Construction et manipulation de maillages - Application aux géosciences

Guilhem Dupuy

Université de Pau
et des Pays de l'Adour

Département de formation doctorale en informatique

École doctorale des sciences exactes et de leurs applications

Construction et manipulation de maillages
Application aux géosciences

THÈSE

présentée et soutenue publiquement le 18 décembre 2008

pour l'obtention du

Doctorat de l'Université de Pau et des Pays de l'Adour

(spécialité informatique)

par

Guilhem Dupuy

Composition du jury

Rapporteurs :	Jean-Daniel Boissonnat	INRIA Sophia Antipolis - Méditerranée
	Bruno Lévy	INRIA Nancy - Grand Est
Examinateurs :	Dominique Attali	Université Joseph Fourier - Grenoble
	Sébastien Guillon	Total
	Nooman Keskes	Total
Directeurs :	Dimitri Komatitsch	Université de Pau et des Pays de l'Adour
	Bruno Jobard	Université de Pau et des Pays de l'Adour

Laboratoire d'Informatique de Pau et des Pays de l'Adour — EA 3000

Mis en page avec la classe thloria.

Remerciements

Les remerciements.

La dédicace

Table des matières

Introduction générale **1**

1 Modélisation 3D en géologie 1

1.1 Image sismique 2

1.2 Horizon 3

1.3 Faille 3

1.4 Corps 5

2 Outils d'aide à l'interprétation 5

2.1 Construction des Horizons 5

2.2 Construction des failles 8

2.3 Construction des corps 9

3 Organisation de la thèse 11

Chapitre 1 Construction et optimisation du modèle structural **13**

1.1 Introduction 14

1.2 Triangulation des failles 15

1.2.1 Triangulation du nuage de lignes brisées projeté 17

1.2.2 Calcul de l'enveloppe de la faille 20

1.2.3 Raffinement de la surface 22

1.3 Arrangement du réseau de failles 22

1.3.1 Approche surfacique 23

1.3.2 Approche volumique 26

1.4 Interpolation des horizons 29

1.4.1 Notion de voisinage 30

1.4.2 Estimateurs 35

1.4.3 Étude de l'ordre de propagation dans la stratégie de remplissage 45

1.4.4 Application des méthodes d'interpolation contraintes par les failles 46

1.5 Conclusion 54

Chapitre 2 Extraction des corps dans les cubes attributs **57**

2.1 Contexte d'interprétation géophysique 58

2.1.1 Attribut sismique 58

2.1.2 Lien entre l'enveloppe et l'isosurface 60

2.1.3 Contraintes liées aux géosciences 61

2.2 Quelques notations 63

2.3 Algorithme des *marching cubes* 63

2.4 Adaptation des *marching cubes* aux contraintes du métier 65

2.4.1 Génération d'une 2-variété 65

2.4.2 Gestion des ambiguïtés 67

2.5 Simplification des isosurfaces 70

2.5.1 Approches par décimation adaptative des cubes 70

2.5.2 Simplification en post-traitement 72

2.5.3 Approches mixtes 78

2.6 Parallélisation des méthodes d'extraction d'isosurfaces 85

2.7 Extension du *tandem algorithm* 87

2.7.1 Migration des composants 87

2.7.2 Algorithme parallèle 90

2.7.3 Implémentation maître/esclave 94

2.8 Évaluation de l'algorithme 96

2.8.1 Consommation mémoire 96

2.8.2 Qualité du maillage 97

2.8.3 Temps de calcul 98

2.9 Conclusion 101

Conclusion générale **103**

Annexes **105**

Annexe A Représentations paramétriques **105**

A.1 Notion de paramétrisation 105

A.2 Polynômes de Lagrange 105

A.3 Régressions 106

A.4 Surfaces splines 107

A.4.1 Bézier 107

A.4.2 B-spline 107

A.4.3 NURBS 108

Annexe B Surfaces de subdivision **109**
B.1 Surfaces de subdivisions : définition . . . 109
B.2 Différents schémas de subdivision approximants . . . 110
B.2.1 Catmull-Clark . . . 110
B.2.2 Loop . . . 111
B.2.3 $\sqrt{3}$ subdivision . . . 111
B.3 Différents schémas de subdivision interpolants . . . 111
B.3.1 Butterfly et Modified Butterfly . . . 111
B.3.2 Kobbelt . . . 111
B.3.3 Doo-Sabin et Midedge . . . 111

Annexe C Calculs parallèles **113**
C.1 Classification des environnements parallèles . . . 113
C.1.1 Single Instruction Single Data . . . 113
C.1.2 Single Instruction Multiple Data . . . 113
C.1.3 Multiple Instruction Single Data . . . 113
C.1.4 Multiple Instruction Multiple Data . . . 114
C.2 Évaluation d'un algorithme parallèle . . . 114
C.2.1 Accélération . . . 114
C.2.2 Efficacité . . . 114
C.2.3 Capacité d'évolution . . . 114
C.2.4 Granularité et parallélisme . . . 115
C.2.5 Répartition de charges . . . 115

Annexe D Rendu des grilles réservoirs sur des coupes sismiques quelconques **117**
D.1 Formulation du problème . . . 118
D.2 Calcul de l'intersection entre la grille réservoir et la coupe . . . 119
D.2.1 Partitionnement de l'espace : Octree . . . 119
D.2.2 Utilisation des tables indexée . . . 119
D.2.3 Intersection avec des *random lines* . . . 120
D.2.4 Traitement des grilles raffinées . . . 120
D.3 Conclusion . . . 121

Bibliographie **123**

Table des figures

1 Le modèle géologique 2
2 Acquisition sismique 2
3 Dépots sédimentaires déformés sous l'action des forces tectoniques 4
4 Mise en évidence d'un corps sur la sismique 5
5 Mise en évidence d'un horizon sur la sismique 6
6 Résultat d'une interprétation manuelle d'un horizon 6
7 Corrélation de traces sismiques 7
8 Deux étapes d'une propagation automatique d'un horizon dans un cube sismique 8
9 Interprétation manuel d'une faille 9
10 Détection automatique du réseau de failles 10
11 Interprétation manuel d'un corps 10
12 Reconstruction automatique d'un corps à partir d'un attribut sismique. 11

1.1 Résultats de l'interprétation 14
1.2 Exemple de modèle cohérent 14
1.3 Reconstruction d'une faille à partir d'un pointé 15
1.4 Méthode de reconstruction locale proposée par Boubekeur 17
1.5 Critère de la sphère vide 18
1.6 Triangulation contrainte 19
1.7 Triangulation de Delaunay contrainte 19
1.8 Intersection de contraintes dans $\mathbb{R}^3$ 20
1.9 Enveloppe convexe 21
1.10 Application des α-shapes sur un pointé de faille 21
1.11 Application d'un algorithme de lissage approximant sur une faille. 22
1.12 Arrangements nécessaires à la construction d'un réseau de failles cohérent 23
1.13 Traitement des failles s'intersectant 24
1.14 Étape de projection d'un faille sur une autre 25
1.15 Réseau de failles et son octree 25
1.16 Problème d'intersection 26
1.17 Illustration des notions de point le plus "proche" et de point "projeté" 27
1.18 À gauche, la faille d'origine. À droite, la faille d'origine (limitée en blanc) et la faille étendue 27
1.19 Intersection de deux failles infinies 28
1.20 Raffinement de l'octree pour s'adapter à la complexité de la faille 29
1.21 Interpolation des horizons 30

1.22 L'influence des points pour l'estimation du pixel bleu est modifiée par la présence des failles : les partie hachurées n'auront plus d'impact dans l'estimation (a) Vue en coupe (b) vue en carte 31
1.23 Construction du voisinage partiel : étape 1 33
1.24 Construction du voisinage partiel : étape 2 33
1.25 Construction du voisinage partiel : étape 3 33
1.26 Construction du voisinage partiel : étape 4 34
1.27 Sélection des triangles ayant une influence dans l'estimation d'un point 35
1.28 Extrapolation d'un bord avec les méthodes barycentriques 39
1.29 Extrapolation d'un bord avec le modèle plan. 41
1.30 Stratégie de remplissage à support statique 42
1.31 Stratégie de remplissage à support dynamique 43
1.32 Amélioration du calcul du plan d'estimation 44
1.33 Influence du critère de propagation 46
1.34 Nuages de points représentant des horizons interprétés 46
1.35 Horizon synthétique 47
1.36 Méthode barycentrique 48
1.37 Méthode barycentrique : vue en coupe 48
1.38 Méthode par régression : vue en coupe 49
1.39 Horizon réel 49
1.40 Méthode par régression 50
1.41 Instabilité de la méthode par régression 50
1.42 Méthode avec support dynamique 51
1.43 L'utilisation d'une stratégie de remplissage dynamique résout les problèmes de stabilité dans les zones bruitées 51
1.44 Horizon réel décimé 52
1.45 Résultat du krigeage 53
1.46 Construction d'un réseau composé de 35 failles 55
1.47 Interpolation d'un horizon bruité avec un réseau de failles complexe 56

2.1 Les différents faciès sismiques 59
2.2 Exemple d'attribut sismique : iso5 59
2.3 Reconstruction de surfaces à partir de l'attribut iso5 60
2.4 Bill Lorensen et Dick Bair 61
2.5 Méthodologie d'extraction des corps contrainte par le modèle structural 62
2.6 Définitions liées au volume 63
2.7 Configurations du marching cubes 64
2.8 Taxonomie des types de grilles 65
2.9 Configuration générant des surfaces non admissibles comme des variétés 66
2.10 Exemple d'artefacts de rendu dûs aux triangles dégénérés 66
2.11 Résolution des problèmes topologiques par ajout de bruit 67
2.12 Ambiguïté sur les faces : illustration en 2D 68
2.13 Ambiguïté sur les faces 68
2.14 Ambiguïté interne 69
2.15 Traitement des ambiguïtés (figure issue de [17]) 70
2.16 Extraction d'une isosurface à différentes résolutions 71
2.17 Gestion des cracks dans la reconstruction basée sur un octree 72
2.18 Raffinement de la surface lié au point de vue 73

2.19 Méthode par décimation . . . 74
2.20 Vertex clustering . . . 75
2.21 Vertex contraction . . . 76
2.22 Edge collapse . . . 76
2.23 Principe de la métrique d'erreur en 2D . . . 77
2.24 Surface reconstruite avec la méthode de Oudot et al . . . 78
2.25 Méthode de simplification de Wu et Kobbelt . . . 79
2.26 Formats décrivant les surfaces triangulées . . . 80
2.27 Principe d'extraction en couche . . . 81
2.28 Principe du *time lag* . . . 84
2.29 Influence du *time lag* . . . 85
2.30 Nombre de triangles générés pour chaque couche par le *marching cube* . . . 87
2.31 Distribution usuelle d'évènements géologiques . . . 88
2.32 Format de fichiers . . . 90
2.33 Partition binaire . . . 91
2.34 Effet du *time lag* étendu sur la qualité de la surface . . . 92
2.35 Extraction d'une isosurface sur un cube de dimensions $401 \times 1051 \times 2001$ . . . 97
2.36 Évolution du nombre de triangles . . . 99
2.37 Anisotropie . . . 99
2.38 Accélération . . . 100
2.39 Efficacité . . . 100
2.40 Isosurface non filtrée . . . 102
2.41 Isosurface filtrée . . . 102

A.1 Notion de paramétrisation . . . 105

B.1 Surfaces de subdivision . . . 109
B.2 Principe des subdivisions interpolantes . . . 110

D.1 Rendu d'une grille réservoir . . . 117
D.2 Rendu d'une grille réservoir sur une coupe sismique . . . 118
D.3 Rendu de grilles grâce à la méthode CIEL . . . 119
D.4 Exemple de random line . . . 120
D.5 Découpe des polygones . . . 120
D.6 Apparition de cracks de la cas de grilles raffinées . . . 121
D.7 Exemple de "random line" . . . 122

Introduction générale

1 Modélisation 3D en géologie

La modélisation du sous-sol est considérée comme stratégique par les compagnies pétrolières. En effet, elle a pour objectif de construire les modèles géologiques sur lesquels seront évalués les réserves de gaz et d'huile des territoires étudiés, ainsi que les placements optimaux des puits de forage exploitant ces réserves. La construction de ces modèles est faite sur l'interprétation de la sismique par les géologues et les géophysiciens. Dans cette optique, le niveau technique des outils d'interprétation a une influence directe sur la qualité des pronostiques qui seront établis. Ces logiciels ont pour vocation d'intégrer au sein d'un même repère de coordonnées 3D, les données sismiques, les interprétations des ingénieurs en géosciences et les modèles venant des différents métiers : géologie structurale et sédimentaire, interprétation sismique, géologie de production, géostatistique et simulation d'écoulements. En stockant toutes les informations et interprétations relatives à un gisement dans un même repère de coordonnées, ces outils permettent de les mettre en cohérence les unes avec les autres et de passer de façon continue du *modèle structural* au modèle de simulation réservoir.

Une phase essentielle du processus de modélisation consiste à construire une structure du sous-sol constituée d'un ensemble d'*horizons* (strates géologiques), de *failles* (fractures de la roches) et de *corps* (amas de roches de natures similaires). Afin de bien comprendre les problématiques liées à cette étape de modélisation nous allons dans premier temps nous intéresser à la nature des données que nous allons manipuler. Nous commencerons par l'image sismique qui contient une information volumique sur le sous-sol, puis nous décrirons les objets géologiques intervenant dans le modèle structural : les horizons, les failles et les corps. Ces objets décomposent le sous-sol en volumes comme l'illustre la figure 1.

Nous nous intéresserons ensuite aux outils qui permettent la réalisation de ce modèle décrivant le sous-sol. Dans un premier temps nous étudierons les méthodes permettant d'extraire l'information décrivant les horizons, les failles et les corps du bloc sismique. Nous verrons les techniques manuelles utilisées par les interprétateurs et décrirons les outils automatiques utilisés pour accélérer la construction de ces objets. Puis nous étudierons les solutions proposées par les géomodeleurs pour construire le modèle surfacique associé à ces objets géologiques.

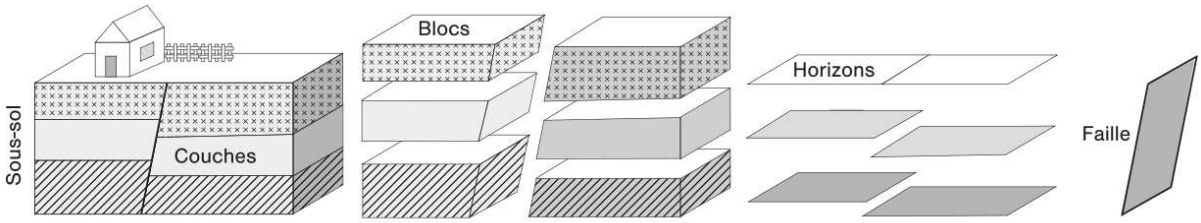


FIG. *1: Le modèle géologique*

1.1 Image sismique

Acquisition sismique

Afin de réaliser le modèle structural il faut avoir une vue du sous-sol. Pour obtenir une image du sous-sol, la technique la plus répandue est la *sismique de réflexion*. Elle consiste à émettre des ondes élastiques à partir d'un point source puis à mesurer les caractéristiques des ondes réfléchies en différents points récepteurs (voir figure 2).

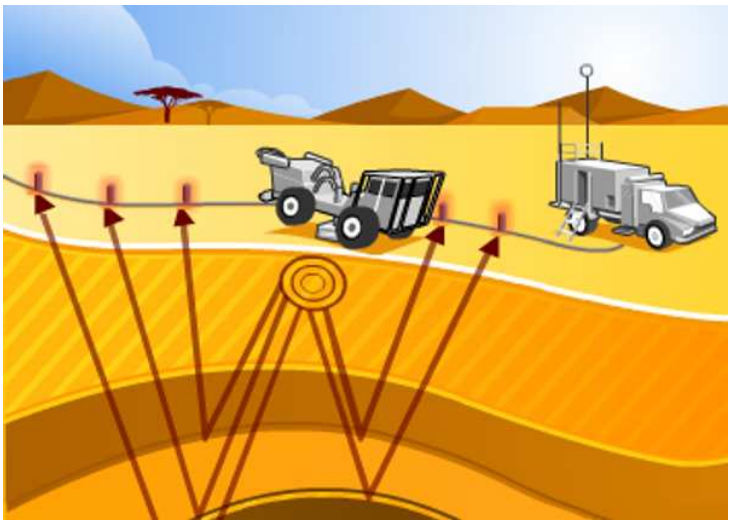


FIG. *2: Acquisition sismique*

L'objectif est de fabriquer, à partir des enregistrements faits en surface, l'image la plus fidèle possible de la réflectivité élastique du sous-sol. Cette image sera utilisée pour définir la géométrie des horizons, c'est-à-dire des évènements qui ont une continuité spatiale et qui sont caractéristiques d'évènements géologiques marquants. Les discontinuités dans l'image sismique permettront de déterminer la présence de failles.

Un autre aspect de l'étude de l'image sismique concerne la mesure quantitative de la réflectivité, c'est-à-dire l'estimation du coefficient de réflexion d'un évènement géologique donné. Ceci fournira

une information sur le type de roches ayant provoquées la réflectivité mesurée. Cette étude des réflectivités, en association ou non avec la géométrie permet de mettre en évidence certains objets géologiques.

Attribut sismique

Un attribut sismisque est une mesure quantitative d'une caractéristique de l'image sismique. L'analyse des propriétés de l'image pour l'interprétation des sismiques de réflexion se fait depuis les années 1930 quand les géophysiciens ont commencé à marquer manuellement les enregistrements sismiques pour convertir les temps de propagation en réflexion cohérente. On distingue désormais de très nombreux attributs sismiques utilisés notamment dans l'interprétation structurale, stratigraphique et dans l'étude des propriétés des roches et des fluides.

L'évolution des attributs sismiques est très liée aux avancées technologiques. Dans les années 1960, les progrès dans l'acquisition ont permis une meilleure mesure de l'amplitude du signal sismique et mis en évidence un lien entre la présence d'hydrocarbure dans la roche et les fortes amplitudes ("bright spots"). Dans les années 1970, l'apparition des imprimantes couleurs permettra de mieux nuancer les variations d'amplitude et de fréquence. L'apparition des stations d'interprétation dans les années 1980 permettra aux interprétateurs d'intégrer dans l'information sismique d'autres données telles que les logs issues des puits de forage. Aujourd'hui, la capacité de calcul des stations de travail permet le calcul et l'utilisation de nombreux attributs sur des images sismiques dont les dimensions et résolutions sont de plus en plus grandes.

Il existe aujourd'hui de très nombreux attributs, chaque compagnie pétrolière adaptant et créant ses propres besoins. Nous n'allons, pas dans cette thèse, faire l'étude des attributs sismiques, nous présenterons une application de certains d'entre eux dans le chapitre 2 lors de la construction d'un modèle géologique.

1.2 Horizon

Un horizon correspond à l'interface qui sépare deux couches ou milieux de natures différentes. Au cours du temps, les sédiments de différentes natures (sables, carbonates, sel, argiles, ...) se sont superposés. Ces couches peuvent avoir été déformées sous l'action de forces tectoniques, ce qui rend leur géométrie très compliquée (voir figure 3).

1.3 Faille

Une faille correspond à une cassure d'un ou plusieurs blocs avec déplacements relatifs des parties séparées. Ces blocs ont été soumis à différentes contraintes (cisaillement, compression, extension) et ont été décalés les uns par rapport aux autres.

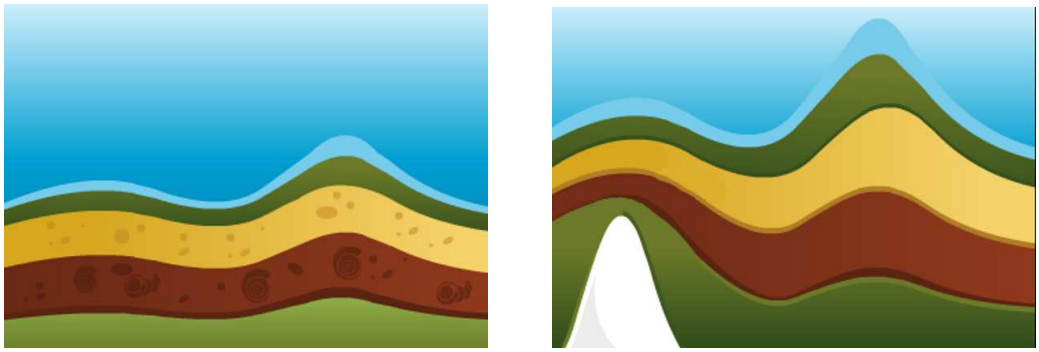
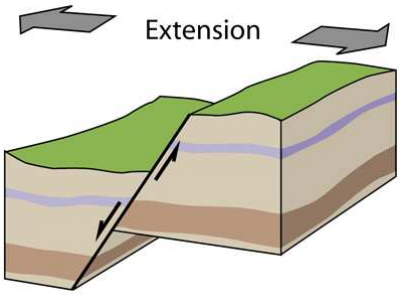


FIG. *3: Dépots sédimentaires déformés sous l'action des forces tectoniques*

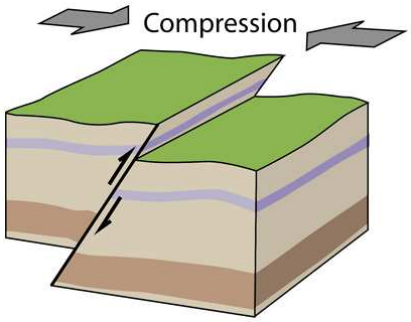
Faille normale



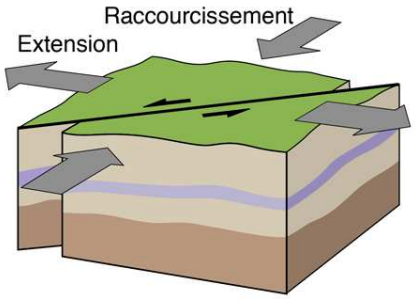
Une *faille normale* accompagne une extension ; le compartiment au dessus de la faille ("toit") descend par rapport au compartiment situé en-dessous de la faille ("mur"). La partie affaissée située entre deux failles normales à pendage opposé est appelée graben. La partie soulevée entre deux failles normales à pendage opposé est appelé *horst*. Nous appelons rejet d'une faille le décalage vertical qu'elle introduit entre le toit et le mur.

Faille inverse

Une *faille inverse*, ou *chevauchement* accompagne une compression ; le compartiment au dessus de la faille ("toit") monte par rapport au compartiment situé en-dessous de la faille ("mur").



Décrochement



Un *décrochement* accompagne un mouvement de coulissage ; les décrochements purs (faille verticale et déplacement horizontal) ne s'accompagnent d'aucun mouvement vertical. Les décrochements peuvent être dextres ou sénestres, suivant que le compartiment opposé à l'observateur se déplace vers la droite ou la gauche (respectivement).

1.4 Corps

Un *corps* est un objet fermé que le géologue ou le géophysicien considère comme entourant une zone du sous-sol aux propriétés similaires. Ces zones peuvent limiter des roches aux propriétés pétrophysiques identiques, par exemple délimiter un chenal ou un réservoir d'huile.

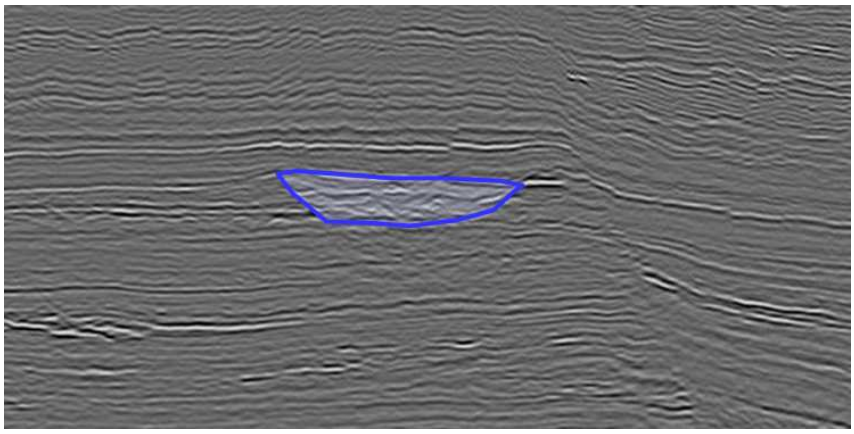


FIG. *4: Mise en évidence (en bleu) d'un corps sur la sismique*

Dans la figure 4 nous avons mis en évidence (en bleu) une zone chaotique. Dans certains contextes, cette zone pourrait être assimilée à la présence de roches contenant de l'huile.

2 Outils d'aide à l'interprétation

L'étape d'interprétation est primordiale dans la compréhension du sous-sol mais reste souvent laborieuse. C'est pourquoi les stations d'interprétation proposent de nombreux outils facilitant cette étape. Nous allons distinguer deux catégories d'outils d'aide à l'interprétation : les *outils manuels* qui nécessitent une intervention quasi-permanente de l'interprétateur et les *outils automatiques* qui identifient les objets géologiques sans interventions extérieures.

2.1 Construction des Horizons

Les interfaces entre les couches géologiques correspondent en général à des contrastes dans l'image sismique. La stratigraphie, les plis, les déformations, les inconformités apparaissent donc nettement sur les images sismiques. On pourra alors proposer des outils permettant la détection de ces objets de manière automatique. Néanmoins l'interprétation manuelle reste nécessaire dans certains cas.

Outils manuels de construction d'horizon

Historiquement l'interprétation des horizons est faite "à la main". Le géophysicien parcours la sismique grâce à des vues en coupes du volume sismique et marque à l'aide de points ou de

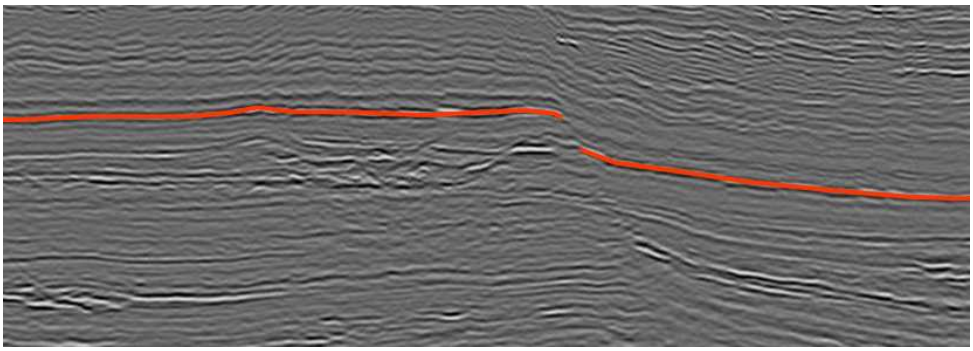


FIG. *5: Vue en coupe d'une image sismique. Dans cette image, un horizon a été pointé (en rouge) par l'interprétateur. Le résultat d'un pointé de cet horizon sur de nombreuses coupes sismiques donnera un nuage de points qui servira de base à la reconstruction de la surface.*

lignes brisée la position des horizons sur celle-ci (voir figure 5). Actuellement cette technique n'est utilisée que dans les cas où les outils automatiques ne fonctionnent pas, ce qui est en général le cas quand la sismique est de mauvaise qualité. Il est utile de préciser que la présence de failles engendre de la sismique bruitée à proximité de celle-ci, en effet les failles piègent les ondes élastiques ce qui génère un effet de flou.

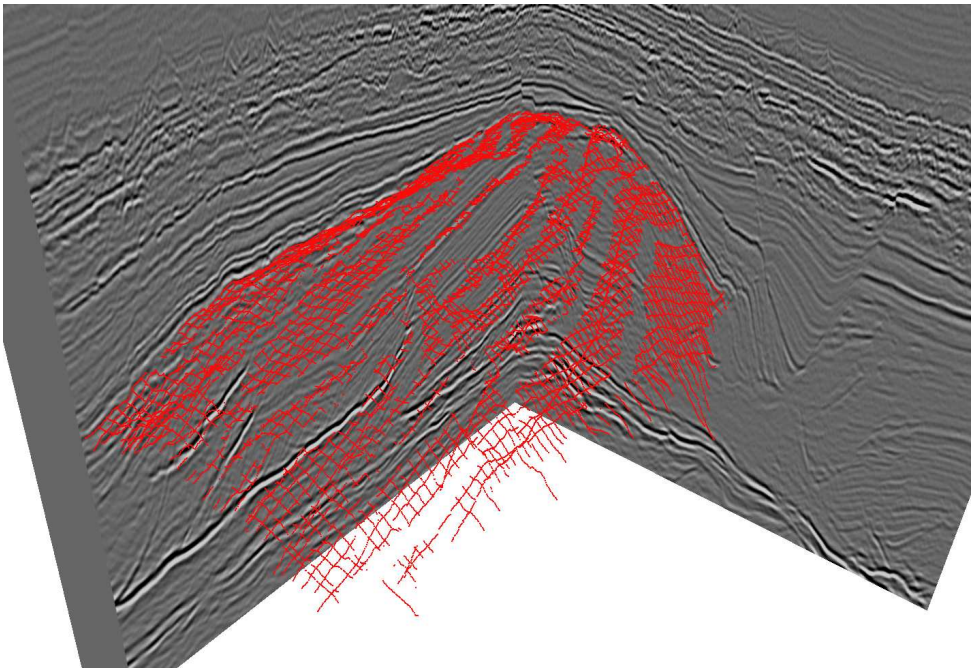


FIG. *6: Résultat d'une interprétation manuelle d'un horizon*

L'interprétation manuelle est aussi nécessaire en présence de failles qui partagent entièrement le volume sismique. En effet, elles peuvent générer des décrochements que l'outil automatique ne pourra pas traiter.

La figure 6 illustre le résultat d'une interprétation manuel. Nous pouvons constater que le résultat est un nuage de points éparses. Il faut donc proposer des techniques d'interpolations qui permettent de traiter ce type de données.

Outil automatique : le propagateur d'horizon

Dans le monde des logiciels d'interprétation sismique, l'une des méthodes d'extraction automatique d'horizon les plus reconnues est le *propagateur* proposé par SISMAGE©. Le propagateur, protégé par un brevet [2] depuis 1998, est décris dans [30] :

1. Un germe de départ G_0 est placé sur une trace. Il est constitué du point sélectionné P_0 et d'une fenêtre verticale F_0 qui définit le nombre de points de la trace utilisés pour le calcul de la corrélation. Sur la figure 7 (a), le curseur de pointé permet de visualiser la fenêtre et le point sélectionné. Le point central du germe est automatiquement associé à l'extrema local le plus proche (le type d'extremum, positif ou négatif, est défini par l'utilisateur). La trace sur laquelle le germe a été posé est définie comme trace courante T_k, k étant le niveau de propagation par rapport au germe initial (dans ce premier cas, k = 0).

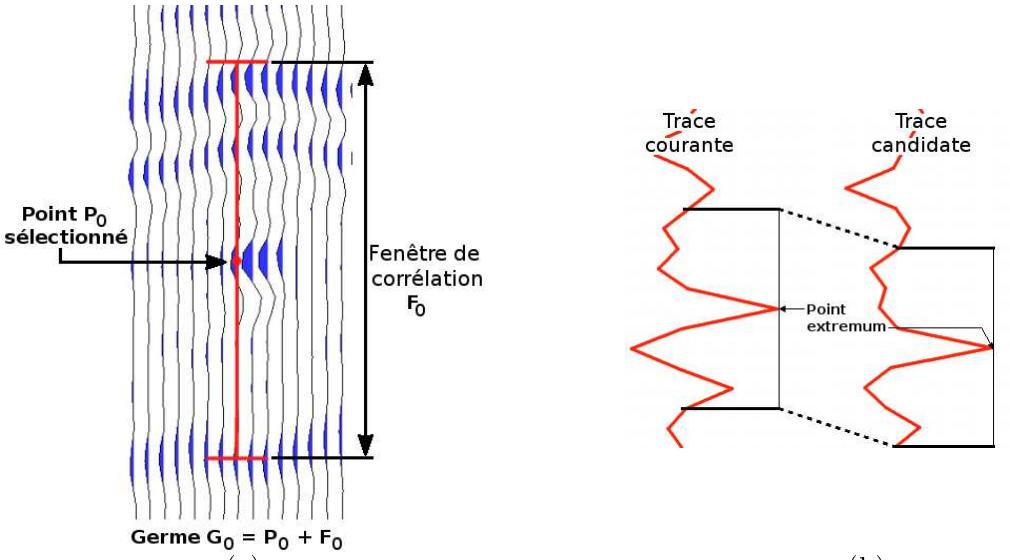


FIG. 7: *(a) Point et sa fenêtre de corrélation (b) Corrélation entre deux germes*

2. Les traces voisines T_{k+1} à la courante T_k non traitées sont sélectionnées.
3. Pour chaque trace T_{k+1} sélectionnée précédemment, nous recherchons le point qui correspond à l'extremum du germe défini sur la trace courante T_k afin de créer un nouveau germe candidat (voir figure 7 (b)). Puis une valeur de corrélation $C_{0,k+1}$ est estimée entre le germe candidat et le germe de départ.
4. Si la valeur de corrélation est supérieure à une valeur seuil, le germe candidat est ajouté à une pile triée par valeur décroissante de corrélation et l'extremum du germe traité est

intégré à l'horizon en construction. Puis le germe situé dans le haut de la pile (i.e. le germe a la corrélation la plus élevée) est dépilé et sa trace associée est définie en tant que trace courante.

5. L'algorithme recommence à l'étape 2 tant que la pile n'est pas vide, c'est-à-dire que tous ses éléments ont été traités.

La figure 8 montre deux étapes d'une propagation d'un horizon dans un cube sismique. Dans ces figures, le cube n'est affiché que partiellement pour des besoins de visualisation.

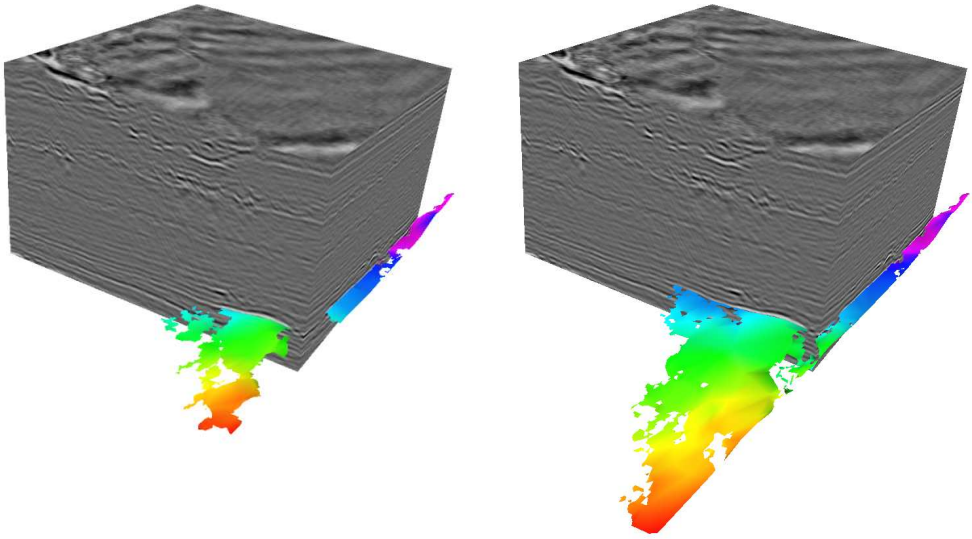


FIG. *8: Deux étapes d'une propagation automatique d'un horizon dans un cube sismique*

Il est aussi possible d'initialiser la propagation avec tous ou une partie des points de contour d'un horizon déjà existant. Ceci permet de travailler de manière incrémentale ou de propager un horizon pointé à la main. Parmi les extensions proposées dans [30] les horizons peuvent être propagés en se servant de plusieurs images sismiques.

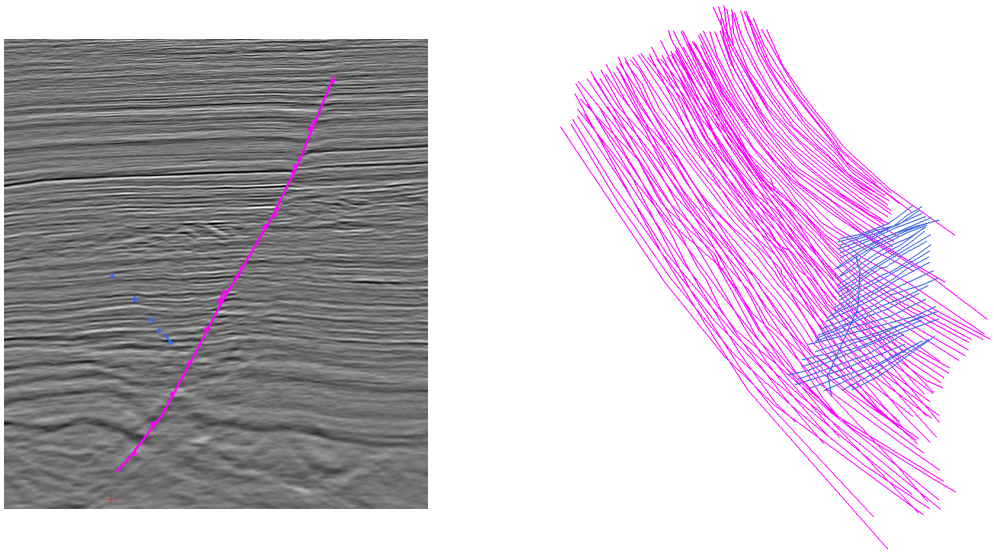
2.2 Construction des failles

Contrairement aux horizons, les failles ne constituent pas des réflecteurs à proprement parler. Elles sont donc visibles à travers des décalages importants entre les horizons, ou par la présence d'une zone de bruit quasi verticale et plane.

Interprétation manuelle

L'interprétation manuelle reste encore aujourd'hui la technique la plus utilisée pour extraire la description des failles dans l'image sismique. Afin de reconstuire la surface décrivant la faille, le géophysicien parcourt le volume sismique suivant de nombreuses directions à la recherche de

discontinuités, il les marque grâce à des lignes brisées dans sa vue (voir figure 9 (a)). La faille est alors décrite par un nuage de lignes brisées (voir figure 9 (b)).



(a) (b)

FIG. *9: Interprétation d'une faille. Dans ces deux images, on distingue le pointé de deux failles, la rose et la bleue. (a) Illustration du pointé d'une faille sur l'image sismique. Les lignes continues ont été marquées sur la coupe courante, les croix sont des intersections entre la coupe courantes et des lignes marquées avec des angles de vue différents. (b) Illustration du nuage de lignes brisées en trois dimensions.*

Outils automatiques

On trouve dans la littérature quelques méthodes permettant de reconstruire de manière automatique le réseau de failles à partir de l'image sismique. SISMAGE© est doté d'un tel outil. La plus grande difficulté rencontrée par les outils de reconstruction automatique est de détecter les discontinuités et de placer les marqueurs définissant les failles aux bons endroits malgré le flou dû au bruit.

La recherche reste active pour la mise au point d'un attribut fiable permettant la détection de faille sans introduire d'artefact.

2.3 Construction des corps

De la même façon que pour les horizons et les failles, les corps étaient à l'origine interprétés "à la main". Avec l'apparition des attributs sismiques qui permettent de mettre en évidence certaines propriétés géologiques sur l'image sismique, certains types de corps peuvent être reconstruits de manière automatique.

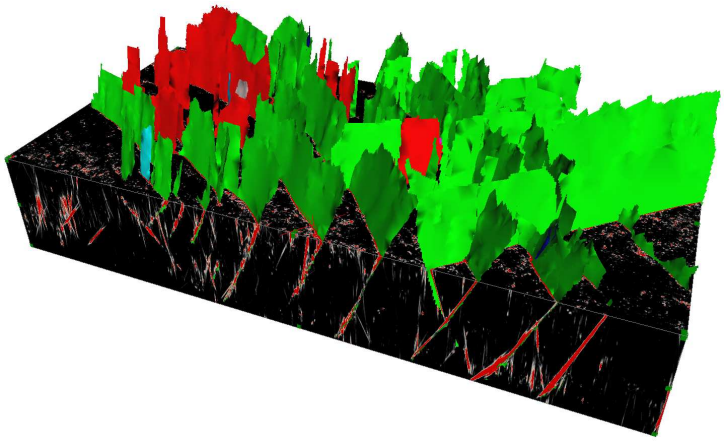


FIG. *10: Détection automatique du réseau de failles*

Interprétation manuelle

Comme pour l'interprétation manuelle des failles, l'interprétateur parcourt le volume sismique dans plusieurs direction afin d'avoir des angles de vue différents. Les corps sont marqués sur une coupe sismique par un ou plusieurs polygones. La figure 11 illustre les données issues de l'interprétation (a) et le résultat attendu (b).

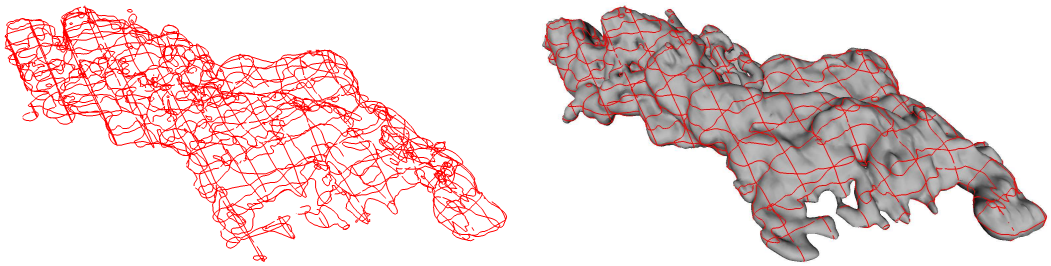


FIG. *11: Interprétation d'un corps. A gauche le pointé manuel. A droite la surface reconstruite à partir de ce pointé.*

La reconstruction de surfaces fermées à partir de polygones appartenant à des plans parallèles est un problème largement abordé par la littérature [7, 10, 29, 51, 66, 86]. Ce n'est que depuis l'année dernière que des recherches ont porté sur la reconstruction de surfaces à partir de polygones appartenant à des plans quelconques. Nous n'aborderons pas cette problématique dans cette thèse mais le lecteur curieux pourra se référer à [11, 61].

Outils automatiques

L'étude de l'image sismique par le biais des attributs sismiques peut mettre en évidence certaines propriétés physiques de la zone étudiée. Il peut donc être intéressant de reconstruire les enveloppes de certaines valeurs physiques que l'on a associé à des matériaux. On pourra ainsi,

suivant les attributs, reconstruire des corps géologiques de manière automatique en précisant une plage de valeurs physiques.

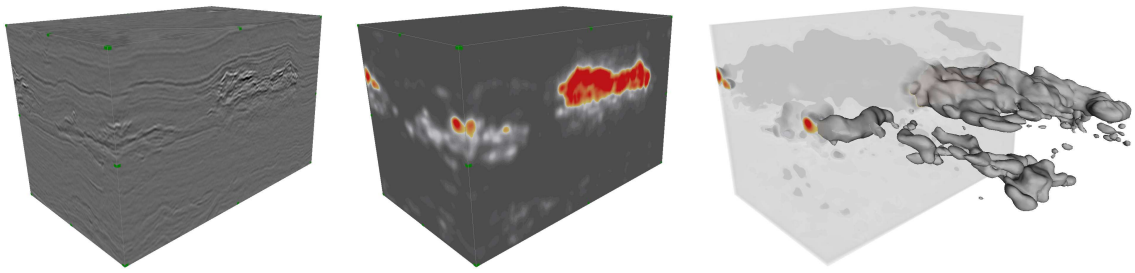


FIG. *12: Reconstruction automatique d'un corps à partir d'un attribut sismique.*

La figure 12 illustre la méthodologie utilisée pour reconstruire un corps automatiquement à partir de la sismique. Les techniques utilisées pour calculer les surfaces enveloppant les valeurs choisies seront détaillées dans le chapitre 2.

3 Organisation de la thèse

Ce mémoire est organisé en deux parties. Le premier chapitre est consacré à la construction du modèle structural comprenant les horizons et les failles. Ce modèle est bâtit à partir de l'interprétation de la sismique par le géophysisien et le géologue. Le second est consacré à la reconstruction de corps géologiques à partir d'attributs sismiques.

Chapitre 1 : Construction et optimisation du modèle structural

Nous proposons dans ce chapitre une étude des outils de modélisation et de leurs applications dans la construction du modèle structural à partir de l'interprétation : nuage de points ou de lignes brisées. Nous proposons une méthologie complète permettant de bâtir un modèle cohérent tenant compte de tous les objets géologiques. La première étape dans la construction du modèle profite de la nature quasi-plane des failles pour reconstruire une surface triangulée grossière qui sera lissée en utilisant les surfaces de subdivision. Chaque faille reconstruite indépendamment sera ensuite plongé dans une scène 3D organisée avec un octree afin de rendre le réseau cohérent géologiquement. La dernière étape consiste à boucher les trous dans les surfaces modélisant les horizons grâce aux méthodes d'interpolation spatiale issues du traitement d'image ou des géostatistiques. Ces méthodes ont été étendues par nos soins afin de prendre en compte le réseau de faille construit dans l'étape précédente.

Chapitre 2 : Extraction des corps dans les cubes attributs

Ce chapitre est consacré à la reconstruction des corps dans les cubes attributs par le bias des isosurfaces. La première partie est un état de l'art des méthodes courantes d'extraction de surfaces

et plus particulièrement de la littérature traitant des *marching cubes*. La taille des données manipulées dans le cadre des géosciences nous a forcé à proposer un algorithme d'extraction d'isosurfaces simplifiées s'exécutant sur des machines parallèles.

Chapitre 1

Construction et optimisation du modèle structural

Sommaire

1.1 Introduction 14
1.2 Triangulation des failles 15
1.2.1 Triangulation du nuage de lignes brisées projeté 17
1.2.2 Calcul de l'enveloppe de la faille 20
1.2.3 Raffinement de la surface 22
1.3 Arrangement du réseau de failles 22
1.3.1 Approche surfacique 23
1.3.2 Approche volumique 26
1.4 Interpolation des horizons 29
1.4.1 Notion de voisinage 30
1.4.2 Estimateurs 35
1.4.3 Étude de l'ordre de propagation dans la stratégie de remplissage 45
1.4.4 Application des méthodes d'interpolation contraintes par les failles . . . 46
1.5 Conclusion 54

1.1 Introduction

Comme cela a été présenté dans l'introduction générale, la modélisation géologique du sous-sol passe par l'interprétation des interfaces géologiques, à savoir les horizons et les failles. Nous avons vu que ces surfaces prennent la forme soit de semis de points, soit de nuages de lignes brisées (voir figure 1.1).

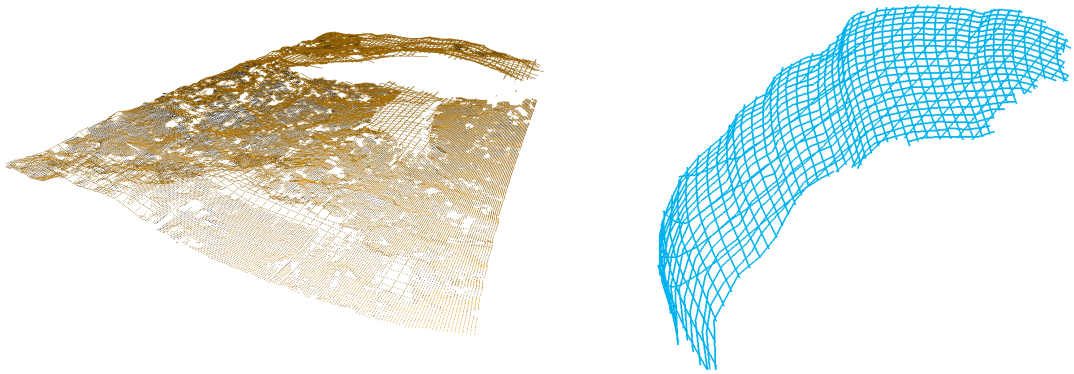


FIG. *1.1: Résultats de l'interprétation : à gauche un semis de point représentant un horizon et à droite un nuage de lignes brisées représentant une faille*

La construction du modèle structural consiste à la fois à interpoler ces surfaces interprétées, et à mettre en cohérence les surfaces entre elles : c'est-à-dire assurer les contacts entre les surfaces de manière à ce qu'elles ne se traversent pas et viennent au contact les unes des autres de manière à former des compartiments dits *étanches*. La figure 1.2 montre un exemple de mise en cohérence du modèle sur une section sismique, cette mise en cohérence du modèle devant être réalisée en trois dimensions.

