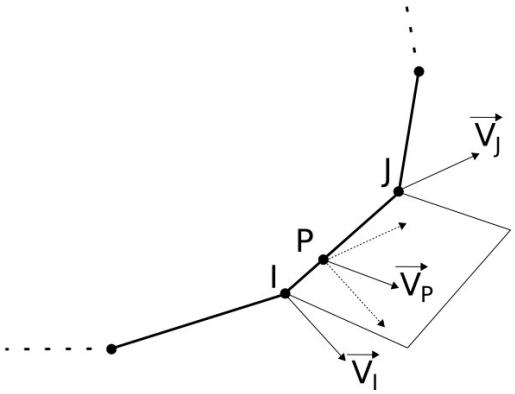


Figure 2.2-2: détermination de la vitesse du point N

On ramène ce problème à la détermination du vecteur vitesse $\boldsymbol{V}_P$ d'un point P du segment IJ quand les vecteurs vitesse sont connus aux points I et J (Figure 2.2-2).

La méthode utilisée pour le calcul de $\boldsymbol{V}_P$ est arbitraire. Si on regarde la signification physique de la vitesse, on peut dire qu'elle change graduellement entre les deux vecteurs limites $\boldsymbol{V}_I$ et $\boldsymbol{V}_J$ avec la position du point P . Pour la valeur du $\left\|\boldsymbol{V}_P\right\|$ on peut donc utiliser une interpolation linéaire entre les deux valeurs $\left\|\boldsymbol{V}_\mathrm{I}\right\|$ et $\left\|\boldsymbol{V}_\mathrm{J}\right\|$:

$$\left\|\boldsymbol{V}_\mathrm{P}\right\|=\left(\left\|\boldsymbol{V}_\mathrm{J}\right\|-\left\|\boldsymbol{V}_\mathrm{I}\right\|\right)\cdot s_{min}+\left\|\boldsymbol{V}_\mathrm{I}\right\| \tag{6}$$

Pour la direction de $\boldsymbol{V}_\mathrm{P}$, on peut déterminer l'angle entre $\boldsymbol{V}_\mathrm{I}$ et $\boldsymbol{V}_\mathrm{J}$ et utiliser encore une fois une interpolation linéaire sur l'angle calculé, selon l'algorithme suivante:

- on calcule le vecteur normal au plan formé par les vecteurs $\boldsymbol{V}_\mathrm{I}$ et $\boldsymbol{V}_\mathrm{J}$: $\boldsymbol{v}_3=\boldsymbol{V}_\mathrm{I}\wedge\boldsymbol{V}_\mathrm{J}$
- on se ramène dans une base locale formée par les vecteurs $\boldsymbol{V}_\mathrm{I}$ et $\boldsymbol{v}_3$. On calcule le dernier axe de la base: $\boldsymbol{v}_2=\boldsymbol{v}_3\wedge\boldsymbol{V}_\mathrm{I}$. Ce vecteur est dans le plan formé par les vecteurs $\boldsymbol{V}_\mathrm{I}$ et $\boldsymbol{V}_\mathrm{J}$.
- on calcul les vecteurs unitaire de la base dans ce plan: $\boldsymbol{u}_1=\frac{\boldsymbol{V}_\mathrm{I}}{\|\boldsymbol{V}_\mathrm{I}\|}$ et $\boldsymbol{u}_2=\frac{\boldsymbol{v}_2}{\|\boldsymbol{v}_2\|}$
- on calcule l'angle entre les deux vecteurs vitesse $\boldsymbol{V}_\mathrm{I}$ et $\boldsymbol{V}_\mathrm{J}$: $\alpha=acos\left(\frac{\sum_{i=1}^{3}V_{1i}\cdot V_{Ji}}{\|\boldsymbol{V}_\mathrm{I}\|\cdot\|\boldsymbol{V}_\mathrm{J}\|}\right)$
- on calcule le vecteur unitaire de la vitesse au point P par interpolation linéaire:

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

- $\boldsymbol{u}_{\mathrm{P}}=\cos\left(\alpha\, s_{min}\right)\cdot\boldsymbol{u}_1+\sin\left(\alpha\, s_{min}\right)\cdot\boldsymbol{u}_2$
- enfin on calcule le vecteur vitesse au point P en utilisant la valeur $\left\|\boldsymbol{V}_{\mathrm{P}}\right\|$ déjà calculée: $\boldsymbol{V}_{\mathrm{P}}=\left\|\boldsymbol{V}_{\mathrm{P}}\right\|\cdot\boldsymbol{u}_{\mathrm{P}}$

2.3 Calcul des vitesses pour chaque nœud du maillage

Pour faire évoluer la level-set normale convenablement [§2.5] et pour intégrer les deux équations de mise à jour des level-set [§3], on doit connaître les vecteurs vitesse normale $\boldsymbol{V}_N$ et tangente $\boldsymbol{V}_T$ pour chaque nœud du maillage. Pour cela il suffit de calculer les directions normale et tangente des level-sets en utilisant les gradients des level-sets.

Malheureusement cette solution n'est pas applicable à tous les nœuds. En effet, après la mise à jour, la propriété d'orthogonalité des level-sets n'est pas garantie [§4.1] pour tous les nœuds. Cette propriété est très importante pour garantir l'orthogonalité des vecteurs vitesse $\boldsymbol{V}_N$ et $\boldsymbol{V}_T$. On doit donc utiliser une autre solution.

En regardant la façon utilisée pour étendre le champ de vitesse aux nœuds du maillage [§2.2,§2.5], on envisage la solution suivante:

1. si la level-set tangente au nœud est négative: elle évolue dans une direction parallèle à la surface de la fissure. On utilise donc les gradients des level-sets pour calculer les directions normale et tangente. En effet, cela signifie que le nœud est dans l'espace de la fissure existante et que la valeur de la level-set normale ne change pas pendant la mise à jour des level-sets contrairement à la valeur de la level-set tangente [§2.5].
2. si la level-set tangente au nœud est positive, on utilise la base locale de la projection du nœud sur le fond de fissure. En effet, l'évolution des level-sets doit être cohérente avec les modifications du fond de fissure.

Pour le premier cas, le calcul des directions normale et tangente ne pose pas de problème parce que les gradients des level-sets φ aux nœuds sont toujours connus: $\boldsymbol{V}_N=V_N^{\mathrm{P}}\cdot\nabla\varphi_n$ et $\boldsymbol{V}_T=V_T^{\mathrm{P}}\cdot\nabla\varphi_t$. Toutefois l es composantes normale V_N^{P} et tangente V_T^{P} de la vitesse $\boldsymbol{V}_{\mathrm{P}}$ doivent être calculées par rapport à la base locale du fond de fissure au point P en utilisant la solution suivante décrite pour le second cas .

Dans ce cas, le calcul est plus difficile parce que la base locale du fond de fissure est connue uniquement pour les points d'intersection entre le fond de la fissure et les faces des éléments du maillage. La situation est visible en figure 2.3-1: on connaît la base locale aux points I et J et on veut calculer la base locale au point P, projection d'un nœud sur le fond de fissure. L'orientation de cette base doit être comprise entre les orientations de la base au point I et de la base au point J selon la position du point P. Cette dernière est bien décrite par le paramètre s_{min} déjà calculé [§2.2].

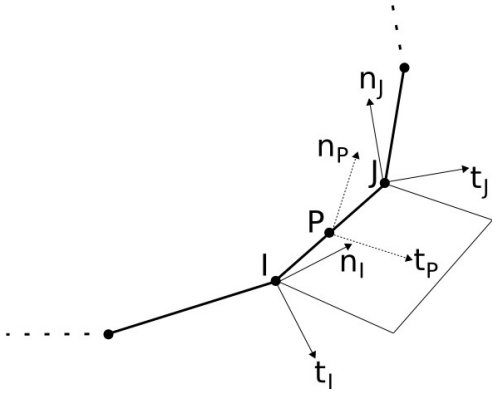


Figure 2.3-1: détermination de la base locale au point P

Si on ne considère pas la translation de l'origine, le changement d'orientation de la base locale peut être vu comme une rotation de la base à partir de l'orientation au point I, pour laquelle $s_{min}=0$. On a la rotation maximale pour

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

$s_{min}=1$, c'est à dire au point J . L'orientation de la base au point P est obtenue en utilisant une rotation inférieure proportionnelle à la valeur de s_{min} .

En utilisant le théorème d'Euler, on peut toujours décrire une rotation quelconque comme une seule rotation (angle d'Euler) autour d'un seul axe (axe d'Euler). La direction de ce dernier et la valeur de la rotation changent pour le segment $\overline{IJ}$ le long du fond de fissure.

La base locale au point P peut donc être obtenue par une rotation autour de l'axe d'Euler calculé pour la rotation qui transforme la base au point I dans la base au point J . La valeur de l'angle de rotation peut être calculée simplement en multipliant l'angle d'Euler par s_{min} . Dans le paragraphe suivant [§2.4], on décrit les équations qu'on utilise pour ce calcul.

Après la détermination de la base locale $\boldsymbol{t}_{\mathrm{P}}$ et $\boldsymbol{n}_{\mathrm{P}}$ au point P , on peut calculer la valeur des composantes de la vitesse $\boldsymbol{V}_{\mathrm{P}}$. Soit :

$$V_N^{\mathrm{P}}=\boldsymbol{V}_{\mathrm{P}}\cdot\boldsymbol{n}_{\mathrm{P}} \text{ et } V_T^{\mathrm{P}}=\boldsymbol{V}_{\mathrm{P}}\cdot\boldsymbol{t}_{\mathrm{P}} \tag{7}$$

Enfin on calcule les deux vecteurs $\boldsymbol{V}_N$ et $\boldsymbol{V}_T$:

$$\boldsymbol{V}_N=V_N^P\cdot\boldsymbol{n}_{\mathrm{P}} \text{ et } \boldsymbol{V}_T=V_T^P\cdot\boldsymbol{t}_{\mathrm{P}} \tag{8}$$

En résumé, pour chaque nœud M du maillage on calcule la projection P sur le fond de fissure. Ensuite on calcule la base locale $\boldsymbol{t}_{\mathrm{P}}$ et $\boldsymbol{n}_{\mathrm{P}}$ au point P et les composantes de la vitesse $\boldsymbol{V}_{\mathrm{P}}$ (qui ont déjà été calculés [§2.2]):

$$\begin{cases} V_N^{\mathrm{P}}=\boldsymbol{V}_{\mathrm{P}}\cdot\boldsymbol{n}_{\mathrm{P}} \\ V_T^{\mathrm{P}}=\boldsymbol{V}_{\mathrm{P}}\cdot\boldsymbol{t}_{\mathrm{P}} \end{cases} \tag{9}$$

Enfin on calcule les vecteurs $\boldsymbol{V}_N$ et $\boldsymbol{V}_T$ au nœud M selon le signe de la level-set tangente :

$$\varphi_t\leqslant 0 \quad \begin{cases} \boldsymbol{V}_N=V_N^{\mathrm{P}}\cdot\nabla\varphi_{\mathrm{n}} \\ \boldsymbol{V}_T=V_T^{\mathrm{P}}\cdot\nabla\varphi_t \end{cases} \text{ ou } \varphi_t>0 \quad \begin{cases} \boldsymbol{V}_N=V_N^P\cdot\boldsymbol{n}_{\mathrm{P}} \\ \boldsymbol{V}_T=V_T^P\cdot\boldsymbol{t}_{\mathrm{P}} \end{cases} \tag{10}$$

Où φ_n et φ_t sont respectivement la level-set normale et la level-set tangente.

2.4 Calcul de la base locale

Comme déjà décrit dans le paragraphe précédent, on connaît la base locale aux points I et J et on veut calculer la base locale au point P en utilisant une rotation.

La première étape est le calcul de l'axe et de l'angle d'Euler qui décrivent la rotation qu'on doit appliquer à la base (t_I,n_I,b_I) au point I pour obtenir la base (t_J,n_J,b_J) au point J . L'algorithme utilisé est le suivant :

- O n calcule le troisième axe b de la base locale aux points I et J . En fait, uniquement les vecteurs unitaires t et n sont stockés :

$$\begin{cases} \boldsymbol{b}_{\mathrm{I}}=\boldsymbol{t}_{\mathrm{I}}\wedge\boldsymbol{n}_{\mathrm{I}} \\ \boldsymbol{b}_{\mathrm{J}}=\boldsymbol{t}_{\mathrm{J}}\wedge\boldsymbol{n}_{\mathrm{J}} \end{cases} \tag{11}$$

- O n calcule la matrice de rotation d'Euler entre les deux bases :

$$\underline{\underline{\boldsymbol{R}_E}}=\begin{bmatrix} \boldsymbol{t}_{\mathrm{J}}\cdot\boldsymbol{t}_{\mathrm{I}} & \boldsymbol{n}_{\mathrm{J}}\cdot\boldsymbol{t}_{\mathrm{I}} & \boldsymbol{b}_{\mathrm{J}}\cdot\boldsymbol{t}_{\mathrm{I}} \\ \boldsymbol{t}_{\mathrm{J}}\cdot\boldsymbol{n}_{\mathrm{I}} & \boldsymbol{n}_{\mathrm{J}}\cdot\boldsymbol{n}_{\mathrm{I}} & \boldsymbol{b}_{\mathrm{J}}\cdot\boldsymbol{n}_{\mathrm{I}} \\ \boldsymbol{t}_{\mathrm{J}}\cdot\boldsymbol{b}_{\mathrm{I}} & \boldsymbol{n}_{\mathrm{J}}\cdot\boldsymbol{b}_{\mathrm{I}} & \boldsymbol{b}_{\mathrm{J}}\cdot\boldsymbol{b}_{\mathrm{I}} \end{bmatrix} \tag{12}$$

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

Chaque colonne de la matrice représente un vecteur de la base au point J (t_J, n_J, b_J) dans la base locale au point I .

- On calcule l'angle d'Euler en utilisant les éléments de la matrice $\underline{\underline{R}}_E$:

$$\vartheta_E = \text{acos}\left(\frac{R_{11}+R_{22}+R_{33}-1}{2}\right) \tag{13}$$

- On calcule l'axe d'Euler $\boldsymbol{e} = \left[e_1 \;\; e_2 \;\; e_3\right]^{\mathrm{T}}$:

$$\begin{cases} e_1 = \dfrac{R_{23}-R_{32}}{2\cdot\sin\vartheta_E} \\ e_2 = \dfrac{R_{31}-R_{13}}{2\cdot\sin\vartheta_E} \\ e_3 = \dfrac{R_{12}-R_{21}}{2\cdot\sin\vartheta_E} \end{cases} \tag{14}$$

L'angle et l'axe d'Euler calculés sont les mêmes pour tous les points P sur l'arête définie par les points I et J. La deuxième étape est le calcul de la base locale pour un point P pour lequel la position est déterminée par le paramètre s_{min} [§2.2]. L'algorithme utilisé est le suivant :

- On calcule l'angle de rotation effective à utiliser :

$$\vartheta_P = \vartheta_E \cdot s_{min} \tag{15}$$

- O n calcule la matrice de la rotation entre la base locale au point I et la base locale au point J pour une rotation autour de l'axe d'Euler $\boldsymbol{e}$ égale à ϑ_P :

$$\underline{\underline{T}}_P = \cos\left(\vartheta_P\right)\cdot\underline{\underline{I}} + \left(1+\cos\left(\vartheta_P\right)\right)\cdot\boldsymbol{e}\cdot\boldsymbol{e}^T - \sin\left(\vartheta_P\right)\cdot\underline{\underline{\epsilon}} \tag{16}$$

où la matrice $\underline{\underline{\epsilon}}$ est la suivante :

$$\underline{\underline{\epsilon}} = \begin{bmatrix} 0 & -e_3 & e_2 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{bmatrix} \tag{17}$$

- O n calcule les axes $\boldsymbol{t}_{\mathrm{P}}$ et $\boldsymbol{n}_{\mathrm{P}}$ de la base locale au point P dans la base locale au point I :

$$\begin{cases} \boldsymbol{t}_{\mathrm{P}\ (t_I, n_I, b_I)} = \underline{\underline{\boldsymbol{T}}}_{\mathrm{P}} \cdot \left[1 \;\; 0 \;\; 0\right]^T \\ \boldsymbol{n}_{\mathrm{P}\ (t_I, n_I, b_I)} = \underline{\underline{\boldsymbol{T}}}_{\mathrm{P}} \cdot \left[0 \;\; 1 \;\; 0\right]^T \end{cases} \tag{18}$$

Les deux vecteurs sont donc coïncidents avec la première et la deuxième colonne de la matrice $\underline{\underline{\boldsymbol{T}}}_P$. En utilisant l'équation ci-dessus pour le calcul de cette matrice, on peut donc écrire:

$$\boldsymbol{t}_{\mathrm{P}\ (t_I, n_I, b_I)} = \begin{bmatrix} \cos\left(\vartheta_{\mathrm{P}}\right) + \left(1-\cos\left(\vartheta_{\mathrm{P}}\right)\right)\cdot e_1^2 \\ \left(1-\cos\left(\vartheta_{\mathrm{P}}\right)\right)\cdot e_1\cdot e_2 - \sin\left(\vartheta_{\mathrm{P}}\right)\cdot e_3 \\ \left(1-\cos\left(\vartheta_{\mathrm{P}}\right)\right)\cdot e_1\cdot e_3 + \sin\left(\vartheta_{\mathrm{P}}\right)\cdot e_2 \end{bmatrix} \tag{19}$$

Et :

$$\boldsymbol{n}_{\mathrm{P}\ (t_I, n_I, b_I)} = \begin{bmatrix} \left(1-\cos\left(\vartheta_{\mathrm{P}}\right)\right)\cdot e_1\cdot e_2 + \sin\left(\vartheta_{\mathrm{P}}\right)\cdot e_3 \\ \cos\left(\vartheta_{\mathrm{P}}\right) + \left(1-\cos\left(\vartheta_{\mathrm{P}}\right)\right)\cdot e_2^2 \\ \left(1-\cos\left(\vartheta_{\mathrm{P}}\right)\right)\cdot e_2\cdot e_3 - \sin\left(\vartheta_{\mathrm{P}}\right)\cdot e_1 \end{bmatrix} \tag{20}$$

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

- Enfin on peut exprimer les vecteurs $\boldsymbol{t}_{\mathrm{P}}^{\mathrm{I}}$ et $\boldsymbol{n}_{\mathrm{P}}^{\mathrm{I}}$ dans la base globale du maillage, qui est utilisée pour tous les calculs. Si on dénote la composante i du vecteur $\boldsymbol{t}_{\mathrm{P}\ (t_I,n_I,b_I)}$ par $t_{\mathrm{P}\ (t_I,n_I,b_I)}(i)$ et la même composante du vecteur $\boldsymbol{n}_{\mathrm{P}\ (t_I,n_I,b_I)}$ par $n_{\mathrm{P}\ (t_I,n_I,b_I)}(i)$, on peut écrire :

$$\begin{cases} \boldsymbol{t}_{\mathrm{P}}=t_{\mathrm{P}\ (t_I,n_I,b_I)}(1)\cdot\boldsymbol{t}_{\mathrm{I}}+t_{\mathrm{P}\ (t_I,n_I,b_I)}(2)\cdot\boldsymbol{n}_{\mathrm{I}}+t_{\mathrm{P}\ (t_I,n_I,b_I)}(3)\cdot\boldsymbol{b}_{\mathrm{I}} \\ \boldsymbol{n}_{\mathrm{P}}=n_{\mathrm{P}\ (t_I,n_I,b_I)}(1)\cdot\boldsymbol{t}_{\mathrm{I}}+n_{\mathrm{P}\ (t_I,n_I,b_I)}(2)\cdot\boldsymbol{n}_{\mathrm{I}}+n_{\mathrm{P}\ (t_I,n_I,b_I)}(3)\cdot\boldsymbol{b}_{\mathrm{I}} \end{cases} \quad (21)$$

2.5 Ajustement du champ de vitesse normale

Afin d'empêcher le déplacement de la fissure préexistante, et de faire évoluer la level-set normale convenablement, il est nécessaire d'ajuster le champ de vitesse normale (bib1). On veut avoir une trajectoire de fissure relativement lissée, telle que :

$$\overline{V}_N(lst)=\begin{cases}0 & \text{si} \quad lst\le 0 \quad (\text{et notamment en } lst=0)\\ V & \text{si} \quad lst=V\cdot\Delta t\end{cases} \quad (22)$$

La valeur du Δt est égale au temps total d'intégration des équations de mise à jour des level-sets [§ 3.1]. Ainsi, le fond de fissure avance bien selon V_T et V_N calculées précédemment. On fait une approximation linéaire en lst.

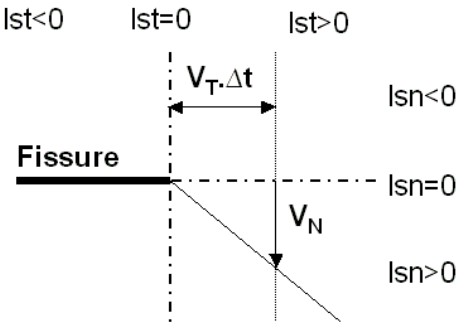


Figure 2.5-1 : Ajustement de la vitesse normale

La formule appliquée sur tous les nœuds du maillage est la suivante :

$$\overline{V}_N=h(lst).\frac{V_N.lst}{V_T.\Delta t} \text{ avec } h(x)=\begin{cases}0 & \text{si} \quad x<0\\ 1 & \text{si} \quad x>0\end{cases} \quad (23)$$

Où V_N et V_T sont les champ de vitesse nodaux, et Δt le pas de temps choisi (voir paragraphe [§3]).

