Triangulations et quadriques

Pascal Desnogues

▶ **To cite this version:**

Pascal Desnogues. Triangulations et quadriques. Géométrie algorithmique [cs.CG]. Université Nice Sophia Antipolis, 1996. Français. NNT : . tel-00771335

HAL Id: tel-00771335
https://theses.hal.science/tel-00771335

Submitted on 8 Jan 2013

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

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

THÈSE

préparée à **l'INRIA Sophia Antipolis**

pour obtenir le titre de

Docteur en Sciences

de

L'UNIVERSITÉ DE NICE - SOPHIA ANTIPOLIS

spécialité

Mathématiques Appliquées

présentée par

Pascal DESNOGUÈS

TRIANGULATIONS et QUADRIQUES

Soutenue le 3 décembre 1996 devant le jury composé de :

Président :	M.	**Jean-Daniel**	**BOISSONNAT**
Rapporteurs :	M.	**Bernard**	**LACOLLE**
	M.	**Jean-Marie**	**MORVAN**
	M.	**Günter**	**ROTE**
Examinateurs :	M.	**André**	**CÉRÉZO**
	M.	**Olivier**	**DEVILLERS**
	M.	**Jean-Michel**	**MOREAU**

Remerciements

Je tiens à exprimer mes plus sincères remerciements

à Olivier DEVILLERS, mon directeur de thèse, qui a su soutenir mes recherches lorsque je commençais à désespérer d'aboutir à un résultat, et ne pas trop s'inquiéter lorsque je n'allais pas le voir.

à André CÉRÉZO, mon second directeur, officiel cette fois-ci, pour avoir lu mon manuscrit et ne pas m'avoir tenu rigueur du manque d'approche théorique qu'il a pu y découvrir.

à Jean-Daniel BOISSONNAT qui m'a fait l'honneur d'accepter la présidence de mon jury de thèse, et qui, surtout, m'a chaleureusement accueilli au sein du projet PRISME.

à Bernard LACOLLE, Jean-Marie MORVAN et Günter ROTE qui ont accepté d'être rapporteurs de ce mémoire, et cela malgré, pour certains, une méconnaissance partielle de la langue ou des thèmes abordés.

à Jean-Michel MOREAU pour m'avoir encouragé dans mon travail, et inspiré pour la rédaction de mon état de l'art ; un grand merci également pour avoir fait partie de mon jury.

à Monique, qui a su rendre la vie de bureau agréable, voire enjouée, et qui sait si bien donner un avis toujours critique (mais juste...) sur la rédaction des documents que tous s'attachent à lui faire relire (je n'ai pas échapppé à la règle...).

à Luc, Franck et Sylvain, tous partis, momentanément (?), sous d'autres cieux après m'avoir précédé dans cette tâche difficile que représente la fin d'une thèse ; leur amitié ainsi que leurs conseils m'ont été particulièrement précieux.

à Stéphane qui, lui, va demeurer pratiquement orphelin-thésard dans le projet : courage, et je compte sur toi pour représenter brillamment les couleurs du FC INRIA.

à Jean-Pierre et son formidable outil de dessin JPdraw sans lequel les figures de cette thèse seraient bien tristes, mais aussi pour sa patience lors de mes fréquentes interruptions dans son travail et de sa promptitude à répondre à mes problèmes.

à Hervé, Mariette, Andreas et Francis et tous ceux qui ont fait partie du projet et grâce à qui ce travail est ce qu'il est.

à Agnès et Corinne, nos assistantes de direction, pour leur gentillesse et leur

constante bonne humeur.

à toutes les personnes de l' INRIA que je ne peux évidemment pas toutes citer, mais que j'ai fréquentées au cours de ces 3 années ; qu'elles sachent que je ne les oublie pas.

à Eric (courage pour la fin...), Sandro et Run qui ont suivi, de plus ou moins près, mon travail et qui m'ont encouragé, ne serait-ce que par leur amitié sincère.

à toute ma famille, bien sûr, et surtout à Beata, pour avoir toujours gardé confiance en moi, et avoir su supporter mes sautes d'humeur sans m'en vouloir.

Table des matières

Liste des figures

Liste des tableaux

Notations

$\mathbb{R}$	ensemble des nombres réels
$\mathcal{E}$	plan euclidien
Π	plan
Ω	enveloppe convexe
(x_i, y_i)	point du plan d'abscisse x_i et d'ordonnée y_i
a, b, c, d	points du plan euclidien
$s, \tilde{s}, s'$	points du plan euclidien
s_i	sommet d'indice i
(x_i, y_i, z_i)	point de l'espace d'abscisse x_i, d'ordonnée y_i et d'élévation z_i
A, B, C, D	sommets de l'espace, en général relevés sur une surface des sommets a, b, c, d
S	sommet de l'espace
S_i	sommet d'indice i
Δ	triangle
Δabc	triangle de sommets a, b, c
Λ	tétraèdre
$\Lambda ABCD$	tétraèdre de sommets A, B, C, D
$\bigcirc(\Delta)$	cercle circonscrit au triangle Δ
$\square abcd$	quadrilatère de sommets a, b, c, d
$\mathcal{Q}$	quadrilatère
$\mathcal{P}$	polygone
$\mathcal{T}$	triangulation
$[s_i s_j]$	segment d'extrémités s_i et s_j
$\mathrm{dist}(s_i, s_j)$	distance euclidienne séparant les points s_i et s_j
$\vec{n}$	vecteur normal
ς	arête
θ	angle
$\mathcal{A}$	ensemble d'arêtes
$\mathcal{S}$	ensemble de points
d	dimension d'un espace

n	nombre de sommets
m	nombre de triangles
f	une fonction, en général définie de $\mathbb{R}^2$ dans $\mathbb{R}$
$\mathcal{F}$	surface représentative de la fonction f $(z = f(x, y))$
$P_{\mathcal{T},f}$	approximation linéaire par morceau de la fonction f, définie par les triangles de la triangulation $\mathcal{T}$
$p_{\Delta,f}$	approximation linéaire de la fonction f, définie par le triangle Δ
$\mathcal{H}$	matrice hessienne
$\mathcal{M}$	matrice
$\binom{n}{p}$	coefficient binomial (n, p)
$\mathcal{PH}$	abréviation désignant le paraboloïde hyperbolique unité

Lexique

Ce lexique a pour but de donner des définitions simples et souvent peu formelles, à certaines des notions de base utilisées dans ce mémoire et peu familières au lecteur non averti. Ces définitions ne sont valables que dans le cadre de la dimension 2.

Maillages

maillage : ensemble composé d'un nombre fini d'éléments (triangles ou quadrilatères) qui recouvrent un domaine donné.

maillage structuré : maillage dans lequel les voisins de tous les éléments sous connus de façon implicite.

maillage non structuré : les voisins des éléments sont donnés de manière explicite.

triangulation : maillage dont les éléments sont des triangles planaires.

triangulation conforme : triangulation dont l'intersection de 2 éléments est réduite soit à l'ensemble vide, soit à un sommet (des 2 éléments), soit à une arête.

triangulation contrainte triangulation d'un ensemble de sommets pour laquelle un ensemble d'arêtes est fixée a priori.

triangulation de Delaunay : unique triangulation telle que aucun élément de l'ensemble de sommets ne soit à l'intérieur du cercle circonscrit de n'importe quel triangle.

triangulation de Steiner : triangulation où il est autorisé d'ajouter des points à l'ensemble initial des sommets.

Sommets

degré d'un sommet : nombre d'arêtes incidentes au sommet, dans un graphe ou une triangulation.

relevé : point de coordonnées $(x, y, f(x, y))$ associé au point (x, y) et à la surface d'équation $z = f(x, y)$.

point de Steiner : point que l'on s'autorise à ajouter à l'ensemble initial des sommets afin d'améliorer les résultats produits par une triangulation.

visible : deux sommets d'un graphe (ou d'une triangulation) sont visibles si le segment qui les relierait n'est pas coupé par une arête du graphe (ou de la triangulation).

Arêtes

swap ou **échange de diagonales :** action de supprimer, si cela est possible, une arête ς et de la remplacer par l'autre diagonale du quadrilatère convexe de diagonale ς.

Divers

densité : nombre de triangles d'un maillage par unité de surface donnée.

enveloppe convexe : plus petit convexe contenant un ensemble de points donnés.

fonction harmonique : fonction f de classe C^2 sur un ouvert U de $\mathbb{R}^n$ à valeurs dans $\mathbb{C}$ telle que :

$$\Delta f = \frac{\partial^2 f}{\partial x_1^2} + \frac{\partial^2 f}{\partial x_2^2} + \ldots + \frac{\partial^2 f}{\partial x_n^2} = 0$$

Introduction

La géométrie algorithmique a pour objectifs de concevoir et d'analyser des algorithmes destinés à résoudre des problèmes de nature géométrique, mais aussi de proposer des structures de données adaptées à ces résolutions. Apparue autour des années 1975, elle s'est depuis considérablement développée, et constitue aujourd'hui un domaine de recherche scientifique à part entière ; elle trouve ses applications dans de nombreuses disciplines, parmi lesquelles la robotique, la vision par ordinateur, la conception assistée par ordinateurs ou encore l'informatique graphique.

Parmi les problèmes que la géométrie algorithmique cherche à résoudre, la triangulation d'un ensemble de points occupe une part intéressante et représente, sans doute, un aspect très important de ses domaines applicatifs. En effet, avoir uniquement pour données des points n'est pas satisfaisant ; savoir les relier afin de construire une partition de l'espace qu'ils occupent est beaucoup plus intéressant, puisque les liaisons topologiques ou géométriques alors définies permettront notamment de repérer rapidement un objet, de dessiner une surface contenant les points, de tracer un chemin entre 2 sommets fixés... Tout cela implique qu'en pratique, les triangulations trouvent leurs applications dans de nombreux domaines : modélisation des surfaces, représentation de terrains ou de structures géologiques, modèles stochastiques...

Un autre problème survient alors : parmi toutes les triangulations possibles (dont le nombre est très supérieur au nombre de sommets), il faut en choisir une qui sera considérée meilleure que les autres. Cette notion de "meilleure" est directement liée à la nature de la question à résoudre, et elle peut se traduire par un ou des critères d'optimalité, que l'on aura pris soin de bien définir.

Ainsi, un nombre d'articles réellement très important, mais aussi de livres ou de thèses, existe et ne traite que des algorithmes de triangulation. En géométrie algorithmique, la triangulation de DELAUNAY [Del34] occupe une place privilégiée car elle optimise certains critères géométriques et on sait la construire de manière efficace : le fait qu'elle permette de retrouver une notion de proximité entre ses sommets (car elle est duale du diagramme de Voronoï) explique son utilisation pour les algorithmes de reconstruction, et sa propriété de maximiser l'angle minimum de ses triangles la rend particulièrement attractive pour mailler un ensemble de points dans la méthode des éléments finis. Le début du premier chapitre sera donc consacré à son étude, ainsi qu'à ses propriétés et à ses algorithmes de construction. La suite

du chapitre, qui se veut être un état de l'art (non exhaustif) des recherches plutôt appliquées, abordera des méthodes directement liées aux triangulations et à deux domaines qui y ont fréquemment recours : la méthode des éléments finis et la représentation des surfaces. Nous verrons alors qu'il existe de nombreuses applications où la triangulation de Delaunay ne convient pas, et où l'on est obligé soit de construire une triangulation tout à fait originale, soit de procéder par modification de Delaunay. Des articles seront détaillés lorsqu'ils présentent un algorithme original ou une méthodologie générale.

La suite de ce manuscrit sera consacrée à l'étude d'un problème plus précis. Il a été prouvé que la triangulation de Delaunay était optimale, au sens des normes L_p, lorsqu'elle était utilisée pour définir l'approximation linéaire d'une certaine surface convexe, le paraboloïde de révolution. Il a alors semblé intéressant d'observer ce qui se produisait si la surface choisie n'était pas convexe ; le paraboloïde hyperbolique a donc été choisi, en tant que surface fortement non convexe, mais relativement simple. Même si le problème parait un peu artificiel, puisqu'en général on ne connait pas la surface à reconstruire (ce qui explique l'utilisation de Delaunay, en raison de ses bonnes propriétés de proximités), il peut être regardé comme un cas d'école qui devrait permettre de déterminer dans quelle mesure il est possible de faire mieux que la triangulation de Delaunay si la surface est connue, et si cela est facile à réaliser.

Le chapitre 2 présente donc les définitions des objets qui seront utilisés, ainsi que les critères d'optimalité choisis et la quadrique sur laquelle une grande partie des recherches sera consacrée, le paraboloïde hyperbolique.

Le chapitre 3 montrera de quelle manière une solution a été trouvée ; cette solution repose essentiellement sur des recherches expérimentales et s'appuie sur des observations formulées par l'intermédiaire à la fois de logiciels de calcul formel et de traceur de courbes. Elle ne représente cependant qu'une solution à un critère d'optimalité local.

Une généralisation ainsi qu'une formalisation seront proposées dans le chapitre 4 ; on pourra alors se rendre compte que la recherche s'applique à toutes les quadriques définies par des fonctions, incluant ainsi les quadriques convexes.

Il a paru essentiel de pouvoir juger la qualité produite par les solutions données. Le chapitre 5 contient un très grand nombre de tests numériques, qui permettent d'évaluer les algorithmes proposés et de quantifier l'écart qui existe entre la triangulation de Delaunay et les triangulations localement optimales.

Le chapitre 6 apporte des éléments de réponse à une question qui a semblé naturelle : les algorithmes présentés ne convergent pas vers une triangulation unique, autrement dit, dans la plupart des cas, les triangulations localement optimales ne sont pas globalement optimales. On s'est alors demandé s'il était possible d'évaluer cette différence, et donc s'il était possible de trouver, même d'une manière peu efficace (car il existe de très fortes présomptions que ce problème soit NP-dur), "la" triangulation optimale d'un ensemble de points.

Chapitre 1

Triangulations pratiques

Les triangulations sont un objet géométrique très fréquemment utilisé dans la mesure où elles permettent de partitionner de façon naturelle une région, donnée par un ensemble de points ou un polytope. De très nombreux algorithmes ou techniques ont ainsi été développés pour construire une triangulation, ou en améliorer une.

La géométrie algorithmique a donné naissance à une triangulation bien spécifique, la triangulation de Delaunay, qui possède des propriétés géométriques remarquables (elle optimise notamment un certain nombre de critères) et que l'on sait construire en temps tout à fait raisonnable. C'est, de plus, l'une des rares triangulations définies de manière unique que l'on sache construire.

Après avoir rappelé la définition, les propriétés et les diverses classes d'algorithmes de construction de la triangulation de Delaunay, nous verrons dans ce chapitre que l'utilisation des triangulations ne se limite pas seulement à celle de Delaunay. Nous explorerons deux des domaines généraux qui y ont fréquemment recours : la méthode des éléments finis et la représentation des surfaces. Nous verrons enfin qu'il est possible d'engendrer d'autres triangulations optimales que celle de Delaunay, afin d'optimiser d'autres critères.

Le but de cet état de l'art n'est surtout pas d'être complet : le domaine choisi est tellement vaste que chaque partie nécessiterait presque l'édition d'un livre entier. Notre volonté a ici été d'essayer de résumer les techniques les plus classiques d'utilisation et de génération de triangulations 2D. Cet état de l'art doit contribuer à donner une idée de ce qui existe et de ce qu'il est possible de faire ; il se veut ainsi être un document pratique, qui ne comporte presque aucun résultat théorique (pour tout ce qui concerne les résultats théoriques sur les triangulations optimales et les maillages, Bern et Eppstein ont déjà publié un excellent article de synthèse [BE92]). Nous détaillerons quelques articles, qui décrivent soit un algorithme original, soit une méthode très générale, surtout lorsqu'ils présentent un intérêt géométrique particulier. Enfin, les résultats ou méthodes qui ne servent pas directement aux triangulations (comme en Conception Assistée par Ordinateur, par exemple) seront très brièvement mentionnés.

De nombreux documents ou informations sont disponibles directement sur Internet, où les pages WWW consacrées à la géométrie algorithmique, et aux maillages en particulier, se développent rapidement. Voici donc une liste d'adresses qui sont susceptibles de contenir une grande partie des références manquantes dans cet état

de l'art :

- `http://www.cs.duke.edu/~jeffe/compgeom.html`
  page maintenue par J. ERICKSON qui contient tout sur la géométrie algorithmique en général (liens vers les chercheurs du domaine, vers les revues, bibliographies, logiciels,...).

- `http://www.geom.umn.edu/software/cglist/`
  pour une liste assez complète de programmes ou logiciels de géométrie algorithmique.

- `http://www.ce.cmu.edu/~sowen/mesh.html`
  pour une bibliographie réellement très complète sur les maillages et les éléments finis, ainsi que des pointeurs vers des publications récentes (pages maintenues par S. OWEN).

- `http://www.ics.uci.edu/~eppstein/meshgen.txt`
  pour obtenir la bibliographie de l'article de synthèse de M. BERN et D. EPPSTEIN sur les maillages et les triangulations optimales pour la méthode des éléments finis.

- `http://www-users.informatik.rwth-aachen.de/%7Eroberts/`
  page où R. SCHNEIDERS présente des liens commentés vers des adresses orientées maillages et éléments finis, ainsi qu'une liste des pages (par pays) des personnes intéressées par le domaine.

- `http://www.cs.cmu.edu/~ph/mesh.html`
  page de P. HECKBERT où l'on trouve, notamment, des liens sur le thème des maillages (pour les surfaces et les éléments finis).

- `http://www.cs.wisc.edu/~deboor/bib/bib.html`
  pour une bibliographie très complète (débutée par L. SCHUMAKER) sur les maillages de surfaces (et en particulier sur les splines).

1.1 Triangulations géométriques

Dans cette section vont être présentées les triangulations qui ne tiennent compte que de la nature géométrique des points donnés, à savoir leurs coordonnées planaires. La triangulation de Delaunay, qui est à ce titre une référence, va être définie et ses diverses propriétés seront rappelées. Puis, d'autres triangulations géométriques seront exhibées, ainsi que les différentes types d'algorithmes qui existent pour construire une triangulation.

1.1.1 Triangulation de Delaunay

Soit $\mathcal{S}$ un ensemble de n points dans le plan (certaines définitions peuvent se généraliser en dimension supérieure sans contraintes supplémentaires).

1.1.1.1 Diagramme de Voronoï

définition 1.1 *le* **polygone de Voronoï** *associé à un point s de $\mathcal{S}$, noté $PV(s)$ est l'ensemble des points du plan plus proches de s que de tout autre site de $\mathcal{S}$:*

$$PV(s) = \{\tilde{s} \in \mathcal{E}, \forall s' \in \mathcal{S}, \mathrm{dist}(\tilde{s}, s) \leqslant \mathrm{dist}(\tilde{s}, s')\},$$

où $\mathcal{E}$ est le plan euclidien, muni de la distance dist.

propriété 1.1 *le polygone de Voronoï d'un point s de $\mathcal{S}$ est non borné si et seulement si s n'est pas sur l'enveloppe convexe de $\mathcal{S}$.*

définition 1.2 *les polygones de Voronoï de tous les sites de $\mathcal{S}$ forment une partition du plan appelée* **diagramme de Voronoï** *de $\mathcal{S}$.*

Cette partition du plan définit une notion de voisinage topologique :

définition 1.3 *deux points s et s' de $\mathcal{S}$ sont* **voisins** *dans le diagramme de Voronoï si leurs polygones associés ont une arête commune.*

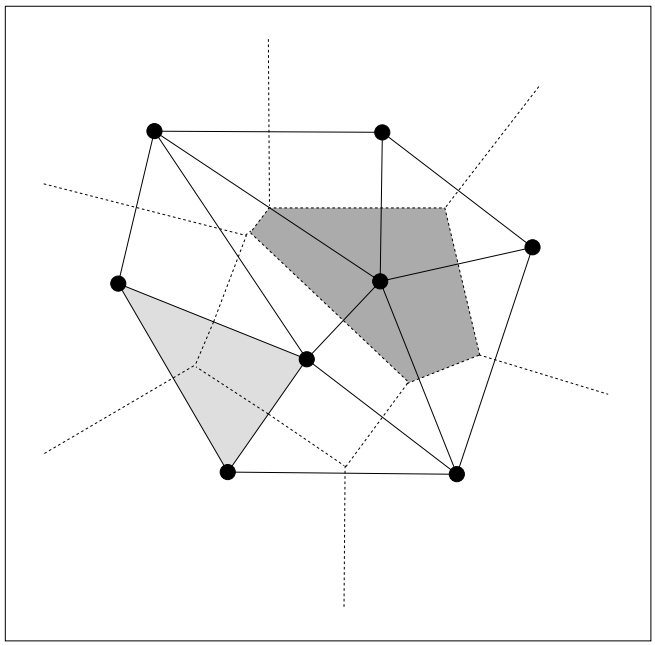
Pour une synthèse sur les diagrammes de Voronoï et leurs applications algorithmiques, le lecteur pourra se référer aux travaux de Aurenhammer [Aur91] et de Fortune [For92].

1.1.1.2 Triangulation de Delaunay

Plusieurs définitions pour la triangulation de Delaunay sont possibles ; l'une d'elles utilise la notion de diagramme de Voronoï :

définition 1.4 *sous l'hypothèse que $\mathcal{S}$ est non dégénéré (il n'existe pas 4 points de $\mathcal{S}$ cocycliques), la* **triangulation de Delaunay** *de $\mathcal{S}$ est le graphe dual du diagramme de Voronoï associé à $\mathcal{S}$: deux sites s et s' seront reliés par une arête si et seulement ils sont voisins dans le diagramme de Voronoï (voir figure 1.1).*

Figure 1.1 – la triangulation de Delaunay et le diagramme de Voronoï sont duaux.



1.1.1.3 Propriétés de la triangulation de Delaunay

La triangulation de Delaunay est la plus communément utilisée de toutes les triangulations, car on sait la construire en temps rapide (voir paragraphe 1.1.3) et parce qu'elle a de nombreuses propriétés géométriques, souvent très utiles. La plupart de ces propriétés sont décrites et prouvées dans le livre d'OKABE, BOOTS et SUGIHARA [OBS92].

Cercles circonscrits

Parmi toutes les propriétés, il y a celle dite du **cercle circonscrit**, qui permet de déduire directement un algorithme de construction de la triangulation.

propriété 1.2 *la triangulation de Delaunay de $\mathcal{S}$ est l'unique triangulation de $\mathcal{S}$ telle que le cercle circonscrit à chaque triangle ne contienne aucun site de $\mathcal{S}$ en son intérieur.*

D'AZEVEDO et SIMPSON prouvent que la triangulation de Delaunay optimise deux autres critères ayant un rapport avec les cercles circonscrits [DS89] :

propriété 1.3 *la triangulation de Delaunay de $\mathcal{S}$ minimise le plus grand cercle circonscrit et le plus grand plus petit cercle englobant des triangles (ce dernier cercle est différent du cercle circonscrit lorsque le triangle est obtus).*

Maximum de l'angle minimum

Une des propriétés les plus "classiques" de la triangulation de Delaunay est qu'elle maximise l'angle minimum de ses triangles ; l'utilité de cette propriété apparait

clairement dans les maillages pour la méthode des éléments finis où on a besoin que les triangles ne comportent pas d'angles trop petits. Ceci explique en partie la très forte utilisation de Delaunay comme triangulation de base pour engendrer des maillages triangulaires.

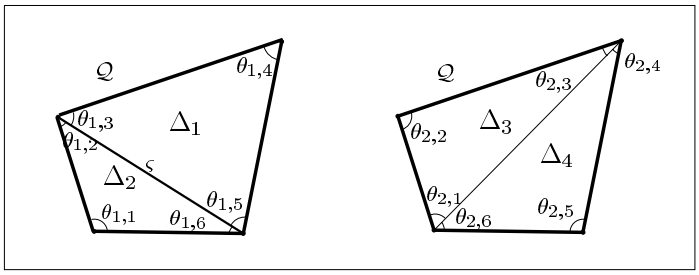
Critère du maximum de l'angle minimum local : soit ς une arête intérieure d'une triangulation $\mathcal{T}$ de l'enveloppe convexe d'un ensemble de points $\mathcal{S} = s_1, \dots, s_n$ et soit $\mathcal{Q}$ le quadrilatère formés par les triangles Δ_1 et Δ_2 de $\mathcal{T}$ ayant pour arête ς. On note $\theta_{1,i}, i = 1, \dots, 6$ les angles intérieurs de Δ_1 et Δ_2. Si $\mathcal{Q}$ est strictement convexe, on note Δ_3 et Δ_4, d'angles intérieurs $\theta_{2,i}, i = 1, \dots, 6$, les triangles obtenus en échangeant ς par l'autre diagonale de $\mathcal{Q}$.

L'arête ς satisfera le critère du maximum de l'angle minimum local si et seulement si une des propriétés suivantes est vérifiée :

- $\mathcal{Q}$ n'est pas convexe,
- $\mathcal{Q}$ dégénère en un triangle (si 3 points sont alignés),
- $\mathcal{Q}$ est strictement convexe, non dégénéré et on a la relation :

$$\min_{1 \leq i \leq 6} \theta_{1,i} \geqslant \min_{1 \leq i \leq 6} \theta_{2,i}.$$

Figure 1.2 – les angles intérieurs aux triangulations possibles d'un quadrilatère convexe $\mathcal{Q}$.



propriété 1.4 *Soit $\mathcal{S}$ un ensemble de n points distincts du plan, dont quatre quelconques ne sont pas cocycliques, et soit $\mathcal{T}$ une triangulation de l'enveloppe convexe de $\mathcal{S}$ recouvrant $\mathcal{S}$. Alors, toute arête intérieure de $\mathcal{T}$ qui satisfait le critère du maximum de l'angle minimum local est de Delaunay, et réciproquement.*

Équiangularité

Concernant la triangulation de Delaunay dans le plan, une des propriétés essentielles est qu'elle est globalement équiangulaire (en fait, SIBSON [Sib78] a d'abord montré qu'elle était localement équiangulaire, puis EDELSBRUNNER [Ede87] a prouvé que ce caractère local était en fait équivalent à l'équiangularité globale) : ceci signifie que de toutes les triangulations possibles, celle de Delaunay donne des triangles les plus équilatéraux possibles. En pratique, ce résultat est essentiel, surtout pour l'utilisation de la triangulation de Delaunay dans la méthode des éléments finis, où l'on a besoin d'une triangulation à triangles les moins plats possibles (sinon, on se

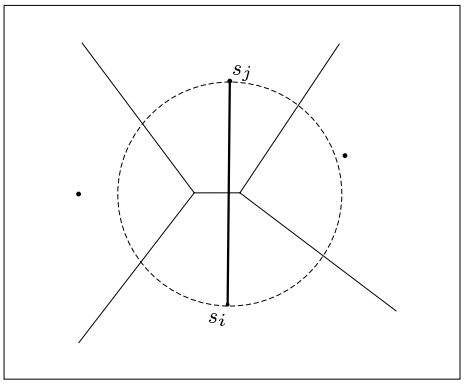
retrouve avec une approximation linéaire dont un coefficient est presque nul, ce qui entraîne de gros problèmes dans les calculs à effectuer ultérieurement).

Graphes

propriété 1.5 *les arêtes extérieures de la triangulation de Delaunay d'un ensemble de points $\mathcal{S}$ constituent la frontière de l'enveloppe convexe de $\mathcal{S}$.*

définition 1.5 *le* **graphe de Gabriel** *[GS69] d'un ensemble $\mathcal{S}$ de points $s_1, \ldots, s_n$ est le graphe dans lequel $[s_i s_j]$ est une arête si et seulement si le cercle ayant $[s_i s_j]$ pour diamètre est un cercle vide.*

Figure 1.3 – [OBS92] une arête du graphe de Gabriel.



propriété 1.6 *$[s_i s_j]$ est une arête du graphe de Gabriel de $\mathcal{S}$ si et seulement si $[s_i s_j]$ est une arête de Delaunay de $\mathcal{S}$ qui coupe l'arête de Voronoï correspondante (voir figure 1.3).*

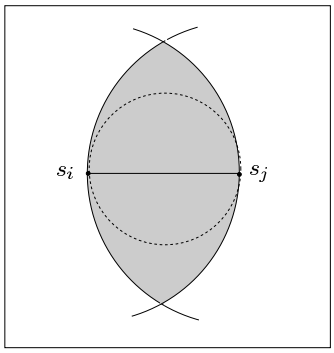
définition 1.6 *Le* **graphe de voisinage relatif** *[Tou80] d'un ensemble de points $\mathcal{S}$ est un graphe géométrique qui contient une arête entre deux sites s_i et s_j de $\mathcal{S}$ si et seulement si :*

$$d(s_i, s_j) \leq \min_{\substack{s \in \mathcal{S} \\ s \neq s_i, s_j}} \max \big(\mathit{dist}(s_i, s), \mathit{dist}(s_j, s) \big).$$

Plus précisément, $[s_i s_j]$ est une arête du graphe de voisinage de $\mathcal{S}$ si et seulement si il n'existe pas d'autre point de $\mathcal{S}$ à l'intérieur de l'intersection entre le disque centré en s_i de rayon $d(s_i, s_j)$ et le disque centré en s_j de même rayon (domaine texturé de la figure 1.4).

propriété 1.7 *le graphe de voisinage relatif d'un ensemble $\mathcal{S}$ de points est un sous-graphe du graphe de Gabriel de $\mathcal{S}$, et donc un sous-graphe de la triangulation de Delaunay de $\mathcal{S}$.*

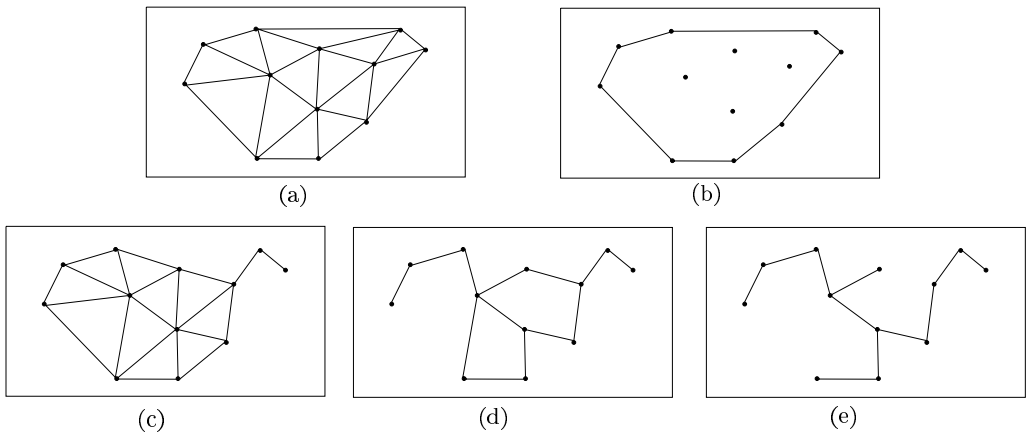
Figure 1.4 – [OBS92] une arête du graphe de voisinage relatif.



définition 1.7 *l'***arbre couvrant minimal euclidien** *d'un ensemble $\mathcal{S}$ de points est l'arbre ayant pour sommets tous les points de $\mathcal{S}$, pour lequel la somme des longueurs eucliennes de toutes les arêtes est minimale parmi tous les arbres ayant pour noeuds les points de $\mathcal{S}$.*

propriété 1.8 *une arête de l'arbre couvrant minimal euclidien d'un ensemble de points $\mathcal{S}$ est aussi une arête du graphe de voisinage relatif de $\mathcal{S}$; par conséquent, l'arbre couvrant minimum euclidien est un sous-graphe de la triangulation de Delaunay.*

Figure 1.5 – [OBS92] des sous-graphes de Delaunay : (a) triangulation de Delaunay, (b) enveloppe convexe, (c) graphe de Gabriel, (d) graphe de voisinage relatif, (e) arbre couvrant minimal euclidien.



Minimum de rugosité

Dans [Rip90], RIPPA montre un résultat original concernant Delaunay et la minimisation de l'intégrale du carré du gradient d'une fonction (une approche plus

simple est donnée par POWAR deux ans plus tard [Pow92]).

définition 1.8 *Soient $\mathcal{S}$ un ensemble de n points du plan et f un vecteur de dimension n dont chaque composante représente l'image d'un point de $\mathcal{S}$ par une certaine fonction. Soit $\mathcal{T}$ une triangulation de l'enveloppe convexe de $\mathcal{S}$ formée de m triangles Δ_i ; on note $P_{\mathcal{T},f}$ l'unique fonction de $\mathbb{R}^2$ dans $\mathbb{R}$, linéaire par morceaux interpolant f pour la triangulation $\mathcal{T}$, qui vérifie :*

$$\forall s \in \mathcal{S}, P_{\mathcal{T},f}(s) = f(s).$$

la ***rugosité*** *du vecteur de données f, pour la triangulation $\mathcal{T}$ est définie par :*

$$R(\mathcal{T}, f) = |P_{\mathcal{T},f}|_{\mathcal{T},1},$$

où $|.|_{\mathcal{T},1}$ est la semi-norme de Sobolev :

$$|g|^2_{\mathcal{T},1} = \sum_{i=1}^{m} |g|^2_{\Delta_i,1} = \sum_{i=1}^{m} \int_{\Delta_i} \left[\left(\frac{\partial g}{\partial x} \right)^2 + \left(\frac{\partial g}{\partial y} \right)^2 \right] dx\, dy = \sum_{i=1}^{m} \int_{\Delta_i} (\nabla g)^2.$$

propriété 1.9 *Soit $\mathcal{S}$ un ensemble de n points distincts et non colinéaires du plan ; alors, la triangulation de l'enveloppe convexe de $\mathcal{S}$ qui minimise la rugosité de n'importe quel vecteur de données (c'est-à-dire en réalité n'importe quelle fonction) est la triangulation de Delaunay.*

Pour établir la preuve de cette propriété, RIPPA exprime localement la rugosité de f pour les deux triangulations possibles d'un quadrilatère convexe ; il montre alors que choisir la bonne diagonale du quadrilatère pour minimiser la rugosité revient à choisir l'arête de Delaunay (en partie, parce que son expression contient le critère du cercle vide).

Géométriquement, la rugosité a une signification ; soit Δabc un triangle de $\mathcal{T}$, et soit $z = \lambda x + \mu y + \nu$ l'équation du plan passant par les points $f(a)$, $f(b)$ et $f(c)$. Alors, la valeur du gradient au carré $(\nabla f_{\mathcal{T}})^2$ sur le triangle Δabc est simplement la constante $\lambda^2 + \mu^2$, et la contribution de Δabc à la rugosité totale est son aire multipliée par cette constante.

Delaunay et le paraboloïde

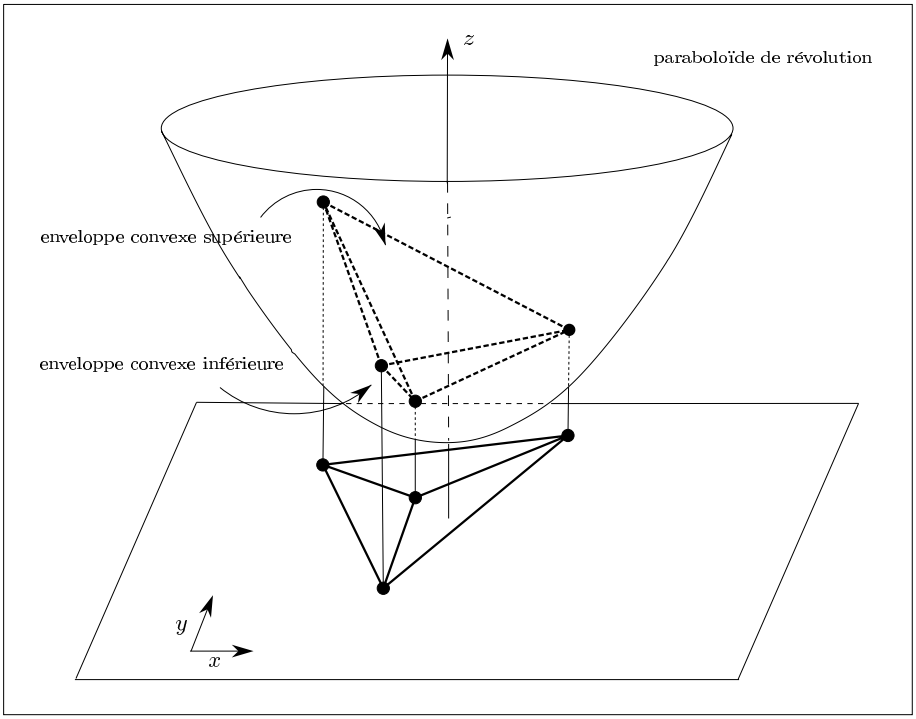
définition 1.9 *Soit $s = (x_s, y_s)$ un point du plan (xOy). On appelle* **relevé** *de s sur la surface d'équation $z = f(x, y)$ (où f représente une fonction quelconque) le point S de l'espace, de coordonnées $(x_s, y_s, f(x_s, y_s))$.*

En partant de cette définition, on peut trouver une relation liant la triangulation de Delaunay d'un ensemble $\mathcal{S}$ de points du plan, et les enveloppes convexes de dimension 3 [Bro79] ; en effet, si tous les points $s_i = (x_i, y_i)$ de $\mathcal{S}$ sont relevés sur le paraboloïde de révolution, on obtient un ensemble de points de l'espace, de coordonnées $(x_i, y_i, x_i^2 + y_i^2)$. L'enveloppe convexe des relevés peut être divisée en deux parties : enveloppe supérieure et inférieure, où l'enveloppe inférieure contient

les triangles dont le plan support sépare l'ensemble de tous les points relevés du demi-espace des z négatifs. On peut alors montrer que :

propriété 1.10 *La triangulation de Delaunay d'un ensemble $\mathcal{S}$ de points du plan est la projection sur le plan (xOy) de l'enveloppe convexe inférieure des relevés de $\mathcal{S}$ sur le paraboloïde de révolution (figure 1.6).*

Figure 1.6 – la projection de l'enveloppe convexe inférieure des relevés donne la triangulation de Delaunay.



Delaunay et fonctions quadratiques en dimension d

La triangulation de Delaunay est celle qui maximise le plus petit angle. Une analyse théorique sur les normes L_p (la norme L_p d'une fonction f à d variables étant égale à la racine p-ième de l'intégrale sur le domaine considéré de la valeur absolue de la fonction élevée à la puissance p) et les interpolations linéaires par morceaux (plans supports des triangles d'une triangulation donnée) d'un ensemble de points montre que l'erreur dépend du plus petit angle de la triangulation. En réalité, on peut montrer que l'important est qu'aucun angle ne soit trop grand. Mais, toutes ces considérations sur les angles ne demeurent vraies que si les dérivées secondes aux points donnés de la fonction f à approcher ont des variations du même ordre ([Rip92b]).

MELISSARATOS [Mel93] s'interroge sur la forme de la triangulation optimale qui servirait à approcher une fonction quadratique convexe, c'est-à-dire une fonction qui s'écrit sous la forme $f(S) = {}^tS\mathcal{H}S + {}^tJS + \nu$ avec :

– $S \in \mathbb{R}^d$, $J \in \mathbb{R}^d$ et $\nu \in \mathbb{R}$,

– $\mathcal{H}$ une matrice symétrique définie positive, de taille $d \times d$ à éléments dans $\mathbb{R}$.

Le critère d'optimalité choisi est une norme L_p où p est quelconque (même ∞), et on se place dans un espace de dimension d elle aussi arbitraire. L'auteur va alors prouver que, bien que l'on sache que la triangulation optimale devra comporter des triangles très loin d'être équiangulaires, il sera possible de la déterminer en recherchant une triangulation de Delaunay.

Pour arriver à ses fins, il montre dans un premier temps que :

théorème 1.1 *l'approximation linéaire par morceaux de la fonction quadratique convexe unité en dimension d ($f(S) = \sum_{i=1}^{d} x_i^2$ si $S = (x_1, \ldots, x_d)$), $d \geqslant 2$, définie par la triangulation de Delaunay, minimise l'erreur L_p, où p est quelconque.*

Par ailleurs, le théorème suivant :

théorème 1.2 *soit f la fonction quadratique convexe donnée par $f(S) = {}^tS\mathcal{H}S + {}^tJS + \nu$. Alors, il existe une transformation linéaire u qui envoie chaque point S_i d'un ensemble $\mathcal{S}$ inclus dans $\mathbb{R}^d$, en un point $S'_i \in \mathbb{R}^d$, de telle sorte que la triangulation de Delaunay de $S'_1, \ldots, S'_n$ corresponde à la triangulation optimale de $\mathcal{S}$ pour la fonction f, par rapport à l'erreur L_p.*

permet d'aboutir au corollaire :

corollaire 1.1 *la triangulation L_p-optimale de n points de $\mathbb{R}^d$ peut être calculée en temps polynomial.*

et la conclusion voulue : rechercher la triangulation optimale dans le sens de la norme L_p avec p quelconque pour l'approximation de fonctions quadratiques convexes dans un espace de dimension quelconque, se fait en se ramenant au calcul d'une triangulation de Delaunay, et donc en temps polynomial. Il faut noter que ce résultat demeure exact pour la norme L_∞.

Cet article fait suite à celui de RIPPA [Rip92b] qui prouvait, entre autres, que la triangulation de Delaunay était bien L_p optimale pour l'approximation de quadriques convexes, mais seulement en dimension 2. En effet, l'auteur, pour arriver à ses fins, utilisait une preuve basée sur un résultat de LAWSON concernant l'échange de diagonales ; or, la propriété en question (une triangulation d'un ensemble de points de $\mathbb{R}^2$ est de Delaunay si l'échange de diagonales faisant appel au critère du cercle vide n'est plus applicable) n'est valable qu'en dimension 2. En revanche, un lemme moins général reste vrai quelle que soit la dimension [Mel93] :

lemme 1.1 *Soit $\mathcal{T}$ une triangulation de n points dans un espace de dimension d. Soit f un $(d-1)$-simplexe de $\mathcal{T}$. Si f peut être swappé suivant le critère de la sphère vide, alors la norme L_p décroit.*

1.1.2 Autres triangulations classiques

Il existe d'autres triangulations très étudiées dans la littérature, parmi lesquelles certaines que l'on ne sait pas construire en temps polynomial.

1.1.2.1 Triangulation de Delaunay contrainte

Soit $\mathcal{S}$ un ensemble de points du plan, et $\mathcal{A}$ un ensemble d'arêtes (les *contraintes*) reliant des points de $\mathcal{S}$.

définition 1.10 *La* **triangulation de Delaunay contrainte** *contient une arête ς entre deux sommets s_i et s_j de $\mathcal{S}$ si et seulement si :*

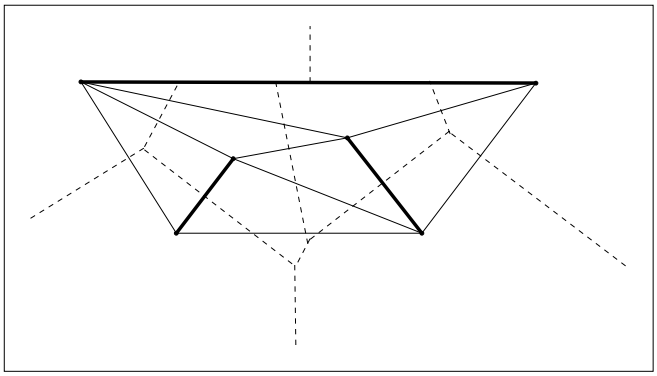
- *s_i est visible de s_j (et inversement),*
- *il existe un cercle passant par s_i et s_j qui ne contient en son intérieur aucun point de $\mathcal{S}$ visible de l'arête ς.*

À l'image de la triangulation de Delaunay classique, on a longtemps cru que la triangulation de Delaunay contrainte était duale d'un autre objet géométrique, le diagramme de Voronoï borné.

définition 1.11 *Le* **diagramme de Voronoï borné** *d'un ensemble de points $\mathcal{S}$ pour un ensemble $\mathcal{A}$ de contraintes (arêtes) est une division du plan en cellules, une pour chaque élément de $\mathcal{S}$, telle que la cellule d'un sommet s soit composée de la région du plan pour laquelle s est le plus proche point de $\mathcal{S}$ visible.*

Un exemple d'un tel diagramme et d'une triangulation de Delaunay contrainte est donné sur la figure 1.7, tirée de [BE92].

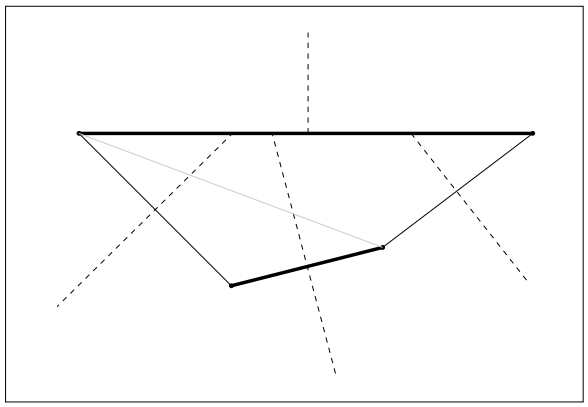
Figure 1.7 – en gras, les contraintes - en lignes fines, la triangulation de Delaunay contrainte - en pointillé, le diagramme de Voronoï borné.



En réalité, la dualité n'est pas totale ; une arête de la triangulation peut exister entre deux sommets de $\mathcal{S}$ dont les cellules du Voronoï borné ne sont pas adjacentes, mais le seraient si elles n'étaient pas traversées par une arête contrainte. Cependant, JOE et WANG [JW93] ont introduit une définition étendue du diagramme de Voronoï contraint pour en faire le dual de Delaunay contraint ; c'est de leur article qu'est reproduit un exemple où le dual du Voronoï borné ne donne pas toute la triangulation de Delaunay contrainte (figure 1.8).

Pour construire une triangulation de Delaunay contrainte, on peut avoir recours à un algorithme d'échanges de diagonales du type LAWSON [Law77] dont la complexité est quadratique, mais qui est facile à mettre en place, ou alors utiliser l'algorithme diviser pour régner de CHEW [Che89a] qui a une complexité optimale de $O(n \log n)$.

Figure 1.8 – le dual du Voronoï borné ne contient pas l'arête grise.



Un résumé sur la triangulation, ses propriétés et ses algorithmes de construction a été publié par De Floriani et Puppo [DP89].

Signalons aussi qu'en dimension 2, tout problème de triangulation contrainte (on ne parle plus seulement ici de Delaunay) admet une solution ; en effet ([BY95], p 278) :

théorème 1.3 *Soit $\mathcal{S}$ un ensemble de points du plan. Tout ensemble maximal de segments joignant les points de $\mathcal{S}$ sans s'intersecter, sauf en une extrémité commune, constitue l'ensemble des arêtes d'une triangulation de $\mathcal{S}$, et réciproquement.*

Ceci prouve que tout ensemble d'arêtes sans intersection sauf en un sommet commun et dont les sommets appartiennent à $\mathcal{S}$ peut être complété en une triangulation de $\mathcal{S}$.

1.1.2.2 Triangulation d'un polygone

La triangulation d'un polygone est un cas particulier de la triangulation contrainte où les arêtes imposées représentent le bord du polygone. Le problème qui se pose est donc le suivant : étant donné $\mathcal{P}$ un polygone simple de sommets $s_1, \ldots, s_n$, il faut ajouter au graphe planaire formé par ce polygone un ensemble de $(n-3)$ nouvelles arêtes (appelées *diagonales*) intérieures à $\mathcal{P}$ et qui ne se coupent pas deux à deux. On obtiendra ainsi un partitionnement du polygone en $(n-2)$ triangles d'intérieurs disjoints.

La triangulation d'un polygone est un des principaux problèmes traités par la géométrie algorithmique, surtout qu'il trouve naturellement sa place dans les problèmes de maillage d'objets où l'objet est généralement donné par sa frontière. De très nombreux travaux traitent ainsi du sujet ; ceux qui traitent de la triangulation de Delaunay de polygones sont répertoriés dans la thèse de Lambert [Lam94], et une bonne partie de ceux qui cherchent à optimiser un critère sont cités dans l'article de synthèse de Bern et Eppstein [BE92]. Parmi les articles les plus récents, on pourra citer :

– un algorithme de Chazelle pour trianguler un polygone simple en temps linéaire [Cha91],

- un article de BERN, DOBKIN et EPPSTEIN [BDE92] qui montre comment trianguler un polygone sans aboutir à des angles supérieurs à 150°, avec $O(n^2)$ triangles (on s'autorise à ajouter des points à l'intérieur du polygone), borne portée pour un polygone convexe à $O(n^{1.85})$ triangles,
- BERN, MITCHELL et RUPPERT [BMR94] ont exhibé un algorithme qui permet de trianguler des régions polygonales avec trous, sans aucun angle supérieur à $\pi/2$, et avec seulement $O(n)$ triangles.

1.1.2.3 Triangulation de poids minimum

définition 1.12 *La* **triangulation de poids minimum** *d'un ensemble de points $\mathcal{S}$ est la triangulation dont la somme des longueurs des arêtes (pour la distance euclidienne) est la plus petite parmi toutes les triangulations de $\mathcal{S}$.*

Cette triangulation est unique pour des points en position générale, mais pour quelques configurations particulières, plusieurs triangulations peuvent donner le même poids total. Elle a aussi été parfois appelée *triangulation optimale*, dans les cas où le poids associé aux arêtes est différent de la distance euclidienne.

Bien qu'il existe un algorithme en $O(n^3)$ pour trianguler suivant ce critère un polygone simple [Kli80], savoir si le problème général de la triangulation de poids minimum est NP-dur demeure toujours une question ouverte [GJ79]. Cependant, très récemment, DICKERSON et MONTAGUE [DM96] ont exhibé un algorithme polynomial capable en général de trouver un sous-graphe de taille suffisamment importante pour réussir à déterminer la triangulation de poids minimum d'un ensemble de points (cet algorithme est décrit en détail au chapitre 6).

1.1.2.4 Triangulation gloutonne

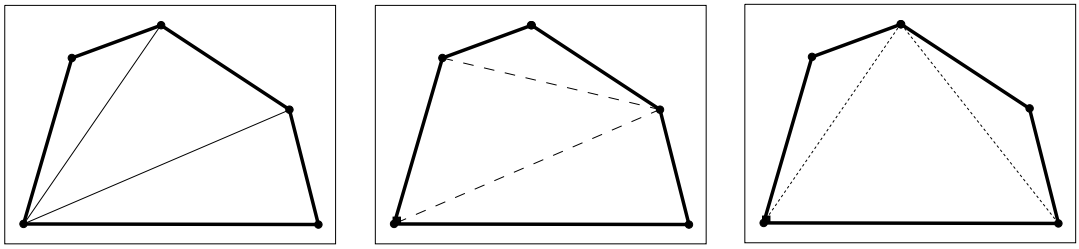
définition 1.13 *La* **triangulation gloutonne** *d'un ensemble $\mathcal{S}$ de points du plan est obtenue en ajoutant une à une les arêtes, en choisissant toujours la plus courte qui ne croise aucune de celles retenues précédemment.*

Cette triangulation minimise lexicographiquement le vecteur trié des longueurs d'arêtes. L'algorithme naïf qui consiste à considérer toutes les arêtes et à les ordonner est en $O(n^2 \log n)$, mais il existe un algorithme en $O(n^2)$ pour la calculer [LL90] (et même $O(n)$ pour des polygones convexes).

Les deux triangulations précédentes sont identiques à celle de Delaunay pour une grande majorité des configurations de points ; cependant, la figure 1.9 (inspirée de [WP84]) montre un cas où elles sont toutes les trois différentes. Par ailleurs, il a été prouvé que la triangulation de Delaunay pouvait aboutir à une somme de longueurs d'arêtes $\Omega(n)$ fois supérieure à celle de la triangulation de poids minimum [Kir80] ; de la même manière, la triangulation gloutonne donne un optimum $\Omega(\sqrt{n})$ fois plus mauvais [MZ79].

Delaunay et les triangulations gloutonnes et de poids minimum sont des *triangulations systématiques* [WP84], c'est-à-dire que l'on peut les construire de manière systématique, par un choix unique d'arêtes, qui dépend en principe de l'emplacement des sommets. Cela implique que, si les points sont en position générale, ces triangulations sont uniques.

Figure 1.9 – un exemple où les trois triangulations sont différentes.



(a) Delaunay (b) gloutonne (c) poids minimum

1.1.3 Construction de triangulations

Une multitude d'articles ont été publiés sur la construction de triangulations d'un ensemble de points, qu'elles soient de Delaunay ou pas ; dans sa thèse [Lam94], LAMBERT a répertorié et classé plus d'une centaine d'entre-eux qui ne concernent que Delaunay (sur une bibliographie qui comporte plus de 350 références).

Voici donc une sélection des méthodes les plus courantes pour réaliser la triangulation de Delaunay d'un ensemble de points. Les complexités de la plupart de ces algorithmes sont détaillées dans [For92] où une comparaison très sommaire des implémentations en temps de calcul est donnée.

1.1.3.1 Algorithmes diviser pour régner

Cette famille d'algorithmes divise récursivement l'ensemble de points jusqu'à ce qu'il ne reste plus que 3 sommets, constituant ainsi un triangle ; une fusion récursive de deux triangulations voisines est alors réalisée de manière à obtenir finalement la triangulation de Delaunay des points initiaux.

La grosse difficulté de cet algorithme devrait se situer dans l'étape de fusion ; il est possible de fusionner deux triangulations de Delaunay quelconques en temps linéaire [Kir79]. Mais, on peut se retrouver dans un cas plus simple si on décide de séparer les triangulations par une droite (souvent choisie parallèle à un axe) ; on sait alors que les nouvelles arêtes à créer couperont la droite de séparation, et on peut même retrouver l'ordre dans lequel elles la croiseront [GS85].

En ce qui concerne la complexité globale de l'algorithme, DWYER [Dwy87] puis KATAJAINEN et KOPPINEN [KK88], en utilisant une séparation effectuée sur l'ensemble des points par à la fois des droites verticales et horizontales, ont obtenu un temps en $O(n)$ pour des points distribués uniformément dans la sphère unité.

1.1.3.2 Algorithmes de balayage

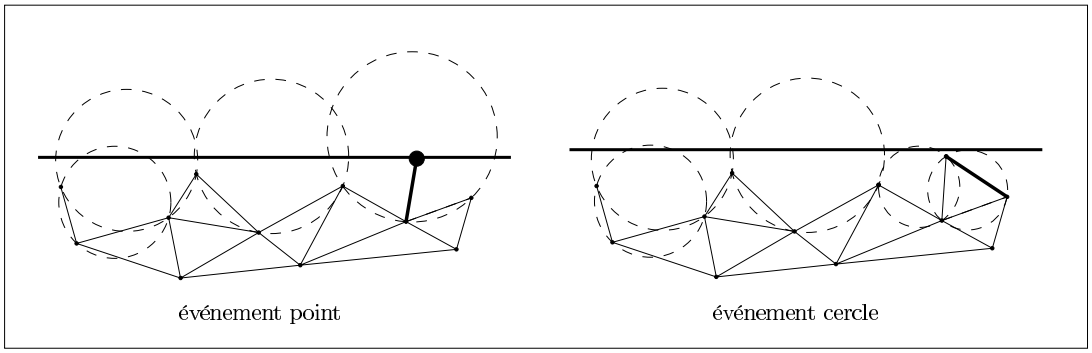
Un algorithme de balayage plan est un algorithme qui parcourt le plan en déplaçant une droite de direction donnée, en général la direction d'un axe de coordonnées. Ce type d'algorithme est couramment utilisé en géométrie algorithmique pour résoudre une grande variété de problèmes ; pour un point de vue général, on pourra se référer à l'ouvrage de PREPARATA et SHAMOS [PS85] ou à celui de BOISSONNAT et YVINEC [BY95].

FORTUNE [For87] a adapté cette méthode pour calculer le diagramme de Voronoï ou la triangulation de Delaunay d'un ensemble de points. Les points sont supposés être en position générale et avoir des ordonnées différentes ; en effet, c'est la direction horizontale qui est choisie pour balayer le plan. Son algorithme (comme tous ceux qui ont recourt au balayage) conserve deux ensembles d'état :

- la *liste des états*, qui contient la *frontière* de la partie de la triangulation déjà construite, et une liste des cercles circonscrits à 3 points et vide de points, au-dessous de la droite de balayage, et donc candidats à créer des triangles de Delaunay.

- la *queue des événements* qui garde une trace de l'endroit où la ligne de balayage doit s'arrêter. Deux sortes d'événements peuvent survenir, illustrés par la figure 1.10 :
  - les événements *points* lorsque la droite atteind un point : si le point est à l'intérieur d'un cercle, alors les 3 points associés à ce cercle ne forment pas un triangle de Delaunay ; on ajoute l'arête formée du nouveau point et de son plus proche voisin, et on met à jour la frontière ainsi que la liste des cercles circonscrits vides.
  - les événements *cercles* lorsque la droite atteind le haut d'un cercle formé par trois sommets adjacents de la frontière ; cela implique que le cercle est vide de tout autre point que ses sommets, et que les 3 sommets associés forment bien un triangle de Delaunay. La troisième arête du triangle est ajoutée et la frontière mise à jour.

Figure 1.10 – les deux sortes d'événements de l'algorithme de balayage de FORTUNE



Le nombre total d'événements qui peuvent se produire est égal au nombre d'arêtes de la triangulation de Delaunay, soit $O(n)$. La frontière est un sous-ensemble de Delaunay, donc de taille $O(n)$. La queue des événements contient au plus un événement pour chaque point, et un événement pour chaque arête de la frontière, soit une taille en $O(n)$. En utilisant les bonnes structures de données, les mises à jour et les recherches peuvent être réalisées en temps $O(\log n)$. La complexité temporelle de l'algorithme de balayage est donc de $O(n \log n)$.

1.1.3.3 Algorithmes incrémentaux

Ce sont les algorithmes les plus simples et ceux pour lesquels la bibliographie est la plus importante. Le principe est d'ajouter les points les uns après les autres et de mettre à jour la triangulation après chaque ajout. Les deux étapes d'un tel algorithme sont donc :

- la *localisation* qui doit permettre de trouver le triangle contenant le point à insérer (on suppose au départ disposer d'un triangle infini qui contient tous les points du plan). Si le point se trouve en dehors de l'enveloppe convexe, il suffit de réactualiser celle-ci. Le problème est un peu différent lorsque le point est à l'intérieur de l'enveloppe convexe ; il faut alors rechercher le triangle qui le contient. Ceci peut être fait en partant d'un point connu de la triangulation, et en visitant de proche en proche tous les triangles traversés par le segment reliant ce point connu au nouveau sommet à insérer. Si les points sont distribués uniformément, il faudra en moyenne parcourir $O(\sqrt{n})$ triangles à chaque étape de localisation. Cette borne peut être améliorée en réordonnant les points de telle sorte que deux sites successifs soient proches et en partant du triangle contenant le dernier point inséré, ou en utilisant des structures de données bien adaptées (comme l'arbre de Delaunay de Boissonnat et Teillaud [BT86]).

- la *mise à jour*, qui peut être réalisée en échangeant les arêtes invalidées par l'insertion du nouveau point (approche d'abord suggérée par Lawson [Law77]). Une arête peut être testée en appliquant le critère du cercle vide ; à chaque fois qu'elle est invalidée, elle est échangée avec l'autre diagonale du quadrilatère où elle était située, et de nouvelles arêtes sont ajoutées à la liste de celles qui doivent être testées. Dans le pire des cas, on peut construire un exemple où pour un ordre d'insertion fixé, $\Omega(n^2)$ tests et échanges seront nécessaires. Cependant, l'expérience montre qu'en moyenne, pour une distribution uniforme ou un ordre d'insertion bien choisi, seulement un nombre constant d'arêtes sont invalidées à chaque insertion, et donc $O(n)$ sur l'ensemble de l'algorithme.

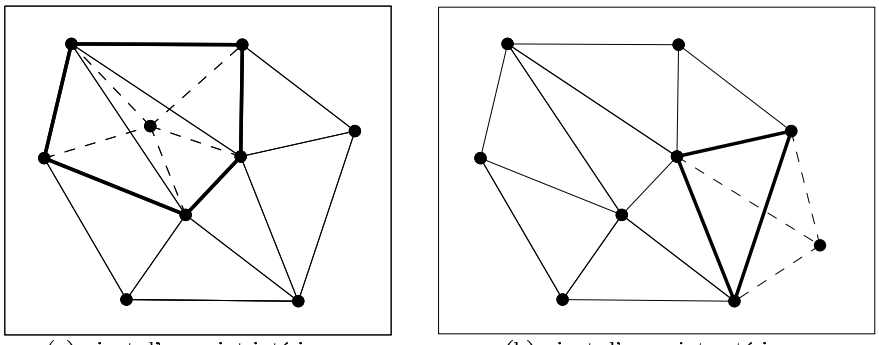
Approche de Watson

au lieu de procéder par échanges d'arêtes dans la phase de mise à jour, Watson [Wat81] propose de détruire tous les triangles en conflit avec le nouveau point (par le critère du cercle vide). Les triangles ainsi supprimés forment un polygone étoilé que l'on triangule en reliant le nouveau sommet à chaque arête de sa frontière. L'algorithme a une complexité moyenne (qui correspond au nombre d'arêtes créées) de $O(n\sqrt{n})$, et de $\Omega(n^2)$ dans le cas le pire. La figure 1.11 illustre les 2 cas qui peuvent se produire lors de l'insertion d'un point, pour l'approche de Watson.

Approche randomisée

dans ce type d'algorithme incrémental, les points sont insérés dans un ordre aléatoire ; cette approche a été utilisée par Boissonnat et Teillaud [BT93] puis Guibas, Knuth et Sharir [GKS92]. Ces derniers ont ainsi prouvé que le nombre moyen de swaps était linéaire, quelle que soit la distribution des points. Pour la phase de localisation, une structure d'arbre est utilisée (l'*arbre de Delaunay* pour [BT93])

Figure 1.11 – la phase de mise à jour d'une triangulation de Delaunay, après insertion d'un nouveau sommet.



(a) ajout d'un point intérieur (b) ajout d'un point extérieur

dans laquelle un noeud interne est un triangle qui a été détruit par l'insertion d'un point à un certain moment ; les feuilles contiennent la triangulation courante. Situer un point p revient alors à localiser p dans toutes les triangulations ayant existé successivement. On peut prouver que la localisation est réalisée avec un coût moyen en $O(\log n)$, et donc que l'algorithme a un coût total moyen de $O(n \log n)$.

1.1.3.4 Algorithmes du type papier cadeau

Aussi appelés algorithmes de *sélection*, la version naïve consiste seulement à considérer les $\binom{n}{3}$ triangles et à regarder s'ils vérifient ou pas le critère du cercle vide en temps $O(n)$, ce qui donne un algorithme en $O(n^4)$.

De manière plus astucieuse, on part d'un triangle de Delaunay et on découvre incrémentalement les suivants, un par un. Chaque nouveau triangle est obtenu à partir d'une arête d'un triangle trouvé précédemment, en cherchant le point qui joint les sommets de l'arête de façon à obtenir un cercle vide. Plusieurs algorithmes utilisant ce principe sont disponibles dans la littérature, mais celui de DWYER [Dwy91] comprend en plus une analyse formelle ; il suppose (en toute dimension) que les points sont uniformément distribués dans la boule unité et montre que l'extension au cube unité n'est pas triviale. Pour arriver à une bonne complexité, il est obligé de trier les points (en utilisant un tri par paquets) de manière à ne pas parcourir trop de sites avant de trouver celui qui forme un cercle vide avec l'arête considérée. Pour des points dans le plan, l'expérience montre qu'en pratique le temps de calcul de son algorithme est linéaire [SD95]. Cet algorithme ainsi que certains autres (de la même famille, mais un peu différents dans les structures utilisées) sont décrits plus en détail dans la thèse de LAMBERT [Lam94].

1.1.3.5 Algorithmes à base d'échanges d'arêtes

Ils diffèrent des algorithmes incrémentaux dans la mesure où les échanges ne sont effectués qu'une fois qu'une triangulation initiale a été construite ; LAWSON [Law77] a prouvé que quel que soit l'ordre des échanges considéré, en utilisant le critère du cercle vide (ou un autre critère équivalent), on convergeait toujours vers

la triangulation de Delaunay. C'est jusqu'à présent le seul cas connu où un critère d'optimalité locale aboutit à une optimalité globale.

Pour débuter, on peut choisir par exemple la triangulation *étoilée* obtenue en insérant les points les uns après les autres et en les reliant aux sommets visibles déjà présents [Bar77]. Puis, si on utilise une liste de triangles, on peut tester chaque triangle et ses trois voisins, et réaliser les échanges convenables : une seule passe de la liste suffira, à condition d'y ajouter les nouveaux triangles en queue.

Comme il y a $\binom{n}{2}$ arêtes possibles, il y aura au plus $O(n^2)$ swaps, ce qui représente la complexité théorique de cette famille d'algorithmes ; en pratique, pour des ensembles généraux, on observe une complexité moyenne moins importante.

1.1.3.6 Algorithmes à base d'enveloppe convexe

EDELSBRUNNER et SEIDEL [ES86] ont trouvé que la triangulation de Delaunay de n points (x_i, y_i) dans le plan correspondait à la projection planaire de l'enveloppe convexe inférieure des points relevés sur le paraboloïde de révolution, points de coordonnées $(x_i, y_i, x_i^2 + y_i^2)$, et que ce résultat s'étendait aux dimensions supérieures. Ce lien entre la triangulation de Delaunay en dimension d et l'enveloppe convexe en dimension $d+1$ permet ainsi d'utiliser les algorithmes de construction d'enveloppe convexe en dimension trois, comme celui en $O(n \log n)$ de PREPARATA et HONG [PH77]. Pour des références récentes concernant les algorithmes de calcul d'enveloppes convexes, on pourra lire la thèse de NIELSEN [Nie96]. Une implémentation robuste (du moins en dimensions inférieures à 6) d'un algorithme de construction d'enveloppe convexe existe (*Quickhull* [BDH93]).

1.1.3.7 Comparaison des méthodes séquentielles

SU et DRYSDALE ont présenté [SD95] une étude expérimentale visant à comparer les performances de diverses méthodes pour obtenir la triangulation de Delaunay d'un ensemble de points :

- l'algorithme diviser pour régner de DWYER [Dwy87],
- l'algorithme de balayage de FORTUNE [For87],
- plusieurs algorithmes incrémentaux :
  - l'arbre de Delaunay [Dev92] programmé par DEVILLERS,
  - une version incluant un tri par paquets des sommets, par OHYA, IRI et MUROTA [OIM84],
  - une version implémentée spécialement par eux, qui utilise un tri par paquets à schéma de parcours des paquets différent,
- un algorithme du type papier cadeau,
- l'algorithme (très robuste) à base d'enveloppes convexes de BARBER, DOBKIN et HUHDANPAA [BDH93].

Leurs comparaisons reposent, malheureusement, sur des tests de programmmes conçus de manières différentes et par des personnes différentes ; et bien qu'ils n'aient

pas seulement observé les temps de calculs (qui sont très fortement liés à la manière de programmer, notamment pour tout ce qui concerne la gestion de la mémoire), il est à peu près évident que leur façon d'envisager leurs tests a favorisé certains codes. De plus, même les tests d'ensembles qu'ils ne considèrent pas comme aléatoires ne correspondent pas à des cas particulièrement dégénérés, qui auraient alors sans doute amélioré les performances des algorithmes ne faisant pas d'hypothèse sur la distribution des points.

Quoiqu'il en soit, leur conclusion est que l'algorithme de DWYER est celui qui donne les meilleurs temps de calcul, que les ensembles de points soient uniformément distribués ou pas (complexité linéaire si la distribution est uniforme, en $O(n \log n)$ sinon) et quelle que soit leur taille. Suivent leur implémentation utilisant des tris par paquet, puis le programme de balayage de FORTUNE. Les algorithmes incrémentaux contenant une structure élaborée (comme l'arbre de Delaunay) et celui à base d'enveloppes convexes sont beaucoup plus lents.

Ces résultats, bien que devant être considérés avec beaucoup de précautions, peuvent être intéressants pour la personne qui voudra se lancer dans la programmation d'un algorithme de triangulation de Delaunay.

1.1.3.8 Algorithmes parallèles

Dans sa thèse [Su94], SU présente des algorithmes parallèles déjà existants dont le gros défaut semble être leur complexité, et leur manque de performance en rapport avec leur complexité. Il propose ainsi notamment une adaptation plus simple d'un algorithme séquentiel incrémental qu'il baptise "concurrent", pour lequel il développe des structures de données spécifiques. Et sa principale conclusion est que (comme cela se produit souvent...) la méthode la plus simple est celle qui fonctionne le mieux en pratique, même si elle est moins élégante que ses concurrentes.

1.2 Triangulations et éléments finis

De nombreux problèmes physiques sont modélisés par des équations aux dérivés partielles dont les solutions sont approchées numériquement par la méthode des éléments finis [ZT91]. Cette méthode consiste à n'évaluer des valeurs approchées du champ recherché qu'en certains points du domaine, puis à déduire de ces valeurs, par interpolation, la solution en tout point. Une première étape va ainsi consister à réaliser un maillage du domaine qui définira, entre autres choses, les points d'évaluation (appelés aussi *noeuds*).

Ce prétraitement est une phase essentielle de la méthode ; le maillage, en définissant les noeuds, va déterminer la convergence de la méthode vers une "bonne" solution. De plus, le domaine peut avoir une géométrie complexe, ce qui implique que l'obtention du maillage n'est pas triviale. Nous nous intéresserons, dans cette section, uniquement aux maillages triangulaires.

1.2.1 Méthodologie générale

Dans son ouvrage [Geo91], GEORGE affirme que la conception d'un maillage (et donc d'une triangulation), pour la méthode des éléments finis, peut être réalisée suivant trois étapes :

- l'**analyse du problème**, qui consiste à étudier la géométrie du domaine ainsi que le problème physique à résoudre ; ce problème compliqué sera décomposé en sous-problèmes plus simples.
- la **construction formelle du maillage** : elle prend en compte les résultats de l'analyse, et définit des objets simples permettant de diviser le travail total en étapes.
- la **réalisation du maillage**, qui comprend deux phases, la création des données nécessaires (noeuds, voisinages) et la création effective du maillage (liaisons entre les noeuds).

Seule la troisième étape, et dans une moindre mesure la seconde, nous intéressent, et dans le cadre restreint des maillages triangulaires. On pourra voir quelles techniques existent qui permettent de donner un certain aspect à une triangulation, ou comment modifier cette triangulation lorsque, malgré l'analyse du problème, le résultat n'est pas satisfaisant.

1.2.2 Classification des maillages

La construction de maillages pour la méthode des éléments finis est un problème très abordé dans la littérature scientifique aussi bien par les numériciens que par les géomètres ; étrangement, peu d'articles proposent de décrire brièvement les diverses méthodes existant. HO-LE [HL88] propose cependant une classification des maillages se basant sur l'ordre dans lequel ses principaux éléments sont créés, à savoir les noeuds et les faces (triangles ou quadrilatères, en général, pour la dimension 2).

Notre centre d'intérêt concerne uniquement les triangulations 2D et le bref résumé de l'article de HO-LE qui va suivre ne s'occupera donc que du cas 2D (alors que l'auteur évoque aussi bien les grilles non triangulaires que les problèmes de dimension 3).

1.2.2.1 Topologie du maillage en priorité

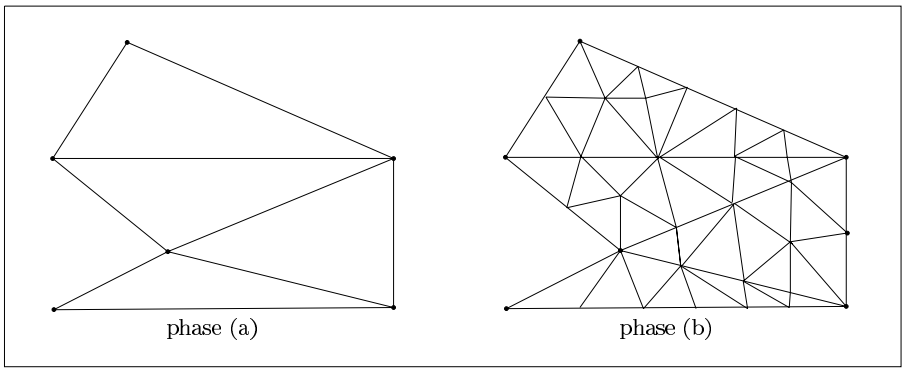
Certaines méthodes déterminent d'abord la topologie du maillage, pour ensuite placer les noeuds, puis, éventuellement, effectuer un lissage du maillage. Un gros problème, est que l'on ne sait, dans la plupart des cas, définir une topologie qu'en ayant recours à d'autres techniques de création de maillages.

1.2.2.2 Noeuds en priorité

Une première approche, notamment développée par WÖRDENWEBER [Wör83, Wör84], consiste à décomposer un objet polygonal, grâce à deux entités : ses sommets et ses arêtes. Chaque entité possède une géométrie et une topologie propres, et c'est la

topologie de l'objet lui-même qui détermine la taille et la forme des éléments grossiers triangulaires qui seront formés (comme sur la figure 1.12 inspirée de [HL88]). Une phase de raffinement est ensuite utilisée.