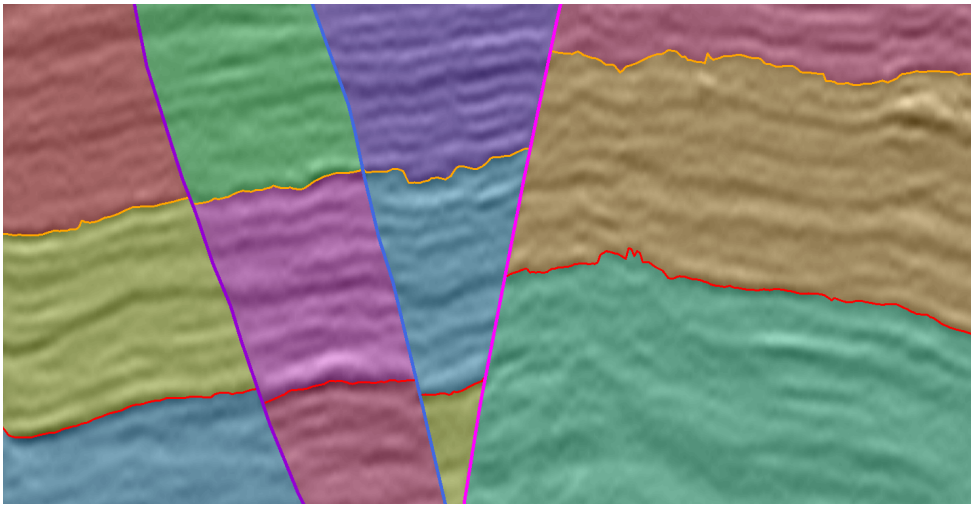


FIG. *1.2: Vue en coupe d'un modèle cohérent : les horizons et les failles créent des compartiments étanches*

Classiquement la construction du modèle structural est réalisée en trois étapes. Dans un premier temps chaque faille est triangulée indépendemment, puis le réseau de failles est mis en cohérence (ajustement des contacts entre surfaces). Enfin, les horizons sont interpolés en tenant compte des failles, ce qui permet de respecter le décalage introduit par la rupture de la roche lors des mouvements tectoniques. Nous détaillons dans la suite de ce chapitre le contenu de ces trois étapes.

1.2 Triangulation des failles

La première étape du processus de construction du modèle structural consiste donc à interpoler chaque pointé de faille afin d'obtenir un ensemble de surfaces respectant ces pointés. La figure 1.3 présente un exemple d'interpolation attendue à partir d'un ensemble de lignes brisées en trois dimensions.

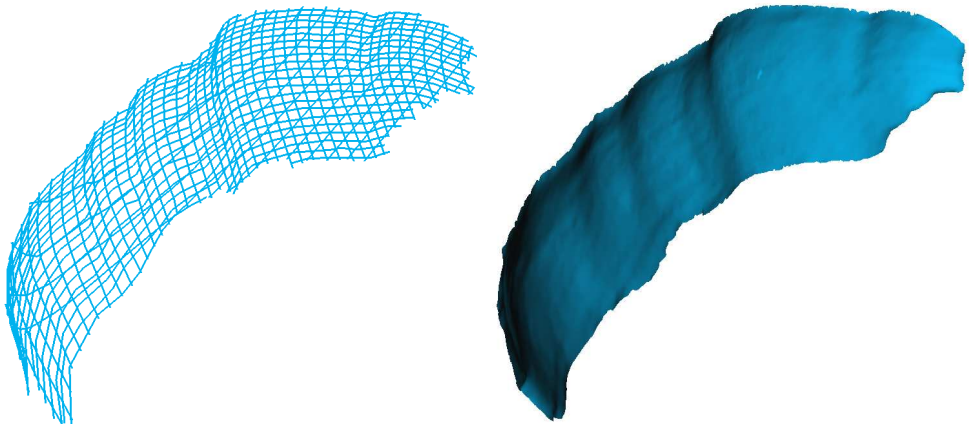


FIG. *1.3: Reconstruction d'une faille à partir d'un pointé. Le résultat d'un pointé est un nuage de lignes brisées (à gauche). La surface modélisant la faille (à droite) doit respecter ce nuage.*

Dans cette optique, nous allons étudier les outils de modélisation de surfaces utilisant des approches en trois dimensions (le nuage de lignes brisées est défini dans $\mathbb{R}^3$). Parmi ces approches, nous pouvons différencier deux catégories : les approches utilisant des représentations paramétriques et celles utilisant des représentations polygonales. Dans cette thèse nous ne nous intéresserons pas en détail aux méthodes paramétriques car, en pratique, il n'est pas toujours possible de trouver le paramétrage adéquat pour des nuages de lignes brisées qui sont des données irrégulièrement réparties. Une description de ces méthodes est disponible dans l'annexe A. Nous consacrons donc la suite de cette section aux techniques de modélisation par des surfaces polygonales.

La représentation polygonale la plus commune et la plus simple à manipuler est la surface triangulée où chaque facette de la surface est un triangle. L'avantage d'utiliser des surfaces

triangulées est que leurs facettes sont toujours planes dans $\mathbb{R}^3$ et sont toujours convexes. Le résultat de l'interprétation des failles est un nuage de lignes brisées, il faut donc reconstruire la surface triangulée respectant ce nuage.

La problématique est proche de la reconstruction d'une surface triangulée à partir d'un nuage de points. Hrádek [47] propose une classification des méthodes utilisées pour reconstruire la surface et distingue quatre catégories :

- les méthodes sculpturales : à partir de la triangulation 3D (pavage de l'espace par des tétraèdres) du nuage de points, la méthode de reconstruction sélectionne un sous-ensemble de triangles pour représenter la surface [3, 4, 5, 10, 21, 22, 28].
- les méthodes volumiques : une première étape consiste à calculer un volume de données scalaires dont l'extraction d'une isosurface, dans une seconde étape, approxime la surface recherchée [43, 44, 45].
- les méthodes par déformation : ces méthodes déforment une approximation initiale de la surface pour l'adapter à la morphologie des points [49, 69, 81].
- les méthodes par propagation : ces méthodes créent la surface en la faisant croître le long de ces bords à partir d'un élément (point, arête, triangle) initial [19, 79].

Ces approches présentent des mécaniques entièrement en trois dimensions pour traiter des nuages de points arbitraires qui peuvent paraître coûteuses. Dans ces travaux, Boubekeur [13] propose une approche originale. Il décompose le nuage de points en groupes de points quasi plan. Il fait alors une reconstruction locale en deux dimensions de ce nuage (voir figure 1.4) en utilisant la méthodologie suivante :

- calculer le plan de régression,
- projeter le nuage de points sur ce plan,
- calculer la triangulation de Delaunay en 2D sur les points obtenus,
- ramener les triangles obtenus dans l'espace 3D,
- et appliquer un algorithme de lissage sur la surface obtenue.

Les failles étant par nature quasi-planaires, l'approche proposée par Boubekeur pour reconstruire localement la surface nous semble mieux convenir à notre problème. Il faudra cependant adapter la méthodologie pour les raisons suivantes :

- nos données d'entrée sont des lignes brisées et non des points, nous devons donc introduire les contraintes de segments dans la triangulation.
- la triangulation dans le cas des failles n'est pas forcement convexe, il faut donc introduire une extraction de contour adaptée aux données.
- l'étape de lissage induisant un écart aux données, nous proposons d'étudier les différentes techniques de subdivision pour mesurer leur impact sur la surface lissée.

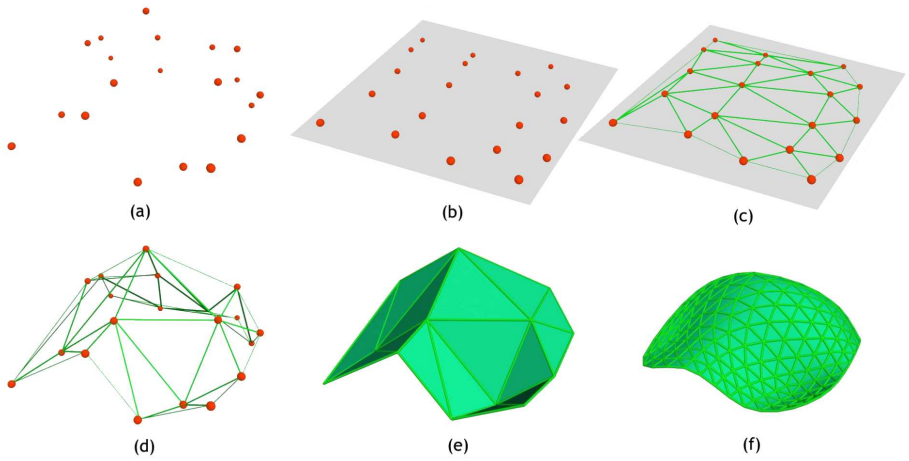


FIG. *1.4: Méthode de reconstruction locale utilisée par Boubekeur dans [13]. (a) Le nuage de points en entrée. (b) Projection du nuage de points dans son plan de régression. (c) Triangulation de Delaunay. (d) Retour de la triangulation en 3D. (e) Surface issue du nuage de points. (f) Application d'un algorithme de subdivision afin de lisser la surface.*

1.2.1 Triangulation du nuage de lignes brisées projeté

Dans ce chapitre, nous allons définir les triangulations et plus particulièrement les triangulations de Delaunay qui sont des structures couramment utilisées, notamment dans le cadre de la géométrie algorithmique. Pour notre application, nous nous intéressons aux triangulations dans $\mathbb{R}^2$ mais nous donnerons dans certains cas les définitions générales. Une description plus approfondie des triangulations est disponible dans [35].

Définition d'une triangulation

Soit S un ensemble de points de $\mathbb{R}^2$, l'enveloppe convexe de S, notée $conv(S)$, définit un domaine Ω de $\mathbb{R}^2$. Si t désigne un triangle alors τ est une triangulation (conforme) de Ω si :

1. l'ensemble des sommets des triangles de τ est exactement S,
2. $\Omega = \bigcup_{t \in \tau} t$,
3. tout triangle t de τ est d'intérieur non vide,
4. l'intersection de deux éléments de τ est soit : l'ensemble vide, un sommet ou une arête.

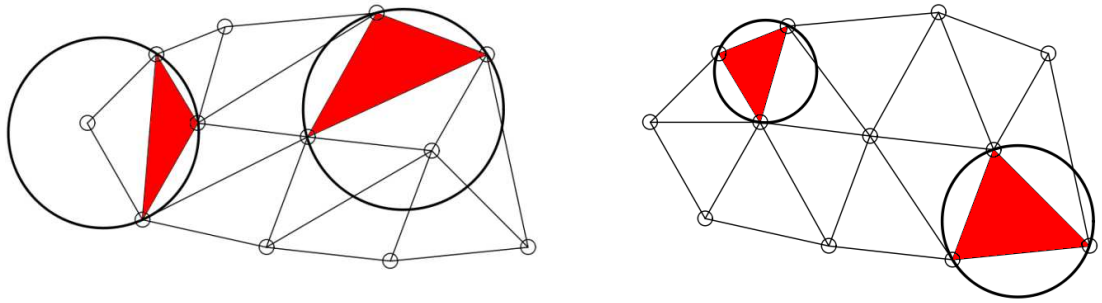
Triangulation de Delaunay

À la fin du 19^{e} siècle, Dirichlet a montré qu'à partir d'un nuage de points de $\mathbb{R}^2$, on peut partitionner le plan en cellules convexes en se basant sur des critères de proximité. C'est au début du 20^{e} siècle que Voronoï étend les résultats de Dirichlet à des espaces d'ordres supérieurs. La

notion de diagramme de Voronoï, ensemble de cellules traduisant cette notion de proximité pour un ensemble de points, est alors introduite.

Dans les années 30, Delaunay montre qu'à partir des diagrammes de Voronoï, on peut déduire par dualité une triangulation. Il a établie un certain nombre de résultats relatifs à cette triangulation tel que le *critère de la sphère vide*. C'est pourquoi elle porte le nom de triangulation de Delaunay.

Critère de la sphère vide Ce critère indique que les boules ouvertes associées aux éléments de la triangulation (les cercles circonscrits des triangles en dimension deux) ne contiennent aucun sommet. La figure 1.5 illustre l'application de ce critère en dimension deux.



(a) (b)

FIG. *1.5: Critère de la sphère vide. A gauche les triangles en rouge ne respectent pas le critère, il y a un sommet dans leur cercle circonscrit. À droite les triangles en rouge respectent le critère.*

Lemme général de Delaunay Delaunay a montré le résultat suivant, qui est à la base de la plupart des algorithmes de construction de triangulations dites de Delaunay :

Si τ est une triangulation quelconque de l'enveloppe convexe d'un nuage de points S, alors si la propriété de la sphère vide est vérifiée pour toute configuration de deux triangles adjacents de τ, elle est vérifiée globalement et τ est une triangulation de Delaunay.

Comme nous l'avons précisé au début de la section, nous ne travaillons pas sur un nuage de points mais sur des lignes brisées. Dans le plan, ces lignes seront traduites par des segments. Il faut alors s'assurer que ces segments soient des arêtes de la triangulation. On parle alors de triangulation contrainte.

Triangulation contrainte Notons C un ensemble de contraintes. Une triangulation τ est contrainte si toutes les contraintes C existent en temps qu'entités de τ tel quel ou sous la forme d'une partition.

Dans notre cas, les contraintes seront les segments composant les lignes brisées. Chaque segment apparaitra dans la triangulation sous la forme d'une ou plusieurs arêtes. La figure 1.6 présente le résultat d'une triangulation contrainte.

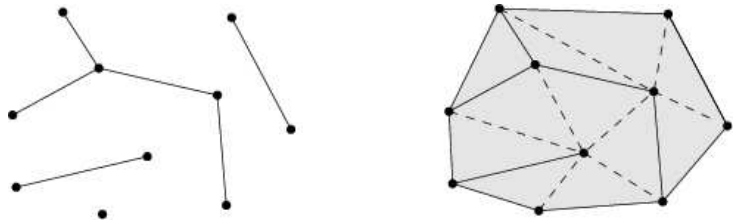


FIG. *1.6: Exemple de triangulation contrainte. Les segments contraints (à gauche) sont des arêtes de la triangulation (à droite)*

Triangulation de Delaunay contrainte Le problème est de construire une triangulation de Delaunay respectant cet ensemble de contraintes. Malheureusement, la triangulation de Delaunay est unique et si les contraintes n'y apparaissent pas toutes, construire cette triangulation sera impossible. Il faut donc adapter le critère de la sphère vide au cas contraint.

Un point est dit *visible* d'un autre point de la triangulation si le segment les joignant ne coupe aucune arête contrainte.

Le critère de la sphère vide adapté au cas contraint indique que les boules ouvertes associées aux éléments de la triangulation ne contiennent aucun sommets visibles.

On appelle triangulation de Delaunay contrainte une triangulation pour laquelle tous les triangles vérifient le critère de la sphère vide adapté au cas contraint (voir figure 1.7).

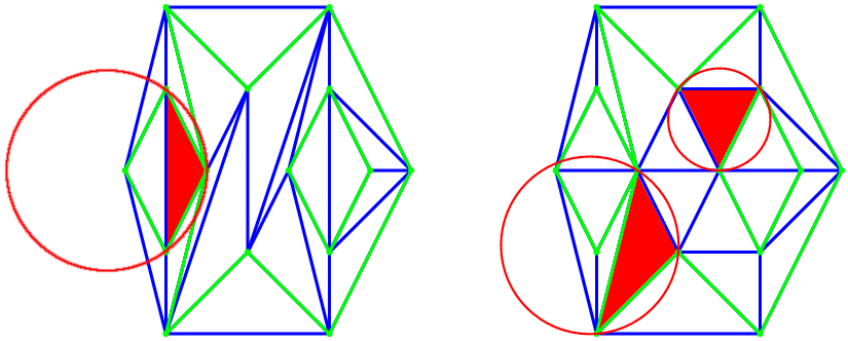


FIG. *1.7: Exemple de triangulation de Delaunay contrainte. Les contraintes sont dessinées en vert. A gauche la triangulation contrainte n'est pas de Delaunay. A droite, la triangulation est dite de Delaunay contrainte car le critère de sphère vide adapté (en rouge) est vérifié pour chaque triangle.*

Gestion des intersections des contraintes Dans une triangulation en deux dimensions, les contraintes sont modélisées par des segments. Si deux contraintes s'intersectent, elles vont alors

introduire un nouveau point dans la triangulation, ce nouveau point étant positionné à l'intersection des deux contraintes. Dans notre cas les segments sont définis dans $\mathbb{R}^3$ mais la triangulation est faite dans $\mathbb{R}^2$ car nous travaillons sur les segments projetés dans le plan de regression. Or deux segments projetés peuvent s'intersecter dans le plan alors que leurs équivalents en trois dimensions ne s'intersectent pas (voir figure 1.8). Dans ce cas, nous calculons la valeur dans $\mathbb{R}^3$ du point d'intersection par une combinaison des points définissant les segments pondérés par leur distance au point dans le plan.

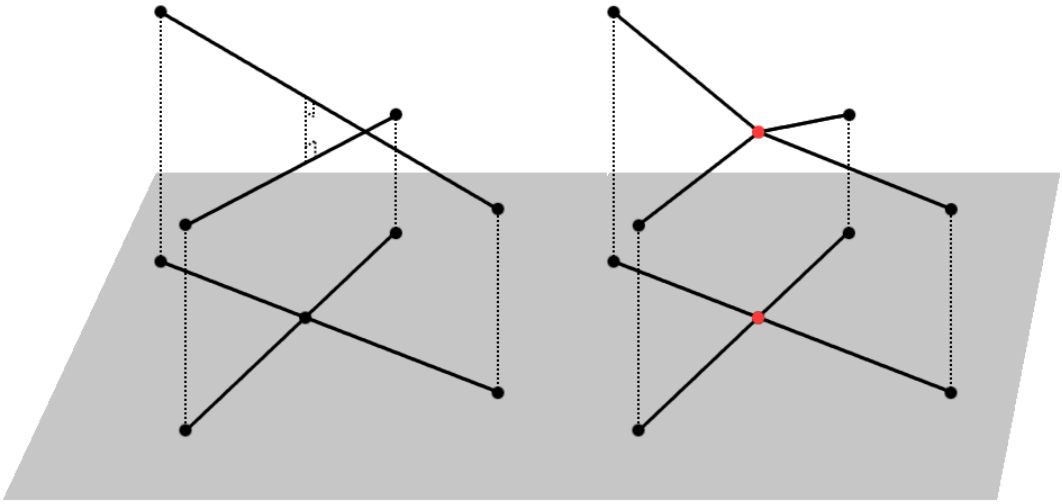


FIG. *1.8: À gauche, un pointé de faille dont les segments projetés s'intersectent alors que les segments originaux ne s'intersectent pas. À droite, le point issue de l'intersection des segments projetés est ajouté dans les lignes brisées du pointé originel.*

1.2.2 Calcul de l'enveloppe de la faille

Nous avons vu que par définition une triangulation est définie dans l'enveloppe convexe du nuage de points. Dans le cas des failles, les surfaces les modélisant ne sont pas obligatoirement convexes. La figure 1.9 présente un exemple de pointé projeté dans son plan de régression et son enveloppe convexe.

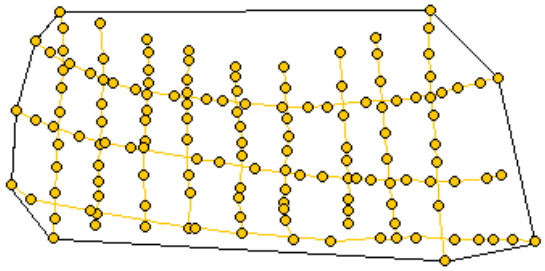


FIG. *1.9: L'enveloppe convexe dans laquelle est définie la triangulation ne correspond pas à l'enveloppe réelle de la faille*

Afin de calculer une meilleur approximation de l'enveloppe réelle de la faille, nous utilisons les α-shapes qui sont une généralisation du concept de l'enveloppe convexe d'un ensemble de points dans l'espace. Une bonne définition des α-shapes et de leurs utilisations est disponible dans la thèse de Da [20]. Nous proposons ici une définition intuitive des α-shapes :

considèrons un ensemble P de points, et un réel α tel que $0 \leq \alpha \leq \infty$, on fait rouler une boule b_α de rayon α, sur le nuage de points. b_α s'appuie sur deux points déterminant une arête unique. Cette arête fait donc partie de l'α-shape. Ensuite on fait pivoter b_α sur un des points de l'arête jusqu'à ce qu'elle soit bloquée par un autre point du nuage de point. L'α-shape de P n'est ni convexe ni en un seul morceau. Pour $\alpha = \infty$, l'α-shape est l'enveloppe convexe de P. Lorsque α diminue, l'α-shape fond en formant des cavités. Ces cavités peuvent se rejoindre et former des tunnels ou des trous.

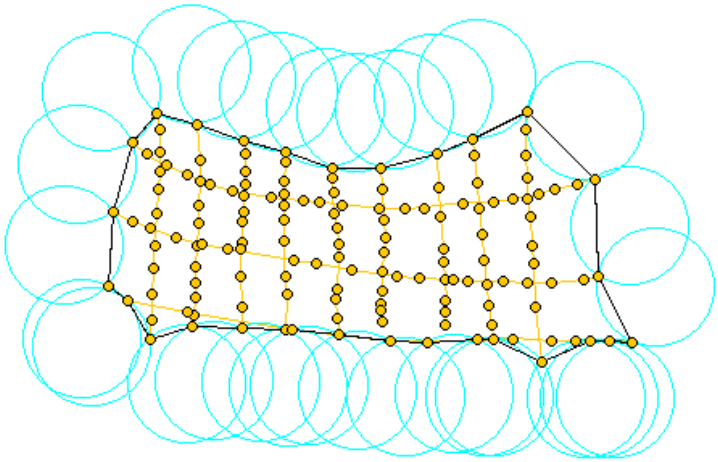


FIG. *1.10: Application des α-shapes sur un pointé projeté dans son plan de régression. Les cercles en bleu modélisent la boule qui roule sur le nuage de points afin de calculer l'enveloppe.*

La figure 1.10 illustre l'application des α-shapes sur un nuage de points issu de la projection du pointé d'une faille sur son plan de régression. On peut déduire une nouvelle enveloppe de la

faille modélisée par un ensemble de segments. Ces segments seront ajoutés dans la triangulation contrainte de la faille calculée dans l'étape précédente en tant que contraintes. Seuls les triangles de la triangulation contrainte appartenant à ce nouveau contour seront conservés dans la faille triangulée.
En pratique nous cherchons la valeur α la plus petite qui génère un seul composant sans trou. Actuellement cette valeur de α est donnée par l'utilisateur. Nous pourrions faire une recherche dichotomique pour identifier cette valeur.

1.2.3 Raffinement de la surface

Ramenée dans $\mathbb{R}^3$, la triangulation du nuage de lignes brisées génère une surface triangulée grossière non lisse. Nous allons utiliser une méthode de subdivision pour la raffiner. Une description des différentes méthodes ainsi que leurs propriétés est disponible dans l'annexe B. Parmi les schémas décrits dans l'annexe, nous ne nous intéressons qu'à ceux s'appliquant aux surfaces triangulées, c'est-à-dire les schémas de Loop [62], $\sqrt{3}$ [53], Butterfly [26] et Modified Butterfly [90].

Ces méthodes peuvent être classées en deux catégories : les méthodes approximantes (Loop et $\sqrt{3}$) et les méthodes interpolantes (Butterfly et Modified Butterfly). On dit qu'un schéma est interpolant si les sommets présents dans la surface polyédrale originelle sont encore des sommets de la surface résultat ; il est approximant sinon. En pratique, nous laissons le choix à l'utilisateur en fonction de la qualité de son pointé : il peut choisir un schéma interpolant si il veut une surface respectant parfaitement son pointé ou un schéma approximant si il tolère une légère différence entre le pointé et la surface lissée. À l'usage, les utilisateurs préfèrent utiliser une méthode approximante (voir figure 1.11).

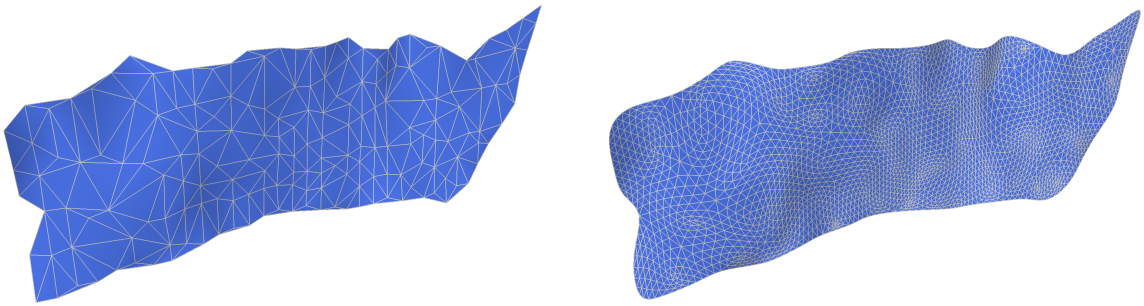


FIG. *1.11: Application d'un algorithme de lissage approximant sur une faille.*

1.3 Arrangement du réseau de failles

La deuxième étape dans la construction du modèle structural est la mise en cohérence du réseau de failles. En effet, un réseau de failles est dû à la rupture de la roche sous l'influence

des forces tectoniques, la roche cède alors en créant une faille dite *principale* et l'apparition de cette faille engendre la création d'une multitude de failles plus petites dites *secondaires* qui se branchent sur la faille principale. Cet arrangement du réseau de failles consiste alors à assurer des contacts propres entre les failles.

La figure 1.12 illustre les arrangements nécessaires entre une faille principale (en vert) et une faille secondaire (en rouge) : extension et coupure d'une surface sur et par une autre.

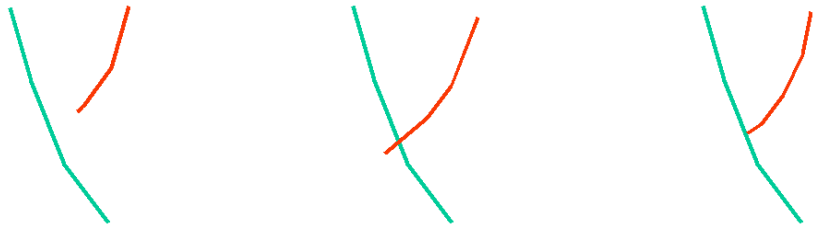


FIG. *1.12: Exemples d'arrangements nécessaires à la construction du réseau de failles. (a) Les failles ne sont pas jointives, il faudra étendre la faille rouge. (b) La faille rouge traverse la faille verte, il faudra la couper. (c) Les failles sont étanches, c'est le résultat attendu.*

La mise en cohérence du réseau de failles nécessite donc de détecter les failles qui s'intersectent et les failles non jointives, les surfaces modélisants ces failles seront respectivement coupées ou étirées. Nous proposons deux approches pour résoudre ce problème : une approche surfacique basée sur les découpes et extensions de surfaces triangulées, et une approche volumique qui résout le problème de manière globale.

1.3.1 Approche surfacique

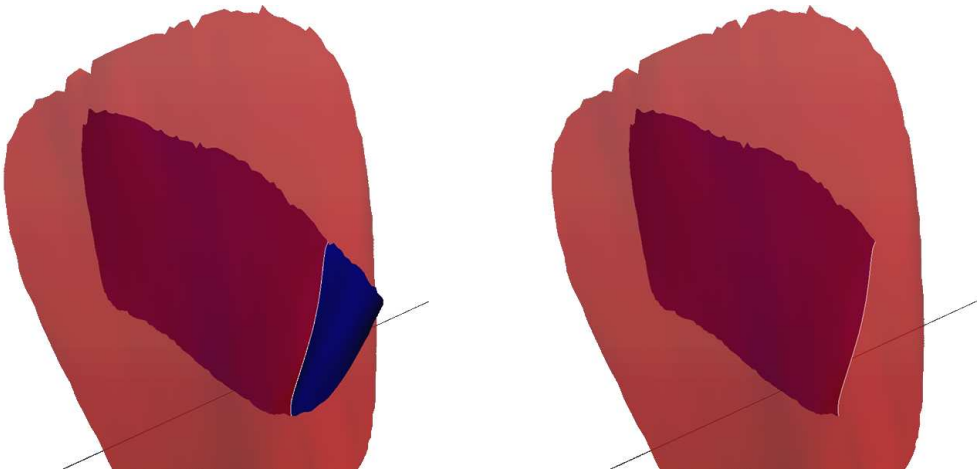
Cette approche consiste à appliquer les opérations illustrées dans la figure 1.12 sur les surfaces triangulées contruites dans l'étape précédente. Elle se décompose en deux étapes : le traitement des failles qui s'intersectent et le cas de failles secondaires non jointives avec leur faille principale.

Gestion des intersections

La première étape concerne la détection des failles qui s'intersectent et leur traitement afin de les rendre cohérentes. Pour celà, nous procédons de la manière suivante :

1. Nus calculons les polygones issus de l'intersection des surfaces triangulées.
2. Nous découpons la faille secondaire le long de ces polygones d'intersections.
3. Nous conservons la plus grande partie de la faille secondaire découpée.

Cette opération est illustrée dans la figure 1.13.



(a) (b)

FIG. *1.13: (a) En bleu, la faille mineure traverse la faille majeure (en rouge). L'intersection entre les failles est visible en blanc. (b) La faille secondaire est coupée selon le polygone d'intersection et la partie la plus petite du résulat est suprimée.*

Calage des failles

Dans la seconde étape, les failles sont étendues pour venir en contact avec les failles voisines si elles sont suffisamment proches. Pour cela nous procédons de la manière suivante :

1. Nous détectons tous les points de bordure.
2. Nous estimons le plan tangent en ces points.
3. Nous lançons un rayon depuis chacun de ces points. Le rayon est défini dans le plan tangent et est perpendiculaire à la bordure. Les rayons peuvent alors intersecter n'importe quelle faille du réseau, seul le contact le plus proche sera retenu si il est inférieur à une distance seuil.
4. La surface est ensuite étendue en joignant les sommets de la bordure aux points de contact calculés lors du lancer de rayons.

Cette opération est illustrée dans la figure 1.14.

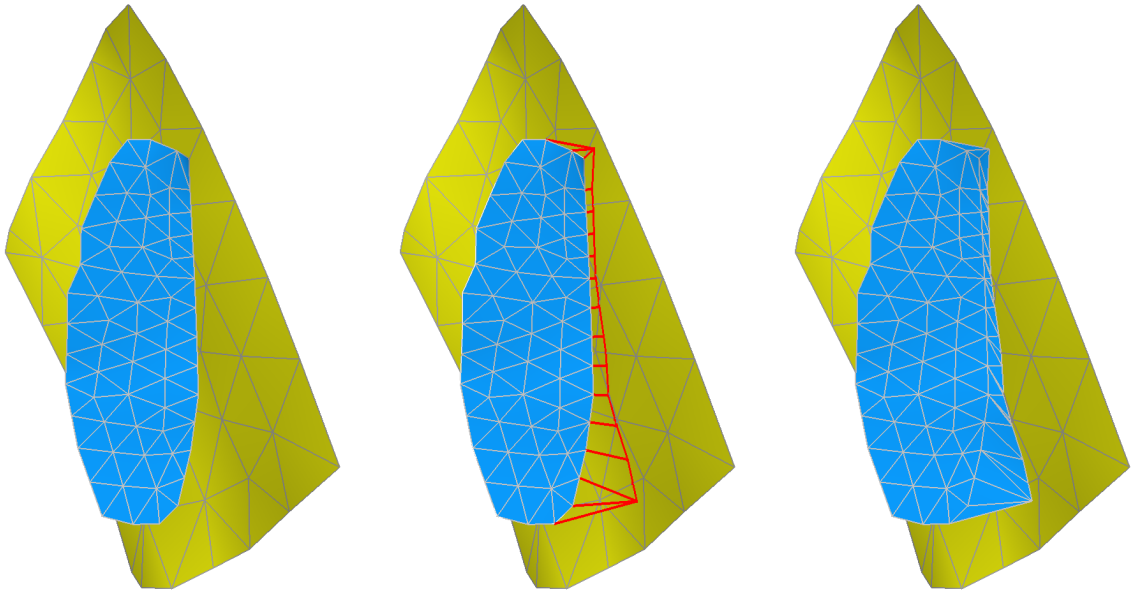


FIG. *1.14: Étape de projection d'un faille sur une autre*

Afin de faciliter les calculs d'intersection, l'ensemble des failles composant le réseau est intégré dans un *octree* (voir figure 1.15). Les octrees sont utilisés pour partitionner récursivement un espace tridimensionnel avec des boites englobantes. Seules les feuilles contiennent les triangles des surfaces triangulées. Dans le but de réduire le nombre de calculs d'intersection sur les triangles, les tests sont faits dans un premier temps sur les boites englobantes afin d'élimer une partie de l'espace et ainsi une partie des triangles.

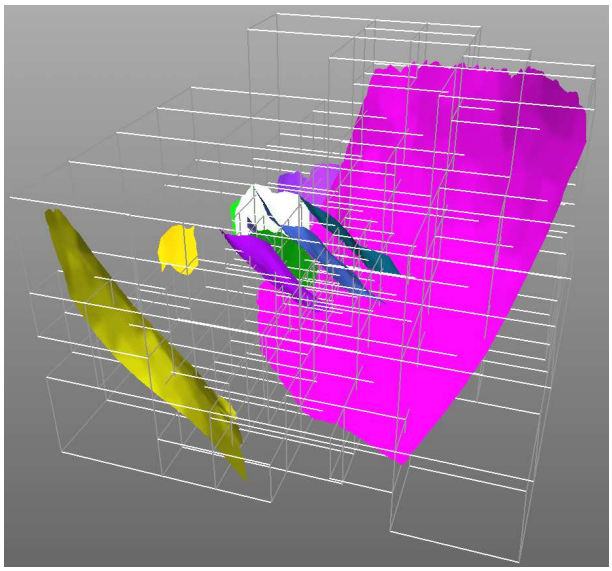


FIG. *1.15: Réseau de failles et son octree*

Dans ce traitement, l'utilisateur doit définir les failles comme étant majeures ou mineures. Si les failles sont majeures, elles ne seront pas modifiées. Par contre si elles sont mineures, alors elles pourront subir des modifications afin de les rendre cohérentes.

Il existe des cas où cette approche n'est pas suffisante, par exemple si deux failles s'intersectent mais ne se partitionnent pas (voir figure 1.16). Dans ce cas, on ne peut pas décider simplement quelle partie de la faille doit être éliminée. C'est pourquoi des travaux de recherche continuent sur ce sujet.

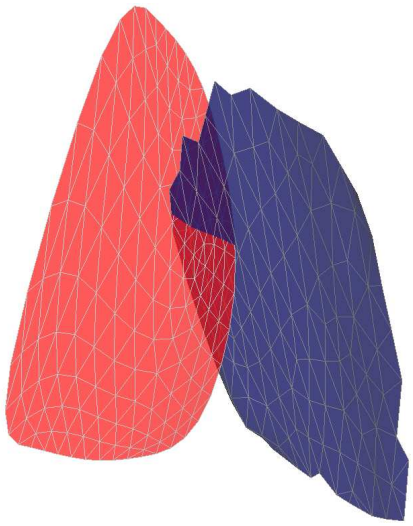


FIG. *1.16: Dans ce cas, les deux s'intersectent mais ne se partitionnent pas*

1.3.2 Approche volumique

Nous présentons ici une approche globale pour résoudre le problème d'arrangement d'un réseau de failles. Cette approche a été développée dans les derniers mois de la thèse en collaboration avec Camille Perin.

L'idée principale repose sur une définition implicite du réseau de failles existant. Nous devons définir pour chaque faille une fonction $f : \mathbb{R}^3 \mapsto \mathbb{R}$ tel que la surface soit définie par l'ensemble des points vérifiant

$$f(x, y, z) = 0$$

Pour une faille φ et un point p, on définit :

- p' le point de φ le plus proche de p,
- p'' la projection orthogonale de p sur $\Pi_{p'}$, le plan tangent à φ en p'.

La figure 1.17 illustre ces notions : dans le cas du point A, sa projection A' est identique au point A". Dans le cas du point B sa projection B" n'appartient pas à φ et est distincte de B'.
Nous définissons alors deux fonctions distances :

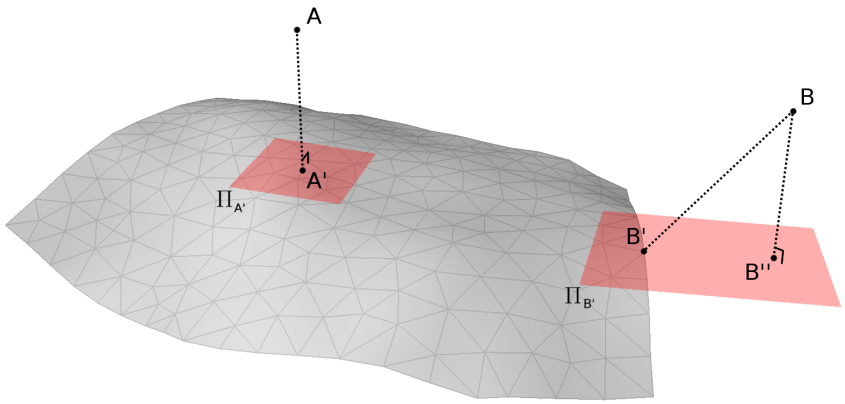


FIG. *1.17: Illustration des notions de point le plus "proche" et de point "projeté"*

- la distance finie notée d_F^φ définie par $d_F^\varphi(p) = d(p, p')$ et
- la distance infinie notée d_{INF}^φ définie par $d_{INF}^\varphi = d(p, p'')$.

où d la distance euclienne entre deux points.

À partir de ces définitions, nous pouvons montrer que :

- la surface d'origine φ est définie par l'ensemble des points vérifiant $d_F^\varphi = 0$,
- la surface étendue (ou extrapolée) est définie par l'ensemble des points vérifiant $d_{INF}^\varphi = 0$

La figure 1.18 illustre les calculs de la surface d'origine et de la surface étendue par les deux fonctions implicites $d_F^\varphi = 0$ et $d_{INF}^\varphi = 0$.

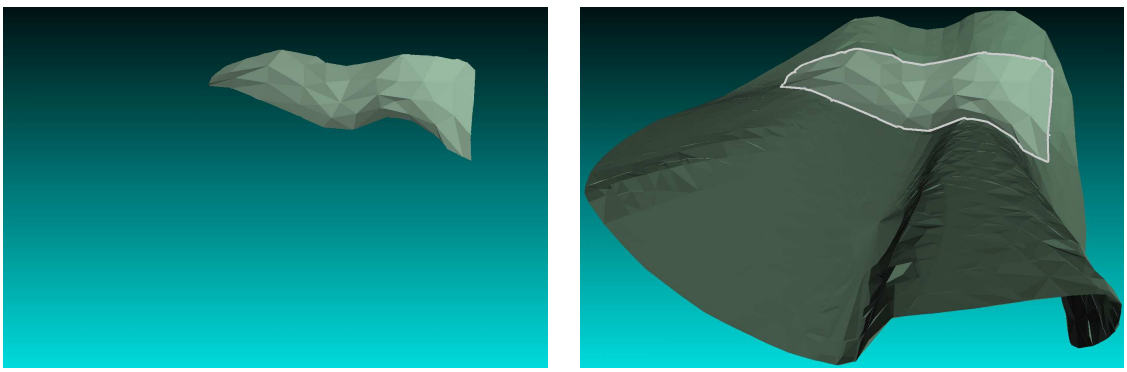


FIG. *1.18: À gauche, la faille d'origine. À droite, la faille d'origine (limitée en blanc) et la faille étendue*

L'ajustement de deux failles (extension et/ou coupe) se fait alors simplement en définissant dans un premier temps l'intersection des deux failles étendues, à savoir les points vérifiant

$$d_{INF}^{\varphi_1} = 0 \quad \text{et} \quad d_{INF}^{\varphi_2} = 0$$

Cette intersection est illustrée par la ligne rouge dans la figure 1.19. L'ajustement des deux failles consiste alors à déformer le contour de la faille secondaire (les contours des failles sont illustrés en blanc dans la figure) pour se caler sur la ligne d'intersection.

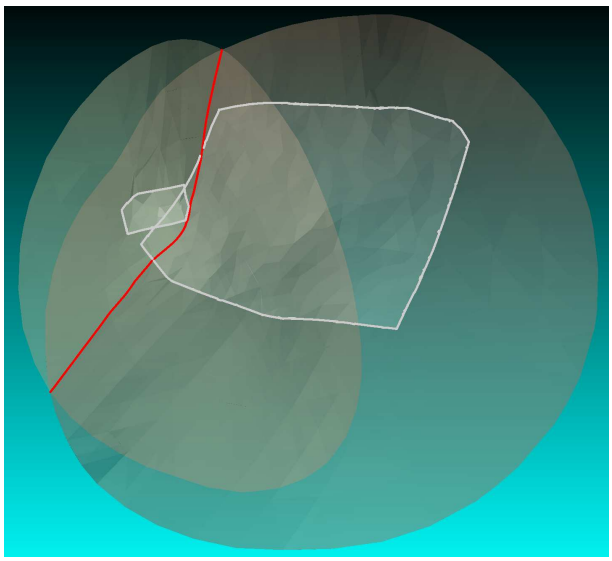


FIG. *1.19: Intersection de deux failles (en rouge)*

Pour implémenter cette approche, nous devons tout d'abord définir un volume d'intérêt englobant le réseau de failles, ici une sphère englobante. Puis dans un second temps nous devons discrétiser ce volume afin d'évaluer les distances d_F et d_{INF} aux noeuds du volume discrétisé. Pour cela nous proposons une discrétisation basée sur la triangulation de Delaunay en trois dimensions générant un pavage du volume par des tétraèdres.

Afin d'obtenir une tétraèdrisation raffinée dans les zones d'intéret : c'est-à-dire à proximité des failles originales, nous construisons un octree pour guider la construction de la grille (voir figure 1.20). L'octree sera raffiné selon deux critères : la profondeur maximale de l'octree qui permet de limiter le volume de données et la complexité de la surface (les grandes surfaces uniformes n'ont pas besoin de beaucoup de détails).

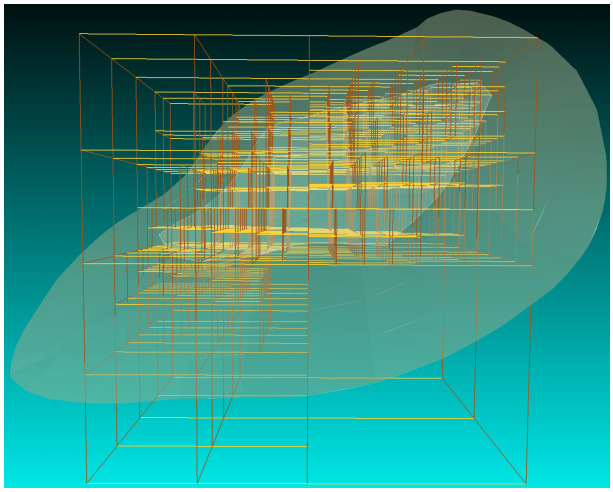


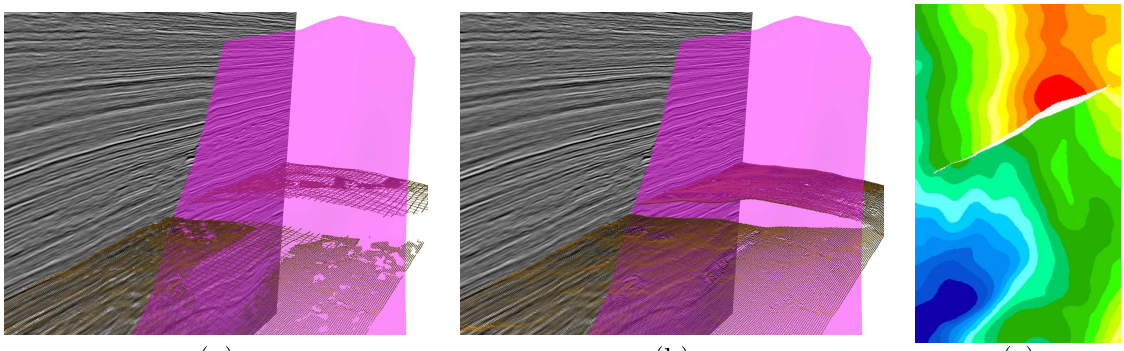
FIG. *1.20: Raffinement de l'octree pour s'adapter à la complexité de la faille*

Les résultats obtenus par cette approche sont très prometteurs, et ces travaux font l'objet d'une nouvelle thèse. En effet de nombreuses optimisations sont à apporté afin de gérer certains cas limites non introduits ici. L'ajustement des failles par déformation de contour n'a pas encore été abordé. Enfin, nous pensons pouvoir, grace à cette approche, fusionner les deux premières étapes de la construction du modèle structural en définissant directement les fonctions distances à partir des lignes brisées modélisant les failles plutôt qu'à partir des surfaces triangulées.

1.4 Interpolation des horizons

Dans l'introduction générale, nous avons vu les méthodes utilisées par les géophysiciens pour extraire les informations décrivant les horizons dans la sismique. Le résultat de l'interprétation est un semis de points incomplet défini sur une grille régulière. L'interpolation de surface consiste alors à estimer une valeur en tout point de la grille. Cependant afin de définir une surface respectant les contraintes géologiques (réseau de failles construit dans l'étape précédente), l'interpolation ne pourra se faire qu'aux points de la grille où la surface existe : les zones non valuée correspondent aux lèvres de failles. La figure 1.21 illustre ce problème d'interpolation sous contrainte et la notion de lèvre de faille.

Nous pouvons envisager deux approches pour interpoler le semis : la première utilisant les surfaces polygonales de manière similaire à la construction des failles et la seconde traitant le



(a) (b) (c)

FIG. *1.21: En rose une faille, en orange un semis de points modélisant un horizon. (a) Horizon avant l'interpolation. (b) Horizon après interpolation. La surface reconstruite est mise en cohérence avec la faille. (c) Vue carte de l'horizon reconstruit, la lèvre de faille est le trou laissé dans la grille.*

semis de point comme une carte de profondeur.
Nous n'avons pas retenue la première méthode car les semis de points contiennent souvent plusieurs millions d'éléments et la surface triangulée serait trop coûteuse en mémoire.

Nous avons vu que les semis de points étant sur une grille régulière, il peuvent alors être assimillés à des images (ou carte de profondeur) dans lesquelles il manquerait des pixels dont il faut trouver la valeur (dans notre cas, nous cherchons la profondeur correspondant à la position du pixel). Il existe plusieurs avantages à l'utilisation de cette représentation pour modéliser les horizons. Le premier est que la résolution de l'image peut-être identique à celle de la sismique, permettant de préserver toute la finesse des détails issus de celle-ci. Un autre avantage est de profiter de la connectivité intrinsèque portée par l'image pour utiliser des algorithmes d'interpolation présents dans la littérature et particulièrement en traitement d'image.

L'étude des algorithmes d'interpolation utilisée en cartographie [67] et en traitement d'image, plus particulièrement dans les techniques *inpainting* [9, 16, 75] nous a permis d'extraire des caractériques propres à ces méthodes.
Pour chaque point de la grille devant être évalué il faut lui associer un *voisinage* et un *estimateur*. Le voisinage détermine l'ensemble des points valués ayant un impact sur ce point, et l'estimateur permet de calculer sa valeur en fonction de son voisinage. Nous devons aussi choisir une *stratégie de remplissage*. En effet l'estimation peut se restreindre à l'utilisation des données de départ ou utiliser les données de départ ainsi que les données calculées au fur et à mesure. Dans le deuxième cas l'ordre influençant le résultat il faut déterminer un ordre d'évaluation.

