HAL
open science

Algorithmes de résolution rapide de problèmes mécaniques sur GPU

Marion Ballage

To cite this version:

Marion Ballage. Algorithmes de résolution rapide de problèmes mécaniques sur GPU. Algorithme et structure de données [cs.DS]. Université Paul Sabatier - Toulouse III, 2017. Français. NNT : 2017TOU30122 . tel-01888716

HAL Id: tel-01888716

https://theses.hal.science/tel-01888716

Submitted on 5 Oct 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *Date de soutenance* **par :**

Marion BALLAGE

Algorithmes de résolution rapide de problèmes mécaniques sur GPU

JURY

Didier AUROUX	Professeur des Universités	Rapporteur
David DUREISSEIX	Professeur des Universités	Rapporteur
Jean-Daniel BELEY	Directeur développement logiciel	Co-encadrant de thèse
Jérôme FEHRENBACH	Maître de conférence	Directeur de thèse
Mohamed MASMOUDI	Professeur des Universités	Co-directeur de thèse
Patrick LABORDE	Professeur des Universités	Examinateur
Vanessa LLERAS	Maître de conférence	Examinatrice

École doctorale et spécialité :
MITT : Domaine Mathématiques : Mathématiques appliquées
Unité de Recherche :
Institut de Mathématiques de Toulouse
Directeur(s) de Thèse :
Jérôme FEHRENBACH et *Mohamed MASMOUDI*

Table des matières

Chapitre 1

Introduction

Cette thèse s'inscrit dans un contexte d'analyse numérique en calcul de structures. L'objectif industriel est un calcul en temps réel des solutions. En effet, la généralisation de l'utilisation du calcul et l'augmentation de la complexité des modèles traités entraînent des besoins de calculs plus rapides. Les processeurs graphiques offrent de nouvelles opportunités pour le calcul hautes performances. Au cours de cette thèse, les processeurs graphiques (GPU) ont été utilisés afin d'implémenter une méthode éléments finis sans maillage.

Lors d'analyses numériques en calcul de structures, la génération de maillages conformes sur des modèles à géométrie complexe conduit à des tailles de modèles importantes, et amène à imaginer de nouvelles approches éléments finis. Le temps de génération d'un maillage conforme est directement lié à la complexité de la géométrie, augmentant ainsi considérablement le temps de calcul global. Implémenter une méthode sans maillage permettra de réduire de façon importante le temps global.

Au départ, les processeurs graphiques ont été développés pour les besoins de l'affichage graphique. Depuis 2003, l'architecture des GPU a été adaptée à la parallélisation de divers calculs numériques. Ces processeurs d'une architecture massivement parallèle ouvrent de nouvelles opportunités pour le calcul scientifique. Le calcul sur processeurs graphiques est une technique récente qui montre de très bons résultats au niveau de la rapidité des calculs.

Certaines méthodes sont naturellement compatibles avec les processeurs graphiques, par exemple les méthodes particulaires, du fait de leur facilité à être parallélisées. Ces méthodes sont utilisées dans les domaines de la mécanique des fluides (PowerFLOW [Exa]) ou de l'électromagnétisme. En optique, l'algorithme du lancer de rayon utilisé pour le rendu d'images, les effets d'illumination a déjà été optimisé pour les processeurs graphiques [Hu+14]. Des développements analogues dans les domaines de la dynamique rapide en mécanique des structures peuvent être imaginés.

Plusieurs différences entre les processeurs centraux et graphiques existent, notamment les processeurs graphiques ont très peu de mémoire rapide compatible avec leurs performances comparativement aux processeurs centraux. Ils sont aussi pensés massivement pa-

rallèles. Pour arriver à de bons résultats lors de la programmation sur GPU, les algorithmes ont besoin d'être repensés, afin de correspondre à l'architecture des processeurs graphiques et de minimiser la mémoire utilisée.

Nous présentons une nouvelle méthode éléments finis sans maillage du domaine. La géométrie est représentée par une fonction surfaces de niveau. Une méthode d'intégration adaptée à cette représentation géométrique est proposée. Cette méthode peut s'avérer coûteuse en temps de calcul, c'est pour cette raison que nous proposons une technique d'apprentissage donnant la matrice élémentaire de rigidité en fonction des valeurs de la fonction surfaces de niveau aux sommets de l'élément considéré.

Le premier chapitre présentera deux méthodes éléments finis. La méthode des éléments finis standard a été conçue pour tourner sur processeurs centraux (CPU) depuis cinquante ans. Cette méthode est utilisée dans différents domaines, en mécanique des fluides, des structures, en thermique. Elle permet de trouver numériquement une solution approchée des équations aux dérivées partielles sous des conditions bien posées. C'est un outil bien connu et maîtrisé dans le milieu industriel. Néanmoins, la mise en œuvre de cette méthode doit être adaptée, car les processeurs graphiques ne traitent pas efficacement les maillages éléments finis non structurés.

La méthode des éléments finis étendus (XFEM) est une méthode plus récente de mécanique. Cette méthode éléments finis propose des améliorations de temps de maillage de la méthode classique dans différents contextes. L'utilisation la plus connue de cette méthode se fait en mécanique de la rupture pour les fissures et le non raffinement lors de leur propagation. L'avantage de cette méthode est qu'elle apporte déjà des solutions pour des simulations éléments finis sur un maillage non conforme, donc mieux pris en charge par les processeurs graphiques.

Cette méthode provient de travaux de Babuška *et al* [BCO94], [BM97], [MB96] qui couplent le principe de partition de l'unité avec la méthode des éléments finis pour permettre une approximation locale tout en assurant une condition de continuité globale. Par la suite, Moës *et al* [MDB99] et Dolbow *et al* [DMB00] utilisent ces travaux pour enrichir par des fonctions discontinues qui permettent une meilleure représentation du saut en déplacements de part et d'autre de la fissure. Strouboulis *et al* étendent cette méthode pour des maillages non conformes au modèle [SCB01].

Le chapitre 3 expose la méthode éléments finis développée au cours de cette thèse. Cette méthode introduit une nouvelle approche en calcul de structures sur processeurs graphiques et sans maillage. La nouvelle méthode définit la géométrie de manière implicite par une fonction surface de niveaux, permettant de représenter des géométries complexes. Le bord de la géométrie n'est pas facettisé conformément à l'usage dans le monde du calcul XFEM. Le bord est représenté par une fonction homographique. Dans ce contexte, la géométrie est trop complexe pour que les formules usuelles d'intégration donnent une bonne approximation. L'intégration de la matrice élémentaire nécessite un grand nombre de points d'intégration, ce qui est incompatible avec nos objectifs d'efficacité. Une méthode d'apprentissage est introduite afin d'améliorer les temps de calcul des matrices de rigidité.

Le chapitre 4 présente la méthode utilisée pour apprendre les matrices élémentaires.

La gestion des résultats est une autre étape clé dans le milieu industriel. En effet, les modèles d'une complexité grandissante et les schémas de résolution de plus en plus précis augmentent considérablement la taille des fichiers résultats. La taille croissante de ces fichiers posent des problèmes aussi bien de stockage que de performances lors de l'exploitation. Les tailles de fichiers posent aussi problème dans le cas de calculs directement effectués sur le cloud. La puissance de calcul de cet outil le rend très attractif. Rapatrier les résultats obtenus par calcul sur le cloud reste néanmoins difficile et très long. Un post-traitement de ces données a alors un intérêt majeur, une technique de compression est mise en place.

La dernière partie est consacrée à la compression par ondelettes de modèles et résultats d'analyse éléments finis. Les ondelettes sont des outils bien connus en traitement du signal et de l'image. La transformée en ondelettes est une technique qui permet une décomposition et une reconstruction sans perte.

La compression est appliquée à des données $3D$ stationnaires ou transitoires. Pour la compression $3D$, une transformée en ondelettes discrète $3D$ est appliquée sur les données. La compression $3D + t$ traite la dimension temporelle de façon différente.

Chapitre 2

Motivations et état de l'art

Dans ce chapitre nous introduisons le contexte général du problème de mécanique. Deux méthodes existantes permettant de résoudre ce problème sont présentées, la méthode des éléments finis dans la section 2.2 et la méthode des éléments finis étendus dans la section 2.3

2.1 Problème posé et notations

Nous nous intéressons à une équation aux dérivées partielles elliptique posée dans un domaine Ω, que l'on peut écrire sous la forme :

$$\mathcal{L}(u) = f \quad \text{dans } \Omega, \tag{2.1}$$

avec $\mathcal{L}$ un opérateur elliptique, u l'inconnue, f les sollicitations et Ω le domaine de calcul.

La formulation variationnelle de l'équation peut s'écrire d'une manière générale

$$\begin{cases} u \in \mathcal{V}_D, \\ a(u, \phi) = L(\phi), \quad \forall \phi \in \mathcal{V}, \end{cases} \tag{2.2}$$

$\mathcal{V}_D$ étant l'espace affine qui prend en compte les conditions de Dirichlet imposées et $\mathcal{V}$ l'espace vectoriel correspondant à des conditions de Dirichlet homogènes, a est la forme bilinéaire coercive associée à notre équation elliptique, et L est la forme linéaire provenant des termes sources et des conditions de bord de Neumann.

Au cours de cette thèse, nous avons développé de nouvelles méthodes de résolution numérique de l'équation elliptique (2.1). Afin de fixer les idées, tout notre travail s'est effectué dans le cadre de l'élasticité linéaire en dimension deux avec des conditions de Dirichlet homogènes. Cependant les concepts et méthodes ont un cadre d'application plus large.

Plus précisément, nous souhaitons résoudre l'équation d'équilibre suivante, sous l'hypothèse des petits déplacements, et des petites perturbations en régime quasi-statique et d'un matériau linéaire, homogène, isotrope :

$$\begin{cases} -div(\boldsymbol{\sigma}) & = & f_V & \text{sur } \Omega, \\ u & = & 0 & \text{sur } \Gamma_0, \\ \boldsymbol{\sigma}(u).n & = & f_S & \text{sur } \Gamma_1, \end{cases} \tag{2.3}$$

avec div la divergence, f_V les forces volumiques, f_S les forces surfaciques, $\boldsymbol{\sigma}$ le tenseur des contraintes et Γ_0 et Γ_1 forment une partition de la frontière $\partial\Omega$. La figure 2.1 illustre les frontières Γ_0, Γ_1 du domaine Ω :

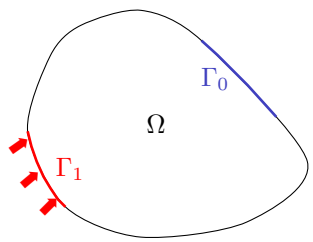


FIGURE 2.1 – Le domaine avec les frontières où sont appliquées des conditions limites

Le tenseur $\boldsymbol{\sigma}$ est donné par la loi de comportement, dans le cas de l'élasticité linéaire il vaut :

$$\boldsymbol{\sigma} = \mathbf{D}\varepsilon(u), \tag{2.4}$$

avec $\mathbf{D}$ l'opérateur linéaire des rigidités élastiques, $\boldsymbol{\varepsilon}$ le tenseur des déformations et u est un vecteur en deux dimensions qui représente les déplacements selon les deux directions d'espace, $u = \begin{pmatrix} u_x \\ u_y \end{pmatrix}$. L'opérateur $\mathbf{D}$ est défini dans l'équation (2.5) :

$$\mathbf{D}\boldsymbol{\varepsilon} = \lambda\,(tr\boldsymbol{\varepsilon})\,Id + 2\mu\boldsymbol{\varepsilon} \tag{2.5}$$

où λ et μ sont les coefficients de Lamé du matériau. Le tenseur $\boldsymbol{\varepsilon}$ est donné par la définition (2.10) :

$$\boldsymbol{\varepsilon}(u) = \frac{\nabla u + (\nabla u)^T}{2} = \frac{1}{2}\begin{pmatrix} 2\dfrac{\partial u_x}{\partial x} & \dfrac{\partial u_x}{\partial y} + \dfrac{\partial u_y}{\partial x} \\ \dfrac{\partial u_y}{\partial x} + \dfrac{\partial u_x}{\partial y} & 2\dfrac{\partial u_y}{\partial y} \end{pmatrix}. \tag{2.6}$$

L'équation (2.3) est transformée en une équation équivalente intégrale :

$$\int_\Omega -div(\boldsymbol{\sigma})v\ d\Omega = \int_\Omega f_V v\ d\Omega \quad \forall v \in \mathcal{V}, \tag{2.7}$$

avec $\mathcal{V}$ l'ensemble des solutions admissibles régulières.

En appliquant une intégration par parties, l'équation (2.7) devient :

$$\int_\Omega \boldsymbol{\sigma}(u) : \boldsymbol{\varepsilon}(v)\ d\Omega = \int_\Omega f_V v\ d\Omega + \int_{\partial\Omega} \boldsymbol{\sigma}.n\ v\ d\Gamma \quad \forall v \in \mathcal{V}. \tag{2.8}$$

L'équation (2.8) est la formulation variationnelle du problème initial. D'après le théorème de Lax-Milgram, il existe une solution à l'équation (2.8) qui est l'unique solution du problème de minimisation de la fonctionnelle suivante :

$$\mathcal{J}(u) = \frac{1}{2}\int_\Omega \boldsymbol{\sigma}(u) : \boldsymbol{\varepsilon}(u)\ d\Omega - \int_\Omega f_V u\ d\Omega - \int_{\partial\Omega} \boldsymbol{\sigma}.n\ u\ d\Gamma \quad \forall u \in \mathcal{V}. \tag{2.9}$$

2.1.1 Notations

Comme les tenseurs considérés sont symétriques, nous décidons d'adopter la notation de Voigt qui remplace l'écriture d'un tenseur de rang deux $\begin{pmatrix} a & c \\ c & b \end{pmatrix}$ par un vecteur $\begin{pmatrix} a \\ b \\ 2c \end{pmatrix}$.

Par exemple le tenseur des déformations en notation de Voigt s'écrit :

$$\varepsilon(u) = \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 2\varepsilon_{xy} \end{pmatrix} = \begin{pmatrix} \dfrac{\partial u_x}{\partial x} \\ \\ \dfrac{\partial u_y}{\partial y} \\ \\ \dfrac{\partial u_x}{\partial y} + \dfrac{\partial u_y}{\partial x} \end{pmatrix}. \tag{2.10}$$

Cette notation permet de noter le tenseur de Hooke D sous forme matricielle :

$$\begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ 2\sigma_{xy} \end{pmatrix} = \underbrace{\begin{pmatrix} \lambda + 2\mu & \lambda & 0 \\ \lambda & \lambda + 2\mu & 0 \\ 0 & 0 & 2\mu \end{pmatrix}}_{D} \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix}. \tag{2.11}$$

2.2 Méthode des éléments finis

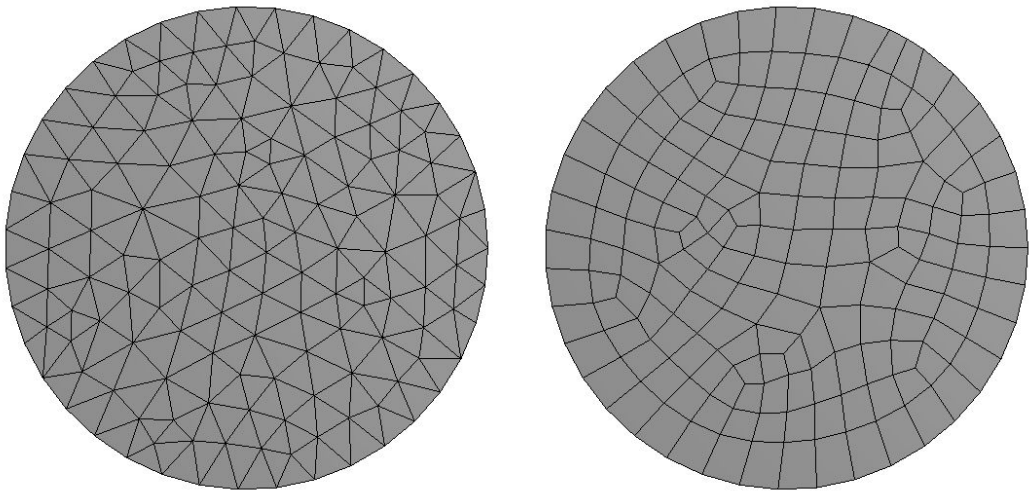
Cette section présente la méthode des éléments finis, qui est la méthode utilisée actuellement dans la plupart des codes industriels pour le calcul de structures. Cette méthode permet une approximation de solutions de tous les problèmes d'équations aux dérivées partielles, et donc dans notre cas de problèmes de mécanique des milieux continus. C'est une méthode de discrétisation de problèmes continus, nous nous intéresserons aux problèmes dits d'équilibre ou de valeurs limites. Une présentation est faite pour le cas de la dimension deux, la généralisation en trois dimensions est bien sûr connue.

2.2.1 Discrétisation du problème

Afin de rendre possible la résolution numérique, il est nécessaire de se placer en dimension finie. Une façon classique de le faire est d'utiliser la méthode de Galerkine. Soit $\mathcal{V}_h \subset \mathcal{V}$ sous-espace vectoriel de dimension finie, nous nous intéressons à la minimisation de l'énergie donnée par l'équation (2.9) sur le sous-espace vectoriel $\mathcal{V}_h$. La formulation variationnelle du problème s'écrit :

$$\begin{cases} u_h \in \mathcal{V}_h, \\ a(u_h, \phi) = L(\phi), \quad \forall \phi \in \mathcal{V}_h. \end{cases} \tag{2.12}$$

Pour la méthode des éléments finis, le domaine Ω est discrétisé par un maillage. Différents types de maillage existent, par exemple les maillages tétraédriques et hexaédriques en trois dimensions, les maillages triangulaires et quadrilatéraux en deux dimensions. La figure 2.2 montre deux exemples de maillage en dimension deux d'un disque. Il est primordial qu'il n'y ait ni recouvrement, ni trou entre deux éléments distincts.



(a) Éléments triangulaires (b) Éléments quadrilatéraux

FIGURE 2.2 – Exemples de maillage

Dans chaque élément Ω_e, comme l'espace d'approximation V_h est de dimension finie, la solution peut être décomposée en une base finie de fonctions $\{N_i\}_{\{i \in 1,\dots,l\}}$ de $\mathcal{V}$. Notre problème revient à trouver les composantes de u_h dans cette base. La fonction u_h peut être écrite en tout point x en faisant une somme finie des fonctions de base.

$$u_h(x) = \sum_{i \in I} u_i N_i(x) \quad \forall x \in \Omega_e, \tag{2.13}$$

où $u_i = u_h(x_i)$, et I représente l'ensemble des nœuds associés aux fonctions N_i. Les x_i sont nos points d'approximation.

Les fonctions N_i, appelées par la suite fonctions de forme valent un en un nombre k de points d'approximation x_i et s'annulent pour les autres.

L'équation variationnelle (2.12) est vraie pour toute fonction $\phi \in \mathcal{V}_h$, elle est en particulier vraie pour les fonctions de forme N_i et en décomposant u_h en fonction de cette base (2.13) l'équation devient :

$$\forall j \in I, \sum_{i \in I} u_i a(N_i, N_j) = L(N_j), \tag{2.14}$$

En remplaçant par les formulations des fonctions $a(u, v)$ et $\mathcal{L}(v)$, elle devient :

$$\sum_{i \in I} u_i \int_{\Omega_e} \sigma(N_i) : \varepsilon(N_i) \; d\Omega_e = \sum_{i \in I} \left(\int_{\Omega_e} f_V N_i \; d\Omega_e + \int_{\partial\Omega_e \cap \partial\Omega} f_S N_i \; d\Gamma \right), \tag{2.15}$$

Pour se ramener à une équation matricielle, posons $U = (u_x, u_y)$ et N la matrice ayant pour composantes les fonctions de base N_i :

$$N = \begin{pmatrix} N_1 & 0 & N_2 & 0 & \dots & N_l & 0 \\ 0 & N_1 & 0 & N_2 & \dots & 0 & N_l \end{pmatrix}. \tag{2.16}$$

De la même manière, soit B la matrice ayant pour composantes les dérivées des fonctions de forme :

$$B = \begin{pmatrix} N_{1,x} & 0 & N_{2,x} & 0 & \dots & N_{l,x} & 0 \\ 0 & N_{1,y} & 0 & N_{2,y} & \dots & 0 & N_{l,y} \\ N_{1,y} & N_{1,x} & N_{2,y} & N_{2,x} & \dots & N_{l,y} & N_{l,x} \end{pmatrix}. \tag{2.17}$$

En prenant en compte les équations (2.4), l'équation (2.15) devient :

$$\sum_{i \in I} \int_{\Omega_e} B^T D B \; d\Omega_e u_n = \sum_{i \in I} \left(\int_{\Omega_e} f_V N u_n \; d\Omega_e + \int_{\partial\Omega_e \cap \partial\Omega} f_S N u_n \; d\Gamma \right), \tag{2.18}$$

L'équation (2.18) donne une forme discrétisée sur chaque élément et ainsi permet de définir la matrice k_e et le vecteur f_e. La matrice k_e appelée matrice de raideur de l'élément est donc égale à :

$$k_e = \int_{\Omega_e} B^T D B \; d\Omega_e, \tag{2.19}$$

et le vecteur des sollicitations f_e vaut :

$$f_e = \int_{\Omega_e} N f_V \; d\Omega_e + \int_{\partial\Omega_e \cap \partial\Omega} N f_S \; d\Gamma \tag{2.20}$$

2.2.2 Calcul de la matrice de raideur élémentaire

Pour simplifier les calculs des matrices de raideur sur chaque élément, un élément de référence est utilisé. Chaque élément réel peut être défini par transport à partir d'un élément de référence. Tout élément réel peut être retrouvé grâce à la transformation géométrique d'un élément de référence, de la même façon tout élément de référence peut être retrouvé grâce à la transformation géométrique inverse.

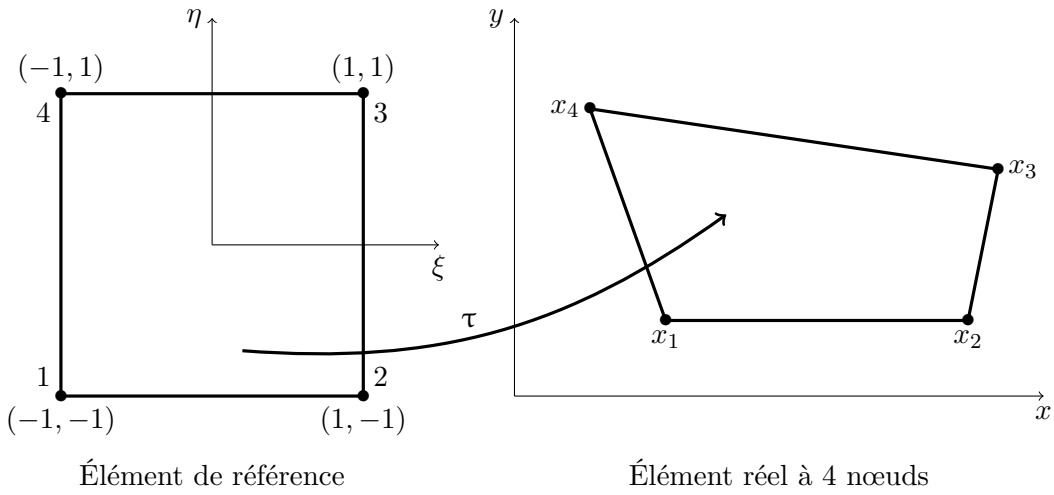


FIGURE 2.3 – Transformation τ

Les coordonnées de l'élément réel dépendent des coordonnées de l'élément de référence et des coordonnées des nœuds de l'élément réel. La transformation τ varie pour chaque élément, en fonction des nœuds géométriques.

$$\tau\colon \xi \mapsto x = \tau(\xi, x_1, x_2, \ldots, x_m). \tag{2.21}$$

Chaque transformation τ doit être bijective. Un nœud de l'élément de référence devient un nœud de l'élément réel, il en est de même pour les frontières. Pour simplifier les calculs, la transformation est choisie linéaire ou quadratique par rapport aux coordonnées des nœuds des éléments réels.

$$\tau\colon \xi \mapsto \tau(\xi) = \bar{N} x_n, \tag{2.22}$$

avec $\bar{N}$ appelées les fonctions de transformation géométrique. Dans le cas d'éléments isoparamétriques, les fonctions d'interpolation de la géométrie sont les mêmes que les fonctions de forme qui interpolent la solution.

La matrice jacobienne de la transformation géométrique est donnée par (2.23), on pose

j la matrice inverse.

$$J = \begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{pmatrix}, \tag{2.23}$$

$$j = J^{-1}. \tag{2.24}$$

L'introduction de cette matrice jacobienne permet le changement de variables dans l'intégration afin de se ramener à une intégrale plus facile à calculer. En effet, sur l'élément de référence dont l'aire est connue, la solution est plus facile à intégrer que sur des éléments réels dont la forme varie. C'est pourquoi l'intégration afin de calculer la matrice de raideur se fait sur l'élément de référence grâce la matrice jacobienne. Le changement de variables doit aussi être mis en place pour B, soit B_ξ une matrice semblable à B définie dans l'équation (2.25),

$$B_\xi = \begin{pmatrix} N_{1,\xi} & 0 & N_{2,\xi} & 0 & \dots & N_{l,\xi} & 0 \\ 0 & N_{1,\eta} & 0 & N_{2,\eta} & \dots & 0 & N_{l,\eta} \\ N_{1,\eta} & N_{1,\xi} & N_{2,\eta} & N_{2,\xi} & \dots & N_{l,\eta} & N_{4,\xi} \end{pmatrix}. \tag{2.25}$$

Une relation entre B_ξ et B peut être trouvée grâce à la matrice jacobienne inverse j de la transformation géométrique :

$$B = jB_\xi. \tag{2.26}$$

La matrice de raideur devient après transformation du domaine :

$$k_e = \int_{V^r} B_\xi^T j^T D\, jB_\xi det(J) d\xi d\eta, \tag{2.27}$$

L'intégration se fait en utilisant une méthode de quadrature. La méthode de quadrature de Gauss permet d'intégrer de manière exacte des polynômes de faible degré. Si n_1 points sont pris dans la direction ξ, et n_2 points dans la direction η, la méthode de Gauss permet d'intégrer exactement un polynôme d'ordre $2n_1-1$ en ξ et d'ordre $2n_2-1$ en η. Usuellement pour un quadrangle à quatre nœuds sur chaque élément du maillage, l'intégration se fait grâce à une intégration de Gauss à deux points par dimension. Les coordonnées des points d'intégration de Gauss ainsi que les poids liés w sont donnés dans les livres sur la méthode des éléments finis [DT87], [TRO92]. La quadrature d'une intégration double sur $[-1,1]^2$ est donnée par l'équation (2.28) :

$$\int_{-1}^{1}\int_{-1}^{1} y\left(\xi,\eta\right) d\xi d\eta \simeq \sum_{i=1}^{n_1,n_2} w_i y\left(\xi_i,\eta_i\right). \tag{2.28}$$

La formulation de la matrice de raideur devient par la quadrature de Gauss :

$$k_e = \sum_{i=1}^{n_1,n_2} w_i B_{\xi,i}^T j_i^T D\, j_i B_{\xi,i} det(J_i). \tag{2.29}$$

2.2.3 Assemblage des matrices

Après le calcul des matrices de raideur sur chaque élément, il faut assembler la matrice K correspondant au système global. Un vecteur global de forces F doit être aussi assemblé en respectant l'adressage fait pour la matrice K.

Une fois ces matrices assemblées, il reste à résoudre une équation matricielle :

$$KU = F, \tag{2.30}$$

avec K la matrice de rigidité, U le vecteur des degrés de liberté du système et F le vecteur des forces externes au système. U contient tous les déplacements en x et y en chaque nœud du domaine.

Par la suite, nous ne parlerons que des éléments quadrilatéraux qui servent de support pour illustrer notre méthode. L'extension au cas hexaédrique de notre méthode ne devrait pas poser de problèmes majeurs. Nous prenons des éléments Q1 quadrilatéraux à quatre nœuds correspondant aux quatre coins de l'élément, montrés sur la figure 2.4.