Figure 1.12 – l'approche de WÖRDENWEBER : (a) les sommets sont reliés pour former des éléments grossiers, puis, (b) le maillage est raffiné.



Une autre méthode : les noeuds, points qui représentent une donnée du problème physique, sont d'abord engendrés, puis sont connectés pour obtenir une triangulation. De conception simple, c'est cette approche qui est la plus répandue. Ses deux phases peuvent être résolues de diverses manières :

- pour la création des noeuds :
  - soient ils constituent une donnée du problème (ils ont été créés manuellement),
  - soit on les engendre à l'aide de différentes techniques, le plus souvent aléatoires, en respectant la densité recherchée, ou d'après une règle de distribution uniforme (par intersections du polygone avec des lignes horizontales espacées régulièrement, ou en utilisant la représentation CSG de l'objet, par exemple).

- pour la création des faces (des triangles dans les cas qui nous intéressent), la plupart des procédés qui existent et qui sont décrits dans l'article de HO-LE reviennent à effectuer une triangulation de Delaunay de l'ensemble des points, ce qui permet d'aboutir en un temps raisonnable à une partition toujours identique du domaine.

1.2.2.3 Les modèles de maillages adaptés

Un modèle simple sert de support à la construction d'un premier maillage qui sera ensuite déformé pour s'adapter au domaine à trianguler. On peut distinguer trois approches différentes :

- le modèle est une grille infinie rectangulaire ou triangulaire, dont les éléments intérieurs sont tous identiques et ont une forme "optimale" ; les éléments du bord dépendent, eux, beaucoup de la méthode de construction utili-

sée, et peuvent avoir des formes bien moins agréables. Dans cette catégorie de maillage, on peut citer le très célèbre "quadtree" ou *arbre quaternaire* [Sam90].

- le modèle est un maillage triangulaire dans le triangle unité (ou rectangulaire dans le rectangle unité) ; l'objet est découpé (manuellement ou automatiquement) en éléments simples à trois côtés, et on utilise une fonction de passage pour appliquer le maillage choisi à chaque élément simple.

- si on désire trianguler un polygone simple $\mathcal{P}$, on peut avoir recours à un autre polygone simple $\mathcal{P}'$, de même nombre de sommets que $\mathcal{P}$, déjà triangulé avec des triangles de forme convenable ; on adapte alors le maillage grâce à la fonction de passage des sommets de $\mathcal{P}'$ à ceux de $\mathcal{P}$.

1.2.2.4 Création simultanée des noeuds et des éléments

La géométrie de l'objet est prise en compte pour essayer de créer de "bons" triangles. BYKAT [Byk76], par exemple, a décrit un algorithme où le domaine polygonal est, dans un premier temps, divisé en parties convexes ; des sommets sont ensuite ajoutés, en respectant la densité du maillage, d'abord sur les bords du domaine puis le long d'une droite. Les deux moitiés définies par cette droite sont alors récursivement divisées jusqu'à obtention de triangles.

Très répandus maintenant, des mailleurs utilisent la technique dite d'**avancée de front** [GS92]. La donnée initiale est un contour discret de l'objet, contour composé d'un ensemble de points reliés par des arêtes ; c'est ce que l'on appelle le front initial, à partir duquel le maillage sera construit. Un triangle est créé à partir du front soit en utilisant un point qui existe déjà, soit en créant un nouveau sommet tel que les triangles engendrés soient valides (aucun autre sommet à l'intérieur) et les plus équilatéraux possibles. Le front est remis à jour après chaque création d'éléments, et l'algorithme s'arrête lorsque l'intérieur du domaine est entièrement triangulé (la figure 1.13 illustre la méthode). Comme pour tous les maillages de type éléments finis, une phase d'optimisation est souvent nécessaire pour obtenir des triangles de la forme souhaitée.

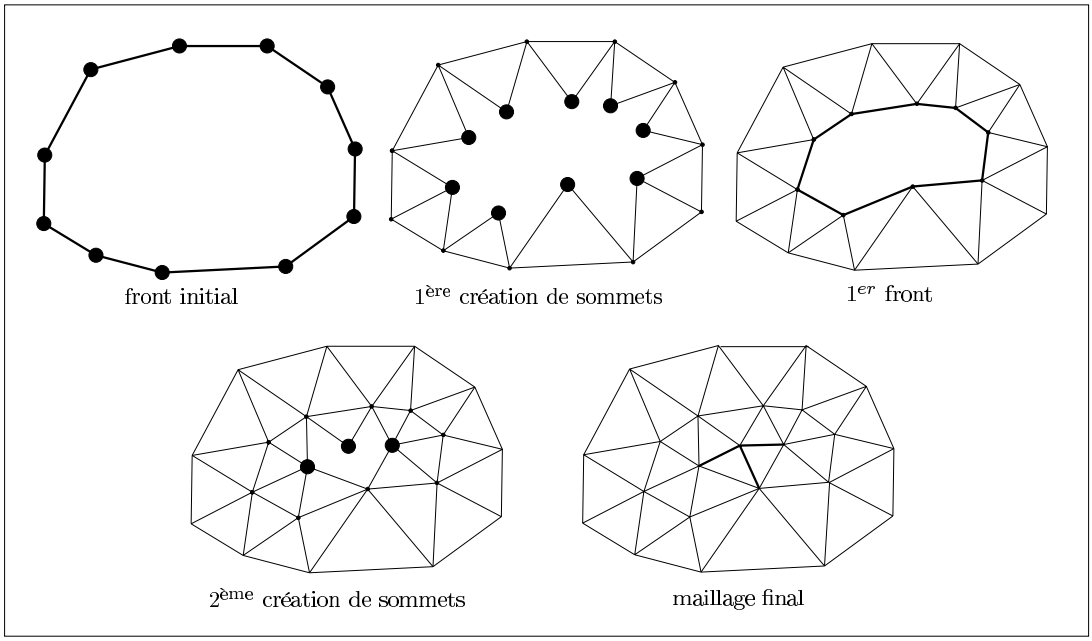
1.2.3 Amélioration de maillages

Une fois qu'un premier maillage est obtenu, il est rare qu'il soit totalement satisfaisant. Il existe alors plusieurs moyens de l'améliorer, que ce soit en ajoutant ou en déplaçant des points, ou encore en effectuant des échanges d'arêtes. Nous allons donc voir ici quelques unes des techniques géométriques les plus courantes, et rappeler certains résultats théoriques et pratiques qu'il est possible d'atteindre grâce à ces améliorations.

1.2.3.1 Raffinement

Le problème du *raffinement* (ou encore *affinage*) est devenue une étape importante dans le processus de la construction d'un maillage, surtout lorsque ce maillage est la base d'un processus tel que la méthode des éléments finis, et que le résultat n'est pas jugé satisfaisant.

Figure 1.13 – les différentes étapes d'un maillage obtenu par une méthode de type frontal.



De manière générale, le raffinement peut être formulé de la manière suivante : étant donné une triangulation valide non dégénérée, on désire affiner localement la triangulation afin d'obtenir des renseignements plus précis dans les zones délicates du problème (comme les contours d'un objet autour duquel on voudrait étudier l'écoulement d'un fluide, ou autour des ailes d'un avion où les perturbations causées par le passage de l'air sont importantes), tout en conservant certaines propriétés indispensables (par exemple garder une borne inférieure acceptable sur les angles de chaque triangle).

Le raffinement selon Rivara

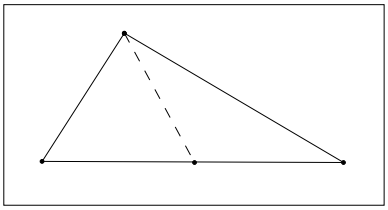
C'est à ce type de problèmes que RIVARA s'intéresse depuis de nombreuses années ; après avoir posé les fondements de sa technique de raffinement dès 1984 [Riv84a, Riv84b], elle a ensuite régulièrement publié soit des améliorations, soit des preuves de propriétés vérifiées par son critère [Riv87, Riv89, Riv91, RL92, Riv93, RI95]. Au travers de ces différents articles, elle a mis en place une méthode qui semble avoir fait ses preuves et qui bénéficie maintenant de résultats théoriques assez variés et intéressants.

Définitions Plus précisément, elle cherche à raffiner localement une triangulation de telle sorte que le plus petit (ou le plus grand) angle soit borné, et on sait combien cette borne peut être importante pour l'utilisation d'une grille dans la méthode des éléments finis [BA76]. Par ailleurs, elle fait en sorte que la triangulation soit *lissée*, c'est-à-dire que l'on ne passe pas brutalement d'un grand à un petit

triangle.

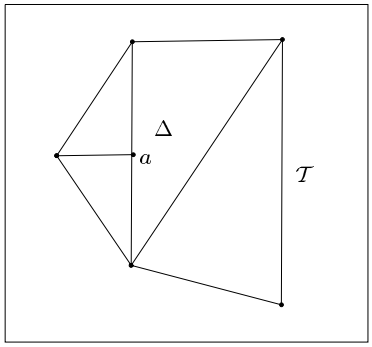
définition 1.14 *La **bissection** du plus long côté (au sens de la distance euclidienne) d'un triangle Δ est la partition du triangle obtenue en ajoutant l'arête reliant le milieu de ce plus long côté au sommet opposé (figure 1.14).*

Figure 1.14 – bissection d'un triangle.



définition 1.15 *Un point est dit **$\frac{1}{2}$-non conforme** si c'est un point intérieur à une arête d'un triangle et le sommet commun à deux autres triangles (figure 1.15).*

Figure 1.15 – un point a, un triangle Δ et une triangulation $\mathcal{T}$ $\frac{1}{2}$-non conformes.



définition 1.16 *Un triangle est **$\frac{1}{2}$-non conforme** si il comporte un point $\frac{1}{2}$-non conforme, et une triangulation est $\frac{1}{2}$-non conforme si elle contient un ou plusieurs triangles $\frac{1}{2}$-non conformes (les autres sont conformes).*

Algorithmes Un algorithme de raffinement global (c'est-à-dire que l'on applique le raffinement à tous les triangles) est donné dans [Riv84a] :

– **pour** tout triangle Δ de $\mathcal{T}$
  on **bissecte** Δ selon sa plus grande arête

fin pour

– $\mathcal{T}^* \leftarrow$ triangulation ainsi créée

– $\mathbf{T} \leftarrow$ ensemble des triangles $\frac{1}{2}$-*non conformes* de $\mathcal{T}^*$

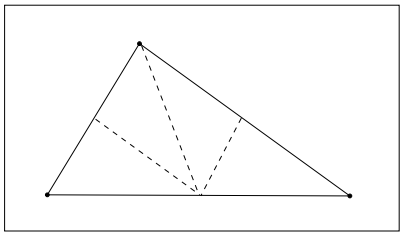
– **pour** tout triangle Δ de $\mathbf{T}$
on **bissecte** Δ selon sa plus grande arête
on **ajoute** les arêtes nécessaires pour que $\mathcal{T}^*$ soit toujours une triangulation

fin pour

Une version locale, un peu plus compliquée, existe également : elle permet de définir seulement une zone, donnée parmi les triangles de départ, où le raffinement sera appliqué [Riv84a, Riv84b]. Cette possibilité de ne choisir qu'un lieu d'affinage est notamment très utile lorsque l'on se trouve autour de singularités dans la méthode des éléments finis.

En remarquant ensuite que, dans les méthodes classiques de multigrilles, chaque élément était divisé en 4 lors d'un raffinement global - ce qui permet de connaitre à l'avance l'effet d'un raffinement sur la grille - alors que son algorithme pouvait diviser un triangle en 2, 3 ou 4 morceaux, RIVARA a eu l'idée d'utiliser "sa" bissection pour séparer un triangle en 4 suivant la méthode présentée sur la figure 1.16. Elle prouve alors que toutes les bonnes propriétés de son raffinement classique à bissection simple sont conservées [Riv87]. Les algorithmes servant à raffiner des triangulations en utilisant ce principe sont pratiquement les mêmes que pour la bissection simple.

Figure 1.16 – un exemple de raffinement par division en 4 du triangle en utilisant le milieu de la plus longue arête.



Propriétés À chaque itération de raffinement (local ou global), une triangulation conforme non dégénérée $\mathcal{T}$ donne une nouvelle triangulation $\mathcal{T}^*$ conforme qui a les propriétés suivantes :

– $\mathcal{T}^*$ est emboîtée dans $\mathcal{T}$, c'est-à-dire que chaque triangle raffiné est inclus dans un seul triangle de $\mathcal{T}$.

– $\mathcal{T}^*$ n'est pas dégénérée : les angles intérieurs de tous ses triangles sont différents de 0.

– La transition entre les grands et les petits triangles de $\mathcal{T}^*$ (cas d'un raffinement local) est lisse, c'est-à-dire que l'on ne passe pas brutalement d'un grand à un petit triangle.

Par ailleurs, ces algorithmes convergent tous vers une certaine triangulation après un nombre fini de bissections ; cela provient notamment de théorèmes montrés par STYNES [Sty80, Riv84a].

Certains résultats intéressants sur ce raffinement ne sont pas dûs directement à RIVARA. Ainsi, la borne donnée pour les angles découle du théorème suivant, dû à ROSENBERG et STENGER [RS75] :

théorème 1.4 *Soit θ_0 le plus petit angle intérieur d'une triangulation initiale donnée $\mathcal{T}_0$. Si on note θ_i le plus petit angle de la triangulation $\mathcal{T}_i$ obtenue en itérant i fois un algorithme de bissection à tous les triangles de $\mathcal{T}_0$, alors :*

$$\theta_i \geqslant \frac{\theta_0}{2}, \forall i.$$

Toute l'astuce de la méthode de RIVARA réside en fait dans le traitement des triangles non conformes ; en effet, avant elle, d'autres personnes avaient déjà imaginé raffiner en utilisant le milieu de la plus longue arête. Mais, elles résolvaient leur problème de non conformité en reliant le milieu ajouté au sommet opposé du triangle voisin, perdant ainsi tout contrôle sur la non dégénérescence de leur maillage (puisqu'apparaissaient alors des triangles avec des angles de plus en plus petits).

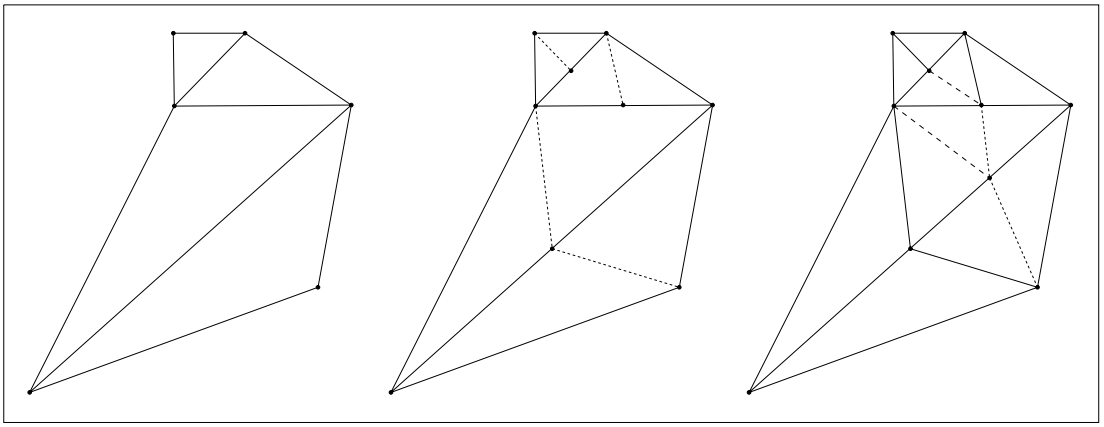
Quant au fait que l'on ne passe pas brutalement de grands triangles à des petits, cela provient de :

théorème 1.5 *Soient $\mathcal{T}_i$ les triangulations obtenues en itérant i fois un algorithme de bissection à tous les triangles d'une triangulation conforme donnée $\mathcal{T}_0$. Si on note Δ_1 et Δ_2 deux triangles voisins de $\mathcal{T}_i$ de diamètres respectifs $diam_1$ et $diam_2$, alors :*

$$\frac{\min(diam_1, diam_2)}{\max(diam_1, diam_2)} \geqslant \upsilon, \text{ avec } \upsilon > 0,$$

où υ dépend seulement du plus petit angle de la triangulation initiale et est indépendant de i.

Figure 1.17 – un exemple de triangulation raffinée une fois par la méthode de RIVARA.

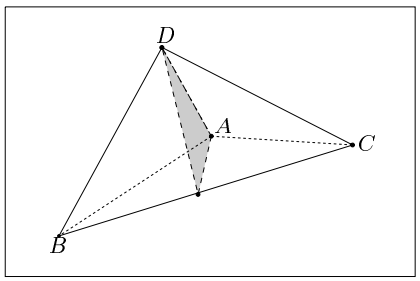


Enfin, des résultats plus pointus sont mis en avant dans [RI95] ; les auteurs donnent une borne du nombre de points à ajouter à l'intérieur et à l'extérieur d'une région circulaire à raffiner par la bissection qui découpe un triangle en 4.

Déraffinement Afin d'appliquer sa bissection à d'autres problèmes, tels que ceux observés en dynamique des fluides ou dans les processus d'avancés de fronts, il manquait à la méthode de RIVARA la faculté de pouvoir déraffiner automatiquement une triangulation de façon à pouvoir revenir en arrière. C'est chose faite dans l'article [Riv89] où l'auteur présente des algorithmes (et des structures de données) permettant de raffiner et déraffiner à volonté tout en conservant les propriétés d'emboîtement des triangulations successives.

Extension à la dimension 3 Dans [Riv91] puis avec LEVIN dans [RL92], RIVARA montre que le raffinement par la bissection s'étend assez naturellement aux tétraédrisations : un tétraèdre Λ étant donné par ses 4 sommets, la bissection générale de Λ est la bissection par le plan passant par le milieu de la plus longue arête de Λ et les deux autres sommets. Par exemple, sur la figure 1.18, la bissection du tétraèdre $\Lambda ABCD$ est effectuée par le plan contenant le milieu de $[BC]$ (plus longue arête de $\Lambda ABCD$) et les points A et D.

Figure 1.18 – bissection d'un tétraèdre.



Un tétraèdre sera **$\frac{1}{2}$-non conforme** s'il existe au moins un point qui soit à la fois milieu de l'arête d'(au moins) un tétraèdre et sommet d'au moins deux autres tétraèdres. Dans un maillage $\frac{1}{2}$-non conforme, il existe des noeuds, des arêtes et des faces qui ne sont pas conformes.

L'algorithme de raffinement global fonctionne exactement de la même façon que pour le cas de la dimension 2 (excepté évidemment que l'on doit rechercher des tétraèdres voisins non conformes d'un point que l'on vient d'ajouter pour les raffiner, et non plus des triangles).

Bien qu'il ait été prouvé que les raffinements successifs donnaient des tétraédrisations dans lesquelles les précédentes étaient toujours incluses, il n'existe aucune démonstration de la convergence (ou divergence) du plus petit angle inclus dans les simplexes. Cependant, RIVARA et LEVIN ont testé la qualité du maillage produit sur des exemples en observant le comportement des angles en chaque point [RL92] ; sur leurs exemples, il semble qu'il y ait clairement une borne asymptotique de convergence de l'angle minimum différente de 0, mais aucune preuve... Par ailleurs, il

apparait que le pourcentage de mauvais tétraèdres (au sens angulaire) tend à décroître lorsque le nombre d'étapes de raffinements effectuées augmente.

Autres algorithmes de raffinement

Beaucoup d'autres auteurs se sont intéressés au problème du raffinement ; parmi eux, BANK, SHERMAN et WEISER [BSW83] avaientt aussi utilisé le principe de la bissection et de la trissection, en ajoutant des sommets au milieu des arêtes des triangles.

On peut également signaler une méthode due à FREY [Fre87], où des sommets sont d'abord ajoutés sur la frontière du domaine suivant une fonction d'espacement ; puis, une triangulation de Delaunay des points de données et des points ajoutés (dits de *Steiner*) est calculée. Des points de Steiner intérieurs sont alors créés d'après le principe suivant : on cherche un triangle Δ qui contient le centre de son cercle circonscrit ; on engendre de manière temporaire un point de Steiner s sur le segment qui relie les centres des cercles inscrit et circonscrit de Δ ; si s n'est pas trop proche des sommets de Δ, on ajoute s et on reconstruit la triangulation de Delaunay.

1.2.3.2 Ajouts de points

L'utilisation de points de Steiner n'est pas très différente du raffinement ; un recours systématique à cette méthode pose cependant le problème d'une trop rapide augmentation du nombre total de sommets. En réalité, les points de Steiner ne servent que dans un contexte de recherche de l'optimalité d'un certain critère, et donnent ainsi surtout des résultats théoriques. Il faut veiller à conserver une borne sur le nombre de ces points, car il est évident que, plus on en ajoutera, plus on se rapprochera de l'optimal (qui sera toujours atteind si on s'autorise l'ajout d'un nombre infini de points...). Les algorithmes qui utilisent des points de Steiner sont donc des algorithmes d'approximation.

BERN et EPPSTEIN [BE92] rappellent les principaux résultats connus pour des triangulations avec ajouts de points ; ces résultats sont surtout théoriques, et c'est pour cette raison que nous ne nous y attarderons pas trop.

Maximisation du plus petit angle

CHEW [Che89b] a exhibé un algorithme qui calcule une triangulation contrainte avec ajout de points, dans laquelle tous les angles sont de mesure comprise entre 30° et 120°. Aucune borne n'est donnée sur le nombre k de points de Steiner nécessaires ; il est juste montré que construire la triangulation se fait en temps $O(n+k)$.

BAKER, GROSSE et RAFFERTY [BGR88], ainsi que MELISSARATOS et SOUVAINE [MS92] ont eux aussi des algorithmes de triangulation avec points de Steiner qui bornent les angles des triangles.

BERN, EPPSTEIN et GILBERT [BEG90] montrent qu'il est possible, en utilisant un algorithme à base d'arbre quaternaire, de trianguler un ensemble $\mathcal{S}$ de points avec $O(m)$ triangles, où m est le nombre minimum de triangles nécessaires pour trianguler $\mathcal{S}$ si on désire que tous les angles soient de mesure supérieure à 20°.

Minimisation du plus grand angle

BERN et EPPSTEIN [BE91] montrent qu'il est possible de trianguler un polygone avec trous à n sommets avec $O(n^2)$ triangles non obtus. Puis, les mêmes auteurs plus DOBKIN [BDE95] montrent que tout polygone à n sommets peut être triangulé avec $O(n \log n)$ triangles dont les angles sont tous inférieurs à 150°.

Maximisation de la plus petite hauteur

Dans le même article, il est prouvé qu'un polygone avec trous peut être triangulé avec $O(n)$ triangles de hauteur $\Omega(d)$, où d est la distance minimale qui existe entre 2 points initiaux.

1.2.3.3 Amélioration par échange d'arêtes

FREY et FIELD proposent [FF91, FF92] de modifier la topologie d'un maillage pour la rendre plus régulière. En effet, pour de nombreuses applications en éléments finis, l'obtention de triangles équilatéraux (cas idéal) est souhaitable ; or, la plupart du temps, ce qui est fait pour parvenir à ce genre de résultats est de créer des points intérieurs au domaine de façon à obtenir le plus possible de triangles non-obtus [BGR88] ou aussi équilatéraux que possible [Fre87]. L'originalité dans leur méthode est de conserver un maillage triangulaire initial pour y réaliser des échanges d'arêtes judicieux, puis, dans une étape finale, lisser le résultat en employant le classique lissage de Laplace (voir paragraphe 1.2.3.4), c'est-à-dire en déplaçant certains sommets.

Une simple remarque de bon sens est à l'origine de leur processus : pour un point intérieur, la somme des angles en ce point pour tous les triangles dont il est un sommet doit faire 360°. Or un triangle équilatéral a des angles de mesures identiquement égales à 60° ; par conséquent, idéalement, un point devrait être sommet de 6 triangles. Par ailleurs, si le degré d'un point est inférieur ou égal à 4, il est impossible d'avoir des triangles avec des angles tous aigus.

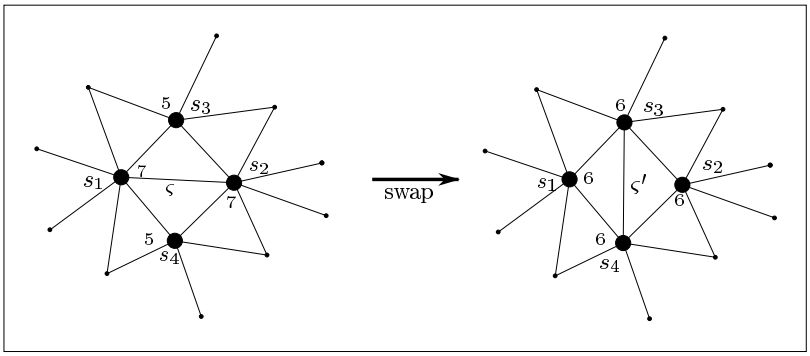
Arêtes à échanger

Pour une arête intérieure ς d'une triangulation donnée, on note s_1 et s_2 ses extrémités et s_3 et s_4 les sommets qui deviendront ses extrémités si elle est effectivement échangée en ς'(on suppose bien sûr que le quadrilatère $\square s_1 s_2 s_3 s_4$ est convexe). Ces sommets ont pour degrés respectifs $d_1^\circ, d_2^\circ, d_3^\circ$ et d_4° (sur la figure 1.19, ces degrés valent 7, 5, 7, 5 avant échange puis 6 après échange : on est bien là dans un bon cas pour effectuer le swap de ς) pour la configuration initiale, et $d_1^{\circ\prime}, d_2^{\circ\prime}, d_3^{\circ\prime}$ et $d_4^{\circ\prime}$ après échange de ς et ς'.

Reste à trouver le critère qui permettra de déterminer exactement quand effectuer un swap. Pour cela, les auteurs utilisent ce qu'ils appellent un *indice de relaxation* E_ς associé à l'arête donné par la formule :

$$\begin{aligned} E_\varsigma &= (d_1^\circ - 6) + (d_2^\circ - 6) - (d_3^\circ - 6) - (d_4^\circ - 6) \\ &= (d_1^\circ + d_2^\circ) - (d_3^\circ + d_4^\circ). \end{aligned}$$

Figure 1.19 – notations adoptées, et effet d'un échange sur les degrés des sommets.



Lorsque l'arête est échangé, on définit un indice de relaxation modifié de valeur :

$$\begin{aligned} E'_\varsigma &= (d_3^{\circ\prime} + d_4^{\circ\prime}) - (d_1^{\circ\prime} + d_2^{\circ\prime}) \\ &= (d_3^\circ + 1) + (d_4^\circ + 1) - (d_1^\circ - 1) - (d_2^\circ - 1), \end{aligned}$$

de sorte que :

$$E'_\varsigma = 4 - E_\varsigma.$$

Il est alors souhaitable d'échanger ς lorsque $E'_\varsigma < E_\varsigma$. Cette condition est vérifiée lorsque $E_\varsigma > 2$ et donne un résultat optimal si $E_\varsigma = 4$ (ce qui était le cas sur la figure 1.19). Il existe également des cas neutres lorsque $E_\varsigma = E'_\varsigma = 2$; ils bénéficient d'une étude spéciale dans les articles où il est expliqué quoi faire dans quels cas.

Comme la méthode consiste à effectuer des swaps et que l'on sait que l'ordre dans lequel ils sont réalisés influence fortement le résultat final (sauf lorsqu'ils aboutissent à une optimisation globale comme cela peut être le cas pour la triangulation de Delaunay), les auteurs suggèrent de commencer par les arêtes d'indice de relaxation $E_\varsigma \geqslant m$, où m a initialement une valeur supérieure à 4 et diminue pas à pas jusqu'à 3, et même 2.

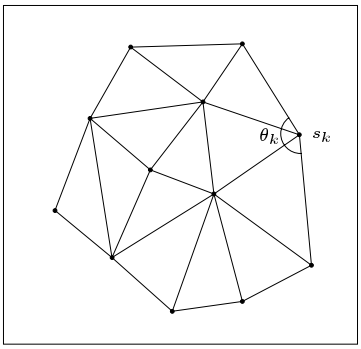
Traitement de la frontière

Évidemment, le degré idéal d'un sommet de la frontière de la triangulation est différent de 6. Soit s_k un tel sommet ; on suppose que l'angle total intérieur qu'il définit avec la frontière vaut θ_k (figure 1.20). Alors le degré idéal de s_k doit valoir $n_k + 1$ où n_k est le nombre entier de fois qu'un angle de 60° rentre dans θ_k. Une définition plus précise (fonction de θ_k) est donnée dans [FF92]. Pour calculer l'indice de relaxation d'une arête qui fait intervenir un sommet de la frontière, il suffira de remplacer son degré réel d_k° par la valeur $d_k^\circ + (6 - (n_k + 1))$.

Test de la méthode

L'exemple donné pour valider cette méthode est assez tendencieux ; les auteurs ont au moins l'honnêteté de le reconnaitre. En effet, le lissage de Laplace nécessite

Figure 1.20 – angle intérieur d'un sommet de la frontière.



des maillages assez réguliers sur les frontières pour bien fonctionner, et si possible un maillage convexe, et, dans leur exemple, les points sont disposés relativement uniformément aussi bien sur la frontière qu'à l'intérieur, ce qui facilite d'autant plus l'obtention de "bons" triangles. Leur résultat final est plutôt bon puisqu'ils obtiennent des triangles aux angles compris entre 37.84° et 94.73° en étant partis d'une triangulation de Delaunay où les angles valaient entre 31.08° et 117.84° (ce qui n'était quand même pas très mauvais...), résultat qui peut être encore amélioré si l'on décide d'effectuer des échanges pour certains cas neutres.

1.2.3.4 Déplacement des sommets

Lissage de Laplace

Apparu il y a maintenant près de 40 ans, le lissage de Laplace consiste à déplacer un sommet intérieur s du maillage au centre de gravité du polygone formé par ses voisins. Le sommet ne doit pas être déplacé s'il se retrouve en dehors du polygone (la triangulation ne serait alors plus conforme). En pratique, cette nouvelle position améliore souvent la taille et la forme des triangles autour de s, bien qu'il n'y ait aucune garantie d'un tel résultat ; c'est pour cela qu'il existe certaines méthodes faisant intervenir des poids pour chaque voisin (voir, par exemple, [Fie88, Her76]).

Barycentrage amélioré

L'apparition de poids non uniformes dans le calcul du barycentre du polygone est décrit dans l'ouvrage de GEORGE [Geo91]. Soient s le sommet que l'on désire déplacer, et $s_j, j = 1, \ldots, n$ les n voisins de s dans la triangulation ; soit ϵ_j le poids associé au sommet s_j (on doit avoir $\sum \epsilon_j = 1$). Alors, s sera repositionné au barycentre pondéré des s_j :

$$s' = \sum_{j=1}^{n} \epsilon_j s_j.$$

Une variante "relaxée" est donnée par le processus :

$$s^{k+1} = (1-\omega)s^k + \omega \sum_{j=1}^{n} \epsilon_j s_j^k,$$

où ω est un paramètre, et s^k la position de s obtenue après k itérations du processus.

TALON [Tal87] propose un algorithme un peu plus complet qui consiste à déplacer un sommet s, de manière itérative, afin d'optimiser un critère de forme qui tient compte du polygone donné par les voisins de s ; cela lui permet d'obtenir de bons résultats (du moins, meilleurs que ceux du barycentrage classique) lorsque le polygone n'est pas convexe.

1.2.4 Quelques techniques de construction particulières

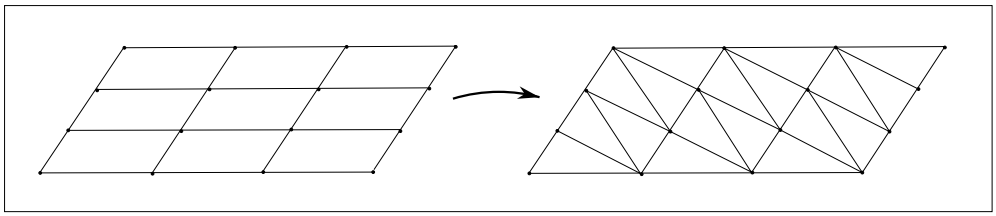
Nous allons voir ici quelques méthodes, que l'on pourrait qualifier d'"exotiques", qui cherchent à obtenir des triangles les plus allongés possibles, contrairement à ce qui est couramment désiré (des triangles les plus équilatéraux). Évidemment, l'utilisation de tels triangles apparait dans des domaines bien précis.

1.2.4.1 Utilisation de grilles structurées et de Delaunay

Dans [PM92], POSENAU et MOUNT expliquent que, dans le domaine de la dynamique des fluides (plus particulièrement pour l'aérospatiale), certaines caractéristiques des grilles structurées "rectangulaires" (en réalité, ce sont des grilles à base de parallélogrammes) sont très utiles, et devraient être conservées pour un maillage triangulaire : c'est notamment le cas pour le suivi des frontières d'un objet.

Cependant, il est connu que les triangles obtenus par Delaunay ont des propriétés qui les rendent très intéressants ; on voudrait donc pouvoir transformer un maillage structuré en triangulation de Delaunay tout en conservant l'orientation des cellules, puisqu'elle est réalisée suivant les frontières de l'objet. On peut voir sur la figure 1.21 un exemple de grille dont l'orientation n'est pas conservée par triangulation.

Figure 1.21 – un exemple où le passage du maillage structuré à la triangulation de Delaunay se fait en perdant l'orientation des cellules.



On considère donc que l'on a une grille structurée composée de rectangles de largeur l et de hauteur h. On suppose maintenant que l'on déforme ces rectangles comme sur la figure 1.22 ; on obtient des parallélogrammes de mêmes dimensions, et de même rapport d'aspect $\frac{l}{h}$. Si on note θ_i et θ_s les angles supportés par la

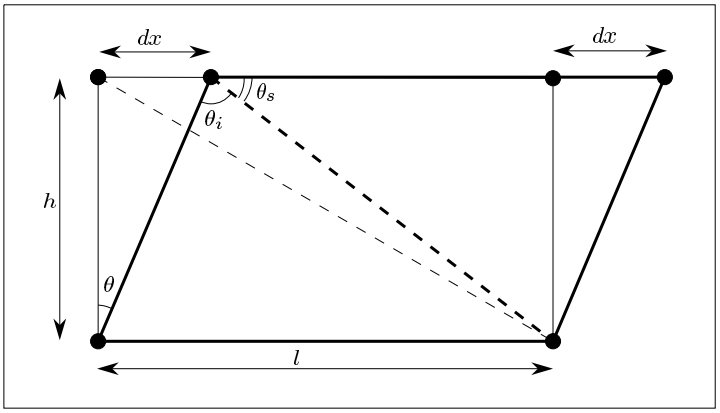
petite diagonale, θ l'angle qui induit la déformation sur le rectangle, et si on pose la définition suivante :

définition 1.17 *Un graphe $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ d'ensemble de sommets $\mathcal{S}$ et d'ensemble d'arêtes $\mathcal{A}$ est* **Delaunay inscriptible** *si $\mathcal{G}$ est un sous graphe de la triangulation de Delaunay de $\mathcal{S}$.*

POSENAU et MOUNT montrent que des bornes bien précises existent pour que les grilles structurées puissent se trianguler avec Delaunay en conservant leur orientation, autrement dit lorsque l'ajout des petites diagonales des parallélogramme donne la triangulation de Delaunay :

lemme 1.2 *tant que $\theta_i < 90°$, la grille structurée déformée est Delaunay inscriptible.*

Figure 1.22 – notations.



lemme 1.3 *si $\frac{l}{h} \leq 2$, la grille structurée déformée sera toujours Delaunay inscriptible.*

théorème 1.6 *lorsque le rapport d'aspect augmente, l'angle maximum θ pour lequel la grille reste Delaunay inscriptible diminue.*

1.2.4.2 Triangulations avec des triangles allongés

Intérêt de ces triangulations

Le point de vue théorique Dans son article [Rip92b], RIPPA discute d'abord des raisons pour lesquelles, pendant longtemps, les numériciens ont considéré que les triangles allongés étaient indésirables et qu'il vallait mieux des triangles pratiquement équiangulaires ; cela provient en grande partie de l'estimation de l'erreur qui est faite lors de l'interpolation de la fonction sur les triangles. En effet, une borne de l'erreur estimée, donnée par BRAMBLE et ZLAMAL [BZ70], contient un terme en

$\frac{1}{\sin\theta}$ qui devient gênant lorsque l'angle est petit. En utilisant d'autres bornes d'estimation de l'erreur, comme celles de BABUSKA et AZIZ [BA76] ou de GREGORY [Gre75], on a alors besoin que le plus grand angle d'un triangle n'approche pas π. Dans les deux cas, la constatation est la même : il faut éviter les triangles longs et fins.

Cependant, les tests de diverses méthodes d'approximation de fonctions (notamment celles qui font appel aux triangulations dépendantes des données [DLR90b], décrites page 38) montrent que de tels triangles peuvent permettre de diminuer de manière substantielle l'erreur d'interpolation. Dans son article, RIPPA montre qu'en réalité, la forme optimale d'un triangle (optimale au sens de la minimisation de l'erreur L_2 pour l'approximation linéaire par les sommets d'un triangle) est liée à la fonction $f(x, y)$ que celui-ci approche : globalement parlant, cette forme optimale doit être longue dans la direction de la plus petite dérivée seconde de f et petite dans la direction de la plus grande dérivée seconde. Les conditions exactes concernant les notions de "grande" et "petite" sont données dans l'article.

La méthode des éléments finis À la suite de recherches liées directement à ce qu'a prouvé RIPPA, des numériciens se sont aperçus que même dans la méthode des éléments finis, il peut arriver que l'on ait besoin de triangles aplatis : c'est notamment vrai lors de l'étude de l'écoulement d'un fluide autour d'un objet. En effet, au voisinage de l'objet, les forces qui entrent en jeu ont une direction très privilégiée, ce qui fait qu'une grosse différence apparaît au niveau des valeurs de leurs dérivées. Dans ce cas précis, l'utilisation de triangles aplatis près de l'objet autour duquel coule le fluide s'avère indispensable.

Tout ceci prouve bien la nécessité de la mise en place d'algorithmes de triangulation faisant intervenir, dans des cas opportuns, des triangles allongés.

Des méthodes

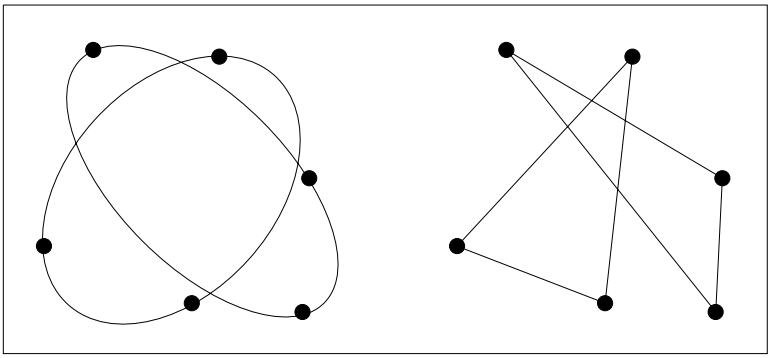
Utilisation de la triangulation de Delaunay Ainsi, MAVRIPLIS [Mav90] préconise l'utilisation d'un maillage non structuré (c'est-à-dire, en dimension 2, une triangulation) à éléments de forme très allongée dans les régions frontières où il y a écoulement du fluide ; il préconise pour cela l'utilisation d'une triangulation de Delaunay modifiée permettant d'aboutir à des triangles "déformés" dans la direction voulue selon l'amplitude désirée.

Un vecteur d'allongement, composé d'une direction et d'une intensité, est défini en chacun des points du maillage initial qui n'est rien d'autre que la triangulation de Delaunay standard ; ce vecteur dépend de la force exercée en son point d'application. Une surface de contrôle est alors construite à partir des données de tous les vecteurs d'allongement, de telle sorte que lorsqu'on relève les points de l'espace physique sur la surface, on se retrouve avec une distribution pratiquement uniforme : les relevés sont à peu près équidistants les uns des autres dans toutes les directions en terme de longueurs d'arcs, alors que les points de l'espace physique étaient éparpillés suivant une direction préférentielle. Il suffit maintenant de construire la triangulation de Delaunay dans l'espace de contrôle et de reprojeter le tout dans l'espace physique (plan réel du maillage à obtenir), faisant apparaître ainsi des triangles allongés aux endroits désirés.

La principale difficulté de cet algorithme réside dans la construction de la surface de contrôle ; pour réussir à la définir, on a besoin de connaitre plus de points dans les régions où il y aura écoulement de fluide (les régions où une variation locale du vecteur d'allongement peut être très importante). Les sommets choisis au départ dépendront donc de la géométrie du problème et seront générés par utilisation d'une grille spécifique.

Utilisation du relèvement sur le paraboloïde Faisant suite à cet article, POSENAU dans [Pos93] remarque que la méthode de MAVRIPLIS est fortement reliée à la façon dont sont engendrés les points de départ. En effet, si l'on ne dispose que d'un ensemble de sommets éparpillés, la fonction distance définie sur la surface de contrôle peut varier rapidement et aboutir à des triangles, non valides, qui se couperont (figure 1.23 tirée de [Pos93]).

Figure 1.23 – une fonction distance qui donne une triangulation non valide.



Afin de rendre l'approche plus robuste (dans le sens où elle sera indépendante de l'ensemble de points choisi), l'auteur propose un autre moyen de parvenir à une triangulation plus "étirée" que celle de Delaunay, fondé sur une des constructions possibles de la triangulation : le relèvement sur le paraboloïde unité. Il est en effet possible de modifier l'aspect des triangles en changeant le convexe sur lequel le relèvement est effectué (la surface choisie pourra être par exemple dépendante de la fonction d'écoulement du fluide).

L'idée est la suivante : soit $\mathcal{F}$ une surface convexe d'équation $z = f(x, y)$, (x_0, y_0) un point du maillage, et $z = \lambda x + \mu y + \nu$ le plan tangent à $\mathcal{F}$ qui passe par $(x_0, y_0, f(x_0, y_0))$. Si on déplace d'une distance υ le plan à l'intérieur du convexe, son intersection avec $\mathcal{F}$ est une courbe qui, projetée dans le plan de base, donnera la forme de l'objet circsconscrit à un simplexe. Par exemple, si la surface est le paraboloïde unité standard, la forme en question est un cercle. Si on choisit une surface un peu différente, on pourra obtenir des formes diverses à direction privilégiée :

- la surface $f(x, y) = x^2 + 10y^2$ donnera des ellipses de grand axe dirigé suivant les abscisses,
- des surfaces comme $f(x, y) = x^2 + y^4$ ou $x^3 + y^3$ donneront respectivement des formes de largeur constante suivant une direction ou d'orientation dominante changeante.

Une approche un peu différente consiste à modifier la méthode de projection des points sur le paraboloïde ; au lieu d'utiliser une simple projection orthogonale, on peut avoir recours à la perspective relative à un point $(0, 0, z_{proj})$. Le problème est qu'il est alors possible de rater la surface lors du relèvement des sommets par la perspective ; il faut donc choisir un z_{proj} suffisamment distant, ou effectuer la perspective relative à un point qui soit à l'"intérieur" de $\mathcal{F}$.

1.3 Triangulations et surfaces

Les triangulations sont très souvent utilisées pour représenter une surface, et cela dans de nombreux domaines, qui ont en commun de vouloir obtenir une structure adaptée au graphisme sur ordinateur, mais aussi simplifiée de façon à être en mesure d'effectuer certains calculs. Parmi ces domaines, on peut citer la cartographie, la géologie [Nul95], l'architecture, l'imagerie médicale [Gei93]....

Cette très grande utilisation des triangulations peut s'expliquer par le fait qu'elles permettent de définir, de manière naturelle, une approximation linéaire à partir d'un ensemble de points de données, et peuvent ainsi servir à décrire la géométrie d'une surface. Leur côté linéaire simplifie certaines opérations telles que la recherche de contours ou le calcul de volumes. Évidemment, l'utilisation des triangulations soulève d'autres problèmes ; suivant les applications, il peut arriver que l'on dispose de points soit trop éparpillés pour représenter correctement la surface, soit, au contraire, trop nombreux pour que l'on puisse raisonnablement tous les prendre en considération, ou encore que l'on doive définir des arêtes entre les sommets afin de représenter une surface inconnue... Des réponses à certains des problèmes les plus courants, qui surviennent avec les triangulations 2D ou 2D-$\frac{1}{2}$ (on utilise ce terme pour désigner l'utilisation d'une triangulation planaire pour des objets plongés dans un espace de dimension 3) sont données dans cette section.

1.3.1 Les triangulations dépendantes des données

Le problème posé dans l'article de DYN, LEVIN et RIPPA [DLR90b] est de trianguler le mieux possible un ensemble $\mathcal{S}$ de points qui, en plus de leurs coordonnées, possèdent une valeur dépendant d'une fonction f définie de $\mathbb{R}^2$ dans $\mathbb{R}$; ces points sont ainsi les représentants choisis de f. Dans ce genre d'approche, au lieu de s'intéresser à la distribution des points dans le plan (comme c'est le cas pour la triangulation de Delaunay), on va essayer d'effectuer une triangulation qui dépend de la valeur prise par la fonction en ces points. Cette méthode s'avère en général très avantageuse dès lors qu'il s'agit d'approximations de surfaces.

L'interpolant de la fonction est défini comme une approximation linéaire par morceaux - polynômes de degré 1 -, un morceau étant un triangle. On supposera que l'on ne connait que les valeurs de la fonction en les points, et pas de données supplémentaires (comme le gradient par exemple, auquel cas il existe d'autres techniques). Une fonction linéaire est définie de manière unique par ses valeurs aux trois sommets du triangle. Comme il existe plusieurs façons de trianguler le domaine de départ, ici l'intérêt des auteurs s'est reporté sur des critères d'optimalité locale faisant intervenir les valeurs de la fonction représentée. Leur motivation est que cette

fonction peut être regardée comme une surface dont on veut obtenir une bonne approximation.

Pour parvenir à une triangulation localement optimale (au sens de l'approximation de la surface $\mathcal{F}$ d'équation $z = f(x, y)$), il suffira que chacune de ses arêtes soit localement optimale : en échangeant une arête avec l'autre diagonale du quadrilatère qui la contient dans la triangulation, on ferait augmenter la valeur du critère qu'il faut minimiser (tous les détails de cette technique sont présentés au chapitre 2). On voit bien ici l'importance que revêt le choix du critère ; il est improbable qu'un seul critère convienne à tous les types d'ensembles de points et de fonctions, mais il faudra plutôt en choisir un qui convienne à nos données. Ainsi, dans leur article, les auteurs définissent un certain nombre de critères possibles, dont les plus intéressants s'appliquent aux triangulations planes et supposent que la fonction à approcher ne varie pas trop entre les points donnés ; ces critères, liés directement aux arêtes communes à deux triangles adjacents sont ce qu'ils appellent des critères du type "Nearly C^1".

1.3.1.1 Coûts associés aux arêtes

À chaque arête ς_i, $i = 1, \ldots, n$ située à l'intérieur d'une triangulation $\mathcal{T}$ de l'enveloppe convexe Ω de $\mathcal{S}$ est associée une fonction coût qui dépend de la fonction f à approcher et de l'approximation linéaire $P_{\mathcal{T},f}$ définie par $\mathcal{T}$; ce coût sera noté coût$(P_{\mathcal{T},f}, \varsigma_i)$. Le coût total de la triangulation sera alors

$$\text{Coût}(P_{\mathcal{T},f}) = \sum_{i=1}^{n} \left|\text{coût}(P_{\mathcal{T},f}, \varsigma_i)\right|.$$

On notera Δ_1 et Δ_2 deux triangles adjacents de $\mathcal{T}$ d'arête commune ς, Π_1 et Π_2 les plans définis par l'interpolation linéaire de f sur Δ_1 et Δ_2, et $\vec{n_1}$ et $\vec{n_2}$ leurs normales respectives orientées suivant les z positifs. Quatre fonctions de coût sont proposées [DLR90b] ; nous donnons, ci-dessous, les deux fonctions qui produisent les meilleurs résultats :

– l'*angle entre les normales* ou A.B.N. ; elle représente la valeur de l'angle aigu θ entre les vecteurs $\vec{n_1}$ et $\vec{n_2}$:

$$\text{coût}(P_{\mathcal{T},f}, \varsigma) = \theta = \arccos\left(\frac{\vec{n_1}.\vec{n_2}}{||\vec{n_1}||.||\vec{n_2}||}\right).$$

– le *saut entre les dérivées normales* ; si $\vec{n}$ est un vecteur unité de $\mathbb{R}^2$ orthogonal à la direction de ς et $\vec{n_i}'$ $(i = 1, 2)$ le vecteur formé des 2 premières composantes de $\vec{n_i}$, cette fonction coût s'exprime sous la forme :

$$\text{coût}(f_{\mathcal{T}}, \varsigma) = \left|\vec{n}.(\vec{n_1}' - \vec{n_2}')\right|.$$

Associées à ces fonctions de coût sont testées 3 normes (normes L_1, L_2 et L_∞ "lexicographique" qui tient compte de l'erreur maximum commise non pas seulement

pour le triangle le pire mais pour tous les triangles) ; un couple fonction de coût - norme définit alors un critère.

1.3.1.2 Algorithmes

Dans [DLR90a], DYN, LEVIN & RIPPA décrivent de manière plus détaillée les algorithmes mis en place pour construire des triangulations dépendantes des données. À la base se trouve ce qu'ils appellent la *procédure d'optimisation locale*, algorithme général de construction de triangulations par échanges de diagonales, directement inspiré par le travail de LAWSON [Law77] :

- **initialisation** : soit $\mathcal{T}$ une triangulation initiale de l'enveloppe convexe des points

- **tant que** $\mathcal{T}$ n'est pas localement optimale **faire**
  pour toute arête ς de $\mathcal{T}$ non localement optimale
  $\mathcal{Q} \leftarrow$ quadrilatère strictement convexe formé des deux triangles ayant ς pour arête commune
  on **remplace** ς dans $\mathcal{T}$ par l'autre diagonale de $\mathcal{Q}$.
  fin pour

  fin tant que

Les auteurs montrent alors, à grand renfort de tests numériques, que l'ordre dans lequel les arêtes sont échangées peut avoir une grande influence sur la triangulation finale d'un point de vue aussi bien qualitatif (forme des triangles) que quantitatif. Ils proposent donc deux stratégies :

– une stratégie dite de *réduction maximale*, qui choisit l'arête qui en étant swappée va donner la triangulation dont le coût sera le plus faible possible (parmi toutes les triangulations que l'on peut obtenir en échangeant une seule arête),

– une stratégie qui consiste à échanger d'abord les arêtes qui laissent le plus possible de quadrilatères convexes pour l'itération suivante de la boucle d'optimisation locale (ce qui revient à commencer par des arêtes qui feront apparaître plus de nouvelles possibilités).

Pour bien montrer l'importance de choisir une bonne stratégie, ils essaient aussi d'effectuer les échanges dans un ordre quelconque, ou suivant une stratégie de réduction *minimale* qui donnera évidemment de mauvais résultats. Leur conclusion est que les deux "bonnes" stratégies sont celles qui fournissent en général les meilleurs résultats ; ils conseillent donc de choisir la première car elle est beaucoup plus facile à mettre en place que la seconde. Il faut cependant noter que, comme tout ce qui touche aux triangulations dépendantes des données, aucune des stratégies n'est la meilleure dans tous les cas.

1.3.1.3 Coûts associés aux sommets

BROWN [Bro91] associe plutôt un coût à chaque sommet s d'une triangulation $\mathcal{T}$, en utilisant des valeurs qui dépendent des triangles ayant s pour sommet. Il essaie alors de modifier $\mathcal{T}$ pour diminuer sa fonction de coût.

L'algorithme utilisé est une nouvelle fois une adaptation de la procédure d'optimisation locale de LAWSON (on suppose que l'on part d'un ensemble de n points $\{s_1, \ldots, s_n\}$) :

- **initialisation** : soit $\mathcal{T}$ une triangulation initiale de l'enveloppe convexe des points

- **répéter**
  repère $\leftarrow 0$
  pour $i = 1, \ldots, n$
    $s \leftarrow s_i$
    pour toute arête ς intérieure à $\mathcal{T}$ de sommet s
      si ς n'est pas localement optimale **alors**
        on **échange** ς
        repère $\leftarrow 1$
      fin si
    fin pour
  fin pour

  jusqu'à repère $= 0$

Parmi toutes les fonctions de coût qu'il a testées, l'auteur en a retenu une dont les résultats sont presque toujours meilleurs que ceux observés avec les fonctions de coût associé aux arêtes données dans [DLR90b] ; soit s un sommet d'une triangulation $\mathcal{T}$. Soient $\vec{n_i}, i = 1, \ldots, m$ les normales aux triangles de $\mathcal{T}$ dont un sommet est s. Si on pose $\vec{n} = \sum_{i=1}^{m} \frac{\vec{n_i}}{||\vec{n_i}||}$ ($\vec{n}$ peut être considéré comme un équivalent linéaire par morceaux au vecteur normal à $\mathcal{T}$ en s), alors la fonction coût est définie par :

$$\text{coût}(P_{\mathcal{T},f}, s) = \sum_{i=1}^{m} \left(\arccos \left(\frac{\vec{n}.\vec{n_i}}{||\vec{n}||.||\vec{n_i}||} \right) \right)^2 .$$

Une remarque intéressante est que si les vecteurs normaux utilisés ne sont pas normalisées, les résultats observés se trouvent dégradés. Une autre remarque : n'importe quelle triangulation localement optimale basée sur un coût associé aux arêtes peut être obtenue en utilisant un coût associé aux sommets.

1.3.2 Méthodes globales de reconstruction d'une surface

En général, les triangulations interviennent dans les reconstructions de surfaces définies comme fonctions au sens mathématique du terme, c'est-à-dire où z est donné par $z = f(x, y)$. Mais, il peut arriver que l'on désire représenter un objet à partir de données 3D par une triangulation, avec comme problème sous-jacent, la propriété qu'un même couple (x, y) puisse avoir plusieurs z. La grande différence entre

une triangulation 2D-$\frac{1}{2}$ et une triangulation 3D est que la première représente une partition du domaine choisi ,alors que la seconde impose une topologie parmi les points 3D qui implique la construction d'un polyèdre, approximation de la surface. Une question essentielle est alors de savoir quel est le meilleur polyèdre possible ; les méthodes décrites dans cette section essaient de fournir une réponse à ce problème.

1.3.2.1 Projection sur un plan tangent

En 1984, BOISSONNAT [Boi84] a proposé un algorithme permettant de construire une approximation polyédrique d'une surface dont la seule connaissance réside dans la donnée d'un ensemble de points de l'espace $\mathcal{S} = \{S_1, \ldots, S_n\}$. Cet algorithme utilise, dans le voisinage d'un sommet S_i, la projection orthogonale sur un plan tangent à $\mathcal{S}$ en S_i. Après avoir montré certaines propriétés qui serviront de base théorique à sa méthode, l'auteur définit sa notion de voisinage de S_i comme l'ensemble des k plus proches voisins du point, ensemble qui peut être déterminé par l'algorithme de FRIEDMAN, BENTLEY et FINKEL [FBF77] en temps logarithmique, en supposant qu'une structure d'arbre (le "k-D tree") ait été construite (en temps $O(n \log n)$) ; le k en question est alors un paramètre de l'algorithme, qui est surtout lié au nombre total de points à insérer dans la triangulation. Différents ensembles sont alors utilisés :

- $\mathcal{D}$, l'ensemble des triangles déjà créés (qui représente le domaine triangulé),
- $\mathcal{C}$, le contour du domaine triangulé (stocké comme une liste doublement chaînée de points),
- $\mathcal{O}$, l'ensemble des points qui n'apparaissent pas encore dans $\mathcal{D}$.

L'initialisation donne une première arête en reliant un des sommets de $\mathcal{S}$ à son plus proche voisin ; $\mathcal{C}$ contient alors les 2 segments ayant pour extrémités ces points-ci. $\mathcal{D}$ est initialisé en prenant 2 points (qui ne sont pas dans $\mathcal{S}$) dans le plan tangent approché à la surface au voisinage de l'arête, donné par une méthode de moindres carrés. La triangulation se developpe ensuite par propagation du contour.

Soit ς une arête de $\mathcal{C}$ d'extrémités S_i et S_j (on suppose que S_j est le successeur de S_i dans $\mathcal{C}$) ; soit Π le plan tangent approché au voisinage de ς. On projette sur Π les k plus proches voisins de S_i et S_j qui appartiennent à $\mathcal{O}$; si on note S_i' le point à l'intérieur de $\mathcal{D}$ qui complète le triangle formé de S_i et S_j, alors on choisira, parmi les voisins de projetés sur Π qui se trouvent du côté opposé à S_i' par rapport à la droite (S_iS_j), le point S_k dont le projeté voit la projection de ς sous le plus grand angle. On ajoute alors S_k à $\mathcal{C}$ et le triangle $\Delta S_iS_jS_k$ à $\mathcal{D}$; puis, éventuellement, on met à jour $\mathcal{O}$ et on poursuit le processus jusqu'à ce qu'aucun triangle ne puisse plus être créé.

Il faut noter que cet algorithme, de complexité $O(n \log n)$, ne garantit l'obtention d'une approximation polyédrique que si la discrétisation initiale est assez fine (la notion de *finesse* étant liée au plus petit rayon de courbure en un point).

Une méthode due à HOPPE, DEROSE, DUCHAMP, MCDONALD et STUETZLE [HDD+92] reprend un peu les mêmes idées, à savoir la projection des sommets sur des plans tangents approchés correctement orientés (sinon, on risque d'avoir des

problèmes dans le rendu de l'image finale) et l'utilisation des k plus proches voisins. Une étape finale consiste à modifier certaines arêtes, tout en conservant la topologie, pour les triangles ayant un mauvais rapport d'aspect.

1.3.2.2 Polyèdres d'aire minimale

O'Rourke [O'R81] a proposé d'utiliser des polyèdres d'aire minimale pour trianguler un ensemble de points $\mathcal{S}$. L'auteur montre qu'un tel polyèdre existe toujours pour tout ensemble de points de $\mathbb{R}^3$ en configuration générale et qu'il n'est pas forcément unique ; il suggère également que trouver ce polyèdre est sans doute un problème NP-dur, car son équivalent en dimension 2 (polygone de longueur minimale) l'est.

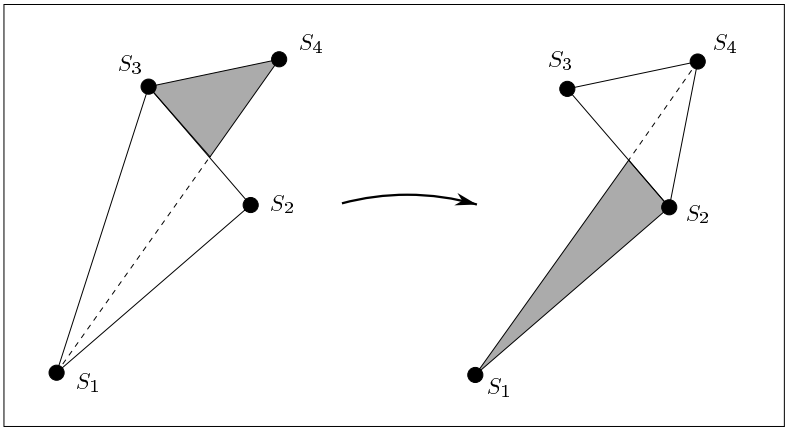
Son algorithme peut être résumé de la manière suivante :

- **initialisation :** soit $\mathcal{P}$ un polyèdre défini par l'enveloppe convexe de $\mathcal{S}$

- **tant qu'** il reste des points de $\mathcal{S}$ à l'intérieur de $\mathcal{P}$ **faire**
  pour tout point S intérieur à $\mathcal{P}$
  $\tilde{S} \leftarrow$ projection de S sur une face $\tilde{f}$ de $\mathcal{P}$ telle que $\tilde{S}$ est à l'intérieur de la face, et telle que la distance euclidienne $S\tilde{S}$ est minimale
  $\upsilon_S \leftarrow$ facteur de modification de l'aire provoquée par la substitution de $\tilde{f}$ par les 3 faces contenant S et une arête de $\tilde{f}$
  fin pour
  $\mathcal{S}' \leftarrow$ sommet S tel que υ_S soit minimal
  $\mathcal{P} \leftarrow$ polyèdre obtenu en remplaçant $\tilde{f}'$ par les 3 faces qui contiennent S' et une arête de $\tilde{f}'$
  si $\mathcal{P}$ perd son caractère minimal **alors**
  on **effectue** la transformation de la figure 1.24
  fin si

  fin tant que

Figure 1.24 – transformation ("swap") destinée à "améliorer" le polyèdre.



La dernière transformation de l'algorithme n'est effective que s'il y a une réelle "amélioration" du polyèdre, amélioration subjective qui tient compte de plusieurs caractères géométriques de $\mathcal{P}$.

Comme tout algorithme basé sur des heuristiques, on n'est pas sûr d'aboutir à l'optimum global, et O'ROURKE note même que son algorithme peut être complètement trompé par certains jeux de données.

1.3.2.3 Obtention de surfaces lisses approchées

CHOI, SHIN, YOON et LEE [CSYL88] proposent un algorithme, finalement assez proche de celui de BOISSONNAT, pour lequel ils rappellent que les schémas d'interpolation destinés à construire des surfaces lisses et basés sur des triangles consistent en 3 étapes :

- on construit d'abord une grille triangulaire en reliant les points de données.
- on construit alors un réseau de courbes lisses s'appuyant sur les arêtes de la triangulation.
- enfin, on construit une surface composite lisse de "patches" de Bézier (par exemple) par interpolation du réseau de courbes.

Pour construire leur triangulation, en partant d'un triangle initial, ils ajoutent un par un des triangles tels que les arêtes du contour de la grille maintiennent un polygone convexe tout en conservant les points non triangulés en dehors de ce polygone. L'originalité dans leur algorithme réside dans le recours à une phase de lissage pour améliorer la triangulation.

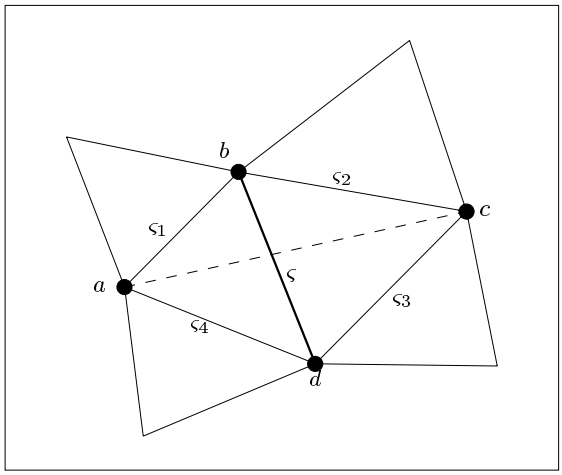
Pour une arête ς donnée de la triangulation, le critère de lissage fait appel aux angles des 4 arêtes bornant le quadrilatère convexe (éventuel) dont ς est la diagonale. L'angle d'une arête est l'angle entre les 2 triangles qui sont de part et d'autre de l'arête. D'après les notations de la figure 1.25 où $\square abcd$ est le quadrilatère dont ς est la diagonale, le critère choisira la diagonale qui maximise le minimum des 4 produits scalaires des vecteurs normaux unitaires des triangles, un produit scalaire étant associé à chaque arête ς_i de $\square abcd$, lorsque le quadrilatère est strictement convexe.

1.3.3 Modification d'une triangulation

Dans des domaines tels que la cartographie (représentation de terrains) ou la vision scientifique (représentation de surfaces), les modèles proposés consistent souvent en des ensembles de milliers, voire de millions, de points. Il est alors irréaliste de vouloir construire une triangulation en incorporant la totalité des sommets.

Un certain nombre d'articles proposent donc des algorithmes qui vont faire en sorte de sélectionner, d'une manière ou d'une autre, les meilleurs représentants pour approcher l'objet initial ; nous allons voir dans cette section certaines des méthodes qui existent pour construire des triangulations en n'utilisant pas tous les points de données.

Figure 1.25 – notations pour le critère de lissage.



1.3.3.1 Raffinement

Les méthodes de raffinements suivent en général le même schéma ; elles commencent avec une approximation initiale, et ajoutent à chaque itération un ou plusieurs points comme sommets de la triangulation, jusqu'à ce qu'un critère soit respecté (les critères les plus communs étant d'arriver à un certain nombre de points, ou atteindre une borne pour l'erreur mesurée). Pour éviter de recalculer entièrement une triangulation à chaque étape, les méthodes incrémentales sont typiquement utilisées, et Delaunay est la triangulation la plus fréquemment employée.

Lorsque l'on cherche une triangulation d'un ensemble de points possédant une élévation propre (en plus de leurs coordonnées planaires) et qui représentent ainsi un relief, la fonction de mesure de l'erreur la plus communément utilisée est la valeur absolue de la différence de hauteur entre l'approximation et le champ élévation de chaque point.

Insertion gloutonne

Un algorithme d'insertion gloutonne est un algorithme qui va ajouter de façon irréversible un ou plusieurs sommets à la triangulation ; si on n'ajoute qu'un seul sommet l'algorithme sera dit *séquentiel*, et *parallèle* sinon.

Méthode de Garland et Heckbert GARLAND et HECKBERT [GH95] proposent divers algorithmes de ce type ; plusieurs mesures ont été testées pour les erreurs parmi lesquelles l'erreur locale d'élévation (la plus courante), une erreur relative à la courbure (pour accentuer les efforts au niveau des pics, des vallées ou des falaises dans un terrain), l'erreur globale d'élévation (somme de toutes les erreurs locales) et même des erreurs mixtes, produits de différents critères. Une première conclusion : c'est l'erreur standard toute simple qui donne en général les meilleurs résultats, et toujours pour le coût le plus faible (en temps machine).

L'algorithme séquentiel de base pour une insertion gloutonne est le suivant :

- **initialisation** : maillage comprenant deux triangles qui couvrent toute le terrain

- **tant que non** *but atteint* **faire**
  $erreur_{max} \leftarrow 0$
  pour tout point de départ s
  si erreur en $s > erreur_{max}$ **alors**
  $erreur_{max} \leftarrow$ erreur en s
  $choix \leftarrow s$
  fin si
  fin pour
  on **insère** *choix* dans la triangulation

  fin tant que

Le but à atteindre peut être choisi parmi les plus standards (nombre de points ou borne de l'erreur L_2 ou L_∞). La complexité temporelle de cet algorithme s'il doit choisir m points parmi n est de $O(m^2n)$, mais son coût moyen réel est de $O(mn)$. Une légère amélioration, qui tient compte du fait qu'à chaque itération il y a très peu de points dont l'erreur ait changé, permet d'aboutir à un coût moyen de $O(n \log m)$. En changeant également les structures de données pour conserver les meilleurs points à insérer dans une pile, on peut avoir une complexité dans le pire des cas de $O(mn)$.

Les auteurs testent alors deux versions de leur algorithme : une qui utilise la triangulation de Delaunay, l'autre qui fait appel aux triangulations dépendantes des données, où une arête est échangée si cela permet de faire décroitre l'erreur. La deuxième, bien qu'intéressante, fait apparaître des triangles très allongés qui peuvent être gênants lors de leur affichage. Pour résoudre le problème, GARLAND et HECKBERT ont eu recours à une triangulation hybride qui utilise le critère du cercle vide lorsque des angles trop pointus risquent d'apparaître et celui de décroissance de l'erreur sinon.

Leurs résultats montrent que l'utilisation des triangulations dépendantes des données permet une diminution de l'erreur d'environ 12% par rapport à Delaunay, mais au prix d'un coût assez important : pour arriver à une borne donnée de l'erreur, il faut 3 ou 4 fois plus de temps.

Enfin, une remarque très intéressante de leur article concerne l'utilisation des algorithmes d'insertions gloutonnes parallèles. En effet, leurs tests prouvent que ces algorithmes fonctionnent beaucoup moins bien, en terme de qualité, que leurs équivalents séquentiels. Leur explication est la suivante : lors d'une insertion parallèle, des petits triangles peuvent être sélectionnés, et un effet boule de neige entraine une subdivision excessive dans certaines zones. La triangulation finale est alors bien moins équilibrée, et les raffinements ne sont plus obtenus aux endroits stratégiques (un exemple frappant est d'ailleurs donné dans leur article).

D'autres algorithmes d'insertion gloutonne Historiquement, le premier algorithme séquentiel d'insertion gloutonne semble venir de DE FLORIANI, FALCIDIENO et PIENOVI [DFP85], puis a été repris et amélioré par de nombreux auteurs

(en plus de GARLAND et HECKBERT) parmi lesquels on pourra citer DE FLORIANI [Flo89] et DE FLORIANI, PUPPO [FP92] où en plus une structure hiérarchique est utilisée. Pour les versions parallèles de l'insertion avec une comparaison sur l'utilisation de Delaunay ou des triangulations dépendantes des données, le travail de POLIS et MC KEOWN [PK93] semble intéressant.

Un article de RIPPA [Rip92a] est lui aussi intéressant ; il généralise l'algorithme de DE FLORIANI [DFP85] en utilisant les triangulations dépendantes des données (permettant de conserver certains reliefs très prononcés) à la place de Delaunay ainsi que des méthodes de moindres carrés. Dans sa boucle d'optimisation à base d'échanges d'arêtes, il teste deux définitions d'erreur globale : la somme des valeurs absolues des angles entre les normales de toutes les paires de triangles adjacents (c'était le critère A.B.N. de [DLR90b]), et la somme des carrés des erreurs d'élévation verticale pour tous les points de données (ce qui correspond à une erreur L_2 discrétisée). Pour des fonctions relativement lisses, RIPPA observe que la triangulation dépendante des données utilisant le critère des normales donne pratiquement toujours de meilleurs résultats que Delaunay, et que le critère d'erreur L_2 peut aboutir à des triangles trop allongés (et qui peuvent poser des problèmes pour une procédure d'ombrage lors d'un affichage) qu'il est impossible d'éliminer car ils ne contiennent aucun point de donnée et donc ont une erreur L_2 nulle. Pour les cas où le critère de l'angle n'était pas satisfaisant, l'auteur a testé un schéma hybride qui compare à chaque itération l'erreur obtenue avec Delaunay avec celle utilisant le critère A.B.N., et qui choisit la meilleure des deux. Les approximations finales sont meilleures mais au prix d'une certaine augmentation du temps d'exécution.

Une autre méthode essayée par RIPPA est basée sur les moindres carrés pour approcher les points initiaux au lieu de les interpoler. Seule l'élévation peut varier, et une combinaison de hauteurs qui minimise la somme globale des erreurs carrées est recherchée. Si ce processus est exécuté après avoir déclenché une insertion gloutonne standard, on peut observer une diminution de moitié de l'erreur pour un coût raisonnable (résolution d'un système $m \times m$ creux).

Utilisation d'un schéma d'interpolation

Si un ensemble de points appartenant à une même surface est trop éparpillé, il peut arriver que l'on ait besoin de lisser la représentation triangulée de cette surface en ajoutant de nouveaux points ; le problème est donc un peu différent de ce que l'on vient de voir, dans la mesure où ce que l'on veut ici, c'est ajouter des points à l'ensemble initial. C'est ce que font JONES et WRIGHT [JW91] ; leur méthode peut être qualifiée de semi-automatique, dans le sens où l'utilisateur doit choisir lui-même un schéma d'interpolation lisse (pas forcément linéaire) susceptible de bien convenir à la surface : six schémas usuels sont proposés. Le but de leur algorithme est donc d'ajouter des points à la triangulation (identifiée à une approximation linéaire de la surface) pour coller au plus près au schéma choisi. Évidemment, si les sommets initiaux ont été mal positionnés, la représentation finale pourra demeurer mauvaise. Un triangle sera jugé bon si le maximum de la différence d'élévation entre l'approximation qu'il définit et la surface donnée par le schéma est faible. Le point de plus forte déviation apparaissant à une étape donnée sera introduit dans la triangulation, et le procédé sera itéré jusqu'à un certain seuil de tolérance. Un

autre critère de jugement peut être de calculer le volume qui sépare un triangle de la surface.

Comme d'habitude, la triangulation de départ est celle de Delaunay, et sa mise à jour lors de l'insertion d'un nouveau sommet est effectuée en réalisant des échanges d'arêtes ; ces swaps sont effectués, soit en suivant le critère du cercle vide (pour retomber sur une triangulation de Delaunay), soit en fixant pour critère la diminution du volume de séparation des triangles obtenus avec la surface du schéma. C'est en utilisant cette dernière méthode que l'on observe des résultats un peu meilleurs, mais au prix d'une assez forte augmentation du temps CPU.

1.3.3.2 Décimation

Le principe de ces méthodes est de partir d'une triangulation pour la simplifier jusqu'à arriver au niveau d'approximation voulu. Il existe plusieurs catégories d'algorithmes de décimation : élimination de sommets, élimination d'un triangle (et de ses 3 arêtes, en prenant garde de toujours bien conserver une triangulation valide), élimination d'un groupe de triangles adjacents.