1.4.1 Notion de voisinage

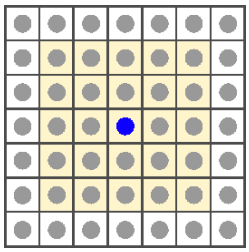
Étant donné un ensemble de points observées $\{s_i = (x_i, y_i)\}$ et un point s_0 de coordonnées (x_0, y_0), on entend par voisinage de s_0, noté $\mathcal{V}(s_0)$, le sous-ensemble des points s_i qui auront une contribution non négligeable dans l'estimation d'une valeur en s_0. Potentiellement tous les

points s_i déjà estimés peuvent appartenir à ce voisinage.
Traditionnellement on relie la contribution d'un point s_i à sa distance avec le point s_0 : plus le point est éloigné et plus sa contribution est faible. Ceci nous permet de limiter le voisinage $\mathcal{V}(s_0)$ à l'ensemble des points les plus proches :

$$\mathcal{V}(s_0) = \{s_i \mid d(s_0, s_i) < d_0\}$$

ou d est une fonction distance et d_0 un seuil.

La forme du masque est alors associée au choix de la fonction distance : circulaire pour une distance euclidienne, carré pour une distance infinie, etc. Par exemple un masque carré de taille 5x5 centré sur le pixel p (en bleu dans la figure ci-contre). Dans ce cas le voisinage $\mathcal{V}(p)$ sera l'ensemble des pixels compris dans ce masque (en jaune dans l'image).



Cette définition du voisinage n'est cependant pas satifaisante dans notre contexte d'utilisation. En effet, dans le cas d'un point s_0 situé à proximité d'une faille, seuls les points du même côté de la faille doivent participer à l'estimation d'une valeur en s_0. La figure 1.22 illustre le problème induit par le prise en compte de tout le voisinage dans l'estimation d'une valeur en s_0 (point bleu). Nous montrons en grisé dans la figure l'omission souhaitée des points du voisinage causé par la présence de la faille.

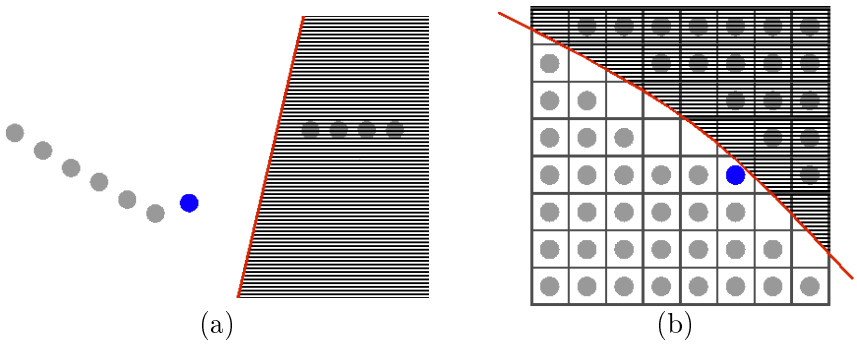


FIG. *1.22: L'influence des points pour l'estimation du pixel bleu est modifiée par la présence des failles : les partie hachurées n'auront plus d'impact dans l'estimation (a) Vue en coupe (b) vue en carte*

Cette observation est à la base de notre solution : la faille segmente localement l'horizon en deux parties, et seuls les points situés du même côté de la faille doivent être pris en compte dans le voisinage. Pour cela nous devons définir la notion de *visibilité* entre deux points s_0 et s_i qui permet d'adapter le voisinage $\mathcal{V}(s_0)$.

Dans la suite nous utiliserons les notations suivantes : nous noterons $z(s)$ une valeur observée et $\hat{z}(s)$ une valeur estimée en un point s de l'image modélisant l'horizon.

Notion de visibilité

Étant donné $\mathcal{V}(s_0)$ voisinage de s_0 défini précédemment. Un point de ce voisinage ne sera pris en compte dans l'estimation que si il est visible par le point estimé à travers le réseau de failles. Nous pouvons formuler ce concept de la manière suivante :

$$\forall s_i \in \mathcal{V}(s_0),\ p_i = \left| \begin{array}{l} x_i \\ y_i \\ z_i = z(s_i) \end{array} \right. \quad \text{et } p_0 = \left| \begin{array}{l} x_0 \\ y_0 \\ z_0 = \hat{z}(s_0) \end{array} \right.$$

Deux points, s_0 et s_i sont dits visibles à travers le réseau de failles Φ (composé de m failles $\{\phi_1, \ldots, \phi_m\}$) si le segment p_0p_i n'intersecte aucune faille de Φ.

Nous pouvons définir un nouveau *voisinage partiel* $\mathcal{W}(s_0)$ du point s_0 qui exclut tous les points de $\mathcal{V}(s_0)$ qui ne sont pas visibles à travers le réseau de failles :

$$\mathcal{W}(s_0) = \{s_i \in \mathcal{V}(s_0) \mid p_0p_i \cap \phi_j = \varnothing, \forall \phi_j \in \Phi\}$$

$\mathcal{W}(s_0)$ est en fait une définition implicite du voisinage car p_0 doit être connu pour calculer les intersections de p_0p_i avec le réseau de failles.
Afin de résoudre ce problème, nous proposons dans la suite une méthode de construction itérative de ce voisinage.

Construction du voisinage partiel

1. Dans un premier temps, on considère $\mathcal{V}(s_0)$, le voisinage complet du point s_0. C'est-à-dire que le voisinage est déterminé sans tenir compte des failles. Ce voisinage apparait en jaune dans la figure 1.23. Le calcul de $\hat{z}_1(s_0)$ est fait avec le voisinage $\mathcal{V}(s_0)$. Cette première estimation du z au point s_0 est modélisé par un point bleu dans la figure figure 1.23.
2. La seconde étape consiste à corriger le voisinage initial $\mathcal{V}(s_0)$ en utilisant le point estimé durant l'étape précedente. Afin de déterminer ce voisinage $\mathcal{W}_1(s_0)$, on élimine du voisinage original $\mathcal{V}(s_0)$ tous les points non visibles par le point estimé $(x_{s_0}, y_{s_0}, \hat{z}_1(s_0))$ à travers les failles. Ce nouveau voisinage est représenté par les points oranges dans la figure 1.24.

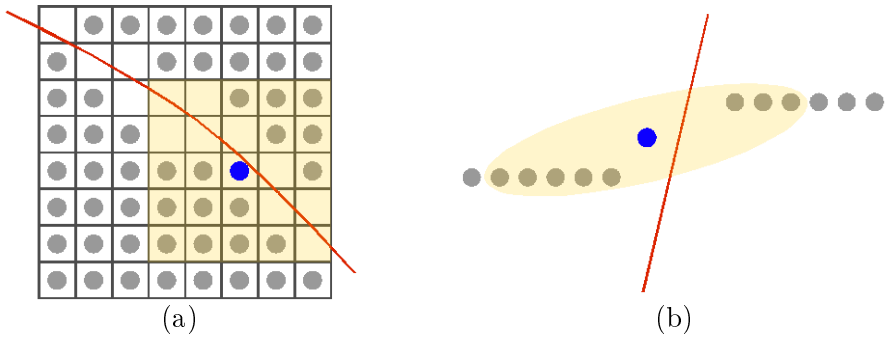


FIG. *1.23: Étape 1 de l'estimation d'un point avec les contraintes de failles. Dans cette étape la position du point est évalué avec un voisinage complet.*

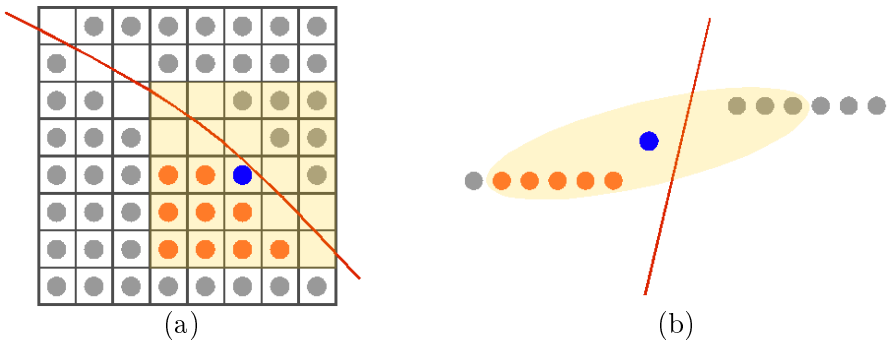


FIG. *1.24: Étape 2 de l'estimation d'un point avec les contraintes de failles.*

3. L'étape précédente nous a permis de réduire le voisinage initial, nous allons donc affiner notre estimation en calculant $\hat{z}_2(s_0)$ une nouvelle estimation de la profondeur de s_0 à partir du voisinage $\mathcal{W}_1(s_0)$. La figure 1.25 nous montre par un point bleu la nouvelle position du point en s_0.

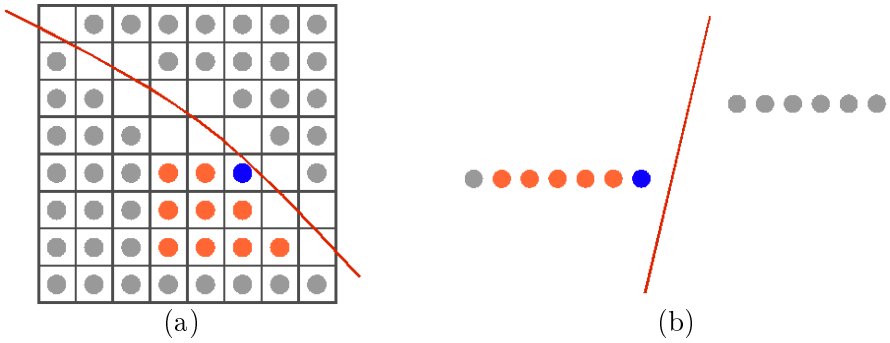


FIG. *1.25: Étape 3 de l'estimation d'un point avec les contraintes de failles.*

4. La dernière étape consiste à valider le choix du voisinage. De la même façon qu'à l'étape 2, on calcule le voisinage $\mathcal{W}_2(s_0)$ qui est un sous-ensemble de $\mathcal{V}(s_0)$ en fonction du point estimé à l'étape précédente $(x_{s_0}, y_{s_0}, \hat{z}_2(s_0))$. On distingue alors deux cas : si le voisinage $\mathcal{W}_1(s_0)$ est identique à $\mathcal{W}_2(s_0)$ on considère que le voisinage est stable et que l'estimation est valide (voir figure 1.26 (a)(b)), sinon on considère que s_0 est dans la lèvre de faille car le point oscille de part et autre de celle-ci (voir figure 1.26 (c)(d)).

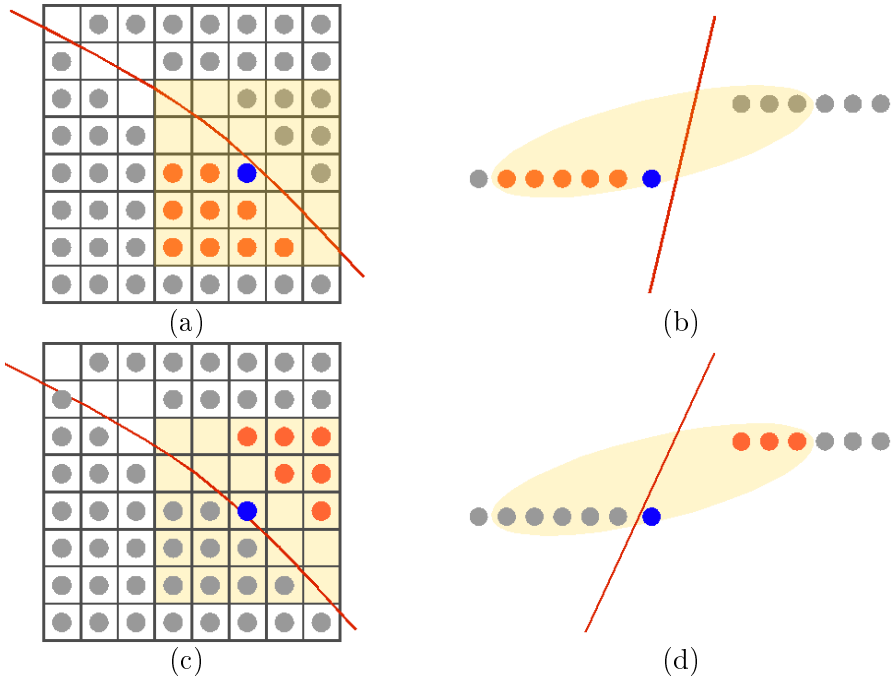
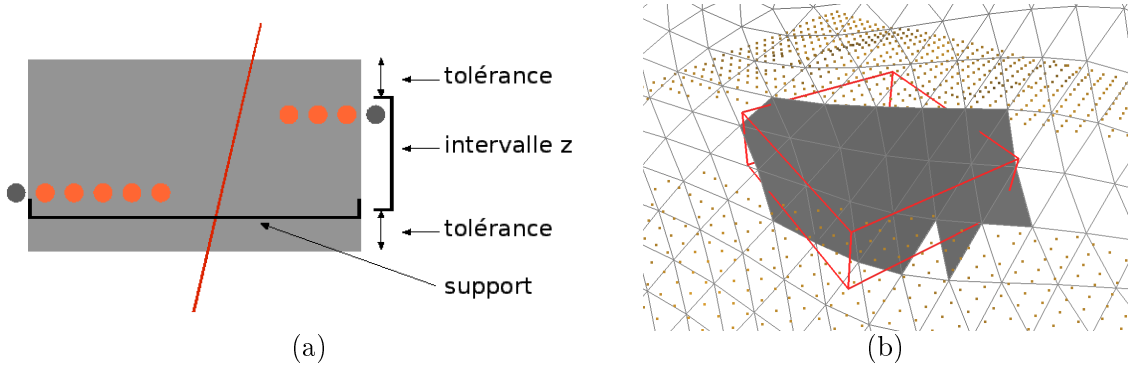


FIG. *1.26: Étape 4 de l'estimation d'un point avec les contraintes de failles. En haut, le voisinage est constant, le point est validé. En bas, le voisinage est instable, on considère que l'on est dans la lèvre de faille.*

La méthode de construction proposée est basée sur des intersections entre les segments, définis par le point estimé et les points de son voisinage, et les surfaces triangulées composant le réseau de failles. Ces nombreux calculs d'intersection sont consommateurs en temps de calcul. Afin de réduire le nombre de ces calculs, nous déterminons, lors de l'estimation d'un point, quels vont être les triangles qui ont une influence sur cette estimation. Ceux-ci appartiennent à la boite définie horizontalement par le support et bornée verticalement par les profondeurs $z(s_i)$ des points appartenant ce support. La profondeur estimée $\hat{z}(s_0)$ pouvant sortir de cet intervalle, nous ajoutons une tolérance verticale à la boite. La figure 1.27 illustre une recherche de triangles intervenant dans l'estimation d'un point avec cette technique.

Le nombre de triangles selectionnés avec cette méthode (en gris dans la figure 1.27 (b)) est alors très faible en comparaison du nombre global de triangles : moins d'une dizaine comparé aux milliers composant le réseau de failles. Le temps de calcul utilisé pour les tests d'intersection



(a) (b)

FIG. *1.27: Sélection des triangles ayant une influence dans l'estimation d'un point (a) En gris clair la boite englobante définie horizontalement par le support et verticalement par l'intervalle des valeurs contenues par le support (b) visualisation en trois dimensions de la boite et des triangles sélectionnés*

est alors drastiquement réduit.

Dans le schéma que nous présentons ici pour calculer le voisinage partiel, nous voyons que l'estimation de $\hat{z}(s_0)$ et la définition du voisinage sont intrinsèquement liés. Le but du prochain paragraphe est de présenté les méthodes d'estimation et de choisir la plus adaptée à notre problème.

1.4.2 Estimateurs

De façon générale, un estimateur peut être défini comme l'opération qui, pour un ensemble de valeurs observées $\{z(s_i)\}$ aux points $\{s_i = (x_i, y_i)\}$, permet d'estimer à la position s_0, de coordonnées (x_0, y_0), la valeur $\hat{z}(s_0)$ qui minimise une erreur donnée.

La littérature est abondante sur le sujet des estimateurs et le but de cette section n'est pas de faire un état de l'art exhaustif du sujet, mais plutôt de présenter les méthodes les plus couramment utilisées dans le domaine de l'interpolation, à savoir :

- les méthodes de krigeage abondamment utilisées en géosciences,
- les méthodes barycentriques qui estiment un point par une combinaison de ces voisins et
- les méthodes de régressions locales qui utilisent un modèle local pour interpoler.

Krigeage

Dans le monde des géosciences, les méthodes de krigeage[1] sont les méthodes les plus utilisées dans le cas de semis de points éparses. Dans cette thèse, nous allons nous restreindre au krigeage

[1]Le krigeage doit son nom à l'ingénieur minier D.G. Krige qui a développé, dans les années cinquante, une série de méthodes statistiques empiriques afin de déterminer la distribution spatiale de minerais à partir d'un ensemble de forages. Le terme "krigeage" et le formalisme de cette méthode sont dûs au français Matheron. Il a aussi été le premier à utiliser le terme "géostatistique". Cette méthode est connue en statistique sous le nom de "méthode d'interpolation de Gauss-Markov".

ordinaire et au krigeage simple, qui supposent que la moyenne et la variance soient stationnaires, c'est-à-dire qu'elles ne dépendent que de la distance entre les points et pas de leur position.

Krigeage ordinaire Le principe du krigeage ordinaire consiste à déterminer l'estimateur linéaire

$$\hat{z}(s_0) = \sum_{i=1}^{n} \lambda_i z(s_i)$$

qui minimise la variance d'estimation

$$\sigma_e^2 = \sigma_0^2 + \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j \sigma_{i,j} - 2\sum_{i=1}^{n} \lambda_i \sigma_{0,i}$$

avec $\sigma_{i,j} = cov(z(s_i), z(s_j))$ la covariance aux points s_i et s_j et $\sigma_i^2 = cov(z(s_i), z(s_i))$ la variance au point s_i.
Afin que l'estimateur soit sans biais, il faut que

$$\sum_{i=0}^{n} \lambda_i = 1$$

On a donc un problème de minimisation sous contrainte. Soient $\lambda = (\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^n$ et $\mu \in \mathbb{R}$. On écrit le lagrangrien :

$$\mathcal{L}(\lambda, \mu) = \sigma_e^2 + 2\mu \left(\sum_{i=1}^{n} \lambda_i - 1 \right)$$

L'idée est de choisir les λ_i de façon à minimiser la variance d'estimation σ_e^2. Pour obtenir ce miminum nous dérivons $\mathcal{L}$ par rapport à λ et μ et nous cherchons les valeurs pour lesquelles ces dérivées partielles valent 0. Nous obtenons donc un système linéaire comportant $n+1$ équations à $n+1$ inconnues :

$$\forall i = 1...n \quad \sum_{j=1}^{n} \lambda_j \sigma_{i,j} + \ \mu = \sigma_{0,i}$$

qui peut s'écrire sous forme matricielle :

$$\begin{pmatrix} \sigma_1^2 & \sigma_{1,2} & . & \sigma_{1,n} & 1 \\ \sigma_{2,1} & \sigma_2^2 & . & \sigma_{n,1} & 1 \\ . & . & . & . & . \\ \sigma_{n,1} & \sigma_{n,2} & . & \sigma_n^2 & 1 \\ 1 & 1 & . & 1 & 0 \end{pmatrix} . \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ . \\ \lambda_n \\ \mu \end{pmatrix} = \begin{pmatrix} \sigma_{0,1} \\ \sigma_{0,2} \\ . \\ \sigma_{0,n} \\ 1 \end{pmatrix}$$

Dans ce système, les valeurs de covariance dans la matrice et le vecteur sont obtenues grâce au modèle de covariance estimé sur le *variogramme expérimental*. Celui-ci s'exprime :

$$\gamma(h) = \frac{1}{2n(h)} \sum_{i=1}^{n(h)} (z(s_i) - z(s_i + h))^2$$

où $n(h)$ est le nombre de paires de points séparés par la distance h.

Krigeage simple Dans le cas du krigeage simple, on suppose que le moyenne m est connue. On peut alors former un estimateur sans biais sans imposer la contrainte que la somme des poids soit égale à 1. On obtient donc :

$$\hat{z}(s_0) = m + \sum_{i=1}^{n} \lambda_i (z(s_i) - m)$$

De la même façon que pour le krigeage ordinaire, il faut choisir les λ_i de façon à minimiser la variance d'estimation σ_e^2. Pour obtenir le minimum, on dérive σ_e^2 par rapport à chacun des λ_i et on cherche pour quelles valeurs ces dérivées partielles sont égales à zéro. On obtient donc un système linéaire comportant n équations à n inconnues :

$$\forall i = 1...n \quad \sum_{j=1}^{n} \lambda_j \sigma_{i,j} = \sigma_{0,i}$$

qui peut s'écrire sous forme matricielle :

$$\begin{pmatrix} \sigma_1^2 & \sigma_{1,2} & \cdot & \sigma_{1,n} \\ \sigma_{2,1} & \sigma_2^2 & \cdot & \sigma_{n,1} \\ \cdot & \cdot & \cdot & \cdot \\ \sigma_{n,1} & \cdot & \cdot & \sigma_n^2 \end{pmatrix} \cdot \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \cdot \\ \lambda_n \end{pmatrix} = \begin{pmatrix} \sigma_{0,1} \\ \sigma_{0,2} \\ \cdot \\ \sigma_{0,n} \end{pmatrix}$$

Comme pour le krigeage ordinaire, la résolution de ce système nous donnera les λ_i dont nous avons besoin pour l'estimation de la profondeur de s_0.

Le krigeage est très adapté aux données éparses, cependant il est coûteux (la taille de la matrice de covariance à inverser est liée au nombre de points dans le voisinage), et dès que la donnée devient dense l'estimateur est similaire aux méthodes barycentriques.

Méthodes barycentriques

L'hypothèse de base de ces estimateurs repose sur le fait que localement la donnée est constante et que la valeur $\hat{z}(s_0)$ est le barycentre des points s_i appartenant au voisinage de

s_0 noté $\mathcal{V}(s_0)$ de cardinalité n. Ces méthodes cherchent la valeur $\hat{z}(s_0)$ qui minimise :

$$\sum_{i=1}^{n} \lambda_i (\hat{z}(s_0) - z(s_i))^2$$

Ces méthodes barycentriques sont donc définies par la formule :

$$\hat{z}(s_0) = \sum_{i=1}^{n} \lambda_i z(s_i)$$

avec $\forall i = 1 \dots n \ \lambda_i \in \mathbb{R}$. Afin de prévenir l'apparition de distorsion dans la surface interpolée, la majorité des méthodes barycentriques applique une contrainte sur les λ_i tel que :

$$\sum_{i=1}^{n} \lambda_i = 1$$

Dans le cas d'une moyenne simple nous définirons les λ_i par une constante : $\lambda_i = \frac{1}{n}$.
Dans d'autres cas, ces poids sont couramment exprimés en fonction de la distance $|s_i - s_0|$ de manière à ce que les points les plus proches aient plus de poids que les points éloignés. L'exemple le plus populaire est la méthode de l'inverse de la distance élevée à une puissance d, couramment appelée *inverse distance*. On pose donc

$$\lambda_i = \frac{|s_i - s_0|^{-d}}{\sum_{i=1}^{n} |s_i - s_0|^{-d}}$$

Afin d'illustrer l'insuffisance de ces approches, nous allons analyser leur comportement dans le cas d'extrapolation d'un bord : ce cas sera fréquent pour caler un horizon sur une faille. La figure 1.28 illustre ce comportement, et nous pouvons observer sur cette figure la discontinuité introduite au niveau de la bordure.

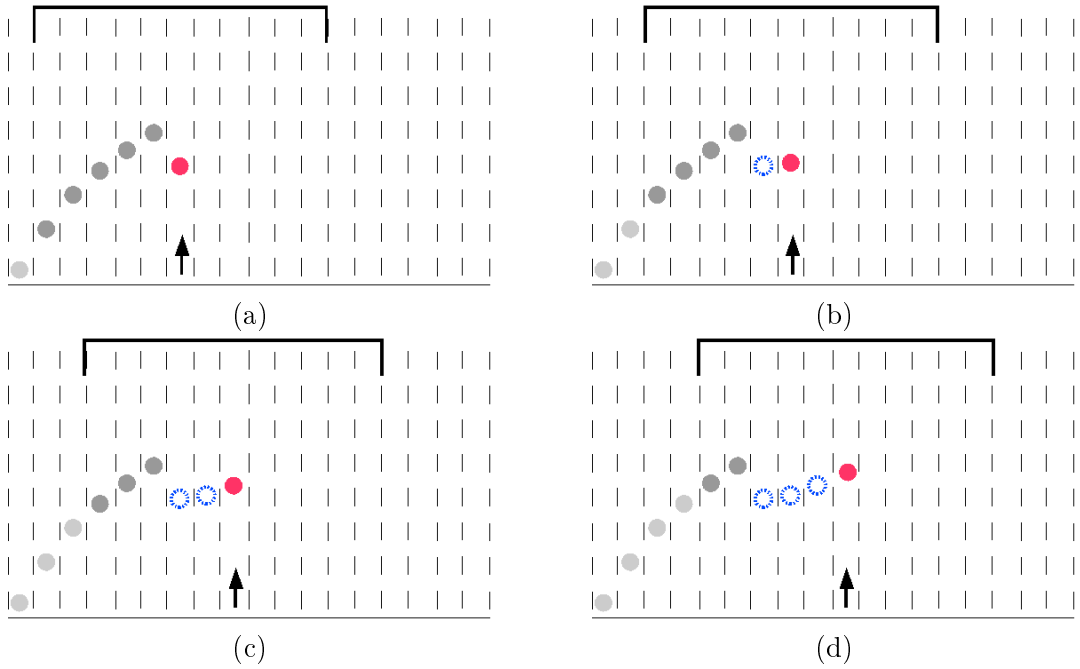


FIG. *1.28: Extrapolation d'un bord avec les méthodes barycentriques*

Cet artefact était prévisible, car les approches barycentriques ne sont pas par définition des extrapolateurs : en effet on utilise une somme pondérée des valeurs observées pour estimer la valeur locale avec une contrainte sur les λ_i : $\sum \lambda_i = 1$. La valeur estimée ne pourra alors pas sortir de l'intervale des valeurs observées. Dans ce cas, la surface interpolée ne pourra pas suivre la tendance locale et peut, dans certains cas, créer des ondulations. Nous allons donc nous intéresser dans la suite à des méthodes utilisant des modèles pour interpoler : les régressions.

Régressions locales

Historiquement les méthodes d'interpolation par régression sont issues des statistiques et consistent à trouver une fonction $f : \mathbb{R}^2 \mapsto \mathbb{R}$ tel que :

$$\hat{z}(s_0) = f(x_0, y_0)$$

la valeur estimée en un point s_0 minimise l'erreur :

$$\varepsilon_0 = \sum_{i=1}^{n} \lambda_i [\hat{z}_{s_0}(s_i) - z(s_i)]^2$$

où s_i est une position possédant une donnée observée et $\hat{z}_{s_0}(s_i)$ la valeur prédite au point s_i. Les coefficients λ_i permettent de pondérer l'influence des points en fonction de la distance au point à estimer.

Les fonctions f les plus utilisées sont les polynômes de degré d :

$$f(x,y) = \sum_{i+j \leq d} \beta_{ij} x^i y^j$$

avec $\beta_{ij} \in \mathbb{R}$. Le choix du degré d du polynôme dépend alors des données à modéliser : plus d sera grand plus le modèle sera souple et pourra suivre des ondulations locales, mais il sera alors d'autant moins robuste (plus sensible au bruit).
Nous pouvons aussi remarquer que le choix de $d = 0$ revient à définir f constante, l'approche est alors identique aux approches barycentriques.

Modèle plan Le modèle plan est alors une restriction à $d = 1$ et s'écrit sous la forme :

$$f(x,y) = ax + by + c$$

avec $a, b, c, \in \mathbb{R}$.
Nous cherchons donc à minimiser la fonction S définie par :

$$S(a,b,c) = \sum_{i=1}^{n} \lambda_i (ax_i + by_i + c - z_i)^2$$

Pour obtenir ce minimum, nous dérivons S par rapport à a, b et c. Nous cherchons pour quelles valeurs ces dérivées partielles valent 0 :

$$\begin{cases} \frac{\partial S}{\partial a}(a,b,c) = 0 \\ \frac{\partial S}{\partial b}(a,b,c) = 0 \\ \frac{\partial S}{\partial c}(a,b,c) = 0 \end{cases}$$

Cela revient à résoudre le système suivant :

$$\begin{cases} \sum_{i=1}^{n} 2\lambda_i x_i (ax_i + by_i + c - z_i) = 0 \\ \sum_{i=1}^{n} 2\lambda_i y_i (ax_i + by_i + c - z_i) = 0 \\ \sum_{i=1}^{n} 2\lambda_i (ax_i + by_i + c - z_i) = 0 \end{cases}$$

Ce système peut s'écrire sous la forme matricielle :

$$\begin{pmatrix} \sum_{i=1}^{n} \lambda_i x_i^2 & \sum_{i=1}^{n} \lambda_i x_i y_i & \sum_{i=1}^{n} \lambda_i x_i \\ \sum_{i=1}^{n} \lambda_i x_i y_i & \sum_{i=1}^{n} \lambda_i x_i^2 & \sum_{i=1}^{n} \lambda_i y_i \\ \sum_{i=1}^{n} \lambda_i x_i^2 & \sum_{i=1}^{n} \lambda_i y_i & \sum_{i=1}^{n} \lambda_i \end{pmatrix} \cdot \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{n} \lambda_i x_i z_i \\ \sum_{i=1}^{n} \lambda_i y_i z_i \\ \sum_{i=1}^{n} \lambda_i z_i \end{pmatrix}$$

La résolution de ce système nous donne les paramètres a, b et c de l'équation du plan qui nous permet d'estimer la profondeur du point s_0.

Afin de montrer le caractère extrapolateur de cette approche, nous illustrons sur la figure 1.29 son comportement dans le cas d'extrapolation d'un bord. On peut observer que contrairement aux méthodes barycentriques, la donnée reste continue et suit correctement la tendance.

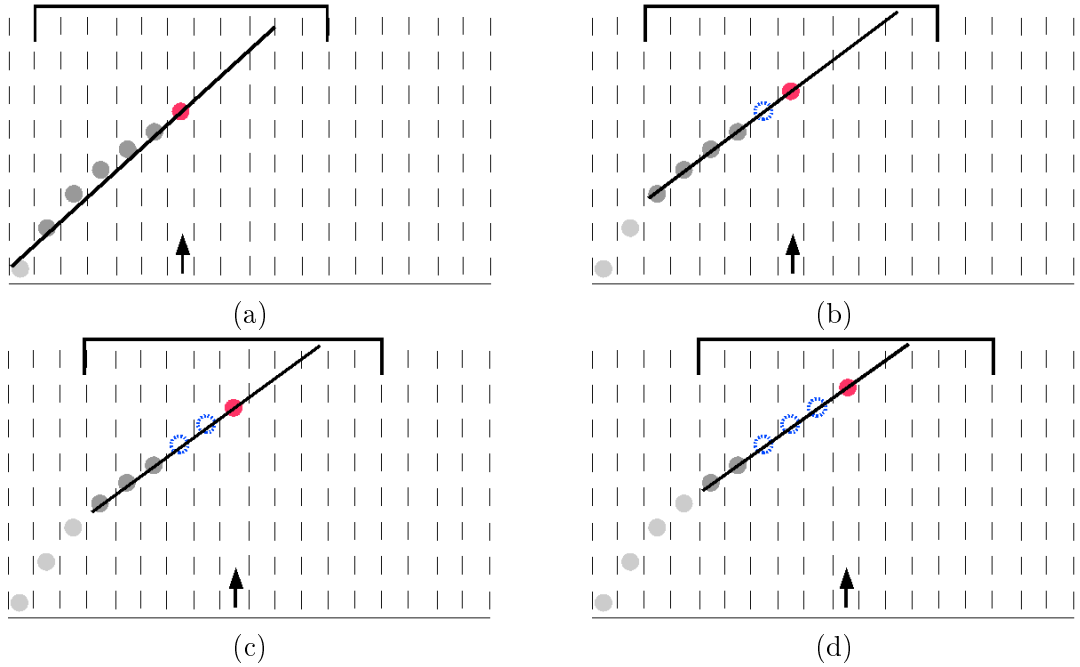


FIG. *1.29: Extrapolation d'un bord avec le modèle plan.*

Cette méthode n'est cependant pas parfaite. En effet, prenons le cas de la figure 1.30 d'une donnée qu'il faut interpoler. Dans cette figure, les points estimés lors d'étapes précédentes apparaissent en bleu. Nous nous attendons à obtenir une forme parabolique alors que la partie créée introduit une ondulation qui ne parait pas naturelle. Ce phénomène est dû au fait que, suivant la position du masque, les plans de régression peuvent être estimés à partir de points de chaque côté du trou (étapes (b), (c), (d) et (e)). Ceci fait descendre le plan de régression à partir duquel est estimé la profondeur du point.

Dans cette exemple, nous utilisons la méthodologie d'interpolation classique : les valeurs estimées ne sont pas prises en compte au fur et à mesure. Il faut noter que dans cette technique d'estimation, l'ordre d'évaluation des points n'a pas d'importance. En effet, les calculs étant basés sur les données initiales qui ne changent pas, ils sont indépendants. Nous qualifierons ces méthodes de remplissage de *stratégies à support statique*.

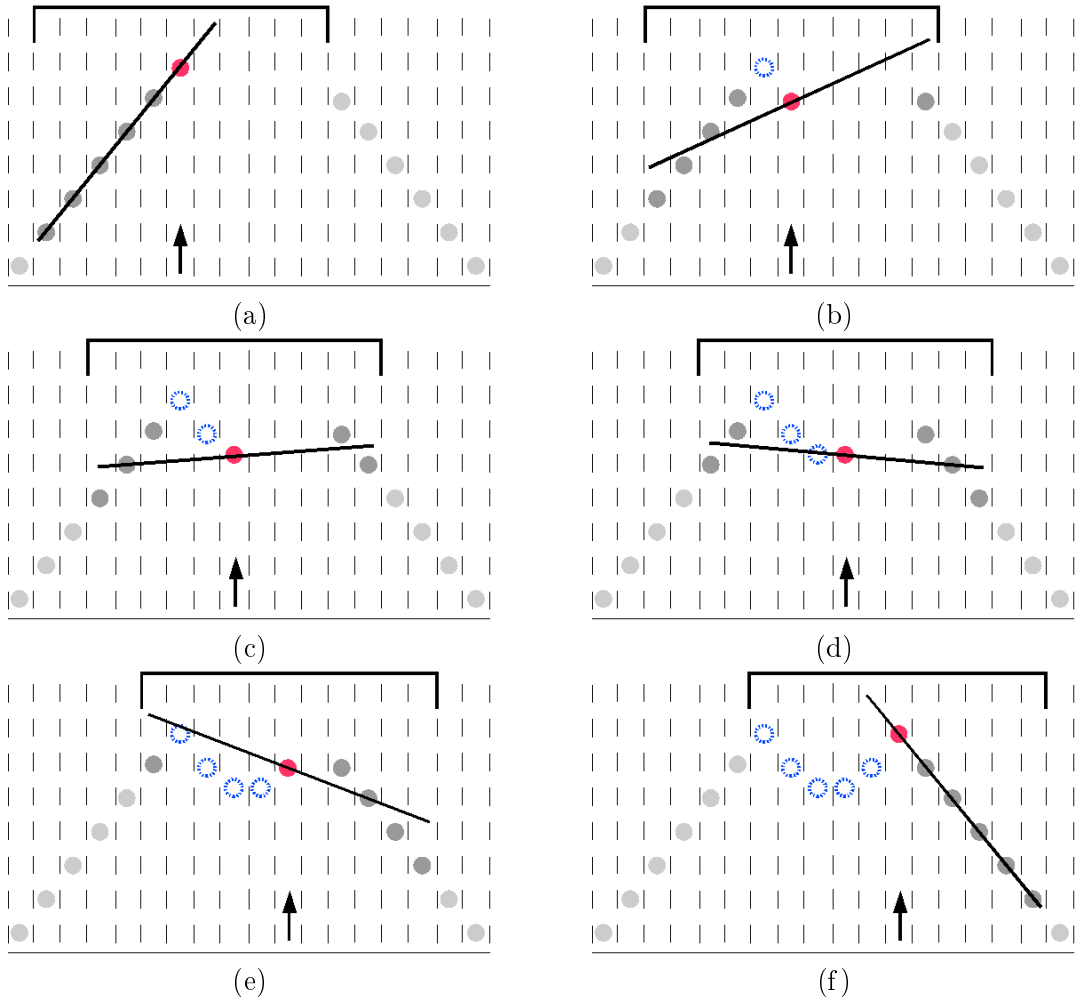


FIG. *1.30: Stratégie à support statique : une nouvelle image est construite à partir de l'image initiale. Dans ces images la position à évaluer est marquée par la flèche et la taille du masque est délimitée par la zone en haut des images. Dans cette série d'images, l'estimation est faite de gauche à droite. Les plans de régression estimés apparaissent sous la forme de segments noirs. Les points estimés lors des étapes précédentes sont dessinés en bleu mais ne sont pas pris en compte lors du calcul du point courant (en rouge).*

Une autre approche consiste à *propager* la donnée existante. Initialement, seule une position voisine des points de départ peut être valuée. L'estimation de cette position est alors ajoutée aux données initiales. Le point ainsi valué est considéré comme un point de départ et peut servir à valuer de nouvelles positions. L'image est alors complétée par l'introduction successive de points valués. Nous qualifierons cette technique de remplissage de *stratégie à support dynamique*.

Nous reprenons l'exemple précédent en utilisant une méthode d'interpolation avec le modèle plan en utilisant une stratégie à support dynamique. Nous illustrons ce cas dans la figure 1.31. Dans cette figure, les points estimés lors d'étapes précédentes sont utilisés au fur et à mesure de l'estimation. L'ondulation présente dans l'interpolation avec un support statique est aténuée

mais reste présente.

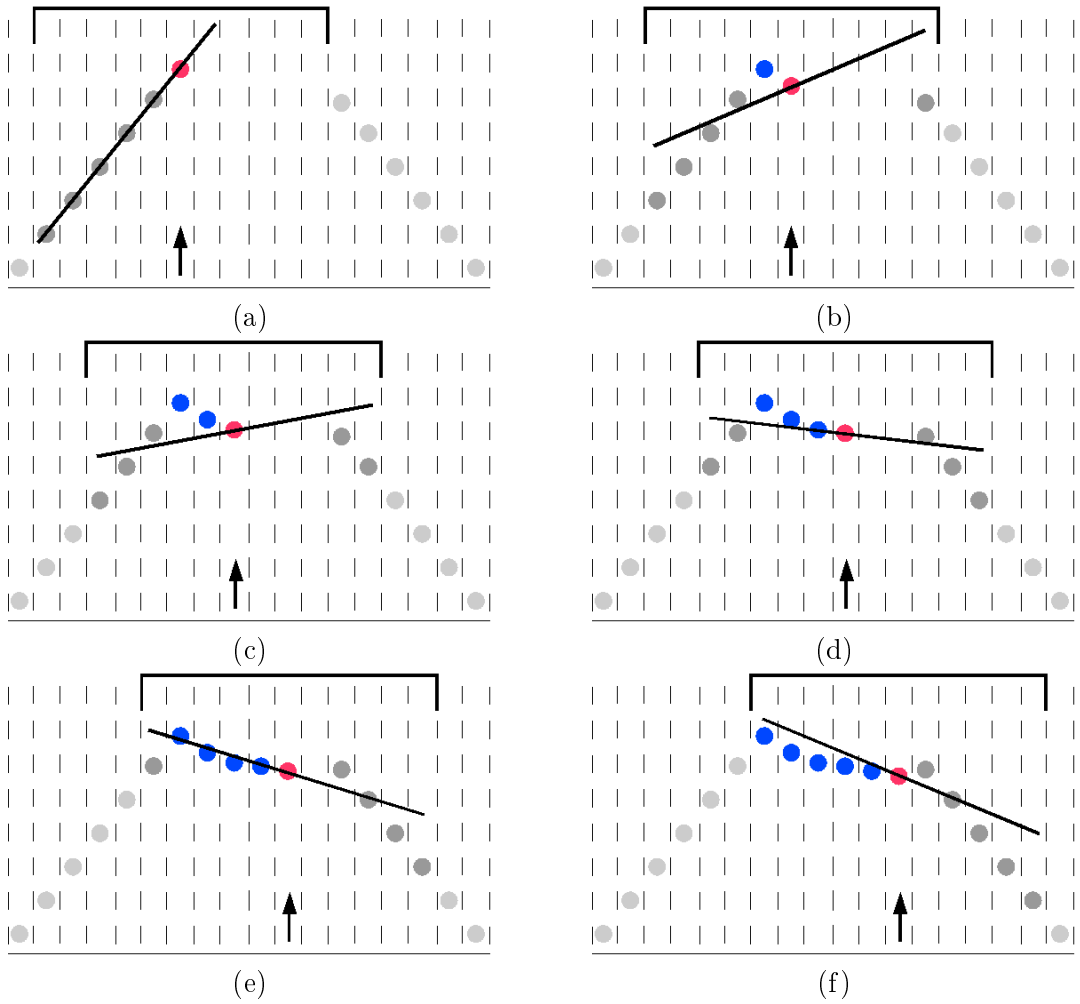


FIG. *1.31: Stratégie de remplissage à support dynamique : seuls les pixels ayant un ou plusieurs voisins valués peuvent être estimés. Comme pour la figure 1.30 la position à évaluer est marquée par la flèche et la taille du masque est délimitée par la zone en haut des images. De même, l'estimation est faite de gauche à droite. Les points estimés lors des étapes précédentes apparaissent en bleu et sont pris en compte dans l'estimation du plan de régression (modélisé par un segment noir).*

Le plan de régression utilisé pour estimer la profondeur en s_0 passe par le barycentre du nuage de points. L'effet d'ondulation est dû aux variations de ce barycentre. Afin de corriger cet effet, nous utilisons le plan *tangent* qui est un plan parallèle au plan de régression et qui passe par le barycentre du voisinage direct. La figure 1.32 illustre l'utilisation de ce plan tangent dans l'interpolation de l'exemple précédent. Avec cette méthode, nous obtenons bien la forme parabolique souhaitée.

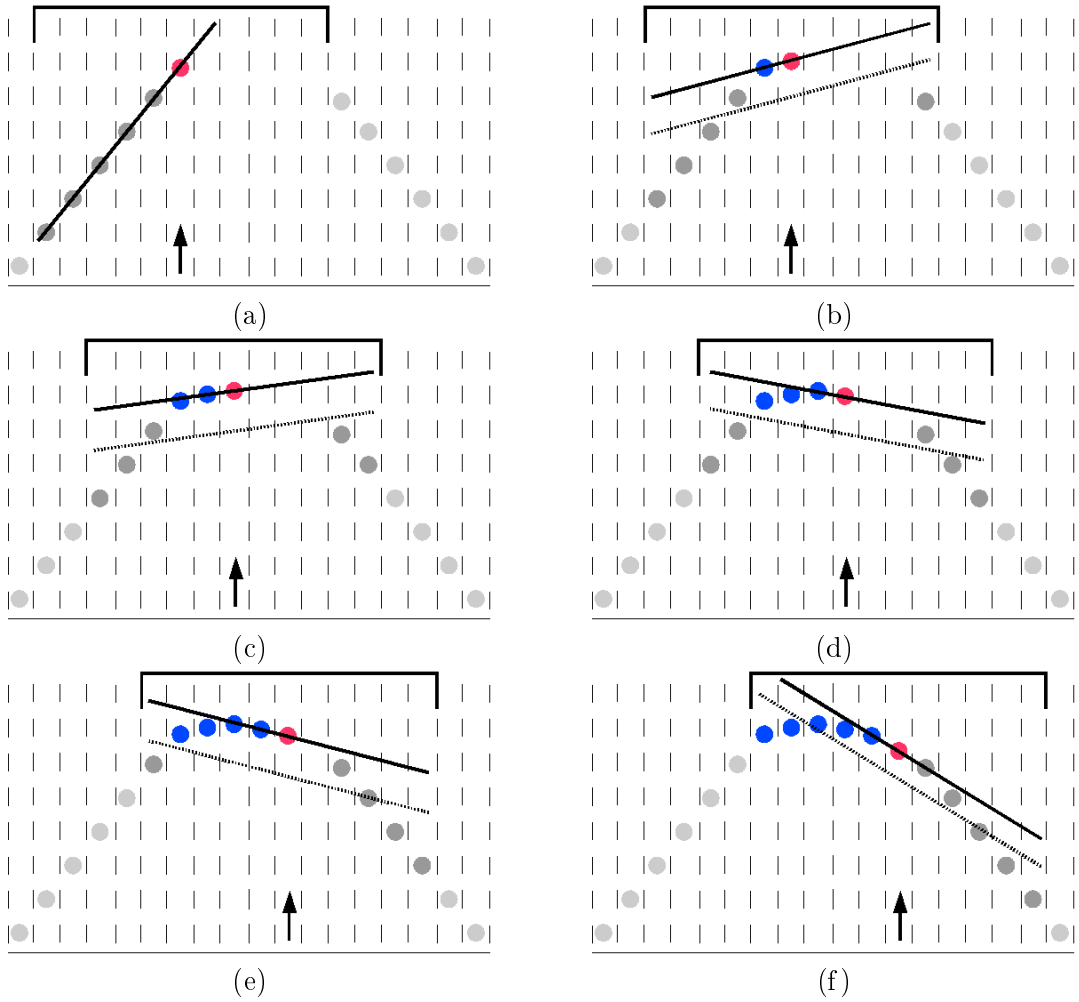


FIG. *1.32: Comme pour la figure 1.31 la position à évaluer est marquée par la flèche et la taille du masque est délimitée par la zone en haut des images. De même, l'estimation est faite de gauche à droite. Les points estimés lors des étapes précédentes apparaissent en bleu et sont pris en compte dans l'estimation du plan de régression (en pointillé). Le plan servant de support à l'interpolation est représenté par un segment continu, il est parallèle au plan de régression et passe par les points adjacents à la position à évaluer.*

L'utilisation d'un support dynamique soulève une nouvelle problèmatique : le fait d'ajouter des nouveaux points au fur et à mesure du calcul modifie constamment le voisinage utilisé pour l'estimation d'une position. En effet, dans l'exemple (figure 1.32) l'évaluation a été faite de la gauche vers la droite. Elle aurait pu être faite de la droite vers la gauche ou en alternant une insertion de chaque côté. L'utilisation d'une telle stratégie de remplissage nécessite donc de s'interroger sur l'ordre d'évaluation et sur son impact dans la surface calculée.

1.4.3 Étude de l'ordre de propagation dans la stratégie de remplissage

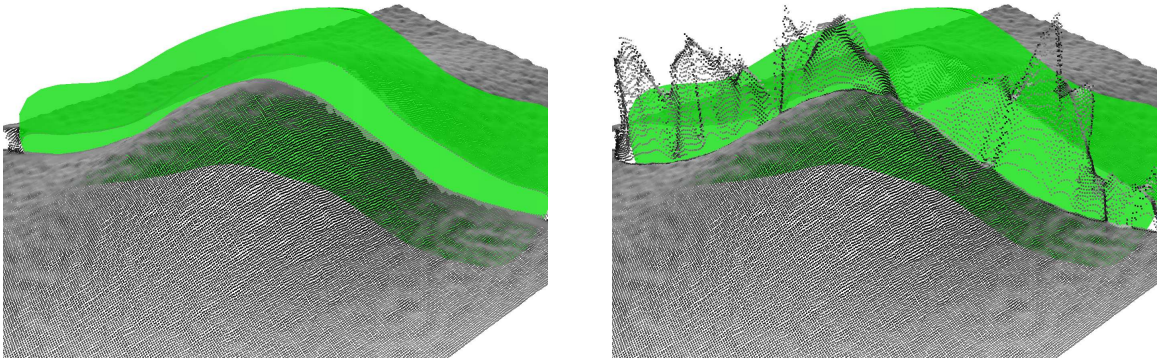
La méthode la plus répendue pour étendre les données est l'approche dite par *dilatation* : on fait croître les bordures de l'image de façon itérative jusqu'à boucher tous les trous. Cette méthode a l'intérêt d'être très intuitive et simple à mettre en œuvre, mais présente l'inconvénient de traiter de façon similaire les zones stables et le zones bruitées.
C'est pourquoi nous proposons de modifier l'approche par dilatation en introduisant un ordre de propagation de façon à optimiser la qualité de la surface reconstruite.