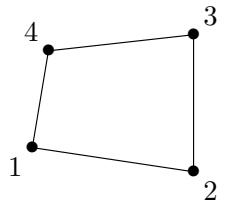


FIGURE 2.4 – Élément quadrilatéral à quatre nœuds

Dans notre cas, les fonctions de forme N_i valent alors un en un seul point d'approximation x_i et s'annulent pour les autres,

$$N_j(x_i) = \begin{cases} 0 & \text{si } i \neq j \\ 1 & \text{si } i = j. \end{cases} \tag{2.31}$$

Les fonctions de forme sont données par les équations suivantes :

$$N_1(\xi, \eta) = \frac{1}{4}(1-\xi)(1-\eta), \tag{2.32}$$

$$N_2(\xi, \eta) = \frac{1}{4}(1+\xi)(1-\eta), \tag{2.33}$$

$$N_3(\xi, \eta) = \frac{1}{4}(1+\xi)(1+\eta), \tag{2.34}$$

$$N_4(\xi, \eta) = \frac{1}{4}(1-\xi)(1+\eta). \tag{2.35}$$

2.2.4 Calcul des contraintes

Une fois les déplacements trouvés, nous nous intéressons au calcul des contraintes. Dans le cas des petites déformations, le tenseur des déformations est égal à :

$$\varepsilon = Bu. \tag{2.36}$$

Afin de retrouver les contraintes, la loi de Hooke est appliquée comme montré précédemment. Le tenseur des contraintes vaut :

$$\sigma = D\varepsilon = DBu. \tag{2.37}$$

Les contraintes peuvent être calculées en n'importe quel point à l'intérieur d'un élément en effectuant le produit DBu. La contrainte est continue à l'intérieur d'un élément mais il peut y avoir un saut à l'interface entre les éléments. Un nœud situé sur un sommet peut être partagé entre plusieurs éléments, la contrainte en ce nœud peut donc posséder des valeurs différentes.

Calcul des contraintes aux nœuds

L'estimation des contraintes aux nœuds est intéressante du point de vue de l'ingénieur. Elle sert à la fois pour une meilleure visualisation et à la fois pour obtenir le maximum des contraintes. Les nœuds des éléments se trouvent sur les bords, là où les contraintes sont souvent fortes, calculer les contraintes en ces points permet donc d'obtenir le maximum des contraintes.

Dans un souci de gain de performance, dans certains codes industriels la matrice B calculée pour former la matrice de rigidité élémentaire, est stockée pour des usages ultérieurs comme le calcul des contraintes. De ce fait, la matrice B n'est connue qu'aux points de Gauss sur chaque élément. Les contraintes ne peuvent être restituées qu'en chaque point d'intégration.

Afin de trouver ces contraintes aux nœuds, le calcul se fait en deux étapes. La première consiste à restituer les contraintes aux points de Gauss sur chaque élément. La deuxième utilise une méthode d'interpolation sur ces contraintes trouvées afin d'estimer les contraintes aux nœuds. Les contraintes aux nœuds et les contraintes aux points d'intégration sont liées par la relation :

$$\sigma_{pg} = N_{pg}\overline{\sigma}, \tag{2.38}$$

avec σ_{pg} la contrainte calculée aux points de Gauss, N_{pg} la matrice des fonctions de forme prises aux points de Gauss et $\overline{\sigma}$ la contrainte aux nœuds. Afin de trouver les contraintes aux nœuds $\overline{\sigma}$, la norme du résidu doit être minimisée :

$$\|r\|_2 = \|\sigma_{pg} - N_{pg}\overline{\sigma}\|_2 . \tag{2.39}$$

Minimiser ce résidu par moindres carrés revient à résoudre l'équation (2.40).

$$N_{pg}^T N_{pg} \overline{\sigma} = N_{pg}^T \sigma_{pg}. \tag{2.40}$$

Sachant que la contrainte est discontinue sur l'interface des éléments, chaque nœud se retrouve avec autant de valeurs que d'éléments qui l'entourent. Une moyenne de ces valeurs est alors effectuée pour déterminer une contrainte en chaque nœud. Ainsi si un nœud est entouré de n_e éléments, la contrainte à ce nœud est calculée de cette façon :

$$\sigma = \frac{1}{n_e} \sum_{i=1}^{n_e} \overline{\sigma_i}. \tag{2.41}$$

2.2.5 Ajout de fonctions de forme : les extra-shapes

Afin d'améliorer la précision des calculs, notamment dans les situations où la flexion est dominante, au lieu d'augmenter le nombre d'éléments, nous nous intéressons à une méthode donnant une base de fonctions de forme plus riche qui comprend tous les monômes de degré deux. Un outil déjà implémenté dans Ansys pour la méthode des éléments finis ajoute deux nouvelles fonctions de forme. Cette méthode appelée « extra-shapes » a été introduite par Wilson, et al [WIL+76] puis Taylor, et al [TBW76]. Cet ajout de fonctions de forme va permettre de nouvelles déformations de l'élément et ainsi mieux approcher la réalité physique.

Base polynomiale

La base de polynômes des fonctions de formes des éléments quadrilatéraux à quatre nœuds est :

$$P = \begin{pmatrix} 1 & \xi & \eta & \xi\eta \end{pmatrix}. \tag{2.42}$$

Une base polynomiale complète d'ordre deux est donnée par (2.43)

$$P = \begin{pmatrix} 1 & \xi & \eta & \xi\eta & \xi^2 & \eta^2 \end{pmatrix}. \tag{2.43}$$

Afin d'obtenir cette base, Wilson, et al [WIL+76] proposent de rajouter deux fonctions pour ajouter les termes quadratiques à la base (2.42). Ces fonctions de forme supplémentaires permettent une déformation parabolique le long des éléments. Les fonctions de forme ajoutées sont définies dans les équations (2.44), (2.45) :

$$h_1 = 1 - \xi^2, \tag{2.44}$$

$$h_2 = 1 - \eta^2. \tag{2.45}$$

La figure 2.5 montre l'allure de ces deux fonctions de forme supplémentaires.

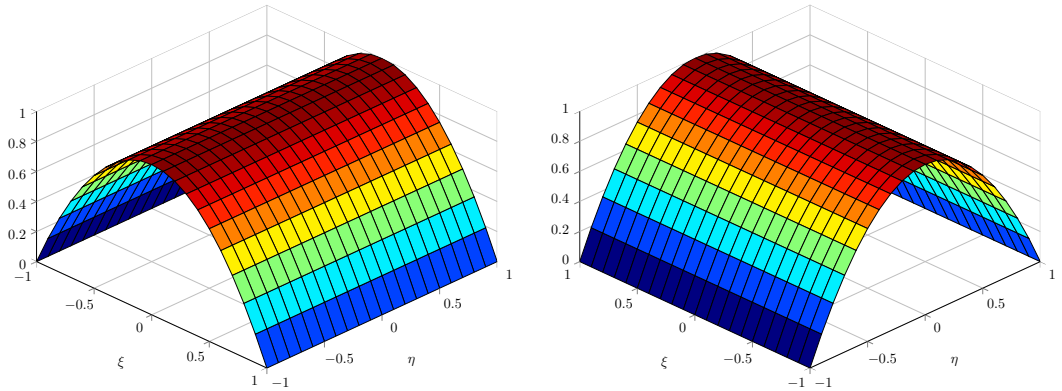


FIGURE 2.5 – Fonctions de forme quadratique

Afin de rajouter ces fonctions de forme, il est nécessaire de changer les calculs des matrices de raideur, des contraintes... Pour trouver leur définition, partons du calcul des déformations. Les déformations sont directement reliées aux fonctions de forme par les équations de l'élasticité :

$$\varepsilon_x = \frac{\partial u_x}{\partial x}, \tag{2.46}$$

$$\varepsilon_y = \frac{\partial u_y}{\partial y}, \tag{2.47}$$

$$2\varepsilon_{xy} = \frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}. \tag{2.48}$$

En prenant en compte l'ajout des fonctions de forme, ces équations deviennent :

$$\begin{pmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_{xy} \end{pmatrix} = \begin{pmatrix} \frac{\partial N_1}{\partial x} & 0 & \frac{\partial N_2}{\partial x} & 0 & \frac{\partial N_3}{\partial x} & 0 & \frac{\partial N_4}{\partial x} & 0 \\ 0 & \frac{\partial N_1}{\partial y} & 0 & \frac{\partial N_2}{\partial y} & 0 & \frac{\partial N_3}{\partial y} & 0 & \frac{\partial N_4}{\partial y} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial y} & \frac{\partial N_3}{\partial x} & \frac{\partial N_4}{\partial y} & \frac{\partial N_4}{\partial x} \end{pmatrix} \begin{pmatrix} u_{x1} \\ u_{y1} \\ u_{x2} \\ u_{y2} \\ u_{x3} \\ u_{y3} \\ u_{x4} \\ u_{y4} \end{pmatrix} + \begin{pmatrix} \frac{\partial h_1}{\partial x} & 0 & \frac{\partial h_2}{\partial x} & 0 \\ 0 & \frac{\partial h_1}{\partial y} & 0 & \frac{\partial h_2}{\partial y} \\ \frac{\partial h_1}{\partial y} & \frac{\partial h_1}{\partial x} & \frac{\partial h_2}{\partial y} & \frac{\partial h_2}{\partial x} \end{pmatrix} \begin{pmatrix} u_{xe1} \\ u_{ye1} \\ u_{xe2} \\ u_{ye2} \end{pmatrix}, \quad (2.49)$$

où u_{xe1}, u_{ye1}, u_{xe2} et u_{ye2} sont les coefficients de la solution selon les éléments de la base « extra-shapes ». L'équation (2.49) peut être réduite, elle devient alors :

$$\varepsilon = B_n u_n + G u_e, \quad (2.50)$$

$$= \begin{pmatrix} B_n & G \end{pmatrix} \begin{pmatrix} u_n \\ u_e \end{pmatrix}, \quad (2.51)$$

$$= \tilde{B}\tilde{u}. \quad (2.52)$$

Dans l'équation décrivant les déformations (2.50), peuvent être retrouvés le calcul habituel des déformations ($B_n u_n$) et une partie additionnelle correspondant à l'ajout des nœuds non géométriques ($G u_e$). Posons la nouvelle matrice $\tilde{B}$ comme la concaténation des matrices B_n et G. La matrice de raideur d'un élément est directement calculée grâce à

cette matrice $\tilde{B}$, le calcul de la matrice de raideur devient :

$$K_e = \int_\Omega \tilde{B}^T D \tilde{B} dV, \tag{2.53}$$

$$= \int_\Omega \begin{pmatrix} B_n^T D B_n & B_n^T D G \\ G^T D B_n & G D G \end{pmatrix} dV, \tag{2.54}$$

$$= \begin{pmatrix} K_{nn} & K_{ne} \\ K_{en} & K_{ee} \end{pmatrix}, \tag{2.55}$$

où K_{nn} prend en compte seulement les noeuds géométriques, K_{ee} seulement les nœuds ajoutés. La matrice de raideur ainsi obtenue est de taille 12×12.

Les déplacements aux nœuds non géométriques sont élémentaires, c'est-à-dire non partagés avec les voisins. Ils peuvent donc être éliminés au niveau de l'élément en utilisant une technique de condensation par complément de Schur. Ainsi la matrice de raideur peut être condensée en une matrice 8×8, comme le nombre de degrés de liberté. Le vecteur force est étendu de façon similaire :

$$F_e = \int_\Omega \begin{pmatrix} B_n^T D \varepsilon \\ G^T D \varepsilon \end{pmatrix} dV, \tag{2.56}$$

$$= \begin{pmatrix} F_n \\ F_e \end{pmatrix}. \tag{2.57}$$

De même que pour la matrice de raideur, le vecteur force peut être écrit avec douze termes puis condensé en un vecteur de huit termes. En partant de l'équilibre au niveau assemblé et du fait que u_e ne dépende que d'un élément, en revenant sur chaque élément, la matrice élémentaire et le vecteur force sont trouvés. La matrice de raideur élémentaire ainsi que le vecteur force obtenus avant condensation sont donnés par l'équation matricielle :

$$\begin{pmatrix} K_{nn} & K_{ne} \\ K_{en} & K_{ee} \end{pmatrix} \begin{pmatrix} u_n \\ u_e \end{pmatrix} = \begin{pmatrix} F_n \\ 0 \end{pmatrix}. \tag{2.58}$$

L'équation (2.58) donne le système d'équations :

$$\begin{cases} K_{nn} u_n + K_{ne} u_e &= F_n \\ K_{en} u_n + K_{ee} u_e &= 0 \end{cases} \tag{2.59}$$

$$\Longleftrightarrow \begin{cases} K_{nn} u_n + K_{ne} u_e &= F_n \\ u_e &= -K_{ee}^{-1} K_{en} u_n. \end{cases} \tag{2.60}$$

La flexibilité apportée par cet ajout de formes peut empêcher l'élément de passer le « patch-test », montré par R.L. Taylor et al dans [TAY+86]. Il n'y a pas continuité du déplacement entre deux éléments lors du test du mode rigide. Pour assurer de passer le test du mode rigide, c'est-à-dire que si le champ de déplacements de l'élément est tel qu'il est à déformations nulles, le champ de déplacements est imposé continu le long de l'arrête commune à deux éléments voisins, c'est a dire u_e prend la valeur zéro lorsque u_n correspond à un déplacement de corps rigide. Sous ces hypothèses, la deuxième équation de (2.60) donne :

$$\int_\Omega G \; d\Omega = 0. \tag{2.61}$$

La preuve que le « patch-test » est satisfait est faite par Taylor, et al dans [TBW76]. La matrice G vaut :

$$G = \begin{pmatrix} \frac{\partial h_1}{\partial x} & 0 & \frac{\partial h_2}{\partial x} & 0 \\ 0 & \frac{\partial h_1}{\partial y} & 0 & \frac{\partial h_2}{\partial y} \\ \frac{\partial h_1}{\partial y} & \frac{\partial h_1}{\partial x} & \frac{\partial h_2}{\partial y} & \frac{\partial h_2}{\partial x} \end{pmatrix}, \tag{2.62}$$

$$= \begin{pmatrix} j_{11} & j_{12} & 0 & 0 \\ 0 & 0 & j_{21} & j_{22} \\ j_{21} & j_{22} & j_{11} & j_{12} \end{pmatrix} \begin{pmatrix} \frac{\partial h_1}{\partial \xi} & 0 & \frac{\partial h_2}{\partial \xi} & 0 \\ \frac{\partial h_1}{\partial \eta} & 0 & \frac{\partial h_2}{\partial \eta} & 0 \\ 0 & \frac{\partial h_1}{\partial \xi} & 0 & \frac{\partial h_2}{\partial \xi} \\ 0 & \frac{\partial h_1}{\partial \eta} & 0 & \frac{\partial h_2}{\partial \eta} \end{pmatrix}, \tag{2.63}$$

$$= \tilde{j} D_e. \tag{2.64}$$

Pour arriver à satisfaire l'équation (2.61), Taylor, et al [TBW76] propose de prendre $\tilde{j}$ au centroïde de l'élément. En posant j_c la matrice $\tilde{j}$ prise au centroïde de l'élément, l'équation (2.64) devient :

$$G_c = j_c D_e. \tag{2.65}$$

Ainsi la matrice de raideur non condensée K_e donnée par l'équation (2.53) est calculée en remplaçant la matrice G par la matrice G_c.

Grâce à la condensation par complément de Schur, cette méthode maintient la taille du système matriciel malgré un ajout de fonctions de forme. Ainsi, il n'y aura pas de perte de temps lors de la résolution du système.

Calcul des contraintes

Cette méthode permet de meilleurs résultats sur les contraintes grâce à la distorsion parabolique qu'elle permet sur chaque élément. L'ajout de fonctions de forme donnera notamment de meilleurs résultats pour des états de contraintes de type flexion. Une autre particularité de cette méthode est qu'elle rend constante la contrainte en x selon x et la contrainte en y selon y sur chaque élément.

2.2.6 Conclusion

La méthode des éléments finis est une méthode très utilisée en mécanique et notamment en mécanique des structures. Elle donne des résultats proches de la solution exacte lorsque le maillage est suffisamment fin. Or, le temps de maillage et surtout de calcul peut considérablement augmenter selon la complexité de la géométrie. Dans le cas d'une analyse d'une propagation de fissure, il est nécessaire de remailler lors de la propagation de la fissure. Ces contraintes importantes de maillage amènent à penser au développement de nouvelles méthodes. La prochaine section montre une autre méthode existante qui s'inspire de la méthode des éléments finis standard tout en proposant une nouvelle approche plus compatible avec les processeurs graphiques, la méthode des éléments finis étendus.

2.3 Méthode des éléments finis étendus

La méthode des éléments finis classique a été appliquée avec succès dans de nombreux problèmes d'analyse numérique en structures mais montrent ses limites lors de la modélisation d'interfaces mouvantes, notamment lors de fissures. La méthode des éléments finis étendus (X-FEM) a été introduite pour adresser les problèmes posés par la mécanique de la rupture [BB99]. Les problèmes de maillage lors de l'analyse d'une fissure de la méthode des éléments finis sont simplifiés tout en gardant la robustesse de cette méthode. Cette méthode discrétise sans mailler explicitement les fissures et évite surtout de remailler au cours de leur propagation [GMB02], [MGB02], [STO+01]. Une convergence est assurée même autour de singularités. Des adaptations de cette méthode sont faites afin de ne pas avoir besoin d'un maillage conforme à la géométrie, dans [BEL+03] le maillage utilisé dans ce cas est cartésien. D'autres adaptations sont faites afin de traiter les cas multi-matériaux [LLK16].

J.M. Melenk et I. Babuška [BM97], [MB96] ont introduit le concept de méthodes de partition de l'unité. Les méthodes de partition de l'unité permettent d'inclure a priori des informations sur le comportement local de la solution dans l'espace éléments finis. Soient Ω un ensemble maillé, I un ensemble de nœuds associés aux fonctions de formes N_i, une solution éléments finis de u est alors définie :

$$u(x) = \sum_{i \in I} u_i N_i(x), \tag{2.66}$$

avec u_i les déplacements nodaux. Une partition de l'unité du domaine Ω est respectée si :

$$\sum_{i \in I} N_i(x) = 1, \quad \forall x \in \Omega. \tag{2.67}$$

Si les fonctions de forme N_i constituent une partition de l'unité du domaine Ω, l'approximation de $u(x)$ peut être enrichie par une fonction $F(x)$. Un avantage d'une méthode respectant la partition de l'unité est qu'elle permet l'enrichissement d'un espace élément fini par une fonction globale $F(x)$ (ou plusieurs) tout en assurant à la matrice de rigidité d'être creuse, ainsi qu'à l'approximation d'être interpolante. La solution u est alors représentée par l'équation (2.68).

$$u(x) = \sum_{i \in I} u_i N_i(x) + \sum_{j \in J} a_j N_j(x) F(x), \tag{2.68}$$

où la première somme représente l'approximation classique faite par la méthode des éléments finis et la deuxième somme l'enrichissement, J est un sous-ensemble de I où sont placés les degrés de liberté enrichis a_j, N_j les fonctions de forme associées à ces degrés de liberté et $F(x)$ la fonction d'enrichissement.

La méthode des éléments finis étendus est une méthode de partition de l'unité. La méthode X-FEM approxime des solutions non lisses, telles qu'une fissure, des cisaillements grâce à un enrichissement des fonctions de forme. Le choix de la fonction $F(x)$ est primordial dans l'amélioration de la méthode des éléments finis. La fonction d'enrichissement est choisie selon la physique que l'on souhaite représenter. Par exemple, dans le cas d'une fissure, la fonction d'enrichissement est choisie discontinue, fonction d'Heaviside, fonction caractéristique ou autres.

Des fonctions de surfaces de niveaux sont parfois utilisées pour montrer les discontinuités dans les éléments. L'utilisation de méthode de surfaces de niveaux couplée à la méthode des éléments finis étendus a été introduite par M. Stolarska et al [STO+01] afin de mieux représenter la fissure. Le niveau zéro de ces fonctions est utilisé pour représenter une discontinuité, de nouveaux nœuds appelés nœuds fantômes sont créés à ces niveaux zéros. La figure 2.6 montre un exemple d'ajout de degrés de liberté sur un maillage cartésien autour d'une fissure.

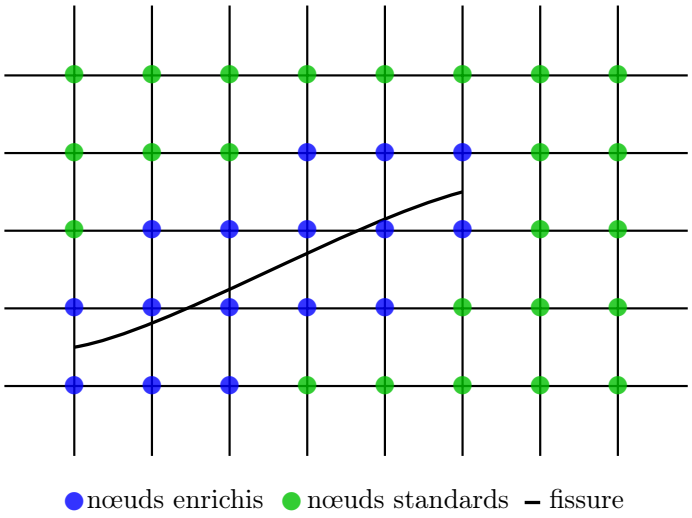


FIGURE 2.6 – Illustration de l'enrichissement autour d'une fissure - Les nœuds enrichis s'ajoutent aux nœuds standards

Le plus souvent dans le cas de fissure, plusieurs fonctions d'enrichissement sont utilisées afin à la fois de représenter la discontinuité ainsi que localiser le front de fissure. Des exemples de fonctions d'enrichissement utilisées dans le cas de fissures sont donnés par J.RANNOU dans [RAN08].

Contrairement à la méthode des éléments finis, qui génère un maillage conforme à la géométrie du modèle, la méthode X-FEM n'a pas besoin d'un maillage conforme, elle fonctionne aussi sur maillage cartésien [BEL+03]. Les éléments sur le bord sont alors enrichis par une fonction adéquate. La figure 2.7 montre un exemple de maillage cartésien et d'enrichissements dans le cas du cercle. Des enrichissements sont aussi possibles afin de traiter le cas multi-matériaux [LLK16].

La fonction d'enrichissement pour la prise en compte du bord peut notamment être la fonction caractéristique du domaine [FB10]. Soit $\phi(x)$ la fonction de distance au bord Γ signée, cette fonction est utile comme fonction surface de niveau :

$$\phi(x) = \pm \min_{x^\star \in \Gamma} \|x - x^\star\| \quad \forall x \in \Omega, \tag{2.69}$$

avec Γ l'interface du domaine Ω. La fonction caractéristique du domaine est donnée par la fonction Heaviside $H(x)$ de cette fonction distance, on pose alors comme fonction d'enrichissement :

$$F(x) = H(\phi(x)). \tag{2.70}$$

Au moment du calcul des matrices de raideur, les discontinuités à l'intérieur d'un élément amènent à chercher d'autres méthodes d'intégration que la méthode de quadrature de Gauss. L'intégration de la méthode des éléments finis étendus se fait de façon différente

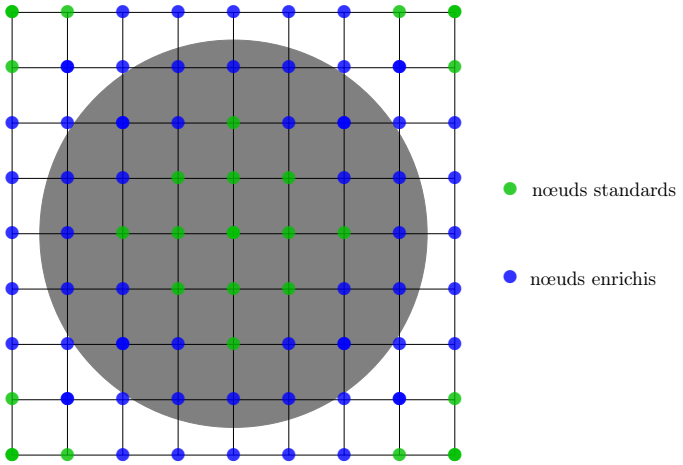


FIGURE 2.7 – Illustration de l'enrichissement X-FEM sur les bords

pour les éléments enrichis ou non. Pour les éléments contenant un seul matériau, l'intégration de Gauss pour un polynôme s'applique toujours. Pour les éléments coupés par la frontière, l'intégration d'une fonction discontinue ne peut se faire de façon exacte par l'intégration de Gauss. Différentes méthodes d'intégration ont été mises en place pour X-FEM, la subdivision d'un élément [FB10], [LAB+05], l'augmentation de l'ordre de la quadrature de Gauss [XK06], ou la recherche d'un polynôme ayant même valeur d'intégrale [VEN06].

La méthode la plus classique, la subdivision en sous-éléments compatibles avec la discontinuité, des triangles en dimension deux et des tétraèdres en dimension trois, permet de se ramener à des formes que l'on sait intégrer et ainsi faire une intégration de Gauss. La figure 2.8 montre deux découpages possibles ainsi que les points de Gauss associés, découpages expliqués par T-P. Fries et T. Belytschko dans [FB10]. Trois points d'intégration sont utilisés par sous-domaine.

G. VENTURA propose une intégration sans subdiviser en sous-domaines [VEN06]. L'idée est d'introduire un polynôme $\tilde{H}$ défini sur tout le domaine Ω, tel que son intégrale donne la valeur exacte de l'intégrale obtenue en divisant les éléments. Il a été prouvé que ce type de polynôme existe. Néanmoins, la partie pleine est toujours facettisée afin de se ramener à des cas connus. Par exemple, en dimension deux avec des éléments quadrilatéraux, G. VENTURA se ramène toujours à deux cas de figure, celui où la discontinuité intersecte deux côtés adjacents, celui où elle intersecte deux côtés opposés. Chaque élément du bord peut être retrouvé grâce à ces deux cas.

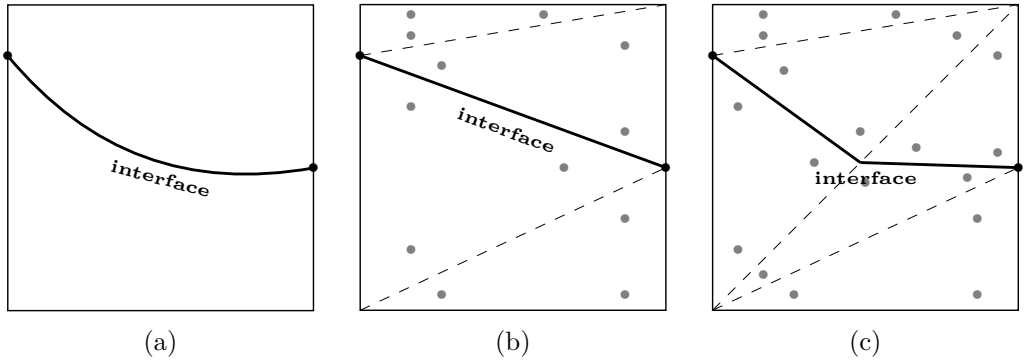


FIGURE 2.8 – Division en sous-domaines. (a) L'élément réel avec l'interface courbe. (b) et (c) deux exemples de découpe, l'interface est interpolée linéairement.

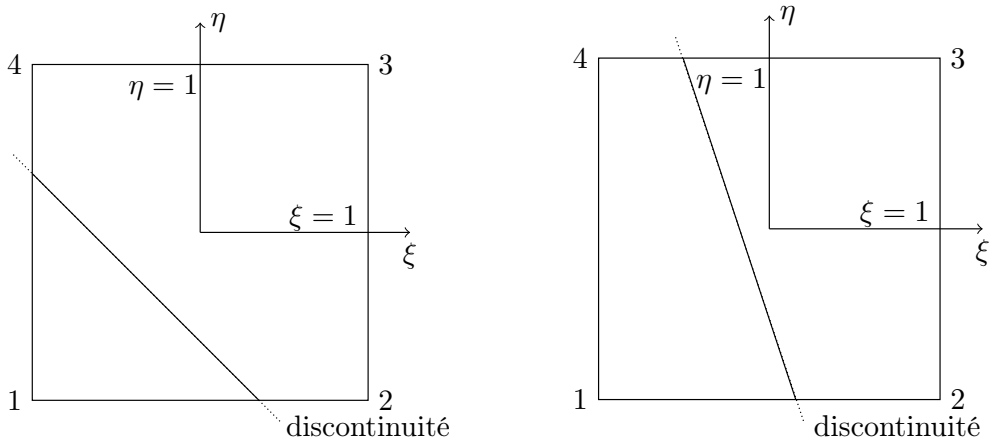


FIGURE 2.9 – Les différentes facettisations des éléments coupés par le bord proposées par G. VENTURA

La méthode éléments finis développée au cours de cette thèse propose une nouvelle méthode d'intégration pour les éléments coupés par la frontière en s'inspirant de la méthode éléments finis étendus.

2.4 Processeurs graphiques

L'objectif final de la thèse étant une implémentation de la nouvelle méthode éléments finis sur processeurs graphiques, cette section fait une brève présentation de ces processeurs et de leur architecture. Cette présentation permet de mieux comprendre pourquoi les méthodes doivent être adaptées lors d'une programmation sur processeurs graphiques.

Les processeurs graphiques ont évolué afin de devenir un outil puissant pour le calcul hautes-performances [OWE+08], [SK10]. Ils disposent notamment d'énormes bande passante et puissance de calcul [OWE+07]. De plus, de nouveaux processeurs graphiques sont en cours de développement et les performances de chaque nouveau processeur évolue rapidement.