Ces algorithmes sont d'autant plus utiles que les triangulations sont maintenant très employées pour la représentation des terrains, car elles sont peu coûteuses en place mémoire et facile à mettre en place même pour des ensembles de points éparpillés. Or, à la base, un terrain est donné par une grille assez importante de points, que l'on aurait besoin de simplifier le plus possible pour réussir à l'utiliser ensuite de manière agréable. De même, les modèles engendrés par des mesures de volumes (grâce à des scanners lasers ou des satellites) sont typiquement sur-échantillonés ; si on les sauvegarde tels quels (au lieu de leurs approximations), on aura besoin d'une grande quantité de stockage et de beaucoup de temps de calcul pour les manipuler ensuite.

La difficulté de donner de bons algorithmes de décimation peut être soulignée par l'article de Agarwal et Suri [AS94], où il est prouvé que calculer une approximation linéaire, qui n'est jamais éloignée de plus d'un facteur ϵ de la surface donnée par un ensemble de points, et qui soit de complexité minimum (c'est-à-dire contienne le moins possible de points, d'arêtes ou de faces), est un problème dont la version décisionnelle est NP-dure. On sait ainsi que seule une approximation du meilleur résultat peut être donnée, et les auteurs proposent eux-mêmes une surface approchante de taille $O(k_0 \log k_0)$, où k_0 est la complexité de la surface optimale satisfaisant les contraintes du problème, en temps polynomial (résultat d'ordre uniquement théorique, car leur complexité temporelle est de $O(n^7)$).

Élimination de sommets

Une première approche est de celle de Lee [Lee89] qui à chaque itération calcule pour chaque point l'erreur d'élévation qui se produirait si le point était supprimé de la triangulation. Il supprime alors le point de plus faible erreur ; la triangulation employée est celle de Delaunay et la borne d'arrêt concerne le nombre de sommets. La qualité des approximations est bonne mais le coût en mémoire et en temps de calcul est important. Cette méthode s'applique aux champs de hauteur.

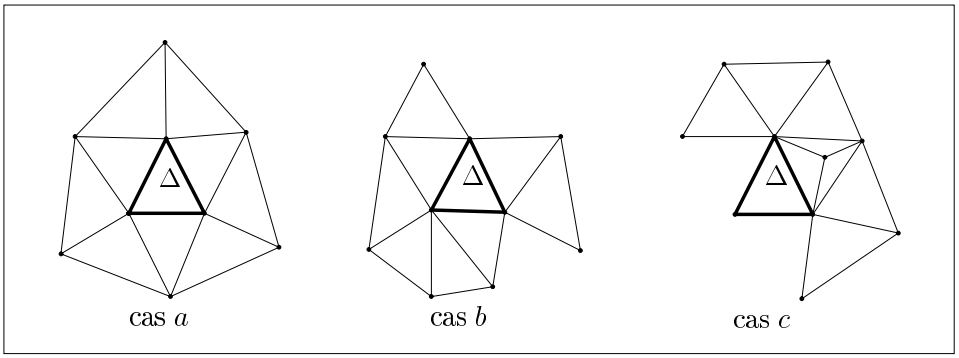
De manière plus générale, Schroeder, Zarge et Lorensen [SZL92] proposent

un algorithme qui s'adapte à n'importe quelle surface triangulée (en réalité une variété à bord). À chaque itération, tous les sommets qui ne sont pas sur la frontière et dont l'erreur est inférieure à une borne fixée sont détruits, et une retriangulation est effectuée ; l'erreur calculée en un sommet est la distance qui le sépare de l'approximation planaire de ses sommets voisins. Par rapport à la méthode de LEE, la mesure effectuée est moins coûteuse et plus appropriée, et l'algorithme permet la destruction simultanée de plusieurs points. Cependant, comme beaucoup de méthodes parallèles (pas au sens architecture de la machine), la qualité est moins bonne mais obtenue plus rapidement.

Élimination de triangles

HAMANN [Ham94] propose un algorithme de simplification de maillage par élimination de triangles ; à notre connaissance, c'est un des rares articles récents à traiter du sujet. Il détermine d'abord quels sont les triangles qui peuvent être remplacés par un sommet unique sans trop de conséquence sur la retriangulation locale (c'est encore une fois Delaunay qui est utilisé en procédant par échanges d'arêtes) ; ceci dépend des éléments voisins du triangle à éliminer, un triangle étant voisin d'un autre s'ils ont au-moins un sommet commun. La figure 1.26 (d'après [Ham94]) illustre la propriété à vérifier : soit Δ le triangle à tester. Si on ne peut pas tourner autour des voisins de Δ de manière continue (cas b) ou s'il existe un cycle parmi les voisins de Δ (cas c), alors on ne peut pas supprimer Δ de la triangulation. Le cas a représente une situation favorable. Il faut remarquer que les cas décrits, notamment le b, peuvent effectivement se produire car l'algorithme fonctionne pour une triangulation non forcément convexe.

Figure 1.26 – les cas où un triangle peut ou ne peut pas être éliminé.



Le poids d'un triangle Δ est alors défini par l'intermédiaire de 2 critères : le premier fait appel aux courbures principales estimées aux sommets, calculées suivant un schéma d'approximation pour les triangulations de surface [Far92]. Si on note κ_{1i} et κ_{2i} (pour $i = 1, 2, 3$) les courbures aux 3 sommet de Δ, ce poids vaudra :

$$\rho = \sum_{i=1}^{3} \left(|\kappa_{1i}| + |\kappa_{2i}| \right).$$

Le second critère est donné par les angles intérieurs au triangle, plus précisément :

$$\sigma = 2\left(\left(\sum_{i=1}^{3}\cos\alpha_i\right) - 1\right),$$

où les α_i sont les angles de Δ. Finalement, le poids de Δ est donné par la formule :

$$\omega = \rho\sigma.$$

À chaque passe de l'algorithme, le triangle Δ_{min} de plus petit poids ainsi que ses 3 sommets seront éliminés (à condition que Δ_{min} vérifie la configuration a de la figure 1.26) et seront remplacés par un sommet unique s dont l'emplacement sera défini par les voisins de Δ_{min} (si les voisins forment une couronne, s sera le centre de gravité de la couronne, les autres cas étant détaillés dans l'article). Une retriangulation locale est alors effectuée, en prenant en compte s.

On s'arrête lorsque l'on a éliminé un pourcentage fixé de points. Si on note n le nombre de points de départ, et m le nombre de triangles que l'on veut dans la représentation finale, l'algorithme général peut se schématiser de la manière suivante :

- **initialisations** :
  $\mathcal{T} \leftarrow$ la triangulation de Delaunay contrainte des points de départ
  on **calcule** le poids de tous les triangles de $\mathcal{T}$

- **tant que** nombre de triangles $> m$ **faire**
  $\Delta \leftarrow$ triangle éliminable de $\mathcal{T}$ de poids minimum
  on **retire** Δ de $\mathcal{T}$ et on le **remplace** par un point s
  $\mathcal{T} \leftarrow$ retriangulation locale de Delaunay
  on **calcule** les poids des nouveaux triangles

  fin tant que

Les résultats proposés sont convaincants, surtout d'un point de vue visuel, ce qui peut sembler normal puisque cette méthode élimine prioritairement les triangles de faible courbure ρ, donc ceux pour lesquels on pouvait supposer qu'il ne se passait pas grand chose. Cependant, l'algorithme précis est assez complexe puisqu'il requiert des tests géométriques à la fois pour éviter des modifications de topologie et pour placer les sommets à ajouter.

Élimination d'un groupe de triangles adjacents

En général, de tels algorithmes ne comportent qu'une seule itération. HINKER et HANSEN [HH93] recherchent des ensembles de triangles dont les normales sont presque parallèles, et retriangule chacun de ces ensembles. Leur méthode donne de mauvais résultats pour des surfaces à forte courbure, mais de très bons pour celles avec une courbure nulle dans une direction. De manière un peu similaire (bien que la construction de la triangulation soit tout à fait différente), KALVIN et TAYLOR [KT94] ont défini des groupes de triangles à partir de la distance à un plan.

VARSHNEY [Var94, VAB+95] propose un algorithme à plusieurs passes qui va chercher de manière exhaustive le plus grand triangle à "insérer" pour chaque étape,

ou du moins celui qui contient le plus possible de points. Chaque nouveau triangle ne doit pas recouvrir une partie d'un triangle déjà inséré ; la triangulation de la zone qu'il couvre est détruite et des petits triangles sont construits pour remplir la partie qui sépare l'ancienne triangulation de la nouvelle. Les problèmes de cet algorithme sont qu'il n'aboutit pas souvent (car, il y a utilisation de 2 approximations linéaires locales, une au-dessus, une au-dessous de la surface, qui ne doivent pas se chevaucher, propriété qui n'est pas possible d'obtenir pour tous les ensembles de points), qu'il n'existe pas de preuve que l'on peut réduire de plus de 30% la taille initiale et qu'il est très lent. En revanche, il donne de bonnes approximations pour les cas où il fonctionne.

1.3.3.3 Triangulations hiérarchiques

Dans les applications où la représentation d'une scène nécessite différents niveaux de détail (comme les simulateurs de vol), une hiérarchie facilite le rendu du niveau voulu ; des objets se situant loin de la scène pourront (ou plutôt devront) avoir une précision bien moindre que ceux du premier plan. Les méthodes de subdivisions hiérarchiques (parmi lesquelles on trouve les triangulations hiérarchiques) proposent une solution à ce problème en général assez rapide et simple, mais aboutissant à une qualité d'approximation bien inférieure aux méthodes "standards" (sauf au prix d'un stockage mémoire excessif).

Heckbert et Garland [HG94], ainsi que De Floriani, Marzano et Puppo [DMP94] proposent des articles de synthèse concernant ce sujet. Les triangulations hiérarchiques peuvent soit partir d'un niveau grossier pour donner des niveaux de plus en plus raffinés obtenus en ajoutant des points de données [DFNP84], soit utiliser une triangulation de Delaunay à chacun de ses niveaux avec plus ou moins de points [Flo89]. L'inconvénient des premières est qu'en général chaque triangle est raffiné indépendamment des autres : cela permet d'obtenir une structure naturelle d'arbre, mais aussi cela donne des triangles allongés à des niveaux de grande précision pour les détails (voir figure 1.27, d'après [BD95]). L'inconvénient des secondes est qu'elles ne permettent pas d'obtenir une structure d'arbre mais seulement de graphe (voir figure 1.28, d'après [BD95]). Ce sont ces problèmes que résolvent De Berg et Dobrindt [BD95].

Figure 1.27 – une hiérarchie et la structure d'arbre associée.

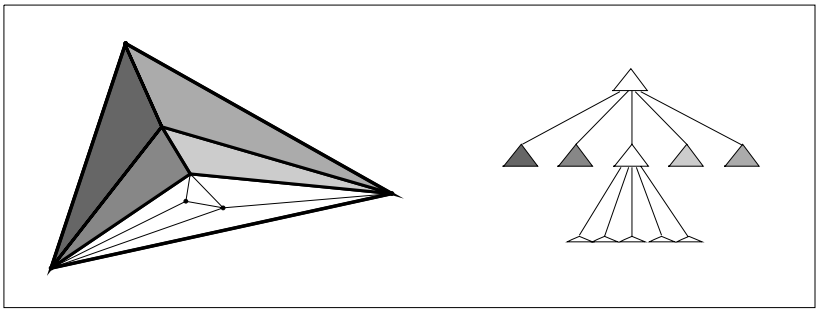
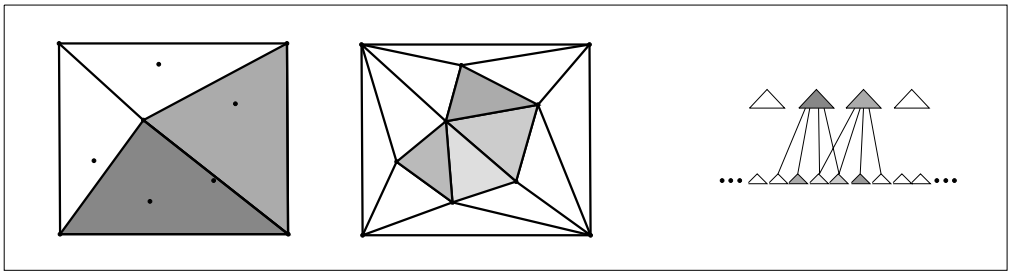
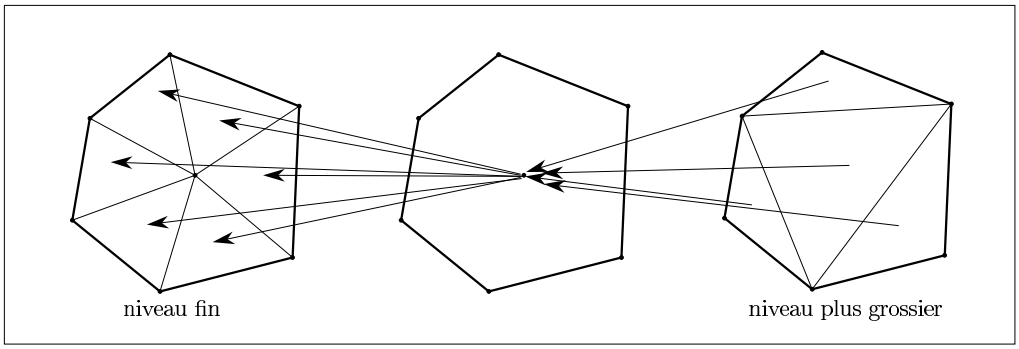


Figure 1.28 – une hiérarchie utilisant Delaunay et la structure de graphe associée.



Le principe de leur hiérarchie est, en partant du niveau de détail le plus fin, d'aller à un niveau plus grossier en retirant un sous-ensemble de sommets non adjacents (tout en conservant la possibilité de fixer de manière automatique ou pas certains sommets indispensables, tels que ceux qui servent à repérer un relief très prononcé) qui ont un degré borné par une certaine constante (en pratique, 12 fonctionne bien) ; alors, les arêtes du polygone étoilé, union des triangles incidents à un sommet que l'on désire retirer, sont des arêtes de la triangulation de Delaunay du niveau suivant. Ce qui permet de bâtir une triangulation de Delaunay hiérarchique avec, presque, une structure d'arbre naturelle associée, où un arc ne relie pas directement un triangle à ceux qu'il a donnés (puisqu'un triangle peut être obtenu à partir de plusieurs) mais où il existe un niveau intermédiaire où un noeud correspond au polygone étoilé du sommet supprimé (figure 1.29 d'après [BD95]).

Figure 1.29 – un niveau intermédiaire pour construire une triangulation de Delaunay hiérarchique.



Leur structure a une complexité mémoire en $O(n)$ où n est le nombre de sommets du niveau le plus fin ; des indications sont données sur la constante cachée et des moyens offerts pour la réduire qui impliquent des pertes d'information ou un passage plus brutal d'un niveau à son suivant. Par ailleurs, il est possible de dire dans quel triangle se trouve un point donné en temps $O(\log n)$.

En pratique, la méthode semble intéressante et permet de combiner très simplement plusieurs niveaux de détail à la fois ; cette propriété apparemment très utile pour les simulateurs de vol est bloquée par le coût mémoire trop important de la

structure. En pratique, il semblerait que les programmeurs de simulateurs préfèrent utiliser du Delaunay dynamique.

Des références concernant d'autres articles sur les triangulations hiérarchiques de surfaces sont données par HECKBERT et GARLAND [HGar], avec des commentaires orientés sur leur utilisation pour simplifier les surfaces.

1.3.4 Point de vue de la "Conception Géométrique Assistée par Ordinateurs"

Ce nom (remplacé dans la suite du chapitre par les initiales CGAO) est une traduction littérale de l'anglais "Computer Aided Geometric Design", domaine concerné par l'approximation et la représentation des courbes et surfaces par un procédé informatique. Le tracé des courbes et des surfaces joue un rôle essentiel dans la construction ou la conception de produits très différents comme les coques des bateaux, les fuselages et les ailes des avions, les pales des hélicoptères, les carrosseries des automobiles, les semelles de chaussures, les bouteilles... mais aussi dans la description de phénomènes géologiques, physiques ou encore médicaux.

Les surfaces en CGAO ne sont pas toujours représentées par des triangulations, et, même lorsque c'est le cas, il existe une phase post-triangulation qui permet d'arriver à un résultat plus "satisfaisant". Des bibliographies très importantes existent sur le domaine, parmi lesquelles une, maintenant un peu ancienne, qui traite des approximations pour les fonctions à plusieurs variables [FS87], mais aussi des articles de synthèse [Bar85] et même des adresses WWW mises à jour régulièrement, par exemple :

`http://www.cs.wisc.edu/~deboor/bib/bib.html`.

Nous allons donc décrire certaines des méthodes les plus classiques d'approximation de surfaces succintement pour celles qui ne font pas appel aux triangulations, et un plus en détail pour les autres. Les premières sont en général globales, alors que les secondes s'appuient sur des propriétés locales (qui expliquent, ou impliquent, le partitionnement des données en objets simples, des triangles). Certaines des méthodes, du moins celles qui existaient à l'époque, ont été testées de manière rigoureuse et complète par FRANKE [Fra79] et un résumé des résultats est paru [Fra82] ; en plus d'un grand nombre de fonctions et d'ensembles de points caractéristiques (qui ont été depuis repris par un bon nombre de chercheurs pour effectuer leurs tests), il propose un tableau comparatif de divers critères de qualité (aspect visuel, fidélité de l'approximation, sensibilité aux paramètres) et de fonctionnalité (temps d'exécution, facilité d'implémentation, capacité de stockage recquise). Il serait sans doute très intéressant de réactualiser un tel article, s'il est encore raisonnablement possible de tester les (trop ?) nombreuses méthodes existant maintenant.

Dans ce qui va suivre, le but sera donc de définir une surface correcte (en général lisse) à partir d'un ensemble $\mathcal{S}$ de points de données $S_1, \ldots, S_n$ où $S_i = (x_i, y_i, f_i)$ avec $f_i = f(x_i, y_i)$ et f fonction de surface représentative $\mathcal{F}$.

1.3.4.1 Méthodes de type Shephard

Ce sont des interpolations utilisant des poids fondés sur l'inverse d'une fonction distance ; la méthode initiale, due à SHEPHARD [She68] (elle a ensuite été généralisée pour éviter certaines dégénérescences), prend un interpolant égal à une moyenne pondérée de la valeur de la dernière coordonnée des points, le poids étant une puissance de la distance inverse, ce qui peut s'écrire :

$$\tilde{f}(x,y) = \sum_{k=1}^{n} f_k \omega_k(x,y),$$

avec

$$\omega_k(x,y) = \frac{\prod_{j\neq k} \mathrm{dist}_j(x,y)^\mu}{\sum_{i=1}^{n} \prod_{j\neq i} \mathrm{dist}_j(x,y)^\mu},$$

où $\mathrm{dist}_j(x,y) = \sqrt{(x-x_j)^2+(y-y_j)^2}$ et $\mu = 2$ en général ; on peut remarquer que $\tilde{f}(x_k,y_k) = f_k$. BARNHILL a proposé [Bar77] de remplacer les f_k par des plans tangents $f_k(x,y) = f_k + \lambda_k(x-x_k) + \mu_k(y-y_k)$, permettant ainsi d'éviter la plupart des dégénérescences (les zones trop plates) puisqu'à ce moment-là, $\tilde{f}$ interpole aussi bien les f_k que leurs plans tangents. Comme dans la majorité des cas, ces plans doivent être approchés, LITTLE a suggéré d'effectuer d'abord une triangulation des points de données [Lit83].

1.3.4.2 Multiquadriques

HARDY [Har71] a créé un schéma simple à implémenter et raisonnable en temps de calcul si le nombre de points n'est pas trop important. Une fonction de base ("quadrique") $Q_k = \sqrt{\mathrm{dist}_k^2 + r^2}$ est associée au k-ième point de donnée ; $\mathrm{dist}_k(x,y)$ est la distance du point (x,y) au point de donnée (x_k,y_k), et r est un paramètre de la méthode. Alors, une combinaison linéaires des fonctions

$$\tilde{f}(x,y) = \sum_{k=1}^{n} \epsilon_k Q_k(x,y)$$

est nécessaire pour interpoler les données. Ce qui implique de résoudre le système d'équations

$$\sum_{k=1}^{n} \epsilon_k Q_k(x_i,y_i) = f_i, \quad i = 1,\ldots,n.$$

L'existence de l'interpolant dépend ainsi de la non singularité de la matrice constituée des $Q_k(x_i,y_i)$. Des articles montrent qu'en pratique, cette méthode fonctionne très bien, souvent mieux que n'importe quelle autre. Elle implique cependant de se limiter sur le nombre de points initiaux.

1.3.4.3 Réseau de norme minimale

Sous les initiales MNN (pour *Minimum Norm Network*) se cache une méthode en 3 étapes, qui fonctionne bien en pratique :

- on construit une *triangulation* (il est en général fait appel à Delaunay) de l'enveloppe convexe de l'ensemble $\mathcal{S}$ des points de données, en utilisant tous les points de $\mathcal{S}$.

- on construit ensuite *un réseau de courbes* qui minimisent certaines valeurs (données par des intégrales de fonctions) pour l'union de toutes les arêtes de la triangulation.

- on *étend* le réseau de courbes à la triangulation, afin d'obtenir une fonction modèle ; cette étape fait souvent appel à des interpolants triangulaires C^1 (ou encore à des splines triangulaires, des polynômes de Bernstein, des surfaces composites du type Bézier) qui s'appuient sur les courbes du réseau.

La dernière étape peut poser certains problèmes, car elle nécessite de résoudre un système linéaire creux de taille $2n \times 2n$, où n est le nombre de points de données.

1.3.4.4 Méthodes s'appuyant sur des triangulations

Moins sophistiquées, ces méthodes construisent directement des fonctions sur chaque triangle d'une triangulation initiale, fonctions qui approchent les points de données, les plans tangents et, si possible, des dérivés d'ordre supérieur (ou leur estimation). Plusieurs de ces fonctions sont décrites dans [BFK84] ; à nouveau, ce sont des schémas quadratiques, ou des splines, ou des Béziers qui sont utilisés.

1.3.4.5 Sélection

FRANKE [Fra87] rappelle que, dans certains cas (il cite l'exemple des applications océanographiques), le nombre de données peut être trop important pour toutes les utiliser ; il faut alors soit considérer des fonctions qui ne tiennent pas compte de tous les paramètres, soit être en mesure de sélectionner un sous-ensemble de points pour leur appliquer une des méthodes globales précédentes.

Il est ainsi possible d'effectuer une approximation par les moindres carrés ; c'est une approximation qui comporte (beaucoup) moins de fonctions de base qu'il n'y a de points. Le problème consiste à choisir correctement ces fonctions. Des travaux ont été réalisés qui utilisent des produits tensoriels de splines cubiques ([Die81] par exemple). Ces méthodes fonctionnent bien en pratique pour des ensembles de points presque uniformément distribués.

Le recours à la sélection peut se faire de l'une des manières suivantes :

- on connait à l'avance des points modèles (par exemple, des points caractéristiques, comme des failles en océanographie) et on va effectuer des choix ultérieurs relatifs à ces modèles, afin d'éviter une déviation trop importante entre l'interpolant et l'ensemble de tous les sommets .

– on subdivise l'ensemble initial en une grille rectangulaire et on choisit des représentants à l'intérieur de chacune des cellules de la grille ; à nouveau, les points non sélectionnés faisant apparaitre une forte déviation avec l'interpolant sont ajoutés.

– on effectue des calculs locaux pour déterminer si un point a une influence significative sur la définition de la surface, et, si c'est le cas, on l'ajoute.

– on peut également procéder par élimination : une grille (ou une triangulation) est construite à l'aide de toutes les données, puis des points sont retirés si l'ensemble ainsi réduit varie de moins d'un paramètre ϵ de la surface détruite de la grille globale.

Toutes ces méthodes sont expliquées et référencées plus en détail dans différents articles de FRANKE, NIELSON, FARIN ou BARNHILL [FN90, Nie93, BFK84].

1.4 Triangulations globalement optimales

Le but de cette thèse est de présenter de nouveaux algorithmes optimaux pour un critère déterminé. Ce n'est évidemment pas la première fois que cela est réalisé, et nous allons décrire, dans cette section, quelques uns des algorithmes géométriques qui optimisent, pour un critère ou pour un autre, la triangulation d'un ensemble de points. D'autres techniques existent et seront présentées au chapitre 6.

1.4.1 Triangulations optimales pour la norme L_∞

1.4.1.1 Triangulation d'un domaine

Dans un article de 1991 [D'A91], D'AZEVEDO désire obtenir un maillage triangulaire *optimal* dans le sens où, un seuil de tolérance étant fixé, la meilleure triangulation sera celle qui contiendra le moins possible de triangles ; le domaine à trianguler est ainsi uniquement donné par son contour, et on ne dispose d'aucun point intérieur. Le but est de calculer un maillage optimal, en supposant que l'erreur tolérée est faible et que la forme du domaine n'influencera pas la triangulation finale. Pour cela, il va avoir recourt à un changement de coordonnées sur le maillage fondé non pas sur des considérations géométriques, mais dérivé directement de propriétés de l'erreur d'interpolation. Son travail s'inscrit directement dans le domaine de la recherche de maillages pour la méthode des éléments finis (méthode indispensable notamment en mécanique des fluides) et fait suite à des travaux commencés par NADLER [Nad85] et D'AZEVEDO et SIMPSON [DS89].

La mesure de l'erreur d'interpolation est effectuée en utilisant la norme L_∞ : l'erreur commise par l'approximation locale de la surface par un triangle sera donnée par le maximum d'écart (en valeur absolue) qui existe entre un point du triangle et son *relevé* sur la surface. Dans cet article, des résultats d'optimalité globale sont obtenus, en partant de considérations locales, pour une classe de fonctions incluant les fonctions harmoniques (fonctions de plusieurs variables à valeurs complexes de Laplacien nul) et les fonctions quadratiques.

Son but étant d'obtenir un maillage contenant le moins possible de triangles, tout son travail va consister à rechercher la forme des triangles d'aire maximale qui obéissent à la tolérance d'erreur donnée.

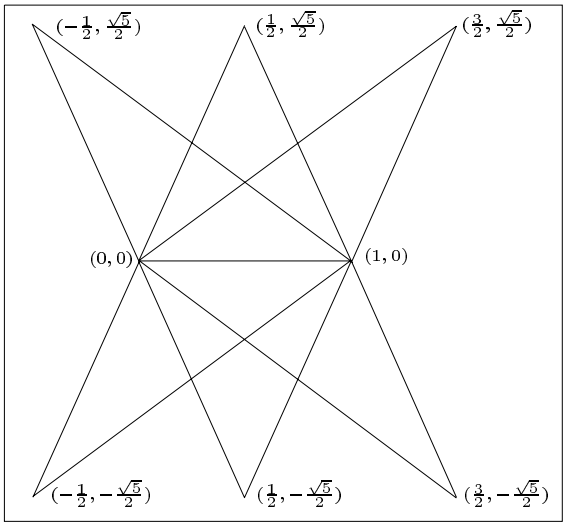
Analyse locale pour les surfaces quadratiques

Cette classe de fonctions se divise en deux groupes :

- les quadriques *convexes* (du type $f(x,y) = \alpha x^2 + \beta y^2$, avec α et β positifs) : l'erreur pour ces fonctions est directement liée à une ellipse circonscrite aux sommets du triangle ; il montre alors que par changement de coordonnées, on peut se ramener au paraboloïde de révolution unité, pour lequel l'ellipse sera un cercle lié directement au maximum d'erreur atteignable [DS89]. Un triangle équilatéral sera donc de forme optimale, et on pourra construire un maillage optimal à l'aide d'un réseau hexagonal déformé aux bords du domaine.

- les quadriques *non convexes* (de type selle de cheval) : par changement de coordonnées, on peut se rendre compte que l'erreur est une fonction harmonique qui atteind ses extremums sur les arêtes composées des trois sommets choisis. En calculant, on se rend alors compte que le maximum local est atteint au milieu des arêtes du triangle, ce qui permet de déduire une forme optimale pour les triangles. Par exemple, sur la figure 1.30, on peut voir les six configurations possibles pour un triangle de forme optimale pour approcher le paraboloïde hyperbolique d'équation $f(x,y) = x^2 - y^2$, et de sommets fixés $(0,0)$ et $(1,0)$.

Figure 1.30 – des triangles de forme optimale pour le paraboloïde hyperbolique unité.



Changement de coordonnées

À partir des observations effectuées sur les quadriques, et du fait que l'approximation locale au deuxième ordre d'une fonction est une quadrique (par son dévelop-

pement de Taylor), il reste à voir sous quelles conditions il va être possible de trouver un changement de coordonnées qui va permettre de se ramener à un résultat exploitable. C'est ce que D'AZEVEDO fait en ayant recours à des propriétés classiques de la géométrie différentielle caractérisant un espace *plat*; il montre qu'un changement peut être calculé en tant que solution d'un système d'équations différentielles établi comme valeur initiale du problème. Finalement, il prouve qu'une classe de fonctions satisfait toujours ce système: les fonctions *harmoniques*.

Algorithme de génération de maillage optimal

Les étapes pour obtenir un maillage optimal pour une fonction donnée sur un domaine Ω sont alors les suivantes:

- on **détermine** le changement de coordonnées qui va transformer le domaine en une région Ω , définie sur un espace *plat*.
- on **génère** une triangulation régulière $\mathcal{T}$ de triangles de forme optimale pour le domaine Ω , la taille des triangles étant donnée par l'erreur de tolérance souhaitée.
- on **calcule** les images de la triangulation $\mathcal{T}$ dans l'espace de départ.

Des simplifications éventuelles interviennent si le domaine Ω choisi est le carré unité.

1.4.1.2 Triangulation d'un ensemble de points

Quelques temps avant de publier cet article, D'AZEVEDO, accompagné de SIMPSON, en avait publié un autre [DS89] dans lequel il s'était intéressé au problème de la triangulation d'un ensemble de points donnés, optimale au sens de la minimisation de l'erreur maximum (norme L_∞) pour l'interpolation d'une surface quadratique convexe par des fonctions linéaires par morceaux. Pour cela, ils étudient la question d'un choix optimal des *incidences* pour les arêtes des triangles.

Les auteurs montrent d'abord que pour une fonction quadratique convexe quelconque $f(x,y) = \alpha x^2 + \beta y^2$ avec $\alpha \geqslant \beta > 0$, si on choisit l'incidence définie par la triangulation de Delaunay, le résultat peut être arbitrairement loin de l'optimal au sens de la minimisation de l'erreur maximum. En effet, en prenant $\alpha = 100$ et $\beta = 1$, l'erreur obtenue en utilisant le critère de Delaunay est pratiquement six fois plus importante que celle donnée par la triangulation optimale (pour un ensemble de 50 points choisis dans le carré unité).

Expression de l'erreur

Soit la fonction $f(x,y) = \alpha x^2 + \beta y^2 + \delta_1 x + \delta_2 y + \delta_3$ avec $\alpha \geqslant \beta > 0$. L'erreur d'interpolation maximum pour un triangle Δ est donnée par:

$$E_{max}(\Delta) = \max_{(x,y)\in\Delta} |p_{\Delta,f}(x,y) - f(x,y)|,$$

où $p_{\Delta,f}(x,y)$ représente l'interpolant linéaire de f aux sommets de Δ. Il est alors montré que l'erreur $E_{max}(\Delta)$ dépend de l'emplacement du centre d'une ellipse circonscrite (particulière) à Δ. Par une transformation affine, les ellipses (qui représentent les courbes de niveau de l'erreur) deviennent des cercles, et dans le plan ainsi transformé, une interprétation géométrique est possible : soit $\bigcirc(\Delta)$ le cercle circonscrit au transformé du triangle Δ et $|\bigcirc(\Delta)|$ l'aire de ce cercle. Si l'image du triangle n'a pas d'angle obtus, alors l'erreur maximum est proportionnelle à $|\bigcirc(\Delta)|$ (pour une quadrique convexe, l'erreur maximum est atteinte au centre du cercle). S'il y a un triangle obtus, l'erreur maximum apparait sur la plus longue arête et cette erreur est proportionnelle à la surface du cercle ayant pour diamètre la plus longue arête.

Triangulation d'incidence optimale

Les auteurs montrent du coup que le problème de construire une triangulation d'incidence localement optimale de n points revient à engendrer une triangulation de Delaunay (ce qui implique que le problème peut être résolu grâce à un algorithme en $O(n \log n)$) ; ils remarquent aussi que cette triangulation est globalement optimale.

Pour cela, ils utilisent la transformation affine définie au paragraphe précédent (celle qui va envoyer les ellipses d'erreur sur des cercles), et montrent successivement les points suivants :

lemme 1.4 *Soit $\square abcd$ un quadrilatère convexe dont le sommet a est extérieur à $\bigcirc(\Delta bcd)$; alors*

$$\max(|\bigcirc(\Delta abc)|, |\bigcirc(\Delta adc)|) \geqslant \max(|\bigcirc(\Delta bcd)|, |\bigcirc(\Delta abd)|).$$

corollaire 1.2 *Le critère du cercle vide appliqué à un qradrilatère convexe choisit la diagonale qui minimise le cercle circonscrit maximum des triangles correspondants.*

lemme 1.5 *Soit $\square abcd$ un quadrilatère convexe dont le sommet a est extérieur à Δbcd ; le critère du cercle vide choisit l'incidence qui minimise l'aire du cercle correspondant à l'erreur d'interpolation maximum sur le quadrilatère.*

Ces deux lemmes permettent de prouver le théorème annoncé :

théorème 1.7 *Une triangulation d'interpolation localement optimale de n sommets est définie par une triangulation de Delaunay dans le plan transformé (et est donc calculable en $O(n \log n)$, ce qui est optimal).*

Et on peut enfin remarquer que l'incidence est globalement optimale, puisque la taille du cercle d'erreur maximum est donné par la triangulation de Delaunay dans le plan transformé, et ce cercle est unique (comme la triangulation de Delaunay).

1.4.2 Insertion d'arêtes

Dans leur article [BEE+93], Bern, Edelsbrunner, Eppstein, Mitchell et Tan présentent un nouveau paradigme, appelé *insertion d'arêtes*, qui permet de

trouver les triangulations optimales pour des critères du genre minmax (minimisation d'une valeur maximale pour tous les triangles) ou maxmin, en temps polynomial. Ceci est d'autant plus remarquable que les approches gloutonnes (par exemple celles qui cherchent à éliminer des triangles en partant du plus mauvais) sont bloquées par la NP-complétude du problème de décision suivant [Llo77] : étant donnés un ensemble de points et d'arêtes, est-ce qu'un certain sous-ensemble des arêtes définit une triangulation des points.

L'algorithme présenté permet de construire la triangulation optimale d'un ensemble de n points pour certains critères en $O(n^2 \log n)$, et pour d'autres critères en $O(n^3)$.

1.4.2.1 Le paradigme insertion d'arêtes

Soit ω une fonction qui associe une valeur réelle à un triangle ; ω est appelée la *mesure* d'un triangle, et sera toujours ici un critère du genre minmax (ou maxmin). C'est-à-dire que la *mesure* ω d'une triangulation $\mathcal{T}$ est donnée par $\omega(\mathcal{T}) = \max \omega(\Delta)$, pour tout $\Delta \in \mathcal{T}$. Si $\mathcal{T}$ et $\mathcal{T}'$ sont deux triangulations du même ensemble de points, $\mathcal{T}'$ est une *amélioration* de $\mathcal{T}$, et on note $\mathcal{T}' < \mathcal{T}$ si $\omega(\mathcal{T}') < \omega(\mathcal{T})$ ou si les deux mesures sont égales et l'ensemble des triangles de $\mathcal{T}'$ qui donnent la valeur de la mesure de $\mathcal{T}'$ forme un sous-ensemble strict de l'ensemble des triangles de $\mathcal{T}$ qui donnent sa mesure. Une triangulation $\mathcal{T}$ sera *optimale* pour ω si aucune amélioration n'est possible.

L'algorithme général utilisant le paradigme peut être décrit de la manière suivante :

- on dispose d'un ensemble $\mathcal{S}$ de n points de $\mathbb{R}^2$.
- **initialisation** : soit $\mathcal{T}$ une triangulation quelconque de $\mathcal{S}$

répéter
  $\mathcal{T}temp \leftarrow \mathcal{T}$
  pour toutes les paires s_i, s_j de $\mathcal{S}$
    $\mathcal{T}' \leftarrow \mathcal{T}$
    on **ajoute** l'arête $[s_i s_j]$ à $\mathcal{T}'$
    on **retire** de $\mathcal{T}'$ toutes les arêtes qui coupent $[s_i s_j]$
    on **retriangule** les régions polygonales de $\mathcal{T}'$ ainsi obtenues
    si $\mathcal{T}' < \mathcal{T}$ **alors**
      $\mathcal{T} \leftarrow \mathcal{T}'$
      on **sort** de la boucle **pour**
    fin si
  fin pour

jusqu'à $\mathcal{T}temp = \mathcal{T}$

L'insertion d'arêtes n'est rien donc qu'une généralisation de l'échanges d'arêtes, où l'on se donne en plus la possibilité d'insérer des segments qui peuvent couper plusieurs autres segments de la triangulation. L'algorithme - naïf - ainsi décrit se termine en un temps $O(n^8)$: la boucle **pour** est itérée $O(n^2)$ fois, l'utilisation de

l'algorithme optimal KLINCSEK [Kli80] pour la retriangulation d'un polygone fait apparaître un $O(n^3)$, et la boucle **répéter** nécessite $O(n^3)$ exécutions.

1.4.2.2 Conditions suffisantes d'application du paradigme

On dit qu'une triangulation $\mathcal{T}$ *coupe* un triangle Δabc en b si elle contient une arête $[bs]$ qui coupe l'arête $[ac]$. Si $\mathcal{T}$ coupe Δabc en b, alors elle ne peut pas le couper en a ou en c. On appelle un point b l'*ancre* d'un triangle vide Δabc d'un ensemble de points $\mathcal{S}$, si toute triangulation $\mathcal{T}$ de $\mathcal{S}$ telle que $\omega(\mathcal{T}) \leqslant \omega(\Delta abc)$ soit contient Δabc, soit coupe Δabc en b; intuitivement, si un triangle comprend une ancre, c'est son plus "mauvais sommet".

On peut maintenant énoncer les deux conditions :

condition (I), dite condition faible de l'ancre
: pour toute triangulation $\mathcal{T}'$ et tout triangle Δ de $\mathcal{T}'$ tel que $\omega(\Delta) = \omega(\mathcal{T}')$, il y a un sommet de Δ qui est une ancre.[1]

condition (II), dite condition forte de l'ancre
: pour toute triangulation $\mathcal{T}'$ et tout triangle Δ de $\mathcal{T}'$, il y a un sommet de Δ qui est une ancre.

1.4.2.3 Algorithme modifié

La version améliorée de l'algorithme restreind le choix des arêtes $[bs]$ à insérer à celles qui coupent un plus mauvais triangle Δabc en son ancre b ; par ailleurs, les deux régions polygonales sont retriangulées, non plus par programmation dynamique, mais en supprimant successivement les oreilles des polygones (triangles formés par deux arêtes du polygone et une diagonale).

Soit $\mathcal{T}$ une triangulation de plus mauvais triangle Δabc d'ancre b. On note $[bs_1]$, $[bs_2], \ldots$ la séquence d'arêtes candidates à l'insertion. L'algorithme devient :

- **initialisation** : soit $\mathcal{T}$ une triangulation quelconque de $\mathcal{S}$.
- **répéter**
  $\mathcal{T}temp \leftarrow \mathcal{T}$
  on **cherche** Δabc un plus mauvais triangle de $\mathcal{T}$ d'ancre b
  $s \leftarrow s_1$
  tant que s existe
    $\mathcal{T}' \leftarrow \mathcal{T}$
    on **ajoute** l'arête $[bs]$ à $\mathcal{T}'$
    on **retire** de $\mathcal{T}'$ toutes les arêtes qui coupent $[bs]$
    on **triangule** partiellement les régions polygonales $\mathcal{P}_1$ et $\mathcal{P}_2$ de $\mathcal{T}'$ ainsi obtenues en isolant les oreilles Δ telles que $\omega(\Delta) < \omega(\Delta abc)$
    si $\mathcal{P}_1$ et $\mathcal{P}_2$ sont totalement triangulées **alors**
      $\mathcal{T} \leftarrow \mathcal{T}'$

1. cela signifie que $\mathcal{T}$ ne peut améliorer $\mathcal{T}'$ que si elle coupe un des plus mauvais triangles de $\mathcal{T}'$ en son ancre.

on **sort** de la boucle **tant que**
sinon
$s \leftarrow$ suivant(s)
fin si
fin tant que

jusqu'à $\mathcal{T}temp = \mathcal{T}$ et tous les plus mauvais triangles de $\mathcal{T}$ ont été testés.

On peut alors prouver (car, il existe au plus $\binom{n}{2}$ insertions d'arêtes à tester) :

théorème 1.8 *Soit $\mathcal{S}$ un ensemble de n points de $\mathbb{R}^2$, et soit ω une mesure qui satisfait la condition (I) et pour laquelle l'ancre d'un plus mauvais triangle soit calculable en temps constant.*

- *une triangulation (contrainte ou non) de $\mathcal{S}$ qui minimise la mesure maximale d'un triangle peut être construite en temps $O(n^3)$.*
- *dans le cas non dégénéré ($\omega(\Delta a'b'c') \neq \omega(\Delta abc)$ sauf si $\Delta a'b'c' = \Delta abc$), la triangulation unique qui minimise lexicographiquement le vecteur des mesures décroissantes des triangles peut être également construite en temps $O(n^3)$.*

De plus, pour les mesures satisfaisant la condition (II), un ordre d'insertion spécial peut être effectué tel que le temps nécessaire à la construction des triangulations optimales du théorème 1.8 soit ramené à $O(n^2 \log n)$.

1.4.2.4 Mesures testées

Quelques mesures sont alors données qui satisfont les conditions (I) ou (II) et dont il est possible de trouver en temps polynomial une triangulation optimale.

- historiquement, la méthode a été trouvée pour résoudre le problème de la construction de la triangulation qui minimise le plus grand angle [ETW92].
- une autre mesure testée est celle qui maximise la plus petite hauteur des triangles (elle permet d'éviter les triangles fins et allongés) ; ces deux mesures satisfont la condition (II).
- la minimisation de l'excentricité maximum : l'*excentricité* ϵ d'un triangle de cercle circonscrit de centre c et de rayon ρ est la plus grande distance qui peut séparer c d'un point du triangle (si c appartient au triangle, ϵ vaudra 0). Cette mesure est directement reliée au plus grand angle d'un triangle.
- si l'ensemble de points représente les projections de points appartenant à une surface f, on définit la *pente* en un point (x, y) comme étant la valeur de la racine carrée de $\nabla^2 f = (\partial f/\partial x)^2 + (\partial f/\partial y)^2$. $\sigma(\Delta)$ est alors la mesure égale à la pente en tout point du triangle Δ, et $\sigma(\mathcal{T})$ est la pente maximale parmi tous les triangles de $\mathcal{T}$.
  σ, comme ϵ, est une mesure qui satisfait la condition (I), mais pas la condition (II).

1.4.2.5 Conclusion

Les algorithmes implémentés par WAUPOTITSCH ont montré que celui de complexité $O(n^2 \log n)$ fonctionnait très bien, même pour des gros ensembles de points, et s'exécutait en pratique en un temps bien inférieur au cas le pire. En revanche, le même phénomère n'a pas été observé pour l'algorithme en $O(n^3)$.

Chapitre 2

Présentation du problème

Nous avons vu, au paragraphe 1.1.1.3, que la triangulation de Delaunay ou une transformée simple approche au mieux, au sens d'une approximation linéaire par morceaux, toutes les quadriques définies par des fonctions convexes. Cette propriété permet de donner une bonne triangulation pour une surface dont la développée de Taylor au second ordre est une fonction quadratique convexe.

Cependant, jusqu'à présent, peu de recherches ont été consacrées à l'étude de surfaces non convexes ; cela est également vrai dans d'autres domaines de la géométrie algorithmique, où la convexité apparait très souvent comme facteur limitant d'une méthode. Il a donc semblé intéressant de poursuivre, en quelque sorte, le travail de MELISSARATOS [Mel93] en s'occupant de l'étude de triangulations optimales pour des surfaces non convexes "simples", c'est-à-dire les paraboloïdes hyperboliques.

Nous allons, dans ce chapitre, définir tous les objets mathématiques qui seront utilisés par la suite ; nous verrons d'abord ce qui a été appelée la *courbe de séparation* d'un triangle pour un critère donné, et les critères choisis pour rechercher cette courbe. Une description complète du paraboloïde hyperbolique sera enfin établie, ainsi qu'une énumération de ses principales propriétés mathématiques.

2.1 Définitions

2.1.1 Courbes de séparation

Soient $\mathcal{S} = \{s_1, \ldots, s_n\}$ un ensemble de n points du plan, et f une fonction à 2 variables de surface représentative $\mathcal{F}$ d'équation $z = f(x, y)$. À chaque sommet $s_i = (x_i, y_i), i = 1 \ldots n$ correspond un point (de l'espace euclidien à 3 dimensions) de la surface $\mathcal{F}$ $S_i = (x_i, y_i, z_i)$ avec $z_i = f(x_i, y_i)$. Soit Ω l'enveloppe convexe de $\mathcal{S}$; à partir de maintenant, on considérera qu'une triangulation $\mathcal{T}$ de $\mathcal{S}$ sera une triangulation de Ω dont les s_i sont les sommets.

Décider qu'une triangulation est meilleure qu'une autre dépend du choix d'un critère : celui-ci peut concerner les sommets, les arêtes et/ou les triangles. Le critère adopté et les éléments auxquels il se rapporte seront décrits au paragraphe suivant.

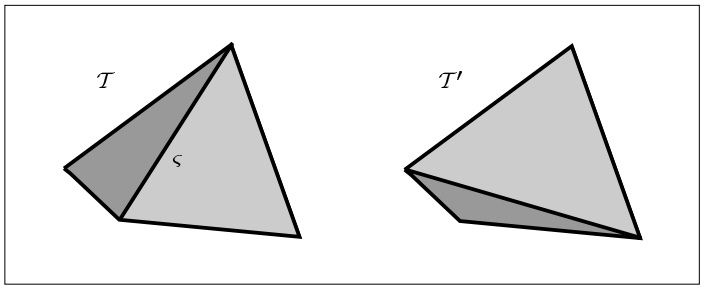
Les définitions qui vont suivre maintenant sont les définitions classiques utilisées dans les différents articles concernant les triangulations dépendantes des données (notamment ceux de DYN, LEVIN et RIPPA [DLR90a, DLR90b] ou encore de BROWN [Bro91]).

Soit ς une arête intérieure de la triangulation $\mathcal{T}$ (ς n'est pas une arête de Ω) ; alors, il existe deux triangles de $\mathcal{T}$ qui ont pour arête commune ς. Ces deux triangles forment un quadrilatère $\mathcal{Q}$. On suppose qu'un critère d'optimalité ν est choisi, c'est-à-dire un critère qui est capable de donner, entre plusieurs alternatives, une préférence pour une triangulation, définissant ainsi un ordre sur l'ensemble des triangulations.

définition 2.1 *Une arête intérieure ς de $\mathcal{T}$ est* localement *optimale pour le critère ν si une des conditions suivantes est valable :*

- *le quadrilatère $\mathcal{Q}$ n'est pas convexe.*
- *le quadrilatère $\mathcal{Q}$ est strictement convexe et $\nu(\mathcal{T}) \leqslant \nu(\mathcal{T}')$ où $\mathcal{T}'$ est la triangulation obtenue à partir de $\mathcal{T}$ en remplaçant ς par l'autre diagonale de $\mathcal{Q}$. (voir figure 2.1)*

Figure 2.1 – les deux triangulations possibles d'un quadrilatère convexe.



La notation $\nu(\mathcal{T}) \leqslant \nu(\mathcal{T}')$ signifie que le critère ν préfère la triangulation $\mathcal{T}$ à la triangulation $\mathcal{T}'$ (on peut considérer le critère comme une fonction de coût : moins le coût est important, meilleure est la triangulation).

définition 2.2 *Une triangulation $\mathcal{T}$ d'un ensemble de points $\mathcal{S}$ est dite* **localement**

optimale par rapport au critère ν si chacune de ses arêtes intérieures est localement optimale pour ν.

définition 2.3 *Une triangulation $\mathcal{T}$ d'un ensemble de points $\mathcal{S}$ est dite* **globalement** *optimale pour le critère ν (ou* **optimale** *tout court) si et seulement si quelle que soit $\mathcal{T}'$ triangulation de $\mathcal{S}$,*

$$\nu(\mathcal{T}) \leqslant \nu(\mathcal{T}').$$

Par conséquent, la triangulation optimale est aussi localement optimale (mais, la réciproque est en général fausse).

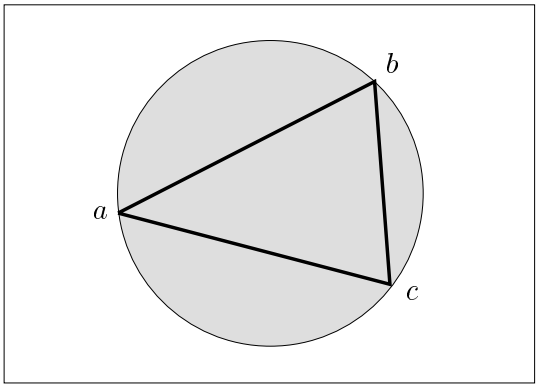
Soit ν un critère s'appliquant aux arêtes ; on peut étendre la définition de ν aux triangles en supposant qu'un triangle t sera localement optimal par rapport à ν si et seulement si ses trois arêtes seront localement optimales pour ν.

On peut alors donner la définition suivante pour les courbes de séparation.

définition 2.4 *Soit $\Delta = \Delta abc$ un triangle dont les sommets appartiennent à $\mathcal{S}$. La* ***courbe de séparation*** *de Δ est la courbe délimitant la zone du plan à l'intérieur de laquelle aucun sommet des triangles voisins de Δ ne doit se trouver si on veut que Δ appartienne à la triangulation localement optimale de $\mathcal{S}$.*

Par exemple, pour la triangulation de Delaunay, on sait que la courbe de séparation d'un triangle est le cercle circonscrit au triangle : si un point se trouve à l'intérieur du cercle, alors le triangle de départ ne sera pas maintenu dans la triangulation finale (cf figure 2.2). Le critère associé à la triangulation de Delaunay peut avoir à faire avec les angles, puisque l'on sait qu'elle maximise le plus petit angle. Il faut aussi rappeler que, pour la triangulation de Delaunay, l'optimalité locale entraine l'optimalité globale [Ede87], mais que cela représente une exception.

Figure 2.2 – courbe de séparation d'un triangle de la triangulation de Delaunay.



L'intérêt de la recherche des courbes de séparation réside surtout dans l'obtention d'un critère facile à utiliser pour tester l'optimalité locale d'une triangulation d'un ensemble de points. La connaissance des courbes de séparation permet une décision simple et rapide, et aboutit ainsi à un algorithme de triangulation aisément programmable.

2.1.2 Critère d'optimalité choisi

On désire étudier les triangulations optimales pour une surface spécifique, le paraboloïde hyperbolique d'équation $z = x^2 - y^2$. Nous allons donc nous intéresser au problème suivant : soit $\mathcal{T}$ une triangulation d'un ensemble $\mathcal{S}$ de points du plan. À chaque triangle Δ de $\mathcal{T}$ correspond un plan contenant les S_j dont les projections donnent les sommets de Δ : ce plan définit une approximation linéaire locale du paraboloïde hyperbolique donnée par les valeurs aux sommets du triangle. Ainsi, choisir une triangulation de $\mathcal{S}$, c'est se donner une certaine approximation linéaire par morceaux de la surface (voir figure 2.3).

On va alors chercher la triangulation (parmi toutes celles possibles de $\mathcal{S}$) qui approche linéairement le paraboloïde hyperbolique le mieux possible au sens de la norme L_2 pour les fonctions.

définition 2.5 *Étant donnée une fonction f de deux variables x et y définie sur un domaine planaire $\mathcal{D}$, la norme L_2 de f est donnée par :*

$$\|f\|_2 = \sqrt{\iint_{\mathcal{D}} f^2(x,y)\,dx\,dy}.$$

définition 2.6 *Étant donnés une fonction $f : \mathbb{R}^2 \to \mathbb{R}$, $\mathcal{S}$ un ensemble de points $s_i = (x_i, y_i), i = 1 \ldots n$ du plan, $\mathcal{T}$ une triangulation de $\mathcal{S}$, à chaque triangle Δ de $\mathcal{T}$ correspond une approximation linéaire p_Δ de la fonction f valable sur toute la surface limitée par Δ. Soit $P_\mathcal{T}$ l'interpolatoin linéaire par morceaux de f définie par les p_Δ ; $P_\mathcal{T}$ est définie sur toute l'enveloppe convexe de $\mathcal{S}$. Alors,* **l'erreur** *L_2 commise par l'approximation linéaire de f suivant la triangulation $\mathcal{T}$ est donnée par la formule :*

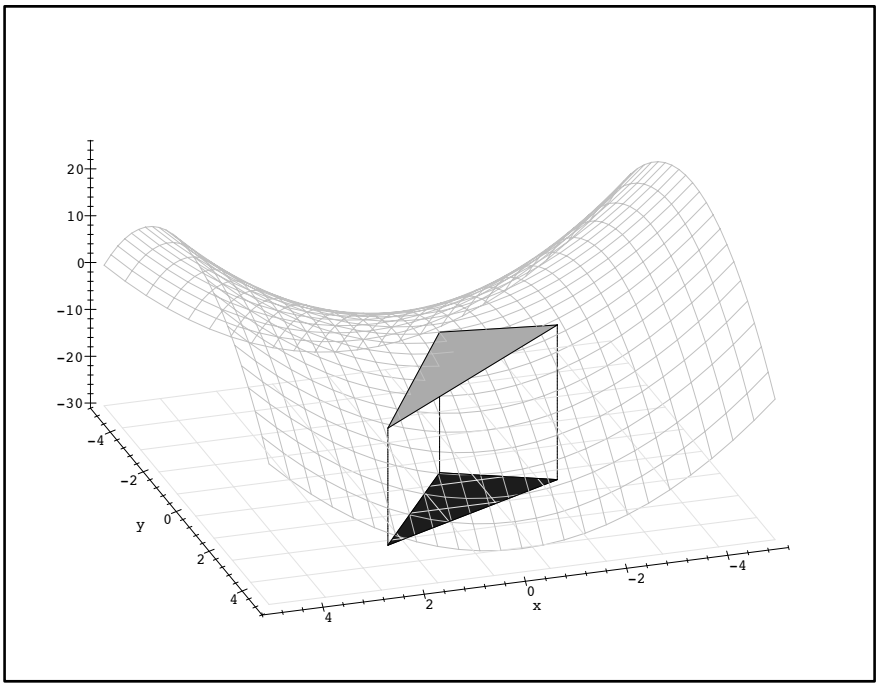
$$\begin{aligned} e(f,\mathcal{T}) &= \|f - P_\mathcal{T}\|_2 \\ &= \sqrt{\iint_\Omega (f(x,y) - P_\mathcal{T}(x,y))^2\,dx\,dy} \\ &= \sqrt{\sum_{\Delta\in\mathcal{T}} \iint_\Delta (f(x,y) - p_\Delta(x,y))^2\,dx\,dy} \\ &= \sqrt{\sum_{\Delta\in\mathcal{T}} \epsilon(f,\Delta)}, \end{aligned}$$

où

$$\epsilon(f,\Delta) = \iint_\Delta (f(x,y) - p_\Delta(x,y))^2\,dx\,dy.$$

Le choix d'un critère s'est tout naturellement porté sur la norme L_2 car elle est la norme la plus fréquemment utilisée dans le cadre des fonctions à plusieurs variables. Par ailleurs, le but étant d'aboutir à des résultats exacts, dont la justification passe par le calcul d'expressions formelles, l'introduction de valeurs absolues (cas de la norme L_1) ou de degrés trop importants (cas des normes L_p, $p > 2$) aurait impliqué l'apparition d'expressions beaucoup trop compliquées, voire inutilisables.

Figure 2.3 – un triangle du plan est relevé sur le paraboloïde hyperbolique ; on se donne alors une approximation linéaire locale de la surface.



Cependant, une étude expérimentale sera menée pour la norme L_1, et on a vu, dans l'état de l'art, qu'il existait des travaux traitant de la norme L_∞.

2.2 Présentation du paraboloïde hyperbolique

De toutes les quadriques, le paraboloïde hyperbolique est une surface particulièrement intéressante du fait de sa non convexité ; par ailleurs, le paragraphe 1.1.1.3 a montré le lien entre la triangulation de Delaunay et le paraboloïde de révolution, et la généralisation de la construction d'une triangulation L_p optimale pour les quadriques convexes. La question logique qui se pose est alors de savoir ce qui se passe pour les quadriques non convexes : est-il possible de leur trouver une triangulation optimale (au moins pour la norme L_2) ?

Nous allons faire ici quelques rappels sur le paraboloïde hyperbolique (pour plus de détails, voir, par exemple, [LFA77]).

2.2.1 Description générale

2.2.1.1 Équation générale

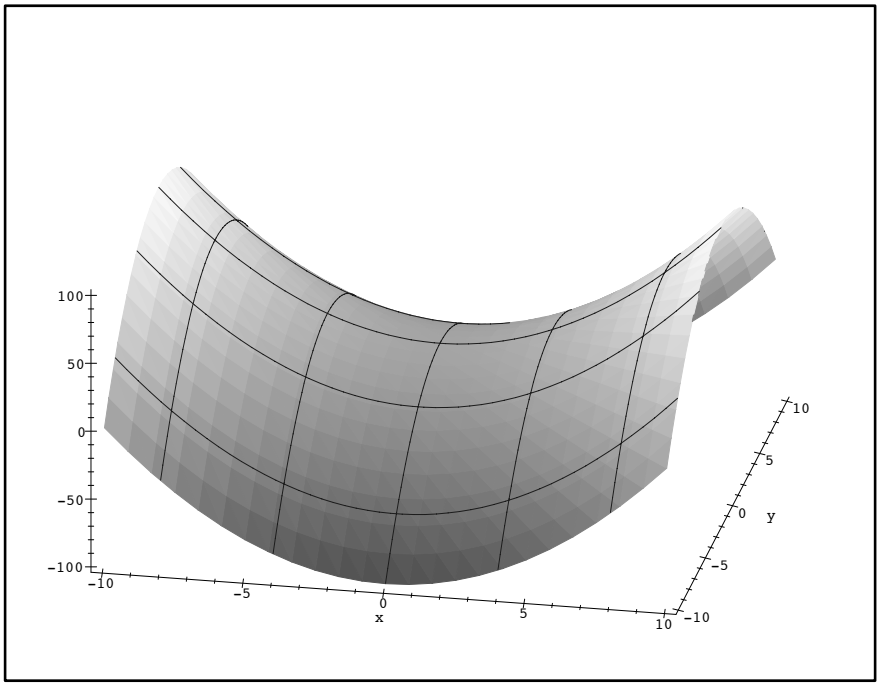
Comme nous l'avons vu, notre surface fait partie de la famille des quadriques, et, plus précisément, le paraboloïde hyperbolique peut être classé parmi les parabo-

loïdes.

définition 2.7 *Une partie $\mathcal{F}$ de l'espace affine euclidien de dimension 3 est appelée un* paraboloïde *s'il existe un repère orthonormal de centre O dans lequel $\mathcal{S}$ admette une équation de la forme :*

$$z = \alpha x^2 + \beta y^2, \quad \text{avec } \alpha \neq 0 \text{ et } \beta \neq 0. \tag{2.1}$$

Figure 2.4 – un paraboloïde hyperbolique et quelques paraboles, intersections de celui-ci avec des plans de la forme $x = c^{te}$ ou $y = c^{te}$.



On remarquera que tous les paraboloïdes admettent 2 plans de symétries, ceux d'équations $x = 0$ et $y = 0$, un axe de symétrie, la droite (Oz), mais aucun centre de symétrie. Par ailleurs, leurs intersections avec les plans d'équations $x = c^{te}$ ou $y = c^{te}$ sont des paraboles (figure 2.4).

Si dans l'équation (2.1), on prend α et β de signes opposés, on peut se ramener au cas où $\alpha > 0$ et $\beta < 0$. Alors, l'équation deviendra :

$$z = \frac{x^2}{a^2} - \frac{y^2}{b^2}, \quad \text{avec } a > 0 \text{ et } b > 0. \tag{2.2}$$

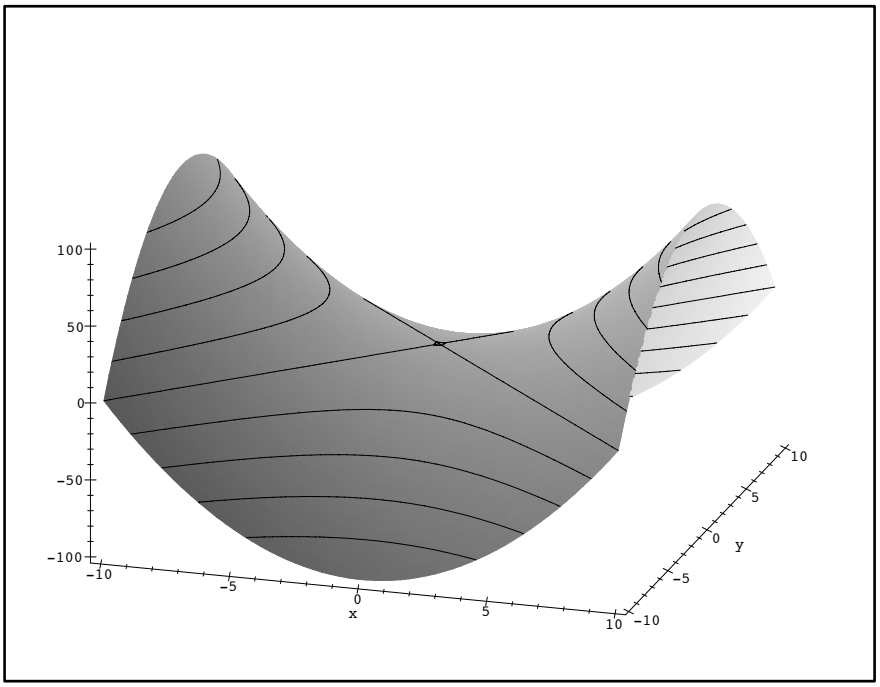
Un paraboloïde possédant une équation de cette forme est dit *hyperbolique*.

2.2.1.2 Intersection avec un plan

La section du paraboloïde hyperbolique avec un plan $z = \lambda$ est une hyperbole si $\lambda \neq 0$, la réunion de deux droites si $\lambda = 0$. De plus, si $\lambda \neq 0$, le grand axe de

l'hyperbole, intersection du plan avec la surface, est parallèle à (Ox) si $\lambda > 0$, et parallèle à (Oy) si $\lambda < 0$ (voir figure 2.5).

Figure 2.5 – un paraboloïde hyperbolique et quelques hyperboles, intersections de celui-ci avec des plans de la forme $z = \lambda$.



Par ailleurs, il existe une propriété du paraboloïde hyperbolique unité qui nous servira plus tard (cette propriété peut être étendue à tout autre paraboloïde hyperbolique pour un autre type d'hyperboles du plan).

propriété 2.1 *toute hyperbole du plan de base, d'asymptotes parallèles aux bissectrices des axes (c'est-à-dire toute hyperbole* équilatère*) peut être définie comme la projection de l'intersection du paraboloïde hyperbolique unité d'équation $z = x^2 - y^2$ avec un plan.*

démonstration :

soit l'hyperbole équilatère d'équation $(x - x_0)^2 - (y - y_0)^2 = p$, où p est un nombre réel quelconque.
On cherche le plan d'équation $z = \lambda x + \mu y + \nu$ dont l'intersection avec la surface $z = x^2 - y^2$ va bien donner l'hyperbole choisie (par projection dans le plan de base).
Cela revient à résoudre le système :

$$\begin{cases} z &= x^2 - y^2 \\ z &= \lambda\, x + \mu\, y + \nu \end{cases}$$

ce qui donne

$$\left(x-\frac{\lambda}{2}\right)^2-\left(y+\frac{\mu}{2}\right)^2=\nu+\frac{\lambda^2-\mu^2}{4},$$

et, par identification avec l'hyperbole choisie

$$\left\{\begin{array}{lcl}\lambda&=&2\,x_0\\ \mu&=&-2\,y_0\\ \nu&=&p+{x_0}^2-{y_0}^2\end{array}\right.$$

Finalement, λ, μ et ν existent toujours et sont uniques. □

2.2.2 Génératrices du paraboloïde hyperbolique

Pour trouver les équations des familles génératrices, nous allons écrire (2.2) sous la forme

$$\begin{array}{rcl}z&=&\dfrac{x^2}{a^2}-\dfrac{y^2}{b^2}\\ &=&\left(\dfrac{x}{a}+\dfrac{y}{b}\right)\left(\dfrac{x}{a}-\dfrac{y}{b}\right).\end{array}$$

On définit ainsi deux familles de droites :

– $\mathcal{D}_\lambda$, $\lambda\in\mathbb{R}$ d'équations

$$\left\{\begin{array}{lcl}\lambda&=&\dfrac{x}{a}+\dfrac{y}{b}\\ z&=&\lambda\left(\dfrac{x}{a}-\dfrac{y}{b}\right)\end{array}\right.$$

– et $\mathcal{D}'_\mu$, $\mu\in\mathbb{R}$ d'équations

$$\left\{\begin{array}{lcl}\mu&=&\dfrac{x}{a}-\dfrac{y}{b}\\ z&=&\mu\left(\dfrac{x}{a}+\dfrac{y}{b}\right)\end{array}\right.$$

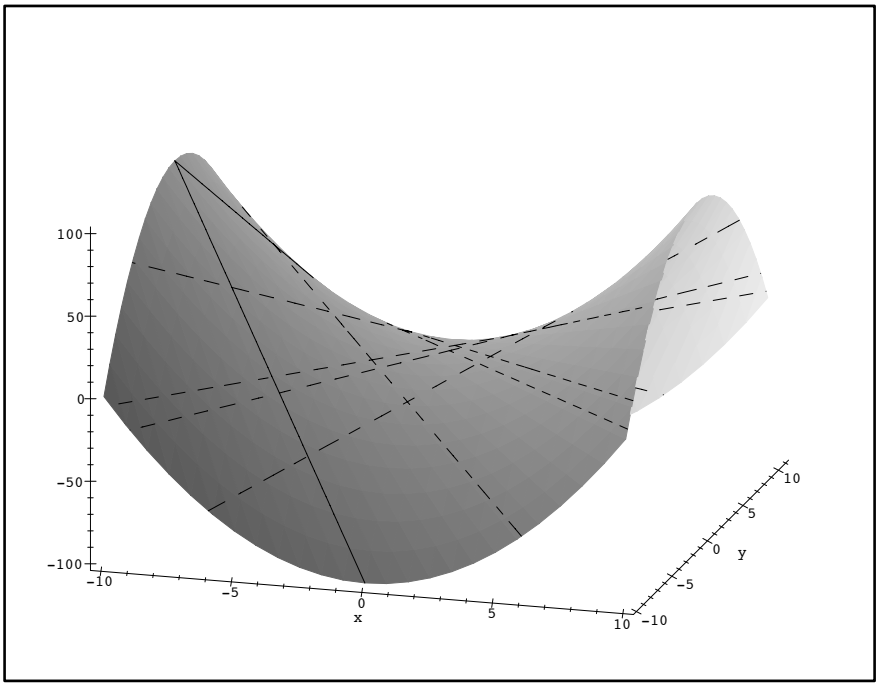
Ces deux familles sont clairement contenues dans le paraboloïde hyperbolique $\mathcal{S}$ d'équation (2.2) ; on dit que ce sont des *génératrices* de $\mathcal{S}$.

Ces familles possèdent un certain nombre de propriétés, que l'on peut retrouver chez les génératrices d'autres quadriques.

propriété 2.2 *Chaque génératrice de l'un des systèmes rencontre chaque génératrice de l'autre système.*

propriété 2.3 *Deux génératrices distinctes d'un même système sont toujours non coplanaires.*

Figure 2.6 – quelques génératrices du paraboloïde hyperbolique unité.



propriété 2.4 *Par chaque point du paraboloïde hyperbolique, il passe une génératrice de chaque système et une seule.*

propriété 2.5 *Le paraboloïde hyperbolique ne contient pas d'autre droite que ses génératrices.*

On peut enfin remarquer que les génératrices appartenant à $\mathcal{D}_\lambda$ sont parallèles au plan d'équation $\frac{x}{a} - \frac{y}{b} = 0$, et celles de $\mathcal{D}'_\lambda$ sont parallèles au plan $\frac{x}{a} + \frac{y}{b} = 0$.

2.2.3 Paramétrisations

Comme toute quadrique, le paraboloïde hyperbolique admet plusieurs paramétrisations. L'une d'elles permet de remarquer que cette surface est bien connexe ; en effet, elle peut être considérée comme la réunion de deux ensembles admettant respectivement les paramétrisations

$$\left\{ \begin{array}{rcccl} x & = & \dfrac{a\,u}{2}(e^v + e^{-v}) & = & a\,u\,\mathrm{ch}(v) \\ y & = & \dfrac{b\,u}{2}(e^v - e^{-v}) & = & b\,u\,\mathrm{sh}(v) \\ z & = & u^2 & & \end{array} \right. ,$$

et

$$\begin{cases} x &= \dfrac{a\,u}{2}(e^v - e^{-v}) &= a\,u\,\mathrm{sh}(v) \\ y &= \dfrac{b\,u}{2}(e^v + e^{-v}) &= b\,u\,\mathrm{ch}(v) \\ z &= -u^2 \end{cases},$$

où $u \in \mathbb{R}$ et $v \in \mathbb{R}$. Ces deux ensembles ayant en commun l'origine, on en déduit la connexité du paraboloïde hyperbolique.

Comme le paraboloïde hyperbolique est une surface doublement réglée, il est possible d'en trouver une représentation paramétrique qui tienne directement compte de ses familles de droites génératrices. Soit $z = \frac{x^2}{a^2} - \frac{y^2}{b^2}$ l'équation de la quadrique considérée. On a alors :

$$z = \left(\frac{x}{a} - \frac{y}{b}\right)\left(\frac{x}{a} + \frac{y}{b}\right),$$

ce qui implique que l'intersection du plan $\frac{x}{a} - \frac{y}{b} = u_0$ avec la surface est la droite donnée par les plans :

$$\begin{cases} u_0 &= \dfrac{x}{a} - \dfrac{y}{b} \\ z &= u_0\left(\dfrac{x}{a} + \dfrac{y}{b}\right) \end{cases}$$

On est conduit à poser :

$$\begin{cases} u &= \dfrac{x}{a} - \dfrac{y}{b} \\ v &= \dfrac{x}{a} + \dfrac{y}{b} \end{cases},$$

de telle sorte que :

$$x = \frac{a}{2}(u+v),\ y = \frac{b}{2}(v-u),\ z = u.v \quad (u \in \mathbb{R}, v \in \mathbb{R}).$$

2.2.4 Utilité des familles génératrices

Dans l'approche qui nous intéresse, ce sont les approximations linéaires choisies qui vont jouer un role primmordial (c'est-à-dire la projection sur le plan de la triangulation construite).

Comme la surface contient des droites, il semble raisonnable de penser que ces droites vont aider à savoir si un triangle (de l'espace) est bon ou pas pour approcher le paraboloïde hyperbolique : par exemple, si une arête du triangle a pour support une des génératrices de la surface, il est clair que l'erreur d'approximation linéaire le long de cette arête sera nulle.

Comme le problème de départ est de construire une triangulation des sommets du plan, on va devoir considérer ce que donnent les génératrices du paraboloïde

hyperbolique reprojetées dans le plan ; on obtient ainsi les équations :

$$x + y = \alpha \quad \text{et} \quad x - y = \beta,$$

où α et β sont quelconques (même égaux à 0).

propriété 2.6 *dans le plan de base, les familles génératrices correspondent à toutes les droites parallèles aux deux bissectrices du repère, que l'on suppose orthonormé.*

Reste à voir que la réciproque est effectivement vraie.

propriété 2.7 *toute droite du plan parallèle à une des bissectrices des axes se relève sur le paraboloïde hyperbolique selon une droite appartenant à une des familles génératrices de la surface.*

démonstration :

on va se limiter ici aux parallèles à la deuxième bissectrice $x + y = 0$, en remarquant que le raisonnement est analogue pour les droites parallèles à la première bissectrice.
Soit donc un point $a = (a_1, a_2)$; on choisit le point b d'abscisse 0 tel que la droite (ab) soit parallèle à la droite d'équation $x + y = 0$. b aura donc pour coordonnées $(0, a_1 + a_2)$.
Par relèvement sur le paraboloïde hyperbolique, on obtient les points :

$$A = \begin{pmatrix} a_1 \\ a_2 \\ {a_1}^2 - {a_2}^2 \end{pmatrix} \quad \text{et} \quad B = \begin{pmatrix} 0 \\ a_1 + a_2 \\ -(a_1 + a_2)^2 \end{pmatrix}.$$

Il faut alors vérifier si les points A et B sont bien sur une même génératrice de la famille donnée par

$$\begin{cases} x + y &= \dfrac{1}{\lambda} \\ x - y &= \lambda z \end{cases}.$$

Deux cas sont possibles :

- si $a_1 + a_2 \neq 0$.
  on a alors $\lambda = \frac{1}{a_1 + a_2}$.
  il faut maintenant voir si la deuxième équation est bien vérifiée pour chaque point :
  - pour le point A :

$$\frac{x - y}{\lambda} = (a_1 - a_2)(a_1 + a_2) = {a_1}^2 - {a_2}^2 = z_A$$

  - pour le point B :

$$\frac{x - y}{\lambda} = -(a_1 + a_2)(a_1 + a_2) = -(a_1 + a_2)^2 = z_B$$

– si $a_1 + a_2 = 0$.
on a $b = (0, 0)$, et donc la droite (ab) est confondue avec la deuxième bissectrice des axes du plan.
son relèvement sur le paraboloïde hyperbolique donne la droite d'équations

$$\begin{cases} z & = & 0 \\ x+y & = & 0 \end{cases},$$

qui correspond bien à une droite de l'une des familles génératrices de la surface.