Pour traiter les zones de confiance en premier, l'ensemble des points de bordure sont stockés dans une file de priorité. Cette file est triée suivant un critère de qualité calculé pour chaque point candidat. Ce critère de qualité peut être défini par variance de l'erreur au plan de régression :

$$\varepsilon_0 = \frac{1}{n}\sum_{i=1}^{n}(\hat{z}(s_i) - z(s_i))^2$$

Initialement, la file de priorité contient tous les points de bordure triés selon leurs critères de qualité. Le processus d'interpolation consiste alors à vider la file en insérant dans la surface le premier élément de celle-ci. L'ajout de cet élement permet d'ajouter des nouveaux points de bordure dans la file.
Idéalement, l'insertion d'un point dans la surface devrait entraîner un nouveau tri de la file : si le voisinage d'un point de la file est modifié, il faut refaire son estimation et donc recalculer son critère d'erreur. En pratique, cette mise-à-jour de la file est trop coûteuse, à la place nous considérons les critères de qualité comme constants et le plan d'estimation d'un point est recalculé avant son insertion.

Pour bien illustrer l'impact de ce critère sur la surface résultat, nous présentons dans la figure 1.33 le résultat de deux interpolations sur une surface légèrement bruitée : la première insère les points de meilleur qualité (ε_0 le plus petit possible) en premier (voir figure 1.33 (a)), et la seconde insère les pires points en priorité (voir figure 1.33 (b)).



(a) (b)

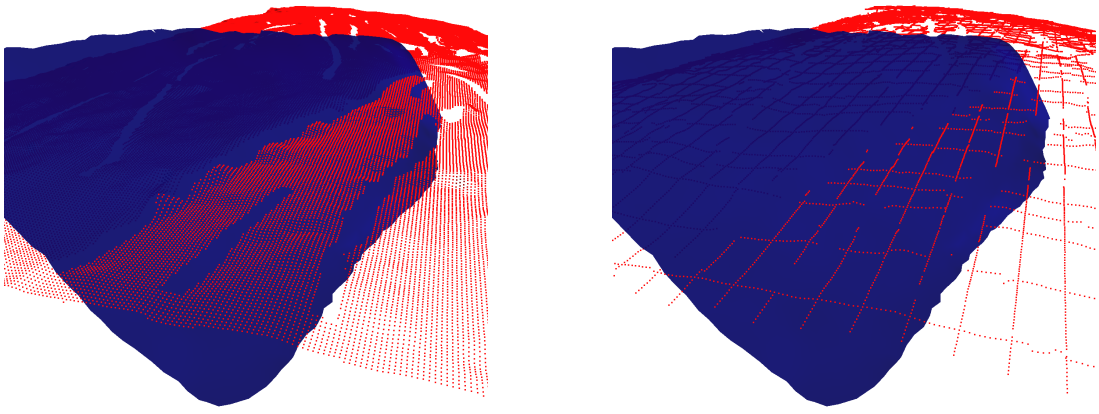
FIG. *1.33: Influence du critère de propagation. (a) La priorité est donnée aux points dont l'erreur ε_i est la plus faible. (b) La priorité est donnée aux points dont l'erreur ε_i est la plus élevée.*

On voit clairement l'impact du critère de qualité sur les surfaces interpolées. Dans la figure de gauche, le contact entre l'horizon et la faille est stable, tandis que dans la figure de droite la surface est très ondulée.

1.4.4 Application des méthodes d'interpolation contraintes par les failles

Nous avons vu les méthodes utilisées par les géophysiciens pour extraire les informations décrivant les horizons dans la sismique. Il faut différencier deux cas :

- si la sismique est de bonne qualité, le géophysicien aura pu propager l'horizon, générant un nuage de points dense excepté à proximité des failles (voir figure 1.34 (a)),
- en revanche si la qualité de la sismique ne permet pas l'utilisation d'outils automatiques, le pointé sera réalisé à la main générant ainsi un nuage de points éparse (voir figure 1.34 (b)).



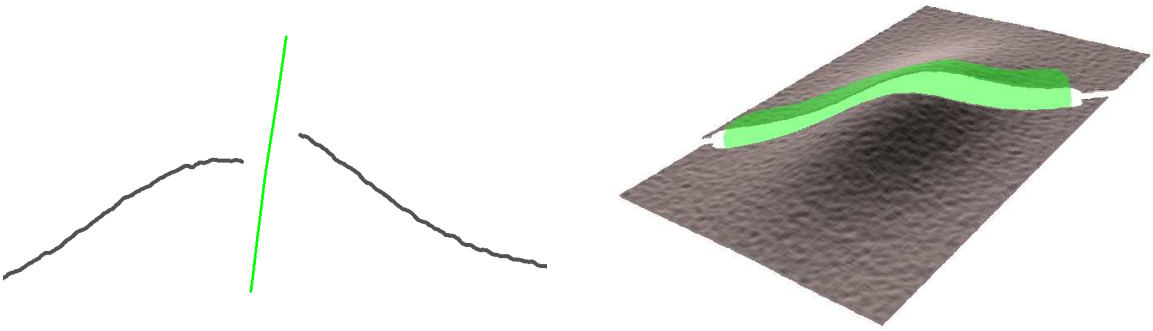
(a) (b)

FIG. *1.34: (a) Horizon propagé par outil un automatique (b) Horizon issu d'un pointé manuel*

Dans cette section, nous proposons une étude des méthodes d'interpolation présentées dans la section précédente. Nous distinguons le cas des horizons propagés par les outils automatiques que nous qualifierons de *denses* et des horizons issus du pointé manuel *éparses*. Afin d'évaluer les méthodes appliquées aux cas denses nous avons choisi deux jeux de données : un synthétique et un issu d'une modélisation réelle.

Cas synthétique dense

Ce cas a été généré synthétiquement à l'aide d'une gaussienne. Nous avons ensuite simulé un glissement dû à une faille plane et introduit un bruit blanc gaussien dans la surface décalée. La dernière étape a consisté à détruire l'information à proximité de la faille, c'est cette partie détruite que les algorithmes d'interpolation devront reconstruire.
La figure 1.35 illustre cet exemple synthétique : (a) est une représentation en coupe de notre jeux de donnée synthétique et (b) est une vue d'ensemble en trois dimensions de ce jeux.



(a) (b)

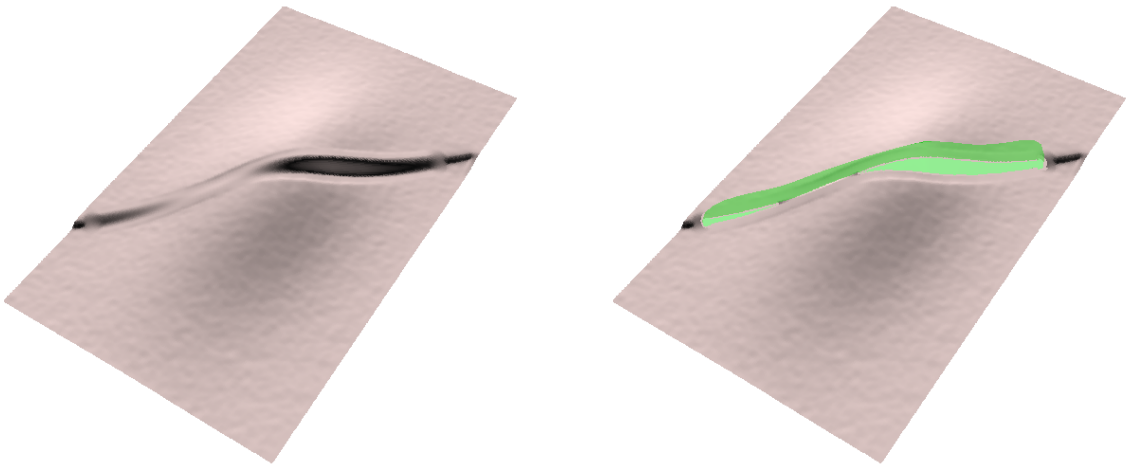
FIG. *1.35: Horizon synthétique (a) vue coupe (b) vue d'ensemble*

Les résultats des processus d'interpolation seront présentés à la fois en trois dimensions et sur cette même coupe afin de bien appréhender la qualité des surfaces obtenues.

Dans un premier temps, nous appliquons les techniques d'interpolation spatiale dites barycentriques avec un support statique. Pour ces tests, nous avons choisi la méthode "inverse distance" la plus couramment utilisée dont les coefficients λ_i sont de la forme :

$$\lambda_i = \frac{|s_i - s_0|^{-2}}{\sum_{i=1}^{n} |s_i - s_0|^{-2}}$$

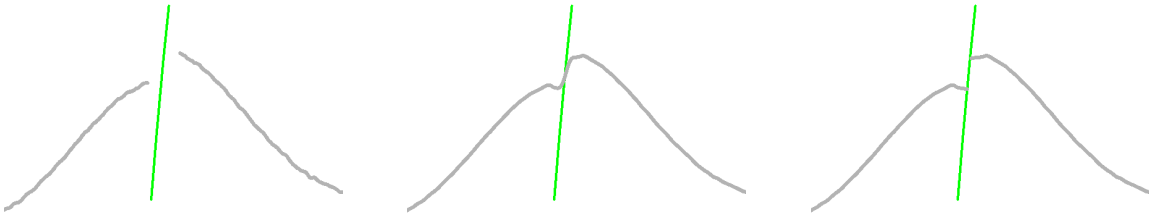
Nous comparons tout d'abord les résultats avec et sans prise en compte du réseau de failles. La figure 1.36 illustre les résultats en trois dimensions des deux interpolations, et nous pouvons voir que la prise en compte de la faille (voir figure 1.36 (b)) donne un résultat plus géologique.



(a) (b)

FIG. *1.36: Méthode barycentrique (a) Utilisation d'un voisinage classique (b) Utilisation du voisinage prenant en compte les failles (ici une faille en vert)*

Afin de mieux illustrer ce résultat, la figure 1.37 présente une vue coupe des interpolations où nous pouvons mieux apprécier comment l'horizon est calé sur la faille.

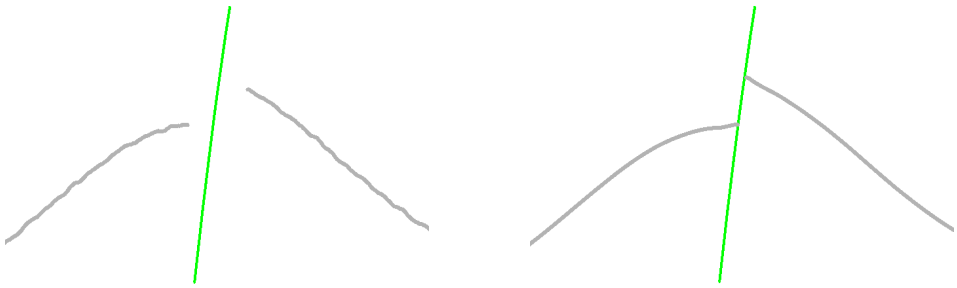


(a) (b) (c)

FIG. *1.37: Application de la recherche du voisinage partiel sur une méthode "inverse distance". La figure présente différents résultats sous la forme de vue en coupe, l'horizon est représenté en gris et la faille en vert. (a) Donnée originale avant interpolation. (b) Interpolation sans prise en compte de la faille. (c) Interpolation avec prise en compte de la faille. On remarque que l'arrêt sur la faille est propre, mais on voit apparaître une ondulation dans la surface à proximité de celle-ci.*

Ce résultat nous permet de démontrer l'intérêt de prendre en compte les contraintes de failles afin d'obtenir un résultat plus géologique. Cependant nous pouvons observer sur ces données l'apparition d'une ondulation liée aux méthodes barycentriques.

Afin d'éviter ces ondulations indésirables nous allons utiliser un estimateur basé sur les régressions locales. Nous allons utiliser le modèle de plan décrit précédemment. La figure 1.38 illustre la même vue en coupe de l'horizon synthétique interpolé que pour l'évaluation des méthodes barycentriques de la figure 1.37.



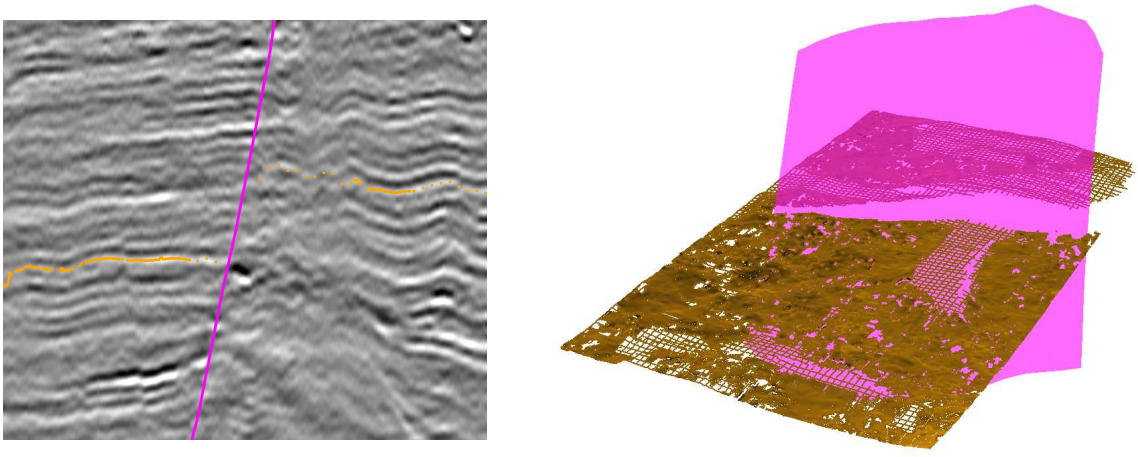
(a) (b)

FIG. *1.38: Vue en coupe de l'horizon synthétique interpolé avec une méthode de régression avec un modèle plan.*

Le résultat est nettement plus satisfaisant car l'interpolation suit la pente locale (appelée pendage) et ne crée pas d'ondulations qui n'ont pas de réalité géologique.

Cas réel dense

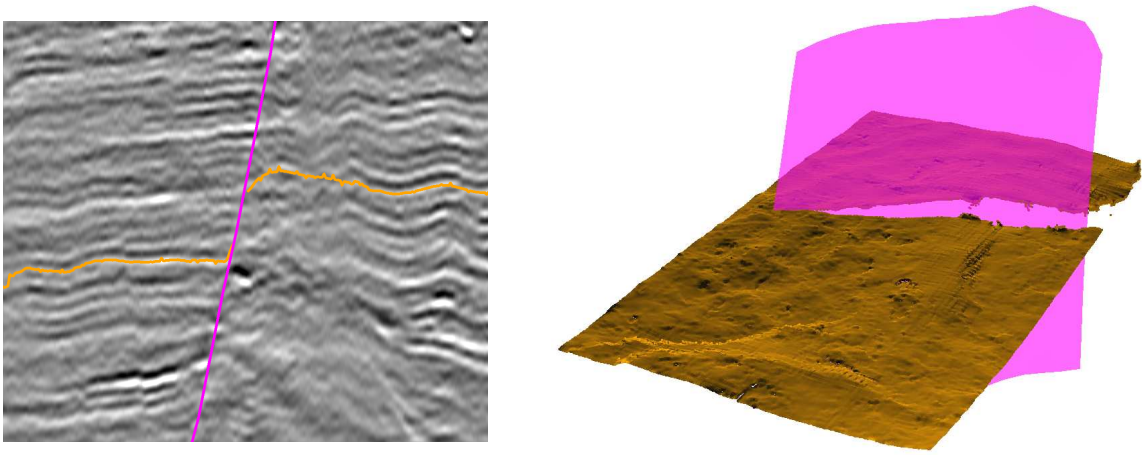
Ce cas est extrait d'une étude opérationnelle où nous nous intéresserons à l'interpolation d'un horizon contraint par une faile. Ce cas est intéressant car il mélange un pointé manuel dense et des données issues d'outils automatiques. Les techniques d'interpolation devront donc remplir à proximité de la faille mais aussi entre le pointé manuel. La figure 1.39 présente deux vues de ce jeux données, une sur section sismique et une vue d'ensemble en trois dimensions.



(a) (b)

FIG. *1.39: Horizon réel (a) vue coupe sur section sismique (b) vue ensemble*

Nous montrons dans la figure 1.40 le résultat de l'interpolation.



(a) (b)

FIG. *1.40: Régression locale avec un modèle plan. (a) Vue sur une coupe sismique de l'horizon interpolé (en orange) en contact avec la faille (en rose). (b) Vue globale en 3 dimensions.*

La vue d'ensemble révèle des instabilités dans la surface reconstruite. Un agrandissement de certains de ces défauts est visible dans la figure 1.41. Ces instabilités sont dûes au calcul du plan de régression local quand le voisinage est composé de très peu de points dont certains sont bruités. Le plan prendra alors en compte les points bruités qui peuvent introduire un fort pendange dans la surface.

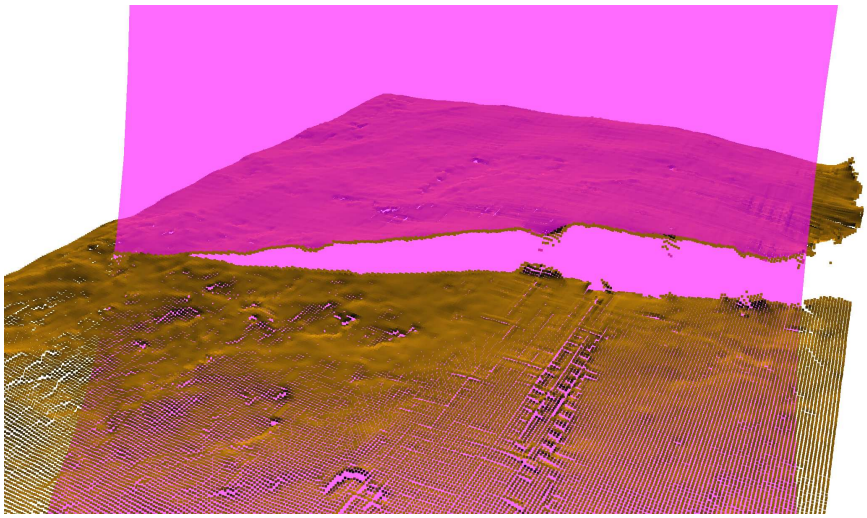


FIG. *1.41: Problème de stabilité du plan de régression local quand le voisinage contient peu de données observées.*

Ces instabilités sont liées à la présence de points bruités dans des zones contenant peu de points. Afin de résoudre ce problème, nous utilisons une stratégie de remplissage à support dynamique contrôlée par le critère de qualité décrit dans la section 1.4.3. Le résultat de cette interpolation est visible dans la figure 1.42).

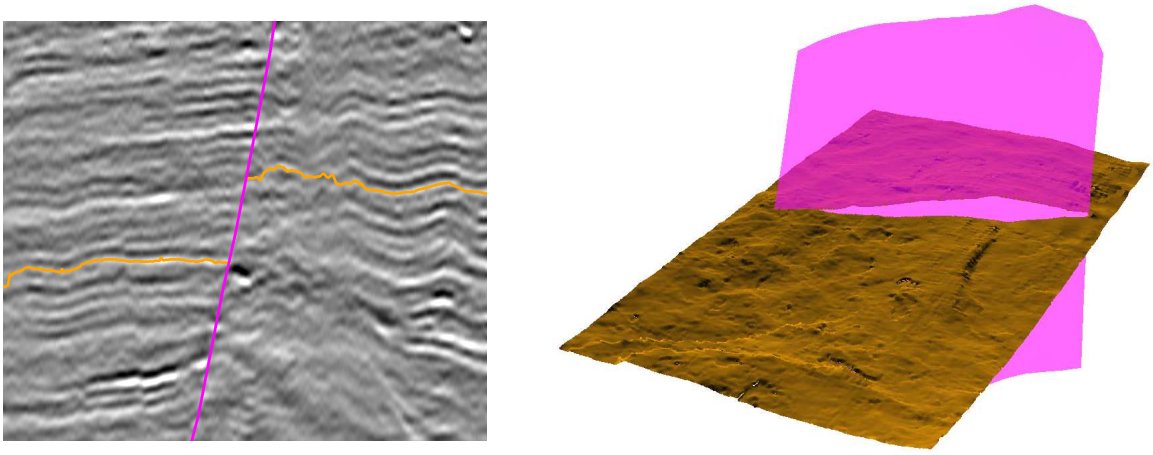


FIG. *1.42: Interpolation du cas réel avec un estimateur par régression et une stratégie de remplissage avec un support dynamique.*

La figure 1.43 présente un agrandissement des zones qui avaient des problèmes de stabilité dans l'interpolation avec une startégie de remplissage à support statique (voir figure 1.41).

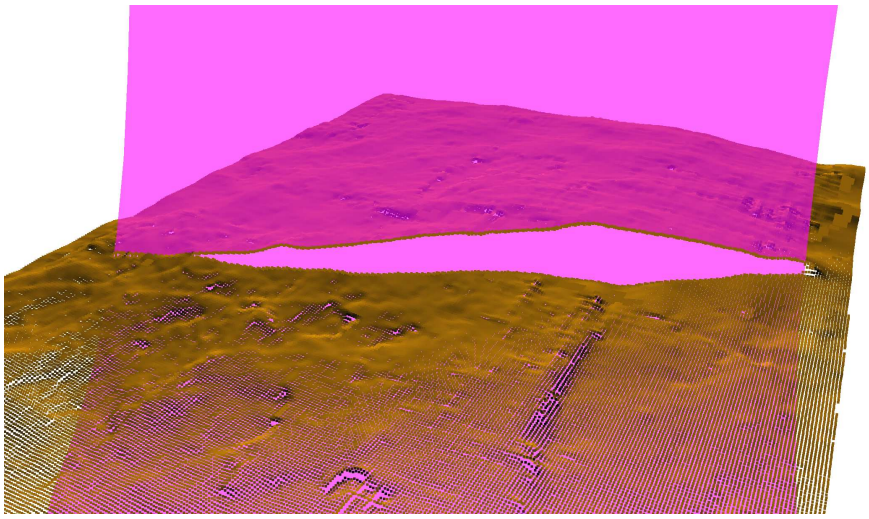


FIG. *1.43: L'utilisation d'une stratégie de remplissage dynamique résout les problèmes de stabilité dans les zones bruitées*

Horizons éparses

Afin de pouvoir évaluer les méthodes d'interpolation sur des données éparses, nous avons décimé l'horizon issu d'un cas d'étude (voir figure 1.44). Les techniques les plus utilisées pour interpoler ce type de données sont les techniques de krigeage.

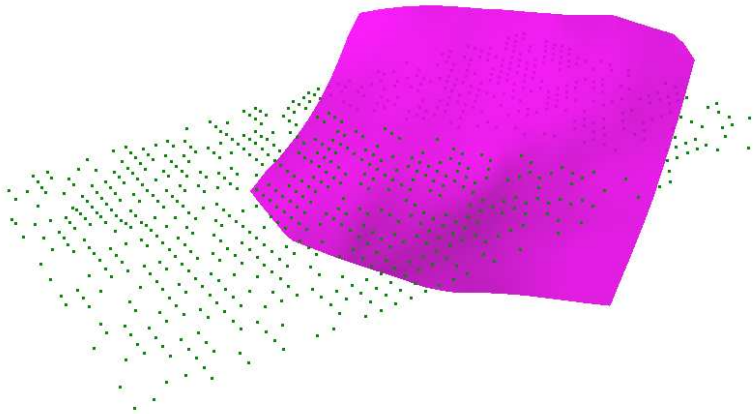
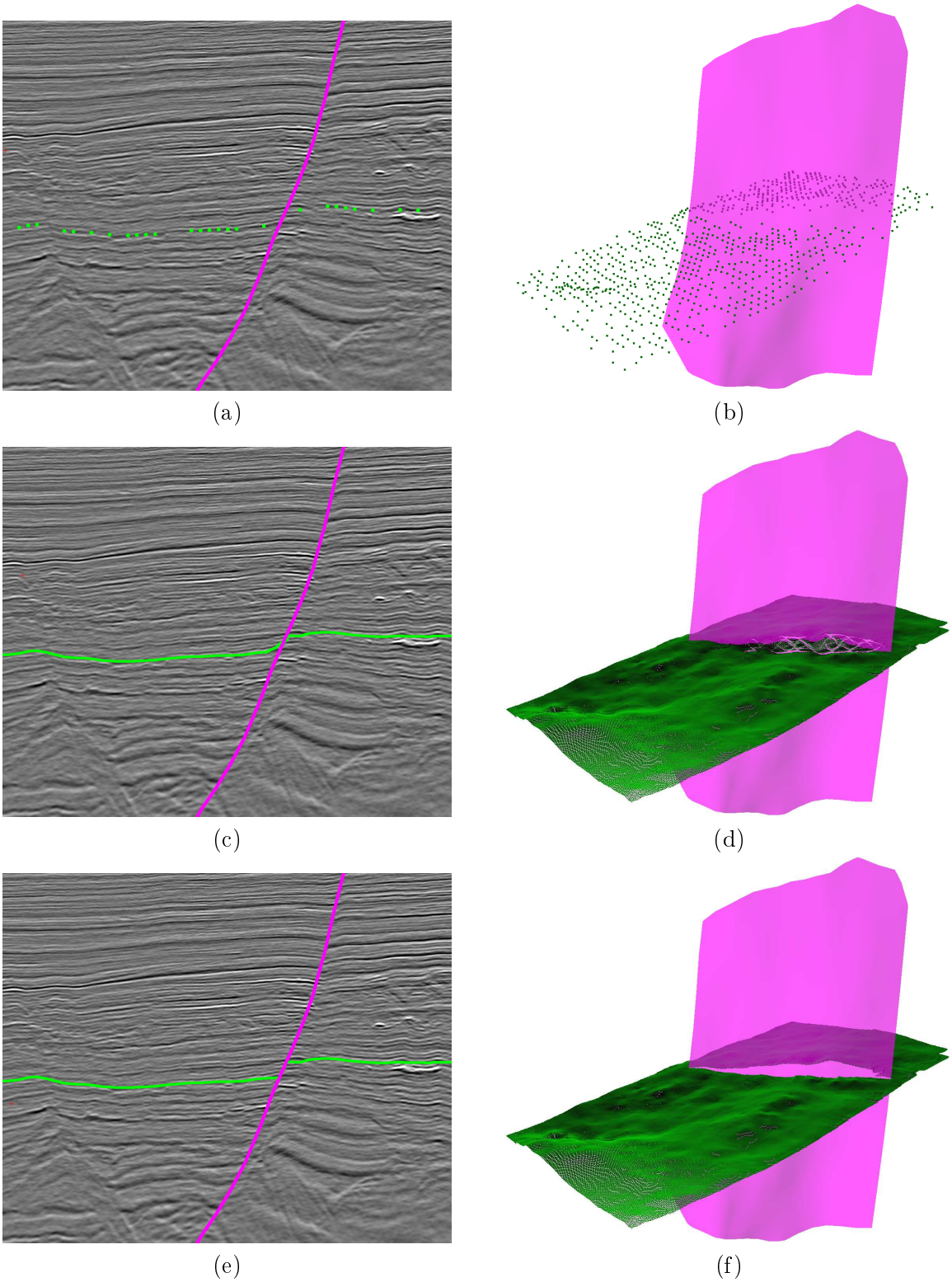


FIG. *1.44: Horizon réel décimé*

La figure 1.45 résume les résulats expérimentaux de l'application de notre technique de remise en cause du voisinage sur un krigeage ordinaire. Sur la gauche les figures sont des vues en coupe des données, elles montrent l'image sismique, la faille triangulée (en rose) et l'horizon (en vert). Sur la droite, nous avons une vision globale de la scène en trois dimensions.

Les figures (a) et (b) montrent les données observées sur lesquelles sont appliquées la technique d'interpolation. Les figures (e) et (f) illustrent le résulat du krigeage en tenant compte de la faille, contrairement aux figures (c) et (d) qui présentent le résultat du calcul si la faille n'est pas prise en compte. On peut constater que les résultats des deux interpolations sont identiques exceptés à proximité de la faille dont la lèvre est bien dessinée si la faille est prise en compte lors du calcul.



(a) (b)

(c) (d)

(e) (f)

FIG. *1.45: Application de notre technique au krigeage*

1.5 Conclusion

Nous avons présenté dans ce chapitre une méthodologie complète permettant de construire un modèle structural cohérent composé de failles et d'horizons. La première étape consiste à reconstruire chaque faille indépendemment les unes de autres. Pour cela nous avons proposé une méthodologie robuste basée sur une triangulation de Delaunay contrainte des données projetées dans leurs plans de régressions. La triangulation grossière obtenue est ensuite lissée en utilisant les surfaces de subdivisions. Ce processus a été mis en œuvre à l'aide de la bibliothèque CGAL [1].

La seconde étape est la mise en cohérence globale du réseau de failles. Nous avons proposé une solution dite "surfacique" qui manipule les surfaces triangulées modélisant les failles afin de les corriger. Une nouvelle thèse va débuter pour continuer nos travaux sur la correction du réseau de failles grâce à une approche "volumique".

La dernière étape dans la construction du modèle est l'interpolation des nuages de points marquant la position des horizons dans la sismique. Nous avons proposé d'introduire les discontinuités dûes aux failles grâce à un nouveau voisinage dit "partiel". Il permet d'estimer les points en ne tenant compte que de points "visibles" à travers le réseau de failles. Nous avons aussi montré l'influence de la stratégie de remplissage lors du calcul des horizons et proposé une approche dynamique robuste au bruit.

Les différentes étapes de cette méthodologie sont intégrées dans SISMAGE© et ont été utilisées avec succès sur plusieurs cas complexes. Les figures 1.46 et 1.47 illustre le résultat de la construction d'un modèle structural composé de 35 failles et d'un horizon fortement bruité. Dans la figure 1.47, l'horizon brut (avant interpolation) a dû être nettoyé autour du réseau de failles : la sismique était flou à proximité des failles et le propagateur d'horizon avait introduit du bruit. Pour cela nous avons effacé les points "trop" proches du réseau, des travaux sont en cours pour proposer des outils de diagnostique et de traitement de ce bruit.

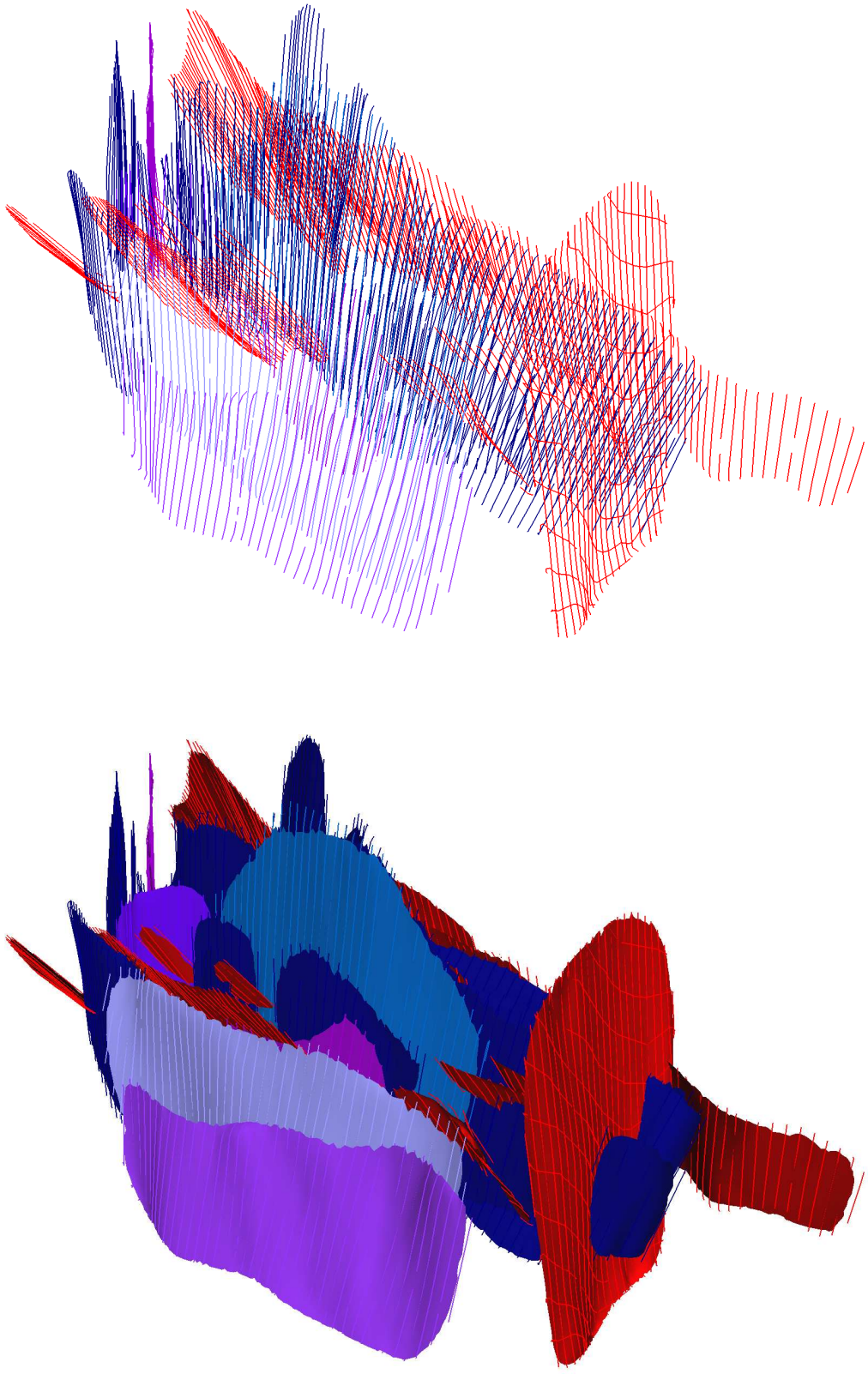


FIG. *1.46: Construction d'un réseau composé de 35 failles*

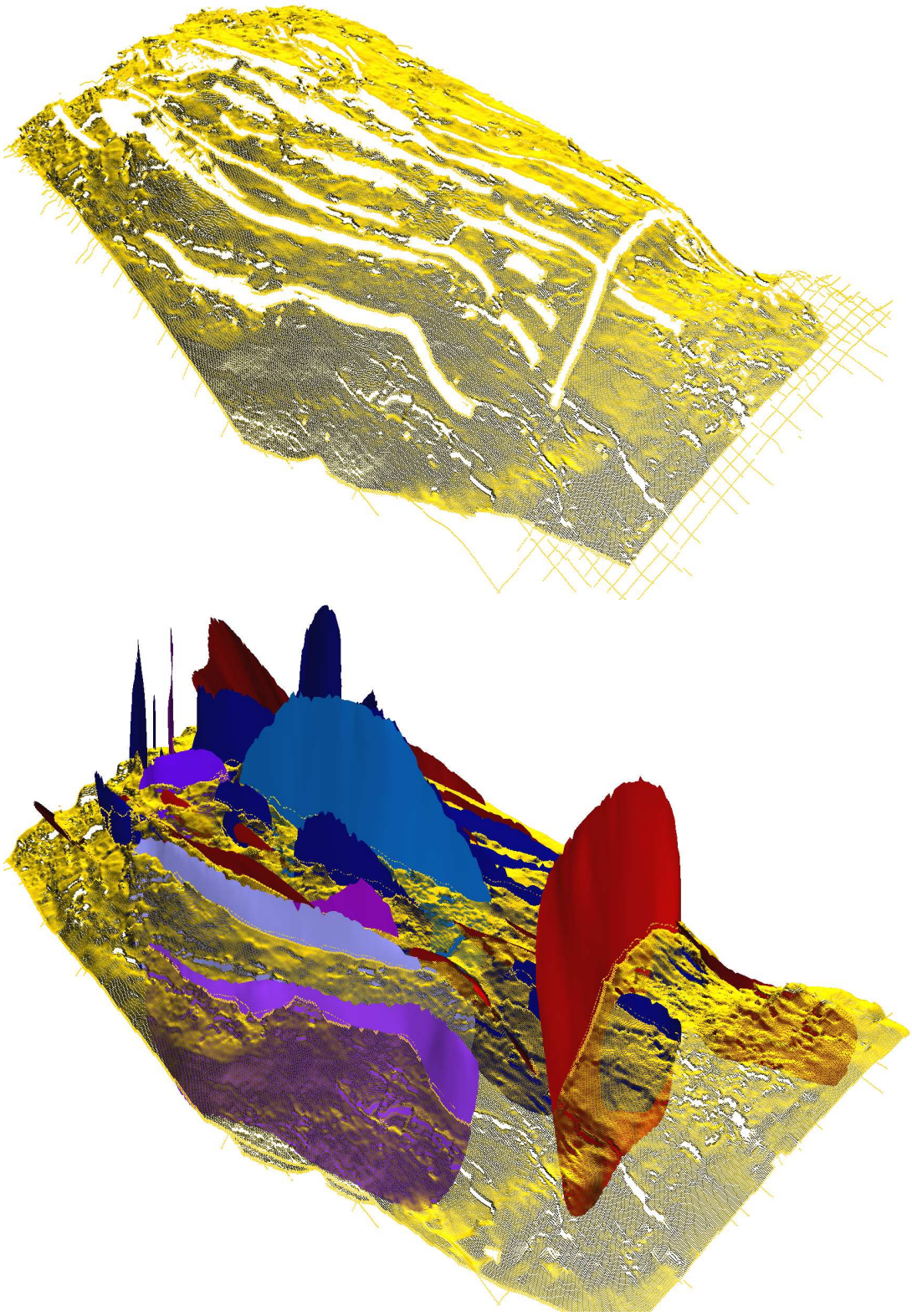


FIG. *1.47: Interpolation d'un horizon bruité avec un réseau de failles complexe*

Chapitre 2

Extraction des corps dans les cubes attributs

Sommaire

2.1 Contexte d'interprétation géophysique 58
2.1.1 Attribut sismique 58
2.1.2 Lien entre l'enveloppe et l'isosurface 60
2.1.3 Contraintes liées aux géosciences 61
2.2 Quelques notations 63
2.3 Algorithme des *marching cubes* 63
2.4 Adaptation des *marching cubes* aux contraintes du métier 65
2.4.1 Génération d'une 2-variété 65
2.4.2 Gestion des ambiguïtés 67
2.5 Simplification des isosurfaces 70
2.5.1 Approches par décimation adaptative des cubes 70
2.5.2 Simplification en post-traitement 72
2.5.3 Approches mixtes 78
2.6 Parallélisation des méthodes d'extraction d'isosurfaces 85
2.7 Extension du *tandem algorithm* 87
2.7.1 Migration des composants 87
2.7.2 Algorithme parallèle 90
2.7.3 Implémentation maître/esclave 94
2.8 Évaluation de l'algorithme 96
2.8.1 Consommation mémoire 96
2.8.2 Qualité du maillage 97
2.8.3 Temps de calcul 98
2.9 Conclusion 101

Dans ce chapitre, nous abordons la problématique de reconstruction de *corps*. Nous avons évoqué en introduction deux problèmes à résoudre :

- le premier concerne la reconstruction de l'enveloppe du corps à partir du pointé manuel. Ce problème n'a pas été abordé dans le cadre de cette thèse et reste un problème ouvert malgré quelques travaux récents [11, 61].
- le second concerne la reconstruction en trois dimensions de l'enveloppe du corps à partir d'un bloc attribut. Un volume attribut, et par contraction un *attribut*, est un volume de données pouvant mettre en évidence des propriétés physiques particulières issues de la sismique. La résolution de ce problème est présentée dans la suite de ce chapitre.

Dans un premier temps, nous donnons une définition des attributs sismiques et précisons leurs utilisations. Nous présentons ensuite les méthodes permettant d'extraire une surface d'un volume. Enfin, nous verrons quelles sont les limites des méthodes actuelles telles que le *tandem algorithm* [6] et comment les adapter à notre problématique.

2.1 Contexte d'interprétation géophysique

2.1.1 Attribut sismique

Définition d'un attribut sismique

Les attributs sismiques ont été popularisés par Taner à la fin des années 70 qui les définit comme "toute l'information obtenue à partir des données sismiques, soit par mesure directe, soit par un raisonnement logique ou basé sur l'expérience" [54].
En d'autres mots, un attribut sismique peut être défini comme la mise en évidence d'une propriété particulière de la donnée sismique.

Une grande famille d'attributs sismiques est dévolue à la détection de faciès sismiques. Un faciès sismique est une région (ou volume) de l'image sismique ayant des caractéristiques homogènes. La littérature identifie six grands types de faciès sismiques visibles dans la figure 2.1. Mesurer la probabilité de présence de tel ou tel faciès constitue un attribut sismique.
La détection de corps consiste alors à déterminer l'enveloppe en trois dimensions isolant dans le cube sismique toutes les régions ayant le faciès recherché.

Pour illustrer les algorithmes de détection mis en œuvre dans le cadre de cette thèse, nous allons nous intéresser à la détection de faciès chaotiques avec l'attribut *iso5* [38]. L'algorithme de détection est bien sûr applicable à tout autre type d'attribut, [77] montre un exemple d'application sur un autre type d'attribut.

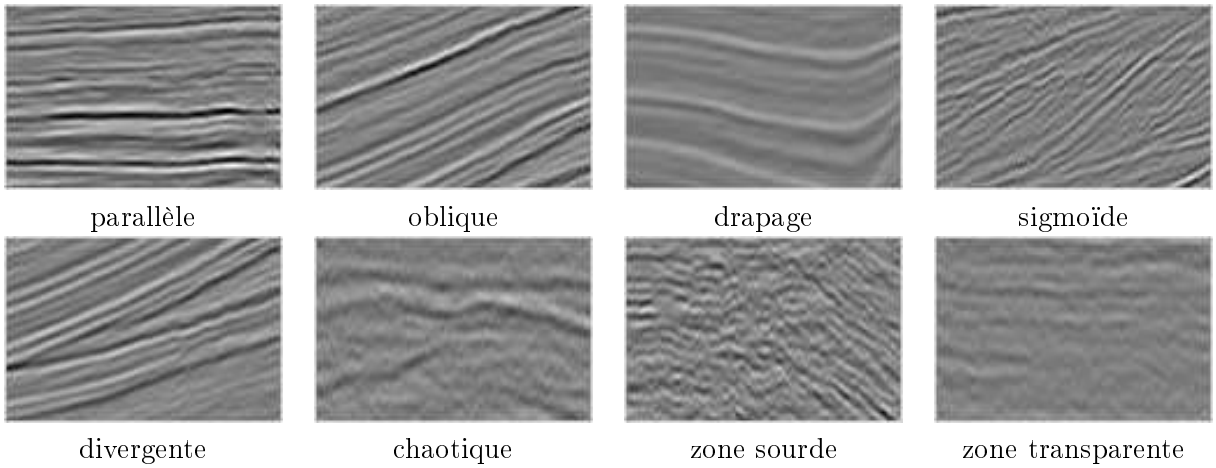


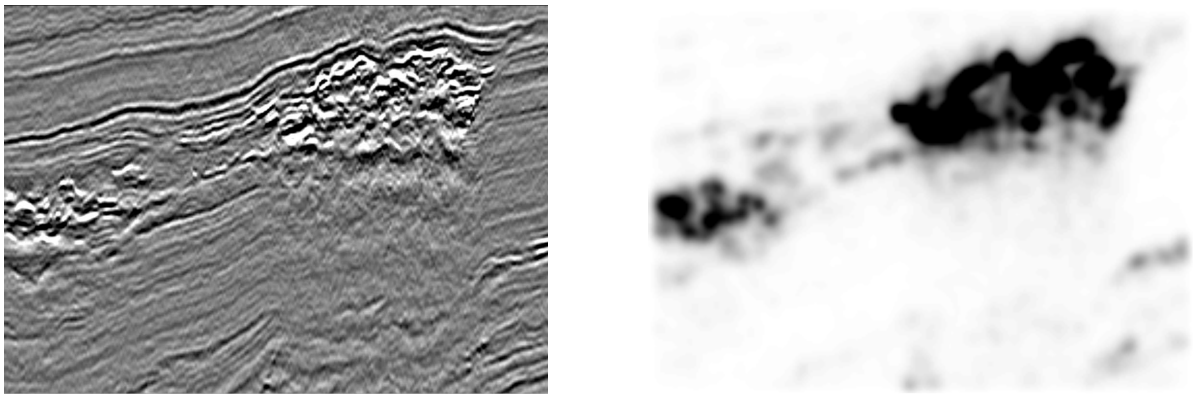
FIG. *2.1: Les différents faciès sismiques*

Exemple d'attribut : iso5

Au sein de l'équipe SISMAGE© de nombreux attributs sismiques ont été développés, comme par exemple l'attribut iso5, qui permet la détection de faciès chaotiques sur l'image (voir figure 2.2). Cet attribut mesure localement le degré d'anisotropie des vecteurs gradients de l'image. Cette mesure est définie par :

$$iso5(s_0) = \frac{\sum_{s \in \mathcal{V}(s_0)} ||\vec{\nabla}(s_0) \wedge \vec{\nabla}(s)||}{\sum_{s \in \mathcal{V}(s_0)} ||\vec{\nabla}(s)||}$$

où s représente un site, c'est-à-dire un pixel de l'image sismique, $\vec{\nabla}(s)$ est le vecteur gradient estimé en s et $\mathcal{V}(s_0)$ définit un voisinage du site s_0.



(a) (b)

FIG. *2.2: Exemple d'attribut sismique : iso5 (a) vue section d'un volume sismique (b) vue de la même section de l'attribut iso5 calculé sur cette sismique*

Ces régions chaotiques de forte énergie sont associées à des zones de chenaux turbiditiques qui peuvent être, dans certaines cas, associées à des réserves d'hydrocarbure. Il est alors primordial d'extraire les surfaces enveloppant les réservoirs afin de pouvoir en analyser les géométries : estimation des volumes, analyse des connectivités, etc.

2.1.2 Lien entre l'enveloppe et l'isosurface

Un attribut sismique peut être défini comme un volume discret mesurant la probabilité de détection d'un faciès sismique particulier. Ce volume est alors associé à une fonction f :

$$\begin{array}{rccc} f: & \mathbb{Z}^3 & \rightarrow & [0,1] \\ & (x,y,z) & \rightarrow & f(x,y,z) \end{array}$$

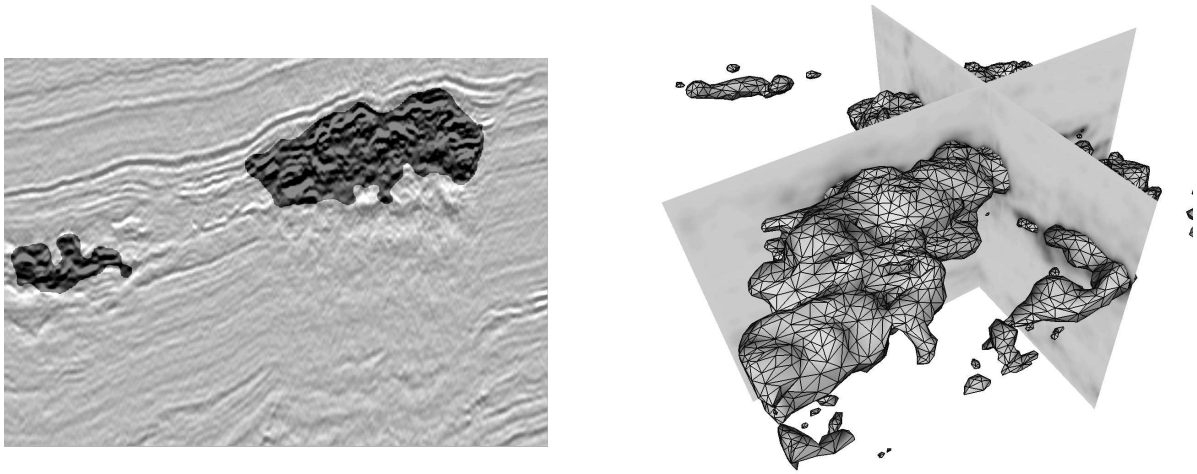
Déterminer l'enveloppe des corps en trois dimensions consiste donc à détecter l'enveloppe de tous les points (x, y, z) du volume vérifiant :

$$f(x,y,z) \geq \alpha$$

avec $\alpha \in [0,1]$ une valeur seuil au delà de laquelle nous considérons que la détection est sûre. L'enveloppe du corps est alors définie par l'ensemble des points vérifiant :

$$f(x,y,z) = \alpha$$

et est définie comme l'isosurface de l'attribut à la valeur α : le préfixe "iso" vient du grec et signifie égal, une isosurface est par définition une surface définie par des points ayant une propriété égale. La figure 2.3 illustre cette détection sur l'attribut sismique iso5.