L'architecture d'un processeur graphique est une grille de blocs qui eux-mêmes contiennent une grille de « threads » [ND10]. La figure 2.10 montre une représentation de cette architecture, pour un exemple à six blocs et n_x sur n_y threads par blocs. Les « threads » peuvent être exécutés simultanément s'ils ont la même instruction. Si deux « threads » divergent d'instruction lors de leur exécution, une perte de performances a lieu. Il est donc important d'arriver à rendre vectorielle les fonctions utilisées. Les processeurs graphiques modernes conviennent très bien à des calculs massivement parallèles.

La mémoire des processeurs graphiques est nettement plus petite que celle des processeurs centraux. Une adaptation des méthodes est ainsi nécessaire afin de ne pas dépasser la taille mémoire des GPU.

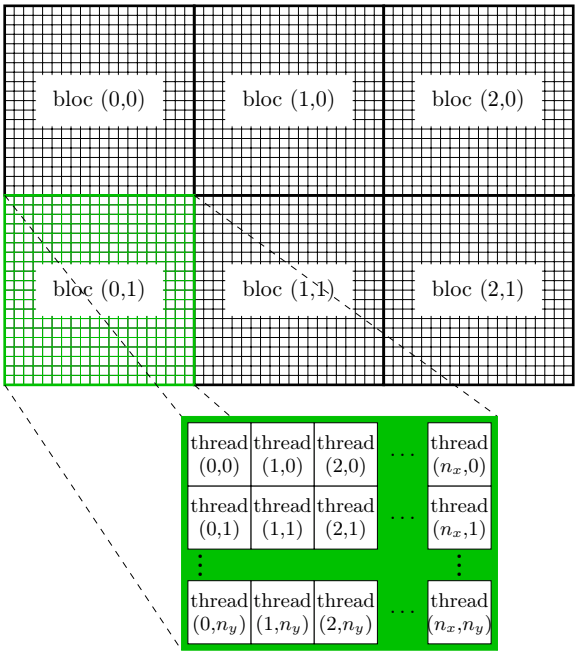


FIGURE 2.10 – Représentation de l'architecture grille des processeurs graphiques en deux dimensions

Le problème général, deux méthodes éléments finis ainsi que les processeurs graphiques viennent d'être présentés. La présentation de l'architecture des processeurs graphiques permet de comprendre pourquoi le maillage conforme de la méthode des éléments finis

n'est pas adapté à ces processeurs.

Le prochain chapitre présente la méthode implémentée au cours de la thèse, une méthode éléments finis compatibles avec les processeurs graphiques modernes.

Chapitre 3

Méthode des éléments finis par intégration implicite de la géométrie

3.1 Description de la méthode

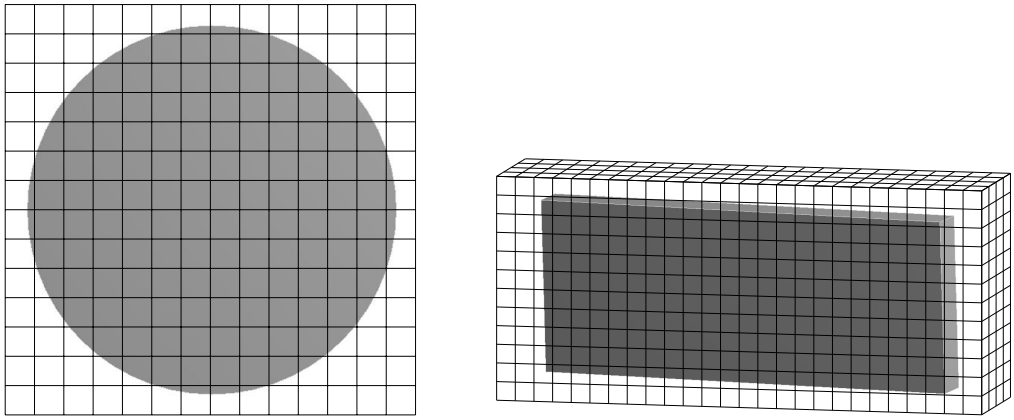
Afin d'utiliser directement l'architecture grille des processeurs graphiques comme base de calcul, il est nécessaire d'adapter les formulations éléments finis à un maillage cartésien. L'utilisation d'un maillage cartésien permettra aussi une bonne parallélisation des calculs. Nous verrons en premier lieu comment nous proposons de représenter une géométrie quelconque sur un maillage cartésien grâce à une fonction surface de niveau (*level set*). Dans un deuxième temps, nous détaillerons la méthode qui nous permet de calculer les matrices éléments finis.

3.2 Représentation d'une géométrie complexe sur un maillage cartésien

La première problématique est la représentation de la géométrie à l'aide d'une grille régulière. En partant de la fonction caractéristique du domaine, nous souhaitons arriver à une bonne représentation de la géométrie tout en stockant peu d'informations. Un couplage de la méthode avec une fonction surfaces de niveau permet une bonne représentation de la géométrie.

De plus, en raison de la mémoire de petite taille dont le processeur graphique dispose, nous souhaitons obtenir les meilleurs résultats sur le maillage le plus grossier possible. Une bonne précision de la représentation de la géométrie permet d'éviter le raffinement du maillage.

Afin de comprendre la difficulté qu'il y a de présenter la géométrie complexe sur un maillage cartésien, des exemples de modèles simples en deux et trois dimensions sont montrés sur la figure 3.1. Ces figures permettent de bien voir qu'aucun maillage n'est fait, la géométrie est simplement immergée dans la grille.



(a) Un disque pris dans une grille régulière 2D

(b) Une poutre prise dans une grille régulière 3D

FIGURE 3.1 – Exemple de modèles définis sur une grille régulière

La géométrie est définie par une fonction de surface de niveau φ représentant une densité. La densité est inférieure à un demi pour les points situés à l'intérieur du domaine Ω et supérieure à un demi pour les autres points. La courbe de niveau $\{\varphi = 0.5\}$ donne le bord du domaine.

En pratique, la fonction de surface de niveau φ est stockée seulement aux nœuds des éléments. Notre approche consiste à utiliser l'approximation φ_h bilinéaire sur chaque élément qui coïncide avec φ aux sommets du maillage.

L'idée est que l'interface est définie comme une surface de niveau d'une fonction à valeurs réelles. L'interface Γ_h du domaine approché Ω_h est définie ainsi :

$$\Gamma_h = \{(x, y) \mid \varphi_h(x, y) = 0.5\}. \tag{3.1}$$

Pour cela, le bord du domaine sur chaque élément est approchée par une fonction bilinéaire en x et y :

$$\Gamma(x, y) = a + bx + cy + dxy. \tag{3.2}$$

Nous partons de la fonction caractéristique du domaine ce qui nous permet d'avoir les points situés sur la frontière. Puis, en partant du fait que la densité vaut un demi sur la frontière, et un sur le point à l'intérieur le plus éloigné de la frontière, quatre points connus sont trouvés facilement afin de retrouver a, b, c et d de l'équation (3.2).

Comparons notre approche avec une méthode existante : les « marching cubes », ou les « marching squares » en 2D, [LZ13]. Cette méthode permet la construction de courbes de niveau par une facettisation de la frontière. La figure 3.2 montre différentes configurations en dimension deux de la partie pleine facettisée par « marching squares ».

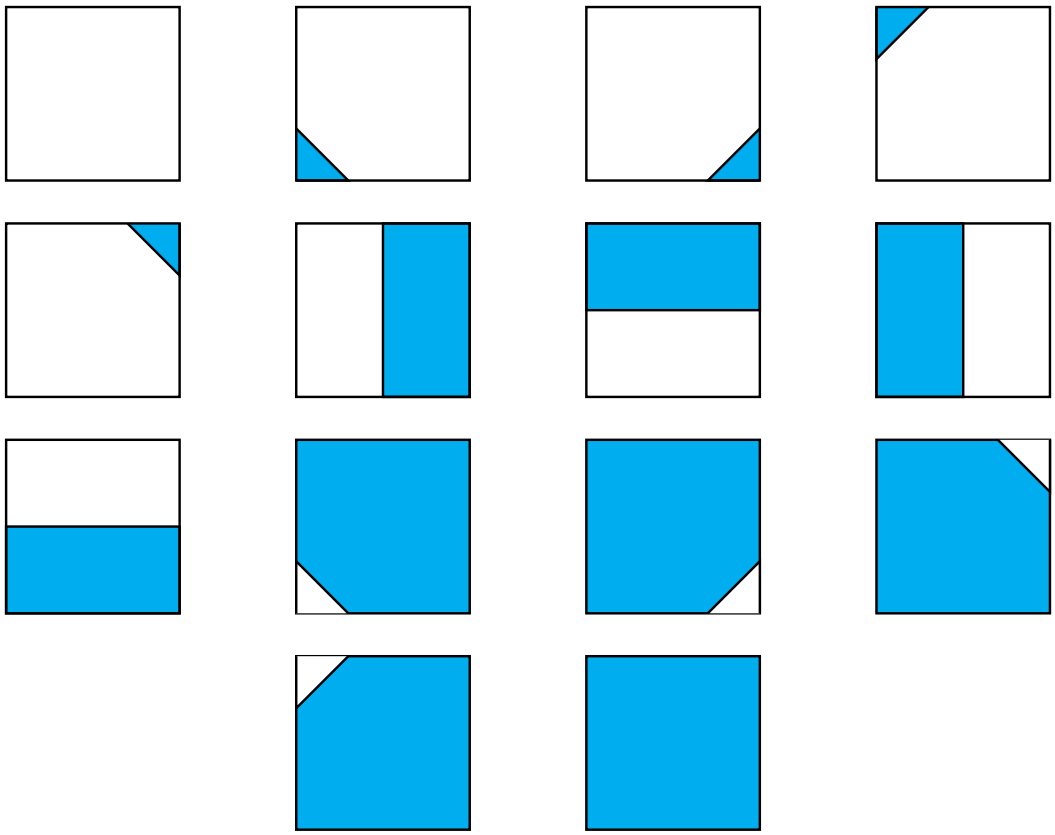


FIGURE 3.2 – Différentes facettisations possibles par « marching squares »

La figure 3.3 montre la facettisation d'un élément contenant une partie de modèle circulaire.

Comme visible sur la figure 3.3, les « marching squares » commettent une erreur qui peut-être très importante.

Notre approximation n'est pas forcément exacte mais meilleure que celle donnée par la méthode de marching squares. La figure 3.4 montre les différences entre l'approximation par marching squares et par notre méthode pour un bord circulaire.

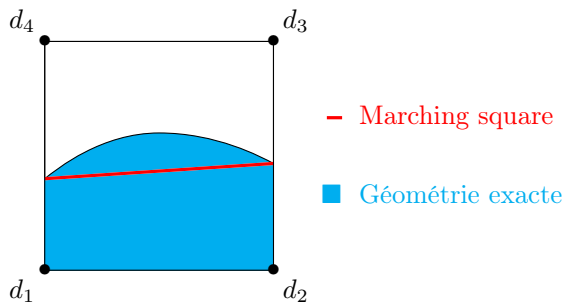


FIGURE 3.3 – Exemple des « marching squares » sur un élément

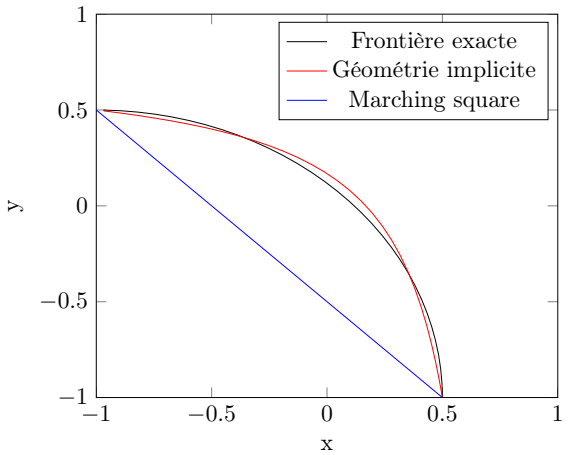


FIGURE 3.4 – Les différentes approximations du bord

L'exemple 3.4 permet de voir la très légère erreur commise par notre méthode. Cette erreur est bien plus faible que celle obtenue par les « marching square ». Un avantage de notre méthode est que la densité en n'importe quel point de l'élément peut être retrouvée par projection bilinéaire. Ainsi, sur chaque élément, il suffit de connaître les quatre densités nodales et les fonctions de formes standards pour retrouver la densité de n'importe quel point.

$$d(x) = \sum_{i=1}^{4} N_i d_i, \tag{3.3}$$

avec $d(x)$ la densité sur l'élément, N_i les fonctions de forme continues. Un autre avantage par rapport aux « marching squares » est que la normale au bord est plus continue entre les éléments. Cela n'est pas négligeable, notamment lors du calcul des contraintes. Les figures 3.5 et 3.6 illustrent la cassure des normales lorsque le bord est approché par facettisation.

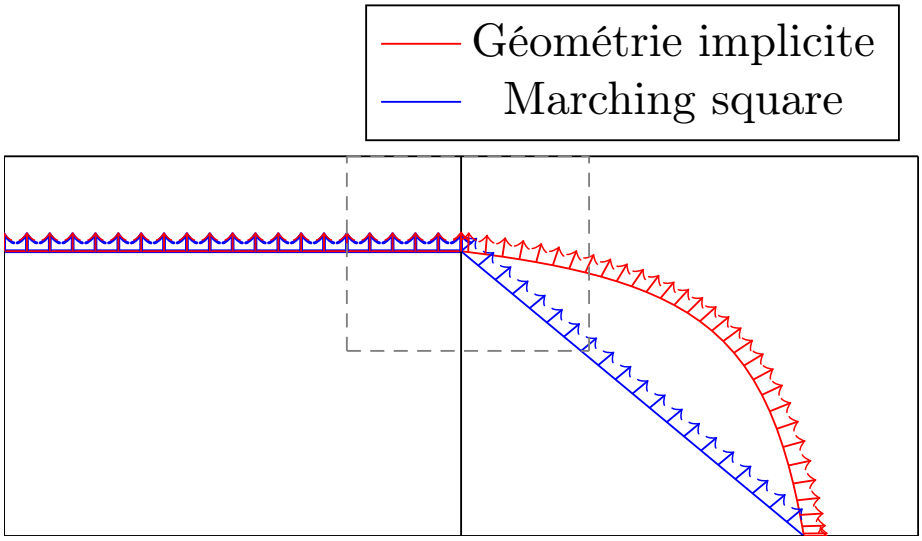


FIGURE 3.5 – La frontière approchée sur deux éléments par « marching squares » et notre méthode, ainsi que les normales au bord. Les pointillés montrent la zone de zoom de la prochaine figure.

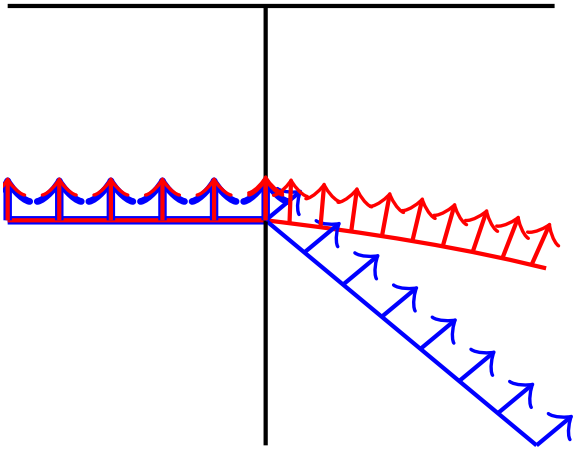


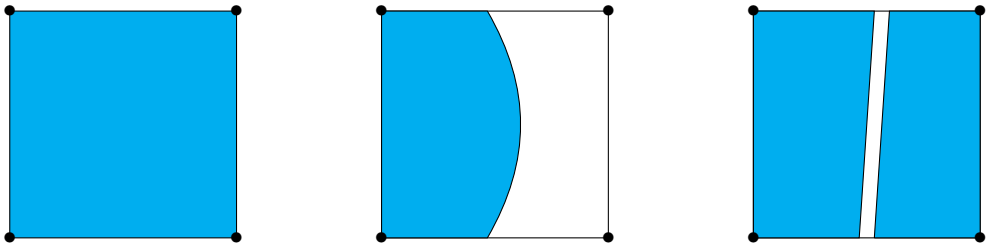
FIGURE 3.6 – Zoom sur la discontinuité des normales au bord entre deux éléments

Les modèles géométriques sont désormais bien définis à l'intérieur de la grille. La prochaine section 3.3 montre la nouvelle intégration utilisée ainsi que l'assemblage des matrices élémentaires qui en découle.

3.3 Intégration des éléments et assemblage

Nous présentons dans cette section notre méthode d'intégration des éléments qui est issue de l'approximation par surface de niveau de la géométrie décrite en section 3.2. L'assemblage des matrices en découle.

Comme le maillage ne coïncide pas exactement avec le bord du domaine ou avec une fissure, différents types d'éléments sont rencontrés. La figure 3.7 présente trois types d'éléments rencontrés sur un maillage cartésien. Au cours de la thèse, nous avons traités seulement les cas des éléments pleins (figure 3.7a) et des éléments coupés par la frontière (figure 3.7b).



(a) Élément plein (b) Élément sur un bord (c) Élément avec une fissure

FIGURE 3.7 – Différents types d'éléments

Comme certains éléments contiennent une discontinuité, la méthode éléments finis standards ne peut être utilisée. La méthode des éléments finis étendus sait en revanche gérer plusieurs matériaux dans un même élément. Une adaptation de cette méthode est faite dans le cas particulier où un des deux matériaux est le vide, ce qui correspond aux éléments coupés par le bord.

Afin d'arriver à intégrer des fonctions discontinues ainsi que des fonctions continues, les fonctions de forme utilisées pour les éléments sur le bord sont modifiées. Sur chaque élément coupé, la fonction de forme va dépendre de la géométrie. La figure 3.8 montre deux fonctions de forme standards sur l'élément de référence, et la figure 3.9 une transformation possible de ces fonctions obtenue en multipliant ces fonctions de forme par la fonction caractéristique du domaine Ω. La figure 3.9 montre les fonctions de forme pour un domaine circulaire (par exemple comme sur la figure 3.1a).

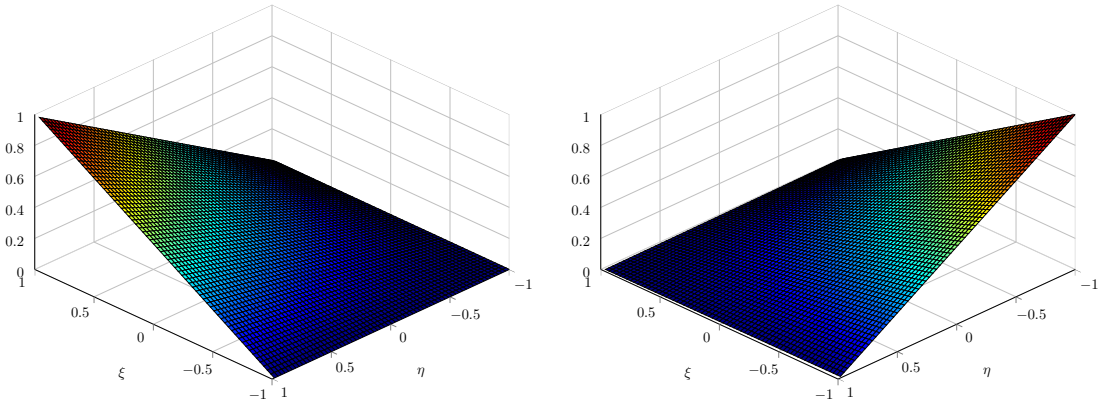


FIGURE 3.8 – Fonctions de forme standard

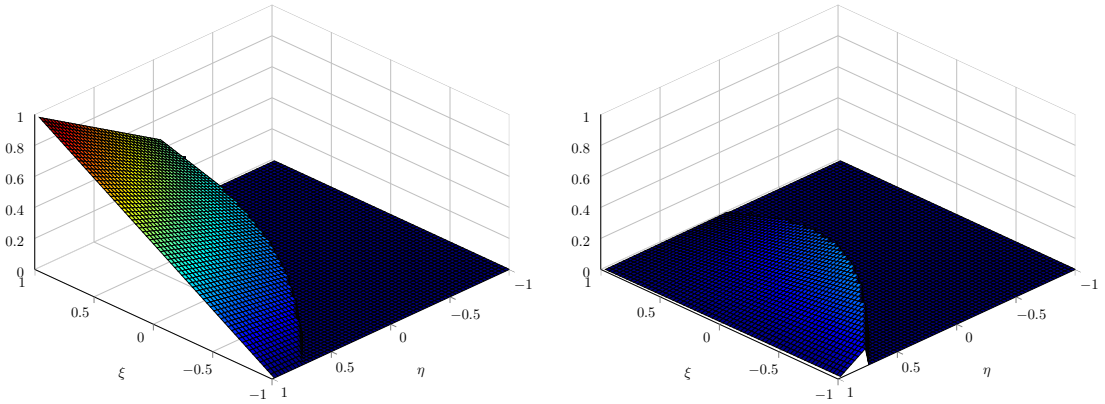


FIGURE 3.9 – Fonctions de forme discontinues

Comme notre maillage n'est plus conforme à la géométrie, il est important de prendre un plus grand nombre de points d'intégration afin d'approcher au mieux l'ensemble Ω pour intégrer la solution sur les éléments coupés. Pour les éléments coupés par la discontinuité du bord, nous n'essayons plus d'approcher un polynôme, les points de Gauss ne donnent donc plus une intégration exacte ni même précise. Nous prenons alors des points d'intégration répartis sur une grille régulière. A chaque point est associé un poids d'intégration.

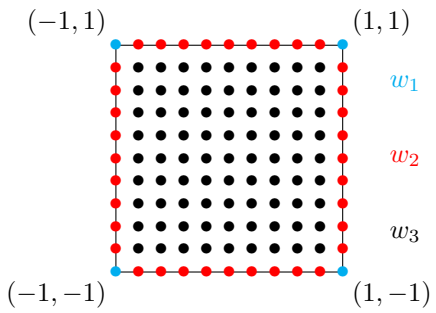


FIGURE 3.10 – Points et poids d'intégration sur l'élément de référence.

La somme des poids d'intégration doit être égale à l'aire de l'élément de référence.

$$\sum_{i=1}^{n_{pi}} w_i = 4, \tag{3.4}$$

avec n_{pi} le nombre de points d'intégration. Chaque point d'intégration correspond à une portion de surface de l'élément de plus ou moins petite taille. Les poids des points sur le coin sont moitié moins importants que ceux sur le bord, et ceux sur le bord sont moitié moins que ceux à l'intérieur afin de correspondre à ces portions de surface. La figure 3.10 montre la répartition des points d'intégration ainsi que leurs poids associés ω_1, ω_2, ω_3.

Grâce à la fonction de surface de niveaux, il est possible de savoir si un point d'intégration est à l'intérieur du domaine ou non. La contribution du point d'intégration est ajoutée seulement s'il est à l'intérieur du domaine.

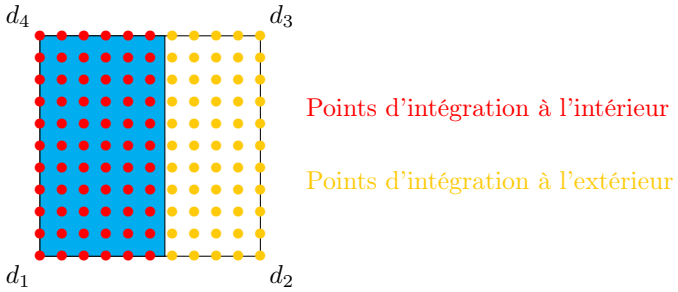


FIGURE 3.11 – Points d'intégration de la méthode à intégration géométrique implicite

Afin de calculer la matrice la raideur, l'équation (2.29) est adaptée pour se ramener à une intégration seulement sur la partie où il y a de la matière. La matrice de raideur est calculée comme montré dans l'équation (3.5).

$$k_e = \sum_{i_k=1}^{s} w_{i_k} B_{i_k}^T D B_{i_k} det(J_{i_k}), \tag{3.5}$$

avec s le nombre de points d'intégration de l'élément à l'intérieur du domaine dont les indices sont notés i_k.

Dans certains cas, une fois l'intégration de k_e faite ainsi que l'assemblage de la matrice de raideur K, la matrice a un conditionnement numérique dégradé, elle est singulière. Pour résoudre ce problème, au lieu de ne pas du tout prendre en compte les contributions des points à l'extérieur du domaine, un ε est introduit, avec ε proche de zéro.

$$k_e = \sum_{i_k=1}^{s} w_{i_k} B_{i_k}^T D B_{i_k} det(J_{i_k}) + \sum_{j_k=1}^{t} \varepsilon w_{j_k} B_{j_k}^T D B_{j_k} det(J_{j_k}), \tag{3.6}$$

avec j_k les indices associés aux t points d'intégration à l'extérieur du domaine.

Une fois ces matrices élémentaires calculées, les assemblages de la matrice K globale ainsi que du vecteur F se font de la même façon que pour la méthode des éléments finis classique. L'équation matricielle (2.30) reste à résoudre. Cette résolution s'est faîte dans le cadre de la thèse de manière directe. Néanmoins, les matrices de raideur des éléments étant égales à un coefficient près, il y a un bon conditionnement pour résoudre le système global de manière itérative.

La prochaine section montre une application en dimension deux de notre méthode.

3.4 Exemples

Dans cette partie, des résultats de la méthode des éléments finis par intégration implicite de la géométrie sont présentés sur une plaque trouée. La plaque est fixée en déplacements en (x, y) sur un côté, une force est appliquée de l'autre côté. Cette plaque trouée, ainsi que ces conditions initiales sont présentées dans la figure 3.12.

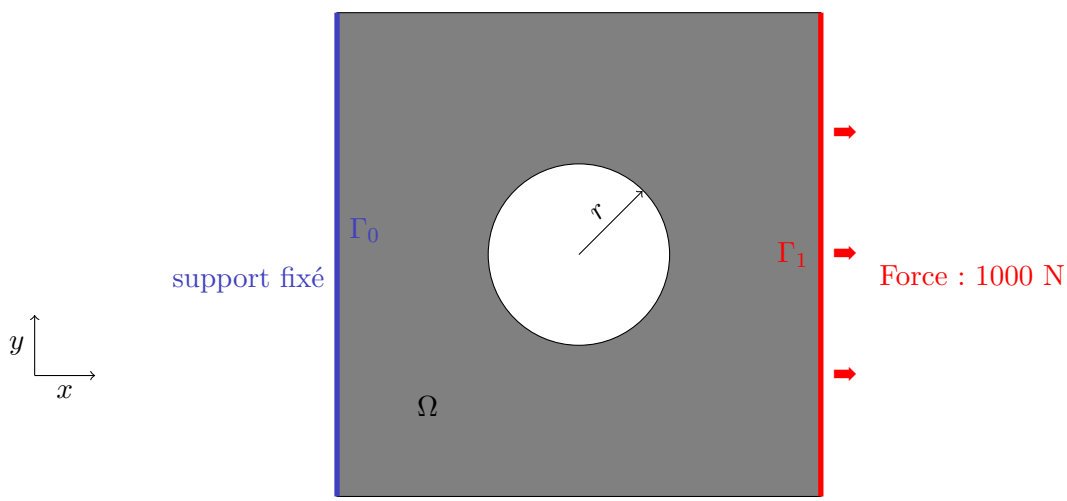
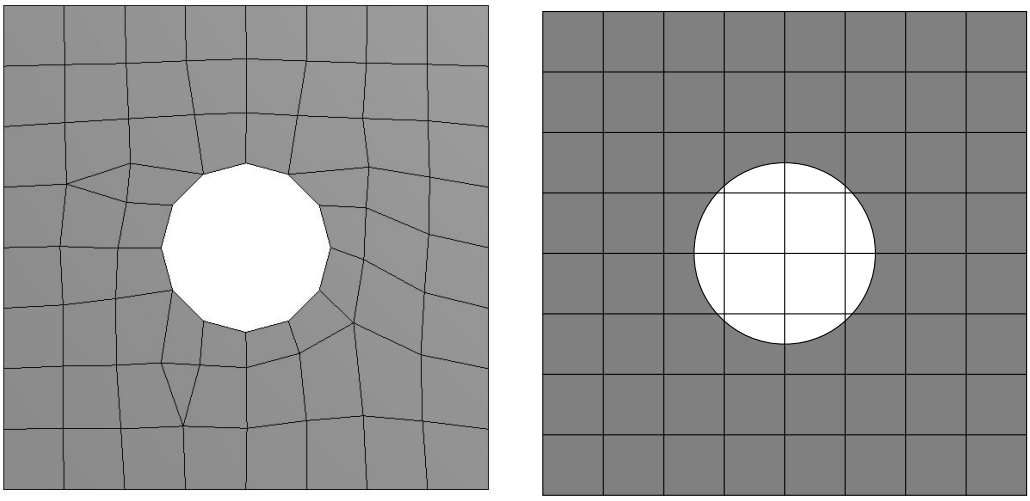


FIGURE 3.12 – Modèle de la plaque trouée avec conditions initiales

On note r le rayon du trou, avec $r > 0$, Ω le domaine, ici la plaque trouée, Γ_1 et Γ_2 sont les deux frontières où des conditions sont appliquées. Le coefficient de poisson ν du materiau est de 0.3 et le module de Young E de 2×10^{11}. Au niveau du trou, une bonne approximation de la solution va être importante, et surtout une bonne approximation des contraintes. En effet, les éléments au niveau du trou vont tous être coupés par la discontinuité du bord, donc plus difficile à intégrer, alors que les contraintes vont être fortes.