Puisque la droite (AB) est inncluse dans la surface, tout point aligné avec (ab) se relève sur un point de (AB). □

Chapitre 3

Recherche pratique

Rien ne laissant entrevoir un moyen radical et totalement formel de trouver les courbes de séparation, une première approche a été d'observer des courbes obtenues de manière expérimentale pour certaines configurations de points. Le problème a donc tout naturellement donné naissance à l'élaboration d'un programme destiné à dessiner ces courbes ; ce programme sera décrit en section 3.1, puis nous verrons comment il nous aidera à apporter quelques observations.

Évidemment, même si ces figures peuvent aider à déduire les équations recherchées, elles ne donnent aucune preuve. Une seconde approche consistera alors à utiliser un logiciel de calcul formel qui nous servira de confirmation ; tout ce processus sera analysé de manière rigoureuse dans la section 3.2 qui sera suivie d'exemples de courbes pour des configurations de points particulières.

Toutes ces premières recherches ont été menées sur le paraboloïde hyperbolique unité, avec le critère d'erreur basé sur la norme L_2. On verra cependant dans la section 3.4 qu'il est possible d'obtenir, pratiquement mais pas formellement, des résultats pour des critères fondés sur les normes L_1 et L_∞ qui présentent un intérêt certain, ne serait-ce que pour leur interprétation géométrique : l'erreur L_1 représente le volume qui sépare la surface de son approximation, et on peut alors penser que si l'approximation est bonne, c'est qu'elle ne doit pas être trop éloignée de la fonction, ce qui implique un faible volume. L'erreur L_∞, elle, va donner le plus grand écart ponctuel qui existe entre un point de la surface et le point correspondant de l'approximation ; à nouveau, une erreur faible laissera penser que approximation et fonction sont proches.

3.1 Observations

On est ramené au problème suivant : étant donnés trois sommets du plan a, b et c, on note A, B et C leurs relevés respectifs sur la surface $z = x^2 - y^2$. Soit $\mathcal{P}_{\Delta ABC}$ le plan qui contient les points A, B et C ; son équation est donnée par $\mathcal{P}_{\Delta ABC} : z = p_{\Delta ABC}(x, y)$. L'erreur L_2 commise localement par l'approximation linéaire du paraboloïde hyperbolique sur le triangle ΔABC est définie par :

$$\begin{aligned} e_2(\Delta abc) &= \sqrt{\epsilon(\Delta abc)} \\ \epsilon(\Delta abc) &= \iint_{\Delta abc} \left(p_{\Delta ABC}(x, y) - (x^2 - y^2)\right)^2 \, dx \, dy. \end{aligned}$$

Il est possible de calculer numériquement le résultat de cette double intégrale en passant soit par des bibliothèques Fortran (*MatLab*), soit par des logiciels de calcul formel (*Maple*). De simples expérimentations permettent d'observer le phénomène suivant :

observation 3.1 *L'erreur L_2 commise par l'approximation linéaire spatiale du paraboloïde hyperbolique pour un triangle Δabc est invariante par translation de a, b et c, le plan servant d'approximation étant celui qui contient leurs relevés A, B et C.*

Autrement dit, si on dispose de trois points quelconques a, b, et c du plan, chercher l'erreur d'approximation linéaire produite par le triangle relevé sur la surface revient au même que chercher l'erreur produite par le relèvement du triangle constitué des points o, $b - a$ et $c - a$, où o désigne le centre du repère du plan de base.

3.1.1 Le problème observé

Soient trois points du plan de base $o = (0, 0)$ (d'après l'observation 3.1, le fait de ramener un des points à l'origine ne change rien), $b = (x_b, y_b)$ et $c = (x_c, y_c)$. Les trois points relevés sur le paraboloïde hyperbolique sont notés $O = (0, 0, 0)$, $B = (x_b, y_b, z_B)$ et $C = (x_c, y_c, z_C)$ où $z_B = {x_b}^2 - {y_b}^2$ et $z_C = {x_c}^2 - {y_c}^2$.

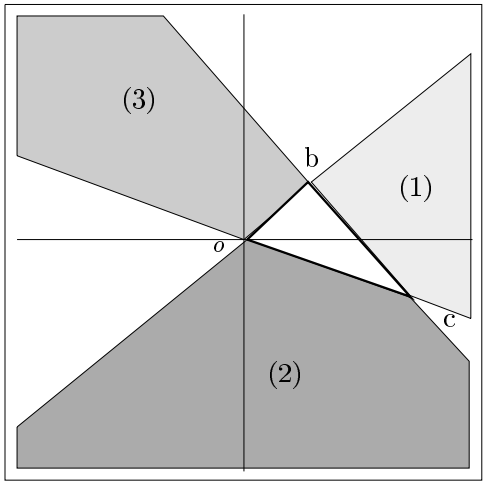
Soit maintenant d un quatrième point du plan. La position de d relative au triangle Δobc divise naturellement le plan en 7 zones (figure 3.1) qui correspondent aux cas suivants :

- d est dans le triangle Δobc : lors de la triangulation, le triangle Δobc va disparaître au profit des triangles Δobd, Δocd et Δbcd. (figure 3.2, premier cas).

- d est dans une des zones non colorées, extérieures au triangle : une seule triangulation des 4 points sera possible (figure 3.2 deuxième cas).

- d est dans la zone colorée marquée (1) (figure 3.3) : cette zone sera dorénavant appelée "zone d'influence" associée au point o, car elle est géométriquement opposée à o par l'intermédiaire de l'arête $[bc]$.
  Deux triangulations sont ici possibles : d'une part celle qui contient les triangles Δobc et Δbcd, et d'autre part celle qui fait intervenir Δobd et Δocd. Le choix de la triangulation se fera en fonction de celle qui minimisera l'erreur

Figure 3.1 – emplacements possibles d'un point autour d'un triangle.



d'approximation produite sur le paraboloïde hyperbolique.
Ainsi, le point d appartiendra à la courbe de séparation du triangle Δobc si et seulement si les deux triangulations possibles sont équivalentes, c'est-à-dire si leurs erreur L_2 sont égales.

- d est dans la zone colorée marquée (2) : idem que le cas précédent sauf que les deux triangulations possibles sont Δobc et Δocd ou Δbod et Δbcd (zone d'influence de b).

- d est dans la zone colorée marquée (3) : on est dans la zone d'influence de c. Les deux triangulations possibles sont Δobc et Δobd ou Δcod et Δcbd.

Il semble donc raisonnable de penser que les seuls morceaux intéressants de la courbe de séparation apparaitront dans les zones d'influences des sommets du triangle. Cependant, comme il n'y a aucune raison que cette courbe ne soit pas continue, tout laisse supposer qu'elle existe dans les autres zones : simplement, pour notre problème, elle ne sera d'aucune utilité puisque, dans ces cas-là, une seule triangulation est possible. Nous verrons cependant plus loin qu'il existe une signification à la présence de morceaux de la courbe de séparation dans ces zones.

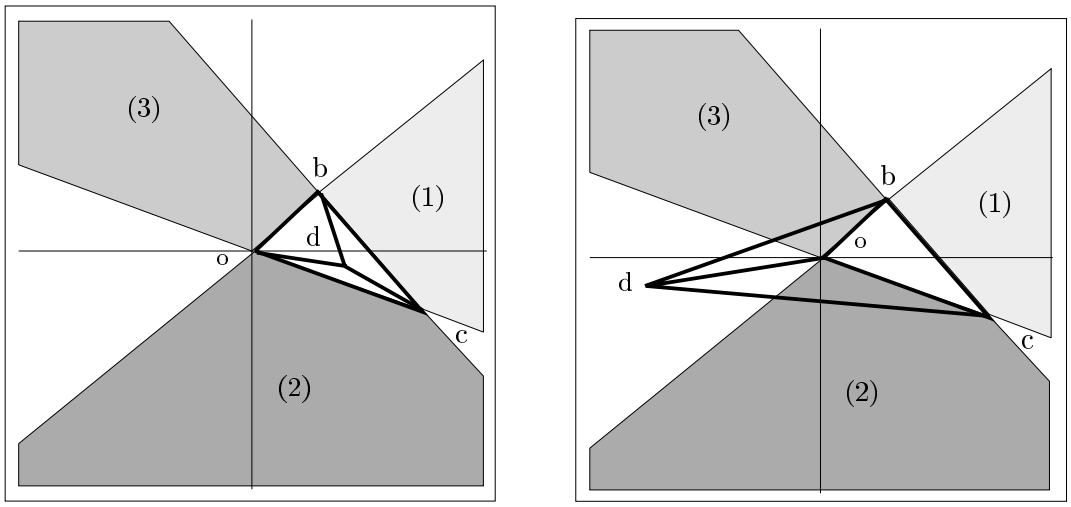
Un programme de recherche de la courbe de séparation d'un triangle a été réalisé en utilisant le procédé suivant :

- soient trois points du plan (on peut en choisir un en $(0, 0)$).

- on fait varier un quatrième point dans les zones où deux triangulations sont possibles (celles que l'on a nommées "zones d'influences" des sommets du triangle de départ).

- on cherche la courbe de séparation en calculant explicitement les erreurs obtenues par les deux triangulations en diverses positions de d ; les endroits où les deux erreurs sont identiques (à un facteur d'erreur près) indiquent les positions des points de la courbe de séparation.

Figure 3.2 – des cas où seule une triangulation est possible.



Différents résultats peuvent être observés sur la figure 3.4 pour des cas a priori quelconques, et sur la figure 3.5 pour des cas dégénérés.

L'examen des courbes obtenues sur de très nombreux exemples (dont certains sont reproduits en section 3.3) permet d'effectuer les observations suivantes :

observation 3.2 *La courbe de séparation d'un triangle semble être, pour des cas non dégénérés, la réunion de deux hyperboles.*

observation 3.3 *Des dégénérescences apparaissent lorsqu'une arête du triangle est portée par une droite parallèle à l'une des bissectrices des axes, c'est-à-dire, d'après la propriété 2.7, lorsqu'une arête se relève sur un segment tracé sur le paraboloïde hyperbolique.*

Il reste maintenant à trouver les équations exactes de ces deux hyperboles.

3.1.2 De l'expérimentation aux courbes

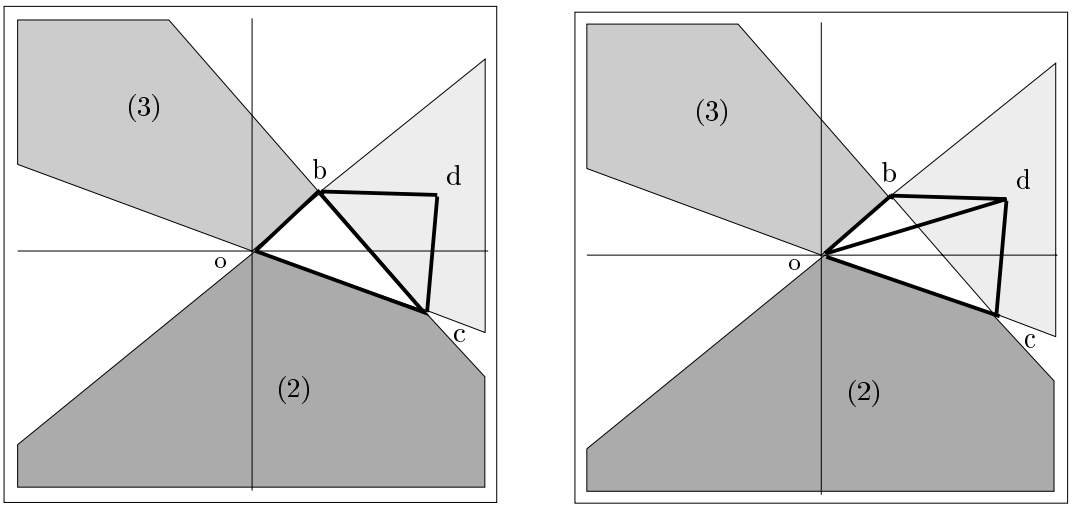
Dans ce paragraphe, $\mathcal{H}_1$ et $\mathcal{H}_2$ désigneront les hyperboles constituant la courbe de séparation recherchée.

3.1.2.1 La première hyperbole

L'étude des courbes de séparation montre clairement qu'une des 2 hyperboles ($\mathcal{H}_1$) passe par les trois sommets du triangle ; cela n'apparaît pas directement sur les figures présentées car seules les zones du plan où 2 triangulations sont possibles ont été utilisées. Cependant, une extrapolation sur les morceaux reproduits permet de percevoir le résultat annoncé.

Une deuxième remarque concerne les asymptotes de $\mathcal{H}_1$: elles sont toujours parallèles aux bissectrices des axes du repère (à nouveau, les familles génératrices du

Figure 3.3 – d est dans la zone d'influence de o: deux triangulations possibles.



paraboloïde hyperbolique semblent avoir un role à jouer). Cela implique que, dans l'équation de l'hyperbole, les coefficients en x^2 et y^2 seront égaux.

On parvient ainsi naturellement à une nouvelle observation :

observation 3.4 *La première hyperbole, partie de la courbe de séparation du triangle Δobc, est l'hyperbole équilatère qui passe par les points o, b et c.*

Déterminer explicitement $\mathcal{H}_1$ devient alors très simple puisque c'est l'unique fonction quadratique d'équation $(x - x_0)^2 - (y - y_0)^2 = p$ vérifiée par les points $(0,0)$, (x_b, y_b) et (x_c, y_c), le signe de p dépendant de la directrice de l'hyperbole (parallèle à l'axe des x ou à l'axe des y).

Il reste à exprimer x_0, y_0 et p en fonction de x_b, y_b, x_c et y_c. On trouve :

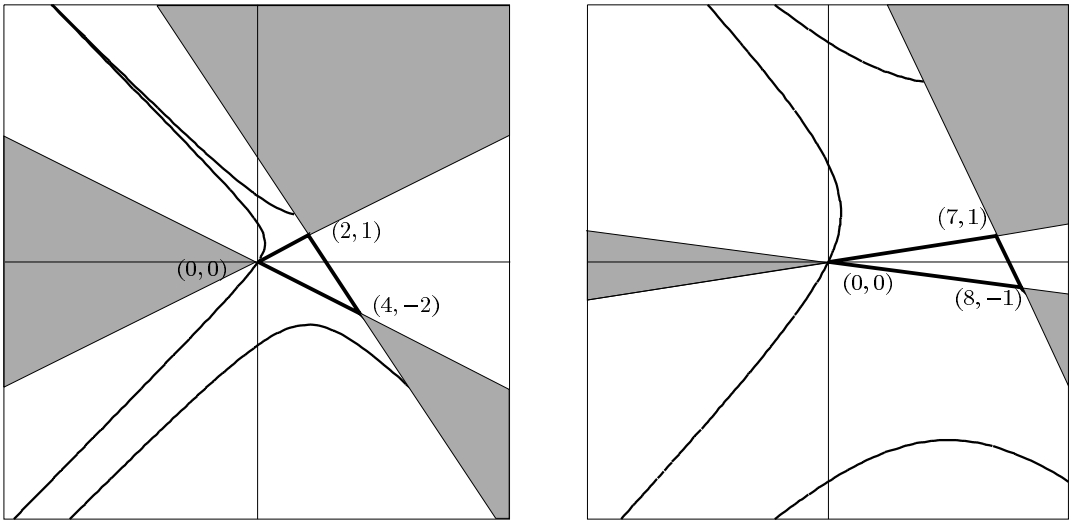
$$\begin{cases} y_0 &= \dfrac{1}{2}\dfrac{x_b\,{x_c}^2 - x_c\,{x_b}^2 + x_c\,{y_b}^2 - x_b\,{y_c}^2}{x_c\,y_b - x_b\,y_c} \\ x_0 &= \dfrac{1}{2}\dfrac{-{x_b}^2\,y_c + {y_b}^2\,y_c + y_b\,{x_c}^2 - y_b\,{y_c}^2}{x_c\,y_b - x_b\,y_c} \\ p &= -\dfrac{1}{4}\dfrac{({x_b}^2 - {y_b}^2)\,({x_c}^2 - {y_c}^2)\,((x_b - x_c)^2 - (y_b - y_c)^2)}{(x_c\,y_b - x_b\,y_c)^2} \end{cases}$$

Ces résultats sont tout de même peu satisfaisants car difficilement exploitables.

Pour caractériser $\mathcal{H}_1$ de manière plus simple, il faut avoir recours à la propriété 2.1 concernant les hyperboles équilatères et le paraboloïde hyperbolique. Elle nous permet de déduire que la première partie de la courbe de séparation correspond à la projection planaire de l'intersection entre le paraboloïde hyperbolique et un plan de l'espace, qui n'est rien d'autre que plan passant par les relevés des trois points sur la surface. Soit la nouvelle observation :

observation 3.5 *la première hyperbole constituant la courbe de séparation d'un triangle Δ est obtenue en reprojetant dans le plan de base l'intersection du paraboloïde*

Figure 3.4 – exemples de courbes de séparations obtenues.



hyperbolique avec le plan passant par les relevés des sommets de Δ.

Ceci semble bien être l'équivalent de la propriété du cercle vide pour la triangulation de Delaunay, où le cercle (circonscrit au triangle) peut être vu comme la projection de l'intersection entre le paraboloïde de révolution unité et le plan passant par les relevés des 3 sommets du triangle considéré.

3.1.2.2 La deuxième hyperbole

L'interprétation de $\mathcal{H}_2$ est beaucoup plus délicate car cette hyperbole apparait moins caractéristique. On peut seulement remarquer, en observant les figures 3.4 et 3.5 (surtout le deuxième cas dégénéré de 3.5), que ses asymptotes sont à nouveau parallèles aux bissectrices des axes du repère, mais surtout qu'elles se coupent au centre de gravité du triangle de départ ; cette propriété est mise en valeur sur la figure 3.6 où les asymptotes de l'hyperbole sont dessinées.

Le problème est que la donnée des asymptotes, donc du centre de symétrie de la conique, n'est pas suffisante pour trouver l'équation d'une hyperbole. En effet, si on note $g = (x_g, y_g)$ le centre de gravité du triangle de départ, pour le moment on sait seulement que l'équation de $\mathcal{H}_2$ est

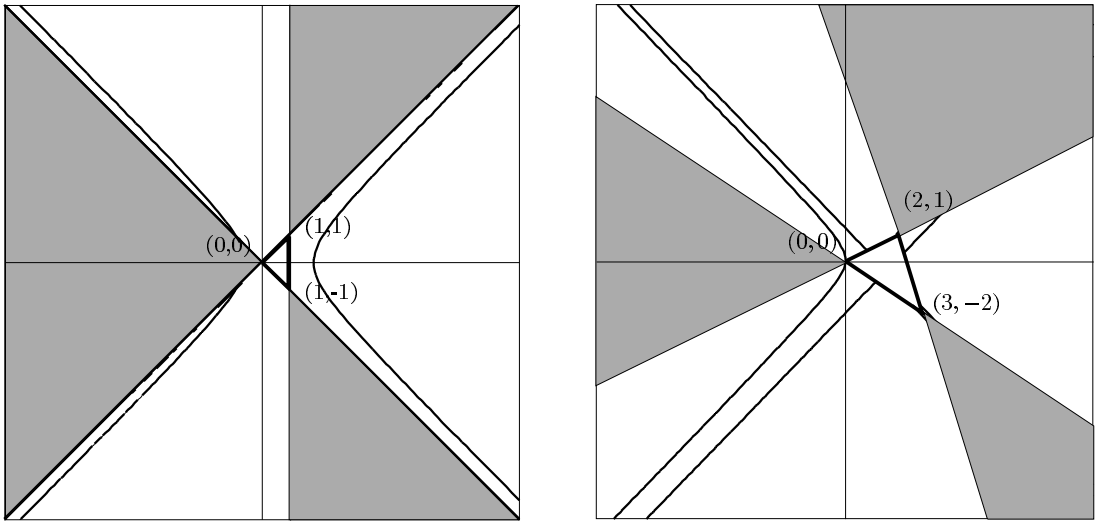
$$(x - x_g)^2 - (y - y_g)^2 = p, \text{ avec } \begin{cases} x_g &= \dfrac{x_b + x_c}{3} \\ y_g &= \dfrac{y_b + y_c}{3} \end{cases}$$

et que celles de ses asymptotes sont :

$$\begin{cases} y &= x + y_g - x_g \\ y &= -x + y_g + x_g \end{cases}.$$

Pour déterminer complètement $\mathcal{H}_2$ et trouver le coefficient manquant p, il n'a pu

Figure 3.5 – exemples de courbes de séparations dégénérées.



être fait appel qu'à l'expérimentation (tout en sachant que la stabilité de l'erreur, et donc des courbes, par translation limite le domaine des recherches).
Pour le calcul de p, on a besoin d'introduire un nouveau point : on note G le relevé de g sur le paraboloïde hyperbolique et H l'isobarycentre des relevés de o, b et c, appelés respectivement O, B et C. On a :

$$\begin{cases} x_G &= \dfrac{x_b + x_c}{3} &= x_g \\ y_G &= \dfrac{y_b + y_c}{3} &= y_g \\ z_G &= x_G^2 - y_G^2 &= \dfrac{(x_b + x_c)^2 - (y_b + y_c)^2}{9} \end{cases}$$

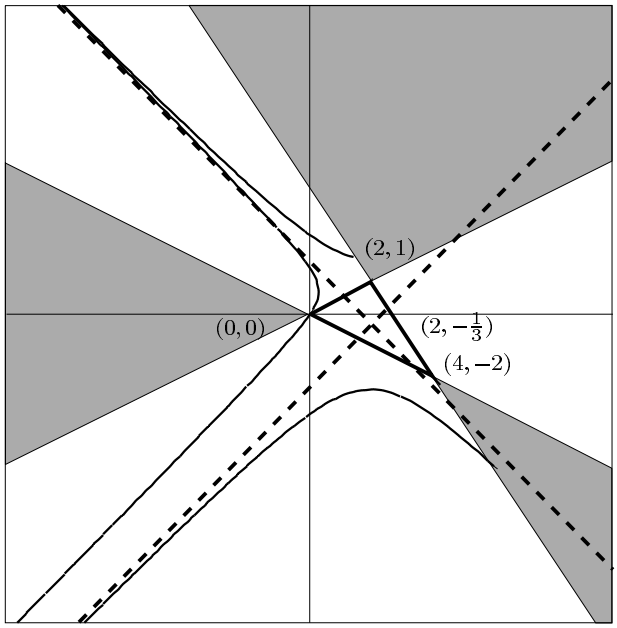
et

$$\begin{cases} x_H &= \dfrac{x_b + x_c}{3} &= x_g \\ y_H &= \dfrac{y_b + y_c}{3} &= y_g \\ z_H &= \dfrac{({x_b}^2 - {y_b}^2) + ({x_c}^2 - {y_c}^2)}{3} \end{cases} .$$

Alors, les expériences montrent que le coefficient p a pour valeur :

$$\begin{aligned} p &= 4\overline{HG} \\ &= 4(z_G - z_H) \\ &= \frac{8}{9}\left({y_b}^2 + {y_c}^2 - y_b y_c - {x_b}^2 - {x_c}^2 + x_b x_c\right). \end{aligned}$$

Figure 3.6 – les asymptotes (en pointillés) de la deuxième hyperbole se coupent au centre de gravité du triangle.



Ainsi, la deuxième hyperbole est donnée par :

observation 3.6 *Avec les notations adoptées précédemment, la deuxième partie de la courbe de séparation d'un triangle est l'hyperbole d'équation :*

$$(x - x_g)^2 - (y - y_g)^2 = 4\overline{HG}.$$

3.1.2.3 La courbe de séparation

Finalement, on vient de voir qu'une recherche expérimentale suffit pour "trouver" la courbe de séparation d'un triangle :

observation 3.7 *Soit Δabc un triangle de sommets $a = (x_a, y_a)$, $b = (x_b, y_b)$ et $c = (x_c, y_c)$, dont les relevés respectifs sur le paraboloïde hyperbolique sont A, B et C. Alors la courbe de séparation de Δabc, pour l'erreur L_2 d'approximation locale par le plan ABC, consiste en deux hyperboles :*

- *l'hyperbole donnée par le système :*

$$\begin{cases} z = x^2 - y^2 \\ z = \alpha\, x + \beta\, y + \gamma \quad (\mathcal{P}) \\ A \in \mathcal{P},\ B \in \mathcal{P},\ C \in \mathcal{P} \end{cases}$$

- *l'hyperbole d'équation :*

$$\left(x - \frac{x_a + x_b + x_c}{3}\right)^2 - \left(y - \frac{y_a + y_b + y_c}{3}\right)^2 = 4\overline{HG}$$

avec

$$G = \begin{pmatrix} \dfrac{x_a + x_b + x_c}{3} \\ \dfrac{y_a + y_b + y_c}{3} \\ {x_G}^2 - {y_G}^2 \end{pmatrix}, \quad H = \begin{pmatrix} \dfrac{x_a + x_b + x_c}{3} \\ \dfrac{y_a + y_b + y_c}{3} \\ \dfrac{z_A + z_B + z_C}{3} \end{pmatrix}$$

3.2 Utilisation du calcul formel

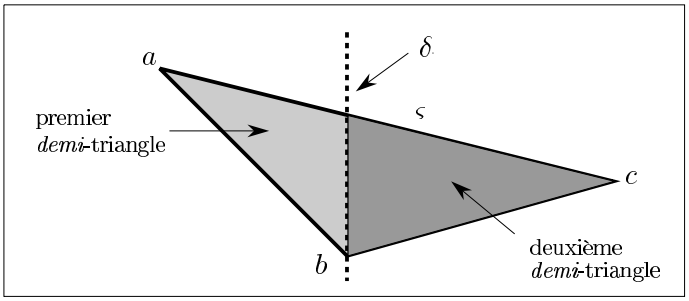
Il parait évident au vue de l'observation 3.7 qu'il devrait être possible d'utiliser un logiciel de calcul formel, tel que *Maple* [CGG+], pour confirmer les équations obtenues pour les courbes de séparation d'un triangle (on peut employer un pluriel, car maintenant on sait qu'il y a effectivement plusieurs courbes). Cette méthode de "confirmation" formelle va utiliser une procédure de calcul symbolique de l'erreur sur un triangle ; à partir de cette procédure, nous verrons qu'il est possible de déduire les erreurs pour les 2 triangulations de 4 points donnés, supposés former un polygone convexe, puis de formuler l'équation des courbes de séparation. De grandes précautions ont été prises sur la façon de calculer les erreurs, puisque c'est de là que les principaux problèmes peuvent surgir. Par ailleurs, *Maple* a besoin d'aide pour simplifier certains calculs, et c'est cela qui va être, en partie, détaillé ici.

Les résultats de cette section ont fait l'objet d'un article présenté à la septième conférence canadienne de géométrie algorithmique (*CCCG7*) [DD95].

3.2.1 Calcul de l'erreur sur un triangle

Le problème se pose en ces termes : étant donnés trois points dans le plan $a = (x_a, y_a)$, $b = (x_b, y_b)$ et $c = (x_c, y_c)$, et A, B et C leurs relevés respectifs sur le paraboloïde hyperbolique, on désire pouvoir évaluer formellement l'erreur L_2 de l'approximation linéaire de $\mathcal{PH}$ sur le triangle ΔABC.

Figure 3.7 – découper un triangle en deux afin de pouvoir mieux intégrer.



Pour cela, il est possible d'utiliser la méthode suivante :

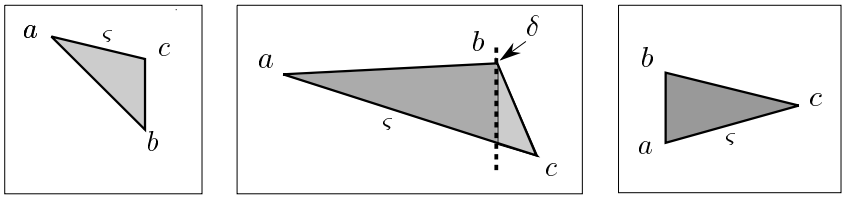
- on donne les sommets dans l'ordre croissant de leur abscisse (dans le cas d'un

calcul réel, on dispose évidemment d'une procédure de tri des points en fonction de leur abscisse).

- on coupe le triangle en deux, par une parallèle à l'axe des ordonnées passant par le sommet intermédiaire du triangle (celui dont l'abscisse n'est pas extrême).
- sur chacun des deux triangles ainsi obtenus, on cherche les équations des droites supports des arêtes non parallèles à l'axe des ordonnées.
- pour parcourir toute la surface définie par un des petits triangles, il suffit de faire varier les x de la valeur minimale à la valeur maximale présentes sur le triangle, et les y d'une arête à l'autre (on prend pour bornes les équations des arêtes : y est ainsi défini en fonction de x).

Pour intégrer, on choisit toujours pour borne supérieure en y, l'arête ς qui présente toute les valeurs possibles des abscisses, c'est-à-dire celle qui a pour extrémités les points de plus petite et de plus grande abscisse (notation adoptée sur les figures 3.7 et 3.8).

Figure 3.8 – les cas possibles de découpage d'un triangle.



Le calcul de l'erreur par un tel processus fait ressurgir certaines caractéristiques :

– si le triangle a un de ses côtés parallèle à l'axe des ordonnées, on n'a pas besoin de séparer le triangle en deux (premier et troisième cas de la figure 3.8).

– comme le signe de l'intégrale est tributaire de l'ordre dans lequel on considère les arêtes, il va être lié à la position de ς relativement aux autres côtés : si ς est au-dessus des deux autres arêtes, le signe de l'intégrale sera positif, sinon il sera négatif (dans le premier cas, lors de l'intégration on ira dans le sens des y croissants).

Il faut bien remarquer que pour un même triangle, le signe de l'intégrale reste le même sur chaque "demi"-triangle ; en effet, la position relative de ς ne change pas d'un demi-triangle à l'autre (voir les figures 3.7 ou 3.8 pour s'en convaincre).

À l'image des figures 3.7 et 3.8, on adoptera désormais les notations suivantes :

– a est le sommet le plus à gauche du triangle, c'est-à-dire celui de plus petite abscisse.

– b est le sommet intermédiaire du triangle : x_b n'est pas une abscisse extrême de Δabc.

- c est le sommet le plus à droite, sommet de plus grande abscisse.
- si a et b ont même abscisse, on note a le point de plus petite ordonnée.
- si b et c ont même abscisse, on note c le point de plus grande ordonnée.
- δ est la droite de séparation du triangle Δabc en deux "demi"-triangles : c'est la parallèle à l'axe des ordonnées qui passe par le point b (droite d'équation $x = x_b$).
- pour les deux "demi"-triangles, la droite support inférieure des y sera (ac) (choisie de manière arbitraire, seulement parce qu'elle est support de l'arête qui contient toutes les abscisses possibles du triangle) ; on notera son équation $y = \alpha_2 x + \beta_2$.
- pour le premier (resp. le deuxième) demi-triangle, la droite support supérieure des y sera (ab) (resp. (bc)) ; son équation sera notée $y = \alpha_3 x + \beta_3$ (resp. $y = \alpha_1 x + \beta_1$).

Un tel choix de notations implique évidemment que l'erreur, toujours positive en pratique, peut être opposée au résultat du calcul par la procédure formelle.

Si on note $z = p_{\Delta abc}(x, y)$ l'équation du plan de l'espace à 3 dimensions qui contient les points A, B et C, pour l'exemple choisi sur la figure 3.7, l'erreur d'interpolation commise par l'approximation linéaire locale du paraboloïde hyperbolique par le triangle ΔABC sera :

$$\epsilon_{Maple}(\Delta abc) \quad = \quad I_1 + I_2$$

avec :

$$\begin{cases} I_1 &= \displaystyle\int_{x=x_a}^{x=x_b} \int_{y=\alpha_3 x+\beta_3}^{y=\alpha_2 x+\beta_2} \left(p_{\Delta ABC}(x,y) - (x^2 - y^2)\right)^2 \, dy\, dx \\ I_2 &= \displaystyle\int_{x=x_b}^{x=x_c} \int_{y=\alpha_1 x+\beta_1}^{y=\alpha_2 x+\beta_2} \left(p_{\Delta ABC}(x,y) - (x^2 - y^2)\right)^2 \, dy\, dx \end{cases}$$

En effet, sur cet exemple, le segment qui contient toutes les abscisses est au-dessus des 2 autres arêtes du triangles ; comme dans I_1 et I_2, on fait varier x dans le sens des abscisses croissantes, il faut faire varier les y dans le sens des ordonnées croissantes pour que les doubles intégrales renvoient bien une valeur positive.

En réalité, la procédure *Maple* associée au calcul du carré de l'erreur pour un triangle fonctionne un peu différemment ; elle suppose toujours que les points sont donnés dans cette configuration. Ce qui signifie que le résultat renvoyé par $\epsilon_{Maple}(\Delta abc)$ implique les hypothèses :

$$\begin{cases} x_a \leqslant x_b \leqslant x_c \\ [ac] \text{ au-dessus de } [ab] \text{ et } [bc] \end{cases} .$$

La première hypothèse n'est pas contraignante, et pour la deuxième si on désire se placer dans le cas contraire, il suffit de récupérer l'opposé du résultat renvoyé par la procédure.

3.2.2 Invariance de l'erreur par translation

propriété 3.1 *L'erreur L_2 commise par l'approximation linéaire spatiale du paraboloïde hyperbolique unité pour un triangle est invariante par translation dans le plan (xOy) des sommets dont les relevés composent le triangle choisi.*

démonstration :

Soient $a = (x_a, y_a)$, $b = (x_b, y_b)$ et $c = (x_c, y_c)$ des points de $\mathbb{R}^2$ de relevés sur le paraboloïde hyperbolique unité A, B, C. On suppose, sans perte de généralité, que les points du plan sont donnés par ordre d'abscisses croissantes. On note $t_{\vec{u}}$ la translation de $\mathbb{R}^2$ de vecteur $\vec{u} = (u_x, u_y)$, $\epsilon_{Maple}(\Delta abc)$ l'erreur L_2 d'approximation locale de $\mathcal{PH}$ par le triangle ΔABC et $\epsilon_{Maple}(t_{\vec{u}}(\Delta abc))$ l'erreur d'approximation pour le triangle image de Δabc par la translation $t_{\vec{u}}$. Alors, les calculs effectués par *Maple* montrent que :

$$
\begin{aligned}
\epsilon_{Maple}(\Delta abc) &= \epsilon_{Maple}(t_{\vec{u}}(\Delta abc)) \\
&= \frac{1}{180}\,(\,y_b\,x_a - y_c\,x_a - x_b\,y_a + x_c\,y_a + x_b\,y_c - x_c\,y_b\,) \\
&\quad (3\,x_a^4 - 6\,x_a^2\,y_a^2 + 3\,y_a^4 - 6\,x_c\,x_a^3 - 6\,x_b\,x_a^3 + 6\,y_c\,x_a^2\,y_a \\
&\quad +6\,y_b\,x_a^2\,y_a + 6\,x_b\,x_a\,y_a^2 + 6\,x_c\,x_a\,y_a^2 - 6\,y_c\,y_a^3 - 6\,y_b\,y_a^3 \\
&\quad +9\,x_c^2\,x_a^2 - 5\,y_c^2\,x_a^2 + 9\,x_b^2\,x_a^2 - 5\,y_b^2\,x_a^2 + 4\,y_c\,y_b\,x_a^2 \\
&\quad -8\,x_b\,y_b\,x_a\,y_a - 4\,x_c\,y_b\,x_a\,y_a - 4\,x_b\,y_c\,x_a\,y_a - 8\,x_c\,y_c\,x_a\,y_a \\
&\quad -5\,x_c^2\,y_a^2 - 5\,x_b^2\,y_a^2 + 4\,x_c\,x_b\,y_a^2 + 9\,y_c^2\,y_a^2 + 9\,y_b^2\,y_a^2 \\
&\quad +6\,y_c^2\,x_c\,x_a + 4\,x_c\,y_b^2\,x_a + 6\,y_b^2\,x_b\,x_a - 4\,x_b\,y_c\,y_b\,x_a \\
&\quad +4\,x_b\,y_c^2\,x_a - 6\,x_c^3\,x_a - 4\,x_c\,y_c\,y_b\,x_a - 6\,x_b^3\,x_a - 6\,y_b^3\,y_a \\
&\quad +6\,x_b^2\,y_b\,y_a - 4\,x_b\,x_c\,y_b\,y_a - 6\,y_c^3\,y_a + 4\,x_c^2\,y_b\,y_a + 6\,y_c\,x_c^2\,y_a \\
&\quad -4\,x_b\,y_c\,x_c\,y_a + 4\,x_b^2\,y_c\,y_a - 6\,x_c^2\,y_c^2 + 6\,y_b^2\,x_c\,x_b + 3\,x_b^4 \\
&\quad +3\,y_b^4 + 6\,y_c\,x_c^2\,y_b - 6\,y_b^2\,x_b^2 - 5\,y_b^2\,x_c^2 + 3\,x_c^4 - 6\,y_c\,y_b^3 \\
&\quad +9\,y_c^2\,y_b^2 + 3\,y_c^4 - 6\,y_c^3\,y_b - 6\,x_b\,x_c^3 + 9\,x_c^2\,x_b^2 - 5\,y_c^2\,x_b^2 \\
&\quad -6\,x_c\,x_b^3 + 6\,x_b\,y_c^2\,x_c + 6\,y_c\,x_b^2\,y_b - 8\,x_b\,y_c\,x_c\,y_b\,)
\end{aligned}
$$

Le premier terme de l'expression obtenue reste évidemment invariant par translation (et même par isométrie) car on reconnait, au signe et à un facteur 2 près, la surface du triangle Δabc. Le deuxième terme étant peu exploitable, on est obligé de faire confiance au logiciel de calcul formel. □

3.2.3 Calcul de l'erreur sur deux triangles adjacents

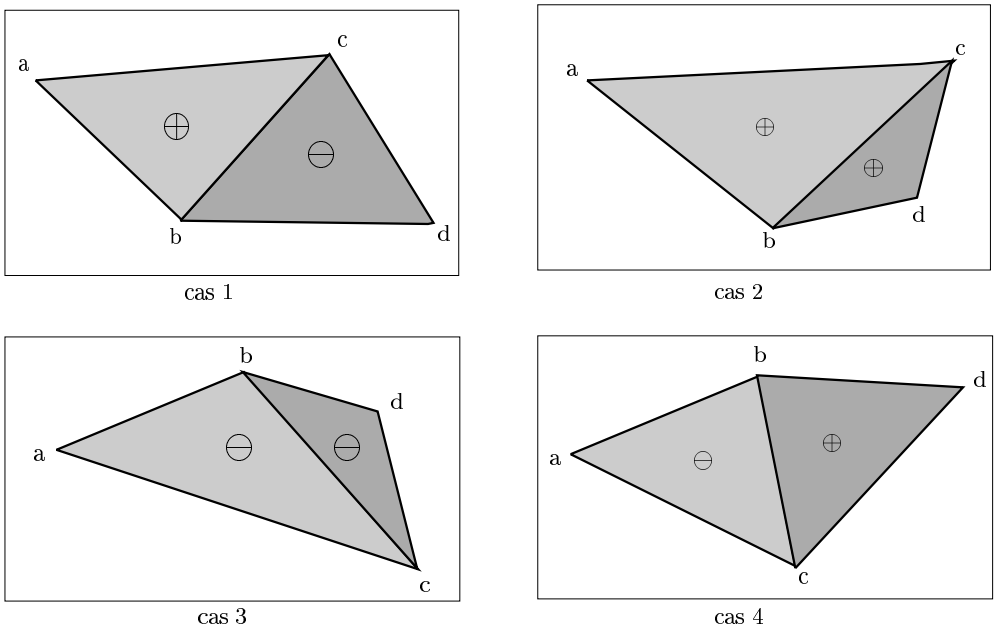
Pour trouver l'équation de la courbe de séparation, on a besoin de calculer l'erreur commise par deux triangles qui ont une arête commune.

Soient donc, par exemple, les points a, b, c et d du plan. On suppose (sans perte de généralité) que les triangles Δabc et Δbcd sont adjacents en position convenable (le quadrilatère $\square abcd$ est convexe), et que les points a, b, c sont rangés par ordre d'abscisses croissantes. On suppose enfin que d se trouve dans la zone d'influence

de a : ceci implique que l'arête commune aux deux triangles est $[bc]$. Les cas où d se trouve dans une autre zone d'influence se traitent de manière similaire.

On a vu que le signe des intégrales calculées lors de la recherche de l'erreur pour un seul triangle dépend des positions relatives de ses sommets ; quatre cas peuvent alors se produire lorsque l'on cherche l'erreur commise par la triangulation des quatre points, cas illustrés sur la figure 3.9.

Figure 3.9 – les quatre cas associés à l'étude proposée.



Pour le triangle Δabc, on sait, par hypothèse, que le côté qui contient toutes les valeurs en x est $[ac]$. Deux possibilités apparaissent :

- l'ordonnée de c est supérieure à celle de b : alors, l'arête $[ac]$ est au-dessus des deux autres, et la procédure de calcul de $\epsilon_{Maple}(\Delta abc)$ renvoie bien ce qu'il faut (cas du haut sur la figure).
- l'ordonnée de c est inférieure à celle de b : il faudra adjoindre un signe négatif au résultat de $\epsilon_{Maple}(\Delta abc)$ (cas du bas).

Pour le triangle Δbcd, le problème est sensiblement le même : l'abscisse de d est soit comprise entre celle de b et celle de c, soit supérieure à celle de c ;

- dans le premier cas, le résultat de $\epsilon_{Maple}(\Delta bcd)$ sera bien celui recherché, puisque le côté contenant toutes les abscisses possibles est $[bc]$, et il se trouve au-dessus de $[bd]$ et de $[cd]$ (cas de droite sur la figure).
- dans le deuxième cas, c'est $[bd]$ qui contient toutes les abscisses de Δbcd et il est placé au-dessous de $[bc]$ et $[cd]$: il faut changer le signe de $\epsilon_{Maple}(\Delta bcd)$.

Suivant les cas, dans l'hypothèse du calcul d'une erreur formelle, il faudra donc soit ajouter, soit soustraire les valeurs obtenues pour chaque triangle de la triangulation.

3.2.4 Justification des équations des courbes de séparation

Comme on vient de le voir, le signe des intégrales varie suivant les positions des points considérés. Cela laisserait supposer que l'on doive connaitre à l'avance ces positions relatives. Ainsi, on se placera d'abord dans une configuration particulière où l'on choisira les positions relatives des points. Tous les détails concernant cette configuration seront donnés, et seront suivis d'une étude de ce qui survient pour les autres cas ; on s'assurera alors que le résultat final est bien indépendant de la position des sommets (et concorde avec les recherches empiriques menées...).

3.2.4.1 Le cas choisi

Soient trois points $o = (0, 0)$ (l'invariance de l'erreur par translation justifiée à la proposition 3.1 autorise ce choix), $b = (x_b, y_b)$ et $c = (x_c, y_c)$ fixés dans le plan, d'abscisses respectives données dans l'ordre croissant $x_o \leqslant x_b \leqslant x_c$, et tels que $y_b \leqslant y_c$.

Soient $O = (0, 0, 0)$, $B = (x_b, y_b, z_B)$ et $C = (x_c, y_c, z_C)$ leurs relevés respectifs sur le paraboloïde hyperbolique, et soit $d = (x, y)$ un point variable dans le plan. On va déterminer pour quelles valeurs de x et y, d se trouve sur la courbe de séparation du triangle Δobc.

On se place dans le cas où d est dans la zone d'influence du sommet o. Deux triangulations sont alors possibles, puisque le quadrilatère $\square obcd$ ainsi défini est convexe : la triangulation qui contient les triangles Δobc et Δbcd, et celle qui contient les triangles Δobd et Δocd.

Leurs erreurs respectives sont données par les calculs suivants :

$$\begin{array}{rcl} e_2(\Delta obc + \Delta bcd) & = & \sqrt{\epsilon(\Delta obc) + \epsilon(\Delta bcd)} \\ e_2(\Delta obd + \Delta ocd) & = & \sqrt{\epsilon(\Delta obd) + \epsilon(\Delta ocd)} \end{array}$$

La valeur qui nous intéresse plus particulièrement est alors :

$$\xi(\Delta obc, d) = [\epsilon(\Delta obc) + \epsilon(\Delta bcd)] - [\epsilon(\Delta obd) + \epsilon(\Delta ocd)]\,.$$

En effet, par définition, le point d se trouvera sur la courbe de séparation du triangle Δobc si et seulement si $\xi(\Delta obc, d)$ vaut 0, puisqu'à ce moment-là, les 2 triangulations possibles donneront les mêmes erreurs d'approximation L_2.

On suppose maintenant que l'abscisse x de d est supérieure à celle de c (on se retrouve dans le cas 1 de la figure 3.9) ; calculer ξ symboliquement implique alors d'effectuer les opérations suivantes :

$$\left\{ \begin{array}{rcl} \epsilon(\Delta obc) & = & \epsilon_{Maple}(\Delta obc) \\ \epsilon(\Delta bcd) & = & -\epsilon_{Maple}(\Delta bcd) \\ \epsilon(\Delta obd) & = & \epsilon_{Maple}(\Delta obd) \\ \epsilon(\Delta ocd) & = & -\epsilon_{Maple}(\Delta ocd) \end{array} \right.$$

Les signes proviennent des remarques faites précédemment pour la première triangulation, et pour la seconde, l'arête qui contient toutes les abscisses est $[od]$, arête *haute* du triangle Δobd (d'où le signe $+$) et arête *basse* de Δocd (signe $-$).

Le résultat final est une équation en x et y que l'on peut comparer avec les hyperboles obtenues expérimentalement. Sans autre connaissance, il n'est a priori pas possible de chercher manuellement à évaluer les erreurs ; les justifications qui vont suivre font donc appel à la procédure de calcul symbolique programmée en *Maple*.

3.2.4.2 Les résultats obtenus sous *Maple*

Les résultats fournis par *Maple* seront sans aucun doute exacts, puisque rechercher l'erreur formelle revient à intégrer 2 fois un polynôme de degré 4, chose que le langage est tout à fait en mesure de réaliser. Il faut cependant remarquer que pour arriver au bout de certains de ces calculs, il vaut mieux connaître le résultat à l'avance afin de pouvoir suggérer des simplifications au logiciel.

Pour donner une idée de la relative complexité du calcul, il faut savoir que le calcul de l'erreur commise pour le triangle Δbcd (qui ne comporte aucune coordonnée chiffrée contrairement à tous les triangles incluant le point o) a nécessité sur une machine raisonnable, du type SUN 4/75 à 32 Mo de RAM, près de 25 minutes de temps CPU avec une allocation mémoire de l'ordre de 20 Méga Octets.

Afin de suivre complètement la procédure mise en place pour justifier le résultat observé dans la section 3.1, il va être ici reproduit directement les résultats obtenus avec *Maple*. Dans le programme, la procédure `ErreurL2 (f, a, b, c)` correspond à $\epsilon_{Maple}(\Delta abc)$, valeur exacte (symbolique) de l'erreur commise par l'approximation linéaire $p_{\Delta ABC}(x,y)$ de la surface `f`, surface d'équation $z = f(x,y)$, pour le triangle Δabc. Autrement dit :

$$|\texttt{ErreurL2 (f, a, b, c)}| = \iint_{\Delta abc} (p_{\Delta abc}(x,y) - f(x,y))^2 \, dx \, dy.$$

Pour notre étude, `f` est la fonction qui à (x,y) associe $x^2 - y^2$. Les sommets sont donnés sous forme de couples de coordonnées (par exemple $[x_a, y_a]$) et dans l'ordre croissant de leur abscisse. On rappelle par ailleurs que la procédure suppose que $y_c \geqslant y_b$, le cas contraire imposant un changement de signe.

Le cas choisi ici impose ainsi les choix :

$$\begin{cases} \epsilon(\Delta obc) = \texttt{err11} = \texttt{ErreurL2 (f, [0,0], [x\_b,y\_b], [x\_c,y\_c])} \\ \epsilon(\Delta bcd) = \texttt{err12} = -\texttt{ErreurL2 (f, [x\_b,y\_b], [x\_c,y\_c], [x,y])} \\ \epsilon(\Delta obd) = \texttt{err21} = \texttt{ErreurL2 (f, [0,0], [x\_b,y\_b], [x,y])} \\ \epsilon(\Delta ocd) = \texttt{err22} = -\texttt{ErreurL2 (f, [0,0], [x\_c,y\_c], [x,y])} \end{cases}$$

On obtient alors sous *Maple* :

```
> f := (x,y) -> x^2 - y^2;
```

$$f := (x,y) \to x^2 - y^2$$

```
> err11 := erreurL2 (f, [0,0], [x_b,y_b], [x_c,y_c]);
```

$$-\frac{1}{180}(\,y_b\,x_c - x_b\,y_c\,)(3\,{x_c}^4 - 6\,{x_c}^3\,x_b - 5\,{x_c}^2\,{y_b}^2 + 6\,{x_c}^2\,y_c\,y_b + 9\,{x_b}^2\,{x_c}^2 - 6\,{x_c}^2\,{y_c}^2 + 6\,x_c\,x_b\,{y_b}^2 - 8\,x_b\,x_c\,y_c\,y_b - 6\,{x_b}^3\,x_c + 6\,x_c\,x_b\,{y_c}^2 + 3\,{y_b}^4 - 6\,y_c\,{y_b}^3 + 9\,{y_b}^2\,{y_c}^2 - 6\,{y_b}^2\,{x_b}^2 - 6\,{y_c}^3\,y_b + 6\,{x_b}^2\,y_c\,y_b + 3\,{x_b}^4 + 3\,{y_c}^4 - 5\,{x_b}^2\,{y_c}^2)$$

```
> err12 := - erreurL2 (f, [x_b,y_b], [x_c,y_c], [x,y]);
```

$$\frac{1}{180}(\,y_c\,x - y_b\,x - x_c\,y + x_b\,y - x_b\,y_c + y_b\,x_c\,)(3\,x^4 - 6\,x^2\,y^2 + 3\,y^4 - 6\,x_c\,x^3 - 6\,x_b\,x^3 + 6\,y_c\,x^2\,y + 6\,y_b\,x^2\,y + 6\,x_b\,x\,y^2 + 6\,x_c\,x\,y^2 - 6\,y_b\,y^3 - 6\,y_c\,y^3 + 9\,{x_b}^2\,x^2 + 9\,{x_c}^2\,x^2 - 5\,{y_b}^2\,x^2 + 4\,y_b\,y_c\,x^2 - 5\,{y_c}^2\,x^2 - 8\,y_c\,x_c\,x\,y - 4\,y_c\,x_b\,x\,y - 8\,x_b\,y_b\,x\,y - 4\,y_b\,x_c\,x\,y - 5\,{x_c}^2\,y^2 + 9\,{y_c}^2\,y^2 + 4\,x_c\,x_b\,y^2 - 5\,{x_b}^2\,y^2 + 9\,{y_b}^2\,y^2 + 6\,x_c\,{y_c}^2\,x + 4\,{y_c}^2\,x_b\,x - 4\,x_c\,y_b\,y_c\,x + 4\,x_c\,{y_b}^2\,x - 6\,{x_b}^3\,x + 6\,{y_b}^2\,x_b\,x - 6\,{x_c}^3\,x - 4\,x_b\,y_b\,y_c\,x - 6\,{y_c}^3\,y - 4\,x_b\,y_c\,x_c\,y + 4\,y_c\,{x_b}^2\,y - 4\,x_b\,y_b\,x_c\,y + 6\,y_c\,{x_c}^2\,y + 6\,y_b\,{x_b}^2\,y + 4\,y_b\,{x_c}^2\,y - 6\,{y_b}^3\,y + 3\,{y_b}^4 + 3\,{x_b}^4 - 5\,{x_b}^2\,{y_c}^2 + 3\,{y_c}^4 + 6\,{x_c}^2\,y_c\,y_b - 6\,{x_c}^2\,{y_c}^2 + 3\,{x_c}^4 - 6\,{y_b}^2\,{x_b}^2 - 5\,{x_c}^2\,{y_b}^2 + 9\,{x_b}^2\,{x_c}^2 - 6\,{x_b}^3\,x_c - 6\,{x_c}^3\,x_b - 6\,y_c\,{y_b}^3 + 6\,{x_b}^2\,y_c\,y_b + 6\,x_c\,x_b\,{y_b}^2 - 6\,{y_c}^3\,y_b - 8\,x_b\,x_c\,y_c\,y_b + 6\,x_c\,x_b\,{y_c}^2 + 9\,{y_b}^2\,{y_c}^2)$$

```
> err21 := erreurL2 (f, [0,0], [x_b,y_b], [x,y]);
```

$$\frac{1}{180}(-y_b\,x + x_b\,y\,)(3\,x^4 - 6\,x^2\,y^2 + 3\,y^4 - 6\,x_b\,x^3 + 6\,y_b\,x^2\,y + 6\,x_b\,x\,y^2 - 6\,y_b\,y^3 + 9\,{x_b}^2\,x^2 - 5\,{y_b}^2\,x^2 - 8\,x_b\,y_b\,x\,y - 5\,{x_b}^2\,y^2 + 9\,{y_b}^2\,y^2 - 6\,{x_b}^3\,x + 6\,{y_b}^2\,x_b\,x + 6\,y_b\,{x_b}^2\,y - 6\,{y_b}^3\,y + 3\,{x_b}^4 + 3\,{y_b}^4 - 6\,{y_b}^2\,{x_b}^2)$$

```
> err22 := - erreurL2 (f, [0,0], [x_c,y_c], [x,y]);
```

$$\frac{1}{180}(\,y_c\,x - x_c\,y\,)(3\,x^4 - 6\,x^2\,y^2 + 3\,y^4 - 6\,x_c\,x^3 + 6\,y_c\,x^2\,y + 6\,x_c\,x\,y^2 - 6\,y_c\,y^3 + 9\,{x_c}^2\,x^2 - 5\,{y_c}^2\,x^2 - 8\,y_c\,x_c\,x\,y - 5\,{x_c}^2\,y^2 + 9\,{y_c}^2\,y^2 - 6\,{x_c}^3\,x + 6\,x_c\,{y_c}^2\,x + 6\,y_c\,{x_c}^2\,y - 6\,{y_c}^3\,y + 3\,{x_c}^4 + 3\,{y_c}^4 - 6\,{x_c}^2\,{y_c}^2)$$

```
> erreur1 := err11 + err12:
> erreur2 := err21 + err22:
```

Avec les notations *Maple* adoptées, on aura :

$$\xi(\Delta obc, d) = \texttt{erreur} = \texttt{erreur2} - \texttt{erreur1}.$$

Pour aider *Maple* à obtenir une valeur lisible de ce polynôme, il faut le forcer à effectuer un tri suivant les puissances en x et en y (par l'utilisation de la fonction prédéfinie `sort`), puis en lui demandant de factoriser l'expression ainsi triée (par la fonction `factor`).

C'est à ces conditions seulement que l'on peut obtenir un résultat lisible pour $\xi(\Delta obc, d)$.

```
> erreur  := sort ((erreur2-erreur1), [x,y]):
> erreur  := factor (erreur, [x,y,x_b,y_b,x_c,y_c]);
```

$$\begin{aligned} \mathit{erreur} := \frac{1}{60} & (3\,x^2 - 3\,y^2 - 2\,x_c\,x - 2\,x_b\,x + 2\,y_c\,y + 2\,y_b\,y + 3\,{x_c}^2 \\ & - 3\,{y_c}^2 - 3\,{y_b}^2 - 2\,x_c\,x_b + 2\,y_c\,y_b + 3\,{x_b}^2) \\ & (y_b\,x_c\,x^2 - y_c\,x_b\,x^2 - y_b\,x_c\,y^2 + y_c\,x_b\,y^2 + y_b\,{y_c}^2\,x \\ & - y_c\,{y_b}^2\,x + y_c\,{x_b}^2\,x - y_b\,{x_c}^2\,x + x_c\,{y_b}^2\,y + {x_c}^2\,x_b\,y \\ & - x_c\,{x_b}^2\,y - {y_c}^2\,x_b\,y) \end{aligned}$$

Il reste maintenant à comparer cette expression avec les hyperboles décrites dans l'observation 3.7. Soient G et H les points définis dans l'onservation. Le plan $\mathcal{P}_{\Delta OBC}$ passant par les sommets O, B et C est renommé `approx` lors de son utilisation avec *Maple*, et on adopte les notations suivantes :

- `hyperb1` correspond à l'équation observée de la première hyperbole de la courbe de séparation de Δobc, équation donnée par l'intersection du paraboloïde hyperbolique avec le plan $\mathcal{P}_{\Delta OBC}$.

- `hyperb2` représente l'équation de la seconde hyperbole :

$$\texttt{hyperb2} = (x - x_G)^2 - (y - y_G)^2 - 4(z_G - z_H).$$

La courbe de séparation obtenue dans l'observation 3.7 est ainsi donnée par :

$$\mathcal{CS}(\Delta obc) = (\texttt{hyperb1})(\texttt{hyperb2}).$$

`DetermineApproximationLineaire (f, a, b, c)` est une procédure qui a pour argument une fonction $f(x, y)$, et 3 points du plan, et qui renvoie, sous forme de liste $[\alpha, \beta, \gamma]$, les coefficients de l'équation du plan $z = \alpha x + \beta y + \gamma$ passant par les relevés de a, b et c sur la surface $z = f(x, y)$ (on suppose toujours ici que $f(x, y) = x^2 - y^2$). Par ailleurs, `expand` et `simplify` sont 2 fonctions prédéfinies de *Maple* qui servent respectivement à développer une expression et à la simplifier.

On aura ainsi :

```
> G := [(x_b+x_c)/3, (y_b+y_c)/3, ((x_b+x_c)^2-(y_b+y_c)^2)/9]:
> H := [(x_b+x_c)/3, (y_b+y_c)/3, (x_b^2-y_b^2+x_c^2-y_c^2)/3]:
```

```
> approx := DetermineApproximationLineaire
>                                   (f, [0,0], [x_b,y_b], [x_c,y_c]);
```

$$approx := \left[\frac{x_b\,(\,{x_c}^2-{y_c}^2\,)-x_c\,(\,{x_b}^2-{y_b}^2\,)}{x_b\,y_c-x_c\,y_b},\right.$$
$$\left.\frac{y_b\,(\,-{x_c}^2+{y_c}^2\,)+y_c\,(\,{x_b}^2-{y_b}^2\,)}{x_b\,y_c-x_c\,y_b},0\right]$$

```
> approx := [simplify(expand(approx[1])),
>                               simplify(expand(approx[2])),0];
```

$$approx := \left[-\frac{-x_b\,{x_c}^2+x_b\,{y_c}^2+x_c\,{x_b}^2-x_c\,{y_b}^2}{x_b\,y_c-x_c\,y_b},\right.$$
$$\left.\frac{-y_b\,{x_c}^2+y_b\,{y_c}^2+y_c\,{x_b}^2-y_c\,{y_b}^2}{x_b\,y_c-x_c\,y_b},0\right]$$

```
> hyperb1 := x^2 - y^2 - approx[1]*y - approx[2]*x;
```

$$hyperb1 := x^2-y^2+\frac{(\,-x_b\,{x_c}^2+x_b\,{y_c}^2+x_c\,{x_b}^2-x_c\,{y_b}^2\,)\,y}{x_b\,y_c-x_c\,y_b}$$
$$-\frac{(\,-y_b\,{x_c}^2+y_b\,{y_c}^2+y_c\,{x_b}^2-y_c\,{y_b}^2\,)\,x}{x_b\,y_c-x_c\,y_b}$$

```
> hyperb2 := (x-G[1])^2 - (y-G[2])^2 - 4*(G[3]-H[3]);
```

$$hyperb2 := \left(x-\frac{1}{3}\,x_b-\frac{1}{3}\,x_c\right)^2-\left(y-\frac{1}{3}\,y_b-\frac{1}{3}\,y_c\right)^2-\frac{4}{9}\,(\,x_b+x_c\,)^2$$
$$+\frac{4}{9}\,(\,y_b+y_c\,)^2+\frac{4}{3}\,{x_b}^2-\frac{4}{3}\,{y_b}^2+\frac{4}{3}\,{x_c}^2-\frac{4}{3}\,{y_c}^2$$

```
> CS := factor (simplify (expand (hyperb1*hyperb2)),
>                                 [x,y,x_b,y_b,x_c,y_c]);
```

$$CS := \frac{1}{3\,(\,x_b\,y_c-x_c\,y_b\,)}(\,{x_b}^2\,y\,x_c-x\,y_c\,{x_b}^2+x^2\,x_b\,y_c-x\,y_b\,{y_c}^2+x\,{y_b}^2\,y_c$$
$$+y\,x_b\,{y_c}^2-y^2\,x_b\,y_c-x_c\,x^2\,y_b-x_c\,y\,{y_b}^2+x_c\,y^2\,y_b-{x_c}^2\,y\,x_b+{x_c}^2\,x\,y_b)$$
$$(3\,{x_b}^2-3\,{y_c}^2-2\,x_b\,x+3\,x^2+2\,y_c\,y_b-3\,{y_b}^2+2\,y\,y_c+2\,y\,y_b-3\,y^2$$
$$-2\,x_c\,x_b-2\,x_c\,x+3\,{x_c}^2)$$

Pour conclure la preuve, il reste à comparer les valeurs respectives de `erreur` et de $\mathcal{CS}$. On trouve :

$$\frac{\mathcal{CS}}{\texttt{erreur}}=\frac{20}{x_b\,y_c-x_c\,y_b},$$

d'où le résultat annoncé à une constante près ; les calculs des erreurs sous *Maple* justifient les équations pour la courbe de séparation d'un triangle de l'observation 3.7, tout au moins dans le cas où d se trouve dans la zone d'influence de o avec $y_b \leqslant y_c$ et $x \geqslant x_c$. En effet, le fait que les constantes soient différentes n'intervient

pas dans les cas d'égalité à 0 qui donnent les points de la courbe de séparation.

3.2.4.3 Les autres cas

Soient a, b, c et d 4 points du plan. On note $g_{\Delta abc}$, $g_{\Delta bcd}$, $g_{\Delta abd}$ et $g_{\Delta acd}$ les fonctions qui ont pour valeur le carré de la différence entre le point sur le paraboloïde hyperbolique et le point correspondant sur le plan approximation linéaire locale :

$$\begin{aligned} g_{\Delta abc}(x,y) &= (p_{\Delta ABC}(x,y) - (x^2 - y^2))^2 \\ g_{\Delta bcd}(x,y) &= (p_{\Delta BCD}(x,y) - (x^2 - y^2))^2 \\ g_{\Delta abd}(x,y) &= (p_{\Delta ABD}(x,y) - (x^2 - y^2))^2 \\ g_{\Delta acd}(x,y) &= (p_{\Delta ACD}(x,y) - (x^2 - y^2))^2 \end{aligned}$$

Pour un triangle Δabc de sommet intermédiaire b, on notera $\epsilon^+(\Delta abc)$ la formule obtenue pour l'erreur L_2 d'interpolation qu'il définit pour le paraboloïde hyperbolique lorsque b est au-dessous de l'arête $[ac]$, et $\epsilon^-(\Delta abc)$ lorsque b est au-dessus (c'est-à-dire que $\epsilon^+ = \epsilon_{Maple}$ et $\epsilon^- = -\epsilon_{Maple}$). La double intégrale représentant une partie de l'erreur sera écrite, par exemple, sous la forme

$$\int_{x_a}^{x_b} \int_{[ab]}^{[ac]} g_{\Delta abc},$$

où la première intégrale sera prise pour des valeurs de x variant entre x_a et x_b, et la deuxième intégrale pour des valeurs de y entre $\alpha_{(ab)}x + \beta_{(ab)}$ et $\alpha_{(ac)}x + \beta_{(ac)}$.

Suivant les positions relatives des sommets, les erreurs seront données par des formules différentes ; par exemple, si on suppose que $x_a \leqslant x_b \leqslant x_c$:

– si $y_c > y_b$:

$$\epsilon^+(\Delta abc) = \int_{x_a}^{x_b} \int_{[ab]}^{[ac]} g_{\Delta abc} + \int_{x_b}^{x_c} \int_{[bc]}^{[ac]} g_{\Delta abc}$$

– si $y_c < y_b$:

$$\epsilon^-(\Delta abc) = \int_{x_a}^{x_b} \int_{[ac]}^{[ab]} g_{\Delta abc} + \int_{x_b}^{x_c} \int_{[ac]}^{[bc]} g_{\Delta abc}$$

– si $x_d > x_c$ et $y_c < y_b$:

$$\begin{cases} \epsilon^+(\Delta bcd) &= \displaystyle\int_{x_b}^{x_c} \int_{[bd]}^{[bc]} g_{\Delta bcd} + \int_{x_c}^{x_d} \int_{[bd]}^{[cd]} g_{\Delta bcd} \\ \epsilon^-(\Delta abd) &= \displaystyle\int_{x_a}^{x_b} \int_{[ab]}^{[ad]} g_{\Delta abd} + \int_{x_b}^{x_d} \int_{[bd]}^{[ad]} g_{\Delta abd} \\ \epsilon^+(\Delta acd) &= \displaystyle\int_{x_a}^{x_c} \int_{[ad]}^{[ac]} g_{\Delta abd} + \int_{x_c}^{x_d} \int_{[cd]}^{[cd]} g_{\Delta abd} \end{cases}$$

– si $x_b < x_d < x_c$ et $y_c > y_b$:

$$\begin{cases} \epsilon^+(\Delta bdc) &= \displaystyle\int_{x_b}^{x_d}\int_{[bd]}^{[bc]} g_{\Delta bcd} + \int_{x_d}^{x_c}\int_{[cd]}^{[bc]} g_{\Delta bcd} \\ \epsilon^+(\Delta abd) &= \displaystyle\int_{x_a}^{x_b}\int_{[ab]}^{[ad]} g_{\Delta abd} + \int_{x_b}^{x_d}\int_{[bd]}^{[ad]} g_{\Delta abd} \\ \epsilon^+(\Delta adc) &= \displaystyle\int_{x_a}^{x_d}\int_{[ad]}^{[ac]} g_{\Delta abd} + \int_{x_d}^{x_c}\int_{[cd]}^{[ac]} g_{\Delta abd} \end{cases}$$

Grâce à ces formules, on peut établir trois lemmes :

lemme 3.1 $\epsilon^-(\Delta) = -\epsilon^+(\Delta)$.

démonstration :

Soient a, b et c les 3 sommets de Δ donnés dans l'ordre croissant de leurs abscisses ; alors :

$$\begin{aligned} \epsilon^-(\Delta abc) &= \int_{x_a}^{x_b}\int_{[ac]}^{[ab]} g_{\Delta abc} + \int_{x_b}^{x_c}\int_{[bc]}^{[ab]} g_{\Delta abc} \\ &= -\int_{x_a}^{x_b}\int_{[ab]}^{[ac]} g_{\Delta abc} - \int_{x_b}^{x_c}\int_{[bc]}^{[ac]} g_{\Delta abc} \\ &= -\epsilon^+(\Delta abc) \end{aligned}$$

□

lemme 3.2 $\epsilon^\pm(\Delta)$ *est invariant par permutation circulaire des sommets du triangle* Δ.

démonstration :

Soient a, b et c les 3 sommets de Δ donnés dans l'ordre croissant de leurs abscisses. On va supposer que b est au-dessous du segment $[ac]$ et travailler ainsi sur ϵ^+ ; la preuve reste la même dans le cas contraire.
On a :

$$\epsilon^+(\Delta abc) = \int_{x_a}^{x_b}\int_{[ab]}^{[ac]} g_{\Delta abc} + \int_{x_b}^{x_c}\int_{[bc]}^{[ac]} g_{\Delta abc}.$$

Par permutation circulaire des points, on obtient :

$$\epsilon^+(\Delta cab) = \int_{x_c}^{x_a}\int_{[ac]}^{[bc]} g_{\Delta cab} + \int_{x_a}^{x_b}\int_{[ab]}^{[bc]} g_{\Delta cab},$$

avec $g_{\Delta abc} = g_{\Delta cab}$, puisque $\mathcal{P}_{\Delta ABC} = \mathcal{P}_{\Delta CAB}$.

D'où :

$$\begin{aligned}\epsilon^+(\Delta cab) &= \int_{x_c}^{x_b}\int_{[ac]}^{[bc]} g_{\Delta abc} + \int_{x_b}^{x_a}\int_{[ac]}^{[bc]} g_{\Delta abc} \\ &\quad + \int_{x_a}^{x_b}\int_{[ab]}^{[ac]} g_{\Delta abc} + \int_{x_a}^{x_b}\int_{[ac]}^{[bc]} g_{\Delta abc} \\ &= \int_{x_a}^{x_b}\int_{[ab]}^{[ac]} g_{\Delta abc} + \int_{x_b}^{x_c}\int_{[bc]}^{[ac]} g_{\Delta abc} \\ &= \epsilon^+(\Delta abc)\end{aligned}$$

□

lemme 3.3 $\epsilon^{\pm}(\Delta)$ *se transforme en son opposé par transposition de deux de ses sommets.*

démonstration :

Soient a, b et c les 3 sommets de Δ donnés dans l'ordre croissant de leurs abscisses. On va supposer que b est au-dessous du segment $[ac]$ et travailler ainsi sur ϵ^+ ; la preuve reste la même dans le cas contraire.
On a :

$$\epsilon^+(\Delta abc) = \int_{x_a}^{x_b}\int_{[ab]}^{[ac]} g_{\Delta abc} + \int_{x_b}^{x_c}\int_{[bc]}^{[ac]} g_{\Delta abc}.$$

Par transposition des 2 sommets extrêmes, on a :

$$\begin{aligned}\epsilon^+(\Delta cba) &= \int_{x_c}^{x_b}\int_{[bc]}^{[ac]} g_{\Delta cba} + \int_{x_b}^{x_a}\int_{[ab]}^{[ac]} g_{\Delta cba} \\ &= -\int_{x_a}^{x_b}\int_{[ab]}^{[ac]} g_{\Delta cba} - \int_{x_b}^{x_c}\int_{[bc]}^{[ac]} g_{\Delta cba} \\ &= -\epsilon^+(\Delta abc) = \epsilon^-(\Delta abc)\end{aligned}$$

car $g_{\Delta cba} = g_{\Delta abc}$ et d'après le lemme 3.1.

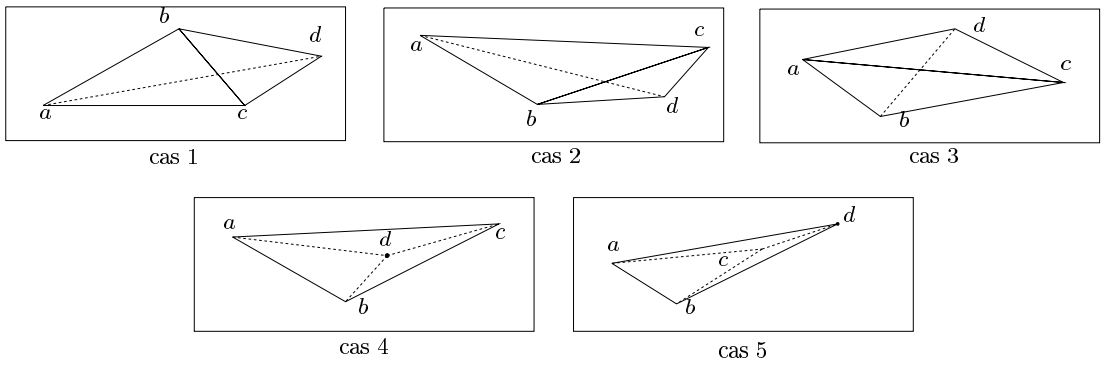
Par transposition du sommet intermédiaire et d'un des sommets extrêmes, par exemple c, on a :

$$\begin{aligned}\epsilon^+(\Delta acb) &= \epsilon^-(\Delta bca), \text{par transposition des extrêmes} \\ &= \epsilon^-(\Delta abc), \text{par permutation circulaire}\end{aligned}$$

car $g_{\Delta acb} = g_{\Delta abc}$ et d'après le lemme 3.1. □

Pour montrer que la formule obtenue dans le cas de la section 3.2.4.1 reste pratiquement la même si les positions des points changent, on ne va étudier ici que les configurations caractéristiques (car il existe beaucoup trop de cas différents pour tous les traiter) de la figure 3.10.