(a) (b)

FIG. *2.3: Reconstruction de surfaces à partir de l'attribut iso5. (a) Vue de section la sismique avec en noir l'ensemble des points définissant le corps. (b) Reconstruction de l'enveloppe.*

Le rôle d'un algorithme d'extraction d'isosurface est d'extraire la surface polygonale (généralement triangulée) qui passe au mieux par cet ensemble de points. Ces méthodes d'extraction ont largement été étudiées, et la méthode la plus répendue est l'algorithme des *marching cubes*.

Un peu d'histoire L'algorithme des *marching cubes* a été présenté au SIGGRAPH de 1987 [64] mais daterait en réalité de 1984. Bill Lorensen (voir figure 2.4) et Harvey Cline auraient eu l'idée de la méthode suite à un séminaire de Carl Crawford[2] vantant les mérites d'un futur moteur graphique, le *Graphicon*, basé sur le rendu de polygones. Il semblerait que le fait que le *Rubick's Cube* soit à la mode à cette période ne soit pas étranger au concept. Depuis, l'algorithme des *marching cubes* est certainement la méthode la plus utilisée pour extraire et visualiser une isosurface. De ce fait une littérature abondante est disponible traitant des optimisations et évolutions de cet algorithme.

FIG. *2.4: Bill Lorensen et Dick Bair regardant le résultat d'une simulation en éléments finis.*

2.1.3 Contraintes liées aux géosciences

Les contraintes liées à notre domaine d'application nécessite d'adapter les méthodes traditionnelles. En effet, les évolutions des méthodes d'acquisitions sismiques permettent de couvrir des zones de plus en plus étendues avec des résolutions croissantes. La taille des données sismiques est donc croissante et dépasse souvent plusieurs dizaines de giga-octets. Les corps construits par

[2]rien à voir avec le joueur de baseball qui aurait eu un peu moins de 3 ans à l'époque

extraction d'isosurfaces sur ces volumes sont donc de plus en plus complexes (la complexité étant croissante avec la taille des zones couvertes). La méthode d'extraction proposée dans cette thèse génère une version simplifiée de ces corps permettant de les afficher sur une station de travail.

Par ailleurs, dans la méthodologie classiquement utilisée en géoscience (voir figure 2.5), le modèle structural construit dans le chapitre précédent peut-être utilisé comme contrainte de coupe pour séparer les corps. Ces contraintes sont introduites dans le processus en modifiant le volume : les valeurs des sommets appartenant aux cellules intersectant les surfaces du modèle structural sont mises à 0 (probabilité d'avoir un faciès donnée). Notons que l'introduction de ces contraintes entraine l'apparition d'arêtes vives dans les surfaces reconstruites.

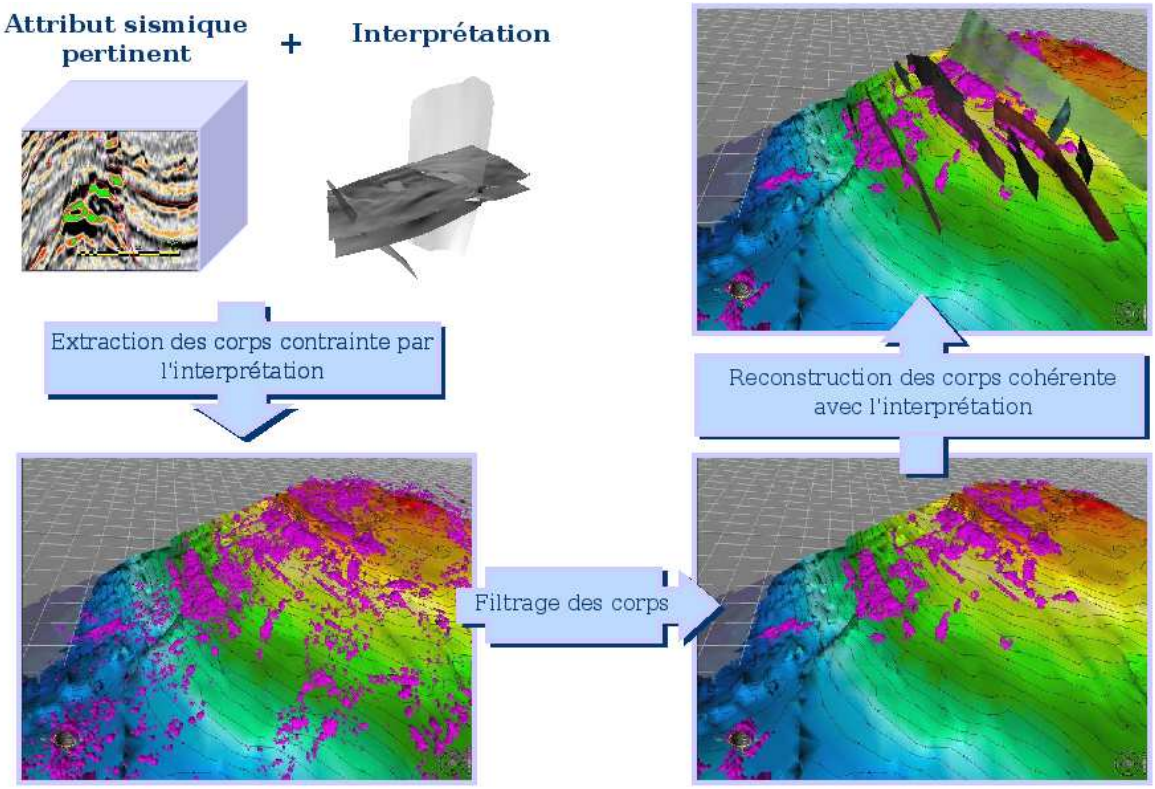


FIG. *2.5: Méthodologie d'extraction des corps contrainte par le modèle structural*

La dernière étape de la méthodologie consiste à filtrer les corps obtenus suivant de nombreux critères tels que la direction de dépôt, la pente, la position, le volume, etc. Nous proposons à la fin de ce chapitre un format de description des surfaces extraites permettant une selection aisée de ces composantes sans avoir à traiter l'ensemble de leurs géométries.

2.2 Quelques notations

Nous définissons ici quelques termes que nous allons utiliser dans la suite de ce chapitre. La figure 2.6 illustre ces différents concepts :

- un **voxel** (volumetric pixel) est le composant élémentaire du volume. Il fait référence à un élément de volume de dimensions $d_i \times d_j \times d_k$ et au point entier (i, j, k) du pavage $\mathbb{Z}^3$.
- une **coupe** k du volume est l'ensemble des voxels correspondant à un même k.
- une **cellule** (i, j, k) est la réunion des 8 voxels $(i+\epsilon_1, j+\epsilon_2, k+\epsilon_3)$ avec $(\epsilon_1, \epsilon_2, \epsilon_3) \in \{0, 1\}$.
- les **sommets** d'une cellule sont les 8 coins de celle-ci. Chaque sommet porte la valeur du voxel qui lui est associé.
- les **arêtes** relient entre-eux les sommets voisins dans une cellule.
- une **couche** k est constituée par deux coupes consécutives k et $k+1$ du volume.

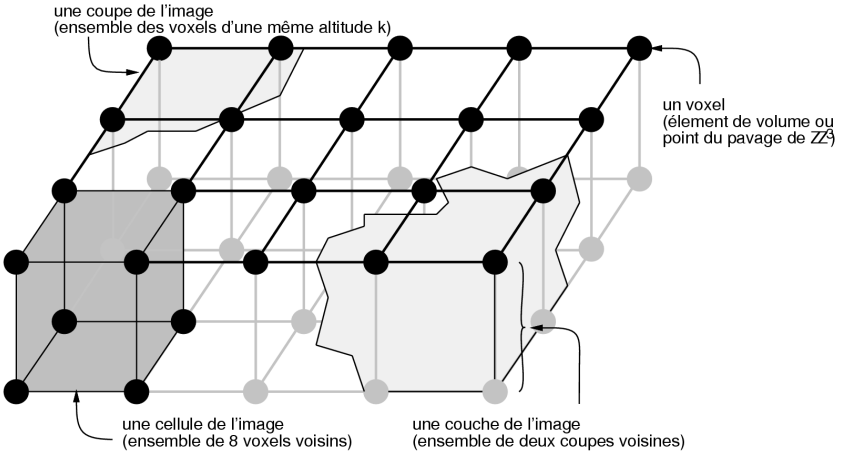


FIG. *2.6: Définitions d'un voxel, d'une coupe, d'une cellule et d'une couche dans un volume. Figure issue de [73].*

2.3 Algorithme des *marching cubes*

L'idée fondamentale de l'algorithme est de résoudre le problème complexe qu'est l'extraction d'une isosurface dans un volume 3D en résolvant une multitude de problèmes plus simples : le calcul des triangles associés à l'isosurface recherchée dans une cellule. Lorensen et Cline ont décrit un certain nombre de combinaisons de triangles pouvant être associées à une cellule. Nous allons décrire la méthodologie permettant de résoudre le problème local ainsi que les notions qui en découlent.

Méthodologie L'algorithme des *marching cubes* balaie le volume cellule par cellule. Chaque cellule est traitée indépendamment. Chaque sommet de la cellule ayant une valeur supérieure ou égale à l'isovaleur est marqué. Nous avons donc $2^8 = 256$ combinaisons de sommets marqués possibles pour une cellule. Lorensen et Cline montrent dans leur article original que ces 256

combinaisons peuvent être ramenées à 15, les autres configurations pouvant être retrouvées par réflexion et symétrie. Une arête de la cellule est considérée comme active si elle possède un et un seul sommet marqué. Afin de retrouver efficacement les arêtes actives et les triangles générés, on utilise des tables pré-calculées dont l'index est un entier où les bits sont associés à un sommet de la cellule. Si le sommet est marqué son bit associé vaut 1, sinon il vaut 0. La figure 2.7 illustre les 15 configurations possibles, les sommets marqués sont représentés par une sphère.

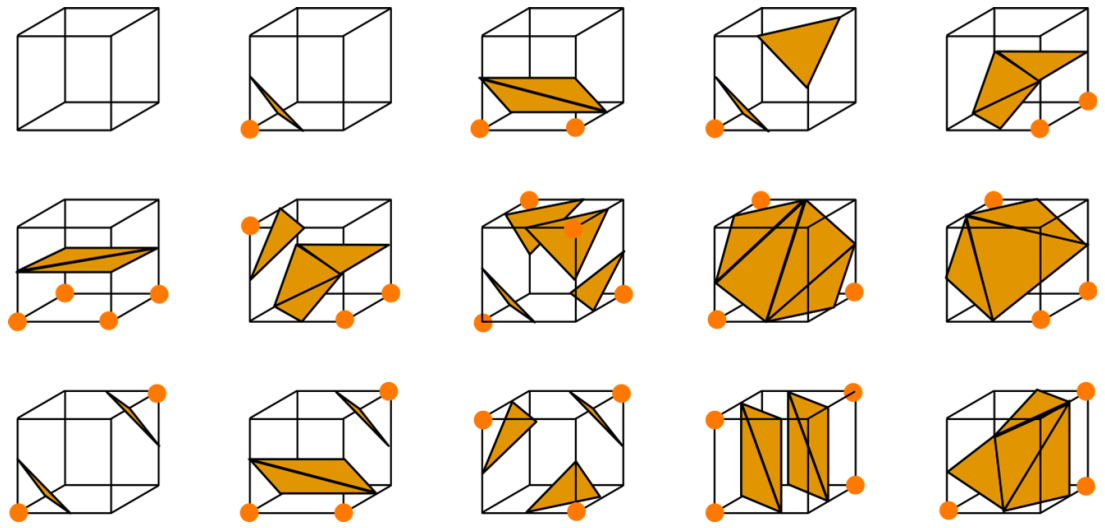


FIG. *2.7: Configurations du marching cubes*

La première étape dans la construction de la géométrie est le calcul des points d'intersections entre l'isosurface et les arêtes actives. Ces points sont calculés par interpolation linéaire. On considère deux sommets s_1, s_2 définissant une arête active ayant pour valeurs respectives v_1 et v_2. La position du point p passant par l'isovaleur α s'exprime :

$$p = s_1 + \frac{\alpha - v_1}{v_2 - v_1} \times (s_2 - s1)$$

La seconde étape est la construction des triangles.

L'algorithme des *marching cubes* est relativement simple à implémenter et fournis de bons résultats pour la visualisation. Initialement il a été appliqué sur des images volumiques qui sont des grilles structurées régulières, mais de nombreuses recherches l'ont étendu à d'autres types de grilles. La figure 2.8 montre une classification des différentes grilles [15].

Par exemple, Goldsmith et Jacobson [37] ont appliqué l'algorithme sur des grilles curvilinéaires. Dans le cas des grilles non structurées, les solutions proposées sont basées sur l'algorithme des *marching tetrahedra* [83] dont le principe est similaire aux *marching cubes*. L'apport des différents articles étant la découpe des cellules en tétraèdres, ainsi que l'interpolation des données sur ces

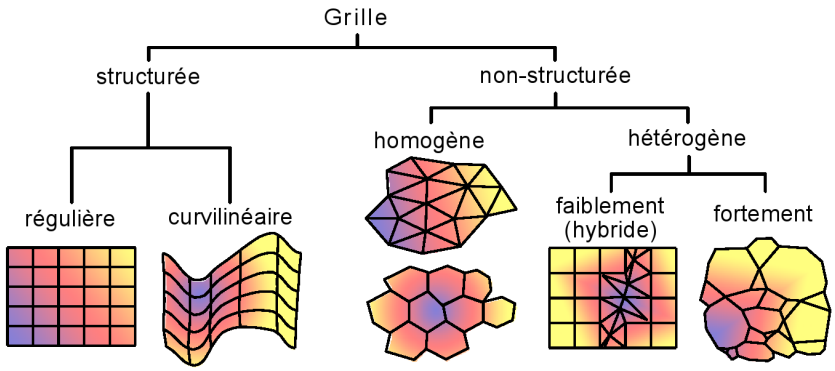


FIG. *2.8: Taxonomie des types de grilles d'après Caumon [15]*

tétraèdres.

Dans ce chapitre nous allons travailler exclusivement sur des grilles structurées régulières, par contre nous verrons dans l'annexe D des applications sur des grilles structurées curvilinéaires et sur des grilles non structurées hétérogènes. Les perspectives que nous avons évoqués dans le chapitre 1 sur l'analyse du réseau de failles sont basées en partie sur l'alogrithme des *marching tetrahedra*.

Les trois prochaines sections sont consacrées à l'étude de la littérature en vue d'adapter l'algorithme des *marching cubes* à notre contexte d'utilisation dans les géosciences.

2.4 Adaptation des *marching cubes* aux contraintes du métier

Dans le domaines des géosciences, les surfaces extraites en utilisant l'algorithme des *marching cubes* servent à de nombreuses applications nécessitant le calcul des volumes, l'étude des connectivités, etc. Pour cela il est nécessaire que ces surfaces soient des 2-variétés et qu'elles soient le plus justes possibles. Ces contraintes n'étant pas respectées initialement par la méthode classique des *marching cubes* nous étudions les solutions disponibles dans la littérature pour remédier à ces problèmes.

2.4.1 Génération d'une 2-variété

Dans [58], Li et Agathoklis montrent que la présence de voxels égaux à l'isovaleur fait apparaître des points dont les voisinages ne sont par homéomorphes à des disques : les surfaces construites ne sont pas des 2-variétés.

Une surface triangulée est une variété si toutes ses arêtes sont incidentes à deux facettes exactement, et si tout sommet possède un voisinage homéomorphe à un disque. Une surface triangulée est dite variété avec bords s'il s'agit d'une variété partout sauf pour un ou plusieurs

cycles disjoints d'arêtes incidentes à une seule face chacune. Ces arêtes sont appelées arêtes de bord, et leurs points extrémaux ont un voisinage homéomorphe à un demi-disque. La figure 2.9 illustre des configurations ne définissant pas des variétés.

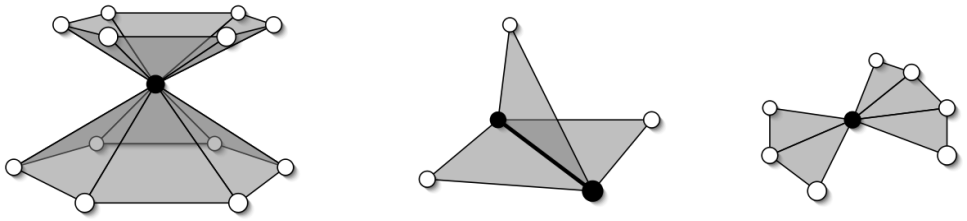


FIG. *2.9: Configurations générant des surfaces non admissibles comme des variétés : à gauche le voisinage du point n'est pas homéomorphe à un disque, au milieu une arête est commune à plus de deux facettes et à droite le voisinage du point n'est pas homéomorphe à un demi-disque.*

De plus, les points générés par l'algorithme appartenant aux arêtes adjacentes à ce sommet seront confondus avec celui-ci générant alors des triangles dégénérés. La présence de ces triangles dégénérés a pour conséquences de perturber l'éclairage à cause de normales mal orientées comme l'illustre la figure 2.10, et de déstabiliser des calculs s'appuyant dessus.

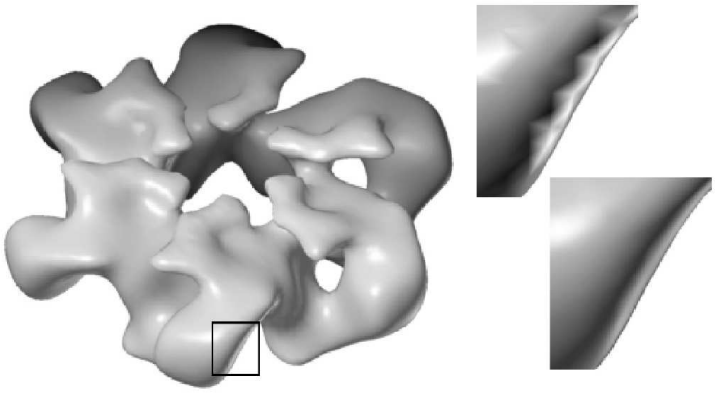


FIG. *2.10: Exemple d'artefacts de rendu dûs aux triangles dégénérés. En haut à droite un agrandissement sur ces artefacts. En bas à droite, un agrandissement sur la même zone montrant le résultat attendu.*

Une solution pour résoudre ce problème est d'éviter la présence de ces voxels égaux à l'isovaleur, soit en choisissant une isovaleur non présente dans le volume [85], ce qui implique de connaître toutes les valeurs présentes, soit en détectant ces voxels et en introduisant du bruit afin de les éliminer [59].
Cette dernière technique est illustrée dans la figure 2.11. Afin de faciliter la compréhension, nous avons choisi de traiter le cas en deux dimensions. Dans cette exemple, nous extrayons l'isoligne passant par 0, or un pixel présent dans la grille est égale à l'isovaleur. Dans ce cas, le résultat

n'est pas correct : les lignes se rejoignent au milieux de la zone grise (figure 2.11 (a)) générant un point adjacent à quatre lignes qui est topologiquement incorrect. L'ajout d'un bruit dans le cas où le pixel est égale à l'isovaleur permet de résoudre ce problème générant ainsi des lignes topologiquement correctes (figure 2.11 (b)). Pour notre application, nous avons choisi d'utiliser cette technique car elle ne nécessite pas de connaître toutes les valeurs présentes dans le cube. L'erreur introduite dans la surface est fonction du bruit et est donc minime pour un bruit très faible.

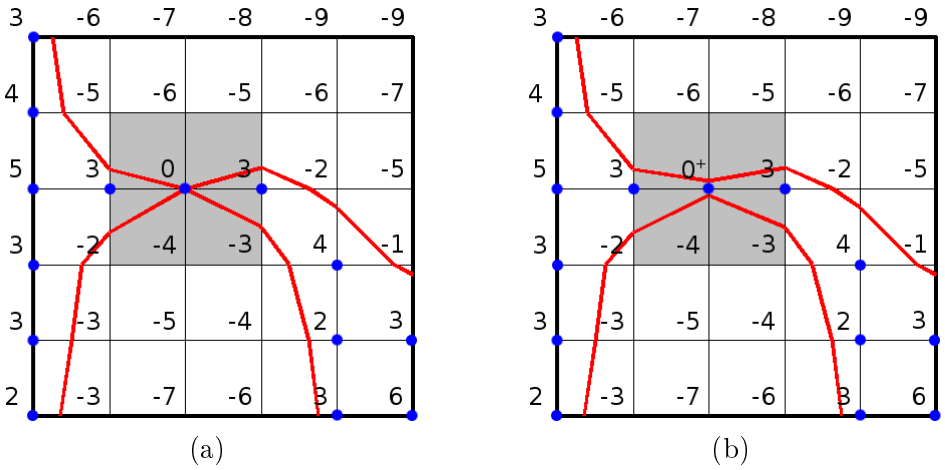


FIG. *2.11: L'ajout de bruit dans la propriété extraite permet de résoudre des problèmes topologiques. (a) Un pixel est égale à l'isovaleur recherchée. (b) L'ajout du bruit résout le problème.*

2.4.2 Gestion des ambiguïtés

L'utilisation des *marching cubes* nécessite de traiter un autre problème lié à l'utilisation des tables pré-calculées. En effet, la table d'origine contient des ambiguïtés : une même combinaison de sommets marqués peut entraîner plusieurs configuration de triangles. Les isosurfaces générées sont alors considérées par la littérature ([17, 27, 71, 85]) comme n'étant ni justes ni consistantes.

La **justesse** mesure l'écart entre la surface extraite et le phénomène discrétisé dans le volume qui est associé à cette surface. Les problèmes de justesse peuvent être dûs à la discrétisation.

Une isosurface est considérée comme topologiquement **consistante** si elle n'a pas de trous excepté aux bords du volume. L'algorithme peut produire dans certains cas des surfaces non consistantes. Elles peuvent aussi être consistantes mais non justes.

Ces problèmes de consistance et de justesse étant liés aux ambiguïtés, de nombreuses recherches ont traité le sujet de la levée des ambiguïtés.

Définition des ambiguïtés

Certaines erreurs topologiques sont dûes à des ambiguïtés dites de “faces”. Ces ambiguïtés arrivent quand les deux sommets opposés sur une même face d'une cellule sont marqués comme étant supérieurs à l'isovaleur contrairement aux deux autres sommets de cette face.
Cette ambiguïté est illustrée en deux dimensions dans la figure 2.12. La cellule en gris met en évidence que deux configurations possibles peuvent apparaître avec la même configuration de sommets marqués. Dans le premier cas les isolignes segmentent l'image en deux composantes, tandis que dans le deuxième cas les deux composantes communiquent.

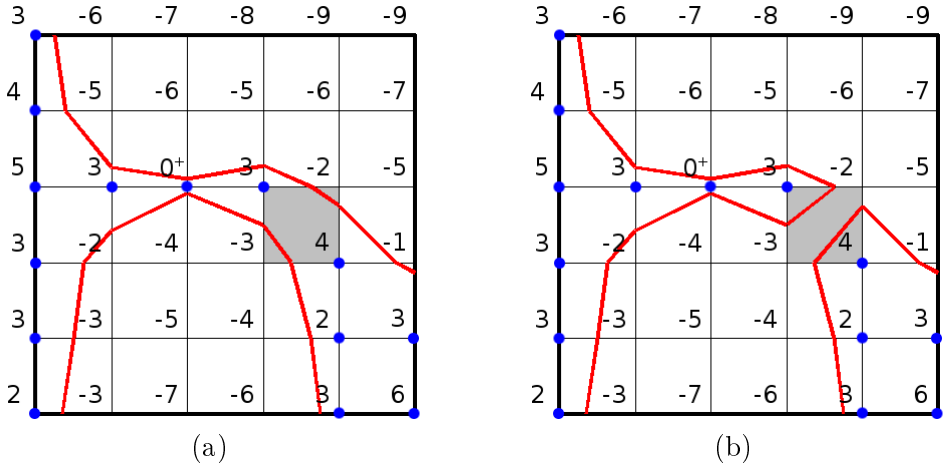


FIG. *2.12: Une même configuration de valeurs aux pixels peut engendrer deux configurations d'isolignes possibles : les arêtes crées sur les faces ambigües ne coincident pas, créant un trou*

En deux dimensions cette ambiguïté ne dégénère pas le résulat mais en trois dimensions, si une face ambigüe est partagée par deux cellules et qu'un choix différent est fait lors de l'extraction de ces cellules, un trou est crée dans l'isosurface la rendant non consistante (voir figure 2.13).

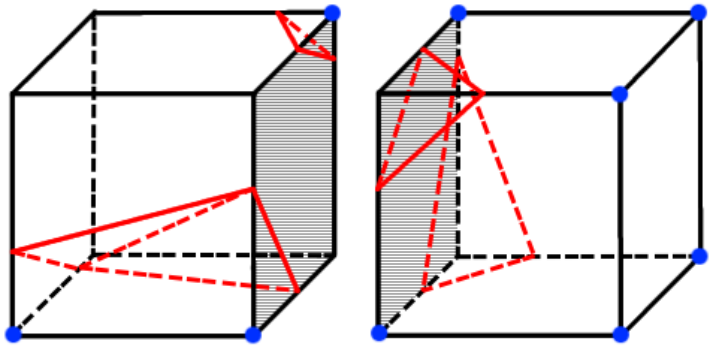


FIG. *2.13: Deux cellules adjacentes partagent une face ayant une configuration ambigüe*

Natarajan [71] met en évidence un comportement similaire à l'ambiguïté de face, observée en deux dimensions dans la figure 2.12, mais à l'intérieur de la cellule : l'isosurface peut être séparée en deux composantes ou un tunnel peut créer un connection entre ces deux composantes. Ces amibuïtés sont dites "internes". Celles-ci ne créent pas d'erreurs topologiques dans l'isosurface mais peuvent générer des isosurfaces non justes. Nous illustrons un cas d'ambiguïté interne dans la figure 2.14.

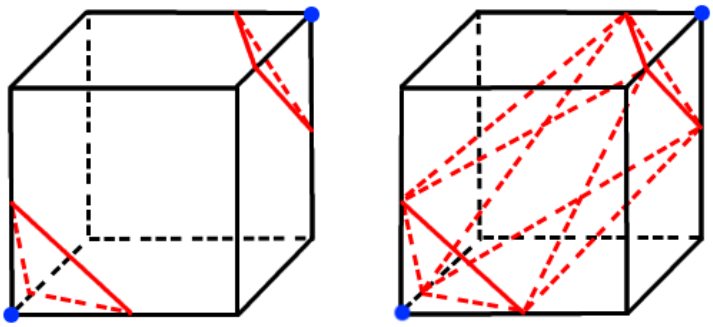


FIG. *2.14: La même configuration engendre deux surfaces de topologies différentes.*

Traitement des ambiguïtés

Plusieurs taxonomies ont été proposées pour classer les méthodes de traitement des ambiguïtés. Dans [85], Van Gelder et Wilhelms font la différence entre les méthodes utilisant les booléens portés par les sommets de la cellule (si la valeur du sommet est supérieure ou non à l'isovaleur) et les méthodes utilisant les valeurs des sommets. Une notion de *méthodes étendues* est ajoutée afin de préciser si celles-ci s'intéressent aux cellules voisines. Dans leur état de l'art Newman et Yi [72] font une classification exhaustive des papiers traitant le sujet en utilisant les mêmes critères.

Dans l'article original [64], la position du point d'isovaleur α sur une arête active est calculée par interpolation linéaire. De la même façon les positions des points d'isovaleur α peuvent être sur une face et sur une cellule respectivement par interpolation bilinéaire et trilinéaire. Chernyaev [17] propose d'étudier le centre de l'hyperbole et de l'hyperboloïde définies par les points d'isovaleur α respectivement sur les faces et les cellules ambigües. Si la valeur du centre est supérieur ou égale à α alors l'isosurface contient ce centre (voir figure 2.15).

Dans notre cas nous avons choisi d'utiliser l'extension de la table proposée par Lewiner et al dans [57] qui présente une implémentation complète de la méthode des *marching cubes 33* décrite par Chernyaev dans [17] et qui génère des surfaces topologiquement correctes. Cette méthode présente aussi l'avantage d'être très performante en temps calcul, ce qui est essentiel quand les jeux de données manipulés sont de très grandes tailles.

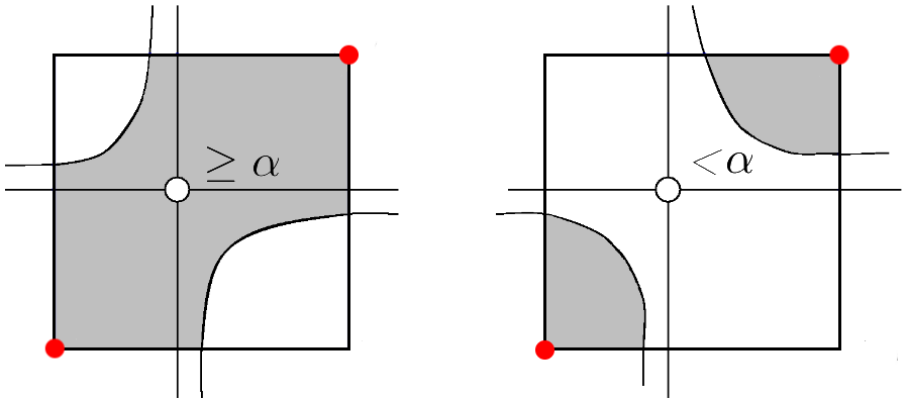


FIG. *2.15: Traitement des ambiguïtés (figure issue de [17])*

Les isosurfaces générées à partir de ces gros volumes de données vont contenir énormément de triangles, c'est pourquoi nous nous intéressons dans la suite à la simplification de ces surfaces.

2.5 Simplification des isosurfaces

Un problème récurent avec l'algorithme des *marching cubes* est le nombre important de triangles qui composent la surface générée. En effet, chaque cellule intersectée par l'isosurface va générer, d'après la table originale, entre 1 et 4 triangles. Les tables proposées afin de lever les ambiguïtés peuvent générer encore plus de triangles. Afin d'exploiter les surfaces générées, par exemple pour les afficher, il est donc nécessaire de les simplifier.

La simplification de surface (de manière générale) est un domaine de recherche très actif, nous allons donc nous concentrer sur les approches adaptées à l'extraction d'isosurfaces. Nous pouvons classifier ces techniques en trois catégories : les techniques décimant les cubes, les approches utilisant un algorithme de simplification en post-traitement et finalement celles mélangeant l'extraction et la simplification que nous qualifierons de mixtes.

2.5.1 Approches par décimation adaptative des cubes

Le principe de ces méthodes est de réduire la résolution du volume contenant l'isosurface afin de diminuer le nombre de cellules le composant et ainsi le nombre de triangles décrivant l'isosurface. Une approche triviale consiste à décimer l'ensemble du volume en utilisant un filtre passe-bas. La figure 2.16 illustre l'extraction d'une méme isosurface sur un volume à différentes résolutions. Les résolutions sont croissantes de gauche à droite. Nous remarquons que la décimation du volume permet de réduire considérablement le nombre triangles. En revanche, ce procédé peut modifier la topologie de l'objet comme l'illustre la surface la plus à gauche.

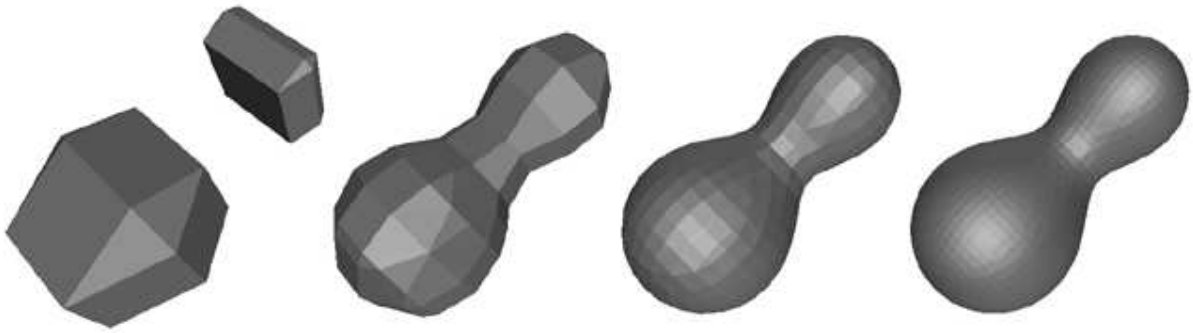
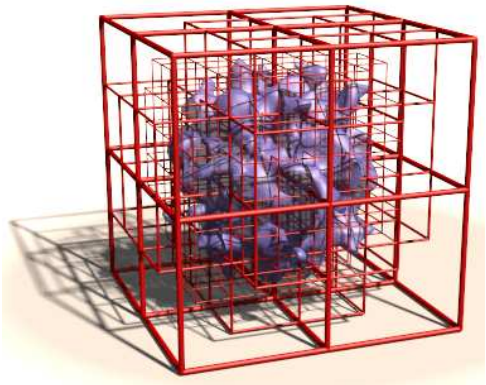


FIG. *2.16: Extraction d'une isosurface sur un volume avec des résolutions différentes*

Un autre problème lié à cette décimation que l'on peut qualifier d'*uniforme* est qu'elle ne s'adapte pas à la morphologie de la surface : les zones complexes sont décimées de la même façon que les zones planes. C'est pourquoi des travaux s'intéressent à des décimations dites *adaptatives* notamment à l'aide d'*octrees*. Les bénéfices des octrees pour une reconstruction plus rapide de l'isosurface ont été mis en évidence par Jane Wilhelms et Allen Van Gelder dans [87]. Chaque noeud contient les valeurs minimum et maximum apparaissant dans ses sous-branches, si l'isovaleur cherchée n'est pas comprise dans cet interval alors l'isosurface n'intersecte ni ce noeud ni ses sous-branches. Cette technique accélére ainsi considérablement la recherche des cellules traversées par la surface.



Cependant cette méthode ne décime pas le volume et donc ne réduit pas le nombre de triangles générés. C'est pourquoi, des évolutions de cet algorithme ont été proposées dans [74, 82] en décomposant le volume en une hiérarchie de sous-volumes à différentes résolutions qui permet une reconstruction adaptative. L'octree est parcouru en partant du haut vers le bas, c'est-à-dire du volume le plus décimé vers les sous-volumes aux résolutions les plus élevées. Quand un sous-volume est considéré comme suffisamment raffiné (au sens d'un critère défini par l'approche), on extrait une sous-partie de la surface sur ce volume en utilisant l'algorithme des *marching cubes*.

Malgrès les avantages apparents de cette représentation hiérarchique, le fait de décimer le volume peut déplacer la surface dans les sous-volumes, voir même en supprimer une partie. Il faut aussi gérer le fait que la surface est extraite sur des volumes ayant des résolutions différentes. La surface générée peut donc contenir des trous (voir figure 2.17).

Les méthodes par décimation adaptative permettent de traiter de gros volumes, mais les données manipulées en géosciences génèrent des surfaces complexes réparties en nombreuses com-

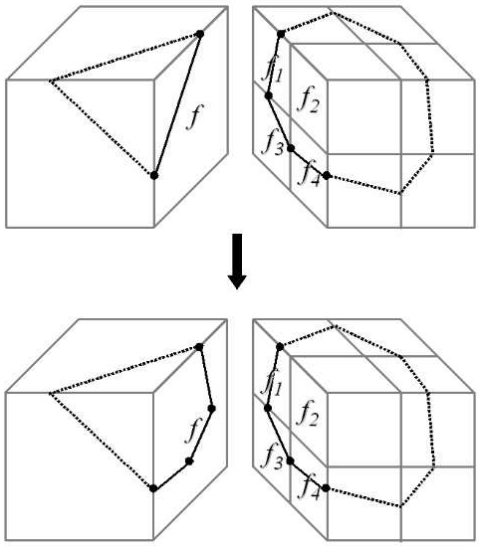


FIG. *2.17: Exemple de reconstruction d'une surface simplifiée à l'aide d'un octree. En haut, les différents niveaux de résolution peuvent introduire des trous dans la surface. En bas, un traitement est fait sur les basses résolutions pour prévenir l'apparition de ces trous. Figure issue de [50]*

posantes de tailles variables et dont la répartition est non uniforme. Dans ces conditions, un découpage par octree ne parait pas alors être adapté. De plus, la triangulation finale obtenue grace aux *marching cubes* reste de qualité médiocre.

2.5.2 Simplification en post-traitement

Une stratégie intuive consiste à extraire la surface globale et à appliquer un algorithme de simplification sur cette surface dans un deuxième temps. Luebke [65] propose de classer les techniques de simplification selon les critères suivants :

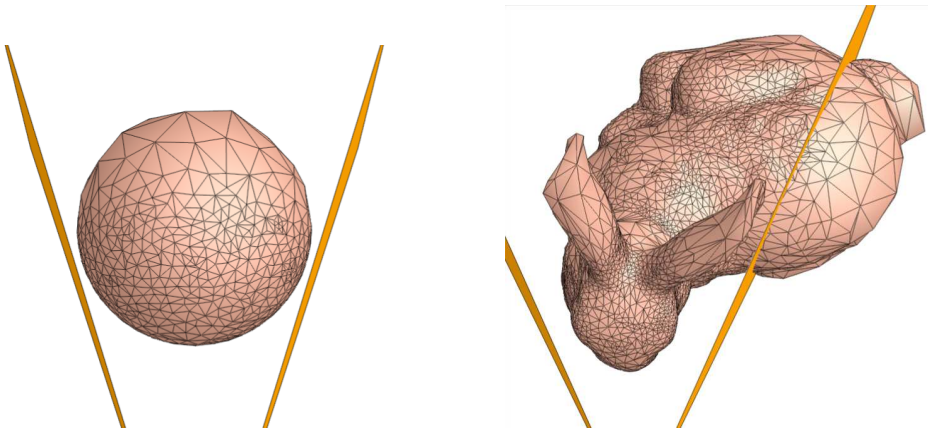
- la dépendance au point de vue sur l'objet lors du rendu,
- la préservation de la topologie de la surface,
- le mécanisme utilisé pour réduire le nombre de triangle,
- la métrique d'erreur contrôlant le processus.

Dans cette section, nous détaillons chacun de ces critères afin de mettre en évidence les caractéristiques essentielles à un algorithme de simplification adapté à nos données.

Dépendance au point de vue

Certaines méthodes de décimation proposent de tenir compte de la position de la caméra lors du rendu de la surface pour adapter le raffinage de celle-ci : les parties visibles de l'objet contiendront plus détails que les parties non visibles. La figure 2.18 nous montre deux exemples de surfaces raffinées en fonction du point de vue, l'angle de vue de la caméra est matérialisé par les plans oranges. Dans nos applications, les surfaces que nous allons manipuler serviront

entre autre comme support pour des calculs de volume, d'intersections, etc. Dans ces différentes utilisations, nous n'avons pas besoin d'une technique tenant compte du point de vue.



(a) (b)

FIG. *2.18: Raffinement de la surface lié au point de vue. Figure issue de [41].*

Préservation de la topologie

La littérature considère deux niveaux de topologie : la topologie locale qui concerne la géométrie du voisinage d'un sommet ou d'une facette et la topologie globale qui concerne la géométrie de la surface dans son intégralité. Un algorithme préserve la topologie locale si il préserve la structure locale, par exemple si il ne crée pas ou ne bouche pas de trous dans la surface. La topologie globale est préservée par un algorithme si il préserve aussi la topologie locale et qu'il ne crée pas d'intersection dans la surface avec elle même.

Préserver la topologie globale nécessite un surcoût non négligeable en temps de calcul et n'est pas indispensable pour nos applications. En effet, les modifications éventuelles de la topologie globale ne sont pas pénalisantes dans les traitements effectués sur les surfaces extraites. Par contre il est important pour nous de choisir un algorithme qui préserve la topologie locale.

Mécanismes de simplification

Les mécanisme de simplification sont généralement classifié suivant quatre tendances : les techniques par échantillonnage, par subdivision adaptative, par décimation et par fusion de sommets.

Échantillonnage Le principe de ces méthodes est d'échantillonner la surface soit en nuage de points soit en volume mesurant la distance à la surface. L'algorithme va alors générer une surface simplifiée respectant au mieux ces échantillons.

Par exemple, He et al [40] propose de transformer la surface en volume. Ce volume est ensuite décimé à l'aide d'un filtre passe bas. Enfin la surface est reconstruite en utilisant l'algorithme

des *marching cubes*. Cela revient à utiliser les méthodes de décimation adapative du volume présentées dans la section 2.5.1. Dans notre contexte d'utilisation, ces techniques ne nous intéressent pas car nous possédons déjà un volume échantillonné (attribut sismique) et nous avons déjà éliminé ces techniques de décimation de volumes.

Subdivision adaptative L'idée principale des méthodes par *subdivision adaptative* est de créer un modèle approximant grossièrement la surface et de le raffiner par subdivisions successives.

Pour illustrer ces méthodes nous pouvons citer la méthode proposée par Garland en 1995 dans [32] pour simplifier une surface modélisée par une grille d'élévation. De manière générale, ces méthodes sont souvent utilisées sur des données modélisant des terrains.

Décimation Les méthodes par *décimation* enlèvent, de manière itérative, des points ou des triangles de la surfaces. Les trous formés dans le modèle sont alors retriangulés (voir figure 2.19).

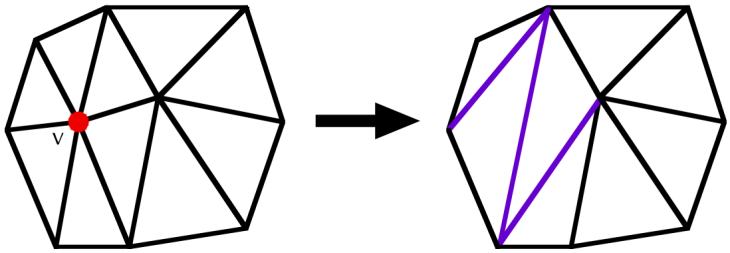


FIG. *2.19: Le sommet v (en rouge sur la figure de gauche) est supprimé, ainsi que ses triangles adjacents. Le trou laissé est retriangulé (en bleu sur la figure de droite).*

L'un des premiers algorithmes de simplification par décimation fut proposé par Schroeder et al [80] en 1992. L'idée de la méthode consiste à itérer sur les points de la surface et à retirer ceux appartenant au plan défini par les points voisins. Un avantage de ce type de méthode est que les sommets de la surface simplifiée sont un sous-ensemble des sommets de la surface originale. C'est une propriété intéressante dans le cas où une information supplémentaire est portée par les sommets, par exemple des coordonnées de texture. Il est cependant difficile de contrôler la qualité des triangles générés par ce type de méthode.

Fusion de sommets L'opération principale de ces algorithmes consiste à fusionner des sommets (deux ou plus). Les facettes reliant ces sommets entre eux deviennent alors dégénérées et peuvent être supprimées réduisant ainsi le nombre de facettes dans la surface.

Le mécanisme de *fusion de sommets* le plus général est le *vertex clustering* (ou amas de sommet). Cette méthode a été originalement introduite par Rossignal et Borrel [78] en 1993. Récemment Linstrom [60] a proposé un algorithme de simplification, nommé *OoCs* adaptée aux surfaces de trés grandes tailles telles que celles générées par l'algorithme des *marching cubes*. La

simplification de la surface se fait par *vertex clustering*. Comme dans l'algorithme original de Rosignal et Borrel, l'idée principale de la méthode est de faire un unique passage sur les triangles et de calculer le *cluster* associé à leurs sommets. L'information de ces clusters est portée par une grille régulière qui est gérée de manière "out-of-core". À chaque *cluster* est associée une fonction d'erreur qui évolue à chaque ajout d'un triangle dans le cluster.

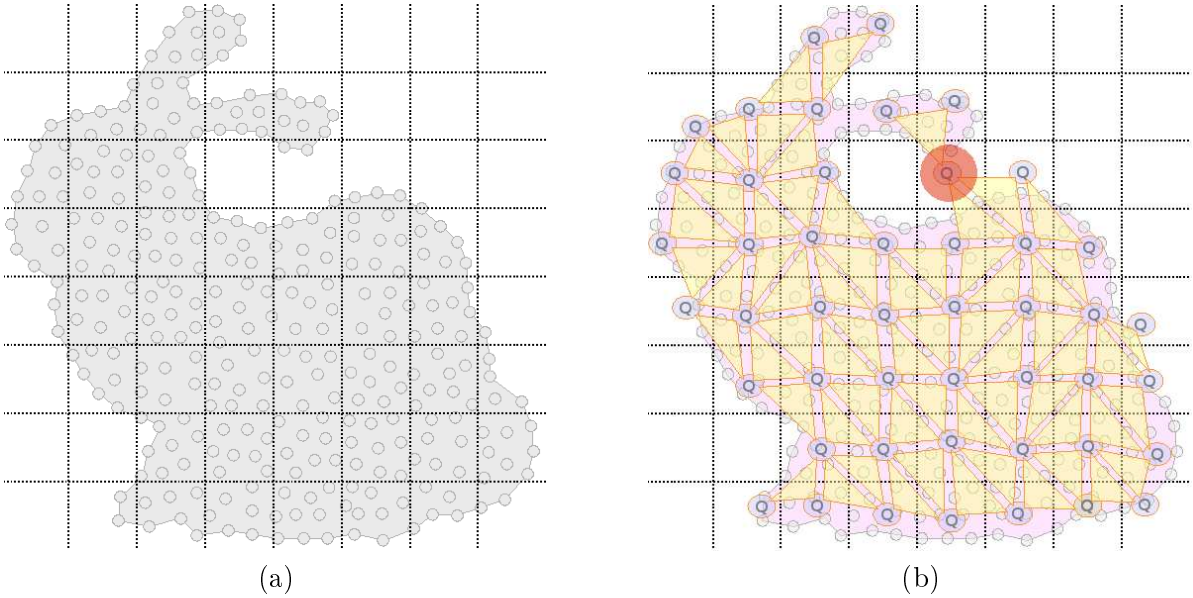


FIG. *2.20: Algorithme de simplification OoCs (a) La grille régulière servant de support à la simplification est supperposée aux sommets originaux de la surface (b) On ajoute le maillage simplifié reliant les sommets portés par la grille. Une erreur topologique est mise en évidence par le disque rouge.*

La position du sommet associé à chaque *cluster* dans la surface simplifiée minimise la fonction d'erreur du cluster. Cet algorithme est très efficace, complexité en $log(n)$, et donne de bons résultats en terme de rendu. Le problème est qu'il ne peut pas garantir que la surface simplifiée soit une 2-variété comme l'illustre la figure 2.20.

Un autre mécanisme couramment utilisé est le *vertex contraction* qui consiste à fusionner deux sommets voisins de la surface. Ce mécanisme est notamment utilisé dans l'algorithme proposé par Garland [31] est illustré dans la figure 2.21. Dans sa méthode, Garland a choisi ce mécanisme car il permet de simplifier la topologie de la surface, ce qui est un avantage dans certains cas, par exemple quand on veut faire du rendu multi-résolution, mais qui est discriminant pour notre application.

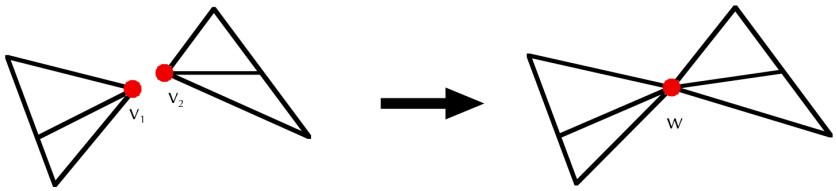


FIG. *2.21: Vertex contraction*

Enfin la méthode la plus utilisée reste le *edge collapse*, introduite par Hoppe en 1993 dans [45], qui fusionne deux sommets en un seul à la condition qu'ils aient une arête commune (voir figure 2.22).

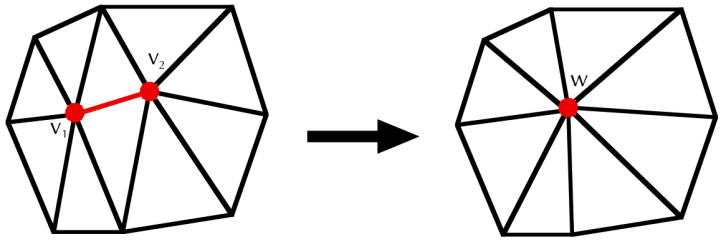


FIG. *2.22: Edge collapse*

Cette technique nous intéresse particulièrement car les travaux de Dey et al [23] permettent de garantir une préservation de la topologie locale lors du processus de simplification.