2.6 Prise en compte du déversement

Dans le cas d'une fissure tridimensionnelle, deux rotations sont possibles lors de la propagation d'une fissure. Soit un point du fond de fissure (P) auquel est associé une base locale (**t**,**n**,**b**). [bib 21] décrit ainsi la rotation de *branchement* comme la rotation du vecteur vitesse $\boldsymbol{V}$ dans le plan $(P,\boldsymbol{t},\boldsymbol{n})$ et β son angle associé, et la rotation de *déversement* comme étant la rotation de la base locale autour du vecteur vitesse, d'angle γ.

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

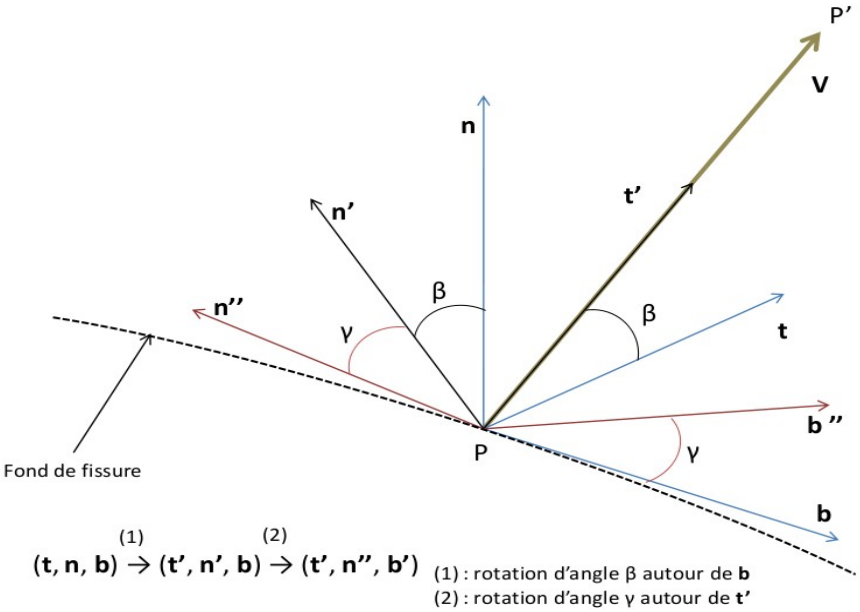


Figure 2.6-1: Définition des angles de branchement et de déversement

La prise en compte du déversement consiste donc à effectuer la rotation des bases locales avant l'étape de mise à jour des level-sets. L'étape de propagation suit donc l'enchaînement suivant :

1. application des rotations de *branchement* (β) et de *déversement* (γ) aux bases locales des points du fond de fissure,
2. projection des points du maillage sur le fond de fissure,
3. attribution d'une base locale et d'une vitesse $\boldsymbol{V}$ de propagation à tous les points projetés par interpolation entre les points du fond de fissure,
4. mise à jour des level-sets dans les bases locale avec la vitesse $\boldsymbol{V}$.

Les bases locales ayant subi la rotation avant l'étape de mise à jour des level-sets, le vecteur vitesse $\boldsymbol{V}$ est donc porté uniquement par le vecteur $\boldsymbol{t}'$ de la nouvelle base locale. Il n'y a donc plus de composante normale à la vitesse dans la nouvelle base locale.

Remarque :

La rotation de déversement ne peut être activée qu'en 3D et lorsque le mot-clé `CRIT_ANGL_BIFURCATION` *de l'opérateur de propagation* `PROPA_FISS` *vaut* `SITT_MAX_DEVER`.
Par défaut la rotation de déversement est forcée à 0 .

2.7 Détermination des angles de bifurcation

D ans le c as où le déversement n'est pas pris en compte, l'angle de branchement est classiquement déterminé par le principe du « Maximum Hoop Stress criterion » [bib 4] en fonction des coefficients d'intensité des contraintes K_I et K_{II} :

$$\beta = 2\arctan\left[\frac{1}{4}.\left(\frac{K_I}{K_{II}} - sign\left(K_{II}\right).\sqrt{\left(\frac{K_I}{K_{II}}\right)^2 + 8}\right)\right] \quad (24)$$

Remarque :

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

L'angle de branchement peut être calculé par l'opérateur `CALC_G` *[R7.02.05] [U4.82.03]. Dans ce cas il est imposé à* `PROPA_FISS` *dans la même table que G et les SIF.*
Il peut aussi être calculé par l'opérateur `POST_RUPTURE` *[U4.82.04] à l'intérieur de l'opérateur* `PROPA_FISS` *[U4.82.11]. Cette option permet de choisir d'autres critères que le* « Maximum Hoop Stress criterion ».

Dans le c as où le déversement est pris en compte, l'angle de branchement et l'angle de déversement sont tous deux calculés avec le critère du « Maximum Hoop Stress criterion » [bib 21] en faisant intervenir K_I , K_{II} et K_{III} :

$$\beta_c = 2\arctan\left[\frac{1}{4}.\left(\frac{1+ x_I-(1-x_{III})^p}{x_{II}} - sign(K_{II}).\sqrt{(\frac{1+ x_I-(1-x_{III})^p}{x_{II}})^2 + 8}\right)\right] \quad (25)$$

Et pour l'angle de déversement :

$$\gamma = \frac{1}{2}\arctan\left[\frac{2\sigma_{\theta z}(\beta_c)}{\sigma_{\theta\theta}(\beta_c)-\sigma_{zz}(\beta_c)}\right] \quad (26)$$

Avec :

$$x_i = \frac{K_i}{K_I+|K_{II}|+|K_{III}|} \text{ et } p = \frac{\sqrt{\pi}-5\nu}{4} \quad (27)$$

Remarque :

L'angle de déversement est calculé par l'opérateur `POST_RUPTURE` *[U4.82.04] à l'intérieur de l'opérateur* `PROPA_FISS` *[U4.82.11] lorsque le mot-clé* `CRIT_ANGL_BIFURCATION` *de ce dernier est égal à* `SITT_MAX_DEVER` *.*

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

3 Mise à jour des level-sets

Après la propagation, la géométrie de la fissure change et les level-sets doivent évoluer pour bien décrire cette nouvelle géométrie. Même si on peut recalculer les level-sets en utilisant leur définition de distance signée, en pratique cette solution est à éviter pour des questions de performance.

On peut recalculer plus rapidement les level-sets en utilisant les équations de mise à jour des level-sets qui décrivent leur évolution suite à la propagation de la fissure.

Dans les équations suivantes, on utilisera les symboles φ_n , φ_t et φ pour indiquer respectivement la level-set normale (lsn), la level-set tangente (lst) et une level-set en général (lsn ou lst).

3.1 Équations de mise à jour

Les level-sets sont mises à jour sur tous les nœuds selon l'équation de Hamilton-Jacobi suivante (bib1):

$$\frac{\partial \varphi}{\partial t} = -V_{\varphi} \cdot \|\nabla \varphi\| \tag{28}$$

Où φ est la level-set, $\nabla \varphi$ est son gradient et V_{φ} la vitesse de propagation de la level-set φ (V_T ou V_N). Cette équation est connue comme équation d'évolution des level-sets. Dans (bib13) et (bib14) il est montré qu'une erreur est présente dans cette formulation de l'équation et que la forme correcte à utiliser est plutôt la suivante :

$$\frac{\partial \varphi}{\partial t} + \nabla \varphi \cdot \boldsymbol{V} = 0 \tag{29}$$

L'intégration de l'équation pose plusieurs problèmes (bib14):

- Le nombre d'itérations d'intégration est important si un schéma explicite est utilisé (par exemple Runge-Kutta) à cause de la limitation sur le pas de temps utilisable à chaque itération (condition CFL). L'ajustement du champ de vitesse normale ([§2.5]) est à l'origine de la très forte diminution de la valeur maximale du pas de temps utilisable et par conséquent de l'incrément du nombre d'itérations nécessaires. Pour éviter ce problème, un schéma implicite peut être utilisé mais cela complique la programmation. Comme troisième alternative, on peut utiliser une méthode de type « fast marching method » (bib3) qui calcule la solution directement et sans itérations (bib15,bib16). Toutefois des erreurs peuvent être présentes dans la solution calculée par cette dernière méthode aux bords du domaine de calcul (bib15,bib16).
- En présence d'une valeur élevée de l'angle de propagation β ([§2.1]), la solution obtenue par intégration de l'équation des level-set peut présenter des instabilités qui vont perturber la convergence des algorithmes d'intégration au cours des itérations de propagation successives.
- Les instabilités sus-citées entraînent une grande sensibilité numérique de la méthode aux erreurs arrondis, ce qui conduit à une variation significative des résultats entre machines. Ce comportement n'est pas souhaitable dans une utilisation industrielle de la méthode.

Une équation alternative de mise à jour est donc proposée dans (bib14). Elle repose sur l'interprétation géométrique de la modification des level-sets conséquente à la propagation de la fissure. Si on regarde l'iso-zéro de la level-set tangente ($lst=0$ de la figure 2.5-1), on peut dire que chaque point de cette surface se déplace de la quantité $V_T \cdot \Delta t$. Par conséquent, la valeur de la level-set au point diminue de la même quantité. Si on regarde l'iso-zéro de la level-set normale ($lsn=0$ de la figure 2.5-1), on peut dire que dans la partie du volume de matériau sain ($lst>0$) la level-set est soumise à une rotation autour du fond de fissure et que chaque point se déplace suivant la direction normale à l'iso-zéro d'une quantité $V_N \cdot \Delta t$. C'est la variation linéaire de la vitesse V_N avec lst ([§2.5]) qui permet d'obtenir le bon déplacement en chaque point. En généralisant au cas 3D comme dans (bib14), on peut donc écrire les équations de mise à jour suivantes :

$$\Delta \varphi_n = -\left(\boldsymbol{V}_N \cdot \nabla \varphi_n\right) \cdot \Delta t \text{ et } \Delta \varphi_t = -\left(\boldsymbol{V}_T \cdot \nabla \varphi_t\right) \cdot \Delta t \tag{30}$$

L'avantage de ces équations est évident : il s'agit d'équations algébriques explicites qui ne nécessitent l'utilisation d'aucun schéma itératif d'intégration. La solution peut être calculée directement avec un temps de calcul très faible. De

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

plus, aucun problème d'instabilité n'est présent même pour des angles de propagation β élevés comme montré dans (bib14).

La valeur de Δt à utiliser dans les équations de mise à jour des level-sets peut être facilement calculée à partir de l'avancée maximale Δa_{max} de la fissure donnée par l'utilisateur :

$$\Delta t_{tot} = \frac{\Delta a_{max}}{\max\limits_{i \in fond} \left\| \boldsymbol{V}\left(\boldsymbol{X}_i\right) \right\|} \tag{31}$$

Où le vecteur $\boldsymbol{V}\left(\boldsymbol{x}_i\right)$ est le vecteur de la vitesse de propagation ($\boldsymbol{V} = \boldsymbol{V}_N + \boldsymbol{V}_T$) d'un point $\boldsymbol{x}_i$ du fond de fissure. Il faut observer que, dans ce contexte, on prend comme point du fond de fissure les points physiques à l'intersection entre le fond de fissure et les faces des éléments du maillage, pour lesquels on connaît la vitesse et l'angle de propagation, mais qu'on prend aussi les points géométriques du front qui sont la projection normale des nœuds du maillage sur le front lui-même [§2.2]. Toutefois la vitesse de propagation est calculée pour ces derniers points en utilisant une interpolation linéaire [§2.2] et pour chaque arête qui forme le fond de fissure la vitesse maximale est obtenue à l'une de ses deux extrémités. On peut donc limiter la recherche de la vitesse maximale aux points d'intersection entre le fond de fissure et les faces des éléments du maillage.

Remarque:

Dans la description ci-dessus, les symboles t et Δt ont été utilisés pour indiquer le temps. Toutefois la grandeur décrite par ces symboles est effectivement le nombre de cycles de fatigue qui cause l'avancée de la fissure. En fait, si on regarde l'équation utilisée ci-dessus pour le calcul de Δt, Δt est obtenu par division entre la distance d'avancée de la fissure (par exemple: millimètres) et la vitesse d'avancée. La vitesse calculée en utilisant une loi de propagation (par exemple la loi de Paris) est donnée comme une avancée de la fissure par cycle de fatigue (par exemple: millimètres/cycle). On en déduit donc que Δt représente le nombre de cycles de fatigue que l'on doit appliquer pour avoir l'avancée maximale voulue. Le lien entre le nombre de cycles de fatigue et le temps est donné par la fréquence d'application du chargement.
Cela est vrai pour toutes les équations utilisées et on peut dire que toutes les grandeurs qui se ramènent au temps (t, Δt, ...) expriment en fait un nombre de cycles de fatigue.

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

4 Réinitialisation et réorthogonalisation des level-sets

4.1 Introduction

Au cours de leurs évolutions, les level-sets doivent impérativement garder une définition relativement proche de fonctions de distance signée.

La première raison en est le calcul des facteurs d'intensité de contraintes en fond de fissure par la méthode G-Thêta, pour lequel les coordonnées polaires (r,θ) sont définies dans la base locale comme $r=\sqrt{lsn^2+lst^2}$ et $\theta=\arctan(lsn/lst)$; l'utilisation de distances signées pour le calcul des coordonnées polaires est donc nécessaire.

De plus, lors de la propagation de fissure, l'équation de mise à jour présentée plus haut (paragraphe [§3]) met en jeu les normes des gradients des level-sets, qui doivent rester suffisamment proches de l'unité, car si $|\nabla\varphi|$ ne vaut pas 1, la fissure n'avance pas avec la vitesse V_φ définie.

Après la phase de mise à jour, il arrive que les level-sets perdent leur sens de fonction de distance signée, notamment dans le cas d'une propagation hors plan (cas d'une sollicitation où $K_{II}\neq 0$), qui introduit une vitesse normale (voir Figure 4.1-1).

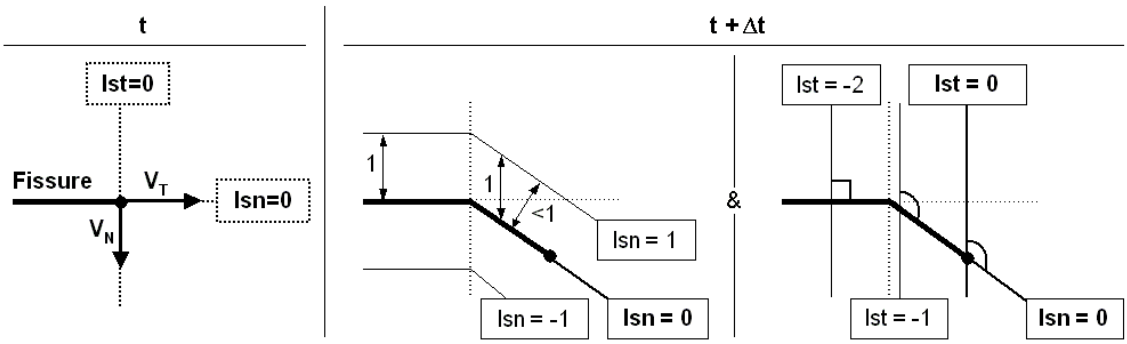


Figure 4.1-1 : Propagation de level-sets et perte de fonction de distance signée et d'orthogonalité

Après propagation il est donc nécessaire de passer par une étape dite de « réinitialisation » dont le but sera de minimiser $\|\nabla lsn|-1|$ et , $\|\nabla lst|-1|$ ainsi qu'une étape de « réorthogonalisation » dont l'objectif sera de minimiser $|\nabla lsn.\nabla lst|$. En effet, les level-sets ne sont rigoureusement définies que si, en tout point de l'espace, $|\nabla lsn|=1$, $|\nabla lst|=1$ et $\nabla lsn.\nabla lst=0$.

Cependant, dans le cas d'une fissure non plane, il est impossible d'avoir deux fonctions dont les gradients sont orthogonaux entre eux et de normes unitaires en tout point de l'espace. L'antagonisme entre les deux propriétés apparaît sur la Figure 4.1-2 sur lesquelles sont définies, à partir d'une iso-zéro de lsn, une fonction lst soit normée, soit orthogonale.

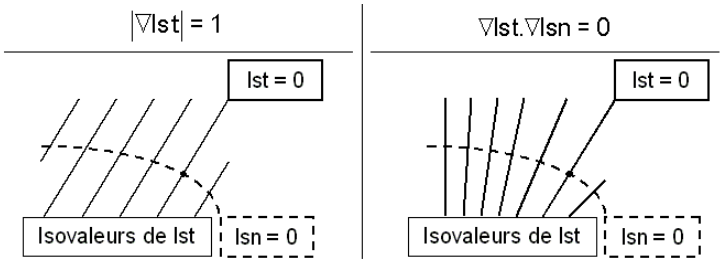


Figure 4.1-2 : Antagonisme Initialisation - Orthogonalisation

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

Il est donc nécessaire de privilégier l'une ou l'autre de ces propriétés. Étant donnée l'utilisation principale de ces fonctions pour la méthode G-Thêta, il apparaît plus important que les gradients soient bien normés, plutôt qu'orthogonaux, afin que les coordonnées polaires (r,θ) aient un sens. Afin d'avoir une base locale orthonormée, on impose aussi sur le fond de fissure que les gradients soient normés et orthogonaux. L'enchaînement des étapes proposé dans (bib1) est donc:

•réinitialisation de lsn :

L'iso-zéro de lsn , d'importance majeure pour la définition de fissure, est inchangée ;

•réorthogonalisation de lst par rapport à lsn fixé :

lst est redressée par rapport à lsn sans modifier le fond de fissure $(lsn=0)\cap(lst=0)$;

•réinitialisation de lst :

La level-set lst est initialisé par rapport à son iso-zéro ayant été orthogonalisée.

4.2 Résolution par la méthode de fast marching

Dans la suite on analysera deux méthodes pour réinitialiser et réorthogonaliser les level-sets. La première méthode (méthode « Upwind ») peut être appliquée seulement à des éléments rectangulaires en 2D et hexaèdres en 3D, une extension à tous types de mailles étant possible avec l'utilisation d'une grille auxiliaire. La deuxième méthode (méthode « Simplexe ») peut être appliquée seulement aux éléments simplexes (triangles en 2D et tétraèdres en 3D), une découpe des mailles sortant de ce cadre permettant d'utiliser cette méthode avec tous types de mailles.

4.2.1 Méthode « Upwind »

On a vu précédemment qu'après la phase de propagation d'une level-set, il fallait réinitialiser celle-ci afin de conserver une fonction distance signée. En d'autres termes, si l'on suppose que la fonction $\varphi(\vec{x},t=0)=\varphi_0(\vec{x})$ vérifie pour chaque nœud du maillage la propriété d'une fonction distance signée donnée par $|\nabla\varphi|=1$, la fonction level-set à $t>0$ n'a plus cette propriété. Il faut donc mettre en place un processus de réinitialisation permettant à φ de redevenir une fonction distance signée.

Nous proposons ici une méthode de fast-marching introduite par *Sethian [19],* qui résout l'équation eikonale de la forme :

$$|\nabla\varphi(x)|=F(x),\ x\in\Omega$$
$$\varphi(x)=0,\ x\in\partial\Omega$$

Avec $\Omega\in\mathbb{R}^n$, $n\in\{2,3\}$ et F une fonction à valeurs positives.

Le point qui nous intéresse ici n'est qu'un cas particulier de cette équation : lorsque $F=1$. La solution donne ainsi la fonction φ distance signée.

Le principe de la méthode fast-marching consiste à propager de proche en proche la donnée de la distance des points depuis les valeurs les plus faibles, c'est-à-dire proches de l'interface, aux zones les plus lointaines.

Pour la propagation des level-sets, on utilise les valeurs autour de $ls=0$ pour propager l'information. Deux appels à la méthode sont nécessaires pour traiter tous les nœuds du maillage : le premier appel traitera les points positifs de la level-set et le second les points négatifs.

Pour cela, la méthode répartit les nœuds du maillage en trois catégories :

- **Frozen points :** région du maillage où les points ont déjà été calculés (cercles noirs) ;
- **Narrow band :** région proche des points déjà calculés (bande grise) ;
- **Far away :** ensemble des points restants (cercles blancs).

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

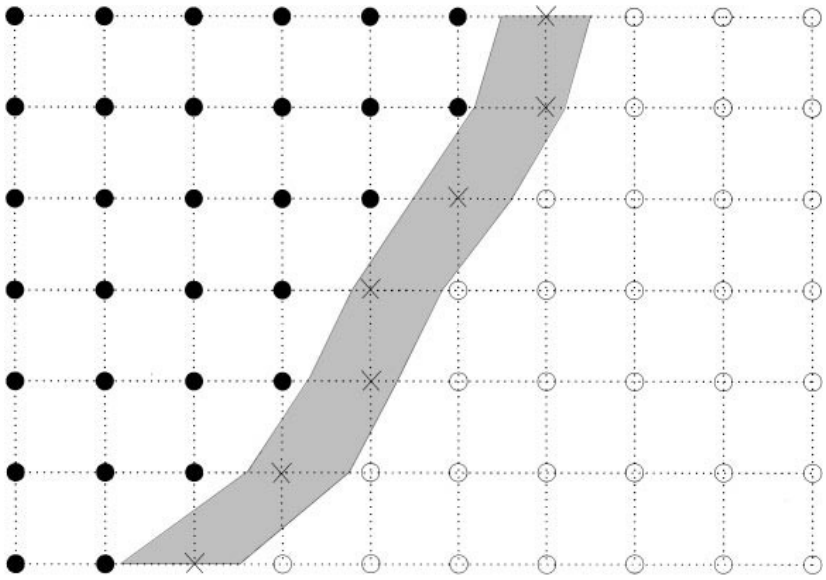


Figure 4.2-1: répartition des nœuds du maillage

La méthode de fast-marching peut se passer de l'étape de réorthogonalisation car une fois la réinitialisation de lsn effectuée, on peut initialiser les valeurs de lst sur son iso-zéro non réorthogonalisée en prenant les valeurs aux nœuds obtenues par un calcul géométrique. On pourra donc utiliser ces valeurs comme condition initiale de la méthode pour propager cette information exact sur lst au domaine tout entier.