Figure 3.10 – les configurations caractéristiques choisies.



On note $\xi_1 = \xi(\Delta abc, d)$ la formule obtenue pour l'expression de la courbe en fonction des $\epsilon^{\pm}$, pour le cas de la section 3.2.4.1. On suppose que $d = (x, y)$ est le point variable, et que a, b et c sont en ordre croissant d'abscisses. On rappelle que :

$$\xi_1 = \epsilon^+(\Delta abc) + \epsilon^-(\Delta bcd) - \epsilon^+(\Delta abd) - \epsilon^-(\Delta acd)$$

Alors, à l'aide des 3 lemmes précédents, on obtient :

– si b est au-dessus de $[ac]$ et $x > x_c$ (cas 1) :

$$\begin{aligned}\xi_2(\Delta abc, d) &= \epsilon^-(\Delta abc) + \epsilon^+(\Delta bcd) - \epsilon^-(\Delta abd) - \epsilon^+(\Delta acd)\\ &= -\epsilon^+(\Delta abc) - \epsilon^-(\Delta bcd) + \epsilon^+(\Delta abd) + \epsilon^-(\Delta acd)\\ &= -\xi_1\end{aligned}$$

– si b reste au-dessous de $[ac]$ mais $x_b < x < x_d$ (cas 2) :

$$\begin{aligned}\xi_2(\Delta abc, d) &= \epsilon^+(\Delta abc) + \epsilon^+(\Delta bdc) - \epsilon^+(\Delta abd) - \epsilon^+(\Delta adc)\\ &= \epsilon^+(\Delta abc) - \epsilon^+(\Delta bcd) - \epsilon^+(\Delta abd) + \epsilon^+(\Delta acd)\\ &= \epsilon^+(\Delta abc) + \epsilon^-(\Delta bcd) - \epsilon^+(\Delta abd) - \epsilon^-(\Delta acd)\\ &= \xi_1\end{aligned}$$

– si d est dans la zone d'influence de b avec $y > y_a$ et $y > y_c$ (cas 3) :

$$\begin{aligned}\xi_2(\Delta abc, d) &= \epsilon^+(\Delta abc) + \epsilon^-(\Delta adc) - \epsilon^+(\Delta abd) - \epsilon^-(\Delta bdc)\\ &= \epsilon^+(\Delta abc) + \epsilon^-(\Delta bcd) - \epsilon^+(\Delta abd) - \epsilon^-(\Delta acd)\\ &= \xi_1\end{aligned}$$

Les 2 cas caractéristiques qui restent (un point à l'intérieur du triangle défini par les 3 autres) impliquent que le choix devra se faire entre une triangulation à 1 seul triangle, et une triangulation à 3 triangles. Cela laisse supposer que l'on s'autorise à éliminer un sommet, ce qui peut ne pas être désirable (puisque le critère ne permet qu'une optimisation locale) ; cependant, on peut vérifier que le résultat pour les

courbes de séparation (et ξ) ne change pas :

– si d est à l'intérieur de Δabc et $x_b < x < x_c$ (cas 4) :

$$\begin{aligned}\xi_2(\Delta abc, d) &= \epsilon^+(\Delta abc) - \epsilon^-(\Delta bdc) - \epsilon^+(\Delta abd) - \epsilon^+(\Delta adc)\\ &= \epsilon^+(\Delta abc) + \epsilon^-(\Delta bcd) - \epsilon^+(\Delta abd) - \epsilon^-(\Delta acd)\\ &= \xi_1\end{aligned}$$

– si b n'est ni à l'intérieur de Δabc, ni dans une zone d'influence, avec $x > x_c$ et $y > y_c$ (cas 5) :

$$\begin{aligned}\xi_2(\Delta abc, d) &= \epsilon^+(\Delta abd) - \epsilon^+(\Delta abc) - \epsilon^+(\Delta acd) - \epsilon^-(\Delta bcd)\\ &= -\epsilon^+(\Delta abc) - \epsilon^-(\Delta bcd) + \epsilon^+(\Delta abd) + \epsilon^-(\Delta acd)\\ &= -\xi_1\end{aligned}$$

Dans tous les cas, on retombe sur $\pm\xi_1$; or l'équation de la courbe de séparation est $\xi_1 = 0$, ce qui implique qu'elle est bien invariante par changement de configuration.

3.3 Exemples de courbes obtenues

Sont regroupés ici quelques exemples de courbes de séparation. À chaque fois, un triangle initial est choisi, et le plan est découpé en zones ; les conventions pour les couleurs utilisées font intervenir le role que jouerait l'insertion d'un quatrième à l'intérieur d'une zone :

- les zones foncées sont celles où une seule triangulation est possible.
- les zones colorées claires sont celles où lorsque deux triangulations sont possibles, c'est celle qui conserve le triangle de départ qui est la meilleure.
- les zones blanches impliquent que la meilleure triangulation ne tiendra pas compte du triangle initial.

Les courbes de séparation ainsi que les axes sont en trait fin, alors que le triangle de départ dont on cherche la courbe de séparation est en trait épais.

Les cas généraux

On obtient des hyperboles équilatères à directrices parallèles (figure 3.11) ou perpendiculaires (figure 3.12), éventuellement dégénérées.

Des cas dégénérés

Il peut arriver qu'une des hyperboles soit dégénérée en une droite (comme c'était le cas sur l'exemple de la figure 3.5), et que même les deux le soit. Ces cas se produisent notamment lorsque la droite passant par deux des sommets est parallèle à une des bissectrices des axes du repère (figure 3.13, courbe de droite).

Figure 3.11 – les deux hyperboles ont des directrices parallèles.

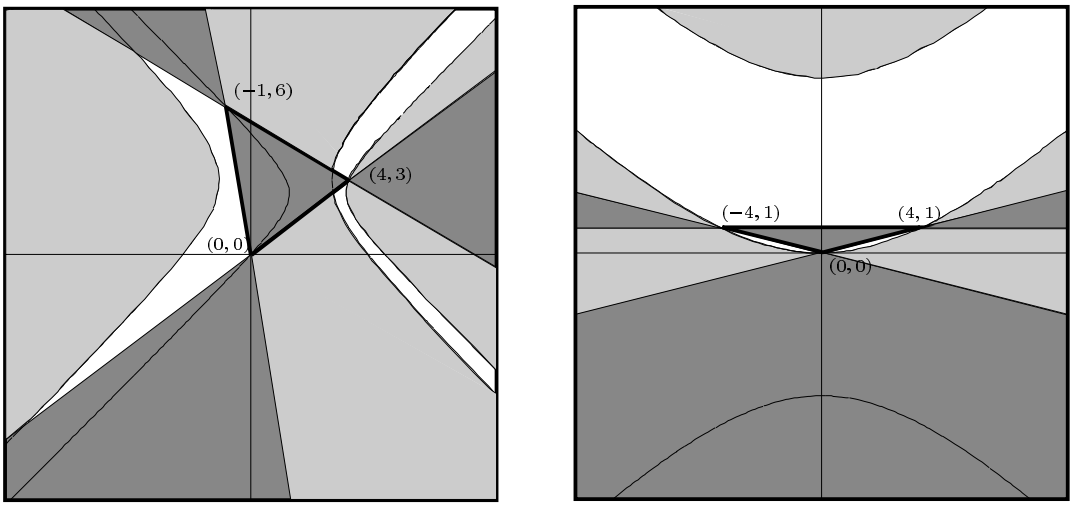


Figure 3.12 – les hyperboles ont des directrices perpendiculaires.

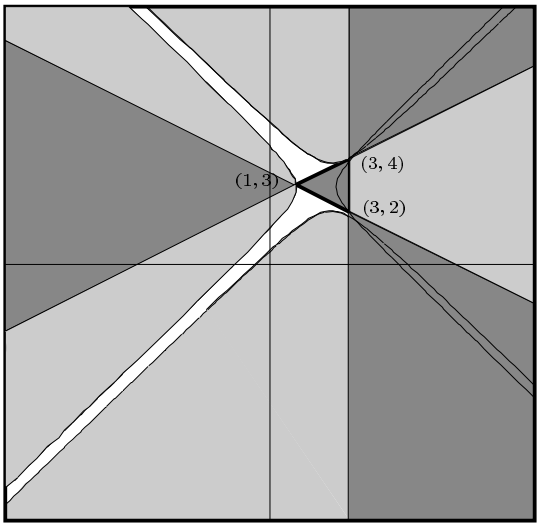
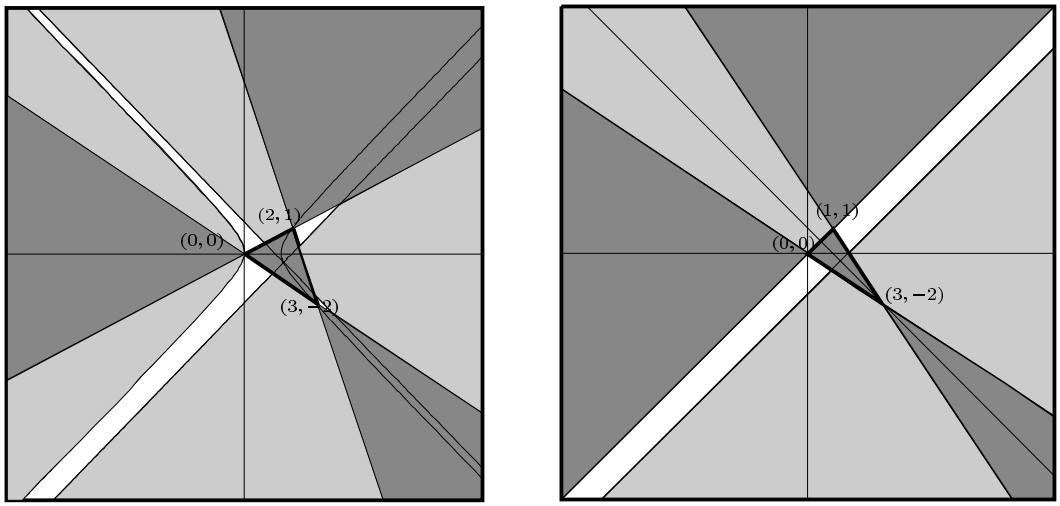


Figure 3.13 – une et même les deux hyperboles peuvent être dégénérées.



Une remarque

En observant les différentes figures, on peut constater qu'en général, la courbe de séparation d'un triangle possède certaines parties dans des zones où une seule triangulation est possible. Cela implique qu'en réalité, il existe des zones du plan dans lesquelles approcher le paraboloïde hyperbolique par un seul triangle donne une meilleure erreur L_2 que l'approcher par 3 triangles. Évidemment, comme, en principe, on désire utiliser tous les points fournis pour réaliser une approximation, et non pas en éliminer certains, cette propriété ne sera pas utilisée en pratique. Par ailleurs, rien ne prouve que, globalement, une élimination pourrait dégrader l'erreur totale (puisque la remarque faite ne considère que le placement de 4 points entre eux, sans tenir compte de l'influence éventuelle des autres sommets de l'ensemble).

3.4 Autres normes

Les succès rencontrés pour la recherche pratique des courbes de séparation de $\mathcal{PH}$ dans le cas de la norme L_2 nous ont amené à nous demander s'il était possible de faire la même chose pour d'autres normes intéressantes.

3.4.1 Norme L_1

3.4.1.1 Définition

définition 3.1 *Étant donnée une fonction f de deux variables x et y définie sur un domaine planaire $\mathcal{D}$ (fermé ou pas), la norme L_1 de f est donnée par :*

$$||f||_1 = \iint_{\mathcal{D}} |f(x,y)|\, dx\, dy.$$

définition 3.2 *Étant donnés une fonction $f : \mathbb{R}^2 \to \mathbb{R}$, $\mathcal{S}$ un ensemble de points $s_i = (x_i, y_i), i = 1 \ldots n$ du plan, $\mathcal{T}$ une triangulation de $\mathcal{S}$, à chaque triangle Δ de $\mathcal{T}$ correspond une approximation linéaire p_Δ de la fonction f valable sur toute la surface limitée par Δ. Soit $P_\mathcal{T}$ l'interpolatoin linéaire par morceaux de f définie par les p_Δ ; $P_\mathcal{T}$ est définie sur toute l'enveloppe convexe de $\mathcal{S}$. Alors,* **l'erreur** L_1 *commise par l'approximation linéaire de f suivant la triangulation $\mathcal{T}$ est donnée par la formule :*

$$\begin{aligned} e_1(f, \mathcal{T}) &= \|f - P_\mathcal{T}\|_1 \\ &= \iint_\Omega |f(x,y) - P_\mathcal{T}(x,y)|\, dx\, dy \\ &= \sum_{\Delta \in \mathcal{T}} \iint_\Delta |f(x,y) - p_\Delta(x,y)|\, dx\, dy \end{aligned}$$

Moins formellement, on peut dire que l'erreur L_1 pour l'approximation d'une fonction est la différence de volume qui existe entre la fonction et son approximation.

Ce qui nous intéresse ici est le cas où la fonction f est le paraboloïde hyperbolique unité d'équation $f(x, y) = x^2 - y^2$.

3.4.1.2 Calcul de l'erreur

Principe général

Le principal inconvénient d'une norme L_1 est qu'elle contient une valeur absolue ; il est donc plus difficile que pour la norme L_2 d'en réaliser le calcul symbolique. Cependant, on va voir qu'il est possible de se passer de cette valeur absolue.

Les intégrales à calculer ont pour domaine de valeur des triangles ; chaque triangle Δ relevé sur $\mathcal{PH}$ est porté par un plan dont l'intersection avec $\mathcal{PH}$ est une hyperbole qui passe par les 3 sommets du triangle. L'intérêt de cette hyperbole est qu'elle indique la séparation entre la partie du triangle qui est au-dessus de $\mathcal{PH}$ et celle qui est au-dessous ; ainsi, en chaque point d'intégration, on sait par quelle formule remplacer la valeur absolue. Parmi les sommets de Δ, 2 ou 3 sont sur le même arc d'hyperbole :

- dans le cas où ce sont tous les sommets, alors le triangle est toujours situé du même côté de la surface : supprimer la valeur absolue devient trivial.

- si seuls 2 sommets s_1 et s_2 sont sur un même arc, alors on peut calculer la partie de l'erreur commise sur Δ comprise entre l'arc d'hyperbole et le segment $[s_1 s_2]$, puis compléter l'erreur totale en utilisant la valeur de l'intégrale prise pour la différence $(f - p_\Delta)$ sans valeur absolue (détails page 105).

Définition et cas à envisager

Soit un triangle Δabc de sommets $a = (x_a, y_a)$, $b = (x_b, y_b)$, $c = (x_c, y_c)$ des points du plan. Va suivre la procédure exacte mise en place permettant le calcul de l'erreur L_1 locale d'approximation de $\mathcal{PH}$ par le triangle Δabc. On note A, B et

C les relevés des points sur le paraboloïde hyperbolique, et on prend la convention suivante :

définition 3.3 *On appellera dorénavant* **direction** *d'une hyperbole celle de la perpendiculaire à la directrice de l'hyperbole. Par exemple, sur la figure 3.11, les hyperboles des courbes de gauche sont dirigées suivant l'axe des x alors que celles de droite sont dirigées suivant l'axe des y.*

On détermine d'abord la direction de l'hyperbole projection de l'intersection entre $\mathcal{PH}$ et le plan ABC d'équation $z = \lambda x + \mu y + \nu$. Cette hyperbole admet pour équation :

$$x^2 - y^2 - \lambda x - \mu y - \nu = 0,$$

soit :

$$\left(x - \frac{\lambda}{2}\right)^2 - \left(y + \frac{\mu}{2}\right)^2 = \nu + \frac{\lambda^2 - \mu^2}{4} = \tau.$$

Ceci implique, puisque c'est une hyperbole équilatère, que sa direction dépend du signe de τ :

- si $\tau = 0$, on est dans le cas dégénéré où l'hyperbole est l'union de 2 droites parallèles aux bissectrices des axes.
- si $\tau < 0$, l'hyperbole est dirigée suivant l'axe des abscisses.
- si $\tau > 0$, l'hyperbole est dirigée suivant l'axe des ordonnées.

Or, τ, comme λ, μ et ν, dépend des coordonnées de a, b et c. Un calcul bien factorisé montre que sa valeur est :

$$\begin{aligned}\tau &= \frac{1}{\eta}(y_a + x_a - y_c - x_c)(y_a - x_a - y_c + x_c)(y_b + x_b - y_a - x_a) \\ &\qquad (y_b - x_b - y_a + x_a)(y_c + x_c - y_b - x_b)(y_c - x_c - y_b + x_b)\end{aligned}$$

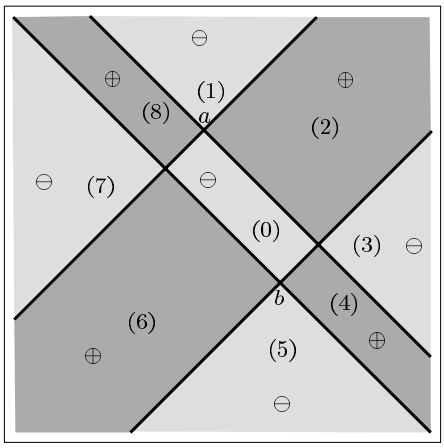
avec

$$\eta = 4\left(-x_a y_b + x_a y_c - x_b y_c + x_b y_a - x_c y_a + x_c y_b\right)^2.$$

Le dénominateur toujours positif de τ est, à une constante près, le carré de la surface du triangle Δabc, ce qui implique qu'il ne peut s'annuler sauf si les points sont alignés. Le signe de τ dépend ainsi des six facteurs de son numérateur, c'est-à-dire des positions des sommets relativement aux parallèles des bissectrices des axes passant par eux.

En effet, chaque signe peut s'interpréter de la manière suivante ; on considère $(y_a + x_a - y_c - x_c)$. Il sera positif (resp. négatif, nul) si a est au-dessus de (resp. au-desous de, sur) la droite d'équation $y = -x + x_c + y_c$, c'est-à-dire la parallèle à la deuxième bissectrice des axes de coordonnées passant par c. La figure 3.14, pour des positions relatives données de a et b indique le signe de τ, donc la direction des

Figure 3.14 – signe de τ pour différentes positions de c.



hyperboles, lorsque c varie dans le plan ; si c est sur l'une des droites en gras, alors τ est nul.

Pour bien comprendre ce qui se passe, le tableau 3.1 résume les cas non dégénérés de la figure 3.14. Dans ce tableau, pour chaque sommet est indiquée sa position relative par rapport aux parallèles des bissectrices des axes passant par les 2 autres sommets ; la convention adoptée est de prendre toujours les sommets dans l'ordre lexicographique, et de considérer d'abord la parallèle à la première bissectrice (celle d'équation $y = x$). Un "+" indique que le sommet est au-dessus de la droite, un "−" qu'il est au-dessous. Cette technique semble plus naturelle que celle qui consisterait à juste évaluer les signes des facteurs de τ, et elle permet tout aussi facilement de donner la direction de l'hyperbole. En effet, on sait que l'on doit toujours obtenir 6 signes "+" et 6 signes "−" ; c'est donc leur répartition entre les différents sommets qui va établir le signe final de τ. Si on a une distribution (pour les signes "−" par exemple) du type 3 nombres différents, alors τ est positif et l'hyperbole dirigée suivant l'axe des abscisses ; si la distribution donne au-moins 2 nombres égaux, alors τ est négatif et l'hyperbole sera dirigée suivant l'axe des ordonnées.

L'explication de ce comportement est toute simple : les signes des facteurs de τ sont donnés par une moitié bien choisie des signes reportés dans le tableau. Par exemple, si on considère tous les signes du sommet a, on aura tous les signes des facteurs de τ qui font intervenir a ; reste donc à regarder dans le tableau les 2 derniers signes de b, ou de façon équivalente les 2 derniers de c, pour obtenir les signes des facteurs de τ qui utilisent b et c. Ce raisonnement fonctionne bien, car on utilise toujours les signes par paires ("+−" revient ainsi au même que "−+", de même que "−−" et "++").

En plus de fournir la direction de l'hyperbole, l'étude des signes permet aussi de déterminer les sommets qui seront sur un même arc ; on peut établir la discussion suivante (on suppose toujours que chaque sommet n'appartient pas à la parrallèle

Tableau 3.1 – direction et choix de l'arc d'hyperbole pour l'exemple de la figure 3.14.

zone	a	b	c	direc.	arc	sommets
0	$++++$	$----$	$--++$	x	$\pm$	abc
1	$++--$	$----$	$++++$	x	$\pm$	abc
2	$+++-$	$----$	$-+++$	y	$+$	ac
3	$+++-$	$--+-$	$-+-+$	x	$-$	ab
4	$++++$	$--+-$	$---+$	y	$-$	bc
5	$++++$	$--++$	$----$	x	$\pm$	abc
6	$++++$	$---+$	$--+-$	y	$-$	bc
7	$++-+$	$---+$	$+-+-$	x	$+$	ab
8	$++-+$	$----$	$+-++$	y	$+$	ac

d'une bissectrice des axes passant par les autres sommets) :

- l'hyperbole est dirigée suivant l'axe des ordonnées ; au-moins 2 des nombres de signes "–" sont égaux :
  - s'ils sont tous égaux, les 3 sommets sont sur le même arc.
  - sinon, ce sont les 2 sommets qui fournissent le même nombre qui sont sur le même arc.
- l'hyperbole est dirigée suivant l'axe des abscisses ; les nombres sont tous différents :
  - s'ils sont tous pairs, les 3 sommets sont sur le même arc.
  - si 2 sont impairs, les 2 sommets correspondants sont sur le même arc.

Pour définir alors quel est le bon arc parmi les 2 possibles, seules les positions relatives des points permettent de conclure. On peut voir ainsi, que dans les zones 0, 1 et 5 de la figure 3.14, suivant la position de c les 2 arcs sont possibles ; en général, ce n'est pas le cas. Dans le tableau, un signe "+" signifie que le bon arc est celui qui inclut le $+\infty$ de l'axe de direction de l'hyperbole.

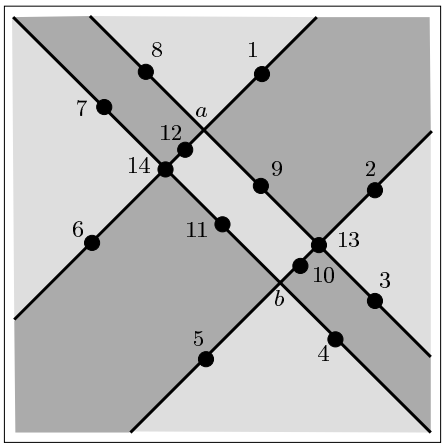
Tous les cas non dégénérés ne sont pas reportés dans le tableau 3.1, car on a supposé avoir ici une configuration particulière de a et b (qui fait apparaître les signes "++" pour a et "––" pour b lorsqu'on regarde leurs situations réciproques). Dans le tableau 3.2 sont données toutes les configurations génériques possibles : on ne donne pas d'ordre spécial aux sommets (qui sont rebaptisés s_1, s_2 et s_3), et on indique seulement le nombre de signes "–" pour chacun. Cela entraine qu'il n'est pas possible de connaitre précisément leurs positions relatives et donc pas possible de déterminer sur quel arc se trouvent les 2 ou 3 sommets (sinon, le nombre de cas serait trop important), mais aussi qu'une répartition "$0-2-4$" est équivalente à une répartition "$4-0-2$" ; seuls les sommets présents sur le même arc changent. Le tableau renvoie aussi les zones de la figure 3.14 et le numéro de la figure qui correspondent à chacun des cas.

Tableau 3.2 – configurations non dégénérées possibles.

cas	s_1	s_2	s_3	direc.	sommets	zone	figure
1	0	2	4	x	$s_1s_2s_3$	0 1 5	3.16 droite
2	0	3	3	y	s_2s_3	4 6	3.17 droite
3	1	1	4	y	s_1s_2	2 8	3.18 droite
4	1	2	3	x	s_1s_3	3 7	3.17 gauche
5	2	2	2	y	$s_1s_2s_3$		3.16 gauche

Pour les cas particuliers où un sommet est sur une des parallèles aux bissectrices des axes passant par un autre sommet, l'hyperbole est dégénérée en deux droites perpendiculaires parallèles aux bissectrices. C'est le (ou les) facteurs de τ qui est nul qui déterminera les points qui seront sur la même droite. Un autre problème apparait alors : la droite qui ne contient qu'un seul sommet peut ou pas couper l'arête opposée du triangle (figure 3.19). Or, être au courant de cette propriété est essentiel, car cela modifie la manière de calculer l'erreur.

Figure 3.15 – positions de c faisant apparaître un cas dégénéré.



Pour mieux comprendre les cas dégénérés, on repart des configurations initiales de a et b de la figure 3.14, et on s'intéresse maintenant aux emplacements que c peut occuper afin d'aboutir à des cas dégénérés ; c'est ce qui est fait sur la figure 3.15. Le tableau 3.3 contient alors toutes les conclusions possibles relatives à cette figure. Sa lecture est identique à celle du tableau 3.1, sauf qu'un 0 apparait lorsque 2 sommets sont sur une parallèle commune à une bissectrice ; le segment indiqué est alors celui qui est supporté par la droite en question, et la dernière colonne indique si l'autre droite va couper ou pas le triangle. On peut observer que le triangle ne peut être coupé par l'autre droite que si une seule de ses arêtes est supportée par l'une des droites, et seulement lorsque le sommet qui n'est pas sur cette arête comporte un nombre pair de signes "–".

Tableau 3.3 – cas dégénérés associés à la figure 3.15.

place	a	b	c	segment	Δ coupé
1	$++0-$	$----$	$0\ +++$	$[ac]$	non
2	$+++-$	$--0-$	$-+0+$	$[bc]$	oui
3	$+++0$	$--+-$	$-0-+$	$[ac]$	oui
4	$++++$	$---0$	$--+0$	$[bc]$	non
5	$++++$	$--0+$	$--0-$	$[bc]$	non
6	$++0+$	$----+$	$0-+-$	$[ac]$	oui
7	$++-+$	$---0$	$+-+0$	$[bc]$	oui
8	$++-0$	$----$	$+++0$	$[ac]$	non
9	$+++0$	$----$	$-0++$	$[ac]$	non
10	$++++$	$--0-$	$--0+$	$[bc]$	non
11	$++++$	$--+0$	$---0$	$[bc]$	non
12	$++0+$	$-----$	$0-++$	$[ac]$	non
13	$+++0$	$--0-$	$-0\ 0+$	$[ac]$ et $[bc]$	non
14	$++0+$	$---0$	$0-+0$	$[ac]$ et $[bc]$	non

Tous les cas dégénérés génériques possibles sont reportés dans le tableau 3.4. Les "0" désignent les sommets qui sont sur une même parallèle aux bissectrices, et "$2p$" ou "$2p+1$" donnent la parité du nombre de signes négatifs pour le troisième sommet. On trouve aussi dans le tableau la correspondance cas générique - place pour la figure 3.15. Enfin, les figures qui illustrent chaque cas sont indiquées dans la dernière colonne.

Tableau 3.4 – configurations dégénérées possibles.

cas	s_1	s_2	s_3	segment	Δ coupé	place	figure
1	0	0	$2p$	$[s_1s_2]$	non	1 4 5 8 9 10 11 12	3.18 gauche
2	0	0	$2p+1$	$[s_1s_2]$	oui	2 3 6 7	3.19 droite
3	0 0	0	0	$[s_1s_2]$	non	13 14	3.19 gauche

Calcul de l'erreur L_1

On note $I_{|.|}$ la valeur de l'erreur L_1 pour le triangle Δabc, I la valeur de l'intégrale pour la fonction prise sans valeur absolue et $I^+_{|.|}$ (resp. $I^-_{|.|}$) la valeur (positive) de l'erreur restreinte à la partie du triangle au-dessous (resp. au-dessus) du paraboloïde

Figure 3.16 – des cas où les 3 sommets du triangles sont sur le même arc d'hyperbole.

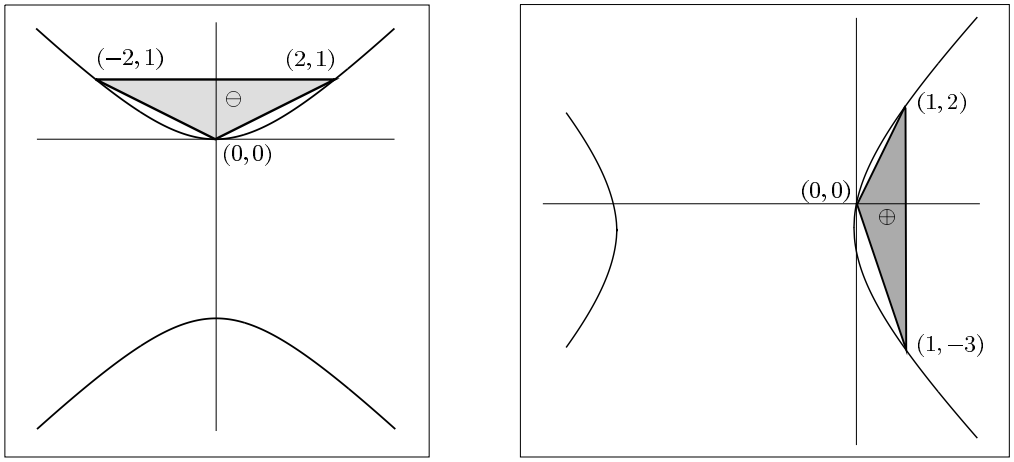
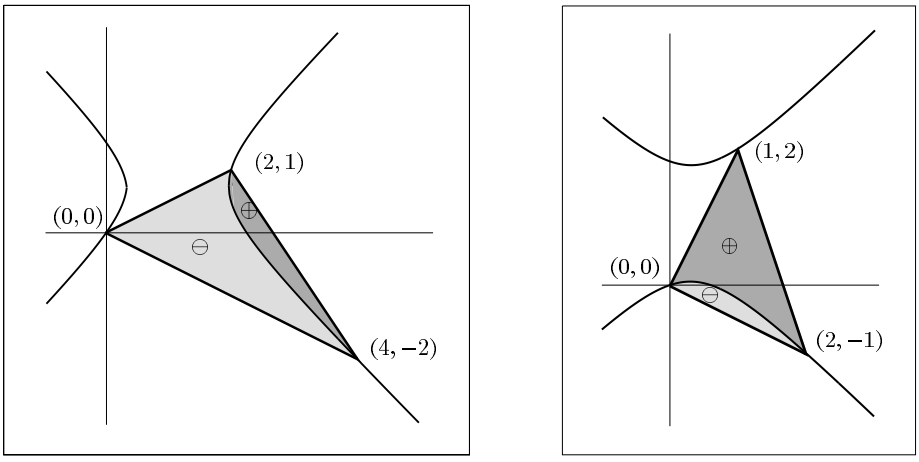


Figure 3.17 – des cas standards, avec 2 sommets sur un même arc.



hyperbolique. Alors, on a les relations :

$$\begin{cases} I_{|.|} &= I^+_{|.|} + I^-_{|.|} &= \iint_{\Delta abc} |f - p_{\Delta abc}| \\ I &= I^+_{|.|} - I^-_{|.|} &= \iint_{\Delta abc} f - p_{\Delta abc} \end{cases}$$

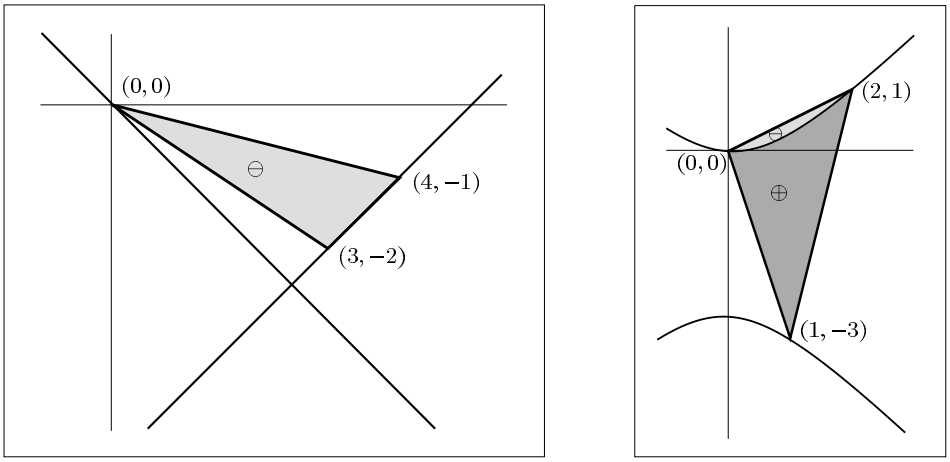
ce qui implique que, si on connait I et $I^+_{|.|}$ (resp. $I^-_{|.|}$), on peut déduire la valeur de $I_{|.|}$: $I_{|.|} = 2I^+_{|.|} - I$ (resp. $I_{|.|} = 2I^-_{|.|} + I$). Le calcul de I est trivial, et choisir les cas où il faut utiliser $I^+_{|.|}$ plutôt que $I^-_{|.|}$ revient à utiliser les discussions précédentes sur la direction des hyperboles d'intersection de $\mathcal{PH}$ et du plan ABC ; si elles sont dirigées suivant l'axe des x (dessin de gauche de la figure 3.17), alors c'est la partie positive qui est facilement calculable, et si elles sont dirigées suivant l'axe des y (dessin de droite de la même figure), il sera plus facile de calculer $I^-_{|.|}$.

Pour les cas dégénérés :

– si les 3 sommets sont sur le même arc d'hyperbole (figure 3.16), alors $I_{|.|}$ vaut exactement la valeur absolue de I, et il est facile de connaitre à l'avance le signe de I (il dépend de la direction de l'hyperbole).

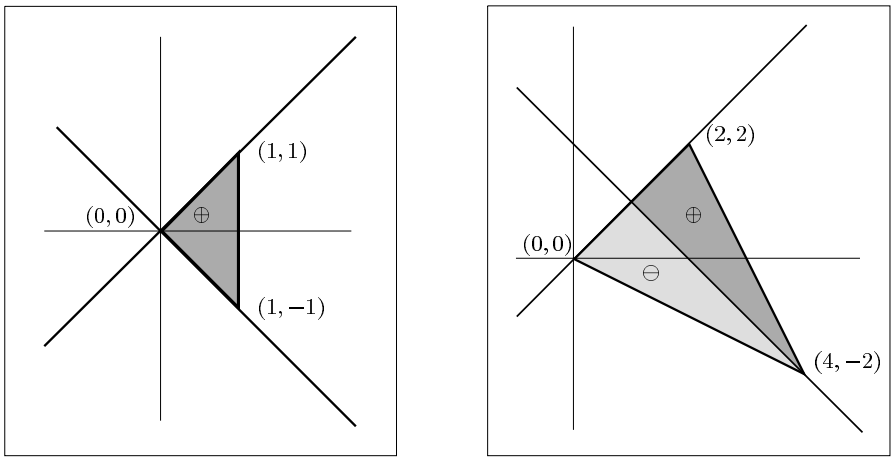
– si les hyperboles sont des droites (figure 3.19), 2 cas se présentent :
  - le triangle ΔABC est situé d'un seul côté du paraboloïde hyperbolique (dessin de gauche) : on retombe dans le même cas de calcul que pour la première dégénérescence.
  - le triangle comporte une partie au-dessus et une au-dessous de la surface (dessin de droite) : cette fois-ci, on obtiendra directement $I_{|.|}$ en faisant $I_{|.|} = I^+_{|.|} + I^-_{|.|}$, où $I^+_{|.|}$ et $I^-_{|.|}$ peuvent alors se calculer de manière évidente.

Figure 3.18 – des cas mixtes.



Le calcul de I revient à utiliser la procédure décrite pour la norme L_2 (le carré en moins) au paragraphe 3.2.1 ; il est donc facile de le faire formellement, à condition

Figure 3.19 – des cas où les hyperboles sont dégénérées.



de connaitre les placements relatifs de a, b et c.
Pour $I^+_{|.|}$ et $I^-_{|.|}$ (cas non dégénérés), on a besoin de :

- connaitre les extrémités s_1 et s_2 (qui sont a, b ou c) du segment limite, ainsi que leurs positions relatives.

- savoir sur quel arc de l'hyperbole se trouvent s_1 et s_2, cela afin d'intégrer dans le bon ordre.

Exemples de calculs

Pour le cas de gauche de la figure 3.17, la discussion des positions par rapport aux bissectrices nous montre que ce sont les sommets $(2,1)$ et $(4,-2)$ qui se trouvent sur l'arc d'hyperbole dirigée vers les x positifs : il va donc être facile de calculer $I^+_{|.|}$, sachant que l'on devra intégrer de l'arc vers le segment pour conserver une valeur positive. Le plan passant par les relevés des points a pour équation $z = \frac{9}{4}x - \frac{3}{2}y$, le segment $[(2,1),(4,-2)]$ est porté par la droite $x = -\frac{2}{3}y + \frac{8}{3}$, et l'arc d'hyperbole passant par ces 2 sommets est celui d'équation $x = \sqrt{(y-\frac{3}{4})^2 + \frac{45}{64}} + \frac{9}{8}$.
Par conséquent, $I^+_{|.|}$ sera donnée par :

$$I^+_{|.|} = \int_{y=-2}^{y=1} \int_{x=\sqrt{(y-\frac{3}{4})^2+\frac{45}{64}}+\frac{9}{8}}^{x=-\frac{2}{3}y+\frac{8}{3}} \left(x^2 - y^2 - \frac{9}{4}x + \frac{3}{2}y \right) \, dx \, dy$$

Le calcul final donne :

$$\begin{cases} I &= -\dfrac{10}{3} \\ I^+_{|.|} &= \dfrac{625}{6144} + \dfrac{2025}{16384}\,\mathrm{Argsh}\left(\dfrac{2\sqrt{5}}{15}\right) + \dfrac{2025}{16384}\,\mathrm{Argsh}\left(\dfrac{22\sqrt{5}}{15}\right) \cong 0.37329 \end{cases}$$

soit

$$I_{|.|} = \frac{10865}{3072} + \frac{2025}{8192}\,\mathrm{Argsh}\left(\frac{2\sqrt{5}}{15}\right) + \frac{2025}{8192}\,\mathrm{Argsh}\left(\frac{22\sqrt{5}}{15}\right) \cong 4.07992.$$

Pour la configuration de droite de la figure 3.17, les sommets $[0,0]$ et $[2,-1]$ sont sur l'arc du bas de l'hyperbole dirigée suivant l'axe des ordonnées d'équation $y = -\sqrt{(x-\frac{3}{10})^2 + \frac{18}{25}} + \frac{9}{10}$; le segment $[(0,0),(2,-1)]$ est porté par la droite $y = -\frac{x}{2}$ et le triangle relevé définit le plan $z = \frac{3}{5}x - \frac{9}{5}y$. Ici, il vaut mieux calculer $I^-_{|.|}$:

$$I^-_{|.|} = \int_{x=0}^{x=2} \int_{y=-\frac{x}{2}}^{y=-\sqrt{(x-\frac{3}{10})^2+\frac{18}{25}}+\frac{9}{10}} \left(\frac{3}{5}x - \frac{9}{5}y - x^2 + y^2\right)\, dy\, dx.$$

Ce qui donne, numériquement :

$$\begin{cases} I &= \dfrac{5}{3} \\ I^-_{|.|} &= \dfrac{231}{2000} - \dfrac{81}{625}\,\mathrm{Argsh}\left(\dfrac{17\sqrt{2}}{12}\right) - \dfrac{81}{625}\,\mathrm{Argsh}\left(\dfrac{\sqrt{2}}{4}\right) \cong 0.11671 \end{cases}$$

soit

$$I_{|.|} = \frac{4307}{3000} + \frac{162}{625}\,\mathrm{Argsh}\left(\frac{17\sqrt{2}}{12}\right) + \frac{162}{625}\,\mathrm{Argsh}\left(\frac{\sqrt{2}}{4}\right) \cong 1.90009.$$

3.4.1.3 Courbes de séparation

L'idéal pour trouver formellement les courbes serait d'appliquer le même genre de raisonnement que pour la norme L_2 (voir section 3.2), et que cela aboutisse. On note ainsi $e_1(\Delta)$ l'erreur L_1 d'approximation locale du paraboloïde hyperbolique pour le triangle Δ. Pour 4 points a, b, c, d tels que le polygone $\Box abcd$ soit convexe, on suppose que les deux triangulations possibles sont celles qui font intervenir Δabc et Δbcd d'une part, et Δabd et Δacd d'autre part. On a alors :

$$\begin{aligned} e_1(\Delta abc + \Delta bcd) &= e_1(\Delta abc) + e_1(\Delta bcd) \\ e_1(\Delta abd + \Delta acd) &= e_1(\Delta abd) + e_1(\Delta acd) \end{aligned}$$

Le problème est donc de réussir à calculer formellement les erreurs pour chacun des triangles ; or, comme on vient de le voir, ces erreurs sont directement liées aux placements relatifs des points. On est ainsi obligé, dans un premier temps, de supposer que les sommets sont dans une configuration précise. Et, on va voir que, même dans ce cas, le résultat est peu exploitable (comme le laissait supposer les formules obtenues qui incluent des Argsh). Pour avoir une idée de ce que cela peut bien donner, nous allons reproduire ici les formules des erreurs sur un triangle obtenues pour

les 3 cas dégénérés, et pour les 2 cas non dégénérés les plus simples :

– cas dégénérés 2 et 3, et cas non dégénérés 1 et 5 :

$$\begin{aligned} e_1(\Delta abc) &= \frac{1}{12}\left(x_a y_b - x_a y_c + x_b y_c - x_b y_a + x_c y_a - x_c y_b\right) \\ &\quad \left(x_a^2 + x_b^2 + x_c^2 - x_a x_b - x_a x_c - x_b x_c - y_a^2 - y_b^2 - y_c^2\right. \\ &\quad \left. + y_a y_b + y_a y_c + y_b y_c\right) \end{aligned}$$

– pour le cas dégénéré 2, il n'est pas possible de réaliser un calcul en partant de points a, b, c quelconques ; mais, on sait que 2 des sommets sont sur une même parallèle à une bissectrice. On peut donc supposer, par exemple, que b est tel que $x_b = x_a + r$ et $y_b = y_a + r$ avec r un réel différent de 0. On obtient alors :

$$\begin{aligned} e_1(\Delta abc) &= \frac{1}{24}\left(x_a - x_c - y_a + y_c\right)^2 \\ &\quad \left(x_a^2 + x_c^2 - 2x_a x_c + 2t x_a - 2t x_c + y_a^2 + y_c^2 - 2y_a y_c\right. \\ &\quad \left. + 2t y_a - 2t y_c + 2t^2 + 2x_a y_a + 2x_c y_c - 2x_a y_c - 2x_c y_a\right) \end{aligned}$$

Pour les autres cas non dégénérés, il est possible de calculer formellement l'erreur ; cependant, le résultat final est illisible (sa reproduction ne tiendrait pas sur moins de 3 pages) et demande un temps de calcul très important.

Évidemment, cela implique que l'on n'est pas en mesure de trouver formellement les équations des courbes de séparation d'un triangle pour la norme L_1 ; en revanche, une adaptation du programme de recherche de courbes (voir paragraphe 3.1) permet de les visualiser, et des exemples sont visibles sur la figure 3.20.

Il semble que ce soit à nouveau deux courbes dont l'une pourrait être l'hyperbole équilatère (ou les droites pour les cas dégénérés) passant par les 3 sommets du triangle. L'autre bien que ressemblant beaucoup à une hyperbole parait être autre chose ; c'est notamment visible sur la première représentation de courbes de la figure 3.20. Cette supposition est confirmée par la présence de logarithmes ou de fonctions arc sinus hyperbolique dans les formules finales.

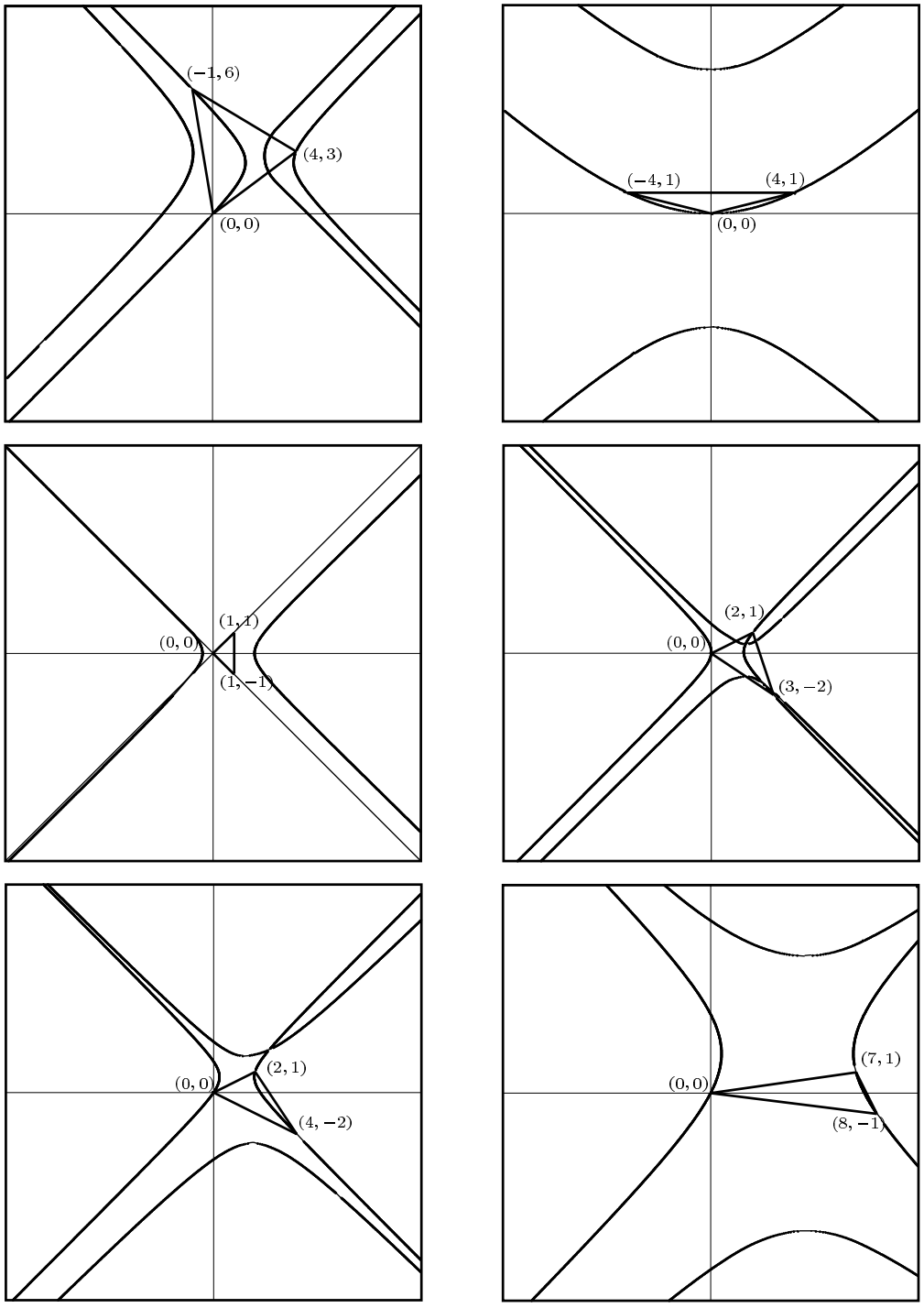
3.4.2 Norme L_∞

définition 3.4 *Étant donnée une fonction f de deux variables x et y définie sur un domaine planaire $\mathcal{D}$ (fermé ou pas), la norme L_∞ de f est donnée par :*

$$\|f\|_\infty = \max_{(x,y)\in\mathcal{D}} |f(x,y)| .$$

définition 3.5 *Étant donnés une fonction $f : \mathbb{R}^2 \to \mathbb{R}$, $\mathcal{S}$ un ensemble de points $p_i = (x_i, y_i), i = 1 \ldots n$ du plan, $\mathcal{T}$ une triangulation de $\mathcal{S}$, à chaque triangle Δ de $\mathcal{T}$ correspond une approximation linéaire p_Δ de la fonction f valable sur toute la surface limitée par Δ. Soit $P_\mathcal{T}$ l'interpolatoin linéaire par morceaux de f définie par les $p_\Delta s$; $P_\mathcal{T}$ est définie sur toute l'enveloppe convexe de $\mathcal{S}$. Alors, l'**erreur** L_∞ commise par l'approximation linéaire de f suivant la triangulation $\mathcal{T}$ est donnée par*

Figure 3.20 – des exemples de courbes de séparation pour le critère de la norme L_1.



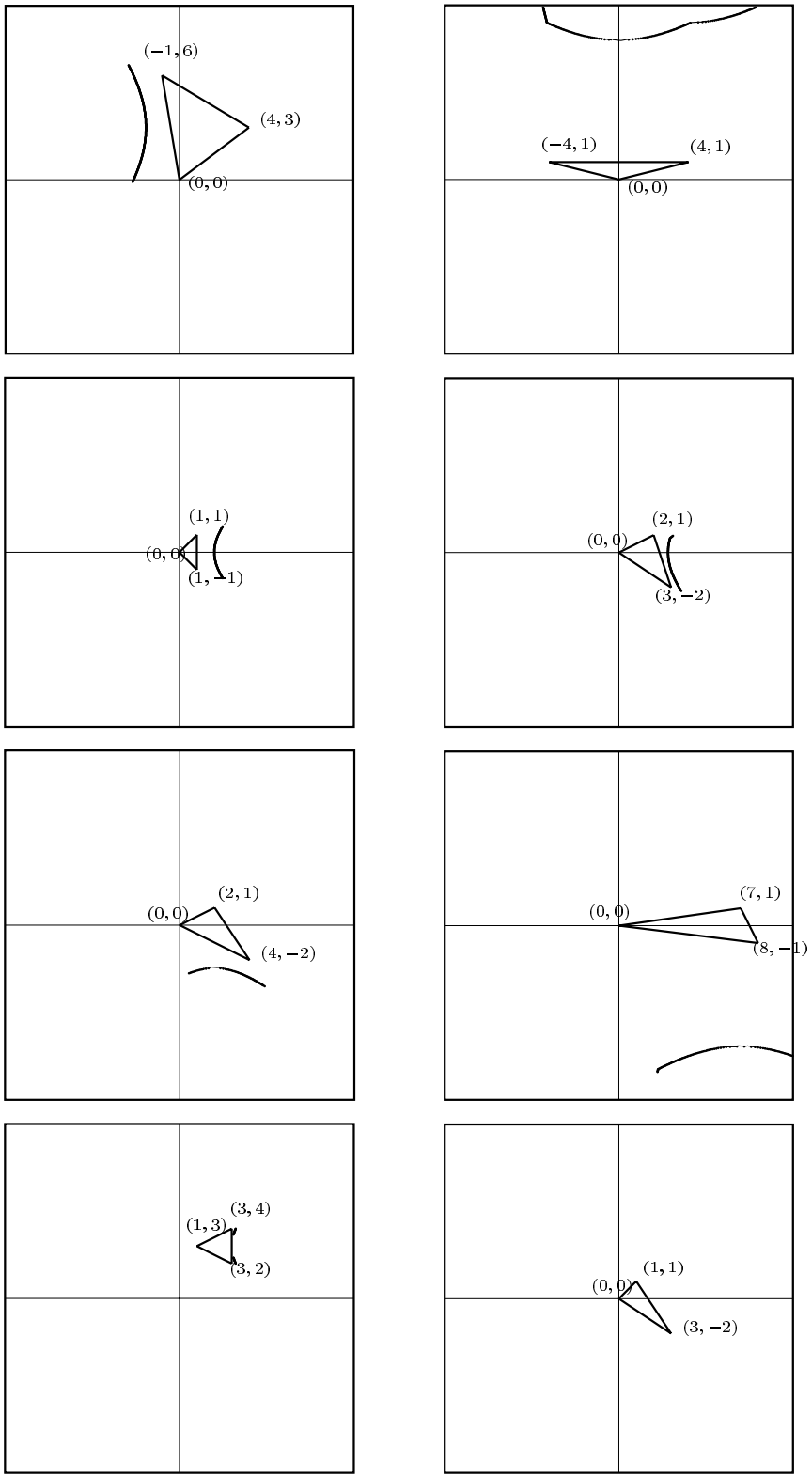
la formule :

$$\begin{aligned} e_\infty(f,\mathcal{T}) &= \|f - P_\mathcal{T}\|_\infty \\ &= \max_{(x,y)\in\Omega} |f(x,y) - P_\mathcal{T}(x,y)| \\ &= \max_{\Delta\in\mathcal{T}} \left(\max_{(x,y)\in\Delta} |f(x,y) - p_\Delta(x,y)| \right). \end{aligned}$$

A FINIR, sachant que :

- il a déjà été prouvé que la valeur maximale de l'erreur sur un triangle, pour l'approximation du paraboloïde hyperbolique, se situe au milieu de l'une des arêtes du triangle.
- je soupçonne très fortement que le paradigme "insertion d'arêtes" de EDELSBRUNNER et al. doit s'adapter à ce problème ; ce qui implique qu'un algorithme de recherche de la triangulation globalement optimale existe, en temps polynomial.

Figure 3.21 – exemples de courbes de séparation pour le critère de la norme L_∞.



Chapitre 4

Généralisation aux quadriques

Nous venons de constater qu'il était possible de trouver les équations des courbes de séparation d'un triangle pour le critère de l'erreur L_2 d'approximation linéaire du paraboloïde hyperbolique, et cela en ayant uniquement recours à des observations et à du calcul formel.

Ce "raisonnement" n'est pas tout à fait satisfaisant ; nous allons donc montrer qu'il est en réalité possible de prouver et de généraliser certains résultats, obtenus pour le paraboloïde hyperbolique, à toutes les quadriques d'équation $z = f(x, y)$.

Dans un premier temps, nous montrerons que l'erreur L_p (où p est un entier fini quelconque) d'approximation linéaire par morceaux d'une telle quadrique est stable par translation des 3 sommets définissant le plan approchant. Nous verrons ensuite que l'on peut exprimer l'erreur L_2 d'approximation sur un triangle, et qu'il est ainsi possible d'obtenir, assez facilement, ses courbes de séparation.

4.1 Stabilité de l'erreur par translation

Avant de voir le théorème clef de ce paragraphe, quelques définitions et propriétés vont être utiles.

Tout d'abord, un petit rappel d'algèbre linéaire: pour calculer l'inverse d'une matrice carrée quelconque d'ordre 3 (matrice à 3 lignes et 3 colonnes), le plus simple est d'utiliser la *comatrice* de celle-ci.

définition 4.1 *Soit* $\mathcal{M} = (a_{ij})_{\substack{1 \leq i \leq n \\ 1 \leq j \leq n}}$ *une matrice carrée d'ordre* n. *Alors, la* ***co-matrice*** *de* $\mathcal{M}$, *notée* $Com\,(\mathcal{M})$, *est la matrice* $(c_{ij})_{\substack{1 \leq i \leq n \\ 1 \leq j \leq n}}$ *où les* c_{ij} *sont obtenus en prenant le déterminant de la matrice d'ordre* $(n-1)$ *construite en supprimant la* i*-ème ligne et la* j*-ème colonne de* $\mathcal{M}$, *et en multipliant ce déterminant par* $(-1)^{i+j}$.

définition 4.2 *Soit* $\mathcal{M}$ *une matrice carrée; si* $\mathcal{M}$ *est inversible (c'est-à-dire que son déterminant est non nul), alors:*

$$\mathcal{M}^{-1} = \frac{1}{det\mathcal{M}}\, Com\,({}^t\mathcal{M}),$$

Nous aurons également besoin de savoir exprimer la surface d'un triangle en fonction du déterminant d'une matrice définie par les coordonnées de ses sommets.

propriété 4.1 *Soit* a, b *et* c *trois points du plan; alors, la surface du triangle* Δabc *(notée* $Surf(\Delta abc)$*) est égale à la moitié de la valeur absolue du déterminant de la matrice*

$$\begin{pmatrix} x_a & x_b & x_c \\ y_a & y_b & y_c \\ 1 & 1 & 1 \end{pmatrix}.$$

Dans toute la suite de ce paragraphe, lorsque l'on parlera d'*approximation linéaire spatiale* d'une surface d'équation $z = f(x, y)$ pour un triangle Δabc, cela voudra dire que l'on utilise le plan passant par les relevés de a, b et c sur la surface.

propriété 4.2 *Soient* a, b *et* c *trois points du plan, et* $\vec{u} = (u_x, u_y)$ *un vecteur quelconque; on note* $t_{\vec{u}}$ *la translation de vecteur* $\vec{u}$. *Soit* f *une quadrique définie par une fonction, c'est-à-dire une quadrique d'équation:*

$$f(x, y) = \alpha x^2 + \beta y^2 + \gamma xy + \delta_1 x + \delta_2 y + \delta_3,$$

où α, β, γ, *et les* δ_i *sont des nombres réels.*
Soit $p_{\Delta abc}$ *l'approximation linéaire spatiale de* f *pour le triangle* Δabc *et* $p'_{\Delta abc}$ *celle définie pour le triangle de sommets les images de* a, b *et* c *par la translation* $t_{\vec{u}}$. *Si on pose* $p_{\Delta abc}(x, y) = \lambda x + \mu y + \nu$ *et* $p'_{\Delta abc}(x, y) = \lambda' x + \mu' y + \nu'$, *alors on passera de* $p_{\Delta abc}$ *à* $p'_{\Delta abc}$ *par le système:*

$$\begin{cases} \lambda' &= \lambda + 2\alpha u_x + \gamma u_y \\ \mu' &= \mu + 2\beta u_y + \gamma u_x \\ \nu' &= \nu - \lambda u_x - \mu u_y + f(u_x, u_y) - 2(\alpha u_x^2 + \beta u_y^2 + \gamma u_x u_y) - \delta_3 \end{cases}$$

démonstration :

nous allons d'abord chercher à exprimer λ, μ et ν en fonction des sommets de départ et de la fonction f.
Par définition de l'approximation spatiale, si on note A, B et C les relevés respectifs de a, b et c sur la surface $\mathcal{S}$ d'équation $z = f(x, y)$, on a :

$$\left\{ \begin{array}{rcccl} z_A & = & f(x_a, y_a) & = & \lambda x_a + \mu y_a + \nu \\ z_B & = & f(x_b, y_b) & = & \lambda x_b + \mu y_b + \nu \\ z_C & = & f(x_c, y_c) & = & \lambda x_c + \mu y_c + \nu \end{array} \right.$$

Par conséquent, les coefficients λ, μ et ν définissant $p_{\Delta abc}$ vérifient le système linéaire :

$$\begin{pmatrix} z_A \\ z_B \\ z_C \end{pmatrix} = \begin{pmatrix} x_a & y_a & 1 \\ x_b & y_b & 1 \\ x_c & y_c & 1 \end{pmatrix} \begin{pmatrix} \lambda \\ \mu \\ \nu \end{pmatrix}.$$

Pour trouver ces coefficients, il suffit de résoudre le système inverse :

$$\begin{pmatrix} \lambda \\ \mu \\ \nu \end{pmatrix} = \begin{pmatrix} x_a & y_a & 1 \\ x_b & y_b & 1 \\ x_c & y_c & 1 \end{pmatrix}^{-1} \begin{pmatrix} z_A \\ z_B \\ z_C \end{pmatrix} = \mathcal{M}^{-1} \begin{pmatrix} z_A \\ z_B \\ z_C \end{pmatrix}.$$

On remarque que :

$$\mathcal{M} = {}^t \begin{pmatrix} x_a & x_b & x_c \\ y_a & y_b & y_c \\ 1 & 1 & 1 \end{pmatrix}$$

Par conséquent, d'après la propriété 4.1,

$$|\det \mathcal{M}| = 2\,\mathrm{Surf}(\Delta abc).$$

On va noter $\overline{\mathrm{Surf}(\Delta abc)}$ une sorte de valeur algébrique de la surface du triangle, qui sera positive si Δabc est un triangle direct, négative sinon (nulle si le triangle est plat).
On peut remarquer ici que le système admettra une solution si et seulement si le triangle choisi n'est pas dégénéré (sinon $\det \mathcal{M} = 0$).
De plus :

$$Com\,({}^t\mathcal{M}) = \begin{pmatrix} \begin{vmatrix} y_b & y_c \\ 1 & 1 \end{vmatrix} & -\begin{vmatrix} y_a & y_c \\ 1 & 1 \end{vmatrix} & \begin{vmatrix} y_a & y_b \\ 1 & 1 \end{vmatrix} \\ -\begin{vmatrix} x_b & x_c \\ 1 & 1 \end{vmatrix} & \begin{vmatrix} x_a & x_c \\ 1 & 1 \end{vmatrix} & -\begin{vmatrix} x_a & x_b \\ 1 & 1 \end{vmatrix} \\ \begin{vmatrix} x_b & x_c \\ y_b & y_c \end{vmatrix} & -\begin{vmatrix} x_a & x_c \\ y_a & y_c \end{vmatrix} & \begin{vmatrix} x_a & x_b \\ y_a & y_b \end{vmatrix} \end{pmatrix}$$

soit

$$Com\,({}^t\mathcal{M}) = \begin{pmatrix} y_b - y_c & y_c - y_a & y_a - y_b \\ x_c - x_b & x_a - x_c & x_b - x_a \\ x_b y_c - y_b x_c & x_c y_a - y_c x_a & x_a y_b - y_a x_b \end{pmatrix}$$

d'où on peut finalement tirer :

$$\begin{pmatrix} \lambda \\ \mu \\ \nu \end{pmatrix} = s.\begin{pmatrix} (y_b - y_c)z_A + (y_c - y_a)z_B + (y_a - y_b)z_C \\ (x_c - x_b)z_A + (x_a - x_c)z_B + (x_b - x_a)z_C \\ (x_b y_c - y_b x_c)z_A + (x_c y_a - y_c x_a)z_B + (x_a y_b - y_a x_b)z_C \end{pmatrix},$$

avec

$$s = \frac{1}{2.\overline{\text{Surf}(\Delta abc)}}.$$

Lorsque l'on applique $t_{\vec{u}}$ à un point du plan, on obtient :

$$\begin{aligned} f(t_{\vec{u}}(x,y)) &= \alpha(x+u_x)^2 + \beta(y+u_y)^2 + \gamma(x+u_x)(y+u_y) \\ &\qquad +\delta_1(x+u_x) + \delta_2(y+u_y) + \delta_3 \\ &= f(x,y) + f(u_x,u_y) + 2\alpha x u_x + 2\beta y u_y \\ &\qquad +\gamma(x u_y + y u_x) - \delta_3. \end{aligned}$$

D'autre part, de manière analogue à ce qui a été fait pour $p_{\Delta abc}$, on peut regarder le système vérifié par $p'_{\Delta abc}$:

$$\begin{pmatrix} z_{A'} \\ z_{B'} \\ z_{C'} \end{pmatrix} = \begin{pmatrix} x_{a'} & y_{a'} & 1 \\ x_{b'} & y_{b'} & 1 \\ x_{c'} & y_{c'} & 1 \end{pmatrix} \begin{pmatrix} \lambda' \\ \mu' \\ \nu' \end{pmatrix} = \mathcal{M}' \begin{pmatrix} \lambda' \\ \mu' \\ \nu' \end{pmatrix}.$$

Or :

$$\det\mathcal{M}' = \begin{vmatrix} x_a + u_x & y_a + u_y & 1 \\ x_b + u_x & y_b + u_y & 1 \\ x_c + u_x & y_c + u_y & 1 \end{vmatrix} = \det\mathcal{M},$$

puisque le déterminant d'une matrice ne change pas lorsqu'à une colonne (resp. une ligne), on ajoute une combinaison linéaire des autres colonnes (resp. lignes) et que :

$$\begin{pmatrix} x_a + u_x \\ x_b + u_x \\ x_c + u_x \end{pmatrix} = \begin{pmatrix} x_a \\ x_b \\ x_c \end{pmatrix} + u_x \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} y_a + u_y \\ y_b + u_y \\ y_c + u_y \end{pmatrix} = \begin{pmatrix} y_a \\ y_b \\ y_c \end{pmatrix} + u_y \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

On remarque alors qu'en effectuant le calcul, les deux premières lignes de $\mathcal{M}'^{-1}$, matrice inverse de $\mathcal{M}'$, sont les mêmes que celles de $\mathcal{M}^{-1}$ et que sa troisième ligne est la transposée du vecteur colonne suivant :

$$\begin{pmatrix} x_b y_c - y_b x_c + u_x(y_c - y_b) + u_y(x_b - x_c) \\ x_c y_a - y_c x_a + u_x(y_a - y_c) + u_y(x_c - x_a) \\ x_a y_b - y_a x_b + u_x(y_b - y_a) + u_y(x_a - x_b) \end{pmatrix}$$

Par conséquent :

$$\lambda' = \frac{1}{\det\mathcal{M}}[(y_b - y_c)z_{A'} + (y_c - y_a)z_{B'} + (y_a - y_b)z_{C'}]$$

avec :

$$\begin{aligned} z_{A'} &= f(t_{\vec{u}}(x_a, y_a)) \\ &= f(x_a, y_a) + f(u_x, u_y) + 2\alpha x_a u_x + 2\beta y_a u_y + \gamma(x_a u_y + y_a u_x) - \delta_3 \\ &= z_A + f(u_x, u_y) + 2\alpha x_a u_x + 2\beta y_a u_y + \gamma(x_a u_y + y_a u_x) - \delta_3 \end{aligned}$$

soit :

$$\begin{aligned} \lambda' = \frac{1}{\det\mathcal{M}}&[(y_b - y_c)z_A + (y_c - y_a)z_B + (y_a - y_b)z_C \\ &+ (y_b - y_c)[f(u_x, u_y) + 2\alpha x_a u_x + 2\beta y_a u_y + \gamma(x_a u_y + y_a u_x) - \delta_3] \\ &+ (y_c - y_a)[f(u_x, u_y) + 2\alpha x_b u_x + 2\beta y_b u_y + \gamma(x_b u_y + y_b u_x) - \delta_3] \\ &+ (y_a - y_b)[f(u_x, u_y) + 2\alpha x_c u_x + 2\beta y_c u_y + \gamma(x_c u_y + y_c u_x) - \delta_3]] \\ = \lambda + \tfrac{1}{\det\mathcal{M}}&[2\alpha u_x \det\mathcal{M} + \gamma u_y \det\mathcal{M}] \\ = \lambda + 2\alpha u_x &+ \gamma u_y \end{aligned}$$

De manière analogue :

$$\mu' = \mu + 2\beta u_y + \gamma u_x.$$

Enfin, le calcul de ν' donne :

$$\begin{aligned} \det\mathcal{M}\,\nu' = &[x_b y_c - y_b x_c + u_x(y_c - y_b) + u_y(x_b - x_c)]\, z_{A'} \\ &+ [x_c y_a - y_c x_a + u_x(y_a - y_c) + u_y(x_c - x_a)]\, z_{B'} \\ &+ [x_a y_b - y_a x_b + u_x(y_b - y_a) + u_y(x_a - x_b)]\, z_{C'} \end{aligned}$$

soit

$$\begin{aligned}
\det\mathcal{M}\,\nu' = (x_by_c - y_bx_c)(&\boxed{z_A + f(u_x,u_y) - \delta_3} \\
&\cancel{= + 2\alpha u_x x_a + 2\beta u_y y_a + \gamma(u_y x_a + u_x y_a)}\ ) \\
+ (x_cy_a - y_cx_a)(&\boxed{z_B + f(u_x,u_y) - \delta_3} \\
&\cancel{= + 2\alpha u_x x_b + 2\beta u_y y_b + \gamma(u_y x_b + u_x y_b)}\ ) \\
+ (x_ay_b - y_ax_b)(&\boxed{z_C + f(u_x,u_y) - \delta_3} \\
&\cancel{= + 2\alpha u_x x_c + 2\beta u_y y_c + \gamma(u_y x_c + u_x y_c)}\ ) \\
+ (u_x(y_c - y_b) + u_y(x_b - x_c))(&\boxed{\boxed{z_A}} \cancel{+ f(u_x,u_y) = \delta_3} \\
&+ \boxed{2\alpha u_x x_a + 2\beta u_y y_a + \gamma(u_y x_a + u_x y_a)}\ ) \\
+ (u_x(y_a - y_c) + u_y(x_c - x_a))(&\boxed{\boxed{z_B}} \cancel{+ f(u_x,u_y) = \delta_3} \\
&+ \boxed{2\alpha u_x x_b + 2\beta u_y y_b + \gamma(u_y x_b + u_x y_b)}\ ) \\
+ (u_x(y_b - y_a) + u_y(x_a - x_b))(&\boxed{\boxed{z_C}} \cancel{+ f(u_x,u_y) = \delta_3} \\
&+ \boxed{2\alpha u_x x_c + 2\beta u_y y_c + \gamma(u_y x_c + u_x y_c)}\ )
\end{aligned}$$

avec

$$\begin{aligned}
\boxed{} &= \det\mathcal{M}.(\nu + f(u_x,u_y) - \delta_3) \\
\boxed{\boxed{}} &= -\det\mathcal{M}.(2\alpha u_x^2 + 2\beta u_y^2 + 2\gamma u_x u_y) \\
\boxed{\boxed{\boxed{}}} &= -\det\mathcal{M}.(\lambda u_x + \mu u_y)
\end{aligned}$$

soit :

$$\begin{aligned}
\det\mathcal{M}\,.\nu' \;=\; \det\mathcal{M}\,.[&-\lambda u_x - \mu u_y + \nu + f(u_x,u_y) \\
&-(2\alpha u_x^2 + 2\beta u_y^2 + 2\gamma u_x u_y + \delta_3)]
\end{aligned}$$

et, le résultat cherché :

$$\begin{aligned}
\nu' &= -\lambda u_x - \mu u_y + \nu - f(-u_x,-u_y) + \delta_3 \\
&= p_{\Delta abc}(-u_x,-u_y) - f(-u_x,-u_y) + \delta_3 \\
&= \nu - \lambda u_x - \mu u_y + f(u_x,u_y) - 2(\alpha u_x^2 + \beta u_y^2 + \gamma u_x u_y) - \delta_3
\end{aligned}$$

□

théorème 4.1 *Soient a, b et c trois points du plan, et soit f une quadrique définie par une fonction. L'erreur L_p, commise par l'approximation linéaire spatiale de f pour le triangle Δabc, est invariante par translation de a, b et c.*

démonstration :

On note $\vec{u}$ et $\vec{v}$, les vecteurs $\vec{ab}$ et $\vec{ac}$; on se place dans le repère barycentrique d'origine a.
Soit q un point du triangle Δabc ; dans le repère choisi, on a :

$$q = a + \lambda_{\vec{u}}\vec{u} + \lambda_{\vec{v}}\vec{v},$$

où $\lambda_{\vec{u}}$ et $\lambda_{\vec{v}}$ sont 2 réels compris entre 0 et 1. Matriciellement, on peut écrire :

$$\begin{pmatrix} x_q \\ y_q \\ 1 \end{pmatrix} = \begin{pmatrix} x_a & x_{\vec{u}} & x_{\vec{v}} \\ y_a & y_{\vec{u}} & y_{\vec{v}} \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ \lambda_{\vec{u}} \\ \lambda_{\vec{v}} \end{pmatrix} = \mathcal{M}_1 \begin{pmatrix} 1 \\ \lambda_{\vec{u}} \\ \lambda_{\vec{v}} \end{pmatrix}.$$

et

$$\begin{pmatrix} x_q^2 \\ y_q^2 \\ x_q y_q \end{pmatrix} = \mathcal{M}_2 \begin{pmatrix} 1 \\ \lambda_{\vec{u}}^2 \\ \lambda_{\vec{v}}^2 \end{pmatrix} + \mathcal{M}_3 \begin{pmatrix} \lambda_{\vec{u}} \\ \lambda_{\vec{u}} \\ \lambda_{\vec{u}}\lambda_{\vec{v}} \end{pmatrix},$$

avec

$$\begin{aligned} \mathcal{M}_2 &= \begin{pmatrix} x_a^2 & x_{\vec{u}}^2 & x_{\vec{v}}^2 \\ y_a^2 & y_{\vec{u}}^2 & y_{\vec{v}}^2 \\ x_a y_a & x_{\vec{u}} y_{\vec{u}} & x_{\vec{v}} y_{\vec{v}} \end{pmatrix} \\ \mathcal{M}_3 &= \begin{pmatrix} 2x_a x_{\vec{u}} & 2x_a x_{\vec{v}} & 2x_{\vec{u}} x_{\vec{v}} \\ 2y_a y_{\vec{u}} & 2y_a y_{\vec{u}} & 2y_{\vec{u}} y_{\vec{v}} \\ x_a y_{\vec{u}} + y_a x_{\vec{u}} & x_a y_{\vec{v}} + y_a x_{\vec{v}} & x_{\vec{u}} y_{\vec{v}} + y_{\vec{u}} x_{\vec{v}} \end{pmatrix} \end{aligned}$$