3.5 Les différents maillages

Sur la figure 3.13, les différences entre un maillage utilisé par la méthode des éléments finis et un utilisé par la méthode des éléments finis par intégration implicite sont montrées dans le cas de la plaque trouée. Avec un maillage conforme à la géométrie, si le nombre d'éléments n'est pas assez important, une erreur va se produire au niveau du trou. Avec le maillage cartésien, l'erreur n'est pas introduite par la construction du maillage mais lors de la construction des surfaces de niveau.

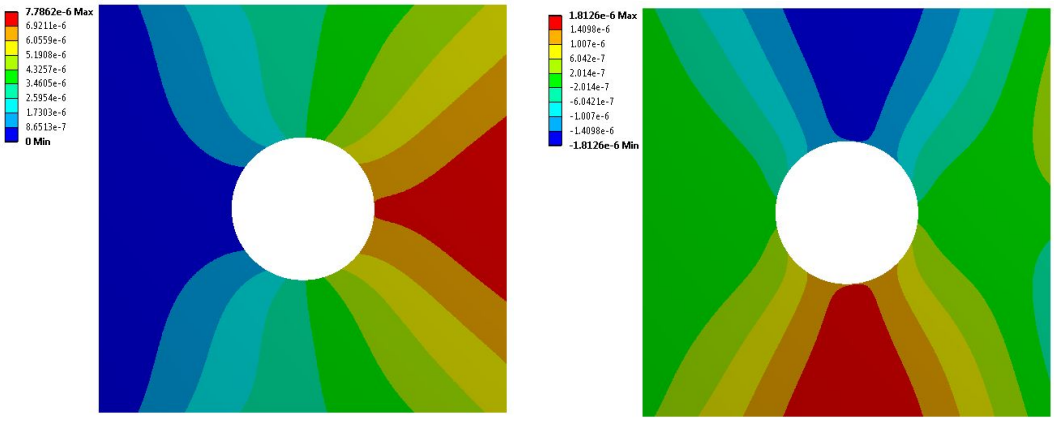


(a) Maillage conforme à la géométrie (b) Maillage cartésien

FIGURE 3.13 – Comparaison d'un maillage conforme et d'un maillage cartésien sur une plaque trouée

3.6 Résultats en déplacements

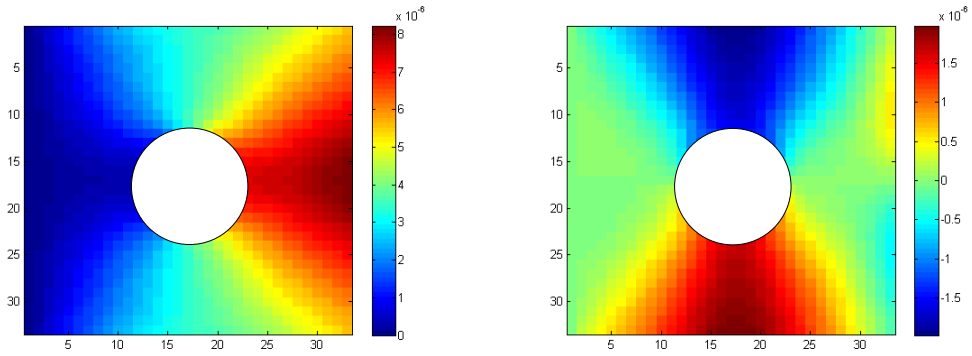
Les résultats en déplacements vont être comparés à des résultats obtenus grâce à Mechanical, ainsi une comparaison avec la méthode des éléments finis classiques pourra être faite. La solution de référence est prise sur un maillage convergé.



(a) Déplacements en x (b) Déplacements en y

FIGURE 3.14 – Les résultats de la méthode des éléments finis en déplacements sur la plaque trouée

Le maximum en déplacements obtenu avec Mechanical selon la direction x est de 7.786×10^{-6}, le maximum selon la direction y de 1.813×10^{-6}. Notre méthode est exécutée avec des grilles de trente-deux par trente-deux éléments, puis soixante-quatre par soixante-quatre. Cinq points d'intégration sont pris par dimension pour le calcul de la matrice élémentaire. Les figures 3.15 et 3.16 montrent les résultats en déplacements de notre méthode. La comparaison point par point des résultats des deux méthodes est chère à mettre en place à cause des maillages différents, l'erreur au maximum est donc regardée.

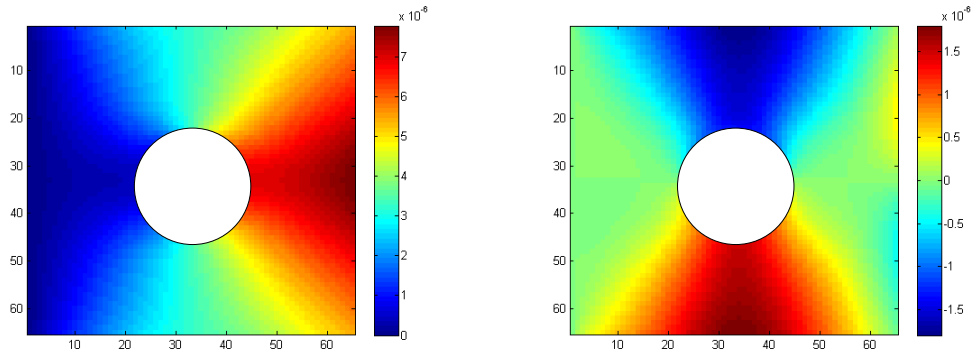


(a) Déplacements en x (b) Déplacements en y

FIGURE 3.15 – Les résultats sur la plaque trouée avec une grille 32×32

Le maximum obtenu en x avec notre méthode et trente-deux points par dimension est

de 7.749×10^{-6}, en y de 1.788×10^{-6}.

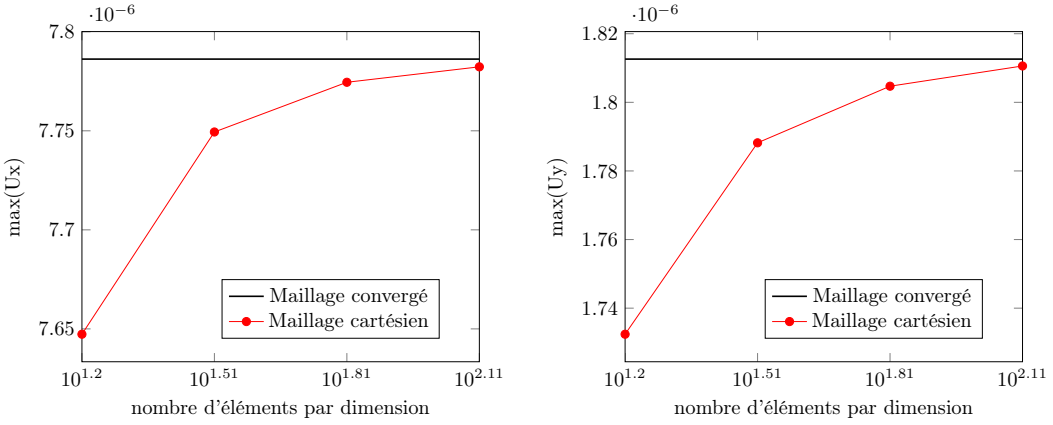


(a) Déplacements en x (b) Déplacements en y

FIGURE 3.16 – Les résultats sur la plaque trouée avec une grille 64×64

Le maximum obtenu en x avec soixante-quatre points par dimension est de 7.775×10^{-6}, en y de 1.805×10^{-6}. L'erreur décroît bien lors d'ajout d'éléments dans la matière et la diminution de la taille de ces éléments.

Afin de vérifier la convergence de la méthode, nous la comparons avec une solution sur un maillage convergé de la méthode des éléments finis standard. Ce maillage de référence contient 330 000 nœuds. Nous nous intéressons à la convergence du maximum des déplacements. La méthode a été lancée sur des grilles de taille 16×16, 32×32, 64×64, puis 128×128. La figure 3.17 montre qu'une convergence vers le maximum en déplacements de la solution sur maillage raffiné a bien lieu.

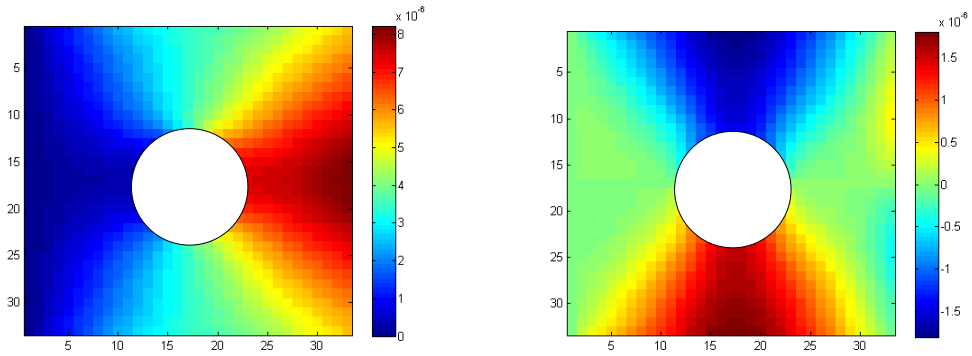


(a) en x (b) en y

FIGURE 3.17 – Convergence du maximum du déplacement

3.6.1 Résultats avec les « extra-shapes »

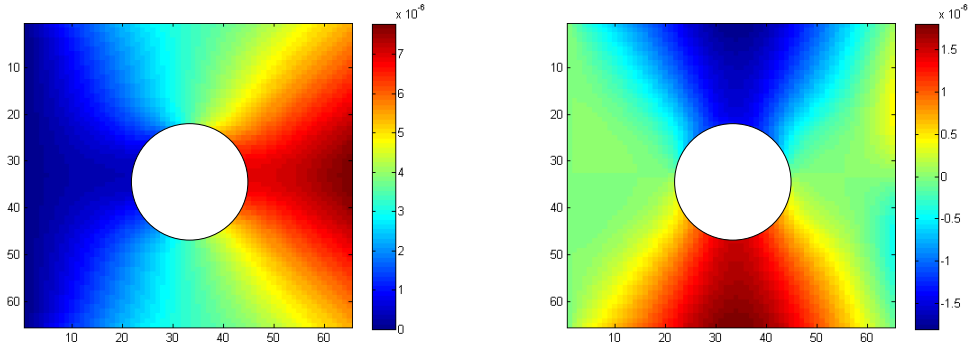
Après ajout des fonctions de forme supplémentaires montrées dans la sous-section 2.2.5, de nouveaux résultats sont obtenus. Les figures 3.18 et 3.19 montrent les résultats en déplacements obtenus après ajout de fonctions de forme.



(a) Déplacements en x (b) Déplacements en y

FIGURE 3.18 – Les résultats en déplacements avec les « extra-shapes » sur la plaque trouée avec une grille 32×32

Le maximum en déplacements avec trente-deux éléments par dimension en x est de 7.779×10^{-6}, en y de 1.808×10^{-6}.

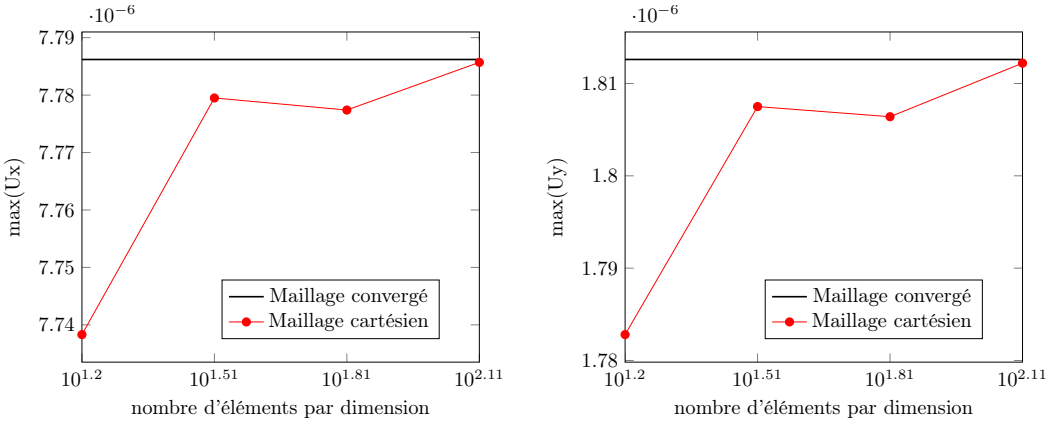


(a) Déplacements en x (b) Déplacements en y

FIGURE 3.19 – Les résultats en déplacements avec les « extra-shapes » sur la plaque trouée avec une grille 64×64

Le maximum en déplacements avec soixante-quatre éléments par dimension en x est de 7.777×10^{-6}, en y de 1.806×10^{-6}. Les maxima obtenus en prenant 32×32 éléments sont

meilleurs que ceux obtenus sans les « extra-shapes ». Ceux obtenus avec 64×64 éléments sont quasiment égaux à ceux obtenus sans les « extra-shapes ». La figure 3.20 affiche le comportement du maximum des déplacements selon le nombre d'éléments par dimension.



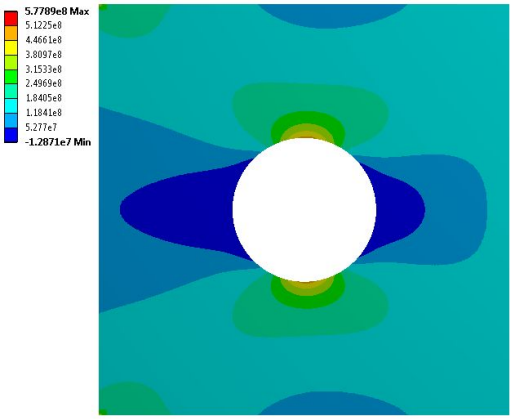
(a) en x (b) en y

FIGURE 3.20 – Convergence du maximum du déplacement

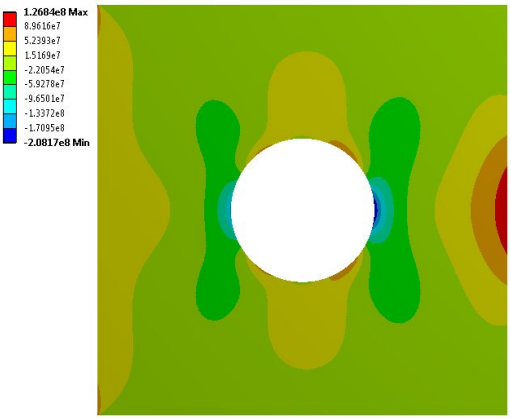
3.7 Résultats en contraintes

Dans cette section, les résultats en contraintes aux nœuds sont exposés. Ces résultats sont de nouveau comparés avec les résultats obtenus par Mechanical sur un maillage convergé. Les résultats obtenus avec la méthode des éléments finis classique sont visibles sur la figure 3.21.

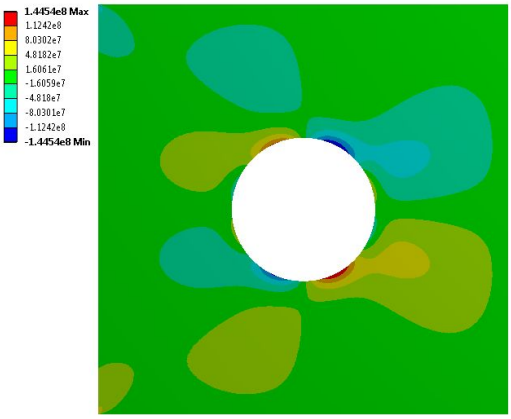
Les contraintes se calculent en post-traitement et ce calcul peut être effectué n'importe où sur l'élément, pas seulement sur les points d'intégration ou les nœuds. C'est pourquoi, pour une meilleure visualisation, les contraintes sont calculées en post-traitement sur des grilles cinq cent douze par cinq cent douze, même si les déplacements restent calculés sur les même grilles que précédemment. Les figures 3.22 et 3.23 montrent les résultats en contraintes obtenus avec une grille 32×32 et 64×64 d'éléments respectivement.



(a) Contraintes en x

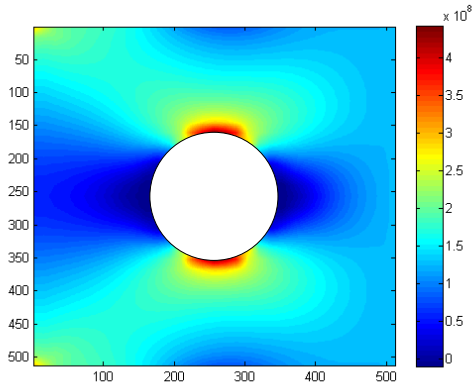


(b) Contraintes en y

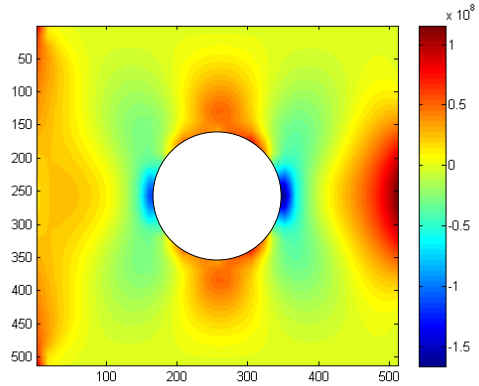


(c) Contraintes en xy

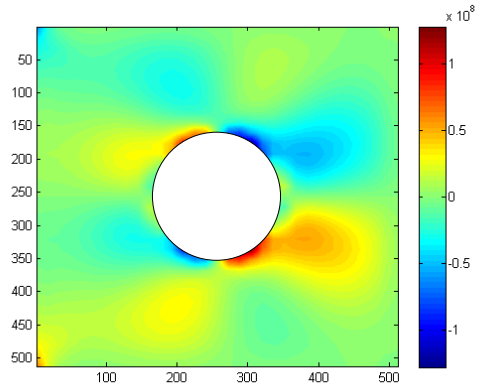
FIGURE 3.21 – Les résultats de la méthode des éléments finis en contraintes aux nœuds sur la plaque trouée obtenus avec Mechanical



(a) Contraintes en x

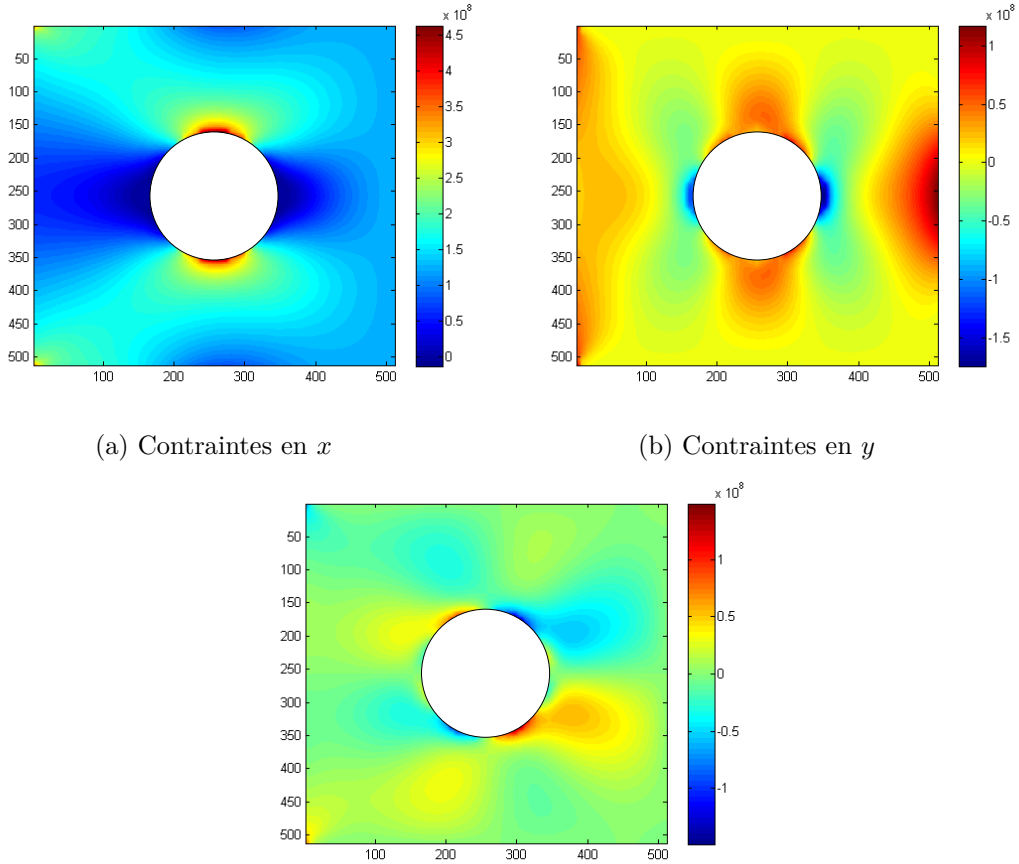


(b) Contraintes en y



(c) Contraintes en xy

FIGURE 3.22 – Les résultats en contraintes sur la plaque trouée après calcul des déplacements sur une grille 32×32



(a) Contraintes en x

(b) Contraintes en y

(c) Contraintes en xy

FIGURE 3.23 – Les résultats en contraintes sur la plaque trouée après calcul des déplacements sur une grille 64×64

La précision des contraintes au voisinage des bords libres est vérifiée. Les contraintes sont vérifiées au niveau des bords libres. Le problème continu a une contrainte normale nulle au niveau des bords libres. La contrainte doit respecter l'équation (3.7).

$$\sigma.n = 0 \quad \text{sur } \Gamma_b, \tag{3.7}$$

avec Γ_b l'ensemble des bords libres. Sur un maillage conforme, cette équation est satisfaite seulement si le maillage est assez fin, et le plus souvent approximativement. Nous regardons donc différents rayons du cercle, et sur chaque chemin nous nous intéressons à $n_x\sigma_x + n_y\sigma_{xy}$ ainsi qu'à $n_x\sigma_{xy} + n_y\sigma_y$, ces deux valeurs doivent s'annuler sur le bord. Sur la figure 3.24, les plans de coupe sur lesquels les contraintes sont affichées sont montrés.

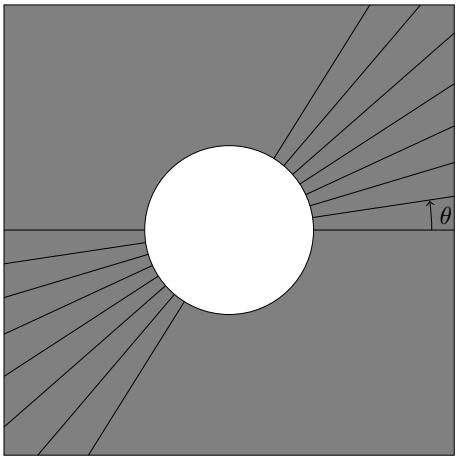
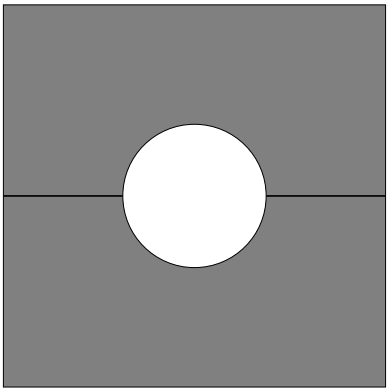


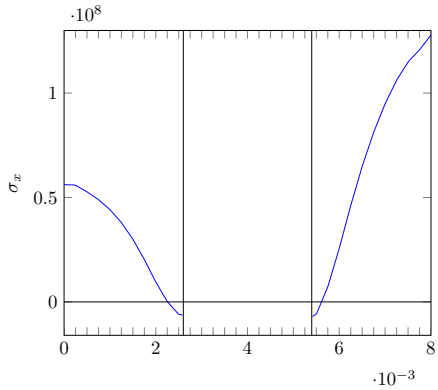
FIGURE 3.24 – Les différents rayons pris pour vérifier les contraintes

Les figures 3.25 et 3.26 montrent les tracés des contraintes sur deux coupes différentes. Sur chaque figure la coupe représentée est montrée en haut à gauche, les contraintes sur trente-deux par trente-deux éléments en haut à droite, et en bas sur soixante-quatre par soixante-quatre éléments et cent vingt-huit par cent vingt-huit éléments respectivement à gauche et à droite.

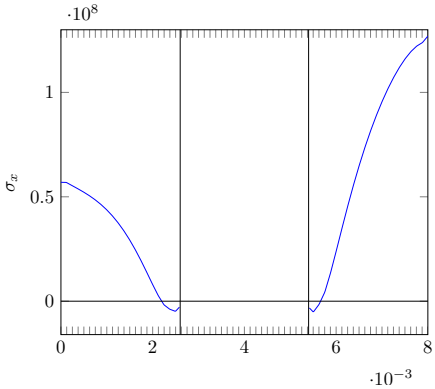
Pour chaque coupe, les contraintes tendent bien vers zéro. De plus, en augmentant le nombre d'éléments par dimension les contraintes aux bords libres sont de plus en plus proches de zéro.