4.2.1.1 Discrétisation du problème

Dans la phase de réinitialisation d'une level-set on cherche à résoudre numériquement un cas particulier de l'équation eikonale :

$$|\nabla \phi| = 1 \tag{32}$$

Si on note Δ_x Δ_Y et Δ_Z les pas d'espace en x , y et z , une discrétisation par différence finie upwind de cette équation est :

$$\left(\frac{\varphi_{i,j,k}-\varphi_1}{\Delta_x}\right)^2+\left(\frac{\varphi_{i,j,k}-\varphi_2}{\Delta_y}\right)^2+\left(\frac{\varphi_{i,j,k}-\varphi_3}{\Delta_z}\right)^2=1 \tag{33}$$

A vec :

$$\begin{aligned}&\varphi_1=min\left(\varphi_{i-1,j,k},\varphi_{i+1,j,k}\right)\\&\varphi_2=min\left(\varphi_{i,j-1,k},\varphi_{i,j+1,k}\right)\quad \varphi_n\in\mathbb{R}\quad \forall n\in 1,3\\&\varphi_3=min\left(\varphi_{i,j,k-1},\varphi_{i,j,k+1}\right)\end{aligned} \tag{34}$$

On cherche les solutions du trinôme du second degré qui approxime l'équation (32) sous la forme :

$$p\left(\varphi_{i,j,k}\right)=0 \tag{35}$$

avec :

$$p\left(\varphi_{i,j,k}\right)=\left(\frac{\varphi_{i,j,k}-\varphi_1}{\Delta_x}\right)^2+\left(\frac{\varphi_{i,j,k}-\varphi_2}{\Delta_y}\right)^2+\left(\frac{\varphi_{i,j,k}-\varphi_3}{\Delta_z}\right)^2-1$$

Les deux solutions de (35) sont :

$$\varphi_{i,j,k}=\frac{\varphi_1\Delta_y^2\Delta_z^2+\varphi_2\Delta_x^2\Delta_z^2+\varphi_3\Delta_x^2\Delta_y^2+\sqrt{\left(\varphi_1\Delta_y^2\Delta_z^2+\varphi_2\Delta_x^2\Delta_z^2+\varphi_3\Delta_x^2\Delta_y^2\right)^2-\left(\Delta_x^2\Delta_y^2+\Delta_x^2\Delta_z^2+\Delta_y^2\Delta_z^2\right)\left(\varphi_1^2\Delta_y^2\Delta_z^2+\varphi_2^2\Delta_x^2\Delta_z^2+\varphi_3^2\Delta_x^2\Delta_y^2-\Delta_x^2\Delta_y^2\Delta_z^2\right)}}{\left(\Delta_x^2\Delta_y^2+\Delta_x^2\Delta_z^2+\Delta_y^2\Delta_z^2\right)} \tag{36}$$

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

La distance augmente lorsque nous nous éloignons de l'interface .La solution $\varphi_{i,j,k}$ est évaluée en un point plus éloigné de l'interface que les points aux distances φ_1 , φ_2 et φ_3 . Ainsi nous avons nécessairement que $\varphi_{i,j,k}>\varphi_1$, $\varphi_{i,j,k}>\varphi_2$ et $\varphi_{i,j,k}>\varphi_3$. Or nous avons :

$$\begin{aligned}&\varphi_{i,j,k}(\Delta_x^2\Delta_y^2+\Delta_x^2\Delta_z^2+\Delta_y^2\Delta_z^2)-(\varphi_1\Delta_y^2\Delta_z^2+\varphi_2\Delta_x^2\Delta_z^2+\varphi_3\Delta_x^2\Delta_y^2)\\ >&\varphi_{i,j,k}(\Delta_x^2\Delta_y^2+\Delta_x^2\Delta_z^2+\Delta_y^2\Delta_z^2)-(max(\varphi_1,\varphi_2,\varphi_3)(\Delta_y^2\Delta_z^2+\Delta_x^2\Delta_z^2+\Delta_x^2\Delta_y^2))\end{aligned} \quad (37)$$

Cela implique :

$$(\varphi_{i,j,k}-max(\varphi_1,\varphi_2,\varphi_3))(\Delta_x^2\Delta_y^2+\Delta_x^2\Delta_z^2+\Delta_y^2\Delta_z^2)>0 \text{ car } \varphi_{i,j,k}>\varphi_n \quad \forall n\in 1,3 \quad (38)$$

Donc la solution (36) est nécessairement la seule acceptable.

4.2.1.2 Implémentation de la méthode

La discrétisation numérique de l'équation (32) par différence finie upwind n'est applicable que sur des maillages hexaédriques (ou rectangulaire en 2D) dont les arêtes sont bien orientées selon trois directions (j_1,j_2,j_3) formant une base de $\mathbb{R}^3$ .Comme pour la méthode upwind, on utilise une grille auxiliaire pour étendre la méthode à des maillages irréguliers.

Nous présentons ici uniquement un algorithme pour le cas φ positif ($lsn>0$ ou $lst>0$), une simple multiplication par -1 de cette fonction permettant de traiter l'autre côté de la level-set.

L'algorithme mis en place est le suivant :

- **Initialisation**
  - On repère les nœuds pour lesquels $ls\geq 0$
  - On repère les points proches de $ls=0$ et $ls\geq 0$,c'est-à-dire les nœuds (narrow band) des mailles coupées par $ls=0$, tels que $ls\geq 0$

    Remarque : é *tant donné que l'on souhaite propager le calcul vers les zones lointaines, il faut s'assurer au préalable que ces points fournissent une initialisation consistante du point de vue d'une fonction distance. On effectue un calcul géométrique des level-sets sur les éléments coupés par* $ls=0$ *. On affecte les autres nœuds positifs de la valeur* $+\infty$ *. Ils appartiendront à l'ensemble* ***far*** *,alors que les points négatifs seront inclus dans un troisième ensemble* ***known*** *.*
- **Boucle**
  - Tant qu'un nœud a $ls\geq 0$, on localise le nœud ayant la plus petite valeur de φ dans la narrowband .
  - Pour chaque nœud autour de ce minimum, on récupère φ_1 , φ_2 et φ_3 comme définis par (34).

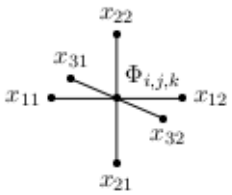


Figure 4.2-2: voisinage du nœud à calculer

  - On calcule pour chaque nœud autour du minimum sa valeur par la formule explicite (36).
- **Fin de la boucle**

4.2.1.3 Fast marching et méthode géométrique

1. **Les n œ uds problématiques**

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

Pour effectuer les calculs par la méthode fast-marching, on a discrétisé l'équation (32) par différences finies. Il faut donc pour calculer la valeur d'un nœud avoir l'information nécessaire pour appliquer la formule explicite (36).Pour chaque nœud calculé, on doit avoir récupéré au préalable φ_1 , φ_2 et φ_3 : si ces valeurs existent alors le calcul par la formule (36) est immédiat. Dans le cas contraire, c'est-à-dire si $\varphi_1=+\infty$ ou $\varphi_2=+\infty$ ou $\varphi_3=+\infty$, on ne peut pas appliquer la formule (36) car le manque d'information entraînera une mauvaise approximation de l'équation (32).

Pour résoudre ce problème, on peut utiliser la méthode géométrique pour le nœud problématique en calculant directement sa valeur. Un simple test dans le code permet ainsi de savoir si le point à calculer est évalué par la méthode fast-marching ou par la méthode géométrique. Pour éviter d'éventuels problèmes, il est judicieux de faire aussi un test sur le signe de la valeur sous la racine dans la formule (36) .

Ainsi si :

$$\begin{aligned}&\left(\varphi_1\Delta_y^2\Delta_z^2+\varphi_2\Delta_x^2\Delta_z^2+\varphi_3\Delta_x^2\Delta_y^2\right)^2-\\&\left(\Delta_x^2\Delta_y^2+\Delta_x^2\Delta_z^2+\Delta_y^2\Delta_z^2\right)\left(\varphi_1^2\Delta_y^2\Delta_z^2+\varphi_2^2\Delta_x^2\Delta_z^2+\varphi_3^2\Delta_x^2\Delta_y^2-\Delta_x^2\Delta_y^2\Delta_z^2\right)>0\end{aligned} \quad (39)$$

Le calcul par la formule (36) est licite. Alors que si :

$$\begin{aligned}&\left(\varphi_1\Delta_y^2\Delta_z^2+\varphi_2\Delta_x^2\Delta_z^2+\varphi_3\Delta_x^2\Delta_y^2\right)^2-\\&\left(\Delta_x^2\Delta_y^2+\Delta_x^2\Delta_z^2+\Delta_y^2\Delta_z^2\right)\left(\varphi_1^2\Delta_y^2\Delta_z^2+\varphi_2^2\Delta_x^2\Delta_z^2+\varphi_3^2\Delta_x^2\Delta_y^2-\Delta_x^2\Delta_y^2\Delta_z^2\right)<0\end{aligned} \quad (40)$$

La formule (36) n'est plus valide et un calcul géométrique sur ce nœud est requis.

2. Réinitialisation de lst

Une fois la réinitialisation de lsn effectuée par la méthode de fast marching, on passe à la réinitialisation de lst sans passer par l'étape de réorthogonalisation. Le principal problème pour cette étape est que l'iso-zéro de lst n'est pas orthogonale à celle de lsn . La fast marching n'étant applicable que sous la condition que la distance augmente en s'éloignant de l'interface, on peut démarrer la méthode autour de l'iso-zéro de la lst non-orthogonalisée en effectuant un calcul géométrique pour ces nœuds afin de récupérer les bonnes valeurs et propager l'information comme défini plus haut. La méthode va ainsi réorthogonaliser la lst et lui attribuer la propriété de fonction distance signée dans un seul et même calcul. Pour augmenter la précision et la localisation du fond de fissure, on effectuera deux réinitialisations de lst à la suite. La première réinitialisation permet de déplacer l'iso-zéro de sorte que celle-ci devienne orthogonale à lsn et la deuxième augmentera la précision sur la localisation du fond de fissure. Le problème peut venir de la position de l'iso-zéro de lst avant la première réinitialisation. Si cette position est trop éloignée de la nouvelle position après réinitialisation, on constate une localisation du fond de fissure moins précise. Avec deux étapes, on s'assure que la localisation de l'iso-zéro de lst ne viendra pas perturber la solution.

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

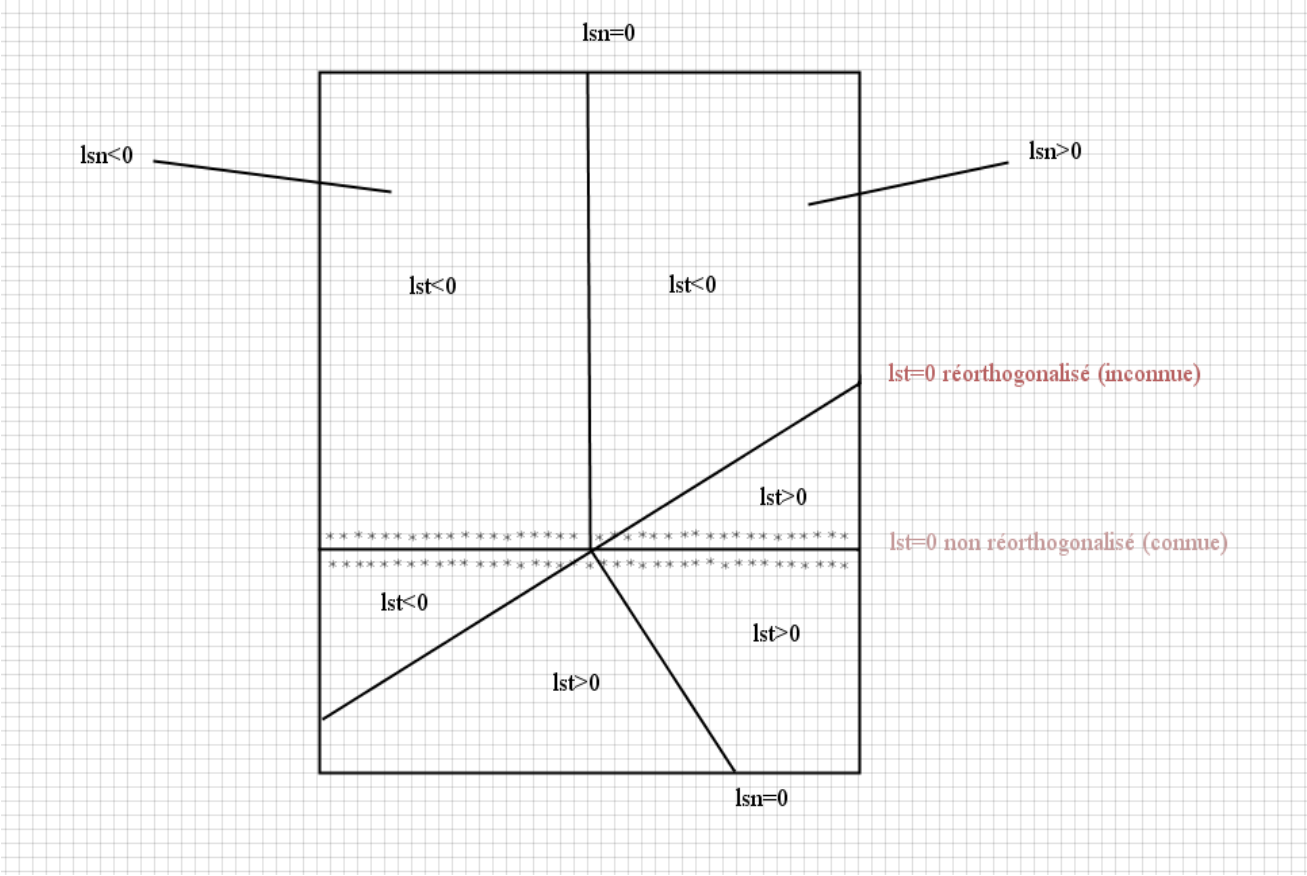


Figure 4.2-3: réinitialisation de lst

4.2.1.4 Application de la méthode upwind à un maillage irrégulier

Pour l'application de la méthode des différences finies, on doit utiliser un maillage très régulier, constitué par des mailles hexaédriques en 3D et des mailles rectangulaires en 2D, dont les arêtes sont bien orientées selon les directions de la base de l'espace $\mathbb{R}^3$ en 3D et $\mathbb{R}^2$ en 2D. Cela représente une très forte restriction pour la création du maillage et donc limite excessivement les cas qui peuvent être traités par la méthode.

Une solution à ce problème consiste à utiliser deux maillages différents (bib 12): un maillage est utilisé pour la structure physique (*maillage physique*) et un maillage est utilisé pour la représentation des level-sets (*maillage level-sets*). Les restrictions de la méthode de différences finies s'appliquent uniquement à ce dernier maillage. Les valeurs des level-sets sur le maillage physique peuvent être calculées par projection des valeurs connues sur le maillage level-sets. Cela est nécessaire pour la localisation du fond de la fissure dans le maillage physique. En fait, les calculs de la mécanique de la rupture sont effectués uniquement sur ce dernier maillage. On peut donc réaliser la mise à jour, la réinitialisation et la réorthogonalisation des level-sets en utilisant uniquement le maillage level-sets. Après on peut projeter les level-sets sur le maillage physique à partir du maillage level-sets. Enfin on peut calculer les facteurs d'intensité des contraintes et la direction de propagation de la fissure en utilisant uniquement le maillage physique.

La projection permet d'obtenir aussi l'indépendance géométrique entre les deux maillages: le maillage des level-sets n'a pas besoin d'être orienté selon les directions de la base choisie pour le maillage physique et il peut uniquement occuper le volume de la zone de propagation de la fissure dans la structure.

Les nœuds du maillage level-set, construit comme décrit ci-dessus, constituent donc un nuage de points alignés selon trois directions orthogonales. La grille formée par ces points peut être utilisée pour l'application de la méthode «upwind». Dans la suite de ce document et dans la syntaxe du *Code_Aster*, on utilisera l'expression ***grille auxiliaire***

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

pour indiquer l'ensemble de ces points. Également, pour simplifier les notations utilisées et les rendre plus claires, on utilisera tout simplement le terme ***maillage*** pour indiquer le maillage physique.

4.2.1.5 Implémentation de la projection entre la grille auxiliaire et le maillage physique

Dans *Code_Aster* la grille auxiliaire n'est pas donnée comme un ensemble de points bien orientés mais comme un maillage et donc quelques étapes intermédiaires sont nécessaires pour l'application de la méthode de différences finies upwind. En effet, pour cette méthode tous les points de la grille, pris coïncidents avec les nœuds du maillage level-set donné, doivent être arrangés selon les directions d'une base locale. On doit donc calculer cette base locale et après on doit ordonner les nœuds et calculer la distance entre deux nœuds successifs.

Le *calcul de la base locale* est très facile parce qu'on peut utiliser la définition des éléments du maillage de la grille auxiliaire. En effet, les arêtes des éléments sont bien orientées par rapport aux axes d'une base locale et on peut définir cette base en prenant des arêtes qui sont orthogonales entre elles. En plus on ne recherche que la direction des axes de la base locale. On peut donc utiliser les arêtes de n'importe quel élément du maillage. En 3D, le premier élément du maillage de la grille auxiliaire permet de déterminer à partir de l'un de ses nœuds, les trois arêtes qui y aboutissent, et ainsi de calculer les trois vecteurs unitaires selon les trois directions des arêtes sélectionnées. En 2D on fait la même opération avec deux arêtes.

Pour *le classement des nœuds* selon les directions des axes de la base locale calculée, on peut utiliser encore une fois les arêtes des éléments. Pour chaque nœud du maillage, on peut sélectionner les éléments qui le contiennent. Pour chacun des éléments sélectionnés, on sélectionne les arêtes qui partent du nœud. Enfin pour chacune des arêtes sélectionnées, on ordonne les points selon une des directions de la base locale du maillage en utilisant le produit scalaire entre l'arête et les vecteurs unitaires de la base locale et on peut calculer la distance entre les nœuds de l'arête. En 3D l'algorithme mis en place est le suivant (tous les vecteurs sont représentés dans la base choisie pour représenter les nœuds du maillage physique):

- On repère les trois vecteurs unitaires de la base locale: $\boldsymbol{j}_1$, $\boldsymbol{j}_2$ et $\boldsymbol{j}_3$
- Boucle sur les nœuds N du maillage
  - On initialise à zéro les valeurs stockées des six nœuds qui sont connectés (successifs ou précédents) au nœud N selon les trois vecteurs unitaires de la base locale
  - On repère les éléments qui incluent le nœud N dans leur définition
  - Boucle sur les éléments sélectionnes
    - On repère les trois arêtes qui partent du nœud N
    - Boucle sur les trois arêtes sélectionnées
      - On calcule le vecteur $\overrightarrow{NA}$ entre les deux nœuds d'extrémité de l'arête. Ce vecteur est orienté selon la direction qui va du nœud N à l'autre nœud A extrémité de l'arête.
      - On calcule les composantes du vecteur $\overrightarrow{NA}$ selon la base locale:

$$NA_1 = \overrightarrow{NA} \cdot \boldsymbol{j}_1$$
$$NA_2 = \overrightarrow{NA} \cdot \boldsymbol{j}_2$$
$$NA_3 = \overrightarrow{NA} \cdot \boldsymbol{j}_3$$

      - On calcule la tolérance qui est utilisée pour vérifier si une composante est nulle. Cette valeur est calculée comme un pourcentage p (fixé à 1% dans *Code_Aster*) de la longueur du vecteur:

$$l_{max} = p \cdot \left| \overrightarrow{NA} \right|$$

      - On repère la composante n plus grande:

$$\Delta = max\left(\left| NA_1 \right|, \left| NA_2 \right|, \left| NA_3 \right|, \right)$$
$$n \in [1,2,3] / \Delta = \left| NA_n \right|$$

      - On vérifie que les deux autres composantes sont nulles ($< l_{max}$), sinon on s'arrête avec un message d'erreur. En fait, cela signifie que les arêtes du maillage ne sont pas toutes alignées selon les trois mêmes directions.
      - Si $NA_n > 0$, alors on stocke le nœud A comme le nœud **successif** du nœud N selon la direction $\boldsymbol{j}_n$. Si $NA_n < 0$, alors on stocke le nœud A comme le nœud **précédent** le nœud N selon la direction $\boldsymbol{j}_n$. On stocke aussi Δ comme la valeur de la distance entre les nœuds A et

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

N selon la direction $\boldsymbol{j}_n$. Si $NA_n=0$, alors on s'arrête avec un message d'erreur parce que cela signifie que plusieurs nœuds de l'élément sont coïncidents.
- Fin de la boucle
- Fin de la boucle
- Fin de la boucle

En 2D on peut utiliser le même algorithme en prenant seulement deux arêtes par nœud et donc deux composantes. Il est très important qu'avant le début de la première boucle on initialise à zéro les valeurs stockées aux six nœuds qui sont connectés au nœud N. En effet, si le numéro stocké d'un de ces nœuds est égal à zéro, on peut en déduire que le nœud N n'a pas un nœud précédent ou successif selon une direction bien définie.