On suppose que la quadrique f a pour équation

$$f(x, y) = \alpha x^2 + \beta y^2 + \gamma + \delta_1 x + \delta_2 y + \delta_3,$$

et on pose

$$z = \lambda x + \mu y + \nu,$$

l'équation du plan passant par les relevés de a, b et c sur la surface définie par f.
Soit $e(q) = f(x_q, y_q) - \lambda x_q - \mu y_q - \nu$. On a :

$$\begin{aligned} e(q) &= \begin{pmatrix} \alpha & \beta & \gamma \end{pmatrix} \begin{pmatrix} x_q^2 \\ y_q^2 \\ x_q y_q \end{pmatrix} \\ &\quad + \left[\begin{pmatrix} \delta_1 & \delta_2 & \delta_3 \end{pmatrix} - \begin{pmatrix} \lambda & \mu & \nu \end{pmatrix} \right] \begin{pmatrix} x_q \\ y_q \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} \alpha & \beta & \gamma \end{pmatrix} \mathcal{M}_2 \begin{pmatrix} 1 \\ \lambda_{\vec{u}}^2 \\ \lambda_{\vec{v}}^2 \end{pmatrix} + \begin{pmatrix} \alpha & \beta & \gamma \end{pmatrix} \mathcal{M}_3 \begin{pmatrix} \lambda_{\vec{u}} \\ \lambda_{\vec{v}} \\ \lambda_{\vec{u}}\lambda_{\vec{v}} \end{pmatrix} \\ &\quad + \left[\begin{pmatrix} \delta_1 & \delta_2 & \delta_3 \end{pmatrix} - \begin{pmatrix} \lambda & \mu & \nu \end{pmatrix} \right] \mathcal{M}_1 \begin{pmatrix} 1 \\ \lambda_{\vec{u}} \\ \lambda_{\vec{v}} \end{pmatrix} \end{aligned}$$

Soit, finalement :

$$e(q) = \varepsilon_1 \lambda_{\vec{u}}^2 + \varepsilon_2 \lambda_{\vec{v}}^2 + \varepsilon_3 \lambda_{\vec{u}} \lambda_{\vec{v}} + \varepsilon_4 \lambda_{\vec{u}} + \varepsilon_5 \lambda_{\vec{v}} + \varepsilon_6.$$

ε_1, ε_2 et ε_3 ne dépendent pas de a. De plus, on a

$$e(a) = e(b) = e(a+\vec{u}) = e(c) = e(a+\vec{v}) = 0,$$

soit

$$\left\{ \begin{array}{rcl} \varepsilon_6 & = & 0 \\ \varepsilon_1 + \varepsilon_4 & = & 0 \\ \varepsilon_2 + \varepsilon_5 & = & 0 \end{array} \right.$$

Finalement, $e(q)$ ne dépend pas de a, mais seulement de $\lambda_{\vec{u}}$, $\lambda_{\vec{v}}$, $\vec{u}$ et $\vec{v}$. □

4.2 Les courbes de séparation pour la norme L_2

4.2.1 Expression de l'erreur d'approximation sur un triangle

Les résultats contenus dans cette section proviennent essentiellement d'un chapitre de la thèse de NADLER [Nad85] : ces résultats ont été adaptés à notre étude et les preuves ont été complétées afin de faciliter la lecture du paragraphe.
On supposera que le plan est muni d'un repère d'origine $o = (0,0)$ et que a, b et c sont trois points quelconques.

lemme 4.1 *Soit Δ_0 le triangle unité (de sommets $(0,0)$, $(1,0)$ et $(0,1)$) ; on a alors :*

$$\int_{\Delta_0} x^m y^n \, dx \, dy = \frac{m!\, n!}{(m+n+2)!}.$$

démonstration :

On note $I_{m,n}$ l'intégrale recherchée ; alors,

$$\begin{aligned} I_{m,n} &= \int_0^1 x^m \int_0^{1-x} y^n dy \, dx \\ &= \int_0^1 x^m \frac{(1-x)^{n+1}}{n+1} dx \\ &= \frac{1}{n+1} \int_0^1 x^m (1-x)^{n+1} dx \end{aligned} \tag{4.1}$$

On réalise une intégration par parties dans l'équation (4.1) en prenant une primitive de $(1-x)^{n+1}$ et la dérivée de x^m.

$$\begin{aligned} I_{m,n} &= \frac{1}{n+1} \left[-x^m \frac{(1-x)^{n+2}}{n+2} \right]_0^1 \\ &\quad + \frac{m}{(n+1)(n+2)} \int_0^1 x^{m-1} (1-x)^{n+2} \, dx \\ &= \frac{m}{(n+1)(n+2)} \int_0^1 x^{m-1} (1-x)^{n+2} dx \\ &= \frac{m}{n+1} I_{m-1,n+1} \end{aligned}$$

En itérant m fois ce processus, on aboutit à la formule :

$$\begin{aligned} I_{m,n} &= \frac{m(m-1)\dots 1}{(n+1)(n+2)\dots(n+m)} I_{m-m,n+m} \\ &= \frac{m!}{(n+1)\dots(m+n)} \frac{1}{m+n+1} \int_0^1 1(1-x)^{m+n+1} dx \\ &= \frac{m!}{(n+1)\dots(m+n)(m+n+1)} \frac{1}{m+n+2} \\ &= \frac{m!\,n!}{(m+n+2)!} \end{aligned}$$

□

propriété 4.3 *Soit Δabc un triangle ; on considère la transformation affine qui envoie Δabc sur le triangle unité Δ_0 :*

$$X = \begin{pmatrix} x \\ y \end{pmatrix} = \mathcal{A} \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} + \vec{a} = \mathcal{A}\tilde{X} + \vec{a}, \text{ avec } \vec{a} = \overrightarrow{oa}.$$

Alors, on a la relation :

$$\big(e_2(f, \Delta abc)\big)^2 = 2.Surf(\Delta abc)\Big(e_2(\tilde{f}, \Delta_0)\Big)^2.$$

démonstration :

Soient $\vec{e_1}$, $\vec{e_2}$ et $\vec{e_3}$ les vecteurs définissant les arêtes de Δabc, orientés comme dans la figure 4.1. Alors, la transformation affine qui envoie Δabc sur Δ_0 sera définie par :

$$X = \mathcal{A}\tilde{X} + \vec{a},\ \mathcal{A} = (\vec{e_1}, -\vec{e_3}),$$

avec

$$\vec{e_1} = \begin{pmatrix} x_b - x_a \\ y_b - y_a \end{pmatrix}, -\vec{e_3} = \begin{pmatrix} x_c - x_a \\ y_c - y_a \end{pmatrix}.$$

Par ailleurs, la définition des erreurs d'interpolation linéaire L_2 et de la transformation affine nous donne :

$$\begin{cases} \big(e_2(f, \Delta abc)\big)^2 &= \displaystyle\int_{\Delta abc} (f(X) - p_{\Delta abc}(X))^2\, dX \\ \big(e_2(\tilde{f}, \Delta_0)\big)^2 &= \displaystyle\int_{\Delta_0} (\tilde{f}(\tilde{X}) - \tilde{p}_{\Delta_0}(\tilde{X}))^2\, d\tilde{X} \end{cases},$$

avec

$$\begin{cases} \tilde{f}(\tilde{X}) &= f(X) \\ \tilde{p}_{\Delta_0}(\tilde{X}) &= p_{\Delta abc}(X) \end{cases}.$$

On effectue le changement de variable $X = \mathcal{A}\tilde{X}+\vec{a}$ dans la première intégrale ; comme $\mathcal{A}$ est la matrice d'un changement de variable linéaire en x et y, elle est aussi égale à la Jacobienne de la transformation affine. Par définition d'un changement de variable et étant donné que $dX = dx\,dy$ et $d\tilde{X} = d\tilde{x}\,d\tilde{y}$, on aura $dX = |\det\mathcal{A}|\,d\tilde{X}$.
Soit, en reportant dans la formulation de l'erreur pour le triangle Δabc :

$$\begin{aligned}\big(e_2(f,\Delta abc)\big)^2 &= \int_{\mathcal{A}^{-1}(\Delta abc-\vec{a})} \Big(\tilde{f}(\tilde{X}) - \tilde{p}_{\mathcal{A}^{-1}(\Delta abc-\vec{a})}(\tilde{X})\Big)^2\, |\det\mathcal{A}|\,d\tilde{X}\\ &= \int_{\Delta_0} \Big(\tilde{f}(\tilde{X}) - \tilde{p}_{\Delta_0}(\tilde{X})\Big)^2\, |\det\mathcal{A}|\,d\tilde{X}\\ &= |\det\mathcal{A}|\,\Big(e_2(\tilde{f},\Delta_0)\Big)^2.\end{aligned}$$

Or, d'après la propriété 4.1 :

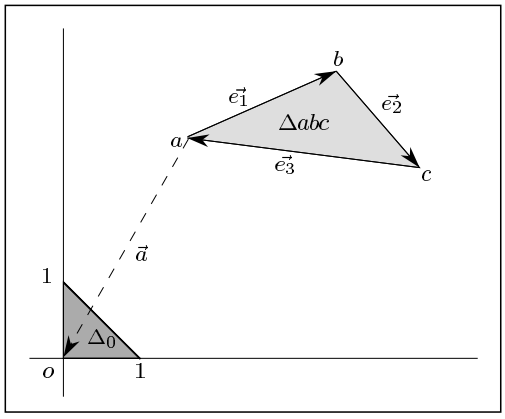
$$\begin{aligned}|\det\mathcal{A}| &= \Big|\begin{vmatrix} x_b - x_a & x_c - x_a \\ y_b - y_a & y_c - y_a \end{vmatrix}\Big|\\ &= 2\,\mathrm{Surf}(\Delta abc).\end{aligned}$$

Soit, finalement :

$$\big(e_2(f,\Delta abc)\big)^2 = 2\,\mathrm{Surf}(\Delta abc)\,\big(e_2(\tilde{f},\Delta_0)\big)^2.$$

□

Figure 4.1 – transformation affine du triangle Δabc en Δ_0.



remarque :
de la même façon, on pourrait montrer que

$$\big(e_p(f,\Delta abc)\big)^p = 2\,\mathrm{Surf}(\Delta abc)\,\big(e_p(\tilde{f},\Delta_0)\big)^p$$

et donc

$$e_p(f, \Delta abc) = \sqrt[p]{2\,\mathrm{Surf}(\Delta abc)}\, e_p(\tilde{f}, \Delta_0).$$

théorème 4.2 *Le carré de l'erreur L_2 commise par l'approximation linéaire d'une quadrique f d'équation $f(x,y) = \alpha x^2 + \beta y^2 + \gamma xy$ sur un triangle Δabc (approximation calculée aux sommets du triangle) est de la forme :*

$$\big(e_2(f, \Delta abc)\big)^2 = \mathit{Surf}(\Delta abc)\, {}^t\vec{v}\,\mathcal{M}\vec{v},$$

avec

$$\mathcal{M} = \frac{1}{180}\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix},$$

et, si on note $\vec{e_1}$, $\vec{e_2}$ et $\vec{e_3}$ les arêtes du triangle (orientées comme sur la figure 4.1),

$$\vec{v} = \begin{pmatrix} \frac{1}{2}\,{}^t\vec{e_1}\,\mathcal{H}\,\vec{e_1} \\ \frac{1}{2}\,{}^t\vec{e_2}\,\mathcal{H}\,\vec{e_2} \\ \frac{1}{2}\,{}^t\vec{e_3}\,\mathcal{H}\,\vec{e_3} \end{pmatrix} = \begin{pmatrix} f(\vec{e_1}) \\ f(\vec{e_2}) \\ f(\vec{e_3}) \end{pmatrix},$$

où $\mathcal{H}$ représente la matrice hessienne (matrice des dérivées secondes) de la fonction f :

$$\mathcal{H} = \begin{pmatrix} 2\alpha & \gamma \\ \gamma & 2\beta \end{pmatrix}.$$

démonstration :

D'après les notations prises sur la figure 4.1, la transformation affine de Δabc en Δ_0 est donnée par :

$$X = \mathcal{A}\tilde{X} + \vec{a}, \quad \mathcal{A} = (\vec{e_1}, -\vec{e_3}).$$

La propriété 4.3 prouve par ailleurs que :

$$\big(e_2(f, \Delta abc)\big)^2 = 2\,\mathrm{Surf}(\Delta abc)\,\big(e_2(\tilde{f}, \Delta_0)\big)^2,$$

avec

$$f(X) = \frac{1}{2}\,{}^t X \mathcal{H} X,$$

et

$$\begin{aligned} \tilde{f}(\tilde{X}) &= \tfrac{1}{2}\,{}^t(\mathcal{A}\tilde{X})\,\mathcal{H}\,\mathcal{A}\tilde{X} + \frac{1}{2}\,{}^t(\mathcal{A}\tilde{X})\,\mathcal{H}\vec{a} + \frac{1}{2}\,{}^t\vec{a}\mathcal{H}\mathcal{A}\tilde{X} + \tfrac{1}{2}\,{}^t\vec{a}\mathcal{H}\vec{a} \\ &= \tfrac{1}{2}\,{}^t\tilde{X}\,{}^t\mathcal{A}\mathcal{H}\mathcal{A}\,\tilde{X} + {}^t\vec{a}\mathcal{H}\mathcal{A}\tilde{X} + \tfrac{1}{2}\,{}^t\vec{a}\mathcal{H}\vec{a}. \end{aligned} \tag{4.2}$$

Par définition de $\mathcal{A}$ ($\mathcal{A} = (\vec{e_1}, -\vec{e_3})$), on a :

$$ {}^t\mathcal{A}\mathcal{H}\mathcal{A} = \begin{pmatrix} {}^t\vec{e_1}\mathcal{H}\vec{e_1} & -{}^t\vec{e_1}\mathcal{H}\vec{e_3} \\ -{}^t\vec{e_1}\mathcal{H}\vec{e_3} & {}^t\vec{e_3}\mathcal{H}\vec{e_3} \end{pmatrix}. $$

D'où, en reportant dans l'équation (4.2) :

$$ \tilde{f}(\tilde{X}) = \tfrac{1}{2}\,{}^t\vec{e_1}\mathcal{H}\vec{e_1}\,\tilde{x}^2 - {}^t\vec{e_1}\mathcal{H}\vec{e_3}\,\tilde{x}\tilde{y} + \tfrac{1}{2}\,{}^t\vec{e_3}\mathcal{H}\vec{e_3}\,\tilde{y}^2 + {}^t\vec{a}\mathcal{H}\mathcal{A}\tilde{X} + \tfrac{1}{2}\,{}^t\vec{a}\mathcal{H}\vec{a}. \quad (4.3) $$

Les deux derniers termes étant respectivement linéaire et constant en les variables $\tilde{x}$ et $\tilde{y}$, ils seront égaux à leur approximation. En revanche, on s'aperçoit que l'on a besoin des approximations linéaires des fonctions $\tilde{x}^2$, $\tilde{x}\tilde{y}$ et $\tilde{y}^2$ sur le triangle Δ_0 : pour les calculer, il suffit de chercher l'équation des plans qui passent par les relevés des sommets de Δ_0 sur les surfaces $\tilde{z} = \tilde{x}^2$, $\tilde{z} = \tilde{x}\tilde{y}$ et $\tilde{z} = \tilde{y}^2$. On se rend alors compte que ces approximations sont respectivement les applications linéaires $\tilde{x}$, 0 et $\tilde{y}$.
En intégrant la formulation obtenue pour $\tilde{f}(\tilde{X})$ dans (4.3) à l'expression de l'erreur pour Δ_0, on aboutit à :

$$ \begin{aligned} \big(e_2(\tilde{f}, \Delta_0)\big)^2 &= \int_{\Delta_0} \big(({}^t\vec{e_1}\mathcal{H}\vec{e_1})\tfrac{1}{2}(\tilde{x}^2 - \tilde{x}) - ({}^t\vec{e_1}\mathcal{H}\vec{e_3})(\tilde{x}\tilde{y}) \\ &\qquad + ({}^t\vec{e_3}\mathcal{H}\vec{e_3})\tfrac{1}{2}(\tilde{y}^2 - \tilde{y}) \big)^2 \, d\tilde{x}\, d\tilde{y} \\ &= \int_{\Delta_0} \big(({}^t\vec{e_1}\mathcal{H}\vec{e_1})\,\omega_1(\tilde{x}, \tilde{y}) - ({}^t\vec{e_1}\mathcal{H}\vec{e_3})\,\omega_2(\tilde{x}, \tilde{y}) \\ &\qquad + ({}^t\vec{e_3}\mathcal{H}\vec{e_3})\,\omega_3(\tilde{x}, \tilde{y}) \big)^2 \, d\tilde{x}\, d\tilde{y} \\ &= {}^t\tilde{\vec{v}}\,\tilde{\mathcal{P}}\,\tilde{\vec{v}}, \end{aligned} \quad (4.4) $$

avec

$$ \tilde{\vec{v}} = \begin{pmatrix} {}^t\vec{e_1}\mathcal{H}\vec{e_1} \\ -{}^t\vec{e_1}\mathcal{H}\vec{e_3} \\ {}^t\vec{e_3}\mathcal{H}\vec{e_3} \end{pmatrix} \text{ et } \tilde{\mathcal{P}} = \begin{pmatrix} \langle\omega_1,\omega_1\rangle & \langle\omega_1,\omega_2\rangle & \langle\omega_1,\omega_3\rangle \\ \langle\omega_1,\omega_2\rangle & \langle\omega_2,\omega_2\rangle & \langle\omega_2,\omega_3\rangle \\ \langle\omega_1,\omega_3\rangle & \langle\omega_2,\omega_3\rangle & \langle\omega_3,\omega_3\rangle \end{pmatrix}, $$

où

$$ \langle\omega_i,\omega_j\rangle = \int_{\Delta_0} \omega_i(x,y)\,\omega_j(x,y)\,dx\,dy. $$

On constate ainsi que les coefficients de la matrice $\tilde{\mathcal{P}}$ peuvent être calculés en utilisant le résultat du lemme 4.1 ; on trouve :

$$ \tilde{\mathcal{P}} = \frac{1}{1440} \begin{pmatrix} 6 & -6 & 5 \\ -6 & 8 & -6 \\ 5 & -6 & 6 \end{pmatrix} $$

Par ailleurs, comme $\vec{e_2} = \vec{e_1} + \vec{e_3}$, on aura :

$$\begin{aligned} {}^t\vec{e_2}\mathcal{H}\vec{e_2} &= {}^t(\vec{e_1} + \vec{e_3})\mathcal{H}(\vec{e_1} + \vec{e_3}) \\ &= {}^t\vec{e_1}\mathcal{H}\vec{e_1} + 2\,{}^t\vec{e_1}\mathcal{H}\vec{e_3} + {}^t\vec{e_3}\mathcal{H}\vec{e_3} \end{aligned}$$

soit

$$-{}^t\vec{e_1}\mathcal{H}\vec{e_3} = \tfrac{1}{2}\,{}^t\vec{e_1}\mathcal{H}\vec{e_1} - \tfrac{1}{2}\,{}^t\vec{e_2}\mathcal{H}\vec{e_2} + \tfrac{1}{2}\,{}^t\vec{e_3}\mathcal{H}\vec{e_3},$$

et, finalement :

$$\tilde{\vec{v}} = \begin{pmatrix} 2 & 0 & 0 \\ 1 & -1 & 1 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} \frac{1}{2}\,{}^t\vec{e_1}\,\mathcal{H}\,\vec{e_1} \\ \frac{1}{2}\,{}^t\vec{e_2}\,\mathcal{H}\,\vec{e_2} \\ \frac{1}{2}\,{}^t\vec{e_3}\,\mathcal{H}\,\vec{e_3} \end{pmatrix} = \mathcal{Q}\,\vec{v}.$$

En reportant dans (4.4), on arrive à :

$$\big(e_2(\tilde{f}, \Delta_0)\big)^2 = {}^t\tilde{\vec{v}}\,\tilde{\mathcal{P}}\,\tilde{\vec{v}} = {}^t\vec{v}\,{}^t\mathcal{Q}\tilde{\mathcal{P}}\mathcal{Q}\,\vec{v}.$$

Or,

$${}^t\mathcal{Q}\tilde{\mathcal{P}}\mathcal{Q} = \frac{1}{360}\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix},$$

ce qui implique

$$\begin{aligned} \big(e_2(f, \Delta abc)\big)^2 &= 2\,\mathrm{Surf}(\Delta abc)\left(e_2(\tilde{f}, \Delta_0)\right)^2 \\ &= 2\,\mathrm{Surf}(\Delta abc)\,{}^t\vec{v}\frac{1}{360}\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}\vec{v} \\ &= \mathrm{Surf}(\Delta abc)\,{}^t\vec{v}\,\mathcal{M}\,\vec{v}. \end{aligned}$$

□

corollaire 4.1 *L'erreur L_2 commise par l'approximation linéaire d'une quadrique sur un triangle Δabc est invariante par translation dans le plan des sommets a, b et c.*

4.2.2 Courbes de séparation d'une quadrique

La simplicité de l'équation démontrée au théorème 4.2 permet de pouvoir généraliser les résultats de l'observation 3.7.

propriété 4.4 *Soit f une quadrique d'équation $z = f(x, y) = \alpha x^2 + \beta y^2 + \gamma$; soit Δabc un triangle de sommets $a = (x_a, y_a)$, $b = (x_b, y_b)$ et $c = (x_c, y_c)$, dont les relevés respectifs sur la surface représentative de f sont A, B et C. Alors la courbe*

de séparation de Δabc, pour l'erreur L_2 d'approximation locale par le plan ABC de la surface définie par f, consiste en deux courbes de degré 2 :

- *la courbe donnée par le système :*

$$\begin{cases} z = f(x,y) \\ z = \lambda\, x + \mu\, y + \nu \quad (\mathcal{P}) \\ A \in \mathcal{P},\, B \in \mathcal{P},\, C \in \mathcal{P} \end{cases}$$

- *la courbe d'équation :*

$$f\left(x - \frac{x_a + x_b + x_c}{3}, y - \frac{y_a + y_b + y_c}{3}\right) = 4\overline{HG}$$

avec

$$G = \begin{pmatrix} \dfrac{x_a + x_b + x_c}{3} \\ \dfrac{y_a + y_b + y_c}{3} \\ f(x_G, y_G) \end{pmatrix}, \quad H = \begin{pmatrix} \dfrac{x_a + x_b + x_c}{3} \\ \dfrac{y_a + y_b + y_c}{3} \\ \dfrac{z_A + z_B + z_C}{3} \end{pmatrix}$$

démonstration :

soit $d = (x, y)$ un sommet variable que l'on suppose être dans la zone d'influence de a (même définition qu'au chapitre 3 ; si on note $\epsilon(\Delta abc)$, $\epsilon(\Delta bcd)$, $\epsilon(\Delta abd)$ et $\epsilon(\Delta acd)$ les carrés des erreurs sur chaque triangle possible, les 2 triangulations du quadrilatère convexe $\square abcd$ définissent des erreurs de carrés respectifs $\epsilon(\Delta abc) + \epsilon(\Delta bcd)$ et $\epsilon(\Delta abd) + \epsilon(\Delta acd)$.
Alors, d sera sur la courbe de séparation du triangle Δabc si et seulement si :

$$\xi(\Delta abc, d) = [\epsilon(\Delta abc) + \epsilon(\Delta bcd)] - [\epsilon(\Delta abd) + \epsilon(\Delta acd)] = 0.$$

Les calculs des ϵ et de ξ, bien qu'un peu longs, se factorisent relativement bien et permettent de déduire le résultat de la propriété. □

Chapitre 5

Résultats numériques

Après avoir trouvé (au moins pour la norme L_2) les courbes de séparation d'un triangle Δ qui permettent de savoir de manière rapide et exacte si Δ doit être conservé dans une triangulation optimale, il est bien sûr intéressant de savoir si ces courbes vont nous aider à construire des triangulations rendant une erreur d'approximation du paraboloïde hyperbolique très inférieure à celle obtenue avec d'autres triangulations, en particulier celle de Delaunay.

Comme on se situe dans un cadre de recherche d'une optimalité locale, les tests qui vont suivre utiliseront un certain nombre de méthodes aux résultats assez différents. Les buts seront ici de construire des algorithmes non pas efficaces en temps de calcul, mais plutôt suffisamment modulables pour ne pas nécessiter une réécriture complète lors d'un changement de méthode, et de comparer leurs résultats.

Le chapitre précédent a prouvé que le critère d'optimisation de la norme L_2 s'étendait à toute la famille des quadriques définies par des fonctions ; mais, comme on sait déjà calculer les triangulations globalement optimales de telles quadriques convexes [Mel93], seul le cas des paraboloïdes hyperboliques est intéressant. De plus, nous nous restreindrons au cas du paraboloïde unité, dans la mesure où si les résultats obtenus sont bons, ils devraient être d'autant meilleurs pour les quadriques à coefficients en x^2 et y^2 très différents (puisqu'à ce moment-là, les surfaces présenteront nettement une direction privilégiée qui rendra mauvaise l'utilisation de la triangulation de Delaunay). Par ailleurs, la procédure de calcul de l'erreur L_1 n'a été réalisée que pour cette unique surface (bien que sa généralisation aux autres quadriques non convexes ne devrait pas poser trop de problèmes).

Ce chapitre contient un nombre important de tableaux de résultats de tests ; afin d'en faciliter la lecture, les points essentiels sont repris sous forme de courbes.

5.1 Algorithmes

Le problème fixé au départ est directement lié aux triangulations dépendantes des données ; il ne faudra donc pas s'étonner de trouver ici de fortes liaisons avec les articles de DYN, LEVIN et RIPPA [DLR90a, DLR90b].

5.1.1 Types d'algorithmes

5.1.1.1 Algorithme de swaps

La grande majorité des algorithmes qui désirent aboutir à des triangulations localement optimales utilise la procédure d'optimisation locale suggérée par LAWSON [Law77], qui fonctionne en échangeant les diagonales des quadrilatères convexes, présents dans une triangulation de départ, susceptibles d'améliorer le critère choisi. On suppose que l'on a un ensemble $\mathcal{S}$ de n points $s_i = (x_i, y_i)$ échantillonnant une surface d'équation $z = f(x, y)$; ces sommets admettent ainsi des relevés respectifs sur la surface qui sont les points $(x_i, y_i, f(x_i, y_i))$. L'algorithme est alors :

- **initialisation :** soit $\mathcal{T}$ une triangulation quelconque de l'enveloppe convexe de $\mathcal{S}$

- **tant qu'** il existe une arête intérieure ς qui n'est pas localement optimale **faire**
  on **échange** les diagonales du quadrilatère strictement convexe de diagonale ς

  fin tant que

On peut facilement prouver qu'un tel algorithme s'arrête toujours, puisque le nombre de triangulations possibles d'un ensemble de points est fini, et qu'à chaque swap, on optimise un peu plus le critère choisi ; chaque amélioration locale impliquant une amélioration globale de la triangulation, il n'est pas possible de boucler (c'est-à-dire de retomber sur une triangulation déjà obtenue avant d'effectuer certains échanges).

5.1.1.2 Algorithme incrémental localement optimal

Une approche un peu différente a aussi été testée ; au lieu de partir d'une triangulation spéciale incluant tous les points de $\mathcal{S}$, on introduit les sommets les uns après les autres et, après chaque nouvelle insertion, on optimise en effectuant des échanges de diagonales. Cela revient en fait à appliquer la procédure d'optimisation locale de LAWSON à chaque nouveau point, en prenant pour triangulation de départ l'optimale précédente. L'algorithme devient :

- **initialisation :** soit $\mathcal{T}$ la triangulation incluant les 3 premiers points de $\mathcal{S}$

- **pour** chaque point s de $\mathcal{S}$
  on **insère** s dans $\mathcal{T}$ en divisant en 3 le triangle qui l'inclut, ou en remettant à jour l'enveloppe convexe

tant qu' il existe une arête intérieure ς qui n'est pas localement optimale
faire
on **échange** les diagonales du quadrilatère strictement convexe de diagonale ς
fin tant que

fin pour

Dorénavant, cette méthode sera désignée par l'abréviation *TOI*, pour *Triangulation Optimale Initiale*.

5.1.1.3 Complexité des algorithmes

Le but de ce chapitre n'étant pas d'exhiber des algorithmes efficaces en temps de calcul, aucune recherche particulière n'a été menée pour accélérer leur exécution. Ce sont donc des implantations "grossières" qui ont été réalisées, dont le but était de pouvoir tester facilement le plus de méthodes possibles.

La procédure d'optimisation locale n'est qu'un algorithme de swaps avec triangulation initiale : il faut donc $O(n \log n)$ pour obtenir la triangulation initiale, et on sait que le nombre d'échanges final peut être, dans le cas le pire, de $O(n^3)$, complexité qui représente le nombre total de triangles vides possibles. La complexité temporelle de l'algorithme est ainsi de $O(n^3)$.

La seconde approche nécessite au plus $O(i^3)$ échanges d'arêtes lors de l'insertion du i-ème sommet. Une complexité surestimée est donc de $O(n^4)$ (puisque, lors de l'insertion d'un nouveau sommet, seules quelques arêtes créées seront susceptibles d'être échangées et d'impliquer de nouveaux échanges, mais certainement beaucoup moins qu'un nombre cubique).

Nous verrons cependant, au paragraphe 5.2.4 que ces complexités théoriques sont assez éloignées des résultats observés en pratique.

5.1.2 Paramètres des algorithmes

Les 2 algorithmes précédemment décrits ne sont pas complètement déterminés, puisqu'il reste à choisir 2 paramètres : l'ordre des échanges d'une part, et soit la triangulation initiale (premier algorithme), soit l'ordre d'insertion des points (deuxième algorithme) d'autre part.

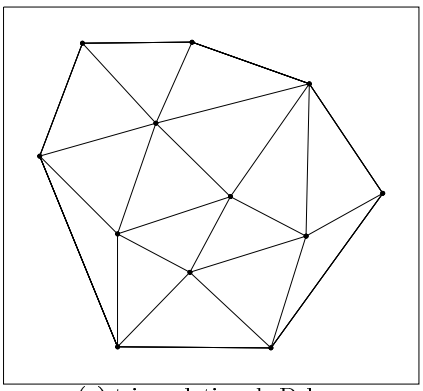
5.1.2.1 Échanges de diagonales

L'ordre dans lequel les swaps vont être effectués est d'une grande importance, surtout qu'il n'y aura sans doute pas unicité de la triangulation localement optimale. Parmi les stratégies proposées dans [DLR90a], une seule a été retenue : commencer par échanger les arêtes qui vont faire disparaître les plus mauvais triangles (technique dite de la *réduction maximale*). Ce choix a été fait non seulement parce qu'il semblait relativement logique, mais aussi parce qu'il a montré toute son efficacité dans les tests proposés dans [DLR90a].

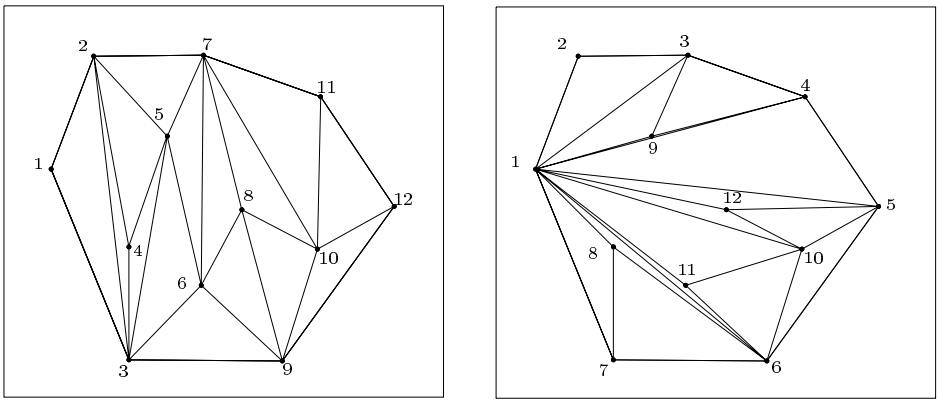
Au cours du chapitre 3, l'équation des courbes de séparation a été établie ; la fonction ξ déduite de cette équation permet de donner une réponse oui/non quant à

l'opportunité d'effectuer un échange de diagonales, grâce au signe qu'elle renvoie. De plus, il a été montré qu'à une constante près, cette fonction est égale à la différence entre les erreurs L_2 des deux triangulations possibles de 4 sommets. Ainsi, si le résultat produit par ξ pour 4 points a, b, c, d est positif, cela signifie que l'échange de la diagonale $[bc]$ doit être effectué ; et, plus ce résultat est grand, meilleure la triangulation sera. Cette fonction, directement tirée de l'équation des courbes de séparation permet ainsi de trier les arêtes suivant la stratégie de la réduction maximale, de manière simple et rapide.

Figure 5.1 – triangulations initiales testées.



(a) triangulation de Delaunay



(b) triangulations étoilées pour des ordres d'insertion différents

5.1.2.2 Paramètres de création

Selon l'algorithme adopté, ces paramètres sont différents :

- pour l'algorithme de type Lawson, c'est la triangulation initiale choisie qui est surtout importante,

- pour l'algorithme incrémental, c'est l'ordre dans lequel on va insérer les points qui compte.

Triangulations initiales

Seules deux triangulations initiales ont été testées (figure 5.1) :

- la triangulation de Delaunay, qui va également nous servir de point de repère pour déterminer si les résultats obtenus sont bons,
- la triangulation étoilée, obtenue en insérant les points les uns après les autres et en n'effectuant pas de swaps (l'ordre d'insertion influence alors la triangulation).

Ces deux triangulations sont très différentes dans la mesure où une va donner des triangles très "réguliers", et l'autre des triangles aux formes très variées (formes liées à l'ordre d'insertion des sommets).

Ordre d'insertion des points

Cet ordre est d'autant plus important qu'en plus d'influencer la forme de la triangulation étoilée pour l'algorithme de type LAWSON, il va jouer un rôle dans l'ordre dans lequel les échanges de diagonales seront effectués lors des méthodes de type TOI.

Il a ainsi été décidé de tester trois heuristiques :

- insérer les sommets en fonction de leur éloignement (croissant) au point le plus en bas à gauche,
- insérer par abscisses (puis ordonnées en cas d'égalité) croissantes,
- insérer sans ordre précis (seulement celui de la création des sommets, qui est presque aléatoire).

C'est cette troisième heuristique qui va permettre de quantifier l'influence de l'ordre d'insertion sur le résultat de chaque méthode.

5.1.3 Ensembles de points

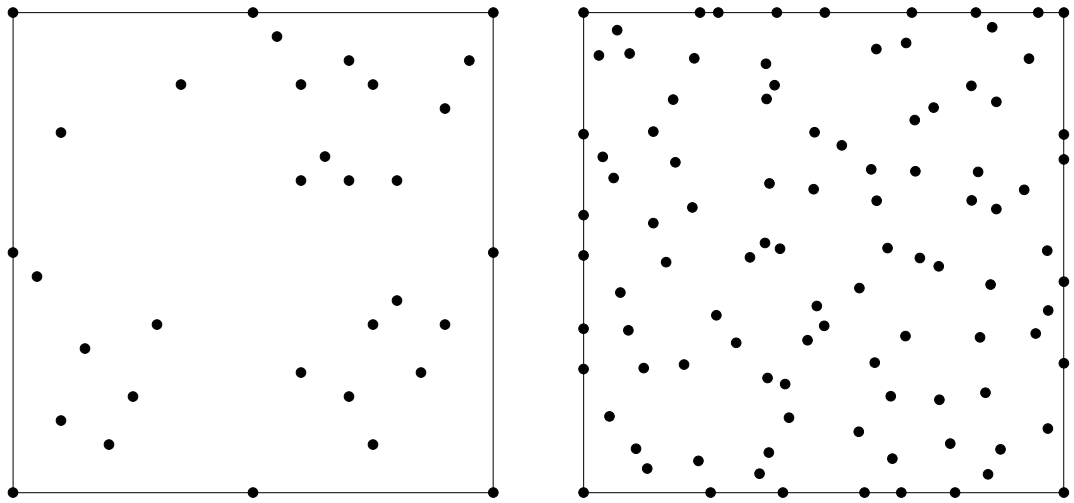
5.1.3.1 Ensembles de Franke

FRANKE a suggéré [Fra82] que deux ensembles bien précis de respectivement 33 et 100 points pouvaient être suffisants pour évaluer les performances d'une triangulation, approximation linéaire d'une surface donnée par un ensemble de points. Les 100 points sont plus ou moins distribués uniformément sur le carré unité, alors que les 33 points sont répartis avec des variations plus importantes sur leur densité. Ces ensembles sont représentés sur la figure 5.2.

5.1.3.2 Ensembles aléatoires

La surface à approcher (le paraboloïde hyperbolique) étant tout de même très différente des surfaces habituellement testées par les numériciens (car elle ne présente ni direction privilégiée, ni changements brutaux d'orientation ou de pente), d'autres

Figure 5.2 – les ensembles proposés par FRANKE.



ensembles de points ont été essayés. Ceux-ci sont générés plus ou moins aléatoirement suivant le processus suivant :

- soit n les nombre total de points de l'ensemble, et n_Ω le nombre de ceux qui sont sur l'enveloppe convexe.
- on décide que l'enveloppe convexe devra toujours être le carré unité, et que les n_Ω sommets seront repartis de façon régulière sur le bord, et créés les premiers.
- on engendre les $n - n_\Omega$ points manquant de manière aléatoire, selon un processus de distribution uniforme sur le carré.

Le choix d'imposer une enveloppe convexe permet de comparer de façon moins ambiguë les différents résultats à n fixé, mais aussi pour des valeurs variées de n, puisque le domaine total d'approximation reste le même.

Les valeurs de n testées ont été 15, 33, 50, 100, 250 et 500, pour des n_Ω variables compris entre 4 et 60. Bien entendu, il peut arriver qu'un sommet aléatoire soit ajouté à l'enveloppe convexe mais sans modifier sa frontière.

5.2 Résultats

‘

5.2.1 Les différentes méthodes

Comme l'a montré la section précédente, il est possible de construire de nombreuses triangulations localement optimales, en modifiant l'algorithme général, la triangulation initiale, l'ordre d'insertion des sommets ou encore l'heuristique sur l'ordre des échanges.

Concernant les critères de décision pour effectuer un swap, la partie théorique montre qu'il est facile d'obtenir une optimisation locale de la norme L_2 par utilisation des courbes de séparation d'un triangle. Mais, on a vu aussi une méthode pour calculer de façon "exacte" (aux imprécisions de la machine près) l'erreur L_1 ; cela nous permet donc de pouvoir réaliser des échanges qui vont localement réduire l'erreur L_1 d'une triangulation. Ce sont ainsi 2 critères qui seront testés, bien que nous n'ayons aucun fond théorique sur l'un d'eux, puisque l'on peut espérer que l'optimalité locale de l'un des critères entrainera dans certains cas l'optimalité de l'autre critère.

5.2.1.1 Premiers tests

Dans un premier temps, comme les choix proposés engendraient de nombreuses méthodes, il a été décidé de toutes les essayer sur un nombre conséquent d'ensembles de points définis, c'est-à-dire 5000 ensembles de 33 points dont 12 sur les bords du carré unité, ainsi que sur les 2 ensembles de FRANKE. Les résultats obtenus permettront alors de décider si l'on teste à nouveau toutes les méthodes pour d'autres ensembles de points, ou si l'on n'en conserve que quelques unes.

Le tableau 5.1 attribue un numéro à chaque méthode :

- la méthode *Del.* correspond toujours à la triangulation de Delaunay.

- la méthode *Tlo* correspond toujours à la meilleure triangulation localement optimale obtenue parmi toutes celles testées (meilleure au sens de la minimisation de l'erreur L_1 ou L_2 suivant les cas).

- la colonne "initiale" donne la triangulation initiale choisie avant échanges ; pour les algorithmes de type *TOI*, on n'a pas besoin d'une telle indication ; la mention "étoilée" signifie que l'on démarre d'une triangulation où aucun échange n'a été réalisé (on s'est juste contenté d'insérer les sommets suivant un certain ordre).

- la colonne "critère" donne le critère d'optimisation suivant lequel les échan-ges seront réalisés.

- la colonne "insertion" indique dans quel ordre les sommets seront insérés :
  - "croissant" signifie que les points de plus petite abscisse (puis ordonnée) seront insérés les premiers.
  - "coin" signifie que ce sont les sommets les plus proches du coin en bas à gauche qui seront d'abord utilisés.
  - "aléatoire" signifie que les sommets seront insérés selon l'ordre de leur création (à noter tout de même que, dans tous les ensembles de points "aléatoires", tous les sommets de l'enveloppe convexe sont toujours les premiers créés, ce qui implique que l'ordre d'insertion n'est aléatoire que pour les points intérieurs).

Tableau 5.1 – les méthodes initialement testées.

méthode	algorithme	initiale	critère	insertion
Del.	Delaunay	Delaunay	cercle vide	
1	TOI		L_2	aléatoire
2	TOI		L_2	croissant
3	TOI		L_2	coin
4	TOI		L_1	aléatoire
5	TOI		L_1	croissant
6	TOI		L_1	coin
7	Lawson	étoilée	L_2	aléatoire
8	Lawson	étoilée	L_2	croissant
9	Lawson	étoilée	L_2	coin
10	Lawson	étoilée	L_1	aléatoire
11	Lawson	étoilée	L_1	croissant
12	Lawson	étoilée	L_1	coin
13	Lawson	Delaunay	L_2	
14	Lawson	Delaunay	L_1	
Tlo	meilleur	meilleure	meilleur	meilleure

5.2.1.2 Ensembles aléatoires

Les tableaux 5.2 et 5.3 reproduisent les résultats obtenus pour les normes L_2 et L_1, accompagnés de certaines statistiques :

- l'"erreur" indiquée est l'erreur moyenne sur les 5000 tests.
- l'"écart" représente l'écart type de l'erreur moyenne pour l'ensemble des tests.
- la colonne "meil." donne le nombre de fois où la méthode en question a rendu l'erreur la plus faible.
- le "rapport" est celui entre la meilleure erreur obtenue par une des triangulations localement optimales testées et l'erreur pour la méthode en cours.
- les pourcentages d'"améliorations" sont calculés entre l'erreur pour Delaunay et l'erreur pour la méthode :

$$\text{amélioration} = \frac{\text{erreur(Delaunay)} - \text{erreur(méthode)}}{\text{erreur(Delaunay)}}.$$

  Sont inclus les valeurs moyennes de ces améliorations, ainsi que la plus mauvaise (très souvent négative) et la meilleure valeur.

Quelques remarques s'imposent :

- il parait clair que la méthode qui donne les meilleurs résultats généraux est celle qui utilise l'algorithme du type LAWSON avec Delaunay pour triangulation initiale ; non seulement, elle implique que l'erreur sera toujours plus faible que

Tableau 5.2 – résultats statistiques pour l'erreur L_2 d'ensembles de 33 points dont 12 sur l'enveloppe convexe.

méthode	erreur L_2	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.013758	0.0029	0	0.8140	0.00	0.00	0.00
1	0.013592	0.0025	84	0.8239	1.21	-87.65	66.30
2	0.012887	0.0029	268	0.8690	6.33	-73.26	51.93
3	0.012854	0.0025	228	0.8712	6.57	-70.27	54.51
4	0.013593	0.0028	66	0.8239	1.20	-94.78	66.06
5	0.012868	0.0029	194	0.8703	6.47	-83.19	52.62
6	0.012841	0.0025	162	0.8721	6.66	-70.27	55.48
7	0.013209	0.0024	273	0.8478	3.99	-68.87	66.32
8	0.011858	0.0022	1321	0.9444	13.81	-41.41	54.80
9	0.012439	0.0023	738	0.9003	9.59	-83.52	62.39
10	0.013276	0.0025	172	0.8435	3.50	-88.77	66.31
11	0.012018	0.0023	618	0.9319	12.65	-80.77	53.08
12	0.012503	0.0023	424	0.8957	9.12	-83.56	62.39
13	0.011719	0.0020	1701	0.9556	14.82	0.25	55.08
14	0.011781	0.0020	767	0.9506	14.37	-1.81	55.00
Tlo	0.011199	0.0015	5000	1.0000	18.60	0.93	66.32

Tableau 5.3 – résultats statistiques pour l'erreur L_1 d'ensembles de 33 points dont 12 sur l'enveloppe convexe.

méthode	erreur L_1	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.010228	0.0018	0	0.7713	0.00	0.00	0.00
1	0.010002	0.0015	45	0.7887	2.20	-84.89	61.01
2	0.008981	0.0018	150	0.8783	12.18	-68.76	47.63
3	0.009099	0.0016	95	0.8669	11.03	-68.30	51.27
4	0.009286	0.0016	186	0.8495	9.21	-84.58	60.78
5	0.008861	0.0017	406	0.8902	13.36	-68.24	52.11
6	0.009046	0.0015	216	0.8720	11.55	-62.07	51.27
7	0.009672	0.0014	82	0.8156	5.43	-85.21	61.03
8	0.008437	0.0014	587	0.9350	17.51	-37.42	54.57
9	0.008813	0.0014	310	0.8951	13.83	-54.94	56.85
10	0.009672	0.0014	121	0.8156	5.43	-84.56	61.14
11	0.008403	0.0015	1112	0.9387	17.84	-47.23	51.46
12	0.008718	0.0015	683	0.9048	14.76	-55.13	56.85
13	0.008278	0.0013	655	0.9529	19.06	-5.46	53.53
14	0.008198	0.0013	1769	0.9623	19.85	1.26	53.53
Tlo	0.007889	0.0011	5000	1.0000	22.87	1.64	61.14

celle de Delaunay, mais surtout c'est elle qui apparait comme la meilleure triangulation localement optimale dans plus d'un tiers des exemples testés.

– d'un autre côté, cela montre aussi que ce n'est pas à chaque fois la même méthode qui donne les meilleurs résultats ; ce phénomène, déjà observé par les auteurs d'articles sur les triangulation dépendantes des données, est une nouvelle fois mis en valeur, puisqu'en valeur moyenne, les algorithmes du type LAWSON à triangulation étoilée, avec un ordre d'insertion convenable, ne sont pas très loin.

– en revanche, les résultats des algorithmes du genre *TOI* ne sont pas bons, contrairement à ce qui était attendu. Une explication peut être qu'en utilisant une triangulation déjà localement optimale, on reste bloqué autour d'une position stable, qui interdit d'effectuer certains échanges. Cette observation sera d'ailleurs confirmée au paragraphe 5.2.4 où l'on verra que le nombre de swaps totaux effectués par les méthodes *TOI* n'est pas beaucoup supérieur à celui réalisé par les méthodes du type LAWSON.

5.2.1.3 Ensembles de *Franke*

Le tableau 5.4 contient les résultats obtenus en testant toutes les méthodes sur les 2 ensembles de FRANKE. On peut, à nouveau, constater le bon comportement des algorithmes de LAWSON avec Delaunay pour triangulation initiale, bien que ce ne soit pas eux qui donnent la meilleure erreur à chaque fois. En revanche, notamment pour l'ensemble à 100 points, les algorithmes de type *TOI* sont pris en défaut, et aboutissent très souvent à une augmentation de l'erreur par rapport à Delaunay.

Tableau 5.4 – erreurs L_1 et L_2 des ensembles de FRANKE pour toutes les méthodes.

méthode	ensemble à 33 points				ensemble à 100 points			
	erreur L_1	amé.	erreur L_2	amé.	erreur L_1	amé.	erreur L_2	amé.
Del.	0.014284	0.00	0.020399	0.00	0.002472	0.00	0.003445	0.00
1	0.013532	5.26	0.019385	4.97	0.002461	0.45	0.003483	-1.09
2	0.017860	-25.04	0.026939	-32.06	0.002370	4.12	0.003446	-0.01
3	0.013505	5.46	0.019369	5.05	0.002499	-1.09	0.003591	-4.23
4	0.012827	10.20	0.019476	4.53	0.002790	-12.87	0.004160	-20.75
5	0.013920	2.55	0.026516	-29.99	0.002425	1.92	0.003514	-2.00
6	0.012515	12.39	0.019506	4.38	0.002502	-1.23	0.003592	-4.26
7	0.012699	11.10	0.019400	4.90	0.002517	-1.83	0.003646	-5.82
8	0.012588	11.87	0.019335	5.21	0.002176	11.99	0.003132	9.10
9	0.013979	12.13	0.019809	2.89	0.002312	6.49	0.003265	5.23
10	0.012123	15.13	0.021618	-5.97	0.002503	-1.27	0.003647	-5.85
11	0.012461	12.77	0.019473	4.54	0.002146	13.19	0.003111	9.72
12	0.013480	5.63	0.019928	2.31	0.002390	3.31	0.003372	2.13
13	0.012584	11.90	0.019335	5.21	0.002189	11.43	0.003157	8.38
14	0.012461	12.77	0.019473	4.54	0.002153	12.92	0.003130	9.16

Il faut cependant prendre certaines précautions quant à l'utilisation de ces résultats ; les ensembles de FRANKE contiennent un grand nombre de sommets qui sont soient alignés 3 à 3, soient cocycliques 4 à 4. Ces dégénérescences peuvent influencer le comportement des algorithmes, dans la mesure où elles risquent d'accentuer les imprécisions de calcul, et ne permettent de définir la triangulation de Delaunay de manière unique.

Par ailleurs, ces résultats sont à comparer avec ceux des ensembles aléatoires ; il semble bien qu'il soit très délicat de conclure quant au test d'une méthode sur 2 ensembles seulement, puisque, par exemple, l'emploi des algorithmes 2 et 5 est bien meilleure en moyenne que ce qui est observé pour les points de FRANKE. C'est pour cette raison qu'une procédure plus complète de tests a été mise en place.

5.2.2 Résultats statistiques concernant l'erreur L_2

Suite aux résultats trouvés pour des ensembles de 33 points dont 12 sur les bords du carré unité, il a été décidé de tester les configurations suivantes :

- 15 points dont 4 sur l'enveloppe convexe,
- 15 points dont 8 sur l'enveloppe convexe,
- 33 points dont 8 sur l'enveloppe convexe,
- 33 points dont 16 sur l'enveloppe convexe,
- 50 points dont 16 sur l'enveloppe convexe,
- 100 points dont 20 sur l'enveloppe convexe,
- 250 points dont 40 sur l'enveloppe convexe,
- 500 points dont 60 sur l'enveloppe convexe.

Chaque série a comporté 500 tests d'ensembles engendrés aléatoirement dans le carré unité, sauf la dernière limitée volontairement à 200 tests. Les nombres de points variables sur l'enveloppe convexe vont permettre de déduire le rôle des triangles du bord, qui ne peuvent souvent pas être détruits par échange d'arête. Par ailleurs, la présence de nombreux points sur l'enveloppe convexe pour les ensembles de taille importante limitera l'erreur de ces triangles, dans la mesure où elle limitera leur taille.

Pour les ensembles de 15 à 100 points, toutes les méthodes ont été essayées ; au delà, on s'est restreint à celles qui ont pratiquement toujours renvoyé les meilleures erreurs, en conservant tout de même une méthode pour chaque type d'algorithme.

L'ensemble des résultats numériques, concernant le calcul des erreurs L_2 d'approximation du paraboloïde hyperbolique, est contenu dans les tableaux 5.5 à 5.8, qui se lisent de manière identique aux tableaux 5.2 ou 5.3.

5.2.2.1 Ensembles de petite taille

Les mesures semblent montrer un comportement général identique à celui observé au paragraphe précédent. Cependant, dans le cas particulier des ensembles à 15 points, on s'aperçoit que plusieurs méthodes peuvent donner la même triangulation (ce qui explique que, si on additionne les nombres de la colonne "meil.", on se retrouve très au-delà de 500). De plus, s'il n'y a que 4 points sur l'enveloppe convexe, les améliorations moyennes par rapport à Delaunay demeurent très faibles, et ne sont

Tableau 5.5 – résultats statistiques pour l'erreur L_2 d'ensembles de 15 points.

méthode	erreur L_2	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.074228	0.0098	3	0.9586	0.00	0.00	0.00
1	0.072872	0.0091	57	0.9764	1.83	-30.88	26.12
2	0.071773	0.0090	231	0.9914	3.31	-22.16	28.41
3	0.071965	0.0088	210	0.9887	3.05	-22.13	27.93
4	0.072909	0.0090	36	0.9759	1.78	-22.12	26.12
5	0.071776	0.0090	225	0.9914	3.30	-22.16	28.41
6	0.071961	0.0089	198	0.9888	3.05	-22.13	27.93
7	0.072982	0.0090	41	0.9750	1.68	-22.12	26.12
8	0.071516	0.0089	294	0.9950	3.65	-13.69	28.41
9	0.071817	0.0089	248	0.9908	3.25	-22.13	27.84
10	0.072999	0.0090	30	0.9747	1.66	-22.12	26.12
11	0.071572	0.0089	254	0.9942	3.58	-13.72	28.41
12	0.071851	0.0089	219	0.9903	3.20	-22.13	27.93
13	0.071295	0.0089	333	0.9980	3.95	0.00	28.41
14	0.071343	0.0089	287	0.9974	3.89	0.00	28.41
Tlo	0.071155	0.0089	500	1.0000	4.14	0.00	28.41

résultats si 4 points sont sur l'enveloppe convexe.

méthode	erreur L_2	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.030792	0.0067	0	0.8441	0.00	0.00	0 .00
1	0.031816	0.0074	41	0.8170	-3.33	-93.03	48.47
2	0.028619	0.0065	132	0.9083	7.06	-66.83	38.90
3	0.029694	0.0072	108	0.8754	3.57	-87.37	45.35
4	0.031981	0.0072	37	0.8128	-3.86	-93.03	48.47
5	0.028735	0.0066	109	0.9046	6.68	-66.83	38.40
6	0.029782	0.0072	94	0.8728	3.28	-87.37	45.35
7	0.032093	0.0072	43	0.8099	-4.23	-93.03	48.47
8	0.027132	0.0052	232	0.9580	11.88	-26.72	44.02
9	0.029041	0.0066	144	0.8950	5.69	-87.37	51.01
10	0.032170	0.0072	39	0.8080	-4.48	-93.03	48.47
11	0.027318	0.0054	183	0.9515	11.28	-52.87	43.58
12	0.029081	0.0068	129	0.8938	5.56	-87.37	51.01
13	0.026768	0.0047	282	0.9711	13.07	0.00	45.66
14	0.026874	0.0048	226	0.9672	12.73	0.00	45.66
Tlo	0.025993	0.0037	500	1.0000	15.59	0.13	51.01

résultats si 8 points sont sur l'enveloppe convexe.

pas du tout du même ordre que celles des autres ensembles ; les valeurs minimales et maximales des améliorations sont assez rapprochées.

Pour les tests à 8 points sur le bord, toutes les méthodes à ordre d'insertion aléatoire renvoient une erreur moyenne supérieure à celle de Delaunay ; ceci peut s'expliquer par le nombre trop réduit de points intérieurs qui ne permet pas un nombre d'échanges suffisamment important (nombre limité encore par l'irrégularité des triangulations due au caractère aléatoire des insertions).

5.2.2.2 Ensembles de taille moyenne

La lecture des tableaux 5.2 et 5.6 permet d'évaluer l'importance du rôle du nombre de points imposés sur l'enveloppe convexe. Sur l'exemple des ensembles à 33 points, où ont été testées des configurations à 8, 12 ou 16 sommets sur les bords, on peut remarquer que les meilleures améliorations moyennes sont respectivement de 10.57%, 18.60% et 23.59%.

S'il est clair que 8 points ne semblent pas suffisants (l'erreur L_2 moyenne dans ce cas est 30% plus grande que pour les deux autres cas), il ne paraissait pas évident, initialement, qu'en prendre 16 (soit un point sur deux) constituerait un choix très judicieux, d'autant plus que l'erreur moyenne de la triangulation de Delaunay est

Tableau 5.6 – résultats statistiques pour l'erreur L_2 d'ensembles de taille moyenne.

méthode	erreur L_2	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.018686	0.0020	0	0.8943	0.00	0.00	0.00
1	0.018216	0.0018	9	0.9174	2.51	-34.60	49.88
2	0.017570	0.0016	36	0.9511	5.97	-26.70	36.39
3	0.017635	0.0018	22	0.9476	5.62	-55.61	36.60
4	0.018326	0.0018	8	0.9119	1.92	-37.35	49.88
5	0.017589	0.0017	31	0.9501	5.87	-44.88	36.19
6	0.017608	0.0018	23	0.9491	5.77	-55.61	36.60
7	0.018238	0.0018	9	0.9163	2.39	-35.76	49.77
8	0.017063	0.0014	126	0.9794	8.68	-5.40	42.46
9	0.017331	0.0015	77	0.9642	7.25	-28.21	49.84
10	0.018299	0.0018	9	0.9132	2.07	-36.88	49.77
11	0.017120	0.0015	90	0.9761	8.38	-23.61	42.46
12	0.017347	0.0015	46	0.9634	7.16	-19.16	49.76
13	0.016936	0.0012	172	0.9867	9.36	0.54	36.45
14	0.016969	0.0012	100	0.9848	9.19	0.54	36.43
Tlo	0.016711	0.0011	500	1.0000	10.57	0.60	49.88

résultats pour 33 points dont 8 sur l'enveloppe convexe.

méthode	erreur L_2	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.014459	0.0034	0	0.7641	0.00	0.00	0.00
1	0.013707	0.0029	14	0.8060	5.20	-86.87	60.21
2	0.013542	0.0035	17	0.8158	6.34	-56.59	46.58
3	0.013128	0.0029	28	0.8416	9.20	-77.11	61.29
4	0.013707	0.0028	16	0.8060	5.20	-87.05	59.95
5	0.013445	0.0036	20	0.8217	7.01	-75.37	46.12
6	0.013016	0.0028	24	0.8488	9.98	-77.11	60.98
7	0.012860	0.0027	57	0.8591	11.06	-63.92	60.05
8	0.012119	0.0027	112	0.9116	16.18	-25.92	44.73
9	0.012517	0.0027	82	0.8826	13.43	-64.04	61.58
10	0.012933	0.0028	40	0.8543	10.55	-87.82	60.37
11	0.012193	0.0028	71	0.9061	15.67	-59.70	43.86
12	0.012555	0.0027	49	0.8800	13.17	-77.13	61.26
13	0.011883	0.0023	120	0.9297	17.81	0.06	52.61
14	0.011960	0.0024	76	0.9237	17.28	-0.20	52.59
Tlo	0.011048	0.0018	500	1.0000	23.59	0.56	61.58

résultats pour 33 points dont 16 sur l'enveloppe convexe.

méthode	erreur L_2	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.008771	0.0015	0	0.8038	0.00	0.00	0.00
1	0.008683	0.0014	5	0.8120	1.01	-56.89	45.06
2	0.008313	0.0016	15	0.8481	5.23	-76.40	42.39
3	0.008195	0.0014	14	0.8603	6.57	-73.67	42.33
4	0.008749	0.0014	8	0.8059	0.26	-67.39	45.06
5	0.008280	0.0017	14	0.8514	5.60	-77.81	39.24
6	0.008192	0.0014	11	0.8606	6.60	-73.69	42.32
7	0.008225	0.0014	25	0.8572	6.23	-58.94	48.47
8	0.007439	0.0012	124	0.9477	15.19	-17.35	48.11
9	0.007869	0.0014	60	0.8959	10.28	-73.02	47.59
10	0.008246	0.0014	17	0.8550	5.99	-64.53	48.47
11	0.007622	0.0013	49	0.9250	13.11	-47.66	46.30
12	0.007909	0.0014	41	0.8914	9.82	-73.06	47.47
13	0.007388	0.0011	146	0.9542	15.76	1.86	44.63
14	0.007420	0.0011	58	0.9502	15.41	1.86	44.49
Tlo	0.007050	0.0009	500	1.0000	19.62	1.86	48.47

résultats pour 50 points dont 16 sur l'enveloppe convexe.

alors plus importante que si on prend 12 sommets sur les bords du carré unité. De plus, le nombre limité de points intérieurs laisse supposer que le nombre d'échanges possible sera plus faible (ce qui est confirmé par le tableau 5.12).

Une explication de ce phénomène pourrait être qu'en ayant augmenté le nombre de sommets sur l'enveloppe convexe, on conserve des triangles au bord de forme meilleure et dont le rôle sur l'erreur totale est d'autant diminué. Comme la triangulation de Delaunay ne permet pas d'obtenir de bons triangles intérieurs, elle donne de plus mauvais résultats, alors que les échanges de diagonales des triangulations localement optimales vont faire baisser l'erreur de manière très significative.

De toutes les manières, les erreurs moyennes de la plupart des méthodes sont globalement identiques, que l'on prenne 12 ou 16 points sur l'enveloppe convexe (ce qui implique, d'ailleurs, que les erreurs des triangles intérieurs sont plus importantes dans le second cas). Seule la triangulation de Delaunay renvoie une erreur "vraiment" supérieure dans le premier cas : cela explique que les améliorations soient moins bonnes pour ce cas-là.

5.2.2.3 Ensembles de taille importante

En augmentant le nombre de sommets, on peut se rendre compte que la meilleure amélioration moyenne (dernière ligne des tableaux) reste à peu près constante, située aux alentours de 20%. On peut également observer un resserrement des améliorations extrémales : plus le nombre de points est important, moins les valeurs minimales sont grandes (on passe, pour la méthode 6, de -38% à -15% pour 100 et 500 points), et, parallèlement, moins les valeurs maximales sont importantes (50% puis 27% toujours pour la méthode 6 et des ensembles à 100 et 500 points).

Mieux : non seulement les améliorations moyennes restent constantes, mais les erreurs moyennes diminuent linéairement par rapport à l'augmentation du nombre de sommets, et cela de manière quasiment exacte. Le tableau 5.7 résume bien ce phénomène.

Tableau 5.7 – facteurs de diminutions des erreurs, lorsque le nombre de points augmente.

nb points	facteur	err. Del.	facteur	err. Tlo	facteur
33 (12)		0.013758		0.011199	
50 (16)	$\times$ 1.5	0.008771	$\div$ 1.57	0.007050	$\div$ 1.59
100 (20)	$\times$ 2	0.004323	$\div$ 2.03	0.003527	$\div$ 2.00
250 (40)	$\times$ 2.5	0.001682	$\div$ 2.57	0.001335	$\div$ 2.64
500 (60)	$\times$ 2	0.000817	$\div$ 2.06	0.000659	$\div$ 2.03

5.2.2.4 Répartition des erreurs et des améliorations

Pour mieux comprendre comment le nombre de points peut influencer les erreurs commises par les triangulations, et les améliorations qui découlent de l'utilisation

Tableau 5.8 – résultats statistiques pour l'erreur L_2 d'ensembles de taille importante.

méthode	erreur L_2	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.004374	0.0007	0	0.8063	0.00	0.00	0.00
1	0.004323	0.0006	8	0.8158	1.16	-71.27	52.97
2	0.004134	0.0007	6	0.8531	5.49	-53.27	34.70
3	0.004066	0.0006	6	0.8674	7.05	-37.93	51.25
4	0.004216	0.0005	5	0.8364	3.61	-44.64	51.26
5	0.004122	0.0007	2	0.8554	5.75	-46.65	28.24
6	0.004045	0.0006	4	0.8719	7.53	-37.97	50.65
7	0.003971	0.0005	34	0.8880	9.20	-52.52	50.24
8	0.003692	0.0005	108	0.9552	15.59	-19.01	51.36
9	0.003839	0.0005	47	0.9185	12.22	-35.52	51.96
10	0.004005	0.0005	24	0.8806	8.44	-54.98	47.52
11	0.003817	0.0007	42	0.9240	12.74	-81.71	43.20
12	0.003857	0.0005	23	0.9144	11.83	-36.35	50.56
13	0.003639	0.0004	150	0.9692	16.81	4.20	51.72
14	0.003660	0.0004	64	0.9635	16.32	3.95	51.32
Tlo	0.003527	0.0003	500	1.0000	19.37	4.91	52.97

résultats pour 100 points dont 20 sur l'enveloppe convexe.

méthode	erreur L_2	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.001682	0.0002	0	0.7936	0.00	0.00	0.00
3	0.001550	0.0002	2	0.8615	7.89	-40.34	33.57
6	0.001538	0.0002	4	0.8680	8.57	-37.69	36.03
8	0.001391	0.0001	127	0.9599	17.33	-25.27	34.19
9	0.001417	0.0001	107	0.9423	15.78	-23.44	39.46
11	0.001436	0.0002	23	0.9296	14.63	-25.30	32.32
12	0.001439	0.0001	34	0.9280	14.48	-27.60	40.40
13	0.001372	0.0001	168	0.9735	18.48	8.00	33.88
14	0.001380	0.0001	42	0.9678	18.00	7.54	32.06
Tlo	0.001335	0.0001	500	1.0000	20.64	8.41	40.40

résultats pour 250 points dont 40 sur l'enveloppe convexe.

méthode	erreur L_2	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.000817	0.0001	0	0.8068	0.00	0.00	0.00
3	0.000751	0.0001	0	0.8779	8.10	-9.33	26.50
6	0.000746	0.0001	1	0.8840	8.73	-14.91	27.01
9	0.000686	0.0000	41	0.9614	16.08	-6.13	31.08
12	0.000695	0.0000	12	0.9487	14.96	-5.94	30.49
13	0.000666	0.0000	126	0.9900	18.51	9.37	28.26
14	0.000671	0.0000	20	0.9825	17.88	8.85	28.91
Tlo	0.000659	0.0000	200	1.0000	19.32	12.20	31.08

résultats pour 500 points dont 60 sur l'enveloppe convexe.

de triangulations localement optimales, la figure 5.3 contient des histogrammes qui se lisent de la manière suivante :

- chaque histogramme est associé à une série de tests pour des ensembles de taille fixée (Ω désigne le bord du carré unité, enveloppe convexe de tous les ensembles).

- un histogramme est une courbe de 50 segments (10 pour les ensembles à 500 points), où chaque sommet représente le nombre des tests qui ont donné une erreur comprise entre 2 valeurs fixées (l'écart entre 2 valeurs consécutives étant toujours identique, égal au cinquantième de la différence entre les erreurs minimales et maximales apparues au cours de tous les tests) ; l'abscisse d'un sommet de la courbe est la valeur inférieure des erreurs, et son ordonnée le nombre de tests pour l'intervalle en question.

– en traits fins est représentée la répartition des erreurs pour la triangulation de Delaunay, en traits pointillés épais celle des erreurs pour la triangulation issue de la méthode 13 (en général, meilleure méthode), en traits épais la répartition des erreurs pour la meilleure triangulation localement optimale parmi toutes celles testées.

Toutes les séries de tests sont représentées, exceptée celle à 15 points dont 4 sur Ω du fait de sa trop grande irrégularité (qui entraine qu'une interprétation de la courbe serait très aléatoire).

Une première observation montre que les courbes de la méthode 13 et de la meilleure triangulation localement optimale sont très proches, ce qui confirme bien que l'utilisation de l'algorithme de LAWSON à partir de Delaunay est particulièrement bien adaptée au problème (ou, du moins, aux mesures d'erreurs et de comparaisons effectuées).

Par ailleurs, on note qu'avec l'augmentation du nombre de sommets, l'écart entre les histogrammes pour les bonnes triangulations et celui pour Delaunay augmente également, jusqu'à n'avoir plus qu'une petite zone commune (pour 250 et 500 points). Cela signifie qu'il est de plus en plus possible d'améliorer une triangulation lorsque les points sont en nombre plus important (bien que le facteur d'amélioration ne change pas beaucoup en moyenne).

La figure 5.4 contient les histogrammes de répartition des améliorations d'erreurs obtenues en passant de la triangulation de Delaunay à une triangulation localement optimale :

– chaque histogramme est associé à une série de tests pour des ensembles de taille fixée.

– un histogramme est à nouveau une courbe de 50 segments (10 pour les ensembles à 500 points), où chaque sommet a pour abscisse x une valeur inférieure d'amélioration (comprise entre 0 et 70%) et pour ordonnée le nombre de tests ayant fourni une amélioration comprise entre x et $x + \frac{70}{50}$.

– en traits fins est représentée la répartition des améliorations pour le passage de la triangulation de Delaunay à celle résultant de l'application de la méthode 13, et en traits épais la répartition des améliorations pour le passage de Delaunay à la meilleure triangulation localement optimale.

Le plus remarquable est sans doute le resserrement des courbes avec l'augmentation du nombre de sommets ; à nombre de tests égaux, les pics sont de plus en plus hauts (67, 88, 100 pour respectivement 50, 100, 250 points à raison de 500 tests).

Une fois encore, les différences entre la méthode 13 et la meilleure triangulation localement optimale ne sont pas très importantes, mis à part, peut-être, pour les ensembles de 33 points dont 16 sur l'enveloppe convexe ; l'histogramme de cette série de tests est d'ailleurs beaucoup plus étendu que les autres (il est comparable en cela à celui des ensembles de 15 points dont 8 sur Ω). Le fait de ne pas laisser suffisamment de points intérieurs fait donc bien apparaître des comportements très irréguliers (qui, comme on l'a déjà expliqué, peuvent provenir de certaines mauvaises triangulations de Delaunay ou d'un nombre de swaps limité).

Figure 5.3 – répartition des erreurs L_2 des triangulations de Delaunay (trait fin), des triangulations construites grâce à la méthode 13 (pointillés) et des meilleures triangulations localement optimales (trait épais), pour les différents ensembles de points aléaloires testés.

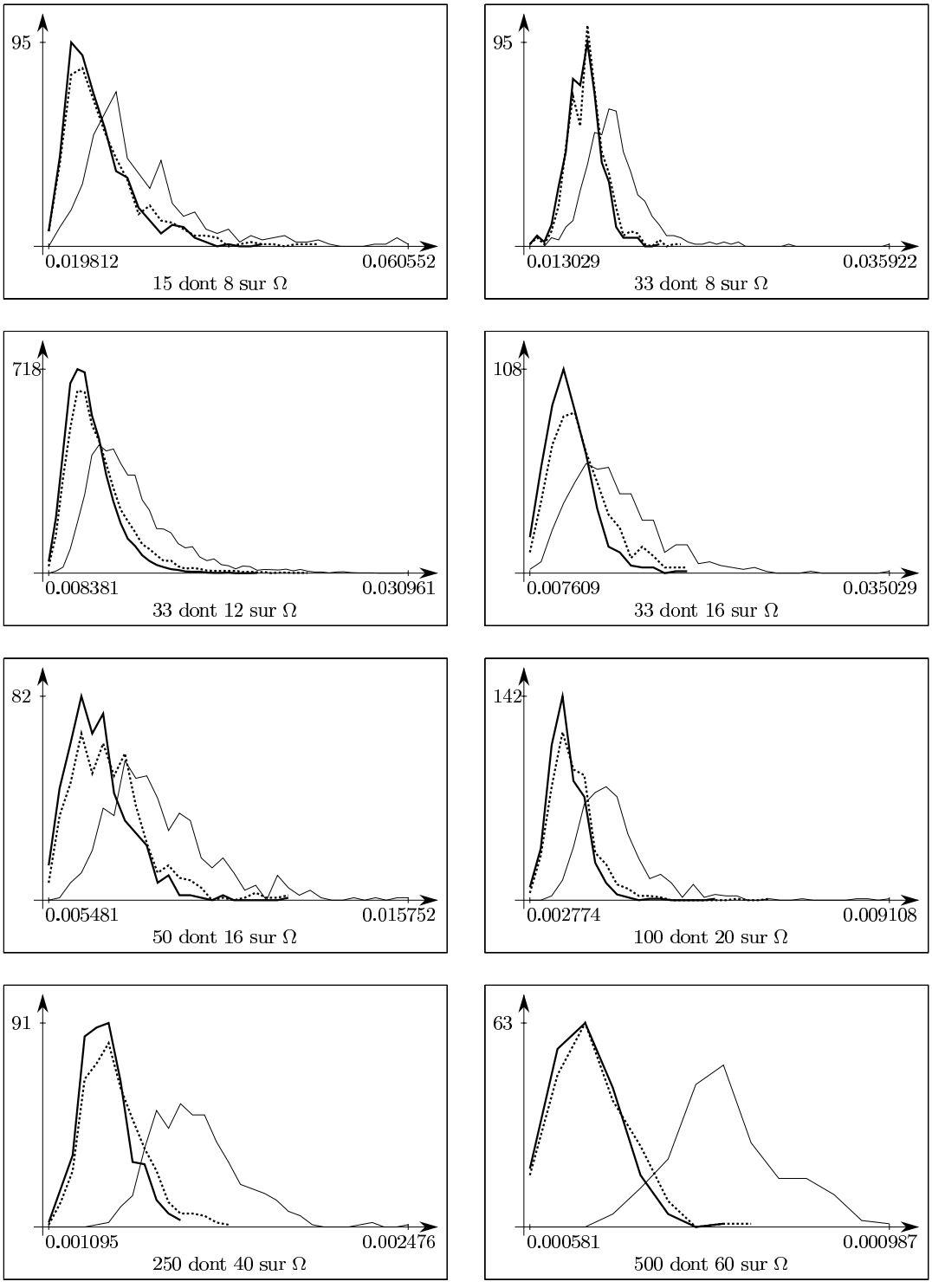


Figure 5.4 – répartition des améliorations (en pourcentages) sur l'erreur L_2 obtenues par les meilleures triangulations localement optimales (trait épais) et par application de la méthode 13 (trait fin), pour les différents ensembles de points aléaloires testés.