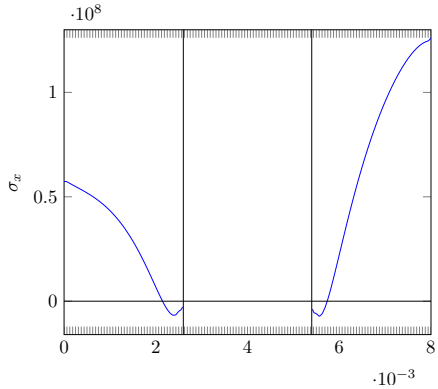
(a) Coupe 1



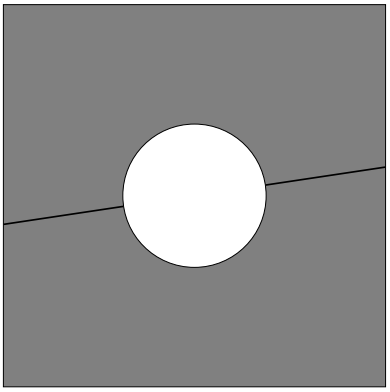
(b) 32 éléments - Coupe 1



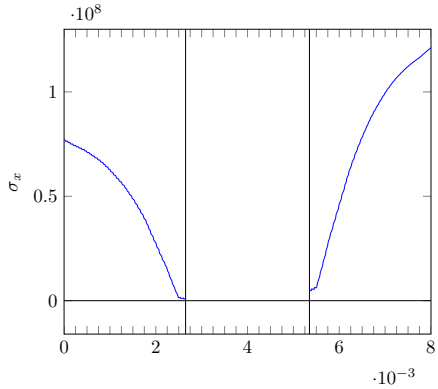
(c) 64 éléments - Coupe 1



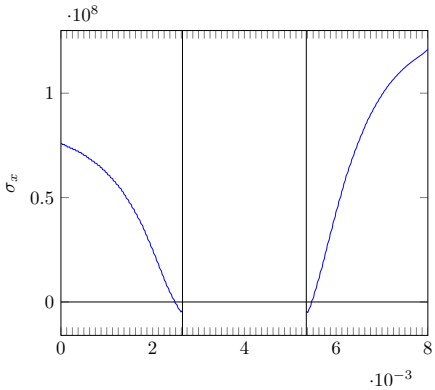
(d) 128 éléments - Coupe 1



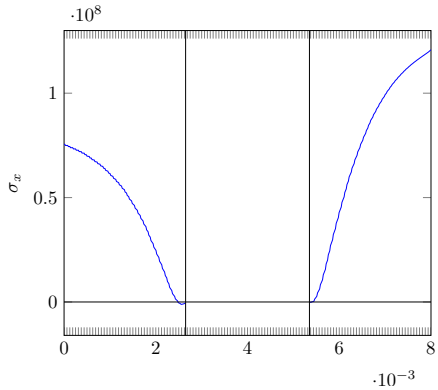
(e) Coupe 2



(f) 32 éléments - Coupe 2

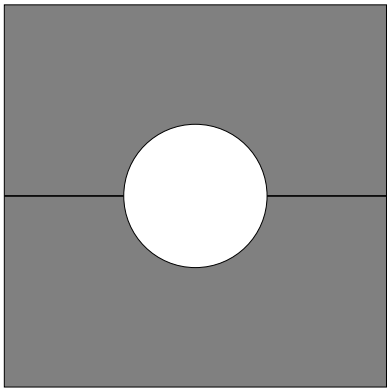


(g) 64 éléments - Coupe 2

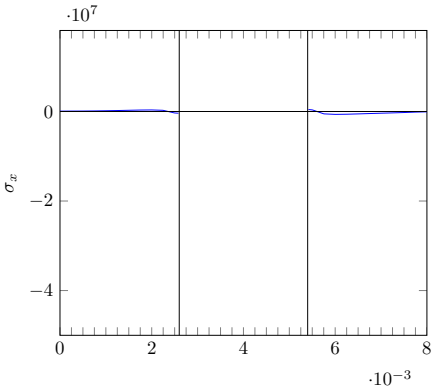


(h) 128 éléments - Coupe 2

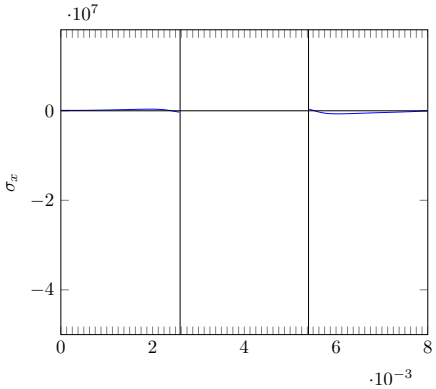
FIGURE 3.25 – Tracés de $n_x\sigma_x + n_y\sigma_{xy}$, sur deux coupes différentes et pour 32, 64 et 128 éléments



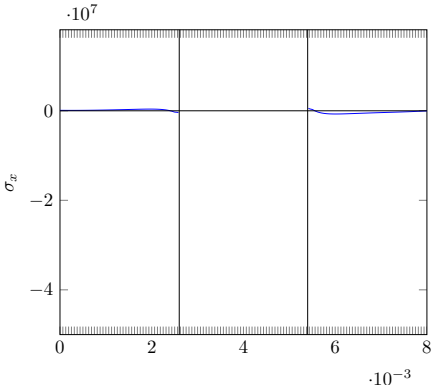
(a) Coupe 1



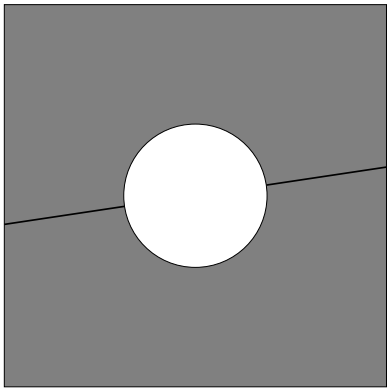
(b) 32 éléments - Coupe 1



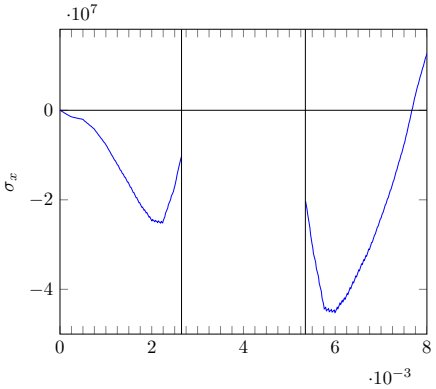
(c) 64 éléments - Coupe 1



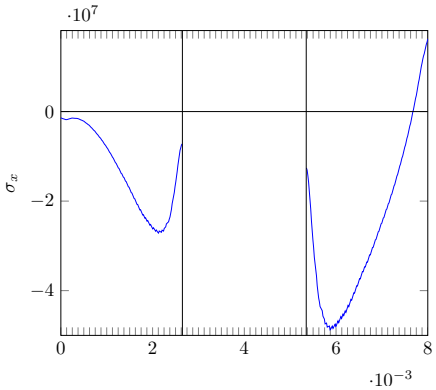
(d) 128 éléments - Coupe 1



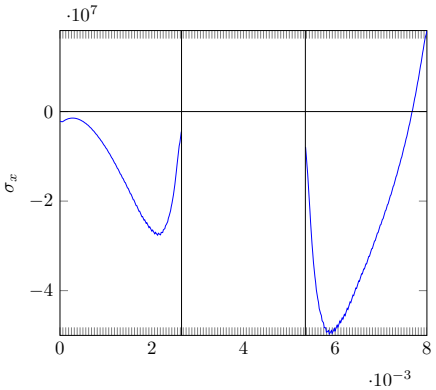
(e) Coupe 2



(f) 32 éléments - Coupe 2



(g) 64 éléments - Coupe 2

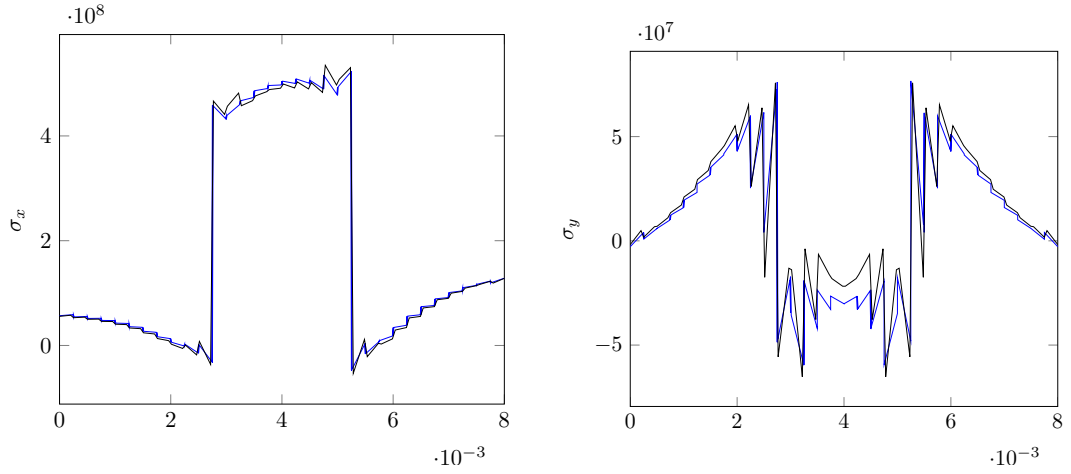


(h) 128 éléments - Coupe 2

FIGURE 3.26 – Tracés de $n_x\sigma_{xy} + n_y\sigma_y$, sur deux coupes différentes et pour 32, 64 et 128 éléments

3.7.1 Comparaison des résultats en contraintes avec la méthode étendus

Afin d'avoir une comparaison possible des contraintes calculées aux points d'intégration, la méthode des éléments finis étendus a été appliquée sur le modèle de la plaque trouée. La méthode des éléments finis étendus prend comme points d'intégration deux points de quadrature de Gauss par dimension. Notre méthode a comme points d'intégration, cinq points d'intégration pris sur une grille régulière. La figure 3.27 montre les contraintes aux points d'intégration des deux méthodes. La continuité de la contrainte sur les éléments ainsi que le saut entre deux éléments est bien visible.



(a) Contraintes en x (b) Contraintes en y

FIGURE 3.27 – Comparaison des résultats en contraintes aux points d'intégration sur un coupe du modèle, en bleu la méthode des éléments finis étendus, en noir la méthode de la thèse

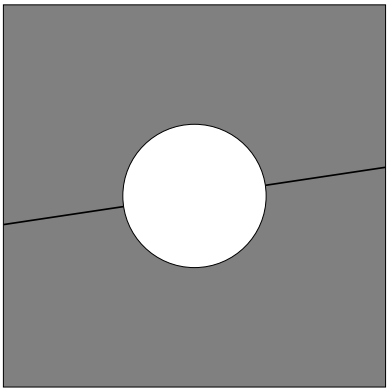
Une comparaison avec la méthode des éléments finis étendus permet de confirmer que les contraintes obtenues en utilisant notre méthode sont correctes. En effet, la figure 3.27 montrent que les deux courbes de contraintes ont même allure, pour les éléments à l'intérieur de la matière. En conclusion, les résultats en contraintes calculées point par point sont bons, proches de ceux obtenus avec la méthode des éléments finis étendus.

3.7.2 Résultats en contraintes avec les « extra-shapes »

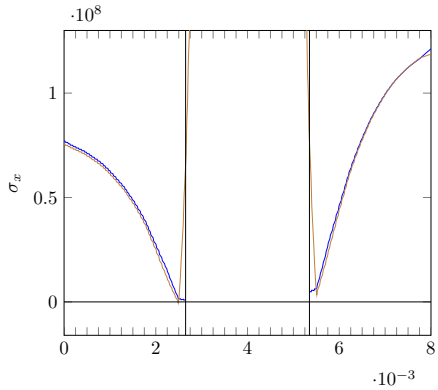
L'avantage des « extra-shapes » est qu'ils permettent un prolongement de la contrainte sur les éléments coupés. La contrainte a donc été calculée partout en faisant le produit DBu. Les figures 3.28 et 3.29 affichent de nouveau les contraintes aux nœuds selon deux coupes du modèle.

Il est intéressant de noter que le prolongement de la contrainte a bien lieu, et donne des bons résultats lors du calcul de $n_x\sigma_{xy} + n_y\sigma_y$. Néanmoins, un saut a lieu lors du calcul de $n_x\sigma_x + n_y\sigma_{xy}$ le long de la coupe 2.

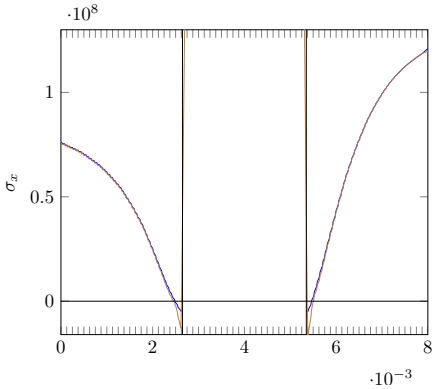
Une nouvelle méthode éléments finis ainsi qu'un exemple de cette méthode sur un modèle en deux dimensions viennent d'être montrés. Le prochain chapitre porte toujours sur cette méthode mais introduit une nouvelle façon d'intégrer les matrices élémentaires : l'apprentissage.



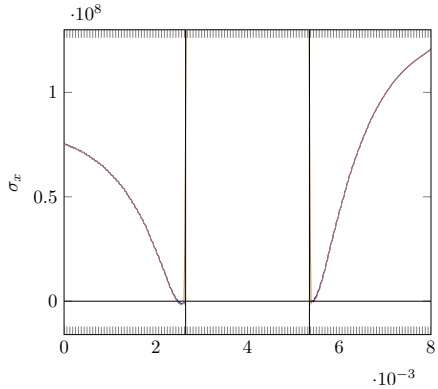
(a) Coupe 2



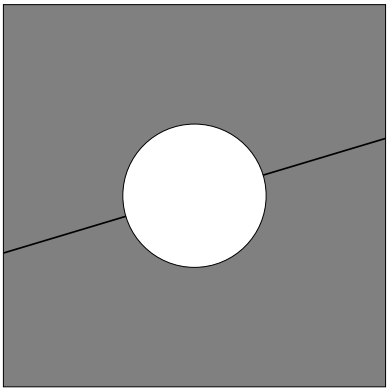
(b) 32 éléments



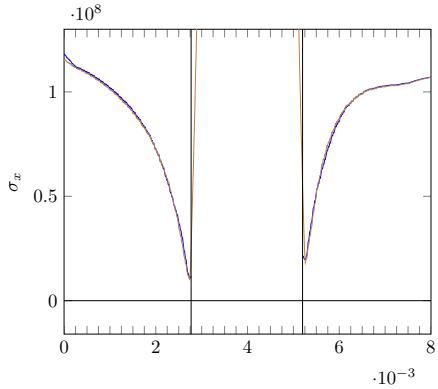
(c) 64 éléments



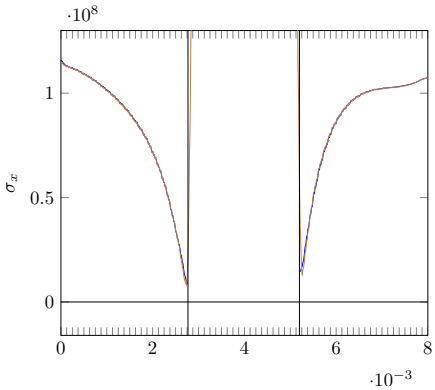
(d) 128 éléments



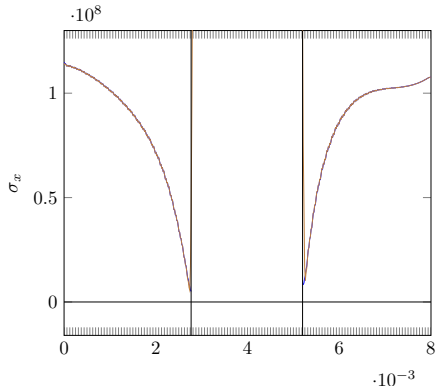
(e) Coupe 3



(f) 32 éléments

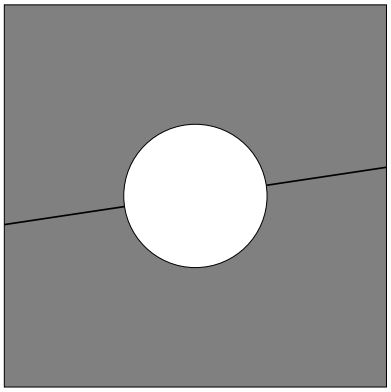


(g) 64 éléments

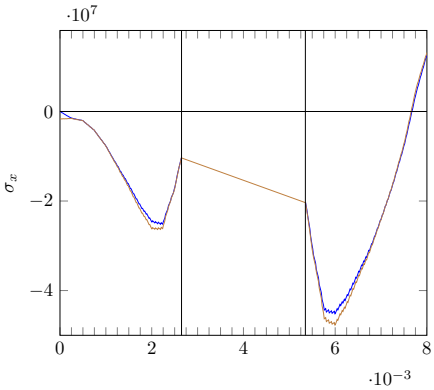


(h) 128 éléments

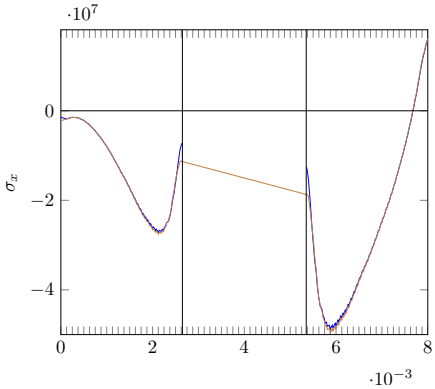
FIGURE 3.28 – Tracés de $n_x\sigma_x + n_y\sigma_{xy}$, sur deux coupes différentes et pour 32, 64 et 128 éléments, en bleu la méthode de la thèse sans « extra-shapes », en marron la méthode de la thèse avec « extra-shapes »



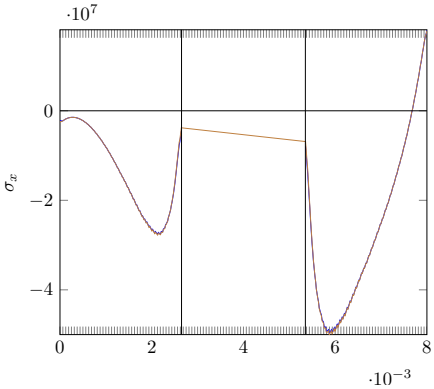
(a) Coupe 2



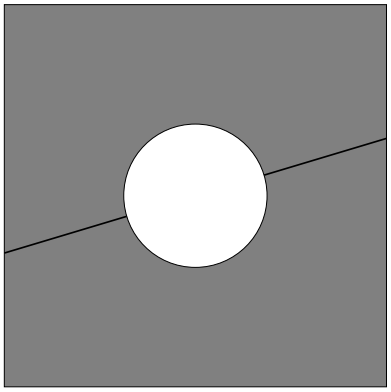
(b) 32 éléments



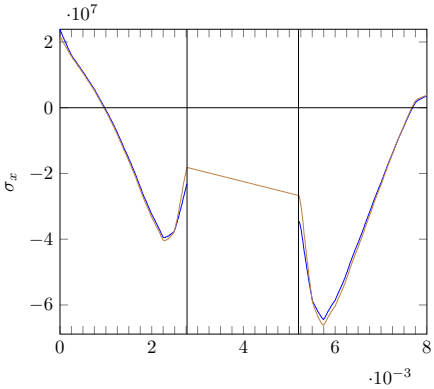
(c) 64 éléments



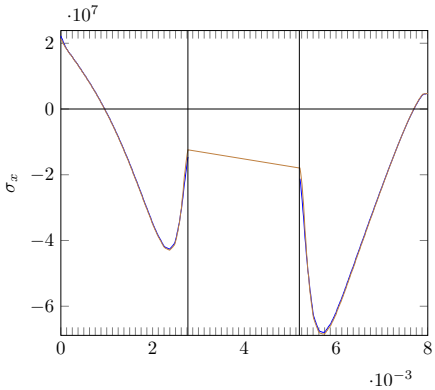
(d) 128 éléments



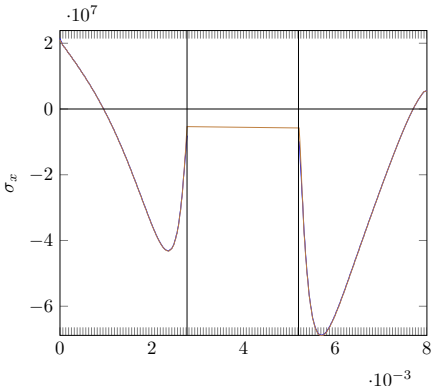
(e) Coupe 3



(f) 32 éléments



(g) 64 éléments



(h) 128 éléments

FIGURE 3.29 – Tracés de $n_x\sigma_{xy} + n_y\sigma_y$, sur deux coupes différentes et pour 32, 64 et 128 éléments, en bleu la méthode de la thèse sans « extra-shapes », en marron la méthode de la thèse avec « extra-shapes »

Chapitre 4

Apprentissage des matrices de raideur

Afin d'accélérer la méthode d'intégration précédemment proposée, nous proposons d'introduire une méthode d'apprentissage. À ce stade, le temps de calcul par élément dépent du nombre de points d'intégration choisi. Avec notre méthode, plus l'erreur d'approximation de la matrice est faible, plus le nombre de points d'intégration va augmenter, plus le nombre de matrices à stocker va être important, et le temps de calcul aussi. Rien qu'avec cinq points d'intégration par dimension, en passant à trois dimensions, cent vingt cinq matrices vont devoir être stockées.

Nous proposons pour parer cela d'utiliser une méthode d'apprentissage de la matrice de raideur élémentaire à partir des valeurs de la surface de niveaux aux quatre sommets. Ainsi peu importe le temps passé à faire l'apprentissage, une fois les matrices de raideur bien apprises, le temps de reconstruction sera moindre. L'apprentissage mis en place est une régression quadratique. L'idée de l'apprentissage est de trouver une fonction, dans notre cas un polynôme de degré deux, telle que n'importe quelle matrice de raideur puisse être retrouvée en fonction des densités.

4.1 Description de la méthode

En premier lieu, un ensemble de m éléments coupés est généré aléatoirement, représentés par leur quatre valeurs de la fonction surface de niveau aux nœuds $(d_1^j, d_2^j, d_3^j, d_4^j)$ pour $j \in \{1, \ldots, m\}$. Les matrices de raideur élémentaires $K_e^j, j = 1, \ldots, m$ sont calculées sur une grille de points d'intégration fine.

L'apprentissage est décomposé en deux étapes détaillées plus bas. La première étape consiste à trouver d'une base de représentation des matrices élémentaires, un espace $\mathcal{W}$ de petite dimension p qui engendre bien les matrice $K_e^j, j = \{1, \ldots, m\}$. Cet espace est décrit par une base $(W_1, \ldots, W_p)$.

La deuxième étape consiste à trouver un polynôme Π de degré deux tel que :

$$\Pi(d_1, d_2, d_3, d_4) = (c_1, \ldots, c_p), \tag{4.1}$$

avec c_i approchant les coefficients des matrices de raideur dans la base.

Etape 1 : Construction de la base

La base $(W_1, \ldots, W_p)$ doit être assez petite afin qu'un minimum de coefficients soient à stocker, et assez grande afin d'assurer une erreur basse. Il est aussi important de conserver les bonnes propriétés des matrices de raideur, leur symétrie. On cherche une base de matrices symétriques, la somme de matrices symétriques sera bien une matrice symétrique.

Nous nous sommes intéressés à la décomposition en valeurs singulières de l'ensemble des matrices. Une matrice Υ contenant toutes les données à apprendre est formée. Les vecteurs orthonormés obtenu forment une bonne base de décomposition. De plus, les vecteurs U obtenus par cette décomposition, remis en matrices ont l'avantage de bien avoir la propriété de symétrie souhaitée. L'erreur faite lors de la décomposition dans la base est connue a priori.

Algorithme 1 : Construction de la base

Fonction `GetBase`
- **input :** d_n : Densités aux nœuds
- **output :** p : Le nombre de vecteurs de la base
  W_i : Les vecteurs de base
- **pour tous les** *éléments* (e) **faire**
  - Calcul de la matrice de raideur élémentaire associée à (e) en fonction des d_n : $K_e(d_i)$;
- **fin**
- Construction de la matrice comprenant l'ensemble des matrices $K_e(d_n)$: Υ ;
- Décomposition en valeurs singulières de Υ : U, S, V;
- Sélection des p premiers vecteurs de V : W_i;

end

Etape 2 : Approximation par un polynôme de degré deux

Soit $(W_i)_{i=1,\ldots,p}$ la base de l'espace $\mathcal{W}$ dans lequel nous allons approximer nos matrices de raideur. Les matrices K_e^j s'écrivent :

$$K_e^j = \sum_{i=1}^{p} c_{i,j} W_i + R^j, \tag{4.2}$$

où le résidu $R^j \in \mathcal{W}^{\perp}$.

Au cours de l'apprentissage, un polynôme de degré deux qui approxime au mieux les coefficients de la base pour l'ensemble des éléments est cherché :

$$\Pi(d_1^j, d_2^j, d_3^j, d_4^j) \approx (c_1^j, \ldots, c_p^j). \tag{4.3}$$

Le sens de $\approx$ s'entendant au sens d'une minimisation quadratique. En pratique, nous effectuons cette régression quadratique de façon matricielle. Soient deux nouvelles matrices Δ (4.4) et C (4.5) telles que :

$$\Delta = \begin{pmatrix} 1 & d_1^1 & d_2^1 & d_3^1 & d_4^1 & {d_1^1}^2 & \ldots & d_3^1 d_4^1 \\ 1 & d_1^2 & d_2^2 & d_3^2 & d_4^2 & {d_1^2}^2 & \ldots & d_3^2 d_4^2 \\ & & \vdots & & & & & \\ 1 & d_1^m & d_2^m & d_3^m & d_4^m & {d_1^m}^2 & \ldots & d_3^m d_4^m \end{pmatrix}, \tag{4.4}$$

avec m le nombre d'éléments appris. La matrice Δ est de taille (m,15), quinze correspond au nombre de termes de la régression quadratique.

$$C = \begin{pmatrix} c_1^1 & c_2^1 & \ldots & c_p^1 \\ c_1^2 & c_2^2 & \ldots & c_p^2 \\ & \vdots & & \\ c_1^m & c_2^m & \ldots & c_p^m \end{pmatrix}. \tag{4.5}$$

Ainsi il ne reste plus qu'à résoudre le système matriciel $\Delta A = C$ au sens des moindres carrés pour obtenir la matrice d'apprentissage. L'algorithme 2 résume les différentes étapes.

Algorithme 2 : Pseudo code de l'apprentissage

Fonction `Learning`

input : p : Le nombre de vecteurs de la base
W_i : Les vecteurs de base
d_n : Densités aux nœuds

output : A : Matrice d'apprentissage

pour tous les *éléments* (e) **faire**

Calcul de la matrice de raideur élémentaire associée à (e) en fonction des d_n : $K_e^j(d_n^j)$;

Trouver les coefficients c_i tels que $\sum_i^p c_{i,j} W_i \simeq K_e^j(d_n^j)$;

fin

Calcul de la matrice des termes quadratiques des densités d^j : Δ;

Calcul de la matrice d'apprentissage (A) par moindres carrés telle que $\Delta \times A = C$;

end

À la fin de l'apprentissage, la matrice A ainsi obtenue est de taille $(15, p)$, quinze pour le nombre de termes de la régression quadratique. Pour calculer une matrice de raideur, il suffit de multiplier notre matrice d'apprentissage par les densités de l'élément afin de récupérer les coefficients $c_{i,j}$, puis d'appliquer l'équation (4.2).

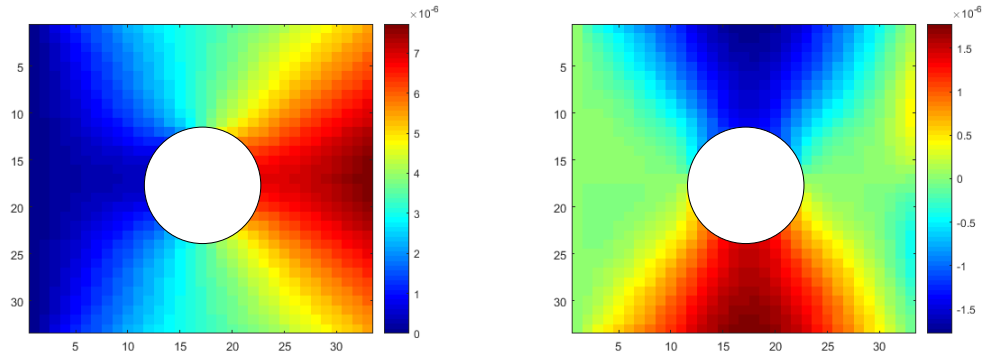
Dans la prochaine section, les résultats obtenus après apprentissage sont présentés.

4.2 Exemple numérique d'apprentissage

Dans l'exemple de la plaque trouée montré dans la section 3.7.2, les éléments à retrouver par apprentissage sont les éléments au niveau du trou. En tout, 17 000 éléments coupés sont appris, avec une intégration de 100×100 points d'intégration. Pour 32×32 éléments, 44 éléments sont coupés et donc à restituer par apprentissage, pour 64×64 éléments, 92 éléments sont coupés.

En comparant les matrices avant et après apprentissage, une erreur relative en norme euclidienne de l'ordre de 10^{-4} est obtenue pour toutes les matrices.

Les figures 4.1 et 4.2 représentent les déplacements en x et y obtenus après simulation avec 32×32 éléments et 64×64 éléments.

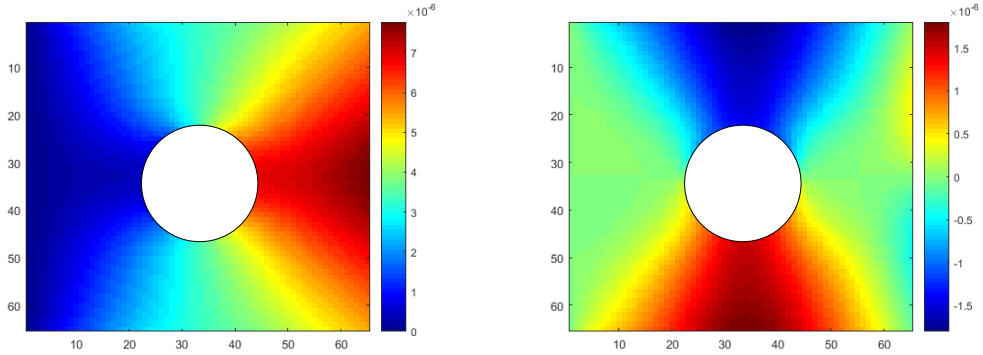


(a) en x (b) en y

FIGURE 4.1 – Déplacements obtenus après apprentissage des matrices au bord, 32×32 éléments

La maximum obtenu avec trente-deux éléments par dimension en x est de 7.725×10^{-6} et en y de 1.778×10^{-6}. L'erreur obtenue sur les déplacements en comparant avec les résultats obtenus par notre méthode sans « extra-shapes » est de l'ordre de 10^{-3} en norme deux. Cette erreur décroît lors de l'ajout d'éléments dans la matière.