L'arrangement des nœuds selon les directions des axes de la base locale de la grille doit être effectué une seule fois avant l'utilisation de la méthode upwind.

Dans la description ci-dessus, on n'a rien dit sur les modifications que l'on doit apporter au code avant les phases de réinitialisation et de réorthogonalisation des level-sets. En fait, il n'y a pas de grands changements à faire. Le calcul des facteurs d'intensité des contraintes et de la vitesse d'avancée du fond de fissure ne sont pas affectés [§2.1]. Les algorithmes utilisés pour les phases d'extension du champ de vitesse au maillage grille [§2.2] et l'ajustement de la vitesse normale [§2.5] ne changent pas mais on doit donner en entrée aux utilitaires correspondants les nœuds de la grille auxiliaire et pas ceux du maillage physique. Pour cela, les changements du code à faire peuvent être réalisés très rapidement parce que les valeurs de la vitesse d'avancée du fond de fissure sont indépendantes du maillage physique. En fait, le calcul de G, K_I et K_{II} est conduit sur le maillage physique mais les valeurs de ces grandeurs sont liées aux points du fond de fissure qui sont stockés de façon indépendante par rapport au maillage utilisé. Cela permet d'effectuer l'extension du champ de vitesse et l'ajustement de la vitesse normale directement sur la grille auxiliaire, par projection sur le fond de fissure discrétisé par segments, aucune projection de champ de vitesse entre le maillage physique et la grille n'étant nécessaire car ils utilisent tous les deux la même discrétisation du fond de fissure. Enfin, pour la phase de mise à jour, rien ne change dans les utilitaires sauf l'information sur le maillage utilisé (physique ou level-set): les données sont les mêmes et sont stockées aux nœuds du maillage.

En présence de maillages constitués d'un grand nombre d'éléments, la projection entre la grille auxiliaire et le maillage physique peut prendre beaucoup de temps de calcul. Si on regarde l'utilisation des level-sets pour le maillage physique, on peut dire qu'il n'est pas nécessaire qu'on connaisse la valeur des level-sets pour tous les nœuds de ce maillage. En effet, dans le maillage physique, on utilise les level-set pour trois calculs:

1. la représentation du fond du fissure,
2. le calcul de la base locale en chaque point du fond du fissure,
3. le calcul énergétique des facteurs d'intensité des contraintes.

Pour le premier et le deuxième point, il suffit que la valeur des level-sets soit connue pour les nœuds des éléments coupés par le fond de fissure. Pour le dernier point il suffit que la valeur des level-sets soit connue pour tous les nœuds qui sont compris dans le volume du maillage utilisé pour le calcul énergétique. Ce volume coïncide avec un tore construit autour du fond de fissure. Le rayon de ce tore est spécifié par l'utilisateur dans l'opérateur énergétique utilisé pour le calcul, par exemple `CALC_G`.

Les trois calculs ci-dessus définissent trois groupes de nœuds pour lesquels la valeur des level-sets doit être connue. Le groupe défini pour le calcul énergétique (point 3) inclut les autres. Il suffit donc qu'on sélectionne dans chaque maillage (physique et grille auxiliaire) les nœuds qui sont compris dans le tore décrit ci-dessus et qu'on fasse la projection uniquement entre les deux groupes définis par ces nœuds.

On peut sélectionner les nœuds à projeter si la distance entre chaque nœud du maillage et le fond de fissure est connue. Pour la grille auxiliaire cette distance est déjà calculée par la routine qui étend aux nœuds du maillage le champ de vitesse connu en fond de fissure [§2.2]. Dans ce cas, la sélection des nœuds à projeter est donc rapide. Par contre, pour le maillage physique cette distance n'est pas connue et elle doit être calculée en utilisant le même algorithme que celui décrit au [§2.2].

La projection peut être mise en œuvre en utilisant l'opérateur `PROJ_CHAMP` disponible dans *Code_Aster.*

Comme décrit ci-dessus, on a déjà sélectionné les nœuds du maillage physique pour lesquels la valeur des level-sets doit être connue. La sélection est faite en prenant tous les nœuds qui sont compris dans le tore utilisé pour les calculs énergétiques (point 3 ci-dessus). Cela permet d'avoir immédiatement le domaine sur lequel le champ est projeté avec

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

la forme demandée par PROJ_CHAMP. Par contre, un travail supplémentaire est nécessaire pour définir le domaine du champ à projeter. On peut sélectionner les nœuds de ce domaine (sur le maillage level-sets) en utilisant le tore déjà utilisé pour définir l'autre domaine. Ensuite on peut définir le domaine du champ à projeter en cherchant les éléments qui ont au moins un des nœuds sélectionnés avec la technique précédente.

Le dernier problème à résoudre est le suivant: quelle valeur doit-on prendre pour le rayon du tore utilisé pour la sélection des nœuds à projeter ? On a déjà dit que la valeur utilisée dans l'opérateur du calcul énergétique (point 3 ci-dessus) est celle que l'on doit utiliser. Malheureusement cette valeur n'est pas connue de l'opérateur PROPA_FISS. On peut donc utiliser la valeur du paramètre RAYON de cet opérateur qui définit le tore autour du fond de la fissure utilisé pour le calcul du résidu local. Il doit être toujours plus grand que la valeur du rayon utilisé dans l'opérateur du calcul énergétique parce que, après la propagation, les propriétés d'orthogonalité et de distance signée des level-sets sont garanties uniquement dans ce tore. Toutefois la distance à utiliser pour la sélection des nœuds doit être calculée entre chaque nœud et la nouvelle position du fond de la fissure après la propagation. Cette position est définie par les nouvelles valeurs des level-sets après la propagation qui sont connues uniquement pour la grille auxiliaire. La nouvelle position du fond sera connue pour le maillage physique après la projection. La sélection des nœuds intéressés par la projection doit donc forcément être mise en œuvre à partir de la position du fond de la fissure avant la propagation. Le tore à utiliser doit donc inclure toutes les possibles nouvelles positions du fond de fissure et tous les nœuds du tore utilisé pour le calcul du résidu local.

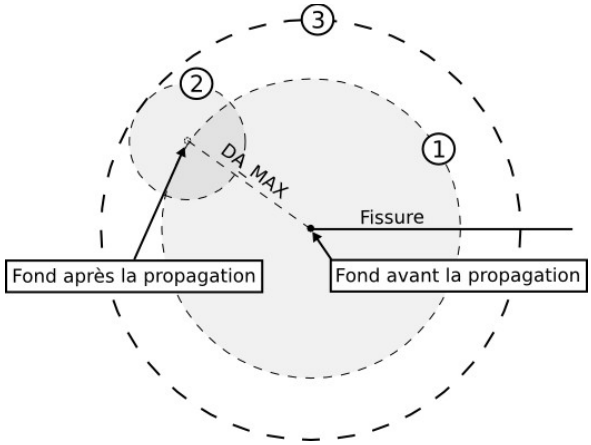


Figure 4.2-4: sélection des nœuds à projeter

On peut regarder l'exemple donné figure 4.2-4 pour une fissure en 2D. L'avancement maximal de la fissure est choisi par l'utilisateur en utilisant le paramètre DA_MAX de l'opérateur PROPA_FISS. La position du fond de la fissure après la propagation est donc comprise dans le cercle de centre coïncidant avec la position du fond avant la propagation et de rayon égal à la valeur de DA_MAX (cercle 1 de la figure). Les nœuds utilisés pour le calcul du résidu local sont compris dans le cercle de centre coïncidant avec la position du fond après la propagation et de rayon égal à la valeur du paramètre RAYON de l'opérateur PROPA_FISS (cercle 2 de la figure). Pour la projection on peut donc sélectionner tous les nœuds compris dans le cercle de centre coïncidant avec la position du fond avant la propagation et de rayon égal à l'addition du DA_MAX et RAYON (cercle 3 de la figure).

Il faut observer enfin que la projection proposée ci-dessus calcule la nouvelle valeur des level-sets uniquement pour les nœuds qui ont été projetés. Toutefois, pour le bon fonctionnement des autres routines X-FEM de *Code_Aster* , il est nécessaire que les level-sets du maillage physique soient définies p our tous les nœuds. Pour cela on peut simplement préserver la valeur des level-sets pour tous les nœuds qui ne sont pas affectés par la projection. À chaque fois, on peut donc prendre les level-sets du maillage physique avant la propagation et on peut changer la valeur des level-sets uniquement pour les nœuds affectés par la projection. Cette solution est très simple à développer dans le code et est très rapide. En plus elle garantit que la surface existante de la fissure et le fond de la fissure sont toujours correctement représentés par les level-sets parce que les résultats de chaque projection sont préservés en arrière du fond de fissure. Un exemple est donné figure 4.2-5 . À gauche de la figure on voit la configuration des level-sets avant la mise à jour. La surface de la fissure est représentée en orange et le fond est marqué avec un petit orange. La configuration des level-sets après la propagation (et la projection) est visible à droite. Seulement les level-sets aux nœuds dans le cercle 3 ont été remplacées par les nouvelles valeurs calculées sur la grille auxiliaire. La nouvelle surface de la fissure est représentée en rouge et la nouvelle position du fond est représentée par un petit rouge. La

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

position du fond avant la propagation est toujours marquée par un point orange. On peut voir que la surface de la fissure qui existait avant la propagation est préservée par la projection (ligne rouge à droite du point orange), ainsi que la continuité de la surface de la fissure entre le domaine de projection (cercle 3) et la partie restante du maillage physique (la ligne rouge est la continuation de la ligne orange). Avec définition du domaine de projection utilisée, les level-sets autour de la position du fond de fissure avant la propagation (point orange) sont toujours affectées par la projection et il n'y a donc pas de risques que de faux fonds de fissure soient détectés ($lsn=lst=0$) après la projection. On peut aussi noter que les level-sets sont discontinues pour le maillage physique mais que cela n'est pas important. En effet, les propriétés d'orthogonalité et de distance signée des level-sets sont garanties pour tous les nœuds impliqués dans les calculs sur le maillage physique.

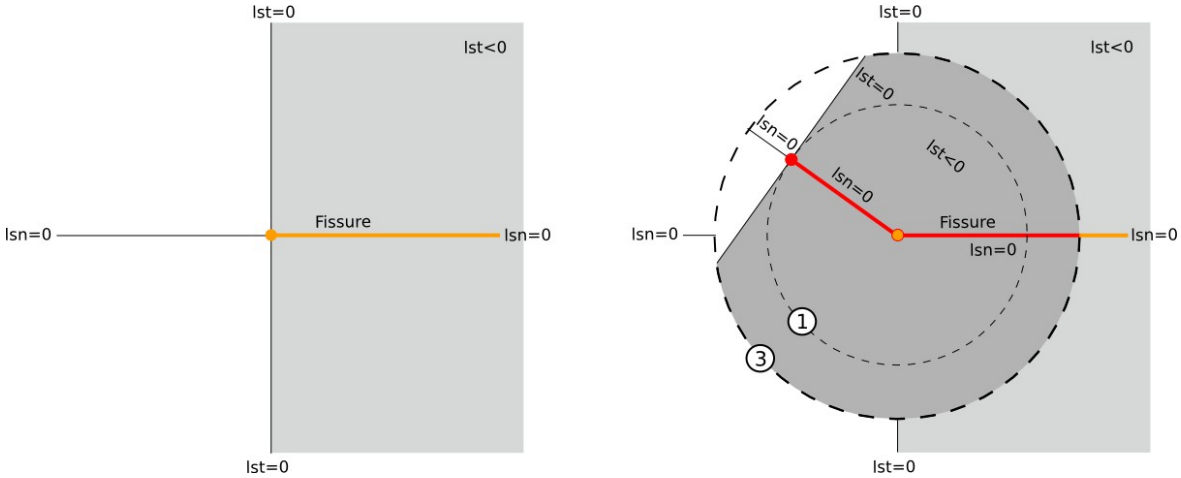


Figure 4.2-5: configuration des level-sets sur le maillage physique avant (à gauche) et après (à droite) la propagation. La zone où la level-set tangente est négative est teintée en gris. La surface et le fond de la fissure sont représentés en orange avant la propagation et en rouge après.

Il faut remarquer que les level-sets sont mises à jour sur tous les nœuds de la grille auxiliaire et que pour ce maillage leur représentation est correcte partout. Donc pour la grille auxiliaire on n'a pas les discontinuités qu'on trouve sur le maillage physique après la projection (figure 4.2-5), sauf dans le cas d'utilisation de la localisation du domaine de calcul décrite au paragraphe [§ 5] , ce qui ne représente pas un problème.

4.2.2 Méthode « Simplexe »

L'utilisation de la grille auxiliaire n'est pas toujours commode pour l'utilisateur, surtout pour des structures complexes. Dans cette optique, une méthode basée sur la même idée que la méthode upwind a été développée. Toujours avec un algorithme de fast marching, nous allons simplement modifier le calcul au niveau de l'élément et travailler directement sur des éléments simplexes [*22*] (triangle, tétraèdre).

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

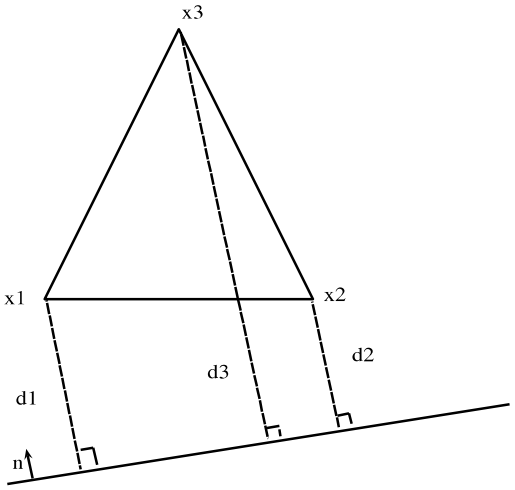


Figure 4.2-6: Méthode simplexe sur un triangle

Pour simplifier les choses, prenons l'exemple d'un cas 2D. On considère une maille triangle avec $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ et $\boldsymbol{x}_3$, les coordonnées des trois nœuds du triangle (voir Error: Reference source not found). En supposant le fond de fissure linéaire par morceaux, on peut projeter chacun des points sur celui-ci. $\boldsymbol{n}$ est donc la normale, inconnue et constante par élément.

En gardant à l'esprit que les trois points se projettent sur une droite de normale $\boldsymbol{n}$, on a d_1 , d_2 et d_3 qui représentent les valeurs de la level-set sur chacun des trois nœuds. d_1 et d_2 étant connues, on cherche donc un moyen de calculer d_3 . L'idée utilisée revient à écrire l'équation de la projection d'un point sur une droite pour $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ et $\boldsymbol{x}_3$. On a :

$$\begin{cases} d_1 = \boldsymbol{x}_1 \cdot \boldsymbol{n} + d \\ d_2 = \boldsymbol{x}_2 \cdot \boldsymbol{n} + d \\ d_3 = \boldsymbol{x}_3 \cdot \boldsymbol{n} + d \end{cases} \tag{41}$$

Ou encore :

$$\begin{cases} d_1 - d_3 = (\boldsymbol{x}_1 - \boldsymbol{x}_3) \cdot \boldsymbol{n} \\ d_2 - d_3 = (\boldsymbol{x}_2 - \boldsymbol{x}_3) \cdot \boldsymbol{n} \end{cases} \Leftrightarrow \begin{cases} d_1 = (\boldsymbol{x}_1 - \boldsymbol{x}_3) \cdot \boldsymbol{n} + d3 \\ d_2 = (\boldsymbol{x}_2 - \boldsymbol{x}_3) \cdot \boldsymbol{n} + d3 \end{cases} \tag{42}$$

On a donc un système de deux équations et trois inconnues. Pour obtenir une solution, on va mettre le système sous forme vectorielle :

$$\boldsymbol{V}^t \boldsymbol{n} + d_3 I = d \tag{43}$$

Avec :

$$\begin{gathered} \boldsymbol{V} = (\boldsymbol{x}_1 - \boldsymbol{x}_{3,} \boldsymbol{x}_2 - \boldsymbol{x}_3) \in M_2(\mathbb{R}) \\ d = (d_1, d_2)^t \\ I = (1,1)^t \end{gathered} \tag{44}$$

V est inversible car les mailles sont non dégénérées, ce qui permet d'écrire l'inconnue n sous la forme suivante :

$$\boldsymbol{n} = \boldsymbol{V}^{-t} (d - d_3 I) \tag{45}$$

$\boldsymbol{n}$ étant un vecteur unitaire, on peut éliminer cette équation de notre système en écrivant que $1 = \boldsymbol{n}^t \boldsymbol{n}$. En utilisant l'expression (45) , on se retrouve avec une équation quadratique où d_3 est la solution :

$$1 = \boldsymbol{n}^t \boldsymbol{n} = (d_3)^2 I^t Q I - 2(d_3) I^t Q d + d^t Q d \quad \text{a vec } Q = (V^t V)^{-1} \tag{46}$$

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

On se retrouve avec deux solutions pour l'équation quadratique. Pour une solution consistante, il faut choisir la plus grande racine de l'équation.

Choix de la solution de l'équation quadratique

La résolution du système (46) conduit à deux solutions :

$$x_1=\frac{I^tQd+\sqrt{(I^tQd)^2-I^tQI\left(d^tQd-1\right)}}{I^tQI} \tag{47}$$

$$x_2=\frac{I^tQd-\sqrt{(I^tQd)^2-I^tQI\left(d^tQd-1\right)}}{I^tQI} \tag{48}$$

Ces deux solutions sont valables, mais pour notre problème, on choisit de garder (47) car c'est la seule qui respecte la condition d'obtenir des valeurs de level-set croissante $d_3>d_2$ ou $d_3>d_1$.

On observe plusieurs configurations possibles dans notre problème. Examinons les toutes pour se convaincre du choix de cette solution.

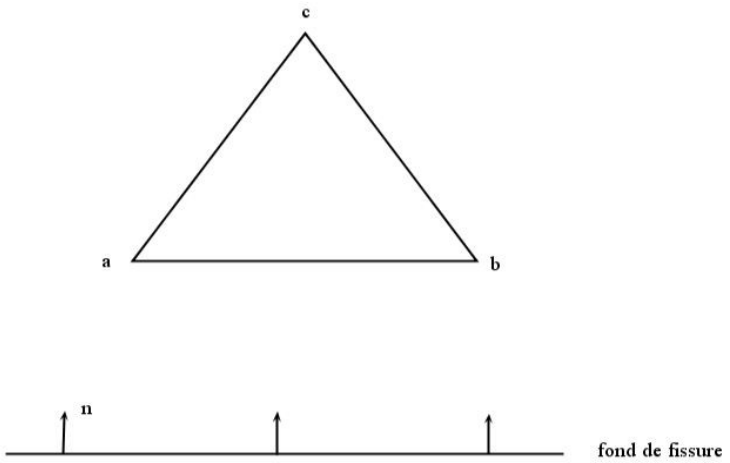


Figure 4.2-7: Première configuration

Dans la première configuration (voir figure (4.2-7)), on connaît la valeur des level-sets aux points a et b et on cherche la valeur au point c . La solution x_1 de (47) donne une normale $\vec{n}_1(0;1)$ et la solution en x_2 de (48) donne une normale $\vec{n}_2(0;-1)$.

La solution (47) est donc celle qui donne la solution exacte de la valeur de la level-set au point c , et qui respecte la condition $ls(c)>ls(b)$ ou $ls(c)>ls(a)$. La valeur obtenue avec la solution (48) dans ce cas-là donne $ls(c)<ls(b)$ et $ls(c)<ls(a)$.

Le domaine de validité de la solution (47) pour notre problème peut être identifié. On regarde le cas ou l'équation quadratique n'a qu'une racine (racine double), voir figure (26).

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

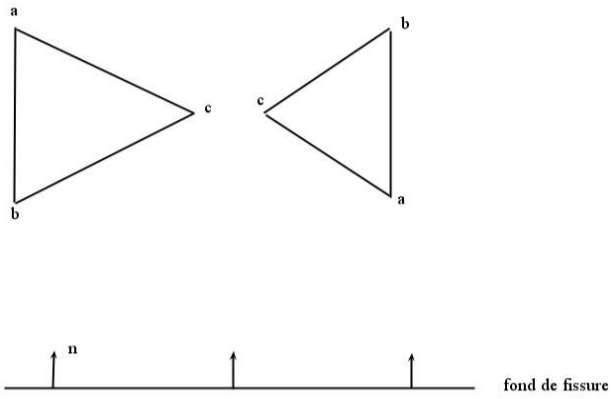


Figure 4.2-8: Deuxième configuration

O n voit les deux configurations d'une maille triangle qui engendrent une solution double pour le problème (discriminant égal à 0). On peut donc identifier les domaines où la solution (47) et la solution (48) sont valables sur la figure (4.2-9).