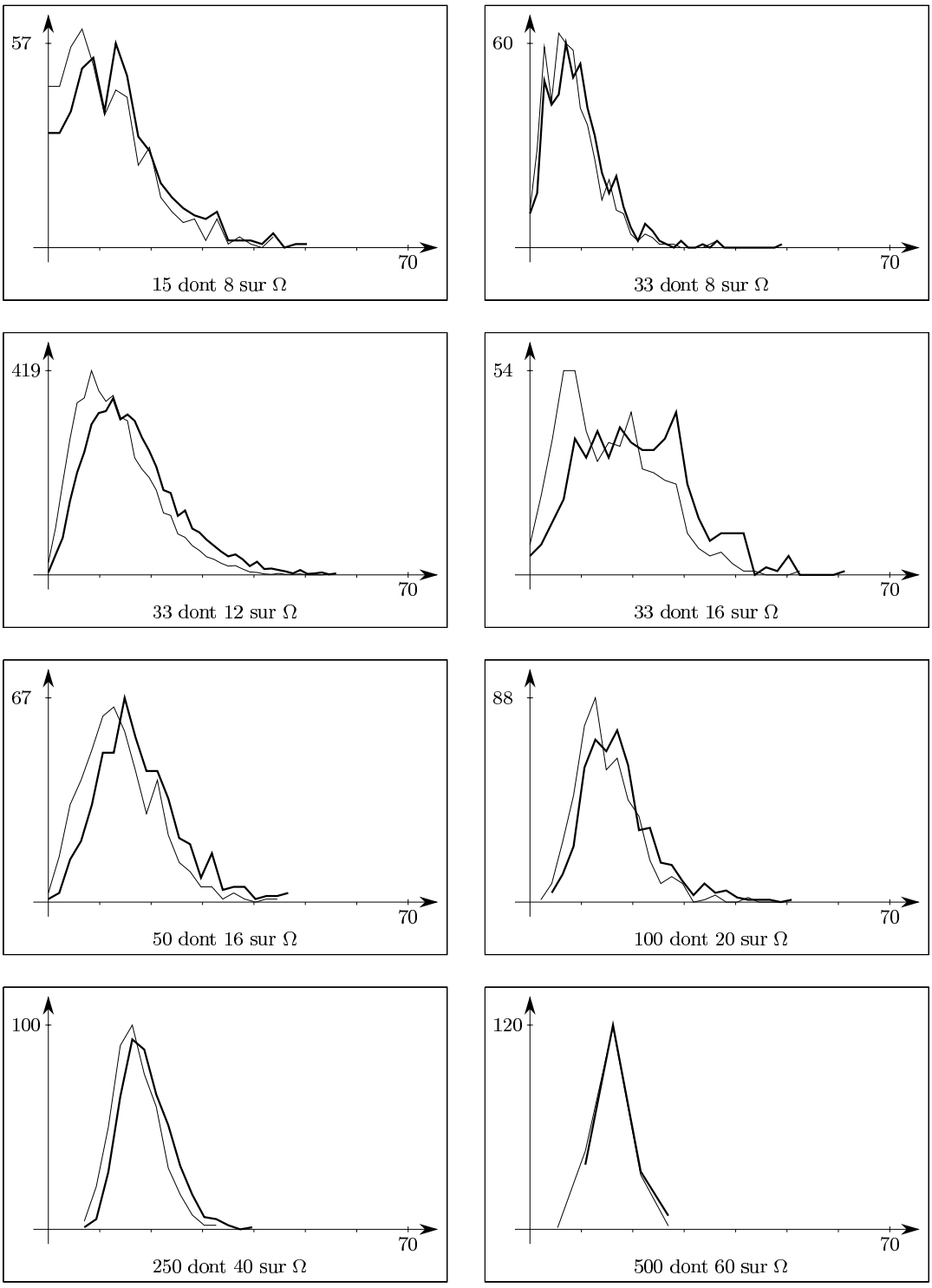


Tableau 5.9 – résultats statistiques pour l'erreur L_1 d'ensembles de 15 points.

méthode	erreur L_1	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.051146	0.0086	1	0.9131	0.00	0.00	0.00
1	0.049154	0.0075	48	0.9501	3.89	-37.50	32.04
2	0.047575	0.0075	175	0.9817	6.98	-28.64	35.05
3	0.047888	0.0074	136	0.9753	6.37	-30.88	33.93
4	0.049219	0.0075	40	0.9489	3.77	-30.84	29.91
5	0.047527	0.0075	225	0.9827	7.08	-28.64	35.05
6	0.047833	0.0074	187	0.9764	6.48	-30.88	33.93
7	0.049320	0.0075	35	0.9470	3.57	-30.88	29.91
8	0.047290	0.0074	214	0.9876	7.54	-22.78	35.16
9	0.047682	0.0074	165	0.9795	6.77	-30.88	33.31
10	0.049330	0.0075	36	0.9468	3.55	-30.88	29.91
11	0.047292	0.0074	263	0.9876	7.54	-22.56	35.05
12	0.047675	0.0075	214	0.9796	6.79	-30.88	33.93
13	0.047037	0.0073	243	0.9929	8.03	0.00	35.16
14	0.047034	0.0073	301	0.9930	8.04	0.00	35.05
Tlo	0.046703	0.0073	500	1.0000	8.69	0.00	35.16

résultats si 4 points sont sur l'enveloppe convexe.

méthode	erreur L_1	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.023996	0.0047	0	0.8251	0.00	0.00	0.00
1	0.023949	0.0047	39	0.8267	0.19	-70.92	44.52
2	0.021593	0.0044	107	0.9169	10.01	-38.54	40.78
3	0.022330	0.0047	91	0.8866	6.94	-68.61	43.63
4	0.024034	0.0045	37	0.8238	-0.16	-70.92	44.52
5	0.021624	0.0044	122	0.9155	9.88	-43.14	41.24
6	0.022364	0.0047	110	0.8853	6.80	-68.61	43.63
7	0.024176	0.0045	32	0.8189	-0.75	-70.92	44.52
8	0.020674	0.0036	184	0.9576	13.84	-27.22	43.63
9	0.021950	0.0044	116	0.9019	8.52	-65.49	47.37
10	0.024196	0.0045	32	0.8182	-0.84	-70.92	44.52
11	0.020705	0.0037	227	0.9562	13.72	-40.32	39.04
12	0.021908	0.0044	154	0.9037	8.70	-65.49	47.37
13	0.020294	0.0031	235	0.9756	15.43	-0.44	43.63
14	0.020296	0.0031	288	0.9754	15.42	0.01	43.63
Tlo	0.019798	0.0026	500	1.0000	17.49	0.39	47.37

résultats si 8 points sont sur l'enveloppe convexe.

5.2.3 Résultats statistiques concernant l'erreur L_1

Pour déterminer s'il existe de grandes différences entre les triangulations localement optimales pour l'erreur L_2 ou pour l'erreur L_1, les tableaux 5.9 à 5.11 et les figures 5.5 et 5.6 résument les statistiques appliquées aux erreurs L_1 des triangulations calculées pour les séries de tests précédentes.

Le rapport entre les erreurs L_2 et L_1 des différentes méthodes pour chaque série est toujours à peu près constant : l'erreur L_2 est 1.3 à 1.5 plus grande que l'erreur L_1. Ce rapport constant implique que le phénomène de décroissance des erreurs avec le nombre de sommets, observé pour les erreurs L_2, est également vrai pour les erreurs L_1.

De plus, les améliorations obtenues sont du même ordre pour les erreurs ; à partir d'un nombre de points raisonnable, on peut construire une triangulation localement optimale 20% meilleure que Delaunay pour l'erreur L_1.

Ce sont aussi les mêmes méthodes qui rendent les meilleurs résultats ; ainsi, globalement, c'est l'algorithme de LAWSON appliqué à la triangulation de Delaunay et dont les échanges vont chercher à faire diminuer l'erreur L_1 (méthode 14) qui est le meilleur, du moins en moyenne et pour plus d'un tiers des tests.

On peut s'apercevoir, et cette remarque s'applique également à l'erreur L_2, que

Tableau 5.10 – résultats statistiques pour l'erreur L_1 d'ensembles de points de taille moyenne.

méthode	erreur L_1	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.013296	0.0016	0	0.8478	0.00	0.00	0.00
1	0.012669	0.0014	8	0.8897	4.71	-38.27	46.43
2	0.012061	0.0013	26	0.9346	9.29	-18.20	32.94
3	0.012117	0.0013	11	0.9303	8.87	-36.93	39.45
4	0.012729	0.0014	10	0.8856	4.26	-38.27	46.43
5	0.012019	0.0013	44	0.9379	9.60	-30.82	33.01
6	0.012046	0.0013	30	0.9358	9.40	-36.93	39.45
7	0.012695	0.0014	9	0.8879	4.51	-34.97	46.05
8	0.011661	0.0012	62	0.9666	12.29	-4.12	39.84
9	0.011881	0.0012	37	0.9488	10.64	-17.58	46.33
10	0.012704	0.0014	12	0.8873	4.45	-31.36	46.05
11	0.011635	0.0012	120	0.9688	12.49	-14.24	39.09
12	0.011833	0.0012	67	0.9526	11.00	-16.29	46.47
13	0.011528	0.0011	74	0.9778	13.29	1.06	41.24
14	0.011505	0.0010	164	0.9797	13.46	1.06	41.14
Tlo	0.011272	0.0009	500	1.0000	15.22	2.11	46.47

résultats pour 33 points dont 8 sur l'enveloppe convexe.

méthode	erreur L_1	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.010394	0.0021	0	0.7694	0.00	0.00	0.00
1	0.009680	0.0015	8	0.8262	6.87	-50.99	56.15
2	0.009455	0.0020	12	0.8458	9.03	-40.83	42.54
3	0.009241	0.0017	16	0.8654	11.09	-44.70	57.61
4	0.009649	0.0015	16	0.8288	7.17	-46.58	56.89
5	0.009310	0.0020	27	0.8591	10.43	-49.03	41.91
6	0.009130	0.0016	29	0.8760	12.16	-43.98	57.93
7	0.009197	0.0015	27	0.8695	11.51	-42.87	56.17
8	0.008687	0.0015	57	0.9206	16.42	-13.83	40.68
9	0.008893	0.0015	41	0.8993	14.44	-40.35	58.04
10	0.009191	0.0015	50	0.8701	11.57	-56.86	57.72
11	0.008618	0.0016	115	0.9281	17.09	-32.02	43.15
12	0.008866	0.0015	70	0.9021	14.71	-43.94	58.36
13	0.008473	0.0013	72	0.9439	18.49	0.34	49.21
14	0.008453	0.0014	139	0.9461	18.67	0.35	49.22
Tlo	0.007998	0.0011	500	1.0000	23.06	1.07	58.36

résultats pour 33 points dont 16 sur l'enveloppe convexe.

méthode	erreur L_1	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.006408	0.0009	0	0.7983	0.00	0.00	0.00
1	0.006218	0.0008	3	0.8227	2.96	-36.25	36.86
2	0.005861	0.0009	6	0.8729	8.55	-36.62	35.93
3	0.005818	0.0008	8	0.8793	9.22	-39.29	38.24
4	0.006237	0.0008	5	0.8203	2.68	-45.45	39.20
5	0.005793	0.0009	23	0.8831	9.61	-36.64	33.90
6	0.005774	0.0008	15	0.8860	9.90	-39.22	38.25
7	0.005917	0.0008	11	0.8646	7.66	-35.62	41.30
8	0.005374	0.0006	69	0.9519	16.13	-7.87	40.75
9	0.005630	0.0007	35	0.9086	12.14	-39.05	45.26
10	0.005897	0.0008	19	0.8676	7.99	-38.99	44.26
11	0.005422	0.0007	87	0.9435	15.39	-28.07	38.95
12	0.005609	0.0008	68	0.9120	12.47	-39.47	45.32
13	0.005316	0.0006	50	0.9623	17.04	3.40	38.38
14	0.005294	0.0006	157	0.9664	17.39	3.40	40.68
Tlo	0.005116	0.0005	500	1.0000	20.17	3.40	45.32

résultats pour 50 points dont 16 sur l'enveloppe convexe.

Tableau 5.11 – résultats statistiques pour l'erreur L_1 d'ensembles de points de taille importante.

méthode	erreur L_1	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.003141	0.0004	0	0.7959	0.00	0.00	0.00
1	0.003016	0.0003	4	0.8287	3.96	-41.76	42.36
2	0.002853	0.0003	3	0.8762	9.17	-19.46	31.94
3	0.002826	0.0003	0	0.8844	10.01	-21.74	39.52
4	0.002911	0.0003	7	0.8588	7.32	-24.67	41.80
5	0.002808	0.0003	3	0.8902	10.59	-22.24	31.53
6	0.002783	0.0003	9	0.8983	11.40	-17.50	41.78
7	0.002800	0.0003	6	0.8928	10.85	-23.40	41.60
8	0.002612	0.0003	60	0.9568	16.82	3.64	44.02
9	0.002706	0.0002	17	0.9237	13.84	-14.85	40.90
10	0.002786	0.0003	26	0.8971	11.28	-22.46	44.21
11	0.002628	0.0003	71	0.9513	16.33	-27.77	37.82
12	0.002685	0.0003	49	0.9308	14.49	-13.50	41.83
13	0.002573	0.0002	60	0.9714	18.07	6.92	44.62
14	0.002557	0.0002	189	0.9774	18.57	7.68	45.10
Tlo	0.002500	0.0002	500	1.0000	20.41	8.03	45.10

résultats pour 100 points dont 20 sur l'enveloppe convexe.

méthode	erreur L_1	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.001170	0.0001	0	0.7834	0.00	0.00	0.00
3	0.001035	0.0001	0	0.8855	11.53	-12.82	26.87
6	0.001011	0.0001	9	0.9065	13.58	-13.86	30.81
8	0.000955	0.0001	37	0.9604	18.43	4.33	29.38
9	0.000973	0.0001	37	0.9421	16.85	-8.66	32.74
11	0.000955	0.0001	62	0.9598	18.38	2.28	29.22
12	0.000963	0.0001	86	0.9515	17.67	-7.99	34.12
13	0.000940	0.0001	25	0.9750	19.65	10.43	30.70
14	0.000929	0.0001	244	0.9866	20.59	11.88	31.44
Tlo	0.000917	0.0001	500	1.0000	21.66	12.65	34.12

résultats pour 250 points dont 40 sur l'enveloppe convexe.

méthode	erreur L_1	écart	meil.	rapport	améliorations (en %)		
					moy.	min.	max.
Del.	0.000562	0.0000	0	0.7806	0.00	0.00	0.00
3	0.000495	0.0000	0	0.8852	11.82	-0.27	23.03
6	0.000482	0.0000	0	0.9088	14.10	1.38	24.22
9	0.000464	0.0000	1	0.9452	17.42	5.57	27.39
12	0.000457	0.0000	32	0.9585	18.56	7.87	29.83
13	0.000448	0.0000	3	0.9790	20.27	15.01	27.98
14	0.000440	0.0000	164	0.9965	21.67	16.45	28.44
Tlo	0.000438	0.0000	200	1.0000	21.94	16.48	29.83

résultats pour 500 points dont 60 sur l'enveloppe convexe.

deux méthodes, qui ne diffèrent que par le type de leur swap, donnent des erreurs moyennes tout à fait similaires. Ce qui signifie bien qu'il n'y a pas beaucoup de différence entre minimiser l'erreur L_1 et l'erreur L_2, et que les similitudes entre les courbes de séparation des deux critères n'étaient pas fortuites.

Concernant les histogrammes, à nouveau les mêmes phénomènes sont observés, mais de manière plus prononcée : la courbe des répartitions des erreurs pour la meilleure triangulation localement et celle issue de la méthode 14 sont encore plus proches, et leur éloignement avec celle de la triangulation de Delaunay est plus marqué, à un point tel que, pour les ensembles à 500 points, elles n'ont pratiquement plus aucune zone d'erreurs commune.

Les courbes de répartition des améliorations se ressemblent également beaucoup, bien que celles pour l'erreur L_1 apparaissent un peu plus régulières.

Figure 5.5 – répartition des erreurs L_1 des triangulations de Delaunay (trait fin), des triangulations construites grâce à la méthode 14 (pointillés) et des meilleures triangulations localement optimales (trait épais), pour les différents ensembles de points aléaloires testés.

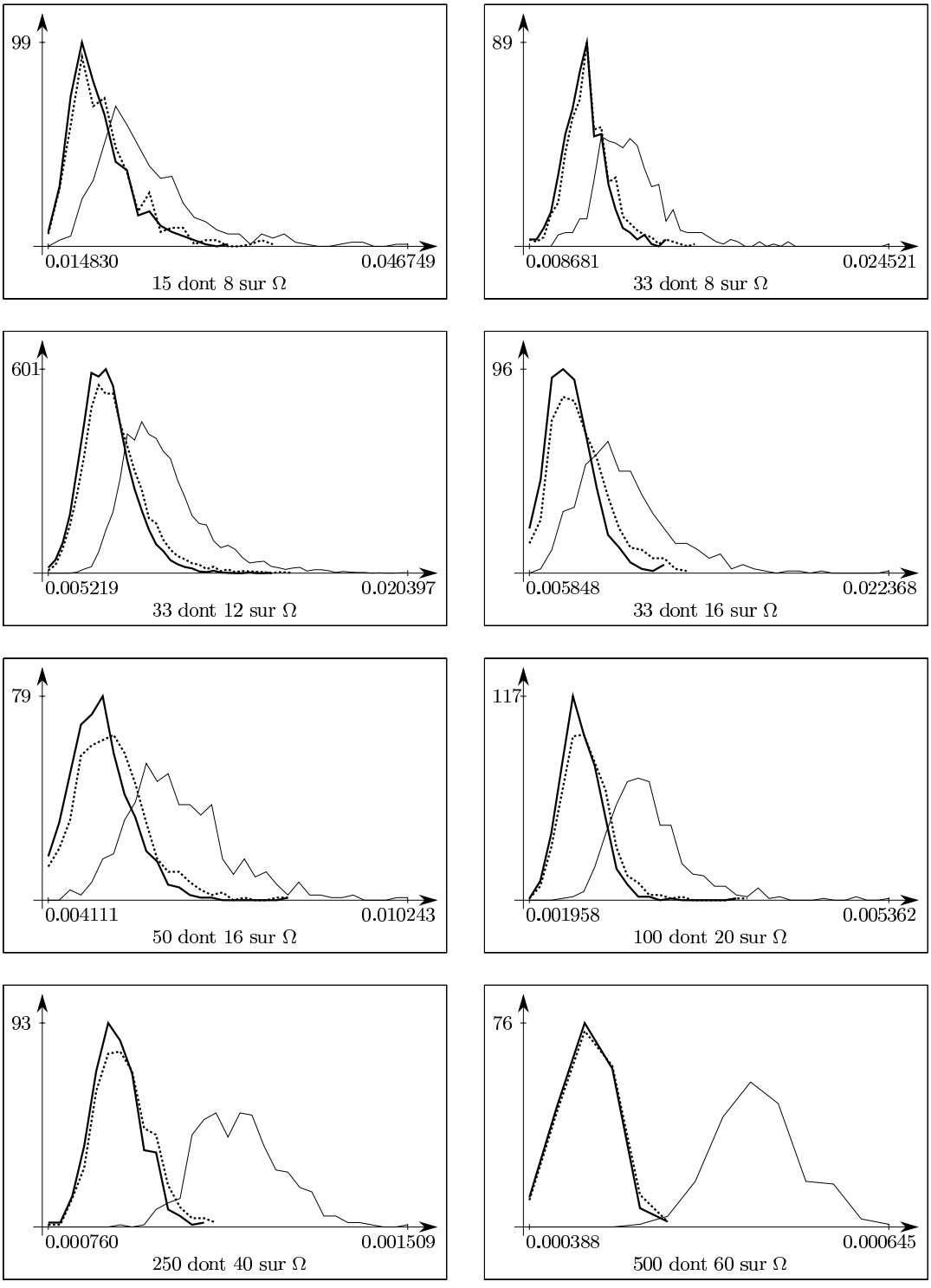
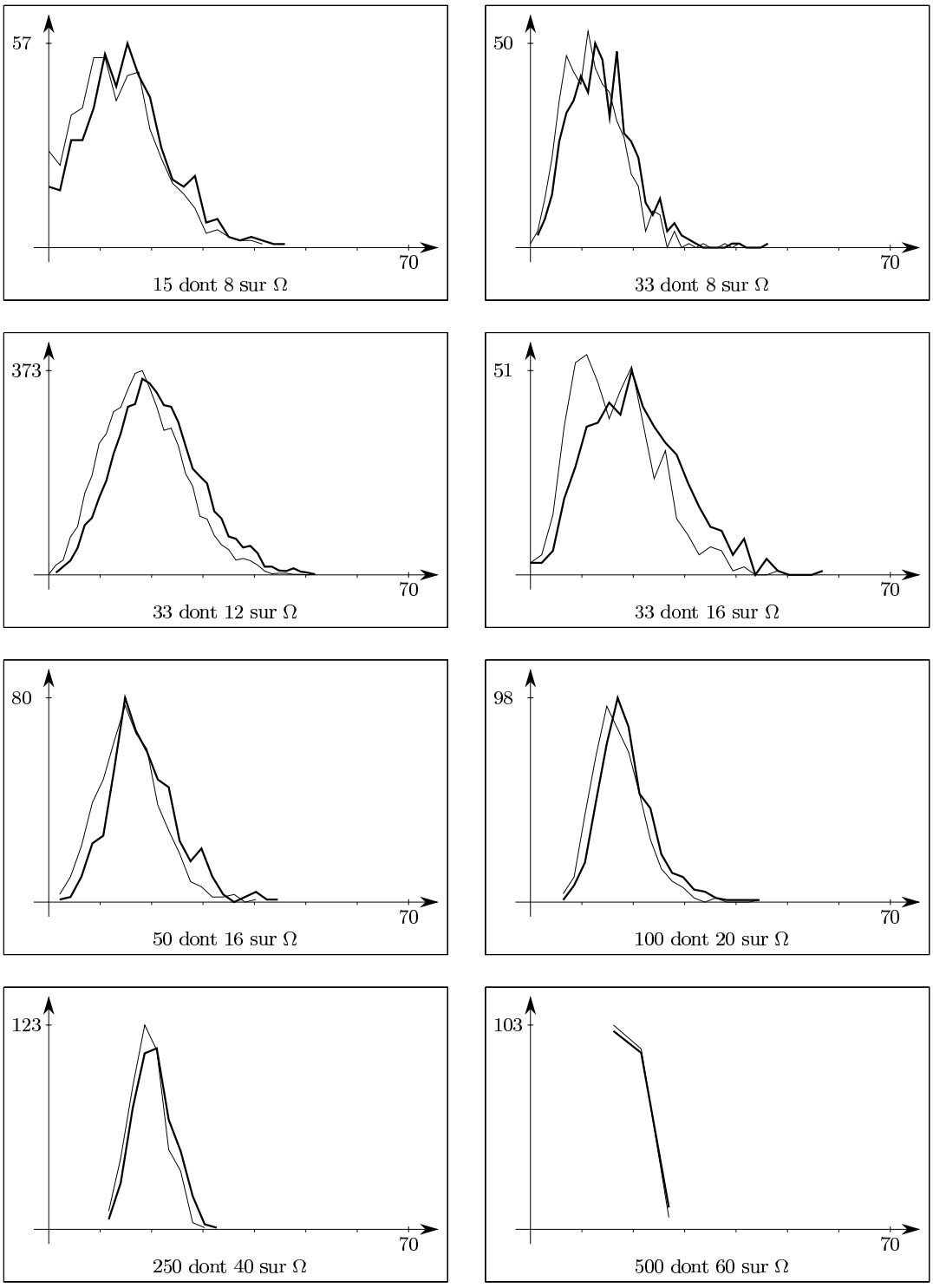


Figure 5.6 – répartition des améliorations (en pourcentages) sur l'erreur L_1 obtenues par les meilleures triangulations localement optimales (trait épais) et par application de la méthode 14 (trait fin), pour les différents ensembles de points aléaloires testés.



5.2.4 Nombre d'échanges et complexités

Pour avoir une idée de la complexité relative des algorithmes, il est possible de compter pour chaque méthode le nombre total d'échanges effectués pour obtenir les triangulations localement optimales. Le tableaux 5.12 contient le nombre d'échanges moyen réalisés sur la totalité des tests de chaque méthode pour toutes les séries (Ω désigne toujours le bord du carré unité).

Tableau 5.12 – nombres d'échanges moyens pour chaque méthode et série de tests.

	15 points		33 points			50 pts	100 pts	250 pts	500 pts
mét.	4 Ω	8 Ω	8 Ω	12 Ω	16 Ω	16 Ω	20 Ω	40 Ω	60 Ω
1	8.90	6.48	30.10	25.95	24.90	48.59	126.78		
2	12.50	12.40	54.29	53.79	52.09	102.73	279.08		
3	9.12	6.35	27.54	25.58	22.87	48.51	131.85	429.95	1003.17
4	8.43	6.98	29.71	28.02	26.04	48.51	138.95		
5	12.48	12.32	54.36	53.84	52.34	102.80	279.54		
6	9.20	6.29	27.80	25.67	23.31	48.78	133.01	436.64	1020.38
7	8.01	5.53	29.11	33.27	35.32	69.28	191.66		
8	12.00	12.48	50.36	49.89	47.89	94.02	254.71	856.14	
9	8.80	6.09	26.41	24.00	21.30	44.92	120.85	390.73	909.94
10	7.86	5.45	28.86	32.95	35.30	68.91	191.58		
11	11.92	12.16	49.99	49.48	47.74	93.17	253.30	857.02	
12	8.79	6.03	26.41	23.84	21.48	44.86	120.72	392.45	915.88
13	4.83	4.41	12.87	12.10	10.94	19.51	43.55	112.89	229.03
14	4.75	4.36	12.86	12.02	10.95	19.44	43.52	116.52	242.33

5.2.4.1 Observations sur les nombres d'échanges

Les échanges dénombrés ne tiennent pas compte de ceux qui sont intervenus dans la construction de la triangulation de Delaunay (méthodes 13 et 14) ; cela explique, en partie, que les nombres en question soient toujours plus faibles que les autres.

Il est intéressant aussi de constater que, finalement, les méthodes du type TOI (1 à 6) ne réalisent pas plus de swaps que celles du type LAWSON. On aurait pu, en effet, s'attendre à ce qu'en essayant de swapper le plus possible après chaque insertion de point, on réalise plus d'échanges qu'en le faisant après avoir obtenu une première triangulation globale. Il n'en est rien, et cela peut expliquer le mauvais comportement général des méthodes du type TOI. Il est, en revanche, difficile d'avancer une explication à ce phénomène, mis à part, peut-être, un blocage des triangulations autour d'un point stable (en insérant les points les uns après les autres, et en retriangulant à chaque fois, il semble que l'on converge vers une position stable pour l'erreur, mais comportant beaucoup de triangles aux formes allongées ; cela entraine qu'en ajoutant un nouveau point, les quadrilatères incluant les arêtes éventuelles à échanger seront très souvent non convexes).

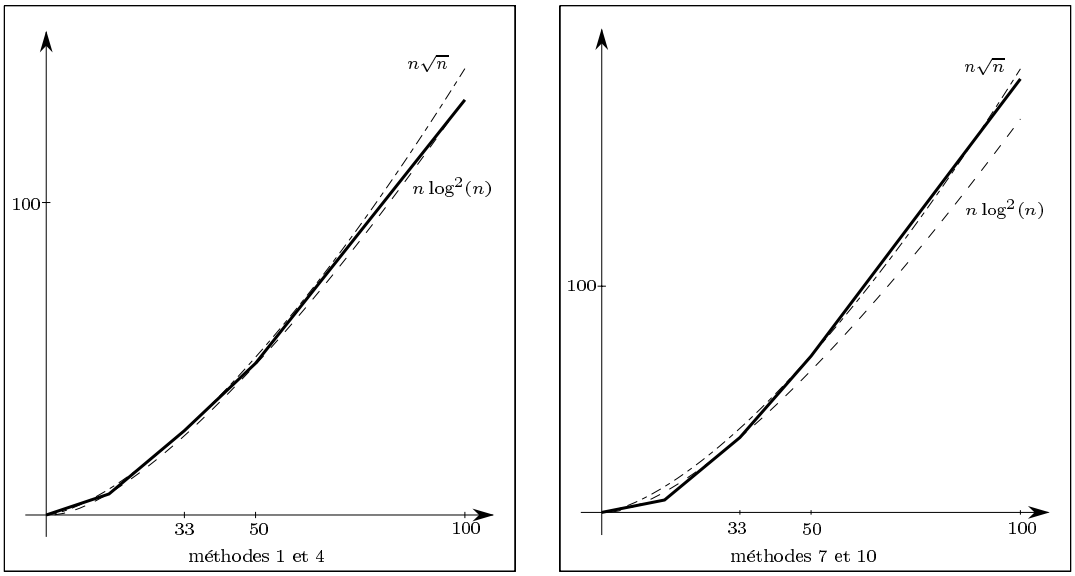
Finalement, l'ordre d'insertion semble jouer un rôle beaucoup plus important que le type d'algorithme, sur le nombre d'échanges. L'ordre aléatoire semble à nouveau

"bloquer" le processus de swapping (alors que l'on peut facilement s'imaginer qu'il devrait impliquer un grand nombre d'échanges, du fait de la non-régularité des triangles construits) ; ce nombre de swaps est plus faible encore pour l'ordre suivant la proximité à un coin. Mais ceci était attendu, dans la mesure où un point ajouté était toujours proche des triangles précédemment formés (qui sont alors plus réguliers).

5.2.4.2 Complexité des différentes méthodes

Comme il est plutôt difficile de se faire une idée de la progression du nombre de swaps en fonction du nombre de points, les figures 5.8 à 5.10 représentent les courbes de ces progressions accompagnées des courbes représentatives des fonctions (à une constante près) dont elles semblent se rapprocher. Les méthodes sont regroupées par deux, car le critère d'échange utilisé (erreur L_2 ou L_1) ne joue pas de rôle sur la quantité de swaps. Les constantes cachées sont de l'ordre de quelques unités. En abscisse se trouve le nombre de points et en ordonnée le nombre d'échanges. Pour les ensembles à 15 et 33 points, on a tenu compte des séries à respectivement 8 et 12 points sur l'enveloppe convexe (car, leurs résultats moyens sur les erreurs se rapprochaient plus de ceux observés pour les séries contenant plus de sommets).

Figure 5.7 – nombre d'échanges et complexité des méthodes ayant un ordre d'insertion aléatoire.



On peut déjà remarquer qu'aucune méthode ne comporte de progression en n^2, mais, dans une grande majorité des cas, il est difficile de déterminer exactement de quelle fonction elle est la plus proche. Cela vient de la taille trop modeste des plus gros ensembles testés (il aurait sans doute fallu essayer des ensembles comportant plusieurs milliers de points pour se faire une idée vraiment précise).

Cependant, il reste possible d'évaluer le comportement moyen des algorithmes :

– d'après la figure 5.10, les méthodes 13 et 14 (LAWSON à partir de Delaunay) ne semblent réaliser qu'un nombre presque linéaire d'échanges (en tous cas

Figure 5.8 – nombre d'échanges et complexité des méthodes insérant les points suivant l'ordre croissant des abscisses.

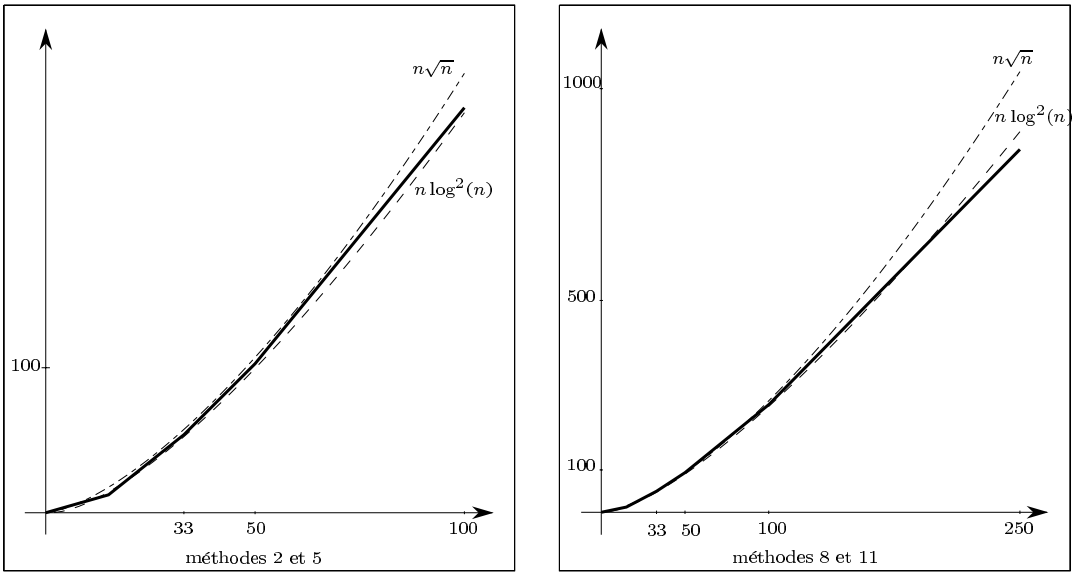


Figure 5.9 – nombre d'échanges et complexité des méthodes insérant les points par ordre de proximité au coin en bas à gauche.

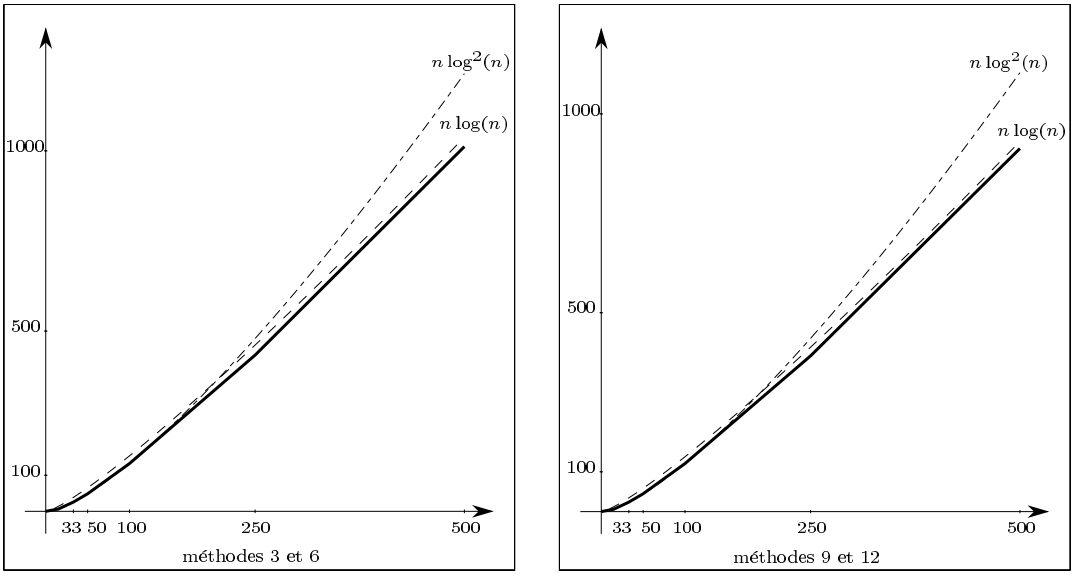
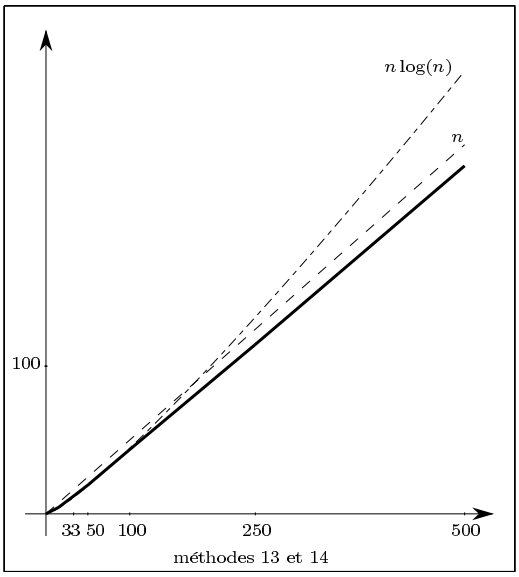


Figure 5.10 – nombre d'échanges et complexité des méthodes utilisant l'algorithme de LAWSON à partir de Delaunay.



inférieur à $n \log n$, qui représente la complexité nécessaire pour calculer la triangulation de Delaunay, point de départ des méthodes), on peut s'attendre à une complexité $O(n \log n)$.

– les méthodes 3, 6, 9 et 12 (ordre d'insertion suivant l'éloignement à un coin - figure 5.9) semblent très proches d'une courbe en $O(n \log n)$.

– pour les autres méthodes (figures 5.7 et 5.8), il est plus délicat d'apporter une réelle conclusion car les plus gros ensembles testés avaient été limités à 250, voire 100 points ; les courbes laissent cependant supposer qu'il y a moins de $O(n\sqrt{n})$ échanges effectués.

Finalement, les méthodes utilisant l'algorithme de type TOI ont un nombre total d'échanges de complexité inférieure à $O(n^2)$; comme à chaque insertion, on a au moins besoin de parcourir une fois toute la liste des nouvelles arêtes, leur complexité totale sera au mieux quadratique, au pire cubique.

Toutes les méthodes à algorithme de type LAWSON nécessitent une phase de prétraitement en $O(n \log n)$ (construction de la triangulation initiale) ; le nombre d'échanges étant toujours inférieur à $O(n\sqrt{n})$, leur complexité est de $O(n\sqrt{n})$ si l'ordre d'insertion est aléatoire et si la triangulation initiale est étoilée, $O(n \log n)$ sinon.

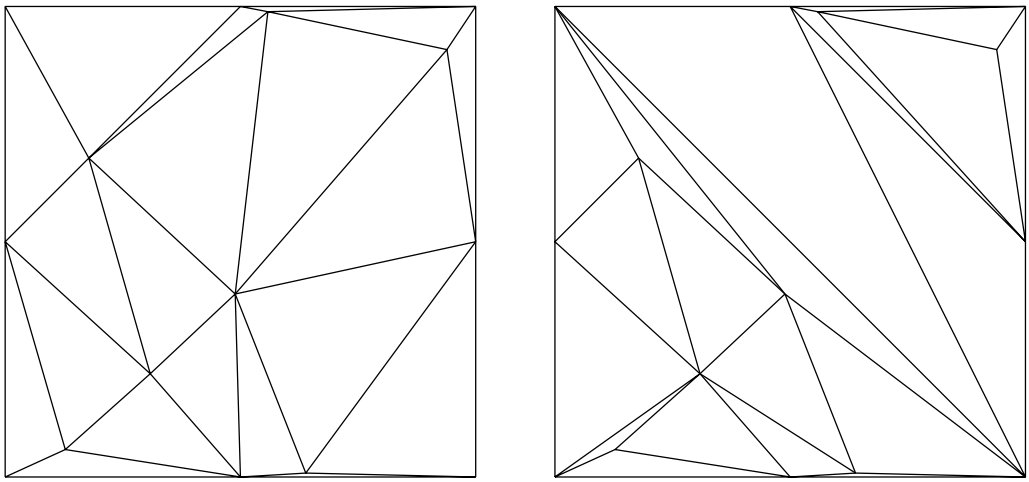
5.2.5 Observation des triangulations

Différents exemples de triangulations obtenues sont présentés sur les figures 5.11 à 5.16 ; sur chaque figure, la triangulation de gauche est celle de Delaunay, et les commentaires ne concernent que les erreurs L_2.

Tout d'abord, des exemples où les méthodes produisent vraiment de mauvais résultats :

- figure 5.11 : la deuxième triangulation est obtenue par la méthode 9 (qui, en moyenne, donne pourtant de bons résultats) sur un ensemble à 15 points, et renvoie une erreur 87% plus mauvaise que celle de Delaunay.
- figure 5.12 : la seconde triangulation provient de la méthode 3, la moins mauvaises des méthodes utilisant l'algorithme de type TOI. Son erreur est 40% supérieure (en valeur) à celle de la triangulation de Delaunay. On peut remarquer la présence de deux gros triangles qui peuvent expliquer sa valeur.

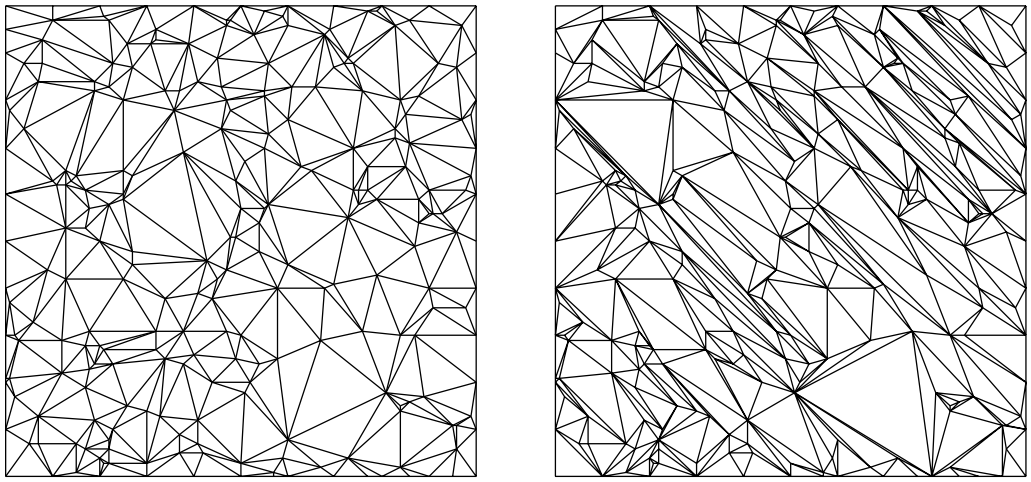
Figure 5.11 – mauvais exemple à 15 points : la triangulation localement optimale (à droite) rend une erreur presque 2 fois plus importante que celle de Delaunay.



Puis des exemples où il a pu être observé de fortes améliorations de l'erreur L_2 :

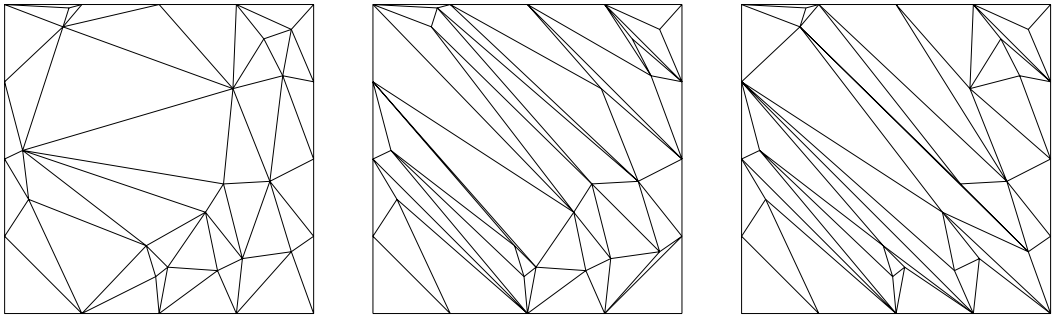
- figure 5.13 : sur cet ensemble à 33 points, la triangulation de Delaunay, particulièrement mauvaise (l'erreur L_2 est plus de 2 fois supérieure à la moyenne obtenue sur ce type d'ensembles), peut-être améliorée de 62% et 60% par les méthodes 9 et 10. On peut remarquer que les triangles ont alors une très forte tendance à être dirigés suivant les bissectrices des axes (direction des familles génératrices du paraboloïde hyperbolique).
- figure 5.14 : sur cet exemple à 100 points, la méthode 14 fournit une triangulation 43% meilleure que Delaunay (dessin du milieu), tandis que l'amélioration passe à 53% pour la méthode 1. Cependant, cette deuxième triangulation contient un grand nombre de triangles très allongés, propriété peu appréciée par certaines applications.
- les figures 5.15 et 5.16 présentent des triangulations d'ensembles de taille importante (respectivement 250 et 500) dont les améliorations ne sont pas négligeables (34% et 28%). On peut à nouveau remarquer que les triangles, par

Figure 5.12 – mauvais exemple à 250 points : la triangulation localement optimale (à droite) rend une erreur 40% plus importante que celle de Delaunay.



rapport à Delaunay, sont beaucoup plus dirigés suivant les bissectrices des axes. Ces deux triangulations ont été calculée par la méthode 13.

Figure 5.13 – bon exemple à 33 points : les triangulations localement optimales, bien que très différentes, sont plus de 2 fois meilleures que la triangulation de Delaunay (à gauche).



5.2.6 Conclusion

Il a semblé utile de résumer finalement les principales observations formulées tout au long de ce chapitre :

– les meilleures triangulations localement optimales sont, en moyenne, obtenues par utilisation de l'algorithme du type LAWSON en partant de la triangulation de Delaunay avec pour critère d'échange celui qui correspond à l'erreur que

Figure 5.14 – bon exemple à 100 points : les deux triangulations localement optimales ont une erreur à peu près 2 fois moins importante que celle de Delaunay (à gauche).

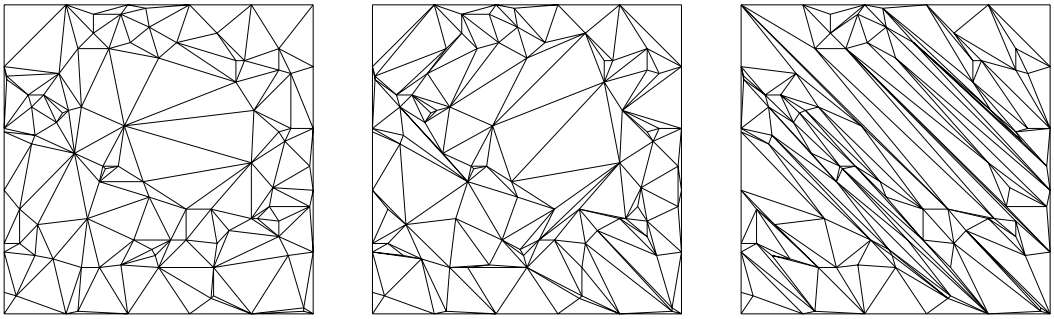


Figure 5.15 – bon exemple à 250 points : la triangulation localement optimale (à droite) a une erreur L_2 35% inférieure à celle de Delaunay.

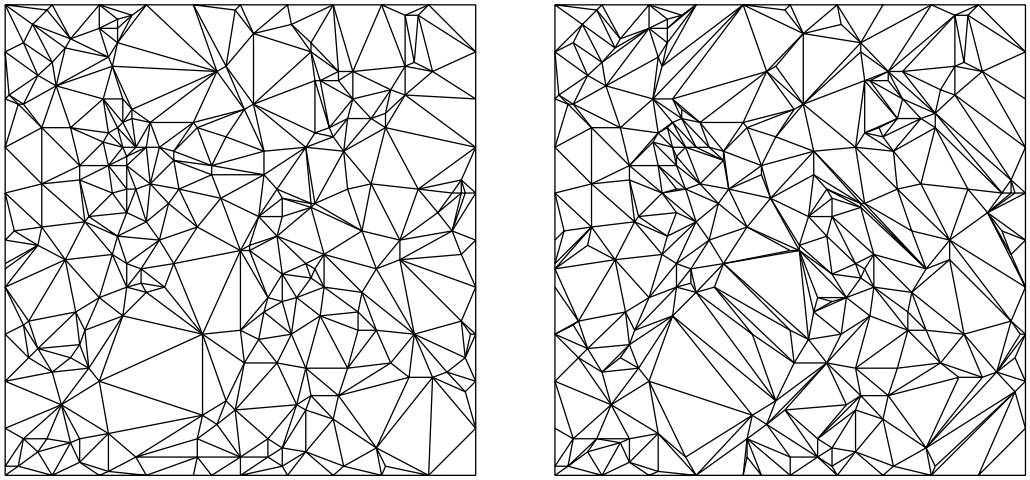
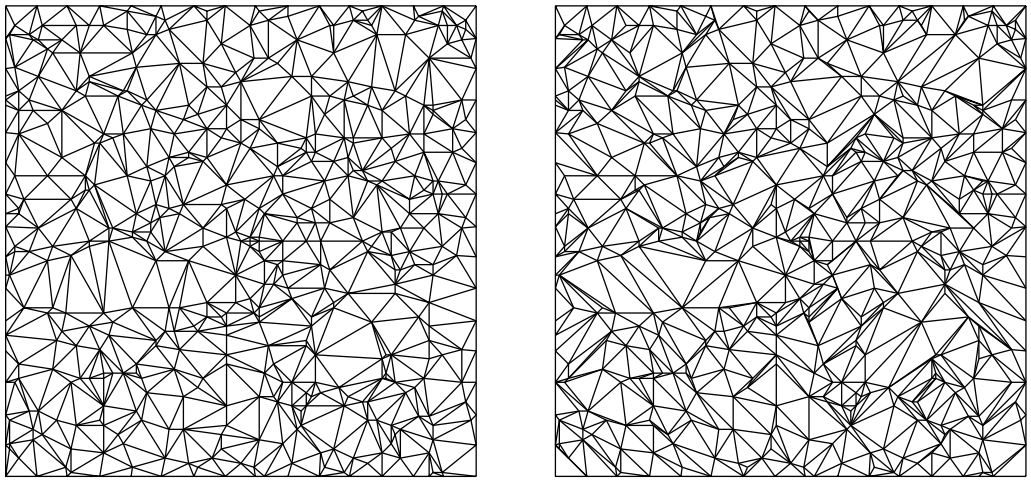


Figure 5.16 – bon exemple à 500 points : la triangulation obtenue par la méthode 13 (à droite) renvoie une erreur 28% plus petite que celle de Delaunay.



l'on veut minimiser. Il faut cependant noter que ce n'est la même méthode qui fournit la meilleure triangulation pour toutes les configurations de points.

- l'amélioration moyenne observée par rapport à la triangulation de Delaunay est de l'ordre de 20%, dès lors que le nombre de points est suffisamment important. Cette amélioration est d'ailleurs liée au nombre de sommets choisis sur l'enveloppe convexe de l'ensemble de points.

- les erreurs moyennes d'approximation diminuent proportionnellement avec l'augmentation du nombre de sommets des triangulations.

- il y a finalement peu de différences entre les triangulations obtenues par utilisation des critères d'optimisation de l'erreur L_1 ou de l'erreur L_2.

- le nombre d'échanges effectués est beaucoup plus lié à l'ordre d'insertion des sommets qu'à l'algorithme utilisé, lorsque la triangulation initiale est étoilée.

Chapitre 6

Algorithmes exacts

Après avoir pu observer le comportement des algorithmes de triangulations localements optimales, il serait intéressant de savoir si leurs résultats sont bons, c'est-à-dire s'ils sont éloignés ou pas de l'optimal global. Évidemment, il est très difficile, à part dans le cas de Delaunay, de construire une triangulation globalement optimale, et il n'existe pas de méthode simple qui soit capable de le faire en temps raisonnable. Nous allons voir cependant qu'en ayant recours à 2 algorithmes très différents, il est possible d'apporter certains éléments de réponses pour la question de l'éloignement de l'optimal global :

- le *recuit simulé*, bien qu'étant assez lent et ne permettant de savoir exactement si l'optimal trouvé est le bon, a permis de faire des essais sur quelques ensembles de points suffisamment importants (33 et 100 points).
- une méthode fondée sur les propriétés des triangulations *localement optimales*, directement adaptée d'un algorithme de DICKERSON et MONTAGUE [DM96], permet de trouver, de manière sure et rapide, les optimaux pour des ensembles de petite taille (15 points), et ainsi de tirer de nouvelles conclusions.

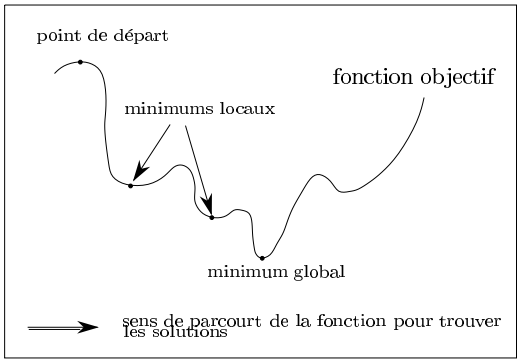
6.1 Recuit simulé

6.1.1 Théorie

6.1.1.1 Présentation générale

L'idée générale d'un algorithme de recuit simulé réside dans la possibilité qu'un extremum d'une fonction peut seulement être local (comme dans l'exemple de la figure 6.1). Pour arriver à l'extremum global de notre fonction, il ne faudra pas s'arrêter, mais accepter de perturber légèrement la solution à laquelle on a abouti, afin d'espérer pouvoir retomber un peu après sur un "meilleur" extremum.

Figure 6.1 – le premier minimum local rencontré n'est pas forcément le meilleur...



Historiquement, le recuit a d'abord été utilisé par les métallurgistes qui cherchaient à faire disparaître progressivement les défauts cristallins d'un métal, en réchauffant et en laissant refroidir progressivement le métal ; ceci est bien expliqué dans la thèse de TALON [Tal89]. C'est dans les années 1950 que MÉTROPOLIS a proposé une méthode du type Monte-Carlo pour simuler l'évolution de l'état du métal jusqu'à son équilibre thermique.

Puis, en 1983, KIRKPATRICK, GELATT et VECCHI [KGV83] ont appliqué ce principe issu de la physique statistique à certains problèmes d'optimisation. Ils proposent un algorithme très général qui peut s'énoncer de la manière suivante :

- **initialisations** :
  - soit f une fonction que l'on cherche à minimiser.
  - soit t un paramètre de valeur initiale très élevée (cela afin de pouvoir accéder à tous les états possibles).
  - soit s_i une solution réalisable.

- **tant que** t est suffisamment grand
  tant que le processus n'est pas stable
  on **calcule** une solution possible s_a voisine de s_i.
  $\Delta = f(s_a) - f(s_i)$.
  si $\Delta < 0$ **alors**
  s_a est un meilleur minimum : $s_i \leftarrow s_a$.

sinon
on **tire** un nombre aléatoire θ entre 0 et 1.
si $\theta \leq e^{-\Delta/t}$ **alors**
$s_i \leftarrow s_a$ (on décide de repartir de s_a même si c'est une moins bonne solution que s_i).
fin si
fin si
fin tant que
on **diminue** t.

fin tant que

Cet algorithme très général demande à être adapté à chaque problème d'optimisation :

- le paramètre t représente la température du problème physique ; on doit pouvoir définir sa valeur initiale ainsi que son processus de décroissance. La valeur initiale est essentielle : elle doit permettre d'atteindre le plus possible de solutions (même si elles sont défavorables). La manière de diminuer la température est elle aussi importante : si on l'abaisse trop rapidement, on a toutes les chances de se retrouver bloqué dans un état qui sera stable, mais qui ne sera pas un optimum global.

- on a également besoin de définir une notion de voisinage pour les solutions ; cela implique la nécessité de disposer d'opérations élémentaires qui permettent de passer d'une solution à une autre. Deux solutions seront voisines si on n'a besoin que d'une seule opération élémentaire pour transformer l'une en l'autre.

- lors de la comparaison de la solution courante s_i et de sa voisine s_a, il est évident que si $f(s_a)$ a une valeur inférieure à celle de $f(s_i)$, on remplacera s_i par s_a puisqu'il y aura amélioration de la fonction objectif f. Si on tombe dans une situation contraire, on décidera d'accepter le changement avec une probabilité qui dépend de la température ; à température faible, seuls des changements peu dégradants seront acceptés.

En procédant de la sorte, on espère éviter de rester bloqué dans des extremums peu profonds. Il est alors clair que les performances d'un algorithme de recuit simulé sont intimement liées au choix des opérations élémentaires, et à la vitesse à laquelle on fait décroître la température. Par ailleurs, la vitesse de convergence de l'algorithme sera fonction de cette vitesse de décroissance : c'est donc elle qui nous permettra de controler (de manière peu précise, certes) le temps d'exécution de l'algorithme.

Quant au bien fondé des performances du recuit simulé, en lisant l'article de KIRKPATRICK, GELATT et VECCHI [KGV83], on peut s'apercevoir sur le problème du voyageur de commerce, qu'avec des opérations élémentaires judicieuses, ils obtenaient de meilleurs résultats que la plupart des heuristiques connues (mais, évidemment, avec un temps de calcul plus important).

6.1.1.2 Adaptation au problème des triangulations planaires

Voyant là apparemment un bon moyen de trouver un optimal, SCHUMAKER a proposé un algorithme de recuit simulé adapté aux problèmes de recherches de triangulations optimales [Sch93]. En effet, en partant de considérations locales, on sait qu'il n'existe guère que la triangulation de Delaunay qui va aboutir à un optimum global pour certains critères (comme la maximisation du plus petit angle) ; on sait trouver assez facilement des extremums locaux, mais pas l'extremum global. Par ailleurs, étant données deux triangulations $\mathcal{T}$ et $\mathcal{T}'$ de l'enveloppe convexe d'un même ensemble de points, on sait que l'on peu toujours passer de $\mathcal{T}$ à $\mathcal{T}'$ par une séquence finie d'échanges de diagonales bien choisis (même si la bonne séquence est difficile à trouver) ; cela provient en partie du fait que le nombre de triangulations possibles est fini, et que le graphe des triangulations ne comporte qu'une seule composante connexe (voir, par exemple, [HN96, HNU96]).

Le swap apparait ainsi comme une opération élémentaire de voisinage bien choisie, et l'algorithme de recuit simulé proposé utilise un critère c pour décider parmi deux triangulations laquelle est la meilleure. On a également besoin d'entiers positifs $temp_{lim}$, $swap_{lim}$ et $bons_{lim}$ ainsi que d'une série de réels $t_1 > t_2 > \ldots > t_{temp_{lim}} > 0$ qui seront les températures choisies à chaque itération de l'algorithme ; un "bon" échange sera un échange qui permettra d'aboutir à une meilleure triangulation (relativement au critère choisi c). Le procédé général peut être décrit de la manière suivante (on suppose qu'en entrant ici, on dispose d'une triangulation initiale $\mathcal{T}$) :

pour $k = 1$ à $temp_{lim}$

nombre de bons swaps $\leftarrow 0$

pour $l = 1$ à $swap_{lim}$

si nombre de bons swaps $\leq bons_{lim}$ **alors**
on choisit aléatoirement une arête a de la triangulation.
si l'arête a peut être swappée **alors**
soit $\mathcal{T}$ la triangulation courante et $\mathcal{T}'$ celle obtenue en swappant a dans $\mathcal{T}$
si $d = c(\mathcal{T}, \mathcal{T}') < 0$ **alors**
on effectue l'échange
sinon
on **choisit** un nombre aléatoire $0 \leq \theta \leq 1$
si $\theta \leq e^{-d/t_k}$ **alors** on effectue l'échange **fin si**
fin si
fin si
fin si

fin pour

fin pour

Dans cet algorithme, $temp_{lim}$ controle le nombre d'étapes de l'algorithme ; les diverses températures choisies t_i vont permettre d'ajuster la probabilité que des mauvais échanges soient effectués. En pratique, les nombres aléatoires θ sont choisis

pour une distribution uniforme de $[0,1]$, ce qui implique que, pour un nombre d donné, la probabilité de faire un mauvais swap est donnée par $P(\theta \leq e^{-d/t_k}) = e^{-d/t_k}$.

Le paramètre $swap_{lim}$ controle le nombre d'échanges à effectuer à chaque étape, et $bons_{lim}$ controle le nombre de bons swaps à faire pour chaque température. L'idée est de ne pas faire rapidement trop de bons échanges, sinon on risque de se retrouver coincé autour d'un minimum seulement local et dans une position relativement stable.

Bien que le recuit simulé soit un bon moyen de chercher une triangulation globalement optimale, on ne peut pas garantir qu'il en trouvera une. En effet, il n'existe aucun moyen simple de déterminer si une triangulation localement optimale l'est aussi globalement. Le recuit est donc une méthode très empirique, qui nécessite de bien choisir ses paramètres, exécuter le programme, et si le résultat n'est pas satisfaisant, recommencer avec des paramètres ajustés. Dans son article [Sch93], SCHUMAKER propose un choix intéressant de paramètres, fondé sur ses observations de divers critères d'optimisation (minimisation du plus grand angle, interpolation dépendante des données, moindres carrés dépendants des données, minimisation lexicographique des plus grands angles) :

- pour la valeur initiale t_0 de la température, son choix se porte sur une valeur environ deux fois supérieure au meilleure changement provoqué par un bon swap dans la triangulation de départ.

- pour les températures suivantes, il propose de prendre $t_k = r^k t_0$ où $0 < r < 1$ est fixé, avec une préférence pour des valeurs situées entre 0.9 et 0.95.

- il choisit $swap_{lim}$ et $bons_{lim}$ égaux à des multiples (fixés) du nombre d'arêtes de la triangulation, avec des préférences pour des facteurs 5 ou 10 (qui semblent mieux adapter).

Dans la pratique, il arrive que l'algorithme de recuit arrête d'effectuer des échanges : on dit dans ce cas qu'il a *convergé*. Si la température initiale est élevée, il faudra faire tourner le programme longtemps avant qu'il ne converge. D'un autre côté, une grosse température initiale accompagnée de fortes valeurs pour $swap_{lim}$ et $bons_{lim}$ permet d'explorer un plus grande nombre de triangulations, augmentant ainsi nos chances de trouver l'optimum global. Si la température initiale est faible, l'algorithme convergera en général très rapidement, mais vers un minimum seulement local.

Un algorithme de recuit simulé pour les triangulations $3D$ est présenté dans la thèse de TALON [Tal89] ; l'auteur a adapté ici la technique du recuit essentiellement pour des maillages de zones polyédriques, en remarquant qu'un raffinement local des paramètres donnait des résultats intéressants. Cependant, l'introduction de ces recuits locaux complique pas mal la lecture de l'algorithme général ; c'est en partie pour cette raison qu'il ne sera pas décrit ici, mais aussi parce que ce qui nous intéresse directement sont les triangulations $2D$.

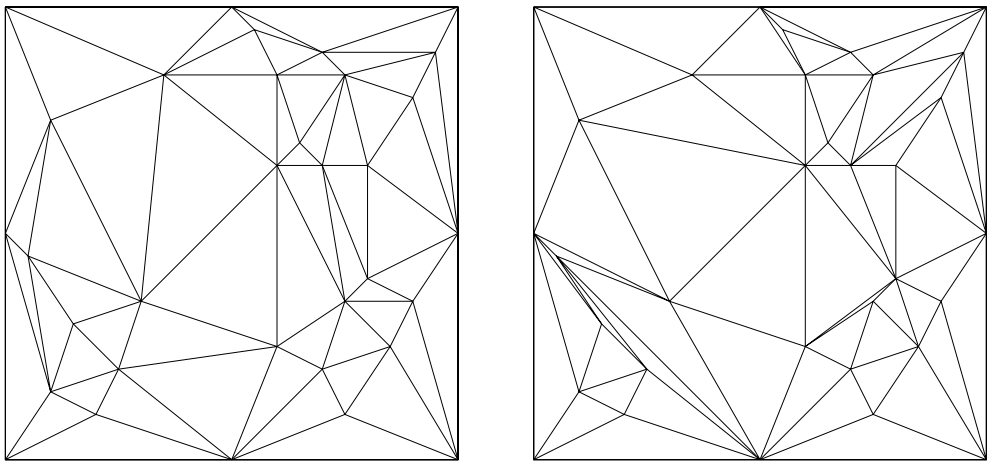
6.1.2 Application

6.1.2.1 Algorithme

Triangulations de départ

Un algorithme de recuit a besoin de partir d'une solution initiale, qu'elle soit bonne ou pas. Le mieux serait sans doute, d'ailleurs, que cette solution ne soit pas trop bonne ; sinon, cela impliquerait qu'aux premières températures, peu de mauvaises décisions pourraient être prises. C'est pourquoi, la triangulation de Delaunay a d'abord été choisie comme point de départ. Pour tester la sensibilité du programme au choix de départ, une triangulation localement optimale, qui définit en principe une solution stable, a aussi été utilisée dans un deuxième temps.

Figure 6.2 – les triangulations initiales choisies pour l'ensemble de FRANKE à 33 points



triangulation de Delaunay — triangulation localement optimale

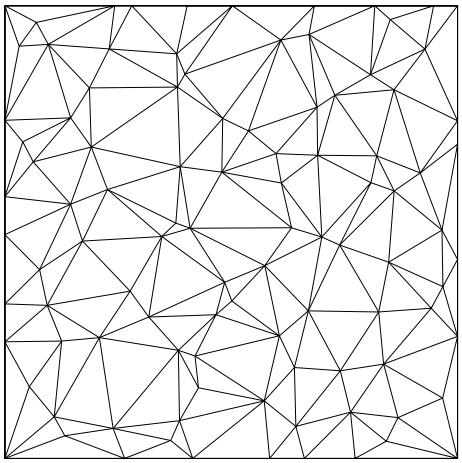
Température initiale

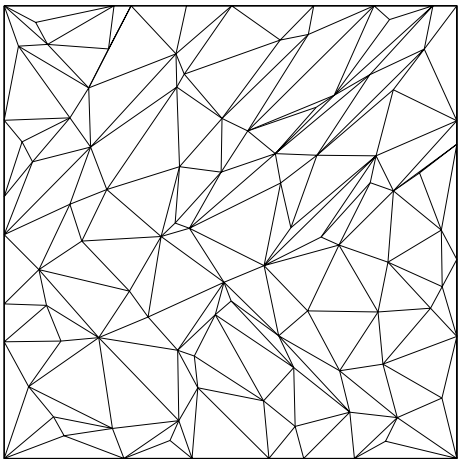
Pour la température initiale, il a été fait confiance aux choix dictés par les expériences de SCHUMAKER : elle a toujours été fixée comme égale à un ordre de grandeur de la valeur maximale d'un échange possible dans la triangulation de Delaunay (choix impossible pour une triangulation localement optimale puisque, par définition, aucun bon swap n'y est possible).

Choix des températures suivantes

Afin d'augmenter le contrôle sur les températures $t_1, \ldots, t_{temp_{lim}}$, plusieurs choix ont été réalisés pour les définir ; cependant, les températures successives sont toujours données sous forme de fonctions par paliers (on a un saut entre 2 températures consécutives) et sont toujours égales au produit d'un facteur par la température initiale, c'est-à-dire que, pour tout entier k ($k \in 1, \ldots, t_{temp_{lim}}$), on aura $t_k = f(k)\, t_0$.

Figure 6.3 – les triangulations initiales choisies pour l'ensemble de FRANKE à 100 points


triangulation de Delaunay


triangulation localement optimale

Ces fonctions f testées sont regroupées en 2 catégories :

- des fonctions dépendantes d'un facteur α à définir, qui sera lié à la vitesse de convergence souhaitée. Parmi elles, on a :
  - la fonction géométrique standard donnée par SCHUMAKER, $f(k) = \alpha^k$.
  - la fonction inverse $f(k) = \frac{1}{k^\alpha}$.
  - la fonction exponentielle $f(k) = e^{-k^\alpha}$.
- des fonctions uniquement liées à l'entier k :
  - la fonction inverse $f(k) = \frac{1}{k}$.
  - la fonction exponentielle $f(k) = e^{-k}$.
  - la fonction logarithme $f(k) = \frac{1}{\ln(k)}$

Le choix des fonctions f va fortement influencer le comportement de l'algorithme de recuit simulé, et plus particulièrement sa vitesse de convergence. Ainsi, parmi les fonctions liées seulement à k, la fonction exponentielle va impliquer une convergence extrèmement rapide ; en effet, comme elle donnera rapidement des valeurs très petites pour une augmentation légère de k, plus aucune modification ne sera possible pour l'extremum sur lequel on sera arrivé. Le phénomène sera opposé pour la fonction logarithme ; elle autorisera longtemps des modifications du même ordre (elle est très faiblement décroissante). Cela entrainera notamment que l'algorithme ne convergera pas dans un temps raisonnable, et que la solution à laquelle on parviendra sera sans aucun doute assez éloignée de l'optimum global recherché, chose que l'on ne peut pas affirmer pour la fonction exponentielle : comme celle-ci converge très vite, on a tout de même une (petite...) chance qu'elle aboutisse à l'extremum recherché.

De toutes les manières, ce deuxième groupe de fonctions a surtout été utile pour juger du comportement des fonctions du premier groupe. On peut en effet modifier le comportement de ces dernières, en ajustant correctement le paramètre α. Ainsi, il a été choisi, de manière certes un peu arbitraire, que la centième température devrait être environ 200 fois inférieure à la première ; cela implique que les modifications

imposées à la solution à ce moment-là auront un seuil de tolérance égal à 0.5% du premier, ce qui représente a priori une marge correcte pour trouver l'optimal global en un peu plus de 100 températures testées.

Critères d'arrêt

Il est évident que l'on ne peut pas se permettre, même pour des ensembles de points de taille raisonnable, de s'arrêter seulement lorsqu'il y a convergence vers une solution stable, d'abord parce que cela pourrait impliquer un temps de calcul beaucoup trop grand, mais aussi parce que l'on ne serait même pas sûr d'avoir convergé vers l'optimal global.

Plusieurs critères d'arrêt ont donc semblé nécessaire : le programme s'arrêtera au bout d'un nombre prédéfini d'itération (une itération étant une température), ou lorsqu'il aura convergé. Par ailleurs, comme il a déjà été signalé, un nombre maximal d'échanges et de tests d'echanges est fixé pour chaque itération.

6.1.2.2 Résultats numériques

Tests mis en place

Les problèmes du recuit simulé sont qu'il est d'une part très difficile de bien controler la convergence de l'algorithme, et que, d'autre part, son exécution peut durer excessivement longtemps. Il a donc été décidé de ne tester que 4 ensembles de points :

- les 2 ensembles de FRANKE (figure 5.2) à 33 et 100 points.
- un ensemble de 33 points qui avait donné une très forte amélioration de l'erreur d'approximation L_2 par rapport à Delaunay, par application d'un algorithme d'optimisation locale.
- un ensemble de 33 points qui n'avait donné qu'une faible amélioration de l'erreur d'approximation L_2 par rapport à Delaunay, par application d'un algorithme d'optimisation locale.

Pour chacun des ensembles de FRANKE, les 6 fonctions de variation des températures décrites précédemment ont été testées, en partant de la triangulation de Delaunay, puis de la meilleure triangulation localement optimale. Pour les 2 autres ensembles, on s'est contenté d'une recherche d'un optimal en utilisant les observations effectuées pour toutes les méthodes.

Concernant les fonctions liées à un paramètre α, les choix pour ce paramètre ont été les suivants :

- pour la fonction géométrique, comme suggéré par SCHUMAKER, on a choisi des valeurs de α de 0.95, 0.925 ou 0.9 ; ces valeurs impliquent une température inférieure à un demi pourcent de t_0 pour les itérations respectives 104, 68 et 51.
- pour la fonction inverse, on a pris α égal à 1.1 ou 1.0574, valeurs qui donnent un k égal à 124 ou 150 pour avoir $t_k \leqslant 0.005\, t_0$.

– pour la fonction exponentielle, les valeurs de α sont 0.34 ou 0.35 qui entrainent des valeurs de k ($t_k \leqslant 0.005\, t_0$) égales à 135 ou 118.

Pour tous les tests, certaines données sont restées constantes :

– la température initiale a été de 0.0001 pour les ensembles à 33 points et 0.000001 pour celui à 100 points. Ces nombres proviennent des observations de SCHUMAKER qui suggéraient d'utiliser environ le double de la plus forte amélioration provoquée par un échange d'arête dans la triangulation initiale (ou, du moins, dans celle de Delaunay pour notre propos).

– à chaque itération, on a volontairement limité le nombre d'échanges réalisés (bons et mauvais), ainsi que le nombre d'arêtes testées, cela afin d'éviter, autant qu'il était possible, une convergence trop rapide. Ces facteurs limitatifs dépendent de l'itération en cours : plus elle est grande, plus ces nombres sont grands. Plus précisément, si n est le nombre total d'arêtes de la triangulation initiale, on a choisi pour facteur limitatif du nombre de tests $10\, n \ln(k)$, et pour facteur limitatif du nombre de bons swaps $5\, n \ln(k)$.

– on a fixé le nombre limite de températures à 200 ; bien que ce nombre puisse paraître peu élevé, la diversité des fonctions de variations pour les températures a toujours permis de se faire une idée précise du résultat recherché, le but n'étant pas d'étudier la convergence d'un algorithme de recuit simulé, mais bien de savoir si nos optimaux locaux étaient éloignés ou pas de l'optimal global.

Tests des ensembles de *Franke*

Les tableaux 6.1 à 6.4 contiennent les optimaux obtenus pour les ensembles de FRANKE décrits dans le chapitre précédent ; 2 batteries d'essais ont été effectuées, l'une en partant de la triangulation de Delaunay, que l'on sait assez éloignée de l'optimal recherché, l'autre en utilisant une "bonne" triangulation localement optimale comme point de départ. Quelques variations ont été apliquées aux α choisis, mais en principe suffisamment peu importantes pour ne pas trop influencer le comportement de l'algorithme.

La lecture de ces tableaux se fait de la manière suivante :

– la première colonne donne la fonction de variation des températures utilisée.