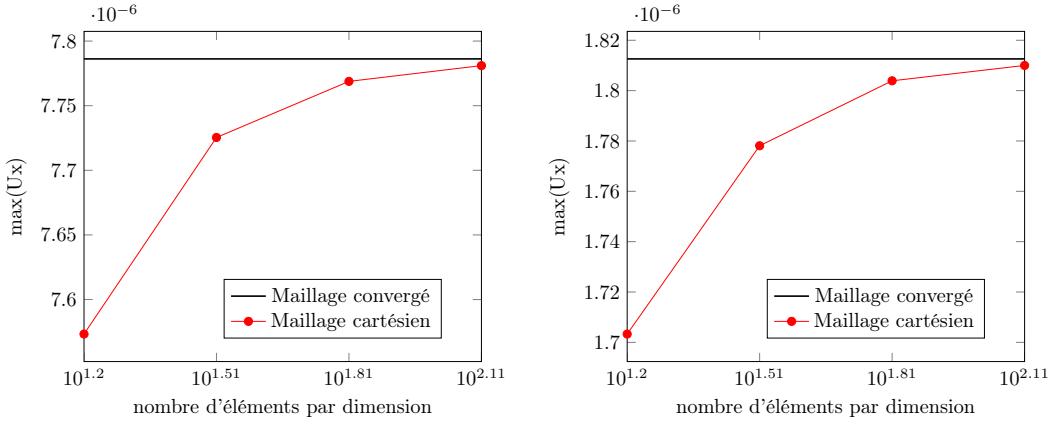
La maximum obtenu pour une simulation 64×64 éléments en x est de 7.769×10^{-6} et en y de 1.804×10^{-6}. L'erreur obtenue est de l'ordre de 10^{-4}. Les maxima obtenus sont légèrement inférieurs à ceux obtenus sans apprentissage, néanmoins une convergence



(a) en x (b) en y

FIGURE 4.2 – Déplacements obtenus après apprentissage des matrices au bord, 64×64 éléments

a toujours lieu. La figure 4.3 illustre cette convergence lorsque le nombre d'éléments par dimension augmente.



(a) en x (b) en y

FIGURE 4.3 – Convergence du maximum des déplacements obtenus après apprentissage des matrices au bord

4.2.1 Les résultats en contraintes

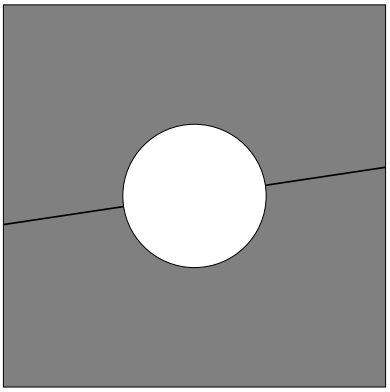
Les résultats en contraintes sont de nouveau montrés selon deux plans de coupe du modèle, afin de s'intéresser aux résultats aux bords libres. Les figures 4.4 et 4.5 affichent les tracés selon deux coupes de $n_x\sigma_x + n_y\sigma_{xy}$ et de $n_y\sigma_y + n_x\sigma_{xy}$. Sur les figures, à la fois

les contraintes de notre méthode avant apprentissage et après apprentissage sont montrées pour comparaison.

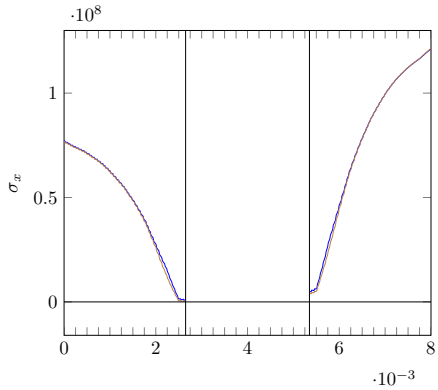
Les figures 4.4 et 4.5 montrent que les contraintes au bord libre tendent bien toujours vers zéro. Bien que les matrices élémentaires apprises se situent au niveau des bords libres, l'apprentissage de ces matrices n'impactent pas sur les résultats en contraintes.

4.2.2 Conclusion

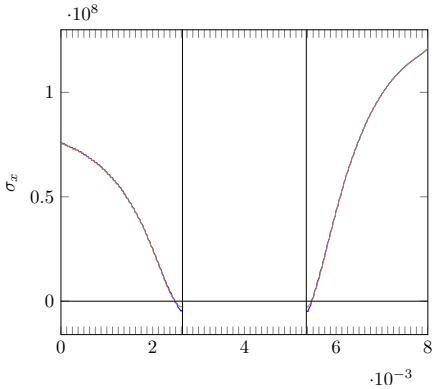
Ce chapitre vient de présenter une nouvelle intégration pour apprendre les matrices élémentaires des éléments coupés, ainsi que des premiers résultats encourageants. Le prochain chapitre a un nouvel objectif, celui de compresser les résultats obtenus par une méthode des éléments finis.



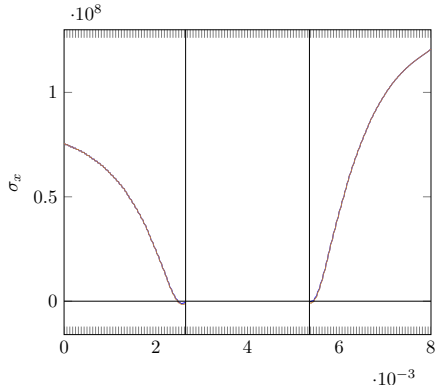
(a) Coupe 2



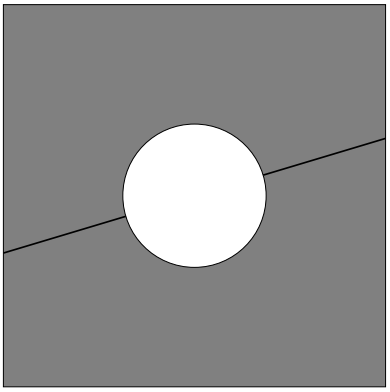
(b) 32 éléments



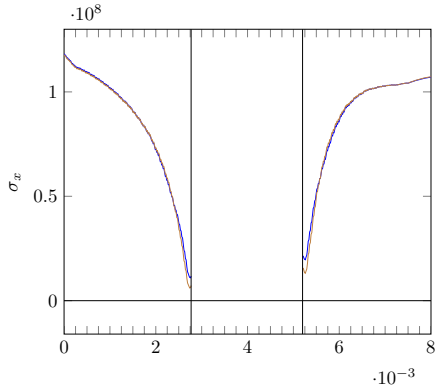
(c) 64 éléments



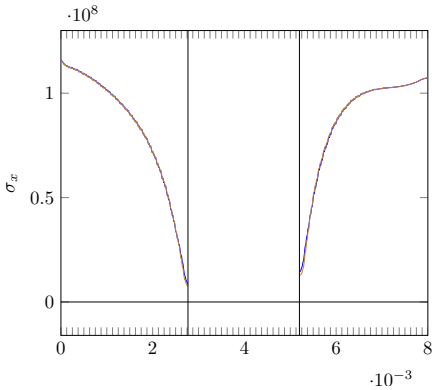
(d) 128 éléments



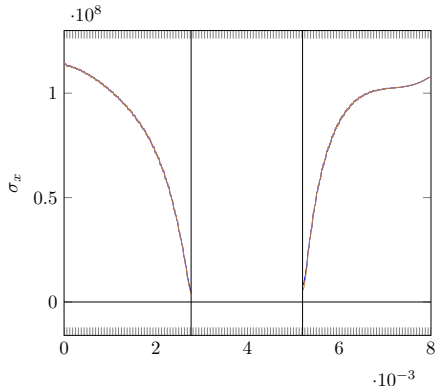
(e) Coupe 3



(f) 32 éléments

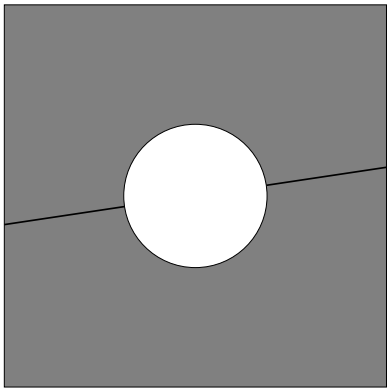


(g) 64 éléments

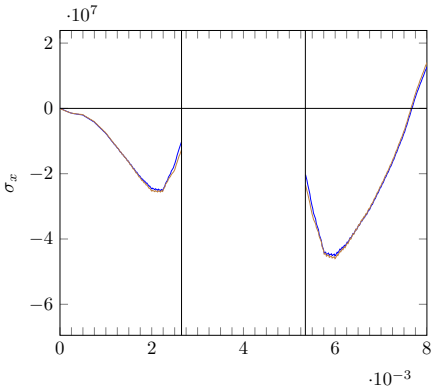


(h) 128 éléments

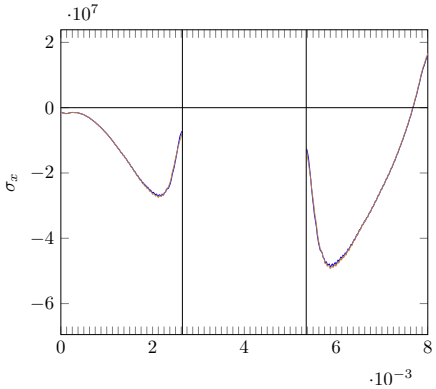
FIGURE 4.4 – Tracés de $n_x\sigma_x + n_y\sigma_{xy}$, sur deux coupes différentes et pour 32, 64 et 128 éléments après apprentissage », en bleu avant apprentissage, en marron après



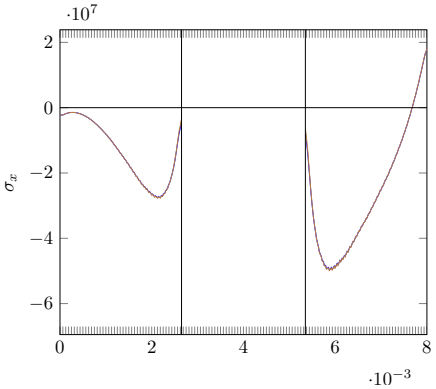
(a) Coupe 2



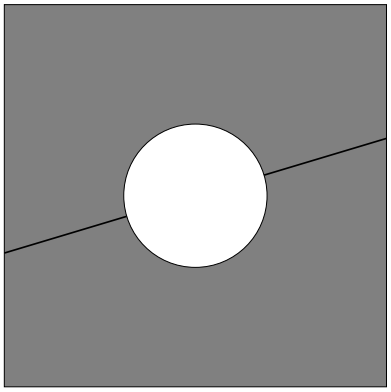
(b) 32 éléments



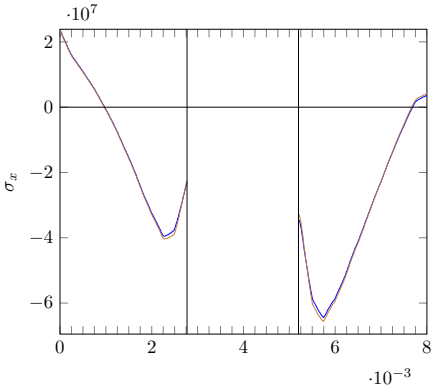
(c) 64 éléments



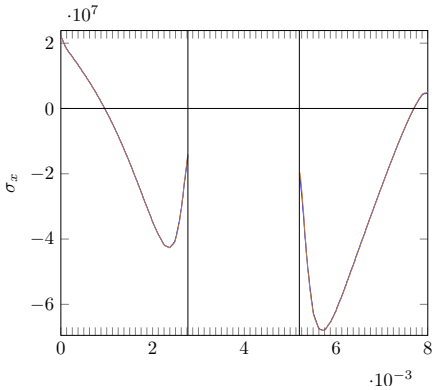
(d) 128 éléments



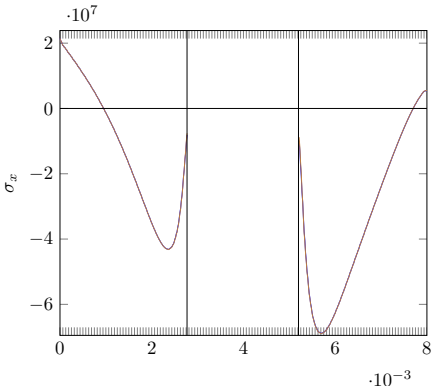
(e) Coupe 3



(f) 32 éléments



(g) 64 éléments



(h) 128 éléments

FIGURE 4.5 – Tracés de $n_x\sigma_{xy} + n_y\sigma_y$, sur deux coupes différentes et pour 32, 64 et 128 éléments après apprentissage, en bleu avant apprentissage, en marron après

Chapitre 5

Gain de place mémoire par compression

5.1 Motivation

Les résultats obtenus après analyse par la méthode des éléments finis standard ou par la méthode des éléments finis sur maillage cartésien sont des tableaux de deux, trois ou quatre dimensions dont la quatrième dimension est le temps. Ces résultats ont une taille qui peut croître énormément selon la complexité des modèles, ainsi que la précision des schémas de résolution. Dans un contexte de programmation sur processeurs graphiques, où la mémoire est limitée, il est intéressant d'arriver à compresser ces données. Nous nous sommes intéressés à la compression des modèles et des résultats éléments finis par la transformée en ondelettes. La compression mise en place aidera aussi pour les problèmes de stockage en réduisant la taille des fichiers générés, et pour la visualisation des données.

Pour faire cette compression, la transformée en ondelettes est utilisée. La compression mise en place grâce à cette transformée se fait en deux étapes. En premier la transformée en ondelettes est appliquée, elle permet de représenter la donnée avec un petit nombre de coefficients. Après application de la transformée, la donnée a moins de valeurs différentes mais elle est toujours de la même taille et chacune de ses valeurs prend la même place qu'avant. C'est pourquoi, après transformation, aucune information n'a été perdue, mais la donnée initiale n'a pas non plus été compressée. La deuxième étape consiste à quantifier la donnée, à passer ses valeurs réelles en valeurs entières. C'est lors de la quantification que la compression va avoir lieu. Cette seconde étape introduit des pertes.

Pour les données $3D$ ayant une dimension temporelle, un autre traitement est appliqué pour compresser cette quatrième dimension.

Après avoir présenté les filtres et les bancs de filtres, une deuxième partie introduira la transformée en ondelettes discrète. La partie 5.4 présentera la compression stationnaire ainsi que la compression appliquée en $3D + t$. La partie 5.5 donnera des résultats de ces

compressions.

5.2 Filtres linéaires et bancs de filtres

En traitement du signal numérique, un filtre linéaire est une fonction permettant de calculer un signal discret à partir d'un autre signal discret. Chaque filtre linéaire a sa particularité, de retirer, ou d'accentuer certaines parties des fréquences d'un signal. Les filtres passe-haut sont des filtres qui atténuent les fréquences en dessous d'une fréquence de coupure donnée. Les filtres passe-bas atténuent les fréquences au dessus d'une fréquence donnée. Les filtres passe-bande ne laisse passer qu'une bande de fréquences comprises entre deux fréquences de coupure donnée.

Les bancs de filtres consistent à mettre en cascade plusieurs filtres.

La convolution de filtres discrets u et v de taille M et N respectivement, est donnée par l'équation (5.1) :

$$(u \star v)(n) = \sum_{m=1}^{M} u(n-m)v(m) \tag{5.1}$$

5.3 La transformée en ondelettes en dimension une

La transformée en ondelettes a été introduite pour palier aux limites de la transformée de Fourier en compression de données. La transformée de Fourier a été la première technique permettant la décomposition et la recomposition sans perte d'un signal.

Le principe d'incertitude d'Heisenberg énonce le fait que les domaines temporel et fréquentiel sont complémentaires [BHL07] [KOU15]. Les éléments de la base de Fourier sont très bien localisés dans le domaine fréquentiel mais pas dans le domaine temporel. En effet, les sinusoïdes sont des fonctions associées à un certain nombre de fréquences (un nombre fini) mais qui n'ont ni début, ni fin (domaine temporel infini). La transformée en ondelettes cherche un meilleur compromis afin d'obtenir des informations à la fois temporelles et fréquentielles. Le principe est de décomposer un signal en une famille de fonctions localisées en temps et en fréquence appelées ondelettes.

Les ondelettes ont été introduites dans les années 70 par Jean Morlet [GM84], qui eut l'idée d'utiliser des fonctions de différentes fenêtres pour l'analyse de différentes bandes de fréquence. De nombreux travaux cherchant à obtenir une base permettant une bonne localisation fréquentielle et temporelle existent. Néanmoins, le terme ondelettes et la base associée ont été introduits par A. Grossmann et J. Morlet en 1984 [GM84]. En 1985, Yves Meyer a construit une base d'ondelettes orthogonales mieux localisées en fréquence et en temps [MEY85],[LM86]. Ingrid Daubechies introduisit les ondelettes à support compact, ainsi que la transition de l'analyse continue à l'analyse discréte. Stéphane Mallat a développé des algorithmes de décomposition rapide, outil indispensable au traitement

des signaux en temps réel [MAL08]. En 1992, Albert Cohen, Jean Feauveau et Ingrid Daubechies ont construit des ondelettes biorthogonales à support compact [CDF92].

La transformation en ondelettes permet d'analyser un signal en conservant les caractéristiques de temps et de fréquence. Une famille d'ondelettes est donnée par translations et dilatations d'une fonction mère (5.2) :

$$\Psi_{a,b}(t) = \frac{1}{\sqrt{s}}\Psi\left(\frac{t-b}{a}\right). \tag{5.2}$$

La transformée en ondelettes continue d'un signal $f(t)$ est donnée par l'équation suivante (5.3) :

$$W_f(a,b) = \frac{1}{\sqrt{a}}\int_{-\infty}^{+\infty} f(t)\Psi\left(\frac{t-b}{a}\right)dt, \tag{5.3}$$

avec a le facteur d'échelle et b le paramètre de translation. Pour la compression, seule la transformée en ondelettes discrète servira. La prochaine section explique cette transformée.

5.3.1 Transformée en ondelettes discrète

Soit $u \in \mathbb{R}^N$ le signal, h_0 un filtre passe-bas et h_1 un filtre passe-haut définis sur $\{0,\ldots,N-1\}$, la transformée en ondelettes rapide s'écrit :

$$a_{1,k} = \sum_{l=0}^{N-1} h_0[l-2k]u[l]\ ,\ d_{1,k} = \sum_{l=0}^{N-1} h_1[l-2k]u[l]\ \ \textit{pour}\ k \in \left\{0,\ldots,\frac{N}{2}-1\right\}, \tag{5.4}$$

où a représente l'approximation du signal et d les détails.

Les ondelettes discrètes sont un cas particulier des bancs de filtres. C'est un découpage en deux sous-bandes d'un signal. Cette décomposition se fait à l'aide de deux filtres, un passe-bas et un passe haut, afin de créer deux signaux représentatifs l'un des basses fréquences et l'autre des hautes. L'allure générale d'un signal est donnée par les fréquences les plus basses et les détails par les fréquences les plus élevées.

Cette décomposition est réversible grâce à des filtres de synsthèse g_0 et g_1 bien choisis. La transformation en ondelettes inverse est donnée par :

$$u_k = \sum_{l=0}^{N-1} g_0[k-2l]a_1[l] + g_1[k-2l]d_1[l]\ \ \textit{pour}\ k \in \{0,\ldots,N\}. \tag{5.5}$$

Afin que le signal initial soit retrouvé sans erreur, il faut que la décomposition et la recomposition soient inverses l'une de l'autre. Pour cela, les filtres h_0, h_1, g_0 et g_1 doivent satisfaire les conditions des bancs de filtres à reconstruction exacte [MC92]. Les filtres g_0 et g_1 sont les filtres miroirs respectifs de h_0 et h_1.

Lors d'une décomposition en ondelettes, le signal contenant les basses fréquences du signal initial est de nouveau décomposé : la transformée en ondelettes est appliquée au

sous-signal a_1. La transformée en ondelettes peut s'appliquer à différents niveaux appelés niveau de décomposition du signal. Le signal est décomposé au niveau un si la transformée est appliquée une fois, au niveau deux si elle est appliquée deux fois ... C'est pourquoi, tout banc de filtres à reconstruction parfaite ne correspond pas à un système d'ondelettes. En effet, il faut faire attention à la stabilité des schémas de décomposition et de recomposition lorsque plusieurs bancs de filtres sont mis en cascade, lorsque la transformée est de nouveau appliquée aux signaux obtenus.

Il existe de nombreuses formes d'ondelettes, le choix de l'ondelette optimale dépend de l'application souhaitée. Pour la compression, il faut trouver la base d'ondelettes dans laquelle le signal s'exprimera avec le moins d'information possible. Des familles d'ondelettes qui permettent une bonne compression sont les ondelettes biorthogonales symétriques CDF. Les initiales CDF viennent du nom de trois scientifiques Cohen, Daubechies, Feauveau qui ont une place importante dans l'analyse par ondelettes et dans la découverte de ces ondelettes [CDF92]. La norme JPEG 2000 se sert de ce type d'ondelettes pour la compression d'images [RJ02]. Dans le cadre de la thèse, la compression est faite avec les filtres CDF 5/3 (5 et 3 étant le nombre de coefficients non nuls des filtres d'analyse et de synthèse). Leurs coefficients sont montrés dans le tableau 5.1 donnés par [PAU06].

	Coefficients des filtres d'analyse		Coefficients des filtres de synthèse	
i	Filtre passe-bas	Filtre passe-haut	Filtre passe-bas	Filtre passe-haut
0	$\frac{3}{4}\sqrt{2}$	$-\frac{1}{2}\sqrt{2}$	$\frac{1}{2}\sqrt{2}$	$\frac{-3}{4}\sqrt{2}$
± 1	$\frac{1}{4}\sqrt{2}$	$\frac{1}{4}\sqrt{2}$	$\frac{1}{4}\sqrt{2}$	$\frac{1}{4}\sqrt{2}$
± 2	$\frac{-1}{8}\sqrt{2}$			$\frac{1}{8}\sqrt{2}$

Figure 5.1 – Tables des coefficients des filtres CDF 5/3

Au niveau deux, il faut garder trois signaux pour pouvoir recomposer le signal : les détails du niveau un, l'approximation et les détails du niveau deux. Ce qui vient d'être présenté est la transformée en ondelettes d'un signal, qui est utilisée pour la transformée en ondelettes d'une donnée de dimension supérieure.

5.3.2 Généralisation aux dimensions supérieures

La transformée en ondelettes 2D est une technique qui est très utilisée en traitement d'images. L'idée est d'utiliser la transformée en ondelettes 1D : la transformation 1D est appliquée à chaque ligne puis à chaque colonne de l'image. Après application de la trans-

formée sur chaque ligne, deux images sont obtenues dont la largeur a été divisée par deux. À ces deux sous-images est appliquée la transformée 1D sur chacune de leurs colonnes. Finalement quatre sous-images en résultent, une contenant l'approximation, une les détails horizontaux, une les détails verticaux et la dernière les détails diagonaux. Le schéma de la figure 5.2 récapitule cette transformation.

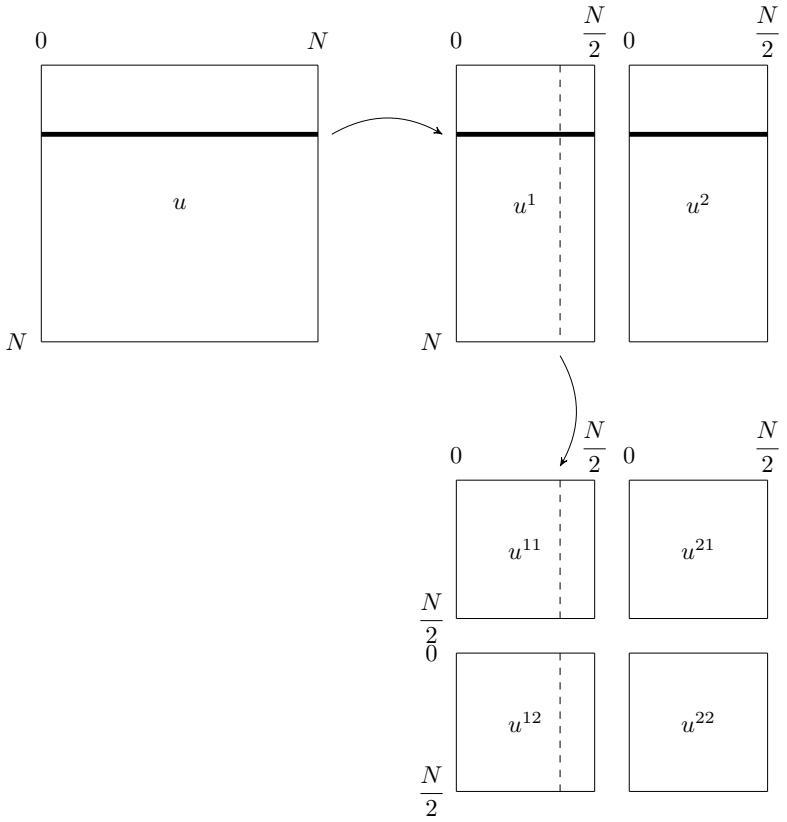


FIGURE 5.2 – La transformée en ondelettes discrète 2D

Sur la figure 5.2, $u \in \mathbb{R}^{N^2}$ représente l'image initiale, u^1 et u^2 sont les images résultant d'une première transformation. Toutes les lignes de u^1 ont été transformées avec le filtre passe-bas h et toutes celles de u_2 avec le filtre passe-haut g. Les images u^{11} et u^{12} sont les résultats de la transformation de u^1, toutes les colonnes de u^{11} ont été transformées avec h et toutes celles de u^{12} avec g. Il en est de même pour u^{21} et u^{22} qui résultent de u^2. Ainsi au niveau un, quatre sous-images résultent de la transformation. Au niveau deux, sept sous-images sont obtenues : les trois images détails du niveau un sont gardées (u^{21},u^{12},u^{22}), et quatre autres images de u^{11} sont obtenues par transformation au niveau deux.

Afin d'illustrer la décomposition en ondelettes 2D à un et deux niveaux, la transformée

a été appliquée sur une vue d'un modèle 3D. Le modèle représente une partie d'un train d'atterrissage d'un avion.

FIGURE 5.3 – Une capture d'une partie d'un train d'atterrissage

La figure 5.5 montre les résultats de la transformée en ondelettes sur l'image de la figure 5.3. Les ondelettes utilisées pour cette transformation sont les ondelettes CDF 5/3.

Comme visible sur la figure 5.5, pour les deux niveaux, la sous-image en haut à gauche représente une approximation de l'image originale. Cette sous-image est le résultat d'une transformation avec le filtre passe-bas dans chacune de ses dimensions.

La transformée en ondelettes d'une donnée de dimension supérieure se fait sur le même principe, c'est à dire en transformant chaque dimension grâce aux ondelettes 1D. Au niveau un de la transformée en ondelettes d'un hypercube 3D, 2^3 sous-hypercubes sont donc obtenus. En effet, pour la transformation d'un hypercube, la transformée en dimension une est d'abord appliquée selon la dimension x, ils en résultent deux hypercubes. La transformée en dimension une est ensuite appliquée selon y, quatre hypercubes sont obtenus. Pour finir, la transformée 1D est appliquée selon z, soient huit hypercubes. Ces huit hypercubes sont visibles sur les figures 5.6a et 5.6b. De même pour un hypercube 4D, ils résultent 2^4 sous-hypercubes de sa transformation au niveau un.

La figure 5.6 permet de voir la transformée 3D aux niveaux un et deux de l'hypercube du modèle dont une coupe a été montrée sur la figure 5.3. Les filtres utilisés pour cette transformée sont les CDF 5/3. De même que pour le cas 2D, un des sous-hypercube est une bonne approximation de l'hypercube initial. Ce premier résultat est intéressant, il permet d'avoir une pré-visualisation qui prendra, dans le cas 3D, huit fois moins de place que la donnée initiale, seize fois moins dans le cas 4D. Une fois la transformée en ondelettes appliquées, la donnée prend toujours autant de place, afin de compresser les données d'autres étapes sont nécessaires. La prochaine section explique notre processus de compression grâce

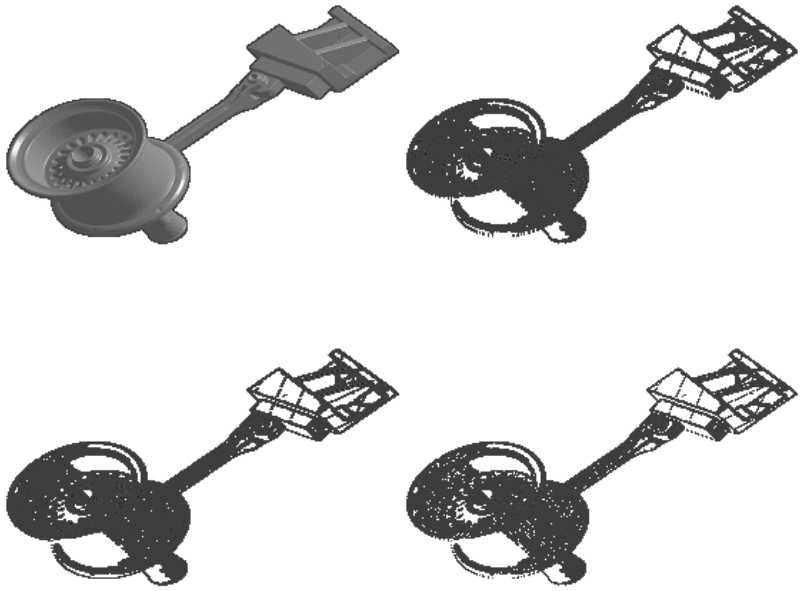


FIGURE 5.4 – Transformée en ondelettes d'une image au niveau un

aux ondelettes.