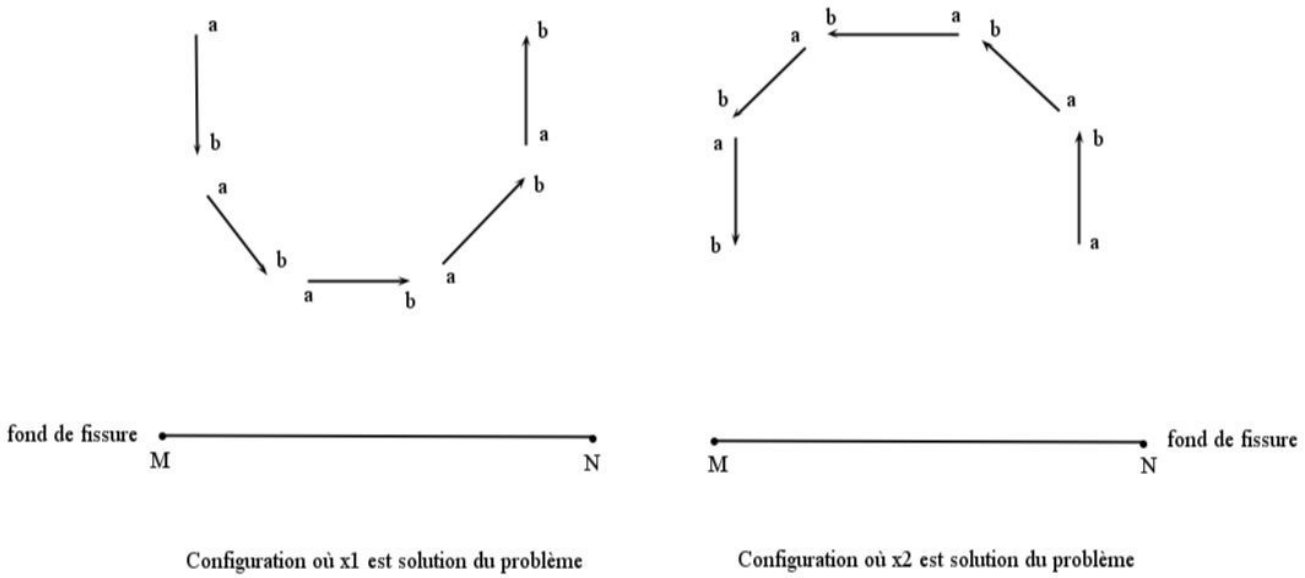


Figure 4.2-9: Troisième configuration

La solution (47) est exacte pour notre problème quand $\overrightarrow{ab}\cdot\overrightarrow{MN}\geq 0$, et la solution (48) quand $\overrightarrow{ab}\cdot\overrightarrow{MN}\leq 0$.

L'idée de la méthode étant de partir des valeurs les plus faibles de level-set, proches de l'iso-zéro et de propager ces valeurs, on voit que la solution (47) est la mieux adaptée. On choisit donc de prendre la plus grande racine de l'équation quadratique dans notre processus de réinitialisation des level-sets, quelle que soit la configuration. La force de la fast marching est de calculer plusieurs fois la valeur de la level-set pour un même nœud dans différentes configurations autour d'un nœud et de garder la meilleure valeur. Cela corrige donc les cas pathologiques où l'on avait pris la solution (47) alors que c'était (48) qui calculait la solution exacte (i.e. il existe au moins une maille contenant le nœud en question pour laquelle la solution (47) est valide).

Pour garantir que la solution est monotone croissante, on peut ajouter un critère au choix de la solution.

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

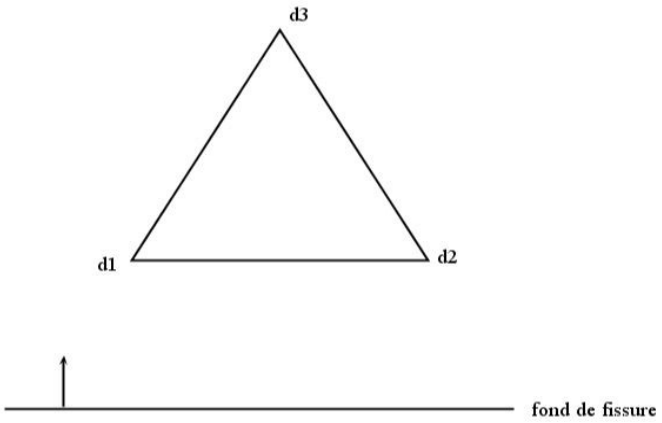


Figure 4.2-10: Quatrième configuration

Soient d_1, d_2 et d_3 les valeurs de la level-set sur une maille triangle quelconque. On souhaite donc que si d_1 et d_2 augmentent alors d_3 augmente aussi. L'idée est de considérer un gradient de level-set :

$$\nabla_d d_3 = \left(\frac{\partial d_3}{\partial d_1} ; \frac{\partial d_3}{\partial d_2} \right) \tag{49}$$

Comme la solution est monotone croissante, on veut $(\nabla_d d_3)_i > 0 , i \in [\![1 ; 2]\!]$. En dérivant l'équation quadratique (46), on a :

$$2 d_3 \nabla_d d_3 . I^t Q I - 2 \nabla_d d_3 . I^t Q d - 2 d_3 Q I + 2 Q d = 0 \tag{50}$$

D 'où :

$$\nabla_d d_3 = \frac{Q(d - d_3 I)}{I^t Q (d - d_3 I)} = \frac{Q V^t n}{I^t Q V^t n} \tag{51}$$

Pour que $(\nabla_d d_3)_i > 0$, il faut que :

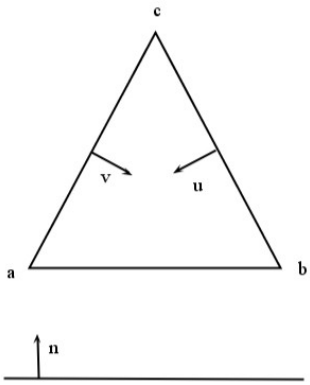
$$\begin{cases} (Q V^t n)_i > 0, i \in [\![1 ; 2]\!] \\ (Q V^t n)_i < 0, i \in [\![1 ; 2]\!] \end{cases} \tag{52}$$

On a vu précédemment que si $d_3 > d_2$ et $d_3 > d_1$, on a $(V^t n)_i < 0$ .Comme Q est symétrique définie positive, la seule possibilité pour que $(\nabla_d d_3)_i > 0$ est que $(Q V^t n)_i < 0, i \in [\![1 ; 2]\!]$. En 2D, cela signifie que $\vec{n}$ doit venir de l'intérieur du triangle.

Preuve

On a $Q V^t V = I$ et on pose $Q V^t = B = \begin{pmatrix} \vec{u} \\ \vec{v} \end{pmatrix}$ .On a donc $BV = I \Leftrightarrow \begin{cases} \vec{u} \cdot \vec{ca} = 1 \\ \vec{u} \cdot \vec{cb} = 0 \\ \vec{v} \cdot \vec{ca} = 0 \\ \vec{v} \cdot \vec{cb} = 1 \end{cases}$

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



Pour satisfaire au problème, on pose $\|\vec{u}\|=\frac{1}{\|\vec{ca}\|\cos(\alpha)}$ et $\|\vec{v}\|=\frac{1}{\|\vec{cb}\|\cos(\beta)}$ où $\cos(\alpha) et \cos(\beta)\neq 0$. On veut $(Bn)_i<0\Rightarrow\begin{cases}\vec{u}\cdot\vec{n}<0\\ \vec{u}\cdot\vec{n}<0\end{cases}\Rightarrow\vec{n}$ vient de l'intérieur du triangle.

4.2.2.1 Algorithme de la méthode simplexe

Le parcours des mailles dans la structure reste similaire à celui de la méthode upwind. On sélectionne le nœud où la level-set atteint son minimum et on explore les triangles autour de celui-ci pour calculer les valeurs inconnues.

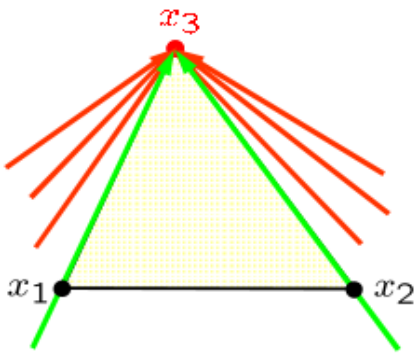
- Pour chaque triangle du maillage, on calcule l'équation quadratique et on garde la plus grande racine
- La direction de propagation $n=V^{-t}(d-d_3 I)$, est utilisée pour vérifier un critère de monotonie.

$$(QV^t n)_i<0, i\in[\![1,2]\!] \tag{53}$$

En remplaçant n dans (53), on a :

$$(Q(d-d_3 I))_i<0, i\in[\![1,2]\!] \tag{54}$$

Ce critère signifie que n doit venir de l'intérieur de triangle. On assure ainsi la monotonie de la solution.



- Si cette condition n'est pas respectée, on prend la valeur minimale d'un coté du triangle.

$$d_3=min\{d_3,\|x1-x3\|_2,\|x2-x3\|_2\} \tag{55}$$

Remarque : La généralisation en 3D est très naturelle, les formules étant les mêmes. La seule différence vient au moment où la condition $(Q(d-d_3 I))_i<0, i\in[\![1,3]\!]$ n'est plus respectée : on calcule alors la plus petite distance sur chaque face du tétraèdre et on garde la plus petite.

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

Contrairement à la méthode upwind, on n'a plus besoin de la méthode géométrique pour les points du bord. La méthode géométrique est utilisée seulement pour obtenir les valeurs d'initialisation de la level-set tangente, toujours dans l'idée que les level-sets doivent rester orthogonales entre elles.

4.3 Calcul des level-sets à proximité des iso-zéros

Les méthodes présentées précédemment ne donnent de résultats exacts que si les iso-zéros des level-sets passent effectivement par des nœuds. Lorsque l'iso-zéro passe entre deux nœuds, elle a tendance à être légèrement déformée lors de la réinitialisation. Un algorithme de calcul des iso-zéros est donc mis en place. Celui-ci calcule géométriquement les distances signées sur les nœuds des mailles coupées ou tangentées par l'iso-zéro de la level-set et fixe la valeur calculée pour la réinitialisation.

Concernant la réorthogonalisation, les valeurs de lst situées sur la surface $(lsn=0)$ ne doivent pas être modifiées ; on profitera donc de l'algorithme de calcul de lsn pour calculer les valeurs de *lst* orthogonales sur les éléments coupés ou tangentés par $(lsn=0)$ et figer ces valeurs pendant la phase de réorthogonalisation. Pour cela, la valeur de lst est interpolée sur les triangles IJK que l'on peut former avec les points d'intersection $(lsn=0)\cap arêtes$. Si pour un nœud M, la projection orthogonale de M est à l'intérieur de IJK, on interpole la valeur de $lst(M)$ comme lst du point projeté. Sinon, $lst(M)$ est calculée comme lst du projeté ramené à l'intérieur du triangle IJK (Figure 4.3-1).

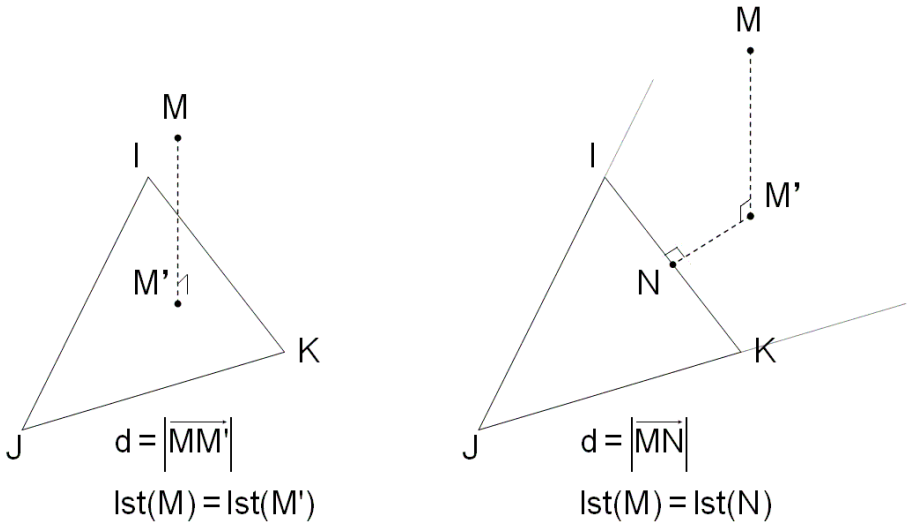


Figure 4.3-1 : Calcul de distance et interpolation de lst sur les éléments coupés par $lsn=0$.

En 3D, l'algorithme mis en place après la propagation de lsn et avant sa réinitialisation est le suivant :

- •Boucle sur les mailles
  - •Si au moins trois nœuds de la maille sont sur $lsn=0$, on la repère comme maille coupée
  - •Si la maille a une arête $[AB]$ telle que $lsn(A).lsn(B)<0$, on repère la maille coupée
- •Fin de boucle
- •Boucle sur les nœuds
  - •Si le nœud appartient à une maille coupée, il sera à calculer
- •Fin de boucle
- •Boucle sur les nœuds P à calculer
  - •Boucle sur les mailles coupées dont le nœud est sommet
    - •Boucle sur les arêtes $[AB]$ de la maille
      - •Si $lsn(A).lsn(B)\leq 0$, on interpole le point I_{AB} :

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

$$\overrightarrow{AI}_{AB}=s.\overrightarrow{AB} \quad \text{avec} \quad s=\frac{|lsn(A)|}{|lsn(A)|+|lsn(B)|}$$

et $lst(I_{AB})=lst(A)+s.(lst(B)-lst(A))$

- Fin de boucle
- Boucle sur tous les triangles IJK que l'on peut former avec les points I_{AB}
  - On calcule le projeté M de P sur le triangle IJK
  - On repère si le point projeté M est à l'intérieur du triangle
  - On stocke la distance $d=PM$
  - On interpole linéairement $lst(M)$ à partir de $lst(I)$, $lst(J)$ et $lst(K)$
- Fin de boucle
- Fin de boucle
- Parmi les distances PM calculées on recherche la plus petite distance avec M à l'intérieur de IJK. Si aucune projection à l'intérieur de IJK n'a été trouvée, on prend juste la plus petite distance calculée et la valeur de lst correspondante.
- On stocke les valeurs de d et lst correspondantes pour le nœud P.
- Fin de boucle
- On remplace lsn et lst sur les nœuds à calculer par les nouvelles valeurs stockées de *d* et *lst*.

Les nœuds repérés comme « nœuds calculés » ont des level-sets qui sont alors figées pendant les phases de réinitialisation de lsn et de réorthogonalisation de lst.

L'algorithme lancé après les phases de propagation et de réorthogonalisation de lst et avant sa réinitialisation est le même, mais il calcule uniquement les distances lst, de la même manière que lsn précédemment, sans toucher aux valeurs de lsn.

L'algorithme décrit ci-dessus est valide seulement en 3D car en 2D on ne peut pas former des triangles. En effet deux points d'intersection seulement peuvent être détectés pour chaque élément.

Pour utiliser en 2D le même algorithme que pour le cas 3D, on peut utiliser la solution suivante. On forme un triangle utilisant les deux points d'intersection (A et B) et un point virtuel (C). Celui-ci est choisi de façon à ce que le triangle formé soit dans un plan orthogonal au plan du modèle (plan $X-Y$). Pour éviter des problèmes numériques, les coordonnés du point virtuel (C) sont calculées de la manière suivante:

- X, Y = coordonnées du point milieu de l'arête reliant les deux points d'intersection (A et B),
- Z = distance entre les deux points d'intersection (A et B).

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

5 Localisation du domaine de calcul des level-sets

Au paragraphe [§4.2.1.5] on a dit que, dans le cas de l'utilisation d'une grille auxiliaire pour la méthode upwind, les valeurs des level-sets sont mises à jour, réinitialisées et réorthogonalisées pour tous les points de la grille mais que seulement les valeurs dans une région autour du fond de la fissure sont projetées sur le maillage de la structure. Cela montre que la mise à jour des level-sets, leur réinitialisation et leur réorthogonalisation sont nécessaires uniquement dans la région autour du fond de la fissure, indépendamment de l'utilisation ou non d'une grille auxiliaire, et que le domaine déjà défini pour la projection entre grille auxiliaire et maillage de la structure ([§4.2.1.5]) peut aussi être utilisé pour localiser le domaine de calcul. Un avantage de cette localisation est évidemment que le temps de calcul nécessaire pour la mise à jour et la réinitialisation des level-sets est inférieur par rapport au cas où tous les nœuds et éléments du maillage ou de la grille auxiliaire sont utilisés. En fait, dans le cas où les level-sets sont mises à jour en des nœuds/points qui sont très loin du fond de la fissure en présence d'oscillations sur l'angle et la vitesse de propagation entre des points successifs du fond de la fissure, les iso-zéros de la level-set normale peuvent présenter des irrégularités très marquées à cause de la variation linéaire imposée à la vitesse normale ([§2.5]). Un exemple illustratif est décrit figure 5-1 et figure 5-2. On y montre une fissure 3D se propageant en mode mixte selon un angle de propagation qui change de façon linéaire le long du fond de la fissure. Des petites perturbations ont été superposées à la valeur théorique attendue, ce qui fait que la variation de l'angle de propagation avec la position sur le fond n'est pas parfaitement linéaire (figure 5-1). Ces perturbations simulent la présence de petites erreurs dans un calcul numérique de valeurs d'angle.

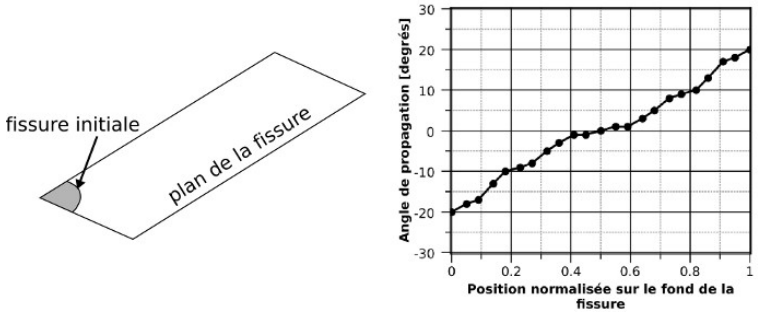


Figure 5-1: configuration initiale (à gauche) de la simulation d'une propagation hors plan d'une fissure 3D selon un angle de propagation qui change de façon presque linéaire par rapport à la position considérée sur le fond de la fissure (graphe à droite).

Même si les perturbations de l'angle de propagation, et donc de la vitesse normale, sont petites, elles sont amplifiées par la variation linéaire imposée à cette composante de la vitesse dans le domaine de calcul, ce qui est très clair si on visualise l'iso-zéro de la level-set normale après la seule phase de mise à jour (figure 5-2). La localisation du domaine de calcul permet de limiter cet effet (figure 5-3).

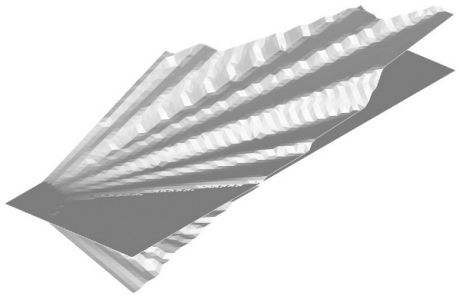
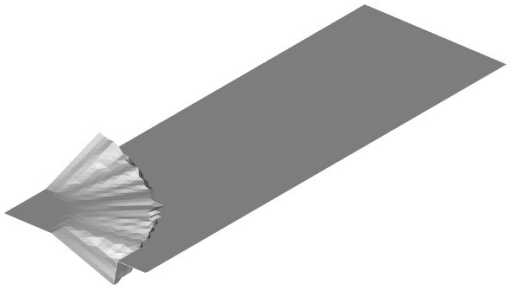


Figure 5-2: iso-zéro de la level-set normale après la seule phase de mise à jour. L'angle de propagation de la figure 5-1 a été imposé. Le plan de la fissure initiale a été superposé pour comparaison.

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

Figure 5-3: iso-zéro de la level-set normale obtenue avec une restriction du domaine de mise à jour à comparer à celle de la figure 5-2. Il faut remarquer que pour cette figure le plan de la fissure initiale n'a pas été représenté, au contraire de ce qui a été fait pour la figure 5-2. On peut donc en conclure que la valeur de la level-set n'a pas été changée pour les points en dehors du domaine de calcul.



5.1 Sélection des nœuds/points et des éléments du domaine de calcul localisé

Le domaine à utiliser pour la localisation est celui déjà défini pour la projection entre la grille auxiliaire et le maillage de la structure ([§4.2.1.5]). En effet ce domaine est étendu à tous les points utilisés pour les calculs mécaniques et il permet de préserver la surface de la fissure existante, comme déjà montré (figure 4.2-5). Les méthodes seront appliquées seulement aux nœuds et aux éléments contenus dans ce domaine, qui est défini par le cercle 3 de figure 4.2-4. En 3D ce domaine coïncide avec un tore construit autour du fond de la fissure et de rayon égal à celui du cercle 3. Dans tout les cas le rayon du domaine localisé est donc égal à:

$$R_{loc}=\Delta a+R_{calcG}$$

où Δa est l'avancée de la fissure à imposer (choisie par l'utilisateur) et R_{calcG} est le rayon du domaine utilisé par CALC_G.
On peut donc tout d'abord sélectionner les nœuds de la grille ou du maillage qui ont une distance au fond de la fissure inférieure ou égale à R_{loc} (figure 5.1-1 à gauche). Cette distance peut être calculée de façon très rapide parce que pour chaque nœud on connaît déjà son projeté sur le fond de fissure ([§ 2.2]). Puis on prend tous les éléments qui ont au moins un des nœuds sélectionnés précédemment (éléments gris de la figure 5.1-1 de droite). Cela permet de sélectionner correctement tous les éléments qui sont coupés par la frontière du domaine et qui sont donc partiellement contenus dans le domaine. De cette façon, toute la surface (ou le volume en 3D) du domaine est correctement recouvert par les éléments sélectionnés. Enfin la liste des nœuds du domaine doit être mise à jour en ajoutant les nœuds des éléments partiellement contenus dans le domaine (nœuds marqués avec un petit cercle sur la figure 5.1-1 de droite). La façon la plus rapide et facile de le faire est de reconstruire la liste en prenant tous les nœuds des éléments sélectionnés. Une fois que la liste des nœuds à été mise à jour, la valeur du rayon R_{loc} du domaine discrétisé, c'est-à-dire du domaine constitué par les nœuds et les éléments sélectionnés, doit être mise à jour en prenant la valeur au nœud du domaine discrétisé le plus loin du fond. Cette opération est nécessaire pour la partie qui va suivre ([§5.3]).