– la deuxième colonne indique la valeur éventuelle du paramètre α.

– la troisième renvoie la valeur de la meilleure triangulation trouvée (qu'il y ait eu convergence ou pas).

– les colonnes suivantes indiquent respectivement :
  - la convergence ou pas de l'algorithme au bout de 200 itérations.
  - la convergence ou pas du programme vers le meilleur optimal qu'il a trouvé.
  - le nombre de températures testées.
  - le nombre d'optimaux successifs trouvés.

Tableau 6.1 – résultats pour l'ensemble à 33 points de FRANKE, en commençant avec la triangulation de Delaunay.

fct	α	optimal	cv	cv op	itér	nb op	k op	k cv
géom.	0.9	0.0176578334	oui	oui	200	50	80	56
inv.	1.1	0.0176332473	non	non	200	27	81	–
expo.	0.35	0.0176332473	non	non	200	51	135	–
inv.	–	0.0176581000	non	non	200	53	177	–
expo.	–	0.0176581000	oui	oui	75	53	8	9
log.	–	0.0178537798	non	non	200	34	172	–

Tableau 6.2 – résultats pour l'ensemble à 33 points de FRANKE, en commençant avec la meilleure triangulation localement optimale.

fct	α	optimal	cv	cv op	itér	nb op	k op	k cv
géom.	0.925	0.0176332473	oui	oui	200	47	73	107
inv.	1.0574	0.0176578345	non	non	200	38	143	–
expo.	0.34	0.0176332471	non	non	200	63	102	–
inv.	–	0.0180413775	non	non	200	46	171	–
expo.	–	0.0176740644	oui	non	27	42	6	9
log.	–	0.0178541298	non	non	200	25	139	–

Tableau 6.3 – résultats pour l'ensemble à 100 points de FRANKE, en commençant avec la triangulation de Delaunay.

fct	α	optimal	cv	cv op	itér	nb op	k op	k cv
géom.	0.925	0.00300827569	oui	oui	200	132	87	133
inv.	1.0574	0.00300878844	non	non	200	132	200	–
expo.	0.34	0.00300721378	non	non	200	168	191	–
inv.	–	0.00302890451	non	non	200	140	193	–
expo.	–	0.00305846056	oui	non	12	113	5	11
log.	–	0.00329644102	non	non	200	13	200	–

Tableau 6.4 – résultats pour l'ensemble à 100 points de FRANKE, en commençant avec la meilleure triangulation localement optimale.

fct	α	optimal	cv	cv op	itér	nb op	k op	k cv
géom.	0.95	0.00301494278	non	non	200	71	103	–
inv.	1.0574	0.00302528980	non	non	200	75	156	–
expo.	0.34	0.00301346852	non	non	200	84	177	–
inv.	–	0.00304086998	non	non	200	62	155	–
expo.	–	0.00308409291	oui	oui	14	53	8	13
log.	–	0.00313186527	non	non	200	0	–	–

- l'itération à laquelle a été rencontré le dernier optimal.
- l'itération éventuelle à partir de laquelle on a pû observer une convergence de l'algorithme.

On peut évidemment remarquer que l'emploi des données suggérées par SCHUMAKER permet une convergence systématique, mais cependant pas autour de l'optimal global recherché ; le minimum atteint est peu éloigné du meilleur trouvé (10^{-10} pour l'ensemble à 33 points, 10^{-6} pour celui à 100). Cette convergence vers un autre optimal peut, sans doute, s'expliquer par l'existence d'un certain nombre de triangulations stables (parmi toutes celles possibles) pour le critère choisi.

Comme on pouvait s'y attendre, l'utilisation de la fonction exponentielle sans paramètre entraine une convergence du recuit excessivement rapide, mais vers une solution peu fiable, car trop rapidement peu modifiable. D'autre part, les fonctions inverse sans paramètre et logarithme donne une variation des températures beaucoup trop faible pour permettre au programme de converger en 200 itérations seulement ; ceci explique que des meilleurs optimaux soient découverts très tard dans l'exécution, et qu'ils soient souvent très éloignés du résultat recherché.

En revanche, toutes les méthodes utilisant un paramètre variable α donnent satisfaction, même si au bout des 200 itérations, elles n'ont pas tout à fait convergé vers une solution stable (surtout si on travaille avec 100 points). Il faut dire que la valeur du paramètre avait été bien sûr choisie pour que les températures décroissent de manière raisonnable.

Concernant le choix de la triangulation initiale, il est clair, lorsque l'on regarde ce qui se produit avec l'ensemble à 100 points, qu'il vaut mieux en général prendre la triangulation de Delaunay. En effet, en partant avec une bonne triangulation localement optimale, l'algorithme de recuit a beaucoup plus de mal à perturber la solution au départ et cela semble alors jouer un grand rôle sur la suite des événements : les solutions finales renvoyées ne sont pas très bonnes, et l'algorithme a plus de mal à converger (du moins, pour les 200 itérations demandées). On peut alors se demander pourquoi le phénomène est moins flagrant pour l'ensemble à 33 points. L'explication pourrait bien être la suivante : la meilleure triangulation localement

optimale pour l'ensemble à 33 points n'améliorait que de 5.2% l'erreur obtenue pour Delaunay, alors que pour 100 points, ce rapport passait à 9.1%. Il se peut ainsi que la triangulation initiale localement optimale choisie pour les 100 points de FRANKE soit trop stable ; cette supposition est renforcée par le fait que l'optimal obtenu par recuit n'améliore que de 4% la meilleure erreur localement optimale, contre 8.8% pour les 33 points (résultats donnés dans le tableau 6.5).

Considérations des améliorations

Le but de l'utilisation du recuit simulé était de savoir si les résultats obtenus en ayant recours aux échanges locaux étaient éloignés de la véritable solution globale. Il est donc intéressant de pouvoir comparer les erreurs respectives de chaque méthode ; c'est ce que fait le tableau 6.5 où sont données les erreurs obtenues pour les 4 ensembles de points, ainsi que les pourcentages d'amélioration entre la triangulation de Delaunay et la meilleure triangulation localement optimale, puis entre celle-ci et la triangulation optimale donnée par recuit simulé. Ont été testés les 2 ensembles de FRANKE ainsi que 2 autres ensembles à 33 points, un dont l'utilisation d'un algorithme d'optimisation locale avait beaucoup diminué l'erreur L_2, un autre pour lequel l'amélioration était insignifiante. Les triangulations correspondant aux erreurs pour ces 2 derniers ensembles se trouvent sur les figures 6.6 et 6.7.

Tableau 6.5 – erreurs et améliorations observées pour 4 ensembles de points.

ensemble	pts	Delaunay	optim. loc.	amél.	optim. rec.	amél.
FRANKE	33	0.02039931	0.01933545	5.22%	0.01763325	8.8%
FRANKE	100	0.00344547	0.00313187	9.10%	0.00300721	3.98%
bon	33	0.02296753	0.01278045	44.35%	0.01165776	8.78%
mauvais	33	0.01644922	0.01642360	0.16%	0.01641059	0.08%

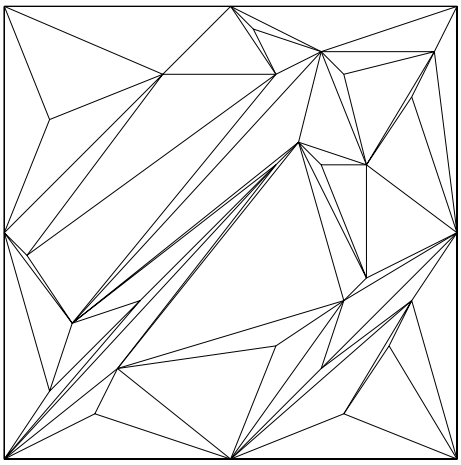
Il est difficile de donner des conclusions générales fondées sur des tests limités à 4 exemples. On peut cependant affirmer qu'il est très rare que les algorithmes d'optimisation locale aboutissent à l'optimum global (du moins, pour des ensembles à plus de 30 points), et que l'amélioration des erreurs est liée aux emplacements des points : ainsi, pour le dernier ensemble testé, la triangulation de Delaunay est une bonne approximation de l'optimum, alors que pour le troisième, elle en est très éloignée.

6.2 Triangulations localement optimales

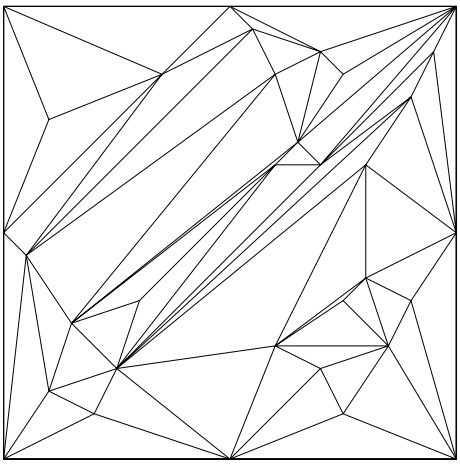
6.2.1 Théorie

Très récemment, DICKERSON et MONTAGUE [DM96] ont présenté un algorithme qui permet de calculer un sous-graphe important de la triangulation de poids minimum d'un ensemble de points ; cette méthode semble d'autant plus intéressante que,

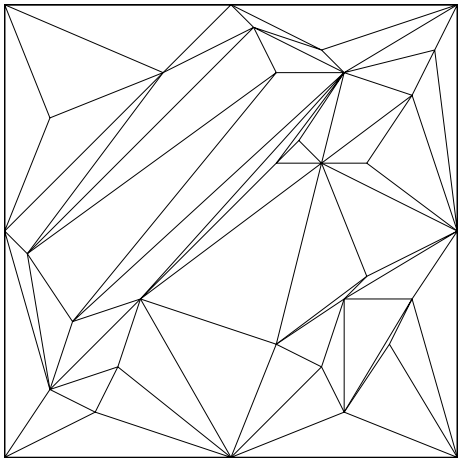
Figure 6.4 – quelques unes des triangulations optimales successives trouvées par l'algorithme de recuit simulé, appliqué à l'ensemble à 33 points de FRANKE.



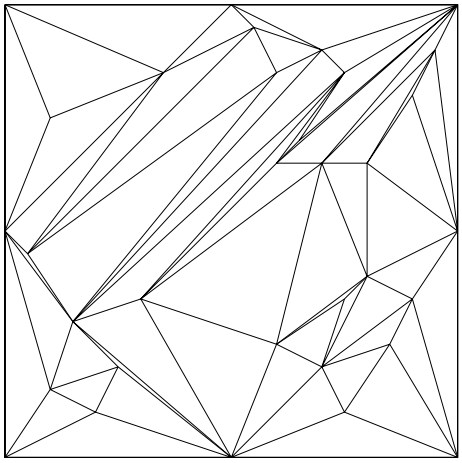
itération : 3 - optimal : 5 - erreur = 0.0190028



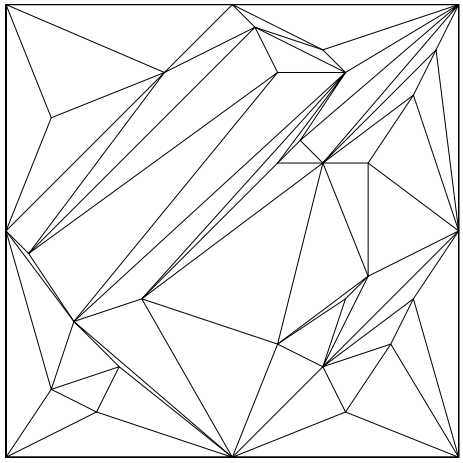
itération : 7 - optimal : 30 - erreur = 0.0180124



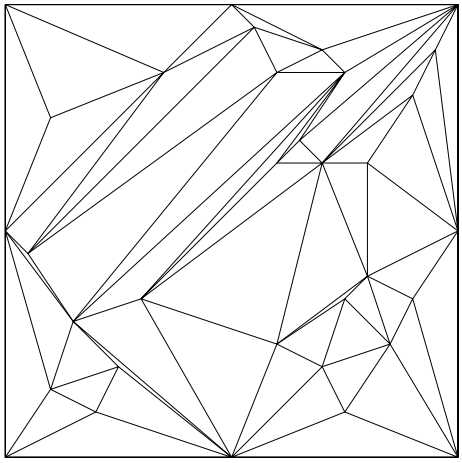
itération : 31 - optimal : 40 - erreur = 0.0177218



itération : 51 - optimal : 50 - erreur = 0.0176731

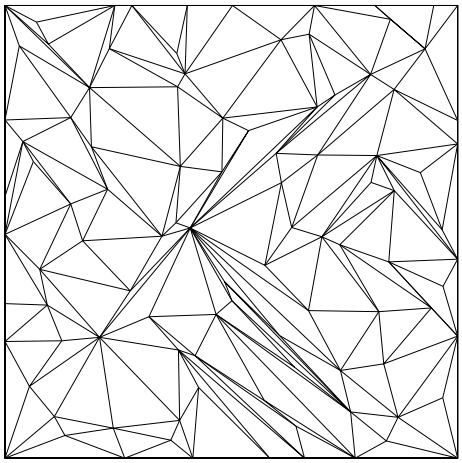


itération : 70 - optimal : 60 - erreur = 0.0176395

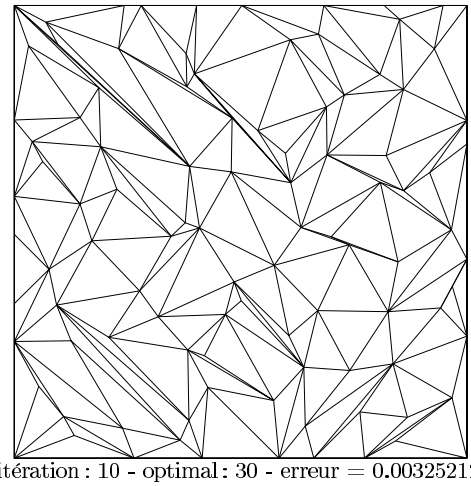


itération : 102 - optimal : 63 - erreur = 0.0176332

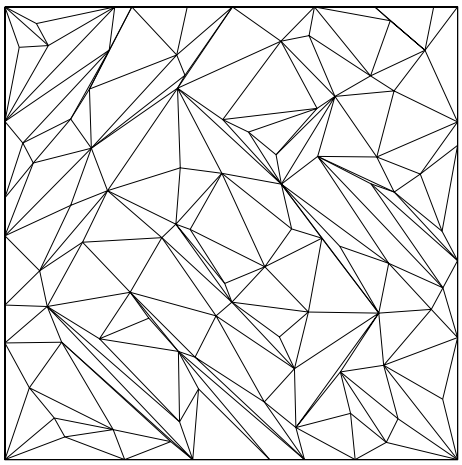
Figure 6.5 – quelques unes des triangulations optimales successives trouvées par l'algorithme de recuit simulé, appliqué à l'ensemble à 100 points de FRANKE.



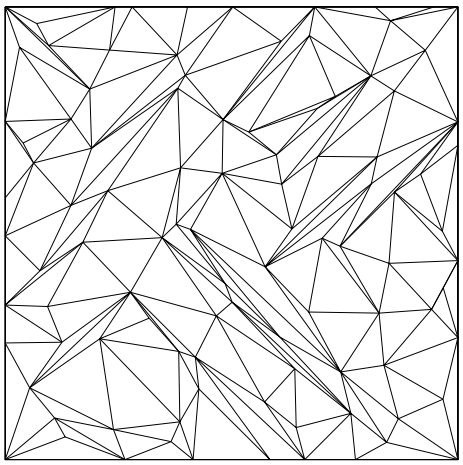
itération : 6 - optimal : 10 - erreur = 0.00338366



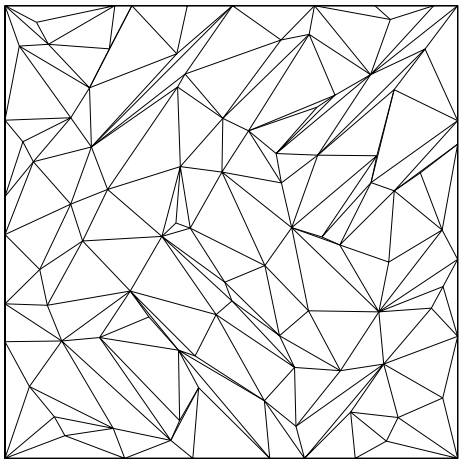
itération : 10 - optimal : 30 - erreur = 0.00325212



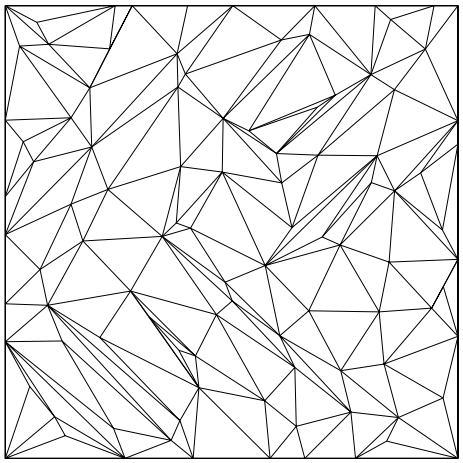
itération : 18 - optimal : 50 - erreur = 0.00319180



itération : 30 - optimal : 75 - erreur = 0.00310749

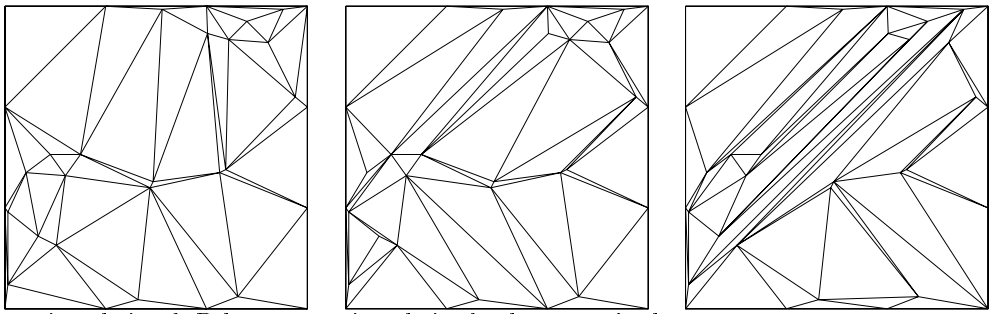


itération : 33 - optimal : 100 - erreur = 0.00305887



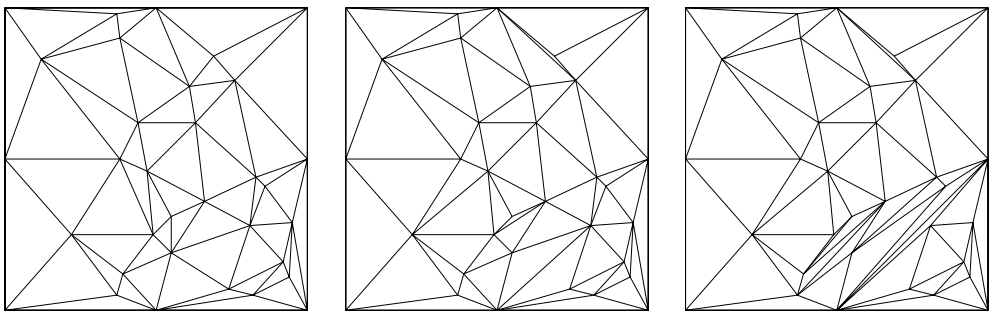
itération : 191 - optimal : 168 - erreur = 0.00300721

Figure 6.6 – 3 triangulations obtenues pour un ensemble à 33 points où l'application du critère d'optimalité locale donne une amélioration très importante.



triangulation de Delaunay | triangulation localement optimale | triangulation par recuit

Figure 6.7 – 3 triangulations obtenues pour un ensemble à 33 points où l'application du critère d'optimalité locale donne une amélioration négligeable.



triangulation de Delaunay | triangulation localement optimale | triangulation par recuit

dans leur cas, ils sont généralement parvenus à des cas où il ne leur reste plus que des polygones simples à trianguler, chose que l'on sait faire (pour le critère choisi) en temps polynomial ([Gil79] ou [Kli80]). Or, leur algorithme est lui-même polynomial, et jusqu'à présent, on ne connait pas la complexité du problème pour un ensemble général de points [GJ79] ; cela fait longtemps que l'on essaie de prouver que c'est un problème NP-dur.

6.2.1.1 Triangulations localement minimales

Dans toute triangulation, une arête a qui n'appartient pas à l'enveloppe convexe borde deux triangles vides. Si le quadrilatère formé de ces deux triangles est convexe, alors il possède une deuxième diagonale a' ; comme on le sait déjà, une triangulation est *localement minimale* si on ne peut jamais remplacer les arêtes a par les arêtes a'.

Ainsi, la triangulation de poids minimum doit être localement optimale ; sinon, il existerait une arête non minimale qui pourrait être swappée pour réduire le poids de la triangulation. Malheureusement, plusieurs triangulations localement optimales sont possibles pour un même ensemble de points, et la plupart ne sont pas globalement optimales ([DLR90b]).

À partir de maintenant, on désignera la triangulation de poids minimum (le poids considéré étant la distance euclidienne qui sépare les deux sommets d'une arête) d'un ensemble de points $\mathcal{S}$ par $MWT(\mathcal{S})$ (pour *Minimal Weight Triangulation*).

6.2.1.2 Quelques lemmes

Pour construire leur algorithme, les auteurs prouvent d'abord quelques petites propriétés :

observation 6.1 *Pour tout ensemble de points $\mathcal{S}$, une arête appartenant à toute triangulation localement optimale de $\mathcal{S}$ est dans $MWT(\mathcal{S})$, et une arête qui n'est dans aucune triangulation localement optimale de $\mathcal{S}$ n'appartient pas à $MWT(\mathcal{S})$.*

Cette observation induit une partition de l'ensemble de toutes les arêtes possibles de $\mathcal{S}$ en trois ensembles disjoints :

- celles qui sont dans *toutes* les triangulations localement minimales,
- celles qui ne sont dans *aucune* triangulation localement optimale,
- les autres, qui appartiennent à une ou *plusieurs* triangulations localement optimales.

lemme 6.1 *Soit a une arête de $\mathcal{S}$. Soient $\mathcal{T}_G(a)$ et $\mathcal{T}_D(a)$ les ensembles de triangles vides qui bordent a, sont d'un côté donné de a (gauche pour $\mathcal{T}_G$ et droit pour $\mathcal{T}_D$) et appartiennent au moins à une triangulation localement optimale. Si a est dans une quelconque triangulation localement optimale, alors il existe un $\Delta_1 \in \mathcal{T}_G(a)$ et un $\Delta_2 \in \mathcal{T}_D(a)$ tels que a soit localement optimale pour le quadrilatère formé par Δ_1 et Δ_2.*

lemme 6.2 *Soit a une arête de $\mathcal{S}$. S'il n'existe pas d'autre arête possible de $\mathcal{S}$ qui croise a et qui soit elle-même dans une triangulation localement optimale, alors a appartient à toutes les triangulations localement optimales de $\mathcal{S}$.*

6.2.1.3 Algorithme

L'algorithme permettant de calculer un sous-graphe de $MWT(\mathcal{S})$ découle directement des lemmes précédents ; il utilise trois listes :

- *candTris*, liste des triangles candidats,
- *candEdges*, liste des arêtes candidates,
- *edgesIn*, liste des arêtes qui sont dans toute triangulation localement optimale.

Cet algorithme est donc :

- **inititialisations** :
  $candTris \leftarrow$ liste de tous les triangles vides de $\mathcal{S}$.
  $candEdges \leftarrow$ liste de toutes les arêtes de $\mathcal{S}$.
  $edgesIn \leftarrow$ liste vide.

- on **enlève** de *candEdges* les arêtes de l'enveloppe convexe et on les ajoute à $edgesIn$.

- **pour** toute arête $a \in candEdges$
  si il n'existe pas de triangles $\Delta_i \in \mathcal{T}_G(a)$ et $\Delta_j \in \mathcal{T}_D(a)$ tels que a soit localement optimale pour le quadrilatère $\Delta_i\Delta_j$ **alors**
  on **supprime** a de *candEdges*.
  on **supprime** tous les triangles contenant a de $candTris$.
  fin si
  si a ne coupe aucune autre arête de *candEdges* ou de $edgesIn$ **alors**
  on **ajoute** a à $edgesIn$.
  fin si

  fin pour

Complexité

La boucle de fin est itérée $O(n^2)$ fois (nombre possible d'arêtes de $\mathcal{S}$) ; comme chaque arête peut avoir un nombre linéaire de triangles de chaque côté, cela implique $O(n^2)$ tests de combinaisons de triangles adjacents. Soit, une complexité totale de $O(n^4)$, les autres étapes nécessitant un temps inférieur.

Améliorations possibles

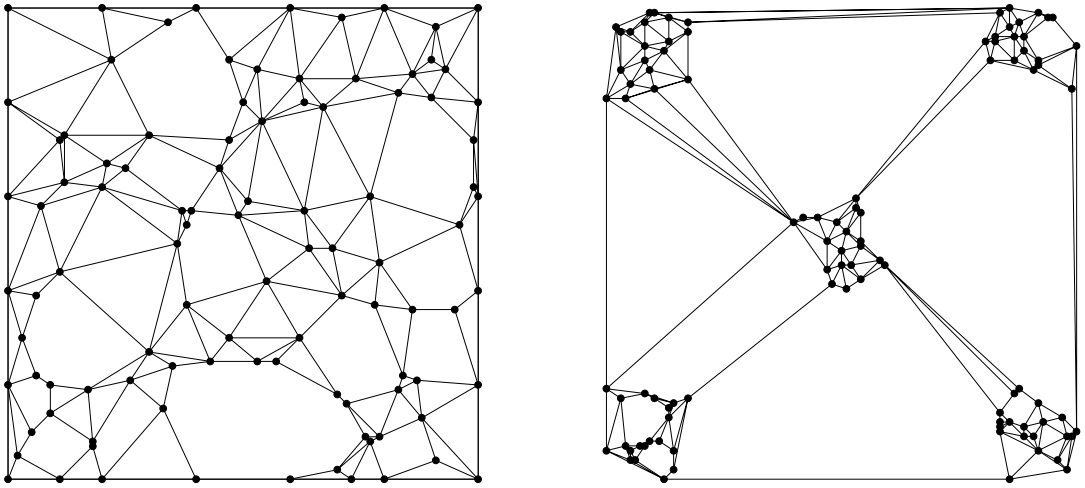
Comme à chaque fois qu'une arête est retirée de *candEdges*, des triangles sont aussi retirés de $candTris$, cela impliquera qu'en réexaminant après coup les arêtes de statut inconnu, on pourra, en repassant par la boucle de fin, trouver le statut de certaines d'entre elles. En itérant cette dernière boucle tant qu'une arête voit son statut changer, on parviendra à un squelette plus complet de la $MWT(\mathcal{S})$, avec une complexité en $O(n^6)$.

Une autre amélioration envisagée serait d'adapter le même genre de technique non plus aux arêtes mais aux triangles ou aux quadrilatères : les mêmes lemmes restent vrais à condition de définir correctement la notion de "minimalité locale". On aboutirait à un algorithme plus lent et beaucoup plus lourd, mais toujours polynomial.

Résultats

Effectivement, la programmation de l'algorithme et l'observation des triangulations données dans l'article montrent en général que, pour un ensemble de points quelconques (au moins pour ceux uniformément distribués), on obtient en moyenne près de 80% des arêtes de la triangulation de poids minimum ; on peut voir sur les exemples de la figure 6.8 que le nombre d'arêtes à ajouter pour obtenir la triangulation finale n'est en effet pas très élevé.

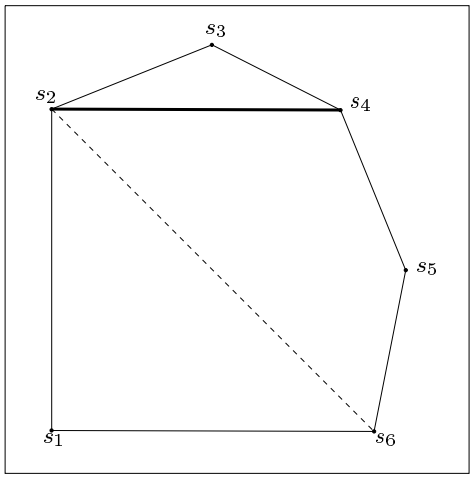
Figure 6.8 – des exemples pour des ensembles de 100 points : un uniformément distribué dans le carré unité, l'autre contenant 5 paquets de 20 points uniformément distribués.



Bien que les résultats paraissent satisfaisants, certaines questions restent en suspens ; parmi elles, une particulièrement intéressante est de savoir si le graphe des arêtes localement optimales construit en suivant la procédure de DICKERSON et MONTAGUE contient effectivement toutes les arêtes qui se trouvent dans toutes les triangulations localement optimales. AICHHOLZER (aidé par HAINZ) a récemment exhibé un exemple très simple où tel n'est pas le cas (figure 6.9). Soit $\mathcal{S}$ l'ensemble contenant les 6 points $s_1 = (0,0)$, $s_2 = (0,10)$, $s_3 = (5,12)$, $s_4 = (9,10)$, $s_5 = (11,5)$ et $s_6 = (10,0)$. Par une étude de cas, ou en regardant les 14 triangulations possibles de $\mathcal{S}$, on s'aperçoit que l'arête $[s_2 s_4]$ est dans toutes les triangulations localement optimales minimales. Par ailleurs, l'algorithme se contente d'exclure l'arête $[s_2 s_6]$ (plus longue arête intérieure) et n'inclut que celles qui sont sur l'enveloppe convexe. Pour toute autre arête intérieure a, il existe un quadrilatère convexe pour lequel a est la diagonale la plus courte, ce qui implique que a ne peut pas être ajoutée au sous-graphe. Finalement, $[s_2 s_4]$, bien qu'apparaissant dans toutes les triangulation localement minimales de $\mathcal{S}$, n'est pas incluse dans le sous-graphe.

Cet exemple montre bien qu'il serait peut-être possible d'étendre ce sous-graphe, afin qu'il contienne l'intersection de toutes les triangulations localement optimales. Ceci reste cependant un problème ouvert.

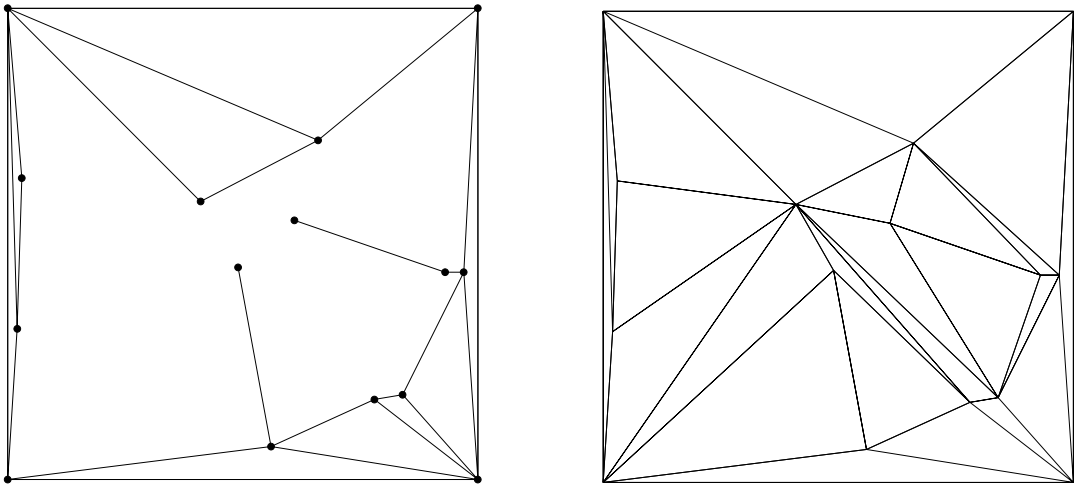
Figure 6.9 – les arêtes de l'enveloppe convexe sont incluses dans le sous-graphe ; l'arête en pointillés en est exclue, mais l'arête en trait épais reste indéterminée.



6.2.2 Application

Bien que l'algorithme présenté par DICKERSON et MONTAGUE semble particulièrement bien adapté à la recherche de la triangulation de poids minimum, il ne comporte aucune contrainte laissant supposer qu'il ne marcherait pas si on utilisait un autre critère d'optimalité. Il a donc semblé naturel d'en "écrire" une version faisant appel au critère d'optimisation de l'erreur L_2 pour le paraboloïde hyperbolique. Évidemment, cela n'a pas nécessité énormément de travail supplémentaire, puisqu'il a suffi de changer un simple appel à une procédure (par ailleurs déjà écrite).

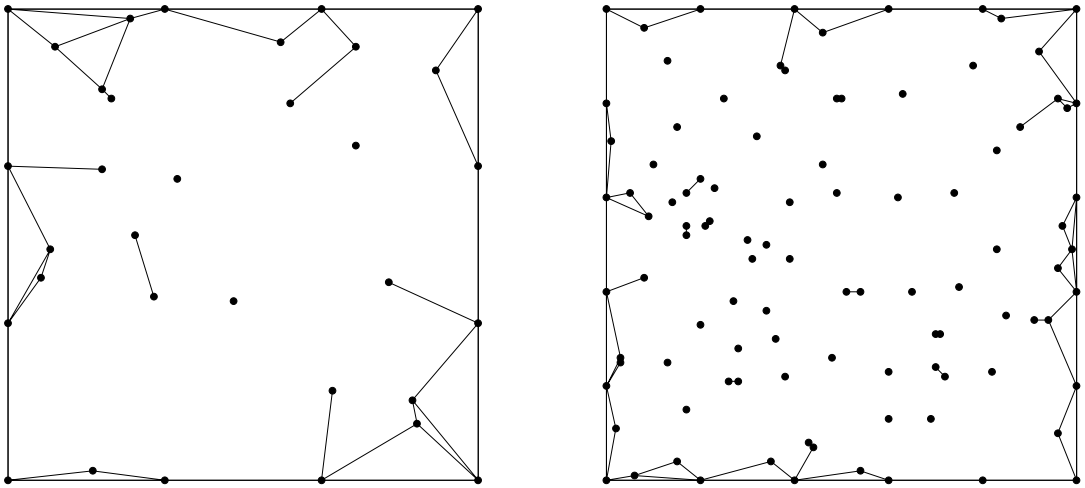
Figure 6.10 – un exemple de triangulation optimale de 15 points pour le critère de l'erreur L_2, résolu en utilisant l'algorithme de DICKERSON et MONTAGUE



Reste à voir si les résultats sont aussi probants. En fait, ça n'est pas du tout le cas : l'algorithme de recherche du sous-graphe s'arrête avec une grande partie des

arêtes en état toujours indéterminé, et pour un sous-graphe très mal connecté. Même s'il est possible de traiter en un temps raisonnable des ensembles de 20 points (la figure 6.10 montre un cas résolu pour un ensemble de 15 points), il est impossible d'aller au delà, comme le prouve la figure 6.11 où les sous-graphes obtenus pour des ensembles de 33 et 100 points restent de taille bien trop petite.

Figure 6.11 – impossible de résoudre des problèmes pour des ensembles comportant plus de 20 points.



Par ailleurs, étant donné que le sous-graphe obtenu n'est en général pas du tout connecté (voir la figure 6.11), pour pouvoir le compléter, il a dû être fait recours à un algorithme de backtracking, dont le principe est le suivant :

- **initialisations :**
  - soit $\mathcal{A}$ l'ensemble des arêtes de nature indéterminée à la fin de l'algorithme de DICKERSON et MONTAGUE.
  - soit T_{fixe} l'ensemble de tous les triangles présents dans le sous-graphe obtenu, $T_{indét.}$ l'ensemble de tous les triangles vides composés d'arêtes de $\mathcal{A}$ ou du sous-graphe.
  - on calcule l'erreur L_2 associée à chaque triangle.
  - on trie $T_{indét.}$ par ordre croissant des erreurs.
  - on calcule la table des intersections des triangles de $T_{indét.}$.
  - soit n le nombre nécessaire de triangles pour compléter le sous-graphe en triangulation.
  - soit m le nombre de triangles de $T_{indét.}$ déjà ajoutés à la solution en cours : $m \leftarrow 1$.
  - soit **T** le tableau à n éléments qui contiendra le complément du sous-graphe pour obtenir une triangulation : **T**[1] $\leftarrow$ le premier élément de $T_{indét.}$.

- **tant que** $m \neq 0$ **faire**
  si $m = n$ **alors**
  on a trouvé une nouvelle triangulation optimale.

on **élimine** de $T_{indét.}$ tous les triangles qui ont une erreur supérieure au nouveau minimum.
sinon
on **cherche** l'indice i du prochain triangle de $T_{indét.}$ compatible avec ceux présents dans **T**.
fin si
si (on n'a pas trouvé de triangle suivant) **ou** ($m = n$) **ou** (l'erreur obtenue est trop grande) **alors**
on **retourne en arrière** : $m \leftarrow m - 1$.
sinon
T[m] $\leftarrow$ i.
$m \leftarrow m + 1$.
fin si

fin tant que

Évidemment, le recours au backtracking implique qu'il ne sera pas possible de trouver la solution à un gros problème en temps raisonnable, d'autant plus que la taille du problème n'est pas donnée par le nombre de points de départ, mais par le nombre n de triangles qu'il reste à trouver ainsi que par le nombre total de triangles possibles $n_{tot.}$. En effet, le nombre théorique de solutions possibles vaut $\binom{n_{tot.}}{n}$.

Des heuristiques permettent cependant de ne pas tester toutes les solutions, et d'éliminer quelques triangles lors du déroulement du programme. Ainsi, comme les triangles sont rangés en ordre croissant d'erreur, si le m-ième triangle a pour indice i et que i est supérieur au nombre de triangles qu'il manque pour obtenir une triangulation, alors on sait que la solution en cours ne peut pas être valide, et on retourne en arrière. La même procédure doit être appliquée si, à un moment donné, l'erreur courante (somme des erreurs de tous les triangles de la solution testée) ajoutée au produit du nombre de triangles manquant par la plus petite erreur suivante est supérieure au minimum déjà trouvé.

Les résultats obtenus pour les tests de 100 ensembles de 15 points uniformément distribués sur le carré unité (dont 4 sont les coins du carré) sont contenus dans le tableau 6.6 ; chaque ligne indique une amélioration, en pourcentage :

- en passant de la triangulation de Delaunay à une triangulation localement optimale,
- en passant d'une triangulation localement optimale à la triangulation globalement optimale,
- en passant de la triangulation de Delaunay à la triangulation globalement optimale.

Le tableau 6.7 résume certaines statistiques concernant :

- le nombre d'arêtes possibles de l'ensemble de points.
- le nombre d'arêtes trouvées en utilisant l'algorithme de DICKERSON et MONTAGUE.

Tableau 6.6 – améliorations observées pour 100 tests d'ensembles comportant 15 points.

amélioration	moyenne	min.	max.
Del. → loc. opt.	3.155	0.006	19.949
loc. opt. → opt.	0.146	0.000	6.494
Del. → opt.	3.292	0.022	20.165

- le nombre de triangles vides de l'ensemble de points.
- le nombre de triangles trouvés en utilisant l'algorithme de DICKERSON et MONTAGUE.
- le nombre de triangles à ajouter pour obtenir une triangulation.
- le nombre de triangles vides qui restent à tester par le programme de backtracking.
- le nombre de solutions effectivement testées par le programme de backtracking.
- le nombre de retours nécessaires.

Tableau 6.7 – quelques statistiques observées pour 100 tests d'ensembles comportant 15 points.

dénomination	moyenne	min.	max.
arêtes init. poss.	104.44	102	105
arêtes Dick.	26.28	22	30
triangles init. poss.	209.18	186	252
triangles Dick.	11.34	8	15
triangles à trouver	12.50	9	16
triangles à tester	63.27	27	130
nb solutions testées	7619770.63	12037	137388187
nb retours effectués	637870.41	1556	9693274

On peut s'apercevoir qu'une bonne triangulation localement optimale donne, dans ces cas, un résultat très proche de l'optimal global. Mais, une telle conclusion ne pourrait sans doute pas être généralisée à des ensembles de points de taille plus importante. En effet, nous avons vu au chapitre précédent que les ensembles à 15 points limitaient fortement le nombre de triangulations possibles, et qu'il était ainsi "facile" de trouver l'optimum global. Le même phénomène n'était pas observé pour un nombre de points plus importants, où l'optimum local trouvé a alors très peu de chances d'être global.

Par ailleurs, les très fortes variations dans les nombres de solutions testées (provoqués par le bon ou mauvais comportement de l'algorithme de DICKERSON et MONTAGUE) laisse antrevoir ce qui va se passer si on essaie de lancer le programme pour 33 points ; pour l'ensemble de FRANKE, par exemple, il y a 56 triangles à trouver, parmi 1236 vides possibles (et 474 arêtes). Le sous-graphe obtenu contient 45 arêtes et 16 triangles : il reste alors 40 triangles à déterminer parmi 351 possibles...

Conclusion

L'étude des courbes de séparation et des triangulations localement optimales pour l'erreur L_2 de l'approximation linéaire du paraboloïde hyperbolique nous permet de répondre aux questions posées en introduction.

Nous nous demandions en effet s'il était possible de faire mieux que la triangulation de Delaunay. La réponse est affirmative dans le sens où un critère d'optimalité local, finalement très simple, a été trouvé, et cela pour toutes les fonctions quadratiques. De plus, au travers des diverses heuristiques testées, nous avons que des algorithmes construisent des triangulations localement optimales pour le paraboloïde hyperbolique unité - qui n'est pas une surface trop "irrégulière", dans la mesure où ses dérivés secondes en chacune des variables sont du même ordre - dont les erreurs sont en moyenne 20% plus petite que celles produites par la triangulation de Delaunay, amélioration pouvant monter jusqu'à plus de 50%, et cela pour des configurations de points tout à fait raisonnables.

Cependant, on ne peut pas conclure fermement par l'affirmative quant à la supériorité des triangulations localement optimales sur Delaunay, car, en réalité, cette amélioration moyenne de 20%, même si elle n'est pas négligeable, n'est pas très importante, et nous savons, de plus, qu'elle pourrait être meilleure, puisque les triangulations obtenues ne sont pas globalement optimales.

Concernant la facilité de trouver ces triangulations, les algorithmes à heuristiques, destinés aux critères locaux, sont évidemment très simples. Mais, le chapitre 6 nous prouve qu'il est sans doute inutile de vouloir rechercher, à tout prix, une triangulation globalement optimale, même pour l'approximation de fonctions aussi simples que les quadriques. Les algorithmes à mettre en place sont en effet beaucoup trop coûteux, en temps de calcul, et leur rendement, comparé à celui des algorithmes de triangulations localement optimales, assez faible (entre 4 et 5%).

Quant à l'utilité de ce travail, l'espoir de tirer des enseignements du cas d'école pour l'approximation de surfaces plus générales reste vain : il ne semble, pour le moment, pas possible d'envisager d'extension pour les courbes de séparation (c'est-à-dire, implicitement, pour la minimisation locale de l'erreur L_2 d'approximation) à une autre famille de fonctions que les quadriques. Delaunay demeure, encore une fois, une solution acceptable, même si l'on sait améliorer ses résultats de façon non négligeable. Ce mémoire définit un nouveau type de critère local, qui s'exprime très simplement et permet de prendre en compte les directions privilégiées des surfaces

auxquelles il s'applique.

Il semble raisonnable de penser qu'une application directe des courbes de séparation des fonctions quadratiques est envisageable : elles devraient permettre de construire une triangulation convenable d'une surface définie par une fonction f, à condition d'avoir quelques données concernant f. En effet, si on connait la matrice hessienne $\mathcal{H}$ associée à f, et si on suppose que f admet des dérivés partielles continues en chacun des points de données jusqu'à l'ordre 2, $\mathcal{H}$ définit une forme quadratique qu'il serait sans doute possible d'utiliser pour effectuer des échanges à partir d'une triangulation initiale (comme Delaunay).

Si on ne connait pas $\mathcal{H}$, il faut trouver une méthode qui, pour une arête ς échangeable donnée, choisit 3 autres sommets (on peut prendre au moins les 2 autres sommets du quadrilatère convexe qui admet ς pour diagonale) ; comme par 5 points ne passe qu'une seule quadrique, on saura toujours qu'elle sera l'équation de la courbe de séparation à appliquer pour l'arête ς. Un problème peut survenir : si la quadrique associée est une ellipse, par exemple, notre critère ne s'applique pas...

Une autre approche théorique (pour laquelle le temps a manqué) aurait peut-être pu donner des résultats, même s'ils avaient été totalement différents dans la mesure où l'on n'aurait alors certainement pas optimisé l'erreur L_2 d'approximation. Cette approche fait appel à la géométrie différentielle des surfaces : dans un premier temps, il faut être capable de déterminer, de manière exacte (formelle), la métrique propre du paraboloïde hyperbolique, grâce à l'obtention des géodésiques (courbes de la surface telle que la longueur de l'arc séparant 2 points d'une de ces courbes soit minimum parmi tous les arcs de la surface reliant les 2 points). Ceci est possible, notamment parce que l'on sait que, pour cette surface, il n'existe qu'une et une seule géodésique qui passe par 2 points donnés (c'est faux dans le cas du paraboloïde de révolution, par exemple).

Si la métrique μ donne une formule exploitable, on peut essayer de calculer un diagramme de Voronoï avec μ, pour un ensemble de points du plan, puis voir si le dual du diagramme est une triangulation ; auquel cas, il est possible que la triangulation spatiale associée rende une forme convenable.

Bibliographie

[AS94] P. K. Agarwal and S. Suri. Surface approximation and geometric partitions. In *Proc. 5th ACM-SIAM Sympos. Discrete Algorithms*, pages 24–33, 1994.

[Aur91] F. Aurenhammer. Voronoi diagrams: A survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23:345–405, 1991.

[BA76] J. Babuska and A. K. Aziz. On the angle condition in the finite element method. *SIAM J. Numer. Anal.*, 13:214–226, 1976.

[Bar77] R. E. Barnhill. Representation and approximation of surfaces. In J. R. Rice, editor, *Math. Software III*, pages 69–120. Academic Press, New York, NY, 1977.

[Bar85] R. E. Barnhill. Surfaces in computer aided geometric design: A survey with new results. *Computer Aided Geometric Design*, 2:1–17, 1985.

[BD95] M. De Berg and K. T. G. Dobrindt. On levels of details in terrains. Technical Report UU-CS-1995-12, Utrecht University, dept of Comp. Scien., 1995. URL=ftp://ftp.cs.ruu.nl/pub/RUU/CS/techreps/.

[BDE92] M. Bern, D. Dobkin, and D. Eppstein. Triangulating polygons without large angles. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 222–231, 1992.

[BDE95] M. Bern, D. Dobkin, and D. Eppstein. Triangulating polygons without large angles. *Internat. J. Comput. Geom. Appl.*, 5:171–192, 1995.

[BDH93] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The Quickhull algorithm for convex hull. Technical Report GCG53, Geometry Center, Univ. of Minnesota, July 1993.

[BE91] M. Bern and D. Eppstein. Polynomial-size nonobtuse triangulation of polygons. In *Proc. 7th Annu. ACM Sympos. Comput. Geom.*, pages 342–350, 1991.

[BE92] M. Bern and D. Eppstein. Mesh generation and optimal triangulation. In D.-Z. Du and F. K. Hwang, editors, *Computing in Euclidean Geometry*, volume 1 of *Lecture Notes Series on Computing*, pages 23–90. World Scientific, Singapore, 1992.

[BEE+93] M. Bern, H. Edelsbrunner, D. Eppstein, S. Mitchell, and T. S. Tan. Edge insertion for optimal triangulations. *Discrete Comput. Geom.*, 10(1):47–65, 1993.

[BEG90] M. Bern, D. Eppstein, and J. Gilbert. Provably good mesh generation. In *Proc. 31st Annu. IEEE Sympos. Found. Comput. Sci.*, pages 231–241, 1990.

[BFK84] W. Böhm, G. Farin, and J. Kahmann. A survey of curve and surface methods in CAGD. *Computer Aided Geometric Design*, 1:1–60, 1984.

[BGR88] B. S. Baker, E. Grosse, and C. S. Rafferty. Nonobtuse triangulation of polygons. *Discrete Comput. Geom.*, 3:147–168, 1988.

[BMR94] M. Bern, S. Mitchell, and J. Ruppert. Linear-size nonobtuse triangulation of polygons. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 221–230, 1994.

[Boi84] J.-D. Boissonnat. Geometric structures for three-dimensional shape representation. *ACM Trans. Graph.*, 3(4):266–286, 1984.

[Bro79] K. Q. Brown. Voronoi diagrams from convex hulls. *Inform. Process. Lett.*, 9:223–228, 1979.

[Bro91] J. L. Brown. Vertex based data dependent triangulations. *Comput. Aided Geom. Design*, 8:239–251, 1991.

[BSW83] R. E. Bank, A. H. Sherman, and A. Weiser. Refinement algorithms and data structures for regular local mesh refinement. In R. Stepleman, editor, *Scientific Computing*, pages 3–17. IMACS / North Holland, 1983.

[BT86] J.-D. Boissonnat and M. Teillaud. A hierarchical representation of objects: The Delaunay tree. In *Proc. 2nd Annu. ACM Sympos. Comput. Geom.*, pages 260–268, 1986.

[BT93] J.-D. Boissonnat and M. Teillaud. On the randomized construction of the Delaunay tree. *Theoret. Comput. Sci.*, 112:339–354, 1993.

[BY95] J.-D. Boissonnat and M. Yvinec. *Géométrie algorithmique*. Ediscience international, Paris, 1995.

[Byk76] A. Bykat. Automatic generation of triangular grid: I — subdivision of a general polygon into convex subregions; II — triangulation of convex polygons. *Internat. J. Numer. Methods Eng.*, 10:1329–1342, 1976.

[BZ70] J. Bramble and M. Zlamal. Triangular elements in the finite element method. *Math. Comp.*, 24:809–820, 1970.

[CGG+] B. W. Char, K. O. Geddes, G. H. Gonnet, B. L. Leong, M. B. Monagan, and S. M. Watt. *MAPLE: Reference Manual: 5th Edition*. University of Waterloo. A COMPLETER.

[Cha91] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete Comput. Geom.*, 6:485–524, 1991.

[Che89a] L. P. Chew. Constrained Delaunay triangulations. *Algorithmica*, 4:97–108, 1989.

[Che89b] L. P. Chew. Guaranteed-quality triangular meshes. Technical Report TR 89-983, Dept. Comput. Sci., Cornell Univ., Ithaca, NY, 1989.

[CSYL88] B. K. Choi, H. Y. Shin, Y. I. Yoon, and J. W. Lee. Triangulation of scattered data in 3d space. *Comput. Aided Design*, 20(5):239–248, June 1988.

[D'A91] E. F. D'Azevedo. Optimal triangular mesh generation by coordinate transformation. *SIAM J. Sci. Statist. Comput.*, 12(4):755–786, 1991.

[DD95] P. Desnoguès and O. Devillers. A locally optimal triangulation of the hyperbolic paraboloid. In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 49–54, 1995. URL=http://www.inria.fr/prisme/personnel/devillers/publis.html.

[Del34] B. Delaunay. Sur la sphère vide. A la memoire de Georges Voronoi. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskih i Estestvennyh Nauk*, 7:793–800, 1934.

[Dev92] O. Devillers. Robust and efficient implementation of the Delaunay tree. Report 1619, INRIA Sophia-Antipolis, Valbonne, France, 1992.

[DFNP84] L. De Floriani, B. Falcidieno, G. Nagy, and C. Pienovi. Hierarchical structure for surface approximation. *Comput. Graph. (UK)*, 8(2):183–193, 1984.

[DFP85] L. De Floriani, B. Falcidieno, and C. Pienovi. Delaunay-based representation of surfaces defined over arbitrarily shaped domains. *Comput. Vision Graph. Image Process.*, 32:127–140, 1985.

[Die81] P. Dierckx. An algorithm for surface-fitting with spline functions. *IMA J. Num. Anal.*, 2:267–283, 1981.

[DLR90a] N. Dyn, D. Levin, and S. Rippa. Algorithms for the construction of data dependent triangulations. In J. C. Mason and M. G. Cox, editors, *Algorithms for Approximation II*, pages 185–192. Chapman and Hall, London, 1990.

[DLR90b] N. Dyn, D. Levin, and S. Rippa. Data dependent triangulations for piecewise linear interpolation. *IMA Journal of Numerical Analysis*, 10:137–154, 1990.

[DM96] M. T. Dickerson and M. H. Montague. A (usually?) connected subgraph of the minimum weight triangulation. In *Proc. 12th Annu. ACM Sympos. on Comput. Geom.*, pages 204–213, 1996.

[DMP94] L. De Floriani, P. Marzano, and E. Puppo. Hierarchical terrain models: Survey and formalization. In *ACM Sympos. on Applied Comput.*, 1994.

[DP89] L. De Floriani and E. Puppo. A survey of constrained Delaunay triangulation algorithms for surface representaion. In G. G. Pieroni, editor, *Issues on Machine Vision*, pages 95–104. Springer-Verlag, New York, NY, 1989.

[DS89] E. F. D'Azevedo and R. B. Simpson. On optimal interpolation triangle incidences. *SIAM J. Sci. Statist. Comput.*, 10(6):1063–1075, 1989.

[Dwy87] R. A. Dwyer. A faster divide-and-conquer algorithm for constructing Delaunay triangulations. *Algorithmica*, 2:137–151, 1987.

[Dwy91] R. A. Dwyer. Higher-dimensional Voronoi diagrams in linear expected time. *Discrete Comput. Geom.*, 6:343–367, 1991.

[Ede87] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, Heidelberg, West Germany, 1987.

[ES86] H. Edelsbrunner and R. Seidel. Voronoi diagrams and arrangements. *Discrete Comput. Geom.*, 1:25–44, 1986.

[ETW92] H. Edelsbrunner, T. S. Tan, and R. Waupotitsch. $O(n^2 \log n)$ time algorithm for the minmax angle triangulation. *SIAM J. Sci. Statist. Comput.*, 13(4):994–1008, July 1992.

[Far92] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design*. Academic Press, San Diego, CA, 3rd edition, 1992.

[FBF77] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.*, 3:209–226, 1977.

[FF91] W. H. Frey and D. A. Field. Mesh relaxation. A new technique for improving triangulations. *Internat. J. Numer. Methods Eng.*, 31(6):1121–1133, May 1991.

[FF92] W. H. Frey and D. A. Field. Mesh relaxation for improving triangulations. In *Proceedings of the SIAM Regional Conference on Geometric Aspects of Industrial Design*, pages 11–24, Philadelphia, PA, 1992. Soc. for Industrial & Applied Mathematics.

[Fie88] D. A. Field. Laplacian smoothing and Delaunay triangulations. *Communications in Applied Numerical Methods*, 4(6):709–712, November 1988.

[Flo89] L. De Floriani. A pyramidal data structure for triangle-based surface description. *IEEE Comput. Graph. Appl.*, 9(2):67–78, March 1989.

[FN90] R. Franke and G. M. Nielson. Scattered data interpolation and applications: A tutorial and survey. In H. Hagen and D. Roller, editors, *Geometric Modeling: Methods and Applications*, pages 131–160, 1990.

[For87] S. J. Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

[For92] S. Fortune. Voronoi diagrams and Delaunay triangulations. In D.-Z. Du and F. K. Hwang, editors, *Computing in Euclidean Geometry*, volume 1 of *Lecture Notes Series on Computing*, pages 193–233. World Scientific, Singapore, 1992.

[FP92] L. De Floriani and E. Puppo. A hierarchical triangle-based model for terrain description. In A. U. Frank et al., editor, *Theories and Methods os Spatio-Temporal Reasoning in Geographic Space*, pages 236–251, Berlin, 1992. Springer-Verlag.

[Fra79] R. Franke. A critical comparison of some methods for interpolation of scattered data. Report NPS-53-79-003, Naval Postgraduate School, Monterey, CA, 1979.

[Fra82] R. Franke. Scattered data interpolation: Tests of some methods. *Mathematics of Computation*, 38(157):181–200, january 1982.

[Fra87] R. Franke. Recent advances in the approximation of surfaces from scattered data. In C. K. Chui, L. L. Schumaker, and F. I. Utreras, editors, *Topics in Multivariate Approximation*, pages 79–98. Academic Press, 1987.

[Fre87] W. H. Frey. Selective refinement: A new strategy for automatic node placement in graded triangular meshes. *Intern. Journal for Numer. Methods in Engineering*, 24:2183–2200, 1987.

[FS87] R. Franke and L. L. Schumaker. A bibliography of multivariate approximation. In C. K. Chui, L. L. Schumaker, and F. I. Utreras, editors, *Topics in Multivariate Approximation*, pages 275–335. Academic Press, 1987.

[Gei93] B. Geiger. *Three-dimensional modeling of human organs and its application to diagnosis and surgical planning.* thèse de doctorat en sciences, École Nationale Supérieure des Mines de Paris, France, 1993.

[Geo91] P. L. George. *Génération Automatique de Maillage - Applications aux Méthodes d'Éléments Finis.* Masson, 1991.

[GH95] M. Garland and P. S. Heckbert. Fast polygonal approximation of terrain and height fields. Technical Report CMU-CS-95-181, School of Comp. Sc., Carnegie Mellon Univ., 1995. URL=http://www.cs.cmu.edu/~garland/scape.

[Gil79] P. D. Gilbert. New results in planar triangulations. Report R-850, Coordinated Sci. Lab., Univ. Illinois, Urbana, IL, 1979.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York, NY, 1979.

[GKS92] L. J. Guibas, D. E. Knuth, and M. Sharir. Randomized incremental construction of Delaunay and Voronoi diagrams. *Algorithmica*, 7:381–413, 1992.

[Gre75] J. A. Gregory. Error bounds for linear interpolation in triangles. In Academic Press, editor, *The Mathematics of Finite Elements and Application*, pages 163–170. J. R. Whiteman, London, 1975.

[GS69] K. R. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.

[GS85] L. J. Guibas and J. Stolfi. Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. *ACM Trans. Graph.*, 4:74–123, 1985.

[GS92] P. L. George and E. Séveno. Génération de maillages par une méthode de type frontal. Technical Report 1725, INRIA, 1992.

[Ham94] B. Hamann. A data reduction scheme for triangulated surfaces. *Comput. Aided Geom. Design*, 11:197–214, 1994.

[Har71] R. L. Hardy. Multiquadric equations of topography and other irregular surfaces. *J. Geophys. Res.*, 76:1905–1915, 1971.

[HDD+92] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Comput. Graphics*, 26(2):71–78, 1992. Proc. SIGGRAPH '92.

[Her76] L. R. Herrmann. Laplacian-isoparametric grid generation scheme. *Journal of Eng. Mech. Div.*, 102, 1976.

[HG94] P. S. Heckbert and M. Garland. Multiresolution modeling for fast rendering. In *Proc. Graphics Interface '94*, pages 43–50. Canadian Inf. Proc. Soc., 1994.

[HGar] P. S. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms. Technical report, CS Dept., Carnegie Mellon University, to appear. URL=http://www.cs.cmu.edu/~ph.

[HH93] P. Hinker and C. Hansen. Geometric optimization. In *Proc. Visualization '93*, pages 189–195, San Jose, CA, 1993.

[HL88] K. Ho-Le. Finite element mesh generation methods: A review and classification. *Comput. Aided Design*, 20(1):27–38, January 1988.

[HN96] F. Hurtado and M. Noy. The graph of triangulations of a convex polygon. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages C7–C8, 1996.

[HNU96] F. Hurtado, M. Noy, and J. Urrutia. Flipping edges in triangulations. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 214–223, 1996.

[JW91] Norman L. Jones and Stephen G. Wright. Algorithm for smoothing triangulated surfaces. *Journal of Computing in Civil Engineering*, 5(1):85–102, January 1991.

[JW93] B. Joe and C. A. Wang. Duality of constrained voronoi diagrams and delaunay triangulations. *Algorithmica*, 9:142–155, 1993.

[KGV83] Kirkpatrick, Gelatt, and Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[Kir79] D. G. Kirkpatrick. Efficient computation of continuous skeletons. In *Proc. 20th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 18–27, 1979.

[Kir80] D. G. Kirkpatrick. A note on Delaunay and optimal triangulations. *Inform. Process. Lett.*, 10:127–128, 1980.

[KK88] J. Katajainen and M. Koppinen. Constructing Delaunay triangulations by merging buckets in quadtree order. *Fundamenta Informaticae*, 11(3):275–288, 1988.

[Kli80] G. T. Klincsek. Minimal triangulations of polygonal domains. *Discrete Math.*, 9:121–123, 1980.

[KT94] A. D. Kalvin and R. H. Taylor. Superfaces: Polyhedral approximation with bounded error. *Medical Imaging: Image Capture, Formatting and Display*, 2164:2–13, 1994.

[Lam94] T. Lambert. *Empty-Shape Triangulation Algorithms*. Ph.D. thesis, Dept. Comput. Sci., Univ. of Manitoba, Winnipeg, MB, August 1994.

[Law77] C. L. Lawson. Software for C^1 surface interpolation. In J. R. Rice, editor, *Math. Software III*, pages 161–194, New York, NY, 1977. Academic Press.

[Lee89] J. Lee. A drop heuristic conversion method for extracting irregular net work for digital elevation models. In *GIS/LIS '89 Proc.*, volume 1, pages 30–39. American Congress on Surveying and Mapping, November 1989.

[LFA77] J. Lelong-Ferrand and J.M. Arnaudiès. *Cours de Mathématiques - Tome 3 - Géométrie et Cinématique*. Dunod Université, Paris, 2ème edition, 1977.

[Lit83] F. F. Little. Convex combination surfaces. In *Surfaces in CAGD*, pages 99–108, Amsterdam, 1983. North Holland.

[LL90] C. Levcopoulos and A. Lingas. Fast algorithms for greedy triangulation. In *Proc. 2nd Scand. Workshop Algorithm Theory*, volume 447 of *Lecture Notes in Computer Science*, pages 238–250. Springer-Verlag, 1990.

[Llo77] E. L. Lloyd. On triangulations of a set of points in the plane. In *Proc. 18th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 228–240, 1977.

[Mav90] D. J. Mavripilis. Adaptive mesh generation for viscous flows using delaunay triangulation. *J. of Comp. Physics*, 2:271–291, 1990.

[Mel93] E. A. Melissaratos. l_p optimal d dimensional triangulations for piecewise linear interpolation: A new result on data dependent triangulations. Technical Report RUU-CS-93-13, Dept. Comput. Science, Utrecht Univ., Utrecht, NL, 1993.

[MS92] E. A. Melissaratos and D. L. Souvaine. Coping with inconsistencies: a new approach to produce quality triangulations of polygonal domains with holes. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 202–211, 1992.

[MZ79] G. K. Manacher and A. L. Zobrist. Neither the greedy nor the Delaunay triangulation of a planar point set approximates the optimal triangulation. *Inform. Process. Lett.*, 9:31–34, 1979.

[Nad85] E. J. Nadler. *Piecewise Linear Approximation on Triangulations of a Planar Region.* PhD thesis, Brown University, 1985.

[Nie93] G. M. Nielson. Scattered data modeling. *IEEE Computer Graphics & Applications*, pages 60–70, jan 1993.

[Nie96] F. Nielsen. *Algorithmes Géométriques Adaptatifs.* PhD thesis, Ecole Normale Supérieure de Lyon, 1996.

[Nul95] S. Nullans. Reconstruction de coupes incomplètes. Rapport de DEA, université Louis Pasteur, Strasbourg, 1995.

[OBS92] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams.* John Wiley & Sons, Chichester, England, 1992.

[OIM84] T. Ohya, M. Iri, and K. Murota. Improvements of the incremental method for the Voronoi diagram with computational comparison of various algorithms. *J. Oper. Res. Soc. Japan*, 27(4):306–336, 1984.

[O'R81] J. O'Rourke. Polyhedra of minimal area as 3-d object models. In *Proc. 7th Internat. Joint Conf. Artif. Intell.*, pages 664–666, 1981.

[PH77] F. P. Preparata and S. J. Hong. Convex hulls of finite sets of points in two and three dimensions. *Commun. ACM*, 20:87–93, 1977.

[PK93] M. F. Polis and D. M. Mc Keown. Iterative tin generation to support large scale simulations. In *Proc. of Auto-Carlo 11*, 1993. URL=http://www.cs.cmu.edu/~MAPSLab.

[PM92] M.-A. K. Posenau and D. M. Mount. Delaunay triangulations and computational fluid dynamics meshes. In *Proc. 4th Canad. Conf. Comput. Geom.*, pages 316–321, 1992.

[Pos93] M.-A. K. Posenau. Approaches to high aspect ratio triangulations. In *Proc. 5th Canad. Conf. Comput. Geom.*, pages 30–35, 1993.

[Pow92] P. L. Powar. Minimal roughness property of the delaunay triangulation. *Computer Aided Geometric Design*, 9:491–494, 1992.

[PS85] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, NY, 1985.

[RI95] M.-C. Rivara and P. Inostroza. A discussion on mixed (longest-side midpoint insertion) delaunay techniques for the triangulation refinement problem. In *Proc. 4th International Meshing Roundtable*, pages 335–346, Albuquerque, NM, 1995. Sandia National Laboratories.

[Rip90] S. Rippa. Minimal roughness property of the Delaunay triangulation. *Comput. Aided Geom. Design*, 7:489–497, 1990.

[Rip92a] S. Rippa. Adaptive approximation by piecewise linear polynomials on triangulations of subsets of scattered data. *SIAM Journal on Scien. Stat. Comput.*, 13(5):1123–1141, 1992.

[Rip92b] S. Rippa. Long and thin triangles can be good for linear interpolation. *SIAM J. Numer. Anal.*, 29(1):257–270, February 1992.

[Riv84a] M.-C. Rivara. Algorithms for refining triangular grids suitable for adaptive and multigrid techniques. *Internat. J. Numer. Methods Eng.*, 20:745–756, 1984.

[Riv84b] M.-C. Rivara. Design and data structure of fully adaptive, multigrid, finite-element software. *ACM Transactions on Mathematical Software*, 10(3):242–264, 1984.

[Riv87] M.-C. Rivara. A grid generator based on 4-triangles conforming mesh-refinement algorithms. *Internat. J. Numer. Methods Eng.*, 24(7):1343–1354, 1987.

[Riv89] M.-C. Rivara. Selective refinement/derefinement algorithms for sequences of nested triangulations. *Intern. Journal for Numer. Methods in Engineering*, 28:2289–2906, 1989.

[Riv91] M.-C. Rivara. Local modification of meshes for adaptive and/or multigrid finite element methods. *Journal of Comput. and Applied Mathem.*, 36:79–89, 1991.

[Riv93] M.-C. Rivara. A discussion on the triangulation refinement problem. In *Proc. 5th Canad. Conf. Comput. Geom.*, pages 42–47, 1993.

[RL92] M.-C. Rivara and C. Levin. A 3-d refinement algorithm suitable for adaptive and multigrid techniques. *Comm. in Applied Numer. Methods*, 8:281–290, 1992.

[RS75] I. G. Rosenberg and F. Stenger. A lower bound on the angles of triangles constructed by bisecting the longest side. *Math. Comp.*, 29:390–395, 1975.

[Sam90] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, Reading, MA, 1990.

[Sch93] L. L. Schumaker. Computing optimal triangulations using simulated annealing. *Comput. Aided Geom. Design*, 10(3–4):329–345, August 1993.

[SD95] P. Su and R. L. S. Drysdale. A comparison of sequential Delaunay triangulation algorithms. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 61–70, 1995.

[She68] D. Shephard. A two-dimensional interpolation function for irregularly spaced data. In *23rd Nat. Conf. ACM*, pages 517–524, 1968.

[Sib78] R. Sibson. Locally equiangular triangulations. *Comput. J.*, 21:243–245, 1978.

[Sty80] M. Stynes. On faster convergence of the bisection method for all triangles. *Math. Comp.*, 35:1195–1201, 1980.

[Su94] P. Su. *Efficient Parallel Algorithms for Closest Point Problems*. PhD thesis, Dept. of Comput. Sci., Dartmouth College, Hanover, NH, November 1994.

[SZL92] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen. Decimation of triangle meshes. *Comput. Graph.*, 26(2):65–70, 1992. Proc. SIGGRAPH '92.

[Tal87] J. Y. Talon. Algorithmes de génération et d'amélioration de maillages en 2d. Technical Report 20, Artemis-Imag, 1987.

[Tal89] J. Y. Talon. *Génération et Amélioration de Maillages pour Eléments Finis en Deux et Trois Dimensions*. PhD thesis, Institut Polytechnique de Grenoble, 1989.

[Tou80] G. T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recogn.*, 12:261–268, 1980.

[VAB+95] A. Varshney, P. K. Agarwal, F. P. Brooks, Jr., W. Wright, and H. Weber. Generating levels of details for large-scale polygonal models. Technical Report CS-1995-20, Dept. of C.S., Duke University, 1995. URL=http://www.cs.duke.edu/department.html#techrept.

[Var94] A. Varshney. *Hierarchical Geometric Approximations*. Ph.D. thesis, Department of Computer Science, University of North Carolina, Chapel Hill, NC 27599-3175, 1994. TR-050-1994.

[Wat81] D. F. Watson. Computing the n-dimensional Delaunay tesselation with applications to Voronoi polytopes. *Comput. J.*, 24(2):167–172, 1981.

[Wör83] B. Wördenweber. Surface triangulation for picture production. *IEEE Comput. Graph. Appl.*, pages 45–51, November 1983.

[Wör84] B. Wördenweber. Finite element mesh generation. *Computer Aided Design*, 16(5):285–291, 1984.

[WP84] D. F. Watson and C. M. Philip. Survey: Systematic triangulations. *Comput. Vision Graph. Image Process.*, 26:217–223, 1984.

[ZT91] O. C. Zienkiewicz and R. L. Taylor. *The Finite element method : 1 : basic formulation and linear problems*. McGraw-Hill, London, New York, Paris, 4th edition, 1991.

Index des auteurs

Index par mots clefs

Triangulations et quadriques

Soit $\mathcal{S}$ un ensemble de points pris sur une surface $\mathcal{F}$ d'équation $z = f(x, y)$; on projette $\mathcal{S}$ dans le plan (xOy), et on désire construire une triangulation de l'enveloppe convexe de la projection de $\mathcal{S}$ qui déterminera une approximation linéaire par morceaux de $\mathcal{F}$, dont la qualité sera liée à une mesure de l'erreur d'approximation de la surface. Il a été récemment prouvé que la triangulation de Delaunay était optimale pour des critères de normes L_p, lorsqu'il s'agissait d'approcher linéairement toute fonction quadratique convexe, dans un espace de dimension quelconque.
En revanche, très peu de recherches ont été menées lorsque la surface n'est pas convexe. Ce mémoire propose donc d'étudier l'approximation par une triangulation, pour des critères de normes L_1 et L_2, d'une surface non convexe d'équation la plus simple possible : le paraboloïde hyperbolique défini par $z = x^2 - y^2$. Une construction est ainsi donnée pour déterminer, de manière naturelle, les courbes de séparation d'un triangle Δ, c'est-à-dire les limites du plan pour lesquelles Δ doit être conservé dans une triangulation localement optimale du paraboloïde hyperbolique. Des algorithmes de triangulation qui font appel à diverses heuristiques fondées sur les courbes de séparation ont été abondamment testés ; une amélioration significative par rapport à la triangulation de Delaunay a été mise en évidence. Une comparaison avec des triangulations globalement optimales, dont l'obtention n'est possible qu'au moyen de programmes de complexité exponentielle, prouve que ces algorithmes rendent finalement de "bonnes" triangulations. Les recherches montrent qu'un tel procédé peut facilement être généralisé à toutes les surfaces définies par des fonctions quadratiques, de la forme $z = \alpha x^2 + \beta y^2 + \gamma xy + \delta_1 x + \delta_2 y + \delta_3$.

Mots-clés : géométrie algorithmique, approximations de surface, triangulations, fonctions quadratiques, optimalité.

Triangulations and quadrics

Given a set $\mathcal{S}$ of data points on a surface $\mathcal{F}$ whose equation is $z = f(x, y)$, we would like to triangulate the convex hull of the projection of $\mathcal{F}$ on the xy-plane. This triangulation determines a linear approximation of $\mathcal{F}$ whose quality is given by a measure of the approximation error. It has been recently proved that the Delaunay triangulation is optimal with respect to L_p-norm criteria, when used for approximating convex quadratic functions.
But, little research has been carried out for non convex surfaces. This work studies the approximation, with respect to L_1- and L_2-norms, of a non convex surface by using a triangulation. We consider a simple case: the hyperbolic paraboloid $z = x^2 - y^2$. A construction is given for finding the separation curves of a triangle Δ, the curves limiting the planar zones where Δ will be kept in a locally optimal triangulation of the hyperbolic paraboloid. Triangulation algorithms that use several heuristics based on the separation curves are amply tested and are shown to be better than the Delaunay triangulation. A comparison with globally optimal triangulations which are obtained by means of exponential programs shows that these algorithms finally give "good" triangulations. Our research proves that such a process can be easily extended to general quadratic functions $z = \alpha x^2 + \beta y^2 + \gamma xy + \delta_1 x + \delta_2 y + \delta_3$.

Keywords: computational geometry, surface approximations, triangulations, quadratic functions, optimality.