Les algorithmes de simplification de surfaces utilisant le mécanisme de *edge collapse* sont, pour la plupart, structurés de la même façon : on itère sur les arêtes de surface et on estime un coût pour la contraction de chacune d'entre elles. Ce coût permet de déterminer l'ordre et la faisabilité de chaque contraction. Les arêtes sont ordonnées dans une file de priorité en fonction de ce coût. A chaque contraction les arêtes adjacentes au nouveau point sont évaluées mettant à jour la file de priorité. Les critères d'arrêts de l'algorithme peuvent être un contrat sur le nombre de triangles composant la surface ou une erreur maximale pouvant être introduite dans la surface en comparaison à la surface originale. L'algorithme doit alors être contrôlé grâce à une *métrique d'erreur* permettant de calculer le coût d'une contraction.

Métrique d'erreur

La métrique permet de mesurer l'erreur introduite dans la surface simplifiée par rapport à la surface originale. De manière générale la métrique est utilisée pour ordonner les opérations de simplification.

La métrique la plus utilisée pour contrôler un processus de simplification basé sur la fusion de points est celle proposée par Garland et Heckbert dans [33]. Elle borne la distance au carré entre

un point et un ensemble de plans définis par les facettes (triangulaires) de la surface.
Si on considère une facette f définissant le plan P_f avec n_f sa normale et p_f un point lui appartenant, l'équation du plan s'écrit :

$$(x - p_f) \cdot n_f = 0$$

La distance au carré d'un point quelconque x à ce plan s'écrit :

$$\begin{aligned} d^2(x, P_f) &= ((x - p_f) \cdot n_f)^2 \\ &= x^T n_f n_f^T x - 2(n_f n_f^T)^T x + x^T n_f n_f^T x \end{aligned}$$

Si on considère un ensemble de facettes $\{f_1, f_2, ..., f_n\}$, l'erreur pour un point x est définie par :

$$erreur(x) = \sum_{i=1...n} w_i d^2(x, P_{f_i})$$

avec w_i l'aire de la facette f_i. On cherche le point qui minimise cette erreur. La figure 2.23 illustre le principe d'une contration d'arêtre contrôlée par cette métrique en deux dimensions.

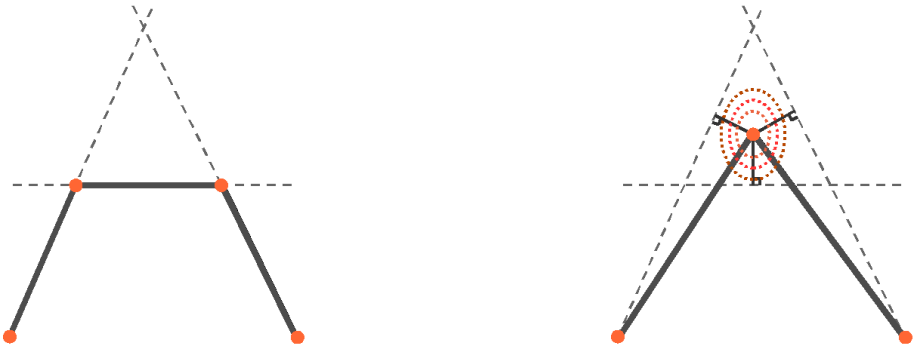


FIG. *2.23: Principe de la métrique en deux dimensions : les sommets du polygones apparaissent en orange, les arêtes en noires et les droites passant par les arêtes sont en pointillés. (a) Le polygone original. (b) Le polygone après une contraction d'arête. Le nouveau sommet minimise la métrique (en pointillés rouges), c'est-à-dire l'écart aux droites.*

Plusieurs raffinements de cette métrique, par exemple [34] et [42], ont été proposés pour prendre en compte d'autre informations : normales, couleurs, ...

La littérature propose des solutions pour simplifier les surfaces de grande taille. Dans notre cas, ces approches nécessitent d'extraire la surface dans sa totalité et de la simplifier dans une seconde étape. Afin d'éviter de devoir stocker cette surface, souvent énorme, en mémoire ou même sur disque, nous allons nous intéresser aux méthodes qui font l'extraction et la simplification en même temps.

2.5.3 Approches mixtes

Parmi ces méthodes, nous avons isolé trois catégories : la méthode de maillage volumique, les méthodes basées sur les formats dit "flux" et le *tandem algorithm*.

Maillage volumique

Nous pouvons citer la méthode de maillage volumique proposé par Oudot et al en 2005 dans [76]. La méthode proposée combine un mailleur de surface avec un mailleur de volume, tous deux basés sur le raffinememt de Delaunay, pour obtenir un algorithme glouton qui échantillonne l'intérieur et la frontière du domaine simultanément. La figure 2.24 illustre le résultat de cette technique. L'algorithme ne nécessite de connaître la frontière du domaine qu'à travers un oracle capable de décider si un point de requête est à l'intérieur ou à l'extérieur du domaine et si un segment de droite intersecte ou non la frontière. Il peut donc être facilement appliqué à de nombreux domaines, en particulier les volumes.

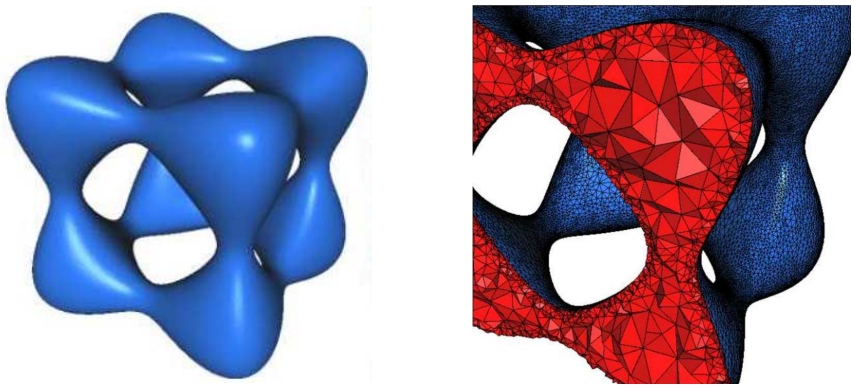


FIG. *2.24: Surface reconstruite avec la méthode de Oudot et al [76]. En bleu la frontière du domaine et en rouge l'intérieur respectent le critère de Delaunay assurant une bonne qualité de maillage.*

Cette méthode est donc une alternative à l'algorithme des *marching cubes*, mais n'est pas adaptée à nos besoins. En effet, lors de la reconstruction l'algorithme accède de manière aléatoire dans le volume, ce qui est très pénalisant dans le cas de très gros volumes. De plus, l'algorithme ne traite pas les surfaces ayant des arêtes vives, surfaces que nous devons traiter dans nos jeux de données.

Formats dits "flux"

Le format de représentation le plus commun pour les surfaces triangulées est le format dit "indexé" où chaque point est décrit par sa position et, après la description de tous les points, les triangles sont décrits par trois références aux points. C'est un format très facile à manipuler et la majorité des formats de stockage sur disque (OFF, PLY, ...) suivent ce modèle. Il est cependant limité car il oblige la lecture de tous les points afin de décoder la géométrie des triangles. Dans

l'optique d'associer un processus d'extraction d'isosurface à un processus de simplification de celle-ci, ce format est donc inadapté car il faudrait extraire la surface entière afin d'associer ces deux processus.

Une autre représentation est la "soupe de triangles", où chaque triangle est décrit par trois positions. Les triangles sont alors indépendants les uns des autres. Cette représentation est alors plus adaptée à la transmission de données d'un processus à un autre. Nous le qualifierons de format de type "flux". Dans ces condition on peut associer un algorithme de simplification de surfaces, prenant en entrée ce flux, à une version de l'algorithme des *marching cubes* le générant.

Parmi les algorithmes de simplification prenant en entrée une soupe de triangle on peut citer celui proposé par Wu et Kobbelt en 2003 [88] qui permet de simplifier des surfaces de grande taille. Celles-ci doivent cependant être triées de manière à recevoir des triangles adjacents. Cette condition est remplie dans le cas où cet algorithme est associé à une méthode d'extraction d'isosurface telle que les *marching cubes* qui procède couche par couche. Dans l'optique de traiter la surface comme un flux, les auteurs distinguent trois parties : la partie qui n'est pas encore lue (sur le disque ou en cours d'extraction), la partie en mémoire vive qui est en cours de simplification et finalement la partie déjà traitée qui peut être déchargée sur le disque. La figure 2.25 illustre ces trois parties.

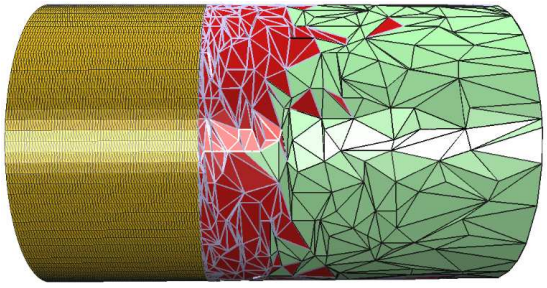


FIG. *2.25: Simplification d'une surface avec la méthode de Wu et Kobbelt. En jaune la surface non simplifiée. En rouge la surface dans le buffer de simplification. En vert la surface simplifiée pouvant être déchargée de la mémoire. Figure issue de [88]*

Il faut noter que la connectivité des triangles est retrouvée grâce à une table de hachage qui associe les indexs des sommets de la surface par leurs positions. Il est tout de même regrettable que cette information connue lors de l'extraction doive être recalculée pour la simplification. Afin de résoudre ce problème, Isemburg et Lindstrom [48] ont décrit un nouveau format pour le stockage des surfaces triangulées. Un triangle est défini par trois références à des points. Ces points ne sont définis qu'au moment opportun : quand un triangle référence un point pour la première fois. De même, si un point n'est plus référencé dans la suite, une balise va marquer la fin de sa nécessité en mémoire. La figure 2.26 résume les différents formats de description des surfaces triangulées.

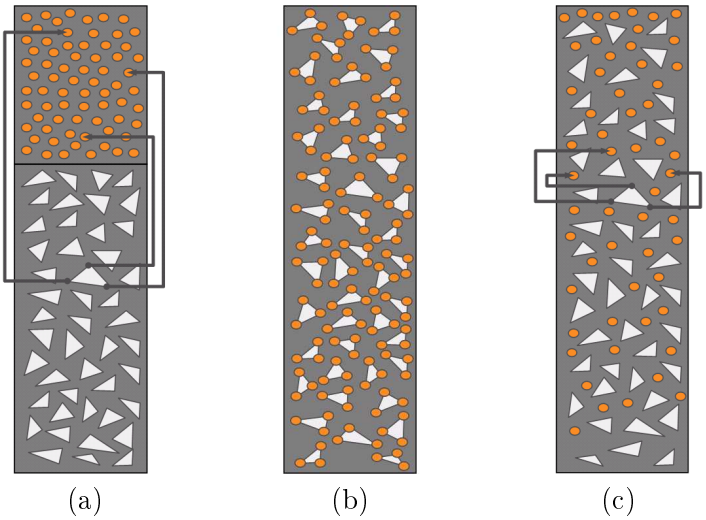


FIG. *2.26: Formats décrivant les surfaces triangulées : (a) le format indéxé, (b) la soupe de triangles et (c) le format "flux".*

De cette manière, l'algorithme de Wu et de Kobbelt pourra être utilisé de manière plus efficace : la relation entre les triangles étant conservée, l'utilisation d'une table de hachage n'est plus nécessaire. De plus, les bords de la surface sont désormais connus et leur simplification pourra commencer plus tôt. Ce format permet de mieux intégrer le processus de simplication au processus d'extraction. Poussant plus loin ce concept d'intégration, nous allons nous intéresser maintenant au *tandem algorithm* qui mélange ces deux processus dans le but de mieux les contrôler.

Le *tandem algorithm*

L'idée principale du *tandem algorithm* [6] est d'alterner une étape d'extraction d'une couche (voir la figure 2.27) avec une étape de simplification appliquée au reste de la surface. Derrière cette idée simple Attali et al ont ajouté des raffinements afin de prévenir l'apparition d'artefacts dans la surface dûs à un manque d'information sur la géométrie durant l'étape de simplification. La gestion de ces artefacts par une le contrôle des deux processus font de cet algorithme une extension puissante et élégante de l'algorithme des *marching cubes*. La solution utilisée sur les données sismiques est bâtie sur cette méthode.

Le *tandem algorithm* peut s'écrire comme une simple boucle itérant sur les couches du volume en appliquant ces deux opérations :

- *extraction* qui ajoute les triangles extraits par l'algorithme des *marching cubes* appliqué sur une couche à la triangulation courante.
- *simplification* qui simplifie la triangulation courante. Les triangles extraits dans la dernière couche sont préservés afin de permettre l'ajout des triangles issues de la prochaine couche.

L'apport du *tandem algorithm* est donc dans cette étape de simplification qui nécessite à la fois :

- de réduire le nombre de triangles tout en préservant une bonne qualité de maillage,

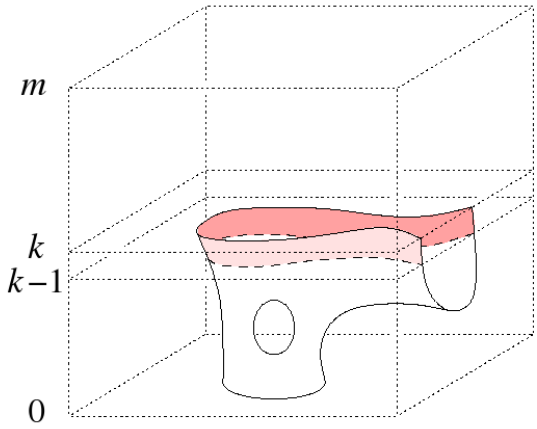


FIG. *2.27: Principe d'extraction en couche : la lecture du volume est faite coupe par coupe. L'extraction de la surface est faite sur la couche courante (en rouge). Figure issue de [6]*

– d'assurer la continuité du maillage avec les couches à extraire.

Nous détaillons ci-dessous ces deux aspects.

Simplification L'étape de simplification consiste à appliquer l'algorithme proposé par Garland et Heckbert [33] utilisant des contractions d'arêtes successives afin de réduire le nombre de triangles. Chaque arête a un coût de contraction qui lui est associé. Ce coût est une mesure de l'erreur qui sera introduite dans la triangulation si l'arête est contractée. Chaque arête candidate pour une contraction est donc gardée dans une file de priorité $\mathcal{Q}$ triée en fonction de leur coût. Afin d'évaluer ce coût, Attali et al ont proposé une nouvelle métrique qui est la somme pondérée entre une *mesure d'écart* (écart à la surface) et une *mesure d'isotropie* (qualité du maillage).

La *mesure d'écart* est en tout point similaire à celle proposée dans le papier original de Garland et Heckbert. Il mesure la distance introduite par la contraction entre les nouveaux points et la surface originale.

Lors d'une contraction d'arête $ab \mapsto c$, on définit le patch U_c comme étant l'ensemble des triangles de la surface d'origine contenant au moins une des arêtes dont les contractions successives sont à l'origine des sommets a et b. On définit la *mesure d'écart* $h_c(x)$ d'un point x par

$$h_c(x) = \frac{1}{W_c} \sum_{t \in U_c} w_t d^2(x, P_t) = \frac{1}{W_c} x^T \mathbf{H}_c x$$

où P_t est le plan défini par le triangle t, w_t l'aire du triangle et $W_c = \sum_{t \in U_c} w_t$ la somme des aires. $\mathbf{H}_c$ est une matrice définie positive.

En pratique, H_c et W_c seront construit de façon récursive par les relations :

$$\begin{aligned} \mathbf{H}_c &= \mathbf{H}_a + \mathbf{H}_b \\ W_c &= W_a + W_b \end{aligned}$$

Ces relations ne sont pas tout à fait exactes, mais les auteurs ont montré qu'elles sont une bonne approximation dans la mesure où ces termes apparaissent sous forme de quotient.

La *mesure d'isotropie* est introduite dans le but d'empêcher l'apparition de triangles longs et fins. On considère S_{ab} l'ensemble des triangles contenant les sommets a, b ou les deux dans la triangulation courante. La *mesure d'isotropie* s'exprime :

$$g_c(x) = \sum_{t \in S_{ab}} w_t(||x - \hat{t}||^2 + avg(t)) = x^T \mathbf{G}_c x$$

avec où $\hat{t}$ est le barycentre du triangle t et $avg(t) = \frac{1}{12}(||p||^2 + ||q||^2 + ||r||^2)$ avec p, q, r les vecteurs de $\hat{t}$ vers les points de t.

Le terme g_c est normalisé par $W = 3area(S_{ab})W_c^{1/2}/E_0$ dans le but d'équilibrer son influence avec h_c dans la fonction de coût globale définie par :

$$\begin{aligned} \varepsilon_\alpha(c) &= \sqrt{(1-\alpha)h_c(c) + \alpha\frac{g_c(c)}{W}} \\ &= \sqrt{c^T[(1-\alpha)\frac{\mathbf{H}_c}{W_c} + \alpha\frac{\mathbf{G}_c}{W}]c} \end{aligned}$$

où $\alpha \in [0, 1]$ est appelé *paramètre d'isotropie*, et représente un compromis entre la *mesure d'écart* et la *mesure d'isotropie*. Une étude de l'impact de α sur la qualité de la surface est disponible dans l'article original, en pratique $\alpha = 0,4$ est un bon compromis.
Lors de la contraction d'une arête, la position du point résultat c est obtenue en minimisant la fonction d'erreur locale ε_α et la valeur d'erreur finale $\varepsilon_\alpha(c)$ est utilisée pour trier la file de priorité $\mathcal{Q}$. En aucun cas, un candidat ne peut être accepté si sa *mesure d'écart*, $\varepsilon_0(c)$, excède un seuil d'erreur E_0 tel que $E_0 \in \mathbb{R}^+$. La fonction *simplification* consiste à vider la file de priorité par contraction d'arêtes successives.

Continuité du maillage L'étape de simplification doit imposer des contraintes fortes pour préserver les arêtes de la dernière couche extraite. Afin de prévenir les artefacts pouvant être introduits par le blocage de ces arêtes, une méthode innovante a été proposée : le *time lag*. L'idée principale est de retarder la contraction des arêtes à proximité du front.

Le *time lag* est basé sur le *rang* d'un point, qui est égale à sa coordonnée le long de l'axe d'extraction (par exemple z si les coupes sont prises le long de l'axe z). Le *rang du front* est le rang maximum qui a été extrait.
Pour tout nouveau point u introduit par l'extraction, on définit :

$$\begin{aligned} height(u) &= rang(u) \\ rad(u) &= 1 \end{aligned}$$

la contraction d'une arête $ab \mapsto c$ implique :

$$\begin{aligned} height(c) &= (height(a) + height(b))/2 \\ rad(c) &= (||a-b|| + rad(a) + rad(b))/2 \\ reach(c) &= height(c) + rad(c) \end{aligned}$$

La contraction d'une arête est retardée tant que la valeur de son *reach* est supérieur ou égale au rang du front avançant. De cette façon les arêtes appartenant à la dernière couche extraite seront préservés. En effet si a et b appartiennent à la dernière couche extraite et si on considère $ab \rightarrow c$ alors $height(c) = rang(front)$ et comme $rad(c) > 0$, alors $reach(c) > rang(front)$ et la contraction de ab sera retardée. De manière similaire si a ou b appartient au front alors le $reach(c)$ sera plus grand que le rang du front. La figure 2.28 illustre le principe du *time lag* en présentant une contraction d'arête autorisée et une retardée.

Les arêtes dont la contraction à été rétardée par le *time lag* sont stockées dans une file de priorité $\mathcal{W}$ ordonnée en fonction du *reach*. La fonction *delay* ajoute toutes les arêtes de la dernière couche extraite k dans $\mathcal{W}$. Une arête passera de $\mathcal{W}$ à $\mathcal{Q}$ si sa valeur de *reach* est inférieure au rang du front. La fonction *activate*, décrite dans l'algorithme 2.1, déplace les arêtes de $\mathcal{W}$ dans $\mathcal{Q}$. La figure 2.29 illustre les effets du *time lag* à proximité du front.

```
tant que reach(top(W)) < k faire
    add top(W) in Q ;
    pop(W) ;
fin
```

ALGORITHME 2.1 : *Fonction activate*

Le *tandem algorithm* peut alors s'écrire comme dans l'algorithme 2.2. Il prend en paramètre E_0, l'erreur maximale pouvant être introduite dans l'écart à la surface d'origine.

```
pour k ← 1 à nombre de coupes - 1 faire
    extract(k) ;
    delay(k) ;
    activate(k) ;
    simplify(E0) ;
fin
activate(∞) ;
simplify(E0) ;
```

ALGORITHME 2.2 : *Tandem Algorithm*

Limitations La méthode proposée par Attali est très élégante et permet de générer une surface de très bonne qualité mais elle reste limitée par la taille du front. En effet, si le front génère trop de triangles, on consomme trop de mémoire et la mise à jour des structures est trop longue. Nous

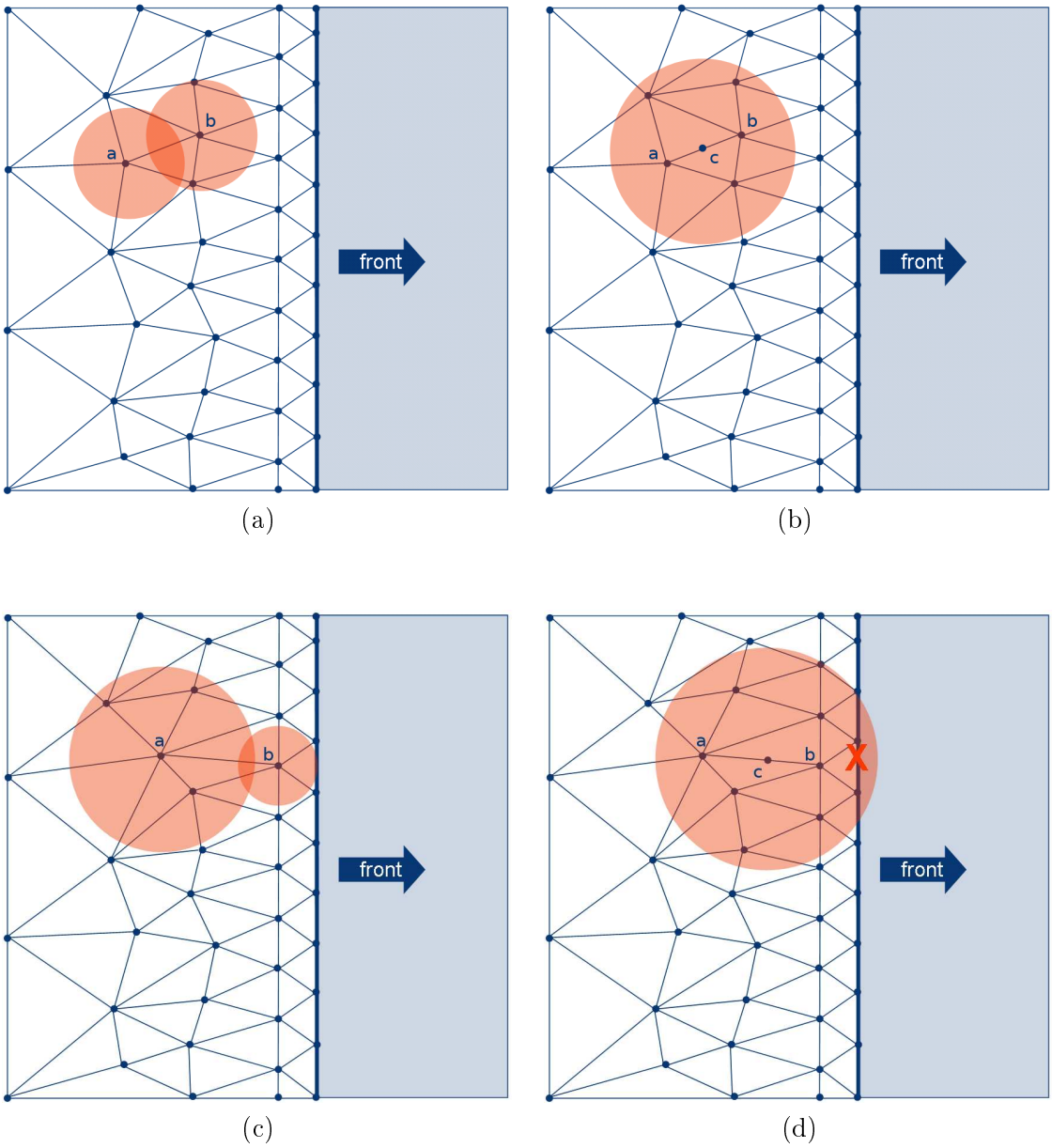


FIG. 2.28: *Schéma illustrant le principe du time lag. Dans les figures, le cercle rouge centré sur un point p est de rayon rad(p). (a) On s'intéresse à la contraction de l'arête dont les points sont les centres des cercles. (b) On estime la position du point généré par la contraction de l'arête, ainsi que le rayon du cercle associé. Dans ce cas la contraction est autorisé car le rayon n'intersecte pas le front. (c) On s'intéresse à la contraction de l'arête contenant le point introduit à l'étape précédente et un point près du front. (d) De la même façon que pour l'étape (b), on estime la position du point généré par la contraction de l'arête, ainsi que le rayon du cercle associé. Dans ce cas la contraction de l'arête est retardée car le cercle intersecte le front.*

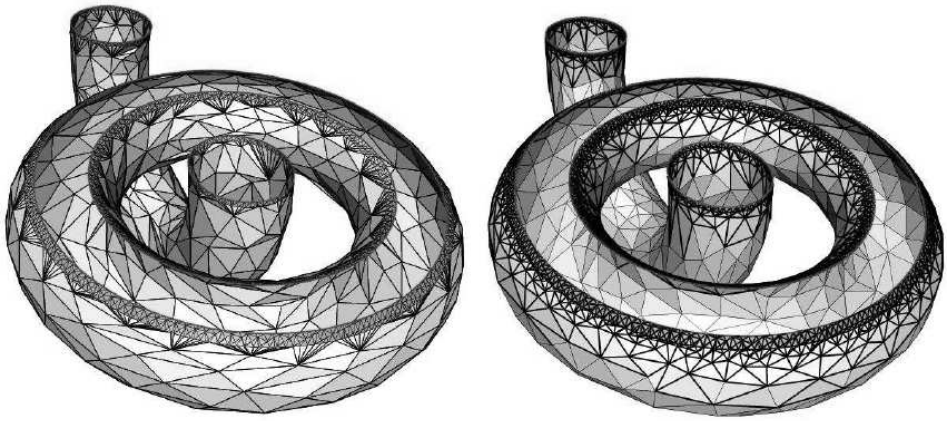


FIG. *2.29: Des surfaces partiellements reconstruites. A gauche, la surface est extraite sans utilisation du time lag. A droite, l'extraction utilise le time lag. Figure issue de [6].*

proposons donc une extension de cet algorithme permettant de l'adapter aux tailles des données manipulées dans les géosciences.

2.6 Parallélisation des méthodes d'extraction d'isosurfaces

A notre connaissance, peu d'algorithmes génèrent une isosurface simplifiée en utilisant un paradigme parallèle. Nous allons dans cette partie décrire quelques méthodes d'extraction d'isosurfaces sur des architectures parallèles. Certaines notions nécessaires à la compréhension des architectures parallèles et à l'évaluation des méthodes sont disponibles dans l'annexe C.

Newman et Yi [72] ont proposé une classification très complète des méthodes parallèles d'extraction d'isosurfaces. Leur classification est basée sur le paradigme de parallélisme utilisé et sur le fait que la répartition de charge soit dynamique ou non.

Il est important de préciser que la stratégie de SISMAGE© pour le calcul parallèle impose une répartition de charge dynamique. En effet, le nombre de machines allouées pour un calcul varie au cours de celui-ci. La stratégie choisie répartit les machines sur les calculs en cours : si un nouveau calcul est lancé sur la grille, des machines seront retirées aux calculs déjà présents qui en utilisent le plus afin que ce nouveau calcul puisse commencer. La machine cible est de type MIMD. Nous allons donc nous intéresser aux méthodes de répartition dynamique sur cette architecture présentes dans la littérature.

Miguet et Nicod Une des premières méthodes d'extraction parallèle d'isosurfaces avec répartition de charge dynamiques a été proposé par Miguet et Nicod [68] en 1995. L'algorithme est destiné à des architectures MIMD avec mémoire partagée. La répartition du volume sur les

processeurs est faite couche par couche. A l'initialisation du calcul, les couches sont distribuées de manière équitable sur tous les processeurs. Durant le calcul, la charge de travail pour l'extraction d'une couche est estimée pour chaque processeur afin de les redistribuer. Une étude expérimentale, faite sur l'accélération, montre le gain de leur méthode de répartition de charges en comparaison d'une répartition équitable des couches.

Bartz et al Dans l'article de 1998, Bartz et al [8] proposent une méthode d'extraction d'isosurfaces basée sur une structure hiérachique du type octree min-max. La construction de l'octree est réalisée en parallèle de manière asynchrone, c'est-à-dire que la liste des noeuds à découper est placée dans une file. Chaque processeur récupère un noeud à traiter (le premier dans la file) et calcule ses valeurs min et max, si le noeud doit être découpé de nouveau alors ses huit fils sont placés à la fin de la file afin d'être traités plus tard. Lors de la phase d'extraction de l'isosurface, l'octree est parcouru afin de déterminer la liste des feuilles à extraire. Cette liste de blocs à traiter sera vidée par les processeurs de la même manière que lors de l'étape de construction de l'octree : quand un processeur est libre, il récupère le bloc suivant dans la file et extrait la surface dans celui-ci. Dans l'article, Bartz et al proposent des mesures d'efficacité de leur algorithme dont les résultats varient beaucoup en fonction du jeux de données traité. Il est alors difficile d'évaluer la performance réelle de la méthode. Ils présentent par contre une méthode dynamique asynchrone pouvant aisément être mise en œuvre.

Gernster et Rumpf Gernster et Rumpf ont proposé dans [36] une méthode parallèle pour extraire une isosurface simplifiée à l'aide d'une représentation multi-résolution. L'algorithme est destiné à des architecture MIMD avec mémoire partagée. Le volume est décomposé en tétraèdres. Ils calculent ensuite une hiérarchie de tétraèdres en le découpant successivement en deux. La répartition des charges est dynamique, en effet chaque processeur est initialement responsable d'une sous partie de la hiérarchie mais si il a terminé son traitement il peut récupérer une sous partie d'un graphe appartenant à autre processeur. Les tests décrits dans le papier ne donnent les temps de calcul que pour un et quatre processeurs. L'accélération pour quatre processeurs est d'environ 2,4. L'utilisation de graphes pour répatir la charge sur les processeurs permet de maintenir une cohérence dans les données leur permettant la génération d'une surface simplifiée.

Chiang et al Dans l'article [18], Chiang et al ont décrit un schéma "out-of-core" adapté aux cluster de calcul. La première étape de leur algorithme est un pré-traitement qui décompose le volume en sous-volumes appelés "meta-cell", stockés sur le disque. Ils construisent ensuite une structure hiérarchique basée sur les boites englobantes de ces sous-volumes. Lors de la phase d'extraction de la surface, un processeur *maître* distribue un sous-volume à chacun des processeurs *esclave*. Ces processeurs reçoivent un nouveau sous-volume à extraire dès qu'ils ont terminé l'extraction courante. Le rendu de l'isosurface est prise en compte dans le modèle proposé par Chiand et al, il utilise un fichier contenant des informations sur les sous-volumes, notamment la boite englobante, afin d'optimiser l'ordre d'extraction et le rendu de l'isosurface.

2.7 Extension du *tandem algorithm*

Le *tandem algorithm* fut écrit pour extraire des isosurfaces sur de gros volumes de données mais les résultats expérimentaux (tester sur un volume de taille $1626 \times 7028 \times 2000$) nous a montré les limites de l'algorithme. Sur ce cube, l'extraction de chaque couche peut générer 1 250 000 triangles (voir figure 2.30). Dans ce cas, la mise à jour des structures (file à priorité) et l'étape de simplification peuvent être trop gourmandes en mémoire. Nous allons donc proposer une extension de cet algorithme pour augmenter sa capacité de montée en charge. Cette extension consiste à migrer les parties terminées de la surface vers le disque et à répartir le volume sur plusieurs processus.

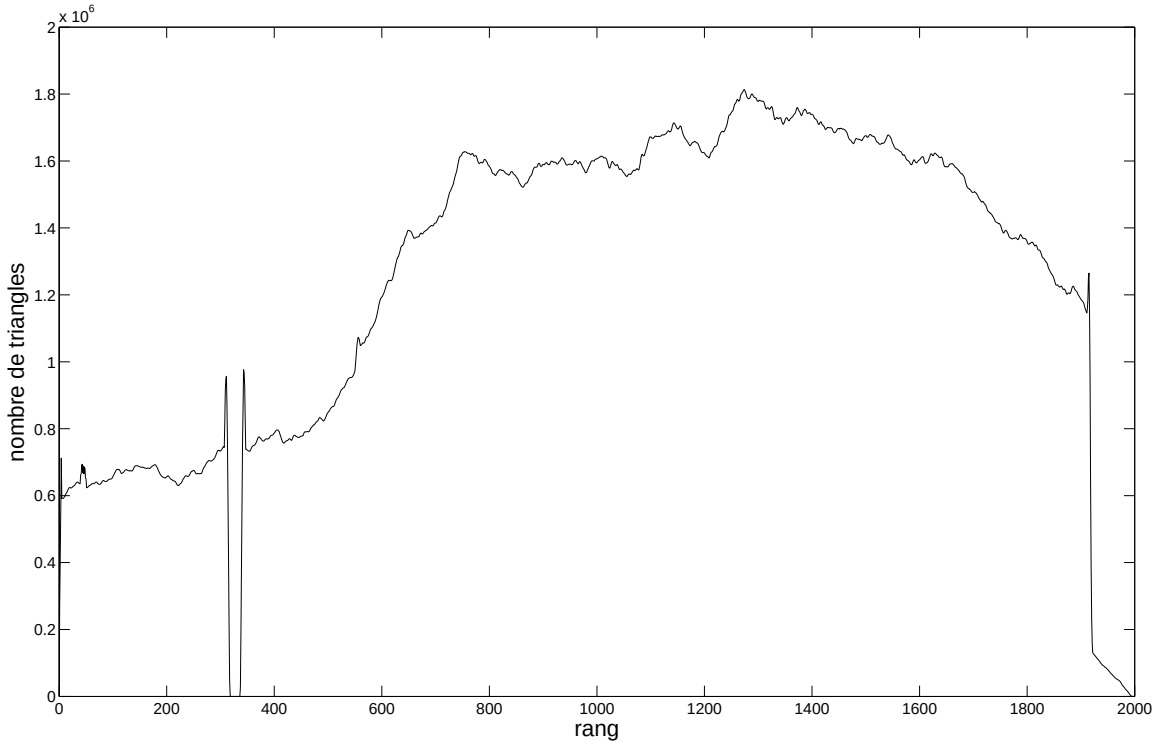


FIG. *2.30: Nombre de triangles générés par l'extraction de chaque couche avec le marching cube sur un cube de dimensions* $1626 \times 7028 \times 2000$. *La portion sans triangles est dûe à l'absence de donnée dans une partie du volume.*

2.7.1 Migration des composants

Les isosurfaces, particulièrement celles modélisant des objets issus des géosciences, génèrent de nombreux composants déconnectés. La figure 2.31 illlustre pour un front donné, un exemple de répartition des objets extraits : les composants en cours d'extraction sont en gris foncés et ceux terminés en gris clair. Nous observons que de nombreux corps sont complètement extraits, et il n'est pas nécessaire de les garder en mémoire dans la suite de l'extraction : nous pouvons ainsi les faire migrer vers le disque de stockage.

FIG. *2.31: Distribution usuelle d'évènements géologiques. Les composants terminés sont en gris clair. Les composants en cours d'extraction ou de simplification sont en gris foncé. La flèche indique la direction d'extraction.*

Les approches "out-of-core" classiques font migrer la surface tout au long du processus sans autre organisation que l'ordre d'extraction. La migration d'un composant nécessite de pouvoir détecter la fin de son extraction ainsi que celle de sa simplification.
Dans le but de formaliser la *terminaison* d'un composant, nous définisssons un composant actif γ comme l'ensemble des points qui définissent sa surface. On pose Γ comme l'ensemble des composants actifs tel que $\Gamma = \{\gamma_i\}$ et V_w l'ensemble de tous les points qui définissent une arête dans la file de priorité $\mathcal{W}$, c'est-à-dire toutes les arêtes dont la simplification a été retardée par le *time lag*. Un composant est alors terminé si il n'a plus d'arêtes pouvant être contractées. Cette proposition peut s'écrire :

$$\gamma \text{ est terminé} \Leftrightarrow \gamma \cap V_w = \emptyset$$

Nous définissons la fonction $save(\gamma)$ qui fait migre un composant vers le disque et la fonction $clear(\gamma)$ qui efface le composant γ de la triangulation courante. Nous pouvons alors écrire la fonction *Dumping* dans l'agorithme 2.3.

```
pour chaque γ ∈ Γ faire
    si γ ∩ V_w = ∅ alors
        save(γ) ;
        clear(γ) ;
    fin
fin
```

ALGORITHME 2.3 : *Fonction de migration : dumping*

La fonction *dumping* peut alors être introduite dans le *tandem algorithm*. On obtient donc l'algorithme 2.4.

```
Entrées : Réel : E_0
pour k ← 1 à nombre de coupes − 1 faire
    extract(k) ;
    delay(k) ;
    activate(k) ;
    simplify(E_0) ;
    dumping() ;
fin
activate(∞) ;
simplify(E_0) ;
dumping() ;
```

ALGORITHME 2.4 : *Tandem algorithm avec migration des composantes terminées*

Lors de la migration d'un composant, nous calculons des propriétés liées à sa géométrie : boite englobante orientée, volume, nombre de points et de facettes, etc. Ces propriétés sont utilisées par le géophysicien pour trier les corps extraits.
Une propriété intrinsèque de notre technique de migration est que le fichier généré est ordoné par composants. Nous pouvons alors accélérer ce tri en indexant les composants dans le fichier afin d'accéder facilement à un sous-ensemble de composants en ne lisant que des sous-parties du fichier. Nous illustrons ce principe dans la figure 2.32, où nous associons au fichier contenant les géométries des composants (nommé "raw data file"), un fichier d'index (nommé "index component file").

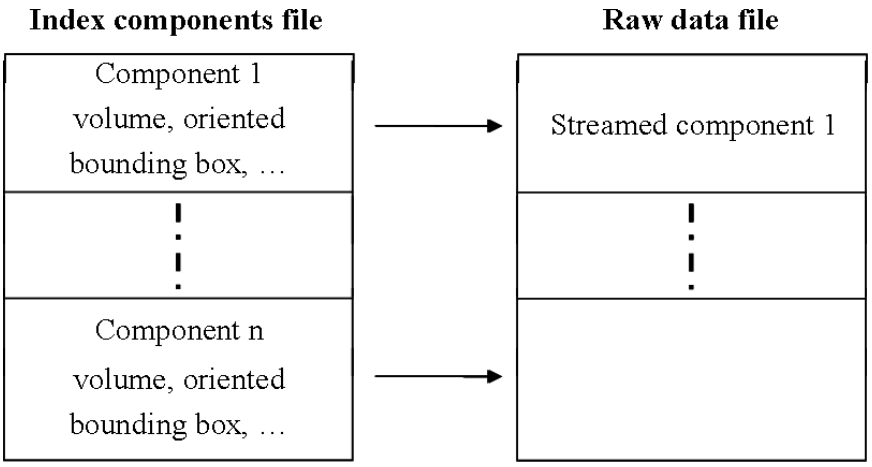


FIG. *2.32: Organisation des fichiers : le fichier d'index (à gauche) référence les composants dans le fichier contenant la géométrie (à droite).*

Un scénario d'exploration pourrait être basé sur un filtre sur le volume des objets comme dans [77] où Pivot et al décrivent une méthologie pour l'interprétation de volume sismique complexe. Ils proposent d'extraire des isosurfaces sur des attributs sismiques et de détruire automatiquement les petites "bulles" (des composantes ayant un petit volume en comparaison aux autres composants). Une autre approche est une analyse des composantes en fonction de la direction de dépot sédimentaire (azimuth).

2.7.2 Algorithme parallèle

Une des particularités de l'algorithme des *marching cubes* est sa granularité très faible. En effet, le calcul des facettes d'une cellule de la grille peut être fait indépendamment des autres cellules de la grille. Nous proposons donc de partager le volume sismique en plusieurs sous volumes, d'extraire les surfaces simplifiées sur ces sous-volumes en utilisant notre extension du *tandem* et finalement de fusionner ces surfaces pour obtenir la surface globale.

Partitionnement du volume

Nous proposons un schéma hiérarchique pour l'extraction de surface. Comme dans [70], le volume est récursivement découpé en deux perpendiculairement à sa direction la plus longue tant que la taille des sous blocs est plus grand qu'un critère donné (voir figure 2.33). L'extraction des isosurfaces est faite sur les feuilles de l'arbre généré par le découpage et fusionnées récursivement pour recomposer la surface globale.

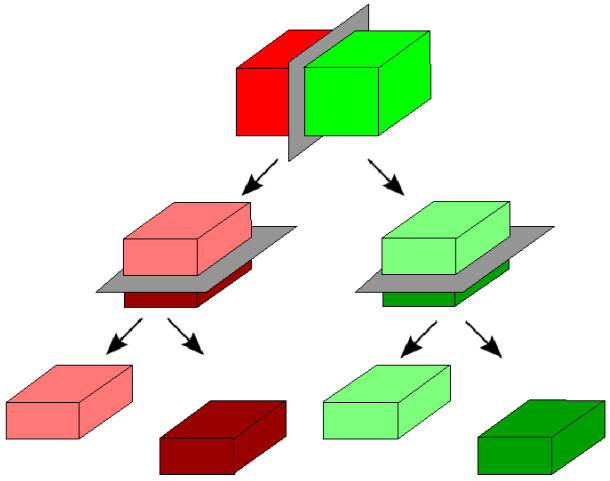


FIG. 2.33: *Partition binaire*

L'extraction des isosurfaces sur les feuilles est faite avec le *tandem algorithm* et la migration des composantes. Dans le but de maintenir les raffinements de la surface le long de sous blocs adjacents et de prévenir des artefacts introduits par la simplification d'une surface incomplète, nous devons étendre la notion de *time lag* sur les bords d'un sous-bloc. De la même façon que pour le *time lag* original le *radius* d'un point c (résultat de la contration $ab \mapsto c$) est défini par :

$$rad(c) = (||a - b|| + rad(a) + rad(b))/2$$

avec $rad(x) = 1$ si x est un sommet original de l'isosurface. La contraction de l'arête ab est retardée tant que la sphère centrée sur le milieu de ab et de rayon $rad(c)$ n'est pas totalement incluse dans le bloc. Cette contrainte s'exprime

$$\begin{array}{lll} | \quad x_{max} - (x_a + x_b)/2 \quad | & < & rad(c) \\ | \quad x_{min} - (x_a + x_b)/2 \quad | & < & rad(c) \\ | \quad y_{max} - (y_a + y_b)/2 \quad | & < & rad(c) \\ | \quad y_{min} - (y_a + y_b)/2 \quad | & < & rad(c) \\ | \quad z_{max} - (z_a + z_b)/2 \quad | & < & rad(c) \\ | \quad z_{min} - (z_a + z_b)/2 \quad | & < & rad(c) \end{array}$$

où $[x_{min}, x_{max}]$, $[y_{min}, y_{max}]$ et $[z_{min}, z_{max}]$ sont les limites du bloc.
Nous montrons dans la figure 2.34 comment l'extension du *time lag* influence le raffinement progressive de la surface aux bords des blocs ainsi que la qualité de la surface recomposée.

Recomposition de la surface globale

Dans notre stratégie de découpage (partitionement binaire), chaque bloc p de la structure hiérarchique est découpé en deux parties p_1 et p_2, appelés enfants de p notés $c_p = \{p_1, p_2\}$.

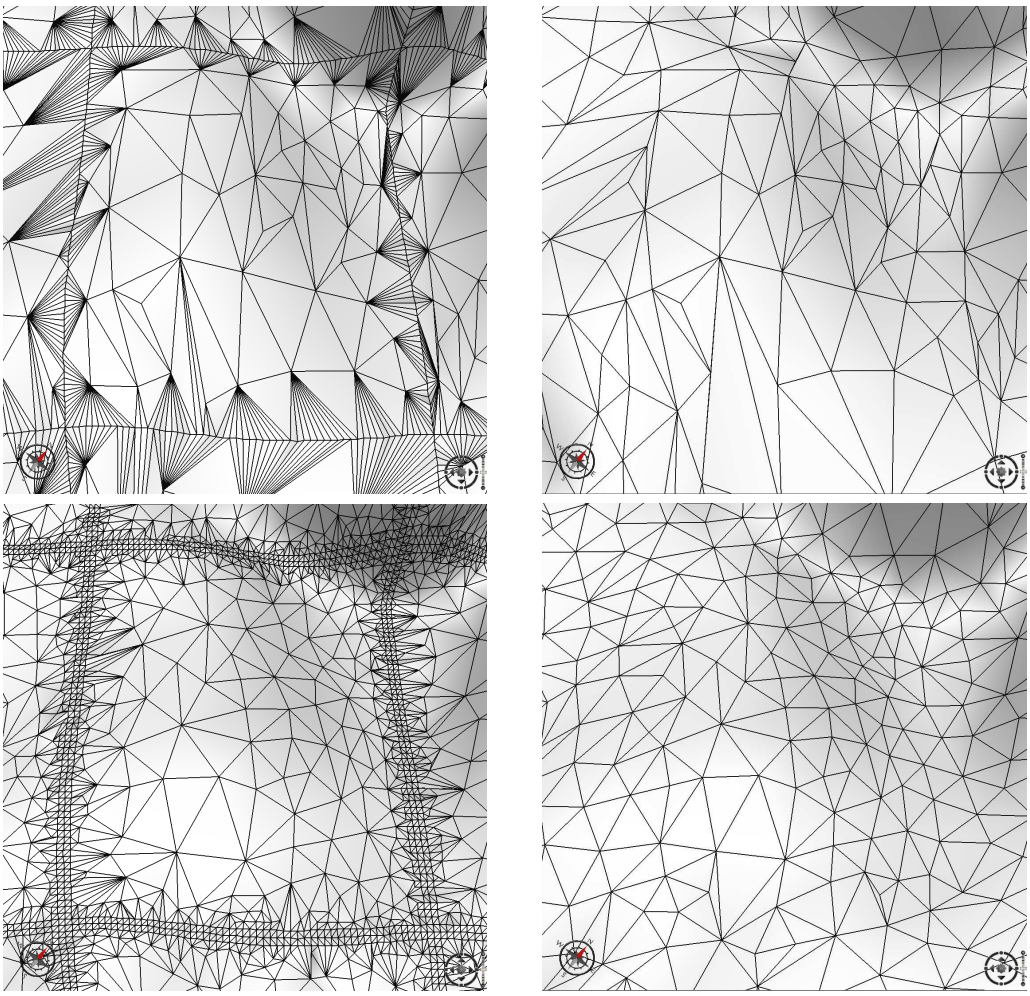


FIG. *2.34: À gauche, les figures présentent les surfaces avant simplification. À droite, les surfaces ont été simplifiées. Dans la ligne du haut, le processus prévenant la contraction des arêtes à proximité des bords utilise un critère simple : si un des points de l'arête appartient à un bord alors on retarde sa contraction. Dans la ligne du bas, les contractions d'arêtes ont été prevenues en utilisant le time lag étendu. Ces figures montrent clairement l'influence du time lag dans l'amélioration de la qualité de la surface simplifiée.*

Réciproquement p est le père de p_1 et p_2 noté respectivement f_{p_1} et f_{p_2}. p_1 est le frère de p_2 noté $b_{p_2} = p_1$ et réciproquement $b_{p_1} = p_2$.

Intuitivement, la stratégie optimale pour recomposer la surface globale serait de fusionner récursivement les surfaces entre frères. En pratique, l'isosurface n'est pas distribuée uniformement dans les sous blocs : ce qui implique que le temps d'extraction de deux frères peut-être drastiquement différent et leur fusion nécessite d'attendre le plus lent. Afin de contourner ce problème, nous autorisons la fusion d'un noeud avec son frère ou avec un enfant de son frère si ils ont un bord commun.