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

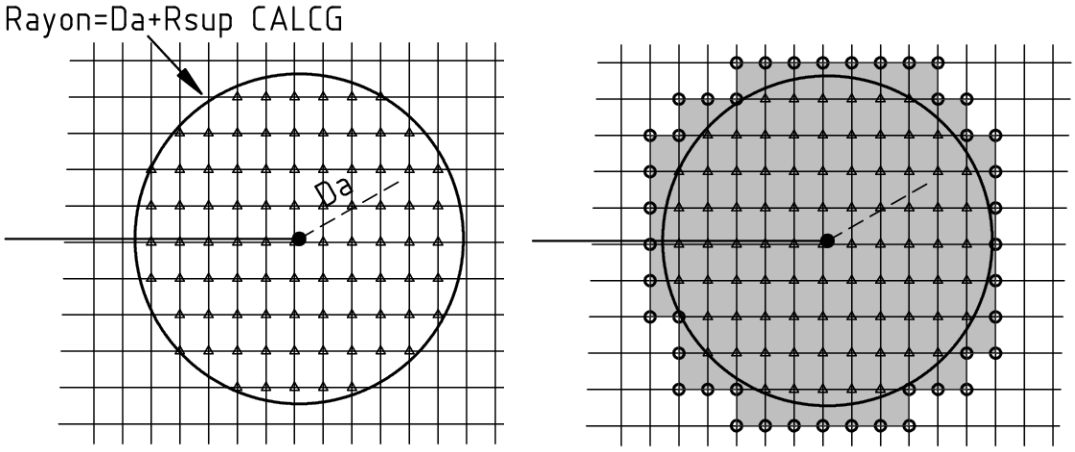


Figure 5.1-1: exemple de sélection des nœuds et des éléments du domaine de localisation. On suppose que ce domaine coïncide avec le cercle de la figure. La première étape (à gauche) est la sélection des nœuds à une distance inférieure ou égale à R_{loc}. Les nœuds sélectionnés sont marqués avec un petit triangle. Ensuite les éléments qui contiennent au moins un des ces nœuds dans leur définition sont sélectionnés (éléments gris à droite). Enfin on sélectionne tous les nœuds des éléments qui ont été pris: les nœuds marqués avec des petits cercles viennent s'ajouter à ceux qui ont été sélectionnés au début.

5.2 Déplacement du domaine localisé en suite à la propagation de la fissure

La mise à jour et la réinitialisation des level-set sont faites exclusivement aux nœuds définissant le domaine localisé (nœuds marqués avec des triangles et des cercles sur la figure 5.1-1 de droite). Cela signifie que les valeurs des level-set aux nœuds à l'extérieur du domaine localisé ne sont pas changées, ce qui pose des problèmes au cours des pas de propagation successifs. La situation est illustrée figure 5.2-1 pour un point P_1 dans le plan $t-n$ (plan e_1-e_2 de la figure). Au début de la propagation le fond de la fissure coïncide avec le point P_1 et le domaine localisé avec le cercle Ω_1. Pour simuler la propagation, les level-set sont mises à jour et réinitialisées seulement dans ce cercle. La nouvelle position du fond après le premier pas de propagation coïncide avec P_2. Au deuxième pas de propagation, le fond de la fissure coïncide avec le point P_2 et un nouveau domaine Ω_2 est construit autour de ce point pour la mise à jour et la réinitalisation des level-sets. La valeur des level-sets n'a pas été mise à jour au pas de propagation précèdent pour tous les points de Ω_2 : seulement les level-sets des points de Ω_2 en commun avec Ω_1 sont à jour (points de la surface grise de la figure 5.2-1). Pour tous les autres points, la valeur des level-sets est celle que l'on avait au début de la propagation. Les bonnes valeurs en ces autres points doivent donc être calculées au début du deuxième pas de propagation.

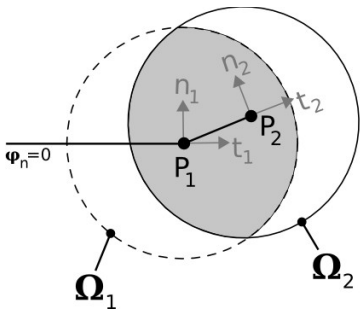


Figure 5.2-1: déplacement du domaine localisé entre deux pas de propagation successifs. Le fond de la fissure se déplace de P_1 à P_2.

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

Le calcul des bonnes valeurs des level-sets pour tous les points dans la région blanche du domaine Ω_2 de la figure 5.2-1 peut être fait en utilisant la définition de distance signée des level-sets (bib14). Pour chaque point M dans cette région (figure 5.2-2), les level-set peuvent être calculées comme ceci:

$$\varphi_n(M)=\overrightarrow{MP_2}\cdot\boldsymbol{n_2}$$
$$\varphi_t(M)=\overrightarrow{MP_2}\cdot\boldsymbol{t_2}$$

Ces équations permettent de calculer la bonne valeur des level-sets avec le bon signe. Il faut remarquer que, dans le cas 3D, le point P_2 est le point projeté du point M sur le fond de la fissure. Pour chaque point M, son projeté et la base n_2-t_2 ont déjà été calculés pendant la phase d'extension de la vitesse du fond de la fissure aux points du domaine de calcul ([§2.2]). Tous les ingrédients du calcul sont donc déjà disponibles et pour chaque point M l'évaluation de la bonne level-set est limitée au calcul de deux produits scalaires, ce qui est très rapide à faire.

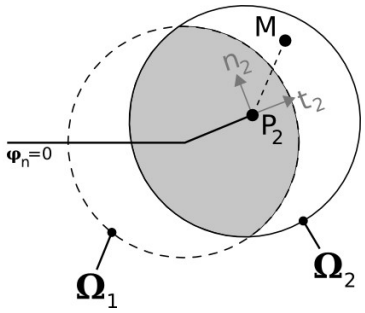


Figure 5.2-2: calcul des bonnes valeurs des level-sets aux points de Ω_2 qui sont à l'extérieur du domaine Ω_1 du pas de propagation précédent. Le calcul est fait tout au début de la propagation courante, c'est-à-dire avant la mise à jour et la réinitialisation des level-sets.

Il faut remarquer que même si les valeurs des level-sets calculées par les équations ci-dessus ne sont pas très précises, la phase de mise à jour ne sert qu'à l'initialisation de ces dernières et les bonnes level-sets seront récupérées par les phases de réinitialisations.

5.3 Limitations sur la valeur du rayon du domaine localisé entre deux propagation successives

Les équations données au paragraphe 33 pour le calcul des bonnes valeurs des level-sets peuvent être appliquées seulement aux points pour lesquels la base n_2-t_2 est valide pour le calcul de la distance signée. Cela signifie que le point M doit appartenir au demi-plan montré figure 5.3-1, qui contient la base n_2-t_2, le fond de la fissure P_2 et qui a pour bord une droite normale à l'arête reliant P_1 et P_2 parallèle à n_2.

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

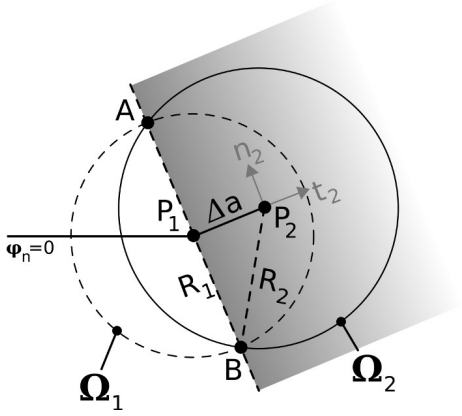


Figure 5.3-1: définition du demi-plan auquel le point M de la figure 5.2-2 doit appartenir pour que les équations de calcul des level-sets du paragraphe 33 soient valides.

Cette condition sur la position du point M se traduit par une condition sur la valeur maximale du rayon R_2 du domaine Ω_2 : la valeur maximale est celle pour laquelle les deux points d'intersection (A et B) entre les deux domaines Ω_1 et Ω_2 sont sur le bord du demi-plan de validité des équations, comme indiqué figure 5.3-1. Ces deux points sont toujours symétriques par rapport à l'arête reliant P_1 et P_2 . La valeur maximale du rayon R_2 peut donc être calculée facilement:

$$R_2 \leq \sqrt{\Delta a^2 + R_1^2}$$

La valeur du rayon du domaine actuel est donc bornée par les valeurs de l'avancée de la fissure et du rayon du domaine utilisés au pas de propagation précédent. Normalement il n'y a aucun intérêt à changer la valeur du rayon du domaine entre deux propagations successives. Dans ce cas, comme R_1 est égal à R_2 , l'inégalité sur R_2 est nécessairement vérifiée quelle que soit la valeur de l'avancée choisie.
Il faut toutefois remarquer que la valeur du rayon du domaine peut changer par rapport à la valeur théorique donnée par l'utilisateur du fait de la discrétisation du domaine ([§ 5.1] et figure 5.2-1 de droite). Donc, même si on donne $R_2 = R_1$ au pas de propagation courant, la valeur effective du rayon du domaine sera la suivante:

$$R_{2\,\mathrm{eff}} = R_2 + \Delta R_2 = R_1 + \Delta R_2$$

où ΔR_2 est l'incrément du rayon R_2 du fait de la discrétisation du domaine, qui est de l'ordre de la longueur moyenne des arêtes des éléments autour de la frontière du domaine localisé (voir figure 5.2-1).
Pour la même raison, la valeur effective du rayon R_1 au pas de propagation précédent peut être différente de la valeur théorique donnée par l'utilisateur:

$$R_{1\,\mathrm{eff}} = R_1 + \Delta R_1$$

où ΔR_1 est l'incrément du rayon R_1 du fait de la discrétisation du domaine, qui est de l'ordre de la longueur moyenne des arêtes des éléments autour de la frontière du domaine localisé au pas de propagation précédent.
Ainsi l'inégalité ci-dessus peut être écrite dans la forme suivante:

$$R_1 + \Delta R_2 \leq \sqrt{\Delta a^2 + \left(R_1 + \Delta R_1\right)^2}$$

Dans le cas où la longueur des arêtes de la grille auxiliaire utilisée est uniforme dans le domaine, $\Delta R_2 \approx \Delta R_1$, et l'inégalité est toujours vérifiée. Par contre, si cette longueur change dans le domaine, on pourrait avoir $\Delta R_2 > \Delta R_1$ et l'inégalité pourrait ne pas être vérifiée.

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

Dans ce dernier cas on doit forcément raffiner le maillage (ou la grille auxiliaire) pour réduire la différence entre ΔR_1 et ΔR_2 ou bien utiliser une avancée de fissure Δa plus élevée.

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

6 Calcul direct des nouvelles level-sets après avancée

Une méthode alternative aux méthodes de mise à jour des level-sets est celle décrite dans (bib 18) . Elle est dérivée de l'analyse des équations de mise à jour des level-sets décrites aux paragraphes 3.1 : on arrive à montrer que le processus de mise à jour des level-sets peut toujours être ramené à un processus 2D en mode $I+II$ dans le plan formé par la tangente et la normale à la surface de la fissure en fond de fissure (plan $n_P - t_P$ de la figure 2.3-1). Cela permet de simplifier énormément le calcul direct des nouvelles valeurs des level-sets après avancée. Ce calcul a été effectivement déjà explicité au paragraphe §5.2 pour ce qui concerne l'évaluation des level-sets pour les points hors domaine de calcul qui sont rajoutés. Les équations qui y sont décrites peuvent être utilisées dans le cas où la nouvelle position du fond est connue. En sachant que le problème de mise à jour des level-sets peut toujours être ramené à un problème 2D, la nouvelle position du fond peut être facilement calculée à partir de l'avancée de la fissure et de l'angle de propagation au point du fond considéré.

6.1 Méthode de calcul des level-sets

Ces idées sont reprises figure 6.1-1 , qui montre la configuration des iso-surfaces des deux level-sets autour d'un point P du fond d'une fissure 3D obtenue par le processus de mise à jour décrit au paragraphe § 3.1

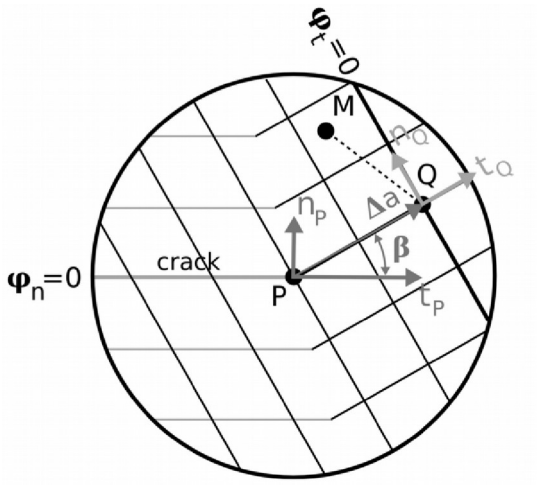


Figure 6.1-1: iso-surfaces des deux level-sets obtenues par le processus standard de mise à jour, avec réinitialisation et réorthogonalisation. P est un point du fond d'une fissure 3D. Q est la nouvelle position de ce point après une avancée de Δa avec un angle de propagation β .

On considère un point M du domaine de calcul où on doit calculer la nouvelle valeur des deux level-sets. On projette ce point sur le fond de la fissure en utilisant le même algorithme que celui décrit au paragraphe 2.2 et on obtient le point P (figure 6.1-1). La base locale (t_P, n_P) et la vitesse d'avancée V_P au point P peuvent être calculées comme décrit aux paragraphes 2.2 , 2.3 et 2.4 . Pour l'angle de propagation, on utilise la même interpolation linéaire que celle utilisée pour le vitesse (voir paragraphe §2.2 et figure 2.2-2:

$$\beta = \left(\beta_J - \beta_I\right) \cdot s_{min} + \beta_I \tag{56}$$

L'avancée de la fissure au point P du fond peut donc être aisément calculée :

$$\Delta \vec{a} = \left\| \boldsymbol{V}_P \right\| \cdot \Delta t \cdot \left(\sin\beta \cdot \boldsymbol{n}_P + \cos\beta \cdot \boldsymbol{t}_P \right) \tag{57}$$

Où Δt est le temps ou, en termes équivalents, le nombre de cycles de fatigue à simuler au pas de propagation courant (voir discussion du paragraphe 3.1).

La nouvelle position Q du point P du fond peut donc être calcul ée comme ceci :

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

$$\overrightarrow{OQ}=\overrightarrow{OP}+\Delta\,\vec{a} \tag{58}$$

Où O est le point origine de la base globale dans laquelle les coordonnées des points sont exprimées.

Enfin la base au point Q peut être calculée par une rotation d'angle β de la base au point P autour de l'axe normal au plan t_P-n_P :

$$\begin{aligned}\boldsymbol{t}_{\mathrm{Q}}&=\cos\beta\cdot\boldsymbol{t}_{\mathrm{P}}+\sin\beta\cdot\boldsymbol{n}_{\mathrm{P}}\\ \boldsymbol{n}_{\mathrm{Q}}&=-\sin\beta\cdot\boldsymbol{t}_{\mathrm{P}}+\cos\beta\cdot\boldsymbol{n}_{\mathrm{P}}\end{aligned} \tag{59}$$

Tous les ingrédients du calcul des level-sets sont maintenant connus. On peut donc utiliser les mêmes équations que celles utilisées au paragraphe 5.2 :

$$\varphi_n(\mathrm{M})=\overrightarrow{\mathrm{QM}}\cdot\boldsymbol{n}_{\mathbf{Q}}\ \text{ et }\ \varphi_t(\mathrm{M})=\overrightarrow{\mathrm{QM}}\cdot\boldsymbol{t}_{\mathbf{Q}} \tag{60}$$

Afin de préserver la surface existante de la fissure (iso-surfaces en gris de la figure 6.1-1), le calcul de la level-set normale avec l'équation ci-dessus est fait uniquement aux points du domaine où $\varphi_t(\mathrm{M})>0$.

6.2 Traitement des points problématiques

Les équations écrites au paragraphe précédent s'appliquent si le vecteur $\overrightarrow{\mathrm{PM}}$ est dans le plan t_P-n_P (figure 6.1-1) (bib18). Cela n'est pas le cas pour tous les points qui n'ont pas un point de projection normale sur le fond de la fissure. Dans ce cas un point extrémité du fond est sélectionné comme point projeté P par l'algorithme de projection décrit au paragraphe 2.2.

Ces points problématiques peuvent être traités par une rotation de la base (t_P,n_P) qui permet de rétablir la condition de coplanarité du plan t_P-n_P et du vecteur $\overrightarrow{\mathrm{PM}}$ ou $\overrightarrow{\mathrm{QM}}$ (bib 18).

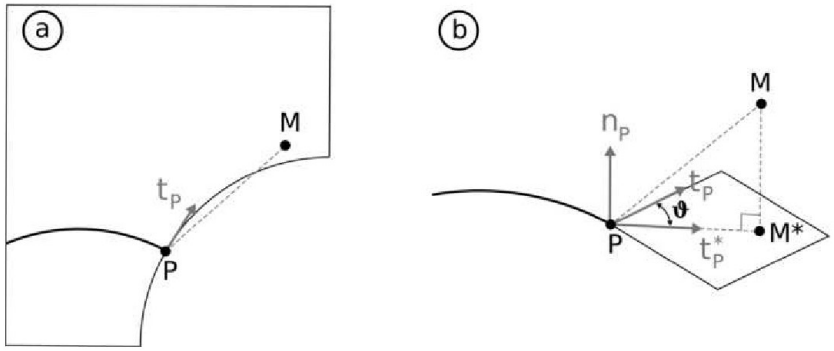


Figure 6.2-1: a) exemple de point problèmatique. b) rotation de la base au point P pour rétablir la condition de coplanarité entre le plan t_P-n_P et les vecteurs $\overrightarrow{\mathrm{PM}}$ ou $\overrightarrow{\mathrm{QM}}$.

Un exemple de point problématique est visible sur la figure 6.2-1a. Pour rétablir la condition de coplanarité, on peut faire une rotation θ de la base (t_P,n_P) autour du vecteur n_P de telle manière que le plan $t_P^*-n_P$ contienne le point M (figure 6.2-1 b). En effet, de cette façon la nouvelle position du point P après propagation (point Q de la figure 6.1-1) est contenue dans le plan $t_P^*-n_P$, ce qui permet de vérifier la condition de coplanarité.

Le nouveau vecteur $\boldsymbol{t}_{\mathrm{P}}^*$ peut être calculé à partir du projeté M^* du point M sur le plan normal à l'axe n_P (figure 6.2-1 b). On calcule d'abord :

$$\overrightarrow{\mathrm{M^*M}}=\left(\overrightarrow{\mathrm{PM}}\cdot\boldsymbol{n}_{\mathrm{P}}\right)\cdot\boldsymbol{n}_{\mathrm{P}} \tag{61}$$

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

Ce qui permet d'obtenir le nouveau vecteur :

$$\boldsymbol{t}_{\mathrm{P}}^{*}=\frac{\overrightarrow{\mathrm{PM}}-\overrightarrow{\mathrm{M}^{*}\mathrm{M}}}{\|\overrightarrow{\mathrm{PM}}\|-\|\overrightarrow{\mathrm{M}^{*}\mathrm{M}}\|} \tag{62}$$

Pour prendre en compte correctement le signe de la level-set tangente, l e vecteur $\boldsymbol{t}_{\mathrm{P}}^{*}$ doit être réorienté suivant l a direction d onnée par $\boldsymbol{t}_{\mathrm{P}}$ (bib 18) :

$$\boldsymbol{t}_{\mathrm{P}}^{'}=\frac{\boldsymbol{t}_{\mathrm{P}}^{*}\cdot\boldsymbol{t}_{\mathrm{P}}}{\|\boldsymbol{t}_{\mathrm{P}}^{*}\cdot\boldsymbol{t}_{\mathrm{P}}\|}\cdot\boldsymbol{t}_{\mathrm{P}}^{*} \tag{63}$$

Si les deux vecteurs $\boldsymbol{t}_{\mathrm{P}}^{*}$ et $\boldsymbol{t}_{\mathrm{P}}$ sont orthogonaux, l'équation ci-dessus ne peut pas être appliquée mais on peut écrire tout de suite que :

$$\boldsymbol{t}_{\mathrm{P}}^{'}=\boldsymbol{t}_{\mathrm{P}}^{*} \tag{64}$$

L'utilisation du vecteur $\boldsymbol{t}_{\mathrm{P}}^{'}$ à la place de $\boldsymbol{t}_{\mathrm{P}}$ dans les équations du paragraphe §6.1 permet de bien calculer les level-sets pour les points problématiques.

6.3 Discontinuité de la level-set normale et limitations de calcul

Le fait de calculer la level-set normale seulement aux points M pour lesquels $\varphi_t(\mathrm{M})>0$ entraîne une discontinuité de la level-set normale, comme démontré dans (bib 18). La situation est illustrée figure 6.3-1 . La level-set normale n'a pas été recalculée dans la zone colorée en gris impliquant une discontinuité des iso-surface entre la zone en gris et la zone où la level-set a été récalculée.