5.4 Compression par ondelettes 3D

La compression en trois dimensions que nous avons mis en place à trois étapes clefs. Tout d'abord, une tranformée en ondelettes est appliquée à la donnée à compresser. La deuxième étape est un processus de quantification qui permet de représenter un signal dans un ensemble discret d'assez petite taille. Une perte est engendrée lors de la quantification. La dernière étape que nous appliquons est un codage entropique qui permettra une meilleure compression de la donnée. Ce codage est purement numérique.

5.4.1 Quantification

A ce stade du processus de compression, l'hypercube de coefficients d'ondelettes est de même taille que l'hypercube de valeurs initial. La quantification va permettre de diminuer cette taille. Chaque coefficient prend une place de trente-deux bits. Le prochain paragraphe explique comment chaque coefficient va être codé sur moins de bits.

La quantification permet une représentation parcimonieuse des coefficients obtenus avec les ondelettes. Elle permet de passer de valeurs dans un ensemble discret de grande taille

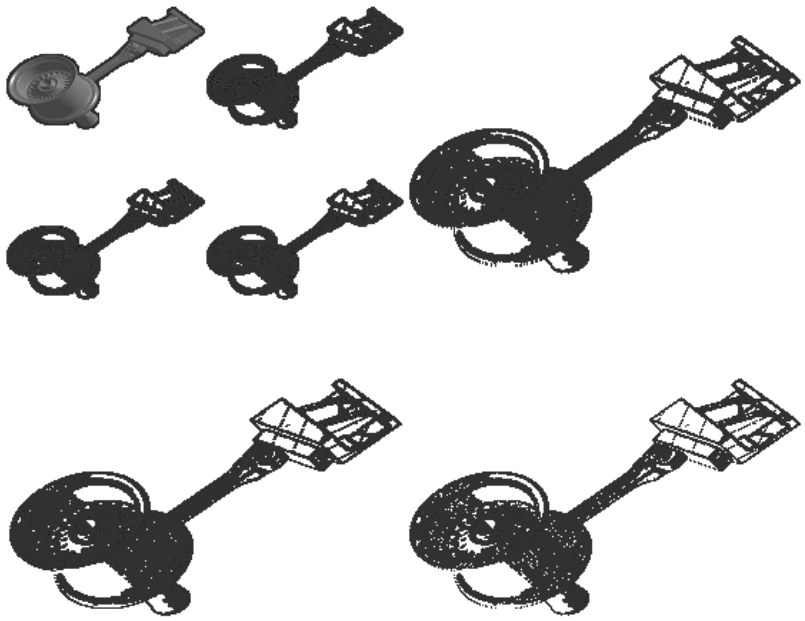


FIGURE 5.5 – Transformée en ondelettes d'une image au niveau deux

à des valeurs dans un ensemble discret d'une taille inférieure. Cette étape introduit une erreur. Avant la quantification, l'erreur maximale autorisée ε est fixée et l'hypercube est découpé en blocs. Par la suite, le travail ne se fera plus sur la donnée totale mais sur un bloc de taille inférieure.

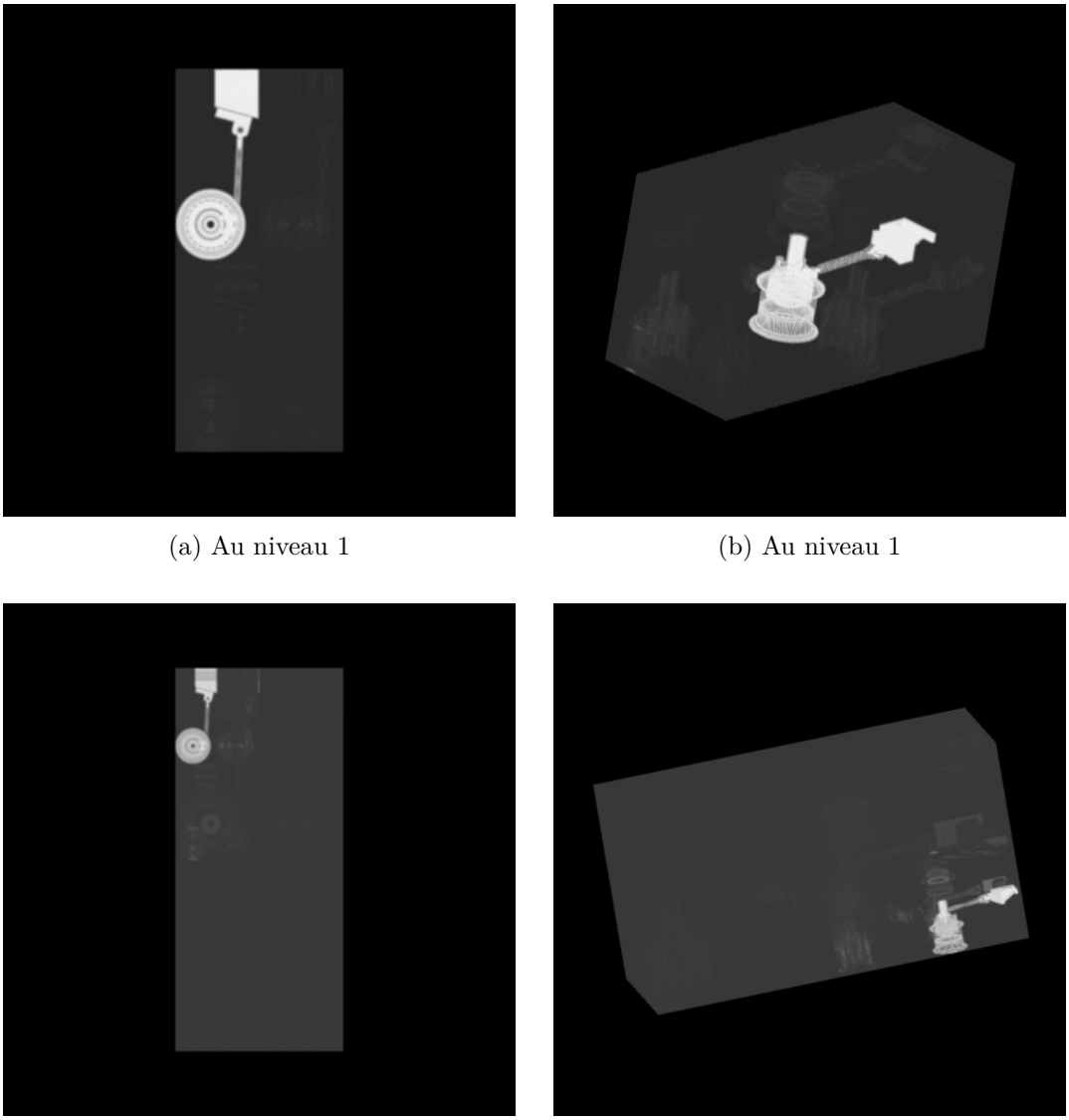
Sur chaque bloc, les valeurs sont comprises entre une valeur minimale v_{min} et maximale v_{max}. Lors de la quantification, cet intervalle de valeurs $[v_{min}, v_{max}]$ est découpé en intervalles de taille maximale 2ε. A chaque intervalle correspond une seule valeur entière. La valeur zéro est attribuée au premier intervalle de valeurs, la valeur un au deuxième et ainsi de suite.

La question qui se pose à présent est de savoir en combien d'intervalles le bloc va être divisé afin que l'erreur finale ne dépasse pas l'erreur fixée. Répondre à cette question revient à calculer le nombre de bits nécessaire pour coder un coefficient. Ce nombre de bits est décidé selon l'erreur maximale envisagée et l'écart entre les valeurs minimale et maximale.

Soient v_{min} et v_{max} les valeurs respectivement minimale et maximale du bloc, ε l'erreur de quantification et n le nombre de bits sur lequel va être codé chaque coefficient alors :

$$\frac{v_{max} - v_{min}}{2^n} < \varepsilon. \tag{5.6}$$

Une inéquation sur le nombre de bits peut en être déduite :



(a) Au niveau 1 (b) Au niveau 1

(c) Au niveau 2 (d) Au niveau 2

FIGURE 5.6 – Transformée en ondelettes d'un hypercube 3D

$$\begin{cases} n > \log_2\left(\dfrac{v_{max} - v_{min}}{\varepsilon}\right) & , \; si \; v_{max} - v_{min} \geq \varepsilon. \\ n = 0 & , \; si \; v_{max} - v_{min} < \varepsilon. \end{cases} \tag{5.7}$$

Un fois la quantification faite, une erreur uniforme $\leq \varepsilon$ est présente sur tout le bloc. La figure 5.7 schématise cette quantification. L'intervalle $[v_{min}, v_{max}]$ est divisé en 2^n valeurs

avec n minimisant l'inéquation 5.7. A chaque intervalle est associée une valeur entière de l'intervalle final.

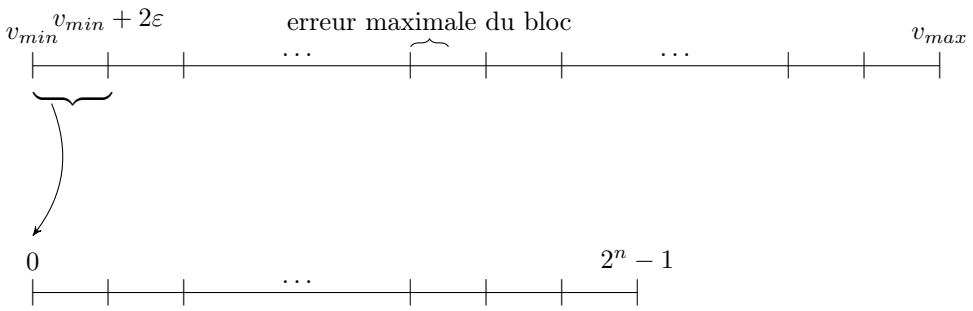


FIGURE 5.7 – Schéma expliquant la quantification

Pour obtenir un bon taux de compression, le nombre de bits n doit être le plus petit possible. D'après l'inégalité (5.7) il est important pour cela que l'écart entre les valeurs maximale et minimale ne soit pas trop important. C'est pourquoi la quantification n'est pas appliquée sur la donnée totale mais sur des sous-blocs. Les sous-blocs utilisés par la suite sont des blocs de taille seize par seize par seize coefficients (ou seize par seize par seize par seize coefficients pour des données 4D). Pour retrouver les coefficients d'ondelettes d'un bloc donné, il faut garder les valeurs minimale et maximale ainsi que les $n * taille\_du\_bloc$ bits des coefficients quantifiés. Le nombre de bits n qui a servi pour coder un coefficient est retrouvé grâce à l'inégalité (5.7).

La transformée en ondelettes et le processus de quantification sont appliqués à l'hypercube du modèle dont une coupe a été montrée précédemment en figure 5.3 ainsi qu'aux résultats obtenus par une analyse modale sur ce modèle, montrés dans la figure 5.8.

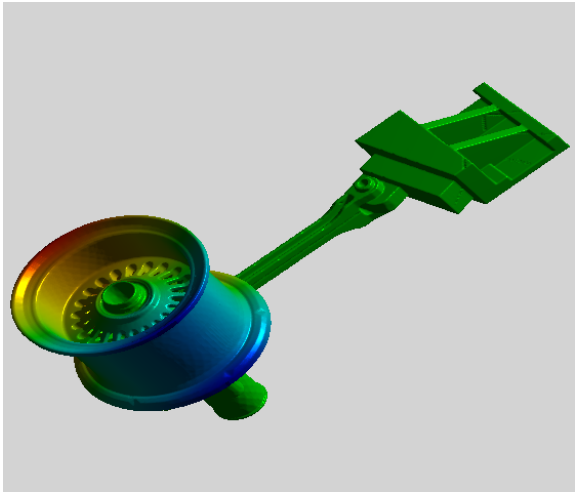


FIGURE 5.8 – Analyse modale sur une partie d'un train d'atterrissage

Une fois la transformée par ondelettes et le processus de quantification appliqués à l'hypercube, des premiers taux de compression sont obtenus. Les tableaux des figures 5.9 et 5.10 montrent ces premiers résultats pour les deux hypercubes vus précédemment.

	Transformée en ondelettes au niveau un		Transformée en ondelettes au niveau deux	
ε	Taille du fichier (MB)	Ratio de compression	Taille du fichier (MB)	Ratio de compression
Initial	24.10	1	24.10	1
10^{-6}	6.86	3.51	7.50	3.21
10^{-4}	4.41	5.46	4.87	4.95
10^{-3}	3.18	7.58	3.53	6.83
10^{-2}	1.98	12.17	2.23	10.81

FIGURE 5.9 – Tableau des ratios de compression de l'hypercube du modèle

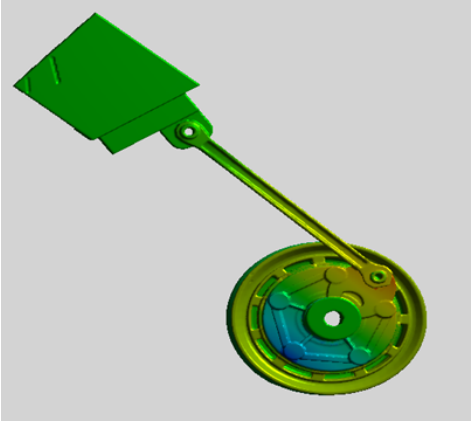
Les tableaux 5.9 et 5.10 montrent que le ratio de compression obtenu pour l'hypercube de données est meilleur que celui du modèle. Cette différence peut s'expliquer par le fait que le modèle a une proportion de valeurs non nulles plus importante que celle des données. Lors de la quantification les sous-blocs de valeurs nulles sont codés sur zéro bit, le gain est très grand. Un meilleur ratio de compression est obtenu pour la transformée en ondelettes au niveau un que deux. Ce comportement se retrouve aux niveaux supérieurs : plus le niveau de la transformée en ondelettes est grand, plus les fichiers sont de taille importante. Les ratios et les tailles de fichiers à un niveau supérieur ne sont donc pas présentés.

La figure 5.11 compare l'hypercube de données initial avec le même hypercube re-

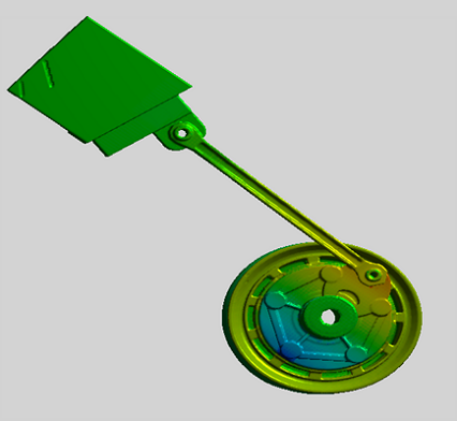
	Transformée en ondelettes au niveau un		Transformée en ondelettes au niveau deux	
ε	Taille du fichier (MB)	Ratio de compression	Taille du fichier (MB)	Ratio de compression
Initial	72.50	1	72.50	1
10^{-6}	14.40	5.03	15.80	4.59
10^{-4}	7.33	9.89	8.13	8.92
10^{-3}	4.21	17.22	4.74	15.30
10^{-2}	1.67	43.41	1.96	36.99

FIGURE 5.10 – Tableau des ratios de compression de l'hypercube des données

constitué. L'hypercube 5.11b a été passé par la transformée en ondelettes puis par une quantification avec une erreur $\varepsilon = 10^{-2}$.



(a) Hypercube initial



(b) Hypercube reconstitué

FIGURE 5.11 – Visualisation de deux hypercubes : initial et avec une erreur de 10^{-2}

Comme visible sur la figure 5.11, une erreur de quantification valant 10^{-2} n'affecte pas la visualisation. L'hypercube de la figure 5.11b prend plus de quarante fois moins de place que l'hypercube initial 5.11a. Ces premiers résultats sont encourageants, la prochaine section montre des meilleurs résultats de compression grâce à un codage entropique.

5.4.2 Codage entropique

Un codage entropique est une méthode de codage sans perte où les valeurs ayant le plus de redondance sont codés sur moins de bit que celles ayant moins de redondance. Des codages entropiques existent déjà, par exemple le codage de Huffman qui range les valeurs

dans un arbre selon leur nombre d'occurrences.

Afin de savoir si la quantification code de façon optimale les coefficients, un histogramme est tracé pour chaque bloc. Ces histogrammes représentent le nombre de fois où chaque valeur entière quantifiée apparaît. Ainsi ces histogrammes permettent de voir si en effet, toutes les 2^n valeurs sont utilisées (n représente toujours le nombre de bits) ou du moins une grande majorité. La figure 5.12 représente l'histogramme du premier bloc dont le nombre de bits est non nul de l'hypercube du modèle vu précédemment. Chaque coefficient de ce bloc a été codé sur seize bits, les valeurs quantifiées appartiennent donc à $[0, 65535]$. Sa valeur minimale est 0 et sa valeur maximale 0.005181.

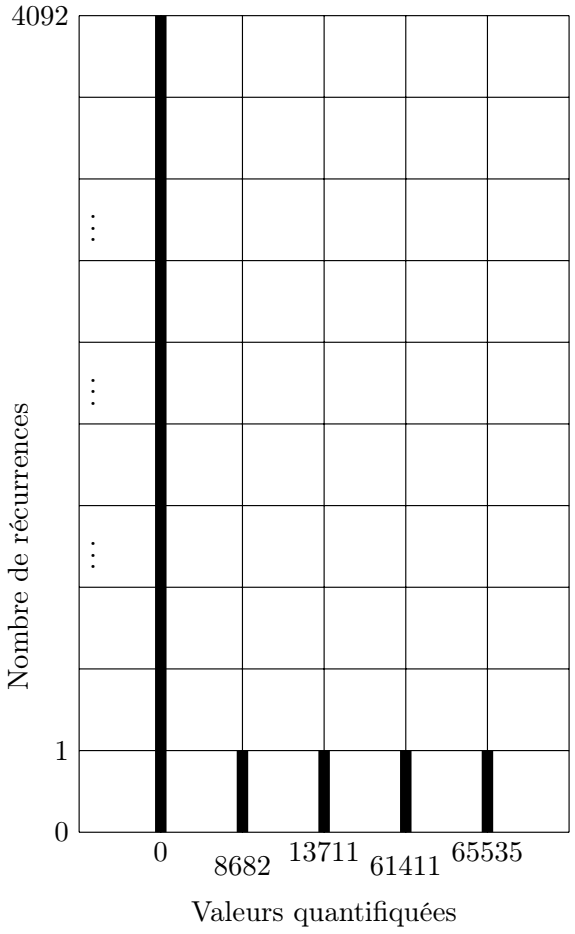


FIGURE 5.12 – Histogramme d'un bloc quantifié sur 16bits

Les valeurs dont le nombre de récurrences est nul n'ont pas été mises sur l'histogramme par souci de lisibilité. Comme on peut le voir, pour ce bloc, seules cinq valeurs entières

différentes sont utilisées. De plus, seule la valeur 0 revient plusieurs fois (4092 fois). Cela est du au fait que notre bloc est sur le bord du modèle, il possède énormément de valeurs nulles.

Le bloc suivant, qui n'est plus au bord du modèle, est codé sur vingt-quatre bits. Ses valeurs minimale et maximale sont -0.178765 et 2.957035. Ce bloc a 1478 valeurs différentes, l'histogramme est trop grand pour être montré. Sur ces différentes valeurs, seule la valeur 956434 revient beaucoup (2617 fois), les autres ne sont utilisées qu'une fois ou deux. En regardant les histogrammes des autres blocs, ce comportement se retrouve. Une valeur par bloc est dominante, les autres sont très peu utilisées voire pas du tout. La quantification ne semble pas optimale. C'est pourquoi une seconde méthode de codage est appliquée après la quantification.

La méthode utilisée est un codage entropique. L'idée de ce codage est de coder sur moins de bits les valeurs qui reviennent le plus souvent. Dans notre cas, seule une valeur revient, le codage utilisé est différent. Pour commencer, la valeur qui domine sur chaque bloc est calculée, celle qui est la plus fréquente. Cette valeur dominante est ensuite codée sur un seul bit, les autres valeurs sont codées sur $n + 1$ bits. La valeur dominante est représentée par 0, les autres valeurs sont représentées par 1 suivi de la valeur quantifiée. La valeur quantifiée est toujours codée sur n bits, elle n'a pas été modifiée. En gardant le nombre de bits n de la quantification et la suite de valeurs re-codées par bloc, le coefficient d'ondelette peut toujours être retrouvé.

La prochaine section présente les résultats de compression sur différents modèles après application des trois étapes : la transformation en ondelettes, la quantification et le codage entropique.

5.4.3 Résultats sur des modéles 3D

Les résultats de compression sur quatre modèles en trois dimensions vont être donnés. Le premier modèle représente un piston, le deuxième une barre, le troisième une partie d'un train d'atterrissage, le dernier modèle représente un treuil hydraulique.

Piston

La figure 5.13 montre une coupe du modèle du piston.

Les résultats de la compression du piston sont montrés dans le tableau 5.14. Une analyse statique a été appliquée au modèle. La même tendance que sur l'autre modèle se retrouve, les données sont mieux compressées au niveau un de la transformée en ondelettes.

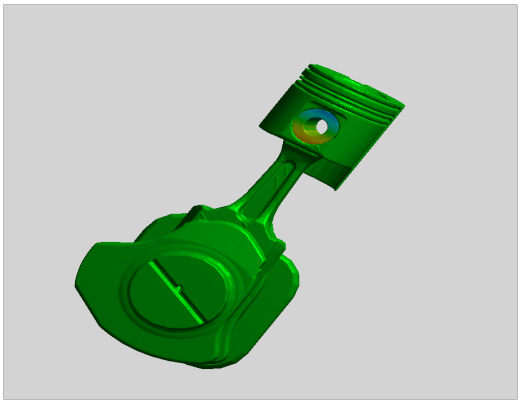


FIGURE 5.13 – Analyse statique sur le modèle piston

	Transformée en ondelettes au niveau un		Transformée en ondelettes au niveau deux	
ε	Taille du fichier (KB)	Ratio de compression	Taille du fichier (KB)	Ratio de compression
Initial	117 601	1	117 601	1
10^{-6}	1 049	112.11	1 094	107.50
10^{-2}	530	221.89	536	219.40

FIGURE 5.14 – Tableau des ratios de compression des résultats sur piston

Barre

Une coupe du modèle de la barre est présentée sur la figure 5.15. Une analyse statique a aussi été appliquée sur ce modèle.

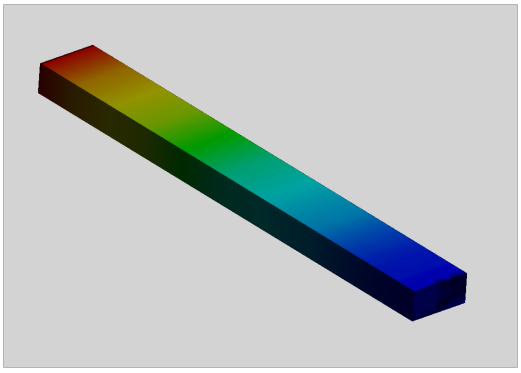


FIGURE 5.15 – Analyse statique sur le modèle barre

Les résultats de compression du modèle barre sont montrés dans le tableau 5.16 :

	Transformée en ondelettes au niveau un		Transformée en ondelettes au niveau deux	
ε	Taille du fichier (KB)	Ratio de compression	Taille du fichier (KB)	Ratio de compression
Initial	7 172	1	7 172	1
10^{-6}	697	10.29	524	13.69
10^{-2}	37	193.84	28	256.14

FIGURE 5.16 – Tableau des ratios de compression des résultats sur barre

Train d'atterrissage

La figure 5.17 affiche une coupe d'une partie d'un train d'atterrissage, c'est le même modèle que celui présenté dans la section 5.4.1.

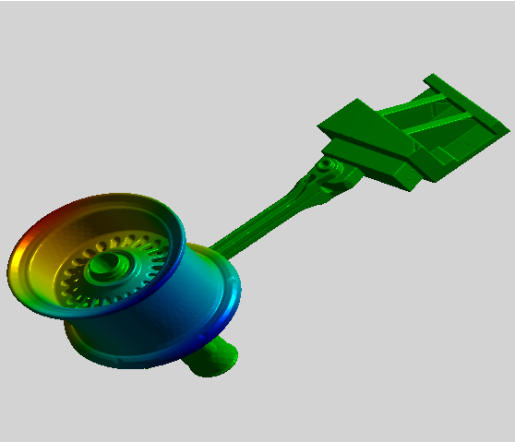


FIGURE 5.17 – Analyse modale sur le modèle train d'atterrissage

Le tableau 5.18 montre les résultats de la compression sur l'analyse modale du train d'atterrissage.

	Transformée en ondelettes au niveau un		Transformée en ondelettes au niveau deux	
ε	Taille du fichier (KB)	Ratio de compression	Taille du fichier (KB)	Ratio de compression
Initial	74 251	1	74 251	1
10^{-6}	1 699	43.70	1 872	39.66
10^{-2}	695	106.84	723	102.70

FIGURE 5.18 – Tableau des ratios de compression des résultats sur train d'atterrissage

Treuil hydraulique

Une analyse transitoire a été lancée pour le modèle treuil hydraulique présenté dans la figure 5.19. La compression en ondelettes 3D va être appliquée à différents pas de temps de cette analyse, cela permettra de voir si un même modèle comprenant des données différentes donne les même taux de compression.

Les résultats de compression en 3D sont montrés seulement pour deux pas de temps sur les figures 5.20 et 5.21.

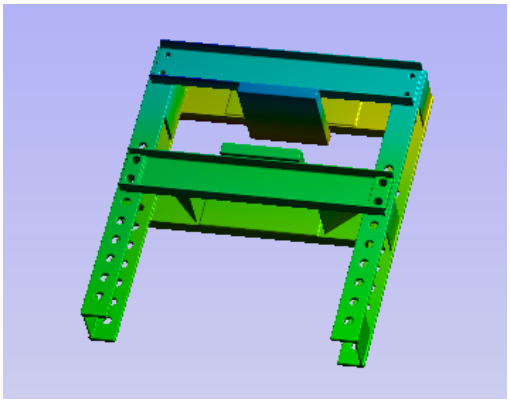
	Transformée en ondelettes au niveau un		Transformée en ondelettes au niveau deux	
ε	Taille du fichier (KB)	Ratio de compression	Taille du fichier (KB)	Ratio de compression
Initial	161 251	1	161 251	1
10^{-6}	15 155	10.64	11 136	14.48
10^{-2}	3 674	43.89	3 240	49.77

FIGURE 5.20 – Tableau des ratios de compression des résultats sur le treuil hydraulique au temps : 1

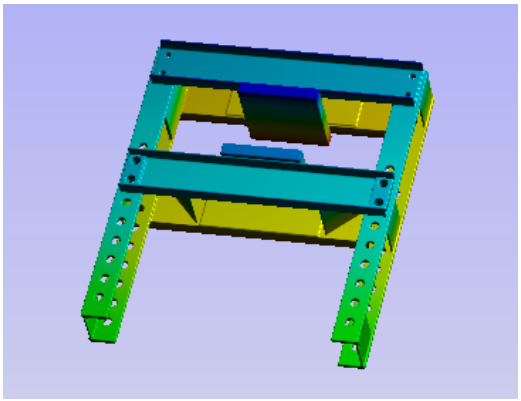
	Transformée en ondelettes au niveau un		Transformée en ondelettes au niveau deux	
ε	Taille du fichier (KB)	Ratio de compression	Taille du fichier (KB)	Ratio de compression
Initial	161 251	1	161 251	1
10^{-6}	11 392	14.15	10 551	15.28
10^{-2}	2 897	55.66	2 127	75.81

FIGURE 5.21 – Tableau des ratios de compression des résultats sur le treuil au temps : 11

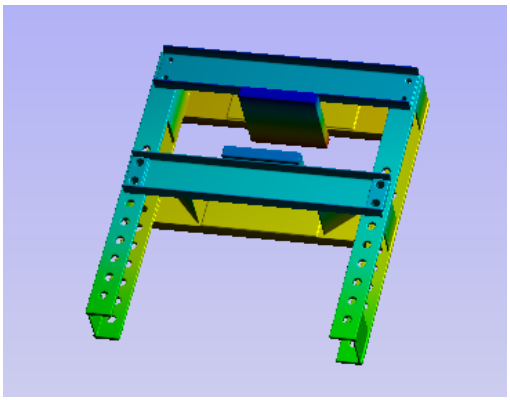
Les ratios de compression sont différents pour les différents pas de temps mais néan-



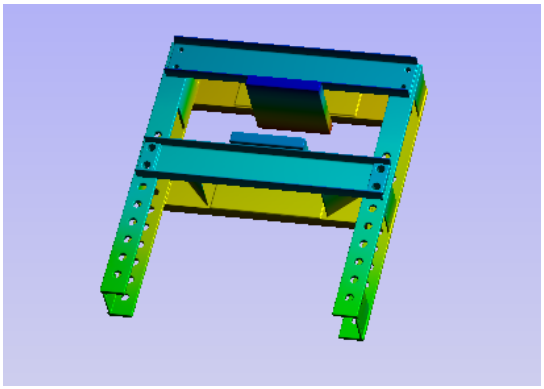
(a) Le modèle avec résultats d'analyse au temps = 1



(b) Le modèle avec résultats d'analyse au temps = 11



(c) Le modèle avec résultats d'analyse au temps = 21



(d) Le modèle avec résultats d'analyse au temps = 31

FIGURE 5.19 – Le modèle treuil hydraulique avec les résultats transitoires

moins proches.