Nous devons donc définir un critère pour déterminer si deux blocs ont un bord en commun. On défini le *type* d'un bloc suivant sa position par rapport au plan qui a partagé son père. Si un noeud est partagé le long de x, p_1 et p_2 sont typés respectivement *gauche* et *droite*. De manière similaire, une coupe le long de y génère des noeuds typés *haut* et *bas*, et une coupe le long z des noeuds *devant* et *derrière*. On défini l'opérateur d'opposition $\overline{\cdot}$ par :

$$\overline{gauche} = droit \text{ et } \overline{droit} = gauche$$
$$\overline{haut} = bas \text{ et } \overline{bas} = haut$$
$$\overline{devant} = derriere \text{ et } \overline{derriere} = devant$$

On définit l'*arbre* de racine p, noté T_p, l'ensemble des noeuds contenant p et ses enfants récursifs. Dans cet arbre, l'*ascendance* d'un bloc n, noté $A_{n,\,p}$, est défini par l'ensemble des blocs contenant n et ses pères récursifs jusqu'à p :

$$A_{n,\,p} = \begin{cases} n & si\ n = p \\ n \cup A_{f_n,\,p} & sinon \end{cases}$$

Si on considère un bloc partagé en p_1 et p_2, alors un bloc p appartenant à l'arbre T_{p_1} possède un bord commun avec p_2 si :

$$\forall\ p' \in A_{p,\,p_1},\ type(p') \neq \overline{type(p_2)}$$

Le block p peut donc fusionner avec p_2 si il vérifie ces conditions.

La fusion de deux blocs nécessite alors d'identifier les sommets des triangles extraits sur ces blocs qui sont communs. En raison des imprécisions numériques, ces points communs ne peuvent pas être identifiés par une simple comparaison de leurs coordonnées. L'identification est faite grâce à une propriété des *marching cubes* : une arête de cellule ne peut contenir que zéro ou un sommet de l'isosurface, alors si deux points appartiennent à la même arête ils sont identiques et fusionnés.

Après cette étape de fusion, il faut simplifier la zone recollée en respectant le même critère d'erreur E_0 que lors de l'extraction des isosurfaces. De la même façon que durant l'extraction, la contraction d'une arête $ab \mapsto c$ est retardée tant que la sphère centrée sur le milieux de ab et de rayon $rad(c)$ n'est pas totalement incluse dans les blocs fusionnées. Nous montrons dans la figure 2.34 la fusion de surfaces et les effets positifs de l'extension du *time lag* sur la qualité de la surface résultat. Finalement, la dernière étape de la fusion de bloc consiste à faire migrer vers le disque les surfaces recomposées qui sont terminées.

2.7.3 Implémentation maître/esclave

Dans SISMAGE©, la machine parallèle utilisée est du type MIMD et l'équilibrage de charge entre les différentes tâches soumises par les utilisateurs est géré par Platform© LSF-HPC. La politique d'utilisation de cette machine dans SISMAGE© consiste à découper notre calcul en tâches indépendantes et à les soumettre au manager (ici LSF-HPC) afin que tout les utilisateurs du logiciel puisse accéder à la machine parallèle. Quand une tâche du calcul est terminée le manager choisit soit d'attribuer le processeur libéré au même calcul si il lui reste des tâches en attente et que la machine parallèle n'est pas sollicitée par d'autres utilisateurs soit d'attribuer le processeur à un autre calcul afin d'équilibrer le nombre de processeurs entre les utilisateurs.

Pour s'adapter aux variations du nombre des processeurs alloués pendant le calcul, nous proposons une architecture composée d'*esclaves* et d'un *maître*. Les esclaves sont des processus distants qui doivent extraire, fusionner et faire migrer les surfaces vers le disque. Le maître est un processus, distant ou non, qui distribue les tâches aux esclaves. Cette architecture, et notre formalisme, sont adaptés à notre machine parallèle contraint par la politique d'attribution de ressource propre à SISMAGE© mais peuvent s'adapter à d'autres architectures parallèles.

Les *esclaves* peuvent s'écrire comme une boucle infinie qui interroge le maître pour connaître la tâche à accomplir. Une tâche est composée d'un *type* et d'un *identifiant* (chaque esclave possède un identifiant unique utilisé pour échanger des messages). On distingue quatre types de tâches :

- *EXTRACT* : l'esclave doit extraire l'isosurface dans un sous bloc du volume. L'extraction est faite avec notre extension du *tandem algorithm* : les arêtes à proximité des bords du sous bloc, exeptés ceux qui sont communs avec les bords du volume, sont préservés en utilisant le *time lag* et les composants terminés sont déplacés vers le disque. L'isosurface extraite est ajoutée à la surface courante de l'esclave permettant ainsi une nouvelle étape de simplification et de migration.
- *SEND* : l'esclave reçoit un identifiant indiquant un esclave *cible*. Il doit envoyer sa surface courante à cet esclave cible qui la fusionnera avec sa propre surface. Après cet envoi, la surface courante de l'esclave est vide.
- *MERGE* : l'esclave doit fusionner sa surface courante avec celle envoyée par un autre esclave. Après cette fusion, il applique une étape de simplification et une étape de migration.
- *FINISHED* : l'esclave doit s'arrêter. Il faut préciser que la durée de vie d'un esclace est différente du temps de calcul. En effet, lors des dernières fusions, le processus a besoin de moins en moins d'esclaves. Il est aussi possible qu'il faille libérer des resources sur la machine parallèle, dans ce cas les processus réduiront le nombre d'esclaves.

L'algorithme 2.5 décrit le pseudo code d'un esclave.

Le *maître* est une boucle qui itère tant que la surface globale n'a pas été totalement extraite et recomposée. Il a une vue globale des processus et distribue dynamiquement les tâches sur les

```
Entrées : Réel : E0
Bouléen : finished ← faux ;
Tâche : task ;
tant que ¬ finished faire
    task ← get_task() ;
    suivant task.type faire
        cas où EXTRACT
            tandem(E0);
            break ;
        fin
        cas où SEND
            send_surface(task.target);
            break ;
        fin
        cas où MERGE
            merge_surface(get_surface(), E0);
            break ;
        fin
        cas où FINISHED
            finished ← vrai;
            break ;
        fin
    fin
fin
```

ALGORITHME 2.5 : *Pseudo code d'un esclave*

esclaves. Ceux-ci demandent de nouvelles tâches dès lors qu'ils ont terminé leurs tâches courantes. Afin de définir le méchanisme du maître (algorithme 2.6), nous avons défini les trois opérations suivantes : *has_block*, *has_neighbor* et *has_surfaces*.

La fonction *has_block* détermine si il reste des cubes (feuilles du découpage binaire récursif) qui n'ont pas encore été extraits. Si il reste des parties du volume a extraire, un bloc est envoyé à un esclave. Si cet esclave possède une triangulation non vide, le bloc à extraire doit avoir au moins un bord commun avec les blocs déjà extraits par celui-ci. Si l'esclave n'a pas de surface courante, alors un bloc quelconque lui est envoyé.

La fonction *has_neighbor* determine si l'esclave réclamant une tâche possède un *voisin* parmi les autres esclaves en cours d'exécution, c'est-à-dire un autre processus dont les blocs ont au moins un bord commun avec les siens. Si l'esclave a des voisins, il doit alors envoyer sa surface à l'un d'entre eux. Si il n'en a pas, le maître cherche un bloc non extrait à lui envoyer.

La fonction *has_surface* détermine si l'esclave a des surfaces envoyées par d'autres *esclaves* en attente. Si c'est le cas l'esclave doit fusionner ces surfaces avec sa surface courante. Cette opération est prioritaire car elle permet de diminuer la consommation mémoire, en effet les

surfaces étant raffinées aux bords, la fusion permet de les simplifier.

```
tant que ¬ finished faire
    id ← get_task_request_id() ;
    si has_surface(id) alors
        send_task(id, MERGE) ;
    sinon
        si has_neighbor(id) alors
            send_task(id, SEND, targetId) ;
        sinon
            si has_block() alors
                send_task(id, EXTRACT) ;
            sinon
                send_task(id, FINISHED) ;
            fin
        fin
    fin
fin
```

ALGORITHME 2.6 : *Pseudo code du maître*

2.8 Évaluation de l'algorithme

Afin d'analyser les performances de notre algorithme, nous nous sommes intéressé à la mémoire utilisée durant l'extraction, à la qualité des surfaces générées et au temps de calcul. Ces évaluation sont faites sur un cube de dimensions $401 \times 1051 \times 2001$. Un résultat d'extraction d'isosurfaces sur ce cube est visible dans la figure 2.35

2.8.1 Consommation mémoire

Un point critique dans l'utilisation des *marching cubes* est le nombre de triangles générés et donc la quantité de mémoire nécessaire à l'extraction de l'isosurface. La figure 2.36 illustre la comparaison entre une approche dites brutale (extraction complète de l'isosurface puis simplification), le *tandem algorithm* avec et sans migration des composants. Dans la figure, les triangles sont comptés juste après le traitement d'une couche k. Nous pouvons constater que le *tandem algorithm* réduit drastiquement le nombre de triangles en comparaison de l'approche classique. Notre stratégie de migration améliore encore la réduction du nombre triangles présents en mémoire. Sur l'exemple choisit, le *tandem algorithm* nécéssite 10 fois moins de mémoire que l'algorithme classique des *marching cubes*, et notre approche 100 fois moins.

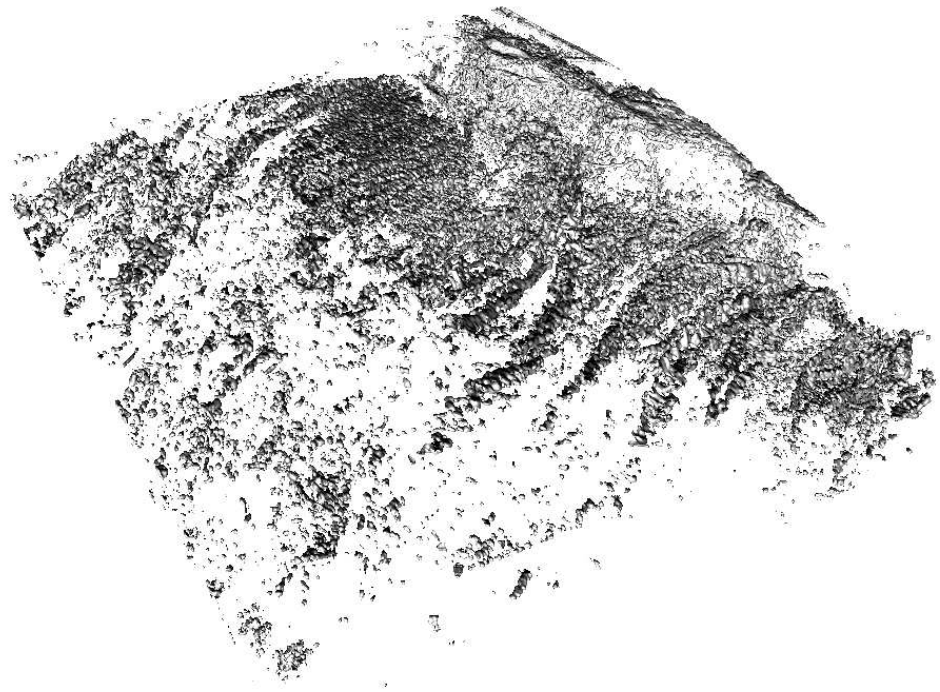


FIG. *2.35: Extraction d'une isosurface sur un cube de dimensions* $401 \times 1051 \times 2001$

2.8.2 Qualité du maillage

De nombreuses applications nécessitent des triangles de bonne qualité et non des triangles longs et fins. Un critère d'évaluation de la forme d'un tiangle $t = abc$ est $\rho(t) = \sqrt{\lambda_2/\lambda_1}$ où $\lambda_1 \geq \lambda_2$ sont les plus grandes valeurs propres de la matrice d'inertie de t. Cette matrice est définie par :

$$M_t = \frac{1}{3}[(a-\hat{t})(a-\hat{t})^T + (b-\hat{t})(b-\hat{t})^T + (c-\hat{t})(c-\hat{t})^T]$$

avec $\hat{t}$ le centroïde du triangle t. $\rho(t) = 1$ si le triangle t est équilatérale ($\lambda_1 = \lambda_2$) et $\rho(t) = 0$ si il est dégénéré ($\lambda_2 = 0$). Attali et al propose une mesure globale de l'*anisotropie* d'une triangulation K (composé de n triangles) :

$$anisotropy(K) = 1 - \frac{1}{n}\sum_t \rho(t)$$

avec $0 \leq anisotropy(K) \leq 1$ et $anisotropy(K) = 0$ si tout les triangles sont dégénérés.

La figure 2.37 montre les variations de l'*anisotropie* en fonction du paramètre d'isotropie α pour trois approches : le *tandem algorithm* séquentiel, la version parallèle avec et sans *time lag*. Nous pouvons clairement constater les effets du *time lag* sur la qualité des maillages générés. La version parallèle avec *time lag* génére une surface de qualité quasi équivalente à la version séquentielle de l'algorithme. En pratique, la fonction coût doit toujours tenir compte de la *mesure*

d'écart prévenant ainsi ces cas dégénérés. Attali [6] montre qu'un bon compromis pour α est 0.4, il assure une bonne qualité des triangles tout en limitant leur nombre.

Remarque : l'augmentation de l'anisotropie (pour α supérieur à 0.9) est dû au fait que le critère de *mesure d'écart* n'est plus pris en compte pour l'estimation des coûts des contractions d'arêtes. L'opération de simplification ne tient compte que de la forme des triangles, les points proposés seront loins de la surface d'origine : la majorité des candidats seront donc rejetés par le seuil d'erreur E_0.

2.8.3 Temps de calcul

Un problème majeur dans l'extraction des isosurfaces sur des volumes de grandes tailles est le temps d'exécution du calcul. C'est pourquoi nous avons proposé une version parallèle de l'algorithme. Nous définissons dans l'annexe C les notions permettants d'évaluer l'apport de la version parallèle par rapport à la version séquentielle.

L'*accélération* quantifie le gain induit par l'utilisation de plusieurs processeurs. Nous comparerons cette accélération linéaire que l'on pourrait qualifier d'idéale : *si j'utilise n processeurs mon calcul dure n fois moins longtemps.* L'*efficatité* est définie par le rapport en l'accélération effective et l'accélération idéal.
Ces deux mesures mettent en évidence des phénomènes similaires et sont indifféremment utilisée dans la littérature. Pour nos tests, nous faisons varier le nombre de processeurs entre 1 et 32.

L'étude de la courbe d'accélération, dans la figure 2.38 montre que nous sommes est très proche de l'accélération idéale tant que l'on ne dépase pas 8 processeurs. Nous pouvons comparer nos résultats à ceux de Gernster et Rumpf [36] qui obtiennent une accélération de 2,4 pour 4 processeurs contre 3,96 pour notre méthode.
La courbe d'efficacité, dans la figure 2.39, met en évidence le même phénomène : l'efficacité diminue à partir de 8 processeurs. On constate même que pour certaines valeurs concernant des calculs avec moins de 8 processeurs l'efficacité est supérieur à 1. Ces bonnes performances sont dûes à un accès très rapides et concurrents aux données par les noeuds de calculs dans notre architecture parallèle. Au dessus de 8 processeurs les latences réseaux dûes aux échanges de surfaces entre les noeuds de calcul deviennent de plus en plus conséquents. De la même façon, malgré les bonnes performances des systèmes de fichiers, les latences d'entrée-sortie augmentent, expliquant la diminution de l'efficacité. Notre paradigme reste cependant efficace et augmenter le nombre de processeurs permet de diminuer le temps de calcul.

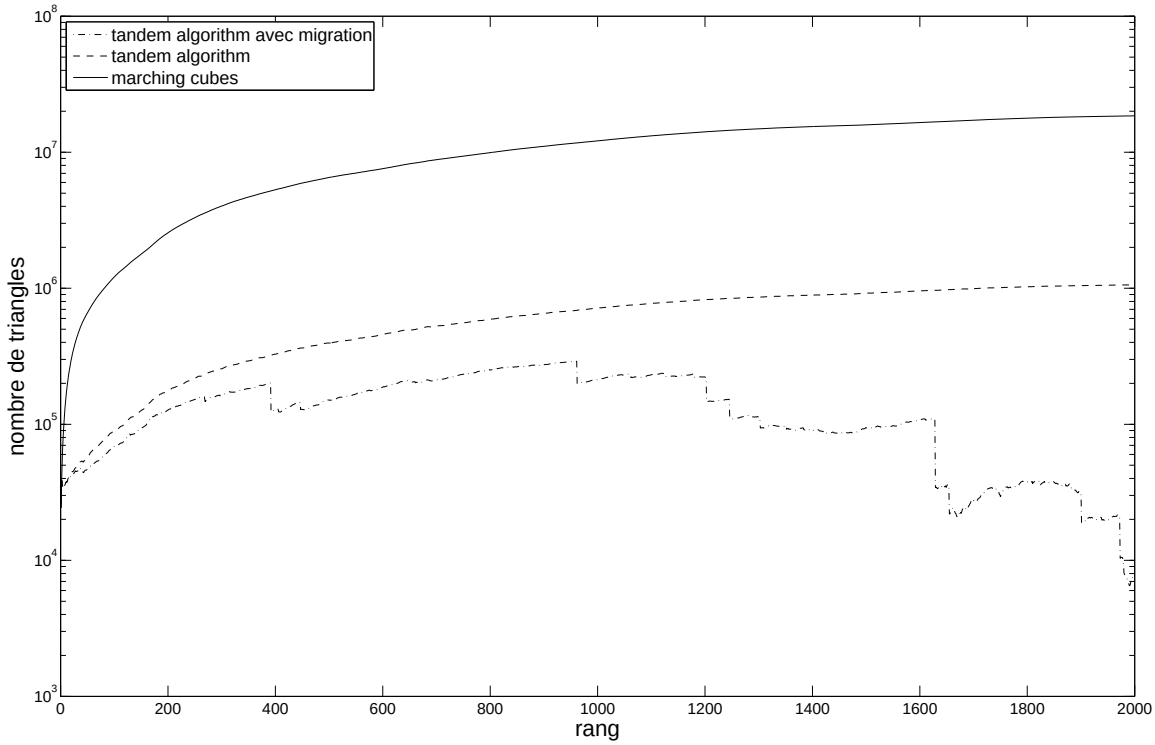


FIG. *2.36: Évolution du nombre de triangles pour trois approches : le marching cubes, le tandem algorithm avec et sans migration des composantes. Les calculs sont réalisés sur un bloc de dimensions* $401 \times 1051 \times 2001$.

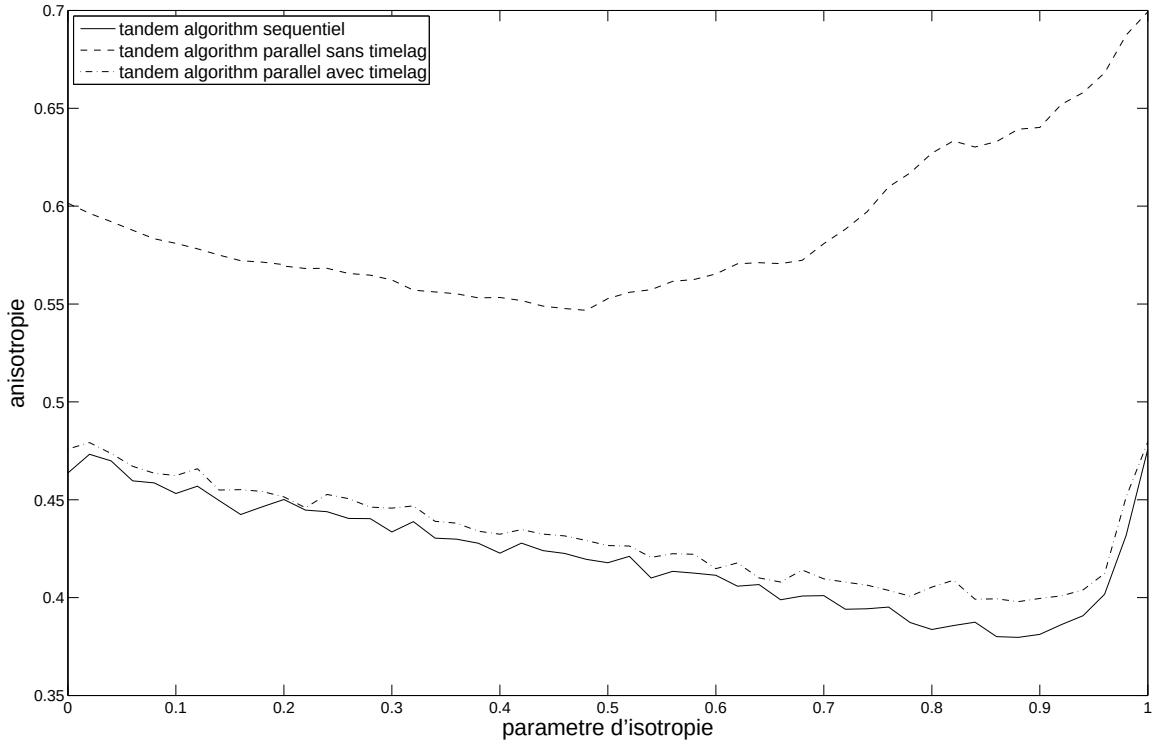


FIG. *2.37: Anisotropie de la surface variant en fonction du paramètre d'isotropie* α

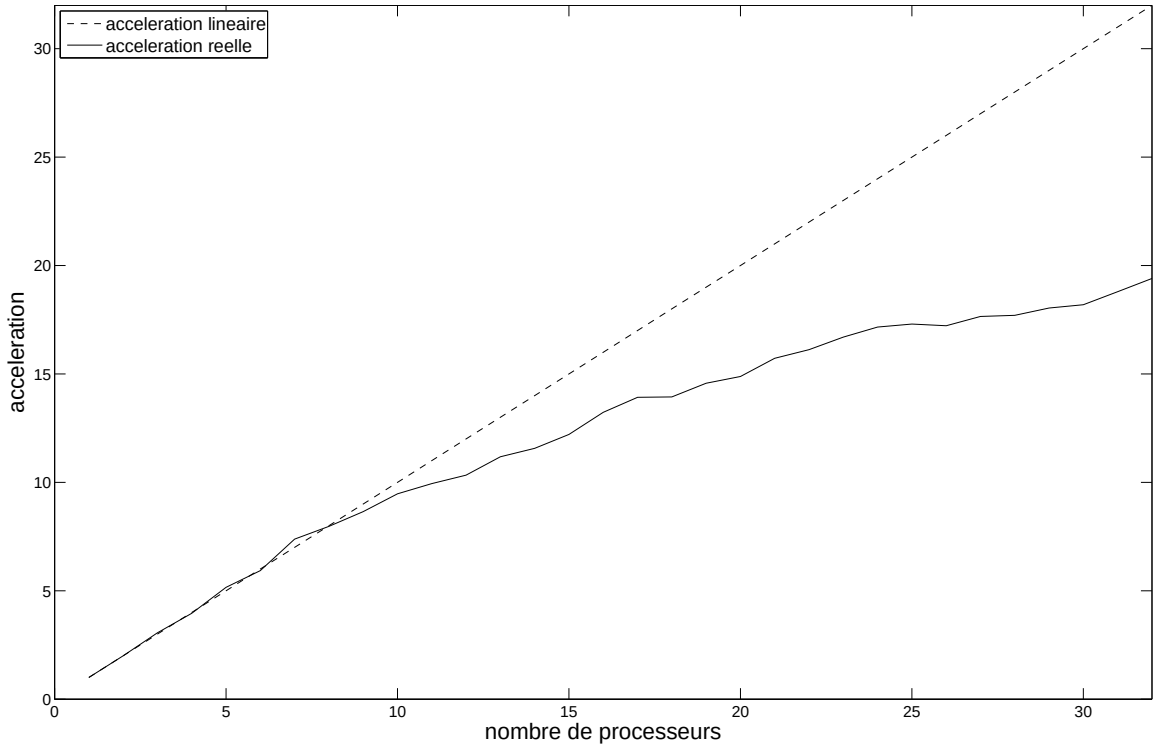


FIG. *2.38: Accélération*

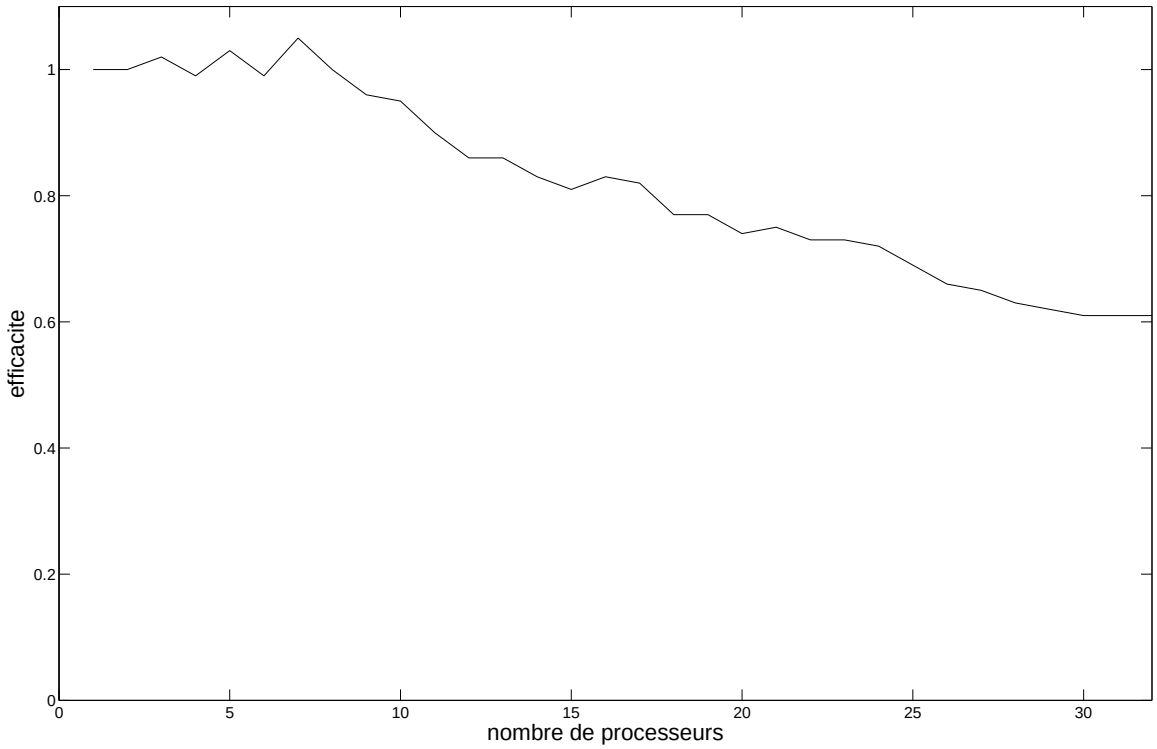


FIG. *2.39: Efficacité*

2.9 Conclusion

Nous avons présenté dans ce chapitre un algorithme parallèle d'extraction d'isosurfaces simplifiées. Notre méthode permet d'extraire des isosurfaces sur des volumes d'attributs sismiques pouvant atteindre des tailles de plusieurs dizaines de giga-octets. Pour cela nous avons défini une stratégie de partitionnement du volume qui le découpe récursivement en deux pour former une structure hiérarchique de blocs voisins. Les isosurfaces partielles sont extraites de ces blocs par des processeurs indépendants exécutant une version étendue de l'algorithme tandem. Nous avons proposé une architecture parallèle composé d'un *maître* qui commande des *esclaves*. Ceux-ci extraient, simplifient, s'échangent et assemblent les morceaux de surfaces ainsi obtenus jusqu'à l'obtention d'une isosurface simplifiée globale. Cette dernière est organisée et stockée par construction en un flux de composantes connexes qui permettra une interaction ultérieure de haut niveau.

Notre algorithme, disponible dans SISMAGE©, s'intègre à la méthologie proposée par Pivot et al dans [77] qui consiste à intégrer le résultat de l'interprétation lors de l'extraction des isosurfaces sur les cubes attributs et à filtrer ces surfaces suivant un critère de volume. Cette démarche a été appliquée sur un volume de dimensions $1626 \times 7028 \times 2000$ occupant 40 Go. Le premier intérêt de notre méthode réside dans son parallélisme qui accélère grandement le temps de calcul en fonction du nombre de processeurs disponibles sur notre machine parallèle : l'extraction a duré environ 5 minutes en utilisant 56 processeurs alors qu'il fallait plus de 3 heures avec la version séquentielle.
La figure 2.40 nous montre le surface extraite. Nous pouvons clairement constater qu'elle est composé d'énormement de petites composantes déconnectées. Profitant de notre format de stockage indexé, nous pouvons rapidement trier les surfaces sans analyser une nouvelle fois leurs géométries. La figure 2.41 illustre le résultat d'un filtre basé sur le volume : seulement 328 composants ont été conservés sur les 18 111 qui composaient initialement la surface.

Les travaux détaillés dans ce chapitre ont été présentés à la conférence *ACM Symposium on Solid and Physical Modeling* [25].

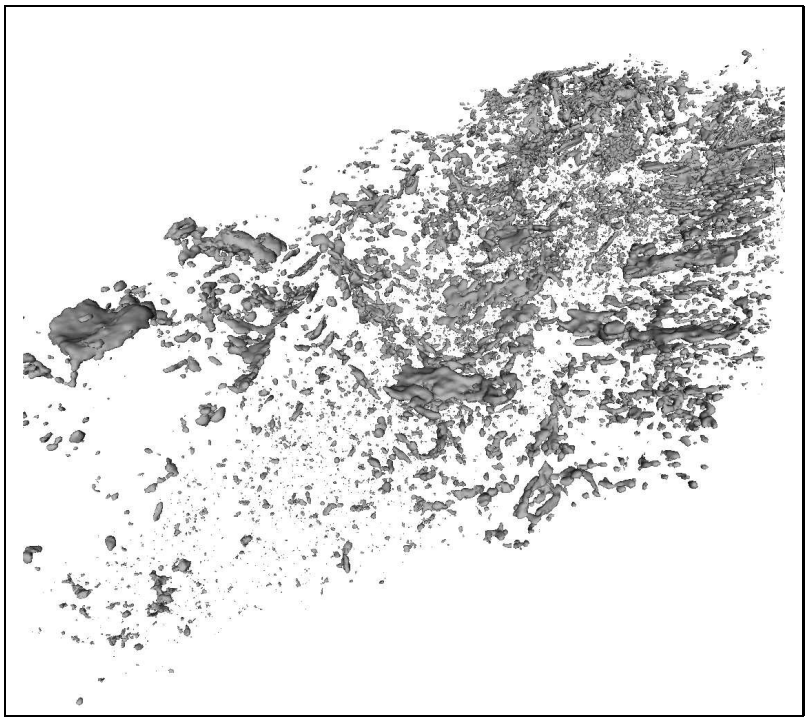


FIG. *2.40: Exemple d'une isosurface générée à partir d'un volume de dimensions* $1626 \times 7028 \times 2000$. *L'isosurface contient 18 111 composants déconnectés.*

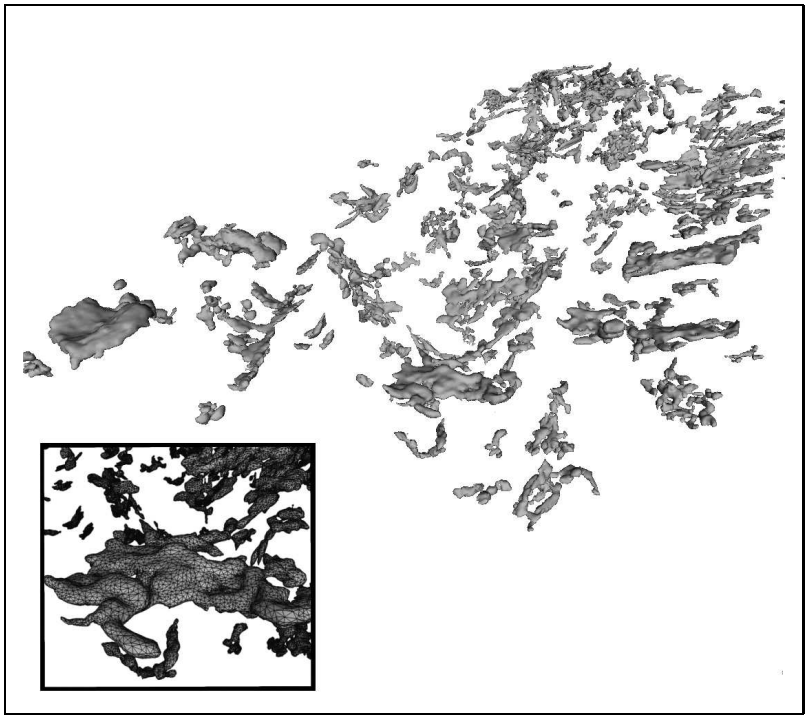


FIG. *2.41: L'isosurface extraite sur le volume de dimensions* $1626 \times 7028 \times 2000$ *(voir figure 2.40) a été filtrée en fonction du volume. Cette surface contient 328 composants déconnectés.*

Conclusion générale

Au cours de cette thèse nous avons développé des algorithmes permettant de reconstruire des surfaces modélisant des objets géologiques tel que les horizons, les failles et les corps. Le premier chapitre est dédié à la construction d'un modèle strucutral cohérent contenant les horizons et les failles. Le second chapitre traite de la reconstruction des corps à partir d'attributs sismique.

La première étape de la construction du modèle structural est la construction de chaque faille indépendamment les unes des autres. L'interprétation de chaque étant un nuage de lignes brisées, le problème est similaire à la reconstruction de surface à partir de nuage de points. Parmis les méthodes décrites dans la littérature, nous avons choisi une approche comparable à celle de Boubekeur [13] qui découpe le nuage de points en sous-ensembles afin de les projeter dans leurs plans de régression. Nous profitons de la nature quasi-planaire des failles pour nous affranchir de cette étape de découpage du nuage. Cependant, la méthode originale traitant uniquement un nuage de points, nous l'avons étendue pour un nuage de lignes brisées en faisant une triangulation de Delaunay contrainte des segments projetés dans son plan de régression. L'enveloppe réelle de la faille dans cette triangulation est déterminée en utilisant les α-shapes. Finalement, la triangulation obtenue est remenée ent trois dimensions et nous utilisons les méthodes de subdivison pour obtenir une surface lisse.

Nous rendons ensuite le réseau de failles cohérent en appliquant des opérations de *coupe* et d'*extrapolation* sur les surfaces. Pour limiter les temps de calcul, nous utilisons un octree contenant l'ensemble du réseau.

Dans la dernière étape, les horizons sont interpolés sous contrainte de ce réseau de failles. Nous introduisons les discontinuités dûes au rejet des failles dans la surface résultat en utilisant les "voisinages partiels" dans la technique d'interpolation. Nous avons mesuré l'impact de ces voisinages dans les techniques d'interpolation les plus utilisées : les méthodes barycentriques, les régressions et le krigeage. Nous faisons ensuite une étude des stratégies de remplissage dans le processus d'interpolation et proposons une méthode robuste au bruit.

Les corps sont construits par extraction d'isosurfaces sur des volumes d'attributs sismiques pouvant atteindre des tailles de plusieurs dizaines de giga-octets. Nous avons donc proposé un algorithme parallèle d'extraction d'isosurfaces simplifiées adapté à la taille de ces données. Pour cela nous découpons récursivement le cube en deux pour construire une structure hiérarchique

de blocs voisins. Des isosurfaces partielles sont extraites de ces blocs par des processeurs indépendants exécutant une version étendue par nos soins du *tandem algorithm* [6]. Ainsi, les noeuds de calcul extraient, simplifient, s'échangent et assemblent les morceaux de surfaces ainsi obtenues jusqu'à l'obtention d'une isosurface simplifiée globale. Nous proposons une organisation de l'isosurface en un flux de composantes connexes qui permettra une interaction ultérieure de haut niveau.

Tous les algorithmes présentés dans ce mémoire ont été intégrés dans SISMAGE© qui est utilisé, en moyenne, par 400 utilisateurs dans le monde.

Nous allons continuer nos travaux afin d'améliorer la construction du modèle structural par des nouvelles approches basées sur la tétraèdrisation du volume englobant le réseau de failles et sur une définition implicite de ce réseau.

Annexe A

Représentations paramétriques

A.1 Notion de paramétrisation

Une paramétrisation de surface 3D est une fonction $\chi : \mathbb{R}^2 \mapsto \mathbb{R}^3$ qui met cette surface en correspondance bijective avec un domaine 2D (voir figure A.1).

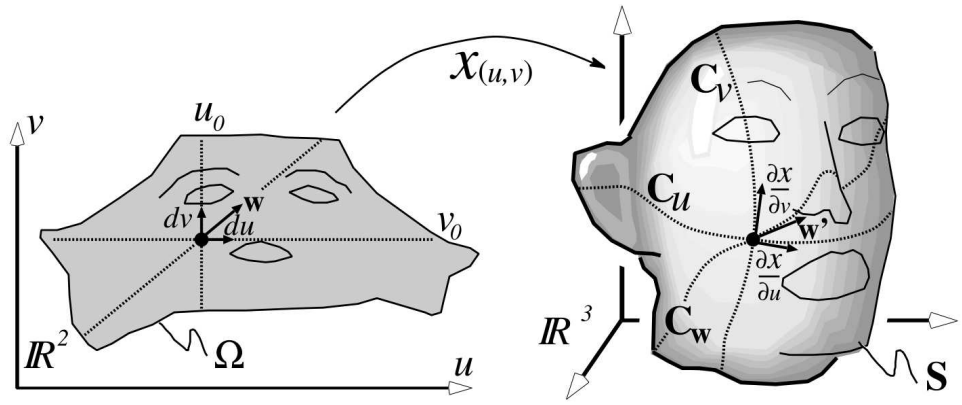


FIG. *A.1: Exemple de paramétrisation. Figure issue de [55]*

Dans le context de cette thèse, l'objectif est recontruire la surface modélisant la faille à partir de l'interprétation de la sismique (ou pointé). Cette interprétation prenant la forme d'un ensemble de lignes brisées, il faut trouver une fonction qui respecte ce nuage. Cette fonction nous permettra d'interpoler la surface entre le pointé.

A.2 Polynômes de Lagrange

Une première approche consiste à utiliser les polynômes de Lagrange basés sur les points définissants les lignes brisées. Nous allons illustrer nos propos en deux dimensions en utilisant les polynômes de Lagrange. Soit un ensemble de n points $p_i \in \mathbb{R}^2$ distincts deux à deux défini par

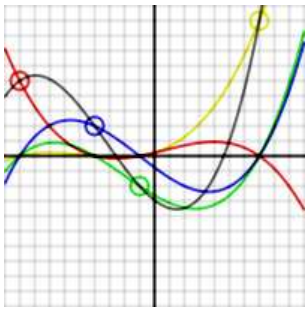
$p_i = (x_i, y_i)$. Les polynômes de Lagrange associés à ces points sont les polynômes définis par :

$$l_j(x) = \prod_{i=0, j\neq i}^{n-1} \frac{x - x_i}{x_j - x_i}$$

On appelle polynôme d'interpolation de Lagrange, le polynôme défini par

$$L(x) = \sum_{j=0}^{n} y_j l_j(x)$$

vérifiant $L(x_i) = y_i$ pour tout i. Ce polynôme d'interpolation nous permet donc de construire une fonction de $\mathbb{R} \mapsto \mathbb{R}^2$.



L'image ci-contre montre le résultat (courbe noire) de l'interpolation de la Lagrange pour quatre points $(-9, 5)$, $(-4, 2)$, $(-1, -2)$, $(7, 9)$ respectivement rouge, bleu, vert et jaune. Le polynôme d'interpolation $L(x)$, de degré trois, est la somme des polynômes de base $l_0(x)$, $l_1(x)$, $l_2(x)$ et $l_3(x)$. Le polynôme d'interpolation passe par les quatre points de contrôle, et chaque polynôme de base l_i passe par son point de contrôle respectif p_i (la couleur du polynôme dans la figure est la même que pour son point de contrôle) et vaut 0 pour les x correspondant aux autres points de contrôle.

En pratique les polynômes de Lagrange ne pourront pas être appliqués à notre problème, le pointé peut rapidement contenir plus d'une centaine de points ce qui implique un polynôme d'un degré quasi-équivalent. Ces polynômes de degré élevé mettent en évidence un phénomène connu sous le nom de "phénomène de Runge" : les polynômes de degré élevé ont tendance à s'écarter de la fonction qu'ils approximent.

Une solution consiste à *approcher* le nuage de points par un polynôme d'un degré fixé, c'est le cas des méthodes de régressions.

A.3 Régressions

Les régressions permettent d'approximer le nuages de lignes brisées par une fonction. La fonction d'interpolation f la plus utilisé est un polynôme de degré d :

$$f(x, y) = \sum_{i+j<d} \beta_{ij} x^i y^j$$

avec $\beta_{ij} \in \mathbb{R}$. En pratique, le degré est souvent inférieur ou égal à trois. Les coefficients du polynôme sont estimés par une méthode d'optimisation. La méthode la plus répandue pour

estimer ces coefficients est la méthode des moindres carrés. Il faut donc trouver les coefficients β_{ij} qui minimisent :

$$\sum_{i=n}^{n}[f(x_i, y_i) - z_i]^2$$

où $p_i = (x_i, y_i, z_i)$ est point du nuage, de cardinalité n, à approcher.

Il est cependant nécessaire de trouver la bonne fonction permettant cette approximation. C'est pourquoi cette méthode d'approximation est peu adaptée à notre problème. Dans le but de représenter des formes complexes, les surfaces peuvent être décrites par un ensemble de fonctions paramétriques notamment des splines.

A.4 Surfaces splines

Il existe dans l'industrie et dans l'infographie le besoin de représenter ces surfaces libres sans qu'elles soient connues à l'avance. A la fin des années cinquante, Pierre Bézier, ingénieur chez Renault a mis au point une méthode permettant de définir une surface par un nombre minimal de points caractéristiques. Cette méthode a pour intérêt de modifier facilement la surface en déplacant un minimum de points.

A.4.1 Bézier

Les courbes de Bézier sont des courbes polynomiales paramétriques. Une courbe de Bézier de degré n est défini par $n+1$ points caractéristiques aussi appelés *points de contrôles*. Elle est décrite par l'équation :

$$b(t) = \sum_{i=0}^{n} B_i^n(t) P_i$$

avec $t \in [O, 1]$, P_i un point de contrôle et $B_i^n(t)$ un polynôme de Bernstein défini par :

$$B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$$

La surface de Bézier est définie par une grille orthogonale définie par n_i et n_j point de contrôles dans les directions respectives i et j. Si on considère $P_{i,j}$ le i-ième et j-ième point de contrôle dans cette grille, on obtient la formulation suivante :

$$b(u, v) = \sum_{i=0}^{n_i} \sum_{j=0}^{n_j} B_i^{n_i}(u) B_j^{n_j}(v) P_{i,j}$$

A.4.2 B-spline

Une spline est une fonction définie par morceaux par des polynômes. Les courbes et surfaces de Bézier sont une formulation particulière des splines qui permettent de préserver l'aspect de la

courbe ou de la surface malgré une rotation de repère. Farin a introduit en 1988 le concept de B-spline (Basis spline) : n courbes de Bézier de degré fixe m, de continuité C^{m-1} qui interpolent $n+1$ points de contrôles.

A.4.3 NURBS

Les NURBS (Non Uniform Rational B-Spline) sont une généralisation de la représentation par les B-splines. Alors que les B-splines sont une représentation polynomiale par morceaux, les NURBS sont une représentation sous la forme de fractions rationnelles par morceaux. Elles permettent ainsi de représenter des objets qui ne peuvent pas l'être avec des B-splines.

De nombreuses recherches se sont intéressées à la construction de ces modèles paramétriques à partir d'autres représentations telles que les surfaces polygonales [46, 55].

En pratique, il n'est pas toujours possible de trouver le paramétrage adéquat lorsqu'on a des données irrégulièrement réparties, ce qui est le cas avec les pointés issus de l'interprétation. C'est pourquoi nous avons choisi d'utiliser les techniques de modélisation par des surfaces polygonales pour reconstruire les failles dans le chapitre 1.

Annexe B

Surfaces de subdivision

B.1 Surfaces de subdivisions : définition

Les surfaces de subdivisions ont été introduites en 1978 par Catmull et Clark [14], Doo et Sabin [24]. Le principe est de raffiner successivement un maillage initial "basse résolution" en subdivisant tout ou seulement une partie (on parle alors de subdivision adaptative) de celui-ci (voir figure B.1). Les surfaces de subdvision ne sont pas continues mais le maillage généré par les subdivions tend à être continue. Ces méthodes font le lien entre les approches continues et les approches polygonales (discrètes).

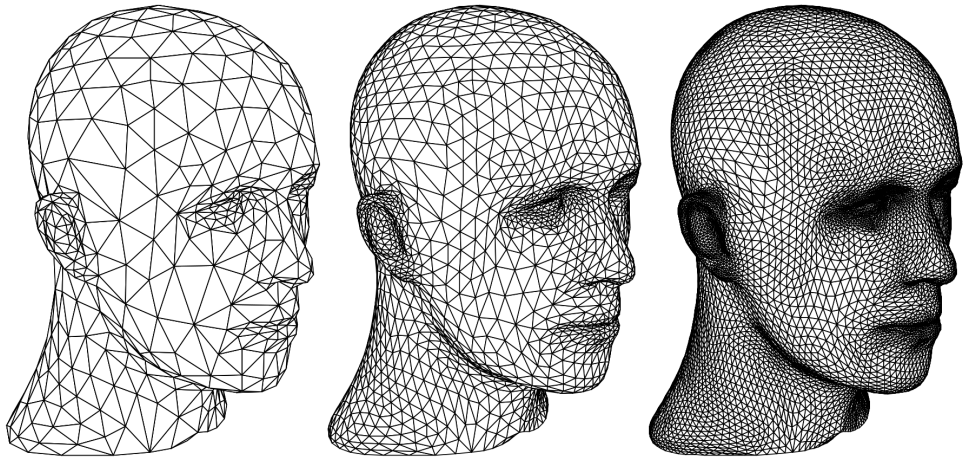


FIG. *B.1: Le maillage triangulaire d'origine (à gauche), et l'application successive de deux passes de subdivision (au centre et à droite). Figure issue de [91].*

Ces dernières années, les surfaces de subdivision ont remplacé les NURBS dans les procédés de modélisation de personnages et dans la création de la géométrie des décors dans les effets visuels de l'industrie cinématographique.

En 2000, Zorin [89] propose une liste de critères non exhaustive pour classer les différents schémas de subdivisions :

- le type de maillages auxquels ils s'appliquent (triangulaires ou quadrangulaires),
- le type de règle de raffinement. Un schéma est dit "primal" quand une face est coupée en plusieurs faces et "dual" quand un point est remplacé par plusieurs points.
- savoir si le schéma approxime ou interpole. On dit qu'un schéma interpole si les sommets présents dans la surface polyhédrale originelle sont encore des sommets de la surface résultat (voir figure B.2), approxime sinon.
- la continuité de la surface limite (C^1, C^2, ...).

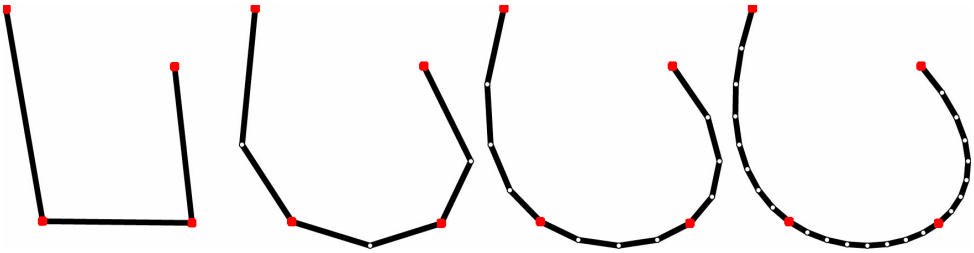


FIG. *B.2: Principe des subdivisions interpolantes : les points d'origines (en rouge) sont présents dans le résultat de chaque étape de subdivision.*

On trouve ainsi, dans la littérature une classification de schémas classiques de subdivision (voir tableau B.1). Dans la suite de cette section nous décrivons brièvement ces schémas suivant qu'ils sont approximants ou interpolants.