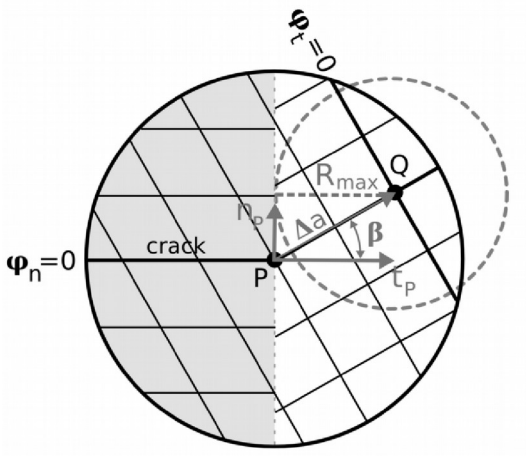


Figure 6.3-1: discontinuité des iso-surface de la level-set normale et rayon maximal R_{max} du domaine de calcul de CALC_G à utiliser.

Comme démontré dans (bib 18), cette discontinuité ne cause aucun problème pour les calculs X-FEM (identification des nouvelles mailles coupées notamment). De plus, la continuité de l'iso-zéro de la level-set normale est préservée, ce qui assure la continuité de la surface de la fissure. La seule condition à respecter pour gérer cette discontinuité porte sur la valeur maximale du rayon de calcul de CALC_G (voir U4.82.03). Cette condition a pour but d'éviter que le domaine de calcul de CALC_G contienne la discontinuité (cercle gris de la figure 6.3-1) :

$$R_{max}=\Delta a_{max}\cdot\cos\beta_{max} \tag{65}$$

Où on prend la valeur maximale de Δa et β le long du fond de la fissure.

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

Si elle restreint l'angle de propagation à des valeurs strictement inférieures à $\pi/2$ (sous peine d'un raffinement de plus en plus important à mesure que l'on se rapproche de $\pi/2$), cette condition n'est pas très restrictive parce qu'il est conseillé de respecter une relation similaire même dans le cas de mise à jour des level-sets par la méthode simplexe ou upwind.

6.4 Localisation du domaine

Comme montré dans (bib18), la localisation du domaine décrite au paragraphe §5 n'est plus nécessaire si on utilise le calcul direct des level-sets.

Tout d'abord, la localisation du domaine n'améliorerait pas la robustesse de l'algorithme. Dans (bib14) , il est montré que de faibles erreurs dans le calcul de l'angle de propagation (ce qui est toujours le cas lors de simulations numériques) sont amplifiées aux points situés loin du front de fissure (du fait de l'interpolation linéaire utilisée pour créer le champ de vitesse V_N). Les iso-surfaces de la level-set normale ne sont alors plus régulières et ceci peut entraîner l'échec des phases de réinitialisation des level-sets . La localisation du domaine permet ainsi de réduire le domaine de mis à jour des level-sets à un domaine englobant le front de fissure ce qui conduit à une limitation de l'effet d'amplification. Cependant la méthode géométrique gère avec succès les iso-surfaces non régulières puisqu'elle est basée sur des équations polynomiales explicites et non sur des équations différentielles. De plus, la méthode géométrique n'utilise pas de processus de réinitialisation et de réorthogonalisation des level-sets (qui sont calculés à chaque pas de propagation). Pour toutes ces raisons, la localisation du domaine n'affecte pas la robustesse de la méthode géométrique.

La localisation du domaine dans le cadre de l'utilisation de la méthode géométrique ne réduirait pas non plus forcément le temps de calcul nécessaire à la mise à jour des level-sets. Il s'avère que le calcul des level-sets en chaque point du domaine est rapide et que le temps nécessaire pour ce calcul peut être plus faible que le temps nécessaire pour sélectionner les points du domaine localisé. L'algorithme utilisé pour la sélection des points appartenant au domaine restreint est basé sur le calcul de la distance entre un point M du domaine et son projeté P sur le front de fissure. Comme le point P doit être calculé également pour le calcul des level-sets, le coût de calcul supplémentaire pour l'algorithme de sélection se réduit au coût du calcul de la distance $\|\overrightarrow{PM}\|$ et la complexité de l'algorithme est de $O(n)$. Même si la complexité de l'algorithme de mise à jour des level-sets par la méthode géométrique est de $O(n)$, il apparaît clair que la somme des calculs effectués par l'algorithme de sélection sera moindre que celle effectuée par l'algorithme de mise à jour des level-sets. Ceci conduit à penser à première vue que la localisation du domaine serait avantageuse en terme de temps de calcul par rapport au calcul des level-sets dans tout le domaine. Cependant, la sélection (basée sur la distance $\|\overrightarrow{PM}\|$) des points à l'intérieur du domaine restreint n'est pas suffisante. En effet, pour le calcul de la base locale en chaque point, il s'agit de déterminer le gradient des level-sets ce qui nécessite la sélection des éléments contenant les points sélectionnés (les fonctions de forme des éléments étant utilisées pour calculer les gradients). Même si l'algorithme de sélection ne change pas nécessairement, le nombre d'opérations effectuées augmente et ceci peut rendre le temps de calcul de la sélection plus élevé que le calcul des level-sets par la méthode géométrique dans tout le domaine. A noter que, dans le même temps, le temps de calcul des gradients des level-sets est aussi plus élevé si la localisation du domaine n'est pas prise en compte.

Finalement, l'avantage de la localisation du domaine en terme de temps de calcul dépend fortement des algorithmes et implémentations choisis et du nombre de points du domaine de calcul. La localisation du domaine pourrait ainsi être avantageuse dans le cas de maillages éléments finis « très gros ». Dans ce cas, la limitation de la mise à jour des level-sets à un groupe de nœuds (fixé et défini par l'utilisateur) définissant la surface potentielle de propagation de la fissure peut être un moyen plus efficace de réduire le temps de calcul plutôt que le choix d'un algorithme de localisation du domaine pour lequel un nouveau domaine restreint doit être calculé à chaque pas de propagation afin de suivre le déplacement du front de fissure.

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

7 Méthode maillage pour la mise à jour des level-sets

La méthode la plus intuitive pour mettre à jour les valeurs des level-sets à chaque pas de propagation de la fissure consiste à utiliser leur propriété de distance signée. Ceci implique une représentation explicite de la fissure (lèvres de la fissure et fond de fissure) propagée afin de calculer les distances des nœuds à la fissure. Cette méthode utilise ainsi deux maillages distincts (fissure et structure) et seul le maillage de la fissure se trouve modifié au cours de la propagation. La mise à jour du maillage de la fissure à chaque pas de propagation consiste concrètement en la création d'une nouvelle rangée de mailles surfaciques pour représenter l'avancée de la fissure et d'une nouvelle rangée de mailles linéiques pour représenter le nouveau fond. Une fois ce maillage mis à jour, on recalcule les nouvelles level-sets par calcul direct de la distance à la fissure (projection orthogonale sur le maillage de la fissure).

7.1 Présentation de la méthode

Le maillage de la fissure est réalisé dans *Code_Aster* par la commande PROPA_FISS. La donnée d'entrée est la valeur du vecteur propagation sur le front de fissure, déterminée à partir des facteurs d'intensité des contraintes et de la loi de Paris. Le principe de la méthode maillage est de calculer les nœuds du nouveau fond à partir des vitesses de propagation et des nœuds du fond actuel. Il se résume aux trois étapes suivantes :

- Calcul par interpolation linéaire du vecteur propagation à chaque nœud du front de fissure discrétisé. Le vecteur propagation est orthogonal au front de fissure : la direction de propagation en un nœud correspond à la moyenne des vecteurs normaux des deux éléments linéaires se partageant ce nœud. Cette étape donne la position de chaque point propagé.
- Ajout d'un nœud en chacun de ces points, création d'une nouvelle surface (ajout d'éléments linéaires pour le nouveau front de fissure et d'éléments quadrangles pour la nouvelle surface) et processus de maillage de cette surface créée.
- Traitement spécifique des nœuds du bord qui, selon les cas, peuvent « sortir » de la structure. Ils sont alors déplacés sur les frontières de la structure.

Pour les modélisations 2D, la méthode est même plus simple. Seul un élément linéaire doit être ajouté à chaque pas de propagation et aucune correction des points du bord n'est nécessaire.

7.2 Algorithme

L'algorithme mis en place pour la propagation par la méthode maillage est le suivant (comme décrit dans (bib20) et sur la figure 7.2-1). A l'itération i :

• Récupération du maillage de la fissure,
• Récupération des points extrémités du front de fissure,
• Correction du maillage de la fissure :

- On modifie les coordonnées des deux nœuds extrémités du front de fissure pour qu'ils correspondent aux points extrémité du front de fissure
- On modifie les coordonnées de tous les autres nœuds du fond pour qu'ils soient équi-répartis entre les deux extrémités (nombre de nœuds du fond $NbNf$).
- Boucle sur les nœuds du fond : $1 \leqslant k \leqslant NbNf$
  - Calcul de $\Delta G(k)$ par interpolation linéaire,
  - Calcul de l'angle de bifurcation β à partir de $K_I(k)$ et $K_{II}(k)$,
  - Calcul du vecteur de propagation par interpolation linéaire :
  - $\Delta a(k) = C\left|\Delta G(k)\right|^{m/2}$ si C et m sont renseignés,
  - $\Delta a(k) = DA_{MAX}\left|\Delta G(k)/max_k(\Delta G(k))\right|^{m/2}$ si DA_{MAX} et m sont renseignés,
  - Calcul de la position du propagé du nœud N_k ,

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

- Ajout d'un nouveau nœud.

- fin de boucle
- Boucle entre 1 et $NbNf-1$
  - ajout d'une nouvelle maille SEG2 entre chacun des nœuds du fond et le nœud suivant,
  - ajout d'une maille QUA4 entre deux nœuds successifs du fond à l'itération $i-1$ et leurs propagés,
  - en 2D : ajout d'une maille POI pour le fond et SEG2 entre le fond précédent et le nouveau fond.
- Fin de boucle,
- Création des mailles $FOND$ (nouveau fond) et $LEVRE$ (lèvre précédente + nouvelles mailles surfaciques),
- Réactualisation du maillage.

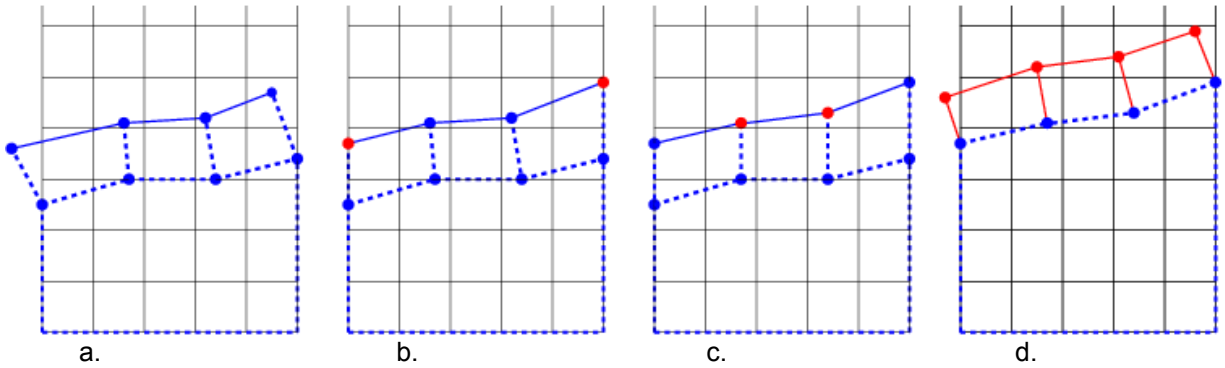


Figure 7.2-1: a. maillage en entrée de la routine, b. étape de correction des points du bord, c. étape d'équirépartition de tous les autres nœuds du front, d. propagation du front discrétisé et création des nouvelles mailles (SEG2 et QUA4).

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

8 Séparation et fusion des fronts de fissure

Différentes méthodes ont été présentées pour suivre la propagation d'une fissure avec X-FEM (ie pour mettre à jour les valeurs des level-sets à chaque pas de propagation) :

- La méthode maillage (ou par projection), voir paragraphe §7 ,
- La méthode géométrique, voir paragraphe§ 6 .
- Les méthodes upwind et simplexe basées sur des algorithmes de fast marching

Il s'agit d'expliquer ici les modifications à apporter à ces différentes méthodes pour rendre compte du comportement d'un front de fissure lorsqu'il rencontre un obstacle (un trou par exemple). Comment gérer la séparation de ce front de fissure en deux, voire en plusieurs autres morceaux ? De même qu'en est-il de la fusion de deux fronts de fissure indépendants en un seul ? Ce sont les level-sets qui déterminent le fond de fissure et donc la présence ou non de plusieurs fonds de fissure. Mais il s'agit dans certains cas de procéder à des améliorations des méthodes précédentes pour qu'elles s'adaptent à la séparation du front de fissure (modification de la création du maillage de la fissure pour la méthode maillage, continuité des level-sets dans le trou pour la méthode upwind avec utilisation d'une grille auxiliaire).

Selon les méthodes utilisées pour la gestion de la mise à jour des level-sets, la stratégie mise en place pour rendre compte de la séparation ou de la fusion de fronts de fissure diffère.

8.1 Méthode maillage

Pour la méthode maillage, une première option est d'étendre ce qui est fait au paragraphe 7.2 au cadre de plusieurs fonds de fissure. Les level-sets donnent des informations sur le nombre et la position des extrémités des fonds de fissure (intersection entre le maillage de la structure et l'iso-zéro des deux level-sets). Tout en conservant les nœuds qui constituent les extrémités déjà existantes à l'instant $i-1$, les nœuds constituant les nouvelles extrémités sont sélectionnés (figure 8.1-1 a.).

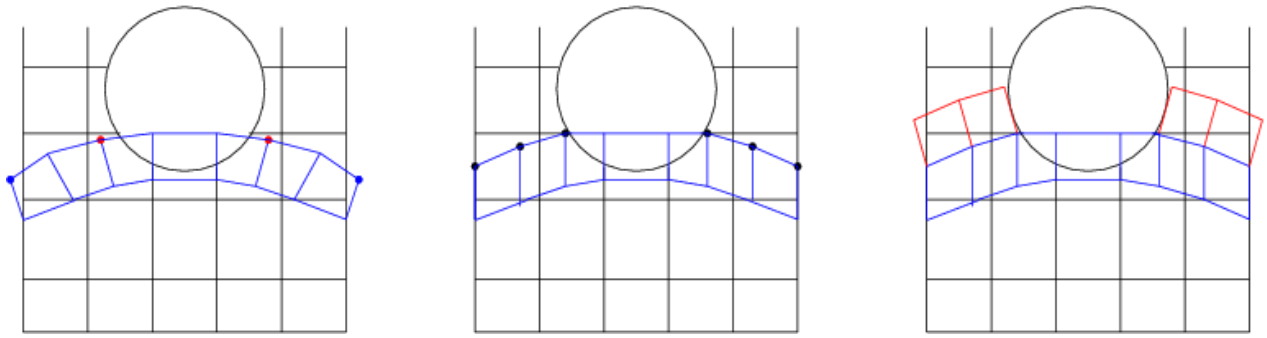


Figure 8.1-1: a. nœuds déterminant les extrémités des fonds de fissure (en bleu : nœuds sélectionnés pour les extrémités déjà existantes, en rouge : nœuds sélectionnés pour les nouvelles extrémités), b. repositionnement des nœuds sélectionnés sur les différents fonds, c. calcul des nœuds propagés et formation des nouvelles mailles.

Comme au paragraphe 7.2 une procédure de repositionnement des nœuds des différents fonds (figure 8.1-1 b.) est effectuée. Il ne reste plus qu'à calculer les nœuds propagés et former les mailles (figure 8.1-1 c.).

Bien que cette méthode fonctionne correctement, elle présente un désavantage : la perte de nœuds à la séparation de deux fonds. Les mailles peuvent, de plus, se déformer de façon importante lors de la phase de repositionnement des nœuds extérieurs à la structure. Ceci peut notamment conduire à une mauvaise discrétisation de la fissure en propagation hors-plan.

Pour pallier ces difficultés, l'idée a été d'augmenter le nombre de nœuds sur chaque fond de fissure. Reste la question de la création de ces nœuds et du maillage. En remarquant que les mailles représentant la fissure ne servent qu'à effectuer des projections, il est possible de créer une rangée de mailles pour l'avancée de fissures non conforme à la

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

rangée de mailles précédente (figure 8.1-2). De cette manière, on peut augmenter le nombre de nœuds entre deux pas de propagation sans modifier la discrétisation induite par les vitesses de propagation.

L'algorithme se résume donc comme suit : recherche des nœuds les plus proches des extrémités du fond, repositionnement sur le bord, création de nouveaux nœuds sur le fond actuel positionnés de façon équidistante les uns des autres, calcul des nœuds du nouveau fond. Comme il y a le même nombre de nœuds entre les deux fonds, des quadrangles respectant les surfaces du fond sont créés.

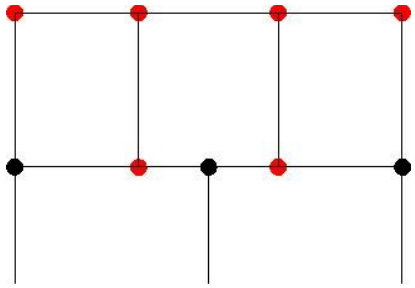


Figure 8.1-2: maillage non conforme (en rouge; nœuds créés pour la nouvelle rangée de mailles).

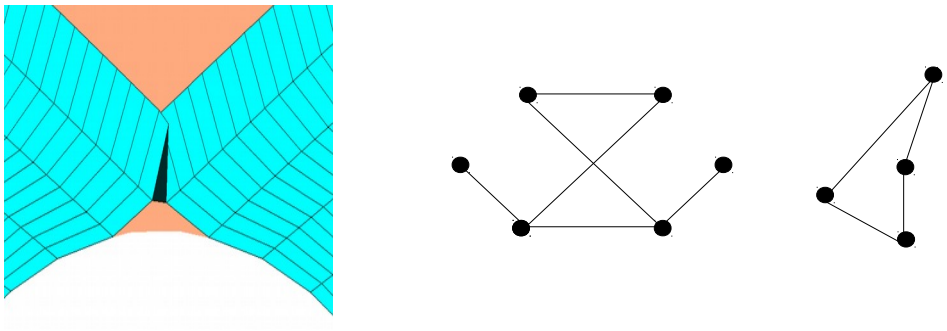
Le nombre total N de nœuds sur tous les fonds de fissure doit rester constant entre deux pas de propagation. Pour cela, on répartit les N_i nœuds par fond proportionnellement au rapport entre la longueur curviligne s_i du fond i et la longueur curviligne totale des n fonds :

$$N_i = E\left(\frac{s_i}{\sum_{j=1}^{n} s_j} N \right) \tag{66}$$

O ù $E(x)$ représente la partie entière de x . Comme il peut y avoir des cas pour lesquels la somme des N_i est inférieure à N , on distribue les nœuds restants aux fonds ayant la plus grande partie décimale.

Remarque :

La méthode maillage peut être problématique quand la fissure est concave. Ce cas peut être notamment rencontré lors de la réunion de deux fonds de fissure (figure 8.1-3 a.) et est absolument à éviter car des problèmes importants sont créés par le calcul des level-sets (valeurs fausses, apparition de faux fonds de fissure).



a. b.

Figure 8.1-3: cas problématiques : a.front de fissure concave lors de la réunion de deux fonds de fissures, b. création de mailles pathologiques.

La création de mailles problématiques (concaves par exemple) peut s'avérer être une source de difficultés (voir figure 8.1-3 b.). Elle peut apparaître lorsque la vitesse de propagation pour les nœuds du bord est imposée tangente à la surface. Cela peut également créer des erreurs dans le calcul de la level-set tangente et impliquer la création de faux

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

fonds de fissure. Pour éviter la concavité d'une maille dans le cas d'un avancement maximal Da_{max} de la fissure trop important, on diminue la valeur de ce dernier pour le calcul suivant. Soit le quadrangle représenté dans le plan (ABC) figure 8.1-4 (si le quadrangle n'est pas plan alors le point D représente la projection du nœud propagé à partir de C sur le plan (ABC)). On calcule les coordonnées du point M d'intersection entre (AB) et (CD) et on compare les distances entre AM et AB, CM et CD. On pose $\left(\vec{E}_1=\frac{\overrightarrow{AB}}{\|\overrightarrow{AB}\|},\ \vec{E}_2=\frac{\overrightarrow{AC}-(\overrightarrow{AC}.\vec{E}_1)\vec{E}_1}{\|\overrightarrow{AC}-(\overrightarrow{AC}.\vec{E}_1)\vec{E}_1\|}\right)$ une base orthonormée, et $\overrightarrow{AD}(d_1,d_2)$, $\overrightarrow{AC}(c_1,c_2)$ et $\overrightarrow{AM}(x,0)$ dans le repère $(A,\vec{E}_1,\vec{E}_1)$. Par définition de M, $det(\overrightarrow{CM},\overrightarrow{CD})=\begin{vmatrix} x-c_1 & d_1-c_1 \\ -c_2 & d_2-c_2 \end{vmatrix}=0$.

On en déduit $x=\frac{d_2c_1-d_1c_2}{d_2-c_2}=\frac{(\overrightarrow{AD}.\vec{E}_2)(\overrightarrow{AC}.\vec{E}_1)-(\overrightarrow{AD}.\vec{E}_1)(\overrightarrow{AC}.\vec{E}_2)}{\overrightarrow{CD}.\vec{E}_2}$.