Au lieu de compresser les modèles transitoires pas de temps par pas de temps, nous nous sommes intéressés à une compression de la dimension temporelle. La prochaine section montre la compression utilisée pour ces données en quatre dimensions.

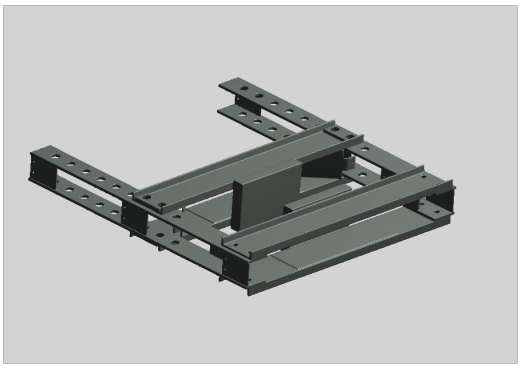
5.5 Phénomène transitoire : Données 3D + t

La première idée a été de considérer la dimension temporelle au même titre que les dimensions spatiales et d'appliquer une compression en ondelettes en dimension quatre sur une donnée comprenant tous les pas de temps. La prochaine section explique la compression par ondelettes en dimension quatre.

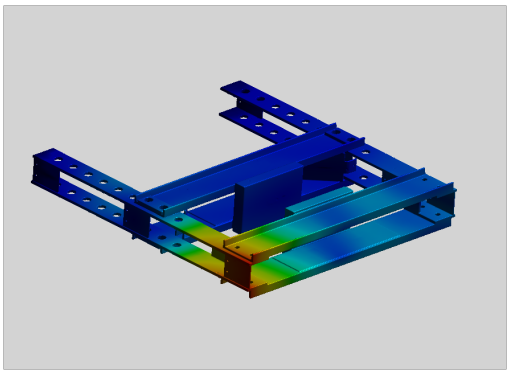
5.5.1 Compression 4D

Le processus de compression en ondelettes de données 4D est le même que celui pour des données de dimension trois. C'est à dire une décomposition par ondelettes suivie par une quantification puis un codage entropique. La transformée en ondelettes 4D a déjà été présentée. La quantification et le codage entropique utilisés fonctionnent de la même façon que dans le cas en dimension trois, mis à part que les blocs sont maintenant de taille seize par seize par seize par seize coefficients. En appliquant la transformée par ondelettes dans les quatre dimensions, des ratios de compression plus importants que dans le cas 3D sont attendus.

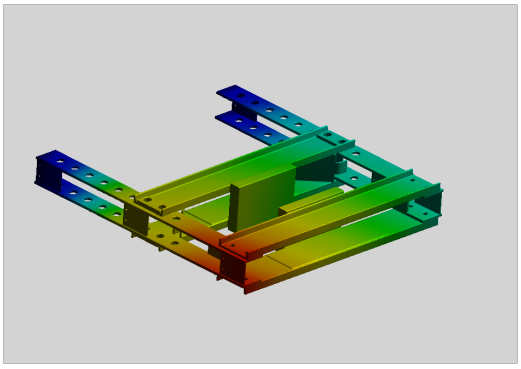
Pour avoir des hypercubes 4D, une analyse est lancée sur plusieurs pas de temps. Dans cette partie, le même modèle que précédemment est utilisé, visible sur la figure 5.22. Ce modèle représente une pièce d'un treuil hydraulique. Les différents résultats sont issus d'une analyse transitoire lancée pour mille pas de temps.



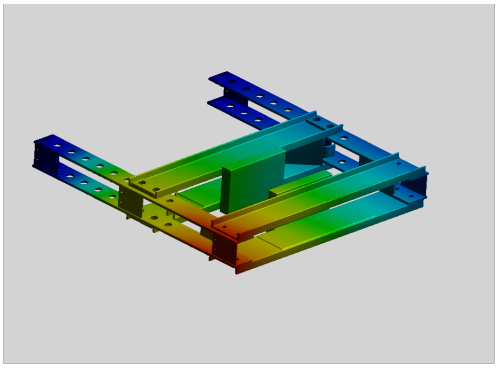
(a) Le modèle



(b) Analyse transitoire au temps = 1



(c) Analyse transitoire au temps = 50



(d) Analyse transitoire au temps = 100

FIGURE 5.22 – Modèle utilisé pour la 4D

La compression a été appliquée seulement sur les quarante premiers pas de temps, les

résultats de la compression sont montrés dans le tableau 5.23.

	Transformée en ondelettes au niveau un		Transformée en ondelettes au niveau deux	
ε	Taille du fichier (KB)	Ratio de compression	Taille du fichier (KB)	Ratio de compression
Initiale	40 × 161 251	1	40 × 161 251	1
10^{-6}	425 559	15.16	443 144	14.56
10^{-2}	75 480	85.45	68 084	94.74

FIGURE 5.23 – Tableau des ratios de compression des résultats sur le treuil pour les 40 premiers pas de temps

En ajoutant une quatrième dimension, des meilleurs résultats de compression étaient attendus. Afin d'améliorer ces résultats, un processus différent est utilisé pour compresser la dimension temporelle, puis la compression en ondelettes en dimension trois sera appliquée sur le résultat.

5.5.2 Décomposition en valeurs singulières incrémentale

La dimension temporelle va être compressée grâce à une décomposition en valeurs singulières incrémentale. L'idée est de ne pas appliquer la décomposition en valeurs singulières sur toute la donnée, le temps de compression peut vite être trop important si la donnée possède énormément de pas de temps. De plus, l'incrément permet de compresser les premiers pas de temps même si la simulation n'est pas finie.

La décomposition en valeurs singulières consiste à factoriser une matrice en trois matrices particulières. Cette méthode permet de décomposer la matrice en modes rangés selon des critères énergétiques. Une fois ces modes ordonnés, la matrice initiale peut être reconstituée avec seulement un certains nombres de lignes (ou colonnes) des matrices particulières, l'erreur de reconstitution est connue. La décomposition en valeurs singulières peut être vue comme une réduction de donnée.

Soit M une matrice $m \times n$ à coefficients complexes, alors M peut s'écrire :

$$M = U\Sigma V^{\star},$$

où U et V sont deux matrices unitaires $m \times n$ et $n \times n$ respectivement et Σ est une matrice diagonale $n \times n$ dont les coefficients de la diagonale sont les valeurs singulières de M.

Les colonnes $u_1, \ldots, u_m$ forment une base orthonormée de K^m, les colonnes $v_1, \ldots, v_m$ forment une base orthonormée de K^n. Les valeurs singulières de M sont triées dans la diagonale de Σ de telle sorte que les valeurs singulières représentant le plus d'informations se trouvent au début. Il suffit alors de ne garder que les k premiers coefficients de Σ ainsi que les k premières colonnes de U, les k premières colonnes de V pour reconstituer une bonne

approximation de la matrice initiale M. Les valeurs singulières stockées sur la diagonale de Σ donne la contribution de chaque vecteur.

Traitement de la dimension temporelle

Cette décomposition se faisant sur des matrices, on a restructuré nos données en espace dans une matrice Ψ. On a ainsi un vecteur par N pas de temps.

Algorithme 3 : Compression de la dimension temporelle

Fonction `CompressSVD`

input : Ψ : La matrice à compresser
N : taille de la dimension temporelle de la matrice Ψ
ε : L'erreur souhaitée

output : Ψ_0 : La matrice compressée
M : taille de la dimension temporelle de la matrice compressée Ψ_0

Construction à partir de Ψ d'une nouvelle matrice comprenant 10 pas de temps : Ψ_{idx} ;
Application de la SVD sur Ψ_{idx} : U_{idx}, Σ_{idx}, V_{idx};
Création d'une nouvelle matrice U_0 avec les k premiers vecteurs de U_{idx} dont la contribution est inférieure à ε : U_0 ;
Projection de la solution selon U_0 : $\Psi_0 = \langle U_0, \Psi \rangle U_0$;

si *l'erreur* $\dfrac{\|\Psi_0\|_\infty}{\|\Psi\|_\infty}$ *est inférieure à l'erreur* ε **alors**
  Arrêt de la fonction ;
sinon
  Découpe de la matrice Ψ_0 en deux nouvelles matrices : Λ_0, Λ_1;
  Application de la même fonction **CompressSVD** sur Λ_0 et Λ_1 ;
fin

end

Résultats sur le treuil hydraulique

Cette méthode est lancée sur le même modèle qu'utilisé précédemment, le treuil hydraulique montré sur les figures 5.19, 5.22. La compression est toujours appliquée sur les quarante premiers pas de temps.

Le modèle étant trop gros pour MATLAB, on ne garde qu'une coupe en (x, y) de notre modèle 3D sur chaque pas de temps. Avant de procéder à une décomposition en valeurs singulières sur seulement dix pas de temps, la décomposition en valeurs singulières a été appliquée sur les quarante pas de temps afin de regarder combien de vecteurs singuliers doivent être gardés par SVD pour arriver à l'erreur souhaitée.

Après décomposition en valeurs singulières, la contribution jusqu'à la valeur singulière j est donnée par :

$$S = \frac{\lambda_1 + \lambda_2 + \ldots + \lambda_j}{trace(\Sigma)}. \tag{5.8}$$

La figure 5.24 trace ces contributions, cela permet d'avoir l'erreur après reconstitution selon le nombre de vecteurs singuliers que l'on garde.

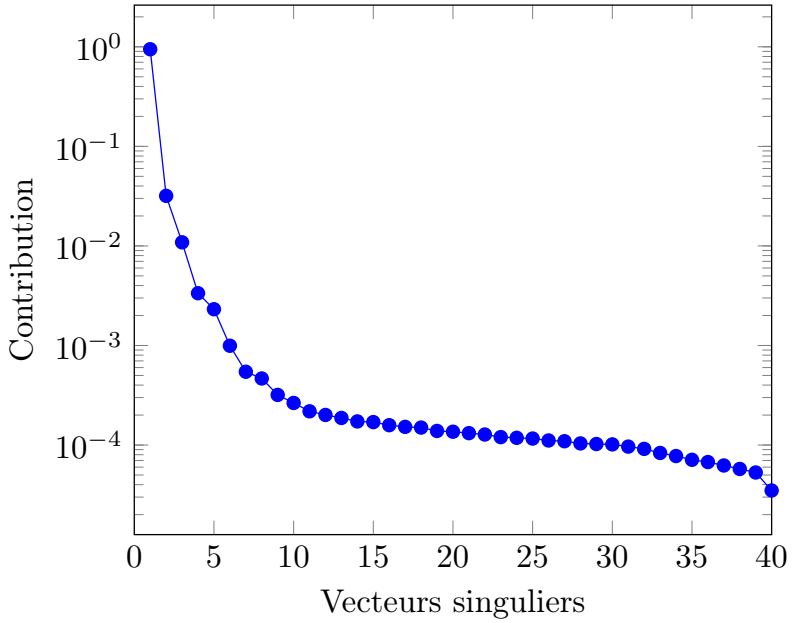


FIGURE 5.24 – Contribution de chaque valeur singulière

Comme visible sur la figure 5.24, il va falloir garder les 40 vecteurs pour obtenir une erreur inférieure à 10^{-6}. Lorsque le procédé de décomposition en valeurs singulières incrémentale expliqué précédemment est appliquée sur cette donnée, le même résultat est retrouvé. Il faut aller assez loin dans l'arborescence, jusqu'à faire une SVD sur seulement un vecteur pour avoir une erreur inférieure à 10^{-3}. Cette méthode est donc testée sur un nouveau modèle, une plaque trouée.

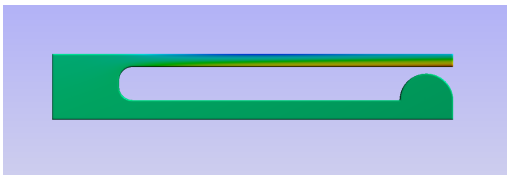
Résultats sur une plaque trouée

Le nouveau modèle utilisé est une plaque trouée comme montrée sur la figure 5.25. Une force est simulée sur la partie haute afin de voir la comportement avant et après le contact entre la partie haute et la partie circulaire. Les résultats de cette analyse sont montrés dans la figure 5.26 pour quatre pas de temps.

FIGURE 5.25 – Le modèle représentant la plaque trouée

(a) Le modèle avec résultats d'analyse au temps = 1

(b) Le modèle avec résultats d'analyse au temps = 10



(c) Le modèle avec résultats d'analyse au temps = 20

(d) Le modèle avec résultats d'analyse au temps = 30

FIGURE 5.26 – Les résultats sur le modèle plaque trouée

Cette fois, la donnée est composée de trente pas de temps et la décomposition en valeurs singulières peut être lancée sur la donnée entière, le modèle étant plus petit. L'erreur selon le nombre de vecteurs singuliers gardés est montrée dans la figure 5.27.

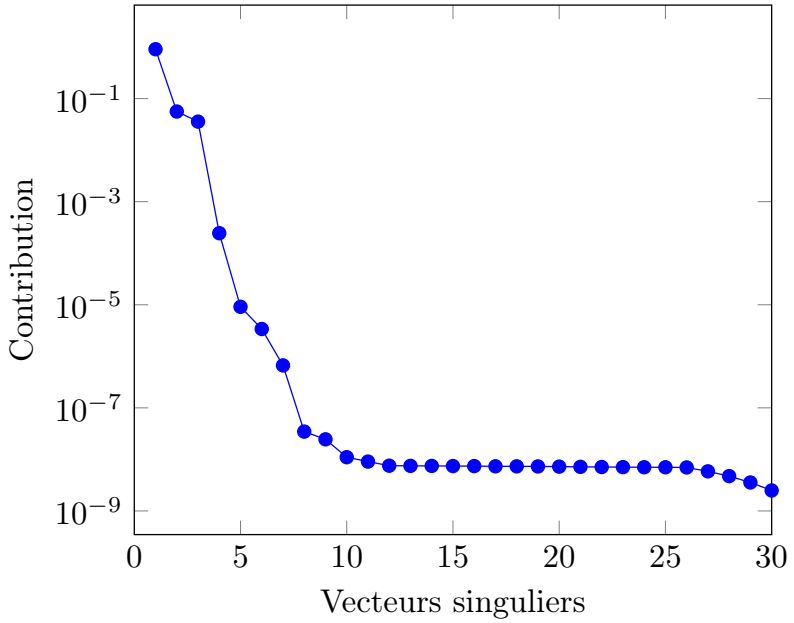


FIGURE 5.27 – Erreur après reconstitution selon le nombre de vecteurs singuliers gardés

La figure 5.27 montre que lorsque la donnée entière est décomposée par SVD, il suffit ensuite de ne garder que sept vecteurs singuliers pour reconstruire la donnée avec une erreur inférieure à 10^{-6}. La décomposition en valeurs singulières incrémentale, montrée dans l'algorithme 3 est appliquée à cette donnée de trente pas de temps. De même que pour la décomposition en valeurs singulières standard, seuls sept vecteurs suffisent pour retrouver la donnée initiale. Ensuite la compression en ondelettes 3D, montrée dans la section 5.4 est appliquée aux sept vecteurs obtenus. Les ratios de compression sont affichés dans le tableau 5.28.

	SVD et Transformée en ondelettes au niveau un	
ε	Taille du fichier (KB)	Ratio de compression
Initiale	$30 \times 4\ 022$	1
10^{-6}	3 814	31.64
10^{-2}	454	265.77

FIGURE 5.28 – Tableau des ratios de compression de la plaque trouée pour les 30 pas de temps

De meilleurs ratios de compression sont bien obtenus que lors de la compression par

ondelettes 4D.

5.5.3 Conclusion

Un processus de compression vient d'être présenté et a été mis en place à la fois pour des données en trois dimensions stationnaires et transitoires. Cette compression donne des résultats intéressants que ce soit pour des données $3D$ ou $3D+t$. Des ratios de compression allant de 10.29 à 112.11 sont obtenus pour la compression en dimension trois avec une erreur de 10^{-6}. Des ratios allant de 43.89 à 221.89 pour une erreur de 10^{-2}, erreur qui permet tout de même une bonne visualisation des données.

De plus, la dimension temporelle étant gérée de façon incrémentale, des données de grande dimension temporelle peuvent être compressées. Au lieu de charger un seul résultat de la donnée la plus grande possible acceptée en mémoire, de 10 à 200 fois plus de résultats peuvent être chargés. Cet aspect peut être particulièrement intéressant pour faire une animation.

Chapitre 6

Conclusions et perspectives

Les processeurs graphiques offrent de nouvelles puissances de calcul que n'exploitent pas les méthodes actuelles. La génération et l'utilisation d'un maillage conforme est clairement une barrière à la performance. Le maillage conforme de la méthode des éléments finis standards rappelée en section 2.2 est incompatible avec la parallélisation souhaitée par les processeurs graphiques. Au cours de cette thèse, nous avons exploité l'idée d'une méthode éléments finis basée sur un maillage cartésien, maillage qui épouse l'architecture grille des processeurs graphiques.

La méthode des éléments finis étendus présentée dans la section 2.3 est déjà bien maîtrisée dans de nombreux domaines, notamment en mécanique de la rupture. Cette méthode peut être implémentée à la fois sur maillage conforme comme non conforme à la géométrie. Notre méthode, inspirée de la méthode des éléments finis étendus a été développée et validée sur des modèles simples, conduisant à des résultats précis même sur un maillage peu raffiné. Cette méthode ainsi qu'une comparaison des résultats avec la méthode des éléments finis classique ont été montrés dans le chapitre 3.

Les performances de cette méthode en terme de temps de calcul ont ensuite été améliorés par l'introduction d'une méthode d'apprentissage expliquée dans le chapitre 4 qui permet de pré-former les matrices élémentaires une fois pour toute, sans perte de précision. Cette nouvelle façon d'intégrer les matrices donnent de bons résultats en deux dimensions et une rapidité d'exécution lors de l'intégration. Si l'extension à la troisième dimension est évidente, de nouveaux éléments coupés se retrouvent à être appris et plus nombreux. Toute la routine d'apprentissage doit donc être refaite. L'apprentissage pourra être pensé différemment, notamment en introduisant des techniques de krigeage. Des recherches d'invariants par rotation de l'élément permettraient également d'accélérer l'apprentissage et le rendre plus précis.

En continuant l'exploitation optimale de la grille du processeur graphique, nous nous sommes ensuite intéressés à la compression des résultats d'analyse, aussi bien en espace qu'en temps. Ce processus de compression expliqué dans le chapitre 5 utilise la transfor-

mée en ondelettes en espace et des méthodes de projection en temps. La transformée en ondelettes, bien connue dans le traitement de l'image, donne également de très bons résultats sur les résultats d'analyse. L'avantage de la transformée en ondelettes est qu'elle est hautement parallélisable et donc bien compatible avec les processeurs graphiques. Il serait intéressant d'exploiter mieux le caractère multi-échelle des ondelettes afin de traiter les résultats de manière hiérarchique.

Bibliographie

[BB99] T. BELYTSCHKO et T. BLACK. « Elastic crack growth in finite elements with minimal remeshing ». In : *International journal for numerical methods in engineering* 45.5 (1999), p. 601–620.

[BCO94] I. BABUŠKA, G. CALOZ et J.E. OSBORN. « Special finite element methods for a class of second order elliptic problems with rough coefficients ». In : *SIAM Journal on Numerical Analysis* 31.4 (1994), p. 945–981.

[BEL+03] T. BELYTSCHKO et al. « Structured extended finite element methods for solids defined by implicit surfaces ». In : *International journal for numerical methods in engineering* 56.4 (2003), p. 609–635.

[BHL07] P. BUSCH, T. HEINONEN et P. LAHTI. « Heinsenberg's uncertainty principle ». In : *Physics reports* 452.6 (2007), p. 155–176.

[BM97] I. BABUŠKA et J.M. MELENK. « The Partition Of Unity Method ». In : *International journal for numerical methods engineering* (1997).

[CDF92] A. COHEN, I. DAUBECHIES et J.C. FEAUVEAU. « Biorthogonal Bases of Compactly Supported Wavelets ». In : *Communications on pure and applied mathematics* 45.5 (1992), p. 485–560.

[DMB00] J. DOLBOW, N. MOES et T. BELYTSCHKO. « Discontinuous enrichment in finite elements with a partition of unity method ». In : *Finite elements in analysis and design* 36.3 (2000), p. 235–260.

[DT87] G. DHATT et G. TOUZOT. *Une présentation de la méthode des éléments finis*. Université de Compiègne, 1987.

[Exa] Exa. *PowerFlow*. URL : `http://www.exa.com/product/powerflow`.

[FB10] T-P. FRIES et T. BELYTSCHKO. « The extended/generalized finite element method : An overview of the method and its applications ». In : *International journal for numerical methods in engineering* 84.3 (2010), p. 253–304.

[GM84] A. GROSSMANN et J. MORLET. « Decomposition of hardy functions into square integrable wavelets of constant shape ». In : *SIAM journal on mathematical analysis* 15.4 (1984), p. 723–736.

[GMB02] A. GRAVOUIL, N. MOËS et T. BELYTSCHKO. « Non-planar 3D crack growth by the extended finite element and level sets - Part II : Level set update ». In : *International journal for numerical methods in engineering* 53.11 (2002), p. 2569–2586.

[Hu+14] Wei Hu et al. « Ray tracing via GPU rasterization ». In : *The Visual Computer* 30.6-8 (2014), p. 697–706.

[KOU15] D.J. KOURI. « Scaled Fourier Transforms and Heisenberg's Uncertainty Principle ». In : *Journal of the chinese chemical society* 63.1 (2015), p. 145–149.

[LAB+05] P. LABORDE et al. « High-order extended finite element method for craked domains ». In : *International journal for numerical methods in engineering* 64.3 (2005), p. 354–381.

[LLK16] P. LIU, Y. LUO et A. KANG. « Multi-material topology optimization considering interface behavior via XFEM and level set method ». In : *Computer methods in applied mechanics and engineering* 308 (2016), p. 113–133.

[LM86] P.G. LEMARIE et Y. MEYER. « Ondelettes et bases hilbertiennes ». In : *Revista Matemática Iberoamericana* 2.1-2 (1986), p. 1–18.

[LZ13] C. LINDER et X. ZHANG. « A marching cubes based failure surface propagation concept for three-dimensional finite elements with non-planar embedded strong discontinuities of higher-order kinematics ». In : *International journal for numerical methods in engineering* 96.6 (2013), p. 339–372.

[MAL08] S. MALLAT. *A wavelet tour of signal processing : the sparse way.* Academic press, 2008.

[MB96] J.M. MELENK et I. BABUŠKA. « The partition of unity finite element method : Basic theory and applications ». In : *Computer methods in applied mechanics and engineering* 139.1-4 (1996), p. 289–314.

[MC92] M.VETTERLI et C.HERLEY. « Wavelets and filter banks : Theory and design ». In : *IEEE transactions on signal processing* 40.9 (1992), p. 2207–2232.

[MDB99] N. MOES, J. DOLBOW et T. BELYTSCHKO. « A finite element method for crack growth without remeshing ». In : *International journal for numerical methods in engineering* 46.1 (1999), p. 131–150.

[MEY85] Y. MEYER. « Principe d'incertitude, bases hilbertiennes et algèbres d'opérateurs ». In : *Séminaire N. Bourbaki* 28 (1985), p. 209–223.

[MGB02] N. MOËS, A. GRAVOUIL et T. BELYTSCHKO. « Non-planar 3D crack growth by the extended finite element and level sets - Part I : Mechanical model ». In : *International journal for numerical methods in engineering* 53.11 (2002), p. 2549–2568.

[ND10] J. NICKOLLS et W. J. DALLY. « The GPU Computing Era ». In : *IEEE Computer Society* 30.2 (2010).

[OWE+07] J.D. OWENS et al. « A Survey of General-Purpose Computation on Graphics Hardware ». In : *Computer graphics forum.* T. 26. 1. Wiley Online Library. 2007, p. 80–113.

[OWE+08] J. D. OWENS et al. « GPU Computing ». In : *Proceedings of the IEEE* 96.5 (2008), p. 879–899.

[PAU06] G. PAU. « Ondelettes et décompositions spatio-temporelles avancées ; application au codage vidéo scalable ». Thèse de doct. Ecole nationale supérieure des télécommunications, 2006.

[RAN08] J. RANNOU. « Prise en compte d'effets d'échelle en mécanique de la rupture tridimensionnelle par une approche X-FEM multigrille localisée non-linéaire ». Thèse de doct. Institut National des Sciences Appliquées de Lyon, 2008.

[RJ02] M. RABBANI et R. JOSHI. « An overview of the JPEG 2000 still image compression standard ». In : *Signal processing : Image Compression* 17.1 (2002), p. 3–48.

[SCB01] T. STROUBOULIS, K. COPPS et I. BABUŠKA. « The generalized finite element method ». In : *Computer methods in applied machanics and engineering* 190.32 (2001), p. 4081–4193.

[SK10] J. SANDERS et E. KANDROT. *CUDA by Example : An Introduction to General-Purpose GPU Programming, Portable Documents.* Addison-Wesley Professional, 2010.

[STO+01] M. STOLARSKA et al. « Modelling crack growth by level sets in the extended finite element method ». In : *International journal for numerical methods in engineering* 51.8 (2001), p. 943–960.

[TAY+86] R. L. TAYLOR et al. « The patch test, a condition for assessing FEM convergence ». In : *International journal for numerical methods in engineering* 22.1 (1986), p. 39–62.

[TBW76] R.L. TAYLOR, P. BERESFORD et E.L WILSON. « A non-conforming element for stress analysis ». In : *Internation journal for numerical methods in engineering* 10.6 (1976), p. 1211–1219.

[TRO92] P. TROMPETTE. *Mécanique des structures par la méthode des éléments finis*. MASSON, 1992.

[VEN06] G. VENTURA. « On the elimination of quadrature subcells for discontinuous functions in the eXtended Finite-Element Method ». In : *International journal for numerical methods in engineering* (2006).

[WIL+76] E.L. WILSON et al. « Incompatible Displacement Models ». In : *Internation journal for numerical methods in engineering* (1976).

[XK06] Q.Z. XIAO et B.L. KARIHALOO. « Improving the accuracy of XFEM crack tip fields using highed order quadrature and statically admissible stress recovery ». In : *International journal for numerical methods in engineering* 66.9 (2006), p. 1378–1410.

Lors d'analyses numériques en calcul de structures, la génération de maillages conformes sur des modèles à géométrie complexe conduit à des tailles de modèles importantes, et amène à imaginer de nouvelles approches éléments finis. Nous présentons une nouvelle méthode éléments finis dont le maillage ne dépend pas de la géométrie. Les processeurs graphiques ont été utilisés afin d'implémenter cette méthode sur maillage cartésien. Ces processeurs offrent une rapidité d'exécution impressionnante grâce à leur architecture massivement parallèle.

La nouvelle méthode est basée sur la formulation de la méthode des éléments finis étendus. La méthode des éléments finis étendus est une méthode introduite pour la mécanique de la rupture. Elle permet de prendre en compte la géométrie et les interfaces à travers un enrichissement adéquat des fonctions de forme. La méthode introduite s'inspire de la méthode des éléments finis étendus, avec une représentation implicite de la géométrie, ce qui permet une bonne approximation de la géométrie et des conditions aux limites sans pour autant s'appuyer sur un maillage conforme.

La géométrie est représentée par une fonction surfaces de niveau. Une méthode d'intégration adaptée à cette représentation géométrique est proposée. L'intégration peut s'avérer coûteuse en temps de calcul, c'est pour cette raison que nous proposons une technique d'apprentissage donnant la matrice élémentaire de rigidité en fonction des valeurs de la fonction surfaces de niveau aux sommets de l'élément considéré.

Generating a mesh on complex geometries can contribute significantly to the total turnaround time of structural finite element simulations. To reduce the meshing time, we introduce a novel finite element method based on a non conformal mesh. Graphics processing units (GPUs) are used to implement this new method on a Cartesian mesh. GPUs are highly parallel programmable processors, delivering real performance gains on computationally complex, large problems.

The novel method relies on the extended finite element formulation wich was introduced in the field of fracture mechanics. The extended method enriches the basis functions to take care of the geometry and the interface. Our method is drawn on the extended finite element method, with a geometry implicitly defined, wich allows for a good approximation of the geometry and boundary conditions without a conformal mesh.

To represent the model on a Cartesian grid, we use a level set representing a density. A new integration technique is proposed, adapted to the geometrical representation. In order to reduce the computational expense, a learning approach is then considered to form the elementary stiffness matrices as function of density values on the vertices of the elements.