	Primal		Dual
	Triangulaire	Quadrangulaire	
Approximant	Loop	Catmull-Clark	Doo-Sabin
Interpolant	Butterfly	Kobbelt	Midedge

TAB. *B.1: Classification des schémas de subdivision classiques*

B.2 Différents schémas de subdivision approximants

B.2.1 Catmull-Clark

Le schéma de Catmull-Clark [14] fut proposé en 1978. Il s'agit d'un schéma dual approximant, pour maillages quadrangulaires. La surface limite a une continuité C^2 pour les maillages réguliers, C^1 sinon. C'est l'un des schémas les plus utilisés dans l'industrie, en partie grâce aux efforts faits par le studio Pixar pour augmenter le pouvoir expressif du maillage de contrôle (arêtes vives, raffinement des coordonnées de textures, animation).

B.2.2 Loop

Charles Loop [62] propose, en 1987, un schéma approximant pour maillages triangulaires. Ce schéma assure un continuité C^2 partout sauf aux sommets irréguliers.

Hoppe et al. [43] proposent une modification de ces règles au bord des surfaces, pour assurer la continuité du plan tangent. Mais ces modifications rendent l'évaluation des bords sensible aux sommets intérieurs proches. Les modifications apportées par Hoppe permettent plus de contrôle sur la surface, en autorisant de marquer des arêtes comme vives. On peut étendre cette idée avec un panel complet de contrôle local de la subdivision.

B.2.3 $\sqrt{3}$ subdivision

En 2000, Kobbelt[53] propose un schéma approximant dont la surface limite est de continuité C^2, sauf aux points extraordinaires où elle est C^1. Ce schéma n'entre pas dans les critères proposés par Zorin dans [89], en effet la retriangulation, après l'insertion d'un point, est basé sur la rotation d'arêtes.

B.3 Différents schémas de subdivision interpolants

B.3.1 Butterfly et Modified Butterfly

C'est un schéma interpolant pour maillages triangulaires. Le schéma Butterfly original [26] ne garantissait pas de continuité C^1 pour les maillages arbitraires. En 1996, zorin [90] propose une modification assurant cette propriété. Mais la surface limite n'est pas C^2, même pour les maillages réguliers.

B.3.2 Kobbelt

En 1996, Kobbelt propose un schéma primal interpolant pour maillages quadrangulaires, de continuité C^1 [52]. Il est basé sur le produit tensoriel des quatre points. Ce schéma est dérivé de l'observation suivante : la subdivision peut s'opérer en deux temps. D'abord on calcule les sommets des arêtes, puis on calcule les sommets des faces d'après les sommets des arêtes.

Contrairement aux autres schémas, les vecteurs propres de la matrice de subdivision ne peuvent être calculés explicitement. Dans ces conditions, le support effectif de ce schéma est trop large et ne permet pas de l'utiliser pour les maillages trop volumineux.

B.3.3 Doo-Sabin et Midedge

Ce sont des schémas duals simples dont les surfaces limites sont garanties C^1. Dans le schéma de Doo-Sabin, le même masque est utilisé pour les sommets pairs et impairs.

Annexe C

Calculs parallèles

C.1 Classification des environnements parallèles

Une machine parallèle est un ensemble de processeurs qui coopèrent et communiquent. La taxonomie de Flynn-Tanenbaum, datant de 1966, distingue quatre types de parallélisme : SISD, SIMD, MISD et MIMD. Cette classification est basée sur les notions de flux de contrôle (la lettre I pour "instruction") et flux de données (la lettre D pour "data"). Le flux de contrôle étant appliqué sur le flux de données.

C.1.1 Single Instruction Single Data

Une machine SISD est la machine communément appelée machine de Von Neuman. Une seule instruction est exécutée à un moment donnée et une seule donnée est traitée à un moment.

C.1.2 Single Instruction Multiple Data

Un machine SIMD est une machine qui exécute à tout instant une seule instruction, mais qui agit en parallèle sur plusieurs données. De manière générale ces machines sont composées d'un très grand nombre de processeurs élémentaires. Historiquement, les machines SIMD les plus connues sont les CRAY 1 et 2, mais plus récemment les GPU (Graphics Processing Units) font partie de cette catégorie.

Il faut aussi citer le paradigme SPMD (Single Program Mutiple Data) qui est l'équivalent logicielle du SIMD sur une machine MIMD. Dans ce paradigme, c'est le même programme qui est exécuté sur les processeurs.

C.1.3 Multiple Instruction Single Data

Dans cette architecture, plusieurs processeurs se partagent une mémoire commune. Chaque processeur effectue une opération différente au même moment sur la même donnée. C'est un modèle théorique et aucune machine n'est construite sur ce modèle.

C.1.4 Multiple Instruction Multiple Data

L'architecture MIMD est la plus générale. Dans cette approche chaque processeur peut exécuter un programme différent sur son flot de donnée. La communication de données ou de résultats entre les processeurs peut se faire à travers une mémoire commune ou un réseau d'interconnection.

En pratique, les machines MIMD sont les plus polyvalentes et les plus performantes. L'architecture SIMD ne prend le dessus que sur des problèmes à grain fin (exemple : certains traitement d'images).

Dans le cadre de cette thèse, les machines parallèles mises à notre disposition sont des grilles de calcul. C'est-à-dire un ensemble d'ordinateurs indépendants reliés entre elles par un réseau efficace. Nous sommes donc sur une architecture MIMD sans mémoire commune.

C.2 Évaluation d'un algorithme parallèle

Afin d'évaluer un algorithme ou de classifier le problème à résoudre il faut introduire quelques définitions.

C.2.1 Accélération

L'accélération quantifie le gain induit par l'utilisation de plusieurs processeurs. Cette mesure est définit le rapport entre le temps d'exécution du meilleur algorithme séquentiel et le temps d'exécution de l'algorithme parallèle :

$$\text{accélération(p)} = \frac{\text{temps d'exécution en séquentiel}}{\text{temps d'exécution avec p processeurs}}$$

L'accélération idéale est accélération(p) = p.

C.2.2 Efficacité

On peut déduire de l'accélération l'efficacité de l'algorithme parallèle, c'est-à-dire le rapport entre l'accélération effective et l'accélération idéale :

$$\text{efficacité(p)} = \frac{\text{temps d'exécution en séquentiel}}{\text{p} * \text{temps d'exécution avec p processeurs}}$$

C.2.3 Capacité d'évolution

La définition formelle de la capacité d'évolution est la suivante : soient deux problèmes P_1 et P_2 avec P_2 n fois plus gros que P_1 et $T(P_i, p)$ le temps d'exécution d'un problème P_i sur un

système avec p processeurs, la capacité d'évolution est défini par la formule suivante :

$$\text{capacité } d'évolution = \frac{T(P_1, n)}{T(P_2, p * n)}$$

La capacité d'évolution est idéale si elle reste égale à 1.

C.2.4 Granularité et parallélisme

Le niveau de parallélisme d'un algorithme est défini par sa granularité. Le grain le plus fin d'un algorithme fait s'exécuter de façon concurrente toute suite d'instructions inséparables. Un grain plus important fait travailler simultanément des blocs d'instructions plus longs. Un algorithme à gros grain - ou à grain grossier - parallélise une application en exécutant des processus complets.

L'algorithme originale des *marching cubes* à un grain très fin, en effet chaque cellule peut-être traitée indépendamment des autres cellules. La granularité des algorithmes de simplification est beaucoup plus discutable et dépend du mécanisme de simplification choisi ainsi que du critère d'erreur.

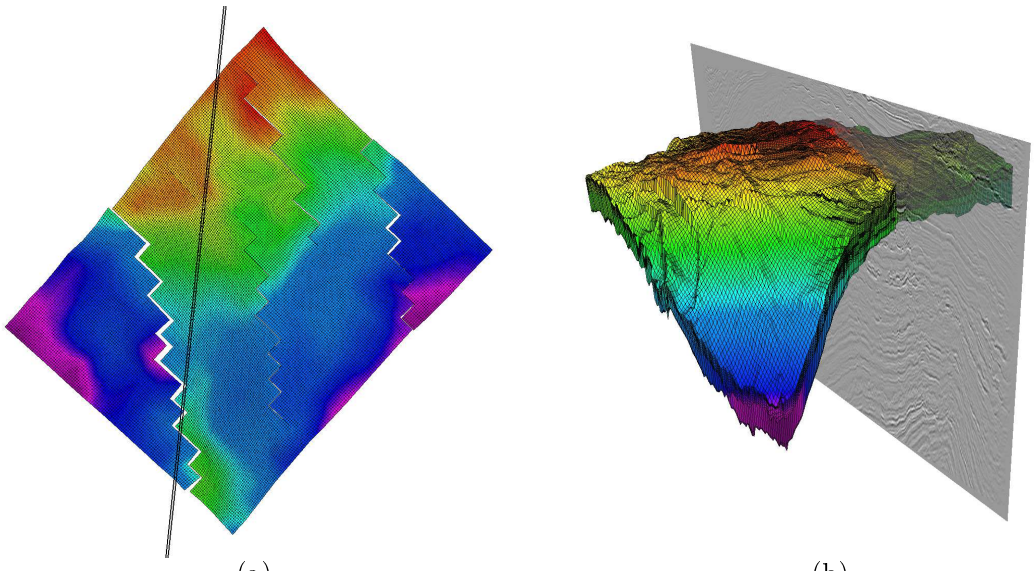
C.2.5 Répartition de charges

En calcul parallèle, la répartition de charges consiste à répartir le volume de travail sur les machines disponibles. Dans le cadre des *marching cubes* la charge d'un processeur est fortement liée au nombre de cellules traitées en particulier celles intersectant l'isosurface.

Annexe D

Rendu des grilles réservoirs sur des coupes sismiques quelconques

L'analyse du sous-sol par les géophysiciens et les géologues a permis la construction du modèle structural. Ce modèle est utilisé par l'ingénieur réservoir pour la construction de la grille réservoir. Cette grille sert de support aux simulations qui, associées à des données réelles[3], permettent de valider le modèle et d'optimiser le placement des nouveaux puits de forage. Un exemple de grille réservoir est visible sur la figure D.1.



(a) (b)

FIG. *D.1: Rendu d'une grille réservoir (a) Vue de dessus, la coupe sismique est représentée par un trait noire. (b) Vue en trois dimensions de la grille et de la coupe sismique.*

La construction de la grille est fortement contraint par les besoins du simulateur utilisé. Le modèle structural est alors simplifié afin de faciliter la construction de la grille. De nombreux

[3]Ces données sont issues de puits déjà forés sur la zone d'intérêt.

contrôles sont alors nécessaires pour valider la grille basée sur ce modèle simplifié voir incomplet. Un de ces tests est le contrôle visuel sur la sismique, ce qui implique de calculer l'intersection entre une coupe sismique et la grille (voir figure D.1). La figure D.2 illustre le rendu d'une grille réservoir sur une coupe sismique.

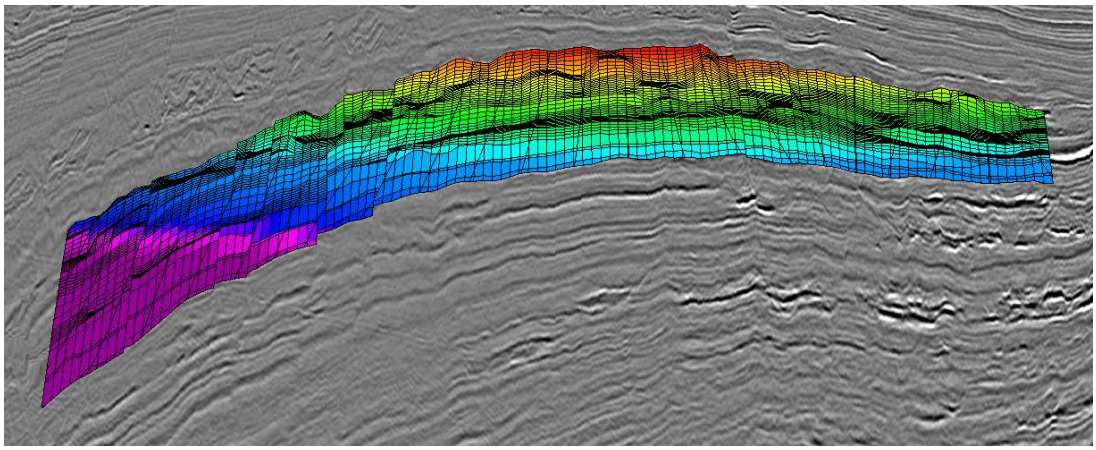


FIG. *D.2: Rendu d'une grille réservoir sur une coupe sismique*

L'enjeu principal dans le rendu est l'interactivité dans le changement de coupe sismique. En effet, un balayage dans la sismique par l'ingénieur réservoir et le géophysicien leur permet une meilleur compréhension du volume. L'algorithme proposé pour calculer l'intersection de la grille avec la coupe sismique devra donc être très efficace car les grilles réservoirs contiennent plusieurs millions de mailles.

D.1 Formulation du problème

La coupe d'une grille réservoir par un plan peut-être exprimée de manière plus générale : il s'agit de calculer l'isosurface en 0 de la fonction définie par le plan sur les cellules de la grille. Cette technique est souvent utilisée pour faire du rendu volumique, un ensemble de surfaces sont extraites entre deux valeurs scalaires bornant la profondeur z dans le repère des coordonnées de vue.

Pour illustrer cette méthode de rendu, on peut citer les travaux de Bruno Levy et al [56] qui ont proposé la structure de donnée CIEL (Circular Incident Edge List) qui permet une recherche efficace des cellules intersectées par les différents plans, ainsi qu'une méthode de construction des polygones pour du rendu volumique (voir figure D.3). Ce rendu est fait en affichant de nombreux plans perpendiculaires à la direction de la caméra.

Dans le cas des grilles réservoirs, les cellules sont toujours des hexaèdres. La méthode d'extraction d'isosurfaces la plus couramment utilisée sur ce type de grille est l'algorithme des *marching*

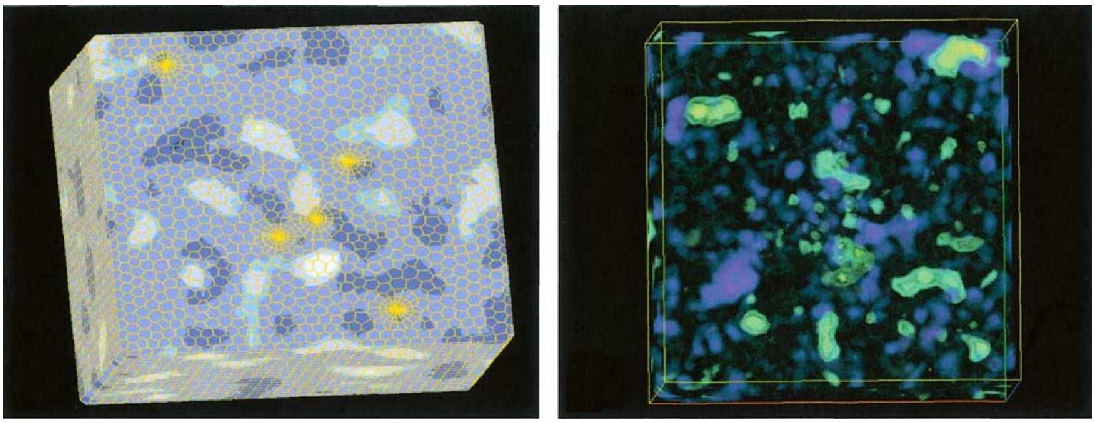


FIG. *D.3: Rendu volumique de grilles grâce à la méthode CIEL. Figure issue de [56]*

cubes que nous avons détaillé dans le chapitre 2.

D.2 Calcul de l'intersection entre la grille réservoir et la coupe

Les grilles réservoirs sont composées de millions de mailles hexaèdriques, nous utilisons une structure de partionnement de l'espace qui nous permet de limiter le nombre de ces calculs d'intersection. Nous proposons aussi d'appliquer cette méthode aux coupes composées de plusieurs panneaux. Enfin nous étendrons notre méthode pour traiter les grilles réservoirs raffinées.

D.2.1 Partitionnement de l'espace : Octree

Comme dans [87] nous utilisons une octree afin d'accélérer la recherche des cellules qui intersectent le plan de coupe. La grille étant stockée en IJK, le découpage récursif est fait suivant ces trois directions .

Comme nous l'avons vu, l'extraction de l'intersection est réalisée en calculant l'isosurface en 0 de la fonction équation du plan. Une différence par rapport à une extraction classique dans une grille hexaédrique quelconque est que la fonction est définie par la géométrie de la grille. Nous n'avons alors pas besoin de stocker les valeurs maximales et minimale de l'équation du plan dans chaque noeud de l'octree. Les parties de l'octree sont élimié du parcourt récursif si les englobantes associés aux noeuds n'intersectent pas le plan.

D.2.2 Utilisation des tables indexée

Afin de calculer les polygones dessinés sur la coupe sismique, nous utilisons une approche similaire aux *marching cubes* : les polygones sont définis dans une table pré-calculé dont l'index est calculé en marquant les sommets comme étant supérieur ou non à l'isovaleur cherché. Nous utilisons la table décrite par Chernyaev dans [17].

D.2.3 Intersection avec des *random lines*

Dans les exemples précédents, nous nous sommes intéressés à la coupe de la grille réservoir par un plan. Nous avons donc supposé que la coupe sismique contenait entièrement la coupe de la grille réservoir. En pratique, cette supposition est souvent vrai car le réservoir représente une petite partie de la zone du sous-sol étudiée et donc du cube sismique. Il peut cependant arriver que la coupe ne traverse pas le plan, c'est souvent la cas quand l'ingénieur réservoir veut étudier la grille non plus sur une coupe sismique simple mais sur une *random line* qui est une succesion de panneau définis à la main sur une carte (voir figure D.4).

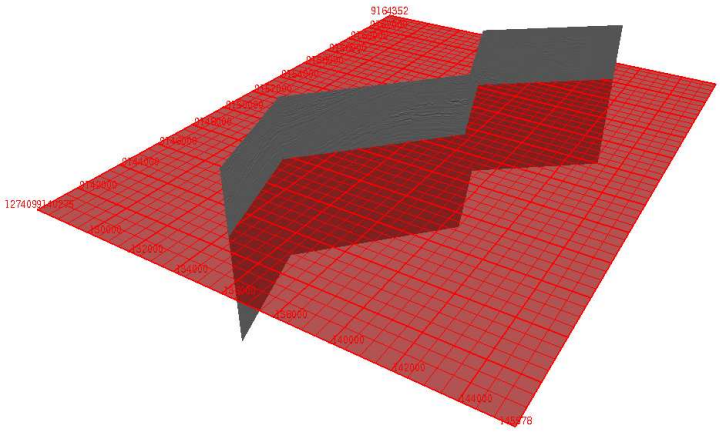


FIG. *D.4: Exemple de random line*

Dans ce cas le calcul est fait sur chaque panneau indépendamment. Le panneau n'est plus seulement défini pas un mais par un rectangle qui le limite dans son plan. Il faut alors extraire les polygones des cellules intersectant le plan et étant partiellement ou entièrement contenu dans le rectangle. Les polygones sont ensuite coupés en utilisant l'algoritme de Sutherland-Hodgman (voir figure D.5).

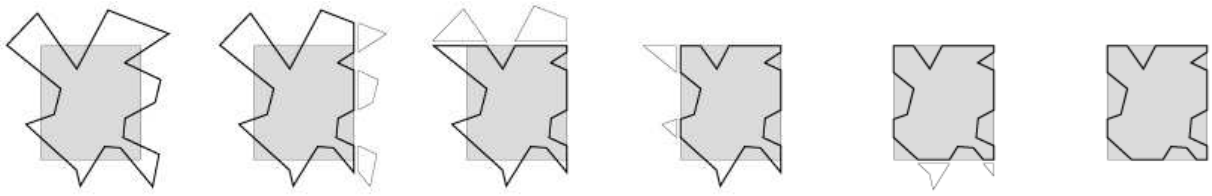
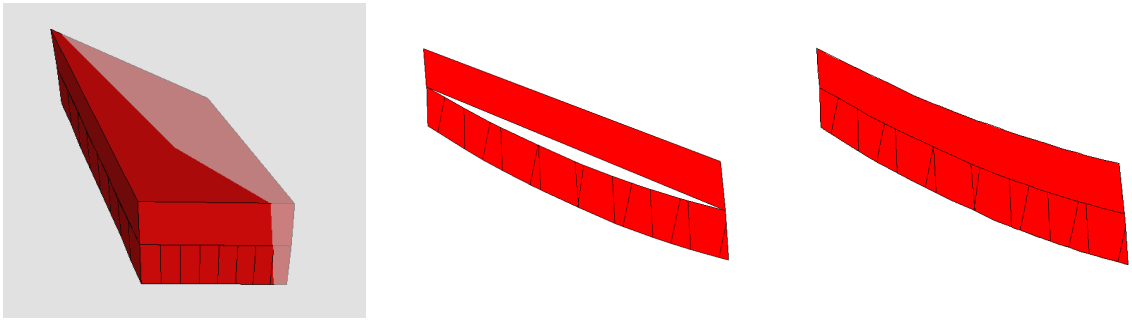


FIG. *D.5: Découpe des polygones en utilisant l'algorithme de Sutherland-Hodgman*

D.2.4 Traitement des grilles raffinées

Certains calculs sur les grilles nécessitent de raffiner les cellules. Ce raffinage peut être local, à proximité de certains évènements géologiques. Le raffinage est exclusivement IJ et les nouveaux

points de la cellule sont calculées par interpolation bilinéaire. La fait que ce raffinage soit partiel entraîne l'apparition de *cracks* quand une cellule raffinée est voisine en k avec une cellule non raffinée (voir figure D.6).



(a) (b) (c)

FIG. *D.6: Apparition de cracks de la cas de grilles raffinées. (a) Une cellule raffinée est située sous une cellule non raffinée. Elles sont intersectées par une coupe sismique (en transparence). (b) Apparition d'un crack dans le rendu. (c) Le raffinement du polygone de la cellule non raffinée évite l'apparition du crack.*

Afin de résoudre se problème sans alourdir les calculs nous proposons une méthode qui raffine les polygones calculés sur les cellules non raffinées évitant ainsi les cracks dans le rendu (voir figure D.6 (c)).

D.3 Conclusion

Nous avons profité des connaissance acquises pour la reconstruction de corps dans les attributs sismiques (décrites dans le chapitre 2) pour proposer une méthode efficace d'intersection entre une coupe sismique quelconque et une grille réservoir. Cet algorithme permet une navigation interactive : l'intersection entre une coupe composée de 6 panneaux et une grille composée de 3 millions de maille prend environ 30 ms, l'intersection d'un panneaux de complexité équivalente avec une grille de 30 millions de mailles prend 250 ms (voir figure D.7). Ces temps sont souvent inférieurs à l'accès à la donnée sismique rendant la latence dûe au calcul d'intersection inexistante pour l'utilisateur.

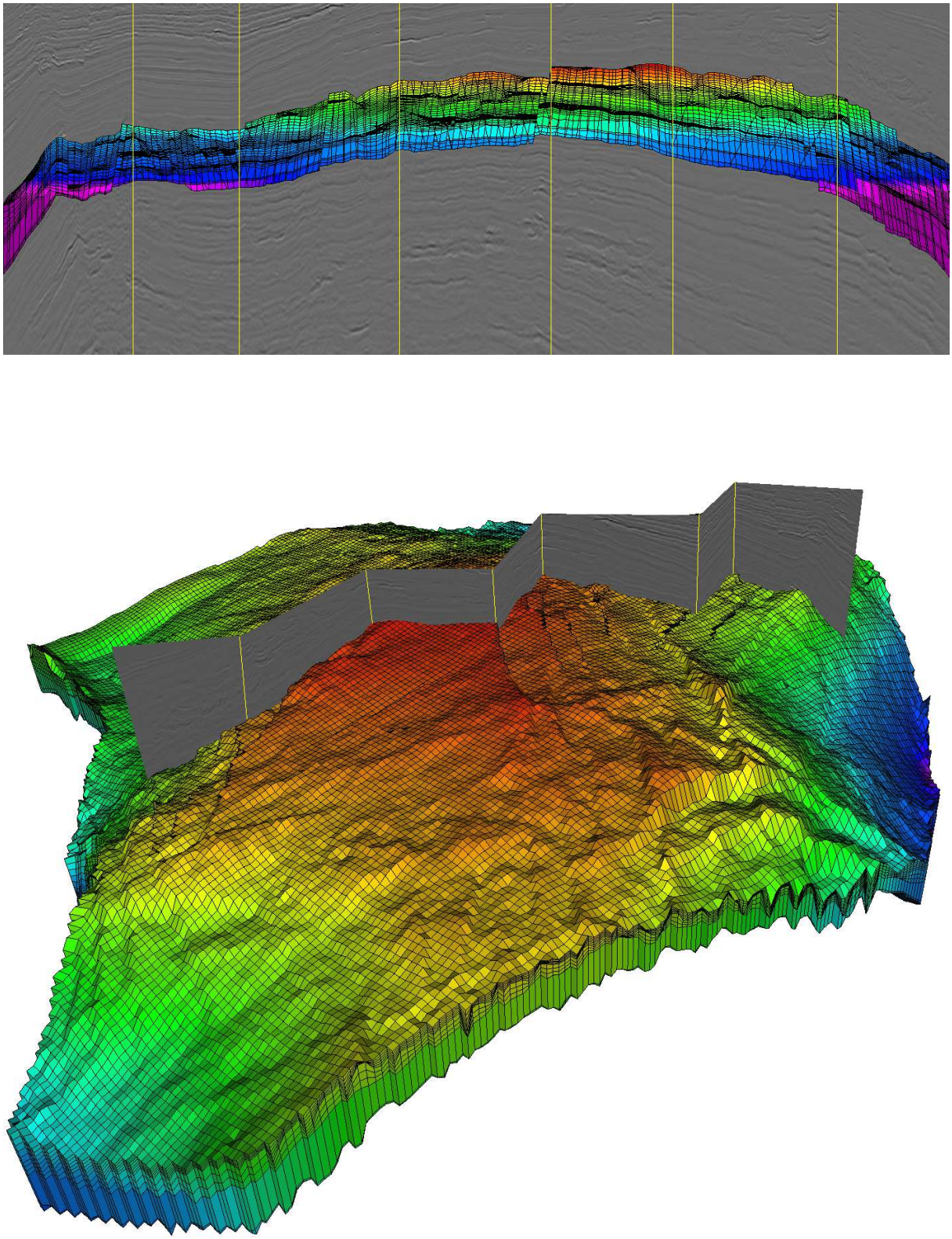


FIG. *D.7: Exemple de "random line"*

Bibliographie

[1] CGAL (Computational Geometry Algorithms Library). http ://www.cgal.org.

[2] Procédé et programme de propagation d'un marqueur sismique dans un ensemble de traces sismiques. brevet français numéro 2 869 693, 1998.

[3] Nina Amenta, Marshall Bern, and Manolis Kamvysselis. A new Voronoi-based surface reconstruction algorithm. *Computer Graphics*, 32(Annual Conference Series) :415–421, 1998.

[4] Nina Amenta and Marshall W. Bern. Surface reconstruction by voronoi filtering. In *Symposium on Computational Geometry*, pages 39–48, 1998.

[5] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust, unions of balls, and the medial axis transform. *Computational Geometry*, 19(2-3) :127–153, 2001.

[6] Dominique Attali, David Cohen-Steiner, and Herbert Edelsbrunner. Extraction and simplification of iso-surfaces in tandem. In *SGP '05 : Proceedings of the third Eurographics symposium on Geometry processing*, page 139, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.

[7] Chandrajit L. Bajaj, Edward J. Coyle, and Kwun-Nan Lin. Arbitrary topology shape reconstruction from planar cross sections. *Graph. Models Image Process.*, 58(6) :524–543, 1996.

[8] D. Bartz, R. Grosso, T. Ertl, and W. Straer. Parallel construction and isosurface extraction of recursive tree structures, 1998.

[9] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *SIGGRAPH '00 : Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 417–424, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[10] Jean-Daniel Boissonnat. Shape reconstruction from planar cross sections. *Comput. Vision Graph. Image Process.*, 44(1) :1–29, 1988.

[11] Jean-Daniel Boissonnat and Pooran Memari. Shape reconstruction from unorganized cross-sections. In *SGP '07 : Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 89–98, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[12] Mario Botsch, Mark Pauly, Leif Kobbelt, Pierre Alliez, Bruno Lévy, Stephan Bischoff, and

Christian Rössl. Geometric modeling based on polygonal meshes. In *SIGGRAPH '07 : ACM SIGGRAPH 2007 courses*, page 1, New York, NY, USA, 2007. ACM.

[13] Tamy Boubekeur. Surface reconstruction with subdivision. Master's thesis, Université de Bordeaux (Master Recherche), June 2004.

[14] E. Catmull and J. Clark. Recursively generated b-spline surfaces on arbitrary topological meshes. *Seminal graphics : poineering efforts that shaped the field*, pages 183–188, 1978.

[15] Guillaume Caumon. *Représentation, visualisation et modification de modèles volumiques pour les Géosciences.* PhD thesis, Institut National Polytechniques de Lorraine, 2003.

[16] Tony F. Chan and Jianhong Shen. Mathematical models for local nontexture inpaintings. *SIAM J. Appl. Math*, 62 :1019–1043, 2001.

[17] Evgeni V. Chernyaev. Marching cubes 33 : Construction of topologically correct isosurfaces, 1995.

[18] Yi-Jen Chiang, Ricardo Farias, Claudio Silva, and Bin Wei. A unified infrastructure for parallel out-of-core isosurface extraction and volume rendering of unstructured grids. In *Proceedings of IEEE parallel and large data visualization and graphics*, pages 59–66, 2001.

[19] David Cohen-Steiner and Tran Kai Frank Da. A greedy delaunay-based surface reconstruction algorithm. *The Visual Computer*, 20 :4–16, 2004.

[20] Tran Kai Frank Da. *L'interpolation de formes.* PhD thesis, Université de Nice, Sophia-Antipolis, janvier 2002.

[21] T. Dey, J. Giesen, and J. Hudson. A delaunay based shape reconstruction from large data, 2001.

[22] T. Dey and S. Goswami. Tight cocone : A water-tight surface reconstructor, 2003.

[23] Tamal K. Dey, Herbert Edelsbrunner, Sumanta Guha, and Dmitry V. Nekhayev. Topology preserving edge contraction. *Publ. Inst. Math. (Beograd) (N.S*, 66 :23–45, 1999.

[24] D. Doo. A subdivision algorithm for smoothing down irregularly shaped polyhedrons. *Proceedings on Interactive Techniques in Computer Aided Design*, pages 157–165, 1978.

[25] Guilhem Dupuy, Bruno Jobard, Sebastion Guillon, Noomame Keskes, and Dimitri Komatitsch. Isosurface extraction and interpretation on very large datasets in geophysics. In *SPM '08 : Proceedings of the 2008 ACM symposium on Solid and physical modeling*, pages 221–229, New York, NY, USA, 2008. ACM.

[26] Nira Dyn, David Levine, and John A. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *ACM Trans. Graph.*, 9(2) :160–169, 1990.

[27] M. J. Dürst. Letters : Additional reference to "marching cubes". In ACM, editor, *ACM Computer Graphics*, volume 22, pages 72–73, 1988.

[28] Herbert Edelsbrunner and Ernst P. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13(1) :43–72, 1994.

[29] H. Fuchs, H. Fuchs, Z. M. Kedem, S. P. Uselton, and S. P. Uselton. Optimal surface reconstruction from planar contours. *Commun. ACM*, 20(10) :693–702, 1977.

[30] Jonathan Gallon. Filtrage de volumes sismiques 3D par propagation de surfaces 4D - application à la sismique avant sommation. Master's thesis, Universite de Pau et des Pays de l'Adour, 2007.

[31] Michael Garland. *Quadric-Based Polygonal Surface Simplification.* PhD thesis, School of Computer Science Carnegie Mellon University, May 1999.

[32] Michael Garland and Paul Heckbert. Fast polygonal approximation of terrains and height fields. Technical Report CMU-CS-95-181, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, September 1995.

[33] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. *Computer Graphics*, 31(Annual Conference Series) :209–216, 1997.

[34] Michael Garland and Paul S. Heckbert. Simplifying surfaces with color and texture using quadric error metrics. In *VIS '98 : Proceedings of the conference on Visualization '98*, pages 263–269, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[35] Paul-Louis George and Houman Borouchaki. *Triangulation de Delaunay et maillage, application aux éléments finis.* Hermes, 1997.

[36] T. Gerstner and M. Rumpf. Multiresolutional parallel isosurface extraction based on tetrahedral bisection. In Springer, editor, *Volume graphics*, pages 267–278, 2000.

[37] Jeff Goldsmith and Allan S. (Bud) Jacobson. Marching cubes in cylindrical and spherical coordinates. *Journal of graphics tools*, 1(1) :21–31, 1996.

[38] Sébastien Guillon. *Filtrage adaptatif non linéaire appliqué au rehaussement et à la segmentation d'images.* PhD thesis, Université de Bordeaux I, novembre 1997.

[39] C Hansen and P Hinker. Massively parallel isosurface extraction. In *Proceedings of visualization'92*, pages 77–83, 1992.

[40] Taosong He, L. Hong, A. Kaufman, A. Varshney, , and S. Wang. Voxel-based object simplification. In *Proc. Visualization '95.* IEEE Comput. Soc. Press, 1995. http ://www.cs.sunysb.edu/ taosong/taosong-papers.html.

[41] Hugues Hoppe. View-dependent refinement of progressive meshes. In *SIGGRAPH '97 : Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 189–198, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[42] Hugues Hoppe. New quadric metric for simplifying meshes with appearance attributes. In *VISUALIZATION '99 : Proceedings of the 10th IEEE Visualization 1999 Conference (VIS '99)*, Washington, DC, USA, 1999. IEEE Computer Society.

[43] Hugues Hoppe, Tony DeRose, Tom Duchamp, Mark Halstead, Hubert Jin, John McDonald, Jean Schweitzer, and Werner Stuetzle. Piecewise smooth surface reconstruction. *Computer Graphics*, 28(Annual Conference Series) :295–302, 1994.

[44] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics*, 26(2) :71–78, 1992.

[45] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In *SIGGRAPH '93 Proc.*, pages 19–26, August 1993. http ://research.microsoft.com/ hoppe/.

[46] Kai Hormann, Bruno Levy, and Alla sheffer. Mesh parameterization : Theory and practice. In *SIGGRAPH Courses Notes*, 2007.

[47] Jan Hrádek. Methods of surface reconstruction from scattered data. Technical Report DCSE/TR-2003-02, University of West Bohemia in Pilsen, march 2003.

[48] M. Isenburg and P. Lindstrom. Streaming meshes. In *Proceedings of Visualization'05*, pages 231–238, 2005.

[49] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes : Active contour models. *International Journal of Computer Vision*, V1(4) :321–331, January 1988.

[50] Michael Kazhdan, Allison Klein, Ketan Dalal, and Hugues Hoppe. Unconstrained isosurface extraction on arbitrary octrees. In *SGP '07 : Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 125–133, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[51] Reinhard Klein, Andreas Schilling, Wolfgang Straßer, and Universität Tübingen. Reconstruction and simplification of surfaces from contours. In *In Pacific Graphics*, pages 429–443, 1999.

[52] L. Kobbelt. Interpolatory subdivision on open quadrilateral nets with arbitrary topology. In *Computer Graphics Forum (Proc. EUROGRAPHICS '96), 15(3)*, pages 409–420, 1996.

[53] Leif Kobbelt. $\sqrt{3}$ subdivision. pages 103–112, 2000.

[54] Emmanuel Labrunye. *Extraction automatique d'information géologique à partir d'images sismques tridimensionnelles.* PhD thesis, Institut National Polytechnique de Lorraine, octobre 2004.

[55] Bruno Levy. Geometrie numerique. *Habilitation a Diriger les Recherches*, fevrier 2008.

[56] Bruno Lévy, Guillaume Caumon, Stéphane Conreaux, and Xavier Cavin. Circular incident edge lists : a data structure for rendering complex unstructured grids. In T. Ertl, K. Joy, and A. Varshney, editors, *IEEE Visualization*, October 2001.

[57] Thomas Lewiner, Helio Lopes, Antonio Wilson Vieira, and Geovan Tavares. Efficient implementation of Marching Cubes' cases with topological guarantees. *Journal of Graphics Tools*, 8(2) :1–15, 2003.

[58] J. Li and P. Agathoklis. A case study of isosurface generation in 3d visualization. In *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, volume 2, pages 622–625, May 1993.

[59] J. Li and P. Agathoklis. An efficiency enhanced isosurface generation algorithm for volume visualization. In *The Visual Computer*, volume 13, 1997.

[60] Peter Lindstrom. Out-of-core simplification of large polygonal models. In Kurt Akeley, editor, *Siggraph 2000, Computer Graphics Proceedings*, pages 259–262. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[61] L. Liu, C. Bajaj, J. O. Deasy, D. A. Low, and T. Ju. Surface reconstruction from non-parallel curve networks. In *COMPUTER GRAPHICS FORUM*, volume 27, pages 155–163. Blackwell Publishing Ltd, 2008.

[62] Charles Teorell Loop. *Smooth subdivision surfaces based on triangles.* PhD thesis, University of Utah, 1987.

[63] A. Lopes and K. Brodlie. Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing, 2002.

[64] William E. Lorensen and Harvey E. Cline. Marching cubes : A high resolution 3d surface construction algorithm. In *SIGGRAPH '87 : Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169, New York, NY, USA, 1987. ACM Press.

[65] David Luebke. A survey of polygonal simplification algorithms. Technical report, Chapel Hill, NC, USA, 1997.

[66] David Meyers, Shelley Skinner, and Kenneth Sloan. Surfaces from contours. *ACM Trans. Graph.*, 11(3) :228–258, 1992.

[67] Arnau Michel and Xavier Emery. *Estimation et interpolation spatiale méthodes déterministes et géostatistiques.* 2000.

[68] S Miguet and J-M Nicod. A load-balanced parallel implementation of the marching-cube algorithm. In *Proceedings of high performance computing symposium'95*, pages 229–239, 1995.

[69] James V. Miller, David E. Breen, William E. Lorensen, Robert M. O'Bara, and Michael J. Wozny. Geometrically deformed models : a method for extracting closed geometric models form volume data. *SIGGRAPH Comput. Graph.*, 25(4) :217–226, 1991.

[70] H. Muller and M. Stark. Adaptive generation of surfaces in volume data, 1993.

[71] B. K. Natarajan. On generating topologically consistent isosurfaces from uniform samples. *Vis. Comput.*, 11(1) :52–62, 1994.

[72] Timothy S. Newman and Hong Yi. A survey of the marching cubes algorithm. *Computers & Graphics*, 30(5) :854–879, October 2006.

[73] Jean-Marc Nicod. *Extraction de Surfaces en Imagerie Médicale : Approches Parallèles.* PhD thesis, Ecole Normale Supérieur de Lyon, janvier 1997.

[74] M. Ohlberger, M. Rumpf, and Abstract Zusammenfassung. Hierarchical and adaptive visualization on nested grids. *Computing*, 59 :4–269, 1997.

[75] Manuel M. Oliveira, Brian Bowen, Richard Mckenna, and Yu sung Chang. Fast digital image inpainting. In *Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP 2001*, pages 261–266, 2001.

[76] Steve Oudot, Laurent Rineau, and Mariette Yvinec. Meshing volumes bounded by smooth surfaces. In *Proc. 14th International Meshing Roundtable*, pages 203–219, 2005.

[77] Frederik Pivot, Guilhem Dupuy, Sebastien Guillon, and Jean Noel Ferry. Complex volumic seismic interpretation by new geobodies extraction strategy. In *London 2007, 69th EAGE Conference & Exhibition incorporating SPE EUROPEC 2007*, 2007.

[78] Jarek Rossignal and Paul Borrel. *Multi-resolution 3D approximations for rendering complex scenes.* 1993.

[79] Carlos E. Scheidegger, Shachar Fleishman, and Cláudio T. Silva. Triangulating point set surfaces with bounded error. In *SGP '05 : Proceedings of the third Eurographics symposium on Geometry processing*, page 63, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.

[80] William J. Schroeder, Jonathan A. Zarge, and William E. Lorensen. Decimation of triangle meshes. *Computer Graphics (SIGGRAPH '92 Proc.)*, 26(2) :65–70, July 1992.

[81] Andrei Sharf, Thomas Lewiner, Ariel Shamir, Leif Kobbelt, and Daniel Cohen-Or. Competing fronts for coarse–to–fine surface reconstruction. In *Eurographics*, pages 389–398, Vienna, september 2006.

[82] Raj Shekhar, Elias Fayyad, Roni Yagel, and J. Fredrick Cornhill. Octree-based decimation of marching cubes surfaces. pages 335–342, 1996.

[83] Peter Shirley and Allan Tuchman. A polygonal approximation to direct scalar volume rendering. In *Procedings San Diego Workshop on Volume Visualization, Computer Graphics*, volume 24, pages 63–70, 1990.

[84] R. Sibson. *Interpreting Multivariate Data*, chapter A brief description of natural neighboring interpolation. John Wiley and Sons, 1981.

[85] Allen van Gelder and Jane Wilhelms. Topological considerations in isosurface generation. *ACM Trans. Graph.*, 13(4) :337–375, 1994.

[86] David Weinstein. Scanline surfacing : building separating surfaces from planar contours. In *VIS '00 : Proceedings of the conference on Visualization '00*, pages 283–289, Los Alamitos, CA, USA, 2000. IEEE Computer Society Press.

[87] Jane Wilhelms and Allen Van Gelder. Octrees for faster isosurface generation. *ACM Trans. Graph.*, 11(3) :201–227, 1992.

[88] J. Wu and L. Kobbelt. A stream algorithm for the decimation of massives meshes, 2003.

[89] Denis Zorin. Subdivision zoo. In *SIGGRAPH 2000 : Subdivision for modeling and animation*, pages 65–102, 2000.

[90] Denis Zorin, Peter Schröder, and Wim Sweldens. Interpolating subdivision for meshes with arbitrary topology. *Computer Graphics*, 30(Annual Conference Series) :189–192, 1996.

[91] Denis Zorin and Peter Schröder. Subdivision for modeling and animation. In *SIGGRAPH Course Notes*, 2000.

Index

α-shape, 20

acquisition sismique, 2
attribut sismique, 3, 10, 54
    iso5, 54

B-spline, 102

calcul parallèle
    accélération, 108
    capacité d'évolution, 108
    efficacité, 108
    environemments, 107
    granularité, 109
    répartition de charges, 109
corps, 5
courbe de Bézier, 101

faille, 3
    décrochement, 4
    inverse, 4
    normale, 4
    principale, 15
    réseau, 15
    secondaire, 15

grille réservoir, 111

horizon, 3
    outils d'interprétation, 5

interpolation spatiale
    estimateur, 27
    stratégie de remplissage, 32
interpolation spatiales, 25
interprétation sismique, 5
isosurface, 55, 112
isovaleur, 56

krigeage, 28, 42
    ordinaire, 29
    simple, 29

marching cubes, 59
    2-variété, 61
    ambiguïté interne, 64
    ambiguïté sur les faces, 63
    justesse et consistance, 63
    méthodologie, 59
    simplification, 65
    types de grilles, 60
modèle géologique, 1
modèle structural, 13
méthode barycentrique, 28, 42

notion de visibilité, 35
NURBS, 102

octree, 65, 113

paramétrisation, 99
pointé
    corps, 10
    faille, 8, 14
polynômes de Lagrange, 99
propagateur d'horizon, 7

qualité d'un maillage, 91

régression

globale, 100
locale, 30, 45, 46
réseau de failles, 14

simplification de surface, 67
dépendance au point de vue, 67
mécanisme de simplification, 68
métrique d'erreur, 72
préservation de la topologie, 67
sismique, 2
surface de subdivision, 21
définition, 103
schéma approximant, 104
schéma interpolant, 105
surface polygonale, 15
surface spline, 101

tandem algorithm, 75
time lag, 77, 85, 91
triangulation, 17
contrainte, 18
de Delaunay, 17
de Delaunay contrainte, 18, 19

variété, 61
voisinage, 27
partiel, 36, 38

Résumé

La *géomodélisation* du sous-sol est considérée comme stratégique par les compagnies pétrolières. Chacune a en effet pour objectif de construire les meilleurs modèles géologiques sur lesquels seront évalués les réserves de gaz et d'huile des territoires étudiés, ainsi que le placement optimal des puits de forage qui exploiteront ces réserves. Ces modèles sont bâtis sur l'interprétation de la sismique par les géologues et les géophysiciens. Le processus de modélisation consiste à structurer le sous-sol en un ensemble d'*horizons* (strates géologiques), de *failles* (fractures de la roche) et de *corps* (amas de roches de même nature). Ces objets géologiques sont au sein de deux problématiques complémentaires abordées dans cette thèse chez Total : la *construction du modèle structural* à proprement parler constitué d'un réseau de failles et d'horizons mis en cohérence et la *constructions des corps* selon des attributs sismiques élaborés.

La construction du modèle structural part des données issues d'interprétations manuelles et/ou semi-automatiques ne décrivant que ponctuellement les objets géologiques : un ensemble de lignes brisées pour les failles et un nuage de points pour les horizons. Une première partie de nos travaux porte sur l'élaboration d'une méthodologie complète de construction de surfaces modélisant les failles et les horizons adaptée à la nature et la densité particulières des données initiales. Dans un premier temps les failles sont reconstruites indépendamment les unes des autres à l'aide d'une triangulation de Delaunay contrainte des données projetées dans leur plan de régression, puis raffinées par subdivision. Ensuite un réseau de failles global est créé et mis en cohérence par des opérations d'extrapolation et de clipping. Enfin, les trous dans les horizons sont bouchés par des méthodes d'interpolation spatiale étendus par nos soins à l'aide des "voisinages partiels" contraints par les failles, pour lesquelles nous montrons l'influence des stratégies de remplissage et leur robustesse au bruit.

Les corps sont construits par extraction d'isosurfaces sur des volumes d'attributs sismiques pouvant atteindre des tailles de plusieurs dizaines de giga-octets. Ces tailles de données nous ont imposées de développer un algorithme parallèle d'extraction d'isosurfaces simplifiées. Le volume est d'abord récursivement découpé pour former une structure hiérarchique de blocs voisins. Des isosurfaces partielles sont extraites de ces blocs par des processeurs indépendants exécutant une version étendue de l'algorithme *tandem*. Ainsi, les noeuds de calcul extraient, simplifient, s'échangent et assemblent les morceaux de surfaces ainsi obtenues jusqu'à l'obtention d'une isosurface simplifiée globale. Cette dernière est organisée et stockée par construction en un flux de composantes connexes qui permettra une interaction ultérieure de haut niveau.

Abstract

Geomodeling is considered by oil companies as a critical aspect of hydrocarbon prospection. Indeed building geological models of subsurface allow them to estimate location and capacity of oil reservoirs in studied territories. Modeling stage consists in studying seismic volume in order to highligt *horizons* (interface between two depositional events), *faults* (fracturation of rocks) and *bodies* (cluster of rocks with same physical properties). Those geological objects are within two problematics treated in this thesis : *building the structural model* containing horizons and faults, and *reconstruction of bodies* based on seismic attributes.

Structural modeling is based on interpreted data, point clouds or set of polylines, representing geological objects. First, each fault is built independently by triangulating projection of its set of polylines in its regression plane. Generated coarse mesh is then smoothed using subdivision schemes. A second step consists in extrapolating and clipping operations applied to fault surfaces in order to build a coherent fault network. During last step, point clouds representing horizon are interpolated in order to complete holes in horizon surfaces due to the noise surrounding the faults in seismic data The interpolation process takes faults into account thanks to our *partial neighboring*. We also show the effects on surface quality of the filling strategies used in the interpolation process.

Body reconstruction is done by isosurface extraction on seismic attributes. In order to deal with the heavy trend in size increase of volumetric datasets, we present in this thesis a parallel, bloc-wise extension of the tandem algorithm, which simplifies on the fly an isosurface being extracted. Our approach minimizes the overall memory consumption using an adequate bloc splitting and merging strategy and with the introduction of a *component dumping* mechanism that drastically reduces the amount of memory needed for particular datasets such as those encountered in geophysics. As soon as detected, surface components are migrated to the disk along with a meta-data index (oriented bounding box, volume, etc) that will allow further improved exploration scenarios (small components removal or particularly oriented components selection for instance).