On introduit le coefficient $Crit=min\left(\frac{AM}{AB},\frac{CM}{CD}\right)$ et l'avancement maximal rectifié Da_{max}' vérifie :

$si \quad Crit < 1.2 \qquad alors \quad Da_{max}'=\frac{Da_{max}.Crit}{1.2}$

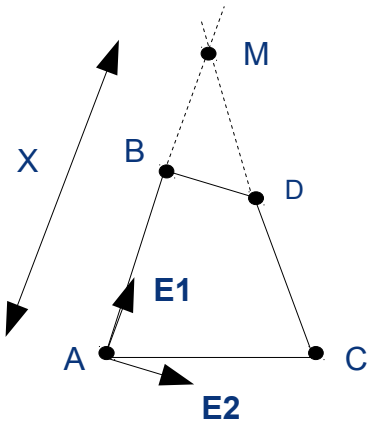


Figure 8.1-4: représentation d'une maille pour laquelle on vérifie que l'avancement n'est pas trop grand.

Le coefficient $1,2$ est un coefficient de sécurité afin d'assurer une propagation non nulle ($Crit=1$ conduisant à un point M confondu avec le(s) point(s) B et/ou D).

Bien qu'avec une telle stratégie, l'avancement maximal risque de tendre vers 0, elle permet cependant de résoudre une partie des cas problématiques.

8.2 Méthode upwind : introduction d'un fond virtuel

Comme remarqué dans le paragraphe 4.2.1.4 , l'utilisation de la méthode upwind s'accompagne très généralement de l'utilisation d'une grille auxiliaire (dont le maillage est régulier) sur laquelle s'effectue la mise à jour des level-sets qui sont par la suite projetées sur le maillage physique de la pièce (généralement beaucoup moins régulier). Cependant une partie de la grille peut être située à l'extérieur du maillage de la pièce, comme par exemple dans le cas d'une structure contenant un trou.

Les nœuds de la grille situés à l'extérieur du maillage de la pièce sont alors projetés sur le même point du front de fissure lors du calcul de la vitesse de propagation des level-sets. Tous ces nœuds se voient donc attribuer les mêmes vitesses et angles de propagation ainsi que les mêmes bases locales (comme décrit au paragraphe 2.3). Il en résulte l'apparition d'une discontinuité des level-sets sur la grille au niveau du bord libre de la pièce (voir figure 8.2-1).

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

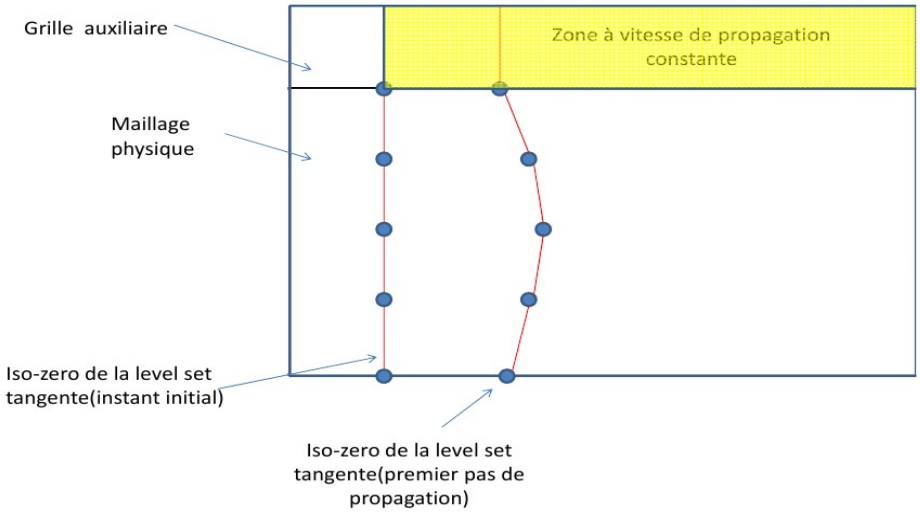


Figure 8.2-1 : exemple de discontinuité des level-sets sur la grille au niveau du bord libre.

Cette discontinuité, bien que située hors du maillage physique, implique une mauvaise définition du gradient des level-sets sur le bord libre et donc une mauvaise définition de la base locale lors du pas de propagation suivant. Elle peut également influer sur la définition des level-sets à l'intérieur du maillage, notamment lors de la propagation de fissures dans des pièces ayant une section irrégulière et/ou des trous. Pour éviter ces problèmes et définir de manière cohérente les level-sets sur la grille au voisinage du maillage physique, une interpolation du front de fissure à l'extérieur du maillage par une droite tangente au front de fissure est proposée. Pour cela, le dernier segment du front de fissure est prolongé par un segment de droite à l'extrémité duquel est introduit un nouveau nœud du front de fissure, qualifié de virtuel. Ce point est alors considéré comme un point du front de fissure. La même base locale que celle du dernier point du front de fissure physique lui est attribuée. La vitesse et l'angle de propagation sont calculés par interpolation linéaire à partir des vitesses et angles de propagation des deux derniers points (figure 8.2-2).

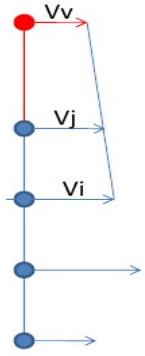


Figure 8.2-2 : représentation de l'approximation linéaire de la vitesse au point virtuel V_v (en rouge) à partir des vitesses des deux derniers points du front de fissure V_i et V_j (en bleu).

Pour créer les points du fond virtuel, on procède de la manière suivante (figure 8.2-3) :

- on pose P_i et P_j deux points situés à l'extrémité du font de fissure et d la distance du point virtuel P_v au dernier point du fond de fissure P_j ,
- on détermine la droite prolongeant le dernier segment du front de fissure au point virtuel : $\overrightarrow{P_{jv}} = d.\vec{b}_j$ où $\overrightarrow{P_{jv}}$ est le vecteur reliant P_j à P_v et $\vec{b}_j$ est le vecteur unitaire tangent au front de fissure au point P_j ,

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

- on calcule les dérivées spatiales de la vitesse et de l'angle de propagation sur le dernier segment du fond virtuel :

$$\frac{dV}{ds}=\frac{V_j-V_i}{\left\|\overrightarrow{P_{ij}}\right\|},\qquad \frac{d\beta}{ds}=\frac{\beta_j-\beta_i}{\left\|\overrightarrow{P_{ij}}\right\|} \tag{67}$$

- on en déduit la vitesse et l'angle de propagation au fond virtuel :

$$V_v=V_j+d.\frac{dV}{ds},\qquad \beta_v=\beta_j+d.\frac{d\beta}{ds} \tag{68}$$

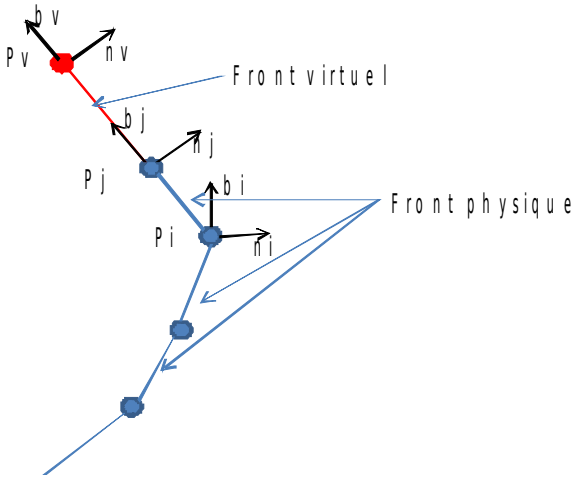


Figure 8.2-3 :création et emplacement du front de fissure virtuel.

On place le point virtuel à une distance d du front physique égale au rayon du tore utilisé pour la propagation des level-sets (voir paragraphe 5). On choisit cette distance de façon à obtenir des level-sets lissées sur l'ensemble de la zone de propagation. Si la localisation n'est pas utilisée, le point virtuel est placé à une distance de sept fois l'avancement maximal de la fissure. Pour des vitesses et des angles de propagation (définis sur les points virtuels) trop importants ou trop faibles, on rapproche le point virtuel du front physique de manière à satisfaire les conditions imposées. On note par exemple que l'on a une importante discontinuité des level-sets si l'angle de propagation est supérieur (respectivement inférieur) à 90 degrés (respectivement -90 degrés). On note également que si la distance du point virtuel au front de fissure physique est inférieure au rayon de localisation, on peut conserver une discontinuité des gradients. Néanmoins, cette discontinuité est située loin du maillage de la pièce. Elle n'influe donc pas sur la définition de la base locale sur le bord libre.

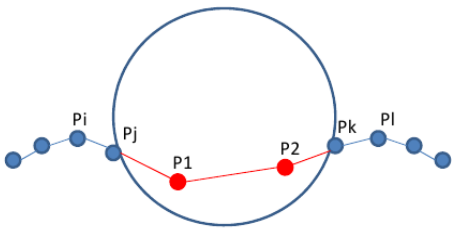


Figure 8.2-4: création d'un front virtuel (en rouge) à l'intérieur d'un trou circulaire. P_1 est créé par prolongement de $[P_iP_j]$ et P_2 par prolongement de $[P_lP_k]$. Le dernier segment du fond virtuel relie P_1 et P_2 et connecte les fronts de fissure.

La méthode du fond virtuel peut alors être immédiatement étendue au cas de trous. Si la pièce présente un trou, il convient en effet également de bien définir les level-sets à l'intérieur de celui-ci. On procède alors comme décrit précédemment en définissant un front de fissure virtuel à l'intérieur du trou par la création de deux points virtuels P_1 et P_2 , un pour chaque fond de fissure délimité par le trou (figure 8.2-4). Deux des trois segments du front virtuel

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

sont des extensions des deux fronts de fissures physiques. Le troisième segment relie quant à lui les deux points virtuels afin de connecter les fronts de fissure. Il ne sert pas à interpoler les level-sets mais à assurer leur continuité dans le trou. Les deux points virtuels ayant des vitesses de propagation différentes, deux nœuds voisins situés au centre du trou peuvent être projetés sur deux nœuds du fond de fissure différents et avoir des vitesses différentes.

Les angles et vitesses de propagation ainsi que la base locale sur le segment $[P_1 P_2]$ sont calculés par interpolation linéaire (comme décrit au paragraphe 2.4) à partir des valeurs attribuées aux deux points virtuels.

Dans le cas de la création d'un fond virtuel dans un trou, on doit s'assurer qu'il n'y a pas de croisement des fonds de fissure. La distance d'un point virtuel au fond de fissure ne doit pas dépasser le tiers de la distance entre les deux fonds. Soit :

$$d \leq \frac{\|P_k - P_j\|}{3} \tag{69}$$

De plus, la vitesse de propagation au point virtuel V_v ne doit pas être trop petite ou trop importante. On impose qu'elle soit au moins égale aux deux tiers de la vitesse Vj du dernier fond de fissure et qu'elle ne doive pas dépasser quatre tiers de cette valeur. Ainsi, on impose que :

$$\frac{2V_j}{3} \leq V_v \leq \frac{4V_j}{3} \tag{70}$$

De même :

$$\text{Si } V_v < \frac{2V_j}{3} \text{ alors on impose } V_v = \frac{2V_j}{3} \tag{71}$$

Et :

$$\text{Si } V_v > \frac{4V_j}{3} \text{ , alors on impose } V_v = \frac{4V_j}{3} \tag{72}$$

On en déduit l'expression de la distance d dans ces deux cas :

$$d = \left| \frac{V_j}{3\frac{dV}{ds}} \right| \tag{73}$$

Néanmoins, la création d'un fond virtuel n'est pas suffisante. Les valeurs des level-sets définies à l'intérieur du trou, issues de la projection sur le segment milieu du fond virtuel, peuvent dans certains cas influencer la définition des level-sets dans le maillage. De plus, l'approximation de la vitesse est plus précise si les level-sets sont initialement droites et si l'intersection des deux iso-zeros est confondue avec le front virtuel. Dans le but de garantir le redressement des level-sets dans le trou lors de la fusion des fronts de fissure après le passage de celui-ci, on recalcule les level-sets pour les nœuds projetés sur un segment virtuel du front de fissure, et ce avant la propagation des level-sets. On calcule les level-sets de manière géométrique (voir paragraphe 6) en projetant le nœud sur le front de fissure (figure 8.2-5). On pose XM un nœud du maillage. On définit alors son projeté XN sur le fond de fissure, les vecteurs $\vec{tp}$ et $\vec{np}$ définissant la base locale en front de fissure en XN . On obtient les level-sets normale et tangente aux nœuds XM simplement par : $lsn(XM) = \overrightarrow{XM\,XN} . \vec{np}$ et $lst(XM) = \overrightarrow{XM\,XN} . \vec{tp}$.

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

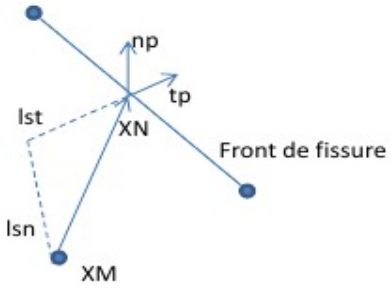


Figure 8.2-5: Calcul des level-sets aux nœuds projetés sur le fond de fissure.

Ce calcul des level-sets permet de corriger l'influence de la définition non physique des level-sets au centre du trou lors de la fusion des deux fonds de fissure. Une fois le fond virtuel créé et les level-sets corrigées, le calcul des vitesses de propagation par projection sur le front de fissure s'effectue comme décrit au paragraphe 2.3 , sans faire de distinction entre le front physique et le fond virtuel.

Remarque :

- L'introduction d'un fond virtuel nécessite une stratégie de vérification de la numérotation des fronts de fissure ainsi que de leur sens de parcours ce qui n'était jusqu'alors pas nécessaire : la connexion des fronts de fissure par un fond virtuel ne les rend plus indépendants les uns des autres. La méthode suivante a été adoptée pour vérifier si les nœuds des différents fronts de fissure sont bien numérotés (figure 8.2-6):
  - on construit un nouveau front de fissure sur la grille auxiliaire qui par définition de la grille n'est constitué que d'un seul morceau avec un unique sens de parcours. On projette ensuite les extrémités E_1^i et E_2^i de chacun des morceaux i du front de fissure réelle sur le front de fissure continu associé à la grille. Ces extrémités sont repérées sur le front de fissure de la grille par leur abscisse curviligne respective. A titre d'exemple, aux extrémités E_1^i et E_2^i correspondent les abscisses curvilignes $s_{P_1^i}$ et $s_{P_2^i}$ (où P_1^i et P_2^i sont respectivement les projetés de E_1^i et E_2^i sur le front de fissure associé à la grille auxiliaire) ;
  - la séquence des abscisses curvilignes ainsi obtenue permet une renumérotation des morceaux des fronts de la fissure réelle ;
  - on évalue le produit scalaire $\left(E_2^i E_1^i\right).\left(P_2^i P_1^i\right)$ pour vérifier que chacun des morceaux est parcouru dans le même sens que le front de fissure sur la grille. Dans le cas contraire, on réoriente le(s) morceau(x) mal orienté(s).

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

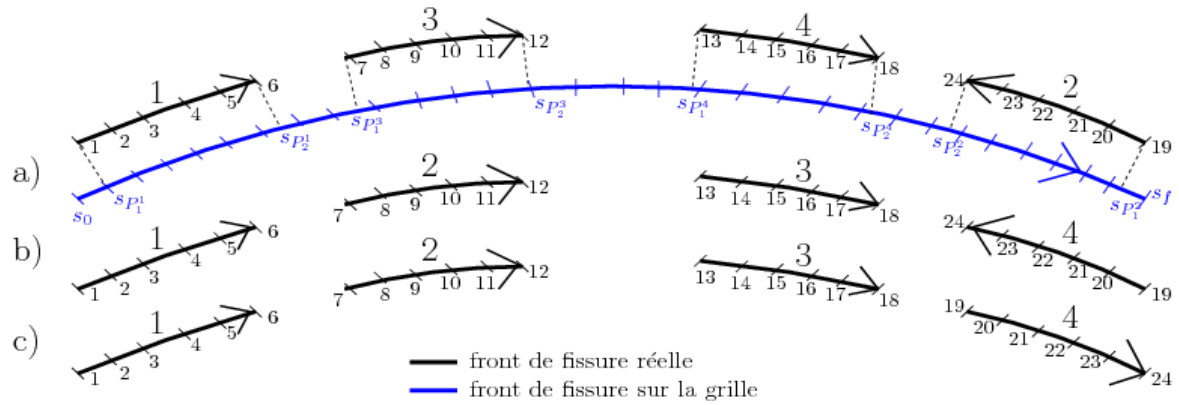


Figure 8.2-6: a. exemple de mauvaise numérotation et orientation du front de fissure réelle (noire) et front de fissure sur la grille (bleu) , b. renumérotation des morceaux du front de fissure réelle, c. réorientation des morceaux du front de fissure réelle

8.3 Méthode géométrique et simplexe

La méthode géométrique et la méthode simplexe ne nécessitant aucune grille auxiliaire, la gestion de la séparation et de la fusion de fronts de fissure est automatique et ne demande aucune modification particulière.

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

9 Bibliographie

1 GRAVOUIL A., MOËS N., BELYTSCHKO T., “Non-planar 3D crack growth by the extended finite element and level-sets - Part II: Level set update”, International Journal for Numerical Methods in Engineering, vol. 53, pp. 2569-2586, 2002

2 OSHER S., SETHIAN J. A., “Fronts propagations with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations“, Journal of Computational Physics, vol. 79, pp. 12-49, 1988

3 SUKUMAR N., CHOPP D.L., MORAN B., “Extended finite element method and fast marching method for three-dimensional fatigue crack propagation”, Engineering Fracture Mechanics, vol. 70, pp. 29-48, 2003

4 ERDOGAN G., SIH G.C ., “ On the crack extension in plates under plane loading and transverse shear”, Journal of Basic Engineering, vol. 85, pp. 519-27, 1963

5 BARTH T.J., SETHIAN J.A., “ Numerical schemes for the Hamilton-Jacobi and level-set equations on triangulated domains”, Journal of Computational Physics, vol. 145, pp. 1-40, 1998

6 CRANDALL M.G., LIONS P.L., “ Two approximations of solutions of Hamilton-Jacobi equations”, Mathematics of Computation, vol. 43, pp. 1-19, 1984

7 DECONINCK H., STRUIJS R., ROE P.L., “ Compact Advection Schemes on Unstructured Grids”, Technical Report, VKI, VKI LS 1993-04, Computational Fluid Dynamics, 1993

8 ROE P.L., “ Linear Advection Schemes on Triangular Meshes”, Technical Report CoA 8720, Cranfield Institute of Technology, 1987

9 ROE P.L., “Optimum Upwind Advection on Triangular Mesh”, ICASE 90-75, 1990

10 PENG D., MERRIMAN B., OSHER S., ZHAO H., KANG M., “A PDE-based fast local level-set method”, Journal of Computational Physics, vol. 155, pp. 410-438, 1999

11 ADALSTEINSSON D., SETHIAN J.A., “A fast level-set method for propagating interfaces”, Journal of Computational Physics, vol. 118, pp. 269-277, 1995

12 PRABEL B., COMBESCURE A., GRAVOUIL A., MARIE S., “Level set X-FEM non matching meshes: application to dynamic crack propagation in elastic-plastic media”, International Journal for Numerical Methods in Engineering, vol. 1, pp. 1-15, 2006

13 DUFLOT M., “A study of the representation of cracks with level-sets”, International Journal for numerical methods in engineering, vol. 70, pp. 1261-1302, 2007

14 COLOMBO, PATRICK MASSIN, « Fast and robust level-set update for 3D non-planar X-FEM crack propagation modelling », Computer Methods in Applied Mechanics and Engineering, 200 (2011), 2160-2180

15 SUKUMAR N., CHOPP D.L., BÉCHET E., MOËS N., « Three-dimensional non-planar crack growth by a coupled extended finite element and fast marching method », International Journal Numerical Methods in Engineering, 76 (2008), 727-748

16 SHI J., CHOPP D., LUA J., SUKUMAR N., BELYTSCHKO T., « Abaqus implementation of extended finite element method using a level-set representation of three-dimensional fatigue crack growth and life predictions », Engineering Fracture Mechanics, 77 (2010), 2840-2863

17 CITARELLA R., BUCHHOLZ F.-G., Comparison of crack growth simulation by DBEM and FEM for SEN-specimens undergoing torsion or bending loading, Engineering Fracture Mechanics, 75 (2008), 489-509

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

18 COLOMBO D., « An implicit geometrical approach to level-sets update for 3D non planar X-FEM crack propagation », submitted to Computer Methods in Applied Mechanics and Engineering

19 SETHIAN J.A., Level Set Methods and Fast Marching Methods, 1999.

20 GALENNE E., « Propagation automatique de fissures 3D avec X-FEM en fatigue : méthode de projection » Compte rendu AMA, 2008

21 HABOUSSA D., « Modélisation de la transition traction-cisaillement des métaux sous choc par la X-FEM », Thèse, INSA Lyon, 2012

22 BRONSTEIN A., BRONSTEIN M., KIMMEL R . « Numerical geometry of non-rigid shapes », Springer, 2007

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

10 Description des versions

Indice document	Version Aster	Auteur(s) Organisme(s)	Description des modifications
A	11	D. COLOMBO (University of Manchester), M. GUITON, M. SIAVELIS (IFPen), S. GENIAUT, V.X. TRAN EDF/R&D AMA, P.MASSIN EDF/R&D LaMSID	Texte initial

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